Mini-Project Report On

Slides Genie (An AI Learning Tool that generates presentations)

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science & Engineering

By

Raphael Tony (U2003162)
Reuben Dinny (U2003166)
Neha Bimal (U2003147)
Shreya Baburaj (U2003200)

Under the guidance of
Ms. Anu Maria Joykutty

Department of Computer Science & Engineering
Rajagiri School of Engineering and Technology (Autonomous)
Rajagiri Valley, Kakkanad, Kochi, 682039

July 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
RAJAGIRI SCHOOL OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
RAJAGIRI VALLEY, KAKKANAD, KOCHI, 682039

CERTIFICATE

This is to certify that the mini-project report entitled ***"Slides Genie (An AI Learning Tool that generates presentations)"*** *is a bonafide work done by* ***Mr. Raphael Tony (U2003162), Mr. Reuben Dinny (U2003166), Ms. Neha Bimal (U2003147), Ms. Shreya Baburaj (U2003200)****, submitted to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in Computer Science and Engineering during the academic year 2022-2023.*

Dr. Preetha K. G.
Head of Department
Dept. of CSE
RSET

Ms. Anu Maria Joykutty
Mini-Project Guide
Asst. Professor
Dept. of CSE
RSET

Ms. Anita John
Mini-Project Coordinator
Asst. Professor
Dept. of CSE
RSET

Mr. Sajanraj T D
Mini-Project Co-Coordinator
Asst. Professor
RSET

ACKNOWLEDGEMENTS

We wish to express our sincere gratitude towards **Dr. P. S. Sreejith**, Principal, RSET, and **Dr. Preetha K. G.**, Professor and Head of the Department of Computer Science and Engineering for providing us with the opportunity to undertake our mini-project, "Slides Genie".

We are highly indebted to our mini-project coordinators, **Ms. Anita John**, Assistant Professor, Department of Computer Science and Engineering, and **Mr. Sajanraj T D**, Assistant Professor for their valuable support.

It is indeed our pleasure and a moment of satisfaction for us to express our sincere gratitude to our mini-project guide **Ms. Anu Maria Joykutty**, Assistant Professor, Department of Computer Science and Engineering, for her patience and all the priceless advice and wisdom she has shared with us.

Last but not the least, we would like to express our sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Raphael Tony

Reuben Dinny

Neha Bimal

Shreya Baburaj

ABSTRACT

'SlidesGenie' addresses the challenges in traditional learning and presentation creation by introducing an innovative AI-driven web application that combines the functionalities of an AI learning tool and a presentation generator. The project's objective is to revolutionize the learning experience by leveraging AI technologies and automation to provide users with captivating presentations and enhanced content retention.

The problem definition centers around the limitations of conventional learning methods and time-consuming manual presentation creation. To overcome these hurdles, 'SlidesGenie' is designed with two components: "SlidesGenie" and "Memory Lane."

In the "SlidesGenie" component, users input a topic, and the AI-powered system dynamically generates visually appealing and informative presentations in Google Slides format. The application relies on ChatGPT API for natural language processing and DALL-E API for generating vivid imagery, ensuring high-quality content for users.

The second component, "Memory Lane," aims to improve the learning process by enabling users to input learning materials and automatically generate interconnected vivid imagery based on identified keywords. These visual associations enhance memory retention, facilitating easy concept recall and making learning engaging and memorable.

The methodology involves the seamless integration of Python Flask, ChatGPT API, Google Slides API, and DALL-E API to create the web application. The frontend is developed using Bootstrap, HTML, CSS, and JS, offering a user-friendly interface across various devices.

The successful implementation of "SlidesGenie" and "Memory Lane" signifies a breakthrough in AI-driven learning and presentation generation. By streamlining the educational process and enhancing content retention, 'SlidesGenie' positions itself as a cutting-edge solution for learners and professionals seeking efficient, engaging, and creative ways to learn and present information.

In conclusion, 'SlidesGenie' presents an advanced and comprehensive learning and presentation tool, demonstrating the potential to transform traditional learning and making knowledge dissemination an immersive and impactful experience.

Contents

List of Figures

Chapter 1

Introduction

In today's fast-paced world, the pursuit of knowledge and the quest for effective learning methods have never been more crucial. As individuals seek to stay informed, educated, and competitive in various domains, the challenges of information retention and comprehension have become more pronounced. Traditional learning approaches, while valuable in their own right, may not always cater to diverse learning styles, resulting in limited engagement and reduced long-term retention.

Recognizing these challenges and driven by the desire to revolutionize the learning experience, our team embarked on the development of "Slides Genie" - a cutting-edge software product designed to be a game-changer in the realm of education and learning.

At its core, "Slides Genie" aims to harness the power of advanced Artificial Intelligence (AI) technologies to transform ordinary text-based materials into captivating, narrative-driven presentations. By integrating the capabilities of the ChatGPT API and DALL-E model, our software endeavors to generate a memorable and engaging learning experience.

The primary objective of "Slides Genie" is to provide learners with a dynamic and interactive tool that not only assists in comprehending complex information but also enhances long-term retention through the art of storytelling. By weaving captivating narratives with visually stunning presentations, the software product seeks to spark curiosity, stimulate creativity, and foster a deeper understanding of the subject matter.

In this report, we delve into the journey of "Slides Genie," unveiling its purpose, design, methodology, and envisioned impact. We outline the problem it seeks to address and explain how existing learning methods fall short in meeting the diverse needs of learners. With a clear problem statement and well-defined objectives, we articulate the scope of the project and the potential it holds for transforming the way we learn.

Throughout the report, we share the architectural design of the software, detailing the modules that power its functionality and the seamless integration of AI technologies.

Additionally, we address the risks and challenges we faced during the development process, demonstrating our commitment to delivering a robust and reliable product.

As we conclude this introduction, we are excited to present "Slides Genie" as an innovative step forward in the realm of education and technology. By revolutionizing the learning experience through the marriage of AI and storytelling, we envision a world where knowledge becomes not only accessible but also delightfully immersive. We invite you to embark on this journey with us, as we witness the dawn of a new era in education and learning with "Slides Genie."

1.1 Background

In recent years, there has been a significant shift in education and learning methods. Traditional classroom-style learning is gradually being supplemented and, in some cases, replaced by technology-driven solutions. The rise of online learning platforms and digital tools has opened up new possibilities for enhancing the learning experience.

However, one common challenge remains: the difficulty many individuals face in remembering complex information or materials. While some traditional learning methods, like rote memorization, have their place, they might not be effective for everyone. This problem is particularly relevant in today's fast-paced world, where information overload is a common occurrence.

As technology evolves, there is a growing demand for innovative solutions that can cater to diverse learning styles and make the learning process more enjoyable and memorable. In response to this demand, the "Slides Genie" project aims to leverage cutting-edge AI technologies to create a software product that enhances learning and information retention.

1.2 Existing System

The existing learning methods primarily rely on static resources, including textbooks, classroom lectures, and conventional slide presentations. While these methods have been widely used, they often lack the interactivity and engagement needed for optimal learning outcomes. Learners might face challenges in grasping abstract or complex concepts and may struggle to remember information for an extended period.

1.3 Problem Statement

The "Slides Genie" project seeks to address the limitations of traditional learning methods by providing a dynamic and engaging learning experience.

The problem at hand is the difficulty learners face in effectively remembering and comprehending complex information.

The project aims to create a solution that caters to different learning styles, enhances retention, and fosters a deeper understanding of the subject matter.

1.4 Objectives

The primary objectives of the "Slides Genie" project are as follows:

1. Develop an intuitive and user-friendly software product that can process text-based input materials.

2. Utilize advanced technologies, such as the ChatGPT API and DALL-E, to generate compelling narratives and relevant visuals based on the input text.

3. Create dynamic and visually-rich Google Slide presentations that engage learners and aid in the learning process.

4. Enhance information retention and comprehension for users by presenting information in a story-based format.

1.5 Scope

The scope of the "Slides Genie" project includes the following aspects:

1. Input Processing: The software product will accept text-based materials as input, accommodating various formats and subjects.

2. AI Integration: "Slides Genie" will leverage the powerful capabilities of the ChatGPT API, an advanced language model, to generate coherent and informative narratives based on the input text. Additionally, the DALL-E model, a state-of-the-art image generation model, will be used to create relevant visuals that complement the generated story.

3. Output Presentation: The final output will be a dynamic Google Slide presentation that combines the narrative and visuals in a cohesive manner. The presentation will be visually appealing, interactive, and engaging.

4. Versatility: The project aims to cover a wide range of topics and subject matters, ensuring that learners from various backgrounds and interests can benefit from the "Slides Genie" platform.

The "Slides Genie" team envisions a future where learning becomes a delightful and effective journey. By incorporating the power of AI-generated narratives and visuals into a user-friendly platform, the project aims to empower learners to grasp complex concepts more efficiently, retain information effectively, and develop a lifelong love for learning.

Chapter 2

Literature Review

Explain the existing methods and their drawbacks. Draw a comparison table to compare the existing methods.

The project aims to develop a software product that utilizes advanced technologies to improve learning and information retention by generating story-based presentations from text-based materials. To understand the context of this project, a literature review was conducted to explore existing methods and solutions for enhancing learning and memory retention. This review reveals various traditional learning techniques and other technology-driven approaches that have been used to address the problem of information retention.

2.1 Existing Methods for Enhancing Learning

2.1.1 Mnemonic Techniques

Mnemonics[10] are memory aids that use associations to facilitate information retention. Techniques like acronyms, visualization, or chunking help learners remember complex information by linking it to familiar or easily memorable concepts. While mnemonic techniques can be effective for remembering specific facts or lists, they may not be applicable or practical for comprehensive learning and understanding of complex materials.

2.1.2 Flashcards

Flashcards are a widely used tool for learning and memorization. They typically consist of a question on one side and the answer on the other. The learner repeatedly reviews the flashcards, reinforcing memory through repetition. While effective for rote memorization, flashcards lack engagement and may not promote deeper comprehension or critical thinking.

