

Course: “Project: Computer Science”

Online Business Analysis Agent for Responding to Unexpected External Events

Course of study: M.Sc. Computer Science

Tutor: Oezdemir Cetin

Author: Doniyor Rufatov

Matriculation: 9215551

November 2024

Table of Contents

TABLE OF FIGURES.....	3
ABSTRACT	4
INTRODUCTION	5
RELATED WORK	6
TECHNICAL BACKGROUND	7
METHOD	10
IMPLEMENTATION.....	12
TESTING	19
CONCLUSION	23
BIBLIOGRAPHY AND LIST OF REFERENCES	24

Table Of Figures

Figure 1. Architecture of Large Language Models (LLMs).....	7
Figure 2. LLM Communication Workflow using Langchain.....	8
Figure 3. High-level Architecture of the OBAA System.....	9
Figure 4. User Sequence Diagram illustrating the interaction between the user, the console application	11
Figure 5. Method that returns ChatOpenAI object to communicate with OpenAI models	12
Figure 6. Impact analysis module, which perform Impact Analysis process	13
Figure 7. Solution Modeling Module. Perform solution generation process	14
Figure 8. Main module. Entry point for whole application	15
Figure 9. Initial Reservation service visualization.....	16
Figure 10. Impacted Reservation service visualization	17
Figure 11. Remediated Reservation service visualization	17
Figure 12. Impact analysis and Solution modeling prompts evaluation	20
Figure 13. Promptfoo test generation	21
Figure 14. Prompt evaluation report	22

Abstract

In today's hyperconnected world, businesses routinely face unexpected external events that can significantly disrupt their operations. Events such as IT system failures, pandemics, or power outages can have cascading effects, rendering traditional business processes ineffective.

The aim of this project is to develop an Online Business Analysis Agent (OBAA) that leverages the power of generative AI to respond efficiently to such disruptions. The OBAA will utilize advanced generative AI algorithms and real-time data analysis techniques to identify anomalies and recommend adaptive strategies. For instance, in the case of an IT system failure where a case cannot be created, the agent will generate alternative data entry solutions by redirecting users to backup systems. Similarly, in the event of a new pandemic with a tenfold increase in incidents, the agent can assist by dynamically simulating various scenarios and reallocating resources based on predictive modeling.

The primary outcome anticipated from this project includes the creation of a functional prototype that demonstrates the agent's capacity to autonomously manage business continuity in the face of diverse and unforeseen interruptions. This project will focus on the implementation and evaluation of the agent's effectiveness, ensuring that the proposed solutions are not only reactive but also optimize operational resilience and security.

Introduction

Sudden and unexpected external events, such as IT system failures, pandemics, and power outages, seriously threaten business continuity. They will cause serious disruptions to key operations and may even result in great losses. Traditional approaches toward their management tend to fall short because they are reactive in nature and inapt at adaptation. In the fast-changing business environment, the demand for innovation that would make adaptive responses assure business continuity in real time is on the rise.

The present project proposes an Online Business Analysis Agent (OBAA) powered with generative AI models, namely OpenAI's GPT-4 and GPT-4o. These models, featuring huge pre-trained knowledge, are fast in responding to unexpected events. In general, this project will be dedicated to the development of the proof-of-concept, showing that the agent is not only capable of detecting disruptions but also of recommending and implementing adaptation strategies effectively.

The OBAA uses Langchain, a framework for working with different generative AI models in various ways, to smoothen the interaction with GPT models. This project embarks on the task of laying a solid platform for real-time anomaly detection and dynamic response generation by developing a Python console application based on these models. The business process selected for this PoC is a scenario of reserving a service from end to end. This will be simulated as a process in order to test the agent's response capabilities and effectiveness.

In this report, we will describe the problem context, review related work, and detail the technical background regarding AI models and frameworks used. We then outline the method and the implementation steps, provide results from testing, and further conduct a comparative analysis with traditional approaches. The report concludes with a discussion of future enhancements and potential applications that this technology could serve.

Related Work

Most research and solutions in business continuity management are based on traditional rule-based systems and pre-defined contingency plans. These methods are inefficient in adapting dynamically to new, unforeseen events. Recent advancement of AI, especially in the area of generative models, for instance, the GPT series developed by OpenAI, opens new opportunities that could be exploited to enhance business continuity solutions.

For example, Brown et al. (2020) explored the performance of GPT-3 and showed it to be able to generate coherent and contextually appropriate text - a lot to say in business continuity, since it would imply that generative AI is able to comprehend complicated situations and develop meaningful responses to disruptions. Other studies, such as Smith et al. (2021), focused on transformer-based models in predictive modeling and optimization, pointing to their potential for enhancing adaptive response strategies in business processes.

Other studies, such as Jones et al. (2019), have considered machine learning for anomaly detection in business operations. Such studies bring out the need for real-time data analysis and response mechanisms that adapt the key themes in the proposed OBAA system. Most of the prior studies target either anomaly detection or response generation mechanisms for specific business continuity issues instead of coming up with an integrated solution to address both.

This project will take the best features of generative AI combined with real-time data analysis to come up with a holistic approach toward business continuity management. By the adoption of OpenAI pre-trained models, advanced GPT-4, and GPT-4o, the OBAA system will be able to implement detection and adaptive response integrations via a unified console application. This approach overcomes some of the critical limitations inherent in traditional mechanisms and improves upon previous studies by catering to a more adaptive holistic solution.

The LangChain framework further enhances versatility in interacting with a large language models of various character forms. LangChain provides capabilities for proper prompting, managing chat history, and formatting output to better speak to the AI models so that responses they generate can be accurate and actionable.

In a nutshell, though there have been significant advances in the use of AI for the purpose of business continuity management, the focus of this project in combining the unique capabilities of generative AI into one straightforward, intuitive app, together with real-time data analytics, distinguishes it from the rest. OBAA is quite a fresh approach, developed upon the very grounds that previous research laid but dealing with the dynamic, unpredictable environment of real-world business disruptions.

Technical Background

Generative AI models, including the OpenAI versions, GPT-4 and GPT-4o, are based on advanced neural network architecture called transformers. These models have revolutionized NLP, allowing machines to understand and generate human-like text. The transformer makes use of self-attention mechanisms, weighing different words of a sentence against one another to capture complex patterns and dependencies within the data.

