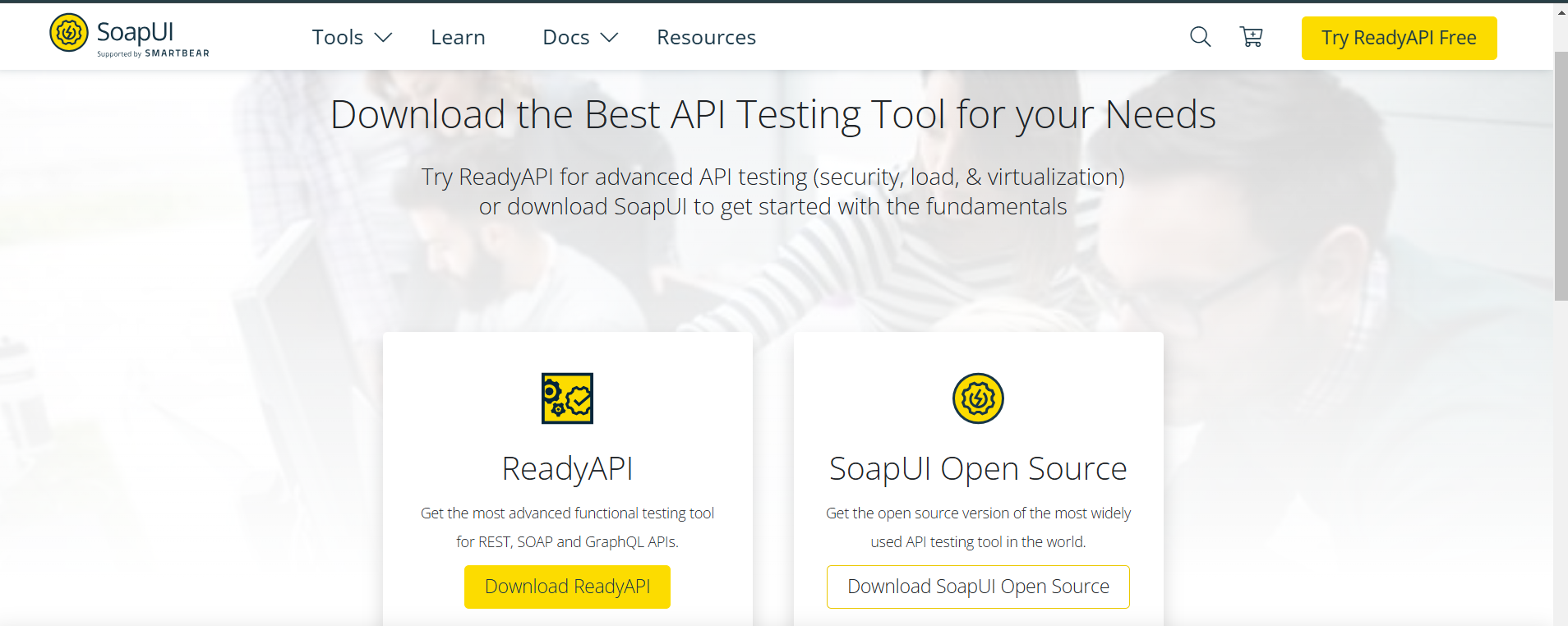
**Week 4**

**Fajar Abdillah Ahmad**

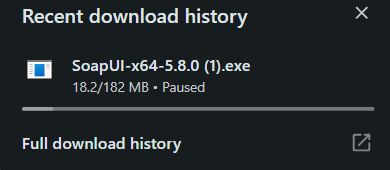
**INSTALL AND SETUP**

1. Open a browser and visit the official SoapUI website https://www.soapui.org/downloads/latest-release/
2. Choice the SoapUi version:



* 1. Open source version(free)
  2. SoapUi Pro (paid version with more features)