2.1.3 Visual Aids and Mind Maps

Visual aids, including diagrams, charts, and mind maps, can enhance learning by presenting information in a visual and organized manner. They appeal to visual learners and help establish connections between different concepts. However, creating comprehensive visual aids for extensive textual content can be time-consuming and may not be feasible for every subject or domain.

2.2 Drawbacks of Existing Methods

2.2.1 Limited Engagement and Interest

Traditional learning methods often lack engagement, leading to reduced interest and motivation among learners. Rote memorization, in particular, can become monotonous and boring, hindering long-term retention.

2.2.2 One-Size-Fits-All Approach

Many existing techniques follow a generic approach, assuming that a single method suits all learners. However, individuals have diverse learning styles, and a single technique may not be effective for everyone.

2.2.3 Time-Consuming and Labor-Intensive

Some methods, such as creating extensive visual aids or mnemonic associations, can be time-consuming and require significant effort from educators or learners.

2.2.4 Limited Application to Complex Materials

Traditional methods may work well for simple facts and lists, but they may not be suitable for grasping complex theories or abstract concepts, where a deeper understanding is required.

2.2.5 Dependency on Short-Term Memory

Relying solely on rote memorization or repetition may lead to learners forgetting the information shortly after learning it, as it engages the short-term memory rather than

promoting long-term retention.

2.3 Promising Future Technologies

2.3.1 Artificial Intelligence in Education

The integration of artificial intelligence (AI) in education has gained significant attention in recent years. Chatbots and conversational agents have been utilized to provide personalized learning experiences and assistance to students.

2.3.2 Natural Language Processing (NLP) and Chatbots

The field of NLP has witnessed remarkable advancements, enabling AI systems to process and generate human-like language. Works like "GPT-4 Technical Report" [7] discuss the development of large-scale language models like GPT-3 and GPT-4. These models have demonstrated remarkable capabilities in language generation and understanding, opening possibilities for various applications, including educational aids.

2.3.3 Visual Content Generation with AI

The use of AI for visual content generation has seen progress with models like DALL-E[8], introduced by OpenAI in 2021. DALL-E is a generative model capable of creating images from textual descriptions. This technology can potentially enhance learning by providing relevant visual aids that align with the narrative generated by the AI-powered storytelling system.

2.3.4 Technology-Enhanced Learning Tools

Existing studies on technology-enhanced learning tools emphasize the importance of engagement, interactivity, and personalization. The use of personalized learning paths and content recommendation systems to improve students' engagement and learning outcomes is a trend we see these days. The integration of AI-driven conversational agents and visual content generation aligns with the principles of effective technology-enhanced learning.

2.4 Conclusion

The literature review reveals that traditional learning methods, such as mnemonic techniques and flashcards, have limitations in terms of engagement, flexibility, and long-term retention. While visual aids and mind maps can be effective, they might not be feasible for extensive textual content. As a result, there is a need for innovative and technology-driven approaches that can cater to diverse learning styles, enhance engagement, and facilitate long-term information retention.

The proposed solution of generating story-based presentations using advanced technologies like ChatGPT API and DALL-E holds the potential to overcome these drawbacks. By creating narratives and accompanying visuals[4], the software product aims to provide an engaging and memorable learning experience that addresses the challenges faced by traditional learning methods. However, it is essential to carefully integrate and validate these technologies to ensure their stability, accuracy, and suitability for the target audience. Additionally, addressing potential limitations and biases in the generated content would be crucial for the success and effectiveness of the software product.

Chapter 3

System Analysis

3.1 Expected System Requirements

3.1.1 Internet Connection

Stable and reliable internet connection to access the web application and generate presentations.

3.1.2 Web Browser

Users should have access to a modern web browser (e.g., Google Chrome, Mozilla Firefox, Safari) with JavaScript enabled to interact with the web application.

3.1.3 Device

The product is accessible through various devices, such as desktop computers, laptops, tablets, and smartphones, to cater to the diverse user base.

3.1.4 Processor

Minimum Intel Core i3 or equivalent.

3.1.5 Memory (RAM)

Minimum of 4GB RAM.

3.2 Feasibility Analysis

3.2.1 Technical Feasibility

The product *SlidesGenie*, an AI slides generator, is technically feasible as the requirements to run the product are very minimal and available with the majority of the potential users.

The integration of AI models and APIs, such as ChatGPT and DALL-E, allows for natural language processing and AI image generation. The use of Python Flask enables seamless communication between frontend and backend components, while the Google Slides API automates presentation creation in the user's Google Drive. The frontend technologies ensure a user-friendly interface. Overall, the architecture is scalable, and the product's technical feasibility is supported by its ability to generate presentations based on user inputs[1] and automate slide creation through AI-powered processes.

3.2.2 Operational Feasibility

The product is operationally feasible. The web application's automated process of generating presentations based on user-inputted topics streamlines the presentation creation workflow, reducing the manual effort required from users significantly. With AI-powered integration, users can efficiently convert their ideas into compelling slides without the need for extensive design or content preparation. The seamless integration of the Google Slides API further enhances operational efficiency, enabling users to access and store their presentations directly in their Google Drive. Moreover, the utilization of widely supported frontend technologies ensures broad accessibility, allowing users to access the application on various devices. Overall, the operational feasibility of *SlidesGenie* is evident through its ability to provide a user-friendly and efficient solution for presentation generation.

3.2.3 Economic Feasibility

Determining the economic feasibility of the product *SlidesGenie* depends on various factors such as development costs, maintenance expenses, and potential revenue streams. While the integration of AI models and APIs, along with backend server infrastructure, may incur initial development costs, the automated nature of the application could lead to reduced operational expenses over time. The availability of open-source technologies and frameworks like Python Flask, Bootstrap, HTML, CSS, and JS[5] may contribute to cost-effectiveness. However, the APIs are paid and monetization strategies for the same, such as freemium models or subscription plans, could generate revenue from users. Considering these aspects, *SlidesGenie* can be said to be economically feasible.

3.3 Hardware Requirements

3.3.1 Processor

Users should have a modern processor (e.g., Intel Core i3 or higher) to ensure smooth performance while using the web application.

3.3.2 Memory (RAM)

A minimum of 4GB RAM is recommended to handle the AI-powered operations and to prevent any slowdowns during the presentation generation process.

3.3.3 Storage Space

As *SlidesGenie* is a web application, there are no specific storage requirements for the software itself apart from Google Drive storage. However, users will need enough free storage space to save the generated presentations locally, if desired.

3.3.4 Display

Users will benefit from a screen with a resolution of at least 1280x720 to visualize the web application's interface and presentations effectively.

3.4 Software Requirements

3.4.1 Operating System

SlidesGenie should be compatible with major operating systems, including but not limited to Windows, macOS, and Linux, to accommodate users on different platforms.

3.4.2 Web Browser

The web application should support popular web browsers like Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari, ensuring broad accessibility.

3.4.3 Google Account

To allow seamless integration with Google Slides API, users will need a valid Google account to authorize the application and enable the automatic presentation creation.

Chapter 4

Methodology

4.1 Proposed Method

The "Slides Genie" project adopts a multi-step methodology that combines advanced AI technologies and presentation generation to create a compelling and interactive learning experience. The proposed method can be broken down into the following steps:

4.1.1 Step 1: Input Processing

The first step involves processing the text-based input materials provided by the user. The input could be in the form of a textbook excerpt, lecture notes, or any other textual content. The system will validate and preprocess the input to ensure it is in a suitable format for further processing.

4.1.2 Step 2: ChatGPT Integration

In this step, the processed input will be passed to the ChatGPT API[6], which is an advanced language model capable of generating coherent and contextually relevant narratives. The API will be primed with prompts to instruct it to create a story based on the input text. The generated narrative will serve as the backbone of the learning content. The output from this will be a JSON file containing the slides format, content for each slide, and if an image is there, image prompt for DALL-E.

4.1.3 Step 3: DALL-E Integration

Simultaneously, the system will utilize the DALL-E model[9] to generate relevant visuals that complement the story. DALL-E is a state-of-the-art image generation model that can create images based on textual descriptions. The generated images will add a visual dimension to the learning experience, making it more engaging and memorable.

4.1.4 Step 4: Google Slides Integration

Once both the narrative from ChatGPT and the visuals from DALL-E are ready, the system will integrate them into a dynamic Google Slide presentation. The Google Slides API[3] will be employed to programmatically create and format the presentation slides. The narrative will be used as the textual content for each slide, while the generated images will be embedded within the presentation.

4.1.5 Step 5: User Interaction

The final output will be a story-based presentation that combines text and visuals, enhancing the learning experience. Users will have the opportunity to interact with the presentation, exploring the content at their own pace. The interactive nature of the slides will promote active learning, enabling users to delve deeper into the subject matter.

4.1.6 Step 6: Performance Evaluation

Throughout the development process, the team will rigorously evaluate the performance of the "Slides Genie" platform. The evaluation will involve user feedback, surveys, and testing to assess the effectiveness of the generated narratives, visuals, and overall learning experience.

The proposed method ensures that learners have access to an engaging and interactive learning tool that leverages the capabilities of advanced AI models. By combining the power of narrative generation with visually appealing content, "Slides Genie" aims to revolutionize the way people learn and retain complex information.