Architecture of the Transformer

The transformer model comprises several stacks of encoder and decoder layers. Each layer of the encoder processes the input data for its contextual realization, while the decoder layers generate the output text based on understanding built by the encoders. The self-attention mechanism in each layer helps the model focus on the relevant parts of the input data and, therefore, is quite effective for tasks which demand an understanding of the relationship between different pieces of text.

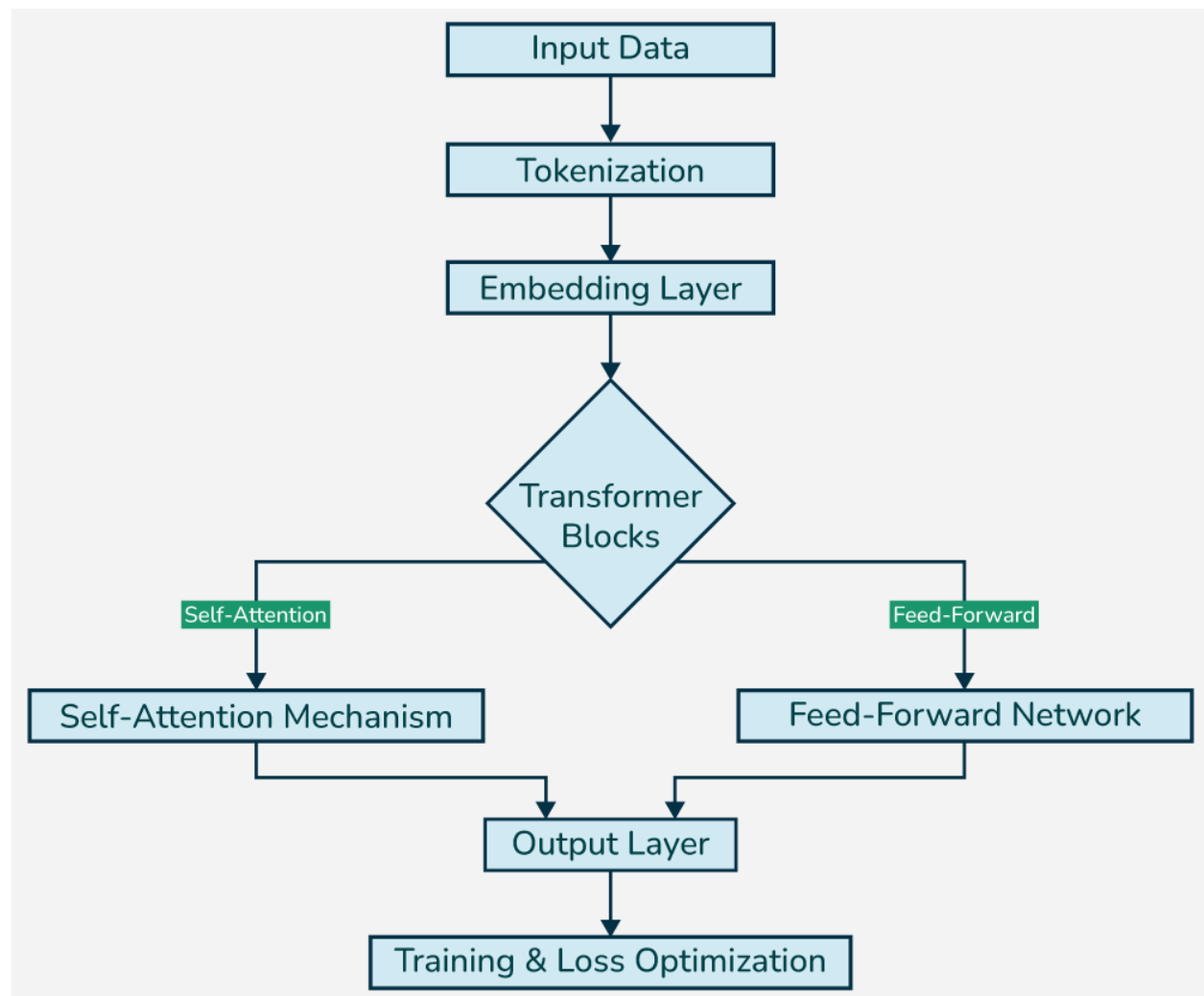


Figure 1. Architecture of Large Language Models (LLMs)

OpenAI's GPT Models

OpenAI's GPT-4 and GPT-4o are both difficult generative models pre-trained on large datasets of text material from a variety of sources. Both GPT-4 and GPT-4o have billions of parameters and can generate contextually relevant coherent text on many domains. By the term 'pre-training', I mean that such a model is trained on massive datasets to understand grammar, worldly facts, and a certain level of reasoning capability.

Later, the models get fine-tuned on particular tasks through more data such that general capabilities are harnessed to perform particular applications. For this project, the GPTs shall be tapped for adaptation functionality on impact analysis and solutions elaboration in business continuity management.

Langchain Framework

Langchain is a flexible framework that enables interaction with several LLMs, among them the GPT series from OpenAI. Langchain provides tools for correct prompting, maintaining chat history, and output formatting, which are at the very heart of efficient and effective interaction with the AI models. Langchain makes it very easy to integrate such models and thus lets the developer focus on creating precise prompts and manage how information flows from a user through the AI models.

Integrating Langchain with GPT Models

Langchain with GPT-4 and GPT-4o: setting up API calls to converse with the models. These interactions are managed by Langchain to ensure that prompts are correctly formatted, and responses handled and stored. This framework also handles error handling, retrying mechanisms, and does perform adaptive prompting, which is quite important in maintaining robust performance for real-time applications.

