Table of Contents

[**1.** **INTRODUCTION** 2](#_Toc103273928)

[1.1 PROJECT DEFINITION 2](#_Toc103273929)

[**2.** **REQUIREMENTS** 2](#_Toc103273930)

[**3.** **DESIGN** 2](#_Toc103273931)

[3.1 CLIENT 2](#_Toc103273932)

[3.2 SERVER 4](#_Toc103273933)

[**4.** **IMPLEMENTATION** 7](#_Toc103273934)

[4.1 SERVER 7](#_Toc103273935)

[4.1.1 GRAPHICAL USER INTERFACE 7](#_Toc103273936)

[4.1.2 RMI 8](#_Toc103273937)

[4.1.3 JSON / WEB API 8](#_Toc103273938)

[4.1.4 AUTHENTICATION / COOKIES 10](#_Toc103273939)

[4.1.4 TIMER EVENT 11](#_Toc103273940)

[4.2 CLIENT 11](#_Toc103273941)

[4.2.1 GRAPHICAL USER INTERFACE 11](#_Toc103273942)

[4.2.2 RMI 11](#_Toc103273943)

[**5.** **TESTING** 12](#_Toc103273944)

[5.1 CLIENT GUI 12](#_Toc103273945)

[5.2 SERVER GUI 13](#_Toc103273946)

# **INTRODUCTION**

The objective of this report is to differentiate and critically question the reasons, context and basic principles of a variety of programming architectures and paradigms relevant to industry standard software solutions. It aims to develop, implement and critically appraise software solutions using different strategies, paradigms and architectures.

## 1.1 PROJECT DEFINITION

The project is to reflect on various concepts, paradigms and architectures related to Software Development. The project is to discuss concepts such as object-orientation, event-driven programming, application interfaces and client/server programming with reference to the code. The project code is to be developed using Java within the Eclipse IDE, based on the case study as well as the following:

* Remote Method Invocation (RMI)
* Interaction with web services (Json)
* Authentication & Cookies

# **REQUIREMENTS**

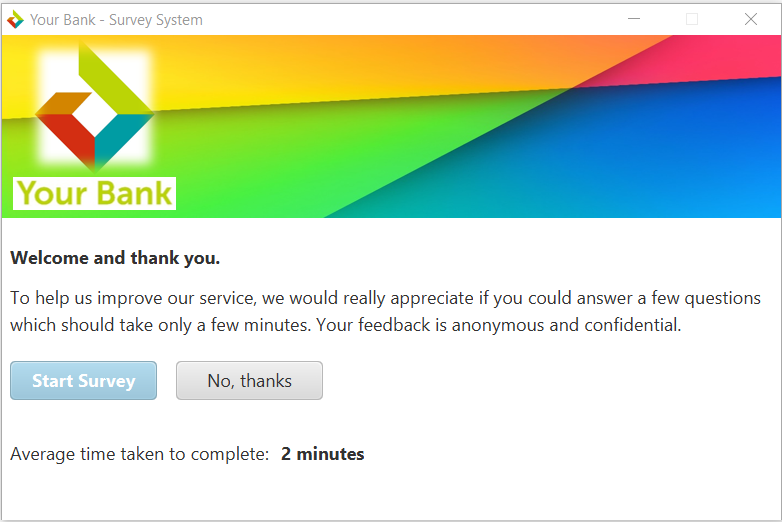
A software system that collects data from a group of people and analyses the data. The software system in question is for a banking organisation collecting feedback from customers about their products. The software has considered the different cultural and individual background of the users.

# **DESIGN**

## 3.1 CLIENT

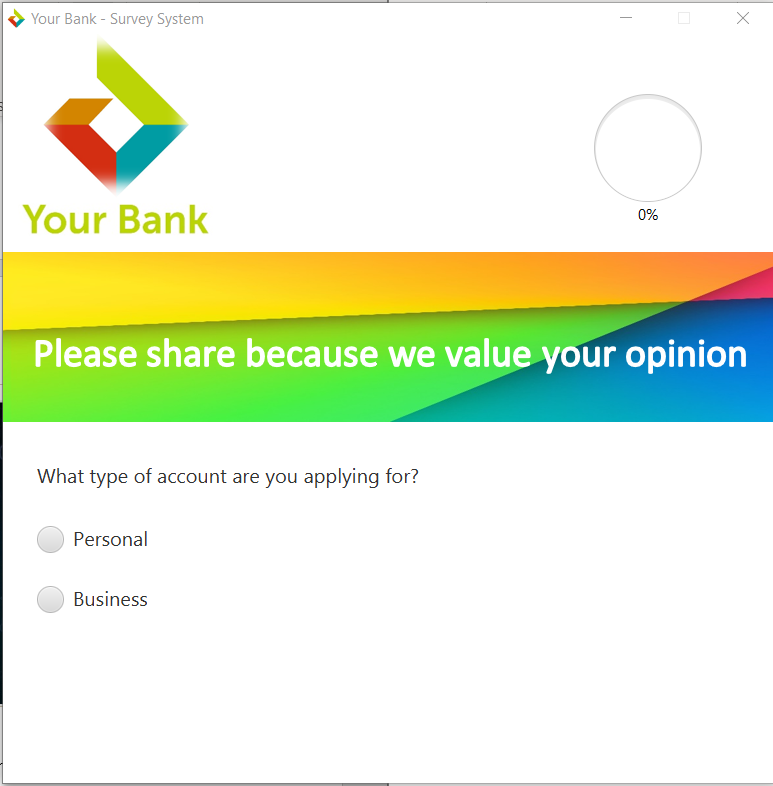
This side of the software is to be operated by clients, precisely customers of the system bank in question.

Welcome View

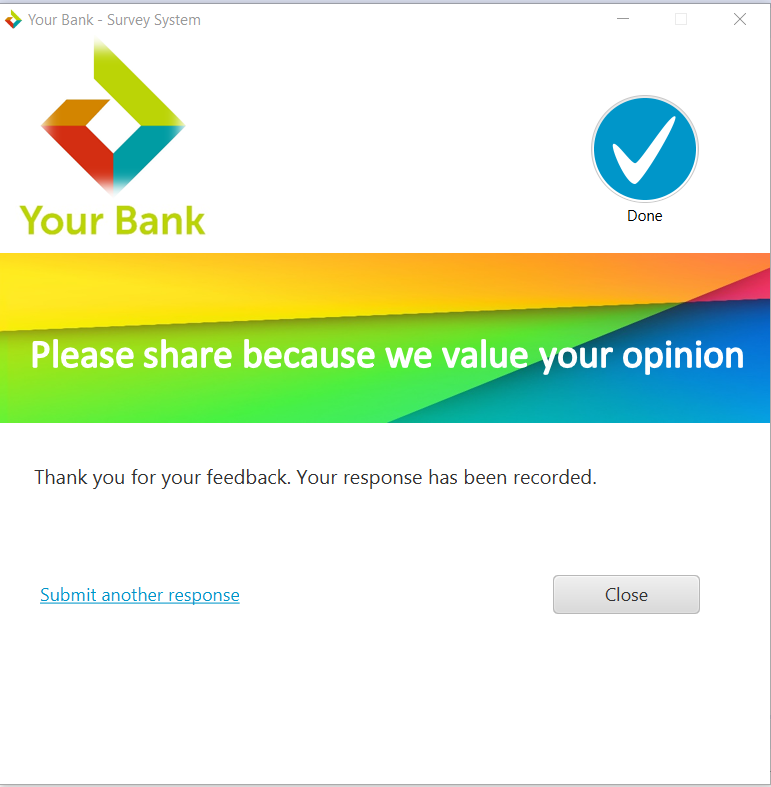
Start-up view. It can be accessed without the aid of the server.

Question Views

Views containing the questions needed to be answered as well as subsequent questions after each question has been answered.



Completed View

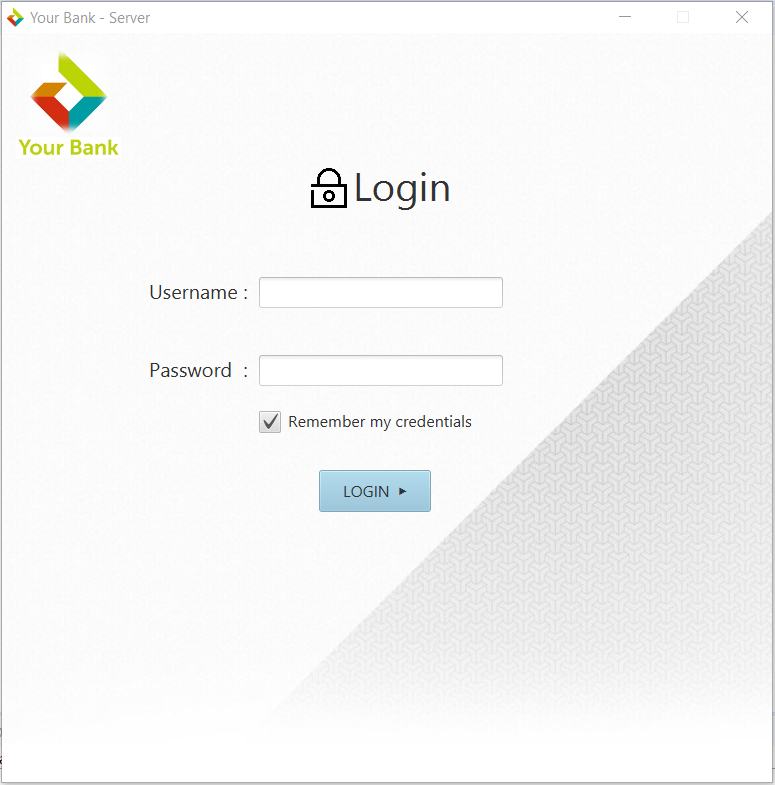
View accessed when the survey has been completed.

## 3.2 SERVER

This side of the software is to be operated by an administrator, for example a bank manager.

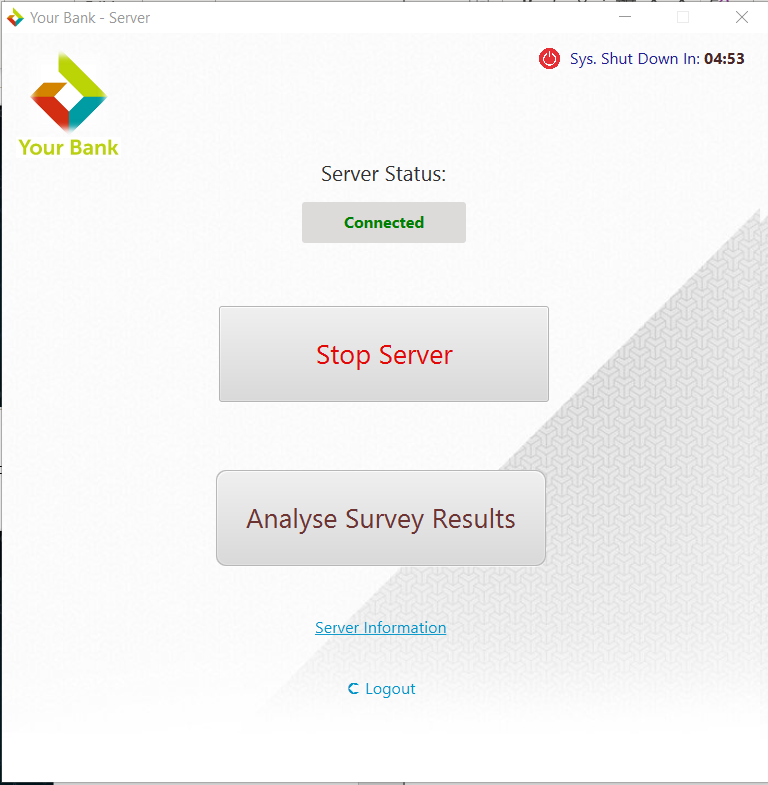
Login View

Start-up view. Only authorised personals can access this side of the program.



