School of Computer Sciences

CAT405 Intelligent Computing Major Project

System Requirements and Design Report

Automating Feedback Suggestion for Written Assignments (SemakLah)

Ching Jia Ying

153463

Supervisor: Dr Jasy Liew Suet Yan

Examiner: Associate Professor Dr. Wong Li Pei

Academic Session

2023/2024

ABSTRAK

Usaha mengajar pelajar membuat tugasan bertulis teknikal boleh menjadi tugasan yang meletihkan terutamanya untuk kelas yang mempunyai ramai pelajar. Ramai pensyarah mungkin memilih untuk memberikan markah akhir tanpa sebarang maklum balas, manakala yang memberikan maklum balas mungkin tidak konsisten disebabkan bilangan besar tugasan yang dihantar. Sebaliknya, pelajar sukar untuk memperbaiki tugasan mereka apabila maklum balas tidak diberikan. Oleh itu, projek ini bertujuan untuk mencipta aplikasi web yang dapat menjalankan penilaian separa automatik untuk membantu instruktor dalam proses penilaian. Objektif sistem ini adalah untuk menyediakan bilik darjah dalam talian di mana pelajar boleh memuat naik tugasan digital mereka sementara instruktor boleh memberikan maklum balas dan menilai tugasan pelajar. Semasa proses penilaian, sistem melakukan ekstraksi teks, dan menerapkan pemprosesan bahasa tabii untuk analisis leksikal, menghasilkan cadangan maklum balas berdasarkan maklum balas sejarah yang diberikan oleh instruktor dengan pemadanan kemiripan teks, serta membolehkan instruktor memilih untuk mengedit maklum balas dan pelajar untuk melihat maklum balas tersebut. Hasil yang dijangka daripada projek ini adalah aplikasi web yang boleh menjalankan pengurusan bilik darjah, pengurusan tugasan, penghasilan maklum balas, dan penilaian. Penyelesaian yang dicadangkan menawarkan beberapa faedah kepada instruktor dan pelajar, termasuk maklum balas yang tepat dan berkesan, beban kerja yang berkurang bagi instruktor, penilaian yang konsisten dan adil, dan hasil akademik yang lebih baik bagi pelajar.

Kata Kunci: Penilaian Separuh Automatik, Penjanaan Maklum Balas, Pemprosesan Bahasa Tabii, Pemadanan Kemiripan Teks

ABSTRACT

It is tedious work for instructors to provide timely feedback on written assignments, particularly for classes with many students. Many instructors will choose to give the final grades without any feedback, while those who choose to give feedback may not be consistent due to the massive numbers of submitted assignments. On the contrary, students could not improve better in their assignments when feedback is not given. Hence, this project aims to create a web application that can perform semi-automatic grading to aid instructor in the grading process. The objectives of this system are to provide an online classroom where students can upload their digital assignments whilst instructors can provide feedback and grade students' assignments. During the grading process, the system performs text extraction, and apply natural language processing for lexical analysis, generate feedback suggestion based on the historical feedback given by instructor using text-similarities matching, and allow instructors to choose to edit the feedback and students to view the feedback. The expected outcomes of the project are a web application that can perform classroom management, assignment management, feedback generation, and grading. The proposed solution offers several benefits to both instructors and students, including timely and effective feedback, reduced workload for instructors, consistent and fair grading, and improved academic outcomes for students.

Keywords: Semi-automatic Grading, Feedback Generation, Natural Language Processing, Text-Similarities Matching

TABLE OF CONTENTS

ABSTRAK....2

ABSTRACT....3

TABLE OF CONTENTS....4

LIST OF TABLES....7

LIST OF FIGURES....8

LIST OF ABBREVIATIONS AND SYMBOLS....10

1 INTRODUCTION....11

1.1. Project Background....11

1.2. Problem Statement....12

1.3. Motivation....12

1.4. System Objectives....13

1.5. Proposed Solution....14

1.5.1. Classroom Management....14

1.5.2. Assignment Management....14

1.5.3. Feedback Generation....15

1.5.4. Grading Module....15

1.6. Benefits and Uniqueness....16

1.7. SDG Alignment....17

1.8. Report Organization....17

2 BACKGROUND & RELATED WORK....18

2.1. Background 18

2.2. Related Work 19

2.2.1. Existing Systems 19

2.2.2. Competitor Analysis 23

2.2.3. Existing Algorithms 24

3 SYSTEM REQUIREMENTS AND DESIGN 26

3.1. Project Scope 26

3.2. System Capabilities 27

3.3. System Limitations 28

3.4. Project Management 29

3.4.1. Work Breakdown Structure (WBS) 29

3.4.2. Gantt Chart 30

3.4.3. SWOT Analysis 31

3.5. Development Methodology 32

3.6. User Requirements 33

3.6.1. Requirement Gathering 33

3.6.2. Functional Requirements 34

3.6.3. Non-Functional Requirements 36

3.7. System Analysis 37

3.7.1. Use Case Diagram 37

3.7.2. Use Case Descriptions 38

3.7.3. Entity Relationship Diagram (ERD) 56

3.7.4. Sequence Diagram 57

3.8. Intelligent Components 65

3.8.1. Lexical Analysis for Language and Grammatical Quality Assessment 65

3.8.2. Feedback Suggestion Generation 68

3.9. Data Source 69

3.10. Technology Deployed 70

3.10.1. Hardware 70

3.10.2. Software 70

3.10.3. Programming Languages/Tools 70

3.10.4. Chosen algorithm 70

4 CONCLUSION 71

LIST OF TABLES

Table 1 Competitor Analysis 23

Table 2 Work Breakdown Structure (WBS) 29

Table 3 SWOT Analysis 31

Table 4 Use Case Description for UC001 Login Account 38

Table 5 Use Case Description for UC002 Register Account 39

Table 6 Use Case Description for UC003 Manage Classroom 40

Table 7 Use Case Description for UC004 Manage Student Enrolment 41

Table 8 Use Case Description for UC005 Manage Assignment 42

Table 9 Use Case Description for UC006 Set Assignment Attributes 43

Table 10 Use Case Description for UC007 Manage Grading Rubric 44

Table 11 Use Case Description for UC008 Manage Student Submissions 45

Table 12 Use Case Description for UC009 Download Assignment 46

Table 13 Use Case Description for UC010 Manage Assignment Submission 47

Table 14 Use Case Description for UC011 View Grade 48

Table 15 Use Case Description for UC012 Review Feedback 49

Table 16 Use Case Description for UC013 Provide Feedback 50

Table 17 Use Case Description for UC014 Generate Feedback Suggestions 51

Table 18 Use Case Description for UC015 Perform Lexical Analysis 52

Table 19 Use Case Description for UC016 Grade Student Assignment 53

Table 20 Use Case Description for UC017 View Dashboard 54

Table 21 Use Case Description for UC018 Return Graded Assignment 55

LIST OF FIGURES

Figure 1 Screenshot of FeedbackFruits...19

Figure 2 Screenshot of gradescope 20

Figure 3 Screenshot of Formative.AI...21

Figure 4 Screenshot of codePost...22

Figure 5 Gantt Chart 30

Figure 6 Use Case Diagram 37

Figure 7 Entity Relationship Diagram (ERD) 56

Figure 8 Sequence Diagram for Login Account..57

Figure 9 Sequence Diagram for Register Account 57

Figure 10 Sequence Diagram for Manage Classroom...58

Figure 11 Sequence Diagram for Manage Student Enrolment...............................58

Figure 12 Sequence Diagram for Manage Assignment 59

Figure 13 Sequence Diagram for Set Assignment Attributes.................................59

Figure 14 Sequence Diagram for Manage Grading Rubric 60

Figure 15 Sequence Diagram for Manage Student Submissions............................60

Figure 16 Sequence Diagram for Download Assignment 61

Figure 17 Sequence Diagram for Manage Assignment Submission 61

Figure 18 Sequence Diagram for View Grade...62

Figure 19 Sequence Diagram for Review Feedback...62

Figure 20 Sequence Diagram for Provide Feedback, Generate Feedback Suggestions 63

Figure 21 Sequence Diagram for Grade Student Assignment 63

Figure 22 Sequence Diagram for View Dashboard 64

Figure 23 Sequence Diagram for Return Graded Assignment 64

Figure 24 General Flow of Lexical Analysis for Language and Grammatical Quality Assessment 65

Figure 25 Flow of Echant Library 66

Figure 26 Flow of language-tool-python Library 67

Figure 27 Flow of Feedback Suggestion Generation 68

LIST OF ABBREVIATIONS AND SYMBOLS

ERD	- Entity Relationship Diagram
IDE	- Integrated Development Environment
LMS	- Learning Management System
WBS	- Work Breakdown Structure
SMC	- Spelling Mistake Correction
TF-IDF	- Term Frequency and Inverse Document Frequency
OS	- Operating System
UML	- Unified Modeling Language
HTML	- HyperText Markup Language
CSS	- Cascading Style Sheets
NLTK	- Natural Language Toolkit

1 INTRODUCTION

1.1. Project Background

Over the past decade, the enrollment in higher education in Malaysia has seen a significant increase with over 400,000 new students enrolling in higher education institutions (HEIs) every year. The gross higher education enrollment rate in Malaysia reached 48% in 2012, representing a 70% increase in enrollment from 2004 to 2014, with 1.2 million students enrolled in both public and private HEIs. According to the Ministry of Education in Malaysian there are currently 20 public universities, 36 polytechnics, and 105 community colleges [1]. The number of students enrolled strikes up to the highest in 2016 with 1,346,858 students. In addition, the general statistic of students and instructor ratio at public universities was approximately 18.93 students for every instructor available in 2022 [2]. However, the number of students in class can be more than a few hundred depending on courses.

With the massive number of students, it is challenging for instructors to provide quality feedback to each student due to the increasing workload and a growing demand for personalized feedback on technical written assignments. This leads to the results of dissatisfaction and limited engagement between instructor and students [3]. Other than that, the grading process can be repetitive especially for open-ended technical written assignments, it is a tedious and time-consuming process for instructor to provide personalized feedback. As for students at university level, they usually do not get any feedback from the instructors on their assignments, which is not helpful for the improvement of students. According to Harvard Graduate School of Education, feedback should be given in a timely manner to help students understand what they are doing correctly and incorrectly, with the focus on what they are doing in their assignments [4].

Therefore, SemakLah, is proposed that aims to create a web application that can perform semi-automatic grading and suggest feedback based on components in a grading rubric on technical written assignments submitted by university level students. On the contrary, students can upload their assignments in PDF or Word format, and they can review the feedback from the instructors.

1.2. Problem Statement

The problem statement is that instructors face difficulties in providing timely and consistent feedback on technical written assignments, particularly when dealing with many students in a classroom. This is due to the increasing workload of the instructor, especially during the feedback and grading process. Grading technical written assignments is a tedious and time-consuming process for instructors. Plus, providing personalized feedback, and going through the assignment for each student in detail is even more challenging. Instructor would not know the lexical performance of the student by having a glance at their assignments.

Additionally, instructor might want to reuse the same feedback for the similar mistakes made by different students. With that being said, providing feedback and grading the same questions repeatedly from different students can be monotonous for instructors, and they could have utilized this time to work on more meaningful tasks such as preparing lessons or conducting research.

The lack of timely feedback on technical written assignments can lead to several problems. Firstly, students do not receive feedback on their work, which means they cannot learn from their mistakes and improve their performance. This can lead to dissatisfaction for both instructors and students, as well as a lack of engagement and motivation for students to improve their work [3]. Secondly, students may not be able to identify their weaknesses and strengths, which can hinder their learning progress.

1.3. Motivation

This project is driven by the need to overcome the challenges faced by the instructor in the current grading process. The motivation of this project is to create a web application that will encourage and help instructors to provide timely, effective, and consistent feedback to students as well as reducing the workload for the instructors. Other than that, the desire of students to learn from their mistakes through receiving feedback for their assignments also motivates the creation of this project.

1.4. System Objectives

The objectives of this system are:

- To provide an online assignment management system in which students can upload their written assignments whilst instructors can semi-automatically review student assignments.
- To perform lexical analysis to access overall language and grammatical quality of the writing assignments.
- To provide instructors with automatic feedback suggestions based on student solutions.
- To provide instructors with grading features to grade students' assignments.

1.5. Proposed Solution

SemakLah, is the proposed web application that can perform semi-automatic grading and generate feedback suggestions based on historical feedback given by instructor in consideration of components in a grading rubric.

SemakLah consists of 4 modules, which are Classroom Management, Assignment Management, Feedback Generation and Grading. Instructor can set up a classroom with assignments for students. Students can download the assignment and they will upload their assignment in (PDF or Word) for the assignment submission. When instructor is reviewing the assignment, the system will perform text extraction for lexical analysis using the natural language processing techniques. The instructor can create feedback for the students, and the system will generate top 3 feedback suggestions (based on the historical feedback given by instructor) when the instructor highlights a similar text. The instructors can choose to accept the suggested feedback as it is, edit or overwrite the suggested feedback. Other than providing feedback, instructors can also grade students' assignments. Once grading is completed and the instructor publishes the feedback and grades, students can view the feedback through SemakLah.

Module Features:

1.5.1. Classroom Management

1. Log in for instructors and students.
2. Registration for new users.
3. Management of online classrooms with different courses.
4. Management of student enrolment to online classrooms.

1.5.2. Assignment Management

1. Creation and editing of assignments.
2. Setting assignment attributes (title, description, instructions, start date, end date, file upload)
3. Management of the list of student submissions.
4. Downloading assignments for students.
5. Uploading, editing, and reviewing assignment submissions by students.

1.5.3. Feedback Generation

1. Extraction of digital text from assignment files.
2. Lexical analysis for spelling and grammatical errors.
3. Similarity matching with feedback history.
4. Top 3 feedback suggestions based on history and highlighted text at the sentence level.
5. Review of the summary of lexical analysis.
6. Highlighting text to provide feedback.
7. Selection of feedback suggestions and editing.

