Opsis - Seal T6

Empowering Indoor Navigation
for the Visually Impaired

Meet the team

- Nathaniel Norman (Coordinator)
- Tee Nguyen (Backend Development)
- Rayeed Zarif (Accessibility & Research)
- Samantha Sarabia (Frontend Development)
- Nowsin Anzum Mozumder (Backend Development)
- Amal Saeed (Accessibility & Research)
- Erika Barron (Documentation)
- Corina Salazar (UI/UX Designer)
- Aryana Khaffaf Sharif Zamin (Team Leader/Coordinator)

Revision History

Changes	Date	Author
Initial draft	09/23/2025	Erika Barron
Documentation edits	10/1/2025	Aryana Khaffaf Sharif Zamin
Revised PPT	10/5/2025	Aryana Khaffaf Sharif Zamin
Revisions	10/6/2025	Aryana Khaffaf Sharif Zamin
Created As-Is and To-Be	10/6/2025	Rayeed Zarif
Final edit	10/14/2025	Aryana Khaffaf Sharif Zamin

Stakeholders

Primarily for

Visually impaired users who need indoor navigation assistance.

Of

Seal T6 developers, ECSS department and corresponding professors.

Developed by

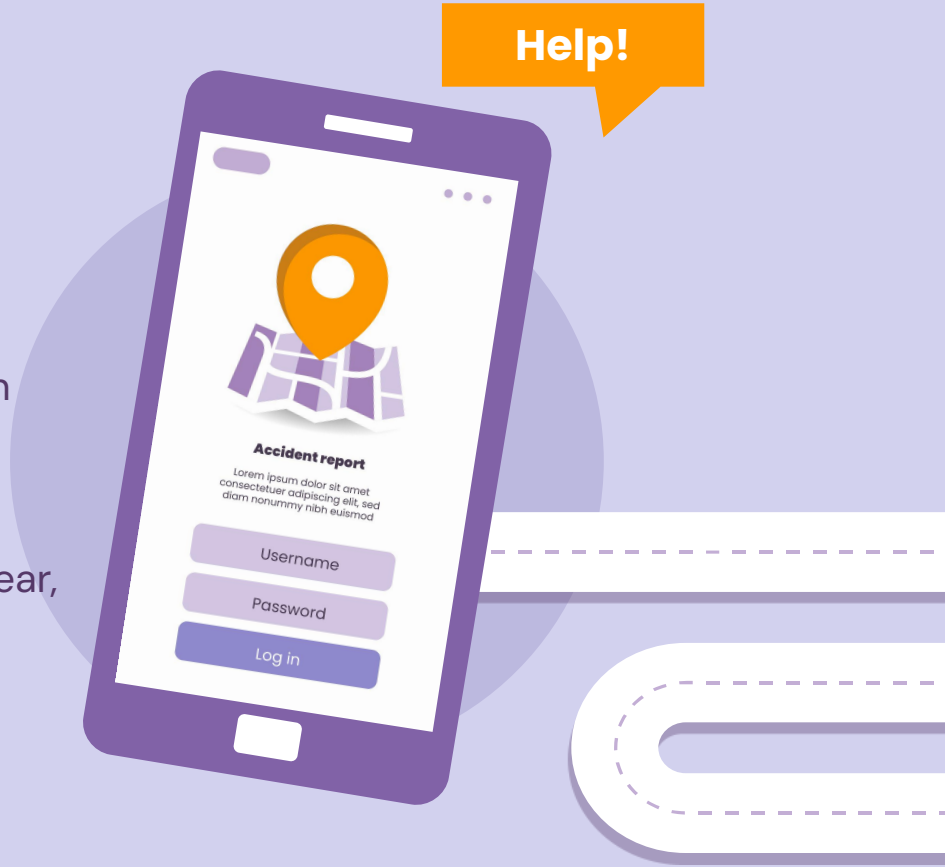
Seal T6 developers.

Tools

- Figma
- Node.js
- React
- Discord
- Google Docs/Slides
- Github

Problem Statement!

