UNIVERSIDAD SURCOLOMBIANA

BodySense

AI System

By: Olaya Pacheco, Angelica María. Andrade Chila, Oscar Julián and Guzman Medina, Catalina

Abstract

This report presents the development and implementation of a software project that aims to provide users with a chat-based platform integrated with artificial intelligence (AI). The software allows users to register and engage in conversations with an AI system, where they can upload images of body parts experiencing pain. The AI analyzes the images and provides potential causes or diseases related to the affected area. Additionally, the system recommends nearby hospitals for further examination and treatment. The project aims to enhance user accessibility to preliminary medical information and facilitate prompt medical attention.

Key words: Body parts, AI, Hurt, Analyzes, Hospital.

INDEX TABLE

1. Introduction

The purpose of this software project is to create a user-friendly chat application embedded with an AI system capable of analyzing images and providing potential diagnoses for areas of bodily pain. In today's fast-paced world, individuals often face challenges in accessing healthcare resources and obtaining immediate medical advice. This software project addresses this issue by harnessing the power of AI to offer preliminary diagnostic insights.

The application facilitates user registration, ensuring a personalized experience while maintaining data privacy and security. By allowing users to upload images of the affected area, the AI analyzes visual cues and employs advanced image processing techniques to identify potential causes or diseases associated with the user's symptoms. Moreover, the system leverages geolocation services to recommend nearby hospitals, ensuring timely access to professional medical care.

2. Materials and Methods

- **Materials:**

For the development of this project "BODYSENSE," the following materials were primarily used: Visual Studio Code as the development environment, Angular as the frontend framework, Python with Flask framework for the backend, Postman for testing backend web services, and PgAdmin for working with a PostgreSQL database.

- **Procedure:**

First, Python, Angular, Visual Studio Code, Postman, and PgAdmin were installed. Data of medical centers in Neiva was collected to store it in a database, aiming to provide the nearest center to the user in case of an emergency. Additionally, a table was created to store basic information of users who create an account in BODYSENSE.

FRONTEND:

- The HOME component was created using the command "ng g c HOME," which generates a folder named HOME containing three files:

home.component.html: File that contains the structure of a section of the page.

home.component.css: File that contains the style of home.component.html.

home.component.ts: File that contains the functionality of this section of the page.

- The CHAT component was created using the command "ng g c CHAT," which generates a folder named CHAT containing three files:

chat.component.html: File that contains the structure of a section of the page.
chat.component.css: File that contains the style of chat.component.html.
chat.component.ts: File that contains the functionality of this section of the page.

- The LOGIN component was created using the command "ng g c LOGIN," which generates a folder named LOGIN containing three files:

login.component.html: File that contains the structure of a section of the page.

login.component.css: File that contains the style of login.component.html.

login.component.ts: File that contains the functionality of this section of the page.

- The SIGN UP component was created using the command "ng g c SIGNUP," which generates a folder named SIGNUP containing three files:

signup.component.html: File that contains the structure of a section of the page.

signup.component.css: File that contains the style of signup.component.html.

signup.component.ts: File that contains the functionality of this section of the page.

A folder named Services was created to hold different services that will be created.

- The REST service was created using the command "ng g s Services/REST," which generates a file named rest.service.ts in the services folder. This service handles communication with the Flask server that will be created later.
- The USERNAME service was created using the command "ng g s Services/USERNAME," which generates a file named username.service.ts in the services folder. This service handles communication between the previously created components.
- A TOKEN service was created for session management on the website.

In the app-routing.module.ts file, which was created during the Angular installation, routing configurations were set.

In the app.module.ts file, which was created during the Angular installation, module configurations were set.

BACKEND:

First, a folder named Backend was created. Inside this folder, another folder named src was created, which contains the different files that make up the entire backend.

Within the src folder, the following files were created: app.py, config.py, .env, and db.py.

The .env file contains the necessary variables.

The config.py file contains server configurations.

The db.py file contains the database connection.

The app.py file contains the backend functionality.

