**System Overview –**

The **Medicine-Sharing App** enables students within a campus/hostel to request and donate medicines in real time.

**System Architecture –**

User (Android App) 🡨🡪 API Gateway 🡨🡪 Authentication (JWT session based)

|🡪 Medicine Request Service

|🡪 User Management Service

|🡪 Notification Service (WebSocket)

|🡪 OTP Verification Service (Caching thru Redis)

|🡪 Storage Service (for medicine images)

|🡪 MongoDB Database

**Key Functional Modules**

**3.1 User Authentication (JWT)**

* Users register/login using email, phone, or social authentication.
* JWT tokens are used for securing API calls.

**3.2 Medicine Request System**

**Request API Flow**

1. **User Requests a Medicine**:
   * Enter medicine name, quantity, tags, and optional image.
   * Send request.
2. **Backend Matches Donors**:
   * Find **available donors** using:
     + **Location-based filtering** (MongoDB geospatial queries).
     + **Availability filtering**
3. **Notification System**:
   * **If online users are available** → Send **real-time notifications (WebSocket)**.
   * **If all are offline** → Save request and notify **when users come online**.
4. **Request Expiry**:
   * If **no donor accepts within 2 hours**, the request is automatically **deleted**.

**3.3 OTP-Based Verification**

* When a donor accepts a request, a **6-digit OTP** is generated.
* The requester **enters the OTP** to confirm the medicine handover.
* **After OTP verification**, the stock is **decreased by one**.
* **Security:** OTP is stored **only in a cache (Redis)** for efficiency.

**Technical Details/ TechStacks we are using –**

**1. Backend (API & Business Logic)**

**Spring Boot (Java)** → Main backend framework.  
**Spring Security** → Authentication & authorization.  
**JWT (JSON Web Token)** → Secure user authentication.  
**RabbitMQ / Kafka** → Event-driven architecture for async processing.

**2. Database & Storage**

**MongoDB** → Main NoSQL database.  
**Redis** → Caching OTPs & frequently accessed data.

**3. Real-Time Notifications**

**4. WebSockets** → Live notifications & request updates.  
**Firebase Cloud Messaging (FCM)** → Push notifications for Android.

**5. Offline & Background Sync**

**WorkManager (Android)** → Handles background sync.