Computer Science Program

Sino-British Collaborative Education

CDUT

Software Engineering

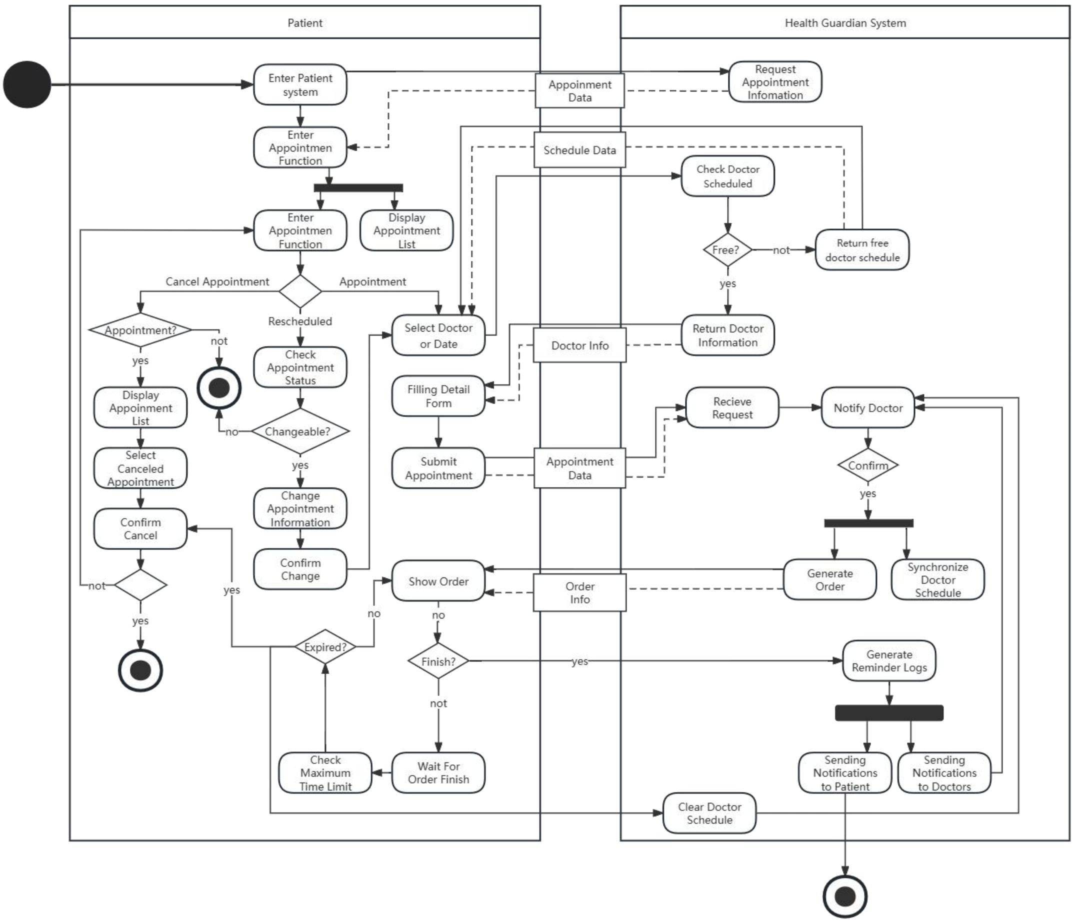
Individual work

L6C1-Noah
2023/12/21

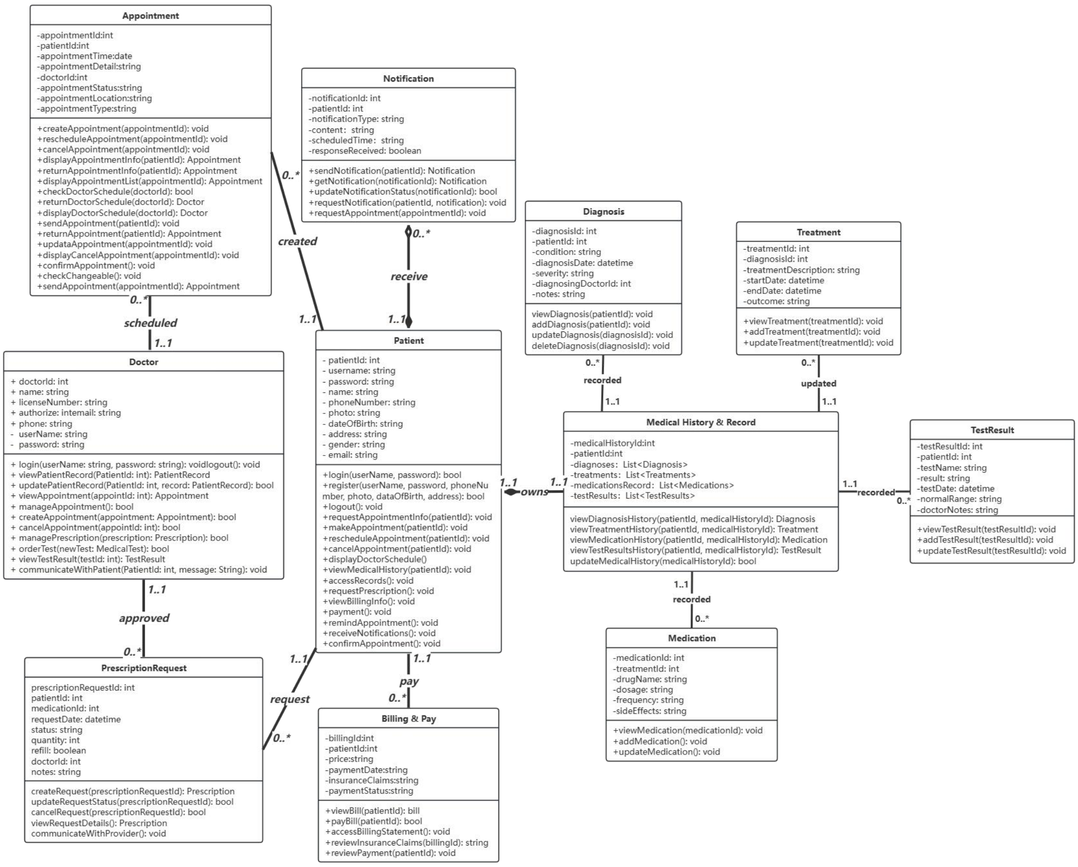
Content Table

1. Task 1: Software Modelling and Specification

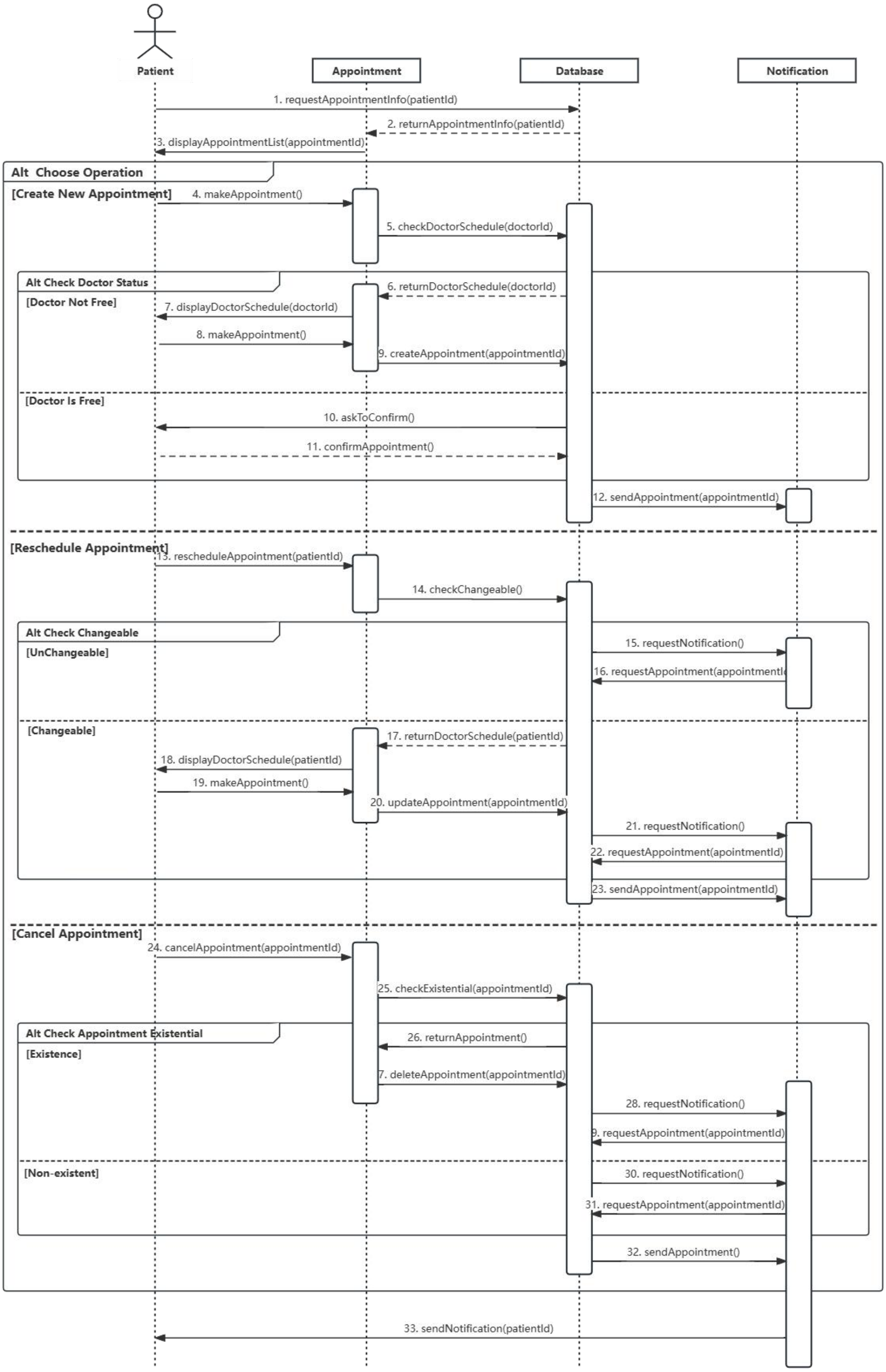
1.1 Activity Model



1.2 Structural Model

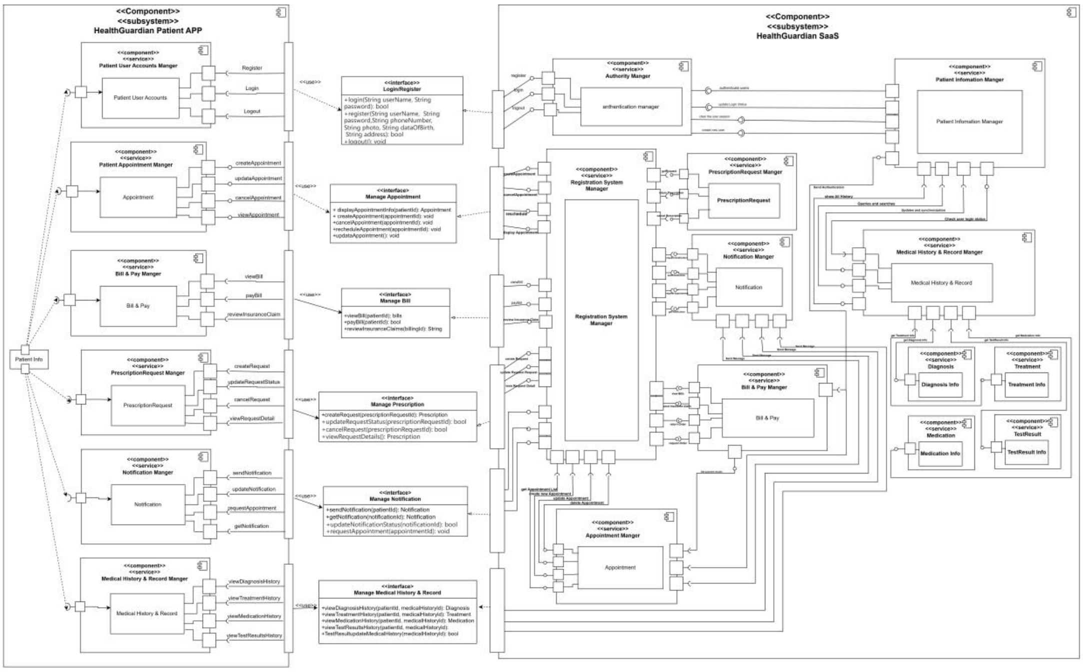

Appointment
-appointmentId:int
-patientId:int
-appointmentTime:date
-appointmentDetail:string
-doctorId:int
-appointmentStatus:string
-appointmentLocation:string
-appointmentType:string
+createAppointment(appointmentId): void
+rescheduleAppointment(appointmentId): void
+cancelAppointment(appointmentId): void
+displayAppointmentInfo(patientId): Appointment
+returnAppointmentInfo(patientId): Appointment
+displayAppointmentList(appointmentId): Appointment
+checkDoctorSchedule(doctorId): bool
+returnDoctorSchedule(doctorId): Doctor
+displayDoctorSchedule(doctorId): Doctor
+sendAppointment(patientId): void
+returnAppointment(patientId): Appointment
+updataAppointment(appointmentId): void
+displayCancelAppointment(appointmentId): void
+confirmAppointment(): void
+checkChangeable(): void
+sendAppointment(appointmentId): Appointment
Notification
-notificationId: int
-patientId: int
-notificationType: string
-content： string
-scheduledTime： string
-responseReceived: boolean
+sendNotification(patientId): Notification
+getNotification(notificationId): Notification
+updateNotificationStatus(notificationId): bool
+requestNotification(patientId, notification): void
+requestAppointment(appointmentId): void
0..*
created
1..1
0..*
receive
1..1
0..*
scheduled
1..1
Doctor
+ doctorId: int
+ name: string
+ licenseNumber: string
+ authorize: intemail: string
+ phone: string
- userName: string
- password: string
+ login(userName: string, password: string): voidlogout(): void
