SEG2105 – Introduction to Software Engineering – Fall 2020

Android Project: Service Novigrad (20%)

This application will implement basic functionality for services offered by an imaginary province called Novigrad to its residents. Think of the services offered by Service Novigrad to be similar to those provided by Service Ontario or Services Québec which allow citizens to obtain a driver's license, photo ID, or health card.

INSTRUCTIONS

1. The project will be done in the same team you joined for your Labs.
2. You will submit each deliverable as a release on GitHub
3. The team must present only one version of the application. For instance, one student having one screen with the search functionality and other one having another different screen (running on a different phone) with the services provider functionality, WILL NOT be accepted. The team must produce a single application with all the required functionality.

The purpose of this project to expand on the theoretical work, allowing students to gain practical experience implementing the concepts learned in class. This project is also designed to allow students to learn how to work with their colleagues and develop mobile applications. Learning outcomes range from increased understanding of concepts relating to software engineering, to overall knowledge of programming for android, management and team-relation skills.

The main outcome of the project is the implementation of a Service Novigrad application for android devices. Students are to implement all components of the project, from their design specification, UML models, graphical assets, and source code. Students are encouraged to use the available toolset in android studio but should refrain from copying whole blocks of code from the internet to implement features. Should a group want to use a non-standard tool/API they should request permission before doing so.

The app will be conceived with three different types of users in mind. The administrator, the Service Novigrad branch employee, and the customers. The administrator manages all the possible services that can be offered to customers by different Service Novigrad branches. The branch employee creates a profile for the branch, selects the services offered by that branch, and specifies the working hours. The customers should be able to search for a Service Novigrad branch by address/type of service provided/working hours.

The features that should be available to each type of user are given below. Note that those are the minimum required features, and you are free to add more features you think might enrich your application.

The administrator can:

1. Login to an administrator account - the developer should pre-create such an account (e.g. username: *admin*, password: *123admin456*)
2. Create, edit, or delete services (at least 3 as elaborated below, but the more the better) that could be offered by Service Novigrad branches, and specify the required information and documents that need to be supplied by a customer requiring a service:
    a. Driver's license:
        i. Form: Including customer first name, last name, date of birth, address, license type (G1, G2, G)
        ii. Documents: Proof of residence (An image of a bank statement or hydro bill that shows the address)
    b. Health card:
        i. Form: Including customer first name, last name, date of birth, address
        ii. Documents: Proof of residence (An image of a bank statement or hydro bill that shows the address) – Proof of status (An image of a Canadian permanent resident card or a Canadian passport)
    c. Photo ID:
        i. Form: Including customer first name, last name, date of birth, address
        ii. Documents: Proof of residence (An image of a bank statement or hydro bill that shows the address) – A photo of the customer
3. Delete accounts of branches and customers.

The Service Novigrad branch employee can:

1. Create an account for the branch and login to that account
2. Select services provided
3. Enter the working hours of the branch
4. View submitted service requests, and either approve or reject each request

The customer can:

1. Create an account and login to that account
2. Search for a Service Novigrad branch by address/type of service provided/working hours
    - You should also show the rating for each service found in the search process
3. Select a service they would like to purchase
4. Fill in the required information and documents, then submit the service request.
5. Rate their experience with the Service Novigrad branch

Note: This course does not focus on interface design; hence, we do not focus on usability aspects. However, students are encouraged to "beautify" their projects, should they be comfortable with user interface design. Consider the Android Design Guidelines when designing your application. This topic will be covered in a tutorial session and detailed information is available at: https://developer.android.com/design/index.html

---

DELIVERABLES

The project is divided into 4 incremental deliverables. Students are required to submit each deliverable by the posted deadline online using Brightspace.

Deliverable	Due date
1 – Github repository and user accounts (3%)	October 19
2 – Admin functionality (3%)	November 1
3 – Service Novigrad user functionality (3%)	November 22
4 – Complete application (9%)	December 6
5 – Demo (2%)	Last week of classes

The project is to be carried out throughout the session and students are strongly encouraged to maintain a log of their project activities, as task allocation and project flow are components of the final report that you will submit at the end of the project. We suggest students keep track of duty assignment, with complexity of allocated tasks and completion dates.

Your application must be written in Java and built using Android Studio (you can use another IDE, but the TAs will only provide support for Android Studio). You should compile your project against the earliest possible SDK version allowed by the API methods you are using. By the end of the semester, you must implement and submit a working application based on the specifications. Firebase or SQLite can be used for storing and retrieving the application data.

