AIR SoapUI USER GUIDE

Produced by: Services Australia

Last Edited: 14/11/2024

Version: 9.0

1. TABLE OF CONTENTS

1. INTRODUCTION

1.1. Overview

This document gives an overview of how to use the servicesaustralia-air-samples-vendor-soapui-project-v5 provided as part of your software vendor integration pack using SoapUI software.

The SoapUI software is an open-source web service testing application for service-oriented architectures and representational state transfers. Its functionality covers web service inspection, invoking, development, simulation and mocking, functional testing, load and compliance testing.

The sample SoapUI Project & SoapUI User Guide are provided to assist developers in developing their software and is optional to use:

The sample SoapUI project will allow developers to:

1. Confirm their connection to the Vendor Environment.
2. Simulate a web service. and
3. Show the request and response for Representational State Transfer (REST) web service calls.

Note: The requests and responses returned in the sample SoapUI project are for demonstration purposes only.

1.2. Purpose

The document provides instructions on how to use the SoapUI project to simulate/mock Australian Immunisation Register (AIR) web services.

1.3. Scope

This document specifies the

- Integration Pack

1.4. Out of Scope

Items explicitly not included in the scope of this document include:

1. Detailed payload specification, which can be found in the AIR Software Developers guide and is available from the Health System Developers Portal, by selecting ‘User Documentation’ > ‘AIR Documentation’.

1.5. Audience

This document is intended for:

Table 1: Profile Audience

Name	Role
Service Consumer	Web service consumers that intend to invoke the AIR service at runtime.
External Services Australia Stakeholders	All external Services Australia stakeholders, including Software Developers.

1.6. Assumptions

1.7. Preconditions

- SOAP UI V5.7.2 or later has been installed.

1.8. Actors

This document describes various criteria and they are categorised by using actors and artefacts. The actors used in this document are:

Table 2: Actors

Actor	Details
Software Developers	Software Developers that consume AIR services
AIR	Australian Immunisation Register (AIR)
SOAP UI	SoapUI version 5.7.2 or later

2. INSTRUCTIONS

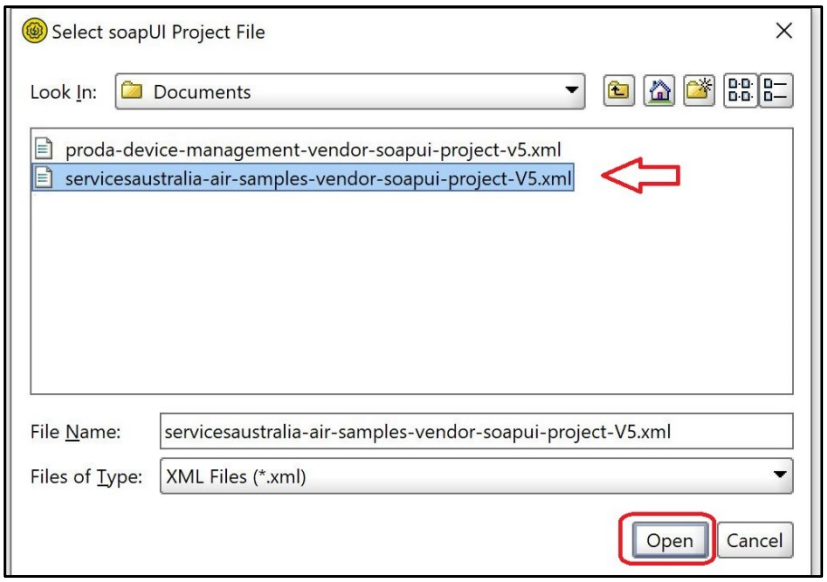
2.1. Pre-requisites

A PRODA Device and Java Key Store (JKS) has been setup as per the 'PRODA Device Management User Guide.pdf' document, which is available from the Health Systems Developers Portal, by selecting 'User Documentation' > 'PRODA Documentation'.

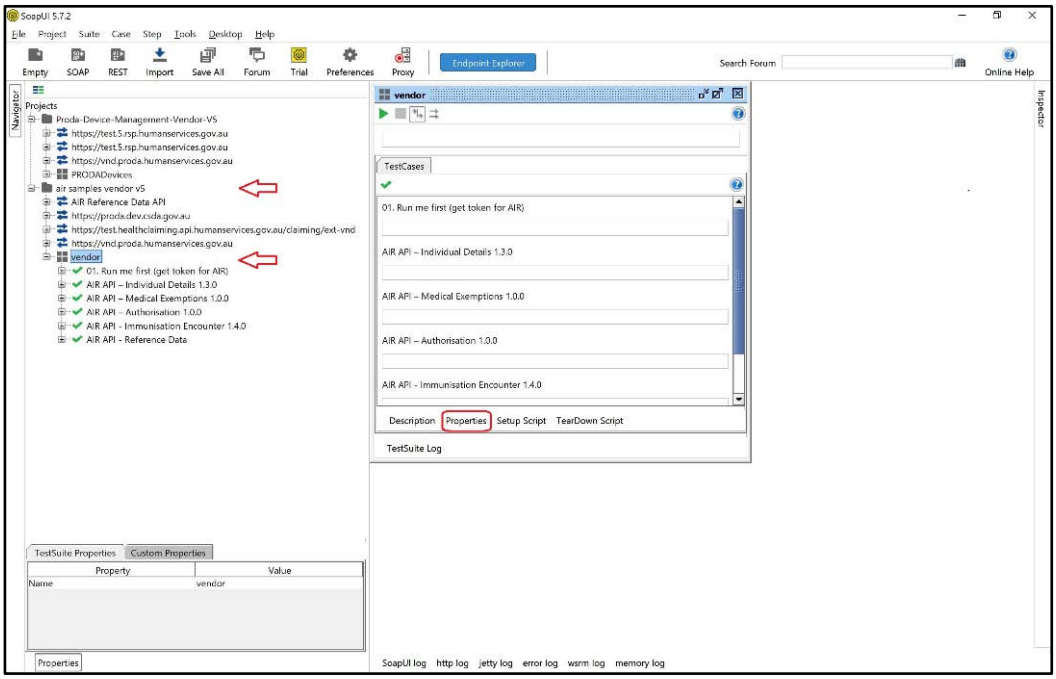
2.2. Setup Instructions

Start the SoapUI software.

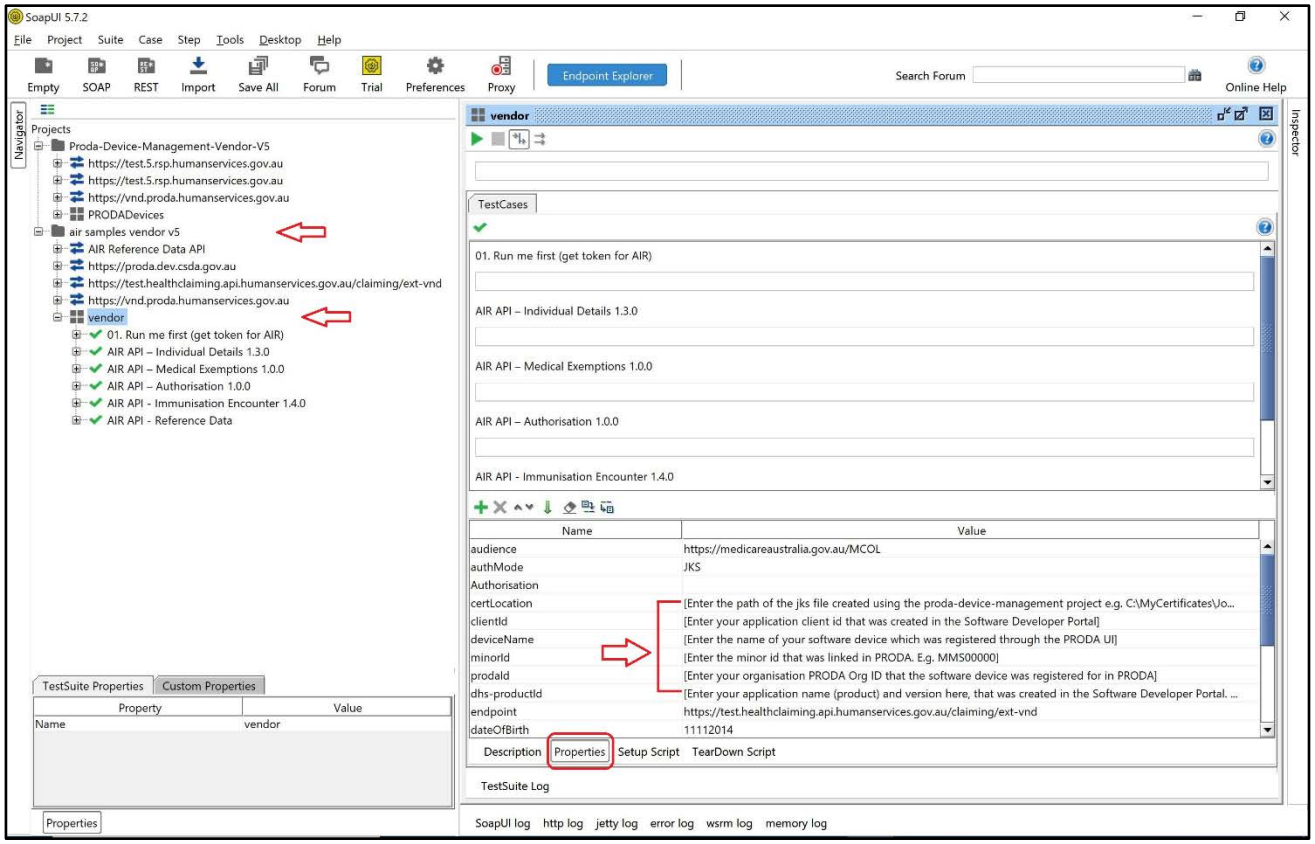
Import the SoapUI project into SoapUI by selecting File > Import Project > select the 'servicesaustralia-air-samples-vendor-soapui-project-v5.xml' file > Open.



