UAB MedDream
K. Petrausko st. 26,
LT-44156, Kaunas, Lithuania
info@meddream.com
www.meddream.com

MedDream

integration MANUAL

(version 8.9.1)

© 2026, MedDream UAB, Kaunas

All rights reserved in the event of granting of patents or registration as a utility patent. All names of companies and products mentioned in this user's manual may be trademarks or registered trademarks. References to products of other manufacturers are for information purposes only. Such references are intended neither as an approval nor a recommendation of these products. MedDream UAB accepts no liability for the performance or use of such products. Other brand names, software and hardware names used in this user's manual are subject to trademark or patent protection. The quoting of products is for informational purposes only and does not represent a trademark misuse. This Servicing Manual is protected by copyright. Unless expressly authorized in writing, dissemination, duplication or other commercial exploitation of this documentation set or communication of its contents or parts of it is not permitted. In case of infringement, the violator may be liable to pay compensation for damages. Specifications due to technical developments are subject to change. This Servicing Manual is not subject to the revision service. Please contact the manufacturer or authorized dealer to request the latest edition of Servicing Manual.

Table of Contents

Table of Contents 3
Document purpose 5
Explanation of symbols used 5
Introduction 6
Minimum system requirements 7
Scenarios for MedDream URL integration 8
Token enabled MedDream URL integration scenarios using MedDream TokenService 8
One step token enabled MedDream URL integration 9
Two steps token enabled MedDream URL integration 11
Token enabled MedDream URL integration scenario using 3rd party token generation & validation service 14
Integration scenario using MedDream token 14
Integration scenario using JSON Web Token (JWT) 16
Token disabled MedDream URL integration scenario 20
MedDream integration interface 22
Integration examples 26
MedDream TokenService description 26
Installation and configuration 27
Installation testing 30
Generate token request 31
Validate token request 35
Invalidate token request 35
Related configuration options 37
Table of Figures 39
Annex I. json structure and parameters description for API v1 40
.json structure example 40
Parameters description 40
Version related configuration 42
Annex II. json structure and parameters description for API v2 43
.json structure example 43
Parameters description 43
Version related configuration 46
API v2 differences from API v1 46
Annex III. json structure and parameters description for API v3 47
.json structure example 47
Parameters description 48
Version related configuration 51
API v3 differences from API v2 51
Annex IV. Changes in “permissions”: […] allowed values in v0.8 52
“permissions”: […] allowed values in v0.8 differences from v0.7 52
Annex V. Changes in “patient”: […] values validation in v0.9 53
“patient”: […] parameters validation in v0.9 differences from v0.8 53
Annex VI. Changes in “permissions”: […] allowed values in v1.0 54
“permissions”: […] allowed values in v1.0 differences from v0.9 54
Annex VII. Changes in “permissions”: […] allowed values in v1.1 55
“permissions”: […] allowed values in v1.1 differences from v1.0 55
Annex VIII. json structure and parameters description for API v4 56
"studies" object and "history" array object: allowed identifiers in v1.2 differences from v1.1 56
.json structure example 56
Parameters description 57
Version related configuration 60
API v4 differences from API v3 60
Annex IX. Changes in authentication in v1.3 61
Version related configuration 61
Annex X. json structure and parameters description for API v4 62
"restrictions" object and "series" array object: allowed identifiers in v1.4 62
“permissions”: […] allowed values in v1.4 differences from v1.1 62
.json structure example 62
Parameters description 63

Version related configuration 67
API v4 differences from API v3 67
Annex XI. json structure and parameters description for API v4 68
"groups" array, "institution" object in "user" object: allowed identifiers in v1.5.....68
.json structure example 68
Parameters description 69
Version related configuration 73
API v4 differences from API v3 73
Index 74

Document purpose

The purpose of this document is to provide a quick guide for integrating MedDream viewing functionality with information system (called Integrating information system or Integrating IS further in the document). It is intended for information system developers who aim for a fast and effective integration of MedDream viewing functionality.

If you have any questions or comments regarding this user's manual, please contact MedDream UAB Customer support: support@meddream.com .

Explanation of symbols used

The symbols used in this document refer to important information which warns against possible integration errors or security issues and contain useful notes. Whenever you see these symbols, read the accompanying information carefully and observe the recommendations.

WARNING! This indicates a hazardous situation which may cause patient death, potential injury or serious health impairment, requiring professional medical intervention.

CAUTION! This indicates a hazardous situation which may cause minor potential injury, not requiring professional medical intervention, or simply cause inconvenience to medical professionals using software without affecting patient health status or health changes.

NOTE! Information, hints and advice for a better understanding of the instructions to be observed in the operation of the instrument.

Introduction

MedDream is a HTML based package for PACS server which is designed to aid professionals in every day's decision-making process, connecting all the medical data into a unified and fast performing network. MedDream ensures a fast and reliable way to present and analyze the medical data (images and video files) on various devices: computers, smart phones, tablets and so forth.

MedDream can be used as a standalone WEB Viewer application as well as an URL-integrated WEB Viewer in other Integrating IS, such as PACS system, Radiology information system (RIS), or Hospital information system (HIS).

⚠ **NOTE!** The same MedDream installation can use both, login (application) and URL, authentication at the same time if particular authentication ways are permitted by configuration (see parameters in section **Related configuration options**). But only one active authentication (either login, or URL) is permitted from the particular work environment (one active session per browser at user 's workplace).

The following information is provided in the document:

- Sample scenarios for MedDream URL integration into RIS/HIS workflow with different URL parameters on page 8;
- Description of MedDream integration interface;
- MedDream TokenService description and integration interface;
- Related MedDream DICOM Viewer configuration options.

Other integration methods:

MedDream can be integrated via its Communication API – a JavaScript-based wrapper over the legacy/token URL method. See the Communication API specification (you can obtain it from support@meddream.com).

There is a third method where the image-displaying part of the frontend is added to your application as a React component, and you implement all buttons by yourself. See the Viewport API specification, (you can obtain it from support@meddream.com).

Minimum system requirements

Minimum hardware and software requirements for MedDream (server side) should be checked in Servicing MANUAL.

Minimum hardware and software requirements for MedDream TokenService are described in section **MedDream TokenService description**.

Scenarios for MedDream URL integration

MedDream URL integration supports two ways of parameter passing:

- Direct parameters in URL, further called as **token disabled mode**;
- Generating the token that is related with a particular parameter set (called **token parameters** further in the document) and passing the token as URL parameter instead of direct parameters usage in URL, further called as **token enabled mode**.

⚠ **NOTE!** One MedDream installation can work only in one mode at a time – either token enabled mode, or token disabled mode – that is enabled by configuration (see parameters in section **Related configuration options**).

The sample scenarios for MedDream URL integration in token enabled mode are described in section **"Token enabled MedDream URL integration scenarios using MedDream TokenService".** The supporting MedDream TokenService, that is required for token generation and validation, is described in section "

Integration examples

Integration into HIS samples can be found at docker hub https://hub.docker.com/r/meddream/orthanc-dicom-viewer.

MedDream TokenService description**"**. Scenario for MedDream URL integration in token disabled mode is described in section "**Token disabled MedDream URL integration scenario**".

⚠ **CAUTION!** Token disabled integration is not security safe.

MedDream URL integration supports three viewer window opening ways:

- Open the new MedDream viewer in browser and display the study or studies in it;
- Open the study or studies in already existing MedDream viewer window. Optional URL parameter 'add=true' should be used. The parameter can be used in both, token enabled and token disabled, modes. The direct parameters in URL mode is further called as token disabled mode;
- Open the new MedDream viewer with displayed study or studies in Iframe.

⚠ **NOTE!** Complete list of supported URL parameters and their combinations is provided in section **MedDream integration interface**.

⚠ **CAUTION!** If Iframe integration is used, it is recommended to set Iframe security configuration (see parameters in section **Related configuration options**) in order to reduce the risk of system hacking.

Token enabled MedDream URL integration scenarios using MedDream TokenService

The Integrating IS should implement two integration points in token enabled MedDream URL integration scenario:

- Integration with MedDream TokenService for token generation;
- Integration with MedDream viewing functionality.

The Integrating IS can implement these two integrations by two different scenarios:

- In one step, when token is generated after view study request from the user (see One step token enabled MedDream URL integration);
- In two separate steps, when Integrating system can initiate token generation and creation of MedDream URL with token, and afterwards the user can use this link (see Two steps token enabled MedDream URL integration).

The sample MedDream Viewer integration architecture with MedDream TokenService installed on separate server machine is displayed in diagram bellow.

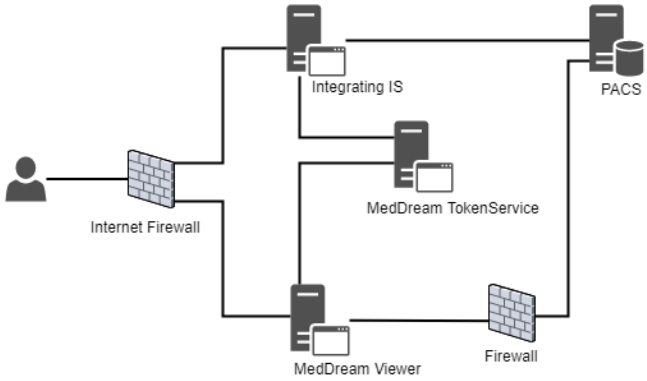

Figure 1. MedDream Viewer and MedDream TokenService integration architecture

The diagram below shows the integration related software components and communication interfaces. The request (numbered text) corresponds to the particular step in one step token enabled integration scenario (see Figure 3) and is described in details in table below the Figure 3.

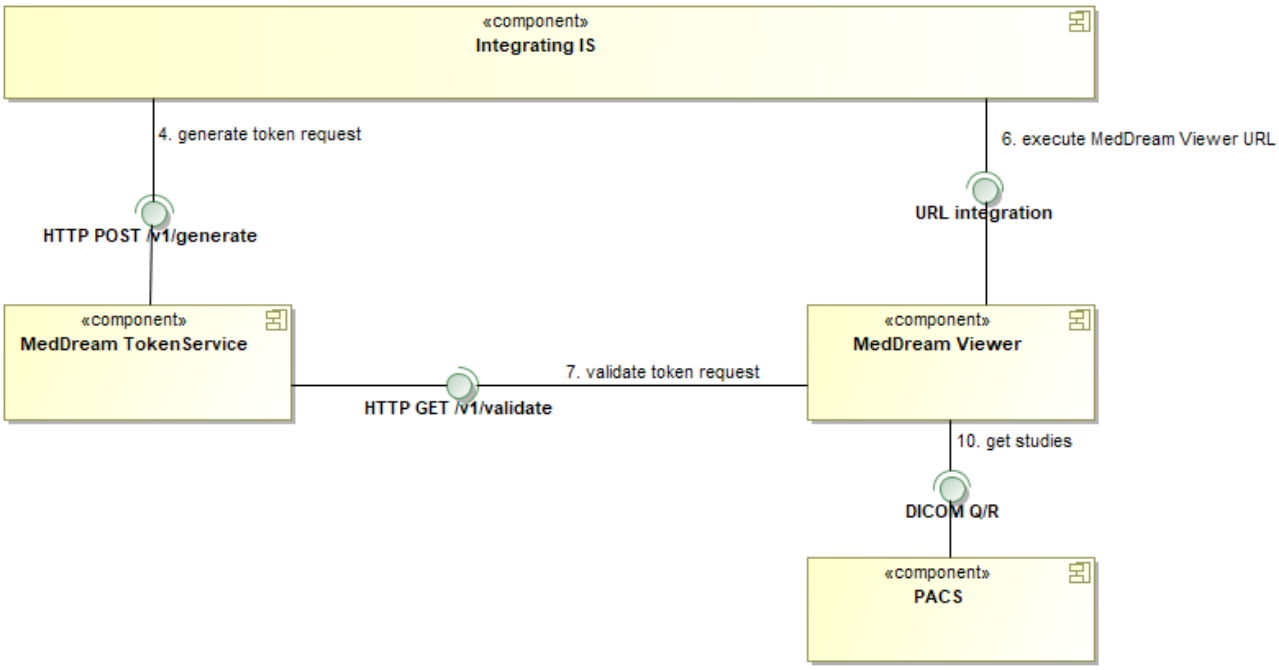

Figure 2. MedDream Viewer and MedDream TokenService integration components

One step token enabled MedDream URL integration

The diagram explains the interaction among user, Integrating information system, MedDream WEB DICOM Viewer and supporting MedDream TokenService. Each step is explained in the table below the diagram.

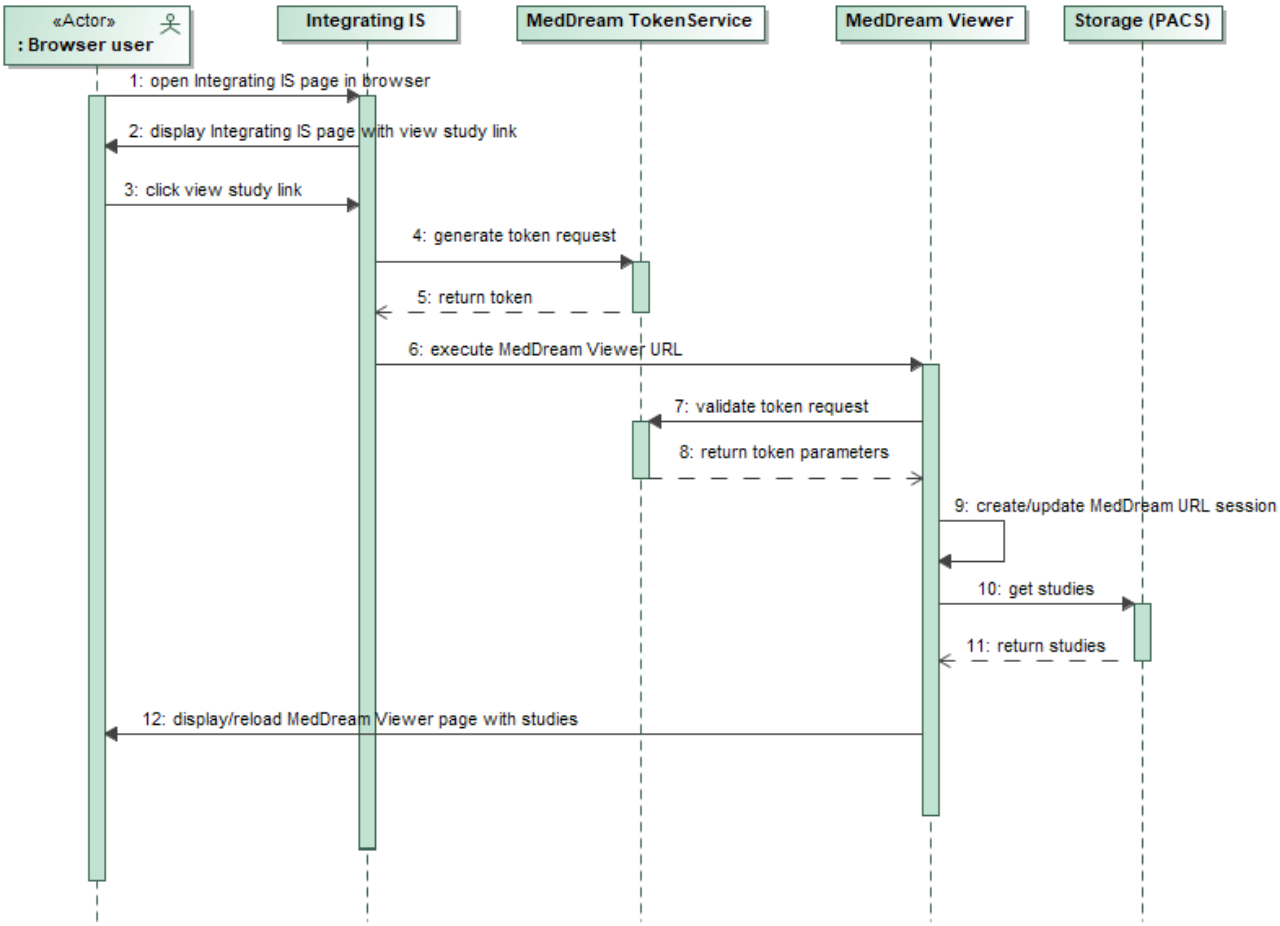

Figure 3. One step token enabled integration scenario

⚠ **NOTE!** The integration related steps that requires handling in the Integrating IS are underlined.

Scenario step	Description
1: open Integrating IS page in browser	The user that is authenticated and working in Integrating IS, asks to open page that contains link for opening the study in MedDream WEB DICOM Viewer page.
2: display Integrating IS page with view study link	The Integrating IS displaying the page with control that activates study or studies opening in MedDream viewer implementation. ⚠ **NOTE!** The user authentication and access to study control should be ensured by the Integrating IS.
3: click view study link	The user activates the studies opening in MedDream viewer.
<u>4: generate token request</u>	The Integrating IS sends the HTTP POST /v1/generate request to MedDream TokenService with parameters included in json body. See detailed request and parameters description in section **Generate token request** on page 31.
<u>5: return token</u>	The MedDream TokenService returns HTTP response with status code and generated token or error message. See detailed response description in section **Generate token request** on page 31.
<u>6: execute MedDream Viewer URL</u>	The Integrating IS creates the MedDream WEB DICOM Viewer URL with token parameter and launches it. See detailed token enabled URL description and launching examples in section **MedDream integration interface** on page 22.