ACADEMIC HONESTY

All work that you do towards the fulfillment of this course's expectations must be your own unless collaboration is explicitly allowed. Viewing or copying another individual's work (even if stored in a public directory) or lifting material from a book, magazine, website, or other source-even in part-and presenting it as your own constitutes academic dishonesty, as does showing or giving your work, even in part, to another student.

DELIVERABLE 1

You need to join our GitHub classroom, create your team, and accept this project. Please refer to "Steps to Create your Team on GitHub" for detailed instructions.

In this deliverable, you need to implement the user account management component. That is, the app needs to allow users to create user accounts.

To simplify the development, there will be a single pre-created admin account (e.g. username: *admin*, password: *123admin456*), but it should be possible to create as many Service Novigrad branch employee accounts and customers as desired.

Once the user logs in, they should see a second screen with the following message, 'Welcome firstname! You are logged in as "role". No additional functionality needs to be implemented at this point.

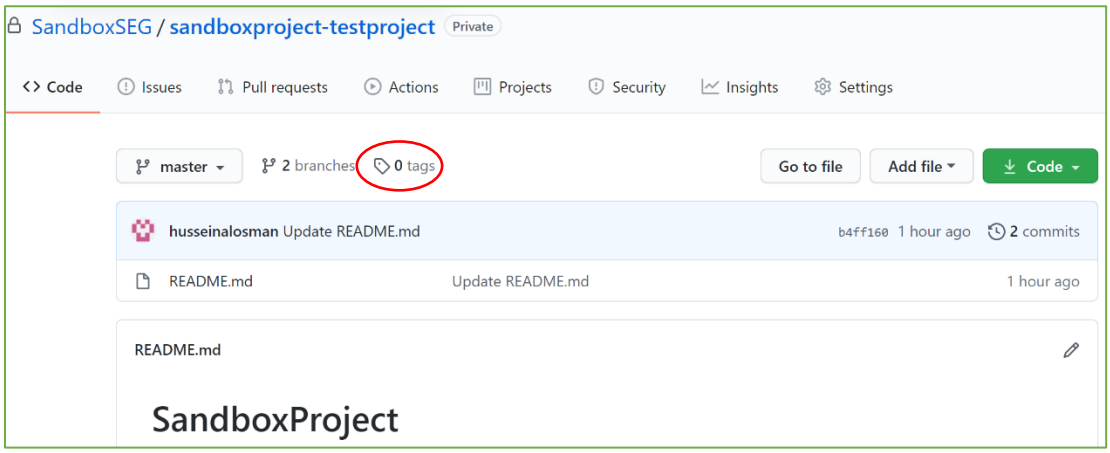
You can use Firebase or SQLite for DB support. If you do not use a DB at this point, you can simply store the information in memory (would be lost when you terminate the app) or on a file.

In your github repository, include a PDF document that contains an UML Class diagram of your domain model. This will only include the UML classes related to deliverable 1. Moreover, in your repository, provide a README file that contains the credentials needed to sign-in as admin (if a predefined admin account has been created).

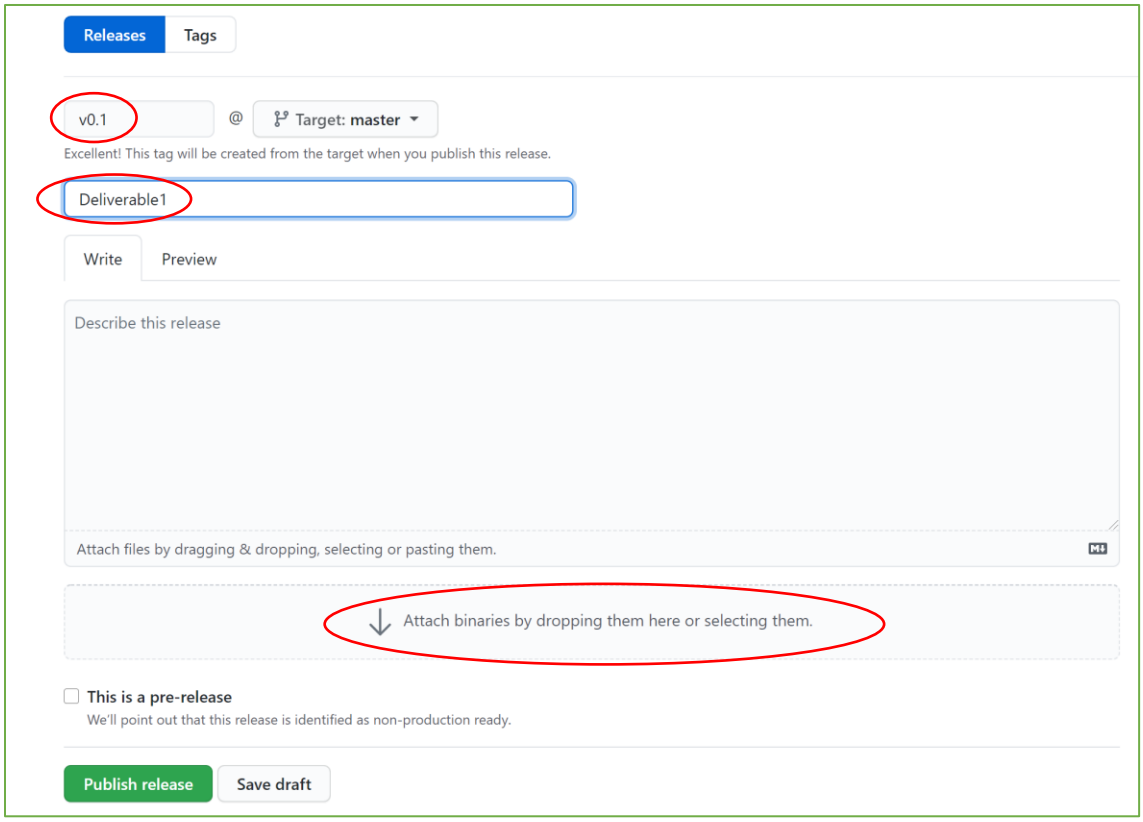
How to submit:

