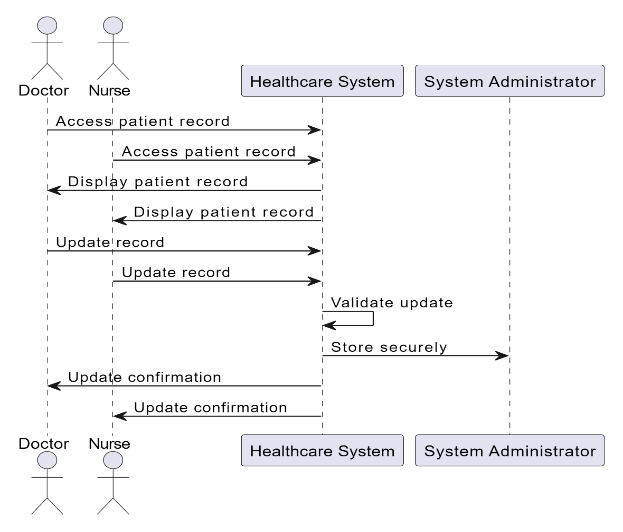
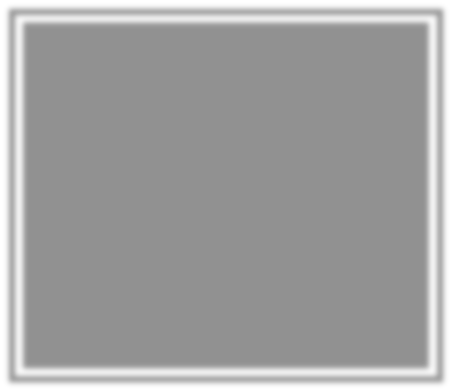
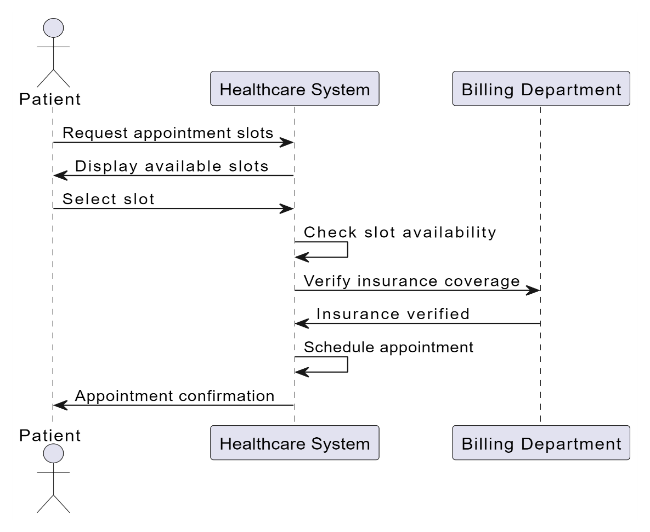
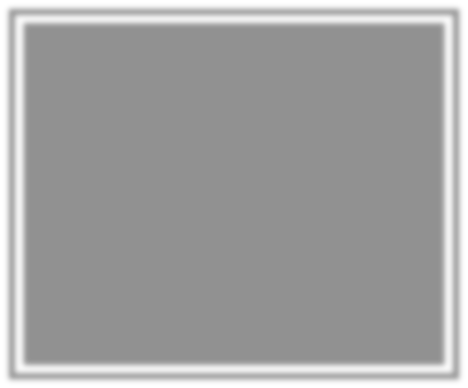
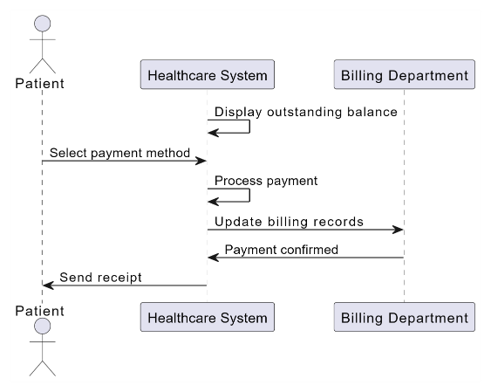
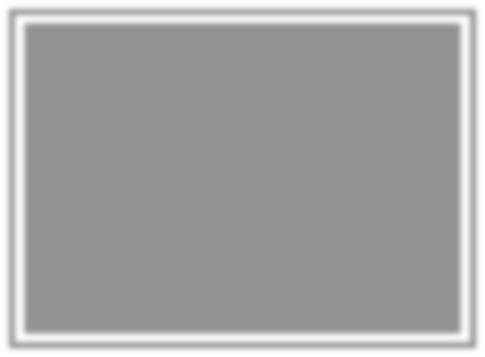
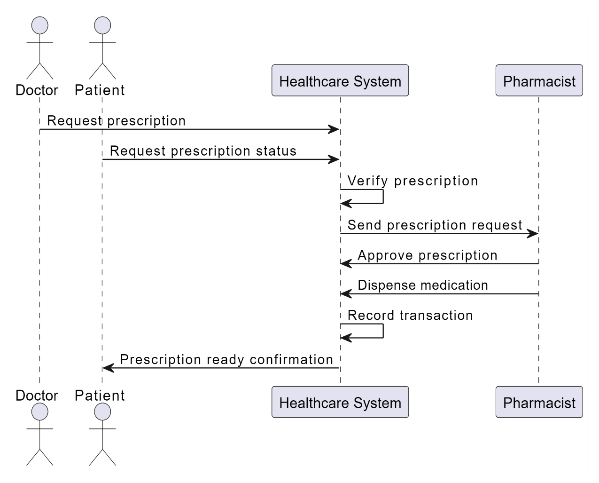
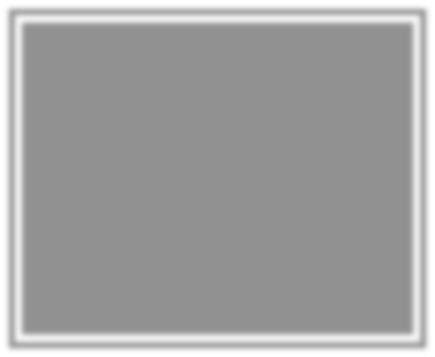


