Requirement Engineering

Deadline: October 20

Overview: In this milestone, you will work as a team to define the use cases and brainstorm the application design you will develop for the term project. Starting this milestone, all the members of each team should submit a "peer evaluation" (more details in the Tasks section).

Deliverables: Please follow the steps under the **"Tasks"** section to (1) understand the app description and features you need to implement, (2) identify use-case scenarios for the application, and (3) formally represent the requirements and initial design of the app. You must create a **"pdf file"** named **Team<team-ID>-Milestone2** and submit it to your TA in the room. Your deliverables are:

1. A pdf file that contains:
    a. Informal requirements
    b. Fully dressed use cases
    c. Class diagram
    d. Component Transition Graph (more details in the Tasks section)
2. Filling and submitting the peer evaluation form (by each team member)

Grading rubric: Please find the rubric below.

Activity	Points	Team/Individual
Informal requirements	10 (2 points each)	Team
Fully dressed use cases	20 (4 points each)	Team
Class diagram	30	Team
Component Transition Graph	30	Team
Peer evaluation	10	Individual
Total	100	**Team***

* Note that those team members that fail to submit the peer evaluation form **will lose 10 points compared to their teammates**. We accept no peer evaluation form submissions after the deadline.
* Students reported by their peers not helping the team to progress on this milestone **will lose**

50% of the total grade.

Tasks:

<<App Description>>

This app should let the users add multiple cities to a list and show them to the users. Upon clicking on each city in the list, the app prints a welcome message containing the name of the selected city, along with options for the user to see more details about the "weather" and "map" of the city of interest. You are expected to implement the following features for the app (in Milestone 3 and Milestone 4):

1. **User login and authentication**

   Each user should be able to create and log into an account to see the added cities of interest

2. **Ability to add new items to the list of locations (or remove them)**

   Each user should be able to add and remove the cities from the list on their home screen

3. **Enhancing the UI and customizing it for different users**

   When creating an account, each user can select a different design theme for the app. The app should customize the UI for each user based on their preferences

4. **Showing weather information for each location**

   Upon user request, the app connects to a weather server, fetches the weather information of the city, and shows it to the user

5. **Ability to use LLM to suggest weather insights**

   When showing the weather information of cities, the app can call LLMs (e.g., Gemini LLM) to generate context-relevant questions and responses based on the current weather data and present them to the user.

6. **Showing the map of each location**

   Upon user request, the app loads the map of the city and shows it to the user

<<Requirement Engineering and Design>>

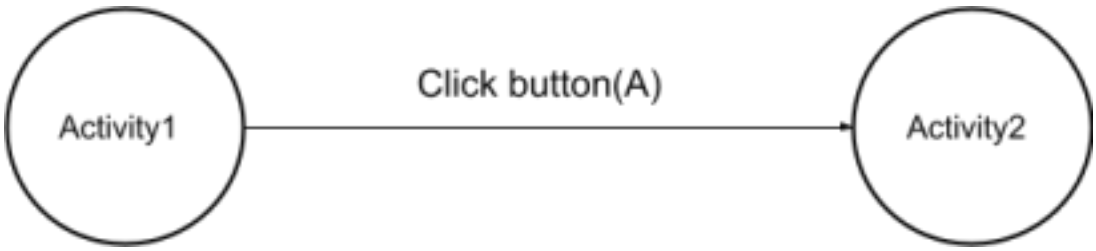
Based on the above app description, prepare the following documents:

- **Informal requirements:** You should identify at least **six requirements** for the application. While only six of them will be graded in this milestone, this will help you with the design

and later implementation of the app. Each requirement should match the following format: **“R<requirement number>:<sentences describing the requirement>”**

- **Fully dressed use cases:** You should identify at least **six use-case scenarios** for the application. Use the simplified template introduced in the course (Week 3, Lecture 3.7) and formalize the requirements. Your fully dressed use cases should be in the form of a table, i.e., one table per each information use-case scenario. Similar to the requirements, while only six of them will be graded in this milestone, more use cases will help you with the design and later implementation of the app.
- **Class diagram:** Brainstorm the classes you require to implement to meet requirements and enable use cases, and draw the UML class diagram. You are expected to follow the best practices you learned in the course.
  Distinguish different Android components using the keywords Activity, Service, Receiver, or ContentProvider in the **“class name”** (for non-Android classes, simply use a name for the class). For example, your class names should be MainActivity, ShowMapActivity, WeatherService, etc. You should understand the fundamentals of Android app development. The following articles in the Android Developers Guide can be helpful:
  https://developer.android.com/guide/components/fundamentals
  https://developer.android.com/guide/components/activities
  https://developer.android.com/guide/components/services
  https://developer.android.com/guide/components/broadcasts
  https://developer.android.com/guide/topics/providers/content-providers

- **Component Transition Graph:** This structure represents how the Android components interact with each other throughout the application's lifetime. Each node on this graph represents an Android component, and the edges show transitions between them. Edges should be labeled by the means that they enable the transition. For example, if the user clicks on button A, and this click causes the transition between Activity1 and Activity2, the transitions between these two Activities in the Component Transition Graph can be represented as below:



  You can use UML State Machine Diagram to draw the component transition graph. This graph would be very helpful once you start the implementation of features.

<<Peer Evaluation>>

Each team member should submit the following form for the peer evaluation **for Milestone 2** (we accept no peer review for this milestone after the milestone deadline):
https://forms.gle/8zSVwQQ1dLt6452AA