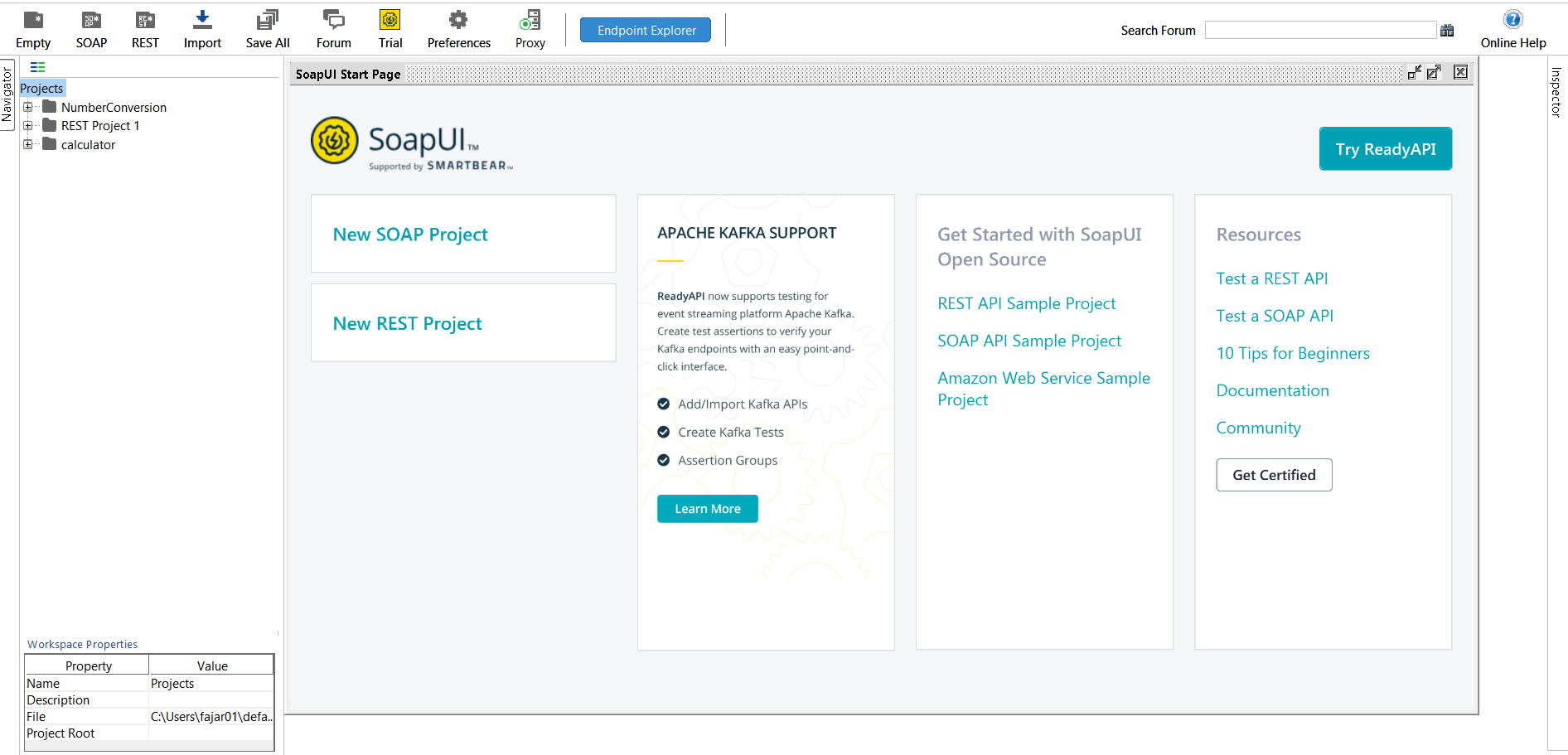
1. Running installer



1. Installation procces

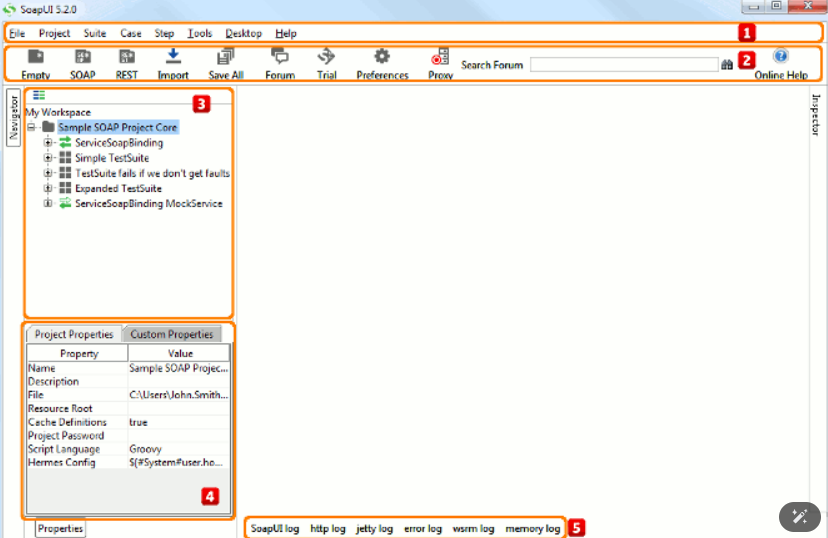
* Click Next on the welcome screen.
* Accept the License Agreement by clicking I Agree.
* Select an installation location or leave the default, then click Next.
* Select the additional components you want to install, such as SoapUI Tutorials, Source, HermitJMS, and others (optional).
* Select the Start Menu location (leave the default or change it as needed).
* Click Next and then Install.

1. Running SoapUi



in this section the application has been downloaded and is ready to run

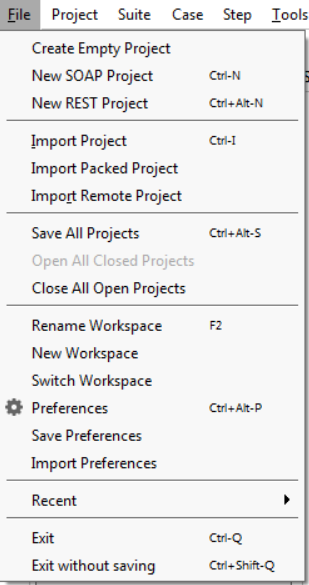
**SOAPUI STRUCTUR**

****

Below is a table explaining each part

|  |  |  |
| --- | --- | --- |
| **No** | **Opinion** | **Description** |
| 1 | Main tolbar | Main toolbar |
| 2 | Icons toolbar | Icon shortcuts toolbar |
| 3 | Navigator Panen | Expandable tree display of worspace content |
| 4 | Properties Pane | Lists available predefined workspace properties |
| 5 | Log Inspectors | Available log inspectors in main window |

