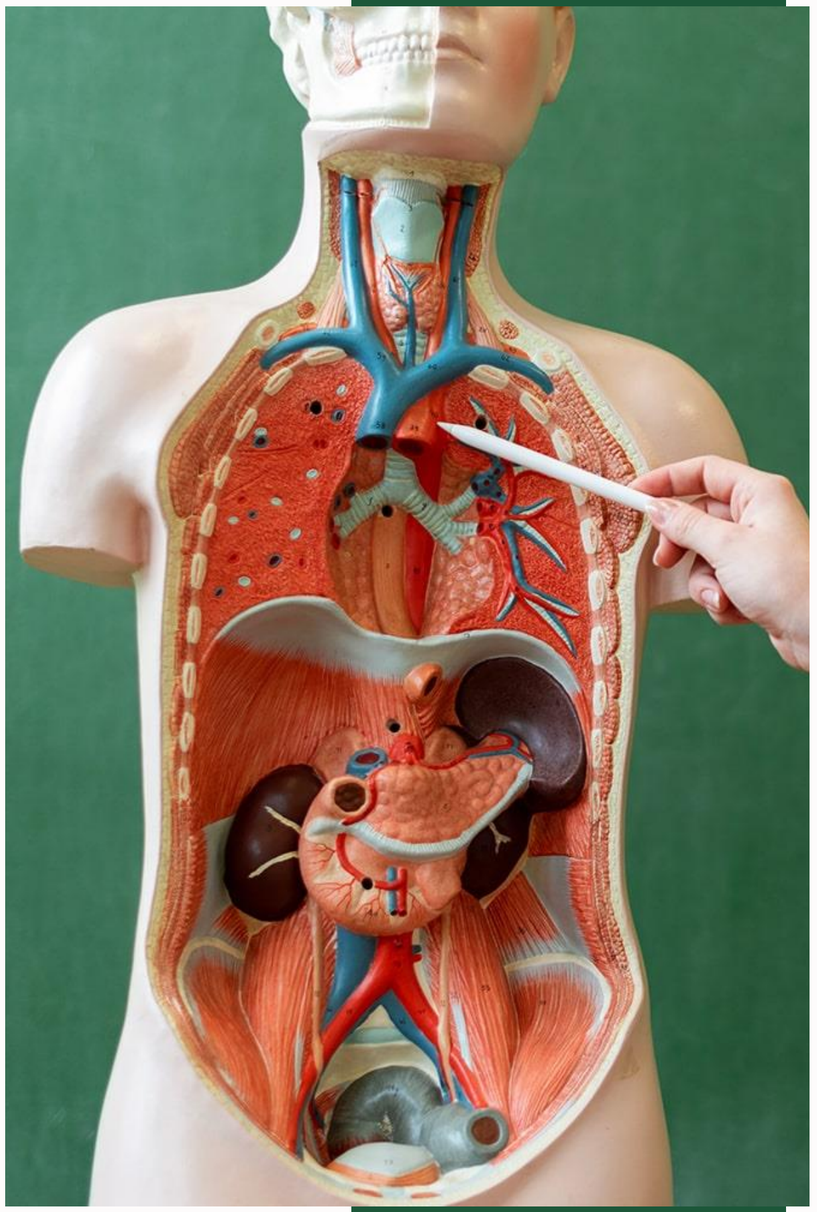
BIOMENTOR - PERSONALIZED E-LEARNING PLATFORM FOR A/L BIOLOGY SUBJECT Students in Sri Lanka

24-25J-257



SLIIT
FACULTY OF COMPUTING

Our Team

Srirajan G. A
IT21375132

Dharane.S
IT21068478

Sujitha.S
IT21264634

Sajeevan.S
IT21204302

N.Thayaparan
External Supervisor

SLIIT
FACULTY OF COMPUTING

Introduction

01 Background

02 Research Problem

03 Research Objectives

04 Overall System Diagram

SLIIT
FACULTY OF COMPUTING

Background

This project aims to create an engaging and effective learning environment that caters to individual learning styles and promotes continuous improvement through detailed feedback and performance tracking.

We’re focusing on this project to provide personalized learning in A-Level Biology using approved government resources. Our advanced technologies aim to offer tailored, engaging experiences that enhance retention and readiness.

SLIIT
FACULTY OF COMPUTING

Research Problem

01

A/L biology students in Sri Lanka struggle with memorizing complex biological terms and their pronunciation. There is a need for an interactive tool that combines spaced repetition with detailed feedback to enhance vocabulary retention and accuracy.

02

Students struggle with extensive biology texts, and existing tools don't effectively extract key concepts or support auditory learners. There is also a lack of tools that provide targeted summaries for exam preparation. A solution is needed to offer concise, accurate content that supports diverse learning styles and aligns with educational standards.

03

Static MCQ platforms do not adapt to students' abilities, leading to ineffective practice and poor identification of knowledge gaps. The objective is to develop an adaptive quiz system that adjusts question difficulty, offers detailed performance analytics, and provides targeted feedback.

04

Existing evaluation systems for biology responses often lack comprehensive feedback and actionable recommendations. The problem is to design a platform that provides accurate answers, detailed feedback, and additional study resources to support student improvement.

Objectives

Objective 01

Biology vocabulary memorization tool using digital flashcards and spaced repetition, providing personalized feedback and adaptive learning paths.

Objective 02

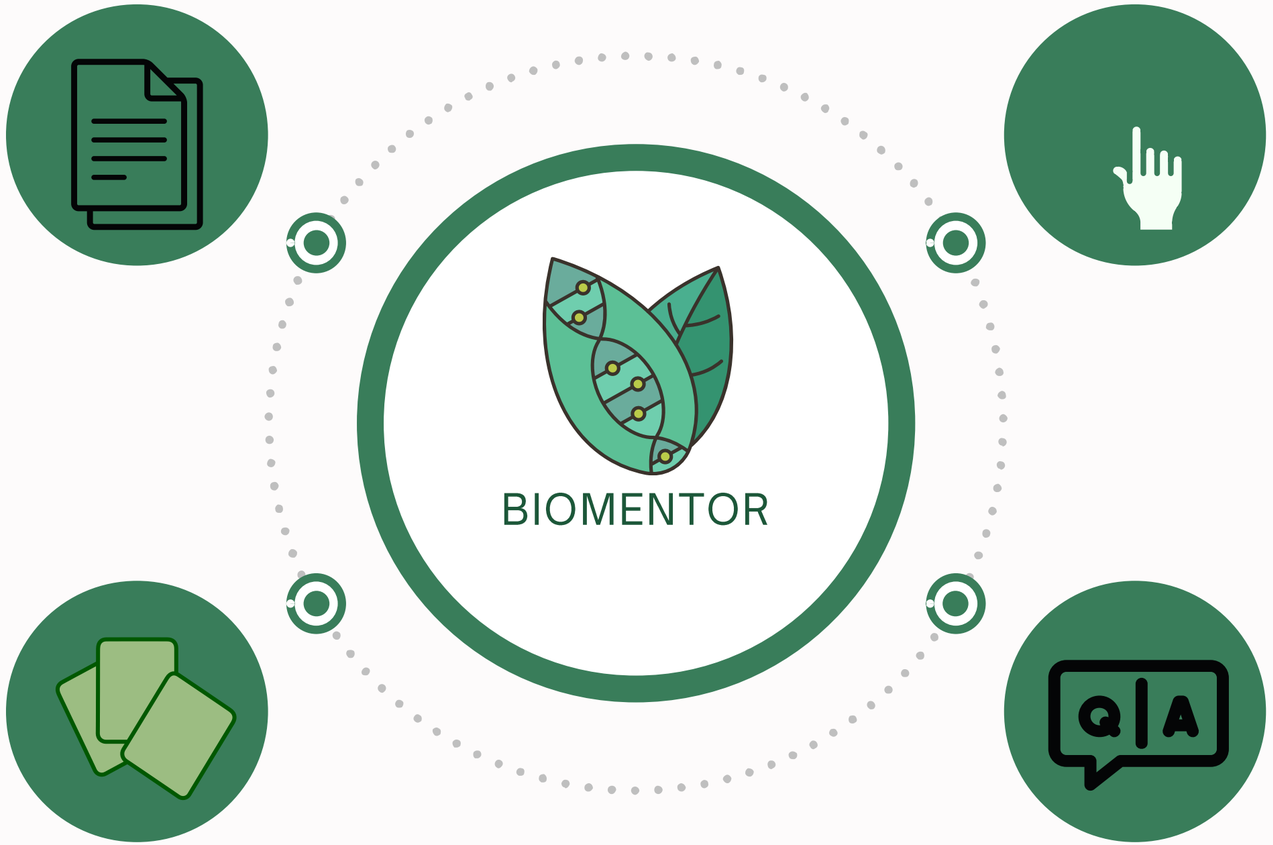
Summarization tool that generates concise, topic-based summaries from uploaded documents and searches through resources to produce summaries on specific topics, with customizable word counts and voice output for diverse learning styles.

Objective 03

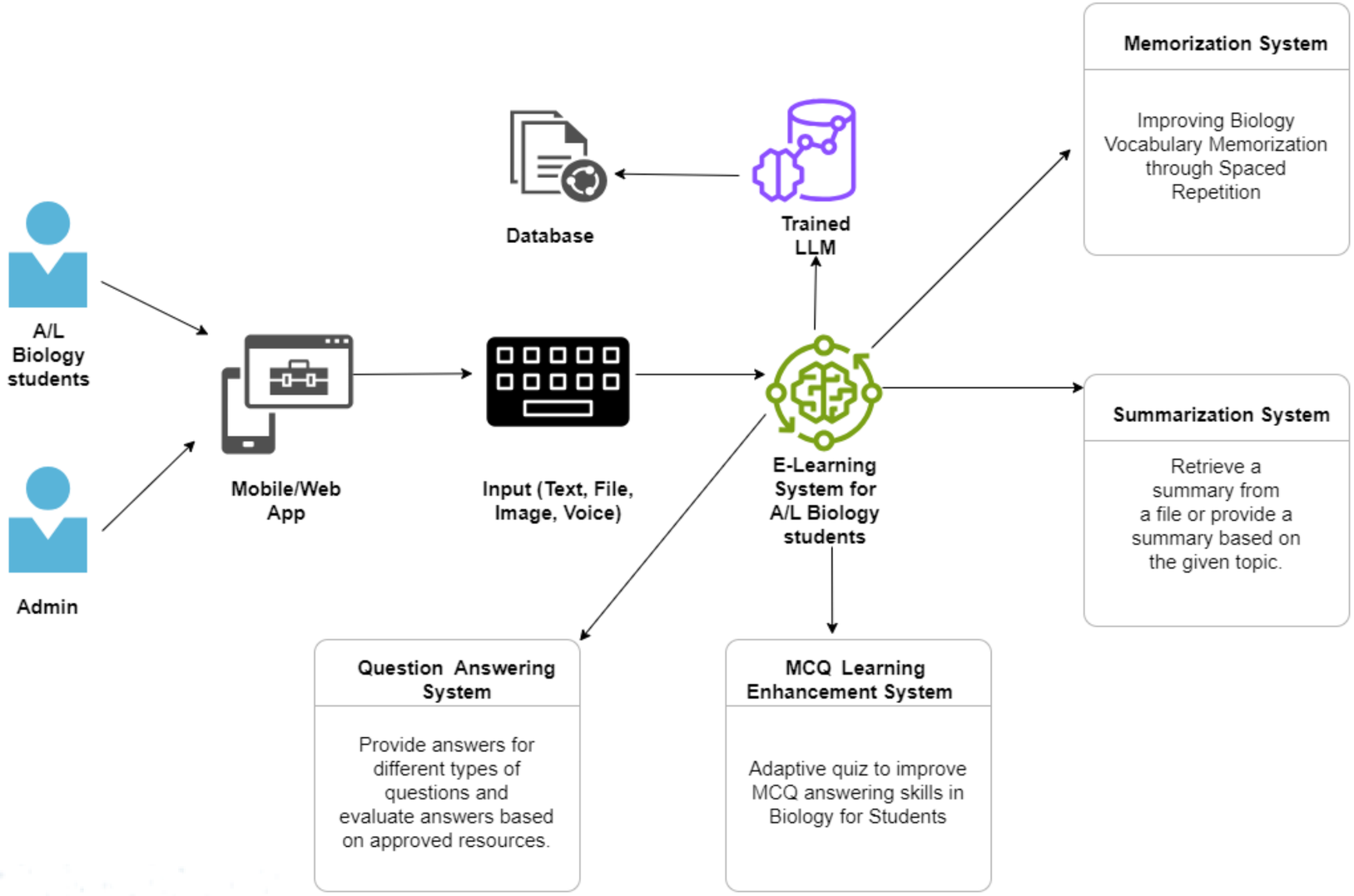
An adaptive quiz platform that dynamically adjusts question difficulty based on student performance, offering targeted practice and detailed performance analysis to enhance learning outcomes.

Objective 04

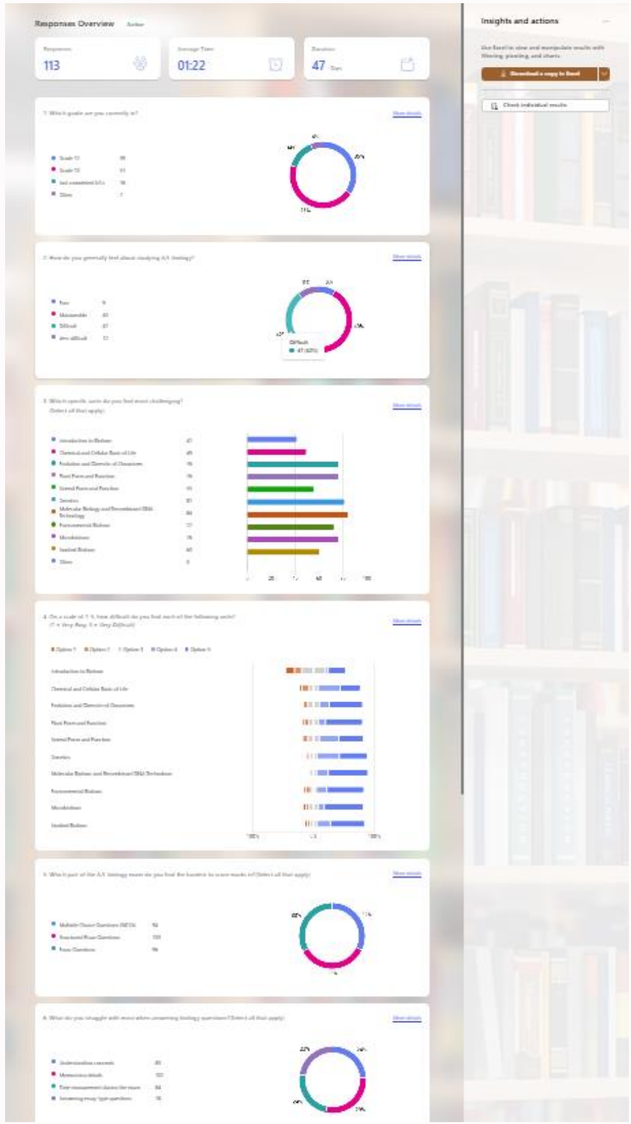
Q & A platform to generate answers for provided questions and evaluate responses, offering feedback and personalized study resources to improve student understanding and performance.



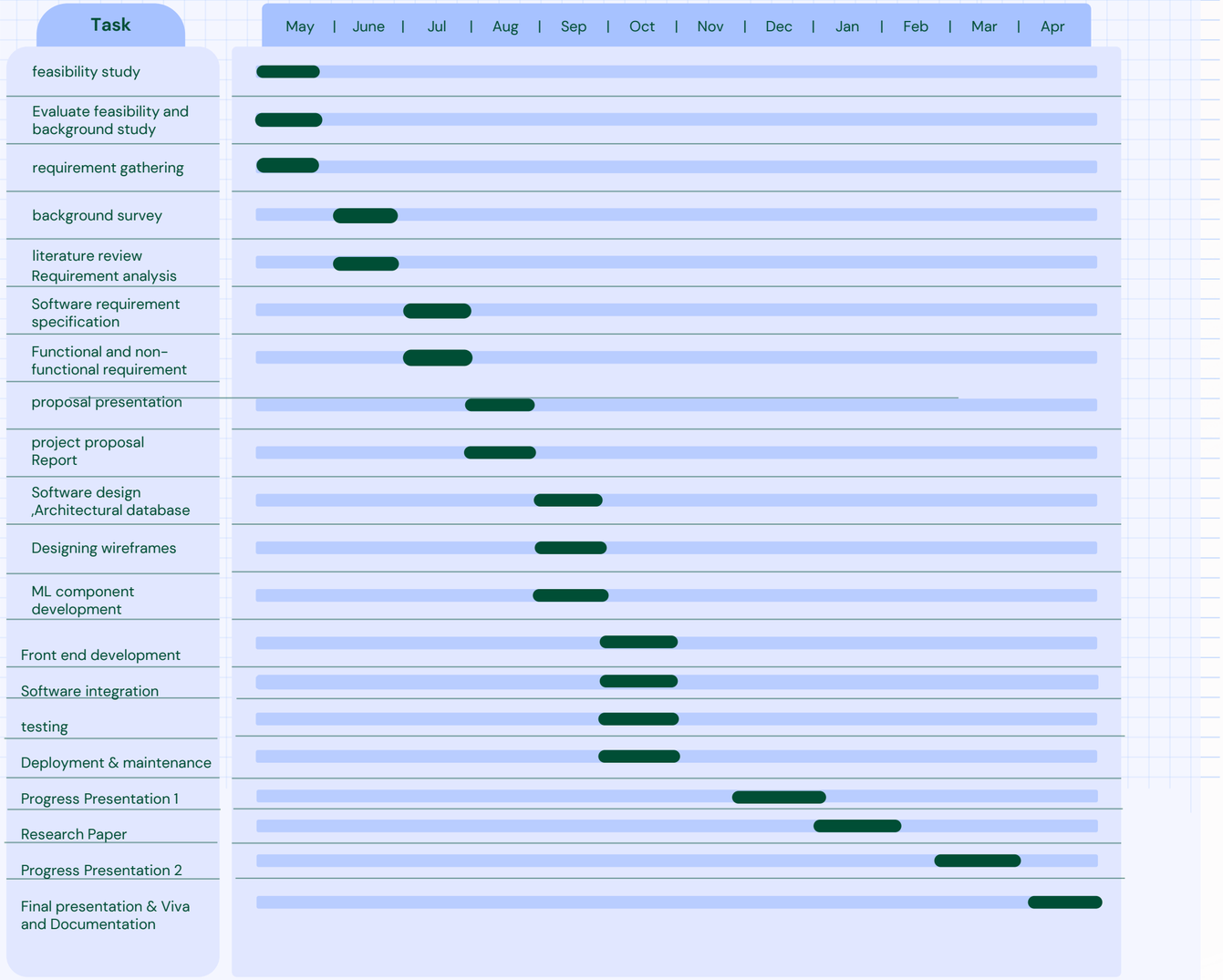
Overall System Diagram



Survey



Gantt Chart


Task
May
June
Jul
Aug
Sep
Oct
Nov
Dec
Jan
Feb
Mar
Apr
feasibility study
Evaluate feasibility and background study
requirement gathering
background survey
literature review Requirement analysis
Software requirement specification
Functional and non-functional requirement
proposal presentation
project proposal Report
Software design ,Architectural database
Designing wireframes
ML component development
Front end development
Software integration
testing
Deployment & maintenance
Progress Presentation 1
Research Paper
Progress Presentation 2
Final presentation & Viva and Documentation

BIOMENTOR

Cost Management Plan

Typs	Cost
Internet use and web hosting	LKR.10,000.00
Training Cost	LKR.30,000.00
Publication Cost	LKR.70,000.00
Stationery	LKR.1,000.00
Total	LKR.111,000.00

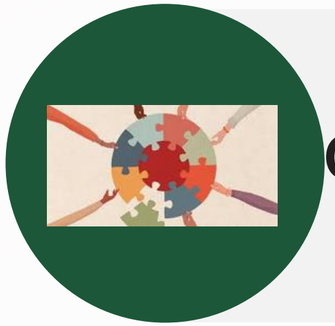
SLIIT
FACULTY OF COMPUTING

BIOMENTOR

Commercialization

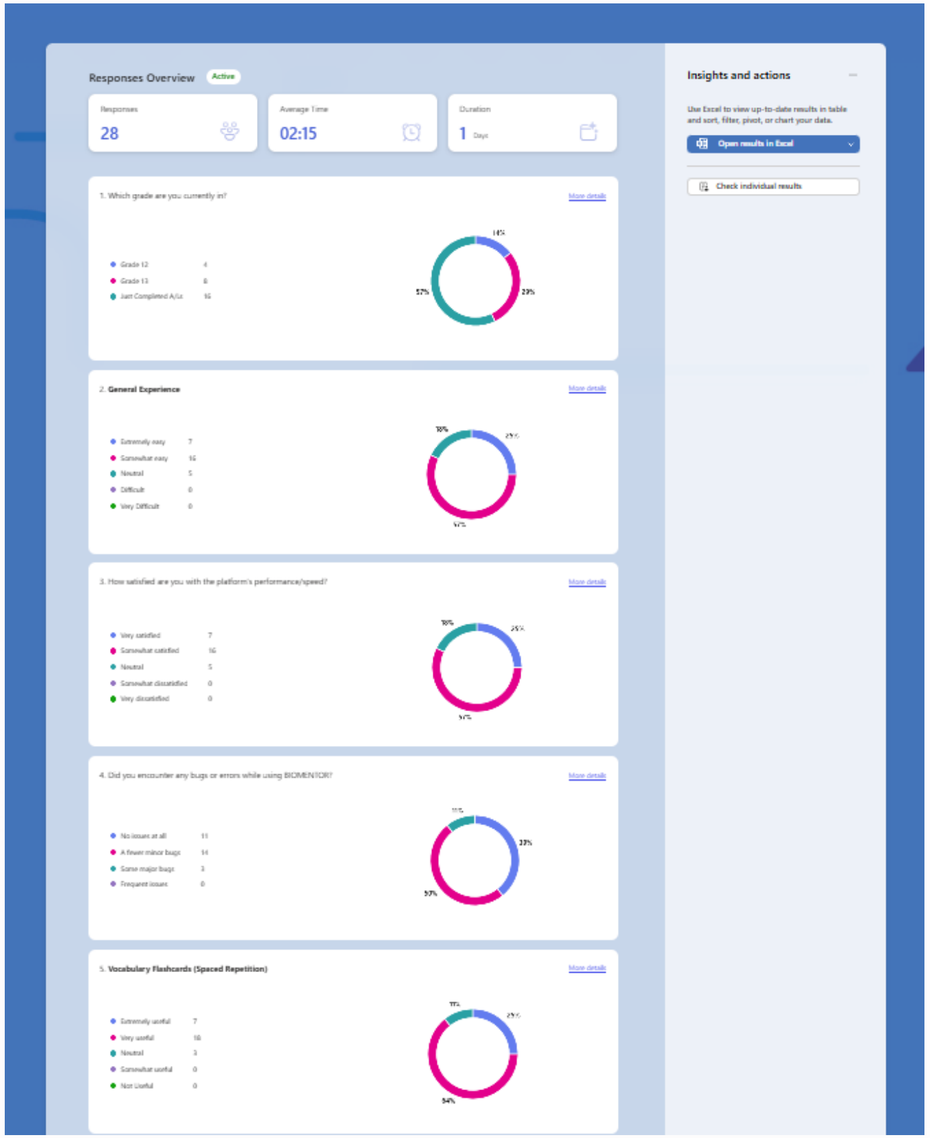
Advertising

Subscription



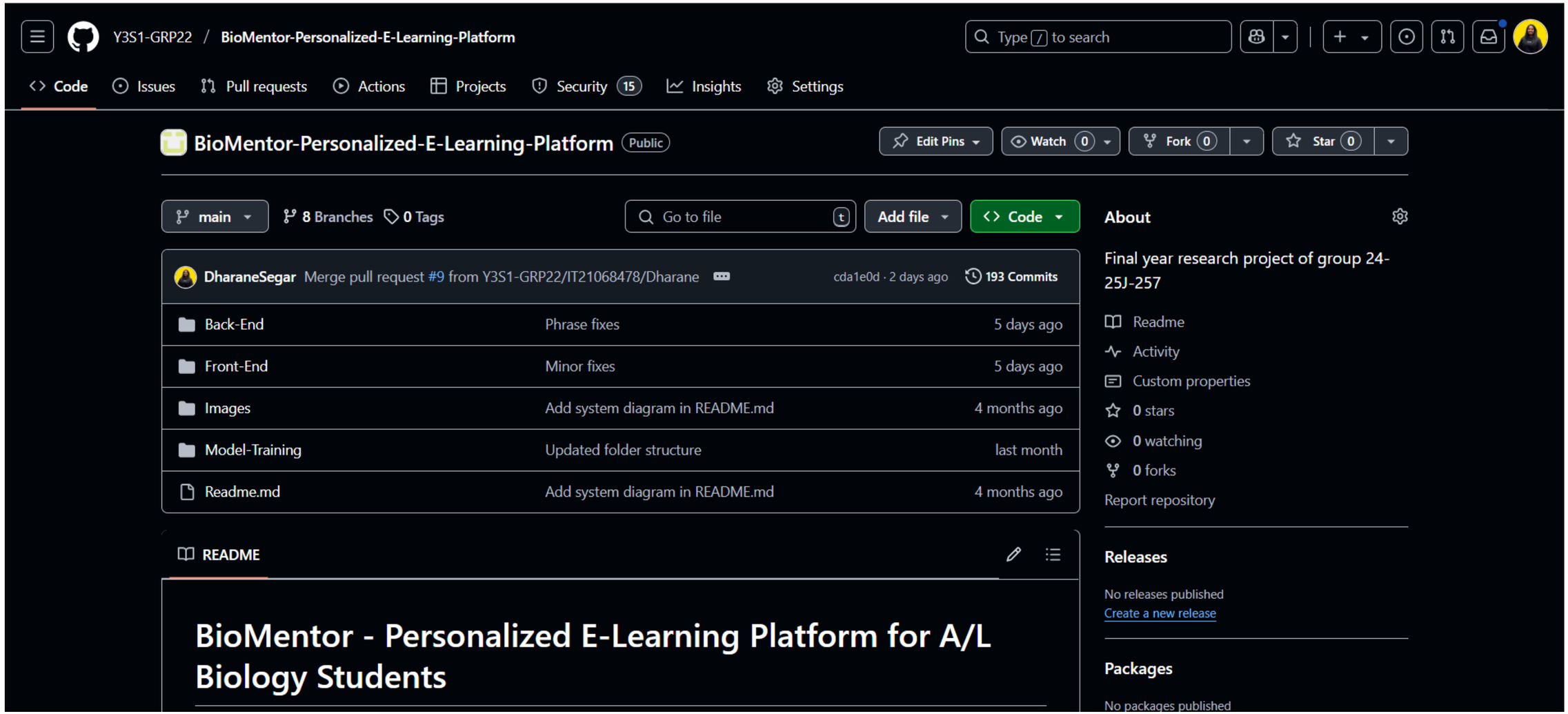
Collaboration

User Feedback



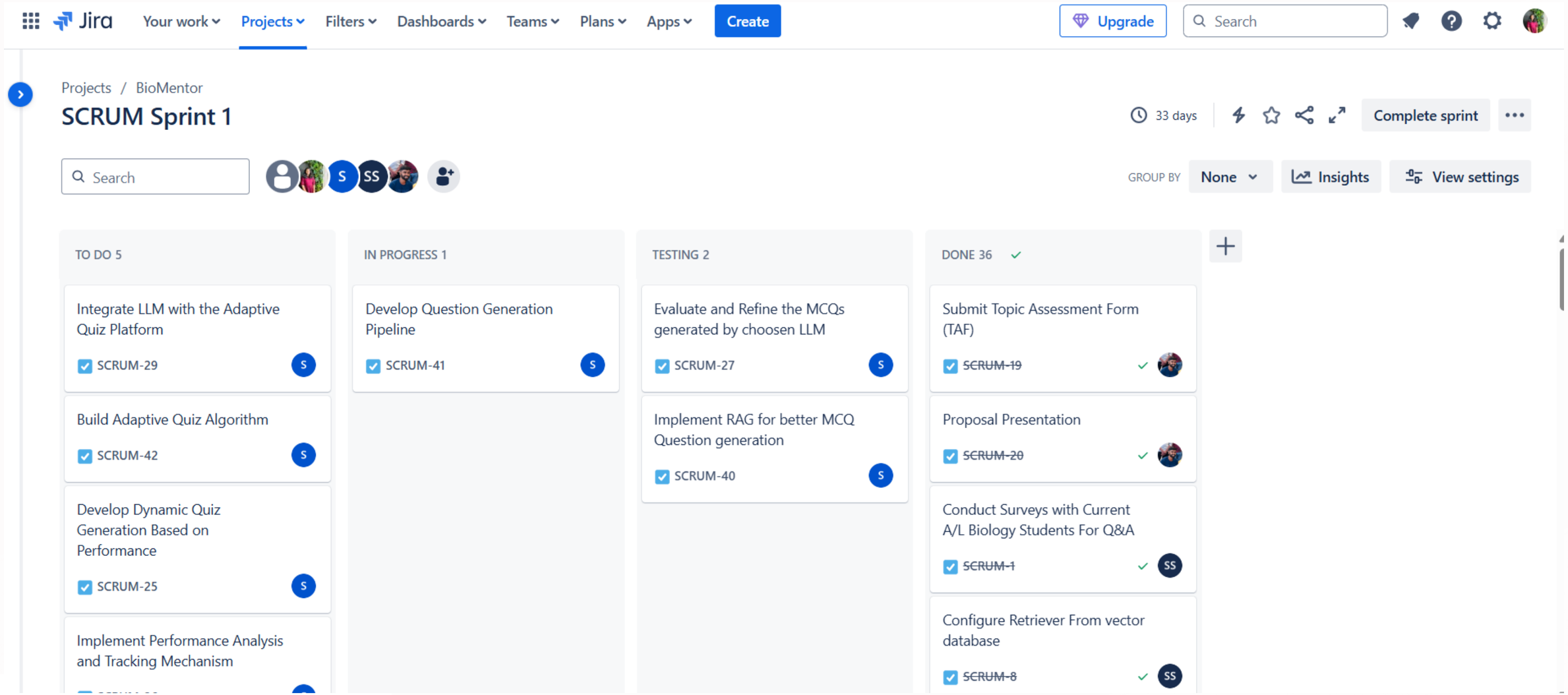
SLIIT
FACULTY OF COMPUTING

GitHub Details



SLIIT
FACULTY OF COMPUTING

Project Management



Confirmation from External Supervisor

Nagalatha Thayaparan
A/L Biology Subject Teacher
Saiva Mangaiyar Vidyalayam
Colombo 06
25th May 2025

To:
The Academic Department
SLIIT, Malabe

Subject: Confirmation of Assistance with Final-Year Project

Dear Sir/Madam,

I am writing to confirm my involvement in supporting the final-year project of the SLIIT Software Engineering students Sujitha S., Dharane S., Sajeevan S., and Abisheak G.S. Their project, a web-based e-learning platform for A/L Biology students, is fully aligned with the A/L Biology syllabus.

I assisted the team by providing past examination papers and syllabus-based materials, as well as helping organize the educational content integrated into the system.

The platform comprises several valuable components. The MCQ generation system performs well, offering questions across different difficulty levels. Most of the questions and answers are accurate and relevant to the syllabus. The Q&A module supports both structured and essay-type questions, provides appropriate answers, and effectively compares students' responses. The summarization tool generates clear and concise notes, and its voice output feature is especially helpful. Additionally, the flashcard feature enables students to review key terms and concepts efficiently.

I also conducted a trial of the system with a group of my students. Based on their feedback, the platform is user-friendly and serves as an effective support tool for exam preparation.

Overall, the system is well-developed and functions smoothly. It will undoubtedly be a valuable resource for A/L Biology students as they prepare for their examinations.

Thank you.

Sincerely,

T. Nagalatha

Nagalatha Thayaparan

IT21375132

Srirajan G.A
Software Engineering

IMPROVE BIOLOGY VOCABULARY MEMORIZATION THROUGH SPACED REPETITION

Introduction

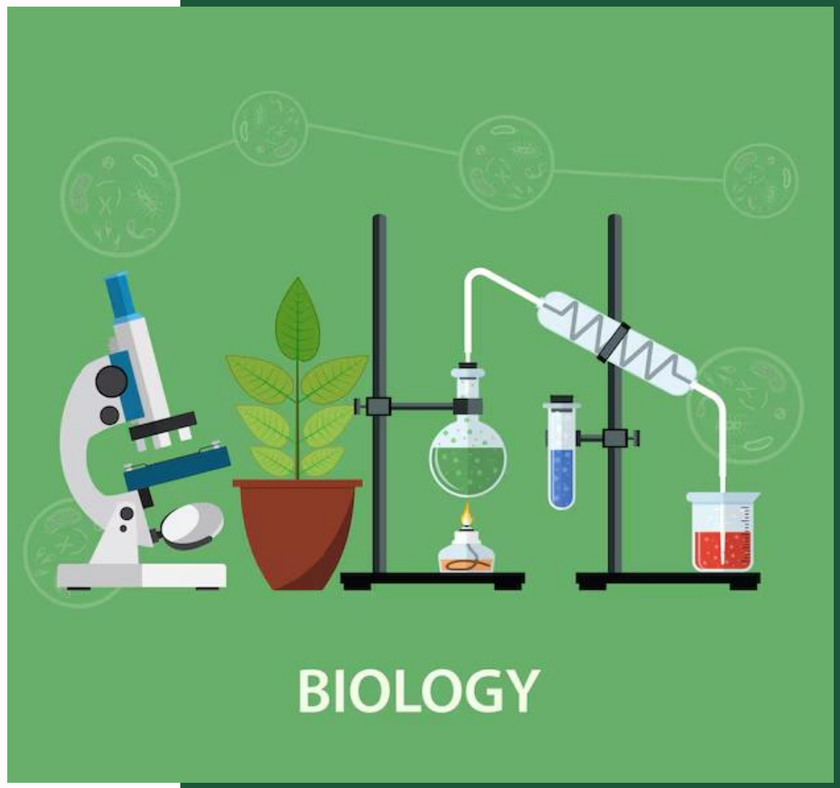
01 Background

02 Research Problem

03 Main and Sub Objectives

04 Methodology

05 Completion Of Project



BACKGROUND

Students struggle with retaining large amounts of biology vocabulary due to the natural forgetting curve, where newly learned information is quickly forgotten if not reviewed.

An innovative system is needed to leverage cognitive science principles, such as spaced repetition, to counteract the forgetting curve.

BACKGROUND

Schistosomiasis

Schistosomiasis is an acute and chronic disease caused by parasitic worms

RESEARCH PROBLEM

01

How can we customize the spaced intervals based on the performance of the user and the difficulty of the vocabulary to maximize memory retention among Advanced Level Biology students?

How can we incorporate multi-sensory spaced repetition techniques on the memorization of biology vocabulary to maximize the memory retention among Advanced Level Biology students?

02

SLIIT FACULTY OF COMPUTING

IT21375132 |Srirajan G.A | 24-25J-257

OBJECTIVES

Objective 1

Create a spaced repetition model that adapts to individual user performance and adjusts review intervals based on the difficulty level of the vocabulary.

Objective 2

Incorporating multi-sensory elements
to cater to different learning styles and enhance the memorization process.

Main Objective

To create an interactive application with flashcards that enhances biology vocabulary memorization through a custom spaced repetition model, which analyzes user performance and the difficulty of the questions and will repeat accordingly.

Objective 3

Incorporate gamification elements to maintain user motivation and engagement throughout the learning process

Objective 4

Implement a model to check the accuracy of the word that was entered.

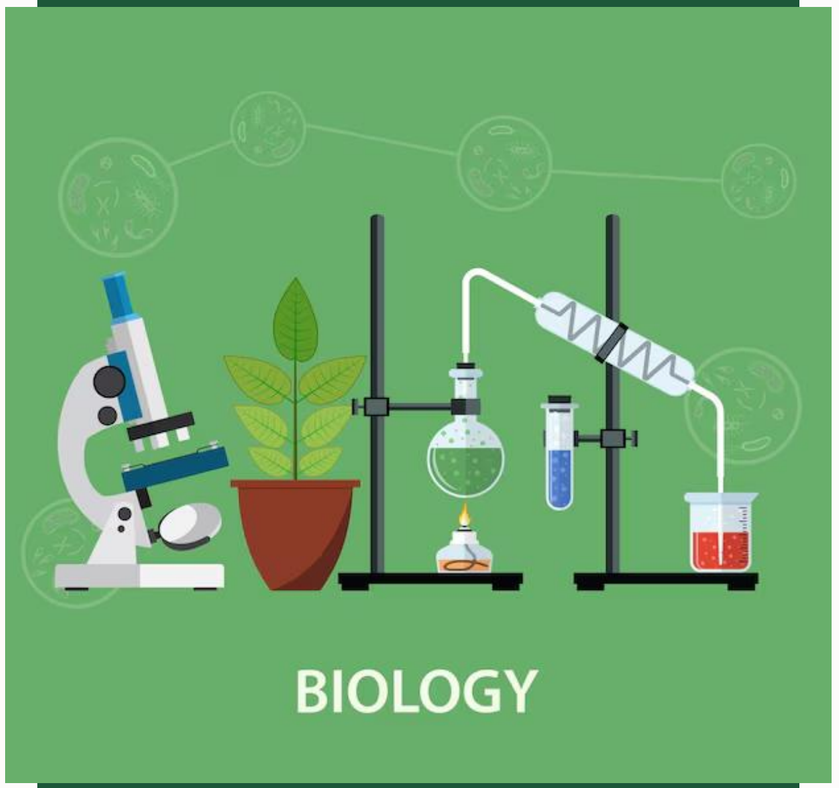
Methodology

01 System Diagram

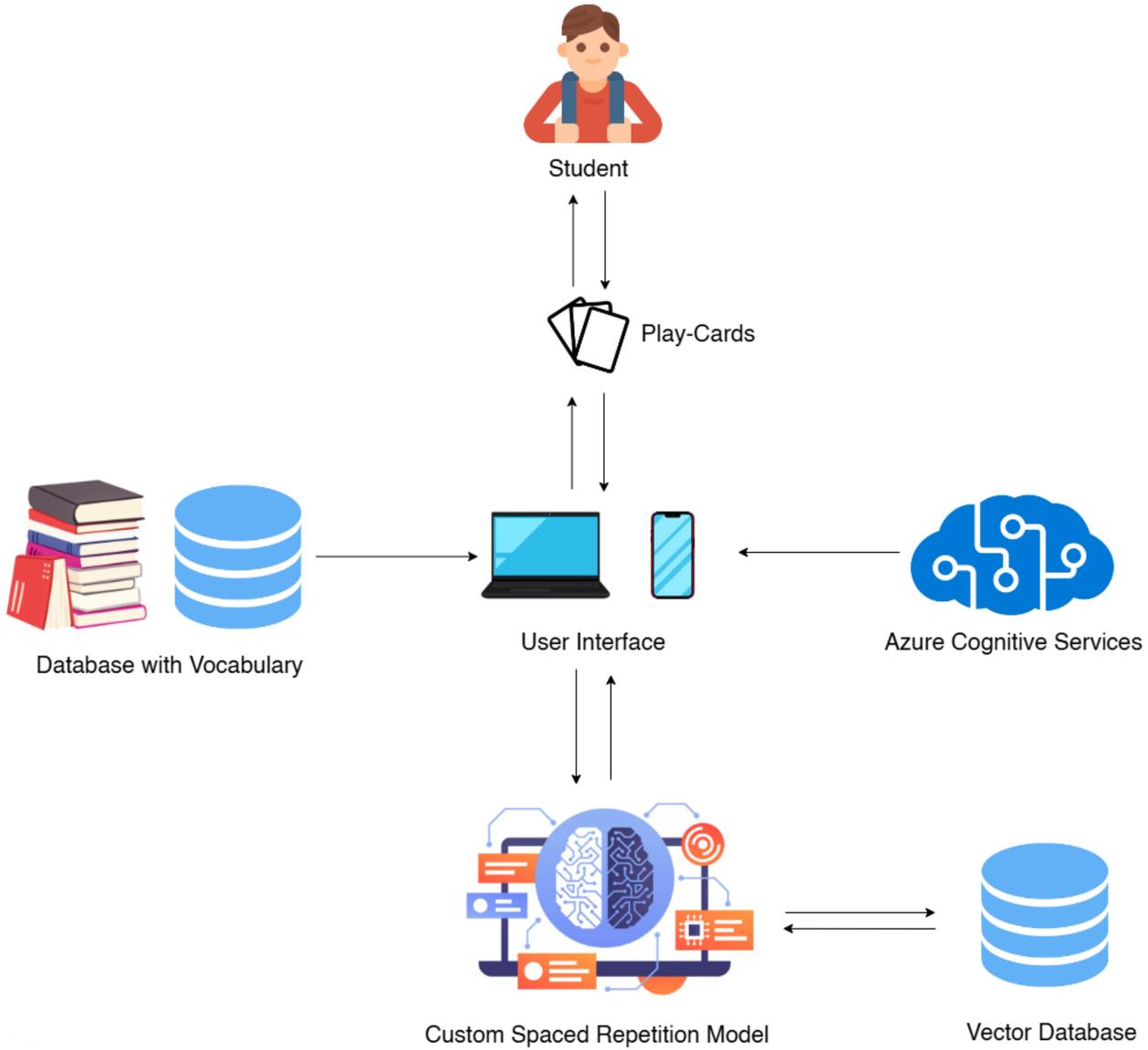
02 Tools and Technologies

03 Best Practices

04 GitHub Commits



System Diagram



Tools & Technologies

Programming Languages

Python
Javascript

Database

Faiss
Mongo DB

Project Management

Jira

Frameworks

Flask
React
Tensorflow

Other tools

Git
Figma
Postman
Azure Cognitive Services

Best Practices

Non-Functional Requirements

- Compatibility
- Accuracy
- Performance
- Usability
- Availability

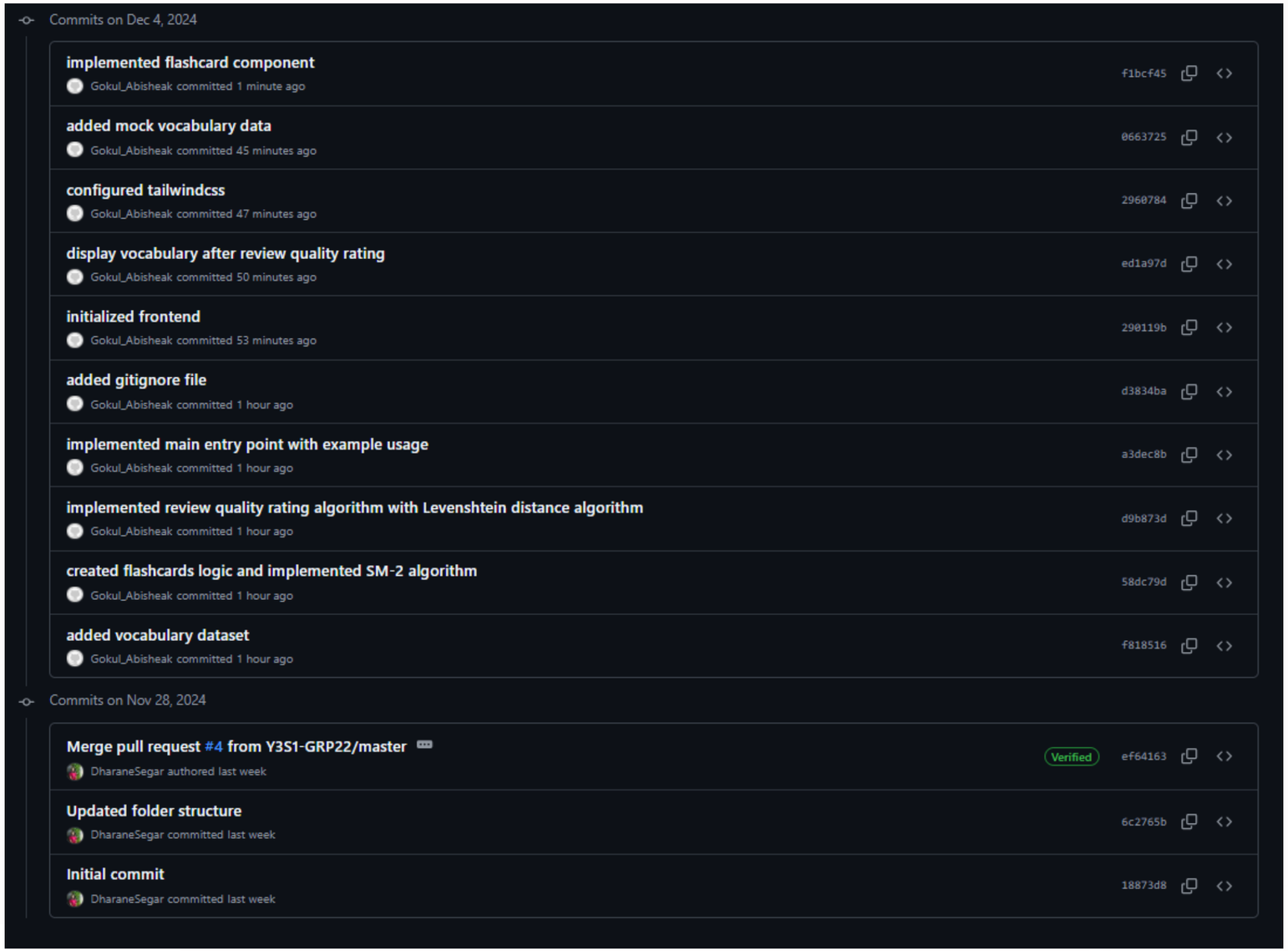
Design Excellence

- Real-Time Performance
- Accuracy and Reliability
- User Experience
- Integration

Standards & Best Practices

- Code Structure & Organization
- Web Security
- Model Quality
- RESTful API Design

GitHub Commits



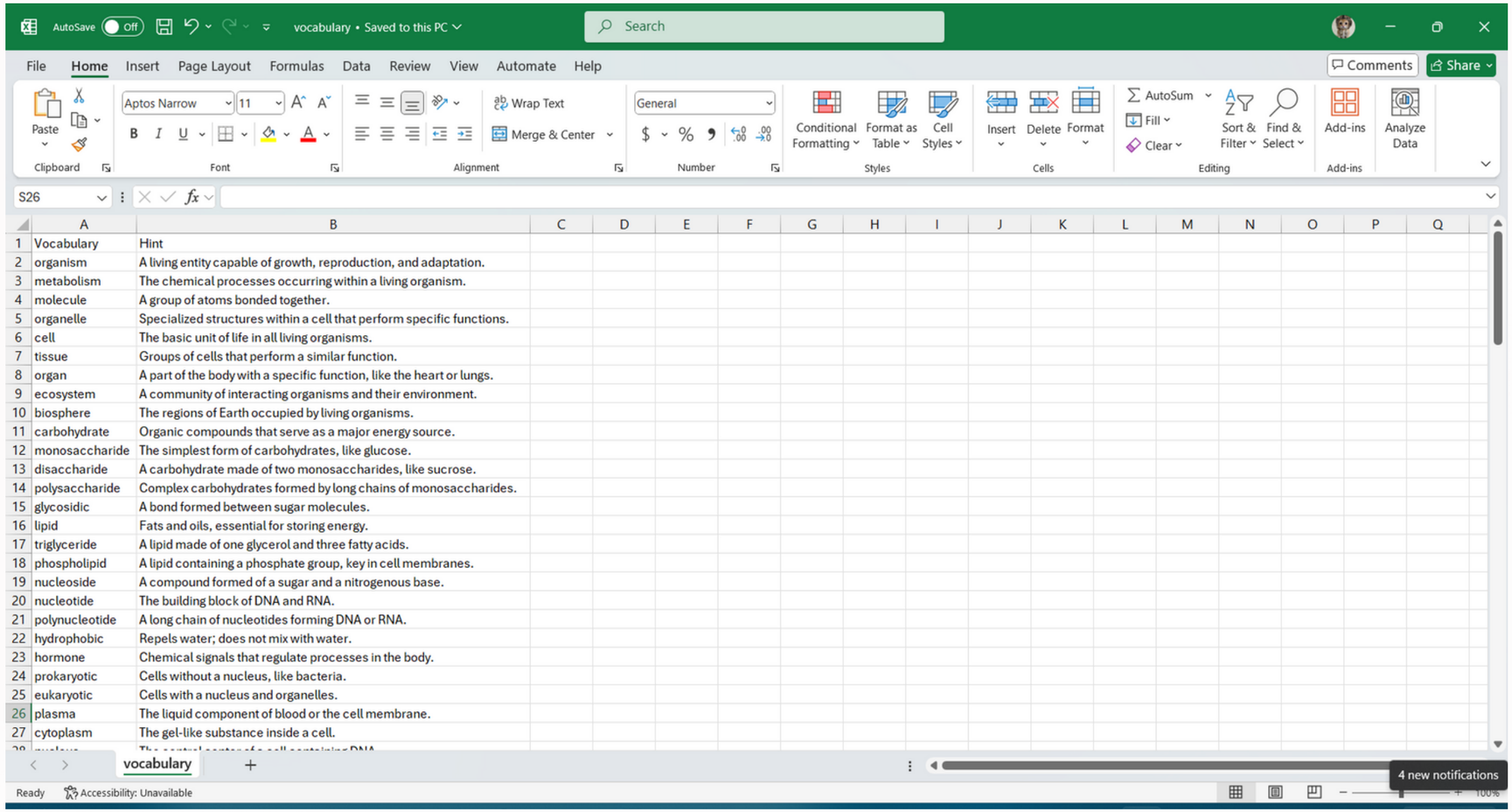
Completion of the project

SLIIT
FACULTY OF COMPUTING

IT21375132 |Srirajan G.A | 24-25J-257

Completion of the project

Data Collection



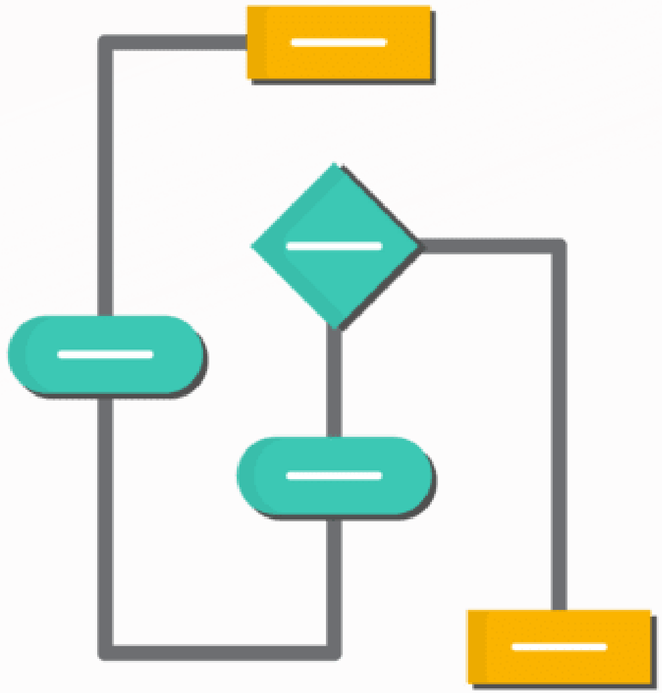
	A	B
1	Vocabulary	Hint
2	organism	A living entity capable of growth, reproduction, and adaptation.
3	metabolism	The chemical processes occurring within a living organism.
4	molecule	A group of atoms bonded together.
5	organelle	Specialized structures within a cell that perform specific functions.
6	cell	The basic unit of life in all living organisms.
7	tissue	Groups of cells that perform a similar function.
8	organ	A part of the body with a specific function, like the heart or lungs.
9	ecosystem	A community of interacting organisms and their environment.
10	biosphere	The regions of Earth occupied by living organisms.
11	carbohydrate	Organic compounds that serve as a major energy source.
12	monosaccharide	The simplest form of carbohydrates, like glucose.
13	disaccharide	A carbohydrate made of two monosaccharides, like sucrose.
14	polysaccharide	Complex carbohydrates formed by long chains of monosaccharides.
15	glycosidic	A bond formed between sugar molecules.
16	lipid	Fats and oils, essential for storing energy.
17	triglyceride	A lipid made of one glycerol and three fatty acids.
18	phospholipid	A lipid containing a phosphate group, key in cell membranes.
19	nucleoside	A compound formed of a sugar and a nitrogenous base.
20	nucleotide	The building block of DNA and RNA.
21	polynucleotide	A long chain of nucleotides forming DNA or RNA.
22	hydrophobic	Repels water; does not mix with water.
23	hormone	Chemical signals that regulate processes in the body.
24	prokaryotic	Cells without a nucleus, like bacteria.
25	eukaryotic	Cells with a nucleus and organelles.
26	plasma	The liquid component of blood or the cell membrane.
27	cytoplasm	The gel-like substance inside a cell.

https://www.nie.lk/pdffiles/tg/eALSyl%20Biology.pdf
https://www.nie.lk/pdffiles/other/eGr12OM%20BioResoBook.pdf

Spaced Repetition Algorithm Selection

Why SM-2?