To submit your code, you will create a release from your repository as follows:

1. In your repository, click on "tag" (see figure below)



2. Click on the "Create new release" button and fill the form as follows (see figure below)
    a. For "Tag version" enter v0.1
    b. For "Release title" enter Deliverable1
    c. From the section of the form where you can attach binaries, upload the APK of your app.
    d. APK can be found in <yourAndroidProject>/app/build/outputs/apk/app-debug.apk. Name the APK after the name of your team "group1_debug_apk".



Deliverable 1 marking scheme:

Feature or Task	% Weight (out of 100)
Github: Team created in Github classroom contains all members of the group	10
Github: Each member of the group has made at least ONE commit to the repository.	20
UML Class diagram of your domain model (-2 for each missing class) (-2 for incorrect generalization) (-0.5 for each incorrect multiplicity) (-0.5 for each missing attribute)	30
APK submitted	5
Can create a Service Novigrad branch employee account	10
Can create a customer account	10
Can see the 'Welcome screen' after successful authentication. Can see the user role Can see the name or username associated to the account	5
Fields are validated in all screens (e.g. you cannot enter an invalid email, name, etc) (-1 for each field in which the user input is not validated)	10
OPTIONAL - Group uses a DB (e.g. Firebase or SQLITE, or other similar technology)	+5 (bonus)

DELIVERABLE 2

In this deliverable, you need to implement the Admin related functionality. That is, the app should allow the admin user to add, edit, and delete services that can be offered by Service Novigrad branches using the application. For each service to be created, a list of required customer information and documents needs to be specified.

In addition, the admin account must be able to delete Service Novigrad branch accounts (branch account is the same as employee account) and customer accounts.

You must include at least 5 unit test cases (simple local tests). There is no need to include instrumentation or Espresso Tests (UI).

In your github repository, include a PDF document that contains an UML Class diagram of your domain model. This will only include the UML classes related to deliverables 1 and 2. Moreover, in your repository, provide a README file that contains the credentials needed to sign-in as admin (if a predefined admin account has been created).

How to submit:

To submit your code, create a release v0.2 in your repository. Make sure the APK file is added to your release.

Notes:

- Categories and subcategories of services are optional. This can however help you organize your screens (imaging listing 50 services in a single screen).
- Make sure your APK is correctly generated.
    - Go to Build -> Build Bundle(s) / APK(s) > Build APK(s)
    - Android Studio saves the APKs you build in project name/modulename/build/outputs/apk/
    - Use the _Debug one for submission.
    - For more info: https://developer.android.com/studio/run/

Deliverable 2 marking scheme:

Feature or Task	% Weight (out of 100)
Updated UML Class diagram of your domain model (-2 for each missing class) (-2 for incorrect generalization) (-0.5 for each incorrect multiplicity) (-0.5 for each missing attribute)	10
APK submitted and ALL features working	5
5 Unit test cases (simple local tests). No need to include instrumentation or Espresso Tests (UI)	30
At least 3 services can be offered by Service Novigrad branches using the application implemented. A service has a name and list of required information and documents	15
Can remove services that are no longer being offered	15
Can edit services	15
All fields are validated. For instance, you should not be able to create a service with no name. This should be implemented along with valid error messages. (-1 for each field in which the user input is not validated)	10
OPTIONAL – Integration with CircleCI to see the automated builds and automated testing.	+5

DELIVERABLE 3

In this deliverable, you need to implement the Service Novigrad branch employee related functionality. That is, the app should allow branch employees to complete the branch profile and associate their branch to a set of services (that were created by the admin). The branch employee must also be able to set the working hours of the branch.

