Due: Friday (28th May)

Distributed system analysis and design A3

Aaron Tan (10471321) and Peter (10507429)

# Executive Summary

In this report, we are reviewing and showcasing an implementation of two versions of the Honours Pre-assessment system (HPaS) by using Remote Method Invocation (RMI) as the chosen architecture of these systems. The two system will be a two-tiered structure based system and a three-tiered structure base system.

The development of the system HPaS will be Java based and the development environment used for the creation of the system will be the Eclipse IDE. For the three-tiered application there will be a need for software to act as the database, as such we are using MYSQL as a Database software for saving and retrieving student details required for server to access for the evaluation of a student’s eligibility for the honours program.

This report also aims to outline a brief introduction about what is RMI and why this architecture is important in the construction and consideration of a distributed system. The report will also list out the main application requirements outlined by the client, describing into depth of how this application is going to be developed and implemented and lastly showcasing the user manual documentation that outlines the setup of the application and how to use it.

Contents

[Executive Summary 1](#_Toc72767507)

[Introduction 3](#_Toc72767508)

[Application requirements 4](#_Toc72767509)

[Introduction RMI 6](#_Toc72767510)

[Application design and implementation procedure 7](#_Toc72767511)

[Test Cases 8](#_Toc72767512)

[User Manual 9](#_Toc72767513)

[Two-tiered 9](#_Toc72767514)

[Three-tiered 14](#_Toc72767515)

[Software Installation 14](#_Toc72767516)

[Eclipse Part 1 15](#_Toc72767517)

[MYSQL setup and installation 16](#_Toc72767518)

[Eclipse Part 2 17](#_Toc72767519)

[Conclusion 20](#_Toc72767520)

[References 21](#_Toc72767521)

# Introduction

The Honours Pre-assessment system is to able to assess a student and their qualifications to participate in Honour study. This system will use a distributed system that will be using the remote method invocation (RMI). This system is based on having a client interact with a server-side application where this server to receive the information from the client device of a student academic history and check if the student is qualified to partake in honours study after they have completed their Bachelors.

This report aims to outline the requirements of the Honours Pre-honours systems (two-tier and three-tier) and the remote method of invocation and how this architecture functions and how it is related to this project. The language chosen and other software required for the development is function of the system will also be elaborated on. There will be also the test cases used to test the systems and user manual for the operation of both systems.

# Application requirements

Application requirements for this Honours Evaluation system is to check the student information check if they are able to enrol into the honours program based on their past academic history.

Using their academic history, the Weighted Average Marks (WAM) is created and will be the defining requirement for them to join this program. While there are exceptions where a coordinator can either reject or accept an applicant’s eligibility based on their personal statement or any previous academic achievements that may influence the decision and entry to this program is not solely on the WAM.

The program is to assume that the student being evaluated is applying for an honour program in their respective bachelors’ field of study. This invitation is only open for students who do well in his or her Bachelor program for the entire of their agreed duration (example: 3 years) whether it was full-time or part-time. The following requirements are:

* If the student’s WAM is more or equal to 70, the DS shall process an invitation to an honours program.
* If the student’s WAM is between 65 and 70, the DS is not guaranteed that a student will be admitted but it requires the coordinator approval and the probability that a student will be admitted into honours is good but not certain.
* If the student’s WAM is between 60 to 65, the DS will not produce an invitation, but it will place a special note for a coordinator to extensively review the applicant’s performance.
* If the student’s WAM is lower than 60, the DS guarantees that the applicant is not qualified to admit into the honours program.

There is also one additional requirement not just based on WAM that automatically disqualifies an applicant’s entry into the honours program:

* If the student fails 6 or more units (unit score < 50) during the entire Bachelor course, then the DS automatically not invite the applicant into honours based on fail units. Even if the other units are exceptional (strong High Distinctions), unfortunately, based on the record and the requirements that were set by the University, he or she cannot appeal to this matter.

This program also assumes no supplementary units or any other requirements that are related to ECU’s full requirements based on this assignment.

The following DS setup and the proposed algorithm (flowchart) are shown below:



As you can see in Figure 1, the requirements for eligibility into WAM are similar except that I assume that the program allows error handling exceptions in the case of inputting the wrong unit score or enter the wrong amount of units. Another assumption is that entering a Student ID is irrelevant to this flowchart since a Student ID and saving a student’s details to the database does not make a significant change to the WAM calculation.

# Introduction RMI

An RMI (Remote Method Invocation) is an API that provides remote communication between a server and a client that allows for the creation a distributed system application. RMI is supplied as part of Sun’s Microsystem’s Java Development Kit (JDK). RMI is implemented using three layers:

1. Stub/Skeleton – A stub program in the client side and the skeleton program at the server end. The sub is an object that acts as a gateway (client side), allowing outgoing requests to route outside of the client’s machine. The skeleton is also like an object that acts as a gateway (server side), allowing incoming requests to route inside of the server’s machine. An illustrated diagram shown in *fig 1.* How a stub and skeleton
2. Remote Reference Layer (RRL) – behaves differently depending on the parameters passed by the calling program. For example, this layer determines whether a request is calling a single remote service or multiple remote programs as a multicast.
3. Transport Connection Layer (TCL) – sets up the main communication traffic between server and client from their requests (example: HTTP requests procedure of packaging the packet and decode that HTTP request to the server allowing a three-way handshake to proceed with the request.)

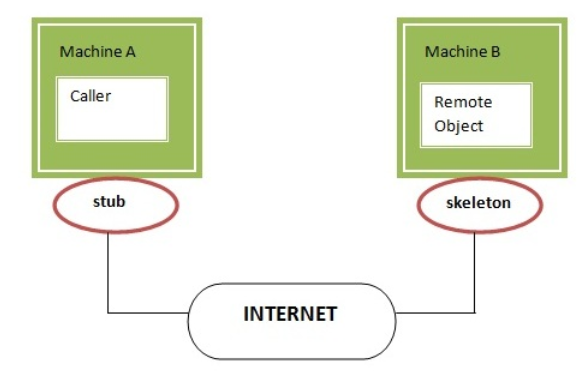
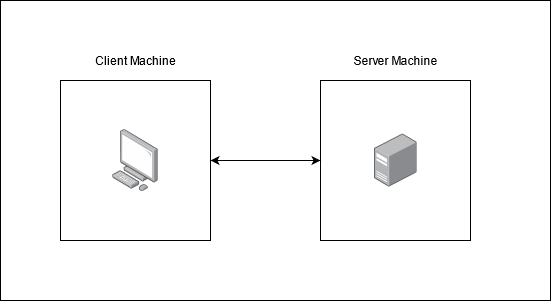


Figure 1 Stub and Skeleton RMI Architecture

# Application design and implementation procedure

  
*Figure 2*



*Figure 3*

In Figure 2, this setup environment that this DS shall be created using Eclipse as an IDE, follow by Java as the main programming language, then for the database application side, I’ll use MYSQL as the main program that resides all the user’s data that will later query its data to display its saved user.

For a distributed system, I created two workspaces one is for the client-side machine application and the other workspace is for server-side that uses the RMI operation. Each workspace contains their on-java files required for this system. The two tiered application in Figure 3 can be used for testing purposes is instead of the three-tiered application that uses a database, we can implement only the client and the server that focuses on the algorithm and software testing (example: white box testing) in the two tiered system for verification before it ready to implement in the three-tier application. That way, a developer is able to focus on the data saving feature and instead of focusing on getting the main algorithmic results to evaluate if the applicant is eligible for honours based on the requirements set by the university when they reach the development of the three-tiered system.

# Test Cases

## Two Tiered System

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Test | Test Method | Expected Result | Real Result |
| 1 | Total number of units lower than 12. | Input 11 into the console and press enter | Console to display incorrect input value. | Console displays  “Invalid input. Please enter only digits.” |
| 2 | Total number of units higher than 30. | Input 31 into the console and press enter | Console to display incorrect input value. | Console displays  “Invalid input. Please enter only digits.” |
| 3 | Input 5 units that user has failed. | Input the unit information 5 times that result in a failed with other unit results that have passed. | System to still return a result and not automatically reject the user for honours program. | Console displays “Invalid, you have inputted that unit 3 or more times. Please input a different unit:” |
| 4 | Authentication of user. | Input valid user details  (Valid input is a or b for both admin and password.  E.g. admin = a password = a) | System to direct the user to the user interface for data input. | Console displays  “Invalid credentials, please try again.”  “You have 2 tries remaining.” |
| 5 | Invalid Student Id | Input invalid student id (eg.123ABC) | Console to display incorrect Student Id. | Console displays “Invalid input. Please enter only digits.” |
| 6 | Valid Student Id | Input Valid student Id (10507429) | Console to have no notification of invalid student id | Console displays  “Please enter between 12-30 units:” |

## Three Tiered System

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Test | Test Method | Expected Result | Real Result |
| 1 | Input number that is not an option in the menu | Input the number 100 | Console loops back and display input error | Console outputs “Please enter either 1 to 3 shown the options below:” |
| 2 | Client-side system not allowing for input if server side is not connected | Do not run the server program before the client-side. | Console to display error | Console outputs “ ERROR: The honours Pre-assessment Program is not running……” |
| 3 | Input student id that is not in the system | Log into the system.  Select option 1  Input Id of student not in the system | Console displays error | Console outputs “This student does not exist in the school’s database” |
| 4 | SQL Injection using student Id | Log into the system.  Select option 1  Input 105; DROP TABLE Students | Console displays invalid Id | System Looping  Error |
| 5 | Invalid Student Id | Log into system  Selection option 2  Input (adfasdf) | Console display invalid Id | System looping error. |

# User Manual

## Two-tiered

**Step 1**

Download the Software from Blackboard.

