Australia’s National
Science Agency

Smart Health Check Application

Practice Management System Integration

9 April 2024

TABLE OF CONTENTS

1. INTRODUCTION

1.1. Purpose

The purpose of this document is to provide an overview of integrating a Practice Management System with the Smart Health Checks App.

This document will be used by software providers of practice management systems that are integrating with the Smart Health Checks App.

1.2. Definitions, acronyms, and abbreviations

The following definitions may be referenced in this document.

Term	Definition
FHIR	Fast Healthcare Interoperability Resources
OAuth	Open Authorization
PMS	Practice Management System
SHCA	Smart Health Checks App
SMART	Substitutable Medical Apps and Reusable Technology
URL	Uniform Resource Locator
UTC	Coordinated Universal Time

1.3. Document Structure

This document has the following sections:

Section 1: Introduction
Section 2: Solution Overview
Section 3: Logical View

2. SOLUTION OVERVIEW

2.1. Overview

The Smart Health Checks App (SHCA) is a standards-based health check data collection application. It allows a Practice Management System (PMS) to support the collection of health check data without the need to implement and maintain each health check within the PMS itself. The PMS is integrated with the SHCA using HL7 FHIR standards and associated implementation guides as part of implementing the App Launch, Authorization Service and FHIR Server Application Programming Interfaces as shown in Figure 1 below.

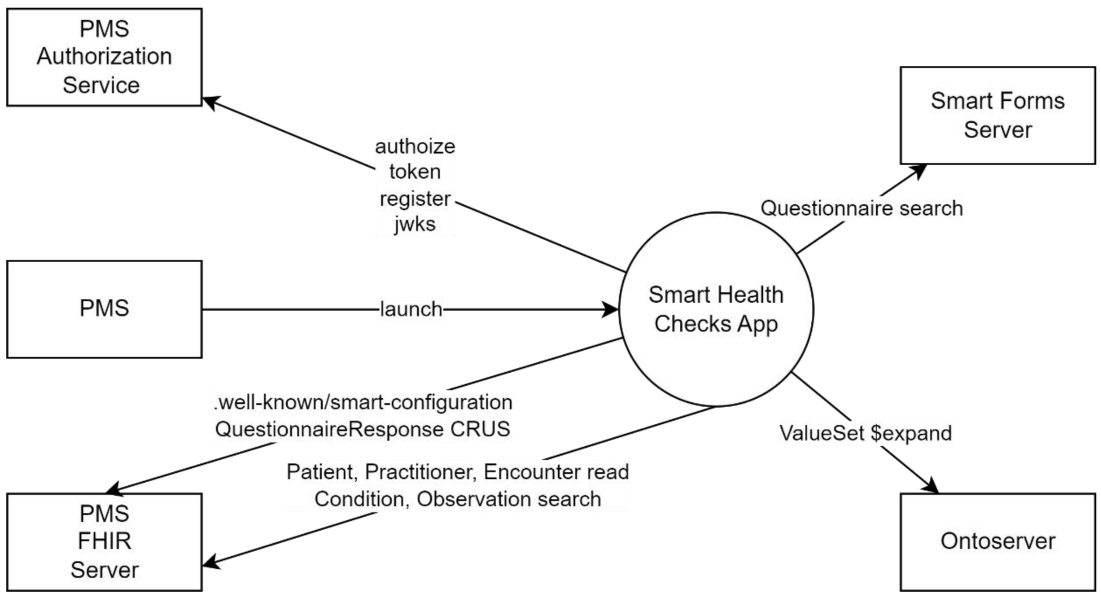
PMS Authorization Service
authoize
token
register
jwks
Smart Forms Server
Questionnaire search
PMS
launch
Smart Health Checks App
.well-known/smart-configuration
QuestionnaireResponse CRUS
Patient, Practitioner, Encounter read
Condition, Observation search
ValueSet $expand
PMS FHIR Server
Ontoserver

Figure 1 Smart Health Checks Application Context Diagram

The PMS uses the FHIR SMART App Launch framework to launch the SHCA with a unique launch context that correlates to the current user, current patient, current patient visit and required health check.

The Smart Health Checks App should be integrated within the PMS using an embedded Web browser or inline frame (iframe). When necessary, the SHCA may be loaded as a separate Web browser instance.

As part of the SHCA launch sequence, the App will request authorization to access the patient health data using the PMS Authorization Service. The authorize request allows the PMS to grant the App an authorization code that enables the App to then retrieve an access token to the FHIR Server and the App launch data associated with the specified launch context.

Using the App launch data, such as the patient ID, user's Practitioner ID and health check questionnaire identifier, the SHCA will execute FHIR read and search requests against the PMS FHIR Server using the retrieved access token, to retrieve the patient health data required to pre-populate the requested health check.

The user records health check data within the SHCA and the health check response data is saved as draft or completed using the FHIR Server QuestionnaireResponse create or update request.

2.2. Scope

The scope of this document includes an overview of integrating a PMS with the Smart Health Checks App. It will describe the Web services and requests required to be implement by the PMS associated with the following:

- App Launch
- Authorization Service
- FHIR Server

2.3. Assumptions and Constraints

This solution integration assumes the following:

1. The PMS is a Desktop Application deployed on-premise within a primary care organisation's private network or cloud-based virtual private network
2. The PMS user agent will have access to SHCA and associated Ontoserver Terminology Service and Smart Health Checks Forms Server using public internet
3. The PMS Authorization Service will allow secure access from the PMS user agent hosting the SHCA within the private network or public internet when required
4. The PMS FHIR Server will allow secure access from the PMS user agent hosting the SHCA within the private network or public internet when required.
5. The PMS Authorization Service will maintain secure storage of launch context, authorization codes and access tokens as recommended by OAuth 2.0 specifications
6. The PMS FHIR Server will maintain storage of FHIR QuestionnaireResponse resources
7. The PMS FHIR Server will retrieve (read or search) the supported patient and user data from the PMS data store

2.4. Decision Register

This document is based on the following design decisions

Date	**Decision Makers**	**Decision Description**
		Should a Refresh Token be supported to allow a user filling out a health check to be able to save the response after more than an hour from launching the Smart Health Checks App? If so, the online_access scope shall be supported (it is recommended that the offline_access scope should not be used for public clients). An alternative is to allow the token expiry to be longer than 1 hour. Would need to agree what the maximum token expiry should be, such as 8 hours?

3. LOGICAL VIEW

3.1. High Level Solution Architecture

The high-level architecture of the Smart Forms solution is shown in Figure 2. The solution consists of a Practice Management System (PMS) integrated with Smart Forms Common Services. The PMS represents any existing or future PMS, on-premise or cloud-based, provided by any vendor whilst the Smart Forms Common Services shared by the multiple and various PMS instances. The Smart Forms Common Service is not limited to a single centralised deployment, it could be deployed within a jurisdiction, regionally or by vendor as deemed appropriate.

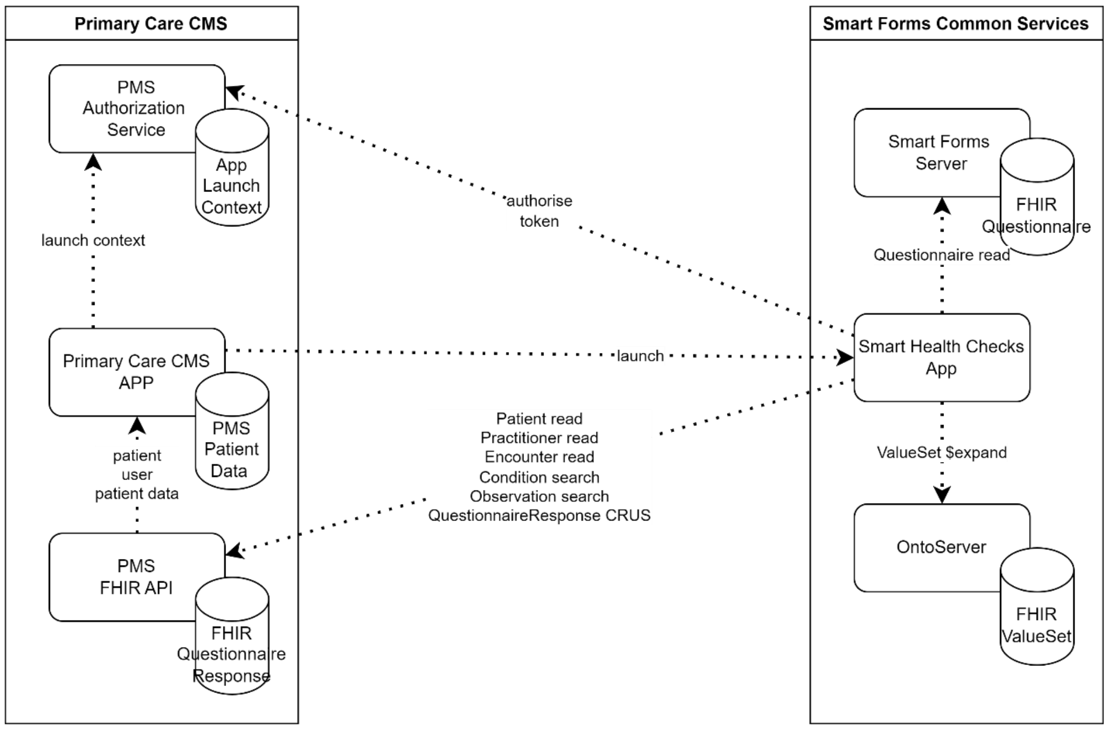

Figure 2 High Level Solution Architecture

3.1.1. Practice Management System Components

3.1.1.1. PMS App

The PMS App is an existing practice management system provided by a PMS software provider. The system may be deployed on-premise where the application and database servers are located within confines or control of the primary care organization. An on-premise system would normally be considered a desktop application, where the application executes on multiple workstations connected to a centralized database server. There are also some systems using a hybrid of on-premise and cloud infrastructure.

In the context of the Smart Health Check solution, the PMS App has a dependency on the PMS Authorization Service to stash launch context details such as the current patient, current user, current patient visit and the health check form to be filled, which can be retrieved later by the Smart Health Checks App.

The PMS also has a dependency on the Smart Health Checks App, which provides the standards-based health check data capture functionality provided by the Smart Health Check solution.

3.1.1.2. *PMS Authorization Service*

The PMS Authorization Service is an OAuth authorisation server that supports SMART App Launch context, which enables a third-party application, such as the Smart Health Checks App, to request access to patient data in the context of a user's workflow.

