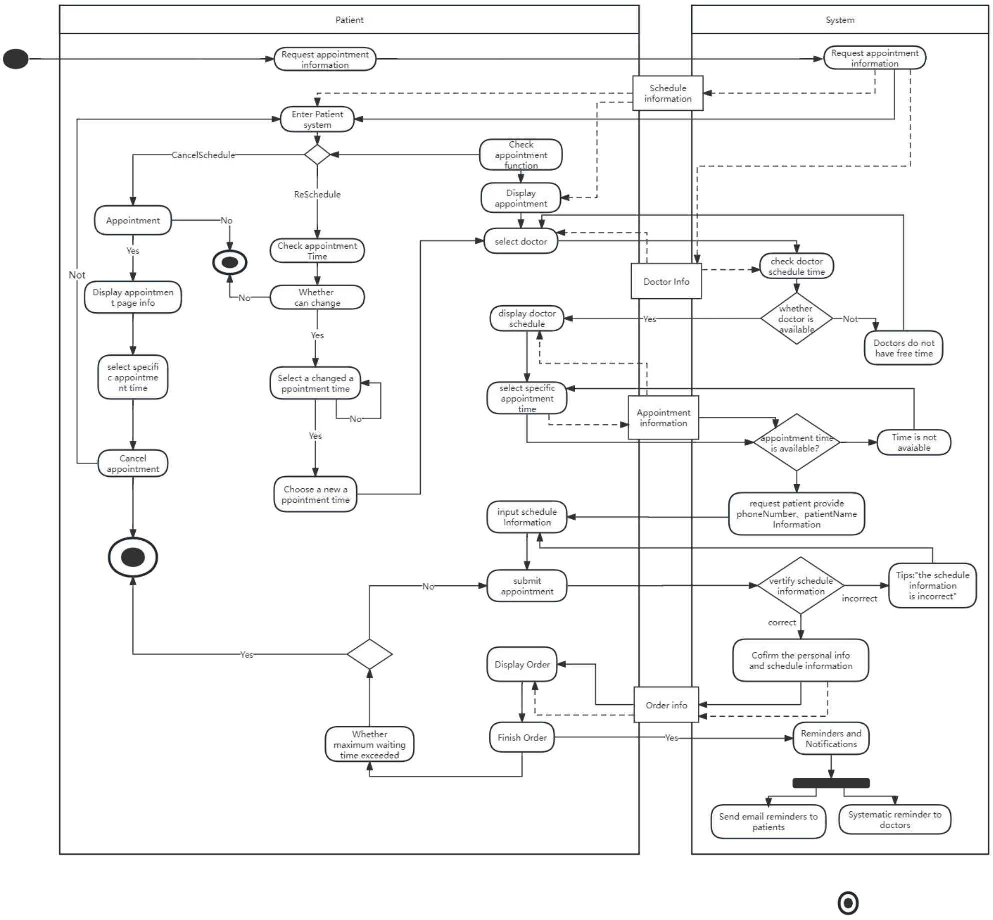
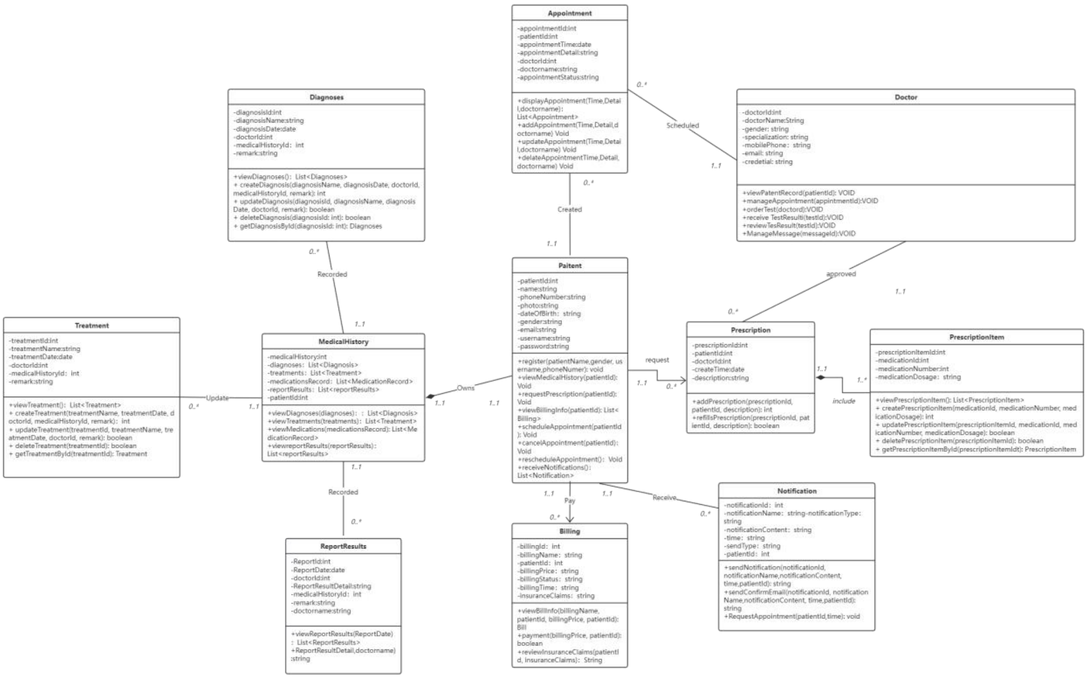
