

Project: Service Novigrad App

**SEG 2105[A] – Software Engineering
Fall 2020**

University of Ottawa

**Course Coordinator: Prof. Hussein Al
Osman**

Teaching Assistants: Faezeh

Olivia

Group # 17;

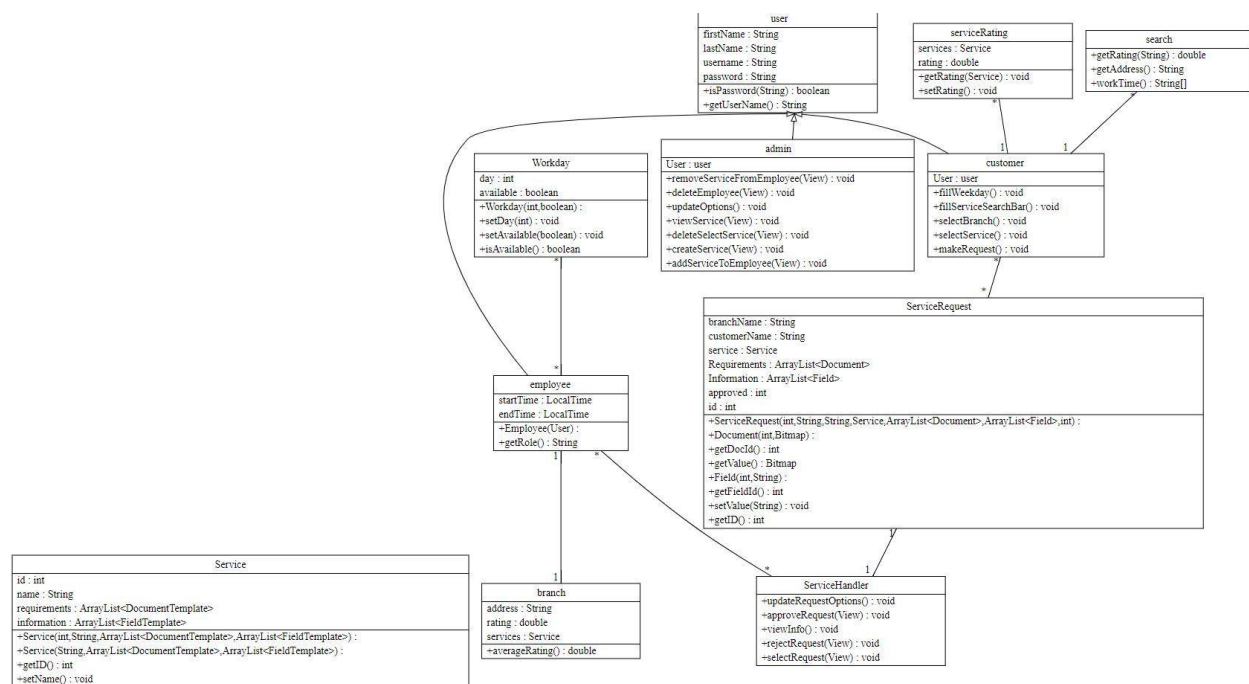
**Wyse Ebbah, 300141935
Kian Mozefarian**

Submission Date: December 6, 2020

INTRODUCTION

The main goal of this project is to create an app where users can request for services from Novigrad province's service branches. Generally, this indicates the use of a client-server framework. However, this implementation utilises only one locally stored SQLite Database and acts as both the "server" and client. In actual production, the workload will be split across multiple "computers" with one server system containing the database and multiple connected clients. Effectively, the project can be split into two parts, the interface and client interaction(Activities) and the objects manipulation and database(Classes). This allows for more clearly defined tasks and workload. The project mainly relies on programming techniques learnt during the course. These include UML design patterns, SQLite Database Manipulation, and JUnit Testing. However, other pieces of information are used in the production of the app, such as Image processing in java, and Activities and Views. The final project consists wholly of Linear Layouts and has a minimum API requirement of API 26. This is due to the nature of the app. The Service app will also be available to use at branches, public libraries etc and in-person. As a result, a higher API requirement is reasonable, due to the multitude of alternatives. The contributors to the creation of the app are Wyse Ebbah and Kian Mozefarian. Although Elliot Kirsch and Shinlong Chen are listed as members of the group, they actually did not contribute to the project and left the group before its production..