3.1.1.3. *PMS FHIR Server*

The PMS FHIR Server is a standards-based Application Programming Interface (API) that enables personal health information to be securely retrieved from the PMS. The FHIR Server uses the HL7 FHIR standard that specifies the type of data and behaviour that can be provided by the API. The FHIR Server also allows health information, such as a health check response (i.e. FHIR QuestionnaireResponse) to be stored in the PMS, which may be supported natively within the PMS data store or as a separate FHIR data repository.

The PMS FHIR Server has a dependency on the PMS App where it accesses the patient data and user records from the PMS data store.

3.1.2. Smart Forms Common Services Components

3.1.2.1. *Smart Health Checks App*

The Smart Health Checks App is a browser-based Web application that uses industry standards to provide a common user interface for the Smart Health Checks solution. The Smart Health Checks App uses a Single Page Application (SPA) paradigm that allows a modern JavaScript enabled Web browser to host the application without any logic executed or patient data stored on the server.

The Smart Health Checks App, uses the HL7 Structured Data Capture (SDC) Implementation Guide (IG) to implement a rendering of health check forms defined using FHIR Questionnaire resources and represent the data captured using FHIR QuestionnaireResponse resources. The SDC IG provides guidance and extensions that enables the Smart Health Checks App to provide advanced rending, pre-population, dynamic behaviour and forms modularisation in a standardised manner. This enables the Smart Health Checks App to be used for any compatible health check form defined in accordance with the HL7 Structured Data Capture (SDC) Implementation Guide (IG).

The Smart Health Checks App has a dependency on the PMS Authorization Service to request access to the PMS patient data with permissions authorised as part of the authorization flow.

The Smart Health Checks App has a dependency on the PMS FHIR Server to access the patient data required to pre-populate the health check form and store the captured health check data.

The Smart Health Checks App has a dependency on the Smart Forms Server that provides the health check forms as a FHIR Questionnaire resource. Since the Smart Health Checks App is hosted within the PMS User agent, the request to retrieve the Questionnaire resource originates from the PMS User agent to the Smart Forms Server over the public internet.

The Smart Health Checks App also has a dependency on the Ontoserver terminology server that maintains the value sets required in a health check form and expand filtered lists of coded concepts such as diagnoses and body sites. The Ontoserver requests that originate from the Smart Health Checks App will also be transported over the public internet.

3.1.2.2. *Smart Forms Server*

The Smart Forms Server is an implementation of a FHIR Server specifically used to support the management of health check forms represented as FHIR Questionnaire resources.

The Smart Forms Server will also be used during the health check form development to maintain reusable Questionnaire modules assembled into the combined health check form used by the Smart Health Checks App.

3.1.2.3. *Ontoserver*

Ontoserver is a terminology server with a FHIR API that allows the Smart Health Checks App to access FHIR ValueSet resources used in health checks and evaluate filtered expansions at runtime based on user entered lookahead requests for concepts such as diagnoses, medications and body sites.

3.2. Practice Management System

3.2.1. PMS App

The PMS App shall launch the Smart Health Checks App using the SMART App Launch sequence specified in the FHIR SMART App Launch Implementation Guide.

The PMS App may implement the Smart Health Checks App Launch using an embedded Web view or external Web browser and within any workflow if it is in accordance with the specified Smart Health Checks App Launch below.

3.2.1.1. *SMART App Launch Context*

To support the Smart Health Checks App Launch, the PMS App shall store the required launch context in representation and secure location accessible by the PMS Authorization Service.

The SMART App Launch context required by the Smart Health Checks App include the following:

patient	required	Current patient identifier used to retrieve the Patient via FHIR Server. Example: `87a339d0-8cae-418e-89c7-8651e6aab3c6`
encounter	optional	Current patient visit identifier used to retrieve the Encounter via FHIR Server. Example: `6673d31c-3cc3-43bc-adb6-cedb3122d881`
sub	required	Unique identifier of the user as known by the PMS authorization service. Example: `f256d3ba-bb70-4613-a631-825d500c57fa`
preferred_username	optional	Optional username used to login to PMS. Example: `janedoe`
fhirUser	required	Current user identifier used to retrieve the user's Practitioner resource via the FHIR API. Example: `Practitioner/8c7b7d27-88b6-44bc-b150-45acf9ce3f6b`

fhirContext	optional	A list of other FHIR resource references that have been requested as launch context. Each fhirContext item has a relative or canonical reference to a FHIR resource and a role represented as the canonical URL of the FHIR resource type. When a specified heath check form is requested health check's Questionnaire url shall be provided in the fhirContext. For example: `[{ "canonical": " http://www.health.gov.au/assessments/mbs/715", "role": "https://smartforms.csiro.au/smart/role/questionnaire-to-display", "type": "Questionnaire" }]`

The SMART App Launch context data shall be accessible by the PMS Authorization Service using a unique and opaque launch identifier, which cannot be guessed and ideally, does not include any launch context. For example, a Version 4 or Version 5 UUID may be generated as a key to access the launch context from an App Launch Context data store.

3.2.1.2. *SMART App Launch*

The Smart Health Checks App is launched using the SMART App Launch request within the embedded or external Web browser or iframe.

The SMART App launch is initiated by invoking a HTTP GET request to the registered launch URL of the Smart Health Checks App. The request shall include the following SMART App Launch query parameters.

iss	Base URL of the PMS FHIR Server. Example: `https://pmsserver.com.au/fhir`
launch	The launch identifier used to retrieve the launch data as part of the authorization process. Example: `15fd8cc0b5ce4e4b9ab1cb83495412f5`

An example of a launch request URL with encoded parameters is shown below.

```
https://smartforms.csiro.au/launch?iss=https%3A%2F%2Fserver.com.au/fhir&lau
nch=15fd8cc0b5ce4e4b9ab1cb83495412f5
```

The response to the SMART App Launch request will be a HTTP response that will be processed as usual by the Web browser agent that initiated the request. Under normal conditions the response will contain a HTML page or a HTTP Redirect response.

3.2.2. PMS Authorization Service

The PMS Authorization Service shall implement the OAuth 2.0 Authorization Code flow as required by the FHIR SMART App Launch IG. This includes the following SMART App Launch authorization endpoints:

- authorize
- token
- register
- jwks

The PMS Authorization Service shall support the retrieval of SMART App Launch context stored by the PMS App using a launch identifier/handle provided as part of the authorization flow.

The PMS Authorization Service shall maintain a data store of registered third-party applications, authorization codes and access tokens.

3.2.2.1. *authorize*

The PMS Authorization Service `authorize` request allows the Smart Health Checks App to authorize the user as part of their current launch context.

Request

The `authorize` request shall be supported using both HTTP GET and HTTP POST methods with the following parameters as specified in the FHIR SMART App Launch IG. The Smart Health Checks App will use the HTTP GET method only, but the HTTP POST method is supported to comply with the FHIR SMART App Launch IG conformance requirements.

Parameter	Description
response_type	Fixed value: `code`
client_id	The client identifier for the registered Smart Health Checks App. Example: `c6807eb65497423e8ef56f05956afb0f`
redirect_uri	The redirect URI for the registered Smart Health Checks App. Example: `https://smartforms.csiro.au`
launch	The launch identifier provided in the launch request correlates to the stashed launch context. Example: `15fd8cc0b5ce4e4b9ab1cb83495412f5`
scope	Space delimited list of authorization scopes requested by the App. Example: `launch openid fhirUser online_access patient/Patient.rs patient/Condition.rs patient/Observation.rs patient/Encounter.rs patient/QuestionnaireResponse.crus`
state	An unpredictable unique identifier used to correlate the authorization request with the subsequent redirect response. Example: `NhlJ741C31hRDf8v`
aud	Base URL of the FHIR server the App is requesting access. Example: `https://pmsserver.com.au/fhir`
code_challenge	PKCE code challenge generated by the App. Example: `E9Melhoa2OwvFrEMTJguCHaoeK1t8URWbuGJSstw-cM`
code_challenge_method	PKCE code challenge method. Example: `S256`

In the case of a HTTP GET request, the query parameters shall be URL encoded as shown in request example below.

```
GET https://https://auth.pmsserver.com.au/oauth/authorize?response_type=cod
e&client_id=c6807eb65497423e8ef56f05956afb0f&scope=launch%20openid%20fhirUs
er%20online_access%20patient%2FPatient.rs%20patient%2FCondition.rs%20patien
t%2FObservation.rs%20patient%2FEncounter.rs%20patient%2FQuestionnaireRespon
se.crus&redirect_uri=https%3A%2F%2Fsmartforms.csiro.au&state=NhlJ741C31hRDf
8v&aud=https%3A%2F%2Fpmsserver.com.au/fhir&launch=15fd8cc0b5ce4e4b9ab1cb834
95412f5&code_challenge=E9Melhoa2OwvFrEMTJguCHaoeK1t8URWbuGJSstw-
cM&code_challenge_method=S256
```

When the request is invoked using HTTP POST, the parameters are provided in the body of the request represented using the Web form URL encoding and indicated in the HTTP content-type header as `application/x-www-form-urlencoded`. An example of HTTP Post request is shown below.

```
POST /oauth/authorize
Content-Type: application/x-www-form-urlencoded
Host: https://auth.pmsserver.com.au

response_type=code&client_id=c6807eb65497423e8ef56f05956afb0f&scope=launch%
20openid%20fhirUser%20online_access%20patient%2FPatient.rs%20patient%2FCond
ition.rs%20patient%2FObservation.rs%20patient%2FEncounter.rs%20patient%2FQu
estionnaireResponse.crus&redirect_uri=https%3A%2F%2Fsmartforms.csiro.au&sta
te=NhlJ741C31hRDf8v&aud=https%3A%2F%2Fpmsserver.com.au%2Ffhir&launch=15fd8c
c0b5ce4e4b9ab1cb83495412f5&code_challenge=E9Melhoa2OwvFrEMTJguCHaoeK1t8URWb
uGJSstw-cM&code_challenge_method=S256
```

Processing Rules

The following processing rules should be applied for the authorize request:

1. Verify all request parameters are provided and valid. Any missing or invalid parameters shall result in an error parameter of invalid_request returned in the redirect response.
2. The client_id parameter is used to retrieve the registered client record from the Authorization data store. An error parameter of unauthorized_client shall be returned in the redirect response when an active client record does not exist.
3. The redirect_uri parameter is verified against the client record as a registered redirect_uri. An error parameter of unauthorized_client shall be returned in the redirect response when the redirect_uri is not registered.
4. The aud parameter is verified against the client record as the target resource server of the client. An error parameter of unauthorized_client shall be returned in the redirect response when the aud is not the registered client's target resource server.
5. The scope parameters are verified against the registered client scope records. Any requested scopes beyond those registered shall be adjusted to align with the registered scopes. For example, if a scope of patient/*.cruds is requested and a registered client scope is patient/*.rs, the requested scope will be adjusted to patient/*.rs. The adjusted scope value will be used in the subsequent processing rules and returned in the response. Any invalid, unknown or malformed scope values shall result in an error parameter of invalid_scope returned in the redirect response.
6. The state parameter is used to retrieve a previous request record. When a previous request with the same state is retrieved, an invalid_request response shall be returned.
7. The launch parameter is used to retrieve the stored launch context record. When a launch context record is not located, an error parameter of invalid_request shall be returned in the redirect response. When a previous request with the same launch parameter is retrieved, an invalid_request response shall be returned.
8. The launch context record is verified against the requested launch scopes to ensure all requested launch data has been provided. For example when scope contains launch/patient, launch/encounter, launch/questionnaire, fhirUser and openid, the launch context shall include the corresponding parameter values. When a launch context parameter is missing from the launch context, an error parameter of invalid_scope shall be returned in the redirect response.