SSDs

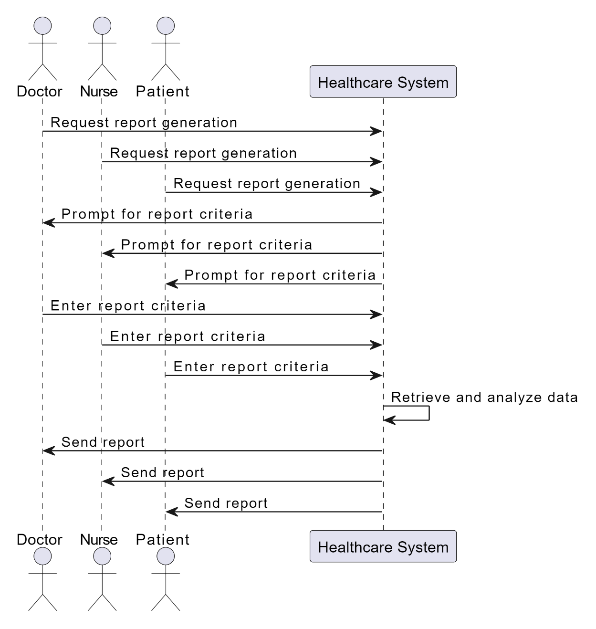
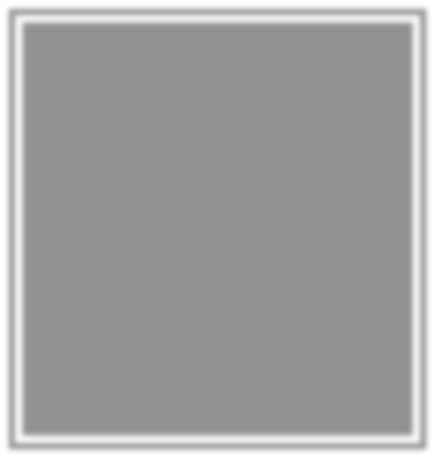
Schedule Appointment Update Medical Records



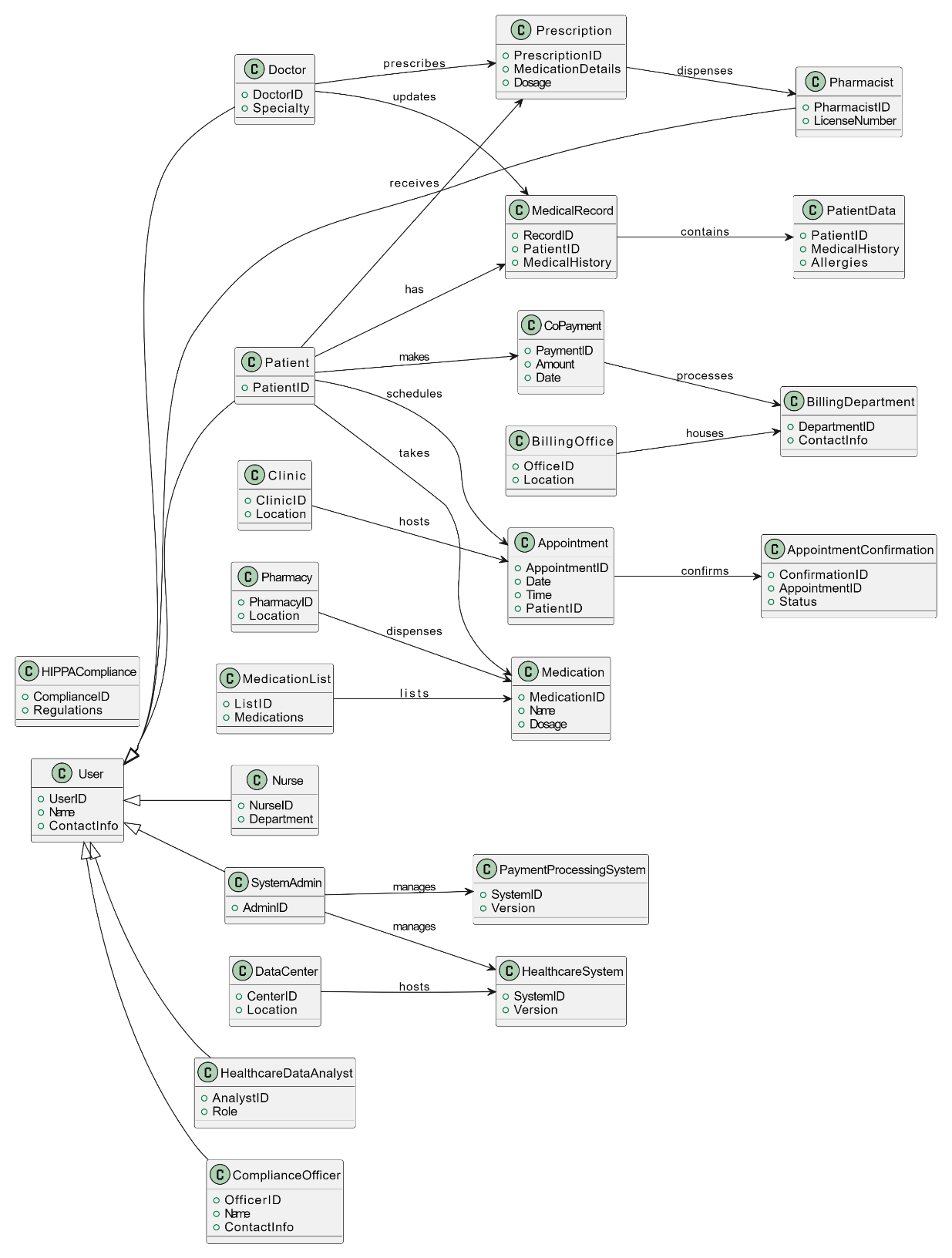
Dispense Medication Process Patient Co-Payment