3. Experimental Setup

First, the user enters the website, which redirects them to the home screen. On the home screen, there are three buttons:

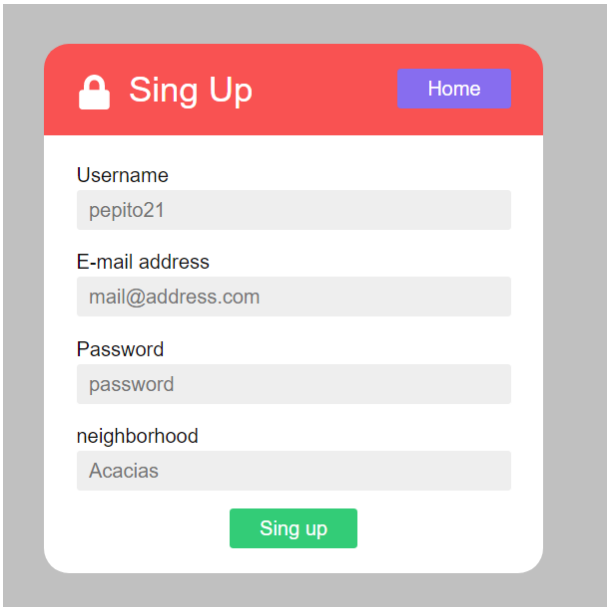
- "Login" button: Takes the user to the login page.
- "Sign In" button: Takes the user to the registration page.
- "Chat" button: Only becomes active if the user is logged in. Takes the user to the chat zone.

On the login page, there are two fields: one for the username and one for the password.

- When the user clicks the "Login" button, it sends a request to the backend to verify the credentials.

- If the credentials are correct, the user is logged in and a message is displayed indicating a successful login. Once the message is closed, the user is redirected to the chat zone.

On the registration page, there are four fields: username, email, password, and neighborhood.

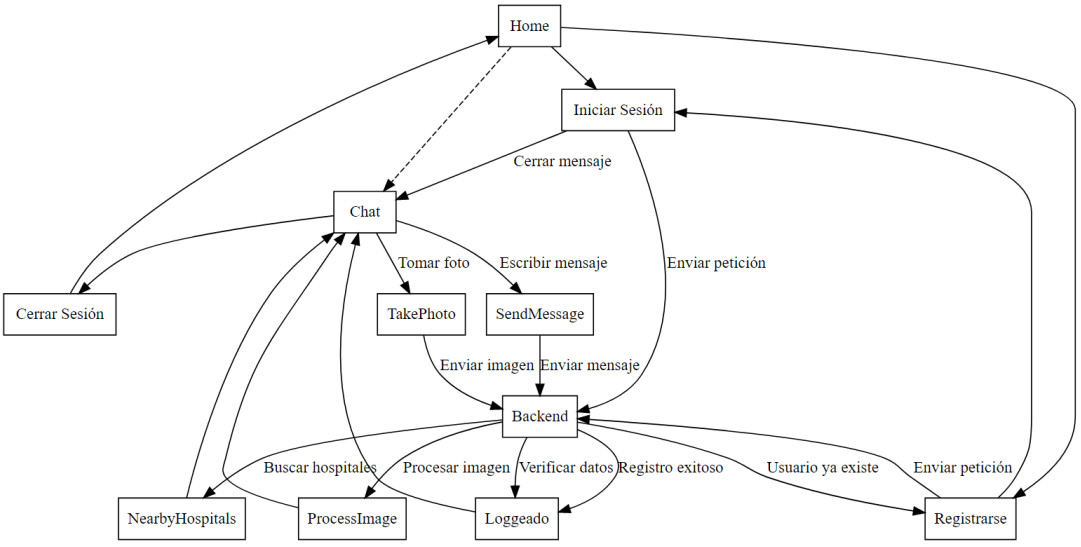


- When the user clicks the "Register" button, it sends a request to the backend to check if the username already exists.
- If the username already exists, a message is displayed asking the user to enter a different username.
- If the username doesn't exist, the user is registered in the database. After registration, the user needs to go to the login page to log in and access the chat zone.

In the chat zone, the user can write messages and take a photo of the affected body part. When the user clicks the "Send" button, the message is displayed in a WhatsApp-like format.

- The system sends a request to the backend with the image to process and identify the body part.
- At the same time, the system displays hospitals in the user's neighborhood along with their username.
- In the chat zone, there is also a "Log Out" button, which ends the session and redirects the user to the home screen

3.1 System Flow Chart



4. Experimental Results and Discussion

Taking into account the purpose for which this project was created, BODYSENSE is capable of meeting the expectations set forth.

- BODYSENSE manages to provide detailed information regarding the needs of each individual who decides to use our services from home.
- By registering in the program's database, individuals can enjoy the services offered by BODYSENSE, which range from recognizing the specific area of the body where the problem is located through a photo, if desired, to providing quick-action recommendations in case of an emergency, as well as information on the nearest medical center based on the individual's location.
- The use of advice and recommendations for personal care while heading to the medical center is also a standout feature of our program. This not only provides individuals with useful and practical information to manage their medical issue before receiving professional attention but also fosters a proactive approach to health and well-being. Users can take preventive measures and follow the provided instructions to minimize the impact of their condition until they receive appropriate care.

One of the main conclusions we can draw from this project is the importance of integrating advanced technologies, such as artificial intelligence and new programming languages, in the field of medicine. Additionally, we have observed that the availability of up-to-date and relevant medical information is crucial for the effectiveness of this type of software. By providing data on the nearest medical centers, the program allows users to make informed decisions about where to seek additional medical care if needed. This functionality is particularly valuable in urgent situations where every minute counts.

5. Conclusions and future work

In conclusion, the development of the chat-based software project with integrated AI has proven successful in providing users with preliminary medical information and facilitating access to nearby hospitals. By capturing and analyzing user-uploaded images, the system offers potential causes and diseases related to areas experiencing pain, assisting users in understanding their symptoms better. The software project's implementation has showcased the effectiveness of AI in healthcare applications, demonstrating its potential to enhance accessibility to medical resources. The user-friendly interface and personalized experience through registration contribute to a seamless interaction between users and the AI system.

Future enhancements to the software project may involve expanding the AI's diagnostic capabilities by incorporating additional medical databases and refining the algorithms used for image analysis. Further integration with electronic health record systems and telemedicine platforms can provide a more comprehensive and holistic healthcare solution.

Overall, this software project represents a significant step towards leveraging AI technology to improve healthcare accessibility and empower users to make informed decisions regarding their health. It demonstrates the potential for AI to augment and support healthcare professionals in providing timely and accurate diagnoses, ultimately leading to better patient outcomes.

Furthermore, we expanded our knowledge by exploring Angular, a widely used framework for frontend development. The use of Angular allowed us to create an attractive and user-friendly interface, providing users with an intuitive experience when interacting with our program. The combination of Python in the backend and Angular in the frontend enabled us to develop a comprehensive and efficient system to meet the needs of individuals requiring medical care from home.

In conclusion, we were able to build upon knowledge from previous semesters, strengthening it with the concepts learned this semester, such as the use of familiar programs but also including the programming language Python and artificial intelligence models.

6. References

MedlinePlus. (s.f.). First Aid. Recuperado el 30 de mayo de 2023, de https://medlineplus.gov/spanish/firstaid.html

Encolombia. (n.d.). Clinics in Armenia. Retrieved May 30, 2023, from https://encolombia.com/salud/dir-salud/clinicas/clinicasenarmenia/

Más que Médicos. (n.d.). Hospitals in Neiva. Retrieved May 30, 2023, from https://masquemedicos.co/hospitales_neiva/

Centros Médicos y Hospitales. (n.d.). Hospitals, Medical Centers, and Clinics in Neiva by Neighborhood. Retrieved May 30, 2023, from https://centrosmedicosyhospitales.com/colombia/Huila/neiva