meddbase®
medical management systems

meddbase®
medical management systems

API

v. 1.0.104

Registered In England.
Company Number 5272668.
Registered Office: 140 Buckingham Palace Road, London SW1W 9SA

Contents

Version	Changes
1.0.40	Membership scheme - All methods may be used also for logged-in patient - A new method GetCurrent returns the membership scheme the patient is joined into. - A new method Join allows the logged-in patient to join to the membership scheme. Authentication - [DEPRECATED] SubmitValidationEmail and [DEPRECATED] SubmitValidationSMS returns an object ActivationConfirmation that includes information about activated account (e.g. *OutstandingInvoice* – an invoice that needs to be paid, usually the membership fee). Authentication - A new method GetPayerTypes returns a list of payer types that can be nominated to pay for services. - Methods GetAppointmentTypes, GetSites, GetClinicians and GetProposedAppointments takes an optional parameter *payerType*. This is used to get appointment types, sites, clinicians and proposed appointment for a specific payer type.

1.0.41	Authentication - GetTermsAndConditions method added - ConfigData includes two more properties: *ForceContactOptions* and *ForceTermsAndConditions*. - NotificationData includes new type *Action Required*. The *ObjectKey* property specifies the action (*SetContactOptions* or *AcceptTermsAndConditions*)

1 Address Finder

This section provides services for searching and retrieving address details. Currently, only the UK is supported.

1.1 Search

Returns a number of possible shallow search address results for the postcode.

JavaScript library method

patientportal.addressFinder.search ({postcode: <postcode>});

HTTP Method

GET

Url

/patientportalapi/address-finder/search

URL Parameters

postcode	string	Case-insensitive postcode with space (e.g. N7 0NH or n7 0nh).

Returned JSON

Shallow Search Result Data[]

1.2 Address

Returns address details for the provided encrypted key.

JavaScript library method

patientportal.addressFinder.address ({key: <key>});

HTTP Method

GET

Url

/patientportalapi/address-finder/address

URL Parameters

key	string	An encrypted key value containing partial address information.

	Returned JSON AddressData Patient - AcceptTermsAndConditions method added - PersonDemographicData includes the *ContactOptions* property which includes a set of permission the patient has given to the company to contact them. See also PatientContactOption.

1.0.42

2 Address Finder

This section provides services for searching and retrieving address details. Currently, only the UK is supported.

2.1 Search

Returns a number of possible shallow search address results for the postcode.

JavaScript library method

patientportal.addressFinder.search ({postcode: <postcode>});

HTTP Method

GET

Url

/patientportalapi/address-finder/search

URL Parameters

postcode	string	Case-insensitive postcode with space (e.g. N7 0NH or n7 0nh).

Returned JSON

Shallow Search Result Data[]

2.2 Address

Returns address details for the provided encrypted key.

JavaScript library method

patientportal.addressFinder.address ({key: <key>});

HTTP Method

GET

Url

/patientportalapi/address-finder/address

URL Parameters

key	string	An encrypted key value containing partial address information.

Returned JSON

AddressData

Patient

PersonDemographicData – new properties: *Key, EmployeeNumber* and *EmployeeDepartment* added.

	Authentication - GetConfig method added Authentication - AppointmentTypeData – new properties *CanBookAppointment* and *CanReferPatient* added New sections Referral Portal API, Users, Patients and Referrals added. New objects User, UserData, ReferralOverviewData, ReferralData and DocumentData added.`
1.0.43	Users - GetUsers – Paging/sorting support added. - ResetPassword added
1.0.44	New section Patient Medical Report Review added.
1.0.45	Referrals - New method GetBasicStatistics added. Notifications - NotificationData includes new type *Referral notification*. The *ObjectKey* property specifies the referral key.
1.0.46	ConfigData – *UserType* added
1.0.47	Support for divisions: - DivisionData added - DepartmentData – new property *Divisions* added - GetUsers , GetReferrals– new filter *division* added Referrals - BulkReallocate added Users - FindUserToReallocate does not need parameter *referral* anymore. Patients - GetPatients added to get list of patients on the referral portal
1.0.48	Patients - GetPatient added to get one specific patient
1.0.49	Authentication - GetDepartments added to get a list of departments/divisions for the registration process on the referral portal Authentication - SaveAndSubmitQuestionnaire added to save answers and submit the questionnaire at once

1.0.50	Patient Medical Report Review - Section renamed. Previous name was Medical Report Review - The javascript section renamed from *medicalReportReview* to *patientReportReview* Manager Medical Report Review section added
1.0.51	Authentication - *is-oh* parameter added within almost all methods PersonDemographicData - *WorkEmailAddress* added. The work email address is used for the referral portal as a login, contact email, email for notifications.
1.0.52	Authentication and Users - GetDepartments method is deprecated - GetDepartmentsAndDivisions and GetDepartmentsAndDivisions added
1.0.53	ReferralOverviewData - *FollowUp* renamed to *IsFollowUp* - *HasFollowUp*, *PreviousReferralKey*, *NextReferralKey* added
1.0.54	Patients - UpdatePatient added
1.0.55	Patients - QuickSearch, GetPatients – *work-email* and *personal-email* parameters added. The previous *email* has been removed.
1.0.56	AppointmentData - Telemedicine, TelemedicineConnection – *Telemedicine and TelemedicineConnection details added* AppointmentTypeData - TelemedicineOption defining if the appointment type is booked as telemedicine

1.0.57	New features: - We have added a new section called Recalls. This section provides an ability to manage patient’s recalls. The booking system is extended and you can book an appointment for a recall. - New section Documents has been added. You can get a list of patient documents or attach a new patient document. - You can now register an employee status against your employee.

3 Address Finder

This section provides services for searching and retrieving address details. Currently, only the UK is supported.

3.1 Search

Returns a number of possible shallow search address results for the postcode.

JavaScript library method

patientportal.addressFinder.search ({postcode: <postcode>});

HTTP Method

GET

Url

/patientportalapi/address-finder/search

URL Parameters

postcode	string	Case-insensitive postcode with space (e.g. N7 0NH or n7 0nh).

Returned JSON

Shallow Search Result Data[]

3.2 Address

Returns address details for the provided encrypted key.

JavaScript library method

patientportal.addressFinder.address ({key: <key>});

HTTP Method

GET

Url

/patientportalapi/address-finder/address

URL Parameters

<table>
<tr><td></td><td><table><tr><td>key</td><td>string</td><td>An encrypted key value containing partial address information.</td></tr></table>
Returned JSON
AddressData
Patient
- Methods GetDepartmentsAndDivisions and GetEmployeeStatuses added
- PersonDemographicData – property *EmployeeStatus* added.
Appointments
- A new optional input parameter recall added within methods: GetPayerTypes, GetAppointmentTypes, GetSites, GetClinicians, GetProposedAppointments, BookProposedAppointment
- GetAppointments added. Use this method to get a list of appointments for a specific patient.
- GetQuestionnaires added. Use this method to get a list of questionnaires for a specific patient.
- AppointmentData – property *Patient* added.
Users
- GetEmployeeStatuses added</td></tr>
<tr><td>1.0.58</td><td>Users
- UserType is obsolete. The API will support it for a while but it is removed from documentation.
- GetUserRights and UserRights added to manage user rights.
- UserData – *Rights* and *AllPatientsVisible* properties added</td></tr>
<tr><td>1.0.59</td><td>New features:
- New section Questionnaires provides methods to work with explicitly requested patient questionnaires.
- New section Questionnaire Request provides methods to allow a patient to complete a questionnaire using Client authentication.
Referrals
- StartFromQuestionnaireRequest added.</td></tr>
<tr><td>1.0.60</td><td>Finance
- Removed GetCurrentPaymentState (Not required anymore.)
- Modified SendInvoiceByEmail to allow payments through email
- ProvidePayment modified to match new payment service
- PayerAccounts added to match new payment service
- RemovePayerAccount to remove an existing card</td></tr>
<tr><td>1.0.61</td><td>[DEPRECATED] SendValidationEmail, [DEPRECATED] SendPasswordResetEmail – portal URL format fixed</td></tr>
<tr><td>1.0.62</td><td>ReferralData now includes *PatientReview* property which provides details about the patient review.</td></tr>
<tr><td>1.0.63</td><td>ProvidePayment on successful validation and loading of iframe, returns a new callback. This new callback promises the result of the actual payment.</td></tr>
</table>

1.0.64	InvoiceData A new "PayableOnline" boolean parameter indicating if an online payment is possible against an invoice.
1.0.65	Membership Scheme - RequiresOnlinePayment and OnlinePaymentAllowed parameters added to MembershipSchemeData - Changed return type of GetCurrent from MembershipSchemeData to CurrentMembershipSchemeData - SetupOnlinePayment method to set up recurring payments for patient membership. - RemoveOnlinePayment method to remove any current online recurring payment method associated with patient's current membership.
1.0.66	Login – *username* and *password* are included in the POST rather than GET parameters Company authentication section removed. The *Company-name* parameter is not needed anymore.
1.0.67	CSFR protection implemented - URL format updated, a *token* parameter added - Authentication - Authentication Token method added
1.0.69	Membership portal - CurrentMembershipData added LastInvoiceBalance - Notification added membership notification type
1.0.70	UserData – `AllNonAssignedPatientsVisible` property added
1.0.71	Telemedicine section added to allow authorisation for a telemedicine access
1.0.72	*Token* value removed from ValidateLogin and GetConfig. *GetToken* method removed.
1.0.73	Anonymous Appointment Search section added. Added missing 'modules' POST parameter to Appointment - GetSites, Appointment - GetClinicians
1.0.74	Added functionality to retrieve services except modules while booking appointment. - Parameter CanAddServices to AppointmentTypeData - Functions to retrieve ServiceTypes and Services - Anonymous appointment functions to retrieve Service Types and Services - Services parameter to getProposedAppointments
1.0.75	Added CurrencySymbol to following: - AppointmentModuleData - ServiceData - InvoiceData - InvoiceItemData - InvoiceCreditNoteData - MembershipSchemeData - CurrentMembershipSchemeData

1.0.76	Minor changes to allow multiple sage pay accounts per chamber / creditor - Payer-account takes an extra optional 'invoiceKey' parameter - PayerAccountData has paytype. - CurrentMembershipSchemeData has last invoice key
1.0.77	Single sign on features - Direct login - Single sign on login details - SSOStatus in config data
1.0.78	Pathways section added
1.0.79	Absence Management section added
1.0.80	Added the RTW Questionnaire API methods to the Absence Management section. - Modified the GetAbsences method to remove the *currently-absent* parameter. - Added the read-only *StartDate* and *EstimatedEndDate* properties to the AbsenceRTWQuestionnaireData object. - Modified the GetAbsences method to add the *questionnaire-status* parameter.
1.0.81	BulkReallocate – parameter *patient* added so referrals of one patient can be realocated.
1.0.82	PersonDemographicData – *Permissions* field added.
1.0.84	PersonDemographicData – InsurerData *added* getInsurer method setInsurer method Companies section Insurers method
1.0.85	AppointmentData – added *AuthorisationCode* field
1.0.87	Case Management section added, including CaseOverviewData and CaseData data objects. Also, added a 'case' parameter to GetReferrals and GetAppointments and a 'case-title' parameter to StartReferral.
1.0.88	Add CompanyType to ChargeBandData
1.0.89	Get Questionnaires under Appointments changed to allow getting questionnaires associated with a particular appointment.
1.0.90	Added the default country calling code '*DefaultCallingCode'* to the return of GetConfig.
1.0.90	GET web method deprecated for methods: - [DEPRECATED] SendPasswordResetEmail - [DEPRECATED] SubmitPasswordResetEmail - AcceptTermsAndConditions Parameters moved to POST. The GET is still supported for backward compatibility and will be removed in future.

1.0.90	We replaced for methods: - [DEPRECATED] SendValidationEmail replaced with SendActivationEmail - [DEPRECATED] SubmitValidationEmail replaced with SubmitActivationEmail - [DEPRECATED] SendValidationSMS replaced with SendActivationSMS - [DEPRECATED] SubmitValidationSMS replaced with SubmitActicationSMS Previous methods has been deprecated and we plan to remove them in future.
1.0.91	Added ProvideDepsandDivs field in ChargeBandData.
1.0.92	Added DaysToReviewDischargeLetterByPatient to ReferralData and ReferralOverviewData
1.0.93	Added GetDefinitions, Availability and StartPathway
1.0.94	Added GetDocument call
1.0.95	Added GetTypes call and DocumentType. Added *AccessibleDocumentTypes* and *AllDocumentTypesVisible* into UserData.
1.0.96	For security reasons we have updated our password reset feature: - [DEPRECATED] SendPasswordResetEmail replaced with Password reset - [DEPRECATED] SubmitPasswordResetEmail replaced with Password reset The deprecated methods are still supported but to make sure these methods are secure we had to update [DEPRECATED] SendPasswordResetEmail so it now does not verify *name*, *surname* and *dateOfBirth*. These parameters (if provided) are being ignored. The demographic details are validated within [DEPRECATED] SubmitPasswordResetEmail.
1.0.97	The ASP.NET session ID and token can now be sent via the HTTP headers “x-session-id" and “x-token” respectively. This is now the recommended method. The previous methods (third party cookie or inclusion within the POST body) are still supported but now deprecated.
1.0.98	Added methods for sending and submitting 2FA codes.
1.0.99	Added methods for gender identities and pronouns. Added genders and pronouns to PersonDemographicData.
1.0.100	Added method for retrieving payment countries PaymentCountries
1.0.101	9.8 GetProposedAppointments and 10.7 GetProposedAppointments: - *from-date* data type changed from DateTime to LocalDateTime ○ the ‘time’ part is now accepted - *to-date, time-from, time-to, number-of-results* filters added New section Anonymous Appointment Booking introduced.
1.0.102	Added methods for searching and booking proposed time slots: GetProposedTimeSlots and BookProposedTimeSlot, and the anonymous versions: GetProposedTimeSlots and BookProposedTimeSlot.
1.0.103	New section Address Finder introduced.
1.0.104	A new user permission ‘*CanModifyPatientEmailAndPhone*‘

4 Communication overview

This section provides basic information how the client communicates with the Meddbase Patient Portal API.

The API provides:

- working with patient's profile
- managing appointments
- filling out questionnaires
- messaging with clinicians
- review patient's medical history

The API is designed as a REST service. The client uses HTTP GET and POST request to communicate with the server. All POST data and all GET response data is formatted in the JSON data format.

4.1 URL format

All URL requests follow the following form:

```
https://api.meddbase.com/patientportalapi/<section>/<method>?<param1>=<value1>&<param2>=<value2>&...&<paramX>=<valueX>&token=<token>
```

For example, the Login method of the Authentication section that requests two parameters: the '*username*' and the '*password*':

```
https://api.meddbase.com/patientportalapi/auth/address-lookup?postcode=<postcode>&house=<house>&token=<token>
```

URL includes:

- `section` – A name of the section of the API that the method belongs to (e.g. 'auth', 'patient', 'finance', etc.). You can find the section name within the method description.
- `method` – A name of the method that the client wants to request (e.g. 'login', 'logout', etc.). You can find the method name within the method description.
- `paramN`, `valueN` – List of parameters. All parameters are separated using the ampersand (&) character. The list of parameters and their possible values are enumerated within the method description. Certain parameters are required while some are optional. Some methods do not require any parameters.
- `token` – A token provided by the API. See Authentication Token.

4.2 POST data format

Certain methods use the HTTP POST method to send data to the server. The client must format all POST data into the JSON format.

For example, the method ValidateProfileData requires two POST parameters: the '*regCode*' and the '*demog*'. The POST data example:

```
{
  "regCode": "< value >"
  "demog": {
    < properties >
  }
}
```

4.3 Response data format

All response data is formatted in the JSON data format. The server response is an envelope that always looks like this example:

```
{
  "status": "ok/error"
  "result": {
    < result object according a called method>
  },
  "error": {
    < a ServiceExceptionData object >
  }
}
```

The response envelope includes:

- status – *‘ok’* for successful response or *‘error’* when an exception throws.
- result – returns object from the server within successful response.
- error – a ServiceExceptionData object that provides information about an exception that was thrown.

Remarks

The JavaScript API library wraps the response envelope into *‘done’* and *‘fail’* methods with an appropriate *result*/*error* object. The client that uses JavaScript API library doesn’t need to consider this envelope.

4.4 Client authentication

The client must provide the client’s identification key in every GET/POST request.

The client identification key is provided by Medical Management Systems to the client. This key is a unique and can be built into the application code like a constant.

The client identification key must be provided in a cookie with the key “*Client-key*”.

Example:

```
Client-key: a2gft68m-5d8b
```

4.5 JavaScript API library

The client can use a JavaScript API library provided to the client. The JavaScript library wraps all methods described by this API for the client. The library also manages the token management for the client.

The JavaScript library is provided at: https://api.meddbase.com/scripts/patientportalapi-1.1.js

The JavaScript library uses the jQuery, so the client must include the jQuery script to the client’s html output. The jQuery version must be at least 1.8.

CORS policy

The client must ask our support team to include their domain in our CORS policy. For testing purposes, we recommend you to setup **.meddbase.com* domain on your PC. You can use the *hosts* file to set up DNS for your local testing domain.

Working with library

To use the JavaScript library the client must create a new instance of the PatientPortalAPI object:

```
patientportal = new PatientPortalAPI();
```

All function calls follow this form:

```
patientportal.<section>.<methodName>({
    param1: value1,
    paramN: valueN
})
.done(function (receivedData) {
        // do something
    })
.fail(function (receivedError) {
        // do something
    });
```

Where:

- section – The name of the section in the API that the method belongs to (e.g. 'auth', 'patient', 'finance', 'appointment', 'prescription', 'feed, 'medicalHistory'). You can find the section name within the method description.
- methodName – A name of the method as provided by this API in camel-case (e.g. 'login', 'logout', 'getAllowedTitles', etc.). You can find the method name within the method description.
- paramN: valueN – List of parameters. The list of parameters and their possible values are enumerated within the method description. Certain parameters are required while some are optional. Some methods don't require any parameters.
- done(function(receiveData){}) – A callback function that is executed if the request succeeds (ends with a HTTP status code 200).
    - receivedData – The data that the server returns. You can find the type of the data within the method description.
- fail(function(receivedError)) – A callback function to be called if the request fails (ends with a HTTP status code other than 200).
    - receivedError – The ServiceExceptionData that the server throws.

Example of using the JavaScript library

This example shows calling of the Login method and the UpdateDemographicData method.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script>
<script src="https://api.meddbase.com/scripts/patientportalapi-1.1.js"></script>

<script type="text/javascript">
    // create a new instance of PatientPortalAPI object
    patientportal = new PatientPortalAPI();
    // set the client identification key
    patientportal.setClientKey('a2gft68m-5d8b');

    function Login() {
        // Calling the API method
        patientportal.auth.login({
            username: 'my name',
            password: 'my password',
        })
        .done(function (receivedData) {
            // callback function that is executed if the user is logged in successfully
            alert('User logged in. SessionID is ' + receivedData.SessionID);
        })
        .fail(function (receivedError) {
            // callback function that is called if the user is not logged in
            alert('User not logged in: ' + receivedError.Message);
        });
    }

    function UpdateDemographicData() {
        // Calling the API method
        patientportal.patient.updateDemographicData({
            password: 'my password',
            demog: {
                Name: 'James'
            }
        })
        .done(function (receivedData) {
            // callback function that is executed if patient's demographic data was updated

            alert('Data updated successfully.');
        })
        .fail(function (receivedError) {
            // callback function that is called if patient's demographic data wasn't update
d
            alert('Update failed: ' + receivedError.Message);
        });
    }
</script>

<button onclick="Login();return false;">login</button>
<button onclick="UpdateDemographicData();return false;">update patient's data</button>
```

5 Authentication

This section describes how the client can open and close a security context and how the client can register a new patient on the Meddbase server.

5.1 Authentication Token

The authentication token needs to be included in every request sent to the API gateway while the user is logged-in. The token is a unique string generated for the current logged-in session. It changes on login/logout and expires together with the session if the session is not accessed for a lengthy period.

The only way to get the token is from the response of the Login request and it is mandatory for all methods where the user is logged-in.

If the passed token is not valid, an Invalid Token exception (see Error handling) is thrown.

One of the purposes of the token is to protect against the Cross-Site Request Forgery (CSRF).

Note that you do not need to manage the authentication token if you use our API Javascript library. It manages this for you automatically.

5.2 Login

Allows the client to authenticate login credentials, initiate the 2FA process (if required) and create the security context (session) with the Meddbase server.

JavaScript library method

patientportal.auth.login({username: <username>, password: <password>, isOH: <is-oh>});

HTTP Method

POST

Url

/patientportalapi/auth/login

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

username	string	Patient's email address.
password	string	Patient's plain text password.

Returns

AuthenticationData

Remarks

If authentication is successful, a new session for the client is created and a new key *ASP.NET_SessionId* is added to the cookie. The client must ensure that this cookie will be sent in all further requests that needs to use the person's security context.

If 2FA is required, the profile will be marked as awaiting 2FA and will not be logged in. Once marked as awaiting 2FA, Submit2faCode must be completed successfully for the profile to be logged in.

The *SessionID* can expire, so the client must use the ValidateLogin method to ensure that the security context is still created and the *SessionID* is still valid. If not, the client must login again.

5.3 Direct Login

Allows the client to authenticate their login using a unique token and to create the security context (session) with the Meddbase server.

JavaScript library method

patientportal.auth.directLogin({loginToken: <sslogin-token>,});

HTTP Method

POST

Url

/patientportalapi/auth/direct-login

URL Parameters

loginToken	guid	Unique login token. This token expires after 20 seconds of creation.

Returns

AuthenticationData

Remarks

This is used for Single Sign-on.

If authentication is successful, a new session for the client is created and a new key *ASP.NET_SessionId* is added to the cookie. The client must ensure that this cookie will be sent in all further requests that needs to use the person's security context.

The *SessionID* can expire, so the client must use the ValidateLogin method to ensure that the security context is still created and the *SessionID* is still valid. If not, the client must login again.

5.4 ValidateLogin

Confirms the validity of the current security context. Returns the current validation token and *RemainingTime* - the number of seconds for the session to stay active without any further request. Note that calling *validateLogin* does keep the session active.

JavaScript library method

patientportal.auth.validateLogin();

HTTP Method

GET

Url

/patientportalapi/auth/validate

Returned JSON

```
{
    "Token": "380da8b1-d4ca-4d92-98d2-bbe3ce258a49",
    "RemainingTime": 569
}
```

Remarks

If the actual security context is valid, HTTP status code is 2xx. Otherwise check the exception (see Error handling) and use Login to create a new security context.

The client must provide the *ASP.NET_SessionId* key in the cookie.

5.5 Logout

Closes the security context.

JavaScript library method

patientportal.auth.logout();

HTTP Method

GET

Url

/patientportalapi/auth/logout

Remarks

If the user is not logged in the exception is thrown (see Error handling).

The client must provide the *ASP.NET_SessionId* key in the cookie.

5.6 ValidateRegCode

Confirms the validity of the online sign up access code.

JavaScript library method

patientportal.auth.validateRegCode({regCode: <reg-code>, isOH: <is-oh>});

HTTP Method

GET

Url

/patientportalapi/auth/validate-reg-code

URL Parameters

reg-code	string	Online sign up access code that is provided to the patient.
is-oh	bool	True for the referral portal. False for the patient portal.

Returns

ChargeBandData

Remarks

If the registration code is invalid, check the exception (see Error handling).

5.7 GetRegCodeInfo

Returns information for the online sign-up access code (i.e. charge band information, because the online sign-up access code is assigned to the charge band).

JavaScript library method

patientportal.auth.getRegCodeInfo({regCode: <reg-code>});

HTTP Method

GET

Url

/patientportalapi/auth/reg-code-info

URL Parameters

reg-code	string	Online sign up access code that is provided to the patient.
is-oh	bool	True for the referral portal. False for the patient portal.

Returns

ChargeBandData

5.8 GetAllowedTitles

Gets allowed titles (for example: Mr, Mrs, etc.) that can be used within registration process.

JavaScript library method

```
patientportal.auth.getAllowedTitles({
        regCode: <reg-code>,
        membershipCode: <membership-code>,
        isOH: <is-oh>
});
```

HTTP Method

GET

Url

/patientportalapi/auth/allowed-titles

URL Parameters

reg-code	string (optional)	Online sign up access code that is provided to the patient.
membership-code	string (optional)	Membership scheme code that is provided by Medical Management Systems to the client. See Membership scheme
is-oh	bool	True for the referral portal. False for the patient portal.

Returns

string[]

Remarks

One of the optional parameter *reg-code* or *membership-code* must be provided. If membership code is provided, registration code is ignored.

5.9 ValidateProfileData

Confirms the validity of the current patient's demographic data.

JavaScript library method

```
patientportal.auth.validateProfileData({
        regCode: <regCode>,
        demog: <demog>,
        membershipCode: <membership-code>,
        isOH: <is-oh>
});
```

HTTP Method

POST

Url

/patientportalapi/auth/validate-profile-data

POST Parameters

regCode	string (partially optional)	Online sign up access code that is provided to the patient.
demog	PersonDemographicData	Defines person's details.
membershipCode	string (partially optional)	Membership scheme code that is provided by Medical Management Systems to the client. See Membership scheme
is-oh	bool	True for the referral portal. False for the patient portal.

Remarks

One of the optional parameter *regCode* or *membershipCode* must be provided. If membership code is provided, regCode is ignored.

If the validation process is successful HTTP status code is 2xx. Otherwise check the exception (see Error handling).

Valid registration data:

- *Name*, *Surname*, *DateOfBirth*, *EmailAddress*, *Mobile*, *Title, Address.PostCode* and *Password* must be provided.
- *Password* should be at least 8 characters long and should contain at least 1 number.

5.10 RegisterPatient

Creates a new patient's portal profile. If the patient already exists in the Meddbase system it joins the patient's account with this portal profile.

JavaScript library method

```
patientportal.auth.registerPatient({
        regCode: <regCode>,
        demog: <demog>,
        membershipCode: <membership-code>,
        isOH: <is-oh>
});
```

HTTP Method

POST

Url

/patientportalapi/auth/register-patient

POST Parameters

regCode	string (partially optional)	Online sign up access code that is provided to the patient.
demog	PersonDemographicData	Defines person's details.
membershipCode	string (partially optional)	Membership scheme code that is provided by Medical Management Systems to the client. See Membership scheme
is-oh	bool	True for the referral portal. False for the patient portal.

Returns

PatientRegistrationResultData

Remarks

One of the optional parameter *regCode* or *membershipCode* must be provided. If membership code is provided, regCode is ignored.

The client is responsible for making sure T&C is provided and accepted and the patient also accepts statistical processing of data.

If the registration process is successful, HTTP status code is 2xx. Otherwise check the exception (see Error handling). If the account already exists on our system the client gets exception where event code is 10001.

The Meddbase system sends the activation email with an activation link to the patient's email address or it sends the activation SMS with an activation code to the patient's mobile number or it sends both (use returned PatientRegistrationResultData to find out what activation is used). The user must activate her account first before being able to log-on. If the email activation is used the client should give the user advice that the confirmation email may be in their junk/spam folder.

The registration creates a standard session and the activation needs to pass a correct session ID within the request.

POST data example

```
{
  "regCode": "1234",
  "demog": {
    "Title": "Mr",
    "Name": "John",
    "Surname": "Lemon",
    "DateOfBirth": "1958-08-02T00:00:00",
    "EmailAddress": "john.lemon@test.com",
    "Password": "jon4535lemon",
    ... <see PersonDemographicData for all properties>
  }
}
```

5.11 SendActivationEmail

Sends the activation email to the patient's email address. The activation email includes a link to the patient portal website and contains the activation key: https://www.yourportal.com/#/confirm-email?key=<key>

JavaScript library method

patientportal.auth.sendActivationEmail({isOH: <is-oh>});

HTTP Method

GET

Url

/patientportalapi/auth/send-activation-email

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

Remarks

Sometimes the patient needs to send the activation email again because (for example) she deletes it before activating her account or the last validation email was marked as a spam and it was deleted automatically.

The client needs to submit confirmation key from the URL by calling SubmitActivationEmail.

Remember the method needs the session ID to be set.

5.12 SubmitActivationEmail

Submits the activation key that the client retrieves from the URL and returns confirmation.

JavaScript library method

patientportal.auth.submitActivationEmail({isOH: <is-oh>});

HTTP Method

POST

Url

/patientportalapi/auth/submit-validation-email

GET Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

key	string	Confirmation key that is included in the URL.

Returns

ActivationConfirmation

Remarks

If the activation completes successfully the API returns the confirmation object. The confirmation may include the outstanding invoice (usually the membership fee) that should be paid. To pay for the invoice the client needs to call ProvidePayment method.

5.13 SendActivationSMS

Sends the activation SMS to the patient's mobile telephone number.

JavaScript library method

patientportal.auth.sendActivationSMS({isOH: <is-oh>});

HTTP Method

GET

Url

/patientportalapi/auth/send-activation-sms

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

Remarks

Sometimes the patient needs to send the activation SMS again because (for example) she deletes it before activating her account.

5.14 SubmitActicationSMS

Submits the activation number from the activation SMS and returns confirmation.

JavaScript library method

patientportal.auth.submitActivationSMS({code: <code>, isOH: <is-oh>});

HTTP Method

GET

Url

/patientportalapi/auth/submit-activation-sms

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

URL Parameters

code	string	Activation code.

Returns

ActivationConfirmation

Remarks

If the activation completes successfully the API returns the confirmation object. The confirmation may include the outstanding invoice (usually the membership fee) that should be paid. To pay for the invoice the client needs to call ProvidePayment method.

5.15 [DEPRECATED] SendValidationEmail

This method has been deprecated. Please use SendActivationEmail.

Sends the validation email to the patient's email address. Validation email includes URL to the patient portal website and contains validation key: http://www.yourportal.com/#/confirm-email?key=<key>

JavaScript library method

patientportal.auth.sendValidationEmail({email: <email>, isOH: <is-oh>});

HTTP Method

GET

Url

/patientportalapi/auth/send-validation-email

URL Parameters

email	string	Patient's email address.
is-oh	bool	True for the referral portal. False for the patient portal.

Remarks

Sometimes the patient needs to send the validation email again because (for example) she deletes it before activating her account or the last validation email was marked as a spam and it was deleted automatically.

The client needs to submit confirmation key from the URL by calling [DEPRECATED] SubmitValidationEmail.

5.16 [DEPRECATED] SubmitValidationEmail

This method has been deprecated. Please use SubmitActivationEmail.

Submits the confirmation key that the client retrieves from the URL and returns confirmation.

JavaScript library method

patientportal.auth.submitValidationEmail({key: <key>, isOH: <is-oh>});

HTTP Method

GET

Url

/patientportalapi/auth/submit-validation-email

URL Parameters

key	string	Confirmation key that is included in the URL.
is-oh	bool	True for the referral portal. False for the patient portal.

Returns

ActivationConfirmation

Remarks

If the activation is completed successfully the API returns the confirmation object. The confirmation may include the outstanding invoice (usually the membership fee) that should be paid. To pay for the invoice the client needs to call ProvidePayment method.

5.17 [DEPRECATED] SendValidationSMS

This method has been deprecated. Please use SendActivationSMS.

Sends the validation SMS to the patient’s mobile telephone number.

JavaScript library method

patientportal.auth.sendValidationSMS({mobile: <mobile>, isOH: <is-oh>});

HTTP Method

GET

Url

/patientportalapi/auth/send-validation-sms

URL Parameters

mobile	string	Patient’s mobile telephone number.
is-oh	bool	True for the referral portal. False for the patient portal.

Remarks

Sometimes the patient needs to send the validation SMS again because (for example) she deletes it before activating her account.

5.18 [DEPRECATED] SubmitValidationSMS

This method has been deprecated. Please use SubmitActicationSMS.

Submits the validation number from the validation SMS and returns confirmation.

JavaScript library method

patientportal.auth.submitValidationSMS({code: <code>, mobile: <mobile>, isOH: <is-oh>});

HTTP Method

GET

Url

/patientportalapi/auth/submit-validation-sms

URL Parameters

code	string	Validation code.
mobile	string	Patient’s mobile telephone number.
is-oh	bool	True for the referral portal. False for the patient portal.

Returns

ActivationConfirmation

Remarks

If the activation is completed successfully the API returns the confirmation object. The confirmation may include the outstanding invoice (usually the membership fee) that should be paid. To pay for the invoice the client needs to call ProvidePayment method.

5.19 [DEPRECATED] SendPasswordResetEmail

This method has been deprecated. Please see Password reset section for more details.

If you still use this method please be aware that parameters *name*, *surname* and *dateOfBirth* are being ignored for a security reasons. The demographic details are validated within [DEPRECATED] SubmitPasswordResetEmail.

Disables patient's account, sends a password reset request email that includes a URL address to the patient portal website that contains password reset key: http://www.yourportal.com/#/password-reset?key=<key>

JavaScript library method