Scenario step	Description
7: validate token request	The MedDream system sends the HTTP GET /v1/validate request to MedDream TokenService with token parameter that was specified in URL. See detailed request description in section **Validate token request** on page 35. **CAUTION!** The single use token will be disabled at the response with status code 200 without considering whether the studies are opened in viewer or process is canceled.
8: return token parameters	The MedDream TokenService returns HTTP response with status code and json file with parameters (studies, storages, permissions and other) for the token or error message. See detailed response description in section **Validate token request** on page 35.
9: create/update MedDream URL session	The MedDream system checks the MedDream authentication status of user's work environment (currently used browser) and performs the required actions: either updates the existing session by granting the access to the studies, viewing of which is authorized by token parameters, or initiates the user authentication based on token parameters and creates a new session. **CAUTION!** If the active MedDream session is not a URL session, the user will be prompted to choose either to close the current application user session and open a new URL session, or to stay logged in as application user. If the active MedDream session is a URL session with the permissions or restrictions, that differ from the permissions or restrictions returned in token parameters, then the active URL session and all opened MedDream windows and tabs will be closed and a new URL session will be created and a new MedDream viewer window will be opened.
10: get studies	The MedDream system requests the study data from storage. **NOTE!** The study identifier and storage identifier are retrieved from token parameters. If token contains multiple studies from multiple storages, multiple requests are made.
11: return studies	The study structure and image data are returned from storage.
12: display/reload MedDream Viewer page with studies	The MedDream system renders the viewer page with the study images and the page is displayed to user. **NOTE!** MedDream viewer integration provides several study opening ways: opening in a new browser window or tab, opening in already existing MedDream viewer window or tab, opening in iFrame included in page of Integrated IS. The opening way depends on launching code that is implemented by Integrating IS in step '6: execute MedDream URL'. See detail URL description and launching examples in section **MedDream integration interface** on page 22. **NOTE!** The system may be configured (see section **Related configuration options**) to open the patient studies modal with studies list instead of opening the studies directly in the Viewer, if token contains studies for only one patient, identified by patient ID, and system properties.

Two steps token enabled MedDream URL integration

The diagram explains the interaction among user, Integrating information system, MedDream WEB DICOM Viewer and supporting MedDream TokenService. Each step is explained in the table below the diagram.

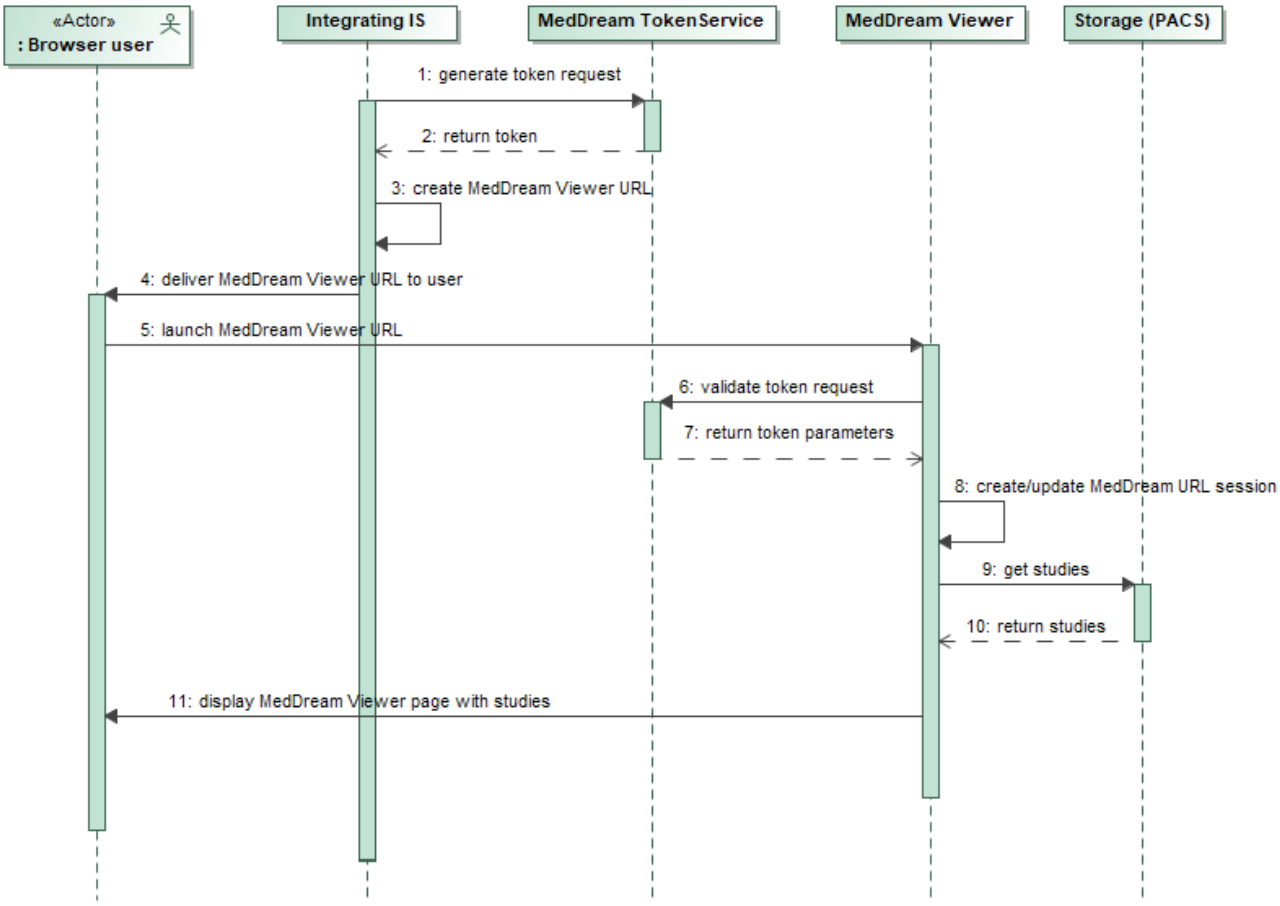

Figure 4. Two steps token enabled integration scenario

NOTE! The integration related steps that requires handling in the Integrating IS are underlined.

Scenario step	Description
1: generate token request	The Integrating IS sends the HTTP POST /v1/generate request to MedDream TokenService with parameters included in json body. See detailed request and parameters description in section **Generate token request** on page 31.
2: return token	The MedDream TokenService returns HTTP response with status code and generated token or error message. See detail response description in section **Generate token request** on page 31.
3: create MedDream Viewer URL	The Integrating IS creates the MedDream WEB DICOM Viewer URL with token parameter. See detailed token enabled URL description and launching examples in section **MedDream integration interface** on page 22.
4: deliver MedDream Viewer URL to user	The Integrating IS delivers the link to user, for example, sends it by e-mail.
5. launch MedDream Viewer URL	The user opens the received link in browser.

Scenario step	Description
6: validate token request	The MedDream system sends the HTTP GET /v1/validate request to MedDream TokenService with token parameter that was specified in URL. See detail request description in section **Validate token request** on page 35. **CAUTION!** The single use token will be disabled at the response with status code 200 without considering whether the studies are opened in viewer or process is canceled.
7: return token parameters	The MedDream TokenService returns HTTP response with status code and json file with parameters (studies, storages, permissions and other) for the token or error message. See detail response description in section **Validate token request** on page 35.
8: create/update MedDream URL session	The MedDream system checks the MedDream authentication status of user's work environment (currently used browser) and performs the required actions: either updates the existing session by granting the access to the studies, viewing of which is authorized by token parameters, or initiates the user authentication based on token parameters and creates the new session. **CAUTION!** If the active MedDream session is not a URL session, the user will be prompted to choose either to close the current application user session and open a new URL session, or to stay logged in as application user. If the active MedDream session is a URL session with the permissions or restrictions, that differ from the permissions or restrictions returned in token parameters, then the active URL session and all opened MedDream windows and tabs will be closed and a new URL session will be created and a new MedDream viewer window will be opened.
9: get studies	The MedDream system requests the study data from storage. **NOTE!** The study identifier and storage identifier are retrieved from token parameters. If token contains multiple studies from multiple storages, multiple requests are made.
10: return studies	The study structure and image data are returned from storage.
11: display MedDream Viewer page with studies	The MedDream system renders the viewer page with the study images and the page is displayed to user. **NOTE!** MedDream viewer integration provides several study opening ways: opening in a new browser window or tab, opening in already existing MedDream viewer window or tab, opening in Iframe included in page of Integrated IS. The opening way depends on link that is created by Integrating IS in step '3: create MedDream URL'. See detail URL description and launching examples in section **MedDream integration interface** on page 22. **NOTE!** The system may be configured (see section **Related configuration options**) to open the patient studies modal with studies list instead of opening the studies directly in the Viewer, if token contains studies for only one patient, identified by patient ID, and system properties. **CAUTION!** Opening MedDream viewer in Iframe is not recommended in two steps scenario.

Token enabled MedDream URL integration scenario using 3rd party token generation & validation service

⚠ **NOTE!** Instead of MedDream TokenService the Integrating IS can use other token generation and validation service implementation as long as it supports the identical token validation interface (see section **Validate token request**) and parameters that are used by MedDream.

Integrating IS can use the 3rd party self-standing application for token generation and validation. In this case the integration architecture and scenarios would be identical as with MedDream TokenService (see section **Token enabled MedDream URL integration scenarios using MedDream TokenService**).

Integrating IS can implement the token generation and validation services itself. The diagrams bellow describes the components and scenario for token enabled MedDream URL integration with token generation and validation implemented by the Integrating IS.

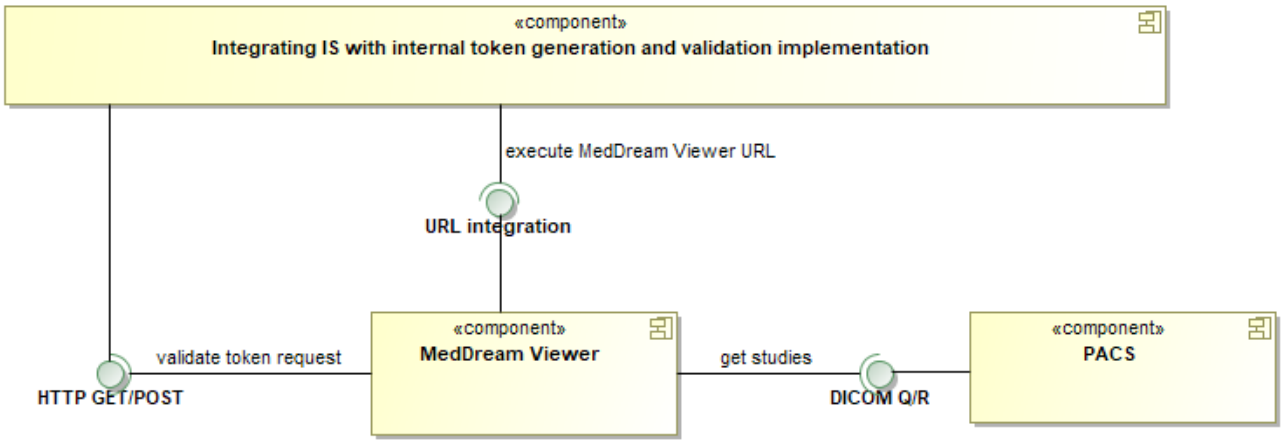

Figure 5. MedDream Viewer and Integrating IS with internal token service implementation integration components

The description of the communication interfaces is provided in the subsections with scenarios description.

The MedDream supports two of 3rd part token using scenarios:

- Scenario when the token is used exactly as in token enabled MedDream URL integration is described in section ;
- Scenario using the JSON Web Token (JWT) is described in section .

Integration scenario using MedDream token

The diagram bellow explains the interaction among user, Integrating information system, and MedDream WEB DICOM Viewer, when integrating IS uses MedDream type token. Integrating IS can implement the token generation and validation services itself, or use the 3rd party self-standing application for token generation and validation. Each step is explained in the table below the diagram.

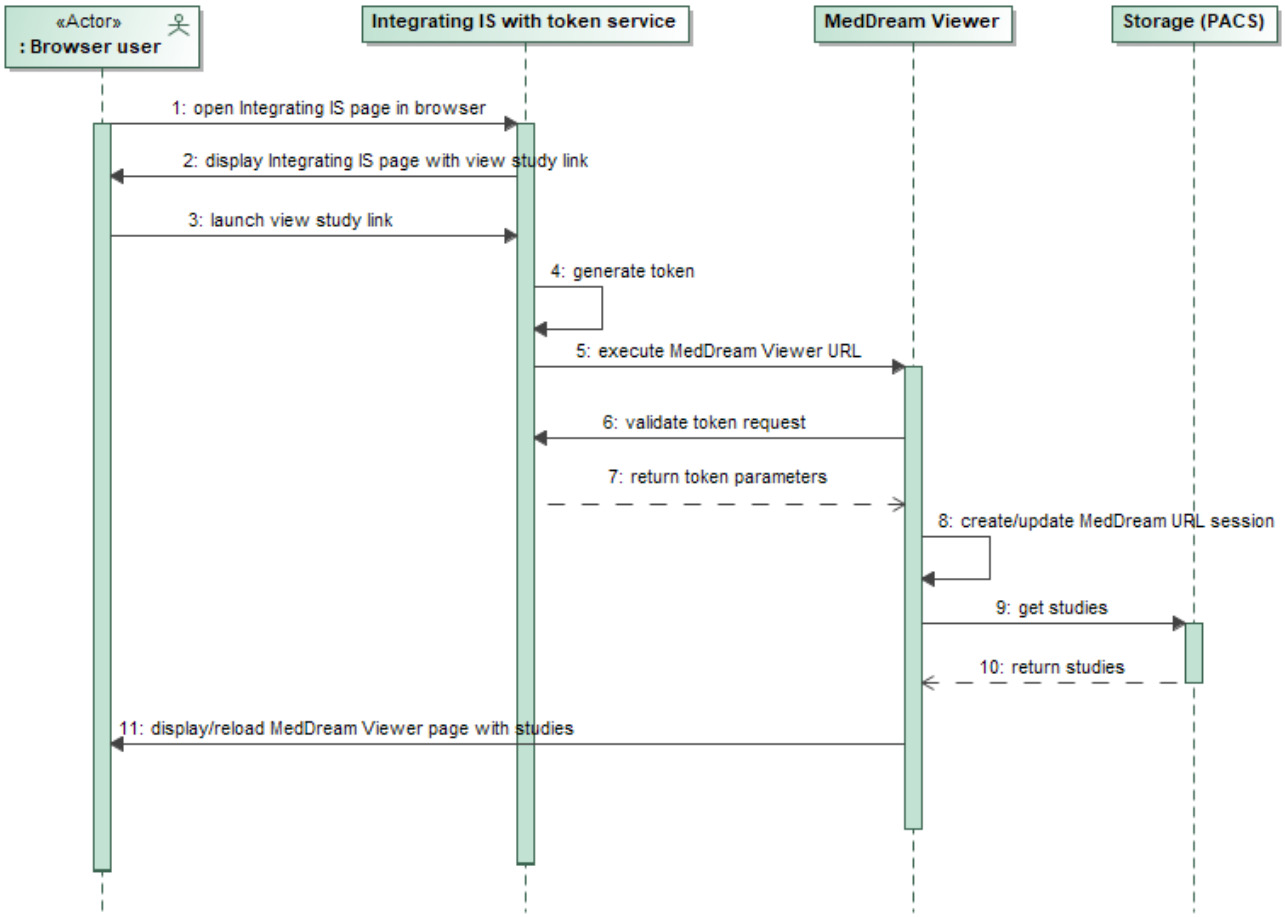

Figure 6. Token enabled integration with MedDream type token service implementation in Integrating IS scenario

NOTE! The integration related steps that require handling in the Integrating IS are underlined.

Scenario step	Description
1: open Integrating IS page in browser	The user that is authenticated and working in Integrating IS, asks to open page that contains link for opening the study in MedDream WEB DICOM Viewer page.
2: display Integrating IS page with view study link	The Integrating IS displaying the page with control that activates study or studies opening in MedDream viewer implementation. **NOTE!** The user authentication and access to study control should be ensured by the Integrating IS.
3: launch view study link	The user activates the studies opening in MedDream viewer.
<u>4: generate token</u>	The Integrating IS should implement the token generation functionality and generate the token, which would be passed as parameter in MedDream URL request and validate token request. **NOTE!** The token should not include URL escape characters.
<u>5: execute MedDream Viewer URL</u>	The Integrating IS creates the MedDream WEB DICOM Viewer URL with token parameter and launches it. See detailed token enabled URL description and launching examples in section **MedDream integration interface** on page 22.

Scenario step	Description
6: validate token request	The MedDream system sends the HTTP GET /v1/validate request to Integrating IS token validation service with token parameter that was specified in URL. The service should return the parameters in json body, as described in section **Validate token request** on page 35. **CAUTION!** The Integrating IS should implement the same token validation API as described in section **Validate token request** on page 35.
7: return token parameters	The Integrating IS token validation service should return HTTP response with status code and json file with parameters (studies, storages, permissions and other) for the token or error message. **CAUTION!** The required parameters must be returned. The optional parameter may be returned, if the parameters-related functionality is required in Integrating IS.
8: create/update MedDream URL session	The MedDream system checks the MedDream authentication status of user's work environment (currently used browser) and performs the required actions: either updates the existing session by granting the access to the studies, viewing of which is authorized by token parameters, or initiates the user authentication based on token parameters and creates a new session. **CAUTION!** If the active MedDream session is not a URL session, the user will be prompted to choose either to close the current application user session and open a new URL session, or to stay logged in as application user. If the active MedDream session is a URL session with the permissions or restrictions, that differs from the permissions or restrictions returned in token parameters, then the active URL session and all opened MedDream windows and tabs will be closed and a new URL session will be created and a new MedDream viewer window will be opened.
9: get studies	The MedDream system requests the study data from storage. **NOTE!** The study identifier and storage identifier are retrieved from token parameters. If token contains multiple studies from multiple storages, multiple requests are made.
10: return studies	The study structure and image data are returned from storage.
11: display/reload MedDream Viewer page with studies	The MedDream system renders the viewer page with the study images and the page is displayed to user. **NOTE!** MedDream viewer integration provides several study opening ways: opening in a new browser window or tab, opening in already existing MedDream viewer window or tab, opening in iFrame included in page of Integrated IS. The opening way depends on launching code that is implemented by Integrating IS in step '6: execute MedDream URL'. See detail URL description and launching examples in section **MedDream integration interface** on page 22. **NOTE!** The system may be configured (see section **Related configuration options**) to open the patient studies modal with studies list instead of opening the studies directly in the Viewer, if token contains studies for only one patient, identified by patient ID, and system properties.

