

# CS 410 Project Documentation Healthcare Assistant Agent

This document has setup instructions, followed by an implementation overview.

## Data Setup

Full instructions can also be found at:

<https://github.com/justthea/CS410-Healthcare-Assistant-Chatbot/blob/main/healthcare-data/README.md>

### 1. Data (please see detailed instructions at [Link](#))

#### 1.1 Database Setup

##### a. Prerequisites

##### i. PostgreSQL 14 Installation:

##### 1. macOS(Using Homebrew):

- a. `brew install postgresql@14`
- b. `brew services start postgresql@14`

##### 2. Linux (Ubuntu/Debian):

- a. `sudo apt update`
- b. `sudo apt install postgresql-14 postgresql-contrib-14`
- c. `sudo systemctl start postgresql`

##### 3. Windows:

- a. Download PostgreSQL 14 installer from <https://www.postgresql.org/download/windows/>
- b. Run the installer and follow the setup wizard
- c. Remember the password you set for the postgres user

##### ii. PostgreSQL Vector Extension Setup

##### 1. macOS:

- a. `brew install postgresql@14`
- b. `psql postgres -c 'CREATE EXTENSION vector;`

##### 2. Linux:

- a. `sudo apt-get install postgresql-14-vector`

##### 3. Windows:

- a. Vector extension is included in PostgreSQL 14+ installation

#### 1.2 Database Creation (if needed)

##### b. Connect to PostgreSQL:

- i. `# macOS/Linux`
- ii. `psql postgres`

- iii. # Windows
- iv. "C:\Program Files\PostgreSQL\14\bin\psql.exe" -U postgres
- v. # Then create the database:
- vi. CREATE DATABASE medical\_db;
- vii. \q

## UI / Web Application Setup

Pre-requisites (Common for Windows/Mac/Linux)

1. Node.js Installation:
  - Ensure you have Node.js installed
  - Download Node.js from <https://nodejs.org/>.
2. Verify Installation:
  - Run the following commands in your terminal or command prompt to verify installation:
    - node -v
    - npm -v

Clone the GitHub repository and navigate to the project directory 'healthcare-ui'.

Steps for Windows/Mac/Linux

1. **Install Dependencies:**
  - Run the following command to install all the dependencies listed in package.json.  
**Command:** npm install
2. **Start the Development Server:**
  - Use the following command to start your React app  
**Command:** npm start
  - This will launch the app in your default browser at <http://localhost:3000>
3. **Open in Browser:**
  - If it doesn't open automatically, visit <http://localhost:3000> in your browser.

## Steps to Set Up the Healthcare LLM

1. **Obtain Gemini API Key**
  - Using a Google Account, create an API Key from <https://aistudio.google.com/app/apikey>
2. **Place API Key in Environment File**
  - Create a .env file in the healthcare-ui folder

- Use VIM or another text editor to include this line replacing your\_api\_key\_here with your API Key:
- REACT\_APP\_GEMINI\_API\_KEY=your\_api\_key\_here

## Common Commands for Troubleshooting

- **If npm install Fails:**
  - Delete node\_modules and package-lock.json, then reinstall.  
**Command for Mac/Linux:** `rm -rf node_modules package-lock.json`  
**Command for Windows:** `rd /s /q node_modules package-lock.json`  
**Then, run** `npm install`
- **Clearing Cache:**
  - Clear npm cache if issues persist  
**Command:** `npm cache clean --force`
- **Port Conflict:**
  - If port 3000 is in use, start the app on a different port.  
**Command for Mac/Linux:** `PORT=3001 npm start`  
**Command for Windows:** `set PORT=3001 && npm start`

# Implementation Overview

Our Healthcare Information Assistant project delivered an intelligent healthcare assistant capable of recommending over-the-counter medications based on user-reported symptoms. The chatbot leveraged large language models (LLMs) and public health data to provide personalized recommendations, demonstrating a functional and user-friendly integration of natural language processing with health informatics.

1. **Frontend Development:**
  - A fully functional healthcare-themed website was created with a professional design, featuring pages for Home, About, Services, Contact, and Appointments.
  - A chatbot interface was seamlessly integrated into the website, positioned as a widget in the bottom-right corner. Users could interact with the chatbot via a clean and responsive input box, simulating a realistic user experience.
2. **Data Integration:**
  - Extensive data was sourced from reliable public health databases, including the NIH, covering over-the-counter medication information, symptom descriptions, and their mappings.
  - The collected data was structured into organized datasets that categorized symptoms and medications and outlined usage guidelines.

- These datasets were converted into a vectorized format to enable efficient retrieval during LLM processing, ensuring real-time performance in responding to user queries.
3. **LLM Implementation:**
- The LLM was fine-tuned to process user symptoms provided in natural language, using advanced NLP techniques like entity recognition and contextual tokenization.
  - Integration with the vectorized data allowed the chatbot to recommend accurate and personalized over-the-counter medications.
  - Robust API-based communication was established between the frontend and the backend, enabling smooth interactions with the LLM.
4. **Evaluation and User Feedback:**
- A systematic evaluation framework was developed to measure the chatbot's effectiveness. Metrics included precision, recall, and user satisfaction, providing a comprehensive assessment of system performance.
  - After comparisons of multiple models, we eventually decided to use Gemini for its consistent accuracy
  - User feedback was derived from real-world interactions and sample user responses. This data informed iterative improvements to enhance user experience and recommendation accuracy.
  - As seen in the video presentation, accuracy for each query is output as well to measure effectiveness.

### **Challenges and Solutions:**

- Initial challenges in selecting an optimal vector database were overcome through team coordination and research into industry best practices.
- Sequential dependencies between frontend and backend tasks were managed efficiently to ensure timely integration.