Compiler Design

Sino-British Collaborative Education

CDUT & OBU

Software Engineering

Individual Coursework

Health Guardian doctor

Student number: 202018010415

Date:21st Nov 2023

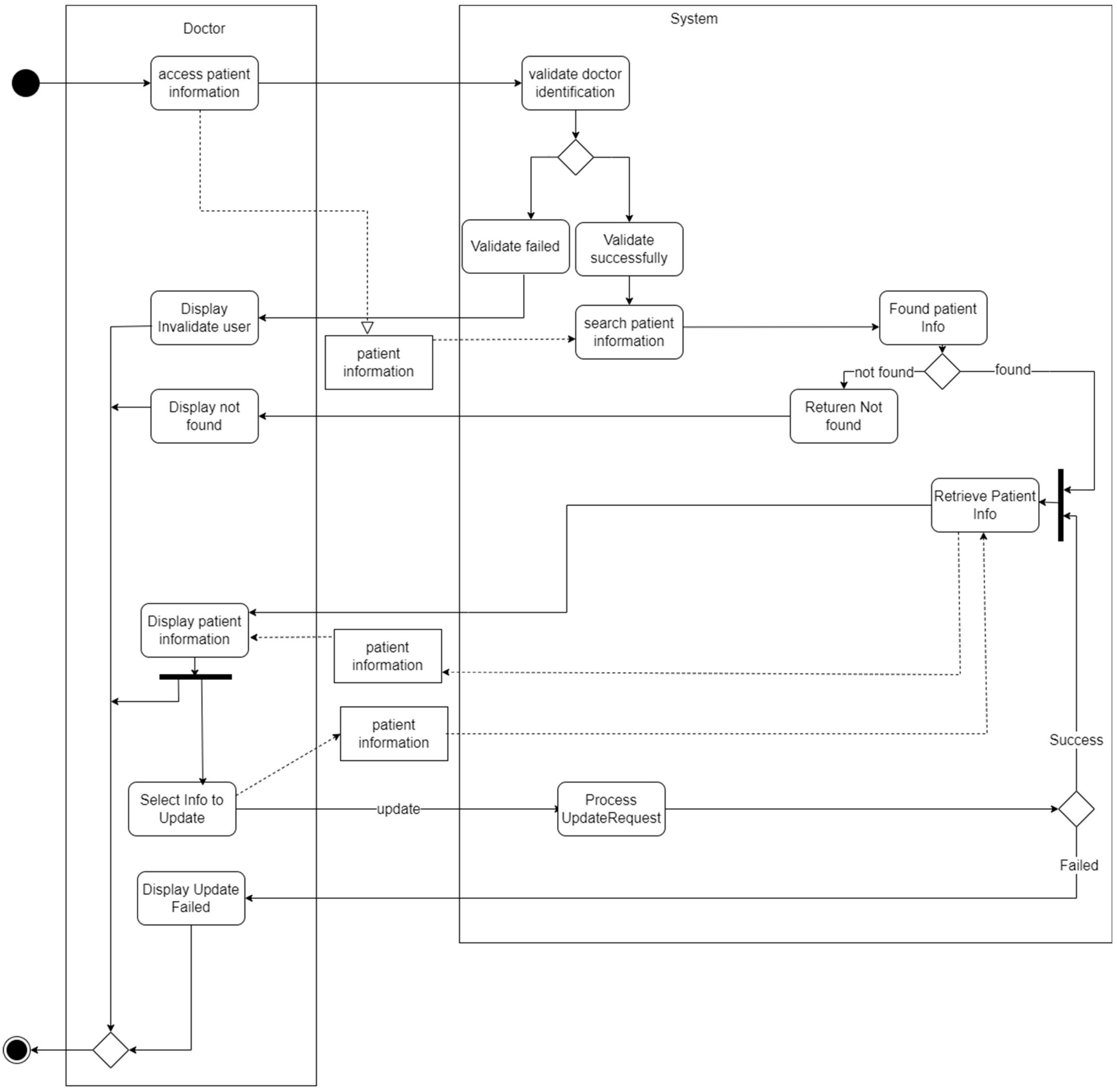
Teacher: Aymen Chebira

Student: Leon

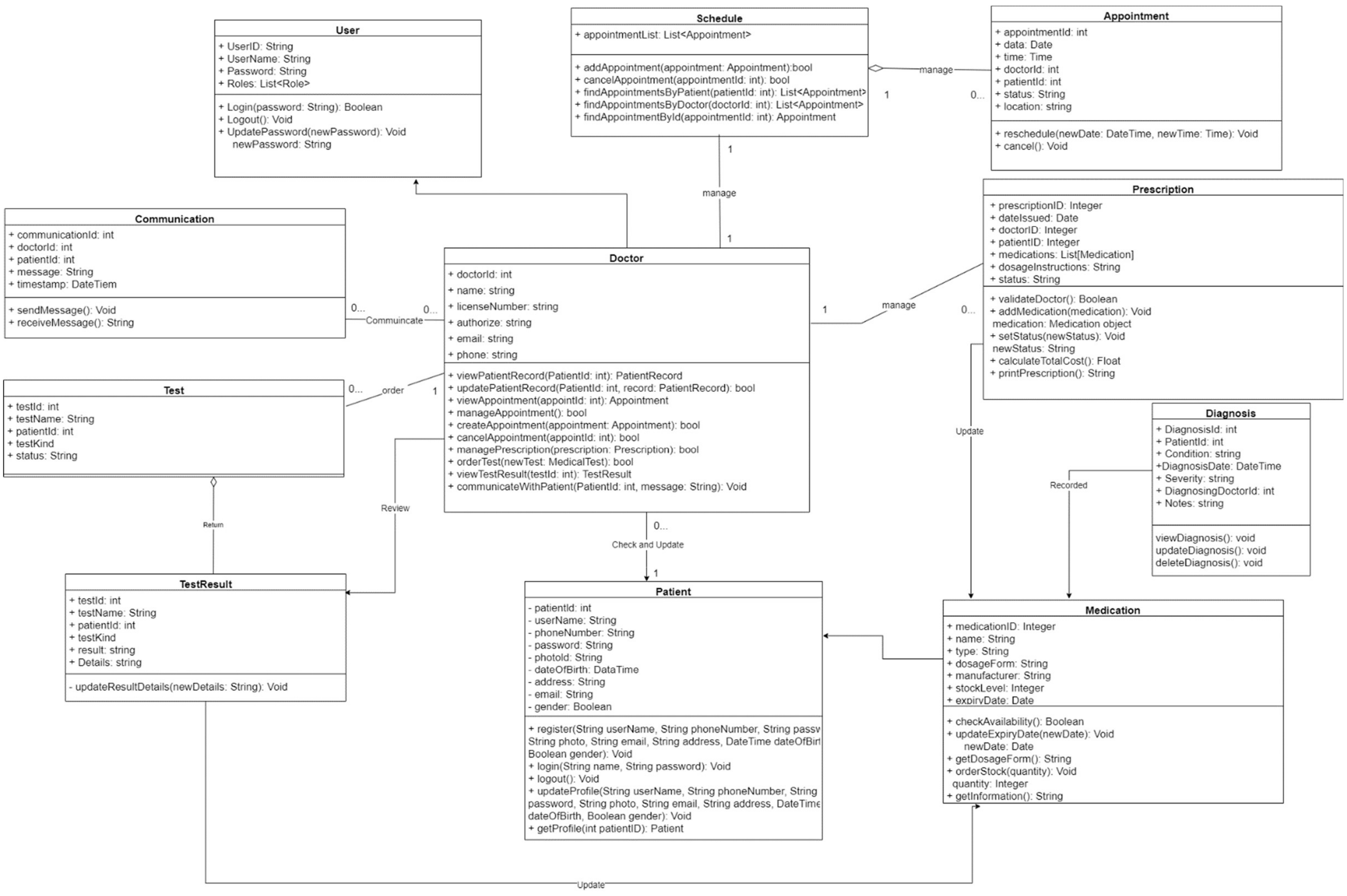
Class: L6C6

1. Task 1

Activity Diagram

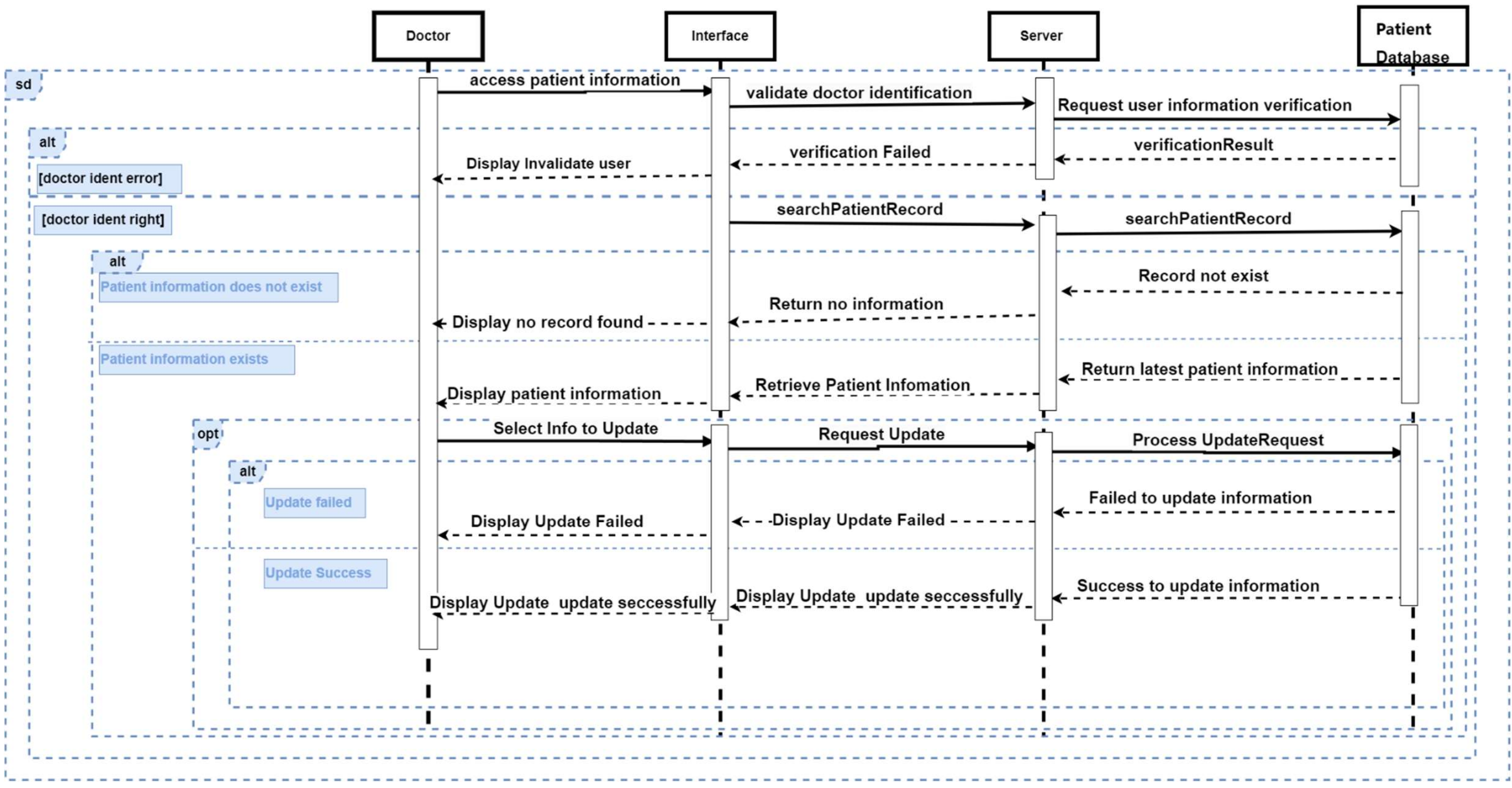


Structural Model



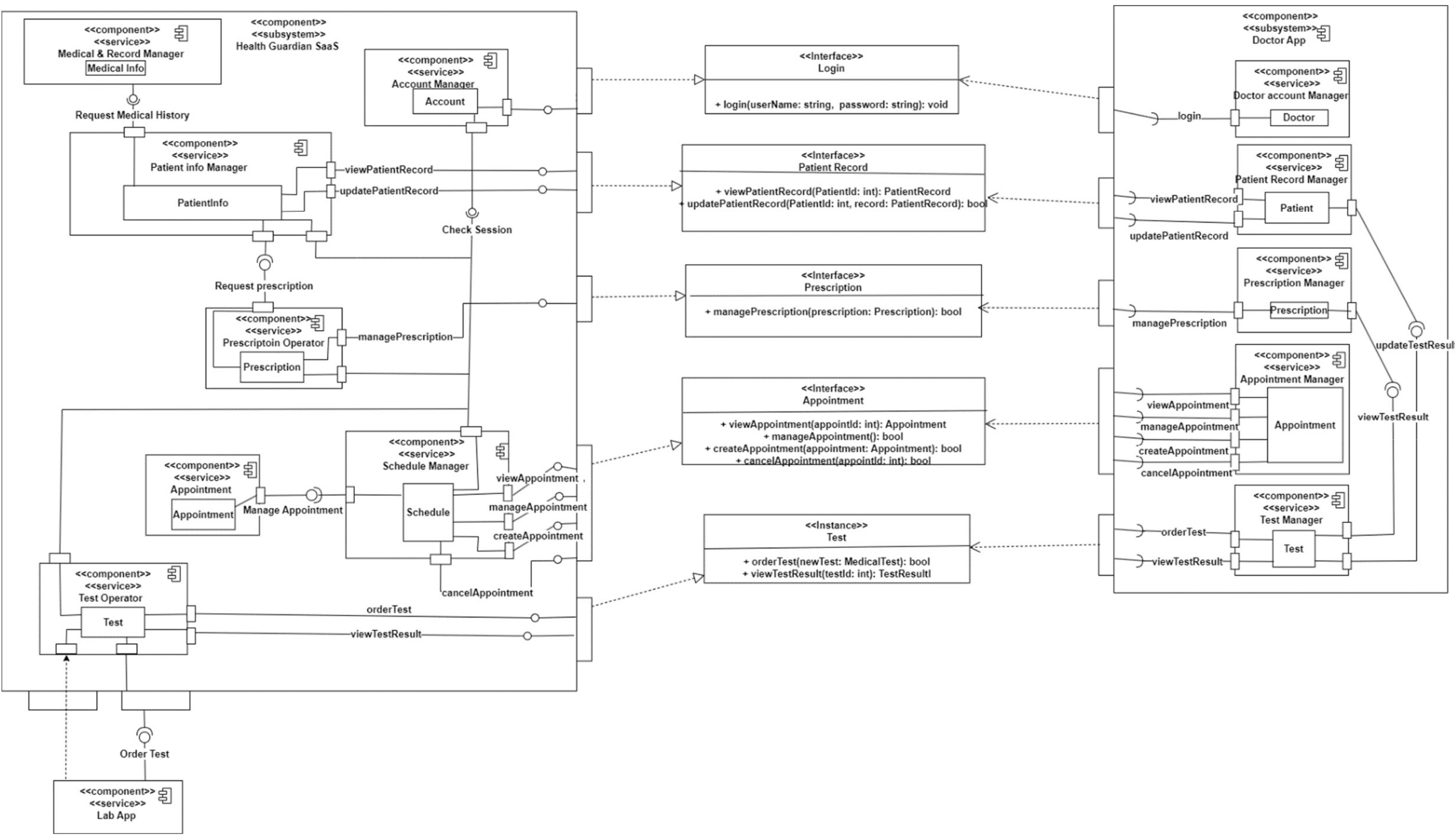
User
+ UserID: String
+ UserName: String
+ Password: String
+ Roles: List<Role>
+ Login(password: String): Boolean
+ Logout(): Void
+ UpdatePassword(newPassword): Void
newPassword: String
Schedule
+ appointmentList: List<Appointment>
+ addAppointment(appointment: Appointment):bool
+ cancelAppointment(appointmentId: int): bool
+ findAppointmentsByPatient(patientId: int): List<Appointment>
+ findAppointmentsByDoctor(doctorId: int): List<Appointment>
+ findAppointmentById(appointmentId: int): Appointment
manage
1
0...
Appointment
+ appointmentId: int
+ data: Date
+ time: Time
+ doctorId: int
+ patientId: int
+ status: String
+ location: string
+ reschedule(newDate: DateTime, newTime: Time): Void
+ cancel(): Void
1
manage
1
Prescription
+ prescriptionID: Integer
+ dateIssued: Date
+ doctorID: Integer
+ patientID: Integer
+ medications: List[Medication]