Integration scenario using JSON Web Token (JWT)

The diagram bellow explains the interaction among user, Integrating information system, and MedDream WEB DICOM Viewer, when integrating IS uses JSON Web Token (JWT). Integrating IS can implement the token generation and validation

services itself, or use the 3rd party self-standing application for token generation and validation. Each step is explained in the table below the diagram.

NOTE! jwt parameter support is available in MedDream starting from version v.7.8.0. MedDream TokenService does not support jwt. Integrating IS should implement custom token generation and validation services.

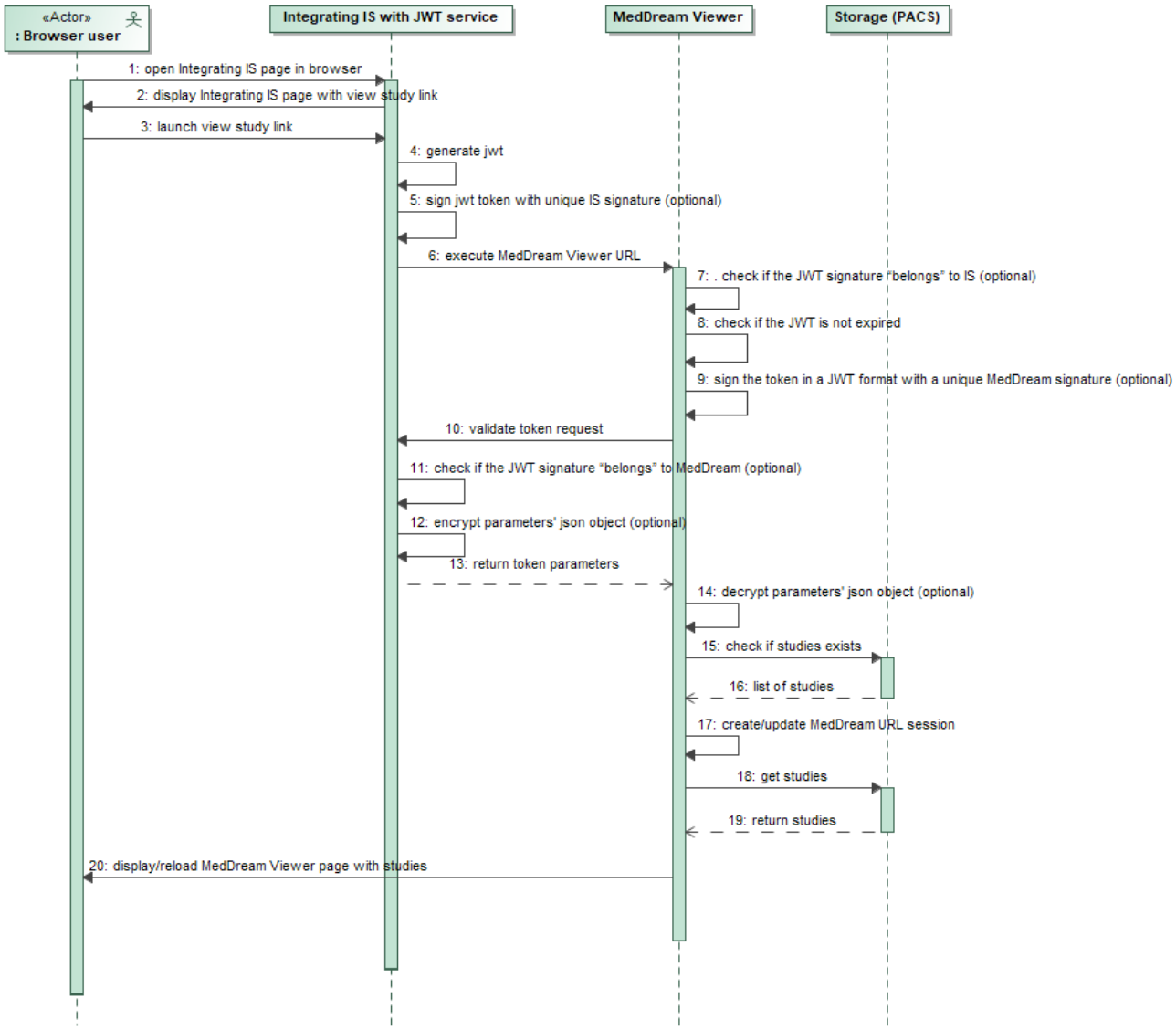

Figure 7. Token enabled integration with JSON Web Token service implementation in Integrating IS scenario

NOTE! The integration related steps that require handling in the Integrating IS are underlined.

Scenario step	Description
1: open Integrating IS page in browser	The user that is authenticated and working in Integrating IS, asks to open page that contains link for opening the study in MedDream WEB DICOM Viewer page.

Scenario step	Description
2: display Integrating IS page with view study link	The Integrating IS displaying the page with control that activates study or studies opening in MedDream viewer implementation. ⚠ **NOTE!** The user authentication and access to study control should be ensured by the Integrating IS.
3: launch view study link	The user activates the studies opening in MedDream viewer.
4: generate jwt	The Integrating IS should implement the token generation functionality and generate the token, which would be passed as parameter in MedDream URL request and validate token request. ⚠ **NOTE!** The token should not include URL escape characters. It should be URL encoded.
5: sign jwt token with unique IS signature (optional)	This step is optional. If IS wants MedDream to validate if jwt was signed by IS, it should use this step and sign jwt token using its own secret key (base64 encoded).
6: execute MedDream Viewer URL	The Integrating IS creates the MedDream WEB DICOM Viewer URL with token parameter and launches it. It uses HTTP GET request: http|https://IP:port/?jwt=<jwt token> See detailed token enabled URL description and launching examples in section **MedDream integration interface** on page 22
7: check if the JWT signature “belongs” to IS (optional)	This step is optional and depends on MedDream configuration. If `authentication.his.jwtServerKey` parameter is set in application.properties, MedDream will use this secret key to validate if token is signed by IS. ⚠ **NOTE!** Step 5 must be executed before step 6. Also, the validation will be applied only if `authentication.his.jwtMedDreamKey` is set.
8: check if the JWT is not expired	If "exp" property of the PAYLOAD: DATA is defined in jwt token, MedDream will check if the value defined in “exp” field is more than current time of the MedDream server.
9: sign the token in a JWT format with a unique MedDream signature (optional)	This step is optional and depends on MedDream configuration. It can be used if IS wants to validate that jwt token was sent by MedDream. MedDream will sign jwt token using its own secret key (base64 encoded) before sending to IS for validation. The following parameter should be set in application.properties to enable signature: `authentication.his.jwtMedDreamKey`. ⚠ **NOTE!** This functionality will work only if step 7 is configured.
10: validate token request	The MedDream system sends the HTTP POST request to Integrating IS jwt token validation service to validate the jwt token. The jwt token will be sent in authorization header using syntax Bearer: <jwt token>. The jwt token will be signed by meddream if steps 8 and 10 are enabled, otherwise, meddream will send jwt token that was specified in URL The service should return the parameters in json body, as described in section **Validate token request** on page 35.
11: check if the JWT signature “belongs” to MedDream (optional)	This step is optional and depends on MedDream configuration. If IS wants to use such validation, steps 5, 7 and 9 must be executed before.

Scenario step	Description

12: encrypt parameters' json object (optional)	MedDream allows sending encrypted parameters json using AES/GCM/NoPadding algorithm. The following properties should be configured in both systems:
	`authentication.his.responseDecryptPassword`
	`authentication.his.responseDecryptSalt`
	If enabled, IS should encrypt parameters json using AES/GCM/NoPadding algorithm and properties defined above.

13: return token parameters	The Integrating IS token validation service should return HTTP response with status code and json file with parameters (studies, storages, permissions and other) for the token or encrypted json file if step 12 is enabled or error message. **NOTE!** If step 12 is enabled, the encrypted string should be URL encoded before sending to MedDream. **CAUTION!** The required parameters must be returned. The optional parameter may be returned, if the parameters-related functionality is required in Integrating IS.

14: decrypt parameters' json object (optional)	This step is optional and depends on MedDream configuration. If the following properties are configured:
	`authentication.his.responseDecryptPassword`
	`authentication.his.responseDecryptSalt`
	MedDream will decrypt parameters json before using it.
	NOTE! This step will work correctly if the step 12 was executed before.

15. check if studies exist	MedDream tries to find the studies in the PACS according to the parameters defined in json files.
16. list of studies	PACS returns list of studies according to the parameters defined in json files.
17: create/update MedDream URL session	The MedDream system checks the MedDream authentication status of user's work environment (currently used browser) and performs the required actions: either updates the existing session by granting the access to the studies, viewing of which is authorized by token parameters, or initiates the user authentication based on token parameters and creates a new session. **CAUTION!** If the active MedDream session is not a URL session, the user will be prompted to choose either to close the current application user session and open a new URL session, or to stay logged in as application user. If the active MedDream session is a URL session with the permissions or restrictions, that differs from the permissions or restrictions returned in token parameters, then the active URL session and all opened MedDream windows and tabs will be closed and a new URL session will be created and a new MedDream viewer window will be opened.
18: get studies	The MedDream system requests the study data from storage. **NOTE!** The study identifier and storage identifier are retrieved from token parameters. If token contains multiple studies from multiple storages, multiple requests are made.
19: return studies	The study structure and image data are returned from storage.

Scenario step	Description
20: display/reload MedDream Viewer page with studies	The MedDream system renders the viewer page with the study images and the page is displayed to user. **NOTE!** MedDream viewer integration provides several study opening ways: opening in a new browser window or tab, opening in already existing MedDream viewer window or tab, opening in iFrame included in page of Integrated IS. The opening way depends on launching code that is implemented by Integrating IS in step '6: execute MedDream URL'. See detail URL description and launching examples in section **MedDream integration interface** on page 22. **NOTE!** The system may be configured (see section **Related configuration options**) to open the patient studies modal with studies list instead of opening the studies directly in the Viewer, if token contains studies for only one patient, identified by patient ID, and system properties.

Token disabled MedDream URL integration scenario

CAUTION! Token disabled integration is not security save.

The diagram explains the interaction among user, Integrating information system and MedDream WEB DICOM Viewer. Each step is explained in the table below the diagram.

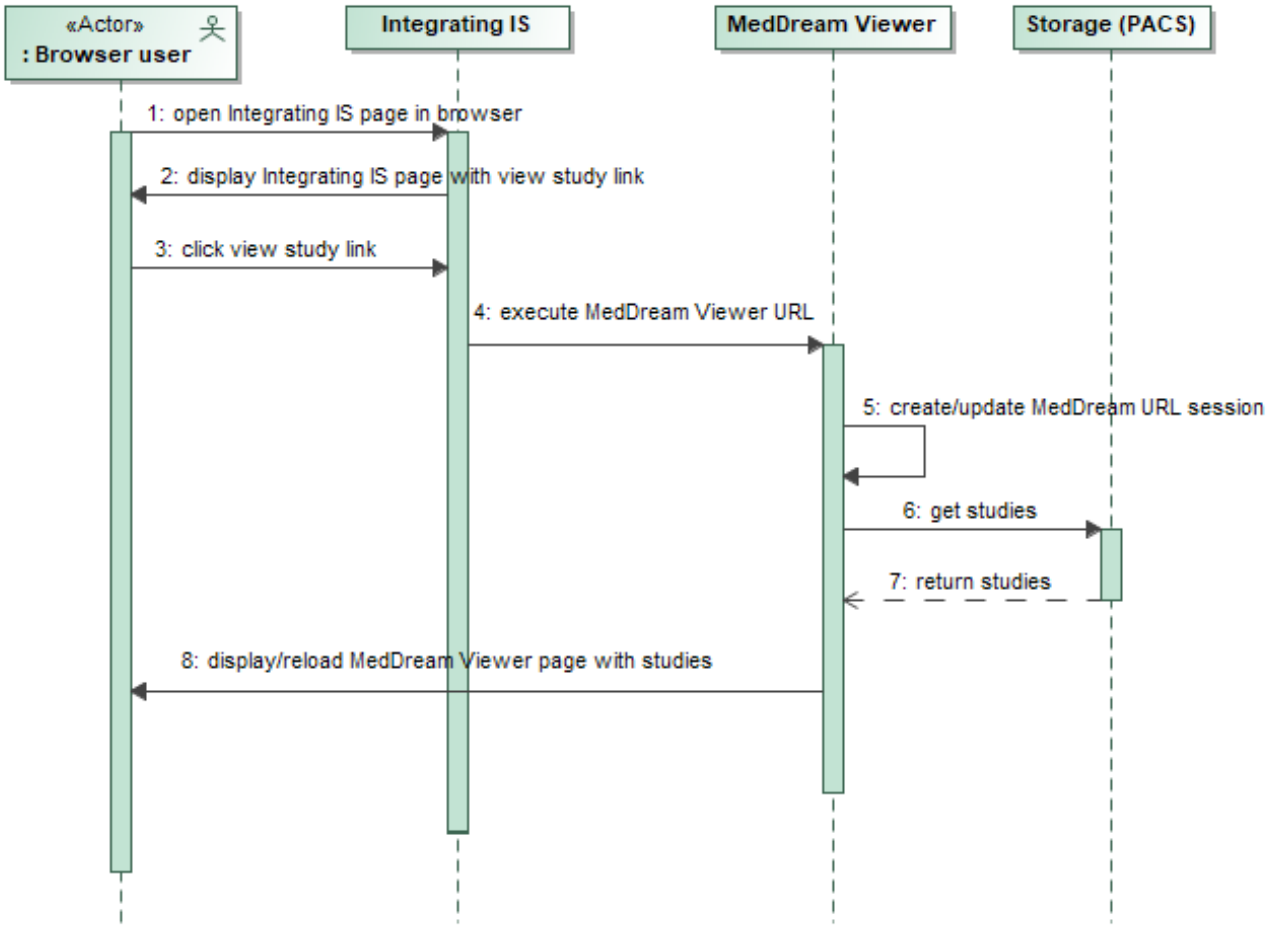

Figure 8. Token disabled integration scenario

⚠ **NOTE!** The integration related steps that requires handling in the Integrating IS are underlined.

Scenario step	Description
1: open Integrating IS page in browser	The user that is authenticated and working in Integrating IS, asks to open page that contain link for opening the study in MedDream WEB DICOM Viewer page.
2: display Integrating IS page with view study link	The Integrating IS displays the page with control that activates study or studies opening in MedDream viewer implementation. ⚠ **NOTE!** The user authentication and access to study control should be ensured by the Integrating IS.
3: click view study link	The user activates the studies opening in MedDream viewer.
4: execute MedDream Viewer URL	The Integrating IS creates the MedDream WEB DICOM Viewer URL with study or patient indicating parameter (for example, study UID) and launches it. See detail URL description and launching examples in section **MedDream integration interface** on page 22.

Scenario step	Description
5: create/update MedDream URL session	The MedDream system checks the MedDream authentication status of user's work environment (currently used browser) and performs the required actions: either updates the existing session by granting the access to the studies, viewing of which is authorized by parameters, or initiates the user authentication based on URL parameters and creates the new session. ⚠ **CAUTION!** If the active MedDream session is not URL session, the user will be prompted to choose either to close the current application user session and open a new URL session, or to stay logged in as application user. ⚠ **CAUTION!** The default behavior is to destroy the active URL session and to create a new URL session with access to studies of the last executes URL. Set 'authentication.his.useSameSession=true' in application properties (see parameters in section **Related configuration options**) to keep access to the studies of currently active. URL session. ⚠ **NOTE!** The token disabled integration use the same permissions for all URL user authentication and these permissions must be configured in application properties (see parameters in section **Related configuration options**).
6: get studies	The MedDream system requests the study data from storage. If multiple storages are configured in the system, multiple requests are made ⚠ **NOTE!** Multiple storage support added in MedDream v8.3.0. ⚠ **NOTE!** By default, the search is performed in all configures storages. Configure and use storage parameter in URL to perform search in particular storages.
7: return studies	The study structure and image data are returned from storage.
8: display/reload MedDream viewer page with studies	The MedDream system renders the viewer page with the study images and the page is displayed to user. ⚠ **NOTE!** MedDream viewer integration provides several study opening ways: opening in a new browser window or tab, opening in already existing MedDream viewer window or tab, opening in iFrame included in page of Integrated IS. The opening way depends on launching code that is implemented by Integrating IS in step '4: execute MedDream URL'. See detail URL description and launching examples in section **MedDream integration interface** on page 22. ⚠ **NOTE!** The system may be configured (see section **Related configuration options**) to open the patient studies modal with studies list instead of opening the studies directly in the Viewer, if patient parameter is used in URL.

MedDream integration interface

The section describes supported MedDream WEB DICOM viewer integration URLs and notes for integration developers.

