1. Requirements
   1. Model implementation
      1. One additional tech
      2. Two tech from class. One new not from class
   2. Model or Application optimization
      1. Process that shows iterative changes in the model
      2. Sean –
      3. Gemini pro to evaluate???
   3. GitHub
   4. Free for all, just make it good!!!
   5. Firas and Sean reviewed and they said we were good.

**First Rough Draft of Project Proposal**

1. Overall topic
   1. Title: Automated graphing and data processing of any data sets using human prompts and voice input.
2. High Level Description
   1. Develop an application that will automate the process of doing EDA, especially custom graphing based on human prompts and voice inputs.
3. Technologies
   1. Streamlit is a web framework, tailored toward data science that makes developing web applications accessible to Python developers. We will use Streamlit to make our application web based.
   2. Langchain and Langraph – Both Langchain and Langgraph will be used as routing mechanisms between Python and a Large Language Model (LLM). The LLM will generate custom python code, which will then be parsed in Python and added to graphing files. Streamlit will then display the graphing files. Any type of graphs which are supported in MatPlotLib and possibly others.

**Title:** **Automated Graphing and Data Processing through Voice Commands and Human Prompts**

**Overview:** This proposal outlines the development of an innovative application designed to revolutionize Exploratory Data Analysis (EDA) by automating custom graphing processes of any dataset provided. Utilizing human prompts and voice inputs, this application aims to streamline the creation and manipulation of data visualizations.

**Detailed Description:** The primary objective is to create a user-friendly, web-based application that facilitates automatic generation of custom graphs for any given tabular dataset through simple voice commands or text prompts. This technology will enable users to engage with data analysis in a more intuitive and efficient manner, significantly reducing the manual effort involved in data processing.

**Technologies:**

1. **Streamlit:** This web framework is specifically tailored for data science applications, allowing Python developers to easily build and deploy web applications. Streamlit will serve as the foundation for our web-based interface, ensuring a seamless user experience.
2. **Langchain and Langraph:** These tools will be employed to establish a robust communication link between Python and a Large Language Model (LLM). The LLM will be responsible for generating custom Python code based on user inputs, which will then be executed to create graphs.
3. **Python and Matplotlib:** The Python code generated by the LLM will leverage Matplotlib, among other potential libraries, to produce a wide array of graph types. These graphs will then be rendered and displayed within the Streamlit application, offering users a dynamic and interactive data visualization tool.

By integrating these technologies, our application will not only enhance data visualization capabilities but also transform the approach to EDA, making it more accessible and efficient for users across various domains.

**Not in Scope / Limitations:** The application will only work with csv files that are in a “long” tabular format, such as a database table. Prior to use with this application, data may need to be cleaned or reformatted to this format.

**To Do’s / Assignments:**

* 1. Readme – Chris
     1. Doug to provide Chris with template for readme section covering how to install applications. (note to self – was there a good for tool for directory tree) -> complete.
     2. 5/16/2024 – 75% plus.
        1. Needs screen shots
     3. Geoff and Doug to review
  2. Slides – Chris
     1. Basic slides since we are demonstrating the application
     2. Slides
        1. Who we are
        2. Project overview
        3. Tech used
        4. Architecture – Doug to provide LangGraph.
        5. Screen shots
        6. Consistent and simple.
        7. Canva
  3. Base application thus far – Doug
     1. 5/15 update. Switched to GPT-4o
     2. Additions
        1. Can specify different graph files.
        2. State flow updated
     3. 3D graphing – now complete.
     4. Debugging.
  4. Jeff – adding speech to text into the prompting
     1. Firas sent material.
     2. Identified streamlit component / widget.
     3. After recording, uses whisper to do audio to text.
  5. GitHub
     1. Invite team – Doug
     2. Geoff – created a branch.
     3. To do – Doug to add branch.
     4. Geoff to present 5/15.
  6. Additional features
     1. 3D graphs and no more at this point.
  7. Research additional sets
     1. Jeff and Chris – 3 and 3
        1. Chris will work on World Data indicator data set.
        2. Jeff – has the Alone dataset
     2. Doug may as B priority, since base application is not quite working
     3. Geoff has one additional set.
     4. Let’s review Geoff’s data. Perhaps no more is needed.

|  |  |  |
| --- | --- | --- |
| **LangGraph Architecture** | | |
| **Type** | **Function** | **Comments** |
| Node and entry point | generate\_graph\_code | Python function that 1) sets up the user and system prompt templates for the LLM call, 2) invokes the LLM, 3) extracts the graphing & filtering code from the LLM response. |
| Node | write\_graph\_code | Python function that writes the extracted graphing & filtering code to the graph file |
| Edge | generate\_graph\_code -> write\_graph\_code | Flow control from generating the graph code to writing the code to the graph file |
| Edge | write\_graph\_code -> END | After writing the graph code, the LangGraph workflow is terminated. |