**Android Project: Service Novigrad**

**SEG 2105 C - Introduction to Software Engineering**

**Fall 2020**

**School of Electrical Engineering and Computer Science**

**University of Ottawa**

Course Coordinator: Dr. Miguel Garzón

Teaching Assistant: Wassim El Ahmar

Pavly Saleh

300012936

Report Due Date: December 6th, 2020

Table of Contents

[Introduction 3](#_Toc58185165)

[UML Diagram 4](#_Toc58185166)

[Lessons Learned 5](#_Toc58185167)

[App Screenshots 6](#_Toc58185168)

[Login Page 6](#_Toc58185169)

[About 7](#_Toc58185170)

[Invalid Login 8](#_Toc58185171)

[Registering a User 9](#_Toc58185172)

[Input Validation 10](#_Toc58185173)

[Main Menu for an Employee 11](#_Toc58185174)

[Creation of Service Novigrad Branch Profile 12](#_Toc58185175)

[Selecting an Address for Profile 13](#_Toc58185176)

[Branch Hours 14](#_Toc58185177)

[Constraining user input with time picker 14](#_Toc58185178)

[Posting a Service – Employee 15](#_Toc58185179)

[See all Services – Employee Updated Point of View 15](#_Toc58185180)

[See Only Employee’s Offered Services 17](#_Toc58185181)

[Accepting/Rejecting Submitted Service Request 18](#_Toc58185182)

[Main Menu for Customers 19](#_Toc58185183)

[“Show me the services” – as a customer 20](#_Toc58185184)

[View Profile 21](#_Toc58185185)

[See Availability 22](#_Toc58185186)

[Book a Service 23](#_Toc58185187)

[Searching for a Service 24](#_Toc58185188)

[Searched Results 25](#_Toc58185189)

[Rating the Service 26](#_Toc58185190)

[See Ratings 26](#_Toc58185191)

[Admin Main Menu 27](#_Toc58185192)

[Show Me the Services – Admin Point of View 28](#_Toc58185193)

[Posting a Service 29](#_Toc58185194)

[Delete Users 30](#_Toc58185195)

# Introduction

GitHub repo: <https://github.com/SEG2105-uottawa/project_GR-55>

CircleCI: <https://app.circleci.com/pipelines/github/SEG2105-uottawa/project_GR-55>

Service is a new app for the residents of Novigrad to enquire about the services offered by the province. Service Novigrad Branches/Employees can offer various services to the citizens from a list of services provided by an Admin. Services such as driver's license, health card and photo ID are offered. Customers/Users can browse the offered services, and search for a Service Novigrad branch by name, rating, time availability, and address.

The application was coded in Java as well as XML using Android Studio. The built-in emulator, numerous unit test cases, as well as CircleCI were used to test the application. The application utilizes Firebase integration. Each service and user’s data are held in Firebase Realtime Database, hence, the application requires an internet connection to access said information.

Coding this application has given me some experience in a range of different technologies used in the field, such as and not limited to: GitHub, continuous integration, implementation of different features, releasing updates, as well as effective troubleshooting.

Table 1 shows the contribution to each deliverable in percentage.

