UNLOCKING ACCESS TO HEALTHCARE: EVALUATING THE EFFICACY OF AI-DRIVEN DIAGNOSIS IN A MOBILE APPLICATION

Mylyn Zheng[1], Adam Grant[2]

[1]Portola High School, 1001 Cadence, Irvine, CA 92618
[2]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

In today's fast-paced world, accessing quality healthcare can often feel like an uphill battle. Whether hindered by geographical barriers, long waiting times, or financial constraints, many individuals struggle to receive timely diagnoses for their health concerns. Recognizing this pressing need, we've developed an innovative solution in the form of our cutting-edge app [7].Harnessing the power of artificial intelligence, our app offers users a seamless pathway to diagnosis based on their reported symptoms. Our journey began with meticulous research to identify the most reliable AI technology, leading us to integrate ChatGPT, a state-of-the-art Large Language Model renowned for its ability to mimic professional medical responses accurately [8]. To ensure seamless functionality, we complemented this AI prowess with a robust Firebase database, facilitating efficient data storage and retrieval [9].Following rigorous development and testing phases, we're thrilled to announce that our app delivers unparalleled accuracy in diagnosis. Our trials revealed an impressive average accuracy rating of 1.8 out of 2 for common illnesses, underscoring the reliability and efficacy of our diagnostic algorithms. While initial evaluations highlighted room for improvement in symptom diversity across a spectrum of 21 illnesses, our focus on refining the app's capabilities for prevalent conditions has yielded remarkable results.Ultimately, our app stands as a beacon of accessibility in the realm of basic healthcare. By bridging the gap between individuals and vital medical expertise, we're empowering users to take proactive control of their well-being, irrespective of their circumstances. With our unwavering commitment to innovation and excellence, we're proud to revolutionize healthcare delivery and make a meaningful difference in people's lives.

KEYWORDS

Artificial Intelligence, Flutter, App Development, Health Tracking

1. INTRODUCTION

Advancing technology was meant to make healthcare more convenient, but thousands, if not millions of people still struggle with affordable and accessible healthcare. While the population grows, the healthcare workforce is also aging and retiring, creating a problematic ratio between patients and workers. To make things worse, burnout has resulted in higher turnover rates in the healthcare department, contributing to the issue of understaffed healthcare units and the waning access to quick and easy healthcare [1]. Access to medical care isn't the only problem, though. The cost of healthcare has increased recently due to the increased cost of prescription and technology. The average primary doctor visit costs $112-$172 with insurance, while ones without insurance can cost two to five times as much [2]. These staggering costs cause more than 20% of Americans to ignore their health for the sake of saving money [3].

One way AI is already present in healthcare is monitoring patients. Certain hospitals use AI to watch over patients 24/7. The problem with this is that it is only provided for those already in the hospital and those who can pay for hospital stay. Our use of AI is accessible to everyone. On the other hand, when trying to make healthcare accessible to everyone, healthcare kiosks were made so that people can get help without going to a hospital and waiting to get seen. The only problem is the people still have to drive to one of these kiosks and, at the end, are still referred to a doctors office or hospital to get seen. Our app allows users to get a diagnosis wherever they are. Lastly, when looking at AI making diagnosis, a project from Cornell University trained a machine learning model to aid doctors and other machines in healthcare diagnosis. Unlike their project, our app directly connects the user to that AI and works with physical health.

To alleviate some of the harms of modern-day healthcare, we propose a solution in the form of an app in which users can consult for simple healthcare advice and track their past symptoms. This app will use Artificial Intelligence to give feedback to users based on symptoms that are given [10]. This solution will allow users to consult something about their health concerns faster than waiting for a doctor's appointment. In addition, the app will be free to use. This will allow people who can't afford basic doctor visits and people who don't visit in fear of high costs to get basic medical advice from their home. In short, this solution should be a quick and easy way for people to inquire about symptoms they are having without the fear of the costs of a doctor's visit.

