Software Requirements Specification

Project Name: Incident Notification and Reporting App (Suraksha)

Version: 1.0

Date: 2023-09-04

Team Members:

1. Rohan Kumar Shetty
2. Sachin R Jangamashetti
3. Kethavath Sai Charan
4. Karri Sri Satya Venkata Kishore
5. Segu Shanmuka Srinivas

Table of Contents

1. Objective

The objective of this project is to develop a web app that allows employees and contractors within Shell to report safety incidents, near-misses, and hazards quickly and efficiently in real-time. The app would enable seamless communication and coordination among personnel, enabling timely responses and actions to mitigate risks and improve overall safety.

2. Scope

The app will cover the following functionality:

- User login and registration
- Incident reporting
- Media attachment
- Notification of designated safety officers and supervisors
- Real-time chat between safety officers and users involved in the incident.
- Incident tracking and management
- Report generation and analytics
- Notification escalation
- Integration with Shell's existing safety management systems

The app will be developed using the following technologies:

- Frontend: React
- Backend: .NET Framework 4.8
- Database: PostgreSQL

3. Functional Requirements

The app must meet the following functional requirements:

- Allow users to log in and register for the app.
- Allow users to report incidents, including the type of incident, location, severity, urgency, and any other relevant details.
- Allow users to attach media to incident reports, such as photos, videos, or audio recordings.
- Notify designated safety officers and supervisors about reported incidents.
- Enable real-time chat between safety officers and users involved in the incident.
- Allow safety officers to track and manage reported incidents.
- Generate reports and analytics to identify common incident types, trends, and patterns.
- Escalate notifications to higher-level personnel if an incident is not acknowledged within a set timeframe.
- Integrate with Shell's existing safety management systems.

4. Non-Functional Requirements

The app must meet the following non-functional requirements:

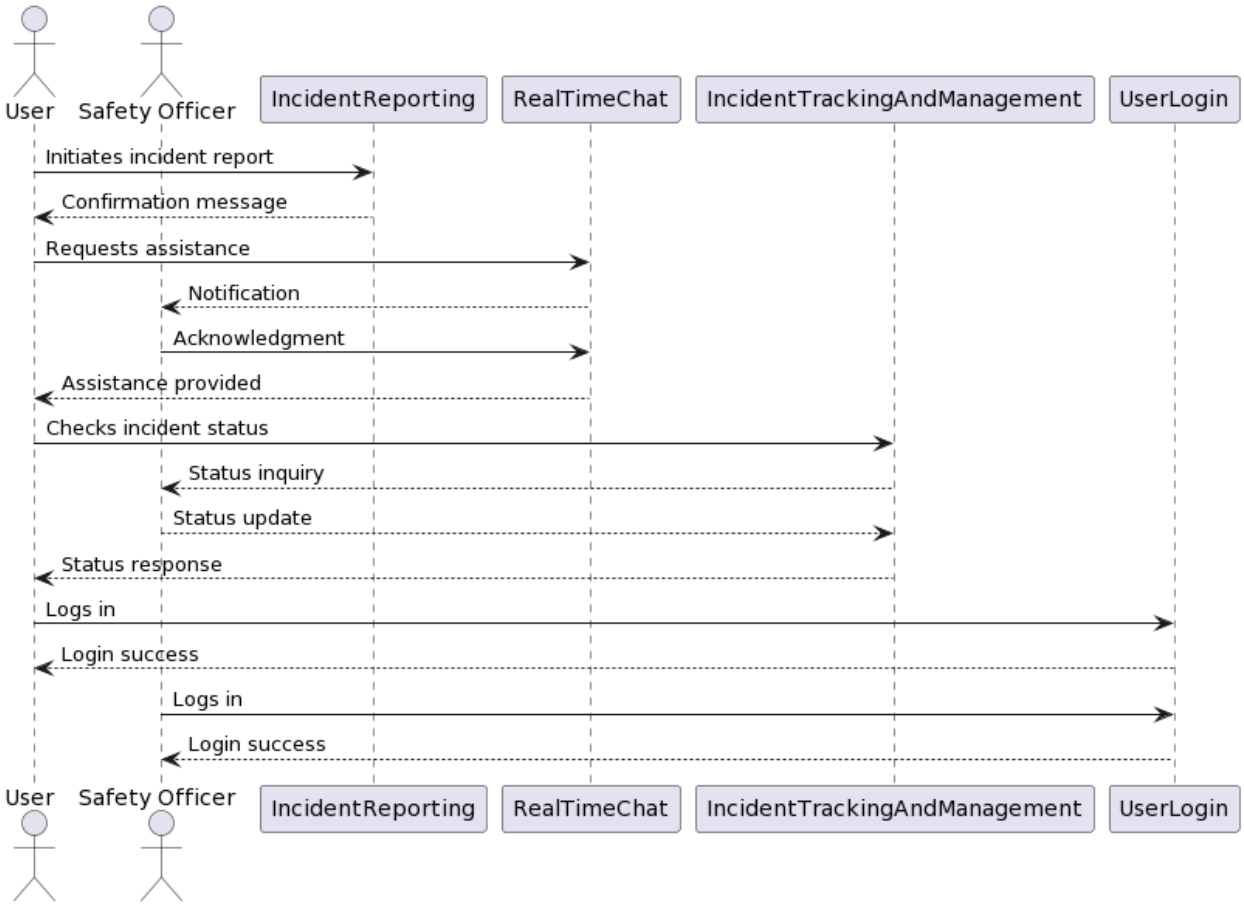
- Security: The app must be secure and protect user data.
- Usability: The app must be easy to use and navigate.
- Scalability: The app must be scalable to handle many users and reports.
- Availability: The app must be available 24/7.
- Backup and recovery: The app must have a backup plan in place to prevent data loss.

5. Users

The following users will interact with the app:

- Employees and contractors within Shell
- Safety officers and supervisors
- Management

6. Sequence Diagram



7. Use Cases

The following use cases illustrate the different ways users will interact with the app:

Use Case 1: User Login

- Description: Allows users to log into the app with their credentials.

- Actors: User

- Preconditions: User must have valid credentials.

- Main Flow:

  - User enters username and password.
  - System validates credentials and grants access.

- Alternative Flows: Invalid credentials result in an error message.

Use Case 2: Incident Reporting

- Description: Allows users to report incidents, near-misses, and hazards.

- Actors: User

- Preconditions: User is logged in.

- Main Flow:

  - User selects incident type.
  - User provides incident details.
  - User attaches media (photos, videos, audio).
  - User rates severity and urgency.
  - User submits the report.

- Alternative Flows: User can add location information.

Use Case 3: Real-time Chat

- Description: Enables real-time chat between safety officers and users involved in the I incident.

- Actors: User, Safety Officer

- Preconditions: Incident report is submitted.

- Main Flow:

  - Safety officer views the incident report.
  - Safety officer initiates a chat with the user.
  - User and safety officer exchange messages.

- Alternative Flows: Safety officer can assign tasks.

Use Case 4: Incident Tracking and Management

- Description: Allows safety officers to view, assign tasks, set deadlines, and monitor the resolution process.

- Actors: Safety Officer

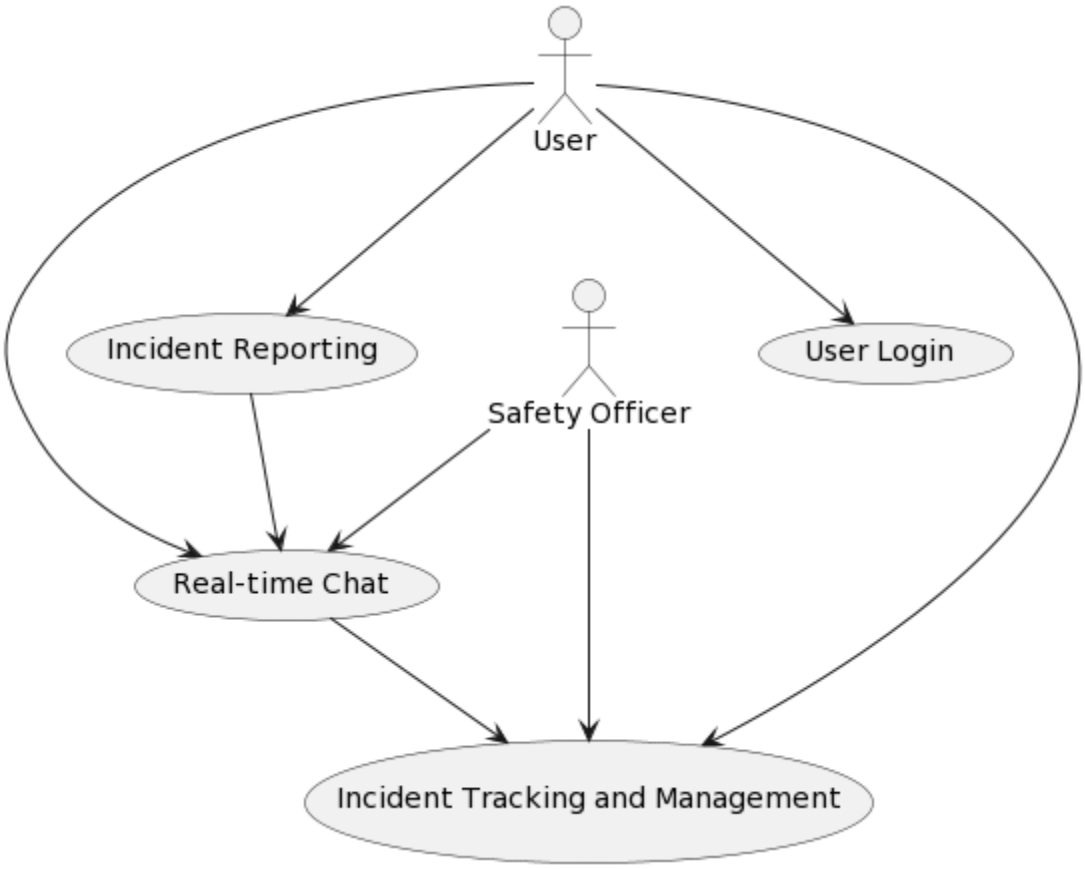
- Preconditions: Incident report is submitted.

- Main Flow:

- Safety officer logs into the dashboard.
- Safety officer views reported incidents.
- Safety officer assigns task and sets deadlines.
- Safety officer monitors the resolution process.

- Alternative Flows: Safety officer generates reports.

8. Use Case Diagram



9. Data Model

Entities:

1. User:
   - Attributes: UserID (Primary Key), Username, Password, UserType, FirstName, LastName, Email, Phone.

2. Incident:
   - Attributes: IncidentID (Primary Key), IncidentType, IncidentDetails, Location, Severity, Urgency, Timestamp.

3. MediaAttachment:
   - Attributes: AttachmentID (Primary Key), IncidentID (Foreign Key), MediaType, FileURL, Timestamp.

4. Task:
   - Attributes: TaskID (Primary Key), IncidentID (Foreign Key), AssigneeID (Foreign Key), Description, Deadline, Status, Timestamp.

5. SafetyOfficer:
   - Attributes: SafetyOfficerID (Primary Key), UserID (Foreign Key).

Relationships:

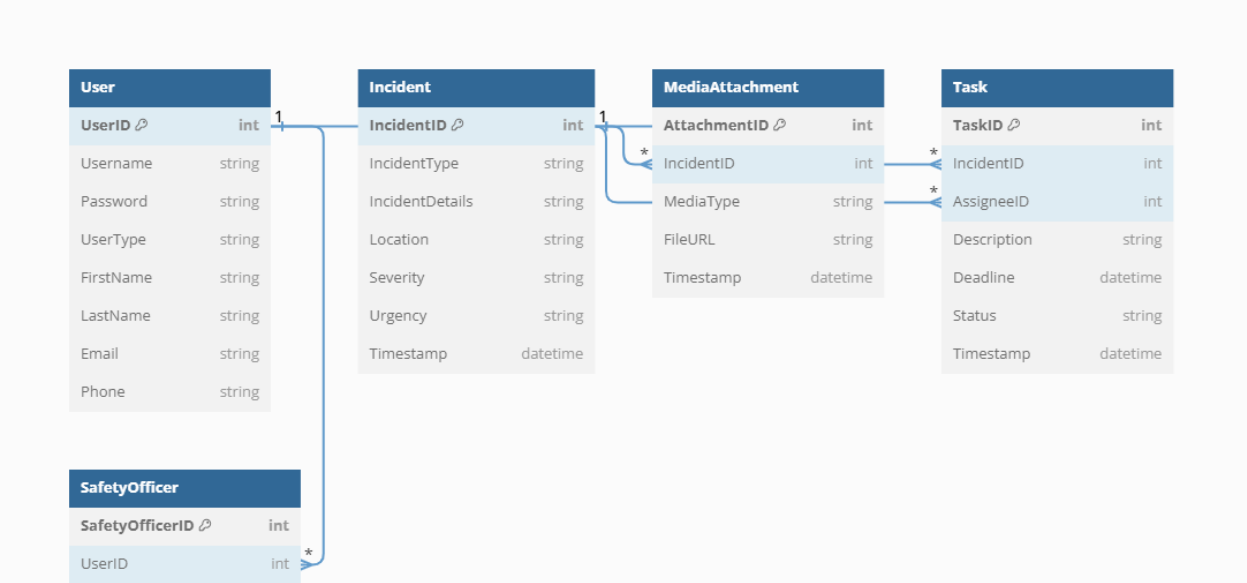
1. User-Initiated Incidents:
   - User (1) ----< Incident (N)

2. Incident-Media Attachments:
   - Incident (1) ----< MediaAttachment (N)

3. Incident-Assigned Tasks:
   - Incident (1) ----< Task (N)

4. Safety Officer-Managed Tasks:
   - SafetyOfficer (1) ----< Task (N)

10. Entity-Relation Diagram



11. Technical Specifications

The following technical specifications are needed to run the app:

Hardware: A web server with at least 4GB of RAM and 20GB of storage space.

Software: .NET Framework 4.8, React, PostgreSQL, and MySQL Workbench.

12. Testing

The app will be tested using the following methods:

- Unit testing
- Integration testing
- System testing
- User acceptance testing

13. Deployment

The app will be deployed to a production environment using the following steps:

- Create a production database.
- Deploy the app code to the production server.
- Configure the app to connect to the production database.
- Promote the app to production.

14. Maintenance

The app will be maintained using the following procedures:

- Regularly monitor the app for performance and security issues.
- Apply security patches and