1.5.4. Grading Module

1. Management of grading rubrics with different weightage.
2. Entry of scores according to grading rubrics.
3. Viewing generated final scores and grades for each student assignment.
4. Reviewing the dashboard of grading progress.
5. Returning feedback and grades for students to review.
6. Reviewing feedback and grades given by instructors for students.

1.6. Benefits and Uniqueness

SemakLah, the proposed web application for semi-automatic grading and feedback generation offers several benefits to both instructors and students.

First, instructors can provide timely and effective feedback to students. This is particularly important in classes with many students, where providing feedback on written assignments can be a tedious task. With SemakLah, instructors can save time on giving feedback for students' assignments, allowing them to focus on other important tasks.

Second, the workload for instructors can be reduced. By automating the grading and feedback generation process, instructors can spend less time grading assignments and more time on other important tasks, such as lesson planning and research. This can lead to increased job satisfaction and a better work-life balance for instructors.

Third, students can receive feedback that is consistent, fair, and objective. With the use of grading rubrics and natural language processing techniques, the feedback provided will be more consistent and objective, ensuring that all students are graded fairly. This can help to improve student engagement and motivation, as they will be able to see how they are progressing and what they need to do to improve their work.

Fourth, students can view their feedback through the application, which can help them improve their work. By providing students with access to their feedback through the application, they can review their work and identify areas for improvement. This can help to improve the overall quality of assignments and reports submitted by students, leading to better academic outcomes.

Finally, SemakLah is unique because it combines the use of grading rubrics with natural language processing techniques to generate feedback at two different levels of analysis: lexical and rubric. The system also allows instructors to choose to accept the suggested feedback as it is, edit or overwrite the suggested feedback.

1.7. SDG Alignment

SemakLah is aligned with the SDG 4: Quality Education, as it aims to improve the quality of education by providing timely and effective feedback to students.

1.8. Report Organization

This report is organised into 4 sections:

Section 1 – Introduction:

- This section includes project background, problem statements, motivation, system objectives, proposed solutions, and benefits and uniqueness of the proposed solution.

Section 2 – Background & Related Work:

- This section includes the existing systems, competitor analysis and existing algorithms/methods.

Section 3 – System Requirements and Designs:

- This section includes the project scope, system capabilities, system limitations, project management, development methodology, requirements, and detailed analysis of the proposed solution.

Section 4 – Conclusion:

- This section concludes this system requirements and design.

2 BACKGROUND & RELATED WORK

2.1. Background

SemakLah is a new project that will be built from scratch and has never been developed before.

Semaklah is a web application designed for semi-automatic grading, incorporating a feedback generation system based on historical feedback from instructors and grading rubric components. The system includes classroom management features, allowing the creation of classrooms, assignments, and grading rubrics. Students download assignments, submit them in PDF or Word format. When the instructor is reviewing, the system performs lexical analysis using natural language processing techniques. Instructors can create feedback, with the system suggesting top 3 feedback based on historical data when similar text is highlighted. Instructors have the flexibility to accept, edit, or overwrite suggested feedback. Additionally, instructors can grade assignments, and once grading is complete, they can return feedback and grades for students to access through the system.

2.2. Related Work

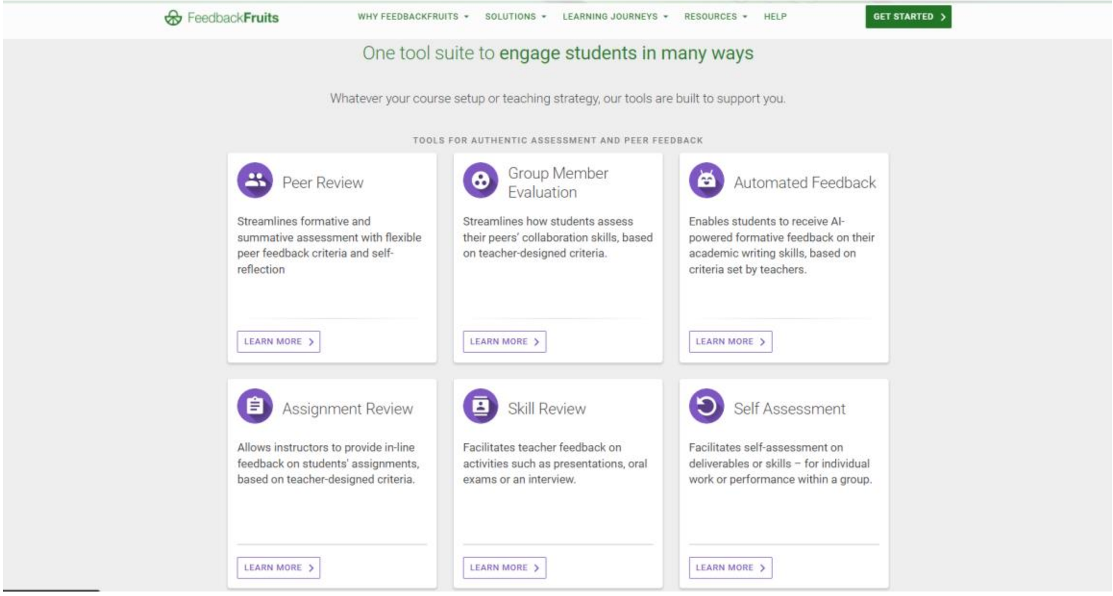
2.2.1. Existing Systems

There are a few existing systems that have similar scope including Feedbackfruit, gradescope, formative.AI and codePost.

2.2..1 FeedbackFruits

FeedbackFruits is a digital teaching tool for higher education that enhances student engagement, collaboration, and feedback in both asynchronous and synchronous learning. FeedbackFruits allows teachers to structure and customize feedback criteria with options such as comment criterion, (Likert) scales, rubrics, or a combination of multiple types. The platform allows teachers to reuse any feedback comment they post within an assignment, saving time in cases where there is recurring feedback that must be given on multiple students' deliverables. However, it does not perform any lexical analysis and it does not automatically generate feedback suggestions based on the historical feedback when the instructor highlights a similar text [5].

Figure 1 Screenshot of FeedbackFruits



One tool suite to engage students in many ways

Whatever your course setup or teaching strategy, our tools are built to support you.

TOOLS FOR AUTHENTIC ASSESSMENT AND PEER FEEDBACK

Peer Review

Streamlines formative and summative assessment with flexible peer feedback criteria and self-reflection

LEARN MORE

Group Member Evaluation

Streamlines how students assess their peers' collaboration skills, based on teacher-designed criteria.

LEARN MORE

Automated Feedback

Enables students to receive AI-powered formative feedback on their academic writing skills, based on criteria set by teachers.

LEARN MORE

Assignment Review

Allows instructors to provide in-line feedback on students' assignments, based on teacher-designed criteria.

LEARN MORE

Skill Review

Facilitates teacher feedback on activities such as presentations, oral exams or an interview.

LEARN MORE

Self Assessment

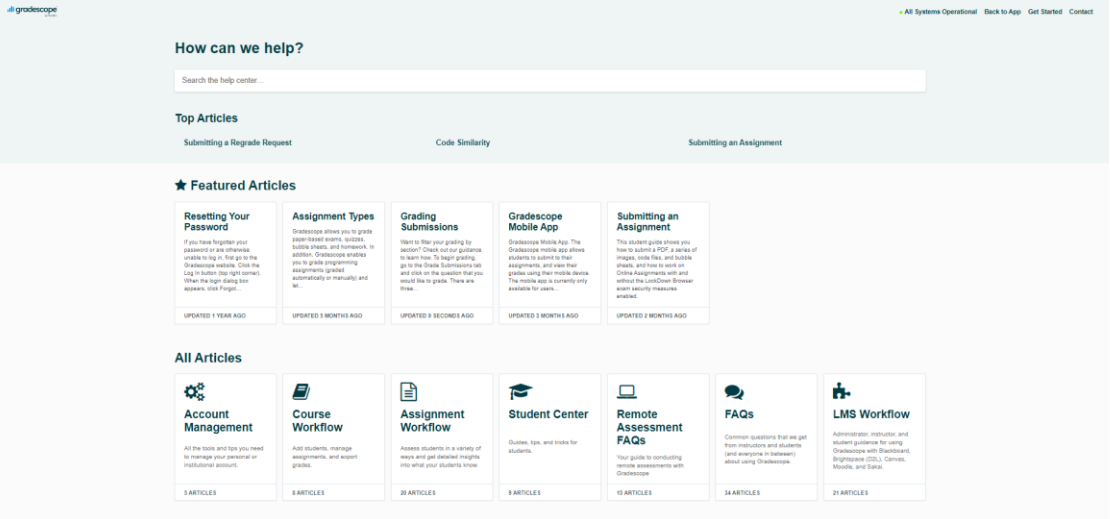
Facilitates self-assessment on deliverables or skills – for individual work or performance within a group.

LEARN MORE

2.2..2 Gradescope

Gradescope is an online tool that allows instructors to grade various types of assignments, including paper-based and programming assignments. It offers features such as creating and using rubrics for grading, managing submissions, and providing detailed analytics. For paper assignments, it supports grading for paragraphs, proofs, diagrams, fill-in-the-blank, true or false, and more. It also allows for the grading of programming assignments, either automatically or manually. Gradescope is primarily used in high school and higher-ed courses in subjects such as Math, Chemistry, Computer Science, Physics, Economics, and Business, which does not cater to university level. In addition to the limitations, Gradescope does not provide lexical analysis, and when an instructor highlights text and provides feedback, it will not generate historical feedback for previous submissions [6].

Figure 2 Screenshot of gradescope



How can we help?

Top Articles

Submitting a Regrade Request

Code Similarity

Submitting an Assignment

★ Featured Articles

Resetting Your Password
If you have forgotten your password or are otherwise unable to log in, first go to the Gradescope website. Click the Log In button (top right corner). When the login dialog box appears, click Forgot...
UPDATED 1 YEAR AGO

Assignment Types
Gradescope allows you to grade paper-based exams, quizzes, bubble sheets, and homework. In addition, Gradescope enables you to grade programming assignments (graded automatically or manually) and let...
UPDATED 5 MONTHS AGO

Grading Submissions
Want to filter your grading by section? Check out our guidance to learn how. To begin grading, go to the Grade Submissions tab and click on the question that you would like to grade. There are three...
UPDATED 9 SECONDS AGO

Gradescope Mobile App
Gradescope Mobile App. The Gradescope mobile app allows students to submit to their assignments, and view their grades using their mobile device. The mobile app is currently only available for users...
UPDATED 3 MONTHS AGO

Submitting an Assignment
This student guide shows you how to submit a PDF, a series of images, code files, and bubble sheets, and how to work on Online Assignments with and without the LockDown Browser exam security measures enabled.
UPDATED 2 MONTHS AGO

All Articles

Account Management
All the tools and tips you need to manage your personal or institutional account.
5 ARTICLES

Course Workflow
Add students, manage assignments, and export grades.
8 ARTICLES

Assignment Workflow
Assess students in a variety of ways and get detailed insights into what your students know.
20 ARTICLES

Student Center
Guides, tips, and tricks for students.
9 ARTICLES

Remote Assessment FAQs
Your guide to conducting remote assessments with Gradescope
15 ARTICLES

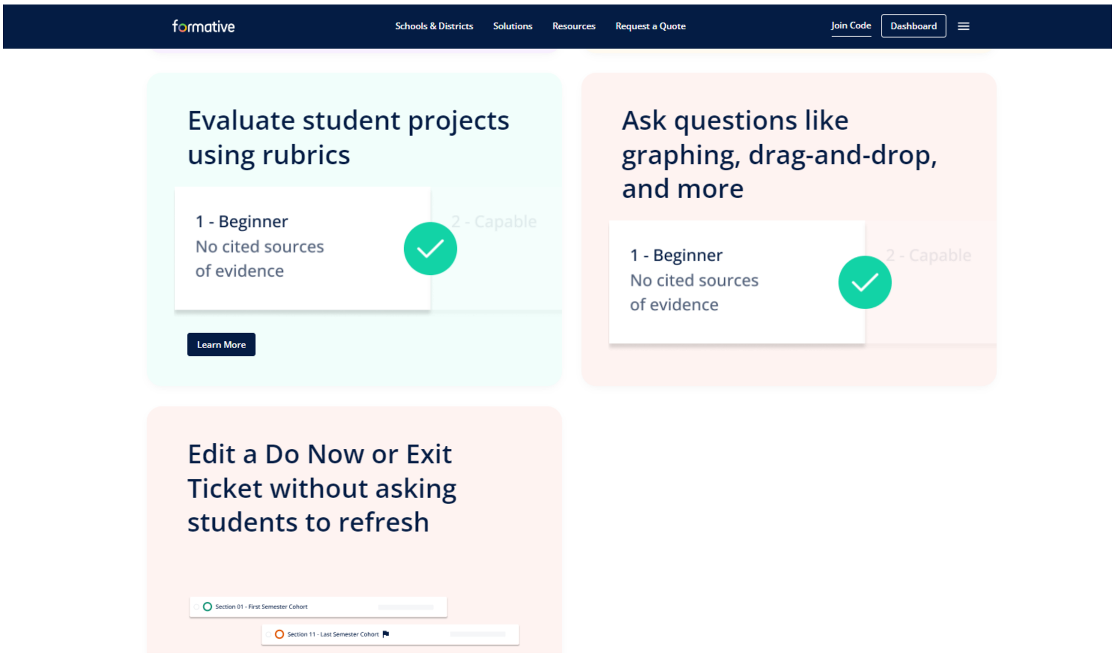
FAQs
Common questions that we get from instructors and students (and everyone in between) about using Gradescope.
34 ARTICLES

LMS Workflow
Administrator, instructor, and student guidance for using Gradescope with Blackboard, Brightspace (D2L), Canvas, Moodle, and Sakai.
21 ARTICLES

2.2..3 Formative.ai

Formative.ai is an emerging artificial intelligence technology that is integrated into the Formative platform, which is a website designed to help educators create digital assignments, monitor learners' work and engagement in real-time, and provide feedback. The strengths of Formative AI include its potential to automate content creation and provide real-time feedback. However, it does not allow the instructor to provide specific feedback in the file directly. Instructor needs to download the file to provide feedback and re-upload the file for students to view. On top of the limitations, the system does not have the function to generate feedback suggestions from the historical feedback based on the text highlighted by the instructor [7].