- Scientifically proven retention.
- Highly customizable algorithm.
- Dynamically adjusts review schedules according to user performance.



SM-2 Algorithm

$$E' = E + 0.1 - (5 - Q) * (0.08 + (5 - Q) * 0.02)$$

$$\text{Interval}' = \begin{cases} 1 & \text{if repetitions} = 1 \\ 6 & \text{if repetitions} = 2 \\ \text{Interval} * E & \text{if repetitions} > 2 \end{cases}$$

If Q < 3

1. Reset repetitions to 0
2. Set interval to 1

SM-2 Algorithm

```python
import datetime
import math
import random
import itertools
import csv

# Define the time format for consistent display
time_fmt = "%Y-%m-%d"

class Card:
    """
    Represents a single flashcard used in spaced repetition.
    Attributes:
        - top: The front side of the card (e.g., question, term).
        - bot: The back side of the card (e.g., answer, definition).
        - time: The last review time.
        - repetitions: Number of successful repetitions.
        - interval: Days until the next review.
        - easiness: Easiness factor (affects interval growth).
    """
    def __init__(self, top, bot, time, repetitions=0, interval=1, easiness=2.5):
        self.top = top
        self.bot = bot
        self.time = time.replace(second=0, microsecond=0)
        self.repetitions = repetitions
        self.interval = interval
        self.easiness = easiness

    @property
    def is_new(self):
        """Determine if the card is new (has not been reviewed yet)."""
        return self.repetitions == 0

    @property
    def next_time(self):
        """Calculate the next review date based on the current interval."""
        return self.time + datetime.timedelta(days=math.ceil(self.interval))
```

```python
def repeat(self, quality, time):
    """
    Update the card's state using the SM-2 algorithm.
    Parameters:
        - quality: Review quality (0-5) based on user performance.
        - time: The current datetime.
    """
    assert 0 <= quality <= 5, "Quality should be between 0 and 5."

    # Adjust easiness factor based on quality
    self.easiness = max(1.3, self.easiness + 0.1 - (5 - quality) * (0.08 + (5 - quality) * 0.02))

    if quality < 3:
        # If performance is poor, reset repetitions and interval
        self.repetitions = 0
        self.interval = 1
    else:
        # If performance is acceptable, increment repetitions
        self.repetitions += 1
        if self.repetitions == 1:
            self.interval = 1  # First repetition interval
        elif self.repetitions == 2:
            self.interval = 6  # Second repetition interval
        else:
            # Subsequent intervals grow based on easiness factor
            self.interval *= self.easiness

    # Update the review time
    self.time = time

def __repr__(self):
    """Provide a human-readable representation of the card."""
    return (f"Card: {self.bot}, Next Review: {self.next_time.strftime(time_fmt)}, "
            f"Repetitions: {self.repetitions}, Interval: {self.interval:.2f}, Easiness: {self.easiness:.2f}")
```

Determine Review Quality

Why Levenshtein Distance Algorithm?

- Captures detailed differences between text inputs providing Accuracy.
- Error Handling allows scores and identifies common user mistakes.
- Customizable thereby can be used for Quantitative Scoring.

H	O		N	D	A	
H	Y	U	N	D	A	I

H	Y	U	N	D	A	I
H		O	N	D	A	

Levenshtein Distance Algorithm

```python
from Levenshtein import distance

def word_accuracy(original_word, entered_word):
    """
    Calculate the accuracy score between two words.

    Parameters:
    - original_word (str): The correct word.
    - entered_word (str): The word entered by the user.

    Returns:
    - int: Accuracy score between 0 (worst) and 5 (best).
    """

    # Normalize the words by converting to lowercase
    original_word = original_word.lower()
    entered_word = entered_word.lower()

    # Calculate Levenshtein distance
    max_length = max(len(original_word), len(entered_word))
    if max_length == 0:  # Handle edge case where both strings are empty
        return 5

    edit_distance = distance(original_word, entered_word)

    # Calculate similarity percentage
    similarity = (max_length - edit_distance) / max_length

    # Scale the similarity to a score between 0 and 5
    scaled_score = round(similarity * 5)

    return max(0, min(scaled_score, 5))  # Ensure score is within bounds
```

Output

```
Day 1: Reviewing cards on 2024-12-04

Front: A living entity capable of growth, reproduction, and adaptation.
Enter your answer: organism
Calculated quality rating: 5

Front: The chemical processes occurring within a living organism.
Enter your answer: metabolism
Calculated quality rating: 5

Front: A group of atoms bonded together.
Enter your answer: molecules
Calculated quality rating: 4

Front: Specialized structures within a cell that perform specific functions.
Enter your answer: organ
Calculated quality rating: 3

Front: The basic unit of life in all living organisms.
Enter your answer: cell
Calculated quality rating: 5

Daily Review Log:
 - {'Card': 'A living entity capable of growth, reproduction, and adaptation.', 'Quality': 5, 'Next Review': '2024-12-05'}
 - {'Card': 'The chemical processes occurring within a living organism.', 'Quality': 5, 'Next Review': '2024-12-05'}
 - {'Card': 'A group of atoms bonded together.', 'Quality': 4, 'Next Review': '2024-12-05'}
 - {'Card': 'Specialized structures within a cell that perform specific functions.', 'Quality': 3, 'Next Review': '2024-12-05'}
 - {'Card': 'The basic unit of life in all living organisms.', 'Quality': 5, 'Next Review': '2024-12-05'}
```

```
Updated Card States:
Card: organism, Next Review: 2024-12-05, Repetitions: 1, Interval: 1.00, Easiness: 2.60
Card: metabolism, Next Review: 2024-12-05, Repetitions: 1, Interval: 1.00, Easiness: 2.60
Card: molecule, Next Review: 2024-12-05, Repetitions: 1, Interval: 1.00, Easiness: 2.50
Card: organelle, Next Review: 2024-12-05, Repetitions: 1, Interval: 1.00, Easiness: 2.36
Card: cell, Next Review: 2024-12-05, Repetitions: 1, Interval: 1.00, Easiness: 2.60
Card: tissue, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: organ, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: ecosystem, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: biosphere, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: carbohydrate, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: monosaccharide, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: disaccharide, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: polysaccharide, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: glycosidic, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: lipid, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: triglyceride, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: phospholipid, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: nucleoside, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: nucleotide, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
```

```
Updated Card States:
Card: organism, Next Review: 2024-12-11, Repetitions: 2, Interval: 6.00, Easiness: 2.60
Card: metabolism, Next Review: 2024-12-11, Repetitions: 2, Interval: 6.00, Easiness: 2.70
Card: molecule, Next Review: 2024-12-11, Repetitions: 2, Interval: 6.00, Easiness: 2.50
Card: organelle, Next Review: 2024-12-11, Repetitions: 2, Interval: 6.00, Easiness: 2.22
Card: cell, Next Review: 2024-12-11, Repetitions: 2, Interval: 6.00, Easiness: 2.70
Card: tissue, Next Review: 2024-12-12, Repetitions: 2, Interval: 6.00, Easiness: 2.60
Card: organ, Next Review: 2024-12-12, Repetitions: 2, Interval: 6.00, Easiness: 2.70
Card: ecosystem, Next Review: 2024-12-07, Repetitions: 1, Interval: 1.00, Easiness: 2.06
Card: biosphere, Next Review: 2024-12-07, Repetitions: 0, Interval: 1.00, Easiness: 1.30
Card: carbohydrate, Next Review: 2024-12-12, Repetitions: 2, Interval: 6.00, Easiness: 2.70
Card: monosaccharide, Next Review: 2024-12-07, Repetitions: 1, Interval: 1.00, Easiness: 2.50
Card: disaccharide, Next Review: 2024-12-07, Repetitions: 1, Interval: 1.00, Easiness: 2.60
Card: polysaccharide, Next Review: 2024-12-07, Repetitions: 1, Interval: 1.00, Easiness: 2.60
Card: glycosidic, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: lipid, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: triglyceride, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: phospholipid, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: nucleoside, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: nucleotide, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
Card: polynucleotide, Next Review: 2024-12-05, Repetitions: 0, Interval: 1.00, Easiness: 2.50
```

Flashcard UI

Completion of the project

Key Functionalities

- Spaced Repetition Algorithm (SM-2): Adjusts review intervals based on user performance to reinforce learning before forgetting.
- Daily Flashcard Quiz: Provides 10 vocabulary words daily, prioritizing due reviews before adding new ones.
- Audio Pronunciation : Integrated text-to-speech (TTS) allows users to hear correct word pronunciations for better retention.
- Performance Tracking & Leaderboard : Tracks user progress, review history, and ranks users based on total scores.

SM-2 Algorithm

$$E' = E + 0.1 - (5 - Q) * (0.08 + (5 - Q) * 0.02)$$

$$\text{Interval}' = \begin{cases} 1 & \text{if repetitions} = 1 \\ 6 & \text{if repetitions} = 2 \\ \text{Interval} * E & \text{if repetitions} > 2 \end{cases}$$

If Q < 3

1. Reset repetitions to 0
2. Set interval to 1

SM-2 Algorithm

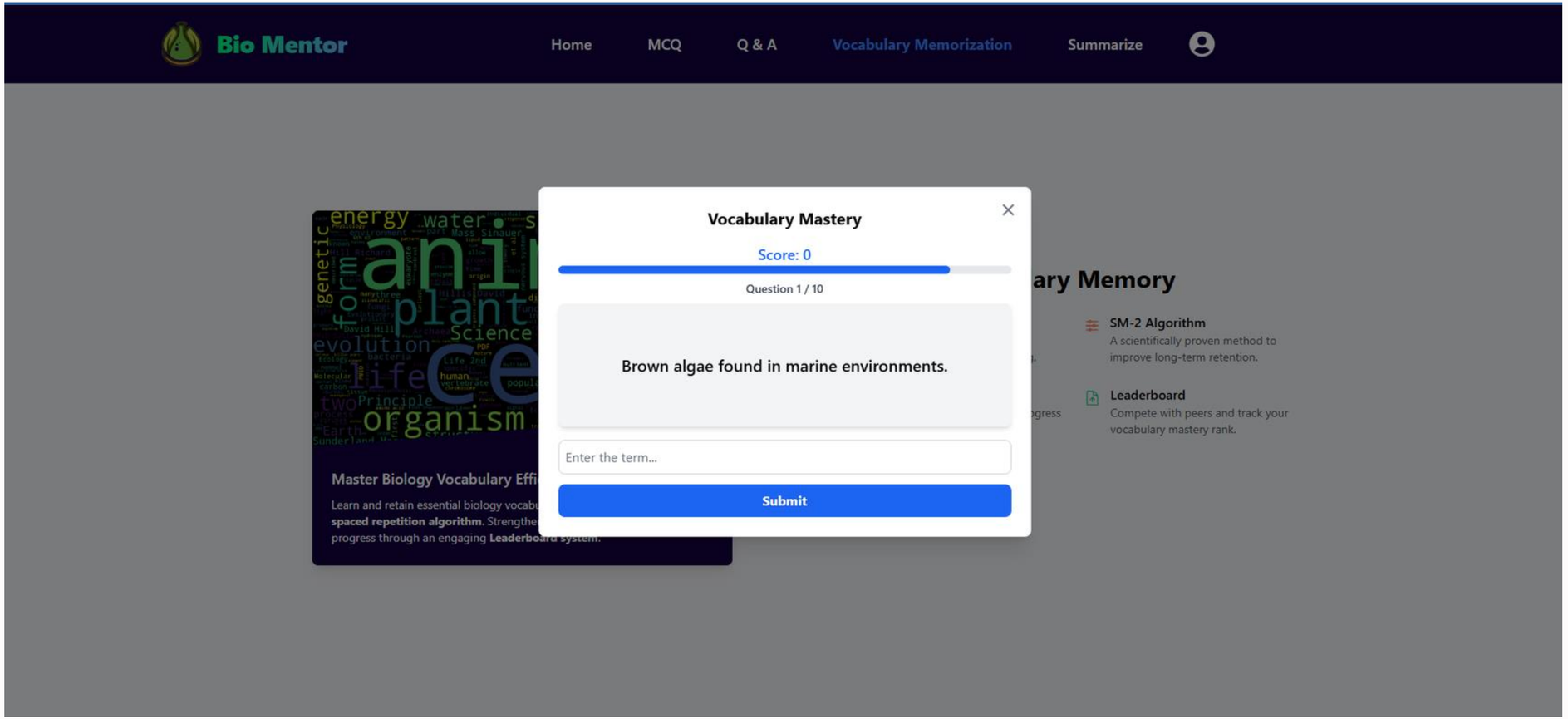
```python
async def assign_daily_questions():
    """Assign exactly 10 random flashcards from MongoDB."""
    cursor = db["flashcards"].aggregate([{"$sample": {"size": 10}}])
    questions = await cursor.to_list(length=10)

    # Convert ObjectId to string
    for q in questions:
        q["_id"] = str(q["_id"])

    return questions

async def get_daily_questions(user_id):
    """Fetch up to 10 daily vocabulary words for a user based on spaced repetition."""
    user = await db["users"].find_one({"_id": ObjectId(user_id)})
    if not user:
        return {"error": "User not found"}

    today = datetime.utcnow().date()
    due_questions = []

    # Collect words due for review
    for entry in user.get("history", []):
        if datetime.strptime(entry["next_review"], "%Y-%m-%d").date() <= today:
            due_questions.append(ObjectId(entry["question_id"]))

    # Fetch due words from MongoDB
    questions = await db["flashcards"].find({"_id": {"$in": due_questions}}).to_list(len(due_questions))

    # Convert _id to string
    seen_ids = {q["_id"] for q in questions}

    # If fewer than 10, fetch additional new ones
    remaining_slots = max(0, 10 - len(questions))
    if remaining_slots > 0:
        cursor = db["flashcards"].aggregate([
            {"$match": {"_id": {"$nin": list(seen_ids)}}},
            {"$sample": {"size": remaining_slots}}
        ])
        new_questions = await cursor.to_list(length=remaining_slots)
        questions.extend(new_questions)

    # Strictly limit to 10 questions
    questions = questions[:10]

    # Convert ObjectId to string for JSON serialization
    for q in questions:
        q["_id"] = str(q["_id"])

    return {"questions": questions}
```

```python
async def update_progress(user_id, question_id, user_answer):
    """Update spaced repetition progress for a specific user."""
    from datetime import datetime, timedelta
    from utils.review_quality import word_accuracy

    # Fetch the vocabulary word
    question = await db["flashcards"].find_one({"_id": ObjectId(question_id)})
    if not question:
        return {"error": "Question not found"}

    # ✅ Compare with vocabulary word, not the hint
    review_score = word_accuracy(question["question"], user_answer)

    # Fetch or create user record
    user = await db["users"].find_one({"_id": ObjectId(user_id)})
    if not user:
        user = {
            "_id": ObjectId(user_id),
            "username": f"user_{user_id}",  # Temporary username
            "total_score": 0,
            "history": []
        }
        await db["users"].insert_one(user)

    # Check if question exists in user history
    history_entry = next((h for h in user["history"] if h["question_id"] == str(question_id)), None)

    if history_entry is None:
        # If first time seeing the word, create a new entry
        next_review = datetime.utcnow() + timedelta(days=1)
        history_entry = {
            "question_id": str(question_id),
            "times_seen": 1,
            "review_score": review_score,
            "next_review": next_review.strftime("%Y-%m-%d")
        }
        await db["users"].update_one(
            {"_id": ObjectId(user_id)},
            {"$push": {"history": history_entry}, "$inc": {"total_score": review_score}}
        )
    else:
        # Update existing entry
        history_entry["times_seen"] += 1
        history_entry["review_score"] = review_score

        # ✅ Implement **Spaced Repetition (SM-2 Algorithm)**
        if review_score == 5:
            interval = 10  # Maximum interval
        elif review_score >= 4:
            interval = 5
        elif review_score >= 3:
            interval = 3
        elif review_score >= 2:
            interval = 2
        else:
            interval = 1  # Immediate review required

        history_entry["next_review"] = (datetime.utcnow() + timedelta(days=interval)).strftime("%Y-%m-%d")

        await db["users"].update_one(
            {"_id": ObjectId(user_id), "history.question_id": str(question_id)},
            {"$set": {"history.$": history_entry}, "$inc": {"total_score": review_score}}
        )
```

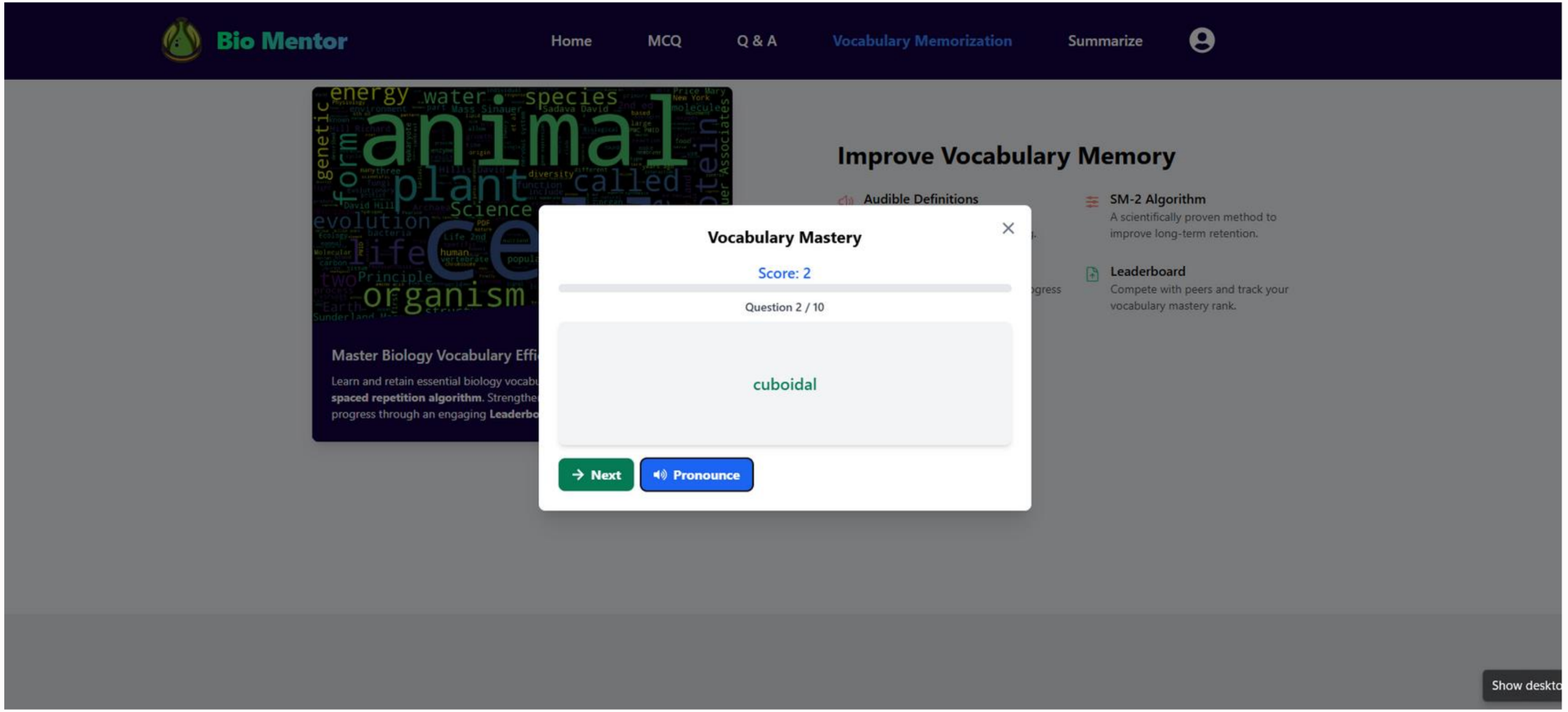
Levenshtein Distance

```python
from Levenshtein import distance

def word_accuracy(original_word, entered_word):
    """
    Calculate the accuracy score between two words using Levenshtein distance.

    Parameters:
    - original_word (str): The correct word.
    - entered_word (str): The word entered by the user.

    Returns:
    - int: Accuracy score between 0 (worst) and 5 (best).
    """

    # Handle empty cases
    if not original_word or not entered_word:
        return 0  # If the answer is empty, give the lowest score

    # Normalize the words by converting to lowercase and stripping spaces
    original_word = original_word.lower().strip()
    entered_word = entered_word.lower().strip()

    # Calculate Levenshtein distance
    max_length = max(len(original_word), len(entered_word))

    # If both are empty after stripping, return max score
    if max_length == 0:
        return 5

    edit_distance = distance(original_word, entered_word)

    # Calculate similarity percentage
    similarity = (max_length - edit_distance) / max_length

    # Scale similarity to 0-5
    if similarity >= 0.95:
        return 5
    elif similarity >= 0.85:
        return 4
    elif similarity >= 0.70:
        return 3
    elif similarity >= 0.50:
        return 2
    elif similarity >= 0.30:
        return 1
    else:
        return 0  # Too different, return lowest score

# ✅ **Test Cases**
test_cases = [
    ("biology", "biology"),   # Expected: 5 (Exact match)
    ("biology", "biolog"),    # Expected: 4 (Small typo)
    ("biology", "biolgy"),    # Expected: 3 (One letter swap)
    ("biology", "bio"),       # Expected: 2 (Partial match)
    ("biology", "chemistry"), # Expected: 0 (Completely different)
    ("hello", ""),            # Expected: 0 (Empty answer)
    ("", ""),                 # Expected: 5 (Both empty)
]

for original, entered in test_cases:
    print(f"Original: {original}, Entered: {entered}, Score: {word_accuracy(original, entered)}")
```

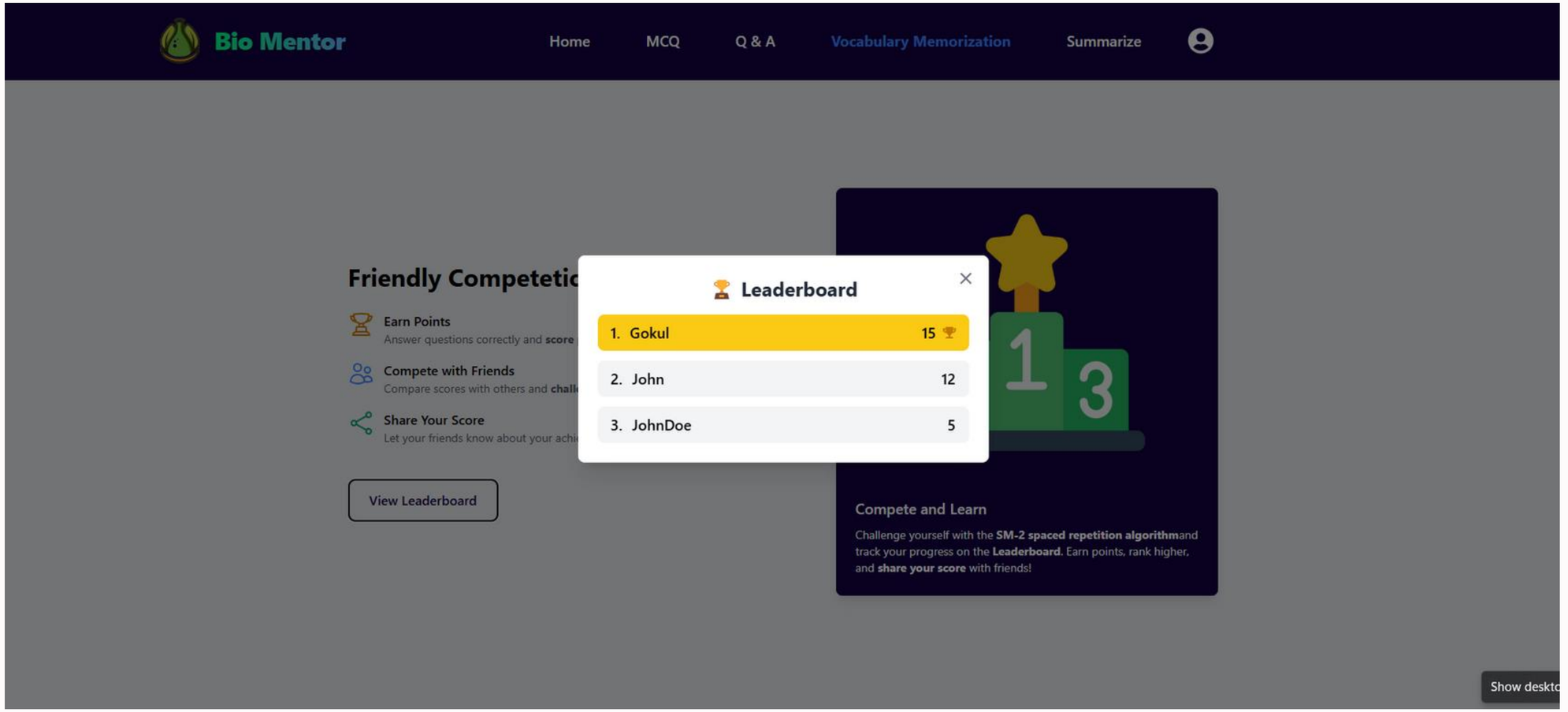
Working Product



Working Product



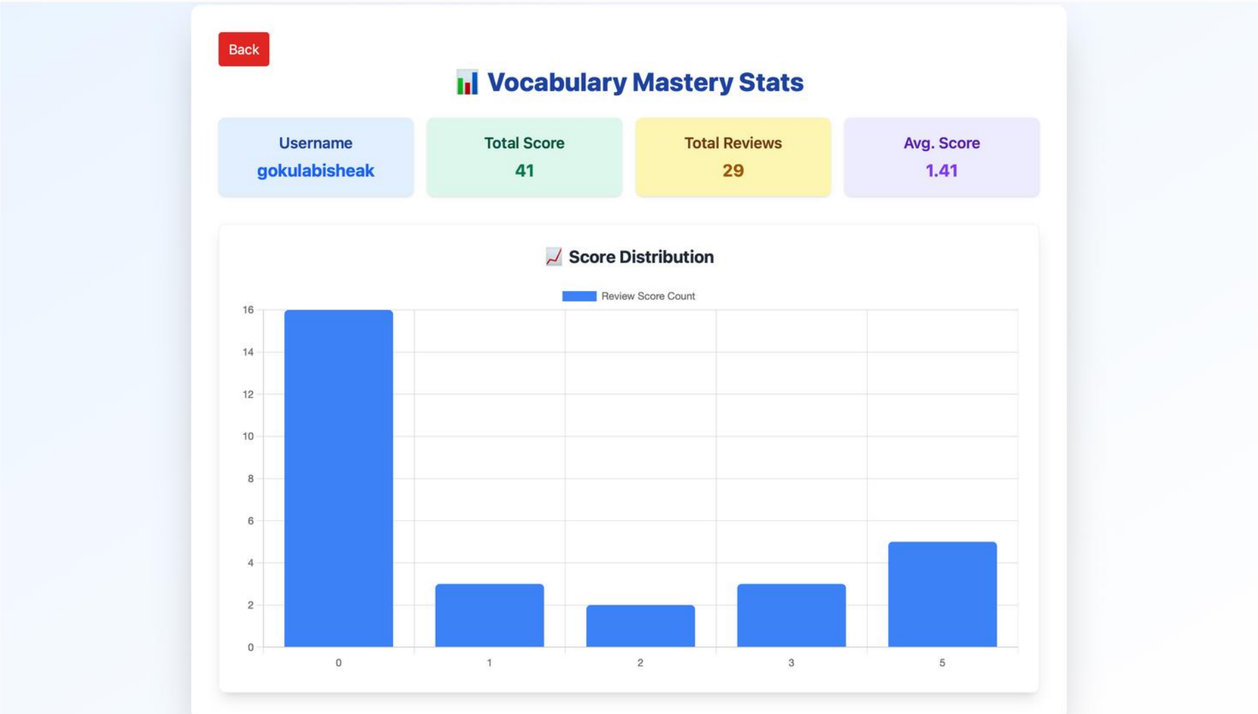
Working Product



Completion of the project

Project Enhancements

Progress Tracking Dashboard - Implemented a progress tracking dashboard for users to track their progress and to view statistics about their performance.



Testing - Unit Testing

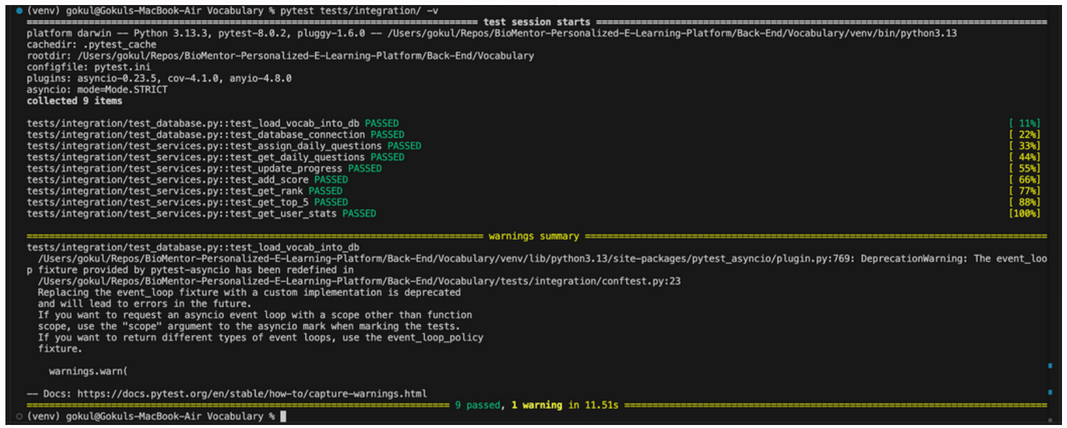
- **Framework:** Pytest

```
============================= test session starts ==============================
platform darwin -- Python 3.13.3, pytest-8.0.2, pluggy-1.6.0 -- /Users/gokul/Repos/BioMentor-Personalized-E-Learning-Platform/Back-End/Vocabulary/venv/bin/python3.13
cachedir: .pytest_cache
rootdir: /Users/gokul/Repos/BioMentor-Personalized-E-Learning-Platform/Back-End/Vocabulary
configfile: pytest.ini
plugins: asyncio-0.23.5, cov-4.1.0, anyio-4.8.0
asyncio: mode=Mode.STRICT
collected 4 items

tests/unit/test_models.py::test_flashcard_model PASSED                   [ 25%]
tests/unit/test_models.py::test_user_progress_model PASSED               [ 50%]
tests/unit/test_models.py::test_user_model PASSED                        [ 75%]
tests/unit/test_models.py::test_leaderboard_entry_model PASSED           [100%]

============================== warnings summary ===============================
venv/lib/python3.13/site-packages/pydantic/_internal/_config.py:295
venv/lib/python3.13/site-packages/pydantic/_internal/_config.py:295
venv/lib/python3.13/site-packages/pydantic/_internal/_config.py:295
venv/lib/python3.13/site-packages/pydantic/_internal/_config.py:295
  /Users/gokul/Repos/BioMentor-Personalized-E-Learning-Platform/Back-End/Vocabulary/venv/lib/python3.13/site-packages/pydantic/_internal/_config.py:295: PydanticDeprecatedSince20: S
upport for class-based `config` is deprecated, use ConfigDict instead. Deprecated in Pydantic V2.0 to be removed in V3.0. See Pydantic V2 Migration Guide at https://errors.pydantic.
dev/2.10/migration/
    warnings.warn(DEPRECATION_MESSAGE, DeprecationWarning)

venv/lib/python3.13/site-packages/pydantic/_internal/_config.py:345
  /Users/gokul/Repos/BioMentor-Personalized-E-Learning-Platform/Back-End/Vocabulary/venv/lib/python3.13/site-packages/pydantic/_internal/_config.py:345: UserWarning: Valid config ke
ys have changed in V2:
  * 'allow_population_by_field_name' has been renamed to 'populate_by_name'
    warnings.warn(message, UserWarning)

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
======================== 4 passed, 5 warnings in 0.07s =========================
```

```python
import pytest
from datetime import datetime
from bson import ObjectId

from models import Flashcard, UserProgress, User, LeaderboardEntry

def test_flashcard_model():
    """Test Flashcard model creation and validation."""
    # Test with string ID
    flashcard = Flashcard(
        _id="507f1f77bcf86cd799439011",
        question="What is photosynthesis?",
        answer="Process by which plants make food"
    )
    assert flashcard.id == "507f1f77bcf86cd799439011"
    assert flashcard.question == "What is photosynthesis?"
    assert flashcard.answer == "Process by which plants make food"

    # Test with ObjectId
    obj_id = ObjectId()
    flashcard = Flashcard(
        _id=str(obj_id),  # Convert ObjectId to string
        question="What is mitosis?",
        answer="Cell division process"
    )
    assert flashcard.id == str(obj_id)

def test_user_progress_model():
    """Test UserProgress model creation and validation."""
    progress = UserProgress(
        _id="507f1f77bcf86cd799439011",
        total_score=100,
        history=[{"question_id": "123", "score": 5}]
    )
    assert progress.id == "507f1f77bcf86cd799439011"
    assert progress.total_score == 100
    assert len(progress.history) == 1
    assert progress.history[0]["question_id"] == "123"

def test_user_model():
    """Test User model creation and validation."""
    user = User(
        _id="507f1f77bcf86cd799439011",
        username="test_user",
        total_score=150,
        history=[{
            "question_id": "123",
            "times_seen": 2,
            "review_score": 4,
            "next_review": "2024-03-20"
        }]
    )
    assert user.id == "507f1f77bcf86cd799439011"
    assert user.username == "test_user"
    assert user.total_score == 150
    assert len(user.history) == 1
    assert user.history[0]["times_seen"] == 2

def test_leaderboard_entry_model():
    """Test LeaderboardEntry model creation and validation."""
    entry = LeaderboardEntry(
        _id="507f1f77bcf86cd799439011",
        username="top_user",
        score=1000
    )
    assert entry.id == "507f1f77bcf86cd799439011"
    assert entry.username == "top_user"
    assert entry.score == 1000
```

Testing - Integration Testing

- **Framework:** Pytest with asyncio

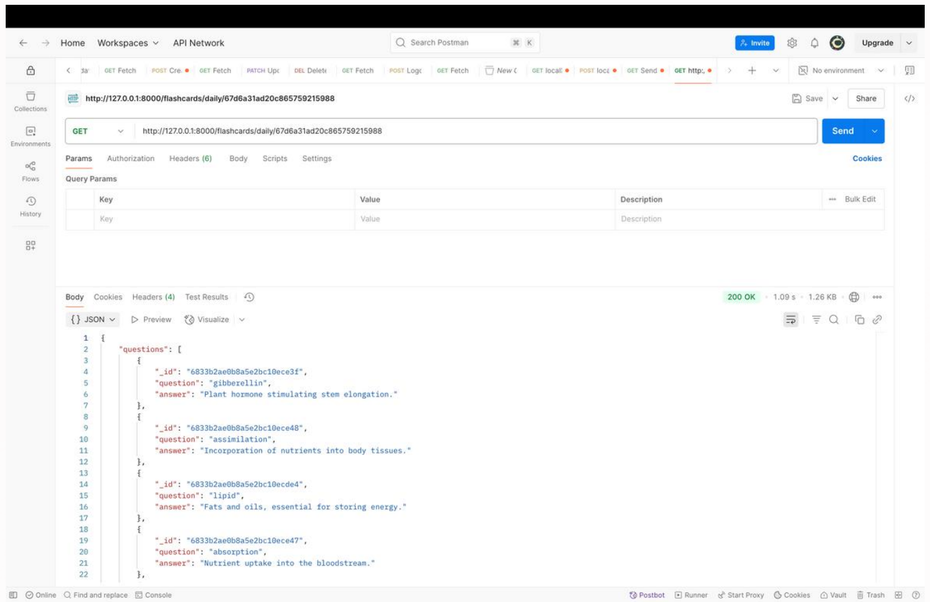
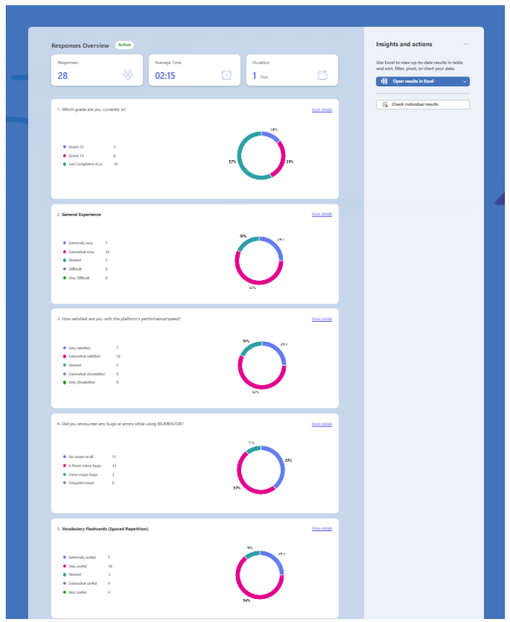


```
(venv) gokul@Gokuls-MacBook-Air Vocabulary % pytest tests/integration/ -v
============================= test session starts ==============================
platform darwin -- Python 3.13.3, pytest-8.0.2, pluggy-1.6.0 -- /Users/gokul/Repos/BioMentor-Personalized-E-Learning-Platform/Back-End/Vocabulary/venv/bin/python3.13
cachedir: .pytest_cache
rootdir: /Users/gokul/Repos/BioMentor-Personalized-E-Learning-Platform/Back-End/Vocabulary
configfile: pytest.ini
plugins: asyncio-0.23.5, cov-4.1.0, anyio-4.8.0
asyncio: mode=Mode.STRICT
collected 9 items

tests/integration/test_database.py::test_load_vocab_into_db PASSED                [ 11%]
tests/integration/test_database.py::test_database_connection PASSED               [ 22%]
tests/integration/test_services.py::test_assign_daily_questions PASSED            [ 33%]
tests/integration/test_services.py::test_get_daily_questions PASSED               [ 44%]
tests/integration/test_services.py::test_update_progress PASSED                   [ 55%]
tests/integration/test_services.py::test_add_score PASSED                         [ 66%]
tests/integration/test_services.py::test_get_rank PASSED                          [ 77%]
tests/integration/test_services.py::test_get_top_5 PASSED                         [ 88%]
tests/integration/test_services.py::test_get_user_stats PASSED                    [100%]

============================== warnings summary ===============================
tests/integration/test_database.py::test_load_vocab_into_db
  /Users/gokul/Repos/BioMentor-Personalized-E-Learning-Platform/Back-End/Vocabulary/venv/lib/python3.13/site-packages/pytest_asyncio/plugin.py:769: DeprecationWarning: The event_loop fixture provided by pytest-asyncio has been redefined in
  /Users/gokul/Repos/BioMentor-Personalized-E-Learning-Platform/Back-End/Vocabulary/tests/integration/conftest.py:23
  Replacing the event_loop fixture with a custom implementation is deprecated
  and will lead to errors in the future.
  If you want to request an asyncio event loop with a scope other than function
  scope, use the "scope" argument to the asyncio mark when marking the tests.
  If you want to return different types of event loops, use the event_loop_policy
  fixture.

    warnings.warn(

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
======================== 9 passed, 1 warning in 11.51s =========================
(venv) gokul@Gokuls-MacBook-Air Vocabulary %
```

```python
import pytest
from datetime import datetime, timedelta
from bson import ObjectId

from database import db
from services import (
    assign_daily_questions,
    get_daily_questions,
    update_progress,
    add_score,
    get_rank,
    get_top_5,
    get_user_stats
)

pytestmark = pytest.mark.asyncio

@pytest.fixture
async def setup_test_data():
    """Set up test data for services tests."""
    # Clear existing data
    await db["flashcards"].delete_many({})
    await db["users"].delete_many({})
    await db["leaderboard"].delete_many({})

    # Insert test flashcards
    test_flashcards = [
        {
            "_id": ObjectId(),
            "question": "What is photosynthesis?",
            "answer": "Process by which plants make food"
        },
        {
            "_id": ObjectId(),
            "question": "What is mitosis?",
            "answer": "Cell division process"
        }
    ]
    await db["flashcards"].insert_many(test_flashcards)

    # Insert test user
    test_user = {
        "_id": ObjectId(),
        "username": "test_user",
        "total_score": 0,
        "history": []
    }
    await db["users"].insert_one(test_user)

    return str(test_user["_id"])

async def test_assign_daily_questions():
    """Test assigning daily questions."""
    questions = await assign_daily_questions()
    assert len(questions) == 10
    assert all("_id" in q for q in questions)
    assert all("question" in q for q in questions)
    assert all("answer" in q for q in questions)

async def test_get_daily_questions(setup_test_data):
    """Test getting daily questions for a user."""
    user_id = await setup_test_data
    result = await get_daily_questions(user_id)
    assert "questions" in result
    assert len(result["questions"]) <= 10

```

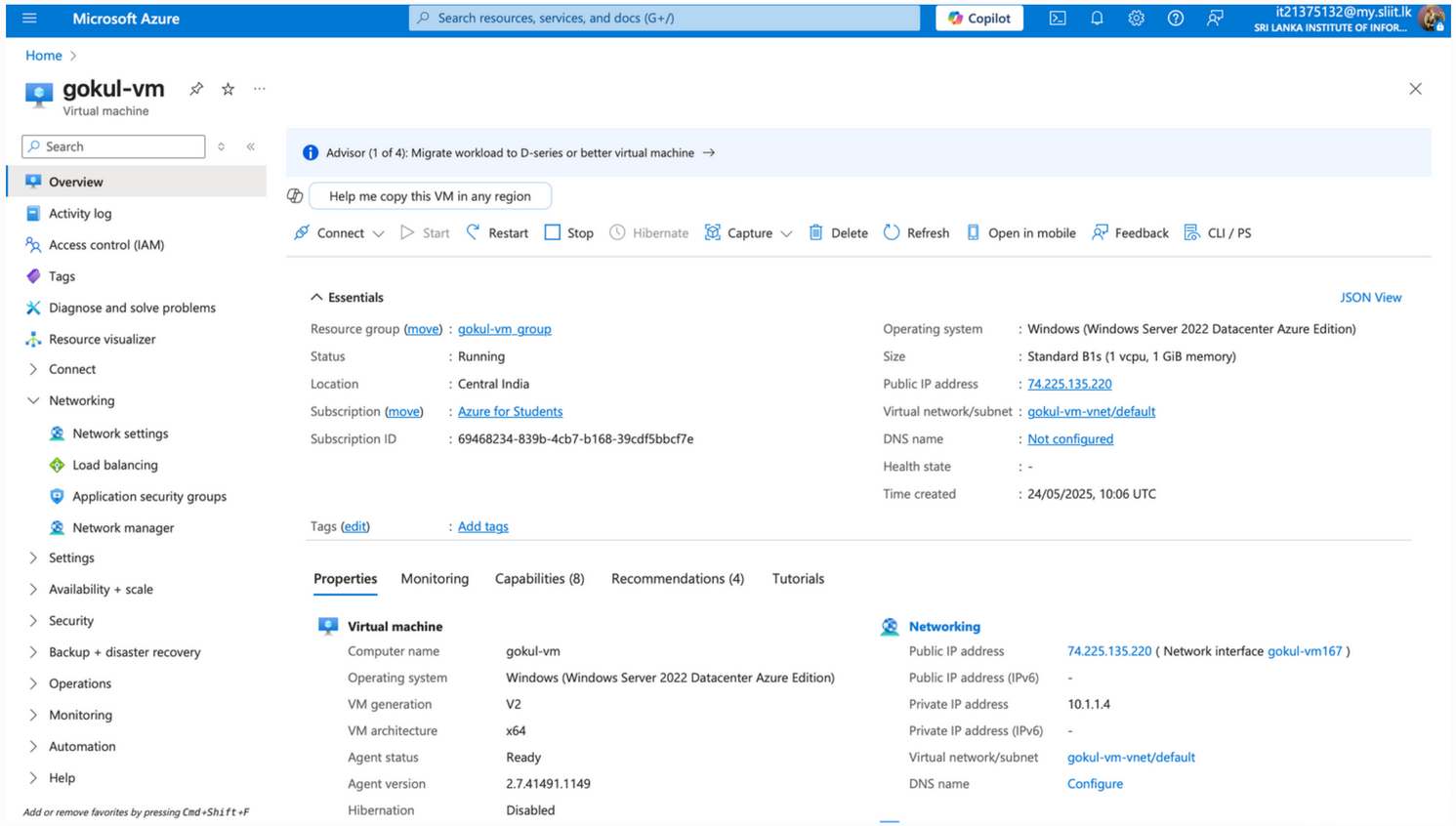
```python
async def test_update_progress(setup_test_data):
    """Test updating user progress."""
    user_id = await setup_test_data
    question_id = str((await db["flashcards"].find_one())["_id"])

    result = await update_progress(user_id, question_id, "test answer")
    assert "review_score" in result
    assert "next_review" in result

    # Verify user history was updated
    user = await db["users"].find_one({"_id": ObjectId(user_id)})
    assert len(user["history"]) == 1
    assert user["history"][0]["question_id"] == question_id

async def test_add_score():
    """Test adding a score to the leaderboard."""
    # Clear leaderboard first
    await db["leaderboard"].delete_many({})

    result = await add_score("test_user", 100)
    assert "rank" in result
    assert "top_5" in result

    # Verify score was added
    entry = await db["leaderboard"].find_one({"username": "test_user"})
    assert entry is not None
    assert entry["score"] == 100

async def test_get_rank():
    """Test getting user rank."""
    # Clear leaderboard first
    await db["leaderboard"].delete_many({})

    # Add multiple scores
    await add_score("user1", 100)
    await add_score("user2", 200)

    result = await get_rank("user1")
    assert "rank" in result
    assert "top_5" in result
    assert result["rank"] == 2  # Second place

async def test_get_top_5():
    """Test getting top 5 scores."""
    # Clear leaderboard first
    await db["leaderboard"].delete_many({})

    # Add multiple scores
    await add_score("user1", 100)
    await add_score("user2", 200)
    await add_score("user3", 300)
    await add_score("user4", 400)
    await add_score("user5", 500)
    await add_score("user6", 600)

    result = await get_top_5()
    assert "top_5" in result
    assert len(result["top_5"]) == 5
    assert result["top_5"][0]["score"] == 600  # Highest score first

async def test_get_user_stats(setup_test_data):
    """Test getting user statistics."""
    user_id = await setup_test_data

    # Add some review history
    question_id = str((await db["flashcards"].find_one())["_id"])
    await update_progress(user_id, question_id, "test answer")

    stats = await get_user_stats(user_id)
    assert "user_id" in stats
    assert "username" in stats
    assert "total_score" in stats
    assert "total_reviews" in stats
    assert "average_score" in stats
    assert "score_distribution" in stats
    assert "review_schedule" in stats
```

Testing - System Testing and UAT

System Testing	User Acceptance Testing
• **Framework:** Postman 	

Deployment

- Backend deployed on Azure VM (Windows - Standard),Region: Central India for low latency



References

[1] J. Su, J. Ye, L. Nie, Y. Cao and Y. Chen, "Optimizing Spaced Repetition Schedule by Capturing the Dynamics of Memory," in IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 10, pp. 10085-10097, 1 Oct. 2023, doi: 10.1109/TKDE.2023.3251721.