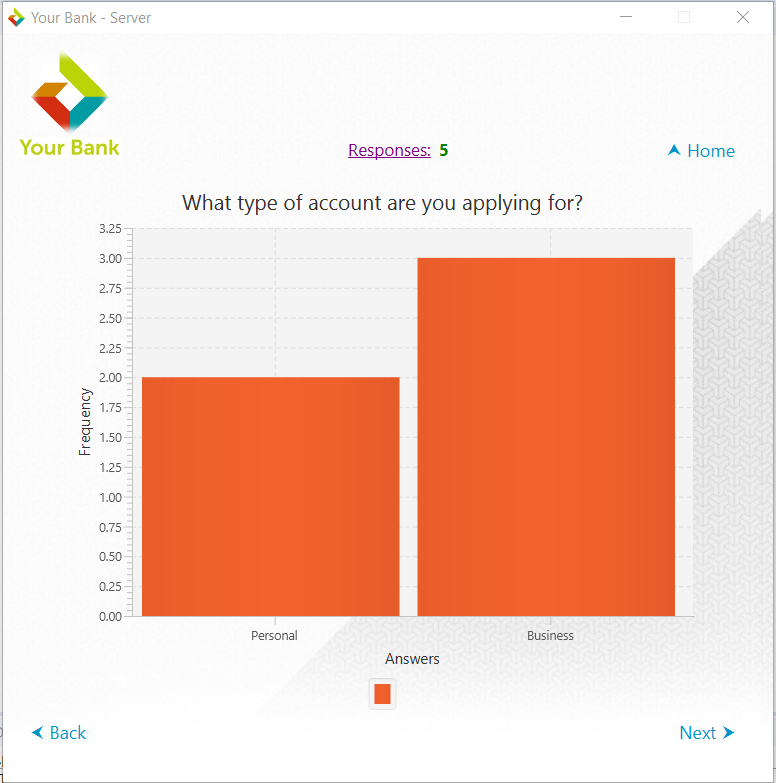
Catalogue View

A view that contains all the necessary and required access to different views in the system. It indicates the server status and also a system count down timer that exits the system after some period of inactivity.



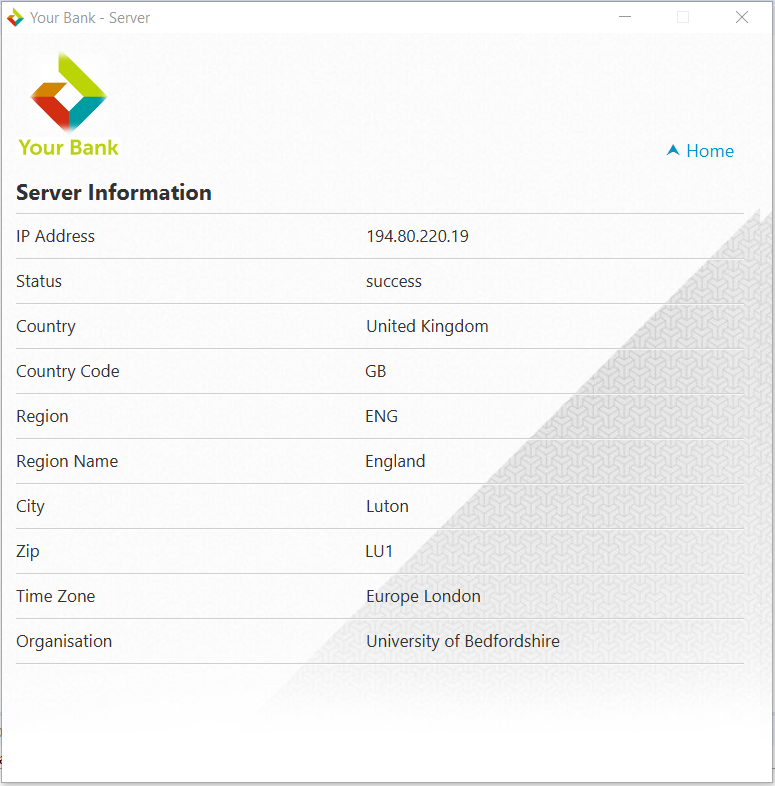
Analyse Survey Result View

A view that shows an analysis of a list of survey answered by customers via the client system.



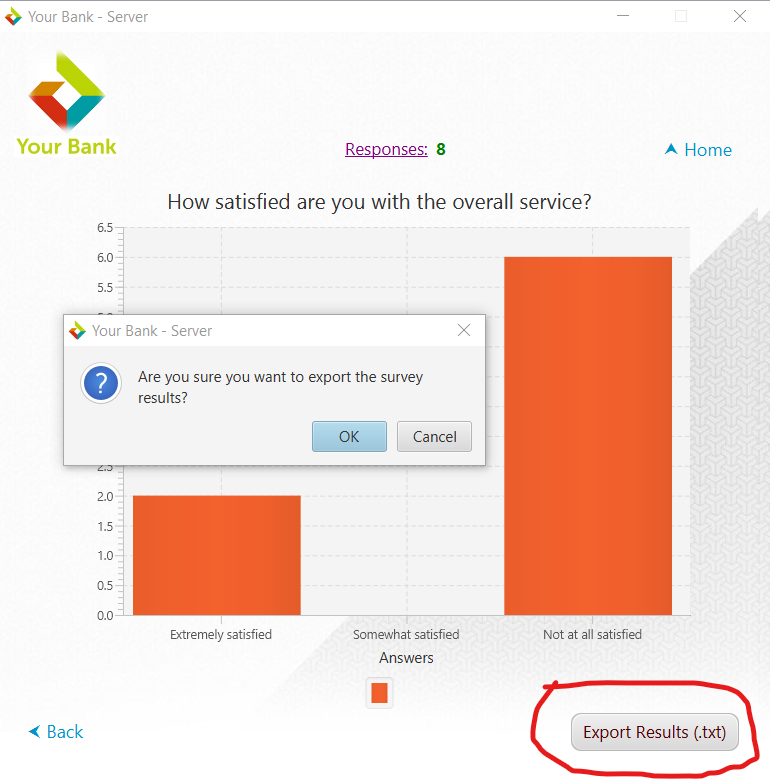
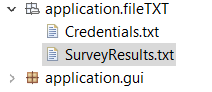
Server Information View