The experiments we ran were to test the accuracy of the AI and the symptoms. First, we ran the app multiple times with symptoms from different illnesses to see if the AI will give the correct diagnosis. The scoring was out of 2 points with 0 being completely incorrect, 1 being somewhat correct and 2 being correct. The average of all the tests came out to be 1.8, meaning the AI was highly accurate. The second test we did was to see if the options for symptoms were enough to give an accurate diagnosis. Many illnesses share most of their symptoms and only have a few unique ones that set them apart. We wanted to see if we could accommodate all these symptoms while also making the app easy to use. To do so, we used a government health website to find the symptoms of 21 different illnesses and see how many of them are present in the app. The final score was a percentage and most were in the 90-100% category. Any that didn't end up in this category were either missing only one or had a very rare disease. Overall, the app functions well when working with common illnesses, which is what the app was made for.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. A suitable AI

The first challenge that we faced was finding a suitable AI to implement into the app. The AI should be at least somewhat credible since the app works with a person's health. Unfortunately, we were not able to find a suitable AI for this project, most likely because AI's that are advanced and credible enough for healthcare are usually kept very private and very expensive. Instead of finding an AI specific to healthcare, we could find one that was trained with a large amount of data, including articles, books, and other texts about healthcare. However, this brings up another problem.

2.2. The information

How can we be sure, with an AI so diverse, we will get the answers we want? There's no guarantee that the AI won't misinterpret what is given, especially when the prompt we will give may not be written perfectly. The AI could decide to give us the definition of a cough instead of a cure, which would defeat the point of this project. To tackle this problem, we decided to include instructions for the AI every time we call it. We will tell it to act as a medical professional in its response to ensure that the user's experience is as similar to visiting one in person as possible.

2.3. Storing the data

The last problem was storing the data. We wanted to make sure the users could look back on what their input and the output they got. Currently, the app does not keep track of the user's answers in each step, which makes it hard to include all responses and inputs at the end of the AI questioning process. Luckily, in Flutter, each page is actually a function which can be customized to require certain parameters. This way, we can pass previous inputs and responses through the screens as parameters.

3. Solution

The three main components of this program are the LLM, or the AI, the database, which stores all the users information, and the user interface, especially during the questioning and response portion. The user starts by creating an account and logging into the app. They are first greeted by the home screen, which allows them to choose between changing their settings by clicking on their profile, adding a symptom, or checking their past entries. If they decide to add a symptom, they will be taken to a screen where they can choose to respond by clicking on a few symptoms, like coughs and fevers, clicking on the image of the human. Clicking on the human will then take them to a screen where they can select a specific body part, describe the discomfort they are feeling by answering an AI generated follow up question, and receive their AI diagnosis on the final results screen [11]. On the other hand, if they chose to pick certain symptoms, they will be taken directly to the results screen. Once they review their diagnosis, the app will take them back to the home screen and their inputs and the results will be saved in the database. Back at the homescreen, users can view all their past symptoms by clicking on the "Past Symptom" button.

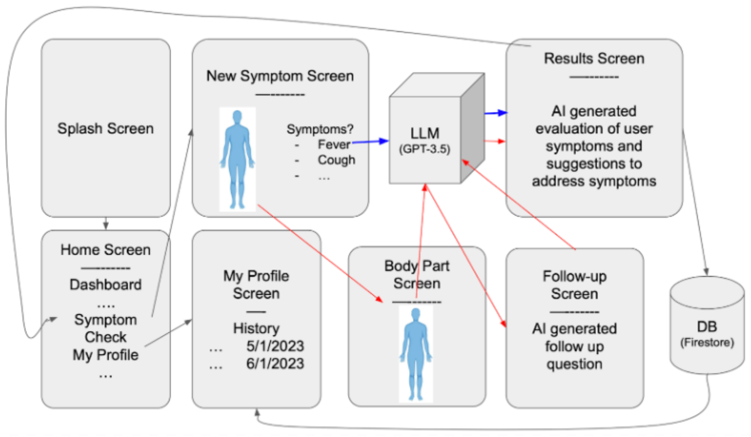
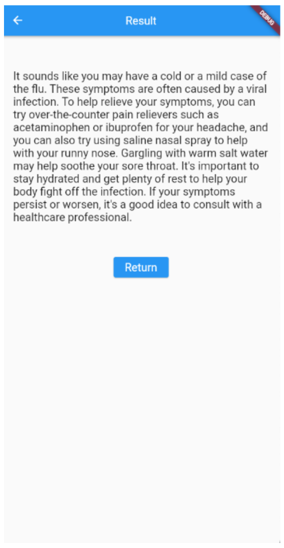


