GPS Based Campus Room Finder

Sprint 1
2025-09-02

Name	**Email Address**
Aaron Downing	aaron.downing652@topper.wku.edu
Ryerson Brower	ryerson.brower178@topper.wku.edu
Kaden Hunt	kaden.hunt144@topper.wku.edu

Michael Galloway
CS 361
Fall 2025
Project Technical Documentation

Contents

List of Figures

1 Introduction

1.1 Project Overview

The GPS Based Campus Room Finder is a mobile application designed to simplify navigation for WKU students and faculty. The primary purpose of this project is to create a consistent and easy-to-use tool that addresses the common problem of navigating a large and unfamiliar campus environment. Using GPS technology, the application will help users quickly determine their current location and find the most efficient route to any building and room number on campus. This tool will eliminate the need for paper maps and provide an important resource for new and current members of WKU.

The final product will be a user-friendly mobile application that gives real-time guidance and an estimation of travel times. This software will be a valuable tool for the university with potential for expansion to include additional features that continue to enhance the campus experience.

1.2 Project Scope

The project scope defines the boundaries, commitments, and outputs required to deliver the GPS-Based Campus Room Finder. This scope covers all activities necessary to design, implement, test, and document a mobile application that meets the client's expectations while remaining usable and maintainable beyond the project timeline.

Deliverables & Outcomes:

- **Written Reports:** Detailed organizational and technical documents submitted at the conclusion of each of the four sprints.
- **Presentations:** A presentation delivered at the end of each sprint to summarize progress and demonstrate results.
- **Evaluations:** Peer evaluation forms submitted individually by team members after each sprint.
- **Final Product:** A fully tested, documented, and maintainable Android mobile application that provides GPS-based navigation to campus buildings and rooms.

Work Required:

- **Tasks:** All development tasks including source code creation, user interface design, system integration, testing, and documentation. Additional requirements will be integrated as identified throughout the project.
- **Team:**
  - Kaden Hunt — Project Manager, Task Manager
  - Aaron Downing — Documentation Draft
  - Ryerson Brower — Research Coordinator
- **Time Commitment:** Work will be divided across four sprints. Each team member will contribute 8–10 hours per week to development, meetings, and documentation.
- **Resources:** GitHub will serve as the version control system and task management platform. The documentation will be written collaboratively in Texmaker. The development will take place on personal laptops running Windows 10 or later which will meet the requirements of Android Studio. of Android Studio.
- **Schedule:** Deliverables align with the four milestone deadlines outlined on Blackboard. Weekly client meetings occur on Tuesdays at 12:35 p.m. in Snell Hall B104. Internal team meetings will take place on Thursdays at 2:00 p.m.

Altogether, this scope establishes what will be delivered, the benefits it provides, and the foundation for successful implementation

1.3 Technical Requirements

1.3.1 Functional Requirements

Mandatory Functional Requirements
The application will use GPS coordinates to determine the user's current location within the campus boundaries.
The application will allow the user to search for a specific building and room number using a text-based input.
The application will generate a step-by-step navigation route from the user's current location to the selected room.
The application will have an interactive display to navigate the user to the building and room.
The application will provide an estimated travel time based on the mobile location of the user.
Extended Functional Requirements
The application will provide voice-guided navigation for hands-free use.
The application will allow users to bookmark or "favorite" frequently visited rooms for quicker searches.
The application will provide a "recent searches" history so users can quickly reselect prior destinations

The functional requirements for the WKU GPS-based campus room finder are designed to help WKU students and faculty easily locate rooms across campus. By using GPS coordinates to determine the user's current location, the application provides accurate, real-time directions, allowing users to navigate campus quickly and effectively. The interactive display offers clear step-by-step guidance to the desired building and room number, featuring a user-friendly interface that makes input, ensuring easy accessibility for all users. To get rid of any other unnecessary confusion, the application will provide an estimated travel time based on the mobile location of the user. This feature also allows users to make better decisions about which route to take depending on their time constraints between classes. The applications goal is to address common problems such as getting lost or arriving late to class, enhancing convenience and creating a smoother, more reliable navigation experience across the WKU campus.