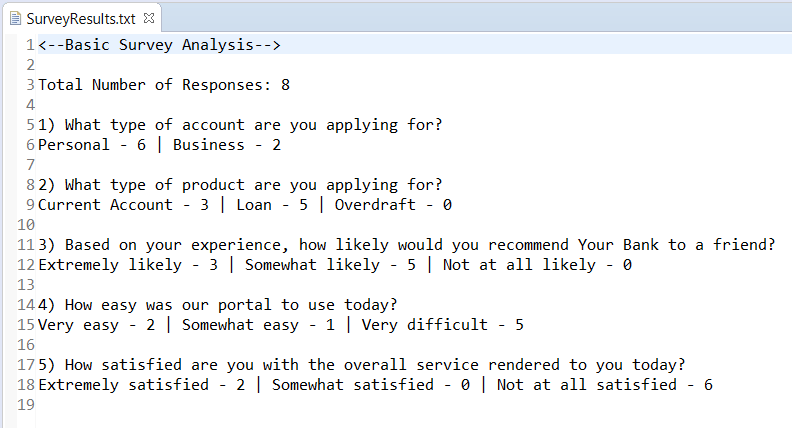
A view that shows the server information. Information is gotten through a web API using the system ip address as the key.



Export Results (.txt)

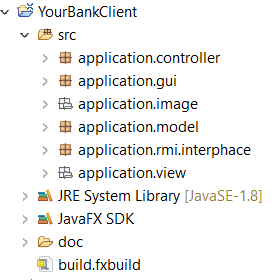
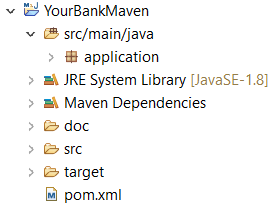
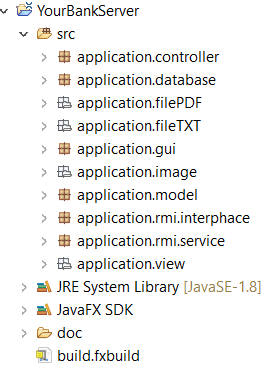
Exports the result analysis of all the responses collected from the client into a text file called SurveyResults.txt.





# **IMPLEMENTATION**

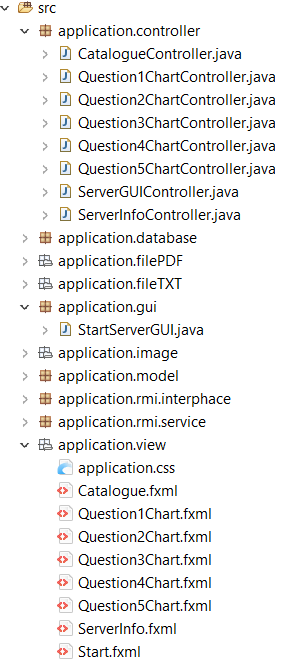
Client and Server Implementation

## 4.1 SERVER

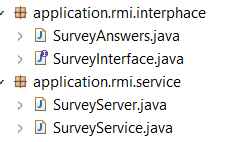
The server side of the system serves as the administrative side of the system software. It consists of several functionalities. The server side talks to a Maven project for an advanced type of API.

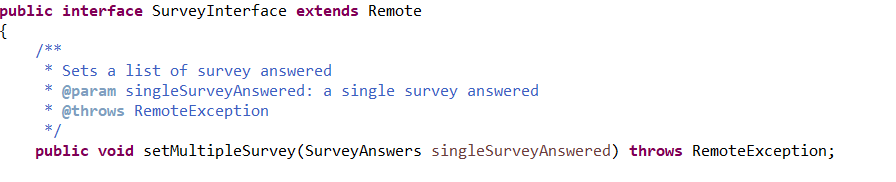
## 4.1.1 GRAPHICAL USER INTERFACE

The GUI consists of .fxml files that are controlled by several controller class. The .fxml files are located in the application.view package while the controller classes are located in the application.controller. The gui can be started with the class called StartServerGUI.jar located in the application.gui package.

