StableAI

CMPE 492
Low-Level Design Report
15.03.2023

Arda Atakol 13624005832
Serdar Hoşver 10061964142
Furkan Özelge 14758028780
Yağız Hikmet Karakuş 44164883604

URL of the webpage: https://furkanozelge.github.io/seniorstedu/

Table of Contents

1. Introduction 4

1.1. Object Design Trade-Offs 5

1.1.1. Accuracy vs. Speed: 5
1.1.2. Storage vs. Efficiency: 5
1.1.3. User Interface vs. Functionality: 5
1.1.4. Data Privacy vs. User Experience: 5
1.1.5. Scalability vs. Performance: 5

1.2. Engineering Standards 6

1.3. Definitions, Acronyms, and Abbreviations 8

1.3.1. Stable Diffusion: 8
1.3.2. AI (Artificial Intelligence): 8
1.3.3. Deep learning: 8
1.3.4. Text-to-image model: 8
1.3.5. Inpainting: 8
1.3.6. Out painting: 8
1.3.7. Image-to-image translation: 8
1.3.8. User base: 8
1.3.9. Sustainability: 8
1.3.10. Risk management: 9
1.3.11. Style Transfer: 9

2. Packages 9

2.1. ReactJS: 9

2.2. NextJS: 9

2.3. ChakraUI: 9

2.4. Bootstrap: 9

2.5. Axios: 9

2.6. React-Redux: 9

2.7. Sass: 9

2.8. FastAPI: 9

2.9. Postman: 9

2.10. Mongoose: 10

2.11. TensorFlow: 10

2.12. Pandas: 10

2.13. NumPy: 10

2.14. Torch: 10

2.15. Transformers: 10

2.16. OpenCV: 10

2.17. Diffusers: 10

2.18. Accelerate: 10

3. Class Interfaces 11

Component 1: Text Processor 11

Component 2: Image Generator 11

Component 3: Data Loader 11

Component 4: Style Transfer 12

Component 5: Sign Up 12

Component 6: Login 12

Component 7: Image Sharing 13

Component 8: Guidance Scale Processor 13

Component 9: Recommendation 13

Component 10: Upload Image 14

Component 11: Download Image 14

Component 12: Image Rating 14

4. Glossary 15

5. References 16

1. Introduction

This is a Stable Diffusion project which is the program that convert the text to the image. A text-to-image model using deep learning is called Stable Diffusion. Although it can be used for various tasks including inpainting, outpainting and creating image-to-image translations directed by text prompts, its primary usage is to generate detailed visuals conditioned on text descriptions. So, with this information, we will create an application & tool that uses deep learning and artificial intelligence, which creates a design idea for the use of designers by using stable diffusion together with a model of pre-trained open-source data. While doing this, we will work on adding various features by going beyond what has been done. Stable Diffusion will reveal very creative ideas and images for designers. In this way, we will give designers a new perspective.

Application will be two window application which has text box window and the login page window. After the users logon the application, they will see the text-box window and when the user enters his/her text to the text box, the program will convert it to the image. For example, when you enter the input ''a photograph of an astronaut riding a horse'', application will read that command and pull the related images from trained data and it will show the equivalent image. User will see a horse with the astronaut after that input.

Our overall goal is to introduce stable diffusion and AI to all users and make a good impression. The ability of people to draw their dreams to artificial intelligence with just a few words or sentences. At this point, our general scope is to reach the entire user base. Some reasons arising from this may lead the project to some restrictions. In order to appeal to the entire user base, there should be no +18 content in this AI project in general. This will have a positive impact on our market value. Because one of the rules of the market is to expand the user base. This will contribute to our budget because we will make a profit that will reduce the cost in the project. In addition, all users will have the right to produce a limited number of photos. Because it costs us a certain cost in each photo generate. For this reason, it is very important to have a border in the middle. At the same time, we will do this by registering and logging in. All users must have an account. There are many reasons for this. We also need this so that we can limit the number of uses. We will also offer suggestions based on previous searches in user accounts, while providing suggested photos to users. We will also get this with the user account. In addition, we will have a very strict limit on terrorism and violence. It will be forbidden to use words of this variety.

Time is also one of the most important issues in a project. This project is done in a team of 4 people and it will cost 4 people to work in various areas of this project for 4-5 months. We need to reflect this time positively on the project as quality. All team members will devote at least 10-15 hours per week to this project. It is very important to be able to use this as quality. If we can't spend the time we spend on making this project a successful one, it will cause a great waste of time. For this reason, the tasks and each branch of the project will be carefully divided and finished with a regular study. 3 Of course, like every project, this project has some risks. It is our

greatest desire to make a project worth the effort we spend, the time we spend and the money we spend. But for this, various risk management should be done. We should consider what measures we can take in various scenarios. We will see this with various tests and reviews before it reaches the end user. On top of that, we will make the project perfect with many updates. All projects would be perfect with updates. We will fix the bugs in the first versions in the next versions. In this way, we will have a sustainable project. Sustainability is the most important thing for a project. Our project is not a project that can be consumed quickly. All kinds of users will get magnificent designs and drawings by pushing the limits of imagination.

Stable diffusion is not a subject that has too much diversity as a resource. For this reason, we will be very limited on research. But in this case, we will step in. In scientific studies or projects, you should be a resource yourself where the resource ends. For this reason, we will not be afraid to develop new technologies ourselves in places where there are no resources.

1.1. Object Design Trade-Offs

1.1.1. Accuracy vs. Speed: The accuracy of the generated image will be determined by the size and complexity of the pre-trained data model used in the Stable Diffusion algorithm. However, a more complex model will require more computational resources, resulting in slower processing times. Therefore, the trade-off between accuracy and speed needs to be carefully considered.

1.1.2. Storage vs. Efficiency: The pre-trained data model required for Stable Diffusion is quite large in size. The model's size may result in storage issues on the user's computer, leading to slower loading times. Therefore, it may be necessary to consider a smaller, more efficient model that can still produce high-quality images.

1.1.3. User Interface vs. Functionality: The user interface design should be intuitive and user-friendly, allowing users to easily input their text and view the resulting image. However, the functionality of the application should not be sacrificed for the sake of the user interface design. Finding the right balance between the two is essential.

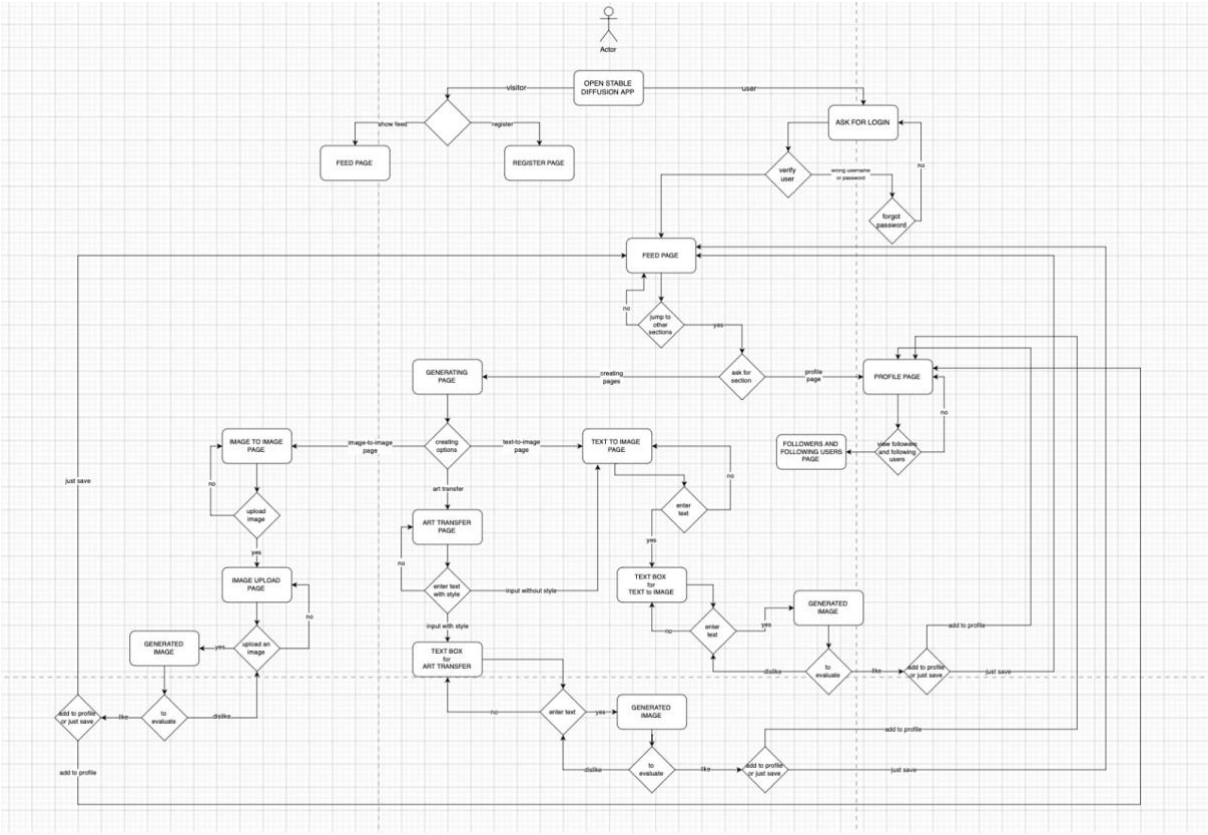
1.1.4. Data Privacy vs. User Experience: The application will need to access and use pre-trained data from third-party sources to generate the images. However, data privacy should be considered and protected, while still ensuring that users have a positive experience using the application.