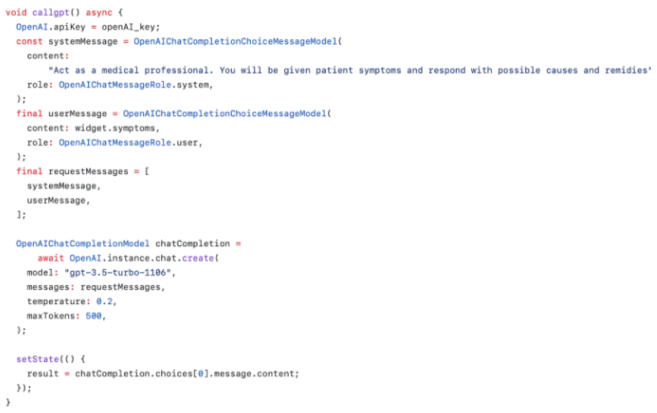
Figure 1. Overview of the solution

The most important part of this program would likely be the AI because it is what's evaluating and responding to the symptoms given. We chose to use ChatGPT, an LLM, or Large Language Model. This LLM is a machine learning model trained with large amounts of text data to, initially, complete user sentences but was later fine tuned to answer user inquiries.



It sounds like you may have a cold or a mild case of the flu. These symptoms are often caused by a viral infection. To help relieve your symptoms, you can try over-the-counter pain relievers such as acetaminophen or ibuprofen for your headache, and you can also try using saline nasal spray to help with your runny nose. Gargling with warm salt water may help soothe your sore throat. It's important to stay hydrated and get plenty of rest to help your body fight off the infection. If your symptoms persist or worsen, it's a good idea to consult with a healthcare professional.

Figure 2. Screenshot of the result



```
void callgpt() async {
  OpenAI.apiKey = openAI_key;
  const systemMessage = OpenAIChatCompletionChoiceMessageModel(
    content:
        "Act as a medical professional. You will be given patient symptoms and respond with possible causes and remidies'
    role: OpenAIChatMessageRole.system,
  );
  final userMessage = OpenAIChatCompletionChoiceMessageModel(
    content: widget.symptoms,
    role: OpenAIChatMessageRole.user,
  );
  final requestMessages = [
    systemMessage,
    userMessage,
  ];

  OpenAIChatCompletionModel chatCompletion =
      await OpenAI.instance.chat.create(
    model: "gpt-3.5-turbo-1106",
    messages: requestMessages,
    temperature: 0.2,
    maxTokens: 500,
  );

  setState(() {
    result = chatCompletion.choices[0].message.content;
  });
}
```

Figure 3. Screenshot of code 1

The code shown created an async functions called “callgpt,” which calls the OpenAI API to get a personalized response for the user [12]. The method is called twice, once if the user decides to describe discomfort in a body part and another time when the user reaches the results screen. When ran, the function takes the input from the user and forms a query that will be given to OpenAI. Along with the query, we have included instructions for the AI to make sure the response given is like ones that doctors will give. In addition, we kept the temperature low, at 0.2, to ensure the reply will not be random but more focused around the users health. After getting a response, the function will dissect the message to only get the necessary information and store it in a variable previously created.

Another component of our program is the UI, or user interface. It is created with the purpose of being easy for the user to use. The app is made of mostly buttons, with only one place where the user must type something in. In addition, there is a clear path of how the user enters their symptom, making it easy for the user to navigate throughout the app.