**Step 2**

Download and install Eclipse Software at <https://www.eclipse.org/downloads/>. For installation instructions please refer to the <https://www.eclipse.org/downloads/packages/installer> for their step by step method on installation.

**Step 3**

Unzip the two-tier system to a folder at your preferred location. For this example, the file location will be the desktop.

**Step 4**

Launch Eclipse IDE on the left of the software under package explorer click on import projects…  
In the figure below, this import projects… is highlighted.

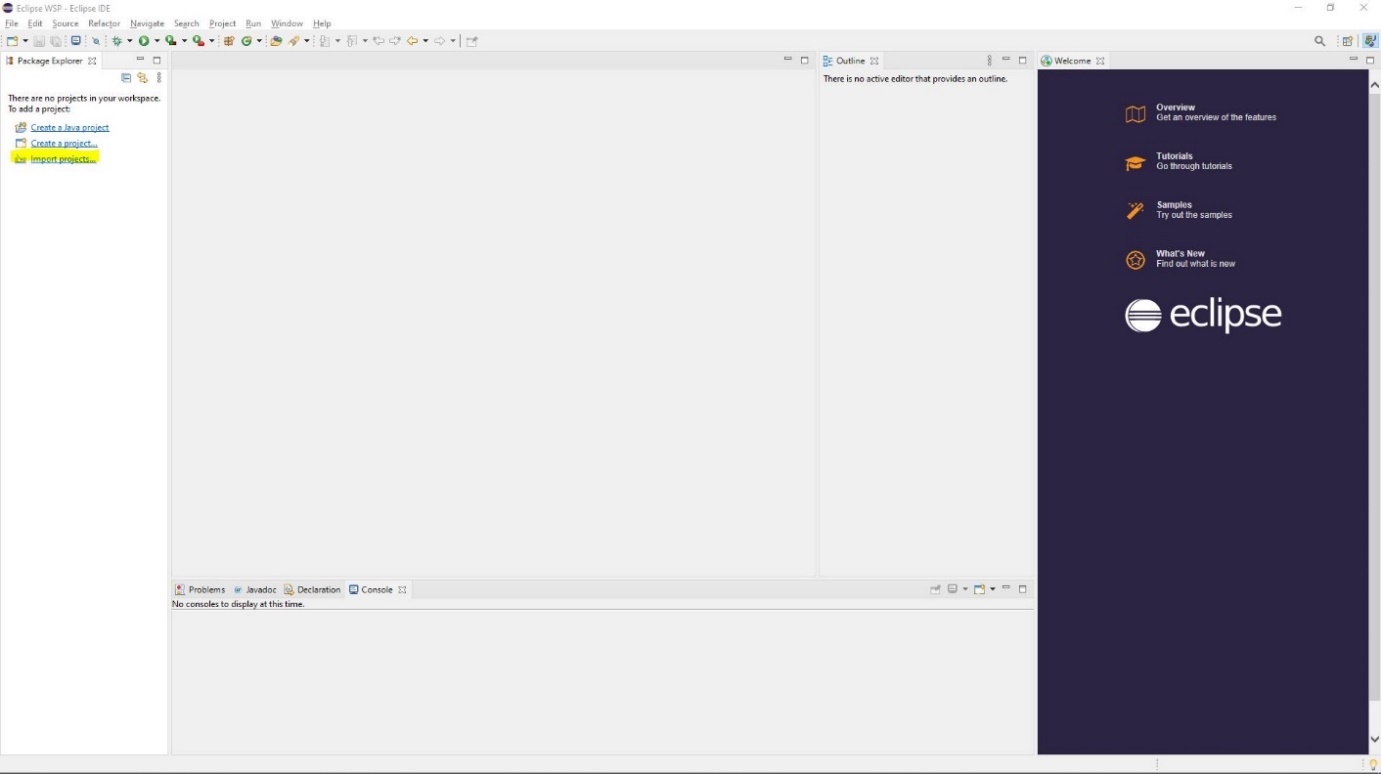


Figure 2 Import Projects

**Step 5**

After clicking the import projects link it will open a window. In this window click on the arrow next to General and a drop down should appear. In this drop down select Projects from Folder or Archive and click on next on the bottom of the window.  
The figure below highlights the items that need to be selected.

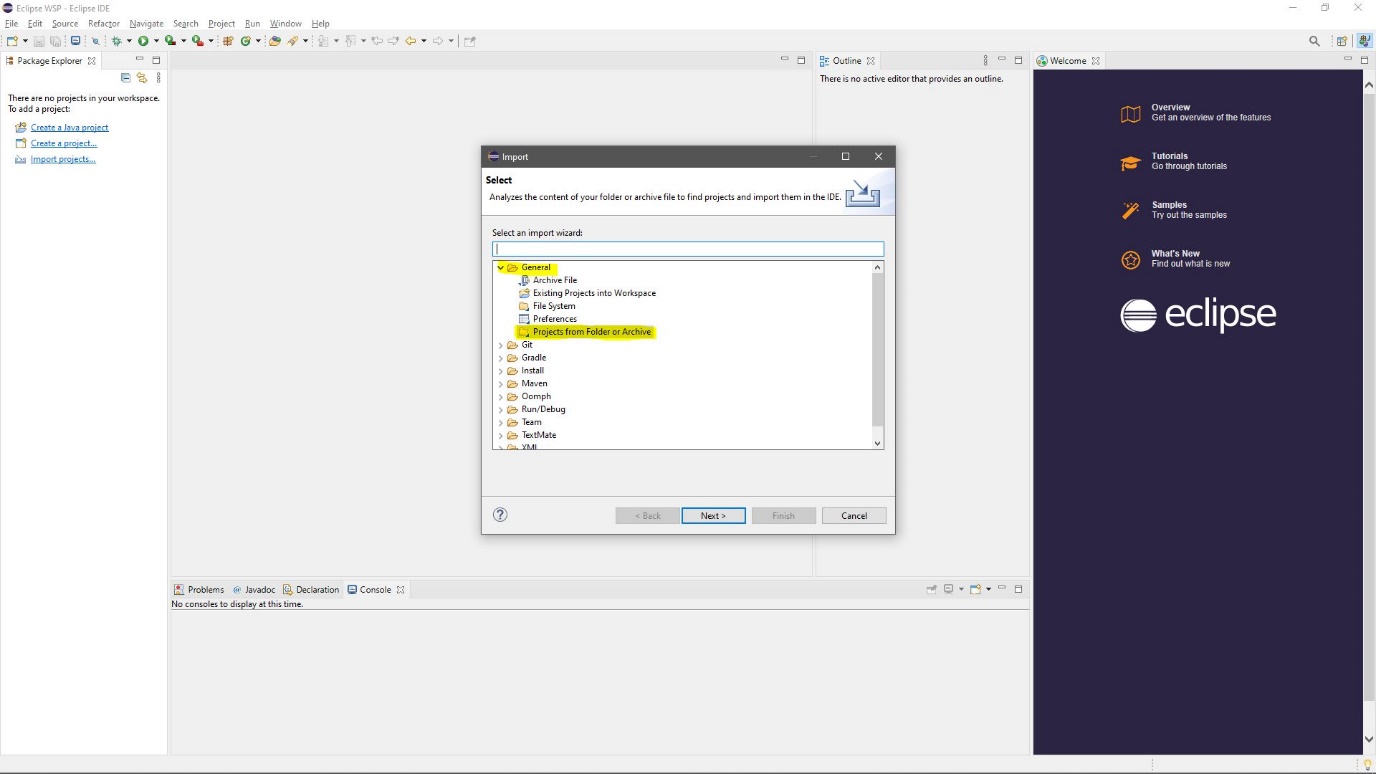


Figure 3 General and Projects from folder or Archive

**Step 6**

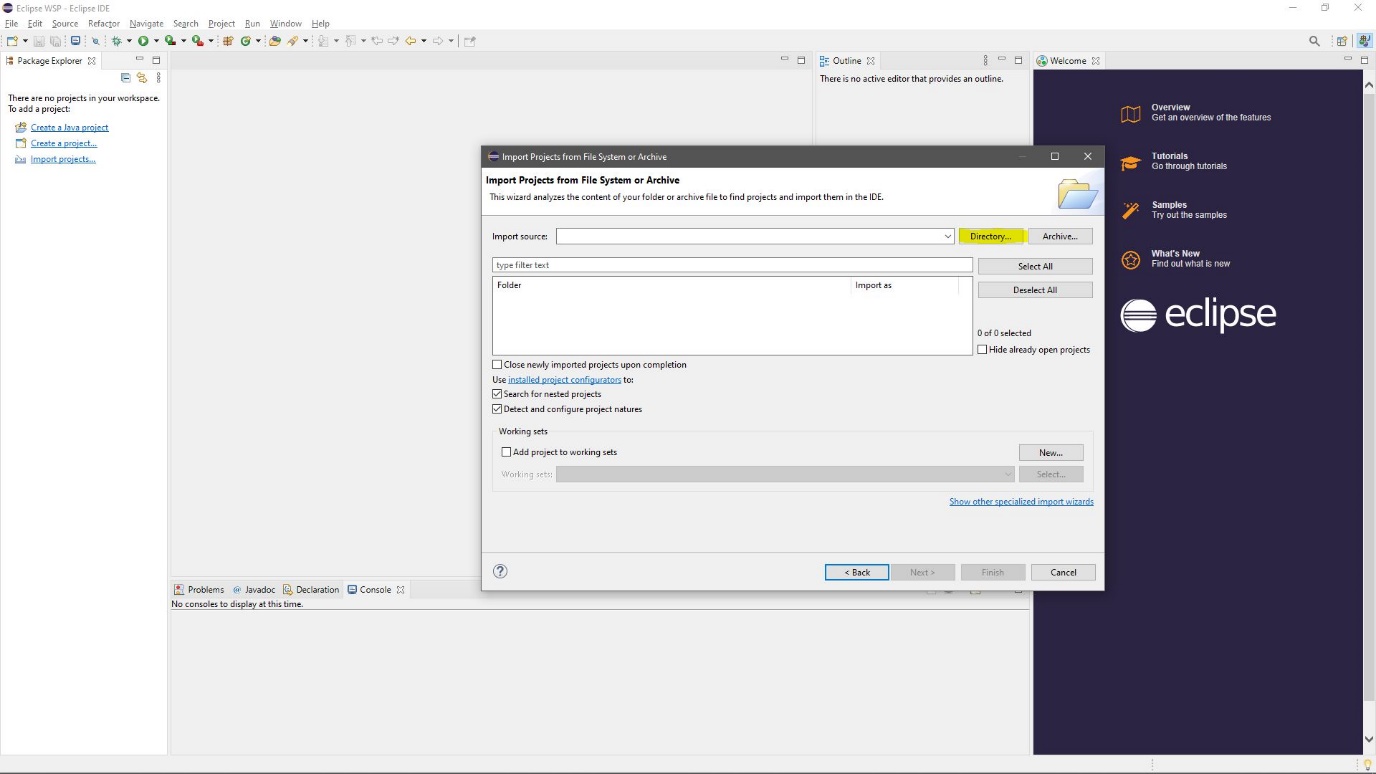
A new window should appear declaring an import source of the project. Click on the directory button and find the location of the program you have downloaded and unzipped and select it as the folder. After you have selected the folder and check with the ***figure 4*** that the window is display the correct information, click on the finish button.   
The figures below show the location of the button and how the files should appear in the window.  


