Software Engineering
Project Deliverable 3

Team Members: Symbat Bekzhigit, Sanjana Nambiar, Saamia Shafqat, and Levith Andrade Cuellar

Use Case Scenarios

Section 1 (5 points):

Include the five Use Case Scenarios that your team will move forward with.

USE CASE: MESSAGING A USER

Primary Actor: Doctor / Nurse

Main success scenario:

1. **Doctor / Nurse** accesses the platform's messaging section.
2. **System** displays the current chats and displays an option to start a new conversation.
3. **Doctor / Nurse** starts a new conversation.
4. **System** displays the hospital's directory composed of other doctors and nurses (name, photo, specialty).
5. **Doctor / Nurse** selects the doctor or nurse they'd like to start a conversation with.
6. **System** displays a chat interface, a location for input, a keyboard, an option to include emojis and a send button.
7. **Doctor / Nurse** inputs their message and sends it.
8. **System** encrypts the message content using the relevant encryption standards.
9. **System** delivers the message to the recipient and updates the interface with the sent message.

Extensions:

<u>Searching Directory:</u>

3a. **System** displays the profiles in alphabetical order.

4a. **Doctor / Nurse** searches for a person that isn't registered under the hospital's directory.

4b. **System** alerts the user that it was unable to locate the person they were searching for.

<u>Acknowledgment Receipts:</u>

8a. **System** displays if the message has been delivered.

8b. **System** displays if the message has been read.

USE CASE: PAGING A USER (Assuming the user is logged into the application)

Primary Actor: Doctor / Nurse

Main success scenario:

1. **Doctor / Nurse** accesses the platform's pager section.
2. **System** displays a set of options to construct the request: recipient name, location, notification type,

and custom message.

3. **Doctor / Nurse** searches for the recipient's name using the hospital’s directory.
4. **Doctor / Nurse** selects a location for the request and a type of notification using a set of options.
5. **Doctor / Nurse** inputs text for the custom message.
6. **Doctor / Nurse** sends the request.
7. **System** delivers the request to the designated recipient.

Extensions:

<u>Request Drafting:</u>

5a. **System** alerts the doctor/nurse when the word count is reached.

6a. **System** notifies the doctor/nurse that there is missing information.

<u>Notification Failure</u>

7a. **System** alerts the doctor/nurse they’re offline and asks them to use a different method for processing an urgent request.

7b. **System** logs the notification failure.

<u>Notification Acknowledgement</u>

8a. **System** forwards the notification to another healthcare professional in case the original recipient does not acknowledge the request in a timely manner.

USE CASE: VIEWING AN ELECTRONIC HEALTH RECORD (EHR)

Primary Actor: Doctor / Nurse

Main success scenario:

1. **Doctor / Nurse** accesses the platform’s records section.
2. **System** displays a list of patients (name, photo, age, gender) that the Doctor / Nurse is currently treating.
3. **System** presents the user with options to search for patient data by various criteria, such as patient name, medical record number, or admission date.
4. **Doctor / Nurse** selects a patient whose information they’d like to view.
5. **System** displays the contents of the patient’s Electronic Health Record (EHR): personal information, medical history, diagnoses, etc.

Extensions:

2a. **System** displays the list in order of treatment start date.

<u>Invalid Search Criteria:</u>

4a. **Doctor / Nurse** searches for a patient that they’re not assigned to.

4b. **System** alerts the user that it was unable to locate the patient they are looking for.

4c. **System** allows the user to revise their search criteria or initiate a new search.

<u>Database Connection Failure:</u>

5a. **System** encounters issues connecting to the hospital's database during the data retrieval process.

5b. **System** displays an error message indicating the database connection failure and advises the user to check their internet connection or try again later.
5c. **Doctor / Nurse** can attempt to reconnect to the database.

USE CASE: CREATING A REMINDER

Primary Actor: Doctor / Nurse

Main success scenario:

1. **Doctor / Nurse** accesses the platform's reminder section.
2. **System** displays the current reminders and displays an option to create a new reminder.
3. **Doctor / Nurse** selects an option to create a new reminder.
4. **System** displays a set of options to construct the reminder: title, type, notes, date, time, and the reminder's recurrence.
5. **Doctor / Nurse** inputs text for the title and notes.
6. **Doctor / Nurse** selects the appropriate date, time and type associated with the reminder.
7. **Doctor / Nurse** set a reminder preference, including the method of notification.
8. **Doctor / Nurse** adds the reminder.
9. **System** saves the reminder and displays it with the other current reminders.

Extensions:

Arranging existing reminders

2a. **System** displays the current reminders in chronological order.
3a. **Doctor / Nurse** selects the option to delete a reminder.
3b. **Doctor / Nurse** selects the option to update a reminder.

Invalid Reminder Details

5a. **Doctor / Nurse** enters incomplete or invalid information for the reminder.
5b. **System** alerts the doctor/nurse when the notes word count is reached.
6a. **System** verifies the chosen date is in the future.
9a. **System** displays an error message prompting the user to provide valid details.

USE CASE: PERFORM AI-POWERED REQUEST

Primary Actor: Doctor / Nurse

Main success scenario:

1. **Doctor / Nurse** accesses the platform's records section.
2. **Doctor / Nurse** selects a specific patient's record to work with.
3. **System** presents predetermined prompts with possible queries that can be made to the AI module.
4. **Doctor / Nurse** selects the query.
5. **System** utilizes the AI module to process the user's query.

6. **System** displays the AI-generated query results.
7. **Doctor / Nurse** reviews the query results.

Extensions:

Inaccurate Summarization:

7a. **Doctor / Nurse** may choose to manually review the patient records for accuracy or request further clarification from the AI module with additional queries.