Generate Healthcare Records



Domain Model



# System Operation Contract - 1

**Name:**

schedule Appointment(patientID, doctorID, appointmentDate, appointmentTime)

**Responsibilities:**

To schedule an appointment for a patient with a specified doctor at a given date and time.

**Type:**

Input

**Cross References:**

Use Case: Schedule Appointment

**Exceptions:**

* PatientIDNotFoundException
* DoctorIDNotFoundException
* TimeSlotUnavailableException

**Output:**

AppointmentConfirmation

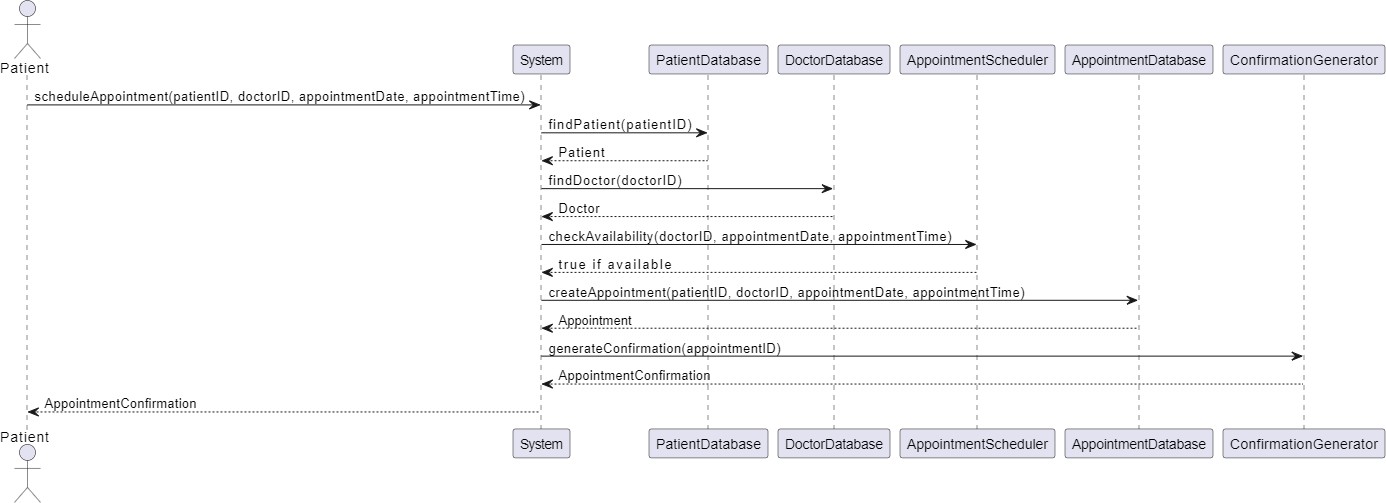
**Pre-conditions:**

* Patient and doctor exist in the system.
* The requested time slot is available.

**Post-conditions:**

* An Appointment object is created with the specified details.
* An AppointmentConfirmation object is generated and returned.

**Diagram:**



**System Operation Contract - 2**

**Name:**

updateMedicalRecord(patientID, recordID, medicalHistory)

**Responsibilities:**

To update the medical record of a specific patient.

