



جامعة طيبة

كلية علوم وهندسة الحاسب الآلي

(قسم الطلاب)

KINGDOM OF SAUDI ARABIA

Ministry of Education

Taibah University

College of Computer Science and

Engineering

(Male Section)

**AI-Powered Requirements Analysis Modeling**

**Graduation Project 2**

**by**

Ali Talal Al-Ahmadi 4100379

Mohammed Hadi Al-Harbi 4101704

Rami Ramadan Al-Mohammadi 4101758

**A project submitted in partial fulfilment of the requirements for the degree of Bachelor of Science Computer Science**

**Supervised by**

Dr. Mohammad M. Alsuraihi

1st Semester - Academic Year 1445 (2023/2024)

# Abstract

The project may face issues with excessive resource consumption, such as time or costs, and frequent changes in requirements can lead to project delays and increased expenses. Additionally, sometimes the techniques or tools used in the analysis and design process are ineffective or outdated. Software maintenance constitutes a pivotal stage within the software development lifecycle, encompassing a substantial portion, varying from 40% to 80% of the total expenses associated with software development. It's worth highlighting that a notable 60% of the overall maintenance expenditure is dedicated to the improvement of existing software functionalities. Consequently, it becomes imperative to meticulously prepare appropriate software documentation at each developmental phase in order to alleviate the financial burdens of maintenance. Using artificial intelligence (AI) to analyze requirements and generate drawings or models can be an effective solution for saving time and costs in the design and analysis processes. The crux of reducing maintenance costs lies in enhancing one's grasp of the software system, as understanding a software system accounts for roughly 50% of the time spent in the maintenance phase.

**Keywords** NLP; ML; AI-Powered; Requirements; Analysis; UML

# Acknowledgement

Alhamdulillah. First and foremost, all praise is due to Allah SWT the worthy of all the praises and compliments for giving us the strength and ability to complete this project. Next, we express gratitude to our parents and families. We would like to extend our thanks to our project supervisor, Dr. Mohammad M. Alsuraihi, who guided us in executing this project, providing invaluable advice, assisting us in challenging times, and significantly contributing to the project's completion.

We also want to express our appreciation to all the professors at Taibah University, College of Engineering and Computer Science, who have been part of our academic journey at the university.

Finally, and not least, we would like to thank those who worked on this project: Ali Talal Al-Ahmadi, Mohammed Hadi Al-Harbi, and Rami Ramadan Al-Mahmoudi, for their contributions to this project. I appreciate their hard work and dedication.

**Contents**

[Abstract ii](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518704)

[Acknowledgement iii](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518705)

[List of Figures vi](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518706)

[List of Tables vii](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518707)

[List of Abbreviations viii](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518708)

[1 Chapter 1: Introduction 1](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518709)

[1.1 Introduction 1](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518710)

[1.2 Project Aim and Objectives 1](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518711)

[1.3 Project Methodology 1](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518712)

[1.4 Project Timeline 2](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518713)

[1.5 Document Organization 3](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518714)

[1.6 Summary 3](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518715)

[2 Chapter 2: Literature Review 4](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518716)

[2.1 Introduction 4](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518717)

[2.2 Research Methodology 4](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518718)

[2.3 AI 6](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518719)

[2.4 Machine Learning 8](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518720)

[2.4.1 Neural Networks (NNS) 9](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518721)

[2.4.2 Supervised Learning 9](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518722)

[2.5 NLP 10](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518723)

[2.5.1 Named Entity Recognition 11](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518724)

[2.5.2 Parsing 13](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518725)

[2.5.3 Pronoun Resolution 15](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518726)

[2.5.4 Semantic Analysis of Requirement Using NLP and ML 16](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518727)

[2.6 System Development for AI 17](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518728)

[2.6.1 NLP application Development 17](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518729)

[2.6.1 ML application Development 19](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518730)

[2.7 Similar Tools 21](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518731)

[2.7.1 The lucid platform 21](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518732)

[2.7.2 Visual paradigm 22](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518733)

[2.7.3 StarUML 23](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518734)

[2.7.4 Enterprise Architect 24](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518735)

[2.7.5 MagicDraw 25](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518736)

[2.7.6 ChatUML 26](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518737)

[2.8 Summary 26](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518738)

[3 Chapter 3: System Analysis 27](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518739)

[3.1 Introduction 27](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518740)

