

**Team members:**

- 1.Jayanta Das-22235103221**
- 2.Mohammad Mahmudul Hasan Rodra-22235103211**
- 3.Ayesha Siddika Mohona-22235103206**
- 4.Md Nazmul Hasan Siam-22235103223**
- 5.Akib Hasan Niloy-22235103649**

# **Software Requirements Specification(SRS)**

## **1. Introduction:**

### **1.1 Purpose:**

The purpose of this project is to design and develop an efficient, user-friendly SOS mobile application that enables users to:

- Instantly alert trusted contacts during emergencies.
- Share real-time location and custom alert messages.
- Operate the app with minimal effort (one-tap or shake-to-alert).
- Help users feel safer in daily life by having emergency support at their fingertips.

### **1.2 Intended Audience:**

1. **SecureYou** mobile application is designed for a wide range of users who may require personal safety support in emergency situations. Intended audience:
2. **General Users:** Individuals seeking a reliable, user-friendly safety application for daily life emergencies.
3. **Women and Vulnerable Groups:** Users at higher risk of harassment or unsafe situations, requiring discreet and rapid alert options.
4. **Elderly and Differently-Abled Persons:** Individuals with limited mobility or health concerns needing quick and accessible emergency assistance.
5. **Emergency Contacts:** Trusted family members, friends, or guardians who will receive alerts, calls, and real-time location updates.
6. **Law Enforcement and Emergency Services:** Authorities and medical responders who may be integrated into the system for efficient response.
7. **Developers, Testers, and Administrators:** Stakeholders responsible for maintaining the system's reliability, security, and continuous improvement.

8. **Community Safety Organizations & NGOs:** Groups working in the domain of public safety, women empowerment, and social welfare that may recommend or integrate the app into their safety programs.
9. **Developers, Testers, and Administrators:** Stakeholders responsible for maintaining the system's reliability, security, and continuous improvement.
10. **Policy Makers & Institutions:** Government agencies, universities, and organizations that can adopt or promote the app to enhance overall community safety.

### **1.3 Intended Use:**

1. **SecureYou** mobile application is intended to provide rapid, reliable, and discreet safety solutions during emergency situations. Intended use:
2. **Emergency Alerting:** Allow users to instantly notify trusted contacts and emergency services through one-tap buttons, SMS, and calls.
3. **Real-Time Location Sharing:** Provide continuous GPS-based location tracking so responders and contacts can accurately reach the user.
4. **Discreet Safety Mode:** Offer silent and camouflaged alerts in scenarios where openly calling for help may put the user at greater risk.
5. **User Profile & Contact Management:** Enable users to configure emergency contacts, personal details, and medical history for quick identification and faster rescue.
6. **Integration with Emergency Services:** Ensure direct communication with law enforcement, healthcare providers, and security agencies for immediate action.
7. **Health & Medical Emergencies:** Assist during medical crises by sharing critical health data (e.g., blood group, allergies, conditions) with responders.
8. **Accident & Travel Safety:** Provide quick alert options for road accidents, travel-related incidents, or unsafe public transport situations.
9. **Natural Disaster Assistance:** Enable users to send SOS alerts and location details during floods, earthquakes, fires, or other large-scale emergencies.
10. **User Accessibility:** Ensure a simple and intuitive interface suitable for elderly, differently-abled persons, and non-tech-savvy individuals.
11. **Community Safety Support:** Encourage use in schools, universities, workplaces, and communities to create a safer environment collectively.
12. **Testing & Reliability:** Validate the app under artificial and real emergency simulations to guarantee accuracy, speed, and dependability.
13. **Data Security & Privacy:** Protect sensitive user data, ensuring communication and location details are securely transmitted and stored.

### **1.4 Product Scope:**

SecureYou mobile application aims at increasing personal safety through speed and trustworthiness during the emergency. Product scope:

- User Interface (UI): It has to be well designed, in a way that is user friendlier and easy to use especially in moments of urgency.
- User Profile Management: Available features to enable users to create and manage profiles such as establishing trusted emergency contacts.
- Emergency Features:
  - One button emergency call.
  - Real-time GPS-tracking to share the location of the user.
  - Incorporation of SMS and call to alert contact and emergency facilities.
- Discreet Safety Modes: Silent and camouflaged alerts in cases where the user is not able to ask help freely.
- Testing: Thorough testing under artificial emergency conditions to substantiate rapidity, reliability and precision.
- Technology Stack: Technologies to be used to develop the application include Java and backend services in providing location services and messaging API to build a seamless communication process.