**Type:**

Input

**Cross References:**

Use Case: Update Medical Record

**Exceptions:**

* PatientIDNotFoundException
* DoctorIDNotFoundException

**Output:**

None

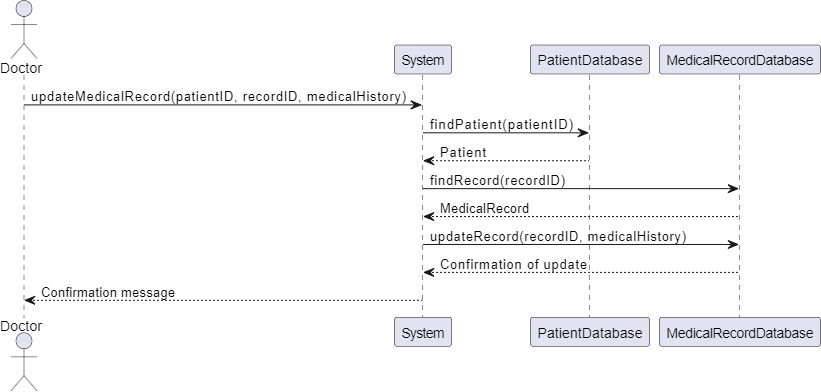
**Pre-conditions:**

* Patient and record exist in the system.

**Post-conditions:**

* The MedicalRecord object is updated with the new medical history.

**Diagram:**



# System Operation Contract - 3

**Name:**

processPayment(patientID, amount, paymentDate)

**Responsibilities:**

To process a payment made by a patient.

**Type:**

Input

**Cross References:**

Use Case: Process Payment

**Exceptions:**

* PatientIDNotFoundException
* InvalidPaymentException

**Output:**

PaymentReceipt

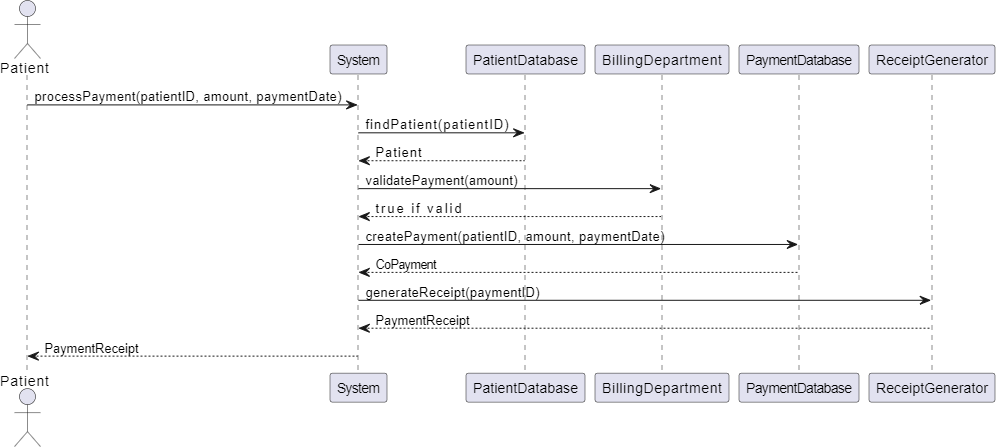
**Pre-conditions:**

* Patient exists in the system.
* The payment amount is valid.

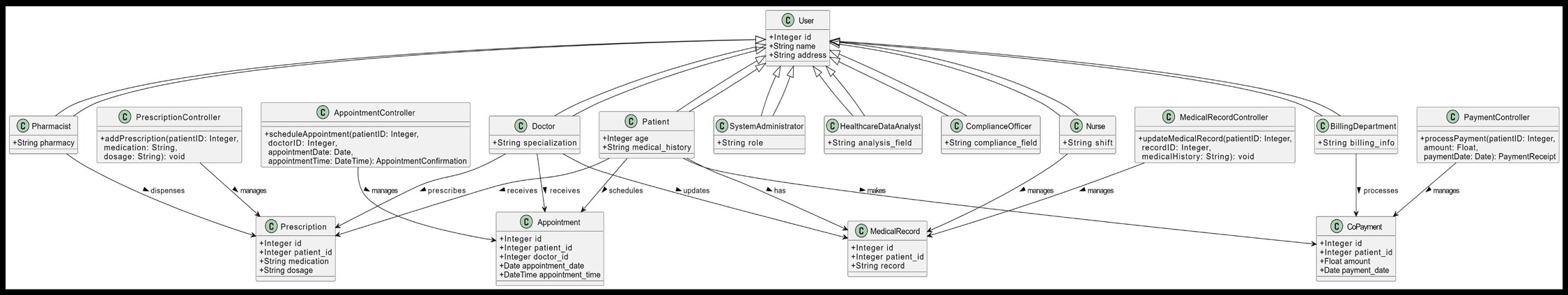
**Post-conditions:**

* A CoPayment object is created with the payment details.
* A PaymentReceipt object is generated and returned.