9. The authorise request presents an Authorisation Confirmation page displaying the name of the client, name of the user, name of the patient, description of the encounter, name of the questionnaire and a description of the remaining allowed requested scopes to allow the authorising user to accept or decline access to the requested patient's record.
10. When a user declines the authorization request, an error parameter of access_declined shall be returned in the redirect response.
11. When a user accepts the authorisation request, a unique authorization code is generated. The authorization code should be generated using a Version 4 UUID to produce a 128 bit random string, or a Version 1 UUID concatenated with the current timestamp in UTC, hashed with SHA1 algorithm and encoded hexadecimal string to produce a 160 bit string. Hyphen characters may be stripped from UUID formatted strings, for example: `6673d31c3cc343bcadb6cedb3122d881`
12. An authorization context record with the authorization code as the key is stored in the authorization data store. The authorization context record shall include the code, client_id, aud, redirect_uri, launch context, adjusted scope, state, code_challenge, code_challenge_method and code expiry timestamp (UTC). The code expiry timestamp shall calculated using a configurable but short timespan, such as 60 seconds. A maximum expiry timespan shall be 10 minutes.
13. A success response shall be returned when the authorization record is successfully generated and stored.

Success Response

The `authorize` response is a HTTP Redirect with a HTTP status code of 302 Found and Location header URL matching the redirect_uri parameter provided in the request with the following parameters:

code	The authorization code. Example: `c1c3b2fe54334efb901e34b095f837dd`
state	The state value provided in the authorization request to correlate this redirect response with the request. Example: `NhlJ741C31hRDf8v`

An example of the response is shown below.

```
HTTP 302 Found
Location: https://smartforms.csiro.au?state=NhlJ741C31hRDf8v&code=c1c3b2fe5
4334efb901e34b095f837dd
```

When the Web browser client receives the authorize request response, it will redirect to this location URL within the Smart Health Checks App, which will process the state and code parameters as part of the next step of the authorization flow to exchange the code for an access token.

Error Response

Error response for the authorize request is returned as a HTTP Redirect with a HTTP status code of 302 Found. The Location header URL matches the redirect_uri parameter provided in the request with the following parameters:

error	The error code from the following set: • invalid_request

	• unauthorized_client • access_denied • unsupported_response_type • invalid_scope • server_error – unexpected server error, equivalent to HTTP 500 Internal Server Error status code • temporarily_unavailable – server temporarily unavailable, equivalent to HTTP 503 Server Unavailable status code
error_description	Optional human readable description of the error

An example of the response is shown below.

```
HTTP 302 Found
Location: https://smartforms.csiro.au?error=unauthorized_client&error_descr
iption=redirect_uri%20does%20not%20match%20client%20registration
```

When the Web browser client receives the authorize request response, the application shall display the error details and not attempt to access the FHIR Server.

Code Challenge

The `code_challenge` parameter provides mitigation against authorization code interception attacks based on the Proof key for Code Exchange (PKCE) internet standard RFC 7636. The following is provided for information only.

A code_challenge is generated by the client and provided in the authorize request by follow the following steps for each authorize request:

1. Generate a code verifier as a random string with a length between 43 and 128 using characters from [A-Z] / [a-z] / [0-9] / "-" / "." / "_" / "~". For example:

   `dBjftJeZ4CVP-mB92K27uhbUJU1p1r_wW1gFWFOEjXk`
2. Hash the code verifier string using SHA-256 hashing algorithm
3. Encode the hashed string using the Base64 URL encoding (i.e. ensure +, / and = characters are replaced as per section 5 of RFC4648).

3.2.2.2. *token*

The PMS Authorization Service `token` request allows the Smart Health Checks App to exchange an authorization token for an access token as part of the Smart Forms authorization process. Further to the access token, the launch context stashed by the PMS App is returned in the token response.

Request

The `token` request is a HTTP POST with the following parameters as specified in the FHIR SMART App Launch IG.

grant_type	Fixed value: `authorization_code`
code	code parameter value returned in authorize response
client_id	The client identifier for the registered Smart Health Checks App. Example: `c6807eb65497423e8ef56f05956afb0f`
redirect_uri	The redirect URI for the registered Smart Health Checks App. Example: `smartforms.csiro.au`

code_verifier	PKCE code used to verify this request against the code_challenge provided in the preceding authorize request. Example: `dBjftJeZ4CVP-mB92K27uhbUJU1p1r_wW1gFWFOEjXk`

An example of the token request is shown below.

```
POST /oauth/token
Content-Type: application/x-www-form-urlencoded
Host: https://auth.pmsserver.com.au
grant_type=authorization_code&code=c1c3b2fe54334efb901e34b095f837dd&client_
id=c6807eb65497423e8ef56f05956afb0f&redirect_uri=https%3A%2F%2Fsmartforms.c
siro.au&code_verifier=dBjftJeZ4CVP-mB92K27uhbUJU1p1r_wW1gFWFOEjXk
```

Processing Rules

The following processing rules should be applied for the authorize request:

1. The client_id shall exist as a registered client
2. The redirect_uri shall match a redirect_uris registered against the client_id
3. The grant_type shall match a grant_types registered against the client_id
4. The code shall not be used by a client more than once. When an authorization code is used more than once, the token request shall reject the request and revoke previously issued access tokens based on the authorization code.
5. The code_verifier shall be hashed using SHA-26 hashing algorithm and encoded using Base64 URL encoding (i.e. ensure +, / and = characters are replaced as per section 5 of RFC4648). The resulting encoded string shall be compared with the code_challenge stored against the authorization_code. When the code_challenge does not match the hashed and encode code_verifier, the request shall be rejected.
6. An ID token is generated as described in the ID Token section below.
7. A unique access token is generated as described in Access Token section below.
8. A token context record with an indexed access token is stored in the authorization data store. The token context record shall include the access token, authorization code, client_id, redirect_uri and token expiry timestamp (UTC). The code expiry timestamp shall be calculated using a configurable timespan in seconds. The expiry should be long enough for a user to complete their App launch task, such as 1 hour (3600 seconds).
9. A token response shall be returned when the token record is successfully generated and stored. The response shall include the following as described in Token Response below:
    a. access token
    b. expiry timespan
    c. adjusted scope from authorization context
    d. encoded id token
    e. launch context

ID Token

The ID Token is an encoded set of security claims used to securely identify the user of the launch session. The claims are represented as a JSON Web Token (JWT) encoded using JSON Web Signature (JWS) Compact serialization. It is composed of three parts:

- JWS Protected Header
- JWS Payload
- JWS Signature

Each of these three parts are BASE64URL encoded and concatenated with a period ('.') as the delimited as outlined below:

```
BASE64URL({JWS Protected Header}) + '.'
+ BASE64URL({JWS Payload}) + '.'
+  BASE64URL({JWS Signature})
```

JWS Protected Header

The JWS Protected Header has two elements as described below.

typ	Fixed value: `JWT`
alg	Signature algorithm (RSA SHA-256). Fixed value: `RS256`

An example is shown below.

```
{"typ":"JWT","alg":"RS256"}
```

This is Base64URL encode as shown below.

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9
```

JWS Payload

sub	Unique identifier of the user as known by the PMS authorization service. Example: `f256d3ba-bb70-4613-a631-825d500c57fa`
iss	Authorization service issuer. Example: `https://auth.pmsserver.com.au`
aud	Client uri. Example: `https://smartforms.csiro.au`
fhirUser	FHIR Practitioner reference. Example: `Practitioner/f256d3ba-bb70-4613-a631-825d500c57fa`
preferred_username	Optional username used by the user to login to the PMS. Example: `janedoe`
iat	Token issued at. Example: `1690903483`
exp	Token expiry time. Example: `1690907083`

An example of the JWS Payload is shown below.