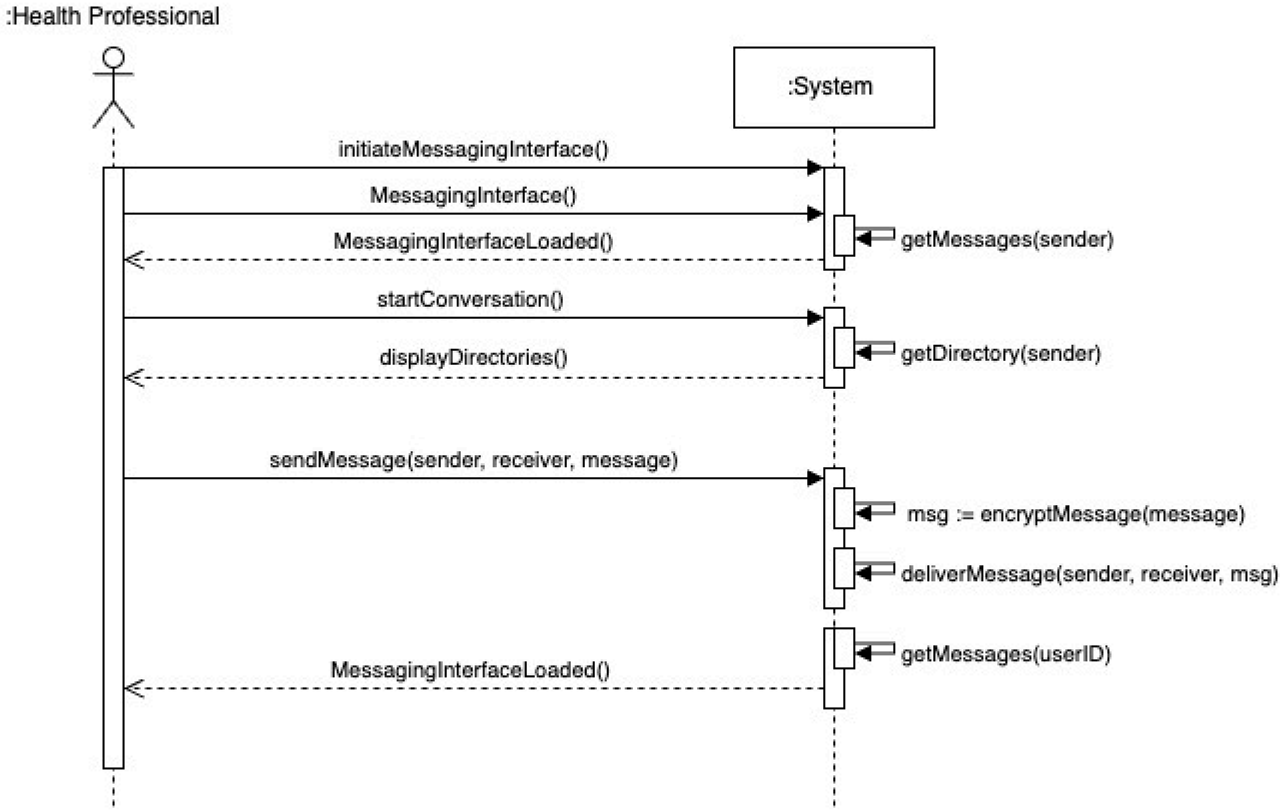
Unresponsive AI:.

1a. **System** displays an error message indicating the unavailability of the AI service and suggests trying again later.

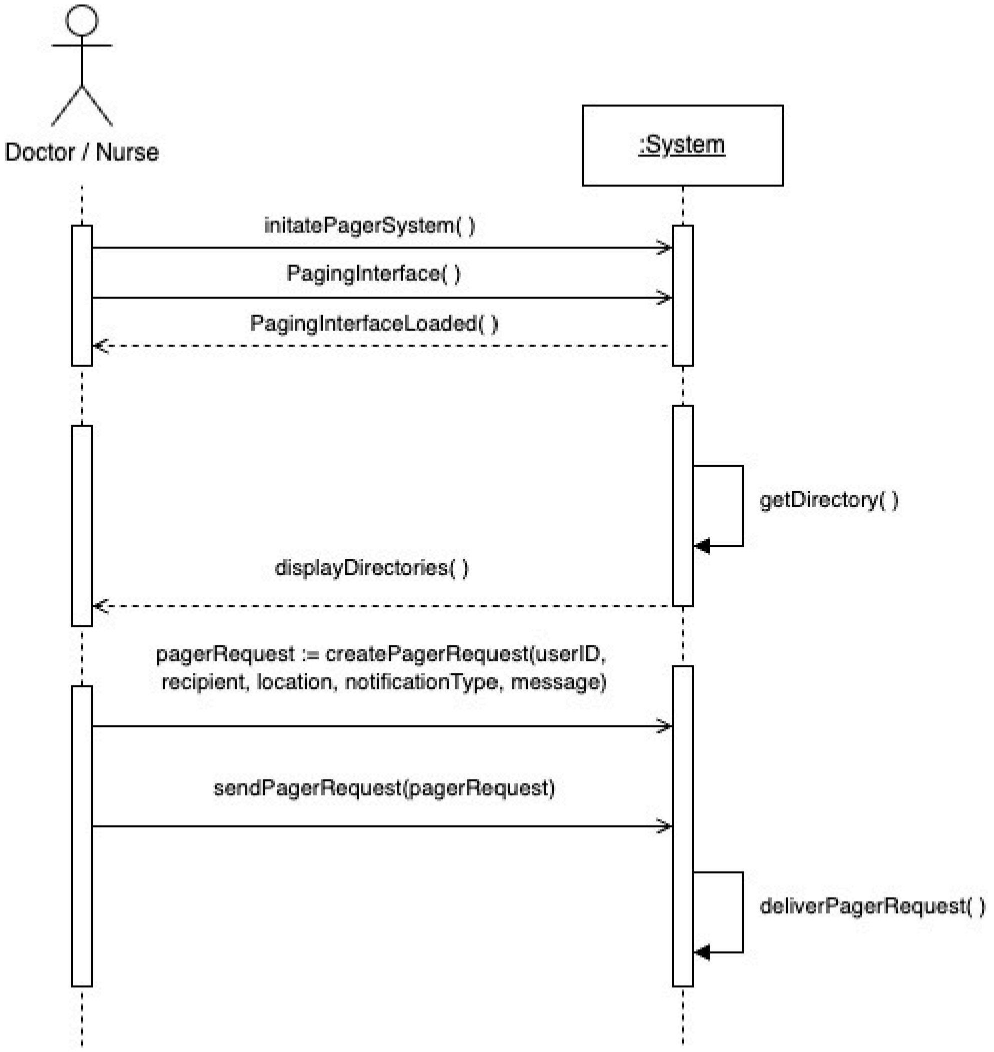
2a. **Doctor / Nurse** may opt to wait for the AI service to become responsive again or proceed with manual review of the patient records in the meantime.

System Sequence Diagram (SSD)

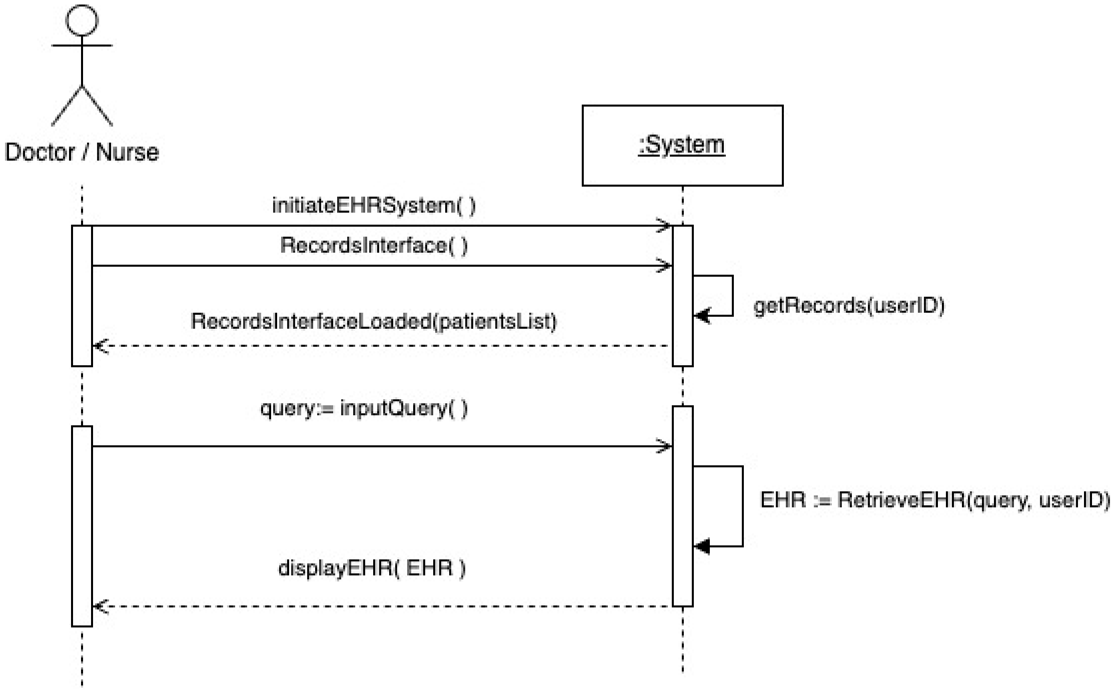
1. **Messaging SSD**


:Health Professional
:System
initiateMessagingInterface()
MessagingInterface()
MessagingInterfaceLoaded()
getMessages(sender)
startConversation()
displayDirectories()
getDirectory(sender)
sendMessage(sender, receiver, message)
msg := encryptMessage(message)
deliverMessage(sender, receiver, msg)
getMessages(userID)
MessagingInterfaceLoaded()