1.1.5. Scalability vs. Performance: As the application grows in popularity and usage, it may be necessary to consider scalability to accommodate a larger user base. However, the application's performance should not be compromised by adding more users. Finding the right balance between scalability and performance is essential.

1.2. Engineering Standards

The Stable Diffusion project will adhere to several IEEE engineering standards to ensure the quality and reliability of the software system.

The following UML Diagram will be used for the project:



A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system. In our UML diagram, we have an actor which is using Stable Diffusion Application. First the actor open the app and he/she has to choose the entry type as a visitor or as an user. If he/she is a visitor, app direct the user to the visitor page and let him/her to select the path register page or feed page. If he/she open the app as an user, app ask her/him for login. If he/she is an user, user must verify user. If the username or password is wrong, user direct to the forgot password section. Otherwise, user direct to the feed page. In this point, user see the sections and he/she has to select one of these 2 sections. These are generating page and profile page. In profile page, user can view the followers and following users. In the other section (Generating Page), user has some creating options which are Image to Image, Text to Image and Art Transfer page. In Image-to-Image part, first the user uploads an image to the image upload page. After that the program gives the generated image to the user to evaluate. If the user like the generated image, he/she

can add it to the profile or just save it. If the user doesn't like the generated image and dislike it, program returns to upload an image part. Suppose that the user liked the generated image. In this point he/she has two options which are just save and add to profile. If the user just saves the generated image, program direct the user to the feed page. Otherwise, if the user selects the add to profile option, the program directs the user to the profile page. Our second creating option is Art Transfer. In this page, user enters text with style. If the input and the style are not valid, program direct the user to the text to image option. Suppose that the user entered valid text with style. In that point user goes to Text Box for Art Transfer. After that user enters a text and program shows the generated image for him/her to evaluate. Again, the program gives the user 2 options as a like or dislike. If the user dislike the generated image, it returns to enter text part. But if the user like the image, he/she may add to profile this image or just save it same as Image-to-Image part. If the user wants to add the generated image to profile, it directs to profile page. Otherwise, it directs to feed page. The last option is text to image part. In this option, user enters a text and program directs the user to the text box for text to image. User enters a text and see the generated image to evaluate. Remaining part is same as other options. If the user dislikes it, it returns to enter text part, if the user like the generated image, he/she see the question, add to profile or just save? If the user want to add the image to the profile program directs the user to the profile page and if the user want just save the image, program directs the user to the feed page.

The following IEEE standards will be used for the project:

IEEE 830: Recommended Practice for Software Requirements Specifications
IEEE 1016: Recommended Practice for Software Design Descriptions
IEEE 1063: Recommended Practice for Software User Documentation
IEEE 1012: Standard for System and Software Verification and Validation
IEEE 1471: Recommended Practice for Architectural Description of Software-Intensive Systems

IEEE 830: standard will be used to define the software requirements specification (SRS). This standard will help us to clearly and unambiguously define the functional and non-functional requirements for the Stable Diffusion system. The SRS will be organized into sections that clearly define the scope of the system, its requirements, and its design constraints. The SRS document will be the basis for evaluating the success of the Stable Diffusion project.