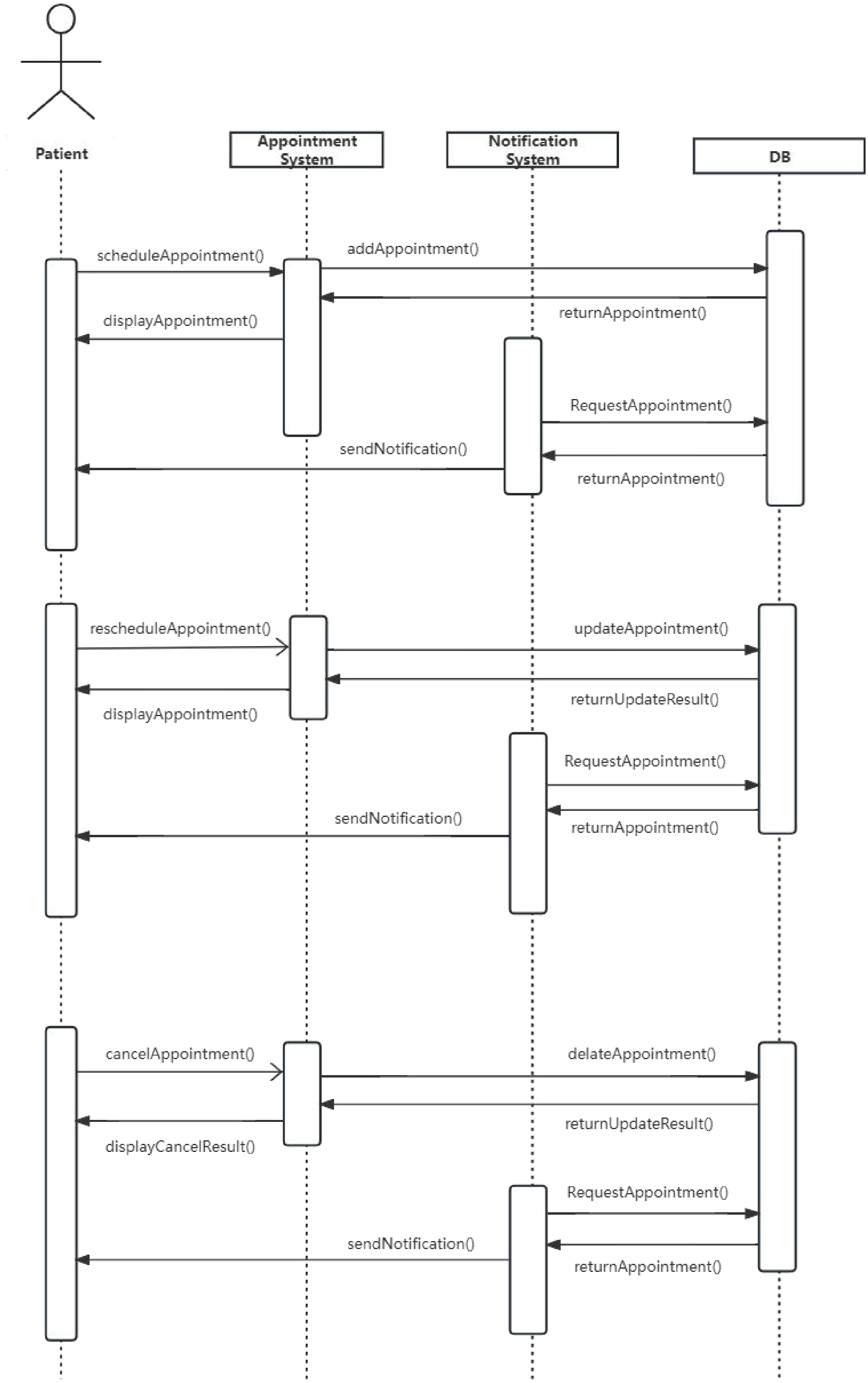
Task 1 : Software Modelling and Specification-Activity Model


