Section 1 - Project Description

1.1 Project Title
Project SafeWalk

1.2 Description
SafeWalk is a mobile application designed to improve user safety during commutes, especially in unfamiliar or unsafe environments. By offering real-time location tracking, emergency notifications, and user-friendly interfaces, the application empowers individuals to feel more secure when walking alone.

1.3 Intended Audience
The intended audience for Project SafeWalk are individuals who require enhanced safety measures during their commutes, particularly in unfamiliar or perceived unsafe areas. This mobile application is aimed at providing peace of mind to users who often travel alone, by offering real-time location tracking, emergency notifications, and a user-friendly interface. It is especially beneficial for users such as students, night-shift workers, and travelers who may find themselves in vulnerable situations, ensuring that they have a reliable means to communicate with trusted contacts and access assistance when needed.

1.4 Revision History

Date	Comment	Author
10/15/2023	Created SDD.	Paramvir Toor Julian Prater Alfredo Gonzalez Yeng Yuatongjerxiong Mayuka Nozaki Aaron Cantu
10/29/2023	Updated sections 4, 5, and 6.	Paramvir Toor Julian Prater Alfredo Gonzalez Yeng Yuatongjerxiong Mayuka Nozaki Aaron Cantu
11/16/2023	Modified diagrams (state chart, class diagram, use case). Made edits to most descriptions in all sections to be more inline with scope. Added Wireframe diagram to section 7.	Paramvir Toor Julian Prater Alfredo Gonzalez Yeng Yuatongjerxiong Mayuka Nozaki Aaron Cantu

Team Members

Paramvir Toor

Julian Prater

Alfredo Gonzalez

Yengtaova Yuatongjerxiong

Mayuka Nozaki

Aaron Anthony Cantu

Table Of Contents

Section 2 - Project Overview

2.1 Purpose

The purpose of Project SafeWalk is to reduce the risk that users are involved in accidents like stalking and kidnapping when walking alone, in unfamiliar places, and in unsafe environments. To achieve this purpose, the application has real-time location tracking to share a current location between trusted users, emergency notifications to warn users of danger, and user-friendly interfaces to improve users' safety when walking alone.

2.2 Scope

The project deliverables encompass the following:

- A database infrastructure to store user data and facilitate real-time communication with emergency contacts.
- Comprehensive user interface design focusing on simplicity, accessibility, and intuitive navigation.
- Provide geolocation notifications/updates with trusted contacts based on the user decisions.

2.3 Project Constraints

Some of the project constraints that are considered for this project are the limitations of what the Ionic/AngularJS framework can do. Other constraints would also come from the capabilities that Node.js and Express can do as well along with our external interfaces such as Google owned technologies, Firebase, and AWS.

2.4 Requirements

2.4.1 Estimates

#	Description	Hrs. Est.
1	UX.UI Design	14
2	Database	20
3	Front End Implementation	15
4	Login/Signup Feature	10
5	Login/Signup with Google	5
6	Location Sharing Feature	15
7	Emergency Contact	8
8	Panic Button	10
9	Push notification id Contact	10
	TOTAL:	107

2.4.2 Traceability Matrix

Id #	SRS Requirement	SDD Module
1	1.1 Product Scope	2.2 Scope
2	2 Functional Requirements	2.3 Requirements

3	3 External Interface Req	8 Other Interfaces
4	5 Definitions and acronyms	11 Glossary

Section 3 - System Architecture

The architecture of this project makes sure that the flow of data is continuous, performs at a high level, and is reliable while prioritizing user safety. The architecture is a combination of client-side applications such as Ionic/Angular framework, server-side processes, database management systems, and third-party integrations.

3.1 Programming Languages

For the frontend of Project SafeWalk, the programming languages used are HTML, CSS, JavaScript/TypeScript all inside the Angular framework. As for the backend the language used is Node.js. You can expect to see client-side interfaces trigger backend database operations when applicable in the project.

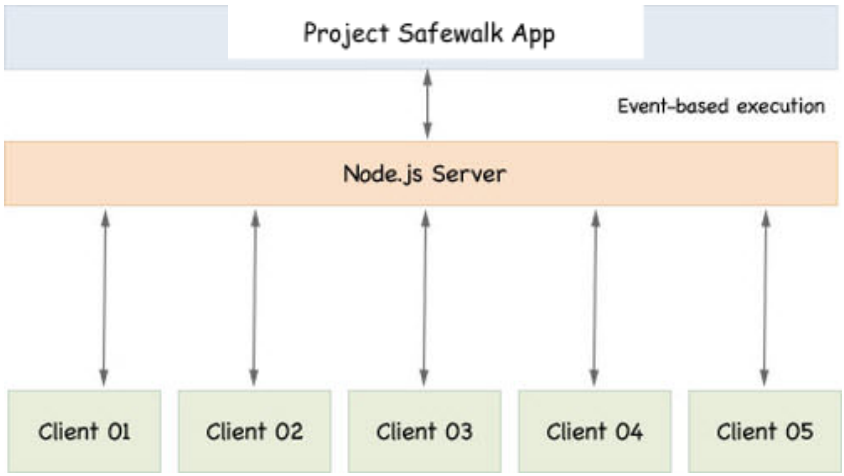


Figure 3.1.1: Sequence diagram.

Section 4 - Data Dictionary

This data dictionary provides detailed information about the data structures and the elements within the module.

Users		
Field	**Notes**	**Type**
user_id	Unique identifier for each user	INT
username	The username chosen by the user	VARCHAR
email	The user's registered email address	VARCHAR
password	The password created by the user	VARCHAR

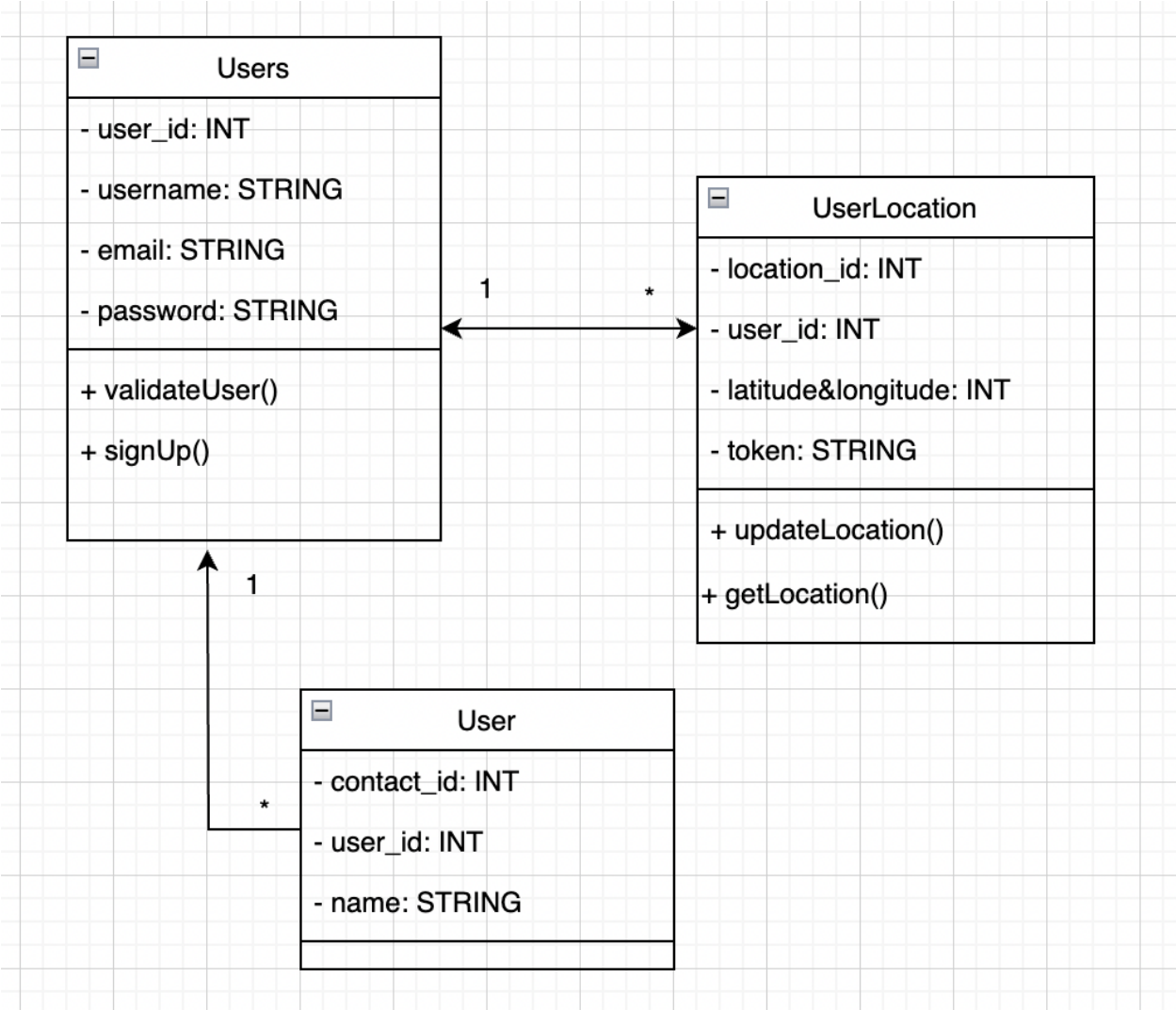
User		
Field	**Notes**	**Type**
user_id	Unique identifier for each user	INT
contact_id	Unique identifier for each contact	INT
name	Contact's name	VARCHAR

userLocation		
Field	**Notes**	**Type**
user_id	Unique identifier for each user	INT
token_number	Contact's push notification id	VARCHAR
location_id	Unique identifier for each location entry	INT
latitude	Latitude coordinate of the location	FLOAT
longitude	Longitude coordinate of the location	FLOAT

Section 5 - Software Domain Design

5.1 Software Application Domain Chart

Below is a representation of the software application domains and the relationship between objects. UML diagrams help visualize the architecture and flow within the system.



5.2 Software Application Domain

Within the project SafeWalk, the software application domain is divided into a number of smaller domains that each include a variety of functions. Each domain contains a family of elements that team up to bring functionality.

5.2.1 User Authentication and Login/Register Domain

This domain handles everything related to user accounts, including registration, login, profile management, and user settings. It integrates with the database domain to securely store user data.

5.2.1.1 Authentication of Users Component

This component ensures that users are who they claim to be through authentication. It also bounds the login and registration procedures.

Data Flow

1. User submits a login/registration form.
    a. If applicable to Google users, OAuth will handle user authentication.
2. Data is sent to the server.
3. Password encryption before storage or comparison.
4. User authentication status is sent back to the client.

5.2.1.1.1 Accepting User Credentials

This task involves accepting the user's credentials, validating them with the stored data, and return either a success message or an error message.

Data Flow

1. Retrieve 'username' and 'password' from a client.
2. Use a function that encrypts the password.
3. Validate the credentials against the 'Users' table.
4. If valid, return the success message, else return an error.

Section 6 – Data Design

For Project SafeWalk, here is the design of the data and their relationship with the different functionalities and external datasets that are in place.

6.1 Persistent/Static Data

The following are descriptions of persistent and static datasets with Project SafeWalk:

6.1.1 Dataset

Relationship with users and trusted/emergency contacts, the relationship between security of the database.

6.1.2 Static Data

User preference settings, Geographical locations, Error messages and notifications, Emergency/Trusted contact list.

6.1.3 Persisted data

User account information, Location data, App logs, and Activity history.

6.2 Transient/Dynamic Data
Real-time status updates, User notifications, Emergency notifications, and Session information (meaning when the user initiates a walk within the app).

6.3 External Interface Data
Social media account linking, location Ionic-based services.

6.4 Transformation of Data
Geolocation coding, formatting of dynamic data for notifications.

Section 7 - User Interface Design

7.1 User Interface Design Overview
Some of the high-level requirements for Project SafeWalk include, but are not limited to:

- User Safety Features
- User registration and Authentication
- Contact Management
- Location Sharing
- User-friendly interface
- Database Management
- Integration with Firebase or other External Services
- Responsive Design
- Compatibility Security