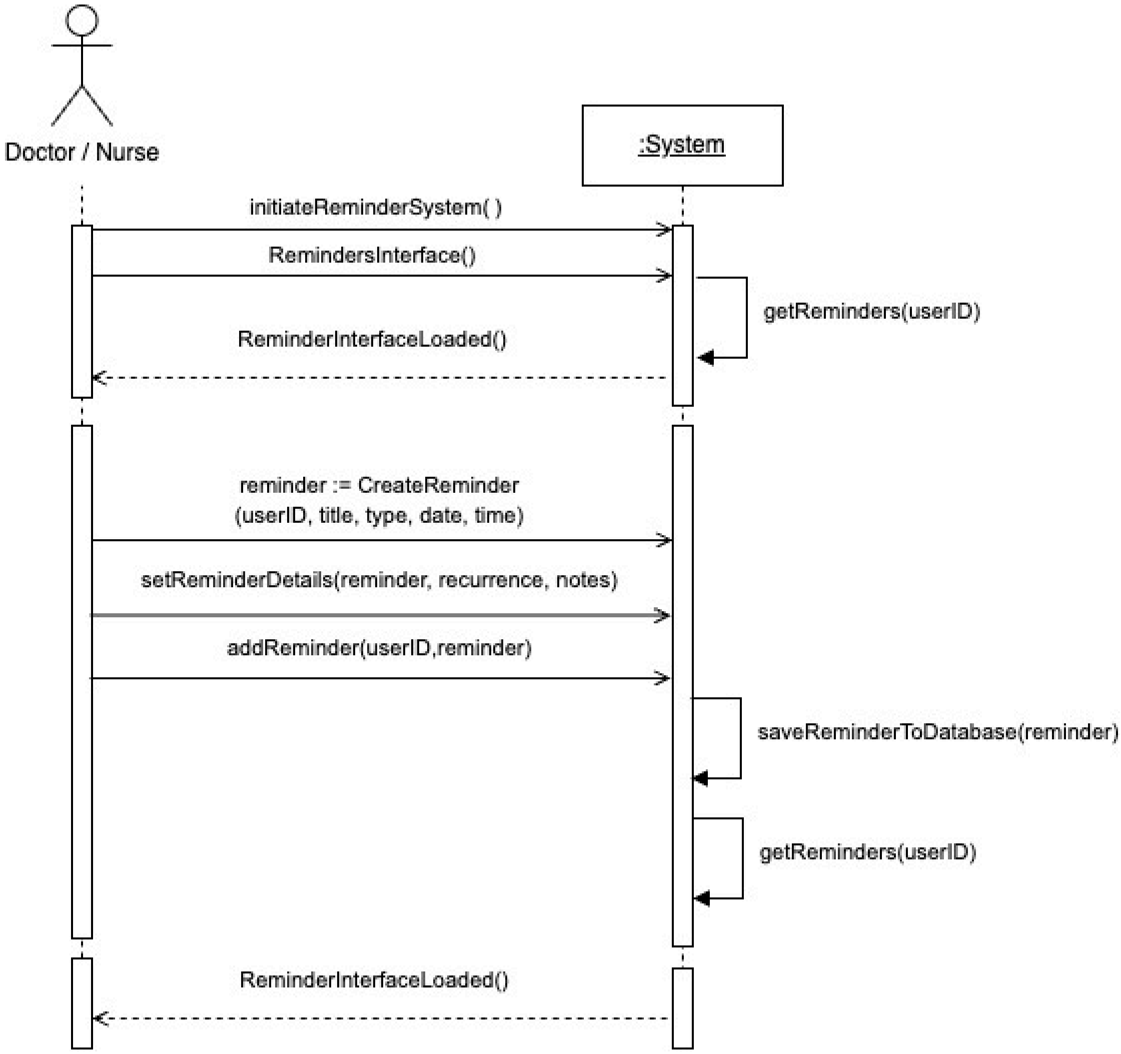
2. **Pager SSD**



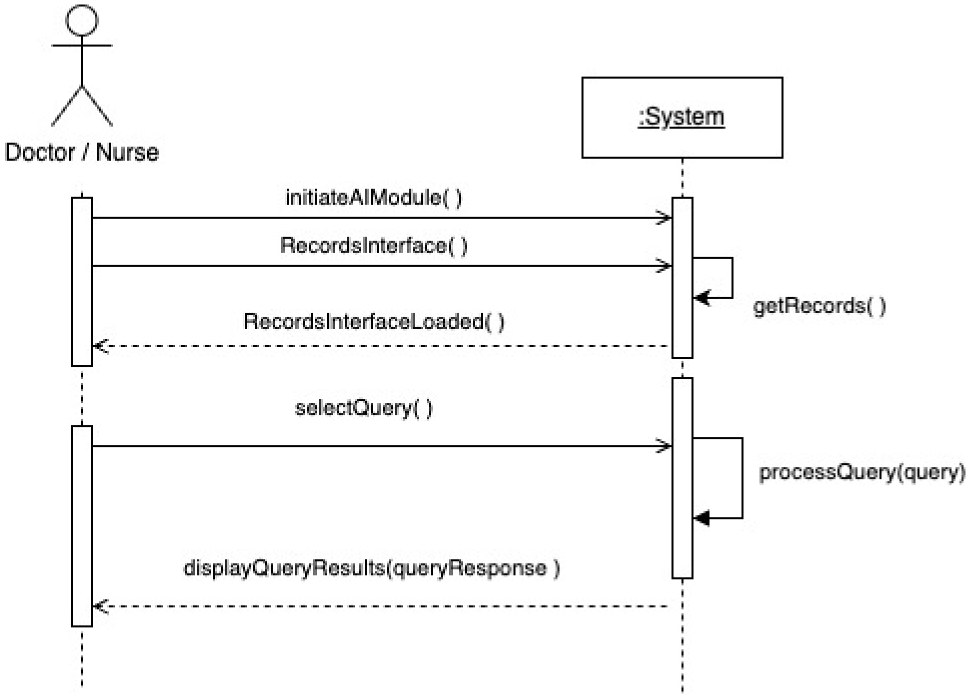
3. **Viewing EHR SSD**



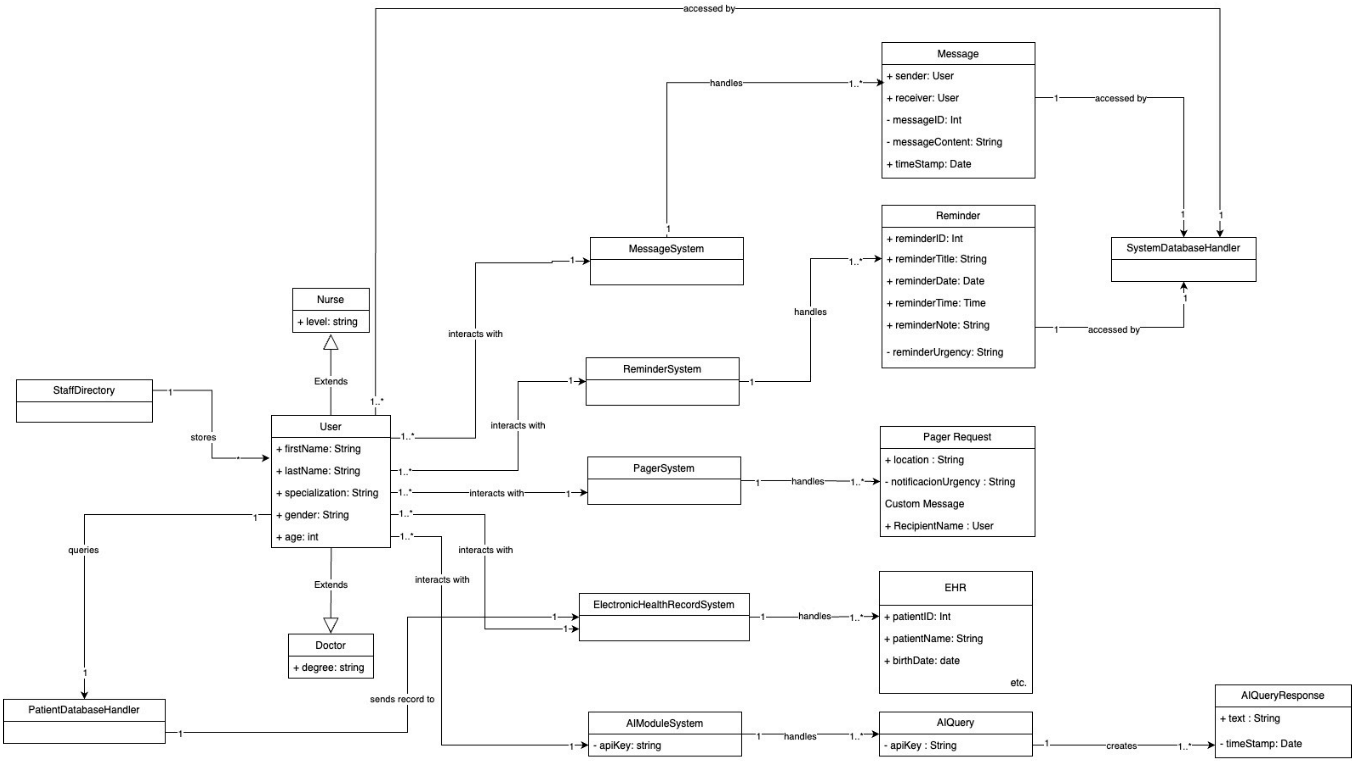
4. **Creating Reminder SSD**


Doctor / Nurse
:System
initiateReminderSystem( )
RemindersInterface()
getReminders(userID)
ReminderInterfaceLoaded()
reminder := CreateReminder
(userID, title, type, date, time)
setReminderDetails(reminder, recurrence, notes)
addReminder(userID,reminder)
saveReminderToDatabase(reminder)
getReminders(userID)
ReminderInterfaceLoaded()

5. AI Module SSD



Domain Model



Operation Contracts

USE CASE: MESSAGE

Name:	sendMessage(message)
Responsibilities:	Facilitates sending a message from one user (sender) to another (recipient). System
Type:	Use Cases: Messaging a user
Cross References:	
Notes:	If the message is empty or null, return an error.
Exceptions:	If the message couldn't be sent then return an error. If the system cannot encrypt or deliver the message, return an error.
Output:	Confirmation that the message was sent. Updates to the messaging interface to reflect the new message.
Pre-conditions:	User IDs, for both the sender and recipient, are known to the system. All arguments were provided.
Post-conditions:	● A new Message object is created with the message content, senderID, and recipientID (*instance creation*). ● *Message.encryptionStatus* is set to true after encryptMessage(message) is executed (*attribute modification*)

	● *Message.deliveryStatus* object is updated after deliverMessage() properly sends the message over (*attribute modification*).
	● Message added to the *conversationList* between the users (*attribute modification*)

USE CASE: PAGER

Name:	createPagerRequest(userID, recipient, location, notificationType, message)
	Enter pager request details and create a request object.
Responsibilities:	System
	Use Cases: Paging a User
Type:	
Cross References:	
Notes:	If User ID is not valid, indicate an error.
Exceptions:	If an argument is missing, indicate an error.
Output:	User ID is known to the system.
Pre-conditions:	All arguments were provided.
	● If a new request, a *PagerRequest* was created (*instance creation*)
Post-conditions:	● *PagerRequest.recipient* was set to *recipient* (*attribute modification*)