UML CLASS DIAGRAM



LESSONS

- Data and Class manipulation should be split across classes, rather than the database. If the database handler object is tasked with assembling classes and manipulating data, the code becomes clunkier, less testable and harder to understand. A better implementation will use factory classes that receive information from a database lookup class to assemble objects that actually perform the data manipulation and calculations.
- Understanding documentation is a huge part of programming. This is important as it is often necessary and much easier to use existing libraries in creating an app. To understand its use, programmers have to use documentation. In addition, the very language being used must be explored through documentation.
- The method of collaboration between group members is key. It is important to ensure that each group member has a reasonable idea of the progress of the other members in their section. This helps avoid a need to overcompensate for the incompleteness of the project. In addition, it is easier for member to help with each other's problems in the case that they arise. On the other hand, the group members have to get clearly defined and separate responsibilities to reduce the amount of "slack" each person gets, so the workload is distributed evenly. In addition, inactive members should be questioned earlier on to avoid to big a defect.
- The attributes of classes should contain as little data as possible and be connected as few complex classes as possible. This ensures the classes functions are well defined and require little assistance from external functions. In addition, when one class is changed, it reduces the amount of work that has to done in other classes. Basically, each class, package, or project should aim to be as self reliant as possible. As such, they can also be adapted really easily in another app.
- The best layout is the one with the most adaptability and least specificity. For example, linear layouts in this app showed much better results and where much more reusable than relative layouts. It is also very easy to make changes to the layout and modify its contents if the markup is simpler. However, when the layout is built, the design specifications can then be added to create a much better looking and easily maintainable app.