The IEEE 1016: standard will be used to document the software design description (SDD). This standard will help us to create a clear and concise description of the architecture and design of the Stable Diffusion system. The SDD document will describe the system components, their interactions, and the data flow between them. The SDD will also provide details on the software interfaces, algorithms, and other design elements that are critical for the Stable Diffusion system.

The IEEE 1063: standard will be used to define the software user documentation. This standard will help us to create user-friendly and easy-to-understand documentation for the Stable

Diffusion system. The user documentation will include detailed instructions on how to use the system, as well as troubleshooting tips and other helpful information.

The IEEE 1012: standard will be used to define the system and software verification and validation plan (SVVP). This standard will help us to create a comprehensive plan for testing the Stable Diffusion system. The SVVP document will define the testing objectives, methods, and procedures for each phase of the testing process. It will also specify the criteria for determining whether the system has met its requirements.

IEEE 1471: standard will be used to define the architectural description of the Stable Diffusion system. This standard will help us to create a clear and comprehensive description of the system architecture, including its components, their interactions, and their relationships. The architectural description will be used to guide the development process and ensure that the system meets its requirements.

In summary, the Stable Diffusion project will adhere to several IEEE engineering standards to ensure the quality and reliability of the software system. The standards will be used to define the software requirements specification, software design description, software user documentation, system and software verification and validation plan, and architectural description. Adherence to these standards will help us to develop a stable, reliable, and effective text-to-image model using deep learning for the Stable Diffusion application.

1.3. Definitions, Acronyms, and Abbreviations

1.3.1. Stable Diffusion: A text-to-image model using deep learning that can be used for various tasks including inpainting, outpainting, and creating image-to-image translations directed by text prompts.

1.3.2. AI (Artificial Intelligence): The ability of machines to perform tasks that typically require human-like intelligence, such as learning, problem-solving, and decision-making.

1.3.3. Deep learning: A subset of machine learning that involves training artificial neural networks on large datasets in order to enable the system to learn and make decisions on its own.

1.3.4. Text-to-image model: A machine learning model that is trained on a dataset of text descriptions and images and is able to generate images based on text input.

1.3.5. Inpainting: The process of filling in missing or damaged parts of an image.

1.3.6. Out painting: The process of generating new images based on a given image or set of images.

1.3.7. Image-to-image translation: The process of generating a new image based on a given image, with the goal of preserving certain features or characteristics of the original image while changing others.

1.3.8. User base: The group of users who are using or potentially interested in using a product or service.

1.3.9. Sustainability: The ability of a system or process to be maintained or continued over time without degrading or depleting resources.

1.3.10. Risk management: The process of identifying, assessing, and prioritizing risks and developing strategies to mitigate or eliminate those risks.

1.3.11. Style Transfer: Style transfer is a technique used to transfer the style of one image or piece of text to another. This can be done through a variety of methods, such as neural networks or other machine learning algorithms. The resulting output is a new image or text that has the content of the original input but is styled like the reference image or text.

2. Packages

2.1. ReactJS: A JavaScript library for building user interfaces. It allows developers to build reusable UI components and update the UI dynamically based on changes in data. ReactJS is used to build web applications and mobile applications.

2.2. NextJS: A framework for building server-side rendered React applications. NextJS provides developers with a set of tools and conventions for building React applications that can be rendered both on the server and the client.

2.3. ChakraUI: A set of UI components and styles for React applications. ChakraUI provides developers with a library of customizable UI components that follow accessibility best practices and can be easily integrated into React applications.

2.4. Bootstrap: A front-end framework for building responsive, mobile-first web applications. Bootstrap provides developers with a set of pre-built HTML, CSS, and JavaScript components that can be customized to create a consistent user interface across different devices.

2.5. Axios: A JavaScript library for making HTTP requests. Axios provides developers with an easy-to-use API for sending HTTP requests and handling responses, making it a popular choice for building web applications.

2.6. React-Redux: A library that provides a predictable state container for React applications. React-Redux allows developers to manage the state of their application in a consistent and predictable way, making it easier to build and maintain complex applications.