[2] F. Schimanke, R. Mertens and B. S. Huck, "Retrieval of Relevant Data for Measuring the Impact of Spaced-Repetition Algorithms on the Learning Success in Mobile Learning Games," 2019 IEEE International Symposium on Multimedia (ISM), San Diego, CA, USA, 2019, pp. 279-2795, doi: 10.1109/ISM46123.2019.00063.

[3] G. RANDAZZO, “Memory models for spaced repetition systems,” Polimi.it, Sep. 2023, doi: http://hdl.handle.net/10589/186407.

[4] F. Schimanke, "The Impact of Spaced Repetition Learning on the Learning Success in Mobile Learning Games," 2021 IEEE International Symposium on Multimedia (ISM), Naple, Italy, 2021, pp. 275-280, doi: 10.1109/ISM52913.2021.00054.

IT21068478

Dharane.S
Software Engineering

LLM BASED ABSTRACTIVE TEXT SUMMARIZATION TOOL WITH VOICE OUTPUT IMPLEMENTED IN DIFFERENT SOFTWARE ARCHITECTURES

Introduction

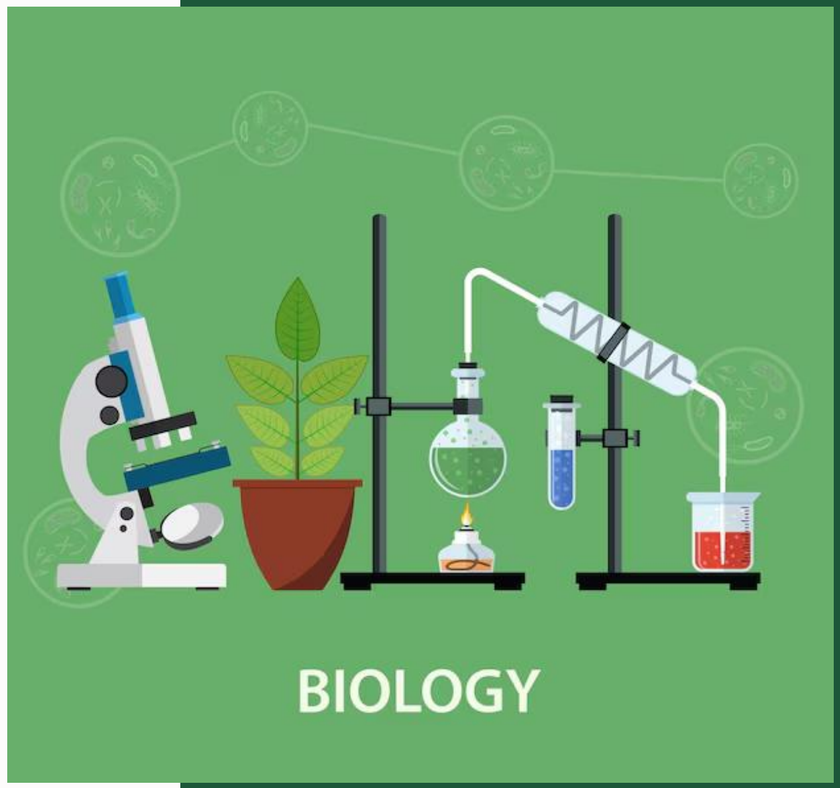
01 Background

02 Research Problem

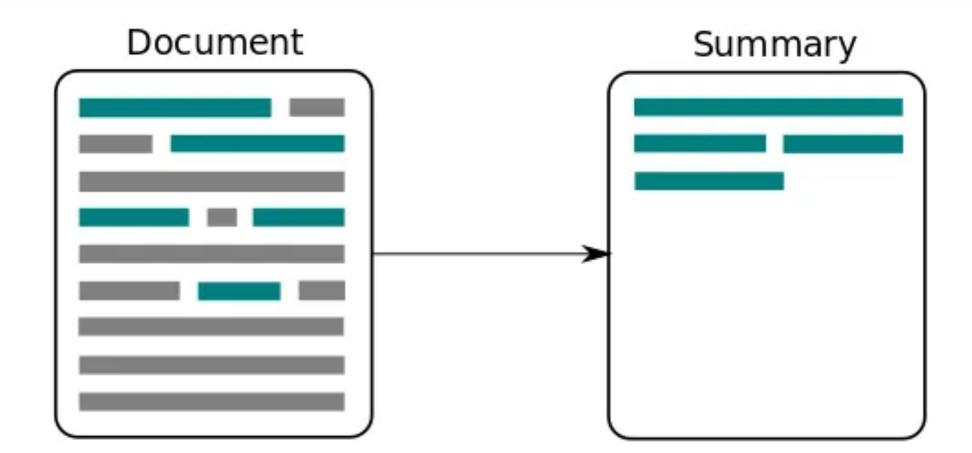
03 Main and Sub Objectives

04 Methodology

05 Completion Of Project



BACKGROUND



What is summarization?

What is NLP?

RESEARCH PROBLEM

01

How can text summarization be optimized to provide concise, exam-focused summaries for A/L biology students?

How can the integration of voice output features with customizable word counts in an e-learning summarization tool enhance accessibility, catering to different learning preferences and needs?

02

OBJECTIVES

Objective 1
Ensure Accuracy: Use government-approved resources to maintain content accuracy and educational standards.

Objective 3
Ensure Content Relevance: Implement filtering mechanisms to detect and eliminate irrelevant content

Main Objective

Create a tool that extracts and summarizes key concepts from A/L biology resources based on a given topic, and generates concise summaries from uploaded text documents, specifically to aid students preparing for A/L exams.

Objective 2
Enhance Accessibility: Integrate a voice output feature to provide audible summaries and provide basic multi-lingual support for notes.

Objective 4
Various implementation: Implement the component in various architectures, compare, and analyze their strengths and weaknesses.

IT21068478 |Dharane. S | 24-25J-257

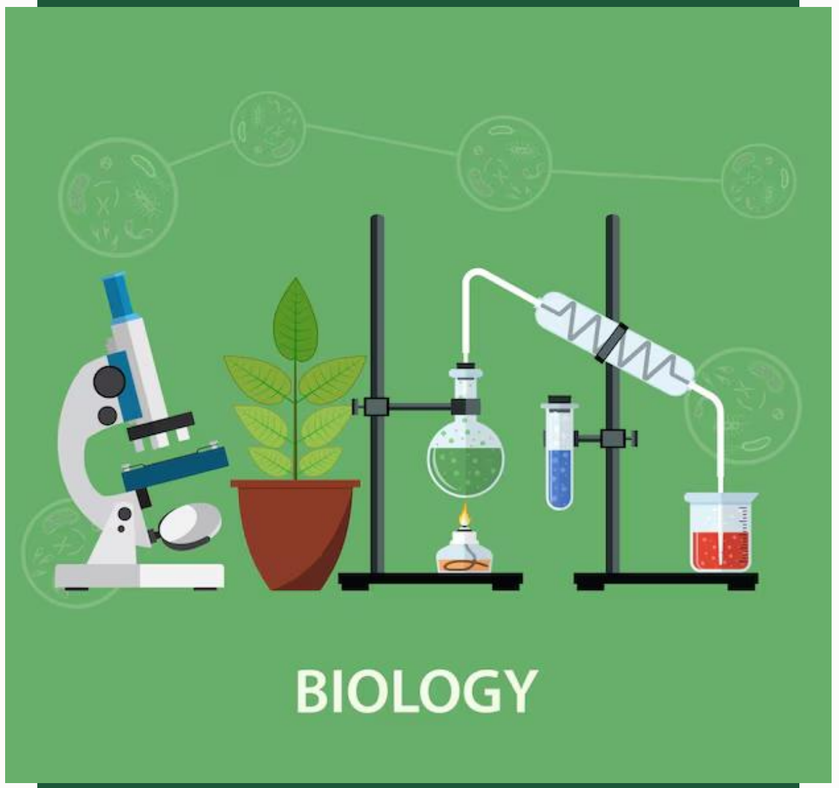
Methodology

01 System Diagram

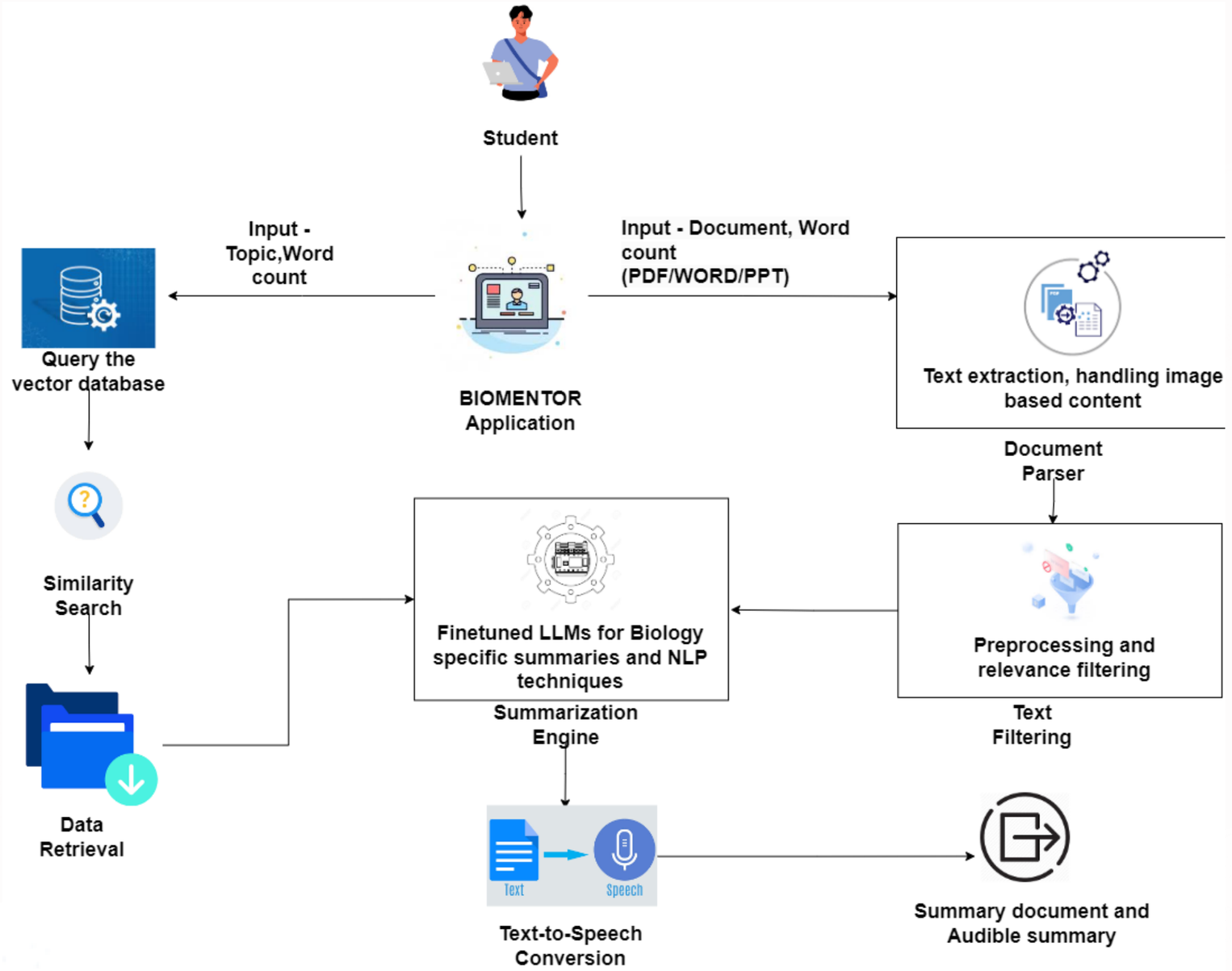
02 Tools and Technologies

03 Best Practices

04 GitHub Commits



System Diagram



Tools & Technologies

Programming Languages

Python
React Js

Database

Faiss
Mongo DB

Project Management

Jira

Other tools

Git
Draw.io
Postman

Frameworks

Transformer model
pandas
Pytorch
gtts
Pytesseract
Deep Translator
FastAPI
numpy

SLIIT
FACULTY OF COMPUTING

IT21068478 |Dharane. S | 24-25J-257

Best Practices

Non-Functional Requirements

- Compatibility
- Accuracy
- Performance
- Usability
- Availability

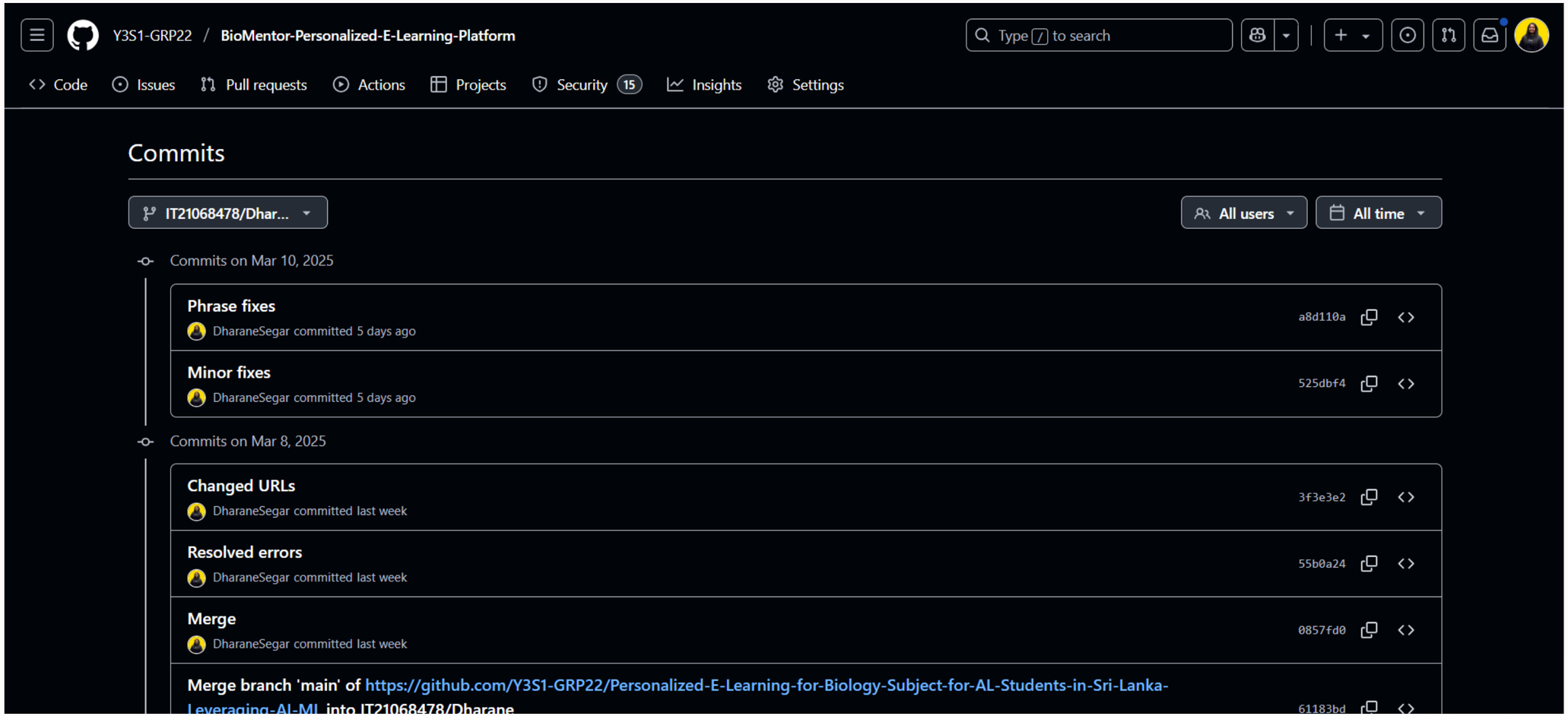
Design Excellence

- Real-Time Performance
- Accuracy and Reliability
- User Experience
- Integration

Standards & Best Practices

- Code Structure & Organization
- Web Security
- Model Quality
- RESTful API Design

GitHub Commits



Completion of the project

SLIIT
FACULTY OF COMPUTING

IT21068478 |Dharane. S | 24-25J-257

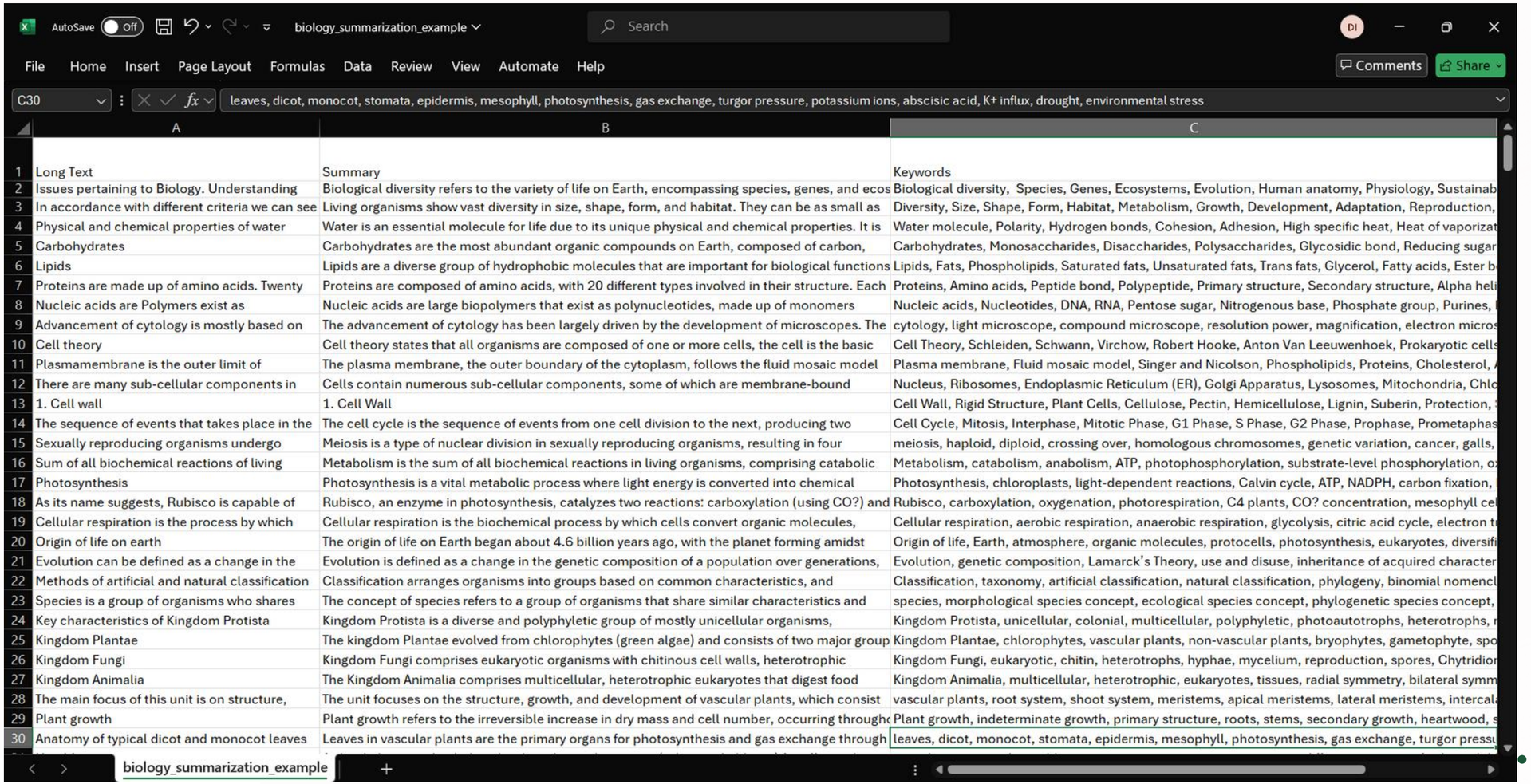
Completion of the project

SLIIT
FACULTY OF COMPUTING

IT21068478 |Dharane. S | 24-25J-257

Data Collection

Summarization Dataset

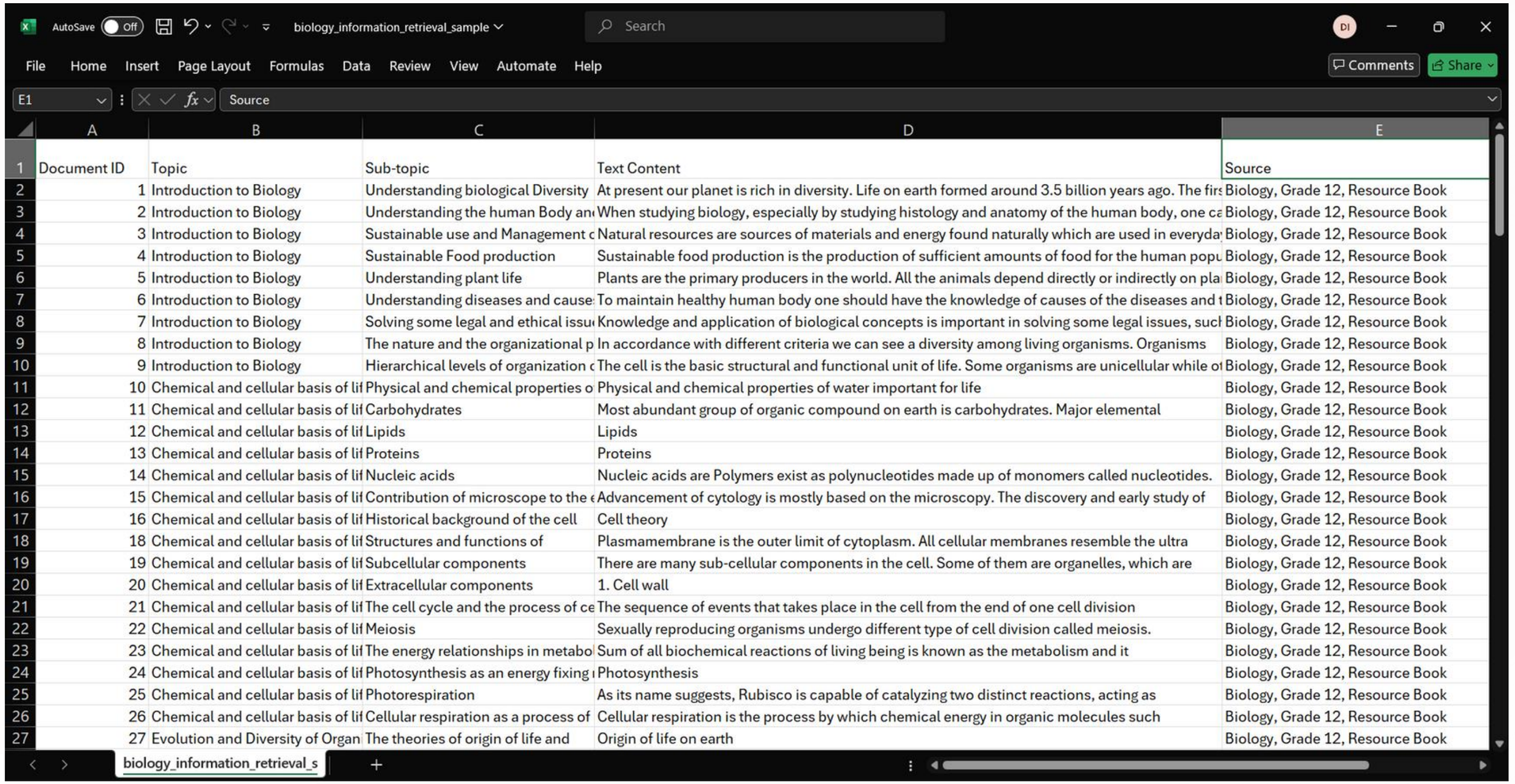


biology_summarization_example

	A	B	C
1	Long Text	Summary	Keywords
2	Issues pertaining to Biology. Understanding	Biological diversity refers to the variety of life on Earth, encompassing species, genes, and ecos	Biological diversity, Species, Genes, Ecosystems, Evolution, Human anatomy, Physiology, Sustainab
3	In accordance with different criteria we can see	Living organisms show vast diversity in size, shape, form, and habitat. They can be as small as	Diversity, Size, Shape, Form, Habitat, Metabolism, Growth, Development, Adaptation, Reproduction,
4	Physical and chemical properties of water	Water is an essential molecule for life due to its unique physical and chemical properties. It is	Water molecule, Polarity, Hydrogen bonds, Cohesion, Adhesion, High specific heat, Heat of vaporizat
5	Carbohydrates	Carbohydrates are the most abundant organic compounds on Earth, composed of carbon,	Carbohydrates, Monosaccharides, Disaccharides, Polysaccharides, Glycosidic bond, Reducing sugar
6	Lipids	Lipids are a diverse group of hydrophobic molecules that are important for biological functions	Lipids, Fats, Phospholipids, Saturated fats, Unsaturated fats, Trans fats, Glycerol, Fatty acids, Ester b
7	Proteins are made up of amino acids. Twenty	Proteins are composed of amino acids, with 20 different types involved in their structure. Each	Proteins, Amino acids, Peptide bond, Polypeptide, Primary structure, Secondary structure, Alpha heli
8	Nucleic acids are Polymers exist as	Nucleic acids are large biopolymers that exist as polynucleotides, made up of monomers	Nucleic acids, Nucleotides, DNA, RNA, Pentose sugar, Nitrogenous base, Phosphate group, Purines,
9	Advancement of cytology is mostly based on	The advancement of cytology has been largely driven by the development of microscopes. The	cytology, light microscope, compound microscope, resolution power, magnification, electron micros
10	Cell theory	Cell theory states that all organisms are composed of one or more cells, the cell is the basic	Cell Theory, Schleiden, Schwann, Virchow, Robert Hooke, Anton Van Leeuwenhoek, Prokaryotic cells
11	Plasmamembrane is the outer limit of	The plasma membrane, the outer boundary of the cytoplasm, follows the fluid mosaic model	Plasma membrane, Fluid mosaic model, Singer and Nicolson, Phospholipids, Proteins, Cholesterol,
12	There are many sub-cellular components in	Cells contain numerous sub-cellular components, some of which are membrane-bound	Nucleus, Ribosomes, Endoplasmic Reticulum (ER), Golgi Apparatus, Lysosomes, Mitochondria, Chlo
13	1. Cell wall	1. Cell Wall	Cell Wall, Rigid Structure, Plant Cells, Cellulose, Pectin, Hemicellulose, Lignin, Suberin, Protection,
14	The sequence of events that takes place in the	The cell cycle is the sequence of events from one cell division to the next, producing two	Cell Cycle, Mitosis, Interphase, Mitotic Phase, G1 Phase, S Phase, G2 Phase, Prophase, Prometaphas
15	Sexually reproducing organisms undergo	Meiosis is a type of nuclear division in sexually reproducing organisms, resulting in four	meiosis, haploid, diploid, crossing over, homologous chromosomes, genetic variation, cancer, galls,
16	Sum of all biochemical reactions of living	Metabolism is the sum of all biochemical reactions in living organisms, comprising catabolic	Metabolism, catabolism, anabolism, ATP, photophosphorylation, substrate-level phosphorylation, o
17	Photosynthesis	Photosynthesis is a vital metabolic process where light energy is converted into chemical	Photosynthesis, chloroplasts, light-dependent reactions, Calvin cycle, ATP, NADPH, carbon fixation,
18	As its name suggests, Rubisco is capable of	Rubisco, an enzyme in photosynthesis, catalyzes two reactions: carboxylation (using CO?) and	Rubisco, carboxylation, oxygenation, photorespiration, C4 plants, CO? concentration, mesophyll ce
19	Cellular respiration is the process by which	Cellular respiration is the biochemical process by which cells convert organic molecules,	Cellular respiration, aerobic respiration, anaerobic respiration, glycolysis, citric acid cycle, electron t
20	Origin of life on earth	The origin of life on Earth began about 4.6 billion years ago, with the planet forming amidst	Origin of life, Earth, atmosphere, organic molecules, protocells, photosynthesis, eukaryotes, diversifi
21	Evolution can be defined as a change in the	Evolution is defined as a change in the genetic composition of a population over generations,	Evolution, genetic composition, Lamarck's Theory, use and disuse, inheritance of acquired character
22	Methods of artificial and natural classification	Classification arranges organisms into groups based on common characteristics, and	Classification, taxonomy, artificial classification, natural classification, phylogeny, binomial nomencl
23	Species is a group of organisms who shares	The concept of species refers to a group of organisms that share similar characteristics and	species, morphological species concept, ecological species concept, phylogenetic species concept,
24	Key characteristics of Kingdom Protista	Kingdom Protista is a diverse and polyphyletic group of mostly unicellular organisms,	Kingdom Protista, unicellular, colonial, multicellular, polyphyletic, photoautotrophs, heterotrophs, r
25	Kingdom Plantae	The kingdom Plantae evolved from chlorophytes (green algae) and consists of two major group	Kingdom Plantae, chlorophytes, vascular plants, non-vascular plants, bryophytes, gametophyte, spo
26	Kingdom Fungi	Kingdom Fungi comprises eukaryotic organisms with chitinous cell walls, heterotrophic	Kingdom Fungi, eukaryotic, chitin, heterotrophs, hyphae, mycelium, reproduction, spores, Chytridior
27	Kingdom Animalia	The Kingdom Animalia comprises multicellular, heterotrophic eukaryotes that digest food	Kingdom Animalia, multicellular, heterotrophic, eukaryotes, tissues, radial symmetry, bilateral symm
28	The main focus of this unit is on structure,	The unit focuses on the structure, growth, and development of vascular plants, which consist	vascular plants, root system, shoot system, meristems, apical meristems, lateral meristems, intercal
29	Plant growth	Plant growth refers to the irreversible increase in dry mass and cell number, occurring through	Plant growth, indeterminate growth, primary structure, roots, stems, secondary growth, heartwood, s
30	Anatomy of typical dicot and monocot leaves	Leaves in vascular plants are the primary organs for photosynthesis and gas exchange through	leaves, dicot, monocot, stomata, epidermis, mesophyll, photosynthesis, gas exchange, turgor pressu

Data Collection

Information Retrieval Dataset



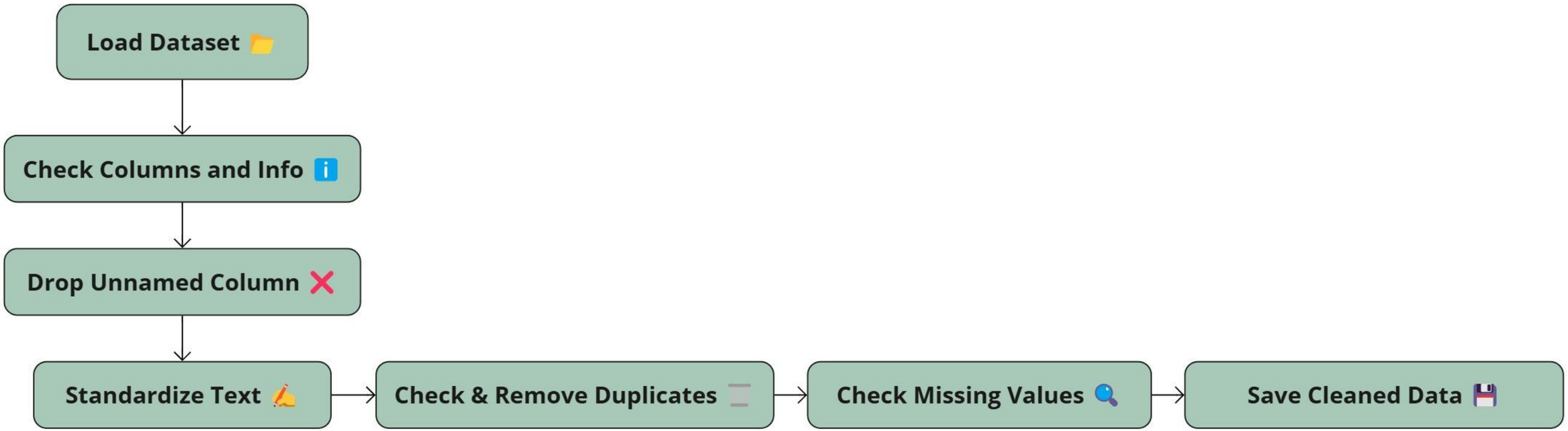
	A	B	C	D	E
1	Document ID	Topic	Sub-topic	Text Content	Source
2	1	Introduction to Biology	Understanding biological Diversity	At present our planet is rich in diversity. Life on earth formed around 3.5 billion years ago. The firs	Biology, Grade 12, Resource Book
3	2	Introduction to Biology	Understanding the human Body an	When studying biology, especially by studying histology and anatomy of the human body, one ca	Biology, Grade 12, Resource Book
4	3	Introduction to Biology	Sustainable use and Management c	Natural resources are sources of materials and energy found naturally which are used in everyda	Biology, Grade 12, Resource Book
5	4	Introduction to Biology	Sustainable Food production	Sustainable food production is the production of sufficient amounts of food for the human popu	Biology, Grade 12, Resource Book
6	5	Introduction to Biology	Understanding plant life	Plants are the primary producers in the world. All the animals depend directly or indirectly on pla	Biology, Grade 12, Resource Book
7	6	Introduction to Biology	Understanding diseases and cause	To maintain healthy human body one should have the knowledge of causes of the diseases and t	Biology, Grade 12, Resource Book
8	7	Introduction to Biology	Solving some legal and ethical issu	Knowledge and application of biological concepts is important in solving some legal issues, such	Biology, Grade 12, Resource Book
9	8	Introduction to Biology	The nature and the organizational p	In accordance with different criteria we can see a diversity among living organisms. Organisms	Biology, Grade 12, Resource Book
10	9	Introduction to Biology	Hierarchical levels of organization c	The cell is the basic structural and functional unit of life. Some organisms are unicellular while ot	Biology, Grade 12, Resource Book
11	10	Chemical and cellular basis of lif	Physical and chemical properties o	Physical and chemical properties of water important for life	Biology, Grade 12, Resource Book
12	11	Chemical and cellular basis of lif	Carbohydrates	Most abundant group of organic compound on earth is carbohydrates. Major elemental	Biology, Grade 12, Resource Book
13	12	Chemical and cellular basis of lif	Lipids	Lipids	Biology, Grade 12, Resource Book
14	13	Chemical and cellular basis of lif	Proteins	Proteins	Biology, Grade 12, Resource Book
15	14	Chemical and cellular basis of lif	Nucleic acids	Nucleic acids are Polymers exist as polynucleotides made up of monomers called nucleotides.	Biology, Grade 12, Resource Book
16	15	Chemical and cellular basis of lif	Contribution of microscope to the c	Advancement of cytology is mostly based on the microscopy. The discovery and early study of	Biology, Grade 12, Resource Book
17	16	Chemical and cellular basis of lif	Historical background of the cell	Cell theory	Biology, Grade 12, Resource Book
18	18	Chemical and cellular basis of lif	Structures and functions of	Plasmamembrane is the outer limit of cytoplasm. All cellular membranes resemble the ultra	Biology, Grade 12, Resource Book
19	19	Chemical and cellular basis of lif	Subcellular components	There are many sub-cellular components in the cell. Some of them are organelles, which are	Biology, Grade 12, Resource Book
20	20	Chemical and cellular basis of lif	Extracellular components	1. Cell wall	Biology, Grade 12, Resource Book
21	21	Chemical and cellular basis of lif	The cell cycle and the process of ce	The sequence of events that takes place in the cell from the end of one cell division	Biology, Grade 12, Resource Book
22	22	Chemical and cellular basis of lif	Meiosis	Sexually reproducing organisms undergo different type of cell division called meiosis.	Biology, Grade 12, Resource Book
23	23	Chemical and cellular basis of lif	The energy relationships in metabo	Sum of all biochemical reactions of living being is known as the metabolism and it	Biology, Grade 12, Resource Book
24	24	Chemical and cellular basis of lif	Photosynthesis as an energy fixing	Photosynthesis	Biology, Grade 12, Resource Book
25	25	Chemical and cellular basis of lif	Photorespiration	As its name suggests, Rubisco is capable of catalyzing two distinct reactions, acting as	Biology, Grade 12, Resource Book
26	26	Chemical and cellular basis of lif	Cellular respiration as a process of	Cellular respiration is the process by which chemical energy in organic molecules such	Biology, Grade 12, Resource Book
27	27	Evolution and Diversity of Organ	The theories of origin of life and	Origin of life on earth	Biology, Grade 12, Resource Book

Model Selection

Why Flan-T5-Base?

