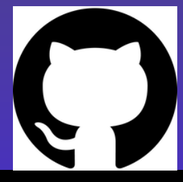
Problem solving and Python programming

ACADEMIC TRACKER

Name - Anshika Maheshwari

Reg no. - 25BCE10595

GitHub repository - https://github.com/anshika25bce10595-tech/Anshika-Maheshwari.git



INTRODUCTION

Managing academic tasks such as assignments, due dates, and grades is a common challenge for students. Without a structured system, they often face missed deadlines, overlooked submissions, and difficulty tracking overall performance. This project presents a console-based Academic Grade & Assignment Tracker, designed to help students organize their academic workload efficiently. Using four standard Python libraries—statistics, pickle, datetime, and os The system enables users to log assignments, calculate averages/GPA, and generate simple performance reports. The goal is to provide a lightweight, practical, and user-friendly tool that supports better planning, improved academic discipline, and stress-free tracking of progress.

PROBLEM STATEMENT

Students often find it difficult to manage their assignments, due dates, and grades in an organized way. Because of this, many end up missing deadlines or losing track of how well they are performing in a subject. Without a simple tool to record tasks and calculate averages, staying on top of studies becomes stressful and confusing. This project aims to build a small, easy-to-use Python program that helps students store their assignments, check their grades, and keep track of important dates so they can manage their academic work more effectively.

FUNCTIONAL REQUIREMENT

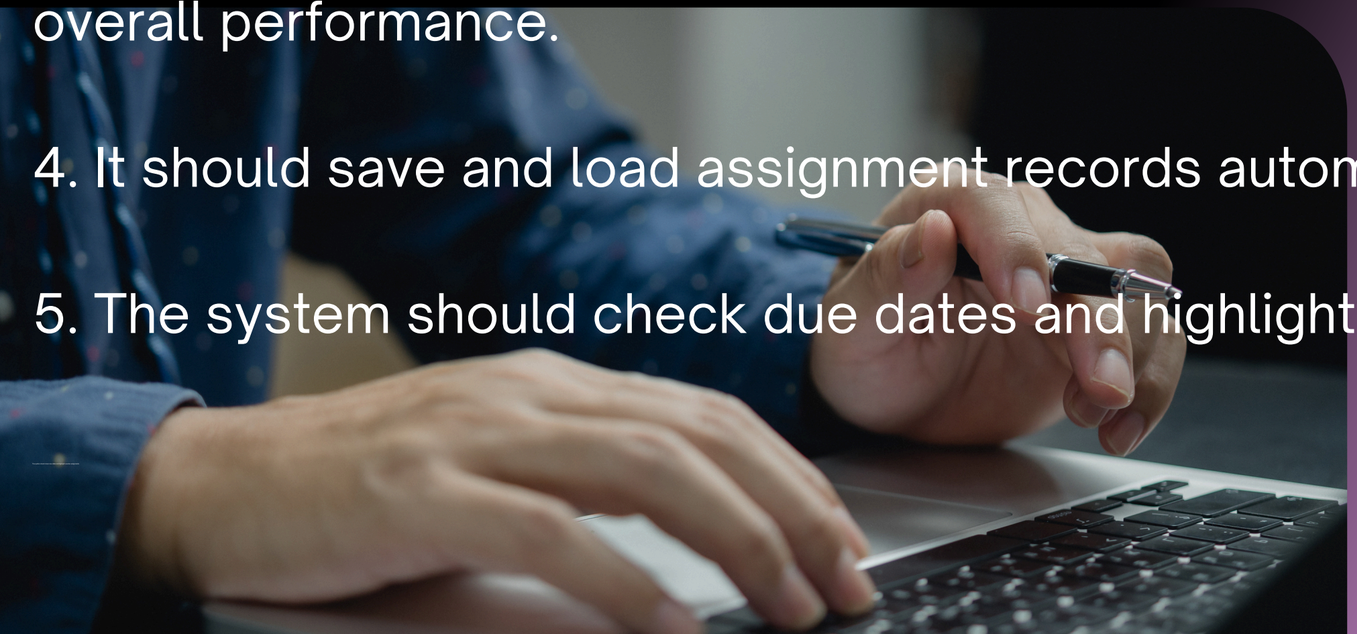
1. The system should allow students to add assignments with name, subject, due date, and grade.