2.7. Sass: A CSS preprocessor that extends the capabilities of CSS. Sass provides developers with a set of tools for creating reusable stylesheets with variables, functions, and mixins.

2.8. FastAPI: A modern, fast web framework for building APIs with Python. FastAPI provides developers with a high-performance web framework for building APIs that can handle large volumes of requests.

2.9. Postman: A tool for testing APIs. Postman allows developers to send requests to APIs, inspect the responses, and automate the testing process.

2.10. Mongoose: A library for working with MongoDB in Node.js applications. Mongoose provides developers with an easy-to-use API for interacting with MongoDB, including support for data modeling and validation.

2.11. TensorFlow: An open-source machine learning library developed by Google. TensorFlow provides developers with a set of tools for building and training machine learning models.

2.12. Pandas: A data manipulation library for Python. Pandas provides developers with a set of tools for working with tabular data, including data cleaning, filtering, and aggregation.

2.13. NumPy: A library for numerical computing in Python. NumPy provides developers with a set of tools for working with arrays and matrices, including support for mathematical operations and linear algebra.

2.14. Torch: A machine learning library for Python. Torch provides developers with a set of tools for building and training deep learning models.

2.15. Transformers: A library for natural language processing (NLP) developed by Hugging Face. Transformers provides developers with a set of pre-trained models for common NLP tasks, including text classification, named entity recognition, and question answering.

2.16. OpenCV: A computer vision library for Python. OpenCV provides developers with a set of tools for working with images and videos, including support for image processing, feature detection, and object recognition.

2.17. Diffusers: A library for generating visualizations of code changes. Diffusers allows developers to see changes in code over time, making it easier to track changes and identify bugs.

2.18. Accelerate: A library for high-performance computing in Swift. Accelerate provides developers with a set of tools for optimizing performance in Swift applications, including support for numerical computing and image processing.

3. Class Interfaces

The Stable Diffusion project will be implemented using a set of classes that are responsible for different tasks within the system. The following is a list of the classes that will be used in the system along with their interfaces:

Component 1: Text Processor

The Text Processor class is responsible for processing the input text and converting it into a format that can be understood by the Image Generator. The following is a description of the interface for the Text Processor class:

Input: A string representing the user's input text.
Output: A list of keywords extracted from the embedded text.

The Text Processor class will extract relevant keywords from the input text that will be used by the Image Generator to generate the corresponding image.

Component 2: Image Generator

The Image Generator class is responsible for generating the image based on the input keywords provided by the Text Processor. The following is a description of the interface for the Image Generator class:

Input: A list of keywords extracted from the input text.
Output: An image corresponding to the input keywords.

The Image Generator class will use deep learning algorithms to generate a detailed image that corresponds to the input keywords provided by the Text Processor. The output image will be displayed to the user via the application's user interface.

Component 3: Data Loader

The Data Loader class is responsible for loading the pre-trained open-source data required for the Style Transfer component to generate images. The following is a description of the interface for the Data Loader class:

Input: User’s uploaded images
Output: A pre-trained model for generating images using style transfer.

The Data Loader class will load the pre-trained data required by the Style Transfer component to generate images based on the user's input.

Component 4: Style Transfer

The Style Transfer class is responsible for generating an image based on the user's input and the pre-trained model loaded by the Data Loader component. The following is a description of the interface for the Style Transfer class:

Input: A image representing the user's input image.
Output: An image corresponding to the user's input image generated using style transfer.

The Style Transfer class will use deep learning algorithms to generate an image that corresponds to the user's input image based on the pre-trained model loaded by the Data Loader component.

Component 5: Sign Up

The Sign Up class is responsible for creating a new user account in the system. The following is a description of the interface for the Sign Up class:

Input: A string representing the user's email address and a string representing the user's password.
Output: JSON Web Token.

The Sign Up class will create a new user account in the system using the provided email address and password.

Component 6: Login

The Login class is responsible for verifying a user's credentials and allowing access to the system. The following is a description of the interface for the Login class:

Input: A string representing the user's email address and a string representing the user's password.
Output: A boolean value indicating whether the login checking with JSON Web Token was successful.

