GridGPT 2.0: AI Virtual Assistants for the Smart Grid Applications

DESIGN DOCUMENT

Team 42
Client: Dr Gelli
Advisers: Dr Gelli
Ian Louis: Power Co-lead
Luke Eitzmann: Power Co-lead
Scott Rininger: Power Co-lead
Aditi Nachnani: AI Co-lead
Ian Bussan: AI Co-lead
sdmay25-42@iastate.edu
Team Website: https://sdmay25-42.sd.ece.iastate.edu/

Revised: 12/2/2024 / Version 1.0

Executive Summary

GridGPT 2.0 is an add-on to GridAI, which is proprietary software created at Iowa State University. GridGPT 2.0 is used to enrich the user experience of GridAI by providing instructions on how to use GridAI and having extra capabilities for GridAI. We will be making six GPTs for this project to be a combined package for GridGPT. This project is important because it increases the usability of GridAI while providing extra functionality the user might require.

The first step when making GridGPT 2.0 is determining what large language model (LLM) to use. The team looked at two main options, OpenAI's ChatGPT or Google's Gemini. The team chose OpenAI for its functionality and cost. When making the GPTs, the team will be using Docker to containerize the GPTs so they are functional in GridAI. The programming language used to create the GPTs is Python. The team is using Python and Go because it simply incorporates OpenAI API with, and is also the standard programming language used for AI.

So far in this project, the team has started on one of the six GPTs and has done background research into the rest. The GPT the team has started is database GPT (db_GPT). db_GPT is used by the user to learn how to add to, remove, and extract from the GridAI database. The remaining GPTs are for distribution system operators (dso_GPT), map view (map_GPT), power forecasting (forecast_GPT), AltDSS software (dss_GPT), and general grid (grid_GPT). The team has done the necessary research needed to make accurate GPTs for each of these purposes.

The work the team has done is progress towards meeting the requirements but does not currently meet the requirements. This is because the prototyping done is not a usability end product, but part of the necessary journey towards a complete and accurate solution.

The next steps for the team are to finish db_GPT and verify the process works. Simultaneously the team will start on dso_GPT, followed by the remaining GPTs. Verifying the first GPT works as intended is of the utmost importance so the team can continue making the remaining GPTs with confidence that they will work.

Learning Summary

Development Standards & Practices Used

The standards used in this project are IEEE 1547, IEEE 3081, IEEE 2030, OWASP, and ISO 17800.

Summary of Requirements

GridGPT 2.0 must be an easy-to-use extension of GridAI. It must be intuitive and have a short learning curve. Users must be able to use GridGPT 2.0 to better interpret data from GridAI and improve their decision-making. GridGPT 2.0 must add new and useful features that complement GridAI's functionality. GridGPT 2.0 must also interface seamlessly with GridAI. GridGPT 2.0's features must be accurate and display the correct data at all times.

Applicable Courses from Iowa State University Curriculum

- COMS 1270: Introduction to Computer Programming
- COMS 2270: Object-oriented Programming
- COMS 2280: Introduction to Data Structures
- COMS 3090: Software Development Practices
- COMS 3630: Introduction to Database Management Systems
- COMS 5090: Software Requirements Engineering
- EE 2850: Problem Solving Methods and Tools for Electrical Engineering
- EE 3030: Energy Systems and Power Electronics
- EE 4250: Machine Learning: A Signal Processing Perspective
- EE 4550: Introduction to Energy Distribution Systems
- EE 4560: Power System Analysis I
- SE 3170: Introduction to Software Testing
- SE 3390: Software Architecture
- SE 3190: Construction of User Interface

New Skills/Knowledge acquired that was not taught in courses

- Go
- OpenAI
- Context State Management
- Fine Tuning Models
- Docker
- TimeGPT
- LLM to Neo4J and InfluxDB
- OpenDSS software
- altDSS python Library
- Distribution System Operators

Table of Contents

List of Figures/Tables/Symbols/Definitions

Figures

Tables

1. Introduction

1.1. Problem Statement

As power grids grow more complex due to the integration of various energy sources through distributed energy resources (DERs), and with the demand for increased reliability, effectively managing the power grid has become challenging for grid modelers around the country. Traditional methods fail to keep up with the rapid advancements in modern energy systems, complicating the integration of new technologies and the handling of increasingly complex data. Additionally, existing grid management software needs help to adapt to increases in the number of DERs. This is causing a poor adaptation process due to human error. This can result in inefficiencies and system failures, posing significant risks to reliability and safety. GridGPT 2.0 provides an AI-powered solution to these issues, allowing operators to use natural language to create and modify distribution system simulator (DSS) scripts. This simplifies the management of complex grid data and minimizes the risk of human error.

1.2. Intended Users

Utilities

Utilities are often the most active entity on the power grid. Most utilities in the United States act as the operators of their local power grid. Current system operators sit in front of many computer monitors and have to take in large amounts of data. Utilities need a better way to collect and interpret data, so they can forecast future loads and generation. GridGPT 2.0 will be able to help utilities process this important data and consult GridGPT 2.0 on their decision-making. This will shorten a long and drawn-out process, take away less manpower and resources from utilities, and allow for faster responses to changes in the power grid. GridGPT 2.0 will also allow utilities to make changes to their DSS models quickly and more accurately.

Independent System Operators (ISOs)

ISOs are non-profit organizations that are responsible for examining regional electricity grids all over the world. ISOs need to be able to accurately forecast load and generation so that they can accurately set market prices for electricity based on these forecasts. However, as energy grids increase in complexity, traditional forecasting methods struggle to process real-time data and make accurate predictions. GridGPT 2.0 can significantly help ISOs by simplifying complex data interpretation and automating the generation of accurate forecasts. Moreover, GridGPT 2.0 will allow them to make quick decisions and reduce the risk of human error.

Prosumers

A prosumer is a single person or household who both produces and consumes electricity, in this case, they are individuals that use electricity from utilities and also generate electricity from solar or wind. The average prosumer will not be that knowledgeable about the electrical grid, what they need is general knowledge about how they will get money back from generating their electricity. Prosumers need to know data about how much electricity costs, how much electricity is generated from their devices, and other statistics. GridGPT 2.0 will benefit prosumers as it will give them ease of access to data about local electrical grid information, they will have to research less due to GridGPT 2.0 will retrieve relevant data for the prosumer rather than them having to exactly look up regional data and calculate data themselves. GridGPT will save time and money for these individuals.