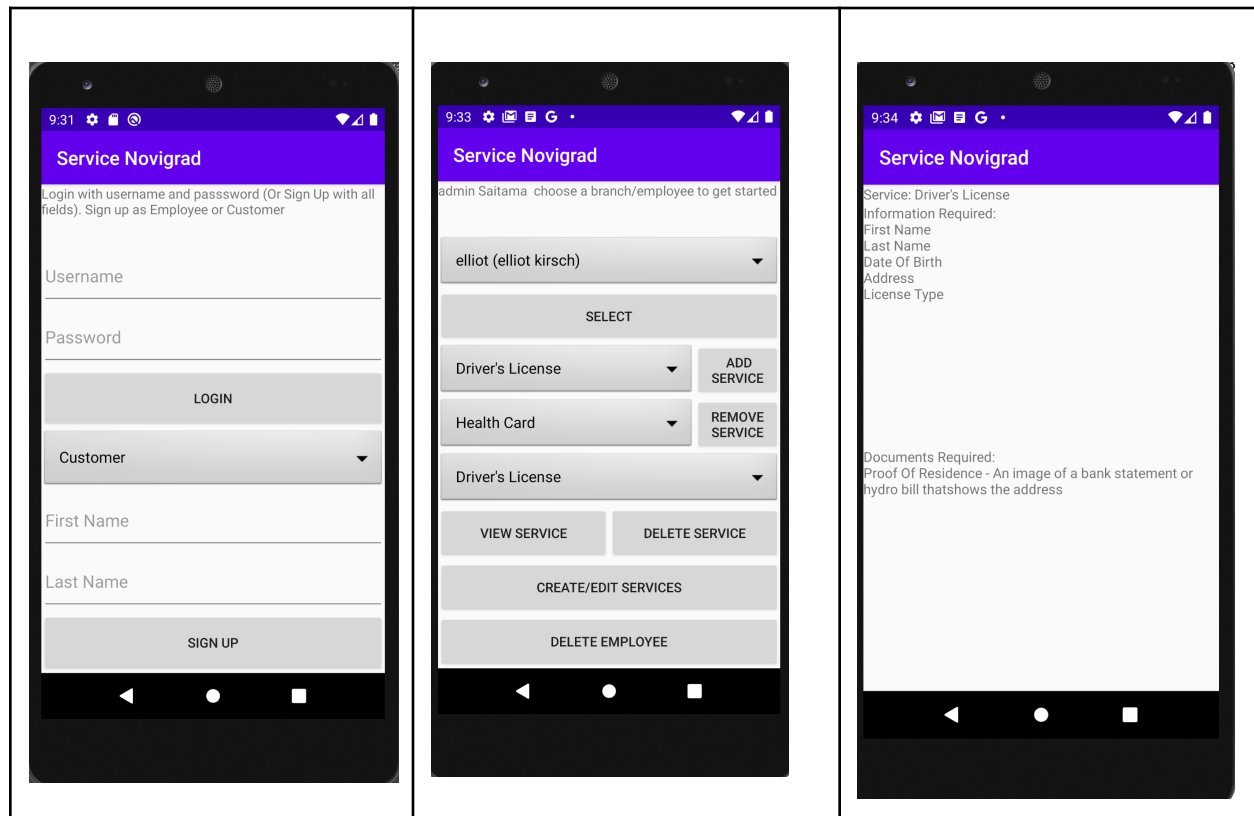
ROLES

Wyse Ebbah	Kian Mozefarian
Deliverable 1	
<ul style="list-style-type: none"> • Create Account Classes and Login Activity • Create MainActivity Layout 	<ul style="list-style-type: none"> • Create simple UML diagram regarding the basic needs of the project scope
Deliverable 2	
<ul style="list-style-type: none"> • Setup SQLite Database • Create Admin Layout and Functionality • Create Service Class and Functionality • Create Tests and Validation 	<ul style="list-style-type: none"> • Redefine the UML diagram to fit the realistic expectations had for the project, changed based on the behaviours of implemented classes • Adding description to admin class on the UML regarding the add, remove, and update branch services and employee status
Deliverable 3	
<ul style="list-style-type: none"> • Create ServiceRequest Class • Create Tests and Validation • Add Employee Information(Work hours) to Database • Create Employee Activity and Functionality 	<ul style="list-style-type: none"> • Update UML to define the employee class with its functionalities • Add the verification for checking close and open times • Assist in Employee XML layout design
Deliverable 4	
<ul style="list-style-type: none"> • Create ServiceRequest Class • Create Tests and Validation • Add Employee Information(Work hours) to Database • Create Employee Activity and Functionality 	<ul style="list-style-type: none"> • Update Customer functionalities & the document purchase/transaction between employee and customer in the UML diagram • Create Validation class for validating user inputs into the app

- Add Support for Images
- Implement Rating System
- Work on Report

- Work on Report

SCREENSHOTS



9:35

Service Novigrad

Choose a name for the service, then specify the information and documents for the service (Case Sensitive)

Library Card

SELECT/CREATE

Proof Of Residence

CHOOSE DOCUMENT

Proof Of Status

REMOVE DOCUMENT

License Type

ADD FIELD

First Name

REMOVE FIELD

9:37

This branch is open

Health Card (Offered)

VIEW SERVICE

Open Time 8:36 AM
9:37 PM
10:37 PM

Close Time 8:35 AM
9:36 PM
10:37 PM

SET TO TIMES

SET TIMES

Monday (Open)

VIEW SERVICE REQUESTS

9:38

Service Novigrad

Request 1 by wyse to elliot for Health Card (approved)

Request 1 by wyse to elliot for Health..

SELECT REQUEST

VIEW REQUEST INFORMATION

REJECT


APPROVE

9:38

Service Novigrad

Request 1 by wyse to elliot for Health Card (approved)
Last Name: ebbah
First Name: wyse

Document Of Document ID 1



9:39

Service Novigrad

Driver's License

SEARCH

Monday

TIME

SEARCH

(Enter a search query and) Choose a branch

Request 2 by wyse to chiggs for Driver's License (rejected) has been rejected
Request 1 by wyse to elliot for Health Card (approved) has been accepted

MAKE REQUEST

RATE

9:40

Service Novigrad

Driver's License

SEARCH

Monday

TIME

SEARCH

elliott (elliott kirsch) 3.5

elliott (elliott kirsch)

SELECT

Health Card

CHOOSE

MAKE REQUEST

RATE

9:42

Service Novigrad

Request 6 by wyse to Elliot for Photo ID

Date Of Birth

2000-1-25

ADD FIELD

Proof Of Residence

ADD DOCUMENT

REVIEW REQUEST

MAKE REQUEST