[3.2 Methodology 27](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518741)

[3.2.1 SDLC 27](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518742)

[3.2.2 Analysis 28](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518743)

[3.3 Analysis of Existing Systems 30](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518744)

[3.4 Requirements Elicitation 31](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518745)

[3.4.1 System Requirements 32](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518746)

[3.4.2 Functional Requirements 32](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518747)

[3.4.3 Non-Functional Requirements 33](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518748)

[3.5 Requirements Specification 34](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518749)

[3.6 Requirements Modeling 38](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518750)

[3.6.1 Use case diagrams 38](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518751)

[3.6.2 Class Diagram 40](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518752)

[3.7 Summary 41](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518753)

[4 Chapter 4: System Design 42](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518754)

[4.1 Introduction 42](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518755)

[4.2 Design Methodology 42](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518756)

[4.3 Architectural Design 43](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518757)

[4.4 Component Design 44](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518758)

[4.5 Data Modeling Design 45](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518759)

[4.6 User Interface Design 45](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518760)

[4.7 Summary 47](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518761)

[5 Chapter 5: Conclusion and Future Work 48](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518762)

[5.1 Conclusion 48](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518763)

[5.2 Goals Achieved 48](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518764)

[5.3 Lessons Learnt 49](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518765)

[5.4 Limitations and Future Work 49](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518766)

[5.4.1 Limitations 49](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518767)

[5.4.1 Future work 50](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518768)

[6 References 51](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc157518769)

# List of Figures

[Figure 1: Project Methodology 3](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058507)

[Figure 2: Methodology Schema 9](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058508)

[Figure 3: AI branch [7] 11](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058509)

[Figure 4: Neural networks architecture [12] 12](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058510)

[Figure 5: Supervised learning process [13] 13](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058511)

[Figure 6: Word2Vec architecture: (a) CBOW; and (b) skip-gram [15] 14](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058512)

[Figure 7: RNN model architecture [15] 15](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058513)

[Figure 8: Examples of experimental results: (a) original text; and (b) NER results [15] 16](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058514)

[Figure 9: Constituency and dependency structures for the sentence ‘the man hit the ball’ [16] 17](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058515)

[Figure 10: An example of a non-projective parse tree [16] 17](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058516)

[Figure 11: difference between "original sentence" and "sentence with resolved Coreference" [18] 18](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058517)

[Figure 12: identify potential spans [18] 19](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058518)

[Figure 13: group spans [18] 19](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058519)

[Figure 14: replace pronouns with real-world entities [18] 19](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058520)

[Figure 15: waterfall for project ((\*): It will be worked on in the second term). 31](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058521)

[Figure 16: Description of stages of analysis 32](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058522)

[Figure 17:Devolper use-case diagram 41](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058523)

[Figure 18: user use-case diagram 42](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058524)

[Figure 19: class diagram 43](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058525)

[Figure 20:Design methodolgy ((\*): It will be worked on in the second term) 45](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058526)

[Figure 21: Architectural Design 47](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058527)

[Figure 22: Home Page 48](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058528)

[Figure 23: Input Page 49](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058529)

[Figure 24: Diagram Page 49](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058530)

# List of Tables

[Table 1: Project plan v1 4](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058531)

[Table 2: Project plan v2 5](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058532)

[Table 3: Related Work Comparison 33](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058533)

[Table 4:use-case (Upload Dataset) 37](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058534)

[Table 5: use-case (Load Dataset) 37](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058535)

[Table 6: use-case (train Model) 37](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058536)

[Table 7:use-case (Analysis) 38](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058537)

[Table 8:use-case (test Model) 38](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058538)

[Table 9: use-case (Enter requirements Document) 39](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058539)

[Table 10: use-case (analysis Document) 39](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058540)

[Table 11: use-case (Create analysis model) 39](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058541)

[Table 12: use-case (save diagram) 40](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058542)

[Table 13: use-case (Draw use-case model) 40](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058543)

[Table 14: use-case (Draw class model) 40](file:///C:\Users\lenovo2020\mmgp\report-project2.docx#_Toc153058544)

# List of Abbreviations

UML Unified Modeling Language

ML Machine Learning

NLP Natural Language Processing

AI Artificial Intelligence

UI User Interface

RNN Recurrent Neural Network

SDLC Software Development Life Cycle

NNS Neural Network Systems

NER Named Entity Recognition

# Chapter 1: Introduction

## Introduction

This project is a continuation of evaluated project 1. This continuation aims to develop and enhance the proposed solution based on the previous discussion and here some Notes:

* Focus on writing and know everything written in the project.
* Focus on clear, understandable, and simple project abstract writing.
* Research in scientific papers on the development of NLP applications.
* Use and benefit from the scikit-learn Library.
* Identifying the algorithms used in the project.
* There must be a system scenario that explains the system process.