Patient
System
Request appointment information
Request appointment information
Schedule information
Enter Patient system
CancelSchedule
Check appointment function
ReSchedule
Display appointment
Appointment
No
Yes
Not
select doctor
Check appointment Time
Display appointmen t page info
Whether can change
No
Doctor Info
check doctor schedule time
whether doctor is available
Yes
Not
Doctors do not have free time
display doctor schedule
Yes
select specifi c appointme nt time
Select a changed a ppointment time
No
select specific appointment time
Appointment information
appointment time is available?
Time is not avaiable
Yes
Cancel appointment
Choose a new a ppointment time
request patient provide phoneNumber、patientName Information
input schedule Information
No
submit appointment
vertify schedule information
incorrect
Tips:"the schedule information is incorrect"
correct
Yes
Cofirm the personal info and schedule information
Display Order
Order info
Whether maximum waiting time exceeded
Finish Order
Yes
Reminders and Notifications
Send email reminders to patients
Systematic reminder to doctors

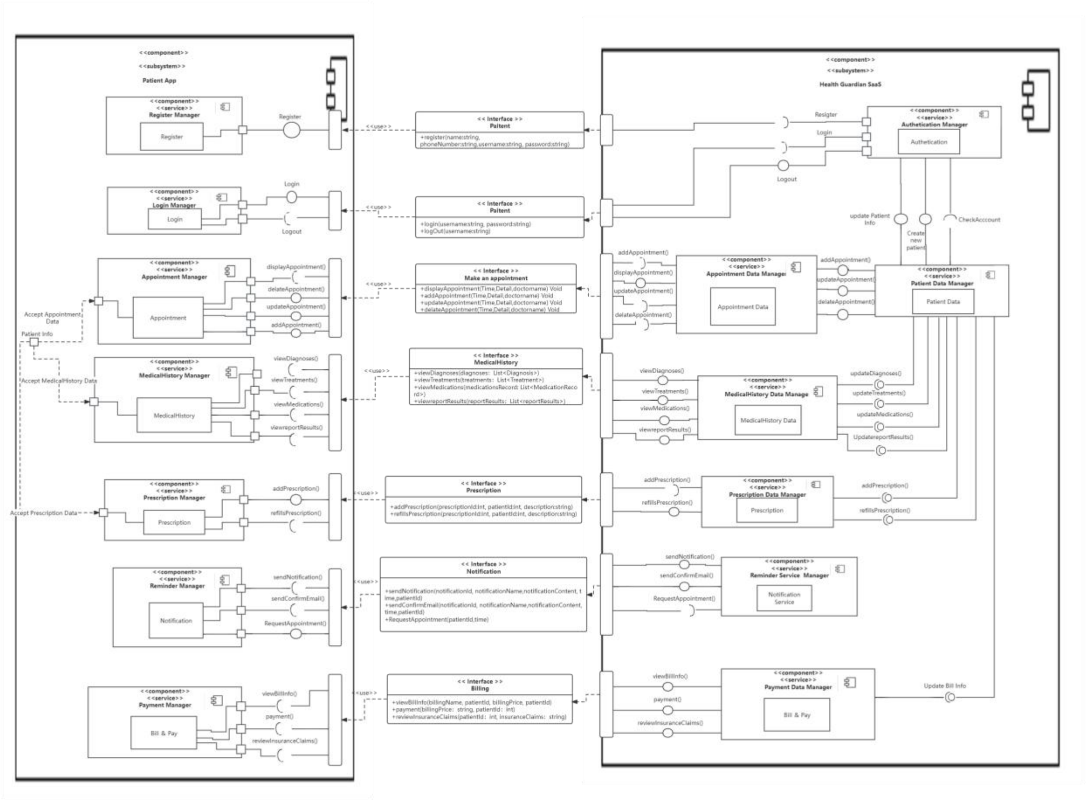
Task 1 : Software Modelling and Specification-Structural Model