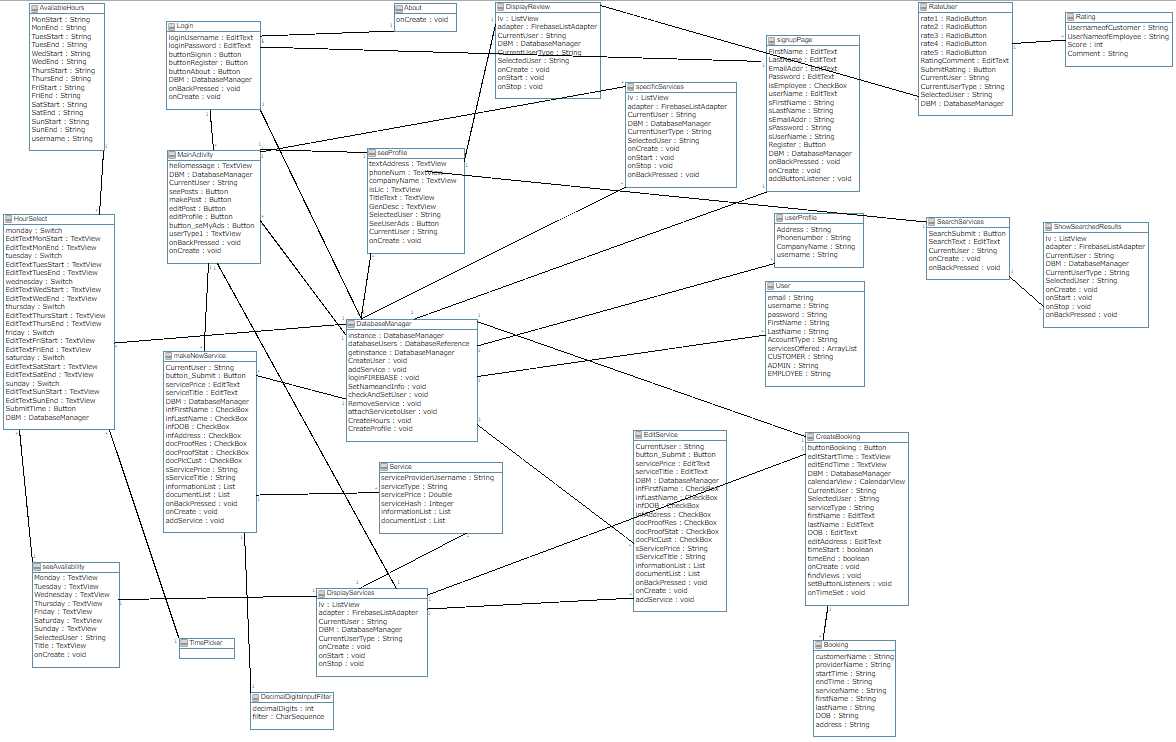
Table 1: Contribution to each deliverable

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Team Member | Deliverable 1 | Deliverable 2 | Deliverable 3 | Deliverable 4 |
| Pavly | 100 | 100 | 100 | 100 |

# UML Diagram

The following is a copy of the UML diagram for this project. As the picture includes multiple elements, which might make it difficult to read, it will also be included as a PDF inside the deliverable submission folder, it will also be found in the GitHub repo.

Created using Umple: <https://cruise.umple.org/umpleonline/umple.php?model=201102erfzkqjx9k05#genArea>



# Lessons Learned

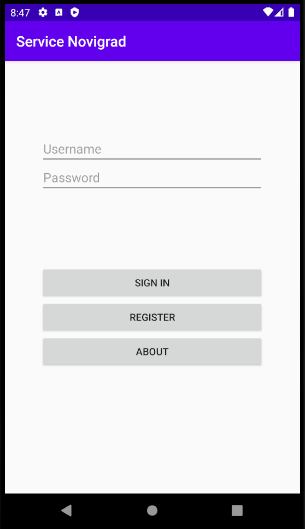
Software development according to best procedures can be difficult to maintain on a large scaled app. Though difficult, it sets up the process to be easier to expand on and allows for better collaboration across team members. It was noticeable in developing this app, as the features that were put together quickly were not as well made as those that followed proper design patterns. This project also taught me the importance of time management. As I was essentially my own group, I had no teammates to rely on to pick up my slack. I was by myself, hence, I had to organize accordingly. I had to manage my time efficiently so that I am able to complete all the objectives of the project, with plenty of time to test and make sure the features implemented work accordingly. I also had to develop proper troubleshooting techniques, as developing an entire app that is meant for a group of 3-5 to complete together is no walk in the park. Numerous issues arose, from buttons not showing up depending on the user type, to firebase not updating the info, not to mention the various headaches caused by CircleCI continuously running out of credits and displaying a very unattractive FAILED button on the project’s GitHub repo page. Once again, I had to be able to navigate through multiple StackOverflow discussion forums, as well as read multiple Java documentation to fix any issue that displayed itself. This proved to be very useful, as I am now a more capable programmer, having taken on this project by myself. It also taught the importance of teamwork, as it was not an easy task to complete deliverable after deliverable by myself, having no second opinion to inquire. I learned to be decisive, which is something I needed to work on in my everyday life, as I am not picky and very indecisive. This project was very useful in advancing my technical abilities, shown by me completing the project and implementing all features asked, and then some, as well as helping me develop qualities that will be useful in my everyday life.