- Traditional tools like canes, guide dogs, and braille signs are limited indoors.
- Blind individuals still face confusion and safety risks when navigating buildings.
- A smarter, interactive aid is needed—one that can think, see, hear, and speak.



Project Goal

Features

- Directs users to destinations via optimal paths.
- Detects and alerts users about obstacles.
- Enables quick emergency contact if needed.

Focus

- Safety
- Independence
- Accessibility

Coverage Comparison

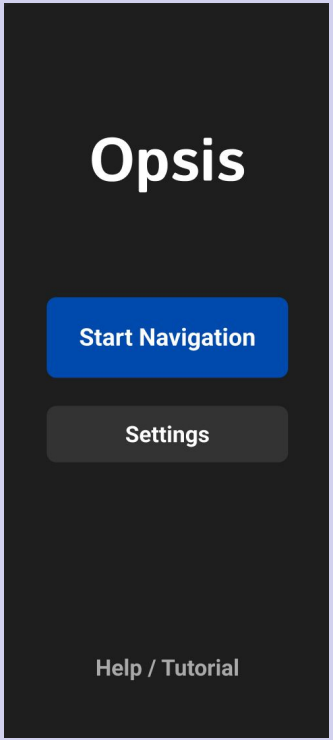
	White cane	Guide Dog	Braille markers	Opsis
Detects nearby obstacles	Yes	Yes	No	Yes
Provides navigation/direction	No	Yes	No	Yes
Works indoors effectively	Yes	Yes	No	Yes
Communicates with user	No	Yes	No	Yes
Alerts about emergencies	No	No	No	Yes
Offers independence and safety	Yes	Yes	No	Yes

Scenario Comparison-AS-IS

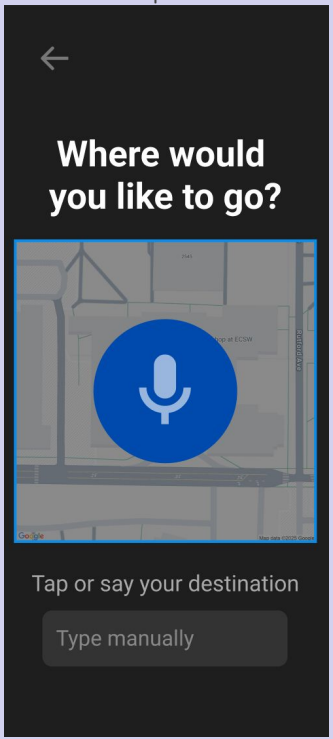
Scenario Comparison-TO-BE

Prototype

Home Screen

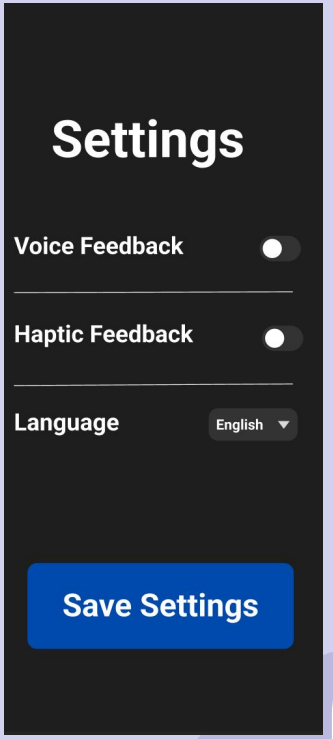


Destination Input



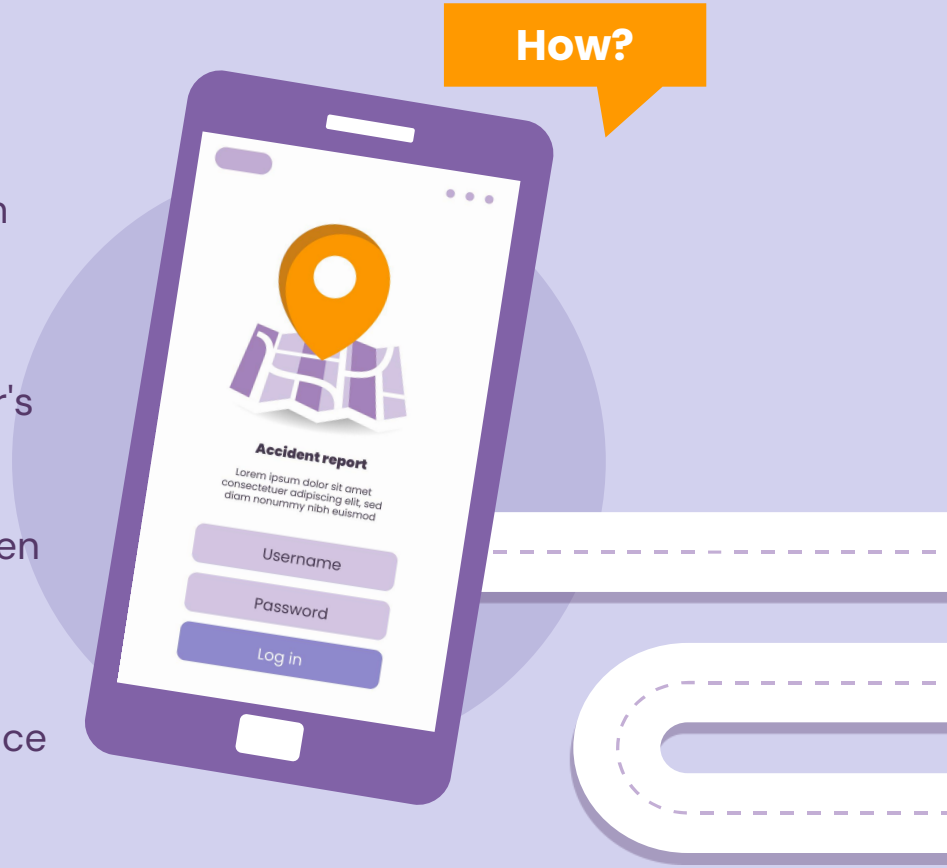
Navigation

Settings

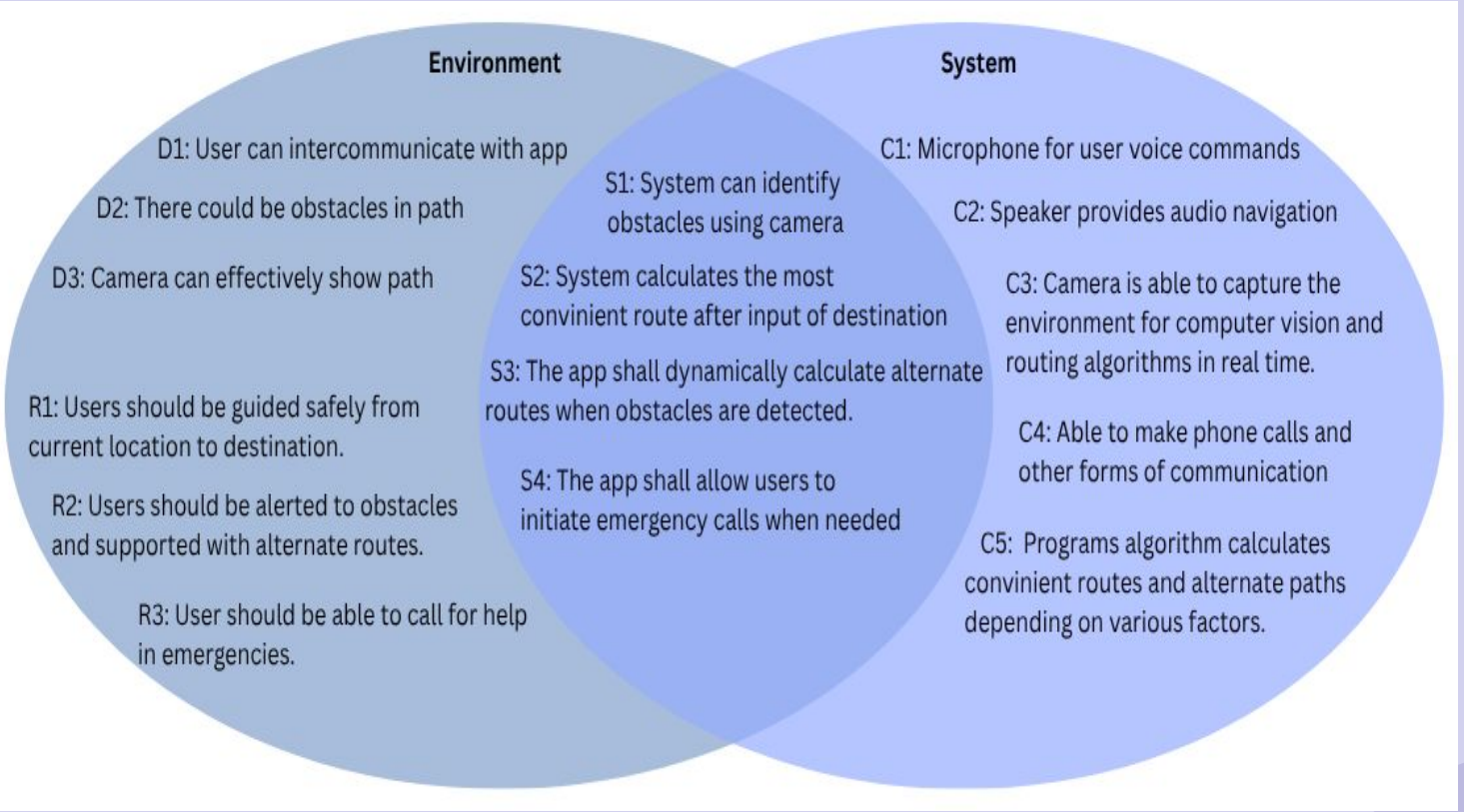


User Manual

- "The system will allow users to set their destination using a simple voice input
- The system will generate indoor routes from the user's current location to the selected destination
- The system will detect obstacles in the user's path and notify them immediately
- The system will update or reroute paths when obstacles and changes occur
- The system will provide step-by-step navigation instructions to the user using voice commands



WRSPM Model


Environment
D1: User can intercommunicate with app
D2: There could be obstacles in path
D3: Camera can effectively show path
R1: Users should be guided safely from current location to destination.
R2: Users should be alerted to obstacles and supported with alternate routes.
R3: User should be able to call for help in emergencies.
System
C1: Microphone for user voice commands
C2: Speaker provides audio navigation
C3: Camera is able to capture the environment for computer vision and routing algorithms in real time.
C4: Able to make phone calls and other forms of communication
C5: Programs algorithm calculates convinient routes and alternate paths depending on various factors.
S1: System can identify obstacles using camera
S2: System calculates the most convinient route after input of destination
S3: The app shall dynamically calculate alternate routes when obstacles are detected.
S4: The app shall allow users to initiate emergency calls when needed

WRS

2.2 Issues with II.2 Software System Requirements: Functional Requirements

2.2.1 : Obstacle detection needs clarification.

- Description: Which sensors and inputs—camera, LiDAR, and accelerometer—will be utilized?
- Choices: external sensors versus camera-based machine learning.
- Decision & Justification: Make use of AI and smartphone cameras (scalable and affordable).

2.2.2 : Route guidance scope unclear.

- Description: Should the app only provide obstacle notifications or provide turn-by-turn navigation?
- Options: include restricted obstacle feedback versus turn-by-turn voice navigation.
- Decision and Justification: For independence, turn-by-turn is ideal, but begin MVP with obstacle + supervision.

2.3 Issues with II.3 Software System Non-Functional Requirements

2.3.1 Issue-A (Latency)

- Description: Our first draft did not specify how fast the app should respond to user commands. For navigation, slow or fast responses could make the app unsafe.
- Options: (a) keep it general, (b) define a measurable standard.
- Decision & Rationale: Define a standard: voice response should occur within 1-2 seconds. This allows for the requirement to be tested and see if the latency is within a reasonable time frame.