You must include at least **2 additional unit test cases** (simple local tests). There is no need to include instrumentation or Espresso Tests (UI).

In your github repository, include a PDF document that contains an UML Class diagram of your domain model. This will only include the UML classes related to deliverables 1, 2, and 3. Moreover, in your repository, provide a README file that contains the credentials needed to sign-in as admin (if a predefined admin account has been created).

Make sure your APK is correctly generated.

- Go to Build -> Build Bundle(s) / APK(s) > Build APK(s)
- Android Studio saves the APKs you build in project name/modulename/build/outputs/apk/
- Use the _Debug one for submission.
- For more info: https://developer.android.com/studio/run/

How to submit:

To submit your code, create a release v0.3 in your repository. Make sure the APK file is added to your release.

Deliverable 3 marking scheme:

Feature or Task	% Weight (out of 100)
Updated UML Class diagram of your domain model (-2 for each missing class) (-2 for incorrect generalization) (-0.5 for each incorrect multiplicity) (-0.5 for each missing attribute)	10
APK submitted and ALL features working (see notes below)	5
2 Unit test cases (simple local tests). No need to include instrumentation or Espresso Tests (UI)	10
Create an account for the branch and login to that account	15
Can add services to the branch account (from the list of services added by the admin, the branch employee can select one or more services.)	10
Can delete services from the branch account when they are no longer being offered	10
Can specify branch working hours. You are free to implement this feature as you want. That is, with a calendar or by selecting predefined timeslots, by week, by day, etc. Usability is key for this feature!	20
Can view and approve or reject service requests submitted to the branch	10
All fields are validated. For instance, you should not be able to enter an invalid phone number or address for the branch. This should be implemented along with valid error messages. (-1 for each field in which the user input is not validated)	10
OPTIONAL – Can edit the working hours	+5

DELIVERABLE 4

In this deliverable, you need to implement the customer related functionality. That is, the app should allow customers to search for a Service Novigrad branch by address/working hours/type of services provided. The application must display the list of available Service Novigrad branches based on the user search and allow the user to submit a request.

You must include at least **5 additional unit test cases** (simple local tests). There is no need to include instrumentation or Espresso Tests (UI).

In your github repository, include a PDF document that includes the following (**this is your final report**):

- A title page
- A Short Introduction
- The UML Class diagrams of your domain model. This will include all the classes you developed.
- A section that discusses the lessons learned
- A table stating the roles and contributions of team members for each deliverable. You must add explanations in those cases where you find that the contributions were not fair.
- Screenshots of your app.

In your repository, provide a README file that contains the credentials needed to sign-in as admin (if a predefined admin account has been created).

Make sure your APK is correctly generated.

- Go to Build -> Build Bundle(s) / APK(s) > Build APK(s)
- Android Studio saves the APKs you build in project name/modulename/build/outputs/apk/
- Use the _Debug one for submission.
- For more info: https://developer.android.com/studio/run/

How to submit:

To submit your code, create a release v0.4 in your repository. Make sure the APK file is added to your release.

Deliverable 4 marking scheme:

Feature or Task	% Weight (out of 100)
Updated UML Class diagram of your domain model (-2 for each missing class) (-2 for incorrect generalization) (-0.5 for each incorrect multiplicity) (-0.5 for each missing attribute)	10
APK submitted and ALL features working Make sure you test your APK. An invalid APK will receive 0.	5
Final report including: - A title page (2.5 points) - Short Introduction (2.5 points) - Updated UML class diagram - Table with the roles in the team and contributions of team members for each deliverable. (15 points) - All the screenshots of your app. (10 points) - Lessons learned (5 points)	35
5 Unit test cases (simple local tests). No need to include instrumentation or Espresso Tests (UI). Test cases must be relevant to the features of deliverable 4	10
Can search for a Service Novigrad branch by: - address - working hours - type of services provided	10
Can fill in the required information and documents and submit the service request	15
Can rate a branch with a rating between 1 to 5 (or another rating system)	5
All fields are validated. For instance, you should not be able to select a date in the past when booking an appointment. This should be implemented along with valid error messages. (-1 for each field in which the user input is not validated)	10
OPTIONAL – Customer receives a notification when their service request is approved/rejected	+10