# App Screenshots

## Login Page

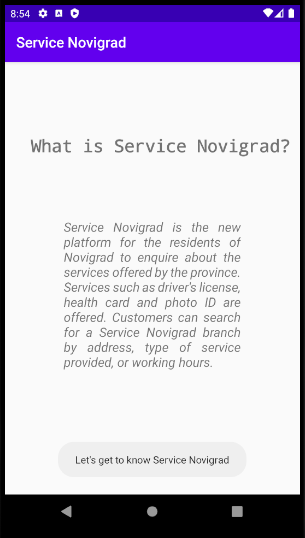
When you first open Service Novigrad, you are greeted with this login page. From this page the user has multiple options:

* They can sign in, if they already have a registered account
* They can register and create an account
* They can check out the About page



## About

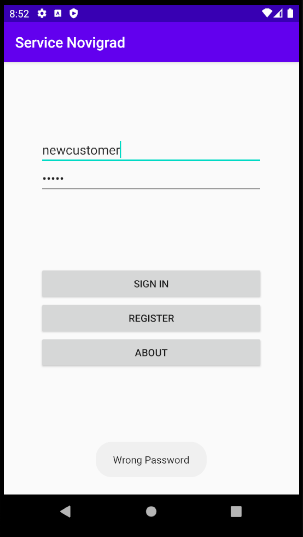
This is just a simple extra page that gives users a brief description of the app, its purpose and the provided functionalities. It is the app’s “About Us” page.

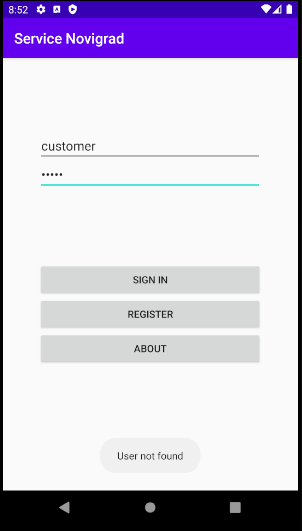


## Invalid Login

One feature that was implemented was that the login information is validated to make sure users are indeed logging in with their correct username and password. 2 different error messages can be displayed:

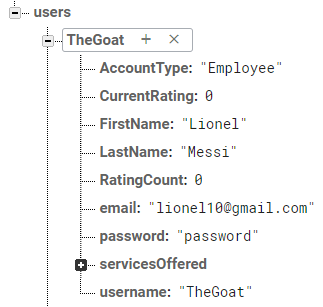
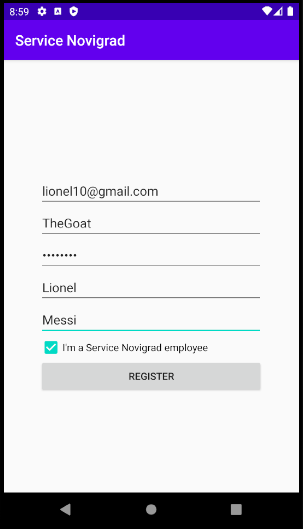
* “User not found” – This message lets the user know that the username provided does not match any registered username on the firebase database
* “Wrong password” – This message lets the user know that their username is in fact in the database, but the password provided does not match the username provided



* 

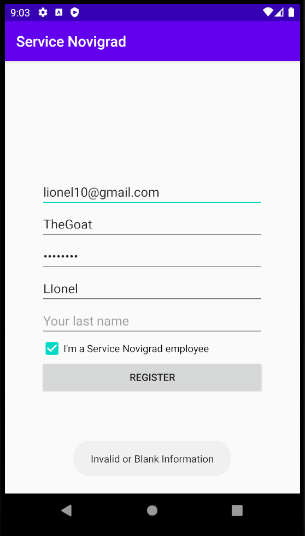
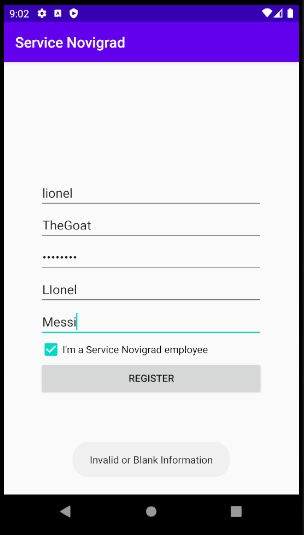
## Registering a User

In this section, new users are able to register an account to be able to use the app. The user can specify if they are a Service Novigrad employee or just a regular customer, once a new user is created, firebase automatically updates and adds the new user to the database.