Chapter 5

System Design

5.1 Architecture Diagram

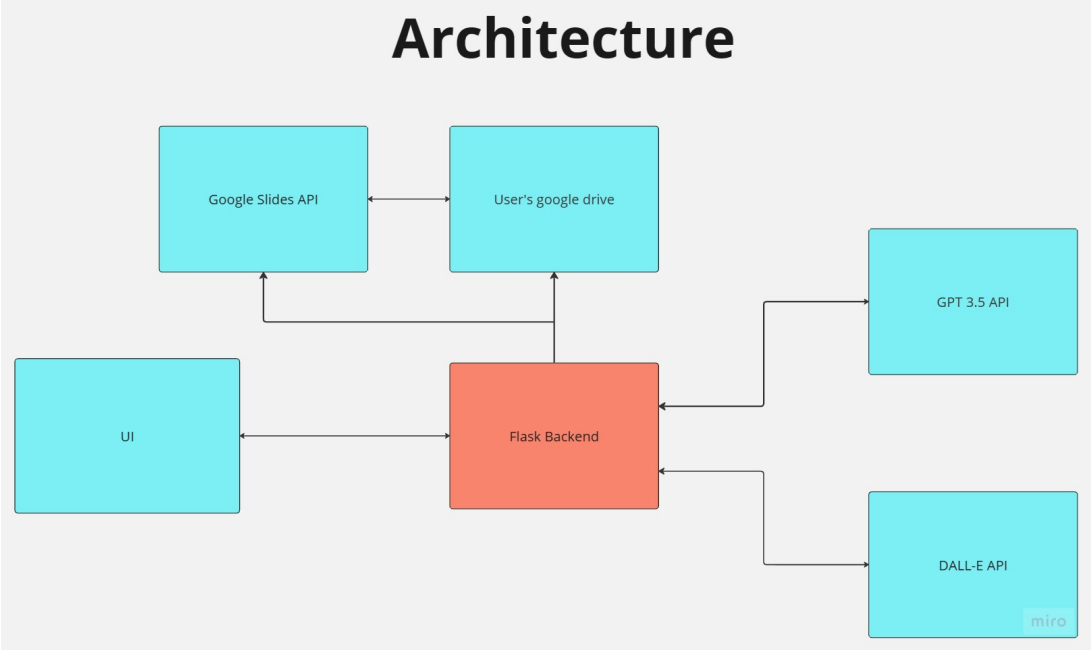

Figure 5.1: Architecture Diagram

5.2 Interaction Diagram

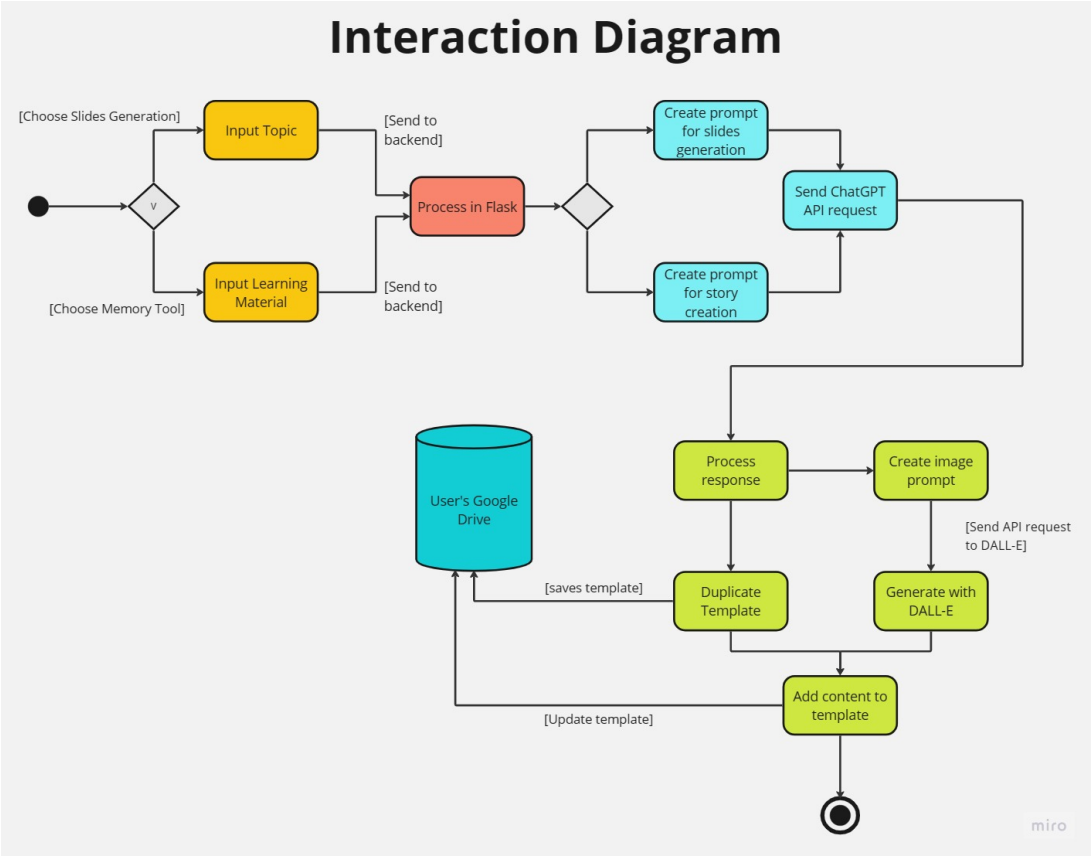

Figure 5.2: Interaction Diagram

5.3 ChatGPT Interaction

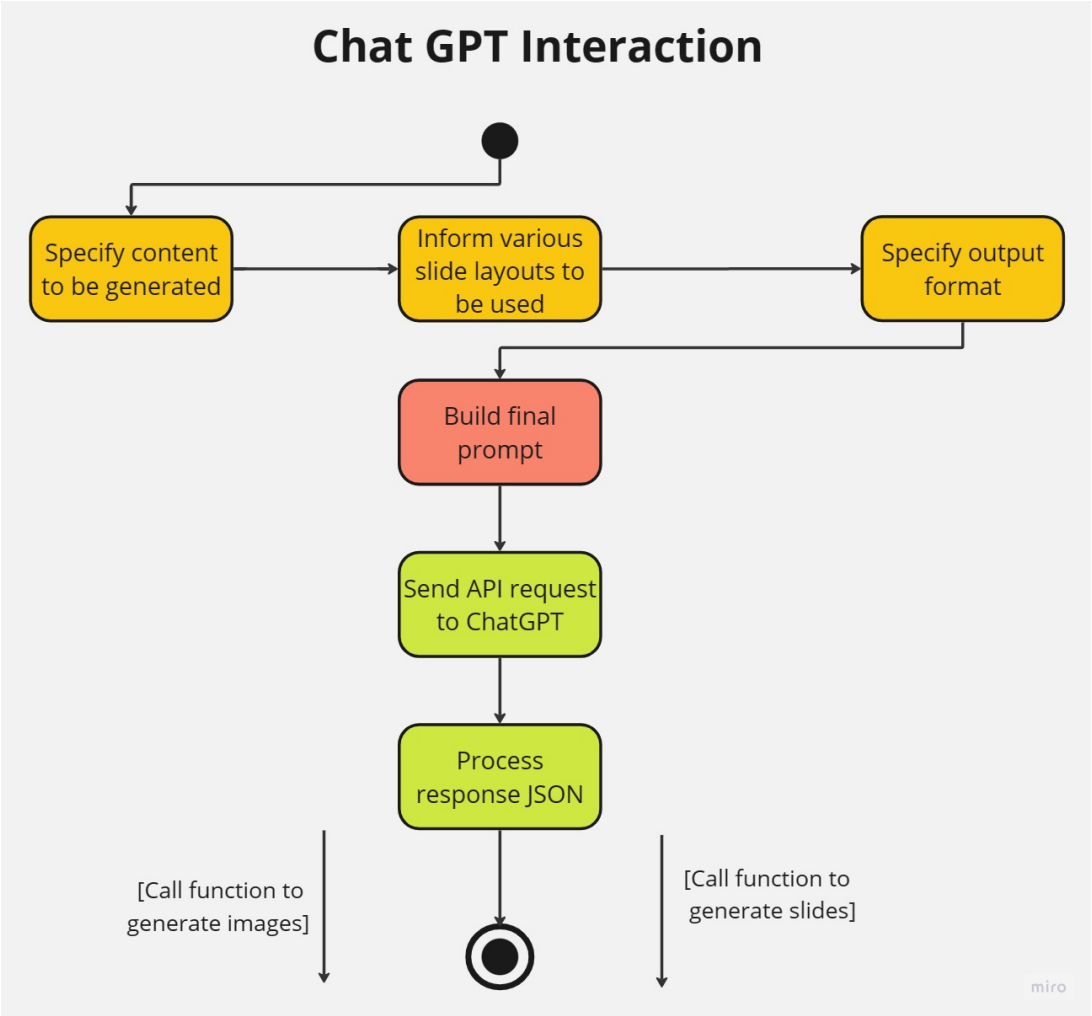

Figure 5.3: ChatGPT Interaction

5.4 Google Slides Interaction

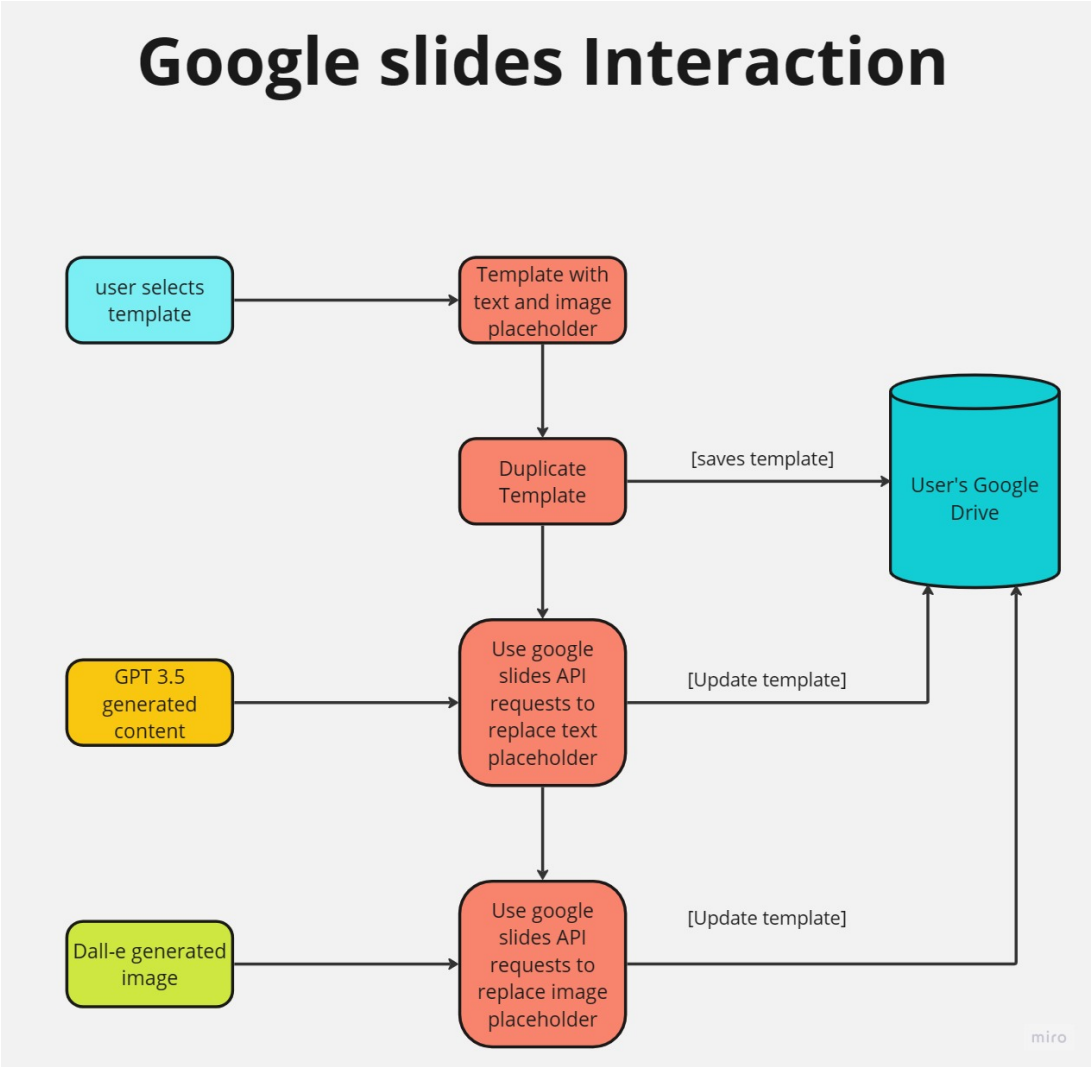

Figure 5.4: Google Slides Interaction

5.5 DALL-E Interaction

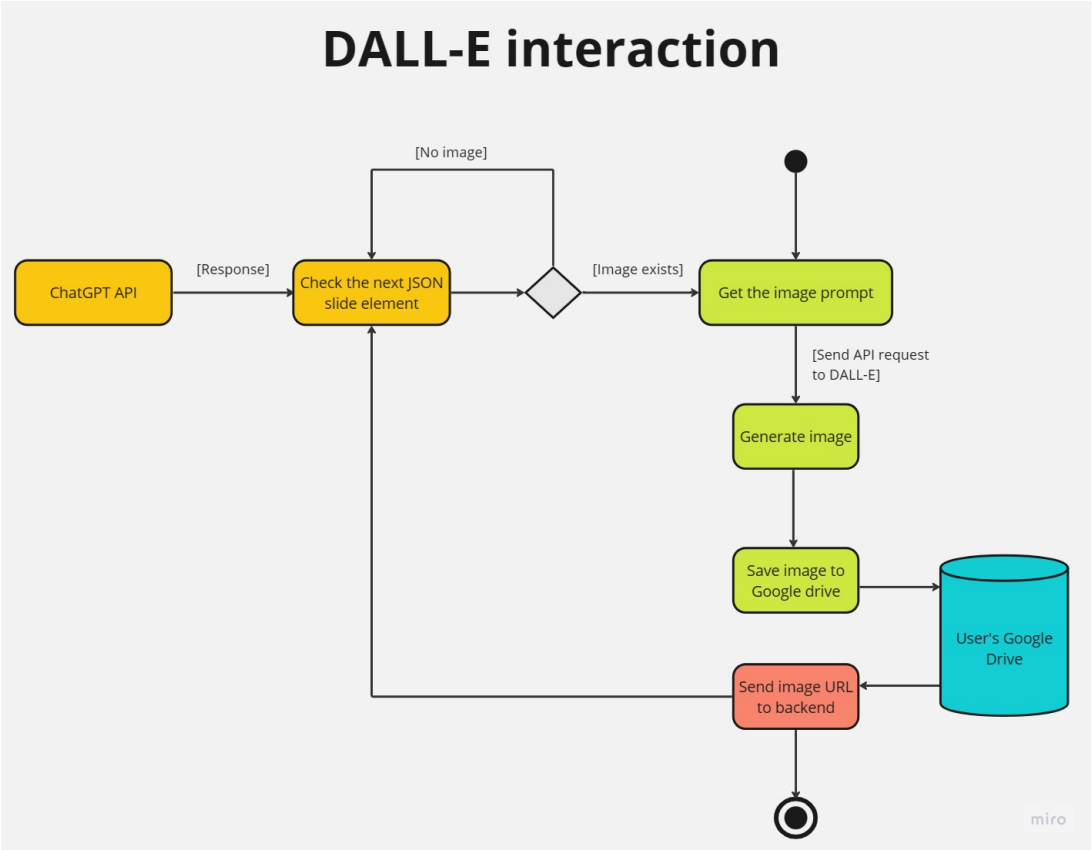

Figure 5.5: DALL-E Interaction

Chapter 6

System Implementation

6.1 GPT

GPT (Generative Pre-trained Transformer) is a cutting-edge language model that has revolutionized natural language processing tasks. Pre-trained on vast amounts of text data, GPT can generate coherent and contextually relevant text based on provided prompts. Its ability to understand and generate human-like text makes it a powerful tool for automating slide content creation in SlidesGenie.

6.1.1 DALL-E

DALL-E, an advanced image generation model developed by OpenAI, plays a vital role in SlidesGenie. By utilizing deep learning and generative adversarial networks (GANs), DALL-E can generate highly customizable images based on textual prompts. In the context of SlidesGenie, DALL-E enables the seamless incorporation of visually stunning and tailored images into the generated slides, enhancing the visual appeal and engagement.

6.2 Flask

The Flask web framework[2] serves as the foundation for the SlidesGenie web application. Flask enables smooth integration of the different modules and provides robust functionality for handling user inputs, managing server-side logic, and rendering dynamic content. With Flask, SlidesGenie delivers a user-friendly experience, allowing users to generate and customize slides efficiently.

6.3 Google Drive

SlidesGenie integrates with Google Drive, a cloud storage platform, to ensure seamless access and storage of generated slides and associated files. Google Drive provides secure storage, sharing, and easy access to the slide-related content. This integration enables SlidesGenie to efficiently handle file management, offering users a centralized and reliable repository for their slide-related assets.

6.4 Google Slides API

Integration with the Google Slides API empowers SlidesGenie to seamlessly interact with Google Slides. The API allows for creating, modifying, and managing presentations. By leveraging the Google Slides API, SlidesGenie enables direct slide creation, editing, and collaboration, enhancing the flexibility and functionality of the generated slides.