**Diagram:**



**Design Class Diagram**



**Rational for any domain classes and domain associations that are eliminated from the domain model, any Pure Fabrications and Indirections introduced, any polymorphism used, and any generalization and specialization:**

**Domain Classes and Associations Eliminated**

* + No Domain Classes or Associations were eliminated

# Pure Fabrications and Indirections

* + The controllers AppointmentController, MedicalRecordController, PaymentController, PrescriptionController were introduced to handle the business logic and can be considered Pure Fabrications and introduce avel of indirection by decoupling the user interface from the business logic.

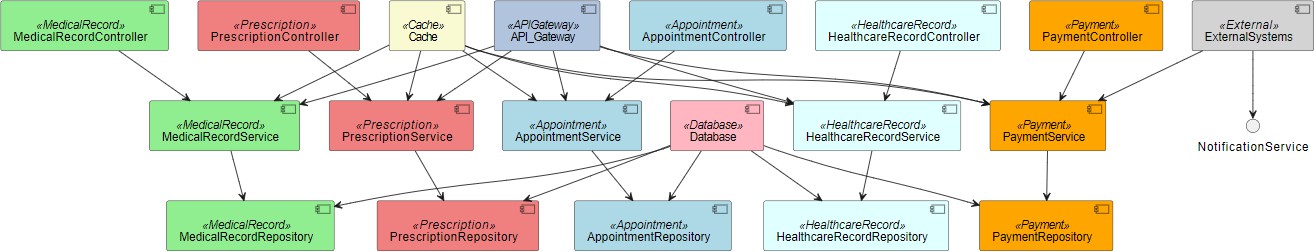
# Polymorphism Used

* + Polymorphism is utilized through the inheritance of the User class by speciﬁc user roles such as Patient, Doctor, Nurse, etc. This allows the User class to be deﬁned generically and extended by the speciﬁc subclasses.

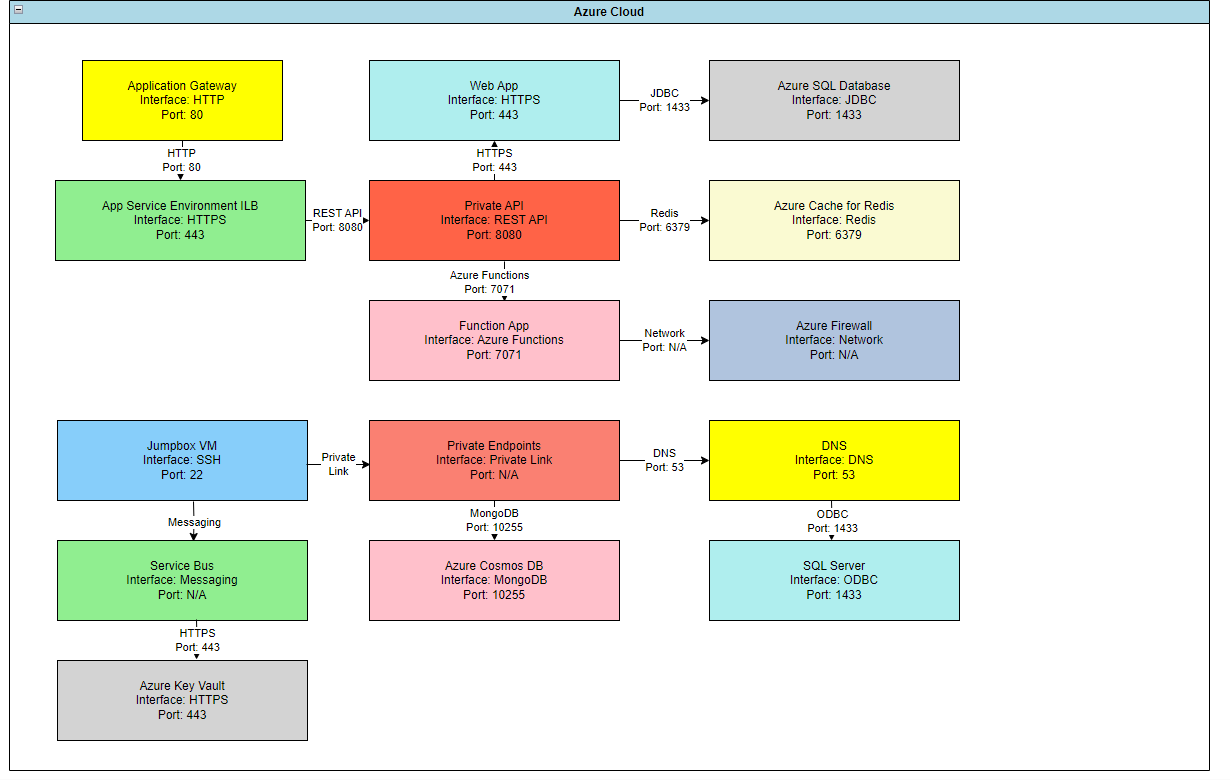
# Generalization and Specialization

* + The User class serves as a generalized class that contains the common attributes (id, name, address). Speciﬁc user roles (Patient, Doctor, Nurse, Pharmacist, BillingDepartment, SystemAdministrator, HealthcareDataAnalyst, ComplianceOfficer) are specialized subclasses that inherit from User. This approach minimizes duplication by consolidating common attributes.

