**Doctor’sNote**

**Design Document**

**Team 7**

- Ben Hardin

- Sanjana Koka

- Nathan Merz

- Jakob Molskness

- Arron Smith

- Ariana Zhu

**Table of Contents**

[**Purpose**](#_j8w5zws38qqo) **4**

[Functional Requirements](#_qwr0tbynpmjo) 4

[Non Functional Requirements](#_lst9c7dprlrr) 8

[**II. Design Outline**](#_o9kteohlns46) **10**

[Design Overview](#_x6naurds8dnu) 10

[iOS application](#_j2kdokjkmvtb) 10

[Web Interface](#_id02ptynyxaw) 10

[Server](#_fgc7dfgoko3q) 10

[Database](#_o43ny3i6j7b2) 10

[Data Structure Overview](#_6n5wsgstfamj) 11

[Class Diagram](#_4tznleyiuad6) 12

[Description of Classes and their Interactions](#_yi7xhjgg16zi) 13

[Server Details](#_3zxhgqwrqrrp) 15

[Database Details](#_ksxw597iqadh) 16

[High-Level Overview](#_uyp8xconfukq) 17

[Sequence of Events Overview](#_85gcck6lavqx) 17

User Login 17

User Sends Messages 18

Adding a Patient 19

[Creating Support Groups](#_o5xquo76je9x) 20

[Activity/State Diagram](#_8jnhc7pcw8e0) 21

[**III. Design Issues**](#_vfhgxjj0sfp4) **2**2

[Functional Issues](#_utxrbbvy17ux) 22

[Non-Functional Issues](#_nokc6ixryi1d) 24

[**IV. Design Details**](#_hdm0hudb4jtr) **27**

[UI Mockup](#_2qdcpwa8ofs9) 27

[API Routes](#_1ctf38my402s) 33

**Purpose**

Doctors frequently need a method to communicate with patients outside of the office to increase availability and promote patient follow-through. Patients often have questions for their doctor that can be quickly and conveniently answered through a messaging service rather than through a scheduled, in-person appointment. However, HIPAA adds restrictions to communication that prohibit using traditional messaging services such as email, SMS, or even WhatsApp.

The purpose of Doctor’sNote is to provide a secure, HIPAA-compliant way for doctors and patients to communicate in a timely manner without fearing for a patient’s privacy. Our mobile application allows secure messaging between patients and doctors, along with appointment scheduling to facilitate their interactions. Existing apps such as MyHealthOnline and OhMD also allow HIPAA-compliant messaging; however, they lack consolidated features such as medication reminders and calendar event exports. Additionally, our app provides access to support groups. By creating a community of support through our platform, we hope to promote patient mental health and wellness.

### **Functional Requirements**

*Authentication*

1. As a patient or administrator, I would like to be able to create an account, with a unique username and password, if I do not already have one so that I can authenticate and use the service.
2. As an administrator, I would like to be able to create an account on behalf of a doctor or registrar, with a unique username and password, if they do not already have one so that they can authenticate and use the service within my group.
3. As a user, I would like to log in if I have not done so yet on the current device so that I can continue to use my account.
4. As a user, I would like to only have to enter my password, rather than both my username and password, if I have already authenticated at least once on the current device so that reauthenticating is as convenient as possible.
5. As a user, I would like to have native support for iOS devices so I can use the app on my Apple devices.
6. As a registrar, I would like to have a web portal to access conversation records on behalf of doctors so that I can record transcripts in a HIPAA compliant manner.
7. As a user, I would like to be automatically logged out after closing the app or after a period of inactivity so that no one unauthorized access sensitive information.
8. As an administrator, I would like to be able to lock doctors and registrars out of their accounts so that I can prese rve information while revoking access.
9. As an administrator, I would like to be able to delete doctors’ and registrars’ accounts so that I can eliminate them from my group after proper information has been transcribed.
10. As a patient or administrator, I would like to be able to delete my own account so that my information is no longer stored.
11. As a user, I would like a way to log out of the app so that multiple people can log into one app or device if desired.
12. As a user, I would like a way to recover my password if I forget it.

*Starting a conversation*

1. As a converser, I would like to be able to search another converser by username so that I can request to start a conversation with them.
2. As an administrator, I would like to be able to search up a patient and request to start a conversation with them on behalf of a doctor so that I have improved personnel management capabilities.
3. As a converser, I would like to receive a nondescript notification when another converser has requested to start a conversation with me and have the option to either accept or decline the conversation, so that the conversation is fully consensual.
4. (If time allows) As a patient, I would like to be able to scan a QR code in my doctor’s office that will automatically initiate a conversation with them, so that I don’t have to search their username manually.
5. As a converser, I would like to be able to search among active conversations so that I can resume a conversation without having to scroll through all of them to find it.
6. As a doctor or administrator, I would like to be able to remotely log a patient out of their account upon a patient’s request, so that the app maintains HIPAA compliance.
7. (If time allows) As a patient, I would like to be able to match with a specialized doctor other than my primary physician based on my conditions and preferences.

*Doctor-Patient Messaging*

1. As a converser, I would like to be able to type messages into a text box and send them within the app so that I can communicate.
2. As a converser, I would like to be able to see past messages in the chat, as well as messages as they are sent, so that I can refer to past messages and see new ones.
3. As a converser, I would like to be able to see when a message has been read by the person I am conversing with so that I can anticipate incoming responses.
4. As a converser, I would like to know when a message fails to send so that I can attempt to resend it or otherwise handle the problem.
5. As a converser, I would like to be able to send images and videos to the person I am conversing with so that I can communicate more effectively.
6. As a converser, I would like to be able to have text to speech for my messages so that I can use the app even if I am visually impared.
7. As a converser, I would like to have voice dictation supported for messages so that I can use the app even if I am visually impared.
8. As a patient, I would like my private health information to be encrypted so that it cannot be accessed without proper authorization.
9. As a converser, I would like to receive discrete notifications about new messages so that I am aware when a new message has been received.
10. As a converser, I would like to be able to retract a sent message so that the person I am conversing with cannot see messages that were sent in error.
11. As a converser, I would like to be able to disconnect from a conversation so that I don’t have old conversations cluttering the app.
12. As a converser, I would like to be able to send additional file formats such as PDF and DICOM so that I can share relevant information in more formats (if time allows).
13. As a converser, I would like to be able to open the camera in-app and send photos immediately, so that the photo does not live outside of the app in an insecure environment.
14. As a developer, I would like a way to notify users if there has been a data breach so that they can be aware of when their information has been compromised.
15. As a developer, I would like messages to be automatically deleted after a certain time period so that I can maintain HIPAA compliance.