Chapter 7

Testing

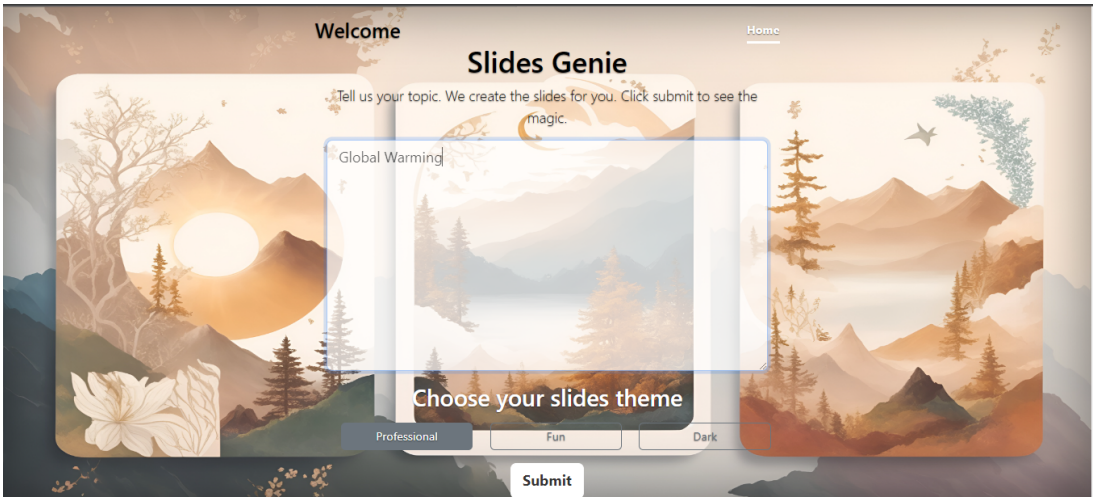

Figure 7.1: Slides Genie - Input topic

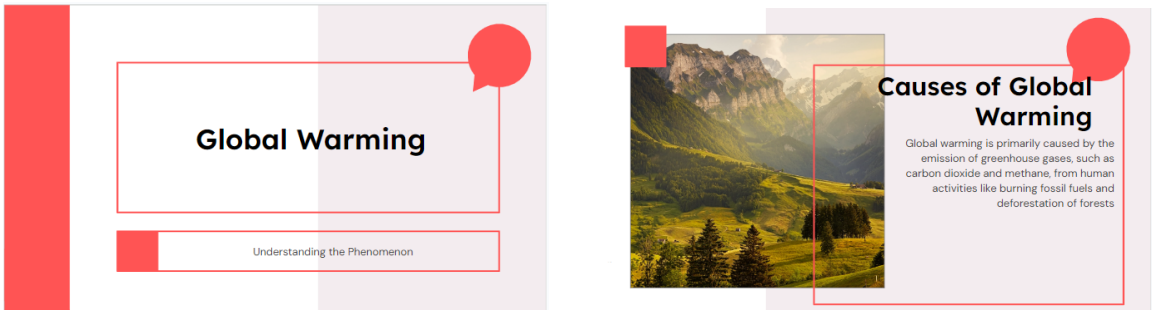

Figure 7.2: Slides Genie - Slides generated

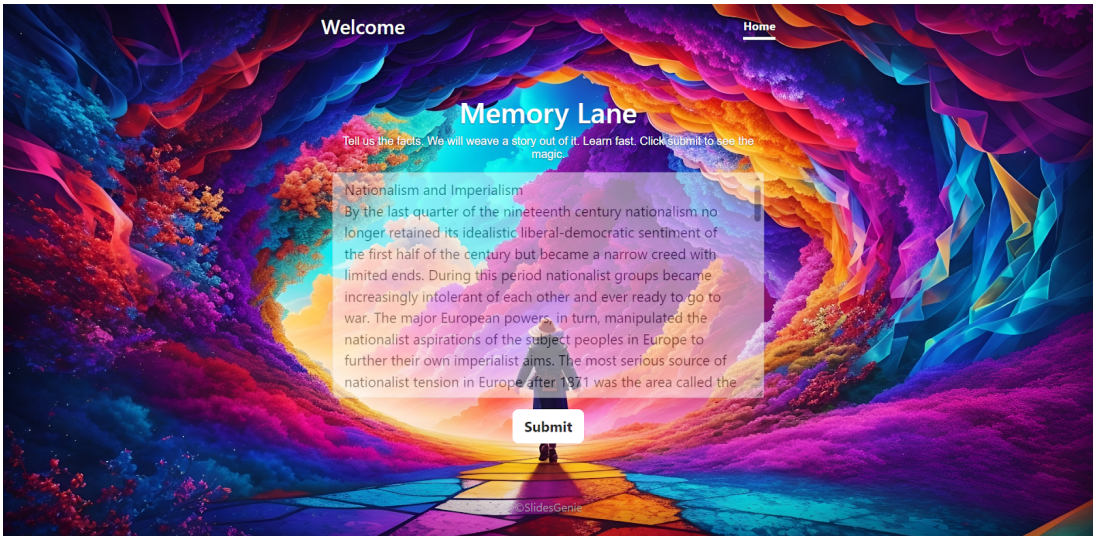

Figure 7.3: Memory Lane - Input material

Figure 7.4: Memory Lane - Slides generated

Chapter 8

Result

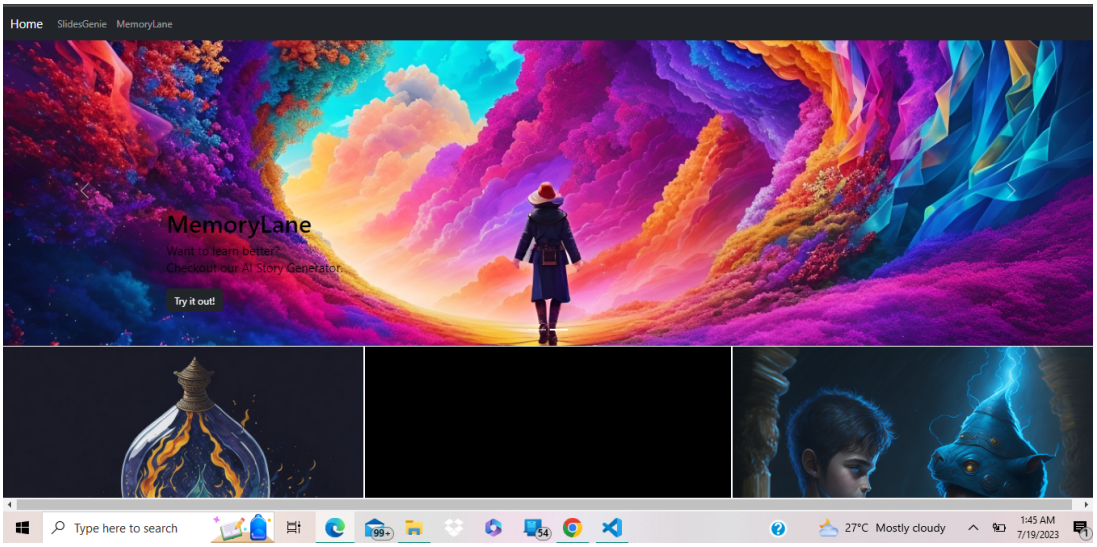

Figure 8.1: Home page

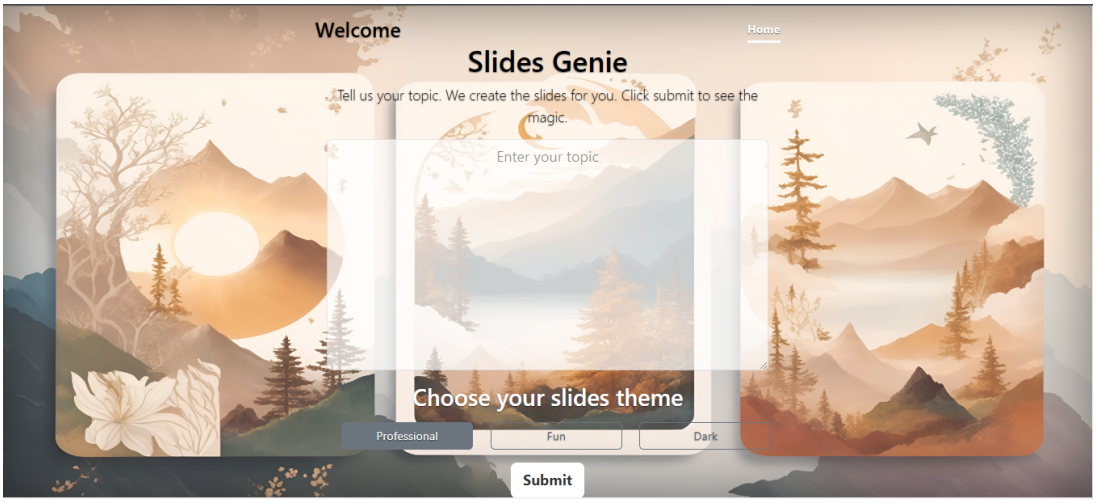

Figure 8.2: SlidesGenie page

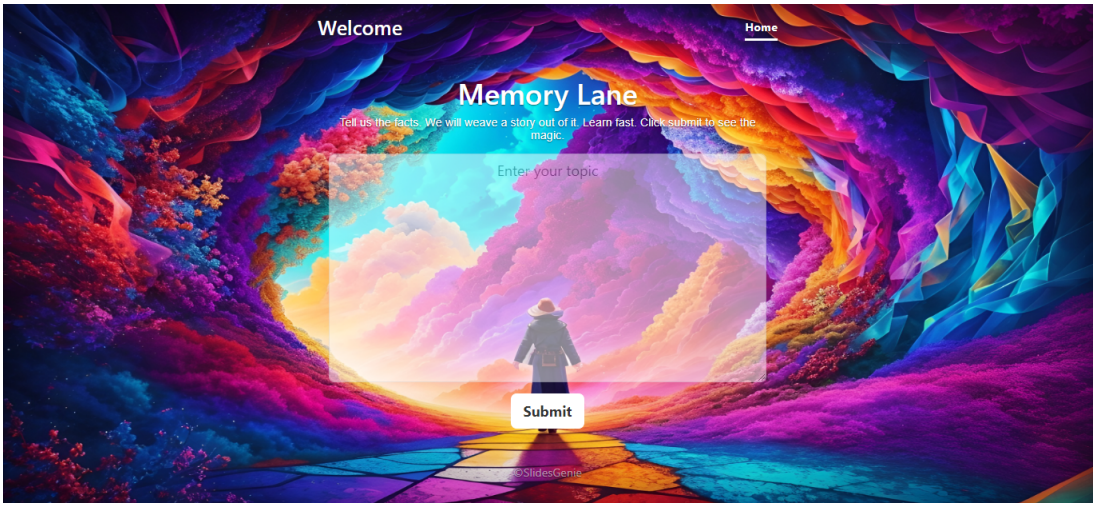

Figure 8.3: MemoryLane page

Chapter 9

Risks and Challenges

1. Availability and Stability of the Google Slides API, ChatGPT API, and DALL-E Model:

The availability and stability of the Google Slides API, ChatGPT API, and DALL-E model are critical factors in the success of the SlidesGenie project. The Google Slides API needs to remain consistently available and stable, ensuring users can seamlessly create, update, and manipulate slides using the generated content and visuals. Similarly, the ChatGPT API and DALL-E model must be reliable and accessible, providing uninterrupted access to the natural language processing and image generation capabilities required for the project's functionalities. Continuous updates and support from Google and OpenAI are crucial to ensure these APIs and models remain accessible and dependable for users.

2. Ensuring Seamless Integration and Communication Between the Different Modules:

To provide a comprehensive solution, SlidesGenie must ensure seamless integration and communication between its different modules. The content generation module, powered by the ChatGPT API, should seamlessly interact with the DALL-E image creation module and the MemoryLane module for incorporating vivid visuals. Efficient data flow and communication protocols must be established to facilitate the transfer of information between these modules. This includes passing generated text from the content generation module to the image creation module, ensuring that visuals align with the generated content. The integration should be designed to minimize latency and optimize the overall performance of the system, creating a seamless user experience.

3. Handling Large Volumes of Input Text Efficiently:

SlidesGenie must efficiently handle large volumes of input text to meet the demands of various presentations. The system should be capable of processing and generating content swiftly, even when dealing with lengthy input texts. This requires the implementation of

efficient algorithms and optimizations that can handle the computational requirements of processing and generating content in real-time. Additionally, the system should allow users to input text in a structured manner, enabling them to define the layout, hierarchy, and formatting of the content. Efficiently handling large volumes of text not only improves the user experience but also ensures that SlidesGenie remains a scalable and versatile solution for presentations of varying lengths and complexities.