+ viewPatientRecord(PatientId: int): PatientRecord
+ updatePatientRecord(PatientId: int, record: PatientRecord): bool
+ viewAppointment(appointId: int): Appointment
+ manageAppointment(): bool
+ createAppointment(appointment: Appointment): bool
+ cancelAppointment(appointId: int): bool
+ managePrescription(prescription: Prescription): bool
+ orderTest(newTest: MedicalTest): bool
+ viewTestResult(testId: int): TestResult
+ communicateWithPatient(PatientId: int, message: String): void
Patient
- patientId: int
- username: string
- password: string
- name: string
- phoneNumber: string
- photo: string
- dateOfBirth: string
- address: string
- gender: string
- email: string
+login(userName, password): bool
+register(userName, password, phoneNumber, photo, dataOfBirth, address): bool
+logout(): void
+requestAppointmentInfo(patientId): void
+makeAppointment(patientId): void
+rescheduleAppointment(patientId): void
+cancelAppointment(patientId): void
+displayDoctorSchedule()
+viewMedicalHistory(patientId): void
+accessRecords(): void
+requestPrescription(): void
+viewBillingInfo(): void
+payment(): void
+remindAppointment(): void
+receiveNotifications(): void
+confirmAppointment(): void
1..1
approved
0..*
PrescriptionRequest
prescriptionRequestId: int
patientId: int
medicationId: int
requestDate: datetime
status: string
quantity: int
refill: boolean
doctorId: int
notes: string
createRequest(prescriptionRequestId): Prescription
updateRequestStatus(prescriptionRequestId): bool
cancelRequest(prescriptionRequestId): bool
viewRequestDetails(): Prescription
communicateWithProvider(): void
1..1
request
0..*
1..1
pay
0..*
Billing & Pay
-billingId:int
-patientId:int
-price:string
-paymentDate:string
-insuranceClaims:string
-paymentStatus:string
+viewBill(patientId): bill
+payBill(patientId): bool
+accessBillingStatement(): void
+reviewInsuranceClaims(billingId): string