## **1.5 Risk Definitions:**

### **1. Offline Map API Integration Risk**

- *Definition:* The application can experience technical challenges when organizing offline maps APIs, and hence loss of functionality in locations without network.
- *Impact:* Users are not likely to receive correct and real-time location updates during emergencies.

### **2. Government Permission Risk**

- *Definition:* The application may demand approval or conformity to government regulation (e.g. emergency SMS routing, telecom permissions, or data-sharing policies). The approval may cause delays or rejection that might cause a negative effect on the launch and usability.

- *Impact:* May not deploy or limit some features such as SOS calling,/SMS services.

### **3. Android Version Compatibility Risk**

- *Definition:* Not every Android version can support the applications functions (GPS tracking, background SMS/call integration, permissions handling).
- *Impact:* Older devices can crash, lack rich functionality or be totally unsupported.

## **2. Overall Description:**

### **2.1 User Classes and Characteristics:**

The Secure You application will serve multiple user groups, each with distinct needs and technical expertise. Understanding their characteristics is essential for designing intuitive interfaces, reliable backend systems, and secure communication channels.

- Primary Users (End Users / General Public):
  - Individuals using the app for personal safety.
  - Limited technical knowledge; require a simple, fast, and intuitive UI.
  - Must be able to quickly trigger SOS alerts, receive incident notifications, and share experiences.
- Emergency Responders (Authorities / Rescue Teams):
  - Police, ambulance, fire departments, or private security services integrated with the system.
  - Require accurate, real-time incident reports and continuous GPS tracking.
  - Need quick access to user details (e.g., medical information, phone number) to provide effective assistance.
- Trusted Contacts (Friends/Family):
  - Selected by the user to receive notifications during emergencies.
  - Expect instant alerts with live location updates and incident details.
  - Not technically skilled but require clear, actionable information.
- Administrators (Backend Operators):
  - System moderators and technical staff responsible for managing the platform.
  - Skilled in database and user management.
  - Require tools for monitoring incidents, verifying stories, analyzing reports, and ensuring smooth system performance.

## **2.2 User Needs:**

Users want an easy and quick way to ask for help in emergencies. The app should allow them to send alerts, share their live location, and reach out to trusted people or emergency services. It should be simple to use in stressful situations while keeping their data safe and private.

### **1. Quick Access to Emergency Features**

- Users need to be able to send an emergency signal quickly when they're in trouble.
- They also need to share their location and send messages to their contacts or emergency services with just a button press.

### **2. Reliable Communication**

- The alerts should reach trusted contacts (like family or friends) and emergency services without any delays or mistakes, especially if the user can't speak.

### **3. Privacy and Security**

- Users need to know that their personal data, like their location or contacts, is kept safe and private while using the app.
- The app must also follow privacy rules to protect this information.

### **4. Location Sharing and Tracking**

- Users want their location to be updated and shared in real-time, so help can find them quickly.

### **5. Simple to Use**

- The app should be easy to navigate, even when the user is in a panic.
- The emergency features should require just a few simple steps to use.

### **6. Discreet Alerts**

- Sometimes, users might need to send an emergency alert without others noticing (like in dangerous situations), so the app should offer a discreet alert mode.

### **7. Always Available Support**

- Users should always have access to emergency features, whether it's live location updates, SMS alerts, or calling emergency services.

## **2.3 Operating Environment:**

The Secure You application will operate within mobile and cloud environments, relying on real-time data exchange and robust security standards. It must function under varying connectivity conditions while maintaining reliability and performance.

- Mobile Platform:
  - Supported on Android (version 8.0 and above) and iOS (version 13 and above).
  - Requires smartphones with GPS, mobile data, and push notification capabilities.
- Backend / Server Environment:
  - Cloud-hosted (AWS, Google Cloud, or Azure) for scalability and high availability.
  - REST APIs used for communication between mobile client and server.
  - Database system (SQL/NoSQL) to store user profiles, incident reports, and shared stories.
- Network Environment:
  - Requires stable internet (4G/5G/Wi-Fi).
  - Includes fallback mechanisms in case of poor connectivity (e.g., retry logic, optional SMS-based SOS trigger).
- Security Environment:
  - Encrypted communication via HTTPS/TLS.
  - Secure storage of sensitive user data (medical records, emergency contacts, GPS).
  - Role-based access control to separate permissions for users, responders, and administrators.
- Physical Environment:
  - Designed for both indoor and outdoor usage.
  - Must operate effectively under variable conditions such as urban areas, rural regions, crowded environments, and low-light scenarios.