+ dosageInstructions: String
+ status: String
+ validateDoctor(): Boolean
+ addMedication(medication): Void
medication: Medication object
+ setStatus(newStatus): Void
newStatus: String
+ calculateTotalCost(): Float
+ printPrescription(): String
Communication
+ communicationId: int
+ doctorId: int
+ patientId: int
+ message: String
+ timestamp: DateTiem
+ sendMessage(): Void
+ receiveMessage(): String
0...
Commuincate
0...
Doctor
+ doctorId: int
+ name: string
+ licenseNumber: string
+ authorize: string
+ email: string
+ phone: string
+ 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
1
manage
0...
Test
+ testId: int
+ testName: String
+ patientId: int
+ testKind
+ status: String
0...
order
1
Review
Return
Update
Diagnosis
+ DiagnosisId: int
+ PatientId: int
+ Condition: string
+DiagnosisDate: DateTime
+ Severity: string
+ DiagnosingDoctorId: int
+ Notes: string
viewDiagnosis(): void
updateDiagnosis(): void
deleteDiagnosis(): void
Recorded
0...
Check and Update
1
TestResult
+ testId: int
+ testName: String
+ patientId: int
+ testKind
+ result: string
+ Details: string
- updateResultDetails(newDetails: String): Void
Patient
- patientId: int
- userName: String
- phoneNumber: String
- password: String
- photoId: String
- dateOfBirth: DataTime
- address: String
- email: String
