

GPS Based Campus Room Finder

Sprint 3
10/27/2025

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

Client: Michael Galloway
CS 360
Fall 2025
Project Organization Documentation

Contents

1	Project Team's Organizational Approach	1
2	Schedule Organization	1
2.1	Gantt Chart v1:	1
2.2	Gantt Chart v2:	1
2.3	Gantt Chart v3:	1
2.4	Final Gantt Chart:	2
3	Progress Visibility	2
3.1	Sprint 1 Progress Visibility	2
3.2	Sprint 2 Progress Visibility	2
3.3	Sprint 3 Progress Visibility	2
3.4	Sprint 4 Progress Visibility	3
4	Software Process Model	3
5	Risk Management	3
5.1	Risk Identification	3
5.2	Risk Planning	3
5.3	Risk Monitoring	4

List of Figures

1 Project Team's Organizational Approach

Sprint 1: Kaden Hunt served as the Project Manager for this sprint. The entire team met in person once per week for hour-long working sessions at the Commons on Thursdays at 2:00 PM. To maintain continuous collaboration, the group utilized a Discord server for ongoing chat, video calls, and file sharing, along with text messaging for quick updates. Furthermore, the team held brief, 20-minute meetings with the client every Tuesday to present a prepared progress report, ensuring alignment and feedback on our development process.

Sprint 2: For this sprint, the team maintained the same overall meeting structure, continuing to gather weekly in person at the Commons on Thursdays at 2:00 PM. These meetings were used to coordinate progress, divide work, and review draft materials together. The client meetings on Tuesdays also continued, providing regular opportunities for feedback. The primary focus of Sprint 2 shifted toward software modeling and design. Team members were responsible for producing UML diagrams and design pattern documentation, which were reviewed collectively during meetings to ensure consistency. This sprint emphasized collaboration around visual modeling, so the group spent more time walking through diagrams together, making sure the representations were aligned with the project. Project Manager Ryerson Brower ensured the task list was up to date and that contributions were balanced across members, while texting was used to coordinate outside of scheduled times.

Sprint 3: For this sprint, the team maintained the same overall meeting structure, continuing to gather weekly in person at the Commons on Thursdays at 2:00 PM. These meetings were used to coordinate progress, divide work, and review draft materials together. The client meetings on Tuesdays also continued, providing regular opportunities for feedback. The primary focus of Sprint 3 progress visibility centered on the development of wireframes and storyboards, along with addressing different security risks, and with getting close to a final product by making our app UI. Team members were responsible for producing wireframes, s, and researching security risks. Which were reviewed collectively during meetings to ensure consistency. Project Manager Ryerson Brower ensured the task list was up to date and that contributions were balanced across members, while texting was used to coordinate outside of scheduled times.

Sprint 4:

2 Schedule Organization

2.1 Gantt Chart v1:

The focus for Sprint 1 was focused on project initiation, team formation, and foundational planning. The primary tasks involved establishing team roles and responsibilities, setting up collaboration tools like GitHub and Google Docs, and drafting the initial versions of both the Organizational and Technical documentation. This chart outlined the critical path for researching project requirements, defining the scope, and performing an initial risk analysis to ensure the project's viability. The chart is located in the project directory at: Topper-Nav/docs/TopperNav_Gantt_Chart_Sprint2.png

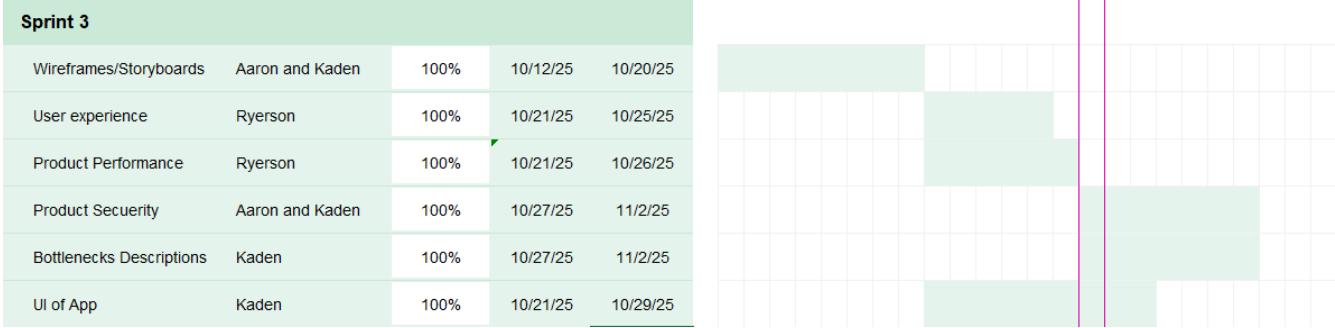
2.2 Gantt Chart v2:

The focus for Sprint 2 was on software modeling and design. Tasks included creating UML diagrams such as use case, sequence, state, component, and deployment diagrams, as well as developing class diagrams tied to design patterns. Additional deliverables included defining system boundaries, producing an updated risk analysis, and a requirements traceability table. The team also worked on both technical and organizational documentation and preparing the Sprint 2 presentation. This sprint represented the transition from planning into detailed design, with an emphasis on visual artifacts that demonstrate how the system will operate. The Gantt chart for Sprint 2, outlining these milestones and due dates, is located in the project directory at: Topper-Nav/docs/TopperNav_Gantt_Chart_Sprint2.png

2.3 Gantt Chart v3:

The focus for Sprint 3 was on software implementation and design. Tasks included, wireframes/storyboards, user experience, product security, product performance. Additional deliverable included defining system boundaries defining different security requirements and techniques, here we also needed to start getting test results.

But our group took a different approach, we wanted to get our UI for the app done first. So at the moment we don't have much for us to test but we got a visual product and a UI. The team also worked on both technical and organizational documentation and preparing the Sprint 3 presentation. This sprint represented the transition from planning into actual implementation of the and getting close to the final product.



2.4 Final Gantt Chart:

3 Progress Visibility

3.1 Sprint 1 Progress Visibility

The team progressed steadily through Sprint 1 and completed the deliverables before the due date. Tasks were divided among members during class and reinforced during weekly meetings. Kaden Hunt managed the task list, prepared the presentation, and contributed to both the Organizational and Technical documentation. Aaron created the Gantt chart and also worked on sections of both documents, while Ryerson focused on contributing to the documentation. Progress visibility within the group was maintained through frequent conversations, updates in the group chat and Discord, and use of the GitHub Projects page. The Gantt chart also provided a clear overview of tasks and deadlines, ensuring accountability. Through these resources, members could report when tasks were complete, request assistance, or flag dependencies. Overall, consistent communication and shared tools allowed the team to remain coordinated and deliver Sprint 1 successfully.

3.2 Sprint 2 Progress Visibility

During Sprint 2, progress visibility centered on the development of diagrams and design materials. Tasks were assigned during meetings and tracked via texts or the next in person meeting. Team members communicated their progress through texts as well, posting drafts of diagrams for feedback before finalizing them. The use of visual diagrams required collaborative review, so much of the work was shared early and discussed during weekly meetings. Updates to documentation and the presentation slides were also reviewed as a group to ensure consistency across deliverables. Progress for the client was summarized in the weekly report and presented at Tuesday meetings, which gave the team an opportunity to confirm the project was on track before moving forward. This cycle of assignment, discussion, and review allowed the group to stay on schedule and adapt when tasks needed additional attention.

3.3 Sprint 3 Progress Visibility

During Sprint 3, progress visibility centered on the development of wire-frames and storyboards, along with addressing different security risks, and with getting close to a final product by making our app UI. Tasks were assigned during meetings and tracked via texts or the next in person meeting. Team members communicated their progress through texts as well, posting drafts of diagrams for feedback before finalizing them. The use of visual diagrams required collaborative review, so much of the work was shared early and discussed during weekly meetings. Updates to documentation and the presentation slides were also reviewed as a group to ensure consistency across deliverables. Progress for the client was summarized in the weekly report and presented at Tuesday meetings, which gave the team an opportunity to confirm the project was on track before moving forward. This cycle of assignment, discussion, and review allowed the group to stay on schedule and adapt when tasks needed additional attention.

3.4 Sprint 4 Progress Visibility

Text goes here.

4 Software Process Model

Our team adopted the Iterative Software Process Model, which emphasizes building the project in small, repeatable cycles. This approach allowed us to steadily refine our work by dividing development into manageable stages and reviewing progress continuously. Unlike a linear model, the iterative approach gave us the flexibility to adjust plans based on what we learned along the way, which helps reduce risk and increases the quality of the final deliverables.

During Sprint 1, our focus was on preparation and foundational work. Quality control steps included producing clear documentation, researching relevant tools and requirements, and delivering a presentation to communicate progress. We also treated environment setup as part of quality assurance by ensuring all team members had the same software installed, including Android Studio, the JDK, TeX Live, and TeXmaker. These checks provided consistency across the team and created a strong base for future development cycles.

During Sprint 2, our team continued with the Iterative Software Process Model, focusing on refinement and incremental progress. Each cycle allowed us to revisit earlier work while introducing new features, ensuring flexibility and adaptability. This helped us incorporate feedback from Sprint 1 and quickly adjust designs or diagrams as our understanding of the project matured.

In this sprint, we placed stronger emphasis on design artifacts and system modeling, including UML diagrams (use case, sequence, state, component, and deployment). These served as both technical blueprints and quality checks, ensuring requirements were being met. Quality assurance also included maintaining consistency across diagrams through member check ins. By combining documentation updates with incremental design work, Sprint 2 advanced the project from planning into concrete models, providing a clearer roadmap for coding in later sprints.

5 Risk Management

5.1 Risk Identification

The following risks are listed and categorized by potential risk level.

- **Database Performance (High Priority):** The database may become too slow under heavy loads, affecting responsiveness and user experience.
- **Route Generation API Failure (High Priority):** The external route generation API may not function as expected, preventing key functionality of the application.
- **Compatibility Issues (Medium Priority):** The system may encounter compatibility problems across different devices, operating systems, or software versions.
- **Access to Android Phones (Medium Priority):** Limited availability of Android devices for testing may hinder progress.
- **Large File Download (Low Priority):** Users may face issues when downloading or uploading large files, leading to performance bottlenecks.

5.2 Risk Planning

The team has outlined the following plans for risk planning:

- **Database Performance:** Implement query optimization, proper indexing, and caching strategies. Load testing will be performed early to identify bottlenecks.
- **Route Generation API Failure:** Identify and test backup APIs during early development.

- **Compatibility Issues:** Use emulators and virtual machines to ensure broader compatibility across various platforms.
- **Access to Android Phones:** Gain access to an android phone temporarily and share them among team members as needed.
- **Large File Download:** Implement file compression. Clearly communicate file size limitations to users.

5.3 Risk Monitoring

The following monitoring strategies will be used for the remaining duration of the project:

- **Database Performance:** Monitor query execution times and system performance.
- **Route Generation API Failure:** Implement automated alerts for API failures or unusual latency.
- **Compatibility Issues:** Maintain a compatibility checklist and update it with each system change.
- **Access to Android Phones:** Track device usage and availability within the team. .
- **Large File Download:** Monitor client feedback related to file transfers.