The Login class will verify the user's credentials and allow access to the system if the provided email address and password are correct.

Component 7: Image Sharing

The Generated Image Sharing class is responsible for allowing users to share generated images with other users in the system. The following is a description of the interface for the Generated Image Sharing class:

Input: An image generated by the Style Transfer or text-to-image/image-to-image component.
Output: None.

The Generated Image Sharing class will allow users to share the generated image with other users in the system.

Component 8: Guidance Scale Processor

The Guidance Scale Processor class is responsible for processing the guidance scales for the image generation process. The following is a description of the interface for the Guidance Scale Processor class:

Input: An integer for guidance scales.
Output: None.

The Guidance Scale Processor class will process the guidance scales for the image generation process to ensure that the generated image meets the user's expectations.

Component 9: Recommendation

The Recommendation To User Depends on User's Keywords class is responsible for providing recommendations to users based on their input keywords. The following is a description of the interface for the Recommendation To User Depends on User's Keywords class:

Input: A list of keywords.
Output: A list of recommended post.

The Recommendation To User Depends on User's Keywords class will provide recommendations to users based on their input keywords to improve their image generation experience.

Component 10: Upload Image

The Image Uploader class is responsible for uploading images to the Image Generator for image-to-image translation. The following is a description of the interface for the Image Uploader class:

Input: An image file.
Output: A message indicating whether the upload was successful or not.

The Image Uploader class will allow users to upload images that can be used as input for image-to-image translations by the Image Generator.

Component 11: Download Image

The Image Downloader class is responsible for downloading generated images. The following is a description of the interface for the Image Downloader class:

Input: An image ID or image file.
Output: The corresponding image file.

The Image Downloader class will allow users to download generated images that they find interesting or useful.

Component 12: Image Rating

The Image Rating class is responsible for allowing users to rate generated images as either like or dislike. The following is a description of the interface for the Image Rating class:

Input: An image ID and a rating (like or dislike).
Output: A message indicating whether the rating was successfully recorded or not(consol).

The Image Rating class will allow users to provide feedback on the generated images, which can be used to improve the system's performance in the future.

By following these class interfaces, the Stable Diffusion project will ensure that the different components of the system can communicate with each other and work together to provide a seamless user experience.

4. Glossary

AI (Artificial Intelligence): A field of computer science focused on creating intelligent machines that can perform tasks that usually require human intelligence, such as visual perception, speech recognition, decision-making, and language translation.

Deep Learning: A subset of machine learning that involves building and training neural networks with multiple layers to extract features from data and make predictions.

Image-to-Image Translation: The process of transforming an input image into an output image of a different style or appearance, such as converting a sketch to a realistic image or changing the color of an object in an image.

Inpainting: The process of filling in missing or damaged parts of an image with plausible content.

Neural Network: A type of machine learning model inspired by the structure and function of the human brain, composed of layers of interconnected nodes or neurons that process and transform input data.

Outpainting: The process of generating new content outside the boundaries of an input image, such as adding a background to a foreground object or extending an image in a specific direction.

Pre-trained Model: A machine learning model that has been trained on a large dataset and can be used as a starting point for fine-tuning on a specific task or dataset.

Stable Diffusion: A text-to-image model that uses deep learning and diffusion-based probabilistic modeling to generate realistic and detailed images conditioned on text descriptions.

Text-to-Image Generation: The process of generating images from textual descriptions, such as converting a sentence or a paragraph into a corresponding image.

User Interface (UI): The visual and interactive elements of a software application that enable users to interact with the system and perform tasks.

Recommendation: Recommendations for similar or related content based on a user's previous search history.

5. References

https://www.acm.org/code-of-ethics

https://standards.ieee.org/

https://en.wikipedia.org/wiki/Low-level_design

http://klabs.org/history/history_docs/sp-8070/ch4/4p1_design_tradeoffs.htm

https://uxdesign.cc/how-to-make-smart-design-tradeoffs-9210cbd8ae1d

https://walkingtree.tech/everything-need-know-low-level-design-sdlc/

https://invoke-ai.github.io/InvokeAI/features/OUTPAINTING/

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/

https://www.lucidchart.com/blog/types-of-UML-diagrams

https://en.wikipedia.org/wiki/Unified_Modeling_Language

https://jwt.io/