- gender: Boolean
+ register(String userName, String phoneNumber, String passv
String photo, String email, String address, DateTime dateOfBirt
Boolean gender): Void
+ login(String name, String password): Void
+ logout(): Void
+ updateProfile(String userName, String phoneNumber, String
password, String photo, String email, String address, DateTime
dateOfBirth, Boolean gender): Void
+ getProfile(int patientID): Patient
Medication
+ medicationID: Integer
+ name: String
+ type: String
+ dosageForm: String
+ manufacturer: String
+ stockLevel: Integer
+ expiryDate: Date
+ checkAvailability(): Boolean
+ updateExpiryDate(newDate): Void
newDate: Date
+ getDosageForm(): String
+ orderStock(quantity): Void
quantity: Integer
+ getInformation(): String
Update

Behavior Model



2. Task 2

Software Architectural Design



<<component>> <<subsystem>> Health Guardian SaaS
<<component>> <<service>> Medical & Record Manager
Medical Info
Request Medical History
<<component>> <<service>> Account Manager
Account
<<component>> <<service>> Patient info Manager
PatientInfo
viewPatientRecord
updatePatientRecord
Check Session
Request prescription
<<component>> <<service>> Prescriptoin Operator
Prescription
managePrescription
<<component>> <<service>> Schedule Manager
Schedule
<<component>> <<service>> Appointment
Appointment