```
patientportal.auth.sendPasswordResetEmail({
        email: <email>,
        name: <name>,
        surname: <surname>,
        dateOfBirth: <date-of-birth>,
        isOH: <is-oh>
});
```

HTTP Method

POST

Url

/patientportalapi/auth/send-password-reset-email

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

email	string	Patient's email address.
name	string	Patient's name.
surname	string	Patient's surname.
dateOfBirth	DateTime	Patient's date of birth.

Remarks

The server send an email with further instructions to the patient's email address. The patient must follow those instructions to finish password reset process. The client should give the user advice that this email may be in their junk/spam folder.

The client needs to submit reset key from the URL and additional info from the patient by calling [DEPRECATED] SubmitPasswordResetEmail.

5.20 [DEPRECATED] SubmitPasswordResetEmail

This method has been deprecated. Please see Password reset section for more details.

Submit the password reset key that the client retrieves from the URL and additional information that the client collets from the patient.

JavaScript library method

```
patientportal.auth.submitPasswordResetEmail({
        key: <key>,
        name: <name>,
        surname: <surname>,
        dateOfBirth: <date-of-birth>,
        postcode: <postcode>,
        newPassword: <new-password>,
        isOH: <is-oh>
});
```

HTTP Method

POST

Url

/patientportalapi/auth/submit-password-reset-email

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

key	string	Reset key that is included in the URL.
name	string	Patient's name.
surname	string	Patient's surname.
dateOfBirth	DateTime	Patient's date of birth.
postcode	string	Patient's address postcode.
newPassword	string	New password for patient's account.

5.21 GetServerState

Returns information about actual server state.

JavaScript library method

```
patientportal.auth.getServerState();
```

HTTP Method

GET

Url

/patientportalapi/auth/server-state

Returns

ServerStateData

Remarks

This method doesn't require any authentication. The client can use this method to test connection to the server.

5.22 [DEPRECATED] AddressLookup

This method has been deprecated, please use the Address Finder section instead. Returns a number of possible address for the postcode.

JavaScript library method

patientportal.auth.addressLookup({postcode: <postcode>, house: <house>});

HTTP Method

GET

Url

/patientportalapi/auth/address-lookup

URL Parameters

postcode	string	Case-insensitive postcode without space (e.g. N70NH or n70nh).
house	string (optional)	House name or house number.

Returns

AddressData[]

5.23 GetTermsAndConditions

Returns the terms and conditions texts.

JavaScript library method

patientportal.auth.getTermsAndConditions();

HTTP Method

GET

Url

/patientportalapi/auth/terms-and-conditions

Returns

TermsAndConditions

5.24 GetConfig

Returns the chamber basic information and the permissions of the currently logged in user has.

JavaScript library method

```
patientportal.auth.getConfig(({
        companyIdentifier: <company-identifier>
});
```

HTTP Method

GET

Url

/patientportalapi/auth/config

URL Parameters

company-identifier	String (optional)	Unique 'api-identifier' for employer's single sign on. If the user is not logged in, this returns as a part of configuration, if single sign on is available for the employer corresponding to the unique identifier. If user is logged in, this is ignored and instead the single sign on information for the logged in user is returned.

Returns

ConfigData

5.25 GetDepartmentsAndDivisions

Returns the list of employer departments and divisions.

JavaScript library method

```
patientportal.auth.getDepartmentsAndDivisions({regCode: <reg-code>, isOH: <is-oh>});
```

HTTP Method

GET

Url

/patientportalapi/auth/departments-divisions

URL Parameters

reg-code	string	Online referral portal sign up access code that is provided to the manager.
is-oh	bool	True for the referral portal. False for the patient portal.

Returned JSON

```
{
  "Departments": [
    <list of DepartmentData>
  ]
  "Divisions": [
    <list of DivisionData>
  ]
}
```

Remarks

For patient portal, providing departments and divisions during registration setting needs to enabled. If the code isn’t the sign up code or the settings is disabled, an exception is thrown.

5.26 GetSSOLoginDetails

Returns unique URL to initiate a service provider initiated single sign on

JavaScript library method

patientportal.auth. getSSOLoginDetails ({companyIdentifier: < company-identifier>});

HTTP Method

GET

Url

/patientportalapi/auth/sso-login-details

URL Parameters

company-identifier	string	Unique identifier to identify the employer’s single sign on configuration.

Returned JSON

```
URL to be redirected to login through an identity provider
```

Remarks

This method is allowed only for the referral portal. Please see Single sign on for more details.

5.27 Send2faCode

Sends generated 2FA code via SMS or email and returns delivery method used.

JavaScript library method

patientportal.auth.send2faCode();

HTTP Method

GET

Url

/patientportalapi/auth/send-2fa-code

Returned JSON

```
{
  "DeliveryMethod": "Mobile"
}
```

Remarks

Code will be sent via SMS if the profile has a mobile number, otherwise it will be sent via email.

5.28 Send2faCodeEmail/Send2faCodeSMS

Sends generated 2FA code via specified method and returns delivery method.

JavaScript library methods

patientportal.auth.send2faCodeEmail();

patientportal.auth.send2faCodeSMS();

HTTP Method

POST

Url

/patientportalapi/auth/send-2fa-code

POST Parameters

useMobile	boolean	True to send 2FA code via mobile. False to send 2FA code via email.

Returned JSON

```
{
  "DeliveryMethod": "Mobile"
}
```

Remarks

Multiple methods are provided in the JavaScript library for ease of use.

5.29 Submit2faCode

Submits 2FA code and completes profile login.

JavaScript library method

patientportal.auth.submit2faCode({code: <code>});

HTTP Method

POST

Url

/patientportalapi/auth/submit-2fa-code

POST Parameters

code	string	2FA code.

Returns

AuthenticationData

6 Address Finder

This section provides services for searching and retrieving address details. Currently, only the UK is supported.

6.1 Search

Returns a number of possible shallow search address results for the postcode.

JavaScript library method

patientportal.addressFinder.search ({postcode: <postcode>});

HTTP Method

GET

Url

/patientportalapi/address-finder/search

URL Parameters

postcode	string	Case-insensitive postcode with space (e.g. N7 0NH or n7 0nh).

Returned JSON

Shallow Search Result Data[]

6.2 Address

Returns address details for the provided encrypted key.

JavaScript library method

patientportal.addressFinder.address ({key: <key>});

HTTP Method

GET

Url

/patientportalapi/address-finder/address

URL Parameters

key	string	An encrypted key value containing partial address information.

Returned JSON

AddressData

7 Patient

This section provides methods for working with the patient's account.

Each method in this section needs the client to provide the *ASP.NET_SessionId* key in a cookie (see Login).

7.1 GetAllowedTitles

Gets allowed titles (for example: Mr, Mrs, etc.) that can be used to update person's details.

JavaScript library method

patientportal.patient.getAllpwedTitles();

HTTP Method

GET

Url

/patientportalapi/patient/allowed-titles

Returns

string[]

7.2 GetDemographicData

Gets the patient's demographic data.

JavaScript library method

patientportal.patient.getDemographicData();

HTTP Method

GET

Url

/patientportalapi/patient/demographic-data

Returns

PersonDemographicData

7.3 GetEmployerDemographicData

Gets the patient's employer demographic data.

JavaScript library method

patientportal.patient.getEmployerDemographicData();

HTTP Method

GET

Url

/patientportalapi/patient/employer-demographic-data

Returns

CompanyDemographicData

7.4 UpdateDemographicData

Update the patient's demographic data.

JavaScript library method

patientportal.patient.updateDemographicData({password: <password>, demog: <demog>});

HTTP Method

POST

Url

/patientportalapi/patient/demographic-data

POST Parameters

password	string	Patient's plain text password to authenticate the patient.
demog	PersonDemographicData	Patient's demographic data to update. If any property is empty (""), it will be updated. If any property is null (or not defined), it will not be updated.

POST data example

```
{
  "password": "pass123",
  "demog": {
    "Title": "Mr",
    "Name": "John",
    "Surname": "Lemon",
    "DateOfBirth": "1958-08-02T00:00:00",
    "EmailAddress": "john.lemon@test.com",
    "Telephone": ""
  }
}
```

In the example above only properties *Title*, *Name*, *Surname*, *DataOfBirth*, *EmailAddress* and *Telephone* will update. All another properties will not be changed because they are not in POST data.

7.5 AcceptTermsAndConditions

Accepts both the Terms and conditions and the statistical processing of the patient data terms. The client needs to call GetTermsAndConditions to get texts of both. The patient have to accept both terms at once.

JavaScript library method

patientportal.patient.acceptTermsAndConditions({password: <password>});

HTTP Method

POST

Url

/patientportalapi/patient/accept-terms-and-conditions

POST Parameters

password	string	Patient’s plain text password to authenticate the patient.

7.6 GetDepartmentsAndDivisions

Returns the list of employer departments and divisions.

JavaScript library method

patientportal.patient.getDepartmentsAndDivisions();

HTTP Method

GET

Url

/patientportalapi/patient/departments-divisions

Returned JSON

```
{
  "Departments": [
    <list of DepartmentData>
  ]
  "Divisions": [
    <list of DivisionData>
  ]
}
```

7.7 GetEmployeeStatuses

Returns a list of employee statuses.

JavaScript library method

patientportal.patient.getEmployeeStatuses();

HTTP Method

GET

Url

/patientportalapi/patient/employee-statuses

Returns

EmployeeStatus[]

7.8 GetInsurer

Gets a patient's insurer. Only available for patient portal. Only public companies have details returned like Name, otherwise the returned object only has Encrypted key.

JavaScript library method

patientportal.patient.getInsurer();

HTTP Method

GET

Url

/patientportalapi/patient/insurer

Returns

InsurerData

7.9 SetInsurer

Sets a patient insurer. Only available for patient portal. Only public companies can be set.

JavaScript library method

patientportal.patient.setInsurer();

HTTP Method

POST

Url

/patientportalapi/patient/insurer

POST Parameters

insurerData	InsurerData	Patient's insurer data to update.

POST data example

```
{
  "insurerData": {
    {
    "Key": "1c5aabbdf813ad78c48cc8e6d8584162",
    "MemberNumber": "Test member number"
  }
  }
}
```

8 Password reset

The section helps you to reset the patient/manager password. The workflow is:

1. SendLink asks for the password reset link to be sent via email
2. ValidateKey validates the password reset link and tells you what kind of validation is enabled for the account (SMS vs. demographics match)
3. If SMS validation is enabled
   a. SendValidation asks for the SMS validation message to be sent
   b. SubmitValidationCode verifies the code and establishes a session with the client
   c. SetPassword sets the new password
4. If the demographics match is enabled (not preferred option)
   a. [OBSOLETE] SetPasswordUsingDemographics sets the new password

The demographics match validation is not recommended, and it is supported until all clients move to SMS validation.

8.1 SendLink

Sends a password reset link via an email to the patient's personal email address or to the manager's work email address. The link includes the portal URL with a password reset key in a format: http://www.yourportal.com/#/password-reset?key=<key>

JavaScript library method

patientportal.passwordReset.sendLink({ email: <email> });

HTTP Method

POST

Url

/patientportalapi/password-reset/send-link

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

email	string	Email address.

Remarks

The email is sent even if the email address isn't registered with us. In that case we inform user the email address is incorrect and we recommend them to try again or register.

8.2 ValidateKey

Validates the password reset key that is included in the URL sent via the email in the previous API method.

JavaScript library method

patientportal.passwordReset.validateKey({ key: <key> });

HTTP Method

POST

Url

/patientportalapi/password-reset/validate-key

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

key	key	The password reset key.

Returns

The method either fails with an exception message (e.g. Link expired) or it passes and tells you what method is used to verify the patient/manager.

- *SMSVerfication* – we verify via a validation code that is sent to the mobile number
    - Use *MobileHint* as so the patient can partially verify what mobile number is going to be used.
- *DemographicsVerification* – we verify via a full demographics match
    - Note this method is not recommended and will be removed in future. It is supported until all clients move to SMS verification.

```
{
  "SMSVerification": true,
  "DemographicsVerification": false,
  "MobileHint": "xxxxxxxx569"
}
```

8.3 SendValidationSMS

Sends a validation code for a specific password reset key to the mobile number.

JavaScript library method

patientportal.passwordReset.sendValidationSMS({ key: <key> });

HTTP Method

POST

Url

/patientportalapi/password-reset/send-validation-sms

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

key	key	The password reset key.

8.4 SubmitValidationCode

Submits the validation code for a specific password reset key. Returns a new Session ID the client could use to set a new password.

JavaScript library method

patientportal.passwordReset.submitValidationCode({ key: <key>, code: <code> });

HTTP Method

POST

Url

/patientportalapi/password-reset/submit-validation-code

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

key	key	The password reset key.
code	code	The validation code.

Remarks

Submitting the validation code creates a temporary security context which is valid for 15 minutes. Within this time frame the patient must complete the password reset process.

The client may use ValidateKey to verify whether the security context runs out and request new validation code to create a new security context.

If something goes wrong please check exception (see Error handling).

Returns

The *SessionId* is used to establish a security context via *ASP.NET_SessionId* cookie parameter.

```
{
  "SessionId":"R9I8Dr8rGQcQ6EOfxVyOBX2r8zt4KTO06vW0O2+GyFc="
}
```

8.5 SetPassword

Sets the new password. This is used if the SMS verification method (see previous API calls) has been used.

JavaScript library method

patientportal.passwordReset.setPassword({ key: <key>, newPassword: <newPassword> });

HTTP Method

POST

Url

/patientportalapi/password-reset/set-password

URL Parameters

is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

key	key	The password reset key.
newPassword	newPassword	The new password.

Remarks

Please remember to provide *ASP.NET_SessionId* cookie with the session ID provided provided upon SubmitValidationCode.

8.6 [OBSOLETE] SetPasswordUsingDemographics

Sets the new password using the demographics match verification. This method is supported for legacy reasons and will be removed in the future.

JavaScript library method

```
patientportal.auth.setPasswordUsingDemographics({
        key: <key>,
        name: <name>,
        surname: <surname>,
        dateOfBirth: <dateOfBirth>,
        postcode: <postcode>,
        newPassword: <newPassword>
});
```

HTTP Method

POST

Url

/patientportalapi/auth/set-password-using-demographics

URL Parameters

Is-oh	bool	True for the referral portal. False for the patient portal.

POST Parameters

key	string	Reset key that is included in the URL.
name	string	Patient’s name.
surname	string	Patient’s surname.
dateOfBirth	DateTime	Patient’s date of birth.
postcode	string	Patient’s address postcode.
newPassword	string	New password for patient’s account.

9 Appointments

This section provides methods for working with appointments and questionnaires.

Each method in this section needs the client to provide the *ASP.NET_SessionId* key in a cookie (see Login).

9.1 GetPayerTypes

Gets payer types (e.g. Patient, Employer, Issuer, School, etc.). This allows the patient to nominate another payer to pay for services.

JavaScript library method

```
patientportal.appointment.getPayerTypes({
        patient: <patient>,
        referral: <referral>,
        recall: <recall>
});
```

HTTP Method

GET

Url

/patientportalapi/appointment/payer-types

URL Parameters

patient	string (optional)	The key of the patient provided by the API upon section Patients. Used to book an appointment for a different patient within your company. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used to book an appointment for a specific recall.

Returns

PayerType[]

Remarks

The method always returns at least one payer type. If there are more payer types the first type is patient’s default payer type.

9.2 GetAppointmentTypes

Gets appointment types (e.g. Consultation, Health Screen, etc.) available for booking.

JavaScript library method

```
patientportal.appointment.getAppointmentTypes({
        payerType: <payer-type>,
        referralTypes: <referral-types>,
        patient: <patient>,
        referral: <referral>,
        recall: <recall>
});
```

HTTP Method

GET

Url

/patientportalapi/appointment/types

URL Parameters

payer-type	String (optional)	Type of the payer provided by the API upon GetPayerTypes.
referral-types	bool (optional)	True to return the list of appointment types the patient can be referred for. Otherwise false. Default false.
patient	string (optional)	The key of the patient provided by the API upon section Patients. Used to book an appointment for a different patient within your company. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used to book an appointment for a specific recall.

Returns

AppointmentTypeData[]

Remarks

If no module is returned for the appointment type, it doesn't contain modules and the booking process can continue without selecting the module.

9.3 GetServiceTypes

Gets service types (except) modules

JavaScript library method

```
patientportal.appointment.getServiceTypes ({
        payerType: <payer-type>,
        appointmentType: <appointment-type>,
        referralTypes: <referral-types>,
        patient: <patient>,
        referral: <referral>,
        recall: <recall>
});
```

HTTP Method

GET

Url

/patientportalapi/appointment/service-types

URL Parameters

payer-type	string	Type of the payer provided by the API upon GetPayerTypes.
appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes
referral-types	bool (optional)	The key of the referral provided by the API upon GetReferrals. Used to get service types for a specific referral.
patient	string (optional)	The key of the patient provided by the API upon section Patients. Get the services specific for the patient. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used get service types for a specific recall.

Returns

ServiceTypeData[]

Remarks

If no service types are returned for the appointment type, either the appointment type does not allow additional services or no services exist in self book rule.

9.4 GetServices

Gets available services

JavaScript library method

```
patientportal.appointment.getServices ({
        payerType: <payer-type>,
        appointmentType: <appointment-type>,
        serviceType: <service-type>,
        serviceName:<service-name>,
        referralTypes: <referral-types>,
        patient: <patient>,
        referral: <referral>,
        recall: <recall>,
        pageSortColumn: <page-sort-column>,
        pageSortDescending: <page-sort-descending>,
        pageNumber: <page-number>,
        pageSize: <page-size>

});
```

HTTP Method

GET

Url

/patientportalapi/appointment/services

URL Parameters

payer-type	string	Type of the payer provided by the API upon GetPayerTypes.
appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes
service-type	string (optional)	Key of the service type to filter services by, provided by GetServiceTypes
service-name	string (optional)	Name of service to filter services by.
referral-types	bool (optional)	The key of the referral provided by the API upon GetReferrals. Used to get service types for a specific referral.
patient	string (optional)	The key of the patient provided by the API upon section Patients. Get the services specific for the patient. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used get service types for a specific recall.
page-sort-column	int (optional)	The column index to sort the result: - 0 – Name - 1 – Code - 2- Service Type - 3- ServiceId Default: 0,2
page-sort-descending	int (optional)	True to sort result descending. Default false.
page-number	int (optional)	Required page number. Default 1.
page-size	int (optional)	Required page size. Default 10. Minimum 5. Maximum 50.

Returns

ServiceData[]

Remarks

If no services are returned for the appointment type, either the appointment type does not allow additional services or no services exist in self book rule.

9.5 GetSites

Gets possible sites for the specific appointment type. A site is a geographically-unique place. Each site may include additional 'locations', which are typically rooms at the site (Room 123, Surgery, etc.), but may also be geographically different places associated with the site (First floor, West wing, Building 3, etc.).

The API will only present you with locations which mean something to a user. So they're unlikely to get a choice about Room 1, 2, 3, etc. but they may get a choice about Site: My Bank, Location: Floor 1, Floor 50, Floor 100, etc.

JavaScript library method

```
patientportal.appointment.getSites({
        appointmentType: <appointment-type>,
        lat: <lat>,
        long: <long>,
        addressType: <address-type>,
        clinicians: <clinicians>,
        payerType: <payer-type>,
        patient: <patient>,
        referral: <referral>,
        recall: <recall>
});
```

HTTP Method

POST

Url

/patientportalapi/appointment/sites

URL Parameters

appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes.
lat long	decimal (optional)	Latitude and Longitude from GPS so server can sort sites by distance. Latitudes and Longitudes are defined using numerals that have a precision to 6 decimal places. For example, “lat=51.541743&long=-0.13715" is a valid value.
address-type	int (optional)	Parameter specifies how the response should be sorted: - 1 = Home address - 2 = Work address The server uses this value only if the GPS coordinate is not provided.
payer-type	string	Type of the payer provided by the API upon GetPayerTypes.
patient	string (optional)	The key of the patient provided by the API upon section Patients. Used to book an appointment for a different patient within your company. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used to book an appointment for a specific recall.

POST Parameters

clinicians	int[] (optional)	Clinicians filter. Array of identifiers provide by the API upon GetClinicians. Null or empty for any clinicians.
modules	AppointmentModuleData [] (optional)	Selection of modules and additional services provided by the API upon GetAppointmentTypes. Available sites will be filtered according to availability of the specified modules.

Returns

ShallowSearchResultData

A shallow search result of a patient/company address.

Properties

Key	string	An encrypted key value containing partial address information used to retrieve complete address details.
Address	string	A single-line search result address.

JSON Example

```
{
  "Key":
"fc560b40315dc7605fd5ca53e0dcaabc357c69bea3faefa8c6e2ce8129909061956910b77338ee2c2
cdbb1c7c5f7c64bcf338d78bc148f81f6786152d3ef2987b3ab5b1e5588b1db7939bb5e0edffec4614
c4511c4a7a0dfd9bc9077749482b152217c572b0f78552c75be542ffcea6446110af6da78213c1f715
69f35abab7d65f82f382f8b8dc663c8e6a1405bf17c331d379f375ffbc6ec3ebc21a985a69355d1062
2db48eceb7f23b38c5037ed2315c3d858268baae1879f6f84b3b65586742086832ec398acdfd56680a
72991d7bb38bbfd1fa61991aebf0bd1982dc06b",
  "Address": " 2 Paradise Street, Liverpool, Merseyside, L1 8JF"
}
```

9.6

SiteData[]

Remarks

If no *lat/long* or *addressType* is provided the server sort sites by the patient’s home address.

9.7 GetClinicians

Gets possible clinicians/doctors for the specific appointment type.

JavaScript library method

```
patientportal.appointment.getClinicians({
        appointmentType: <appointment-type>,
        sites: <sites>,
        locations: <locations>,
        payerType: <payer-type>,
        patient: <patient>,
        referral: <referral>,
        recall: <recall>
});
```

HTTP Method

POST

Url

/patientportalapi/appointment/clinicians

URL Parameters

appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes.
payer-type	string	Type of the payer provided by the API upon GetPayerTypes.
patient	string (optional)	The key of the patient provided by the API upon section Patients. Used to book an appointment for a different patient within your company. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used to book an appointment for a specific recall.

POST Parameters

sites	int[] (optional)	Sites filter. Array of identifiers provide by the API upon GetSites. Null or empty for any sites.
locations	int[] (optional)	Locations filter. Array of identifiers provide by the API upon GetSites. Null or empty for any locations.
modules	AppointmentModuleData [] (optional)	Selection of modules and additional services provided by the API upon GetAppointmentTypes. Available clinicians will be filtered according to availability of the specified modules.

Returns

ClinicianData[]

9.8 GetProposedAppointments

Gets proposed appointments according given filters.

JavaScript library method

```
patientportal.appointment.getProposedAppointments({
        appointmentType: <appointment-type>,
        clinician: <clinician>,
        clinicianSex: <clinician-sex>,
        site: <site>,
        location: <location>,
        fromDate: <from-date>,
        toDate: <to-date>,
        timePreference: <time-preference>,
        timeFrom: <time-from>,
        timeTo: <time-to>,
        modules: <modules>,
        payerType: <payer-type>,
        patient: <patient>,
        referral: <referral>,
        recall: <recall>,
        numberOfResults: <number-of-results>
});
```

HTTP Method

POST

Url

/patientportalapi/appointment/proposed

URL Parameters

appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes.
clinician	int (optional)	Clinician filter. Identifier provide by the API upon GetClinicians. Allow 0 for any clinician.
clinician-sex	int (optional)	Clinician’s gender filter: - 0 = any sex - 1 = male - 2 = female
site	int (optional)	Site filter. Identifier provide by the API upon GetSites. Allow 0 for any site.
location	int (optional)	Location filter. Identifier provide by the API upon GetSites. Allow 0 for any location.
from-date	LocalDateTime (optional)	Filters appointments on or after provided date/time (inclusive). The parameter is a local date/time on a site. If you search across multiple sites in different time zones the search looks for the local date/time at each site.
to-date	LocalDateTime (optional)	The end date/time for the search (exclusive). If not provided the API is looking for days/months into the future to satisfy the requested *number-of-results*. This may take some time. If you need to get slots for a specific day only it is recommended to provide *to-date* to make the request quick.
time-preference	int (optional)	Preferred time filter: - 0 = any time - 1 = Morning 00:00 – 12:00 - 2 = Afternoon 12:00 – 00:00 Note this parameter is ignored if any from *time-from* or *time-to* are provided.
time-from	string “HH:ss” (optional)	Time filter for the appointment start time. E.g 09:30.
time-to	string “HH:ss” (optional)	Time filter for the appointment finish time. E.g 11:00.
payer-type	string (optional)	Type of the payer provided by the API upon GetPayerTypes.
patient	string (optional)	The key of the patient provided by the API upon section Patients. Used to book an appointment for a different patient within your company. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.

recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used to book an appointment for a specific recall.
number-of-results	Int (optional)	Required number of proposed slots. Value must be in a range 1 – 100.

POST Parameters

modules	AppointmentModuleData[] (optional)	Selection of modules and additional services provided by the API upon GetAppointmentTypes.
services	ServiceData[] (optional)	Selection of services provided by GetServices

Returns

AppointmentData[]

Remarks

The server sorts the result by day and time of the appointment (the closer appointment first).

9.9 BookProposedAppointment

Books and returns the new appointment. Returned appointment contains the amount owned by the patient.

JavaScript library method

```
patientportal.appointment.bookProposedAppointment({
            proposedAppointment: <proposedAppointment> ,
            patient: <patient>,
            referral: <referral>,
            recall: <recall>
});
```

HTTP Method

POST

Url

/patientportalapi/appointment/book

URL Parameters

patient	string (optional)	The key of the patient provided by the API upon section Patients. Used to book an appointment for a different patient within your company. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used to book an appointment for a specific recall.

POST Parameters

proposedAppointment	AppointmentData	The proposed appointment provided by the API upon GetProposedAppointments.

Returns

AppointmentData

Remarks

If the appropriate appointment is taken by another user in a meanwhile the exception is throw. See Error handling. If the response contains the invoice the client use the invoice number and unpaid part of invoice to provide payment.

9.10 GetExistingAppointments

Gets a number of previous and all future logged-in patient appointments.

JavaScript library method

patientportal.appointment.getExistingAppointments();

HTTP Method

GET

Url

/patientportalapi/appointment/existing-appointments

Return

AppointmentData[]

9.11 GetAppointments

Gets the patient's appointments.

JavaScript library method

```
patientportal.appointment.getAppointments({
        patient: <patient>,
        appointmentTypeName: <appointment-type-name>,
        startFrom: <start-from>,
        startTo: <start-to>,
        case: <case>,
        pageSortColumn: <page-sort-column>,
        pageSortDescending: <page-sort-descending>,
        pageNumber: <page-number>,
        pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/appointment/appointments

URL Parameters

patient	string (optional)	The key of the patient provided by the API upon section Patients. Default is undefined which returns appointments for all patients.
appointment-type-name	string (optional)	The appointment type name filter.
start-from	DateTime (optional)	The appointment start date filter.
start-to	DateTime (optional)	The appointment start date filter.
case	int (optional)	The case key for the appointment.
page-sort-column	int (optional)	The column index to sort the result: - 0 –Start date - 1 – Appointment type name Default: 0
page-sort-descending	int (optional)	True to sort result descending. Default false.
page-number	int (optional)	Required page number. Default 1.
page-size	int (optional)	Required page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of AppointmentData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10,
  "SortColumn": 0,
  "SortDescending": false
}
```

9.12 GetAppointmentCancellationInfo

Returns information about cancellation. The client should call this method when the user wants to cancel an appointment. The user will receive information about fees and any important conditions for the cancellation.

JavaScript library method

patientportal.appointment.getAppointmentCancellationInfo({appointment: <appointment>});

HTTP Method

GET

Url

/patientportalapi/appointment/cancellation-info

URL Parameters

appointment	string	The key of the existing appointment provided by the API upon GetExistingAppointments.

Return

AppointmentCancellationData

9.13 CancelExistingAppointment

Cancels the existing appointment and returns an invoice if any cancellation fee applies.

JavaScript library method

patientportal.appointment.cancelExistingAppointment({appointment: <appointment>});

HTTP Method

GET

Url

/patientportalapi/appointment/cancel

URL Parameters

appointment	string	The key of the existing appointment provided by the API upon GetExistingAppointments.

Return

InvoiceData

Remarks

Returns null if no cancellation fee was applied.

9.14 GetQuestionnairesForFutureAppointments

Gets questionnaires for future appointments for logged in patient. Gets complete, incomplete and partially complete questionnaires.

JavaScript library method

patientportal.appointment.getQuestionnairesForFutureAppointments();

HTTP Method

GET

Url

/patientportalapi/appointment/future-questionnaires

Returns

QuestionnaireOverviewData[]

Remarks

Overview data doesn't contain definitions of questionnaires (questionnaire markup). The client has to call GetQuestionnaireDetail to retrieve it.

9.15 GetQuestionnaires

Gets all questionnaires for future appointments based on filters. Gets complete, incomplete and partially complete questionnaires.

JavaScript library method