Figure 4 Directory Button

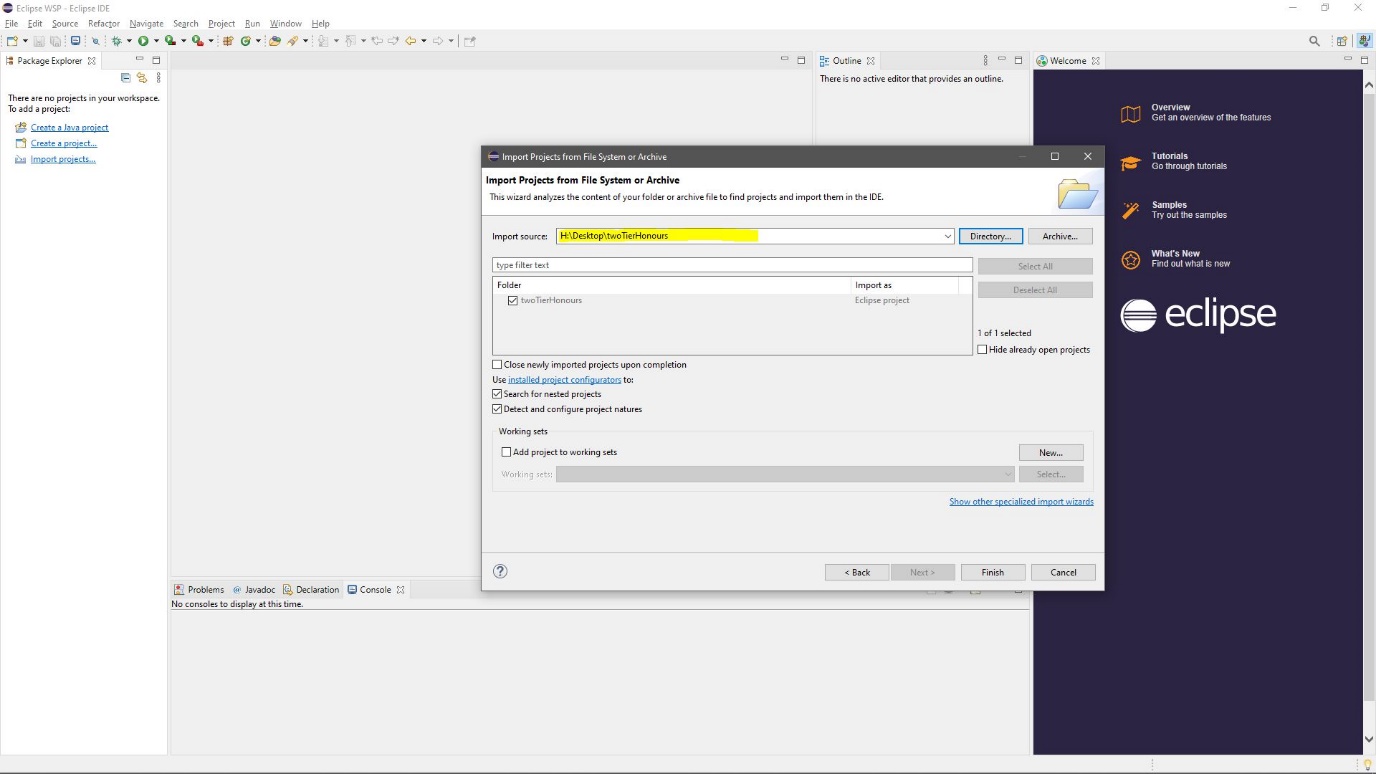


Figure 5Window displaying the files

**Step 7**

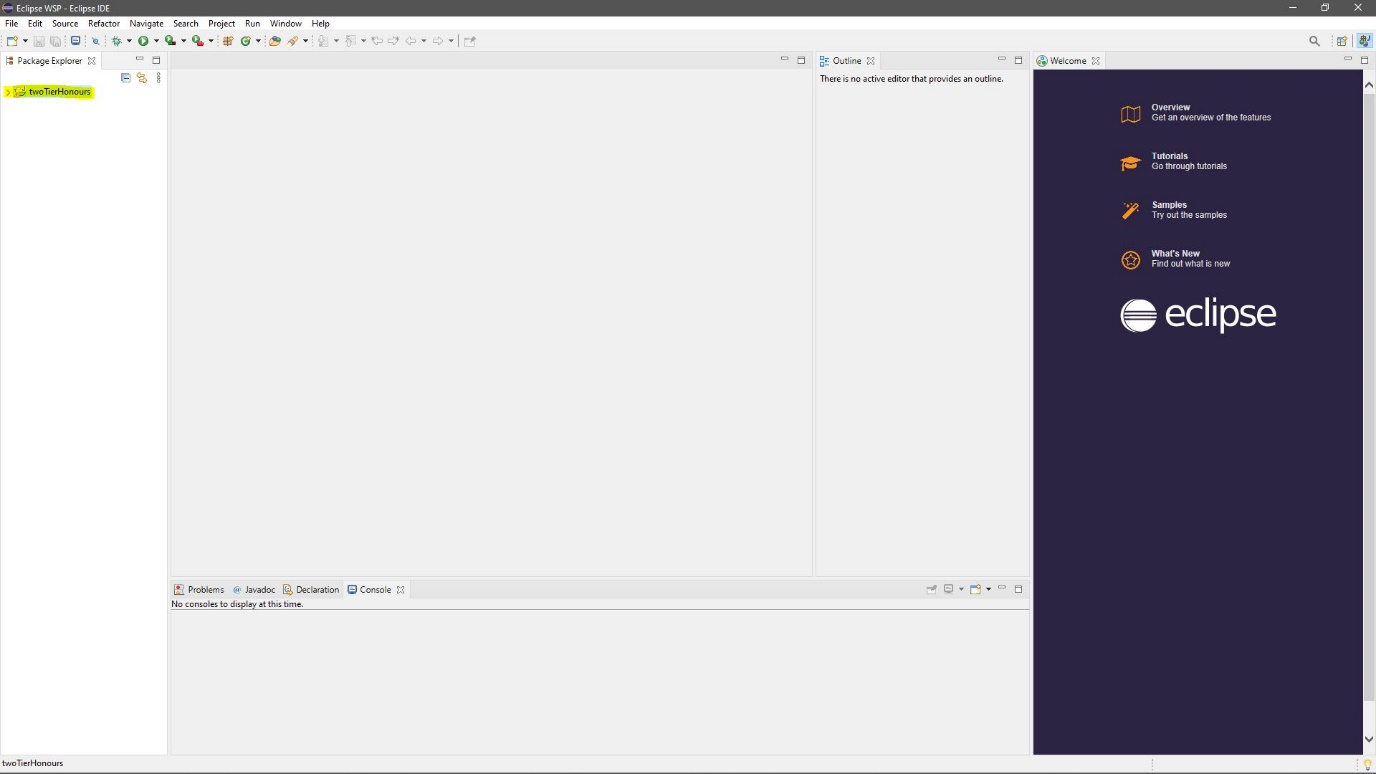
You should now be greeted with as screen shown in ***figure 5***, click on the arrow next to the file name and continue until you see the files highlighted in ***figure 6***.

