Software Requirement Specification

Student Name: THARUN D V (7376221CS334)

Seat Number: 266

Project ID: 26

Project Name: Indoor Location Tracking App

Stack:

Python Stack:

Frontend	VUE JS
Backend	NODE JS EXPRESS JS
Database	MONGODB

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 foster an indoor area following application for understudies, resources, and guardians with constant route help, occasion reconciliation, and so on. The application could follow unique labs, homerooms, blocks, class lobbies, personnel lodges, and authoritative workplaces. The application gives adaptability and unwavering quality in unique conditions. Indoor area following applications give different benefits to understudies, guardians, and workforce, they could save time by rapidly tracking down their ideal area.

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:**
    - This category includes students, faculty, guests, and parents. Users can access the map but cannot modify it. They can suggest changes to the administrator and utilize the app's navigation and tracking features.
2. **Admin:**
    - The admin has the ability to modify map labels and mark event locations. If students need changes made, they must submit their suggestions along with the necessary documentation.

3.2 Features:

1. **Login and Registration:**
    - Students and faculty members must create an account and log in, or they can log in using an existing account. Guests and parents can access the map without logging in. Logging in allows students and faculty to suggest changes to the administrator.
2. **BIT Campus Map:**
    - The app includes a detailed map of the college campus, with separate maps for each floor. The map adjusts based on the user's selected location and shows various areas such as classrooms, special labs, seminar halls, event locations, lifts, and staircases.
3. **Live Location Tracking:**

- Users can view and track their current location on campus. The app uses this information to help guide them to their desired destination.

4. **Search and Navigate:**
    - Users can search for specific locations on the map. The app also provides navigation assistance from their current location to the desired destination.
5. **Shortest Path:**
    - When a user searches for a location, the app determines the shortest route from their current position to the selected destination. This feature is especially useful when navigating between different locations.
6. **Suggest for Events:**
    - Logged-in users can propose changes to the map for specific events by submitting the required documentation to the admin.
7. **Multi-level Structure:**
    - The app includes maps for each floor, allowing users to navigate between floors.
8. **Change Map Details:**
    - This feature is exclusive to admins, who can update map labels and locations.

3.3 Functional Requirements:

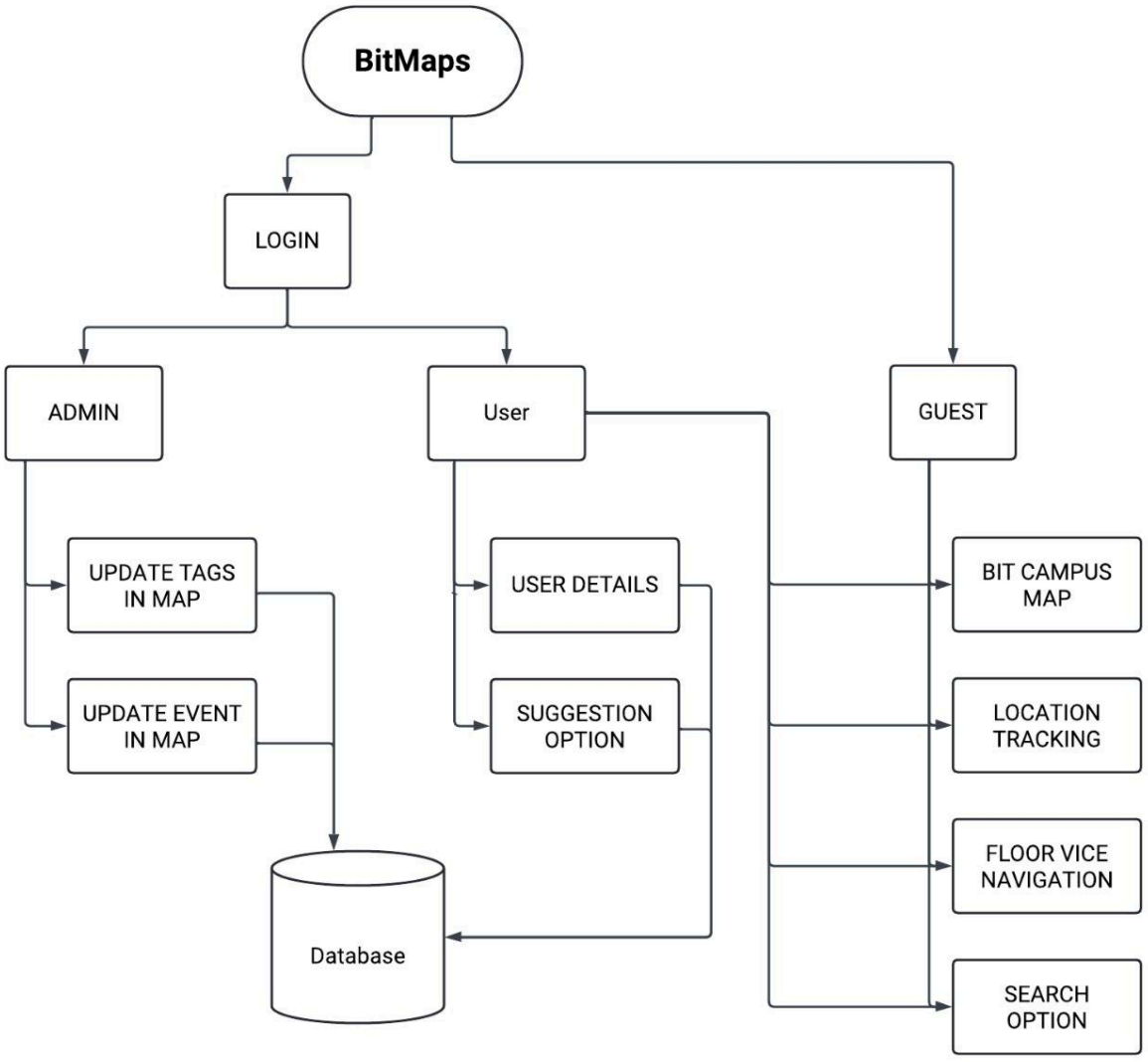
1. **User Management:**
    - Students and faculty can register and log in to the web application. Admins have the ability to modify map tags and details.
2. **Live Location Tracking:**
    - Users can monitor their live location on campus.
3. **Navigation Assistance:**
    - The app provides direction assistance, enabling users to navigate between locations. Users can specify the source and destination to receive live navigation guidance.
4. **Update Map:**
    - Admins have access to map details, including tags and names, and can update the map during events to facilitate easier navigation.

