UML DIAGRAMS FOR SWASTHSETU

Use Case Diagram & Class Diagram

PART 1: USE CASE DIAGRAM

Overview
A Use Case Diagram shows the interactions between actors (users) and the system. It illustrates what functionalities the system provides to different types of users.

1. ACTORS (System Users)

- Patient
  - End-user who accesses telemedicine services for consultations and health management

1. Key Responsibilities:
    - Register and manage account
    - Book consultations with doctors
    - Access personal health records
    - Use symptom checker
    - Search for medicine availability
    - Join video consultations
    - View consultation history
    - Manage profile and preferences

- Doctor
  - Healthcare provider who conducts consultations and manages patient care

2. Key Responsibilities:
    - Register as doctor with credentials
    - Set availability and manage schedule
    - View pending consultations
    - Conduct video consultations
    - Update patient health records
    - Write prescriptions

    - View consultation analytics
    - Provide medical advice and guidance

- Pharmacy Staff
    - Pharmacy employee who manages medicine inventory and fulfills prescriptions

3. Key Responsibilities:
    - Register pharmacy in system
    - Update medicine inventory
    - View prescriptions issued by doctors
    - Receive notifications for new prescriptions
    - Manage medicine stock and pricing
    - Fulfill prescription orders

- Administrator
    - System administrator responsible for overall system management and monitoring

4. Key Responsibilities:
    - Manage user accounts (create, edit, delete)
    - Monitor system performance and health
    - View system analytics and reports
    - Manage system policies and configurations
    - Audit user activities
    - Generate reports for stakeholders

- External Services
    - Third-party services integrated with the system

5. Key Responsibilities:
    - Payment gateway services
    - SMS notification service
    - Email delivery service
    - Cloud storage services

2. USE CASES (System Features)

1. Patient Use Cases

- **Register Account:** Patient creates a new account using phone number and OTP verification
- **Login:** Patient logs into the system using credentials
- **Book Consultation:** Patient schedules an appointment with available doctor
- **View Health Records:** Patient accesses personal medical history and documents
- **Check Medicine Availability:** Patient searches for medicines and locates nearest pharmacies
- **Use Symptom Checker:** Patient inputs symptoms and receives AI-powered preliminary assessment
- **Join Video Consultation:** Patient participates in scheduled video call with doctor
- **View Consultation History:** Patient reviews past consultations, notes, and summaries
- **Update Profile:** Patient modifies personal information, address, and preferences
- **Receive Notifications:** Patient gets appointment reminders and system updates
- **Download Prescription:** Patient retrieves issued prescription in PDF format
- **Rate Consultation:** Patient provides feedback and rating after consultation

2. Doctor Use Cases

- **Register as Doctor:** Doctor creates account and provides credentials and specialization
- **Set Availability:** Doctor defines working hours and available time slots
- **View Pending Consultations:** Doctor sees list of scheduled consultations awaiting
- **Conduct Video Consultation:** Doctor participates in real-time video call with patient
- **Update Health Records:** Doctor adds diagnosis, symptoms, and medical findings to patient record
- **Write Prescription:** Doctor creates prescription with medicines, dosage, and instructions
- **View Analytics:** Doctor reviews consultation statistics and performance metrics
- **Respond to Inquiries:** Doctor answers patient questions between consultations
- **Access Patient History:** Doctor views complete medical history before consultation
- **Generate Report:** Doctor creates consultation summary and medical report

3. Pharmacy Use Cases

- **Register Pharmacy:** Pharmacy staff registers pharmacy with location and contact details
- **Update Inventory:** Pharmacy staff updates available medicines and stock levels
- **View Prescriptions:** Pharmacy staff sees patient prescriptions requiring fulfillment
- **Receive Notifications:** Pharmacy staff gets alerts for new prescription orders
- **Manage Medicine Stock:** Pharmacy staff tracks medicine expiry and reorder levels
- **Search Medicine:** Pharmacy staff searches for medicine availability across network

- **Update Pricing:** Pharmacy staff modifies medicine prices in inventory

4. Administrator Use Cases

- **Manage Users:** Administrator creates, edits, or removes user accounts
- **View Analytics:** Administrator monitors consultation volume and system metrics
- **Monitor Performance:** Administrator tracks system uptime and response times
- **Generate Reports:** Administrator creates usage reports for stakeholders
- **Manage Policies:** Administrator configures system policies and settings
- **Audit Activities:** Administrator reviews user activity logs for security
- **Configure System:** Administrator sets up system parameters and integrations

3. USE CASE RELATIONSHIPS

Include Relationships (<<include>>)
These relationships show mandatory dependencies - one use case must include another.