2. It should calculate average grades or GPA using stored data.

3. The program should display reports showing pending tasks, completed assignments, and overall performance.

4. It should save and load assignment records automatically so data isn’t lost between sessions.

5. The system should check due dates and highlight overdue assignments.



NON – FUNCTIONAL REQUIREMENTS

1. The system should be easy to use with a simple, clear console menu.

2. Data should remain saved across sessions for reliability and consistency.

3. The program should run smoothly on any system with Python installed.

4. It should handle incorrect inputs gracefully and avoid crashes.

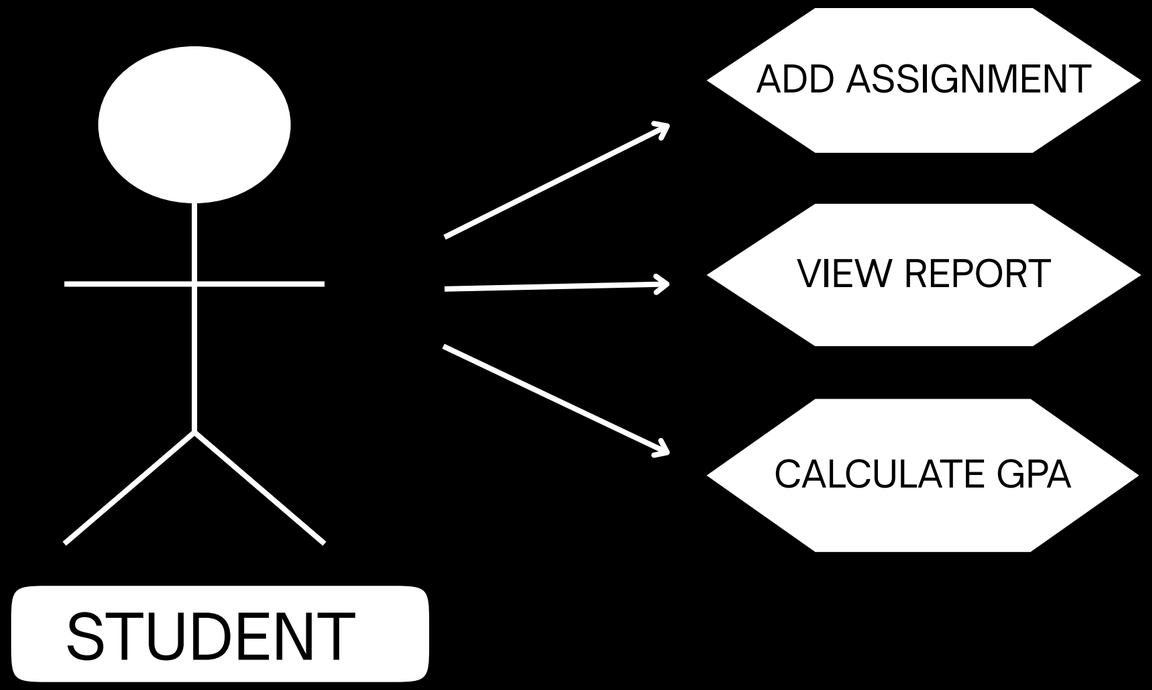
5. The design should be lightweight, fast, and suitable for basic academic management.

SYSTEM ARCHITECTURE

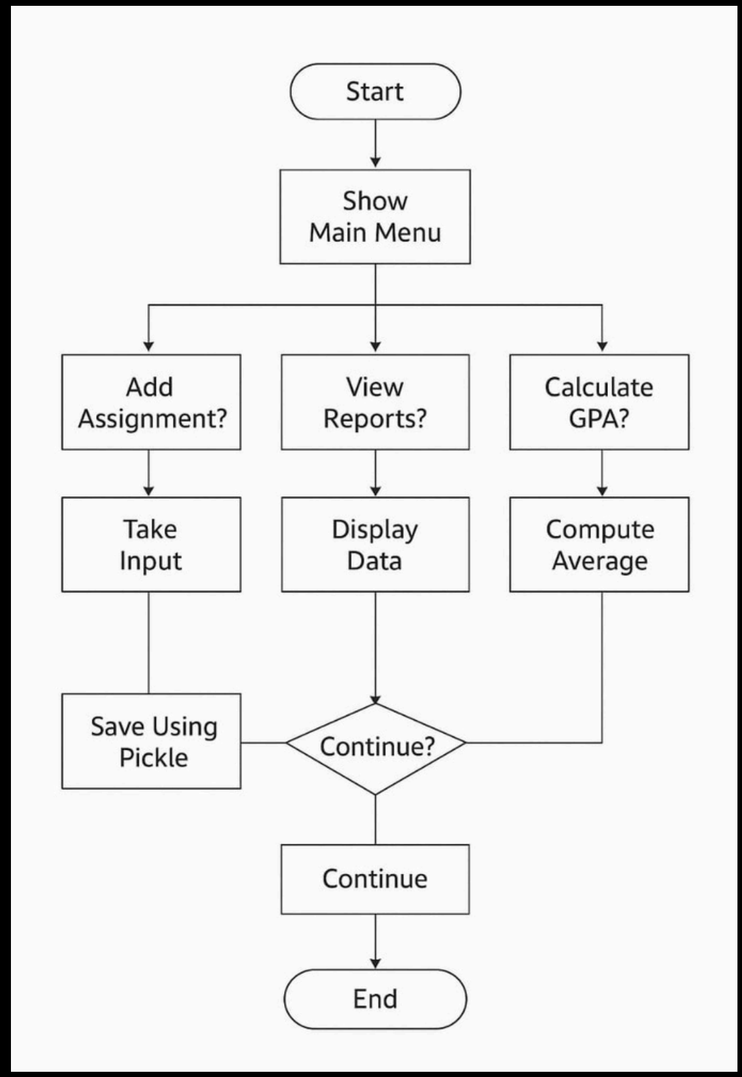
The system is built in a simple, modular way. At the center is a main menu that guides the user through different options like adding assignments, checking grades, or viewing reports. Each of these tasks is handled by its own function to keep the program clean and easy to manage. All assignment information is stored in a file using pickle, so the data stays safe even after closing the program. The statistics library helps calculate averages, and datetime is used to handle and compare due dates. All these parts work together to create a smooth and organized flow for managing student records.