## Project Aim and Objectives

The project’s goal is to deepen our understanding of the related work in terms of design and code, improve the system analysis process, complete the design phase with a focus on the data model and components, and program and test the system. All of this aims to speed up and facilitate the work of system modeling analysts and save their time.

To achieve this goal, we must achieve the following objectives:

1. Deepen the understanding of the related work in terms of design and code.
2. Improve the system analysis process.
3. Complete the design phase with a focus on the data model and components.
4. Program and test the system.
5. Conclude with the lessons learned and the knowledge and experience gained from working on this project.

## Project Timeline

This section presents the timeline plan versions of our project as the following:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **May** | | **April** | | | | **March** | | | | **February** | | | | **January** | **Months**  **(2024)** |
| **2** | **1** | **4** | **3** | **2** | **1** | **4** | **3** | **2** | **1** | **4** | **3** | **2** | **1** | **4** | **Weeks** |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Define project objectives and review discussion notes | **Tasks** |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Review existing systems |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Improve the system analysis process |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Complete design the system |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Program system and test |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Documentation |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Presentation |

Table 1: Project plan v1

1. **Define project objectives and review discussion notes (1 week):**

We will define the main objectives of the project and review all the notes from previous discussions.

1. **Review** **existing systems (2 weeks):**

We will review the existing systems to know what languages were used in developing the previous systems and how they work.

1. **Improve the system analysis process (2 weeks):**

We will work on improving the system analysis process to ensure we get the most accurate results.

1. **Complete design the system (2 weeks):**

We will improve the system design and draw data models and components.

1. **Program system and test (6 weeks):**

We will program the system and test it to ensure it works as expected.

1. **Documentation:**

This task spans throughout the project's duration as it involves continuous documentation of the work.

1. **Project Presentation:**

The project should be presented during this time to showcase the results.

## Document Organization

* **Chapter 1.** In this chapter we discussed the plan and objectives and identified the problem that the project solves.
* **Chapter 2.** In this chapter, we reviewed existing systems.
* **Chapter 3.** Our focus in this chapter was on identifying functional and non-functional requirements using UML diagrams, as well as identifying user requirements and search methodologies.
* **Chapter 4.** In this chapter, we designed the system, identified the components of the system, modeled the data, and designed the user interfaces.
* **Chapter 5.** In this chapter, we programmed the system and conducted a series of tests to ensure that the system is working well.
* **Chapter 6.** A presentation of the project's conclusion was given, along with a proposal for new work to improve the current work, and a statement of whether the original objectives of the project had been met.

## Summary

In this chapter, we documented the feedback from the project's first graduation thesis and outlined the project's goals. We also defined a plan to address the feedback and execute the project modifications.

In the second chapter, we will delve into similar previous works, gathering and discussing all relevant information.

# Chapter 2: Literature Review

## Introduction

In this unit, our focus will be on exploring various programs that share similar with our project. Through this exploration, we aim to gain idea into the development, implementation, and testing processes used in these similar projects.

## Lucidchart

Lucidchart assists teams with several activities including building schemes such as UML, resource allocation for the project and the design of low-fidelity prototypes for user testing. [1]

**Program Design:** The program uses a microservices architecture model. This makes maintenance, expansion, and development easier because functions are divided into small, independent services. [2]

**Implementation:**

**Front-end:** The web application uses HTML5, CSS3, and JavaScript. [2]

**Back-end:** The server is written as a PHP application, with a component written in C. [3] The cloud infrastructure uses Google Cloud Platform, and Google Cloud Storage is used for storage. [2]

### StarUML

### ChatUML

ChatUML is an AI-powered diagram generator that allows you to create and edit PlantUML diagrams using natural language conversations. [4]

**Implementation:**

**Front-end:** The web application uses HTML5, CSS3, and JavaScript.

## Summary

In this chapter, we studied previous systems in depth. We learned about the model used in each system, how it was implemented, and how it was tested. This study helped us understand the systems better and identify rules that can help us develop our system.