+reviewPayment(patientId): void
1..1
owns
1..1
Diagnosis
-diagnosisId: int
-patientId: int
-condition: string
-diagnosisDate: datetime
-severity: string
-diagnosingDoctorId: int
-notes: string
viewDiagnosis(patientId): void
addDiagnosis(patientId): void
updateDiagnosis(diagnosisId): void
deleteDiagnosis(diagnosisId): void
0..*
recorded
1..1
Treatment
-treatmentId: int
-diagnosisId: int
-treatmentDescription: string
-startDate: datetime
-endDate: datetime
-outcome: string
+viewTreatment(treatmentId): void
+addTreatment(treatmentId): void
+updateTreatment(treatmentId): void
0..*
updated
1..1
Medical History & Record
-medicalHistoryId:int
-patientId:int
-diagnoses： List<Diagnosis>
-treatments： List<Treatments>
-medicationsRecord： List<Medications>
-testResults： List<TestResults>
viewDiagnosisHistory(patientId, medicalHistoryId): Diagnosis
viewTreatmentHistory(patientId, medicalHistoryId): Treatment
viewMedicationHistory(patientId, medicalHistoryId): Medication
viewTestResultsHistory(patientId, medicalHistoryId): TestResult
updateMedicalHistory(medicalHistoryId): bool
1..1
recorded
0..*
TestResult
-testResultId: int
-patientId: int
-testName: string
-result: string
-testDate: datetime
-normalRange: string
-doctorNotes: string
+viewTestResult(testResultId): void
+addTestResult(testResultId): void
+updateTestResult(testResultId): void
1..1
recorded
0..*
Medication
-medicationId: int
-treatmentId: int
-drugName: string
-dosage: string
-frequency: string
-sideEffects: string
+viewMedication(medicationId): void
+addMedication(): void
+updateMedication(): void

1.3 Behaviour Model


Patient
Appointment
Database
Notification
1. requestAppointmentInfo(patientId)
2. returnAppointmentInfo(patientId)
3. displayAppointmentList(appointmentId)
Alt Choose Operation
[Create New Appointment]
4. makeAppointment()
5. checkDoctorSchedule(doctorId)