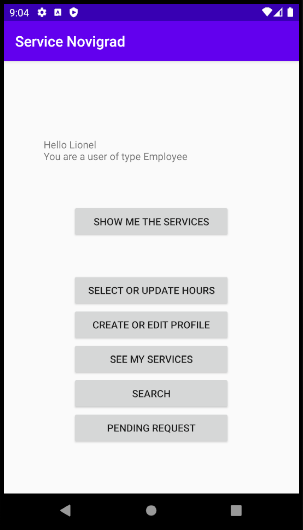
## Input Validation

In here, this simply shows that a user cannot register if they do not have a proper email address or if they leave a field blank.

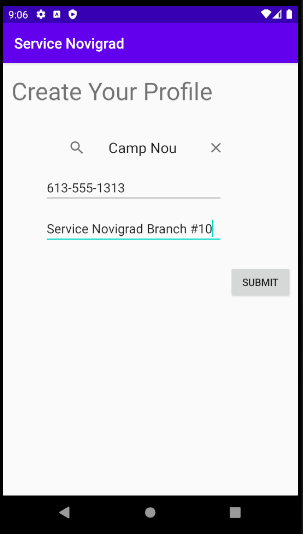
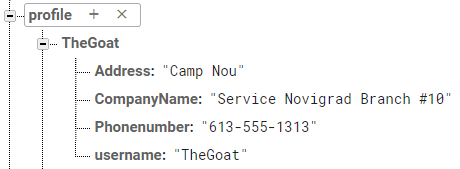
## Main Menu for an Employee

This shows the different options available for a user of account type employee



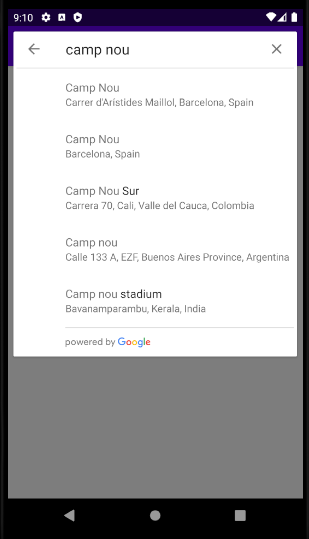
## Creation of Service Novigrad Branch Profile

Here, users of type employee are associated with a branch. The employee is tasked with creating a profile for their branch. Once their profile is created, this is updated on firebase as can be seen.



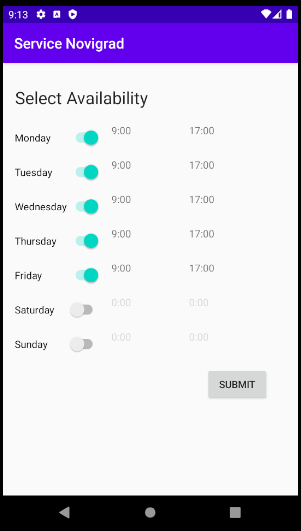
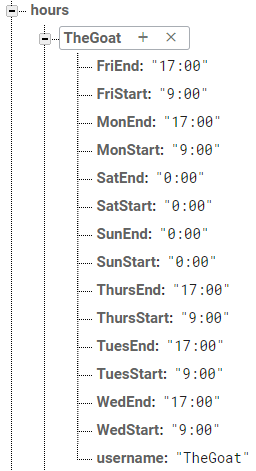
### Selecting an Address for Profile

Using the Google Maps Places API, I implemented an autocomplete fragment, which made input validation easier for me, and provides users with a familiar experience as plenty of other apps use this feature as well.



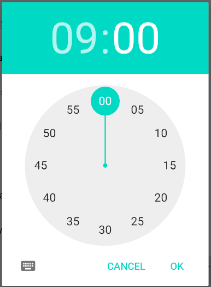
## Branch Hours

In here, the employee is tasked with providing the hours of operations for their branch, once completed, this is also reflected on firebase



### Constraining user input with time picker

Employee hours inputs are only to be picked as time values. This is specified as their inputs are constrained using a time picker class.