The following 'air samples vendor V5' project will be imported to SoapUI as below:



Double Click the ‘vendor’ test suite, click the ‘Properties’ toggle at the bottom of the page.



Populate the properties values with relevant values.

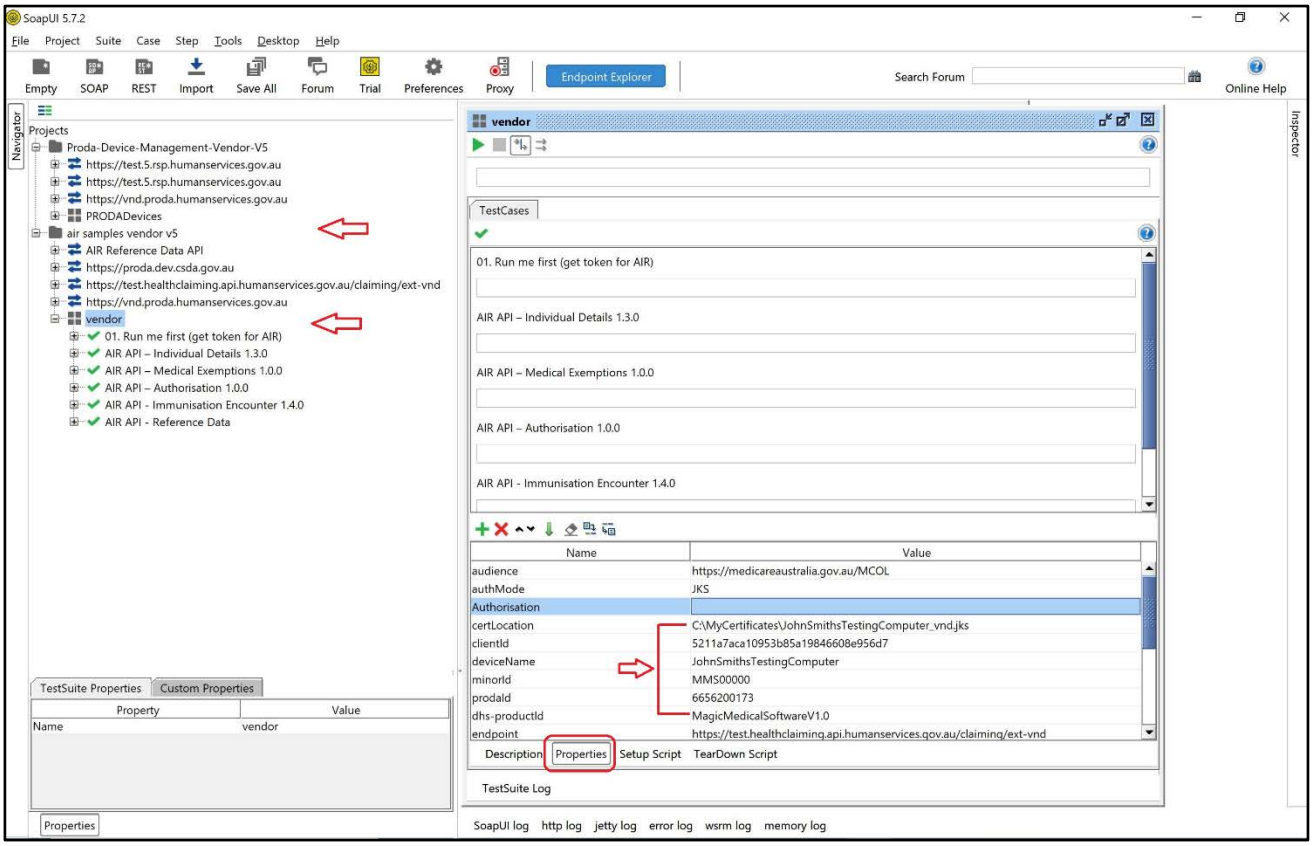
Note: Remove the brackets when you populate the values.

Name (Field)	Value (Description)	Source (Where is this obtained from)
certLocation	This points to a Java KeyStore (JKS) which will be in the directory on your development environment (computer), where generated certificates will be stored. E.g. C:\MyCertificates\JohnSmithsTestingComputer_vnd.jks	This is defined by the developer. Please enter the location of the JKS store created when you used the PRODA.Device.Management.Vendor-V5.xml SoapUI project. Note: You need to specify that actual location of the file. E.g. C:\MyCertificates\JohnSmithsTestingComputer_vnd.jks
clientId (Note: referred to as 'X-IBM-Client-Id' in the TechSIS documents)	Enter your application client id that was created in the Health Systems Developers Portal.	This is created when you enter the name of your application\product in the software developers' portal. It is recommended that you also include a version number in the name of your application\product. E.g. MagicMedicalSoftwareV1.0 The clientId is 32 characters and will be something similar to: 5211a7aca10953b85a19846608e956d7 Note: the above value is supplied as an example and is not valid. E.g. See example video 'Creating an Application and Subscribing to web services.mp4', which is available on the Developers Portal by selecting 'User Documentation' > 'Instructions and Guides'.
deviceName	This is the software instance (device) name registered with PRODA for the organisation.	This is created when you successfully Register a Device to your PRODA Organisation Vendor account. E.g. See example video '2. Linking a Proda Organisation Vendor Account and creating Device etc for use in the Vendor environment.mp4', which is available on the Developers Portal by selecting 'User Documentation' > 'PRODA Documentation'.

minorId	Enter the minor id that was linked in PRODA. E.g. MMS00000	This is minor id used when you added the MedicareOnline/Eclipse/DVA/AIR service through PRODA using the Add Service Provider option. E.g. See example video '2. Linking a Proda Organisation Vendor Account and creating Device etc for use in the Vendor environment.mp4', which is available on the Developers Portal by selecting 'User Documentation' > 'PRODA Documentation'.
prodaId	This is the PRODA organisation identifier (RA) of the relevant organisation	This is created when you successfully link your PRODA Individual Vendor account to your PRODA Organisation Vendor account. E.g. See example video '2. Linking a Proda Organisation Vendor Account and creating Device etc for use in the Vendor environment.mp4', which is available on the Developers Portal by selecting 'User Documentation' > 'PRODA Documentation'.
dhs-product-id	Enter your product name and version here. E.g. MagicMedicalSoftwareV1.0	This is defined by the developer when they create a new application in the developer portal. It is recommended that you also include a version number. E.g. MagicMedicalSoftwareV1.0 E.g. See example video 'Creating an Application and Subscribing to web services.mp4', which is available on the Developers Portal by selecting 'User Documentation' > 'Instructions and Guides'.
Authorisation	This field is automatically populated with the bearer token (JWT) created when executing the 'run me first (get token for AIR)' call. Note: The correct bearer token (JWT) must be obtained for the service you are simulating\mocking. E.g. • Australian Immunisation Register (AIR) • Medicare Online • MyMedicare • PBS Online A full listing of the values for the 'accesstokenAudience' is included in the 'PRODA B2B Software Developers Guide.pdf' file, (under the PRODA Service Provider Audience String section of the document), which is available from the Health Systems Developers Portal by selecting 'User Documentation' > 'PRODA Documentation'.	This is obtained when you obtain the correct bearer token for Medicare Online and AIR. Please follow the instructions in this document from page 10 to 14 (inclusive).

individualIdentifier	Placeholder individual identifier from a successful response of Identify Individual web service. In the SoapUI project populate with a 1 for the purpose of simulating the web services.	Obtained from a successful response of the Identify Individual web service.

See example shown below, which is only shown for display purposes;



Select File > Save All Projects.

A) Obtain a token for the AIR Service Provider and the authorization value.

In order to call a Medicare Online or AIR web service a JSON web token is required for the AIR Service Provider (The Audience).

Therefore, the next step is to execute a GetToken request from the 'servicesaustralia-air-samples-vendor-soapui-project-v5.xml' SoapUI project using SoapUI software. In the sample SoapUI project it is entitled:

- run me first (get token for AIR)

Note: The 'run me first (Get Token for AIR)' request audience has already been updated to:

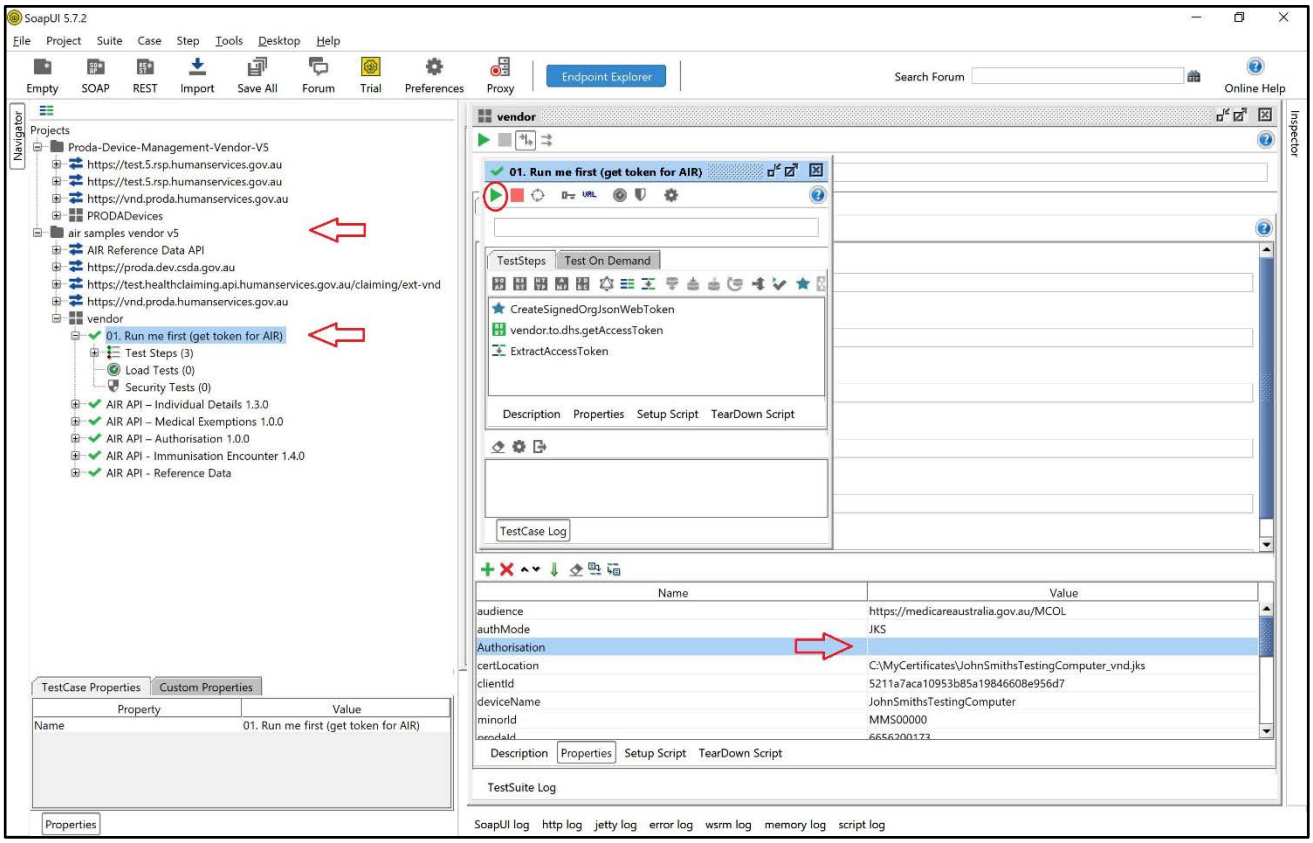
- **https://medicareaustralia.gov.au/MCOL**

A full listing of the values for the 'accesstokenAudience' is included in the 'PRODA B2B Software Developers Guide.pdf' file (under the PRODA Service Provider Audience String section) which is available from the Health Systems Developers Portal, by selecting 'User Documentation' > 'PRODA Documentation'.

Open the SoapUI software.

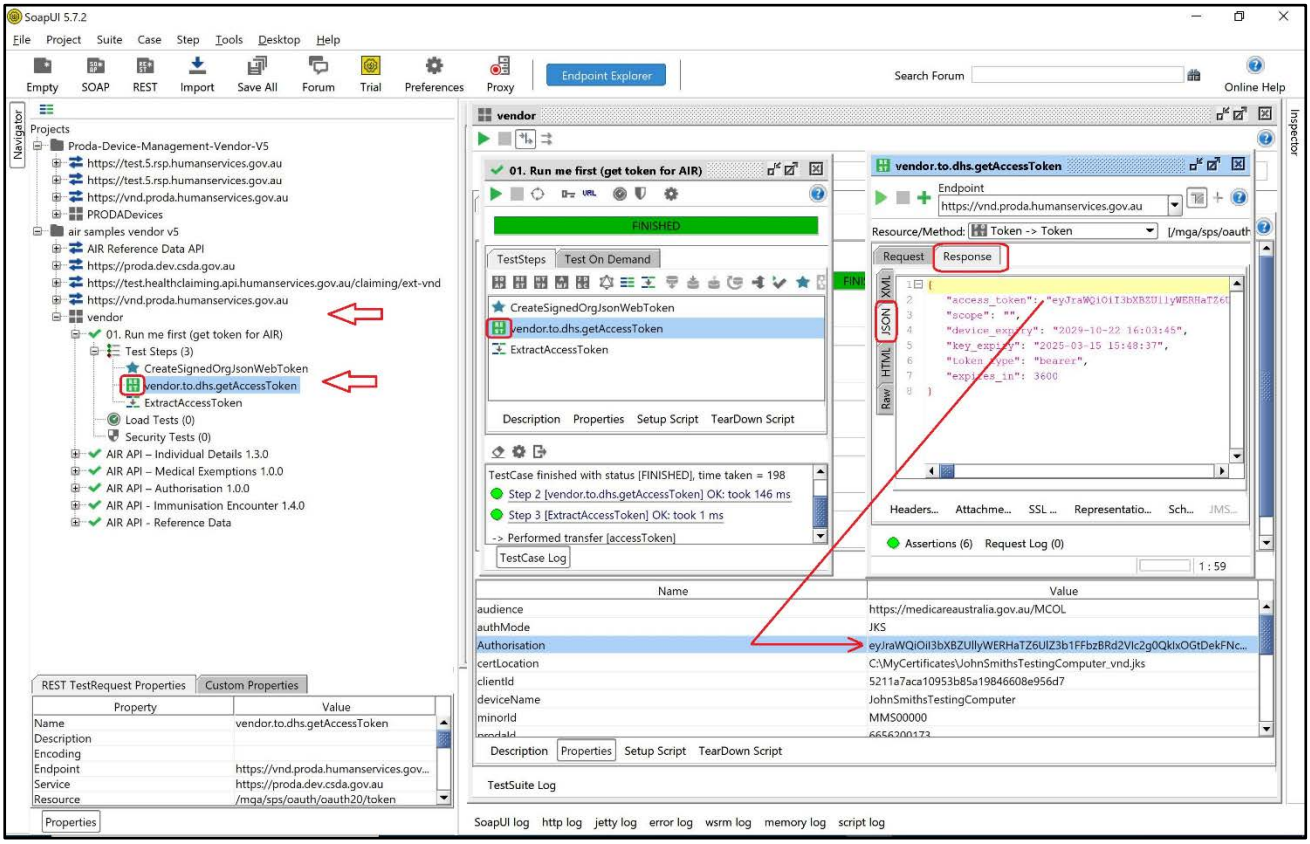
Simulate the **'run me first (get token for AIR)'** web service.

Press the play button 



The test should run correctly. I.e. FINISHED

By double clicking on the test step  , you can see the request and response data for the REST calls, by selecting the Raw tab on the Request tab and the Json tab on the Response tab.



The "access_token" component will have been automatically populated into the Authorisation value in the Properties for the AIR SoapUI project.