DESIGN DIAGRAM

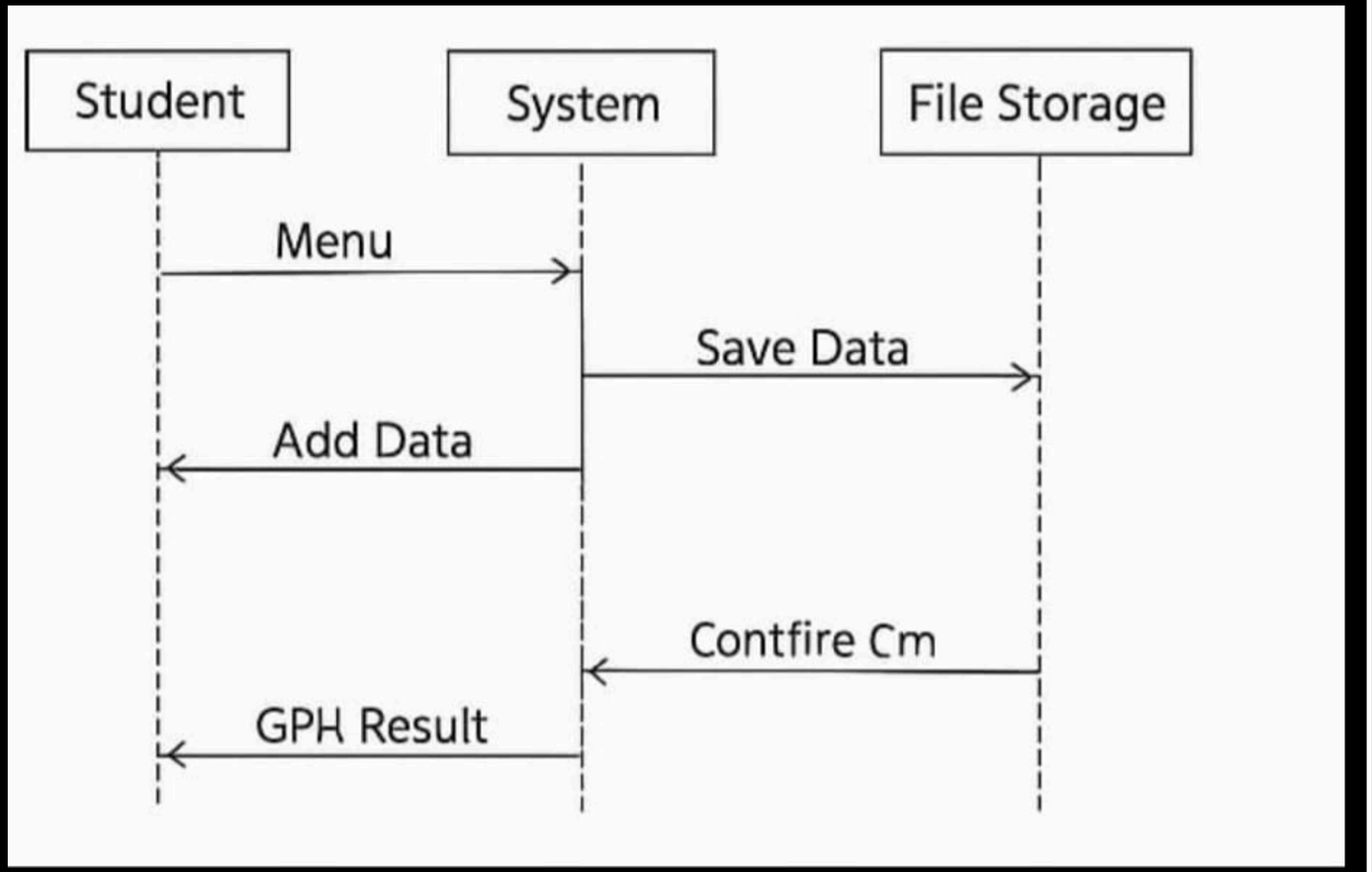
CASE DIAGRAM



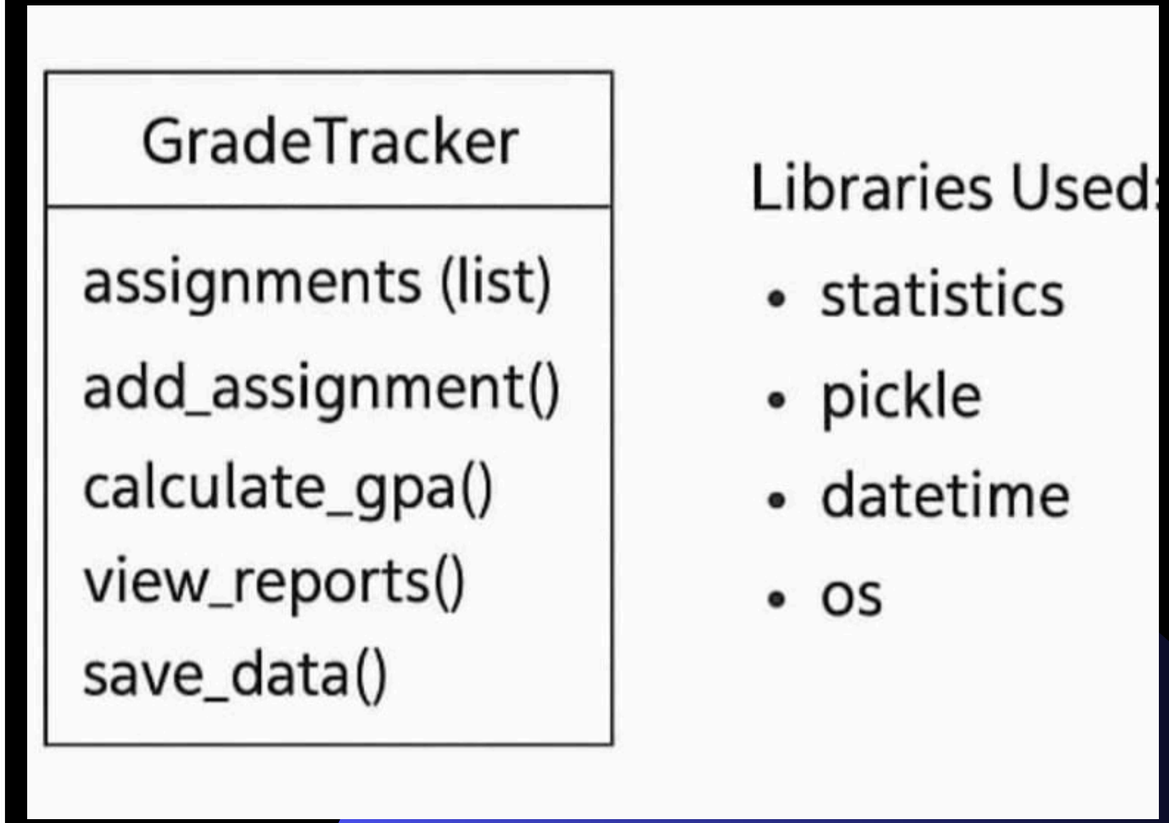
WORKFLOW DIAGRAM



SEQUENCE DIAGRAM



CLASS DIAGRAM



DESIGN DECISIONS AND RATIONALE

1. The project uses a modular design to make maintenance and future updates easier.
2. A console-based interface was chosen for simplicity and portability across systems.
3. Only standard Python libraries (statistics, pickle, datetime, os) are used for educational purposes.
4. Assignment and grade data is saved persistently using the pickle library.
5. A menu-driven interface ensures that the application is user-friendly and easy to navigate.
6. Input data, such as grades and due dates, is validated to prevent errors and inconsistencies.
7. Functions are clearly separated into modules for adding assignments, calculating GPA, and generating reports.
8. The design focuses on providing a practical and educational tool for students to manage their academic performance.