- **Book Consultation <<includes>> Authenticate User:** User must login before booking
- **View Health Records <<includes>> Authenticate User:** User must be authenticated
- **Write Prescription <<includes>> Store Health Records:** Prescription must be saved to records
- **Conduct Video Consultation <<includes>> Encrypt Data:** Video call must be encrypted
- **Update Medicine Inventory <<includes>> Sync Offline Data:** Changes must sync with cloud
- **Update Health Records <<includes>> Store Health Records:** Records must be persisted
- **Process Payment <<includes>> Validate Payment:** Payment must be validated first

Extend Relationships (<<extend>>)
These relationships show optional extensions - one use case may optionally extend another.

- **View Consultation History <<extends>> View Health Records [when patient selects history]:** Extended view with detailed history
- **Rate Consultation <<extends>> Join Video Consultation [after consultation ends]:** Optional rating after call
- **Process Payment <<extends>> Book Consultation [if premium service]:** Payment only for paid consultations
- **Send Reminder Notification <<extends>> Receive Notifications [24 hours before appointment]:** Conditional reminder

PART 2: CLASS DIAGRAM

Overview
A Class Diagram shows the structure of the system through classes, their attributes, methods, and relationships. It forms the foundation for database design and object-oriented development.

1. CLASS DEFINITIONS WITH ATTRIBUTES & METHODS

A. User (Abstract) - Base class for all user types in the system

1. Attributes:

Name	Type	Modifier	Description
userId	String	PK	Unique user identifier
name	String		Full name of user
email	String		Email address
phone	String		Phone number
address	String		Residential address
profilePicture	String		URL to profile image
createdAt	DateTime		Account creation timestamp
updatedAt	DateTime		Last update timestamp
password	String	Private	Encrypted password
isActive	Boolean		Account status

2. Methods:

Method Name	Return Type	Description
+ register()	Boolean	Create new user account
+ login(credentials)	Boolean	Authenticate user
+ logout()	void	End user session
+ updateProfile(data)	void	Modify user information
+ getProfile()	UserProfile	Retrieve user details
+ resetPassword()	void	Initiate password reset
+ changePassword(oldPwd, newPwd)	Boolean	Change user password
+ {abstract} getRolePermissions()	String[]	Get role-based permissions
+ deleteAccount()	void	Deactivate user account

B. Patient - Represents a patient user who accesses healthcare services (Extends User)

1. Attributes:

Name	Type	Modifier	Description
patientId	String	PK	Unique patient identifier
dateOfBirth	Date		Patient birth date
gender	String		Gender (M/F/Other)
bloodType	String		Blood group
allergies	String[]		List of known allergies
emergencyContact	String		Emergency contact name and number
insuranceId	String		Insurance policy number
languagePreference	String		Preferred language (Hindi/Punjabi/English)
consultationCount	Integer		Total consultations completed
lastConsultation	DateTime		Date of most recent consultation

2. Methods:

Method Name	Return Type	Description
+ bookConsultation(doctorId, dateTime)	Consultation	Schedule appointment
+ viewHealthRecords()	List<HealthRecord>	Retrieve medical history
+ searchMedicines(medicineName)	List<Medicine>	Search for medicines
+ updateMedicalHistory(data)	void	Add new medical information
+ getConsultationHistory()	List<Consultation>	Get past consultations
+ uploadDocument(file)	Document	Upload medical document
+ setLanguagePreference(language)	void	Change UI language
+ getOfflineData()	LocalData	Retrieve cached data
+ cancelConsultation(consultationId)	Boolean	Cancel booked appointment
+ rateConsultation(consultationId, rating)	void	Provide consultation feedback

- **C. Doctor -** Represents a healthcare provider conducting consultations (Extends: User)

1. Attributes:

Name	Type	Modifier	Description
doctorId	String	PK	Unique doctor identifier
specialization	String		Medical specialty
licenseNumber	String		Medical license number
experience	Integer		Years of medical experience
qualification	String[]		Educational qualifications
consultationFee	Decimal		Fee per consultation
rating	Float		Average patient rating (0-5)
totalConsultations	Integer		Total consultations conducted
isVerified	Boolean		License verification status
availabilityStatus	String		Current availability (Online/Offline)

2. Methods:

Method Name	Return Type	Description
+ setAvailability(timeSlots)	void	Define working schedule
+ getAvailableSlots(date)	List<TimeSlot>	Get free time slots
+ viewPendingConsultations()	List<Consultation>	See upcoming appointments
+ conductConsultation(consultationId)	void	Start video consultation
+ updatePatientRecords(patientId, data)	void	Add diagnosis and notes
+ writePrescription(patientId, medicines)	Prescription	Create prescription
+ getConsultationAnalytics()	Analytics	View performance metrics
+ respondToPatient(patientId, message)	void	Send message to patient
+ endConsultation(consultationId)	void	Complete consultation
+ generateConsultationReport(consultationId)	Document	Create medical report