Alt Check Doctor Status
[Doctor Not Free]
6. returnDoctorSchedule(doctorId)
7. displayDoctorSchedule(doctorId)
8. makeAppointment()
9. createAppointment(appointmentId)
[Doctor Is Free]
10. askToConfirm()
11. confirmAppointment()
12. sendAppointment(appointmentId)
[Reschedule Appointment]
13. rescheduleAppointment(patientId)
14. checkChangeable()
Alt Check Changeable
[UnChangeable]
15. requestNotification()
16. requestAppointment(appointmentI
[Changeable]
17. returnDoctorSchedule(patientId)
18. displayDoctorSchedule(patientId)
19. makeAppointment()
20. updateAppointment(appointmentId)
21. requestNotification()
22. requestAppointment(apointmentId)
23. sendAppointment(appointmentId)
[Cancel Appointment]
24. cancelAppointment(appointmentId)
25. checkExistential(appointmentId)
Alt Check Appointment Existential
[Existence]
26. returnAppointment()
7. deleteAppointment(appointmentId)
28. requestNotification()
9. requestAppointment(appointmentId)
[Non-existent]
30. requestNotification()
31. requestAppointment(appointmentId)
32. sendAppointment()
33. sendNotification(patientId)

2. Task 2: Software Architectural Design

2.1 Architecture of the subsystem



2.2 Software Architectural Design (Table)

2.2.1 Component

Component	**Patient User Accounts Manager**
Description	Manages user accounts within the HealthGuardian system, handling user registration, login, and logout processes.
Type	Health Guardian Patient APP
Required Interfaces:	Register, Login, Logout, Patient Info
Provided Interfaces:	None

Component	**Patient Appointment Manager**
Description	Responsible for processing and maintaining patient

	appointment information, including appointment creation, updates, cancellations, and inquiries.
Type	Health Guardian Patient APP
Required Interfaces:	createAppointment, updataAppointment, cancelAppointment, viewAppointment, patient Info
Provided Interfaces:	None