Figure 3 Screenshot of Formative.AI



Evaluate student projects using rubrics

1 - Beginner
No cited sources of evidence

2 - Capable

Ask questions like graphing, drag-and-drop, and more

1 - Beginner
No cited sources of evidence

2 - Capable

Edit a Do Now or Exit Ticket without asking students to refresh

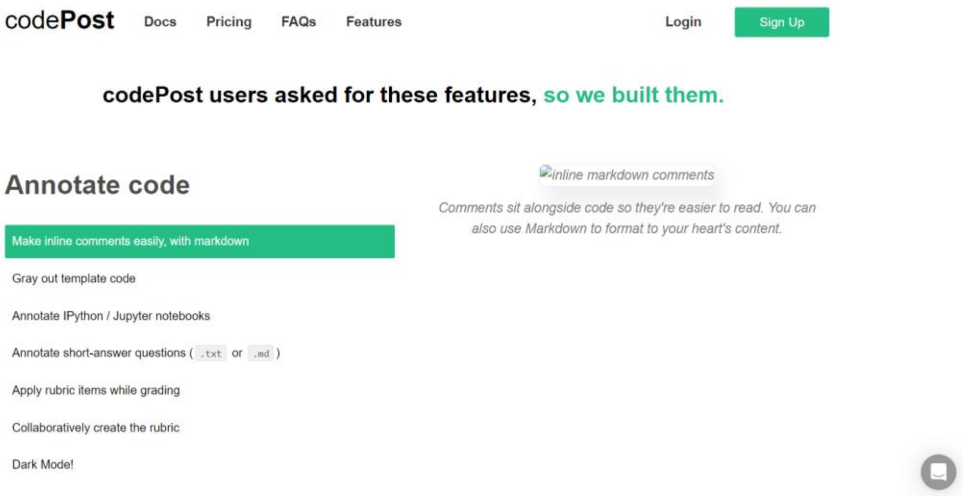
Section 01 - First Semester Cohort

Section 11 - Last Semester Cohort

2.2..4 CodePost

CodePost is an online platform designed to help computer science instructors grade and provide feedback on student code. It allows instructors to create assignments, set up grading rubrics, and provide feedback on student code. CodePost also includes features such as plagiarism detection and code auto grading. One of the strengths of CodePost is its ability to handle large classes and complex assignments. However, it is more focused on programming assignments and does not support lexical analysis. Other than that, it does not have the automatic generation of historical feedback feature as well [8].

Figure 4 Screenshot of codePost



codePost users asked for these features, so we built them.

Annotate code

- Make inline comments easily, with markdown
- Gray out template code
- Annotate IPython / Jupyter notebooks
- Annotate short-answer questions (`.txt` or `.md`)
- Apply rubric items while grading
- Collaboratively create the rubric
- Dark Mode!

inline markdown comments

Comments sit alongside code so they're easier to read. You can also use Markdown to format to your heart's content.

2.2.2. Competitor Analysis

Table 1 Competitor Analysis

Competitors	SemakLah	Feedbackfruit	Gradescope	Formative.AI	codePost
Features					
Create Classroom	Yes	Yes	Yes	Yes	Yes
Manage Assignment	Yes	Yes	Yes	Yes	Yes
Set Assignment Attributes	Yes	Yes	Yes	Yes	Yes
Support PDF and Docx Submission	Yes	Yes	No	Yes	No
Customizable criteria of rubrics	Yes	Yes	Yes	Yes	Yes
Generate Feedback Suggestion Based on Historical Feedback	Yes	No	No	No	No
Configurable grading	Yes	Yes	Yes	Yes	Yes
Provide Lexical Analysis	Yes	No	No	Yes	No
Monitor Grading Progress	Yes	Yes	Yes	Yes	Yes

2.2.3. Existing Algorithms

This section will discuss the existing algorithms that have been applied to lexical analysis for language and grammatical quality assessment, text-matching solutions.

2.2..1 Existing Algorithm for Lexical Analysis for Language and Grammatical Quality Assessment

Golding and Schabes [9] introduced a hybrid approach called 'Tribayes' that combines the trigram and Bayesian methods, which proved to be effective in correcting real-word spelling errors. The trigram model, a second-order Markov process for language modeling, encodes the context using part-of-speech trigrams to identify the correct word from a confusion set of real words generated by single edit operations. The Bayesian approach is a feature-based method that uses context words and collocations to determine the probability that a word is correct. The 'Tribayes' approach used in the spelling mistake correction (SMC) model for real-word error correction involves the use of contextual information and statistical language modeling. The trigram model captures the contextual information using part-of-speech trigrams, while the Bayesian approach incorporates feature-based methods to improve the accuracy of real-word error correction in the SMC model.

2.2..2 Existing Algorithm for Text-Matching Solution

Fuzzy string matching is a technique for finding strings that match a pattern approximately. Li demonstrated the application of the FuzzyWuzzy library in Python, which used the Levenshtein Distance to calculate the differences between strings. The article provides examples of how fuzzy string matching can be used, such as spell checking, spelling-error correction, and checking for duplicate records. It also explains the importance of fuzzy string matching in scenarios where exact string matching is not sufficient, such as when dealing with misspelled words or partial input. The FuzzyWuzzy library in Python is used to perform fuzzy string matching by calculating the differences between strings, allowing for approximate matching of strings even in the presence of misspellings or partial input. The article provided examples of using the FuzzyWuzzy library to compare strings and demonstrates its application in scenarios such as spell checking and duplicate record detection [10].

2.2..3 Existing Feedback Generation Method

The study compares three feedback comment generation methods: retrieval-based, simple generation, and retrieve-and-edit. The retrieval-based method encodes the input with its context and predicts the feedback comment with attention and copy mechanisms. It uses independent networks called retriever and editor. The simple generation method directly generates feedback comments without using retrieval-based techniques. It involves encoding the input with its context and predicting the feedback comment using a neural network. On the other hand, the retrieve-and-edit method is a hybrid of retrieval-based and simple generation. It first retrieves candidate feedback comments from a retrieval-based system and then edits them using a neural network to generate the final feedback comment. The study found that the performance order is not as expected, with different methods outperforming others in specific types of feedback comment generation. For instance, simple generation outperforms the other two methods in preposition feedback comment generation, while retrieval-based exhibits the best performance in general feedback comments [11].

3 SYSTEM REQUIREMENTS AND DESIGN

3.1. Project Scope

The proposed project is a web application tailored for both instructors and students at the university level, specifically designed to streamline the process of managing and evaluating technical written assignments. This comprehensive platform aims to enhance the efficiency and effectiveness of academic interactions by providing a centralized hub for instructors to assign assignments, offer feedback, and grade submissions. Simultaneously, students will benefit from a user-friendly interface that facilitates the submission of assignments, access to assignment rubrics, and feedback from instructors. Overall, the project endeavors to create a robust digital environment that fosters collaboration, communication, and academic excellence within the university setting.

3.2. System Capabilities

1. Classroom Management:
   a. Creation of classrooms.
   b. Manage student enrolment.

2. Assignment Management:
   a. Instructor manages assignments.
   b. Students can download assignments.
   c. Support for file uploads in PDF or Word format.
   d. Efficient organization and tracking of assignments for each class.

3. Feedback Generation:
   a. Natural Language Processing (NLP) techniques for extracting text and performing lexical analysis that shows the overall language and grammatical quality of writing assignments.
   b. Instructors can create feedback for students.
   c. Historical feedback data is used to generate suggestions for feedback based on similar text.
   d. Flexibility for instructors to accept, edit, or overwrite suggested feedback.

4. Grading:
   a. Instructors can assign grades to student assignments.
   b. Efficient grading process facilitated by the system.
   c. Instructors can return graded assignments.
   d. Students can access their feedback and grades through the system.

3.3. System Limitations

1. Instructor limitation in a classroom and assignment:
    a. The system only allows one instructor in one classroom.
    b. The system only allows one instructor to create an assignment, it does not include collaboration between instructors.

2. Student-student interaction is limited:
    a. The student will not be able to interact with one another in any form in using this system.

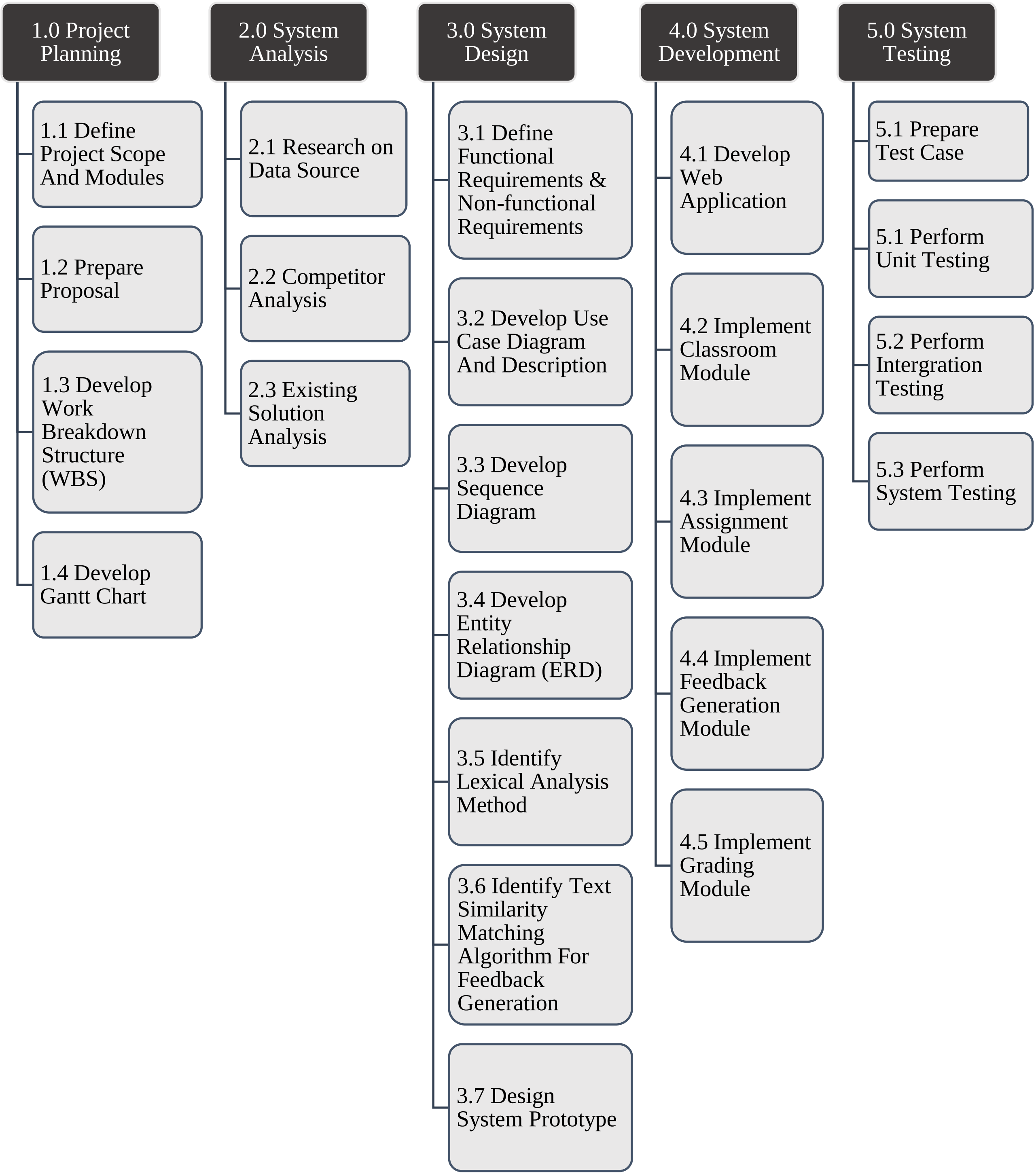
3. Language is limited:
    a. Lexical analysis is limited to only English language as NLP tools can only support English.

4. The structure of grading rubrics is fixed:
    a. The system will have one type of grading rubrics available for all instructors.