3.4 Non-Functional Requirements:

1. **Performance:**
    - The system should respond to user actions within 2 seconds to ensure efficient usability and handle a concurrent load of at least 100 users without significant performance drops.
2. **Usability:**
    - The user interface should be intuitive and user-friendly, with clear and concise error messages to guide users in case of input errors or system issues.
3. **Reliability:**
    - The system should be available 24/7 with minimal downtime and include a backup and recovery mechanism to prevent data loss in case of failures.

4. **Scalability:**
    - The system should be designed to accommodate a growing number of users and data over time, and it should be scalable to support additional features and functionalities as needed in the future.

4. Flowchart:



WORKFLOW :

1. Project Planning and Requirements Gathering

- **Define Project Scope:**
  - Identify the primary goals and objectives.
  - Specify the target users (students, faculty, guests, parents, admins).
- **Requirement Analysis:**
  - Gather detailed functional and non-functional requirements.
  - Conduct interviews with stakeholders (students, faculty, IT staff).
  - Document user stories and use cases.
- **Technology Stack Selection:**
  - Choose the platforms (e.g., iOS, Android, web).
  - Select technologies for frontend, backend, and database.
  - Consider integration with existing campus systems.

2. Design Phase

- **System Architecture Design:**
  - Design the overall architecture (client-server model, database structure).
  - Define the components (e.g., user management, map module, tracking engine).
  - Plan for scalability and reliability.
- **UI/UX Design:**
  - Create wireframes and mockups for different user interfaces (student, faculty, admin).
  - Design intuitive navigation and search functionalities.
  - Design the map interface for floor navigation and event location tracking.
- **Database Design:**
  - Create an ERD (Entity Relationship Diagram) for user data, map data, and event details.
  - Plan for data storage and retrieval mechanisms.

3. Development Phase

- **Frontend Development:**
  - Implement UI components for login, registration, map view, and navigation.
  - Develop responsive interfaces for different devices (mobile, tablet, desktop).
- **Backend Development:**
  - Develop the server-side logic for user management, map updates, and live location tracking.
  - Implement API endpoints for data exchange between frontend and backend.
  - Integrate third-party services if required (e.g., GPS, campus databases).
- **Database Implementation:**
  - Set up the database according to the design.
  - Implement data models for users, maps, events, and locations.
  - Ensure secure data handling and storage.
- **Admin Portal Development:**
  - Develop features for admins to modify map labels, add/delete locations, and manage events.
  - Implement security measures for admin access.

4. Testing Phase

- **Unit Testing:**
  - Test individual components (e.g., user login, map navigation, tracking features).
- **Integration Testing:**
  - Ensure that all components work together seamlessly (frontend, backend, database).
  - Test data flow between user interfaces and the backend.
- **User Acceptance Testing (UAT):**
  - Conduct testing sessions with real users (students, faculty, admins).
  - Gather feedback on usability, functionality, and performance.
- **Performance Testing:**
  - Test the system's response time and load capacity (e.g., 100 concurrent users).

    - Optimize the app for faster performance and scalability.

5. Deployment Phase

- **Deployment Preparation:**
    - Set up production environments (servers, databases).
    - Configure security protocols (SSL, encryption).
- **Go-Live:**
    - Deploy the application to the production environment.
    - Monitor the system for any issues during the initial launch.
- **Post-Deployment Support:**
    - Provide immediate support for any post-launch bugs or issues.
    - Monitor system performance and user feedback.

6. Maintenance and Updates

- **Regular Maintenance:**
    - Monitor system uptime and performance.
    - Perform routine updates and patches.
- **User Feedback Integration:**
    - Collect ongoing feedback from users and admins.
    - Plan and implement updates and new features based on user needs.
- **Scalability Planning:**
    - Prepare the system for future growth in user base and data.
    - Plan for new features and enhancements based on evolving requirements.