```
patientportal.appointment.getQuestionnaires ({
        patient: <patient>,
        questionnaireTypeName: <questionnaire-type-name>,
        expirationFrom: <expiration-from>,
        expirationTo: < expiration-to>,
        status: <status>,
        appointment: <appointmentId>,
        pageSortColumn: <page-sort-column>,
        pageSortDescending: <page-sort-descending>,
        pageNumber: <page-number>,
        pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/appointment/questionnaires

URL Parameters

patient	string (optional)	The key of the patient provided by the API upon section Patients. Default is undefined which returns questionnaires for all patients.
questionnaire-type-name	string (optional)	The questionnaire type name filter. Optional if appointment id is provided
expiration-from	DateTime (optional)	The questionnaire start date filter.
expiration-to	DateTime (optional)	The questionnaire start date filter.
status	int (optional)	Status code of the questionnaire: - 0 = Incomplete - 1 = Complete - 2 = Partially Complete - -1 = all Default: -1
appointment	Int (optional)	Appointment id to get questionnaires of a booked appointment. Optional if questionnaire name is provided.
page-sort-column	int (optional)	The column index to sort the result: - 0 –Expiration date - 1 – Status Default: 0
page-sort-descending	int (optional)	True to sort result descending. Default false.
page-number	int (optional)	Required page number. Default 1.
page-size	int (optional)	Required page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of QuestionnaireOverviewData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10,
  "SortColumn": 0,
  "SortDescending": false
}
```

Remarks

Overview data doesn’t contain definitions of questionnaires (questionnaire markup). The client has to call GetQuestionnaireDetail to retrieve it.

9.16 GetQuestionnaireDetail

Gets detail of the specific questionnaire (definitions of questions and saved patient's answers).

JavaScript library method

patientportal.appointment.getQuestionnaireDetail({questionnaire: <questionnaire>});

HTTP Method

GET

Url

/patientportalapi/appointment/questionnaire-detail

URL Parameters

questionnaire	string	The key of the questionnaire provided by the API upon GetQuestionnaires.

Returns

QuestionnaireData

9.17 SaveQuestionnaire

Saves partial answers of specific questions in the questionnaire.

JavaScript library method

```
patientportal.appointment.saveQuestionnaire({
        questionnaire: <questionnaire>,
        answers: <answers>
});
```

HTTP Method

POST

Url

/patientportalapi/appointment/save-questionnaire

POST Parameters

questionnaire	string	The key of the questionnaire provided by the API upon GetQuestionnaires.
answers	QuestionnaireAnswerData[]	Collection of answers.

Remarks

The client can call this method as many time as it needs. Only answers attached in parameter *Answers* will be updated. The sort-order of questions isn't important.

After the first call of this method the questionnaire's status changes to *Partially Complete*.

The client must call SubmitQuestionnaire after all required questions are answered.

9.18 SubmitQuestionnaire

Submits the specific questionnaire.

JavaScript library method

```
patientportal.appointment.submitQuestionnaire({questionnaire: <questionnaire>});
```

HTTP Method

GET

Url

/patientportalapi/appointment/submit-questionnaire

URL Parameters

questionnaire	string	The key of the questionnaire provided by the API upon GetQuestionnaires.

Remarks

The client calls this method when all required questions are answered. After submitting the questionnaire the status will change to *Complete*. The questionnaire is still accessible using GetQuestionnaires until the appointment starts (in real-life).

When any required question is not answered the exception with event code 40001 is thrown (see Error handling).

9.19 SaveAndSubmitQuestionnaire

Combination of SaveQuestionnaire and SubmitQuestionnaire. This method can be used on the latest questionnaire page where you may save the answers on the latest page and submit the questionnaire at once.

JavaScript library method

```
patientportal.appointment.saveAndSubmitQuestionnaire({
        questionnaire: <questionnaire>,
        answers: <answers>
});
```

HTTP Method

POST

Url

/patientportalapi/appointment/save-submit-questionnaire

URL Parameters

questionnaire	string	The key of the questionnaire provided by the API upon GetQuestionnaires.
answers	QuestionnaireAnswerData[]	Collection of answers.

Remarks

The client calls this method when all required questions are answered. After submitting the questionnaire the status will change to *Complete*. The questionnaire is still accessible using GetQuestionnaires until the appointment starts (in real-life).

When any required question is not answered the exception with event code 40001 is thrown (see Error handling).

9.20 GetProposedTimeSlots

Gets proposed time slots according to given filters.

JavaScript library method

```
patientportal.appointment.getProposedTimeSlots({
        appointmentType: <appointment-type>,
        clinician: <clinician>,
        clinicianSex: <clinician-sex>,
        site: <site>,
        location: <location>,
        fromDate: <from-date>,
        toDate: <to-date>,
        timePreference: <time-preference>,
        timeFrom: <time-from>,
        timeTo: <time-to>,
        payerType: <payer-type>,
        patient: <patient>,
        referral: <referral>,
        recall: <recall>,
        modules: <modules>,
        services: <services>
});
```

HTTP Method

GET

Url

/patientportalapi/appointment/proposed-time-slots

URL Parameters

appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes.
clinician	int (optional)	Clinician filter. Identifier provide by the API upon GetClinicians. Allow 0 for any clinician.
clinician-sex	int (optional)	Clinician’s gender filter: - 0 = any sex - 1 = male - 2 = female
site	int (optional)	Site filter. Identifier provide by the API upon GetSites. Allow 0 for any site.
location	int (optional)	Location filter. Identifier provide by the API upon GetSites. Allow 0 for any location.
from-date	LocalDateTime (optional)	Filters appointments on or after provided date/time (inclusive). The parameter is a local date/time on a site. If you search across multiple sites in different time zones the search looks for the local date/time at each site.
to-date	LocalDateTime	The end date/time for the search (exclusive).
time-preference	int (optional)	Preferred time filter: - 0 = any time - 1 = Morning 00:00 – 12:00 - 2 = Afternoon 12:00 – 00:00 Note this parameter is ignored if any from *time-from* or *time-to* are provided.
time-from	string “HH:mm” (optional)	Time filter for the appointment start time. E.g 09:30.
time-to	string “HH:mm” (optional)	Time filter for the appointment finish time. E.g 11:00.
payer-type	string (optional)	Type of the payer provided by the API upon GetPayerTypes.
patient	string (optional)	The key of the patient provided by the API upon section Patients. Used to book an appointment for a different patient within your company. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used to book an appointment for a specific recall.

POST Parameters

modules	AppointmentModuleData[] (optional)	Selection of modules and additional services provided by the API upon GetAppointmentTypes.
services	ServiceData[] (optional)	Selection of services provided by GetServices

Returns

TimeSlotsData

Remarks

The server sorts the result by day and time of the appointment (the closer appointment first).

9.21 BookProposedTimeSlot

Books and returns the new appointment from the given time slot. Returned appointment contains the amount owed by the patient.

JavaScript library method

```
patientportal.appointment.bookProposedTimeSlot({
        proposedTimeSlot: <proposedTimeSlot> ,
        clinician: <clinician>,
        clinicianSex: <clinicianSex>,
        method: <method>,
        price: <price>,
        patient: <patient>,
        referral: <referral>,
        recall: <recall>
});
```

HTTP Method

POST

Url

/patientportalapi/appointment/book-time-slot

URL Parameters

patient	string (optional)	The key of the patient provided by the API upon section Patients. Used to book an appointment for a different patient within your company. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used to book an appointment for a specific recall.

POST Parameters

proposedTimeSlot	AppointmentData	The proposed time slot data, which can be constructed using the data provided by the GetProposedTimeSlots.
clinician	int (optional)	Clinician filter. Identifier provide by the API upon GetClinicians. Allow 0 for any clinician.
clinicianSex	int (optional)	Clinician’s gender filter: - 0 = any sex - 1 = male - 2 = female
method	int (optional)	The method that will be used to determine which slot to select if multiple are found that match the given input criteria: - 0 = random
price	decimal (optional)	Limit the bookable slots to only those that match the given price exactly

POST data example

```
{
  "proposedTimeSlot": {
    "Location": {
      "Key": 214
    },
    "Modules": [
      {
        "Key": 10382
      }
    ],
    "PayerType": "IN",
    "Services": [],
    "Site": {
      "Key": 1123
    },
    "Start":"2025-01-01T09:30:00",
    "Type": {
      "Key": "00000000fc0af8e9"
    }
  },
  "clinicianSex": 1,
  "method": 0,
  "price": 100
}
```

Returns

AppointmentData

Remarks

If no appropriate slot can be found for the proposed time slot, for example it was booked in the meantime, then an error is returned, see Error handling.

10 Anonymous Appointment Search

This section provides methods for making anonymous (i.e. not authenticated) appointment searches. The methods are deliberately similar to the appointment search API documented in section 9 'Appointments'.

Each method in this section needs the client to provide the *ASP.NET_SessionId* key in a cookie (see Login).

10.1 GetAppointmentTypes

Gets appointment types (e.g. Consultation, Health Screen, etc.) available for booking.

JavaScript library method

```
patientportal.anonAppointment.getAppointmentTypes({
        payerType: <payer-type>
});
```

HTTP Method

GET

Url

/patientportalapi/anon-appointment/types

URL Parameters

payer-type	string	The signup code of a chargeband, that will provide eligibility and price information for the search.

Returns

AppointmentTypeData[]

Remarks

If no module is returned for the appointment type, it doesn't contain modules and the booking process can continue without selecting the module.

10.2 GetSites

Gets possible sites for the specific appointment type. A site is a geographically-unique place. Each site may include additional 'locations', which are typically rooms at the site (Room 123, Surgery, etc.), but may also be geographically different places associated with the site (First floor, West wing, Building 3, etc.).

The API will only present you with locations which mean something to a user. So they're unlikely to get a choice about Room 1, 2, 3, etc. but they may get a choice about Site: My Bank, Location: Floor 1, Floor 50, Floor 100, etc.

JavaScript library method

```
patientportal.anonAppointment.getSites({
        appointmentType: <appointment-type>,
        lat: <lat>,
        long: <long>,
        payerType: <payer-type>
});
```

HTTP Method

GET

Url

/patientportalapi/anon-appointment/sites

URL Parameters

appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes.
lat long	decimal (optional)	Latitude and Longitude from GPS so server can sort sites by distance. Latitudes and Longitudes are defined using numerals that have a precision to 6 decimal places. For example, “lat=51.541743&long=-0.13715" is a valid value.
payer-type	string	The signup code of a chargeband, that will provide eligibility and price information for the search.

POST Parameters

clinicians	int[] (optional)	Clinicians filter. Array of identifiers provide by the API upon GetClinicians. Null or empty for any clinicians.
modules	AppointmentModuleData [] (optional)	Selection of modules and additional services provided by the API upon GetAppointmentTypes. Available sites will be filtered according to availability of the specified modules.

Returns

ShallowSearchResultData

A shallow search result of a patient/company address.

Properties

Key	string	An encrypted key value containing partial address information used to retrieve complete address details.
Address	string	A single-line search result address.

JSON Example

```
{
  "Key":
"fc560b40315dc7605fd5ca53e0dcaabc357c69bea3faefa8c6e2ce8129909061956910b77338ee2c2
cdbb1c7c5f7c64bcf338d78bc148f81f6786152d3ef2987b3ab5b1e5588b1db7939bb5e0edffec4614
c4511c4a7a0dfd9bc9077749482b152217c572b0f78552c75be542ffcea6446110af6da78213c1f715
69f35abab7d65f82f382f8b8dc663c8e6a1405bf17c331d379f375ffbc6ec3ebc21a985a69355d1062
2db48eceb7f23b38c5037ed2315c3d858268baae1879f6f84b3b65586742086832ec398acdfd56680a
72991d7bb38bbfd1fa61991aebf0bd1982dc06b",
  "Address": " 2 Paradise Street, Liverpool, Merseyside, L1 8JF"
}
```

10.3

SiteData[]

10.4 GetClinicians

Gets possible clinicians/doctors for the specific appointment type.

JavaScript library method

```
patientportal.anonAppointment.getClinicians({
            appointmentType: <appointment-type>,
            sites: <sites>,
            locations: <locations>,
            payerType: <payer-type>,
            patient: <patient>,
            referral: <referral>,
            recall: <recall>
});
```

HTTP Method

POST

Url

/patientportalapi/anon-appointment/clinicians

URL Parameters

appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes.
payer-type	string	The signup code of a chargeband, that will provide eligibility and price information for the search.

POST Parameters

sites	int[] (optional)	Sites filter. Array of identifiers provide by the API upon GetSites. Null or empty for any sites.
locations	int[] (optional)	Locations filter. Array of identifiers provide by the API upon GetSites. Null or empty for any locations.
modules	AppointmentModuleData [] (optional)	Selection of modules and additional services provided by the API upon GetAppointmentTypes. Available clinicians will be filtered according to availability of the specified modules.

Returns

ClinicianData[]

10.5 GetServiceTypes

Gets service types (except) modules

JavaScript library method

```
patientportal.anonAppointment.getServiceTypes ({
            payerType: <payer-type>,
            appointmentType: <appointment-type
});
```

HTTP Method

GET

Url

/patientportalapi/anon-appointment/service-types

URL Parameters

payer-type	string	Type of the payer provided by the API upon GetPayerTypes.
appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes

Returns

ServiceTypeData[]

Remarks

If no service types are returned for the appointment type, either the appointment type does not allow additional services or no services exist in self book rule.

10.6 GetServices

Gets available services

JavaScript library method

```
patientportal.anonappointment.getServices ({
            payerType: <payer-type>,
            appointmentType: <appointment-type>,
            serviceType: <service-type>,
            serviceName:<service-name>,
            pageSortColumn: <page-sort-column>,
            pageSortDescending: <page-sort-descending>,
            pageNumber: <page-number>,
            pageSize: <page-size>

});
```

HTTP Method

GET

Url

/patientportalapi/anon-appointment/services

URL Parameters

payer-type	string	Type of the payer provided by the API upon GetPayerTypes.
appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes
service-type	string (optional)	Key of the service type to filter services by, provided by GetServiceTypes
service-name	string (optional)	Name of service to filter services by.
page-sort-column	int (optional)	The column index to sort the result: - 0 – Name - 1 – Code - 2- Service Type - 3- ServiceId Default: 0,2
page-sort-descending	int (optional)	True to sort result descending. Default false.
page-number	int (optional)	Required page number. Default 1.
page-size	int (optional)	Required page size. Default 10. Minimum 5. Maximum 50.

Returns

ServiceData[]

Remarks

If no services are returned for the appointment type, either the appointment type does not allow additional services or no services exist in self book rule.

10.7 GetProposedAppointments

Gets proposed appointments according given filters.

JavaScript library method

```
patientportal.appointment.getProposedAppointments({
        appointmentType: <appointment-type>,
        clinician: <clinician>,
        clinicianSex: <clinician-sex>,
        site: <site>,
        location: <location>,
        fromDate: <from-date>,
        toDate: <to-date>,
        timePreference: <time-preference>,
        timeFrom: <time-from>,
        timeTo: <time-to>,
        modules: <modules>,
        payerType: <payer-type>,
        numberOfResults: <number-of-results>
});
```

HTTP Method

POST

Url

/patientportalapi/appointment/proposed

URL Parameters

appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes.
clinician	int (optional)	Clinician filter. Identifier provide by the API upon GetClinicians. Allow 0 for any clinician.
clinician-sex	int (optional)	Clinician's gender filter: - 0 = any sex - 1 = male - 2 = female
site	int (optional)	Site filter. Identifier provide by the API upon GetSites. Allow 0 for any site.
location	int (optional)	Location filter. Identifier provide by the API upon GetSites. Allow 0 for any location.
from-date	LocalDateTime (optional)	Filters appointments on or after provided date/time (inclusive). The parameter is a local date/time on a site. If you search across multiple sites in different time zones the search looks for the local date/time at each site.
to-date	LocalDateTime (optional)	The end date/time for the search (exclusive). If not provided the API is looking for days/months into the future to satisfy the requested *number-of-results*. This may take some time. If you need to get slots for a specific day only it is recommended to provide *to-date* to make the request quick.
time-preference	int (optional)	Preferred time filter: - 0 = any time - 1 = Morning 00:00 – 12:00 - 2 = Afternoon 12:00 – 00:00 Note this parameter is ignored if any from *time-from* or *time-to* are provided.
time-from	string "HH:ss" (optional)	Time filter for the appointment start time. E.g 09:30.
time-to	string "HH:ss" (optional)	Time filter for the appointment finish time. E.g 11:00.
from-date	DateTime (optional)	Filters appointments on or after this date. Note: Only day is accepted. Time is ignored.
time-preference	int (optional)	Preferred time filter: - 0 = any time - 1 = Morning - 2 = Afternoon
payer-type	String (Optional)	The signup code of a chargeband, that will provide eligibility and price information for the search.
number-of-results	Int (optional)	Required number of proposed slots. Value must be in a range 1 – 100.

POST Parameters

modules	AppointmentModuleData[] (optional)	Selection of modules and additional services provided by the API upon GetAppointmentTypes.
services	ServiceData[] (optional)	Selection of services provided by GetServices

Returns

AppointmentData[]

Remarks

The server sorts the result by day and time of the appointment (the closer appointment first).

10.8 GetProposedTimeSlots

Gets proposed time slots according to given filters.

JavaScript library method

```
patientportal.anonAppointment.getProposedTimeSlots({
	appointmentType: <appointment-type>,
	clinician: <clinician>,
	clinicianSex: <clinician-sex>,
	site: <site>,
	location: <location>,
	fromDate: <from-date>,
	toDate: <to-date>,
	timePreference: <time-preference>,
	timeFrom: <time-from>,
	timeTo: <time-to>,
	payerType: <payer-type>,
	patient: <patient>,
	referral: <referral>,
	recall: <recall>,
	modules: <modules>,
	services: <services>
});
```

HTTP Method

GET

Url

/patientportalapi/anon-appointment/proposed-time-slots

URL Parameters

appointment-type	string	Type of the appointment provided by the API upon GetAppointmentTypes.
clinician	int (optional)	Clinician filter. Identifier provide by the API upon GetClinicians. Allow 0 for any clinician.
clinician-sex	int (optional)	Clinician's gender filter: - 0 = any sex - 1 = male - 2 = female
site	int (optional)	Site filter. Identifier provide by the API upon GetSites. Allow 0 for any site.
location	int (optional)	Location filter. Identifier provide by the API upon GetSites. Allow 0 for any location.
from-date	LocalDateTime (optional)	Filters appointments on or after provided date/time (inclusive). The parameter is a local date/time on a site. If you search across multiple sites in different time zones the search looks for the local date/time at each site.
to-date	LocalDateTime	The end date/time for the search (exclusive).
time-preference	int (optional)	Preferred time filter: - 0 = any time - 1 = Morning 00:00 – 12:00 - 2 = Afternoon 12:00 – 00:00 Note this parameter is ignored if any from *time-from* or *time-to* are provided.
time-from	string "HH:mm" (optional)	Time filter for the appointment start time. E.g 09:30.
time-to	string "HH:mm" (optional)	Time filter for the appointment finish time. E.g 11:00.
payer-type	string (optional)	The signup code of a chargeband, that will provide eligibility and price information for the search.
patient	string (optional)	The key of the patient provided by the API upon section Patients. Used to book an appointment for a different patient within your company. Default is the logged in patient.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals. Used to book an appointment for a specific referral.
recall	string (optional)	The key of the recall provided by the API upon method GetRecalls. Used to book an appointment for a specific recall.

POST Parameters

modules	AppointmentModuleData[] (optional)	Selection of modules and additional services provided by the API upon GetAppointmentTypes.
services	ServiceData[] (optional)	Selection of services provided by GetServices

Returns

TimeSlotsData

Remarks

The server sorts the result by day and time of the appointment (the closer appointment first).

11 Anonymous Appointment Booking

The previous section Anonymous Appointment Search is used to find an available slot. This section allows the patient to book an appointment quickly without dealing with the standard 5.10 RegisterPatient process.

The workflow is:

- find an available slot via GetProposedAppointments
- register the patient with minimum demographics via 12.3 RegisterPatient
    - use *SessionId* for further communication
- verify 2FA (ResendValidationCode, SubmitValidationCode)
- book an appointment via BookProposedAppointment

Note that this workflow does not log the patient in. Once 2FA/booking completes you cannot use

12 Address Finder

This section provides services for searching and retrieving address details. Currently, only the UK is supported.

12.1 Search

Returns a number of possible shallow search address results for the postcode.

JavaScript library method

patientportal.addressFinder.search ({postcode: <postcode>});

HTTP Method

GET

Url

/patientportalapi/address-finder/search

URL Parameters

postcode	string	Case-insensitive postcode with space (e.g. N7 0NH or n7 0nh).

Returned JSON

Shallow Search Result Data[]

12.2 Address

Returns address details for the provided encrypted key.

JavaScript library method

patientportal.addressFinder.address ({key: <key>});

HTTP Method

GET

Url

/patientportalapi/address-finder/address

URL Parameters

key	string	An encrypted key value containing partial address information.

Returned JSON

AddressData

Patient section (or any other) to gather details about the patient.

It is recommended to provide *Login* button within the UI so the user could login if they already have an account. This way you avoid unnecessary duplicate records that could be created in the system.

12.3 RegisterPatient

Prepares a new registration. The patient is not fully created within the system until the appointment is booked.

JavaScript library method

```
patientportal.anonBooking.registerPatient({
        payerType: <payerType>,
        demog: <demog>
});
```

HTTP Method

POST

Url

/patientportalapi/anon-booking/register-patient

POST Parameters

payerType	String (Optional)	The signup code of a chargeband, that will provide eligibility and price information for the search. This is the same payer type like the one provided into the slot search GetProposedAppointments
demog	PersonDemographicData	Person demographics. Mandatory fields are: - Name - Surname - EmailAddress - Mobile - DateOfBirth ○ based on *DateOfBirthRequiredForPatients* in GetConfig

Returned JSON

```
{
  "SessionId": "R9I8Dr8rGQcQ6EOfxVyOBX2r8zt4KTO06vW0O2+GyFc="
}
```

Remarks

A 2FA email and SMS message are automatically sent to the patient.

12.4 ResendValidationCode

Resends the 2FA email and SMS. Please use the *SessionId* provide within registration and provide the *ASP.NET_SessionId* key in a cookie.

JavaScript library method

patientportal.anonBooking.resendValidationCode();

HTTP Method

POST

Url

/patientportalapi/anon-booking/resend-validation-code

12.5 SubmitValidationCode

Submits the 2FA code and allows the booking process to continue.

JavaScript library method

patientportal.anonBooking.submitValidationCode({ code: <code> });

HTTP Method

POST

Url

/patientportalapi/anon-booking/submit-validation-code

POST Parameters

code	code	The 2FA code.

12.6 BookProposedAppointment

Creates the patient in our system and books an appointment. Returns the new appointment.

JavaScript library method

```
patientportal.anonBooking.bookProposedAppointment({
        proposedAppointment: <proposedAppointment>
});
```

HTTP Method

POST

Url

/patientportalapi/anon-booking/book

POST Parameters

proposedAppointment	AppointmentData	The proposed appointment provided by the API upon GetProposedAppointments

Returns

AppointmentData

Remarks

If the appropriate appointment is taken by another user in a meanwhile the exception is throw. See Error handling.

12.7 BookProposedTimeSlot

Creates the patient in our system and books an appointment from the time slot. Returns the new appointment.

JavaScript library method

```
patientportal.anonBooking.bookProposedTimeSlot ({
        proposedTimeSlot: <proposeTimeSlot>,
        clinician: <clinician>,
        clinicianSex: <clinicianSex>,
        method: <method>,
        price: <price>
});
```

HTTP Method

POST

Url

/patientportalapi/anon-booking/book-time-slot

POST Parameters

proposedTimeSlot	AppointmentData	The proposed time slot data, which can be constructed using the data provided by the GetProposedTimeSlots.
clinician	int (optional)	Clinician filter. Identifier provide by the API upon GetClinicians. Allow 0 for any clinician.
clinicianSex	int (optional)	Clinician's gender filter: - 0 = any sex - 1 = male - 2 = female
method	int (optional)	The method that will be used to determine which slot to select if multiple are found that match the given input criteria: - 0 = random
price	decimal (optional)	Limit the bookable slots to only those that match the given price exactly

POST data example

```
{
  "proposedTimeSlot": {
    "Location": {
      "Key": 214
    },
    "Modules": [
      {
        "Key": 10382
      }
    ],
    "PayerType": "example",
    "Services": [],
    "Site": {
      "Key": 1123
    },
    "Start":"2025-01-01T09:30:00",
    "Type": {
      "Key": "00000000fc0af8e9"
    }
  },
  "clinicianSex": 1,
  "method": 0,
  "price": 100
}
```

Returns

AppointmentData

Remarks

If no appropriate slot can be found for the proposed time slot, for example it was booked in the meantime, then an error is returned, see Error handling.

13 Finance

This section provides methods for working with invoices.

Each method in this section needs the client to provide the *ASP.NET_SessionId* key in a cookie (see Login).

13.1 GetInvoices

Returns a number of invoices according given filters.

JavaScript library method

```
patientportal.finance.getInvoices({
            toDate: <to-data>,
            paid: <paid>
});
```

HTTP Method

GET

Url

/patientportalapi/finance/invoices

URL Parameters

to-date	DateTime (optional)	Returns N invoices leading up to this date. N is server defined, but is likely to be between 10 - 20. Use this to implement infinite-scroll behaviour by passing the oldest date in the returned InvoiceData[] to subsequent calls to GetInvoices. This also makes it easier to implement classic timeline based features of jumping to years, months, etc.
paid	bool (optional)	True to get only paid invoices, false to get only unpaid and partially paid invoices, null (not present) to get all types of invoices (paid, partially paid and unpaid.

Returns

InvoiceData[]

Remarks

If no optional (filter) parameters are present the server returns all unpaid and partially paid invoices and a number of paid invoices.

13.2 GetInvoiceDetail

Returns an invoice.

JavaScript library method

```
patientportal.finance.getInvoiceDetail({
        invoice: <invoice>,
        invoiceNumber: <invoice-number>
});
```

HTTP Method

GET

Url

/patientportalapi/finance/invoice-detail

URL Parameters

invoice	string (partially optional)	The key of the invoice provided by the API upon GetInvoices.
invoice-number	string (partially optional)	The invoice number provided by the API upon GetInvoices.

Returns

InvoiceData[]

Remarks

One of optional parameters *invoice* or *invoice-number* must be used.

13.3 SendInvoiceByEmail

Sends the specific invoice to the patient’s email address.

JavaScript library method

```
patientportal.finance.sendInvoiceByEmail({invoiceNumber: <invoice-number>});
```

HTTP Method

GET

Url

/patientportalapi/finance/send-by-email

URL Parameters

invoice-number	string	The invoice number provided by the API upon GetInvoices.

Remarks

The client should show a warning message that sending an invoice over email isn't encrypted and that the patient should use this feature at their own risk. If online payment is active, the email contains a link to the portal to pay online for unpaid invoices. The link is in the following format:

https://{portal}/pay?invoiceKey={encrypted invoice key}

The invoice key can be used in logged-in or anonymous mode to process payment using ProvidePayment.

13.4 ProvidePayment

Provides online payment for the patient. Loads a payment frame within the provide iframediv to process the payment. The invoice corresponding to invoice key should have the 'PayableOnline' parameter true (retrieved through GetInvoiceDetail function.)

JavaScript library method

```
patientportal.finance.providePayment({
        invoiceKey: <invoice-key>,
        payerAccountId: <payer-accountid>,
        saveDetails: <save-payment-details>,
        description: <description>,
        iframeDiv: <iframediv>,
        firstname: <firstname>,
        surname: <surname>,
        address1: <address1>,
        address2: <address2>,
        city: <city>,
        postcode: <postcode>,
        country: <country>,
        phone: <phone>,
        email: <email>});
```

HTTP Method

POST

Url

/patientportalapi/payment-gateway/create-payment

URL Parameters

invoice-key	int	Encrypted Key of the invoice provided by the API.
payer-accountid	int	Id of the existing card details for the patient retrieved using PayerAccounts. 0 if new card details are to be used. This is only valid for logged in sessions.
save-payment-details	bool	If the user wants to save the card details for future use, only considered if the user is logged-in.
description	string	Description of the payment: 100 characters
iframediv	string	Control id for the div to load iframe in. The javascript will load create the iframe control and append it to the provided div.
PayerAccountData	PayerAccountInputData	This data is not required if the system provides a valid 'payer-account-id'. These are only used if the 'payer-account-id' is '0'.

Remarks

The server validates all the input parameters and if any of the validation fails, an error is returned in the 'on fail' call-back. On successful validation, an iframe is loaded in the provided iframediv control to process the payment, and a new callback is returned in the OnDone callback. Once the user has processed the iframe, result of payment is returned via the OnDone (if payment success) or OnFail(if payment failed) of the new callback:

Following data is included in the call-back:

OnDone:

```
        Sender: “PAYMENT-GATEWAY”,
        Status: “OK”,
        StatusDetail: Details of the payment status,
        TransactionId: Transaction id of the payment - GUID,
        ClientPaymentId: Identifier of the payment
```

OnFail:

```
        Sender: “PAYMENT-GATEWAY”,
        Status: failure status,
        StatusDetail: Details of the payment failure,
        TransactionId: Transaction id of the payment - GUID,
```

13.5 PayerAccounts

Returns a list of existing stored cards for the current logged in person, which they can use for a new payment. Returns an empty list, if no stored cards exist for the logged in person (payer).

JavaScript library method

patientportal.finance.prototype.payerAccounts(invoiceKey: <invoice-key>)

HTTP Method

GET

Url

/patientportalapi/payment-gateway/payer-accounts

URL Parameters

invoice-key	int	Encrypted Key of the invoice provided by the API. (Optional). If provided, the payer accounts are filtered by the portal payment type for that invoice. This should always be provided for online payments, and removed for payer account management.

Returns

PayerAccountData[]

13.6 RemovePayerAccount

Removes (disables) an existing stored card for the current logged in person.

JavaScript library method

patientportal.finance.prototype.removePayerAccount ({

```
        payerAccountKey: <payer-account-key>
})
```

HTTP Method

GET

Url

/patientportalapi/payment-gateway/remove-payer-account

URL Parameters

payer-account-key	int	Payer Account Key of the account to be disabled as retrieved from PayerAccounts

Returns

bool

13.7 PaymentCountries

Returns a list of countries that may be used for the billing address country when making a payment.

JavaScript library method

patientportal.finance.prototype.getPaymentCountries()

HTTP Method

GET

Url

/patientportalapi/payment-gateway/payment-countries

Returns

CountryData[]

14 Feeds

This section provides methods for working with message feeds.

Each method in this section needs the client to provide the *ASP.NET_SessionId* key in a cookie (see Login).

14.1 GetFeeds

Gets a number of the newest feeds.

JavaScript library method