7.2 User Interface Navigation Flow & Wire Frame

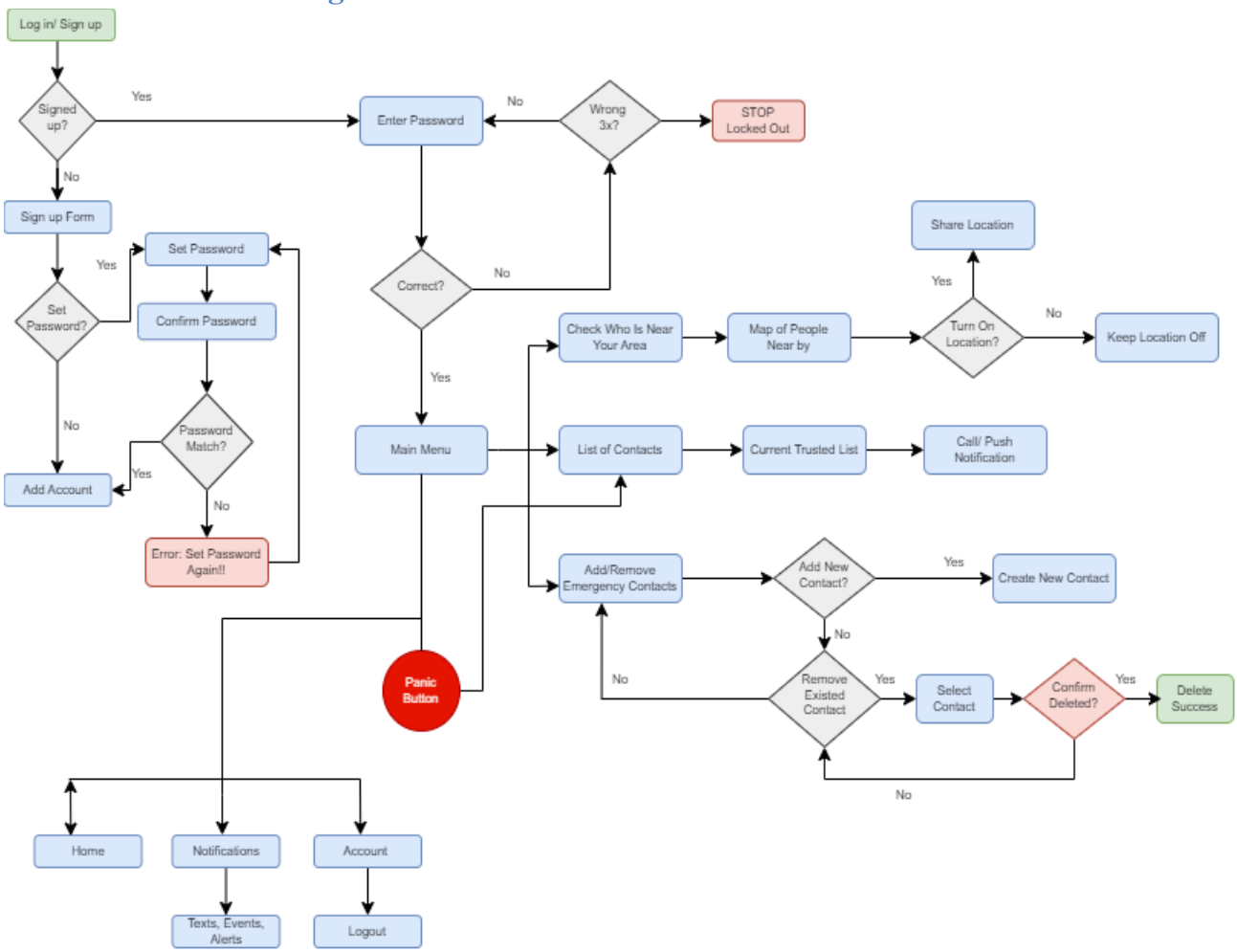


Figure 7.2.1: A State Chart diagram of how a user interacts and the path that will be followed.