Based on what we learned in this chapter, we are now in a better position to move to the next chapter. There, we will analyze our system and write the steps to perform operations in the system.

# Chapter 3: System Analysis

## Introduction

In this chapter, we will focus on defining the system requirements using simple diagrams. We will use UML tools to explain how the system operates and its requirements. We will specify both functional and non-functional requirements and discuss the development methodology. We will also analyze an existing system.

## Methodology

### SDLC

In our project, we used the waterfall methodology for project execution. In the first step, the project plan. In the second step Literature Review. in the third step, analyzed the system. in the fourth step, system design.

Stability of Requirements: The Waterfall model requires defining project requirements early. It fits well when requirements are stable and undergo minimal changes.

Path Determination: The Waterfall model follows a linear and organized approach, allowing the team to clearly outline the workflow and task sequence. This structure is beneficial for projects requiring precision and strict timelines.



Figure 15: waterfall for project ((\*): It will be worked on in the second term).

### Analysis

In the analysis phase we have identified 4 basic activities to fully analyze the components of the system. Before starting with system design in general we must collect the requirements and analyze them correctly. During the analysis phase of the system, we used this existing strategy in Figure 16.



Figure 16: Description of stages of analysis

In the process of systems analysis existing, we will compare work related to our system and identify points of difference between them and mention the advantages of each work like our system.

In the process of requirements elicitation, we will describe system requirements and functional and non-functional requirements in the natural language, and we will Elicit them through brainstorming, use similar systems requirements, use the Lucid platform tool to generate ideas for the requirements.

In the process of requirements specification, we will construct tables to describe both functional and non-functional requirements.

In the process of requirements modeling, we will draw use cases that show all possible interactions with the system and write the description for them.

## Analysis of Existing Systems

At this stage, we compared our system with similar systems through features. Then we used some of the features of similar systems at the stage of the elicitation requirements.

The checkmark  represents that the advantage will be achieved in the system whereas the  shows that it will not be achieved in the system. “**?**” it shows that it is possible to try to make a feature in the system. The differences between our system and other comparable work are shown in Table 3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| System  Feature | Lucid platform | Visual paradigm | StarUML | Enterprise Architect | MagicDraw | ChatUML | Our project |
| Create UML diagrams |  |  |  |  |  |  |  |
| Code Generation |  |  |  |  |  |  | ? |
| Smart Requirements Management |  |  |  |  |  |  |  |
| Automated generation of drawings. |  |  |  |  |  |  |  |
| Intelligent pattern recognition |  |  |  |  |  |  |  |
| Real-time Diagram validation |  |  |  |  |  |  |  |
| Suggesting Relevant Elements |  |  |  |  |  |  |  |
| Detecting Errors |  |  |  |  |  |  |  |
| Predictive Modeling |  |  |  |  |  |  |  |
| Conflict Analysis |  |  |  |  |  |  |  |
| Idea Generation |  |  |  |  |  |  |  |
| Idea Sorting |  |  |  |  |  |  |  |
| Idea Summarization |  |  |  |  |  |  |  |

Table 3: Related Work Comparison

We have identified some features that will not be achieved in the system for many reasons for each feature:

* Intelligent pattern recognition: because we have identified that the system analyzes the text or document through the context of the text through Named Entity Recognition. It classifies specific words such as names, locations and objects. It is divided into two sections: structural and semantic information.
* Real-time Diagram validation: because our system allows the user to enter text by typing and by upload file and at upload file, we do not need to process changes in real-time.
* Detecting Errors: because our system allows a certain percentage of requirements errors to be ignored by the user.
* Predictive Modeling: because our system does not predict future data and trends.
* Conflict Analysis: because our system does not identify inconsistencies between requirements and models but only generates them and displays them to the user.
* Idea Generation and Idea Sorting and Idea Summarization: because the system does not have the functions of generating ideas and sorting the ideas generated and not summarizing them.

## Requirements Elicitation

Requirements engineering is the process of discovering all system services, restrictions imposed on it, and services that meet user needs, analyzing, documenting, and verifying these services [5]. It includes high-level activities, including elicitation, analysis, and other activities. Requirements elicitation is the collection of all requirements, including system requirements and user requirements [5]. System requirements describe all the requirements that we want the system to perform [5]. After the elicitation process, we extracted the system requirements, and from them we identified the functional and non-functional requirements through brainstorming, using similar requirements tools, and using the Lucid platform tool.