Figure 6 Program Files

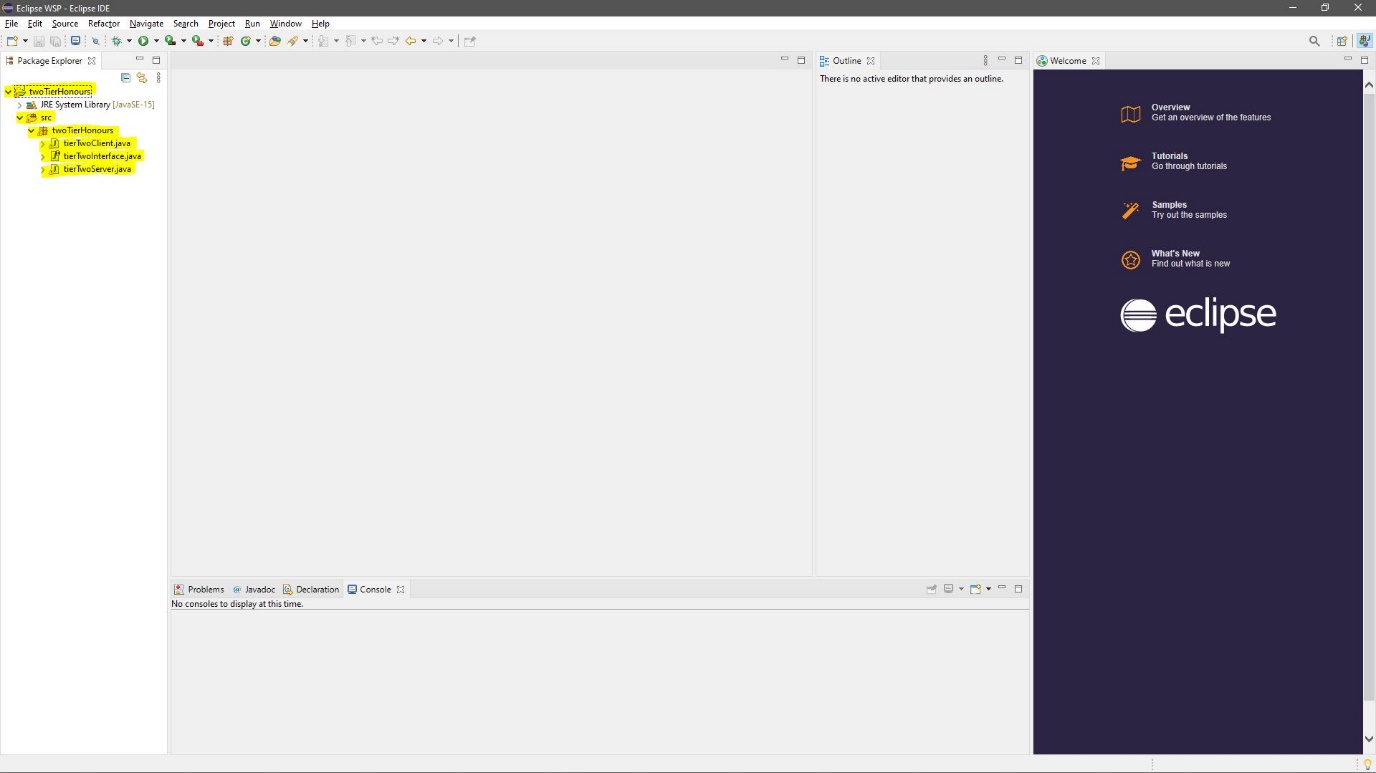


Figure 7 Highlighted Files Required for the Program

**Step 8**

Select the **tierTwoServer.java** file and right click on the file. You should be able to see a drop down with many options. Find the Run As option and click on it and a new option will be displayed as 1 Java Application. Click on this and the server side of the program will start its process. ***figure 7*** displays the drop downs required to run this part of the program. To know when the server is running successfully the console below will display “The Honours Application is running…..”, this is highlighted in ***figure 8.***

