**Chat with your PDF - Gen AI App - With Amazon Bedrock, RAG, S3, Langchain and Streamlit**

we will build a CHATBOT like application with AWS Amazon Bedrock, docker, python, Langchain, and Streamlit. We will use Retrieval-Augmented generation concept to provide context to the Large Language model along with user query to generate response from our Knowledgebase

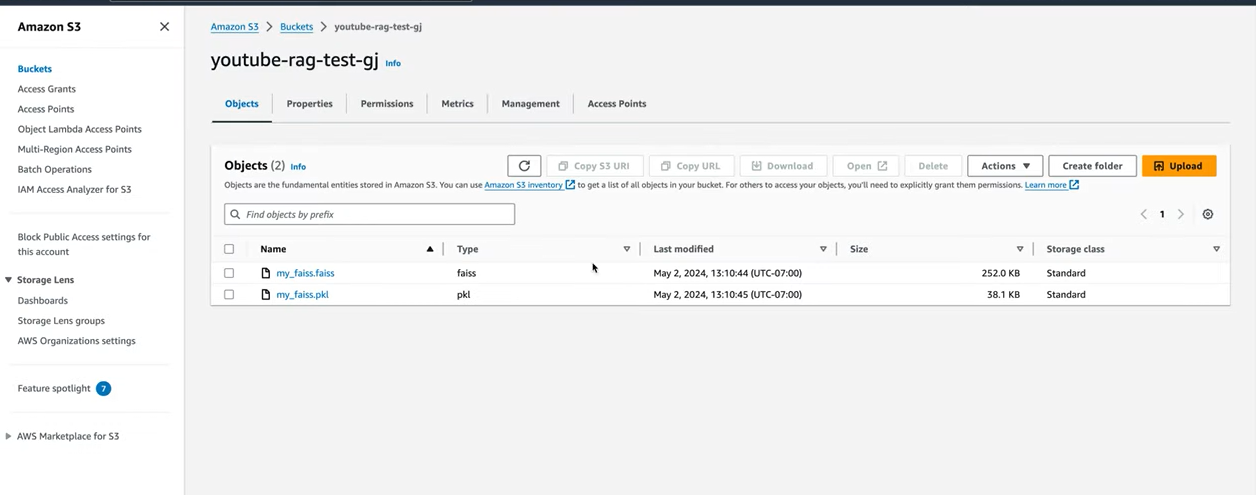
I will demonstrate the following: -

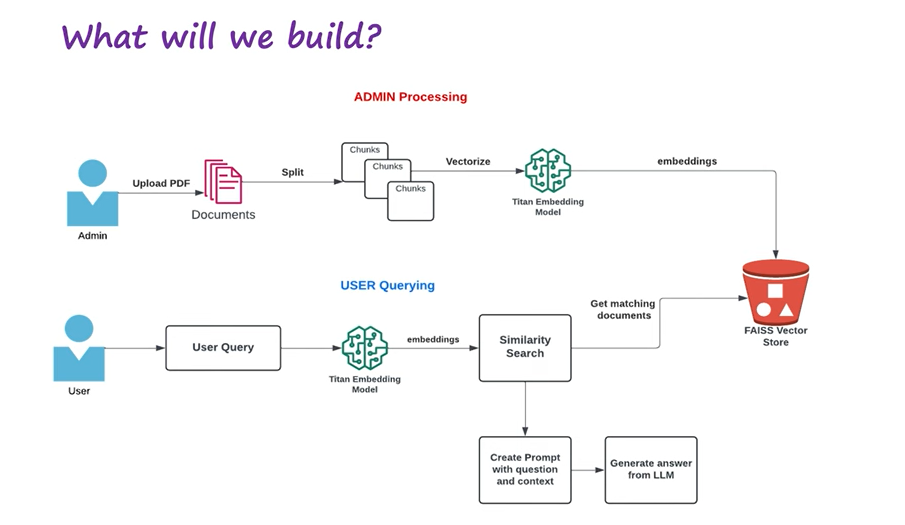
Architecture of the applications - Build 2 applications (ADMIN and USER) and create DOCKER images –

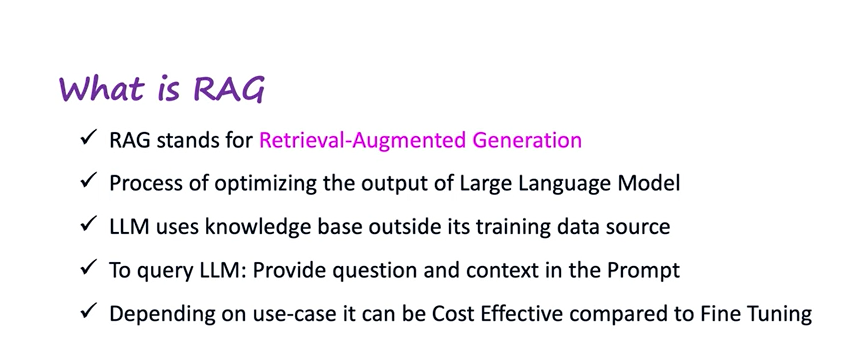
ADMIN Application: - Build Admin Web application where AdminUser can upload the pdf. - The PDF text is split into chunks - Using the Amazon Titan Embedding Model, create the vector representation of the chunks - Using FAISS, save the vector index locally - Upload the index to Amazon S3 bucket (You can use other vector stores like OpenSearch, Pinecone, PgVector etc., but for this demo, I chose cost effective S3) –

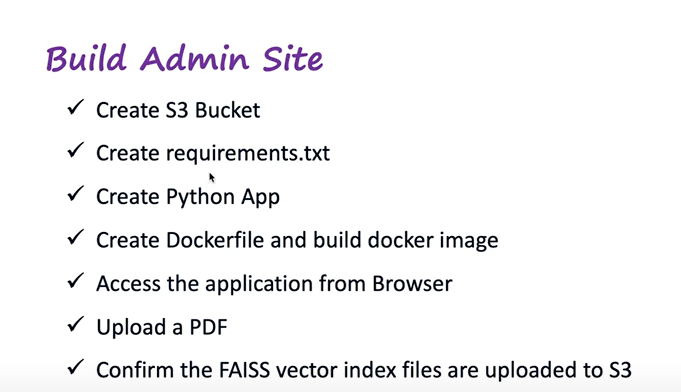
USER Application: - Build User Web application where users can query / chat with the pdf. - At the application start, download the index files from S3 to build local FAISS index (vector store) - Langchain's RetrievalQA, does the following: - Convert the User's query to vector embedding using Amazon Titan Embedding Model (Make sure to use the same model that was used for creating the chunk's embedding on the Admin side) - Do similarity search to the FAISS index and retrieve 5 relevant documents pertaining to the user query to build the context - Using Prompt template, provide the question and context to the Large Language Model. We are using Claude model from Anthropic. - Display the LLM's response to the user.

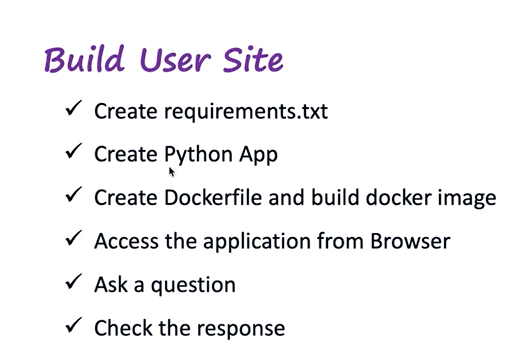
# PDF .. Loaded in this folder- <https://www.versusarthritis.org/media/22726/what-is-arthritis-information-booklet.pdf>



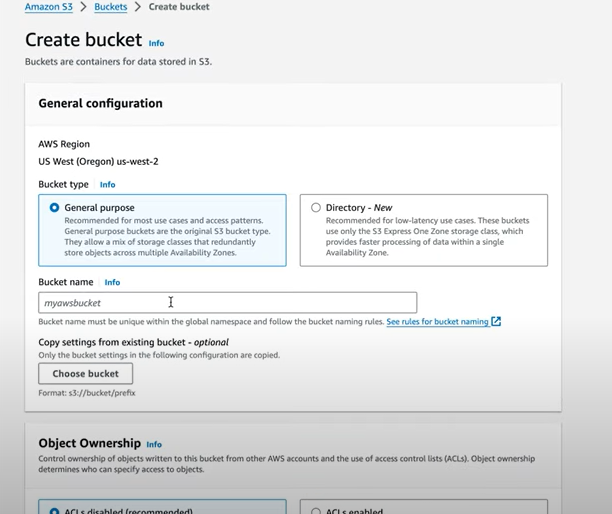


