Group Members:

Anna Anello

Alyssa Reinagle

Avi Rathod

Table of Contents

1.0 Objective

EZ-Health was designed to aid the doctor-patient relationship when receiving and accessing medical information and making it accessible to those who are unfamiliar with medical jargon. The purpose of this document is to display the creation process of this application and how to use it.

2.0 Problem Statement

2.1 Problem

The difficulty and usability issues of modern patient-facing health applications have been well documented. The caveat of these systems is that there are no viable alternatives to medical software that prioritize the patient's experience and understanding of their own medical information. Miscommunication with doctors and caregivers after appointments can lead to patient confusion. Online patient portals are difficult to use and are typically eyesores that are overwhelming to navigate even for tech-savvy patients.

2.2 Solution

The aim of our project is to create a web application to streamline the user interface in order to serve the patient a more user-friendly approach to healthcare documentation with a focus in UI/UX design, providing information in a format that the patient can easily understand. Utilizing simple, straightforward language and visuals in online documentation will provide a more user-friendly solution to the current standard of patient portals.

2.3 Client

Our target client will at first be local healthcare practices and then we would like to expand across the country. Our stakeholders include those that will be using this system like doctors and patients.

2.4 Users

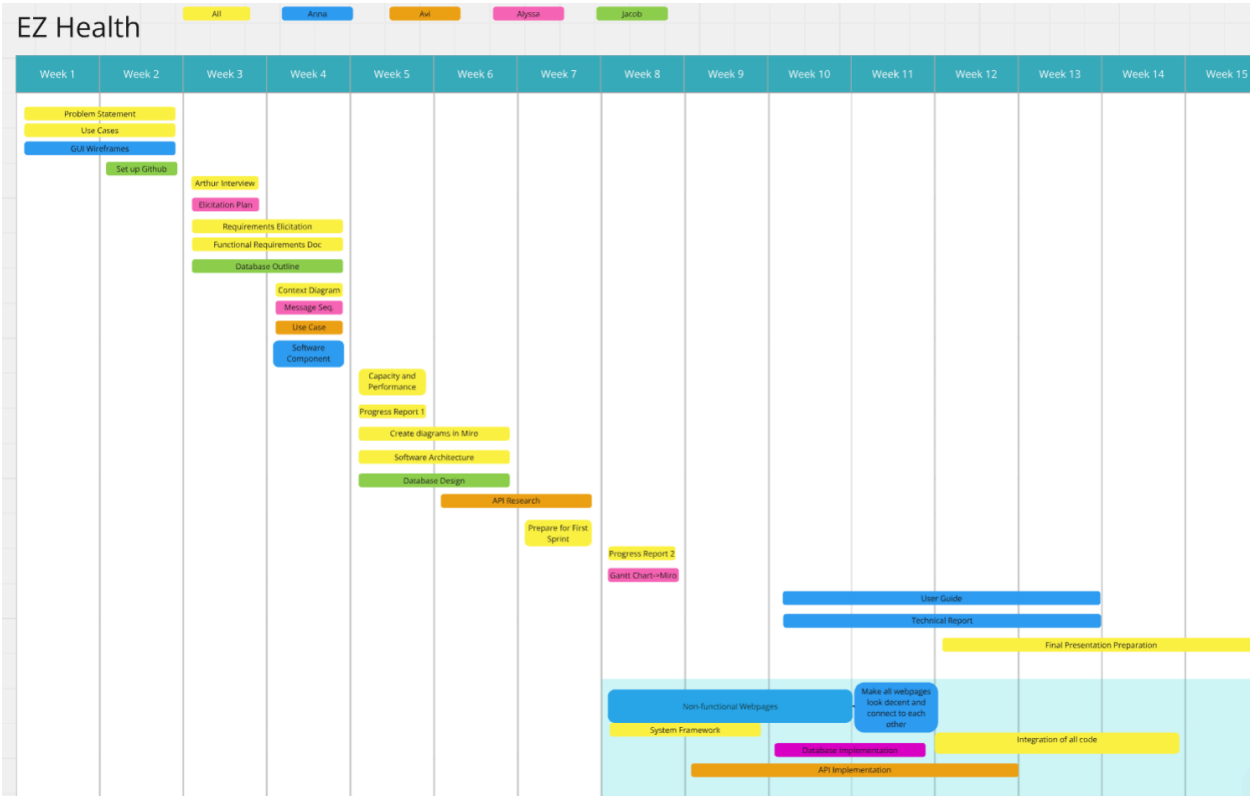
Our primary users will be doctors, healthcare professionals, and patients. As long as the user is able to access the internet they will be able to access the website.

2.5 Client Constraints

Due to protection of privacy, only doctors and a patient's medical team can access a patient's medical information. A doctor will have a list of patients and can view each individually. The application itself is only accessible through an internet browser. It is recommended to use a laptop or a PC instead of a mobile device to access the website.

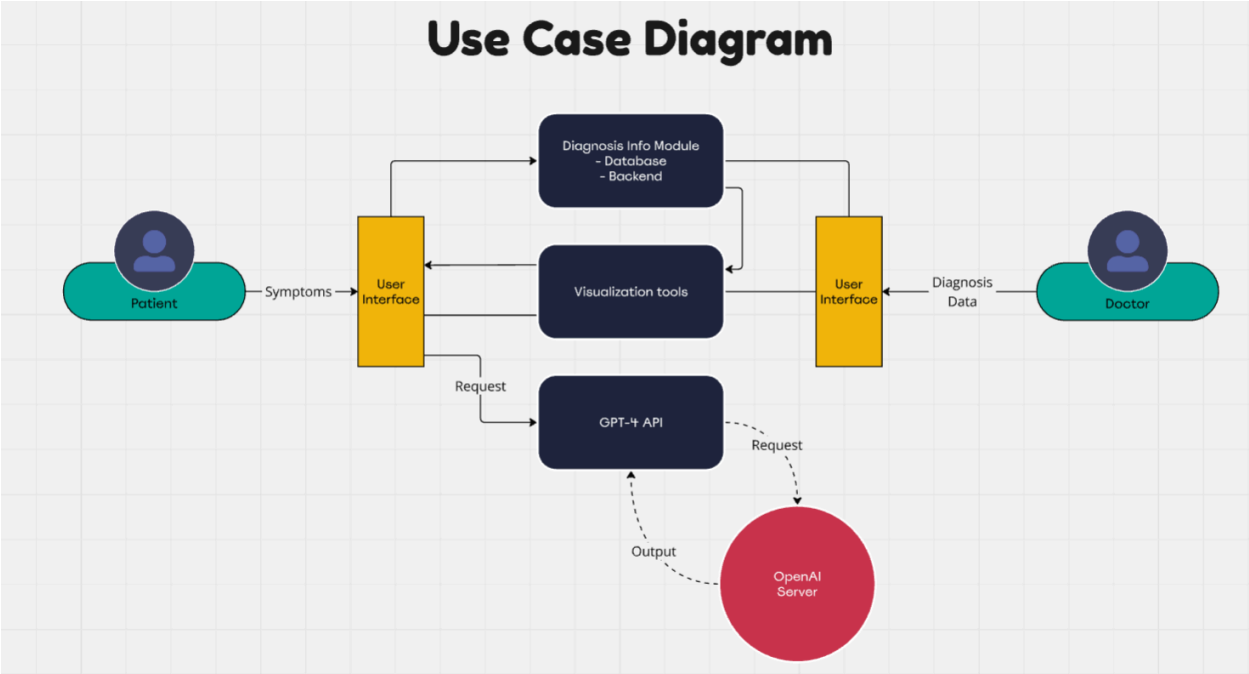
3.0 Elicitation Plan and Results

Our plan involves meeting with a former medical professional in order to discuss system requirements and application functionality. We have created interview questions for our medical professionals. We also have personal connections to a few medical professionals at Cleveland Clinic who have given advice. We took their advice and integrated it into our application in order to keep in mind that medical jargon is a large issue in the healthcare field. There is also a study that shows that 91% of patients interviewed preferred a lack of jargon being used in order to more easily understand(6). To keep track of progress we created a gantt chart on Miro(2).



4.0 Usage Scenarios

There are two main types of users: patients and doctors. In order to use the application the user must access the website.



4.1 Patient

The patient will be able to access the homepage, login, create and view a user profile, log symptoms, ask a chatbox questions, and also submit support tickets.

4.2 Doctor

The doctor/ medical professional will be able to access the homepage, login, create and view their profile, view their patient's profiles, add symptoms, upload a patient's medical information and notes, log a patient's symptoms, and submit support tickets.

5.0 User Interfaces

This is the primary way our users will interact with our application.

5.1 Website

We have created a website in order to use our application. It is accessible to any device with an internet browser and a stable connection. We recommend viewing the application on a laptop or a PC rather than a mobile device.

5.2 Menu and Website Navigation

There is a drop down menu in the upper left corner of each webpage in order to ensure easy and quick navigation between all of the different pages of the website.

5.3 Screens

5.3.1 Homepage

This is the page that a user will first land on. It has a brief description of the website's capabilities and there is a button to allow the user to login.

5.3.2 Login

Allows either a patient or a medical professional to login to the application.

5.3.3 Patient Profile

The patient's profile allows the patient to create and view their user profile on the application. This profile is able to be viewed by their medical professional.

5.3.4 Doctor Profile

The doctor profile allows the medical professional to create and view their user profile. Their patients can view their profile.

5.3.5 Ask an AI chat box

The patient can ask an AI chat box if they have any medical related questions.

5.3.6 Patients List

For the doctor view they can view a list of their patients and be able to click on the patient in order to add medical information.

5.3.7 Symptoms

Users can visualize and track their symptoms. Doctors can add symptoms as well.

5.3.8 AI Notes

The doctor can upload a patient's medical notes and then the AI will remove any jargon so the patient can more easily understand.

5.3.9 Support Ticket

If a user has an issue with the application they can submit a support ticket.

5.4 Reports and Contents

For a fully deployed and live project, our report generation could look like website traffic, frequency of issues submitted by users via help tickets, error tracking for the database and OpenAI API connections, server errors, and performance reports.

5.5 Documentation and Training Guides

These will be used to aid the user on how to use the application.

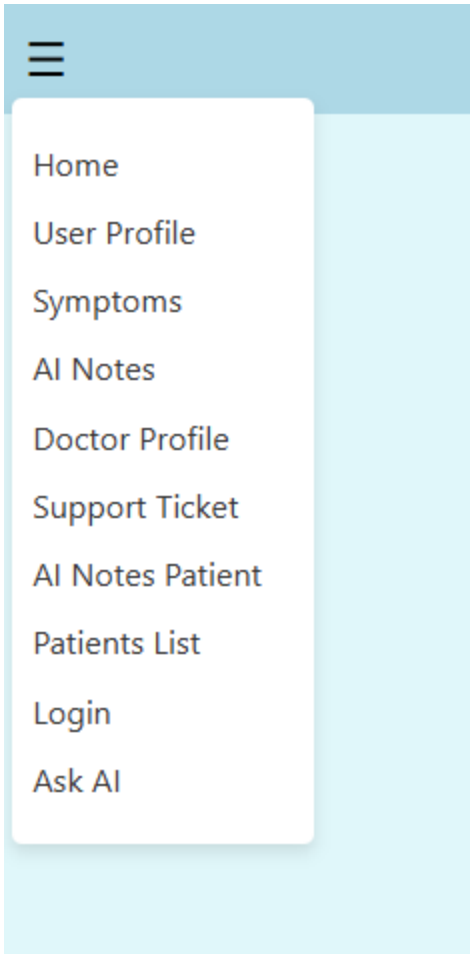
5.5.1 Patient Training Guide

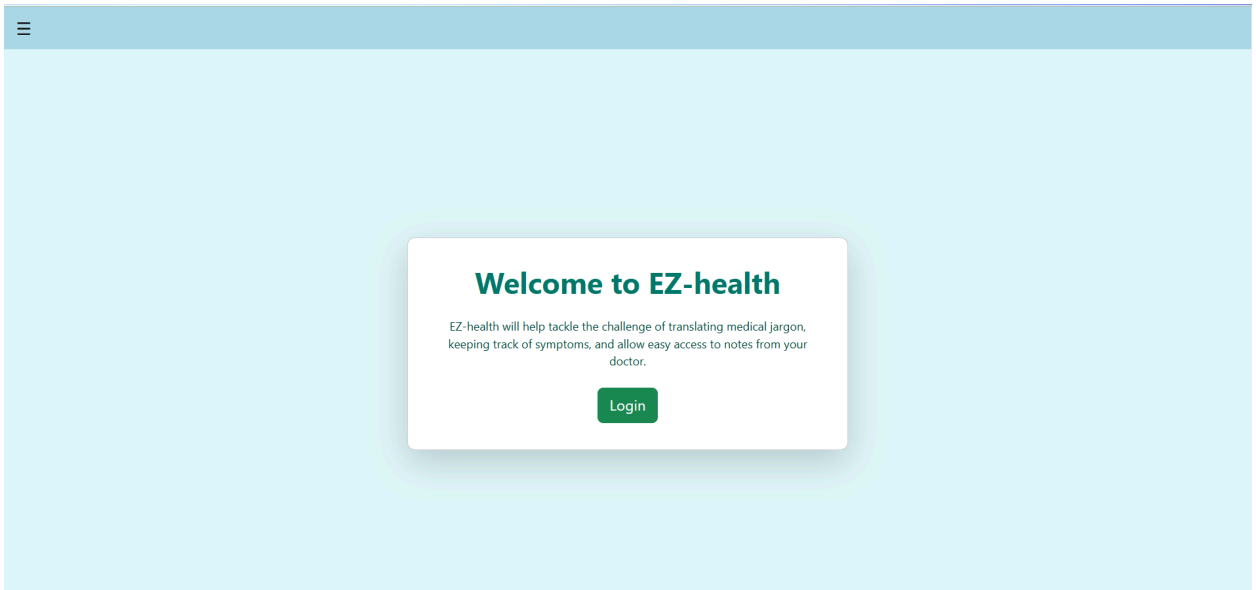
Please open your internet browser of choice and head to our github https://github.com/jakebutkovic/EZHealth.
Click on the blue button that says code and open a codespace. Once in the codespace you will need to open the code in visual studio code and download Node.js. You will also need to npm install, npm install openai(5), npm install react-router-dom, npm install –lts, npm use –lts, and npm react-bootstrap. Once this is all installed you can run the application. Type cd ez-health/ into the terminal. To run type npm start into the terminal and the application will open in your internet browser. The user will be able to login and create a profile, view and add symptoms, view medical information, and ask the AI chatbot any questions.

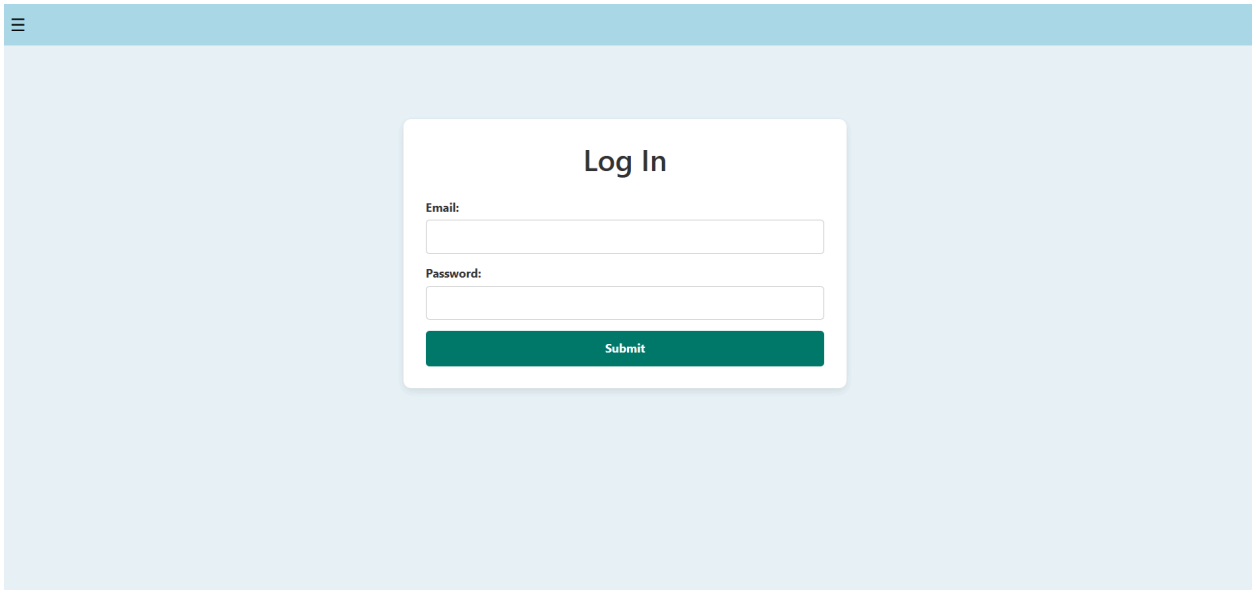
5.5.2 Doctor Training Guide

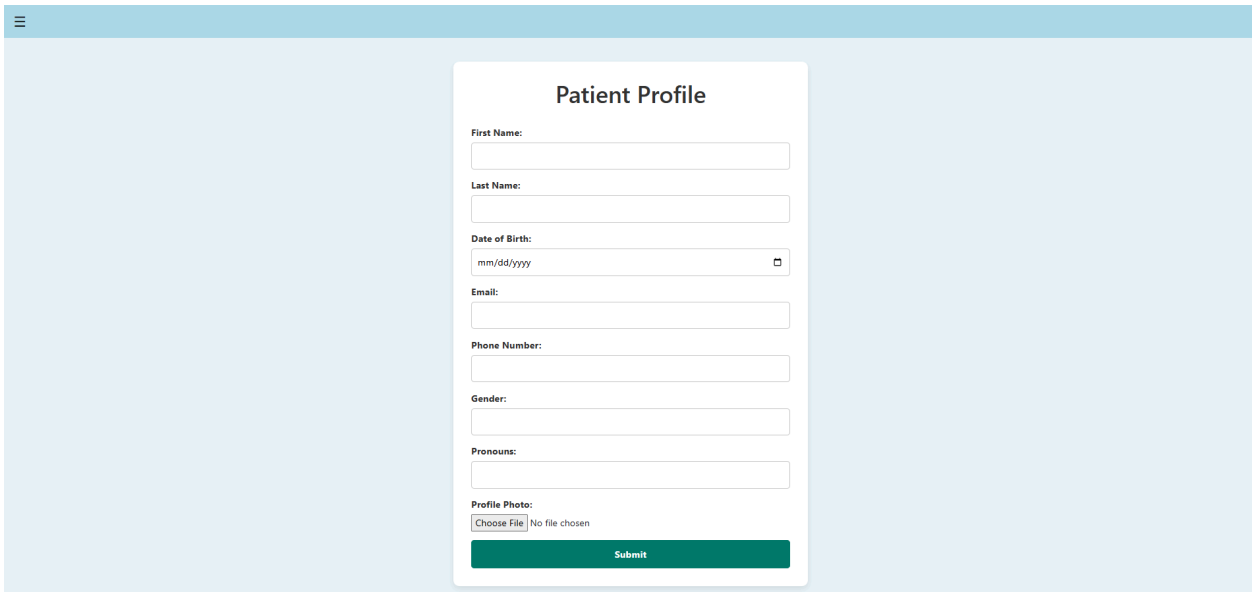
Please open your internet browser of choice and head to our github https://github.com/jakebutkovic/EZHealth.
Click on the blue button that says code and open a codespace. Once in the codespace you will need to open the code in visual studio code and download Node.js. You will also need to npm install, npm install openai, npm install react-router-dom, npm install –lts, npm use –lts, and npm react-bootstrap. Once this is all installed you can run the application. Type cd ez-health/ into the terminal. To run, type npm start into the terminal and the application will open in your internet browser. The user will be able to login and create a profile, view and add to their patient list, view and add symptoms, view medical information of their patients, add medical information, and ask the AI chatbot any questions.

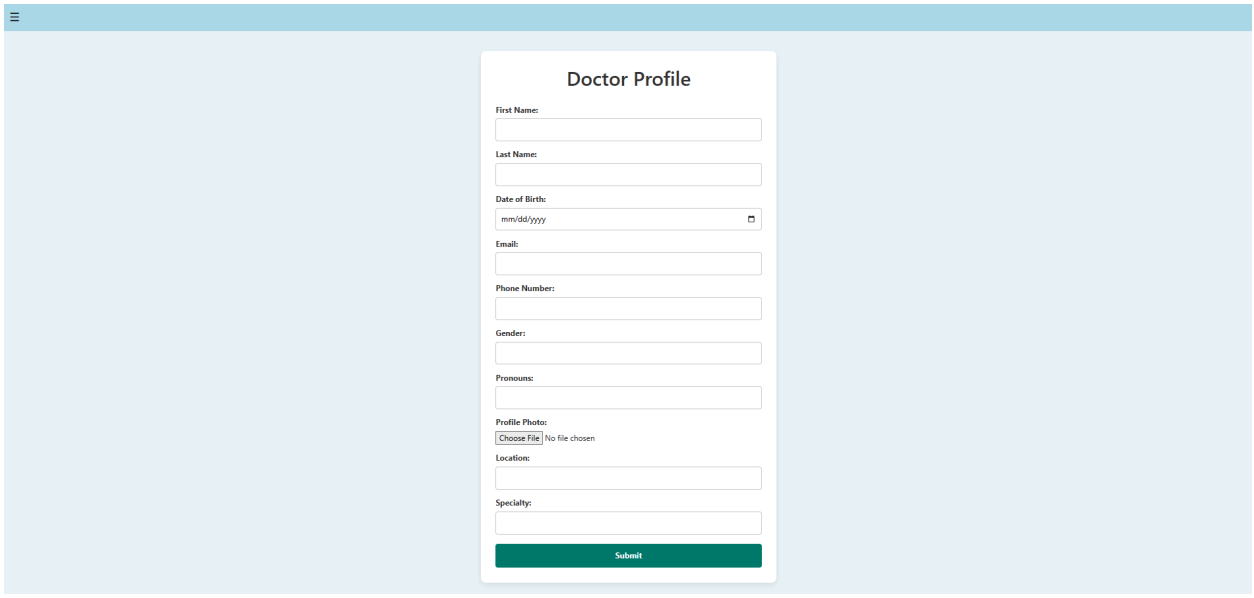
5.6 Application Screenshots

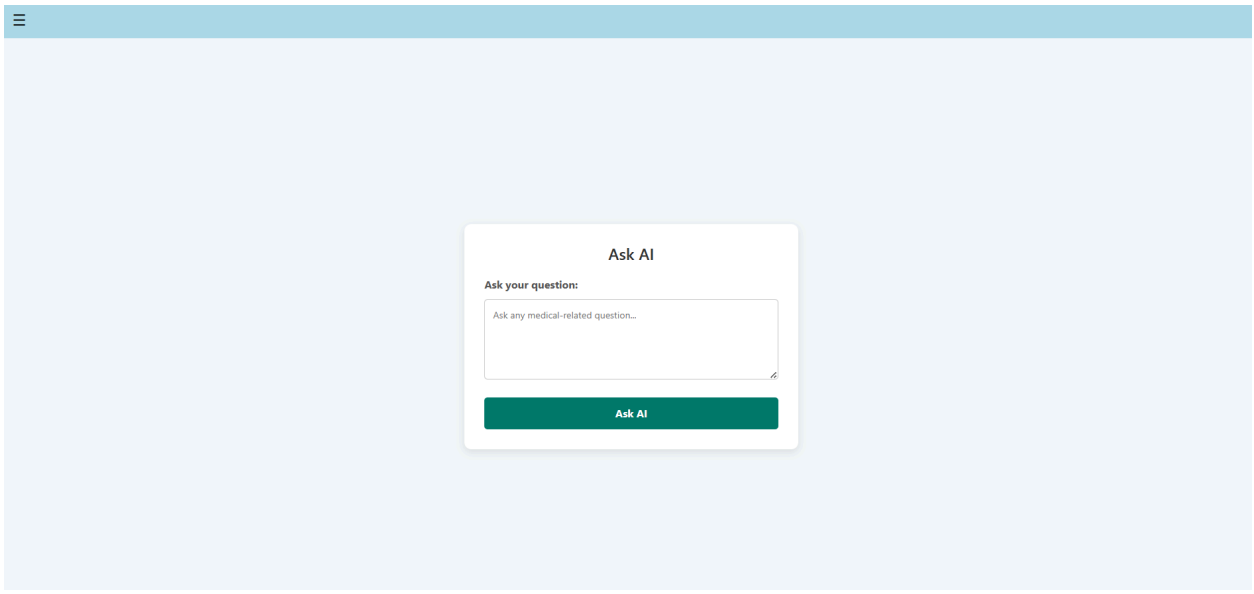

Home
User Profile
Symptoms
AI Notes
Doctor Profile
Support Ticket
AI Notes Patient
Patients List
Login
Ask AI

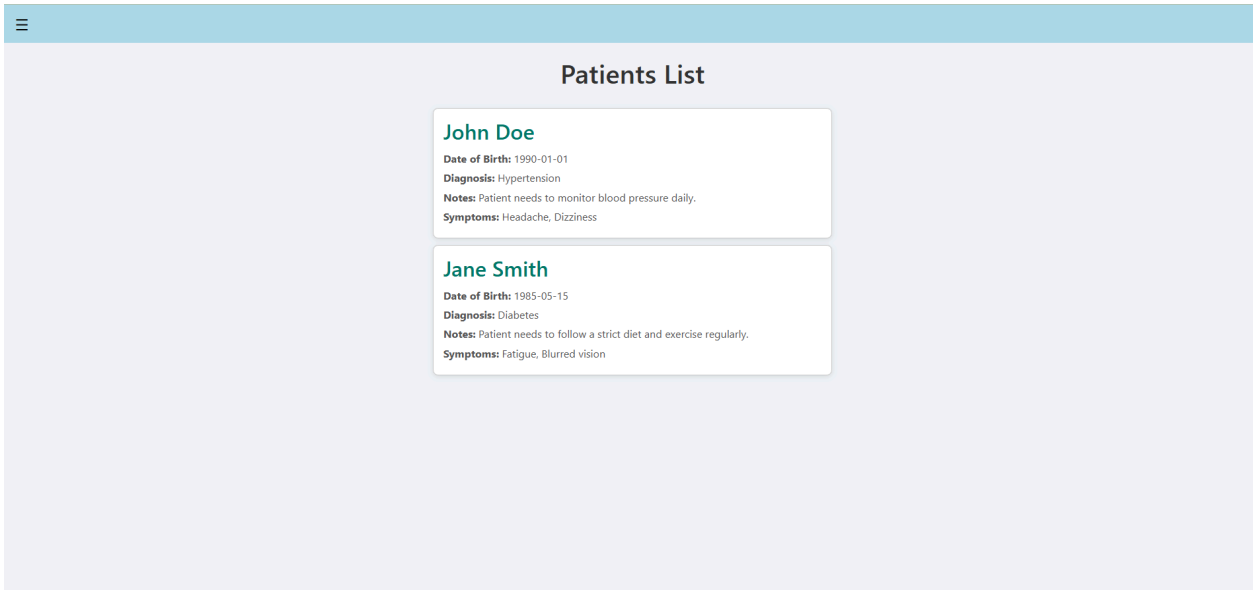

Welcome to EZ-health
EZ-health will help tackle the challenge of translating medical jargon, keeping track of symptoms, and allow easy access to notes from your doctor.
Login


Log In
Email:
Password:
Submit

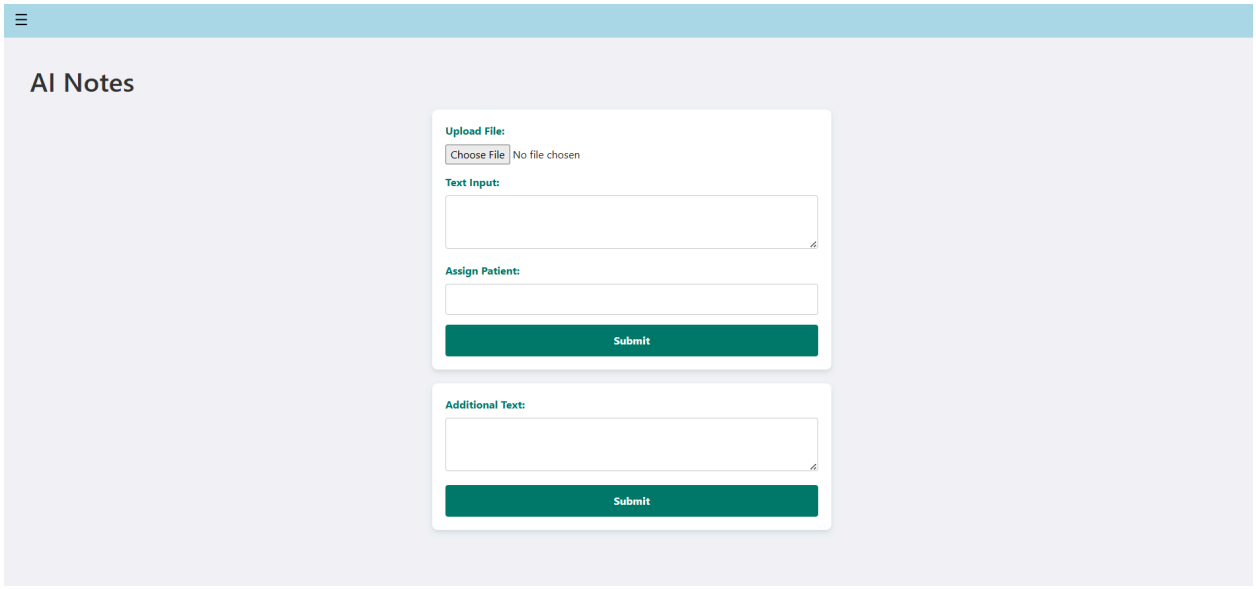

Patient Profile
First Name:
Last Name:
Date of Birth:
mm/dd/yyyy
Email:
Phone Number:
Gender:
Pronouns:
Profile Photo:
Choose File
No file chosen
Submit

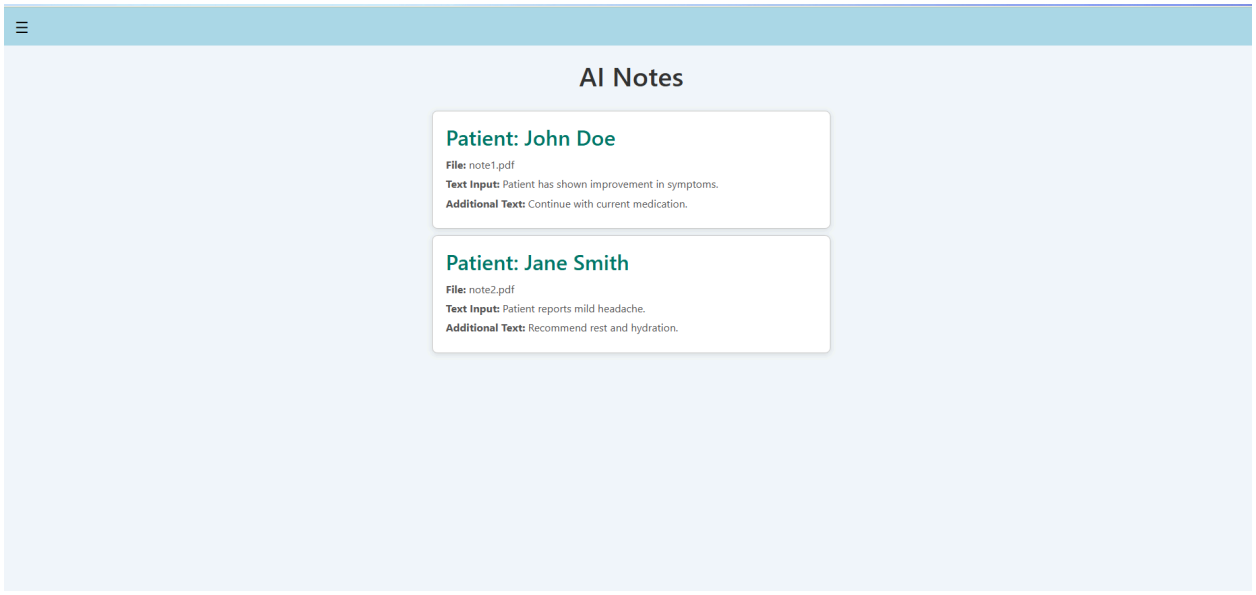

Doctor Profile
First Name:
Last Name:
Date of Birth:
mm/dd/yyyy
Email:
Phone Number:
Gender:
Pronouns:
Profile Photo:
Choose File
No file chosen
Location:
Specialty:
Submit

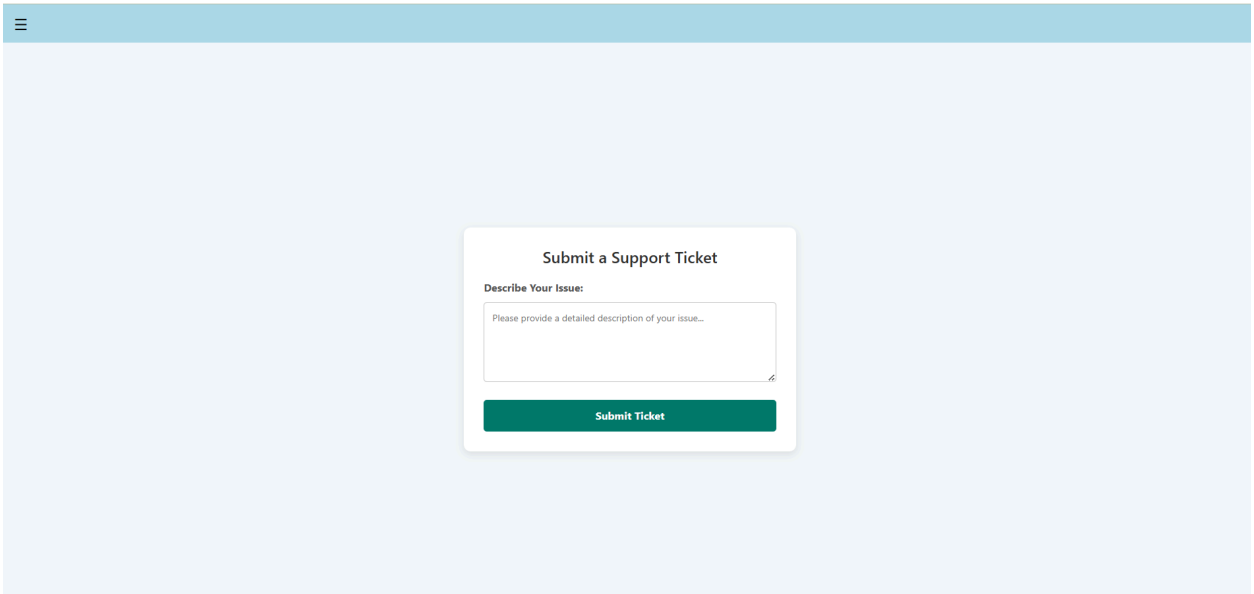

Ask AI
Ask your question:
Ask any medical-related question...
Ask AI


Patients List
John Doe
Date of Birth: 1990-01-01
Diagnosis: Hypertension
Notes: Patient needs to monitor blood pressure daily.
Symptoms: Headache, Dizziness
Jane Smith
Date of Birth: 1985-05-15
Diagnosis: Diabetes
Notes: Patient needs to follow a strict diet and exercise regularly.
Symptoms: Fatigue, Blurred vision


Symptoms
Headache
Date: 2023-10-01
Severity: 5
Notes: Mild headache in the morning
Date: 2023-10-02
Severity: 7
Notes: Severe headache in the evening
Fever
Headache Severity Over Time
2023-10-01: Severity 5
2023-10-02: Severity 7


AI Notes
Upload File:
Choose File
No file chosen
Text Input:
Assign Patient:
Submit
Additional Text:
Submit


AI Notes
Patient: John Doe
File: note1.pdf
Text Input: Patient has shown improvement in symptoms.
Additional Text: Continue with current medication.
Patient: Jane Smith
File: note2.pdf
Text Input: Patient reports mild headache.
Additional Text: Recommend rest and hydration.



6.0 Application Functionality

The application is a website created using React(7), MySQL(4), and Flask(1). Users and doctors alike can view and update medical information. The AI is done using an OpenAI API(3) in order to reduce medical jargon and answer any questions the patient may have.

7.0 Systems Architecture

This portion covers the system architecture and design of our application. We created a website which allows the majority of users to be able to access and use our application.

7.1 Conceptual Systems Overview

We wanted a single website that allows two sets of users to create accounts and view and edit their medical information. To run everything all user info is stored in a MySQL database.

7.2 Database Schema

The database consists of 7 tables. The tables are as follows: users, doctors, patients, symptoms, files, lifestyle recommendations, tickets. Our schema visual was generated using MySQL Workbench, however we ultimately chose to use SQLite for our database.

7.2.1 User Table

The user table holds login information for users such as email and password, as well as name and user ID information.

7.2.2 Doctor Table

The doctor table holds profile information relevant to the doctor’s professional practice such as specialization information.

7.2.3 Patient Table

The patient table is a central component of the database schema. Because of this, relatively little patient information is stored in the patient table, besides connection

relationships to their doctor, user ID and profile creation date. The following tables use patient_id as their foreign key.

7.2.4 Symptom Table

Patients' symptom logs are entered into the symptoms table with information on the severity of the symptom, a short description, and the date. This information is accessed to create symptom tracker charts.

7.2.5 File Table

Patients can upload files to their profile for medical recordkeeping.

7.2.6 Lifestyle Recommendations Table

If a patient requests lifestyle recommendations generated by AI, they can be saved in the table for later reference.

7.2.7 Ticket Table

The ticket table holds a history of technical help ticket submissions.

7.3 Database Relationship Functionality

The Relational Database Diagram (Figure 7.3) shows the relationships between the tables. Users have a one to one relationship with patients and doctors because each patient or doctor has a user ID. Users have a one to many relationship with tickets because a user can submit many tickets. Doctors have a one to many relationship with patients. In our schema, we chose to assign one doctor per patient, but if multiple doctors per patient were necessary, our diagram could be updated accordingly. Patients have a one to many relationship with symptoms, lifestyle recommendations, and files. Patients can have many of these types of inputs.

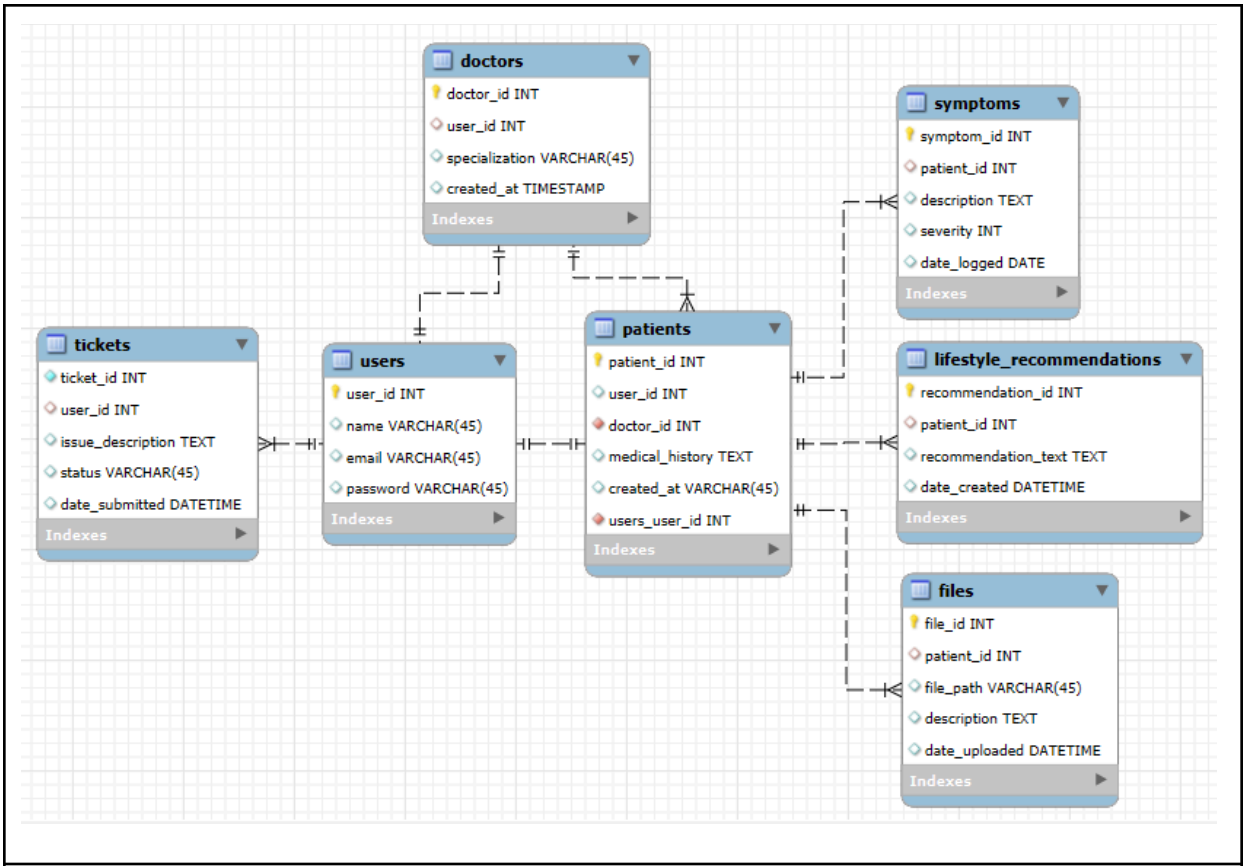


Figure 7.3: EZ-Health Relational Database Diagram

7.4 Performance and Throughput

For the best performance, we recommend at least a 20 Mbps wifi connection for our application to run.

8.0 Error Recovery

This helps resolve some of the errors that could occur while using EZ-Health.

8.1 Database Connections

Database connection errors can occur due to a number of reasons, including incorrect file opening and loading, invalid connection, specified port is unavailable, or a technical design flaw in the database schema resulting in errors. These issues should be submitted to a help ticket.

8.2 Web Server Connection

There's a few different reasons the web server may create an error. An error could be created by any of the following: high network traffic, the database may be down, or a faulty WIFI connection. In order to resolve these issues please check your internet connection first. If that doesn't work, wait a few minutes and then refresh the application.

8.3 User Access

This error occurs when the user login attempts fail. The user will be notified that their login attempt failed and will be prompted to try again.

9.0 References

1. Flask. (n.d.). *Flask documentation (stable release).* Retrieved November 20, 2024, from https://flask.palletsprojects.com/en/stable/
2. Miro. (n.d.). *Miro online collaborative whiteboard platform.* Retrieved November 20, 2024, from https://miro.com/
3. MuleSoft. (n.d.). *What is an API?* Retrieved November 20, 2024, from https://www.mulesoft.com/api/what-is-an-api
4. MySQL. (n.d.). *MySQL documentation.* Retrieved November 20, 2024, from https://dev.mysql.com/doc/
5. OpenAI. (n.d.). *Platform overview.* Retrieved November 20, 2024, from https://platform.openai.com/docs/overview
6. PubMed Central (PMC). (2024). *Perceptions of jargon-free communication in healthcare.* Retrieved November 20, 2024, from https://pmc.ncbi.nlm.nih.gov/articles/PMC9983080/
7. React. (n.d.). *React API reference.* Retrieved November 20, 2024, from https://react.dev/reference/react

10.0 Glossary

Application Programming Interface (API)- A software intermediary that allows two applications to communicate

Application- A program that helps a user perform a task

Artificial Intelligence (AI)- Technologies that allow computers to perform advanced tasks through a variety of algorithms

Data- Value or a set of values that represent a specific concept or concepts

Database- structured set of data

Documentation- Material that provides official information on a topic

Desktop- Personal computer which is designed for casual use at a desk or other stationary location

Elicitation- Process of gathering information

Flask- python based web framework usually used for backend development

Hardware- physical components of a computer

Hierarchy- System of organization

Homepage- Introductory page of a website

Implementation- process of putting a plan into effect

MySQL- Open-source relational database management system

SQLite- Serverless database

Personal Computer (PC)- computer designed for personal use

React- Javascript user interface development framework

Schema- Representation of a plan or a theory in the form of a model or an outline

Software- Programs and other opening information used by a computer

Stakeholder- A person with an interest in the product

Throughput- Amount of data that can pass through a system or a process

UI- User interface, visual elements that the user can interact with

Usage Scenarios- description of individuals using the particular software

UX- User experience

Web Browser- Software application that allows users to access and interact with the world wide web

Webpage- A single page that is hosted on a website

Web Server- a computer system that stores and delivers content for websites to clients

WIFI- wireless networking technology that uses radio waves to provide wireless high-speed internet access