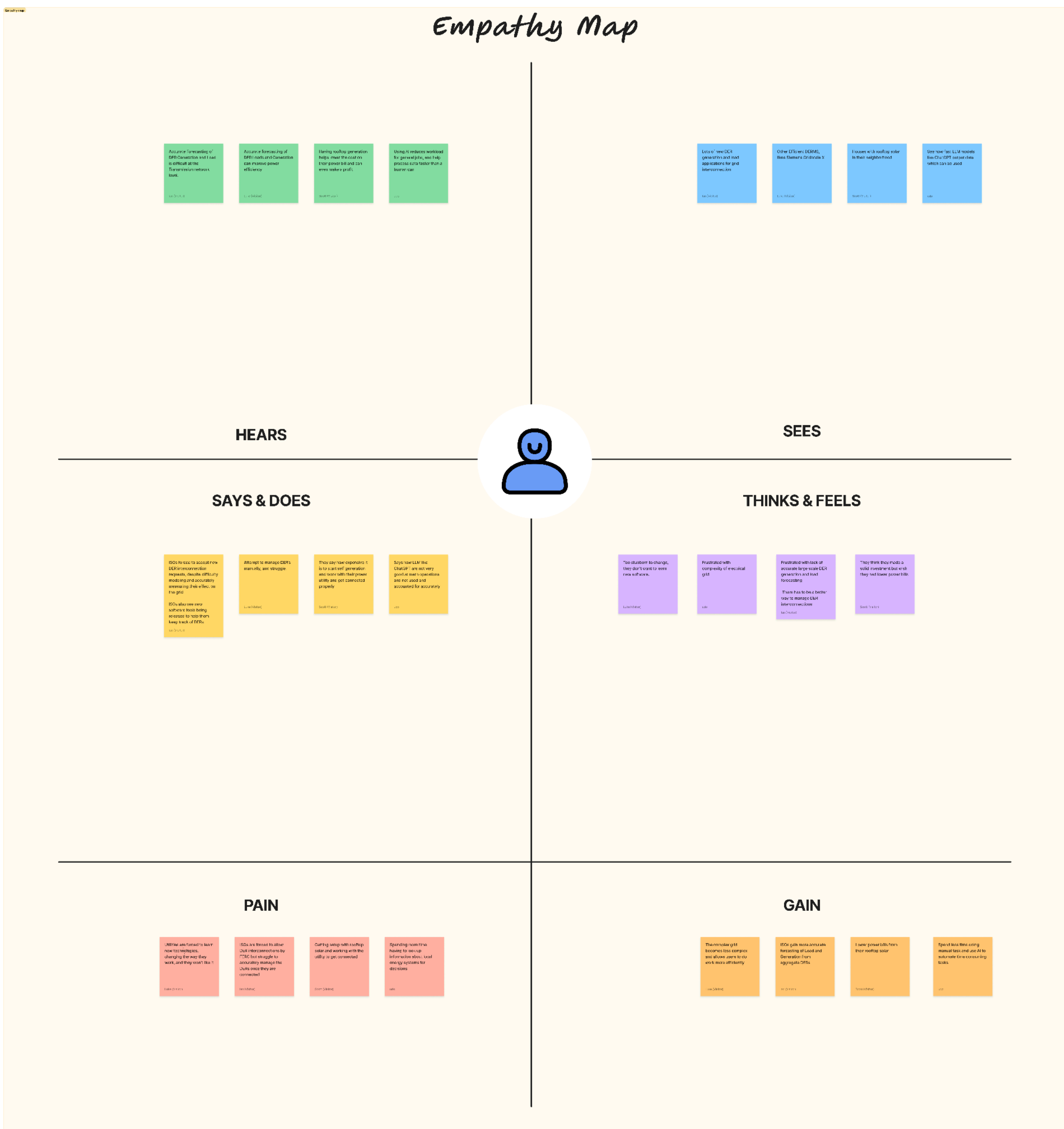


FIGURE 1

2. Requirements

2.1. Requirements and Constraints

Functional Requirements

- Create Fine-Tuned models through OpenAI for purpose-built applications
- Must interact with GridAI to retrieve, process, and present grid data to users.
- Must be able to interact with software like OpenDSS
- Must comply with engineering standards
- Shall ensure personal data can only be accessed by authorized users

Resource Requirements

- Must possess sufficient processing power to fine-tune LLM's and AI model
- Virtual Machines: VMs to access specific software not available on our laptops, and to provide safe development environments

Physical Requirements

- Use physically secure data cloud providers

Aesthetic Requirements

- Modern UI components: ensure components appear clean and have good UX design
- Uses an easy-to-read and aesthetically pleasing color scheme

User Experience Requirements

- Quick and responsive design of the website
- Shall be scalable to accommodate an increasing number of users
- The AI assistant must provide responses instantly or with an acceptable delay

UI Requirements

- Shall have a user-friendly interface that allows users of varying technical skills and visual acuity to navigate without needing extensive training
- Shall provide the option to repeat any tasks

Economic/Market Requirements

- Understands and creates accurate models for the modern
- Creates cost-effective advice for the user to make an informed decision.
- Ensure the product is cost-efficient and within the budget
- The product should be comparatively better than its competitors.

2.2. Engineering Standards

The standards that apply to our project:

- IEEE 1547
  - This standard covers how DERs will interconnect to the grid. This will be important because GridGPT 2.0 will interact heavily with DERs
- IEEE 3081
  - This standard covers the electricity market operations. This will be important because one of the users of GridGPT 2.0 will be independent system operators. So GridGPT 2.0 must be designed with this standard in mind.
- IEEE 2030
  - This standard covers smart grid operations. This will be important for utilities and prosumers as most DERs are part of smart grids. This will apply to GridGPT 2.0 because smart grid operations are an important application of GridGPT 2.0.
- OWASP
  - This standard is about how LLMs process large amounts of sensitive data and must keep data as secure as possible to prevent leaks. GridGPT 2.0 should use the principle of least privilege to prevent invalid access to data.
- ISO 17800
  - Defines a set of data that supports a wide range of energy management applications, including on-site generation, demand response, and electrical storage, all of which will be used by our GridGPT 2.0 project.

3 Project Plan

3.1 Project Management/Tracking Procedures

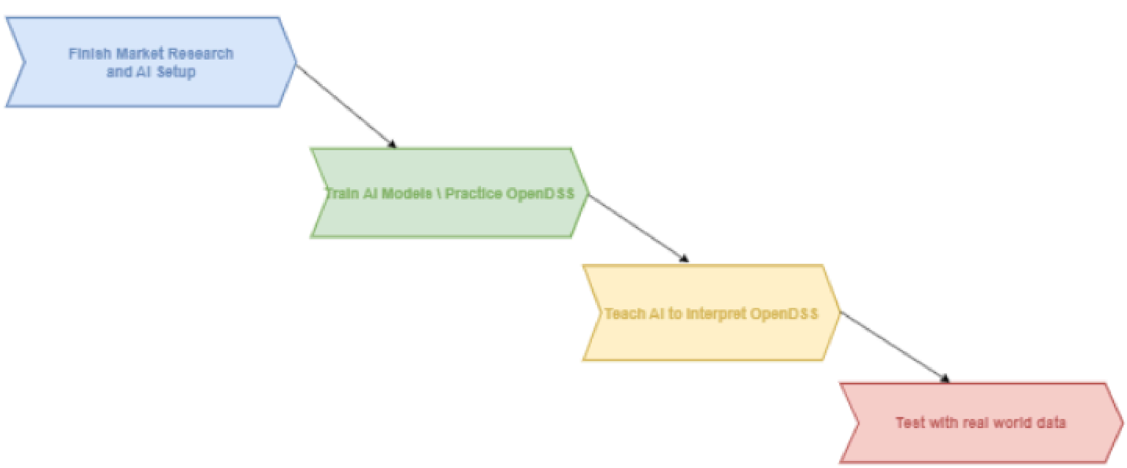


Figure 2

Our project management style is waterfall. We are using this model to allow us to stay focused on the big picture. We chose the different milestones on the waterfall to represent the big-picture goals of the project. We are also tracking our project progress by storing all of our weekly reports in our shared Google Drive. This allows us the ability to go back and see what everyone worked on each week. We are also using GitLab to track the progress of our software. This allows us to know who made what changes and when they made them. By using Google Drive and GitLab we are able to keep everyone on our team accountable for the progress they are making on the project.

3.2 Task Decomposition

We split our group into two different teams, the grid team and the AI team. The natural decomposition is that the grid team will handle the power-focused problems, such as integrating GPT with OpenDSS and specific power system-related use cases, while the AI team will focus on coding-related problems, such as programming the back brain for the GPTs in GridGPT 2.0. Some of the tasks we have created are,

- Understand the existing GridGPT application
- Understand the purpose and how many GPTs are needed for our project
- Find training data for the GPTs
- Test and iterate the code to get the expected results
- Finetune all the GPTs for better responses
- Create the backend for GridGPT 2.0
- Create a user interface for GridGPT 2.0

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Milestone 1: FineTune GPT models for query data from Neo4j.

- Increased accuracy of GPT models by up to 50 percent.
  - Prevented GPT from creating ciphers that were invalid and more accurately related to the schema.

Milestone 2: Fine-tune GPT models for answering user questions.

- Made responses more accurate for the user, making sure they get relevant data for the user.
  - Fine-tuned to use multiple sources for getting information related to user questions, using FluxDB, Neo4j, and more.

Milestone 3: GridGPT 2.0 gives the user an answer that the user can understand.