3.4. Project Management

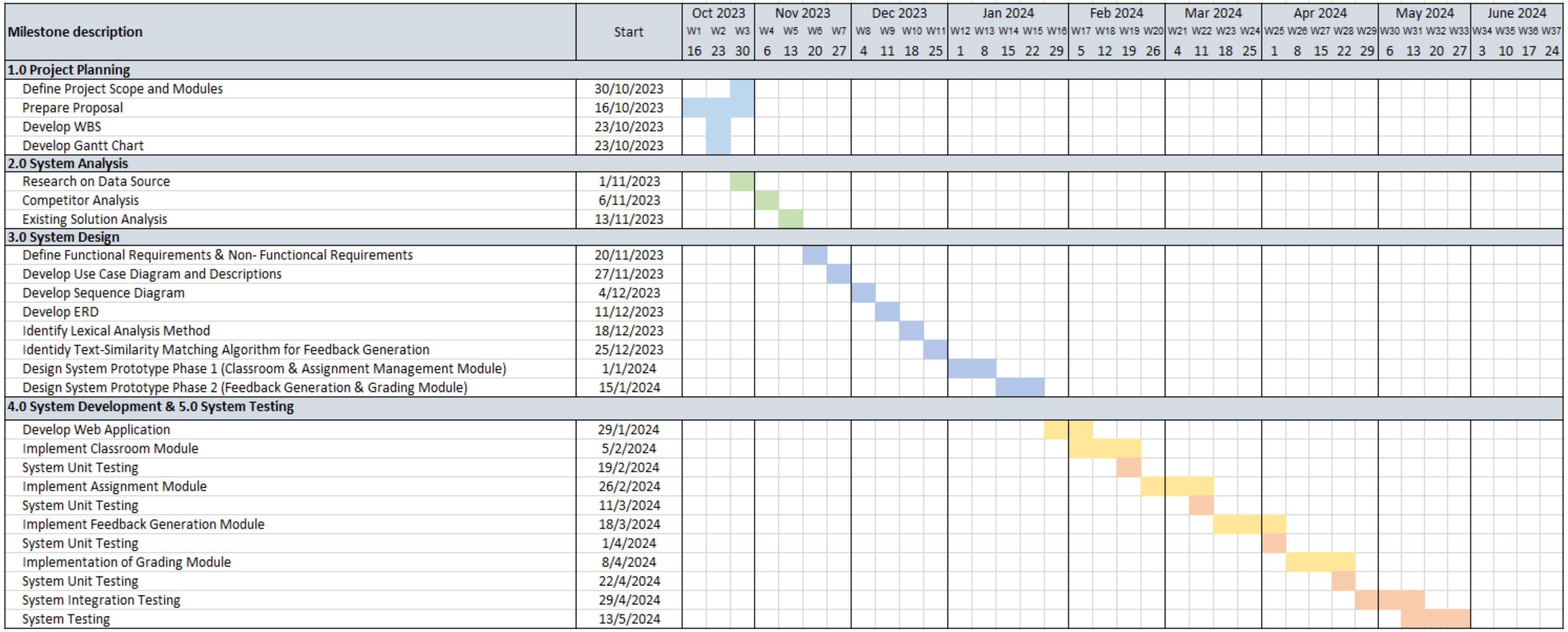
3.4.1. Work Breakdown Structure (WBS)

Table 2 Work Breakdown Structure (WBS)


1.0 Project Planning
1.1 Define Project Scope And Modules
1.2 Prepare Proposal
1.3 Develop Work Breakdown Structure (WBS)
1.4 Develop Gantt Chart
2.0 System Analysis
2.1 Research on Data Source
2.2 Competitor Analysis
2.3 Existing Solution Analysis
3.0 System Design
3.1 Define Functional Requirements & Non-functional Requirements
3.2 Develop Use Case Diagram And Description
3.3 Develop Sequence Diagram
3.4 Develop Entity Relationship Diagram (ERD)
3.5 Identify Lexical Analysis Method
3.6 Identify Text Similarity Matching Algorithm For Feedback Generation
3.7 Design System Prototype
4.0 System Development
4.1 Develop Web Application
4.2 Implement Classroom Module
4.3 Implement Assignment Module
4.4 Implement Feedback Generation Module
4.5 Implement Grading Module
5.0 System Testing
5.1 Prepare Test Case
5.1 Perform Unit Testing
5.2 Perform Intergration Testing
5.3 Perform System Testing

3.4.2. Gantt Chart

Figure 5 Gantt Chart



Milestone description	Start
1.0 Project Planning	
Define Project Scope and Modules	30/10/2023
Prepare Proposal	16/10/2023
Develop WBS	23/10/2023
Develop Gantt Chart	23/10/2023
2.0 System Analysis	
Research on Data Source	1/11/2023
Competitor Analysis	6/11/2023
Existing Solution Analysis	13/11/2023
3.0 System Design	
Define Functional Requirements & Non- Functioncal Requirements	20/11/2023
Develop Use Case Diagram and Descriptions	27/11/2023
Develop Sequence Diagram	4/12/2023
Develop ERD	11/12/2023
Identify Lexical Analysis Method	18/12/2023
Identidy Text-Similarity Matching Algorithm for Feedback Generation	25/12/2023
Design System Prototype Phase 1 (Classroom & Assignment Management Module)	1/1/2024
Design System Prototype Phase 2 (Feedback Generation & Grading Module)	15/1/2024
4.0 System Development & 5.0 System Testing	
Develop Web Application	29/1/2024
Implement Classroom Module	5/2/2024
System Unit Testing	19/2/2024
Implement Assignment Module	26/2/2024
System Unit Testing	11/3/2024
Implement Feedback Generation Module	18/3/2024
System Unit Testing	1/4/2024
Implementation of Grading Module	8/4/2024
System Unit Testing	22/4/2024
System Integration Testing	29/4/2024
System Testing	13/5/2024

3.4.3. SWOT Analysis

Table 3 SWOT Analysis

Strength	Weaknesses
- Comprehensive tools for instructors to manage classrooms, assignments, and student submission efficiently. - Feedback generation is based on historical data, facilitating consistency and efficiency in providing constructive feedback. - Provide insights into spelling and grammatical error	- The system heavily relies on consistent internet connectivity, which may pose challenges for users in areas with limited internet access. - Automated feedback suggestions may produce false positives or negatives, requiring careful review by instructors. - The system relies on the instructor's input for creating feedback and grading, and the quality of suggestions is contingent on the historical data available.
Opportunity	**Threats**
- Adoption of online learning environments since covid-19. - The maturity of certain AI technologies, such as natural language processing, presents an opportunity to enhance the system’s text extraction and analysis capabilities. - The demand for customizable and adaptive online learning environment	- Instructor and students may resist adopting the new system, especially if they are accustomed to existing methods and tools. - Data security and privacy concerns related to text extraction and student information. - Competition from other educational platforms offering similar features.

3.5. Development Methodology

The proposed semi-automatic grading web application adopts an agile development methodology that aligns with the project needs for continuous adaptation to evolve requirements through iterative development cycles. Agile methodology can be applied to all individual modules.

In the classroom management module, the agile methodology will be applied through short sprints. The focus of Sprint 1 is on user authentication and account registration, while sprint 2 is dedicated to classroom creation and student enrolment management.

In assignment management module, sprint 3 might focus on assignment management which includes creation, viewing, updating, deleting, while sprint 4 will handle the submission management. This approach ensures that each functionality is developed incrementally and can be refined based on user feedback at the end of each sprint.

In the feedback generation module, sprint 5 and sprint 6 will focus on text extraction and lexical analysis respectively. The iterative development allows for early testing and refinement, ensuring that the system meets the instructor's feedback needs effectively.

Finally, each functionality within the grading module can be addressed in separate sprints. Sprint 7 might involve the development of grading rubric management, while Sprint 8 could focus on the implementation of score entry and viewing functionalities. This approach allows for regular validation and adjustments based on instructor feedback.

By breaking down the development process into manageable sprints, each lasting for a week, the project can regularly reassess its goals, accommodate changes, and receive ongoing feedback.

3.6. User Requirements

3.6.1. Requirement Gathering

The requirement gathering process for the proposed system was informed by survey of existing feedback and grading systems, together with collaborative brainstorming sessions with my supervisor. The survey involved a thorough observation of similar platforms, identifying their features, functionalities, and areas for improvement. Additionally, engaging in brainstorming sessions with my supervisor provided valuable insights into the specific needs and challenges faced by instructors in the context of grading and providing feedback.

3.6.2. Functional Requirements

Classroom Management Module

1. Instructor and student shall log in to their respective accounts.
2. Instructor and student shall register for an account if they are new users.
3. Instructor shall manage online classrooms with different courses.
4. Instructor shall manage student enrolment to the online classroom.

Assignment Management Module

1. Instructor shall create assignments or edit existing assignments by managing assignments.
2. Instructor shall set assignment attributes such as title, description, instructions, assignment start date, assignment end date, and upload the assignment question in PDF and Word format.
3. Instructor shall manage the list of student submissions.
4. Student shall download the assignments.
5. Student shall manage their assignment submission by uploading the assignment file in PDF or Word format, editing, or reviewing the submitted assignment.

Feedback Generation Module

1. System shall extract digital text from the assignment files.
2. System shall perform lexical analysis which includes analyzing the number of spelling errors and number of grammatical errors.
3. System shall perform similarity matching between the highlighted text and feedback history (i.e., text units in prior assignments with feedback).
4. Instructor will receive top 3 feedback suggestions based on feedback history when similar text is highlighted at sentence level.
5. Instructor shall be able to review the summary of lexical analysis.
6. Instructor shall highlight text to provide feedback.
7. Instructor shall select on the feedback suggestions and make edits for the feedback.

Grading Module

1. Instructor shall manage grading rubrics with different weightage.
2. Instructor shall enter scores according to the grading rubrics for each assignment submitted by the student.
3. Instructor shall view the generated final scores and grade for each student assignment.
4. Instructor shall review the dashboard of grading progress.
5. Instructor shall return the feedback and grades for students to review.
6. Student shall review the feedback and grades given by the instructors when the instructor returns the feedback and grading.

3.6.3. Non-Functional Requirements

1. Performance:

- The system should provide a response time of less than 3 seconds for user interactions (e.g., logging in, creating assignments, submitting assignments).
- The system should be able to handle concurrent logins and interactions from at least 1000 users without significant performance degradation.

2. Scalability:

- The system should be designed to accommodate a minimum of 5000 users and 100 classrooms initially, with the ability to scale easily to support growth.
- The system must be scalable enough to support 500,000 visits at the same time while maintaining optimal performance.

3. Portability:

- The system should work well on Windows 10 and Windows 11 without any change in its behavior and performance.

4. Availability:

- The system should be available to Malaysia users 99.9% of the time every month.
- Planned maintenance should be scheduled during off-peak hours to minimize impact on users.

5. Security:

- User authentication and authorization should be secured.
- The system should have role-based access control to ensure that only authorized users can perform specific actions.

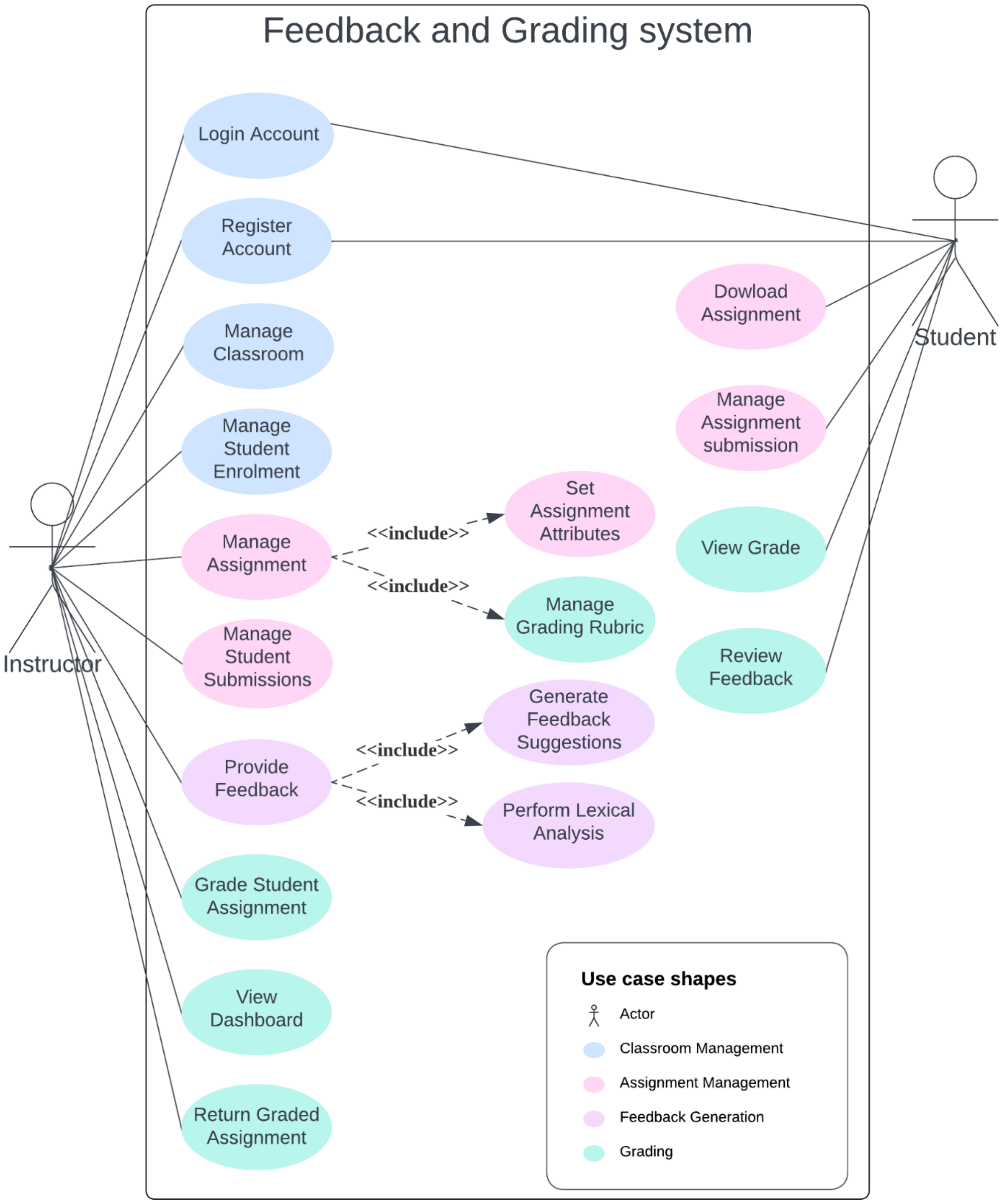
6. Localization:

- The date format must be as follows: dd/mm/yyyy

3.7. System Analysis

3.7.1. Use Case Diagram

Figure 6 Use Case Diagram



3.7.2. Use Case Descriptions

UC001: Login Account

Table 4 Use Case Description for UC001 Login Account

Use Case Name:	Login Account	
Scenario:	A registered user attempts to access the system.	
Triggering Event:	User opens SemakLah website and initiates the login process.	
Brief Description:	Users (Instructor and student) log in to their respective accounts.	
Actors:	Instructor and Student	
Related use cases:	N/A	
Stakeholders:	N/A	
Preconditions:	Users have valid credentials.	
Postconditions:	User successfully logged into the system.	
Flow of activities:	**Actor**	**System**
	• User navigates to the login page. • User enters their username and password. • If credentials are valid, user is redirected to the dashboard.	• Verifies credentials. • Grants access if authentication is successful. • Displays an error message if authentication fails.
Exception Condition:	Invalid credentials: - User receives an error message and is prompted to re-enter credentials.	