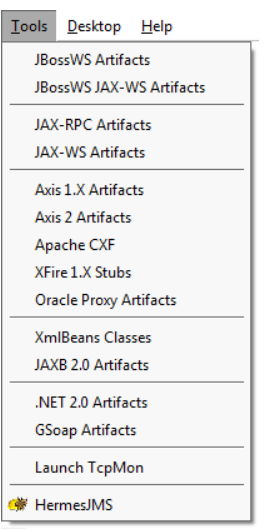
Below is table for the menu file



|  |  |
| --- | --- |
| Option | Description |
| New ReadyAPIject | Creates a new ReadyAPIject in this workspace |
| Import Project | Adds an existing project into this workspace |
| Import Remote Project | Imports a remote project into this workspace |
| Save All Projects | Saves all projects in the current workspace |
| Open All Closed Projects | Opens all closed projects in the current workspace |
| Close All Opened Projects | Closes all opened projects in the current workspace |
| Rename | Renames this workspace |
| New Workspace | Creates a new workspace |
| Switch Workspaces | Loads another workspace file |
| Clear Workspace | Removes all projects from the workspace |
| Preferences | Sets global soapUI preferences |
| Save Preferences | Saves all global preferences |
| Import Preferences | Imports soapUI settings from another settings -file |
| [Recent](https://www.soapui.org/working-with-soapui/reference/main-window.html) | Reopens recent items |

