

Part 1 - Analysis, Architecture Design and Detailed Design

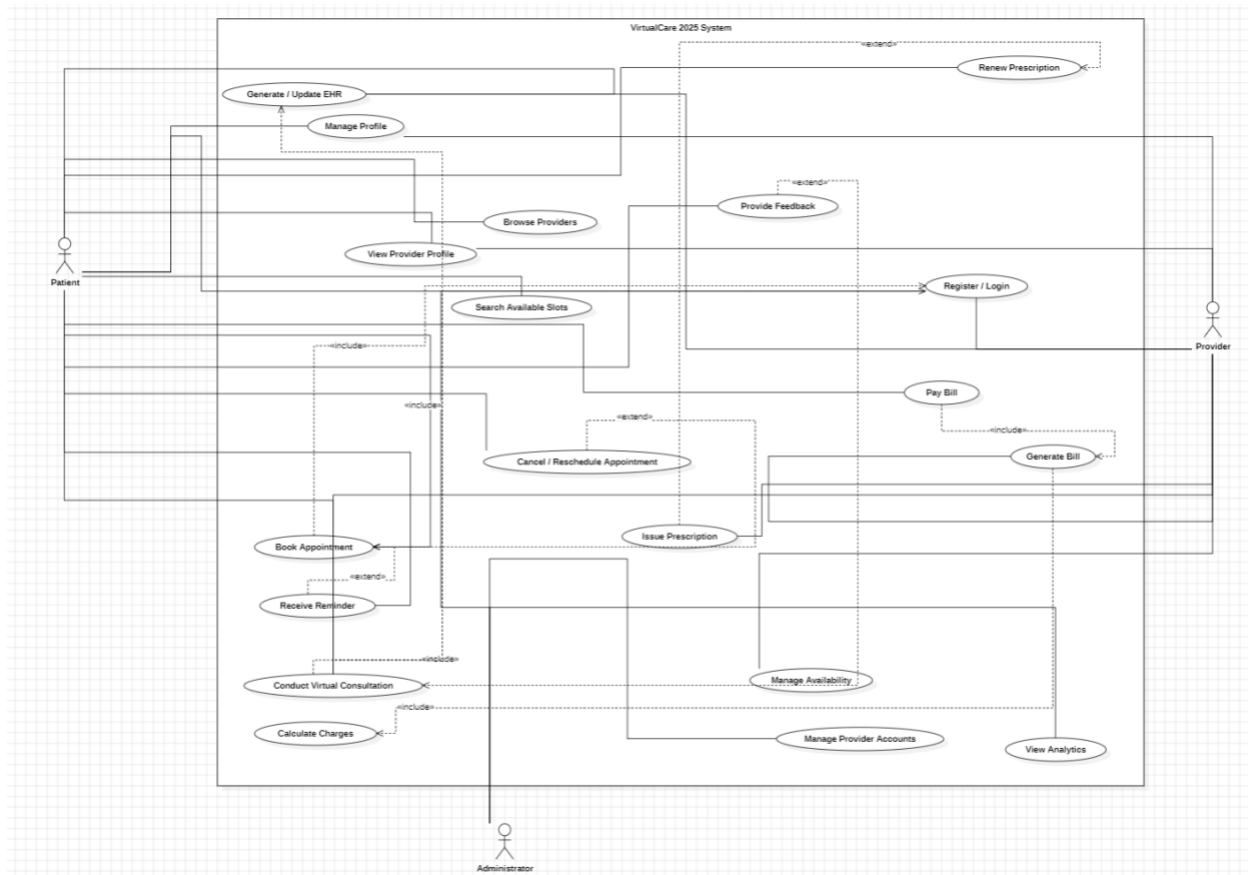


Figure 1: Use Case Diagram for Virtual Healthcare System (VHS)

Use Case Relationships:

- Actors ↔ Use Cases
- Patient → Register / Login
- Patient → Manage Profile
- Patient → Browse Providers
- Patient → View Provider Profile
- Patient → Search Available Slots
- Patient → Book Appointment
- Patient → Cancel / Reschedule Appointment
- Patient → Receive Reminder
- Patient → Conduct Virtual Consultation