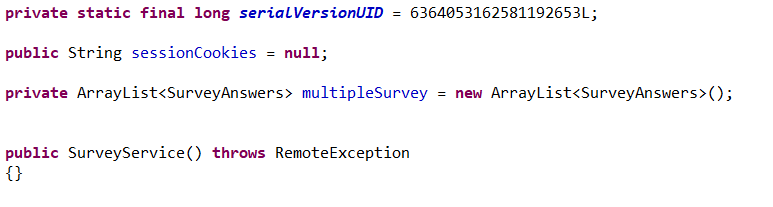
## 4.1.2 RMI

Package structure of remote method invocation.

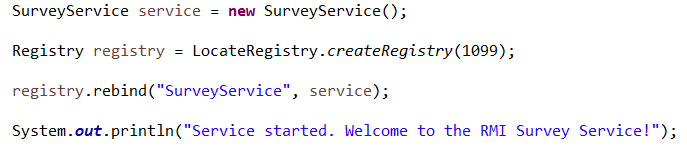


The remote interface class 

The service class implements the interface class



The server registers the service class into a registered port



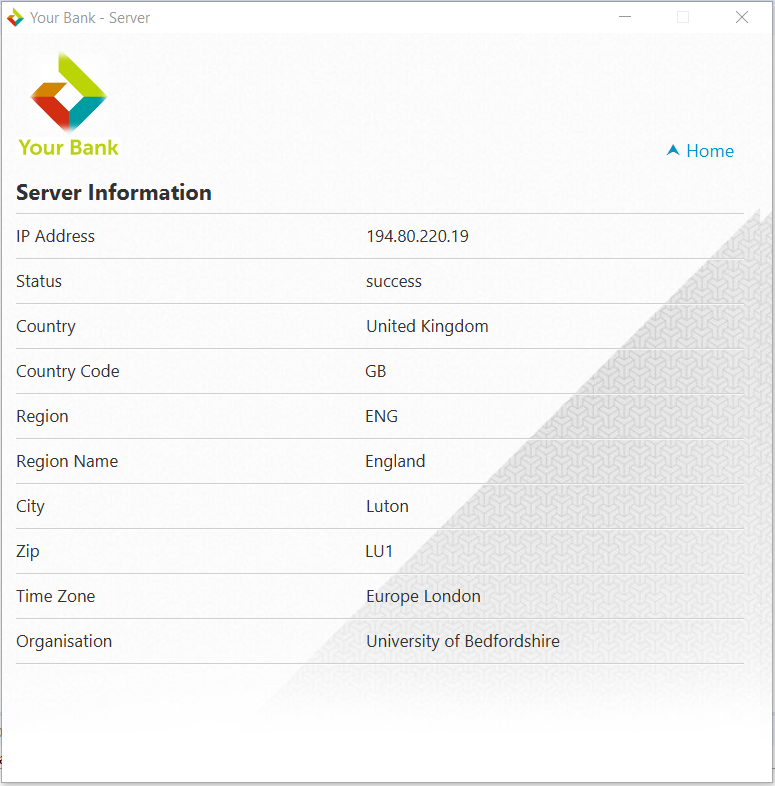
## 4.1.3 JSON / WEB API

First Web API

Initialises a request to an external geolocation web API service

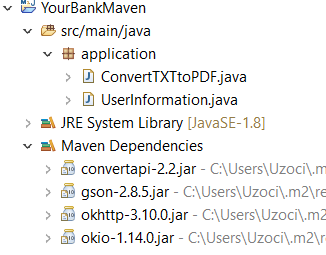


Displays the result unto the GUI.

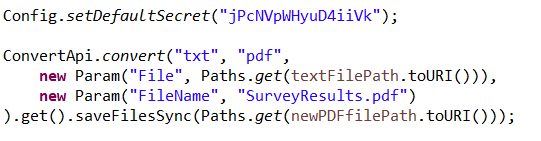
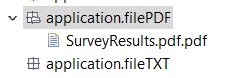


Second Web API

Maven project used to convert text document to a pdf document using an external web service



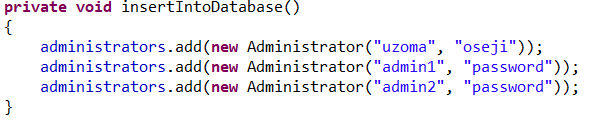
Converting text file to pdf using Convert API from the Maven project to the Server project