```
{
 "sub":"f256d3ba-bb70-4613-a631-825d500c57fa",
 "iss":"https://auth.pmsserver.com.au",
 "aud":"https://smartforms.csiro.au",
 "fhirUser":"Practitioner/f256d3ba-bb70-4613-a631-825d500c57fa",
 "preferred_username":"janedoe",
 "iat":1690903483,
 "exp":1690907083
}
```

Below is Base64URL encode JWS Payload.

eyAKICJzdWIiOiJmMjU2ZDNiYS1iYjcwLTQ2MTMtYTYzMS04MjVkNTAwYzU3ZmEiLCAKICJpc3Mi
OiJodHRwczovL2F1dGgucG1zc2VydmVyLmNvbS5hdSIsIAogImF1ZCI6Imh0dHBzOi8vc21hcnRm

b3Jtcy5jc2lyby5hdSIsIAogImZoaXJVc2VyIjoiUHJhY3RpdGlvbmVyL2YyNTZkM2JhLWJiNzAtNDYxMy1hNjMxLTgyNWQ1MDBjNTdmYSIsIAogInByZWZlcnJlZF91c2VybmFtZSI6ImphbmVkb2UiLCAKICJpYXQiOjE2OTA5MDM0ODMsIAogImV4cCI6MTY5MDkwNzA4MyAKfSA

JWS Signature

The JWS Signature uses the RSA PKCS1 SHA-256 algorithm where the input string is the concatenation of the JWS Protected Header and JWS Payload delimited with a period (‘.’).

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyAKICJzdWIiOiJmMjU2ZDNiYS1iYjcwLTQ2MT
MtYTYzMS04MjVkNTAwYzU3ZmEiLCAKICJpc3MiOiJodHRwczovL2F1dGgucG1zc2VydmVyLmNvb
S5hdSIsIAogImF1ZCI6Imh0dHBzOi8vc21hcnRmb3Jtcy5jc2lyby5hdSIsIAogImZoaXJVc2Vy
IjoiUHJhY3RpdGlvbmVyL2YyNTZkM2JhLWJiNzAtNDYxMy1hNjMxLTgyNWQ1MDBjNTdmYSIsIAo
gInByZWZlcnJlZF91c2VybmFtZSI6ImphbmVkb2UiLCAKICJpYXQiOjE2OTA5MDM0ODMsIAogIm
V4cCI6MTY5MDkwNzA4MyAKfSA
```

The input string is signed using the Authorization server’s private key and the resulting octets are encoded using BASE64URL as shown below.

```
XHuRvmNvJJvJotJ9EOOy49Prc_zv9krh8bLdXUqYdp85A55woHaogGPrK-
_XltDrltV13NTqJQVMUD1YaXUwyZGsVITjdjHsURt0ei09wWfkVB9aMFDkO6p5WuEOmkr--
ZOmnDl298WVROnkKvqCRgBVc8-dfk9BvCo1o_ZdWER5zcvlD9xFlZIqJ7-
iM0s9c0YhOMjp2ZJTvSOxEBEcSV3xn2M1ZYQMTRrUlUtuBHtasUcqXVpqBYmWJGIo_GqWi7aD82
1kIMvv-
E275wnp5H76CXLQYxMLvnuUVK2ogmULjegn9JcOI9gOZz0a3B2VzkE3Cc6naIOoP_c8zZnF9w
```

Encoded ID token

The resulting ID token after concatenating the three parts is shown below.

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyAKICJzdWIiOiJmMjU2ZDNiYS1iYjcwLTQ2MT
MtYTYzMS04MjVkNTAwYzU3ZmEiLCAKICJpc3MiOiJodHRwczovL2F1dGgucG1zc2VydmVyLmNvb
S5hdSIsIAogImF1ZCI6Imh0dHBzOi8vc21hcnRmb3Jtcy5jc2lyby5hdSIsIAogImZoaXJVc2Vy
IjoiUHJhY3RpdGlvbmVyL2YyNTZkM2JhLWJiNzAtNDYxMy1hNjMxLTgyNWQ1MDBjNTdmYSIsIAo
gInByZWZlcnJlZF91c2VybmFtZSI6ImphbmVkb2UiLCAKICJpYXQiOjE2OTA5MDM0ODMsIAogIm
V4cCI6MTY5MDkwNzA4MyAKfSA.XHuRvmNvJJvJotJ9EOOy49Prc_zv9krh8bLdXUqYdp85A55wo
HaogGPrK-
_XltDrltV13NTqJQVMUD1YaXUwyZGsVITjdjHsURt0ei09wWfkVB9aMFDkO6p5WuEOmkr--
ZOmnDl298WVROnkKvqCRgBVc8-dfk9BvCo1o_ZdWER5zcvlD9xFlZIqJ7-
iM0s9c0YhOMjp2ZJTvSOxEBEcSV3xn2M1ZYQMTRrUlUtuBHtasUcqXVpqBYmWJGIo_GqWi7aD82
1kIMvv-
E275wnp5H76CXLQYxMLvnuUVK2ogmULjegn9JcOI9gOZz0a3B2VzkE3Cc6naIOoP_c8zZnF9w
```

This ID token can be decode and verified using the following JSON Web Key (JWK) using the JWT debugger at https://jwt.io/.[1]

```
{
 "kty": "RSA",
 "kid": "cc34c0a0-bd5a-4a3c-a50d-a2a7db7643df",
 "use": "sig",
 "n":
"pjdss8ZaDfEH6K6U7GeW2nxDqR4IP049fk1fK0lndimbMMVBdPv_hSpm8T8EtBDxrUdi1OHZfM
hUixGaut-3nQ4GG9nM249oxhCtxqqNvEXrmQRGqczyLxuh-fKn9Fg--
hS9UpazHpfVAFnB5aCfXoNhPuI8oByyFKMKaOVgHNqP5NBEqabiLftZD3W_lsFCPGuzr4Vp0YS7
```

[1] RSA private and public keys used are from https://connect2id.com/products/nimbus-jose-jwt/examples/jwk-generation

```
zS2hDYScC2oOMu4rGU1LcMZf39p3153Cq7bS2Xh6Y-
vw5pwzFYZdjQxDn8x8BG3fJ6j8TGLXQsbKH1218_HcUJRvMwdpbUQG5nvA2GXVqLqdwp054Lzk9
_B_f1lVrmOKuHjTNHq48w",
 "e": "AQAB"
}
```

Access Token

The access token is opaque to the client and its representation should only be known by the Authorization Service, and potentially for convenience the FHIR Server.

Therefore, the access token shall be either a UUID used to retrieve the token context from the Authorization Service using an introspection operation, or an encoded JWT using JWS.

UUID access token

A UUID may be generated as the access token similar to the authorization code, using a Version 4 UUID to produce a 128 bit random string, or a Version 1 UUID concatenated with the current timestamp in UTC, hashed with SHA1 algorithm and encoded hexadecimal string to produce a 160 bit string. Hyphen characters may be stripped from UUID formatted strings.

The benefit of a UUID access token is it is smaller in size, easy to generate and use to retrieve the token context. The downside of the UUID access token is that the token must be introspected on each request to ensure validity and retrieve token context.

An example of a UUID access token is shown below.

```
a88c5315a42e42d59acdcc67f69924b6
```

JWT encode access token

An access token generated using JWT may use the following elements.

sub	Unique identifier of the user as known by the PMS authorization service. Example: `f256d3ba-bb70-4613-a631-825d500c57fa`
iss	Authorization service issuer. Example: `https://auth.pmsserver.com.au`
aud	Client uri. Example: `https://smartforms.csiro.au`
fhirUser	FHIR Practitioner reference associated with the launch context. Example: `Practitioner/f256d3ba-bb70-4613-a631-825d500c57fa`
preferred_username	Optional username used by the user to login to the PMS. Example: `janedoe`
scope	List of authorized access scope delimited by space. Example: `launch openid fhirUser online_access patient/Patient.rs patient/Condition.rs patient/Observation.rs patient/Encounter.rs patient/QuestionnaireResponse.crus`
exp	Token expiry time. Example: `1690907083`
patient	Identifier of the patient associated with the launch context. Example: `87a339d0-8cae-418e-89c7-8651e6aab3c6`
encounter	Identifier of the current patient visit associated with the launch context. Example: `6673d31c-3cc3-43bc-adb6-cedb3122d881`

fhirContext	A list of other FHIR resource references provided in the launch context. Each fhirContext item has a relative or canonical reference to a FHIR resource and a role represented as the canonical URL of the FHIR resource type. When a specified heath check form is requested health check's Questionnaire URL shall be provided in the fhirContext. For example: `[{ "canonical": " http://www.health.gov.au/assessments/mbs/715", "role ": "https://smartforms.csiro.au/smart/role/questionnaire-to-display", "type": "Questionnaire" }]`

An example of the access token JWT is shown below.

```
{
 "sub":"f256d3ba-bb70-4613-a631-825d500c57fa",
 "iss":"https://auth.pmsserver.com.au ",
 "aud":"https://smartforms.csiro.au",
 "fhirUser":"Practitioner/f256d3ba-bb70-4613-a631-825d500c57fa",
 "preferred_username":"janedoe",
 "scope":"launch openid fhirUser online_access patient/Patient.rs
patient/Condition.rs patient/Observation.rs patient/Encounter.rs
patient/QuestionnaireResponse.crus",
 "iat":1690903483,
 "exp":1690907083
}
```

Below is Base64URL encoding of the access token JWT.

eyAKICJzdWIiOiJmMjU2ZDNiYS1iYjcwLTQ2MTMtYTYzMS04MjVkNTAwYzU3ZmEiLCAKICJpc3MiOiJodHRwczovL2F1dGgucG1zc2VydmVyLmNvbS5hdSIsIAogImF1ZCI6Imh0dHBzOi8vc21hcnRmb3Jtcy5jc2lyby5hdSIsIAogImZoaXJVc2VyIjoiUHJhY3RpdGlvbmVyL2YyNTZkM2JhLWJiNzAtNDYxMy1hNjMxLTgyNWQ1MDBjNTdmYSIsIAogInByZWZlcnJlZF91c2VybmFtZSI6ImphbmVkb2UiLCAKICJzY29wZSI6ImxhdW5jaCBvcGVuaWTigK9maGlyVXNlciBvbmxpbmVfYWNjZXNzIHBhdGllbnQvUGF0aWVudC5ycyBwYXRpZW50L0NvbmRpdGlvbi5ycyBwYXRpZW50L09ic2VydmF0aW9uLnJzIHBhdGllbnQvRW5jb3VudGVyLnJzIHBhdGllbnQvUXVlc3Rpb25uYWlyZVJlc3BvbnNlLmNydWRzIiwgCiAiaWF0IjoxNjkwOTAzNDgzLCAKICJleHAiOjE2OTA5MDcwODMgCn0Using the JWS encoding described for ID token above, the full access token is shown below.

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyAKICJzdWIiOiJmMjU2ZDNiYS1iYjcwLTQ2MT
MtYTYzMS04MjVkNTAwYzU3ZmEiLCAKICJpc3MiOiJodHRwczovL2F1dGgucG1zc2VydmVyLmNvb
S5hdSIsIAogImF1ZCI6Imh0dHBzOi8vc21hcnRmb3Jtcy5jc2lyby5hdSIsIAogImZoaXJVc2Vy
IjoiUHJhY3RpdGlvbmVyL2YyNTZkM2JhLWJiNzAtNDYxMy1hNjMxLTgyNWQ1MDBjNTdmYSIsIAo
gInByZWZlcnJlZF91c2VybmFtZSI6ImphbmVkb2UiLCAKICJzY29wZSI6ImxhdW5jaCBvcGVuaW
TigK9maGlyVXNlciBvbmxpbmVfYWNjZXNzIHBhdGllbnQvUGF0aWVudC5ycyBwYXRpZW50L0Nvb
mRpdGlvbi5ycyBwYXRpZW50L09ic2VydmF0aW9uLnJzIHBhdGllbnQvRW5jb3VudGVyLnJzIHBh
dGllbnQvUXVlc3Rpb25uYWlyZVJlc3BvbnNlLmNydWRzIiwgCiAiaWF0IjoxNjkwOTAzNDgzLCA
KICJleHAiOjE2OTA5MDcwODMgCn0.G2QgyuuGmRbR7wwvztcs057FP8yWBDd_Rvd9KzqKYwv-
dMaPZV05bVOpNgUskQX9qAi0zAibCJ_zFNWDRQFmICDeF2flzX_EuO987sbQ-X-k96q0cAfXAV-
R7fWVL4EUIlPgPK7XdRprqetsTVHA9z9LtpMc4sYXnxkEBWV2py8hvrwumMX6ysmN2bPoISONPB
GJYb38lj6plk4xbloZjERG707sJXQK22E_5dF4Wz_9ngVmygVbMjxwCAsUkF2LKwMnFQafrrccu
FOmljaLooZwjIklQLjmOB7AcAZr-v4MesashNFUeX5t2VSeJ-f-oNsulK-fRLbNE4X77mutcw
```