- GridGPT runs off of the OpenAI API, which is very good at making simple explanations.
  - The power grid is not simple, leading to complex answers.
  - We will use fine-tuning to make the answer easier to understand for the user.

Milestone 4: Incorporate OpenDSS into GridGPT.

- OpenDSS is a common tool used by power utilities.
  - GridGPT will be capable of using OpenDSS files to formula answers for the user.

Milestone 5: Implement GridGPT into GridAI.

- Create a new docker cluster for GridGPT, this would be able to use the other docker containers from GridAI.
    - Able to use docker related to GridAI, using GridAI ML models and resources like FluxDB and Neo4j.

3.4 Project Timeline/Schedule

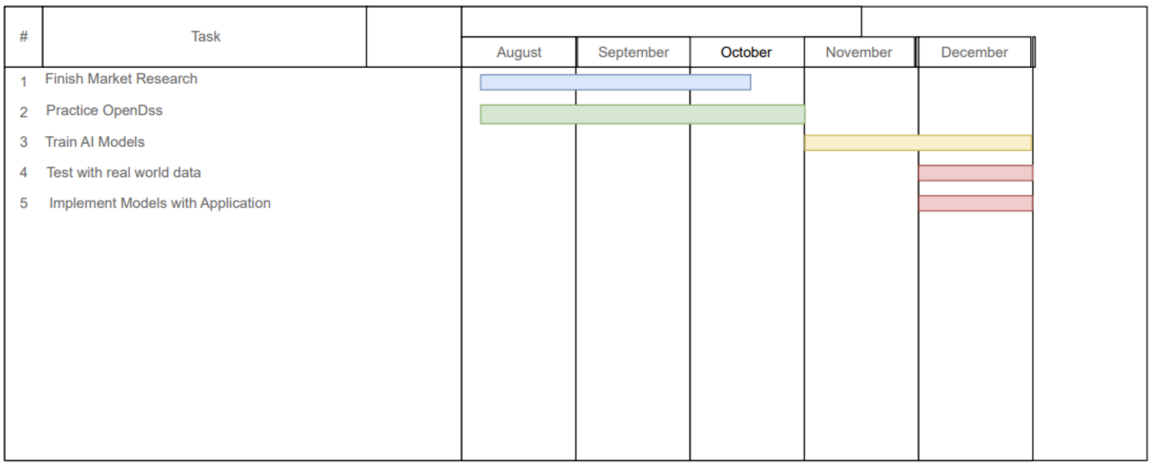


Figure 3

3.5 Risks and Risk Management/Mitigation

Task 1: Finish Market Research

Risk: Having difficulty understanding complex concepts and not knowing what to research

Risk Probability: 60%.

Mitigation:

1. Break down complex topics into manageable concepts.
2. Use external and credible resources to understand the concepts.
3. Take comprehensive notes to retain the information.

Task 2: Train AI Models

Risk: Lack of prompts and data to train the model.

Risk Probability: 66%.

Mitigation:

1. Perform comprehensive research and gather data from various sources like public datasets.
2. Use prompt engineering and feedback loops to clarify prompts.

Task 3: Test with real-world data

Risk: Not obtaining sufficient real-world data to fine-tune the model.

Risk Probability: 20%.

Mitigation:

1. Generate synthetic data by slightly altering existing data.
2. Use publicly available datasets
3. Test extensively until satisfied with the model's responses

Task 4: Implement the model with applications

Risk: Not correctly combining GridGPT and GridAI.

Risk Probability: 40%.

Mitigation:

1. Ensure we understand the GridAI API and codebase to prevent unintended errors.
2. Perform incremental testing until we reach at least a 96% success rate.

3.6 Personnel Effort Requirements

Tasks	Time
Market Research	80
VPP Research	20
OpenDSS Coding	20
AltDSS-Python	20
AI Research	50
Fine-tuning	20
Model Coding	20

Table 1

Fine Tuning Applications	Time
Neo4j Fine Tuning	10
Retrieving Data Fine Tuning	5
Answering User Questions Fine Tuning	5

Table 2

3.7 Other Resource Requirements

Access to GridAI

Access to Windows VM

Access to OpenDSS

Access to OpenDSS data

Access to OpenAI

Access to Neo4j

4 Design

4.1 Design Context

4.1.1 Broader Context

Area	Description	Examples
Public health, safety, and welfare	The power grid is used by virtually everybody in the United States. Our product will make the power grid more stable and reliable for all users, providing them with power when they need it most.	Hospitals, police stations, and fire departments all rely and power to function. Making their access to power more reliable gives them more opportunities to perform life-saving work.
Global, cultural, and social	Our project reflects the goals of utilities and engineers to solve problems in the best and most efficient way.	GridGPT removes human error from the distribution simulation process allowing for more efficient work.
Environmental	There will be an environmental impact due to the energy usage of our product. The demand for energy for our product will cause environmental pollution from the power generation.	Increasing electric demand and environmental pollution
Economic	This product will allow power grid stakeholders to make better and more informed decisions about their distribution grid. This will save them money in the long term and will bring down energy costs.	Lower energy costs will benefit the economy by making prices for goods cheaper.

Table 3

4.1.2 Prior Work/Solutions

We are following the GridGPT 1.0 team that is a semester ahead of us. Even though we are following their team, our design doesn't build off of theirs. We are also using several academic papers for the VPP optimization function of dso_GPT. These papers are cited at the end of our report.

4.1.3 Technical Complexity

1. The design consists of multiple components/subsystems that each utilize distinct scientific, mathematical, or engineering principles. This is because the project has six different subsystems in the form of our GPTs, and each subsystem requires significant software development to complete.
2. The problem scope contains multiple challenging requirements that match or exceed current solutions or industry standards. There is currently no published software that accomplishes all of the functions of our project.

4.2 Design Exploration

4.2.1 Design Decisions

OpenAI: 4o-mini

- We are using OpenAI LLM model 4o-mini, this allows us to have fast LLM model responses along with using an LLM model that is relatively cheap to run.
- OpenAI models allow us to use JSON-formatted responses for a more code-based structure, allowing better accuracy and code usage.

OpenDSS

- We are using OpenDSS software to model the distribution grid. We chose this software because GridAI, the existing project we are working with, also uses OpenDSS.

MongoDB

- We are using MongoDB for context storage and storage for the GPT prompts
- NoSQL design allows us to have flexible data storage design for context storage and user data

4.2.2 Ideation

Model	Require T5?	Input Token per 1M	Output Token per 1M	Input Per 1000 tokens	Output Price per 1,000 Tokens
01 preview	YES	$15.00	$60.00	$0.015	$0.06
01 mini	YES	$3.00	$12.00	$0.003	$0.012
40	NO	$5.00	$15.00	$0.005	$0.015
Gemini 1.5 Flash	NO	$0.075	$0.30	$0.000075	$0.0003
40 mini	NO	$0.150	$0.600	$0.00015	$0.0006

Table 4

4.2.3 Decision-Making and Trade-Off

We chose option 40 mini based on the weighted matrix score result.

We first had to pick a LLM Provider, for us, this was between Gemini and OpenAI. We had to focus on important factors like speed, accuracy, and cost. Based on this, we rated each model on these, a higher score is better and resulted in 40 mini being the best overall. This is due to overall being fast, decently accurate, and cheap.