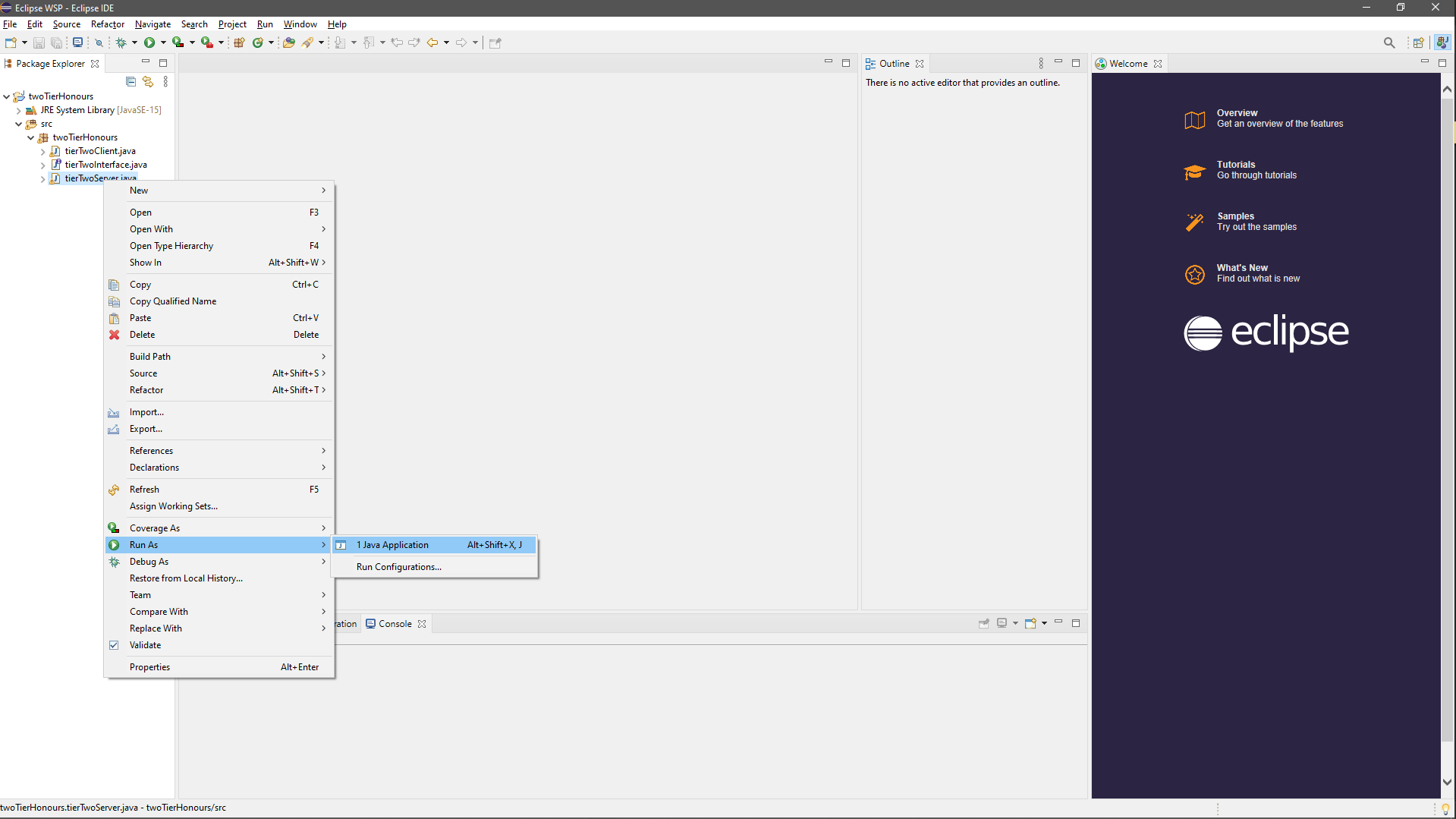


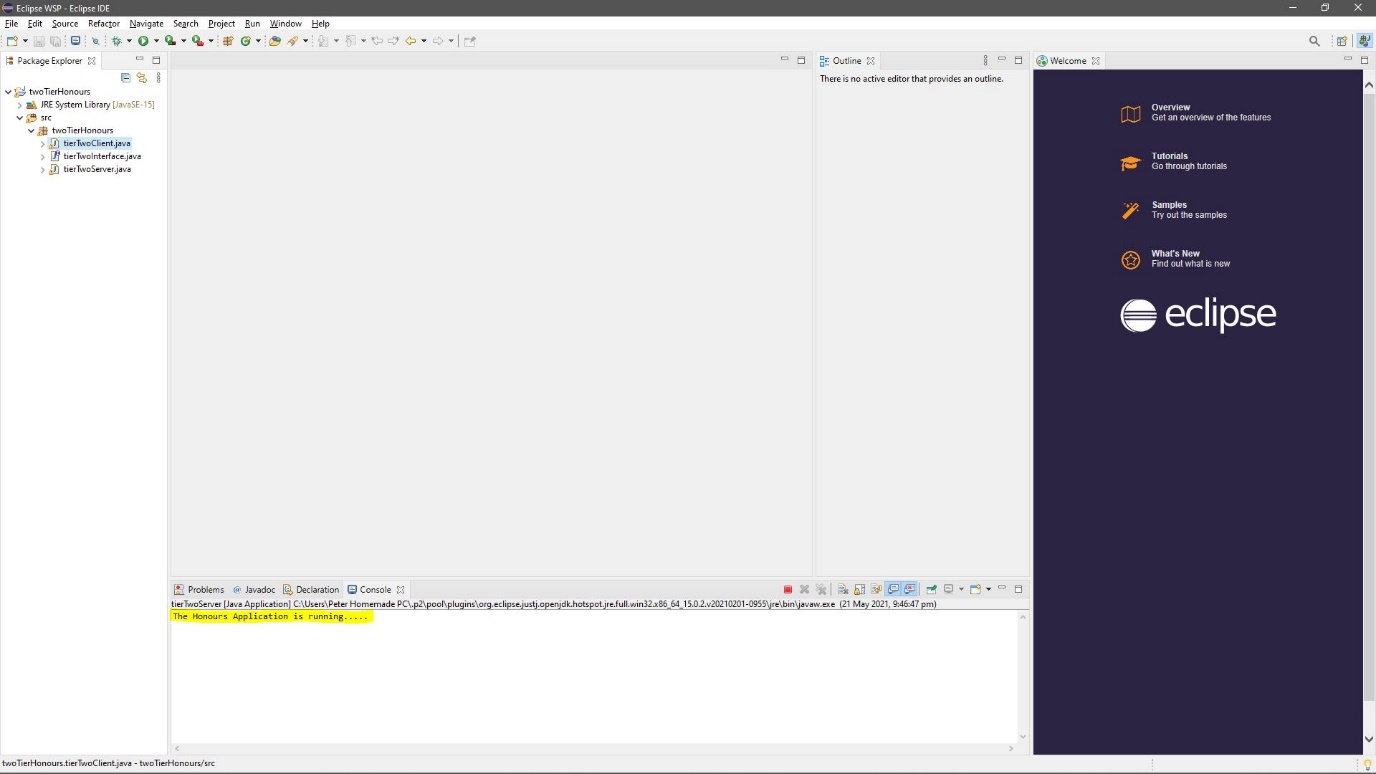
Figure 8 Drop downs to start the Server

Figure 9 Console displaying application running

**Step 9**

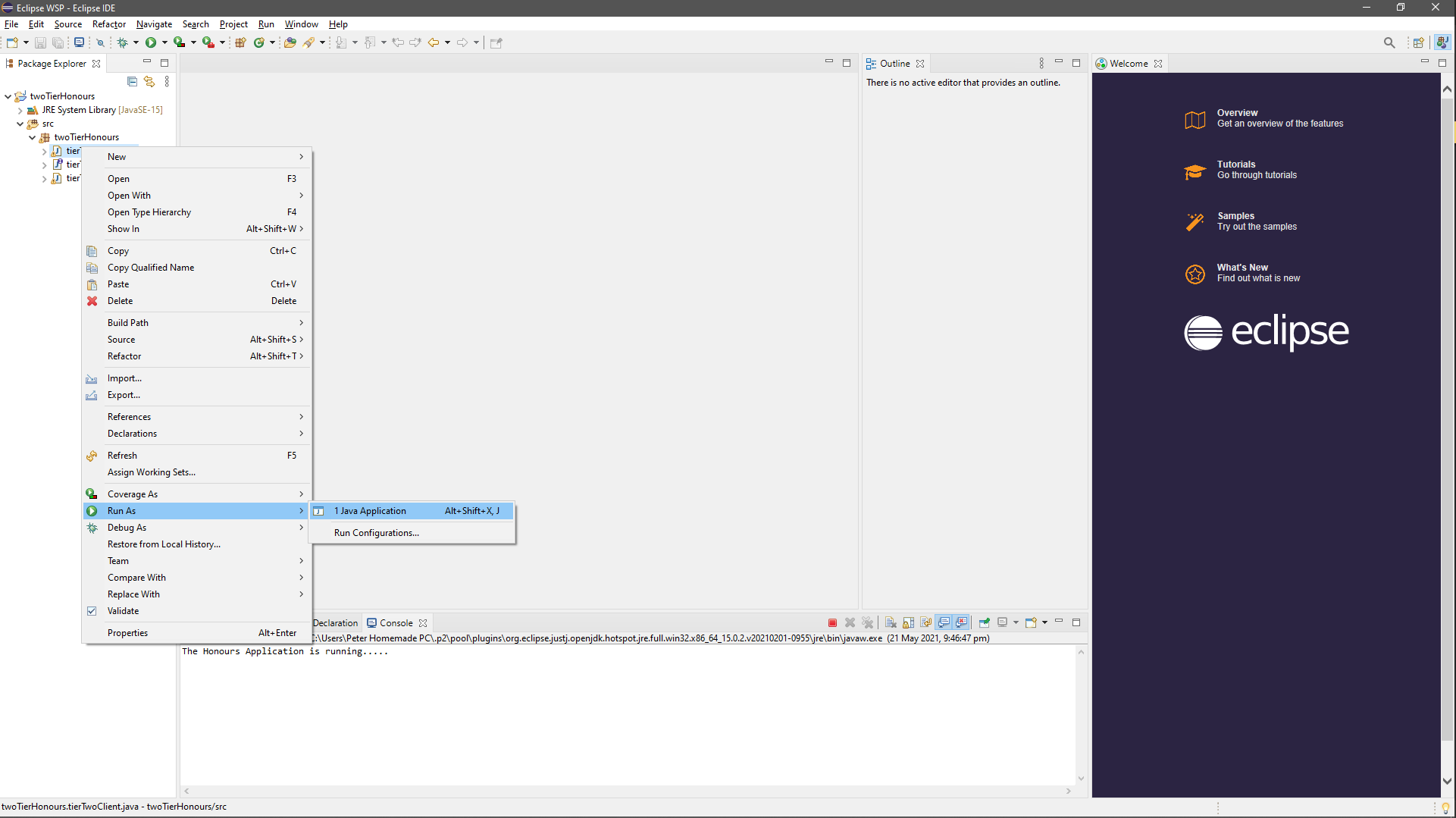
Repeat the steps taken for Step 8 but select **tierTwoClient.java.** When the application is running, it will display in the console and is able to take inputs via the console. To interact with the application, select the console and begin typing and when you are ready to submit the information related to the prompt, press enter and the application should be able to accept the input if the information is validated and continue for the rest of the other prompts.

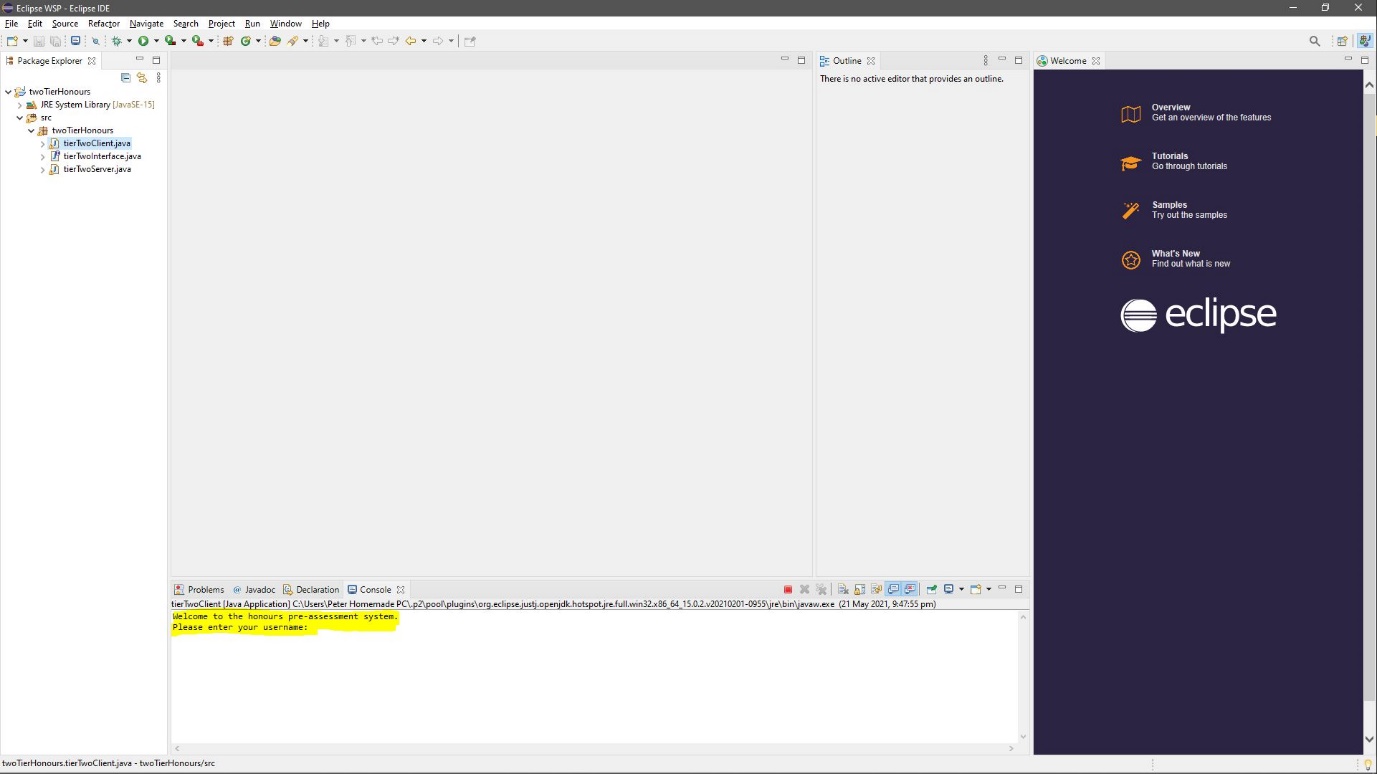
Figure 10 Drop downs to start the Client 