Manage Appointment
viewAppointment
manageAppointment
createAppointment
cancelAppointment
<<component>> <<service>> Test Operator
Test
orderTest
viewTestResult
Order Test
<<component>> <<service>> Lab App
<<Interface>> Login
+ login(userName: string, password: string): void
<<Interface>> Patient Record
+ viewPatientRecord(PatientId: int): PatientRecord
+ updatePatientRecord(PatientId: int, record: PatientRecord): bool
<<Interface>> Prescription
+ managePrescription(prescription: Prescription): bool
<<Interface>> Appointment
+ viewAppointment(appointId: int): Appointment
+ manageAppointment(): bool
+ createAppointment(appointment: Appointment): bool
+ cancelAppointment(appointId: int): bool
<<Instance>> Test
+ orderTest(newTest: MedicalTest): bool
+ viewTestResult(testId: int): TestResultl
<<component>> <<subsystem>> Doctor App
<<component>> <<service>> Doctor account Manager
Doctor
login
<<component>> <<service>> Patient Record Manager
Patient
viewPatientRecord
updatePatientRecord
<<component>> <<service>> Prescription Manager
Prescription
managePrescription
updateTestResult
<<component>> <<service>> Appointment Manager
Appointment
viewAppointment
manageAppointment
createAppointment
cancelAppointment
viewTestResult
<<component>> <<service>> Test Manager
Test
orderTest
viewTestResult

