Text to Scene Generator

Via Artificial Intelligence

(A Web-Based Application)

By

Wasi Haider

Supervised By

Dr. Onaiza Maqbool

Department of Computer Science

Quaid-i-Azam University

Islamabad

Session (2015-2019)


بسم الله الرحمن الرحيم

Acknowledgment

In the name of ALLAH the most beneficent and most merciful. Thanks ALLAH who gave me enough wisdom, knowledge and strength to complete this milestone and to complete the project this far. First of all I am thankful to my parents, my brother and my uncle who have supported me at each step from my childhood to today. Then I am so thankful to my worthy supervisor and head of department **Dr. Onaiza Maqbool**, without her assistance this was difficult for me, she made it easy for me by her guidance. She always believed in me.

Special thanks to Dr. Shoaib karim, Dr. SM Naqi, Dr.Asim Rafiq, Miss Memoona Afsheen, Miss Ifrah farrukh, Dr. Mudassar Azam Sindhu, Dr. Khalid Salim and all visiting teachers for their cooperation during my degree.

I also want to thank all workers, lab assistants. There are some people who have encouraged me in every way, I have accomplished this with their support as well, so special thanks to Hafiz M. Aon, Syed Hannan Shah, Saifullah uz zaman, Wajahat Ali, Huzaifa Sheikh, Ahsan Sheraz, Faheem Nawaz and specially Sikandar waheed who has guided me in this journey. These people made this memorable for me.

Wasi Haider

2015-2019

Abstract

Text to scene generator is a web-based application powered by artificial intelligence. This system will be helpful for story writers and comic book writers to generate scenes for their story. When we tell someone a story or we read it, we picture the scene in our minds but we can't see it actually. In this proposed system the user will be able to give text input in natural language and then the system will generate a scene according to the objects specified in the sentences given and then using artificial intelligence it will parse the sentence and generate a tree for parent objects to the children objects with their positional relations. For each object in the tree it will place the image of the object to main scene according to the relation found with other objects. Finally the user will be able to save the scene as it will be 2D image. There will be an administrator side which will allow the administrator to add images of the objects and sentences for model training.

Table of Contents

List of Figures

List of Tables

Chapter 1: Project Management Plan

1 SOFTWARE PROJECT MANAGEMENT PLAN

1.1 Introduction

This introduction provides background information for the rest of the document. It briefly describe the project, the client deliverables, the project plan, the tasks for completion of the project and the project milestones.

1.1.1 Project Overview

The Text to Scene Generator is web based artificial intelligence tool which reduce the effort in such cases where story writers add pictures to depict a scene especially for children story books. This product will help the story writers to generate scenes for their stories.

1.1.2 Project Deliverables

There are two deliverables for this project

- Project Documentation by the end of this semester (Fall 2018).
- Implementation of the project with complete documentation by end of next semester (Spring 2019).

1.2 Project Organization

Project organization consists of software process model, roles and responsibilities and tools and techniques.

1.2.1 Software Process Model

I'll use Incremental Model as a software process model for this project due to following reasons:

- This model is flexible, less costly to change scope and requirements.
- It is easier to test and debug during an iteration.

- It is easy to check the progress after each iteration.
- Generate working software earlier during software lifecycle.

1.2.2 Roles and Responsibilities

As this project is assigned to a single person so I'll be doing everything related to this project.

1.2.3 Tools and Techniques

Following are the tools used in this project

- MS Word
- ProjectLibre
- ARGO UML
- PyCharm (IDE)

1.3 Project Management Plan

Following tasks will be performed for the completion of this project:

		Name	Duration	Start	Finish	Predecessors	Resource Names
1		Create Text to Scene Generator	158 days	11/12/18 8:00 AM	6/19/19 5:00 PM		
2		Problem understanding	1 day	11/12/18 8:00 AM	11/12/18 5:00 PM		
3		Make Software Project Management Plan	4 days	11/13/18 8:00 AM	11/16/18 5:00 PM	2	Wasi;PC;MS Word
4		Write Introduction	1 day	11/13/18 8:00 AM	11/13/18 5:00 PM		
5		Define Project Organization	2 days	11/13/18 8:00 AM	11/14/18 5:00 PM		
6		Define Project Management Plan	2 days	11/15/18 8:00 AM	11/16/18 5:00 PM	5	Project Libre
7		Make Requirements document	58 days	11/12/18 8:00 AM	1/30/19 5:00 PM	6	Wasi;PC;MS Word
8		Make Software Requirement Specification Document	25 days	11/12/18 8:00 AM	12/14/18 5:00 PM		
9		Give Introduction and Overview	1 day	11/12/18 8:00 AM	11/12/18 5:00 PM		
10		Define Scope	1 day	11/12/18 8:00 AM	11/12/18 5:00 PM		
11		Define Purpose and objective	1 day	11/12/18 8:00 AM	11/12/18 5:00 PM		
12		Review and refine scope and plan	2 days	11/13/18 8:00 AM	11/14/18 5:00 PM	11	Madam Omaiza
13		Identify Specific Requirements	1 day	11/15/18 8:00 AM	11/15/18 5:00 PM	12	
14		Explain External Interfaces	1 day	11/16/18 8:00 AM	11/16/18 5:00 PM	13	
15		Identify Use Cases	2 days	11/19/18 8:00 AM	11/20/18 5:00 PM	14	
16		Make UseCase Diagram	1 day	11/21/18 8:00 AM	11/21/18 5:00 PM	15	ArgoUML
17		Review and Refine UC Diagram	2 days	11/22/18 8:00 AM	11/23/18 5:00 PM	16	Madam Omaiza
18		Define UseCase descriptions	2 days	11/22/18 8:00 AM	11/23/18 5:00 PM	16	
19		Review and Refine UC Description	2 days	11/26/18 8:00 AM	11/27/18 5:00 PM	18	Madam Omaiza
20		Define System Attributes	2 days	11/26/18 8:00 AM	11/27/18 5:00 PM	18	
21		Identify DataBase Requirements	1 day	11/28/18 8:00 AM	11/28/18 5:00 PM	20	
22		Identify Entities	2 days	11/29/18 8:00 AM	11/30/18 5:00 PM	21	Wasi
23		Make Entity Relation Diagram	2 days	12/3/18 8:00 AM	12/4/18 5:00 PM	22	Wasi;PC;ArgoUML
24		Review and Refine ERD	2 days	12/5/18 8:00 AM	12/6/18 5:00 PM	23	Madam Omaiza
25		Make System Sequence Diagrams	2 days	12/7/18 8:00 AM	12/10/18 5:00 PM	24	ArgoUML
26		Review and refine SSD	2 days	12/11/18 8:00 AM	12/12/18 5:00 PM	25	Madam Omaiza

Figure 1.1Project Plan Part 1

		Name	Duration	Start	Finish	Predecessors	Resource Names
27		Make Domain Model	2 days	12/11/18 8:00 AM	12/12/18 5:00 PM	25	ArgoUML
28		Review and Refine SRS	2 days	12/13/18 8:00 AM	12/14/18 5:00 PM	27	Madam Onaiza
29		**Make Software Design Description Document**	**16 days**	**12/17/18 8:00 AM**	**1/7/19 5:00 PM**	**28**	**Wasi;PC;MS Word**
30		Give Introduction and Overview	1 day	12/17/18 8:00 AM	12/17/18 5:00 PM		
31		Make Activity Diagrams	4 days	12/17/18 8:00 AM	12/20/18 5:00 PM		ArgoUML
32		Review and Refine Activity Diagram	2 days	12/21/18 8:00 AM	12/24/18 5:00 PM	31	Madam Onaiza
33		Make System Architectural Design	2 days	12/17/18 8:00 AM	12/18/18 5:00 PM		ArgoUML
34		Review and Refine Architecture Diagram	2 days	12/19/18 8:00 AM	12/20/18 5:00 PM	33	Madam Onaiza
35		Make Sequence Diagrams	2 days	12/21/18 8:00 AM	12/24/18 5:00 PM	33;34	ArgoUML
36		Review and Refine SD	2 days	12/25/18 8:00 AM	12/26/18 5:00 PM	35	Madam Onaiza
37		Identify Classes	2 days	12/27/18 8:00 AM	12/28/18 5:00 PM	36	
38		Make Class Diagram	2 days	12/31/18 8:00 AM	1/1/19 5:00 PM	37	ArgoUML
39		Review and Refine Class Diagram	2 days	1/2/19 8:00 AM	1/3/19 5:00 PM	38	Madam Onaiza
40		Review and Refine Software Design Description	2 days	1/4/19 8:00 AM	1/7/19 5:00 PM	39	Madam Onaiza
41		**Make Software Test Document**	**6 days**	**1/8/19 8:00 AM**	**1/15/19 5:00 PM**	**40**	
42		Make Test Cases	4 days	1/8/19 8:00 AM	1/11/19 5:00 PM		MS Word
43		Review and Refine Test Document	2 days	1/14/19 8:00 AM	1/15/19 5:00 PM	42	Madam Onaiza
44		Review Analysis and Design Document	10 days	1/16/19 8:00 AM	1/29/19 5:00 PM	43	Madam Onaiza
45		Provide 1st Deliverable	1 day	1/30/19 8:00 AM	1/30/19 5:00 PM	44	
46		**Project Implementation**	**100 days**	**1/31/19 8:00 AM**	**6/19/19 5:00 PM**	**45**	**PyCharm;Wasi;Madam O...**
47		Create user Interfaces(Front End)	10 days	1/31/19 8:00 AM	2/13/19 5:00 PM		Wasi;PyCharm
48		Create Database for Date set	40 days	1/31/19 8:00 AM	3/27/19 5:00 PM		Wasi;PyCharm
49		Create Back End(Agent)	30 days	2/14/19 8:00 AM	3/27/19 5:00 PM	47	Wasi;PyCharm
50		Make Database Connectivity	10 days	3/28/19 8:00 AM	4/10/19 5:00 PM	48;49	Wasi;PyCharm
51		Train Agent	50 days	4/11/19 8:00 AM	6/19/19 5:00 PM	50	Wasi;PC;PyCharm
52		Provide 2nd Deliverable	1 day	6/20/19 8:00 AM	6/20/19 5:00 PM	51	