1.3.2 Non-Functional Requirements

Mandatory Non-Functional Requirements
The application will provide location updates with an accuracy of at least ± 5 meters under clear sky conditions.
The application will deliver route generation results within 2 seconds of the user's search request.
The application will be compatible with either Android or iOS mobile operating systems.
The application will provide visual and text-based route guidance.
The application will support operation in both portrait and landscape orientations without loss of functionality.
All project source code must be developed by the CS 360 project team.
The project must use a database.
Performance metrics should be gathered and optimized.
Security metrics should be gathered and optimized
User interface metrics should be gathered and optimized.
Extended Non-Functional Requirements
The application should maintain functionality with limited or no internet connection
The application should consume minimal battery power while running in the background.
The application should be designed with a clean, intuitive user interface that prioritizes ease of use.

The mandatory non functional requirements for the WKU GPS-based campus room finder ensure the application performs reliably, efficiently, and securely while providing users with a positive application experience. By requiring location updates with an accuracy of within 5 meters under clear sky conditions, the app guarantees precise position for navigating campus. Delivering route generation results within 2 seconds ensures that users receive routes efficiently without unnecessary delays. Visual and text-based route guidance enhances accessibility and makes navigation possible for all users. Requiring all project source code must be developed by the CS 360 project team helps achieve the desired learning outcomes of the class and encourages accountability throughout the team in a real-world setting. Using a database enables efficient storage and retrieval of all building and rooms on campus. Additionally, gathered and optimized security metrics will ensure the application remains fast, safe,

and easy to use, while also meeting quality standards set out by the client. Collectively, these requirements provide a reliable and high-quality tool for campus navigation.

1.4 Target Hardware Details

We will create a mobile app for students and faculty around campus. The target hardware for our mobile app will primarily be for smart phones on Android. The minimum requirements are: The minimum CPU required is a quad-core ARM-based processor (or the processor that's in most smart phones) to ensure real time GPS processing and navigation. Our test case for the CPU would be to run a continuous navigation to confirm smooth updating with no lag. You would need at least 2 GB of RAM so the product can also run things like GPS tracking at the same time and the rendering of the map. Our test case for the RAM would be to monitor memory usage under a heavy load. A minimum of 200 MB of persistent storage is needed for the application installation, location files, and maps. Our storage test case would be to install the app and see how much it takes up. Network connectivity will be necessary via Wi-Fi or 4G/5G mobile data, with at least 1 Mbps of sustained bandwidth for map updates and routing queries. The targeted output device will be a touchscreen of a smart phone.

We don't have a plan to place this app on PC or computers but it would be something to possible implement in the future. A software we are using called Android studios also has some hardware requirement. These are: A OS of 64-bit Windows 10 or newer, RAM with 16 GB, a CPU with a processor with visualization support (Intel VT-x or AMD-V), micro architecture from after 2017. 16 GB of free disk space, preferably on a Solid State Drive (SSD). A GPU with at least 4 GB of VRAM.

1.5 Software Product Development

The software we are using already are Google doc, TexLive, github, Android Studio, SQL and VScode. Google doc we use to keep up with each others documents and any document we need to print out. For our documentations we are using TexLive to edit both or Organizational and Tech Docs. Github we used to get a repository that is easy to access and easy for us to place our updated docs and code. Git hub also helps use be able to access everything without having to send files back and forth. We already planed on using VScode and the coding language we will use to code our app is JAVA. VS code with the help of github makes it really easy to pull everyone's code when they edit it so once again we are not sending a ton of files and getting them mixed up. One of the more important software we will use is Android Studio. This will allow use to code and Android app easier. This will also let use visualize the app when we don't have a Android phone accessible. For our database to hold all the data for location of rooms and routes we will use SQL.