Component	**PrescriptionRequest Manger**
Description	Responsible for managing patient prescription requests. This includes creating new medication prescription requests, updating the status of prescription requests, canceling requests that have already been sent, and viewing the details of prescription requests.
Type	Health Guardian Patient APP
Required Interfaces:	createReques, updateRequestStatus, cancelRequest, viewRequestDetail
Provided Interfaces:	None

Component	**Bill & Pay**
Description	Manage patient billing and payment processes, including bill viewing, payment processing, and insurance claim review.
Type	Health Guardian Patient APP
Required Interfaces:	viewBill, payBill, reviewInsuranceClaim
Provided Interfaces:	None

Component	**Notification Manger**
Description	Manage system notifications, including sending notifications, updating notification status, processing appointment request notifications, and getting notification details.
Type	Health Guardian Patient APP
Required Interfaces:	sendNotification, updateNotification, requestAppointment,

	getNotification
Provided Interfaces:	None

Component	**Medical History & Record Manger**
Description	Responsible for maintaining and displaying patient medical history and records, including historical information on diagnosis, treatment, medication, and test results.
Type	Health Guardian Patient APP
Required Interfaces:	viewDiagnosisHistory, viewTreatmentHistory, viewMedicationHistory, viewTestResultsHistory
Provided Interfaces:	None

Component	**Authority Manger**
Description	This component serves as the central authority within an authentication system, managing user sessions and handling authentication processes. It is designed to control access to the system by verifying user credentials and maintaining session states.
Type	HealthGuardian SaaS
Required Interfaces:	Rigister, Login, Logout
Provided Interfaces:	authenticate users, update Login Status, clear the user session, create new user

Component	**Patient Information Manger**
Description	This component specializes in the management and retrieval of patient-related information within a healthcare system. It ensures that patient data, including medical history and treatment records, is accurately maintained, securely accessed, and efficiently updated.
Type	HealthGuardian SaaS
Required Interfaces:	show All History, Queries and searches, Updates and

	synchronization
Provided Interfaces:	Send Authentication, Check user login status

Component	**Medical History & Record Manger**
Description	This service is responsible for the centralized management of a patient's medical history and records within a healthcare system. It processes and organizes critical patient information, such as treatments, diagnoses, medication, and test results, to support healthcare delivery.
Type	HealthGuardian SaaS
Required Interfaces:	get Treatment Info, get Diagnosis Info, get TestResult Info, get Medication Info
Provided Interfaces:	show All History, Queries and searches, Updates and synchronization

Component	**PrescriptionRequest Manger**
Description	This service is responsible for the centralized management of a patient's medical history and records within a healthcare system. It processes and organizes critical patient information, such as treatments, diagnoses, medication, and test results, to support healthcare delivery.
Type	HealthGuardian SaaS
Required Interfaces:	None
Provided Interfaces:	getRequest, return Prescription, cancel Prescription

Component	**Notification Manger**
Description	This component operates as a communication hub within a service ecosystem, primarily focusing on the distribution and management of notifications. It handles the sending, requesting, and receiving of notifications, potentially including appointment reminders, service alerts, and other important messages.
Type	HealthGuardian SaaS

Required Interfaces:	None
Provided Interfaces:	requestNotification, sendNotification, requestAppointment, getNotification

Component	**Bill & Pay Manager**
Description	This component is designed to manage financial transactions related to billing and payments within a service framework. It likely handles the generation and viewing of bills, processing insurance claims, and managing orders and payments.
Type	HealthGuardian SaaS
Required Interfaces:	send Insurance Claim, request Order
Provided Interfaces:	view Bills, return Order

Component	**Appointment Manager**
Description	The Appointment Manager component is a service that orchestrates the scheduling, updating, and cancellation of appointments. It serves as a central hub for managing appointment-related operations, ensuring that scheduling is efficient and that any changes to appointments are reflected across the system.
Type	HealthGuardian SaaS
Required Interfaces:	get Appointment List, create new Appointment, update Appointment, delete Appointment
Provided Interfaces:	Get payment results

Component	**Appointment Manager**
Description	The Appointment Manager component is a service that orchestrates the scheduling, updating, and cancellation of appointments. It serves as a central hub for managing appointment-related operations, ensuring that scheduling is efficient and that any changes to appointments are reflected across the system.
Type	HealthGuardian SaaS

Required Interfaces:	get Appointment List, create new Appointment, update Appointment, delete Appointment
Provided Interfaces:	Get payment results

Component	**Diagnosis**
Description	This service component is dedicated to managing and providing information related to patient diagnoses. It facilitates the retrieval of diagnosis details that are crucial for determining the appropriate course of treatment.
Type	HealthGuardian SaaS
Required Interfaces:	None
Provided Interfaces:	get Diagnosis Info

Component	**TestResult**
Description	The TestResult service component handles information related to the test results of patients. It organizes and provides access to test outcomes, which are essential for diagnosing and treating conditions.
Type	HealthGuardian SaaS
Required Interfaces:	None
Provided Interfaces:	get Medication Info

Component	**Medication**
Description	This service component manages detailed information regarding patient medications. This includes prescribing information, dosages, and schedules.
Type	HealthGuardian SaaS
Required Interfaces:	None
Provided Interfaces:	get Medication Info

