GridGPT: AI Virtual Assistants for the Smart Grid Applications

Design Document

sddec24-20

Ravikumar, Gelli - **Advisor**

Tin Ngo - **AI Integration Lead**

Nicholas Doty - **Power Systems Lead**

Jackson Phillips - **AI Training Lead**

Emma Heithoff - **Power Systems Lead**

Eddy Andrade-Robles - **Frontend Lead**

sddec24-20@iastate.edu

sddec24-20.sd.ece.iastate.edu

Revised: April 14, 2024

Executive Summary

Development Standards & Practices Used

Practices Used

- Agile Methodology
- Git Version Control
- GitLab CI/CD
- Remote VM Testbeds
- AI Model Development
- Cloud Integration

Engineering Standards

- ISO/IEC 12207 (Software Lifecycle Processes)
- ISO/IEC 27001 for Information Security
- WCAG 2.1 for Accessibility
- ISO/IEC/IEEE 26514 for Design and Development of Information for Users
- IEEE 1782-2022 for Electric Power Distribution Interruption Events
- IEEE 1547 for Interconnection and Interoperability of Distributed Energy Resources

Summary of Requirements

Functional Requirements

- Data Interpretation: AI to accurately interpret grid data into actionable insights.
- User Interaction: Virtual assistants enable intuitive natural language interactions for grid data control.
- AI Model Implementation: Incorporate and customize a pre-trained Hugging Face AI model.

Resource Requirements

- Computational Resources: Adequate processing power and GPU access for LLM training and inference.
- Virtual Machines: Essential for scalable, safe development environments that reflect production settings.

Physical Requirements

- Software Footprint: Lightweight software that does not degrade host system performance.
- Implementation: Versatile software application compatible with any computer.

Aesthetic Requirements

- UI Design and Organization: Clean, intuitive, and well-organized user interface following modern design principles.

User Experiential Requirements

- Accessibility: System accessible to all user skill levels with supportive features like help guides.
- Response Time: Prompt AI responses for efficient decision-making.

Economic/Market Requirements

- Cost-Efficiency: Minimize costs while maintaining system performance.
- Market Viability and Competitive Advantage: Address specific market needs and outperform competitors through AI integration.
- Development Cost: Stay within budget while fulfilling all functional requirements.

Environmental Requirements

- Energy Efficiency: Optimize resource use to reduce energy consumption.

Technical Constraints

- Data Security: Comply with data protection standards.
- Integration: Seamless integration with existing grid management systems and data formats.

Applicable Courses from Iowa State University Curriculum

COM S 227: Object-oriented Programming

COM S 228: Intro to Data Structures

COM S 309: Software Development Practices

COM S 363: Intro to Database Management Systems

COM S 409: Software Requirements Engineering

S E 317: Introduction to Software Testing

S E 319: Construction of User Interfaces

S E 339: Software Architecture and Design

CPRE 412: Formal Methods in Software Engineering

CPRE 416: Software Evolution and Maintenance

CPRE 419: Software Tools for Large Scale Data Analysis

CPRE 450/ 550: Distributed Systems and Middleware

EE 303: Introduction to Power Systems

EE 456: Power System Analysis 1

EE 457: Power Systems Analysis 2

CPRE 459X: Security and Privacy in Cloud Computing

New Skills/Knowledge acquired that was not taught in courses

- Hugging Face
- OpenAI
- High Performance Computing/Slurm
- Docker/Containerization
- AI Integration
- AI Training
- Flask
- Databases (InfluxDB, Neo4jDB, MongoDB)
- Defining grid models with .dss files and their OpenDSS simulation
- React coding language
- Next.js framework
- Reading up and modifying component from documentation (Mantine)
- Python implementation of OpenDSS functionalities

Table of Contents

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

Figures

Tables

1. Introduction

1.1. Problem Statement

As the complexity of power grids increases with the incorporation of diverse energy sources and the need for enhanced reliability, the challenge of efficiently managing these grids intensifies. Traditional methods struggle to keep pace with the rapid evolution of modern energy systems, making it challenging to integrate new technologies and manage the growing data complexity effectively. Concurrently, current grid management software could be faster to adapt to the rapid introduction of diverse energy sources, making the adaptation process tedious and prone to human error. This can lead to inefficiencies and potential system failures, posing significant risks to reliability and safety. GridGPT offers a transformative AI-powered solution to these challenges by enabling operators to use natural language to generate and modify distribution system simulator (DSS) scripts, simplifying the management of complex grid data and reducing the likelihood of errors.

1.2. Intended Users

Utility Operators

Utility operators face the challenge of managing increasingly complex power grids that require rapid decision-making under pressure. They need to monitor large amounts of data from different energy sources and types to ensure grid stability. Also, they need to adapt quickly to changes or emergencies. GridGPT aids these operators by using AI to interpret complex grid data through natural language to simplify decision-making and reduce the risk of errors. This supports them in making informed decisions to enhance the reliability of the grid in preparation for an outage or new infrastructure.

Distribution and Transmission System Focused Electrical Engineers

Smart grid engineers strive to incorporate new technologies into power systems while maintaining efficiency and reliability. Their challenge lies in integrating advancements in the industry as seamlessly as possible. GridGPT supports these engineers by providing tools for real-time data analysis and simulation. They optimize the performance of the grid using college training in the theory of how changes in a point of the system affect the behavior of the connected network.

Grid Modelers

Grid modelers specialize in developing models that simulate the behavior and performance of power grids under various scenarios. They model what is set by an engineer and predict grid responses to different conditions, such as weather and energy demand patterns at endpoints. GridGPT assists them by providing recommendations for various scenarios they may face. Simplifying the data to include the most relevant information allows them to save time but increase model accuracy in collaboration with electrical engineers.

2. Requirements, Constraints, And Standards

2.1. Requirements & Constraints

Functional Requirements

- Data Interpretation: AI must accurately interpret grid data, translating complex datasets into understandable insights for operators.
- User Interaction: Virtual assistants must facilitate natural language interactions, allowing users to query and control grid data intuitively.
- AI Model Implementation: Ability to incorporate a pre-trained AI model from Hugging Face, with provisions for further training and customization.

Resource Requirements

- Computational Resources: Sufficient processing power to train and run LLMs, including GPU access for model training and inference.
- Virtual Machines: Access to VMs. VMs are crucial for creating safe development environments that mirror the production environment. They also offer scalability to manage resources based on workload and task complexity.

Physical Requirements

- Software Footprint: The software should be lightweight, not significantly impacting the performance of the host system it attaches to.
- Implementation: The design will be a software application that can be used on any computer.

Aesthetic Requirements

- UI Design: The user interface should be clean and intuitive, adhering to modern UI design principles to enhance user engagement.
- UI Organization: The UI should be well-organized for easy interpretation, as well as navigation.

User Experiential Requirements

- Accessibility: The system must be accessible to users with varying levels of technical expertise, with features such as help guides and tooltips.
- Response Time: System interactions, especially AI responses, should be timely to ensure efficient operator decision-making.

Economic/Market Requirements

- Cost-Efficiency: Development and operational costs should be minimized without compromising system performance and reliability.
- Market Viability: The product should address precise market needs in the innovative grid management sector.
- Development Cost: The software development should remain within budget while meeting all functional requirements.
- Competitive Advantage: The addition of AI capabilities should provide a clear advantage over similar solutions in the market.

Environmental Requirements

- Energy Efficiency: The system should optimize computing resources to minimize energy consumption.

Technical Constraints

- Data Security: Adherence to data protection standards to securely handle user and operational data.
- Integration: The system must integrate seamlessly with existing grid management platforms and data formats (.dss, .csv).

2.2. Engineering Standards

ISO/IEC 12207 (Software Lifecycle Processes)

- To standardize the software development process, ensuring quality and efficiency from planning to maintenance. Ensures a structured development approach, enhancing software reliability and maintainability.

ISO/IEC 27001 for Information Security

- Critical for protecting data during AI model training and operation, especially when handling sensitive information. Maintains the confidentiality, integrity, and availability of data, fostering user trust.

WCAG 2.1 for Accessibility

- Ensures the software's UI is accessible to a broad audience, including people with disabilities. Expands the user base and complies with legal and ethical standards for digital accessibility.

ISO/IEC/IEEE 26514 for Design and Development of Information for Users

- This standard guides the presentation and clarity of user information. Reinforces what information should be shown to the user and how the information can be presented.

IEEE 1782-2022 for Collecting, Categorizing, and Utilizing Information related to Electric Power Distribution Interruption Events

- Provides guidelines for handling data related to power distribution interruptions. Offers a structured approach to data gathering during faults and differentiates between fault event types, aligning with our goal to enhance the power grid's reliability.