It will be something similar to below:

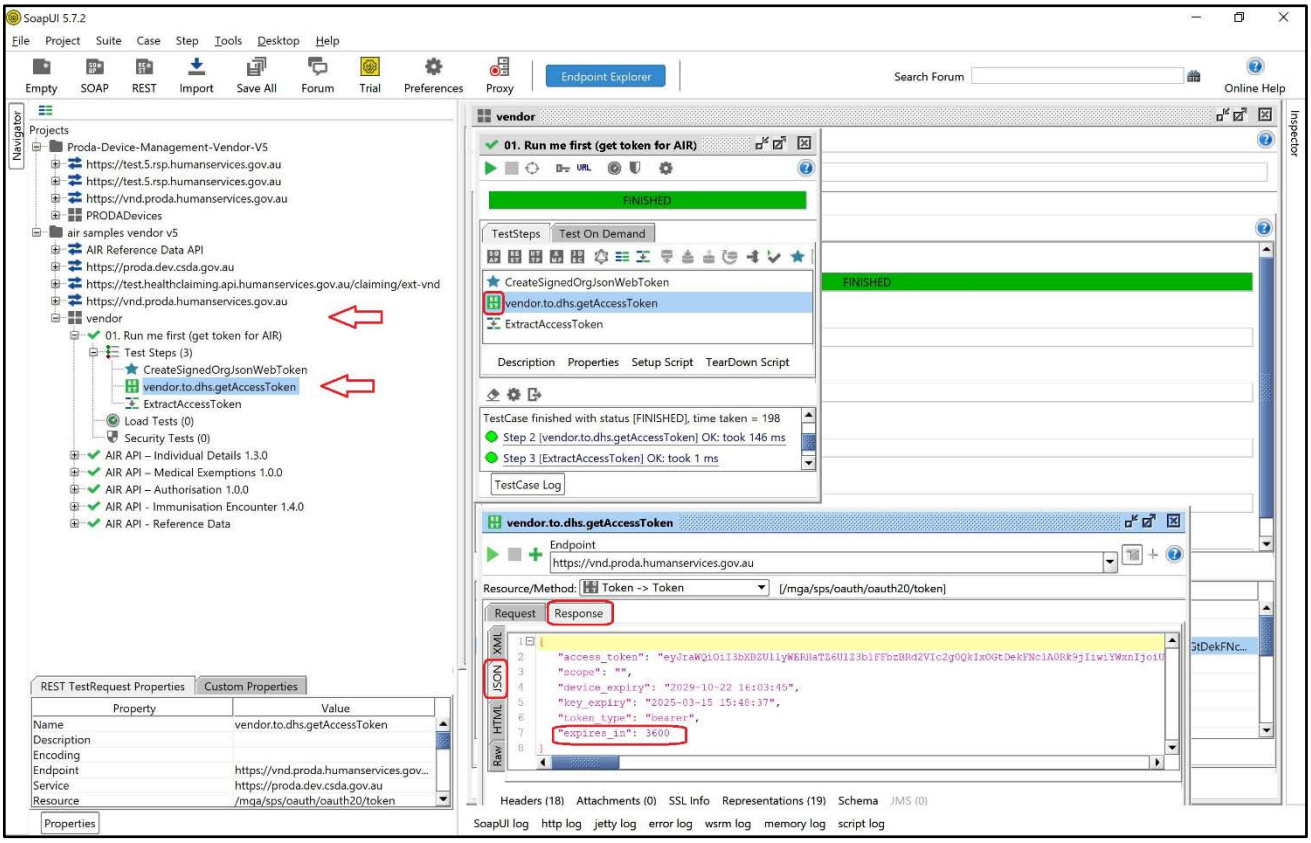
eyJraWQiOiI3bXBZUllyWERHaTZ6UlZ3b1FFbzBRd2VIc2g0QkIxOGtDekFNc1A0Rk9jIiwiYWxnIjoiUlMyNTYifQ.eyJzdWIiOiI2NjU2MjAwMTczIiwiYXVkIjoiUFJPREEuVU5BVFRFTkRFRC5CMkIiLCJwcm9kYS5zd2luc3QiOiJKb2huU21pdGhzVGVzdGluZ0NvbXB1dGVyIiwicHJvZGEudHlwZSI6IlVOQVRURU5ERUQuQjJCIiwicHJvZGEub3JnIjoiNjY1NjIwMDE3MyIsInByb2RhLnJwIjoiTUNPTCIsInByb2RhLnNwIjpbIk1DT0wiXSwicHJvZGEuYXVkIjoiaHR0cHM6Ly9tZWRpY2FyZWF1c3RyYWxpYS5nb3YuYXUvTUNPTCIsImlzcyI6Imh0dHBzOi8vcHJvZGEuaHVtYW5zZXJ2aWNlcy5nb3YuYXUiLCJpYXQiOjE3MjQ2Mzk4MTksImV4cCI6MTcyNDY0MzQxOX0.dosDtODaKz4ot0ZPczp9k1m4LDw1pj-EfnzEsH-RATWeaEZyErbZ_N6kwu2qSDdMpZFOM4TsnqEFGVssTyuRhHFoLTW6WZ0-O_SQuJ7oSVjEV4Q7Q7yE6pBTnEqfl_epwz0-XbXbFQi8dIrqFpz0MIA15c066HuB5RgcMu1pBDFxgFmL7eDjCUUHpTRzRjGothpKDhIzYMGiuFixfuef2h57iP4YbNyq1UDM5VMTiFG0BLfVBdGy8bXz_9deam7X99TQ-E6PpGRu6hEwMlbbyzdTmvMOsMMxO9zLfFUjfPn6znfUkhPA2cQn23129mj0j-_HSdqPGhCcueFFTZLRNw

When the token is successfully retrieved, the response from PRODA will contain various expiry dates\times.

The Developer needs to extract this information. (See below)

Access Token - Key Expiry

Note: The access token will expire as per the value shown in the JSON response. – see example below



For example in the image above the field in the Json response is:

- “expires_in”: 3600

The “expires_in” is expressed in seconds. Therefore in the example shown above this access token is expiring in 60 minutes.

I.e. 3600 seconds = 60 minutes.

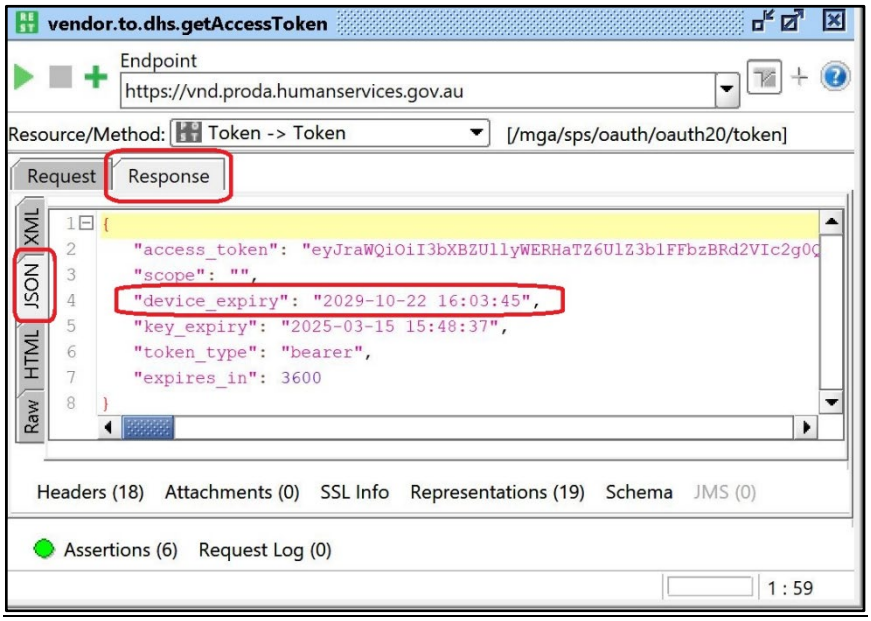
On successful verification of an Access Token request the PRODA authorisation service will return an OAuth 2.0 Access Token, which can be then used in a request to a Service Provider API in the Authorization header of the HTTP request.

A token can live up to 60 minutes, although this may depend on the security requirements of the Service Provider. When it expires, the software is required to obtain another Access Token using the same service. The Access Token expiry time is included in the Access Token response. Write your code to anticipate the possibility that the token might no longer work. We suggest tracking expiration time and requesting a new token before it expires, rather than handling a token expiration error.

Therefore, software developers should write code to keep track of the expiry time of the token and request a new token before the expiry time is reached. The token expiry time can be determined in one of two ways:

- *either by decoding the JWT access token to obtain the ‘exp’ (expiry) value, which is a Unix timestamp format, or*
- *by adding the “expires_in” seconds from the token request’s response to the time of the request.*

Device Expiry



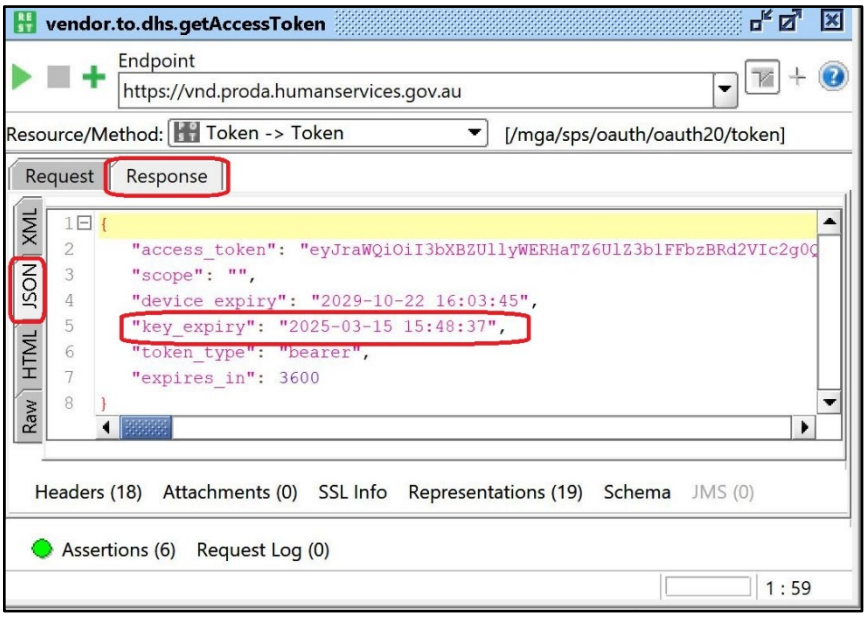
Also shown in the sample response image above is the device_expiry, which is the date and time that the Software Instance (or Device) will expire and no longer be able to request access tokens from PRODA. To ensure business continuity the Software Instance must be refreshed by an authorised Organisational user via the PRODA web site before this expiry timestamp to prevent deactivation, causing disruption to the Organisation's business. Use this value to determine when the device needs to generate a message to the user that the device needs to be refreshed in PRODA by an authorised user.