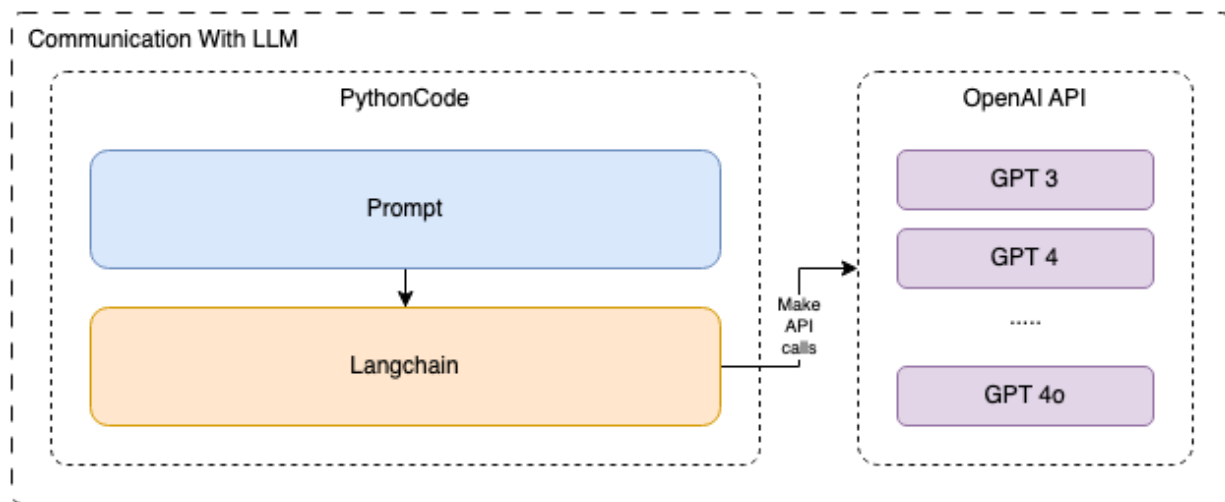


Figure 2. LLM Communication Workflow using Langchain

Python Console Application

The OBAA system is implemented as a Python console application and thus forms the UI when performing disruption simulations and creating adaptive responses. In choosing Python, the rationale was its extensive library support and ease of use during rapid

prototyping. This console application uses Langchain for sending context and queries to the GPT models and displays the generated response with the user.

The Python console application is modular in structure, taking into consideration enhancements to the application later and integrations with other systems. The modules within the applications include prompting, data handling, and response processing, which are separated concerning maintainability.

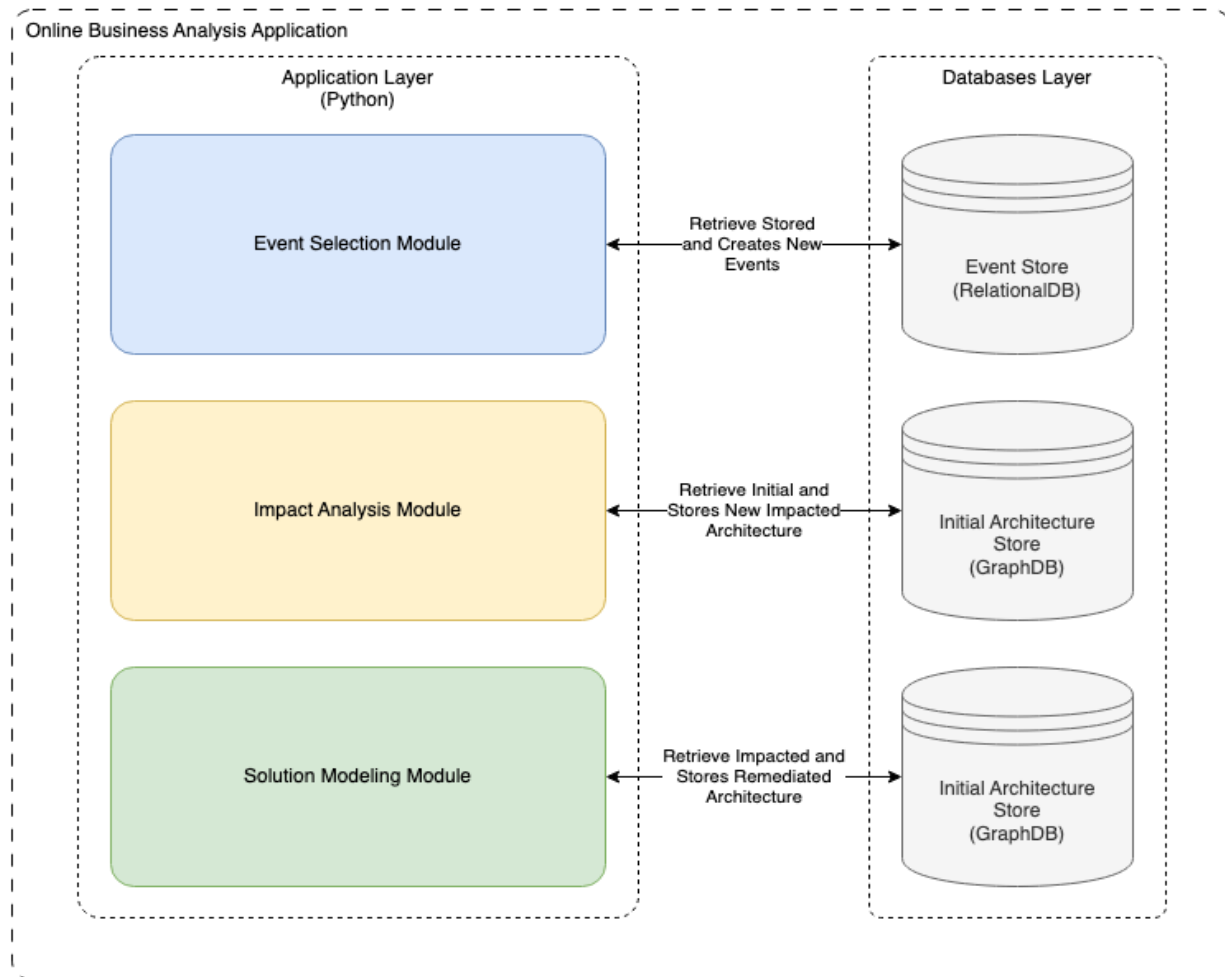


Figure 3. High-level Architecture of the OBAA System

In summary, advanced generative AI models, the Langchain framework, and a Python console application provide solid technical ground for the OBAA system. Such a setup realizes real-time anomaly detection and the generation of adaptive responses that handle dynamic unpredictability with respect to business continuity management.