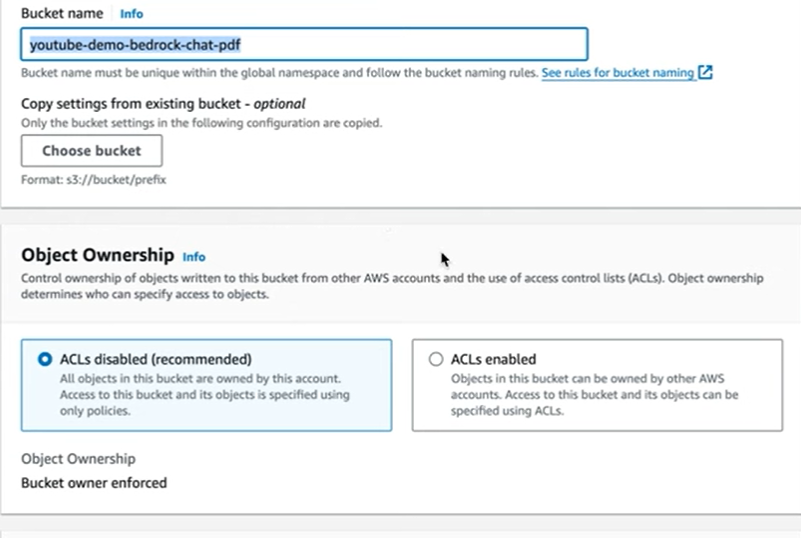


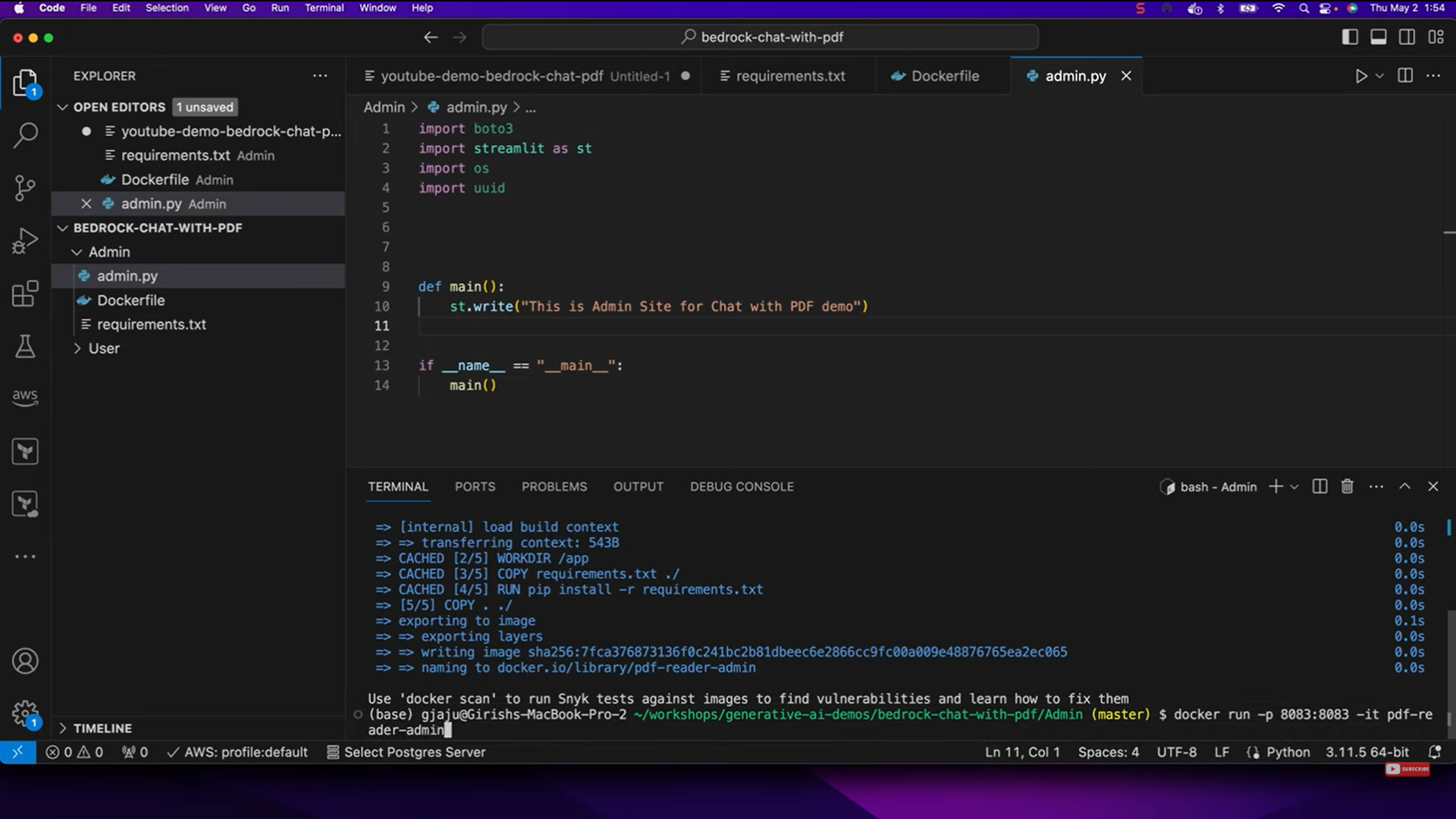




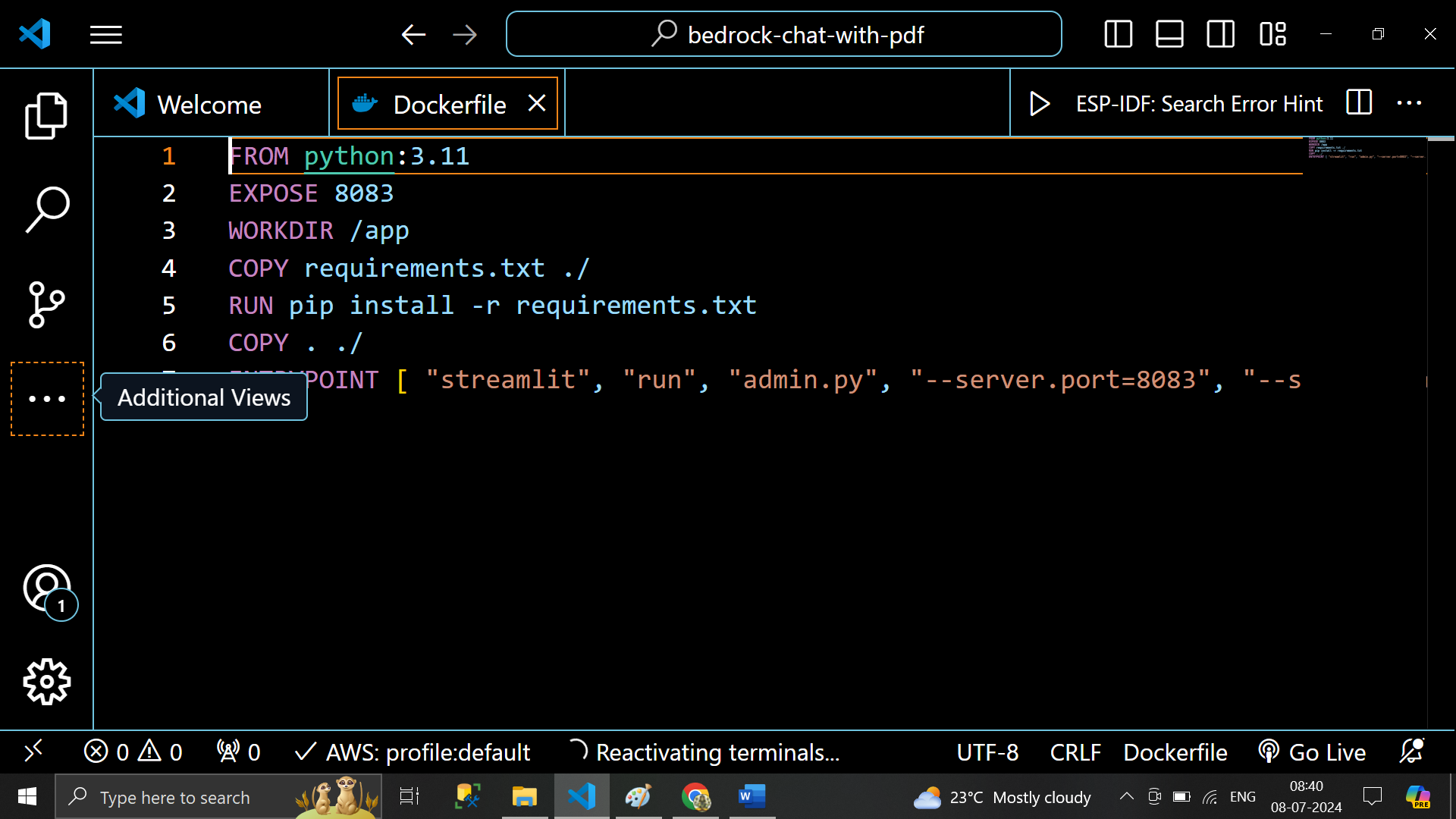
# Go to aws console - s3- create S3 bucket