3. Task 3

Unit test plan

Interface: Appointment

viewAppointment(appointId: int): Appointment		
Test Case	**Method and Parameters**	**Expected Output**
View with valid ID	**viewAppointment(1)**	Appointment object with details
View with ID at upper boundary	**viewAppointment(MAX_INT)**	Appointment object or error message
View with ID at lower boundary	**viewAppointment(0)**	Appointment object or error message
View with invalid negative ID	**viewAppointment(-1)**	Error message
View with non-existing high ID	**viewAppointment(99999999)**	Error message or null object
View with non-integer input	**viewAppointment("a")**	Error message or type exception
View with SQL injection attempt	**viewAppointment(1 OR 1=1)**	Error message or security exception
View without passing any ID	**viewAppointment()**	Compile-time error or method overload

createAppointment(appointment: Appointment): bool		
Test Case	**Method and Parameters**	**Expected Output**

Create with valid appointment object	**createAppointment(new Appointment(...))**	True (success)
Create with null appointment object	**createAppointment(null)**	False/Error message
Create with incomplete appointment	**createAppointment(new Appointment(invalid))**	False/Error message
Create with appointment in the past	**createAppointment(new Appointment(past))**	False/Error message
Create with conflicting appointment	**createAppointment(new Appointment(conflict))**	False/Error message
Create with max number of appointments	**createAppointment(new Appointment(limit))**	False/Error message

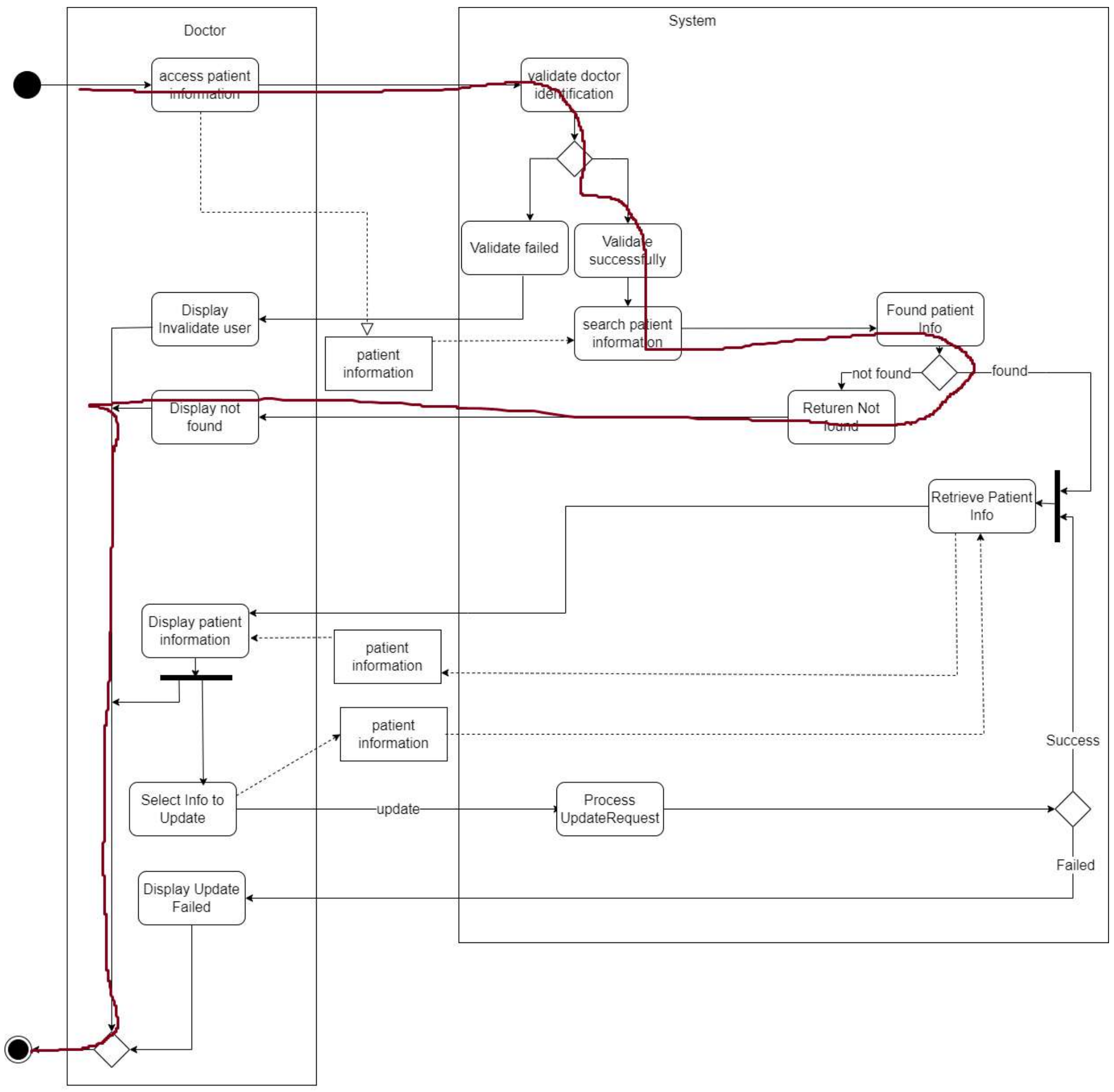
cancelAppointment(appointId: int): bool		
Test Case	**Method and Parameters**	**Expected Output**
Cancel with valid ID	**cancelAppointment(1)**	True (success)
Cancel with non-existing ID	**cancelAppointment(999999)**	True / Error message
Cancel with invalid negative ID	**cancelAppointment(-1)**	False/ Error message