Model	Speed	Accuracy	Low Cost	Score
01 preview	5	10	1	50
01 mini	5	10	5	500
Gemini 1.5 Flash	5	5	5	125
40	5	9	5	225
40 mini	10	8	8	640

Table 5

4.3 Proposed Design

4.3.1 Overview

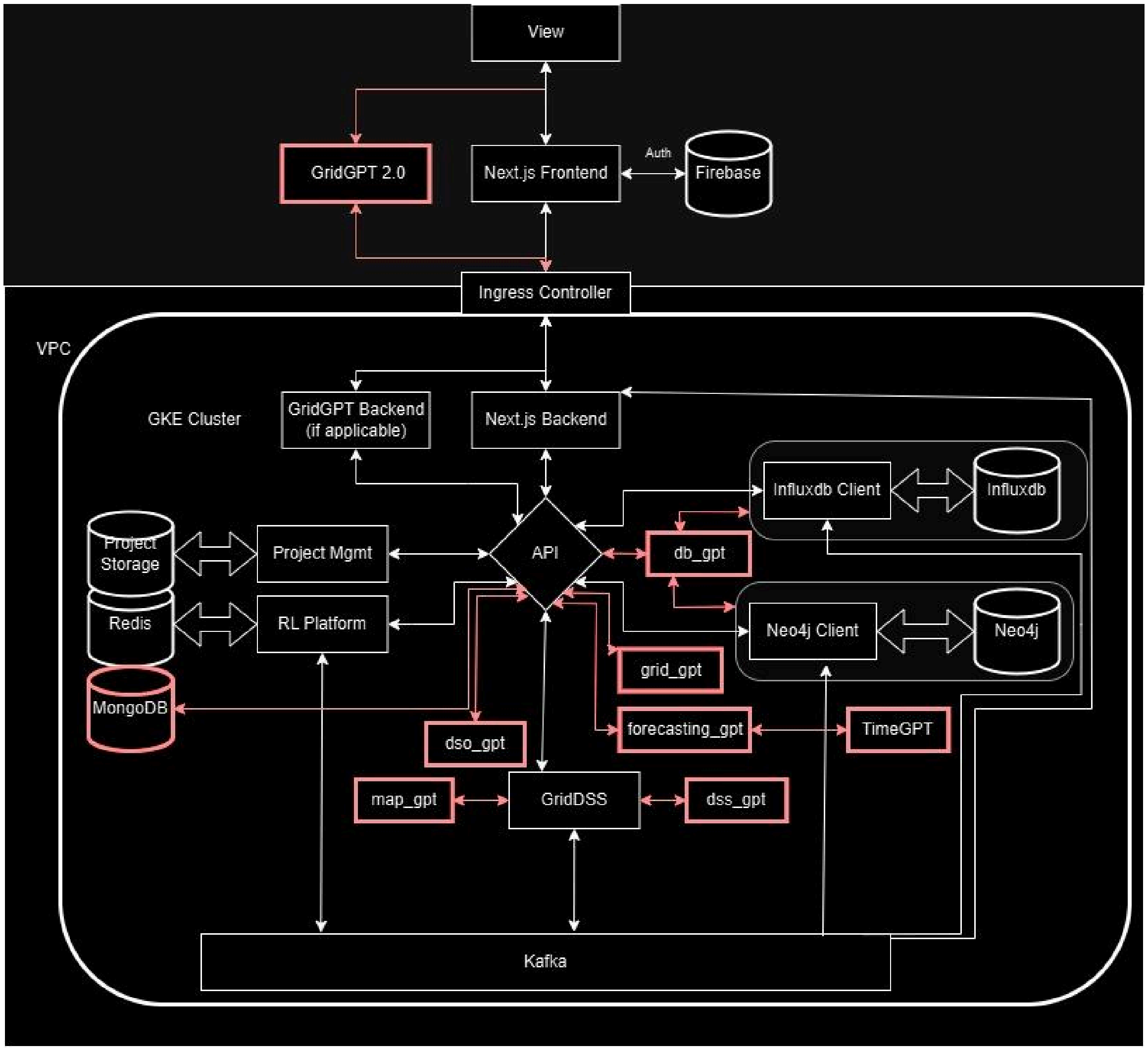


Figure 4

Our design for GridGPT 2.0 works by having a user input a prompt which GridGPT 2.0 processes. GridGPT 2.0 will then decipher what the user is asking for, and then utilize GridAI to perform the user request. The user prompt follows two different paths. The first path is the user asking for specific data about the system. This will have GridGPT 2.0 accessing the Neo4j grid model and returning the values it finds to the user. The other path has the user asking GridGPT 2.0 to change aspects of the model. This has GridGPT 2.0 processing the user request to understand what the user is asking. Then GridGPT 2.0 interacts with GridAI and has GridAI make direct changes to the OpenDSS model.

4.3.2 Detailed Design and Visual(s)

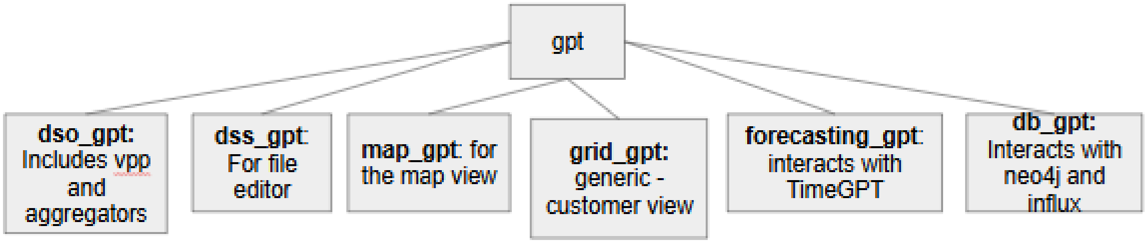


FIGURE 5

Our design has two main components, the frontend and the backend. The front end is what the user will see and interact with. It will be housed at the top right of the GridAI user interface. The backend will be the code section that is doing all of the work. This is the part that will take in the user request, go through the databases to find relevant information to the request, and then manipulate the information from the databases to give an accurate answer to the user. We are using a containerized approach, each of the components we are building are docker components that will be deployed on GridAI. Our chatbot will use our containerized application to retrieve data from relevant information from databases, these include Neo4j, FluxDB, and MongoDB. This information will be sent back to the chatbot for OpenAI to answer user questions. The backend has 6 different GPTs, each one performs a different function. The dso_GPT aggregates DERs together to form virtual power plants (VPPs). The dss_GPT edits the OpenDSS files for the user. The map_GPT displays the OpenDSS results in a more readable and easy-to-understand format. The grid_GPT is a traditional chatbot that answers users' questions about the model and data. The forecasting_GPT predicts future generation output and demand of the system loaded in GridGPT 2.0. The db_GPT uses the OpenDSS data to build neo4j and influx models of the system.

4.3.3 Functionality

In the real world, our design is intended to be accessed by engineers and other industry professionals. The goal of our design is to allow the user to retrieve information and make changes to their distribution models as easily as possible. Our design will operate by having the user submit a task to a chatbot that is connected to GridGPT 2.0. Then GridGPT 2.0 would confirm with the user what it thinks the user is asking it to do. If GridGPT 2.0 and the user agree on its task, then GridGPT 2.0 will perform the task with the assistance of GridAI. It will utilize one of the six GPTs we are creating to perform the task. Then the user will receive the output from GridGPT 2.0. If the user's task involves changing the OpenDSS model, then GridAI will return the updated model to the user. The user will then review the updated model to ensure the correct changes were made. Then the user will tell GridGPT 2.0 that the model was successfully updated. GridGPT 2.0 will then save the changes it made to the model.

4.3.4 Areas of Concern and Development

The current design is on track to satisfy the project and user requirements. We are still in the initial stages of prototyping our concepts, so our design is very premature. However, we are making strides toward a fully functioning project. Currently, our primary concern is building out the chatbot, data management, and virtual power plant optimization functionalities of GridGPT 2.0. Another concern is finishing the development of all of the GPTs. Another concern is ensuring that GridAI and GridGPT 2.0 are compatible. To fix these issues, we will spend more time working on and designing these features. With more time spent on building and designing these features we will make progress and finish these features. We also plan to clarify questions about the integration of GridGPT 2.0 and GridAI with Dr. Gelli to help make design decisions. This will allow us to better understand our design constraints and make progress on the overall design. It will also prevent us from using software techniques and libraries that are incompatible.

4.4 Technology Considerations

The technologies we had to consider were using either OpenAI or Gemini API for LLM Models. We also had to decide how to interact with the OpenDSS files, so we chose to use a Python library called altDSS.

OpenAI

Strengths:

- Scalability: OpenAI models can handle huge amounts of data requests, with a token limit of 16,384 per request.
- Fine-Tuning Customization: OpenAI models can be fine-tuned to ensure that the model is able to respond to specific tasks and decrease the amount of errors.