	● *PagerRequest.location* was set to *location* (*attribute modification*)
	● *PagerRequest.notificationType* was set to *notificationType* (*attribute modification*)
	● *PagerRequest.message* was set to *message* (*attribute modification*)

USE CASE: CREATE REMINDER

Name:	createReminder(userID, title, type, date, time)
Responsibilities:	Enter reminder details, create a reminder and add it to the user's reminders.
	System
Type:	Use Cases: Create Reminder
Cross References:	
Notes:	If User ID is not valid, indicate an error.
Exceptions:	If an argument is missing, indicate an error.
Output:	User ID is known to the system.
Pre-conditions:	All arguments were provided.
	● If a new reminder, a *Reminder* was created (*instance creation*)

Post-conditions:	● If a new reminder, the new *Reminder* was associated with the *User Reminders* that matches *user ID* (*association formed*). ● *Reminder.title* was set to *title* (*attribute modification*) ● *Reminder.type* was set to *type* (*attribute modification*) ● *Reminder.date* was set to *date* (*attribute modification*) ● *Reminder.time* was set to *time* (*attribute modification*)

USE CASE: VIEW EHR

Name:	RetrieveEHR(userID, query)
Responsibilities:	To display an Electronic Health Record based on a specific query by the user.
Type:	System Operation
Cross References:	Use Cases: Viewing an Electronic Health Record
Notes:	This operation is integral to the system's functionalities, allowing users to access and review EHRs.
Exceptions:	If the query does not match any records, the system should inform the user that no matching records were found.
Output:	Displays the Electronic Health Record that matches the query or an error message detailing what went wrong.
Pre-conditions:	The user who is searching must have access to the records they're querying. A query must be provided by the user to specify which EHR is to be retrieved.
Post-conditions:	● If a new request, *EHR* was created (*instance creation*) ● *EHR.patientName* was set based on the retrieved record (*attribute modification*) ● *EHR.dateOfRecord* was set to *date of creation of the record* (*attribute modification*) ● *EHR.details* is filled with the medical history, diagnosis, treatment plans, and notes contained within the EHR (attribute modification).