Cancel with ID at upper boundary	**cancelAppointment(MAX_INT)**	False/ Error message
Cancel with ID at lower boundary	**cancelAppointment(0)**	False/ Error message
Cancel without passing any ID	**cancelAppointment()**	Compile-time error or method overload

manageAppointment(): bool		
Test Case	**Setup / Preconditions**	**Expected Output**
Manage with valid system state	Precondition: System ready	True (success)
Manage during system maintenance	Precondition: System maintenance mode	False/Error message
Manage with heavy load	Precondition: System under heavy load	False/Error message
Manage with network failure	Precondition: Simulated network failure	False/Error message
Manage with database connection failure	Precondition: DB connection failure	False/Error message

System test plan

1. Use case: Access Patient information
Scenario: Access a patient who does not exist.



Doctor	Health Gaudian System
1. Access the record of specific patient	2Valid the doctor identity (include license and authorized.
	3, Search patient Information by [patientId]
	4Get account information from patient account manager. (Such as name. birthday photo, phone number)
	5.Patient not found
	6. return Error message: “not found”
7. display Error message “not found“	

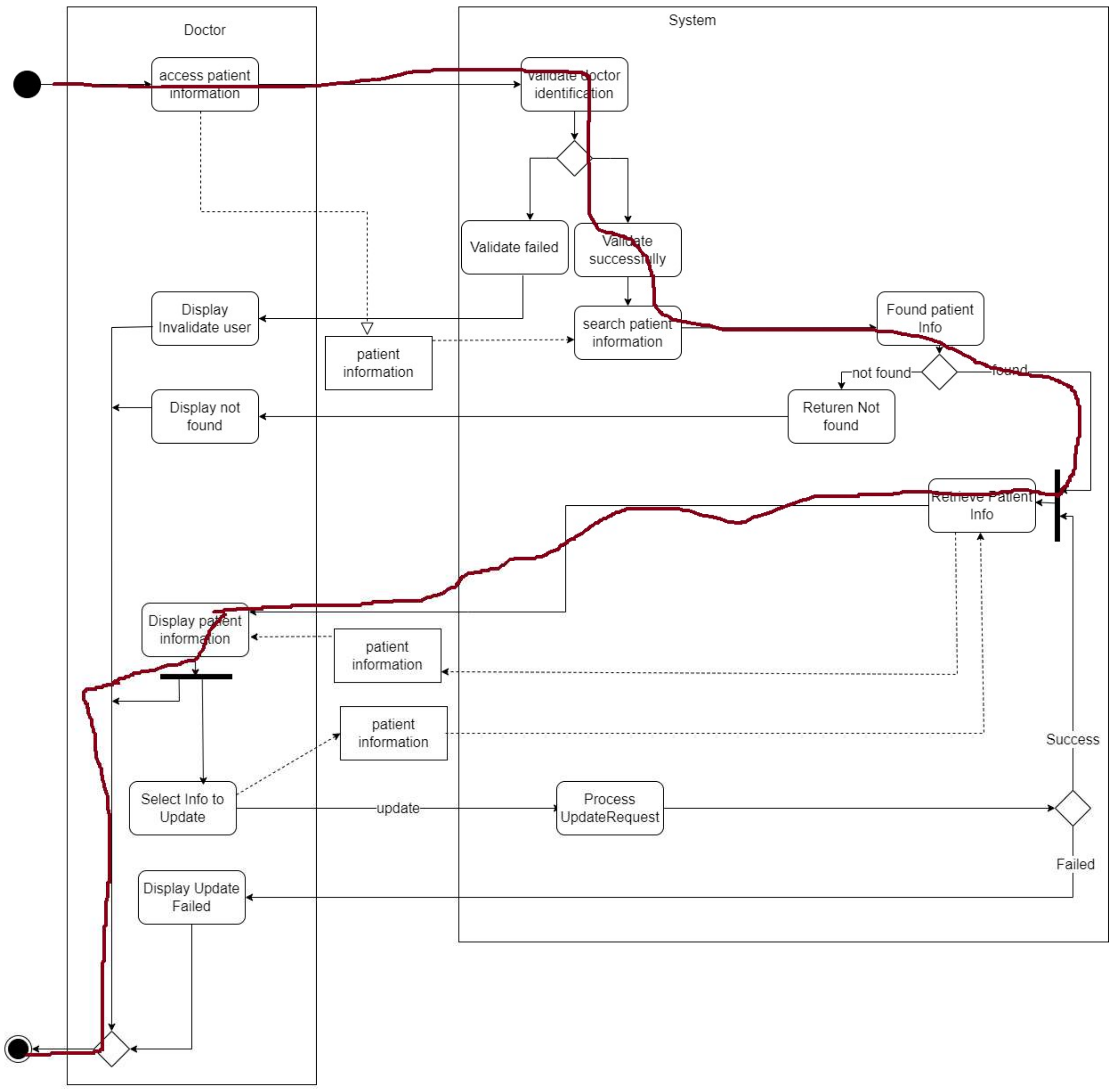
Test data:

Input: patientId:9999999999999999999(invalid)

Stored data: noting stored.

Output: Error message: “not found"

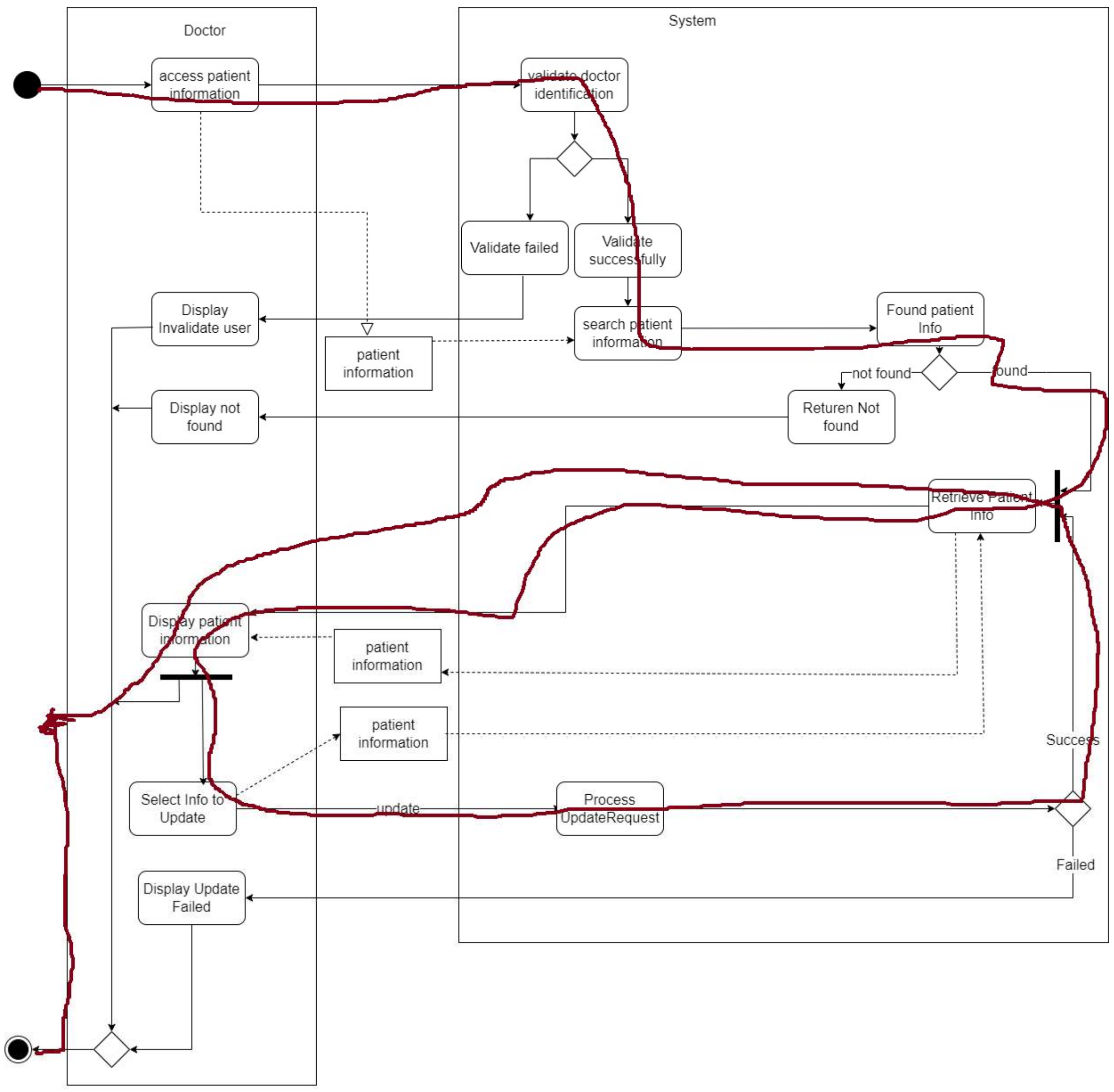
2. Use case: Access Patient information
Scenario: Access patient information then exit the program



Doctor	Health Gaudian System
1. Access the record of specific patient+	2.Valid the doctor identity (include license and authorized.
	3, Search patient Information by [patientld]
	4.Get account information from patient account manager. (Such as name. birthday photo, phone number)
	5.Patient founded
	6.return patient information
7.display patient information	

Test data:
Input: patientId:123456(valid)
Stored data: noting stored.
Output: patient information