- Instruction-Following Capability
- Versatility in Fine-tuning
- Efficiency and Scalability
- Customizability
- Pre-trained Knowledge
- Resource Compatibility

Data pre-processing



Data pre-processing

DataPreprocessing.ipynb

DataPreprocessing.ipynb > # Step 3: Inspect the dataset

+ Code + Markdown | Run All Clear All Outputs | Outline ···

```
#Removing unwanted characters, space, special characters
import re

# Function to standardize text: convert to lowercase, remove irrelevant symbols
def standardize_text(text):
    # Convert to lowercase
    text = text.lower()
    # Remove irrelevant symbols (e.g., unusual bullet points, etc.)
    text = re.sub(r'[¦]', '', text)
    # Remove any remaining special characters (e.g., non-alphanumeric symbols)
    text = re.sub(r'[^a-zA-Z0-9\s]', '', text)
    # Remove extra whitespace
    text = re.sub(r'\s+', ' ', text).strip()
    return text

# Apply the function to standardize the 'Long Text' and 'Summary' columns
data['Long Text'] = data['Long Text'].apply(standardize_text)
data['Summary'] = data['Summary'].apply(standardize_text)

# Display a sample to verify the standardization process
data[['Long Text', 'Summary']].head()
```

	Long Text	Summary
0	issues pertaining to biology understanding bio...	biological diversity refers to the variety of ...
1	in accordance with different criteria we can s...	living organisms show vast diversity in size s...
2	physical and chemical properties of water impo...	water is an essential molecule for life due to...
3	carbohydrates most abundant group of organic c...	carbohydrates are the most abundant organic co...
4	lipids diverse group of hydrophobic molecules ...	lipids are a diverse group of hydrophobic mole...

```
# Checking and removing duplicates based on specific columns 'Long Text' and 'Summary' only, which are the main content columns
data_cleaned = data.drop_duplicates(subset=['Long Text', 'Summary'])

# Display the number of rows before and after removing duplicates
original_count = data.shape[0]
cleaned_count = data_cleaned.shape[0]

original_count, cleaned_count
```

[18]

DataPreprocessing.ipynb

DataPreprocessing.ipynb > # Step 3: Inspect the dataset

+ Code + Markdown | Run All Clear All Outputs | Outline ···

```
# Checking for missing values in the essential columns 'Long Text' and 'Summary'
missing_long_text = data['Long Text'].isnull().sum()
missing_summary = data['Summary'].isnull().sum()

# Displaying the count of missing values in each essential column
missing_long_text, missing_summary
```

[19]

··· (0, 0)

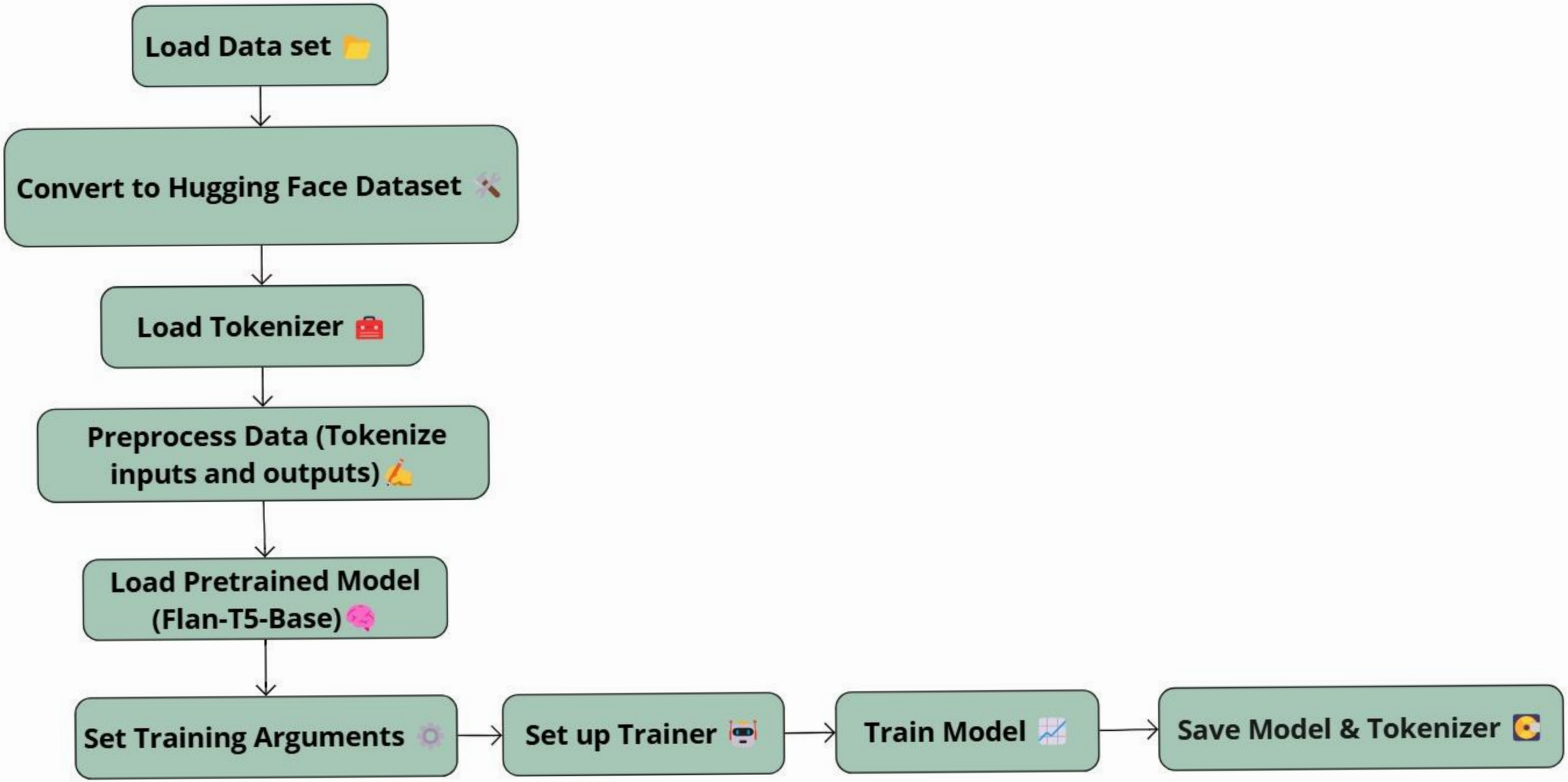
```
# Saving the cleaned dataset to a new CSV file named 'bio_summary.csv'
output_path = 'D:/Downloads/RP/Summarization/Summary_Description/bio_summary.csv'
data_cleaned.to_csv(output_path, index=False)

output_path
```

··· 'C:/Users/dinon/Desktop/Summary/bio_summary.csv'

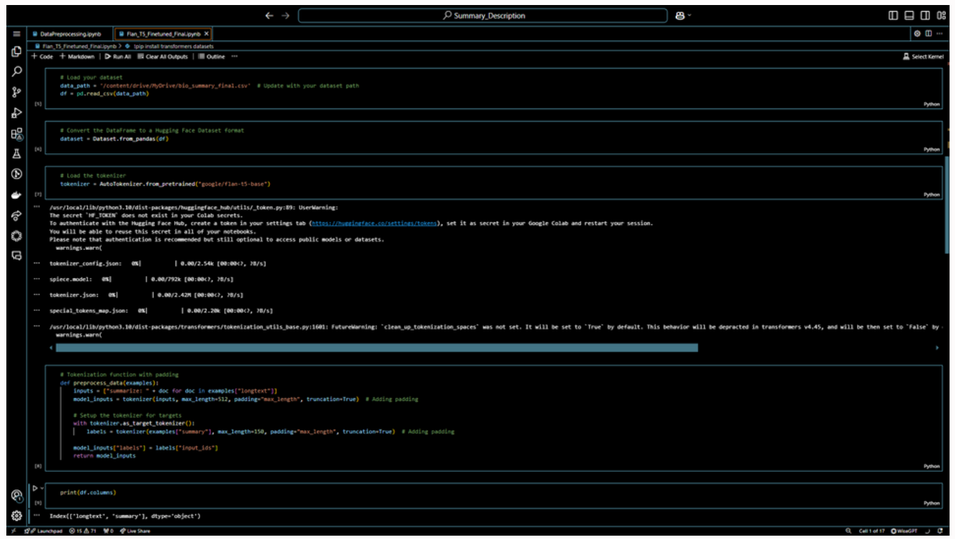
```
# Selecting only 'Long Text' and 'Summary' columns and renaming the headers to lowercase
data_final = data_cleaned[['Long Text', 'Summary']].rename(columns={'Long Text': 'long text', 'Summary': 'summary'})

# Saving this final cleaned dataset to a new CSV file
output_path_final = 'D:/Downloads/RP/Summarization/Summary_Description/bio_summary_final.csv'
data_final.to_csv(output_path_final, index=False)

output_path_final
```

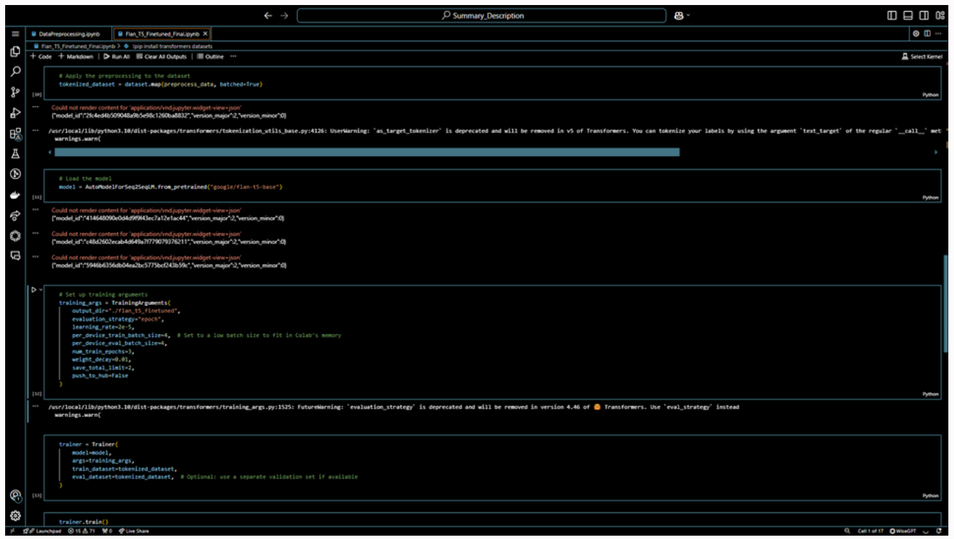
··· 'C:/Users/dinon/Desktop/Summary/bio_summary_final.csv'

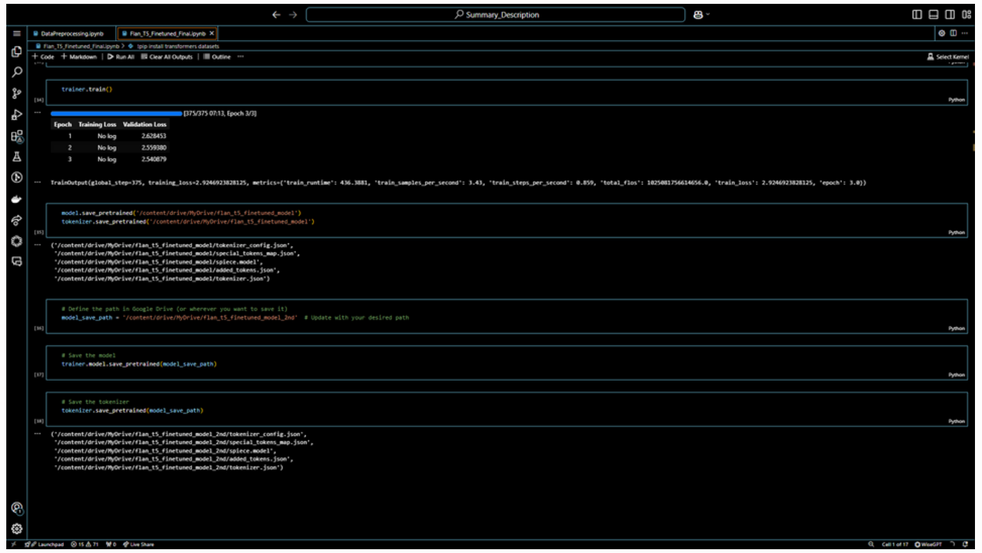
Fine-tuning the LLM



Fine-tuning the LLM







SLIIT
FACULTY OF COMPUTING

IT21068478 |Dharane. S | 24-25J-257

Evaluate the Fine-tuned model

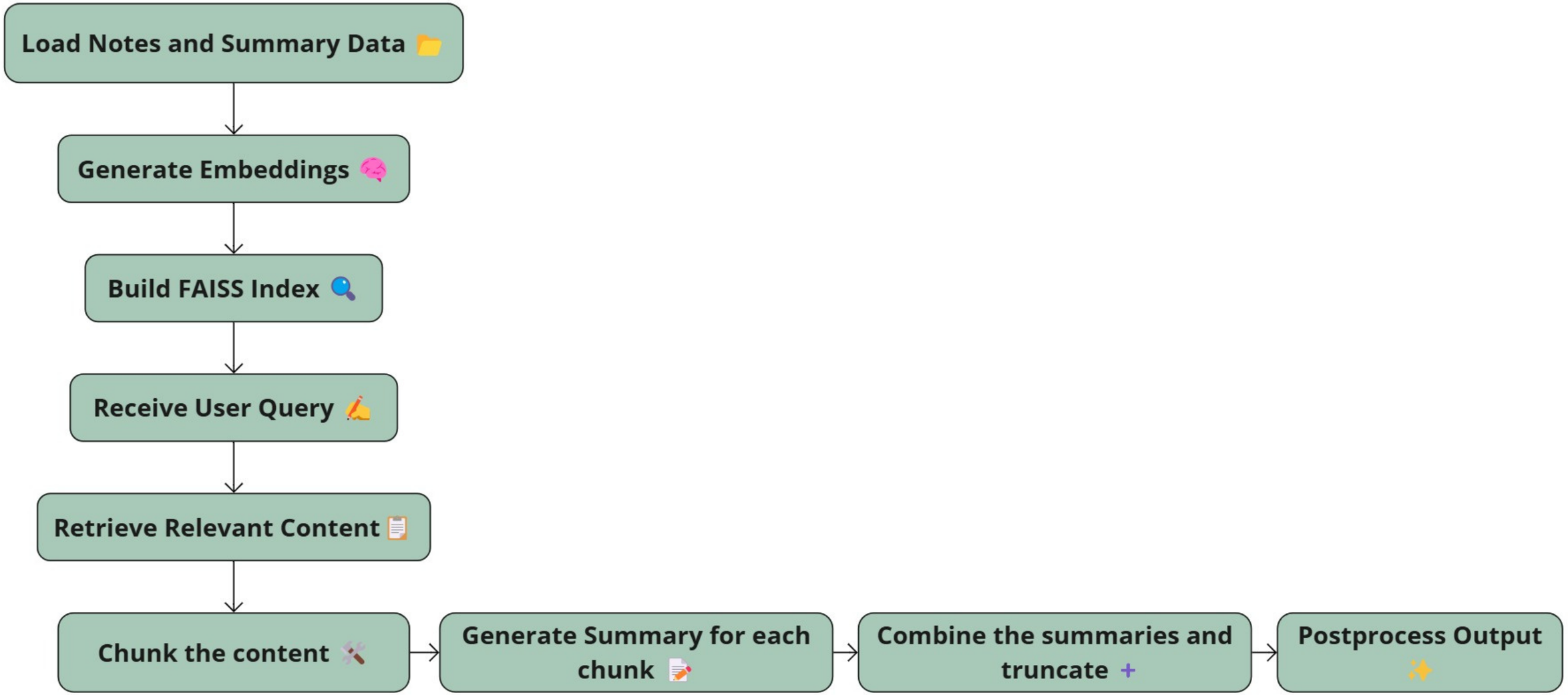
Evaluation Metric: ROUGE(Recall-Oriented Understudy for Gisting Evaluation)

- Measures word overlap between the generated summary and the reference summary.
- Types of ROUGE Scores
  - ROUGE-1: Unigram (single word) overlap.
  - ROUGE-2: Bigram (two-word) overlap.
  - ROUGE-L: Longest Common Subsequence (LCS) match.

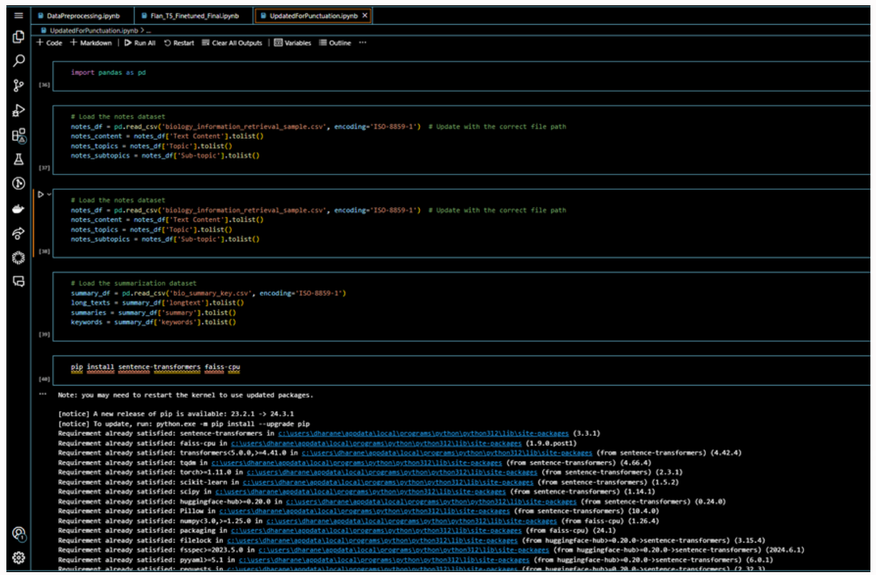
Evaluate the Fine-tuned model

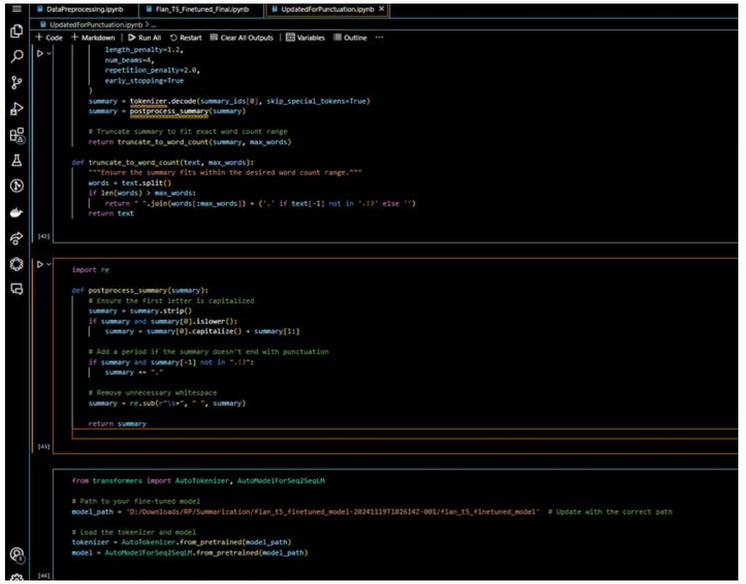
{'**rouge1**': Score(**precision=0.7414965986394558**, recall=0.5989010989010989, fmeasure=0.662613981762917 9), '**rouge2**': Score(**precision=0.4041095890410959**, recall=0.3259668508287293, fmeasure=0.3608562691131499), '**rougeL**': Score(**precision=0.5442176870748 3**, recall=0.4395604395604395 5, fmeasure=0.4863221884498480 4)}

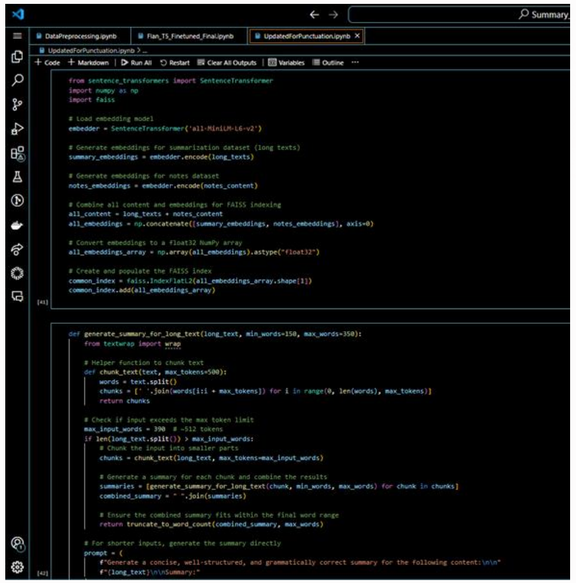
RAG Implementation



RAG Implementation







Completion of the project

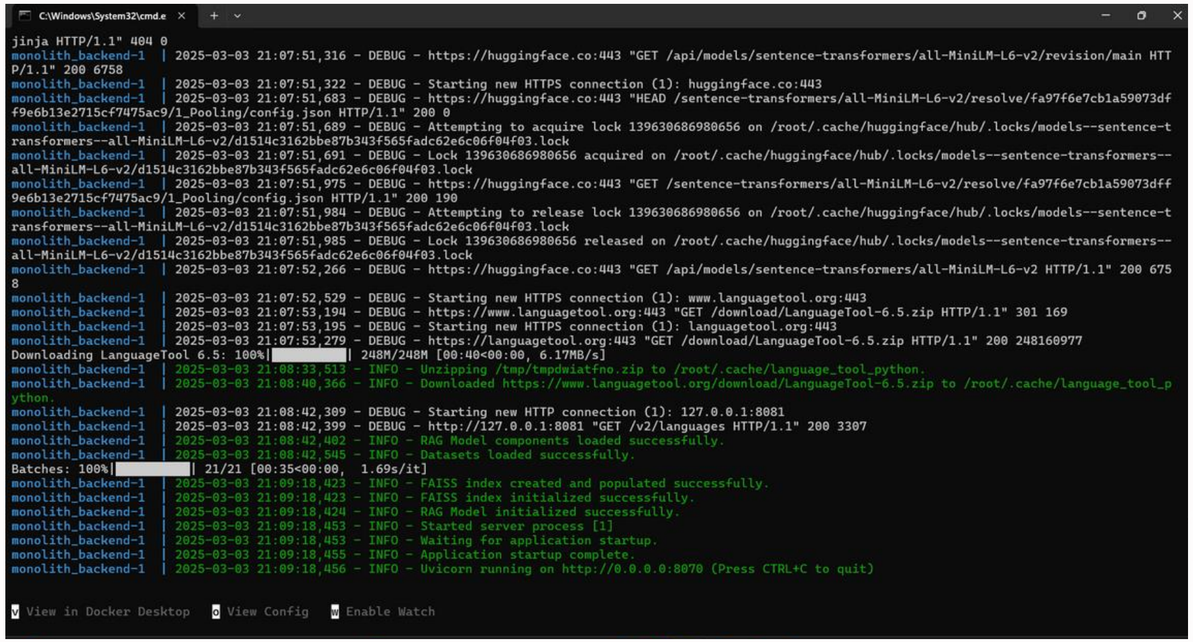
SLIIT
FACULTY OF COMPUTING

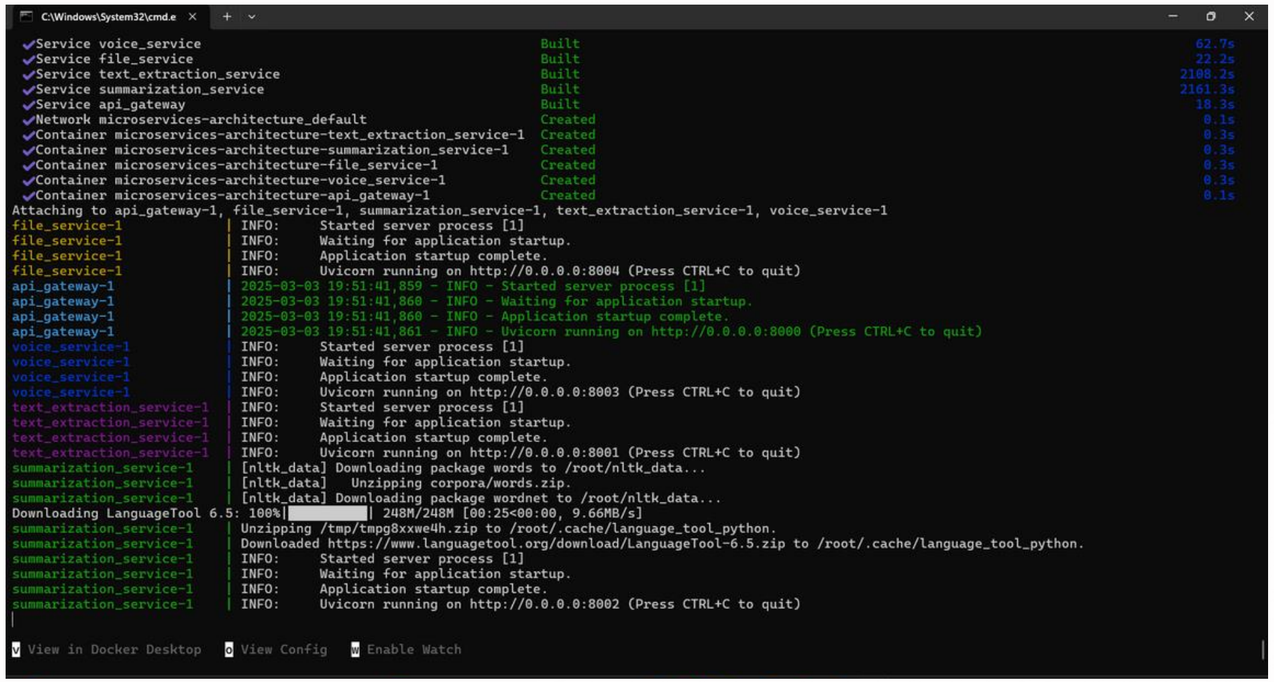
IT21068478 |Dharane. S | 24-25J-257

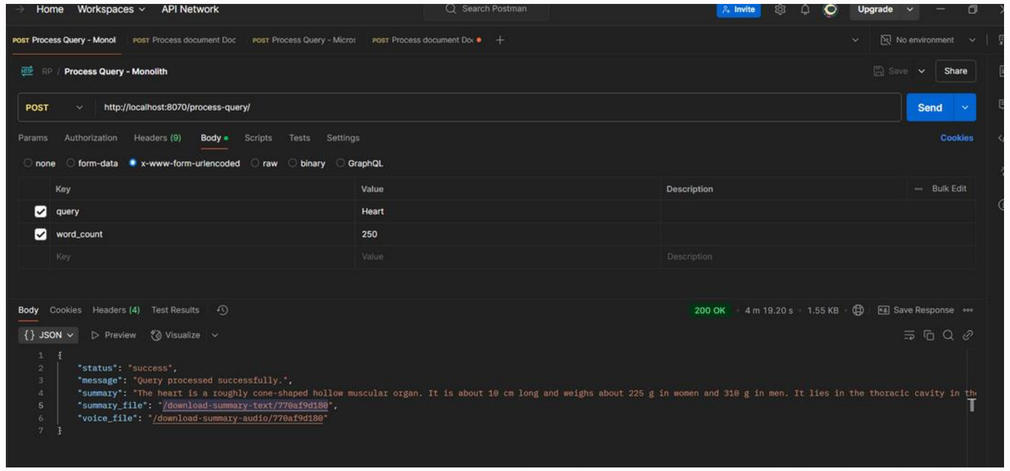
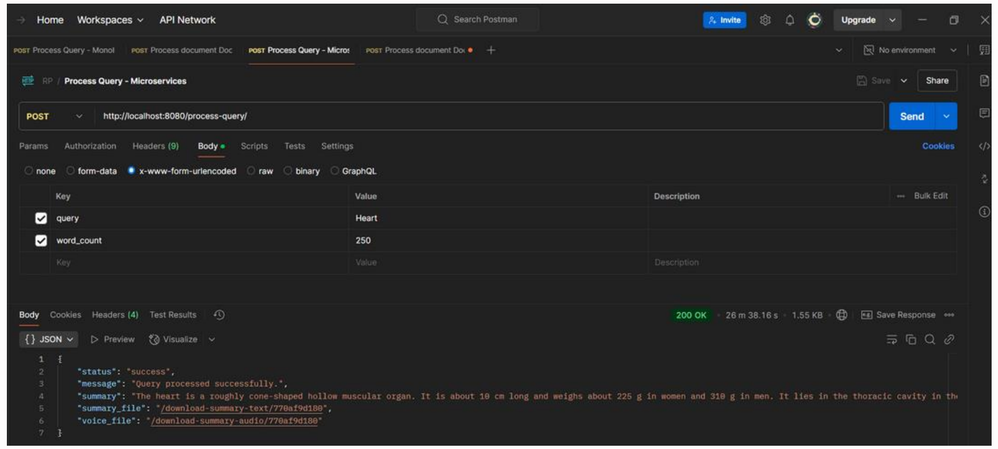
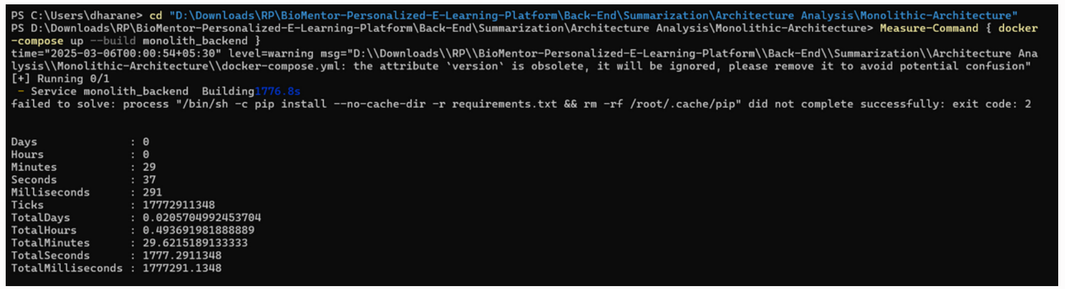
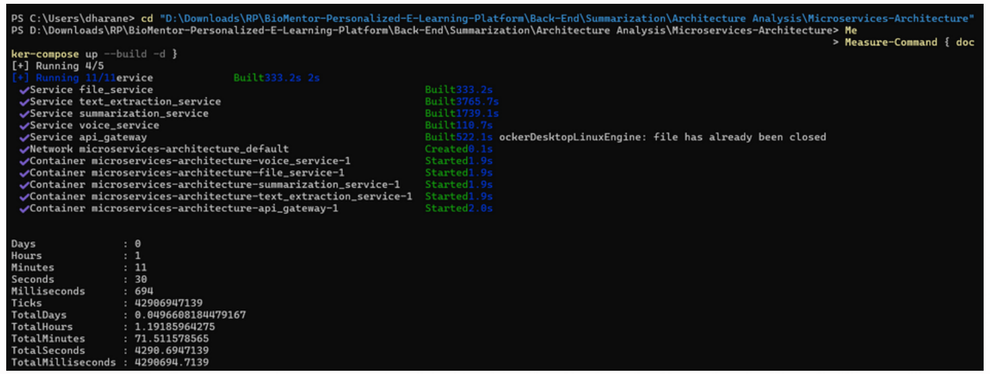
Key Functionalities

- File Processing – Extracts text and tables from .docx, .pptx, .pdf, and .txt using python-docx, python-pptx, PyMuPDF, Tabula, pdf2image, and pytesseract.
- Content Retrieval (RAG Model) – Embeds & retrieves relevant content via Sentence Transformers and FAISS.
- Inappropriate Content Detection –Uses better-profanity for explicit words, TextBlob for sentiment analysis, and wordnet for valid word verification..
- Text Summarization – Uses Flan-T5 with autocorrect & language_tool_python for clean, structured summaries.
- Text-to-Speech – Converts summaries into .mp3 using gTTS.
- Basic Multilingual Support – Translates notes into Tamil and Sinhala using DeepTranslator (GoogleTranslator).
- File Management – Serves summaries & structured notes via FastAPI endpoints.

Architecture Analysis



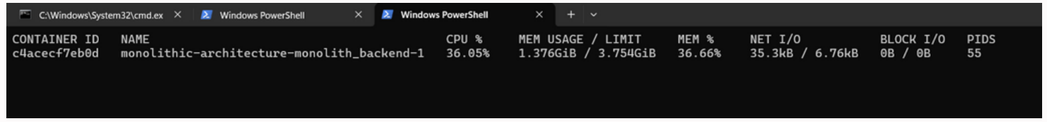


Metrics	Monolithic Architecture	Microservices Architecture
Response Time	85% faster, no API overhead (4.5 min) 	Slower, inter-service delays.(26.5 min) 
Deployment Speed	47% faster (37.8 min) 	Slower (71.5 min) 

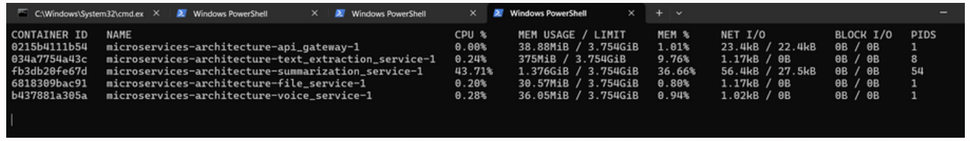

```
PS C:\Users\dharane> cd "D:\Downloads\RP\BioMentor-Personalized-E-Learning-Platform\Back-End\Summarization\Architecture Analysis\Monolithic-Architecture"
PS D:\Downloads\RP\BioMentor-Personalized-E-Learning-Platform\Back-End\Summarization\Architecture Analysis\Monolithic-Architecture> Measure-Command { docker
-compose up --build monolith_backend }
time="2025-03-06T00:00:54+05:30" level=warning msg="D:\\Downloads\\RP\\BioMentor-Personalized-E-Learning-Platform\\Back-End\\Summarization\\Architecture Ana
lysis\\Monolithic-Architecture\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 0/1
 - Service monolith_backend  Building1776.8s
failed to solve: process "/bin/sh -c pip install --no-cache-dir -r requirements.txt && rm -rf /root/.cache/pip" did not complete successfully: exit code: 2


Days              : 0
Hours             : 0
Minutes           : 29
Seconds           : 37
Milliseconds      : 291
Ticks             : 17772911348
TotalDays         : 0.0205704992453704
TotalHours        : 0.493691981888889
TotalMinutes      : 29.6215189133333
TotalSeconds      : 1777.2911348
TotalMilliseconds : 1777291.1348
```

```
PS C:\Users\dharane> cd "D:\Downloads\RP\BioMentor-Personalized-E-Learning-Platform\Back-End\Summarization\Architecture Analysis\Microservices-Architecture"
PS D:\Downloads\RP\BioMentor-Personalized-E-Learning-Platform\Back-End\Summarization\Architecture Analysis\Microservices-Architecture> Me
                                                                                                                                      > Measure-Command { doc
ker-compose up --build -d }
[+] Running 4/5
[+] Running 11/11ervice                 Built333.2s 2s
 ✔Service file_service                                      Built333.2s
 ✔Service text_extraction_service                           Built3765.7s
 ✔Service summarization_service                             Built1739.1s
 ✔Service voice_service                                     Built110.7s
 ✔Service api_gateway                                       Built522.1s ockerDesktopLinuxEngine: file has already been closed
 ✔Network microservices-architecture_default                Created0.1s
 ✔Container microservices-architecture-voice_service-1      Started1.9s
 ✔Container microservices-architecture-file_service-1       Started1.9s
 ✔Container microservices-architecture-summarization_service-1  Started1.9s
 ✔Container microservices-architecture-text_extraction_service-1  Started1.9s
 ✔Container microservices-architecture-api_gateway-1        Started2.0s


Days              : 0
Hours             : 1
Minutes           : 11
Seconds           : 30
Milliseconds      : 694
Ticks             : 42906947139
TotalDays         : 0.0496608184479167
TotalHours        : 1.191859642275
TotalMinutes      : 71.511578565
TotalSeconds      : 4290.6947139
TotalMilliseconds : 4290694.7139
```

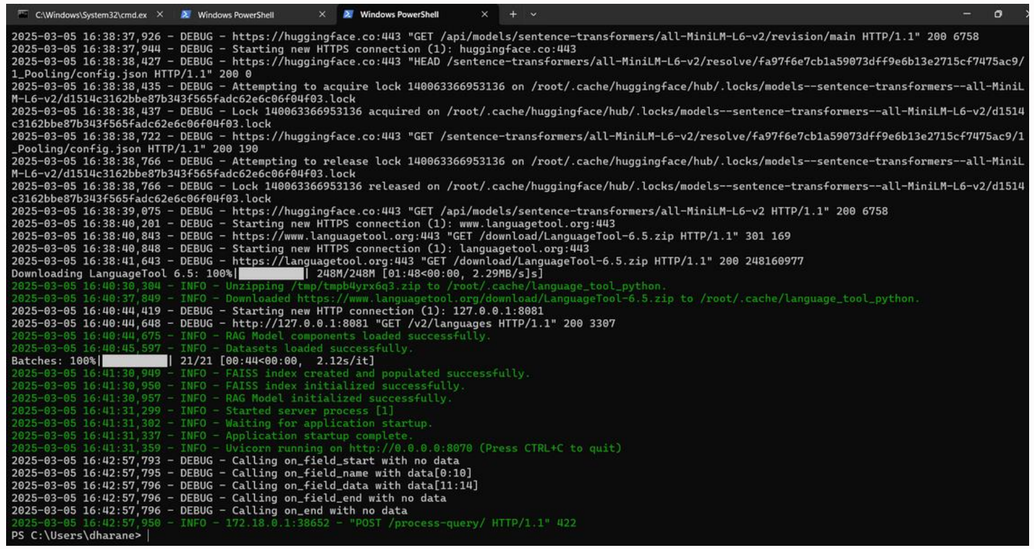
Metrics	Monolithic Architecture	Microservices Architecture
CPU and Memory Usage	Lower (~34% CPU, 28-36% RAM)	Higher (`43-62% CPU, 37-40% RAM)
Debugging	Easier, centralized logs	Harder, distributed logs



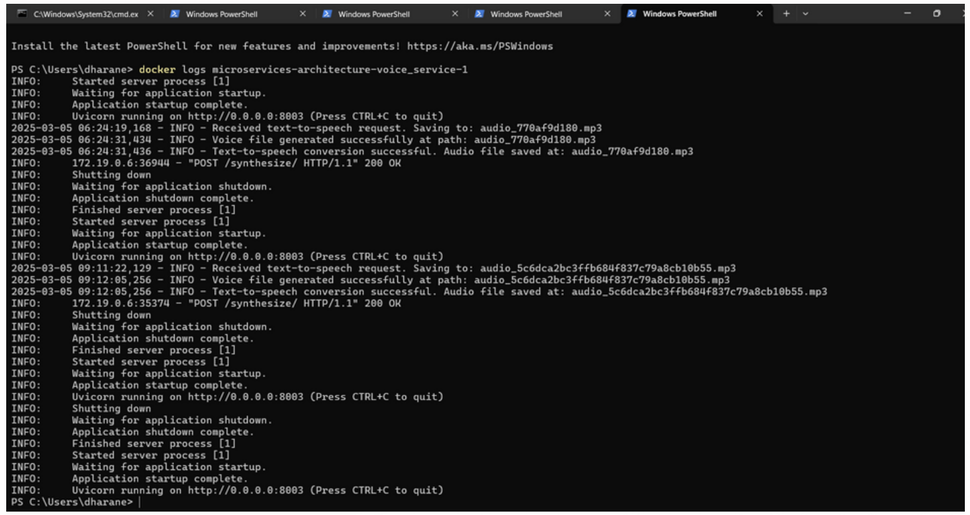
```
CONTAINER ID   NAME                                          CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O   PIDS
c4acecf7eb0d   monolithic-architecture-monolith_backend-1   36.05%    1.376GiB / 3.754GiB   36.66%    35.3kB / 6.76kB   0B / 0B     55
```



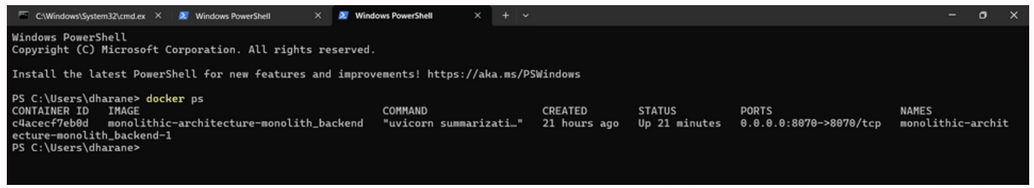
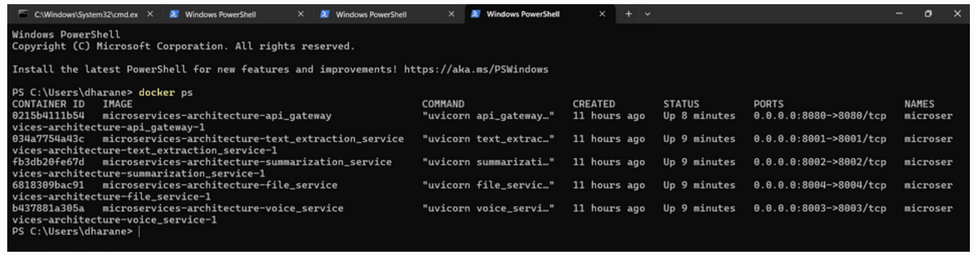
```
CONTAINER ID   NAME                                                CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O   PIDS
0215b4111b54   microservices-architecture-api_gateway-1             0.00%     38.88MiB / 3.754GiB   1.01%     23.4kB / 22.4kB   0B / 0B     1
034a7754a43c   microservices-architecture-text_extraction_service-1   0.24%     375MiB / 3.754GiB     9.76%     1.17kB / 0B       0B / 0B     8
fb3db20fe67d   microservices-architecture-summarization_service-1   43.71%    1.376GiB / 3.754GiB   36.66%    56.4kB / 27.5kB   0B / 0B     54
6818309bac91   microservices-architecture-file_service-1            0.20%     30.57MiB / 3.754GiB   0.80%     1.17kB / 0B       0B / 0B     1
b437881a305a   microservices-architecture-voice_service-1           0.28%     36.05MiB / 3.754GiB   0.94%     1.02kB / 0B       0B / 0B     1
```



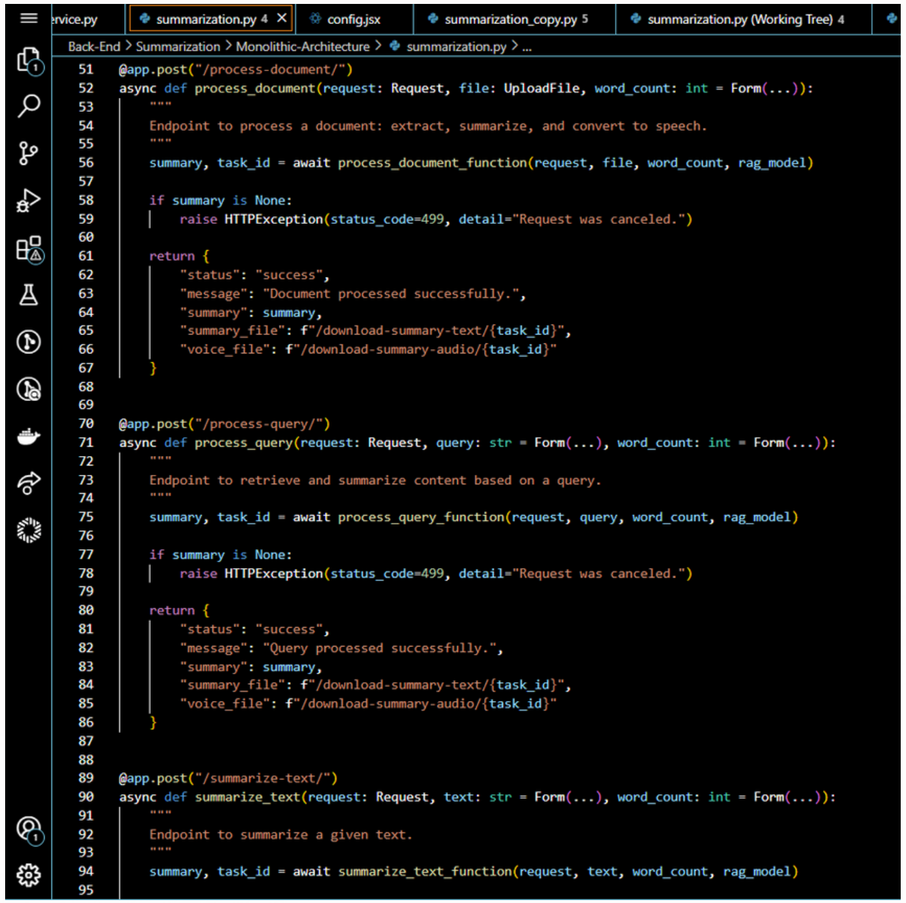
```
2025-03-05 16:38:37,926 - DEBUG - https://huggingface.co:443 "GET /api/models/sentence-transformers/all-MiniLM-L6-v2/revision/main HTTP/1.1" 200 6758
2025-03-05 16:38:37,944 - DEBUG - Starting new HTTPS connection (1): huggingface.co:443
2025-03-05 16:38:38,427 - DEBUG - https://huggingface.co:443 "HEAD /sentence-transformers/all-MiniLM-L6-v2/resolve/fa97f6e7cb1a59073dff9e6b13e2715cf7475ac9/1_Pooling/config.json HTTP/1.1" 200 0
2025-03-05 16:38:38,435 - DEBUG - Attempting to acquire lock 140063366953136 on /root/.cache/huggingface/hub/.locks/models--sentence-transformers--all-MiniLM-L6-v2/d1514c3162bbe87b343f565fadc62e6c06f04f03.lock
2025-03-05 16:38:38,437 - DEBUG - Lock 140063366953136 acquired on /root/.cache/huggingface/hub/.locks/models--sentence-transformers--all-MiniLM-L6-v2/d1514c3162bbe87b343f565fadc62e6c06f04f03.lock
2025-03-05 16:38:38,722 - DEBUG - https://huggingface.co:443 "GET /sentence-transformers/all-MiniLM-L6-v2/resolve/fa97f6e7cb1a59073dff9e6b13e2715cf7475ac9/1_Pooling/config.json HTTP/1.1" 200 190
2025-03-05 16:38:38,766 - DEBUG - Attempting to release lock 140063366953136 on /root/.cache/huggingface/hub/.locks/models--sentence-transformers--all-MiniLM-L6-v2/d1514c3162bbe87b343f565fadc62e6c06f04f03.lock
2025-03-05 16:38:38,766 - DEBUG - Lock 140063366953136 released on /root/.cache/huggingface/hub/.locks/models--sentence-transformers--all-MiniLM-L6-v2/d1514c3162bbe87b343f565fadc62e6c06f04f03.lock
2025-03-05 16:38:39,075 - DEBUG - https://huggingface.co:443 "GET /api/models/sentence-transformers/all-MiniLM-L6-v2 HTTP/1.1" 200 6758
2025-03-05 16:38:40,201 - DEBUG - Starting new HTTPS connection (1): www.languagetool.org:443
2025-03-05 16:38:40,843 - DEBUG - https://www.languagetool.org:443 "GET /download/LanguageTool-6.5.zip HTTP/1.1" 301 169
2025-03-05 16:38:40,848 - DEBUG - Starting new HTTPS connection (1): languagetool.org:443
2025-03-05 16:38:41,643 - DEBUG - https://languagetool.org:443 "GET /download/LanguageTool-6.5.zip HTTP/1.1" 200 248160977
Downloading LanguageTool 6.5: 100%|██████████| 248M/248M [01:48<00:00, 2.29MB/s]s]
2025-03-05 16:40:30,304 - INFO - Unzipping /tmp/tmpb4yrx6q3.zip to /root/.cache/language_tool_python.
2025-03-05 16:40:37,849 - INFO - Downloaded https://www.languagetool.org/download/LanguageTool-6.5.zip to /root/.cache/language_tool_python.
2025-03-05 16:40:44,419 - DEBUG - Starting new HTTP connection (1): 127.0.0.1:8081
2025-03-05 16:40:44,648 - DEBUG - http://127.0.0.1:8081 "GET /v2/languages HTTP/1.1" 200 3307
2025-03-05 16:40:44,675 - INFO - RAG Model components loaded successfully.
2025-03-05 16:40:45,597 - INFO - Datasets loaded successfully.
Batches: 100%|██████████| 21/21 [00:44<00:00,  2.12s/it]
2025-03-05 16:41:30,949 - INFO - FAISS index created and populated successfully.
2025-03-05 16:41:30,950 - INFO - FAISS index initialized successfully.
2025-03-05 16:41:30,957 - INFO - RAG Model initialized successfully.
2025-03-05 16:41:31,299 - INFO - Started server process [1]
2025-03-05 16:41:31,302 - INFO - Waiting for application startup.
2025-03-05 16:41:31,337 - INFO - Application startup complete.
2025-03-05 16:41:31,359 - INFO - Uvicorn running on http://0.0.0.0:8070 (Press CTRL+C to quit)
2025-03-05 16:42:57,793 - DEBUG - Calling on_field_start with no data
2025-03-05 16:42:57,795 - DEBUG - Calling on_field_name with data[0:10]
2025-03-05 16:42:57,796 - DEBUG - Calling on_field_data with data[11:14]
2025-03-05 16:42:57,796 - DEBUG - Calling on_field_end with no data
2025-03-05 16:42:57,796 - DEBUG - Calling on_end with no data
2025-03-05 16:42:57,950 - INFO - 172.18.0.1:38652 - "POST /process-query/ HTTP/1.1" 422
PS C:\Users\dharane>
```



```
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\dharane> docker logs microservices-architecture-voice_service-1
INFO:     Started server process [1]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8003 (Press CTRL+C to quit)
2025-03-05 06:24:19,168 - INFO - Received text-to-speech request. Saving to: audio_770af9d180.mp3
2025-03-05 06:24:31,434 - INFO - Voice file generated successfully at path: audio_770af9d180.mp3
2025-03-05 06:24:31,436 - INFO - Text-to-speech conversion successful. Audio file saved at: audio_770af9d180.mp3
INFO:     172.19.0.6:36944 - "POST /synthesize/ HTTP/1.1" 200 OK
INFO:     Shutting down
INFO:     Waiting for application shutdown.
INFO:     Application shutdown complete.
INFO:     Finished server process [1]
INFO:     Started server process [1]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8003 (Press CTRL+C to quit)
2025-03-05 09:11:22,129 - INFO - Received text-to-speech request. Saving to: audio_5c6dca2bc3ffb684f837c79a8cb10b55.mp3
2025-03-05 09:12:05,256 - INFO - Voice file generated successfully at path: audio_5c6dca2bc3ffb684f837c79a8cb10b55.mp3
2025-03-05 09:12:05,256 - INFO - Text-to-speech conversion successful. Audio file saved at: audio_5c6dca2bc3ffb684f837c79a8cb10b55.mp3
INFO:     172.19.0.6:35374 - "POST /synthesize/ HTTP/1.1" 200 OK
INFO:     Shutting down
INFO:     Waiting for application shutdown.
INFO:     Application shutdown complete.
INFO:     Finished server process [1]
INFO:     Started server process [1]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8003 (Press CTRL+C to quit)
INFO:     Shutting down
INFO:     Waiting for application shutdown.
INFO:     Application shutdown complete.
INFO:     Finished server process [1]
INFO:     Started server process [1]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8003 (Press CTRL+C to quit)
PS C:\Users\dharane>
```