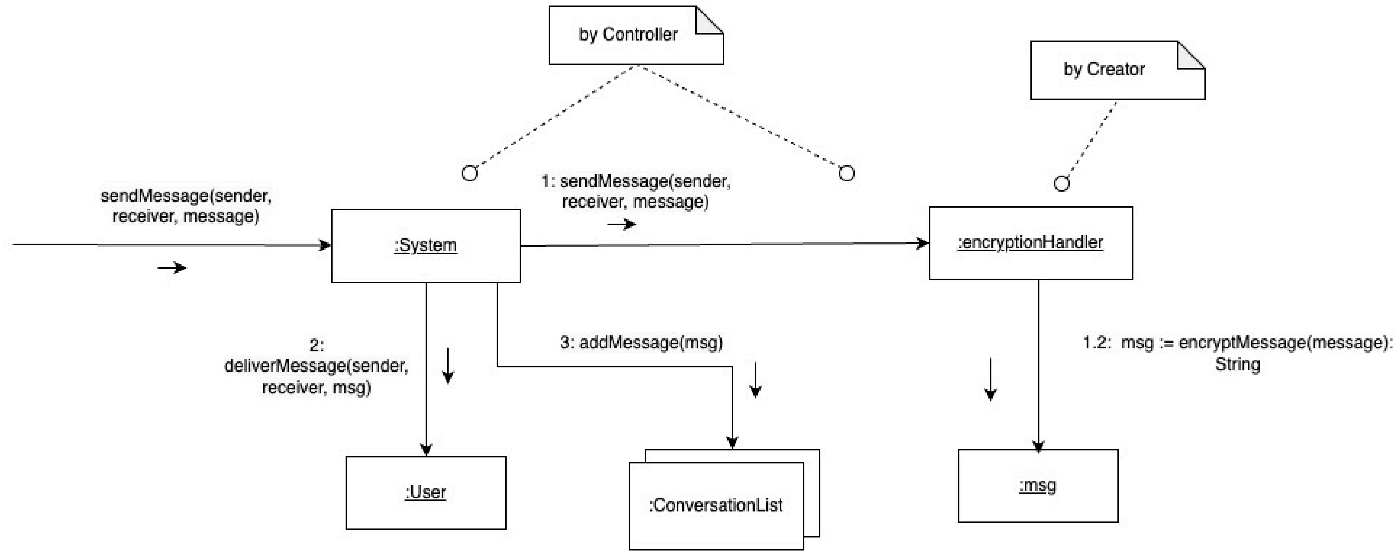
USE CASE: PERFORM AI-POWERED REQUEST

Name:	processQuery(ehr, query)
Responsibilities:	To perform a specific AI query relating to a selected EHR.
Type:	System Operation
Cross References:	Use Cases: Perform an AI Query
Notes:	Will adhere to healthcare data compliance standards.
Exceptions:	If the AI server is down, then an error warning will be generated

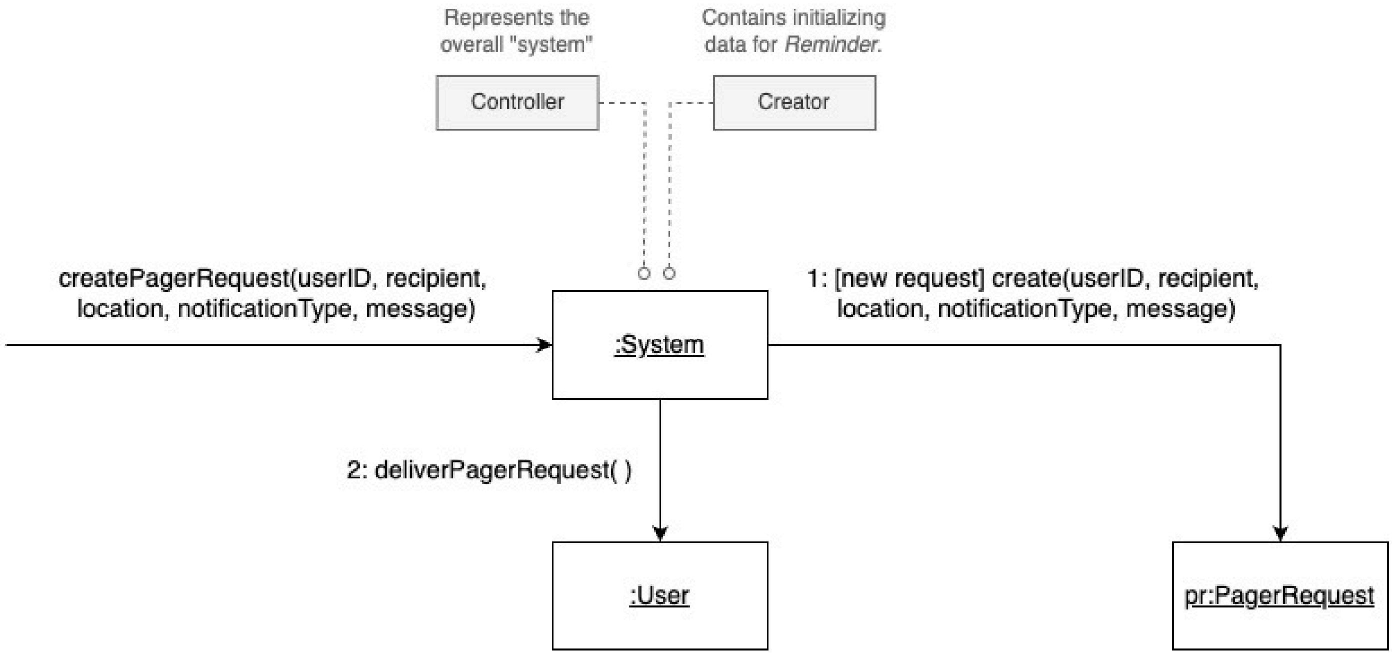
Output:	An AI-generated summary result of the patient's health record based on the query. A warning message that the results might contain inaccuracies.
Pre-conditions:	The user has already retrieved the EHR record of the patient. The user has selected the query to be given to the AI.
Post-conditions:	● If a new query, a *queryResponse* was created *(instance creation)* ● *queryResponse.text* was set to *AI query results (attribute modification).* ● *queryResponse.time* was set to AI query *time (attribute modification).*