*Support Group Chat Room*

1. As a user, I would like to join different topic threads to meet with other people with similar conditions.
2. As a user, I would like to create a new thread in the chat room if my topic of interest is not already available.
3. As a user, I should see a confirmation upon joining a support group stating that support groups are more public and the user should not share any personal or confidential information.
4. As a user, I would like to set my own display name in the support group.
5. As a user, I should be able to report issues in the support group to a support group administrator so that bullies and toxic behavior is not tolerated within support groups.
6. As a user, I should be the support group moderator if I make a request to the hospital administrator to create the support group.
7. As an administrator, I would like to approve new support group requests to be hosted by the hospital I represent.
8. As an administrator, I would like the ability to terminate any inappropriate support groups.

*Metrics*

1. As a developer, I would like to maintain a count of sent messages, as well as a count of messages that initially failed to send but were eventually sent, so that I can be aware of server side problems before users self report it.
2. As a user, I would like a way to send feedback to the developers to report issues so that they can be fixed.
3. As a developer, I would like a way to view submitted feedback so that I can fix the issues accordingly.
4. As a developer, I would like a way to purge inactive doctor accounts so that they do not clutter the service.

*Scheduling*

1. As a patient, I would like to request in-app or in-person appointments appointments with the doctor so that I can meet with them if necessary.
2. As a patient, I would like to export appointments to my calendar, so that I can easily keep track of my appointments.
3. As a patient, I would like to be able to conveniently find and connect with an anonymous support group, so I can get help dealing with my condition.
4. As a doctor, I would like the ability to schedule an appointment with the patient with whom I am chatting so that an in-person appointment can be made if necessary.
5. As an administrator, I would like a way to initiate and terminate doctor-patient relationships so that the patients and doctors are not required to set them up.
6. As an administrator, I would like an interface to view summary data so that I can gauge the effectiveness of the service.
7. As an administrator, I would like to add and remove doctors to and from the network so that they can be paired with patients when available.
8. As a registrar, I would like to be able to export transcripts and other records so that my employer can maintain HIPAA compliance.
9. As a compliance coordinator, I would like a way to view information access history so that I can enforce compliance controls.
10. As a patient, I would like to add, remove, and edit reminders to take medications.
11. As a doctor, I would like to create and remove reminders for patients to take medications.

*Push Notifications*

1. As a user, I would like to receive non-descriptive notifications of any new messages to ensure confidentiality.
2. As a user, I would like to receive notifications about appointment requests and confirmation.
3. As a user, I would like to receive notifications for any changes or updates to my appointments.
4. As a user, I would like to be notified if a message I sent was not successfully delivered.
5. As a patient, I would like to be notified when it is time to take my medication.
6. As a patient, I would like to be notified if there is any update to the frequency/time I need to take my medication.
7. As a user, I would like to turn on and off my notifications.
8. As a user, I would like to receive notifications about any responses from the support group chat room.
9. As a user, I would like to toggle my notification preferences from topic and frequency for the chat room.

*Other*

1. As a user, I would like UI elements to have accessibility tags so that services such as VoiceOver can help me navigate the app if I am visually impaired.
2. As a patient, I would like the ability to import data from Apple Health to share with doctors (if time permits).
3. As a patient, I would like a link in the app to take my to my healthcare provider’s website for more information.
4. As a patient, I would like the ability to see a profile of my doctor including their name, photo, and normal work hours.
5. As a doctor, I would like the ability to set up a profile with my name, photo, and normal work hours.

### **Non Functional Requirements**

*Performance*

The service should have a median uptime of 99.9%, and the user should be able to send messages at any time. Doctors will only be required to respond within normal business hours. This should have a latency within 500ms in order to ensure that the user experiences responsiveness inside of the app. Our service should be scalable up to 10,000 simultaneous requests since that is the number of requests we estimate are needed to serve the state of Indiana assuming each patient talks to their doctor for 30 minutes a month. It is important that we communicate with the database in a way that minimizes calls to reduce data for the user and requests that can quickly become enormous as our service reaches new customers.

*Security*

The app should be locked with TouchID, FaceID, or a PIN so that if someone has access to a user’s device they cannot easily log into the app. The service will also require Two-Factor Authentication on the first login to each device to provide greater verification of the patient. Finally, end to end encryption will be used to secure communications between doctors and patients so that unauthorized third parties cannot obtain private health information, even if they have access to the database.

*Usability*

The app should be clear and easy to use for patients with diverse backgrounds and ages. The interface should be large enough to be readable for the elderly and should be easy to navigate for those with little experience with technology.

*Hosting/Development*

The backend will be hosted by Amazon Web Services (AWS) and the frontend will be hosted on the Apple App Store. Each can receive updates at different times in case of a bug that occurs in only one of the systems.

*Maintainability*

Our project should follow common software engineering patterns to maintain a readable and well-formatted codebase. Before deploying any code, we will conduct peer reviews and confirm that tests are in place to verify the robustness of the app and service.

*Reliability*

The code base should have at least 80% line coverage under unit tests to assert reasonable confidence that our system works as expected. The service will be tested with data size up to the specification.

*Extensibility*

The classes shall have an easily extensible structure where generic classes define interactions and are extended to add specific features.