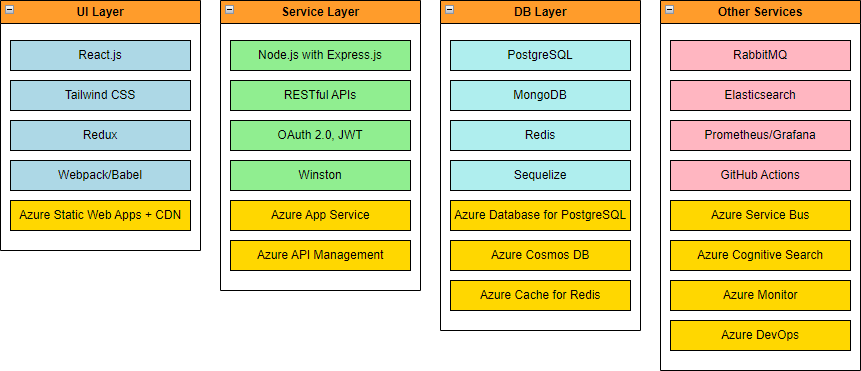
Component Diagram



Deployment Diagram



Tech Stack and Cloud Service Provider



## Rational for Choosing Speciﬁc Deployment Patterns and Component Structure UI Layer:

* React.js provides an efficient and ﬂexible way to build user interfaces.
* Tailwind CSS allows for rapid and customizable styling.
* Redux manages the application state effectively.
* Webpack/Babel ensures code is compatible across different environments.
* Azure Static Web Apps and CDN offer fast, globally distributed content delivery.

## Service Layer:

* Node.js with Express.js is perfect for building lightweight and efficient APIs.
* RESTful APIs are a standard for designing networked applications, making the service interactions

straightforward and reliable.

* OAuth 2.0 and JWT ensure secure authentication and authorization, protecting data.
* Winston helps with logging, which is crucial for tracking and troubleshooting issues in the application.
* Azure App Service provides a managed environment for deploying applications easily.
* Azure API Management helps secure, publish, and monitor our APIs effectively.

## DB Layer:

* PostgreSQL is a robust relational database, ideal for handling complex queries and transactions.
* MongoDB provides ﬂexibility for NoSQL data storage.
* Redis offers high-speed, in-memory caching to improve application performance.
* Sequelize simpliﬁes database management and interactions.
* Azure Database for PostgreSQL provides a managed, scalable, and secure relational database

service.

* Azure Cosmos DB offers a globally distributed, multi-model database service.
* Azure Cache for Redis enhances performance with a managed caching solution.

## Other Services:

* RabbitMQ provides reliable messaging between services.
* Elasticsearch enhances search capabilities with its powerful search and analytics engine.
* Prometheus/Grafana are essential for monitoring application performance and health.
* GitHub Actions streamline CI/CD processes.
* Azure Service Bus manages messaging between applications.
* Azure Cognitive Search improves data retrieval and search functionalities.
* Azure Monitor ensures application health.
* Azure DevOps supports the CI/CD pipeline.

## Scaling

**100 Concurrent Users or 100 TPS:** My current setup doesn’t require any modiﬁcation. The existing architecture can handle 100 users efficiently with built-in auto-scaling.

## 1,000 Concurrent Users or 1,000 TPS:

* **Service Layer**: Will need to increase instance count in Azure App Service or move to Azure Kubernetes Service (AKS).
* **DB Layer**: Will need to scale up Azure Database for PostgreSQL, add read replicas, and optimize Azure Cosmos DB, and enhance caching with Azure Cache for Redis.

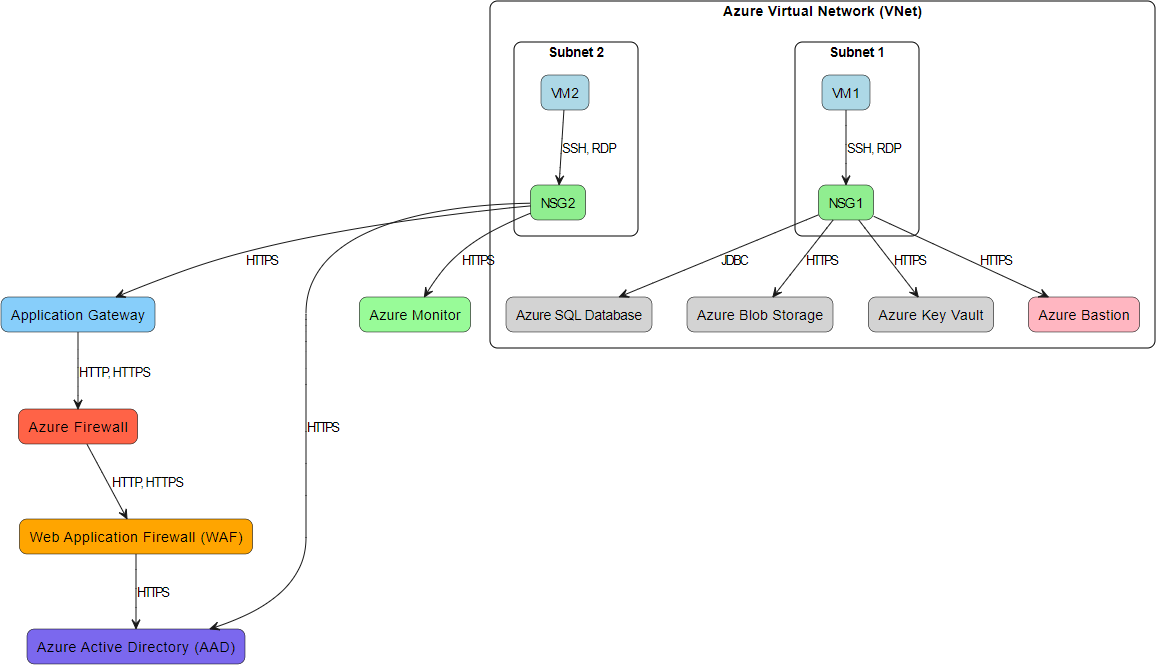
## 10,000 Concurrent Users or 3,000 TPS:

* **Service Layer**: Will need to transition fully to AKS.
* **DB Layer**: Will need to implement sharding in PostgreSQL and partition data in Cosmos DB and increase Redis instances.
* **Other Services**: Will need to use Azure Event Grid with Azure Service Bus for better event handling.

## 100,000 Concurrent Users

* **UI Layer**: Will need to use multiple CDN providers alongside Azure CDN.
* **Service Layer**: Will need to use AKS with auto-scaling and service mesh for traffic management.
* **DB Layer**: Will need to use advanced sharding and multi-region deployments for PostgreSQL and Cosmos DB as well as implement multi-tier caching with Redis and Azure Blob Storage.
* **Other Services**: Will need to implement distributed tracing and advanced monitoring with Azure Monitor and Log Analytics.

## Security Architecture



**IAC Template using Terraform**

provider "azurerm" { version = "=3.0.0" features {}

}

resource "azurerm\_resource\_group" "integris\_rg" {

name = "rg-integris-healthcare" location = "Oklahoma City"

}

resource "azurerm\_virtual\_network" "integris\_vnet" { name = "vnet-integris-healthcare" address\_space = ["10.0.0.0/16"]

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name

}

resource "azurerm\_subnet" "app\_subnet" { name = "subnet-app-integris"

resource\_group\_name = azurerm\_resource\_group.integris\_rg.name

virtual\_network\_name = azurerm\_virtual\_network.integris\_vnet.name address\_preﬁxes = ["10.0.1.0/24"]

}

resource "azurerm\_subnet" "db\_subnet" { name = "subnet-db-integris"

resource\_group\_name = azurerm\_resource\_group.integris\_rg.name virtual\_network\_name = azurerm\_virtual\_network.integris\_vnet.name address\_preﬁxes = ["10.0.2.0/24"]

}

resource "azurerm\_network\_security\_group" "app\_nsg" {

name = "nsg-app-integris"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name

security\_rule {

name = "SSH"

priority = 1001

direction = "Inbound"

access = "Allow"

protocol = "Tcp" source\_port\_range = "\*" destination\_port\_range = "22" source\_address\_preﬁx = "\*" destination\_address\_preﬁx = "\*"

}

}

resource "azurerm\_network\_security\_group" "db\_nsg" {

name = "nsg-db-integris"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name

security\_rule {

name = "HTTPS"

priority = 1001

direction = "Inbound"

access = "Allow"

protocol = "Tcp" source\_port\_range = "\*" destination\_port\_range = "443" source\_address\_preﬁx = "\*" destination\_address\_preﬁx = "\*"

}

}

resource "azurerm\_network\_interface" "app\_nic" {

name = "nic-app-integris"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name

ip\_conﬁguration {

name = "internal"

subnet\_id = azurerm\_subnet.app\_subnet.id private\_ip\_address\_allocation = "Dynamic"

}

}

resource "azurerm\_network\_interface" "db\_nic" {

name = "nic-db-integris"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name

ip\_conﬁguration {

name = "internal"

subnet\_id = azurerm\_subnet.db\_subnet.id private\_ip\_address\_allocation = "Dynamic"

}

}

resource "azurerm\_linux\_virtual\_machine" "app\_vm" {

name = "vm-app-integris"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name network\_interface\_ids = [azurerm\_network\_interface.app\_nic.id] size = "Standard\_DS1\_v2"

os\_disk {

name = "appOsDisk"

caching = "ReadWrite"

storage\_account\_type = "Premium\_LRS"

}

os\_proﬁle {

computer\_name = "app-server" admin\_username = "adminuser" admin\_password = "Password1234!"

}

os\_proﬁle\_linux\_conﬁg { disable\_password\_authentication = false

}

source\_image\_reference { publisher = "Canonical" offer = "UbuntuServer" sku = "18.04-LTS"

version = "latest"

}

}

resource "azurerm\_linux\_virtual\_machine" "db\_vm" {

name = "vm-db-integris"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name network\_interface\_ids = [azurerm\_network\_interface.db\_nic.id]

size = "Standard\_DS1\_v2"

os\_disk {

name = "dbOsDisk"

caching = "ReadWrite"

storage\_account\_type = "Premium\_LRS"

}

os\_proﬁle {

computer\_name = "db-server" admin\_username = "adminuser" admin\_password = "DontUsePassword1234!"

