Product Design

TEAM < STACK 5 >

Edidiong Anwan, Priscilla Osineye, Tedros Tecle, Vicky Wainaina, Aimua Ilogienboh

Note: this is a "living document", meaning its content will change with the implementation of the project. Use it to capture key project concepts and document your product design and decisions. Make sure the design is traceable to the requirements. Remove/replace the blue text and the descriptive paragraphs in each section prior to your submission as directed by your instructor. REMOVE THE ITALICIZED TEXT BEFORE SUBMISSION.

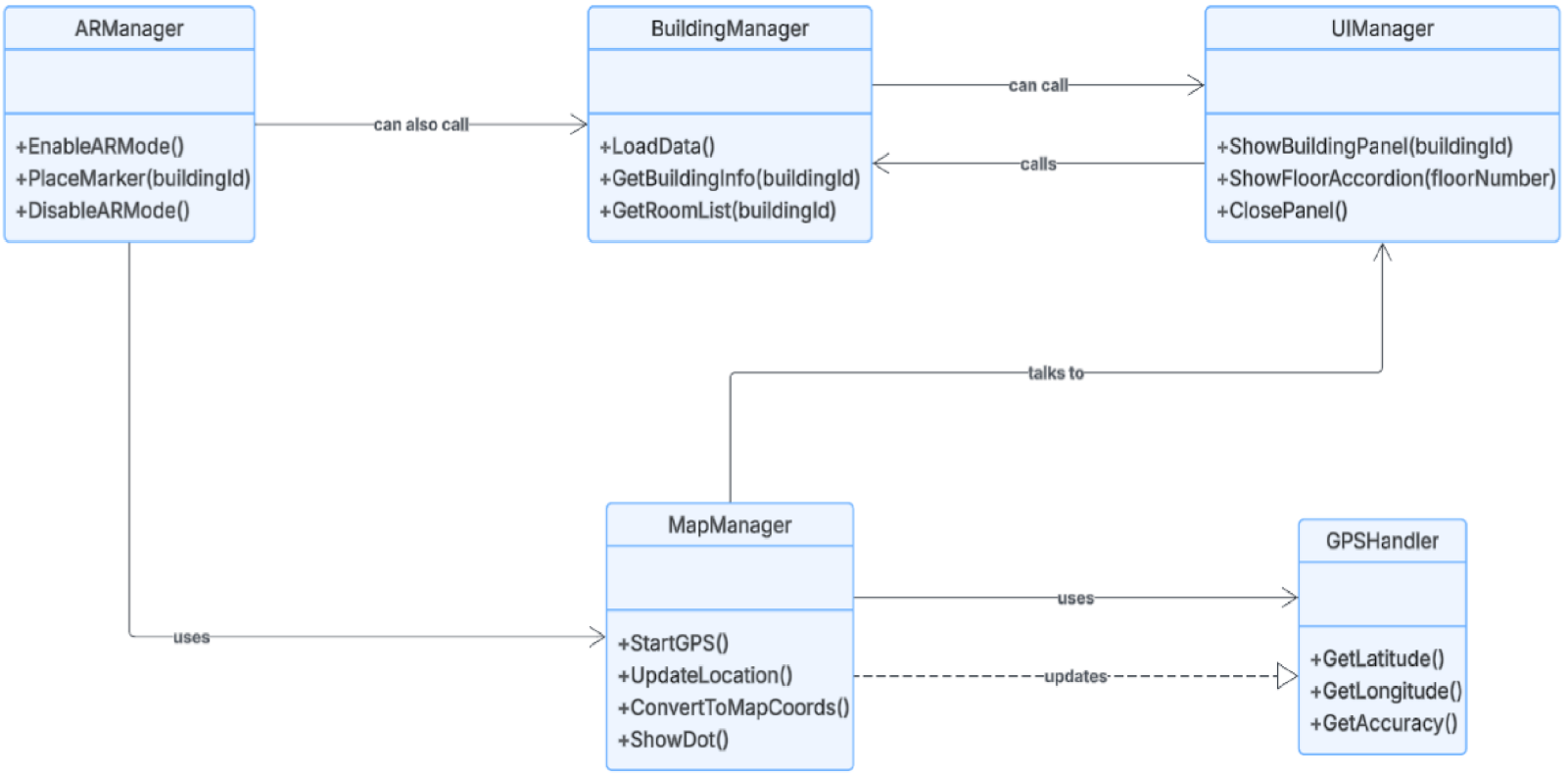
THIS DOCUMENT IS A STARTING POINT, YOUR TEAM IS EXPECTED TO ADD/MODIFY ALL NECESSARY SECTIONS.

You may use any drawing tool for your UML diagrams. If your diagrams are too big to cut and paste into this document, provide a reference to the external image files(s) [JPG or PNG] where they can be found or segment your image into legible sections to make them fit.

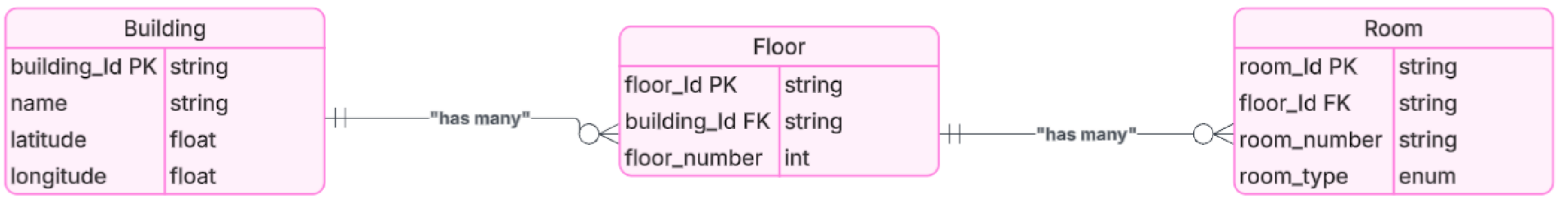
Revision Number	Revision Date	Summary of Changes	Author(s)
0.1	10/3/2025	Updates to Er diagram, information arch, design summary, and class diagrams	Everyone
0.2	11/6/2025	Added Mapbox integration for map rendering and navigation visualization in Unity. Updated project dependencies and configured API keys.	Everyone
0.3	12/3/2025	Updated the floor plans by digitizing them. To help start the navigation of the map	Tedros, Priscilla, Vicky

The Revision Table above must be augmented after any version of this document is updated. Insert any necessary rows at the bottom of the table.