## **II. Design Outline**

### Design Overview

The system will have four primary components: an iOS application for patients and doctors, a web interface for administrators and developers, a server to provide central logic and access to database storage, and a database to store messages and user data. A client-server model will be used with both the application and web interface serving as clients. The specific breakdown between responsibilities will be as follows:

#### iOS application

* Perform encryption of sensitive data
* Manage doctor and patient interfacing
* Provide native features (e.x. Text-to-speech or Calendar support)

#### Web Interface

* Provide an auditing and configuration interface
* Perform client-side decryption as needed

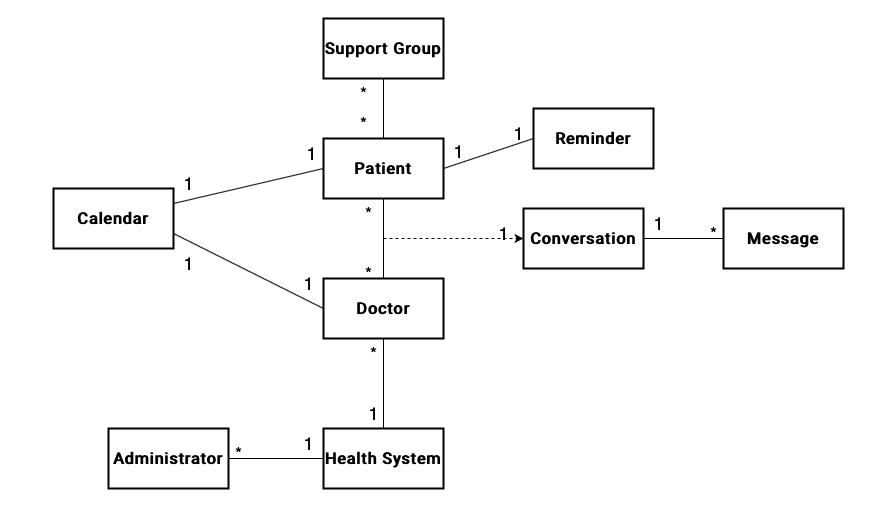
#### Server

* Authenticate client
* Serve data from database
* Format incoming data for database
* Hash and salt password

#### Database

* Store user data and configurations

### Data Structure Overview



#### 

#### Class Diagram

#### Description of Classes and their Interactions

###### User:

* Basic user of the program
* Contains an id, a name, a conversation list, and a calendar
* Allows login
* Getters for attributes
* Can create a Conversation and a SupportGroup

###### Admin:

* Extends User
* Linked to a hospital system
* Can add doctors and patients
* Can also delete and lock accounts

###### Doctor:

* Extends User
* Contains list of patients

###### Patient:

* Extends User
* Contains list of doctors
* Contains personal information

###### Personal Information:

* Belongs to a patient
* Lists information of the patient useful to the doctor
* Contains address

###### Address:

* Lists address of patient
* Contained in personal information
* Getters and setters for attributes

###### Conversation:

* Made up of Messages
* Contains the recipient and a list of X previous messages
* Can post and delete Messages

###### Support Group:

* Similar to a conversation, but there are multiple users
* Made up of Messages
* Post and delete Messages

###### Message:

* The basic components of the Conversation and the Support Group
* Contains a text field, metadata, and an id number

###### Calendar:

* Each user has a calendar
* Made up of a list of Events
* Can add and remove events

###### Event:

* Basic building blocks of Calendar
* Contain an id, title, description, and date
* Getters and setters for attributes

###### Hospital System:

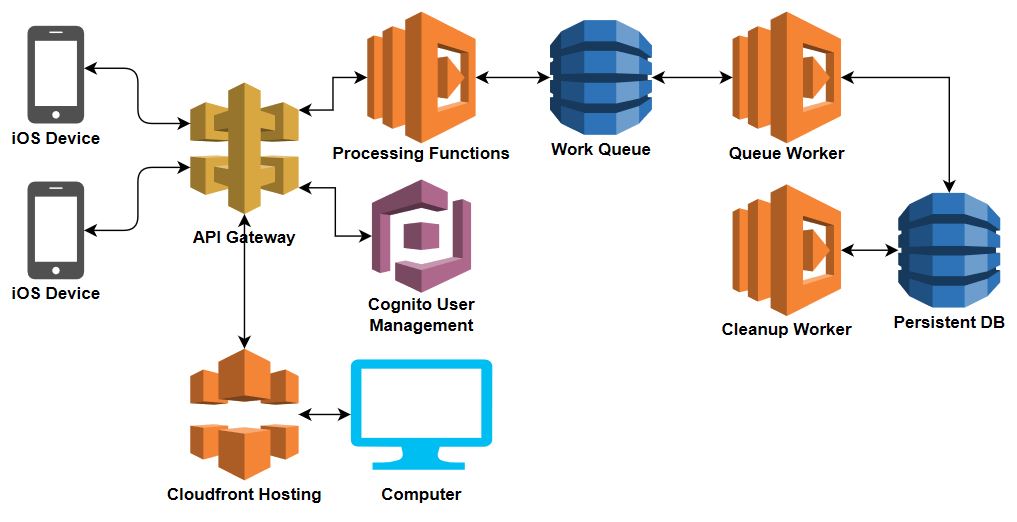
* Contains systemID and name
* Contains list of Doctors and Patients linked to system
* Getters for attributes