3.Use case: Access Patient information
Scenario: Access patient information then Update patient information (success)



Doctor	Health Gaudian System
1. Access the record of specific patient+	2.Valid the doctor identity (include license and authorized.
	3, Search patient Information by [patientId]
	4.Get account information from patient account manager. (Such as name. birthday photo, phone number)
	5.Patient founded
	6.return patient information

7.display patient information	
8.update patient information	
	9.Update successfully
	10.return patient information
11.display updated patient information	

Test data:

Input: patientId:123456(valid)

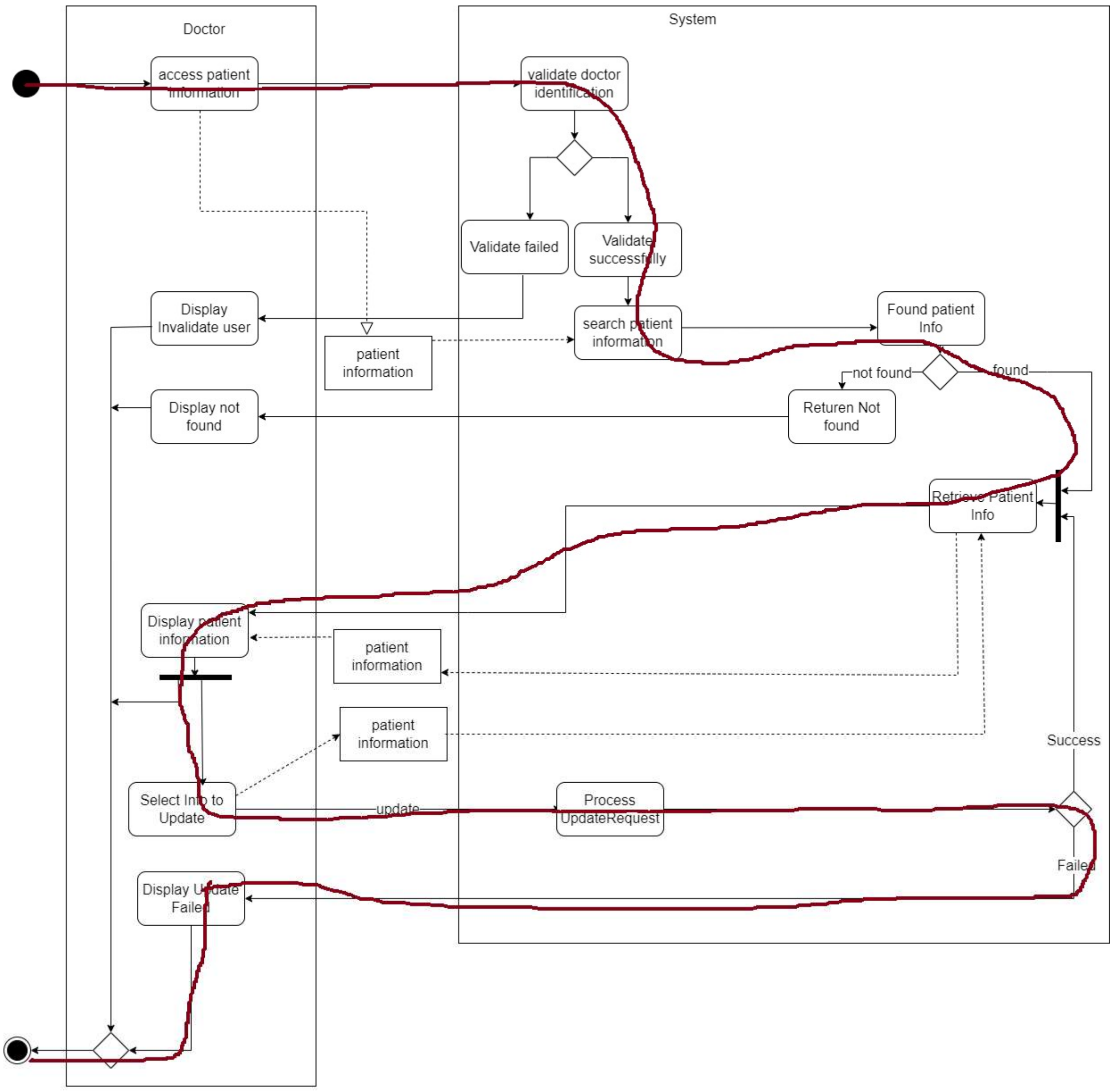
: patient updated information (Such as name ,birthday photo, phone number)

Stored data: patient updated information.

Output: patient updated information

4.Use case: Access Patient information

Scenario: Access patient information then Update patient information (failed)



Doctor	Health Gaudian System

1. Request the information of specific patient+	2.Valid the doctor identity (include license and authorized.
	3, Search patient Information by [patientld]
	4.Get account information from patient account manager. (Such as name. birthday photo, phone number)
	5.Patient founded
	6.return patient information
7.display patient information	
8.update patient information	
	9.Update failed.
	10.return Error message: Update failed.
11.display Error message: Update failed.	

Test data:

Input: patientld:123456(valid)

: patient updated information (Such as name, birthday photo, phone number)

Stored data: none

Output: Error message: “Update failed”