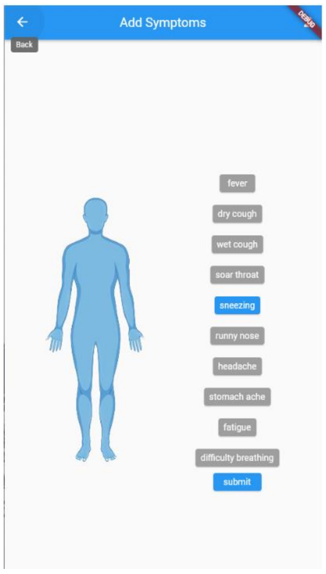
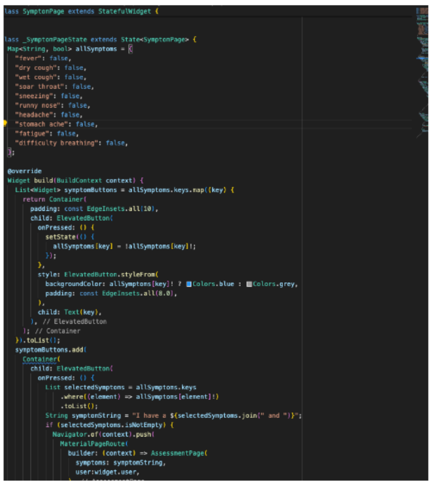


Figure 4. Screenshot of the add symptoms

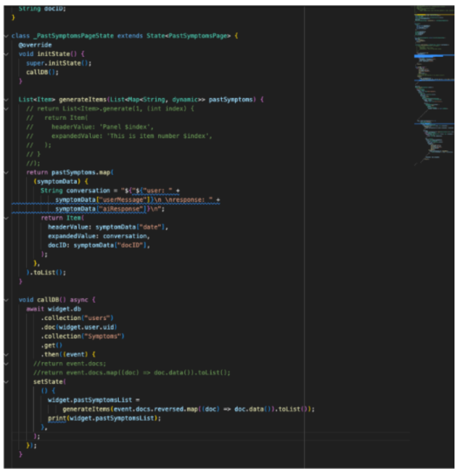


```
lass SymptonPage extends StatefulWidget {


lass _SymptonPageState extends State<SymptonPage> {
 Map<String, bool> allSymptoms = {
   "fever": false,
   "dry cough": false,
   "wet cough": false,
   "soar throat": false,
   "sneezing": false,
   "runny nose": false,
   "headache": false,
   "stomach ache": false,
   "fatigue": false,
   "difficulty breathing": false,
 };

 @override
 Widget build(BuildContext context) {
   List<Widget> symptomButtons = allSymptoms.keys.map((key) {
     return Container(
       padding: const EdgeInsets.all(10),
       child: ElevatedButton(
         onPressed: () {
           setState(() {
             allSymptoms[key] = !allSymptoms[key]!;
           });
         },
         style: ElevatedButton.styleFrom(
           backgroundColor: allSymptons[key]! ? Colors.blue : Colors.grey,
           padding: const EdgeInsets.all(8.0),
         ),
         child: Text(key),
       ), // ElevatedButton
     ); // Container
   }).toList();
   symptomButtons.add(
     Container(
       child: ElevatedButton(
         onPressed: () {
           List selectedSymptoms = allSymptoms.keys
               .where((element) => allSymptoms[element]!)
               .toList();
           String symptomString = "I have a ${selectedSymptoms.join(" and ")}";
           if (selectedSymptoms.isNotEmpty) {
             Navigator.of(context).push(
               MaterialPageRoute(
                 builder: (context) => AssessmentPage(
                   symptoms: symptomString,
                   user:widget.user,
```

Figure 5. Screenshot of code 2

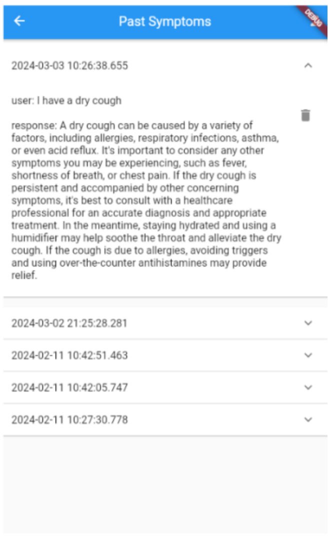
The code shown uses the list of possible symptoms given to create a column of buttons. The buttons act as toggles, where the user only has to click each button to add the symptom. Once they click the button, the button will change from grey to blue and the code will acknowledge that the symptom was chosen and when the submit button is clicked, the code will then send this information to the OpenAI code shown previously. This is one of the two ways the user can enter their symptoms and this path does not require any typing, just a few clicks, making it easy for users to use. The human shown on the left of the buttons is another way the user can enter their symptoms. That path requires the user to click the part of the body that is bothering them then answer one short question on how it is bothering them. Similar to the other path, this path is also short and easy to use.

