
Software Requirements Specification

for

MindGate

Version 0.1 Prototype

Prepared by Archit Anant

MindGate

19.12.2025

Table of Contents

Table of Contents	ii
Revision History	ii
1. Introduction	2
1.1 Purpose	2
1.2 Document Conventions	2
1.3 Project Scope	2
1.4 References	2
2. Overall Description	2
2.1 Product Perspective	2
2.2 User Classes and Characteristics	2
2.3 Operating Environment	2
2.4 Design and Implementation Constraints	2
2.5 Assumptions and Dependencies	2
3. System Features	2
3.1 System Feature 1	2
3.2 System Feature 2 (and so on)	2
4. Data Requirements	2
4.1 Logical Data Model	2
4.2 Data Dictionary	2
4.3 Reports	2
4.4 Data Acquisition, Integrity, Retention, and Disposal	2
5. External Interface Requirements	2
5.1 User Interfaces	2
5.2 Software Interfaces	2
5.3 Hardware Interfaces	2
5.4 Communications Interfaces	2
6. Quality Attributes	2
6.1 Usability	2
6.2 Performance	2
6.3 Security	2
6.4 Safety	2
6.5 [Others as relevant]	2
7. Internationalization and Localization Requirements	2
8. Other Requirements	2
Appendix A: Glossary	2
Appendix B: Analysis Models	2

Revision History

Name	Date	Reason For Changes	Version
Archit Anant	19.12.25	First Draft	0.1-beta-v1

1. Introduction

1.1 Purpose

The purpose of this document is to define the functional and non-functional requirements for Version 1.0 (Minimum Viable Product) of the MindGate mobile application.

This SRS describes the requirements for the native Android application, which must be developed strictly using Kotlin and Jetpack Compose. It is intended to guide the development team in building the frontend client that facilitates AI interactions, audio/video streaming, and community engagement.

1.2 Document Conventions

The following conventions are used throughout this document:

Must / Shall: Indicates a mandatory requirement. Development cannot proceed without these.

Should: Indicates a recommended requirement that may be deferred to a later version if necessary.

TBD(^): To Be Determined (information pending final decision).

Specific Terminology:

Seeker (User): The end-user looking for mental health support.

Empathic Listener: A non-clinical trained listener (Audio/Chat only).

Professional: A certified psychologist (Clinical, Adolescent, or Corporate) capable of conducting Video sessions.

AI Companion: The LLM-powered chatbot interface.

1.3 Project Scope

MindGate is an AI-powered mental wellness platform that provides users with three distinct pathways to support ("The 3 Connects").

In-Scope Functions (MVP):

Connect to AI:

A text-based chat interface powered by an LLM.

Logic to suggest human support if the AI detects distress keywords.

Connect to Human:

Option A: Empathic Listener: Real-time Text Chat and Audio Call functionality.

Option B: Professional: Video Conference functionality with categorized specialists (Clinical, Adolescent, Corporate).

Community Connect:

A feed or list view to browse community topics.

Functionality to join and view live community sessions.

Technical Constraints (Strict):

Target Platform: Android (API Level 29+).

Language: Kotlin.

UI Framework: Jetpack Compose (No XML layouts).

Out-of-Scope (Not in MVP):

iOS Application.

Web Interface for Seekers.

AI Voice Mode (Text-only for MVP).

Offline Mode.

1.4 References

Figma UI/UX Designs: [Figma](#)

Backend API Documentation: TBD

2. Overall Description

2.1 Product Perspective

MindGate is a **native Android** application that serves as the client-side interface for the MindGate mental wellness platform.

System Context: The application acts as a frontend for users ("Seekers") to access services. It communicates via **REST/WebSocket APIs[^]** with a **centralized backend server[^]** which manages user authentication, chat history, and matching logic.

External Integration: The app integrates with third-party services for specific features:

- **LLM Provider (e.g., OpenAI/Anthropic)[^]:** For the "Connect to AI" chatbot.
- **Video/Audio SDK (e.g., Agora/Twilio)[^]:** For real-time "Connect to Human" sessions.

2.2 User Classes and Characteristics

The application supports three distinct user roles with varying levels of access and functionality:

1. The Seeker (End User)

Description: The primary user of the application seeking emotional support or mental health advice.