2 Modeling and Design

2.1 System Boundaries

2.1.1 Physical

The physical system boundaries for the GPS-Based Campus Room finder are limited to mobile devices, primarily Android smart phones used by WKU students and faculty. The application relies on the mobile phones built-in hardware components such as the GPS system, touchscreen interface, and mobile network for accurate navigation. It will not include any external hardware devices not included in the user's smart phone. The system interacts with Google Maps API (or similar) for real-time navigation and requires the campus buildings and room data stored in the project database. Any devices outside of android mobile devices, such as iphones, PCs, and kiosks, lie out of the scope of the project. The application requires minimum resources, 2GB of RAM and 100mb of storage will be enough which is available on most android phones. Security is ensured by relying on the built-in authentication systems on the phone. The application is easily scalable by adding functionality for more android phones and adding a larger database.

2.1.2 Logical

The logical system boundaries for the GPS-Based Campus Room Finder defines the flow of the information and functions managed by the application. Internally, the system handles location detection, room and building

searches, and route generation. It manages the retrieval of the campus building and room number data from the project database and uses the GPS coordinates for navigation. Externally, the system communicates with Google Maps API and user-interface to display directions and built-in mobile operating systems for device-level functions such as notifications. Any process beyond navigation, such as class scheduling and campus event times remain outside the logical scope of the project.

2.2 Wireframes and Storyboard

Text goes here.

2.3 UML

2.3.1 Class Diagrams

Text goes here.

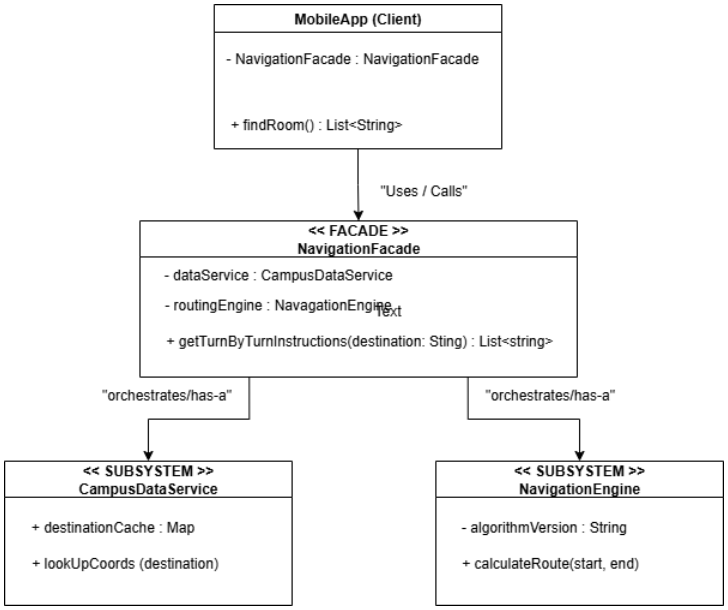

Figure 1: Structural Design Pattern Facade.

2.3.2 Use Case Diagrams

Text goes here.

2.3.3 Use Case Scenarios Developed from Use Case Diagrams (Primary, Secondary)

Text goes here.

2.3.4 Sequence Diagrams

The following sequence diagram shows the process of the actor (user) getting into the UI and requesting a location. This would then go through the database to get location data. It would then create a route from the users location and the desired room. As the user is moving the route would update in relation to the location of the user.

2.3.5 State Diagrams

The following state diagram models the process of a user searching for a campus room. The diagram begins when the user enters a room number and submits it. The system then validates the input: if it is invalid, the