Class Diagram(s)



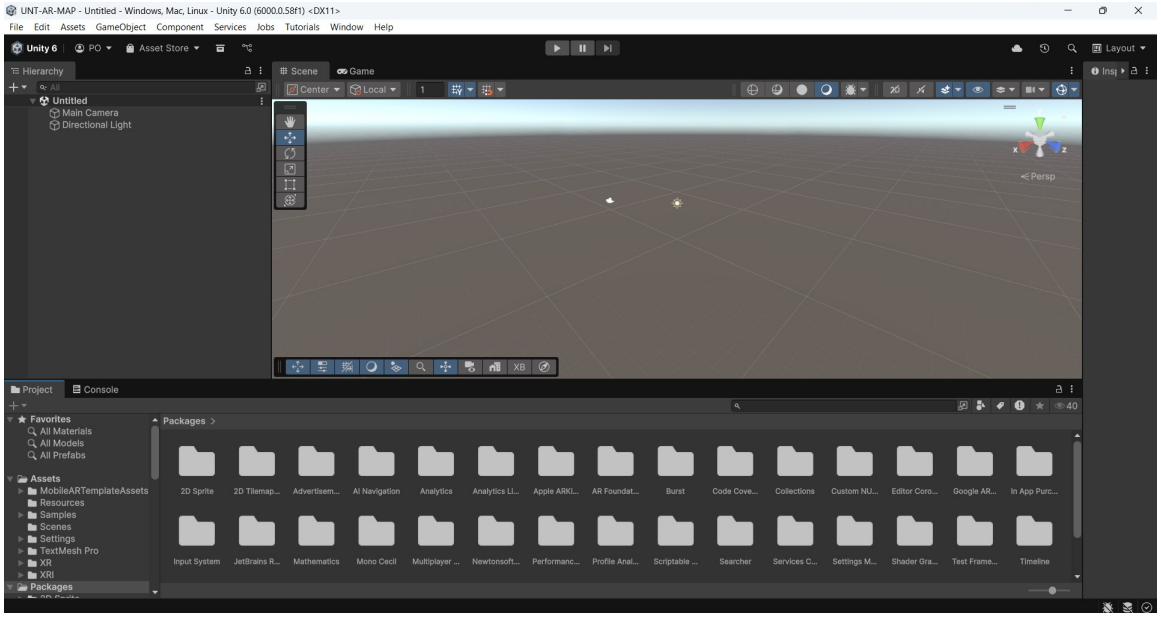
ER Diagrams(s)



Information Architecture Diagram

Start Screen → Map View → Building Info → Floor → Room Info → AR Mode

User Interface Wireframe(s)/Screenshot(s)



Design Summary

Our design divides the work into different managers, such as GPSHandler, MapManager, ARManager, UIManager, and BuildingManager. This makes it modular: GPSHandler manages location, ARManager controls the AR display, and BuildingManager handles data retrieval. The ER diagram helps organize the data into Buildings, Floors, and Rooms with clear connections.

Design Rationale

Our team adopted a modular design architecture to make the Unity-based AR navigation system more structured, maintainable, and scalable. Core components such as **GPSHandler**, **MapManager**, **ARManager**, **UIManager**, and **BuildingManager** were created to handle separate functions. This organization allows different features—like real-time location tracking, AR rendering, and building data management—to be

developed and tested independently. It also supports future updates without affecting other parts of the system.

Early in development, we considered two main approaches: a single, unified controller that handled all logic and a modular, manager-based system. The unified design was initially appealing for its simplicity but quickly proved less flexible for collaboration and debugging. We chose the modular structure because it divides responsibilities clearly, reduces code dependencies, and makes troubleshooting easier—important advantages when working as a team in Unity.

Another key design choice involved how to manage and store data. We evaluated using **local storage** versus a **cloud-based Firebase database**. Since our app relies on real-time updates and dynamic map information, we selected Firebase to handle building, floor, and room data. This internet-dependent solution ensures that updates can be made instantly and shared across devices. Although it requires a stable network connection, it provides scalability and makes future feature integration, such as live event data or maintenance updates, much easier.

Finally, we decided to focus our navigation system exclusively on **Discovery Park** instead of the entire UNT campus. This narrowed scope allows the team to refine accuracy, optimize AR performance, and create a stronger proof of concept before expanding. The interface flow—**Start Screen → Map View → Building Info → Floor → Room Info → AR Mode**—was designed for clarity and consistency. Moving forward, this section will continue to document design changes, challenges, and reasoning to maintain a clear record of our project's evolution.

Mapbox Integration:

To enhance location visualization, we integrated the Mapbox SDK into our Unity environment. Mapbox provides detailed, real-time maps that synchronize with GPSHandler for accurate user tracking. It allows for dynamic map rendering, supports zoom levels, and overlays Discovery Park building data for smoother transitions between Map View and AR Mode. This integration makes the navigation experience more realistic.