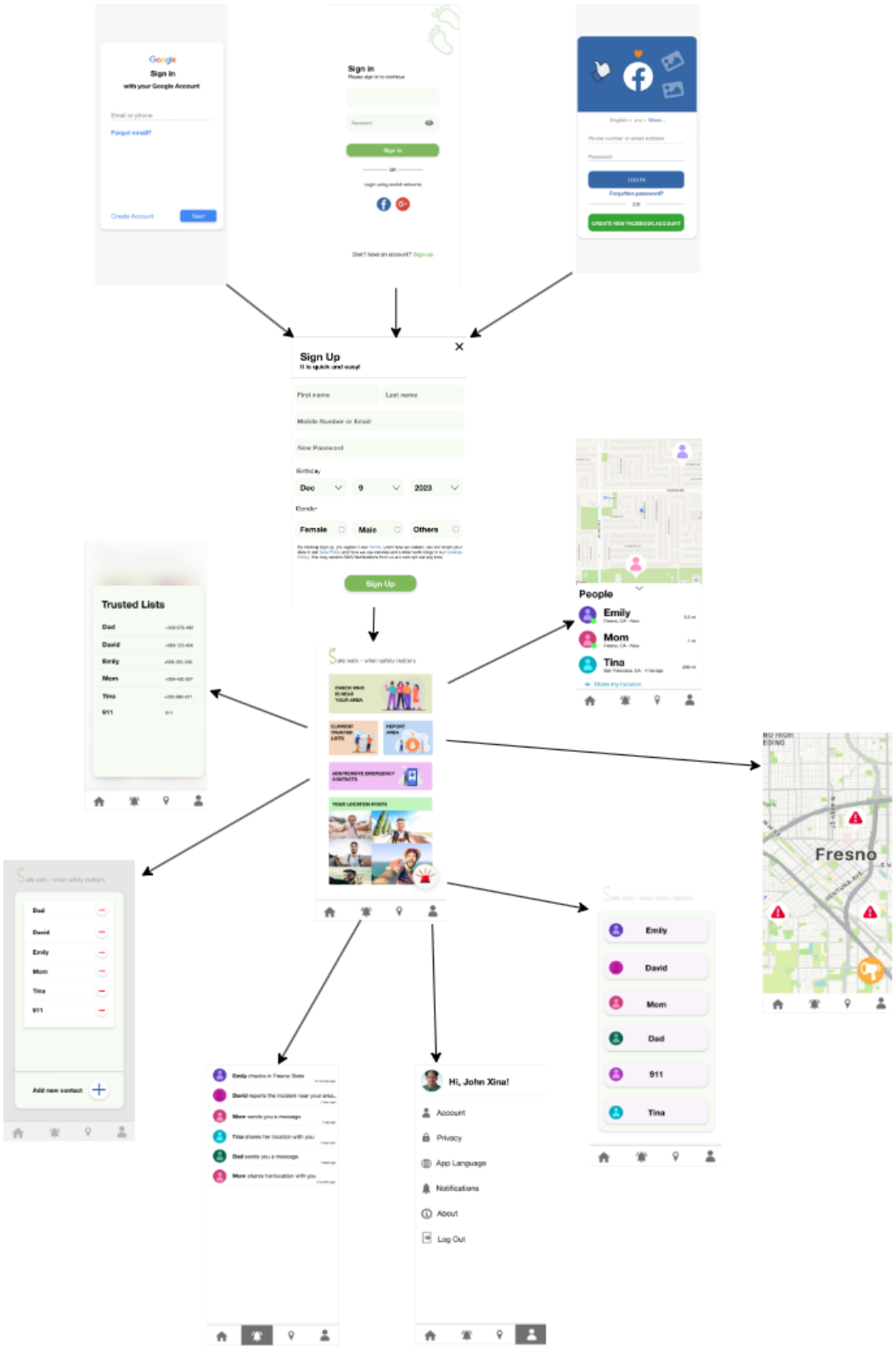


Figure 7.2.2: Wireframe Diagram of basic flow of user interactions. [Link to full UX/UI Design Here.]

7.3 Use Cases / User Function Description

When using Project SafeWalk, the user can expect to utilize the app's functions in a predictive manner. The following use cases will serve as examples.

1. *User Registration and Login:*
    a. **Description:** A new user wants to utilize the features of SafeWalk. They navigate to the app and register with their email or social media account. They then log in to access the app's features.
    b. **Actors:** User, System.
    c. **Flow:** For login, The user will enter their credentials, email, and password and the system will process their credentials on the back-end database and verify if it is an existing user or not. For Sign Up, the user will enter their first and last name, email, username, and password. This will then add them to the database on the backend.
2. *Adding/Removing Emergency Contacts:*
    a. **Description:** A user wants to set up their emergency contacts in the SafeWalk app.
    b. **Actors:** User, System.
    c. **Flow:** The user accesses the settings menu, selects the emergency contacts option, and inputs the contact details. The system validates and saves the contact information for future use.
3. *Sharing Real-Time Location:*
    a. **Description:** A user feels unsafe during their commute and wants to share their real-time location with their emergency contacts. They open the app and initiate the location-sharing feature.
    b. **Actors:** Emergency Contacts, User, System.
    c. **Flow:** The user activates the location-sharing feature, the system retrieves the user's location, and the app sends this information to the saved trusted contacts.
4. *Panic Button:*
    a. **Description:** In a potentially dangerous situation, a user presses the panic button to send an immediate alert to their emergency contacts. The app also notifies local authorities of the user's location.
    b. **Actors:** Emergency Services, User, System.
    c. **Flow:** User activates the panic button, the system sends signals to the emergency contacts, and also alerts local authorities with the user's real-time location for immediate

assistance.