### System Requirements

A structured document setting out detailed descriptions of the system’s services and operational constraints [6].

Defines what should be implemented so may be part of a contract between client and contractor [6].

* 1. To build an AI website system to receive software development requirements texts and documents and use NLP techniques to analyse them.
  2. To extract classes, functions (methods / procedure) and relations between classes and function.
  3. To generate UML diagrams: use-cases and class diagram.

### Functional Requirements

The requirements determine the system's behavior, what it should and shouldn't provide, and how it interacts with inputs [5].

1. To build an AI website system to receive software development requirements texts and documents and use NLP techniques to analyses them.
   1. The system shall allow the user to enter requirements either by typing directly or by uploading a document.
   2. The system shall allow the user to create analysis model.
   3. The system shall allow the user to choose type of UML diagram such as (class, use case).
   4. The system shall prepare to clean and organize the initial data to perform the text analysis process.
   5. The system should analyze the text by doing Tokenization, Parsing and Relation Extraction.
   6. The system should allow the user to export the diagram by png format.
2. To extract classes, functions (methods / procedure) and relations between classes and function.
   1. The system shall analyses the requirements context using NLP techniques.
3. To generate UML diagrams: use-cases and class diagram.
   1. The system shall draw the diagram after generated by NLP techniques.

### Non-Functional Requirements

It describes the characteristics of the system and the constraints imposed on the services provided by the system such as time constraints and other constraints. It also describes specific standards for some of the system's services [5].

1. To build an AI website system to receive software development requirements texts and documents and use NLP techniques to analyses them.
2. The system must be fast, ensuring that the processing of requirements and drawing (use case or class diagram) does not exceed (write the time).
3. The resulting diagrams in the system should have an accuracy of no less than (write the percentage).
4. It should be able to handle (write the range of users) simultaneously.
5. The system should be user-friendly and adaptable to various devices to provide a seamless experience.
6. The system should perform its functions with minimal steps.
7. To extract classes, functions (methods / procedure) and relations between classes and function.
8. The system should be able to recognize and analyze a minimum of (write the percentage) of user requirements entered, with a permissible margin for error not exceeding (write the percentage).

## Requirements Specification

|  |  |
| --- | --- |
| Upload Dataset | Use case |
| Developer, DBMS | Actor |
| 1. the user uploading a dataset into the system. 2. After the user uploads successfully the data uploads to Training Dataset and Testing Dataset.  * Training Dataset: The training dataset is used to train the machine learning model. * Testing Dataset: The testing dataset is a separate set of examples that the model has not seen during training. | Description |
| The system should be in a state ready to receive the dataset, with sufficient storage and processing resources available. | Pre-condition |
| The dataset is successfully uploaded and stored in the system. | Postcondition |

Table 4:use-case (Upload Dataset)

|  |  |
| --- | --- |
| load Dataset | Use case |
| DBMS | Actor |
| 1. After the Training Dataset and Testing Dataset 2. The data load to use to train Model and test model and Prepare data | Description |
| The data loading system is set up and accessible to users. | Pre-condition |
| After load Dataset can use the loaded dataset to generate train Model and test model and Prepare data | Postcondition |

Table 5: use-case (Load Dataset)

|  |  |
| --- | --- |
| train Model | Use case |
| Developer | Actor |
| 1. After load data from training dataset 2. The train model Training involves the model learning patterns and relationships within the data, which is a form of analysis. 3. After the training, the developer analyzes how well the model is learning from the data. 4. The developer assesses the performance of the trained model using a separate testing dataset. | Description |
| A labeled dataset is available for training. | Pre-condition |
| The machine learning model has completed the training process successfully. | Postcondition |

Table 6: use-case (train Model)

|  |  |
| --- | --- |
| Analysis | Use case |
| Developer | Actor |
| 1. After training model 2. Prepare data from load dataset perform initial cleaning steps.  * Tokenization: tokenization is the process of breaking the text into individual units for analysis. * Clean Data: This step is crucial to ensure accuracy and quality of results in later stages.  1. Extraction NERs; typically involves identifying and classifying entities, such as names of people. 2. Relation Extraction: module identifies and classifies relationships between entities in unstructured text data. 3. Extraction Event: A module extracts events from text, offering insights into specific occurrences. 4. After analysis is completed generate Diagram Specs get all the Analysis specifications like number of the classes and the relation between them and then draw | Description |
| A dataset has been successfully loaded into the data processing system | Pre-condition |
| the analysis is completed and ready to draw | Postcondition |