Keep track of the device's expiry, and notify the end user of impending expiry:

- 30 days before expiry
- 7 days before expiry
- Every day from 7 days till expiry occurs

Once device has been refreshed and expiry time is reset, messages should no longer generated until next impending expiry event is trigger.

Key Expiry



Also shown in the sample response image above is the key_expiry, which is the date and time that the Software Instance's signing key will expire. The Software Instance must refresh the signing key before it expires, otherwise an authorised Organisational user will be required to reactivate the Software Instance via the PRODA web site, causing disruption to the Organisation's business. Use this value to determine when the device needs to generate a new key and submit it to PRODA.

Note: For further information on the PRODA development requirements, please refer to the following documents on the Health Systems Developers' Portal:

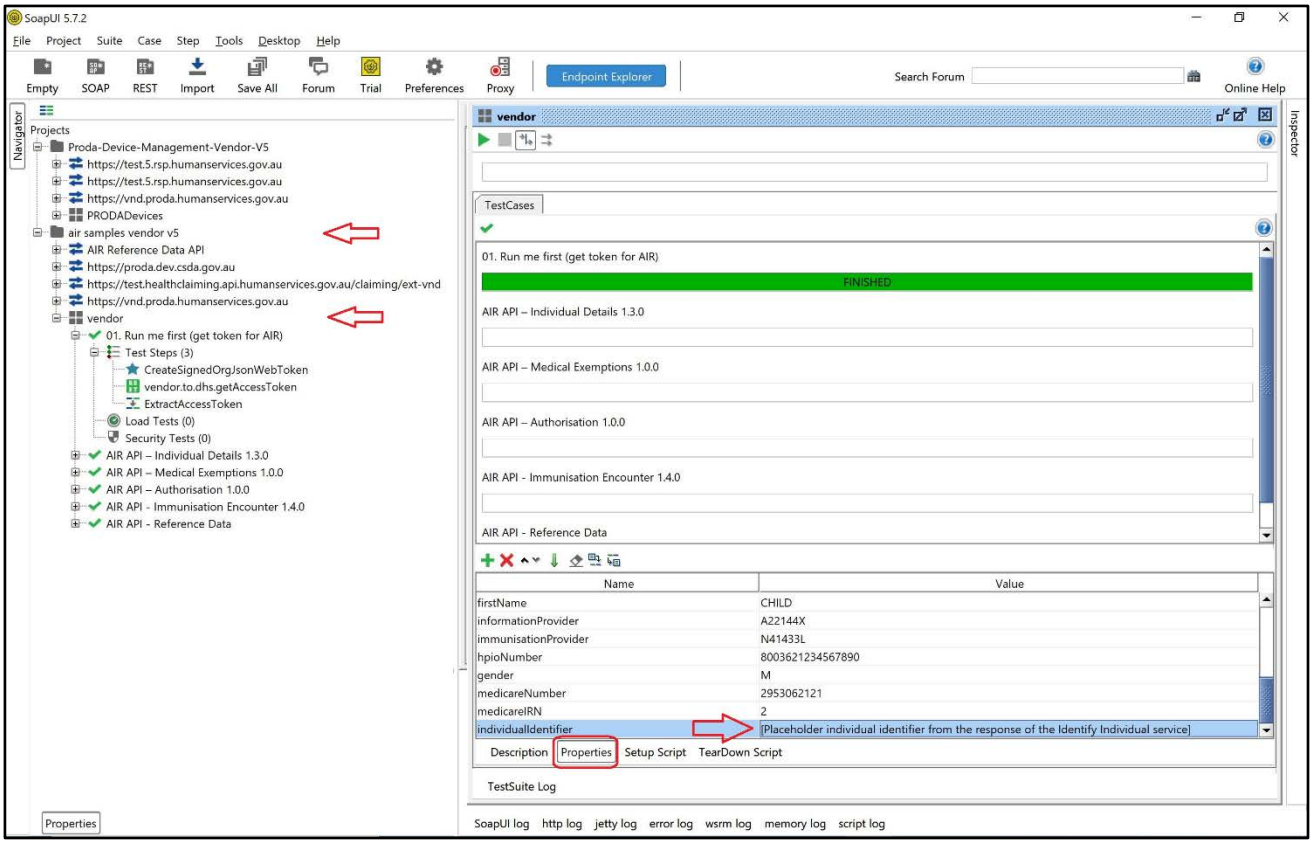
- PRODA B2B Software Developers Guide
- PRODA B2B High Level - 3rd Party Vendors
- B2B Best Practice - Developers Guide

Please see the following extracts below from the 'PRODA B2B Software Developers Guide.pdf' document.

4	Refresh Device Public Key	Vendor Software	The software instance generates a new RSA key-pair and uploads the public key as a JWK to PRODA. **Important Information:** The Vendor Software must regularly refresh the public key within the time parameters[1]. If the public key is not refreshed within this timeframe, the software instance will be unable to gain authorisation from PRODA to access Service Provider services.

B) Updating the properties for the Identify Individual web service request

Once you have entered the values in the properties above, the next step is to send the' Identify Individual' web service request. Double Click the 'vendor' test suite, click the 'Properties' toggle at the bottom of the page and scroll to the last data item. I.e. individualIdentifier



The following will be displayed:

- [Placeholder individual identifier from the response of Identify Individual service]

For the purposes of simulating the web services update the 'individualIdentifier' to a number.

In this case update it to 1.

The 'Identify Individual 1.1' web service is included in the 'AIR API – Individual Details 1.3' test suite. (See section E)

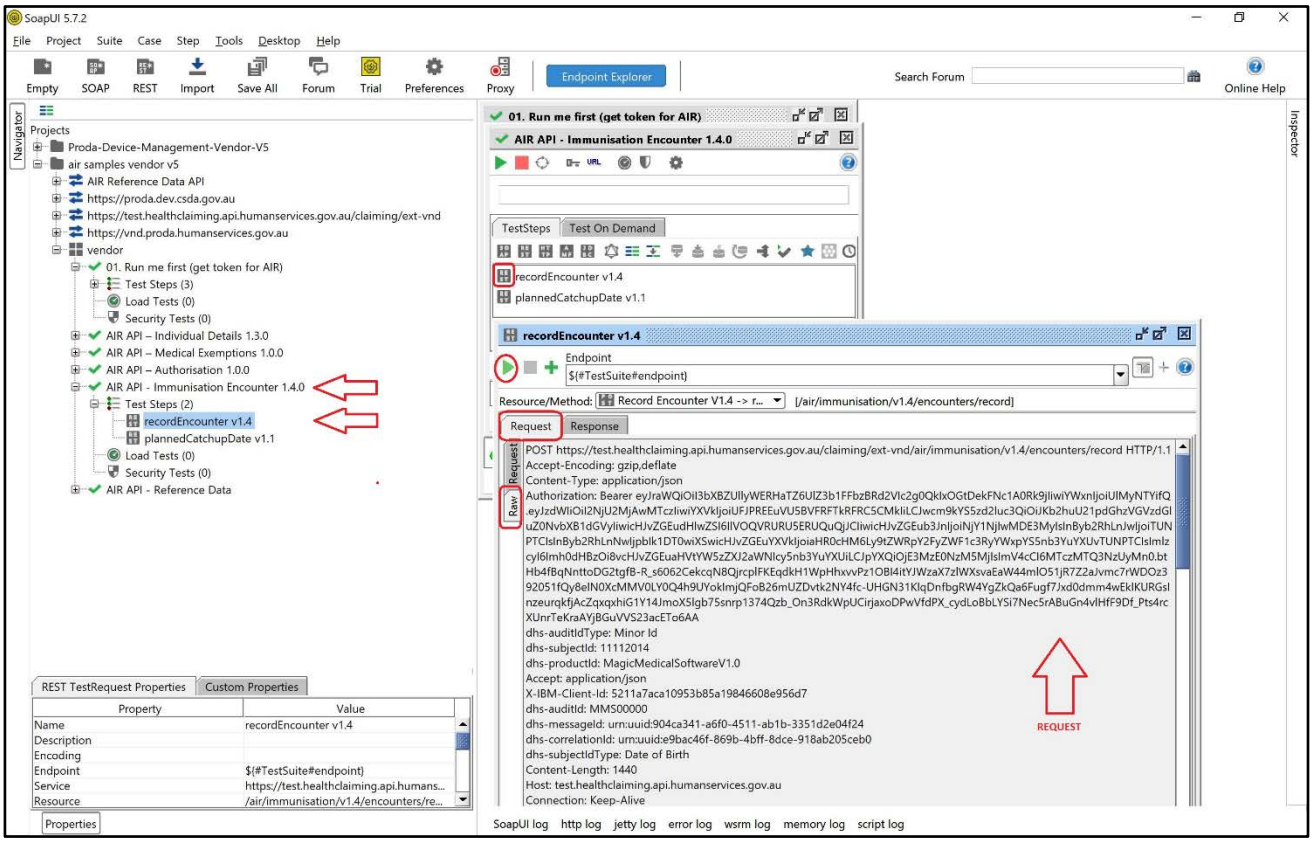
Select File > Save All Projects.

