

2024

# Capstone Final Report

Medical Chatbot for Enhanced Patient Data Insights

Ron Hankey

Springboard Data Science Cohort – Apr 2024

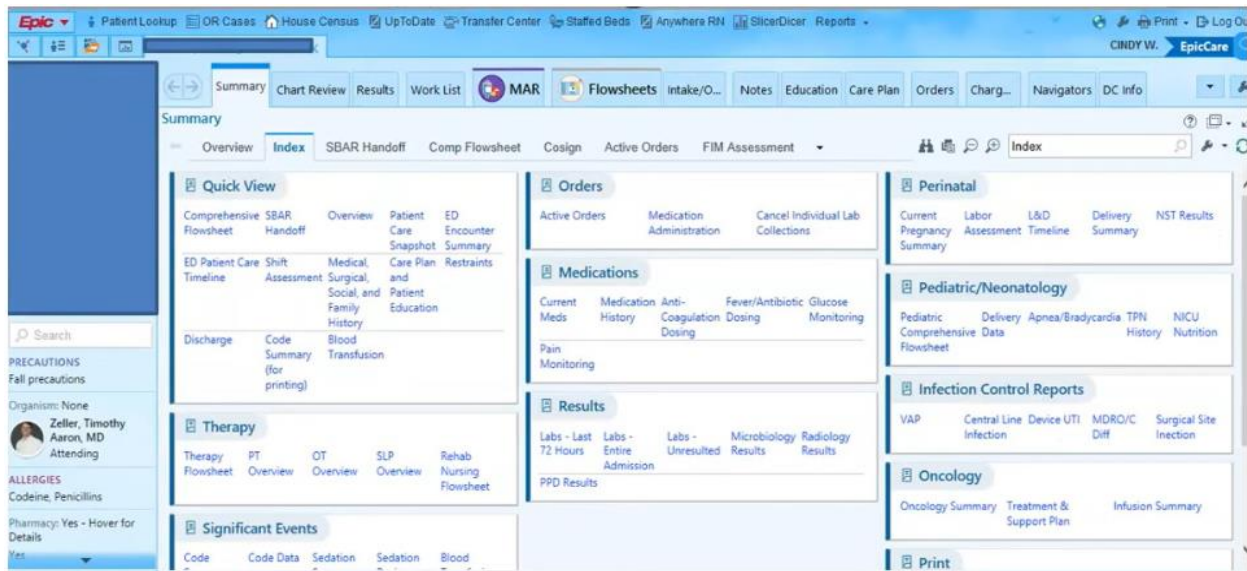
4/10/2024

## Contents

Problem identification.....	2
Data wrangling .....	4
Exploratory Data Analysis.....	7
Pre-processing and training data development.....	8
Modeling .....	9
Documentation .....	11
Summary .....	14

## Problem identification

Physicians, nurses, hospital administration and other patient caregivers need to review patient records in electronic health records (EHR). As the screen image from Epic, one of the most popular EHR's, there is an enormous amount of data on just the summary page. Then there are multiple tabs with additional information that is important for patient care.



The amount of time it just takes a care giver to review a patient chart has been studied and one paper published stated this:

"The amount of time that providers spend using electronic health records (EHRs) to support the care delivery process is a concern for the U.S. health care system. Given the potential effect on patient care and the high costs related to this time, particularly for medical specialists whose work is largely cognitive, these findings warrant more precise documentation of the time physicians invest in these clinically focused EHR functions. ... **Physicians spent an average of 16 minutes and 14 seconds per encounter using EHRs, with chart review (33%),**

documentation (24%), and ordering (17%) ... The proportion of time spent on various clinically focused functions was similar across specialties.” ([Physician Time Spent Using the Electronic Health Record During Outpatient Encounters: A Descriptive Study](#))

**The deliverable for this project is a Generative AI (GenAI) model utilizing retrieval augmented generation (RAG) to leverage Large Language Models (LLM's) in an attempt to reduce the amount of time physicians, nurses and other caregivers spend in chart review.**

## Data wrangling

The project needed data that included a combination of unstructured text for natural language processing (NLP) and structured data for database querying. The unstructured data is needed in order to show the power of the LLM and the structured data was needed to show how a LLM could build the SQL to query the data base.