Figure 1.2 Project Plan Part 2

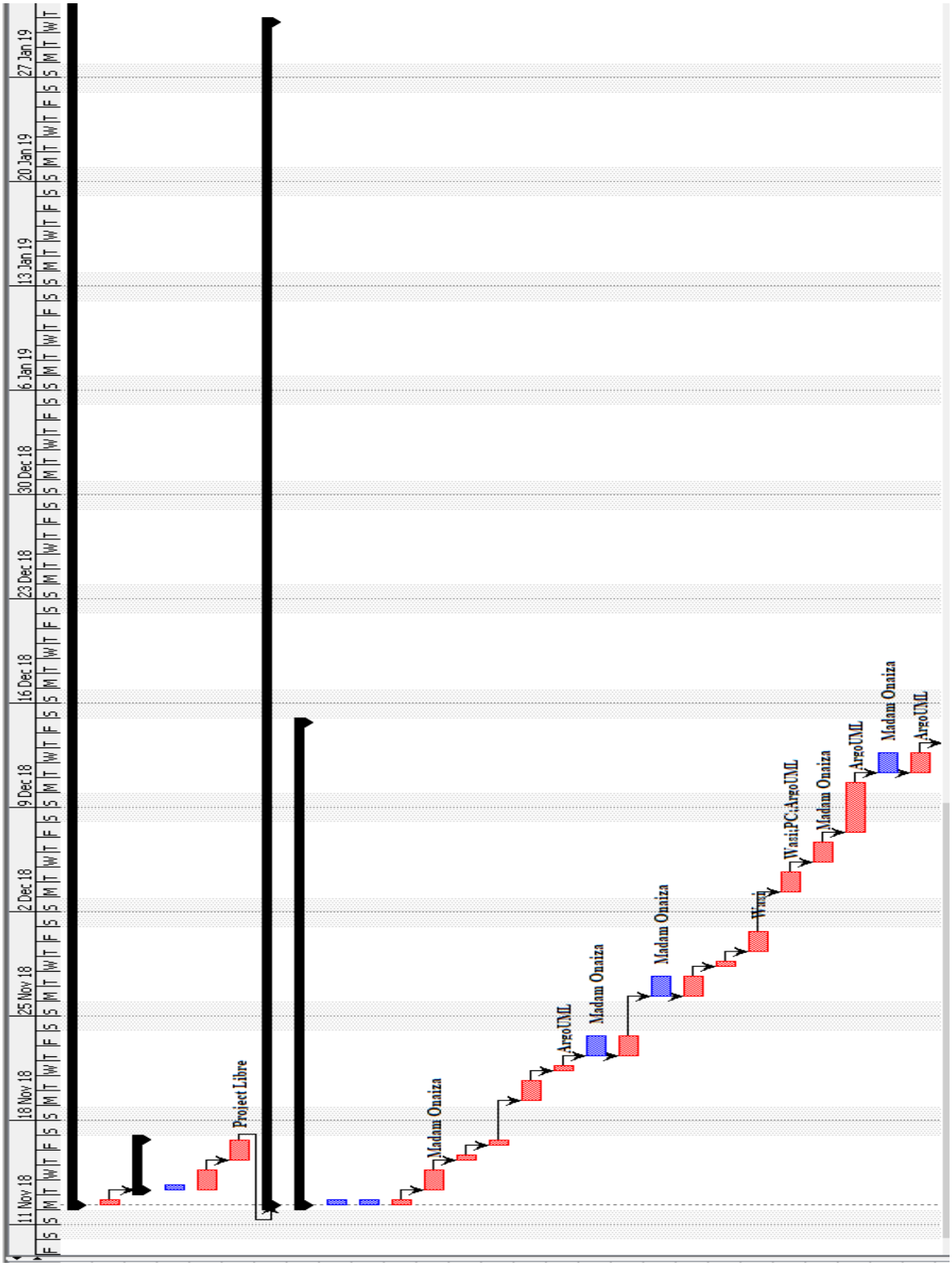

Figure 1.3 Gantt-Chart Part 1

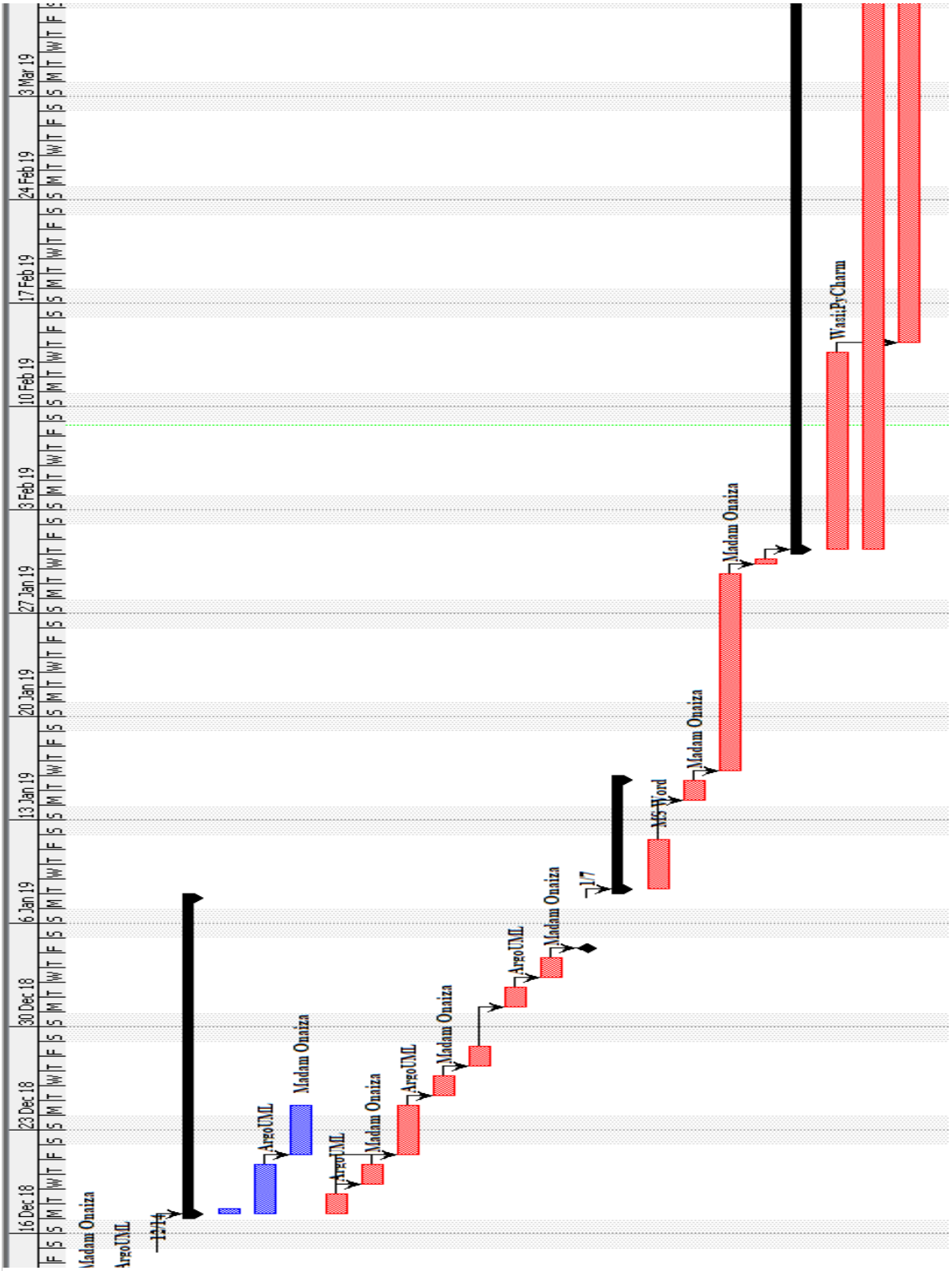

Figure 1.4 Gantt-Chart Part 2

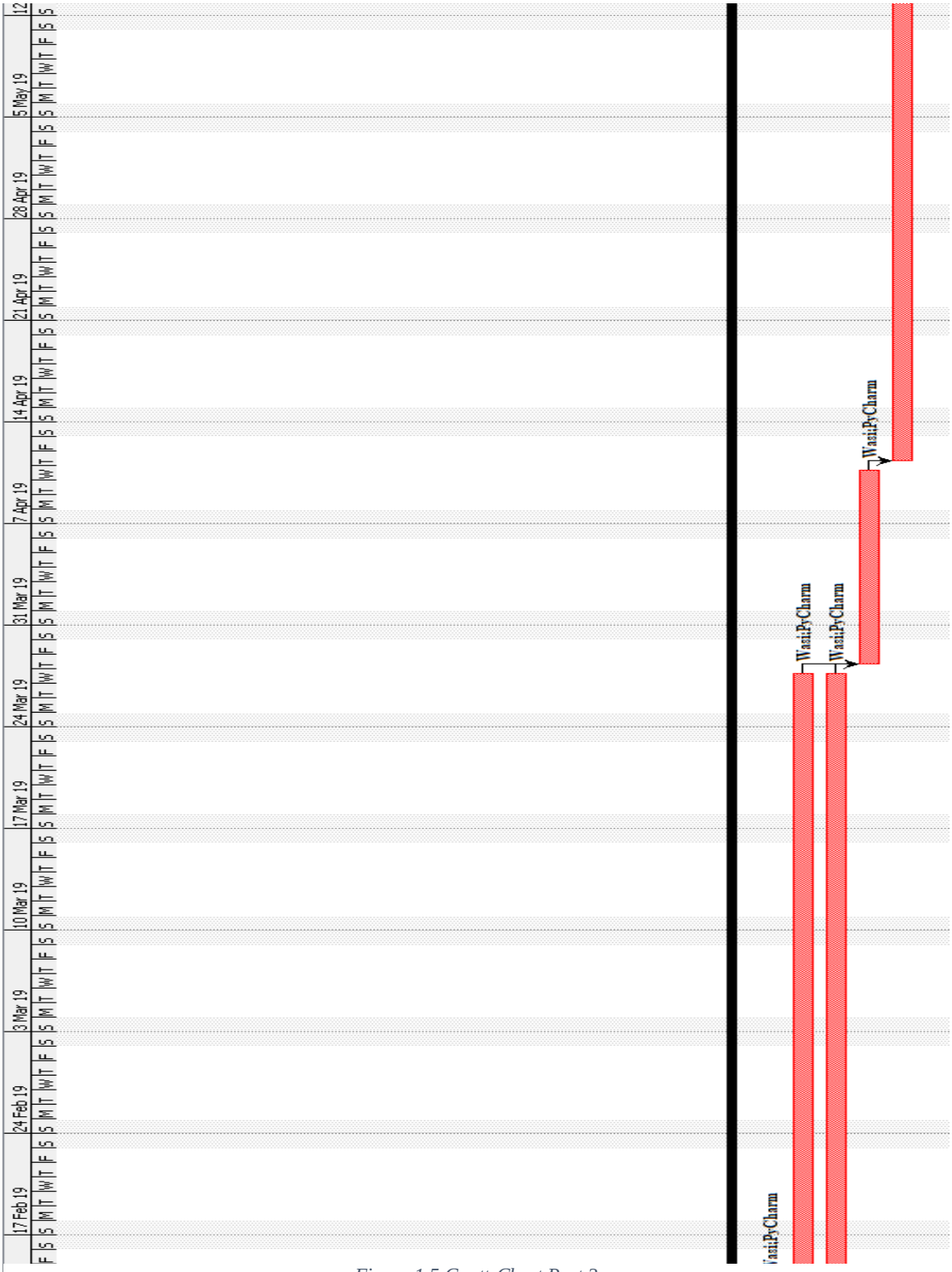

Figure 1.5 Gantt-Chart Part 3

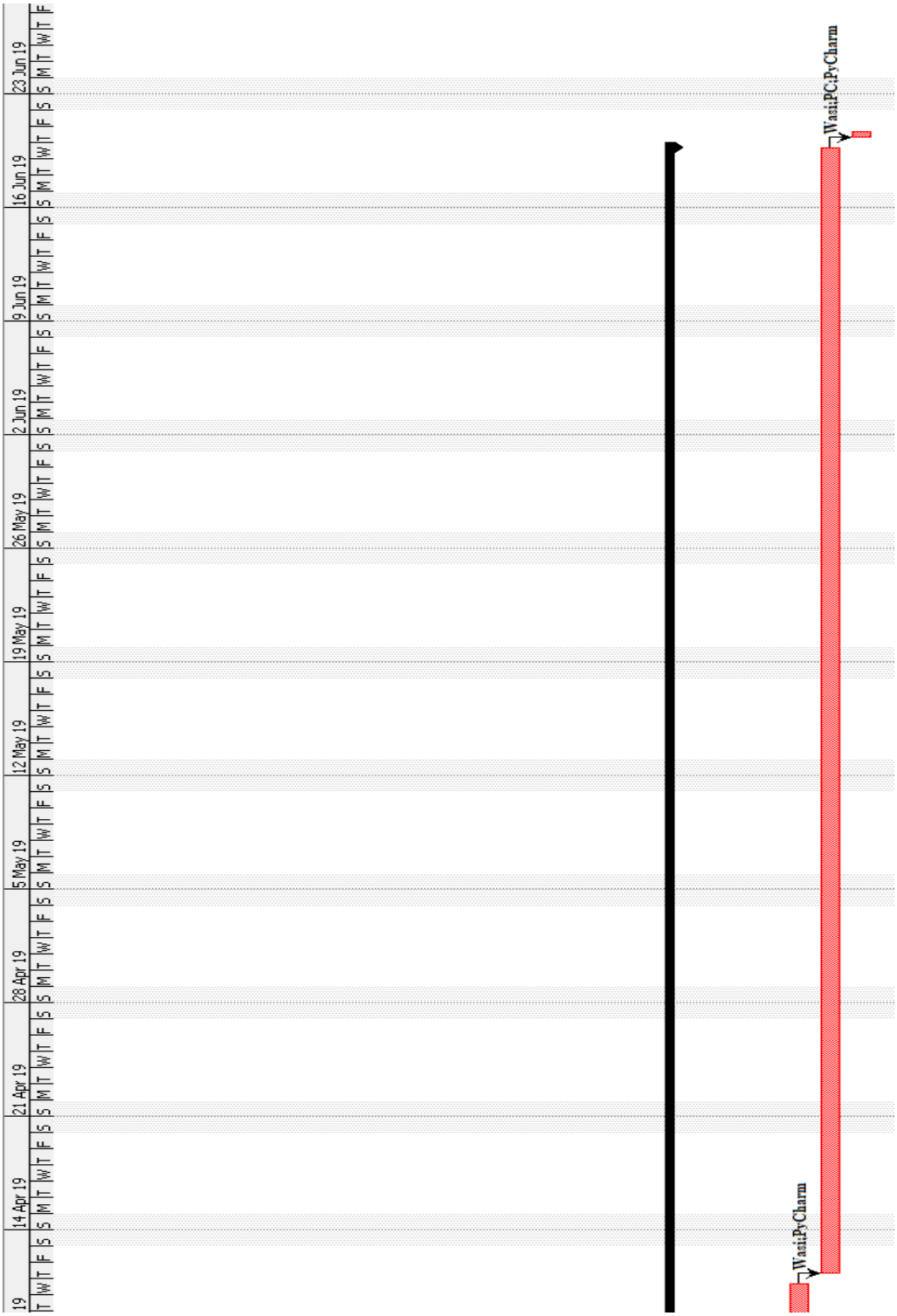

Figure 1.6 Gantt-Chart Part 4

1.3.1 Description and Understanding Project

This is the first task in which the project developer will understand the project and will make an overall description.

This task has no subtasks. It has no deliverable because it is only for the understanding of the project.

1.3.2 Make Software Project Management Plan

- **Description**

  In this task project plan will be made.

- **Deliverables and Milestone**
  - Its deliverable is SMTP Document
  - There is no milestone at this level

- **Dependencies**

  Dependency is shown in the figure 1.3.1.

- **Risk and Contingencies**

  There is no risk in this task.

1.3.3 Make Software Requirement Specification (SRS)

- **Description**

  In this task the software requirements will be defined

- **Deliverables and Milestone**
  - Its deliverable is SRS Document
  - There is a milestone after completion of this task and previous task (SPMP).

- **Dependencies**

  Its dependency is the previous task which is SMTP Document. The Dependency is shown in figure 1.3.1.

- **Risk and Contingencies**

  There is no risk in this task.

1.3.4 Make Software Design Description

- **Description**

  In this task design description of the product will be defined.

- **Deliverables and Milestone**
  - Its deliverable is SDD Document.
  - There is no milestone for this task.
- **Dependencies**

  Its dependency is the previous task which is SRS Document. The dependency is shown in figure 1.3.2.

- **Risk and Contingencies**

  There is no risk in this task.

1.3.5 Make Software Test Document (STD)

- **Description**

  In this task the software test document will be defined

- **Deliverables and Milestone**
  - Its deliverable is STD and whole project documentation.
  - There is a milestone after this task.
- **Dependencies**

  The dependency is shown in figure 1.3.2.

- **Risk and Contingencies**

  There is no risk in this task.

1.3.6 Assignments

As this project is assigned to a single person so I'll be doing everything related to this project.

Chapter 2: Project Background

2 PROJECT BACKGROUND

This chapter describes the concept of Machine Learning and the machine learning techniques and the algorithms used for this project development.

2.1 Machine Learning

Machine Learning is an application of Artificial intelligence which enables the system to learn and make certain improvements from experience and we don't have to program this all. Machine learning is based on data and its analysis. On the base of analysis a machine learn. Self-Driving Cars and Google Assistant are the best example for machine learning [4].

2.2 Types of Machine Learning

Following are the types of machine learning on the basis of learning style [4]:

2.2.1 Supervised Learning

This technique provides machine a labeled dataset through which it learns future predictions and make decisions.

2.2.2 Unsupervised Learning

This technique provides machine unlabeled and unclassified data. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data.

2.2.3 Semi-supervised Learning

This is sometimes used for improving machine prediction accuracy. It somehow falls between both supervised and unsupervised learning. The machine is provided by both labeled and unlabeled data.

2.2.4 Reinforcement Learning

In this learning technique system interacts with its environment by producing actions and then discovers errors or rewards. The reward on trial helps the system learn.

2.3 Artificial Intelligence Techniques

This section describes the techniques used in development of this project

2.3.1 Natural Language Processing

Natural Language Processing (NLP) is branch of artificial intelligence which helps the system to learn human languages and understand their structure [5].

It is widely used for text to speech conversion and also for chatbots. NLP includes tokenization of the sentence into part of speech and parsing them according to part of speech tagging

In general terms, NLP tasks break down language into shorter, elemental pieces, try to understand relationships between the pieces and explore how the pieces work together to create meaning.

2.3.2 Sentence Parsing in Natural Language Processing

Sentence parsing in NLP is a technique of finding the structure of a language based on its grammar [6].

See this example grammar below, in this example each line of CFG indicates a rule of the grammar to be applied to example sentence “Tom ate an apple”. Here Tom is a noun and “ate” is verb and so on. The picture shows how it works.

```
sentence -> noun_phrase, verb_phrase
noun_phrase -> proper_noun
noun_phrase -> determiner, noun
verb_phrase -> verb, noun_phrase
proper_noun -> [Tom]
noun -> [apple]
verb -> [ate]
determiner -> [an]
```

Figure 2.1 Grammar Example

The outcome of the grammar after is the parse tree which indicates the structure of the sentence by making a root node and its dependent leaves. The root node is the base of the sentence which then breaks it to other parts of the sentence to finally making a tree of whole sentence.

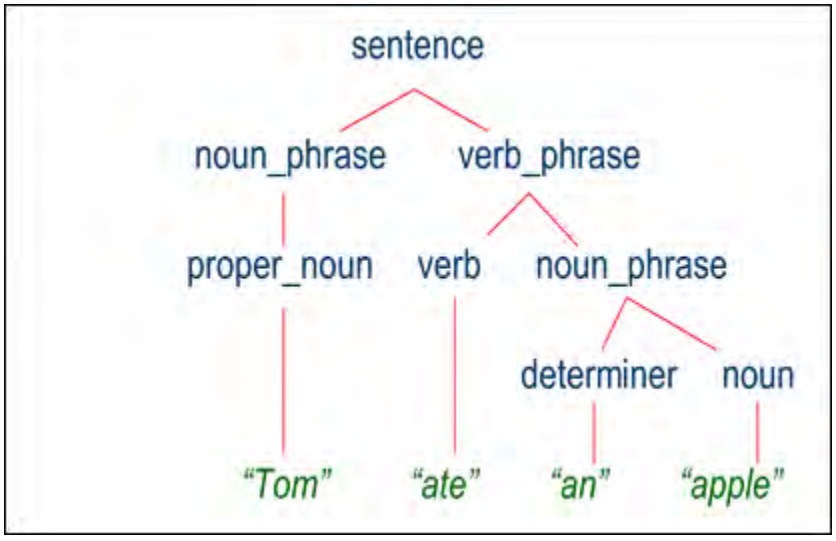

Figure 2.2 Parse Tree

2.3.3 Semantic Sentence Parsing in Natural Language Processing

Semantic parsing is technique of converting a natural language sentence to a logical form. It makes a parse tree but it is based on the main objects involve in the sentence and then it finds out the relation between the objects used in the sentence [7].

It looks for the dependencies between the objects.

For example “There is a table in the room. A plate is on the table and a cake is in the plate. A chair is next to table”. Following will be the parse tree

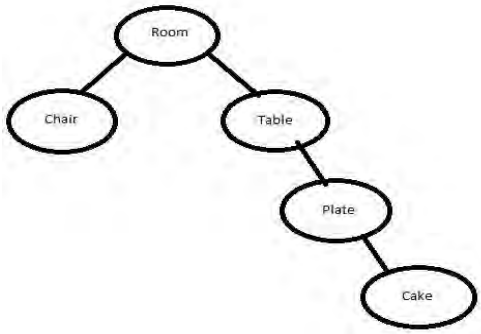

Figure 2.3 Parse Tree

2.3.4 Natural Language Toolkit (NLKT)

NLTK is library to work with python to make NLP projects. It has a good interface for classification, tokenization, stemming, and tagging, parsing, and semantic reasoning, wrappers for industrial-strength [8].

2.3.5 Classification in Artificial Intelligence

When a machine learns a dataset through learning techniques then it predicts the data came in future in forms of classes and categories which is called as classification [9].

In short, Classification algorithms let machines assign a category to a data point based on training data.

Chapter 3: Software Requirement Specification

3 SOFTWARE REQUIREMENTS SPECIFICATION

3.1 Introduction

This section gives a scope description and overview of everything included in this SRS document.

3.1.1 Purpose

The purpose of this document is to give a detailed description of the requirements for the "Text to Scene Generator" (TTSG) software. It will illustrate the purpose of the development of system. This document covers the functional as well non-functional requirements of the system. It covers the essential features of the system that are fixed, known, and agreed to be delivered. In addition to basic requirements, it will describe the external interface requirement, non-functional requirements and overall description, features.

3.1.2 Scope

The Project that is going to be presented in this document is named as "Text to Scene Generator". This is an Artificial Intelligent web application that can be used by any user.

This software system will allow the user to enter text as input to generate scene. The system will only be able to generate the scenes of room in house by looking at the target words in input text. The target words are those words which indicates the objects like furniture (Words are domain specific, the domain will be of 10 objects in a room). System will also look for the target words which tells the size and the location of objects (like near, below, left, right). The system will search a local database of the images according to target words. These target words will help the user to generate a suitable image for the text. There will be a database handler which can add images to the database.

This system will be completed by the end of Fall semester 2019.

3.1.3 Objective

The main target of the "Text to Scene Generator" are the story writers. The story writers often describe a scene in a story but it is only textual. It will make their story more interesting if they generate scenes through this system and put scenes along with their story. Definitions, Acronyms and Abbreviations

Table 3.1: Definitions

Term	Definition
TTSG	Text to Scene Generator
User	Someone who interacts with the system
Stakeholder	Any person who has interaction or has indirect interest with the system and is not a developer
UC	Use Case
TC	Test Case
SSD	System Sequence Diagram
SD	Sequence Diagram
ERD	Entity Relationship Diagram

3.1.4 Overview

The remainder of this document includes two sections. The second one provides an overview of the system functionality and system interaction with other systems. This section also introduces different types of stakeholders and their interaction with the system. Further, the section also mentions the system constraints and assumptions about the product.

The third section provides the requirements specification in detailed terms and a description of the different system interfaces.

3.2 Overall Description

This section will give an overview of the whole system. The system will be explained in its context to discuss the system interaction with other systems. It will also describe what type of stakeholders will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

3.2.1 Product Perspective

This product is independent and totally self-contained.

3.2.2 Product Function

The list of all major functions are given below:

- **Generate Scene**
  The System will allow user to generate scene according to the text the user provides.
- **Login**
  The System will allow only the admin to login to the system to manage software training model and images.
- **Add image**
  The system will allow the admin to add new images of the objects.
- **Train Model**
  The system will allow the admin to train the model on a dataset given by the Admin. The trained model classifies the sentences that will be added to the system database.

3.2.3 User Interfaces

Following will be the user interfaces:

- **Generate Scene**
  By using this interface the user will be able to generate scene. The input will be the text and the output will be the generated scene.
- **Login**
  By using this interface the Admin will be able to login to the system. The input will be the username and password provided to Admin.
- **Add Image**
  By using this interface the Admin will be able to add new images of the.
- **Train Model**

By using this interface the Admin will be able to train the model for sentence classification. The model then be able to know the parent child relation of objects.

3.2.4 Hardware Interface

As it is web based application so it will be run on any computer that supports web browsers.

3.2.5 Software Interface

The web browsers will be needed to access the software system as it is web based. It can be used on any operating system.

3.2.6 Communication Interface

Internet connection will be required for this software system to be used by the user.

3.2.7 User Characteristics

User should have a basic knowledge of using web browser and understands English language. There are two users of the Admin and common users. Common users can generate scenes and save the scenes. The Admin can add new images of the objects. Admin can also generate and modify scenes as a common user.

3.2.8 Constraint

The System takes the text input in just English Language. The target words will be of just 10 objects of room and just 8 prepositional words will be taken as target words for the relation. Moreover for object characteristics (adjectives) can also be taken which includes the size (big/small) and the color of the object.

3.2.9 Assumptions and Dependencies

As for now I am going to create this product and going to run this on my PC. So, the data size will be dependent on the size of the hard drive of my PC.

3.3 Specific Requirements

This section contains the functional requirements of this product. It gives detailed description of the product functions.

3.3.1 List of Use Cases

Following are the list of product functions (Use Cases):

Common Use Cases

- Generate Scene
- Save Scene

Admin Use Cases

- Login
- Add Image
- Delete Image
- Train Model
- Add Sentences

3.3.2 Use Case Diagram

A UML use case diagram is the primary form of system/software requirements for a new software program under developed. Use cases specify the expected behaviour (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (such as UML). A key concept of use case modeling is that it helps us design a system from end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior [12].