```
patientportal.feed.getFeeds({
        toDate: <to-date>,
        referral:<referral>
});
```

HTTP Method

GET

Url

/patientportalapi/feed/feeds

URL Parameters

to-date	DateTime (optional)	Returns N feeds leading up to this date. N is server defined, but is likely to be between 10 - 20. Use this to implement infinite-scroll behaviour by passing the oldest date in the returned FeedData[] to subsequent calls to GetFeeds. This also makes it easier to implement classic timeline based features of jumping to years, months, etc.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals.

Returns

FeedData[]

Remarks

If *to-date* is not used the server returns a number of the newest feeds but not all patient’s feeds.

14.2 GetFeed

Gets a specific feed.

JavaScript library method

```
patientportal.feed.getFeed({
        feed: <feed>,
        referral:<referral>
});
```

HTTP Method

GET

Url

/patientportalapi/feed/feed

URL Parameters

feed	string	Key of the feed provided by the API upon GetFeeds.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals.

Returns

FeedData

14.3 AddFeed

Adds a new feed to the patient’s feeds and returns it.

JavaScript library method

```
patientportal.feed.addFeed({
        messageText: <messageText>,
        referral:<referral>
});
```

HTTP Method

POST

Url

/patientportalapi/feed/add-feed

POST Parameters

messageText	string	Text of the first message within a new feed. Text can be formatted as HTML or plain text.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals.

Returns

FeedData

14.4 AddMessage

Adds a new message to the existing feed.

JavaScript library method

```
patientportal.feed.addMessage({
	feed : <feed>,
	messageText: <messageText>,
	referral:<referral>
});
```

HTTP Method

POST

Url

/patientportalapi/feed/add-message

POST Parameters

feed	string	Key of the feed provided by the API upon GetFeeds.
messageText	string	Text of the new message. Text can be formatted as HTML or plain text.
referral	string (optional)	The key of the referral provided by the API upon GetReferrals.

15 Notifications

This section provides methods for working with notifications.

Each method in this section needs the client to provide the *ASP.NET_SessionId* key in a cookie (see Login).

15.1 GetNotifications

Returns notifications about appointments, invoices, questionnaires etc.

JavaScript library method

patientportal.notification.getNotifications({type: <type>});

HTTP Method

GET

Url

/patientportalapi/notification/notifications

URL Parameters

type	int (optional)	The type of the notification the client requests: - 1 = Appointments that coming up soon - 2 = Outstanding questionnaires - 4 = Outstanding invoices - 8 = Outstanding messages - 16 = Action required - 32 = Referral notification - 64 = Membership payment required Note: The client can use a combination (sum) of several types. For example 1+4=5 returns notifications about appointments that coming up soon and outstanding invoices. Default value is 95 (all but referral notifications)

Returns

NotificationData[]

15.2 ConfirmNotification

Confirms notification so the notification will not be returned by GetNotifications further.

JavaScript library method

patientportal.notification.confirm({notification: <notification>});

HTTP Method

GET

Url

/patientportalapi/notification/confirm

URL Parameters

notification	string	Key of the notification provided by the API upon GetNotifications.

Remarks

Only certain notification types are allowed to confirm. Types that are allowed by current version of the API are:

- 8 = Outstanding Messages

16 Membership scheme

This section provides methods for working with membership scheme.

16.1 ValidateCode

Confirms the validity of the membership scheme code.

JavaScript library method

patientportal.membershipScheme.validateCode({code: <code>});

HTTP Method

GET

Url

/patientportalapi/membership-scheme/validate-code

URL Parameters

code	string	Membership scheme code that is provided by Medical Management Systems to the client.

Remarks

If the code is valid, HTTP status code is 2xx. Otherwise check exception (see Error handling).

16.2 GetSchemeInfo

Returns information for the membership scheme (name, billing frequency, price, etc.).

JavaScript library method

patientportal.membershipScheme.getSchemeInfo({code: <code>});

HTTP Method

GET

Url

/patientportalapi/membership-scheme/scheme-info

URL Parameters

code	string	Membership scheme code that is provided by Medical Management Systems to the client.

Returns

MembershipSchemeData

16.3 GetSchemes

Returns list of possible membership schemes the patient can join.

JavaScript library method

patientportal.membershipScheme.getSchemes();

HTTP Method

GET

Url

/patientportalapi/membership-scheme/schemes

Returns

MembershipSchemeData[]

16.4 GetCurrent

Returns the current membership scheme data the patient is currently joined into.

JavaScript library method

patientportal.membershipScheme.getCurrent();

HTTP Method

GET

Url

/patientportalapi/membership-scheme/current

Returns

CurrentMembershipSchemeData

16.5 Join

Joins the patient into the membership scheme.

JavaScript library method

patientportal.membershipScheme.join({code: <code>});

HTTP Method

GET

Url

/patientportalapi/membership-scheme/join

URL Parameters

code	string	Membership scheme code that is provided by Medical Management Systems to the client.

Remarks

If the patient is joined successfully then the HTTP status code is 2xx. Otherwise check the exception (see Error handling).

16.6 SetupOnlinePayment

Sets up an online recurring payment for the current membership scheme of the patient. Loads a payment frame within the provide iframediv to process the payment. The current membership of the patient GetCurrent should have the OnlinePaymentAllowed parameter set and Online Payments enabled for the chamber. If the membership's last invoice has a balance associated, the user is made to pay that invoice and the details of that transaction are used for future recurring payments. If the last invoice associated with the membership scheme does not have a balance, a penny transaction is performed which is immediately refunded. Details of this transaction are used for future recurring payments.

JavaScript library method

```
patientportal.membershipScheme.setupOnlinePayment ({
        payerAccountId: <payer-accountid>,
        saveDetails: <save-payment-details>,
        iframeDiv: <iframediv>,
        firstname: <firstname>,
        surname: <surname>,
        address1: <address1>,
        address2: <address2>,
        city: <city>,
        postcode: <postcode>,
        country: <country>,
        phone: <phone>,
        email: <email>});
```

HTTP Method

POST

Url

/patientportalapi/membership-scheme/setup-online-payment

URL Parameters

payer-accountid	int	Id of the existing card details for the patient retrieved using PayerAccounts. 0 if new card details are to be used.
save-payment-details	bool	If the user wants to save the card details for future use.
iframediv	string	Control id for the div to load iframe in. The javascript will load create the iframe control and append it to the provided div.
PayerAccountData	PayerAccountInputData	This data is not required if the system provides a valid ‘payer-account-id’. These are only used if the ‘payer-account-id’ is ‘0’.

Remarks

The server validates all the input parameters and if any of the validation fails, an error is returned in the ‘on fail’ call-back. On successful validation, an iframe is loaded in the provided iframediv control to process the payment, and a new callback is returned in the OnDone callback. Once the user has processed the iframe, result of payment is returned via the OnDone (if payment success) or OnFail(if payment failed) of the new callback:

Following data is included in the call-back:

OnDone:

```
        Sender: “PAYMENT-GATEWAY”,
        Status: “OK”,
        StatusDetail: Details of the payment status,
        TransactionId: Transaction id of the payment - GUID,
        ClientPaymentId: Identifier of the payment
```

OnFail:

```
        Sender: “PAYMENT-GATEWAY”,
        Status: failure status,
        StatusDetail: Details of the payment failure,
        TransactionId: Transaction id of the payment – GUID
```

16.7 RemoveOnlinePayment

Removes the recurring payment method from patient’s current membership scheme.

JavaScript library method

patientportal.membershipScheme.removeOnlinePayment();

HTTP Method

GET

Url

/patientportalapi/membership-scheme/remove-online-payment

17 Medical history

This section provides methods for reading the patient’s medical history.

The patient’s medical history is stored in the Medical History tree. This tree looks like:

```
/
    /appointment
        /appointment/122
        /appointment/526
            /appointment/526/conclusion
    /numeric-data
        /numeric-data/weight
        /numeric-data/height
        /numeric-data/BMI
    /document
        /document/5865
        /document/5996
```

Each node can consist from one or more another nodes. The tree is dynamically created by the server and it may change from time to time. The client shouldn't cache the tree for more than an hour at a time. The client should always request the root node and let patient browse through tree dynamically.

The client can split history-item paths on the forward-slash character to aid parent-node navigation.

Each method in this section needs the client to provide the *ASP.NET_SessionId* key in a cookie (see Login).

17.1 GetMedicalHistoryTreeNode

Return node from the Medical History tree. To understand how to read the patient's medical history read chapter Reading the Medical History tree.

JavaScript library method

```
patientportal.medicalHistory.getMedicalHistoryTreeNode({
            nodePath: <node-path>,
            sendByEmail: <send-by-email>
});
```

HTTP Method

GET

Url

/patientportalapi/medical-history/node/<node-path>

URL node path

node-path	string	The path of the node provided by the API upon GetMedicalHistoryTreeNode. If the path is specific the requested node is always fully loaded. If the path is empty the server returns root node. This parameter is not a query parameter. It is a part of the URL path. For example: The URL for the node 'numeric-data/weight': /patientportalapi/medical-history/node/numeric-data/weight

URL parameters

send-by-email	bool (optional)	If true the server sends the specific medical history item to the patient's email address. The client should show a warning message that sending medical information over email isn't encrypted and that the patient should use this feature at their own risk. For example: The URL for the node 'numeric-data/weight' where the client wants to send the medical history item to the patient's email address: /patientportalapi/medical-history/node/numeric-data/weight?send-by-email=true

Returns

MedicalHistoryNodeData

Remarks

If the specific node doesn't exist the client gets exception where event code is 50001. The client should go back to the parent node.

17.2 Reading the Medical History tree

The client needs to call GetMedicalHistoryTreeNode first time without parameters to get a root node. The root node may or may not contains all nodes of tree. It always contains top level nodes and may contains children of some nodes.

The server doesn’t always return all nodes fully loaded to ensure the response is not excessively large or for performance reasons. If the client needs to show any nodes that is not fully loaded the client needs to call GetMedicalHistoryTreeNode with requested node path.

Example

The first client request is:

URL: /patientportalapi/medical-history/node

JavaScript: patientportal.medicalHistory.getMedicalHistoryTreeNode();

The response (root node) looks like

```
{
  "DataType": 1,
  "Label": "Medical History",
  "Path": "/",
  "Loaded": true,
  "Children": [
    {
      "DataType": 1,
      "Label": "Medical examinations",
      "Details": "",
      "Path": "/appointment",
      "Loaded": true,
      "Children": [
        {
          "DataType": 1,
          "Label": "Full medical test",
          "Details": "25/05/2013",
          "Path": "/appointment/2556",
          "Loaded": false       < The server doesn’t return the whole node for
                               performance reasons (e.g. Children, Data fields are
                               not included). There will be a many child nodes
                               (Preface – Letter, Body Composition, etc.). To get
                               a full node the client must request this specific
                               node using the path ‘/appointment/2556’. >
        },
        {
          "DataType": 1,
          "Label": "Blood test",
          "Details": "16/04/2013",
          "Path": "/appointment/5526",
          "Loaded": true,       < The children contains only a few nodes so the
                               server returns it straight away. >
          "Children": [
            {
              "DataType": 2,
              "Loaded": true,
              "Data": "<p>Dear John, it was pleasure to see you...</p>"
            },
            {
              "DataType": 3,
              "Label": "Last blood test result",
              "Loaded": true,
              "Data": {
                "Size": "1256856",                  < The file size in bytes. >
                "MIMEType": "image/jpeg",
                "Url": "https://images.meddbase.com/image.jpg"
              }
            }
          ]
        }
     ]
    },
    {
      "DataType": 1,
      "Label": "Documents",
      "Details": "",
      "Path": "/document",
      "Loaded": true,
      "Children": [
        {
          "DataType": 2,
          "Label": "Pathology result",
          "Details": "Result from 04/07/2013. Requested by Dr. House",
          "Path": "/document/12345",
          "Loaded": false
        }
        ...< There is a lot of big HTML documents. The server doesn’t return it
             because it would be excessively large. The client should show only a
             list of documents to the patient and requests a specific document
             using the node’s ‘Path’ when the patient chooses one.   >...
      ]
    },
    {
      "DataType": 1,
      "Label": "Numerical data",
      "Details": "",
      "Path": "/num-data",
      "Loaded": true,
      "Children": [
        {
          "DataType": 4,
          "Label": "BMI Graph",
          "Details": "",
          "Path": "/num-data/12",
          "Loaded": false
        },
        {
          "DataType": 4,
          "Label": "Height",
          "Details": "",
          "Path": "/num-data/13",
          "Loaded": false
        },
    {
     "DataType": 4,
     "Label": "Width",
     "Details": "",
     "Path": "/num-data/14",
     "Loaded": false
    }
       ...< There is a lots of numerical data. The server doesn’t return
            fully loaded nodes because it would be excessively large and for
            performance reasons. The client should show only a list and let
            the patient choose one. The client use the node’s ‘Path’ to get
            graph’s data.   >...
   ]
  }
 ]
}
```

Let’s imagine the patient wants to see BMI graph. She chooses ‘Numerical data’ and after that she chooses ‘BMI Graph’. The client doesn’t have appropriate node fully loaded (Loaded=False) so it requests this node using the path ‘*/num-data/12*’. The request URL looks like:

Url: /patientportalapi/medical-history/node/num-data/12

JavaScript: patientportal.medicalHistory.getMedicalHistoryTreeNode({nodePath: “/num-data/12”});

The response from the server:

```
{
  "DataType": 4,
  "Label": "BMI graph",
  "Details": "",
  "Path": "/num-data/12",
  "Loaded": true,
  "Data": {
    "XAxis": {
      "Title": "Years",
      "Data": [2011, 2012, 2013]
    },
    "Series": [
      {
        "Title": "Sarah",
        "Data": [25.4, 30.4, 27.6]
      }
    ]
  }
}
```

If she wants to see full medical test the client use path ‘*/appointment/2556*’. The URL:

Url: /patientportalapi/medical-history/tree/appointment/2556

JavaScript: patientportal.medicalHistory.getMedicalHistoryTreeNode({nodePath: “/appointment/2556”});

The response from the server:

```
{
  "DataType": 1,
  "Label": "Full medical test",
  "Details": "25/05/2013",
  "Path": "/appointment/2556",
  "Loaded": true,
  "Children": [
    {
      "DataType": 1,
      "Label": "Preface - letter",
      "Details": "",
      "Path": "/appointment/2556/letter",
      "Loaded": false,
    },
    {
      "DataType": 1,
      "Label": "Body Composition",
      "Details": "",
      "Path": "/appointment/2556/body",
      "Loaded": false,
    },
    {
      "DataType": 1,
      "Label": "Cardiovascular Assessment",
      "Details": "",
      "Path": "/appointment/2556/ca",
      "Loaded": false,
    },
    {
      "DataType": 1,
      "Label": "Diabetes Assessment",
      "Details": "",
      "Path": "/appointment/2556/letter",
      "Loaded": false,
    }
  ]
}
```

If she wants to see the ‘Preface – letter’ the client use path ‘*/appointment/2556/letter*’. The URL:

Url: /patientportalapi/medical-history/tree/appointment/2556/letter

JavaScript: patientportal.medicalHistory.getMedicalHistoryTreeNode({nodePath: “/appointment/2556/letter”});

The response from the server:

```
{
  "DataType": 1,
  "Label": "Preface - letter",
  "Details": "",
  "Path": "/appointment/2556/letter",
  "Loaded": true,
  "Children": [
    {
      "DataType": 2,
      "Data": "<p>Date: 01/01/2013<br/>Subject: Medical Examination<br/>Ref/nr:
                ABC-1231243<br/><br/>Name: John<br/>Age: 30<br/>Gender: Male</p>
    },
    {
      "DataType": 2,
      "Data": "<p>Dear John</p><p></p><p>It was a pleasure to see you for your
                 medical examination on the 1st of January 2013 and as promised, I
                 enclose a copy of your results...</p>
    }
  ]
}
```

18 Referral Portal API

The Referral Portal is a chargeable feature that allows company's managers to:

- refer patients
- manager users (managers)
- access basic referral reports

The Referral Portal API extends the Patient Portal API with three new sections:

- Users – provides methods to manage managers
- Patients – provides methods to find patients or create new patients
- Referrals – provides methods to start a new referral, work with referral and read reports

The Referral Portal API also extends the existing Patient Portal API methods with new parameters (usually parameter *referral* and *patient* within sections *Appointments* and *Feeds*).

19 Users

This section provides methods for managing users. Only logged-in user which has got *CanManageUsers* right (see GetConfig) is able to work with this section.

19.1 GetAllowedTitles

Gets allowed titles (for example: Mr, Mrs, etc.).

JavaScript library method

patientportal.users.getAllowedTitles();

HTTP Method

GET

Url

/patientportalapi/patient/allowed-titles

Returns

string[]

19.2 GetDepartmentsAndDivisions

Returns the list of employer departments and divisions.

JavaScript library method

patientportal.users.getDepartmentsAndDivisions();

HTTP Method

GET

Url

/patientportalapi/users/departments-divisions

Returned JSON

```
{
  "Departments": [
    <list of DepartmentData>
  ]
  "Divisions": [
    <list of DivisionData>
  ]
}
```

19.3 GetEmployeeStatuses

Returns a list of employee statuses.

JavaScript library method

patientportal.users.getEmployeeStatuses();

HTTP Method

GET

Url

/patientportalapi/patient/employee-statuses

Returns

EmployeeStatus[]

19.4 GetUserRights

Returns the list of possible user rights.

JavaScript library method

patientportal.users.getUserRights();

HTTP Method

GET

Url

/patientportalapi/users/user-rights

Returns

User[]

19.5 GetUsers

Returns the list of users.

JavaScript library method

```
patientportal.users.getUsers({
            text: <text>,
            name: <name>,
            department: <department>,
            division: <division>,
            email: <email>,
            employeeNumber: <employee-number>,
            pageSortColumn: <page-sort-column>,
            pageSortDescending: <page-sort-descending>,
            pageNumber: <page-number>,
            pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/users/users

URL Parameters

text	string (optional)	Any text the API will try to filter by name, email or employee number.
name	string (optional)	Name filter.
department	string (optional)	Key of the department provided by the API upon GetDepartments.
division	string (optional)	Key of the division provided by the API upon GetDepartments.
email	string (optional)	Email filter.
employee-number	string (optional)	Employee number filter.
page-sort-column	int (optional)	The column index to sort the result: - 0 – User's name - 1 – Email address - 2 – Account type - 3 – Employee number By default the result is sorted by the user name.
page-sort-descending	int (optional)	True to sort result descending.
page-number	int (optional)	Required page number. Default 1.
page-size	int (optional)	Required page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of UserData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10,
  "SortColumn": 0,
  "SortDescending": false
}
```

19.6 GetUser

Returns the specific user.

JavaScript library method

```
patientportal.users.getUser({
        user: <user>
});
```

HTTP Method

GET

Url

/patientportalapi/users/user-data

URL Parameters

user	string	The key of the user provided by the API upon GetUsers.

Returns

UserData

19.7 CreateUser

Creates a new patient’s portal profile. If the patient already exists in the Meddbase system it joins the patient’s account with new portal profile.

JavaScript library method

```
patientportal.users.createUser({
	userData: <userData>
});
```

HTTP Method

POST

Url

/patientportalapi/users/create-user

POST Parameters

userData	UserData	Defines user’s details.

Returns

UserData

Remarks

At least *Name*, *Surname*, *DateOfBirth*, *EmailAddress*, *Mobile*, *Title,* and *Address.PostCode* must be provided. Once the user is created she has the manager rights.

The Meddbase system sends the activation email with an activation link to the patient’s email address The user must activate her account first before being able to log-on. The client should give the user advice that the confirmation email may be in their junk/spam folder.

If the patient

19.8 UpdateUser

Updates the user data.

JavaScript library method

```
patientportal.users.updateUser({
            userData: <userData>
});
```

HTTP Method

POST

Url

/patientportalapi/users/user-data

POST Parameters

userData	UserData	Defines user’s details.

Returns

UserData

19.9 RemoveAccess

Removes the manager’s rights so the user will not be able to access the Referral Portal sections.

JavaScript library method

```
patientportal.users.removeAccess({
            user: <user>
});
```

HTTP Method

GET

Url

/patientportalapi/users/remove-access

URL Parameters

user	string	The key of the user provided by the API upon GetUsers.

19.10 ResetPassword

Resets user’s current password and sends an email request to reset password.

JavaScript library method

```
patientportal.users.resetPassword({
        user: <user>
});
```

HTTP Method

GET

Url

/patientportalapi/users/reset-password

URL Parameters

user	string	The key of the user provided by the API upon GetUsers.

20 Patients

This section provides methods for searching for patients. Only logged-in user which has got *CanReferPatients* right (see GetConfig) is able to work with this section.

20.1 QuickSearch

Returns the patient that meets the search criteria.

JavaScript library method

```
patientportal.patients.quickSearch ({
        employeeNumber: <employee-number>,
        personalEmail: <personal-email>,
        workEmail: <work-email>
        });
```

HTTP Method

GET

Url

/patientportalapi/patients/patient

URL Parameters

employee-number	string	The employee number of the patient.
personal-email	string	The patient’s personal email.
work-email	string	The patient’s work email.

Returns

PersonDemographicData

Remarks

Throws various exceptions if there are duplicate records or if the user is not allowed to access the patient record. The client must to display the error message and allow user to put full patient’s details.

20.2 FullSearch

Returns the patient that meets the search criteria.

JavaScript library method

```
patientportal.patients.fullSearch({
        demog: <demog>
});
```

HTTP Method

POST

Url

/patientportalapi/patients/patient

POST Parameters

demog	PersonDemographicData	Defines person’s details.

Returns

PersonDemographicData

Remarks

The full search uses patient’s name, surname, email address, employee number, home address and data of birth to find the patient.

Throws various exceptions if there are duplicate records or if the user is not allowed to access the patient record.

20.3 FullSearchOrCreate

Returns the patient that meets the search criteria. If such as patient doesn’t exist then creates a new one.

JavaScript library method

```
patientportal.patients.fullSearchOrCreate({
        demog: <demog>
});
```

HTTP Method

POST

Url

/patientportalapi/patients/create-patient

POST Parameters

demog	PersonDemographicData	Defines person’s details.

Returns

PersonDemographicData

Remarks

The full search uses patient’s name, surname, email address, employee number, home address and data of birth to find the patient.

If the patient doesn’t exist then the API creates a new patient.

Throws various exceptions if there are duplicate records or if the user is not allowed to access the patient record.

20.4 GetPatients

Returns the list of patients. Note that the list includes only patients the logged in user is allowed to view.

JavaScript library method

```
patientportal.patients.getPatients({
        text: <text>,
        name: <name>,
        surname: <surname>,
        department: <department>,
        division: <division>,
        personalEmail: <personal-email>,
        workEmail: <work-email>,
        employeeNumber: <employee-number>,
        dobFrom: <dob-from>,
        dobTo: <dob-to>,
        pageSortColumn: <page-sort-column>,
        pageSortDescending: <page-sort-descending>,
        pageNumber: <page-number>,
        pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/patients/patients

URL Parameters

text	string (optional)	Any text the API will try to filter by name, email or employee number.
name	string (optional)	Name filter.
surname	string (optional)	Surname filter.
department	string (optional)	Key of the department provided by the API upon GetDepartments.
division	string (optional)	Key of the division provided by the API upon GetDepartments.
personal-email	string (optional)	Personal email filter.
work-email	string (optional)	Work email filter.
employee-number	string (optional)	Employee number filter.
dob-from	string (optional)	Filter by DOB from. If provided the patients without DOB are filtered out.
dob-to	string (optional)	Filter by DOB to. If provided the patients without DOB are filtered out.
page-sort-column	int (optional)	The column index to sort the result: - 0 – Patient's name - 1 – Email address - 2 - <ignored> - 3 – Employee number By default the result is sorted by the user name.
page-sort-descending	int (optional)	True to sort result descending.
page-number	int (optional)	Required page number. Default 1.
page-size	int (optional)	Required page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of PersonDemographicData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10,
  "SortColumn": 0,
  "SortDescending": false
}
```

20.5 GetPatient

Returns the specific patient.

JavaScript library method

```
patientportal.patients.getPatient({
	patient: <patient>
});
```

HTTP Method

GET

Url

/patientportalapi/patients/patient

URL Parameters

patient	string	The key of the patient provided by the API upon GetPatients.

Returns

PersonDemographicData

20.6 UpdatePatient

Updates the patient demographic data.

JavaScript library method

```
patientportal.patients.updatePatient({
	demog: <demog>
});
```

HTTP Method

POST

Url

/patientportalapi/patients/update-patient

POST Parameters

demog	PersonDemographicData	Defines person's details.

Returns

PersonDemographicData

Remarks

Throws an exception if the patient doesn't exist.

20.7 GetManagers

Returns the list of patient managers.

JavaScript library method

```
patientportal.patients.getManagers({
        patient: <patient>
});
```

HTTP Method

GET

Url

/patientportalapi/patients/managers

URL Parameters

patient	string	The key of the patient provided by the API upon GetPatients.

Returns

PersonDemographicData[]

21 Documents

This section provides the ability to download or upload patient's documents.

21.1 GetDocuments

Returns the list of document for a specific patient.

JavaScript library method

```
patientportal.documents.getDocuments({
        patient: <patient>,
        name: <name>,
        createdFrom: <created-from>,
        createdTo: <created-to>,
        pageSortColumn: <page-sort-column>,
        pageSortDescending: <page-sort-descending>,
        pageNumber: <page-number>,
        pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/documents/documents

URL Parameters

patient	string	The key of the patient provided by the API upon GetPatients.
name	string (optional)	Filters the documents by the name.
created-from	DateTime (optional)	Filters the documents by the created data.
created-to	DateTime (optional)	Filters the documents by the created data.
page-sort-column	int (optional)	The column index to sort the result: - 0 – Name - 1 – Created date Default: 0
page-sort-descending	int (optional)	True to sort result descending.
page-number	int (optional)	Required page number. Default 1.
page-size	int (optional)	Required page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of DocumentData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10,
  "SortColumn": 0,
  "SortDescending": false
}
```

21.2 AttachDocument

Uploads and attaches a new document to the patient's records.

JavaScript library method

```
patientportal.documents.attachDocument({
        patient: <patient>,
        documentName: <document-name>,
        documentComments: <document-comments>,
        uploading: {
                file: <file>,
                onProgress: <onProgress>,
        }
});
```

HTTP Method

POST

Url

/patientportalapi/documents/attach

URL Parameters

patient	string	The key of the patient provided by the API upon GetPatients.
document-name	string	The name of the document.
document-comments	string (optional)	The document comments/description.
uploading.file	javascript: File	File instance provided by JavaScript input element: `file:document.getElementById("myFile").files[0]`

uploading.onProgress	javascript :
	function (int)
		Progress callback. For example:
		`onProgress: function (progress) {`
		`      console.log("Progress: " +`
		`            progress + "% uploaded.");`
		`}`

Returns

DocumentData

Remarks

Please note the body of the request is the actual document content. For that reason you need to include *ASP.NET_SessionId* in cookies and the token in URL.

21.3 GetDocument

Returns a document details.

JavaScript library method

```
patientportal.documents.getDocument({
	document: <document>
});
```

HTTP Method

GET

Url

/patientportalapi/documents/document

URL Parameters

document	document	The key of the document provided by the API.

Returns

DocumentData

21.4 GetTypes

Returns a document types the user may have an access to. The feature needs to be enabled. You could check whether the feature is enabled via GetConfig by checking the *DocumentTypeSecurityEnabled* flag.

JavaScript library method

```
patientportal.documents.getTypes();
```

HTTP Method

GET

Url

/patientportalapi/documents/types

Returns

DocumentType

22 Referrals

This section provides methods for working with referrals. Only logged-in user which has got *CanReferPatients* right (see GetConfig) is able to work with this section.

22.1 GetReferrals

Returns the referrals that meets the search criteria.

JavaScript library method

```
patientportal.referrals.getReferrals({
            patient: <patient>,
            text: <text>,
            status: <status>,
            referralNumber: <referral-number>,
            patientName: <patient-name>,
            referrerName: <referrer-name>,
            department: <department>,
            division: <division>,
            case: <case>,
            pageSortColumn: <page-sort-column>,
            pageSortDescending: <page-sort-descending>,
            pageNumber: <page-number>,
            pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/referrals

URL Parameters

patient	string (optional)	The key of the patient provided by the API upon GetPatients.
text	string (optional)	Any text the API will try to filter by patient's/referrer's name, email, employee number or by referral number if the text is a number.
status	int (optional)	Status filter: - 0 – <disabled, filter no active> - 1 – Questionnaire required - 2 – Documents required - 3 – In progress - 4 – Closed only - 5 – Opened only
referral-number	string (optional)	The number of the referral.
patient-name	string (optional)	The patient name.
referrer-name	string (optional)	The referrer name.
department	string (optional)	Key of the department provided by the API upon GetDepartments.
division	string (optional)	Key of the division provided by the API upon GetDepartments.
case	int (optional)	The case key for the referral.
page-sort-column	int (optional)	The column index to sort the result: - 0 – Last modification date - 1 – Referral number - 2 – Patient name - 3 – Referrer name - 4 – Date created By default the result is sorted by last modification date.
page-sort-descending	int (optional)	True to sort result descending. Note that this parameter is ignored for column index 0.
page-number	int (optional)	Required page number. Default 1.
page-size	int (optional)	Required page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of ReferralOverviewData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10,
  "SortColumn": 0,
  "SortDescending": false

}
```

22.2 GetReferral

Returns the referral full details.

JavaScript library method

```
patientportal.referrals.getReferra({
            referral: <referral>
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/referral

URL Parameters

referral	string	The key of the referral provided by the API upon GetReferrals.

Returns

ReferralData

22.3 Start

Starts and returns a new referral.

JavaScript library method

```
patientportal.referrals.start({
            patient: <patient>,
            appointmentType: <appointment-type>,
            caseTitle: <case-title>
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/start

URL Parameters

patient	string	The key of the patient provided by the API upon section Patients.
appointmentType	string	The key of the appointment type provided by the API upon GetAppointmentTypes. To find out how to work with referral appointments please read section **Error! Reference source not found.**.
caseTitle	string	The reason for starting the referral.

Returns

ReferralData

22.4 StartFromQuestionnaireRequest

Starts and returns a new referral using the patient associated with the specified questionnaire request. A default appointment type is used, and the questionnaire results documents are attached to the referral. Only ‘complete’ questionnaires can be referred.

JavaScript library method

```
patientportal.referrals.startFromQuestionnaireRequest({
        questionnaireRequest: <questionnaire-request>
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/start-from-questionnaire-request

URL Parameters

questionnaire-request	string	The key of the questionnaire request returned by CreateRequest or GetQuestionnaires

Returns

ReferralData

22.5 AttachDocument

Uploads and attaches a new document to the existing referral.