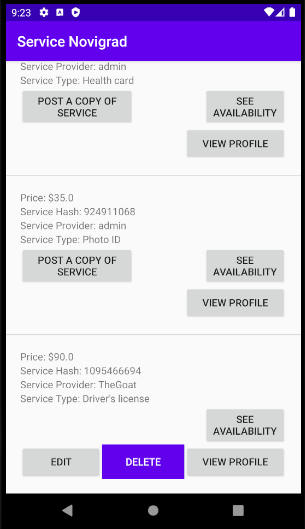


## Posting a Service – Employee

Since employees can only post services based on those provided by the admin, they go on the page that displays all the services. With that, they are able to post a copy of a service posted by the admin. Once they press on “Post a copy of service”, this is reflected in firebase

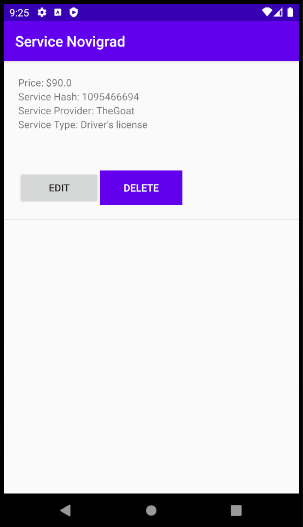
## See all Services – Employee Updated Point of View

Once the employee posts a copy of the service they wish to offer at their branch, the view changes. Here, they can repost an admin approved service, see profiles and availabilities of other branches, and manage their own posted service.



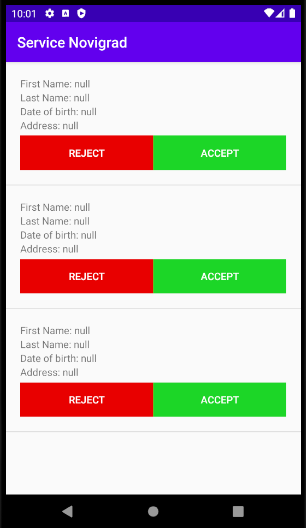
## See Only Employee’s Offered Services

When clicking on “See my services” from the main page, only the employee’s currently posted services are shown



## Accepting/Rejecting Submitted Service Request

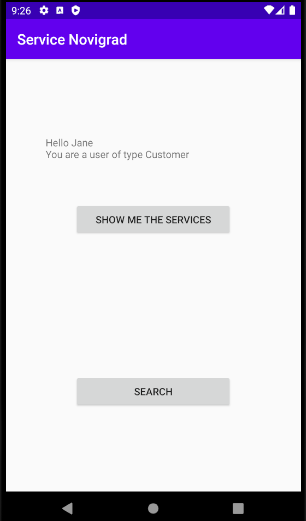
Users of type employee as well as admin can accept or reject a service booking request submitted by a customer.



## Main Menu for Customers

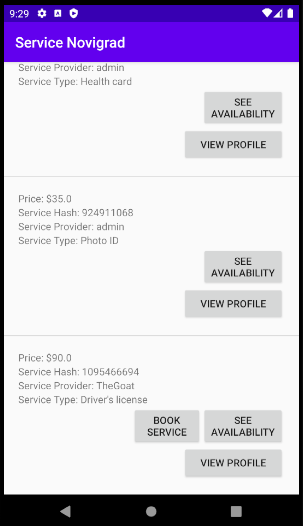
Two options are available for a user of type customer:

* Show all services available
* Search for a specific service



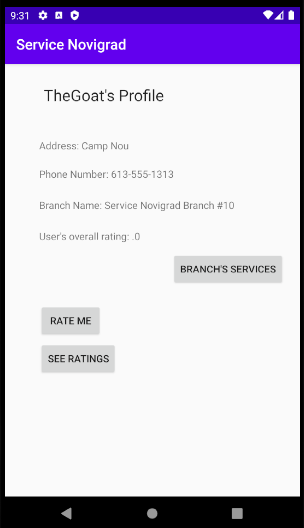
## “Show me the services” – as a customer

Here the user can look through all posted services and perform actions on specific ads



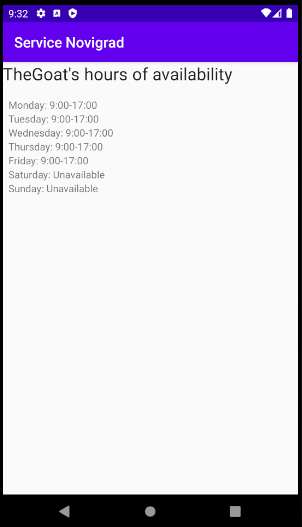
## View Profile

The branch’s previously configured profile populates from firebase. Options for further information also pop up here.