A use case diagram is usually simple. It does not show the detail of the use cases

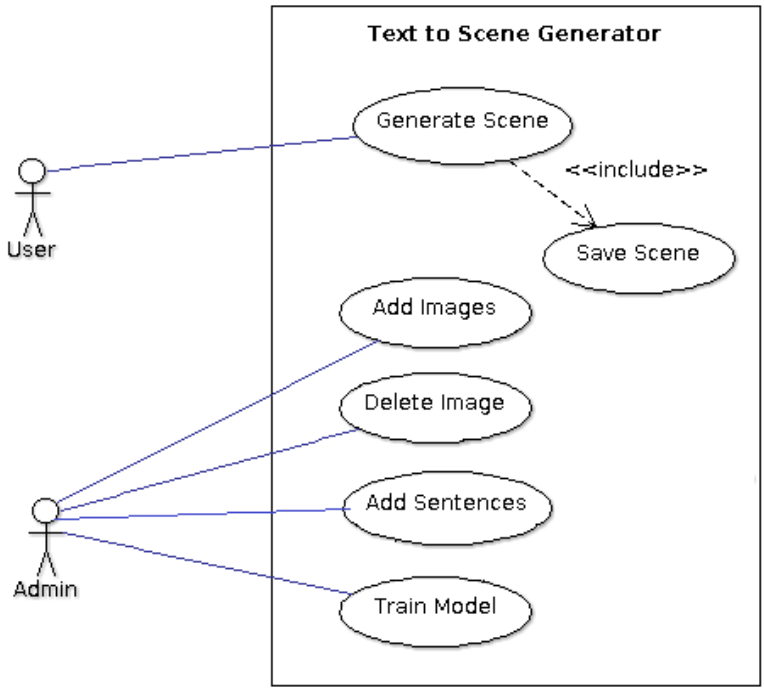

Figure 3.1 Use Case Diagram

3.3.3 Use Case Descriptions

UC-1: Generate Scene

Table 3.2 UC1: Generate Scene

Use Case No	UC1
Use Case Name	Generate Scene
Primary Actor	Common User
Pre-Condition	The user has opened the Text to Scene Generator
Post Condition	The System shows a generated scene and saved it
Main Success Scenario	1. System shows text input field 2. User enters text to generate a scene 3. User selects "Generate Scene" 4. System tokenizes the input text 5. System generates a parse tree according to dependency using semantic parsing 6. System fetches images of each target words (objects) 7. System plot the images according to the dependencies between the objects 8. System locate for the target words and according to the target words it generates a scene 9. The user save the image via use case UC3: Save Scene
Alternate Scenario	2 a). User enter the object name that is not in the list shown 1. The system shows a message "Please Enter the object name from the list given".

UC-2: Save Scene

Table 3.3 UC2: Save Scene

Use Case No	**UC3**
Use Case Name	Save Scene
Primary Actor	Common User
Pre-Condition	The user has generated a scene
Post Condition	The user successfully saves the scene
Main Success Scenario	1. User selects "Save Scene" 2. System shows download dialogue box 3. User select "browse" and selects the directory to save the scene 4. System saves the image to the selected directory
Alternate Scenario	3 a). User does not browse the directory 1. System saves the image in default downloads directory

UC-3: Login

Table 3.4 UC3: Login

Use Case No	**UC4**
Use Case Name	Login
Primary Actor	Admin
Pre-Condition	The user has opened the Text to Scene Generator
Post Condition	The user is successfully logged in
Main Success Scenario	1. User selects “login” 2. System shows the login form 3. User enters username and password 4. User selects “login” 5. System verifies the username and password 6. System redirects to the admin interface
Alternate Scenario	4 a). User entered username or password are wrong 1. System shows a message “username or password is wrong”

UC-4: Add Image

Table 3.5 UC4: Add Image

Use Case No	UC5
Use Case Name	Add Image
Primary Actor	Admin
Pre-Condition	The user is logged in
Post Condition	The user successfully added image to the database
Main Success Scenario	1. User selects “Add Image” 2. System shows add image form 3. User selects “browse” 4. System shows operating system’s directory popup” 5. User select the image 6. User selects “upload” 7. System saves the image
Alternate Scenario	6 a). User does not selected any image yet 1. System shows the message “select an image first”

UC-5: Delete Image

Table 3.6 UC5: Delete Image

Use Case No	**UC6**
Use Case Name	Delete Image
Primary Actor	Admin
Pre-Condition	The user is logged in
Post Condition	The user successfully deletes the image
Main Success Scenario	1. User selects "Delete Image" 2. System shows the list of categories 3. User selects a category 4. System shows list of images in the selected category 5. User selects an image and selects "Delete" 6. System deletes the image

UC-6: Train Model

Table 3.7 UC6: Train Model

Use Case No	UC7
Use Case Name	Train Model
Primary Actor	Admin
Pre-Condition	None
Post Condition	The System will update the trained model
Main Success Scenario	1. User selects to "Train Model" 2. System shows Train Model Form 3. User enters the Epoch, learning rate and training parameters and selects "Start Training". 4. System shows the message "The training has been started."
Alternative Scenario	3a). There is already a training in progress. 1. System shows a message "there is already a training in progress. Wait for the training to be done".

UC-7: Add Sentences

Table 3.8 UC7: Add Sentences

Use Case No	**UC7**
Use Case Name	Add Sentences
Primary Actor	Admin
Pre-Condition	None
Post Condition	The System will add sentences for training
Main Success Scenario	1. User selects to "Add Sentences" 2. System shows to choose a file 3. User chooses the file 4. System shows the message "The sentences added."
Alternative Scenario	3a). There is already a training in progress. 2. System shows a message "there is already a training in progress. Wait for the training to be done".
Special Requirements	1. The file format should be 'csv'

3.3.4 Logical Data Requirements

The database will be on the server side. Following will be the data that the server manages

- **Dataset:**
  The model will be trained using a dataset which will be stored on the server side.
- **Trained Model:**
  Training model that is the output after training on the dataset will be stored in a file and will be used for recognizing the images in the future.

3.3.5 Software System Attributes

- **Availability**

  There is no constraint on the availability of this application as this a web application. As far as internet is connected the software will be available to use. So, the availability is 24 hours a day if the internet is connected and the server is running.

- **Security**

  There is no risk of security because the TTSG does not access any private data of the user and has no confidential data as well. There are just object images in the database. However the user has limited access as they can just generate modify and save the scene. Only the Admin can access the database of the software system.

- **Maintainability**
  During the development period, all the things will be properly documented so that we can easily make changes and upgrade our application. Also the software system will be flexible to upgrade.
- **Portability**

  As this is a web application so the user can access it through any browser from anywhere so it is portable

- **Reliability**

The probability of failure is zero. The system shall never crash, other than as the result of an operating system error. If there will any error occur then it will display an appropriate messages such that user will not feel any ambiguity while using this product. So over all reliability of system is approximately is more than 70-80%.

- **Performance**

  As it is web based application so its performance will be dependent on the speed of the internet. So it will show results if the internet speed is good or it may take time if the internet speed is poor.

3.3.6 System Sequence Diagrams

In software engineering, a system sequence diagram (SSD) is a sequence diagram that shows, for a particular scenario of a use case, the events that external actors generate, their order, and possible inter-system events. System sequence diagrams are visual summaries of the individual use cases [13].