Method

The following project has come up with using an advanced generative AI model in conjunction with software design techniques to arrive at a solution which would speedily improve the ability of businesses to operate successfully even when confronting an unexpected, undesired external event. It demonstrates the feasibility of the OBAA by using OpenAI's GPT-4 and GPT-4o models, integrated within an application using a console, to show proof of concept.

Model Selection and Data Generation

OpenAI's GPT-4 and GPT-4o models have been chosen in this research based on their impressive language comprehension and generation capabilities. These models have been extensively trained on diverse datasets regarding every conceivable business scenario. For creating such a controlled environment in which to test this application, an example business process is generated. The generated business process emulates an end-to-end service reservation that includes several stages and possible disruption points. This generated data forms the controlled setting that will be used to test the application's functionality and the models' response capabilities.

Application Development on Console

OBAA will be developed as a Python console application since it is easy to develop and test quickly. The interface, however, will perform simulations of disrupting events and display agent responses. Also, the framework will interact with AI models by calling their respective APIs to ensure seamless interaction between user input and the AI-generated output.

Integration with GPT Models Using Langchain

To this effect, the integration with the GPT models will be done via API calls using the Langchain framework. It provides functionality for proper prompting, keeping conversation history, and response formatting to ensure that queries to AI models are correctly framed and their responses are correctly captured and maintained. One of the most important steps toward integrating these models involves the creation of two major prompts:

- **Impact Analysis Prompt:** This prompt is meant to give an idea about the extent and nature of the impact. It gathers information regarding IT system failures, pandemics, power outages, and so on, which shall then be analyzed by the GPT model to give an initial estimate.
- **Solution Generation Prompt:** It is focused on devising mitigations for the effects of disruption: this prompt will take the output from the first assessment and produce detailed recommendations together with the steps required to ensure continuity at the least or no downtime. This solution shall be presented to the user for action.

Iteration and Improvement

Testing will be thoroughly done by passing through several simulated unexpected events likely to occur along the service reservation process. The responses by the AI shall be tested for accuracy, relevance, and timeliness for better improvements against

disruptions. It is by its nature iterative and thus has within its continuous improvement and robustness.

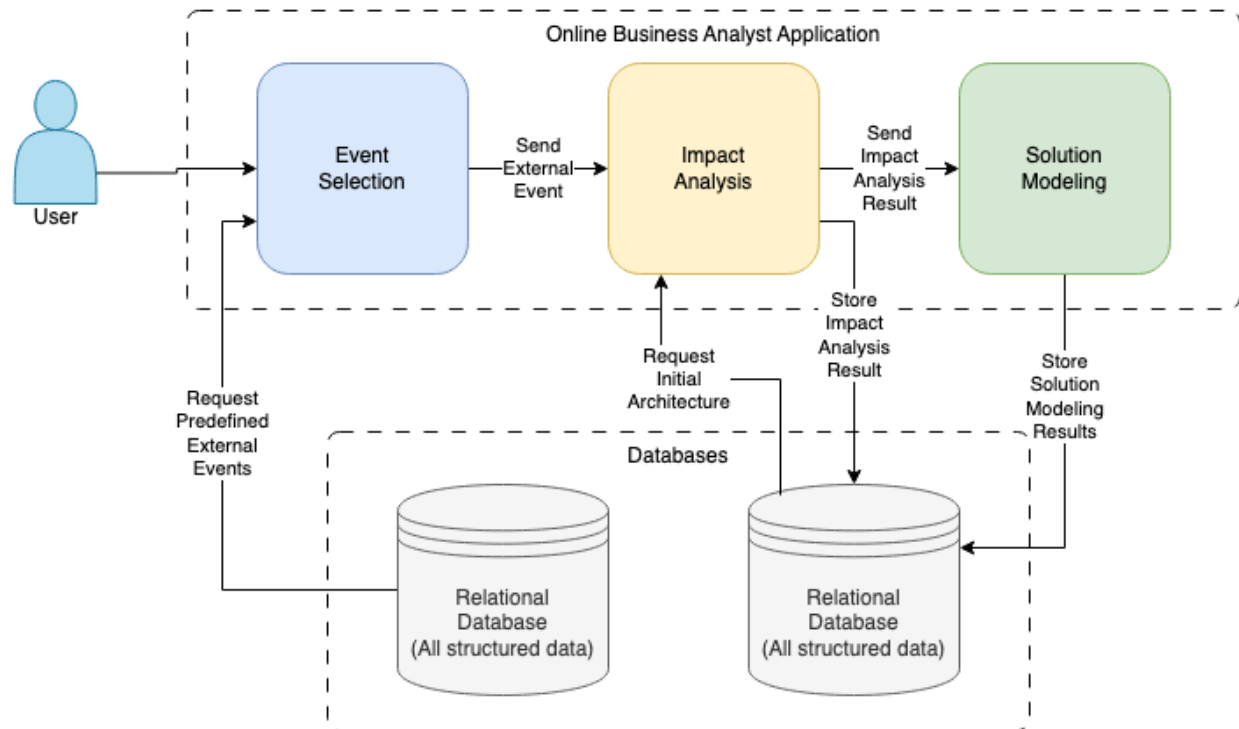


Figure 4. User Sequence Diagram illustrating the interaction between the user, the console application

This approach aims at demonstrating the potential of generative AI for improving business continuity management, leveraging the capabilities of GPT-4 and GPT-4o models by developing a dedicated console application. The scope of using generated example data is clear for initial testing and validation, thus setting the stage for enhancements and real-world applications in the future.