Frequency of Use: High / On-demand.

Key Functions:

Interact with AI Chatbot.

Request connection to Empathic Listeners (Audio/Chat).

Book/Join sessions with Certified Psychologists (Video).
Participate in Community sessions.

Characteristics: Tech proficiency varies. The UI must be simple, calming, and minimize friction to finding help.

2. The Professional (Service Provider)

Description: Individuals vetted by MindGate to provide support. This class is further divided into two sub-types based on verification level:

Sub-type A: Empathic Listener: Permitted to conduct Text and Audio sessions only.

Sub-type B: Certified Psychologist: Permitted to conduct Video sessions (Clinical, Adolescent, Corporate).

Frequency of Use: Scheduled / Shift-based.

Key Functions:

Set status (Online/Offline).

Accept incoming request notifications.

Manage session logs/notes (if applicable).

Characteristics: Requires a stable environment for calls. Needs a "Dashboard" view within the app to manage incoming requests.

3. The Administrator

Description: MindGate internal staff responsible for platform health and safety.

Frequency of Use: Intermittent / Monitoring.

Key Functions:

Moderation of Community content (ban/mute users).

Verification and approval of "Professional" accounts.

Override controls for safety issues.

Characteristics: High privilege level. (Note: Depending on implementation, Admins may perform complex tasks via a Web Portal, but may use the Android app for community monitoring).

2.3 Operating Environment

Hardware Platform: Android Smartphones. (Tablet support is not required for MVP).

Operating System: Android 10.0 (Q) - API Level 29 and higher.

Network: The application requires an active data connection (Wi-Fi, 4G, or 5G). High-bandwidth connection is assumed for Video Conferencing.

2.4 Design and Implementation Constraints

This project has strict technical constraints to ensure modern code standards:

Programming Language: The application must be written strictly in **Kotlin**. Java is not permitted.

UI Toolkit: The User Interface must be implemented using **Jetpack Compose**. Legacy XML (View System) layouts are not permitted.

Architecture: The app should follow **MVVM** (Model-View-ViewModel)

Asynchronous Processing: Kotlin Coroutines and Flow must be used for background operations and data streams.

Third-Party SDK Constraints:

Video/Audio features must be implemented using TBD.

Payment processing (if applicable) must be implemented using TBD.

Dependency Injection: Dagger-Hilt should be used as the dependency injection framework.(Subject to complete removal after testing)

2.5 Assumptions and Dependencies

Permissions: It is assumed the user will grant Microphone and Camera permissions. If denied, the "Connect to Human" features must handle the error gracefully.

Backend Availability: The app depends on the backend API for all content; there is no offline content mode.

AI Latency: The speed of the AI Chatbot response depends on the third-party LLM provider's API latency.

3. System Features

3.1 User Authentication

3.1.1 Description

The entry point for the application. Users must be able to create a secure account to maintain their session history and wallet/payment details.

3.1.2 Functional Requirements

REQ-1: The system shall allow users to Sign Up and Log In using Email and Password.

REQ-2: The system shall allow users to Sign Up and Log In using Google Sign-In (OAuth).

REQ-3: The system must validate email formats and enforce a minimum password strength (e.g., 8+ characters).

REQ-4: Upon successful login, the app must store a secure Auth Token (JWT) locally to maintain the user session without requiring re-login on every launch.

3.1.1 Functional Requirements

TBD

3.2 Connect to AI (Chatbot)

3.2.1 Description

An always-available, text-based chat interface powered by an external AI backend. This is the default support layer.

3.2.2 Functional Requirements

REQ-5: The user shall be able to type and send text messages to the AI bot.

REQ-6: The UI must display a "Typing Indicator" animation while waiting for the backend response.

REQ-7: The chat interface must support auto-scrolling to the newest message.

REQ-8: (Safety) If the backend identifies a message as a crisis, the app must render a specific "Crisis Alert" UI component instead of a standard chat bubble.

3.3 Connect to Human: Empathic Listener

3.3.1 Description

This feature connects the user to a non-clinical listener for an Audio Call. This is an on-demand service charged on a "Pay-Per-Call" basis.