SSD-1: Generate Scene

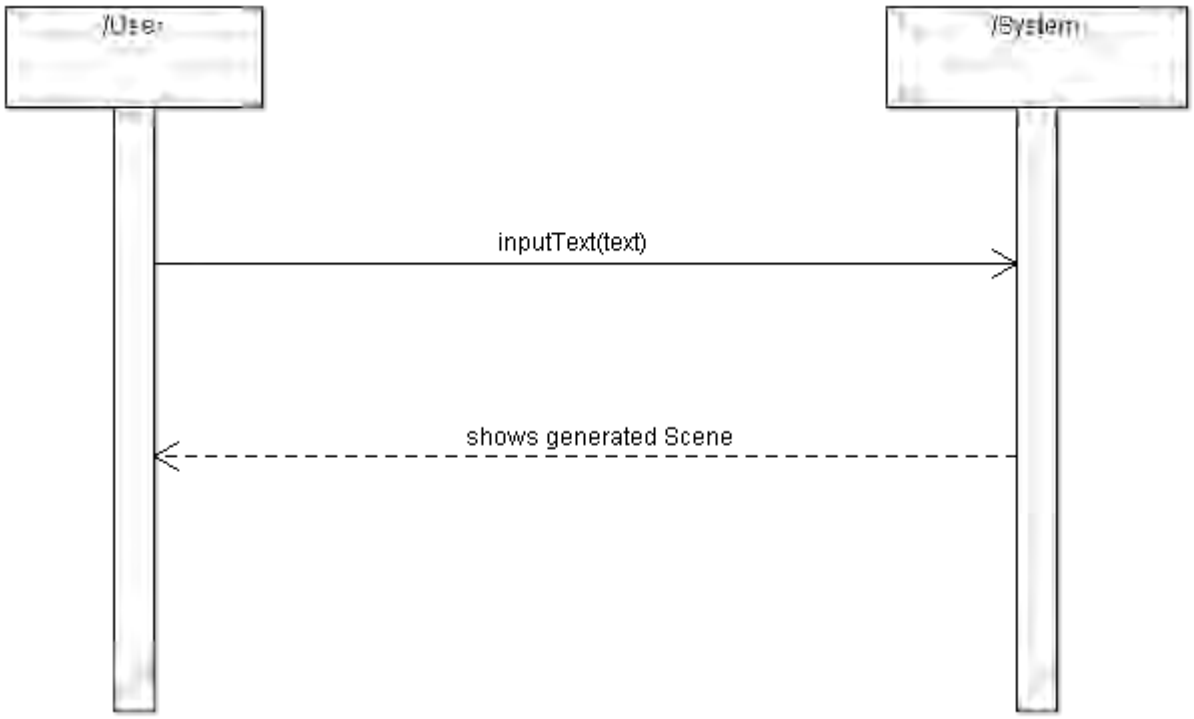

Figure 3.2 SSD-1: Generate Scene

SSD-2: Save Scene

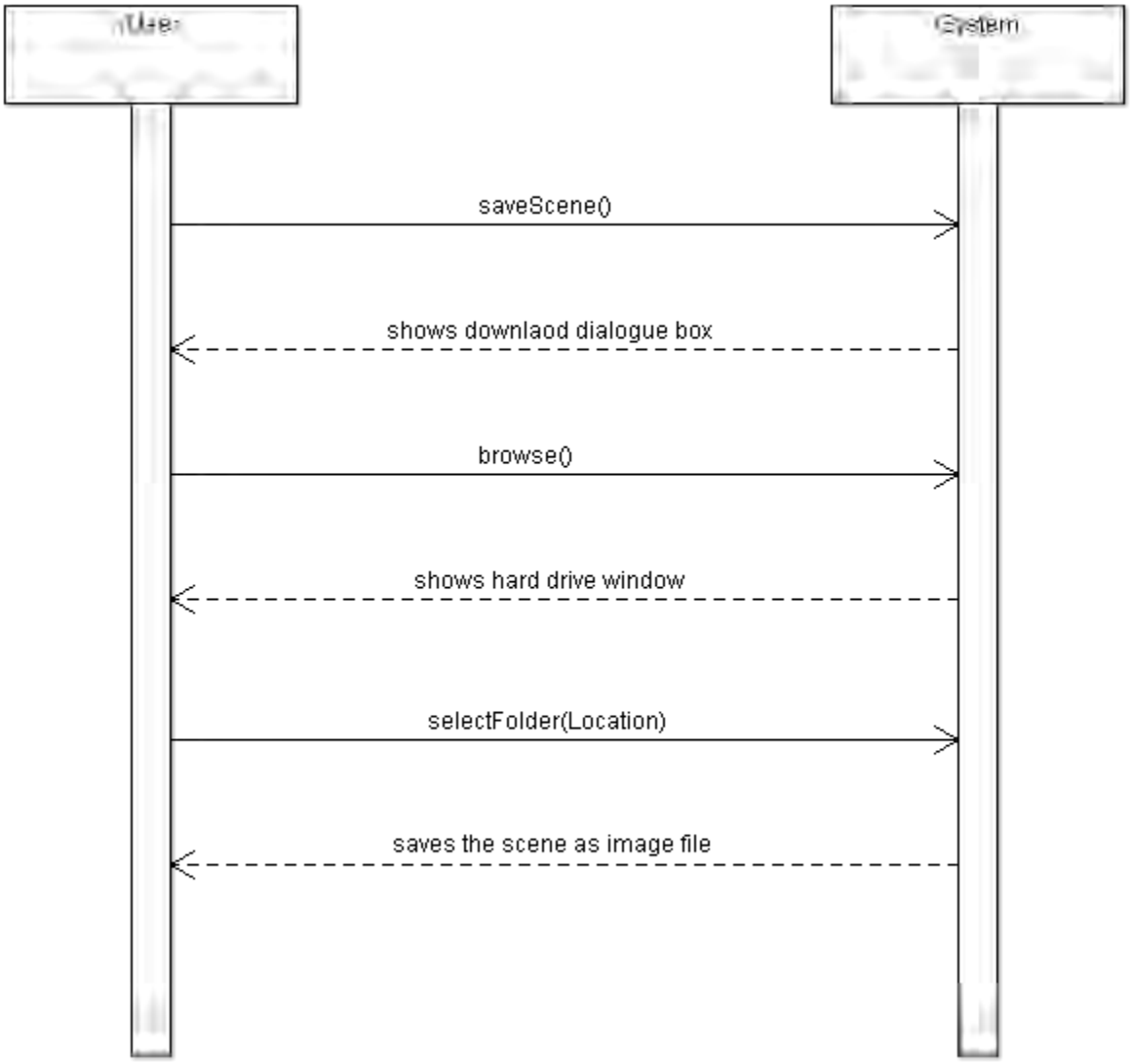

Figure 3.3 SSD-2: Save Scene

SSD-3: Add Image

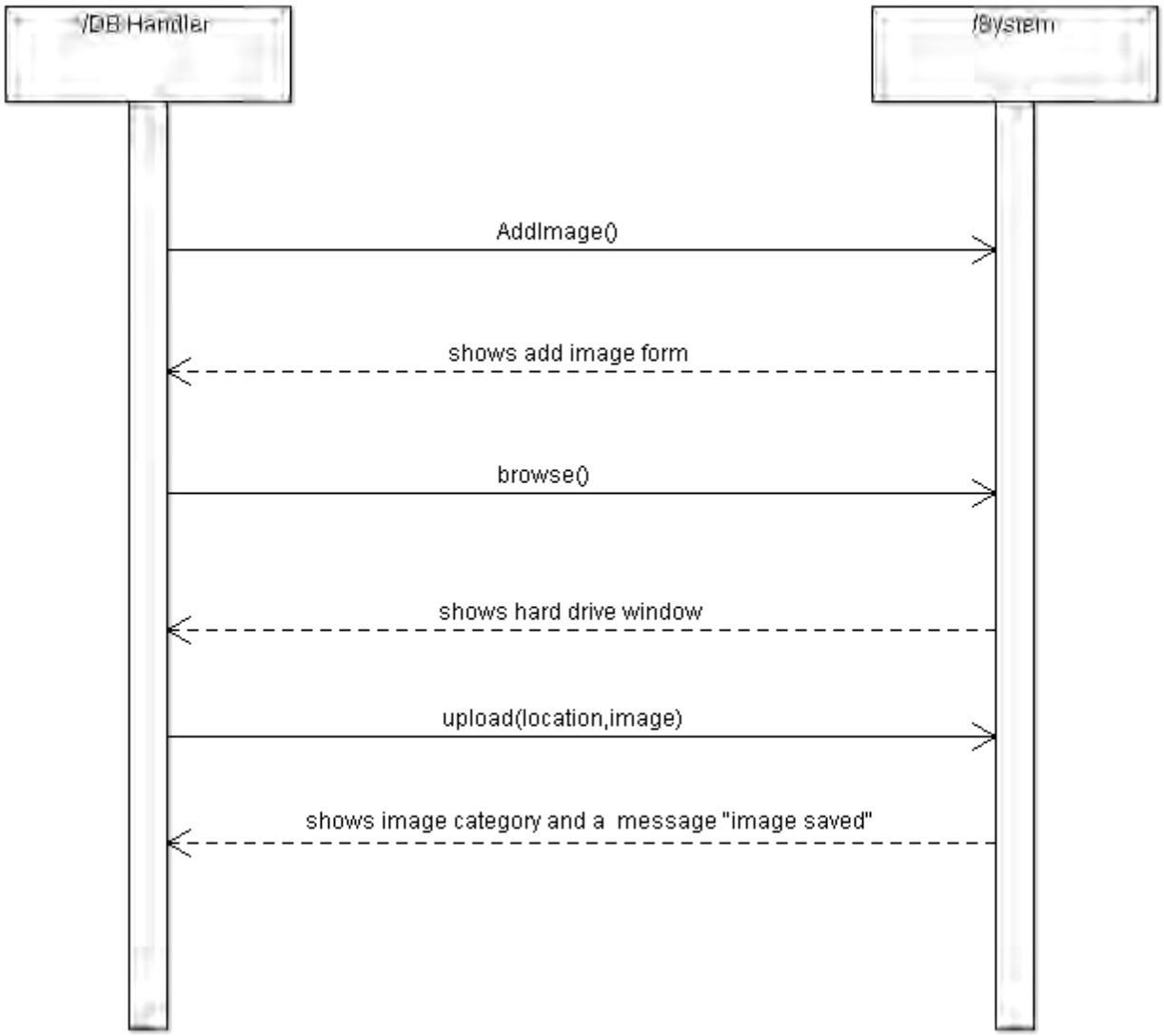

Figure 3.4 SSD-3: Add Image

SSD-4: Delete Image

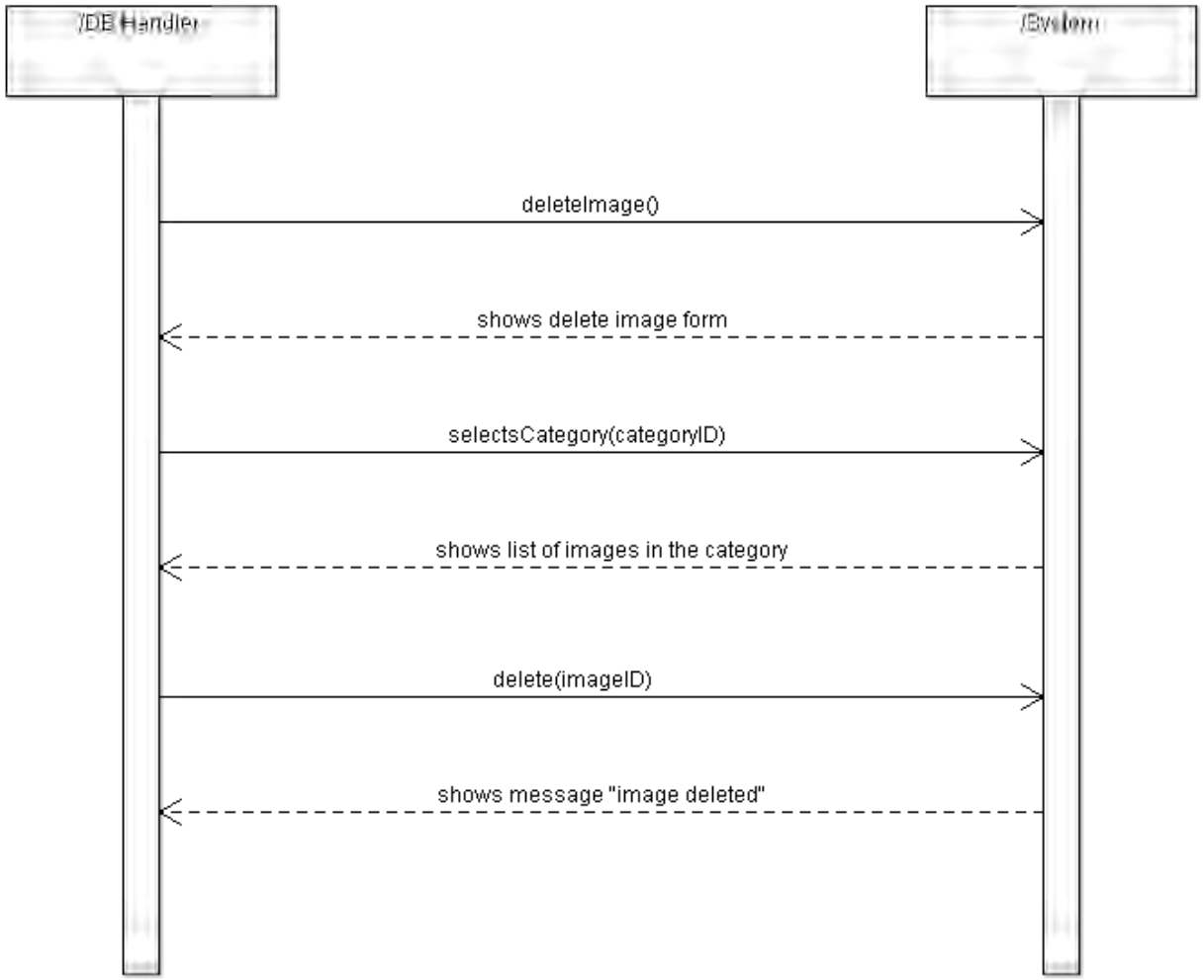

Figure 3.5 SSD-4: Delete Image

SSD-5: Train Model

Figure 3.6 SSD-5: Train Model

3.3.7 Domain Model

A domain model is a visual representation of conceptual classes or real - situation objects in a domain [M095, Fowler96]. Domain models have also been called conceptual models [14].

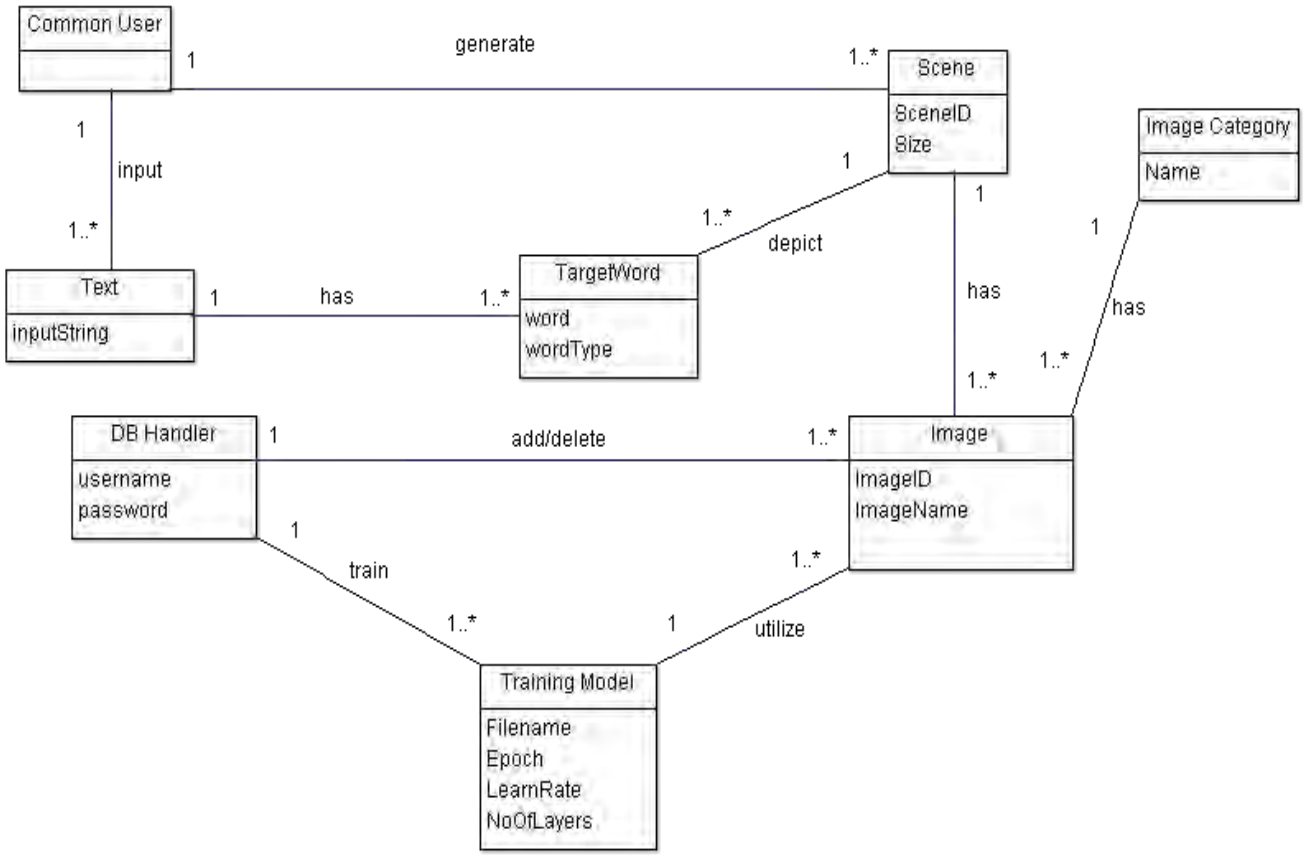

Figure 3.7 Domain Model

Chapter 4: Software Design Description

4 SOFTWARE DESIGN DOCUMENT

4.1 Introduction

The Software design document (SDD) describes the design for the development of this project Text to Scene Generator. This document provides description for the architecture and components involved in it. Moreover this document also provides the collaboration diagrams (Sequence Diagram and Class diagram). It also provides the user interfaces of the project to be developed.

4.1.1 Requirement Traceability Matrix

Requirement Traceability Matrix or RTM captures all requirements proposed by the client or software development team and their traceability in a single document delivered at the conclusion of the life-cycle [15].

