Software Requirement Specification

Student Name: Logesh G (7376222CB128)

Seat Number: 186

Project ID: 26

Project Name: Indoor Location Tracking App

Stack:

Python Stack:

Frontend	HTML, CSS, JS
Backend	PYTHON, DJANGO
Database	MYSQL

Problem Statement:

Bannari Amman Institute of Technology campus consists of many blocks and classrooms. The names of the classrooms are frequently changing. Students, faculty, and parents find it hard to identify and navigate to the desired location. Departments frequently conduct events where the guests and other college students face difficulty in finding the classrooms and blocks. The problem statement is to develop an application that accurately tracks a person on campus and helps them navigate to a particular location.

1. Purpose:

To develop an indoor location-tracking application for students, faculties, and parents with real-time navigation assistance, event integration, etc. The app could track special labs, classrooms, blocks, seminar halls, faculty cabins, and administrative offices. The app provides scalability and reliability in dynamic environments. Indoor location tracking apps give various advantages to students, parents, and faculty, they could save time by quickly finding their desired location path.

2. Scope:

- The indoor navigation app will serve as the location tracker and navigation assistant app for students and faculties. Using this app students can find different special labs, classrooms, and seminar lab locations easily. The app will assist them in navigating to such places with ease.

- This application is also useful to parents, and students visiting BIT campus to navigate to different places inside the campus. The application also updates the location of event occurrence, so that other college students can find location with ease.
- The application consists of an administrator login, where the admin can modify the map and its labels. Admin can add or delete places/events from the BIT campus map.

3. System Overview:

3.1. Users:

1. User:

Students, faculties, guests, and parents come under the user category. They could only access the map and could not modify the map. The user can suggest a change to the administrator. Users can use the navigation and tracking features of the application.

2. Admin:

Admin can change or modify the map labels. Admin can mark the locations for the events where a student has to suggest the change with respective documents.

3.2 Features:

1. Login and registration:

Students and faculties first register the account and login or login with their existing account. Guests and parents do not need a login and can access the map directly. Student and faculty logins enable them to suggest the changes to the administrator.

2. BIT campus map:

The application contains a map of the college campus. The application contains different maps for each floor. The maps change based on the user's desired location. The map contains different locations like classrooms, special labs, seminar halls, event locations, etc. It also contains locations of lifts and staircases.

3. Live location tracking:

Users can track their current live location on the campus. The application uses the information to track and navigate the user to the desired location.

4. Search and Navigate:

Users can search for a particular location on the map using the search option. Navigation assistance from the current/from location to the desired location is also available.

5. Shortest Path:

When the user searches for a location, the application finds the shortest path possible from the current location to the searched location. This feature helps when users try to navigate from one location to another.

6. Suggest for Events:

Users with a login can suggest the changes in the map for an event to the admin with the required documents.

7. Multi-level structure:

The application contains a map for each floor. The user can navigate from floor to floor.

8. Change map details:

This feature is only available for admin login. Admin can change the map labels, and update locations.

3.3 Functional Requirements:

1. **User Management:**
   - Students / Faculty can register and login to the web application.
   - Admin has access to modify the map tags and details.
2. **Live Location Tracking:**
   - Users can track their live location on campus.
3. **Navigation Assistance:**
   - Users can navigate from one location to another with direction assistance.
   - Users can specify the source and destination locations and get live-track navigation assistance.
4. **Update Map:**
   - Admin can access the map details like tags and names. Admin can update the map in event time for easier navigation.