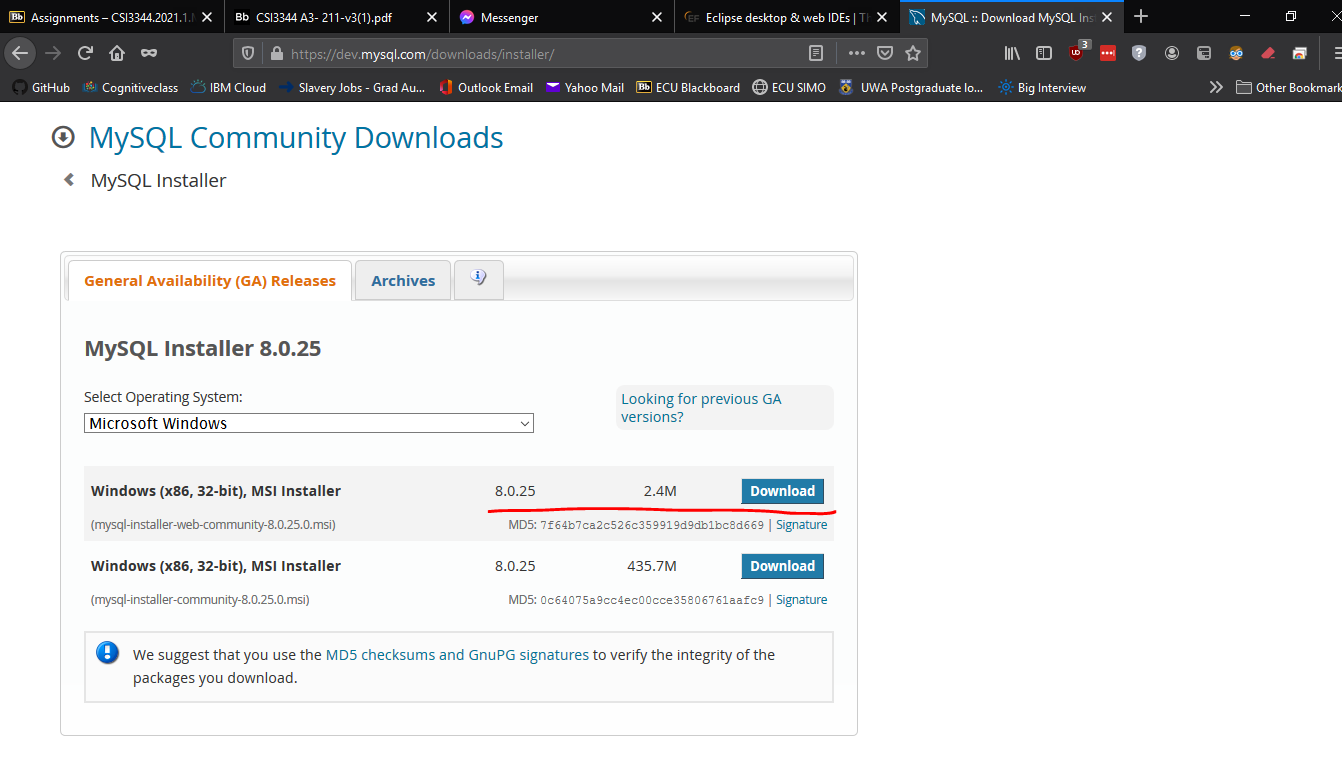
Figure 11 Console displaying Application

## Three-tiered

### Software Installation

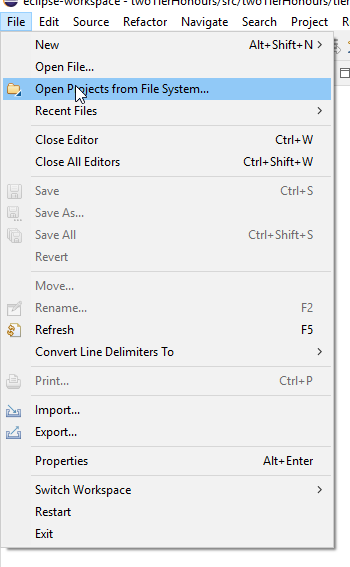
To use the honours evaluation application, you are required to download 2 software:

1. Eclipse IDE – <https://www.eclipse.org/ide/>
2. MYSQL – <https://dev.mysql.com/downloads/installer/>

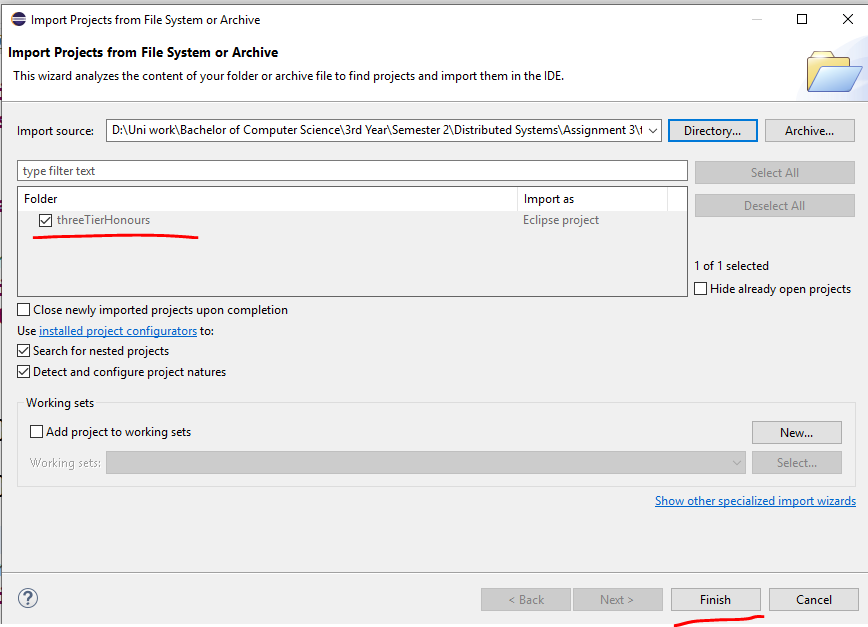


### Eclipse Part 1

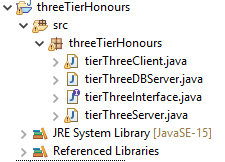
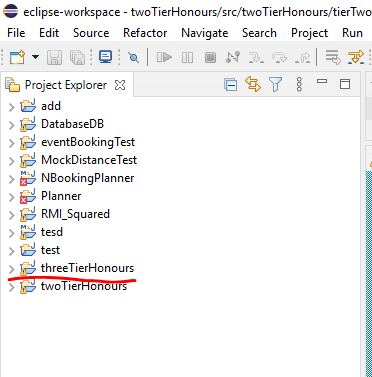
Once you have downloaded Eclipse, run the application. Once Eclipse is started, go to File -> Open Projects from File System to import the threeTierHonours folder.



After that, click on Directory and direct to the threeTierHonours folder. Once you have directed the correct folder, click Finish.



You should see a Project Explorer with the folder threeTierHonours from the left side of the screen. Click on the little arrow and it show be in fig 1. Keep clicking on the little arrows until you see in *fig 1.*



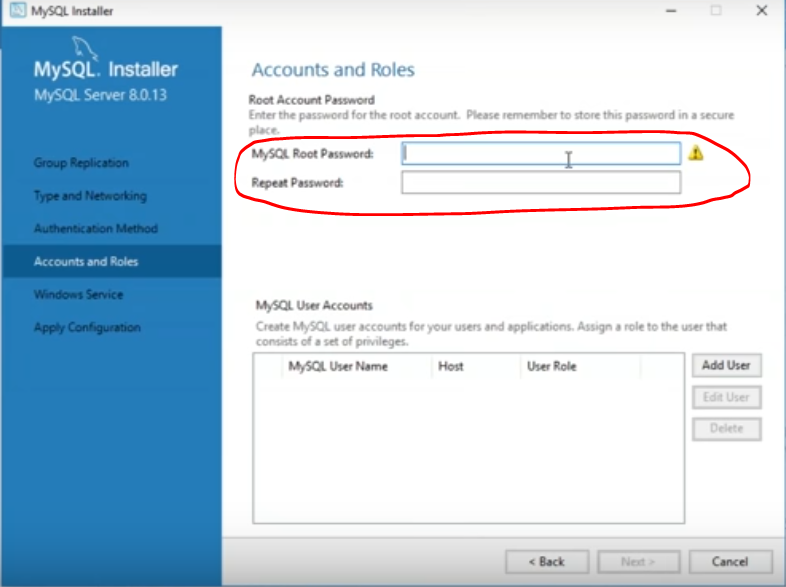
*Fig 1.*

You should have 4 files and each of these files are important when starting up a RMI application.