Appointment
-appointmentId:int
-patientId:int
-appointmentTime:date
-appointmentDetail:string
-doctorId:int
-doctorname:string
-appointmentStatus:string
+displayAppointment(Time,Detail,doctorname): List<Appointment>
+addAppointment(Time,Detail,doctorname) Void
+updateAppointment(Time,Detail,doctorname) Void
+delateAppointmentTime,Detail,doctorname) Void
0..*
Scheduled
1..1
Doctor
-doctorId:int
-doctorName:String
-gender: string
-specialization: string
-mobilePhone: string
-email: string
-credetial: string
+viewPatentRecord(patientId): VOID
+manageAppointment(appintmentId):VOID
+orderTest(doctord):VOID
+receive TestResulti(testId):VOID
+reviewTesResult(testId):VOID
+ManageMessage(messageId):VOID
Diagnoses
-diagnosisId:int
-diagnosisName:string
-diagnosisDate:date
-doctorId:int
-medicalHistoryId: int
-remark:string
+viewDiagnoses(): List<Diagnoses>
+ createDiagnosis(diagnosisName, diagnosisDate, doctorId, medicalHistoryId, remark): int
+ updateDiagnosis(diagnosisId, diagnosisName, diagnosisDate, doctorId, remark): boolean
+ deleteDiagnosis(diagnosisId: int): boolean
+ getDiagnosisById(diagnosisId: int): Diagnoses
0..*
Created
1..1
0..*
Recorded
1..1
approved
1..1
0..*
Paitent
-patientId:int
-name:string
-phoneNumber:string
-photo:string
-dateOfBirth: string
-gender:string
-email:string
-username:string
-password:string
+register(patientName,gender, username,phoneNumer): void
+viewMedicalHistory(patientId): Void
+requestPrescription(patientId): Void
+viewBillingInfo(patientId): List<Billing>
+scheduleAppointment(patientId): Void
+cancelAppointment(patientId): Void
+rescheduleAppointment(): Void
+receiveNotifications(): List<Notification>
Treatment
-treatmentId:int
-treatmentName:string
-treatmentDate:date
-doctorId:int
-medicalHistoryId: int
-remark:string
+viewTreatment(): List<Treatment>
+ createTreatment(treatmentName, treatmentDate, doctorId, medicalHistoryId, remark): int
+ updateTreatment(treatmentId, treatmentName, treatmentDate, doctorId, remark): boolean
+ deleteTreatment(treatmentId): boolean
+ getTreatmentById(treatmentId): Treatment
Update
0..*
1..1
MedicalHistory
-medicalHistory:int
-diagnoses: List<Diagnosis>
-treatments: List<Treatment>
-medicationsRecord: List<MedicationRecord>
-reportResults: List<reportResults>
-patientId:int
+viewDiagnoses(diagnoses): : List<Diagnosis>
+viewTreatments(treatments): List<Treatment>
+viewMedications(medicationsRecord): List<MedicationRecord>
+viewreportResults(reportResults): List<reportResults>
Owns
1..1
1..1
request
1..1
0..*
Prescription
-prescriptionId:int
-patientId:int
-doctorId:int
-createTime:date
-description:string
+addPrescription(prescriptionId, patientId, description): int
+refillsPrescription(prescriptionId, patientId, description): boolean
1..1
1..*
include
PrescriptionItem
-prescriptionItemId:int
-medicationId:int
-medicationNumber:int
-medicationDosage: string
+viewPrescriptionItem(): List<PrescriptionItem>
+ createPrescriptionItem(medicationId, medicationNumber, medicationDosage): int
+ updatePrescriptionItem(prescriptionItemId, medicationId, medicationNumber, medicationDosage): boolean
+ deletePrescriptionItem(prescriptionItemId): boolean
+ getPrescriptionItemById(prescriptionItemIdt): PrescriptionItem
1..1
Recorded
0..*
1..1
Pay
0..*
1..1
Receive
0..*
ReportResults
-ReportId:int
-ReportDate:date
-doctorId:int
-ReportResultDetail:string
-medicalHistoryId: int
-remark:string
-doctorname:string
+viewReportResults(ReportDate): List<ReportResults>
+ReportResultDetail,doctorname):string
Billing
-billingId: int
-billingName: string
-patientId: int
-billingPrice: string
-billingStatus: string
-billingTime: string
-insuranceClaims: string
+viewBillInfo(billingName, patientId, billingPrice, patientId): Bill
+payment(billingPrice, patientId): boolean
+reviewInsuranceClaims(patientId, insuranceClaims): String
Notification
-notificationId: int
-notificationName: string-notificationType: string
-notificationContent: string
-time: string
-sendType: string
-patientId: int
+sendNotification(notificationId, notificationName,notificationContent, time,patientId): string
+sendConfirmEmail(notificationId, notificationName,notificationContent, time,patientId): string
+RequestAppointment(patientId,time): void