C) Sending an AIR Record Encounter web service request

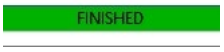
The next step is to send a 'recordEncounter' web service request.

Expand the vendor test suite and double click the 'AIR API - Immunisation Encounter 1.4.0' test step.

Select the 'recordEncounter v1.4' test step.



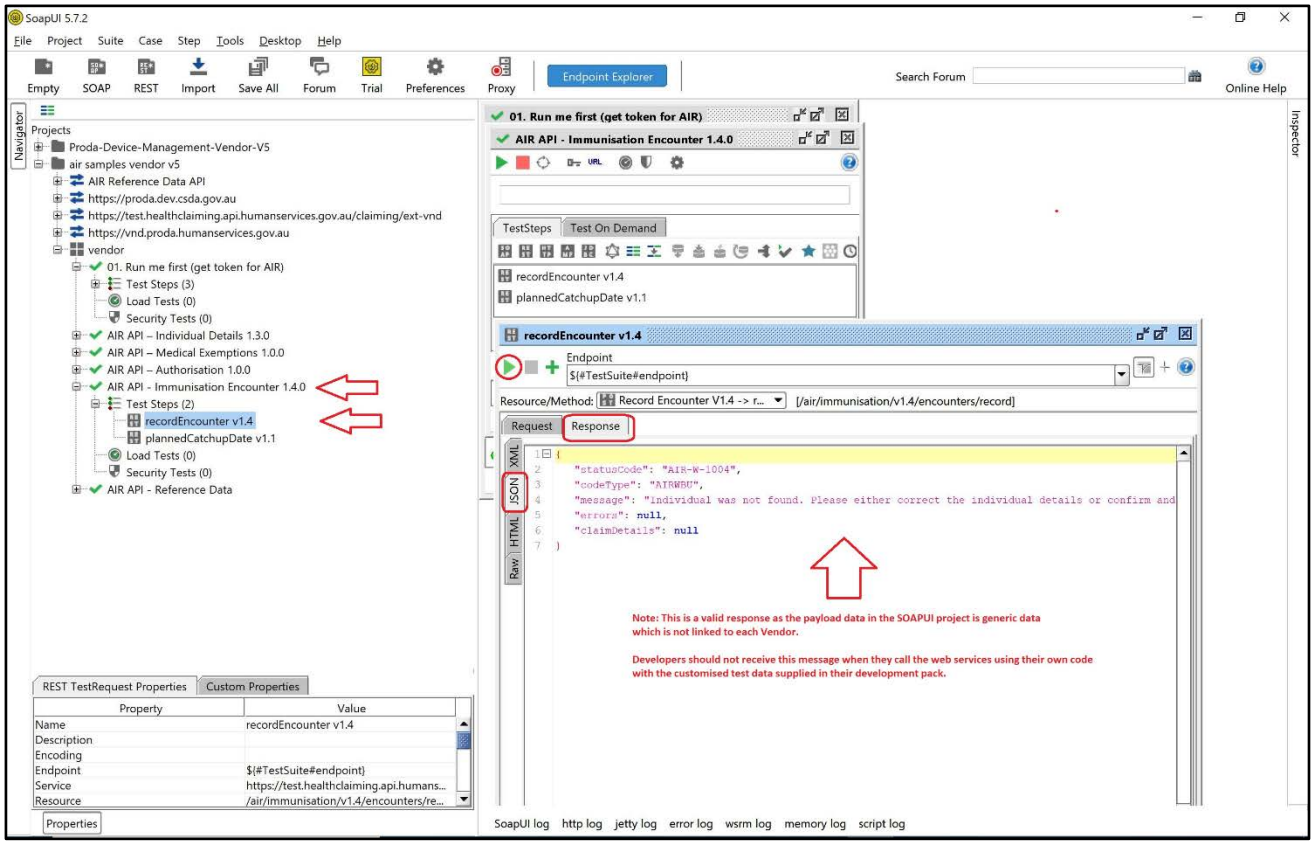
Modify the JSON request as required and press the play button  to submit the request.

The test should run correctly. I.e. 

You can see the request and response data for the REST calls by selecting the 'Raw' or 'JSON' tabs.

See the sample Raw request (above) for the 'recordEncounter v1.4' transaction.

See the JSON response (below) for the ‘recordEncounter v1.4’ transaction.

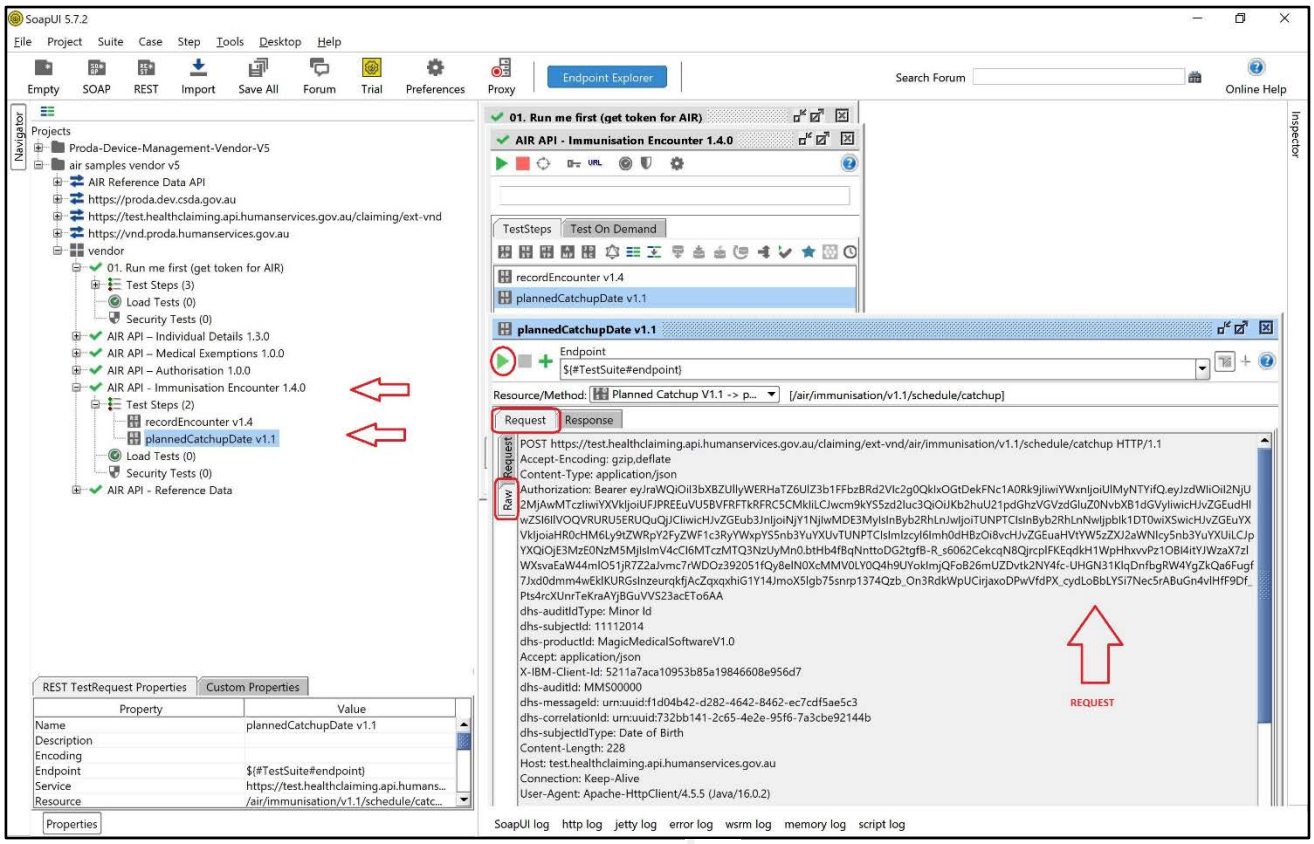


D) Sending an AIR Planned Catch-up Date web service request

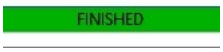
The next step is to send a 'plannedCatchupDate' web service request.

