For this project, I developed a policy chatbot using six official documents from the U.S. Department of Commerce (DOC). These documents cover topics ranging from enterprise strategy, data privacy, cybersecurity, to data compliance. The chatbot was designed to allow users to ask questions about these policies and receive relevant responses.

My approach was inspired by Code With Aarohi's YouTube channel, which helped me understand how to structure a chatbot using document input and a retrieval-augmented generation (RAG) approach. From there, I experimented with different methods for loading documents, selecting an LLM model, and deploying a simple UI to make the chatbot functional and user-friendly.

One of the first challenges I faced was loading policy documents directly from online PDF URLs. I initially followed Aarohi's approach, using the Unstructured package, but it didn't work as expected. Therefore I switched to PyPDFLoader from the LangChain family, which successfully extracted content from the PDFs.

At first, I used Hugging Face models because they were straightforward to use and didn't require API key management. However, these models failed to properly answer questions, often just pulling relevant text instead of generating meaningful responses. I then switched to OpenAI's API, which provided much better responses by actually understanding and summarizing the content.

Using OpenAI meant I had to manage API key confidentiality. Initially, I didn't know the best way to do this, but I eventually decided to create a .env file to store the API key securely and prevent it from being uploaded to GitHub. I also updated .gitignore to ensure .env is never included in public repositories.

For the user interface, I used Gradio, which made the chatbot much easier to interact with. However, I ran into issues when trying to launch it from the terminal. It took me a while to realize that I forgot to set the website to public (share=True), which was preventing it from opening correctly. Once I fixed that, Gradio worked perfectly.

For future improvements, the first priority is to expand the input data to generate a more comprehensive understanding of the policy area. Currently, I manually entered links for six policy documents, but this approach isn't scalable for a larger dataset. To make input handling more efficient and automated, I need to explore methods such as web scraping or API integration to fetch policy documents from government repositories. Another key improvement is to experiment with different LLM models. While OpenAI's API provides high-quality responses, open-source models (e.g., Mistral-7B, Llama 3, Falcon-7B) could be a viable alternative to reduce API costs and offer more control over fine-tuning. Testing different models would allow me to compare accuracy, performance, and cost-efficiency, ensuring the chatbot remains effective while being financially sustainable.

Creating this chatbot was an incredibly fun and rewarding experiment. I felt a deep sense of fulfillment and accomplishment after seeing the final product being created. More than just a technical project, this experience reaffirmed my excitement for the work that Statt has been doing—not just conceptually, but also from a hands-on, technical perspective.