3.4 Non-Functional Requirements:

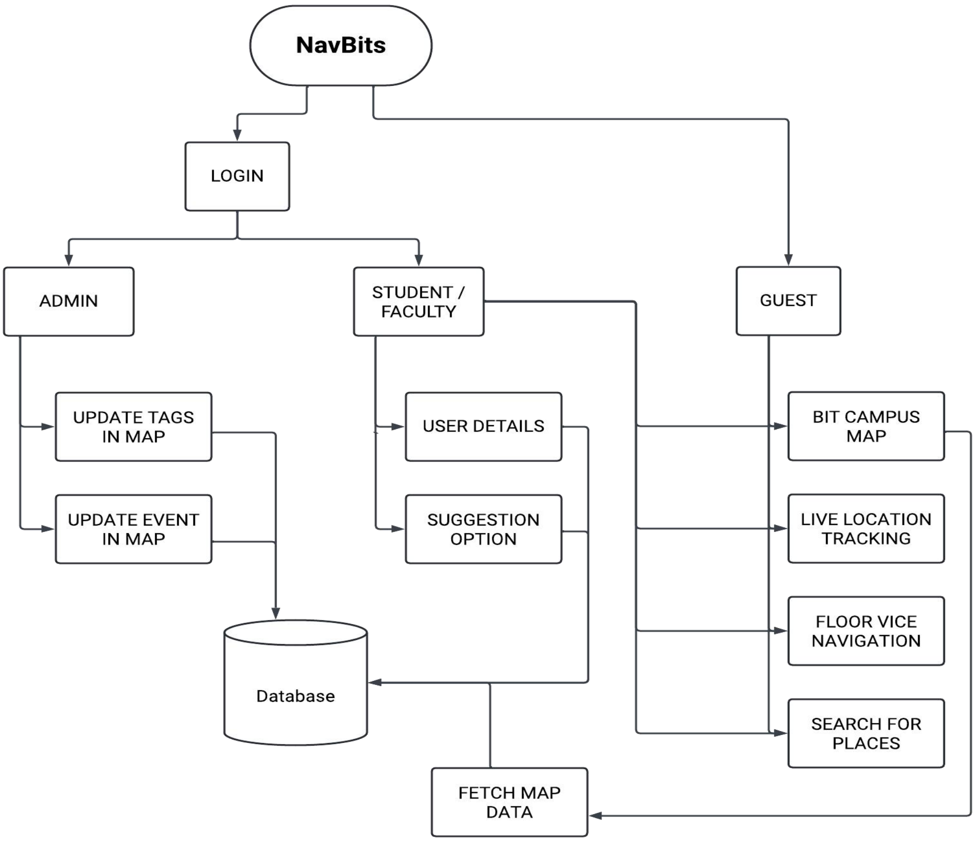
1. Performance: The system must respond to user actions within 2 seconds to ensure efficient usability and must handle a concurrent user load of at least 100 users without significant performance degradation.

2. Usability: The user interface should be intuitive and user-friendly, with clear and concise error messages provided to guide users in case of input errors or system failures.

3. Reliability: The system should be available 24/7 with minimal downtime and should have a backup and recovery mechanism in place to prevent data loss in case of system failures or crashes.

4. Scalability: The system should be designed to accommodate an increasing number of users and data volume over time, and it should be scalable to support additional features and functionalities as per future requirements.

4. Flowchart:



5. Workflow:

1. Project Setup:

1.1. Environment Setup

- Install Django: Set up a Django project using django-admin startproject and create a Django app.
- Front-end Setup: Create the initial structure for HTML, CSS, and JavaScript files.

1.2. Database Setup

- Use Django's ORM to define models for storing map data, GPS coordinates, user data, etc.
- Migrate the models to the database using python manage.py makemigrations and python manage.py migrate.

2. Backend Development:

2.1. Models

- **Map Model**: Store SVG map files and related metadata.
- **Location Model**: Store GPS coordinates and link them with map positions.

2.2. Routing

- Define URL patterns for accessing the indoor maps and user navigation.

3. Frontend Development:

3.1. HTML Structure

- Create the basic layout with sections for the map, user inputs, and navigation buttons.

3.2. CSS Styling

- Style the map, inputs, and navigation components to create a responsive UI.
- Use media queries to ensure the app works on various screen sizes.

3.3. SVG Map Integration

- Embed SVG files within HTML to render maps.
- Use CSS or JavaScript to style and manipulate the SVG map elements (e.g., highlighting paths).

3.4. GPS Data Handling

- Use the Geolocation API in JavaScript to fetch real-time GPS coordinates from the user's device.
- Implement logic to convert GPS data to corresponding positions on the SVG map.

3.5. Map Interaction

- Implement panning and zooming on the SVG map for better navigation.
- Allow users to select destinations on the map and calculate the shortest path using Dijkstra's algorithm or A*.

3.6. Pathfinding and Navigation

- Use JavaScript to calculate paths based on SVG coordinates and GPS data.
- Render the calculated path on the map using lines or highlighted areas.
- Provide real-time updates as the user moves.

3.7. Map Handling

- Use the SVG map on the leaflet.js map for better location understanding.
- Each floor in the location has its own SVG maps and map handling is done accordingly.

4. Testing and Deployment:

4.1. Testing

- Test each layer individually (unit testing) and then together (integration testing).
- Test GPS accuracy, map rendering, and user interactions.

5. Maintenance and Updates:

- Regularly update map data and improve the GPS-to-map conversion logic.
- Gather user feedback for continuous improvement.