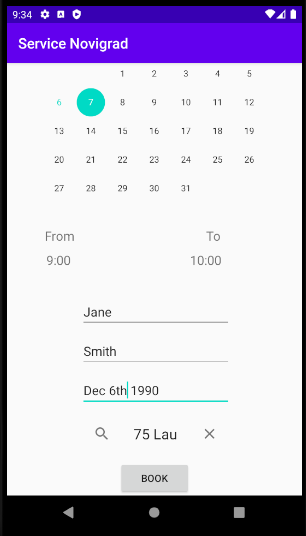


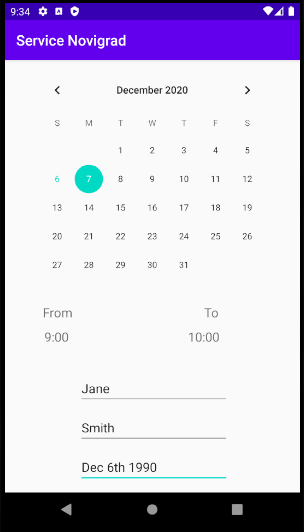
## See Availability

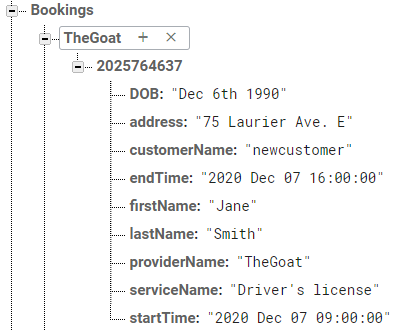
The branch’s previously configured hours populate from firebase.



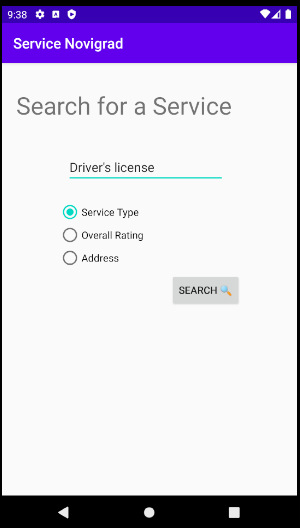
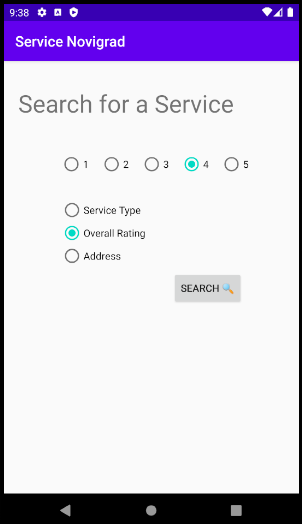
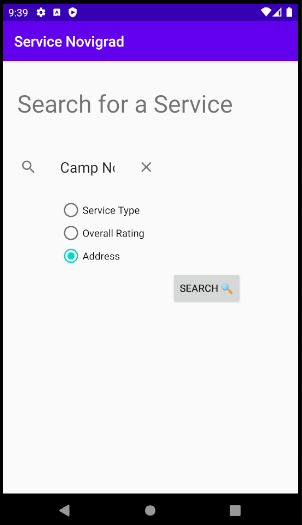
## Book a Service

Customers are able to book a service, complete the required information for that service, and submit their request. Their booking shows up in firebase once submitted



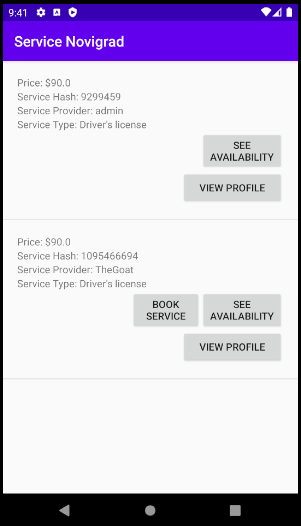


## Searching for a Service

Users have the ability to search for a service by different terms

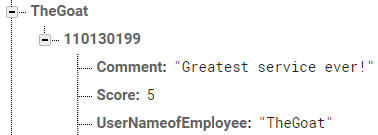
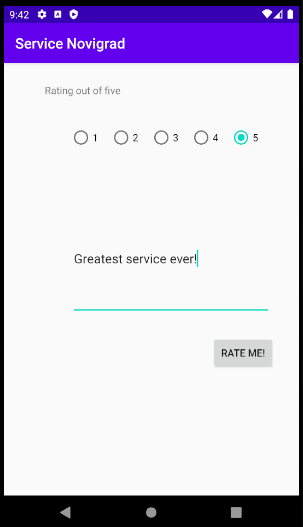
## Searched Results