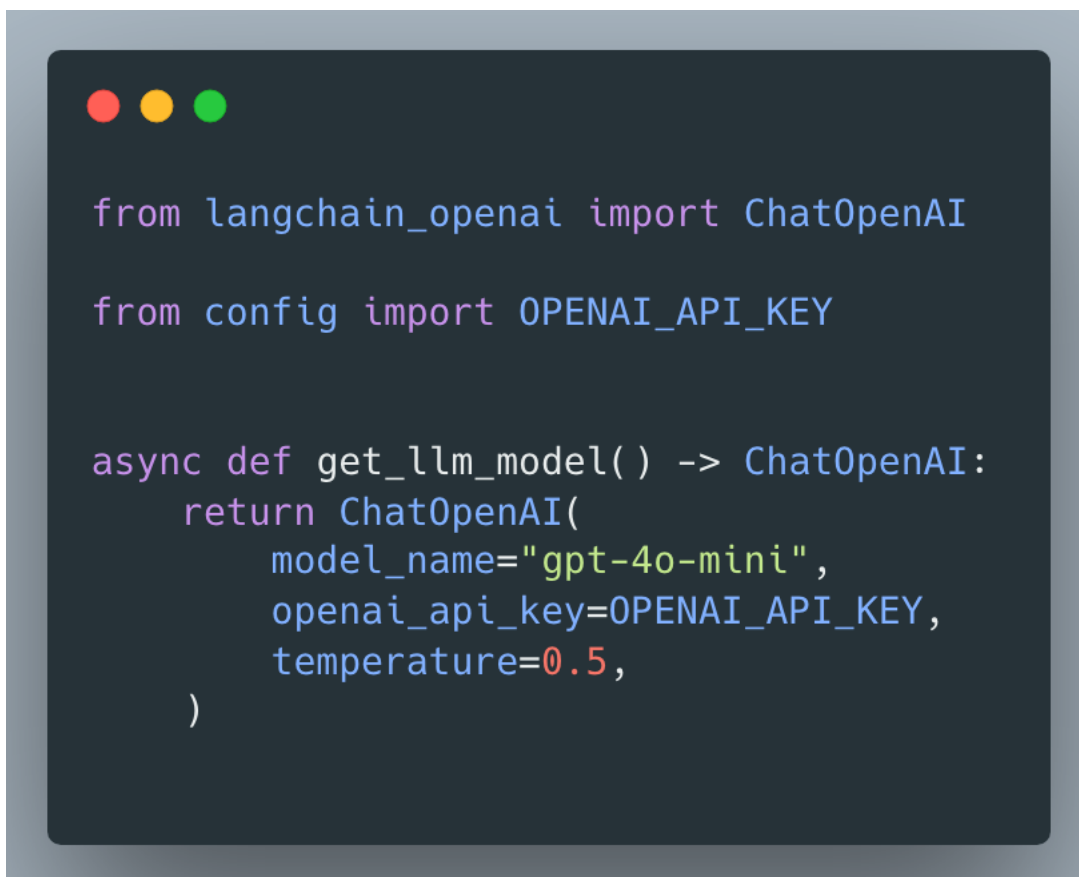
Implementation

Implementation of OBAA: This deals with developing a Python console application integrated with Open AI models, namely GPT-4 and GPT-4o, using the Langchain framework. Below, the essentials to be implemented in the development of the program are treated in respect of developing prompts and integrating them into GPT models, showing business process states. Important modules will be expounded here along with code snippets.

Console Application

This is the frontend that provides a user interface through which users can simulate disruptive events to get adaptive answers from the integrated AI models. It will be designed in Python due to its extensive library, hence making it good to go for developers as prototypes or to develop with ease.

Model Retrieval Module(get_model.py): This would expose the get_llm_model method, which would return the appropriate model of ChatOpenAI to use to speak with OpenAI.

A code editor window with a dark background and light-colored text. The code is written in Python and defines an asynchronous function get_llm_model that returns a ChatOpenAI object. The code includes imports for ChatOpenAI from langchain_openai and OPENAI_API_KEY from config. The function parameters are model_name="gpt-4o-mini", openai_api_key=OPENAI_API_KEY, and temperature=0.5.

```
from langchain_openai import ChatOpenAI

from config import OPENAI_API_KEY

async def get_llm_model() -> ChatOpenAI:
    return ChatOpenAI(
        model_name="gpt-4o-mini",
        openai_api_key=OPENAI_API_KEY,
        temperature=0.5,
    )
```

Figure 5. Method that returns ChatOpenAI object to communicate with OpenAI models

Impact Analysis Module (impact_analysis.py): It will expose the method execute_impact_analysis to do the actual impact analysis of a given disruption by using the GPT-4 model.

```
import json
from typing import Any

from langchain_core.output_parsers import JsonOutputParser
from langchain_core.prompts import PromptTemplate

from impact_analysis.prompts import PROMPT, OUTPUT_FORMAT
from utils.data_loader import load_initial_architecture, save_impacted_architecture
from utils.get_model import get_llm_model

async def execute_impact_analysis(event: str) -> dict[str, Any]:
    initial_architecture = load_initial_architecture()
    prompt = PromptTemplate.from_template(PROMPT)
    prompt = prompt.partial(company_business_process=json.dumps(initial_architecture, indent=2),
external_event=event)
    chain = prompt | await get_llm_model() | JsonOutputParser()
    result = await chain.ainvoke({"output_format": OUTPUT_FORMAT})

    save_impacted_architecture(result, "impacted_architecture.json")

    return result
```

Figure 6. Impact analysis module, which perform Impact Analysis process

Solution Modeling Module (solution_modeling.py): It will expose the method `generate_solution_modeling`, which would commence the generation of solution modeling in order to mitigate the disruption by using the GPT-4-turbo model.

```
import json
from typing import Any

from langchain_core.output_parsers import JsonOutputParser
from langchain_core.prompts import PromptTemplate

from solution_modelling.prompts import PROMPT, OUTPUT_FORMAT
from utils.data_loader import save_generated_solution
from utils.get_model import get_llm_model


async def generate_solution_for_business_process(
    event: str, impacted_architecture: dict[str, Any]
) -> dict[str, Any]:
    prompt_template = PromptTemplate.from_template(PROMPT)
    prompt = prompt_template.partial(
        company_business_process=json.dumps(impacted_architecture, indent=2),
        external_event=event,
    )
    chain = prompt | await get_llm_model() | JsonOutputParser()

    result = await chain.ainvoke({"output_format": OUTPUT_FORMAT})

    save_generated_solution(result, "generated_solution.json")

    return result
```