• **Supported URL** **Short description** ***URL examples***
Token enabled URLs: • http|https://IP:port/?token=…

- http|https://IP:port/?token=…&add=true
- http|https://IP:port/?token=…&replace=true

The token that is generated by MedDream TokenService should be passed in URL. The optional parameter 'add=true' should be used if opening the studies from different URLs in one MedDream viewer window is required. The optional parameter 'replace=true' should be used if the studies, that are opened in MedDream viewer window, should be replaced with the studies from new URL in the same MedDream viewer window. Otherwise a different URL will result in a new independent viewer window with a different set of studies.

https://demo.meddream.com/?token=
ebk8n56tRY_SX0gbvSIAfLHEWl3_AK6xGI1vRCMbkDYDfSjyoROo7_uZJ1pSZo6N5IXE49HAd1i4IVF7Idw7_vJYKzQocwX-1iZxtRNfm6w=

http://127.0.0.1:8085/?token=
ebk8n56tRY_SX0gbvSIAfLHEWl3_AK6xGI1vRCMbkDYDfSjyoROo7_uZJ1pSZo6N5IXE49HAd1i4IVF7Idw7_vJYKzQocwX-1iZxtRNfm6w=&add=true

http://127.0.0.1:8085/?token=
ebk8n56tRY_SX0gbvSIAfLHEWl3_AK6xGI1vRCMbkDYDfSjyoROo7_uZJ1pSZo6N5IXE49HAd1i4IVF7Idw7_vJYKzQocwX-1iZxtRNfm6w=&replace=true

- http|https://IP:port/?jwt=…
- http|https://IP:port/?jwt=…&add=true
- http|https://IP:port/?jwt=…&replace=true

NOTE! jwt parameter support is available in MedDream starting from version v.7.8.0. MedDream TokenService does not support jwt. Integrating IS should implement custom token generation and validation services.

The jwt as parameter should be passed in URL. The optional parameter 'add=true' should be used if opening the studies from different URLs in one MedDream viewer window is required. The optional parameter 'replace=true' should be used if the studies, that are opened in MedDream viewer window, should be replaced with the studies from new URL in the same MedDream viewer window. Otherwise a different URL will result in a new independent viewer window with a different set of studies.

http://127.0.0.1:8085/?jwt=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJqdGkiOiI2NDQzN2YzMC0yM2MyLTQ0NDEtOThkNi0wNWUxZGEzYzYzNGEiLCJuYmYiOjE2MjI0NTExNTcsImV4cCI6MTYyMjQ1MTIxNywiaWF0IjoxNjIyNDUxMTU3fQ.SkZR_Y2I1PfLiO_Z9Uk1TZS7B35DNZyX43SA_pgnfz0&add=true

Token disabled URLs:

CAUTION! To keep the studies from previously opened URLs accessible in separate or the same (if optional parameter 'add=true' was used) Viewer window, 'authentication.his.useSameSession=true' must be set in application properties (see parameters in section **Related configuration options**).

NOTE! If parameter's value has symbols, that could not be used in URL, such values should be encoded.

NOTE! By default, token disabled URL session will not be created, if more than one study is found according to the URL parameters. Set authentication.his.maxObjects to required number (see parameters in section **Related configuration options**), if URL refers several studies and should open them in Viewer window.

- http|https://IP:port/?study=…
- http|https://IP:port/?study=…&add=true
- http|https://IP:port/?study=…&replace=true

- http|https://IP:port/?study=…,…,…
- http|https://IP:port/?study=…,…,…&add=true
- http|https://IP:port/?study=…,…,…&replace=true
- http|https://IP:port/?study=…,…,…&storage=…,…

The study UID should be passed in URL. Comma separated multiple study UIDs are allowed. The optional parameter ‘add=true’ should be used if opening the studies from different URLs in one MedDream viewer window is required. The optional parameter ‘replace=true’ should be used if the studies, that are opened in MedDream viewer window, should be replaced with the studies from new URL in the same MedDream viewer window. The optional parameter ‘storage’ should be used, if multiple storages are configured and search should be performed in particular storages. Comma separated storage names are allowed. The passed names are validated according MedDream storage configuration.

http://demo.meddream.com/?study=1.2.826.0.1.3680043.8.1055.1.20160922221651432.55928341.45596087

https://demo.meddream.com/?study=1.2.826.0.1.3680043.8.1055.1.20160922221651432.55928341.45596087&add=true

https://demo.meddream.com/?study=1.2.826.0.1.3680043.8.1055.1.20160922221651432.55928341.45596087&replace=true

https://demo.meddream.com/?study=1.2.826.0.1.3680043.8.1055.1.20160922221651432.55928341.45596087, 1.2.840.113619.2.203.4.2147483647.1461855195.632174

https://demo.meddream.com/?study=1.2.826.0.1.3680043.8.1055.1.20160922221651432.55928341.45596087, 1.2.840.113619.2.203.4.2147483647.1461855195.632174&storage=samplePACS1,samplePACS2

- http|https://IP:port/?accnum=…
- http|https://IP:port/?accnum=…&add=true
- http|https://IP:port/?accnum=…&replace=true
- http|https://IP:port/?accnum=…&storage=…,…

The accession number should be passed in URL. The optional parameter ‘add=true’ should be used if opening the studies from different URLs in one MedDream viewer window is required. The optional parameter ‘replace=true’ should be used if the studies, that are opened in MedDream viewer window, should be replaced with the studies from new URL in the same MedDream viewer window. The optional parameter ‘storage’ should be used, if multiple storages are configured and search should be performed in particular storages. Comma separated storage names are allowed. The passed names are validated according MedDream storage configuration.

http://demo.meddream.com/?accnum=2016042610594598

https://demo.meddream.com/?accnum=2016042610594598&add=true

https://demo.meddream.com/?accnum=2016042610594598&replace=true

http://demo.meddream.com/?accnum=2016042610594598&storage=samplePACS1,samplePACS2

- http|https://IP:port/?patient=…
- http|https://IP:port/?patient=…&add=true
- http|https://IP:port/?patient=…&storage=…,…

The patient ID should be passed in URL. The optional parameter ‘add=true’ should be used if opening the studies from different URLs in one MedDream viewer window is required. The optional parameter ‘replace=true’ should be used if the studies, that are opened in MedDream viewer window, should be replaced with the studies from new URL in the same MedDream viewer window. The optional parameter ‘storage’ should be used, if multiple storages are configured and search should be performed in particular storages. Comma separated storage names are allowed. The passed names are validated according MedDream storage configuration.

https://demo.meddream.com/?patient=0

http://demo.meddream.com/?patient=0&add=true

http://demo.meddream.com/?patient=0&replace=true

https://demo.meddream.com/?patient=0&storage=samplePACS1,samplePACS2

- http|https://IP:port/?patient=…&accnum=…
- http|https://IP:port/?patient=…&accnum=…&add=true
- http|https://IP:port/?patient=…&accnum=…&replace=true
- http|https://IP:port/?patient=…&accnum=…&storage=…,…

The patient ID and accession number should be passed in URL. Both parameters are required. The optional parameter 'add=true' should be used if opening the studies from different URLs in one MedDream viewer window is required. The optional parameter 'replace=true' should be used if the studies, that are opened in MedDream viewer window, should be replaced with the studies from new URL in the same MedDream viewer window. The optional parameter 'storage' should be used, if multiple storages are configured and search should be performed in particular storages. Comma separated storage names are allowed. The passed names are validated according MedDream storage configuration.

http://demo.meddream.com/?patient=0&accnum=2016042610594598

https://demo.meddream.com/?patient=0&accnum=2016042610594598&add=true

https://demo.meddream.com/?patient=0&accnum=2016042610594598&replace=true

http://demo.meddream.com/?patient=0&accnum=2016042610594598&storage=samplePACS1,samplePACS2

- http|https://IP:port/?file=…
- http|https://IP:port/?file=…&add=true
- http|https://IP:port/?file=…&replace=true
- http|https://IP:port/?file=…,…,…
- http|https://IP:port/?file=…,…,…&add=true
- http|https://IP:port/?file=…,…,…&replace=true

The relative path from rootDirectory (property of FileSystem plugin's configuration) to studies DICOM file or catalog with DICOM files should be passed in URL. The maxDepth (property of FileSystem plugin's configuration) defines the depth of search starting from the passed catalog. Comma separated multiple values are allowed. The optional parameter 'add=true' should be used if opening the studies from different URLs in one MedDream viewer window is required. The optional parameter 'replace=true' should be used if the studies, that are opened in MedDream viewer window, should be replaced with the studies from new URL in the same MedDream viewer window.

⚠ **NOTE!** The file parameter may be used only with FileSystem plugin. The PATIENT_HISTORY permission should not be used with URLs using file parameter.

http://demo.meddream.com/?study= test_catalog

https://demo.meddream.com/?study=test_catalog/test_study_0/1.2.826.0.1.3680043.8.1055.1.20160525124158043.638645944.8314464.dcm &add=true

https://demo.meddream.com/?study=test_catalog/test_study_0/1.2.826.0.1.3680043.8.1055.1.20160525124158043.638645944.8314464.dcm &replace=true

https://demo.meddream.com/?study=test_catalog/test_study_0,test_catalog/test_study_001

⚠ **NOTE!** If parameter ‘add=true‘ or ‘replace=true’ is used, the javascript function window.open should be used for opening the study from browser page to ensure that an existing MedDream viewer window is found and the study is opened in it. Using a static link in form of HTML HREF attribute may not found the existing viewer and would open the study in a new viewer.

⚠ **NOTE!** When add study link is open directly in browser URL, the primary window is left open after success link open in MedDream viewer. The window cannot be closed due to browser security (Scripts may close only the windows that were opened by it).

Integration examples

Integration into HIS samples can be found at docker hub https://hub.docker.com/r/meddream/orthanc-dicom-viewer.

MedDream TokenService description

The MedDream TokenService is a JAVA application that provides token generation and token validation services. The section describes the following:

- MedDream TokenService installation and configuration guidelines;
- Token generation interface;
- Token validation interface;
- Token invalidation interface;
- Token parameters.

Token parameters, that may be used in json body and passed to TokenService, vary depending on API version. The table below describes the API support in different TokenService versions, and provides the link to appendix with parameters description for a particular API version. In addition, information about compatibility with MedDream versions is provided.

TokenService version	API list	Support of released API in MedDream
v0.5	Released API v1 (see description in Annex I): • POST /v1/generate • GET /v1/validate Supported API versions: v1	MedDream v7.5.1, v7.5.2, v7.6, v7.7.0, v7.8.0, v7.8.1, v7.9.0
v0.6	Released API v2 (see description in Annex II): • POST /v2/generate • GET /v2/validate Supported API versions: v1, v2	MedDream v7.6, v7.7.0, v7.8.0, v7.8.1, v7.9.0
v0.7	Released API v3 (see description in Annex III): • POST /v3/generate • GET /v3/validate • DELETE /v3/invalidate Supported API versions: v1, v2, v3	MedDream v7.8.0, v7.8.1, v7.9.0
v0.8	Updated API v3 (see description in Annex IV) parameters validation Supported API versions: v1, v2, v3	New values support: MedDream v7.9.0

v0.9	Updated *"patient": ["pat2"]* validation (see description in Annex V). Supported API versions: v1, v2, v3	MedDream v7.9.0, v8.0.0
v1.0	Updated *" permissions": […]* allowed values (see description in Annex VI).	New values support: MedDream v8.1.0
v1.1	Updated *" permissions": […]* allowed values (see description in Annex VII). Updated service configuration for multiple instances (load balance) support, secret-key and initialization-vector parameters added. See configuration items in next section.	New values support: MedDream v8.2.0
v1.2	Updated "studies" object and "history" array object allowed study or studies identifiers (see description in Annex VIII). Released API v4 (see description in Annex VIII): • POST /v4/generate • GET /v4/validate • DELETE /v4/invalidate Supported API versions: v1, v2, v3, v4	Limited support starting from MedDream v7.5.1 All token values support starting from MedDream v8.3.0
V1.3	Authentication for generate and validate endpoints implemented.	Authentication support: starting from MedDream v8.4.0
V1.4 Internal [2.1.8]	Updated “restrictions” object and “series” array object allow to filter which series of the study to display. Other series will be not be displayed. Some limitations apply (see description in Annex X). Updated " permissions": […] allowed values (see description in Annex X).	Filtering series which will be displayed: starting from MedDream v.8.5.0 and token-service 2.1.8 New “permissions” values support from MedDream v.8.5.0 and token-service 2.1.8
V1.5 Internal [2.1.10]	Updated “user” object with new parameters. “groups” array allows to indicate the group which settings would be loaded to MedDream Viewer. “institution” object provides institution information needed for IHE_PDI study export. Max allowed items count parameter implemented.	Groups array and Institution object support; New parameter (token.items-max-count) for setting maximum number of items loaded in token-enabled integration support: starting from MedDream v.8.6.0 and token-service 2.1.10

CAUTION! MedDream automatically urldecodes the token string on reception but doesn't urlencode it when passing to the token validator. The best way is to use URL-safe values for the token, for example, encode binary content with base64url instead of base64.

NOTE! The Integrating IS can use other token generation and validation software as long as it supports the identical token validation interface and parameters that are used by MedDream.

Installation and configuration

MedDream TokenService can be installed on MedDream server machine without additional hardware or software upgrade.

If MedDream TokenService is installed on separate machine, the following minimum requirements should be maintained:

- Hardware: processor 2.33GHz or higher x64-compatible; hard drive 500 MB; memory 8 GB of RAM; network 100 Mbit/s.

- Operating systems: Windows Server 2019, 2022, 2025 and newer; Windows 10 (64 bit) and newer; Linux (64 bit).
- Supporting software: JAVA 17.

Deployment:

1. Install java 17.
2. Copy jar file to wanted directory.
3. If you want to change default properties add application.properties file in the same directory as the jar, and change properties in this file.
4. Open directory in command line and run the command `java -jar token-service.jar`.

NOTE! If you need to configure SSL encryption, see "SSL for the bundled Token Service" in Install Manual.

The MedDream TokenService configuration options are listed in the table.

Configuration item type
• **Item name: default value** **Short description**
Service configuration options
• com.softneta.token.cache.time-to-idle-seconds: 180 The maximum number of seconds the token can exist in the cache without being accessed.
• com.softneta.token.cache.clean-rate-milliseconds: 10000 Fixed period in milliseconds between cache clean invocations.
• com.softneta.token.one-time-token: false If true token can be fetched only once. NOTE: the options is not applied for tokens with LIVESHARE_GUEST permission. Use invalidate API to invalidate such tokens.
• security.user.name Manager user name. Manager can access management endpoints (url sample ‘http://IP:port/manage‘ for default management context path).
• security.user.password Manager password.
• security.generate.ip-white-list Comma separated IPs whose are granted to access token service generate endpoint, example: ‘127.0.0.1,0:0:0:0:0:0:0:1‘. Use a single asterisk, "*", to allow everyone.
• security.validate.ip-white-list Comma separated IPs whose are granted to access token service validate endpoint, example: ‘127.0.0.1,0:0:0:0:0:0:0:1‘. Use a single asterisk, "*", to allow everyone.
• management.context-path: /manage Context path for management endpoints.

• com.softneta.token.encryption.secret-key If empty then will be generated. Secret key must be 256 bits long. Must be provided if multiple instances are used in load balancing. Applicable starting from v1.1.
• com.softneta.token.encryption.initialization-vector If empty then will be generated. Initialization vector must be 16 bytes long. Must be provided if multiple instances are used in load balancing. Applicable starting from v1.1.
• validate.storage-configurations.parameter-list Comma separated list of valid parameter names, example: “dbUser“, “dbUserPassw”. • Applicable starting from v0.6, for API v2 and later.
• security.service.name Value used in basic authentication as username. • Applicable starting from v1.3.
• security.service.password Value used in basic authentication as password. Applicable starting from v1.3.
• token.items-max-count Value used to set maximum allowed items count in token-enabled integration. Default value is 50 Applicable starting from v1.5.
Logging configuration
• logging.path: Location of the log file, example ‘`/var/log`‘.
• logging.file: token_service.log Log file name.
• logging.level.*: ERROR Log levels severity mapping. For instance `logging.level.ROOT=DEBUG`
Embedded Server Configuration
• server.compression.enabled: true Whether response compression is enabled.
• server.compression.mime-types: application/json Comma-separated list of MIME types that should be compressed.
• server.port: 8080 Server HTTP port.
• server.context-path: / Context path of the application.
• server.session.timeout: 240

JAVA session timeout in seconds.
Authentication Authentication setup consist of the profile inclusion, which specifies the authentication you want to use. Either white-list, or basic, authentication profile must be included to start the service. Only one, either white-list, or basic, authentication profile may be used at a time. **CAUTION!** You should set basic authentication for generate and validate endpoints. the If basic authentication is used only for one endpoint, the other endpoint will be accessible without authentication.
• ip-whitelist-authentication IP white list authentication. Applicable starting from v1.3.
• basic-authentication-for-generation HTTP basic authentication for token generation. Applicable starting from v1.3.
• basic-authentication-for-validation HTTP basic authentication for token validation. Applicable starting from v1.3.

NOTE! API version specific TokenService configuration options are described in Annexes with description of particular API version.

Installation testing

The section provides the samples of the cURL requests with response body, that can be used to test the TokenService after installation:

- Generate token

```
curl --location --request POST 'IP:port/v1/generate' \
--header 'Content-Type: application/json' \
--data-raw '{
    "items": [
        {
            "studies": {
                "accnum": "test_number",
                "storage": "test_storage"
            }
        }
    ],
    "permissions": [
        "PATIENT_HISTORY", "SEARCH"
    ]
}
'
```

NOTE! Replace the '`IP:port`' string to the host IP and port, on which the TokenService is running.