UC002: Register Account

Table 5 Use Case Description for UC002 Register Account

Use Case Name:	Register Account	
Scenario:	A new user creates an account.	
Triggering Event:	User navigates to SemakLah registration page.	
Brief Description:	Users (Instructor and Student) create accounts to gain access to the system.	
Actors:	Instructor and Student	
Related use cases:	N/A	
Stakeholders:	N/A	
Preconditions:	User does not have an account.	
Postconditions:	User account is successfully registered.	
Flow of activities:	**Actor**	**System**
	• Provides required registration information. (e.g. first name, last name, gender, gmail and password)	• Validates information. • Creates a new account if validation is successful. • Displays an error message if validation fails.
Exception Condition:	Duplicate Gmail Account - System displays an error message, prompting the user to choose a different Gmail.	

UC003: Manage Classroom

Table 6 Use Case Description for UC003 Manage Classroom

Use Case Name:	Manage Classroom	
Scenario:	Instructor oversees online classroom.	
Triggering Event:	Instructor selects the “Manage Classroom” option.	
Brief Description:	Instructor manages online classrooms based on different courses.	
Actors:	Instructor	
Related use cases:	N/A	
Stakeholders:	N/A	
Preconditions:	Instructor is logged into the system.	
Postconditions:	Instructor successfully manages the selected classroom.	
Flow of activities:	**Actor**	**System**
	• Selects the “Manage classroom” option.	• Display a list of actions including create, view, update, and remove.
	• Create classroom. • Provide required information. (e.g. Course Code, Course Description, Classroom limit) • Save	• Validates information. • Creates a new classroom if validation is successful. • Displays an error message if validation fails.
	• View classroom details.	• Display classroom details
	• Update classroom details. • Update required information (e.g. Course Code, Course Description, Classroom limit) • Save	• Validates information. • Update classroom information if validation is successful. • Displays an error message if validation fails.
	• Remove classroom.	• Delete classroom
Exception Condition:	N/A	

UC004: Manage Student Enrolment

Table 7 Use Case Description for UC004 Manage Student Enrolment

Use Case Name:	Manage Student Enrolment	
Scenario:	Instructor add or remove student enrolment in an online classroom.	
Triggering Event:	Instructor selects the "Manage Student Enrolment" option for a specific classroom	
Brief Description:	Instructor manages student enrolment for a selected online classroom	
Actors:	Instructor and Student	
Related use cases:	N/A	
Stakeholders:	N/A	
Preconditions:	1. Instructor is logged into the system. 2. Classroom exists. 3. Student exists.	
Postconditions:	Instructor successfully manages student enrolment for the selected classroom.	
Flow of activities:	**Actor**	**System**
	• Select the "Manage Student Enrolment" option	• Display the list of students in the online classroom
	• Add Student • Save	• Updates the list of students in the online classroom.
	• Remove Student • Save	• Updates the list of students in the online classroom.
Exception Condition:	Exceed the class limit: - System displays a message indicating that the class limit is exceeded. Student does not have an account: - System displays a message indicating that student not found.	

UC005: Manage Assignment

Table 8 Use Case Description for UC005 Manage Assignment

Use Case Name:	Manage Assignment	
Scenario:	Instructor creates, edits, or deletes assignments.	
Triggering Event:	Instructor selects the "Manage Assignment" option for a specific classroom.	
Brief Description:	Instructor manages assignments for a selected classroom.	
Actors:	Instructor	
Related use cases:	UC006: Set Assignment Attributes UC007: Manage Grading Rubrics	
Stakeholders:	N/A	
Preconditions:	1. Instructor is logged into the system. 2. A classroom exists.	
Postconditions:	Instructor successfully manages assignments for the selected classroom.	
Flow of activities:	**Actor**	**System**
	• Select "Manage Assignment"	• Display the list of assignments
	• Create Assignment	• Prompt user to set assignment attributes
	• Edit Assignment	• Prompt user to update assignment attributes
	• Remove Assignment	• Delete Assignment
Exception Condition:	N/A	

UC006: Set Assignment Attributes

Table 9 Use Case Description for UC006 Set Assignment Attributes

Use Case Name:	Set Assignment Attributes	
Scenario:	Instructor sets attributes for a specific assignment.	
Triggering Event:	Instructor selects on each section of the assignment attributes.	
Brief Description:	When managing assignment, instructor can set assignment attributes such as title, description, instructions, assignment start date, assignment end date, and upload the assignment question in PDF and Word format.	
Actors:	Instructor	
Related use cases:	UC005: Manage Assignment	
Stakeholders:	N/A	
Preconditions:	1. Instructor is logged into the system. 2. Instructor is managing assignment in a classroom.	
Postconditions:	Assignment attributes are successfully set	
Flow of activities:	**Actor**	**System**
	• Fills in each section of the assignment attribute such as title, description, instructions, deadline, rubrics and upload the assignment question in PDF and Word format.	• Validates and saves the assignment attributes.
Exception Condition:	Invalid file format: - System displays an error message and prompts the instructor to upload a valid assignment file.	

UC007: Manage Grading Rubric

Table 10 Use Case Description for UC007 Manage Grading Rubric

Use Case Name:	Manage Grading Rubric	
Scenario:	Instructor creates, edits, or deletes grading rubrics with different criteria.	
Triggering Event:	Instructor selects the “Managing Grading Rubric” option for a specific assignment.	
Brief Description:	Instructor manages grading rubric for assignments.	
Actors:	Instructor	
Related use cases:	UC005: Manage Assignment	
Stakeholders:	N/A	
Preconditions:	Instructor is logged into the system. Instructor is managing assignment in a classroom.	
Postconditions:	Instructor successfully manages the grading rubric for the selected assignment.	
Flow of activities:	**Actor**	**System**
	• Select “Manage Grading Rubric” option	• Display a list of actions including create, view, update, and delete.
	• Creates grading rubric. • Save	• Creates a new grading rubric.
	• Add new criteria. • Save	• Update grading rubric
	• Remove existing criteria. • Save	• Update grading rubric
	• Update existing criteria. • Save	• Update grading rubric
	• View grading rubric	• Display grading rubric
	• Delete grading rubric	• Delete grading rubric
Exception Condition:	N/A	

UC008: Manage Student Submissions

Table 11 Use Case Description for UC008 Manage Student Submissions

Use Case Name:	Manage Student Submissions	
Scenario:	Instructor views, modifies student submission for a specific assignment.	
Triggering Event:	Instructor selects the "Manage Student Submissions" option for a specific assignment.	
Brief Description:	Instructors manage student submissions for a selected assignment.	
Actors:	Instructor	
Related use cases	N/A	
Stakeholders:	N/A	
Preconditions:	1. Instructor is logged into the system. 2. A submission exists.	
Postconditions:	Instructor successfully manages student submissions for the selected assignment.	
Flow of activities:	**Actor**	**System**
	• Select the assignment to manage student submissions	• Displays the list of student submissions
	• Download student submissions	• Download a copy of student submission in local device
	• Modify student submission save	• Update the list of student submissions
Exception Condition:	N/A	

UC009: Download Assignment

Table 12 Use Case Description for UC009 Download Assignment

Use Case Name:	Download Assignment	
Scenario:	Students download assignment questions.	
Triggering Event:	Student selects the "Download Assignment" option.	
Brief Description:	Student downloads assignment.	
Actors:	Student	
Related use cases	N/A	
Stakeholders:	N/A	
Preconditions:	1. Student is logged into the system. 2. Assignment in a classroom is available for download.	
Postconditions:	Assignment is successfully downloaded to their local device.	
Flow of activities:	**Actor**	**System**
	• Selects the assignment to download	• System retrieves the assignment file and download into local device
Exception Condition:	N/A	

UC010: Manage Assignment Submission

Table 13 Use Case Description for UC010 Manage Assignment Submission

Use Case Name:	Manage Assignment Submission	
Scenario:	Students uploads, edits, or review their assignment submission.	
Triggering Event:	Student selects the "Manage Assignment Submission" option.	
Brief Description:	Student manages their assignment submission.	
Actors:	Student	
Related use cases	N/A	
Stakeholders:	N/A	
Preconditions:	1. Student is logged into the system. 2. Assignment submission phase is active.	
Postconditions:	Student successfully manages their assignment submission.	
Flow of activities:	**Actor**	**System**
	• Upload assignment file. • Save	• Update the assignment submission
	• Edit assignment file. • Save	• Overwrite the assignment submission
	• Delete assignment file. • Save	• Delete the assignment submission
Exception Condition:	N/A	

UC011: View Grade

Table 14 Use Case Description for UC011 View Grade

Use Case Name:	View Grade	
Scenario:	Student views the grade for their assignment after the instructor returns the grading.	
Triggering Event:	Student selects the specific assignment.	
Brief Description:	Student views their grades.	
Actors:	Student	
Related use cases	N/A	
Stakeholders:	N/A	
Preconditions:	1. Student is logged into the system. 2. Grades are available for viewing.	
Postconditions:	Student successfully views their grades.	
Flow of activities:	**Actor**	**System**
	• Selects the specific assignment	• Displays the submission and corresponding grades.
Exception Condition:	N/A	

UC012: Review Feedback

Table 15 Use Case Description for UC012 Review Feedback

<table>
<tr><td><b>Use Case Name:</b></td><td colspan="2">Review Feedback</td></tr>
<tr><td><b>Scenario:</b></td><td colspan="2">Student reviews the feedback for their assignment after instructor returns the feedback.</td></tr>
<tr><td><b>Triggering Event:</b></td><td colspan="2">Student selects the specific assignment.</td></tr>
<tr><td><b>Brief Description:</b></td><td colspan="2">Students reviews feedback given by instructors.</td></tr>
<tr><td><b>Actors:</b></td><td colspan="2">Student</td></tr>
<tr><td><b>Related use cases</b></td><td colspan="2">N/A</td></tr>
<tr><td><b>Stakeholders:</b></td><td colspan="2">N/A</td></tr>
<tr><td><b>Preconditions:</b></td><td colspan="2">1. Student is logged into the system.
2. Feedback is available for reviewing.</td></tr>
<tr><td><b>Postconditions:</b></td><td colspan="2">Student can see individual assignment feedback.</td></tr>
<tr><td rowspan="2"><b>Flow of activities:</b></td><td><b>Actor</b></td><td><b>System</b></td></tr>
<tr><td>• Selects the specific assignment</td><td>• Displays the submission and respective feedback.</td></tr>
<tr><td><b>Exception Condition:</b></td><td colspan="2">N/A</td></tr>
</table>

UC013: Provide Feedback

Table 16 Use Case Description for UC013 Provide Feedback

Use Case Name:	Provide Feedback	
Scenario:	Instructor provides feedback on student assignment.	
Triggering Event:	Instructor selects the "Provide Feedback" option.	
Brief Description:	The instructor highlights specific text and provides feedback on a student assignment.	
Actors:	Instructor	
Related use cases	UC014: Generate Feedback Suggestions UC015: Perform Lexical Analysis	
Stakeholders:	N/A	
Preconditions:	1. Instructor is logged into the system. 2. Student submission exists.	
Postconditions:	Feedback is successfully provided	
Flow of activities:	**Actor**	**System**
	• Highlights text to provide feedback.	• Updates the feedback for the assignment.
	• Edits the feedback suggestion	• Updates the feedback for the assignment.
	• Accept the feedback suggestions.	• Updates the feedback for the assignment.
Exception Condition:	N/A	

UC014: Generate Feedback Suggestions

Table 17 Use Case Description for UC014 Generate Feedback Suggestions

Use Case Name:	Generate Feedback Suggestions	
Scenario:	The system generates feedback suggestions for the instructor based on the historical feedback.	
Triggering Event:	Instructor highlights on a text.	
Brief Description:	Instructors receive feedback suggestions based on history when similar text is highlighted.	
Actors:	Instructor	
Related use cases	UC013: Provide Feedback	
Stakeholders:	N/A	
Preconditions:	1. Instructor is logged into the system. 2. Instructor is providing feedback on a specific assignment.	
Postconditions:	Instructor can view and use generated feedback suggestions.	
Flow of activities:	**Actor**	**System**
	• Highlight a text	• Extract the text. • Perform similarity matching with the historical text. • Generate a feedback suggestion.
Exception Condition:	No feedback history: - System will not generate any feedback suggestions	

UC015: Perform Lexical Analysis

Table 18 Use Case Description for UC015 Perform Lexical Analysis

Use Case Name:	Perform Lexical Analysis	
Scenario:	The system performs lexical analysis on assignment files.	
Triggering Event:	Instructors review the lexical analysis	
Brief Description:	System extracts text from assignments and performs lexical analysis (number of spelling and grammatical errors).	
Actors:	Instructor	
Related use cases	UC013: Provide Feedback	
Stakeholders:	N/A	
Preconditions:	The system must have access to the assignment files.	
Postconditions:	Instructor can view the statistic of lexical analysis for each student	
Flow of activities:	**Actor**	**System**
	• Selects a student assignment.	• Loads the student assignment. • Perform lexical analysis. • Display the result.
Exception Condition:	N/A	

UC016: Grade Student Assignment

Table 19 Use Case Description for UC016 Grade Student Assignment

Use Case Name:	Grade Student Assignment	
Scenario:	Instructor grades student assignment	
Triggering Event:	Instructor selects the "Grade Student Assignment" option.	
Brief Description:	Instructor enters scores according to grading rubrics.	
Actors:	Instructor	
Related use cases	N/A	
Stakeholders:	N/A	
Preconditions:	1. Instructor is logged into the system. 2. Grading rubric is available for the assignment. 3. Student submission exists.	
Postconditions:	Assignment is successfully graded.	
Flow of activities:	**Actor**	**System**
	• Enters scores according to the grading rubrics for each assignment.	• Updates the scores for the assignment.
Exception Condition:	N/A	

