CPEN 321 Software Engineering
Winter 2023

M1: Android App with Node.js Back-end (Monday, September 18, 9pm PDT)

In this milestone, you will implement a simple **Android mobile app** (front-end) with a **Node.js server** (back-end).

This milestone is an **individual** assignment. Working in groups is strictly prohibited.

If any assistive AI technology was used in any stage of working on this assignment you **must** explain how it was used. No points will be deduced for documented use. However, undocumented use will be considered academic misconduct and will be treated accordingly.

Front-end App Requirements

- The mobile app must be written natively for Android in Java.
  - Using other tools and frameworks, e.g., React Native or Expo, is not permitted.
- The app must work on a Google Pixel 3 emulator running Android Q (API 29).
- The main app screen must have 4 buttons, as illustrated at the end of this document. Each button provides a separate functionality and must function independently of others.

Button 1: My Favorite City [15 points].

- Opens Google Maps and shows the location of your favorite city (excluding Vancouver), with the name of the city clearly visible as the screen opens, without the need to make any additional clicks or gestures.

Button 2: Phone Details [15 points]

- Displays on the screen the name of the city where the phone is located, phone manufacturer, and phone model.
  - On a real device, the phone manufacturer and model could be “Google” and “Pixel 3”.
  - On an emulator, it could be: “Google” and “Android SDK built for X86”.

Button 3: Login and Server Info [55 points]

- Opens the login screen of Google or Facebook to authenticate the user [20 points]
- Once the user is authenticated, it connects to the back-end and displays the following information:
  - Server public IP address (either IPV4 or IPV6) [5 points]
  - Client IP address [5 points]
  - Server local time (hh:mm:ss) [5 points]
  - Client local time (hh:mm:ss) [5 points]
  - Your name, obtained via a back-end API (first, last) [5 points]
  - The name of the user logged into the app with their Google/Facebook credentials (first, last) [10 points]

Button 4: Surprise us with any additional interesting implementation! [15 points]

- Creative ideas will receive higher marks.
- The following are not considered an "interesting implementation":
  - Google Maps and sign-in, as they are part of the app already.
  - Showing information about the device which can be obtained by a simple call to an Android framework API, without performing any additional logic (like Button 2).
  - Showing static text or image, available on the device or extracted via an http connection, without performing any additional logic.
- Useful/interesting functionality should include some logic. It can be implemented from scratch (e.g., a puzzle) or can incorporate external services (weather, Uber, Twitter, etc.)
  - A few examples of external API can be found here: https://github.com/public-apis/public-apis

Back-end Server Requirements

- The back-end must be implemented in Node.js.
  - If your back-end uses a database, you can choose between MySQL and MongoDB.
  - Using MongoDB Atlas/Realm is not allowed.
- You can host your back-end on a cloud infrastructure of your choice.
  - Most vendors, such as Amazon, Microsoft, Google, and IBM, provide free student
  - subscriptions.
  - The course staff can also provide you with a pre-paid subscription to Microsoft Azure. If interested, send a private request to the instructor on Piazza, with the title "Azure Access Request". In the body of the message, list
    - Your first name,
    - Your last name,
    - Your Microsoft account id, if available, or a UBC email address that will become your Microsoft account id.
- On the server, create three APIs, to return
  1. Server IP address
  2. Server local time, when the API is called (hh:mm:ss)
  3. Your first and last name

These APIs will respond to requests issued by Button 3 of the front-end app.

Submission

You must submit on Canvas:

1. An APK file of your working app.
2. A pdf file outlining
   - the purpose and implementation details of the functionality in Button 4.
   - whether any generative AI technology was used while working on this assignment and, if so, how.

Before submission, ensure that:

- The app works on a Google Pixel 3 emulator running Android Q (API 29):
  - Build the APK and install it on the emulator by dragging and dropping the APK onto the emulator screen.
  - Verify that all the implemented functionality is working correctly.

- The back-end server is up and running from the submission deadline until the assignment grades are posted on Canvas. If the server is unreachable when a TA grades your assignment, you will lose all marks.

A wireframe below illustrates how the application should look like: