System Design Document (SDD) - [Project Name: Healthcare SaaS Platform]

1. Introduction

1.1. Purpose

This System Design Document (SDD) outlines the design for a multi-tenant SaaS platform aimed at extending the reach of small to medium-sized healthcare providers (SMEs) to a broader patient base. The platform offers a cost-effective digital solution, enhances patient convenience through online appointment management, digital prescriptions, and teleconsultations, and improves customer satisfaction and market share for healthcare providers. The initial focus is on outpatient services, with a scalable framework to incorporate additional services and stakeholders in the future, and will also include AI-driven diagnosis.

1.2. Scope

This document covers the design and development of a multi-tenant SaaS application designed to facilitate patient registration, medical profile creation, appointment booking (both in-person and virtual), and online payment processing. Healthcare providers manage service listings, onboard/manage practitioners, manage appointments, upload prescriptions, and generate reports. Doctors view appointments, update medical histories, and create prescriptions. Patient data will be fetched, inserted, and updated within the client's EHR system without being stored within the application's cloud platform, ensuring data privacy and regulatory compliance (HIPAA, GDPR). Patients will have access to an AI-driven symptom checker.

1.3. Goals and Objectives

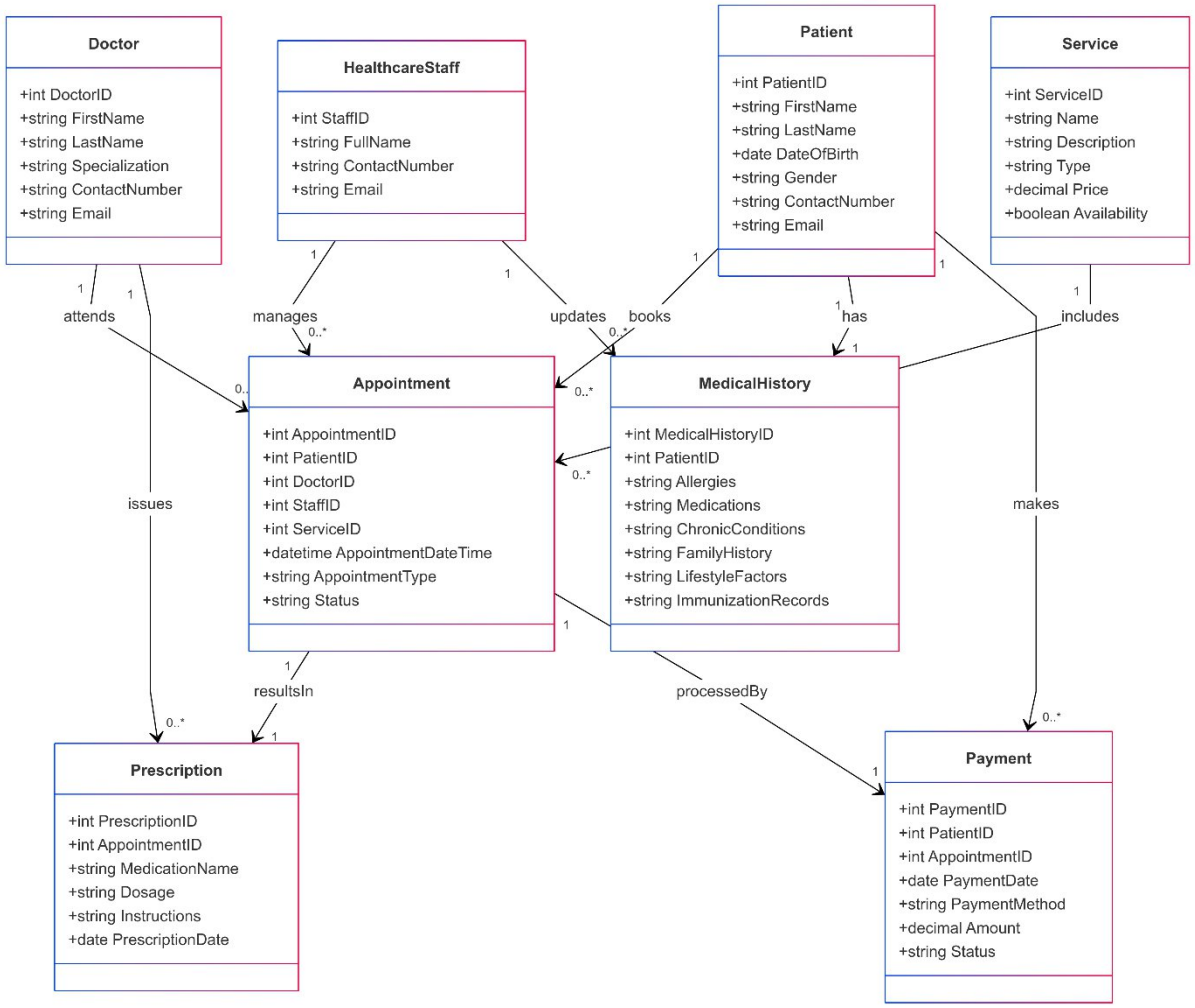
- Provide a user-friendly platform for patients to manage their healthcare needs.
- Enable healthcare providers to efficiently manage their services and patient interactions.
- Ensure the security and privacy of patient data in compliance with relevant regulations.
- Create a scalable architecture to accommodate future growth and feature additions.
- Improve patient outcomes through AI-driven symptom analysis and timely access to care.

2. Definitions, Acronyms & Abbreviations

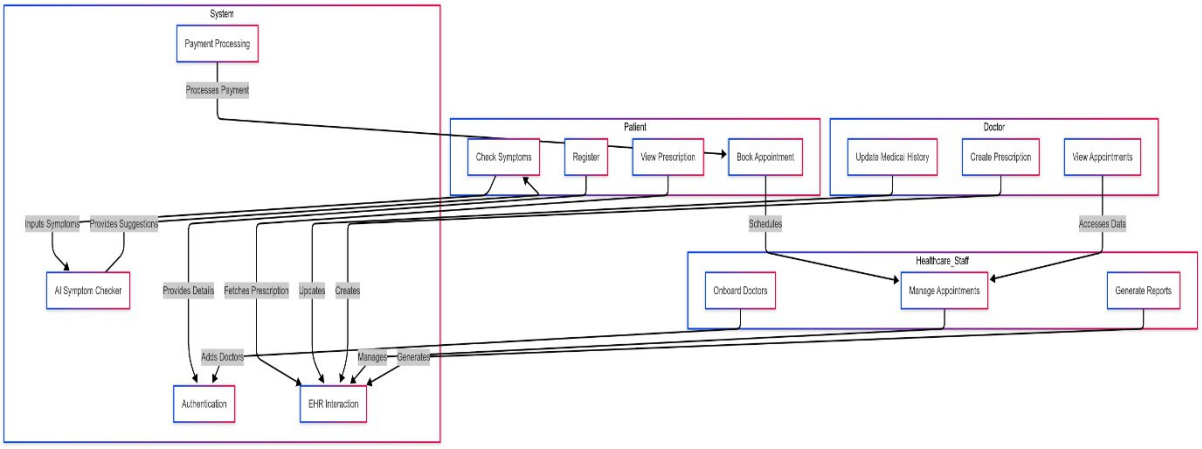
Term/Acronym	Definition
EHR	Electronic Health Record
HIPAA	Health Insurance Portability and Accountability Act
GDPR	General Data Protection Regulation
HL7	Health Level Seven