Below is table for tools menu

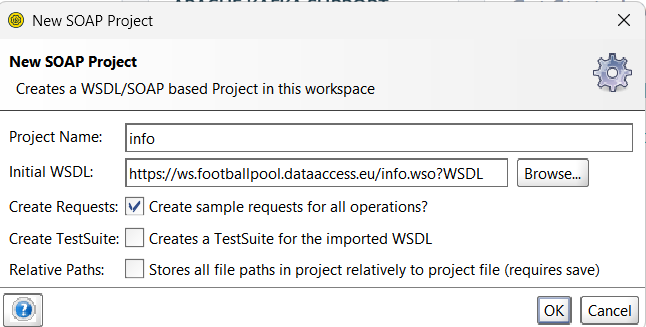
|  |  |
| --- | --- |
| Option | Description |
| JBossWS Artifacts | Generates JBoss WS artifacts using the JBoss wstools utility |
| JBossWS JAX-WS Artifacts | Generates JBossWS JAX-WS artifacts using JBoss WS wsconsume |
| JAX-RPC Artifacts | Generates JAX-RPC artifacts using wscompile |
| JAX-WS Artifacts | Generates JAX-WS artifacts using wsimport |
| Apache CXF | Generates Apache CXF using WSDL2Java utility |
| XFire 1.X Stubs | Generates XFire 1.X stubs using wsgen utility |
| Oracle Proxy Artifacts | Generates Oracle Proxy artifacts using the wsa.jar utility |
| XmlBeans Classes | Generates XmlBeans classes |



**CREATE NEW PROJECT**

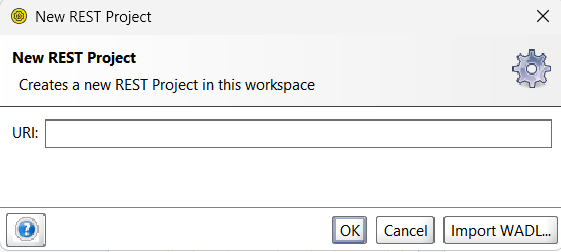
* To create a new *SOAP project*, select **File > New SOAP Project**.

Specify the name for your new project and the WSDL file that SoapUI will use for the initial configuration, then select the necessary options.



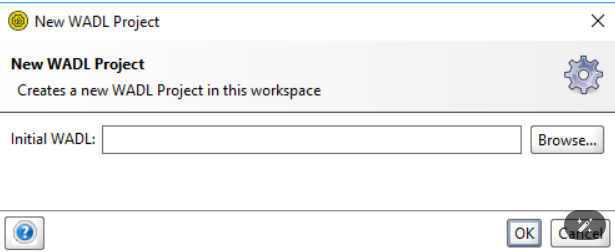
Click **OK**. SoapUI will generate the service and, optionally, test suite and mock service.

* To create a new *REST project*, select **File > New REST project**.



Specify an URI to use for the project generation in the subsequent dialog and click **OK**.

You can also click **Import WADL** to switch to the **New WADL project** dialog.

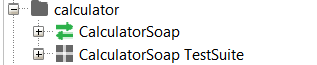


* To create a new empty project, select File > Create Empty Project. It will immediately appear in the Navigator, and the editor for the project will become available in the Workspace.

**TEST CASE AND ASSERTIONS**

In SoapUI, there are several components in testing. Each test has its own definition and materials to be tested. Here are some tests that can be performed using SoapUI.

1. Test Suite



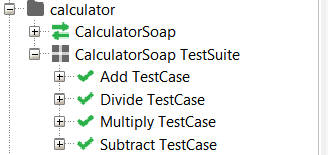
Test Suite is a collection of several Test Cases used to group tests based on specific modules or features. Used to group tests based on specific features or API services. For example, in testing the banking system, we could have a Test Suite "Fund Transfer" which contains several Test Cases such as:

Test Case 1: Transfer to a valid account.

Test Case 2: Transfer to an invalid account.

Test Case 3: Transfer with insufficient funds.