UML Interaction Diagram

<u>USE CASE: MESSAGE</u>

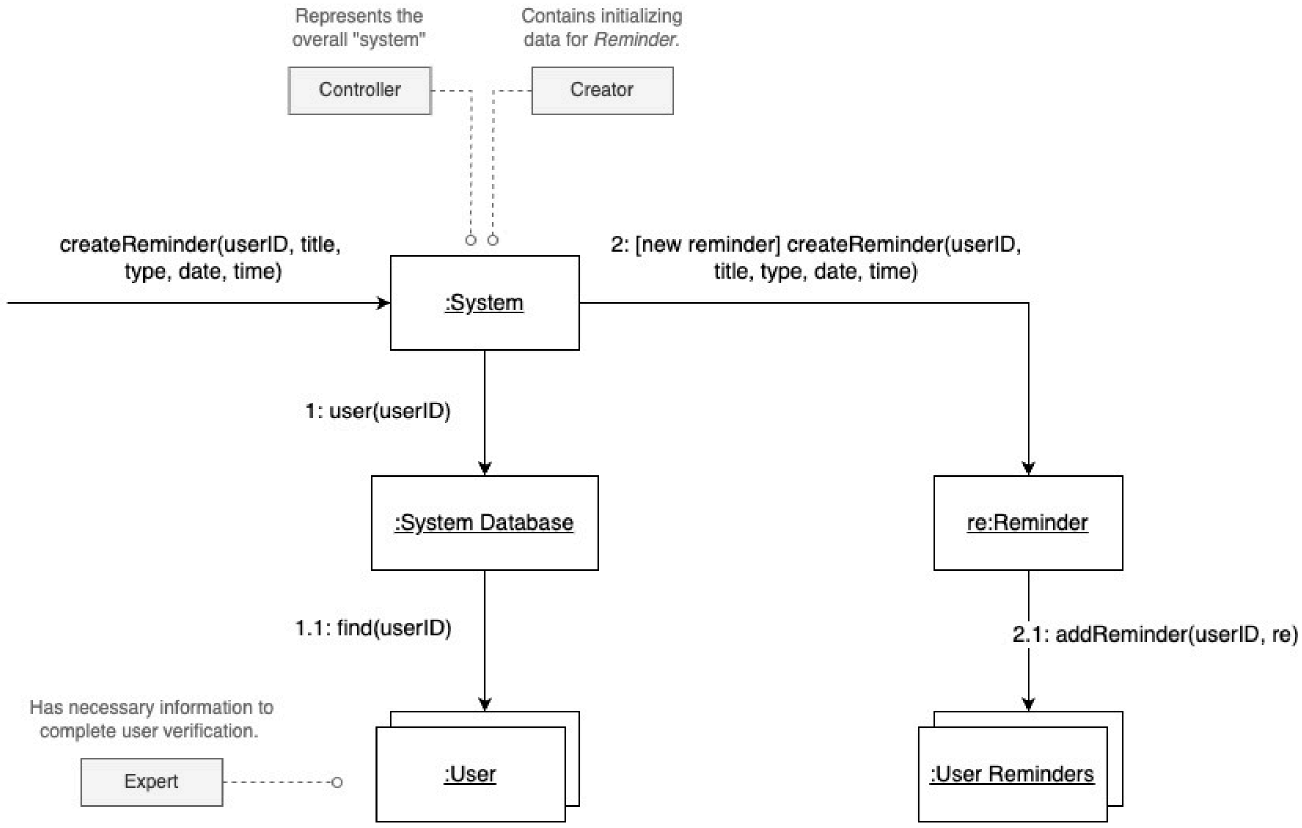

by Controller
by Creator
sendMessage(sender, receiver, message)
:System
1: sendMessage(sender, receiver, message)
:encryptionHandler
2: deliverMessage(sender, receiver, msg)
3: addMessage(msg)
1.2: msg := encryptMessage(message): String
:User
:ConversationList
:msg

USE CASE: PAGER



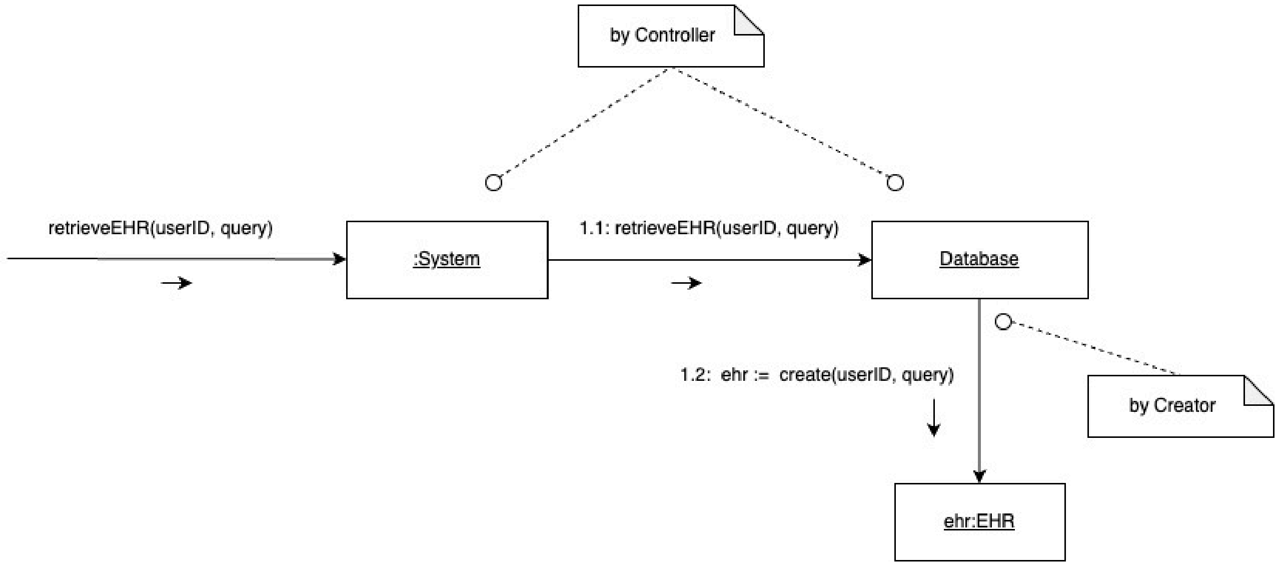
Represents the overall "system"
Contains initializing data for *Reminder*.
Controller
Creator
createPagerRequest(userID, recipient, location, notificationType, message)
:System
1: [new request] create(userID, recipient, location, notificationType, message)
2: deliverPagerRequest( )
:User
pr:PagerRequest