## **2.4 Constraints:**

### **1. Operating System Compatibility**

- The application will work across Android 8.0 (Oreo) and greater and iOS 13.0 and greater. Older releases of the operating system will not be supported because of limitations in permission handling, security, and background activity needed to support features such as GPS and SOS notifications.

### **2. Government & Regulatory Approval**

- Features like SOS calling, SMS alerting and GPS tracking may need government approvals or telecom compliance, which may restrict such implementations, or delay implementation.

### **3. API Dependency**

- The application uses third-party applications such as GPS, SMS/call, and off line maps. Any variation in the terms and prices charged by the API or alteration in the availability of the same can have a negative effect on performance.

### **4. Offline Map Limitation**

- Though the application is designed to work using offline maps, full live location sharing and SOS tracking might need intermittent internet connectivity to be more accurate and get synced.

### **5. Security and Privacy Compliance**

- The app should be in compliance with data protection regulations , which constraints how much data can be stored and how sensitive user data has to be encrypted.

## **2.5 Assumptions:**

We assume that users will have access to data or the internet, that they will have set up trusted contacts, and that they'll use Android devices with location services enabled. We also expect that the app will comply with necessary regulations for emergency service integration.

### **1. Mobile Data or Internet**

- The app assumes users will have internet or mobile data for features like real-time location sharing and notifications. However, some features (like offline maps) may work without an internet connection.

### **2. Trusted Contacts Set Up**

- We assume that users will have selected emergency contacts in advance, so these contacts can be notified in case of an emergency.

### **3. Device Compatibility**

- The app assumes the user's device is running an Android version higher than 8.0 and supports the necessary features for tracking location and making calls.

### **4. Location Services Enabled**

- The app assumes the user has turned on location services, so location-based features work correctly.

### **5. Government and Telecom Approvals**

- We assume that the app will follow all required regulations and policies related to emergency services, SMS, and calling systems

### **3. Requirements:**

#### **3.1 Functional Requirements:**

##### **1.Account Creation:**

###### **Front-End:**

As a user, I need to register in the app by providing my name and phone number.

After registration, the app will prompt me to allow location access and select an emergency contact from a dropdown list.

###### **Back-End:**

- The user will register by submitting their name and phone number.
- The system will request location access and store the user's location.
- The system will display a dropdown with options (mom, dad, best friend) to select an emergency contact number.
- The selected emergency contact number will be saved in the user's profile.

##### **Confirmation:**

###### **1.Success:**

- Registration completed successfully.
- Location access granted and saved.
- Emergency contact selected and saved.

###### **2.Failure:**

- Registration failed – please check the details and try again.
- Location access denied – please enable location services.
- Failed to save emergency contact – please try again.

##### **2.Login:**

###### **Front-End:**

As a user, I want to log in with my registered email/username and password, so that I can securely access my personal account and app features.

### **Back-End:**

- Given the user enters valid credentials, when they click "Login", then the system must verify the credentials against stored data.
- Passwords must be encrypted and matched securely.
- If the credentials are valid, then the system must create a session/token and grant access.
- If the credentials are invalid, then the system must reject the login attempt with an appropriate error message.
- After successful login, the system must log the activity with timestamp and device info.

### **Confirmation:**

1. **Success –**
  - The user is redirected to the dashboard/home page.
  - Session/token is created and stored securely.
  - Login activity is logged.
2. **Failure –**
  - “Invalid email or password” error displayed.
  - “Account locked due to multiple failed attempts” message shown.
  - “Server error, please try again later” message shown.

### **3. Access Permission:**

#### **Front-End:**

As a user, I want the app to request and manage permissions for location, SMS, and call access, so that the system can function correctly while respecting my privacy.

#### **Back-End:**

If the app requires access permissions then the system must:

1. Prompt the user to grant permissions for:
  - a. Location access
  - b. SMS sending
  - c. Call access

2. Verify whether each permission is granted before performing actions that require it.
3. Handle scenarios where permissions are denied by providing appropriate warnings or alternative actions.