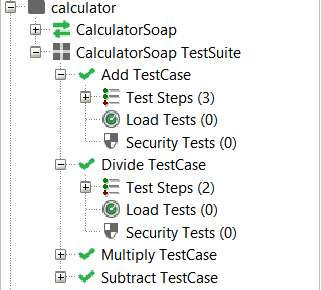
1. Test Case



A Test Case is a collection of Test Steps used to test a specific scenario in an API service. Used Manage the execution sequence of Test Steps in API testing. In the Test Case 'Transfer to a Valid Account', the steps taken may include:

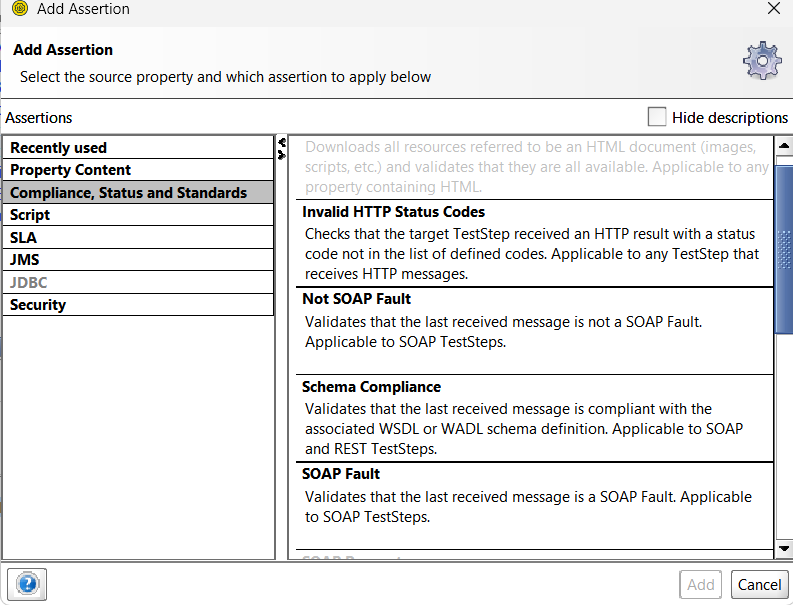
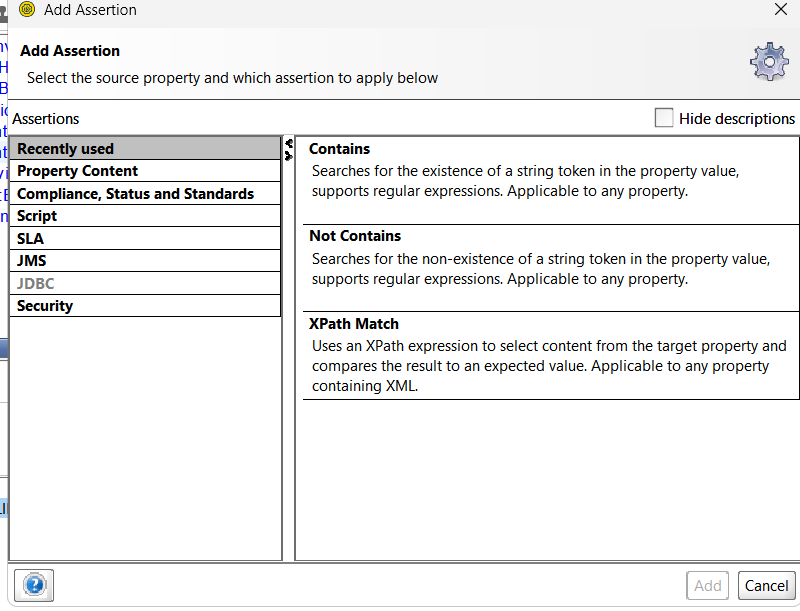
* Sending an API request to transfer funds.
* Verifying the response (whether successful or failed).
* Performing additional validation, for example, checking the balance after the transfer.

1. Test Step

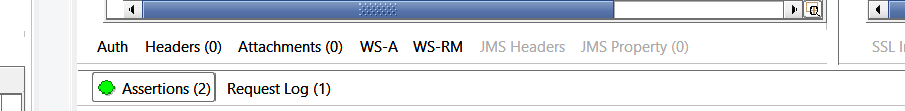


Test Step is the individual steps in a Test Case that define specific actions to be taken. Purpose to perform specific operations in testing, such as sending requests, conducting validations, or accessing the database.