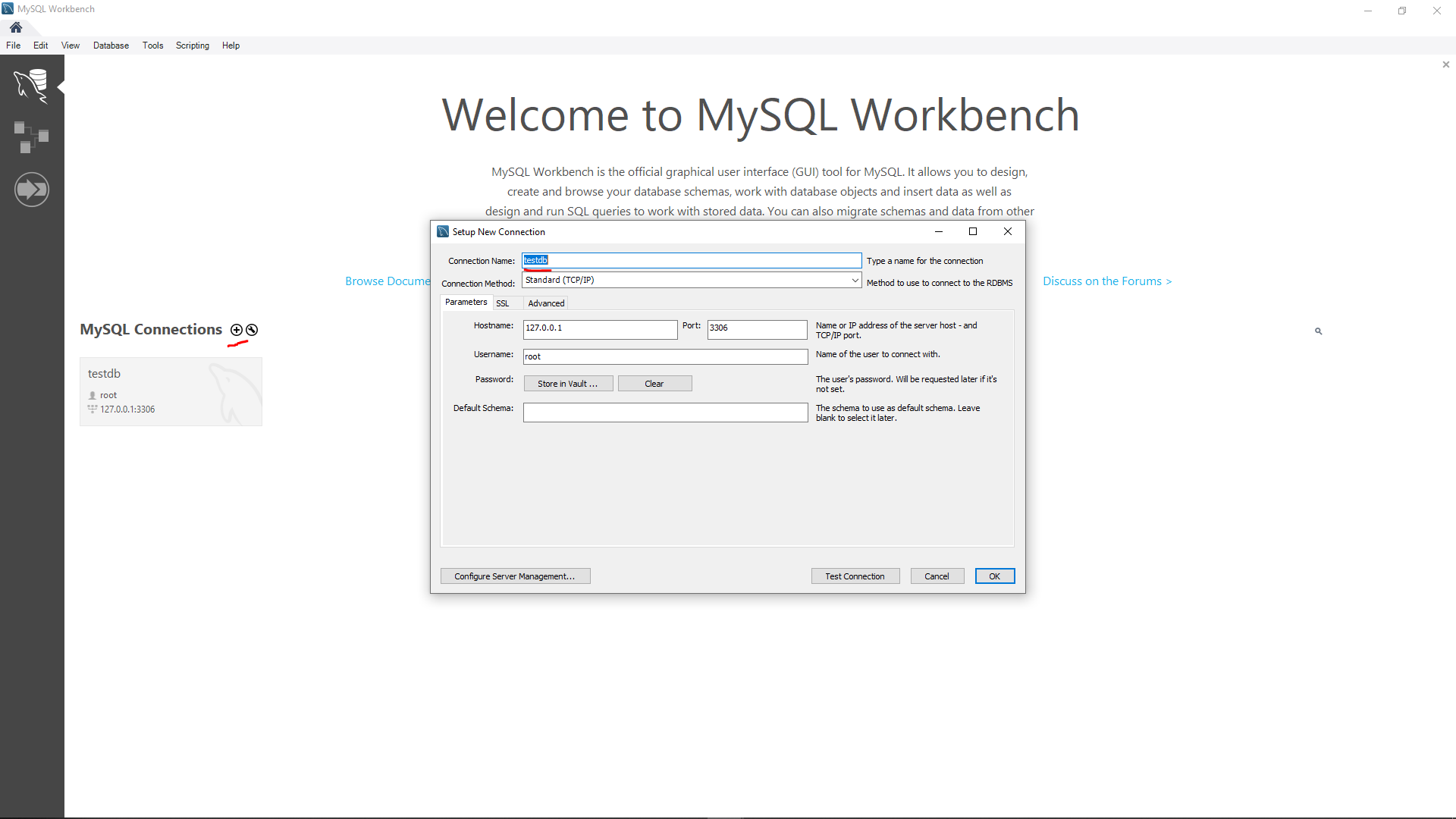
### MYSQL setup and installation

Before you get started on the application, ensure that you have MYSQL installed. Follow this link that explains how to install MYSQL <https://www.youtube.com/watch?v=GIRcpjg-3Eg>

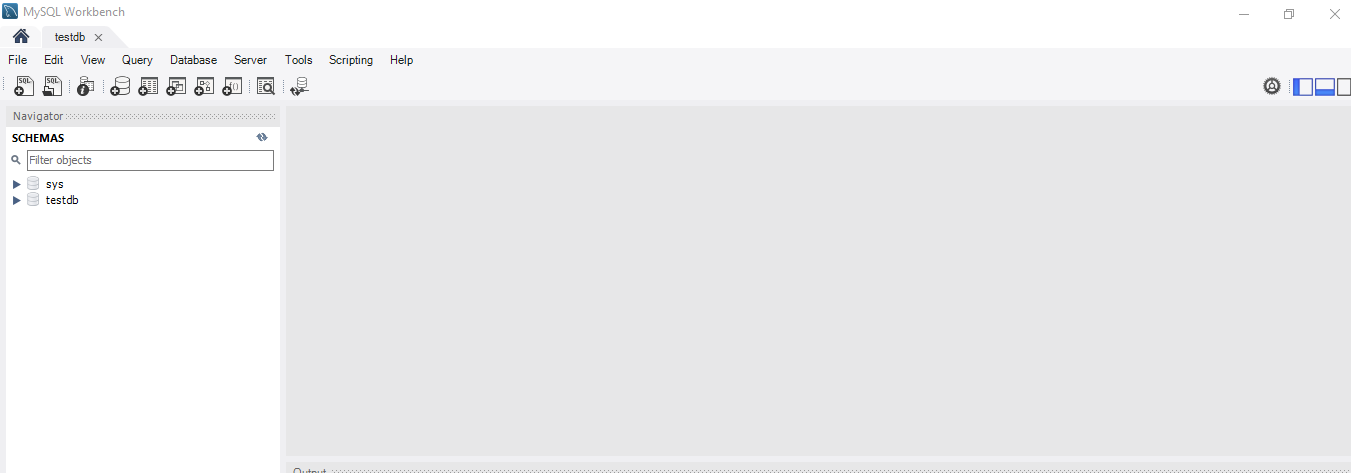
Please ensure that your MYSQL Root password is “root”.

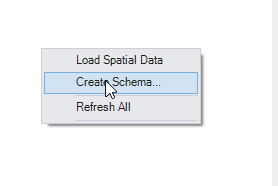


Once you have installed MYSQL, go to MYSQL Workbench and you should see testdb on your MYSQL Connections. If not, create a connection by clicking an add button and name the connection “testdb”.

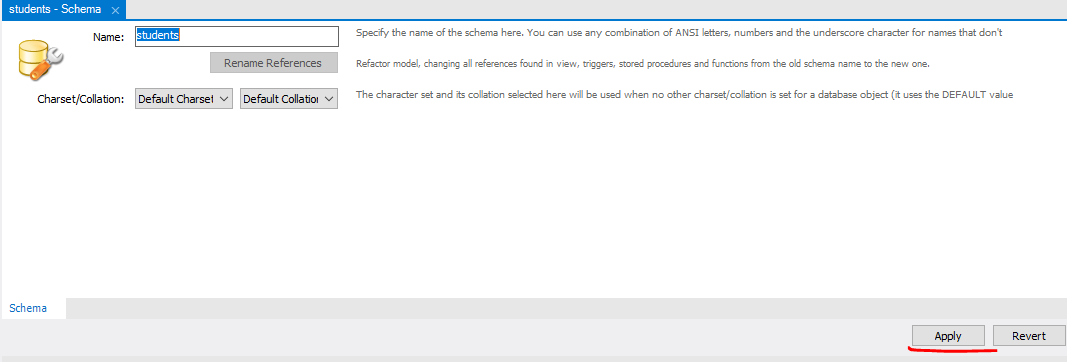
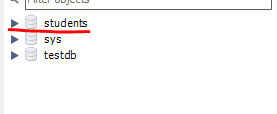


Double click on that and you should see this screen. Right click on an empty space in the schema space and click on create schema.





Name the schema “students” and click Apply. Keep on following the instructions by keep on clicking apply. You should now able to see “students” as an official database schema required for using this application.