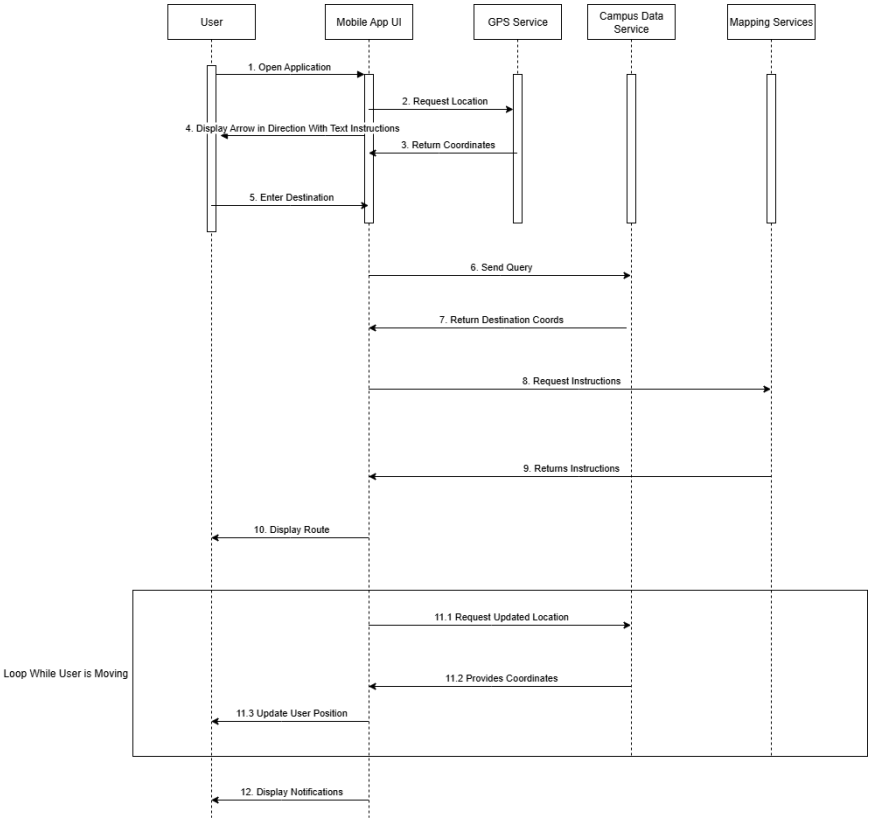

Figure 2: Sequence diagram for the user route request, generation, and update.

user is prompted to retry; if valid, the system queries the database. If the room is found, a path is generated and directions are displayed. If the room is not found, or a query fails, the user can correct their input and resubmit. This diagram focuses only on the search and route-display feature in Sprint 2, since implementation has not yet begun.

2.3.6 Component Diagrams

Text goes here.

2.3.7 Deployment Diagrams

The diagram shows the physical side of the system, which is the user device like a smart phone or tablet.Also related to the physical side there is the client UI. Then there is also the virtual that has database, and routing engine that makes the route. The virtual also holds all the logic and back end APIs.

2.4 Version Control

Text goes here.

2.5 Requirements Traceability Table

Text and table goes here.

2.6 Data Dictionary

Text and table goes here.

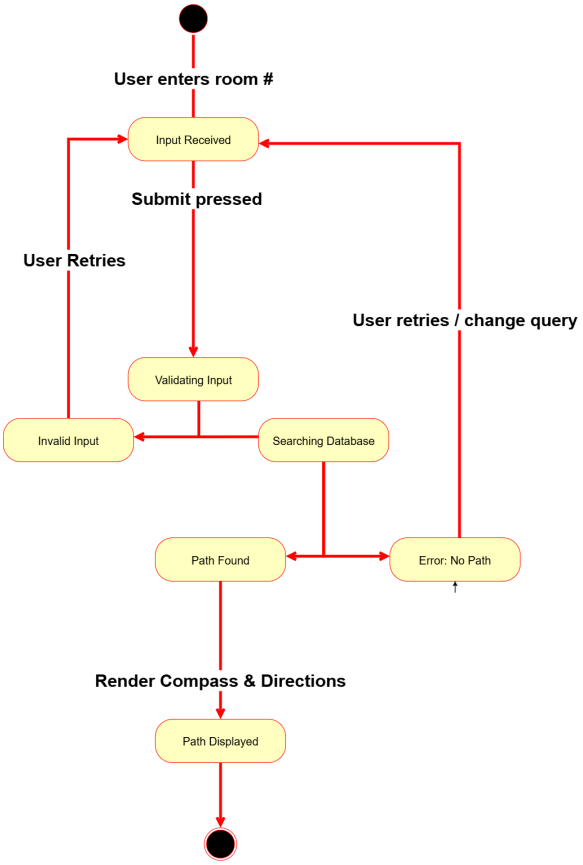

Figure 3: State diagram for the room search feature, showing user input, validation, query, error handling, and path display.

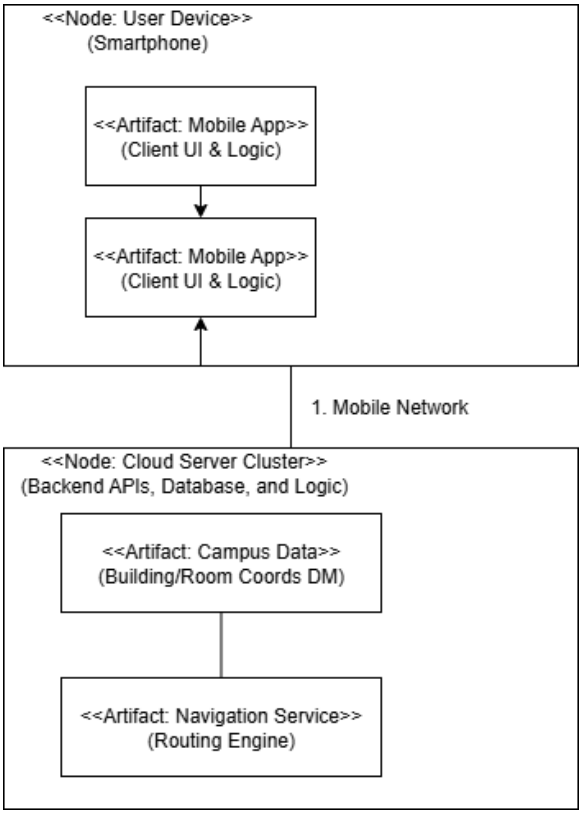

Figure 4: Deployment diagram for the mobile Room Finder app.

2.7 User Experience

2.7.1 Gameplay Diagram

Text goes here.

2.7.2 Gameplay Objectives

Text goes here.

2.7.3 User Skillset

Text goes here.

2.7.4 Gameplay mechanics

Text goes here.

2.7.5 Gameplay Items

Text goes here.

2.7.6 Gameplay Challenges

Text goes here.

2.7.7 Gameplay Menu Screens

Text goes here.

2.7.8 Gameplay Heads-Up Display

Text goes here.

2.7.9 Gameplay Art Style

Text goes here.

2.7.10 Gameplay Audio

Text goes here.

3 Non-Functional Product Details

3.1 Product Security

3.1.1 Approach to Security in all Process Steps

Text goes here.

3.1.2 Security Threat Model

Text goes here.

3.1.3 Security Levels

Text goes here

3.2 Product Performance

3.2.1 Product Performance Requirements

Text goes here.

3.2.2 Measurable Performance Objectives

Text goes here.

3.2.3 Application Workload

Text goes here.

3.2.4 Hardware and Software Bottlenecks

Text goes here.

3.2.5 Synthetic Performance Benchmarks

Text goes here.

3.2.6 Performance Tests

Text goes here

4 Software Testing

4.1 Software Testing Plan Template

Test Plan Identifier:
Introduction:
Test item:
Features to test/not to test:
Approach:
Test deliverables:
Item pass/fail criteria:
Environmental needs:
Responsibilities:
Staffing and training needs:
Schedule:
Risks and Mitigation:
Approvals:

4.2 Unit Testing

Text goes here.

4.2.1 Source Code Coverage Tests

Text goes here.

4.2.2 Unit Tests and Results

Text goes here.

4.3 Integration Testing

Text goes here.

4.3.1 Integration Tests and Results

Text goes here.

4.4 System Testing

Text goes here.

4.4.1 System Tests and Results

Text goes here.

4.5 Acceptance Testing

Text goes here.

4.5.1 Acceptance Tests and Results

Text goes here.

5 Conclusion

Text goes here.

6 Appendix

6.1 Software Product Build Instructions

Text goes here.

6.2 Software Product User Guide

Text goes here.

6.3 Source Code with Comments

Text goes here.