Component	**Treatment**
Description	This component is responsible for handling information

	pertaining to patient treatments. It provides detailed records of treatment procedures, plans, and outcomes.
Type	HealthGuardian SaaS
Required Interfaces:	None
Provided Interfaces:	get Treatment Info

2.2.2 Interface Table

Interface Name	**Login/Register**
Provider	Health Guardian Saas
	Operation: **Login**
	Request Parameters: String username, String password **Response parameters:** On success: The user is redirected to the landing page URL. This typically involves sending a status code indicating success (e.g., HTTP 200 OK) along with the URL of the landing page. On failure: An error message is displayed to the user. This could involve sending a status code indicating an unauthorized attempt (e.g., HTTP 401 Unauthorized) along with a message detailing the reason for the failure (e.g., "Invalid username or password"). **Function:** Validates user login details against the database. If the credentials match, the user is logged in successfully. If the credentials do not match, an error message is displayed.
	Operation: **Register**
	Request Parameters: String username, String password, String phoneNumber, String photo, String dataOfBirth, String address **Response parameters:** On success: The user's information is updated in the database. This could involve sending a status code indicating success (e.g., HTTP 200 OK) and a confirmation message (e.g., "Registration successful"). On conflict: An error message is displayed to the user. This could involve sending a status code indicating a conflict (e.g., HTTP 409 Conflict) along with a message detailing the reason

	for the conflict (e.g., "Username already exists"). **Function:** Registers a new user. It involves adding new user information to the system after verifying that there is no conflict with existing database information. If the new information does not conflict, the new user is registered and their information is added to the database. If there is a conflict (e.g., the username already exists), an error message is returned.
	Operation: **logout**
	Request Parameters: None **Response parameters:** Success: Clears the user session and returns to the login screen Failure: Error message "Logout Failed" is displayed. Function: Clear user login status

Interface Name	**Manage Appointment**
Provider	Health Guardian Saas
	Operation: **Display Appointment Information**
	Request Parameters: int patientId **Response parameters:** Success: Returns an Appointment object with the appointment details. Failure: An error message is displayed. **Function:** Retrieves and displays detailed information about the appointments of a patient, identified by their patient ID.
	Operation: **Create Appointment**
	Request Parameters: String appointmentId

	Response parameters: Success: Acknowledges the creation of a new appointment. Failure: An error message is displayed. **Function:** Creates a new appointment in the system using the provided appointment ID.
	Operation: **Cancel Appointment**
	Request Parameters: String appointmentId **Response Parameters:** Success: Acknowledges the cancellation of the appointment. Failure: An error message is displayed. **Function:** Cancels an existing appointment based on the appointment ID.
	Operation: **Reschedule Appointment**
	Request Parameters: String appointmentId **Response Parameters:** Success: Acknowledges the rescheduling of the appointment. Failure: An error message is displayed. **Function:** Reschedules an existing appointment to a new date and/or time, using the appointment ID.
	Operation: **Update Appointment**
	Request Parameters: None

	Response Parameters: Success: Acknowledges the rescheduling of the appointment. Failure: An error message is displayed. **Function:** Reschedules an existing appointment to a new date and/or time, using the appointment ID.

Interface Name	**Manage Bill**
Provider	Health Guardian Saas
	Operation: **Manage Bill**
	Request Parameters: String patientId **Response Parameters:** Returns a 'bills' object or data structure containing the billing details for the specified patient ID. **Function:** Provides the ability to view the billing information associated with a patient's account.
	Operation: **Pay Bill**
	Request Parameters: String patientId **Response Parameters:** Success: Returns true, indicating the bill has been paid. Failure: Returns false, along with an error message indicating why the payment failed. **Function:** Processes the payment for the patient's bill using their unique patient ID.
	Operation: **Review Insurance Claims**
	Request Parameters: String billingId **Response Parameters:** Returns a String that contains the

	status or details of the insurance claim associated with the billing ID. **Function:** Allows for the review and status checking of insurance claims related to the billing process.

Interface Name	**Manage Prescription**
Provider	Health Guardian Saas
	Operation: **Create Request**
	Request Parameters: int prescriptionRequestId **Response Parameters:** Returns a Prescription object or data structure containing the details of the newly created prescription. **Function:** Initiates a request for a new prescription using the provided prescription request ID.
	Operation: **Update Request Status**
	Request Parameters: int prescriptionRequestId **Response Parameters:** Success: Returns true, indicating the prescription request status has been updated. Failure: Returns false, along with an error message indicating why the update failed. **Function:** Updates the status of an existing prescription request, such as from 'pending' to 'approved' or 'denied'.
	Operation: **Cancel Request**
	Request Parameters: int prescriptionRequestId **Response Parameters:** Success: Returns true, indicating the prescription request has been successfully canceled.

	Failure: Returns false, indicating the cancellation was unsuccessful, along with an error message. **Function:** Cancels an existing prescription request using the provided prescription request ID.
	Operation: View Request Details
	Request Parameters: None **Response Parameters:** Returns a Prescription object or data structure containing the details of a prescription request. **Function:** Provides the details of a specific prescription request. The details retrieved would typically include medication names, dosages, instructions for use, and any other relevant information.