JavaScript library method

```
patientportal.referrals.attachDocument({
        referral: <referral>,
        documentName: <document-name>,
        documentComments: <document-comments>,
        uploading: {
                file: <file>,
                onProgress: <onProgress>,
        }
});
```

HTTP Method

POST

Url

/patientportalapi/referrals/attach-document

URL Parameters

referral	string	The key of the referral provided by the API upon GetReferrals.
document-name	string	The name of the document.
document-comments	string (optional)	The document comments/description.
uploading.file	javascript: File	File instance provided by JavaScript input element: `file:document.getElementById("myFile").files[0]`

uploading.onProgress	javascript :
	function (int)
		Progress callback. For example:
		`onProgress: function (progress) {`
		`      console.log("Progress: " +`
		`            progress + "% uploaded.");`
		`}`

Returns

ReferralData

Remarks

Please note the body of the request is the actual document content. For that reason you need to include *ASP.NET_SessionId* in cookies and the token in URL.

22.6 Send

Sends the referral.

JavaScript library method

```
patientportal.referrals.send({
        referral: <referral>,
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/send

URL Parameters

referral	string	The key of the referral provided by the API upon GetReferrals.

Returns

ReferralData

22.7 Cancel

Cancels the referral.

JavaScript library method

```
patientportal.referrals.cancel({
        referral: <referral>,
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/cancel

URL Parameters

referral	string	The key of the referral provided by the API upon GetReferrals.

Returns

ReferralData

22.8 FindUserToReallocate

Search for user to re-allocate the referral. The search provides only basic user's data: *Key*, *Name*, *Surname, EmployeeNumber* and *Title*.

JavaScript library method

```
patientportal.referrals.findUserToReallocate ({
        employeeNumber: <employee-number>
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/find-user-to-reallocate

URL Parameters

employee-number	string	The user's employee number.

Returned JSON

```
{
  "Key": "u1324",
  "Title": "Mr."
  "Name": "John",
  "Surname": "Smith",
  "EmployeeNumber": "1234"
}
```

22.9 Reallocate

Re-allocate the referral to another user.

JavaScript library method

```
patientportal.referrals.reallocate({
        referral: <referral>,
        user: <user>
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/reallocate

URL Parameters

referral	string	The key of the referral provided by the API upon GetReferrals.
user	string	The key of the user provided by the API upon FindUserToReallocate.

Returns

ReferralData

22.10 BulkReallocate

Re-allocate all referrals created by one user to a different user. Returns count of reallocated referrals.

JavaScript library method

```
patientportal.referrals.bulkReallocate({
        fromUser: <from-user>,
        patient: <patient>,
        toUser: <to-user>
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/bulk-reallocate

URL Parameters

from-user	string (optional*)	The key of the user provided by the API upon FindUserToReallocate. Use this filter to reallocate referrals assigned to a specific user.
patient	string (optional*)	The key of the patient provided by the API upon section Patients. Use this filter to reallocated referrals of a specific patient.
to-user	string	The key of the user provided by the API upon FindUserToReallocate.

Remarks

* Either *from-user* or *patient* parameter must be provided.

Returned JSON

```
{
  "Referrals": 18
}
```

22.11 FollowUp

Sends the follow up referral.

JavaScript library method

```
patientportal.referrals.followUp({
            referral: <referral>,
            reason: <reason>
});
```

HTTP Method

GET

Url

/patientportalapi/referrals/follow-up

URL Parameters

referral	string	The key of the referral provided by the API upon GetReferrals.

POST Parameters

reason	string	The reason why the user sends the follow up referral.

Returns

ReferralData

22.12 GetBasicStatistics

Returns the basic referral statistics.

JavaScript library method

```
patientportal.referrals.getBasicStatistics();
```

HTTP Method

GET

Url

/patientportalapi/referrals/basic-statistics

Returns JSON

```
[
  {
    "Name": "In progress",
    "Count": 21,
    "Percent": 47
  },
  {
    "Name": "Discharged",
    "Count": 21,
    "Percent": 53
  }
]
```

23 Patient Medical Report Review

This section provides methods to manage patient's review of the medical report for the patient. The patient doesn't need to be logged in. The client has to provide the client key only (see Client authentication).

Workflow: The patient gets the email which includes URL address to the online portal and the validation *key* (URL get parameter). The client needs to provide validation *key* within each request in this section.

Obvious reviewing process goes through those steps:

- Validation process through email or phone
- Reviewing
- Releasing or refusing the report with/without factual changes or comments

23.1 SendValidationSMS

Sends the validation code through the text message.

JavaScript library method

```
patientportal.patientReportReview.sendValidationSMS({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/patient-report-review/send-validation-sms

URL Parameters

key	string	The validation key provided in the URL.

Remarks

Method does not return any data. If something goes wrong please check exception (see Error handling).

23.2 SendValidationEmail

Sends the validation code through the email.

JavaScript library method

```
patientportal.patientReportReview.sendValidationEmail({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/patient-report-review/send-validation-email

URL Parameters

key	string	The validation key provided in the URL.

Remarks

Method does not return any data. If something goes wrong please check exception (see Error handling).

23.3 SubmitValidationCode

Submits the validation code and returns the medical report review data. Once the validation code is submitted a temporary security context is created.

JavaScript library method

```
patientportal.patientReportReview.submitValidationCode({
        key: <key>,
        code: <code>
});
```

HTTP Method

GET

Url

/patientportalapi/patient-report-review/submit-validation-code

URL Parameters

key	string	The validation key provided in the URL.
code	string	The validation code the patient gets through the email or text message.

Returns

PatientReportReviewData

Remarks

Submitting the validation code creates a temporary security context which is valid for 30 minutes. Within this time frame the patient needs to review the medical report and submit the authorisation.

The client may use ValidateKey to verify whether the security context runs out and request new validation code to create a new security context.

If something goes wrong please check exception (see Error handling).

23.4 ValidateKey

Validates whether the security context for the last submitted validation code is still valid.

JavaScript library method

```
patientportal.patientReportReview.validateKey({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/patient-report-review/validate-key

URL Parameters

key	string	The validation key provided in the URL.

Remarks

Method does not return any data. If something goes wrong please check exception (see Error handling).

If the security context runs out the patient may request another validation code and submit it to create new security context.

23.5 GetMedicalReportData

Returns the medical report data.

The method is only valid if the current temporary security context is valid (see SubmitValidationCode).

JavaScript library method

```
patientportal.patientReportReview.getMedicalReportData({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/patient-report-review/review-data

URL Parameters

key	string	The validation key provided in the URL.

Returns

PatientReportReviewData

Remarks

If something goes wrong please check exception (see Error handling).

23.6 SubmitAuthorisation

Release or refuse the medical report and stores patient's comments and requested factual changes. The method is only valid if the current temporary security context is valid (see SubmitValidationCode).

JavaScript library method

```
patientportal.patientReportReview.submitAuthorisation({
        key: <key>,
        authorised: <authorised>,
        comments: <comments>,
        factualChanges:[
                {
                        Subject: <Subject>,
                        Comments: <Comments>
                },
                ...
        ]
});
```

HTTP Method

GET

Url

/patientportalapi/patient-report-review/review-data

URL Parameters

key	string	The validation key provided in the URL.
authorised	bool	true if the patient releases the medical report, false if the patient refuses it.

POST Parameters

comments	string	Comments provided by the patient.
factualChanges	[{ Subject: string, Comments: string } ,...]	List of factual changes requested by the patient. Subject – Subject of possible factual changes provided by the API (see PatientReportReviewData) Comments – Patient's comments

Returns

PatientReportReviewData

Remarks

Calling this method removes the security context so no another command (GetMedicalReportData, SubmitAuthorisation) is allowed without new validation. If something goes wrong please check exception (see Error handling).

24 Manager Medical Report Review

This section provides methods to manage a review of the medical report by the manager. The manager doesn't need to be logged in. The client has to provide the client key only (see Client authentication).

Workflow: The manager gets the email which includes URL address to the online portal and the validation *key* (URL get parameter). The client needs to provide validation *key* within each request in this section.

24.1 SendValidationSMS

Sends the validation code through the text message.

JavaScript library method

```
patientportal.managerReportReview.sendValidationSMS({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/manager-report-review/send-validation-sms

URL Parameters

key	string	The validation key provided in the URL.

Remarks

Method does not return any data. If something goes wrong please check exception (see Error handling).

24.2 SendValidationEmail

Sends the validation code through the email.

JavaScript library method

```
patientportal.managerReportReview.sendValidationEmail({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/manager-report-review/send-validation-email

URL Parameters

key	string	The validation key provided in the URL.

Remarks

Method does not return any data. If something goes wrong please check exception (see Error handling).

24.3 SubmitValidationCode

Submits the validation code and returns the medical report review data. Once the validation code is submitted a temporary security context is created.

JavaScript library method

```
patientportal.managerReportReview.submitValidationCode({
        key: <key>,
        code: <code>
});
```

HTTP Method

GET

Url

/patientportalapi/manager-report-review/submit-validation-code

URL Parameters

key	string	The validation key provided in the URL.
code	string	The validation code the manager gets through the email or text message.

Returns

ManagerReportReviewData

Remarks

Submitting the validation code creates a temporary security context which is valid for 30 minutes. Within this time frame the manager needs to review the medical report and submit the authorisation.

The client may use ValidateKey to verify whether the security context runs out and request new validation code to create a new security context.

If something goes wrong please check exception (see Error handling).

24.4 ValidateKey

Validates whether the security context for the last submitted validation code is still valid.

JavaScript library method

```
patientportal.managerReportReview.validateKey({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/manager-report-review/validate-key

URL Parameters

key	string	The validation key provided in the URL.

Remarks

Method does not return any data. If something goes wrong please check exception (see Error handling).

If the security context runs out the manager may request another validation code and submit it to create new security context.

24.5 GetMedialReportData

Returns the medial report data.

The method is only valid if the current temporary security context is valid (see SubmitValidationCode).

JavaScript library method

```
patientportal.managerReportReview.getMedicalReportData({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/manager-report-review/review-data

URL Parameters

key	string	The validation key provided in the URL.

Returns

ManagerReportReviewData

Remarks

If something goes wrong please check exception (see Error handling).

25 Recalls

This section provides the ability to work with recalls.

25.1 GetRecalls

Returns the list of recalls.

JavaScript library method

```
patientportal.recalls.getRecalls({
        patient: <patient>,
        dueFrom: <due-from>,
        dueTo: <due-to>,
        pageSortColumn: <page-sort-column>,
        pageSortDescending: <page-sort-descending>,
        pageNumber: <page-number>,
        pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/recalls/recalls

URL Parameters

patient	string (optional)	The key of the patient provided by the API upon GetPatients.
due-from	DateTime (optional)	Due date filter..
due-to	DateTime (optional)	Due date filter.
page-sort-column	int (optional)	The column index to sort the result: - 0 – Due date - 1 – Created date - 2 – Reminder date - 3 – Appointment type name Default: 0
page-sort-descending	int (optional)	True to sort result descending.
page-number	int (optional)	Required page number. Default 1.
page-size	int (optional)	Required page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of RecallData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10,
  "SortColumn": 0,
  "SortDescending": false
}
```

25.2 DeleteRecall

Removes the recall.

JavaScript library method

```
patientportal.recalls.deleteRecall({
        recall: <recall>
});
```

HTTP Method

GET

Url

/patientportalapi/recall/delete

URL Parameters

recall	string	The key of the recall provided by the API upon method GetRecalls.

26 Questionnaires

This section provides the ability to work with explicitly requested patient questionnaires (as opposed to questionnaires linked to appointments). Only a logged in user with the *`CanViewPatient`* right can use this section. Specific rights are also required for individual methods as well.

Note that all the methods which take a ‘category’ parameter only accept the category ‘ppq’. Other categories may be supported in the future.

26.1 GetQuestionnaireForms

Returns the list of questionnaire forms that can be requested for any patient. If a module is specified then only the forms included in that module are returned.

JavaScript library method

```
patientportal.questionnaires. getQuestionnaireForms ({
        moduleKey: <module-key>,
        category: <’ppq’>,
        formName: <form-name>,
        pageNumber: <page-number>,
        pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/questionnaires/forms

URL Parameters

module-key	string (optional)	The key of the module provided by the API upon GetModules.
category	string (required)	Always ‘ppq’.
form-name	string (optional)	Filter for form name.
page-number	int (optional)	Page number. Default 1.
page-size	int (optional)	Page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of QuestionnaireFormData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10
}
```

26.2 GetModules

Returns the list of questionnaire modules that are visible for the logged in user. Modules are visible if they are created by the user or shared by another user.

JavaScript library method

```
patientportal.questionnaires. getModules ({
        category: <'ppq'>,
        moduleName: <module-name>,
        pageNumber: <page-number>,
        pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/questionnaires/modules

URL Parameters

category	string (required)	Always ‘ppq’.
module-name	string (optional)	Filter for module name.
page-number	int (optional)	Page number. Default 1.
page-size	int (optional)	Page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of QuestionnaireModuleData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10
}
```

26.3 GetQuestionnaires

Returns the list of requested questionnaires. Questionnaires requested together (with or without a specific module) are listed together.

JavaScript library method

```
patientportal.questionnaires.getQuestionnaires({
            patient: <patient>,
            category: 'ppq',
            status: <status>,
            pageSortColumn: <page-sort-column>,
            pageSortDescending: <page-sort-descending>,
            pageNumber: <page-number>,
            pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/questionnaires/questionnaires

URL Parameters

patient	string (optional)	The key of the patient provided by the API upon GetPatients.
category	string (required)	Always 'ppq'
status	int (optional)	Status code filter.
page-sort-column	int (optional)	The column index to sort the result: - 0 – Created order - 1 – Status Default: 0
page-sort-descending	int (optional)	True to sort result descending.
page-number	int (optional)	Page number. Default 1.
page-size	int (optional)	Page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of QuestionnaireOverviewData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10,
  "SortColumn": 0,
  "SortDescending": false
}
```

26.4 GetQuestionnaireResults

Get the results of a questionnaire request. The result of this call is an array of result objects, one for each questionnaire in the request. Each result has the HTML markup of a results document created from the answers.

JavaScript library method

```
patientportal.questionnaires. getQuestionnaireResults ({
        questionnaireRequest: <questionnaire-request>
});
```

HTTP Method

GET

Url

/patientportalapi/questionnaires/results

URL Parameters

questionnaire-request	string	The key of the questionnaire request returned by CreateRequest or GetQuestionnaires

Returns

`QuestionnaireResultData[]`

26.5 SaveModule

Create or update a module. Modules can only be shared if the logged in user has the *CanModifyQuestionnaireForms* right. Returns the key of the module created.

JavaScript library method

```
patientportal.questionnaires. saveModule ({
        moduleKey: <module-key>,
        category: <'ppq'>,
        moduleName: <module-name>,
        shared: <shared>,
        formKeys: <form-keys>
});
```

HTTP Method

POST

Url

/patientportalapi/questionnaires/save-module

URL Parameters

module-key	string (optional)	The key of the module provided by the API upon GetModules or a previous call to SaveModule. If not specified, a new module will be created.
category	string (required)	Always 'ppq'.
module-name	string (required)	Module name.
shared	bool (optional)	Share the module with other users. Default false.

POST Parameters

form-keys	string[] (required)	List of form keys returned from GetQuestionnaireForms.

Returned JSON

```
"ae71266c1548673052d1240115466e63a8ece5c5d3cf06abe1d006f7d975ff76"
```

26.6 DeleteModule

Delete the specified module. The user must have the *CanModifyQuestionnaireForms* right.

JavaScript library method

```
patientportal.questionnaires.getQuestionnaires({
        moduleKey: <module-key>

});
```

HTTP Method

GET

Url

/patientportalapi/questionnaires/delete-module

URL Parameters

module-key	string (required)	The key of the patient provided by the API upon GetModules.

26.7 CreateRequest

Create a new questionnaire request. Either a module or a list of ad-hoc questionnaire forms can be requested. One or the other must be specified. Returns the key of the request, the same keys that are returned by GetQuestionnaires.

JavaScript library method

```
patientportal.questionnaires. createRequest ({
        patient: <patient>,
        moduleKey: <module-key>,
        formKeys: <form-keys>
});
```

HTTP Method

POST

Url

/patientportalapi/questionnaires/create-request

URL Parameters

patient	string (optional)	The key of the patient provided by the API upon GetPatients.
module-key	string (optional)	The key of the module provided by the API upon GetModules or a previous call to SaveModule. If not specified, a new module will be created.

POST Parameters

form-keys	string[] (optional)	List of form keys returned from GetQuestionnaireForms.

Returned JSON

“ae71266c1548673052d1240115466e63a8ece5c5d3cf06abe1d006f7d975ff76”

27 Questionnaire Request

This section provides methods for a patient to complete a questionnaire without having to register with the portal or use a password. The client has to provide the client key only (see Client authentication).

Workflow: The patient gets the email which includes URL address to the online portal and the validation *key* (URL get parameter). The client needs to provide validation *key* within each request in this section.

The process goes through these steps:

- Validation process through email or phone
- Save questionnaire (they are allowed to come back later)
- Submit questionnaire

27.1 SendValidationSMS

Sends the validation code through the text message.

JavaScript library method

```
patientportal.questionnaireRequest.sendValidationSMS({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/questionnaire-request/send-validation-sms

URL Parameters

key	string	The validation key provided in the URL.

Remarks

Method does not return any data. If something goes wrong please check exception (see Error handling).

27.2 SendValidationEmail

Sends the validation code through the email.

JavaScript library method

```
patientportal.questionnaireRequest.sendValidationEmail({
	key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/questionnaire-request/send-validation-email

URL Parameters

key	string	The validation key provided in the URL.

Remarks

Method does not return any data. If something goes wrong please check exception (see Error handling).

27.3 SubmitValidationCode

Submits the validation code and returns the questionnaire data. Once the validation code is submitted a temporary security context is created.

JavaScript library method

```
patientportal.questionnaireRequest.submitValidationCode({
	key: <key>,
	code: <code>
});
```

HTTP Method

GET

Url

/patientportalapi/questionnaire-request/submit-validation-code

URL Parameters

key	string	The validation key provided in the URL.
code	string	The validation code the patient gets through the email or text message.

Returns

QuestionnaireData

Remarks

Submitting the validation code creates a temporary security context which is valid for 30 minutes. Within this time frame the patient needs to complete the questionnaire and save/submit it.

The client may use ValidateKey to verify whether the security context runs out and request new validation code to create a new security context.

If something goes wrong please check exception (see Error handling).

27.4 ValidateKey

Validates whether the security context for the last submitted validation code is still valid.

JavaScript library method

```
patientportal.questionnaireRequest.validateKey({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/questionnaire-request/validate-key

URL Parameters

key	string	The validation key provided in the URL.

Remarks

Method does not return any data. If something goes wrong please check exception (see Error handling).

If the security context runs out the patient may request another validation code and submit it to create new security context.

27.5 SaveQuestionnaire

Saves partial answers of specific questions in the questionnaire.

JavaScript library method

```
patientportal.questionnaireRequest.saveQuestionnaire({
        key: <key>,
        answers: <answers>
});
```

HTTP Method

POST

Url

/patientportalapi/questionnaire-request/save-questionnaire

POST Parameters

key	string	The validation key provided in the URL.
answers	QuestionnaireAnswerData[]	Collection of answers.

Remarks

The client can call this method as many time as it needs. Only answers attached in parameter *Answers* will be updated. The sort-order of questions isn't important.

After the first call of this method the questionnaire's status changes to *Partially Complete*.

The client must call SubmitQuestionnaire after all required questions are answered.

27.6 SubmitQuestionnaire

Submits the specific questionnaire.

JavaScript library method

patientportal.questionnaireRequest.submitQuestionnaire({key: <key>});

HTTP Method

GET

Url

/patientportalapi/questionnaire-request/submit-questionnaire

URL Parameters

key	string	The validation key provided in the URL.

Remarks

The client calls this method when all required questions are answered. After submitting the questionnaire the status will change to *Complete*.

When any required question is not answered the exception with event code 40001 is thrown (see Error handling).

27.7 SaveAndSubmitQuestionnaire

Combination of SaveQuestionnaire and SubmitQuestionnaire. This method can be used on the latest questionnaire page where you may save the answers on the latest page and submit the questionnaire at once.

JavaScript library method

```
patientportal.questionnaireRequest.saveAndSubmitQuestionnaire({
        key: <key>,
        answers: <answers>
});
```

HTTP Method

POST

Url

/patientportalapi/questionnaire-request/save-submit-questionnaire

URL Parameters

key	string	The validation key provided in the URL.
answers	QuestionnaireAnswerData[]	Collection of answers.

Remarks

The client calls this method when all required questions are answered. After submitting the questionnaire the status will change to *Complete*.

When any required question is not answered the exception with event code 40001 is thrown (see Error handling).

28 Telemedicine

This section provides methods for Telemedicine video conference integration.

28.1 Authorise

Authorises the key and returns the token for the client app to connect to the video.

JavaScript library method

patientportal.telemedicine.authorise({key: <key>});

HTTP Method

GET

Url

/patientportalapi/telemedicine/authorise

URL Parameters

key	string	Key to verify and generate token against. This key is part of the URL provided to access telemedicine conference.
username	string	The name of the participant.

Returns

AuthorizeTelemedicineData

Remarks

Authorises the user’s access to telemedicine conference and returns the token that could be used to lunch the video app.

Twilio integration

Currently the Meddbase application supports Twilio provider only. Twilio provides a multi-party video application to demonstrate how you could built a solution with Twilio's Programmable Video JS SDK, Twilio's Conversations JS SDK, and Create React App.

The authorise method generates an access token (in JWT format) that is used to connect to a room. A step-by-step guide how to join to a room could be found here.

29 Single sign-on

Meddbase system provides integration with SAML2 based single sign on providers.

For identity provider initiated single sign on, the SAML request is sent to the API Gateway as a POST. The request is sent to following URL:

```
https://api.meddbase.com/ssoapi
```

The gateway generates a unique login token (uuid) and the user is redirected to the portal url configured in the application as follows:

https://portalurl/#/ssologin?loginToken=token

This token is valid for 20 seconds from the time of creation and is marked as expired after that time. The token can be used to login using Direct Login. Any error are passed as "error" query parameter.

For service provider initiated single sign on calls, SSO Login Details call is used to retrieve the URL to redirect to the identity provider to login. The user is redirected to the provided URL, where once logged in, the identity provider sends a SAML response to the URL above, following the same workflow.

30 Pathways

This section provides methods to work with pathways – getting a list of pathways and completing pathways tasks. Pathways feature may be disabled or enabled for a chamber and can be checked through the GetConfig function.

30.1 Pathway structure and task types

The pathway is a set of nested tasks. A task could be either a simple action or a container (either serial or parallel). Containers include other nested tasks. An example of a pathway is provided here PathwayData. Below is the list of action types you can receive via API.

Type	**Name**	**Description**
WF	Workflow	The workflow is a container of tasks that runs in order. The workflow is a logical unit like 'Book first consultation' that than includes more task like attach document, book an appointment, pre-appointment questionnaire and arrive appointment.
SQ	Sequence	The sequence is a container of tasks that runs in order like the workflow but does not represent a big standalone unit like a workflow. The sequence is used for small sequence tasks within a workflow.
ST	Step	A container of parallel tasks.
PL	Parallel	A container of parallel tasks.
BA	Book an appointment	The task to book an appointment. The task provides the appointment type and optional services that are required to be booked. To book the appointment see Appointments section.
AA	Arrive an appointment	The purpose of this task is to wait for the appointment. However if the booked appointment has been cancelled in the meantime the *IsBooked* field of AppointmentTaskData is *false* and the patient needs to book an appointment again in the same way as with the Book an appointment task.
QR	Questionnaire	The patient questionnaire to complete. Use *Key* field from QuestionnaireTaskData to get the questionnaire details and complete the questionnaire (see GetQuestionnaireDetail).
AD	Attach document	A task to attach a document. The document can be attached via AttachDocument API method.
GT	General task	The general task provides some text (task *Description*) that needs to be accepted via GetTask API metnod.
CH	Choice *(Make a decision pathway task)*	The choice provides a different list of options and the patient needs to choose one of them. You can submit the selected option via ChooseOption API method.

30.2 GetPathways

Returns the list of pathways shared with the currently logged-in patient.

JavaScript library method

```
patientportal.pathways.getPathways ({
            pageSortColumn: <page-sort-column>,
            pageSortDescending: <page-sort-descending>,
            pageNumber: <page-number>,
            pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/pathways/pathways

URL Parameters

page-sort-column	int (optional)	The column index to sort the result: - 0 – Last modified - 1 – Pathway ID - 3 – Due date - 4 – Pathway name Default: 0
page-sort-descending	int (optional)	True to sort result descending.
page-number	int (optional)	Page number. Default 1.
page-size	int (optional)	Page size. Default 10. Minimum 5. Maximum 50.

Returned JSON

```
{
  "Items": [
    <list of PathwayOverviewData>
  ]
  "TotalCount":24,
  "CurrentPage":1,
  "PageSize":10
}
```

30.3 GetPathway

Returns a specific pathway.

JavaScript library method

```
patientportal.pathways.getPathway ({
            pathway: <pathway>
});
```

HTTP Method

GET

Url

/patientportalapi/pathways/pathway

URL Parameters

pathway	int	Pathway ID

Returns

PathwayData

30.4 GetTask

Returns a full task description.

JavaScript library method

```
patientportal.pathways.getTask({
            pathway: <pathway>,
            task: <task>
});
```

HTTP Method

GET

Url

/patientportalapi/pathways/task

URL Parameters

pathway	string	Pathway key
task	string	Task key

Returns

PathwayTaskData

30.5 AcceptTask

Accepts the general task within the pathway.

JavaScript library method

```
patientportal.pathways.getPathway({
        pathway: <pathway>,
        task: <task>
});
```

HTTP Method

GET

Url

/patientportalapi/pathways/accept-task

URL Parameters

pathway	string	Pathway key
task	string	Task key

Returns

PathwayData

30.6 ChooseOption

Selects an option within the choice task.

JavaScript library method

```
patientportal.pathways.chooseOption({
        pathway: <pathway>,
        task: <task>,
        index: <index>
});
```

HTTP Method

GET

Url

/patientportalapi/pathways/choose-option

URL Parameters

pathway	string	Pathway key
task	string	Task key
index	int	The index of the option. Note that he index starts from 1.

Returns

PathwayData

30.7 AttachDocument

Attaches a document to a pathway task.

JavaScript library method

```
patientportal.pathways.attachDocument({
        pathway: <pathway>,
        task: <task>,
        documentName: <document-name>,
        documentComments: <document-comments>,
        uploading: {
                file: <file>,
                onProgress: <onProgress>,
        }
});
```

HTTP Method

POST

Url

/patientportalapi/pathways/attach-document

URL Parameters

pathway	string	Pathway key
task	string	Task key
document-name	string	The name of the document.
document-comments	string (optional)	The document comments/description.
uploading.file	javascript: File	File instance provided by JavaScript input element: `file:document.getElementById("myFile").files[0]`

uploading.onProgress	javascript :
	function (int)
		Progress callback. For example:
		`onProgress: function (progress) {`
		`      console.log("Progress: " +`
		`           progress + "% uploaded.");`
		`}`

Returns

PathwayData

Remarks

Please note the body of the request is the actual document content. For that reason you need to include *ASP.NET_SessionId* in cookies and the token in URL.

30.8 GetDefinitions

Gets the pathways definitions allowed for patient to run.

JavaScript library method

```
patientportal.pathways.getDefinitions();
```

HTTP Method

GET

Url

/patientportalapi/pathways/get-definitions

Returns

PathwayDefinitionData

30.9 Availability

Returns availability for a particular pathway definition if that pathways can be run by the patient.

JavaScript library method

```
patientportal.pathways.availability ({
            pathwayDef: <pathwayDef>
            }
});
```

HTTP Method

GET

Url

/patientportalapi/pathways/availability

URL Parameters

pathwayDef	string	Pathway Definition key

Returns

PathwayAvailabilityData

30.10 StartPathway

Starts a pathway if allowed.

JavaScript library method

```
patientportal.pathways.startPathway ({
            pathwayDef: <pathwayDef>
            }
});
```

HTTP Method

GET

Url

/patientportalapi/pathways/start-pathway

URL Parameters

pathwayDef	string	Pathway Definition key

Returns

Pathway Id (int)

31 Absence Management

This section provides methods to work with Absence Management – getting a list of absences, detailed absence information and updating absences in various ways.

31.1 GetAbsences

Returns a list of absence records filtered by any of the optional parameters specified. Does not show deleted absences.

JavaScript library method

```
patientportal.absences.getAbsences({
        patient: <patient>,
        from: <date-from>,
        to: <date-to>,
        status: <status>,
        text: <text>,
        department: <department>,
        division: <division>,
        questionnaireStatus: <questionnaire-status>,
        pageSortColumn: <page-sort-column>,
        pageSortDescending: <page-sort-descending>,
        pageNumber: <page-number>,
        pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/absences/absences

URL Parameters

patient	string (optional)	The absentee's key id.
date-from	DateTime (optional)	The earliest date to look for absences.
date-to	DateTime (optional)	The latest date to look for absences.
status	int (optional)	Open = 0, Closed = 2 (Deleted absences are not shown in the API, so cannot be filtered for).
text	string (optional)	Checks if any of the following attributes of the absentee starts with the specified text: Surname, Name, WorkEmail, Email, LegacyNr.
department	string (optional)	Key of the department provided by the API upon GetDepartments.
division	string (optional)	Key of the division provided by the API upon GetDepartments.
questionnaireStatus	Int (optional)	0 = Not Sent, 1 = Sent, 2 = Completed.
page-sort-column	int (optional)	The column index to sort the result: - 0 – From date - 1 – To date - 2– Reason Default: 0
page-sort-descending	int (optional)	True to sort result descending.
page-number	int (optional)	Page number. Default 1.
page-size	int (optional)	Page size. Default 5. Minimum 5. Maximum 50.

Returned JSON

```
{
    "status": "ok",
    "result": {
        "Items": [
            <list of AbsenceOverviewData>
        ],
        "TotalCount": 15,
        "CurrentPage": 1,
        "PageSize": 10,
        "SortColumn": 0,
        "SortDescending": false
    }
}
```

31.2 GetAbsence

Returns an absence record.

JavaScript library method

```
patientportal.absences.getAbsence({
        absence: <absence>
});
```

HTTP Method

GET

Url

/patientportalapi/absences/absence

URL Parameters

absence	string	The key of the absence.