Medical data consisting of patient records is extremely hard to find due to Medical data is extremely hard to find due to Health Insurance Portability and Accountability Act (HIPAA) privacy regulations. According the CDC's website: "The Health Insurance Portability and Accountability Act of 1996 (HIPAA) is a federal law that **required the creation of national standards to protect sensitive patient health information from being disclosed without the patient's consent or knowledge.**"

There were two data sets found on Kaggle that, while they were not ideal, they Offered a partial solution by combining the two data sets into one. These data sets were the Medical Records Dataset, this dataset contains simulated medical records for a fictional group of patients. The dataset was generated using the Python Faker library to create realistic but fake data ([Medical Records Dataset](#)).

The data set is described on Kaggle as having these columns:

Patient ID: A unique identifier for each patient (integer).

Name: A randomly generated full name (string).

Date of birth: A randomly generated date of birth with ages between 1 and 100 years old (date).

Gender: A randomly selected gender (M or F) (string).

Medical conditions: **A list of three random, unique words representing medical**

**conditions (string).**

Medications: **A list of three random, unique words representing medications (string).**

Allergies: **A list of three random, unique words representing allergies (string).**

Last appointment date: A randomly generated date within the range of the last 2 years (date).

The random words in the data set were not medical related, they were simply random words pulled from a dictionary so were not realistic in a medical sense.

There was still a need to a data set that provided unstructured text that corresponded to medical notes. That dataset also, found on Kaggle was named Medical Transcriptions. This dataset is described as follows on Kaggle: “Medical data is extremely hard to find due to HIPAA privacy regulations. This dataset offers a solution by providing medical transcription samples.” The dataset was de-identified so there was nothing to link any of the records to a specific patient. The data set consisted of the following columns:

**description** (Short description of transcription) (String)

**medical\_specialty** (Medical specialty classification of transcription) (String)

**sample\_name** (Transcription title) (String)

**transcription** (Sample medical transcriptions) (String)

**keywords** (Relevant keywords from transcription) (String)

Merging these two data sets would create a dataset for the needs of the generative AI model that was built for this project. From the Medical Records Dataset, the following columns were kept:

Patient ID: A unique identifier for each patient (integer).

Name: A randomly generated full name (string).

Date of birth: A randomly generated date of birth with ages between 1 and 100 years old (date).

Gender: A randomly selected gender (M or F) (string).

From the Medical Transcriptions dataset all of the columns were kept and a new medications and allergies column were added that contained no data from the Medical Records Dataset.

## Exploratory Data Analysis

The data was examined for concepts that could test the generative AI model. This was done by examining the medical concepts in the transcription column and the columns that would require calculation. The result was a series of questions that would be used to the model. The questions were as follows:

Which patients use cigarettes?

Which patients are over 50 years of age?

Which patients take percocet, search columns TRANSCRIPTION and MEDICATIONS?

Which patients have had a colonoscopy? Include patient ID and name?

Who is the oldest patient and how old are they?

Who is the youngest patient and what is their age?

Can you write one page with recommendations for a patient to help them lose weight.

Write a one page document describing how to lower blood sugar naturally.

List all the patient ID's and names in the database that have appointments today.

List all the patient ID's and names in the database that have appointments tomorrow.

A series of SQL statements were created that would then be used in testing to verify that generative AI model was producing the correct answer.