Metrics	Monolithic Architecture	Microservices Architecture
Infrastructure	Simple, single container 	Complex, multiple containers 
Fault Tolerance	Lower, single failure risk	Higher, independent services

summarization.py

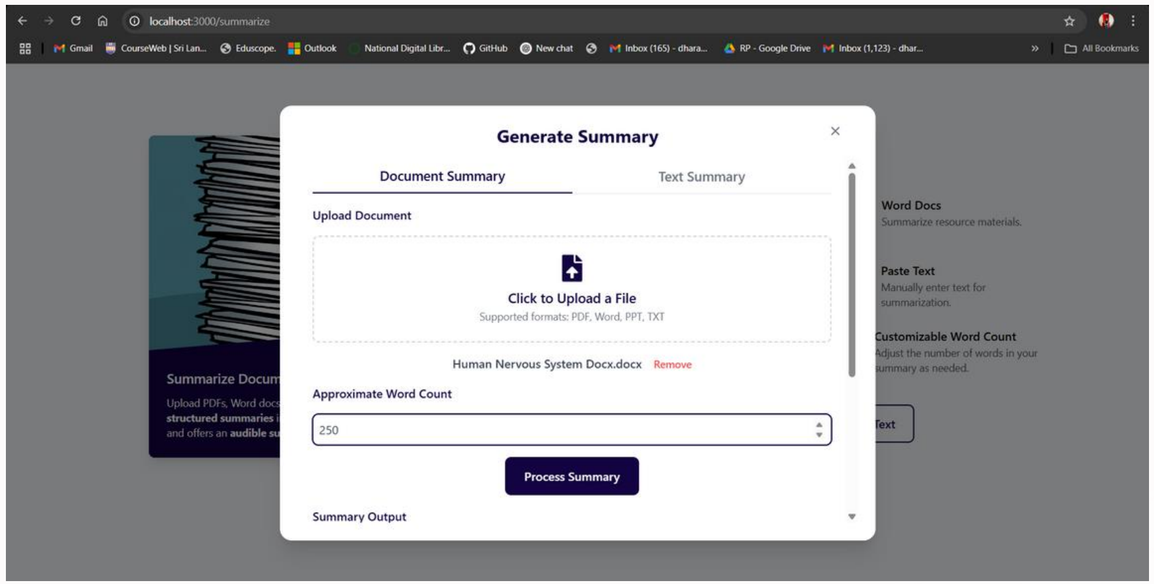


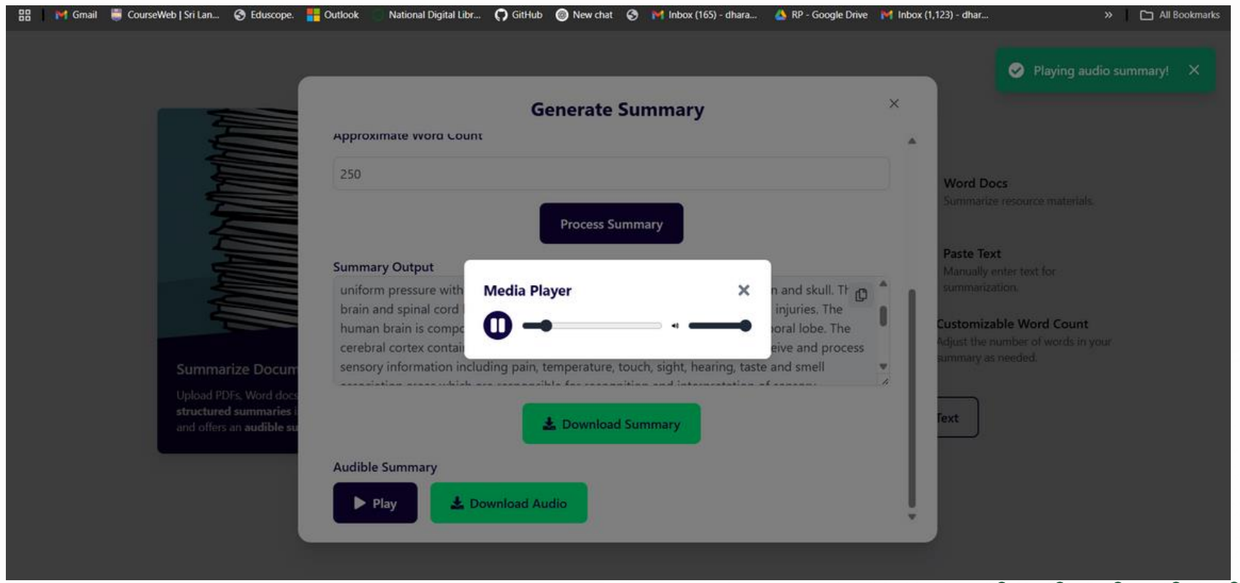
Summarization.jsx

SLIIT FACULTY OF COMPUTING

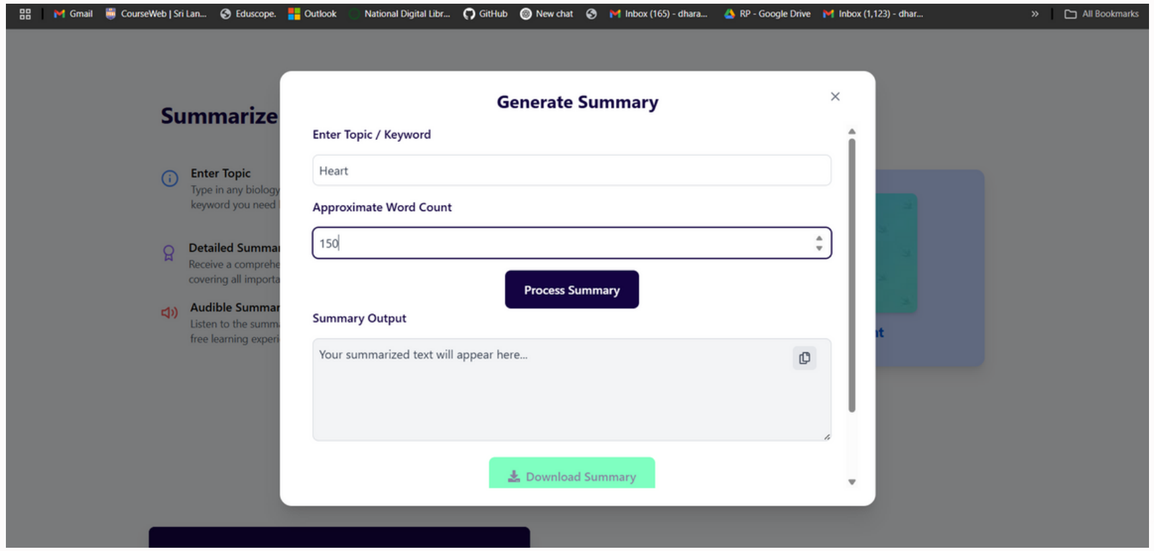
IT21068478 |Dharane. S | 24-25J-257

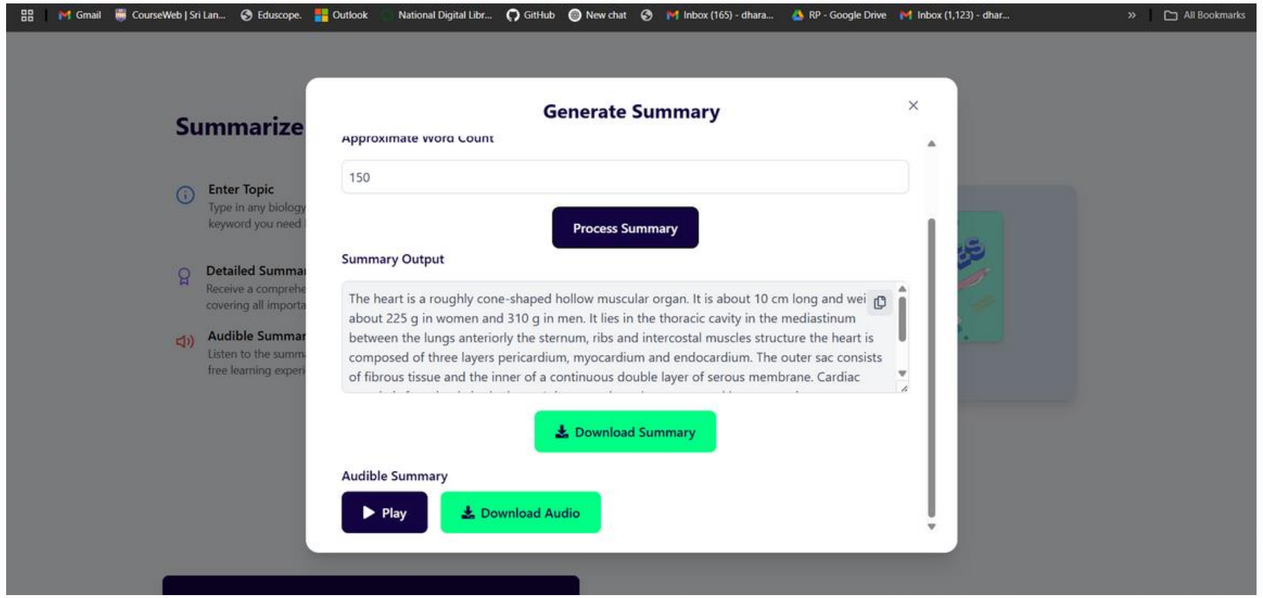
Working Product



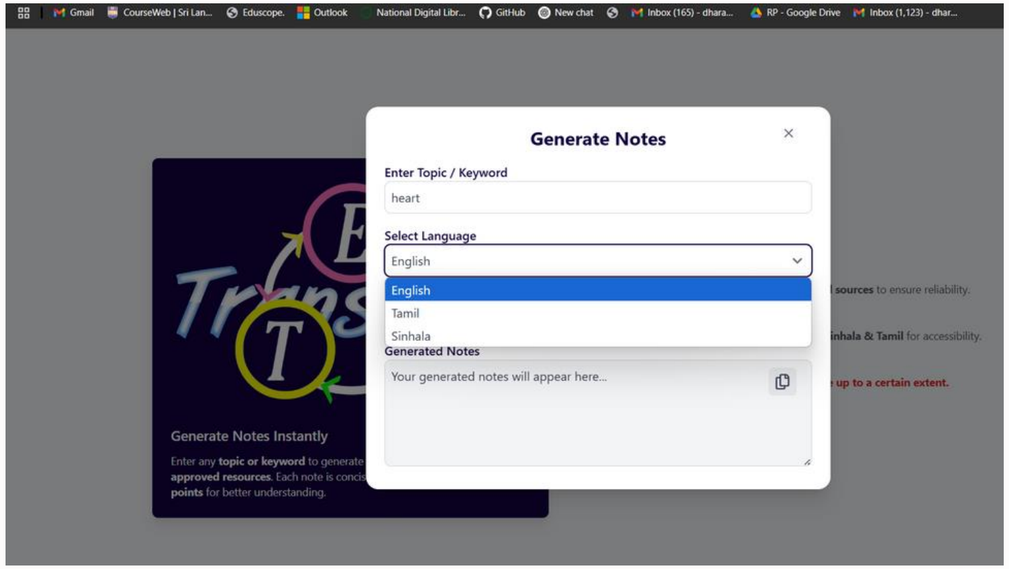


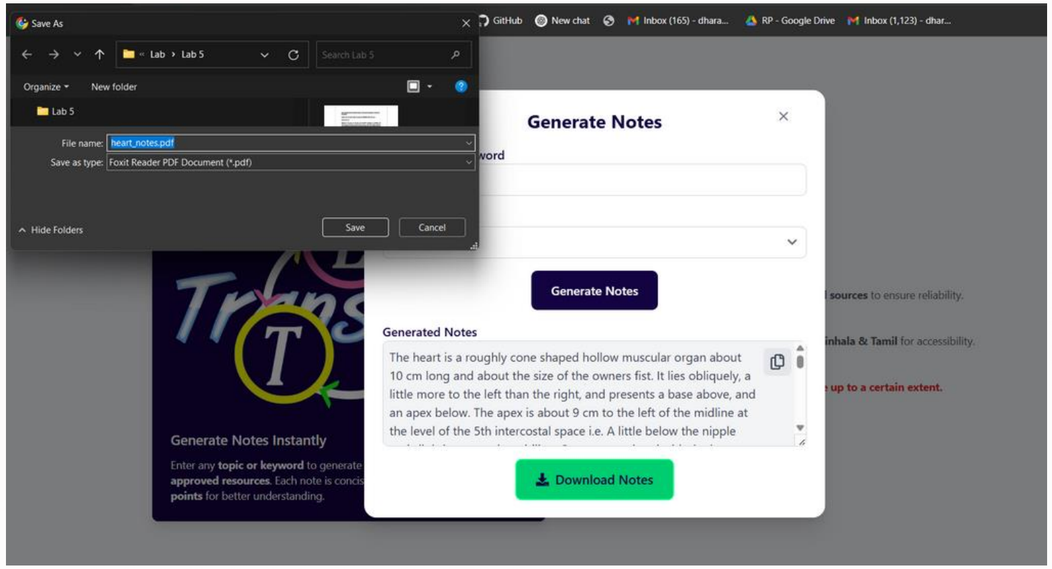
Working Product

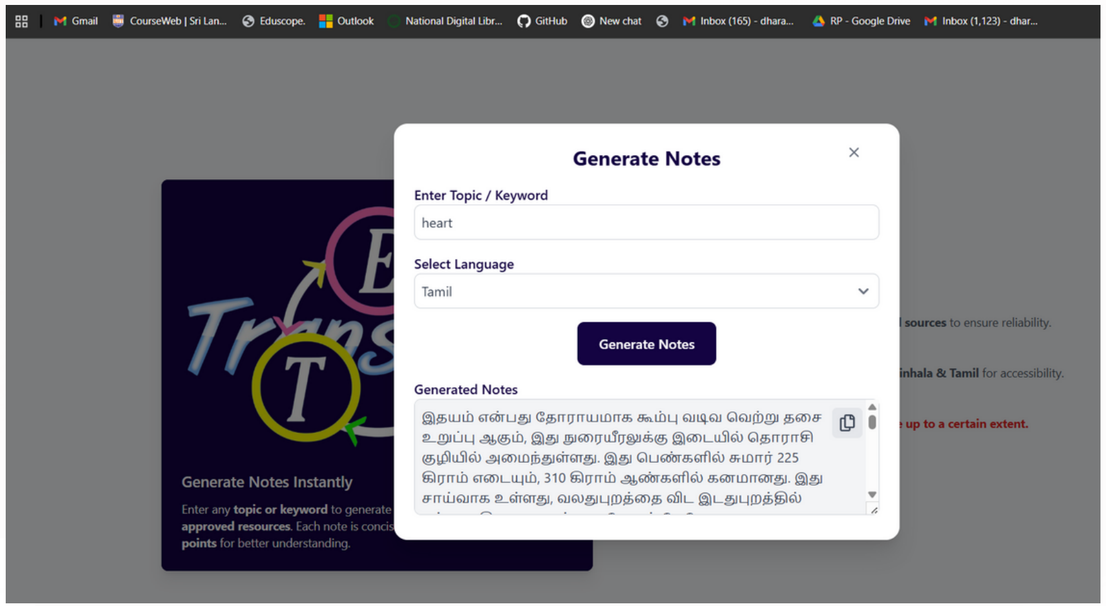




Working Product



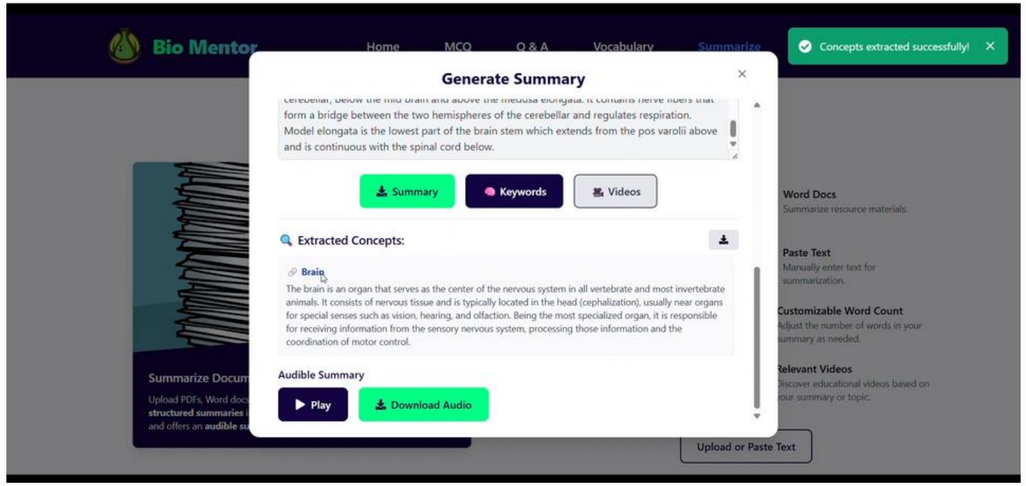


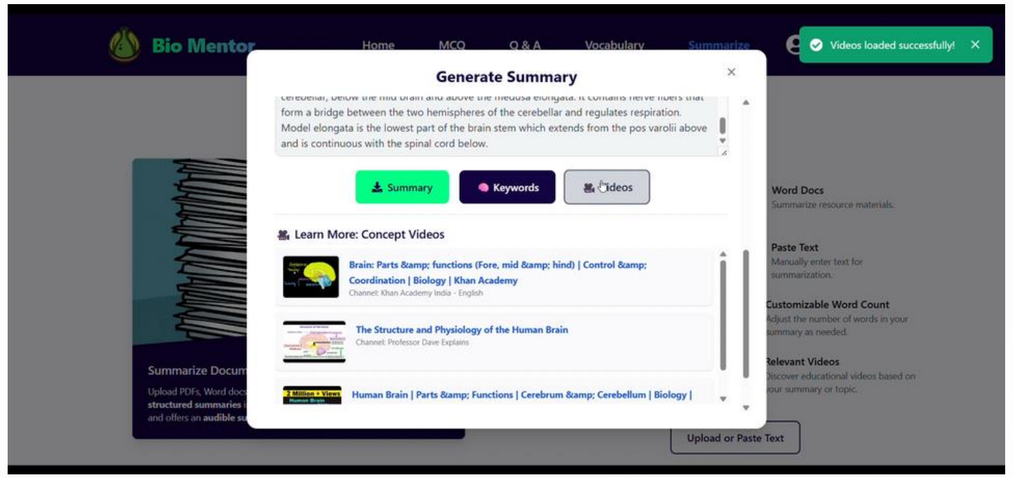


Completion of the project

Project Enhancements

- **Concept Extraction** – Identifies scientific keywords using nltk + Wikipedia API and returns their definitions.
- **YouTube Video Suggestion** – Retrieves educational biology videos by extracting core topics from summaries or queries using TF-IDF + YouTube Data API.
- **Inappropriate Content Detection** – Incorporated zero-shot classification model





Testing - Unit Testing

- **Framework:** Pytest

- **Modules Tested:**
  - test_file_handler.py: File saving, PDF generation
  - test_text_extraction.py: Text cleaning & grammar correction
  - test_rag.py: Summarization trimming & formatting
  - test_voice_service.py: TTS output in memory

Back-End > Summarization > Monolithic-Architecture > tests > test_rag.py > test_truncate_to_word_count

```python
import pytest
import sys
import os
sys.path.append(os.path.abspath(os.path.join(os.path.dirname(__file__), '..')))
from rag import RAGModel

class DummyModel:
    def generate_summary_for_long_text(self, text, max_words=100):
        return "summary"

def test_truncate_to_word_count():
    text = "This is a test sentence. " * 50
    model = DummyModel()
    result = RAGModel.truncate_to_word_count(text, max_words=20)
    assert len(result.split()) <= 20

def test_postprocess_summary():
    summary = "this is a test. this is another sentence."
    model = DummyModel()
    result = RAGModel.postprocess_summary(model, summary)
    assert result[0].isupper()
    assert result.endswith(".")

```

test_file_handler.py 2, U

Back-End > Summarization > Monolithic-Architecture > tests > test_file_handler.py > ...

```python
import pytest
from fastapi import UploadFile
from io import BytesIO
import sys
import os
sys.path.append(os.path.abspath(os.path.join(os.path.dirname(__file__), '..')))
from file_handler import save_uploaded_file, generate_pdf

def test_save_uploaded_file(tmp_path):
    dummy_content = b"Hello, this is test content."
    file = UploadFile(filename="test.txt", file=BytesIO(dummy_content))
    path = save_uploaded_file(file)
    assert os.path.exists(path)
    os.remove(path)  # Clean up

def test_generate_pdf():
    content = "Line 1\nLine 2"
    title = "Test PDF"
    result = generate_pdf(content, title)
    assert isinstance(result, bytes)
    assert result.startswith(b"%PDF")
```

Testing - Integration Testing

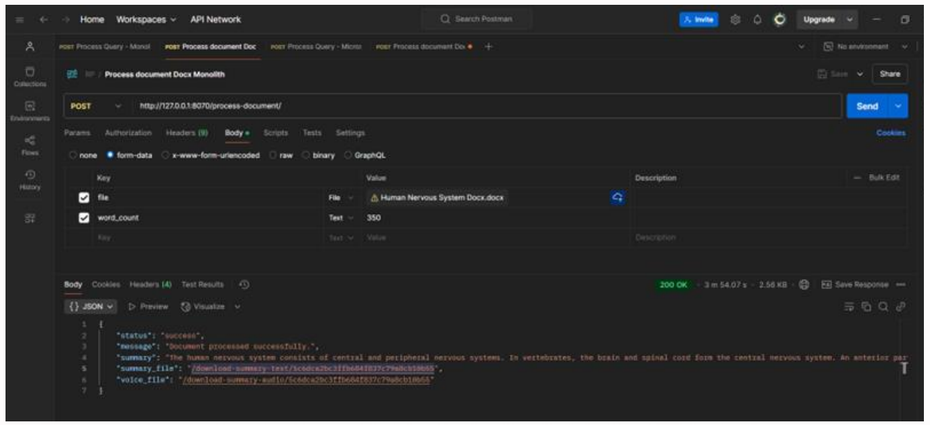
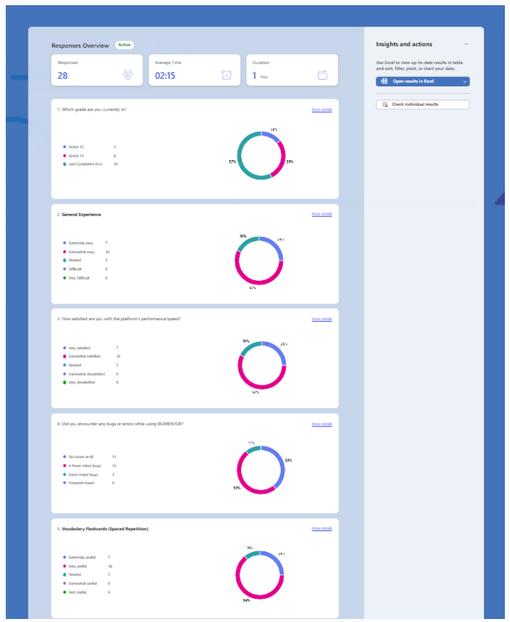
- **Framework:** Pytest with FastAPI TestClient

- **Workflows Tested:**
  - /process-document/
  - /summarize-text/
  - /process-query/
  - /generate-notes/

```python
import os
import io
import pytest
from fastapi.testclient import TestClient
from summarization import app
from summarization_functions import file_store
from reportlab.pdfgen import canvas

client = TestClient(app)

@pytest.fixture
def dummy_pdf():
    buffer = io.BytesIO()
    c = canvas.Canvas(buffer)
    c.drawString(100, 750, (
        "Photosynthesis is a process used by plants and other organisms to convert light energy into
chemical energy. "
        "This energy is stored in carbohydrate molecules, such as sugars, which are synthesized from
carbon dioxide and water."
    ))
    c.save()
    buffer.seek(0)
    return buffer

def test_summarize_text():
    response = client.post("/summarize-text/", data={
        "text": "Photosynthesis is the process by which plants convert sunlight into chemical energy.",
        "word_count": 30
    })
    assert response.status_code == 200
    data = response.json()
    assert "summary" in data
    assert isinstance(data["summary"], str)

def test_process_query():
    response = client.post("/process-query/", data={
        "query": "Heart",
        "word_count": 50
    })
    assert response.status_code == 200
    data = response.json()
    assert "summary" in data
    assert isinstance(data["summary"], str)

def test_generate_notes():
    response = client.post("/generate-notes/", data={
        "topic": "Heart",
        "lang": "en"
    })
    assert response.status_code == 200
    data = response.json()
    assert "structured_notes" in data
    # Ensure PDF file was registered
    if "download_link" in data:
        filename = data["download_link"].split("/")[-1]
        assert filename in file_store
        assert file_store[filename].startswith(b"%PDF")

def test_process_document_and_download(dummy_pdf):
    response = client.post(
        "/process-document/",
        files={"file": ("test.pdf", dummy_pdf, "application/pdf")},
        data={"word_count": 50}
    )
    assert response.status_code == 200
    data = response.json()
    assert "summary_file" in data
    assert "voice_file" in data

    task_id = data["summary_file"].split("/")[-1]

    # Fake file_store entries if backend failed to write files
    file_store.setdefault(f"summary_{task_id}.pdf", b"PDF test data")
    file_store.setdefault(f"summary_{task_id}.mp3", b"MP3 test data")

    summary_response = client.get(f"/download-summary-text/{task_id}")
    assert summary_response.status_code == 200
    assert summary_response.headers["content-type"] == "application/pdf"

    audio_response = client.get(f"/download-summary-audio/{task_id}")
    assert audio_response.status_code == 200
    assert audio_response.headers["content-type"] == "audio/mpeg"

def test_download_notes_direct():
    filename = "notes_Cell_Division_English.pdf"
    file_store[filename] = b"PDF notes content"
    response = client.get(f"/download-notes/{filename}")
    assert response.status_code == 200
    assert response.headers["content-type"] == "application/pdf"
```

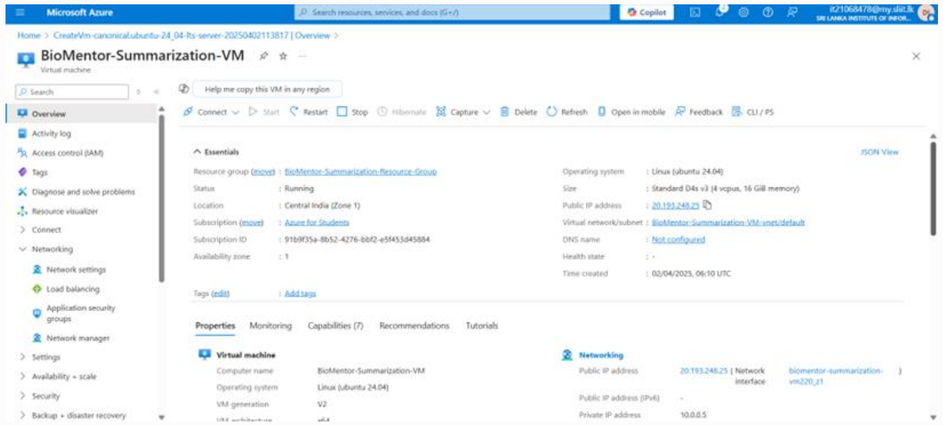
```
platform win32 -- Python 3.10.11, pytest-8.3.5, pluggy-1.5.0
rootdir: D:\Downloads\RP\Summarization\RP\Back-End\Summarization\Monolithic-Architecture\tests
configfile: pytest.ini
plugins: anyio-4.8.0
collected 13 items

tests\test_file_handler.py ..
tests\test_integration.py .....
tests\test_rag.py ..
tests\test_text_extraction.py ...
tests\test_voice_service.py .

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
============================================= 13 passed, 7 warnings in 308.41s (0:05:08) ====
sys:1: DeprecationWarning: builtin type swigvarlink has no __module__ attribute
(venv) PS D:\Downloads\RP\Summarization\RP\Back-End\Summarization\Monolithic-Architecture>
```

Testing - System Testing and UAT

System Testing	User Acceptance Testing
• **Framework:** Postman 	

Deployment

- Model hosted on Hugging Face Hub
- Backend deployed on Azure VM (Ubuntu, 4 vCPUs, 16 GB RAM),Region: Central India for low latency
- CI/CD is handled via GitHub Actions, which on every push to main, SSHs into the Azure VM, pulls the latest code, installs dependencies, and restarts the backend using systemctl.



References

[1] Abstractive Text Summarization Using BART. [ONLINE] https://ieeexplore.ieee.org/document/9972639 [Accessed 15 May 2024]

[2] NLP-Enhanced Long Document Summarization: A Comprehensive Approach for Information Condensation. [ONLINE] https://ieeexplore.ieee.org/document/10551101 [Accessed 20 May 2024]

[3] Topic level summary generation using BERTinduced Abstractive Summarization Model. [ONLINE] https://ieeexplore.ieee.org/document/9972639 [Accessed 01 June 2024]

[4] Speech-to-Text and Text-to-Speech Recognition using Deep Learning. [ONLINE] https://ieeexplore.ieee.org/document/9972639 [Accessed 05 June 2024]

[5] A Novel Text To Speech Conversion Using Hierarchical Neural Network. [ONLINE] https://ieeexplore.ieee.org/document/10533516 [Accessed 05 June 2024]

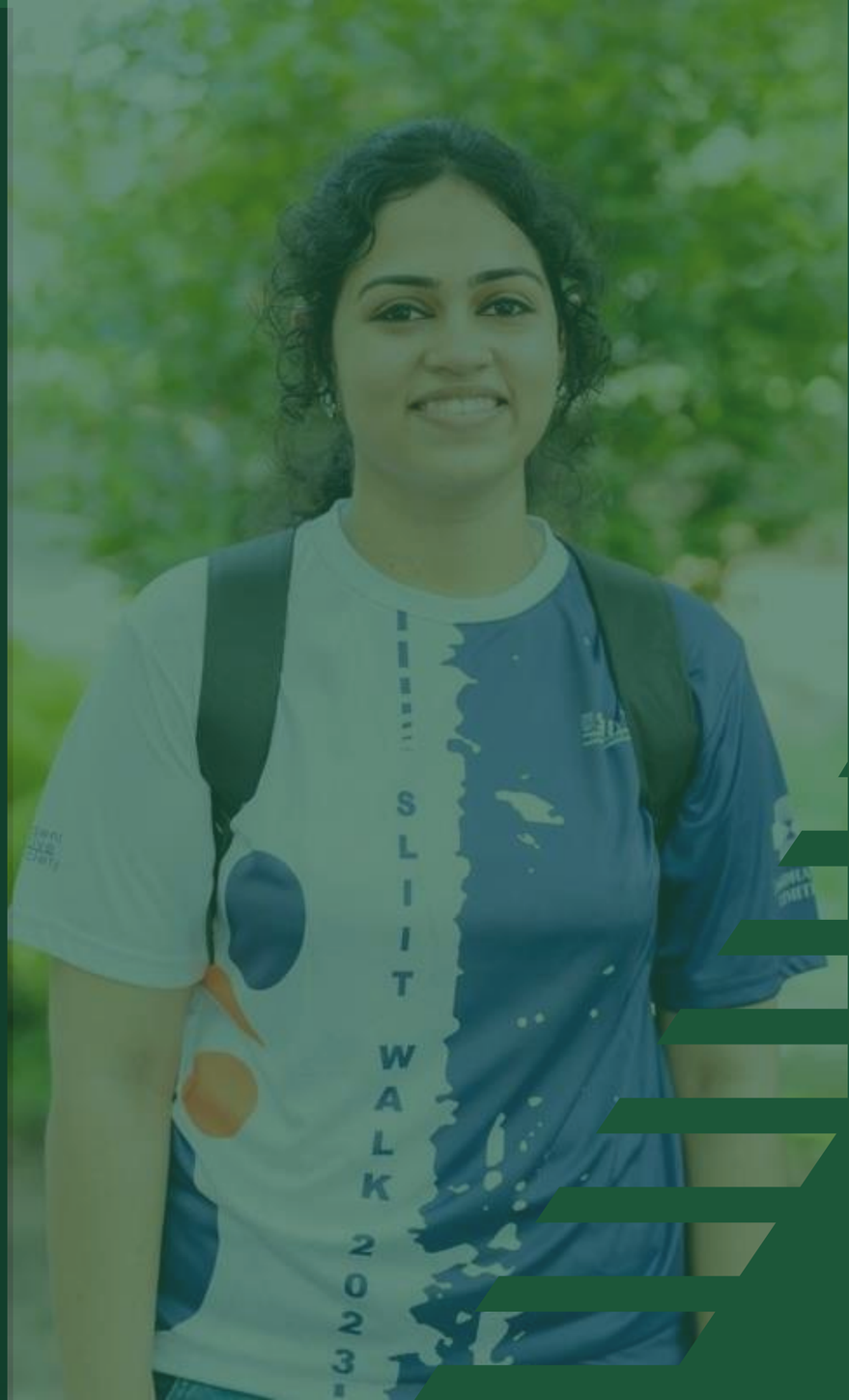
[6] An Abstractive Summarization and Conversation Bot using T5 and its Variants. [ONLINE]. https://ieeexplore.ieee.org/document/9972639 [Accessed 01 Aug 2024]

[7] Monolithic vs. Microservice Architecture: A Performance and Scalability Evaluation[ONLINE]. https://ieeexplore.ieee.org/document/9717259 [Accessed 01 Nov 2024]

IT21264634

Sujitha.S
Software Engineering

LLM BASED ADAPTIVE QUIZ PLATFORM TO IMPROVE MCQ ANSWERING SKILLS IN BIOLOGY FOR STUDENTS



Introduction

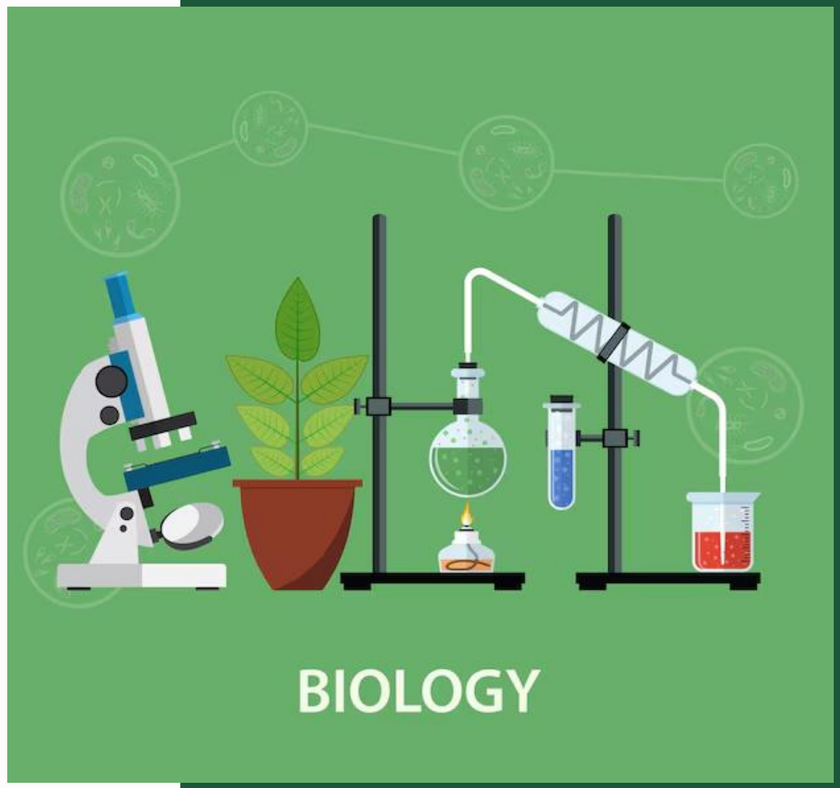
01 Background

02 Research Problem

03 Main and Sub Objectives

04 Methodology

05 Completion Of Project



BACKGROUND

Increasing demand for personalized learning experiences in education.

Need for adaptive assessment tools that cater to individual learning paces and capabilities.

RESEARCH PROBLEM

01

Traditional quiz platforms lack real-time adaptability, limiting their ability to adjust question difficulty based on a learner's performance trends.

02

Existing assessment systems do not effectively utilize performance-based difficulty adjustments, leading to less personalized and less efficient learning experiences.

OBJECTIVES

Objective 1

Develop a diverse set of MCQs categorized by difficulty levels for every user to engage different cognitive skills effectively.

Main Objective

Develop an intelligent adaptive quiz system that personalizes MCQs based on student performance, integrates government-approved educational resources, and dynamically adjusts difficulty to cater to varying proficiency levels.

Objective 2

Enhance the platform's ability to identify performance and provide targeted practice questions.

Objective 3

Implement a continuous performance tracking system that allows users to monitor their progress and improvements over time.

Objective 4

Ensure a user-friendly interface with accessible features to accommodate diverse learner needs.

IT21264634 |Sujitha. S | 24-25J-257

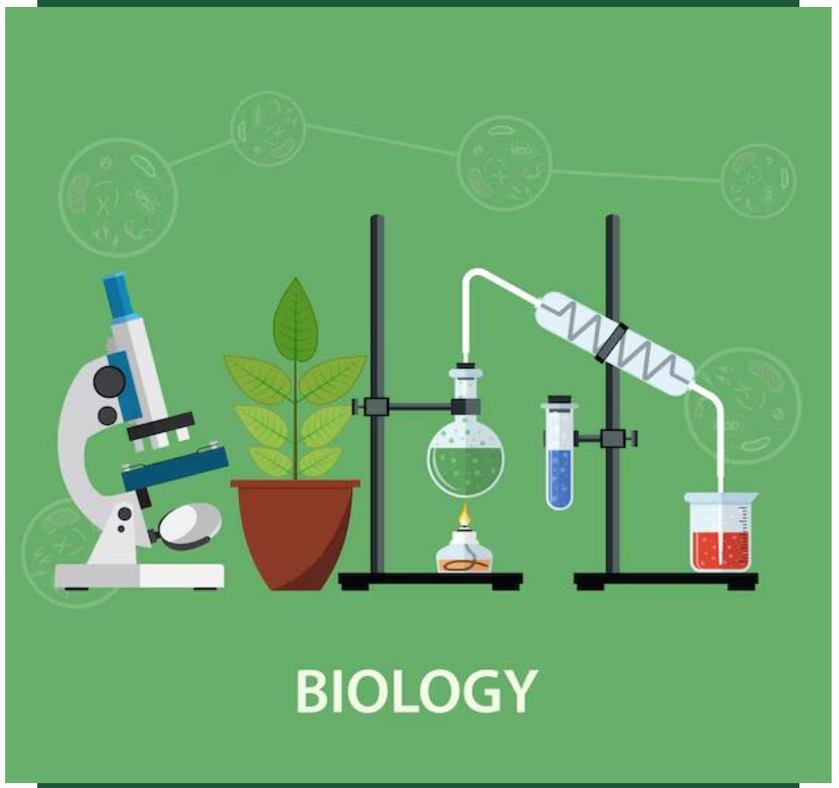
Methodology

01 System Diagram

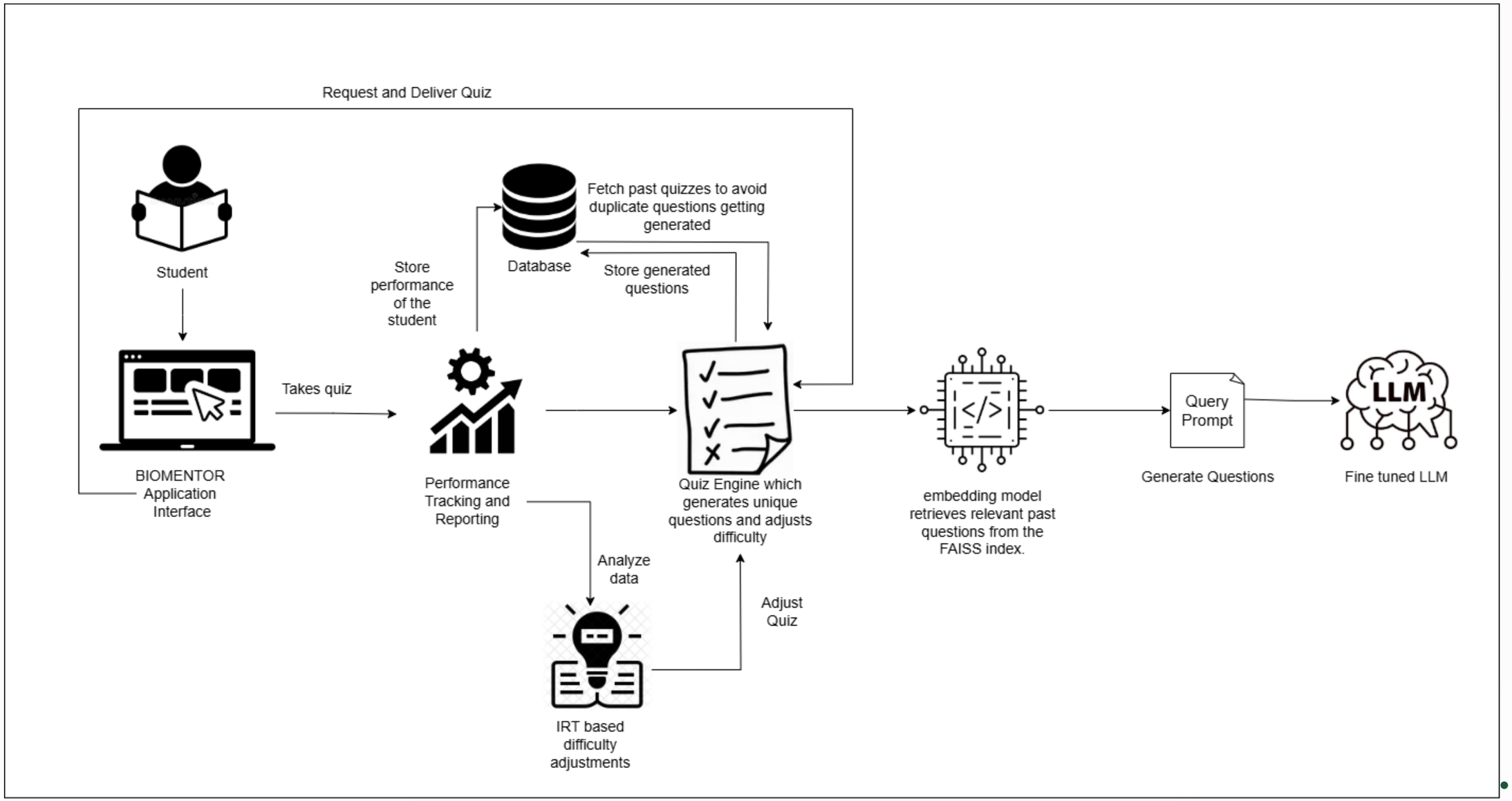
02 Tools and Technologies

03 Best Practices

04 GitHub Commits



System Diagram



Tools & Technologies

Programming Languages

Python
React Js

Database

Faiss
PMongoDB

Project Management

Jira

Other tools

Git
Draw.io
Postman
Figma

Frameworks

Transformers
Pandas
PyTorch
Flask
NumPy
Pandas
Matplotlib

Best Practices

Non-Functional Requirements

- Compatibility
- Accuracy
- Performance
- Usability
- Availability

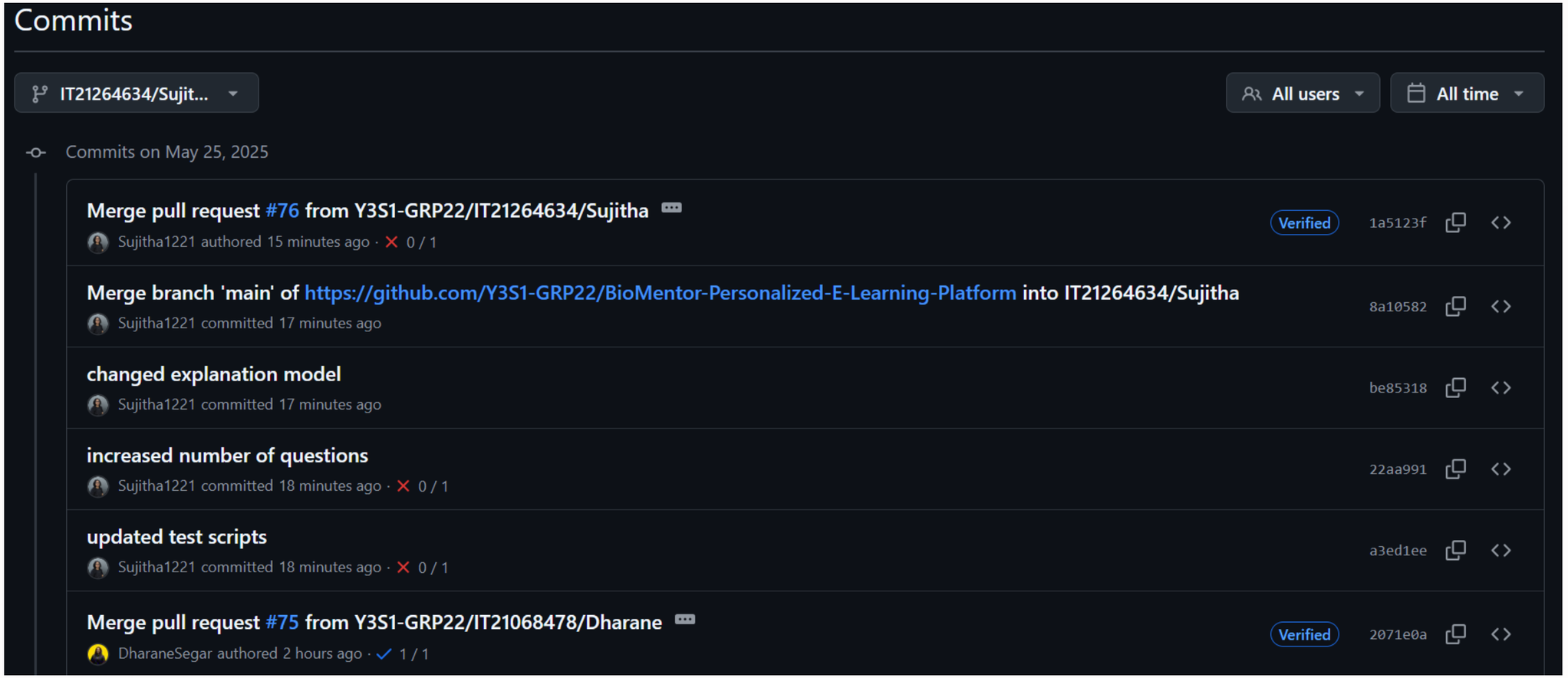
Design Excellence

- Real-Time Performance
- Accuracy and Reliability
- User Experience
- Integration

Standards & Best Practices

- Code Structure & Organization
- Web Security
- Model Quality
- RESTful API Design

GitHub Commits



Completion of the project

SLIIT
FACULTY OF COMPUTING

IT21264634 |Sujitha. S | 24-25J-257

Completion of the project

Data Collection

AutoSave Off merged_mcq_dataset Search SS

File Home Insert Page Layout Formulas Data Review View Automate Help Acrobat Comments Share

A12 Which of the following statements regarding vascular tissues of plants is correct?

	A	B	C	D	E	F	G	H
1	Question Text	Option 1	Option 2	Option 3	Option 4	Option 5	Correct Answer	Difficulty L
2	A feature common to lysosomes and peroxisomes is?	They are single membr	They transport resid	They contain oxidisir	They are important 1	They digest worn ou	They are single me	easy
3	Two characteristics that can be seen only in living organisms are?	adaptation and growth	movement and irrita	change with time an	metabolism and here	synthesis and deco	metabolism and he	easy
4	Which of the following statements is correct regarding transmission electro	Specimens are magnif	Less electrons may	Living specimens ca	Three-dimensional a	Specimens scatter	Living specimens c	hard
5	Which response which correctly indicates the event and phase in the eukar	DNA replication - G0 p	Synthesis of protein	Chromatin formatior	Production of cellula	Duplication of cent	Synthesis of protei	medium
6	In allosteric regulation of enzymes, which of the following statements is cor	regulatory molecules b	regulatory molecule	an activator molecul	inhibitory molecules	ATP functions as ar	regulatory molecul	medium
7	In ethyl alcohol fermentation, which of the following statements is correct?	one molecule of gluco	pyruvate is reduced	one molecule of CO2	final hydrogen accep	two molecules of A	two molecules of A	easy
8	Which of the following statements regarding glycosis of one molecule of glu	There is a net yield of f	Two hydrogen ions a	It partially depends c	Two NADH molecule	Part of glycolysis ta	Two NADH molecu	medium
9	Some events that took place during the evolution of organisms are as follow	A, B, C and D.	C, A, B and D.	C, B, A and D.	D, A, B and C.	D, B, A and C.	D, B, A and C.	hard
10	Which of the following pairs of organisms have the highest number of comm	Bat and crow.	Lizard and turtle.	Ichthyophis and Tae	Ulva and Pogonatum	Pinus and Cycas.	Lizard and turtle.	easy
11	Which of the following are unique characteristics of some phyla of the kingd	A and C only.	A and D only.	B and C only.	B and D only.	C and D only.	B and C only.	medium
12	Which of the following statements regarding vascular tissues of plants is co	Xylem tissues of ptero	Xylem vessel elemer	Tracheids provide su	Companion cells are	Pits are present be	Xylem tissues of pt	easy
13	Some structures of plants and their functions are shown below: A - Lenticels	A – P, B – R, C – Q.	A - R, B – P, C - P.	A - P, B - Q, C - R.	A - Q, B - P, C - P.	A – R, B-Q, C- R.		
14	Transport of water molecules due to physical adsorption by hydrophilic mat	imbibition.	osmosis.	facilitated diffusion.	bulk flow.	mass flow.		
15	Some steps in the process of opening and closing of stomata are given belo	A, B, C, D, E and F.	A, C, B, D, E and F.	A, C, D, B, E and F.	A, E, B, D, C and F.	A, E, C, D, B and F.		
16	A macronutrient and respectively a micronutrient which cause chlorosis in p	Mg and Mn.	Fe and Ni.	P and Mo.	N and S.	Cu and B.		
17	Two plant hormones that promote root formation are?	auxin and gibberellins.	cytokinins and absci	ethylene and auxin.	ethylene and gibbere	cytokinins and gibb		
18	Which of the following statements regarding epithelia is correct?	Stratified squamous e	Pseudostratified col	Simple columnar epi	Simple cuboidal epitl	Simple squamous e		
19	Which of these combinations of the three types of symbiosis seen among or	A only.	B only.	C only.	A and B only.	A and C only		
20	Which is the correct route of blood through the human heart from systemic	Left atrium, bicuspid v	Right atrium. tricusp	Left atrium. tricuspic	Left ventricle. bicusp	Right atrium. bicus		
21	Select the pair/pairs where an increase in (i) causes an increase in (ii. X - (i	X only.	Y only.	Z only.	X and Y only.	X and Z only.		
22	Which of the following indicates the forms that transport the lowest and high	Dissolved CO? - Carba	HCO?? - Carbamino	Carbaminohemoglol	HCO?? - Dissolved C	Dissolved CO? - HC		
23	If the tidal volume. residual volume. inspiratory reserve volume and expirato	1600 ml.	1700 ml.	3600 ml.	4700 ml.	5200 ml.		
24	Which of the following is part of the parasympathetic division of the autonon	inhibits saliva secretio	dilates the pupil of e	relaxes bronchi in lu	stimulates the releas	stimulates gall blac		
25	Which of the following statements regarding human vision is correct?	Changing refractory pc	Convergence occurs	Accommodation is ir	Photopsin in rods pro	Correct perception	Accommodation is	hard

merged_mcq_dataset

Ready Accessibility: Unavailable 100%

SLIIT FACULTY OF COMPUTING

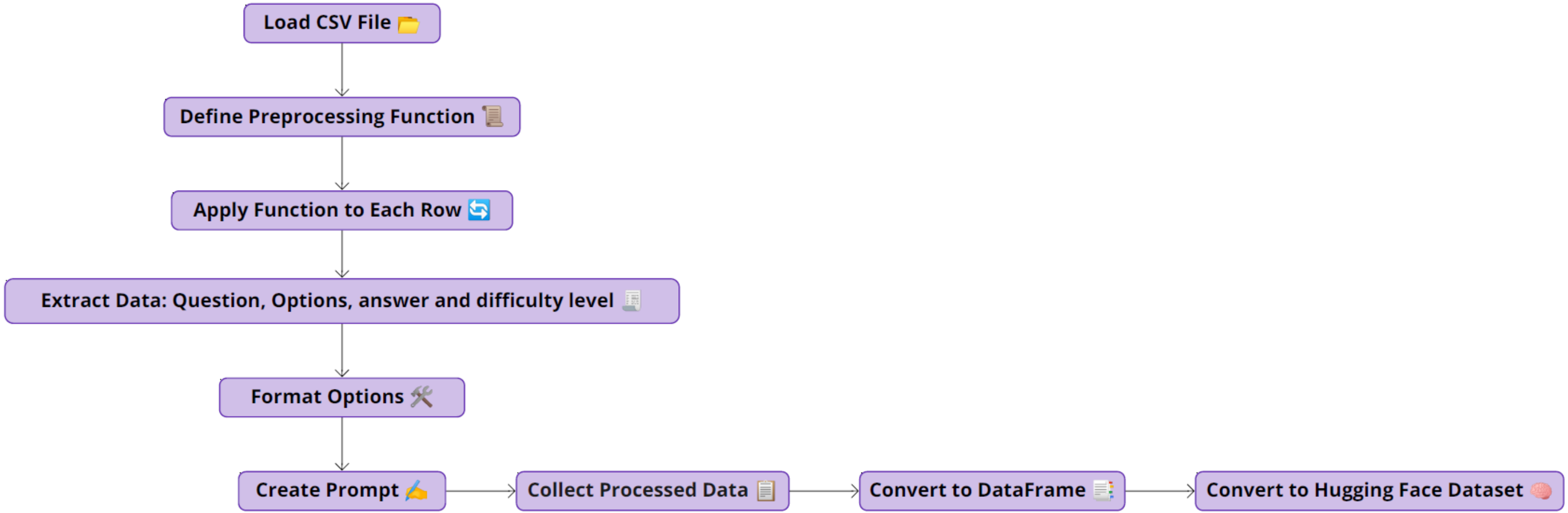
IT21264634 |Sujitha. S | 24-25J-257

Model Selection

Why llama-2-7b-chat-hf?

- Pretrained on Diverse Data
- Fine-Tuning Capability
- Natural Language Understanding
- Customization for Domain-Specific Terminology
- Cost-Effective and Accessible
- Scalability and Efficiency
- Contextual Responses

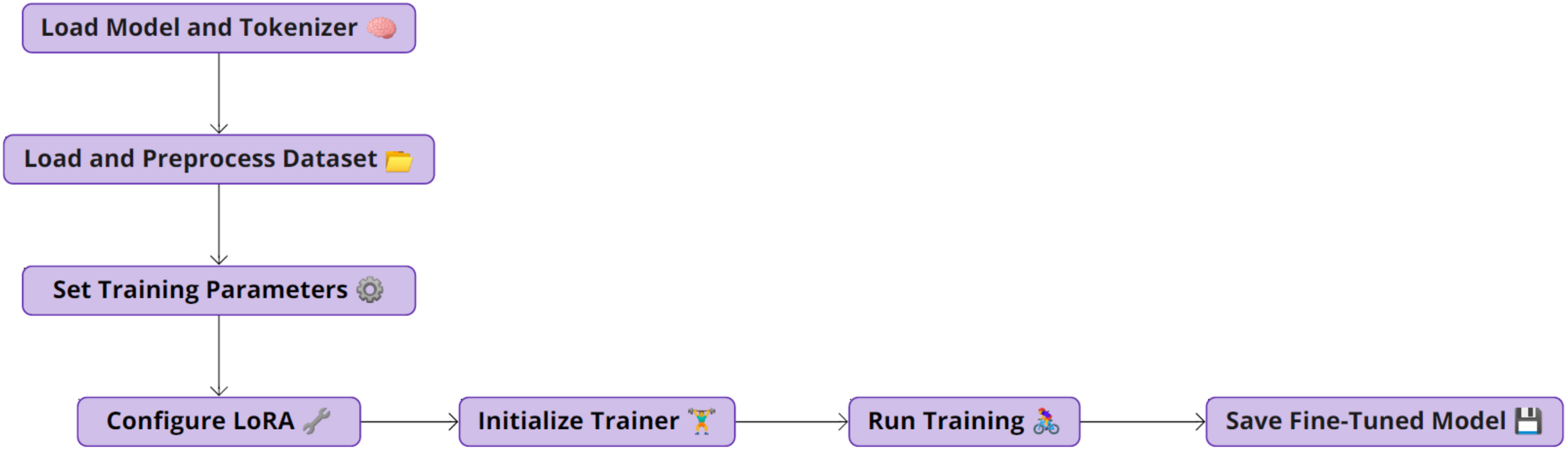
Data pre-processing



Data pre-processing

```python
# Load and preprocess your CSV dataset
df = pd.read_csv(file_path, encoding='ISO-8859-1')

# Define a preprocessing function to format the questions, options, and difficulty
def preprocess_data(row):
    question = row["Question Text"]
    options = [row[f"Option {i}"] for i in range(1, 6) if pd.notna(row[f"Option {i}"])]
    correct_answer = row["Correct Answer"]
    difficulty = row["Difficulty Level"]

    # Format question prompt for training
    options_text = " ".join([f"{chr(65+i)}. {opt.strip()}" for i, opt in enumerate(options)])
    prompt = f"Question ({difficulty}): {question}\nOptions: {options_text}\nChoose the correct answer:"

    return {
        "text": prompt,
        "label": correct_answer
    }

# Apply preprocessing to the dataset and convert to DataFrame
processed_data = df.apply(preprocess_data, axis=1).tolist()
processed_df = pd.DataFrame(processed_data)

# Convert the preprocessed DataFrame into a Hugging Face Dataset format
dataset = Dataset.from_pandas(processed_df)
```

Fine-tuning the LLM

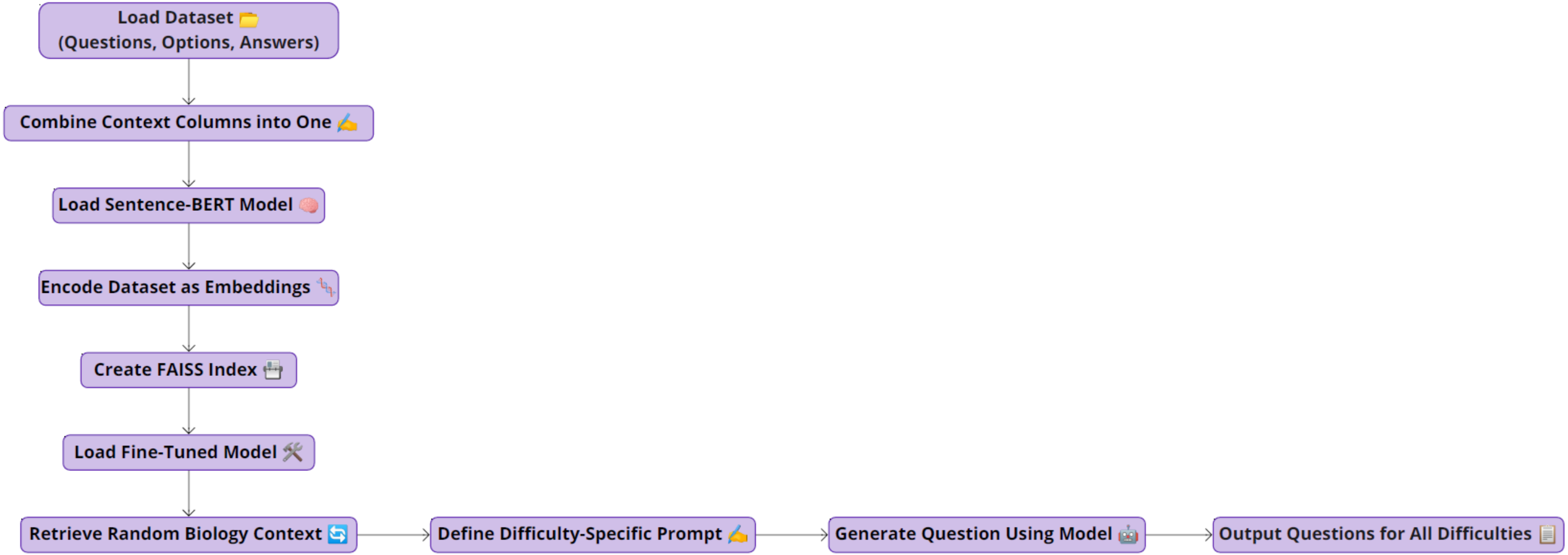


Fine-tuning the LLM

```python
# Configure bitsandbytes
compute_dtype = getattr(torch, bnb_4bit_compute_dtype)
bnb_config = BitsAndBytesConfig(
    load_in_4bit=use_4bit,
    bnb_4bit_quant_type=bnb_4bit_quant_type,
    bnb_4bit_compute_dtype=compute_dtype,
    bnb_4bit_use_double_quant=use_nested_quant,
)

# GPU compatibility check for bfloat16
if compute_dtype == torch.float16 and use_4bit:
    major, _ = torch.cuda.get_device_capability()
    if major >= 8:
        print("=" * 80)
        print("Your GPU supports bfloat16: accelerate training with bf16=True")
        print("=" * 80)

# Load model and tokenizer
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    quantization_config=bnb_config,
    device_map=device_map
)
model.config.use_cache = False
model.config.pretraining_tp = 1

# Load tokenizer
tokenizer = AutoTokenizer.from_pretrained(model_name, trust_remote_code=True)
tokenizer.pad_token = tokenizer.eos_token
tokenizer.padding_side = "right"

# LoRA configuration
peft_config = LoraConfig(
    lora_alpha=lora_alpha,
    lora_dropout=lora_dropout,
    r=lora_r,
    bias="none",
    task_type="CAUSAL_LM",
)
```

```python
# Set training parameters
training_arguments = TrainingArguments(
    output_dir=output_dir,
    num_train_epochs=num_train_epochs,
    per_device_train_batch_size=per_device_train_batch_size,
    gradient_accumulation_steps=gradient_accumulation_steps,
    optim=optim,
    save_steps=save_steps,
    logging_steps=logging_steps,
    learning_rate=learning_rate,
    weight_decay=weight_decay,
    fp16=fp16,
    bf16=bf16,
    max_grad_norm=max_grad_norm,
    max_steps=max_steps,
    warmup_ratio=warmup_ratio,
    group_by_length=group_by_length,
    lr_scheduler_type=lr_scheduler_type,
    report_to="tensorboard"
)

# Initialize the trainer with the processed dataset and LoRA configuration
trainer = SFTTrainer(
    model=model,
    train_dataset=dataset,
    peft_config=peft_config,
    dataset_text_field="text",
    max_seq_length=None,  # Or specify max sequence length if needed
    tokenizer=tokenizer,
    args=training_arguments,
    packing=False,
)

# Train the model
trainer.train()
```

RAG Implementation



RAG Implementation

```python
import pandas as pd
import numpy as np
import faiss
from transformers import AutoModelForCausalLM, AutoTokenizer
from sentence_transformers import SentenceTransformer

# Load the dataset
file_path = "/content/drive/My Drive/MCQ Question Generation/merged_mcq_dataset.csv"
dataset = pd.read_csv(file_path, encoding="latin1")
dataset.fillna("", inplace=True)

# Combine all context columns into one
dataset['Combined'] = (
    dataset['Question Text'].astype(str) + " " +
    dataset['Option 1'].astype(str) + " " +
    dataset['Option 2'].astype(str) + " " +
    dataset['Option 3'].astype(str) + " " +
    dataset['Option 4'].astype(str) + " " +
    dataset['Option 5'].astype(str)
)

# Load Sentence-BERT for semantic search
semantic_model = SentenceTransformer('all-MiniLM-L6-v2')
embeddings = semantic_model.encode(dataset['Combined'].tolist(), show_progress_bar=True)

# Create a FAISS index
dimension = embeddings.shape[1]
faiss_index = faiss.IndexFlatL2(dimension)
faiss_index.add(np.array(embeddings))

# Load the fine-tuned model
model_name = "/content/drive/My Drive/MCQ Question Generation/saved_model"  # Update this path
tokenizer = AutoTokenizer.from_pretrained(model_name, trust_remote_code=True)
generation_model = AutoModelForCausalLM.from_pretrained(model_name)

# Retrieve diverse biology contexts using FAISS
def retrieve_diverse_contexts_faiss(k=3):
    # Randomly select embeddings from the FAISS index
    random_indices = np.random.choice(len(dataset), k, replace=False)
    random_embeddings = np.array([embeddings[i] for i in random_indices])

    # Perform a FAISS search on these random embeddings
    _, indices = faiss_index.search(random_embeddings, k=1)  # Retrieve nearest neighbor for each
    return dataset.iloc[indices.flatten()]['Combined'].tolist()
```

```python
# Define the function to generate questions for all difficulties
def generate_biology_questions():
    difficulties = ["easy", "medium", "hard"]
    questions = {}

    for difficulty in difficulties:
        # Retrieve a random biology context
        retrieved_context = retrieve_context_biology(k=1)
        context = " ".join(retrieved_context)

        # Create a biology-specific prompt
        # Create a refined biology-specific prompt with difficulty-specific instructions
        prompt = f"""
        Based on the following biology context:
        {context}

        Your task is to generate a **biology multiple-choice question** for the **{difficulty} level**:

        - **For Easy Level**: The question should test basic biology concepts, simple definitions, or fundamental facts that are straightforward and easy to recall.
        - **For Medium Level**: The question should involve intermediate biology concepts, processes, or applications that require some reasoning or understanding of relationships between concepts.
        - **For Hard Level**: The question should test advanced biology concepts, detailed mechanisms, or require critical thinking and analysis of biological principles.

        - Provide exactly **five distinct answer options** labeled a, b, c, d, and e.
        - Only one answer option should be correct.
        - Clearly indicate the correct answer.

        Please output in the following format only:
        - Question: <Your question>
        - a) <Option 1>
        - b) <Option 2>
        - c) <Option 3>
        - d) <Option 4>
        - e) <Option 5>
        - Correct Answer: <Correct option letter>

        Do not include any additional text, explanations, or examples.
        """

        # Tokenize and generate
        input_ids = tokenizer.encode(prompt, return_tensors="pt", padding=True, truncation=True)
        output = generation_model.generate(input_ids, max_length=500, temperature=0.7, top_k=50, top_p=0.85)

        # Decode the response
        question = tokenizer.decode(output[0], skip_special_tokens=True)
        questions[difficulty] = question

    return questions

# Example usage
questions = generate_biology_questions()

# Print the questions
for difficulty, question in questions.items():
    print(f"Difficulty: {difficulty.capitalize()}\n{question}\n")
```

SLIIT
FACULTY OF COMPUTING

IT21264634 |Sujitha. S | 24-25J-257

Completion of the project

Key Functionalities

The back-end generates the first quiz with an equal difficulty distribution. Subsequent quizzes adjust difficulty dynamically based on student performance for personalized learning.

- **FAISS (Facebook AI Similarity Search)** – Retrieves context from the dataset for new question generation and detects duplicates, ensuring unique and diverse MCQs.
- **SentenceTransformer ("all-MiniLM-L6-v2")** – Converts questions and answers into vector embeddings for semantic comparison and retrieval of similar past questions.
- **Item Response Theory (IRT)** – Adjusts quiz difficulty dynamically based on student performance, ensuring adaptive learning.
- **Cosine Similarity** – Filters out duplicate or highly similar MCQs within the same quiz and across past quizzes.
- **Student Performance Tracking** – Records accuracy, response time, and consistency to monitor learning progress.

adaptive_quiz_routes.py

```python
import uuid
import logging
import time
from bson import ObjectId
from fastapi import APIRouter, HTTPException, Depends
from utils.user_mgmt_methods import get_current_user
from database.database import quizzes_collection, users_collection
from utils.quiz_generation_methods import fetch_questions_from_db, get_irt_based_difficulty_distribution
from utils.generate_question import generate_mcq_based_on_performance
import traceback
import sys

router = APIRouter()

# Logging configuration
logging.basicConfig(level=logging.INFO, format="%(asctime)s - %(levelname)s - %(message)s")

@router.get("/generate_adaptive_mcqs/{user_id}/{question_count}")
def generate_next_quiz(user_id: str, question_count: int, current_user: str = Depends(get_current_user)):
    """Generate a new adaptive quiz based on user's previous performance."""
    try:
        logging.info(f"📝 Starting adaptive quiz generation for user {user_id} with {question_count} questions.
        sys.stdout.flush()  # Force log flushing

        existing_user = users_collection.find_one({"_id": ObjectId(user_id)})
        if not existing_user:
            logging.error(" User not found")
            sys.stdout.flush()
            raise HTTPException(status_code=404, detail="User not found. Please register before generating a qu

        if current_user != user_id:
            logging.error(" Unauthorized access")
            sys.stdout.flush()
            raise HTTPException(status_code=403, detail="Unauthorized access")

        logging.info(" User found, fetching past performance...")
        sys.stdout.flush()

        difficulty_distribution = get_irt_based_difficulty_distribution(user_id, question_count)
        logging.info(f"📊 Difficulty distribution: {difficulty_distribution}")
        sys.stdout.flush()

        current_quiz_questions = set()
        mcqs = []

        for difficulty, count in difficulty_distribution.items():
            generated = 0
            failed_attempts = 0
            while generated < count:
                logging.info(f"⚙️ Generating MCQ (Difficulty: {difficulty}) - Attempt {generated + 1}/{count}")
                sys.stdout.flush()

                mcq = generate_mcq_based_on_performance(user_id, difficulty)
```

```python
                logging.info(f"📩 Received MCQ response: {mcq}")
                sys.stdout.flush()

                if not mcq or not isinstance(mcq, dict) or "question" not in mcq:
                    failed_attempts += 1
                    logging.warning(f"⚠️ Failed MCQ generation for {difficulty}. Retrying... ({failed_attempts}/3)")
                    sys.stdout.flush()

                    if failed_attempts >= 3:
                        logging.error(f" Skipping {difficulty} MCQ after 3 failed attempts.")
                        sys.stdout.flush()
                        break  # Move to the next difficulty level

                    continue  # Retry another question

                mcq_text = mcq["question"]

                if mcq_text in current_quiz_questions:
                    logging.warning(f" Duplicate detected: {mcq_text}, retrying...")
                    failed_attempts += 1
                    sys.stdout.flush()
                    continue

                formatted_mcq = {
                    "question_text": mcq_text,
                    "option1": mcq.get("options", {}).get("A", "N/A"),
                    "option2": mcq.get("options", {}).get("B", "N/A"),
                    "option3": mcq.get("options", {}).get("C", "N/A"),
                    "option4": mcq.get("options", {}).get("D", "N/A"),
                    "option5": mcq.get("options", {}).get("E", "N/A"),
                    "correct_answer": mcq.get("correct_answer", "N/A"),
                    "difficulty": difficulty
                }

                mcqs.append(formatted_mcq)
                current_quiz_questions.add(mcq_text)
                generated += 1
                sys.stdout.flush()

        if len(mcqs) < question_count:
            remaining_needed = question_count - len(mcqs)
            logging.warning(f"⚠ Not enough questions generated. Fetching {remaining_needed} from DB.")
            sys.stdout.flush()

            db_questions = fetch_questions_from_db(remaining_needed)
            mcqs.extend(db_questions)

        quiz_id = str(uuid.uuid4())
        quiz_data = {
            "quiz_id": quiz_id,
            "user_id": user_id,
            "difficulty_distribution": difficulty_distribution,
```

IRT Based difficulty Distribution

```python
# Method to get IRT-based difficulty distribution for a user
def get_irt_based_difficulty_distribution(user_id, total_questions):
    """Dynamically adjusts quiz difficulty based on user performance trend and API success rate."""

    #  🔹 Fetch recent quiz performance (last 3 quizzes)
    recent_quizzes = list(quizzes_collection.find(
        {"user_id": ObjectId(user_id)},
        {"difficulty_distribution": 1, "questions": 1}
    ).sort("created_at", -1).limit(3))

    # Track how many correct answers per difficulty in recent quizzes
    correct_easy = correct_medium = correct_hard = 0
    total_easy = total_medium = total_hard = 1  # Avoid division by zero

    for quiz in recent_quizzes:
        for question in quiz["questions"]:
            difficulty = question.get("difficulty", "medium")
            is_correct = question.get("user_answered_correctly", False)

            if difficulty == "easy":
                total_easy += 1
                correct_easy += int(is_correct)
            elif difficulty == "medium":
                total_medium += 1
                correct_medium += int(is_correct)
            elif difficulty == "hard":
                total_hard += 1
                correct_hard += int(is_correct)

    # Compute accuracy per difficulty level
    easy_accuracy = correct_easy / total_easy
    medium_accuracy = correct_medium / total_medium
    hard_accuracy = correct_hard / total_hard

    # Base ratios on user's performance
    if easy_accuracy > 0.8:  # User is doing well on easy
        easy_ratio = 0.1
    elif easy_accuracy < 0.5:
        easy_ratio = 0.4
    else:
        easy_ratio = 0.2
```

```python
    if medium_accuracy > 0.7:
        medium_ratio = 0.5
    elif medium_accuracy < 0.4:
        medium_ratio = 0.3
    else:
        medium_ratio = 0.4

    if hard_accuracy > 0.6:  # User is improving in hard
        hard_ratio = 0.4
    else:
        hard_ratio = 0.3 if hard_accuracy < 0.4 else 0.2  # Reduce hard if they are failing too much

    # Ensure total = 100%
    total_ratio = easy_ratio + medium_ratio + hard_ratio
    easy_ratio /= total_ratio
    medium_ratio /= total_ratio
    hard_ratio /= total_ratio

    return {
        "easy": round(easy_ratio * total_questions),
        "medium": round(medium_ratio * total_questions),
        "hard": total_questions - (round(easy_ratio * total_questions) + round(medium_ratio * total_questions))
    }
```

Other Functionalities

User Performance Dashboard: Displays quiz history, accuracy trends, response times, and strongest/weakest topics.

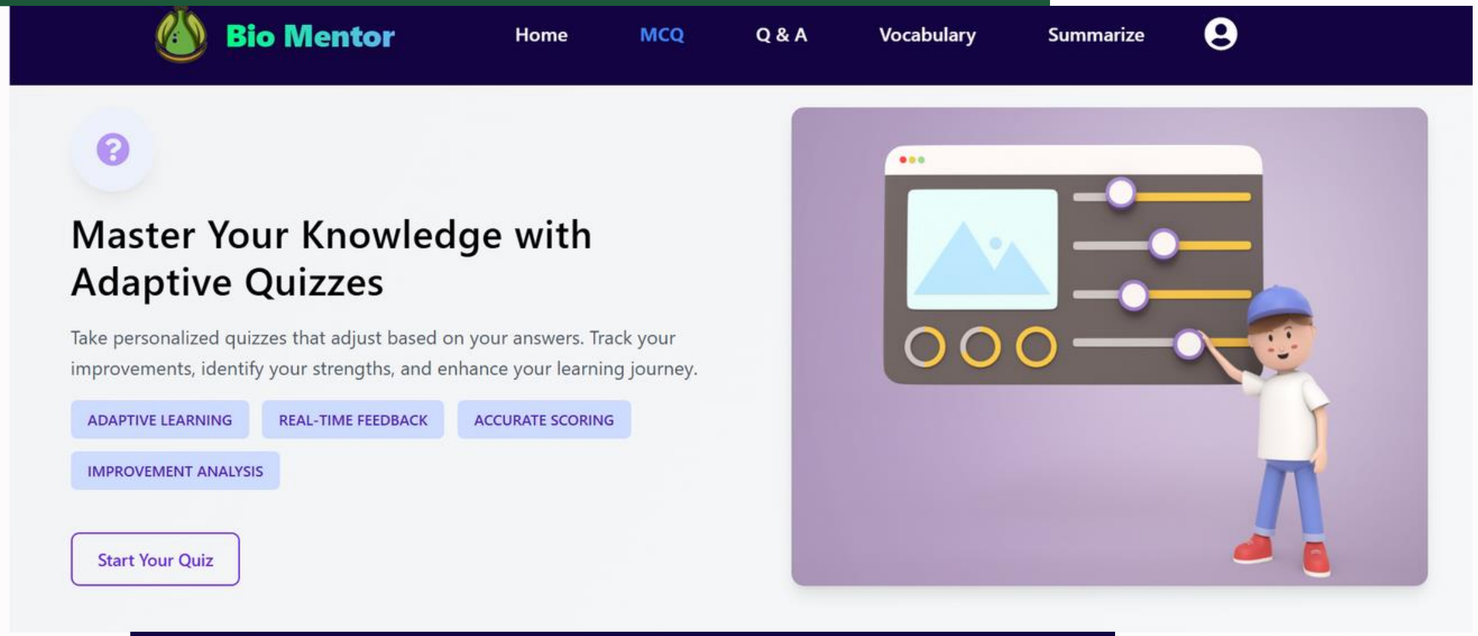
Unit-Based Quizzes: Allows users to do unit based MCQ Questions
Retake Adaptive Quizzes (Excluded from Performance Calculation)

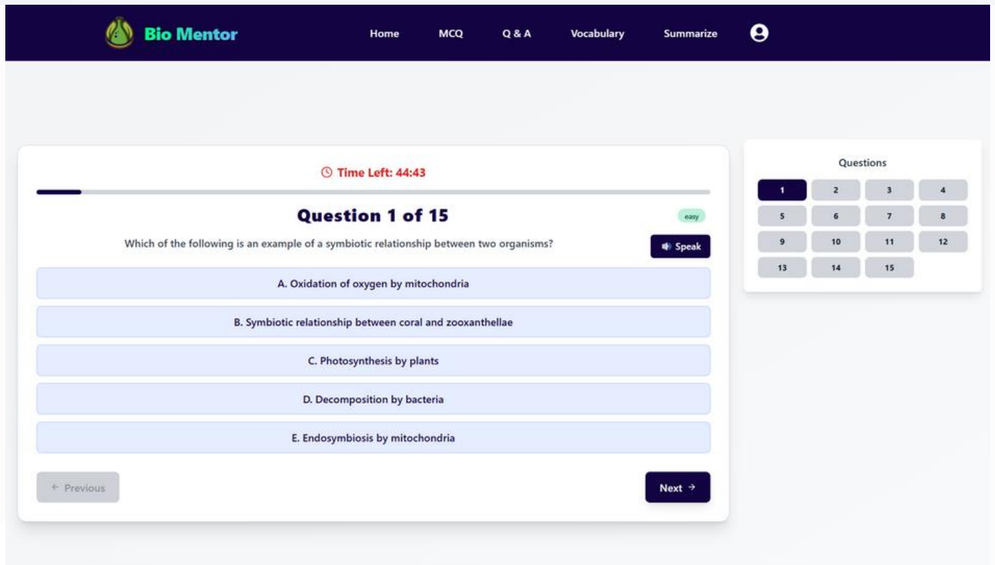
Retake Adaptive Quizzes (Excluded from Performance Calculation): Users can retake adaptive quizzes to improve understanding without affecting their official performance metrics.

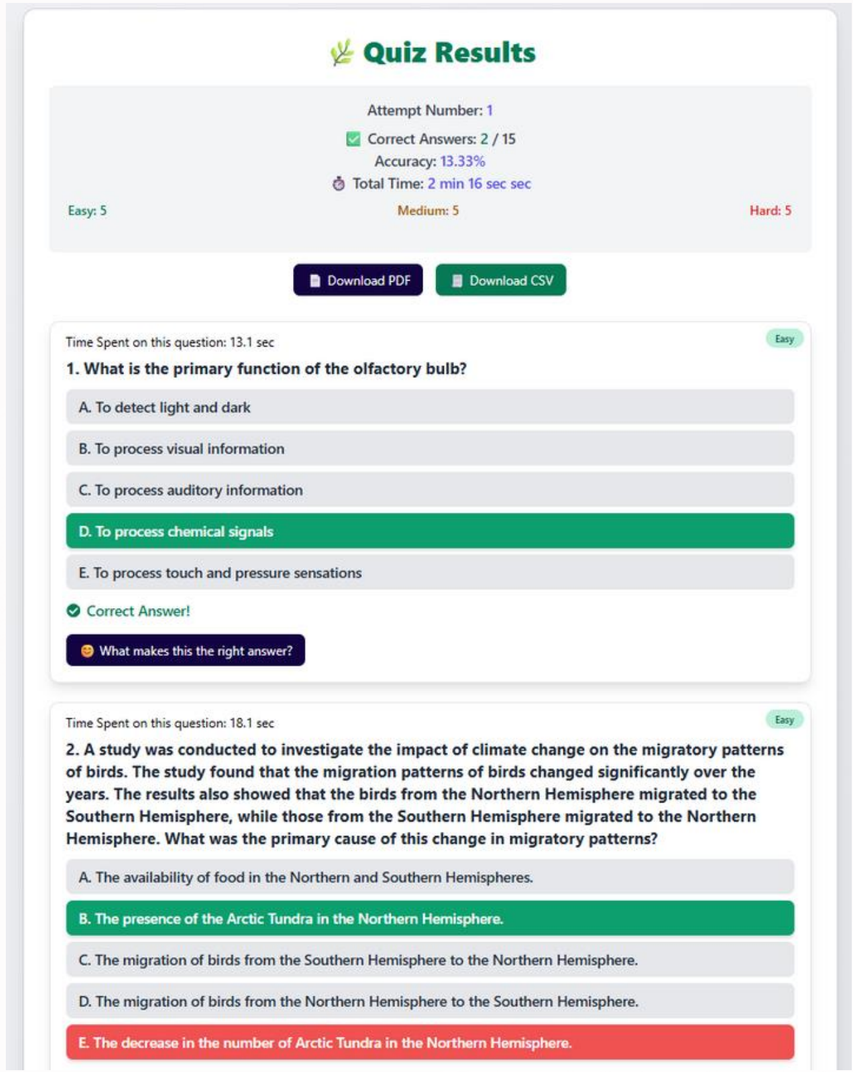
```python
@router.get("/performance_graph/{user_id}")
def generate_performance_graph(user_id: str,current_user: str = Depends(get_current_user)):
    """Generates graphs showing user improvement and consistency."""
    user_data = users_collection.find_one({"_id": ObjectId(user_id)})
    if not user_data:
            raise HTTPException(status_code=404, detail="User not found.")

    if current_user != user_id:
            logging.error(f" Unauthorized access attempt by {current_user}")
            raise HTTPException(status_code=403, detail="Unauthorized access")

    if not user_data or "performance" not in user_data or "last_10_quizzes" not in user_data["performance"]:
        return {
            "quiz_numbers": [],
            "scores": [],
            "graph_image": None,
            "message": "No quiz performance data available. Start taking quizzes to see your progress!"
        }
    history = user_data["performance"]["last_10_quizzes"]
```

```python
quiz_numbers = list(range(1, len(history) + 1))
scores = [quiz["accuracy"] for quiz in history]

plt.figure(figsize=(8, 4))
plt.plot(quiz_numbers, scores, marker="o", linestyle="-", color="b", label="Accuracy (%)")
plt.xlabel("Quiz Attempt")
plt.ylabel("Score (%)")
plt.title("User Performance Over Time")
plt.legend()

img = io.BytesIO()
plt.savefig(img, format="png")
img.seek(0)
return {
    "quiz_numbers": quiz_numbers,
    "scores": scores,
    "graph_image": base64.b64encode(img.getvalue()).decode(),
    "message": "Performance data loaded successfully."
}
```

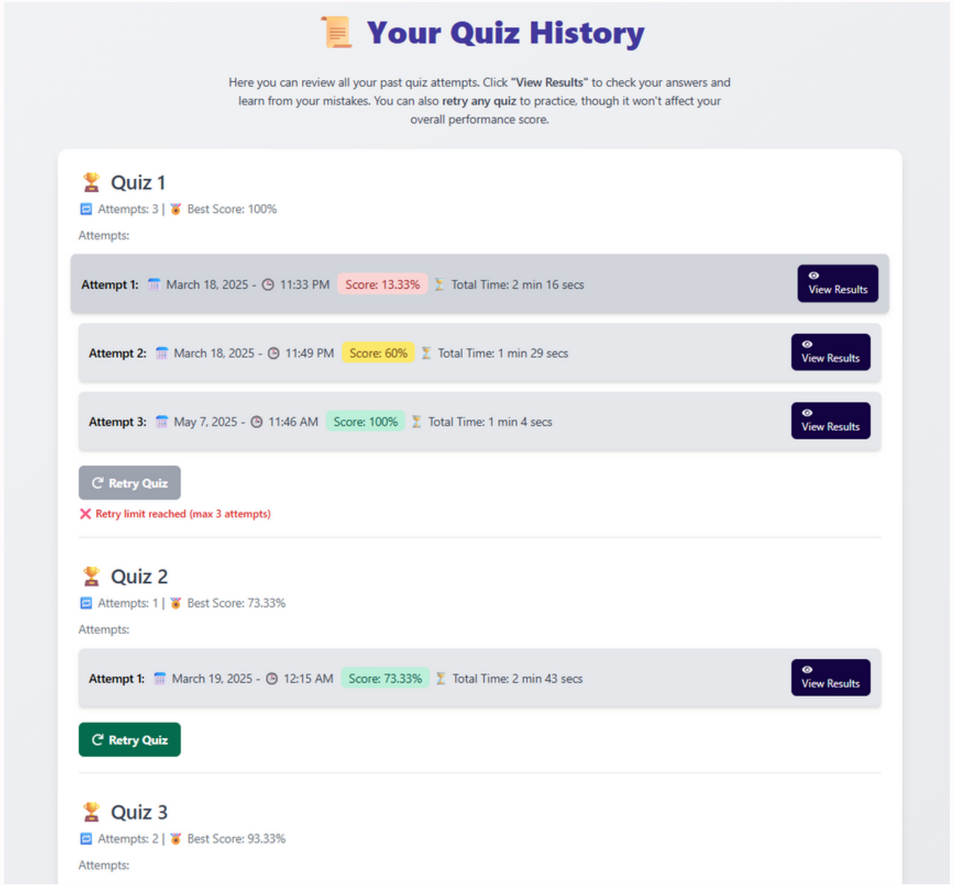
Working Product

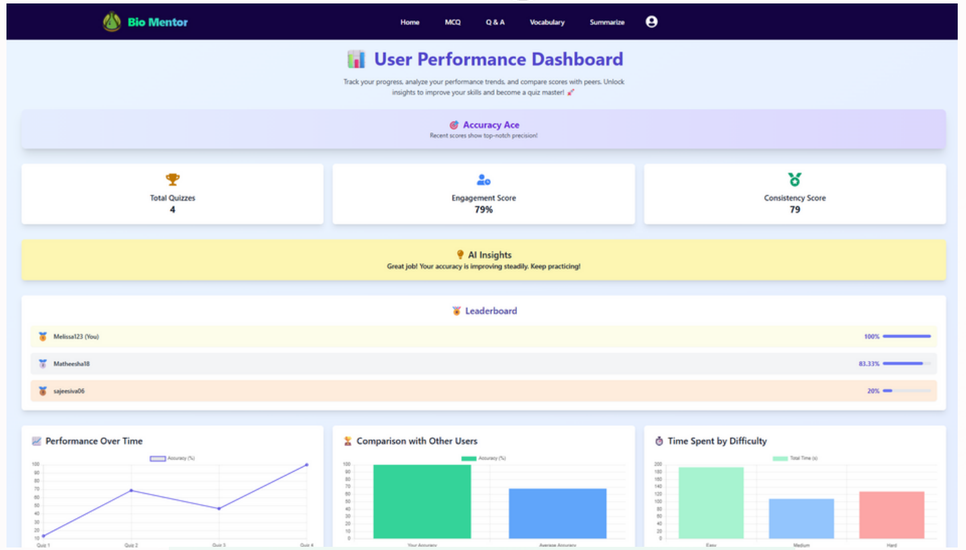


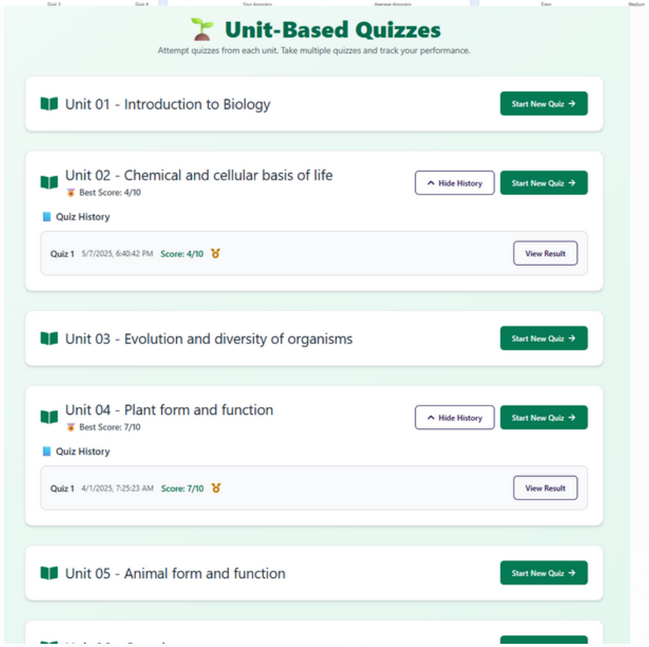




Working Product





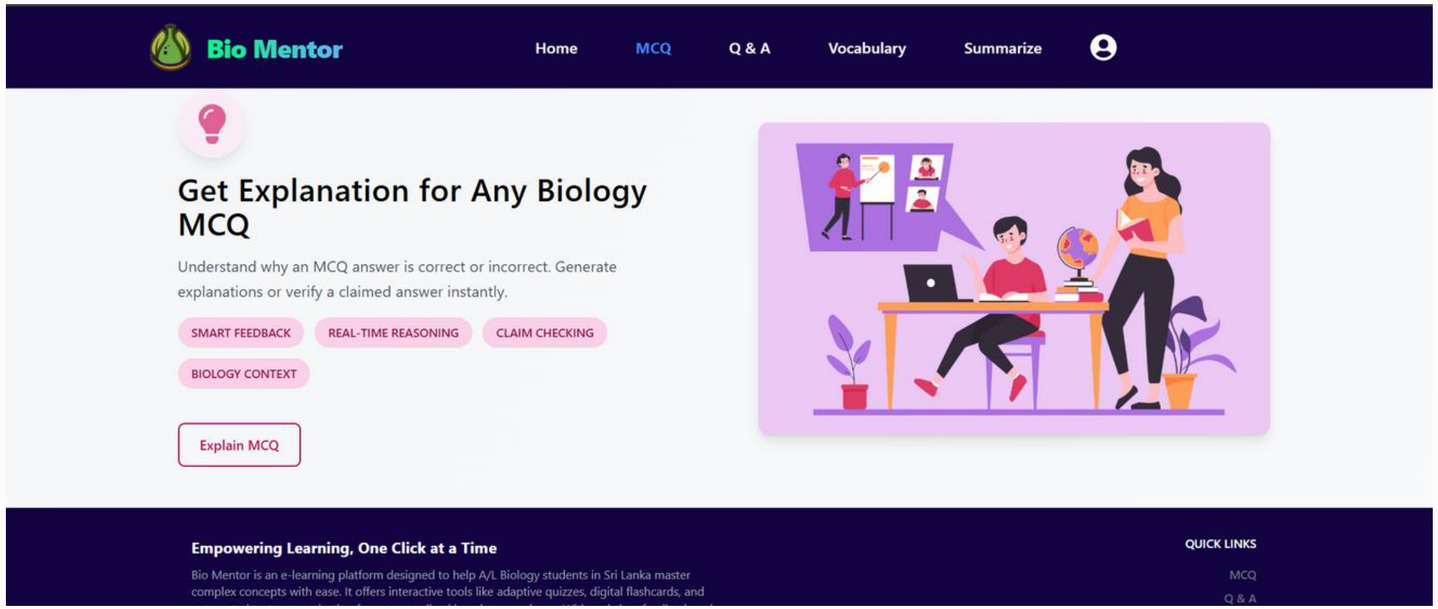


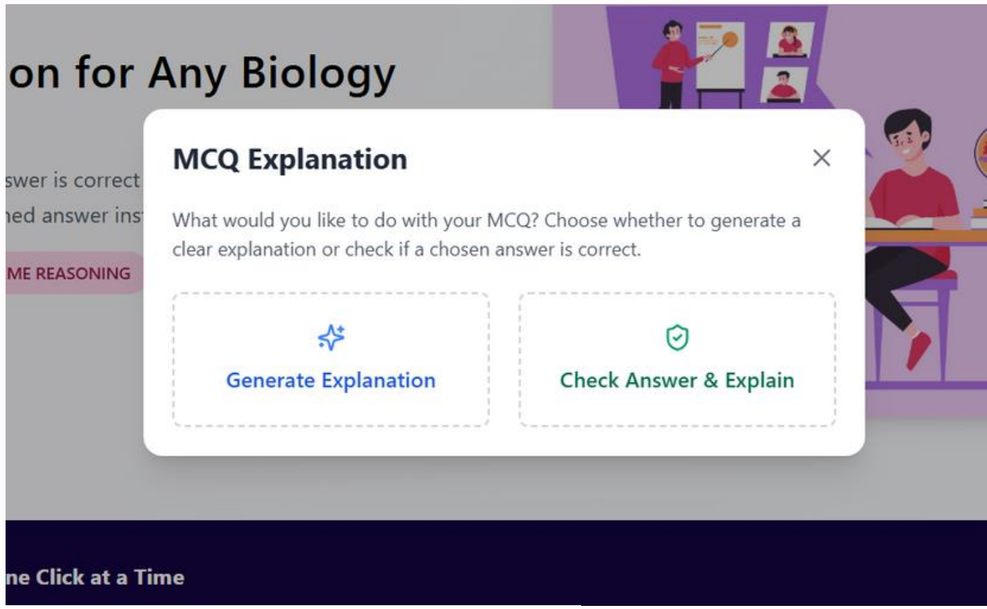
Completion of the project

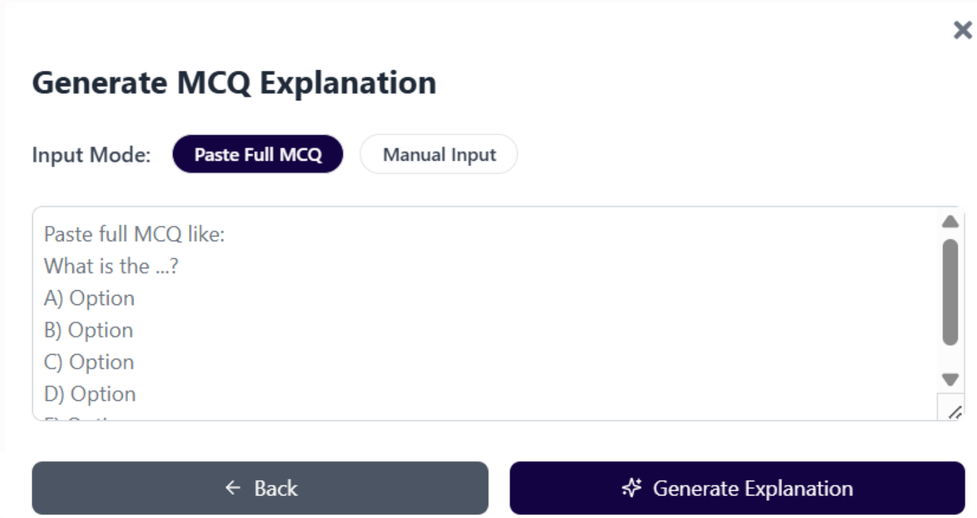
🔧 Final Enhancements After 90% Viva

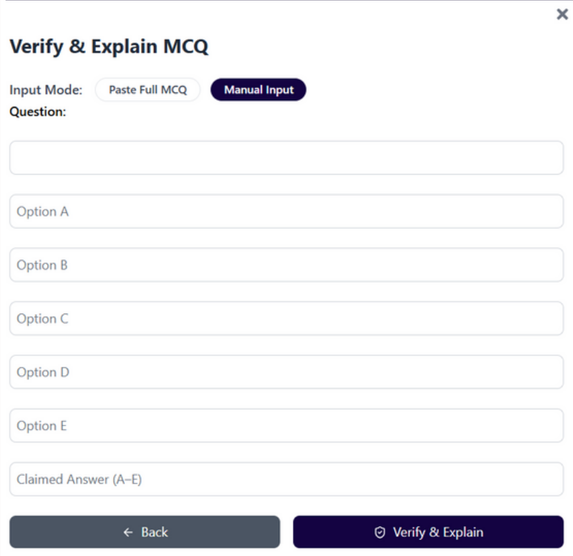
- Increased total quiz questions for deeper evaluation
- Prompt generates max 3 MCQs to avoid redundancy in context
- Improved MCQ extraction to handle multiple questions & formats
- Converted model to .gguf Q8 for resource efficiency with minimal loss
- Added:
    1) Answer Explanation component
    2) Answer Verification mechanism

Working Product









Testing - Unit Testing

Framework: Pytest
Purpose: Test individual functions in isolation using mocks.

Modules & Focus Areas:

1) test_text_parsing_utils.py
   - Tests MCQ extraction and answer cleaning logic
2)test_generate_mcq.py
   - Tests MCQ generation with mocked model, FAISS, and embeddings
3)test_irt.py
   - Verifies IRT logic for difficulty and discrimination values

```
# Dynamically add the absolute path to project root (MCQ) to sys.path
sys.path.append(os.path.abspath(os.path.join(os.path.dirname(__file__),
'..')))
dummy_user_id = str(ObjectId())

from utils.quiz_generation_methods import (
    assign_difficulty_parameter,
    assign_discrimination_parameter,
    get_irt_based_difficulty_distribution
)

def test_assign_difficulty_parameter_easy_range():
    value = assign_difficulty_parameter(dummy_user_id, "easy")
    assert -2.0 <= value <= 1.0

def test_assign_difficulty_parameter_hard_range():
    value = assign_difficulty_parameter(dummy_user_id, "hard")
    assert isinstance(value, float)

def test_assign_discrimination_parameter_range():
    value = assign_discrimination_parameter()
    assert 0.5 <= value <= 2.0

def test_get_irt_based_difficulty_distribution_low_theta():
    distribution = get_irt_based_difficulty_distribution(dummy_user_id, 10)
    assert sum(distribution.values()) == 10
    assert isinstance(distribution, dict)

def test_get_irt_based_difficulty_distribution_high_theta():
    distribution = get_irt_based_difficulty_distribution(dummy_user_id, 10)
    assert sum(distribution.values()) == 10
    assert isinstance(distribution, dict)
```

```
def test_extract_mcqs_parsing_single_question():
    prompt = "Ignore this prompt."
    raw_output = """
    Question 1: What is the powerhouse of the cell?
    A) Nucleus
    B) Ribosome
    C) Mitochondria
    D) Chloroplast
    E) Endoplasmic Reticulum
    Correct Answer: C
    """
    parsed = extract_mcqs(prompt, raw_output)
    assert len(parsed) == 1
    assert parsed[0]["question"].startswith("What is")
    assert parsed[0]["correct_answer"] == "C"
    assert "A" in parsed[0]["options"]
    assert len(parsed[0]["options"]) == 5


@pytest.mark.parametrize("raw_input, expected", [
    ("Correct Answer: A", ["A"]),
    ("Answer: C and D", ["C", "D"]),
    ("B, D", ["B", "D"]),
    ("(E)", ["E"]),
    ("A, A, C", ["A", "C"]),
    ("123", []),
    ("", []),
    ("Correct Answer: A, C, E", ["A", "C", "E"]),
])
def test_clean_correct_answer_letters(raw_input, expected):
```

Testing - Integration Testing

Frameworks: Pytest, FastAPI TestClient
Purpose: Validate full workflows across routes, DB, and logic layers.

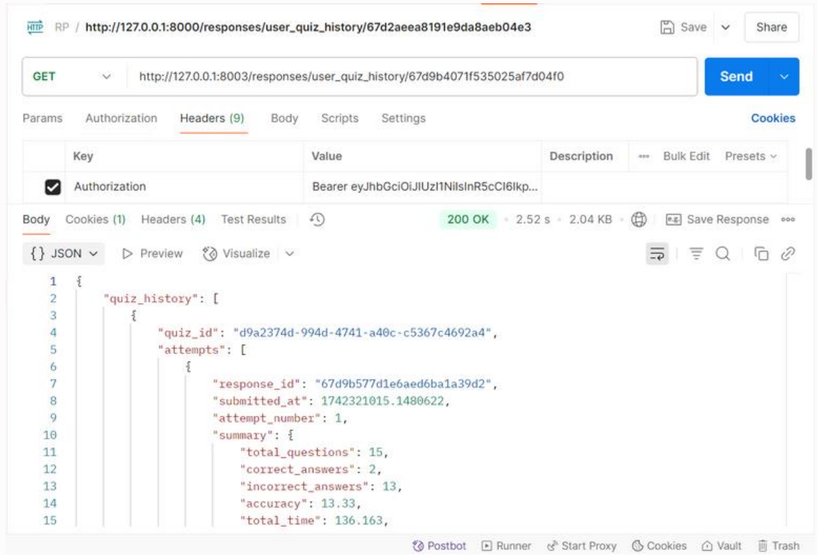
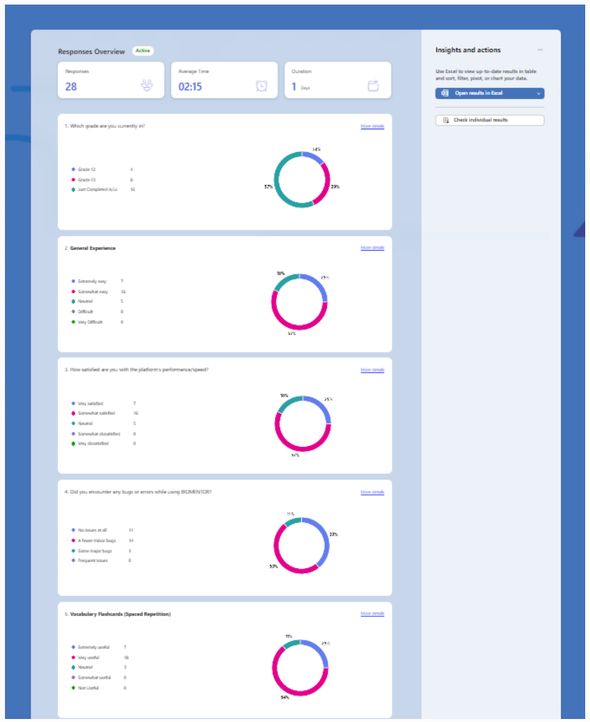
Modules & Focus Areas:

1) test_adaptive_routes.py
   - End-to-end adaptive quiz generation and submission
   - Endpoints: /generate_adaptive_mcqs, /submit_quiz, /get_quiz

2)test_routes.py
   - Unit quiz generation and performance routes
   - Endpoints: /unit_quiz/generate, /unit_quiz/submit, /dashboard_data

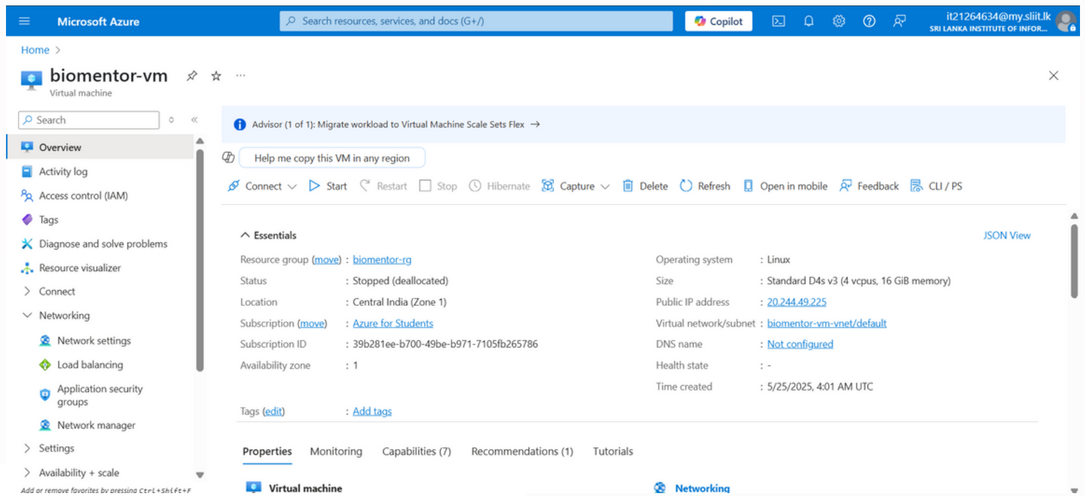
```python
import pytest
from fastapi.testclient import TestClient
from main import app
from database.database import users_collection, quizzes_collection
from bson import ObjectId
from utils.user_mgmt_methods import get_current_user

TEST_USER_ID = "000000000000000000000003"

def override_get_current_user():
    return TEST_USER_ID

app.dependency_overrides[get_current_user] = override_get_current_user
client = TestClient(app)

@pytest.fixture(scope="module", autouse=True)
def setup_user():
    users_collection.insert_one({
        "_id": ObjectId(TEST_USER_ID),
        "username": "test_user",
        "performance": {
            "total_quizzes": 1,
            "accuracy_easy": 80,
            "accuracy_medium": 70,
            "accuracy_hard": 60,
            "time_easy": 5,
            "time_medium": 5,
            "time_hard": 5,
            "strongest_area": "easy",
            "weakest_area": "hard",
            "consistency_score": 90,
            "last_10_quizzes": [{
                "accuracy": 75,
                "total_time": 45,
                "timestamp": 1712724584.0
            }]
        }
    })
    yield
    users_collection.delete_many({"_id": ObjectId(TEST_USER_ID)})

def test_generate_mcqs():
    response = client.get(f"/generate_mcqs/{TEST_USER_ID}")
    assert response.status_code in [200, 404]
    if response.status_code == 200:
        data = response.json()
        assert "quiz_id" in data
        assert "mcqs" in data

def test_generate_adaptive_mcqs():
    response =
client.get(f"/generate_adaptive_mcqs/{TEST_USER_ID}/5")
    if response.status_code == 200:
        data = response.json()
        assert "quiz_id" in data
        assert "mcqs" in data


def test_submit_quiz_and_get_quiz():
    # First generate quiz
    gen = client.get(f"/generate_mcqs/{TEST_USER_ID}")
    if gen.status_code != 200:
        pytest.skip("MCQ quiz not available for submission test")

    quiz_data = gen.json()
    quiz_id = quiz_data["quiz_id"]
    questions = quiz_data["mcqs"]

    # Submit fake answers
    submission_payload = {
        "user_id": TEST_USER_ID,
        "quiz_id": quiz_id,
        "responses": [
            {
                "question_text": q["question_text"],
                "selected_answer": "A",
                "time_taken": 3.0
            } for q in questions
        ]
    }
    submit = client.post("/submit_quiz/", json=submission_payload)
    assert submit.status_code == 200
    assert "summary" in submit.json()

    # Fetch quiz using the same quiz_id
    fetch = client.get(f"/get_quiz/{quiz_id}")
    assert fetch.status_code == 200
    assert fetch.json()["quiz_id"] == quiz_id
```

Testing - System Testing and UAT

System Testing	User Acceptance Testing
• **Framework:** Postman 	

Deployment

- The MCQ generation backend was deployed on an Azure Virtual Machine using FastAPI and a quantized LLaMA 2 model (.gguf format).
- The backend runs as a persistent systemd service, with Nginx configured as a reverse proxy for public access.
- GitHub Actions CI/CD was integrated to automate deployment on code push.



References

[1] Automatic Question Answer Generation using T5 and NLP. [ONLINE] https://ieeexplore.ieee.org/document/9972639 [Accessed 13 May 2024]

[2] Generation of Multiple-Choice Questions From Textbook Contents of School-Level Subjects. [ONLINE] https://ieeexplore.ieee.org/document/9964056 [Accessed 20 May 2024]

[3] Automatic question generation for intelligent tutoring systems. [ONLINE] https://ieeexplore.ieee.org/document/9972639 [Accessed 06 June 2024]

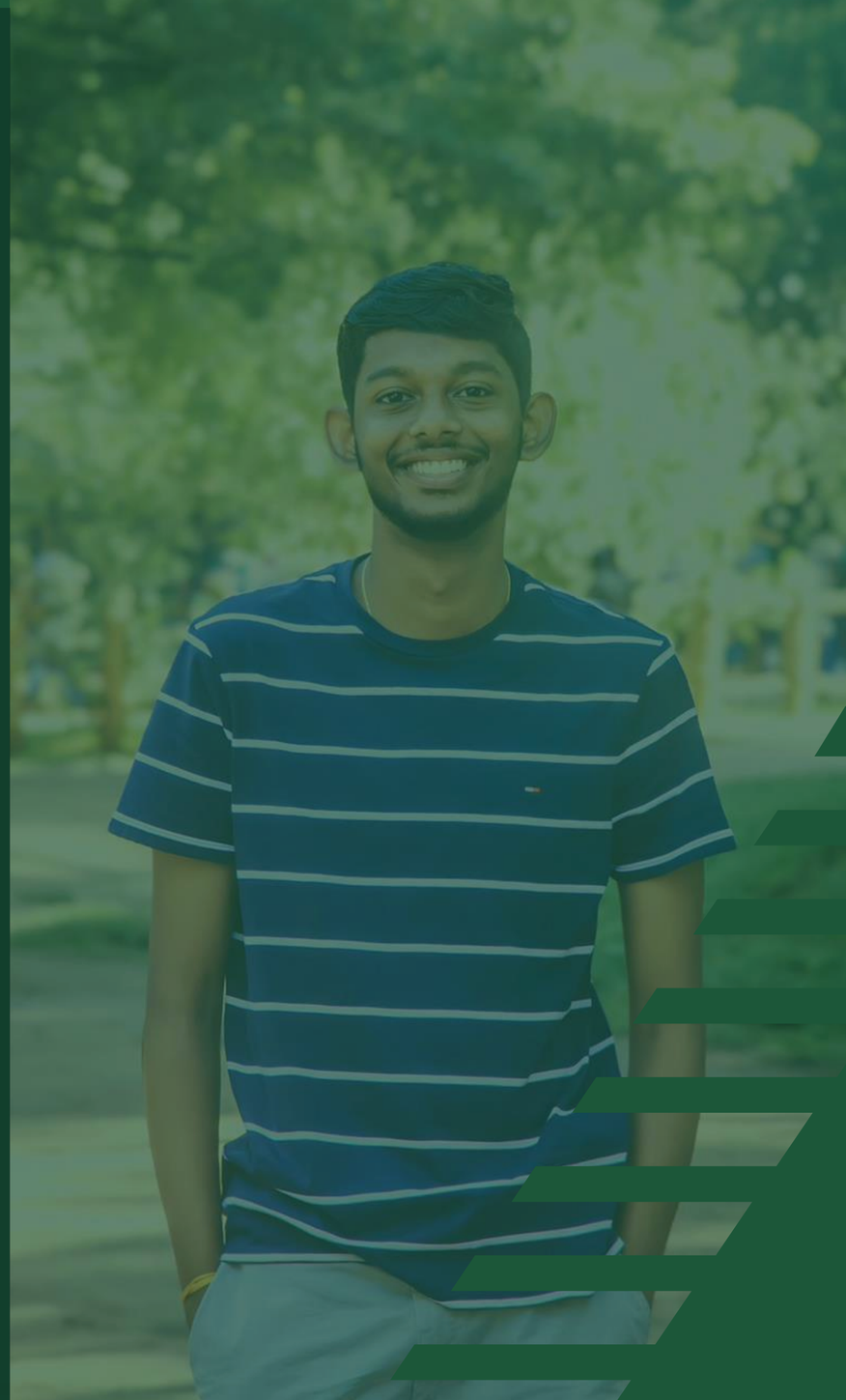
[4] MCQGen: A Large Language Model-Driven MCQ Generator for Personalized Learning. [ONLINE] https://ieeexplore.ieee.org/document/9972639 [Accessed 05 June 2024]

[5] Generation of Multiple Choice Questions from Indian Educational Text. [ONLINE] https://ieeexplore.ieee.org/document/10270551 [Accessed 16 June 2024

IT21204302

Sajeevan S
Software Engineering

LLM BASED PROVIDE ANSWERS FOR STRUCTURED AND ESSAY TYPE OF QUESTIONS AND EVALUATE ANSWERS BASED ON APPROVED RESOURCES.



Introduction

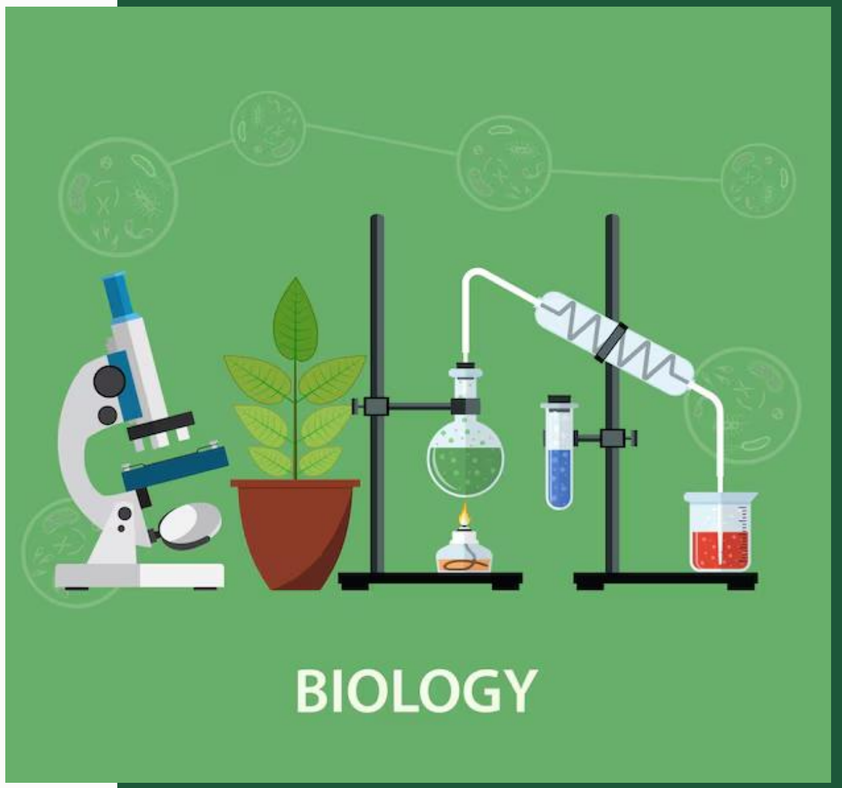
01 Background

02 Research Problem

03 Main and Sub Objectives

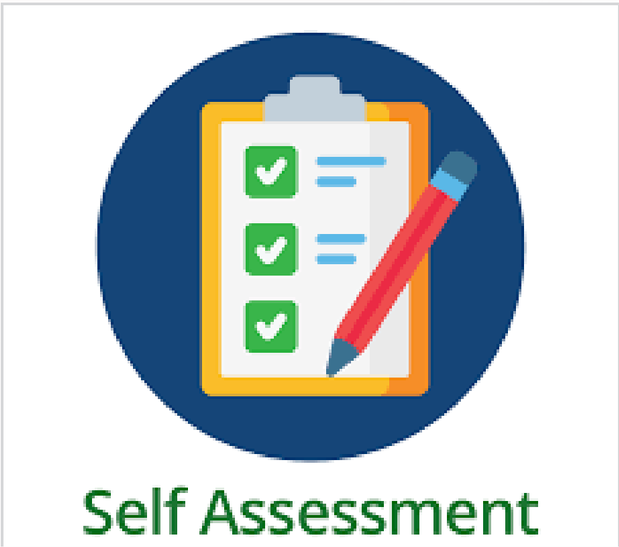
04 Methodology

05 Completion Of Project



BACKGROUND

The Importance of Independent Learning in Modern Education



The Challenges of Self-Evaluation in structured essay-type answering

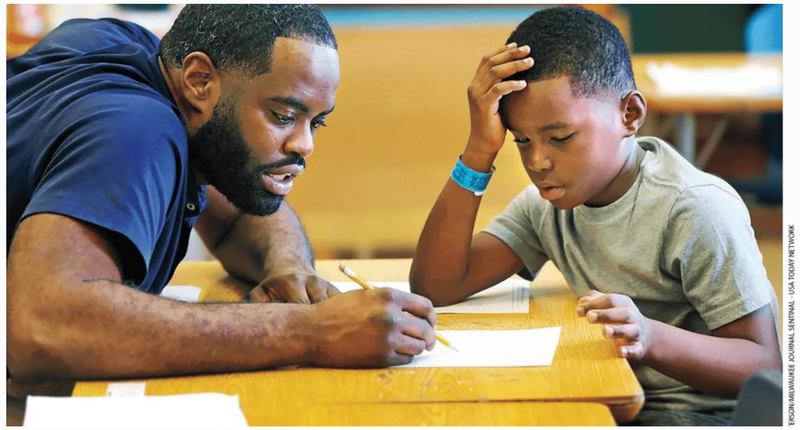
RESEARCH PROBLEM

01

How can technology-based tools be developed and utilized to assist students in improving their essay-writing skills independently, without relying on mentor support?

What strategies can students employ to self-evaluate and improve their structured essay-type answers in the absence of mentor guidance?

02



ERSON/MILWAUKEE JOURNAL SENTINAL - USA TODAY NETWORK

OBJECTIVES

Objective 1

Answer Generation for a Question

Objective 2

Enhance Self-Directed Learning in Advanced-Level Biology

Main Objective

If students provide a question, the system will generate an answer, and if students also provide their corresponding answer, the system will evaluate their response and offer suggestions to improve the answer.

Objective 3

Provide Suggestions for Improvement

Objective 4

Provide Feedback for Answers

IT21204302 |Sajeevan. S | 24-25J-257

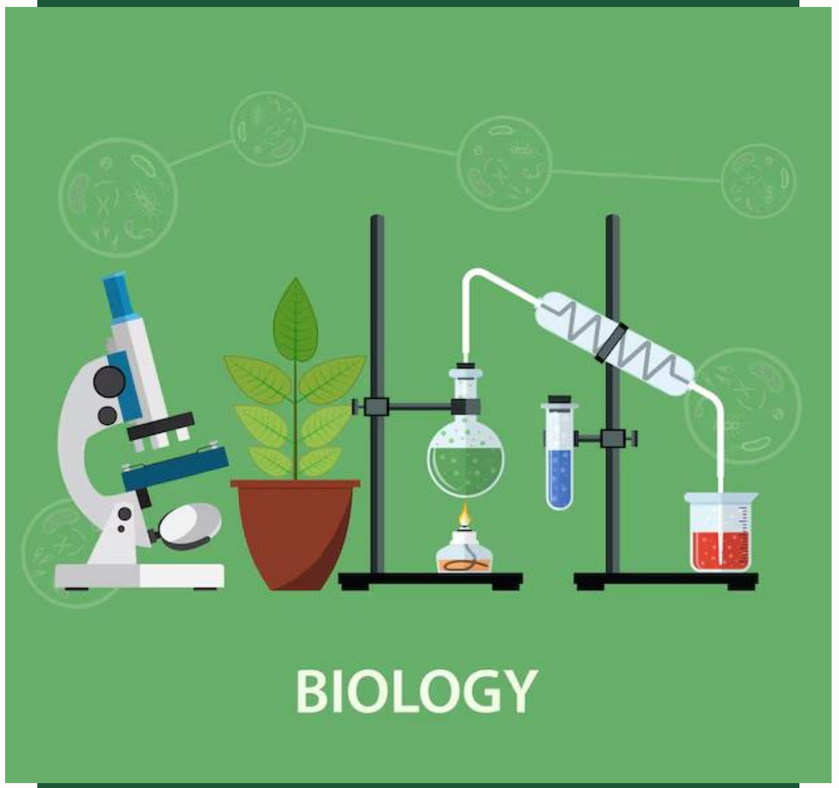
Methodology

01 System Diagram

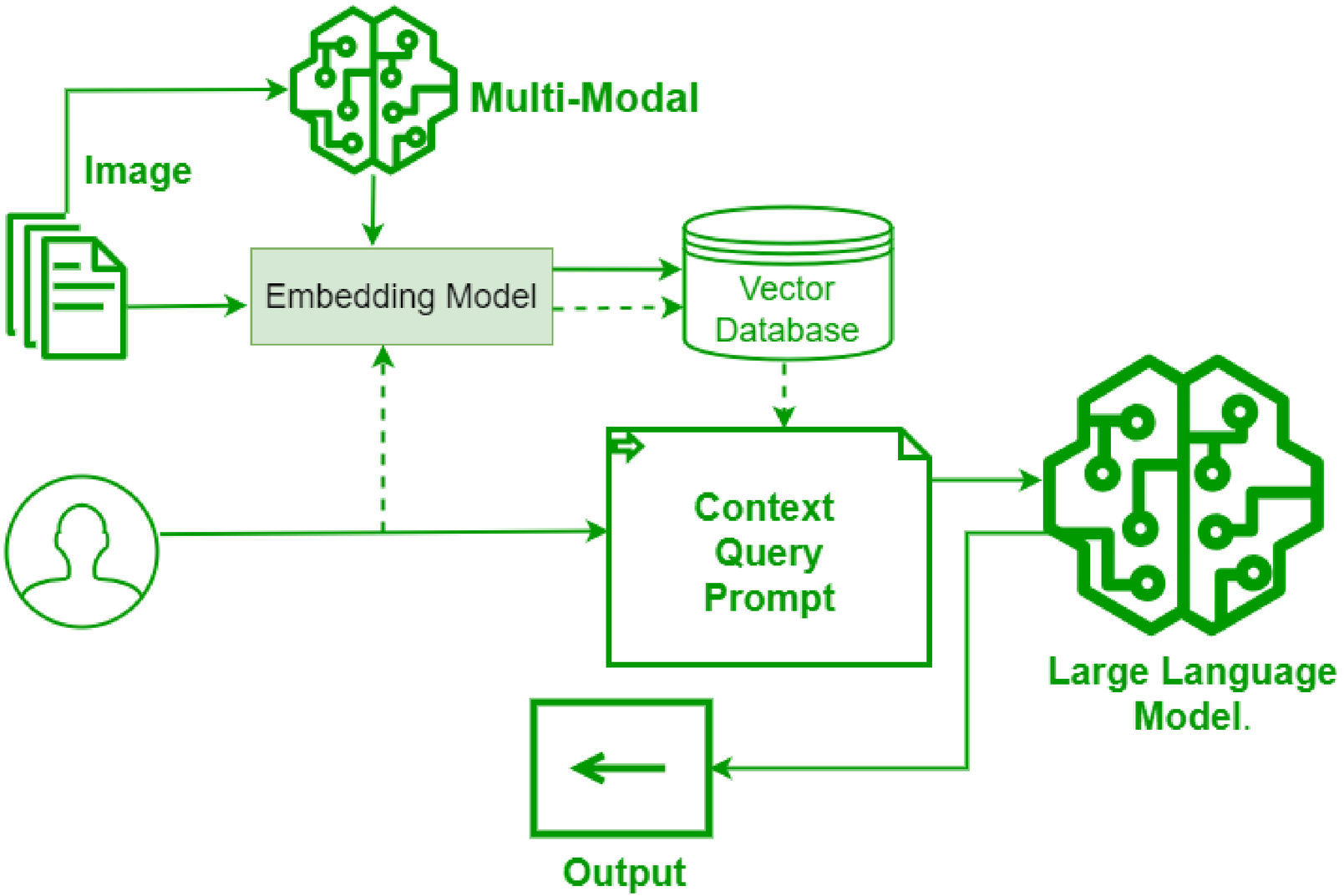
02 Tools and Technologies

03 Best Practices

04 GitHub Commits



System Diagram



Tools & Technologies

Programming Languages

Python
React JS

Database

Faiss
Mongo DB

Project Management

Jira

Frameworks

Transformer model
Flask
Pytorch
OCR

Other tools

Git
Draw.io
Postman

Best Practices

Non-Functional Requirements

- Compatibility
- Accuracy
- Performance
- Usability
- Availability

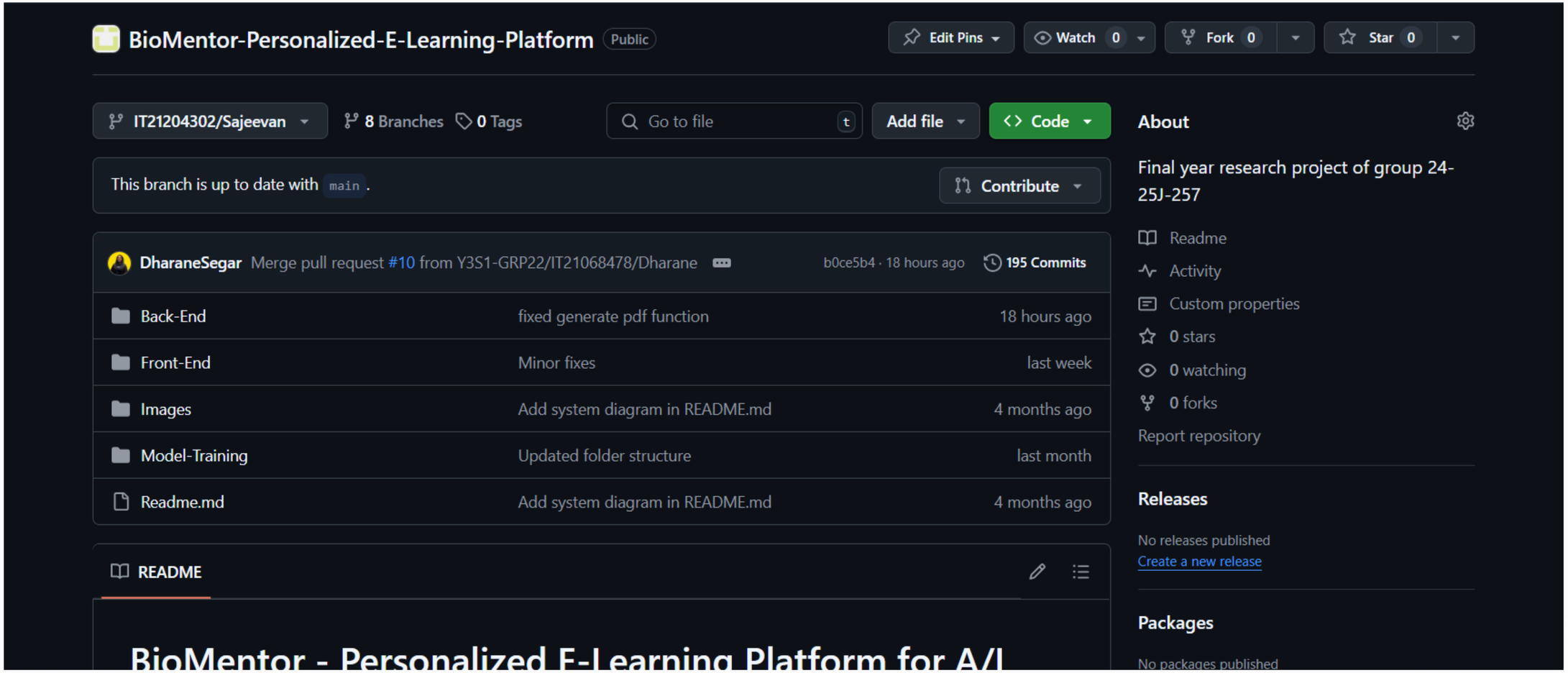
Design Excellence

- Real-Time Performance
- Accuracy and Reliability
- User Experience
- Integration

Standards & Best Practices

- Code Structure & Organization
- Web Security
- Model Quality
- RESTful API Design

GitHub Commits



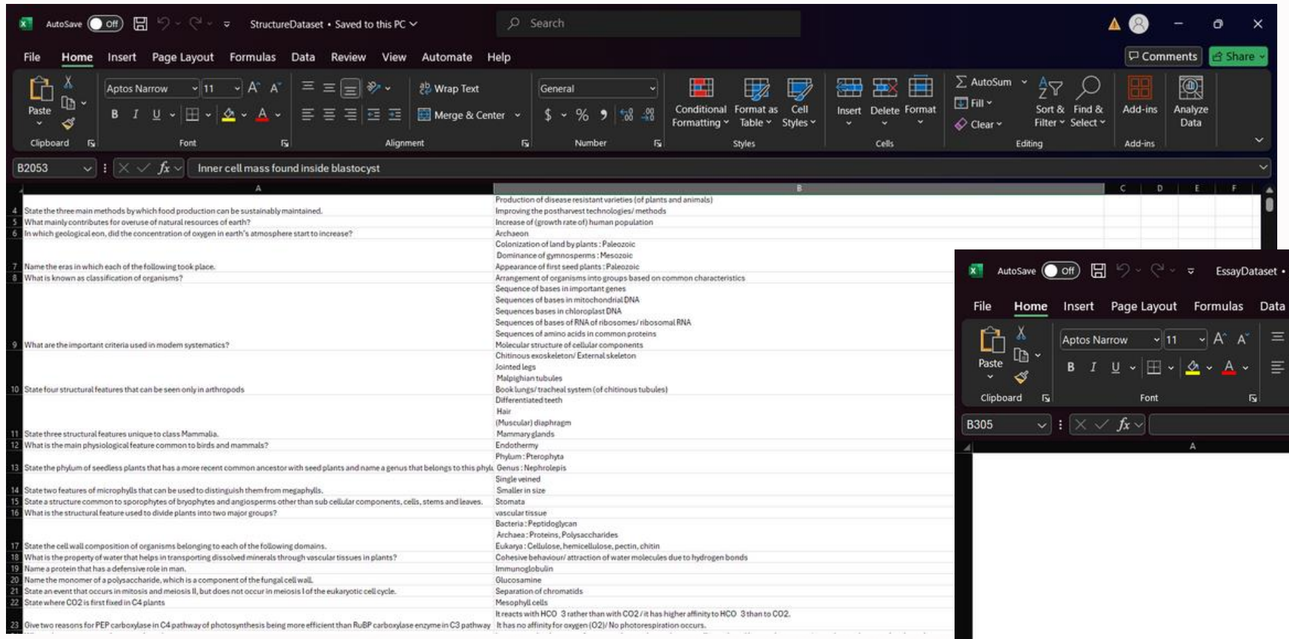
Completion of the project

SLIIT
FACULTY OF COMPUTING

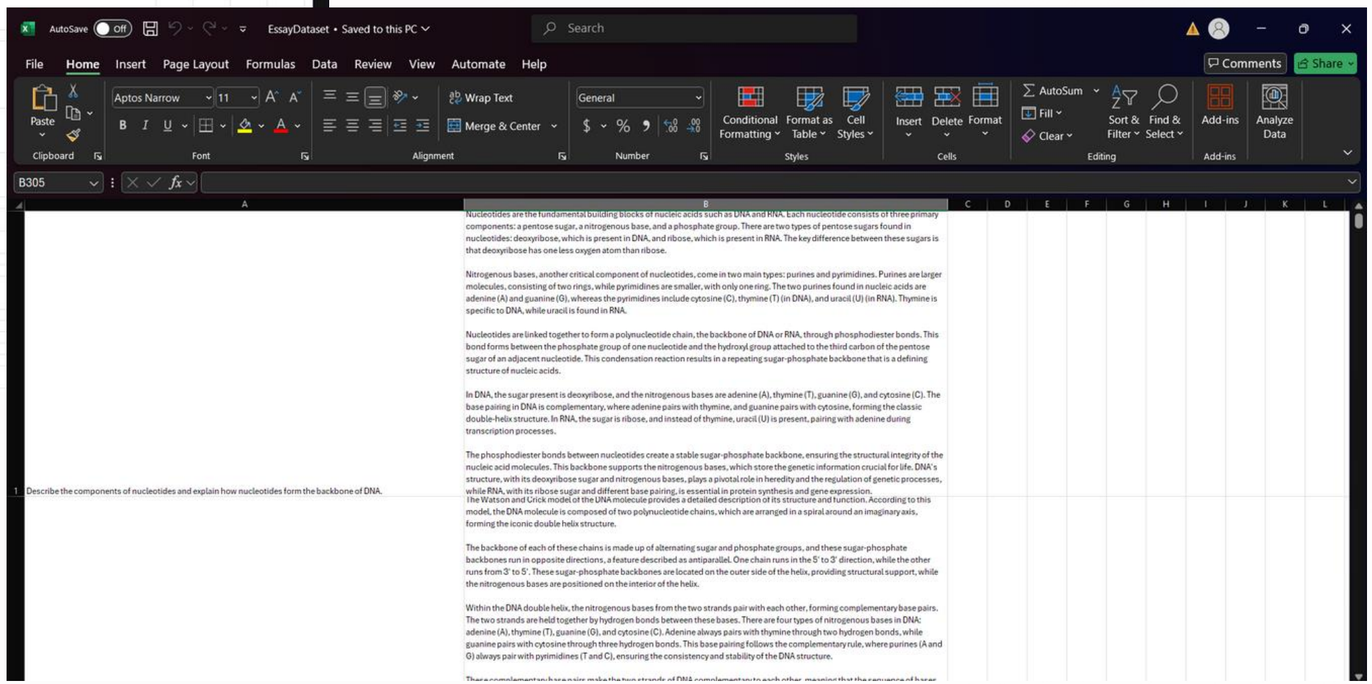
IT21204302 |Sajeevan. S | 24-25J-257

Completion of the project

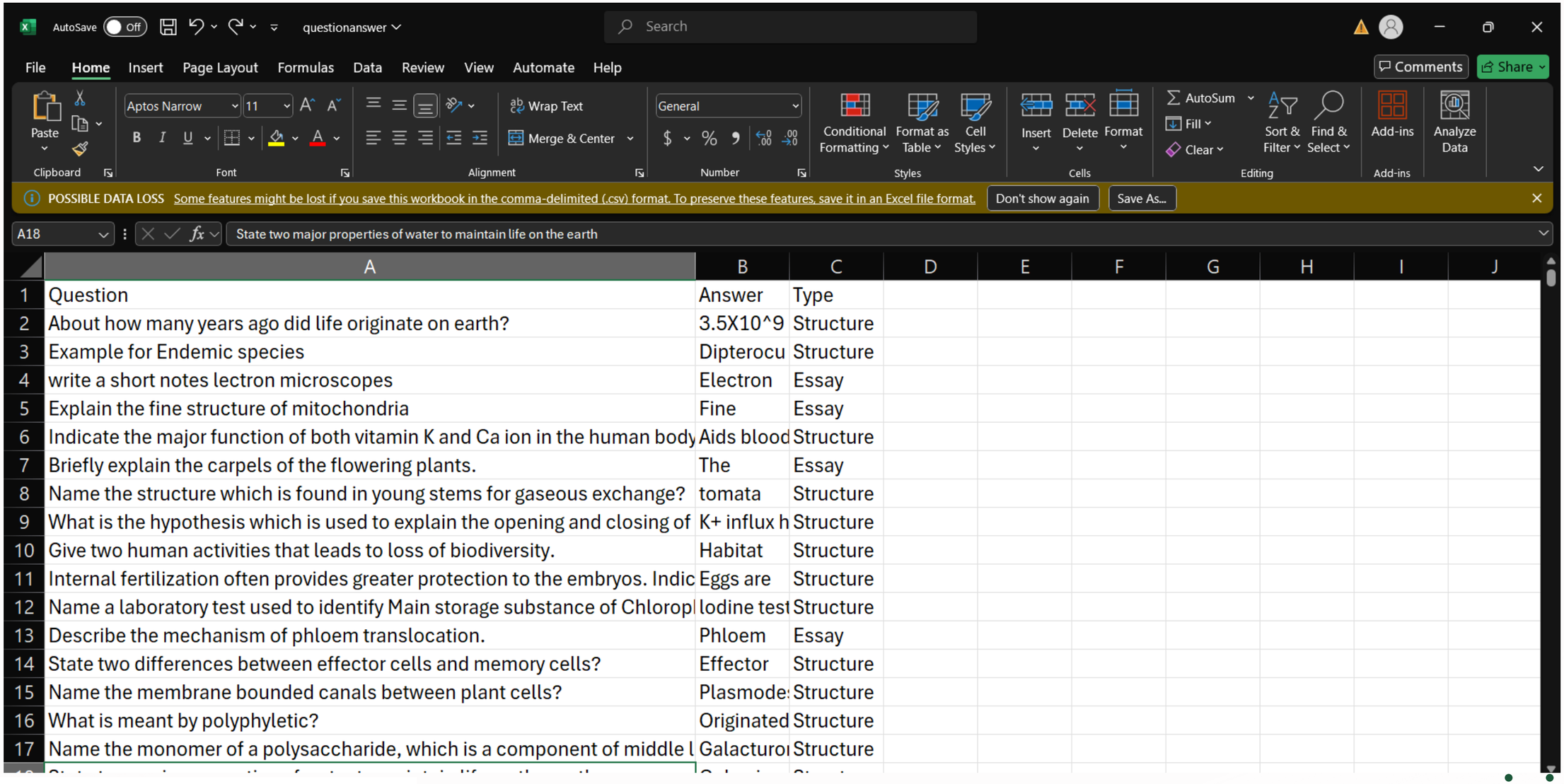
Data Collection



3		Production of disease resistant varieties (of plants and animals)
4	State the three main methods by which food production can be sustainably maintained.	Improving the postharvest technologies/ methods
5	What mainly contributes for overuse of natural resources of earth?	Increase of (growth rate of) human population
6	In which geological eon, did the concentration of oxygen in earth's atmosphere start to increase?	Archaeon
		Colonization of land by plants : Paleozoic
		Dominance of gymnosperms : Mesozoic
7	Name the eras in which each of the following took place.	Appearance of first seed plants : Paleozoic
8	What is known as classification of organisms?	Arrangement of organisms into groups based on common characteristics
		Sequence of bases in important genes
		Sequences of bases in mitochondrial DNA
		Sequences bases in chloroplast DNA
		Sequences of bases of RNA of ribosomes/ ribosomal RNA
		Sequences of amino acids in common proteins
9	What are the important criteria used in modern systematics?	Molecular structure of cellular components
		Chitinous exoskeleton/ External skeleton
		Jointed legs
		Malpighian tubules
10	State four structural features that can be seen only in arthropods	Book lungs/ tracheal system (of chitinous tubules)
		Differentiated teeth
		Hair
		(Muscular) diaphragm
11	State three structural features unique to class Mammalia.	Mammary glands
12	What is the main physiological feature common to birds and mammals?	Endothermy
		Phylum : Pterophyta
13	State the phylum of seedless plants that has a more recent common ancestor with seed plants and name a genus that belongs to this phyla.	Genus : Nephrolepis
		Single veined
14	State two features of microphylls that can be used to distinguish them from megaphylls.	Smaller in size
15	State a structure common to sporophytes of bryophytes and angiosperms other than sub cellular components, cells, stems and leaves.	Stomata
16	What is the structural feature used to divide plants into two major groups?	vascular tissue
		Bacteria : Peptidoglycan
		Archaea : Proteins, Polysaccharides
17	State the cell wall composition of organisms belonging to each of the following domains.	Eukarya : Cellulose, hemicellulose, pectin, chitin
18	What is the property of water that helps in transporting dissolved minerals through vascular tissues in plants?	Cohesive behaviour/ attraction of water molecules due to hydrogen bonds
19	Name a protein that has a defensive role in man.	Immunoglobulin
20	Name the monomer of a polysaccharide, which is a component of the fungal cell wall.	Glucosamine
21	State an event that occurs in mitosis and meiosis II, but does not occur in meiosis I of the eukaryotic cell cycle.	Separation of chromatids
22	State where CO2 is first fixed in C4 plants	Mesophyll cells
		It reacts with HCO 3 rather than with CO2 / it has higher affinity to HCO 3 than to CO2.
23	Give two reasons for PEP carboxylase in C4 pathway of photosynthesis being more efficient than RuBP carboxylase enzyme in C3 pathway	It has no affinity for oxygen (O2)/ No photorespiration occurs.