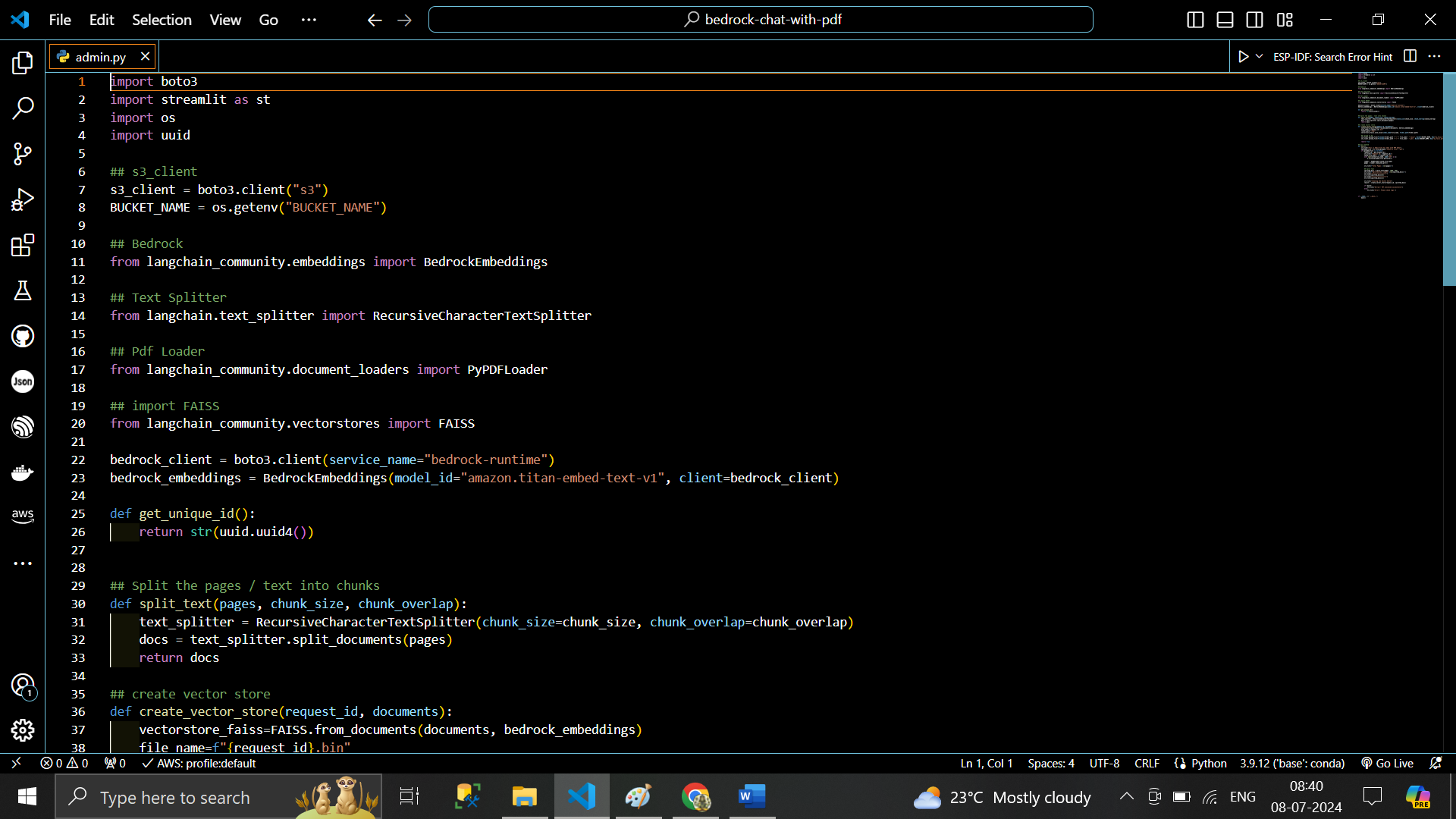




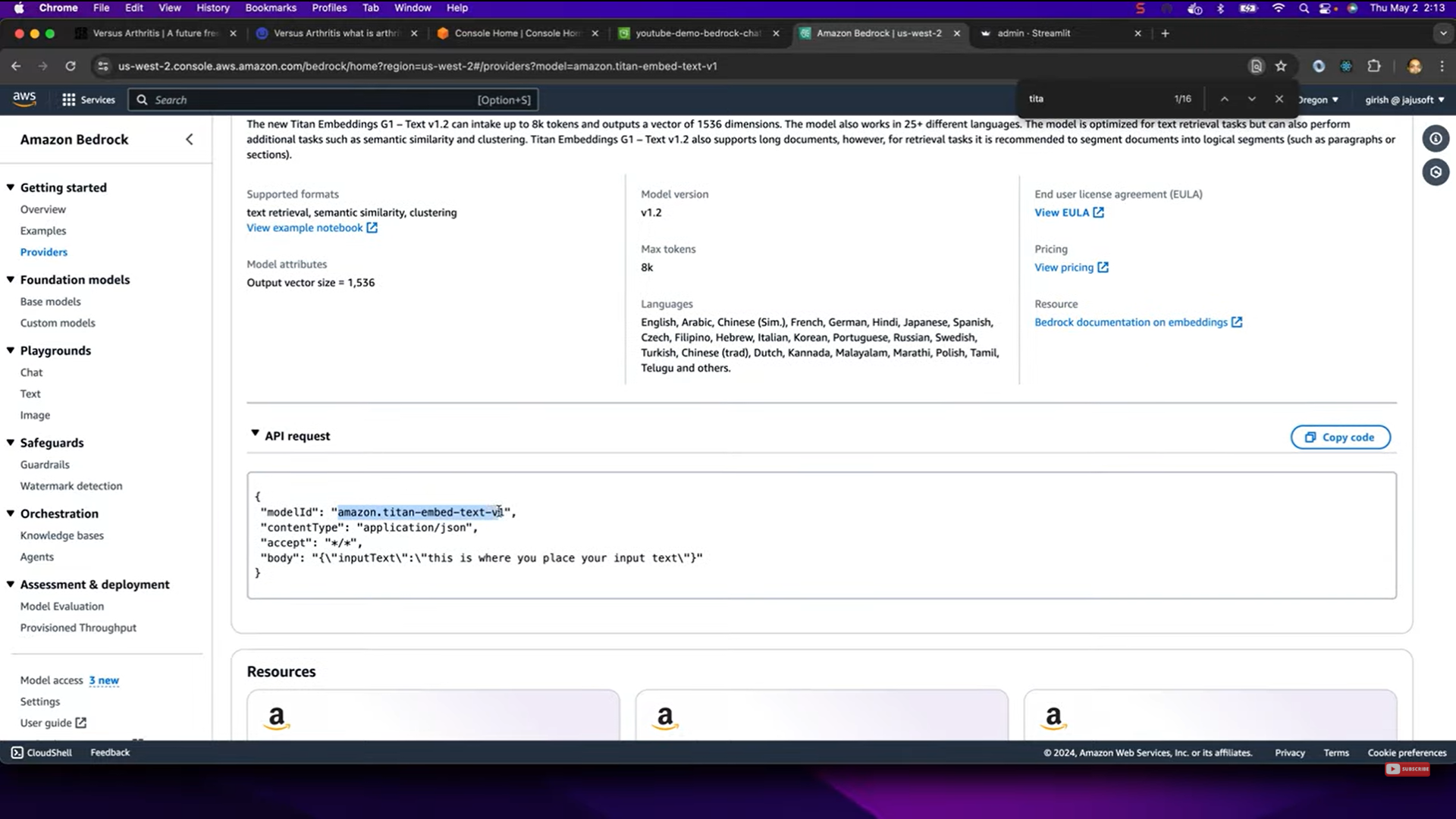


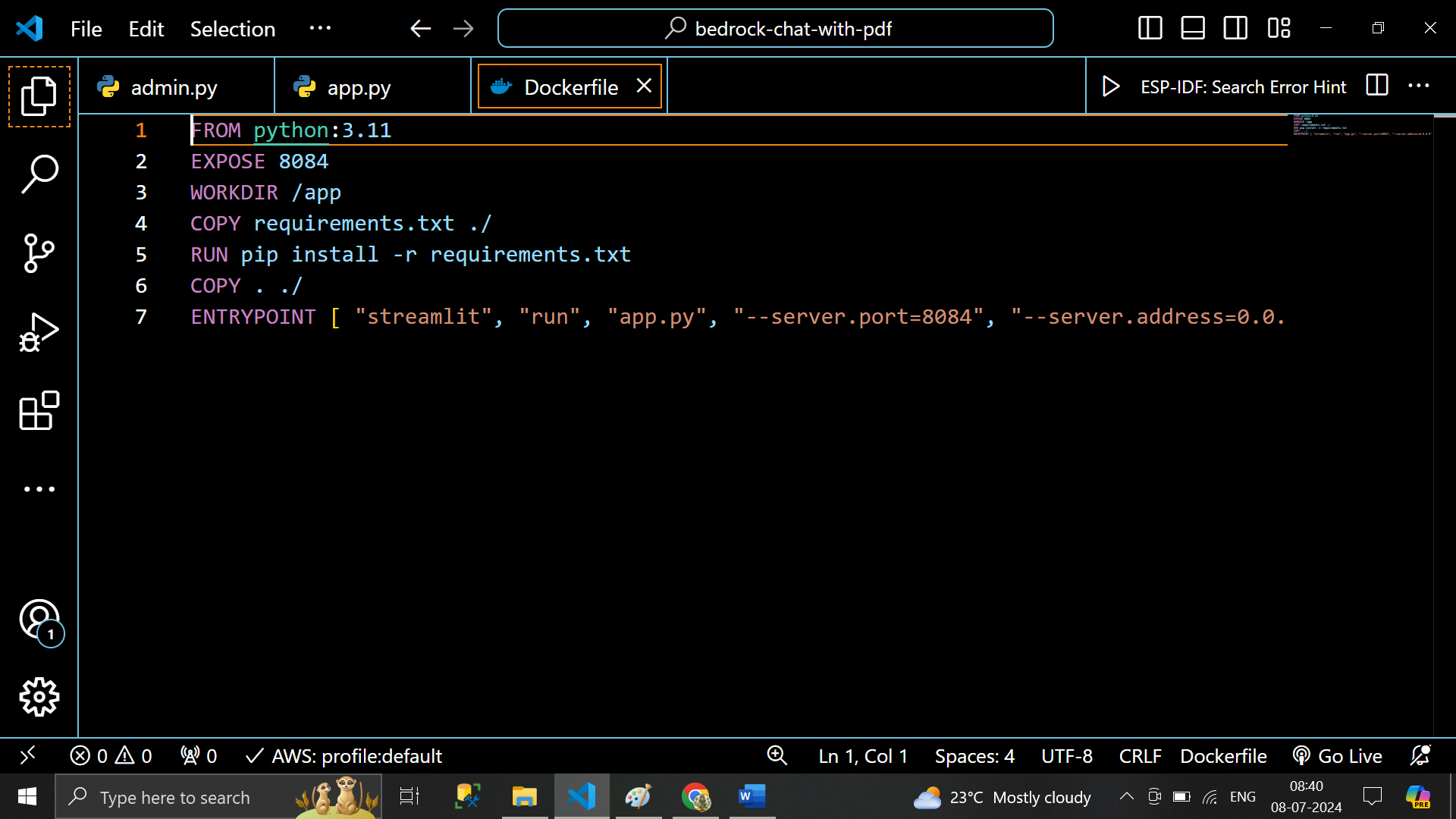
# docker file

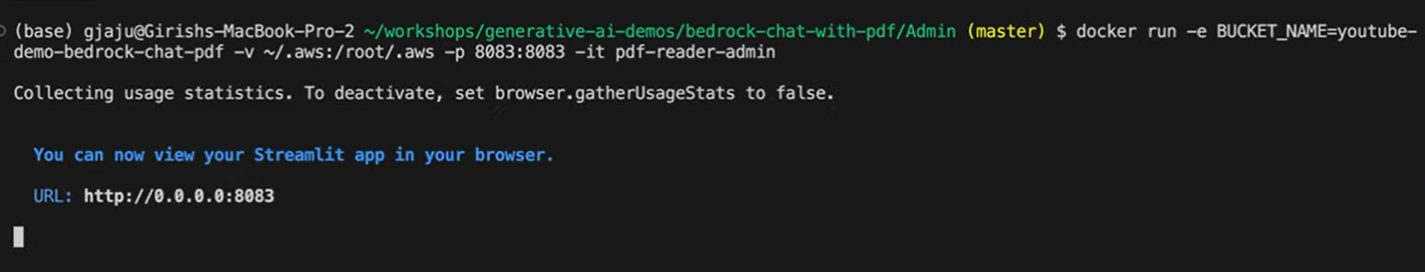




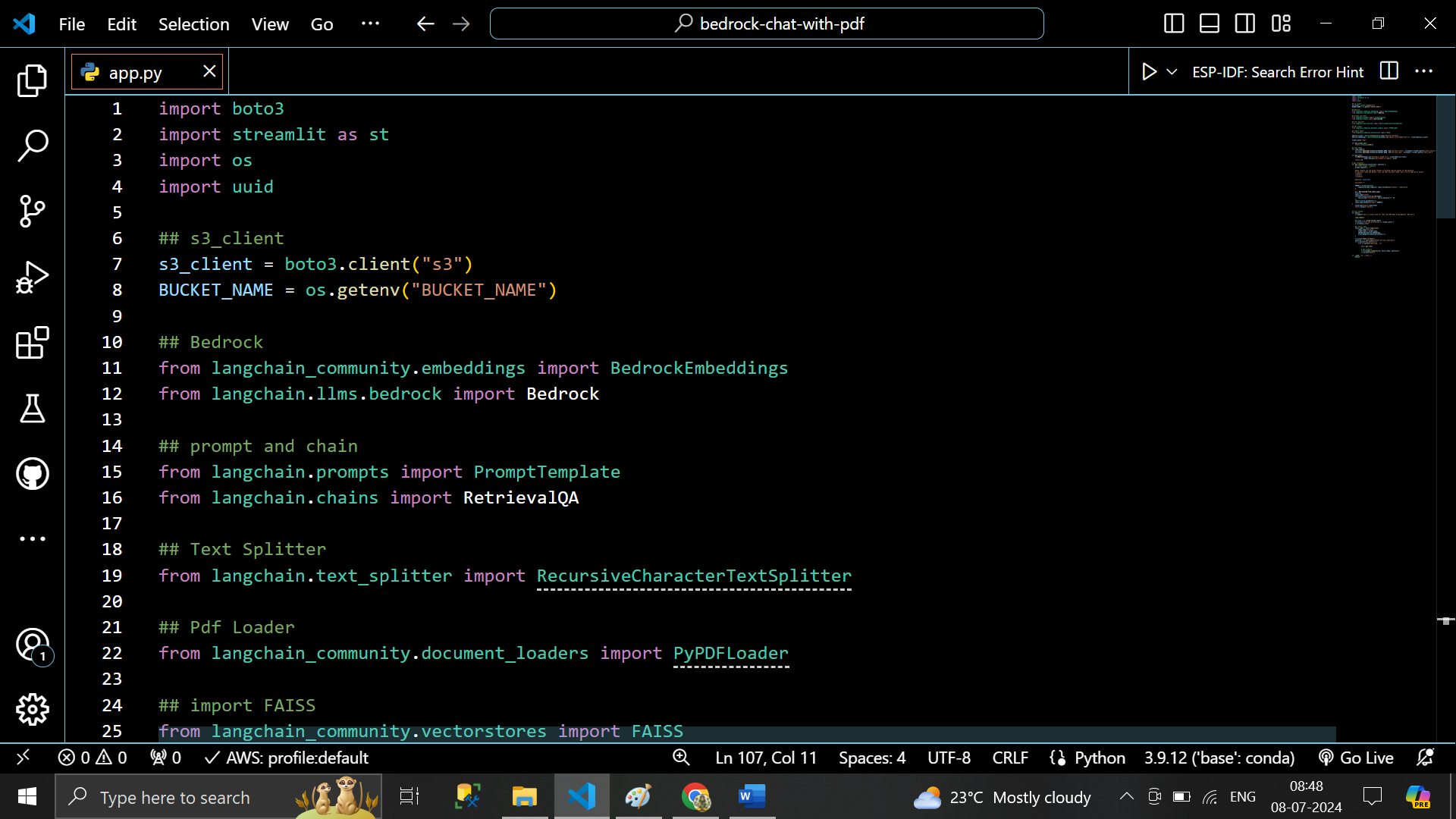
# Admin -side .. AWS Bedrock