Table 7:use-case (Analysis)

|  |  |
| --- | --- |
| test Model | Use case |
| Developer | Actor |
| 1. The Developer loads data from testing dataset Use a separate dataset not used during training Model. 2. After loading data start representing real-world scenarios and cover the kind of cases the model may encounter. 3. The developer applies the loaded model to the testing dataset to make predictions. 4. The developer analyzes the model's predictions and evaluates its performance using metrics such as accuracy, precision, recall, and F1 score. | Description |
| A testing dataset, distinct from the training data, is available. | Pre-condition |
| The model's performance is evaluated based on the testing dataset. | Postcondition |

Table 8:use-case (test Model)

|  |  |
| --- | --- |
| Use case | Enter requirements Document |
| Actor | User |
| Description | 1. The user clicks on the text box. 2. The user enters requirements by typing.   Or The user clicks on the select doc. |
| Pre-condition | The user enters the website. |
| Post-condition | The requirements are entered successfully |

Table 9: use-case (Enter requirements Document)

|  |  |
| --- | --- |
| Use case | Analysis |
| Actor | User |
| Description | 1. After the user entered the requirements. 2. Preparation: Analysts clean and structure raw data for analysis  * Tokenization: Resumes are tokenized into individual words or phrases * Clean Data: This step is crucial to ensure accuracy and quality of results in later stages.  1. Extraction NERs; typically involves identifying and classifying entities, such as names of people. 2. Relation Extraction: module identifies and classifies relationships between entities in unstructured text data. 3. information Extraction: system gathers structured information from various sources, preparing it in a usable format for analysis. 4. Event Extraction: A module extracts events from text, offering insights into specific occurrences. 5. After analysis is completed generate Diagram Specs get the analysis like number the class and the relation between them and then draw |
| Pre-condition | Data is available for analysis. |
| Post-condition | the analysis is completed |

Table 10: use-case (analysis Document)

|  |  |
| --- | --- |
| Use case | Draw analysis model |
| Actor | User |
| Description | 1. The user clicks on select type. 2. The system shows to type of model (class model, use-case model)   If the user chooses class model   1. The system shows class model on screen after the user start generate.   If the user chooses class model   1. The system shows use-case model on screen after the user start generate |
| Pre-condition | The system will determine whether it is the context or Document to Draw the Diagram. |
| Post-condition | The selected model type is displayed on the screen. |

Table 11: use-case (Create analysis model)

|  |  |
| --- | --- |
| Use case | save diagram |
| Actor | User, DBMS |
| Description | 1. After the system draw the model (class, use-case). 2. The user click save diagram. 3. The system prompts the user to choose a location and provide a name for the saved diagram. 4. The user selects the destination folder and enters a name for the diagram file. 5. The system determines the type of model (e.g., class diagram, use case diagram) based on the previously chosen model type. 6. The system saves the diagram data to the designated location using the DBMS for storage. 7. The system displays a confirmation message indicating that the diagram has been successfully saved. |
| Pre-condition | The selected model type is displayed on the screen. |
| Post-condition | The model is successfully saved to the specified location, and the user receives a confirmation message. |

Table 12: use-case (save diagram)

|  |  |
| --- | --- |
| Use case | use-case diagram |
| Actor | User |
| Description | 1. After analysis completed the system draws a use-case model after the requirements. 2. The system displays a use-case model on screen. |
| Pre-condition | The system is ready and prepared to start drawing the diagram |
| Post-condition | The diagram has been successfully drawn by the system. |

Table 13: use-case (Draw use-case model)

|  |  |
| --- | --- |
| Use case | class diagram |
| Actor | User |
| Description | 1. The system draws a class model after the analysis requirements are completed. 2. The system displays class models on screen. |
| Pre-condition | The system is ready and prepared to start drawing the diagram |
| Post-condition | The diagram has been successfully drawn by the system. |

Table 14: use-case (Draw class model)