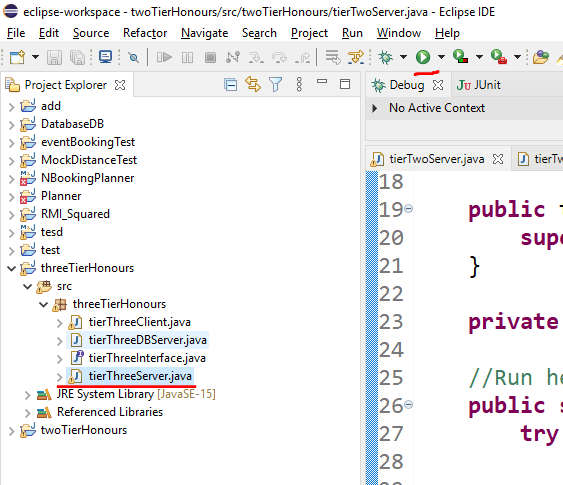
 

### Eclipse Part 2

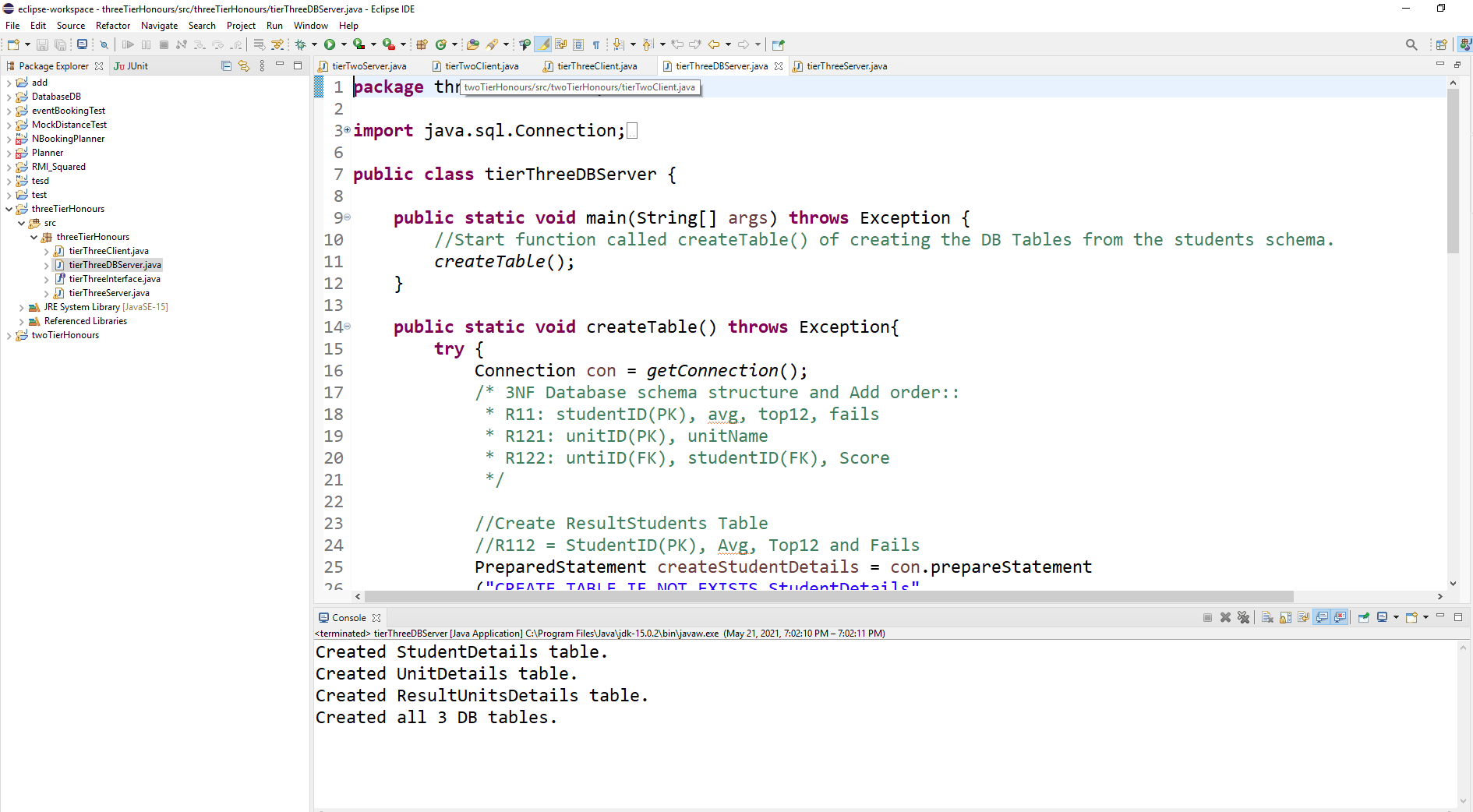
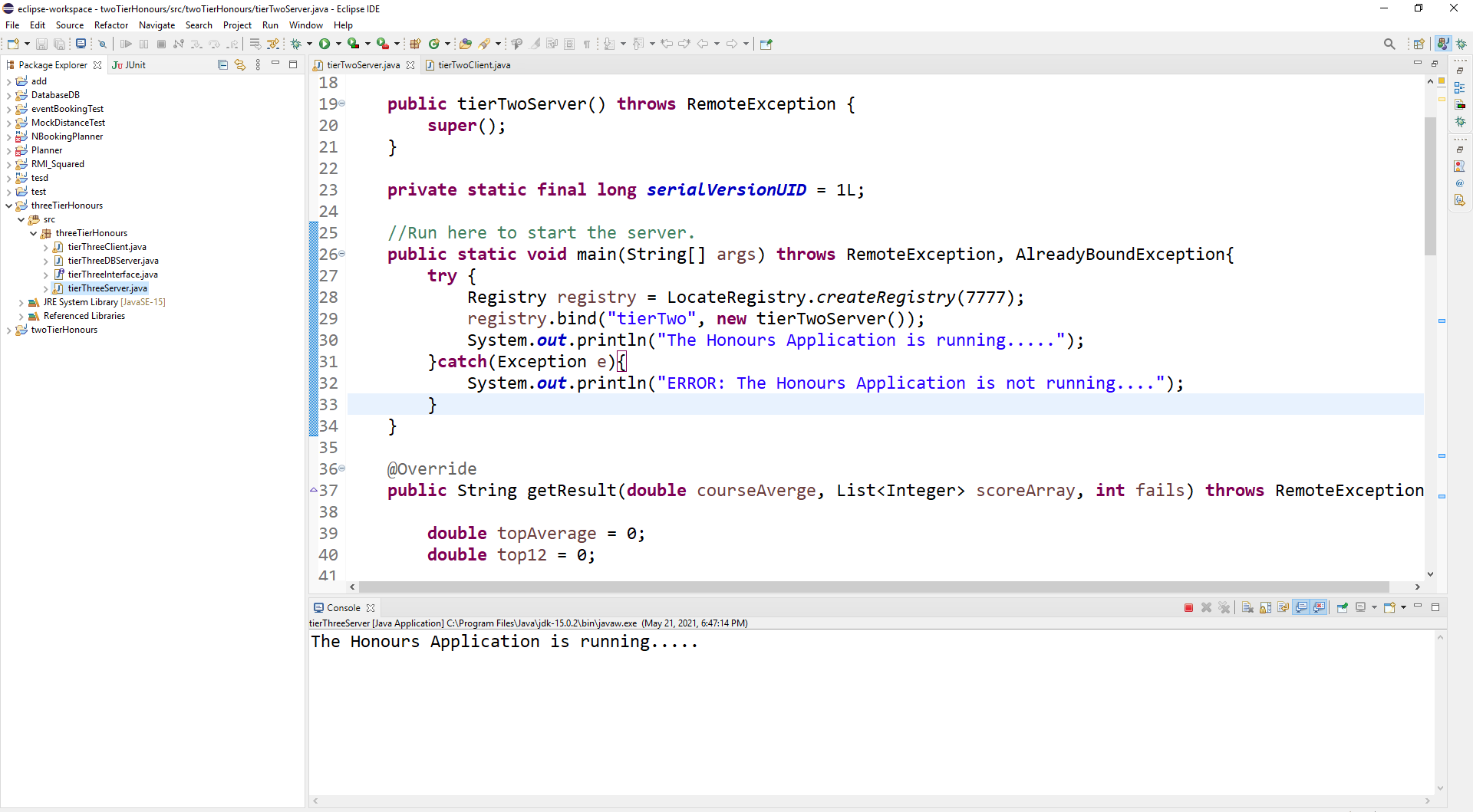
Once the MYSQL and schema has been setup, go to Eclipse and import the threeTierHonours if you have not followed the correct steps. To run this application, you need to execute the following java files in this order:

1. tierThreeServer.java – this basically setup the registry and binds the tierThreeinterface.java required for starting an RMI application.
2. tierThreeDBServer.java – creates the DB tables from the “students” schema required for this application.
3. tierThreeClientClient.java – the main application that the client supposed to see.

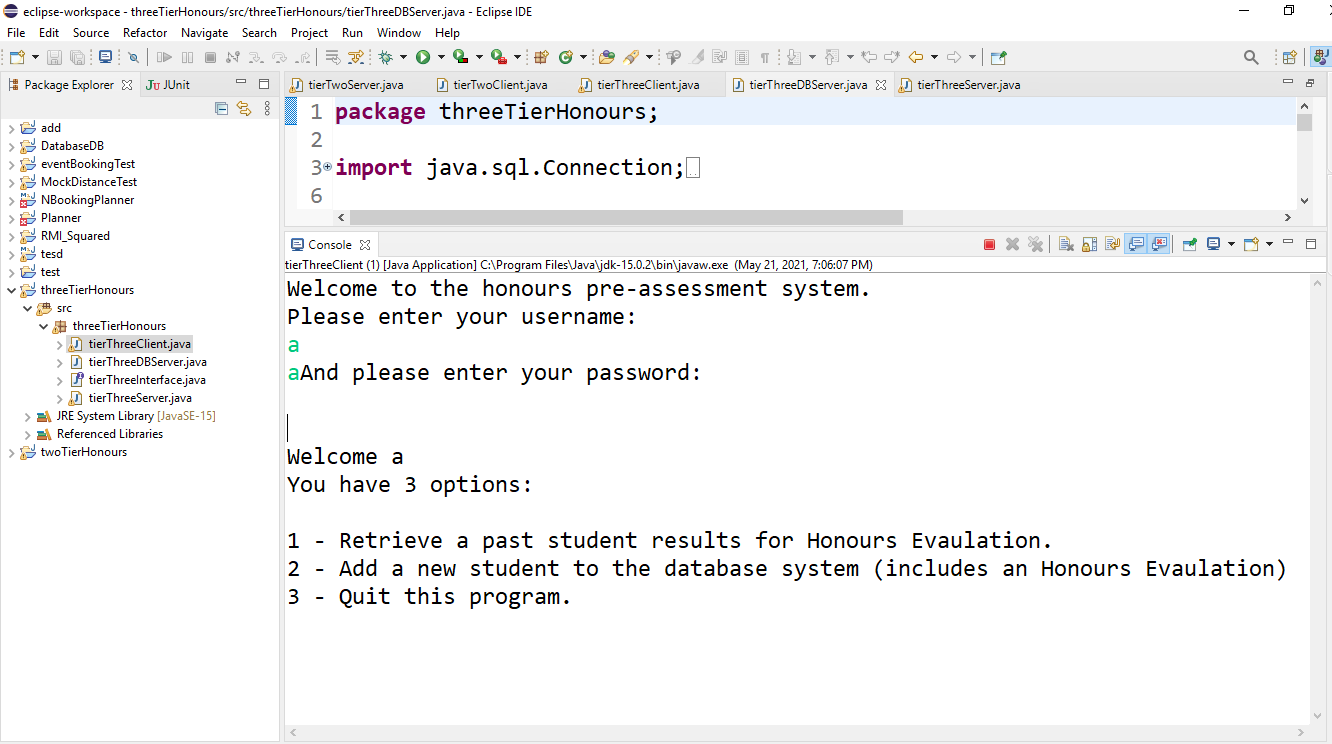
Click on tierThreeServer.java file and click on the green start arrow shown below this screenshot. Do the same processes for tierThreeDBServer.java and tierThreeClientClient.java



You should be able to see successful messages from tierThreeServer.java and tierThreeDBServer.java screenshot shown below on your console.



Once those successful messages appear, you can start the tierthreeClient.java application and let the application guide you.



# Conclusion

In conclusion of the report, this report has summarised the design processes and the requirements of the Honours Pre-assessment. The report has gone into detail on how Remote Method Invocation is used in distributed systems. By using RMI and the stub and skeleton object allows for the system to use existing network communications (TCL and RRL) for communication between the client-side and server-side. Using the RMI method is allows for the client and server communicate with each other without needing to create a new network for their communication in the Honours Pre-assessment system. How this Honours Pre-assessment program was developed using Java and the Eclipse IDE and for the tier-three system the use of MySQL acting as the database for the system for storing and accessing the student information.

To conclude this report, the Honours Pre-assessment system is a based on the architecture RMI and is a developed using Java and Eclipse with use of MySQL for the three-tier system. While the system is not perfect it is a simple display of the power of how distributing resource demand on to other device allow for a more dynamic system in a whole and that even the most basic systems can benefit from using a distributed system.

# References

JavaTpoint. *RMI (Remote Method Invocation).* Retrieved from JavaTpoint: <https://www.javatpoint.com/RMI>

TheServerSide. *Remote Method Invocation.* Retrieved from TheServerSide: <https://www.theserverside.com/definition/Remote-Method-Invocation-RMI>