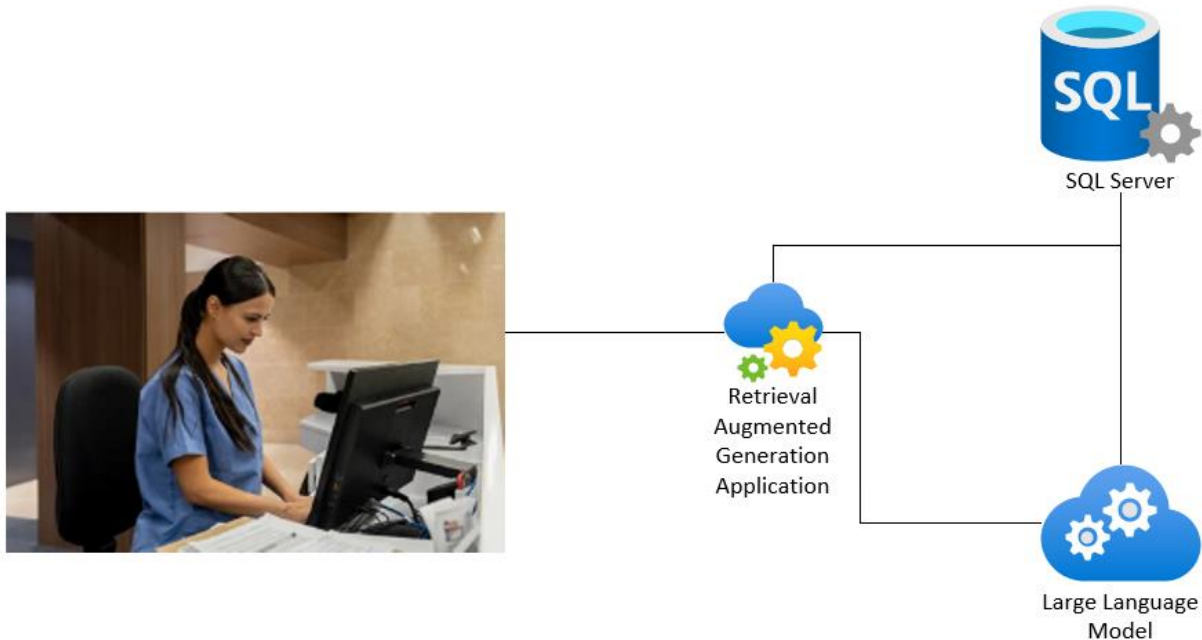
## Pre-processing and training data development

The following steps were taken to pre-process the data.

- 1) Randomly assign patients from the Medical Records Dataset to rows in the Medical Transcriptions based on descriptions in the transcription. This was done to align medical notes with the correct gender (M, F).
- 2) Next the first names also had to be adjusted to align with gender.
- 3) The clinical notes also did not align with gender since they were randomly assigned. Used a group of medical conditions that should only apply to either female or male to assign the correct note to the patient.
- 4) Then generated meaningful data for allergies and medicine. The data that was generated by python faker was just a jumble of words that had nothing to do with allergies or medicines.

Traditional ML models rely on supervised learning with labeled structured data and manual feature engineering, generative AI models using RAG utilize self-supervised learning, typically with unstructured data. The generative AI models are trained on vast amounts of unlabeled data, learning to generate data points that are similar to those observed during training. They don't require explicit labels but with RAG it references an authoritative knowledge base outside of its training data sources before generating response.

## Modeling



- **Generative AI (GenAI)**
  - A type of AI that can create new content and ideas, including conversations, stories, images, videos, and music
- **Large Language Model (LLM)**
  - Type of artificial intelligence (AI) program that can recognize and generate text, among other tasks.
- **Retrieval Augmented Generation (RAG)**
  - The process of optimizing the output of a large language model, so it references an authoritative knowledge base outside of its training data sources before generating response.

- **Langchain**
  - LLM orchestration framework that helps developers build generative AI applications or retrieval-augmented generation (RAG) workflows.
- **Streamlit**
  - Open-source Python framework for data scientists and AI/ML engineers to deliver dynamic data apps with only a few lines of code
- **Database**
  - Microsoft SQL Server but any relational database would work

## Documentation

The application is on my github repository at:

[https://github.com/lykkelig/Springboard\\_Final\\_Project](https://github.com/lykkelig/Springboard_Final_Project)