Returned JSON

```
{
    "status": "ok",
    "result": {
        <AbsenceData>
    }
}
```

31.3 SaveAbsence

Saves changes to an absence record. Any of the attributes in the AbsenceData object can be modified and sent back to the API, but the only ones for which changes will be saved to the database are:

- EndDate
- LostWork.Days
- LostWork.Hours
- AccidentAtWork

JavaScript library method

```
patientportal.absences.saveAbsence({
        absenceData: <absence-data>
});
```

HTTP Method

POST

Url

/patientportalapi/absences/absence

URL Parameters

absence-data	AbsenceData	The AbsenceData object received from a call to the GetAbsence method.

Returned JSON

```
{
    "status": "ok",
    "result": {
        <AbsenceData>
    }
}
```

31.4 GetPatientAbsenceOverview

Returns a patient absence overview, which is limited to whether the patient is currently absent and the Bradford Factor score for the specified time period, or if no time period is specified, the previous 52 weeks. Does not show deleted absences.

JavaScript library method

```
patientportal.absences.getPatientAbsenceOverview({
        patient: <patient>,
        from: <date-from>,
        to: <date-to>
});
```

HTTP Method

GET

Url

/patientportalapi/absences/overview

URL Parameters

patient	string	The patient key.
date-from	DateTime (optional)	The beginning date with which to calculate the patient's Bradford Factor score (defaults to now minus 52 weeks).
date-to	DateTime (optional)	The end date with which to calculate the patient's Bradford Factor score (defaults to now).

Returned JSON

```
{
    "status": "ok",
    "result": {
        <PatientAbsenceData>
    }
}
```

31.5 ResendRTW

Resends the questionnaire request email / SMS for the specified absence record to the patient. Sets the absence record's questionnaire status to 'sent' if not already set.

JavaScript library method

```
patientportal.absences.resendRTW ({
        absence: <absence>
});
```

HTTP Method

GET

Url

/patientportalapi/absences/resend-rtw

URL Parameters

absence	string	The key of the absence.

Returned JSON

```
{
    "status": "ok",
    "result": {
        <AbsenceData>
    }
}
```

31.6 ApproveAndClose

Sets the status of the specified absence record to 'closed'.

JavaScript library method

```
patientportal.absences.approveAndClose ({
        absence: <absence>
});
```

HTTP Method

GET

Url

/patientportalapi/absences/approve-and-close

URL Parameters

absence	string	The key of the absence.

Returned JSON

```
{
    "status": "ok",
    "result": {
        <AbsenceData>
    }
}
```

31.7 GetManagers

Gets the demographic data of the patient’s managers.

JavaScript library method

```
patientportal.patients.getManagers ({
        patient: <patient>
});
```

HTTP Method

GET

Url

/patientportalapi/patients/managers

URL Parameters

patient	string	The patient key.

Returned JSON

```
{
    "status": "ok",
    "result": {
        <list of PatientDemographicData>
    }
}
```

32 Absence questionnaire

32.1 SendValidationSMS

Sends a validation code to the mobile phone number of the patient who is the focus of the absence record specified by the absence key.

JavaScript library method

```
patientportal.rtwQuestionnaire.sendValidationSMS ({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/rtw-questionnaire/send-validation-sms

URL Parameters

key	string	The absence key.

Returned JSON

```
{
    "status": "ok",
    "result": “”
}
```

32.2 SubmitValidationCode

Validates the patient using the code sent to them via a SMS message received because of a request sent to the SendValidationSMS method.

JavaScript library method

```
patientportal.rtwQuestionnaire.submitValidationCode ({
        key: <key>.
        code: <code>
});
```

HTTP Method

GET

Url

/patientportalapi/rtw-questionnaire/submit-validation-code

URL Parameters

key	string	The absence key.
code	string	The code sent to the patient via a SMS message received because of a request sent to the SendValidationSMS method.

Returned JSON

```
{
    "status": "ok",
    "result": {
        <AbsenceRTWQuestionnaireData>
    }
}
```

32.3 ValidateKey

Checks whether the code sent to the patient because of a request to the SendValidationSMS method is still valid.

JavaScript library method

```
patientportal.rtwQuestionnaire.validateKey ({
        key: <key>
});
```

HTTP Method

GET

Url

/patientportalapi/rtw-questionnaire/validate-key

URL Parameters

key	string	The absence key.

Returned JSON

```
{
    "status": "ok",
    "result": “”
}
```

or:

```
{
    "status": "error",
    "error": {
        "Message": "Validation code has expired. Please re-send new validation
code.",
        "EventType": 2,
        "StatusCode": 500,
        "EventCode": 0
    }
}
```

32.4 SubmitQuestionnaire

Checks whether the code sent to the patient because of a request to the SendValidationSMS method is still valid.

JavaScript library method

```
patientportal.rtwQuestionnaire.submitQuestionnaire ({
        rtwQuestionnaireData: <rtw-questionnaire-data>
});
```

HTTP Method

POST

Url

/patientportalapi/rtw-questionnaire/submit-questionnaire

URL Parameters

rtw-questionnaire-data	AbsenceRTWQuestionnaireData	An AbsenceRTWQuestionnaireData object, as received from the SubmitValidationCode method.

Returned JSON

```
{
    "status": "ok",
    "result": “”
}
```

33 Companies

33.1 Insurers

Returns a list of insurers available for the current chamber. Only insurers marked as ‘public’ are returned.

JavaScript library method

PatientPortalCompanies.prototype.getInsurers ();

HTTP Method

GET

Url

/patientportalapi/companies/insurers

Returns

CompanyData[]

34 Ethnicities

34.1 Ethnicities

Returns a list of the available ethnicities to select from when updating the patient's details.

JavaScript library method

patientportal.ethnicities.getEthnicities ();

HTTP Method

GET

Url

/patientportalapi/ethnicities/ethnicities

URL Parameters

Key	string	The absence key.

Returns

`EthnicityData[]`

35 Referrers

35.1 Referrers

Returns a list of the available referrers to select from when updating the patient’s details.

JavaScript library method

patientportal.referrers.getReferrers ();

HTTP Method

GET

Url

/patientportalapi/referrers/referrers

URL Parameters

Key	String	The absence key.

Returns

`ReferrerData[]`

36 Case Management

This section provides methods to work with Case Management – getting a list of cases, detailed case information and updating cases in various ways.

36.1 GetCases

Returns a list of cases filtered by any of the optional parameters specified.

JavaScript library method

```
patientportal.cases.getCases({
            patient: <patient>,
            text: <text>,
            status: <status>,
            case: <case>,
            title: <title>,
            patientName: <patient-name>,
            department: <department>,
            division: <division>,
            pageSortColumn: <page-sort-column>,
            pageSortDescending: <page-sort-descending>,
            pageNumber: <page-number>,
            pageSize: <page-size>
});
```

HTTP Method

GET

Url

/patientportalapi/cases/cases

URL Parameters

patient	string (optional)	The patient’s key id.
text	string (optional)	Checks if any of the attributes of the CaseOverviewData object contain <text>.
status	int (optional)	0. No filter (Default) 1. Open cases 2. Closed cases
case	Int (optional)	The case’s key id.
title	string (optional)	The public title of the case.
patientName	string (optional)	The patient’s name.
department	string (optional)	Key of the department provided by the API upon GetDepartments.
division	string (optional)	Key of the division provided by the API upon GetDepartments.
page-sort-column	int (optional)	The column index to sort the result: 0. Case Key 1. Person Name 2. Case Title 3. Opened Date 4. Closed Date 5. Last Updated Date By default the result is sorted by last updated date.
page-sort-descending	int (optional)	True to sort result descending.
page-number	int (optional)	Page number. Default 1.
page-size	int (optional)	Page size. Default 5. Minimum 5. Maximum 50.

Returned JSON

```
{
    "Items": [
        <list of CaseOverviewData>
    ],
    "TotalCount": 15,
    "CurrentPage": 1,
    "PageSize": 10,
    "SortColumn": 0,
    "SortDescending": false
}
```

36.2 GetCase

Returns a case.

JavaScript library method

```
patientportal.cases.getCase({
        case: <case>
});
```

HTTP Method

GET

Url

/patientportalapi/cases/case

URL Parameters

case	string	The key of the case.

Returned JSON

`CaseData`

36.3 OpenCase

Opens a case and returns the newly opened case.

JavaScript library method

```
patientportal.cases.openCase({
        case: <case>
});
```

HTTP Method

GET

Url

/patientportalapi/cases/open-case

URL Parameters

case	string	The key of the case.

Returned JSON

`CaseData`

36.4 CloseCase

Closes a case and returns the newly closed case.

JavaScript library method

```
patientportal.cases.closeCase({
	case: <case>
});
```

HTTP Method

GET

Url

/patientportalapi/cases/close-case

URL Parameters

case	string	The key of the case.

Returned JSON

CaseData

36.5 FollowUp

Works the same way as the eponymously named endpoint on the referral API. Returns the case.

JavaScript library method

```
patientportal.cases.followUp({
	case: <case>,
	reason: <reason>
});
```

HTTP Method

GET

Url

/patientportalapi/cases/follow-up

URL Parameters

case	string	The key of the case.

POST Parameters

reason	string	The reason why the user sends the follow up referral.

Returned JSON

CaseData

36.6 DefaultTitles

Returns a list of the public case titles from the common catalogue.

JavaScript library method

```
patientportal.cases.defaultTitles();
```

HTTP Method

GET

Url

/patientportalapi/cases/default-titles

Returned JSON

string[]

37 Gender Identity

37.1 Genders

Returns a list of the available genders to select from when updating the patient’s details.

JavaScript library method

patientportal.genderIdentity.getGenders();

HTTP Method

GET

Url

/patientportalapi/gender-identity/genders

Returns

`GenderData[]`

37.2 Pronouns

Returns a list of the available pronouns to select from when updating the patient’s details.

JavaScript library method

patientportal.genderIdentity.getPronouns();

HTTP Method

GET

Url

/patientportalapi/gender-identity/pronouns

Returns

`PronounData[]`

38 Objects and data types

38.1 AbsenceData

A full absence record data object which includes full patient demographic data.

Properties

Key	string	The key of the absence record.
Patient	PatientDemographicData	The patient's demographic details.
StartDate	DateTime	The date that the absence started.
EstimatedEndDate	DateTime	The estimated return-to-work date.
EndDate	DateTime (optional)	The actual return-to-work date.
AbsenceStatusCode	int	The status code of the absence record. - 0 – Open - 1 – Deleted - 2– Closed
AbsenceStatusName	string	The name of the status code of the absence record, as specified in the *AbsenceStatusCode*.
QuestionnaireStatusCode	int	The status code of the absence. - 0 – Not Sent - 1 – Sent - 2– Completed
QuestionnaireStatusName	string	The name of the status code of the absence record's questionnaire request, as specified in the *QuestionnaireStatusCode*.
LostWork	LostWork	The details about the amount of work lost due to the absence.
Reason	string	The name of the reason for the absence.
AccidentAtWork	bool	Whether the absence was a result of an accident in the workplace.
ReasonSharedWithEmployer	bool	Whether the reason for the absence is to be shared with the employer.

JSON Example

```
{
    "Patient": {
        <PatientDemographicData>
    },
    "Key": "efdc81efac0fd1a7f7b1833697a3ec3b",
    "StartDate": "2018-06-11T00:00:00",
    "EstimatedEndDate": "2018-06-15T00:00:00",
    "EndDate": "2018-06-15T00:00:00",
    "AbsenceStatusCode": 2,
    "AbsenceStatusName": "Closed",
    "QuestionnaireStatusCode": 0,
    "QuestionnaireStatusName": "Not sent",
    "LostWork": {
        <LostWork>
    },
    "Reason": "Brain Tumour",
    "AccidentAtWork": false,
    "ReasonSharedWithEmployer": true
}
```

38.2 AbsenceOverviewData

A full absence record data object which includes the patient's full name.

Properties

Key	string	The key of the absence record.
PatientName	string	The name of the patient.
StartDate	DateTime	The date that the absence started.
EstimatedEndDate	DateTime	The estimated return-to-work date.
EndDate	DateTime (optional)	The actual return-to-work date.
AbsenceStatusCode	int	The status code of the absence record. - 0 – Open - 2– Closed
AbsenceStatusName	string	The name of the status code of the absence record, as specified in the *AbsenceStatusCode*.
QuestionnaireStatusCode	int	The status code of the absence. - 0 – Not Sent - 1 – Sent - 2– Completed
QuestionnaireStatusName	string	The name of the status code of the absence record's questionnaire request, as specified in the *QuestionnaireStatusCode*.
LostWork	LostWork	The details about the amount of work lost due to the absence.
Reason	string	The name of the reason for the absence.
AccidentAtWork	bool	Whether the absence was a result of an accident in the workplace.
ReasonSharedWithEmployer	bool	Whether the reason for the absence is to be shared with the employer.
DepartmentName	string	The name of the employee department that the patient belongs to.
DivisionName	string	The name of the employee division that the patient belongs to.

JSON Example

```
{
    "Key": "efdc81efac0fd1a7f7b1833697a3ec3b",
    "PatientName": "Vercruse, Katarina",
    “DepartmentName”, “Support”,
    “DivisionName”, “Information Technology”,
    "StartDate": "2018-06-11T00:00:00",
    "EstimatedEndDate": "2018-06-15T00:00:00",
    "EndDate": "2018-06-15T00:00:00",
    "AbsenceStatusCode": 2,
    "AbsenceStatusName": "Closed",
    "QuestionnaireStatusCode": 0,
    "QuestionnaireStatusName": "Not sent",
    "LostWork": {
        <LostWork>
    },
    "Reason": "Brain Tumour",
    "AccidentAtWork": false,
    "ReasonSharedWithEmployer": true
}
```

38.3 AbsenceRTWQuestionnaireData

The data returned in the response of a call to the SubmitValidationCode method, to be modified if necessary and posted back to the API in a request to the SubmitQuestionnaire method.

Properties

Key	int	The key of the absence record.
StartDate	DateTime (read-only)	The date that the absence started.
EstimatedEndDate	DateTime (read-only)	The estimated return-to-work date.
EndDate	DateTime (optional)	The actual return-to-work date.
LostWork	LostWork	The details about the amount of work lost due to the absence.
AccidentAtWork	bool	Whether the absence was caused by an accident at work.
IsClosed	bool	Whether the absence has been closed.

JSON Example

```
{
    "status": "ok",
    "result": {
        "Key": "fc89c03626ab7f45ed3cf410ef1318b0",
        "StartDate": "2018-11-16T00:00:00",
        "EstimatedEndDate": "2018-11-19T00:00:00",
        "EndDate": "2018-11-26T00:00:00",
        "LostWork": {
            <LostWork>
        },
        "AccidentAtWork": true,
        "IsClosed": false
    }
}
```

38.4 ActivationConfirmation

Contains the information about the activated account.

Properties

OutstandingInvoice	InvoiceData	The invoice the patient needs to pay. Usually the membership fee. Can be null if there is no outstanding invoice.

JSON Example

```
{
  "OutstandingInvoice": {
      < See InvoiceData to see all properties. >
    }
}
```

38.5 AddressData

Address of the patient/company.

Properties

Address1	string	
Address2	string	
Address3	string	
City	string	
County	string	
PostCode	string	
Country	string	
Distance	decimal	Distance in miles from another geographical coordinate. Used within GetSites to provide distance from patient's home/work address or GPS coordinate.

JSON Example

```
{
  "Address1": "127 Northchurch Rd",
  "Address2": "Borough of Islington",
  "Address3": "",
  "City": "London",
  "County": "",
  "PostCode": "N1 3PA",
  "Country": "United Kingdom",
}
```

38.6 AppointmentCancellationData

Appointment cancellation information.

Properties

AppointmentKey	string	Key of the appointment.
Fee	decimal	Cancellation fee that the patient must pay.
Detail	string	Any information about the cancellation.

JSON Example

```
{
  "AppointmentKey": "apt54898",
  "Fee": 30,
  "Detail": "Cancellations within 8 to 21 days before appointment date will incur
a 50% cancellation fee, between 0 and 7 days will incur a 100% cancellation fee."
}
```

38.7 AppointmentData

Information about the appointment.

Properties

<table>
<tr><td>Key</td><td>string</td><td>Key of the existing appointment.</td></tr>
<tr><td>Type</td><td>AppointmentTypeData</td><td>Type of appointment.</td></tr>
<tr><td>Start</td><td>DateTime</td><td>Start time.</td></tr>
<tr><td>Finish</td><td>DateTime</td><td>Finish time.</td></tr>
<tr><td>Site</td><td>ShallowSearchResultData
A shallow search result of a patient/company address.
<b>Properties</b>
<table>
<tr><td>Key</td><td>string</td></tr>
<tr><td>Address</td><td>string</td></tr>
</table>
<b>JSON Example</b>
<pre>
{
  "Key": "fc560b40315dc7605fd5ca53e0dcaabc357c69bea3faefa8c6e2ce8129909061956910b77338ee2c2cdbb1c7c5f7c64bcf338d78bc148f81f6786152d3ef2987b3ab5b1e5588b1db7939bb5e0edffec4614c4511c4a7a0dfd9bc9077749482b152217c572b0f78552c75be542ffcea6446110af6da78213c1f71569f35abab7d65f82f382f8b8dc663c8e6a1405bf17c331d379f375ffbc6ec3ebc21a985a69355d10622db48eceb7f23b38c5037ed2315c3d858268baae1879f6f84b3b65586742086832ec398acdfd56680a72991d7bb38bbfd1fa61991aebf0bd1982dc06b",
  "Address": " 2 Paradise Street, Liverpool, Merseyside, L1 8JF"
}
</pre>
<b>38.8</b>
SiteData</td><td>Where the appointment is placed.</td></tr>
<tr><td>Location</td><td>LocationData</td><td>Specific location within the site.</td></tr>
<tr><td>Patient</td><td>PersonDemographicData</td><td>The patient data.</td></tr>
<tr><td>Clinician</td><td>ClinicianData</td><td>Clinician of this appointment.</td></tr>
</table>

State	string	The state code of the appointment. Possible values: - NotArrived - Cancelled - DNA - Discharged - BeingSeen - Arrived - Booked
StateDisplayName	string	The friendly name of the appointment state.
StateColor	string	The preferred colour of the appointment state. Red for Cancelled and DNA. Otherwise green.
Modules	AppointmentModuleData[]	Modules related to the appointment.
Services	ServiceData[]	Services related to the appointment
Slots	AppointmentSlot[]	List of appointment's slots. Used for booking.
PrimaryAttendeeSlot	AppointmentSlot	Primary attendee slot. Used for booking.
Invoice	InvoiceData	Invoice related to this appointment. The client can provided information about payment status of the invoice and if the invoice is not paid the client can offer online payment.
Telemedicine	bool	Indicates whether the received appointment is telemedicine or not. On booking appointment the server ignores this parameter and books the appointment as telemedicine based on the TelemedicineOption in AppointmentTypeData
TelemedicineConnection	string	URL for telemedicine connection to be launched in browser window. Whilst booking an appointment, this is left empty
AuthorisationCode	string	The authorisation code provided when the insurer is the payer.

JSON Example

```
{
  "Key": "apt58462",
  "Type": {
    "Key": "CN",
    "Name": "Consultation",
  },
  "Start": "2013-08-07T09:30:00",
  "Finish": "2013-08-07T10:30:00",
  "Telemedicine": true,
  "TelemedicineConnection":
"http://Meddbase.sandboxga.com/flex.html?roomdirect.html&key=Wchof474qrCO2QLrNK1SE
MdrRY",
  "AuthorisationCode": "ABC123",
  "Site": {
    "Key": 1123,
    "Name": "2CP (Eye Room)",
    "Address":{
      "Address1": "2 Clifton Park Ave",
      "Address2": "",
      "Address3": "",
      "City": "London",
      "County": "",
      "PostCode": "SW20 8BD",
      "Country": "United Kingdom"
    }
  }
  "Location": {
    "Key": 214,
    "Name": "Room 1"
    "Address":{
      "Address1": "2 Clifton Park Ave",
      "Address2": "",
      "Address3": "",
      "City": "London",
      "County": "",
      "PostCode": "SW20 8BD",
      "Country": "United Kingdom"
    }
  },
  "Clinician": {
    "Key": 84,
    "Name": "Dr. House"
  },
  "Slots": [
    {
      < See AppointmentSlot to see all properties. >
    },
    {
      < See AppointmentSlot to see all properties. >
    }
  ],
  "PrimaryAttendeeSlot": {
    < See AppointmentSlot to see all properties. >
  },
  "Invoice": {
    "Date": "2013-08-02",
    "Number": "1865",
    "NetPrice": 200,
    "Vat": 40,
    "GrossPrice": 240,
    "Paid": 150,
    "Creditor": {
      "Name": "AXA PPP Healthcare",
      < See InvoiceData to see all properties. >
    },
    "Debtor": {
      "Name": "Mr. John Lemon",
      < See InvoiceData to see all properties. >
    }
  }
}
```

38.9 AppointmentModuleData

Module of the appointment.

Properties

Key	int	Key of the module.
Name	string	Name of the module.
Notes	string	
CurrencyCode	string	Three-letter currency code (e.g. "GBP", "EUR", "USD", etc.)
CurrencySymbol	string	Symbol for the currency
NetPrice	decimal	
Tax	decimal	
GrossPrice	decimal	

JSON Example

```
{
  "Key": 5631,
  "Name": "Short Consult",
  "CurrencyCode": "GBP",
  "CurrencySymbol": "£",
  "NetPrice": 50,
  "Tax": 10,
  "GrossPrice": 60
}
```

38.10 AppointmentSlot

Information about the appointment slot that the booking system needs to book the appointment.

Properties

Type	string	Type of appointment.
Start	DateTime	Start time.
Finish	DateTime	Finish time.
SiteKey	int	Key of the site the appointment is placed.
LocationKey	int	Key of the location within the site.
ResourceKey	int	Key of resources (stuff, rooms, etc.). Used for booking.

JSON Example

```
{
  "Type": "VC"
  "Start": "2013-08-07T09:30:00",
  "Finish": "2013-08-07T10:30:00",
  "SiteKey": 123,
  "LocationKey": 546,
  "ResourceKey": 468,
}
```

38.11 AppointmentTaskData

An additional details for the book/arrive appointment task.

Properties

AppointmentType	string	The type of the appointment that needs to be booked.
AppointmentTypeName	string	The name of the appointment type..
ServiceId	int	The optional service ID that needs to be included within the booking process.
IsBooked	bool	True if the appointment is already booked.
Clinician	string	The name of the clinician if the appointment is already booked.
StartDate	DateTime	The appointment start date if the appointment is already booked.
FinishDate	DateTime	The appointment finish date if the appointment is already booked.

38.12 AppointmentTypeData

Type of the appointment.

Properties

Key	string	
Name	string	
Notes	string	
CancellationPolicy	string	The cancellation policy message. This cancellation policy will apply within cancellation the existing appointment. Use the GetAppointmentCancellationInfo to retrieve exact cancellation message and fee for the concrete appointment at certain time.
Modules	AppointmentModuleData[]	Possible modules for the appointment type. If no modules are provided the appointment type doesn't contains modules.
CanBookAppointment	bool	Defines whether the logged in patient is allowed to book an appointment for this appointment type.
CanReferPatient	bool	Defines whether the logged in patient is allowed to refer patient for this appointment type.
TelemedicineOption	bool	Defines whether the appointment IS booked as a telemedicine appointment or not.
CanAddServices	bool	Defines whether the appointment type allows adding of additional services besides appointment modules.

JSON Example

```
{
  "Key": "CN",
  "Name": "Consultation"
  "CancellationPolicy": "You will be charged at 50% of the full price if you
     cancel the appointment within 72 hours. You will be charged at 90% of the
     full price if you do not turn up.",
  "CanBookAppointment": true,
  "CanReferPatient": false,
  "TelemedicineOption": true,
  "CanAddServices": false
}
```

38.13 AttachDocumentTaskData

An additional details for the attach document task.

Properties

IsAttached	bool	True if the document has already been attached.
DocumentName	string	The name of the attached document.
DocumentUrl	string	The URL of the attached document.

38.14 AuthenticationData

Contains the information about current authenticated security context.

Properties

SessionID	string	Identifies the existing security context.
Config	ConfigData	Defines that the patient should be forced to set up their contact options.
Token	string	The current token. See Authentication Token.
Require2FA	Boolean	True if the profile requires 2FA (see Send2faCode). False if the profile does not require 2FA.

JSON Example

```
{
  "SessionID": "4oytawakgy0njpaajyliq3md",
  "Config": "<see ConfigData>",
  "Token": "7a413802-f186-42c6-ae92-646531749624",
  "Require2FA": "true"
}
```

38.15 AuthorizeTelemedicineData

Returns authorisation data for a telemedicine connection.

Properties

Token	string	Secure token generated for the user joining the telemedicine conference.
Version	int	A version of the telemedicine integration. Currently always version 3 which means Twilio.

JSON Example

```
{
  "Version": "3",
  "Token": "ZS1Q8LUazO3Q2fRI...",
}
```

38.16 CaseData

A full case data object which includes full patient demographic data.

Properties

Key	string	The unencrypted case key.
Patient	PatientDemographicData	The patient's demographic details.
OpenedDate	DateTime	The date the case was opened.
ClosedDate	DateTime (optional)	The date the case was closed, if at all.
Title	string	The public case title.
IsOpen	bool	Whether the case is still open or not.

38.17 CaseOverviewData

A full case data object which includes the patient's full name.

Properties

Key	string	The unencrypted case key.
PatientName	string	The name of the patient.
OpenedDate	DateTime	The date the case was opened.
ClosedDate	DateTime (optional)	The date the case was closed, if at all.
LastUpdatedDate	DateTime (optional)	The date that the latest change was made to any of the referrals on the case. If there are no referrals on the case, this date is null.
Title	string	The public case title.
IsOpen	bool	Whether the case is still open or not.

38.18 ChargeBandData

Provides information about a chargeband.

Properties

Name	string	Name of the chargeband.
RegCode	string	Online signup code of the chargeband.
CompanyType	string	Type of the company to which the signup code belongs
ProvideDeptsAndDivs (Optional)	bool	Flag that shows if registering person is allowed to see and select department and division. The value is only returned in patient portal.

JSON Example

```
{
  "Name": "Online chargeband",
  "RegCode": "d01m",
  "CompanyType" : "M"
}
```

38.19 ChoiceTaskData

An additional details for the choice task.

Properties

Options	string[]	List of options. The index starts from 1.
SelectedIndex	int	The selected option. -1 if not selected.

38.20 ClinicianData

Information about the clinician.

Properties

Key	int	
SexType	string	Sex: - 0 = any - 1 = male - 2 = female
FullName	string	

JSON Example

```
{
  "Key": 84,
  "FullName": "Dr. House"
}
```

38.21 CompanyData

Data returned in response to call to Insurers. Only companies marked as ‘Public’ are returned.

Properties

Key	string	Encrypted key of the company
Name	string	Public name of the company
Type	String	Type of the company: Insurer, etc.
IsPublic	Bool	True: Only public companies are returned

JSON Example

```
  {
    "Key": "1c5aabbdf813ad78c48cc8e6d8584162",
    "Name": "Public Insurance",
    "Type": "Insurer",
    "IsPublic": true
  }
```

38.22 CompanyDemographicData

Contains the company’s demographic information.

Properties

Code	string	
Name	string	
VatNo	string	
Address	AddressData	
Contact	ContactDemographicData	

JSON Example

```
{
  "CODE": "AXA",
  "Name": "AXA PPP Healthcare",
  "Address": {
    "Address1": "44 Pall Mall",
    "City": "London",
    "PostCode": "SW1Y",
    "Country": "United Kingdom"
  },
  "Contact": {
    "Title": "Mr",
    "Name": "Robert Murian",
    "Mobile": "0795523411",
    "Telephone": "+444 525 111 555",
    "EmailAddress": "robert@comp.com",
    "Address": {
      "Address1": "Pall Mall",
      "Address2": "",
      "Address3": "",
      "City": "London",
      "County": "",
      "PostCode": "SW1Y",
      "Country": "United Kingdom"
    }
  }
}
```

38.23 ConfigData

Provides informations about the current session, company and logged-in user.

Properties

IsLoggedIn	bool	True if the user is logged in.
CompanyName	string	Name of the current company.
Phone	string	Official Company phone number.
Permissions	object	List of patient's/user's permissions.
Configuration	object	General app configuration.
Configuration\SSOStatus	SSOStatus	Optional single sign on configuration
Configuration\SessionTimeout	Int	The number of seconds from the last operation the session will be active for. Use ValidateLogin to get remaining time of the session lifetime.
Configuration\ DateOfBirthRequiredForPatients	bool	True if DOB is mandatory for a patient registration.
ForceContactOptions	bool	Defines that the user should be forced to set up their contact options.
ForceTermsAndConditions	bool	Defines that the user should be forced to accept the Terms and conditions. See also GetTermsAndConditions.

JSON Example

```
{
  "IsLoggedIn": true,
  "CompanyName": "Test Company",
  "Phone": "077123456789",
  "Permissions": {
    "Patient": {
      "CanBookAppointment": false,
      "CanViewAppointments": false,
      "CanCancelAppointment": false,
      "CanViewInvoices": false,
      "CanPayInvoice": false,
      "CanViewMedicalHistory": false,
      "CanManageQuestionnaires": false,
      "CanManageFeeds": false,
      "CanViewCompanyLibrary": false,
      "CanViewPathways": false,
      "CanChangePassword" : false

    },
    "User": {
      "CanManageUsers": true,
      "CanViewReferrals": true,
      "CanViewReferralReports": true,
      "CanReferPatient": true,
      "CanBulkReallocateReferrals": true,
      "CanViewRecall": true,
      "CanViewDocument": true,
      "CanViewAppointment": true,
      "CanChangePassword" : true,
      "CanViewPathways" : true
    }
  }
  "Configuration": {
      "OnlinePaymentsEnabled": true,
      "DateOfBirthRequiredForPatients": true,
      "SSOStatus": {
             "Identifier": "shiny",
             "Enabled": true,
             "IsOH": true
      },
      "PasswordPolicy": {
             "Length": 10,
             "Numbers": 1,
             "Letters": 1,
             "NonAlphaNumeric": 0,
             "MixCaps": true,
             "Description": "at least 10 characters in length, contain at least
             one letter an done number and must mix upper and lower-case letters"
      },
      "DefaultCallingCode": "+44",
      "NhsConsentEnabled": false,
      "SessionTimeout": 120000,
      "DefaultPaymentCountry": "GB"
   },
  "ForceContactOptions": false,
  "ForceTermsAndConditions": false
}
```

38.24 ContactDemographicData

Contains the contact’s demographic information.

Properties

Title	string	
Name	string	
Details	string	
Fax	string	
Mobile	string	
Telephone	string	
EmailAddress	string	
Address	AddressData	

JSON Example

```
{
  "Title": "Mr",
  "Name": "Robert Murian",
  "Mobile": "0795523411",
  "Telephone": "+444 525 111 555",
  "EmailAddress": "robert@comp.com",
  "Address": {
    "Address1": "Pall Mall",
    "Address2": "",
    "Address3": "",
    "City": "London",
    "County": "",
    "PostCode": "SW1Y",
    "Country": "United Kingdom"
  }
}
```

38.25 CountryData

Contains ISO 3166 country data.

Properties

Key	string	
Name	string	

JSON Example