4. Addressing Potential Limitations and Biases in the Generated Story and Visuals:

While AI models like ChatGPT and DALL-E have demonstrated remarkable capabilities, they may still exhibit limitations and biases in the generated content and visuals. SlidesGenie must address these potential issues to provide users with accurate and unbiased results. This includes ongoing model improvement, feedback loops, and user input to refine the models and reduce biases over time. Transparency in the model's performance and potential biases is essential, along with regular monitoring and evaluation. By adhering to responsible AI development practices, SlidesGenie can minimize unintended biases and ensure that the generated story and visuals are fair, accurate, and reliable, aligning with ethical considerations in AI development.

Chapter 10

Conclusion

SlidesGenie provides users with a comprehensive and efficient solution for generating high-quality slides. The utilization of GPT ensures that the content generated is coherent, relevant, and tailored to the user's needs. Furthermore, DALL-E's ability to create custom images based on textual descriptions adds an extra layer of visual appeal and uniqueness to the slides.

One of the standout features of SlidesGenie is the MemoryLane module, which leverages vivid visuals to aid in memory retention. By incorporating visually engaging elements that align with the content being presented, the system helps users grasp and retain information more effectively. This not only enhances the overall presentation experience but also promotes better knowledge retention for both presenters and audiences.

The successful implementation of SlidesGenie opens up a wide range of possibilities for various industries and professionals who heavily rely on slide presentations. Whether it's in academia, business, or any other field, the project offers a transformative solution that saves time, enhances content quality, and maximizes the impact of presentations.

While the SlidesGenie project has achieved significant milestones in combining GPT, DALL-E, and MemoryLane to automate the creation of slides, there is still room for improvement. Further research and development can focus on refining the AI models, expanding the range of content and visuals they can generate, and exploring additional features to cater to diverse presentation needs.

Bibliography

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020.

[2] Flask. Flask Documentation. ĥttps://flask.palletsprojects.com/en/2.3.x/ (Jun. 2023).

[3] Google. Google Slides API Documentation. ĥttps://developers.google.com/slides/api/reference/rest (Jun. 2023).

[4] Hassan Mahdi. The effect of using bizarre images as mnemonics to enhance vocabulary learning. , 24:113–135, 03 2018.

[5] MDN. Mozilla Developer Network - Web Technology Documentation. M̂ozilla Developer Network - Web Technology Documentation (Jun. 2023).

[6] OpenAI. OpenAI Platform Documentation. https://platform.openai.com/docs/introduction/overview (Jun. 2023).

[7] OpenAI. Gpt-4 technical report, 2023.

[8] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022.

[9] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021.

[10] Thayne L Sweeten. Mnemonic mechanisms for making memories. *Journal on Empowering Teaching Excellence*, 1(2), 2017.

Appendix A: Sample Code

Flask main

```
from flask import Flask, render_template,request,redirect

from backend.slides_manip import copy_presentation, get_presentation,
create_slide_copy, delete_template_slides, reorder_slides
from backend.GPT_engine import content_generation
from backend.memory_lane import visualize
from backend.slides_creation import
create_title_slide,create_left_image_slide,create_right_image_slide,
create_title_sub_text_slide,create_image_slide
app = Flask(__name__)

SLIDE_DARK="1Qw0oqIpGSrEyQZkFhFnzxj6-4kMq8X1TnSdqpG94Ch8"
SLIDE_PROF="159B-JLzNz0rWHMHYbQFdfitqoqObyxrMfj-RtEXG91M"
SLIDE_FUN="1Cu60Vh1dYYaEqb5vqM97PwfmuWFUSxgAoRAaqqdgmZA"

@app.route("/")
def hello_world():
    return render_template("index.html")


@app.route("/slidesgenie")
def get_slides_genie():
    return render_template("slides_genie.html")

@app.route("/memorylane")
def get_mem_lane():
    return render_template("mem_lane.html")

#common submit endpoint for both slidesgenie and memorylane
@app.route('/submit/', methods = ['POST'])
def process_submit():
    new_presentation_id=""
    if request.method == 'POST':
        form_data = dict(request.form)
        source = request.headers["Referer"].split('/')[-1] #Getting from
which the submit request came
        if(str(source) == 'slidesgenie'):
            new_presentation_id +=
slides_genie(form_data['userInput'],form_data['styleSelect'])
        elif(str(source) == 'memorylane'):
            new_presentation_id += memory_lane(form_data['userInput'])
        else:
            return "Something went wrong"
```

```
    return f"https://docs.google.com/presentation/d/{new_presentation_id}"

def memory_lane(user_input):

    response = visualize(user_input)

    new_presentation_id =
copy_presentation("1oBjYbkCRWQwhOiNC4hHTMsuwIaURzPQvpYjXm2QVYLM",response[
'slides'][0]['inputs']['title'])
    new_slides = get_presentation(new_presentation_id)

    counter = 0

    for slide in response['slides']:

create_slide_copy(new_presentation_id,new_slides,slide['type_id'],counter)

        if(slide['type_id'] == 'title'):

create_title_slide(new_presentation_id,slide['inputs'],counter)

        elif(slide['type_id'] == 'image-text'):
            create_image_slide(new_presentation_id,slide['inputs'],
counter)

        counter = counter + 1
    delete_template_slides(new_presentation_id,new_slides)
    reorder_slides(new_presentation_id,counter)
    return new_presentation_id


def slides_genie(user_input,slideStyle):
    if slideStyle=='dark':
        slideStyle = SLIDE_DARK
    elif slideStyle=='professional':
        slideStyle = SLIDE_PROF
    elif slideStyle=='fun':
        slideStyle = SLIDE_FUN


    response = content_generation(user_input)
    print(response)
    new_presentation_id =
copy_presentation(slideStyle,response['slides'][0]['inputs']['title'])
    new_slides = get_presentation(new_presentation_id)

    counter = 0
```

```
    for slide in response['slides']:

create_slide_copy(new_presentation_id,new_slides,slide['type_id'],counter)

        if(slide['type_id'] == 'title'):

create_title_slide(new_presentation_id,slide['inputs'],counter)

        elif(slide['type_id'] == 'left-image-text'):
            create_left_image_slide(new_presentation_id,slide['inputs'],
counter)

        elif(slide['type_id'] == 'right-image-text'):
            create_right_image_slide(new_presentation_id,slide['inputs'],
counter)

        elif(slide['type_id'] == 'title-sub-text'):

create_title_sub_text_slide(new_presentation_id,slide['inputs'], counter)

        counter = counter + 1
    delete_template_slides(new_presentation_id,new_slides)
    reorder_slides(new_presentation_id,counter)
    return new_presentation_id
```

Google Cloud Console Connection

```
import os.path

from google.auth.transport.requests import Request
from google.oauth2.credentials import Credentials
from google_auth_oauthlib.flow import InstalledAppFlow
from googleapiclient.discovery import build
from googleapiclient.errors import HttpError

def get_creds():
    # Setting up the authentication:
    SCOPES = ['https://www.googleapis.com/auth/presentations',
'https://www.googleapis.com/auth/drive','https://www.googleapis.com/auth/d
rive.file']
    creds = None
    # The file token.json stores the user's access and refresh tokens, and
is
```

```
    # created automatically when the authorization flow completes for the
first
    # time.
    if os.path.exists('token.json'):
        creds = Credentials.from_authorized_user_file('token.json',
SCOPES)
    # If there are no (valid) credentials available, let the user log in.
    if not creds or not creds.valid:
        if creds and creds.expired and creds.refresh_token:
            creds.refresh(Request())
        else:
            flow = InstalledAppFlow.from_client_secrets_file(
                'credentials.json', SCOPES)
            creds = flow.run_local_server(port=0)
        # Save the credentials for the next run
        with open('token.json', 'w') as token:
            token.write(creds.to_json())
    return creds


def get_slides_service():
    try:
        slides_service = build('slides', 'v1', credentials=get_creds())
    except HttpError as error:
        print(f"An error occurred: {error}")
    return slides_service

def get_drive_service():
    try:
        drive_service = build('drive', 'v3', credentials=get_creds())

    except HttpError as error:
        print(f"An error occurred: {error}")
    return drive_service
```

GPT Connection for SlidesGenie

```
import openai
import json
from dotenv import load_dotenv
import os

load_dotenv()
openai.api_key = os.getenv("OPENAI_API")
```

prompt = ''' Imagine you are an AI Slides generating tool called 'SlidesGenie'. Based on the input of the user, create the structure of the slides presentation as a JSON object.
You have 4 slide formats at your disposal to use. Each slide has a type_id and takes different inputs. Slides with images have a special input called image_prompt. This should be a description of an image that can be generated using a text-to-image model. The description should match with the contents of the slide. The slide formats are:
1. Title Slide
The title slide consists of a title and a subtitle.
type_id:title
inputs: title,subtitle
2. Slide with Image on left
The slide consists of an image on the left, a title on the right, and a body of text under the title.
type_id: left-image-text
inputs: title, image_prompt, body.
3. Slide with Image on Right
The slide consists of an image on the right, a title on the left, and a body of text under the title.
type_id: right-image-text
inputs: title, image_prompt, body.
4. Slide with only text
The slide consists of a title, subtitle and body
type_id: title-sub-text
inputs: title, subtitle, body