In other words, it is a document that maps and traces user requirement with test cases. The main purpose of Requirement Traceability Matrix is to see that all test cases are covered so that no functionality should miss while doing Software testing

Table 4.1 Requirement Traceability Matrix

Requirement ID	Requirement Name	Test Case ID	Sequence Diagram ID
R1	Generate Scene	TC-1	SD-1
R2	Save Scene	TC-2	SD-1
R3	Admin Login	TC-3	-
R4	Add Image	TC-4	SD-2
R5	Delete Image	TC-5	SD-2
R6	Train Model	TC-6	SD-3

4.2 Software Architecture Design

An architecture is the set of significant decisions about

- The organization of a software system,
- The selection of the structural elements and their interfaces
- Their behavior as specified in the collaborations among those elements
- The composition of these structural and behavioral elements into progressively larger subsystems
- The architectural style that guides this organization---these elements and their interfaces, their collaborations, and their composition.

4.2.1 Chosen System Architecture

I am going to use 3-Tier Architecture for the development of TTSG. The top layer is the presentation layer. Second layer is Application Layer which has business logic. The bottom layer is Data layer which contains the database of the system.

- **Presentation Layer**

  This layer allows the users or Admin to directly communicate as this layer provides the interfaces. User can interact through this layer and the outputs are also shown to the users in this layer.

- **Application Layer**

  This layer has the main logic of the system. It communicates directly with the data in the data layer. The actual processing is done in this layer. It has the following modules.

  - Machine Learning Module
  - Parser
  - Generator

- **Data Layer**

  This layer has data in 3 parts

  1. Dataset
  2. Trained Model

  Application layer communicate directly to this layer for training of model, classification, retrieving or storing images.

4.2.2 Architecture Diagram

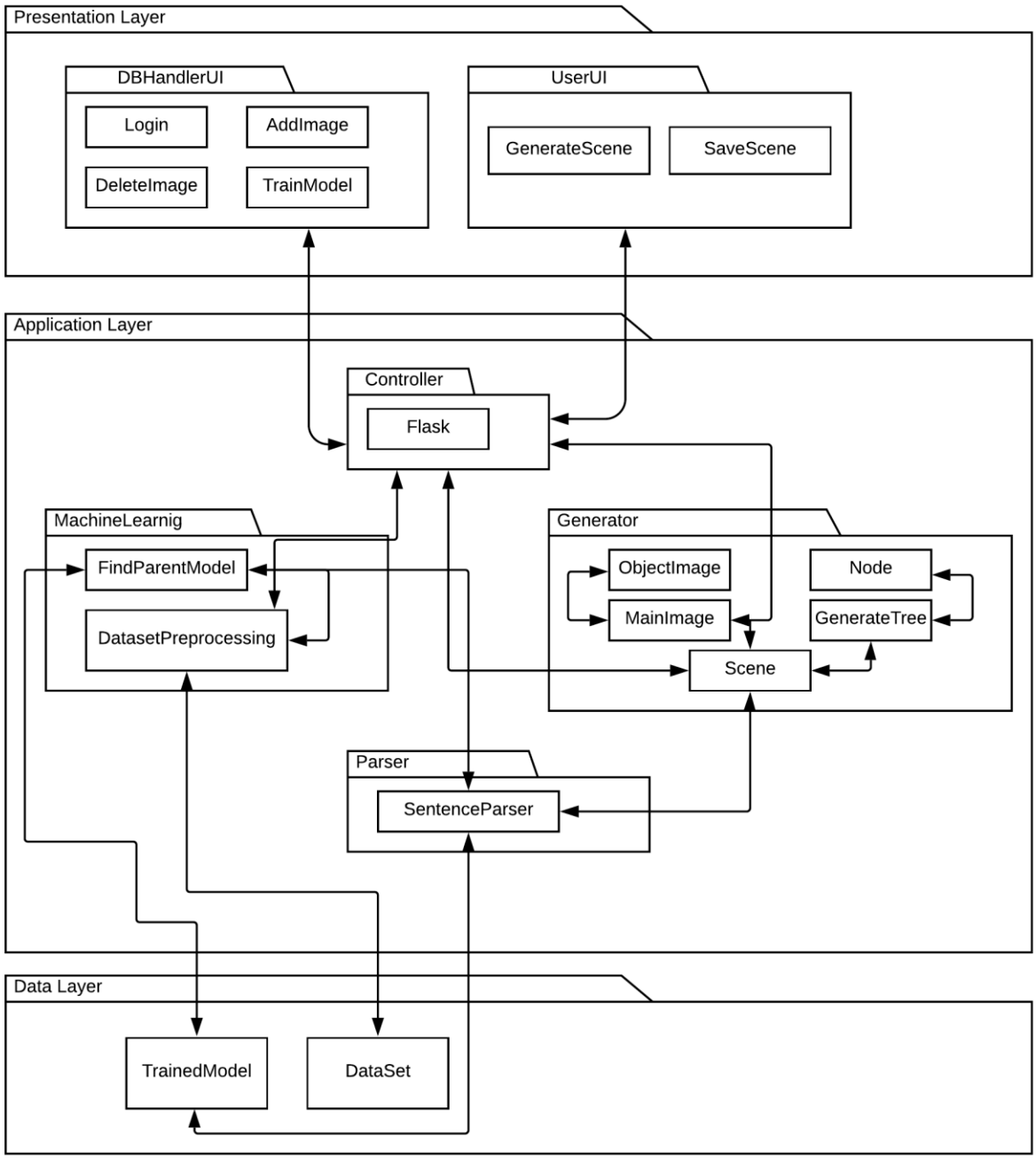

Figure 4.1 Architecture Diagram

4.2.3 System Interface Description

System interface describes the flow of resources. It is the logical characteristics of each interface between the software product and the hardware components of the system. It clearly shows how different entities of tool are interacting with each other.

4.3 Detailed Description of Components

Following are the components and their description and functionality.

4.3.1 Admin

This component provides interfaces to the database handler to add or delete images to or from the database by communicating to this module. And the Admin can also train the model through this. This component sends images to the controller and then controller is responsible for sending it to database through machine learning component. The images can be deleted and the model can be trained for classification through the same hierarchy.

4.3.2 User

This component provides interfaces to user to generate and modify the generated scene. The user can send input text through this component. The text is sent to controller.

4.3.3 Controller

This component takes request from the presentation layer and sends the requests to the corresponding components. It also takes output from generator and send it back to presentation layer.

4.3.4 Parser

This component takes the text input came from user through controller and generates a parse tree and send it to generator

4.3.5 Generator

This component takes the parse tree came from parser and fetches images from database and plot them in an image file according to the parse tree and send it to controller.

4.3.6 Machine Learning Module

This component is used to train the model and also for classification of the sentences that has to be stored in the database.

4.3.7 Database

This component has dataset of the sentences

4.3.8 Train Model

This component has the trained model of image classifier.

4.3.9 Component Diagram

The UML component diagram describes the components used in a system and collaboration between them it does not describe the functionality.

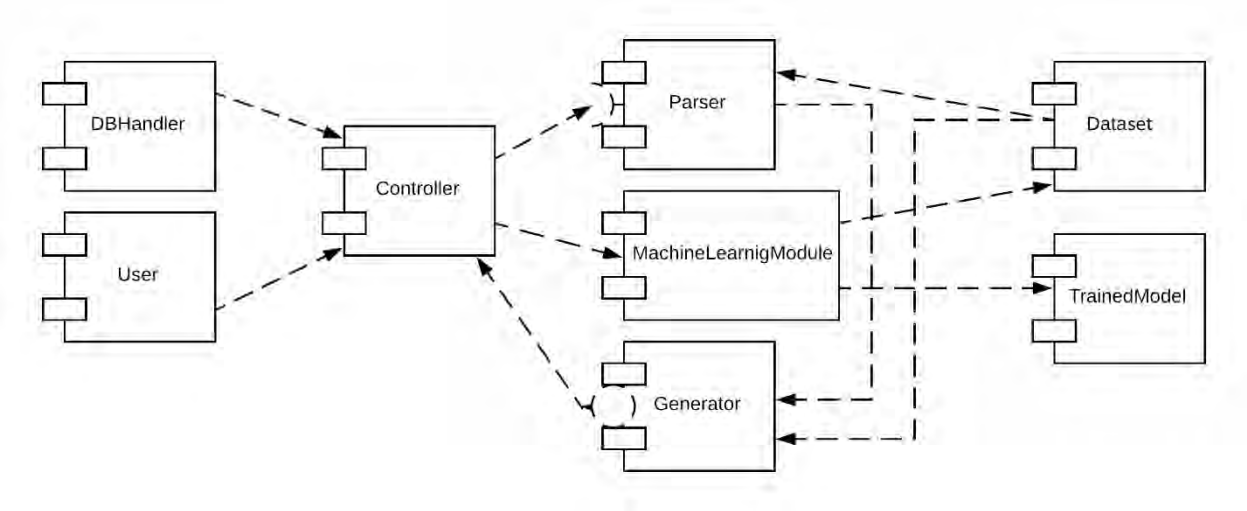

Figure 4.2 Component Diagram

4.4 Sequence Diagrams

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time. They're also called event diagrams. A sequence diagram is a good way to visualize and validate various runtime scenarios [16].

