Got you. I’ll create a **full, professional-level SRS (Software Requirements Specification)** document for your app.
This will follow IEEE SRS format, clear enough for developers, investors, and government/NGO partners.

Software Requirements Specification (SRS)

Project: ***Humanity Safety Volunteer Network App***

Version: 1.0

Date: 07-11-2025

Prepared by: You (Concept Owner)

Assisted by: GPT-5

---

1. Introduction

1.1 Purpose

This document defines the functional and non-functional requirements for a mobile application that enables individuals in unsafe or vulnerable situations to request help from **nearby volunteer helpers**. The system is based on **community-driven, zero-cost, humanity-first assistance**, without monetary transactions.

1.2 Scope

The application allows:

- Users to **request immediate safety assistance**
- Verified local volunteers to **respond**
- Communication through **location sharing, chat, voice, or safe escort**
- Optional escalation to **trusted contacts or emergency services** if needed.

This system promotes a **safer society** through **kindness, presence, and mutual support**.

1.3 Definitions

Term	Meaning
User	Person who may request help
Volunteer	Person who has registered to offer help
SOS Request	A help request message broadcast to nearby volunteers

Term	Meaning
Safe Escort	Volunteer accompanying the user to a safe place
Verified Volunteer	Volunteer with confirmed identity and reputation score

1.4 References

None (original concept).

---

2. Overall Description

2.1 Product Perspective

This is a **mobile-first location-based community safety network system**.
The app consists of:

- Mobile App (Android + iOS + Web Progressive App)
- Cloud Backend (Authentication, Location, Notifications, Logging)
- Optional Admin Dashboard (for moderation)

2.2 Product Functions Summary

Function	Description
Send Help Request	User sends SOS to nearby volunteers
Volunteer Matching	System finds nearest verified volunteers
Real-time Map View	Both parties see live location
Communication	Secure chat / voice call within app
Emergency Escalation	Notify police/trusted contacts if no response
Reputation & Feedback	Ratings to build trust

2.3 User Classes

User Type	Role
General User	Can request help anytime
Volunteer	Offers help to users nearby
Moderator	Handles user verification and misuse reports
Admin	Maintains system & database

2.4 Constraints

- Must maintain **user privacy and location safety**
- Must avoid **false assistance or malicious volunteers**
- Must handle **real-time response with low latency**
- Must comply with **local cyber & safety laws**

2.5 Assumptions

- Users will behave with goodwill.
- Volunteers will not expect monetary rewards.
- Internet and GPS access are available.

---

3. System Requirements

3.1 Functional Requirements

FR-1: User Registration & Authentication

- User can sign up with phone number, email, Aadhaar/ID optional.
- Volunteers require additional verification (e.g., ID upload + selfie + address).

FR-2: Location Tracking

- The app continuously updates approximate user location.
- Location precision adjusts based on safety mode.

FR-3: Initiating Help Request

- User taps **"Need Help"** → Sends broadcast to volunteers in X km range.
- Request includes:
  - User location
  - Basic reason (optional: voice recording for speed)

FR-4: Volunteer Response

- Volunteers receive push notification.
- First volunteer to accept becomes primary helper.
- Others remain backup standby.

FR-5: In-App Communication

- Secure chat (text + voice).
- Option to share **live tracking link** with trusted contacts.

FR-6: Safety Mode Timer

- If no volunteer responds in T seconds → escalate to:
  1. User's emergency contacts
  2. Local police helpline

FR-7: Reputation & Rating System

- After each help event:
    - User rates volunteer
    - Volunteer rates user behavior
- This builds **trust profiles**.

3.2 Non-Functional Requirements (NFR)

NFR	Requirement
Performance	Help request notification delivered ≤ 3 seconds
Availability	System uptime ≥ 99%
Privacy	Location visible *only* during active help
Security	All data encrypted (AES-256, HTTPS)
Scalability	Must handle 1M+ users in national rollout
Usability	Single button emergency UI, 3-tap max to request help

---

4. System Architecture

Suggested Tech Stack

Component	Technology
Mobile App	Flutter / React Native
Backend	Node.js / Python (FastAPI)
Database	Firebase / PostgreSQL
Real-Time Location	Google Maps API / Mapbox
Notifications	Firebase Cloud Messaging
Identity Verification	Aadhaar KYC / DigiLocker API (optional)

---

5. Risk Analysis & Mitigation

Risk	Impact	Mitigation
Fake volunteers	Safety Threat	Identity verification + rating system + auto-flag behavior
False SOS requests	Wasting volunteer time	Pattern detection + cool-down limits
Slow response time	User loses trust	Volunteer density building + auto escalation
Legal issues	Liability	Clear terms of usage + NGO or civic partnership

---

6. Future Enhancements

- AI danger level prediction (voice stress + environment).
- Wearable device integration (smartwatch SOS).
- Public safe-hubs & partnered stores visible in app.
- Government & police APIs for faster response.

---

7. Conclusion

This app is **unique**, **socially transformative**, and **practically impactful**.
With careful **verification, trust-building, and phased rollout**, it can **save lives**, reduce fear, and create a **culture of community safety and humanity.**

✅ Next Step (choose one):

Reply with one number:

1□Create **UI/UX Wireframes**
2□Create **Database Schema**
3□Create **App Flow Diagram**
4□Start **Development Plan & Timeline**
5□Generate **App Name + Logo Concepts**

Which one should we do next? ?