UC017: View Dashboard

Table 20 Use Case Description for UC017 View Dashboard

Use Case Name:	View Dashboard	
Scenario:	Instructor wants to view the dashboard of grading process.	
Triggering Event:	Instructor selects the "View Dashboard" option.	
Brief Description:	Instructors may review the dashboard of grading progress.	
Actors:	Instructor	
Related use cases	N/A	
Stakeholders:	N/A	
Preconditions:	Instructor is logged into the system.	
Postconditions:	Dashboard is successfully displayed.	
Flow of activities:	**Actor**	**System**
	• Select assignment	• Displays the grading progress dashboard including the number of assignments graded, pending assignments, and overall progress.
Exception Condition:	N/A	

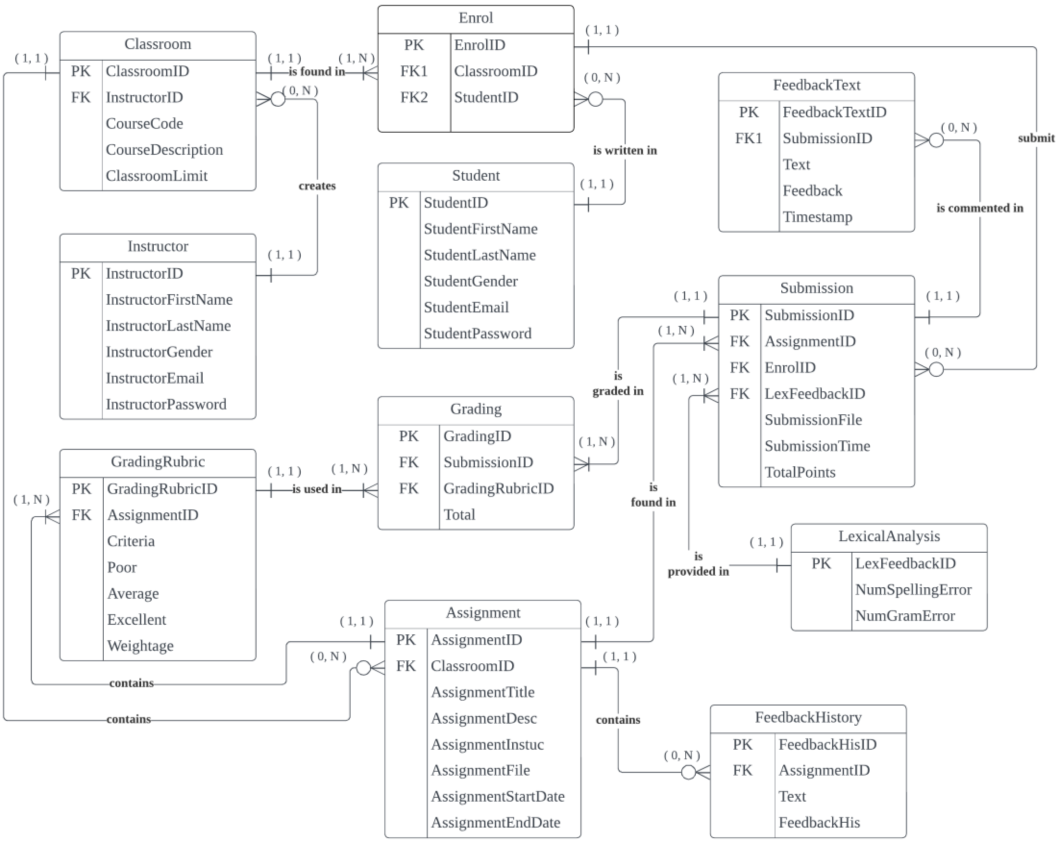
UC018: Return Graded Assignment

Table 21 Use Case Description for UC018 Return Graded Assignment

Use Case Name:	Return Graded Assignment	
Scenario:	Instructor wants to return the feedback and grades students to review.	
Triggering Event:	Instructor selects the "Return Graded Assignment" option.	
Brief Description:	Instructor returns the feedback and grades for students to review once instructor is done with all grading.	
Actors:	Instructor	
Related use cases	N/A	
Stakeholders:	N/A	
Preconditions:	1. Instructor is logged into the system. 2. Assignments are graded.	
Postconditions:	Graded assignments are successfully made available to students.	
Flow of activities:	**Actor**	**System**
	• Select on the return button	• Enable all students in the classroom to view the feedback and grade of their assignment.
Exception Condition:	N/A	

3.7.3. Entity Relationship Diagram (ERD)

Figure 7 Entity Relationship Diagram (ERD)