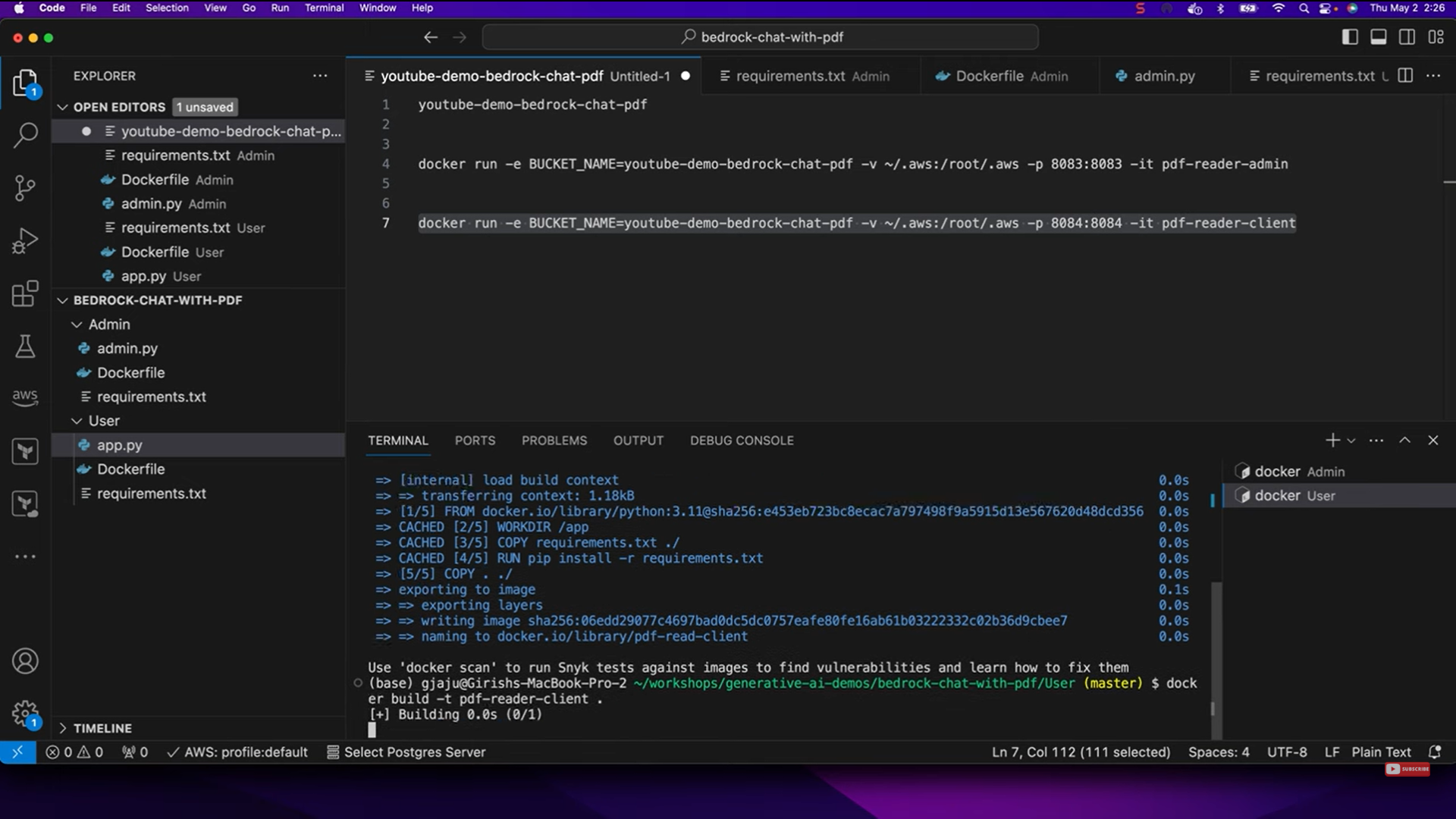


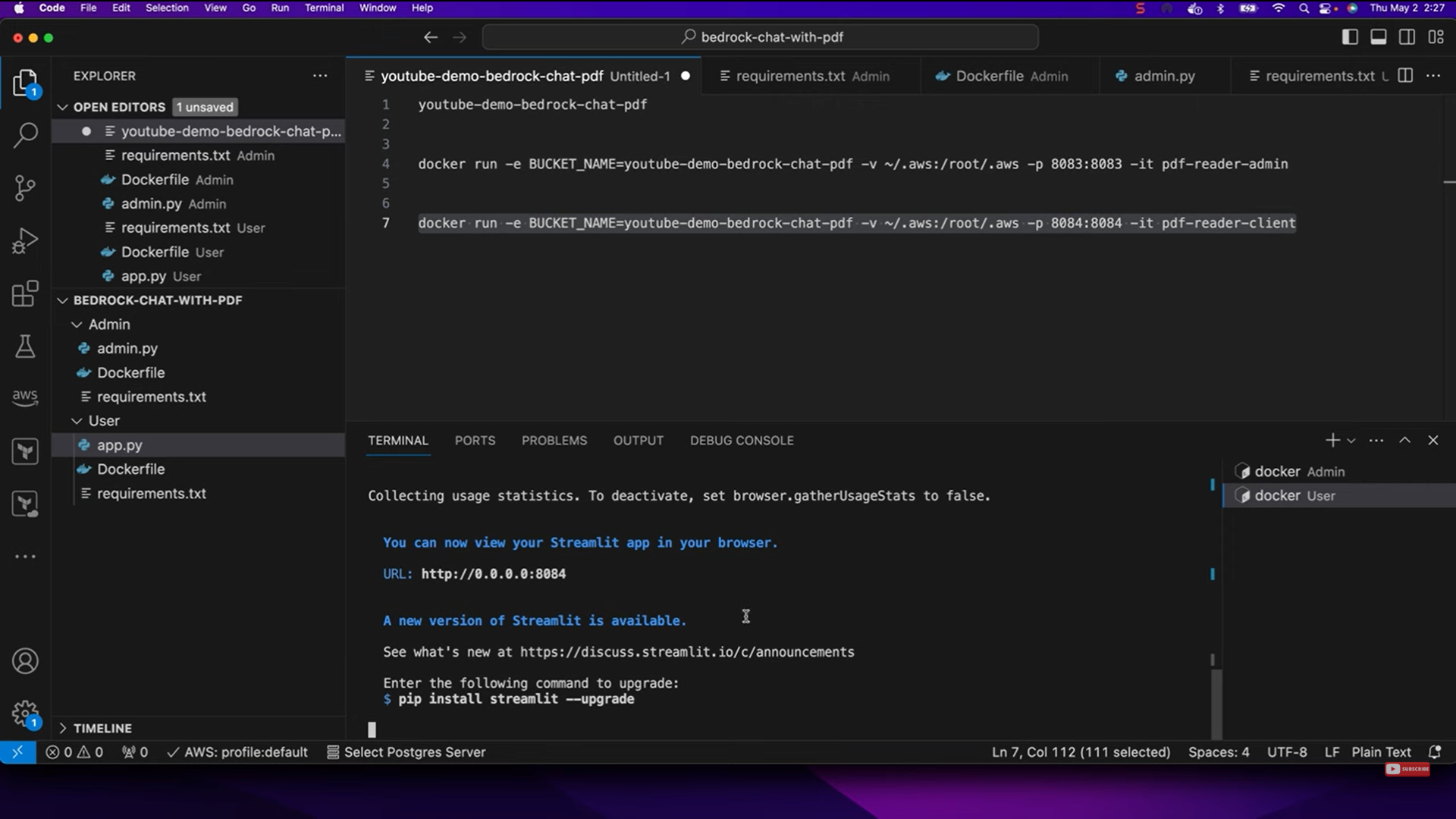




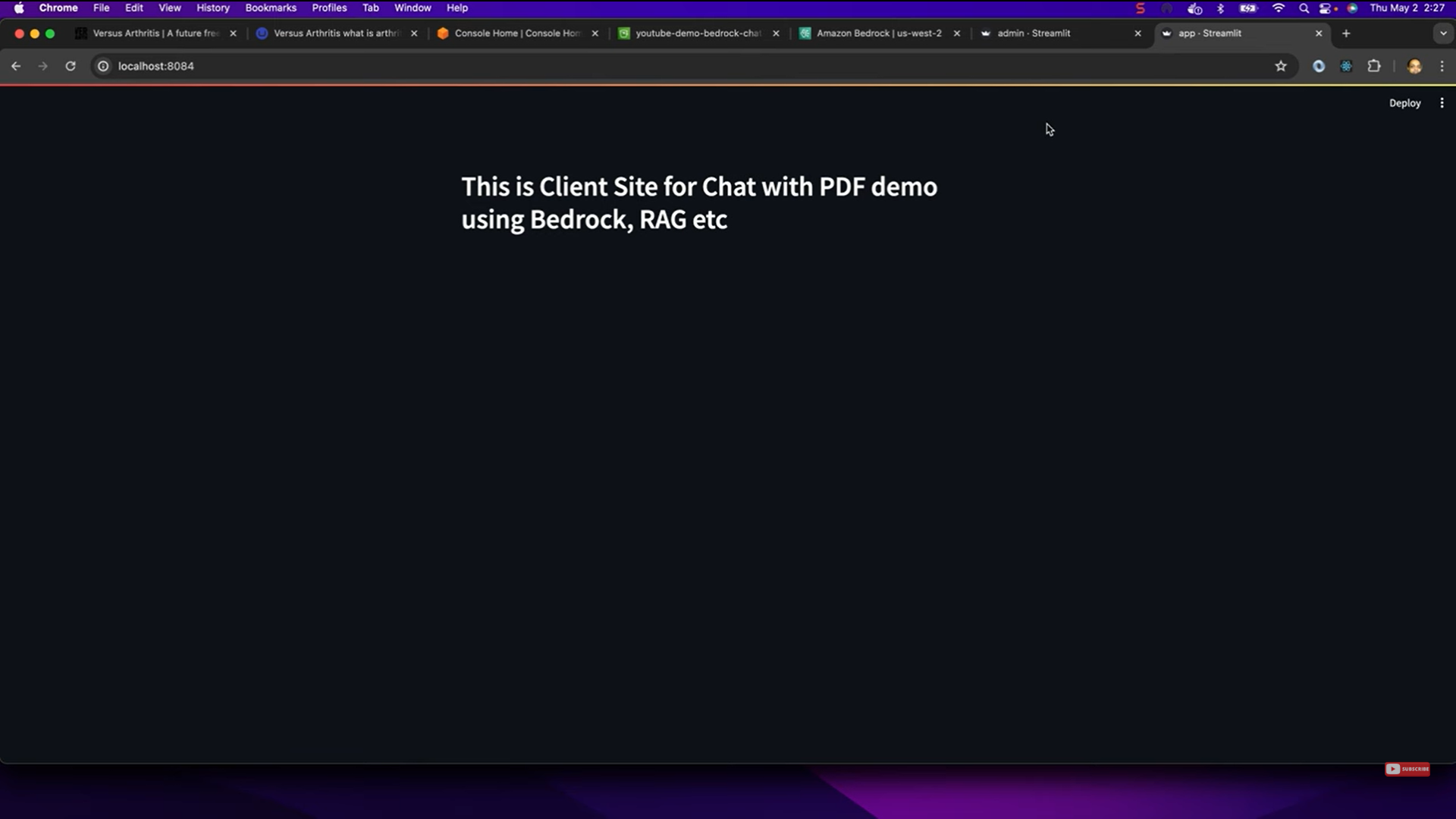
User- Data- app.py file