Response

The `token` response contains a JSON payload in the body with the following elements:

access_token	The access token. Example:
	`eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyAKICJzdWIiOiJ`
	`mMjU2ZDNiYS1iYjcwLTQ2MTMtYTYzMS04MjVkNTAwYzU3ZmEiLCA`
	`KICJpc3MiOiJodHRwczovL2F1dGgucG1zc2VydmVyLmNvbS5hdSI`
	`sIAogImF1ZCI6Imh0dHBzOi8vc21hcnRmb3Jtcy5jc2lyby5hdSI`
	`sIAogImZoaXJVc2VyIjoiUHJhY3RpdGlvbmVyL2YyNTZkM2JhLWJ`
	`iNzAtNDYxMy1hNjMxLTgyNWQ1MDBjNTdmYSIsIAogInByZWZlcnJ`
	`lZF91c2VybmFtZSI6ImphbmVkb2UiLCAKICJzY29wZSI6ImxhdW5`
	`jaCBvcGVuaWTigK9maGlyVXNlciBvbmxpbmVfYWNjZXNzIHBhdGl`
	`lbnQvUGF0aWVudC5ycyBwYXRpZW50L0NvbmRpdGlvbi5ycyBwYXR`
	`pZW50L09ic2VydmF0aW9uLnJzIHBhdGllbnQvRW5jb3VudGVyLnJ`
	`zIHBhdGllbnQvUXVlc3Rpb25uYWlyZVJlc3BvbnNlLmNydWRzIiw`
	`gCiAiaWF0IjoxNjkwOTAzNDgzLCAKICJleHAiOjE2OTA5MDcwODM`
	`gCn0.G2QgyuuGmRbR7wwvztcs057FP8yWBDd_Rvd9KzqKYwv-`
	`dMaPZV05bVOpNgUskQX9qAi0zAibCJ_zFNWDRQFmICDeF2flzX_E`
	`uO987sbQ-X-k96q0cAfXAV-`
	`R7fWVL4EUIlPgPK7XdRprqetsTVHA9z9LtpMc4sYXnxkEBWV2py8`
	`hvrwumMX6ysmN2bPoISONPBGJYb38lj6plk4xbloZjERG707sJXQ`
	`K22E_5dF4Wz_9ngVmygVbMjxwCAsUkF2LKwMnFQafrrccuFOmlja`
	`LooZwjIklQLjmOB7AcAZr-v4MesashNFUeX5t2VSeJ-f-oNsulK-`
	`fRLbNE4X77mutcw`

token_type	Fixed value: `Bearer`
expires_in	Number of seconds the access token is valid. By default the tokens expires in one hour, represent as shown in example below: `3600`

scope	Space delimited list of scopes granted to the App as part of the authorize process. The list of scopes may not be the same as what was requested. Example:
	`launch openid fhirUser online_access`
	`patient/Patient.rs patient/Condition.rs`
	`patient/Observation.rs patient/Encounter.rs`
	`patient/QuestionnaireResponse.crus`

id_token	Encoded identity using JSON Web Token (JWT) the OpenID Connect Core 1.0 specifications. When decoded and verified, the token contains identity details of the user that launched the app and a FHIR Practitioner resource reference representing the user. The `id_token` is provided when the `openid fhirUser` scopes are requested and granted. Example:
	`eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyAKICJzdWIiOiJ`
	`mMjU2ZDNiYS1iYjcwLTQ2MTMtYTYzMS04MjVkNTAwYzU3ZmEiLCA`
	`KICJpc3MiOiJodHRwczovL2F1dGgucG1zc2VydmVyLmNvbS5hdSI`
	`sIAogImF1ZCI6Imh0dHBzOi8vc21hcnRmb3Jtcy5jc2lyby5hdSI`
	`sIAogImZoaXJVc2VyIjoiUHJhY3RpdGlvbmVyL2YyNTZkM2JhLWJ`
	`iNzAtNDYxMy1hNjMxLTgyNWQ1MDBjNTdmYSIsIAogInByZWZlcnJ`
	`lZF91c2VybmFtZSI6ImphbmVkb2UiLCAKICJzY29wZSI6ImxhdW5`
	`jaCBvcGVuaWTigK9maGlyVXNlciBvbmxpbmVfYWNjZXNzIHBhdGl`
	`lbnQvUGF0aWVudC5ycyBwYXRpZW50L0NvbmRpdGlvbi5ycyBwYXR`
	`pZW50L09ic2VydmF0aW9uLnJzIHBhdGllbnQvRW5jb3VudGVyLnJ`
	`zIHBhdGllbnQvUXVlc3Rpb25uYWlyZVJlc3BvbnNlLmNydWRzIiw`
	`gCiAiaWF0IjoxNjkwOTAzNDgzLCAKICJleHAiOjE2OTA5MDcwODM`
	`gCn0.XHuRvmNvJJvJotJ9EOOy49Prc_zv9krh8bLdXUqYdp85A55`
	`woHaogGPrK-`
	`_XltDrltV13NTqJQVMUD1YaXUwyZGsVITjdjHsURt0ei09wWfkVB`
	`9aMFDkO6p5WuEOmkr--ZOmnDl298WVROnkKvqCRgBVc8-`
	`dfk9BvCo1o_ZdWER5zcvlD9xFlZIqJ7-`
	`iM0s9c0YhOMjp2ZJTvSOxEBEcSV3xn2M1ZYQMTRrUlUtuBHtasUc`
	`qXVpqBYmWJGIo_GqWi7aD821kIMvv-`
	`E275wnp5H76CXLQYxMLvnuUVK2ogmULjegn9JcOI9gOZz0a3B2Vz`
	`kE3Cc6naIOoP_c8zZnF9w`

~~refresh_token~~	~~An token that can be used to request a new access token to continue a session beyond the current access token expiry time. A refresh token is only provided when requested using the `online_access` scope.~~
patient	FHIR Patient id provided in the launch context. Example: `87a339d0-8cae-418e-89c7-8651e6aab3c6`
encounter	FHIR Encounter id provided in the launch context. Example: `6673d31c-3cc3-43bc-adb6-cedb3122d881`
fhirContext	A list of other FHIR resource references that have been requested as launch context. Each fhirContext item has a relative or canonical reference to a FHIR resource and a role represented as the canonical URL of the FHIR resource type. When a specified heath check form is requested, the Questionnaire canonical URL shall be provided in the fhirContext. For example: `[{ "canonical": " http://www.health.gov.au/assessments/mbs/715", "role": "https://smartforms.csiro.au/smart/role/questionnaire-to-display", "type": "Questionnaire" }]`

An example of the token response is shown below.

```
{
  "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyAKICJzdWIiOiJmMjU
2ZDNiYS1iYjcwLTQ2MTMtYTYzMS04MjVkNTAwYzU3ZmEiLCAKICJpc3MiOiJodHRwczovL2F1dG
gucG1zc2VydmVyLmNvbS5hdSIsIAogImF1ZCI6Imh0dHBzOi8vc21hcnRmb3Jtcy5jc2lyby5hd
SIsIAogImZoaXJVc2VyIjoiUHJhY3RpdGlvbmVyL2YyNTZkM2JhLWJiNzAtNDYxMy1hNjMxLTgy
NWQ1MDBjNTdmYSIsIAogInByZWZlcnJlZF91c2VybmFtZSI6ImphbmVkb2UiLCAKICJzY29wZSI
6ImxhdW5jaCBvcGVuaWTigK9maGlyVXNlciBvbmxpbmVfYWNjZXNzIHBhdGllbnQvUGF0aWVudC
5ycyBwYXRpZW50L0NvbmRpdGlvbi5ycyBwYXRpZW50L09ic2VydmF0aW9uLnJzIHBhdGllbnQvR
W5jb3VudGVyLnJzIHBhdGllbnQvUXVlc3Rpb25uYWlyZVJlc3BvbnNlLmNydWRzIiwgCiAiaWF0
IjoxNjkwOTAzNDgzLCAKICJleHAiOjE2OTA5MDcwODMgCn0.G2QgyuuGmRbR7wwvztcs057FP8y
WBDd_Rvd9KzqKYwv-
dMaPZV05bVOpNgUskQX9qAi0zAibCJ_zFNWDRQFmICDeF2flzX_EuO987sbQ-X-k96q0cAfXAV-
R7fWVL4EUIlPgPK7XdRprqetsTVHA9z9LtpMc4sYXnxkEBWV2py8hvrwumMX6ysmN2bPoISONPB
GJYb38lj6plk4xbloZjERG707sJXQK22E_5dF4Wz_9ngVmygVbMjxwCAsUkF2LKwMnFQafrrccu
FOmljaLooZwjIklQLjmOB7AcAZr-v4MesashNFUeX5t2VSeJ-f-oNsulK-fRLbNE4X77mutcw",
  "token_type": "Bearer",
  "expires_in": 3600,
  "scope": "launch openid fhirUser online_access patient/Patient.rs
patient/Condition.rs patient/Observation.rs patient/Encounter.rs
patient/QuestionnaireResponse.crus",
  "id_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyAKICJzdWIiOiJmMjU2ZDN
iYS1iYjcwLTQ2MTMtYTYzMS04MjVkNTAwYzU3ZmEiLCAKICJpc3MiOiJodHRwczovL2F1dGgucG
1zc2VydmVyLmNvbS5hdSIsIAogImF1ZCI6Imh0dHBzOi8vc21hcnRmb3Jtcy5jc2lyby5hdSIsI
AogImZoaXJVc2VyIjoiUHJhY3RpdGlvbmVyL2YyNTZkM2JhLWJiNzAtNDYxMy1hNjMxLTgyNWQ1
MDBjNTdmYSIsIAogInByZWZlcnJlZF91c2VybmFtZSI6ImphbmVkb2UiLCAKICJzY29wZSI6Imx
hdW5jaCBvcGVuaWTigK9maGlyVXNlciBvbmxpbmVfYWNjZXNzIHBhdGllbnQvUGF0aWVudC5ycy
BwYXRpZW50L0NvbmRpdGlvbi5ycyBwYXRpZW50L09ic2VydmF0aW9uLnJzIHBhdGllbnQvRW5jb
3VudGVyLnJzIHBhdGllbnQvUXVlc3Rpb25uYWlyZVJlc3BvbnNlLmNydWRzIiwgCiAiaWF0Ijox
NjkwOTAzNDgzLCAKICJleHAiOjE2OTA5MDcwODMgCn0.XHuRvmNvJJvJotJ9EOOy49Prc_zv9kr
h8bLdXUqYdp85A55woHaogGPrK-
_XltDrltV13NTqJQVMUD1YaXUwyZGsVITjdjHsURt0ei09wWfkVB9aMFDkO6p5WuEOmkr--
ZOmnDl298WVROnkKvqCRgBVc8-dfk9BvCo1o_ZdWER5zcvlD9xFlZIqJ7-
iM0s9c0YhOMjp2ZJTvSOxEBEcSV3xn2M1ZYQMTRrUlUtuBHtasUcqXVpqBYmWJGIo_GqWi7aD82
1kIMvv-
E275wnp5H76CXLQYxMLvnuUVK2ogmULjegn9JcOI9gOZz0a3B2VzkE3Cc6naIOoP_c8zZnF9w",
  "patient": "87a339d0-8cae-418e-89c7-8651e6aab3c6",
  "encounter": "6673d31c-3cc3-43bc-adb6-cedb3122d881"
  "fhirContext": [{ "canonical": "
http://www.health.gov.au/assessments/mbs/715", "role": "https://smartforms.
csiro.au/smart/role/questionnaire-to-display", "type": "Questionnaire" }]
}
```