Template:
```
{
  "slides": [
    {
      "type_id": "title",
      "inputs": {
        "title": "<insert-title>",
        "subtitle": "<insert-subtitle-here>"
      }
    },
    {
      "type_id": "left-image-text",
      "inputs": {
        "title": "<insert-title>",
        "image_prompt": "<insert-image-generating-prompt>",
        "body": "<insert-body>"
      }
    },
```

```
    {
      "type_id": "right-image-text",
      "inputs": {
        "title": "<insert-title>",
        "image_prompt": "<insert-image-generating-prompt>",
        "body": "<insert-body>"
      }
    },
    {
      "type_id": "title-sub-text",
      "inputs": {
        "title": "<insert-title>",
        "subtitle": "<insert-subtitle>",
        "body": "<insert-body>"
      }
    }
  ]
}
```
The above JSON contains a list of the 4 slide templates: title, left-image-text, right-image-text and title-sub-text. Use this to create 10 slides. For each slide, pick an appropriate slide template from the 4 templates given in the JSON and generate the response. Be creative and factual with the content. Comply with the user's input.
RESPOND WITH JSON ONLY

```
'''
def content_generation(user_input):
  user_input = "user input : "+ user_input

  response = openai.ChatCompletion.create(
      model="gpt-3.5-turbo",
      messages=[
              {"role": "system", "content": prompt},
              {"role": "user", "content": user_input},
          ]
  )

# print(response["choices"][0]["message"]["content"])

  d = json.loads(response["choices"][0]["message"]["content"])
  print(d)

  return d
```

GPT Connection for Memory Lane

```
import openai
import json
from dotenv import load_dotenv
import os

load_dotenv()
openai.api_key = os.getenv("OPENAI_API")

prompt1 ='''You are an AI called MemoryLane.
You help people to remember study material
by using vivid strange imagery.
Consider the following user input.
Identify the main topic and the keywords related to that topic.
Generate a vivid imagery for each of the keywords in the paragraph in such
a way that each imagery connects to the next one.
'''

prompt2 = '''
You have to convert the content as a slides presentation. Create the
structure of the slides presentation as a JSON object.
You have 2 slide formats. Each slide has a type_id and takes different
inputs. The slide formats are:
1. Title Slide
The title slide consists of a title and a subtitle.
type_id:title
inputs: title
2. Slide with visual description
The slide consists of the keyword and the visual description.
type_id: image-text
inputs: keyword, visual.
Template:
```
{
  "slides": [
    {
      "type_id": "title",
      "inputs": {
        "title": "<insert-title>"
      }
    },
    {
      "type_id": "image-text",
      "inputs": {
```
```

```
        "keyword": "<insert-keyword>",
        "visual": "<insert-keyword-visual-description>"
      }
    },

  ]
}

```
The above JSON contains two slide templates: title, image-text. Use this to generate the slides for each keyword and nest the required details accurately. For each keyword, use the image-text slide and generate the response. RESPOND WITH JSON ONLY

'''

```
def visualize(user_input):
    user_input = "user input : "+ user_input

    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=[
                {"role": "system", "content": prompt1},
                {"role": "user", "content": user_input},
            ]
    )

    content = "content: "+
str(response["choices"][0]["message"]["content"])

    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=[
                {"role": "system", "content": prompt2},
                {"role": "user", "content": content},
            ]
    )


    d = json.loads(response["choices"][0]["message"]["content"])
    print(d)

    return d

visualize(txt)
```

DALL-E Image Generation

```
import openai
import json
from dotenv import load_dotenv
import os

load_dotenv()
openai.api_key = os.getenv("OPENAI_API")

def image_generation(image_prompt):
    image_prompt = image_prompt + ",digital art, 4k, highly detailed,
storybook"
    response = openai.Image.create(
    prompt= image_prompt,
    n=1,
    size="512x512"
    )
    image_url = response['data'][0]['url']

    print(image_url)
    return image_url
```

Interacting with Slides

```
from __future__ import print_function

from googleapiclient.errors import HttpError
from backend.connection import get_drive_service
from backend.connection import get_slides_service

#Connecting to the drive service
drive_service = get_drive_service()
slides_service = get_slides_service()

def copy_presentation(presentation_id, copy_title):
    """
           Creates the copy Presentation the user has access to.
           Load pre-authorized user credentials from the environment.
           TODO(developer) - See https://developers.google.com/identity
           for guides on implementing OAuth2 for the application.
           """
    try:
```

```
        body = {
            'name': copy_title
        }
        drive_response = drive_service.files().copy(
            fileId=presentation_id, body=body).execute()
        presentation_copy_id = drive_response.get('id')

    except HttpError as error:
        print(f"An error occurred: {error}")
        print("Presentations  not copied")
        return error

    return presentation_copy_id

# new_id =
copy_presentation("1Qw0oqIpGSrEyQZkFhFnzxj6-4kMq8X1TnSdqpG94Ch8","My
presentation")
# print(new_id)

def create_slide_copy(presentation_id,slides,slide_type,counter):
    if slide_type == 'title':
        pageId = slides[0]['objectId']
    elif slide_type == 'left-image-text':
        pageId = slides[1]['objectId']
    elif slide_type == 'right-image-text':
        pageId = slides[2]['objectId']
    elif slide_type == 'title-sub-text':
        pageId = slides[3]['objectId']

    #for memorylane
    elif slide_type == 'image-text':
        pageId = slides[1]['objectId']

    requests = {
    "requests" : [
    {
      "duplicateObject": {
        "objectId": pageId,
        "objectIds": {
          pageId: "copiedSlide" + str(counter),
        }
      }
    }
    ]
    }

    try:
```

```
        response = slides_service.presentations() \
            .batchUpdate(presentationId=presentation_id,
body=requests).execute()

    except HttpError as error:
        print(f"An error occurred: {error}")
        print("Slide not copied")
        return error


def get_presentation(presentation_id):
    presentation =
slides_service.presentations().get(presentationId=presentation_id).execute
()
    slides = presentation.get('slides')
    print(slides)
    return slides

def delete_template_slides(presentation_id, slides):
    requests = {"requests":[]}
    for slide in slides:
        requests["requests"].append(
            {
            "deleteObject": {
                "objectId": slide["objectId"],
                }
            }
        )

    try:
        response = slides_service.presentations() \
            .batchUpdate(presentationId=presentation_id,
body=requests).execute()

    except HttpError as error:
        print(f"An error occurred: {error}")
        print("Template slides deleted")
        return error

def reorder_slides(presentation_id, counter):
    requests = {"requests":[]}
    for i in range(counter-1,-1,-1):
        requests["requests"].append(
            {
                "updateSlidesPosition": {
                "slideObjectIds": [
                    f"copiedSlide{i}"
```

```
                ],
        "insertionIndex": 0
                }
            }
        )

    try:
        response = slides_service.presentations() \
            .batchUpdate(presentationId=presentation_id,
body=requests).execute()

    except HttpError as error:
        print(f"An error occurred: {error}")
        print("Error reordering")
        return error
```

Slide Creation

```
from __future__ import print_function

from googleapiclient.discovery import build
from googleapiclient.errors import HttpError
from backend.connection import get_drive_service
from backend.connection import get_slides_service
from backend.image_engine import image_generation

#Connecting to the drive service
drive_service = get_drive_service()
slides_service = get_slides_service()


def create_title_slide(presentation_id, content,counter):
    # Since memory lane does not have a subtitle field,have to manually
set it
    if 'subtitle' not in content.keys():
        content['subtitle']=''
    requests = [
            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<title>>',
                        'matchCase': True
                    },
                    'replaceText': content['title'],
```

```
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            },
            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<title-subtitle>>',
                        'matchCase': True
                    },
                    'replaceText': content['subtitle'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            }
        ]

        # Execute the requests for this presentation.
    body = {
        'requests': requests
    }
    response = slides_service.presentations().batchUpdate(
        presentationId=presentation_id, body=body).execute()


def create_left_image_slide(presentation_id, content,counter):
    requests = [
            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<left-image-text_title>>',
                        'matchCase': True
                    },
                    'replaceText': content['title'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            },

            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<left-image-text_body>>',
                        'matchCase': True
                    },
                    'replaceText': content['body'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            },
```

```
            {
                'replaceAllShapesWithImage': {
                    'imageUrl': image_generation(content['image_prompt']),
                    'replaceMethod': 'CENTER_INSIDE',
                    'containsText': {
                        'text': '<<left-image-text_image>>',
                        'matchCase': True
                    },
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            }
    ]

    body = {
        'requests': requests
    }
    response = slides_service.presentations().batchUpdate(
        presentationId=presentation_id, body=body).execute()

def create_right_image_slide(presentation_id, content,counter):
    requests = [
            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<right-image-text_title>>',
                        'matchCase': True
                    },
                    'replaceText': content['title'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            },

            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<right-image-text_body>>',
                        'matchCase': True
                    },
                    'replaceText': content['body'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            },

            {
                'replaceAllShapesWithImage': {
                    'imageUrl': image_generation(content['image_prompt']),
                    'replaceMethod': 'CENTER_INSIDE',
```

```
                    'containsText': {
                        'text': '<<right-image-text_image>>',
                        'matchCase': True
                    },
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            }
    ]

    body = {
        'requests': requests
    }
    response = slides_service.presentations().batchUpdate(
        presentationId=presentation_id, body=body).execute()



def create_title_sub_text_slide(presentation_id, content,counter):
    requests = [
            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<title-sub-text_title>>',
                        'matchCase': True
                    },
                    'replaceText': content['title'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            },

            {
                     'replaceAllText': {
                    'containsText': {
                        'text': '<<title-sub-text_sub>>',
                        'matchCase': True
                    },
                    'replaceText': content['subtitle'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            },

            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<title-sub-text_body>>',
                        'matchCase': True
                    },
```

```
                    'replaceText': content['body'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            }
        ]

    # Execute the requests for this presentation.
    body = {
        'requests': requests
    }
    response = slides_service.presentations().batchUpdate(
        presentationId=presentation_id, body=body).execute()


def create_image_slide(presentation_id, content,counter):
    requests = [
            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<image-text_title>>',
                        'matchCase': True
                    },
                    'replaceText': content['keyword'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            },

            {
                'replaceAllText': {
                    'containsText': {
                        'text': '<<image-text_body>>',
                        'matchCase': True
                    },
                    'replaceText': content['visual'],
                    'pageObjectIds':[f'copiedSlide{counter}']
                }
            },

            {
                'replaceAllShapesWithImage': {
                    'imageUrl': image_generation(content['visual']),
                    'replaceMethod': 'CENTER_INSIDE',
                    'containsText': {
                        'text': '<<image-text_image>>',
                        'matchCase': True
                    },
                    'pageObjectIds':[f'copiedSlide{counter}']
```

```
            }
        }
]

body = {
    'requests': requests
}
response = slides_service.presentations().batchUpdate(
    presentationId=presentation_id, body=body).execute()
```