IEEE 1547 for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces

- Sets standards for integrating renewable energy resources into existing power systems. Regulates the reliability specifications of renewable energy additions with testing procedures for performance expectations, ensuring the grid maintains reliability. Our project will test new power source integrations to verify the modified grid's reliability for users.

3 Project Plan

3.1 Project Management/Tracking Procedures

As a professional team, we are committed to adopting and integrating Agile management principles into our workflow. To achieve this, we will organize our tasks into sprints, ensuring they are completed within the designated timeframe. Upon reconvening, we will provide updates on our progress, address any challenges that have arisen, and request assistance as needed to maintain efficiency and productivity throughout the project.

Our primary tool for managing tasks will be GitHub's issue board, which has been meticulously organized and labeled to facilitate efficient tracking of responsibilities. By consistently updating this platform, we ensure that all team members have a clear understanding of the progress made on individual tasks as well as the overall project status. This approach promotes transparency and collaboration within our professional environment.

3.2 Task Decomposition

We plan to break the project down into multiple parts.

- We plan to first research and understand how AI models work. This way, we have a much better understanding of what goes into these AI models.
- Train our AI model.
- How to interpret and use DSS data.
- Work with React and Next.js framework.
- Design the Frontend implementation of the AI model.
- Design the Backend implementation using RESTful API to connect everything together seamlessly.

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Milestone 1.1: The AI model can run successfully.

- The AI model can run without encountering any bugs/errors.
    - This will be determined if the AI model can return an output we expect without encountering any bugs, delays, or incorrect information.

Milestone 1.2: The AI model can successfully interpret a DSS file.

- The model successfully goes through a DSS file with no bugs/errors.
    - This will be determined by asking the AI model to view a specific part of the DSS file and see if it returns the correct interpretation. We will first view and verify the correct answer of said DSS file before allowing the AI model to view it itself.

Milestone 1.3: The AI model can run efficiently.

- The model can run at a reasonable speed and power consumption rate.
    - This will be determined by viewing how long it takes the AI model to respond to a given prompt. Depending on the length of time, it can take up some resources from the computer or power. We will make sure it can run at a reasonable rate based on resources and power consumption.

Milestone 1.4: Virtual Machine setup

- We have our master Virtual Machine set up with our project directory and Docker running appropriately.
    - This will allow us to have an environment we can work on for this project. We each will get our own Virtual Machines that will be cloned from the master VM. Before being comfortable with the setup, we needed to verify with Docker that the database was communicating with the Frontend (as in, the user was being linked to the files being added).

Milestone 2.1: The AI model can successfully interpret a DSS file and return something back to the user.

- The model can run and return something back to the user, regardless of what format it returns.
    - This will be determined by allowing a user-typed prompt to be sent to the Backend, where it will make an AI model call to respond to the given request. The AI model then returns a response to the Backend. If the Frontend successfully receives the response the Backend received from the AI model, then this Milestone is successfully completed.

Milestone 2.2: The AI model can successfully interpret a DSS file and return its interpretation to the user in a language the user can understand.

- The model can run and return what it can interpret in a language that is legible to the user (in this case, English)
    - Check that the response is efficiently written for the context of a fast-paced, serious work environment

3.4 Project Timeline/Schedule

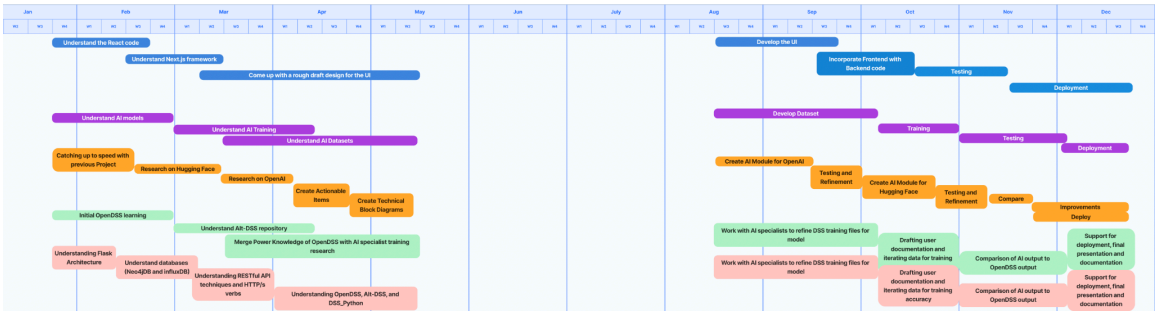


Figure 1
See Appendix 8.4.1 for larger image.

3.5 Risks And Risk Management/Mitigation

Task	Risk	Risk Probability	Mitigation
Researching and Understanding AI Models	Difficulty in grasping complex AI concepts, leading to delays in understanding.	66%	Allocate more time for research, provide additional training resources, and seek guidance from experienced professionals or mentors. Break down complex concepts into smaller, manageable parts.
Training AI Model	Insufficient data for effective training, leading to poor model performance.	49.5%	Perform thorough data analysis beforehand to ensure data adequacy. Implement data augmentation techniques if necessary. Consider transfer learning to leverage pre-trained models. Regularly validate and monitor model performance during training.
Designing Frontend Implementation	Design inconsistencies or usability issues impacting user experience.	66%	Conduct user research and testing to gather feedback early in the design process. Iterate designs based on user feedback and usability testing results. Ensure compatibility across different devices and screen sizes. Follow UI/UX design principles and guidelines.
Designing Backend Implementation with RESTful API	Challenges in API design or implementation leading to communication issues between frontend and backend.	82.5%	Design APIs following RESTful principles and industry best practices. Document APIs comprehensively to facilitate integration and usage. Implement thorough testing, including unit tests and integration tests, to identify and resolve issues early. Consider using API management tools for better monitoring and control.

Table 1

3.6 Personnel Effort Requirements

3.6.1 Effort Requirements for General Epic

Task	**Estimated Hours**
Explore Project Codebase	45
Set up Development Environments	10
Commitment to Team Discussions	20

Table 2

Task	Actual Hours
Explore Project Codebase	20
Set up Development Environments	25
Commitment to Team Discussions	30

Table 2.1

3.6.2 Effort Requirements for Frontend Epic

Task	Estimated Hours
Initial Learning of React Coding Language	20
Next.js Framework Introductory Course	15
Rough Draft of User Interface Implementation	10
Develop the User Interface	35
Incorporate Frontend with Backend	20
Testing	20
Deployment	40

Table 3

Task	Actual Hours
Initial Learning of React Coding Language	6
Next.js Framework Introductory Course	6
Rough Draft of User Interface Implementation	4
Develop the User Interface	33

Incorporate Frontend with Backend	15
Testing	0
Deployment	0

Table 3.1

3.6.3 Effort Requirements for Artificial Intelligence Model Training Epic

Task	Estimated Hours
Understand AI Models	30
Research AI Training	25
Develop Code for Training and Prompting	40
Understand HPC	20
AI Model Training	25
Testing	30
Deployment	20

Table 4

Task	Actual Hours
Understand AI Models	30
Research AI Training	25
Develop Code for Training and Prompting	55
Understand HPC	30
AI Model Training	0
Testing	0
Deployment	0

Table 4.1

3.6.4 Effort Requirements for Artificial Intelligence Model Integration Epic

Task	Estimated Hours
Catching up to speed with previous project	25
HuggingFace Research	25
OpenAI Research	20
Create Actionable Items	15
Create Technical Block Diagrams	15
Create AI Module for OpenAI	30
Testing and Refinement	15
Create AI Module for HuggingFace	20
Testing and Refinement	15
Comparisons between AI Models	10
Improvements	35
Deployment	35

Table 5

Task	Actual Hours
Catching up to speed with previous project	25
HuggingFace Research	15
OpenAI Research	20
Create Actionable Items	5
Create Technical Block Diagrams	5
Create AI Module for OpenAI	60
Testing and Refinement	10
Create AI Module for HuggingFace	10

Testing and Refinement	5
Comparisons between AI Models	0
Improvements	20
Deployment	10

Table 5.1

3.6.5 Effort Requirements for .dss File Analysis Epic

Task	**Estimated Hours**
Initial OpenDSS Learning	30
Understand Alt-DSS Python repository	35
Merge Research with AI Model Research	45
Refine DSS datasets for AI training	30
Iterate dataset for accurate AI output	20
Compare AI output to OpenDSS output in tests	25
Power Specialist support for overall deployment	15

Table 6

Task	**Actual Hours**
Initial OpenDSS Learning	40
Understand Alt-DSS Python repository	40
Merge Research with AI Model Research	25
Refine DSS datasets for AI training	30
Iterate dataset for accurate AI output	0
Compare AI output to OpenDSS output in tests	0

Power Specialist support for overall deployment	20

Table 6.1

3.6.6 Total Effort Requirements by Epic

Epic	Estimated Hours
General	70
Frontend	160
AI Model Training	190
AI Model Integration	260
.dss File Analysis	200

Table 7

Epic	Actual Hours
General	75
Frontend	124
AI Model Training	200
AI Model Integration	185
.dss File Analysis	145

Table 7.1

3.6.7 Total Individual Effort

Team Member	Actual Hours
Tin	187
Jackson	140
Nick	123
Eddy	132
Emma	125

Table 8

3.7 Other Resource Requirements

- We used Iowa State's High Performance Computing (HPC) cluster for LLM training. Training jobs require a large amount of compute power we would not have access to without HPC.
- We used VMs that were to be provided to us by our advisor for development. This way we could ensure no data would be lost and we had access to the project wherever we went. A master VM was created and configured and cloned 5 times to set up a VM for each team member.

4 Design

4.1 Design Context

4.1.1 Broader Context

We are designing our product for use by utility/energy companies.

The community using our product includes people who are responsible for implementing and maintaining power grids.

The communities affected by our project include anyone who has access to energy from a power grid. Nearly everyone in the United States relies on electricity every day and they can all be impacted by this project.

Area	Description	Examples
Public health, safety, and welfare	There are nearly limitless benefits to maintaining stable power grid operation. While our project would not solve problems directly, early detection of developing issues in the grid could be the difference between a normal winter storm and the infamous 2021 Texas power crisis.	In Iowa, with extreme temperatures most of the year, a stable power grid is crucial for safety in both warm and cold seasons so that people can heat and cool their homes and businesses.
Global, cultural, and social	Our project reflects the values of the groups we affect well. Our project is not restricted to any specific cultures, nations, or workplaces as it impacts everything.	Without reliable power, Iowa State's campus would struggle to operate. Our project helps ensure a steady power source.
Environmental	Our project will help reduce the need to get energy from nonrenewable sources.	A power outage may lead businesses to use inefficient diesel power generators in order to provide their space with power.
Economic	This project will save energy companies and local governments money by preventing power outages and other grid issues and solving these issues more quickly.	During power outages businesses cannot process credit cards and will likely need to cease operations during the outage.

Table 9

4.1.2 Prior Work/Solutions

EPRI's OpenDSS runs as a local, desktop application which excludes the opportunity for grid operators at different substation locations to modernize the grid and prepare for faults efficiently. Management of the evolving power grid would be improved with Artificial Intelligence insights on the large amount of data for this perspective of the grid's overall power flow. GridGPT will be in connection with the GridAI power grid model of a previous design group under Professor Gelli. This multiple year working repository provides a large advantage to our beginning skill set in AI model training. A shortcoming of this previous project is the adaptation to their design without the direct contact of being in their process when it was happening.

Security constraints unit commitment, the scheduling technique for when generators are on or off, is producing vast data after it is processed many times a day with minimal input change. Researchers at Argonne National Laboratory [2] developed an AI model with the ability to solve this problem 12 times faster than running data from scratch. Others see the potential for AI in moving from preventive to predictive analysis of power grid equipment with the amount of data collected in

the operation of the component in the grid rather than solely manufacturer nominal values. The new asset management technique would contribute to the economic health of the power grid by supporting technicians and operators with better information in advance to manage reliability for their many customers, even considering complex weather pattern interactions with the equipment parameters [3]. Also, on the path to net zero emissions by 2050, AI provides a practical, more cost efficient implementation in comparison to completely replacing parts of the electricity grid infrastructure [4].

Pros

- Our advisor and client, Professor Gelli, gives constructive feedback on our work to help us reach a functional product. Graduate students are available to help our team assimilate into the work of the previous design project, GridAI, more seamlessly.
- Our team has performed separate research tasks on their roles in the project in order to prepare for the AI assistant well with our limited background knowledge.
- Our connection to the GridAI project provides us a basic understanding of the end user and how our AI assistant can aid them within the current project infrastructure.

Cons

- Compared to an industry setting, our year long course is shorter than the flexibility of moving a deadline for final deployment back.
- We have a variety of engineering types in our team, from software/computer to electrical, to learn how to work on Artificial Intelligence for the first time with varied coursework backgrounds.

4.1.3 Technical Complexity

This design will consist of a backend storing power grid data, including parameters of equipment and historical data, a fine tuned AI model for .dss files, and a frontend to display the response of the AI process of simulating a grid action. Our backend subsystem will receive the user input of an action to be simulated which triggers an API call to the AI model. The AI model will interpret the user prompt and create a response based on the output refined during the training phase. This demonstrates the applicability of a specific dataset on a pretrained model transitioning to more specific file training. The frontend displays the model's response to the user. These components culminate in a built application from connecting various software engineering areas. Engineering judgment will be necessary to compare the AI model output to the current OpenDSS simulation's output from identical input.

Artificial Intelligence training and detailed analysis of the power grid are complex topics both in themselves and more so when integrated with one another. We expect challenges when training the AI model for dss files with many computations, various configurations, and varied natural language prompts. In order to create an application that adapts to the nature of a grid operator's work, we will need to utilize many parameters and observe the accuracy of the responses for feasible use for grid reliability. The backend and frontend integration with the AI model is a collaborative effort that will pose challenges and create changes in our design schedule and approach.

4.2 Design Exploration

4.2.1 Design Decisions

1. Following similar characteristics to other AI implementations with code editors.
    a. This decision is important because we need a way for our users to access the AI model to be able to chat with it. There are two models that we have looked at to get a general idea of how it is implemented. We looked at Google's Colab and GitHub's Copilot to see how their models are implemented from a Frontend perspective. These are vital in order to have consistency with other code editors and their AI models and to keep the overall project organized so it does not confuse the user.

2. Utilizing Python to generate a dataset of DSS scripts for training an applicable AI model.
    a. Professor Gelli recommended a repository of a Python OpenDSS extension that will help us accomplish a chat that could be used for different grid configurations and various electrical calculations. It is beneficial in setting our AI model up for success when we fine tune Hugging Face models, another suggestion from him. Research was done on the Python extension to contribute to our design for a substantial amount of time as a decision of how to use our time and current OpenDSS development. We know the potential in this as a contributor to our diverse training dataset and will gain experience implementing current technologies into our final product. The OpenDSS documentation was a valuable reference to understand the Python OpenDSS extension because neither member of the dataset team had ran power flow simulations with it before.

3. Determining the best GPT models to use for DSS Python script generation.
    a. We determined Meta's Llama 3.1 via HuggingFace was the best model for our project because it has a large context window, has options for large parameter models and small parameter models, and natural language processing is built into this model.

4.2.2 Ideation

—Determining the best GPT models to use for DSS Python script generation.

	OpenAI GPT 4 turbo	OpenAI GPT 3.5 turbo	Hugging Face Meta AI Llama 3.1	Hugging Face OpenAI GPT 2
Parameters	1.76 Trillion	175 Billion	8 - 405 Billion (multiple versions)	137 Million
Context Window	128,000 tokens / 96,000 words	16,385 tokens / 12,288 words	128,000 tokens / 96,000 words	1024 tokens / 768 words
Cost	Input: $0.03 / 1K tokens Output: $0.06 / 1K tokens	Input: $0.0005 / 1K tokens Output: $0.0015 / 1K tokens	Free with ISU High-Performance Computing clusters	Free with ISU High-Performance Computing clusters

Pros	Little setup and is easy to use. GPT 4 is pre-trained. Large context window	Little setup and is easy to use. GPT 3.5 is pre-trained.	Open-source, we control all data. Large context window.	Open-source, we control all data. Much smaller than other open source models.
Cons	We cannot control the security of our data.	We cannot control the security of our data. Less advanced than GPT 4. Small context window.	Lots of fine-tuning is required.	Lots of fine-tuning is required. Small context window.