5. *General User Interface Navigation:*
    a. **Description:** A user wants to explore different features within the SafeWalk app.
    b. **Actors:** User, System.
    c. **Flow:** The user interacts with the app interface, navigating through different sections using menu options, buttons, and links to access desired features and information.

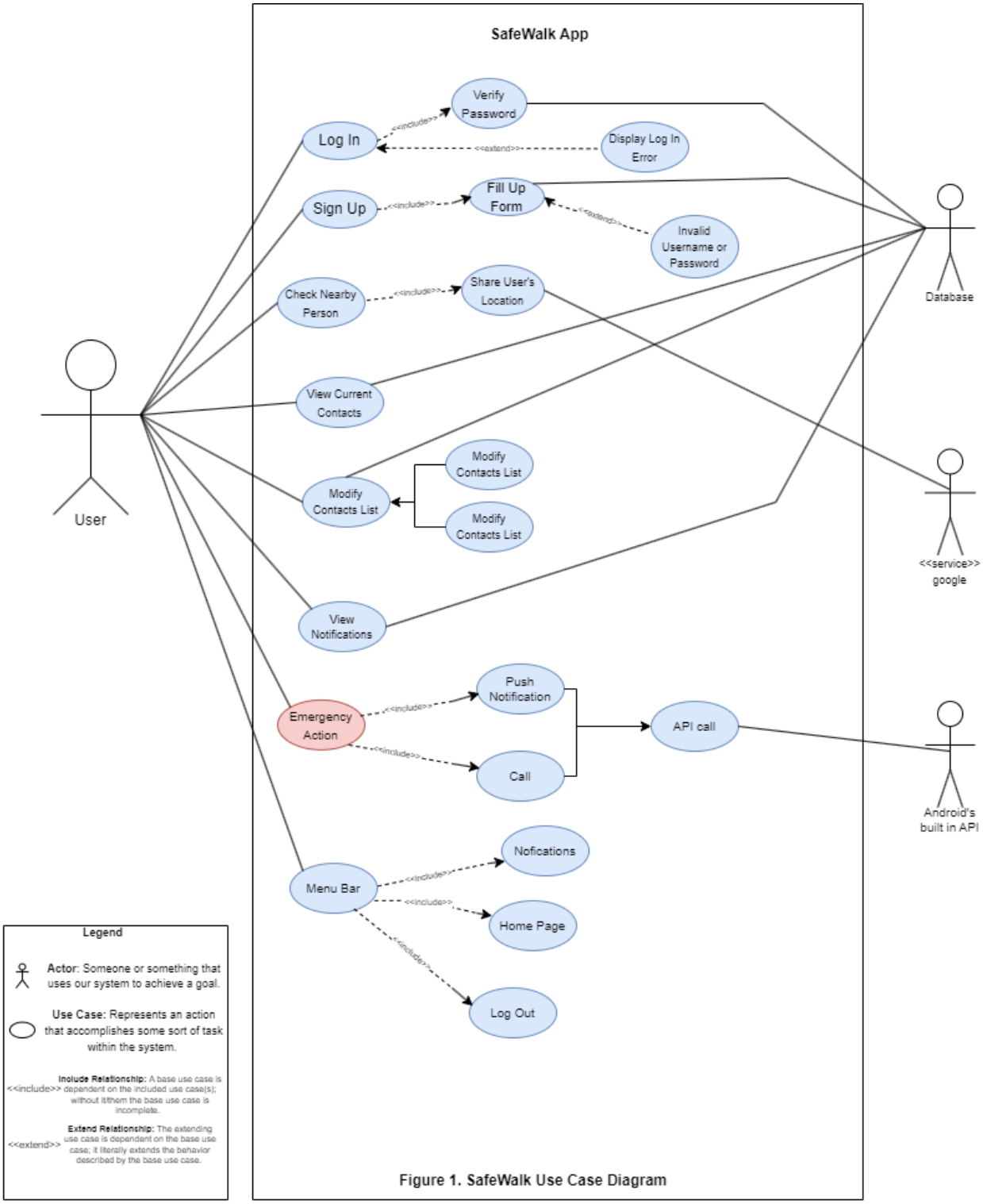


Figure 1. SafeWalk Use Case Diagram

Section 8 - Other Interfaces

External interfaces used in the execution of this module include;
Firebase Authentication API (Firebase SDK) that handles user registration and sign-ins.
Firebase Cloud Firestore Database API (Firebase SDK) used to store and retrieve user data, including contacts and locations.