4.4.1 SD-1: Generate and Save Image

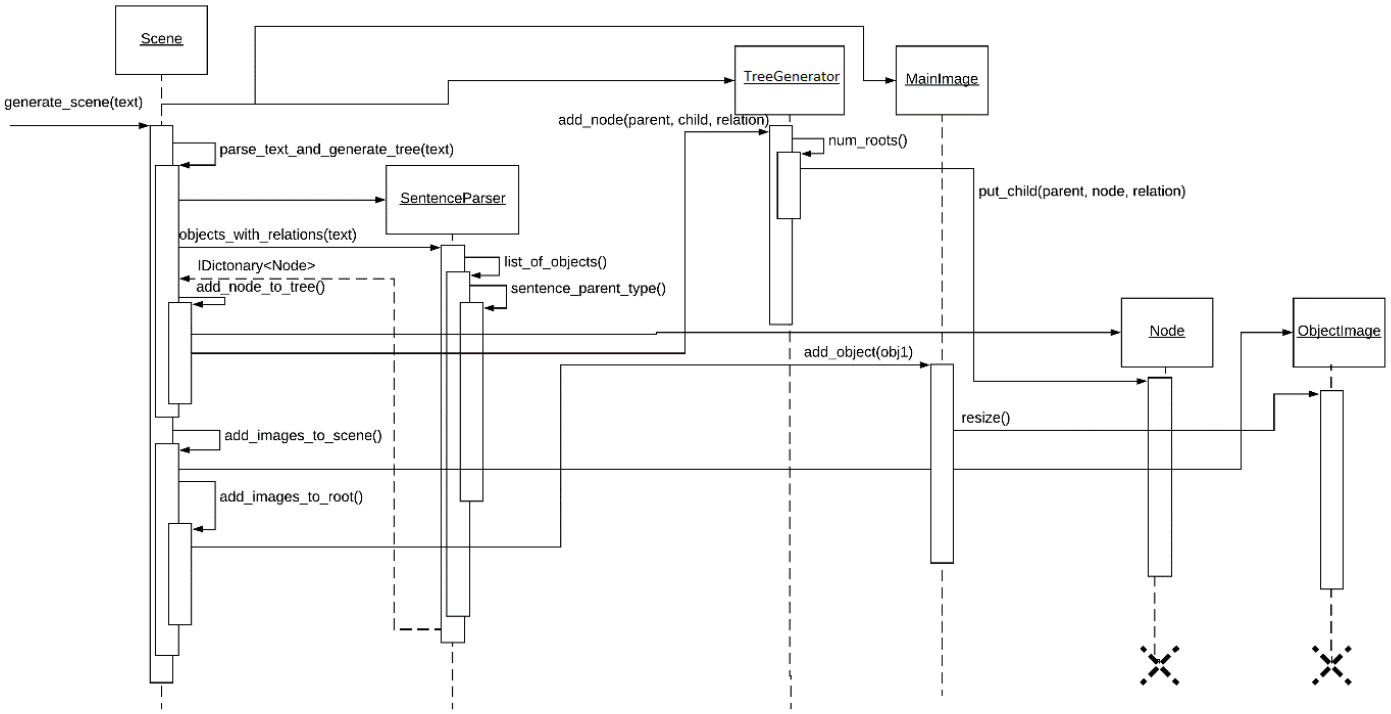

Figure 4.3 SD-1: Generate, Modify and Save Scene

4.4.2 SD-3: Add and Delete Image

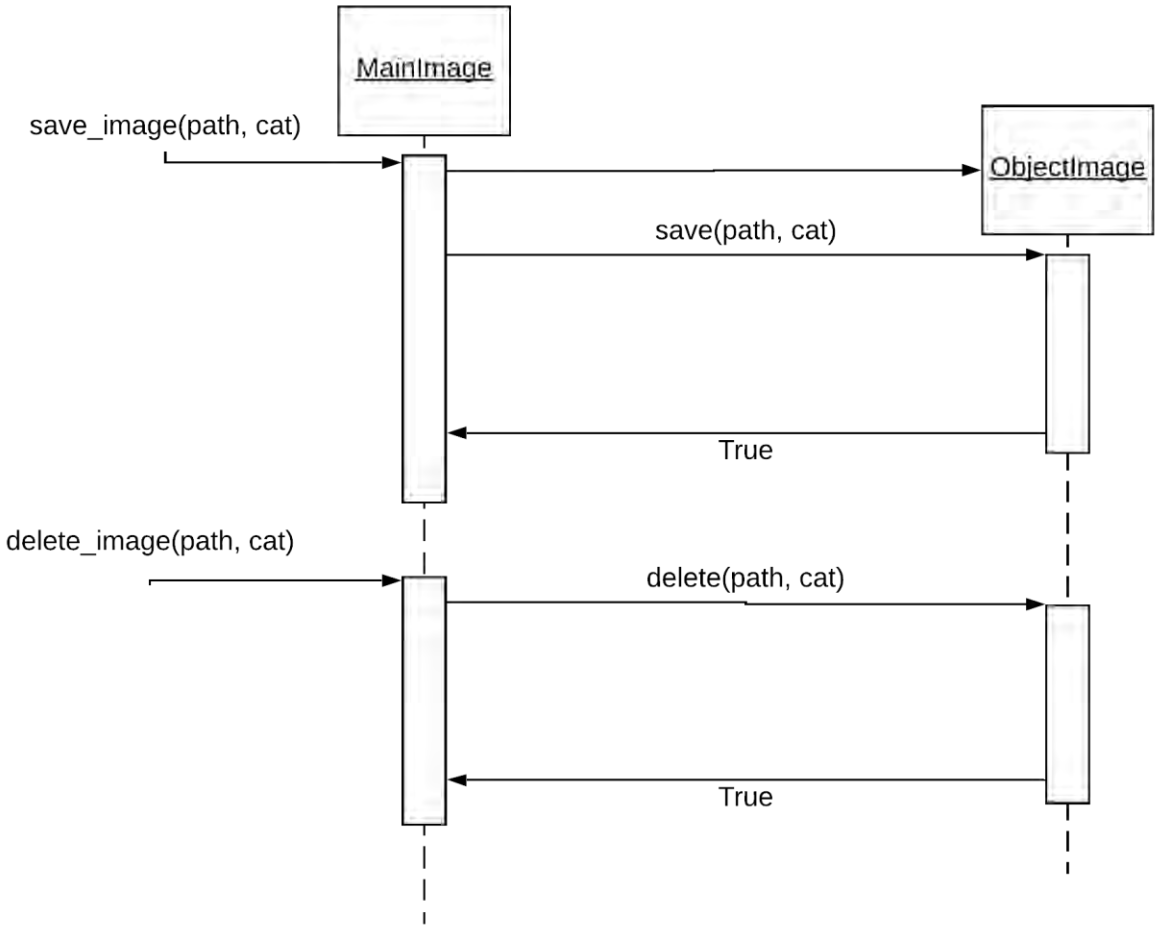

Figure 4.4: SD-2: Add and Delete Image

4.4.3 SD-4: Train Model

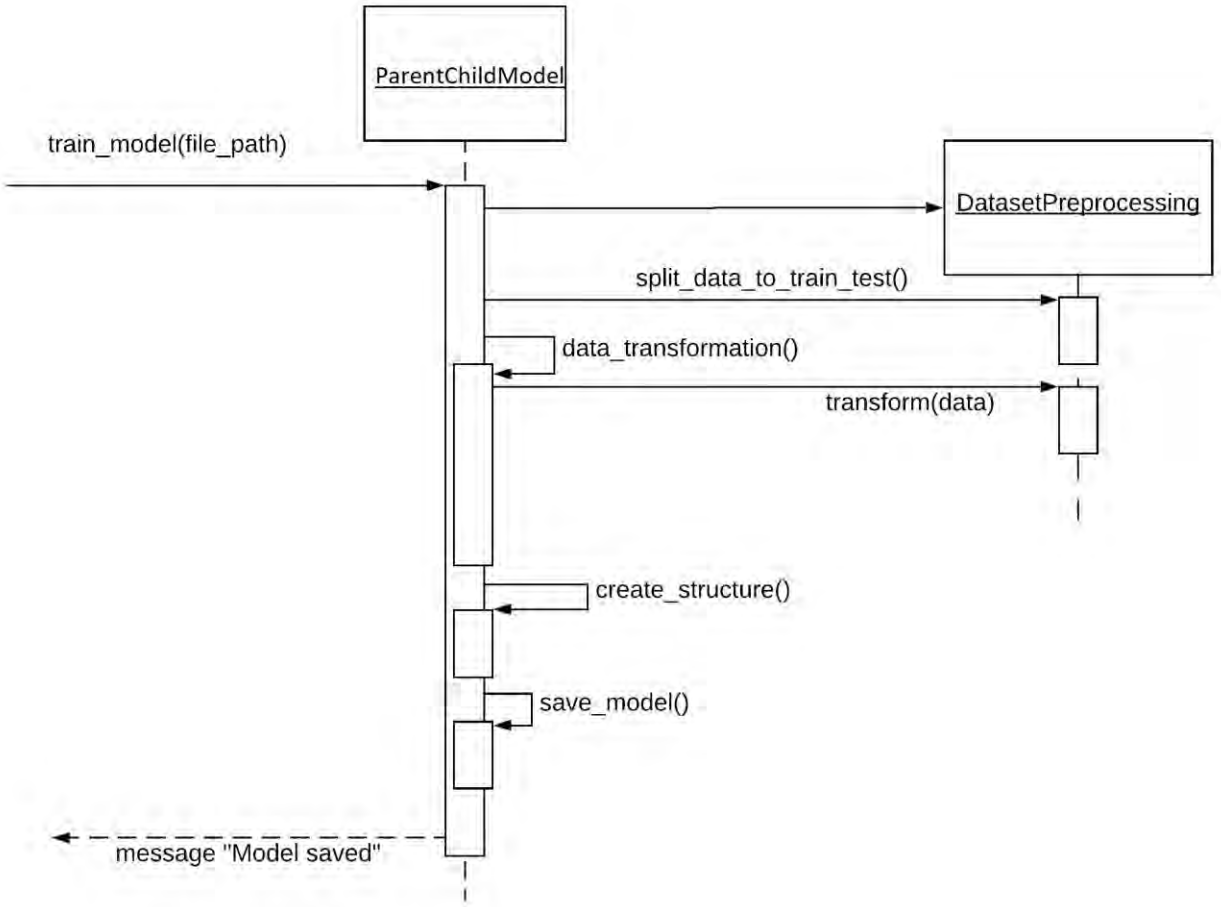

Figure 4.5: SD-3: Train Model

4.5 Class Diagram

Class diagrams depict the software classes and their relationships. This diagram defines individual classes along with their attributes, types of the attributes, and operations, associations between classes and navigability (direction of association) that define attribute visibility, and also define non-attribute visibility.

Following are the classes used

- **Scene:** For scene generation
- **Sentence Parser**: For parsing the sentence and getting objects name and their relations
- **Tree Generator:** For creating a parsed tree of the objects according to their relations
- **Main Image:** For holding the complete scene and to save and generate it
- **Object Image:** For holding info related to each object image
- **Node:** For holding each individual object properties
- **Parent Child Model:** For dataset training
-
- **Dataset Preprocessing:** For data set pre-processing

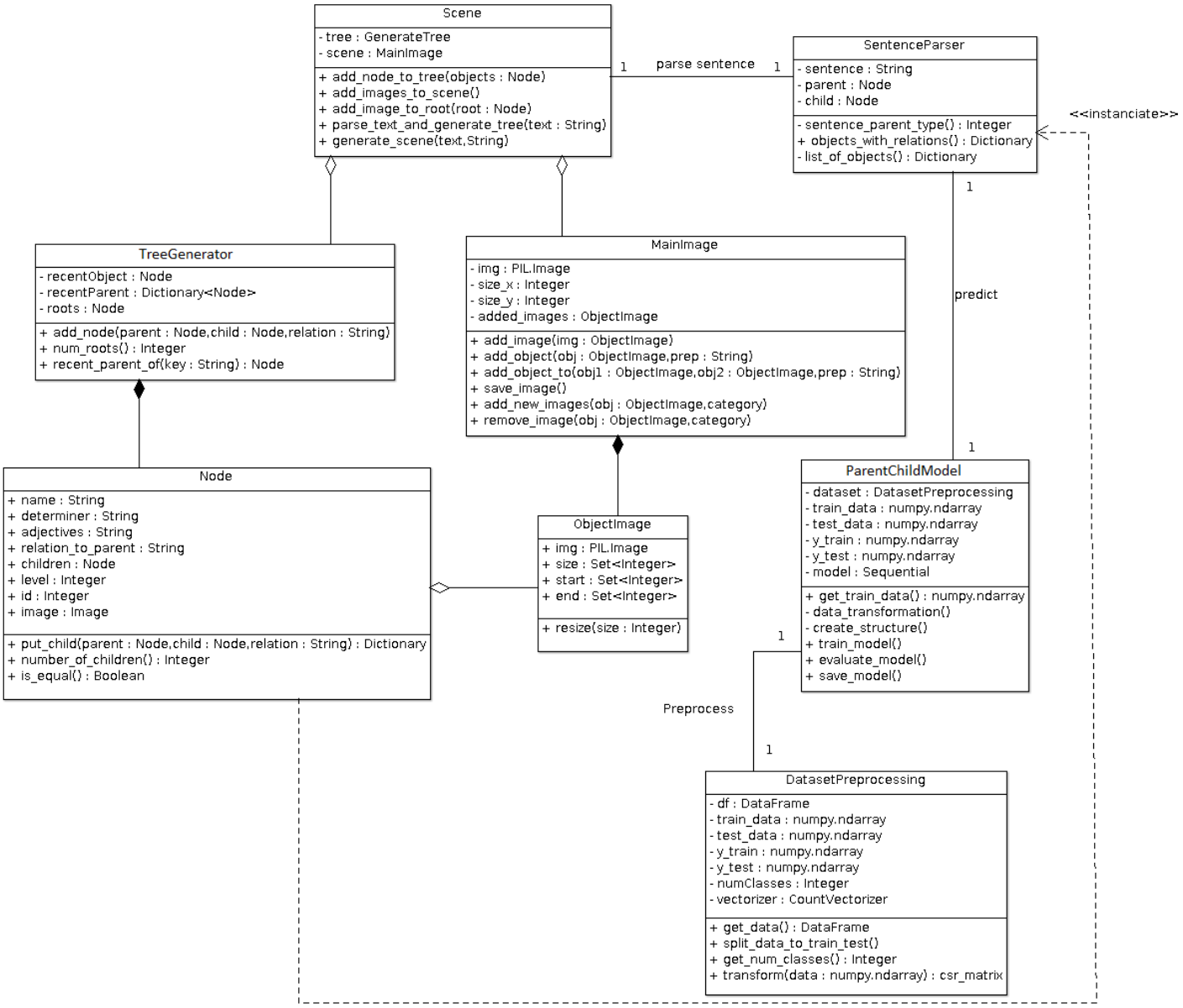

Figure 4.6 Class Diagram

4.6 User Interfaces

4.6.1 Generate Scene

On this interface user enters a text input and he /she can then click on the generate scene button to generate a scene as below interface

Figure 4.7 Generate Scene UI

4.6.2 Save Scene

On this interface user can download the generated scene

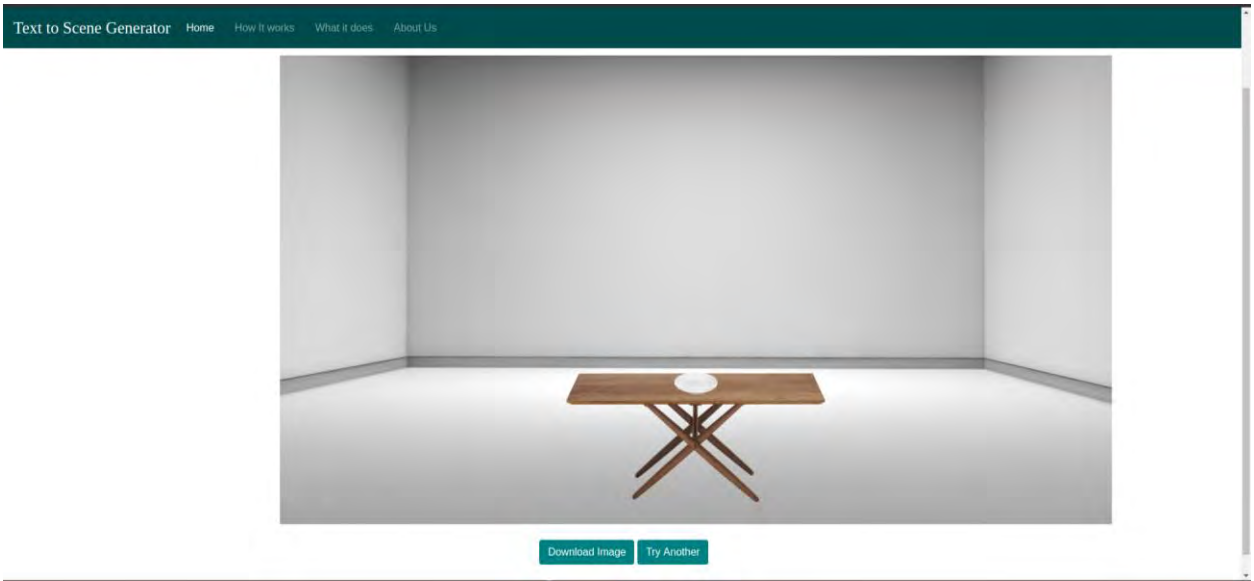

Figure 4.8 Save Scene UI

Chapter 5: Implementation Details

5 IMPLEMENTATION DETAILS

5.1 Project Boundary

5.1.1 User input:

The user can generate scene but following restrictions are applied for input:

- The input can contain more than one sentence but each sentence must end with a dot.
- The input must contain one or two objects. List of objects which a user can give as an input are given in appendix.
- The input can have the object properties and relations which are also listed in appendix.