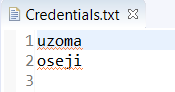
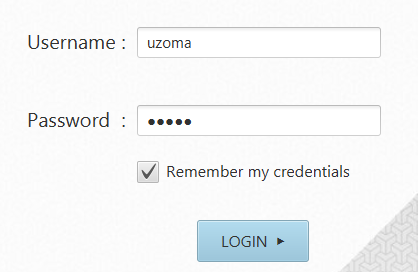
## 4.1.4 AUTHENTICATION / COOKIES

Database

Database where the admin login details are set

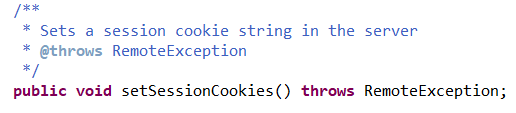


Credentials can be remembered with the aid of saving the credentials into a text file called Credentials.txt.

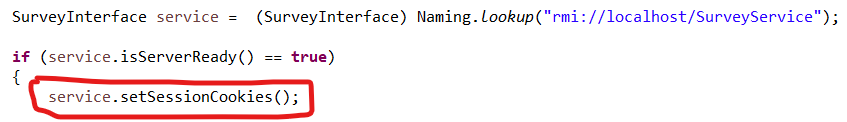


Cookies

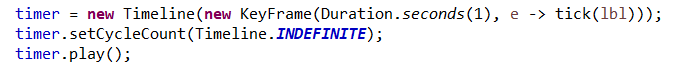
Cookies created in the rmi interface class.



Set in the client system.



## 4.1.4 TIMER EVENT



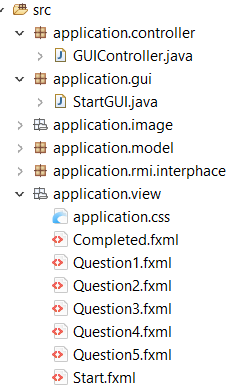


## 4.2 CLIENT

The client side of the system serves as the customer side of the system software. It works are a survey system collecting data and sending to the server.

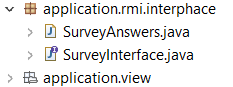
## 4.2.1 GRAPHICAL USER INTERFACE

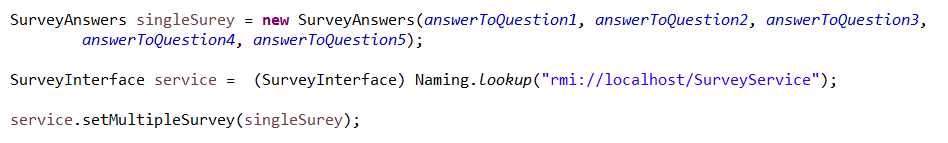
Structured packages



## 4.2.2 RMI

Remoted method invocation located used to connect the server and the client

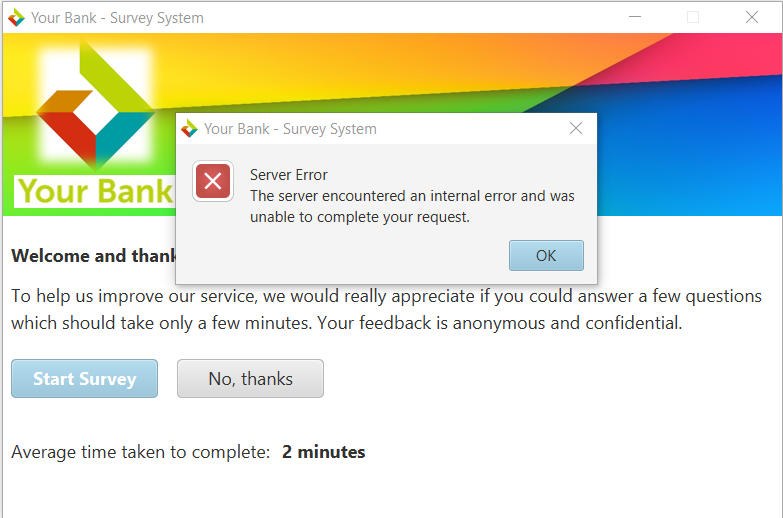




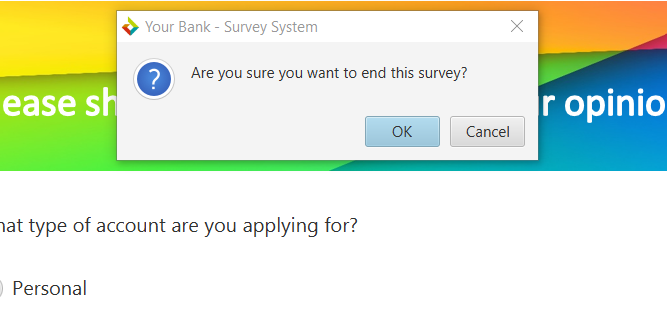
# **TESTING**

## 5.1 CLIENT GUI

Errors message shows when server is not connected.

