Project Proposal

Title: CureSphere : A One Stop Shop for Managing Your Healthcare

Team members:
Andreas Loutzidis – al4419 Weiyao Li – wl2872
Shwetha Subramanian – ss6618 Lynn Kim – ck3127

Problem Statement:

Our goal is to simplify the process of finding and receiving healthcare services for patients by providing a user-friendly platform. Our platform allows patients to search for doctors, book appointments, receive treatment, and manage their healthcare needs, all in one place. Furthermore, our app streamlines the process of receiving verbal instructions from doctors by enabling them to write short comments about medication and other treatments within our app. Additionally, we aim to address the issue of cost by including a medicine price comparison functionality. Overall, our full-stack, cloud-based platform is designed to optimize the healthcare process and provide patients with a more convenient and efficient way to manage their healthcare needs.

Motivation:

The healthcare system can be complex and difficult to navigate, which can make it challenging for patients to receive the care they need. By providing a user-friendly platform, this app aims to streamline the healthcare process and provide a more convenient and efficient way for patients to manage their healthcare needs. In addition to providing a way for patients to manage all their healthcare records in one place, it also provides a medicine price comparison functionality, which aims to address the issue of cost and help patients save money on their healthcare expenses. Overall, the goal of this app is to provide patients with a more positive and satisfactory healthcare experience.

Literature Review:

There are several healthcare-related apps available that aim to simplify the process of receiving medical care, but each has its unique value proposition. For example, apps such as Zocdoc, Doctor On Demand, and Teladoc focus on providing a platform for patients to find doctors, book appointments, and receive virtual medical consultations.

The unique value proposition of our app is that it aims to provide patients with a comprehensive solution that allows them to complete their entire treatment cycle assisted only by the app. Unlike other apps, this app not only helps patients find and book appointments with doctors but also assists them throughout their treatment process, enabling doctors to provide short comments on medication and other treatments, add appointments to the calendar, maintain a history of all their records and provide medicine price comparison functionality. This full-cycle care approach, which streamlines the healthcare process and provides a more convenient and efficient way for patients to manage their healthcare needs, sets this app apart from other healthcare-related apps available on the market.

Target Audience:

The intended target audience for this project are the following:

- People who do not have access to in-person medical services.
- Patients with different preferences over the doctors based on distance, average ratings, and previous comments.
- People who want a full cycle care not only “search-doc” functionality.
- People who want to track the progress of their medical conditions over time, and not just one-time appointment service.
- People who want a medicine price comparison with the consideration of the hidden prices (location, delivery time, delivery fee, etc).

Success Criteria:

The project is successful if the target audience is able to:

1. Sign up via email - set a username and password
2. Log in via the username and password set
3. Fill up a question form to record symptoms and other inputs
4. Record the data in his patient record associated with the account
5. Get a recommendation for a doctor as per inputs given in the form
6. Book a consultation with a doctor based on the slots provided
7. Get the option to book a calendar invite and reminders for the consultation appointment booked
8. Chat with the doctor or have a video call with the doctor in the slot
9. Record the prescription in his patient record for instant access at any time
10. Encrypt the patient records since they are confidential
11. Get recommendations for ordering medicines and alternatives if the medicine is not available
12. Able to do a price comparison of the medicine and be redirected to the seller to order it

Data Sources:

Data Usage:

The data that our healthcare service platform would deal with includes:

1. Patient information: This includes personal information such as name, age, gender, contact details, and medical history. This data would be collected directly from the patients during the registration process or imported from other healthcare providers with the patient's consent. While storing this information in patient records, it would be encrypted to maintain confidentiality.
2. Doctor information: This includes the name, specialization, contact details, and availability of doctors. This data would be sourced from various medical directories and databases, as well as collected directly from doctors who register on our platform.
3. Appointment schedules: This includes the date and time of scheduled appointments, doctor availability, and patient preferences. This data would be generated when patients book appointments on our platform.
4. Treatment information: This includes the medication prescribed, treatment plans, and the progress of treatment. This data would be entered by doctors in the form of short comments within our app, which would then be accessible to patients.

5. Cost information: This includes the price of medications and treatments, as well as insurance coverage and discounts. This data would be sourced from various healthcare providers, insurance companies, and pharmacies.

The sources of this data would include various healthcare providers, databases, and directories, as well as the patients and doctors themselves. We would ensure that all data is collected and stored in compliance with relevant data protection and privacy laws.

APIs:
Doctor Directory APIs: These APIs provide access to directories of healthcare providers, including doctors, dentists, and other healthcare professionals. Examples of such APIs include the BetterDoctor API and the Healthgrades API.

Appointment Scheduling APIs: These APIs provide the ability to schedule appointments and manage calendars. Examples of such APIs include the Doctor On Demand API and the Zocdoc API.

Treatment and Medication APIs: These APIs provide access to information on medications, dosages, and treatment plans. Examples of such APIs include the Epocrates API and the MedlinePlus API.

Cost Comparison APIs: These APIs provide access to pricing information for medications and other medical treatments. Examples of such APIs include the GoodRx API and the RxNav API.

YAML file:

Our YAML file can be found here.

In the following image is a list of the implemented APIs.

Architecture Diagram:
Due to size limitations of the architecture, our implemented architecture can be found on the following link in our GitHub repository. You might have to scroll right, depending on the size of your monitor.

Access the image architecture here

Link of the clickable Prototype:
Below there are two links for our prototype, one for the patient view and one for the doctors view. You will have to create a dummy account with an email and password in the authentication to move forward.

Patient : https://ck3127.wixsite.com/curesphere
Doctor : https://ck3127.wixsite.com/website

Sample images from the prototype:

CURESPHERE

Home About Us My Account

A

aloutzidis

My Medications My Appointments My Account

My Medications

Methylphenidate	Oseltamivir Phosphate		
		-	-
more info price comparison	more info price comparison		

Home About Us My Account

A

aloutzidis

My Medications My Appointments My Account

My Appointments

View and manage the appointments you've made.

Upcoming Appointments

10am, March 10, 2023

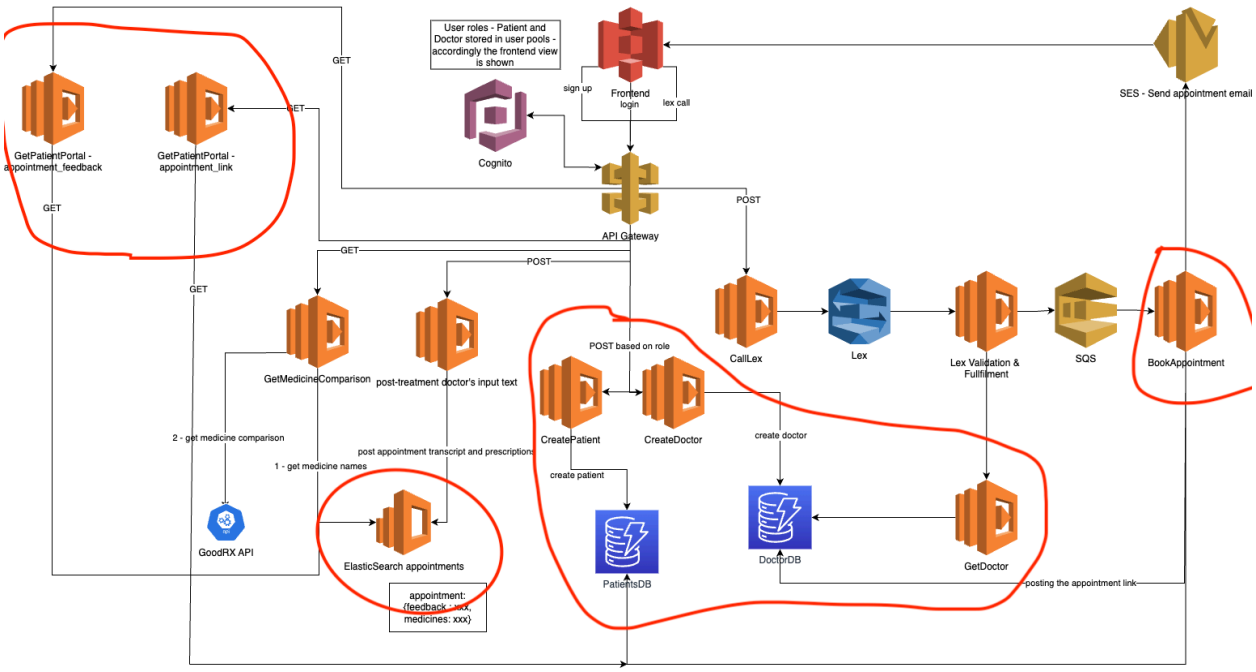
Calendly link : www.calendly.com/BookAppointment

Checkpoint 2

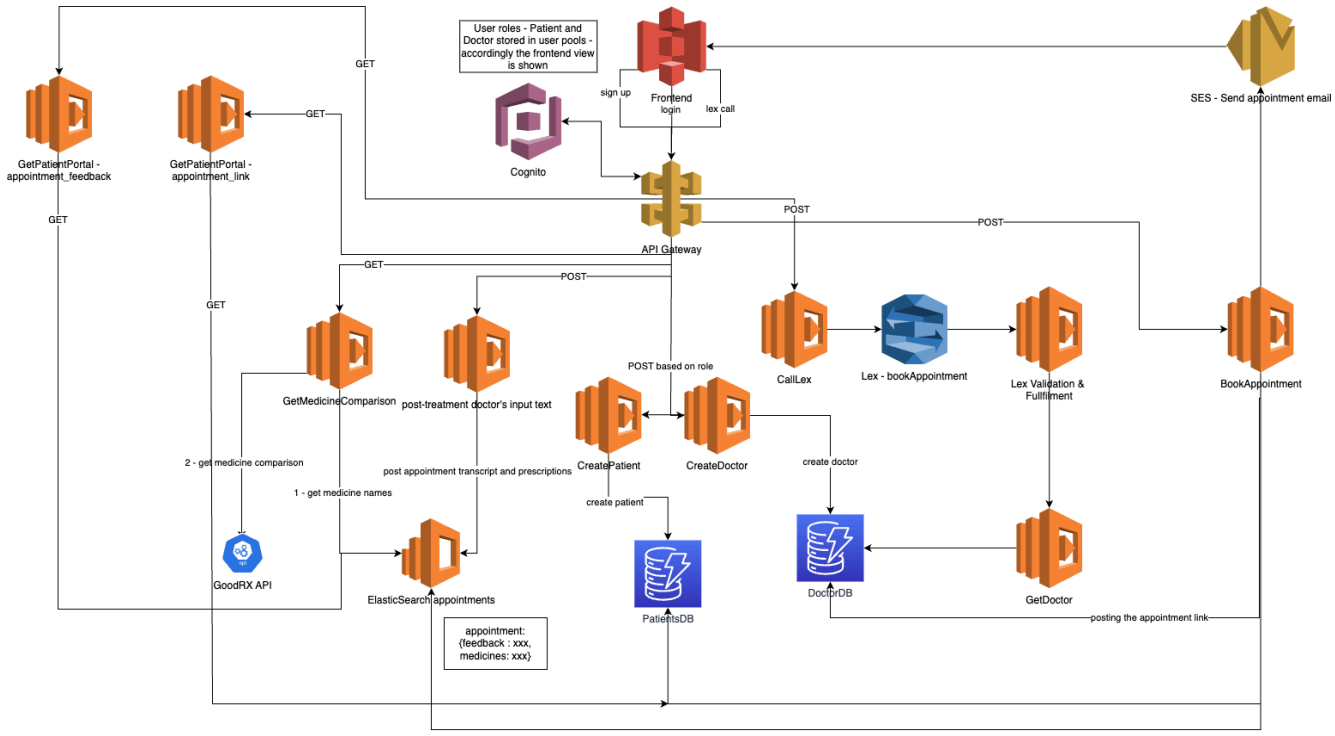
We plan to implement:

- Lambdas
  - GetPatientPortal - appointment_feedback
  - GetPatientPortal-appointment_link
  - CreatePatient
  - CreateDoctor
  - GetDoctor
  - BookAppointment
- DynamoDB
  - PatientsDB
  - DoctorsDB
- OpenSearch
  - ElasticSearch-appointments

We are also including an attached image that highlights the parts of our architecture that we will be implementing.



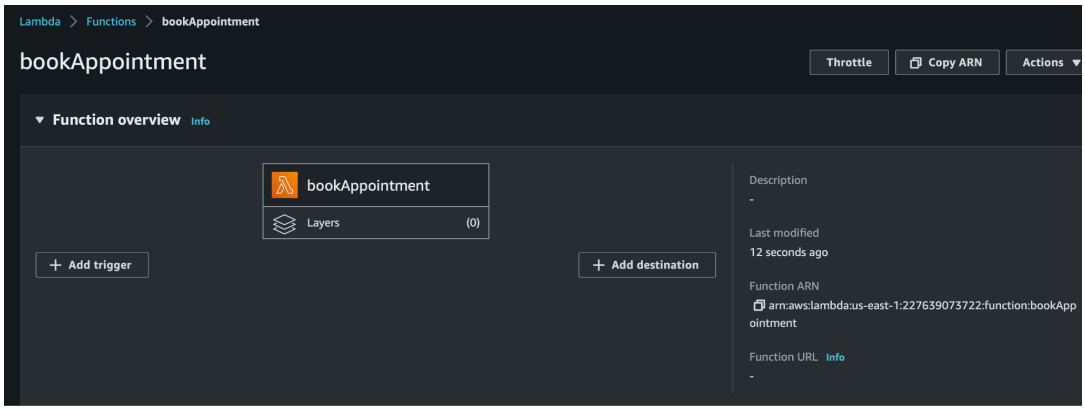
CP2 Update:



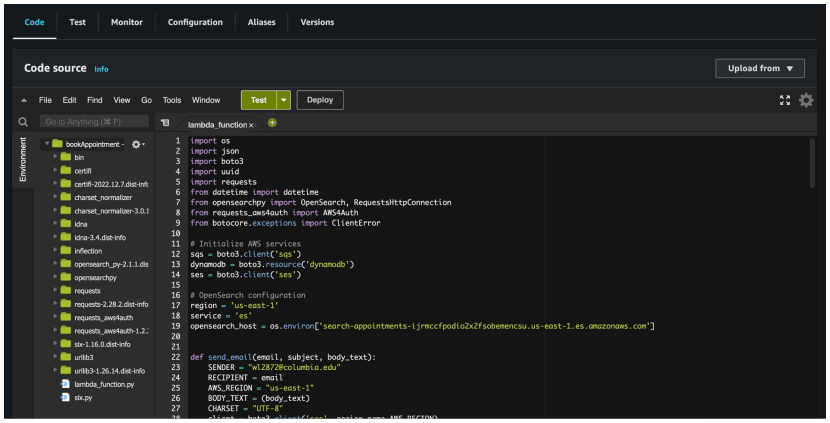
BookAppointment:
We have successfully completed the development of BookAppointment, which involves a Lambda function in AWS that is triggered by an SQS queue message. The message contains essential appointment details, including the doctor ID, patient ID, date, and time. The function then retrieves the necessary doctor and patient information from a DynamoDB table and creates a Calendly appointment for the user based on the provided date and time. Following that, the Doctor and Patient DynamoDB tables are updated with the appointment information, and the appointment is indexed in OpenSearch. Moreover, the function sends a confirmation email to both the patient and the doctor using Amazon SES. Finally, the function sends a response message indicating that the appointment has been booked and a confirmation email has been sent.

Here is a direct link to this API:
https://github.com/Weiyao-Li/CureSphere/blob/main/Lambda/bookAppointment.py

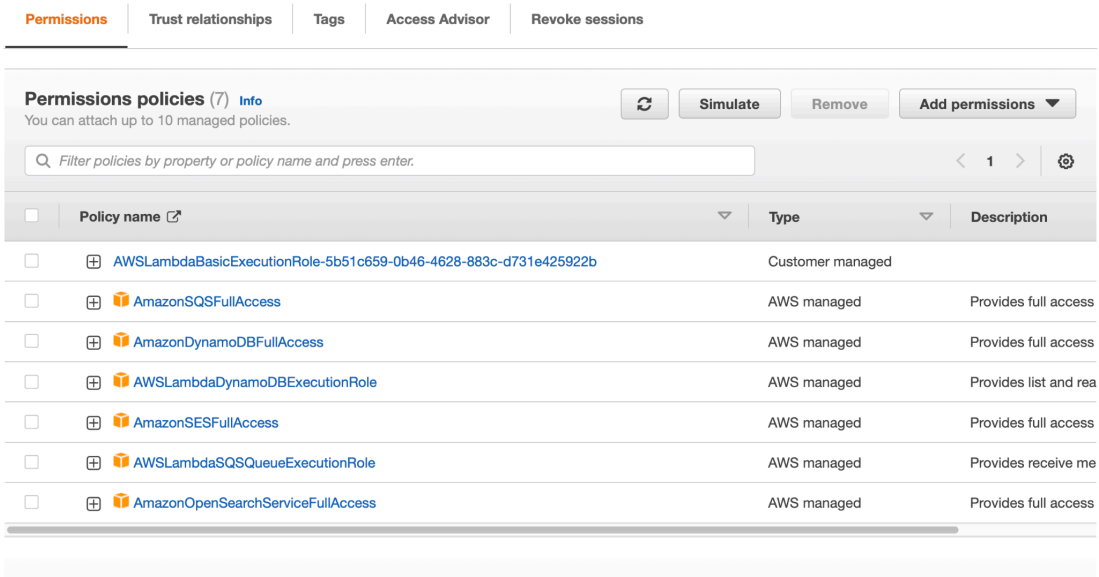
BookAppointment lambda has been created below:



Code has been placed into lambda_function with all necessary packages imported:



All necessary permissions have been granted:



CreateDoctor, CreatePatient and GetDoctor:
We have created the DynamoDB tables for patient and doctor records. We have created lambdas - createDoctor and createPatient, which are called by APIGateway, to create the doctor and patient records in the DynamoDB. The doctor and patient data are passed through HTTP headers. The appointment ID is initially set as an empty string, which is populated by bookAppointment whenever an appointment is booked.

getDoctor lambda is called by Lex to search the doctors DB for doctor recommendations. This lambda may be updated or replaced in future based on Lex implementation for doctor recommendations.

Demo video of lambdas called from API Gateway and records being created:
https://drive.google.com/file/d/1a-5vXui8ZcZf2aE0nDGMldQ442uaaCzL/view?usp=sharing

createDoctor lambda code:
https://github.com/Weiyao-Li/CureSphere/blob/main/Lambda/create_doctor.py

createPatient lambda code:
https://github.com/Weiyao-Li/CureSphere/blob/main/Lambda/create_patient.py

getDoctor lambda code:
https://github.com/Weiyao-Li/CureSphere/blob/main/Lambda/get_doctor.py

Doctors DB schema:
Primary key: doctor_id

Callout: We would be adding a validation in the frontend while calling CreatePatient and CreateDoctor where we check if the available_days and available_time_slots have equal length. (assumption: one time slot allowed to be mentioned per day)

```javascript
{
      doctor_id :str(email),
      firstName: str,
      lastName: str,
      department: str,
      specialty: str,
```

```
    available_days:[str, str],
    available_time_slots:[str, str],
    city: str,
    clinic_zip_code: int,
    email: str,
    appointment_id: str
}
```

Patients DB schema:
Primary key: patient_id

```JavaScript
{
    patient_id : str(email),
    firstName: str,
    lastName: str,
    age: int,
    gender: str,
    insurance_provider: str,
    city: str,
    zip_code: int,
    email: str,
    appointment_id: str
}
```

CallLex
Performs invocation of Lex chatbot for booking an appointment
https://github.com/Weiyao-Li/CureSphere/blob/main/Lambda/callLex.py

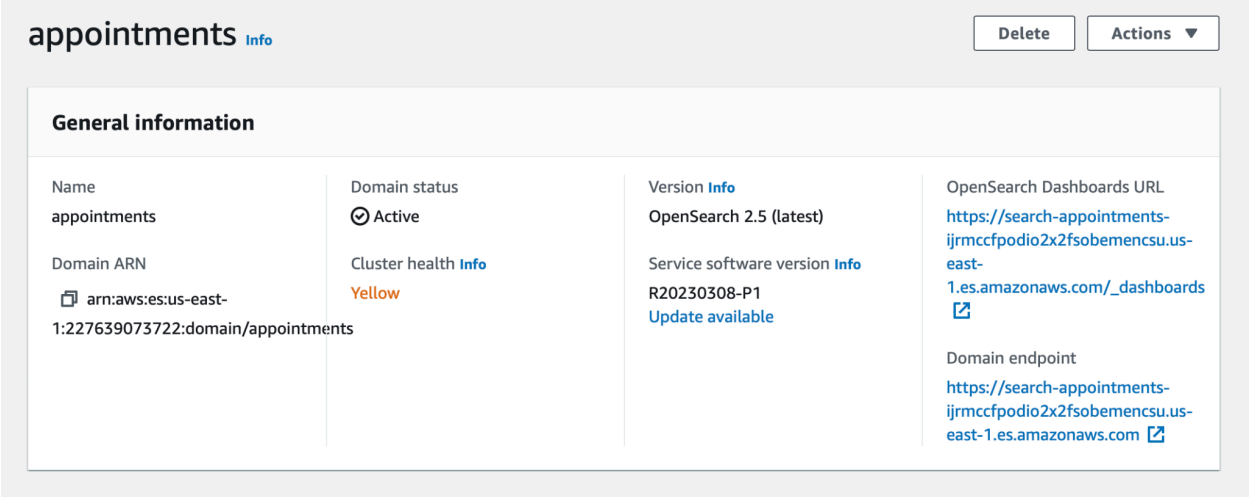
LexValidationFulfillment

The lambda that handles the lex outputs, sends message to SQS
https://github.com/Weiyao-Li/CureSphere/blob/main/Lambda/lexValidationFulfillment.py

Elastic Search, getAppointmentLink, getFeedback

We have created Elastic Search “appointments” and connected with 2 lambda functions, getAppointmentLink, getFeedback. The Instance type is t3.small.search, and the number of nodes is 1. The key is appointment_id, which would be generated when the appointment is booked and will be populated in the DB. For the purpose of this demo, we have hard-coded the appointment ID in the lambda code. Once we have done the APIGateway integration for this, we will query the appointment ID from the appropriate DB (patients or doctors) based on the role input and primary key (doctor_id or patient_id).

Feedback, Medicine, Appointment link can be searched with the key. (a_id, feedback, medicine, a_link) With getAppointmentLink, the appointment link, which is relevant to the query appointment_id, is returned, and with getFeedback, feedback is returned. Those lambda functions were tested, and we have attached the demo video link below.




```
HOST = 'search-appointments-ijrmccfpodio2x2fsobemencsu.us-east-1.es.amazonaws.com'
INDEX = 'appointment'
```

Search Result Example :

```
{'a_id': 'a_id1', 'feedback': 'fb1', 'medicine': 'm1', 'a_link':
'a_link1'}
```

Demo video for testing lambda functions :
https://drive.google.com/file/d/1CVRcNPadqKezeokVKXfBcMLW1SdrOG1A/view?usp=sharing

getAppointmentLink lambda code :
https://github.com/Weiyao-Li/CureSphere/blob/main/Lambda/getAppointmentLink.py

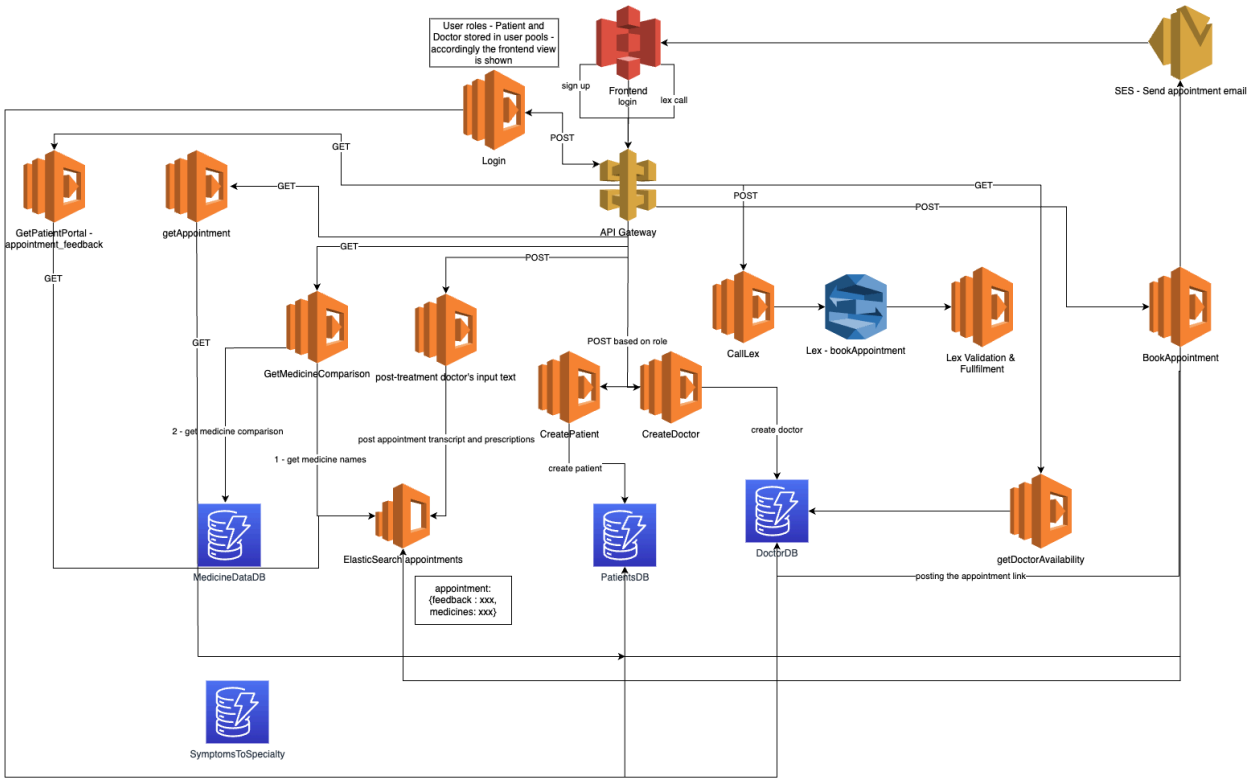
getFeedback lambda code :

https://github.com/Weiyao-Li/CureSphere/blob/main/Lambda/getFeedback.py

Extra to CP2:

- Having decided to use the React framework for the frontend, we have built the starter code with all the necessary packages included, which now runs successfully. We will begin our frontend work from this point.
  Here is a direct link to the frontend:
  https://github.com/Weiyao-Li/CureSphere/tree/main/frontend

Final Architecture:



User flow:

Register as Patient on CureSphere

Log in

Get medicine comparison by location and price of sellers

Patient Portal

Open CureSphere Chatbot

Provide 2-3 symptoms you are facing and your zipcode location

Displays a list of relevant doctors closest to you

Select the doctor and book an appointment

Email sent to doctor and patient

Appointment shown in current appointments

Doctor portal

Provide prescription and medicines

post appointment phase

Reflects on patient portal

Reflects on patient portal

Appointment phase

Log in

Register as Doctor on CureSphere