Table 10

4.2.3 Decision-Making and Trade-Off

1. Following similar characteristics to other AI implementations with code editors.
    a. We decided to design our frontend implementation similar to other code-generating AI assistants because we think it is a proven design that works well. A skeleton of this frontend is already implemented in the GridAI infrastructure that is given to us.
2. Utilizing Python to generate a dataset of DSS scripts for training an applicable AI model.
    a. We are using the open-source DSS repository because we need a large amount of complete DSS scripts in order to train our model. We may want to do further research on more DSS data, so we have as much as possible.
    b. A possible issue we could have with this is that there could be some unneeded files in the repository we are using.
3. Determining the best GPT models to use for DSS Python script generation.
    a. In order to decide on the best model for our use cases, we will evaluate the pros and cons of each model in relation to our use cases.
    b. Some pros could be data security, speed, accuracy, a large context window, and how easy it will be to deploy.
    c. Some cons could be a lack of security, too small of a context window, and how difficult it will be to deploy.

4.3 Proposed Design

4.3.1 Overview

In client meetings, we have focused on the following subsystems within the primary AI focus of our project. The diagram below shows that the database created by our team is preprocessed to best train an accurate Natural Language Processing AI model. The queries entered by users will interact with OpenAI or Hugging Face fine tuned models to output the natural language answer to the grid operator. We will be fine tuning a model to our dataset through machine learning with a generated diverse OpenDSS dataset to produce precise output for assistance to grid operators doing maintenance and future grid planning. This fine tuning includes using python to generate a diverse set of grid configurations and electrical values in order to not produce a project that is only accurate for very few power grid situations.

4.3.2 Detailed Design and Visual(s)

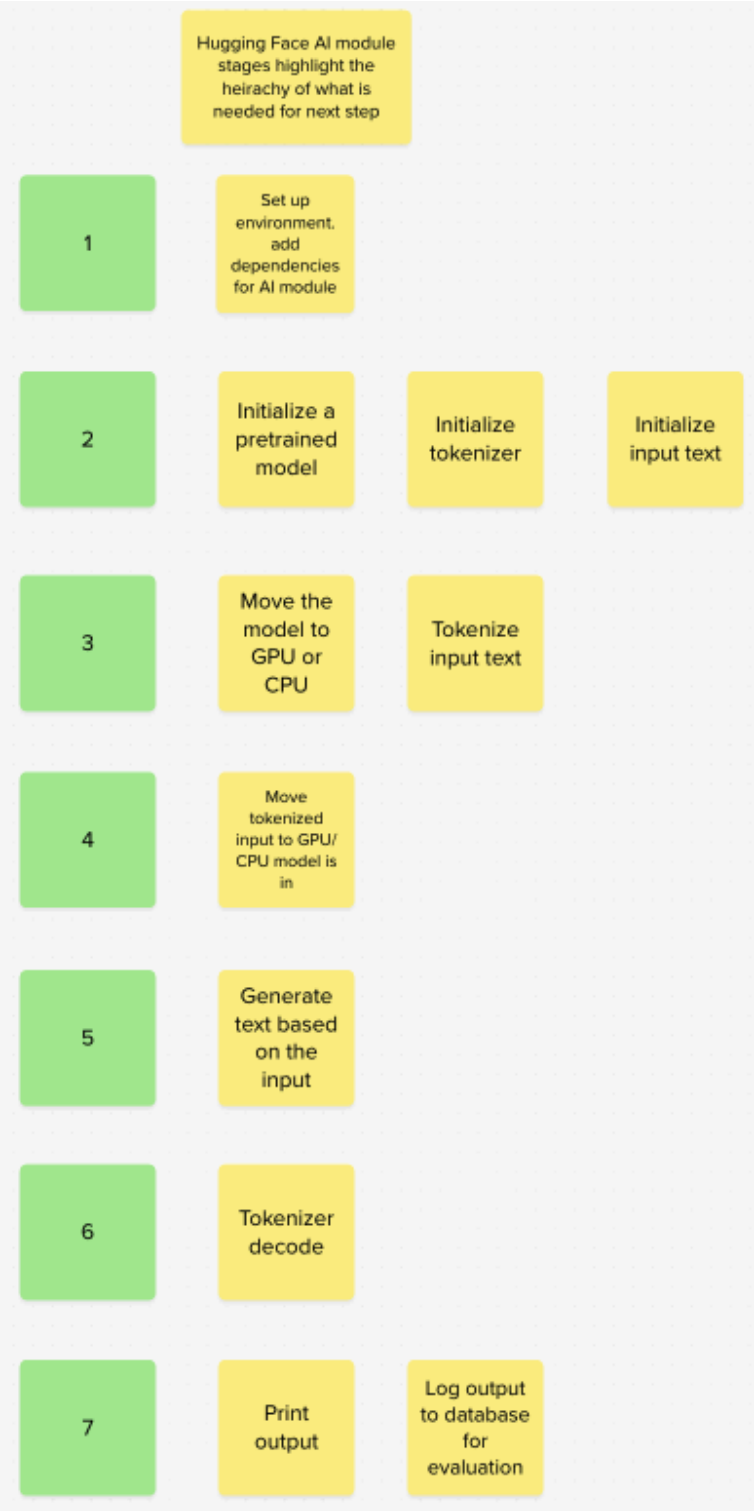


Figure 2

Detailed Description of Each Stage for the Hugging Face Module

Stage 1: Set up environment

- Function: Prepare the software environment required for the AI operations.
- Internal Operations:
    - Containerize the environment in a Docker container
    - Install necessary libraries such as torch and transformers.
- Implementation Details: Ensure that all dependencies are listed in a requirements.txt file for ease of setup. Use of virtual environments for dependency management.

Stage 2: Initialize pre-trained model, tokenizer, and input text

- Function: Load the necessary components for text processing and generation.
- Internal Operations:
    - Load a pre-trained language model, tokenizer, and model configuration from Hugging Face.
- Implementation Details: Use secure and reliable methods to access pre-trained models. Ensure the model name is correctly specified and the system has internet access if needed.

Stage 3: Move the model to GPU or CPU, Tokenize input text

- Function: Optimize model operations by utilizing available hardware and prepare input data.
- Internal Operations:
    - Detect available hardware (GPU or CPU) and transfer the model to this device.
    - Convert raw text input into a format suitable for the model using the tokenizer.
- Implementation Details: Error handling for hardware detection and allocation is crucial, especially in environments with shared resources.

Stage 4: Move tokenized input to the GPU/CPU the model is in

- Function: Ensure consistency in device allocation for both the model and the input data.
- Internal Operations:
    - Transfer tokenized inputs to the same device as the model to avoid cross-device data transfer delays during computation.
- Implementation Details: This step is critical for performance optimization, especially in deep learning models where data transfer can become a bottleneck.

Stage 5: Generate text based on the input

- Function: Execute the model to generate predictions based on the tokenized input.
- Internal Operations:
    - Use the model's generate method to produce output tokens, specifying the maximum length and other generation parameters.
- Implementation Details: Parameter tuning can significantly impact the quality and relevance of generated text, necessitating experiments to find optimal settings.

Stage 6: Tokenizer decodes

- Function: Convert the model's output tokens back into human-readable text.
- Internal Operations:
    - Decode the generated tokens using the tokenizer's decode function, ensuring that special tokens are skipped if necessary.
- Implementation Details: Accurate decoding is essential for the usefulness of the generated text, requiring correct configuration of the decode method.

Stage 7: Print output and log output to the database for evaluation

- Function: Display the generated text and store it for further analysis or auditing.
- Internal Operations:
    - Print the decoded text to the console or another output medium.
    - Log the output in a database for performance tracking and model evaluation.
- Implementation Details: Implement robust logging mechanisms and consider privacy and data security when storing outputs, especially if sensitive or proprietary information is involved.

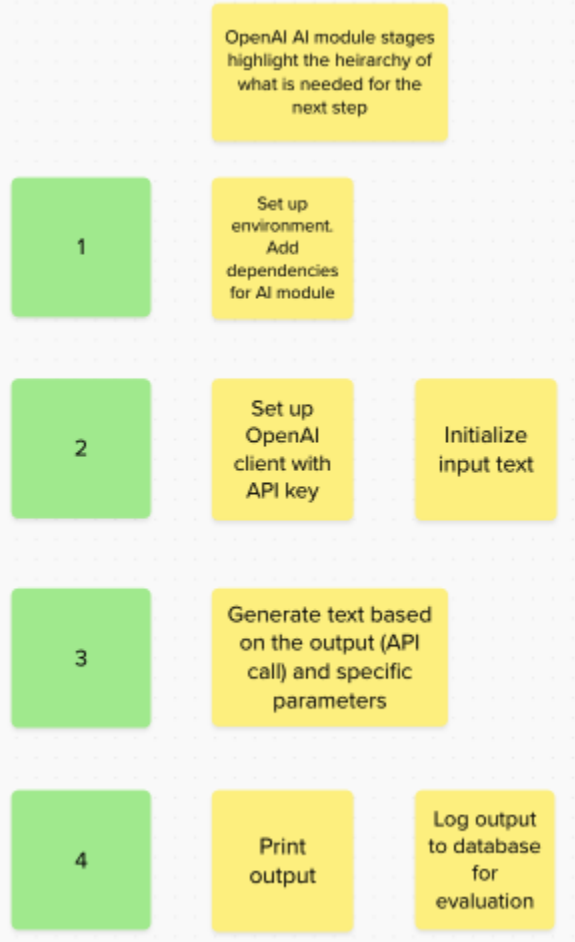


Figure 3

Detailed Description of Each Stage for the OpenAI Module

Stage 1: Set up environment

- Function: Prepare the software environment necessary for the AI operations. This involves installing necessary libraries and dependencies.
- Internal Operations:
    - Containerize the environment in a Docker container
    - Install the openAI library, which is essential for making API calls to the GPT model.
    - Set up other dependencies that might be required for enhanced functionality or integration.
- Implementation Details: Use a package manager like pip for Python. Dependencies should be listed in a requirements.txt file to streamline the setup process in different environments.

Stage 2: Set up OpenAI client with API key, Initialize Input Text

- Function: Configure the OpenAI client with the necessary credentials and prepare the initial input for the model.
- Internal Operations:
    - Initialize the OpenAI API with a valid API key to authenticate the requests.
    - Prepare and verify the input text that will be sent to the model.
- Implementation Details: The API key will be stored securely in encrypted storage. Validation checks for the input text can ensure that it meets the required format or length.

Stage 3: Generate text based on the output (API call) and specific parameters

- Function: Interact with the OpenAI API to generate responses based on the input text.
- Internal Operations:
    - Construct and send an HTTP request to the OpenAI API with parameters like engine, prompt, max_tokens, temperature, top_p, and stop.
    - Receive and process the API response, extracting generated text.
- Implementation Details: Exception handling should be robust, covering scenarios like API rate limits, network failures, and invalid API responses.

Stage 4: Print Output

- Function: Display or output the generated text to the user or another system and log the output to a database for evaluation
- Internal Operations:
    - Simple print statement or integration with a logging system or output module.
    - Log the output in a database for performance tracking and model evaluation.
- Implementation Details: For a production environment, consider implementing a more sophisticated logging or output display system that can handle large outputs or integrate with other interfaces.

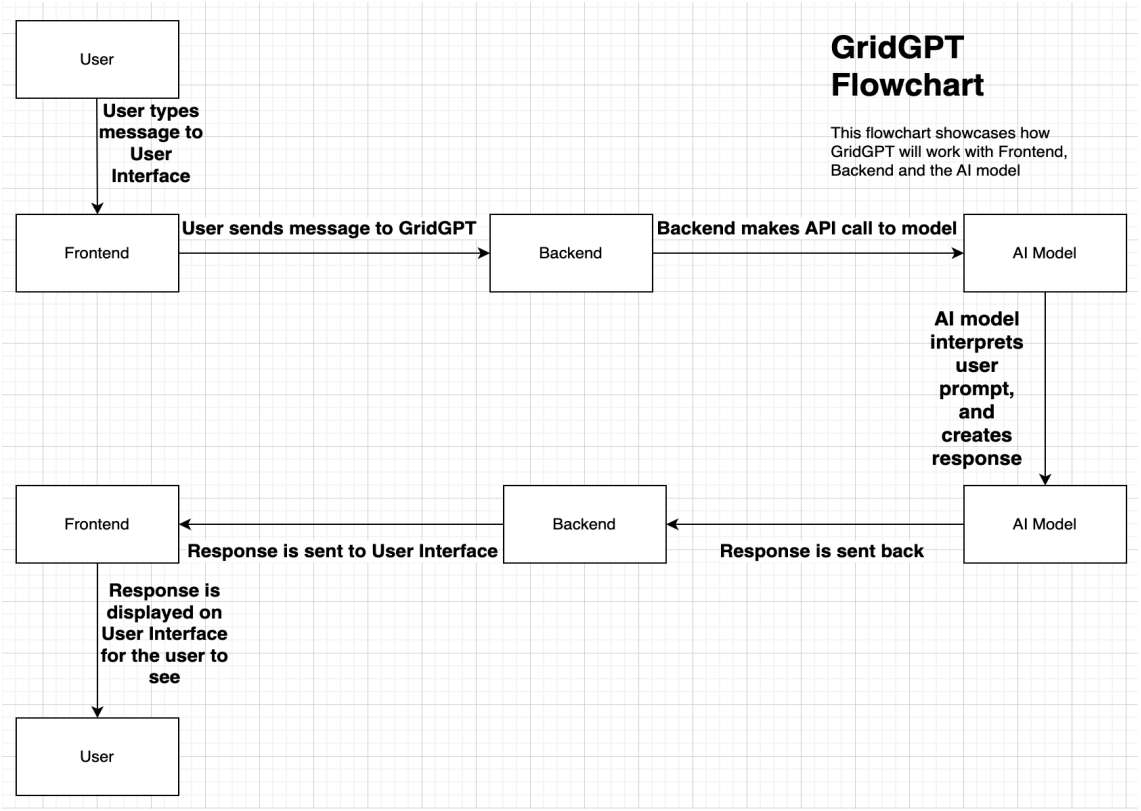


Figure 4

Detailed Description of Data flow from the User Interface to the AI Model and back

Stage 1: GridGPT Sidebar Window

- Function: Pops up a sidebar view on the Code Editor screen of the application, allowing access to the GridGPT chat window.
- Internal Operations:
    - Implement button on the top navigation bar of the Code Editor to access the chat window.
    - Sidebar menu will contain the following:
        - An edit text box, where the user types in their question.
        - A chat history window where the chat between the user and GridGPT.
        - Send button so the user can send their response to GridGPT.
        - An “X” button where the user can exit the window.
- Implementation Details: Use React and Next.js framework to implement the window and the respected components.

Stage 2: Take in User Input

- Function: Take in user input and send the data to the Backend
- Internal Operations:
    - When the user presses the “send” button, the user's prompt will be sent to the Backend.
- Implementation Details: The Frontend will make a Backend call with a JSON call to an endpoint.

Stage 3: Make a call to the AI Model with the given data

- Function: With the given user input, send the data to the AI model to begin processing. access transformers library
- Internal Operations:
    - Call on the AI model and send the users input to the AI model.
    - The AI model will take in the response and begin to create a response.
- Implementation Details: The Backend sends the users input to OpenAI's model, the Backend will make an API call to the model and send the input to the model.

Stage 4: Gather response from the AI model

- Function: AI model sends response back to the Backend.
- Internal Operations:
    - Wait for the AI model to send a response back to the Backend to collect.
- Implementation Details: Within the AI model call, wait for a response from the model and collect it once it responds.

Stage 5: Send response to the User Interface (i.e., print AI model response on GridGPT chat window)

- Function: Collect AI model response from Backend and display it on the chat window
- Internal Operations:
    - Collect the response from the Backend.
    - Display response on chat window.
- Implementation Details: The Backend will send the response to the Frontend, and from there the Frontend will display it on the chat history. This will be GridGPT's response.

4.3.3 Functionality

Grid data is represented by Distribution System Simulation (.dss) files. Grid operators look through these files to observe and analyze grids. This way, they are able to see if there is something wrong with the power grids they're observing. However, as technology advances these .dss files becomes increasingly complex, making it difficult for grid operators to interpret the .dss files and come to conclusions. In our project, the grid operator can view the .dss file on the File explorer on the User Interface. From there, the file will appear in the code view/editor. Now that they are viewing the file, if it is too complex for their understanding, they can click the GridGPT button on the top right to access the GridGPT window. In there, they can type a message saying if the model could help interpret the file to see if there is anything wrong with the power grid. Once the operator submits it, the model will then wait a bit while the prompt is sent to the Backend and to the actual model. When it has finished analyzing the file, the model will send a response to the Backend and then to the Frontend, where it will be displayed in the natural language of the user.

4.3.4 Areas of Concern and Development

In the design of our overall project, we have identified some questions and/or areas of concern. Some of which we have solutions. For one, our client asks us to have the data private or be able to maintain it. However, that depends on the model we choose (either OpenAI or HuggingFace). OpenAI claims that the data is secure with them, but we cannot guarantee that. The main concern is the possibility of a bad actor coming in and attacking OpenAI in some way to gather information. That could lead to the information of our model being compromised, which is very bad. HuggingFace has the advantage of storing the data locally, meaning we will have access to the data.