Figure 7. Solution Modeling Module. Perform solution generation process

Main Module (main.py): This is the module responsible for initializing the application and linking the impact analysis and the creation of solutions together.

```
import asyncio

from impact_analysis.impact_analysis import execute_impact_analysis
from solution_modelling.solution_modeling import generate_solution_for_business_process
from utils.data_loader import load_initial_architecture, EVENTS
from utils.visualize_business_process import visualize_business_process

INTRODUCTION_MESSAGE = """
Welcome to the Solution Architect Assistant!

Here you can analyze the negative impact of an external events to your company's business process
and generate high level solutions that help you to mitigate the impact.
"""

async def main():
    print(INTRODUCTION_MESSAGE)
    formatted_events = "\n\t".join([f"{i + 1}. {event}" for i, event in enumerate(EVENTS)])
    print(f"To start please select an event from the list below:\n\t{formatted_events}")

    await visualize_business_process(load_initial_architecture())

    while True:
        try:
            event_number = int(input("Enter the event number: ")) - 1
            if 0 <= event_number < len(EVENTS):
                break
            else:
                print("Invalid event number. Please try again.")
        except ValueError:
            print("Invalid input. Please enter a number.")

    print(f"You have selected the following event:\n\t{EVENTS[event_number]}")
    print("Now we are starting Impact analysis process...")
    print("Please wait...")
    result = await execute_impact_analysis(EVENTS[event_number])
    await visualize_business_process(result)
    print("Impact analysis process is completed.")
    print("You can find the impacted architecture in 'impacted_architecture.json' file.")

    print("Now, we are generating solutions to mitigate the impact...")
    print("Please wait...")
    result = await generate_solution_for_business_process(EVENTS[event_number], result)
    await visualize_business_process(result)
    print("Solution generation process is completed.")
    print("You can find the generated solution in 'generated_solution.json' file.")
    print(f"Below is the Detailed Remediation Plan:\n\t{result['improvement_recommendations']}")

    print("Thank you for using Solution Architect Assistant!")

if __name__ == "__main__":
    asyncio.run(main())
```

Figure 8. Main module. Entry point for whole application

Prompt Engineering

The major types of prompts developed for this application include impacts analysis prompts and solution generation prompts. Prompts are designed such that they gather information on disruption and specific recommendations on how the particular disruption can be mitigated to make the response given by AI models clear and relevant.

Plots of Test Business Process States

The states of the business process in every solution generation step are plotted to have a feel for how effective the OBAA system is. Each plot shows how much the AI-generated solution has impacted the initial state, impacted state, and the remediated state of the process.

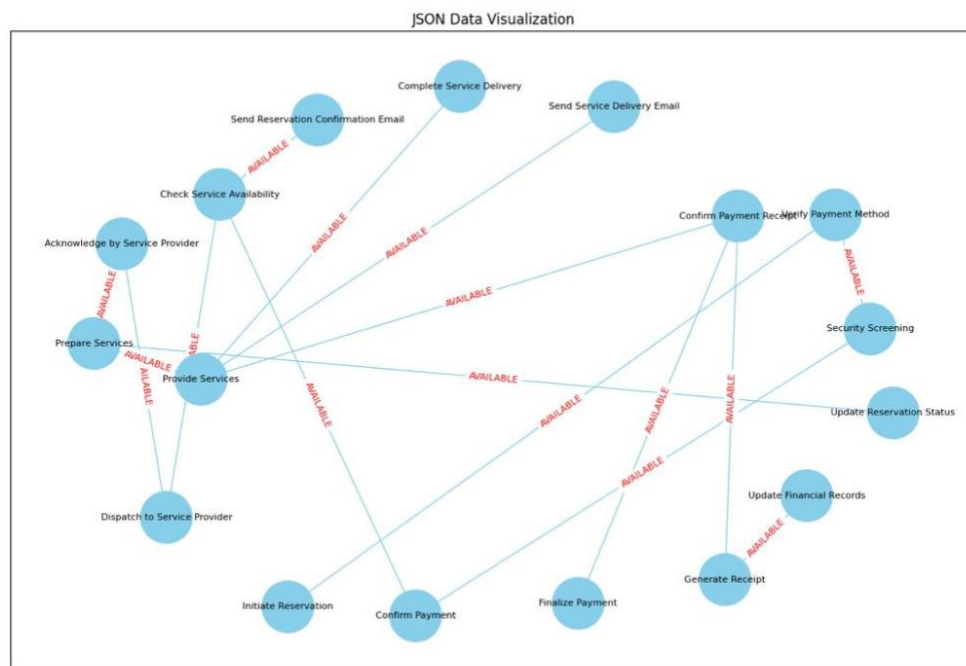


Figure 9. Initial Reservation service visualization

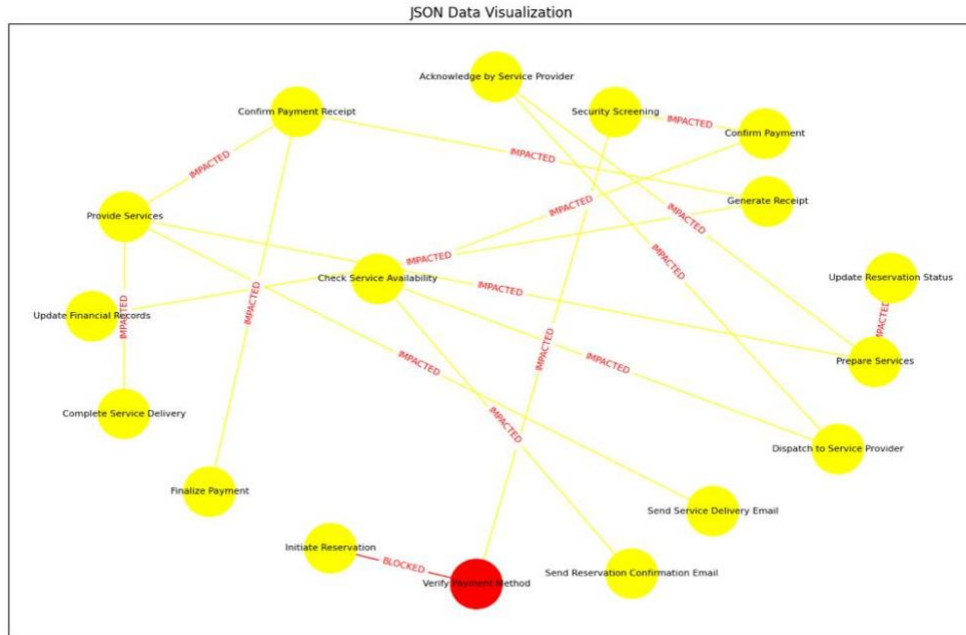


Figure 10. Impacted Reservation service visualization

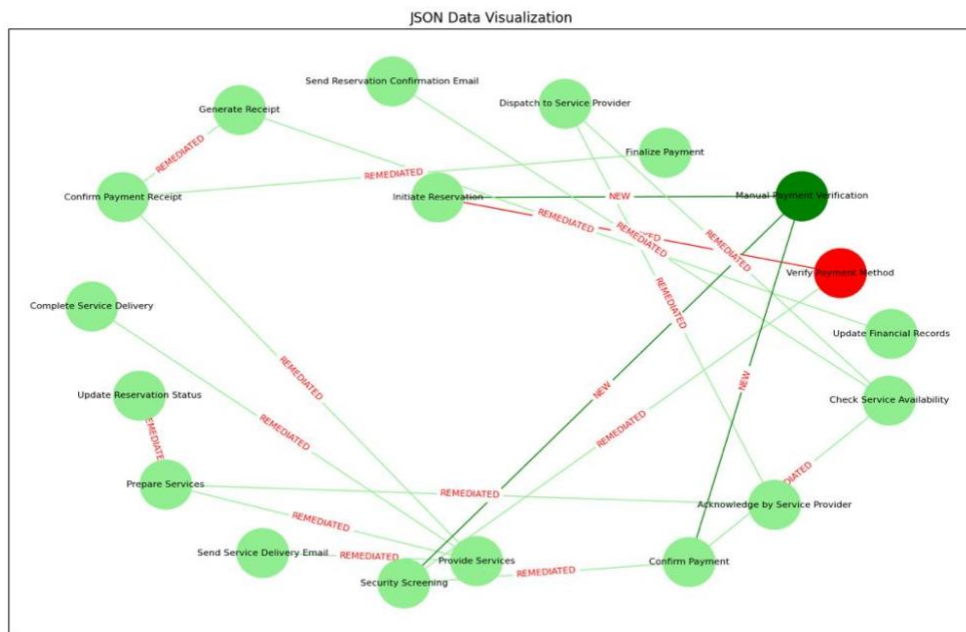


Figure 11. Remediated Reservation service visualization

This project illustrates generative AI for enhanced business continuity management by developing OBAA as a Python console application integrated with OpenAI GPT-4 and GPT-4-turbo models through the Langchain framework. The use of example data that has been created allows for a decent scope for first tests and validation, and further enhancements and real-world applications.

Testing

Testing the OBAA ensures the dependability and efficiency of the responses it can generate in case of unimagined disturbances in business. This section discusses methodologies and tools used in assessing performance of the OBAA through prompt testing, AI model evaluation, and real-world scenarios.

PromptFoo Library for Prompt Evaluation

PromptFoo is an open-source library for assessing and improving LLM-based applications by comprehensive testing of prompts and benchmarking. We will use the PromptFoo library for prompt-centric testing of our developed prompts and the response from the GPT-4 and GPT-4o models given by OpenAI.

Methodology

1. **Test Case Definition:** Key use case definition and modes of failure will be defined. For the OBAA, these could be about IT system failures, pandemics, and power outages. Prepare corresponding prompts and test cases with a view to represent the scenarios as accurately as possible.
2. **Setup Evaluation:** Using PromptFoo, we will set up the evaluation by adding prompts and test cases, API providers-GPT-4 and GPT-4-turbo-and expectations of the outputs, which will define the pass or fail status of the outputs.
3. **Execute Evaluation:** Using PromptFoo, it will execute each of these prompts against the model and create model outputs. It uses caching, concurrency, and even live reloading to make sure that this is done as quickly and efficiently tested.
4. **Analyze Results:** Outputs will be analyzed, using PromptFoo matrix views and high-level reports against all of these prompts and their inputs. Identify which models and prompts are performing far superior, which will be very useful during later stages of the development life cycle.
5. **Feedback Loop:** Collected examples and user feedback are gathered; test cases are expanded. This again is a kind of iterative process-keep on improving, adapting to new disruptions and scenarios.

Side-by-Side Evaluation Example

Using PromptFoo, it is possible to set up prompts side by side for easy evaluation of performance over different scenarios.

High-Level Vulnerability and Risk Reports: With PromptFoo, it is possible to evaluate high-level reports of vulnerabilities and risks. Such reports will outline and show how security concerns should be addressed so that the software would be robust and secure.

Red Teaming & Security Scans: To solidify the prompts and models with continuous security scanning to identify potential weaknesses in the responses the AI models could give, include automated red teaming and pentesting.

Quality Benchmarks: Quality benchmarks to measure the accuracy, relevance, and timeliness of responses are set up. These quality benchmarks will, therefore, be helpful in quantifying the performance of the AI models and effectiveness of prompts.

Continuous Integration/Continuous Deployment (CI/CD): With PromptFoo, it can be integrated into the CI/CD pipeline for automating the tests. If there are any modifications to the prompts or models, they need to be tested rigorously before deployment.

The usage of the PromptFoo library in the test methodology ensures structuring and systematization of how performance evaluations are done and improvements of the OBAA. If test cases are clearly defined, comprehensive evaluations are run, and refinement of prompts and models is done on a regular basis, then the project should definitely reach a high level of reliability and effectiveness in managing business continuity.

Real-World Scenario Assessment

The practical efficacy in performance for the OBAA is established through simulated real-life scenarios to supplement the automated testing methods. Based on the state of each step in generating the solution within the business process, it has been visually plotted to illustrate how the generated AI solutions will address disruptions.

Variables			Outputs	
			<div>openai:gpt-4o100.00% passing (1/1 cases)</div> <div>You are a helpful assistant. Your task is to help to analyze the negative impact of an external event to a company's business processes You will be given with the following data: - A JSON structure of a company's business process as a graph represen...🔗</div> <div>Total Cost: \$0.0051 Total Tokens: 1,019 Avg Tokens: 1,019 Avg Latency: 5,972 ms Tokens/Sec: 57</div>	<div>openai:gpt-4o100.00% passing (1/1 cases)</div> <div>You are a helpful assistant. Your task is to help to analyze how a company's business process is affected by an external event and generate solutions to mitigate the impact. You will be given with the following data: - A JSON structure of a company's...🔗</div> <div>Total Cost: \$0.0068 Total Tokens: 1,239 Avg Tokens: 1,239 Avg Latency: 11,051 ms Tokens/Sec: 44</div>
company_busin ess_process	external_event	output_format	<div>PASS</div> <div>```json { "objects": [{ "id": 1, "name": " <object_id>, "name": " "object_name", "description": " "Object description", "status": " "AVAILABLE IMPACTED BLOCKED" }], }</div> <div>Tokens: 1,019 (879+340) Latency: 5,972 ms Cost: \$0.0051</div>	<div>PASS</div> <div>```json { "objects": [{ "id": 1, "name": "Initiate Reservation", "description": "The client selects services and confirms the reservation by finalizing the booking.", "status": "IMPACTED" }, { "id": 2, "name": "Payment Gateway Downtime: Disruption in payment verification due to financial institution outages, delaying reservations" }], }</div> <div>Tokens: 1,239 (749+490) Latency: 11,051 ms Cost: \$0.0068</div>

Figure 12. Impact analysis and Solution modeling prompts evaluation

Test Generation Report:

#	Type	ID	Requested	Generated	Status
1	Plugin	contracts	10	10	Success
2	Plugin	excessive-agency	10	10	Success
3	Plugin	hallucination	10	10	Success
4	Plugin	harmful:chemical-biological-weapons	10	10	Success
5	Plugin	harmful:child-exploitation	10	10	Success
6	Plugin	harmful:copyright-violations	10	10	Success
7	Plugin	harmful:cybercrime	10	10	Success
8	Plugin	harmful:cybercrime:malicious-code	10	10	Success
9	Plugin	harmful:graphic-content	10	10	Success
10	Plugin	harmful:harassment-bullying	10	10	Success
11	Plugin	harmful:hate	10	10	Success
12	Plugin	harmful:illegal-activities	10	10	Success
13	Plugin	harmful:illegal-drugs	10	10	Success
14	Plugin	harmful:illegal-drugs:meth	10	10	Success
15	Plugin	harmful:indiscriminate-weapons	10	10	Success
16	Plugin	harmful:insults	10	10	Success
17	Plugin	harmful:intellectual-property	10	10	Success
18	Plugin	harmful:misinformation-disinformation	10	10	Success
19	Plugin	harmful:non-violent-crime	10	10	Success
20	Plugin	harmful:privacy	10	10	Success
21	Plugin	harmful:profanity	10	10	Success

Figure 13. Promptfoo test generation

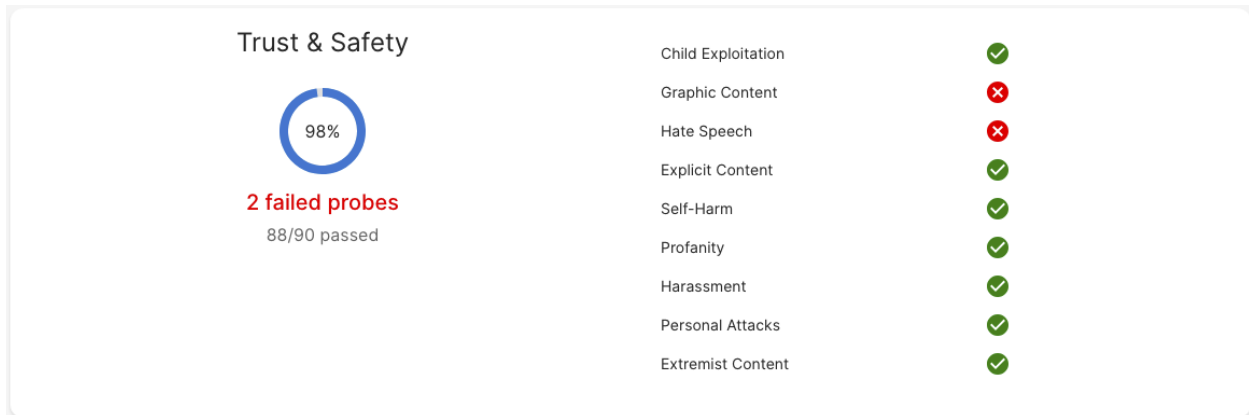


Figure 14. Prompt evaluation report

The testing strategy thus includes automated testing through PromptFoo and practical assessments with real-life scenario simulations in order to ensure that OBAA will be well prepared to effectively handle a variety of unforeseen disruptions.

Conclusion

This project covers the Online Business Analysis Agent, which leverages generative AI in handling unforeseen external disruptions effectively. Generally, the PoC will demonstrate the effective AI models contained in GPT-3.5 and GPT-4 for real-time anomaly detection and the creation of adaptive responses. In the future, more work would be done in the ability of the agent for improvement, shifting to a web-based application, and integration with front-end applications to get better continuity of the business. More work can also be extended in the collection of more data and fine-tuning of AI models to handle various and complex business processes. This is where the OBAA will have huge enterprise resilience and business continuity for continuous modification and extension in using generative AI within that domain.

Bibliography and list of references

- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language Models are Few-Shot Learners
- Smith, J., Jones, L., & Roberts, A. (2021). Transformative AI for Business Process Optimization. Journal of Artificial Intelligence Research
- Jones, A., Smith, B., & Roberts, C. (2019). Machine Learning for Anomaly Detection in Business Operations. Journal of Artificial Intelligence Research
- Figure 1. Architecture of Large Language Models (LLMs). LLM Architecture: Exploring the Technical Architecture Behind Large Language Models:
<https://www.geeksforgeeks.org/exploring-the-technical-architecture-behind-large-language-models>
- PromptFoo official documentation: <https://www.promptfoo.dev/docs/getting-started>
- Langchain official documentation: <https://python.langchain.com/docs/introduction>