

# 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	email kaden.hunt144@topper.wku.edu

CS 361  
Fall 2025  
Project Technical Documentation

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Project Overview . . . . .	1
1.2	Project Scope . . . . .	1
1.3	Technical Requirements . . . . .	2
1.3.1	Functional Requirements . . . . .	2
1.3.2	Non-Functional Requirements . . . . .	2
1.4	Target Hardware Details . . . . .	3
1.5	Software Product Development . . . . .	3
<b>2</b>	<b>Modeling and Design</b>	<b>3</b>
2.1	System Boundaries . . . . .	3
2.1.1	Physical . . . . .	3
2.1.2	Logical . . . . .	3
2.2	Wireframes and Storyboard . . . . .	4
2.3	UML . . . . .	4
2.3.1	Class Diagrams . . . . .	4
2.3.2	Use Case Diagrams . . . . .	4
2.3.3	Use Case Scenarios Developed from Use Case Diagrams (Primary, Secondary) . . . . .	4
2.3.4	Sequence Diagrams . . . . .	4
2.3.5	State Diagrams . . . . .	4
2.3.6	Component Diagrams . . . . .	4
2.3.7	Deployment Diagrams . . . . .	4
2.4	Version Control . . . . .	4
2.5	Requirements Traceability Table . . . . .	4
2.6	Data Dictionary . . . . .	4
2.7	User Experience . . . . .	4
2.7.1	Gameplay Diagram . . . . .	4
2.7.2	Gameplay Objectives . . . . .	4
2.7.3	User Skillset . . . . .	4
2.7.4	Gameplay mechanics . . . . .	5
2.7.5	Gameplay Items . . . . .	5
2.7.6	Gameplay Challenges . . . . .	5
2.7.7	Gameplay Menu Screens . . . . .	5
2.7.8	Gameplay Heads-Up Display . . . . .	5
2.7.9	Gameplay Art Style . . . . .	5
2.7.10	Gameplay Audio . . . . .	5
<b>3</b>	<b>Non-Functional Product Details</b>	<b>5</b>
3.1	Product Security . . . . .	5
3.1.1	Approach to Security in all Process Steps . . . . .	5
3.1.2	Security Threat Model . . . . .	5
3.1.3	Security Levels . . . . .	5
3.2	Product Performance . . . . .	5
3.2.1	Product Performance Requirements . . . . .	5
3.2.2	Measurable Performance Objectives . . . . .	5
3.2.3	Application Workload . . . . .	5
3.2.4	Hardware and Software Bottlenecks . . . . .	6
3.2.5	Synthetic Performance Benchmarks . . . . .	6
3.2.6	Performance Tests . . . . .	6

4

Software Testing

6

4.1

Software Testing Plan Template . . . . .

6

4.2

Unit Testing . . . . .

6

4.2.1

Source Code Coverage Tests . . . . .

6

4.2.2

Unit Tests and Results . . . . .

6

4.3

Integration Testing . . . . .

6

4.3.1

Integration Tests and Results . . . . .

6

4.4

System Testing . . . . .

6

4.4.1

System Tests and Results . . . . .

6

4.5

Acceptance Testing . . . . .

7

4.5.1

Acceptance Tests and Results . . . . .

7

5

Conclusion

7

6

Appendix

7

6.1

Software Product Build Instructions . . . . .

7

6.2

Software Product User Guide . . . . .

7

6.3

Source Code with Comments . . . . .

7

## 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

<b>Mandatory Functional Requirements</b>
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.
<b>Extended Functional Requirements</b>
Ext. Req 1
Ext. Req 2
Ext. Req 3

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

<b>Mandatory Non-Functional Requirements</b>
The application will provide location updates with an accuracy of at least $\pm 5$ 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.
<b>Extended Non-Functional Requirements</b>
Ext. Req 1
Ext. Req 2
Ext. Req 3

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 possibly 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 virtualization 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 our 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

Text goes here.

### **2.1.2 Logical**

Text goes here.

## **2.2 Wireframes and Storyboard**

Text goes here.

## **2.3 UML**

### **2.3.1 Class Diagrams**

Text goes here.

### **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**

Text goes here.

### **2.3.5 State Diagrams**

Text goes here.

### **2.3.6 Component Diagrams**

Text goes here.

### **2.3.7 Deployment Diagrams**

Text goes here.

## **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.

## **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.