Expand the vendor test suite and double click the 'AIR API - Immunisation Encounter 1.4.0' test step.

Select the 'plannedCatchUpDate v1.1' test step.



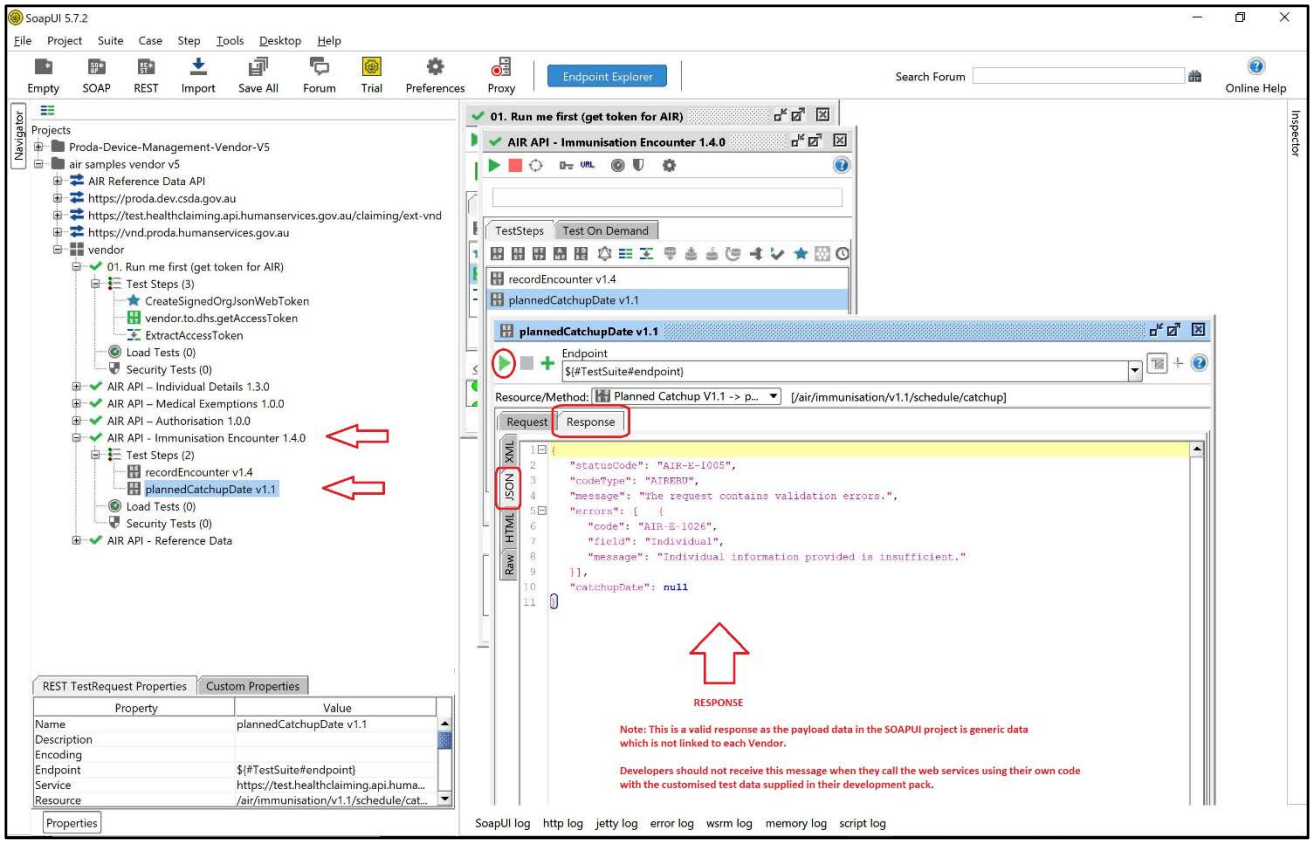
Modify the JSON request as required and press the play button  to submit the request.

The test should run correctly. I.e. 

You can see the request and response data for the REST calls by selecting the 'Raw' or 'JSON' tabs.

See the sample Raw request (above) for the 'plannedCatchUpDate v1.1' transaction.

See the JSON response (below) for the ‘plannedCatchUpDate v1.1’ transaction.



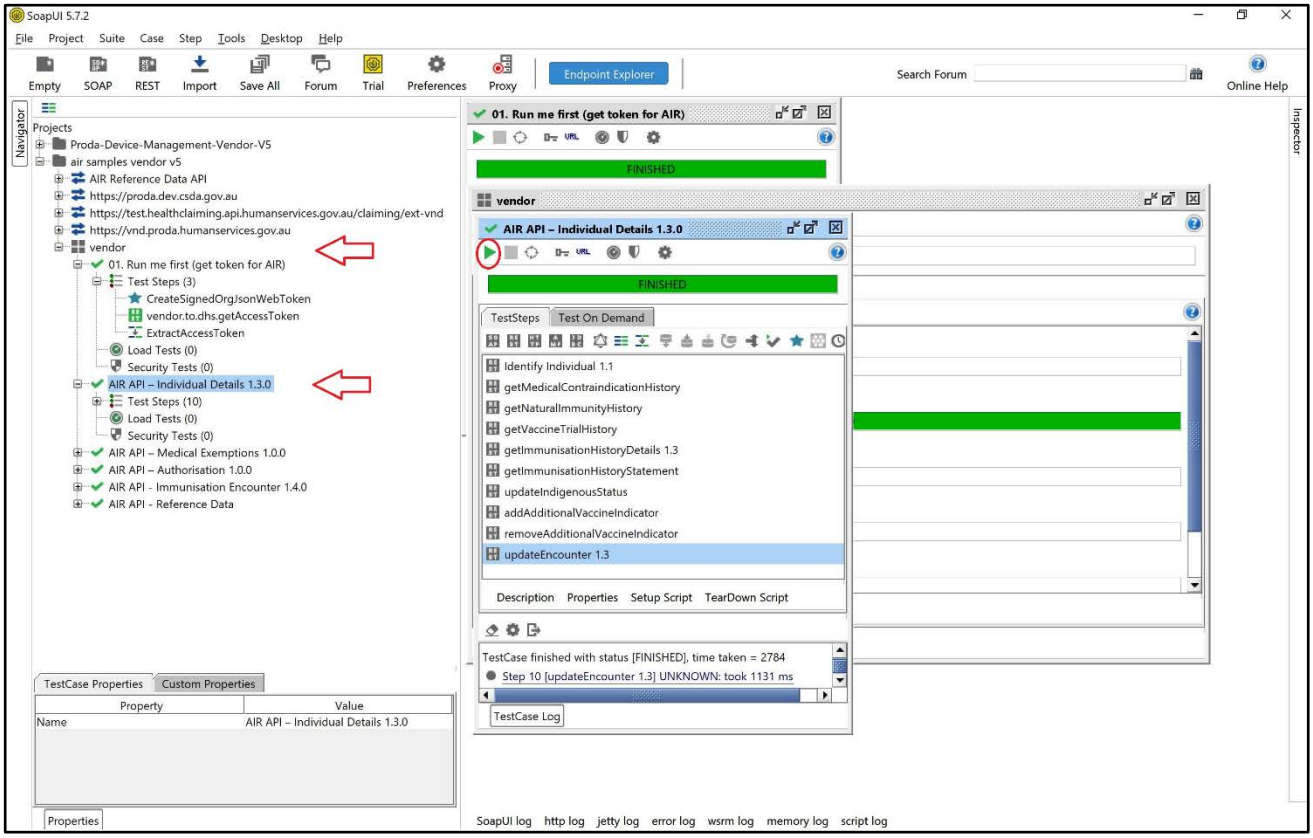
E) Sending an AIR Identify Individual web service request and other web services.

Sending the other web service requests.

There are 30 other web services contained under the following test suites:

- AIR API – Individual Details 1.3.0
- AIR API – Medical Exemptions 1.0.0
- AIR API – Authorisation 1.0.0
- AIR API - Reference Data

Expand the vendor test suite and double click the 'AIR API – Individual Details 1.3.0 test suite.