3.3.2 Functional Requirements

REQ-9: The user shall be able to view a list of currently online Listeners.

REQ-10: Upon selecting a listener, the system must prompt the user to authorize a Payment Transaction.

REQ-11: The Audio Call shall only initiate after payment confirmation is received from the payment gateway.

REQ-12: The app must request Microphone Permission before the call begins.

REQ-13: The app must provide controls to Mute Microphone and End Call.

3.4 Connect to Human: Professional Psychologist

3.4.1 Description

This feature connects the user to a certified professional for a Video Conference. This is a Scheduled service charged on a "Pay-Per-Session" basis.

3.4.2 Functional Requirements

REQ-14: The user shall be able to view a professional's availability calendar.

REQ-15: The user shall be able to select a time slot and Schedule an appointment.

REQ-16: The system must prompt for Payment at the time of booking.

REQ-17: The system must schedule a local notification (AlarmManager) to remind the user 15 minutes before the session starts.

REQ-18: At the scheduled time, the user shall be able to join the Video Room.

REQ-19: The app must request Camera and Microphone Permissions before joining the video room.

3.5 Community Connect

3.5.1 Description

A "YouTube Live" style broadcast feature where professionals host sessions for multiple users.

3.5.2 Functional Requirements

REQ-20: The user shall be able to browse a feed of upcoming Community Sessions.

REQ-21: The user shall be able to purchase a "Ticket" or "Pass" to access a specific session.

REQ-22: The user interface shall include a Video Player (Receive-only) to watch the host.

REQ-23: The user interface shall include a text-chat stream to interact with the host and other community members.

4. Data Requirements

4.1 Logical Data Model

Subjected to major changes

The system will manage the following high-level data entities:

User Profile (Seeker):

- System ID (UUID), Email, Password Hash, Google Auth ID.
- Nickname (for anonymity), Wallet Balance/Payment History.

Professional/Listener Profile:

- System ID, Real Name, Certification Documents.
- Category (Listener vs. Clinical/Adolescent/Corporate), Verification Status.
- Availability Schedule (Time slots).

Appointment/Booking:

- Booking ID, User ID, Professional ID.
- Scheduled Date/Time, Status (Confirmed, Completed, Cancelled).
- Payment Transaction ID.

Session Record:

- Timestamp, Duration, Type (AI Chat / Audio Call / Video Call).
- Metadata only (No recording of audio/video content).

Community Session:

- Session ID, Host ID, Topic, Scheduled Time, Ticket Price.
- Participant List (Users who paid).

More TDB

4.2 Data Dictionary

Timestamps: All dates and times (Appointment slots, Chat logs) must be stored in UTC format (ISO 8601) on the backend to handle time zone conversions correctly.

Currency: All monetary values (Session fees, Wallet balance) shall be stored as Integers (e.g., cents/paisa) to prevent floating-point rounding errors.

Auth Token: The application shall utilize JWT (JSON Web Tokens) for session validation, formatted as a standard Bearer token string.

4.3 Reports

Transaction Receipt: Upon successful payment for a call or session, the system must generate a digital receipt record accessible in the user's "Payment History."

Session Summary: After a session ends, a summary record (Duration, Professional Name, Date) is generated for the User's history log.

4.4 Data Acquisition, Integrity, Retention, and Disposal

Retention Policy: Text chat logs (AI and Listener) shall be retained for TBD years for legal/safety compliance and then automatically purged.

Video Privacy: Video and Audio streams must not be recorded or stored on the server. Only the metadata of the call (time started, time ended) shall be saved.

Disposal: If a user requests account deletion, all Personally Identifiable Information (PII) must be removed from the active database within 30 days (Right to be Forgotten).

5. External Interface Requirements

5.1 User Interfaces

Design System: The application must be built using Material Design 3 components within Jetpack Compose.

Theme: The app must support a Light Theme by default (Dark Theme is optional for MVP). The color palette should utilize calming tones to align with mental wellness goals.

Orientation: The application interfaces shall be restricted to Portrait Mode only to simplify MVP development.