```
{
  "Key": "GB",
  "Name": " United Kingdom of Great Britain and Northern Ireland(the)"
}
```

38.26 CurrencyData

Properties

Code	string	Three-letter currency code (e.g. "GBP", "EUR", "USD", etc.)
Symbol	string	Symbol for the currency.

JSON Example

```
{
  "Code": "GBP",
  "Symbol": "£"
}
```

38.27 CurrentMembershipSchemeData

Provides information about the current membership scheme of the patient.

Properties

Name	string	Name of the scheme.
Code	string	Membership code of the scheme.
BillingFrequency	string	Billing frequency of current scheme. Weekly, monthly, etc.
CurrencyCode	string	Three-letter currency code (e.g. "GBP", "EUR", "USD", etc.)
CurrencySymbol	string	Symbol for the currency
NetPrice	decimal	
Tax	decimal	
GrossPrice	decimal	
RequiresOnlinePayment	bool	Scheme requires an online recurring payment method set up
OnlinePaymentAllowed	bool	The scheme allows online recurring payments.
JoinedDateTime	DateTime	The date patient joined the scheme.
NextInvoiceDateTime	DateTime	Date of next billing cycle
OnlinePaymentMethod	string	Associated online payment method (formatted string)
LastInvoiceBalance	decimal	Balance on the last invoice for the memberhsip
LastInvoiceKey	String	Encrypted key of the last invoice.

JSON Example

```
{
  "Name": "Basic Membership Program",
  "Code": "ms001",
  "BillingFrequency": "Monthly",
  "CurrencyCode": "GBP",
  "CurrencySymbol": "£",
  "NetPrice": 100,
  "Tax": 20,
  "GrossPrice": 120,
  "RequiresOnlinePayment": false,
  "OnlinePaymentAllowed": false,
  "JoinedDateTime": "2016-12-16T16:13:40.747",
  "NextInvoiceDateTime": "2017-01-17T00:00:00",
  "OnlinePaymentMethod": "VISA (****0014) Expiry:  01/28",
  "LastInvoiceBalance": 160
  "LastInvoiceKey" : ""2e0bba51a59f90a4c23be92ca6c89b0a"
}
```

38.28 DateTime

DataTime format uses standard ISO 8601: "yyyy-MM-ddTHH:mm:ss.ffffff±HH:mm".

If off-set is not specify the Meddbase servers uses London's offset what is 00:00 in winter time, +01:00 in summer time.

Examples:

```
a) 8. January 2013 = "2013-01-08T00:00:00"
b) 1. February 2013, 13:2:25 = "2013-02-01T13:02:25"
c) 1. August 2013, 18:25:5.9525711 = "2013-08-01T18:25:05.9525711"
d) 1. August 2013, 18:25:5.9525711 = "2013-08-01T18:25:05.9525711+01:00"
      o  Note: case c) and d) are same because in summer time the default
         offset for the Meddbase servers is +01:00
```

38.29 DepartmentData

Contains the information about the company department.

Properties

Key	string	
Name	string	
Divisions	DivisionData[]	
ParentKey	int	The ID of the parent department, if it has one

JSON Example

```
{
  "Key": "D123",
  "Name": "HR"
}
```

38.30 DivisionData

Contains the information about the company division.

Properties

Key	string	
Name	string	

JSON Example

```
{
  "Key": "Dv123",
  "Name": "Division 123"
}
```

38.31 DocumentData

Provides information about the document.

Properties

Name	string	The name of the document.
Author	User	The type of the user account.
Comments	string	Short description or comment.
DateCreated	DateTime	Created date.
Url	string	Url to download the document.
MIMEType	string	MIME type of the document.
Size	long	Size in bytes. Can be null.
PatientKey	String	The key of the patient.
DocumentTypeName	String	The name of the document type the document belongs to. Provided only if the *DocumentTypeSecurityEnable* flag is set (see GetConfig)

JSON Example

```
{
  "Name": "Referral letter.html",
  "Author": "Mr. Will Smith",
  "Comments": "I would like to refer Mr. John Smith",
  "DateCreated": "2015-03-13T14:22:12.483",
  "Url": "https://api.meddbase.com/referrals/download?d=123",
  "MIMEType": "text/html",
  "Size": 20824,
  "PatientKey": "iydksy58dujyhiee78",
  "DocumentTypeName": "Portal Documents"
}
```

38.32 DocumentType

Provides information about the document type.

Properties

Key	string	The key of the document type.
Name	string	The name of the document type.
AlwaysAccessible	bool	True if the type is built-in and always accessible. This is usually ‘Portal Documents’ because it is used for uploading and all managers has to have access to that type. Note this applies if the *DocumentTypeSecurityEnable* flag is set (see GetConfig)

JSON Example

```
{
  "Key": "22c45458525be86005e08d219b79c5d7",
  "Author": "Portal Documents",
  "AlwaysAccessible": true
}
```

38.33 EmployeeStatus

Contains the information about the employee status.

Properties

Key	string	
Name	string	

JSON Example

```
{
  "Key": "1",
  "Name": "Employed"
}
```

38.34 EthnicityData

Data returned in response to the `getEthnicity` method

Properties

Key	string	Encrypted key of the ethnicity
Name	String	The name of the ethnicity

38.35 FeedData

Feed of the feeds system.

Properties

Key	string	Key of the feed.
StartDate	DateTime	Day and time when the feed was created.
AuthorName	string	Name of the author who created this feed.
LastMessage	FeedMessageData	The last message only.
Messages	FeedMessageData[]	All messages sorted by date descending. The message on the first position is the latest message of the feed.
OutstandingMessageNotification	NotificationData	Outstanding message notification is present if there is a new message in the feed. Property is null or undefined if there is no new message in the feed. Use ConfirmNotification provided by the API to confirm this notification.

JSON Example

```
{
  "Key": "f562874",
  "StartDate": "2013-08-02T14:45:58.9525711+01:00",
  "AuthorName": "Dr. Ben",
  "LastMessage": {
    "AuthorName": "Dr. Ben",
    "Date": "2013-08-02T14:55:15.1875264+01:00",
    "Text": "<div>Thanks John</div>"
  },
  "Messages": [
    {
      "AuthorName": "Dr. Ben",
      "Date": "2013-08-02T14:55:15.1875264+01:00",
      "Text": "<div>Thanks John</div>"
    },
    {
      "AuthorName": "John",
      "Date": "2013-08-02T14:50:05.5684528+01:00",
      "Text": "Yes, I can."
    },
    {
      "AuthorName": "Dr. Ben",
      "Date": "2013-08-02T14:45:58.9525711+01:00",
      "Text": "<div><b>Hi John, could you...</b></div>"
    }
  ],
  "OutstandingMessageNotification": {
    "Key": "f-1548",
    "Type": 8,
    "Message": "New message from Dr. Ben: <div>Thanks John</div>",
    "ObjectKey": "f562874"
  }
}
```

38.36 FeedMessageData

One message in the message feed.

Properties

AuthorName	string	Name of the author of this message.
Date	DateTime	Date when the message was created.
Text	string	Text of the message. HTML or plain text.

JSON Example

```
{
  "AuthorName": "Dr. Ben",
  "Date": "2013-08-02T14:45:58.9525711+01:00",
  "Text": "<div><b>Hi John</b></div>"
}
```

38.37 GenderData

Represents a person’s gender identity, NOT their sex at birth.

Properties

Key	string	The encrypted key of the gender.
Name	string	The name of the gender.

JSON Example

```
{
  "Key": "1c5aabbdf813ad78c48cc8e6d8584162",
  "Name": "Female"
}
```

38.38 InsurerData

Data returned in response to call to Insurers. Only companies marked as 'Public' are returned.

Properties

Key	string	Encrypted key of the company
Name	string	Public name of the company (empty if company is private)
Type	String	Type of the company: Insurer, etc.
IsPublic	Bool	Company is public or private
MemberNumber	String	Member number of the patient

JSON Example

```
  {
    "Key": "1c5aabbdf813ad78c48cc8e6d8584162",
    "Name": "Public Insurance",
    "Type": "Insurer",
    "IsPublic": true,
    "MemberNumber": "Test member number"
  }
```

38.39 InvoiceAddressData

Contains information about the invoice's creditor or debtor.

Properties

Name	string	Name of the account.
IsCompany	bool	True if the account is a company. False if the account is a patient.
Address	AddressData	Account address.
Account	CompanyDemographicData PersonDemographicData	A demographic data of the account. If the IsCompany is true, the type of this property is CompanyDemographicData, else the type is PersonDemographicData.

JSON Example

```
{
  "Name": "Mr. John Lemon",
  "IsCompany": false,
  "Address": {
    "Address1": "Studio 99",
    "Address2": "Backlok Street",
    "Address3": "Camden",
    "City": "London",
    "County": "",
    "PostCode": "N1 7NK",
    "Country": "United Kingdom"
  },
  Account: {
    "Title": "Mr",
    "Name": "John",
    "Surname": "Lemon",
    "SexType": 1,
    "Initials": "JL",
    "DateOfBirth": "1958-08-02T00:00:00",
    "Mobile": "+444 895 523 411",
    "Telephone": "+444 525 111 555",
    "EmailAddress": "john.lemon@test.com",
    "Address": {
      "Address1": "Studio 99",
      "Address2": "Backlok Street",
      "Address3": "Camden",
      "City": "London",
      "County": "",
      "PostCode": "N1 7NK",
      "Country": "United Kingdom"
    }
  }
}
```

38.40 InvoiceCreditNoteData

Contains information from one credit note of an invoice.

Properties

Key	string	Key of the item.
Date	DateTime	When was a credit note raised.
ServiceName	string	Name of the service the credit note was raised for. If empty the credit note is raised for whole invoice.
Comments	string	A reason why was this credit note raised.
CurrencyCode	string	Three-letter currency code (e.g. "GBP", "EUR", "USD", etc.)
CurrencySymbol	string	Symbol for the currency
NetPrice	decimal	
Tax	decimal	
GrossPrice	decimal	

JSON Example

```
{
  "Key": "156480",
  "Date": "2013-10-05T00:00:00",
  "ServiceName": "Consultation",
  "Comments": "Cancellation",
  "CurrencyCode": "GBP",
  "CurrencySymbol": "£",
  "NetPrice": 100,
  "Tax": 20,
  "GrossPrice": 120,
}
```

38.41 InvoiceData

Contains data of the invoice.

Properties

Key	string	Key of the invoice.
Date	DateTime	Date when was the invoice raised.
Number	int	Number of the invoice. The client can use it to provide a payment.
CurrencyCode	string	Three-letter currency code (e.g. "GBP", "EUR", "USD", etc.)
CurrencySymbol	string	Symbol for the currency
TotalNet	decimal	
Tax	decimal	
TotalGross	decimal	
Paid	decimal	How much is already paid.
Items	InvoiceItemData[]	Invoice items.
CreditNotes	InvoiceCreditNoteData[]	Credit notes.
Creditor	InvoiceAddressData	
Debtor	InvoiceAddressData	
PayableOnline	bool	If the invoice is payable online (the billing company has an online payment account.)

JSON Example

```
{
  "Key": "inv4645",
  "Date": "2013-02-06",
  "Number": "1865",
  "CurrencyCode": "GBP",
  "CurrencySymbol": "£",
  "TotalNet": 200,
  "Tax": 40,
  "TotalGross": 240,
  "Paid": 150,
  "Creditor": {
    "Name": "AXA PPP Healthcare",
    "IsCompany": true,
    "Address": {
      "Address1": "44 Pall Mall",
      "City": "London",
      "PostCode": "SW1Y",
      "Country": "United Kingdom"
    },
    "Account": {
      "CODE": "AXA",
      "Name": "AXA PPP Healthcare",
      "Address": {
        "Address1": "44 Pall Mall",
        "City": "London",
        "PostCode": "SW1Y",
        "Country": "United Kingdom"
      }
    }
  },
  "Debtor": {
    "Name": "Mr. John Lemon",
    "IsCompany": false,
    "Address": {
      "Address1": "Studio 99",
      "Address2": "Backlok Street",
      "Address3": "Camden",
      "City": "London",
      "County": "",
      "PostCode": "N1 7NK",
      "Country": "United Kingdom"
    },
    Account: {
      "Title": "Mr",
      "Name": "John",
      "Surname": "Lemon",
      "SexType": 1,
      "Initials": "JL",
      "DateOfBirth": "1958-08-02T00:00:00",
      "Mobile": "+444 895 523 411",
      "Telephone": "+444 525 111 555",
      "EmailAddress": "john.lemon@test.com",
      "Address": {
        "Address1": "Studio 99",
        "Address2": "Backlok Street",
        "Address3": "Camden",
        "City": "London",
        "County": "",
        "PostCode": "N1 7NK",
        "Country": "United Kingdom"
      }
    }
  },
  "PayableOnline": true
}
```

38.42 InvoiceItemData

Contains information from one row of an invoice.

Properties

Key	string	Key of the item.
TypeName	string	Full name of the item type (e.g. Appointment, Module, Product, etc.). Item type specify a group that the item belongs to.
Name	string	Name of the item.
Code	string	Code of the item.
CurrencyCode	string	Three-letter currency code (e.g. "GBP", "EUR", "USD", etc.)
CurrencySymbol	string	Symbol for the currency
NetPrice	decimal	
Tax	decimal	
GrossPrice	decimal	

JSON Example

```
{
  "Key": "5642",
  "TypeName": "Appointment",
  "Name": "Consultation",
  "Code": "123",
  "CurrencyCode": "GBP",
  "CurrencySymbol": "£",
  "NetPrice": 100,
  "Tax": 20,
  "GrossPrice": 120,
}
```

38.43 LocalDateTime

Same like DateTime but without information about a time zone. It is used for example to search for an appointment slot across time zones.

Imagine this situation: the client is based in -6 zone but runs clinics also in both -7 and -6 zones. The patient is currently in -6 zone (or on holiday in -9 zone) and is looking for an appointment at 10am. For the patient 10am is local time and he expects slots in all three zones to start at 10am at site time.

Required formatting: *"yyyy-MM-ddTHH:mm:ss"*.

Example:

```
February 2013, 13:2:25 = "2013-02-01T13:02:25"
```

38.44 LocationData

The location within the site.

Properties

Key	int	
Name	string	
Address	AddressData	

JSON Example

```
{
  "Key": 214,
  "Name": "Room 1"
  "Address":{
    "Address1": "2 Clifton Park Ave",
    "Address2": "",
    "Address3": "",
    "City": "London",
    "County": "",
    "PostCode": "SW20 8BD",
    "Country": "United Kingdom"
  }
}
```

38.45 LostWork

A small object used in the AbsenceData and AbsenceOverviewData to encapsulate the number of hours and days lost due to the patient's absence. The **Hours** and **Days** in the object are separate, in that they do not mean that *the patient was absent for four days and 32 hours*, they mean that *the patient was absent for four days, which is equivalent to 32 hours*.

Properties

Hours	decimal (optional)	The number of hours that the patient was absent.
Days	decimal (optional)	The number of days that the patient was absent.

JSON Example

```
{
    "Hours": 32,
    "Days": 4
}
```

38.46 ManagerReportReviewData

Provides information about the manager review.

Properties

ReportUrl	string	The URL address of the full medical report. The report is PDF document. We recommend to use <object> tag to display embedded PDF: <object type="application/pdf" data="<url>"></object>
ReferredBy	string	The name of the referrer.
Referral	ReferralData	The referral data.

JSON Example

```
{
  "ReportUrl": "http://api.meddbase.com/patientportalapi/....",
  "ReferredBy": "Mr. John Smith",
  "Referral": {<...see ReferralData...>},
}
```

38.47 MedicalHistoryNodeData

Provide information about one node of the Medical History tree.

Properties

DataType	int	Type of data: - 0 = Text - 1 = Group (can contain child nodes) - 2 = HTML - 3 = Document - 4 = Graph
Label	string	Labe of the current node.
Detail	string	Detail of the current node.
Path	string	Path of the current node. Can be null or empty if it is fully loaded (*Loaded* property is *True*) and it is not possible to request this node alone. It is usually for *Separator*, *HTML*, *Title* or *Text* data types.
Loaded	bool	True if *Data* and *Children* properties are fully loaded. The server doesn't always return a *Children* and *Data* properties fully loaded. It happens when the response would be excessively large or for performance reasons. The server doesn't return it so the client has a faster response. If the patient wants to see it the client must request the specific node using GetMedicalHistoryTreeNode and pass the *Path* property as the *NodePath*.
Data	object	Contains data of the node. Specific for every *DataType*. See all examples below to find out how data looks.
Children	MedicalHistoryNodeData[]	Children of the current node. Only *Group* data type item has children.

JSON Example for Text data

```
{
  "DataType": 0,
  "Loaded": true,
  "Data": "Date: 01/01/2013\r\nSubject: Medical Examination\r\nRef/nr: ABC-
1231243\r\n\r\nName: Mister X\r\nAge: 30\r\n\r\nGender: Male"
}
```

JSON Example for not fully loaded Group data

```
{
  "DataType": 1,
  "Label": "Full medical test",
  "Details": "16/4/2013",
  "Path": "/appointment/2756",
  "Loaded": false
}
```

JSON Example for fully loaded Group data

```
{
  "DataType": 1,
  "Label": "Full medical test",
  "Details": "16/4/2013",
  "Path": "/appointment/2756",
  "Loaded": true,
  "Children": [
    {
      "DataType": 0,
      "Loaded": true,
      "Data": "Dear John, there is your blood test result."
    },
    {
      "DataType": 3,
      "Label": "Blood test result",
      "Loaded": true,
      "Data": {
        "Size": "1256856",             < The file size in bytes. >
        "MIMEType": "image/jpeg",
        "Url": "https://images.meddbase.com/image.jpg"
      }
    },
    {
      "DataType": 0,
      "Loaded": true,
      "Data": "And there is your BMI graph."
    },
    {
      "DataType": 4,
      "Label": "BMI Graph",
      "Details": "",
      "Path": "/appointment/2756/bmi-graph",
      "Loaded": false
    }
  ]
}
```

JSON Example for HTML data

```
{
  "DataType": 2,
  "Loaded": true,
  "Data": "<p>Date: 01/01/2013<br/>Subject: Medical Examination<br/>Ref/nr: ABC-
1231243<br/><br/>Name: Mister X<br/>Age: 30<br/>Gender:
Male</p><hr/><table><tr><td>Table</td><td>Example</td></tr></table>"
}
```

JSON Example for Document data

```
{
  "DataType": 3,
  "Label": "Blood test result",
  "Loaded": true,
  "Data": {
    "Size": "48123",                   < The file size in bytes. >
    "MIMEType": "image/jpeg",
    "Url": "https://images.meddbase.com/image.jpg"
  }
}
```

Note: The client must provide *ASP.NET_SessionId* key in the cookie to receive the file.

JSON Example for Document data

```
{
  "DataType": 3,
  "Label": "Pathology result",
  "Loaded": true,
  "Data": {
    "Size": "1256856",                 < The file size in bytes. >
    "MIMEType": "application/pdf",
    "Url": "https://docs.meddbase.com/pathology-result.pdf"
  }
}
```

Note: The client must provide *ASP.NET_SessionId* key in the cookie to receive the file.

JSON Example for Graph data

```
{
  "DataType": 4,
  "Label": "BMI Graph",
  "Details": "",
  "Path": "/num-data/123",
  "Loaded": true,
  "Data": {
    "XAxis": {
      "Title": "Years",
      "Data": [2011, 2012, 2013]
    },
    "Series": [
      {
        "Title": "John",
        "Data": [25.4, 30.4, 27.6],
        "Color": "Red"
      }
    ]
  }
}
```

38.48 MembershipSchemeData

Provides information about a membership scheme.

Properties

Name	string	Name of the scheme.
Code	string	Membership code of the scheme.
BillingFrequency	string	Billing frequency of current scheme. Weekly, monthly, etc.
CurrencyCode	string	Three-letter currency code (e.g. "GBP", "EUR", "USD", etc.)
CurrencySymbol	string	Symbol for the currency
NetPrice	decimal	
Tax	decimal	
GrossPrice	decimal	
RequiresOnlinePayment	bool	Scheme requires an online recurring payment method set up
OnlinePaymentAllowed	bool	The scheme allows online recurring payments.

JSON Example

```
{
  "Name": "Basic Membership Program",
  "Code": "ms001",
  "BillingFrequency": "Monthly",
  "CurrencyCode": "GBP",
  "CurrencySymbol": "£",
  "NetPrice": 100,
  "Tax": 20,
  "GrossPrice": 120,
  "RequiresOnlinePayment": false,
  "OnlinePaymentAllowed": true
}
```

38.49 NextOfKinDemographicData

Contains the person’s next of kin demographic information.

Properties

Relationship	string	
Name	string	
Surname	string	
EmailAddress	string	
Telephone	string	
Mobile	sting	
WorkTelephone	string	
Address	AddressData	

JSON Example

```
{
  "Relationship": "Mam",
  "Name": "Mariel",
  "Surname": "Lemon",
  "EmailAddress": "adam.lemon@test.com",
  "Mobile": "+444 895 523 222",
  "Telephone": "+444 525 111 333",
  "WorkTelephone": "+444 533 234 234",
  "Address": {
    "Address1": "Studio 99",
    "Address2": "Backlok Street",
    "Address3": "Camden",
    "City": "London",
    "County": "",
    "PostCode": "N1 7NK",
    "Country": "United Kingdom"
  }
}
```

38.50 NotificationData

Provides information about notification (outstanding invoice or questionnaire, etc.).

Properties

Key	string	Key of the notification.
Type	int	The type of the notification: - 1 = Appointment coming up soon - 2 = Outstanding questionnaire - 4 = Outstanding invoice - 8 = Outstanding messages - 16 = Action required - 32 = Referral notification
Message	string	Message of the notification.
ObjectKey	string	Key of the related object to this notification according to the Type of the notification (e.g. notification of type 2 ‘Outstanding questionnaire’ relates to the questionnaire so the object key means the questionnaire key. By the *Type* the *ObjectKey* means: - 1 = appointment key - 2 = questionnaire key - 4 = invoice key - 8 = feed key - 16 = ○ ‘SetContactOptions’ to force patient to set their contact options. ○ ‘AcceptTermsAndConditions’ to force patient to re-accept the Terms and conditions. - 32 = referral key The client can use this key to redirect/show more details about related objects.

JSON Example

```
{
  "Key": "2-1568",
  "Type": 1,
  "Message": "You need to complete pre-medical questionnaire.",
  "ObjectKey": "1568"
}
```

38.51 PathwayData

Provides full details about the pathway.

Properties

Key	string	The key of the pathway.
Number	string	The number of the pathway.
Name	string	The name of the pathway.
Patient	string	The name of the patient.
ReadOnly	bool	True if the pathway is currently in a read-only mode.
Tasks	PathwayTaskOverviewData[]	The list of pathways tasks.

JSON Example

```
{
  "Key": "a1d3e2d1",
  "Number": "123",
  "Name": "Book & Arrive",
  "Patient": "Mr. Will Smith",
  "ReadOnly": false,
  "Tasks": [
    {
      "Name": "GT",
      "ActionType": "GT",
      "ActionTypeName": "General task",
      "StateType": "ST",
      "StateName": "Started",
      "CanProcess": true,
      "Key": "62cf82832e12453faf8c9813f7817f7e",
    },
    {
      "Name": "AD",
      "ActionType": "AD",
      "ActionTypeName": "Attach a document",
      "StateType": "FT",
      "StateName": "Future task",
      "CanProcess": false,
      "Key": "df19272dc84b4f2c511ed2a04b616d1a",
    },
    {
      "Name": "BA",
      "ActionType": "BA",
      "ActionTypeName": "Book an appointment",
      "StateType": "FT",
      "StateName": "Future task",
      "CanProcess": false,
      "Key": "0ddf1c6b831d08fe5ccdb65fd76a4665",
    },
    {
      "Name": "AQ",
      "ActionType": "QR",
      "ActionTypeName": "Answer a questionnaire",
      "StateType": "FT",
      "StateName": "Future task",
      "CanProcess": false,
      "Key": "81aadeae6b9535135ed1c8dfc83a81b7",
    },
    {
      "Name": "AA",
      "ActionType": "AA",
      "ActionTypeName": "Arrive an appointment",
      "StateType": "FT",
      "StateName": "Future task",
      "CanProcess": false,
      "Key": "86149d568252ae476859a3a60835369b",
    }
  ]
}
```

38.52 PathwayOverviewData

Provides overview information about the pathway.

Properties

Key	string	The key of the pathway.
Number	string	The number of the pathway.
Name	string	The name of the pathway.
Patient	string	The name of the patient.
State	string	The current pathway state.
CurrentAction	string	The action that is currently active.
LastUpdate	DateTime	Last modification date.
DueDate	DateTime	The current active action due date.

JSON Example

```
{
  "Key": "a1d3e2d1",
  "Number": "123",
  "Name": "Book & Arrive",
  "Patient": "Mr. Will Smith",
  "State": "In progress",
  "CurrentAction": "Book an appointment",
  "LastUpdate": "2015-03-13T14:22:12",
  "DueDate": "2015-03-13T14:52:30",
}
```

38.53 PathwayTaskData

A full task description within a pathway.

Properties

Key	string	The key of the task.
Name	string	The name of the task.
ActionType	string	The type of the task. See Pathway structure and task types.
ActionTypeName	string	The user-friendly name of the action type.
Description	string	The description of the task.
StateType	string	The state of the task. - FT – Future pending task, not-started yet - ST – Started task - IP – Task in progress - CP – Task completed - SK – Task has been skipped - FL – Task failed Note that even if the task is in ST (Started) state the client need to check *CanProcess* field to allow any action.
StateName	string	The user-friendly name of the state.
CanProcess	bool	True if the task and the logged-in user has rights to process the task.
AppointmentTask	AppointmentTaskData	The details about the book/arrive appointment task.
AttachDocumentTask	AttachDocumentTaskData	The details about the attach document task.
ChoiceTask	ChoiceTaskData	The details about the choice task.
QuestionnaireTask	QuestionnaireTaskData	The details about the questionnaire task.

JSON Example

```
{
  "Name": "AA",
  "ActionType": "AA",
  "ActionTypeName": "Arrive an appointment",
  "Description": "Please arrive to your appointment.",
  "StateType": "FT",
  "StateName": "Future task",
  "CanProcess": false,
  "AppointmentTask": {
    "AppointmentType": "0001a9748715e6a2",
    "AppointmentTypeName": "Consultation",
    "IsBooked": false,
  },
  "Key": "86149d568252ae476859a3a60835369b",
}
```

38.54 PathwayTaskOverviewData

An overview of a single task within a pathway.

Properties

Key	string	The key of the task.
Name	string	The name of the task.
ActionType	string	The type of the task. See Pathway structure and task types.
ActionTypeName	string	The user-friendly name of the action type.
StateType	string	The state of the task. - FT – Future pending task, not-started yet - ST – Started task - IP – Task in progress - CP – Task completed - SK – Task has been skipped - FL – Task failed Note that even if the task is in ST (Started) state the client need to check *CanProcess* field to allow any action.
StateName	string	The user-friendly name of the state.
CanProcess	bool	True if the task and the logged-in user has rights to process the task.
Tasks	PathwayTaskOverviewData[]	The list of nested tasks.

JSON Example

```
{
  "Name": "AA",
  "ActionType": "AA",
  "ActionTypeName": "Arrive an appointment",
  "StateType": "FT",
  "StateName": "Future task",
  "CanProcess": false
  "Key": "86149d568252ae476859a3a60835369b",
}
```

38.55 PathwayDefinitionData

An overview of a pathway definition

Properties

Key	string	The key of the definition.
Name	string	The name of the definition.
Prompt	string	Prompt to show to patient
Description	string	Description for the pathway.
Path	string	Tree path for the pathways.

JSON Example

```
{
      Description: ""
      Key: 1
      Name: "Patient Document Upload"
      Path: "Patient Document Upload"
      Prompt: "Patient Document Upload"
}
```

38.56 PathwayAvailabilityData

Availability data for a pathway definition

Properties

IsAvailable	bool	Availability Boolean for the pathway definition
Error	string	Reason for not being available.

JSON Example

```
{
      Error: "Maximum instances of pathways already running."
      IsAvailabe: false
}
```

38.57 PatientAbsenceOverviewData

A small overview object containing the patient’s Bradford Factor score and a flag indicating whether they’re currently absent or not.

Properties

Key	string	The key of the patient.
IsCurrentlyAbsent	bool	A flag indicating whether the patient is currently absent or not.
BradfordFactor	int	The Bradford factor score for the patient for the time-period specified, or if not time-period is specified, the last 52-week period.

JSON Example

```
{
    "BradfordFactor": 24,
    "IsCurrentlyAbsent": true,
    "Key": "83075be08d41fd6096c3c53bc8c4bda7"
}
```

38.58 PatientContactOption

Contains the information about a permission the patient has given to the company to contact them on a specific subject by a specific way.

Properties

Key	string	The key of the contact option. This key is used to identify the contact option within UpdateDemographicData.
Subject	string	The name of subject.
Email	bool	Whether the patient has given a permission to contact them by email.
SMS	bool	Whether the patient has given a permission to contact them by a text message.

JSON Example

```
{
  "Key": "Email.General",
  "Subject": "General human contact",
  "Email": true,
  "SMS": "true"
}
```

38.59 PatientRegistrationResultData

Contains the details information about successful registration.

Properties

ActivationType	int	How can the patient activate her account: - 1 = email activation - 2 = SMS activation - 3 = email or SMS activation The server choose activation type according actual configuration. The client uses this information to provide right information about activation to the patient.

JSON Example

```
{
  "ActivationType": 2
}
```

38.60 PatientReportReviewData

Provides information about the patient medical report review for the patient.

Properties

ReportUrl	string	The URL address of the full medical report. The report is PDF document. We recommend to use <object> tag to display embedded PDF: <object type="application/pdf" data="<url>"></object>
ReferredBy	string	The name of the referrer.
AutoReleaseTime	DateTime	Defines the time when the automatic release is scheduled.
PossibleFactualChanges	string[]	List of possible factual changes the patient can request.
Submitted	bool	Defines whether the review has already been submitted. It is not allowed to submit the review more than once.
Comments	string	The comments provided by the patient for the referrer. Provided only if the Submitted==true
FactualChanges	object[] see example	The list of factual changes requested by the patient. Provided only if the Submitted==true

JSON Example