Response body sample:

`f6wZ4bcNOYUPOotYp6hxQ9sE2QRbSncpNzN1_ovb0CCcprb3Ansd8RcRiJSZQqKRCaF9gKIoycQ3dtQTMPxHocriZnPDpvTvujHD3SyGyq8=`

NOTE! Response would be the randomly generated token and would not equal to the sample.

- Validate token

```
curl --location --
request GET '192.168.11.16:8085/v1/validate?token=f6wZ4bcNOYUPOotYp6hxQ9sE2QRbSncpNzN1_ovb0CCcprb3Ansd8RcRiJSZQqKRCaF9gKIoycQ3dtQTMPxHocriZnPDpvTvujHD3SyGyq8='
```

NOTE! Replace the token (string part in bold after the '=' sign) with the token, that was returned from the generate request.

Response body:

`{"items":[{"studies":{"accnum":"test_number","storage":"test_storage"}}],"permissions":["PATIENT_HISTORY","SEARCH"]}`

Generate token request

The token generation interface is implemented via HTTP POST request with token parameters passed in json body. The request and response samples are included in the table below. The structure of json body and token parameters varies depending on API version and are described in section Annexes.

Request and response samples
Request header sample: POST /v3/generate HTTP/1.1 Host: 127.0.0.1:8088 Content-Type: application/json
Request body sample 1: Opens two studies, identified by Study Instance UID, and stored in different PACS's (or other storages). The *"permissions": […]* array is not included into token, therefore the system will use the permissions values from application properties parameter authorization.defaultHisPermissions (see parameters in section **Related configuration options**). If the authorization.defaultHisPermissions is not configured, the default permissions, allowing to view the studies images, are used, and no of customizable MedDream functionality is allowed. *{* *"items": [* *{* *"studies": {* *"study": "1.2.840.113619.2.55.3.4271045733.996.1449464144.595",* *"storage": "Orthanc"* *}* *},* *{* *"studies": {* *"study": "1.2.826.0.1.3680043.8.1055.1.20160922221651432.55928341.45596087",* *"storage": "PacsOne"* *}*

```
      }
    ]
}
```

Request body sample 2:

Opens one (or several, if several studies have the same accession number) study, identified by accession number. Retrieving and viewing patient's historical studies is allowed by including *"permissions": ["PATIENT_HISTORY"]* in json body. List of history studies (*"history": […]*) is not included in json body. When PATIENT_HISTORY is granted and historical studies list is not provided, the default history retrieve algorithm will be used for each study (if more than one is related with provided accession number). The default history is collected by selecting the studies with the same patient ID from the same storage.

```
{
   "items": [
      {
         "studies": {
            "accnum": "20160602151858",
            "storage": "PacsOne"
         }
      }
   ],
   "permissions": [
      "PATIENT_HISTORY"
   ]
}
```

Request body sample 3:

Opens study, identified by accession number and patient ID pair. Retrieving and viewing patient's history studies is allowed by including *" permissions": ["PATIENT_HISTORY"]* in json body. List of history studies is collected from items in *"history": […]* array: list of studies with patient ID equal to '0', and selected from storages PacsOne and Orthanc.

```
{
    "items": [
        {
            "studies": {
                "accnum": "20160602151858",
                "patient": "0",
                "storage": "PacsOne"
            },
            "history": [
                {
                    "patient": "0",
                    "storage": "PacsOne"
                },
                {
                    "patient": "0",
                    "storage": "Orthanc"
                }
            ]
        }
    ],
    "permissions": [
        "PATIENT_HISTORY"
    ]
}
```

Request body sample 4:

Opens two studies, identified by Study Instance UID, and stored in different PACS's (or other storages). For all opened studies, retrieving and viewing patient's history studies is allowed by including *" permissions": ["PATIENT_HISTORY",…]* in json body. For the first item (*"study": "1.2.826.0.1.3680043.8.1055.1.20160922221651432.55928341.45596087"*), list of history studies is collected from studies in *"history": […]* array: list consists of two studies, including the study itself, identified by Study Instance UID. For the second item (*"study": "1.2.826.0.1.3680043.8.1055.1.20180719151246227.498555329.93002"*), historical studies list is not provided, and the default history retrieve algorithm will be used: list consists of the studies with the same patient ID that are selected from the same storage. For all opened studies, export to ISO archive and/or CD/DVD burn is allowed by including *" permissions": […,*

" EXPORT_ISO"] in json body.


```
{
   "items": [
      {
         "studies": {
            "study": "1.2.826.0.1.3680043.8.1055.1.20160922221651432.55928341.45596087",
            "storage": "PacsOne"
         },
         "history": [
                  {
                   "study": "1.2.826.0.1.3680043.8.1055.1.20131219214044458.87898881.58786776",
                    "storage": "PacsOne"
                  },
                  {
                   "study": "1.2.826.0.1.3680043.8.1055.1.20160922221651432.55928341.45596087",
                   "storage": "PacsOne"
         }]
      },
      {
         "studies": {
            "study": "1.2.826.0.1.3680043.8.1055.1.20180719151246227.498555329.93002",
            "storage": "Orthanc"
         }
      }
   ],
   "permissions": [
      "PATIENT_HISTORY", "EXPORT_ISO"
   ]
}
```


Request body sample 5:

Opens one (or more, if several studies have the same accession number) study ONLY if the study's patient ID equals to the value passed in restrictions array *"restrictions": {"patient": ["pt-014597"]}*. Retrieving and viewing patient's history studies is allowed by including *" permissions": ["PATIENT_HISTORY"]* . List of history studies is collected by searching the studies according the accession numbers in *"history": […]* array, but ONLY includes the studies of patients which patient IDs are listed in restrictions *"restrictions": {"patient": ["pt-014597"]}*.


```
{
   "items": [
      {
         "studies": {
            "accnum": "2016_000027",
            "storage": "PacsOne"
         },
         "history": [
                  {
                   "accnum": "2016_000095",
                    "storage": "PacsOne"
                  },
                  {
                   "accnum": "2013_131935",
                    "storage": "PacsOne"
         }]
      }
   ],
   "permissions": [
      "PATIENT_HISTORY"
   ] ,
    "restrictions": {
         "patient": ["pt-014597"]
    }
```

}

Request body sample 6:

Opens one (or more, if DICOM files of several studies are in folder ‘test_study_0’ or its sub-folder) study ONLY if the study’s patient ID equals to the value passed in restrictions array *"restrictions": {"patient": ["0"]}*. Retrieving and viewing patient’s history studies is allowed by including *" permissions": ["PATIENT_HISTORY"]* . FileSystem plugin doesn’t support the default history search, therefor the list of files or folder for collecting the patient’s history studies must be passed in *"history": […]* array.

```
{
   "items": [
      {
         "studies": {
            "file": "test_catalog/test_study_0",
            "storage": "FileSystem"
         },
         "history": [
                  {
                   "file": " test_catalog/other_sudy_0",
                    "storage": " FileSystem"
                  }]
      }
   ],
   "permissions": [
      "PATIENT_HISTORY"
   ] ,
    "restrictions": {
         "patient": ["0"]
    }
}
```

Response sample 1:

HTTP/1.1 200 OK
Date: Thu, 12 Aug 2021 19:48:08 GMT
Content-Length: 108
Content-Type: text/plain;charset=UTF-8

w10BqAQBWbtnfSInd2u5US7fFIZ8AeET_hVZbtaM4YdXj3aBQLrUpAzJ2NSurqg5OhuvkXKCRhuClSmE_JGt88jXYR33-z29ZAuMyZ-k3B8-hgVNf-A=

Response sample 2:

HTTP/1.1 400 Bad Request
Date: Thu, 12 Aug 2021 19:50:36 GMT
Content-Length: 38
Content-Type: text/plain;charset=UTF-8

Incorrect combination: patient + study

Response status codes:

- 200 OK: the token is returned as plain text.
- 400 Bad Request: the passed parameter or values does not correspond to requirements. The additional information is returned as plain text error.

⚠ **NOTE!** The token generation request cannot be sent directly from Integrating system browser page due to Cross-origin resource sharing (CORS) security. The request should be sent either from Integrating system server side, or the proxy to the token service should be configured.

Validate token request

The token validation interface is implemented via HTTP GET request with token passed as request parameter and the token parameters returned in json body. The request and response samples are included in the table below. The structure of returned json body and token parameters varies depending on API version and are described in section Annexes.

⚠ **NOTE!** It is required to use the same API version number for validate request, that was used for token generation.

Request and response samples

Request sample:

GET
/v3/validate?token=w10BqAQBWbtnfSInd2u5US7fFIZ8AeET_hVZbtaM4YdXj3aBQLrUpAzJ2NSurqg5OhuvkXKCRhuClSmE_JGt88jXYR33-z29ZAuMyZ-k3B8-hgVNf-A=
HTTP/1.1
Host: 127.0.0.1:8088

Response sample 1:

HTTP/1.1 200 OK
Date: Thu, 12 Aug 2021 19:54:12 GMT
Content-Type: application/json;charset=UTF-8


```
{
   "items": [
      {
         "studies": {
            "accnum": "20180816124124",
            "storage": "Orthanc"
         }
      }
   ],
   " permissions": [
      "PATIENT_HISTORY"
   ]
}
```


Response sample 2:

HTTP/1.1 404 Not Found
Date: Thu, 12 Aug 2021 19:59:01 GMT
Content-Length: 0

Response status codes:

- 200 OK: the token parameters are returned in json body.
- 404 Not Found: the passed token is invalid.

Invalidate token request

The token validation interface is implemented via HTTP DELETE request with token passed as request parameter and no content returned. The request and response samples are included in the table below. The structure of returned json body and token parameters varies depending on API version and are described in section Annexes.

NOTE! It is required to use the same API version number for invalidate request, that was used for token generation.

Request and response samples
Request sample: DELETE /v3/invalidate?token=w10BqAQBWbtnfSInd2u5US7fFlZ8AeET_hVZbtaM4YdXj3aBQLrUpAzJ2NSurqg5OhuvkXKCRhuClSmE_JGt88jXYR33-z29ZAuMyZ-k3B8-hgVNf-A= HTTP/1.1 Host: 127.0.0.1:8088
Response sample 1: *HTTP/1.1 204 No Content* *Date: Thu, 12 Aug 2021 19:55:43 GMT*
Response status codes: • 204 No Content: the token was deleted, if found.

Related configuration options

The section list MedDream configuration parameters that are related to MedDream URL integration. The detail MedDream configuration description should be checked in MedDream Servicing manual.

Configuration item type
• **Item name & values example** **Short description**
Application properties
• spring.profiles.include=auth-inmemory,auth-his Comma-separated list of supported authentication modes. Value 'auth-his' enables URL authentication, 'auth-inmemory' enables login authentication with internal MedDream users.
• authentication.his.token-service-address=http://127.0.0.1:8085/v1/validate IP and port of TokenService. Defining the parameter enables the token mode and disables the direct URL parameter passing.
• authentication.his.valid-his-params[0]=study • authentication.his.valid-his-params[1]=patient,accnum Supported URL parameter. Defining the parameter enables the direct URL parameter passing and disables the token mode. The allowed values: study; file; patient; accnum; patient,accnum; storage. The array index from 0 should be used to configure several parameters. **CAUTION!** If both authentication.his.token-service-address and authentication.his.valid-his-params is defined the system returns error.
• authentication.his.allowOnePatientViewOnly=true If set to true, enables restriction, that only one patient medical data may be retrieved and viewed per URL session. Default value is false.
• authentication.his.maxObjects=0 Allowed number of studies for HIS session at authentication. Starting from MedDream v7.6, parameter is not used by default (default value is 0).
• authentication.his.useSameSession=false If set to true, the studies of executed URL are added to existing URL session instead of creating a new session. Default value is false. **NOTE!** The parameter is applicable only for token disabled integration. In token enabled integration the studies are always added to existing URL session.
• authentication.his.patient-integration-open-studies=false If set to true, the modal dialog with patient studies list is opened instead of opening the studies directly in the viewer for the URL with patient parameter or token, that contains one patient UID. Set it to false, if such behavior is not needed. If not set in application properties, the default value is true.
• security.frameOptionsPolicy: SAMEORIGIN Configures the X-Frame-Options HTTP response header. Valid values: NONE,DENY,SAMEORIGIN. When NONE then header is not set. DENY disallows IFRAME embedding completely. SAMEORIGIN is the default and allows embedding MedDream only on the server that hosts it.
• authorization.defaultHisPermissions=PATIENT_HISTORY,UPLOAD_DICOM_LIBRARY,ADMIN Comma separated list of MedDream operations, that are granted for token disabled URL session. The list of supported

operations: EXPORT_ISO, EXPORT_ARCH, FORWARD, REPORT_VIEW, REPORT_UPLOAD, PATIENT_HISTORY, UPLOAD_DICOM_LIBRARY, 3D_RENDERING, ADMIN, ANONYMOUS_VIEW, DOCUMENT_VIEW, BOUNDING_BOX_VIEW, BOUNDING_BOX_EDIT, FREE_DRAW_VIEW, FREE_DRAW_EDIT, LIVESHARE_GUEST. SMART_DRAW_VIEW, SMART_DRAW_EDIT, COPY_TO_DICOM values added to the allowed values in v0.8. USER_SETTINGS, CLEAR_CACHE values added to the allowed values in v1.0. PACSONE_VIEW_ONLY_PUBLIC, SHORTCUTS_EDIT, and HANGING_PROTOCOLS_EDIT value added to allowed values in v1.1.

⚠ **NOTE!** See detail description of MedDream's built-in authentication and authorization in the Install manual of the used MedDream version.

- pacs.configurations[x].strictSearchIsEnabled=true

Enable or disable strict search based on patient ID, accession number and other attributes.
Using *null* (property is missing or commented) is recommended for HIS integrations and activates the following custom search behavior for URL session:
- exact matching is implemented when searching according the URL (or token) parameters;
- non-exact matching is implemented when searching according user-selected parameters in SEARCH window (if enabled).

⚠ **CAUTION!** Check the service manual if the custom search behavior is implemented for the PACS system that you are using.

- authentication.his.jwtServiceAddress=http://127.0.0.1:8085/validate

IP and port of jwt validation service. Defining the parameter enables the jwt mode and allows jwt URL parameter passing.

- authentication.his.jwtServerKey=s/c7157H54vNu0s7yppIPbgvQvVDHQ6sSnf4qF5T9WmI5f9ne2cUl29Sdro9iyDfpU40oQqOm/D/o/hKKRA0ig==

Base64 secret key to verify jwt signature. Signature is not verified, if parameter is not set. The signature will be verified only if authentication.his.jwtMedDreamKey= is set.

- authentication.his.jwtMedDreamKey= DjSWFtWmpv0g56OjI4xWVVOKve5hN2PNpBDisnU5QTIw94TINFaqbkG4htn0/tpQ7W4/bu9ctEEGWO66QpsVzg==

Base64 secret key used for resign jwt, before sending it to jwt validation service. Not signed token is sent, if parameter is not set. It will work only if authentication.his.jwtServerKey is set.

- authentication.his.responseDecryptPassword=g9RxMRzhXfF/GG4HeV4A/w+M9zadeYVP+265oan67Z4=
- authentication.his.responseDecryptSalt=/9kzgAP3kcmXnhluapYdnw==

Base64 encoded password and salt for response from jwt validation service decryption. Not encrypted parameters json should be returned, if parameters are not set.

System properties

- features.patientHistory=false

If set to true, allows access to patient history functionality. Default value is false.

- features.export=true

If set to true, allows access to studies export to .burn/.iso archive functionality. Default value is true.

- features.archive=true

If set to true, allows access to studies export to .zip archive functionality. Default value is true.

- features.search=true

If set to true, allows access to search window and search functionality. Default value is false.

Table of Figures

Figure 1. MedDream Viewer and MedDream TokenService integration architecture 9
Figure 2. MedDream Viewer and MedDream TokenService integration components 9
Figure 3. One step token enabled integration scenario 10
Figure 4. Two steps token enabled integration scenario 12
Figure 5. MedDream Viewer and Integrating IS with internal token service implementation integration components 14
Figure 6. Token enabled integration with MedDream type token service implementation in Integrating IS scenario 15
Figure 7. Token enabled integration with JSON Web Token service implementation in Integrating IS scenario 17
Figure 8. Token disabled integration scenario 21

Annex I. json structure and parameters description for API v1

Annex describes the json body structure, parameters and related TokenService configuration options for version v1.

.json structure example

```
{
    "items": [
        {
            "studies": {
                "accnum": "acc2",
                "patient": "pat2",
                "study": "stu1",
                "storage": "s4"
            },
            "history": [
                {
                "accnum": "acc2",
                "patient": "pat2",
                "study": "stu1",
                "storage": "s4"
                }
            ]
        }
    ],
    "permissions": [
        "DOCUMENT_VIEW",
        " ADMIN"
    ],
    "restrictions": {
        "patient": ["pat2"]
    }
}
```

Parameters description

The table below provides the structure of json body and explains the parameters: name, description and validation rules.