3.7.4. Sequence Diagram

Figure 8 Sequence Diagram for Login Account

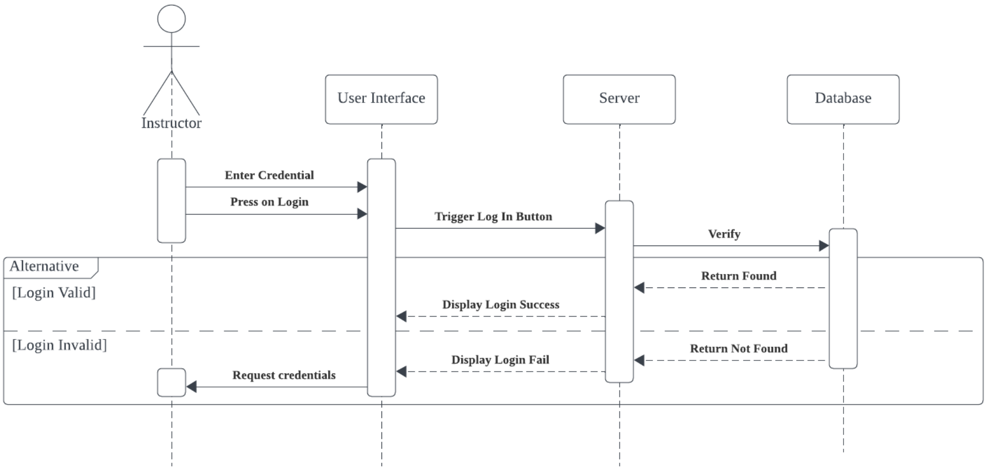


Figure 9 Sequence Diagram for Register Account

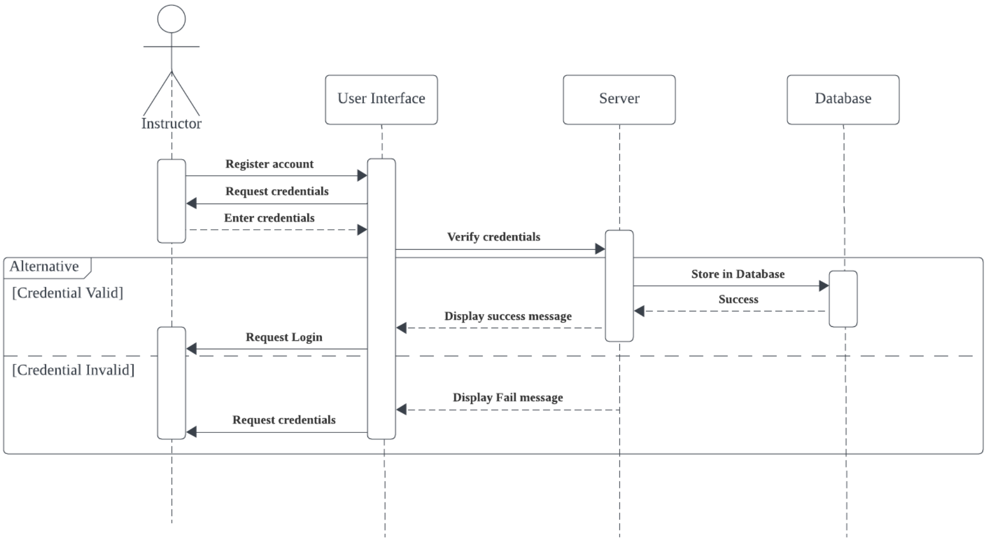


Figure 10 Sequence Diagram for Manage Classroom

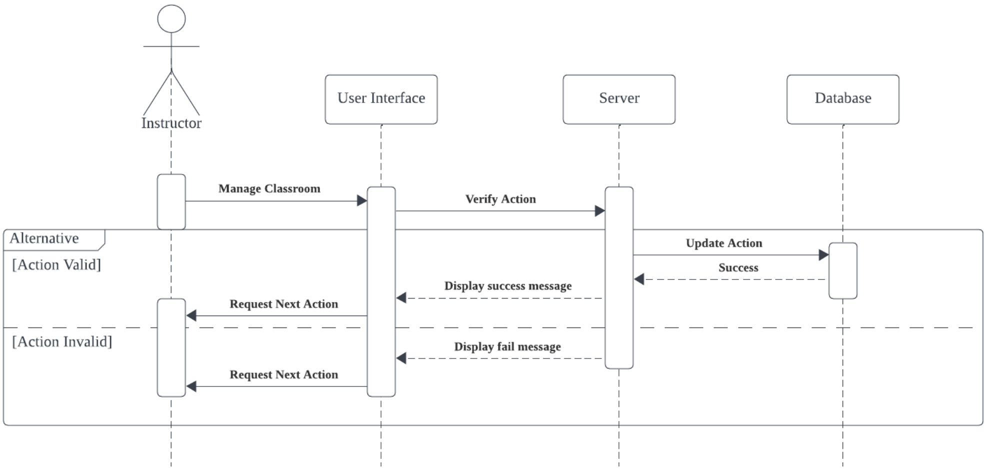


Figure 11 Sequence Diagram for Manage Student Enrolment

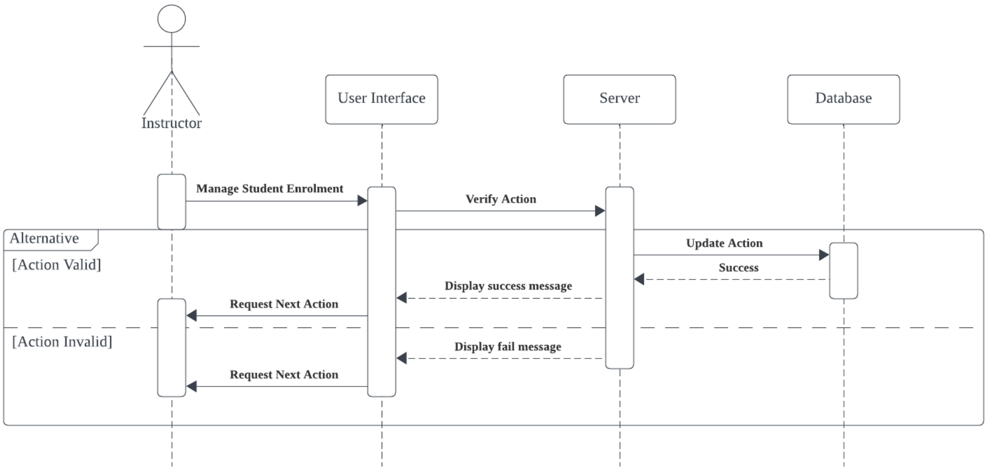


Figure 12 Sequence Diagram for Manage Assignment

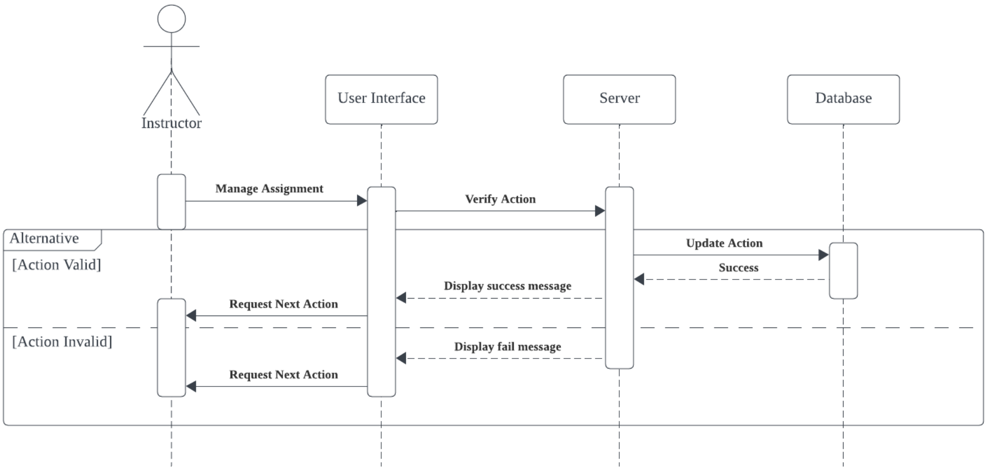


Figure 13 Sequence Diagram for Set Assignment Attributes

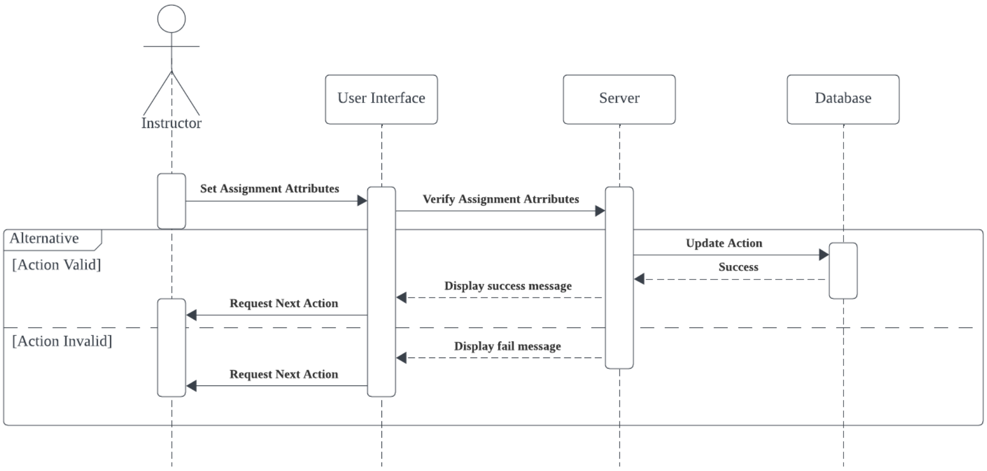


Figure 14 Sequence Diagram for Manage Grading Rubric

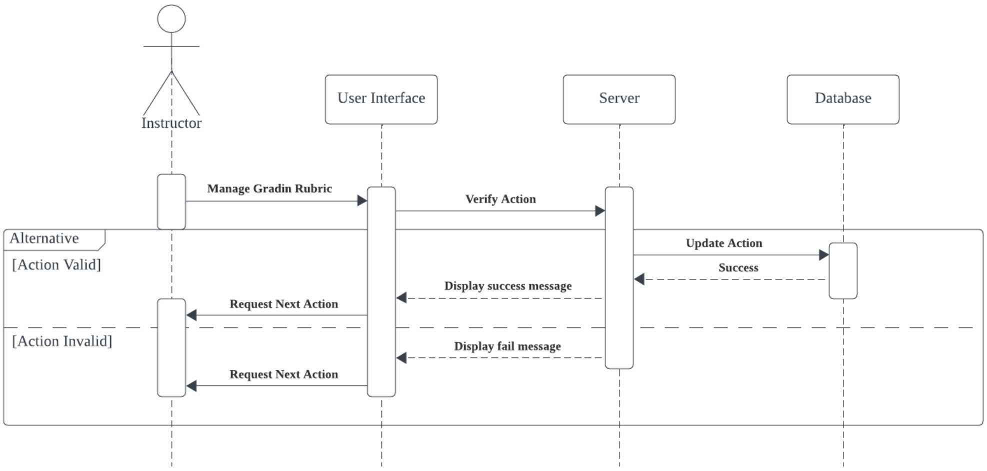


Figure 15 Sequence Diagram for Manage Student Submissions

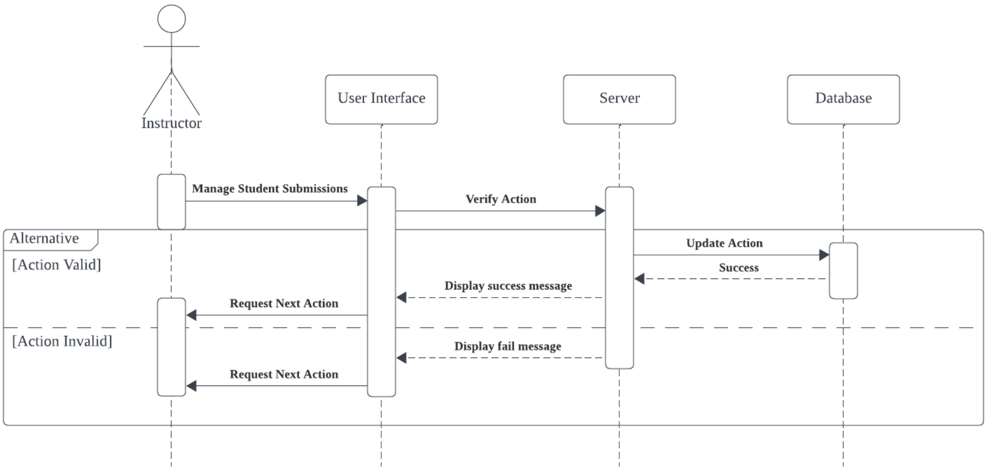


Figure 16 Sequence Diagram for Download Assignment

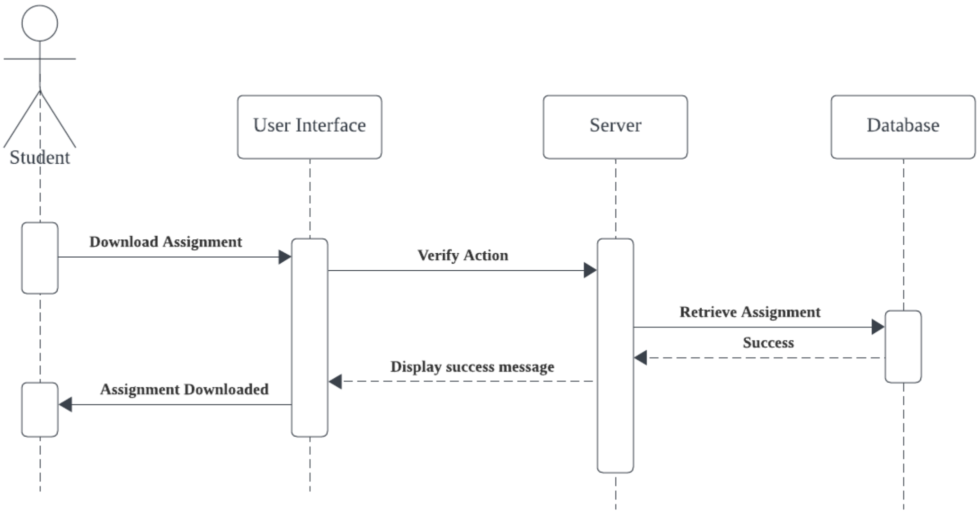


Figure 17 Sequence Diagram for Manage Assignment Submission

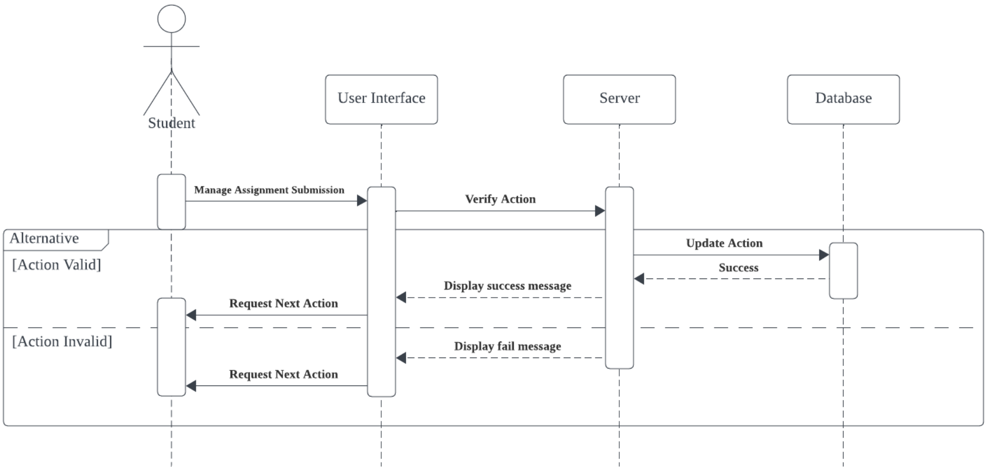


Figure 18 Sequence Diagram for View Grade

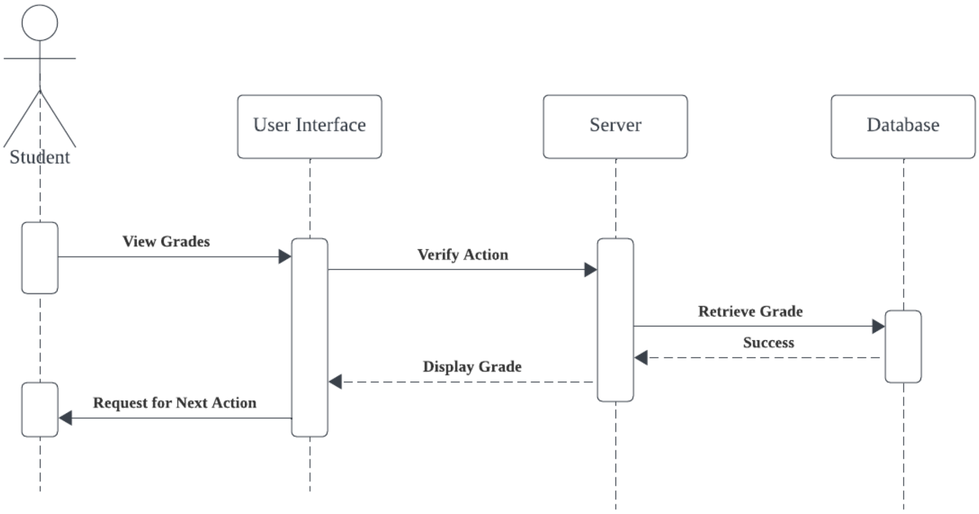


Figure 19 Sequence Diagram for Review Feedback

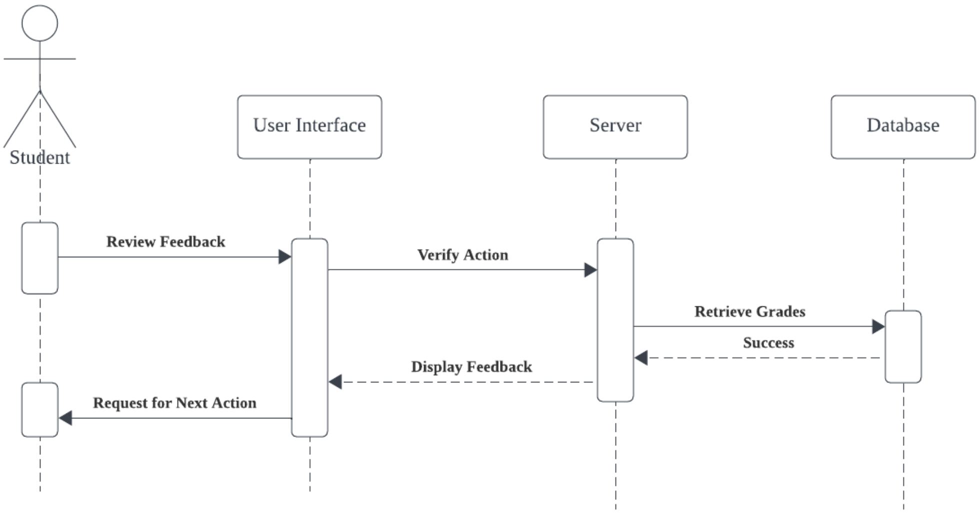


Figure 20 Sequence Diagram for Provide Feedback, Generate Feedback Suggestions

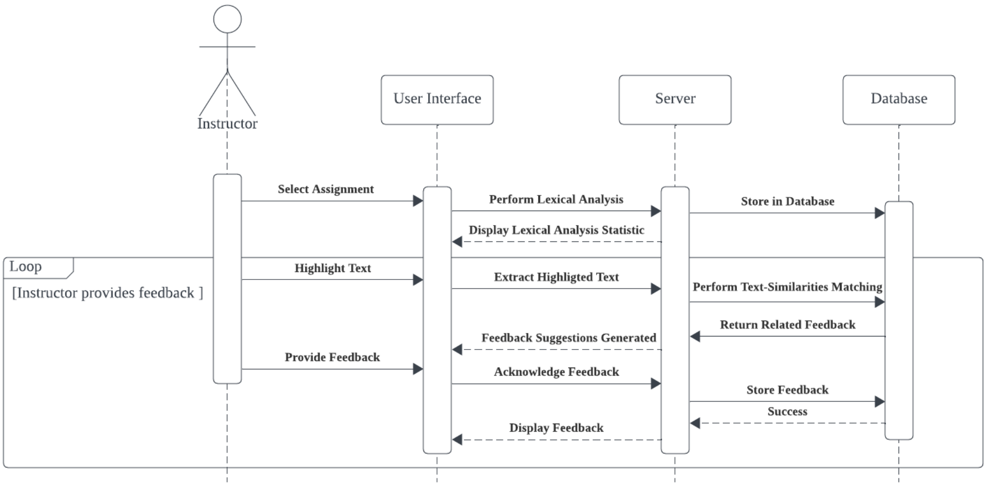


Figure 21 Sequence Diagram for Grade Student Assignment

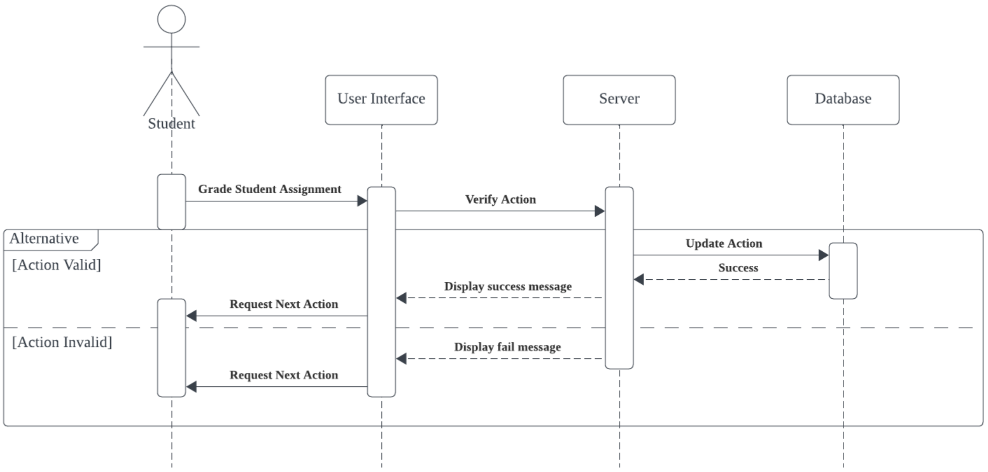


Figure 22 Sequence Diagram for View Dashboard

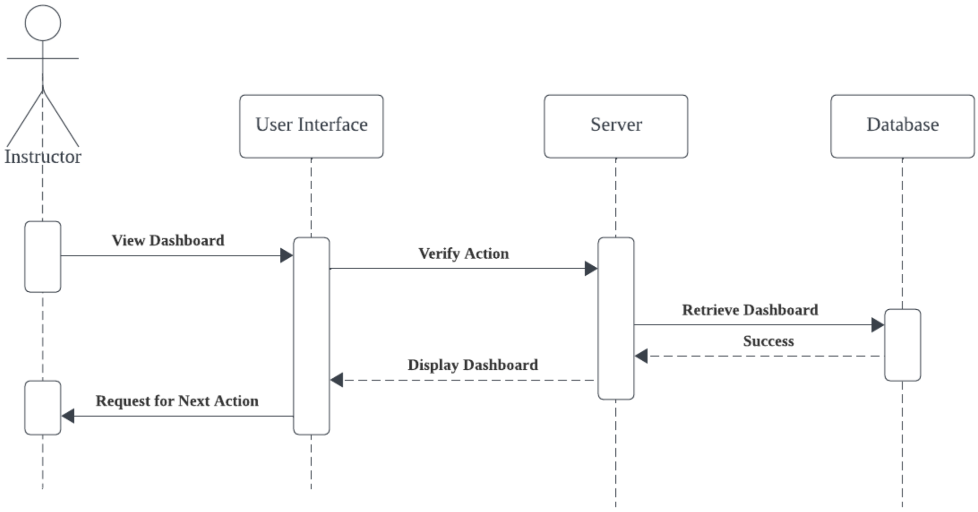
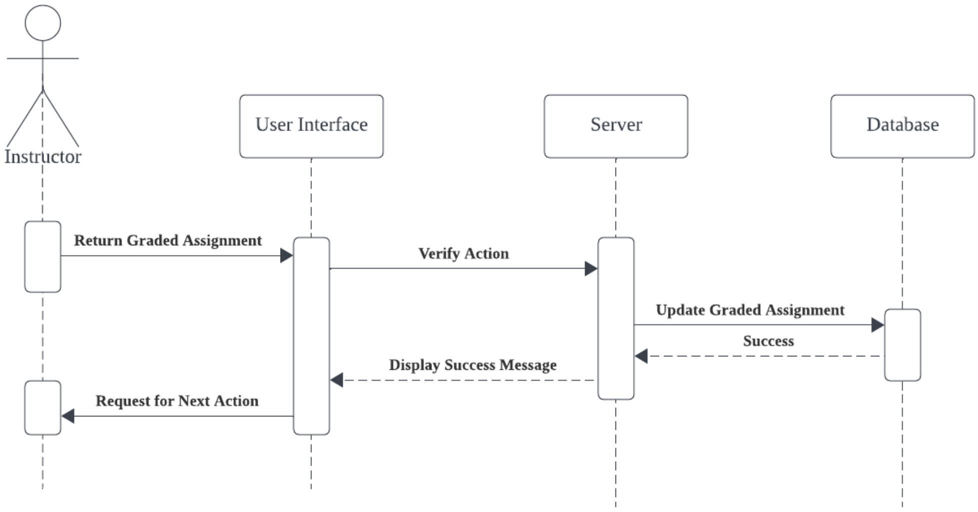


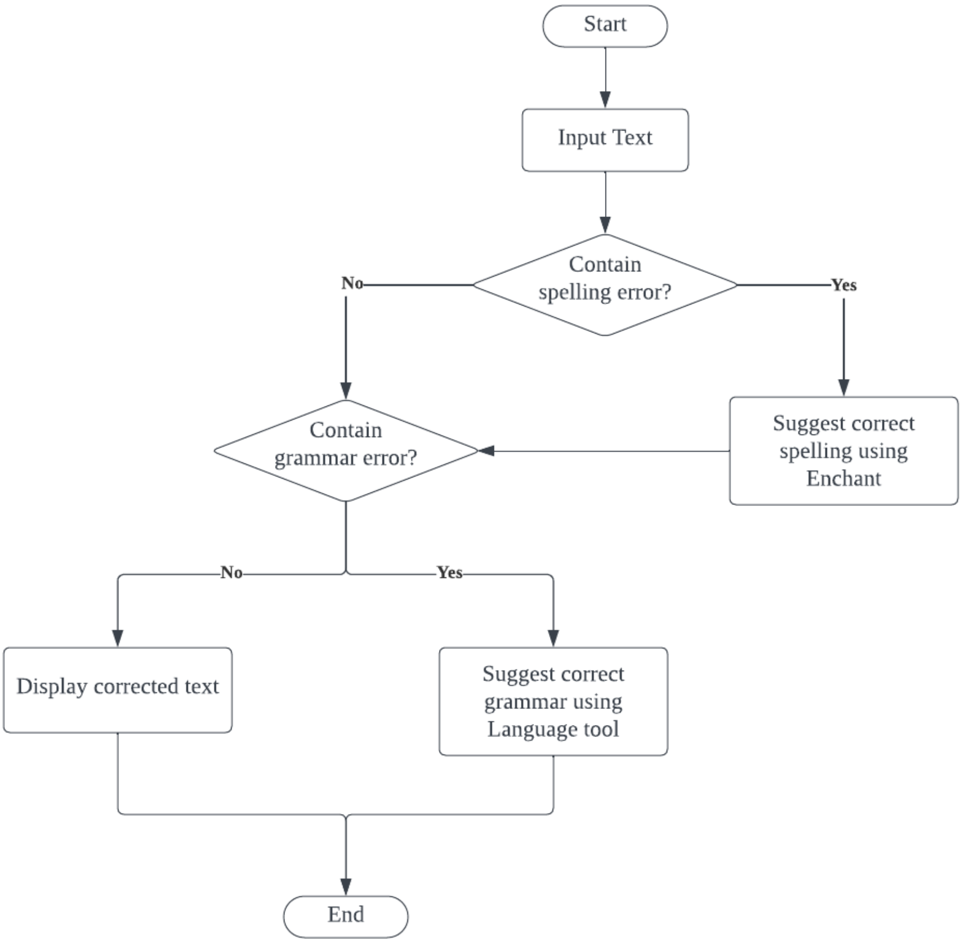
Figure 23 Sequence Diagram for Return Graded Assignment



3.8. Intelligent Components

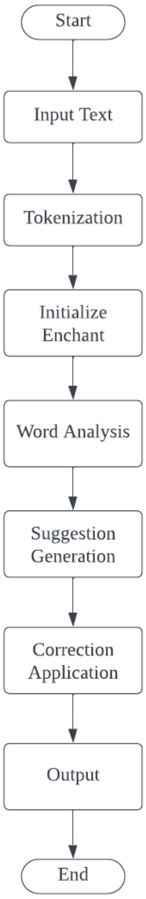
3.8.1. Lexical Analysis for Language and Grammatical Quality Assessment

Figure 24 General Flow of Lexical Analysis for Language and Grammatical Quality Assessment



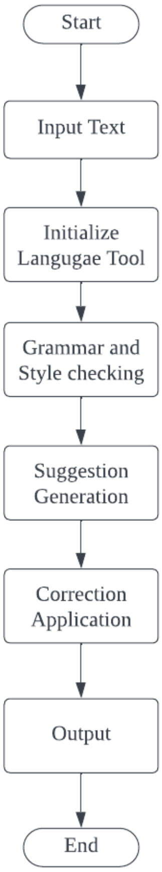
This is the general process flow when the digital text in the assignment is extracted for lexical analysis. Enchant library will be used for spell-checking while the LanguageTool API will be used for grammar checking.

Figure 25 Flow of Echant Library



The enchant library is used for spell checking in natural language processing tasks. In the context of a typical workflow, the process begins with the submission of text for review. The submitted text undergoes tokenization, breaking it down into individual words or tokens. Moving on, enchant is initialized, and the spell-checking backend (e.g., Hunspell, Aspell, or MySpell) is configured. The library then analyzes each word/token for spelling errors, generating correction suggestions for identified issues. At the end of the spell-checking process, the final corrected text is then output [12].

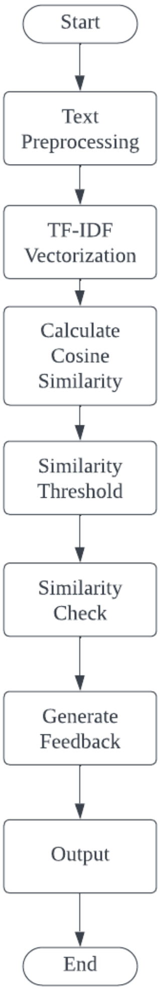
Figure 26 Flow of language-tool-python Library



The language-tool-python library facilitates grammar and style checking using the LanguageTool API. The process begins with the importation of the library and the creation of a LanguageTool instance, specifying the desired language. The input text is then checked for grammar and style issues, producing a list of matches. Each match represents a detected issue with details such as rule ID, error message, and suggested corrections. The suggested corrections will be applied to the original text. The final corrected or improved text will be presented as output, concluding the grammar and style checking process. The language-tool-python library serves as a bridge between Python applications and the LanguageTool service, allowing for seamless integration of grammar and style checking capabilities [13].

3.8.2. Feedback Suggestion Generation

Figure 27 Flow of Feedback Suggestion Generation



The process commences with the retrieval of historical text stored in database. Once retrieved, both the historical and highlighted texts will undergo text preprocessing by converting it into a bag of words, removing stop words and punctuation, and converting the text into lowercase. Next, the TF-IDF Vectorization is applied using the TfidfVectorizer, to convert them into numerical representations. The cosine similarity between the TF-IDF vectors of the historical and highlighted text is then calculated. If the computed similarity exceeds a predefined threshold (e.g., 0.7), the system proceeds to generate feedback. The final output consists of the feedback generated based on the degree of similarity between the two texts [14].

3.9. Data Source

The dataset for this project is derived from a collection of past written assignments provided by the instructor. It is considered as a secondary dataset, as it is not directly collected for the purpose of this project but is rather repurposed from existing written assignment submitted by students.