On the other hand, OpenAI has the benefit of being a large company with the capabilities to hire professional cybersecurity experts to constantly update and maintain their frameworks, especially considering that they are aware that they are one of the largest Artificial Intelligence makers on the planet. We, on the other hand, are students with very limited resources and knowledge compared to those experts. So, although HuggingFace would be more private in the sense that it can be stored locally if an experienced bad actor somehow got access to our system (whether physically accessing or compromising a machine connected to our system), it could potentially be fairly easy for them to steal our information.

With that in mind, using HuggingFace's model is a possible solution.

One question that we do have, and it is something that we will keep an eye out for is if our HuggingFace model will be as good as OpenAI's GPT models. Currently, OpenAI's models rank in the Top 20 in the Chatbot Arena Leaderboard, meaning that their models give good and accurate responses. HuggingFace, however, isn't even ranked on the leaderboard. We're wondering if HuggingFace's models are worked on over time, will it improve to be on par with or better than OpenAI's GPT models?

4.4 Technology Considerations

The two technologies under consideration were using one of the OpenAI GPT models or an AI model on Hugging Face.

OpenAI

Strengths:

- Larger Context Window: With a token limit of 16,385, OpenAI's models can process extensive inputs, allowing for more complex queries and responses. This capability is crucial for interpreting and acting upon the vast amounts of data in smart grid management.
- Cost-Effectiveness: OpenAI operates on a pay-per-use model, which can be more economical for applications with variable demand. The pricing structure is straightforward and potentially more affordable for projects with efficient token usage.
- Ease of Use: The API-based integration simplifies the deployment process, making it accessible even to teams with limited infrastructure management expertise.
- Scalability and Reliability: OpenAI's cloud-based infrastructure offers robust scalability and reliability, ensuring that the AI services can handle varying loads and maintain performance without significant downtime.

- Accuracy and Training: OpenAI's models are among the top-ranked for accuracy in their applications, benefiting from extensive training and vast datasets.

Weaknesses:

- Data Privacy: Utilizing an external API like OpenAI's introduces potential data privacy concerns, as sensitive information is processed outside the local infrastructure.
- Dependence on External Services: Relying on OpenAI's infrastructure means depending on their service availability, updates, and pricing changes, which might affect long-term project stability.

Hugging Face

Strengths:

- Data Privacy: By hosting AI models locally or in a controlled cloud environment, Hugging Face allows for greater data privacy and security, addressing significant concerns in handling sensitive grid data.
- Customization and Control: Local deployment provides more control over the model, including customization, updates, and integration with existing systems, which is crucial for specialized applications in smart grids.

Weaknesses:

- Infrastructure Management: Managing your infrastructure adds complexity and requires significant expertise in cloud services, data security, and AI model maintenance.
- Cost: The fixed cost of cloud instance runtime can be higher, especially for continuous operation, unlike OpenAI's usage-based pricing.
- Scalability Challenges: Self-hosting can limit scalability and might require substantial additional investment to match the demand spikes or growing data volumes.
- Accuracy and Maintenance: Local models might not match the accuracy and performance of OpenAI's continuously updated models, requiring additional effort and resources for training and updates.

Trade-offs and Design Alternatives:

- Hybrid Approach: Considering a hybrid model that uses OpenAI for non-sensitive operations and Hugging Face for data-intensive, privacy-critical tasks could leverage the strengths of both technologies while mitigating their weaknesses.

4.5 Design Analysis

We haven't been able to actually build or train any model. We've been having delays in getting our Virtual Machines (to work on the project), access to the High-Performance Computing (HPC) devices, and OpenAI access. When we received access to these, the next step was to train our AI models on the dataset for OpenDSS functionalities specifically. Personal circumstances did not allow the dataset to be created but the team collaborated on how to progress in other areas of the project. We speak with our client to get more insight and ideas on improving our designs weekly to prepare for the industry review panel.

5 Testing

5.1 Unit Testing

We will have test coverage covering important parts of our system such as the AI module, API endpoints, etc. For most of the backend, we will be using unittest, a popular Python testing framework. We will have automated tests in our CI/CD pipeline, which will ensure confidence in our deployment.

5.2 Interface Testing

Between the AI models and the backend, we will be testing data feeding and result processing by using Postman to hit one of our backend endpoints. We will provide a prompt to the endpoint and take note of the output and performance. We will be comparing both AI models and using OpenAI as the benchmark. We will also test the frontend application and the backend server by simulating interactions between the two using integration tests with tools like Cypress.

5.3 Integration Testing

For our integration tests, we will utilize the Python unittest library to check the integration between different software components, such as the AI processing backend and other API endpoints that interact with the AI processing. Through this comprehensive testing approach, we document all results meticulously to refine the system and enhance its performance, ensuring all components operate effectively together and meet industry standards. We will ensure that the test covers all of the components necessary from the input to the output of our AI assistant. We will also ensure that all of the systems work well together.

5.4 System Testing

We will use a combination of unit testing, integration testing, and regression testing to ensure that each system component functions as expected and communicates smoothly with other components. The system will be configured as microservices, so we will test each microservice to ensure that it works correctly and communicates effectively. To perform system testing, we will combine Python unittest, Cypress, Postman, and CI/CD pipelines.

5.5 Regression Testing

We use docker-compose to run tests before deploying. These tests will be isolated to not interrupt any production level data. If all tests pass, then there shouldn't be any breaking changes if we have good test coverage. Every time we deploy with our CI/CD pipelines that run the scripted automated tests, we significantly reduce the risk of introducing bugs into production, ensuring a more reliable and stable software release. Some critical components to make sure do not break are our systems ability to integrate and process data to feed our AI models, accuracy and performance for overall system, and data privacy and management, which could include secrets, passwords, or other sensitive data.

5.6 Acceptance Testing

We will execute multiple tests from the usage of various input commands to output natural language regarding the power flow solutions. Next, a comparison between the current simulation

process, OpenDSS, and our model will occur to ensure the accuracy of the model for the power flow context. We will analyze the user experience and check for efficiency of our product in the current market of power simulation.

5.7 Security Testing

Due to the wide impact on a community, grid data is private to government entities and related utility companies. Testing will focus on confirming that data would not have an opportunity to be intercepted as the model formulates a response to the AI chat input. User interfaces will be tested for privacy. The API security was previously tested by the senior design group in the larger project of GridGPT named GridAI.

5.8 Results

Testing	Process (how to ensure compliance)	Results
Unit	unittest	Ensure that components are functioning as expected.
Interface	unittest, Postman, Cypress	Test interface functions correctly for all API endpoints.
Integration	Cypress	Ensure the AI module and datasets interact for desired result.
System	Cypress	Test each component for functionality before integration with others.
Regression	CI/CD	Ensure Backend, Frontend, and AI models retain their individual functionalities as all components are integrated together.
Acceptance	Manual QA	Ensuring GridGPT meets client requirements with the end user in mind.
Security	Manual QA	Simulate different vulnerabilities to confirm system security.

Table 11

6 Implementation

We have been focusing on getting up to speed with where the previous teams left off with the project. We have spent our time this semester on preparing our development environments, ensuring we have access to our resources, distributing tasks, and learning new skills for the technologies we are not familiar with, such as DSS and AI. At the beginning of Fall 2024, we will pick up where we left off and begin working on our model on the HPC platform here at ISU. This will begin the design portion of our project and the integration of the current backend code with the frontend plan and training techniques detailed in this document. We aim to produce effective GPT output in relation to the current power simulators available; Though we hope for completion of a viable AI assistant for power systems, we hope to provide meaningful progress towards doing so by working efficiently through each of our roles and together. At the end of our second semester, we have not trained an AI model due to the dataset not being finished. Valuable work aside from successfully finishing our solution was presented each week.

7 Professional Responsibility

"Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment" [5] categories are defined as follows in comparison to IEEE and NSPE standards written in GridAI final documentation by sdmay23-38 [1].

7.1 Areas of Responsibility

Area	Definition	IEEE	NSPE
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	I.6 to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations;	Perform services only in areas of their competence; Avoid deceptive acts.
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	I.3 to avoid unlawful conduct in professional activities, and to reject bribery in all its forms; I.4 to avoid unlawful conduct in	Act for each employer or client as faithful agents or trustees.

		professional activities, and to reject bribery in all its forms;	
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders.	1.5 to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others;	Issue public statements only in an objective and truthful manner; Avoid deceptive acts.
Health, Safety, Well-Being	Minimize risks to safety, health, and well-being.	I.1 to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment;	Hold paramount the safety, health, and welfare of the public.
Property Ownership	Respect property, ideas, and information of clients and others.	II.5 …and to credit properly the contributions of others;	Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.

Sustainability	Protect environment and natural resources locally and globally.	I.1 …to strive to comply with ethical design and sustainable development practices… and to disclose promptly factors that might endanger the public or the environment;	Not applicable.
Social Responsibility	Produce products and services that benefit society and communities.	II. To treat all persons fairly and with respect, to not engage in harassment or discrimination, and to avoid injuring others.	Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.

7.2 Project Specific Professional Responsibility Areas

Work Competence

- We will "perform work of high quality, integrity, timeliness, and professional competence." Our project requires work competence in Power Systems industry events and current Artificial Intelligence techniques. Currently, we are meeting this within the research in our respective roles. We have potential to grow in our competence as a team, from medium to high level, by implementing the Power Specialist's research into the AI training research.

Financial responsibility

- We will "deliver products and services of realizable value and at reasonable costs." Financial responsibility pertains to our AI model training; Our team has highly considered the financial responsibility in the model training research process. Given the vast amount of parameters to create a power grid fine tuned model, our AI specialists have designed a plan to use pre-trained Hugging Face and OpenAI infrastructure to use only necessary costs for fine tuning rather than from scratch.

Communication honesty

- We will "report work truthfully, without deception, and be understandable to stakeholders." Our approach upholds ethical and professional responsibilities because we are passionate about this project. We want to see this through, but in order to do so, we need to be honest with ourselves and with our team. The team contract allows us to take accountability for our mistakes in the event that one of us doesn't adhere to it. Contracts like ours are common in the workforce, where teams working on a project have a contract outlining

everyone's responsibilities and what actions will be taken in case if someone doesn't comply with it.

Health, Safety, Well-Being

- We will "minimize risks to safety, health, and well-being of stakeholders." As we have not yet built our model, we have only highly considered the effect our simulations will have on the power flow for many customers. It is necessary to place this at the highest importance throughout the duration of planning and building of GridGPT due to the risks of faulty power on many customers.

Property ownership

- We will "respect property, ideas, and information of clients and others." We have focused on the privacy that grid data requires, but we can grow throughout the building process because the application of our research requires more insight into this area. There are many customers, substations, generation schedules, and parameters in the interconnected power grid that will be used within simulations which are only government and utility knowledge.

Sustainability

- We will "protect the environment and natural resources locally and globally." GridGPT aims to help optimize the power grid for net-zero emissions in the next few decades. We will continue to place this goal at the forefront by allowing simulations for the grid to easily be optimized with clean energy sources in connection with current power infrastructure.

Social Responsibility

- We will "produce products and services that benefit society and communities." We have not focused on this aspect of the project at more than a low level currently, however, it is important to consider the different types of energy customers we are serving: residential, commercial, industrial. Our design will confirm that our customers are receiving reliable power and the conversion to clean energy is happening swiftly.

7.3 Most Applicable Professional Responsibility Area

Though each area is applicable to our project, the most important is Health, Safety, Well-Being. The power grid is in need of updating for higher reliability and integration of clean energy, however, it is only done through a high safety standard. The simulations done by the AI Assistant need to adhere to the National Electric Code and any applicable safety regulations each time. The safety of users is our highest priority in a utility environment with many power variables at one time.

8 Closing Material

8.1 Discussion

To ensure the success of this project, we will need to analyze both OpenAI and a HuggingFace model, such as Open Orca, in order to determine their efficiency and compatibility with ALTDSS files. Our goal is to provide an accurate description of the data while presenting it in a presentable manner using databases like InfluxDB and Neo4jDB. This will enable grid operators of all experience levels to easily read and diagnose problems within the system.

8.2 Conclusion

Frequently, there is a trend in new technologies being made. Many companies and organizations take part in these trends to use them to their advantage, some even to benefit their customers. Entering the 2020's, being able to see the rise of Artificial Intelligence over the past couple of years has been exciting to be a part of. AI's use has grown exponentially, helping many people across the world with an easier way to gather information and understand complex questions. GridGPT plans to follow along with this trend and help Power Grid Operators understand complex power grid data as these power grids become more innovative and complex. To accomplish this, our team focused on understanding .dss files, which is where the power grid data is stored, training an AI model to interpret those files, and create an organized User Interface to easily access and communicate with the AI model to gain a better understanding.

On the path to a final product, we gained skills in communicating when the project plan was not linear, collaborated with roles in our project that were separate in 491, and assessed our milestones in weekly advisor meetings to guide our next steps. The high growth in knowledge of artificial intelligence and power systems in our team of mixed majors prepares us for applying new knowledge quickly in our future professional endeavors.

8.3 References

List technical references and related work / market survey references. Do professional citation style (ex. IEEE). See link:
https://ieee-dataport.org/sites/default/files/analysis/27/IEEE%20Citation%20Guidelines.pdf

[1] Ravikumar Gelli | Research Assistant Professor | Iowa State University Department of Electrical and Computer Engineering. sdmay23-38: GridAI | Site

https://sdmay23-38.sd.ece.iastate.edu/. Accessed 14 April 2024.

[2] U.S. Department of Energy, "Artificial Intelligence Can Make the U.S. Electric Grid Smarter and More Resilient," energy.gov. [Online]. Available: https://www.energy.gov/technologytransitions/articles/artificial-intelligence-can-make-us-electric-grid-smarter-and-more. [Accessed Apr. 16, 2024].

[3] "TSG Intelligence Artificielle_EN_vDEF_WEB.pdf," Think Smart Grids, Oct. 2022. [Online]. Available: https://www.thinksmartgrids.fr/wp-content/uploads/2022/10/TSG_Intelligence_Artificielle_EN_vDEF_WEB.pdf. [Accessed Apr. 16, 2024].

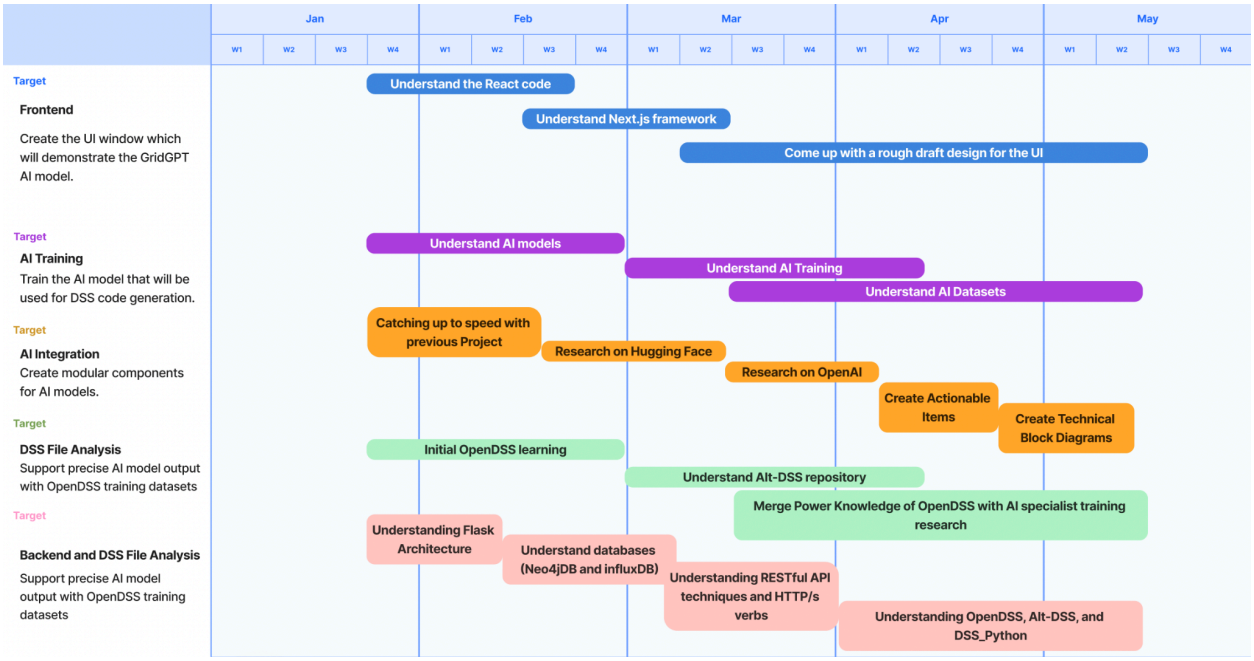
[4] CFA Institute, "AI to the Rescue: Overhauling the US Power Grid on the Path to Net Zero," blogs.cfainstitute.org, Apr. 15, 2024. [Online]. Available: https://blogs.cfainstitute.org/investor/2024/04/15/ai-to-the-rescue-overhauling-the-us-power-grid-on-the-path-to-net-zero/. [Accessed Apr. 16, 2024].

