TECHFIESTA 2025

TITLE PAGE

- **Problem Statement ID – T2K25D1**
- **Problem Statement Title- Emergency Alert System**
- **Domain – Women & Child Safety**
- **Team Name – 2nd Place Warriors**
- **Team Leader Name – Tirthraj Mahajan**

Idea/Approach Details

❖ Proposed Solution

Guardian 360 is a comprehensive safety platform that integrates cutting-edge technology to enhance the security of women and children by providing a proactive, real-time, and adaptable solution for safety concerns.

The solution is designed with both **online** and **offline** capabilities **to ensure accessibility** in any situation.

1. **Travel Alerts**

Users can configure safe-zone alerts (arrival/departure). App sends automatic notifications to chosen contacts, even in offline mode

2. **One-Touch Distress Alert** (Online/Offline):

A mobile app interface that triggers an SOS alert with a single button press.
For **Offline Mode**, it utilizes **SMS fallback mechanisms** to send structured alerts to a server or emergency contacts. Techniques include SMS parsing, compressed message formatting (e.g., UID: 123; LOC: 12.971598,77.594566; INCIDENT: DISTRESS)
For **Online Mode** it sends real-time location and metadata to a centralized server for instant dissemination to emergency contacts, authorities, or volunteer networks.

3. **Adaptive Distress Messaging:**

It uses contextual data like time of day, location risk level, etc. to adjust alert urgency

4. **Automatic Video/Audio Recording:**

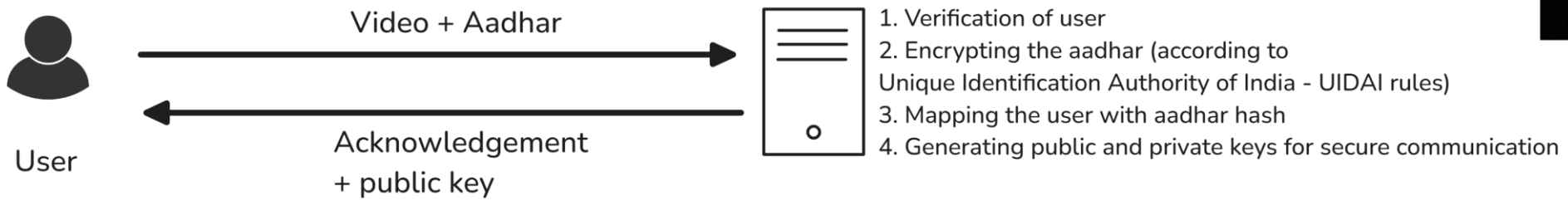
Starts recording immediately after an SOS trigger.
Offline Mode: Stores **encrypted** recordings locally, which are uploaded in segments when connectivity is restored.
Online Mode: Streams video/audio directly to the server for real-time evidence storage
Provides a quick “Cancel” or “I’m Safe” option to minimize unnecessary alerts and reduce server/authority workload.

5. **Incident Reporting & Response System**

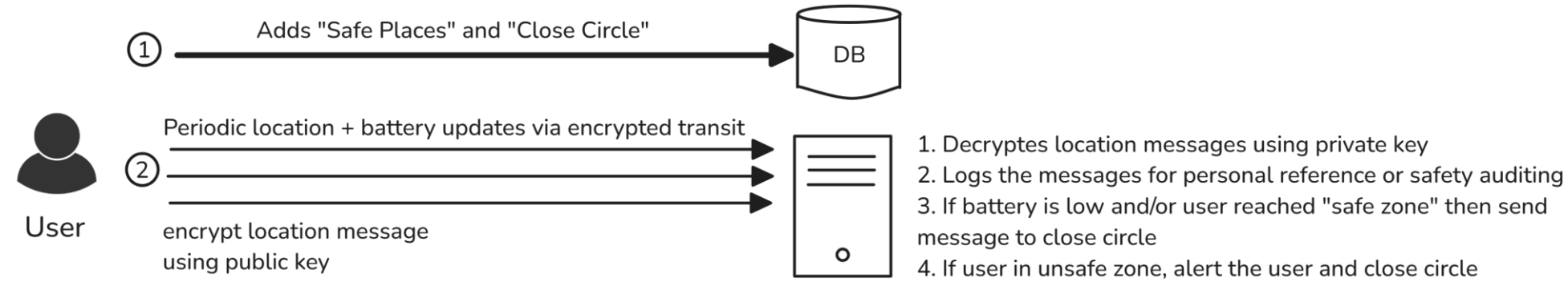
Reporting interface for emergencies or incidents witnessed. AI/ML-based categorization of incidents for quicker triaging (e.g., assault, theft, harassment). Allows users to track the status of reported incidents (e.g., “Investigating,” “In Progress,” “Resolved”).