### **Confirmation:**

#### **1. Success –**

- Granting permissions location, SMS, and call access verified.
- User notified by “All required permissions granted. Emergency features fully enabled.”
- The dashboard updates automatically to reflect the enabled emergency features.

#### **2. Failure –**

- Location access denied.
- Unable to notify contacts via SMS.
- Unable to contact emergency services via call.

## **4.Admin Dashboard:**

### **Front-End:**

As an admin, I want to access a dashboard where I can view and manage users, monitor system activities, and generate reports, so that I can maintain and oversee the platform effectively.

### **Back-End:**

- Given the admin is authenticated, when they log in, then the system must provide access to the dashboard.
- The dashboard must fetch and display user data, activity logs, and system status in real time.
- The system must allow the admin to manage user accounts (add, update, deactivate).
- When the admin applies filters (e.g., by date, user role, or status), then the system must update and show the correct information.
- When the admin generates reports, then the system must prepare downloadable/exportable files ( PDF/Excel).

- All sensitive operations must be logged with admin ID and timestamp for audit purposes.

#### **Confirmation:**

##### **1. Success -**

- Admin successfully logs in and sees the dashboard.
- Admin can view, filter, and manage user accounts.
- Reports are generated and downloadable.
- Activity logs are updated instantly.

##### **2. Failure -**

- “Unauthorized access” message shown if non-admin attempts access.
- “Unable to fetch data, please refresh” error displayed.
- “Report generation failed, try again later” message shown.
- System logs the failed attempt for audit.

#### **5. User Dashboard:**

##### **Front-End:**

As a user, I want to see my username, live location, and a big emergency message send button on the home dashboard. The dashboard will also include a hamburger menu for other navigation options.

##### **Back-End:**

- The system will display the user's username and live location on the home screen.
- A large emergency button will be visible to the user.
- When the button is pressed, a 10-second timer window will appear. If the user does not cancel within the time, the system will send an emergency signal to the user's set emergency contact and the national emergency number.
- If the user cancels within the 10 seconds, the system will revert to the normal State.

#### **Confirmation:**

##### **Success:**

- Emergency signal successfully sent to emergency contacts and national emergency number.
- Emergency alert received by the contacts.

##### **Failure:**

- Failed to send an emergency signal – please try again.

- Timer expired – emergency signal sent automatically.
- No emergency contact set – alert not sent.

## **6.Emergency Button:**

### **Front-End:**

As a user, I press an emergency button to send my live location and a message to my trusted contacts, so they can help me immediately.

### **Back-End:**

- When the emergency button is pressed, the system must:
  1. Capture the user's current live location immediately.
  2. Generate an emergency alert package (location, time, user ID, custom message).
  3. Dispatch the alert via multiple channels:
    - Push notification to in-app trusted contacts.
    - Background trigger for SMS/Call Integration module (to notify 999 + trusted contacts).
  4. Log the event in the system for audit and tracking.
  5. Ensure data encryption in transit and restrict delivery only to authorized contacts.

### **Confirmation:**

- **Success:**
  - Emergency alert package successfully generated.
  - Real-time push notification sent to trusted contacts.
  - Emergency SMS/Call integration triggered.
- **Failure:**
  - Location access denied → “*Please enable location services.*”
  - Failed to generate or send alerts.
  - Unauthorized contacts blocked.

## **7.SMS/call integration:**

### **Front-End:**

As a user, I want the system to send an SMS or make a call automatically when I trigger an emergency alert, so that I can ensure my trusted contacts and emergency services are notified immediately even if they are not using the app.

### **Back-End:**

If the user has triggered an emergency alert then the system must:

1. Send an SMS containing the user's live location and emergency message to:
  - a. The emergency service 999
  - b. All authorized trusted contacts of the user
2. Ensure that all SMS and call data are transmitted securely to protect privacy.
3. Verify that each contact is authorized before sending the SMS or making the call.

### **Confirmation:**

#### **1. Success –**

- SMS successfully sent to 999
- SMS successfully sent to emergency contacts
- Calls successfully connected to 999 and trusted contacts.

#### **2. Failure –**

- SMS sending failed
- Unable to connect to 999 or trusted contacts.
- SMS/call sent only to authorized recipients.