```
{
  "ReportUrl": "http://api.meddbase.com/patientportalapi/....",
  "ReferredBy": "Mr. John Smith",
  "AutoReleaseTime": "2015-05-14T06:00:00",
  "PossibleFactualChanges": [
    "DOB incorrect",
    "Name incorrect",
    "other"
  ],
  "Submitted": false,
  "Comments": "<Patient’s previous comments if submitted>",
  "FactualChanges": [
    {
       "Subject": "DOB incorrect",
       "Comments": "My DOB is 1.1.1985"
    },
    {...}
  ]
}
```

38.61 PatientReportReviewState

Provides information about the state of the patient medical report review.

Properties

Required	bool	True if the patient requires the medical report review.
InProgress	bool	True if the review is in progress (i.e. an invitation email has been sent to the patient).
Finished	bool	True if the review has been finished.
State	string	The current state of the review.
StartDate	DateTime (optional)	The review start date and time.
PatientFirstAttempt	DateTime (optional)	The time of the patient’s first attempt to read the report.

JSON Examples

```
{
  "Required": true,
  "InProgress": false,
  "Finished": false,
  "State": "Not started"
}
```

Or

```
{
  "Required": true,
  "InProgress": false,
  "Finished": true, ,
  "State": "Refused",
  "StartDate": "2016-11-23T08:00:01.483",
  "PatientFirstAttempt": "2016-11-23T14:22:12.483"
}
```

38.62 PayerAccountData

Returns list of payer accounts:

Properties

Id	int	Identifier for the account
Last4Digits	string	Last4digits of the card
ExpiryDate	DateTime	ExpiryDate of the card
LastUsed	DateTime	Last used date time
PaymentType	String	Payment Type: Visa, Master Card, etc.
PayType	String	PayType

38.63 PayerAccountInputData

Arguments for a new payer acccount

Properties

firstname	string	Billing firstname on the payer card: 20 characters
surname	string	Billing surname on the payer card: 20 characters
address1	string	Billing address line 1: 100 characters
address2	string	Billing address line 2: 100 characters (Optional)
city	string	Billing city: 40 characters
postcode	string	Billing postcode: 10 characters
phone	string	Billing phone: 20 characters (optional)
email	string	Customer email: 255 characters (optional)

38.64 PayerType

Type of the payer.

Properties

Key	string	
Name	string	

JSON Example

```
{
  "Key": "PA",
  "Name": "Patient"
}
```

38.65 PersonDemographicData

Contains the person’s demographic information (e.g. patient’s demographic data).

Properties

Key	string	Key of the person.
Title	string	
Name	string	
Surname	string	
SexType	int	Biological sex at birth: - 0 = any - 1 = male - 2 = female
Gender	GenderData	The person’s gender identity.
Pronoun	PronounData	The person’s preferred pronouns.
Initials	string	
DateOfBirth	DateTime	
Mobile	string	
Telephone	string	
EmailAddress	string	Personal email address. This address is used to log into the patient portal.
WorkTelephone	string	
WorkEmailAddress	string	Work email address. This address is used to log into the referral portal.
Password	string	Used only for registration. Should be at least 8 characters long and should contain at least 1 numbers, 1 capital letter and 1 small letter. API never returns the password. It is for submitting only.
Address	AddressData	
NextOfKin	NextOfKinDemographicData	
ContactOptions	PatientContactOption[]	
TermsAndConditionsAccepted	bool	
StatisticalProcessingAccepted	bool	
EmployeeNumber	string	The employee number.
EmployeeDepartment	DepartmentData	The department the employee belongs to. Could be null.
EmployeeDivision	DivisionData	The division the employee belongs to. Could be null.
EmployeeStatus	EmployeeStatus	The status of the employee.
Permissions	object See GetConfig for details.	The permissions of the logged-in user for this person. This could be different to the permissions provided upon GetConfig.

Insurer	InsurerData	Insurer Data for the patient. Only available for patient portal. When setting patient demographics, only public company key's can be used to set the Insurer.
EmployerName	String	The name of the employer.
EmployerAddress	AddressData	The employer address.
ReferrerKey	string	Encrypted key of the referrer. The set of valid values are obtained by call to getReferrers. Can be null.
EthnicityKey	string	Encrypted key of the person's ethnicity. The set of valid values are obtained by call to getEthnicities. Can be null.

JSON Example

```
{
  "Title": "Mr",
  "Name": "John",
  "Surname": "Lemon",
  "SexType": 1,
  "Initials": "JL",
  "DateOfBirth": "1958-08-02T00:00:00",
  "Mobile": "+444 895 523 411",
  "Telephone": "+444 525 111 555",
  "EmailAddress": "john.lemon@test.com",
  "WorkEmailAddress": "john.lemon@mywork.com",
  "Password": "jon4535lemon",
  "Address": {
    "Address1": "Studio 99",
    "Address2": "Backlok Street",
    "Address3": "Camden",
    "City": "London",
    "County": "",
    "PostCode": "N1 7NK",
    "Country": "United Kingdom"
  },
  "NextOfKin":  {
    "Relationship": "Mam",
    "Name": "Mariel",
    "Surname": "Lemon",
    "Mobile": "+444 895 111 222",
    "WorkTelephone": "+444 525 111 555",
    "Address": {
      "Address1": "Studio 1",
      "Address2": "Cardwell Roa",
      "Address3": "Camden",
      "City": "London",
      "County": "",
      "PostCode": "N1 7NK",
      "Country": "United Kingdom"
    }
  }
}
```

38.66 PrescriptionData

Contains information about the prescription.

Properties

Key	string	Key of the prescription.
Type	string	Type of prescription. Possible values are: - 'R' (repeat) - 'A' (acute)
DrugKey	string	Key of the drug to prescribe.
DrugName	string	Name of the drug to prescribe.
Quantity	decimal	Quantity in a pack.
QuantityUnit	string	Unit of the quantity (e.g. tablets, capsules, etc.).
Dosage	string	How often is the medicine taken (e.g. One to be taken five times a day).
Duration	int	Count of days the medicine is taking.
Method	string	How the medicine is taken (e.g. swallow without chewing, apply liberally, etc.).
Route	string	Route of administration (e.g. Oral Route, Intraocular route, etc.).
PharmacyText	string	Any comments for Pharmacist.
LastIssueDate	DateTime	Last date when this drug was issued.

JSON Example

```
{
  "Key": "d2546",
  "Type": "R",
  "DrugKey": "d2546",
  "DrugName": "Aspirin Tablets 75 mg",
  "Quantity": 28,
  "QuantityUnit": "tablets",
  "Dosage": "One To Be Taken Each Day",
  "Duration": 28,
  "Method": "",
  "Route": "Oral route",
  "PharmacyText": ""
}
```

38.67 PronounData

Represents a person’s pronoun preferences.

Properties

Key	string	The encrypted key of the pronoun.
Name	string	The name of the pronoun.

JSON Example

```
{
  "Key": "1c5aabbdf813ad78c48cc8e6d8584162",
  "Name": "She/Her/Hers"
}
```

38.68 QuestionnaireAnswerData

Answer for the questionnaire question.

Properties

QuestionId	long	Id of the question from the questionnaire markup.
Answer	string	Answer of the question.
Answers	string[]	Answers of the question. Used if the answer includes more answers.

JSON Example

```
{
  "QuestionId": 236,
  "Answer": "Yes",
  "Answers": ["Yes"]
}
```

38.69 QuestionnaireData

Contains the information about the questionnaire including questions and answers.

Properties

Key	string	Key of the questionnaire.
Name	string	Name of the questionnaire.
Description	string	Description of the questionnaire.
StatusCode	int	Status code of the questionnaire: - 0 = Incomplete - 1 = Complete - 2 = Partially Complete
Status	string	String representation of the status code.
AppointmentKey	string	The appointment the questionnaire is created for.
Expiration	DateTime	When the questionnaire expires.
Answers	QuestionnaireAnswerData[]	All saved answers for this questionnaire.
Markup	string	XML contains questions. See Questionnaire markup.

JSON Example

```
{
  "Key": "q13788",
  "Name": "Basic questionnaire",
  "Description": "",
  "StatusCode": 0,
  "Status": "Incomplete",
  "AppointmentKey": "apt5821",
  "Expiration": "2013-08-25",
  "Answers": [
    {
      "QuestionId": 236,
      "Answer": "Yes"
    },
    {
      "QuestionId": 237,
      "Answer": "No"
    }
  ],
  Markup = "<?xml version=\"1.0\" encoding=\"utf-8\"?><Questionnaire> see
Questionnaire markup to see full markup"
}
```

38.70 QuestionnaireFormData

Provides information about a questionnaire form.

Properties

Key	string	The key of the questionnaire form.
Category	string	Always ‘ppq’.
Name	string	Name.
Description	string	Longer description.

JSON Example

```
{
    "Key": "3bdd966cf6f9c0c6872ee0551da74c4d",
    "Category": "ppq",
    "Name": "Test",
    "Description": “Longer description”,
}
```

38.71 QuestionnaireModuleData

Provides information about a questionnaire module.

Properties

Key	String	The key of the questionnaire module.
Category	string	Always ‘ppq’.
Name	string	Name.
FormKeys	string []	The keys of questionnaire forms included in the module.

JSON Example

```
{
    "Key": "3bdd966cf6f9c0c6872ee0551da74c4d",
    "Category": "ppq",
    "Name": "Test",
    "FormKeys": [“4bdd966cf6f9c0c6872ee0551da74c4e”],
}
```

38.72 QuestionnaireOverviewData

The questionnaire overview contains basic information about the questionnaire.

Overview data doesn’t contains questions and answers.

Properties

Key	string	Key of the questionnaire.
Name	string	Name of the questionnaire.
Description	string	Description of the questionnaire.
StatusCode	int	Status code of the questionnaire: - 0 = Incomplete - 1 = Complete - 2 = Partially Complete
Status	string	String representation of the status code.
AppointmentKey	string	The appointment the questionnaire is for.
Expiration	DateTime	When the questionnaire expires.
Sent	DateTime (optional)	When the questionnaire was sent.
ReferralKey	string (optional)	If present, the key of a referral that was created from the questionnaire request.
CanRefer	bool (optional)	Whether the questionnaire indicates a referral.

JSON Example

```
{
  "Key": "q13788",
  "Name": "Basic questionnaire",
  "Description": "",
  "StatusCode": 0,
  "Status": "Incomplete",
  "AppointmentKey": "apt5821",
  "Expiration": "2013-08-25",
  "Sent": "2013-07-01",
  "ReferralKey": "8ebb55f10382d73e494e7207d1bd7272",
  "CanRefer": 0,
}
```

38.73 QuestionnaireResultData

Provides the results of a questionnaire request.

Properties

Complete	bool	True if the patient has finally submitted the results.
Name	string	Name of the questionnaire.
Html	string	The Html markup of the results document which contains the formatted results.

JSON Example

```
{
    "Complete": true,
    "Name": "Test",
    "Html": "…",
}
```

38.74 QuestionnaireTaskData

An additional details for the questionnaire task.

Properties

Key	string	The key of the questionnaire. Use this key to get the questionnaire details within GetQuestionnaireDetail.

38.75 RecallData

Provides information about the recall.

Properties

Key	string	The key of the recall.
Patient	PersonDemographicData	The patient data.
Reason	string	The reason for the recall.
AppointmentType	AppointmentTypeData	The appointment type.
DueDate	DateTime	Recall due date.
Clinician	ClinicianData	The clinician who triggered the recall.

JSON Example

```
{
    "Key": "3bdd966cf6f9c0c6872ee0551da74c4d",
    "DueDate": "2017-03-14T00:00:00",
    "Reason": "Test",
    "AppointmentType": {...},
    "Patient": {...},
    "Clinician": {...}
}
```

38.76 ReferralData

Provides full information about the referral. Includes all properties of the ReferralOverviewData.

Properties

<ReferralOverviewData properties>		The object inherits the ReferralOverviewData
Appointments	AppointmentData[]	List of appointments related to the referral.
Questionnaires	QuestionnaireOverviewData[]	List of questionnaires related to the referral.
Documents	DocumentData[]	List of attached documents.
CanBookAppointment	bool	Defines whether the user has got right to book an appointment directly.
CanCancel	bool	Defines whether the user has got right to cancel the referral.
CanSend	bool	Defines whether the user has got right to send the referral.
CanReallocate	bool	Defines whether the user has got right to reallocate the referral.
CanCollaborate	bool	Defines whether the user has got right to collaborate.
CanAttachDocuments	bool	Defines whether the user has got right to attach documents.
CanFollowUp	bool	Defines whether the user has got right to request the follow up.
PatientReview	PatientReportReviewState (optional)	The state of the patient medical report review. Not provided if the referral isn't in the appropriate state or if the patient doesn't' require the review.
AbsenceRecordKey	string	The key of the absence record the referral is joined to.
RejectionReason	string	The reason that the referral was rejected.
PublicReasonForReferral	string	The public reason for the referral (public = shared with the employer).
DaysToReviewDischargeLetterByPatient	int	The number of days a patient has to review the referral discharge letter

JSON Example

```
{
  "Key": "R123",
  "PatientName": "Mr. John Smith",
  "ReferredBy": "Mr. Will Smith",
  "CreatedDate": "2015-03-13T14:22:12",
  "ModifiedDate": "2015-03-13T14:52:30",
  "State": "InProgress",
  "StateDisplayName": "In progress",
  "StateColor": "green",
  "ReferralNumber": 13911,
  "AppointmentType": {
    "Name": "Referral",
    "Key": "REF1",
    "Notes": "",
    "CanBookAppointment": true,
    "CanReferPatient": true
  }
  "Appointments": [
     {<see AppointmentData>}
    ],
  "Questionnaires": [
    {
      "Key": "q123",
      "Name": "Referral",
      "Description": "",
      "StatusCode": 1,
      "Status": "1",
      "Expiration": "2015-03-26T14:37:00.2003719+00:00"
    }
  ],
  "Documents": [
    {
      "Name": "Referral letter.html",
      "Author": "Mr. Will Smith",
      "Comment": "I would like to refer Mr. John Smith",
      "DateCreated": "2015-03-13T14:22:12.483",
      "Url": "https://api.meddbase.com/referrals/download?d=123",
      "MIMEType": "text/html",
      "Size": 20824
    }
  ],
  "CanCancel": false,
  "CanBookAppointment": false,
  "CanSend": false,
  "CanReallocate": true,
  "CanCollaborate": true,
  "CanAttachDocument": false,
  "CanFollowUp": false,
  “PatientReview”: {
    <see PatientReportReviewState>
  },
  “AbsenceRecordKey”: “123”,
  “RejectionReason”: “Holidays”
  “PublicReasonForReferral”: “Angina”,
  "DaysToReviewDischargeLetterByPatient": 3
}
```

38.77 ReferralOverviewData

Provides overview information about the referral.

Properties

Key	string	The key of the referral.
PatientName	string	The name of the patient.
PatientKey	string	The key of the patient.
ReferredBy	string	The name of the referrer.
CreatedDate	string	Created date.
ModifiedDate	string	Last modification date.
State	string	The state of the referral: - *QuestionnaireRequired* – the referral questionnaire hasn't been finished. - *AttachDocuments* – the referral hasn't been sent. The user may attach documents and send the referral. - *InProgress* – the referral in progress. No user action is needed. - *Completed* – the referral is completed. The list of states is not complete. There may be also another states.
StateDisplayName	string	The friendly name of the state.
StateColor	string	The color of the state.
IsFollowUp	bool	The referral is a follow up for the previous referral.
HasFollowUp	bool	The referral has a follow up referral.
PreviousReferralKey	bool	The key of the previous referral if IsFollowUp is true.
NextReferralKey	bool	The key of the next referral if HasFollowUp is true.
SLARequired	bool	True if the SLA has been required for the referral.
SLAFailed	bool	True if the SLA has failed for this referral.
SLAFailedReason	string	The reason why SLA failed.
ReferralNumber	string	The number of the referral.
AppointmentType	AppointmentTypeData	The type of the referral.
AbsenceRecordKey	string	The key of the absence record the referral is joined to.
DaysToReviewDischargeLetterByPatient	int	The number of days a patient has to review the referral discharge letter.

JSON Example

```
{
  "Key": "R123",
  "PatientName": "Mr. John Smith",
  "ReferredBy": "Mr. Will Smith",
  "CreatedDate": "2015-03-13T14:22:12",
  "ModifiedDate": "2015-03-13T14:52:30",
  "State": "InProgress",
  "StateDisplayName": "In progress",
  "StateColor": "green",
  "SLARequired": true,
  "SLAFailed": true,
  "SLAFailedReason": "The appointment was not placed within the SLA (5 days):
Patient cannot meet SLA.",
  "ReferralNumber": 13911,
  "AppointmentType": {
    "Name": "Referral",
    "Key": "REF1",
    "Notes": "",
    "CanBookAppointment": true,
    "CanReferPatient": true
  },
 "DaysToReviewDischargeLetterByPatient": 3
}
```

38.78 ReferrerData

Data returned in response to the `getReferrer` method

Properties

Key	string	Encrypted key of the referrer
Name	String	The name of the referrer

38.79 ServerStateData

Provides information about an actual server state.

Properties

Name	string	Name of the server.
StateCode	int	Current server state: - 1 = running - 2 = updating - 3 = suspending
State	string	String representation of the current server state.
ServiceVersion	string	Current version of the server in format: “<major>.<minor>.<build>”. For example: “1.60.2995”.
ProtocolVersion	string	Service communication protocol version in format: “<major>.<minor>.<build>”. For example: “1.0.1” The build number can change without the interface changing (adding optional parameters, adding new methods). Change in the major or the minor version means the interface change and the client who doesn’t support new version of interface shouldn’t continue working and needs to be upgraded.

JSON Example

```
{
  "Name": "Meddbase server",
  "StateCode": 1,
  "State": "Running",
  "ServiceVersion": "1.60.2995",
  "ProtocolVersion": "1.0.1"
}
```

38.80 ServiceData

Information about the Service.

Properties

Key	int	Unique key of the service.
Code	string	Code of the service
Name	string	Name of the service
CurrencyCode	string	Code of the currency
CurrencySymbol	string	Symbol for the currency
NetPrice	decimal	Net price of the service
Tax	decimal	Tax on the service
GrossPrice	decimal	Gross price
ServiceType	ServiceTypeData	Type of the service

JSON Example

```
{
  "Code": "A1250",
  "CurrencyCode":"GBP",
  "CurrencySymbol": "£",
  "GrossPrice": 0,
  "Key":593,
  "Name": "Creation of subcutaneous cerebrospinal fluid reservoir",
  "NetPrice": 0,
  "Tax": 0,
  "ServiceType" : {
                    "Key": 84,
                     "Name": "Blood profile"
                    }
}
```

38.81 ServiceExceptionData

Provides information about the exception.

Properties

Message	string	Message of the exception.
EventType	int	Type of the exception: - 1 = Critical - 2 = Error - 4 = Warning - 8 = Information
EventCode	int	Specific code to identify the exception (argument missing, not logged in, not all required questions of the questionnaire are answered, etc.).
HttpStatusCode	int	Same like HTTP status code. See Error handling
Description	string	More details of the exception.
ReferenceNumber	string	A unique reference number that represents an event in the Meddbase logs. This can be traced by Meddbase support.

JSON Example

```
{
  "Message": "Password isn't set.",
  "EventType": 2,
  "EventCode": 0,
  "HttpStatusCode": 400,
  "Description": null,
  "ReferenceNumber": null
}
```

38.82 ServiceTypeData

Information about the Service Type.

Properties

Key	string	Unique key of the service type
Name	string	Name of the service type

JSON Example

```
{
  "Key": 84,
  "Name": "Blood profile"
}
```

38.83 ShallowSearchResultData

A shallow search result of a patient/company address.

Properties

Key	string	An encrypted key value containing partial address information used to retrieve complete address details.
Address	string	A single-line search result address.

JSON Example

```
{
  "Key":
"fc560b40315dc7605fd5ca53e0dcaabc357c69bea3faefa8c6e2ce8129909061956910b77338ee2c2
cdbb1c7c5f7c64bcf338d78bc148f81f6786152d3ef2987b3ab5b1e5588b1db7939bb5e0edffec4614
c4511c4a7a0dfd9bc9077749482b152217c572b0f78552c75be542ffcea6446110af6da78213c1f715
69f35abab7d65f82f382f8b8dc663c8e6a1405bf17c331d379f375ffbc6ec3ebc21a985a69355d1062
2db48eceb7f23b38c5037ed2315c3d858268baae1879f6f84b3b65586742086832ec398acdfd56680a
72991d7bb38bbfd1fa61991aebf0bd1982dc06b",
  "Address": " 2 Paradise Street, Liverpool, Merseyside, L1 8JF"
}
```

38.84

38.85 SiteData

Geographic place/residence of the doctor/clinic.

Properties

Key	int	Key of site.
Name	string	Name of the site.
Address	AddressData	Address of the site.
Locations	LocationData[]	Possible locations within this site. Note: Sometimes the location may be placed on geographically different place than the site. To show the location the client should use the location's address.

JSON Example

```
{
  "Key": 1123,
  "Name": "2CP (Eye Room)",
  "Address":{
    "Address1": "2 Clifton Park Ave",
    "Address2": "",
    "Address3": "",
    "City": "London",
    "County": "",
    "PostCode": "SW20 8BD",
    "Country": "United Kingdom"
  },
  "Locations": [
    {
      "Key": 45,
      "Name": "Surgery"
    },
    {
      "Key": 214,
      "Name": "Room 1"
    }
  ]
}
```

38.86 SSOStatus

Provides information about the single sign on

Properties

Identifier	string	Unique api identifier for the employer for single sign on
Enabled	bool	Is enabled
IsOh	bool	Is for referral / managers

JSON Example

```
{
   "Identifier": "shiny",
   "Enabled": true,
   "IsOH": true
}
```

38.87 TermsAndConditions

Provides information about the terms and conditions.

Properties

TermsAndConditionsText	string	The text of the terms and conditions.
StatisticalProcessingText	string	The text of the announcement that the patient data will be used for statistical processing.

JSON Example

```
{
  "TermsAndConditionsText": "<p>Confidentiality of patient information is of our
upmost concern. ...etc...</p>",
  "StatisticalProcessingText": "<p>The anonymised anonymous aggregated medical
data may be used for quality control and research or statistical purposes. The
personal data will not be transferred to any third party for any purpose unless
the patient has consented to the release of such information.</p>"
}
```

38.88 TimeSlotsData

Properties

TimeSlots	TimeSlotData[]	The list of proposed time slots.
Type	AppointmentTypeData	Appointment type of the proposed time slots.
Sites	SiteData[]	The deduplicated list of sites for the found time slots, you can use the SiteKey in a TimeSlotData to find the relevant SiteData, and use the LocationKey to find the relevant LocationData inside the Locations inside a SiteData. Note that the Locations within each Site are only the locations that are relevant to found TimeSlots, not all Locations for the site.
Services	ServiceData[]	Services related to the appointment.
Modules	AppointmentModuleData[]	Modules related to the appointment.
Currency	CurrencyData	The currency data that represents the currency for all prices in the returned time slots.

JSON Example

```
{
  "TimeSlots": [
    {
      "Start": "2025-01-01T09:30:00",
      "Finish": "2025-01-01T10:30:00",
      "SiteKey": 1123,
      < See TimeSlotData to see all properties. >
    },
    {
      "Start": "2025-01-01T10:30:00",
      "Finish": "2025-01-01T11:30:00",
      "SiteKey": 1123,
      "LocationKey": 214,
      < See TimeSlotData to see all properties. >
    }
  ],
  "Type": {
    "Key": "CN",
    "Name": "Consultation"
  },
  "Sites": [
    {
      "Key": 1123,
      "Name": "2CP (Eye Room)",
      "Address": {
        "Address1": "2 Clifton Park Ave",
        "Address2": "",
        "Address3": "",
        "City": "London",
        "County": "",
        "PostCode": "SW20 8BD",
        "Country": "United Kingdom"
      },
      "Locations": [
        {
          "Key": 214,
          "Name": "Room 1"
          "Address": {
            "Address1": "2 Clifton Park Ave",
            "Address2": "",
            "Address3": "",
            "City": "London",
            "County": "",
            "PostCode": "SW20 8BD",
            "Country": "United Kingdom"
          }
        }
      ]
    }
  ],
  "Services": [
    {
        < See ServiceData to see all properties. >
    },
    {
        < See ServiceData to see all properties. >
    }
  ],
  "Modules": [
    {
        < See AppointmentModuleData to see all properties. >
    },
    {
        < See AppointmentModuleData to see all properties. >
    }
  ],
  "Currency": {
    "Code": "GBP",
    "Symbol": "£"
  }
}
```

38.89 TimeSlotData

Properties

Start	DateTime	Start time.
Finish	DateTime	Finish time.
SiteKey	int	Key of the site the time slot is placed.
LocationKey	int	Key of the location the time slot is placed.
GrossPrice	decimal	The potential gross price of the appointment if booked using this time slot, might differ depending on the exact slot booked into.
NetPrice	decimal	The potential net price of the appointment if booked using this time slot, might differ depending on the exact slot booked into.

JSON Example

```
{
  "Start": "2025-01-01T09:30:00",
  "Finish": "2025-01-01T10:30:00",
  "SiteKey": 123,
  "LocationKey": 546,
  "GrossPrice": 240,
  "NetPrice": 200
}
```

38.90 UserData

Provides information about the user. The UserData object inherits the PersonDemographicData.

Properties

Key	string	The key of the user. This key doesn’t equal the key of the patient.
<PersonDemographicData properties>		The UserData object inherits the PersonDemographicData.
Rights	User[]	The list of user rights.
AllPatientsVisible	bool	If *True* then all patients within the company are visible and the list of accessible departments is ignored.
AllNonAssignedPatientsVisible	bool	If True then all patients who do not have a department set up within the company are visible.
AccessibleDepartments	DepartmentData[]	The list of departments the user has got an access to manage.
AllDocumentTypesVisible		If *True* then all document typse within the company are visible. This applies if the *DocumentTypeSecurityEnable* flag is set (see GetConfig)
AccessibleDocumentTypes	DocumentType[]	The list of document types the user has got an access to see. This applies if the *DocumentTypeSecurityEnable* flag is set (see GetConfig)
LastLogin	DateTime	The last login date. Date could be null.

JSON Example

```
{
  "<PersonDemographicData properties >": <values>,
  "AllPatientsVisible": false,
  "AllNonAssignedPatientsVisible": false,
  "Rights": [
   {
      "Key": "0f0f8997-a161-455e-a498-96138096f539",
      "Name": "Case management",
      "Description": "The user can refer patients and manage the referrals."
   }
  ],
  "AccessibleDepartments": [
    {
      "Key": "D01",
      "Name": "HR"
    },
    {
      "Key": "D02",
      "Name": "Research"
    }
  ],
  "LastLogin": "2015-01-01T10:41:10.547"
}
```

38.91 UserRights

Provides information about the user rights.

Properties

Key	string	
Name	string	
Description	string	

JSON Example

```
{
  "Key": "0f0f8997-a161-455e-a498-96138096f539",
  "Name": "Case management"
  "Description": "The user can refer patients and manage the referrals."
}
```

39 Error handling

When an exception occurs the HTTP status code is set to an appropriate code from the table below and the response from the server contains information about an occurred exception.

39.1 HTTP status codes

In general, the Meddbase API can generate these codes:

200	OK	
201	Created (Saved)	
204	No Content.	Request was successful and does not need to return a response.
400	Bad request	Bad parameter format, missing parameter etc.
401	Unauthorised	You are not authorised (If a session ID was provided then an Unauthorised exception indicates session expiry). See Login to find out more about authorization.
403	Forbidden	You don’t have rights to perform current operation.
404	Not found	
405	Method not allowed	For example when you use GET instead of POST.
424	Method Failure	
500	Internal Server Error	
501	Not implemented	

The table above doesn’t contain all HTTP status codes that can be generated by the Meddbase server and not all responses from the server contains the ServiceExceptionData object.

39.2 EventCode

The client can use the *EventCode* property of the ServiceExceptionData object to identify the concrete type of the exception. Next table shows generally used codes but you can find another codes that can be thrown (these codes are described usually within *Remakrs* section of the API methods).

0	Not specified exception
1	Parameter is missing
2	Parameter has incorrect format (usually when integer contains letters or when date-time format is not valid)
3	Client authentication key is missing.
4	Client authentication key not valid.
5	Unknown API section (the sections means for example: finance, patient, auth, etc.)
6	Unknown API method.
7	Invalid or missing POST data.
8	User is not logged in.
9	Company name is missing.
10	Application is starting. Please wait and send the request later.
11	Invalid token. See Authentication Token.
12	Input includes HTML markup.
13	The client IP address is not in the allowed range specified in the employer settings.
14	User has been blocked because of too many failed login attempts.
10001	This account is already registered on our system as a private patient.
40001	Not all required questions of the questionnaire is answered.
50001	Medical history node you are looking for is not found.

40 Questionnaire markup

The questionnaire markup contains the definition of a whole questionnaire: pages, instructions and questions.

Full questionnaire markup example

```xml
<?xml version="1.0" encoding="utf-8"?>
<Questionnaire>
  <Pages>
    <Page title="Assure - Health Screening Questionnaire">
      <Information>
        Please answer the following questions as far as you can. If in doubt please
        ask the doctor later.
      </Information>
      <Field id="314">Job Title</Field>
      <Field required="false" id="315">Department</Field>
      <Information>
        Is there a family history of any of the following (Grandparents, parents,
        brothers, sisters)?
      </Information>
      <Question id="328">
        <Text>Heart Conditions / Angina</Text>
        <Answers>
          <Answer>Yes</Answer>
          <Answer>No</Answer>
        </Answers>
      </Question>
    </Page>
    <Page title="Exercise">
      <Field id="389">What exercise do you do and how often?</Field>
      <Field required="false" id="391">What exercise do you enjoy?</Field>
      <Field required="false" id="392">What exercise do you dislike?</Field>
      <Question required="false" id="394">
        <Text>
          How active is your job?
        </Text>
        <Answers>
          <Answer>Light</Answer>
          <Answer>Moderate</Answer>
          <Answer>Vigorous</Answer>
        </Answers>
      </Question>
      <Table id="10963" required="false">
        <Text>Occupational History</Text>
        <Column header="Employer" />
        <Column header="Job title" />
        <Column header="From" datatype="date" />
        <Column header="Until" datatype="date" />
      </Table>
    </Page>
  </Pages>
</Questionnaire>
```

Markup description

Document root `Questionnaire` contains a list of `Pages`.

- The client should can show progress by counting the number of Page tags and using that to show a percentage progress-indicator.

Every `Page` can contain tags:

- `Information` – Plain text that guides the patient and describes/explains the following questions.
- `Field` – Free-text field. The value of this tag contains the field label.
    - The client must use the `id` attribute to submit answers (see *SaveQuestionnaire*).
    - If the `required` attribute isn't present, then the answer is required. Only if `required="false"` the answer is not required.
- `Question` – question with more answers.
    - The `Text` element contains the question-text.
    - The `Answers` element contains possible answers.
    - The initial state of the UI must be un-set (no answer is selected).
    - Only one answer can be selected.
    - The client must use the `id` attribute to submit answer (see *SaveQuestionnaire*).
    - If the `required` attribute isn't present, then the answer is required. Only if `required="false"` the answer is not required.