Code Structure API and functionalities Explanation

Our project consists of service provider and service consumer and the API. Both service consumer and provider are implemented in two different languages python and java, respectively. We used SOAP protocol to establish a connection between the two.

1. Service Provider

Our provider consists of two files *ServiceImpl.java* and *Provider.java*

Our *ServiceImpl.java* has the actual business implementation and it consists of three functions GetOsName(), Reboot(), and Screenshot() and it was implemented by java.

We do not care about how the connection is established or how to pass the results. We care about the input of our function and the returned result, the connection and how result are sent is handled by the SOAP web services.

To get connected and use the functionalities of the Provider the *Provider.java* publishes the implementation under the location http://localhost:9000/ServiceImpl to respond to the consumer invocations which is turned will be accessed by the WSDL file that will describe the functions and the return types of the provider.

2. Service Consumer

Our consumer was implemented using javascript. The client doesn't care about how the connection is established or how the results returned, it cares about what is returned from a method invocation.

Moreover, the calls for our functions were done async to the service provider using JavaScript concurrency model. We used promise based async calls.

The consumer calls the WSDL file to know the functionalities offered by the service provider and the function name and the type of data returned. The consumer has a fake implementation, and it is invoked by calling the WSDL file using

```
let url = 'http://localhost:9000/ServiceImpl?wsdl'
```

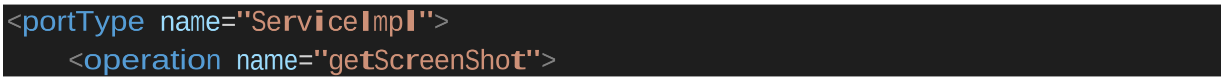
The invocations our functionalities which is getOsName, reboot, getScreenshotis done by referring the serviceimpl which is the fake implementation in our consumer side.

Our javascript was excuted using node run to generate the json package.

3. Service API

Our service API which is our WSDL file contains the definition of our three functionalities and how to get the returned values from the provider. The WSDL file refers to the location of the implementation published by the provider. The WSDL file has what are the methods that can be called, the arguments and the return type.

An example of that would be this line:



```
<portType name="ServiceImpl">
    <operation name="getScreenShot">
```

```
      <input wsam:Action="http://provider.homework2.rpc.csc3374.sse.aui.ma/ServiceImpl/getScreenShotRequest" message="tns:getScreenShot"/>
      <output wsam:Action="http://provider.homework2.rpc.csc3374.sse.aui.ma/ServiceImpl/getScreenShotResponse" message="tns:getScreenShotResponse"/>
    </operation>
```

It contains the name of the file which is 'ServiceImpl" and the name of the function which is "GetScreenShot" and how we can get hold of that function to operation through an http link.

Our WSDL file is an XML based, it can be read by any language.

How to use the code:

These instructions will help you get a local copy of the project up and running for testing purposes.

Installation: We used VS code, java installation package, python language, and gradle to support a Multilanguage environment to make our program up and running.

The WSDL generation mechanism is not supported by Java 11, so try to install Java 8 and add the following snippet in *build.gradle:*

```
compileJava {
    sourceCompatibility ='1.8'
    targetCompatibility  = '1.8'
}
```

VS code Installation: https://code.visualstudio.com/download

Java 8 Installation : https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html

Python Installation : https://www.python.org/downloads/release/python-3710/

Gradle Installation : https://gradle.org/install/

Running the code :

Open the Homework2CSC3374 file on VS code:

- Build the project using ./gradlew build
- Genrate the wsdl file using "*wsgen -wsdl -cp app/build/classes/java/main/ -d app/build/classes/java/main/ -r app/src/main/resources/ ma.aui.sse.csc3374.rpc.homework2.provider.ServiceImpl* "
- And then run the provider side using ./gradle run

On the other hand, compile the Consumer side using:

- *cd src/main/java/ma/aui/sse/csc3374/rpc/homework2/consumer/consumer/js*
- npm start.