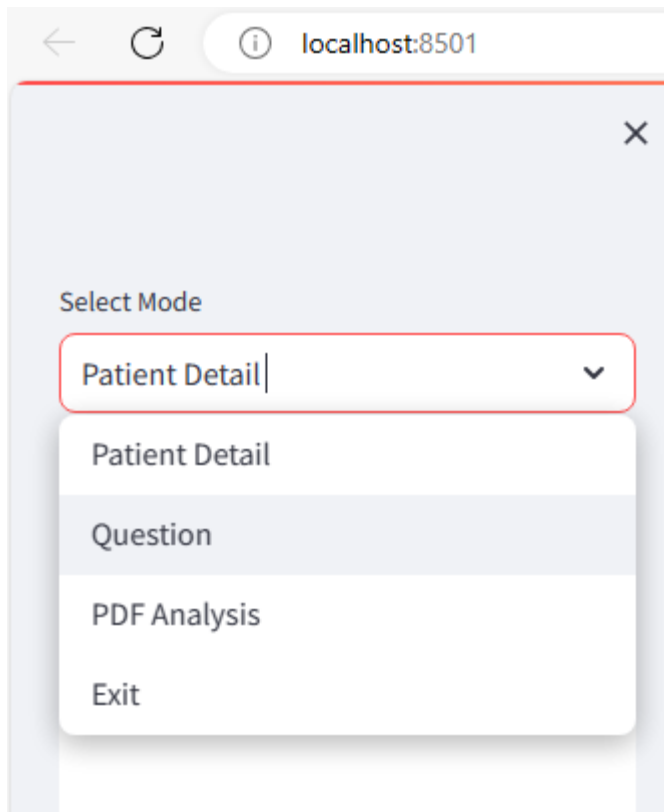
The application was developed using Visual Studio Code since the application was developed using Streamlit as Jupyter notebook does not support the execution of streamlit.

In order to run the application locally the following criteria need to be met:

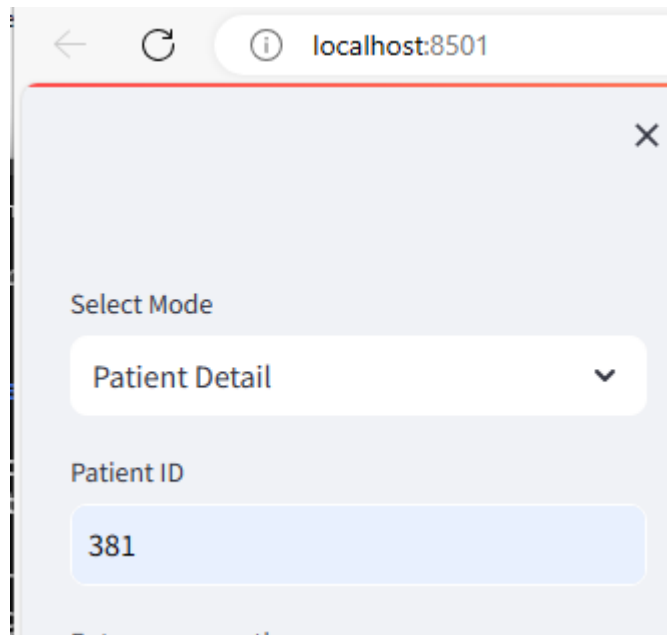
- 1) Install SQL Server Express on local PC, this is the free edition.
- 2) Create the MED\_DATA schema
- 3) Create the PATIENT and PATIENT\_NOTES tables using the DDL on the github.

The DDL for PATIENT is Create\_PATIENT\_Table and the DDL for the PATIENT\_NOTES is Create\_PATIENT\_NOTES\_Table .

- 4) Insert the data into the tables using Create\_Patient\_Records SQL
- 5) Run the Write\_DataBase\_Rows.ipynb program using the combined\_patient\_records\_20.xlsx file to add data to PATIENT\_NOTES
- 6) Create an account on OPENAI and generate an API key
- 7) Add the API key to the .env file
- 8) To run the application enter the command:  
  