[5] J. Doe et al., "Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment," in Proceedings of the IEEE International Conference on Engineering Education (ICEED)

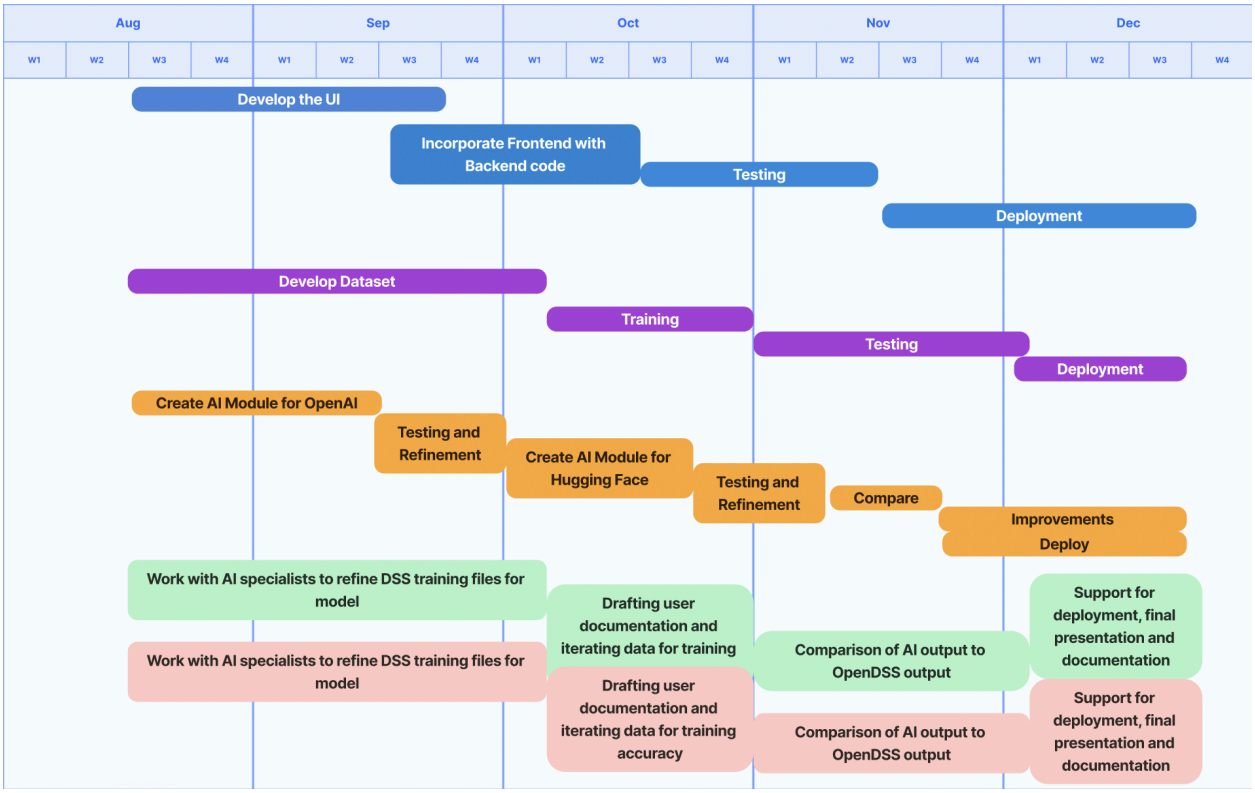
8.4 Appendices

8.4.1 Gantt Chart

Spring semester (491)



Fall semester (492)



9 Team

9.1 Team Members

Tin Ngo

Nicholas Doty

Jackson Phillips

Emma Heithoff

Eddy Andrade-Robles

9.2 Required Skill Sets for Your Project

- Hugging Face
- OpenAI
- Docker/Containerization
- AI Integration
- AI Training
- Flask
- Databases (InfluxDB, Neo4jDB, MongoDB)
- Defining grid models with .dss files and their OpenDSS simulation

- UI website design

9.3 Skill Sets covered by the Team

- Various coding languages (Java, Python, C, C++, MATLAB)
- Backend experience in COM S 309
- Experience in power grid simulation and analysis of different system configurations

9.4 Project Management Style Adopted by the team

Agile

9.5 Initial Project Management Roles

Tin Ngo - AI Integration Lead
Nicholas Doty - Power Systems Lead
Jackson Phillips - AI Training Lead
Emma Heithoff - Power Systems Lead
Eddy Andrade-Robles - Frontend Lead

9.6 Team Contract

Team Name: GridGPT

Team Members:
1) Tin Ngo
2) Jackson Phillips
3) Nicholas Doty
4) Eddy Andrade
5) Emma Heithoff

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:

- We have decided to meet with our advisor on Wednesdays from 1:00 PM to 2:00 PM in Coover Hall, room 2222
- We will be meeting with each other on Discord every Sunday from 1:00 PM to 2:00 PM.

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):

- For our preferred method of communication updates, reminders, issues, and scheduling, we will use a combination of phone texting, discord calls, and face-to-face.

3. Decision-making policy (e.g., consensus, majority vote):

- When deciding on something, we will debate the pros and cons of both sides and make a vote. Majority wins. Bring in outside perspectives if necessary.

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):

- During meetings, we will use GitLab's features to track meetings, contributions, etc.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:

- Attendance is mandatory, with certain exceptions. Minimum participation is updating the team on what you have done.

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

- The team has agreed to complete their tasks one day early, allowing for an extra day to review their work or assist each other in finishing up any pending tasks.

3. Expected level of communication with other team members:

- Full transparency on what we're working on and what we've completed. Try to respond within 24 hours. Stay consistent with response time.

4. Expected level of commitment to team decisions and tasks:

- Be engaged in the project and get your work done. Help others where you can. Everyone has a say in decision-making.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):

- Tin Ngo: AI Integration Specialist - Integrate AI into the software application
- Jackson Phillips: AI Integration Specialist - Integrate AI into the software application
- Nicholas Doty: Power Systems Lead - Understand power grid data and make sure the application meets the acceptance criteria.
- Eddy Andrade: Frontend Lead - Create the UI for the application
- Emma Heithoff: Power Systems Lead - Understand power grid data and make sure the application meets the acceptance criteria.

- We created two AI Integration Specialist roles because we recognized the need for more expertise in this area.

2. Strategies for supporting and guiding the work of all team members:

- After trying to figure out something for 15 minutes, post a message in Discord for help. Post a response to the question to let them know you've seen it.

3. Strategies for recognizing the contributions of all team members:

- We will have a backlog in GitLab, and each person will record their contributions.

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.

- Tin Ngo: SE, Java, Python, etc.
- Nicholas Doty: CPRE, Java, C, C++, soldering
- Jackson Phillips: CPRE, Java, C, Python, soldering
- Eddy Andrade: very good with coding (C, Java, etc). Fast learner.
- Emma Heithoff: Power course sequence, quick learner, data analysis.

2. Strategies for encouraging and supporting contributions and ideas from all team members:

- Good feedback on all ideas. Putting items in the backlog for all ideas in GitLab. Executing on ideas. Have faith.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)

- Talking to the team, TA, or advisor about any issues. Use Mural sticky notes.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

- Learn from the project and get a good grade. Having the project completed. Build collaboration alongside new technical skills.

2. Strategies for planning and assigning individual and teamwork:

- We will use GitLab's features during our meetings to update the team and assign work. GitLab allows you to create items in the backlog. Everyone will pick up items from the backlog based on priority and their own expertise.

3. Strategies for keeping on task:

- Providing regular updates to the team and actively contributing. Holding everyone accountable.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

- Communicate with each other that we have expectations that are required of each other, and those expectations aren't being met. That way, we're all on the same page, and we could potentially handle issues as soon as possible.

2. What will your team do if the infractions continue?

- Escalate the issue to a TA, our advisor, or one of the professors.

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.
b) I understand that I am obligated to abide by these terms and conditions.
c) I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1) Tin Ngo DATE 12/04/2024
2) Jackson Phillips DATE 12/04/2024
3) Eddy Andrade DATE 12/04/2024
4) Nicholas Doty DATE 12/04/2024
5) Emma Heithoff DATE 12/04/2024