Weaknesses:

- Data privacy and security issues: Using an external API can lead to data breaches if sensitive information is being transmitted.
- Prediction Error: Most LLM models are prone to make errors sometimes if not trained properly.

Gemini API

Strengths:

- Integration with other Google products: Gemini allows easy integration with other Google products that can enhance productivity if the goal is to plan additional Google features.
- Multimodal Capabilities: Gemini is able to process text, code, and images and perform across different types of media.

Weaknesses:

- Performance inconsistency: GeminiAPi shows weaknesses in responding to questions related to math and coding logic. Certain models are also unable to generate diagrams.
- Not proven: Since Gemini API was launched in 2024, it is relatively new and doesn't offer a lot of resources to troubleshoot.

4.5 Design Analysis

We have implemented components of our project. Such as building components of the chatbot will use: Neo4j, FluxDB, GridDSS, and DERs applications. We have successfully built applications and functional work. However, our overall design must still implement these features together so our chatbot can use these components.

We have working application code with Python scripts, and we need our future design to consider deploying components as docker containers. These containers will be components of GridAI to be used by the chatbot on the front end. We believe that our design is feasible as we have to expand our current components and correctly attach them to the existing GridAI application.

5 Testing

Testing is an important aspect of our project. This is a software project so our responsibility is to ensure that the software works as expected and is correct. We have to ensure that the data flow between the frontend, backend, and databases is correct; we will use Swagger, Postman, Influx CLI, and other software testing tools to quantify the correctness of the data flow. In addition ,we also must make LLM results have high accuracy as part of having correct responses to users. In addition we will also be using CI/CD testing to make sure that the previous implementations are still working as expected. Overall we are focusing on a system of testing that consists of iterative testing and detecting issues early on the project to reduce further issues. We plan to use unit, interface, integration, system, regression, acceptance, and security testing as part of the requirements specified for this project.

5.1 Unit Testing

We will be using unit testing to test individual methods and systems as a unit and quantify the correctness of these methods and systems. We can test individual methods with Postman and Swagger to determine the correctness of a method by validating the results. We will also be testing systems and larger-scale interactions as a unit of testing and be using tools like Jest to test end-to-end tests and see if the system works as expected.

5.2 Interface Testing

Our interface testing consists of determining the correctness between our components of the application; this is done with Swagger and data flow testing.

We are using Swagger for documentation of the API and testing of endpoints. Swagger is an interface that can be used for testing where the user can input values for the parameters test and endpoint. We will use this to validate that all endpoints are working as expected and do not have any errors.

Another Interface for testing will be using data flow testing. We can have data flow tests between the frontend, backend, and database systems to determine that the information is processed correctly and displayed visually correctly on the frontend.

5.3 Integration Testing

Our Integration testing consists of testing the most important paths in our system and validating the correctness of these systems, we will be focusing on front-end backend integration and Backend

to Database testing.
Frontend-to-backend integration is critical as that information from the systems must be correctly shown to the user and sent back to the system to process data correctly. This will be done with Swagger and Postman to validate queries, send requests, and real time data.

Backend to Database testing, test the database systems' accuracy and responsiveness and results. We will test for the Cipher queries' correctness and data retrieval. We will use Neo4j Browser and InfluxDB CLI to determine the correctness of these integrations between the backend and the databases.

5. 4 System Testing

The system level testing ensures that the systems work as expected and meet the requirements. We will be using End to End workflow testing, making sure that all components and modules of the project work as expected. Our requirements define that we need to make an accurate system and ensure security, and responsiveness. We will be using tools like Swagger and Postman to ensure these requirements suffice for the system-level testing.

5. 5 Regression Testing

We use Git CI/CD testing to ensure all the previous components work. This CI/CD is set so that it tests other components of the code base that were previously built, like all the GridAI dockers. With this functionality, when we add new features and test if they impact the other features.

5.3 Acceptance Testing

With acceptance testing, we will ensure that both the function and non-functionality are met, this includes involving our client in the acceptance testing. We will demonstrate that we have completed the function and non-functional requirements specified. We will test these functional requirements by testing the GridAI 6 docker application with the expected workflow and test the accuracy of these results. Also will make sure we involve the client by having the client provide feedback based on our implementation results of the project to make sure results are as expected.

5.4 Security Testing

We will be using Swagger and Postman to test the security of the API such that only authorized users should be able to access specific data; this ensures that our endpoints are made to only support the correct users and types of users when accessing data.

5.5 Results

The results of our testing so far are that we have components of the CI/CD and Swagger working and have used them on some new implementations of code. We will be testing iteratively as we develop more of the project and use further tools for testing. We use these to ensure the information is correct and useful for the end user. They demonstrate compliance with the user's needs. In regards to our user types: Utilities, ISOs, and Prosumers, they are in need of a system that is accurate and provides accurate statistics, reduces the risk of human error, and helps save time. These are important results that we need to have tested. Our further next steps consist of using the tools and testing methods described above as we develop our project.

6 Implementation

Neo4j Implementation: This implementation is testing the system of using Gpt4 to create queries from the neo4j model. This was a mock model used to represent a grid model to test querying from a working Neo4j database. We plan to use this in the db_docker.

```
n
(:PowerPlant {name: "Plant A",capacity: 1000})
(:PowerPlant {name: "Plant B",capacity: 800})
(:Substation {name: "Substation 1"})
(:Substation {name: "Substation 2"})
(:Transformer {name: "Transformer 1",rating: "500kVA"})
(:Transformer {name: "Transformer 2",rating: "250kVA"})
(:Consumer {name: "House 1",consumption: 5})
(:Consumer {name: "House 2",consumption: 3})
(:Consumer {name: "Business 1",consumption: 20})
```

Figure 6

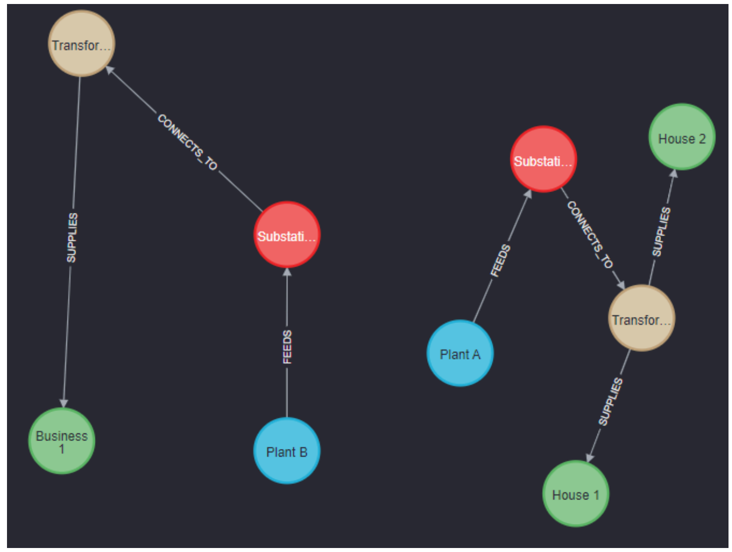


Figure 7

dss_GPT implementation: The code shown below allows our dss_GPT to access OpenDSS files and find the requested information.

```python
api_key = os.getenv("OPENAI_API_KEY")
client = OpenAI(api_key=api_key)

dss_files = ['Transformers.dss']

def read_dss_file(file_path):
    with open(file_path, 'r') as file:
        data = file.readlines()

    return data

dss_contents = {file_path: read_dss_file(file_path) for file_path in dss_files}
##print(dss_contents)
dss_contents_summary = []#extracting and parsing
for file_path, lines in dss_contents.items():
    for line in lines:
        if line.startswith("New"):
            dss_contents_summary.append(line.strip())

dss_info_summary = dss_contents_summary[:10]

def getResponse(userInput):
    context = "\n".join(dss_info_summary)
    prompt = f"{context}\n\nUser Query: {userInput}\nResponse:"

    try:
        response  = client.chat.completions.create(
            model="gpt-4o",
            messages=[
                {
                    "role": "user",
                    "content": prompt
                }
            ]
        )
        return response.choices[0].message.content
    except Exception as e:
        return f"An error occurred: {e}"


if __name__=="__main__":
    userInput = input("You: ")
    response = getResponse(userInput)
    print(f"GridGPT: {response}")
```