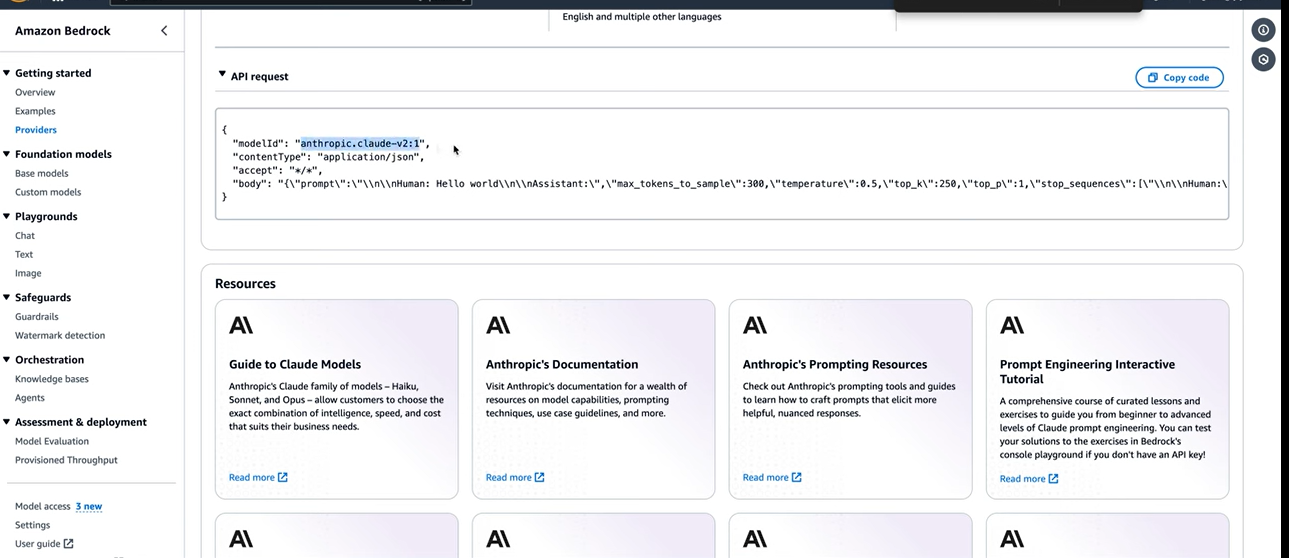


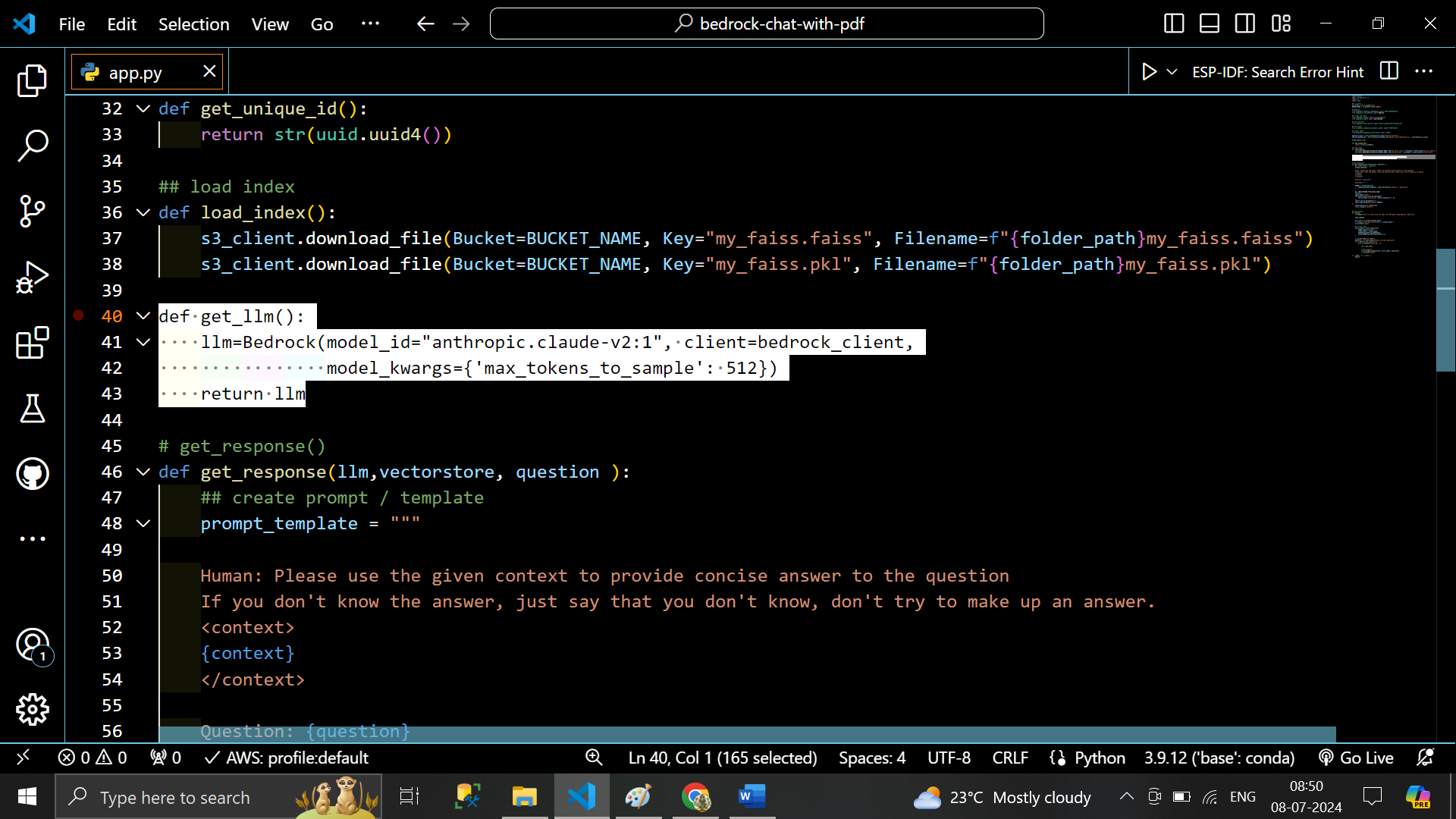


# Loading app

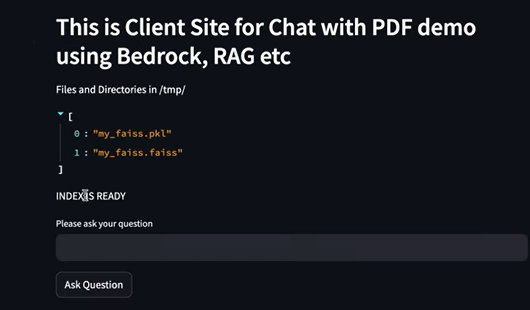