- Patient → Generate / Update EHR
- Patient → Renew Prescription
- Patient → Pay Bill
- Patient → Provide Feedback
- Provider → Register / Login
- Provider → Manage Profile
- Provider → View Provider Profile
- Provider → Manage Availability
- Provider → Conduct Virtual Consultation
- Provider → Generate / Update EHR
- Provider → Issue Prescription
- Provider → Generate Bill
- Admin → Register / Login
- Admin → Manage Provider Accounts
- Admin → View Analytics

Include Relationships:

- Book Appointment → <<include>> Register / Login
- Conduct Virtual Consultation → <<include>> Register / Login
- Conduct Virtual Consultation → <<include>> Generate / Update EHR
- Generate Bill → <<include>> Calculate Charges
- Pay Bill → <<include>> Generate Bill

Extend Relationships:

- Cancel / Reschedule Appointment → <<extend>> Book Appointment
- Receive Reminder → <<extend>> Book Appointment
- Provide Feedback → <<extend>> Conduct Virtual Consultation
- Renew Prescription → <<extend>> Issue Prescription

Use Case Diagram — Figure Explanation:

Figure 1: Virtual Care 2025 Use Case Diagram

The use case diagram illustrates the main system actors (*Patient*, *Provider*, and *Admin*) and their interactions with the Virtual Healthcare System. It identifies all functional behaviors derived from the system scenario, showing how each actor engages with the system's services.

All standard UML elements — system boundary, actors, and use cases — are represented with clear associations and meaningful naming conventions.



1. **User** → inherits → Patient
2. **User** → inherits → Provider
3. **User** → inherits → Admin
4. **Patient** → 1 to many (0..*) → Appointment

5. **Provider** → 1 to many (0..*) → Appointment
6. **Patient** → 1 to 1 → EHR
7. **Provider** → 1 to many (0..*) → Prescription
8. **Patient** → 1 to many (0..*) → Bill
9. **Bill** → 1 to 1 → Payment
10. **Appointment** → 1 to 1 → Bill
11. EHR *-- Prescription (**Composition**)

Class Diagram — Figure Explanation:

Figure 2: VirtualCare 2025 Class Diagram

The class diagram illustrates the main entities of the Virtual Healthcare System and their relationships.

It demonstrates **inheritance** from the User superclass (specialized into Patient, Provider, and Admin), as well as clear **associations**, **composition**, and **cardinalities** that reflect real-world healthcare interactions.

- The composition between EHR and Prescription indicates that prescriptions are integral parts of a patient's health record and cannot exist independently.
- One-to-many associations (e.g., between Patient and Appointment, or Provider and Prescription) capture the system's data multiplicity accurately.
- Each class name and attribute uses intuitive terminology aligned with the healthcare domain.

Class-to-Subsystem Mapping:

- **Auth Service:** User, Admin
- **Appointment Service:** Appointment, Patient, Provider
- **EHR Service:** EHR, Prescription
- **Billing Service:** Bill, Payment