Term/Acronym	Definition
FHIR	Fast Healthcare Interoperability Resources
PHI	Protected Health Information
SaaS	Software as a Service
SME	Small to Medium Enterprise
CRUD	Create, Read, Update, Delete
IPD	Inpatient Department
API	Application Programming Interface
TLS	Transport Layer Security
HTTPS	Hypertext Transfer Protocol Secure
RBAC	Role-Based Access Control
AI	Artificial Intelligence

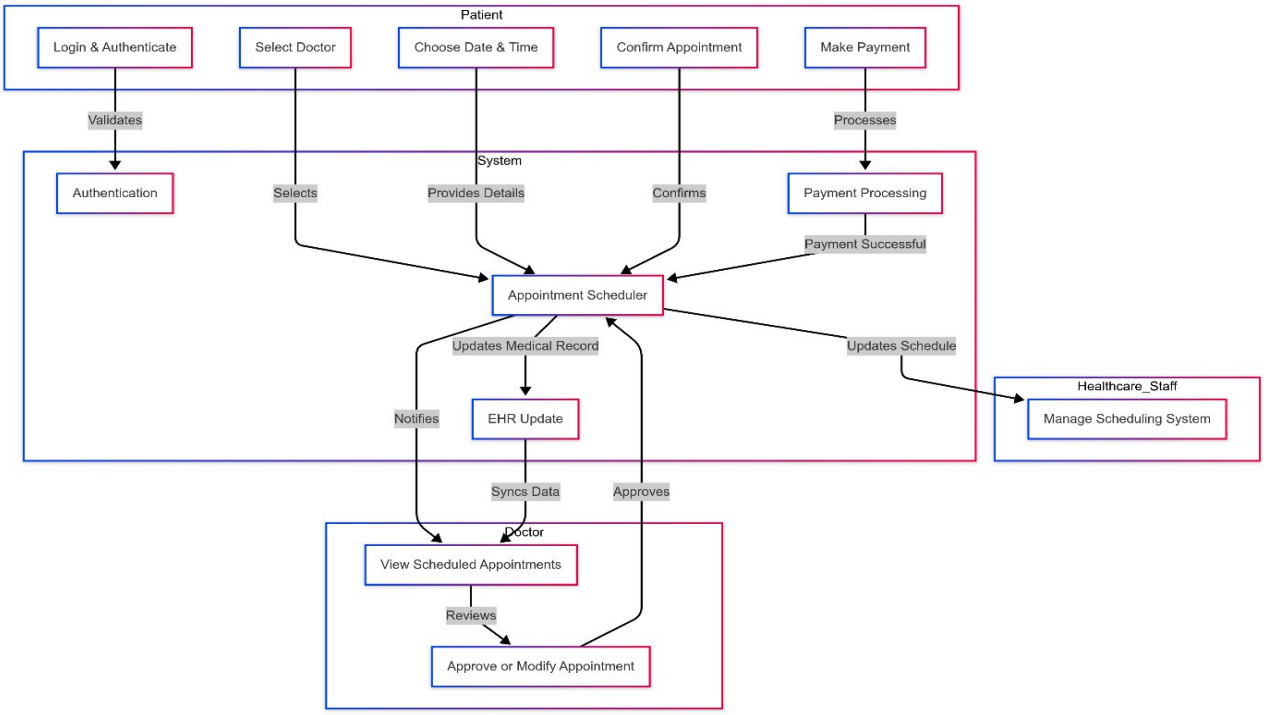
3. System Overview



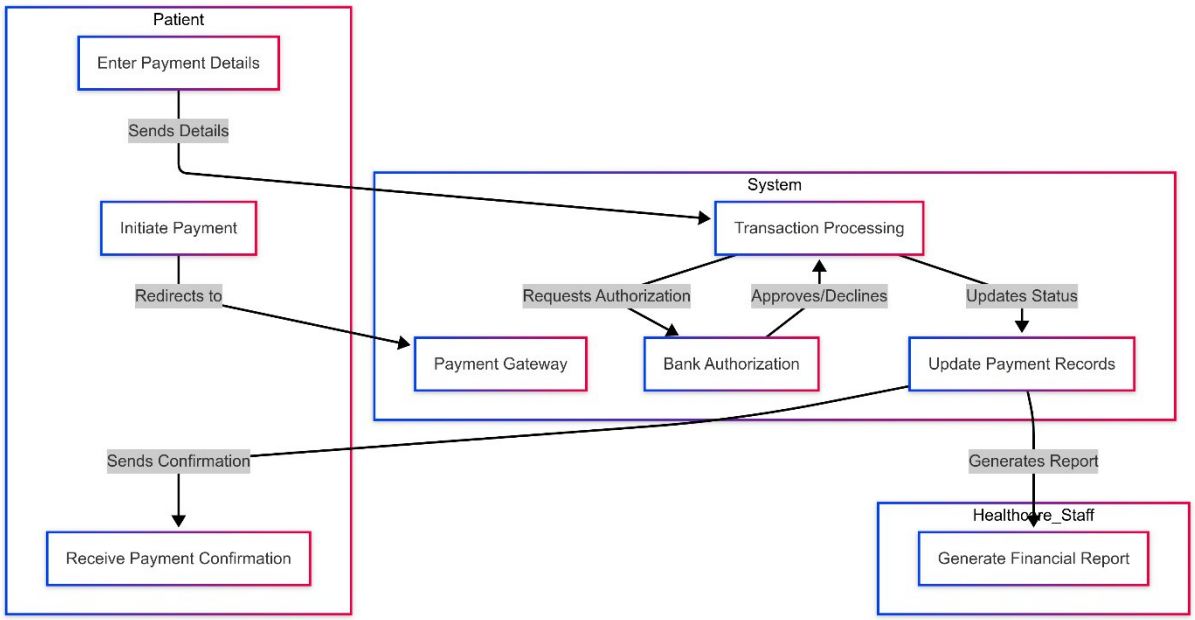
UML DIAGRAM



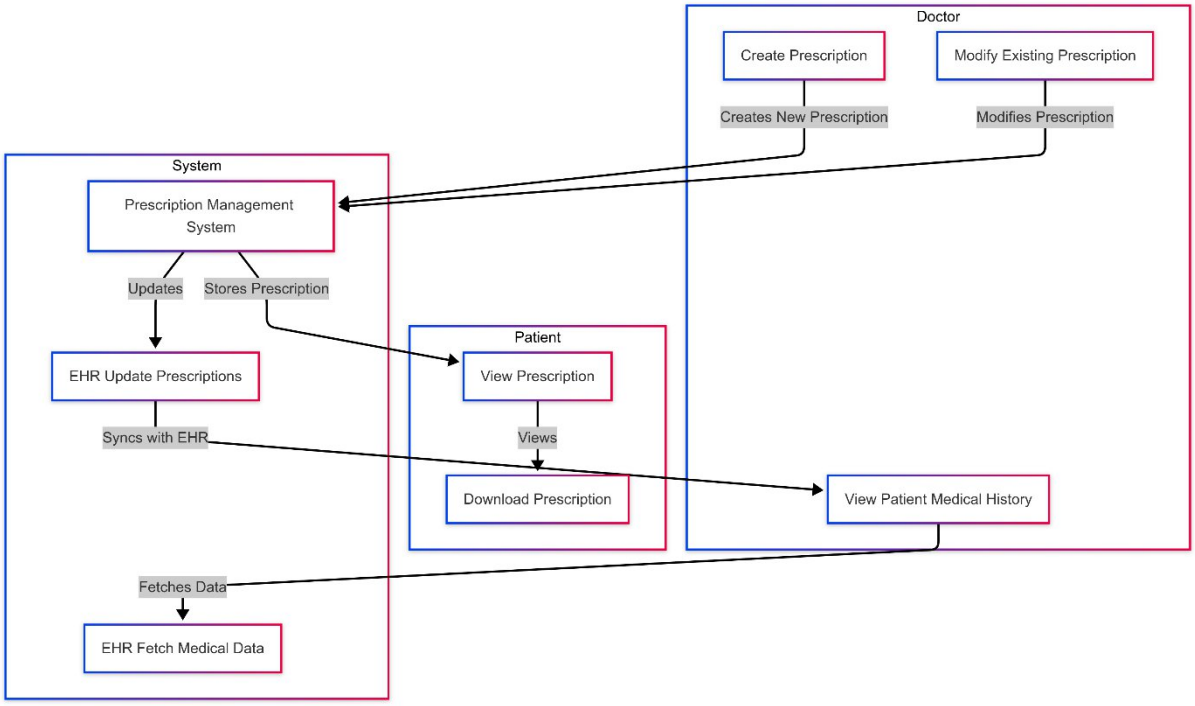