Task 2 :Software Modelling and Specification-Behaviour Model



Task 2: Software Architectural Design-Subsystem architecture:



Task 2: Software Architectural Design-Subsystem archiecture(Table)

Component	Register Manager
Description	Manages user registration, including capturing and storing user details.
Type	Patient App
Required Interface	Patient
Provided Interface	Register

Component	Login Manager
Description	Manages user logins, including authentication and session
Type	Patient App
Required Interface	Patient
Provided Interface	Login, Logout
Component	Appointment Manager
Description	Handles appointment scheduling, modifications, and cancellations.
Type	Patient App
Required Interface	displayAppointment()
Provided Interface	delateAppointment(),updateAppointment(),addAppointment()

Component	MedicalHistory Manager
Description	Manages the storage and retrieval of patients' medical history records.
Type	Patient App
Required Interface	viewDiagnoses(),viewTreatments(),viewMedications(), viewreportResults()
Provided Interface	None

Component	Prescription Manager
Description	Manages the processing and refilling of patient prescriptions.
Type	Patient App
Required Interface	refillsPrescription()
Provided Interface	addPrescription()

Component	Reminder Manager
Description	Sends notifications and reminders to patients for appointments and medications.
Type	Patient App
Required Interface	sendNotification(),sendConfirmEmail()
Provided Interface	RequestAppointment()

Component	Payment Manager
Description	This component is responsible for handling the billing process, including issuing invoices, processing payments, and managing payment records for healthcare services.
Type	Patient App
Required Interface	viewBillInfo(),payment(),reviewInsuranceClaims()
Provided Interface	None

Component	Authetication Manager
Description	This is a microservice that runs on the hospital's cloud system and is used to store, read and authenticate each patient's personal information.
Type	Health Guardian SaaS
Required Interface	Resigter, Login, CheckAcccount
Provided Interface	Logout, update Patient Info,Create new patient

Component	Appointment Data Manager
Description	This is a database for storing, modifying, updating, and deleting patient appointment information.
Type	Health Guardian SaaS
Required Interface	addAppointment(),updateAppointment(),delateAppointment()
Provided Interface	displayAppointment()