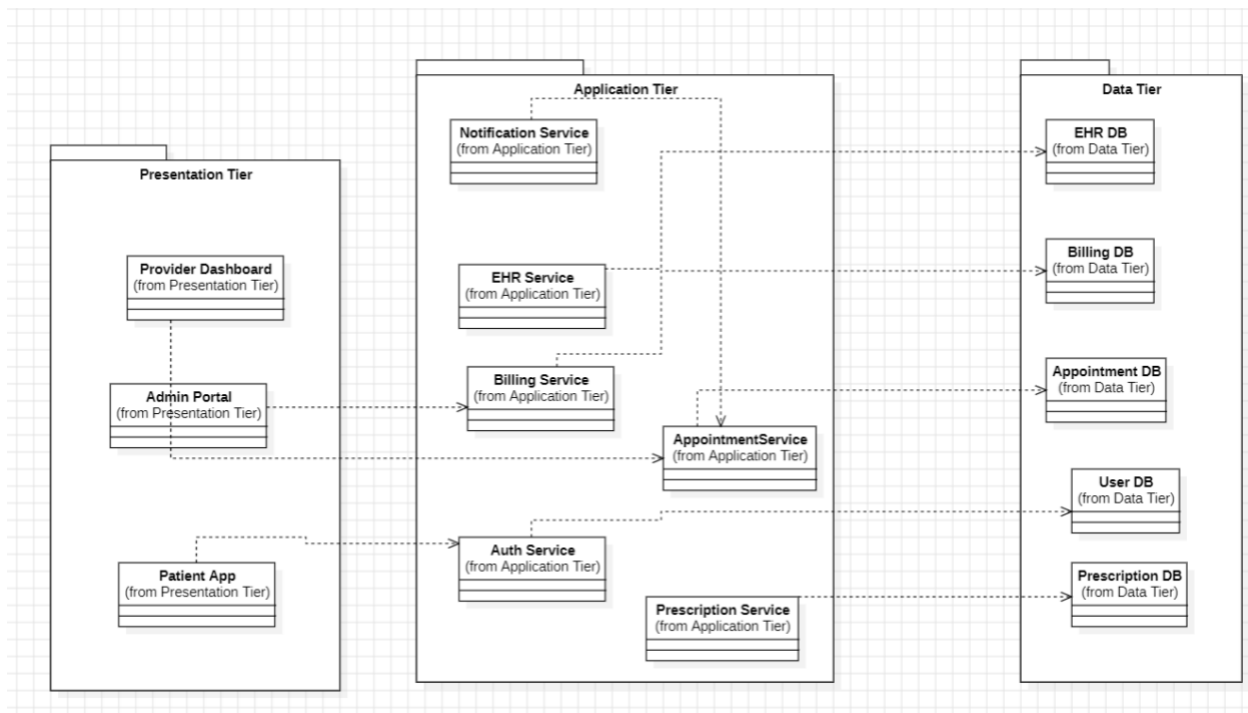


Figure 3: 3 Tier Architecture Model for Virtual Healthcare System (VHS)

Dependencies:

1. Patient App → Auth Service
2. Provider Dashboard → Appointment Service
3. Admin Portal → Billing Service
4. Auth Service → User DataBase
5. Appointment Service → Appointment DataBase
6. EHR Service → HER DataBase
7. Prescription Service → Prescription DataBase
8. Billing Service → Billing DataBase
9. Notification Service → Appointment Service

Architecture Diagram — Figure Explanation:

Figure 3: VirtualCare 2025 Architecture Model:

The system follows a **three-tier architecture** composed of:

1. Presentation Layer — Interfaces for Patients (mobile/web app), Providers (dashboard), and Admins (portal).
2. Application/Logic Layer — Core backend services including Authentication, Appointment, Consultation, EHR, Prescription, Billing, and Notification Services.
3. Data Layer — Databases storing users, appointments, EHR records, prescriptions, and billing data.

Dependencies flow **top-down** from the presentation layer to the data layer, maintaining modularity and clear separation of concerns. Each service interacts only with its respective database, ensuring encapsulation.

Security Controls:

- **Authentication** – Users log in with unique, secure credentials
- **Authorization (RBAC)** – Role-based access for Patients, Providers, and Admins
- **Encryption** – Sensitive data is encrypted during storage and transmission
- **Audit Logging & Data Privacy** – All access to health data is logged, and only authorized roles can view EHRs.

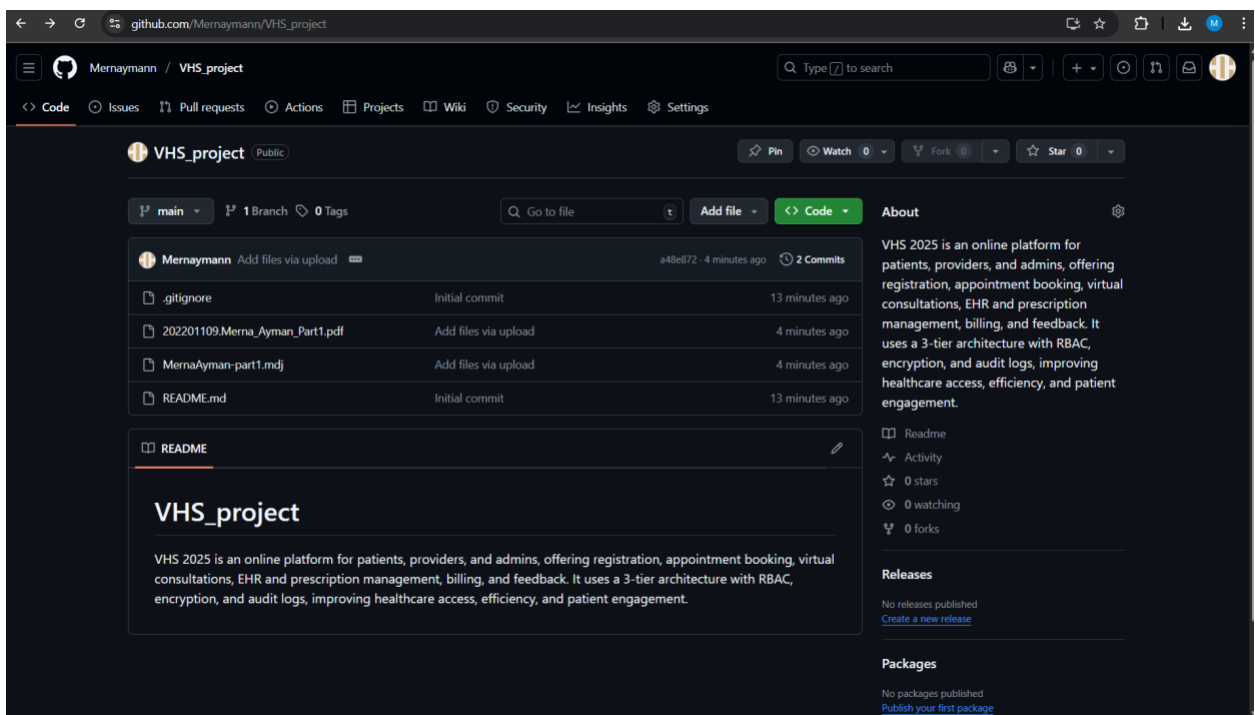


Figure4.1: Screenshot attached showing commits

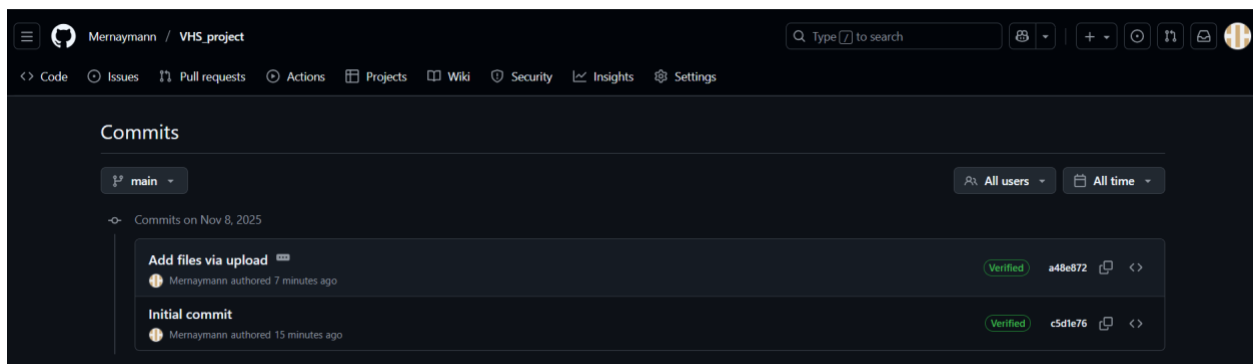


Figure4.2: Screenshot attached showing commits

Link: https://github.com/Mernaymann/VHS_project.git