USE CASE: CREATE REMINDER

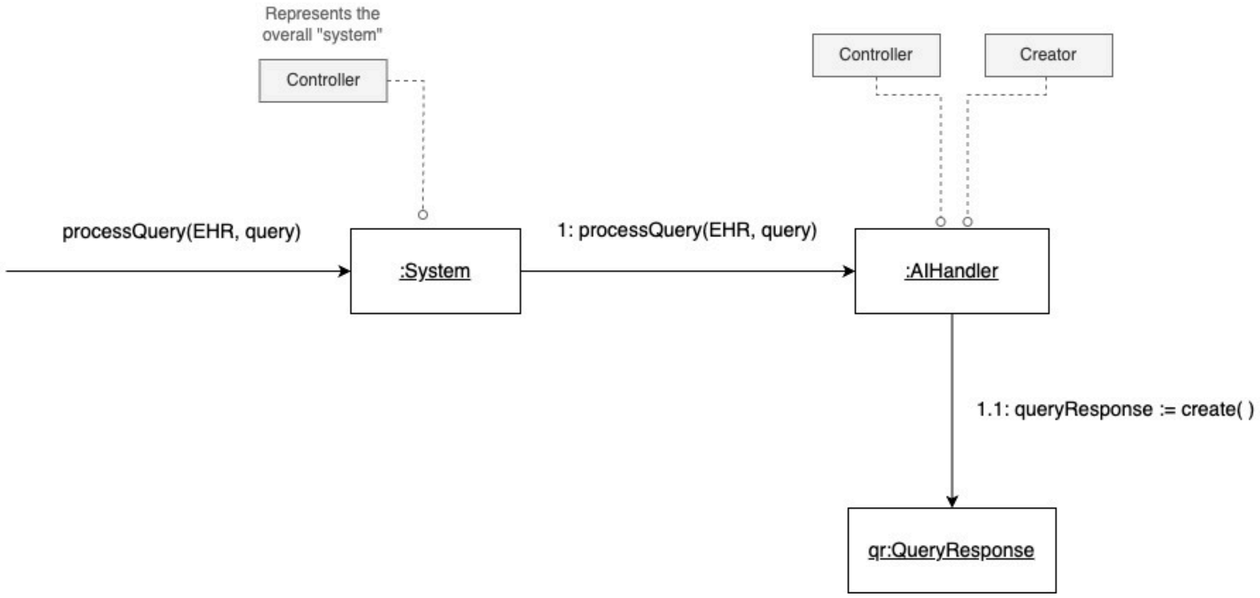


Represents the overall "system"
Contains initializing data for *Reminder*.
Controller
Creator
createReminder(userID, title, type, date, time)
:System
2: [new reminder] createReminder(userID, title, type, date, time)
1: user(userID)
:System Database
re:Reminder
1.1: find(userID)
2.1: addReminder(userID, re)
Has necessary information to complete user verification.
Expert
:User
:User Reminders

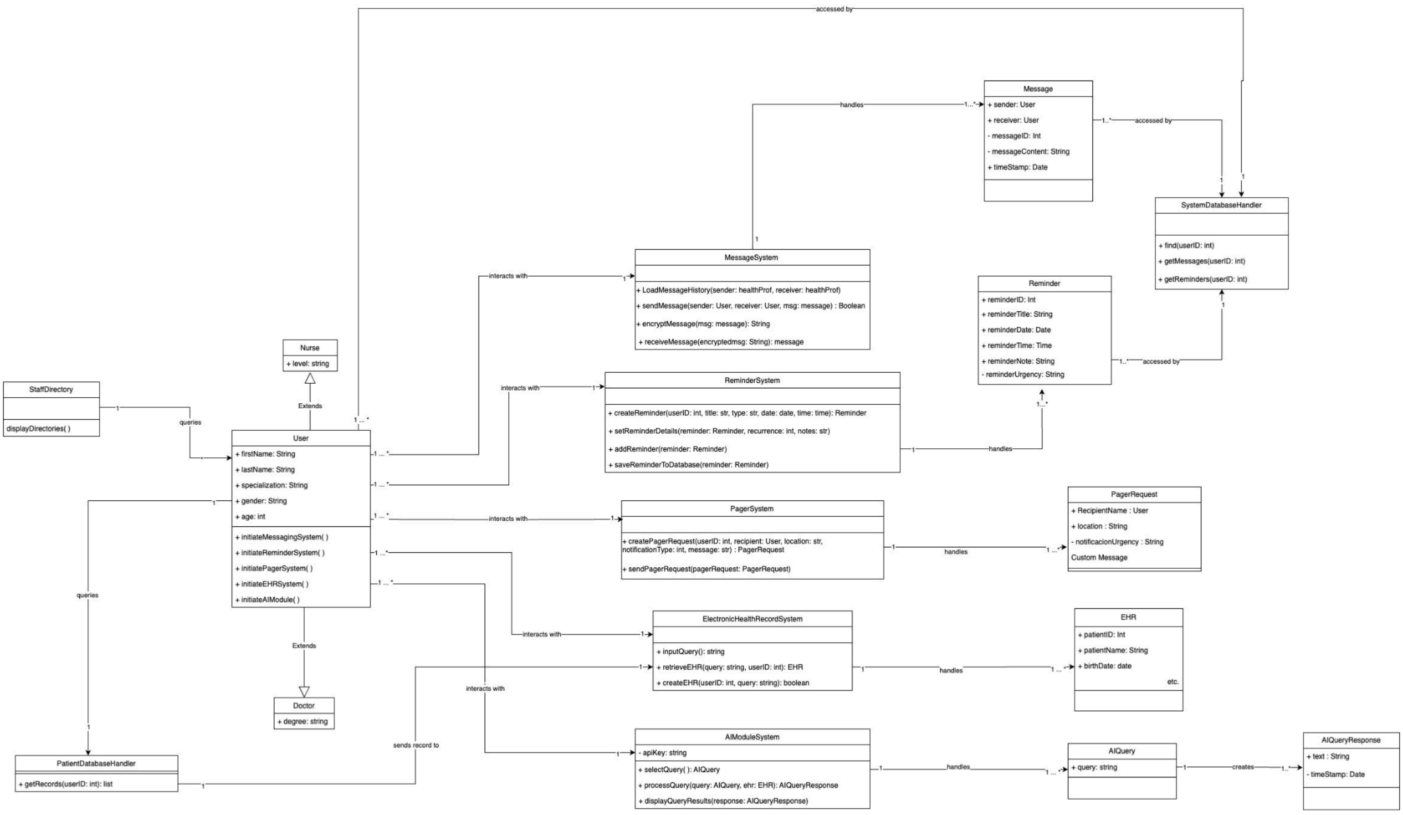
USE CASE: VIEW EHR


by Controller
retrieveEHR(userID, query)
:System
1.1: retrieveEHR(userID, query)
Database
1.2: ehr := create(userID, query)
by Creator
ehr:EHR