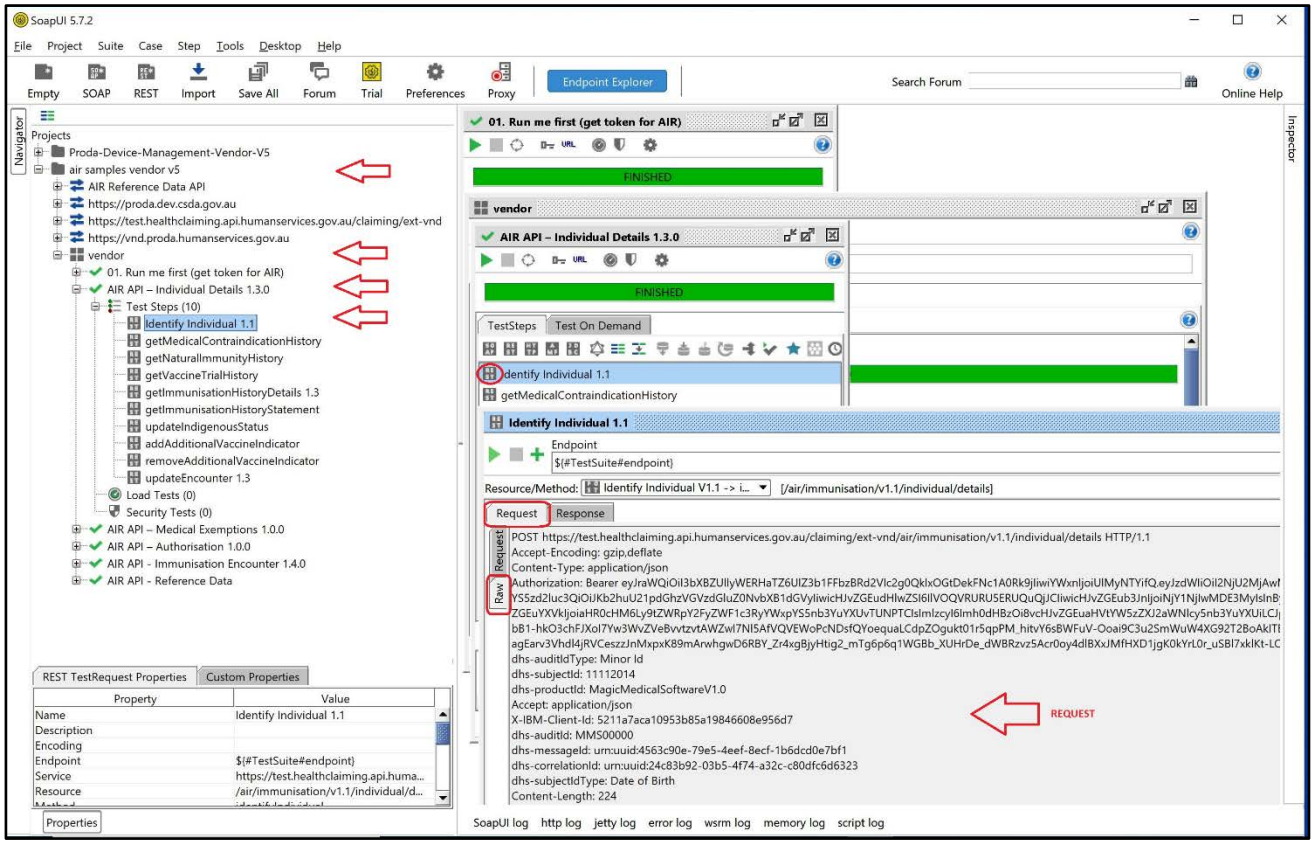
Modify the JSON request as required and press the play button  to submit the request.

The 10 tests should run correctly. I.e. 

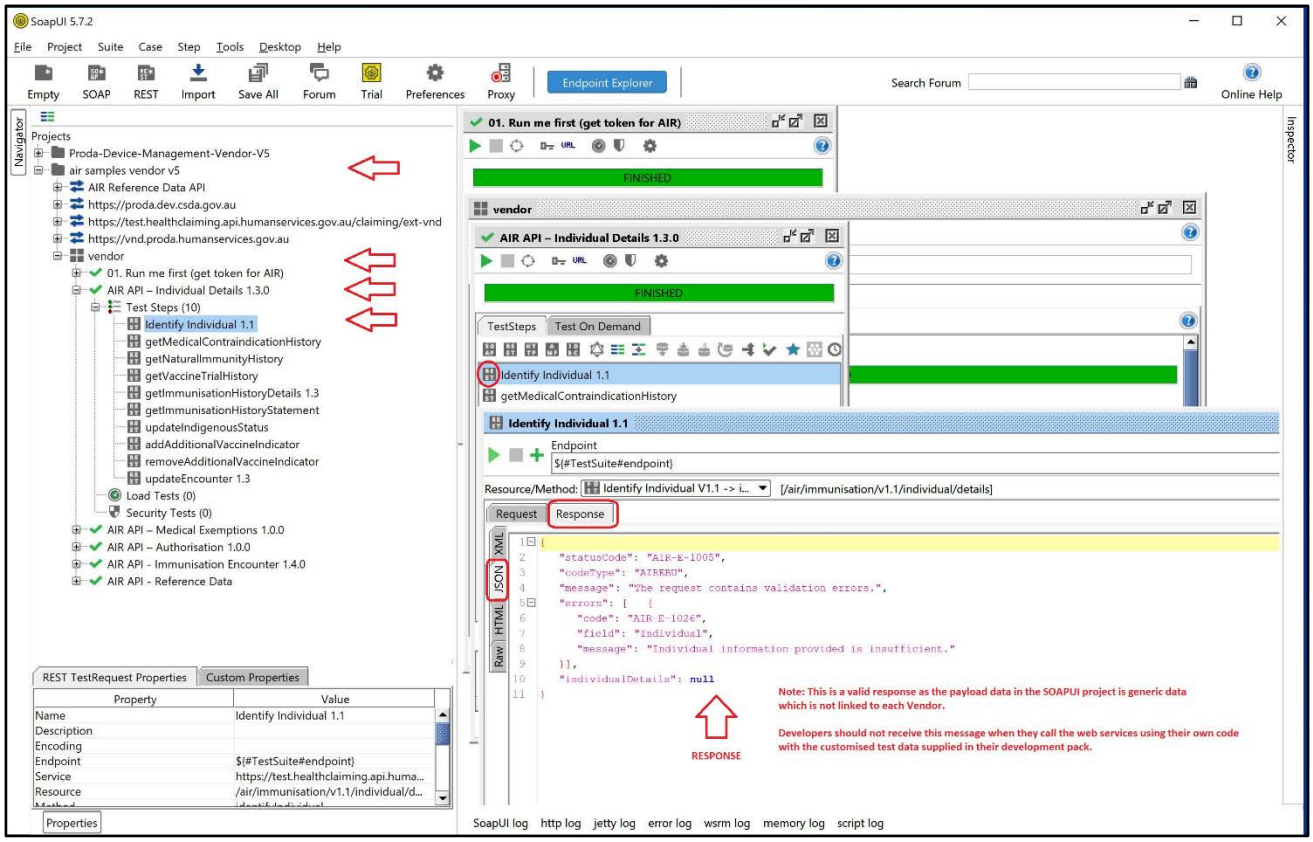
By double clicking on the test step 

, you can see the request and response data for each individual REST call, by selecting the Raw tab on the Request tab and the Json tab on the Response tab.

For example, see the Request below for the 'Identify Individual 1.1' web service below;



For example, see the example Response below for the 'Identify Individual 1.1' web service below;



Using the same steps as above to run the remaining 20 web services under the following test suites:

- AIR API – Medical Exemptions 1.0.0
- AIR API – Authorisation 1.0.0
- AIR API - Reference Data

Important Note: The AIR SoapUI project is supplied to Software Developers to assist in their development. The same SoapUI project is supplied to all Software Developers, with the following underlying generic populated payload data contained in the properties of the project:

- dateOfBirth
- familyName
- firstName
- informationProvider
- immunisationProvider
- providerNumber
- hpioNumber
- gender
- medicareNumber
- medicareIRN

Therefore, when calling each web service using the SoapUI project you may receive information messages in the JSON response advising that the web service was not completed, such as:

- "statusCode":"AIR-E-1005","codeType":"AIREBU","message":"The request contains validation errors.","errors": [{"code":"AIR-E-1026","field":"Individual","message":"Individual information provided is insufficient."}],"catchupDate": null
- "statusCode": "AIR-W-1004", "codeType": "AIRWBU", "message": "Individual was not found. Please either correct the individual details or confirm and accept individual details are correct.",
- "errors": null, "claimDetails": null }"statusCode": "AIR-E-1005", "codeType": "AIREBU", "message": "The request contains validation errors.", "errors": [{ "code": "AIR-E-1061", "field": "Individual Identifier", "message": "Individual Identifier is invalid or has expired."
- "statusCode": "AIR-E-1005", "codeType": "AIREBU", "message": "The request contains validation errors.", "errors": ["code": "AIR-E-1063", "field": "Provider Number", "message": "Information provider A22144X is not authorised to use this service."

The responses returned are **valid responses,** confirming that you have connectivity to the Vendor Environment.

However Software Developers should not receive these messages when they call each web service using their developed product, as they should be using the customised data previously supplied in their development pack by the Developer Liaison Team (developerliaison@servicesaustralia.gov.au). This customised data is specially prepared for each Software Developer with the necessary linkages and dependencies.

Note: Software Developers with technical queries can email them to onlineclaiming@servicesaustralia.gov.au