- **D. PharmacyStaff -** Pharmacy employee managing medicine inventory (Extends: User)

1. Attributes:

Name	Type	Modifier	Description
pharmacyStaffId	String	PK	Unique staff identifier
pharmacyId	String	FK	Associated pharmacy
position	String		Job title (Manager/Technician)
department	String		Department assignment
joinDate	Date		Employment start date
performanceRating	Float		Staff performance score

2. Methods:

Method Name	Return Type	Description
+ updateInventory(medicineId, quantity)	void	Modify stock levels
+ viewMedicineStock()	List<Medicine>	Check available medicines
+ receivePrescription(prescriptionId)	void	Get prescription order
+ searchMedicine(medicineName)	Medicine	Find medicine details
+ updateMedicineDetails(medicineId, data)	void	Modify medicine info
+ generateInventoryReport()	Report	Create stock report
+ checkMedicineExpiry()	List<Medicine>	Find expiring medicines
+ notifyLowStock()	void	Alert on low inventory

E. Administrator - System administrator with full control (Extends: User)

1. Attributes:

Name	Type	Modifier	Description
adminId	String	PK	Unique admin identifier
role	String		Admin role (Super/Regional/Local)
department	String		Department assignment
accessLevel	Integer		Permission level (1-5)
lastLoginTime	DateTime		Last system access

2. Methods:

Method Name	Return Type	Description
+ **manageUsers(action, userId)**	void	Create/edit/delete users
+ **viewSystemAnalytics()**	Analytics	Get system metrics
+ **monitorPerformance()**	PerformanceMetrics	Check system health
+ **generateReports(type, dateRange)**	List<Report>	Create reports
+ **managePolicies(policy, value)**	void	Configure system policies
+ **auditLogs(criteria)**	List<AuditLog>	Review activity logs
+ **suspendUser(userId, reason)**	void	Disable user account
+ **viewSystemErrors()**	List<Error>	Check system errors

F. Consultation - Represents a doctor-patient consultation session

1. Attributes:

Name	Type	Modifier	Description
consultationId	String	PK	Unique consultation ID
patientId	String	FK	Associated patient
doctorId	String	FK	Associated doctor
scheduledTime	DateTime		Appointment time
actualStartTime	DateTime		When call actually started
actualEndTime	DateTime		When call actually ended
status	String		Status (Pending/Ongoing/Completed/Cancelled)
duration	Integer		Call duration in minutes
consultationType	String		Type (Video/Audio/Text)
topic	String		Chief complaint/reason
notes	String		Doctor notes and findings
prescription	Prescription	FK	Associated prescription
recordingURL	String		Video call recording link
rating	Float		Patient rating (1-5)
feedback	String		Patient feedback comment
createdAt	DateTime		Booking timestamp
totalFee	Decimal		Consultation charges

2. Methods:

Method Name	Return Type	Description
+ scheduleConsultation()	Boolean	Book appointment
+ startConsultation()	void	Initiate call
+ endConsultation()	void	End call and finalize
+ rescheduleConsultation(newTime)	Boolean	Change appointment
+ cancelConsultation(reason)	void	Cancel booking
+ generateSummary()	String	Create call summary
+ attachDocument(document)	void	Add medical document
+ getRecording()	String	Retrieve recording URL
+ submitFeedback(rating, comment)	void	Provide rating

G. HealthRecord - Patient medical records and history

1. Attributes:

Name	Type	Modifier	Description
recordId	String	PK	Unique record ID
patientId	String	FK	Associated patient
doctorId	String	FK	Doctor who created record
consultationId	String	FK	Related consultation
diagnosis	String		Medical diagnosis
symptoms	String[]		Reported symptoms
medications	List<Prescription>		Current medications
labResults	List<LabResult>		Test results
vitalSigns	VitalSigns		Blood pressure, temperature, etc
notes	String		Clinical notes
attachedDocuments	List<Document>		Supporting documents
createdAt	DateTime		Record creation date
updatedAt	DateTime		Last modification date

2. Methods:

Method Name	Return Type	Description
+ createRecord(data)	void	Initialize new record
+ updateRecord(data)	void	Modify record
+ getRecordDetails()	RecordDetails	Retrieve all data
+ addDiagnosis(diagnosis)	void	Add diagnosis
+ addLabResults(results)	void	Add test results
+ generateSummary()	String	Create medical summary
+ syncOfflineData()	void	Synchronize with cloud
+ encryptData()	void	Encrypt sensitive data
+ exportToFile(format)	Document	Export as PDF/XML