Once they perform their search, a list displaying the results appears. Here, similarly to the see all the services menu page, users can perform actions on the displayed services results



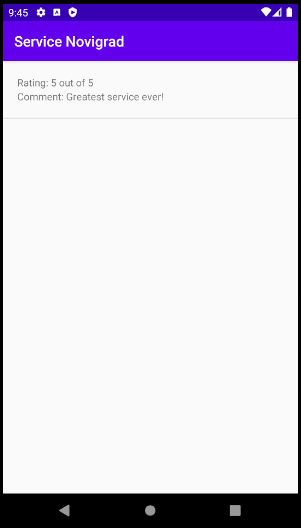
## Rating the Service

After the service, customers can rate the branch/employee and leave a comment! This once again reflected in firebase



## See Ratings

Once a rating has been submitted, users can see all ratings and scroll through them if there are multiple ones available

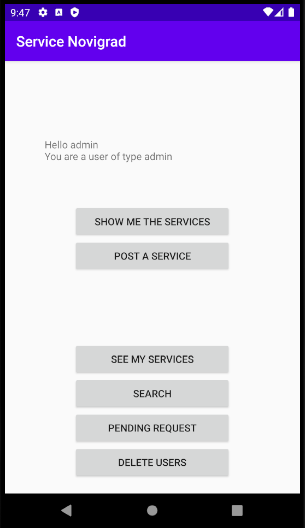


## Admin Main Menu

Multiple options are available for a user of type admin once they login

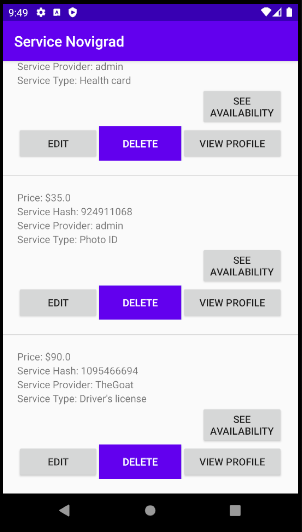
* Username: admin
* Password: admin

They are the only type of user that can post a service (employees can only copy a service that was posted and approved by the admin



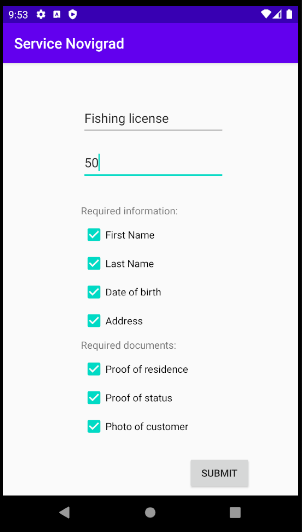
## Show Me the Services – Admin Point of View

Admin has the ability to edit and delete any service in the list, even if it’s not theirs



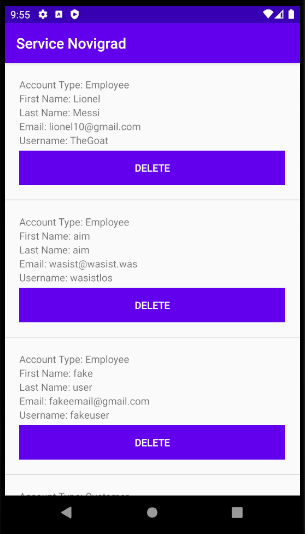
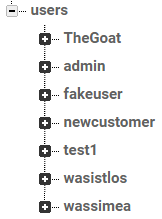
## Posting a Service

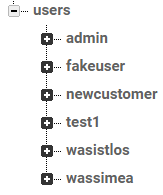
Admin can post a service and specify the details, as well the required information/documents of the customer. Once submitted, this is reflected and updated on firebase



## Delete Users

As an admin, this user has the ability to delete any user registered on the app and found in the database. Notable omission is the fact that the admin user account is not displayed on this list, so the admin does not accidentally delete themselves. Once a user is deleted, they are also removed from the firebase database.

  Before

 After