3.2.2.3. *register*

The PMS Authorization Service `register` request allows the Smart Health Checks App to be registered as a client of the PMS FHIR Server. The Smart Health Checks App will allow a PMS Administrative user to specify the Authorization Service URL and their user credentials to register the Smart Health Checks App as a client in the PMS Authorization Service. As part of the registration, the Smart Health Checks App will provide a name and URL that may be displayed when a user authorizes the application access to patient information along with other authorization requirements the Smart Health Checks App has, including redirect URIs and, launch and access scopes.

Request

The `register` request is a HTTP POST with the following parameters as specified in RFC 7591.

client_name	Name of the client presented to the user during authorization. Example: `Smart Health Checks App`
client_uri	URL to a web page providing information about the client. Example: `https://smartforms.csiro.au`
launch_uri	URL to the SMART App launch page. Example: https://smartforms.csiro.au/launch
redirect_uris	List of registered redirect URIs for the client used to authenticate public clients. Example: `["https://smartforms.csiro.au"]`
grant_types	List of OAuth grant types that may be used by the client. Example: `["authorization_code"`~~`, "refresh_token"`~~`]`
response_types	List of OAuth response types that the client may use. Example: `["code"]`
token_endpoint_auth_method	The App is a public client, and hence does not have a client secret. Example: `none`
scope	String containing a space-separated list of scope values that the client can use when requesting access tokens. Example `launch openid fhirUser online_access patient/Patient.rs patient/Condition.rs patient/Observation.rs patient/Encounter.rs patient/QuestionnaireResponse.crus`
~~jwks_uri~~	~~URL string referencing the client's JSON Web Key (JWK) Set document, which contains the client's public keys. Example: `https://client.example.org/my_public_keys.jwks`~~

The `register` request shall be secured using Basic Authentication where the request includes a HTTP Authorization header with a token type of Basic. The Basic Authentication token is a Base64 encoded username and password joined by a colon (I.e. EncodeBase64({username}:{password}). For example where username is “james” with password “abc123”:

EncodeBase64("james:abc123") = amFtZXM6YWJjMTIz

The resulting Authorization header is shown below

```
Authorization: Basic amFtZXM6YWJjMTIz
```

The PMS Authorization Server shall decode and authenticate the user credentials provided in the Authorization header. The user used invoke the `register` request shall be an existing PMS administrative user that has the permission to allow a third party application such as the Smart Health Checks App to register as a client of the PMS Authorization Service and access patient data using the PMS FHIR Server.

An example of the `register` request is shown below.

```
POST /oauth/register HTTP/1.1
Content-Type: application/json
Host: https://auth.pmsserver.com.au
Authorization: Basic amFtZXM6YWJjMTIz

{
 "client_name": "Smart Health Checks App",
 "client_uri": "https://smartforms.csiro.au",
 "launch_uri": "https://smartforms.csiro.au/launch",
 "redirect_uris": ["https://smartforms.csiro.au"],
 "grant_types": ["authorization_code"],
 "response_types": ["code"],
 "token_endpoint_auth_method": "none",
 "scope": "openid fhirUser launch launch/patient launch/encounter
launch/questionnaire patient/*.rs patient/QuestionnaireResponse.crus
user/Practitioner.r "
}
```

Created Response

The `register` response contains a JSON payload in the body with the following elements.

client_id	Example: `34e08e809a754f94b10716ef02c80a57`

An example of the `register` response is shown below.

```
HTTP/1.1 201 Created
Content-Type: application/json

{
  "client_id": "34e08e809a754f94b10716ef02c80a57"
}
```

Error Response

The `register` error response contains a JSON payload in the body with the following elements.

error	Error code with following options: • invalid_redirect_uri One or more redirection URIs is invalid • invalid_client_metadata The value of one of the client metadata fields is invalid and the server has rejected this request Example: `invalid_redirect_uri`
error_description	Human-readable ASCII text description of the error used for debugging.

An example of the `register` error response is shown below.

```
HTTP/1.1 400 Bad Request
```

```
Content-Type: application/json

{
  "error": "invalid_redirect_uri",
  "error_description": "One or more redirection URIs is invalid"
}
```

3.2.2.4. jwks

The `jwks` endpoint allows the Smart Health Checks App to retrieve the JSON Web Key set from the Authorization Service to verify the ID Token retrieved as part of the token response.

Request

The `jwks` request is a HTTP GET with no parameters. For example:

```
https://auth.pmsserver.com.au/oauth/jwks
```

Response

The jwks response has the follow elements.[2]

keys	Any array of JSON Web Keys.
kty	Type of key. Example: `RSA`
kid	Key identifier. Example: `cc34c0a0-bd5a-4a3c-a50d-a2a7db7643df`
use	Intended use of the key. Example: `sig`
n	Modulus of the RSA public key in Base64URL encoding. Example: `pjdss8ZaDfEH6K6U7GeW2nxDqR4IP049fk1fK0lndimbMMVBdPv_ hSpm8T8EtBDxrUdi1OHZfMhUixGaut- 3nQ4GG9nM249oxhCtxqqNvEXrmQRGqczyLxuh-fKn9Fg-- hS9UpazHpfVAFnB5aCfXoNhPuI8oByyFKMKaOVgHNqP5NBEqabiL ftZD3W_lsFCPGuzr4Vp0YS7zS2hDYScC2oOMu4rGU1LcMZf39p31 53Cq7bS2Xh6Y- vw5pwzFYZdjQxDn8x8BG3fJ6j8TGLXQsbKH1218_HcUJRvMwdpbU QG5nvA2GXVqLqdwp054Lzk9_B_f1lVrmOKuHjTNHq48w`
e	Public exponent of the RSA key in Base64URL encoding. Example: `AQAB`

An example of a `jwks` response is shown below.

```
{
 "keys": [
  {
   "kty": "RSA",
   "kid": "cc34c0a0-bd5a-4a3c-a50d-a2a7db7643df",
   "use": "sig",
   "n":
"pjdss8ZaDfEH6K6U7GeW2nxDqR4IP049fk1fK0lndimbMMVBdPv_hSpm8T8EtBDxrUdi1OHZfM
hUixGaut-3nQ4GG9nM249oxhCtxqqNvEXrmQRGqczyLxuh-fKn9Fg--
hS9UpazHpfVAFnB5aCfXoNhPuI8oByyFKMKaOVgHNqP5NBEqabiLftZD3W_lsFCPGuzr4Vp0YS7
zS2hDYScC2oOMu4rGU1LcMZf39p3153Cq7bS2Xh6Y-
vw5pwzFYZdjQxDn8x8BG3fJ6j8TGLXQsbKH1218_HcUJRvMwdpbUQG5nvA2GXVqLqdwp054Lzk9
_B_f1lVrmOKuHjTNHq48w",
   "e": "AQAB"
  }
}
```

[2] https://medium.com/javarevisited/json-web-key-set-jwks-94dc26847a34

3.2.3. PMS FHIR Server

The PMS FHIR Server provides the foundations of the PMS FHIR Server by implementing a FHIR compatible service that supports the requirements of the Smart Health Checks App. This includes:

- .well-known/smart-configuration
- Patient read
- Practitioner read
- Encounter read
- Condition search
- Observation search
- QuestionnaireResponse search, read, create, update

The PMS FHIR Server shall be responsible for storing and maintaining the FHIR QuestionnaireResponse resources.

The following FHIR Server endpoints all use a common FHIR Server base URL denoted as `{fhir-base-url}`. An example of a FHIR Server base URL is as follows:

```
https://pmsserver.com.au/fhir
```

3.2.3.1. .well-known/smart-configuration

The PMS FHIR Server shall support the .well-known/smart-configuration request as specified in Smart App Launch Implementation Guide 2.1.0 – Release http://hl7.org/fhir/smart-app-launch/STU2.1/app-launch.html#retrieve-well-knownsmart-configuration with the required elements specified in http://hl7.org/fhir/smart-app-launch/STU2.1/conformance.html#using-well-known.

Request

The `.well-known/smart-configuration` request is a HTTP GET with no parameters.