H. Prescription - Medication prescription issued by doctor

1. Attributes:

Name	Type	Modifier	Description
prescriptionId	String	PK	Unique prescription ID
consultationId	String	FK	Related consultation
patientId	String	FK	Prescribed to
doctorId	String	FK	Issued by
medicines	List<Medicine>		List of medicines
dosage	String		Medicine dosage strength
duration	String		Treatment duration (e.g., 10 days)
frequency	String		Taking frequency (e.g., twice daily)
instructions	String		Special instructions
issuedDate	DateTime		When prescription created
expiryDate	DateTime		When prescription expires
status	String		Status (Active/Expired/Completed)
medicineQuantities	Map		Quantity per medicine

2. Methods:

Method Name	Return Type	Description
+ **issuePrescription()**	void	Create and save prescription
+ **getAvailableMedicines()**	List<Medicine>	Check which medicines available
+ **notifyPharmacies()**	void	Alert pharmacies
+ **trackMedicineAvailability()**	void	Monitor stock
+ **generatePrescriptionPDF()**	Document	Create PDF version
+ **validatePrescription()**	Boolean	Verify all data
+ **updatePrescription(data)**	void	Modify prescription
+ **expirePrescription()**	void	Mark as expired

I. Medicine - Medicine/drug information

1. Attributes:

Name	Type	Modifier	Description
medicineId	String	PK	Unique medicine ID
name	String		Brand name
genericName	String		Generic name
manufacturer	String		Manufacturing company
strength	String		Dosage strength (mg/ml)
form	String		Form (Tablet/Capsule/Liquid)
category	String		Medicine category
price	Decimal		Base price
sideEffects	String[]		Known side effects
contraindications	String[]		When not to use
dosageInfo	String		Recommended dosage
expiryDate	Date		Shelf life/expiry

2. Methods:

Method Name	Return Type	Description
+ getMedicineDetails()	MedicineDetails	Get complete info
+ checkAvailability()	Boolean	Check if in stock
+ getSubstitutes()	List<Medicine>	Find alternative medicines
+ getPrice()	Decimal	Retrieve current price
+ validateDosage(dosage)	Boolean	Verify safe dosage
+ getSideEffects()	String[]	List side effects
+ checkInteractions(otherMedicines)	List<String>	Drug interactions

J. Pharmacy - Pharmacy/medicine store information

1. Attributes:

Name	Type	Modifier	Description
pharmacyId	String	PK	Unique pharmacy ID
name	String		Pharmacy name
address	String		Physical location
phone	String		Contact number
email	String		Email address
latitude	Float		GPS latitude
longitude	Float		GPS longitude
operatingHours	String		Open/close times
rating	Float		Customer rating
inventory	List<MedicineStock>		Available medicines
staff	List<PharmacyStaff>		Pharmacy employees
isRegistered	Boolean		System registration status

2. Methods:

Method Name	Return Type	Description
+ registerPharmacy(data)	void	Register in system
+ updateLocation(lat, lng)	void	Update GPS coordinates
+ updateInventory(medicineId, quantity)	void	Change stock
+ getMedicineStock(medicineId)	MedicineStock	Check availability
+ calculateDistance(patientLat, patientLng)	Float	Distance to patient
+ generateInventoryReport()	Report	Stock report
+ notifyStaff(notification)	void	Alert pharmacy staff
+ getOperatingHours()	String	Retrieve hours

2. CLASS RELATIONSHIPS

Inheritance (Generalization)

- **Patient extends User** - Patient is a type of User
- **Doctor extends User** - Doctor is a type of User
- **PharmacyStaff extends User** - Pharmacy staff is a type of User
- **Administrator extends User** - Administrator is a type of User

Associations (Has-A Relationships)

From Class	To Class	Multiplicity	Description
Patient	HealthRecord	1..*	Patient has many health records
Patient	Consultation	1..*	Patient books many consultations
Doctor	Consultation	1..*	Doctor conducts many consultations
Consultation	HealthRecord	1..1	Consultation generates health record
Consultation	Prescription	0..1	Consultation may result in prescription
Doctor	Prescription	1..*	Doctor writes many prescriptions
Prescription	Medicine	1..*	Prescription contains many medicines
Pharmacy	MedicineStock	1..*	Pharmacy stocks many medicines
MedicineStock	Medicine	1..1	Stock tracks one medicine
Pharmacy	PharmacyStaff	1..*	Pharmacy employs many staff
Patient	SymptomChecker	1..*	Patient uses symptom checker
User	Notification	1..*	User receives notifications
HealthRecord	Document	1..*	Record has supporting documents

3. MULTIPLICITY NOTATION

Notation	**Meaning**
1	Exactly one
0..1	Zero or one
*****	Zero or more
1..*	One or more
n	Exactly n elements
0..n	Zero to n elements