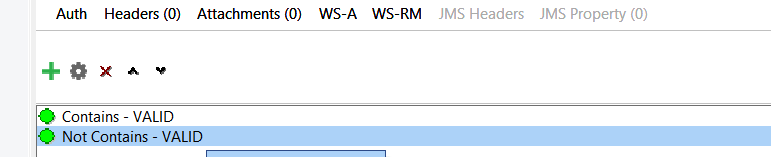
Assertions in SoapUI are used to verify that the response received from the API meets expectations. Assertions help ensure that the API operates correctly based on the specified conditions. Below some assertions available in SoapUi:



for the start assertion in soapui we can klik text assertion then klik add button, if finish assertion l’ll display in assertion window, below some example add assertion.



then if you have assertion, soapui can display to you

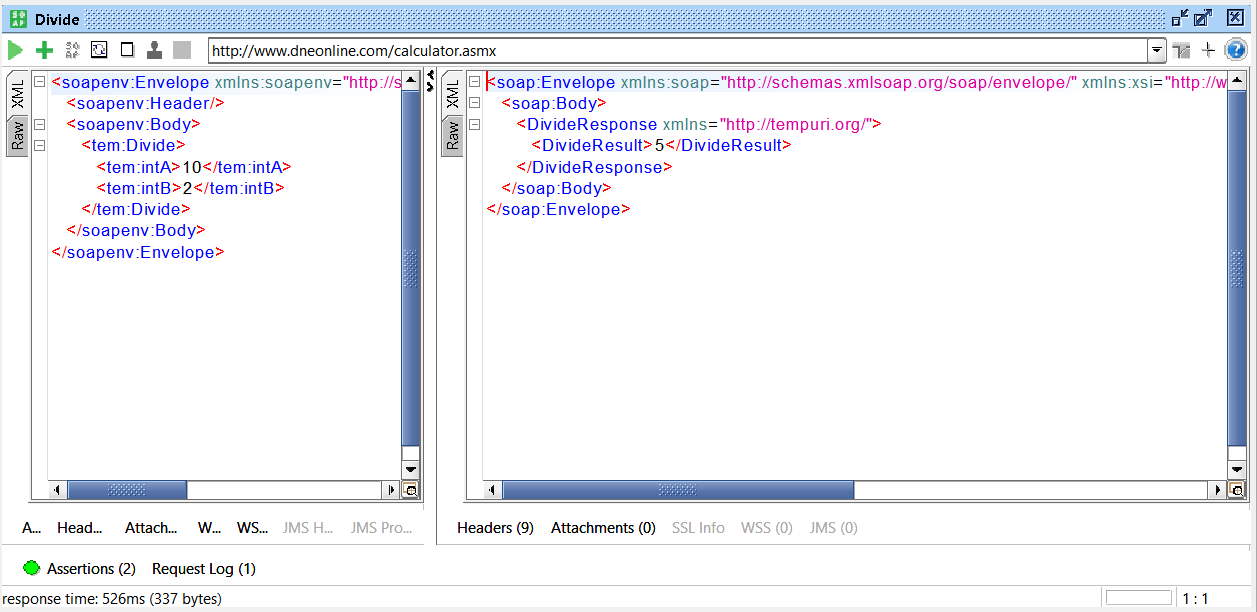


**AUTOMATION AND SCRIPTING**

Data-driven testing is when you store test data (inputs, requests, parameters, results, etc.) in some external storage (spreadsheet, XML, etc) and then use that data to drive them automatically. For example, to test your phone lookup service, you might have a list of names and their related phone numbers in a spreadsheet. You can hook up the spreadsheet to your test so you can use that data to 'drive' your tests, checking that each name gets the right phone number back.

Groovy is a dynamic Java-based programming language that runs on the Java Virtual Machine (JVM). Groovy has a more concise and flexible syntax than Java, and supports scripting, functional programming, and meta-programming. Groovy Script is Groovy-based code that can be used for automation, data manipulation, and testing in a variety of Java-based applications, including within the SOAP UI for API testing. below some example for testing with grovvy:

this is example request to divide number, and this successfully



after that using grovvy for the testing with programming syntax, below example for testing divide request with grovvy.

