TO: SalesPatriot Team
FROM: Aayush Palai, Aayan Ilyas, Defne Balkan, Ella Jeon, Ivan Martynenkov, Matthew Tybur, Nitesh Kamanuru Purushotham, and William Tanudyaya
TRACK: B
DATE: 06/27/25

Problem Scope

For Sales Patriot, an AI platform managing and automating the complex federal contracting space, we must offer value beyond just accurate responses to best serve customers. Aspects of Sales Patriot's product such as latency are highly important for companies wanting to obtain highly competitive defense contracts. These companies must act quicker than competitors also seeking the same contract. Sales Patriot's AI integration must be fast enough to deliver results. If not, clients will see less inherent value in the service because they are less likely to gain more valuable contracts.

Transparency is also essential to Sales Patriot's product, as it builds trust with customers. The contracting industry is legally complex, and therefore, providing clear visibility into the logic behind SalesPatriot's AI recommendations is critical for client legal teams to proceed with confidence. Without the ability to trace the logic behind AI output, SalesPatriot clients risk having legal issues, acting on unsound information, and being inefficient.

In summary, latency and transparency must be prioritized within Sales Patriot's AI algorithm to ensure it not only accelerates clients' ability to act on contract opportunities but also empowers them with the clarity and confidence needed to make informed, legally sound decisions in a high-stakes, high-regulation environment.

Section B: What You Did

We selected Track B, a simulated UI where users can query ChatGPT and see the sources pulled to formulate the models response. We thought this was the best course of action to expand on our research from the previous week because it built on our artificial intelligence model programming, adding additional features like web session handling, data parsing and retrieval-augmented generation (RAG). We started by using code examples as a starting point, incorporating Python's langchain package and OpenAI. The reason we used Python / LangChain was because of its simplicity. We could load a model in a few lines and do meaningful things with it in just a few more. Because of Python's inherent simplicity, our development process was rapidly accelerated. Then, we created a session handler in order to ensure all PDFs were assigned with their chats. In order to actually set up the chats on the backend, we first used PyPDF2 to parse the PDFs, then broke them up / tokenized using lang chain-splitters. Finally, we put those into a vector store in order for the model to easily interpret them, then put it in a RAG chain, where relevant context was fed for the model. A problem we ran into and had to address while developing was ensuring the PDFs were assigned to their chats so they could be quoted. In

order to solve this, we created a robust session handler that effectively organized the chats and placed them in sessions, each with their own id.

Section C: Results or Insights

Much on the contrary to its requirement for needing to provide a source, one setback was that the model was unable to not provide a source. For example, when asked a basic addition question, the bot claimed the source for that answer was able to be found in page two of the document (which it was not). This could potentially be an example of hallucination; however, this issue likely arose from our own lack of expertise when coding. Putting the source issue aside, the model was able to accurately interpret bad questions and respond with quotes, as you can see in this video:
https://www.loom.com/share/da6e7087f23b4dea8ec836152cb75f8a?sid=48d6db51-3293-42ad-bcc8-2348142592be.
Overall, we were pleasantly surprised by the accuracy of the model, even when interpreting more tricky file formats like PDFs.

Section D: Recommendations for SalesPatriot

In order to ensure accurate sources (which is a major portion of any recommendation engine) we recommend that SalesPatriot extends the human feedback portion of their application to source tracking in a similar way to how they claim to do it already. With more feedback, the model can be refined to a point where sources are nearly always accurate, which is important for trust-building and, more importantly, transparency when making recommendations.
Additionally, given the accuracy of the solution provided taking out of consideration the sources, SalesPatriot shouldn't have to worry about actual response accuracy. A quick win here would be displaying sources from recommendations and maybe even an overview of why users were matched (which can greatly boost transparency and user confidence).

Section E: Limitations + Next Steps

As mentioned above, the bot was unable to provide accurate sources, which is a huge limitation. However, we are almost certain that this is because of our own errors when coding the project. (Personally, I suspect the chunking of data - the quotes were always from page two) The next steps to expand on this project would be to 1) improve source accuracy 2) integrate human feedback into the model, and 3) ensure consistency across all formats. Additionally, their own in-house model already trained on user feedback would have to be integrated into langchain (likely via HuggingFace) in order for LangChain to work with it.