Project NavCampus

Navigation made simple.

Questionnaire Overview / Multiple-Choice Questions

- How are you navigating on-campus?
  - Mostly between different buildings
  - Mostly within one building, on different floors
  - Mostly within one building, on the same floor

- What kind of mobile device do you use?
  - Smartphone
  - Smartwatch
  - Tablet

- What senses do you use to operate your preferred mobile device?
  - Hearing
  - Touch
  - Sight

- How do you interact with your preferred mobile device?
  - Braille Keyboard
  - Voice controls
  - Touch feedback/controls

- What do you value *most* in an interface?
  - High-contrast
  - No reliance on colors
  - Customization options

- What platform do you use?
  - iOS
  - Android

3.1.7 Stakeholder issue – User Differences

- There are a lot of different types of blindness. Some just see bad, some have fragments in their sight and others cannot see anything at all. The app would need to cater all those different needs. This would require a lot of resources.

- Options: There is the option to create a version for every kind of blindness and making the app adjustable. The other option is to create one version for everybody that is based on the needs of completely blind persons.

- Decision: In this project it is the best possible solution to just create just one version of the app because of the limited time and the lack of time and resources.

3.1.8 Stakeholder issue – Challenges with Being Blind

- The users have difficulties finding their way around because they cannot interact with the visual designs for navigation. Therefore, they often don't know where they are exactly. The hard terrain on the campus, that is due to being built on a hill further makes it hard to find out on which floor you came in.

- Options: One way would be to put identifiers on the door of every building so the app can notice these. A sign with braille would generally also be a good idea.

- Decision: For the sake of the app an identifier works best, because it makes it easier to locate the exact location.

3.1.9 Functional Requirement Issue – Safety

- The app needs to be safe to use. That means the user can depend on the accurate functioning of it, so that no accidents happen that were caused by the app itself. Examples include, that the app doesn't tell the user about an object and the user falls because of it, which could lead to injuries or the inability to orientate themselves. Also, it should be able to guide a safe route.

- Options: walk with cane → two systems, fall sensor, so that the phone can call for help in case of emergency

- Decision: There is no guarantee for safety when not using the cane at the same time and a fall sensor will be implemented that can call for help.

3.2.5 Functional Requirement Issue - User Accessibility

- The app is to be designed for visually impaired students. Therefore, the app can't only be operated by normal means. This means it needs to be usable without seeing the actual display of the smartphone.

- Options: voice commands and audio-based navigation, special blind keyboard, compatibility with screen readers and accessibility features

- Decision: audio-based navigation, voice commands and compatibility

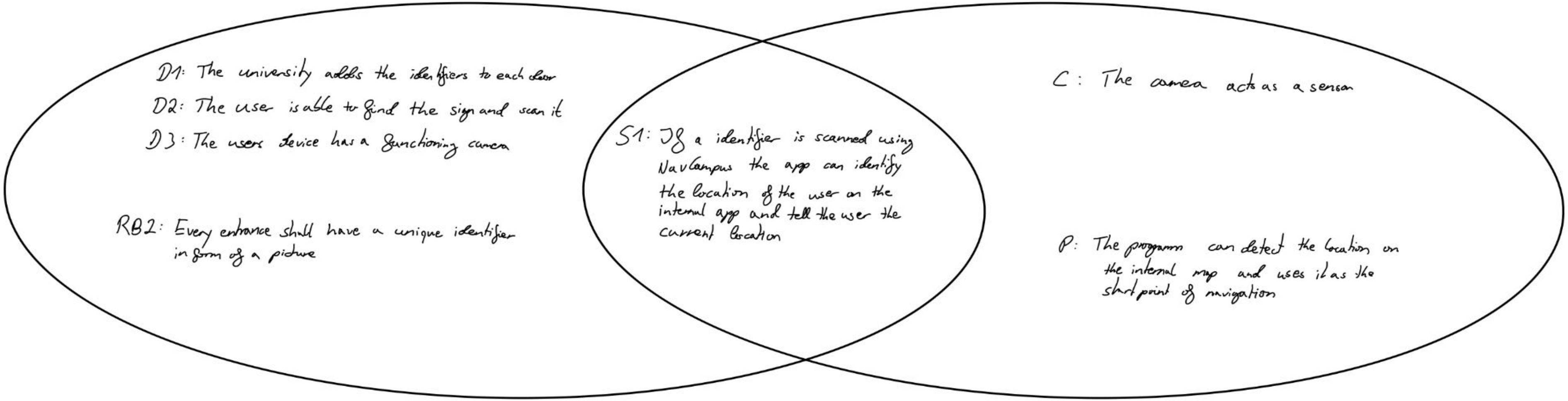
3.3.3 Non-functional Objective Issue - Data Security

- Data security is key when working with camera detection and the collection of the location data of the users.
- Options: implement data encryption and data protection vaults, update security protocols, saving as less data as possible
- Decision: Because the data is needed for the machine learning to function there needs to be data encryption and the data is only to be seen by the system itself.

3.3.1 Non-functional Objective Issue – Performance

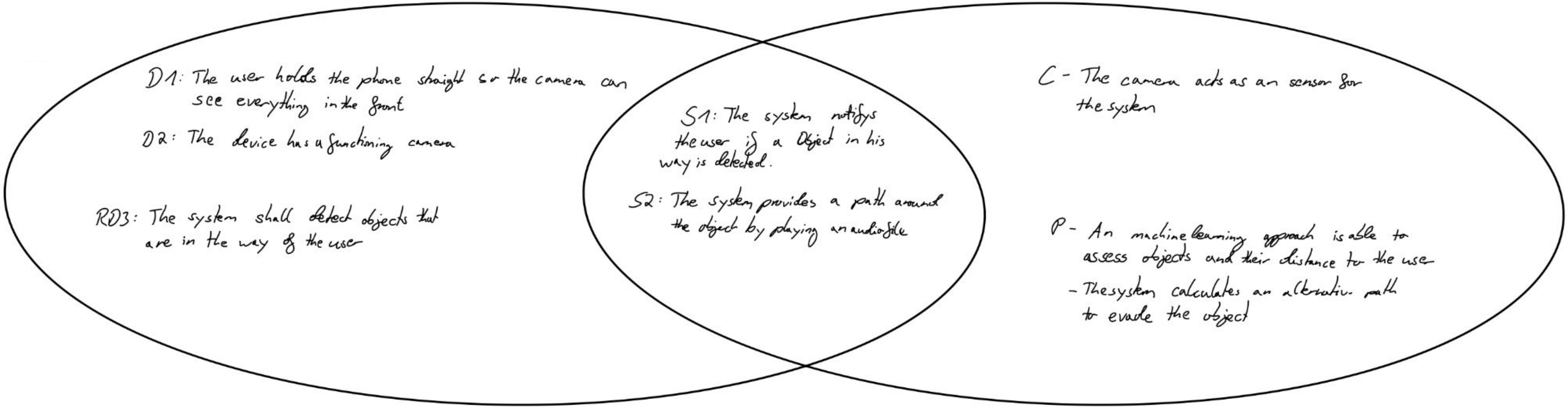
- The app handling real-time data means that it could be hardware heavy and could lead to performance issues on hardware-weaker devices. This could lead to failure to assist the user or even could lead to safety issues when the user cannot be notified about obstacles.
- Options: optimize algorithm for minimal resource consumption, minimum hardware requirements
- Decision: Because of the lack of resources there will be hardware requirements even so the optimization would be the better option to help as many students/staff members as possible even when they don`t have the financial ability to pay for the newer devices.

Reference Model



eh: All activities of the user as well as the general presence of a identifier
ev: When there is an identifier, the scanning of it as well as the users interactions with the device
sv: the algorithm to find the users location as well as well as the map itself
sh: the acustic confirmation of the location

Reference Model

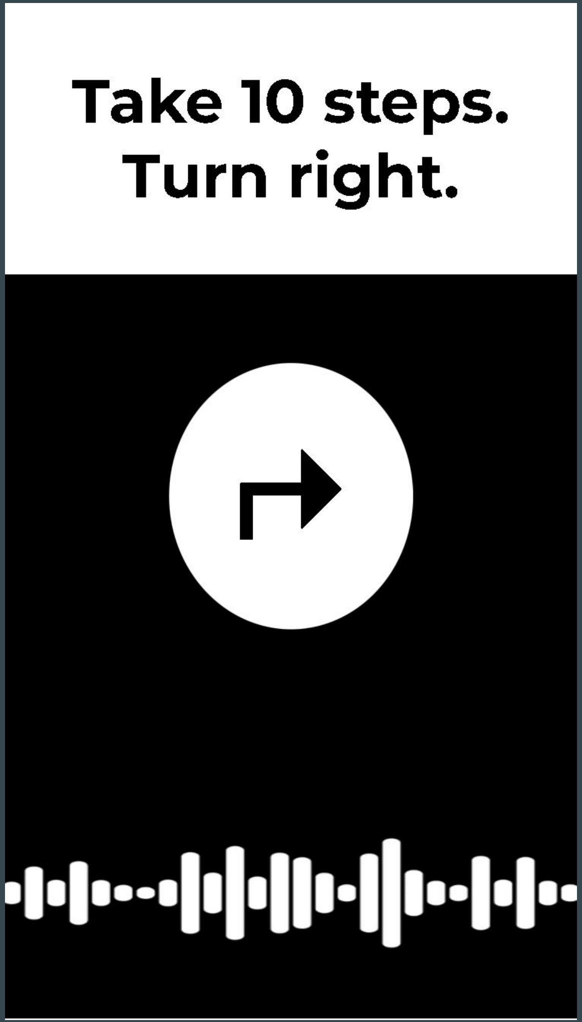


eh: The user uses the device correctly (straight), the user is able to hear the system
ev: Objects/Path in front of the camera
sv: Notification of detected object + instructions of the alternate path
sh: the algorith that calculate the distance, the detection and the alternative path

How navigation works

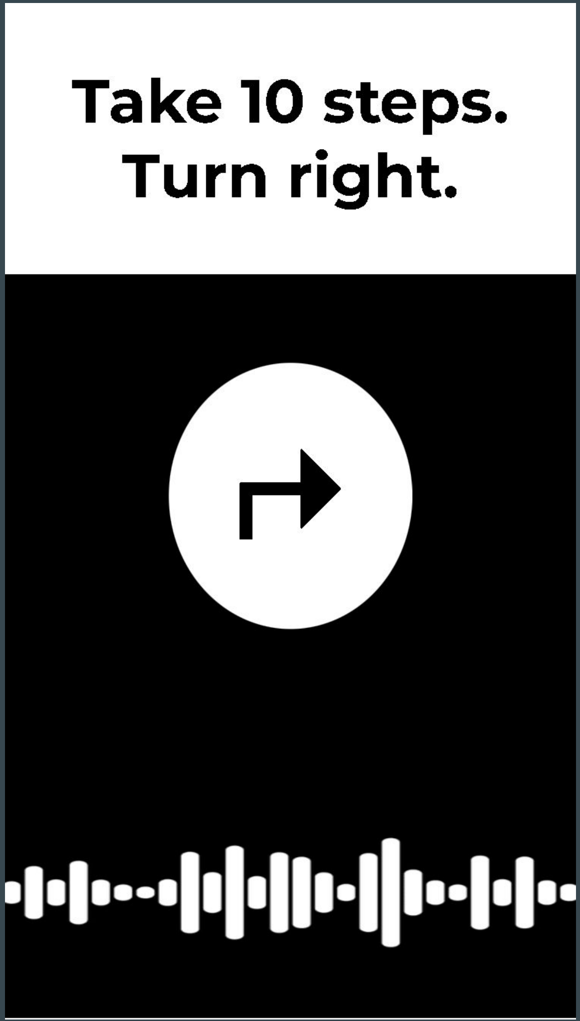
- The app uses a internal map of the building to guide users. This map includes:

  1. Room location
  2. Entrance location
  3. Bigger furniture location



How navigation works

- Every location (entrances, rooms) has a unique identifier.
- The app shall detect the identifier and be able to tell which floor the user is on, and what rooms are around them.
- The system shall take the GPS to detect the direction the user is facing and in which direction he/she/they are moving.
- The system shall use the collected health data of the phone to get the average step length of the user.



AS-IS

The current state of things

- Alice is a visually impaired student at Cal Poly Humboldt.
- She relies on her cane and auditory cues from people around her to find the correct building and room.
- She cannot use indoor maps. Furniture placement can change without her knowledge.
- Alice must memorize the layout of each building and remember the locations of her classrooms.

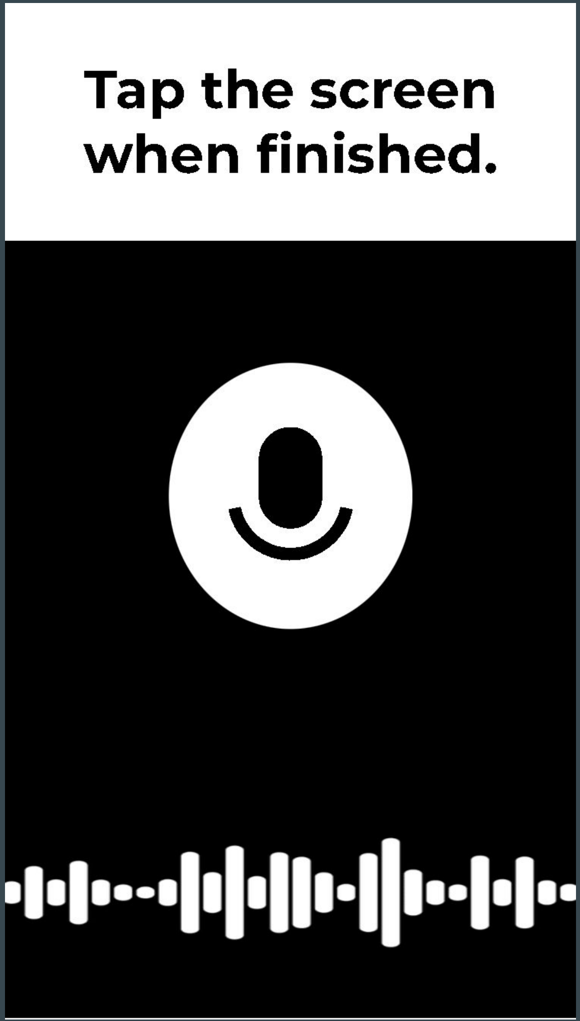
TO-BE

How can we do better?

- Alice uses the NavCampus app on her mobile device.
- She uses the audio input feature to specify her desired location, like a building and room number.
- The app's detection system alerts her to any obstacles in her path, and suggests an alternative route.
- The app provides clear audio directions, such as the nearest entrance/route to a desired location.
- In case of an unexpected fall, the app's detection system starts a cancelable countdown. In the worst case, the app alerts the uni police dept. to her location.

Voice Input

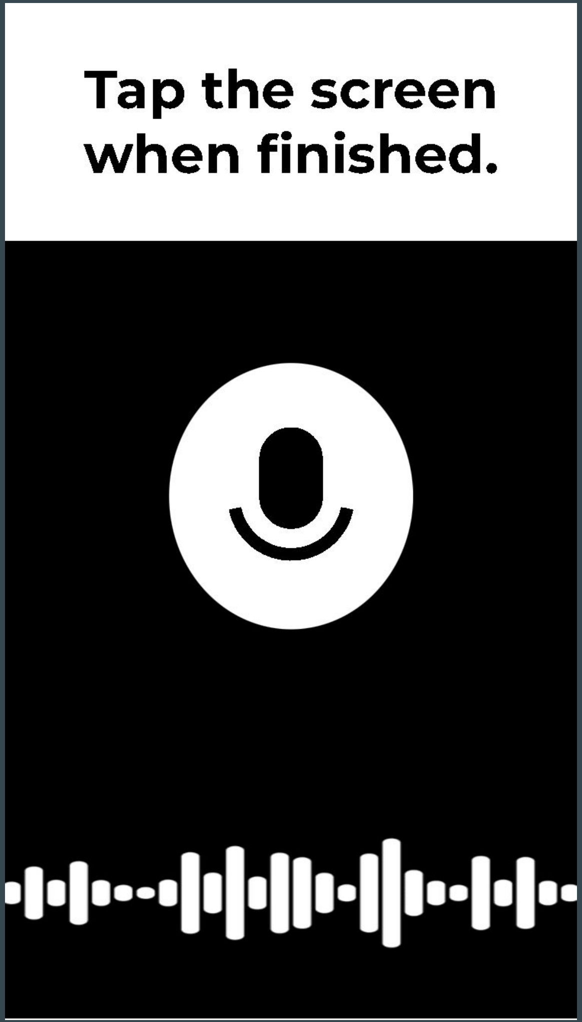
- To activate the voice input feature, Alice simply taps anywhere on her mobile device's screen.
- She then states the building and/or room number she would like to travel to.
- Once Alice is finished speaking, she taps anywhere on her mobile device's screen again.
- The app determines where Alice is, and where she needs to go, all "behind the scenes". This is not shown.
- Or, alternatively, if voice input is not understood, the app audibly instructs Alice to tap the screen and activate the voice input feature again.



What functional requirements are satisfied?

4.2.1 (RA) UI

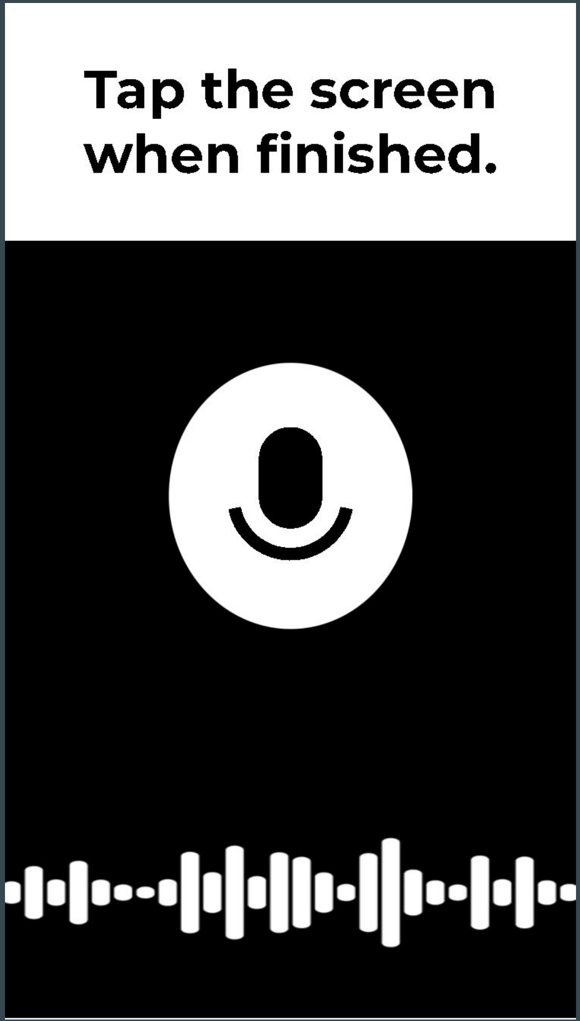
- RA1. The app shall have a voice input feature, to give it the desired location.
- RA3. The app shall instruct the user how to use it with audio lines.
- RA6. The voice command shall end if the user taps the screen.
- RA7. When using the voice command, the system shall notify the user after 30 seconds to tap the screen to end the voice command.



What non-functional requirements are satisfied?

<u>4.2.2 Non-functional Requirements</u>

- NR1. The system shall be usable without the need to see.
  - The system shall provide audio lines for every instruction.
  - The system shall allow text to speech systems.
  - The app shall use high-contrast colors.



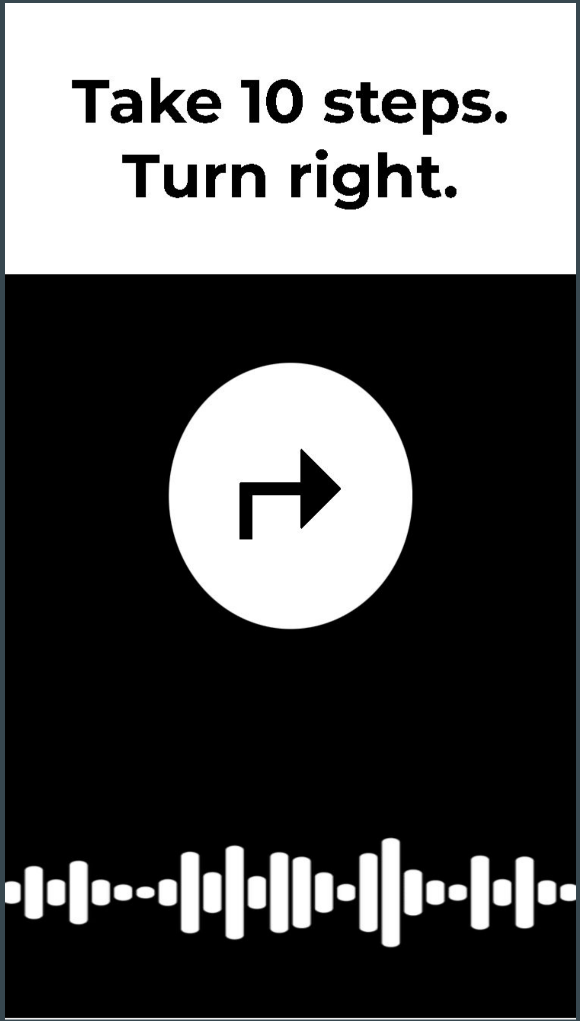
Navigation

- The app's navigation interface is simple. Only text is shown, and is spoken aloud. Also, a simple directional arrow is displayed.

- This arrow on screen changes with Alice's direction/proximity to her destination.

- The text on screen is spoken aloud every thirty seconds, to ensure Alice is still able to use the app even if she is completely blind.

- If Alice is not satisfied with these directions, she can tap anywhere on the screen to activate the voice input feature again.



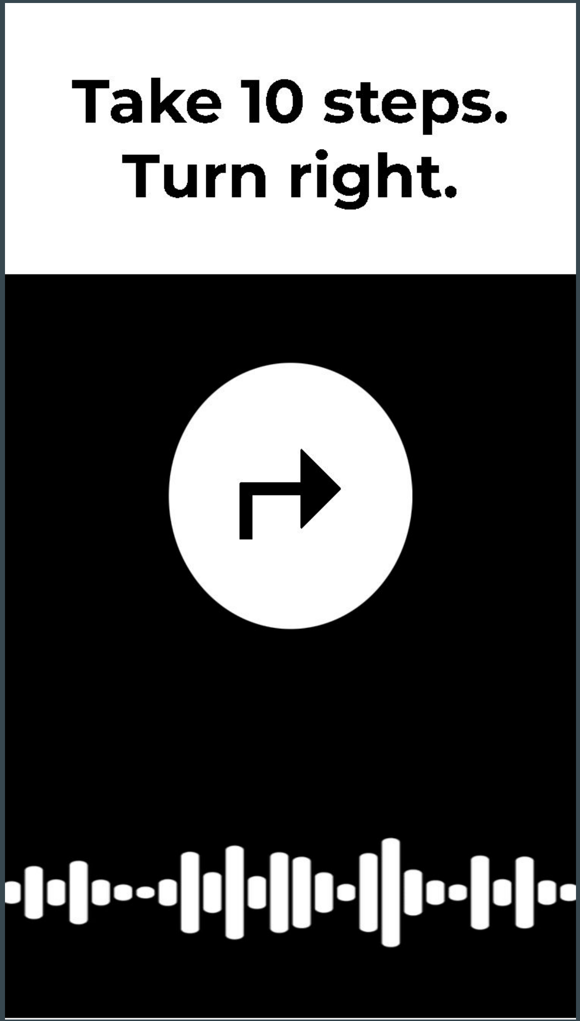
What functional requirements are satisfied?

4.2.1 (RA) UI

- RA5. The app shall show directions in form of an arrow and the needed steps until then.
- RA5. An arrow will point to the direction of navigation.
- RA4. The app shall give the navigation data via audio.

4.2.1 (RC) Navigation

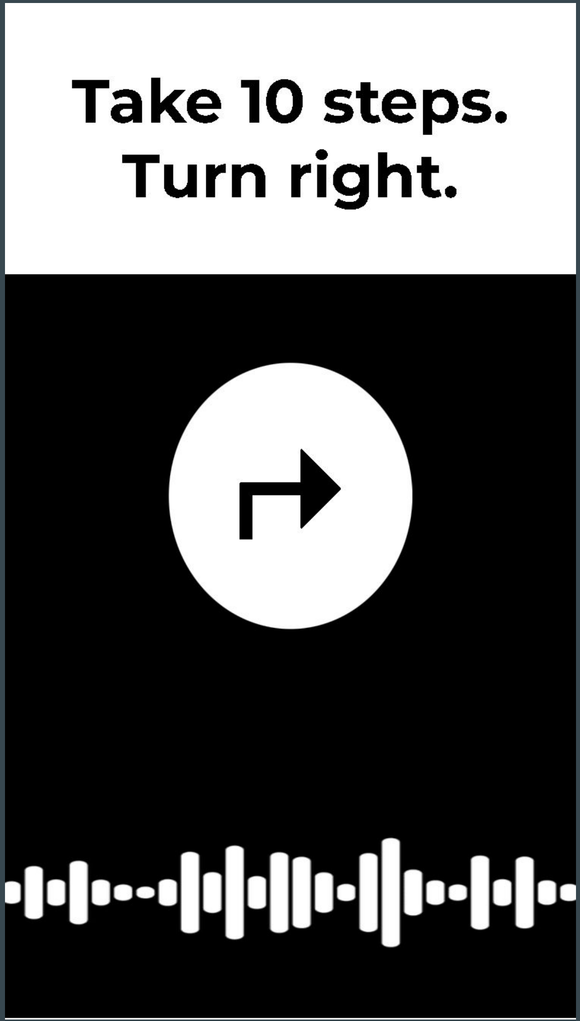
- RC6. The step count shall be used to calculate how many steps are needed to reach the next turn point.



What non-functional requirements are satisfied?

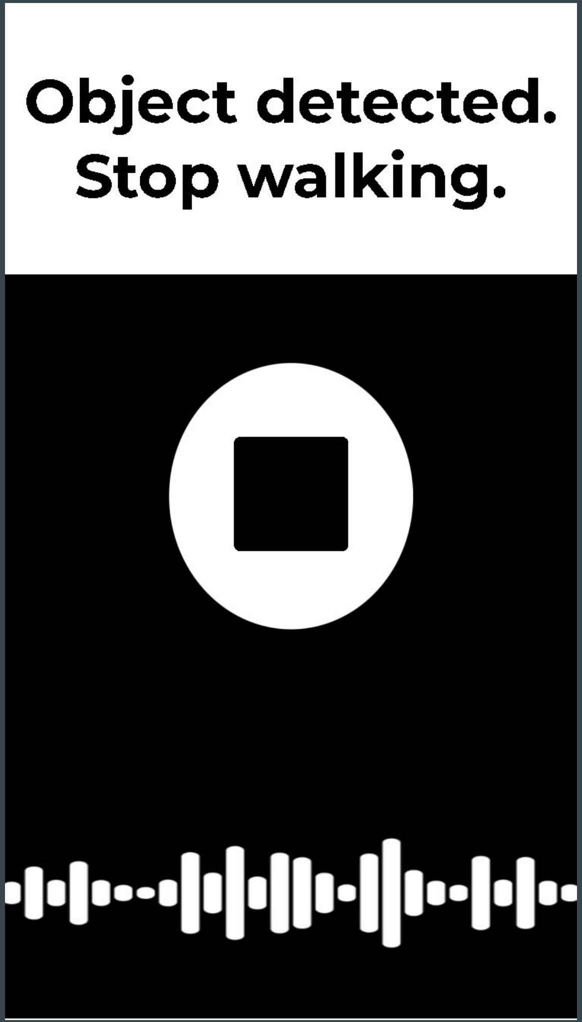
<u>4.2.2 Non-functional Requirements</u>

- NR1. The system shall be usable without the need to see.
  - The system shall provide audio lines for every instruction.
  - The system shall allow text to speech systems.



Object Detection

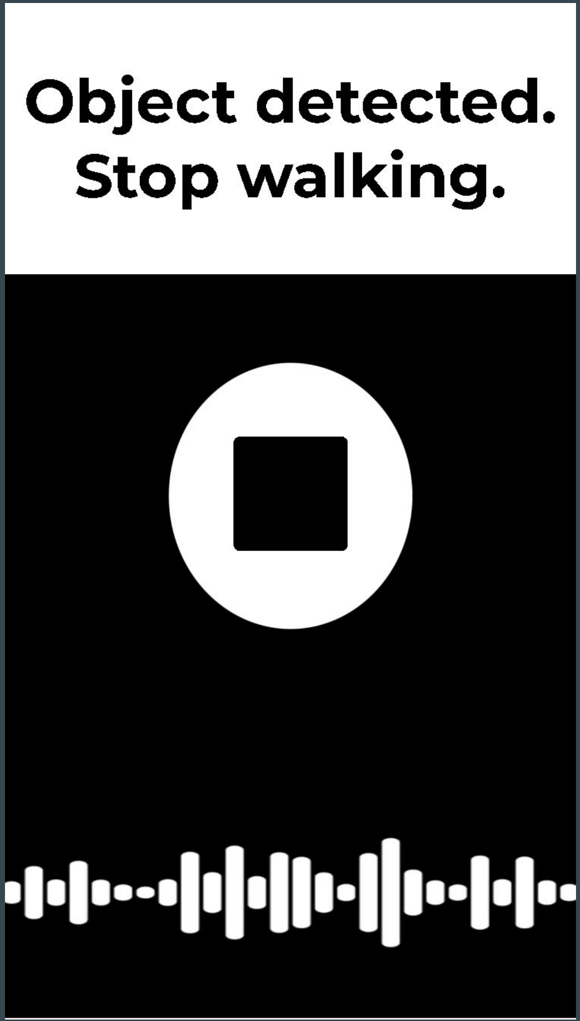
- The app uses Alice's mobile device's camera to detect any objects that may be in her path.
- The camera feed is checked against a map of the building's layout, which is updated consistently.
- If an object is detected that may hinder Alice's movement, the device audibly instructs her (every 10 seconds) to stop walking, as shown.
- Once Alice is stationary, the app shifts back to navigation mode; and so navigates her around the object, or uses an alternate route.



What functional requirements are satisfied?

<u>4.2.1 (RD) Object Detection + Notification</u>

- RD3. The system shall detect objects that are in the way of the user, excluding the cane.
- RD5. The system shall notify the user where the object is with an audio line, and on the screen.
- RD4. The system shall have a machine learning algorithm to learn to detect objects and the distance to the user, using the camera of the device.

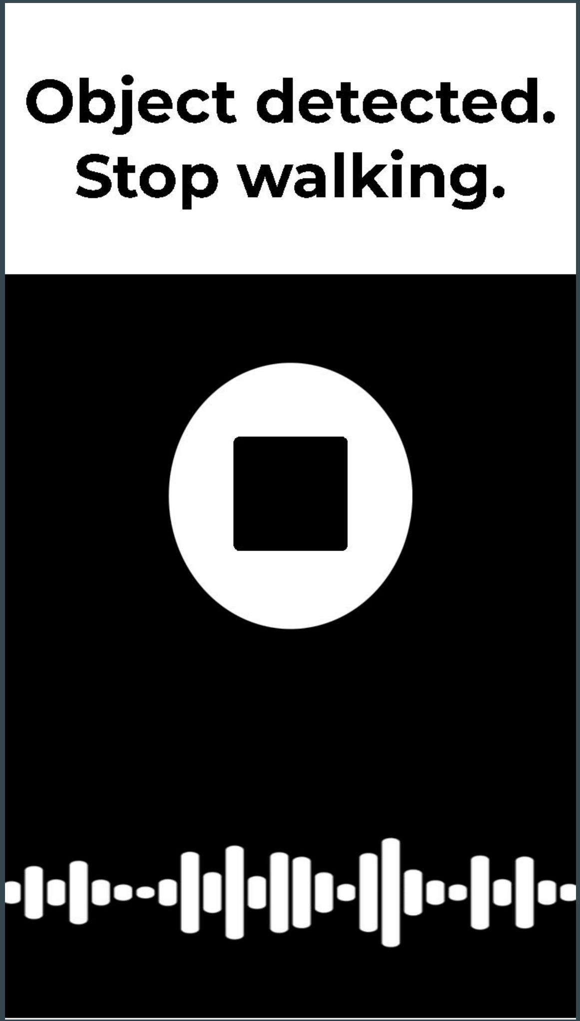


What non-functional requirements are satisfied?

<u>4.2.2 Non-functional Requirements</u>

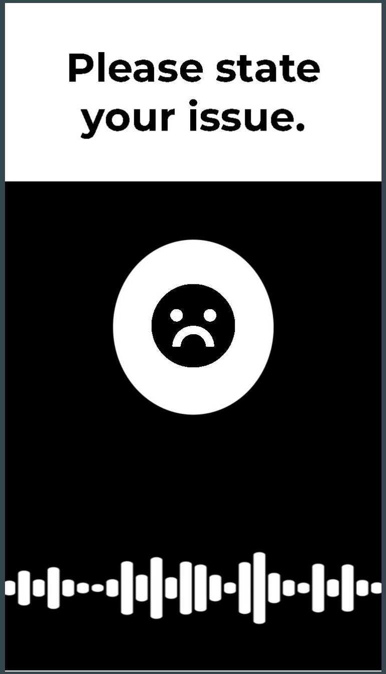
- NR1. The system shall be usable without the need to see.
  - The system shall provide audio lines for every instruction.
  - The system shall allow text to speech systems.

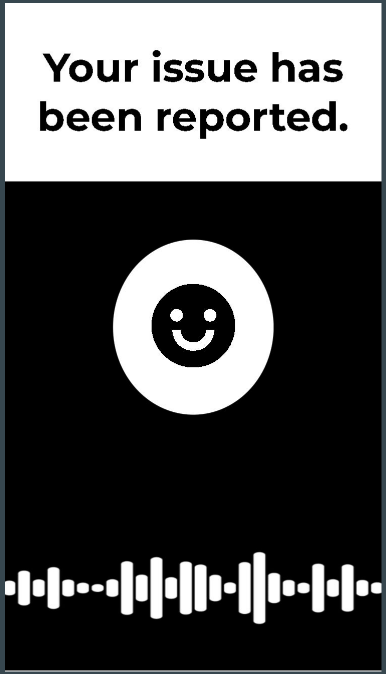
- NR5: The system's main concern shall be safety.
  - The user shall use the cane while usage, when the user normally uses one.
  - The system shall have a fall detector by using the device's sensors.
  - The system shall take the safest route.



Reporting Feedback

- If Alice is unhappy with her directions, she can activate voice input mode by tapping the screen *once*, as previously stated.
- However, if Alice has a deeper issue with the app that she cannot fix, she may tap the screen *three* times.
- The app audibly instructs Alice to state her issue. Once Alice is finished speaking, she may simply tap the screen *once* to stop recording.
- This audio recording file is then sent, via email, to a developer's inbox, where it may be inspected and acted upon.
- Alice is returned to the voice input screen, where she can dictate more directions, if desired.

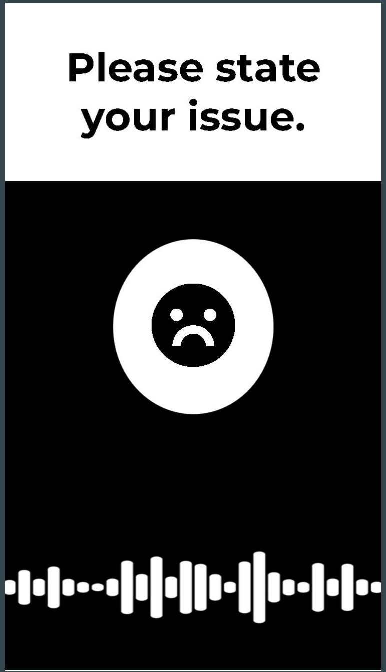


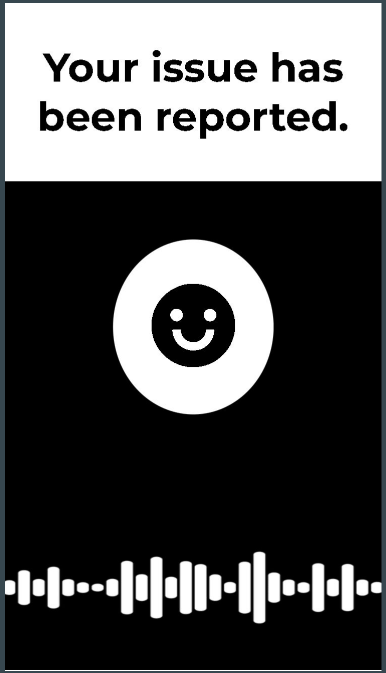


What functional requirements are satisfied?

4.2.1 (RF) User Feedback

- RF1. The user shall be able to give feedback inside the app through a direct message.
    - When the device detects a “triple tap”, and the app is not in fall detection mode, the user feedback interface shall open.
    - The app will begin recording. The user can state their issues/feedback, and tap the screen to end recording.
    - The app saves this recording as an audio file, and sends this file directly to the developer’s email address.

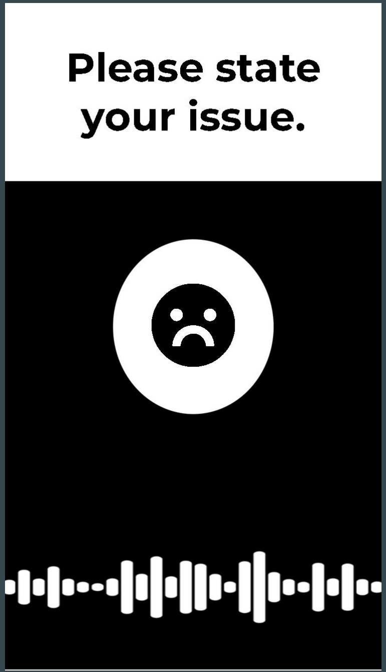


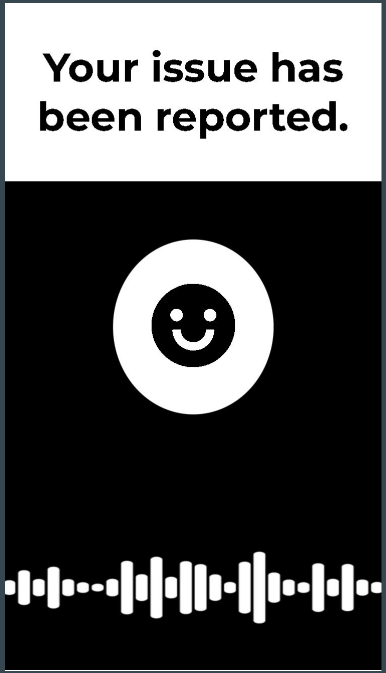


What non-functional requirements are satisfied?

<u>4.2.1 (RF) User Feedback</u>

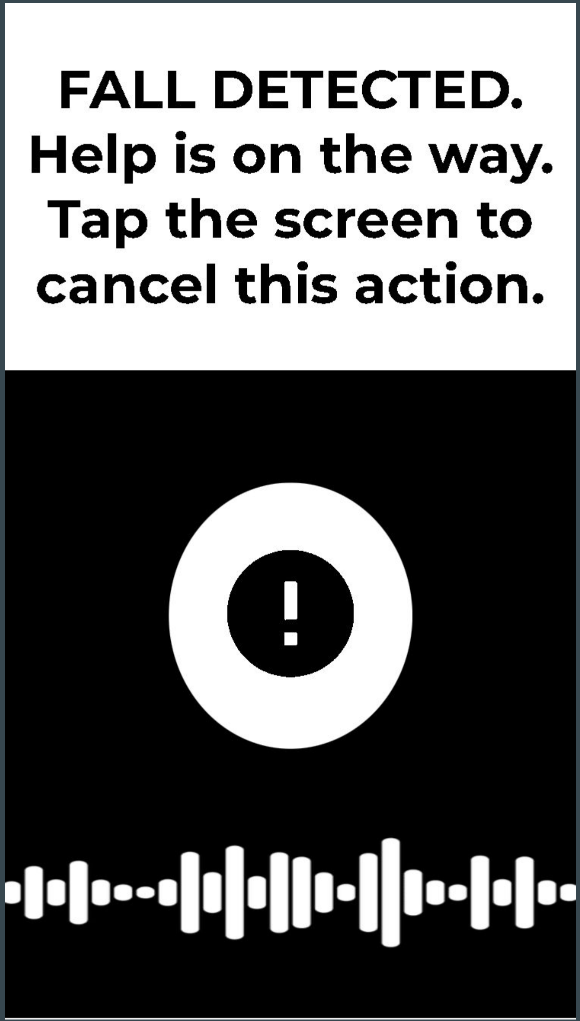
- RF1. The user shall be able to give feedback inside the app through a direct message.
- RF3. The message shall be sent to the developers mail address.





Fall Detection

- The app detects Alice’s general speed/movement via the device’s internal sensors, as is common with other apps.
- If a sudden change to Alice’s speed is detected, or an intense “shake” motion, the fall detection mode is activated.
- The onscreen text is audibly spoken every 10 seconds, to ensure Alice (or any bystanders) can hear it.
- If there is no emergency, Alice may simply “triple tap” the screen to cancel this mode. The app resumes what it was doing previously, whatever it was.
- If this is a genuine emergency, a message is sent to the university police/emergency response equivalent after 30 seconds.

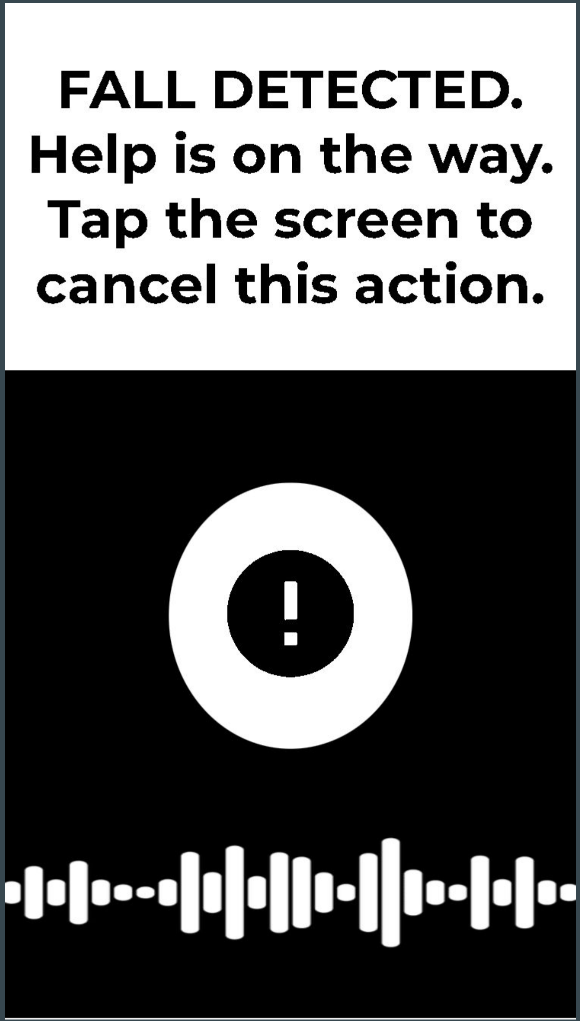


What functional requirements are satisfied?

<u>4.2.1 (RE) Emergency Call + Fall Detection</u>

- RE1. The fall sensor shall be activated throughout the use of the app.
- RE2. In case of a fall a countdown shall be activated.
  - The countdown is 30 sec long.
  - During the countdown the app uses audio lines to get the user to triple tap on the screen.
  - The countdown deactivates after a triple tap.

- RE3. In case that the countdown is not deactivated, the system shall call the university police department to get help. The system shall provide them with the current location.



13%

Our estimated creeping rate

Why NavCampus?

- The user interface is simple: tap, speak, or both
- Power efficient, doesn’t do anything it doesn’t need to
- Anyone can use it, regardless of vision impairment level
- Focused on the lowest common denominator; makes use of features most mobile devices already have
- Doesn’t need external hardware for navigation and is therefore cheap