Parameters description	**Structure of json body**
	{
Required. Empty array not allowed. Maximums 50 items are allowed. Array of items, that should be allowed to access in MedDream search or viewer windows. Each item contains the required study or studies identifiers, and optional historical studies array for the item.	*"items":*
	[*{*
Required. Empty object not allowed. Identifiers for retrieving the study or studies.	*"studies":*
	{
Required study or studies identifier. Allowed one of listed: • "study" – one study, identified by Study Instance UID; • "patient" – one or more studies for the patient, identified by patient ID; • "accnum" – one or more studies, identified by accession number; • "accnum" , "patient" – one or more studies, identified by accession number AND patient ID;	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"file": "path1",*

• “file” – path to studies DICOM file or folder with studies DICOM files; • "patient" , “studyDate” – one or more studies, identified by patient ID AND study date, parameters pair added in v1.2. Empty value is not allowed. If “file” identifier is used, the other identifiers are not allowed to be used in the same token.	
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s4"*
	},
Optional. Empty array not allowed. Array of historical studies. Each array object contains the required study or studies identifiers and storage identifier. ⚠ **NOTE!** To enable access to historical studies, you must include *permissions=["PATIENT_HISTORY"]*. ⚠ **NOTE!** List of history studies is collected from studies, that are retrieved according the identifiers in *"history": […]* array, and adding the study or studies themselves, if not present. ⚠ **NOTE!** The same historical studies list is assigned to each study, if more than one retrieved according the identifiers in *studies: {…}* object, and to study from the historical studies list, if such study is not included in any in *studies: {…}* object and hasn’t historical studies list assigned. ⚠ **CAUTION!** If *permissions=["PATIENT_HISTORY"]* is set, and *"history": […]* array is not provided, the default historical studies collection is used: the studies with the same patient ID that are selected from the same storage are collected as historical studies. If you want to prevent such behavior, include into *"history": […]* array object equal to *studies: {…}* object, and you will have historical studies list including only the study or studies themselves. The default historical studies selection is not supported with FileSystem plugin and “file” identifier.	*"history":*
	[*{*
Required. Empty value is not allowed. See parameters description and validation rules above, in *studies{…}* object.	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"storage": "s4"*
	} *]* *}* *],*
Optional. Empty array not allowed. Customizable MedDream functionality, that is granted for the URL request using the token. List of allowed values: EXPORT_ISO, EXPORT_ARCH, FORWARD, REPORT_VIEW, REPORT_UPLOAD, PATIENT_HISTORY, UPLOAD_DICOM_LIBRARY, 3D_RENDERING, ADMIN, ANONYMOUS_VIEW, DOCUMENT_VIEW. SMART_DRAW_VIEW, SMART_DRAW_EDIT, COPY_TO_DICOM values added to the allowed values in v0.8. USER_SETTINGS, CLEAR_CACHE values added to the allowed values in v1.0.	*" permissions":* *[* *"PATIENT_HISTORY",* *" ADMIN"* *],*

PACSONE_VIEW_ONLY_PUBLIC, SHORTCUTS_EDIT, and HANGING_PROTOCOLS_EDIT value added to allowed values in v1.1. If not included, the system uses the permissions from application properties parameter authorization.defaultHisPermissions. ⚠ **NOTE!** See detail description of MedDream's built-in authentication and authorization in the Install manual of the used MedDream version. ⚠ **NOTE!** The access to customizable MedDream functionality may also be restricted by license, system properties (system.json), and settings (global.json). Please note, that access to functionality is granted only if no other source (license, system properties, or settings) restricts it.	
Optional. Empty object not allowed. Data access restrictions, that are applied for the URL request using the token.	*"restrictions":*
	{
Optional. Empty array not allowed. Empty object in the array not allowed, validation added in v0.9. Array of patient IDs. Patients, which studies is allowed to be retrieved and viewed, should be listed. Used on patient portals to provide access to only to the medical data of authenticated patient. ⚠ **NOTE!** The patient array with only one value is required, if system property allowOnePatientViewOnly is set to true.	*"patient": ["pat2"]*
	} *}*

Version related configuration

TokenService does not have version v1 specific configuration options.

Annex II. json structure and parameters description for API v2

Annex describes the json body structure, parameters and related TokenService configuration options for version v2.

.json structure example

```
{
  "items": [
    {
      "studies": {
        "accnum": "acc1",
        "patient": null,
        "study": null,
        "storage": "storage1"
      },
      "history": [
        {
          "accnum": "acc1",
          "patient": null,
          "study": null,
          "storage": "storage1"
        }
      ]
    }
  ],
  "permissions": [
    "PATIENT_HISTORY"
  ],
  "restrictions": {
    "patient": [
      "p001"
    ]
  },
  "user": {
    "id": "123",
    "name": "user name"
  },
  "storageConfiguration": [
    {
      "storage": "s1",
      "parameters": [
        {
          "name": "parameter name",
          "value": "parameter value"
        }
      ]
    }
  ]
}
```

Parameters description

The table below provides the structure of json body and explains the parameters: name, description and validation rules.

Parameters description	**Structure of json body**
	{
Required. Empty array not allowed. Maximums 50 items are allowed. Array of items, that should be allowed to access in MedDream search or viewer windows. Each item contains the required study or studies identifiers, and optional	*"items":*

historical studies array for the item.	
	[*{*
Required. Empty object not allowed. Identifiers for retrieving the study or studies.	*"studies":*
	{
Required study or studies identifier. Allowed one of listed: • "study" – one study, identified by Study Instance UID; • "patient" – one or more studies for the patient, identified by patient ID; • "accnum" – one or more studies, identified by accession number; • "accnum" , "patient" – one or more studies, identified by accession number AND patient ID; • “file” – path to studies DICOM file or folder with studies DICOM files; • "patient" , “studyDate” – one or more studies, identified by patient ID AND study date, parameters pair added in v1.2. Empty value is not allowed. If “file” identifier is used, the other identifiers are not allowed to be used in the same token.	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"file": "path1",*
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s4"*
	},
Optional. Empty array not allowed. Array of historical studies. Each array object contains the required study or studies identifiers and storage identifier. ⚠ **NOTE!** To enable access to historical studies, you must include *permissions=["PATIENT_HISTORY"]*. ⚠ **NOTE!** List of history studies is collected from studies, that are retrieved according the identifiers in *"history": […]* array, and adding the study or studies themselves, if not present. ⚠ **NOTE!** The same historical studies list is assigned to each study, if more than one retrieved according the identifiers in *studies: {…}* object, and to study from the historical studies list, if such study is not included in any in *studies: {…}* object and hasn’t historical studies list assigned. ⚠ **CAUTION!** If *permissions=["PATIENT_HISTORY"]* is set, and *"history": […]* array is not provided, the default historical studies collection is used: the studies with the same patient ID that are selected from the same storage are collected as historical studies. If you want to prevent such behavior, include into *"history": […]* array object equal to *studies: {…}* object, and you will have historical studies list including only the study or studies themselves. The default historical studies selection is not supported with FileSystem plugin and “file” identifier.	*"history":*
	[*{*
Required. Empty value is not allowed. See parameters description and validation rules above, in *studies{…}* object.	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",*

	"storage": "s4"
	} *]* *}* *],*
Optional. Empty array not allowed. Customizable MedDream functionality, that is granted for the URL request using the token. List of allowed values: EXPORT_ISO, EXPORT_ARCH, FORWARD, REPORT_VIEW, REPORT_UPLOAD, PATIENT_HISTORY, UPLOAD_DICOM_LIBRARY, 3D_RENDERING, ADMIN, ANONYMOUS_VIEW, DOCUMENT_VIEW. SMART_DRAW_VIEW, SMART_DRAW_EDIT, COPY_TO_DICOM values added to the allowed values in v0.8. USER_SETTINGS, CLEAR_CACHE values added to the allowed values in v1.0. PACSONE_VIEW_ONLY_PUBLIC, SHORTCUTS_EDIT, and HANGING_PROTOCOLS_EDIT value added to allowed values in v1.1. If not included, the system uses the permissions from application properties parameter authorization.defaultHisPermissions. ⚠ **NOTE!** See detail description of MedDream's built-in authentication and authorization in the Install manual of the used MedDream version. ⚠ **NOTE!** The access to customizable MedDream functionality may also be restricted by license, system properties (system.json), and settings (global.json). Please note, that access to functionality is granted only if no other source (license, system properties, or settings) restricts it.	*" permissions":* *[* *"PATIENT_HISTORY",* *" ADMIN"* *],*
Optional. Empty object not allowed. Data access restrictions, that are applied for the URL request using the token.	*"restrictions":*
	{
Optional. Empty array not allowed. Empty object in the array not allowed, validation added in v0.9. Array of patient IDs. Patients, which studies is allowed to be retrieved and viewed, should be listed. Used on patient portals to provide access to only to the medical data of authenticated patient. ⚠ **NOTE!** The patient array with only one value is required, if system property allowOnePatientViewOnly is set to true.	*"patient": ["pat2"]*
	},
Optional. Empty object not allowed. Information about the user, who connects to the system with the token.	*"user":*
	{
Optional. Empty value is not allowed. Key for user identification in HIS and MedDream systems.	*"id": "UserID",*
Optional. Empty value is not allowed. User name, middle name, surname, used for presentation purposes.	*"name": "User name"*
	},
Optional. Empty array not allowed.	*"storageConfiguration":*

Information for granting and configuring user's access to data.	
	[*{*
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s1",*
Required. Empty array is not allowed. Parameter for connecting to the PACS data storage and data access configuration.	*"parameters":* *[*
Parameter name and value pair. Both attributes required, and empty values are not allowed. Parameter name should be validated according the TokenService configuration. See the description of supported parameters per particular plugin in the Install manual of the used MedDream version.	*{* *"name": "Name",* *"value": "Value"* *}*
	] *}* *]*
	}

Version related configuration

The MedDream TokenService configuration options, that are related to version v2 parameters, are listed in the table.

Configuration item type
• **Item name: default value** **Short description**
• validate.storage-configurations.parameter-list: Comma separated list of valid parameter names, example: “dbUser“, “dbUserPassw”.

API v2 differences from API v1

The changes in v2, comparing to v1:

- “user” object added in json body;
- "storageConfiguration" object (array) added in json body;
- validate.storage-configurations.parameter-list parameter added to TokenService configuration (application.yml).

Annex III. json structure and parameters description for API v3

Annex describes the json body structure, parameters and related TokenService configuration options for version v3.

.json structure example


```
{
  "items": [
    {
      "studies": {
        "accnum": "acc1",
        "patient": null,
        "study": null,
        "storage": "storage1"
      },
      "history": [
        {
          "accnum": "acc1",
          "patient": null,
          "study": null,
          "storage": "storage1"
        }
      ]
    }
  ],
  "permissions": [
    "PATIENT_HISTORY"
  ],
  "restrictions": {
    "patient": [
      "p001"
    ]
  },
  "user": {
    "id": "123",
    "name": "user name"
  },
  "storageConfiguration": [
    {
      "storage": "s1",
      "parameters": [
        {
          "name": "parameter name",
          "value": "parameter value"
        }
      ]
    }
  ],
  "segmentation": {
    "segments": [
      {
        "instance": "inst1",
        "storage": "s1"
      }
    ]
  }
}
```

Parameters description

The table below provides the structure of json body and explains the parameters: name, description and validation rules.

Parameters description	**Structure of json body**
	{
Required. Empty array not allowed. Maximums 50 items are allowed. Array of items, that should be allowed to access in MedDream search or viewer windows. Each item contains the required study or studies identifiers, and optional historical studies array for the item.	*"items":*
	[*{*
Required. Empty object not allowed. Identifiers for retrieving the study or studies.	*"studies":*
	{
Required study or studies identifier. Allowed one of listed: • "study" – one study, identified by Study Instance UID; • "patient" – one or more studies for the patient, identified by patient ID; • "accnum" – one or more studies, identified by accession number; • "accnum" , "patient" – one or more studies, identified by accession number AND patient ID; • “file” – path to studies DICOM file or folder with studies DICOM files; • "patient" , “studyDate” – one or more studies, identified by patient ID AND study date, parameters pair added in v1.2. Empty value is not allowed. If “file” identifier is used, the other identifiers are not allowed to be used in the same token.	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"file": "path1",*
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s4"*
	},
Optional. Empty array not allowed. Array of historical studies. Each array object contains the required study or studies identifiers and storage identifier. ⚠ **NOTE!** To enable access to historical studies, you must include *permissions=["PATIENT_HISTORY"]*. ⚠ **NOTE!** List of history studies is collected from studies, that are retrieved according the identifiers in *"history": […]* array, and adding the study or studies themselves, if not present. ⚠ **NOTE!** The same historical studies list is assigned to each study, if more than one retrieved according the identifiers in *studies: {…}* object, and to study from the historical studies list, if such study is not included in any in *studies: {…}* object and hasn’t historical studies list assigned. ⚠ **CAUTION!** If *permissions=["PATIENT_HISTORY"]* is set, and *"history": […]* array is not provided, the default historical studies collection is used: the studies with the same patient ID that are selected from the same storage are collected as historical	*"history":*

studies. If you want to prevent such behavior, include into *"history": […]* array object equal to *studies: {…}* object, and you will have historical studies list including only the study or studies themselves. The default historical studies selection is not supported with FileSystem plugin and “file” identifier.	
	[*{*
Required. Empty value is not allowed. See parameters description and validation rules above, in *studies{…}* object.	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"storage": "s4"*
	} *]* *}* *],*
Optional. Empty array not allowed. Customizable MedDream functionality, that is granted for the URL request using the token. List of allowed values: EXPORT_ISO, EXPORT_ARCH, FORWARD, REPORT_VIEW, REPORT_UPLOAD, PATIENT_HISTORY, UPLOAD_DICOM_LIBRARY, 3D_RENDERING, ADMIN, ANONYMOUS_VIEW, DOCUMENT_VIEW, BOUNDING_BOX_VIEW, BOUNDING_BOX_EDIT, FREE_DRAW_VIEW, FREE_DRAW_EDIT, LIVESHARE_GUEST. SMART_DRAW_VIEW, SMART_DRAW_EDIT, COPY_TO_DICOM values added to the allowed values in v0.8. USER_SETTINGS, CLEAR_CACHE values added to the allowed values in v1.0. PACSONE_VIEW_ONLY_PUBLIC, SHORTCUTS_EDIT, and HANGING_PROTOCOLS_EDIT value added to allowed values in v1.1. If not included, the system uses the permissions from application properties parameter authorization.defaultHisPermissions. ⚠ **NOTE!** See detail description of MedDream's built-in authentication and authorization in the Install manual of the used MedDream version. ⚠ **NOTE!** The access to customizable MedDream functionality may also be restricted by license, system properties (system.json), and settings (global.json). Please note, that access to functionality is granted only if no other source (license, system properties, or settings) restricts it.	*" permissions":* *[* *"PATIENT_HISTORY",* *" ADMIN"* *],*
Optional. Empty object not allowed. Data access restrictions, that are applied for the URL request using the token.	*"restrictions":*
	{
Optional. Empty array not allowed. Empty object in the array not allowed, validation added in v0.9. Array of patient IDs. Patients, which studies is allowed to be retrieved and viewed, should be listed. Used on patient portals to provide access to only to the medical data of authenticated patient. ⚠ **NOTE!** The patient array with only one value is required, if system property allowOnePatientViewOnly is set to true.	*"patient": ["pat2"]*
	},
Optional. Empty object not allowed. Information about the user, who connects to the system with the token.	*"user":*

	{
Optional. Empty value is not allowed. Key for user identification in HIS and MedDream systems.	*"id": "UserID",*
Optional. Empty value is not allowed. User name, middle name, surname, used for presentation purposes.	*"name": "User name"*
	},
Optional. Empty array not allowed. Information for granting and configuring user's access to data.	*"storageConfiguration":*
	[*{*
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s1",*
Required. Empty array is not allowed. Parameter for connecting to the PACS data storage and data access configuration.	*"parameters":* *[*
Parameter name and value pair. Both attributes required, and empty values are not allowed. Parameter name should be validated according the TokenService configuration. See the description of supported parameters per particular plugin in the Install manual of the used MedDream version.	*{* *"name": "Name",* *"value": "Value"* *}*
	] *}* *],*
Optional. Empty object not allowed. Segmentation objects to be viewed or edited by the user.	*"segmentation":* *{*
Optional. Empty array allowed. Array with list of accessible RTSTRUCT instances, that contains saved segments to be viewed or edited by the user. ⚠ **NOTE!** To enable segmentation edit rights, you must include any of *permissions=["BOUNDING_BOX_EDIT", "FREE_DRAW_EDIT"]*. To enable segmentation view rights, you must include any of *permissions=["BOUNDING_BOX_VIEW", "FREE_DRAW_VIEW"]*. If edit permission, for example *"BOUNDING_BOX_EDIT",* is granted, the view permission *"BOUNDING_BOX_VIEW"* is granted automatically for the accessible RTSTRUCT instances. ⚠ **NOTE!** If edit permission, for example *permissions=["BOUNDING_BOX_EDIT"]*, is set, and empty *"segments":[]* array is passed, the created segmentations will be stored in a new RTSTRUCT instance in a new series. ⚠ **NOTE!** If any of view or edit permission is granted, and *"segments"* array is not passed (NULL), the default segmentations retrieve and/or saving procedure is applied: the newest RTSTRUCT version from all the RT series is returned as accessible for the URL session.	*"segments":[*
Required RTSTRUCT instance identifier and storage identifier: • "instance" – RTSTRUCT SOP instance UID; • "storage" – Unique source VNA/PACS identifier. Value should be identical as	*{* *"instance": "inst1",* *"storage": "s4"* *}*

configured in MedDream application properties. Empty values are not allowed.	
	] }
	}

Version related configuration

TokenService does not have version v3 specific configuration options.

API v3 differences from API v2

The changes in v3, comparing to v2:

- “segmentation” object added in json body;
- BOUNDING_BOX_VIEW, BOUNDING_BOX_EDIT, FREE_DRAW_VIEW, FREE_DRAW_EDIT, LIVESHARE_GUEST added to allowed values of “permissions” array;
- Custom handling implementation for LiveShare guest session tokens added: one-time-token=true configuration option is ignored for tokens with LIVESHARE_GUEST permission.
- DELETE /v3/invalidate API for token invalidation added.

Annex IV. Changes in “permissions”: […] allowed values in v0.8

Annex describes the changes in json parameter “permissions”: […] allowed values implemented in TokenService v0.8. The changes apply for TokenService API versions v1, v2, and v3. The complete description and examples of corresponding TokenService API versions see in Annexes I, II, and III.

“permissions”: […] allowed values in v0.8 differences from v0.7

The changes in “permissions”: […] allowed values in v0.8:

- SMART_DRAW_VIEW, SMART_DRAW_EDIT, COPY_TO_DICOM added to allowed values of “permissions” array.

Annex V. Changes in “patient”: […] values validation in v0.9

Annex describes the changes in json parameter “patient”: […]values validation implemented in TokenService v0.9. The changes apply for TokenService API versions v1, v2, and v3. The complete description and examples of corresponding TokenService API versions see in Annexes I, II, and III.

“patient”: […] parameters validation in v0.9 differences from v0.8

The changes of parameters validation in v0.9:

- Empty object is not allowed in array “patient”: […].

Annex VI. Changes in “permissions”: […] allowed values in v1.0

Annex describes the changes in json parameter “permissions”: […] allowed values implemented in TokenService v1.0. The changes apply for TokenService API versions v1, v2, and v3. The complete description and examples of corresponding TokenService API versions see in Annexes I, II, and III.

“permissions”: […] allowed values in v1.0 differences from v0.9

The changes in “permissions”: […] allowed values in v1.0:

- USER_SETTINGS, CLEAR_CACHE added to allowed values of “permissions” array.

Annex VII. Changes in “permissions”: […] allowed values in v1.1

Annex describes the changes in json parameter “permissions”: […] allowed values implemented in TokenService v1.1. The changes apply for TokenService API versions v1, v2, and v3. The complete description and examples of corresponding TokenService API versions see in Annexes I, II, and III.

“permissions”: […] allowed values in v1.1 differences from v1.0

The changes in “permissions”: […] allowed values in v1.1:

- PACSONE_VIEW_ONLY_PUBLIC added to allowed values of “permissions” array;
- SHORTCUTS_EDIT added to allowed values of “permissions” array;
- HANGING_PROTOCOLS_EDIT added to allowed values of “permissions” array.

Annex VIII. json structure and parameters description for API v4

Annex describes the changes in json "studies" object and "history" array object: new allowed study or studies identifiers values implemented in TokenService v1.2. The changes apply for TokenService API versions v1, v2, v3, andv4. The complete description and examples of corresponding TokenService API versions see in Annexes I, II, III, and VIII (this Annex).

Annex describes json body structure, parameters and related TokenService configuration options for version v4.

"studies" object and "history" array object: allowed identifiers in v1.2 differences from v1.1

The changes in "studies" object and "history" array object allowed identifiers in v1.2:

- "patient", “studyDate” – identifiers pair added to allowed study or studies identifiers.

.json structure example


```
{
  "items": [
    {
      "studies": {
        "accnum": "acc1",
        "patient": null,
        "study": null,
        "storage": "storage1"
      },
      "history": [
        {
          "accnum": "acc1",
          "patient": null,
          "study": null,
          "storage": "storage1"
        }
      ]
    }
  ],
  "permissions": [
    "PATIENT_HISTORY"
  ],
  "restrictions": {
    "patient": [
      "p001"
    ]
  },
  "user": {
    "id": "123",
    "name": "user name"
  },
  "storageConfiguration": [
    {
      "storage": "s1",
      "parameters": [
        {
          "name": "parameter name",
          "value": "parameter value"
        }
      ]
    }
  ],
  "segmentation": {
    "segments": [
      {
```

```
      "instance": "inst1",
      "storage": "s1"
    }
  ]
 },
 "pluginConfigurations": [
  {
    "pluginName": "name",
    "parameters": [
      {
        "name": "parameter name",
        "value": "parameter value"
      }
    ]
  }
 ]
}
```

Parameters description

The table below provides the structure of json body and explains the parameters: name, description and validation rules.

Parameters description	**Structure of json body**
	{
Required. Empty array not allowed. Maximums 50 items are allowed. Array of items, that should be allowed to access in MedDream search or viewer windows. Each item contains the required study or studies identifiers, and optional historical studies array for the item.	*"items":*
	[*{*
Required. Empty object not allowed. Identifiers for retrieving the study or studies.	*"studies":*
	{
Required study or studies identifier. Allowed one of listed: • "study" – one study, identified by Study Instance UID; • "patient" – one or more studies for the patient, identified by patient ID; • "accnum" – one or more studies, identified by accession number; • "accnum" , "patient" – one or more studies, identified by accession number AND patient ID; • “file” – path to studies DICOM file or folder with studies DICOM files; • "patient" , “studyDate” – one or more studies, identified by patient ID AND study date, parameters pair added in v1.2. Empty value is not allowed. If “file” identifier is used, the other identifiers are not allowed to be used in the same token.	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"file": "path1",*
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s4"*
	},

Optional. Empty array not allowed. Array of historical studies. Each array object contains the required study or studies identifiers and storage identifier. ⚠ **NOTE!** To enable access to historical studies, you must include *permissions=["PATIENT_HISTORY"]*. ⚠ **NOTE!** List of history studies is collected from studies, that are retrieved according the identifiers in *"history": […]* array, and adding the study or studies themselves, if not present. ⚠ **NOTE!** The same historical studies list is assigned to each study, if more than one retrieved according the identifiers in *studies: {…}* object, and to study from the historical studies list, if such study is not included in any in *studies: {…}* object and hasn't historical studies list assigned. ⚠ **CAUTION!** If *permissions=["PATIENT_HISTORY"]* is set, and *"history": […]* array is not provided, the default historical studies collection is used: the studies with the same patient ID that are selected from the same storage are collected as historical studies. If you want to prevent such behavior, include into *"history": […]* array object equal to *studies: {…}* object, and you will have historical studies list including only the study or studies themselves. The default historical studies selection is not supported with FileSystem plugin and "file" identifier.	*"history":*
	[*{*
Required. Empty value is not allowed. See parameters description and validation rules above, in *studies{…}* object.	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"storage": "s4"*
	} *]* *}* *],*
Optional. Empty array not allowed. Customizable MedDream functionality, that is granted for the URL request using the token. List of allowed values: EXPORT_ISO, EXPORT_ARCH, FORWARD, REPORT_VIEW, REPORT_UPLOAD, PATIENT_HISTORY, UPLOAD_DICOM_LIBRARY, 3D_RENDERING, ADMIN, ANONYMOUS_VIEW, DOCUMENT_VIEW, BOUNDING_BOX_VIEW, BOUNDING_BOX_EDIT, FREE_DRAW_VIEW, FREE_DRAW_EDIT, LIVESHARE_GUEST. SMART_DRAW_VIEW, SMART_DRAW_EDIT, COPY_TO_DICOM values added to the allowed values in v0.8. USER_SETTINGS, CLEAR_CACHE values added to the allowed values in v1.0. PACSONE_VIEW_ONLY_PUBLIC, SHORTCUTS_EDIT, and HANGING_PROTOCOLS_EDIT value added to allowed values in v1.1. If not included, the system uses the permissions from application properties parameter authorization.defaultHisPermissions. ⚠ **NOTE!** See detail description of MedDream's built-in authentication and authorization in the Install manual of the used MedDream version. ⚠ **NOTE!** The access to customizable MedDream functionality may also be restricted by license, system properties (system.json), and settings (global.json). Please note, that access to functionality is granted only if no other source (license,	*" permissions":* *[* *"PATIENT_HISTORY",* *" ADMIN"* *],*

system properties, or settings) restricts it.	
Optional. Empty object not allowed. Data access restrictions, that are applied for the URL request using the token.	*"restrictions":*
	{
Optional. Empty array not allowed. Empty object in the array not allowed, validation added in v0.9. Array of patient IDs. Patients, which studies is allowed to be retrieved and viewed, should be listed. Used on patient portals to provide access to only to the medical data of authenticated patient. ⚠ **NOTE!** The patient array with only one value is required, if system property allowOnePatientViewOnly is set to true.	*"patient": ["pat2"]*
	},
Optional. Empty object not allowed. Information about the user, who connects to the system with the token.	*"user":*
	{
Optional. Empty value is not allowed. Key for user identification in HIS and MedDream systems.	*"id": "UserID",*
Optional. Empty value is not allowed. User name, middle name, surname, used for presentation purposes.	*"name": "User name"*
	},
Optional. Empty array not allowed. Information for granting and configuring user's access to data.	*"storageConfiguration":*
	[*{*
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s1",*
Required. Empty array is not allowed. Parameter for connecting to the PACS data storage and data access configuration.	*"parameters":* *[*
Parameter name and value pair. Both attributes required, and empty values are not allowed. Parameter name should be validated according the TokenService configuration. See the description of supported parameters per particular plugin in the Install manual of the used MedDream version.	*{* *"name": "Name",* *"value": "Value"* *}*
	] *}* *],*
Optional. Empty object not allowed. Segmentation objects to be viewed or edited by the user.	*"segmentation":* *{*
Optional. Empty array allowed. Array with list of accessible RTSTRUCT instances, that contains saved segments to be viewed or edited by the user.	*"segments":[*

⚠ **NOTE!** To enable segmentation edit rights, you must include any of *permissions=["BOUNDING_BOX_EDIT", "FREE_DRAW_EDIT"]*. To enable segmentation view rights, you must include any of *permissions=["BOUNDING_BOX_VIEW", "FREE_DRAW_VIEW"]*. If edit permission, for example *"BOUNDING_BOX_EDIT",* is granted, the view permission *"BOUNDING_BOX_VIEW"* is granted automatically for the accessible RTSTRUCT instances. ⚠ **NOTE!** If edit permission, for example *permissions=["BOUNDING_BOX_EDIT"]*, is set, and empty *"segments":[]* array is passed, the created segmentations will be stored in a new RTSTRUCT instance in a new series. ⚠ **NOTE!** If any of view or edit permission is granted, and *"segments"* array is not passed (NULL), the default segmentations retrieve and/or saving procedure is applied: the newest RTSTRUCT version from all the RT series is returned as accessible for the URL session.	
Required RTSTRUCT instance identifier and storage identifier: • "instance" – RTSTRUCT SOP instance UID; • "storage" – Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties. Empty values are not allowed.	*{* *"instance": "inst1",* *"storage": "s4"* *}*
	] *},*
Optional. Empty array not allowed. Information for granting and configuring Viewer plugins.	*"pluginConfigurations":*
	[*{*
Required. Empty value is not allowed. Unique plugin name. Value should be identical as in MedDream Viewer plugins configuration.	*"pluginName": "name",*
Required. Empty array is not allowed. Viewer plugin configuration parameters.	*"parameters":* *[*
Parameter name and value pair. Both attributes required, and empty values are not allowed. Parameter name should be allowed in Viewer plugin configuration and is validated by MedDream application. See the description of supported Viewer plugin parameters in the Install manual of the used MedDream version.	*{* *"name": "Name",* *"value": "Value"* *}*
	] *}* *],*
	}

Version related configuration

TokenService does not have version v4 specific configuration options.

API v4 differences from API v3

The changes in v4, comparing to v3:

- “pluginConfigurations” object added in json body.

Annex IX. Changes in authentication in v1.3

Annex describes the changes in TokenService authentication and related configuration parameters implemented in v1.3. The complete parameters description see in section “Installation and configuration”.

Version related configuration

The changes in authentication and authentication configuration in v1.3:

- Basic authentication for generate and validate endpoints implemented;
- Authentication profiles ip-whitelist-authentication, basic-authentication-for-generation, and basic-authentication-for-validation configuration added;
- Parameters security.service.name and security.service.password parameters added to configuration for basic authentication.

Annex X. json structure and parameters description for API v4

Annex describes the changes in json "restrictions" array object: new allowed study or studies identifiers values implemented in TokenService v1.4. Available from MedDream version 8.5.0 and Token-service version 2.1.8. The changes apply for TokenService API versions v4 generate and validate endpoints (example: http://127.0.0.1:8088/v4/generate and http://127.0.0.1:8088/v4/validate).
The complete description and examples of corresponding TokenService API versions see in Annex X (this Annex).
Annex describes json body structure, parameters and related TokenService configuration options for version v4.

"restrictions" object and "series" array object: allowed identifiers in v1.4

The changes in "restrictions" object and "series" array object allowed identifiers in v1.4:

- “series”, “study”, “series” – identifiers in “restrictions” part allows to filter and display only indicated series in the array. Other series of the study will not be displayed.

“permissions”: […] allowed values in v1.4 differences from v1.1

The changes in “permissions”: […] allowed values in v1.4:

- KO_PR_VIEW added to allowed values of “permissions” array to allow viewing KO/PR;
- KO_PR_EDIT added to allowed values of “permissions” array to allow creating/viewing KO/PR

.json structure example

```
{
 "items": [
   {
     "studies": {
       "accnum": “accnum1”,
       "patient": null,
       "study": null,
       "storage": "storage1"
     },
     "history": [
       {
         "accnum": null,
         "patient": “pat2”,
         "study": null,
         "storage": "storage1"
       }
     ]
   }
 ],
 "permissions": [
   "PATIENT_HISTORY"
 ],
 "restrictions": {
   "patient": [
     "pat2"
   ],
   "series": [
       {
         "study": "1.2.3.4.6.7.8",
          "series": ["1.2.3”,”7.8.9"]
       },
       {
         "study": "9.10.11.12.13.14",
          "series": ["1.1.2”,”1.2.4"]
       }
   ]
 },
 "user": {
```

```
    "id": "123",
    "name": "user name"
  },
  "storageConfiguration": [
    {
      "storage": "s1",
      "parameters": [
        {
          "name": "parameter name",
          "value": "parameter value"
        }
      ]
    }
  ],
  "segmentation": {
    "segments": [
      {
        "instance": "inst1",
        "storage": "s1"
      }
    ]
  },
  "pluginConfigurations": [
    {
      "pluginName": "name",
      "parameters": [
        {
          "name": "parameter name",
          "value": "parameter value"
        }
      ]
    }
  ]
}
```

Parameters description

The table below provides the structure of json body and explains the parameters: name, description and validation rules.

Parameters description	**Structure of json body**
	{
Required. Empty array not allowed. Maximums 50 items are allowed. Array of items, that should be allowed to access in MedDream search or viewer windows. Each item contains the required study or studies identifiers, and optional historical studies array for the item.	*"items":*
	[*{*
Required. Empty object not allowed. Identifiers for retrieving the study or studies.	*"studies":*
	{
Required study or studies identifier. Allowed one of listed: • "study" – one study, identified by Study Instance UID; • "patient" – one or more studies for the patient, identified by patient ID; • "accnum" – one or more studies, identified by accession number;	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",*

• "accnum" , "patient" – one or more studies, identified by accession number AND patient ID; • “file” – path to studies DICOM file or folder with studies DICOM files; • "patient" , “studyDate” – one or more studies, identified by patient ID AND study date, parameters pair added in v1.2. Empty value is not allowed. If “file” identifier is used, the other identifiers are not allowed to be used in the same token.	*"file": "path1",*
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s4"*
	},
Optional. Empty array not allowed. Array of historical studies. Each array object contains the required study or studies identifiers and storage identifier. ⚠ **NOTE!** To enable access to historical studies, you must include *permissions=["PATIENT_HISTORY"].* ⚠ **NOTE!** List of history studies is collected from studies, that are retrieved according the identifiers in *"history": […]* array, and adding the study or studies themselves, if not present. ⚠ **NOTE!** The same historical studies list is assigned to each study, if more than one retrieved according the identifiers in *studies: {…}* object, and to study from the historical studies list, if such study is not included in any in *studies: {…}* object and hasn’t historical studies list assigned. ⚠ **CAUTION!** If *permissions=["PATIENT_HISTORY"]* is set, and *"history": […]* array is not provided, the default historical studies collection is used: the studies with the same patient ID that are selected from the same storage are collected as historical studies. If you want to prevent such behavior, include into *"history": […]* array object equal to *studies: {…}* object, and you will have historical studies list including only the study or studies themselves. The default historical studies selection is not supported with FileSystem plugin and “file” identifier.	*"history":*
	[*{*
Required. Empty value is not allowed. See parameters description and validation rules above, in *studies{…}* object.	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"storage": "s4"*
	} *]* *}* *],*
Optional. Empty array not allowed. Customizable MedDream functionality, that is granted for the URL request using the token. List of allowed values: EXPORT_ISO, EXPORT_ARCH, FORWARD, REPORT_VIEW, REPORT_UPLOAD, PATIENT_HISTORY, UPLOAD_DICOM_LIBRARY, 3D_RENDERING, ADMIN, ANONYMOUS_VIEW, DOCUMENT_VIEW, BOUNDING_BOX_VIEW, BOUNDING_BOX_EDIT, FREE_DRAW_VIEW, FREE_DRAW_EDIT, LIVESHARE_GUEST.	*" permissions":* *[* *"PATIENT_HISTORY",* *" ADMIN"* *],*

SMART_DRAW_VIEW, SMART_DRAW_EDIT, COPY_TO_DICOM values added to the allowed values in v0.8. USER_SETTINGS, CLEAR_CACHE values added to the allowed values in v1.0. PACSONE_VIEW_ONLY_PUBLIC, SHORTCUTS_EDIT, and HANGING_PROTOCOLS_EDIT value added to allowed values in v1.1. KO_PR_VIEW, KO_PR_EDIT values added to allowed values in v.1.4. If not included, the system uses the permissions from application properties parameter authorization.defaultHisPermissions. ⚠ **NOTE!** See detail description of MedDream's built-in authentication and authorization in the Install manual of the used MedDream version. ⚠ **NOTE!** The access to customizable MedDream functionality may also be restricted by license, system properties (system.json), and settings (global.json). Please note, that access to functionality is granted only if no other source (license, system properties, or settings) restricts it.	
Optional. Empty object not allowed. Data access restrictions, that are applied for the URL request using the token.	*"restrictions":*
	{
Optional. Empty array not allowed. Empty object in the array not allowed, validation added in v0.9. Array of patient IDs. Patients, which studies is allowed to be retrieved and viewed, should be listed. Used on patient portals to provide access to only to the medical data of authenticated patient. ⚠ **NOTE!** The patient array with only one value is required, if system property allowOnePatientViewOnly is set to true.	*"patient": ["pat2"],*

Optional. Empty array not allowed.
 	
Provided array of Series Instance UIDs will load only these series for the study. All other series of the study fill be filtered and not displayed.
 	
⚠ **NOTE!** There are the following limitations using these restrictions in MedDream Viewer:
When using Forward functionality whole study (with all series) will be forwarded.
When using Export functionality whole study (with all series) will be downloaded.
When using Live Share functionality, the guest will see all series thumbnails.
	"series": [
	{
	"study": "1.2.3.4.6.7.8",
	"series": ["1.2.3”,”7.8.9"]
	},
	{
	"study": "9.10.11.12.13.14",
	"series": ["1.1.2”,”1.2.4"]
	}
	]

	},
Optional. Empty object not allowed. Information about the user, who connects to the system with the token.	*"user":*
	{
Optional. Empty value is not allowed. Key for user identification in HIS and MedDream systems.	*"id": "UserID",*
Optional. Empty value is not allowed. User name, middle name, surname, used for presentation purposes.	*"name": "User name"*
	},
Optional. Empty array not allowed.	*"storageConfiguration":*

Information for granting and configuring user's access to data.	
	[*{*
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s1",*
Required. Empty array is not allowed. Parameter for connecting to the PACS data storage and data access configuration.	*"parameters":* *[*
Parameter name and value pair. Both attributes required, and empty values are not allowed. Parameter name should be validated according the TokenService configuration. See the description of supported parameters per particular plugin in the Install manual of the used MedDream version.	*{* *"name": "Name",* *"value": "Value"* *}*
	] *}* *],*
Optional. Empty object not allowed. Segmentation objects to be viewed or edited by the user.	*"segmentation":* *{*
Optional. Empty array allowed. Array with list of accessible RTSTRUCT instances, that contains saved segments to be viewed or edited by the user. ⚠ **NOTE!** To enable segmentation edit rights, you must include any of *permissions=["BOUNDING_BOX_EDIT", "FREE_DRAW_EDIT"]*. To enable segmentation view rights, you must include any of *permissions=["BOUNDING_BOX_VIEW", "FREE_DRAW_VIEW"]*. If edit permission, for example *"BOUNDING_BOX_EDIT",* is granted, the view permission *"BOUNDING_BOX_VIEW"* is granted automatically for the accessible RTSTRUCT instances. ⚠ **NOTE!** If edit permission, for example *permissions=["BOUNDING_BOX_EDIT"]*, is set, and empty *"segments":[]* array is passed, the created segmentations will be stored in a new RTSTRUCT instance in a new series. ⚠ **NOTE!** If any of view or edit permission is granted, and *"segments"* array is not passed (NULL), the default segmentations retrieve and/or saving procedure is applied: the newest RTSTRUCT version from all the RT series is returned as accessible for the URL session.	*"segments":[*
Required RTSTRUCT instance identifier and storage identifier: • "instance" – RTSTRUCT SOP instance UID; • "storage" – Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties. Empty values are not allowed.	*{* *"instance": "inst1",* *"storage": "s4"* *}*
	] *},*
Optional. Empty array not allowed. Information for granting and configuring Viewer plugins.	*"pluginConfigurations":*
	[*{*
Required. Empty value is not allowed.	*"pluginName": "name",*

Unique plugin name. Value should be identical as in MedDream Viewer plugins configuration.	
Required. Empty array is not allowed. Viewer plugin configuration parameters.	*"parameters":* *[*
Parameter name and value pair. Both attributes required, and empty values are not allowed. Parameter name should be allowed in Viewer plugin configuration and is validated by MedDream application. See the description of supported Viewer plugin parameters in the Install manual of the used MedDream version.	*{* *"name": "Name",* *"value": "Value"* *}*
	] *}* *],*
	}

Version related configuration

TokenService does not have version v4 specific configuration options.

API v4 differences from API v3

The changes in v4, comparing to v3:

- “restrictions” object has additional array of “series”, “study”, “series” object in json body.

Annex XI. json structure and parameters description for API v4

Annex describes the changes in json “user” object: new “group” array and “institution” object implemented in TokenService v1.5. Available from MedDream version 8.6.0 and Token-service version 2.1.10. The changes apply for TokenService API versions v4 generate and validate endpoints (example: http://127.0.0.1:8088/v4/generate and http://127.0.0.1:8088/v4/validate).
The complete description and examples of corresponding TokenService API versions see in Annex XI (this Annex).
Annex describes json body structure, parameters and related TokenService configuration options for version v4.
Also new token.items-max-count parameter implemented in v1.5. The complete parameter description see in section “Installation and configuration”.

"groups" array, "institution" object in "user" object: allowed identifiers in v1.5

The changes in "user" object: new "groups" array and "institution" object allowed identifiers in v1.5:

- “groups” – array in “user” part allows to indicate specific global.json which can be loaded to MedDream Viewer. Specific settings file needs to be named by the following format *groupname_g_global.json* and stored in MedDream\sys\settings folder. In addition it is possible to create hanging protocols and shortcuts for groups by creating groupname_g_hangingProtocolsV2.json, groupname_g_hangingProtocols.json, groupname_g_shortcuts.json
  For example, setting specific settings for the “patients” group and creating patients_g_global.json file would load these settings to MedDream Viewer if *"groups": ["patients"]* is passed in token JSON.
  NOTE: When using “groups” you cannot pass the USER_SETTINGS or ADMIN permissions as this would search for specific user_global.json or global.json file and would ignore the group indicated in “groups” array.
  NOTE: If multiple groups are provided, MedDream will sort groups by alphabetic order and load the first found *groupname_g_global.json* settings.

- “institution” – object in “user” part allows to provide the following information: Institution name, department name, institution address, institution phone number, email address. This information is used for IHE_PDI when exporting ZIP or ISO file of the study. Provided information is added to readme.txt and index.html files in the exported ZIP/ISO file. To see more information how to enable IHE_PDI export check the MedDream Install manual of Viewing functionalities.

.json structure example

```
{
 "items": [
  {
   "studies": {
    "accnum": "accnum1",
    "patient": null,
    "study": null,
    "storage": "storage1"
   },
   "history": [
    {
     "accnum": null,
     "patient": "pat2",
     "study": null,
     "storage": "storage1"
    }
   ]
  }
 ],
 "permissions": [
  "PATIENT_HISTORY"
 ],
 "user": {
  "id": "123",
  "name": "user name",
   "groups": ["groupname"],
   "institution":{
       "name": "institution name",
       "department": "department name",
       "address": "institution address",
```

```
        "phones": "institution phone numbers",
        "email": "institution email address"
        }
  },
  "storageConfiguration": [
    {
      "storage": "s1",
      "parameters": [
        {
          "name": "parameter name",
          "value": "parameter value"
        }
      ]
    }
  ],
  "segmentation": {
    "segments": [
      {
        "instance": "inst1",
        "storage": "s1"
      }
    ]
  },
  "pluginConfigurations": [
    {
      "pluginName": "name",
      "parameters": [
        {
          "name": "parameter name",
          "value": "parameter value"
        }
      ]
    }
  ]
}
```

Parameters description

The table below provides the structure of json body and explains the parameters: name, description and validation rules.

Parameters description	**Structure of json body**
	{
Required. Empty array not allowed. Maximums 50 items are allowed. Array of items, that should be allowed to access in MedDream search or viewer windows. Each item contains the required study or studies identifiers, and optional historical studies array for the item.	*"items":*
	[*{*
Required. Empty object not allowed. Identifiers for retrieving the study or studies.	*"studies":*
	{
Required study or studies identifier. Allowed one of listed: • "study" – one study, identified by Study Instance UID; • "patient" – one or more studies for the patient, identified by patient ID; • "accnum" – one or more studies, identified by accession number; • "accnum" , "patient" – one or more studies, identified by accession number AND patient ID;	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"file": "path1",*

• “file” – path to studies DICOM file or folder with studies DICOM files; • "patient" , “studyDate” – one or more studies, identified by patient ID AND study date, parameters pair added in v1.2. Empty value is not allowed. If “file” identifier is used, the other identifiers are not allowed to be used in the same token.	
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s4"*
	},
Optional. Empty array not allowed. Array of historical studies. Each array object contains the required study or studies identifiers and storage identifier. ⚠ **NOTE!** To enable access to historical studies, you must include *permissions=["PATIENT_HISTORY"]*. ⚠ **NOTE!** List of history studies is collected from studies, that are retrieved according the identifiers in *"history": […]* array, and adding the study or studies themselves, if not present. ⚠ **NOTE!** The same historical studies list is assigned to each study, if more than one retrieved according the identifiers in *studies: {…}* object, and to study from the historical studies list, if such study is not included in any in *studies: {…}* object and hasn’t historical studies list assigned. ⚠ **CAUTION!** If *permissions=["PATIENT_HISTORY"]* is set, and *"history": […]* array is not provided, the default historical studies collection is used: the studies with the same patient ID that are selected from the same storage are collected as historical studies. If you want to prevent such behavior, include into *"history": […]* array object equal to *studies: {…}* object, and you will have historical studies list including only the study or studies themselves. The default historical studies selection is not supported with FileSystem plugin and “file” identifier.	*"history":*
	[*{*
Required. Empty value is not allowed. See parameters description and validation rules above, in *studies{…}* object.	*"accnum": "acc2",* *"patient": "pat2",* *"study": "stu1",* *"storage": "s4"*
	} *]* *}* *],*
Optional. Empty array not allowed. Customizable MedDream functionality, that is granted for the URL request using the token. List of allowed values: EXPORT_ISO, EXPORT_ARCH, FORWARD, REPORT_VIEW, REPORT_UPLOAD, PATIENT_HISTORY, UPLOAD_DICOM_LIBRARY, 3D_RENDERING, ADMIN, ANONYMOUS_VIEW, DOCUMENT_VIEW, BOUNDING_BOX_VIEW, BOUNDING_BOX_EDIT, FREE_DRAW_VIEW, FREE_DRAW_EDIT, LIVESHARE_GUEST. SMART_DRAW_VIEW, SMART_DRAW_EDIT, COPY_TO_DICOM values added to the allowed values in v0.8.	*" permissions":* *[* *"PATIENT_HISTORY",* *" EXPORT_ISO"* *],*

USER_SETTINGS, CLEAR_CACHE values added to the allowed values in v1.0. PACSONE_VIEW_ONLY_PUBLIC, SHORTCUTS_EDIT, and HANGING_PROTOCOLS_EDIT value added to allowed values in v1.1. KO_PR_VIEW, KO_PR_EDIT values added to allowed values in v.1.4. If not included, the system uses the permissions from application properties parameter authorization.defaultHisPermissions. ⚠ **NOTE!** See detail description of MedDream's built-in authentication and authorization in the Install manual of the used MedDream version. ⚠ **NOTE!** The access to customizable MedDream functionality may also be restricted by license, system properties (system.json), and settings (global.json). Please note, that access to functionality is granted only if no other source (license, system properties, or settings) restricts it.	
Optional. Empty object not allowed. Information about the user, who connects to the system with the token.	*"user":*
	{
Optional. Empty value is not allowed. Key for user identification in HIS and MedDream systems. ⚠ **NOTE!** Provided user ID will be visible in MedDream Viewer User details menu.	*"id": "UserID",*
Optional. Empty value is not allowed. User name, middle name, surname, used for presentation purposes. ⚠ **NOTE!** Provided user name will be visible in MedDream Viewer User details menu.	*"name": "User name",*
Optional. Empty value is not allowed. Provided array of groups. If groupname_g_global.json exists it will load MedDream settings for provided group. ⚠ **NOTE!** If using “groups” you cannot pass the USER_SETTINGS or ADMIN permissions as this would search for specific user_global.json or global.json file and would ignore the group indicated in “groups” array.	*"groups": ["groupname"],*
Optional. Provided object “institution” allows to set the following information: Institution name, department name, institution address, institution phone number, email address. This information is used for IHE_PDI when exporting ZIP or ISO file of the study. Provided information is added to readme.txt and index.html files in the exported ZIP/ISO file.	*"institution": {*
Optional. Institution name.	*"name": "institution name",*
Optional. Department name.	*"department": "department name",*
Optional. Institution address	*"address": "institution address",*
Optional.	*"phones": "institution phone numbers",*

Institution phone number	
Optional. Institution email address	*"email": "institution email address"*
	}
	},
Optional. Empty array not allowed. Information for granting and configuring user's access to data.	*"storageConfiguration":*
	[*{*
Required. Empty value is not allowed. Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*"storage": "s1",*
Required. Empty array is not allowed. Parameter for connecting to the PACS data storage and data access configuration.	*"parameters":* *[*
Parameter name and value pair. Both attributes required, and empty values are not allowed. Parameter name should be validated according the TokenService configuration. See the description of supported parameters per particular plugin in the Install manual of the used MedDream version.	*{* *"name": "Name",* *"value": "Value"* *}*
	] *}* *],*
Optional. Empty object not allowed. Segmentation objects to be viewed or edited by the user.	*"segmentation":* *{*
Optional. Empty array allowed. Array with list of accessible RTSTRUCT instances, that contains saved segments to be viewed or edited by the user. ⚠ **NOTE!** To enable segmentation edit rights, you must include any of *permissions=["BOUNDING_BOX_EDIT", "FREE_DRAW_EDIT"]*. To enable segmentation view rights, you must include any of *permissions=["BOUNDING_BOX_VIEW", "FREE_DRAW_VIEW"]*. If edit permission, for example *"BOUNDING_BOX_EDIT",* is granted, the view permission *"BOUNDING_BOX_VIEW"* is granted automatically for the accessible RTSTRUCT instances. ⚠ **NOTE!** If edit permission, for example *permissions=["BOUNDING_BOX_EDIT"]*, is set, and empty *"segments":[]* array is passed, the created segmentations will be stored in a new RTSTRUCT instance in a new series. ⚠ **NOTE!** If any of view or edit permission is granted, and *"segments"* array is not passed (NULL), the default segmentations retrieve and/or saving procedure is applied: the newest RTSTRUCT version from all the RT series is returned as accessible for the URL session.	*"segments":[*
Required RTSTRUCT instance identifier and storage identifier: • "instance" – RTSTRUCT SOP instance UID; • "storage" – Unique source VNA/PACS identifier. Value should be identical as configured in MedDream application properties.	*{* *"instance": "inst1",* *"storage": "s4"* *}*

Empty values are not allowed.	
	] }, "pluginConfigurations":
Optional. Empty array not allowed. Information for granting and configuring Viewer plugins.	
	[{
Required. Empty value is not allowed. Unique plugin name. Value should be identical as in MedDream Viewer plugins configuration.	"pluginName": "name",
Required. Empty array is not allowed. Viewer plugin configuration parameters.	"parameters": [

Parameter name and value pair. Both attributes required, and empty values are not allowed. Parameter name should be allowed in Viewer plugin configuration and is validated by MedDream application.
 	
See the description of supported Viewer plugin parameters in the Install manual of the used MedDream version.
	{
	"name": "Name",
	"value": "Value"
	}

	] }],
	}

Version related configuration

The changes in tokenService configuration in v1.5:

Parameter token.items-max-count added to configuration for setting maximum number of items loaded in token-enabled integration.

API v4 differences from API v3

The changes in v4, comparing to v3:

- “user” object has additional array of “groups” and “institution” object in json body.

Index

API v1 description 40
API v1 json structure 40
API v1 parameters description 40
API v1 related configuration 42
API v1, v2, v3 changes in ‘patient’ values validation in v0.9 53
API v1, v2, v3 changes in ‘permissions’ allowed values in v0.8 52
API v1, v2, v3 changes in ‘permissions’ allowed values in v1.0 54
API v1, v2, v3 changes in ‘permissions’ allowed values in v1.1 55
API v2 description 43
API v2 json structure 43
API v2 parameters description 43
API v2 related configuration 46
API v3 description 47
API v3 json structure 47
API v3 parameters description 48
API v3 related configuration 51
API v4 description 56, 62, 68
API v4 json structure 56, 62, 68
API v4 parameters description 57, 63, 69
API v4 related configuration 60, 67, 73
Authentication related configuration 61
Changes in authentication in v1.3 61
Document purpose 5
Explanation of symbols used 5
Generate token request 31
Index 74
Installation and configuration 27
Integration scenario using JSON Web Token (JWT) 16
Integration scenario using MedDream token 14
Introduction 6
Invalidate token request 35
MedDream integration interface 22
Minimum system requirements 7
One step token enabled MedDream URL integration 9
Related configuration options 37
Scenarios for MedDream URL integration 8

Table of Contents 3
Table of Figures 39
Token disabled MedDream URL integration scenario 20
Token enabled MedDream URL integration scenario using 3rd party token generation & validation service 14
Token enabled MedDream URL integration scenarios using MedDream TokenService 8
TokenService description 26
Two steps token enabled MedDream URL integration 11
Validate token request 34

MedDream is manufactured by MedDream UAB.
Medical device class: Regulation (EU) 2017/745
Class IIb medical device
FDA cleared K222320
ID of the notified body: 0197
Document version 1.0
Date of issue: 2026-10-01
Language: EN

CE 0197
Class IIb certified

FDA K222320 510(k)
CLEARED

MedDream UAB
K. Petrausko st. 26
LT-44156, Kaunas, Lithuania