Process flow diagram

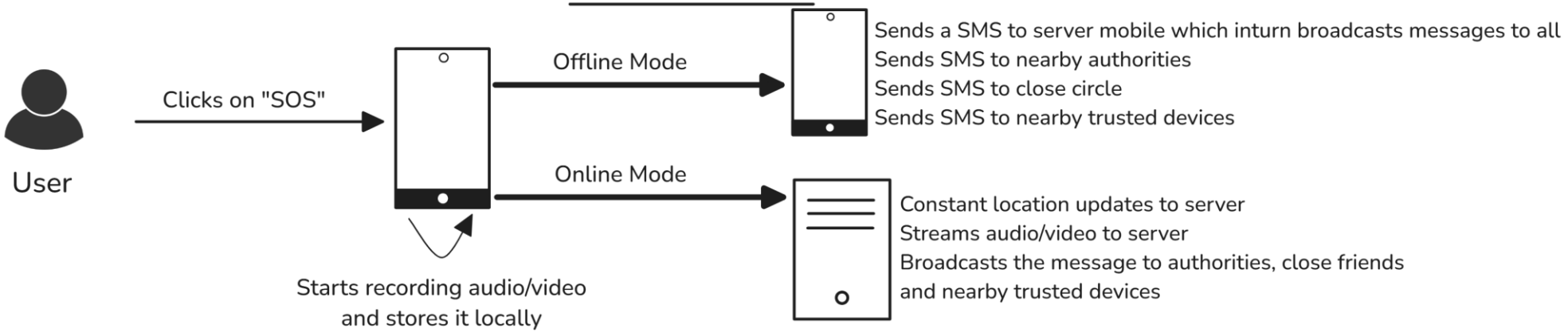
Know Your Customer (KYC)



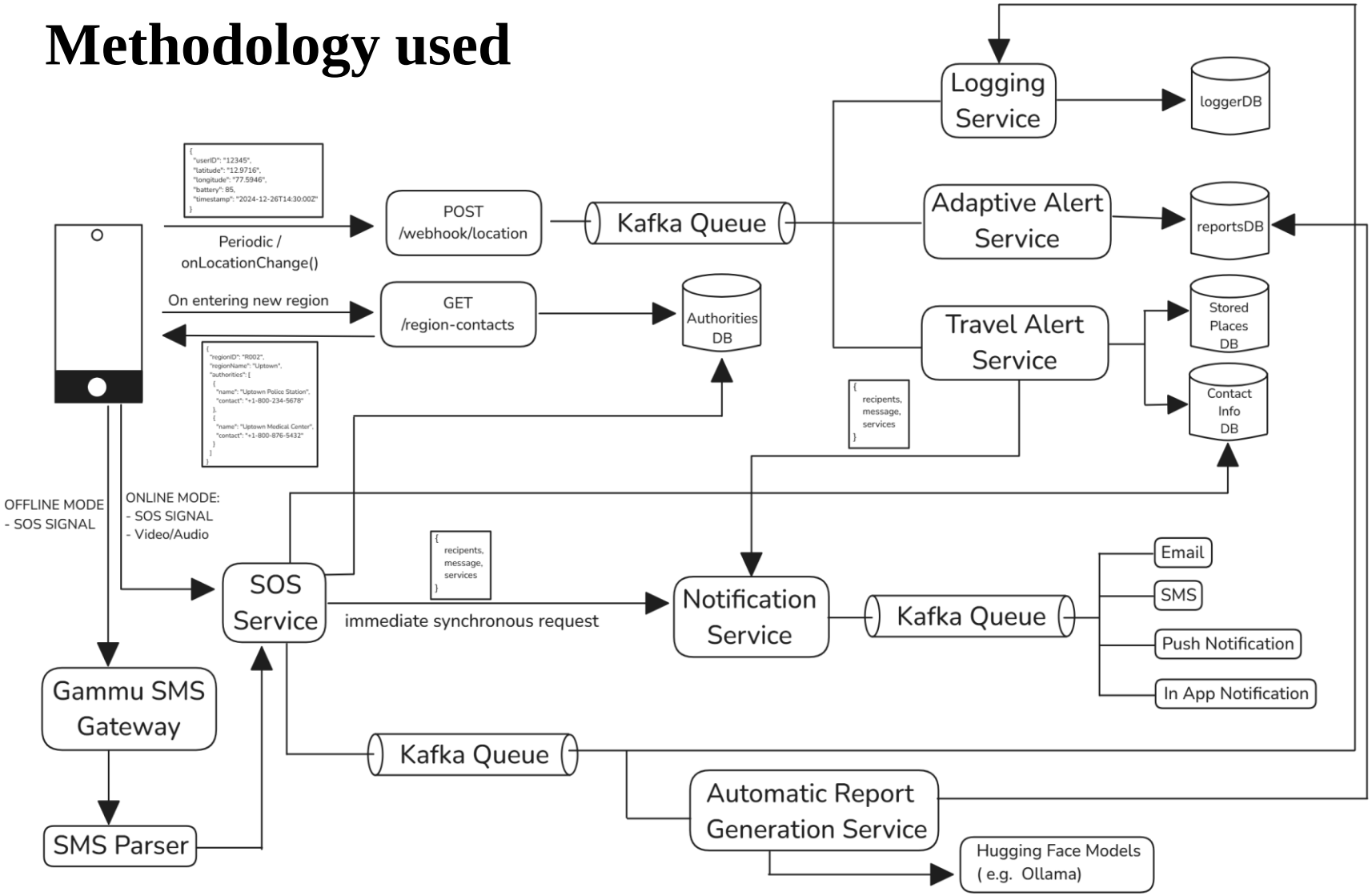
Travel Alerts



SOS Alert



Methodology used



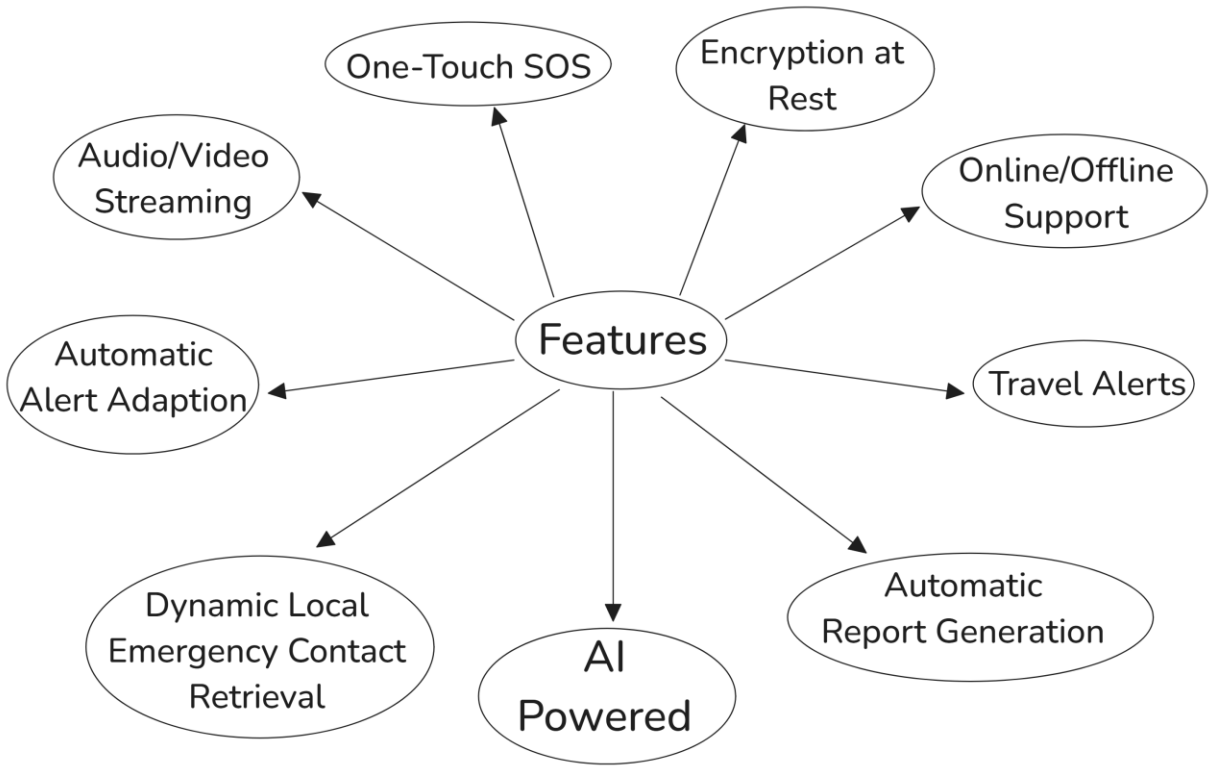
Architecture:
Microservice based
<u>One-Touch SOS Alert:</u>
Offline Mode:
SOS alerts are sent via **SMS** through the **Gammu SMS Gateway**. SMS messages are parsed and forwarded to the **Kafka Queue** for further processing by services like Notification and Logging.
Online Mode:
The mobile app sends SOS alerts with **real-time video/audio streams** and **location data.** Alerts are processed by the SOS Service, which triggers the Notification Service for emergency contact alerts.
<u>Location Tracking:</u>
Periodic location updates are sent to the **Location Webhook**.
Data enters the Kafka Queue and is consumed by: **Logging Service,** for **audits and reports**. **Travel Alert Service,** for geofencing and notifying the close circle about reaching destination

<u>Adaptive Alert Service</u>: Uses AI/ML models (e.g., Hugging Face) to analyze incident urgency and refine alert types. Stores categorized incidents in the **Reports DB** (Elasticsearch).
<u>Automatic Report Generation</u>: AI-driven service generates detailed reports of incidents, using Hugging Face Models for contextual understanding.
<u>Region-Based Updates</u>: When users enter a new region. the Authorities DB is queried for nearby contact information. Notifications are updated for new local emergency services.

Solution Concept and Feasibility

Core Idea:

Guardian 360 is a personal safety application that provides **online and offline SOS capabilities** along with real-time location tracking, adaptive distress alerts, and automatic incident reporting. It uses a microservices-based architecture to ensure scalability, reliability, and modularity.



Feasibility Analysis:

Technical Analysis:

Microservices and Kafka Queues ensure modularity, asynchronous and independent scaling of critical components resulting in **Robustness.**

Offline and Online mode ensures **accessibility** at most of the times

Logging of the locations are useful while auditing

SMS fallback ensures **reliability** in areas with no internet

Locally hosted LLMs enable **accurate incident categorization, adaptive alerts** and **ensuring secrecy**.

Data encryption ensures compliance with privacy regulations

Financial Feasibility:

Utilizes open-source technologies (Kafka, PostgreSQL, Flask), **minimizing licensing fees**.

Potential partnerships with corporate safety programs or NGOs for deployment.

While microservices introduce initial cost, **independent scaling and pay as you go models** ensure cost reduction while scaling.

Cross-platform development with Flutter/React Native ensures compatibility across devices.

Use cases & description

Emergency SOS Alerts

Scenario: A user feels unsafe or is in immediate danger (e.g., harassment, medical emergency, or accident).
The app sends real-time location and distress information to emergency contacts and local authorities.

Incident Reporting and Tracking

Scenario: A user or bystander witnesses an incident (e.g., theft or assault) and needs to report it securely.
The app allows simplified reporting, status tracking, and the option to remain anonymous.

USE CASES

Offline SOS Functionality

Scenario: A user is in a remote area with no internet connectivity. The app sends structured SOS messages via SMS to a server or directly to predefined emergency contacts.

Community Safety Alerts

Scenario: Volunteers or community watchers want to assist in emergencies.
The app sends alerts to nearby volunteers to enable community-driven safety responses.

Travel Safety Monitoring

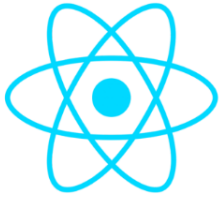
Scenario: A user is traveling in unfamiliar or high-risk areas. The app tracks their location, notifies contacts upon safe arrival, and alerts them if the user enters unsafe zones.

Technology stack used

TECH FIESTA HACKATHON

Mobile

Flutter



React Native

Backend

PostgreSQL

Firebase

Web Frontend

Libraries

Crypto JS

Hugging Face

Constraints

Challenge
Ensuring reliable offline functionality, especially in areas with poor or no network coverage.
Strategy
1. Use structured and compressed SMS formats to minimize message size and ensure compatibility with SMS gateways. 2. Implement SMS retry mechanisms and fallback to alternate emergency numbers if the server mobile is unreachable.

Challenge
Scalability issues as the user base grows, leading to high traffic for SOS alerts and location updates.
Strategy
1. Use a microservices-based architecture with message queues (Kafka) to distribute the load efficiently. 2. Implement autoscaling on cloud infrastructure to dynamically allocate resources during peak usage.

Challenge
Delays in real-time location updates and SOS notifications due to network congestion.
Strategy
Deploy geo-redundant servers closer to user regions to reduce latency. Use lightweight data formats and prioritize critical notifications in the queue.

Challenge
Difficulty in integrating with local emergency services and authorities across various regions.
Strategy
Build partnerships with NGOs, local authorities, and emergency services.

THANK YOU