}

os\_proﬁle\_linux\_conﬁg { disable\_password\_authentication = false

}

source\_image\_reference { publisher = "Canonical" offer = "UbuntuServer" sku = "18.04-LTS"

version = "latest"

}

}

resource "azurerm\_sql\_server" "sql" { name = "sql-integris"

resource\_group\_name = azurerm\_resource\_group.integris\_rg.name location = azurerm\_resource\_group.integris\_rg.location version = "12.0"

administrator\_login = "sqladmin" administrator\_login\_password = "Password1234!"

}

resource "azurerm\_sql\_database" "sqldb" { name = "sqldb-integris"

resource\_group\_name = azurerm\_resource\_group.integris\_rg.name location = azurerm\_resource\_group.integris\_rg.location server\_name = azurerm\_sql\_server.sql.name

edition = "Standard"

requested\_service\_objective\_name = "S1"

}

resource "azurerm\_storage\_account" "storage" {

name = "storageaccountintegris"

resource\_group\_name = azurerm\_resource\_group.integris\_rg.name location = azurerm\_resource\_group.integris\_rg.location account\_tier = "Standard"

account\_replication\_type = "LRS"

}

resource "azurerm\_storage\_container" "storage\_container" {

name = "integris-container"

storage\_account\_name = azurerm\_storage\_account.storage.name container\_access\_type = "private"

}

resource "azurerm\_key\_vault" "keyvault" {

name = "keyvaultintegris"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name tenant\_id = data.azurerm\_client\_conﬁg.current.tenant\_id sku\_name = "standard"

soft\_delete\_retention\_days = 7 purge\_protection\_enabled = true

}

resource "azurerm\_bastion\_host" "bastion" { name = "bastionhostintegris"

resource\_group\_name = azurerm\_resource\_group.integris\_rg.name location = azurerm\_resource\_group.integris\_rg.location dns\_name = "bastionhostintegris"

ip\_conﬁguration {

name = "conﬁguration"

subnet\_id = azurerm\_subnet.app\_subnet.id public\_ip\_address\_id = azurerm\_public\_ip.bastion.id

}

}

resource "azurerm\_public\_ip" "bastion" { name = "bastionpublicip"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name allocation\_method = "Static"

sku = "Standard"

}

resource "azurerm\_application\_gateway" "appgw" { name = "appgatewayintegris"

resource\_group\_name = azurerm\_resource\_group.integris\_rg.name location = azurerm\_resource\_group.integris\_rg.location

sku {

name = "Standard\_v2" tier = "Standard\_v2" capacity = 2

}

gateway\_ip\_conﬁguration { name = "conﬁguration"

subnet\_id = azurerm\_subnet.db\_subnet.id

}

frontend\_port {

name = "frontendport"

port = 80

}

frontend\_ip\_conﬁguration { name = "frontend"

public\_ip\_address\_id = azurerm\_public\_ip.appgw.id

}

backend\_address\_pool { name = "backendpool"

}

http\_settings {

name = "http\_settings" cookie\_based\_affinity = "Disabled" port = 80

protocol = "Http"

request\_timeout = 20

}

listener {

name = "listener" frontend\_ip\_conﬁguration\_name = "frontend" frontend\_port\_name = "frontendport" protocol = "Http"

}

routing\_rule {

name = "routingrule"

rule\_type = "Basic" http\_listener\_name = "listener"

backend\_address\_pool\_name = "backendpool"

backend\_http\_settings\_name = "http\_settings"

}

}

resource "azurerm\_public\_ip" "appgw" { name = "appgwpublicip"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name allocation\_method = "Static"

sku = "Standard"

}

resource "azurerm\_ﬁrewall" "ﬁrewall" { name = "ﬁrewallintegris"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name sku = "AZFW\_VNet"

threat\_intel\_mode = "Alert"

ip\_conﬁguration {

name = "conﬁguration"

subnet\_id = azurerm\_subnet.db\_subnet.id public\_ip\_address\_id = azurerm\_public\_ip.ﬁrewall.id

}

}

resource "azurerm\_public\_ip" "ﬁrewall" { name = "ﬁrewallpublicip"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name allocation\_method = "Static"

sku = "Standard"

}

resource "azurerm\_monitor\_diagnostic\_setting" "monitor" {

name = "monitor"

target\_resource\_id = azurerm\_ﬁrewall.ﬁrewall.id log\_analytics\_workspace\_id = azurerm\_log\_analytics\_workspace.law.id

log {

category = "FirewallPolicy" enabled = true retention\_policy {

enabled = true

days = 30

}

}

}

resource "azurerm\_log\_analytics\_workspace" "law" {

name = "loganalyticsintegris"

location = azurerm\_resource\_group.integris\_rg.location resource\_group\_name = azurerm\_resource\_group.integris\_rg.name sku = "PerGB2018"

}