For example the sentences could be “A plate on a table.”, “a chair near the table.”

5.1.2 Objects:

Mainly the objects in a scene are nouns. For this project I used nouns and objects for example: chair and table.

The project was intended to make a generic solution but by using a small domain to start with. The domain of the system is only for a room and objects which are related to room. The objects to which the domain in restricted are mentioned in the appendix.

5.1.3 Objects Relations:

Objects relations are the prepositions which describes the positions of objects like chair is near to table or a cake is on the plate. In these sentences ‘near’ and ‘or’ are two prepositions which are describing the positions of the objects with relation to other objects.

In the example sentences the chair’s position is defined by the preposition near, as it is important to know the positions of the objects in a scene depiction, that’s why I chose the prepositions to place the objects correctly.

5.1.4 Object Properties:

Properties are the adjectives used to describe the object like a big table or a brown chair

The project is also limited to the features of these objects. Following object properties are used:

- The size of the object and words are only (small and big)
- The color of the object. All CSS colors

5.2 Dataset

The dataset contains the sentences and a label

5.2.1 Sentences

The sentences contains following information

- Each of the sentence contains at most 2 objects.
- There are one relation in each sentence
- Out of the objects specified are also used to make it generic for example "A fridge is near a bed"

5.2.2 Labels

The label are 0 and 1

- The 0 stands for the object came first in the sentence is a parent of the second. For example in sentence "A table has a plate on it" the first object "table" is parent object.
- The 1 stands for the object came second in the sentence is a parent of the first. For example in sentence "A cake is on the plate" the second object "plate" is the parent.

5.3 Techniques Used:

- Classification: With neural network
- Parsing: with some help of nltk
- Tree Data structure for objects and relations

5.4 Approaches used (Project Flow):

5.4.1 Dataset Training

For training I used neural networks as classifier. This neural network classifies the label 0 or 1 which was explained above. The model trained from this dataset is able to indicate the parent object in the sentence.

Neural Network Properties:

- For neural network to be build and trained I used tensorflow keras.
- The neural network is of keras Sequential type.

- The neural network contains 3 layers
    - Input layer: Which has 10 neurons
    - Hidden layer: which contains 20 neurons
    - Output layer: contains 2 neurons as out is of two labels

The number of neurons and the number of layers are decided by the accuracy of the model. By increasing or decreasing the number of neurons and layers the accuracy was disturbed. So after trying with multiple hyper parameters I ended up with the structure explained above

5.4.2 Parsing

First I tokenize the words and find their parts of speech type by using python nltk library

- NN (Noun) parts of speech found are objects.
- JJ (adjectives) are properties of objects
- IN (prepositions) are relation between objects
- DT (determiner) for objects (The, A)

This module then predict who is parent and who is child from the model trained above. If there are more objects the first 2 are taken. If the relations are more than one then the first one is taken

This information then passed to Tree generation

5.4.3 Tree Generation

This module takes the information from parsing and create a node object for each (parent and child

Following are the cases for parent and child which parser sends:

- If the parent is none that means the child object sent is root of tree. For example user says. ‘There was a table”
- If the word **‘The’** is used with the parent in determiner it means it is telling us about the recent parent object added with this name. For example the user says ‘There is a plate on table. A cake on the plate’. In this case the cake will be added to the plate recently added.
- If the word **‘it’** is used which means the object recently added. For example user says: ‘There was a table. A plate was near it’

The type of the tree is a general tree. For which each node can have multiple objects. The traversing used is in order.

For example user gives following input, “A plate on a table. A cake is on the plate. Chair is near the table.” Then the tree generated would be like this.

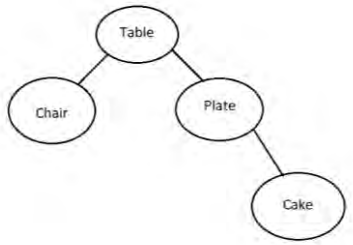

Figure 5.1Example Tree

5.4.4 Image Generation

This module takes the whole tree and traverse it in in order and for each object came it retrieves the image from project storage (in files) and paste the image on a main room image.

- If the object is root then it will be added on floor
- If the object has relation then it will set according to its parent

5.5 Tools used:

- PyCharm (For backend)
- Sublime text (For frontend)

5.6 Libraries Used:

- NLTK natural language processing library: used for POS tagging [8]
- PIL python image library: For image opening, resizing, pasting, saving

Chapter 6: Software Test Document

6 SOFTWARE TEST DOCUMENT

6.1 Introduction

Software testing is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect free in order to produce the quality product.

Software testing is a verification and validation process which guides the tester through a sequence of steps to validate whether a software application is free of bugs and working as required by the end user. Software testing is a very important process that should be done during the development process because it is very useful to assess the quality of the product.

This document provides test plan and acceptance test for text to scene generator.

6.1.1 Test Approach

This section describes the strategy used for the testing of the project. I am going to use black box testing technique for this project. For that purpose I will perform boundary value analysis and then define the acceptance criteria in form of acceptance tests.

6.2 Test Plan

A test plan is a detailed document that outlines the test strategy, testing objectives, resources required for testing, test schedule and test deliverables. The test plan serves as a blueprint to conduct software testing activities.

It also describes that which features to be tested and which features are not to be tested.

6.2.1 Features to be tested

Following are the features that I am going to test

- Login
- Train Model
- Add Image
- Delete Image
- Generate Scene
- Save Scene

6.2.2 Features not to be tested

Processing Memory and speed is not tested here.

6.2.3 Testing Tools and Environment

Following are the tools and environment that are going to be used for the testing

- PC/Laptop
- Web Browser e-g Google Chrome, Mozilla Firefox
- Windows Operating System

6.3 Test Case

A Test Case is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly. The process of developing test cases can also help find problems in the requirements or design of an application [17].

6.4 Acceptance Test

Acceptance testing, a testing technique performed to determine whether or not the software system has met the requirement specifications. The main purpose of this test is to evaluate the system's compliance with the business requirements and verify if it is has met the required criteria for delivery to end users [21].

6.4.1 TC-1: Generate Scene

Table 6.1 TC-1: Generate Scene

ID	TC-1
Requirement ID	R-1
Requirement Name	Generate Scene
Tester	User
Setup	1. The Database must have one image of each table, cake and plate.
Input	1. Text Input "There is a table in a room. A plate is on the table".
Expected Result	A Scene image should be return with table and plate on the table There should be no distance between the objects.
Actual Result	As expected
Verdict (Pass/Fail)	pass

Figure 6.1Generated Scene

6.4.2 TC-2: Save Scene

Table 6.2 TC-2: Save Scene

ID	TC-2
Requirement ID	R-2
Requirement Name	Save Scene
Tester	User
Setup	1. System must have images of each, the table and plate 2. User inputs “a table with a plate on it” 3. System generates a scene with a table and plate on the table
Input	1. User clicks to save image 2. System asks for the saving path 3. User browse or input path of the disk
Expected Result	System saves the image in the disk at right path by sending a download link
Actual Result	As expected
Verdict (Pass/Fail)	pass

6.4.3 TC-3: Admin Login

Table 6.3 TC-3: Admin Login

ID	TC-3
Requirement ID	R-3
Requirement Name	Admin Login
Tester	Admin
Setup	1. Register Admin with username=”admin” and password=”ttsgadmin”.
Input	1. Admin input username “admin” 2. Admin inputs password “ttsgadmin”
Expected Result	Admin should be successfully logged in to the system
Actual Result	As expected
Verdict (Pass/Fail)	pass

6.4.4 TC-4: Add Image

Table 6.4 TC-4: Add Image

ID	TC-4
Requirement ID	R-4
Requirement Name	Add Image
Tester	Admin
Setup	1. System has trained model
Input	1. Image file of table 2. Latitude= 29.5 3. Longitude = 22.7
Expected Result	System classifies this image as “table” and stores it with label table1
Actual Result	As expected
Verdict (Pass/Fail)	pass

6.4.5 TC-5: Delete Image

Table 6.5 TC-5 Delete Image

ID	TC-5
Requirement ID	R-5
Requirement Name	Delete Image
Tester	Admin
Setup	1. Add an image of a table 2. Selects to Delete image 3. System shows image category 4. Selects image and selects to delete
Input	1. Image ID=111
Expected Result	Deletes image with ID 111
Actual Result	As expected
Verdict (Pass/Fail)	Pass

6.4.6 TC-6: Train Model

Table 6.6 TC-6: Train Model

ID	TC-6
Requirement ID	R-6
Requirement Name	Train Model
Tester	Admin
Setup	1. Admin adds new image of a spoon 2. Admin adds new sentences in the dataset 3. Admin selects to train model
Input	1. Epoch value=1000 2. Learning rate=0.02 3. Labeled dataset
Expected Result	Training starts
Actual Result	As expected
Verdict (Pass/Fail)	pass

Conclusion:

The text to scene generator is able to generate room objects scenes. The software parses the sentences given in natural language and then extracts the objects and their relations from the text input. It also extracts the adjectives of objects. The system now allows the admin to login and add more images for all of the objects specified in the domain. The admin can now train the classification model by providing more sentences.

Future Enhancements:

- We can make the domain for scenes broader
- We can allow user to modify the scene with some drag and drop options
- We can give user suggestions for the scene that are already generated from the software
- We can provide a word cloud of objects to generate scenes instead of giving the long text

References

[1] IEEE Software Engineering Standards Committee, "IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications", June 25, 1998.

[2] Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative development, Third Edition by Craig Larman.

[3] Pattern Recognition and Machine Learning Book by Bishop's

[4] What is machine learning?https://www.expertsystem.com/machine-learning-definition/

[5] Natural Language Processing https://www.sas.com/en_us/insights/.../what-is-natural-language-processing-nlp.html

[6] Sentence Parsing https://www.quora.com/What-is-parsing-in-NLP

[7] Semantic Parsing https://en.wikipedia.org/wiki/Semantic_parsing

[8] NLKT https://www.nltk.org/

[9] Classification https://www.mediaupdate.co.za/marketing/143399/artificialintelligence-what-is-classification

[10] Learning Spatial Knowledge for Text to 3D Scene Generation, Angel X. Chang, Manolis Savva, and Christopher D. Manning, EMNLP 2014.

[11] Zhang, H., Xu, T., Li, H., Zhang, S., Huang, X., Wang, X., & Metaxas, D. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. IEEE Int. Conf. Comput. Vision. 2017.

[12] Chat Painter: Improving Text to Image Generation using Dialogue, Shikhar Sharma1, Dendi Suhubdy, Vincent Michalski, Samira Ebrahimi Kahou1 Yoshua Bengio

Appendix A: List of objects

The nouns in the sentences refers to objects in a scene.

Following are the objects which are in domain of the project

Table	Chair	Plate	Book	Pen
Sofa	Jug	Bottle	Cake	Phone

Appendix B: List of Prepositions

Prepositions in sentences refer the relation between two objects. Following are the list of prepositions which are in this project domain

Near	Far	On	Below
Under	Left	Right	In

Appendix C: List of Adjectives

Adjectives are the characteristics of object itself. Following adjectives are under the domain of this system.

The size of the object which can only be described by following two words.

Small	Big

The color of the object. The color can be any of CSS colors.