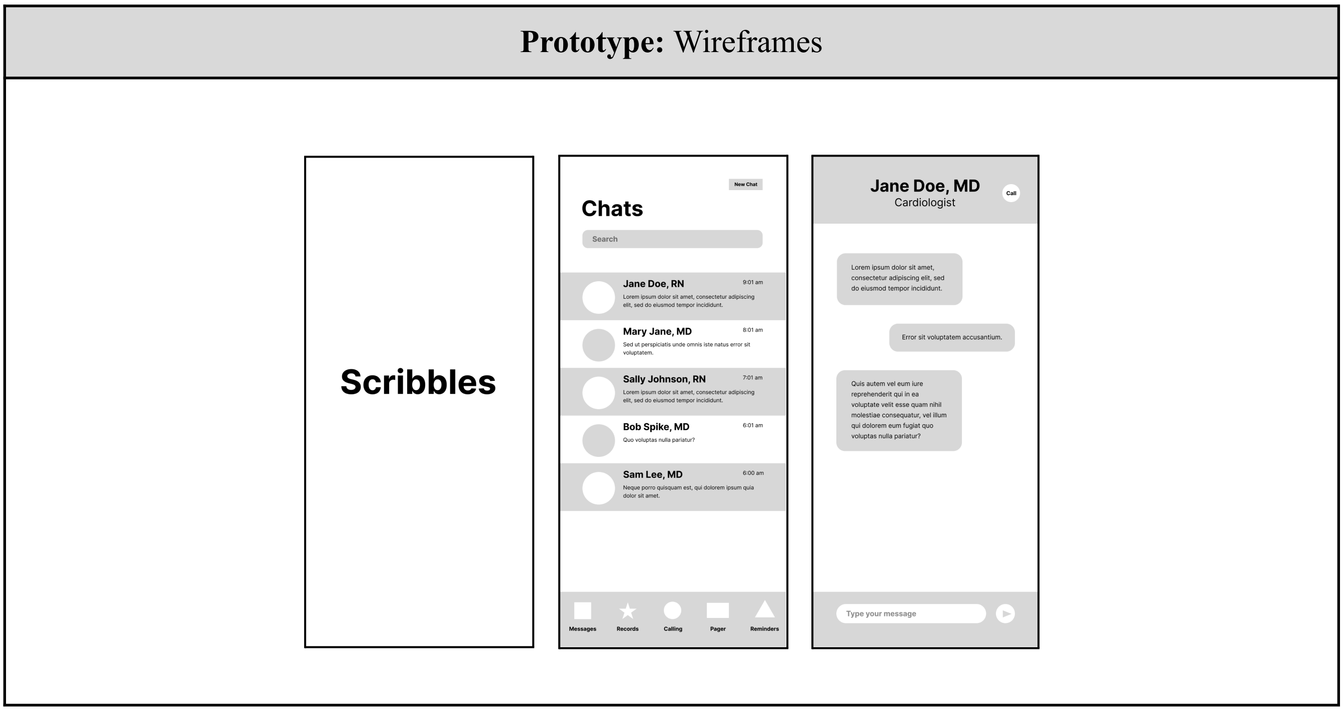
USE CASE: PERFORM AI-POWERED REQUEST


Represents the overall "system"
Controller
Controller
Creator
processQuery(EHR, query)
:System
1: processQuery(EHR, query)
:AIHandler
1.1: queryResponse := create( )
qr:QueryResponse

Class Diagram


accessed by
Message
+ sender: User
+ receiver: User
- messageID: Int
- messageContent: String
+ timeStamp: Date
handles
1...*
1..*
accessed by
1
1
SystemDatabaseHandler
+ find(userID: int)
+ getMessages(userID: int)
+ getReminders(userID: int)
MessageSystem
+ LoadMessageHistory(sender: healthProf, receiver: healthProf)
+ sendMessage(sender: User, receiver: User, msg: message) : Boolean
+ encryptMessage(msg: message): String
+ receiveMessage(encryptedmsg: String): message
Reminder
+ reminderID: Int
+ reminderTitle: String
+ reminderDate: Date
+ reminderTime: Time
+ reminderNote: String
- reminderUrgency: String
1..*
accessed by
1
Nurse
+ level: string
Extends
StaffDirectory
displayDirectories( )
1
queries
1 ... *
User
+ firstName: String
+ lastName: String
+ specialization: String
+ gender: String
+ age: int
+ initiateMessagingSystem( )
+ initiateReminderSystem( )
+ initiatePagerSystem( )
+ initiateEHRSystem( )
+ initiateAIModule( )
interacts with
interacts with
1 ... *
1
ReminderSystem
+ createReminder(userID: int, title: str, type: str, date: date, time: time): Reminder
+ setReminderDetails(reminder: Reminder, recurrence: int, notes: str)
+ addReminder(reminder: Reminder)
+ saveReminderToDatabase(reminder: Reminder)
handles
1...*
PagerSystem
+ createPagerRequest(userID: int, recipient: User, location: str, notificationType: int, message: str) : PagerRequest
+ sendPagerRequest(pagerRequest: PagerRequest)
handles
PagerRequest
+ RecipientName : User
+ location : String
- notificacionUrgency : String
Custom Message
queries
Extends
interacts with
ElectronicHealthRecordSystem
+ inputQuery(): string
+ retrieveEHR(query: string, userID: int): EHR
+ createEHR(userID: int, query: string): boolean
handles
EHR
+ patientID: Int
+ patientName: String
+ birthDate: date
etc.
Doctor
+ degree: string
interacts with
sends record to
PatientDatabaseHandler
+ getRecords(userID: int): list
AIModuleSystem
- apiKey: string
+ selectQuery( ): AIQuery
+ processQuery(query: AIQuery, ehr: EHR): AIQueryResponse
+ displayQueryResults(response: AIQueryResponse)
handles
AIQuery
+ query: string
creates
AIQueryResponse
+ text : String
- timeStamp: Date

Prototype Screens



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt.

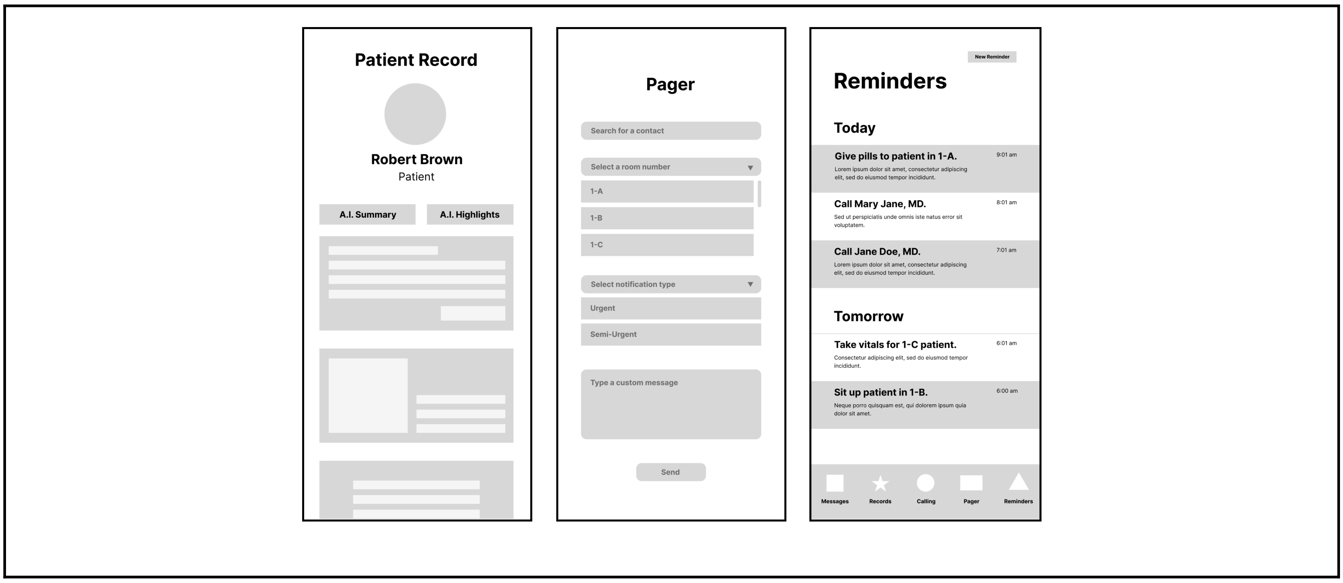
Quo voluptas nulla pariatur?

Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt.

Error sit voluptatem accusantium.

Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?



Patient Record

Robert Brown
Patient

Pager

Reminders

Today

Give pills to patient in 1-A. 9:01 am
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt.

Call Mary Jane, MD. 8:01 am
Sed ut perspiciatis unde omnis iste natus error sit voluptatem.

Call Jane Doe, MD. 7:01 am
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt.

Tomorrow

Take vitals for 1-C patient. 6:01 am
Consectetur adipiscing elit, sed do eiusmod tempor incididunt.

Sit up patient in 1-B. 6:00 am
Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet.