Client Server Architecture Course Work Description

# Student Details

**(Add here your Student Details)**

# Functionalities

**(Tick the boxes to illustrate the functionalities you have implemented)**

* □ Login Service (Tick here if you have implemented the Login Service)
* □ Multiple Threads (Tick here if you have implemented a Blog Service with multiple threads)

□ Single Thread (Tick here if you have implemented a Blog Service with a single threads)

Additional Notes: you can add multiple threads and you can also edit it

# Client/Server Technology

**(Tick the boxes to illustrate the Client/Server Technology you have used)**

* □ Web Services (Tick here if you have used Web Services to implement communication between the client and the server)

□ Sockets (Tick here if you have used Sockets to implement communication between the client and the server)

□ None (Tick here if you DID NOT implement communication between the client and the server)

Additional Notes:

# Data Persistency Technology

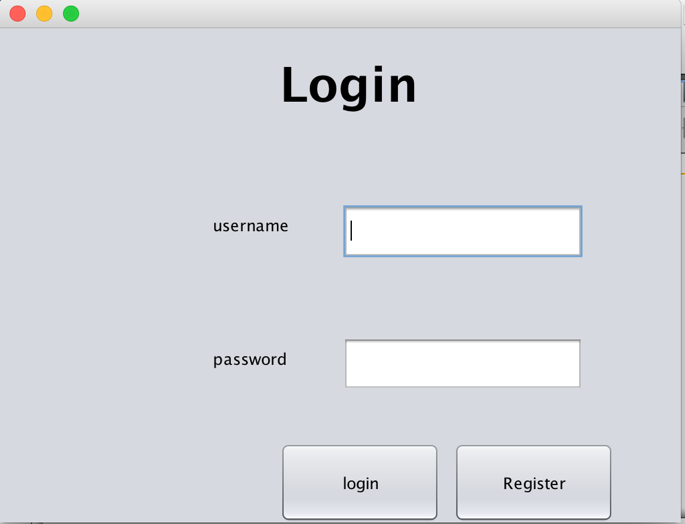
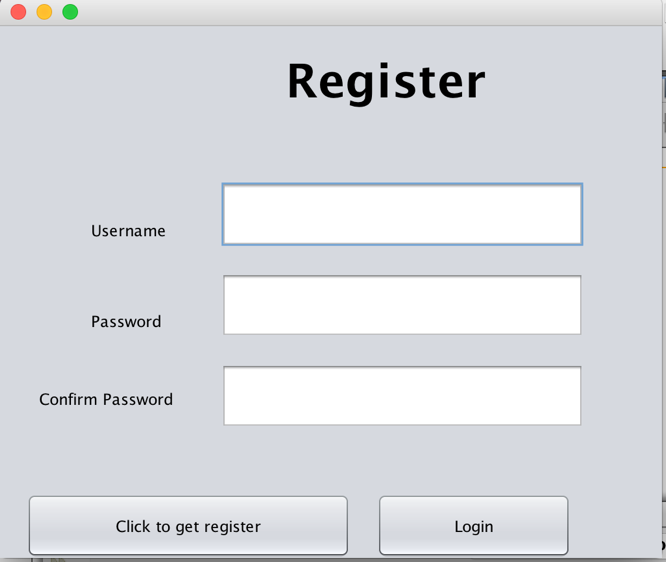
**(Tick the boxes to illustrate the Data Persistency Technology you have used)**

* □ Serialization (Tick here if you have used Serialization to implement Data Persistency)
* □ DataBase (Tick here if you have used a Database to implement Data Persistency)
* □ Custom-Defined File (Tick here if you have used a Custom-Defined file to implement Data Persistency)

□ None (Tick here if you DID NOT implement any Data Persistency)

Additional Notes:

# GUI Screenshots

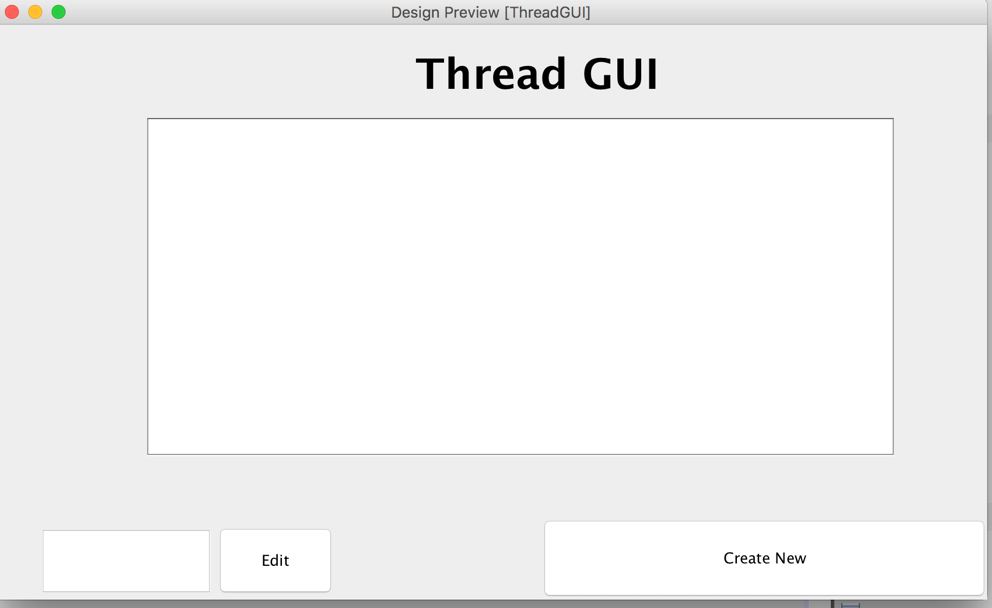
**(Report here your GUI Screenshots and illustrate how you can navigate through them. As an example: When the user performs a successful login (GUI Screenshot 1), he is then directed to the Select Thread Page (GUI Screenshot 2)**

The screenshot on the left shows the registering GUI where it allows user to register their detail in order to access the threads.

Once they register their detail and they can click the login button and from their they can access the main GUI.

The screenshot on the left shows the login GUI, when user run the client chat, they will be presented with the login GUI where they have to Login to access the thread.

If the user doesn’t have the login. They can create one by clicking on the register button at the bottom where it will take them to registration GUI



The screenshot on the left shows the Main GUI thread, where you can see all the threads.

User can select create new threads or edit the one of threads from the list

When user click on edit the threads it will allow them to write chats with the server by oping a new GUI and same for create new thread it will open new GUI

# 

The screenshot on the left shows the Chat GUI.

In Chat GUI user can write and messages and also who wrote the messages by click on add button

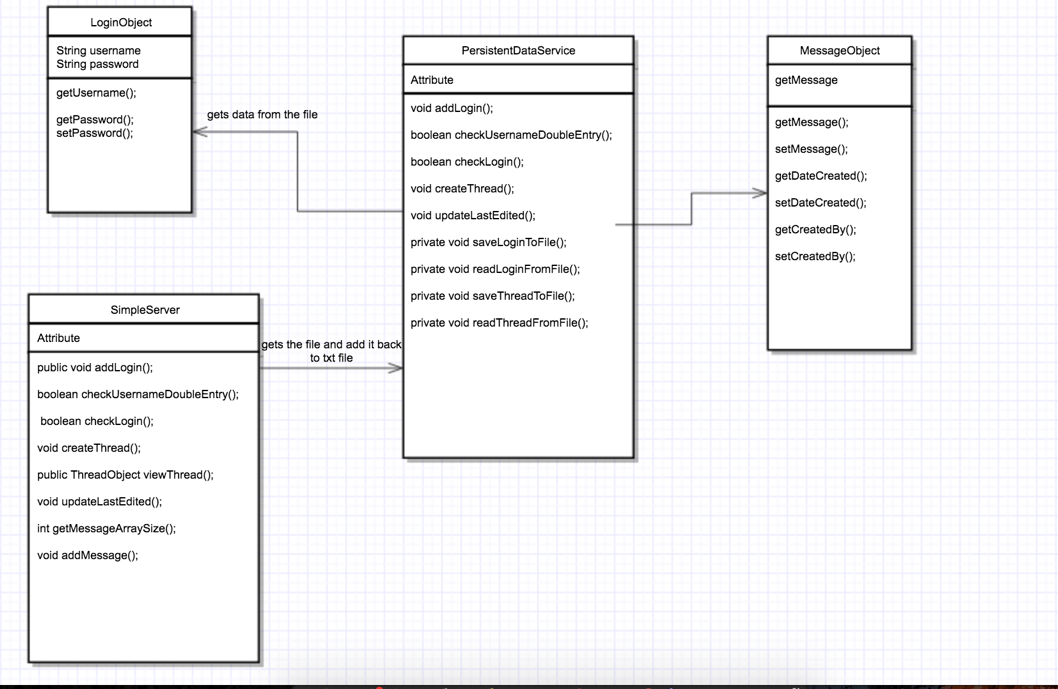
# 

The screenshot on the left shows the creating new thread GUI

It will as for Thread title and who created the thread. And once the required field are match it will create new thread and take back to the main thread GUI

# UML Class Diagram

**(Add here your UML Class Diagram)**

****

# UML Interaction Diagram

**(Add here your UML Interaction Diagram - only the sequence diagram is necessary)** [**https://www.tutorialspoint.com/uml/uml\_interaction\_diagram.htm**](https://www.tutorialspoint.com/uml/uml_interaction_diagram.htm)

# Development Approach

The first step I took was creating 6 different GUI using NetBeans Java swing jframe to make sure I finish with the easy bit and also inside the GUI I added the buttons and textfield to allow users to interactive with the GUI**,** To implement the coursework, I had to refresh my knowledge on Swing GUI and Java Arrays. Then I developed a prototype GU for which I took a bit of help from lectures and tutorial. They hard part I found was implementation a prototype to test my data structure and main methods of the server, where it took me a lot of time to try figure out how to implement these prototype to make it work, after completing the prototype then I added data persistency with serialization and a Web Service communication system to connect the GUI to the server. The final step was debugging of the code.