2.3.2 Issue-B (Device Compatibility)

- Description: The preliminary definition did not clarify if the app will work on both iOS and Android.
- Options: (a) start with one and eventually build the other, (b) build both simultaneously
- Decision & Rationale: We will focus on building one (Android) since it is more flexible for prototyping.

2.3.3 Security and privacy

- Description: The draft did not address how user data will be protected. Even in a class project, ignoring privacy could weaken credibility.
- Options: (a) Skip for now, (b) Add a simple privacy requirement.
- Decision & Justification: We will include a requirement that no sensitive data (like navigation history or audio recordings) is stored unnecessarily.

Questionnaire

- Background Information
- Navigation Experience
- Technology
- Features
- Concerns & Suggestions

Functional Requirements

FR1	Navigate users indoors with precision.
FR2	Request current location to plan optimal routes.
FR3	Alert users of obstacles in real time.
FR4	Recalculate alternate paths dynamically.
FR5	Guide users safely around crowds or obstacles.

Non-Functional Requirements

Usability	**Simple, clear voice-based interface, can be learned within a few minutes.**
Testability	**Easily tested across realistic indoor environments with at least 95% reproducibility in navigation.**
Accessibility	**Full voice input/output support.**
Operability	**Reliable performance; handles invalid inputs safely, response under 10 seconds.**
Security	**Protected from malware and unauthorized access.**

Requirements Creep Management

- About 15-20%
- Implement strict version control and documentation.
- Hold weekly reviews to confirm feature scope.
- Add new features only after stakeholder approval.
- Regular feedback loops with end-users for clarity.

Why We're the Best!

We solve a real problem — empowering the visually impaired to move freely and safely indoors.
Opsis doesn't assist — it empowers. It thinks, sees, and speaks for the user.
All-in-one solution: Navigation, obstacle detection, and emergency support in one app.

References

“A template for a presentation” - Dr. Chung

“Project Phase I: Requirements Elicitation: Initial Understanding” - Dr. Chung

IEEE Software Requirements Specification (SRS) format

Appendix

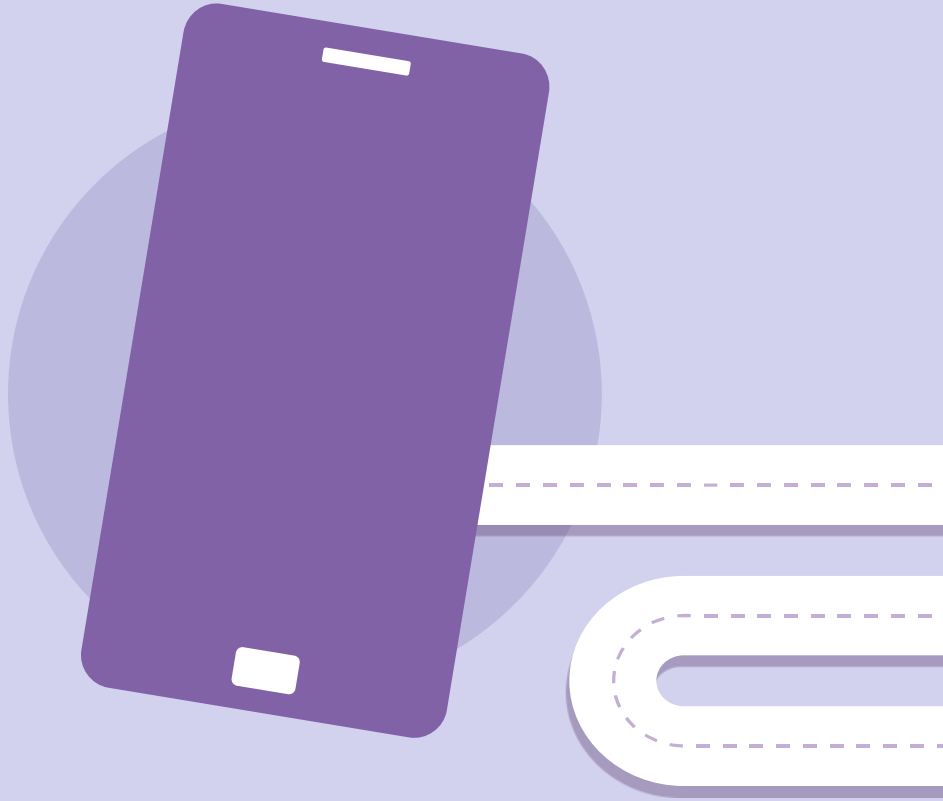
A. Project Links
- Team Website: https://sgs240001.wixsite.com/theia-spmp
- AS-IS Scenario Video
-TO-BE Scenario Video

