Collaborative Session 1

Tom Materne – a1825150, Devang Shetty – a1894311, Sourav Debnath – a1900755

We decided to simply implement the sum function that was given as an example in the specifications. To setup the project we decided to use Gradle, as it gives an easy template for implementing testing frameworks within Java.

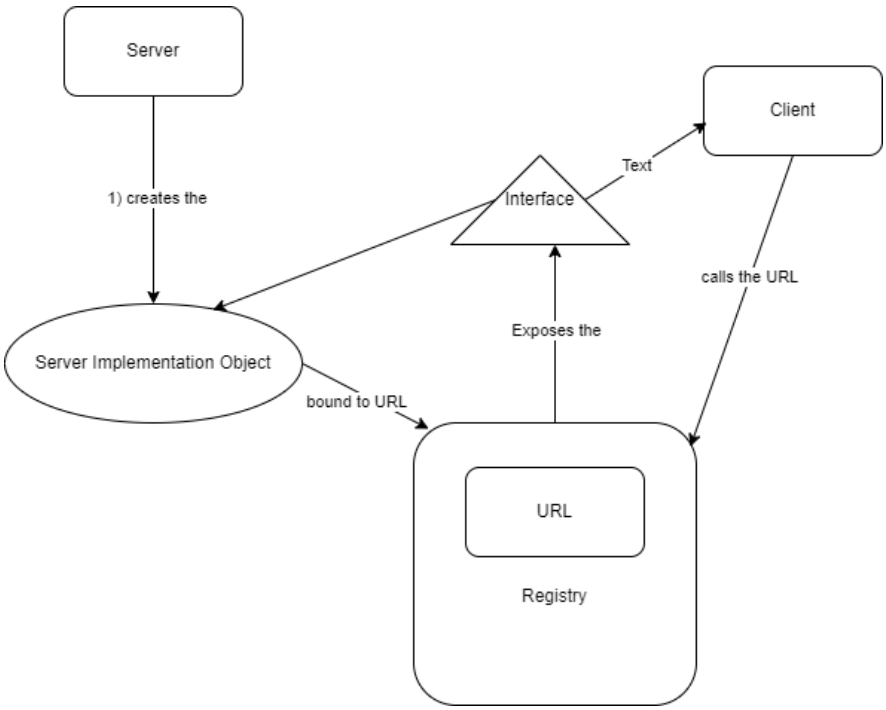
In order to implement the sum functionality we created an Interface with the method int sum(int a, int b) in ServerInterface.java. The server then needed an implementation of this interface, which we named ServerInterface.java, and simply implements the sum function and returns the addition of a and b. Then we needed to create a Server class that would host our implementation of the ServerInterface on the RMI Registry, so we created a registry on port 1099 and bound "//localhost/simple" to our implementation object.

The client then had to simply connect to the server via the URL "//localhost/simple" via the Naming.lookup method exposed in java.rmi package, and then expose the interface's methods within it's own functions. Each of these functions needed to be surrounded by a try/catch block, as the interface is defined as throwing RemoteException.

For the stateful functionality (a stack stored server-side), we implemented two new functions in the ServerInterface: get() and push(int num), with get() returning the entire stack. These methods mutate the state on the server, and this is verified in the testing workflow, as two different clients connect to the server.

For the testing workflow, we simply used Gradle's built-in junit testing suite, and the diagrams are given below for how these tests work. These tests are not comprehensive in that they do not test multiple clients connecting at the same time, but since that is the main problem to be solved in the first assignment, we decided it was out of scope for this collaborative assignment.

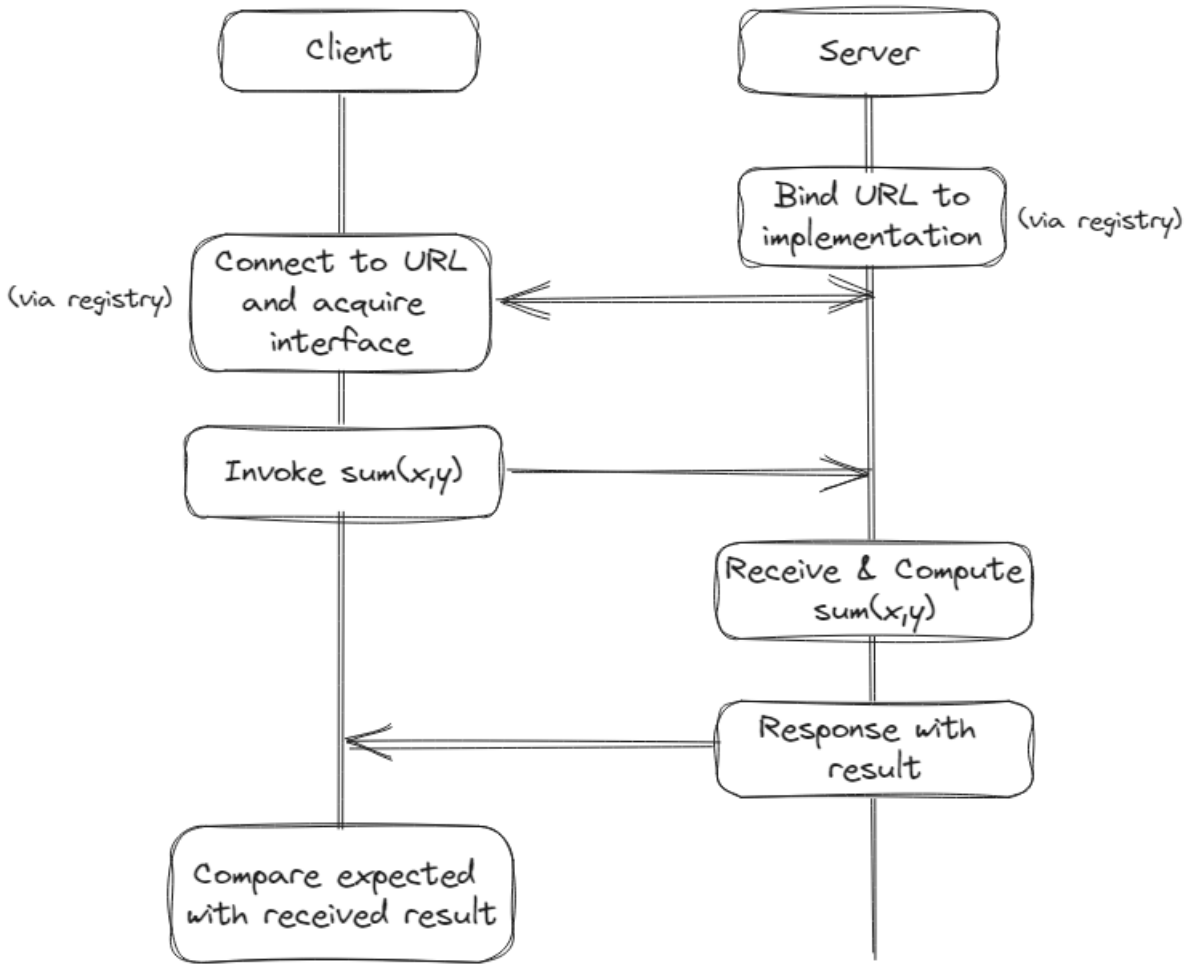
Below is our diagram for how the server & client interact in the application, and on the following pages are the sequence diagrams for the test cases.



Testing Workflow

Simple Method

1) Build & Run Server
2) Run & Execute Client



Stateful Method

1) Build & Run Server
2) Run & Execute Clients