Data Flow Diagram (DFD) - Level 1



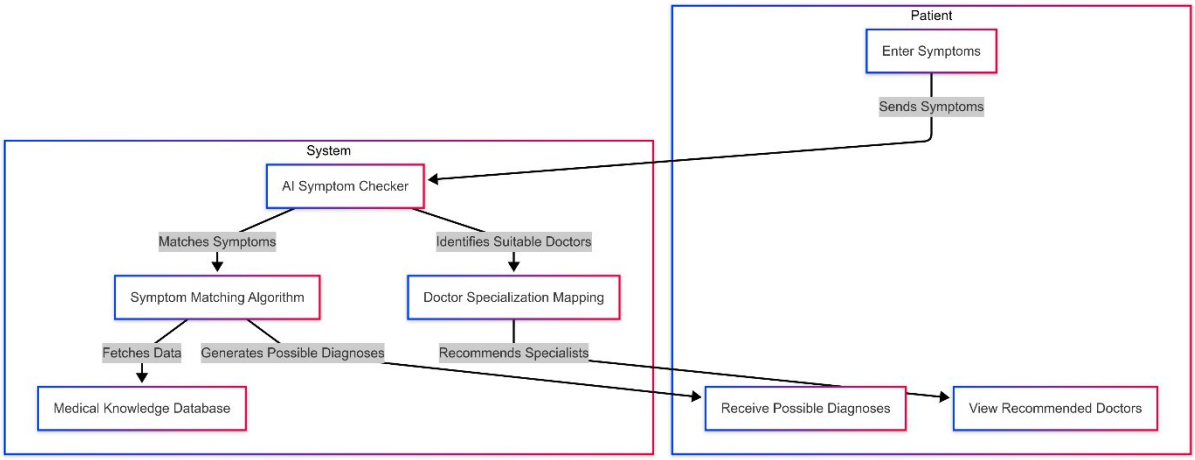
Data Flow Diagram (DFD) - Level 2 (Appointment Booking)



Data Flow Diagram (DFD) - Level 2 (Payment Processing)



Data Flow Diagram (DFD) - Level 2 (Prescription Handling)



Data Flow Diagram (DFD) - Level 2 (AI Symptom Checker)

3.1. Description

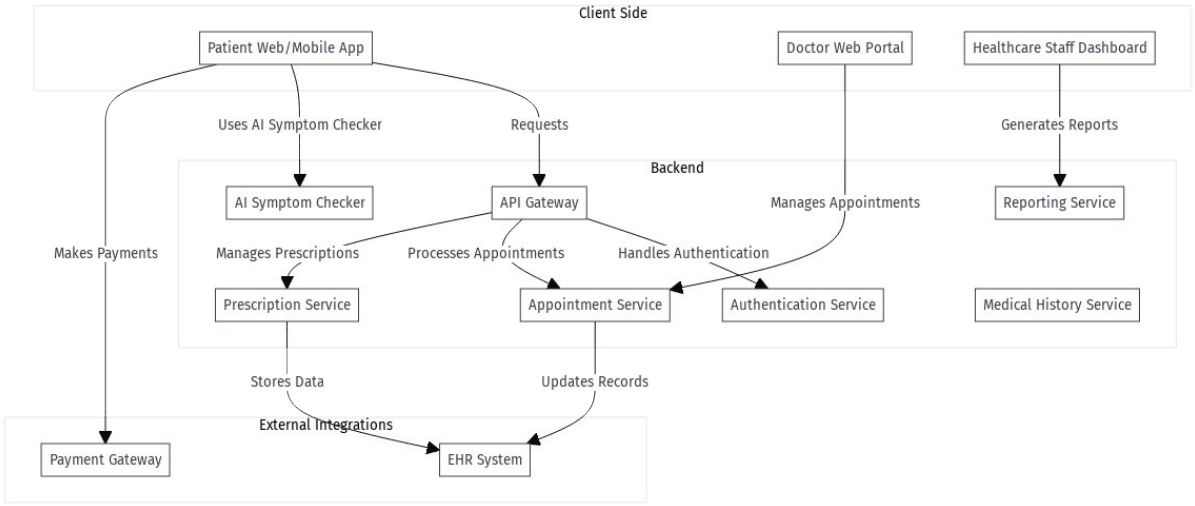
The Healthcare SaaS Platform is a multi-tenant SaaS application designed to streamline patient registration, appointment management, EHR integration, and provide AI-driven symptom checking capabilities.