## Requirements Modeling

### Use case diagrams



Figure 17:Devolper use-case diagram

The developer upload dataset and then the data uploading to Training Dataset and Testing Dataset The data load to use to train Model and test model and Prepare data the train Model load data from training dataset, the training model learning patterns, and relationships within the data, which is a form of analysis. the developer analyzes how well the model is learning from the data and assesses the performance of the trained model using a separate testing dataset after the Analysis It contains Prepare and Tokenization, Clean Data Extraction NERs Relation Extraction Event and then After analysis is completed generate Diagram Specs get all the Analysis specifications and draw analysis model uses this specifications to draw the diagram, test model it use separate dataset not used during training Model represent real-world scenarios, The developer applies the loaded model to the testing dataset to make predictions and analyzes the model's predictions evaluates its performance using such as accuracy, precision, recall, and F1 score,



Figure 18: user use-case diagram

The user can enter requirements and then Upload data and then load data to analyze by Prepare and Tokenization, Clean Data Extraction NERs Relation Extraction Event and then After analysis is completed generate Diagram Specs get all the Analysis specifications and draw analysis model uses these specifications to draw the diagram whither it is Usecases diagram, class diagram depends on the user what need

### Class Diagram



Figure 19: class diagram

The GenerateDiagramSpecs class identifies the characteristics of diagram generator we take these properties from class analyze and use class diagram.

The class analyze consists of five classes:

1. ExtractNERs class:

This class identifying and classifying entities from the text.

1. ExtractEvent class:

This class is about finding events from the text.

1. ExtractRelations class:

This class identifies and classifies relationships between entities.

1. DataPreparation class:

This class does clean data and Tokenization for text.

1. ExtractRelations class:

This class identifies and classifies relationships between entities.

Class diagram take name and the boundary of diagram and drawing the diagram such as class digarm and use case .

User class it allows to user to entre requirement.

Developer class it allows to train and test the model and he can upload the data set.

DBMS class it can opload and load the data .

Model class it can evaluate the dataset after train and test the data

Trin class train and test do testing training for data.

## Summary

In this chapter, we analyzed the system, identified the methodology of analysis and development, identified the functional and non-functional requirements, and displayed them in the format of tables and UML diagrams such as use case and class diagram.

After analyzing the system, we will be ready in chapter 4 to design the system and determine the architecture of the system and design the user interface.

# Chapter 4: System Design

## Introduction

This chapter covers system design, including structural design, component design, data modelling, and user interface design. Structural design focuses on organizing the system's structure to ensure efficient performance, while component design addresses the interaction of software components to achieve functional goals. Data modelling design deals with efficiently organizing and storing data, and user interface design highlights improving the user experience through an effective and attractive interface.

## Design Methodology



Figure 20:Design methodolgy ((\*): It will be worked on in the second term)

**Architectural Design:**

The process of determining the overall structure and organization of a software system, involving high-level decisions on system configuration, key component specification, and their interactions.

**Component Design:**

The process of dividing a system into manageable units, with each unit representing a specific module or function within the software.

**Data Modeling Design:**

The process of defining and organizing data requirements for a system, creating a conceptual representation of data, defining relationships between entities, and specifying data storage and access.

**User Interface Design:**

The process of creating an easily understandable and visually appealing interface for users to interact with software, including the design of structure, navigation, and visual elements. Goals include enhancing user experience through user-friendly and efficient interface design.

## Architectural Design

The architectural design is concerned with understanding how a system is organized and designing the overall structure of the system, it involves identifying major system components and their communications.



Figure 21: Architectural Design

We opted for the client-server architecture because it allows for the distribution of tasks between the client and the server. The client requests services or data from the server, and the server fulfils these requests. In our system, we will employ the 3-Tier client-server model.

The second tier consists of a layered model with six layers, representing a sequence of steps that can be gradually executed on the data sent from the client. This division facilitates a comprehensive and organized examination, improving the quality of the transmitted data.

In our system, the client in the first tier sends requirements. In the second tier, these requirements undergo examination and analysis before reaching the third tier, which is our system responsible for fulfilling the client's requests.

## Component Design

Our system is divided into two main components: document analysis and drawing diagram. Each component has several classes. The component document analysis includes preparing Data, Analyse, training the model, testing the model, and uploading the Dataset. The component diagram drawing includes a draw analysis model, use case diagram and class diagram.

This phase will be executed in the second stage of the project due to time constraints.