The dataset revolves around computer science subjects, topics, and technical writing. The size of the dataset is substantial, containing a significant number of assignments collected over multiple academic terms. Each assignment is associated with metadata including assignment title, description, file etc. The attributes of the dataset include the textual content of the assignments, feedback provided by instructor as historical feedback etc. Overall, this dataset serves as a valuable resource for training and testing natural language processing algorithms, enabling the development of systems for text extraction, feedback generation, and lexical analysis within the field of computer science education.

3.10. Technology Deployed

3.10.1. Hardware

1. RAM : 8 GB
2. GPU : 2.0 GHz
3. Monitor Resolution : 1920 x 1080 pixels
4. Internet : Wi-Fi or mobile data hotspot should be connected

3.10.2. Software

1. OS system : Windows
2. Browser : Google Chrome
3. Development IDE : Visual Studio Code
4. Version Control : GitHub
5. Database : MySQL
6. Diagramming Application : Lucidchart
7. Design & Prototyping tools : Figma

3.10.3. Programming Languages/Tools

1. Programming Language : HTML, CSS, JavaScript, Python
2. Frameworks : React.js, As the front-end framework.
   Django, As the backend python web framework.
3. Library : NLTK, scikit-learn, pyenchant,
   language-tool-python

3.10.4. Chosen algorithm

1. Enchant library for spell-checking.
2. language-tool-python library for grammar and style checking text-similarity matching.
3. Text-similarity matching using TF-IDF Vectorization and Cosine Similarity.

4 CONCLUSION

In conclusion, the development of a web application for semi-automatic grading and feedback generation represents a significant step towards addressing the challenges faced by instructors in providing timely and consistent feedback on written assignments, particularly in large classes. The proposed solution, outlined in this project, aims to streamline the feedback process by incorporating features such as classroom management, assignment management, natural language processing for lexical analysis, and grading. By leveraging historical feedback data, the system generates suggestions that offer instructors a valuable tool for maintaining consistency in feedback provision. The anticipated outcomes of the project encompass a comprehensive web application that facilitates efficient classroom management, fosters timely and effective feedback, and enhances the overall academic experience for both instructors and students. The benefits extend to reducing the workload on instructors, ensuring fair and consistent grading, and ultimately contributing to improved academic outcomes. Overall, using technology in education has the potential to make feedback stronger and support students in getting better.

REFERENCES

[1] D. L. B. Muhammad, "Malaysia Educational Statistic - Quick Facts 2023," Educational Macro Data Planning Sector, Putrajaya, 2023.

[2] M. O. H. E. M. Policy Planning And Research Division, "Higher Education Report: [Malaysia]," Unesco National Commission, Paris, 2022.

[3] D. Nicol, "From Monologue To Dialogue: Improving Written Feedback Processes In Mass Higher Education,," *Assessment & Evaluation In Higher Education,* Vol. 35, P. 17, 2010.

[4] E. Boudreau, "How Teachers Can Help Students Receive And Learn From Comments And Critiques," 21 January 2022. [Online]. Available: Https://Www.Gse.Harvard.Edu/Ideas/Usable-Knowledge/22/01/Importance-Feedback.

[5] "Feedbackfruits | Digital Teaching Tools For Higher Ed," [Online]. Available: Https://Feedbackfruits.Com/.

[6] "Gradescope | Save Time Grading," [Online]. Available: Https://Www.Gradescope.Com/.

[7] "Formative | Questions & Hints Powered By Chatgpt," Chatgpt, [Online]. Available: Https://Www.Formative.Com/Ai-Powered.

[8] "Codepost: Autograder And Code Review For Computer Science Courses," Codepost.Io, [Online]. Available: Codepost: Autograder And Code Review For Computer Science Courses.

[9] S. G. Sumit Sharma, "A Correction Model For Real-Word Errors," *Procedia Computer Science,* Vol. 70, Pp. 99-106.

[10] S. Li, "Natural Language Processing For Fuzzy String Matching With Python," Medium, 6 12 2018. [Online]. Available: Https://Towardsdatascience.Com/Natural-Language-Processing-For-Fuzzy-String-Matching-With-Python-6632b7824c49.

[11] R. N. K. I. Kazuaki Hanawa, "Analyzing Methods For Generating Feedback Comments For Language Lernaer," *Journal Of Natural And Processing,* Vol. 29, No. 3, 2022.

[12] R. Kelly, "Pyenchant," 2011. [Online]. Available: Https://Pyenchant.Github.Io/Pyenchant/Tutorial.Html.

[13] J. Moris, "Language-Tool-Python: Checks Grammar Using Languagetool.," Pypi, 18 4 2022. [Online]. Available: Https://Pypi.Org/Project/Language-Tool-Python/.

[14] L. Ramadhan, "Tf-Idf Simplified," Medium, 4 2 2021. [Online]. Available: Https://Towardsdatascience.Com/Tf-Idf-Simplified-Aba19d5f5530.