Figure 8

```
PS A:\SR\Senior Design - 1> python test.py
You: tell me about transformer P1R_nCCTT0,
GridGPT: The transformer labeled "P1R_nCCTT0" is configured as follows:

- **Type**: Three-phase transformer with 2 windings.
- **Primary Winding (Wdg 1)**:
  - **Tap Setting**: 1

- **Secondary Winding (Wdg 2)**:
  - **Bus**: P1R_nCCTT0_BT
  - **Rated Voltage (kV)**: 0.48
  - **Power Rating (kVA)**: 75
  - **Tap Setting**: 1.03

- **Leakage Reactance (XHL)**: 3.24%
- **Load Loss**: 1.74408%
- **No-Load Loss**: 0.6%
- **Base Frequency**: 60 Hz
- **Substation Association**: Not associated with a substation
  - **Tap Setting**: 1

- **Secondary Winding (Wdg 2)**:
  - **Bus**: P1R_nCCTT0_BT
  - **Rated Voltage (kV)**: 0.48
  - **Power Rating (kVA)**: 75
  - **Tap Setting**: 1.03

- **Leakage Reactance (XHL)**: 3.24%
- **Load Loss**: 1.74408%
- **No-Load Loss**: 0.6%
- **Base Frequency**: 60 Hz
- **Substation Association**: Not associated with a substation

This transformer steps down from a medium voltage of 12.47 kV to a lower voltage level
```

Figure 9

dso_GPT implementation: This code shows how our python scripts will leverage the altDSS library to access the OpenDSS files. This code also shows how we will generate the optimal virtual power plant for the DER data in the OpenDSS files.

```
[ ] import csv
    from altdss import altdss

    altdss(f'''
        Clear
        Redirect "{IEEE123Solar_PATH}"
    ''')
```

```
from dss import SolveModes
altdss.Solution.Mode = SolveModes.DutyCycle
altdss.Solution.Solve()
```

```
[ ]
    import matplotlib.pyplot as plt


    loadshape = altdss.LoadShape.Name
    import numpy as np


    points = np.arange(1, 301)

    p_values = altdss.LoadShape.PMult

    p_values = p_values[1]*2

    # Plot the loadshape
    plt.plot(points, p_values)
    plt.xlabel("seconds")
    plt.ylabel("MW")
    plt.title(f"Loadshape: Solar Ramp")
    plt.grid(True)
    plt.show()
```

```
heatProductionCost = []
for i in range(len(heatGeneration)):
  heatProductionCost.append(heatGeneration(heatGeneration[i].Heat_Flow,heatGeneration[i].heatFlowAplha))

electricPowerGenerated = []
for i in range(24):
  electricPowerGenerated.append(electricPowerPlants[0][i].sum())

electricProduction = [len(electricPowerPlants)][24]
electricProductionCost = [24]
for i in range(len(electricPowerPlants)):
 for j in range(24):
  electricProduction[i][j]=(Electric_Production(electricPowerPlants[i][j].electricAlpha,electricPowerPlants[i][j].PowerGenerated,electricPowerPlants[i][j].electricBeta,electri
for k in range(24):
  electricProductionCost[k]= electricProduction[0][k].sum()

combinedProductionCost = []
coGeneratedPower = []
for i in range(len(combinedGeneration)):
  Power, hourlyCost = (CHP_Production(combinedGeneration[i].CHP_Kc,combinedGeneration[i].CHP_Hc,combinedGeneration[i].CHP_alpha))
  combinedProductionCost.append(hourlyCost)
  coGeneratedPower.append(Power)

gridExchangeCost = []
powerExchangedWithGrid = []
for i in range(24):
  powerExchangedWithGrid.append(TotalElectricDemandHourly[i] - electricPowerGenerated[i])
  gridExchangeCost.append(gridExchange(Cpgrid[i],powerExchangedWithGrid[i]))
```

```
for i in range(24):
      m.setObjective((heatProductionCost[i].sum())+(electricProductionCost[i].sum())+(combinedProductionCost[i].sum()+gridExchangeCost[i].sum()),GRB.MINIMIZE)
      m.addConst((electricPowerPlants[i].PowerGenerated.sum())+(coGeneratedPower[i].sum()+powerExchangedWithGrid[i].sum()-TotalElectricDemandHourly == 0))
      m.addConst((heat_Production[i].Heat_Flow.sum())+(heat_Production[i].CHP_Hc.sum())>= 0)
      m.optimize()
```

Figure 10

Figure 11

7 Ethics and Professional Responsibility

In developing GridGPT 2.0, we prioritize engineering ethics and professional responsibility by committing to public safety, transparency, fairness, sustainability, and data security. Our team defines these principles as ensuring our tool enhances grid reliability while promoting societal and environmental well-being. We follow rigorous testing protocols to uphold safety standards and conduct bias assessments to ensure equitable outcomes. Additionally, we design with sustainability in mind, facilitating renewable energy integration and minimizing environmental impact. Regular team discussions guide our efforts to responsibly address ethical challenges.

7.1 Areas of Professional Responsibility/Codes of Ethics

This discussion is with respect to the paper by J. McCormack and colleagues titled "Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment", International Journal of Engineering Education Vol. 28, No. 2, pp. 416–424, 2012

Areas of Responsibility	Definition	IEEE Code of Ethics	Team Interaction
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	We have heavily researched all of the work in the field of our project.
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	I.3 to avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist	Though our product will likely use a lot of power, the economic benefits to the power industry will heavily outweigh its consumption.
Communication Honesty	Reports work truthfully, without deception, and are understandable to stakeholders.	I.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 the contributions of others properly	We have honestly reported all of our work weekly and documented it for potential clients.
Health, Safety, and Well-Being	Minimize risks to safety, health, and well-being.	I.1 ... to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment	Our AI chatbots will provide no risks to the public.
Property Ownership	Respect the property, ideas, and	I.5 ... and to credit properly the	It will be a tool that will have no effect on

	information of clients and others.	contributions of others	the property and ideas of others.
Sustainability	Protect the environment and natural resources locally and globally.	I.1 to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices …	Our product is meant to assist with the planning of the integration of new sustainable energies
Social Responsibility	Produce products and services that benefit society and communities.	I.2 to improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems;	Our chatbots will have a positive influence on the power utility industry, which will, in turn, have a positive influence on society.

TABLE 6

7.1.1 Social Responsibility:

One IDEALS we have performed well in is social responsibility.

- Our chatbots are meant to enhance the usefulness of GridAI in the power utility industry.
- By using computers instead of humans to perform operations, we remove human error from an essential service
- The utility industry is suffering from a shortage of engineers, so tools to reduce their workload will be beneficial

7.1.2 Communication Honesty:

One IDEALS we have not performed well in is communication honesty.

- We say the GridGPT is accurate and predictable
  - It's an LLM, it could say anything
- We communicate that GridGPT will not replace anyone's job
  - It probably will replace jobs, but also create new ones
- We claim that GridGPT is fast and easy to use
  - There is a large learning curve

7.2 Four Principles

	Beneficence	Nonmaleficence	Respect for Autonomy	Justice
Public Health, Safety, and Welfare	Improves usefulness of GridAI, to help design a more safe power system	Avoid inefficient management, of power	Let users make their own choices for design	Promotes personal benefits to all people
Global, cultural, and social	Provides safer, more reliable power management	Does not negatively affect society	Lets users make decisions based on cultural norms	Does not have social bias when decision-making
Environmental	Improved Power grouping can reduce carbon emissions.	This will likely cause an increase in carbon emission	You get to provide an eco-friendly design option	Implementation would not disturb the environment
Economic	Can improve power production efficiency, saving money	Reduces human error, which saves companies money	You don't have to buy GridGPT 2.0 or gridAI to use OpenDSS	Would not unfairly financially affect any people

Table 7

7.2.1 Beneficence to the Public Health, Safety, and Welfare:

This section refers to design that helps improve the quality of life of users. It's a benefit that we felt reflects the goals of our project the most. GridGPT 2.0 will improve the efficiency and ease of use of the more advanced GridAI. Achieving this will improve our users' experience by designing more secure and reliable power systems, achieving a higher quality of life for users and society.

7.2.2 Nonmaleficence to the Environment:

The six GPTs we plan to create will likely draw large amounts of power, which will, in turn, burn through more energy and create more carbon emissions. However, the benefits brought on by our GPTs, improving GridAI's capability to design more sustainable power flow, will outweigh the power consumption of GridGPT 2.0.

7.3 Virtues

7.3.1 Team Virtues

- Clear and Thorough Documentation
  - We were able to achieve this through frequent updates and continuous collaboration through group meetings.
- Commitment to Quality
  - We spent our first few weeks undergoing research into our project fields to ensure we were well-equipped. We also fostered team accountability, encouraging each other to provide quality work
- Perseverance
  - Though our project involved interacting with multiple programs with steep learning curves, we were resilient and overcame the challenges with growth mindsets.

7.3.2 Individual Virtues

7.3.2.1 Luke Eitzmann's Virtues

- **Accountability:** Accountability is important to me because it encourages team members to take responsibility and helps teams identify and address issues proactively. I've shown accountability in my work with programs such as OpenDSS and AltDSS, as I took accountability for learning and providing material for my team in these programs.
- **Time Management**: This virtue is important to me because it ensures steady progress and it ensures the meeting of deadlines. I have struggled this semester with time management as I've often put off my senior design work until the last minute. In the upcoming semester, I hope to prioritize my senior design work earlier in the week.

7.3.2.2 Ian Bussan's Virtues

- **Responsibility**: Responsibility is important to me because I am accountable for my actions and decisions that I have made throughout this project. In regard to this project, It was important that I felt responsible for making sure the information being provided was correct and secure so that the information was not incorrectly being used. I felt a responsibility to make this as best as possible.
- **Proactivity:** Being able to address issues and tasks before being told what to do is an important component of being a good engineer. In this project, I felt that I was being told exactly what to build every week, so there were no proactive tasks that I was creating for myself to build.

7.3.2.3 Ian Louis's Virtues

- **Cooperativeness**: This virtue is important to me because, without cooperativeness, our project will never be finished. Our team needs to cooperate to finish our project by the deadline. I have demonstrated this virtue by making sure to help my team members with their tasks and by sharing my ideas and listening to their ideas about the project.
- **Sustainability**: This virtue is important to me because I believe that creating projects with sustainability in mind will allow them to have a better and greater impact. To do this, I plan

to make decisions about how to move forward with the project with sustainability in mind. For example, the increased energy costs of creating a product that uses six different GPTs and deciding if the increased energy costs is worth the functionality that comes with six GPTs.

7.3.2.4 Aditi Nachnani's Virtues

- **Curiosity**: Curiosity is essential because it drives growth and continuous learning. I have demonstrated this in this project by performing research on new software, tools, and techniques such as Gemini. By doing this, I am able to get a deeper understanding and insights into the requirements of our project and ensure we are constantly improving it.
- **Patience:** I believe that patience is important because it allows team members to navigate through stressful situations and resolve conflicts. I find that sometimes I feel stressed out when I hit a roadblock. To demonstrate this virtue more, I will break down big tasks into smaller steps to ensure that I am managing frustration and not feeling overwhelmed.

7.3.2.5 Scott Rininger's Virtues

- **Honesty**: I believe honesty is a very important virtue since it is heavily linked to trust. I need to be honest with the team when reporting to the team, whether it is what they want to hear or not. If I am dishonest with the team, they will lose faith in me and will not trust me to do high-quality work. I have shown honesty through my communication with the team on what is and is not working.
- **Motivation**: Being motivated is an important part of working on a project as it will increase the quality and the perceived time spent on the project. It has been hard for me to find the motivation to work on this project since my other classes are more interesting and take up a lot of my time. I need to find motivation and keep it until the end of the project.

8 Closing Material

8.1 Conclusion

So far, we have created a plan for how to finish GridGPT 2.0, created an architecture of how the GPTs will operate, and started prototyping db_GPT. We have the goal of creating six GPTs that will enrich the user experience when using GridAI. The GPTs will provide context and instruction for the user of GridAI, while also providing extra features and computation. Our plan to reach our goals is to finish db_GPT, verify that it works, and then finish the remaining GPTs based on the results we determined from making db_GPT. We have yet to reach our goals because of time. When making the GPTs, we have to be competent in the subject area to make an accurate GPT. Becoming competent takes time throughout the semester. Next semester we will be able to make our goals. Something we could have done differently from the start is to better define our goals. This would have saved us time and effort from working towards the wrong objectives.

8.2 References

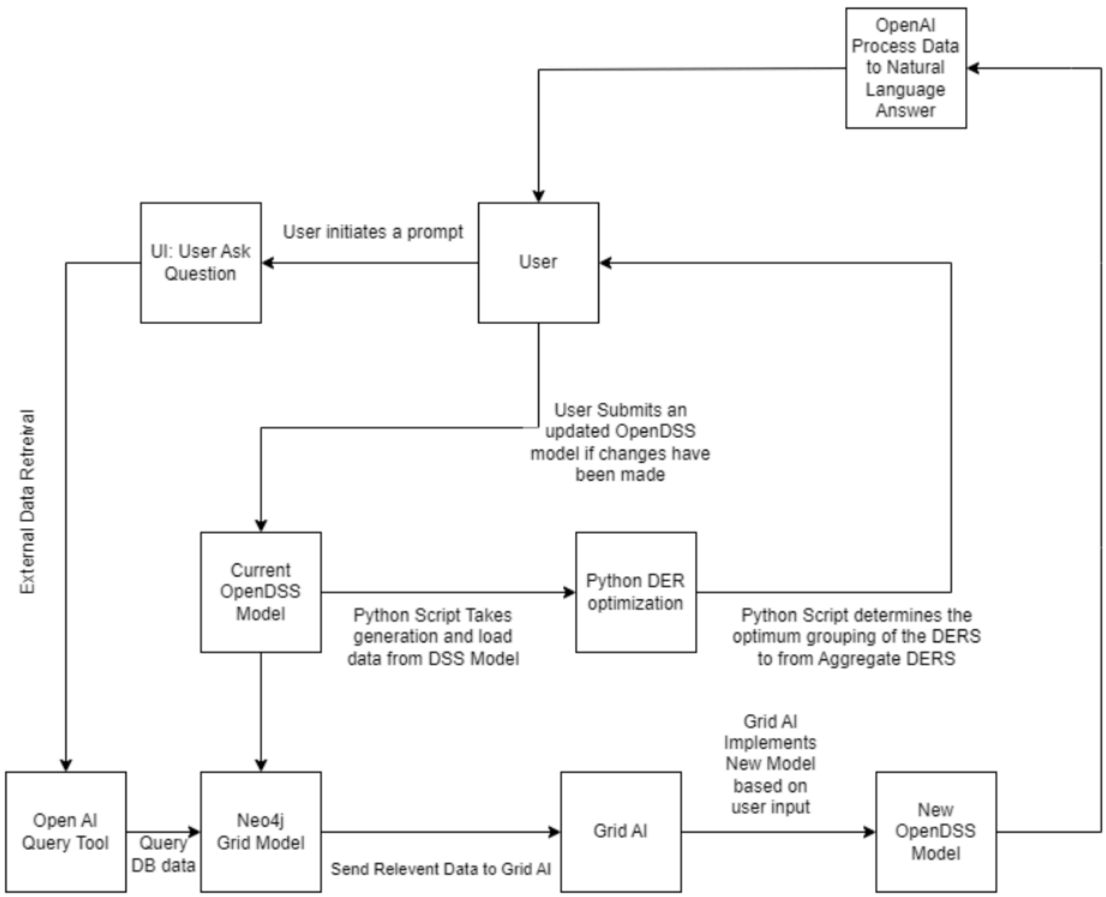
[1] R. Caldon, R. Turri, and A. R. Patria, (PDF) Optimisation Algorithm for a virtual power plant operation, https://www.researchgate.net/publication/4164357_Optimisation_algorithm_for_a_virtual_power_plant_operation (accessed Dec. 2, 2024).