## Data Modeling Design

Designing a data model involves creating a structure to organize data in the system for efficient storage and management. We'll divide the data into two parts: one for training the model and another for testing it. The aim is to use the training part to teach the model using available data, adjusting its settings to improve performance. On the flip side, we use the testing part to see how well the model works on new data. This helps by giving the model new data and checking how accurately it can make predictions. This split lets us check the model's efficiency and gives a precise idea of how well it handles new information. It all contributes to making machine learning better and evaluating statistical models.

This phase will be executed in the second stage of the project due to time constraints.

## User Interface Design



Figure 22: Home Page



Figure 23: Input Page



Figure 24: Diagram Page

## Summary

In this chapter, we have created the Architectural design and utilized the client-server pattern. The component and data modeling design will be developed in the next phase. Finally, we have initiated the design of the user interface.

In the upcoming chapter, we will summarize everything we have accomplished since the beginning of the project. We will discuss how we achieved our goals, the lessons Learnt, and future work.

# Chapter 5: Conclusion and Future Work

## Conclusion

In this section, we will summarize all the work we have done from the beginning to the end of the project. We started by identifying the problem and how we would solve it, then defined the project's goals, methodology, and project plan.

In the second chapter, we researched topics related to the problem we are solving, including AI, ML, and NLP, and how to develop a system for artificial intelligence. We also explored similar systems.

In the third chapter, we analyzed existing systems, identified requirements from these systems, conducted brainstorming sessions, and then created the use-case diagram. We described the use cases and drew the class diagram.

In the fourth chapter, we developed the architectural design and user interface design.

## Goals Achieved

We have successfully achieved our project goals. Here's a overview of the objectives and how we accomplished them.

1. **To review previous works that can be related to our project domain:**

We conducted research on technologies related to our project and similar systems.

1. **To finish the Analysis phase of the proposed system to be built:**

We identified requirements after analyzing similar systems and used brainstorming techniques. We also created a use-case and described them.

1. **To do the preliminary Design tasks of the suggested system:**

We developed initial designs for the project, including class diagrams, architectural engineering design, and the user interface.

1. **To conclude with the lessons learnt and knowledge and experience gained from working on this project:**

We documented any new knowledge and experience gained from the beginning to the end of the project.

## Lessons Learnt

1. **Working as a team-** This project helped us to work together, learn and benefit from each other when we needed each other.
2. **Project planning**- involves setting the goals of the project, creating a task list for each goal, and developing a weekly schedule. Tasks are reviewed and assigned every weekend, and completed work is presented to the supervisor for feedback and approval.
3. **Learn NLP**- NLP is a new system for us, and we did not study anything about it in our university courses.
4. **Improve writing skills-** One of the most important things we learned in our project is to improve writing skill.
5. **Learning System Development for AI (NLP, ML)**- taught us the development steps in the fields of Natural Language Processing (NLP) and Machine Learning (ML).
6. **Researched and Read papers**- Acquired knowledge of methods for searching for reliable references and extracting information from them.
7. **Writing References**- Acquired knowledge of how to write references in the IEEE style.
8. **Every software project has different types of requirements**.

## Limitations and Future Work

### Limitations

The work we haven't done yet is component design and data modeling design due to time constraints. We will implement them in the second phase of the project.

### Future work

The tasks we will undertake in the second phase of the project include:

1. Improving analysis and design, which we worked on in this phase of the project.
2. Component design
3. Data modeling design
4. Project development
5. Project testing

# References

|  |  |
| --- | --- |
| [1] | "lucidchart vs lucidspark," innovation training, [Online]. Available: https://www.innovationtraining.org/lucidchart-vs-lucidspark-when-why-and-how-to-use-them-both/. |
| [2] | A. Tisdale, "Quora," [Online]. Available: https://www.quora.com/What-is-the-architecture-of-LucidChart/answer/Ashley-Tisdale-Digital-Marketing-Expert. |
| [3] | B. Dilts, "Quora," [Online]. Available: https://www.quora.com/What-is-the-architecture-of-LucidChart/answer/Ben-Dilts. |
| [4] | https://docs.chatuml.com/docs/overview/know-the-ui, 2024. |
| [5] | S. Ian, Software Engineering, 11th ed., Addison-Wesley,, 2015. |
| [6] | R. J. Leach, Introduction to Software Engineering, 2016. |