Appendix B: CO-PO and CO-PSO mapping

COURSE OUTCOMES:

After completion of the course the student will be able to

SL. NO	DESCRIPTION	Blooms' Taxonomy Level
CO1	Identify technically and economically feasible problems (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO2	Identify and survey the relevant literature for getting exposed to related solutions and get familiarized with software development processes (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO3	Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions of minimal complexity by using modern tools & advanced programming techniques (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO4	Prepare technical report and deliver presentation (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO5	Apply engineering and management principles to achieve the goal of the project (Cognitive Knowledge Level: Apply)	Level 3: Apply

CO-PO AND CO-PSO MAPPING

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO3
CO1	3	3	3	3		2	2	3	2	2	2	3	2	2	2
CO2	3	3	3	3	3	2		3	2	3	2	3	2	2	2
CO3	3	3	3	3	3	2	2	3	2	2	2	3			2
CO4	2	3	2	2	2			3	3	3	2	3	2	2	2
CO5	3	3	3	2	2	2	2	3	2		2	3	2	2	2

3/2/1: high/medium/low

JUSTIFICATIONS FOR CO-PO MAPPING

MAPPING	LOW/ MEDIUM/ HIGH	JUSTIFICATION
100003/CS622T.1-PO1	**HIGH**	Identify technically and economically feasible problems by applying the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
100003/CS622T.1-PO2	**HIGH**	Identify technically and economically feasible problems by analysing complex engineering problems reaching substantiated conclusions using first principles of mathematics.
100003/CS622T.1-PO3	**HIGH**	Design solutions for complex engineering problems by identifying technically and economically feasible problems.
100003/CS622T.1-PO4	**HIGH**	Identify technically and economically feasible problems by analysis and interpretation of data.
100003/CS622T.1-PO6	**MEDIUM**	Responsibilities relevant to the professional engineering practice by identifying the problem.
100003/CS622T.1-PO7	**MEDIUM**	Identify technically and economically feasible problems by understanding the impact of the professional engineering solutions.
100003/CS622T.1-PO8	**HIGH**	Apply ethical principles and commit to professional ethics to identify technically and economically feasible problems.
100003/CS622T.1-PO9	**MEDIUM**	Identify technically and economically feasible problems by working as a team.
100003/CS622T.1-PO10	**MEDIUM**	Communicate effectively with the engineering community by identifying technically and economically feasible problems.
100003/CS622T.1-P011	**MEDIUM**	Demonstrate knowledge and understanding of engineering and management principles by selecting the technically and economically feasible problems.
100003/CS622T.1-PO12	**HIGH**	Identify technically and economically feasible problems for long term learning.
100003/CS622T.1-PSO1	**MEDIUM**	Ability to identify, analyze and design solutions to identify technically and economically feasible problems.
100003/CS622T.1-PSO2	**MEDIUM**	By designing algorithms and applying standard practices in software project development and Identifying technically and economically feasible problems.
100003/CS622T.1-PSO3	**MEDIUM**	Fundamentals of computer science in competitive research can be applied to Identify technically and economically feasible problems.
100003/CS622T.2-PO1	**HIGH**	Identify and survey the relevant by applying the knowledge of mathematics, science, engineering fundamentals.

100003/CS622T.2-PO2	**HIGH**	Identify, formulate, review research literature, and analyze complex engineering problems get familiarized with software development processes.
100003/CS622T.2-PO3	**HIGH**	Design solutions for complex engineering problems and design based on the relevant literature.
100003/CS622T.2-PO4	**HIGH**	Use research-based knowledge including design of experiments based on relevant literature.
100003/CS622T.2-PO5	**HIGH**	Identify and survey the relevant literature for getting exposed to related solutions and get familiarized with software development processes by using modern tools.
100003/CS622T.2-PO6	**MEDIUM**	Create, select, and apply appropriate techniques, resources, by identifying and surveying the relevant literature.
100003/CS622T.2-PO8	**HIGH**	Apply ethical principles and commit to professional ethics based on the relevant literature.
100003/CS622T.2-PO9	**MEDIUM**	Identify and survey the relevant literature as a team.
100003/CS622T.2-PO10	**HIGH**	Identify and survey the relevant literature for a good communication to the engineering fraternity.
100003/CS622T.2-PO11	**MEDIUM**	Identify and survey the relevant literature to demonstrate knowledge and understanding of engineering and management principles.
100003/CS622T.2-PO12	**HIGH**	Identify and survey the relevant literature for independent and lifelong learning.
100003/CS622T.2-PSO1	**MEDIUM**	Design solutions for complex engineering problems by Identifying and survey the relevant literature.
100003/CS622T.2-PSO2	**MEDIUM**	Identify and survey the relevant literature for acquiring programming efficiency by designing algorithms and applying standard practices.
100003/CS622T.2-PSO3	**MEDIUM**	Identify and survey the relevant literature to apply the fundamentals of computer science in competitive research.
100003/CS622T.3-PO1	**HIGH**	Perform requirement analysis, identify design methodologies by using modern tools & advanced programming techniques and by applying the knowledge of mathematics, science, engineering fundamentals.
100003/CS622T.3-PO2	**HIGH**	Identify, formulate, review research literature for requirement analysis, identify design methodologies and develop adaptable & reusable solutions.

100003/CS622T.3-PO3	**HIGH**	Design solutions for complex engineering problems and perform requirement analysis, identify design methodologies.
100003/CS622T.3-PO4	**HIGH**	Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
100003/CS622T.3-PO5	**HIGH**	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools.
100003/CS622T.3-PO6	**MEDIUM**	Perform requirement analysis, identify design methodologies and assess societal, health, safety, legal, and cultural issues.
100003/CS622T.3-PO7	**MEDIUM**	Understand the impact of the professional engineering solutions in societal and environmental contexts and Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions.
100003/CS622T.3-PO8	**HIGH**	Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions by applying ethical principles and commit to professional ethics.
100003/CS622T.3-PO9	**MEDIUM**	Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.
100003/CS622T.3-PO10	**MEDIUM**	Communicate effectively with the engineering community and with society at large to perform requirement analysis, identify design methodologies.
100003/CS622T.3-PO11	**MEDIUM**	Demonstrate knowledge and understanding of engineering requirement analysis by identifying design methodologies.
100003/CS622T.3-PO12	**HIGH**	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change by analysis, identify design methodologies and develop adaptable & reusable solutions.
100003/CS622T.3-PSO3	**MEDIUM**	The ability to apply the fundamentals of computer science in competitive research and prior to that perform requirement analysis, identify design methodologies.
100003/CS622T.4-PO1	**MEDIUM**	Prepare technical report and deliver presentation by applying the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
100003/CS622T.4-PO2	**HIGH**	Identify, formulate, review research literature, and analyze complex engineering problems by preparing technical report and deliver presentation.

100003/CS622T.4-PO3	**MEDIUM**	Prepare Design solutions for complex engineering problems and create technical report and deliver presentation.
100003/CS622T.4-PO4	**MEDIUM**	Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions and prepare technical report and deliver presentation.
100003/CS622T.4-PO5	**MEDIUM**	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools and Prepare technical report and deliver presentation.
100003/CS622T.4-PO8	**HIGH**	Prepare technical report and deliver presentation by applying ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
100003/CS622T.4-PO9	**HIGH**	Prepare technical report and deliver presentation effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.
100003/CS622T.4-PO10	**HIGH**	Communicate effectively with the engineering community and with society at large by prepare technical report and deliver presentation.
100003/CS622T.4-PO11	**MEDIUM**	Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work by prepare technical report and deliver presentation.
100003/CS622T.4-PO12	**HIGH**	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change by prepare technical report and deliver presentation.
100003/CS622T.4-PSO1	**MEDIUM**	Prepare a technical report and deliver presentation to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas.
100003/CS622T.4-PSO2	**MEDIUM**	To acquire programming efficiency by designing algorithms and applying standard practices in software project development and to prepare technical report and deliver presentation.
100003/CS622T.4-PSO3	**MEDIUM**	To apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs by preparing technical report and deliver presentation.
100003/CS622T.5-PO1	**HIGH**	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
100003/CS622T.5-PO2	**HIGH**	Identify, formulate, review research literature, and analyze complex engineering problems by applying engineering and management principles to achieve the goal of the project.

100003/CS622T.5-PO3	**HIGH**	Apply engineering and management principles to achieve the goal of the project and to design solutions for complex engineering problems and design system components or processes that meet the specified needs.
100003/CS622T.5-PO4	**MEDIUM**	Apply engineering and management principles to achieve the goal of the project and use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
100003/CS622T.5-PO5	**MEDIUM**	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO6	**MEDIUM**	Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities by applying engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO7	**MEDIUM**	Understand the impact of the professional engineering solutions in societal and environmental contexts, and apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO8	**HIGH**	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice and to use the engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO9	**MEDIUM**	Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO11	**MEDIUM**	Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team. Manage projects in multidisciplinary environments and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO12	**HIGH**	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PSO1	**MEDIUM**	The ability to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas. Apply engineering and management principles to achieve the goal of the project.

100003/CS622T.5-PSO2	**MEDIUM**	The ability to acquire programming efficiency by designing algorithms and applying standard practices in software project development to deliver quality software products meeting the demands of the industry and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PSO3	**MEDIUM**	The ability to apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur and apply engineering and management principles to achieve the goal of the project.