3.2. Key Features

- Patient Registration
- Appointment Management
- Medical Profile Management
- Prescription Management
- Service Management
- Practitioner Management
- Reporting
- Payment Processing
- AI-Driven Symptom Checker
- EHR Interactions

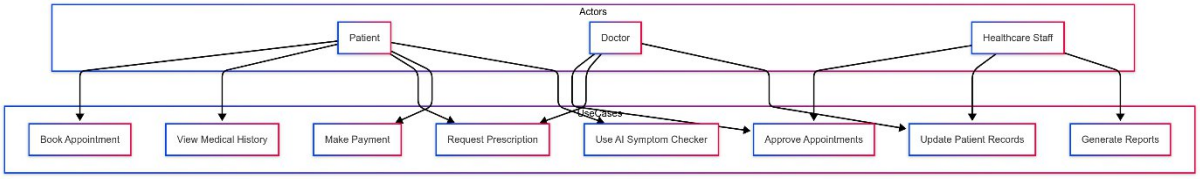
3.3. Architecture

The platform employs a multi-tenant SaaS architecture where the data of each service provider is kept secure and distinct. Patient medical records (EHR) are maintained within the client's facility. The application fetches, inserts, and updates EHR data but does not store the data in the application's multi-tenant cloud platform.

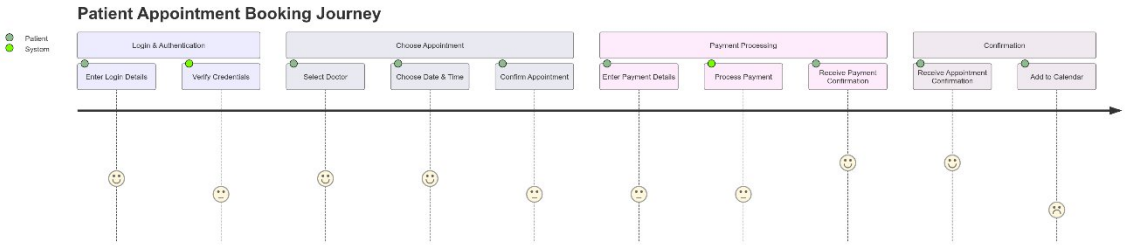


System Architecture Diagram

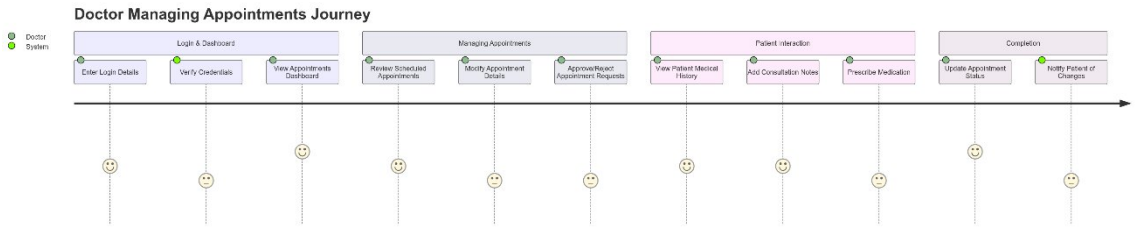
4. Functional Requirements



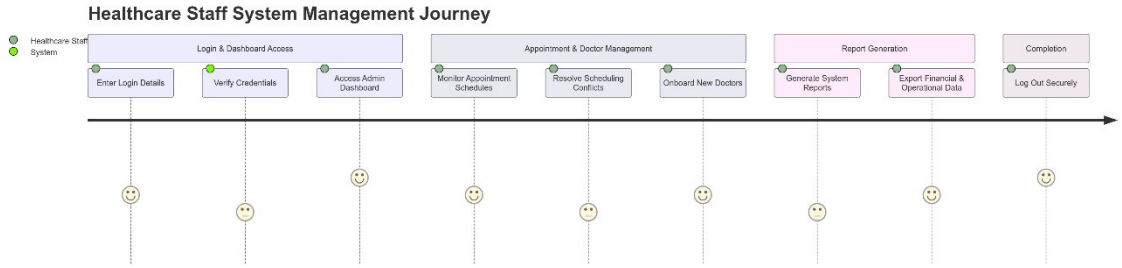
Use Case Diagram - Healthcare SaaS Platform



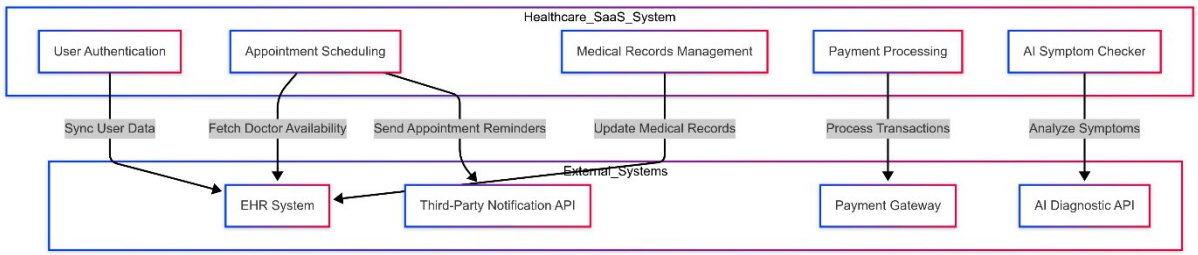
User Journey - Patient Booking an Appointment



User Journey - Doctor Managing Appointments



User Journey - Healthcare Staff Managing System



External System Integration Flow - Healthcare SaaS

4.1. Patient Registration

- Allow patients to register and create medical profiles.
- Automate the registration process.

4.2. Appointment Management

- Enable patients to book in-person or virtual appointments.
- Allow patients to reschedule appointments (with a cutoff, e.g., 24 hours prior).
- Enable healthcare staff to delete appointments.

- Enable doctors to modify appointments.
- Allow patients to book follow-up visits.
- Enable doctors to view their appointments.
- Function to update appointment status.

4.3. Medical Profile Management

- Enable patients to upload past medical history.
- Enable doctors and healthcare staff to update patient medical history.

4.4. Prescription Management

- Enable doctors and healthcare staff to create and modify prescriptions.
- Enable patients to view and download prescriptions.
- Enable doctors to generate a printout of the prescription for the patient.

4.5. Service Management

- Enable healthcare providers to manage their services list.
- Include service name, description, type, price, and availability.

4.6. Practitioner Management

- Enable healthcare providers to onboard and remove healthcare practitioners.

4.7. Reporting

- Enable healthcare staff to generate management reports (financial, operational, customer data).

4.8. Payment Processing

- Process online payments through multiple channels.

4.9. AI-Driven Symptom Checker

- Provide an AI-driven symptom checker for patients to analyze symptoms and suggest possible causes and the type of doctor to consult.

4.10. EHR Interactions

- Fetch, insert, and update patient data within the client's EHR system.

5. Non-Functional Requirements

5.1. Security

- Comply with HIPAA and GDPR regulations.
- Implement access controls to limit PHI access to authorized users only.
- Encrypt PHI at rest and in transit.
- Maintain audit logs of all activities related to PHI.
- Secure authentication processes.

- Role-based access control (Guest, Patient, Healthcare Staff, Doctor, System Admin).

5.2. Performance

- Ensure minimal clicks and assistance are needed for patients to navigate the app.
- Provide high user experience journeys.

5.3. Scalability

- Design the framework to allow faster addition of newer services and more stakeholder journeys.
- The application should efficiently cover 80% of the SME in the healthcare provider market of any metropolitan city.

5.4. Data Privacy

- Patient data will be maintained in the client facility EHR.
- The application will not store patient data in its multi-tenant cloud platform.

5.5. Multi-tenancy

- The application is hosted on a multi-tenant SaaS platform.
- Data of each service provider is kept secure and distinct from other providers.

5.6. Compliance

- All authentication processes must comply with relevant data protection laws (e.g., HIPAA, GDPR).

5.7. Data Validation

- Implement data validation rules to ensure data integrity and consistency.

6. API Design

6.1. API Endpoints

Endpoint	Description	Method	Request Body	Response Body
/patients	Manages patient records. Supports CRUD operations.	GET/POST/PUT/DELETE	Patient object (JSON)	Patient object (JSON) or list of patient objects (JSON)
/doctors	Manages doctor records. Supports CRUD operations.	GET/POST/PUT/DELETE	Doctor object (JSON)	Doctor object (JSON) or list of doctor objects

Endpoint	Description	Method	Request Body	Response Body
				(JSON)
/appointments	Manages appointment records. Supports CRUD operations.	GET/POST/PUT/DELETE	Appointment object (JSON)	Appointment object (JSON) or list of appointment objects (JSON)
/services	Manages service listings. Supports CRUD operations.	GET/POST/PUT/DELETE	Service object (JSON)	Service object (JSON) or list of service objects (JSON)
/medical-history/{id}	Retrieves/Updates medical history for a given patient ID.	GET/PUT	N/A (GET), Medical History object (JSON) (PUT)	Medical History object (JSON)
/prescriptions	Manages prescriptions. Supports CRUD operations.	GET/POST/PUT/DELETE	Prescription object (JSON)	Prescription object (JSON) or list of prescription objects (JSON)
/symptoms/check	Accepts patient-reported symptoms and returns possible causes and suggested doctor types (from AI Symptom Checker).	POST	Symptom List (JSON)	List of possible causes and suggested doctor types (JSON)
/ehr/patients/{id}	Retrieves patient	GET	N/A	Patient data from

Endpoint	Description	Method	Request Body	Response Body
	information from the EHR system.			EHR (Format depends on EHR system - HL7/FHIR)

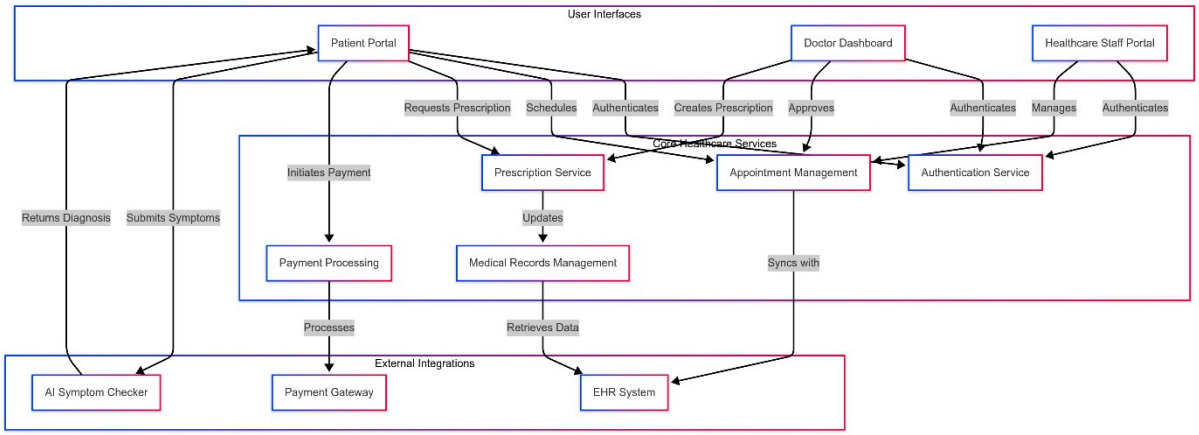
6.2. Protocols

- RESTful APIs (JSON payloads)
- HL7 (for EHR integration)
- FHIR (for EHR integration - preferred)

