

Telemedicine Platform

Video Consults • Messaging • E-Prescriptions

Our Goal


Connect patients and doctors for safe video calls, chat, triage, and online prescriptions

● Secure Access ● Fast Video Quality ● 24/7 Availability

Requirements Pack

Stakeholders

 Patients

 Doctors

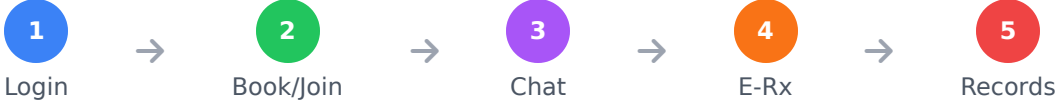
 Admins

 Pharmacists




User Stories

- Patient books a video visit and chats with the doctor
- Doctor starts consultation and sends e-prescription
- Admin manages users and audits logs

Use-Case Flow



Constraints & Assumptions

-  Must follow healthcare rules
-  Video must work on weak networks
-  All data must be encrypted

Core Features

Comprehensive telemedicine platform capabilities



Patient & Doctor Portals

Dedicated interfaces for patients and healthcare providers



Appointment Scheduling

Smart booking system with availability management



WebRTC Video Calls

High-quality video consultations with real-time communication



Real-time Chat

Instant messaging between patients and doctors



E-Prescriptions

Digital prescription management and pharmacy integration



Notification System

Automated alerts and reminders for appointments



Audit Logs

Comprehensive tracking of all platform activities



Triage & Symptom Reporting

AI-powered symptom assessment and priority routing

Architecture Overview

Trade-offs Analysis

Monolith

Simple but not scalable

Microservices

Best for scaling video & messaging

Serverless

Good for events but not for video calls

Final Choice

Microservice + layered services + micro-frontends

Key Components

Media Service

SFU/MCU for video

API Gateway

Request routing

Auth Service

Authentication

Chat Service

Real-time messaging

E-Rx Service

Prescriptions

Notification

Alerts & updates

Message Queues

Event handling

CDN

Static content

Database Design

Scalable data architecture for telemedicine platform

Data Types



Relational DB
Users, appointments, PHI



Document Store
Medical notes



Audit Store
All actions



Object Storage
Attachments, reports

ERD Summary



Users
→ Roles



Patients
→ Appointments



Doctors
→ Consult notes



Prescriptions
→ Pharmacy

Indexing & Scaling



Indexing
Index on userId, appointmentId



Partitioning
Partition by region



Sharding
Shard video logs due to size

Design Patterns Used

Architectural patterns and best practices for scalable telemedicine platform



Creational Patterns

- **Builder Pattern**
Build e-prescriptions safely with step-by-step validation



Structural Patterns

- **Proxy Pattern**
Hide media complexity behind simple interfaces
- **API Gateway**
Acts as reverse proxy for microservices



Behavioral Patterns

- **CQRS**
Command-Query separation for better performance
- **Retry + Idempotency**
Safe retry mechanisms for critical operations



Anti-Patterns Avoided






- **No "God Service"**
Avoid monolithic services doing everything
- **Avoid Tight Coupling**
Maintain loose coupling between services
- **No Shared Mutable State**
Prevent race conditions and data corruption

Security, Performance & Reliability

Ensuring robust, scalable, and secure telemedicine operations








Security

-  **Strong Authentication**
OIDC implementation
-  **RBAC Access Control**
Fine-grained permissions
-  **End-to-End Encryption**
Transit + at rest
-  **OWASP Top 10**
Comprehensive protection
-  **DLP Monitoring**
Sensitive data checks








Performance

-  **In-Memory Caching**
Redis/Memcached
-  **CDN Distribution**
Global content delivery
-  **Load Balancing**
Traffic distribution
-  **Backpressure Handling**
Flow control
-  **HA Replication**
High availability



Reliability

-  **Circuit Breakers**
Failure isolation
-  **Retry Mechanisms**
With exponential backoff
-  **Disaster Recovery**
Comprehensive DR plan
-  **Health Monitoring**
Real-time metrics
-  **Data Backup**
Automated backups

API, Observability & Tech Stack



API Design



REST/gRPC APIs

Auth, Appointments, Video Session, Chat, E-Rx



Clear Versioning

API version management & backward compatibility



Error Handling

Error codes + idempotency keys



Observability



Comprehensive Logs

Logs for every request & system event



Key Metrics

Latency, uptime, video quality monitoring



Distributed Traces

End-to-end request tracing across services



Alerts & Runbooks

Proactive monitoring & incident response



Tech Stack



Frontend

React + Micro Frontends



Backend

Node.js/Go/Python microservices



Database

PostgreSQL + MongoDB



Media

WebRTC SFU (Janus/Mediasoup)



Queue

Kafka/RabbitMQ



Deployment

Docker + Kubernetes