The last component is the database component. It stores the user's symptom and the AI response each time they complete their AI interaction [13]. The code then displays it in the "Past Symptoms" screen for them to refer to later on.



```dart
  String docID;
}

class _PastSymptomsPageState extends State<PastSymptomsPage> {
  @override
  void initState() {
    super.initState();
    callDB();
  }

  List<Item> generateItems(List<Map<String, dynamic>> pastSymptoms) {
    // return List<Item>.generate(1, (int index) {
    //   return Item(
    //     headerValue: 'Panel $index',
    //     expandedValue: 'This is item number $index',
    //   );
    // }
    //);
    return pastSymptoms.map(
      (symptomData) {
        String conversation = "${"${"user: " +
            symptomData["userMessage"]}\n \nresponse: " +
            symptomData["aiResponse"]}\n";
        return Item(
          headerValue: symptomData["date"],
          expandedValue: conversation,
          docID: symptomData["docID"],
        );
      },
    ).toList();
  }

  void callDB() async {
    await widget.db
        .collection("users")
        .doc(widget.user.uid)
        .collection("Symptoms")
        .get()
        .then((event) {
      //return event.docs;
      //return event.docs.map((doc) => doc.data()).toList();
      setState(
        () {
          widget.pastSymptomsList =
              generateItems(event.docs.reversed.map((doc) => doc.data()).toList());
          print(widget.pastSymptomsList);
        },
      );
    });
  }
```

Figure 6. Screenshot of the code 3



2024-03-03 10:26:38.655

user: I have a dry cough

response: A dry cough can be caused by a variety of factors, including allergies, respiratory infections, asthma, or even acid reflux. It's important to consider any other symptoms you may be experiencing, such as fever, shortness of breath, or chest pain. If the dry cough is persistent and accompanied by other concerning symptoms, it's best to consult with a healthcare professional for an accurate diagnosis and appropriate treatment. In the meantime, staying hydrated and using a humidifier may help soothe the throat and alleviate the dry cough. If the cough is due to allergies, avoiding triggers and using over-the-counter antihistamines may provide relief.

2024-03-02 21:25:28.281

2024-02-11 10:42:51.463

2024-02-11 10:42:05.747

2024-02-11 10:27:30.778

Figure 7. Screenshot of past symptoms

After the first component, the AI, is ran and the response is displayed, the users inputs and the AI's response is stored in the database under the users name. Then, from the homescreen, the user can look at their past interactions by going to the "Past Symptoms" page, which will run the code shown above. The code accesses the database, in this case the Firestor database, and looks for the ID that matches the user's ID. Once it is found, the code will access all the data under that ID and create a list of the data that will then be displayed as a drop down item as shown above. The code will use the date it was created as the title and will also display both what the user imputed and what the AI responded with. The code also allows the user to delete any set of data that they want with the trash icon.

4. Experiment

4.1. Experiment 1

One thing we wanted to test was the accuracy of the AI. The AI is the biggest part of our project and it is important that it runs well since it is giving medical advice.

To test the accuracy, we will find common illnesses that people have and the symptoms that go along with it. Then, we will randomly choose between some of the symptoms and put it in the app to test the AI. The response given will be given a rating between 0 and 2 depending on how accurate it is. If the illness is mentioned as one of the main possibilities, then the response will get two. If the illness is mentioned as a second possibility or a less specific version of the illness is mentioned as one of the main ones (a viral infection instead of a flu) then the response gets a 1. If the broader version of the illness is mentioned as a second option or the illness is not mentioned at all, the response gets a 1. The illnesses we chose were the flu, strep throat, and allergies. Each illness was tested 5 times with different symptoms to get a score out of 10 and the overall is the average of all the points.

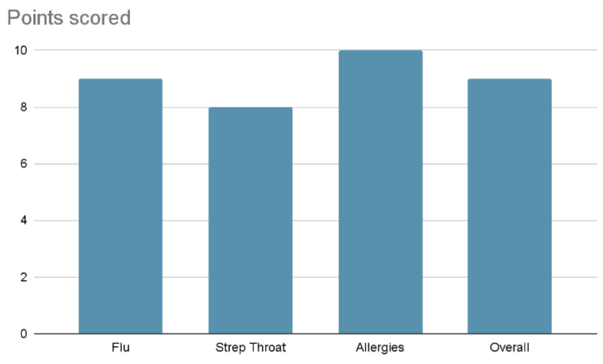


Figure 8. Figure of experiment 1

The average of the three different illnesses was a 9 and the average for each test was a 1.8. The highest rating received was a 2 while the lowest was a 1. More than two thirds of that test got a 2, meaning the AI was very accurate. The tests that received a 1 were the ones where the symptoms given could apply to any illness, meaning the AI could not give a definite answer. The high outcome is likely a result of certain symptoms being related to certain illnesses. For example, allergies always had a two if shortness of breath, itchiness, or rash was one of the symptoms and the same with pain while swallowing and strep throat.

4.2. Experiment 2

One blind spot that we wanted to test was the number of choices. The options of symptoms the user is given is limited to a few common ones that are seen every day. This means that some symptoms are unable to be logged and can affect the AI's response.

To test if the number of options given is enough, we choses to compare it to symptoms given online. We used a government certified health website, the Illinois Department of Public Health, to find the symptoms of 21 illnesses. We then compared them to the options we have to see how many of the symptoms given match the options we give. We then convert this data into a percentage for each illness. Of course, we cannot accommodate every single symptom, but a percentage will allow us to look past very specific ones and focus on the important and common ones that we lack.

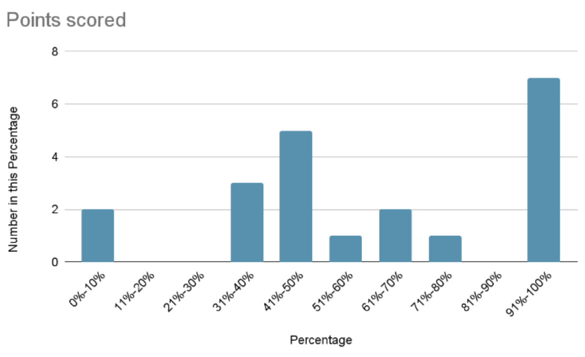


Figure 9. Figure of experiment 2

Most of the results were concentrated in the 91%-100% category or the 41%-50% category. In these two categories, most of the data was at the end of the range as in 50% or 100%. The 50% often came from missing one possible symptom out of the given 2. The highest scoring results usually came from common illnesses such as different viruses like Covid, while the lower scoring results came from less commonly occurring and less easily diagnosed illnesses such as colon cancer. This implies that this app would be good for diagnosing common everyday illnesses and not very dangerous and less seen ones. This works well with our goal of providing basic healthcare to everyone.

5. Related Work

AI has been implemented in areas such as monitoring recovering patients to reduce the need for a nurse 24/7 [4]. Shaheen discusses various multimodal machine learning algorithms from multiple different healthcare sectors. These applications of machine learning are highly accurate with data trained for the purpose of health care. In addition, it does help reduce the need for nurses and will inform any medical professional of a problem. The downsides are that these kinds of machines are not open for the general public and are very expensive to implement. This means that people not already in the hospital and using this equipment still need medical attention no matter how serious their situation is. The health app created in this project is accessible to regular mobile users. They do not have to already be in the hospital, nor do they have to pay a great amount for the machine, instead they only have to use their phone.

Falling away from AI, in 2008, there was a project to implement healthcare kiosks to ensure accessible healthcare to everyone. These kiosks would work like ATMs, located everywhere and customized to a person once they are logged in. The kiosks take the user's information, including their heartbeat, breathing, and other information usually given at a doctor's appointment before referring the user to a hospital that will help with their specific issue [5]. Even though the kiosks take data more accurately than the app does, its main disadvantage is the traveling that must be done. First, the user has to find an available kiosk to go to. Then, once the results are given, the user then has to go to the provided healthcare facility. Our solution is stored in the user's mobile device, meaning they can use it wherever they are.

Lastly, similarly to how we used a large language model in our app, a project from Cornell University trained a machine learning model to produce diagnosis and remedies based on a few symptoms [6]. The difference is that their model works with already given data and does not work directly with patients while ours directly work with users and give them feedback in real time. In addition, their machine learning model focuses on mental health while ours focuses on a person's basic physical health.

6. CONCLUSIONS

One limitation of the app, as stated previously, is the lack of options available and the lack of accuracy when choosing the symptoms [15]. For the section where you can choose your symptoms, you can only choose from common symptoms like coughs or fevers, and for the area where you choose your discomfort in a body part, where you clicked in not fully accurate since the code divides the body into groups, such as the whole are or the whole leg. If We had time, we would either create smaller sections that you can click to be more accurate or potentially use AI to determine where it was clicked. Another limitation we would like to fix is how the app is a one and done deal. Even though you can look at your past symptoms, there is not much else to help with long term. We would have liked to use AI to flag certain conversations that could be long term or very dangerous and to look over the past logs over a certain range of time to see, based on long term data, if there are any problems that need medical attention.

Overall, the app does its job by providing basic healthcare advice faster than going to a doctor would and at no cost. This project shows how Artificial Intelligence can improve the healthcare field and make it more accessible to everyone [14].

REFERENCES

[1] Malik, Shafaq, et al. "Mr. Doc: a doctor appointment application system." arXiv preprint arXiv:1701.08786 (2017).
[2] Chan, Leighton, et al. "Disability and health care costs in the Medicare population." Archives of physical medicine and rehabilitation 83.9 (2002): 1196-1201.
[3] Newhouse, Joseph P. "Medical care costs: how much welfare loss?." Journal of Economic perspectives 6.3 (1992): 3-21.
[4] Shaheen, Mohammed Yousef. "Applications of Artificial Intelligence (AI) in healthcare: A review." ScienceOpen Preprints (2021).
[5] Verma, Anubha, Harsh Dhand, and Abhijit Shaha. "Healthcare kiosk next generation accessible healthcare solution." HealthCom 2008-10th International Conference on e-health Networking, Applications and Services. IEEE, 2008.
[6] Ji, Shaoxiong, et al. "Mentalbert: Publicly available pretrained language models for mental healthcare." arXiv preprint arXiv:2110.15621 (2021).
[7] Iyer, Lakshmi Shankar. "AI enabled applications towards intelligent transportation." Transportation Engineering 5 (2021): 100083.
[8] Kasneci, Enkelejda, et al. "ChatGPT for good? On opportunities and challenges of large language models for education." Learning and individual differences 103 (2023): 102274.
[9] Li, Guoliang, Xuanhe Zhou, and Lei Cao. "AI meets database: AI4DB and DB4AI." Proceedings of the 2021 International Conference on Management of Data. 2021.
[10] Holzinger, Andreas, et al. "Causability and explainability of artificial intelligence in medicine." Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 9.4 (2019): e1312.
[11] Keane, Pearse A., and Eric J. Topol. "With an eye to AI and autonomous diagnosis." NPJ Digital Medicine 1.1 (2018): 40.
[12] Sun, Albert Yu, et al. "Does fine-tuning GPT-3 with the OpenAI API leak personally-identifiable information?." arXiv preprint arXiv:2307.16382 (2023).
[13] Yang, Qian, et al. "Re-examining whether, why, and how human-AI interaction is uniquely difficult to design." Proceedings of the 2020 chi conference on human factors in computing systems. 2020.
[14] Shaheen, Mohammed Yousef. "Applications of Artificial Intelligence (AI) in healthcare: A review." ScienceOpen Preprints (2021).
[15] Janowski, Lucjan, and Margaret Pinson. "The accuracy of subjects in a quality experiment: A theoretical subject model." IEEE Transactions on Multimedia 17.12 (2015): 2210-2224.