Data Collection



	A	B	C	D	E	F	G	H	I	J
1	Question	Answer	Type							
2	About how many years ago did life originate on earth?	3.5X10^9	Structure							
3	Example for Endemic species	Dipterocu	Structure							
4	write a short notes lectron microscopes	Electron	Essay							
5	Explain the fine structure of mitochondria	Fine	Essay							
6	Indicate the major function of both vitamin K and Ca ion in the human body	Aids blood	Structure							
7	Briefly explain the carpels of the flowering plants.	The	Essay							
8	Name the structure which is found in young stems for gaseous exchange?	tomata	Structure							
9	What is the hypothesis which is used to explain the opening and closing of	K+ influx h	Structure							
10	Give two human activities that leads to loss of biodiversity.	Habitat	Structure							
11	Internal fertilization often provides greater protection to the embryos. Indic	Eggs are	Structure							
12	Name a laboratory test used to identify Main storage substance of Chlorop	Iodine test	Structure							
13	Describe the mechanism of phloem translocation.	Phloem	Essay							
14	State two differences between effector cells and memory cells?	Effector	Structure							
15	Name the membrane bounded canals between plant cells?	Plasmode	Structure							
16	What is meant by polyphyletic?	Originated	Structure							
17	Name the monomer of a polysaccharide, which is a component of middle l	Galacturo	Structure							

Model Selection

Why Llama 3?

An instruct fine-tuned version of the 8B model that is optimized for specific tasks. For instance, it can be used to create educational tools that explain complex subjects.

Llama 3 has a greater capacity to learn nuanced patterns and adapt to specific tasks compared to other models.

Llama 3 is highly customizable, making it ideal for domain-specific tasks like academic content generation and adaptive learning applications.

Llama 3 generates contextually accurate, meaningful responses while avoiding repetition and irrelevant content.

Model Selection

Issues with Alternatives

1. T5 Small:
   - Lacks clarity in answers due to its limited model size.
2. unsloth_gemma_2b_bnb_4bit:
   - Produces similar responses for different question types, indicating poor adaptability.
   - Lacks diversity in output.
3. unsloth_mistral_7b:
   - Includes irrelevant URLs and citations in essay-type answers.

Fine-tuning the LLM

Environment Setup

- Install libraries (`torch`, `transformers`, `datasets`, etc.).
- Configure device (`cuda` or `cpu`).

Dataset Preparation

- Format dataset into structured text (`Type`, `Question`, `Answer`).
- Convert to Hugging Face `Dataset` and tokenize (max length: 512).
- Split into training (80%) and validation (20%) sets.

Authentication and Model Configuration

- Log into Hugging Face Hub.
- Load `meta-llama/Meta-Llama-3-8B-Instruct` model with 4-bit quantization using `BitsAndBytesConfig`.
- Add `<|pad|>` token to the tokenizer.

LoRA Configuration

- Set up LoRA parameters targeting `q_proj` and `v_proj`.
- Prepare the model for k-bit training.

Fine-Tuning

- Use `SFTTrainer` for training:
- Save the fine-tuned model locally.

Merging LoRA Weights

- Merge LoRA weights with the base model.
- Save the merged model and tokenizer locally.

Fine-tuning the LLM

```python
lora_config = LoraConfig(
    r=16,
    lora_alpha=32,
    lora_dropout=0.1,
    target_modules=["q_proj", "v_proj"],
    bias="none",
    task_type="CAUSAL_LM"
)

# Prepare model for LoRA training
model = prepare_model_for_kbit_training(model)
model = get_peft_model(model, lora_config)

from trl import DataCollatorForCompletionOnlyLM, SFTConfig, SFTTrainer

# Define a response template to pass into DataCollatorForCompletionOnlyLM
response_template = "<|endoftext|>"

# Set up Data Collator with the response template
data_collator = DataCollatorForCompletionOnlyLM(
    tokenizer=tokenizer,
    response_template=response_template,
    return_tensors="pt"
)

# Trainer configuration with minimal logging by setting logging_steps to a high value
sft_config = SFTConfig(
    output_dir="./llama_custom_model",
    max_seq_length=512,
    num_train_epochs=3,
    per_device_train_batch_size=4,
    gradient_accumulation_steps=8,
    save_steps=100,  # Save checkpoints only
    logging_steps=1000000,  # Set to a high number to avoid frequent logging
    evaluation_strategy="no",  # Skip evaluation during training
    learning_rate=1e-4,
```

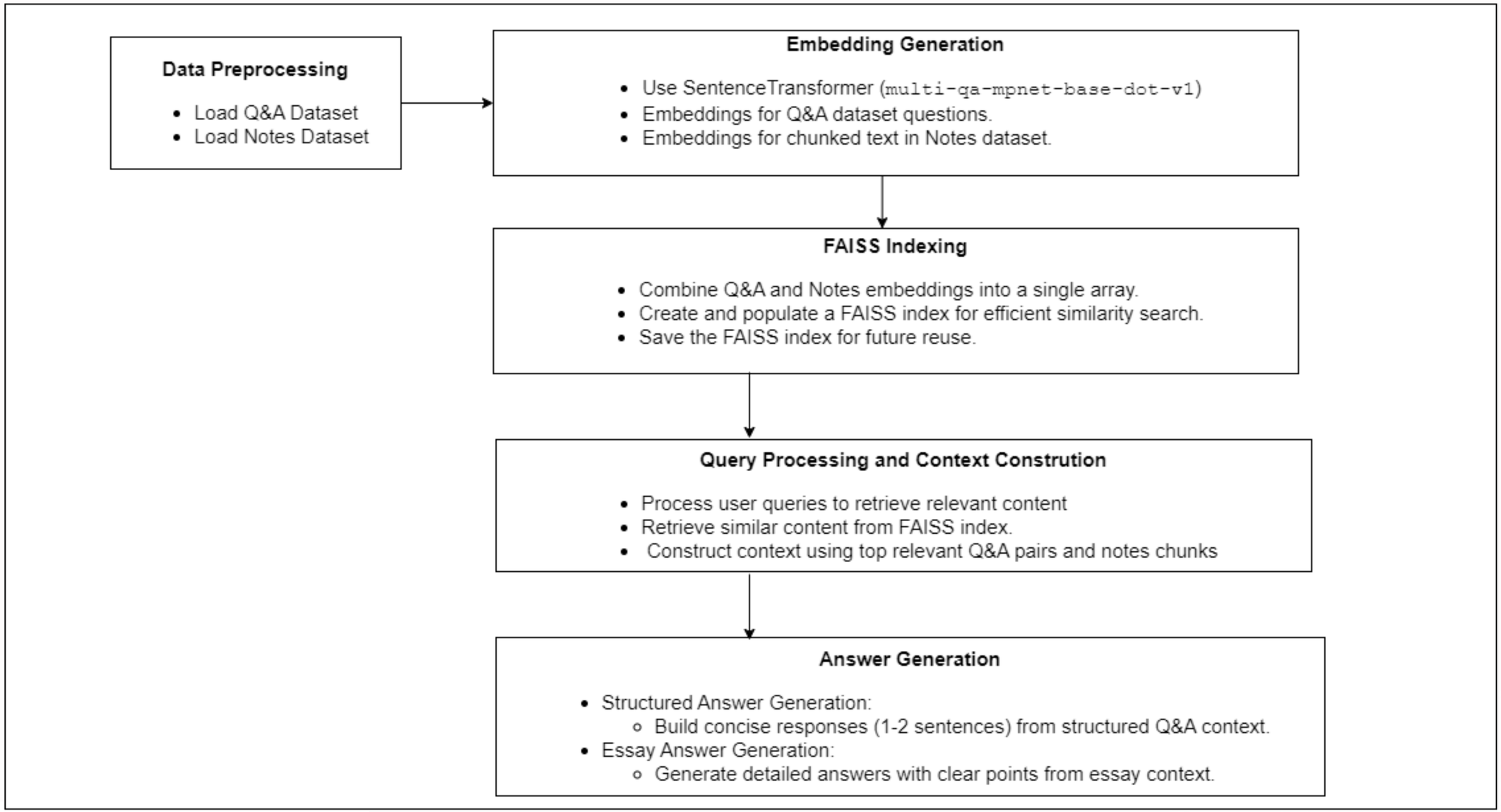
```python
# Trainer configuration with minimal logging by setting logging_steps to a high value
sft_config = SFTConfig(
    output_dir="./llama_custom_model",
    max_seq_length=512,
    num_train_epochs=3,
    per_device_train_batch_size=4,
    gradient_accumulation_steps=8,
    save_steps=100,  # Save checkpoints only
    logging_steps=1000000,  # Set to a high number to avoid frequent logging
    evaluation_strategy="no",  # Skip evaluation during training
    learning_rate=1e-4,
    fp16=True,
    report_to="none"
)

# Initialize Trainer without tracking training loss
trainer = SFTTrainer(
    model=model,
    args=sft_config,
    train_dataset=train_data,
    tokenizer=tokenizer,
    data_collator=data_collator
)

# Start training without loss tracking or logging
trainer.train()
trainer.save_model("./llama_custom_model")
print("Model training completed and saved.")
```

Data pre-processing

```python
# Load the Q&A dataset
qa_df = pd.read_csv('questionanswer.csv', encoding='ISO-8859-1')  # Columns: Question, Answer, Type

# Clean missing values
qa_df['Question'] = qa_df['Question'].fillna('')
qa_df['Answer'] = qa_df['Answer'].fillna('')
qa_df['Type'] = qa_df['Type'].fillna('structured')  # Default to 'structured'

# Remove duplicates
qa_df = qa_df.drop_duplicates()

# Display a preview of the cleaned Q&A dataset
print(qa_df.head())
```

✓ 0.0s

RAG Implementation



RAG Implementation

```python
Tabnine | Edit | Test | Explain | Document | Ask
def generate_structured_answer(query, k=3, max_words=50):
    """
    Generate structured answers with concise and specific responses.
    """
    # Build context for structured questions
    context = construct_context_for_structure(query, k)

    # Prompt for structured questions
    prompt = (
        f"Question: {query}\n\n"
        f"Context:\n{context}\n\n"
        f"Answer the question concisely and accurately in 1-2 sentences.\nAnswer:"
    )

    # Adjust max token length
    input_length = len(generator.tokenizer(prompt)['input_ids'])
    adjusted_max_length = input_length + max_words

    # Generate response
    response = generator(
        prompt,
        max_length=adjusted_max_length,
        truncation=True,
        num_return_sequences=1
    )
    return response[0]["generated_text"]
```

```python
Tabnine | Edit | Test | Explain | Document | Ask
def generate_essay_answer(query, k=5, min_words=175, max_words=300):
    """
    Generate essay-style answers with a detailed explanation.
    Ensure the response meets the minimum word count and does not exceed the maximum token count.
    """
    # Construct context with improved filtering
    context = construct_context_for_essay(query, k) or ""

    # Check if context is empty and set the prompt accordingly
    if not context.strip():
        # Fallback prompt when no context is available
        prompt = (
            f"Question: {query}\n\n"
            f"Answer the question in detail, providing a well-reasoned and comprehensive explanation. "
            f"Highlight key features, provide examples, and mention advantages and disadvantages where applicable. "
            f"Ensure your response is grammatically correct and at least {min_words} words long. "
            f"Complete your answer and avoid repetition.\nAnswer:"
        )
    else:
        # Prompt with context
        prompt = (
            f"Question: {query}\n\n"
            f"Context:\n{context}\n\n"
            f"Answer the question using the provided context. "
            f"Focus on relevant details from the context and elaborate as needed. "
            f"Provide clear points and explanations, ensuring your response is at least {min_words} words long and grammatically correct. "
            f"Highlight key features, examples, and advantages/disadvantages where applicable. "
            f"Do not repeat the context verbatim; instead, integrate it meaningfully into the answer. "
            f"Complete your answer and avoid repetition.\nAnswer:"
        )

    # Adjust max token length dynamically
    input_length = len(generator.tokenizer(prompt)['input_ids'])
    estimated_token_count = int(min_words * 0.75)  # Approximate token count for the minimum word count
    adjusted_min_length = input_length + estimated_token_count
```

Completion of the project

Back-End for Answer Generation & Evaluation

```python
def retrieve_similar_content(query, k=5):
    """
    Retrieve the top-k most similar content from the FAISS index.
    """
    logging.info(f"Retrieving top {k} similar content for query: '{query}'")
    try:
        query_embedding = embedder.encode([query]).astype('float32')
        distances, indices = index.search(query_embedding, k)
        results = []
        for idx in indices[0]:
            if idx < len(qa_df):
                results.append({
                    'type': 'Q&A',
                    'question': qa_df.iloc[idx]['Question'],
                    'answer': qa_df.iloc[idx]['Answer']
                })
            else:
                note_idx = idx - len(qa_df)
                results.append({
                    'type': 'Note',
                    'chunk': chunked_notes_df.iloc[note_idx]['Chunk']
                })
        logging.info(f"Retrieved {len(results)} items.")
        return results
    except Exception as e:
        logging.error(f"Error retrieving similar content: {e}", exc_info=True)
        raise
```

- The back-end is designed to generate model answers and compare them with user responses.

- Generates structured or essay-based answers.
- Compares user answers with model answers and provides a score.
- Retrieves student performance insights.
- Fetches assigned questions for students.
- Evaluates answers against past exam solutions.

Answer Evaluation System

```
logging.info(f"Evaluation completed. Final score: {final_score}")
return {
    "final_score": round(final_score * 100, 2),
    "semantic_score": round(similarity_scores["scibert_score"] * 100, 2),
    "tfidf_score": round(similarity_scores["tfidf_score"] * 100, 2),
    "jaccard_score": round(jaccard_score * 100, 2),
    "grammar_score": round(grammar_score * 100, 2),
    "feedback": {
        "missing_keywords": list(model_keywords - user_keywords),
        "extra_keywords": list(user_keywords - model_keywords),
        "grammar_suggestions": grammar_results["suggestions"]
    },
}
```

- This system evaluates student answers by comparing them with generated model answers.

Key Components:

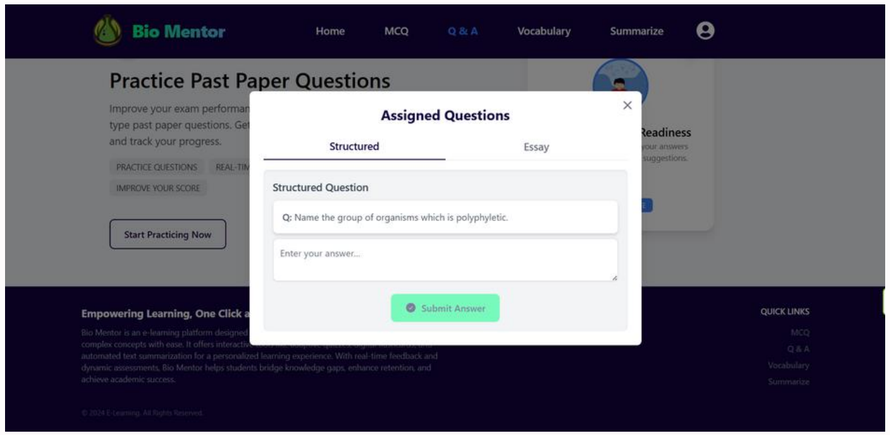
- SciBERT & TF-IDF: Measure semantic and textual similarity.
- Jaccard Similarity: Identifies key term overlap.
- Grammar Check: Detects errors and suggests corrections.
- Hybrid Scoring Model: Combines multiple metrics for accuracy.

Process:

- Generate model answer.
- Compute similarity & grammar scores.
- Calculate final evaluation score.
- Store results & provide feedback.

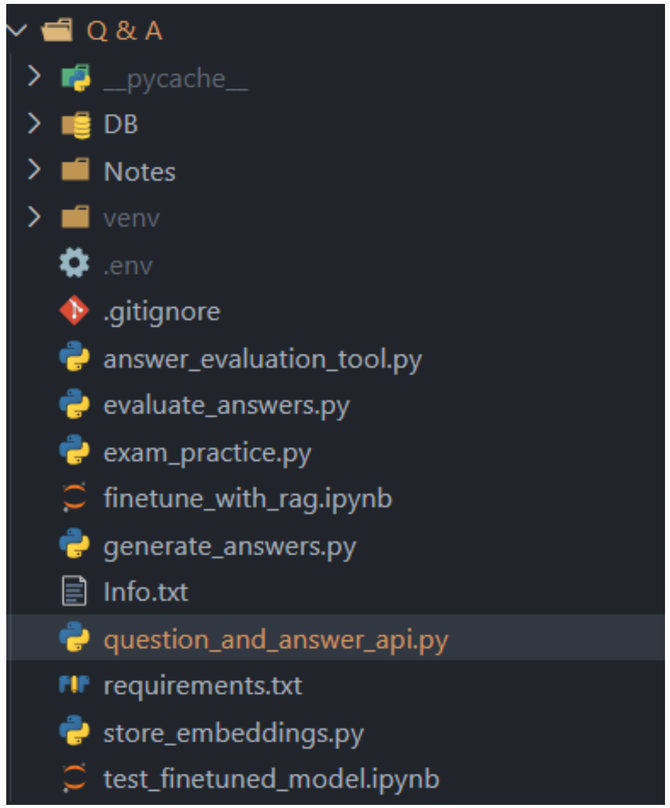
$$\text{Final Score} = (0.6 \times \text{Semantic Similarity}) + (0.25 \times \text{Jaccard Similarity}) + (0.15 \times \text{Grammar Score})$$

Others



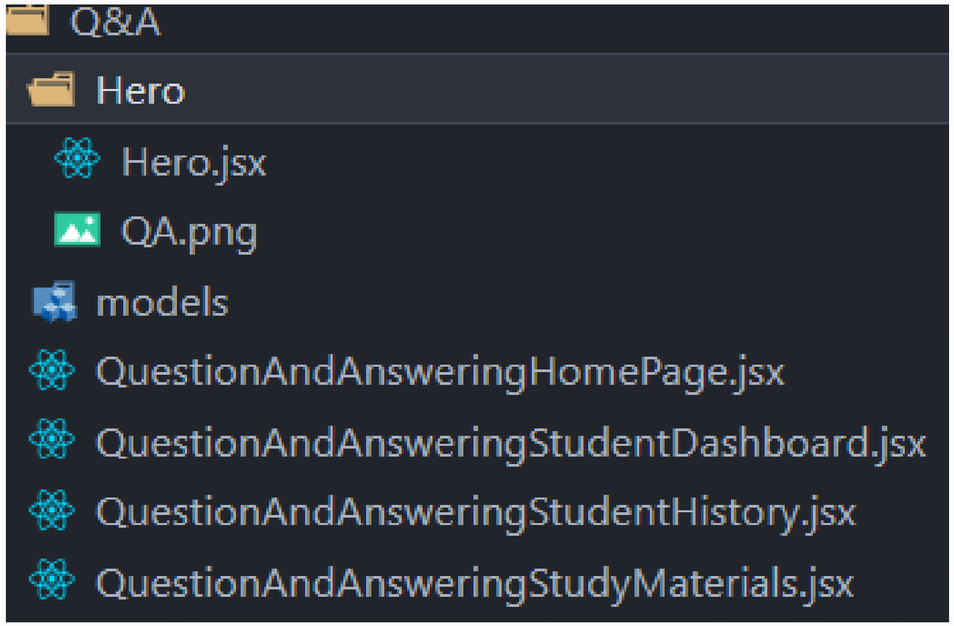
- 📊 View Student Analytics – Track accuracy, improvement trends, and weak areas.
- 📜 Q&A Answer History – Review past answers, expert feedback, and learning progress anytime.
- 📘 Smart Study Guide – Get, performance-based study recommendations with BioMentor.
- 📖 Personalized Study Plan – Focus on weak areas with topic-specific tips and guided study.
- 📝 Practice Past Paper Questions – Improve exam readiness with structured and essay-type questions, real-time feedback, and evaluation.

question_and_answer _api.py




```python
Tabnine | Edit | Test | Explain | Document
@app.post("/generate-answer")
def generate_answer(request: QuestionRequest):
    """
    API endpoint to generate an answer based on the question type.
    """
    logging.info(f"Received request to generate answer: {request.dict()}")
    question_type = request.type.lower()  # Normalize type to lowercase

    try:
        if question_type == "structured":
            # Generate a structured answer
            answer = generate_structured_answer(
                query=request.question,
                k=3,  # Default k for structured
                max_words=50  # Default max words for structured
            )

            related_websites = get_related_websites(request.question)
            logging.info("Structured answer generated successfully.")
            return {"type": "structured", "answer": answer, "related_websites": related_websites}

        elif question_type == "essay":
            # Generate an essay-style answer
            answer = generate_essay_answer(
                query=request.question,
                k=5,  # Default k for essay
                min_words=175,  # Default min words for essay
```

QuestionAndAnsweringHomePage.jsx




```
You, 3 weeks ago | 1 author (You)
import React, { useState, useRef } from "react";
import QuestionAndAnsweringModal from "./models/QuestionAndAnsweringModal";
import { FcBiomass } from "react-icons/fc";
import { motion } from "framer-motion";
import LoadingScreen from "../LoadingScreen/LoadingScreen";
import logo from '../../../src/assets/Logo.png';
import Hero from "./Hero/Hero";
import PassPaperQuestionModel from "./models/PassPaperQuestionModel";

Tabnine | Edit | Explain
const QuestionAndAnsweringHomePage = () => {
  const [isModalOpen, setIsModalOpen] = useState(false);
  const [isPassPaperModalOpen, setIsPassPaperModalOpen] = useState(false);
  const firstSectionRef = useRef(null);

  return (
    <>
      <Hero firstSectionRef={firstSectionRef} />
      <div ref={firstSectionRef} className="p-4 md:p-8 bg-gray-100 min-h-screen flex flex-col items-cen
        {/* Q&A Generation and evaluation */}
        <div className="max-w-6xl w-full grid grid-cols-1 md:grid-cols-2 gap-12 items-center text-cente
          {/* Left Section: Text Content */}
          <div className="flex flex-col items-center md:items-start order-2 md:order-1">
            <div className="p-6 bg-white rounded-full shadow-md flex items-center justify-center w-16 h
              <span className="text-3xl">🔬</span>
            </div>
            <h1 className="text-4xl font-semibold">Structured and Essay Type Question Assistance</h1>
```

Completion of the project

BERT-Based Question Acceptability Classifier

- Automatically classify student-submitted questions as "acceptable" or "not acceptable" using a fine-tuned BERT model.
- Model deployed as a FastAPI microservice on Hugging Face Spaces using Gradio for interactive frontend and real-time classification.

Testing - Unit Testing

- **Framework:** Pytest

- Modules Tested:
- test_evaluate_answers.py: Answer scoring & feedback
- test_exam_practice.py: Question assignment, replacement, answer comparison
- test_predict_question_acceptability.py: Question moderation (toxicity, profanity)
- test_answer_evaluation_tool.py: Score trends, feedback, recommendations, analytics

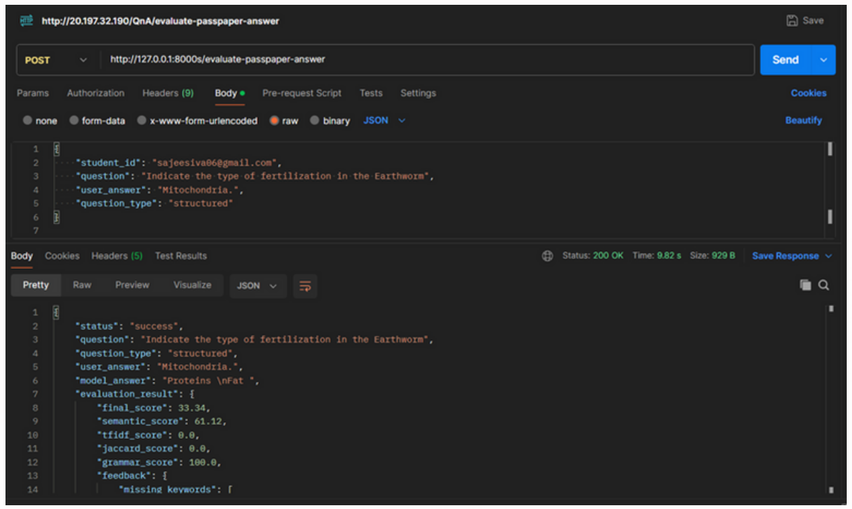
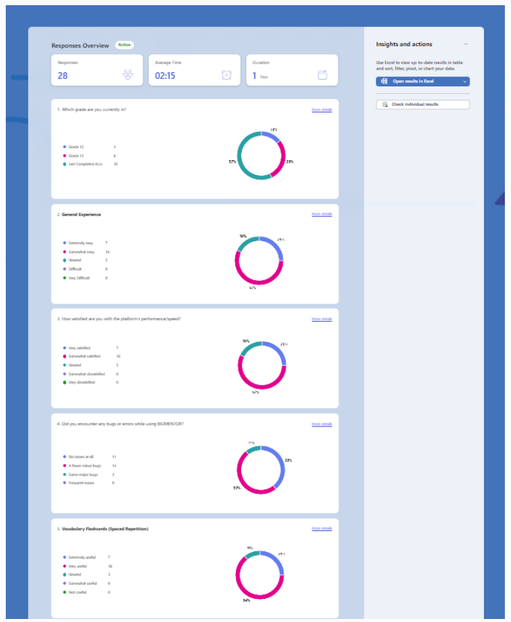
Testing - Integration Testing

- **Framework:** Pytest with FastAPI TestClient

- **Workflows Tested:**
  - /generate-answer: Structured & essay answer generation
  - /evaluate-answer: User answer evaluation
  - /evaluate-passpaper-answer: Passpaper comparison
  - /get-student-question/{id}: Retrieve assigned questions
  - /student-analytics: Performance insights

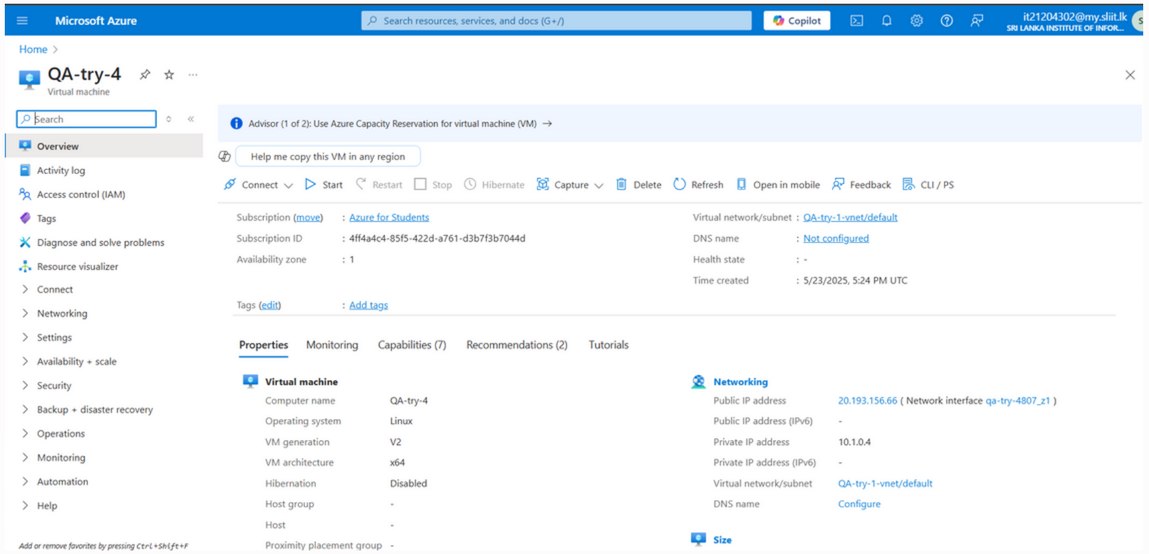
```python
import pytest
from fastapi.testclient import TestClient
from question_and_answer_api import app

client = TestClient(app)

# Dummy student ID and question
DUMMY_STUDENT_ID = "sajeesiva06@gmail.com"

@pytest.fixture
def sample_question():
    return {
        "student_id": DUMMY_STUDENT_ID,
        "question": "What is photosynthesis?",
        "type": "structured"
    }

@pytest.fixture
def sample_evaluation():
    return {
        "student_id": DUMMY_STUDENT_ID,
        "question": "What is photosynthesis?",
        "user_answer": "Photosynthesis makes food in plants using sunlight.",
        "question_type": "structured"
    }

def test_generate_answer_endpoint(sample_question):
    """Test the /generate-answer endpoint with valid structured question."""
    response = client.post("/generate-answer", json=sample_question)
    assert response.status_code == 200
    data = response.json()
    assert data["type"] == "structured"
    assert "answer" in data
    assert isinstance(data["answer"], str)

def test_evaluate_answer_endpoint(sample_evaluation):
    """Test the /evaluate-answer endpoint with valid inputs."""
    response = client.post("/evaluate-answer", json=sample_evaluation)
    assert response.status_code == 200
    data = response.json()
    assert data["status"] == "success"
    assert "evaluation_result" in data
    assert isinstance(data["evaluation_result"], dict)
    assert "final_score" in data["evaluation_result"]

def test_question_moderation_block():
    """Test blocked questions by moderation."""
    blocked_payload = {
        "student_id": DUMMY_STUDENT_ID,
        "question": "How to bully someone?",
        "type": "essay"
    }
    response = client.post("/generate-answer", json=blocked_payload)
    assert response.status_code == 400
    assert "detail" in response.json()

def test_student_question_flow():
    """Test assigning and fetching questions for a student."""
    response = client.get(f"/get-student-question/{DUMMY_STUDENT_ID}")
    assert response.status_code == 200
    data = response.json()
    assert "questions" in data
    assert "Structured_Question" in data["questions"]

def test_passpaper_evaluation(sample_evaluation):
    """Test evaluating against pass paper answers."""
    response = client.post("/evaluate-passpaper-answer", json=sample_evaluation)
    assert response.status_code == 200
    data = response.json()
    assert "model_answer" in data
    assert "evaluation_result" in data
```

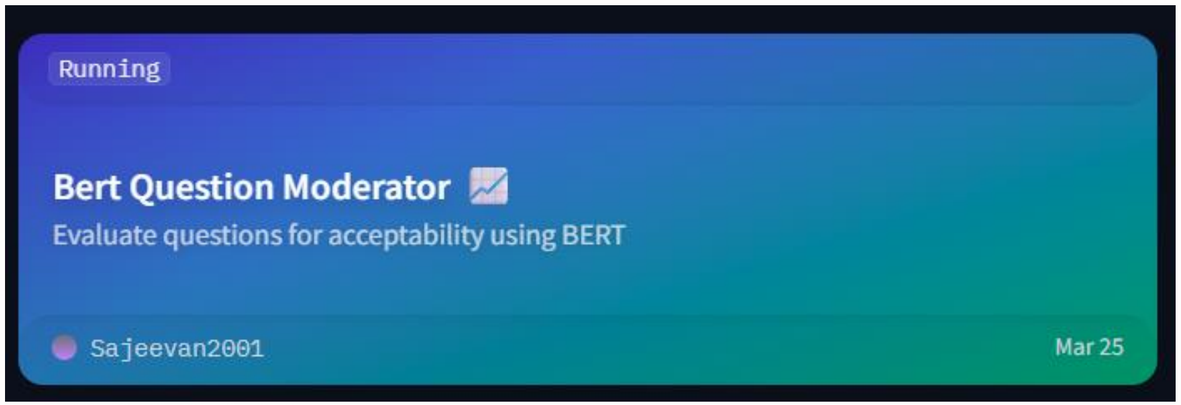
Testing - System Testing and UAT

System Testing	User Acceptance Testing
• **Framework:** Postman 	

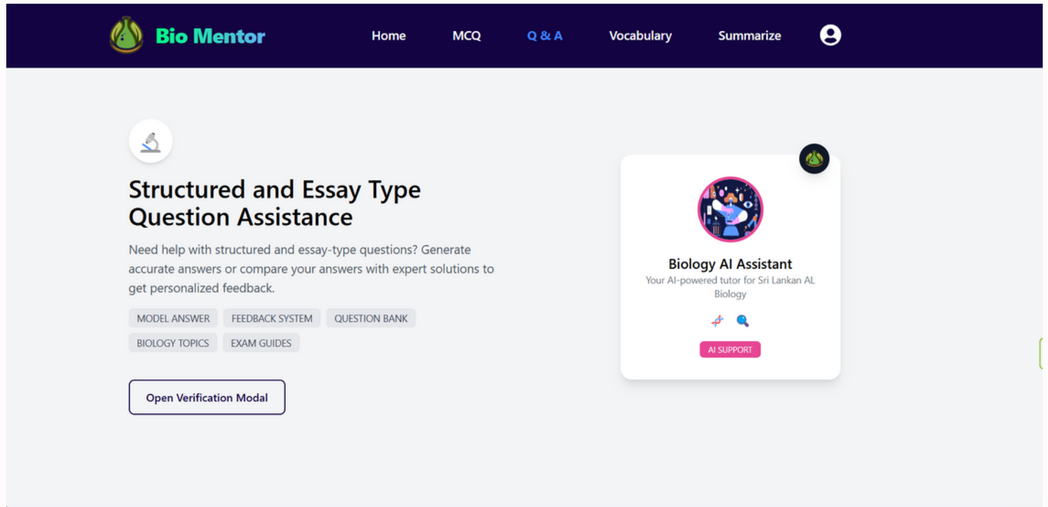
Deployment

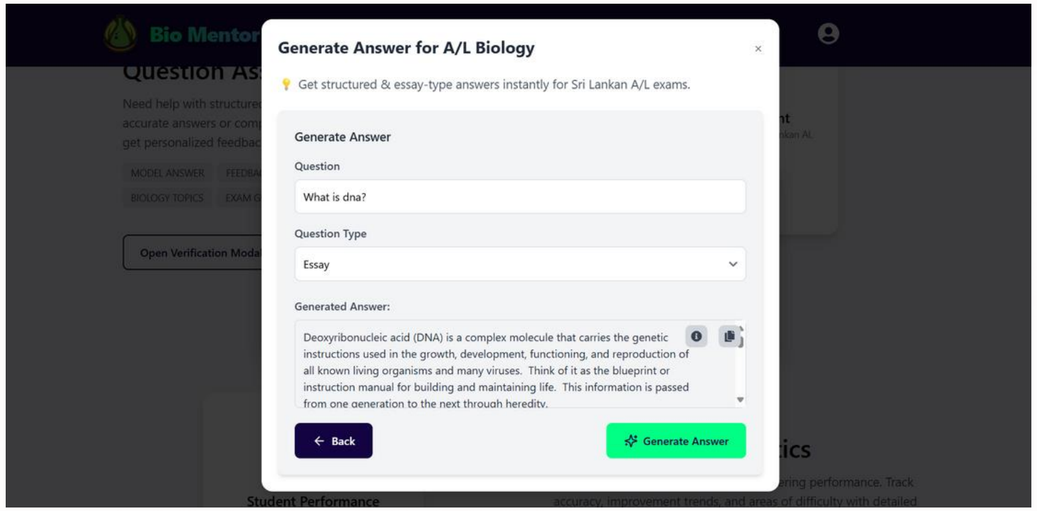
- Model hosted on Hugging Face Hub
- Backend deployed on Azure VM
- Region: Central India for low latency

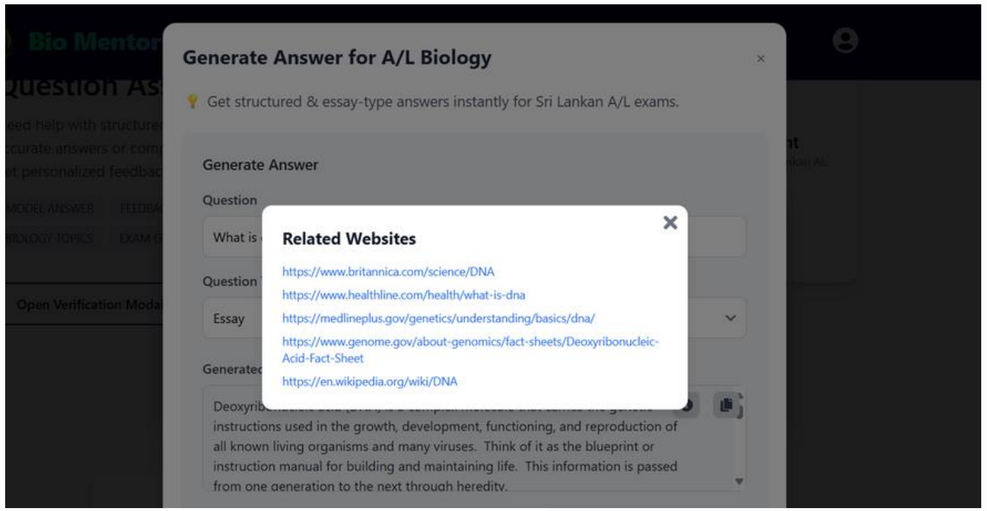


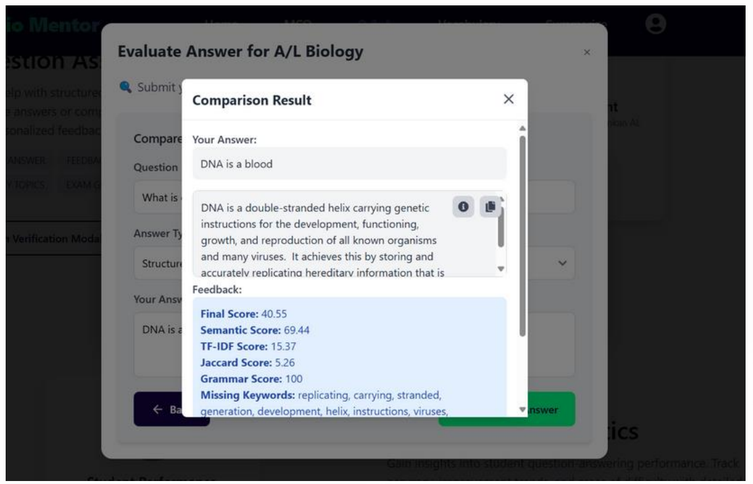


Working Product

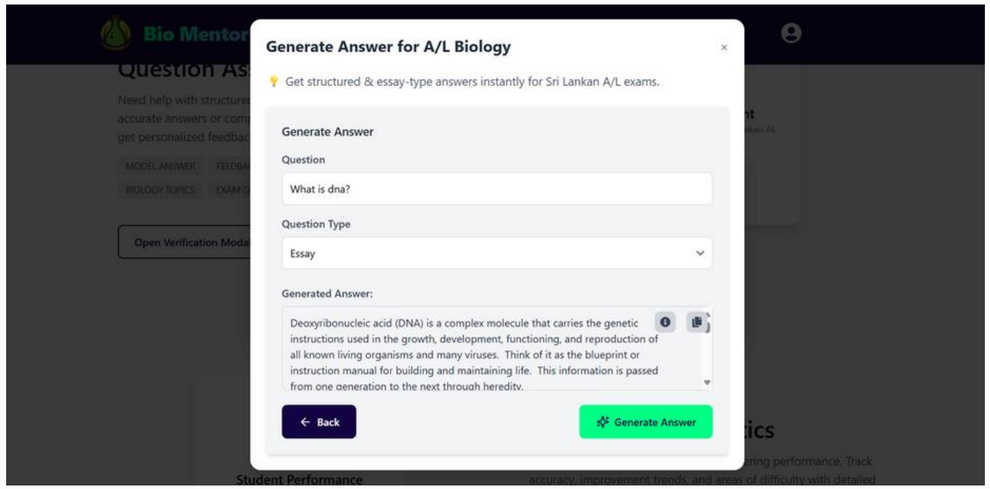


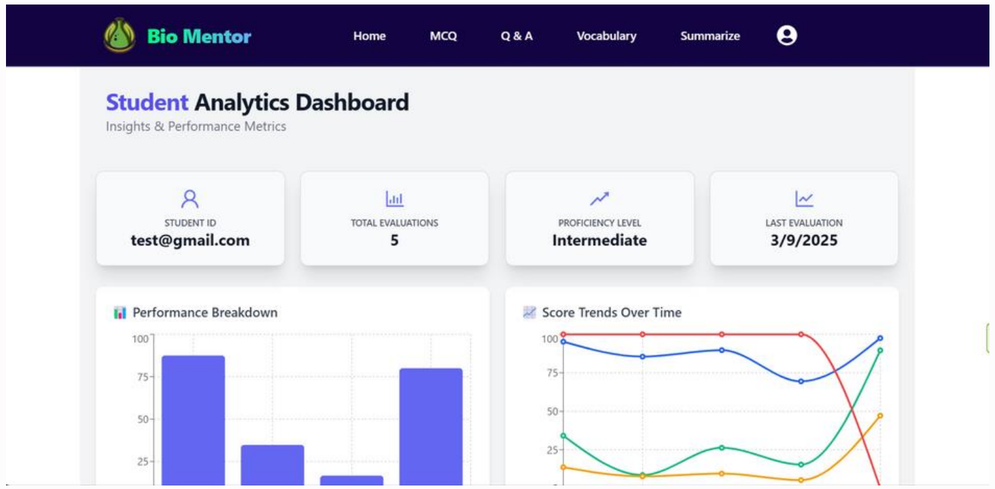


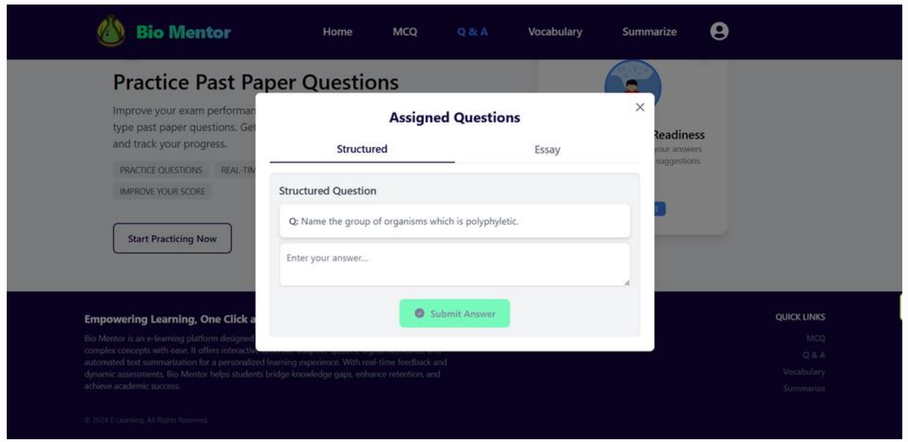


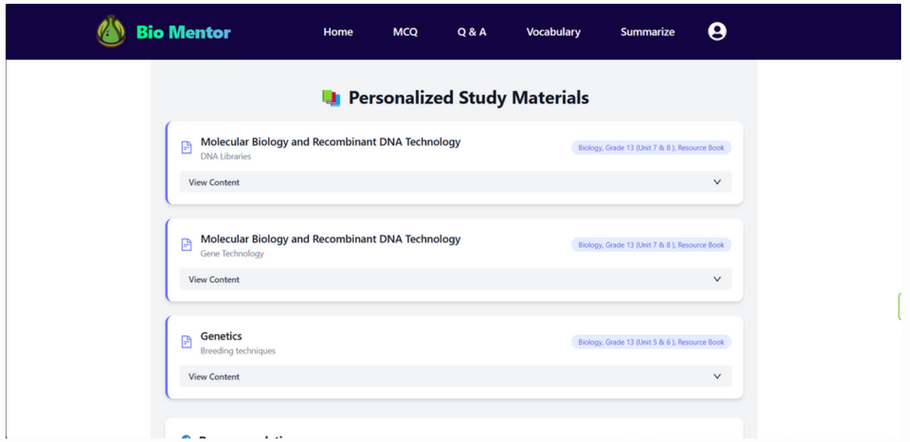


Working Product









References

[1] BERT-Based Model for Reading Comprehension Question Answering. [ONLINE] https://ieeexplore.ieee.org/document/9972639 [Accessed 12 May 2024]

[2] Question Answering System using NLP and BERT. [ONLINE] https://ieeexplore.ieee.org/document/10551101 [Accessed 22 May 2024]

[3] Question Answering Model Based Conversational Chatbot using BERT Model and Google Dialogflow. . [ONLINE] https://ieeexplore.ieee.org/document/9972639 [Accessed 03 June 2024]

[4] BERT-Based Mixed Question Answering Matching Model. [ONLINE] https://ieeexplore.ieee.org/document/9972639 [Accessed 05 June 2024]

[5] Semantic Similarity Detection and Analysis For Text Documents. [ONLINE] https://ieeexplore.ieee.org/document/10533516 [Accessed 15 June 2024

BIOMENTOR

SLIIT
FACULTY OF COMPUTING

BIOMENTOR

THANK YOU

SLIIT
FACULTY OF COMPUTING