Component	Patient Data Manager
Description	It's a database for storing all the patient's personal information
Type	Health Guardian SaaS
Required Interface	Authetication,Appointment Data, MedicalHistory Data
Provided Interface	Patient Data

Component	Prescription Data Manager
Description	Maintained patient prescription database.
Type	Health Guardian SaaS
Required Interface	addPrescription(),refillsPrescription()
Provided Interface	addPrescription(),refillsPrescription()

Component	MedicalHistory Data Manager
Description	Maintains the database of medical histories, diagnoses, and treatments.
Type	Health Guardian SaaS
Required Interface	None
Provided Interface	viewDiagnoses(),viewTreatments(),viewMedications(), viewreportResults()

Component	Reminder Service Manager
Description	Oversees the reminder services, ensuring timely notifications for appointments and medications.
Type	Health Guardian SaaS
Required Interface	RequestAppointment()
Provided Interface	sendNotification(),sendConfirmEmail()

Component	Payment Data Manager
Description	Responsible for the secure storage and retrieval of payment data. This component interacts with the data layer to manage payment records and billing information.
Type	Health Guardian SaaS
Required Interface	UpdateBillInfo()
Provided Interface	UpdateBillInfo(),viewBillInfo(),payment(),reviewInsuranceClaims()

Interface:

Interface Name	Patient	
Provider	Patient App	
	Operation:	register
	Request Parameters: username (string), phoneNumber (string), password (string) **Response Parameters:** successMessage (string), if registration is successful; errorMessage (string), if failure. **Function:** Registers a new patient by collecting their username, phone number, and password, creating a new patient record in the system.	
	Operation:	login
	Request Parameters: username (string), password (string) **Response Parameters:** sessionToken (string), if login is successful; errorMessage (string), if failure. **Function:** Authenticates a patient by verifying their username and password against stored records and generates a session token for successful logins.	
	Operation:	logout
	Request Parameters: sessionToken (string) **Response Parameters:** successMessage (string), confirming successful logout. **Function:** Terminates a patient's session in the system, ensuring their authentication token is invalidated.	

Interface Name	Make an Appointment	
Provider	Appointment Manager	
	Operation:	displayAppointment
	Request Parameters: patientId (string), date (date) **Response Parameters:** appointmentDetails (AppointmentDetail[]), list of appointments.	

	Function: Retrieves and displays all scheduled appointments for a given patient on a specified date.	
	Operation:	addAppointment
	Request Parameters: patientId (string), appointmentDetail (AppointmentDetail) **Response Parameters:** appointmentId (string), confirming the creation of the appointment. **Function:** Schedules a new appointment for the patient and stores the details in the system.	
	Operation:	updateAppointment
	Request Parameters: appointmentId (string), updatedDetail (AppointmentDetail) **Response Parameters:** successMessage (string), confirming the update. **Function:** Updates the details of an existing appointment.	
	Operation:	deleteAppointment
	Request Parameters: appointmentId (string) **Response Parameters:** successMessage (string), confirming the deletion. **Function:** Removes an existing appointment from the system.	

Interface Name	MedicalHistory	
Provider	MedicalHistory Manager	
	Operation:	viewDiagnoses
	Request Parameters: patientId (string) **Response Parameters:** diagnosesList (Diagnosis[]) **Function:** Retrieves a list of past diagnoses for the patient.	
	Operation:	viewTreatments
	Request Parameters: patientId (string) **Response Parameters:** medicationsList (Medication[])	

	Function: Retrieves a list of medications that have been prescribed to the patient in the past	
	Operation:	viewLabResults
	Request Parameters: patientId (string) **Response Parameters:** labResultsList (LabResult[]) **Function:** Retrieves a list of lab results for the patient.	
	Operation:	deleteAppointment
	Request Parameters: appointmentId (string) **Response Parameters:** successMessage (string), confirming the deletion. **Function:** Removes an existing appointment from the system.	