[2]M. A. Salmani, S. M. M. Tafreshi and H. Salmani, "Operation optimization for a virtual power plant," 2009 IEEE PES/IAS Conference on Sustainable Alternative Energy (SAE), Valencia, Spain, 2009, pp. 1-6, doi: 10.1109/SAE.2009.5534848. keywords: {Power generation;Power supplies;Distributed control;Information technology;Power systems;Mesh generation;Power markets;Power system modeling;Power system simulation;Production;Distributed generation;Operational optimization;Virtual Power Plant;Profit maximization;Control Coordination Center;Local Controller},

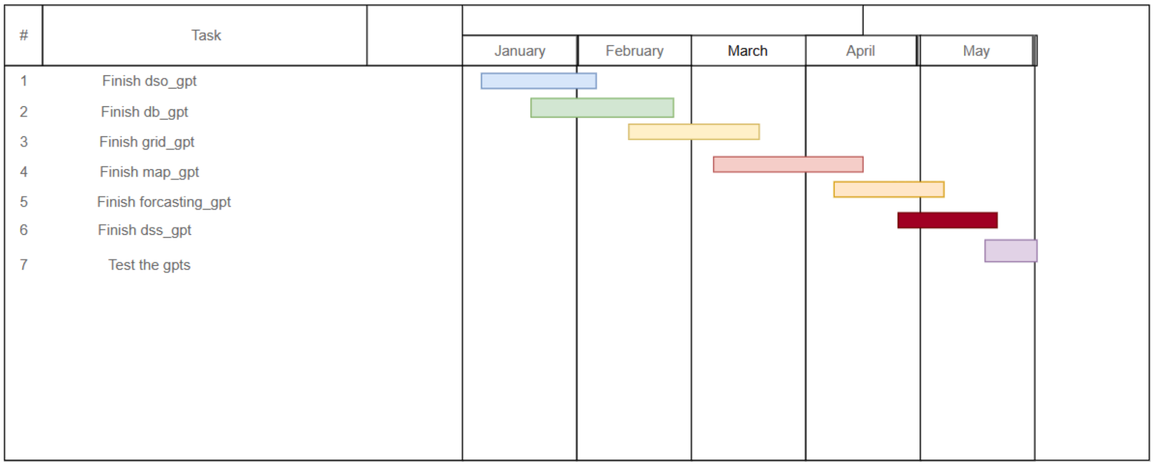
[3] G. Weishang, W. Qiang, L. Haiying, and W. Jing, “A trading optimization model for virtual power plants in day-ahead power market considering uncertainties,” Frontiers, https://www.frontiersin.org/journals/energy-research/articles/10.3389/fenrg.2023.1152717/full (accessed Dec. 2, 2024).

8.3 Appendices

GPT block diagram



Spring 2025 Plan



9 Team

9.1 Team Members

- Ian Louis
- Aditi Nachnani
- Luke Eitzmann
- Scott Rininger
- Ian Bussan

9.2 Required Skill Sets for Your Project

- Experience with Object Oriented Programming

- Knowledge of Electric Distribution Systems
- Knowledge of three-phase power
- Experience with OpenDSS
- Experience with Python
- Experience with altDSS
- Experience with GitLab
- Experience with OpenAI
- Experience with software development

9.3 Skill Sets covered by the Team

Grid Team: Ian Louis, Luke Eitzmann, Scott Rininger

- Knowledge of Electric Distribution Systems
- Knowledge of Three-phase Power
- Experience with OpenDSS
- Experience with altDSS
- Experience with Python

AI Team: Aditi Nachnani, Ian Bussan

- Experience with Object Oriented Programming
- Experience with GitLab
- Experience with OpenAI
- Experience with software development

9.4 Project Management Style Adopted by the team

We have choses a waterfall project management style. This allows us to have a big picture goal of how we will complete the project. This management style also prevents us from getting caught up in the small details instead of the big picture.

9.5 Initial Project Management Roles

Power Co-Leads

- Ian Louis, Luke Eitzmann, Scott Rininger
  - Responsible for all altDSS and OpenDSS-related tasks

AI Co-Leads

- Aditi Nachnani, Ian Bussan
  - Responsible for all OpenAI and GPT-related tasks

9.6 Team Contract

Team Members

- Ian Louis
- Aditi Nachnani
- Luke Eitzmann
- Scott Rininger
- Ian Bussan

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
   - Face to face meeting on Wednesdays, 2-3:30, Coover 2222
2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
   - SnapChat Group: SDMay25_42
3. Decision-making policy (e.g., consensus, majority vote):
   - Majority Vote
4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):
   - Reports will be drawn up immediately after our meetings while everything is fresh in our minds

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:
   - Everyone is expected to attend all meetings on time
2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:
   - Everyone takes full responsibility for any and all tasks assigned to them
3. Expected level of communication with other team members:
   - Expect team members to respond to our SnapChat group in a timely manner, and be respectful in communication.
4. Expected level of commitment to team decisions and tasks:
   - Expect each team member to contribute to their aspect of the project. Make sure to complete the required tasks before our team meeting.

Leadership

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

   Two teams:

- AI Team
    - Ian B - Full Stack AI Co-Lead, Client Interaction
    - Aditi - Full Stack AI Co-Lead
- Grid Team
    - Luke - Power Co-Lead
    - Ian Louis - Power Co-Lead
    - Scott - Power Co-Lead

2. Strategies for supporting and guiding the work of all team members:
    - Both teams work separately through the week, then combine work after meetings on Wednesdays
3. Strategies for recognizing the contributions of all team members:
    - Each team is responsible for making sure its members are contributing to the project

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.
    - Ian Louis - Internship Experience in the Power Industry and Power Systems and Programming Coursework
    - Ian Bussan - Full stack development experience, ML and GPT model work experience
    - Luke Eitzmann - Multiple Internships in the power distribution industry with experience with distribution automation and distribution system modeling
    - Aditi Nachnani - Internship experience in full stack development and a research assistant for a ML project
    - Scott Rininger - Two internships in the power industry with experience in substation design
2. Strategies for encouraging and supporting contributions and ideas from all team members:
    - Aditi Nachnani: provide any help I can if a team member needs it and encourage team members to ask for help and share if they are feeling stressed out
    - Ian Bussan: help coordinate with the team, and be helpful to other team members.

- Luke Eitzmann: Bringing a positive attitude and helping carry the workload.
- Scott Rininger: Being available to help with the workload and support other team members
- Ian Louis: Being respectful of everyone's ideas and having an open mind to new ideas

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?)
    - Members can bring up the issue during weekly meetings or in the group chat. Then they can express the problem they are facing, and the team will come to an understanding and an agreement.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:
    - Learn more about AI and power systems. Enhance team communication and development skills.
2. Strategies for planning and assigning individual and teamwork:
    - Assign tasks to group members with the skills needed to complete the task. Ensure that tasks are equally split between team members to prevent any one person from having too much work.
3. Strategies for keeping on task:
    - Communication about keeping team members accountable and on time.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?
    - Let the team member know that they are not following the obligations of the contract and what they need to fix their behavior.
2. What will your team do if the infractions continue?
    - Inform one of our professors that our team member is not following the team contract and ask our professor how we should resolve the situation.

****

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) Aditi Nachnani DATE 12/2/2024

2) Ian Louis DATE 12/2/2024

3) Luke Eitzmann DATE 12/2/2024

4) Scott Rininger DATE 12/2/2024

5) Ian Bussan DATE 12/2/2024