Push notification will be handled through google service which is Firebase Cloud Messaging.

8.1 Node.js

This is used as the server. We will be using npm(node package module) to use multiple libraries. The libraries will help with interacting with other api's. Express.js will be used for api routes. Firebase admin sdk to communicate with the database.

8.2 Express

Express runs on a node.js server. Routes are made for get and post requests from the client. Each post or get is set in a function where data will be processed for a response to the client.

8.3 Angular/Ionic/Capacitor

This is a frontend framework to help make an application for multiple devices in a small frame of time. Angular gives us a cli to use ionic and capacitor. Using ionic, we have UI components so our app can look the same through different devices. Capacitor works well with ionic and can essentially convert ionic projects into ios or android clients.

8.4 Firebase SDK

A tool used in node.js to communicate with the firebase api. With the api, we can send data to the firestore database and store private information with the user. The firebase sdk can send push notifications to the client side device. The client must send the backend a token from firebase api.

8.5 Amazon Web Services (AWS)

We can use an EC2 server to host the node.js server and ionic/angular client side.

8.6 Google Maps

This is a google service to help with permission handling. We can collect precise location data for users and create custom markers to help the UI look more interactable.

8.7 Google OAuth 2.0

Allows angular to provide the client with a unique token to authenticate when communicating with the backend when requesting or sending data. Token expiration can be set. This will mainly be used for login.

Section 9 - Extra Design Features / Outstanding Issues

9.1 Innovative Features

These are some Innovative features that can be considered for future iterations or development of Project SafeWalk:

Emergency Beacon:

Description: This feature allows users in distress to send out an emergency beacon signal to other SafeWalk users within a certain radius. It would be particularly useful in situations where immediate assistance from nearby individuals could make a crucial difference when authorities are not available.

User Interface: A prominent, easily accessible button in the app would activate the beacon. Users would be informed about the privacy implications and the radius of the signal reach.

Community Watch Integration(s):

Feature Description: SafeWalk could integrate a community watch feature that allows users to report suspicious activities or safety hazards in their area, contributing to a shared safety map.

User Interface: An interactive map with a reporting feature would allow users to view and contribute to community safety data.

Personalized Safety Recommendations:

Feature Description: Using AI, the app could analyze user route data and historical safety metrics to suggest safer routes or times for travel.

User Interface: Users would receive notifications with personalized recommendations and could access a 'Safety Advisor' feature within the app.

9.2 Issues

This is a list of possible issue that may arise for Project SafeWalk:

Privacy Concerns:

Description: The handling of location data and emergency contacts can raise privacy concerns that need to be addressed transparently.

Scalability:

Description: As the user base grows, the infrastructure must be able to scale without performance degradation.

Section 10 – References

Firebase Documentation provides details on how to securely and efficiently use Firebase for backend operation.
Ionic Framework Documentation helps understand the framework's capabilities and limitations.
Node.JS Documentation aids with understanding the backend syntax of the language and libraries used.
AngularJS Documentation
https://github.com/150-lab2/Project_SafeWalk

Section 11 – Glossary

API (Application Programming Interface): A set of rules that allows different software entities to communicate with each other.
DBMS (Database Management System): Software system that uses a standard method to store and organize data.
GPS (Global Positioning System): Satellite-based navigation system.
RTLS (Real-time Location System): Systems that automatically identify and track the location of objects or people in real time.
UML (Unified Modeling System) Diagram: Standardized modeling language enabling developers to specify, visualize, construct, and document the artifacts of software systems.
ER (Entity-Relationship) Diagram: A graphical representation that depicts the entities involved, and the relationships between these entities.
JWT (JSON Web Token): A token that defines a compact and self-contained way for securely transmitting information between applications as a JSON object.