Interface Name	Prescription	
Provider	Prescription Manager	
	Operation:	addPrescription
	Request Parameters: patientId (string), prescriptionDetail (PrescriptionDetail) **Response Parameters:** prescriptionId (string), confirming the creation of the prescription. **Function:** Adds a new prescription record for the patient.	
	Operation:	refillPrescription
	Request Parameters: prescriptionId (string), quantity (integer) **Response Parameters:** successMessage (string), confirming the refill. **Function:** Processes a request to refill an existing prescription for the patient.	

Interface Name	Notification	
Provider	Reminder Manager	
	Operation:	sendNotification
	Request Parameters: patientId (string), notificationDetail (NotificationDetail) **Response Parameters:** successMessage (string), confirming the sending of the notification. **Function:** Sends a notification to the patient for appointments or medication reminders.	
	Operation:	sendConfirmationEmail
	Request Parameters: patientId (string), emailDetail (EmailDetail) **Response Parameters:** successMessage (string), confirming the sending of the email. **Function:** Sends an email confirmation for appointments or other interactions with the patient.	
	Operation:	updateAppointment
	Request Parameters: appointmentId (string), updatedDetail (AppointmentDetail) **Response Parameters:** successMessage (string), confirming the update. **Function:** Updates the details of an existing appointment.	
	Operation:	deleteAppointment
	Request Parameters: appointmentId (string) **Response Parameters:** successMessage (string), confirming the deletion. **Function:** Removes an existing appointment from the system.	

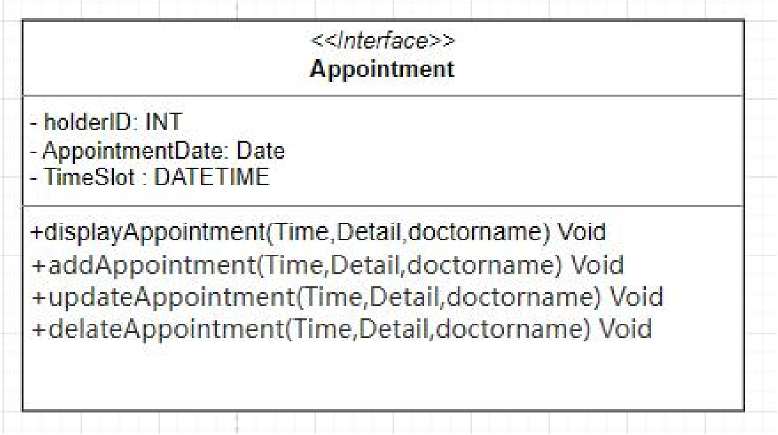
Interface Name	Billing	
Provider	Payment Data Manager	
	Operation:	viewBillInfo
	Request Parameters: billingName: string, patientId: int, billingPrice: decimal, patientId: int	

	Response Parameters: billDetails **Function:** Retrieves a comprehensive view of the bill for a patient, including the breakdown of charges.	
	Operation:	payment
	Request Parameters: billingPrice: string, patientId: int **Response Parameters:** paymentConfirmation: boolean, transactionId: string **Function:** Processes the payment for the patient’s bill based on the amount due and updates the billing status.	
	Operation:	reviewInsuranceClaims
	Request Parameters: appointmentId (string), updatedDetail (AppointmentDetail) **Response Parameters:** claimsStatus **Function:** Reviews and processes insurance claims associated with the patient's account, including coordination with insurance providers.	

Task 3：Software Testing

1. Unit test plan

Test case interface：



<<Interface>>
Appointment

- holderID: INT
- AppointmentDate: Date
- TimeSlot : DATETIME

+displayAppointment(Time,Detail,doctorname) Void
+addAppointment(Time,Detail,doctorname) Void
+updateAppointment(Time,Detail,doctorname) Void
+delateAppointment(Time,Detail,doctorname) Void