## **8.Location Tracking:**

### **Front-End:**

As a user, I want to share my live location with trusted contacts, so they can track me in real time if needed

### **Back-End:**

- Given the user has enabled location sharing. When the user moves, Then the system must update their location in the backend every 10 seconds.
- The data must be encrypted in transit
- When a trusted contact requests the location, then the system must verify that the requester is authorized before sharing.
- When the location updates in the backend, then the system must push the update to the contact in near real-time

## **Confirmation:**

### **1. Success –**

- Trusted contacts see refreshed real time location.
- Authorized contact requests location access granted.
- Trusted contacts receive updates instantly.
- Location access entry created.

### **2. Failure –**

- Unable to update location, please try again.
- Secure channel required, update rejected.
- Unauthorized contact requests Access denied.
- Update delayed, please refresh.
- Location access blocked, try later.

## **9. Share Incident Stories:**

### **Frontend:**

- As a user, I want to post an incident story I experienced or witnessed so that others can be informed and learn from it.
- As a user, I want to like, comment, or save incident stories so that I can interact with and reference them later.

### **Backend:**

#### **Post Incident Story:**

- Given the user submits a new incident story.
- Then the system must create a new story record with:
  - User details (name, profile, optional contact info)
  - Location (optional GPS coordinates)
  - Story content (text, images, videos)
  - Timestamp of the story
  - Category/Tags (optional, e.g., accident, fire, crime)

**Confirmation:****Success:**

1. User notified that their story is posted successfully
2. Other users in the area can see and interact with the story

**Failure:**

1. Story submission fails → retry posting
2. Unauthorized submission → denied

**10.Recent Incident:****Frontend:**

- As a user, I want to see around my location if anything happened to anyone recently so that I can avoid or prepare myself.
- As a user, I want to receive a notification + live location when my someone triggers SOS

**Backend:**

## Trigger SOS:

- Given the user presses the SOS button.
- Then the system must create a new incident record with:
  - User details (name, phone, medical info, etc.)
  - Location (GPS coordinates, updated every 10s).
  - Timestamp of incident and share it to the app

**Confirmation****Success:**

1. User notify with location and details
2. Trusted contacts see a refreshed location.

**Failure:**

1. User update fails → retry posting.
2. Unauthorized case update request → denied.

**3.2 Non Functional Requirements:****1. Performance**

- The system must send an emergency SMS/Call within 3 seconds after the emergency button is triggered.
- Location updates must refresh at least every 10 seconds with minimal delay.
- The dashboard should load within 2 seconds under normal network conditions.

## **2. Reliability & Availability**

- The system must be available 99.5% of the time to ensure emergency services are always functional.
- Emergency alerts must have at least 95% delivery success rate to trusted contacts and 999.
- Automatic retry mechanism must be attempted at least 3 times if SMS/call fails.

## **3. Security & Privacy**

- Sensitive data like location, contacts, incidents, and login details must be encrypted during transfer and storage.
- User passwords must be stored in a secure, encrypted form.
- Only authorized contacts can receive location and emergency alerts.
- Admin activities must be logged with timestamp, admin ID, and action type for audit purposes.

## **4. Usability & Accessibility**

- The emergency button must be clearly visible and accessible from the main dashboard.
- The app interface must be simple enough for a new user to trigger an SOS in 1 click/tap.

## **5. Scalability**

- The system must support at least 5,000 concurrent users without performance degradation.
- Incident stories and recent incidents must scale to store at least 10,000 records.

## **6. Interoperability**

- The app must work seamlessly on Android and iOS devices.
- SMS/Call services must work across different telecom operators.

- Location services must integrate with Google Maps.

## 7. Maintainability

- The codebase must follow modular design so new features can be added without affecting existing ones.
- Admin dashboard logs must be downloadable for problem fixing and review.
- System errors must generate error logs with clear messages for debugging.

## 8. Compliance

- Follow telecom and data protection regulations.

## 9. Fault Tolerance & Recovery

- If SMS sending fails, the system must attempt an alternative route like internet push notification or call retry.
- If the server crashes, it must recover from the last stable state within 1 minute.
- Offline fallback: if no internet, SMS should still go via cellular network.

## 10. Auditability & Logging

- All emergency events must be logged with timestamp, user ID, location, and delivery status.
- Admin dashboard activities must be stored for at least 1 year.
- Users must be able to see their recent emergency history in their profile.