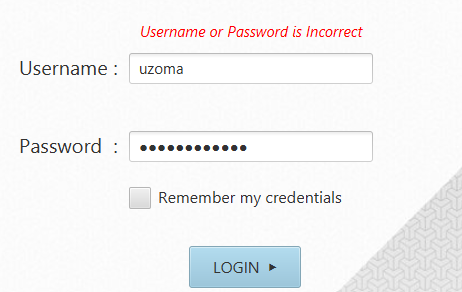


Confirmation message comes up when you try to end the system while the survey is in progress

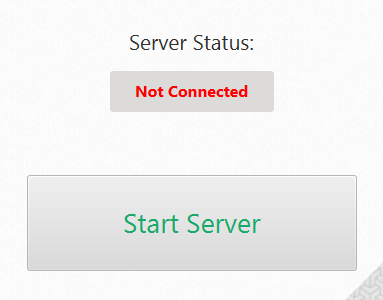


## 5.2 SERVER GUI

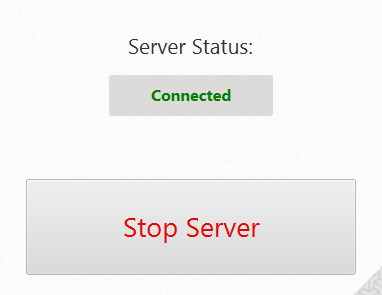
Error text comes up when the username of password is incorrect



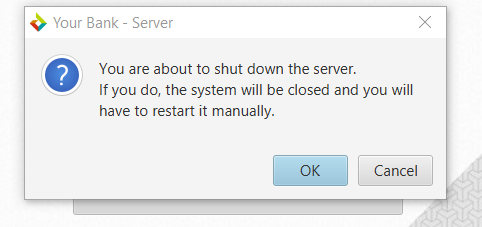
Server status when the server is not connected.



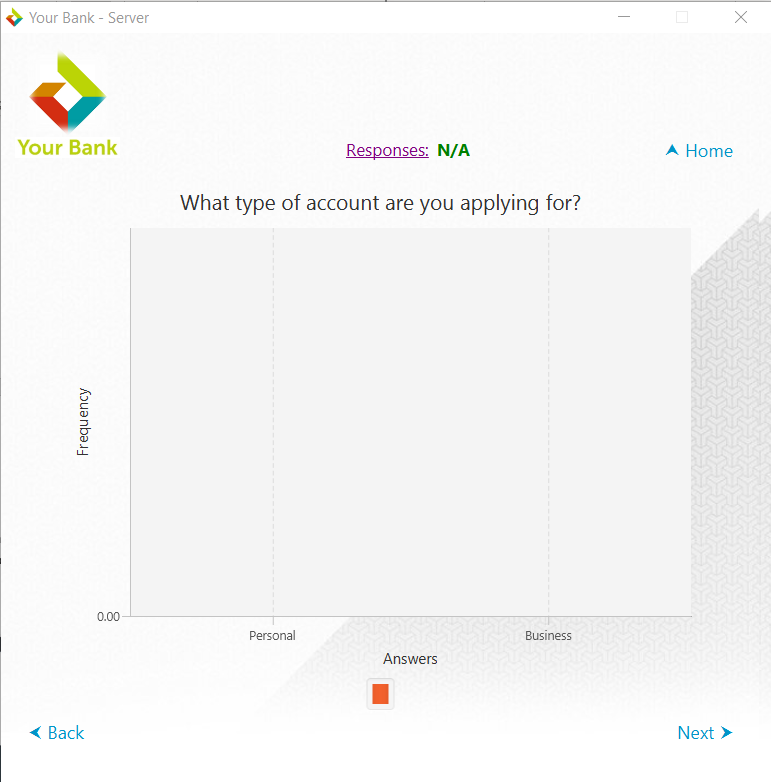
Server status when the server is connected



Confirmation message that comes up when the server is to be stopped.



Graph when there are no responses in the server.



Text file when there are no responses.