B. Tools Used
- Figma – UI/UX Design
- Node.js – Backend
- React – Frontend
- Discord – Communication
- Google Docs/Slides – Documentation
- GitHub – Version Control

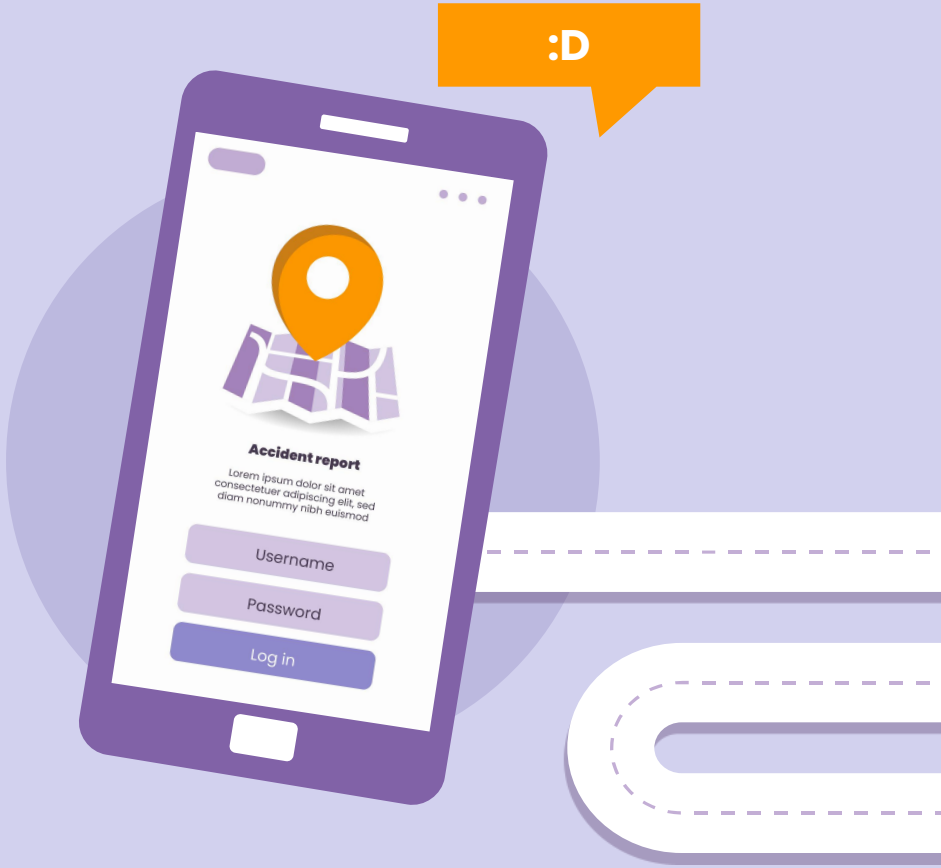
C. Supporting Documents
- Revision History Log
- Questionnaire Results
- WRSPM Model Diagram

Index

Questions?



Thank you!


:D
Accident report
Lorem ipsum dolor sit amet consectetuer adipiscing elit, sed diam nonummy nibh euismod
Username
Password
Log in