streamlit run <directory>\GenAI\_Chat\_SQL\_Server.py

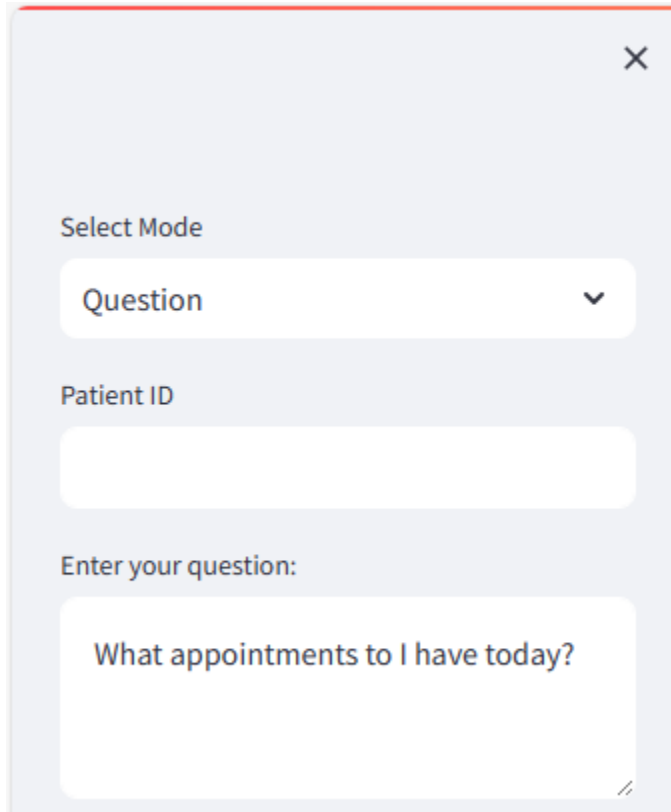


- 9) On the left hand side select Patient Detail and then enter the patient number you want to view.



- 10) To ask a question, change the "Select Mode" to "Question" and enter your

question and press “Submit”.



A screenshot of a mobile application interface. At the top right is a close button (X). Below it is a section titled "Select Mode" with a dropdown menu currently showing "Question". Underneath is a "Patient ID" label followed by an empty text input field. Below that is the label "Enter your question:" followed by a larger text input area containing the text "What appointments to I have today?". A small double-slash icon is visible at the bottom right of the text input area.

- 11) Use “Select Mode” to and change to “Exit” then click the “Submit” button to exit the application.

## Summary

The "Medical Chatbot" project aims to address a critical issue in healthcare: the significant time and effort required by medical professionals to navigate and utilize Electronic Health Records (EHR). These systems, although crucial for modern healthcare, often involve complex interfaces that demand considerable time from caregivers, with studies showing an average of over 16 minutes spent per patient encounter.

To combat this inefficiency, the project proposes the development of a Generative AI (GenAI) model. This model employs Retrieval Augmented Generation (RAG) and leverages Large Language Models (LLMs) to streamline the process of reviewing patient charts. The goal is to significantly reduce the time spent by physicians, nurses, and other caregivers on tasks such as chart review.

The data for this project was challenging to source due to stringent HIPAA regulations which protect patient information. Eventually, two datasets from Kaggle were utilized: the Medical Records Dataset, which consists of simulated medical records created using the Python Faker library, and the Medical Transcriptions dataset, which provides de-identified medical transcription samples. These datasets were merged to create a comprehensive set that includes both structured and unstructured data. This hybrid dataset supports both database querying and the natural language processing capabilities of the LLM, enabling realistic interaction with the chatbot.

The Generative AI model built into this chatbot utilizes technologies such as Streamlit for app development and Microsoft SQL Server for backend database management. The LLM's output is enhanced by the RAG process, which ensures that responses are not only generated based on the internal knowledge of the model but also checked against an authoritative external database.

The exploratory data analysis involved creating queries that the AI model could potentially answer, such as identifying patients with specific conditions or habits, like smoking or recent medical procedures. The model's effectiveness in generating accurate responses was verified using a series of SQL statements.

Overall, this project aims to optimize the interaction between medical professionals and EHR systems, thereby improving efficiency in patient care and reducing the cognitive load on healthcare providers. This is achieved through a sophisticated blend of AI technologies that automate and assist in the data-heavy tasks of medical record review.