An example of the `.well-known/smart-configuration` request is shown below.

```
{fhir-base-url}/.well-known/smart-configuration
```

Response

The `.well-known/smart-configuration` response contains a JSON payload with the following elements.

issuer	1..1	Authorization server's Issuer URI (Authorization server base URL). Example: `https://auth.pmsserver.com.au`
jwks_uri	1..1	Authorization server's JSON Web Key Set URL. Example: `https://auth.pmsserver.com.au/oauth/jwks`
authorization_endpoint	1..1	Authorization server's authorization URL Example: `https://auth.pmsserver.com.au/oauth/authorize`

grant_types_supported	1..*	Authorization endpoint grant types supported. Allowed values: • authorization_code Example: `["authorization_code"]`
token_endpoint	1..1	Authorization server's token URL Example: `https://auth.pmsserver.com.au/oauth/token`
token_endpoint_auth_method	0..1	The App is a public client, and hence does not have a client secret. Example: `none`
registration_endpoint	1..1	Authorization server registration endpoint. Example: `https://auth.pmsserver.com.au/oauth/register`
scopes_supported	1..*	List of authorization and launch context scopes supported by the authorization server as specified in http://hl7.org/fhir/smart-app-launch/STU2.1/scopes-and-launch-context.html. Allowed values: • openid • fhirUser • launch • launch/patient • launch/encounter • launch/questionnaire • Patient/{resource-type|*}.cruds • User/Practitioner.r Example: `["openid","fhirUser","launch","launch/patient","launch/encounter","launch/questionnaire","Patient/*.rs","Patient/QuestionnaireResponse.crus","User/Practitioner.r"]`
response_types_supported	1..*	Authorization response types supported. Allowed values: • code Example: `["code"]`
code_challenge_methods_supported	1..*	Fixed: S256

capabilities	1..*	SMART capabilities supported by the server. Allowed values: • launch-ehr • authorize-post • client-public • context-ehr-patient • context-ehr-encounter • permission-v2 • permission-patient • permission-user Example: `["launch-ehr","authorize-post","client-public","context-ehr-patient","context-ehr-encounter","permission-v2","permission-patient","permission-user"]`

An example of the `.well-known/smart-configuration` response is shown below.

```
{
  "issuer": "https://auth.pmsserver.com.au",
  "jkws_uri": "https://auth.pmsserver.com.au/oauth/jwks"
  "authorization_endpoint":
"https://auth.pmsserver.com.au/oauth/authorize",
  "grant_types_supported": ["authorization_code"],
  "token_endpoint": "https://auth.pmsserver.com.au/oauth/token",
  "token_endpoint_auth_method": "none",
  "registration_endpoint": "https://auth.pmsserver.com.au/oauth/register",
  "scopes_supported": ["openid", "fhirUser","launch","launch/patient",
"launch/encounter","launch/questionnaire","Patient/*.rs",
"Patient/QuestionnaireResponse.crus","User/Practitioner.r""],
  "response_types_supported": ["code"],
  "code_challenge_methods_supported": "S256",
  "capabilities": ["launch-ehr","authorize-post","client-public",
"context-ehr-patient","context-ehr-encounter","permission-v2",
"permission-patient","permission-user"]
}
```

3.2.3.2. *Patient read*

The Patient read request allows the Smart Health Checks App to retrieve a PMS patient record from the PMS datastore as a FHIR Patient resource.

The PMS FHIR Server shall support the Patient Read request as specified in the HL7 FHIR Release 4 RESTful API, http://hl7.org/fhir/R4/http.html#read. The response body shall comply with the AU Core Implementation Guide 0.3.0-ballot https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-patient.html.

The FHIR Server shall ensure that the id in the request and response resource matches the patient context associated with the Authorization access token.

On success, the FHIR server shall return a response with HTTP status 200 OK, with the retrieved resource as the body.

On failure, the FHIR server shall return a response with one of the following HTTP status and where possible, an OperationOutcome as the response body providing further details about the error.

401 Not Authorized	Authorization is required for the server interaction
404 Not Found	Resource with the specified id does not exist
500 Internal Server Error	Unexpected server error caused by an otherwise unhandled exception.

3.2.3.3. *Practitioner read*

The Practitioner read request allows the Smart Health Checks App to retrieve a PMS user record from the PMS datastore as a FHIR Practitioner resource.

The PMS FHIR Server shall support the Practitioner Read request as specified in the HL7 FHIR Release 4 RESTful API, http://hl7.org/fhir/R4/http.html#read. The response body shall comply with the AU Core Implementation Guide 0.3.0-Ballot https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-practitioner.html.

The FHIR Server shall ensure that the id in the request and response resource matches the fhirUser context associated with the Authorization access token.

On success, the FHIR server shall return a response with HTTP status 200 OK, with the retrieved resource as the body.

On failure, the FHIR server shall return a response with one of the following HTTP status and where possible, an OperationOutcome as the response body providing further details about the error.

401 Not Authorized	Authorization is required for the server interaction
404 Not Found	Resource with the specified id does not exist
500 Internal Server Error	Unexpected server error caused by an otherwise unhandled exception.

3.2.3.4. *Encounter read*

The Encounter read request allows the Smart Health Checks App to retrieve a Patient consultation visit record from the PMS datastore as a FHIR Encounter resource.

The PMS FHIR Server shall support the Encounter Read request as specified in the HL7 FHIR Release 4 RESTful API, http://hl7.org/fhir/R4/http.html#read. The response body shall comply with the AU Core Implementation Guide 0.3.0-ballot https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-encounter.html.

The FHIR Server shall ensure that the id in the request and response resource matches the encounter context associated with the Authorization access token.

On success, the FHIR server shall return a response with HTTP status 200 OK, with the retrieved resource as the body.

On failure, the FHIR server shall return a response with one of the following HTTP status and where possible, an OperationOutcome as the response body providing further details about the error.

401 Not Authorized	Authorization is required for the server interaction
404 Not Found	Resource with the specified id does not exist
500 Internal Server Error	Unexpected server error caused by an otherwise unhandled exception.

3.2.3.5. *Condition search*

The Condition search request allows the Smart Health Checks App to retrieve problem list items from the PMS App datastore as FHIR Condition resources.

The PMS FHIR Server shall support the Condition Search request as specified in the HL7 FHIR Release 4 RESTful API, http://hl7.org/fhir/R4/http.html#search. Note that both GET and POST methods shall be supported by the FHIR Server to be complaint with the FHIR specification. The Smart Health Checks App will use the GET method request.

The Condition search parameters shall also comply with Smart Forms Launcher Server for the Aboriginal and Torres Strait Islander Health Check IG Capability Statement https://build.fhir.org/ig/aehrc/smart-forms-ig/CapabilityStatement-SFLauncherServerAboriginalTorresStraitIslanderHealthCheck.html#Condition1.

The search response entry resources shall comply with the AU Core Implementation Guide 0.3.0-ballot https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-condition.html.

The FHIR Server shall ensure that the patient search parameter, when provided, matches the patient context associated with the Authorization access token.

The FHIR Server shall ensure that the returned results have a subject element that matches the patient context associated with the Authorization access token.

On success, the FHIR server shall return a response with HTTP status 200 OK, with a Bundle resource with type “searchset”. Each Bundle entry shall include the following elements.

total	Total number of matching resources, irrespective of the _count parameter
link.relation	Fixed: self
link.url	The full URL to execute the same search request on this FHIR server, including the _count and _sort parameters when provided.
timestamp	Timestamp the search results were processed
id	A unique id of the searchset bundle for this request.

Each Bundle entry shall include the following elements

search.mode	Fixed: match
fullUrl	The full URL to read the matched resource from this FHIR server
resource	The resource that matched the search parameters

When the _count parameter is specified, the results shall not return more than the number specified.

The FHIR Server is not required to support paging as specified in the HL7 FHIR RESTful API, http://hl7.org/fhir/R4/http.html#paging, and hence it is not required to specify any additional link occurrences other than the self link described above.

When the _sort parameter is specified, the results shall be returned in the specified order, including those records not returned in the response due to the _count parameter being specified.

On failure, the FHIR server shall return a response with one of the following HTTP status and where possible, an OperationOutcome response as the body, providing further details of the error.

400 Bad Request	Resource could not be parsed or failed basic FHIR validation rules
401 Not Authorized	Authorization is required for the server interaction

500 Internal Server Error	Unexpected server error caused by an otherwise unhandled exception.

3.2.3.6. *Observation search*

The Observation search request allows the Smart Health Checks App to retrieve various vital signs and risk factors from the PMS App datastore as FHIR Observation resources.

The PMS FHIR Server shall support the Observation Search request as specified in the HL7 FHIR Release 4 RESTful API, http://hl7.org/fhir/R4/http.html#search. Note that both GET and POST methods shall be supported by the FHIR Server to be complaint with the FHIR specification. The Smart Health Checks App will use the GET method request.

The Observation search parameters shall also comply with Smart Forms Launcher Server for the Aboriginal and Torres Strait Islander Health Check IG Capability Statement https://build.fhir.org/ig/aehrc/smart-forms-ig/CapabilityStatement-SFLauncherServerAboriginalTorresStraitIslanderHealthCheck.html#Observation1.

The search response entry resources shall comply with the following implementation guide profiles:

AU Core Smoking Status	https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-smokingstatus.html
AU Core Body Height	https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-bodyheight.html
AU Core Body Weight	https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-bodyweight.html
AU Core Head Circumference	https://hl7.org.au/fhir/core/0.2.2-preview/StructureDefinition-au-core-headcircum.html
AU Core Waist Circumference	https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-waistcircum.html
AU Core Blood Pressure	https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-bloodpressure.html
AU Core Heart Rate	https://hl7.org.au/fhir/core/0.3.0-ballot/StructureDefinition-au-core-heartrate.html
AU Core Lipid Result	https://hl7.org.au/fhir/core/0.2.2-preview/StructureDefinition-au-core-lipid-result.html

The FHIR Server shall ensure that the patient search parameter, when provided, matches the patient context associated with the Authorization access token.

The FHIR Server shall ensure that the returned results have a subject element that matches the patient context associated with the Authorization access token.

On success, the FHIR server shall return a response with HTTP status 200 OK, with a Bundle resource with type "searchset". Each Bundle entry shall include the following elements.

total	Total number of matching resources, irrespective of the _count parameter
link.relation	Fixed: self
link.url	The full URL to execute the same search request on this FHIR server, including the _count and _sort parameters when provided.
timestamp	Timestamp the search results were processed
id	A unique id of the searchset bundle for this request.

Each Bundle entry shall include the following elements

search.mode	Fixed: match
fullUrl	The full URL to read the matched resource from this FHIR server
resource	The resource that matched the search parameters

When the _count parameter is specified, the results shall not return more than the number specified.

The FHIR Server is not required to support paging as specified in the HL7 FHIR RESTful API, http://hl7.org/fhir/R4/http.html#paging, and hence it is not required to specify any additional link occurrences other than the self link described above.

When the _sort parameter is specified, the results shall be returned in the specified order, including those records not returned in the response due to the _count parameter being specified.

On failure, the FHIR server shall return a response with one of the following HTTP status and where possible, an OperationOutcome response as the body, providing further details of the error.

400 Bad Request	Resource could not be parsed or failed basic FHIR validation rules
401 Not Authorized	Authorization is required for the server interaction
500 Internal Server Error	Unexpected server error caused by an otherwise unhandled exception.

3.2.3.7. QuestionnaireResponse create

The QuestionnaireResponse create request allows the Smart Health Checks App to save a new QuestionnaireResponse resource to the PMS FHIR Server's Questionnaire Response datastore.

The PMS FHIR Server shall support the QuestionnaireResponse create request as specified in the HL7 FHIR Release 4 RESTful API, http://hl7.org/fhir/R4/http.html#create, where the body complies with the QuestionnaireResponse resource profile specified in HL7 Structured Data Capture 3.0.0 – STU 3 Implementation Guide http://hl7.org/fhir/uv/sdc/STU3/StructureDefinition-sdc-questionnaireresponse.html.

The FHIR Server shall ensure that the subject reference within the resource matches the patient context associated with the Authorization access token.

The FHIR Server shall allocate an initial versionId and lastUpdated timestamp in UTC representing when the resource is created on the server.

On success, the FHIR server shall return a response with HTTP status 201 Created. The response body should be empty unless the request included the HTTP header below, in which case the created resource shall be returned in the body.

```
Prefer: return=representation
```

The response shall include the following HTTP headers.

Location	Absolute or relative URL to the new resource using the following pattern: /QuestionnaireResponse/{id}/_history/{versionId}. For example: `Location: /QuestionnaireResponse/da694d6f-f4a8-43bf-b48b-201d6f9f7d5b/_history/1`
Last-Modified	Resource lastUpdate timestamp in UTC formatted as:. {day-name}, {day} {month} {year} {hour}:{minute}:{second} GMT • day-name: 3 characters, capital case • day: 2 digits • month: 3 characters, capital case • year: 4 digits • hour: 2 digits • minute: 2 digits • second: 2 digits For example: `Last-Modified: Sat, 02 Feb 2013 12:02:47 GMT`
ETag	Resource versionId as a weak ETag. For example: `ETag: W/"1"`

On failure, the FHIR server shall return a response with one of the following HTTP status and where possible, an OperationOutcome response as the body that provides further details of the error.

400 Bad Request	Resource could not be parsed or failed basic FHIR validation rules
401 Not Authorized	Authorization is required for the server interaction
422 Unprocessable Entity	Resource violated applicable FHIR profiles or server business rules
500 Internal Server Error	Unexpected server error caused by an otherwise unhandled exception.

3.2.3.8. *QuestionnaireResponse update*

The QuestionnaireResponse update request allows the Smart Health Checks App to save an update to an existing QuestionnaireResponse resource by id in the PMS FHIR Server's Questionnaire Response datastore.

The PMS FHIR Server shall support the QuestionnaireResponse update request as specified in the HL7 FHIR Release 4 RESTful API, http://hl7.org/fhir/R4/http.html#update, where the body complies with the QuestionnaireResponse resource profile specified in HL7 Structured Data Capture 3.0.0 – STU 3 Implementation Guide http://hl7.org/fhir/uv/sdc/STU3/StructureDefinition-sdc-questionnaireresponse.html.

The FHIR Server shall ensure that the subject reference within the resource matches the patient context associated with the Authorization access token.

When the request contains the HTTP header `If-Match`, as shown in the example below, the FHIR Server shall ensure that the most recent version of the resource matches the versionId provided in the header value. If the versionId does not match, the FHIR Server shall return a 412 Precondition Failed response.

```
If-Match: W/"23"
```

The FHIR Server shall store each version of the QuestionnaireResponse resource. Each version of the resource shall be allocated a unique versionId within the scope of the resource Id, and lastUpdated timestamp in UTC representing when the version of the resource was created on the server.

On success, the FHIR server shall return a response with HTTP status 200 OK. The response body should be empty unless the update request included the HTTP `Prefer` header below. In this case, the updated resource shall be returned in the body.

```
Prefer: return=representation
```

The response shall include the following HTTP headers.

Last-Modified	Resource lastUpdate timestamp in UTC formatted as:. {day-name}, {day} {month} {year} {hour}:{minute}:{second} GMT For example: `Last-Modified: Sat, 02 Feb 2013 12:02:47 GMT`
ETag	Resource versionId as a weak ETag. For example: `ETag: W/"2"`

On failure, the FHIR server shall return a response with one of the following HTTP status and where possible, an OperationOutcome response as the body that provides further details of the error.

400 Bad Request	Resource could not be parsed or failed basic FHIR validation rules
401 Not Authorized	Authorization is required for the server interaction
412 Precondition Failed	The versionId does not match the value specified in the If-Match header in the request.
422 Unprocessable Entity	Resource violated applicable FHIR profiles or server business rules
500 Internal Server Error	Unexpected server error caused by an otherwise unhandled exception.

3.2.3.9. *QuestionnaireResponse search*

The QuestionnaireResponse search request allows the Smart Health Checks App to search existing QuestionnaireResponse resources in the PMS FHIR Server's Questionnaire Response datastore.

The PMS FHIR Server shall support the QuestionnaireResponse search request as specified in the HL7 FHIR Release 4 RESTful API, http://hl7.org/fhir/R4/http.html#search. Note that both GET and POST methods shall be supported by the FHIR Server to be complaint with the FHIR specification. The Smart Health Checks App will use the GET method request.

The QuestionnaireResponse search parameters shall also comply with Smart Forms Launcher Server for the Aboriginal and Torres Strait Islander Health Check IG Capability Statement https://build.fhir.org/ig/aehrc/smart-forms-ig/CapabilityStatement-SFLauncherServerAboriginalTorresStraitIslanderHealthCheck.html#QuestionnaireResponse1.

In particular, the following search parameter and combination shall be supported, and at least one search parameter shall exist in the request.

patient	Patient id or, type and id. Example: `Patient/87a339d0-8cae-418e-89c7-8651e6aab3c6`
questionnaire	Questionnaire canonical url. Example: `http://www.health.gov.au/assessments/mbs/715`
status	QuestionnaireResponse status code or code and system. Example: `completed`
patient + questionnaire	Matches both patient id and questionnaire canonical url.

patient + status	Matches both patient id and status.

In addition, the following search result parameters shall be supported.

_count	max number resources returned in a single page. For example: `20`
_sort	search parameters used to order the return results. Multiple parameters may be provided, separated by a comma. Descending order may be specified using a – prefix. Required sort parameters include: • authored Example: `_sort= -authored`

The FHIR Server shall ensure that the patient search parameter, when provided, matches the patient context associated with the Authorization access token.

The FHIR Server shall ensure that the returned results have a QuestionnaireResponse.subject that matches the patient context associated with the Authorization access token.

On success, the FHIR server shall return a response with HTTP status 200 OK, with a Bundle resource with type "searchset". Each Bundle entry shall include the following elements.

total	Total number of matching resources, irrespective of the _count parameter
link.relation	Fixed: self
link.url	The full URL to execute the same search request on this FHIR server, including the _count and _sort parameters when provided.
timestamp	Timestamp the search results were processed
id	A unique id of the searchset bundle for this request.

Each Bundle entry shall include the following elements

search.mode	Fixed: match
fullUrl	The full URL to read the matched resource from this FHIR server
resource	The resource that matched the search parameters

When the _count parameter is specified, the results shall not return more than the number specified.

The FHIR Server is not required to support paging as specified in the HL7 FHIR RESTful API, http://hl7.org/fhir/R4/http.html#paging, and hence it is not required to specify any additional link occurrences other than the self link described above.

When the _sort parameter is specified, the results shall be returned in the specified order, including those records not returned in the response due to the _count parameter being specified.

On failure, the FHIR server shall return a response with one of the following HTTP status and where possible, an OperationOutcome response as the body, providing further details of the error.

400 Bad Request	Resource could not be parsed or failed basic FHIR validation rules

401 Not Authorized	Authorization is required for the server interaction
500 Internal Server Error	Unexpected server error caused by an otherwise unhandled exception.

3.2.3.10. QuestionnaireResponse read

The QuestionnaireResponse read request allows the Smart Health Checks App to retrieve an existing QuestionnaireResponse resource by id from the PMS FHIR Server's Questionnaire Response datastore.

The PMS FHIR Server shall support the QuestionnaireResponse read request as specified in the HL7 FHIR Release 4 RESTful API, http://hl7.org/fhir/R4/http.html#read, where the response body complies with the HL7 Structured Data Capture 3.0.0 – STU 3 Implementation Guide http://hl7.org/fhir/uv/sdc/STU3/StructureDefinition-sdc-questionnaireresponse.html.

The FHIR Server shall ensure that the returned resource has a QuestionnaireResponse.subject that matches the patient context associated with the Authorization access token.

On success, the FHIR server shall return a response with HTTP status 200 OK, with the retrieved resource as the body. The resource meta element shall include the versionId and lastUpdated elements as recorded against the most recent version of the resource on the server.

The response shall include the following HTTP headers.

Last-Modified	Resource lastUpdate timestamp in UTC formatted as:. {day-name}, {day} {month} {year} {hour}:{minute}:{second} GMT For example: `Last-Modified: Sat, 02 Feb 2013 12:02:47 GMT`
ETag	Resource versionId as a weak ETag. For example: `ETag: W/"2"`

On failure, the FHIR server shall return a response with one of the following HTTP status and where possible, an OperationOutcome as the response body providing further details about the error.

401 Not Authorized	Authorization is required for the server interaction
404 Not Found	Resource with the specified id does not exist
500 Internal Server Error	Unexpected server error caused by an otherwise unhandled exception.

For further information

Australian e-Health Research Centre
Liam Barnes
Liam.Barnes@csiro.au

Australian e-Health Research Centre
Kylynn Loi
Kylynn.Loi@csiro.au