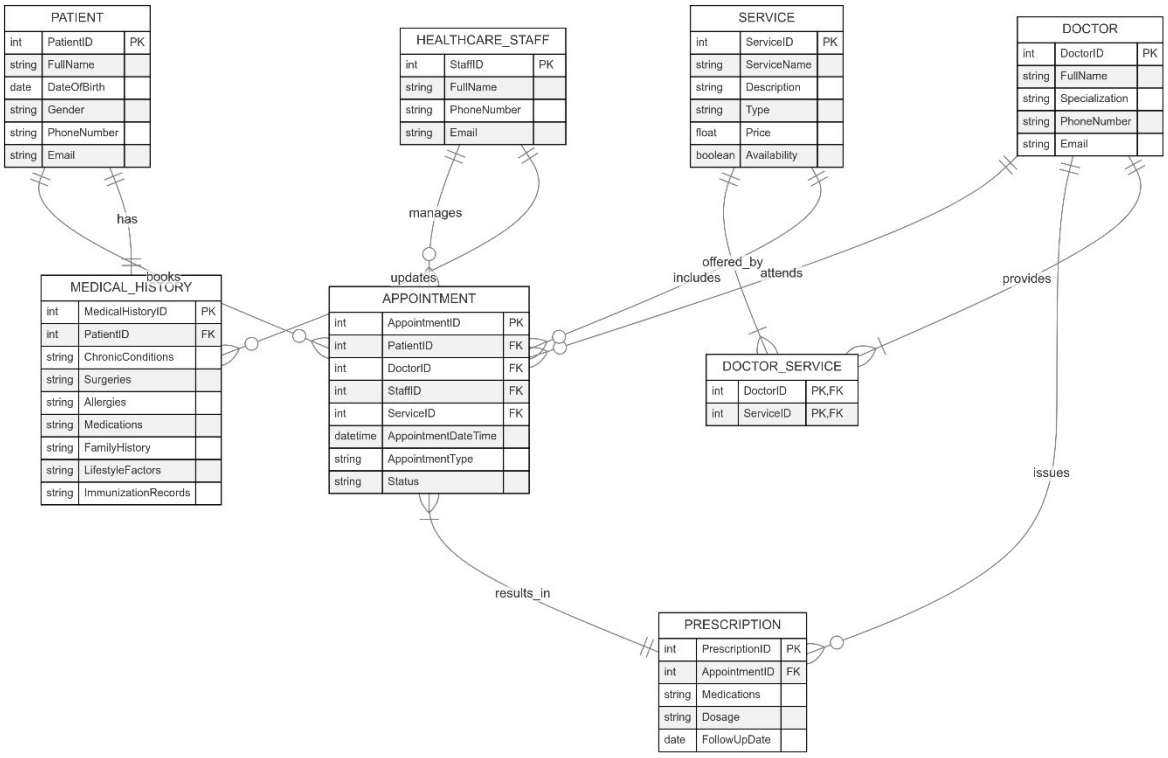
6.3. Security

- OAuth 2.0 (or similar) for authentication and authorization.
- Compliance with HIPAA and GDPR for data exchange.
- API keys for service providers to track usage and prevent abuse. Rate limiting will also be implemented.

7. System Architecture



Component Diagram - Healthcare SaaS Platform



Entity Relationship Diagram (ERD) for Healthcare Application

8. Security & Compliance

8.1. Regulations

- HIPAA
- GDPR

8.2. Access Control

- Role-based access control (RBAC) with the following roles: Guest, Patient, Healthcare Staff, Doctor, System Admin. Each role will have specific permissions assigned.

8.3. Data Encryption

- PHI encryption at rest and in transit.

- **At Rest:** AES-256 encryption for data stored in the database and file storage.
- **In Transit:** TLS 1.3 or higher for all communications between client and server, and between internal services.

8.4. Audit Logging

- Maintain comprehensive audit logs of all activities related to PHI, including access, modification, and deletion. Logs will include timestamps, user IDs, and details of the actions performed.

8.5. Authentication

- Secure authentication processes using multi-factor authentication (MFA) where possible.
- Strong password policies enforced.
- Regular security audits and penetration testing.

8.6. EHR Compliance

- Data exchange complies with HIPAA and GDPR regulations through secure and compliant APIs (HL7 or FHIR).
- Data mapping and transformation to ensure accurate and compliant data transfer between the platform and EHR systems.

9. Data Encryption Strategy

9.1. Data at Rest

- Encryption of PHI stored in databases and file systems using AES-256 encryption.
- Database encryption will be implemented using Transparent Data Encryption (TDE) where supported by the database system.

9.2. Data in Transit

- Encryption of PHI during transmission between application components and external systems (e.g., EHR systems) using protocols like HTTPS and TLS 1.3 or higher.
- Virtual Private Networks (VPNs) or secure tunnels for connections to external EHR systems where applicable.

9.3. Key Management

- Secure storage and management of encryption keys using a Hardware Security Module (HSM) or a secure key management service.
- Key rotation policies to ensure keys are regularly updated.
- Access control policies to restrict access to encryption keys to authorized personnel only.

10. API Endpoints (Detailed Specification)

See Section 6.1. This section provides a high-level overview. Detailed API specifications (including request/response schemas, error codes, and authentication requirements) will be documented separately in an API documentation portal (e.g., using Swagger/OpenAPI). Example below:

Example: /patients (POST)

- **Description:** Creates a new patient record.
- **Request Body:**

```
{
 "FirstName": "John",
 "LastName": "Doe",
 "DateOfBirth": "1990-01-01",
 "Gender": "Male",
 "ContactNumber": "555-123-4567",
 "Email": "john.doe@example.com"
}
```

- **Response Body (Success - 201 Created):**

```
{
 "PatientID": 123,
 "FirstName": "John",
 "LastName": "Doe",
 "DateOfBirth": "1990-01-01",
 "Gender": "Male",
 "ContactNumber": "555-123-4567",
 "Email": "john.doe@example.com",
 "RegistrationDate": "2024-01-01"
}
```

- **Response Body (Error - 400 Bad Request):**

```
{
 "error": "Invalid request body",
 "message": "DateOfBirth is invalid"
}
```

11. Appendices

11.1. Glossary

Term	Definition
Authentication	The process of verifying the identity of a user, device, or system.
Authorization	The process of determining what actions a user, device, or system is allowed to perform.
Encryption	The process of converting data into an unreadable format to protect its confidentiality.
Multi-Factor Authentication (MFA)	An authentication method that requires the user to provide two or more verification factors to gain access to a resource.
RESTful API	An API that conforms to the architectural constraints of the Representational State Transfer (REST) style.
SaaS	Software as a service, a software distribution model in which a third-party provider hosts applications and makes them available to customers over the Internet.
Tokenization	Replacing sensitive data with non-sensitive substitute values, called tokens.

11.2. Data Dictionary

Field Name	Entity	Data Type	Description	Constraints
PatientID	Patient	INT	Unique identifier for the patient.	Primary Key, Auto-Incrementing
FirstName	Patient	VARCHAR	Patient's first name.	Not Null, Max Length: 50
LastName	Patient	VARCHAR	Patient's last name.	Not Null, Max Length: 50
DateOfBirth	Patient	DATE	Patient's date of birth.	Not Null
Gender	Patient	VARCHAR	Patient's gender.	Max Length: 20
ContactNumber	Patient	VARCHAR	Patient's contact	Max Length:

Field Name	Entity	Data Type	Description	Constraints
			phone number.	20
Email	Patient	VARCHAR	Patient's email address.	Max Length: 100, Unique
RegistrationDate	Patient	DATE	Date when the patient registered on the platform.	Not Null
MedicalHistoryID	Patient	INT	Foreign key referencing the MedicalHistory table.	Foreign Key
DoctorID	Doctor	INT	Unique identifier for the doctor.	Primary Key, Auto-Incrementing
Specialization	Doctor	VARCHAR	Doctor's area of specialization.	Max Length: 100
HealthcareProviderID	Doctor	INT	Foreign key referencing the HealthcareProvider table.	Foreign Key
AppointmentID	Appointment	INT	Unique identifier for the appointment.	Primary Key, Auto-Incrementing
AppointmentDateTime	Appointment	DATETIME	Date and time of the appointment.	Not Null
AppointmentType	Appointment	VARCHAR	Type of appointment (In-person/Virtual).	Max Length: 20
Status	Appointment	VARCHAR	Status of the appointment (Scheduled, Completed, Cancelled).	Max Length: 20
MedicationName	Prescription	VARCHAR	Name of the	Not Null,

Field Name	Entity	Data Type	Description	Constraints
			prescribed medication.	Max Length: 100
Dosage	Prescription	VARCHAR	Dosage instructions for the medication.	Not Null, Max Length: 100
HealthcareProviderID	Service	INT	Foreign key referencing the HealthcareProvider table.	Foreign Key
Name	Service	VARCHAR	Name of the service offered.	Not Null, Max Length: 100
Type	Service	VARCHAR	Type of service.	Max Length: 50
Price	Service	DECIMAL	Price of the service.	Not Null

11.3. Sample API Requests

1. Get Patient by ID:

- Endpoint: /patients/{id}
- Method: GET
- Request:

```
GET /patients/123 HTTP/1.1
Host: api.healthcaresaas.com
Authorization: Bearer <token>
```

- Response:

```
{
 "PatientID": 123,
 "FirstName": "John",
 "LastName": "Doe",
 "DateOfBirth": "1990-01-01",
 "Gender": "Male",
 "ContactNumber": "555-123-4567",
```

"Email": "john.doe@example.com",
"RegistrationDate": "2023-01-15",
"MedicalHistoryID": 456
}

2. Create Appointment:

- Endpoint: /appointments
- Method: POST
- Request:

```
POST /appointments HTTP/1.1
Host: api.healthcaresaas.com
Authorization: Bearer <token>
Content-Type: application/json

{
"PatientID": 123,
"DoctorID": 789,
"AppointmentDateTime": "2024-02-10T10:00:00",
"AppointmentType": "Virtual",
"Notes": "Follow-up consultation"
}
```

- Response:

{
"AppointmentID": 101,
"PatientID": 123,
"DoctorID": 789,
"AppointmentDateTime": "2024-02-10T10:00:00",
"AppointmentType": "Virtual",
"Status": "Scheduled",
"Notes": "Follow-up consultation"
}

11.4. Risk Assessment

Risk ID	Risk Description	Potential Impact	Likelihood	Severity	Mitigation Strategy
R1	Data breach exposing PHI	Legal penalties, reputational damage, loss of patient trust	Medium	High	Implement strong access controls, data encryption, regular security audits, and intrusion detection systems.
R2	System downtime affecting patient care	Disruption of services, delayed appointments, potential harm to patients	Low	High	Implement redundancy and failover mechanisms, robust monitoring, and disaster recovery plan.
R3	Non-compliance with HIPAA/GDPR	Legal penalties, fines, reputational damage	Medium	High	Implement compliance policies and procedures, regular training for staff, and ongoing monitoring.
R4	Vulnerabilities in third-party APIs (e.g., payment gateway)	Data leakage, financial loss, service disruption	Medium	Medium	Conduct thorough security assessments of third-party vendors, implement strong authentication and authorization, and monitor API usage.
R5	Insider threat (malicious or negligent employee)	Data theft, data alteration, system sabotage	Low	Medium	Implement background checks, access control policies, regular security awareness training, and monitoring of employee activity.
R6	AI Symptom	Misdiagnosis,	Low	High	Disclaimers

Risk ID	Risk Description	Potential Impact	Likelihood	Severity	Mitigation Strategy
	Checker provides incorrect diagnosis	incorrect treatment, potential harm to patients			stating AI provides possible causes, suggest consulting a doctor, not a replacement for professional medical advice, continuous monitoring and improvement of AI model

11.5. Security Incident Response Plan

1. Purpose:

This Security Incident Response Plan (SIRP) outlines the procedures for identifying, responding to, and recovering from security incidents affecting the Healthcare SaaS Platform.

2. Scope:

This plan applies to all systems, applications, and data associated with the Healthcare SaaS Platform, including but not limited to:

- Production environment
- Development and testing environments
- Cloud infrastructure
- Third-party integrations

3. Incident Response Team:

The Incident Response Team (IRT) is responsible for managing and coordinating the response to security incidents. The IRT consists of the following roles:

- **Incident Commander:** Overall responsibility for managing the incident response.
- **Technical Lead:** Leads the technical investigation and recovery efforts.
- **Communications Lead:** Manages internal and external communications.
- **Legal/Compliance Lead:** Ensures compliance with legal and regulatory requirements.

4. Incident Response Process:

The incident response process consists of the following phases:

- **Detection:** Identifying potential security incidents through monitoring, alerts, and user reports.

- **Analysis:** Assessing the scope and impact of the incident.
- **Containment:** Isolating affected systems and preventing further damage.
- **Eradication:** Removing the root cause of the incident.
- **Recovery:** Restoring affected systems and data to normal operation.
- **Post-Incident Activity:** Documenting the incident, analyzing the response, and implementing preventative measures.

5. Incident Reporting:

All suspected security incidents must be reported immediately to the IRT via [Contact Information].

6. Communication Plan:

The Communications Lead will be responsible for communicating with stakeholders, including:

- Internal staff
- Customers
- Law enforcement
- Regulatory agencies

7. Plan Maintenance:

This SIRP will be reviewed and updated at least annually, or more frequently as needed, to reflect changes in the threat landscape, system architecture, or regulatory requirements.

8. Contact Information:

- Incident Response Team Email: security@healthcaresaas.com
- Incident Response Team Phone: 555-SAFE-NOW

11.6. Data Models

{

"entities": [

{

"name": "Patient",

"attributes": [

{"name": "PatientID", "type": "INT", "primaryKey": true},

{"name": "FirstName", "type": "VARCHAR"},

{"name": "LastName", "type": "VARCHAR"},

{"name": "DateOfBirth", "type": "DATE"},

{"name": "Gender", "type": "VARCHAR"},

{"name": "ContactNumber", "type": "VARCHAR"},

{"name": "Email", "type": "VARCHAR"},
{"name": "RegistrationDate", "type": "DATE"},
{"name": "MedicalHistoryID", "type": "INT", "foreignKey": true}
]
},
{
"name": "Doctor",
"attributes": [
{"name": "DoctorID", "type": "INT", "primaryKey": true},
{"name": "FirstName", "type": "VARCHAR"},
{"name": "LastName", "type": "VARCHAR"},
{"name": "Specialization", "type": "VARCHAR"},
{"name": "ContactNumber", "type": "VARCHAR"},
{"name": "Email", "type": "VARCHAR"},
{"name": "HealthcareProviderID", "type": "INT", "foreignKey": true}
]
},
{
"name": "HealthcareProvider",
"attributes": [
{"name": "HealthcareProviderID", "type": "INT", "primaryKey": true},
{"name": "Name", "type": "VARCHAR"},
{"name": "Address", "type": "VARCHAR"},
{"name": "ContactNumber", "type": "VARCHAR"},
{"name": "Email", "type": "VARCHAR"}
]
},
{
"name": "Appointment",
"attributes": [
{"name": "AppointmentID", "type": "INT", "primaryKey": true},
{"name": "PatientID", "type": "INT", "foreignKey": true},

{"name": "DoctorID", "type": "INT", "foreignKey": true},
{"name": "AppointmentDateTime", "type": "DATETIME"},
{"name": "AppointmentType", "type": "VARCHAR"},
{"name": "Status", "type": "VARCHAR"},
{"name": "Notes", "type": "TEXT"}
]
},
{
"name": "Prescription",
"attributes": [
{"name": "PrescriptionID", "type": "INT", "primaryKey": true},
{"name": "PatientID", "type": "INT", "foreignKey": true},
{"name": "DoctorID", "type": "INT", "foreignKey": true},
{"name": "PrescriptionDate", "type": "DATE"},
{"name": "MedicationName", "type": "VARCHAR"},
{"name": "Dosage", "type": "VARCHAR"},
{"name": "Instructions", "type": "TEXT"}
]
},
{
"name": "Service",
"attributes": [
{"name": "ServiceID", "type": "INT", "primaryKey": true},
{"name": "HealthcareProviderID", "type": "INT", "foreignKey": true},
{"name": "Name", "type": "VARCHAR"},
{"name": "Description", "type": "TEXT"},
{"name": "Type", "type": "VARCHAR"},
{"name": "Price", "type": "DECIMAL"},
{"name": "Availability", "type": "VARCHAR"}
]
},
{

"name": "MedicalHistory",
"attributes": [
{"name": "MedicalHistoryID", "type": "INT", "primaryKey": true},
{"name": "PatientID", "type": "INT", "foreignKey": true},
{"name": "Allergies", "type": "TEXT"},
{"name": "Medications", "type": "TEXT"},
{"name": "Conditions", "type": "TEXT"},
{"name": "Notes", "type": "TEXT"},
{"name": "LastUpdated", "type": "DATETIME"}
]
},
{
"name": "Payment",
"attributes": [
{"name": "PaymentID", "type": "INT", "primaryKey": true},
{"name": "PatientID", "type": "INT", "foreignKey": true},
{"name": "AppointmentID", "type": "INT", "foreignKey": true},
{"name": "PaymentDate", "type": "DATE"},
{"name": "PaymentMethod", "type": "VARCHAR"},
{"name": "Amount", "type": "DECIMAL"},
{"name": "Status", "type": "VARCHAR"}
]
}
],
"relationships": [
{"from": "Patient", "to": "MedicalHistory", "type": "1:1"},
{"from": "Patient", "to": "Appointment", "type": "1:N"},
{"from": "Doctor", "to": "Appointment", "type": "1:N"},
{"from": "HealthcareProvider", "to": "Doctor", "type": "1:N"},
{"from": "HealthcareProvider", "to": "Service", "type": "1:N"},
{"from": "Patient", "to": "Prescription", "type": "1:N"},
{"from": "Doctor", "to": "Prescription", "type": "1:N"},

{"from": "Patient", "to": "Payment", "type": "1:N"},

{"from": "Appointment", "to": "Payment", "type": "1:1"}

]

}

11.7. Database Schema (Example - MySQL)

```
-- Patient Table
CREATE TABLE Patient (
 PatientID INT PRIMARY KEY AUTO_INCREMENT,
 FirstName VARCHAR(50) NOT NULL,
 LastName VARCHAR(50) NOT NULL,
 DateOfBirth DATE NOT NULL,
 Gender VARCHAR(20),
 ContactNumber VARCHAR(20),
 Email VARCHAR(100) UNIQUE,
 RegistrationDate DATE NOT NULL,
 MedicalHistoryID INT,
 FOREIGN KEY (MedicalHistoryID) REFERENCES MedicalHistory(MedicalHistoryID)
);

-- Doctor Table
CREATE TABLE Doctor (
 DoctorID INT PRIMARY KEY AUTO_INCREMENT,
 FirstName VARCHAR(50) NOT NULL,
 LastName VARCHAR(50) NOT NULL,
 Specialization VARCHAR(100),
 ContactNumber VARCHAR(20),
 Email VARCHAR(100),
 HealthcareProviderID INT,
 FOREIGN KEY (HealthcareProviderID) REFERENCES
HealthcareProvider(HealthcareProviderID)
);

-- HealthcareProvider Table
```

```
CREATE TABLE HealthcareProvider (
 HealthcareProviderID INT PRIMARY KEY AUTO_INCREMENT,
 Name VARCHAR(100) NOT NULL,
 Address VARCHAR(255),
 ContactNumber VARCHAR(20),
 Email VARCHAR(100)
);

-- Appointment Table
CREATE TABLE Appointment (
 AppointmentID INT PRIMARY KEY AUTO_INCREMENT,
 PatientID INT,
 DoctorID INT,
 AppointmentDateTime DATETIME NOT NULL,
 AppointmentType VARCHAR(20),
 Status VARCHAR(20),
 Notes TEXT,
 FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
 FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
);

-- Prescription Table
CREATE TABLE Prescription (
 PrescriptionID INT PRIMARY KEY AUTO_INCREMENT,
 PatientID INT,
 DoctorID INT,
 PrescriptionDate DATE,
 MedicationName VARCHAR(100) NOT NULL,
 Dosage VARCHAR(100) NOT NULL,
 Instructions TEXT,
 FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
 FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
```

```
);

-- Service Table
CREATE TABLE Service (
 ServiceID INT PRIMARY KEY AUTO_INCREMENT,
 HealthcareProviderID INT,
 Name VARCHAR(100) NOT NULL,
 Description TEXT,
 Type VARCHAR(50),
 Price DECIMAL(10, 2),
 Availability VARCHAR(255),
 FOREIGN KEY (HealthcareProviderID) REFERENCES
HealthcareProvider(HealthcareProviderID)
);

-- MedicalHistory Table
CREATE TABLE MedicalHistory (
 MedicalHistoryID INT PRIMARY KEY AUTO_INCREMENT,
 PatientID INT,
 Allergies TEXT,
 Medications TEXT,
 Conditions TEXT,
 Notes TEXT,
 LastUpdated DATETIME,
 FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

-- Payment Table
CREATE TABLE Payment (
 PaymentID INT PRIMARY KEY AUTO_INCREMENT,
 PatientID INT,
 AppointmentID INT,
 PaymentDate DATE,
```

```
 PaymentMethod VARCHAR(50),
 Amount DECIMAL(10, 2),
 Status VARCHAR(50),
 FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
 FOREIGN KEY (AppointmentID) REFERENCES Appointment(AppointmentID)
);
```