###### 

#### 

#### Server Details

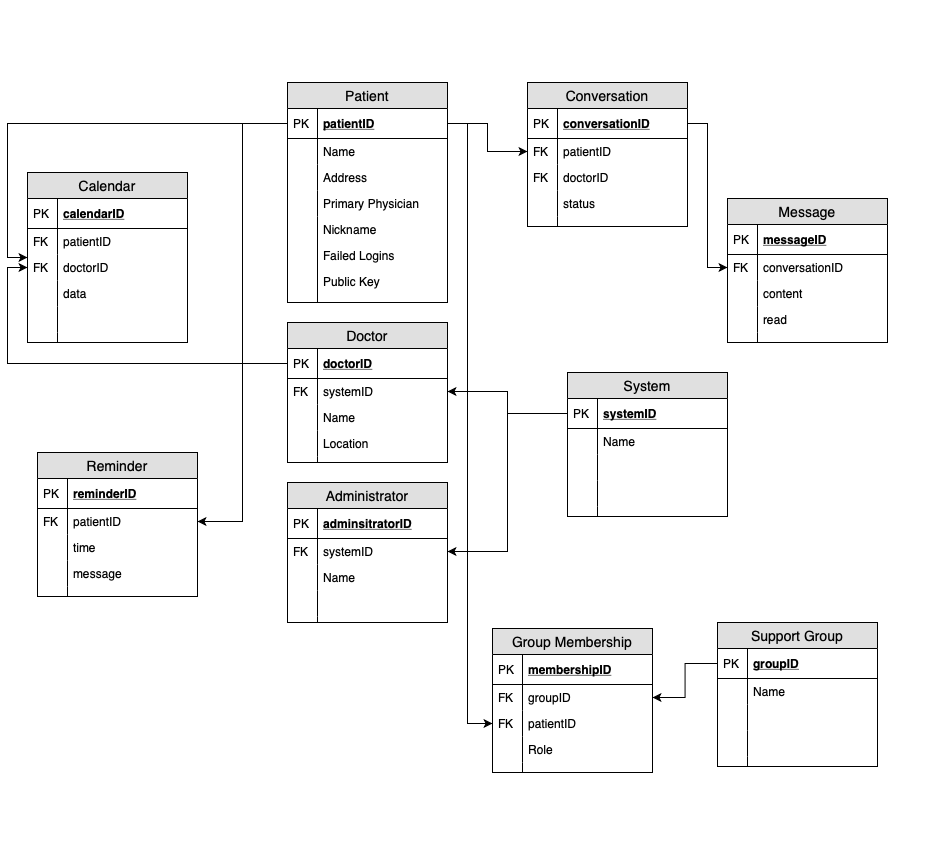
The backend will be run on a series of distributed AWS services. Both front end systems (iOS and in-browser) will make calls to a series of restful APIs hosted on AWS API Gateway over HTTPS, with data passed in a JSON format. Upon receiving the request, API Gateway will verify the requester is authenticated using AWS Cognito, which provides secure user and role management. Once verified, API Gateway will then call the corresponding Lambda function, again passing the data received from the original API call in a JSON format. The Lambda will process the request into a serviceable database work item, which it will then write into a work item queue in MariaDB in order to prevent concurrency issues with the persistent database. Simultaneously, a queue worker (also implemented by a Lambda function) will monitor the work queue and process work items one at a time, making the necessary reads or writes to the persistent MariaDB database. Additionally, a final Lambda running a cron job will periodically perform cleanup operations on the database, such as removing messages over a certain age. Prior to the API gateway, the web portal for registrars and administrators will be hosted on cloudfront with a relatively thin front end client. Both this portal and the native iOS app will provide user interfaces to make authenticated calls to the API Gateway. Effectively distributing the workload across a number of AWS products in this manner will reduce cost and make testing easier, improving scalability, reliability, and maintainability.



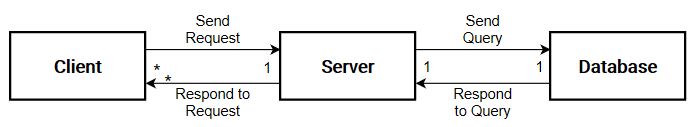
#### 

#### Database Details

The database will be an instance of MariaDB hosted on Amazon Web Service’s RDS service. The database will be configured to back up periodically as part of the RDS service. Access to the database will be managed by the server. The anticipated layout of the database is below:

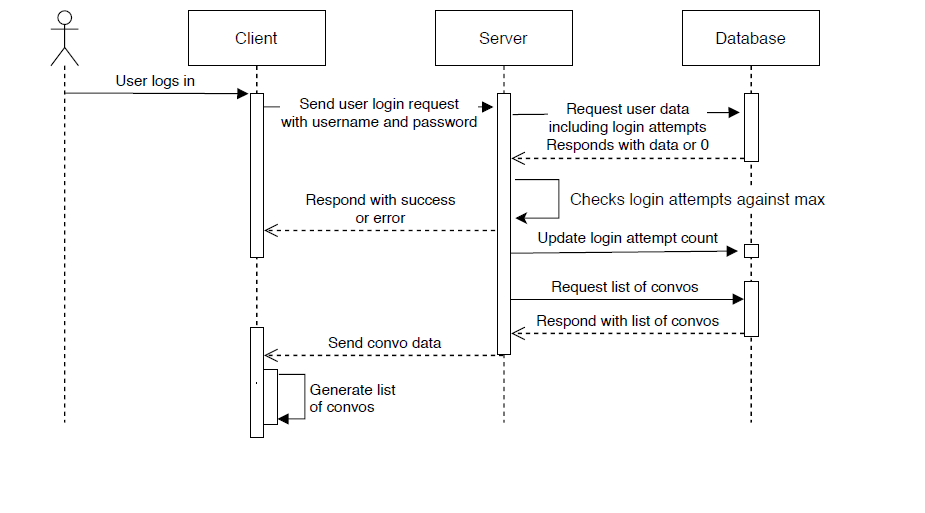


### High-Level Overview

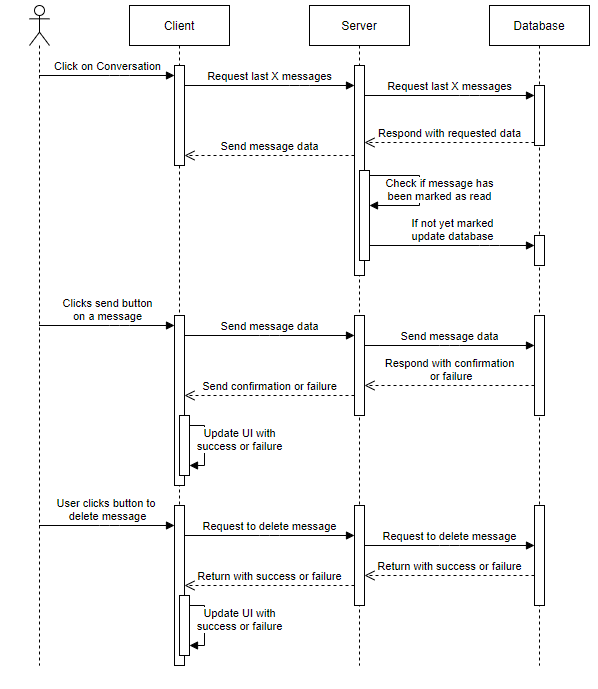


### Sequence of Events Overview

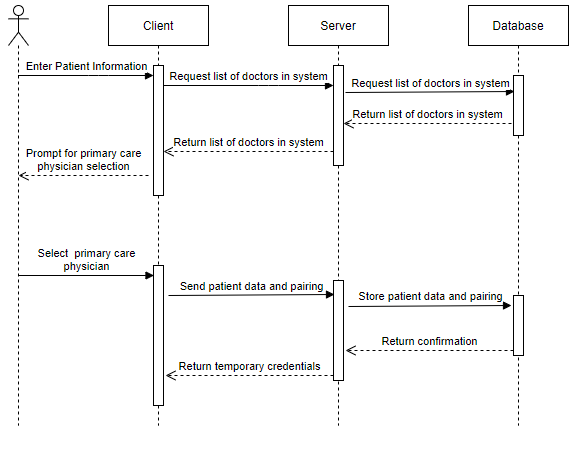
User login



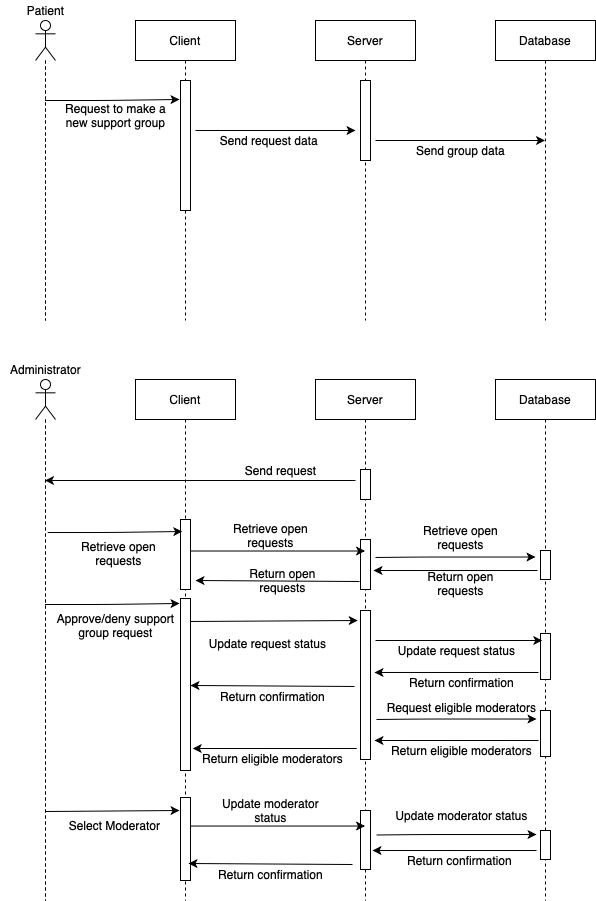
User sends messages



Adding a patient

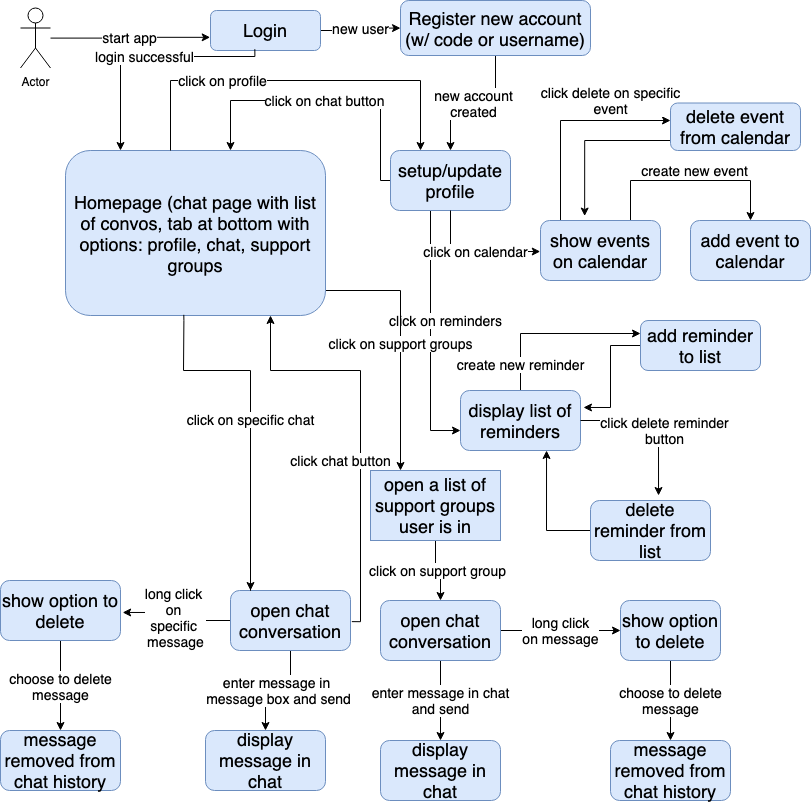


#### Creating Support Groups



### Activity/State Diagram

The user logs into the homepage with their account. If they don’t have an account they can register a new one using a code given to them by a doctor/admin or create a username. A list of their chat conversations opens upon loading. There is a bar at the bottom of each page that contains the options: profile, chat, and support groups.

****

## III. Design Issues

### Functional Issues

1. What information will we need from a patient in order to connect them to our service?

Option 1: Username, password, contact information, primary physician

Option 2: Username, password, personal and contact information, primary physician

Option 3: Username, password, personal and contact information, location, primary

physician

Choice: Option 2

Justification: Although some information is necessary to validate a user’s identity upon login, given the sensitive nature of our service we wanted to collect as little additional information as possible. This both makes the signup process less burdensome for the user and reduces the impact of a potential data breach. Necessary information includes a user’s phone number and email address in order to verify their identity and enable two factor authentication, but we do not need to know their location as this is not relevant to the administrator since their home address will provide enough information.

1. Should a user’s message history appear across multiple devices?

Option 1: Only the device that sent the message can see the message.

Option 2: Any device in which the user is logged in can see all user messages.

Choice: Option 2

Justification: The user should be able to log into another device and see the same messages as the device on which they sent the messages in order to provide a more consistent experience. Additionally, it is not difficult to imagine situations where viewing recent message history would be helpful. A user could, for example, have finished messaging their doctor about a prescription, believed they have no further use for the app, and deleted it from their device. Then, one week later, they remember they are supposed to taper off the medication but may not remember how quickly or over what time span. Being able to redownload the app, authenticate, and view old messages would provide the information the patient needs, rather than necessitating they message their doctor again to confirm the information. Additionally, doctors will likely share iPads or other devices across the hospital system and being able to log into any device would improve their ability to function within such an environment.

1. How should a user gain access to the service?

Option 1: The patient creates an account, searches for their health provider, and the health provider approves or rejects the patient as part of their system.

Option 2: The health provider adds a client and the client is sent a request or code to join the service.

Option 3: Allows for either Option 1 or Option 2 when creating a patient account.

Choice: Option 3.

Justification: We do not have a strong preference for which method of account creation is used, as both are considered equally secure and manageable. However, we also recognize that end users, specifically hospital systems, may. As such, we chose to implement both strategies to give end users the greatest flexibility possible to integrate our service within their existing workflow to make the transition as seamless as possible.

1. How should we allow patients to communicate in anonymous support groups?

Option 1: Chat rooms (channels) organized by topic

Option 2: Q&A style discussion board

Choice: Option 1.

Justification: A chat room style support group will allow for more natural communication. Users will likely develop more meaningful connections through a chat room style and will receive a greater sense of encouragement from the shared experiences and thoughts of others. Additionally, a chat room strategy allows us to build on the one-to-one conversations we will already be implementing, rather than starting from scratch.

1. How will the support groups be moderated?

Option 1: Report abuse button

Option 2: Vote Kick

Option 3: Moderator

Choice: Moderator

Justification: Having one trustworthy person to moderate the conversation allows for greater assurance that the conversations will be uplifting. Report abuse and vote kick buttons do not allow for guiding the conversation while a moderator can actively steer the conversations in a positive manner. If there is no moderator, an individual may also not be likely to report another individual for fear of being viewed as unsupportive of the other members of the group.

1. Who should moderate the support groups?

Option 1: Administrators

Option 2: Group creators

Choice: Option 2.

Justification: There will likely be too many support groups for administrators to maintain on top of their normal job responsibilities. Additionally, their role is not intended to have direct interactions with patients as a doctor would. Having a moderator who is part of the support group also allows for them to provide more helpful guidance in the discussions. If the moderator wishes to leave the chat, they can pass the responsibility along to another group member or close the group if they wish.

1. How should appointments be scheduled?

Option 1: Doctors and patient send individual appointment requests to be accepted by

the other party

Option 2: Doctors’ availability is listed for the patient to request to be confirmed by the

Doctor.

Option 3: Combination of Options 1 and 2.

Choice: Choice 3.

Justification: If the doctor has been chatting with the patient, it may be helpful for them to immediately send an appointment that the patient can accept, reject, or modify. However, if the patient has a random concern and would like a in-person appointment without chatting first, it may be ideal for them to immediately send an appointment request to the doctor.

### Non-Functional Issues

1. How should we encrypt information?

Option 1: Plain text messaging

Option 2: Relying on HTTPS encryption

Option 3: Point-to-point encryption

Option 4: End-to-end encryption

Choice: Option 4

Justification: Although they are much simpler from a technical perspective, options one and two would not be HIPAA compliant and may place users’ private health information at risk. Achieving HIPAA compliance is one of the primary goals of this service, so using plain text or solely standard HTTPS is a nonstarter. Additionally, our strategy for encrypting messages will likely be a central part of our sending and receiving infrastructure, so we want it to be as robust as possible. Point-to-point encryption systems are typically less flexible and more prone to obsolescence over time than end-to-end encryption systems, so by going with the latter we are able to improve our code’s maintainability and robustness.

1. Cloud Hosting Provider

Option 1: AWS

Option 2: Google Cloud

Option 3: Azure

Option 4: IBM Cloud

Choice: AWS

Justification: Because AWS provides a limited free tier to students, choosing them as our cloud hosting provider will allow us to use industry grade tools at the lowest cost possible. In addition, as the current leader in cloud market share, AWS has extensive documentation created by the company and third parties on how to use its services. Furthermore, AWS can be used to create HIPAA compliant applications and has the scale and infrastructure to accommodate future scaling needs.

1. What platform will we use for our service?

Option 1: iOS

Option 2: Android

Option 3: Web-based Client

Choice: Option 1

Justification: iOS provides better security features that should prove useful in transmitting and displaying sensitive private information. iOS generally provides a more simple user experience and the vast majority of tablets in hospital systems run iOS. Additionally, our team members have more experience with iOS programming and will find it a simpler learning curve for the scope of this project. Apple provides excellent documentation for iOS and many built in security frameworks that we will find helpful in creating our service.

1. What iOS language will we use for the app?

Option 1: Swift 5

Option 2: Objective-C

Option 3: React

Option 4: Flutter

Choice: Swift.

Justification: Swift has developed greatly and become Apple’s recommended language. There is greater documentation and support for Swift and should be easier for team members to learn Swift quickly. Additionally, Swift is well-optimized for the iOS platform and has the best performance.

1. What database backend will be used?

Option 1: MariaDB

Option 2: MongoDB

Option 3: PostgreSQL

Option 4: DynamoDB

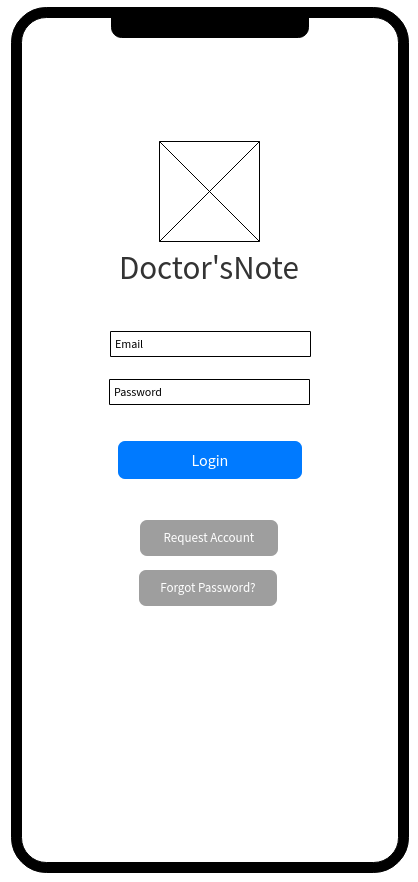
Choice: MariaDB

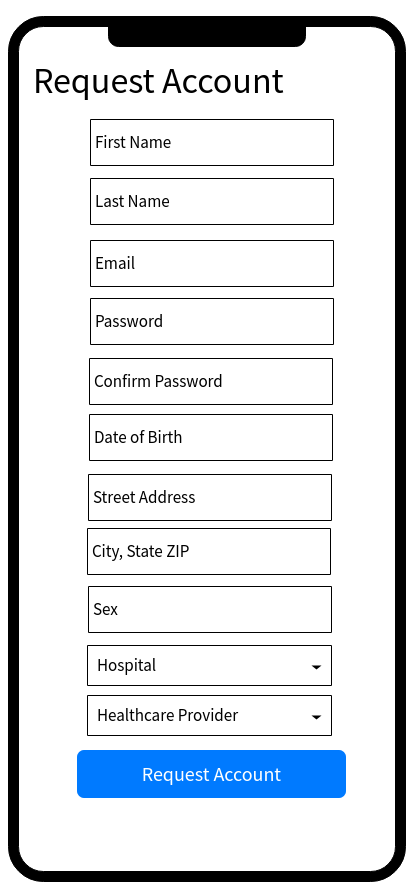
Justification: Since we have strong relationships in the data, we would like a strongly relational database. Neither DynamoDB or MongoDB are relational, while MariaDB and PostgreSQL are. Between the two of those, MariaDB is significantly less expensive to use. In addition, MariaDB works with Amazon’s RDS service, allowing it to be more easily incorporated into the rest of our proposed architecture. Finally, MariaDB has a very large body of documentation, mitigating the potential risks of not going with the more frequently used services.

## IV. Design Details

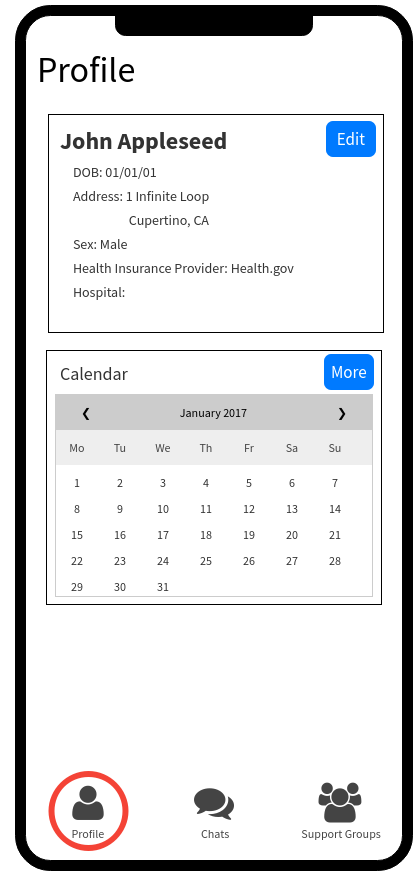
### UI Mockup

When the user first opens the app, they will see the login page. If the hospital administrator has not already set up an account for them, the user will press the Request Account button and be taken to the next page.

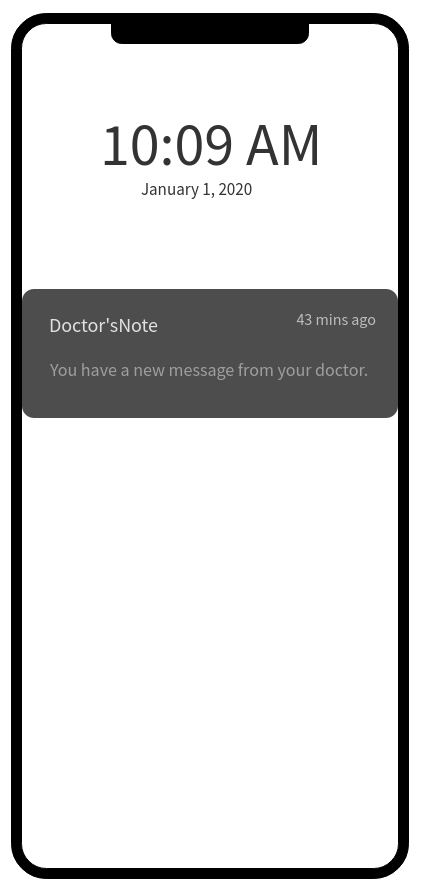
****

****

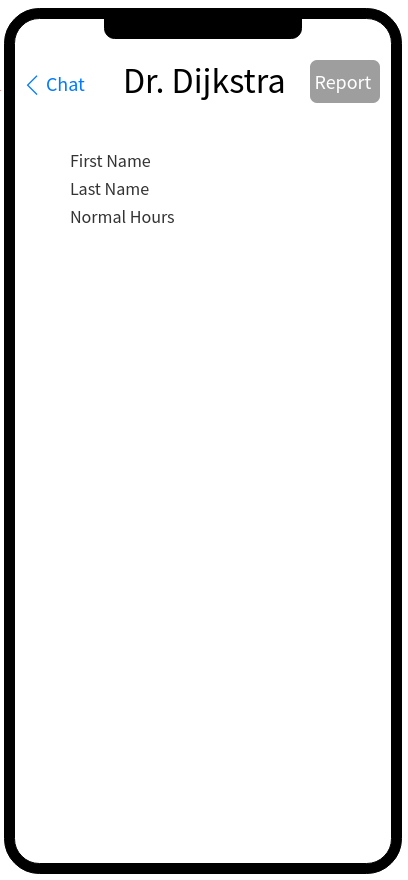
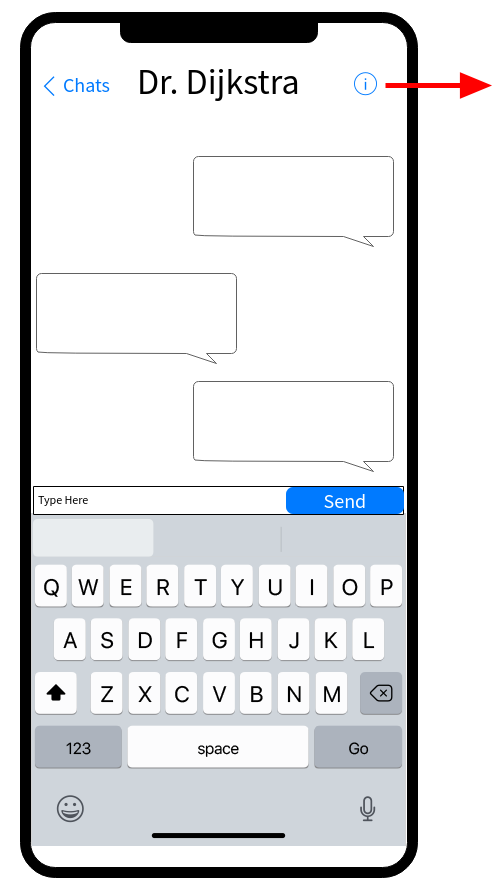
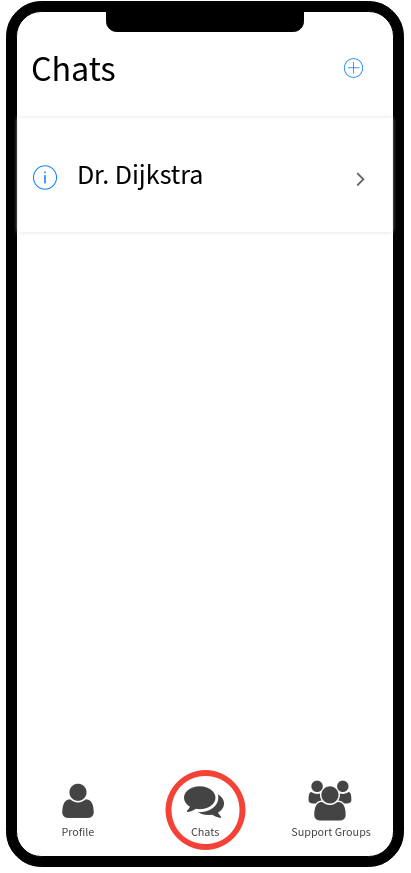
On the Request Account page, the user will be able to type in the necessary information to create an account. Each field will verify that the input is of the correct format for that data type. Once the information is entered correctly, the user will select the Request Account button. Their information will be sent to their hospital administrator for approval.

****

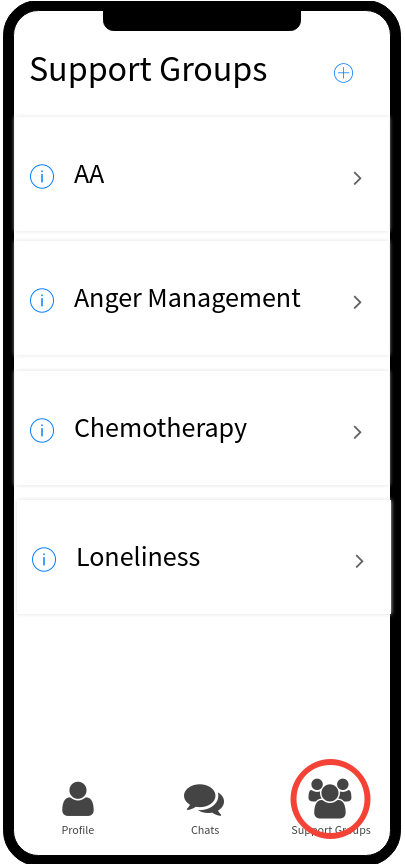
Once the user’s account has been created, they will be able to log in with the credentials they requested. The app will open and they can select the left-most tab bar icon to go to their Profile page. On this page, they will be able to see and modify basic information about themselves in case they change addresses or health insurance providers. The profile page also contains a small calendar that will show appointments. By pressing “More”, the user will be able to see a greater detailed view of the calendar to see specifics of appointments. The detailed view will allow the user to export the calendar events to iCal or GCal. There will be a third view for medication reminders below the calendar view.

****

The app will support push and local notifications. The push notifications will give discrete notifications that a doctor has sent a message to the patient or vice versa. The local notifications will be used to remind the user of scheduled events such as appointments or medication reminders.



The second item on the tab bar controller is the Chats screen where all conversations between doctors and patients will appear. In this section the conversation can be selected and it will open in a new view showing all of the messages and allowing for messages to be sent. The user can select the information icon next to the name of the person with whom they are messaging. Selecting this opens a new view with information about the person.



The third icon on the tab bar allows opens a page dedicated to Support Groups. This option is only available to patients. Here, the patient will see all of the support groups they have joined. To search and find other groups, they can select the plus icon at the top of the page to search and join other available groups. Clicking on the icon of a group that they have joined will open a similar conversation page to the one between patient and doctor.

### API Routes

|  |  |  |
| --- | --- | --- |
| Route | HTTPS methods | Logical action(s) |
| /user | GET | Retrieve own id |
| /user/{id}/conversations | GET | Retrieve conversations |
| /user/{id}/calendar | GET, POST | Retrieve/modify calendar |
| /user/{id}/reminders | GET, POST | Retrieve/modify reminders |
| /conversation | PUT | Create a new conversation |
| /conversation/{id} | GET, POST | Retrieve conversation messages, send a new message |
| /group/{id}/messages | GET, POST | Retrieve messages, post messages |
| /group/{id}/membership | GET, PUT | Retrieve members, change membership |
| /group/all | GET | Retrieve list of groups |
| /group/new | PUT | Make a new group |