Key UI Components:

- **Bottom Navigation Bar:** To switch between Home, Connect (Human), Community, and Profile.
- **Video Interface:** A full-screen view containing the remote video feed, a smaller local camera preview (PIP), and floating controls (Mute, Camera Toggle, End Call).
- **Payment Sheet:** A modal bottom sheet to confirm transaction details before connecting to a human.

5.2 Software Interfaces

The Android client interacts with the following external software systems(like apis, servers and DBs) :

- TBD

5.3 Hardware Interfaces

The application requires direct access to specific mobile hardware:

Camera: Required for "Connect to Professional" (Video). Access is requested only when the user enters the video room.

Microphone: Required for "Connect to Listener" (Audio) and "Connect to Professional" (Video).
Audio Output: The app must manage audio routing, automatically switching between the Earpiece (for private calls) and Speakerphone (for video calls), or routing to Bluetooth headsets if connected.

Network Module: The app must monitor connectivity. If the network drops during a call, the app must attempt to reconnect for 30 seconds before terminating the session.

5.4 Communications Interfaces

Transport Security: All network traffic must use HTTPS (TLS 1.2 or higher). Cleartext traffic (HTTP) is strictly prohibited.

Data Serialization: API requests and responses shall be formatted in JSON.

Push Notifications: The app communicates with Firebase Cloud Messaging (FCM) to receive system alerts, appointment reminders (15 mins prior), and "Session Starting" notifications.(SUBJECT TO TESTING)

6. Quality Attributes

6.1 Usability

Simplicity: The "Panic" or "Get Help" flow must require no more than 3 taps from the home screen to initiate a connection.

Accessibility: The app UI must support Android's native Screen Reader (TalkBack) for visually impaired users.

Color Theory: The UI color palette should use calming tones (e.g., pastels, blues, greens) and avoid aggressive colors (bright reds) except for error states.

6.2 Performance

Launch Speed: The application must be interactive within 2 seconds of launch (Cold Start).

Video Latency: Video latency for Professional sessions must remain under 400ms on a standard 4G connection to prevent "talking over each other."

Battery Usage: The app should not consume more than 5% of battery during a 30-minute background audio session.

6.3 Security

Encryption at Rest: Any sensitive user data stored locally on the Android device (e.g., cached chat history in Room Database) must be encrypted using SQLCipher or the Android Keystore system.

Encryption in Transit: All network communications between the App and the Server must be encrypted using TLS 1.2+ (HTTPS/WSS).

Screenshot Prevention: To protect user privacy during sensitive interactions, the application must disable the operating system's screenshot/screen-recording functionality on the Chat and Video Call screens using WindowManager.LayoutParams.FLAG_SECURE.

Session Management: User authentication tokens (JWT) should have a predefined expiry time (e.g., 30 days). The app must automatically refresh these tokens or prompt the user to re-login if the token becomes invalid.

No PII in Logs: The application must strictly avoid logging Personally Identifiable Information (PII) such as email addresses, passwords, or chat content into the Android System Log (Logcat) to prevent data leaks during debugging.

6.4 Safety

Crisis Interception: If the User inputs text into the AI Chatbot indicating immediate self-harm (keywords: suicide, kill, hurt, die), the system must override the AI response and present a "**Crisis Overlay**" with immediate helpline numbers.

Disclaimer: Every "Connect to Listener" session must begin with a visible toast or banner stating: "**Listeners are not clinical professionals.**"

Report Mechanism: Seekers must have a "Report" button available during all Human interactions to flag inappropriate behavior.

7. Internationalization and Localization Requirements

Language: For Version 1.0 (MVP), the user interface and AI interactions will be supported in **English only**.

Currency: All Professional session fees (if applicable) will be displayed in **INR**.

Date/Time: Time slots for appointments must automatically convert to the **User's local device time zone**, but be stored in **UTC on the server**.

Appendix A: Glossary

Seeker: The end-user looking for support.

Empathic Listener: A trained peer supporter (non-clinical).

Professional: A licensed psychologist.

MVP: Minimum Viable Product.

LLM: Large Language Model (the technology behind the AI).

Composable: A UI component in Jetpack Compose.

FCM: Firebase Cloud Messaging (used for notifications).

More TBD