Interface Name	**Manage Notification**
Provider	Health Guardian Saas
	Operation: **Create Request**
	Request Parameters: int patientId **Response Parameters:** Returns a Notification object or data structure with the details of the sent notification. **Function:** Sends a notification to a patient using their unique patient ID.
	Operation: **Get Notification**
	Request Parameters: int notificationId **Response Parameters:** Returns a Notification object or data structure with the details of a specific notification. **Function:** Retrieves the details of a notification using a unique notification ID.
	Operation: **Update Notification Status**
	Request Parameters: int notificationId

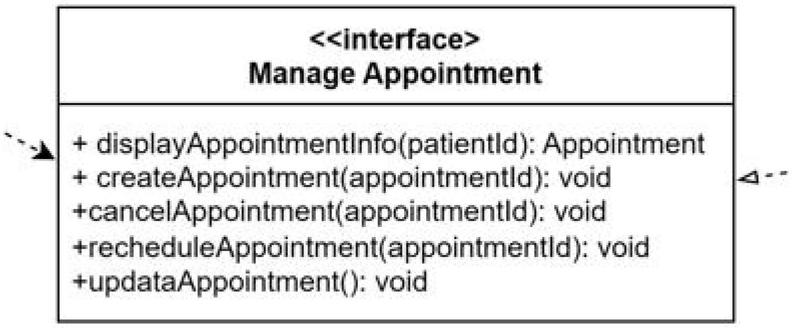
	Response Parameters: Success: Returns true, indicating the notification status has been updated. Failure: Returns false, along with an error message explaining the failure. **Function:** Updates the status of a notification, such as marking it as read or acted upon, using the notification ID.
	Operation: **Request Appointment**
	Request Parameters: int appointmentId **Response Parameters:** No return value is indicated (void), which implies the function does not return a response but performs an action. **Function:** Initiates a request for an appointment using the provided appointment ID, likely triggering a notification or other subsequent actions.

Interface Name	**Manage Medical History & Record**
Provider	Health Guardian Saas
	Operation: **View Diagnosis History**
	Request Parameters: int patientId, int medicalHistoryId **Response Parameters:** Returns a Diagnosis object or data structure with the patient's diagnosis history. **Function:** Retrieves the diagnosis history for a patient using their unique patient ID and medical history ID.
	Operation: **View Treatment History**
	Request Parameters: int patientId, int medicalHistoryId **Response Parameters:** Returns a Treatment object or data structure with the patient's treatment history.

	Function: Retrieves the treatment history for a patient using their unique patient ID and medical history ID.
	Operation: **View Medication History**
	Request Parameters: int patientId, int medicalHistoryId **Response Parameters:** Returns a Medication object or data structure with the patient's medication history. **Function:** Retrieves the medication history for a patient using their unique patient ID and medical history ID.
	Operation: **View Test Results History**
	Request Parameters: int patientId, int medicalHistoryId **Response Parameters:** Returns a data structure with the patient's test results history. **Function:** Retrieves the test results history for a patient using their unique patient ID and medical history ID.
	Operation: TestResult update Medical History
	Request Parameters: int medicalHistoryId **Response Parameters:** Success: Returns true, indicating the medical history has been updated with the new test results. Failure: Returns false, along with an error message explaining the failure. **Function:** Updates the medical history record of a patient with new test results using the medical history ID.

3. Task3: Software Testing

3.1 Unit Test Plan for Manage Appointment Interface


<<interface>>
Manage Appointment
+ displayAppointmentInfo(patientId): Appointment
+ createAppointment(appointmentId): void
+cancelAppointment(appointmentId): void
+recheduleAppointment(appointmentId): void
+updataAppointment(): void

3.1.1 Test Cases for 'displayAppointmentInfo()' Method

Test Case	Method and Parameters	Expected Output
View Appointment	displayAppointmentInfo(2023)	'Appointment' object with details
Invalid Patient	displayAppointmentInfo(-1)	Error or 'null'

3.1.2 Test Cases for 'createAppointment()' Method

Test Case	Method and Parameters	Expected Output
Create Appointment	createAppointment (2023)	Method call with no return value
Create Appointment with Invalid ID	createAppointment(-1)	Error

3.1.3 Test Cases for 'cancelAppointment()' Method

Test Case	Method and Parameters	Expected Output
Cancel Appointment	cancelAppointment (2023)	Method call with no return value
Cancel Non-existent Appointment	cancelAppointment (-1)	Error

3.1.4 Test Cases for ‘rescheduleAppointment()’ Method

Test Case	Method and Parameters	Expected Output
Reschedule Appointment	rescheduleAppointment (2023)	Method call with no return value
Cancel Non-existent Appointment	rescheduleAppointment(-1)	Error

3.1.5 Test Cases for ‘updateAppointment()’ Method

Test Case	Method and Parameters	Expected Output
Update Appointment	updateAppointment ()	Method call with no return value
Cancel Non-existent Appointment	Update Without Details	Error or Method call with no return value

3.2 System Test Plan

Use Case: Appointment
Scenario: Make a Appointment

Test Progress for Appointment:

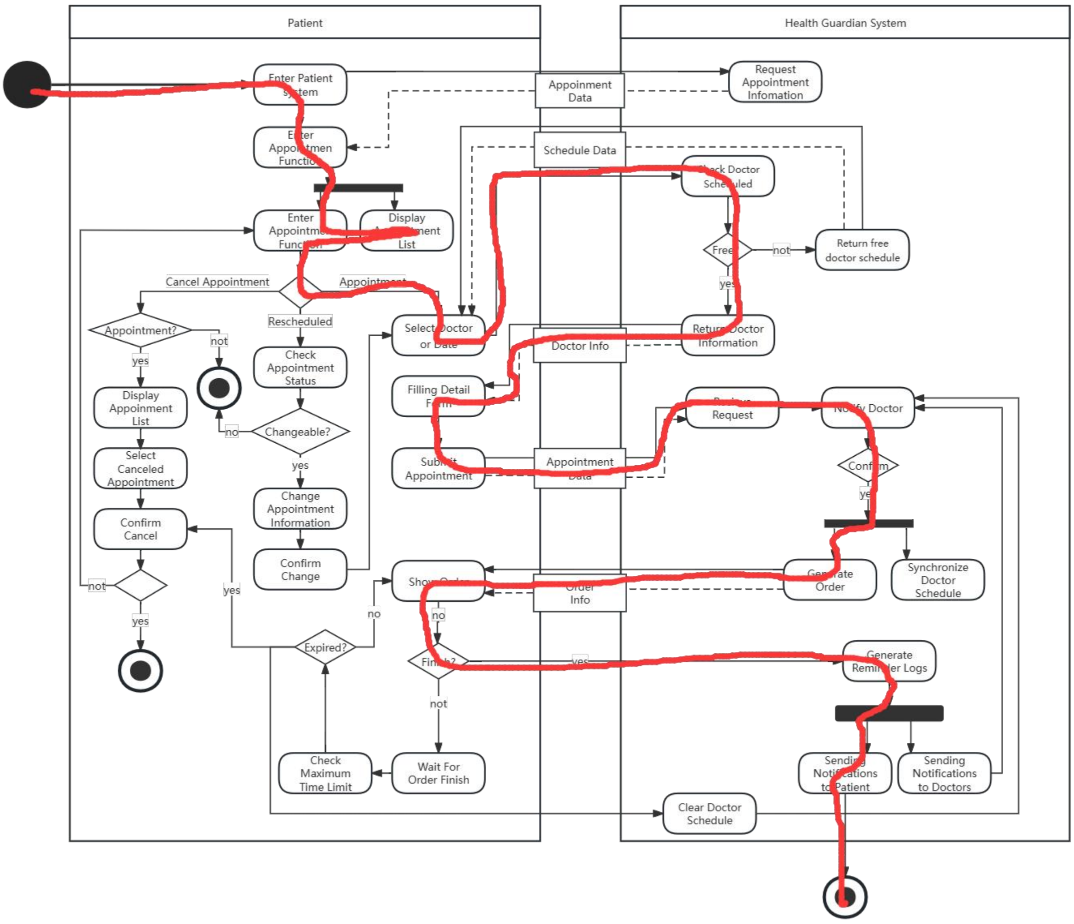
Actor(Patient)	System (HealthGuardian Scheduling Software)
Login the system to create an appointment.	Authenticates the patient and displays the appointment schedule
The patient makes a selection of the appointments that need to be made, here selecting to make an appointment.	The system provides forms and allows the patient to choose a doctor and fill in the information.
Patient's choice of doctor and appointment time	The system will check the doctor's schedule and return an idle doctor schedule if the doctor is busy and a doctor information if the doctor is idle.

Users refine more detailed forms and submit the reservation information.	The system receives the request and reminds the doctor to confirm. The system generates the order and synchronizes the doctor's schedule after the doctor's confirmation.
Patient confirms order and makes payment.	The system generates notifications and sends them to doctors and patients.

Test Data:

Input: ‘appointmentId = 2023’ (and any required appointment details)

Expected Output: Confirmation of scheduled appointment and patient receive message.



Patient
Health Guardian System
Enter Patient System
Appointment Data
Request Appointment Infomation
Enter Appointmen Function
Schedule Data
Check Doctor Scheduled
Enter Appointment Function
Display Appointment List
Free
not
Return free doctor schedule
Cancel Appointment
Appointment
yes
Appointment?
not
Rescheduled
Select Doctor or Date
Return Doctor Information
Doctor Info
yes
Display Appoinment List
Check Appointment Status
Filling Detail Form
Receive Request
Notify Doctor
no
Changeable?
Select Canceled Appointment
yes
Submit Appointment
Appointment Data
Confirm
yes
Confirm Cancel
Change Appointment Information
Confirm Change
Show Order
Order Info
Generate Order
Synchronize Doctor Schedule
not
yes
no
no
yes
Expired?
Finish?
yes
Generate Reminder Logs
not
Check Maximum Time Limit
Wait For Order Finish
Sending Notifications to Patient
Sending Notifications to Doctors
Clear Doctor Schedule