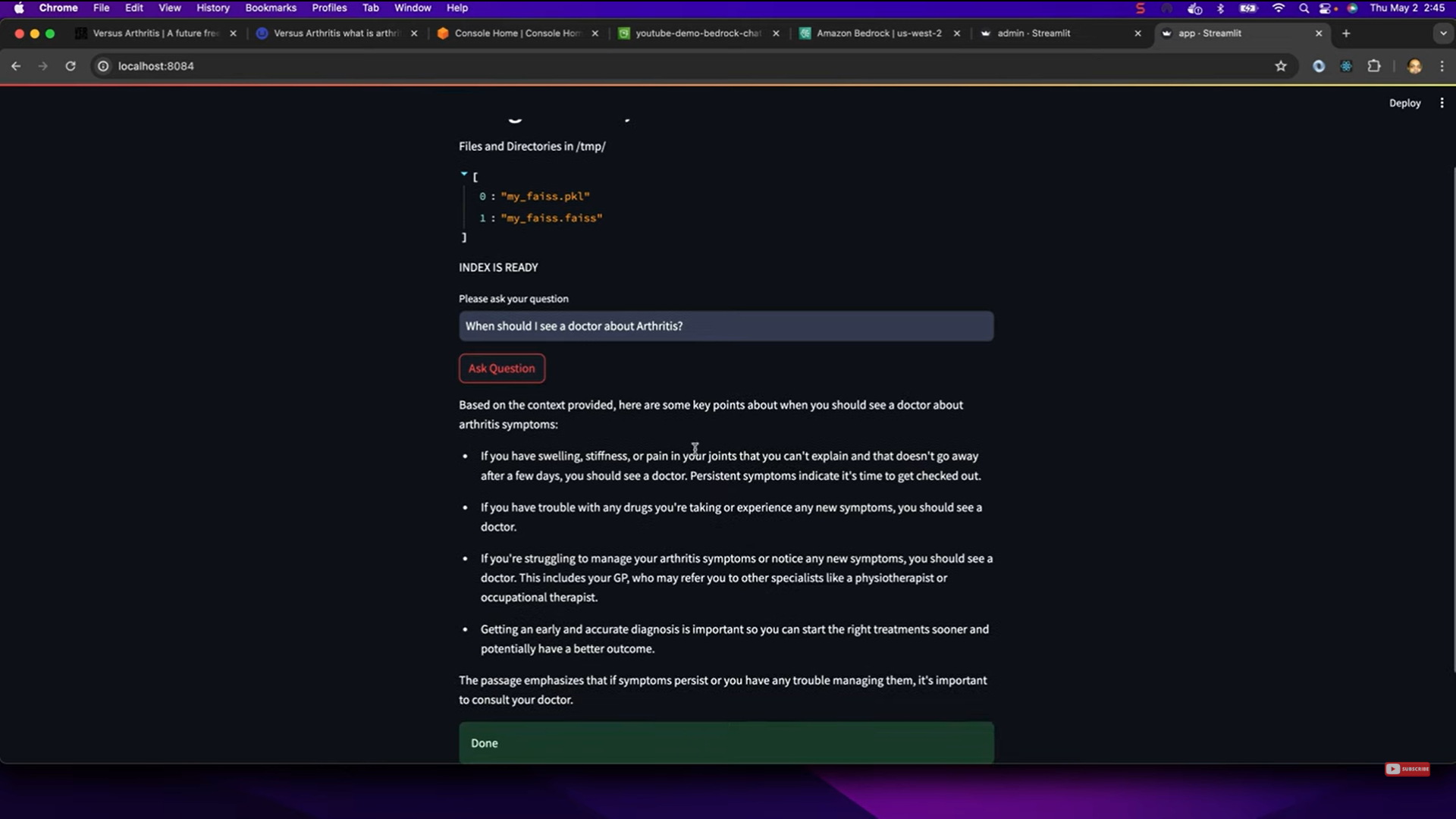
# Go to bedrock base Models





# Output .. index it ready





# Ask question Outside PDF File