Test case：

Test Case	Method and Parameters	Expected Output
Display Appointments Success	displayAppointments(Shu Yujun, 2023-12-20,14:00)	Successfully retrieved appointment information
Display Appointments Fail	displayAppointments(Shu Yujun, 2023-12-20,14:00)	Failed to retrieve appointment information
Add Appointment Success	addAppointment(Shu Yujun, 2023-12-20,14:00)	Successfully added appointment information
Add Appointment Fail	addAppointment(Shu Yujun, 2023-12-20,14:00)	Failed to add appointment information
Delate Appointment Success	delateAppointment(Shu Yujun, 2023-12-20,14:00)	Successfully removed appointment information
Delate Appointment Fail	delateAppointment(Shu Yujun, 2023-12-20,14:00)	Failed to remove appointment information
Update Appointment Success	updateAppointment(Shu Yujun, 2023-12-20,14:00)	Successfully updated appointment information
Update Appointment Fail	updateAppointment(Shu Yujun, 2023-12-20,14:00)	Failed to update appointment information

2. System test plan

Use case: Make an appointment

Scenario: Successful remove

Patient App	Health Guardian SaaS
1. Request to view available appointment slots.	2. Find available slots for the requested date and time.
3. Receive available slots information.	4. Provide available slots information to the Patient App.
5. Request to schedule an appointment in a chosen time slot.	6. Process appointment scheduling request.
7. Receive confirmation of scheduled appointment.	8. Confirm appointment and update schedule.
9. Optionally, request to view or update appointment details.	10. Optionally, provide updated appointment details.

Test data:

Input: patient ID, desired appointment date, desired time slot

Stored data: patient ID, appointment date, time slot

Output: patient ID, appointment date, time slot

Test progress:

Patient App	Health Guardian SaaS
1. Request to view available slots for patient ID: 123456, on 2023-12-20.	2. Find available slots for patient ID: 123456, on 2023-12-20.
3. Get the data of available slots for the specified date and time.	4. Return the data of available slots for the specified date and time.
5. View the available slots and select the desired time slot: 15:00.	6. Confirm the selection and process the appointment request for the time slot: 15:00.
7. Request to schedule the appointment at the desired time slot: 15:00.	8. Schedule the appointment and update the system with the new booking information.
9. Receive confirmation of the scheduled appointment for 2023-12-20 at 15:00.	10. Provide confirmation of the scheduled appointment to the Patient App.

Additional test cases:

1. Add Appointment

Patient App	Health Guardian SaaS
1. Request to add appointment data for patient ID: 123456, date: 2023-12-20, time slot: 15:00.	2. Find the schedule and check availability for the requested date and time slot.
3. Get confirmation of the appointment booking possibility.	4. Return available status for the requested date and time slot.
5. Confirm the booking of the appointment for the desired time slot.	6. Process the booking and add appointment to the schedule.
7. Receive confirmation of the appointment being successfully added.	8. Return the confirmation of the new appointment booking for patient ID: 123456.

2. Display Appointment

Patient App	Health Guardian SaaS
1. Request to view appointment data for patient ID: 123456, date: 2023-12-20.	2. Retrieve the appointment details for the requested date for patient ID: 123456.
3. Get the data of the scheduled appointment.	4. Return the details of the scheduled appointment.
5. View the data of the appointment booking.	6. Display the appointment booking information to the patient.

3. Delete Appointment

Patient App	Health Guardian SaaS
1. Request to remove appointment data for patient ID: 123456, date: 2023-12-20, time slot: 15:00.	2. Find the appointment to be deleted for the requested date and time slot.
3. Get confirmation of appointment deletion eligibility.	4. Return the status of the appointment found for the specified time slot.
5. Confirm the deletion of the appointment.	6. Process the deletion and remove the appointment from the schedule.
7. Receive confirmation of the appointment being successfully deleted.	8. Return the confirmation of the appointment deletion for patient ID: 123456.

4. Update Appointment

Patient App	Health Guardian SaaS
1. Request to update appointment data for patient ID: 123456 to a new time slot: 15:00.	2. Find the existing appointment and check availability for the new time slot.
3. Get the details of the appointment that can be updated.	4. Return the current details of the appointment eligible for update.
5. Confirm the update of the appointment to the new time slot.	6. Process the update and change the appointment details in the schedule.
7. Receive confirmation of the appointment being successfully updated.	8. Return the updated appointment details for patient ID: 123456.