IMPLEMENTATION

Tools Used: Python 3.14, Jupyter notebook, statistics, pickle, datetime, os

Data Structure: List of dictionaries storing assignment info

Functions:

1. add_assignment() → Collects and saves assignment data
2. calculate_gpa() → Computes average grades
3. view_reports() → Displays pending/completed tasks

SCREENSHOTS

```python
import statistics
import pickle
import datetime
import os
g_FILE = 'grades.pkl'
assignments = []
def load_data():
    global assignments
    if os.path.exists(g_FILE):
        try:
            with open(g_FILE, 'rb') as f:
                assignments = pickle.load(f)
        except pickle.PickleError:
            assignments = []
def save_data():
    with open(g_FILE, 'wb') as f:
        pickle.dump(assignments, f)
def add_assignment():
    n = input("Assignment name: ")
    sub = input("Subject: ")
    due_str = input("Due date (YYYY-MM-DD): ")
    try:
        due = datetime.datetime.strptime(due_str, '%Y-%m-%d').date()
        grade = input("Grade (leave blank if not graded): ")
        grade = float(grade) if grade else None
        assignments.append({'name': n, 'subject': sub, 'due': due_str, 'grade': grade})
        save_data()
        print(f"Added '{n}' for {sub}, due {due_str}.")
    except ValueError:
        print("Invalid date or grade!")
```

```python
def calculate_gpa():
    grades = [a['grade'] for a in assignments if a['grade'] is not None]
    if grades:
        gpa = statistics.mean(grades)
        print(f"Overall GPA: {gpa:.2f}")
    else:
        print("No grades recorded yet!!")
def view_reports():
    today = datetime.date.today()
    pending = [a for a in assignments if datetime.datetime.strptime(a['due'], '%Y-%m-%d').date() >= today and a['grade'] is None]
    completed = [a for a in assignments if a['grade'] is not None]
    print("\n--- Pending Assignments ---")
    for a in pending:
        print(f"{a['name']} ({a['subject']}) - Due: {a['due']}")
    print("\n--- Completed Assignments ---")
    for a in completed:
        print(f"{a['name']} ({a['subject']}) - Grade: {a['grade']}")
load_data()
while True:
    print("\n1. Add Assignment\n2. Calculate GPA\n3. View Reports\n4. Exit")
    choice = input("Choose: ")
    if choice == '1':
        add_assignment()
    elif choice == '2':
        calculate_gpa()
    elif choice == '3':
        view_reports()
    elif choice == '4':
        save_data()
        print("Keep it up!")
        break
    else:
        print("Invalid choice!!!")
```

```
1. Add Assignment
2. Calculate GPA
3. View Reports
4. Exit
Choose: 1
Assignment name: CSE Assignment
Subject: CSE
Due date (YYYY-MM-DD): 2025-11-25
Grade (leave blank if not graded):
Added 'CSE Assignment' for CSE, due 2025-11-25.

1. Add Assignment
2. Calculate GPA
3. View Reports
4. Exit
Choose: 2
No grades recorded yet!!

1. Add Assignment
2. Calculate GPA
3. View Reports
4. Exit
Choose: 3

--- Pending Assignments ---
CSE (CSE) - Due: 2025-11-25
CSE Assignment (CSE) - Due: 2025-11-25

--- Completed Assignments ---

1. Add Assignment
2. Calculate GPA
3. View Reports
4. Exit
Choose: 4
Keep it up!
```

OUTPUT

TESTING APPROACH

1.Developed and tested interactively in Jupyter Notebook.

2.Verified adding assignments, calculating GPA, and viewing reports.

3.Checked input validation for grades and dates.

4.Ensured data persistence using pickle.

5.Confirmed accurate display of pending/completed assignments.

CHALLENGES FACED

1.Validating dates and grades to prevent errors.

2.Ensuring data persistence across multiple sessions.

3.Displaying pending and completed assignments clearly.

4.Handling empty assignment lists gracefully.

5.Managing modular code structure for easy updates.

LEARNING & KEY TAKEAWAYS

1.Gained hands-on experience with Python standard libraries (statistics, pickle, datetime, os).

2.Learned to design a modular, menu-driven console application.

3.Understood data persistence and input validation techniques.

4.Improved skills in tracking and managing academic data efficiently.

5.Developed practical problem-solving skills for real-world student challenges.

FUTURE ENHANCEMENTS

1.Develop a GUI version using Tkinter for better usability.

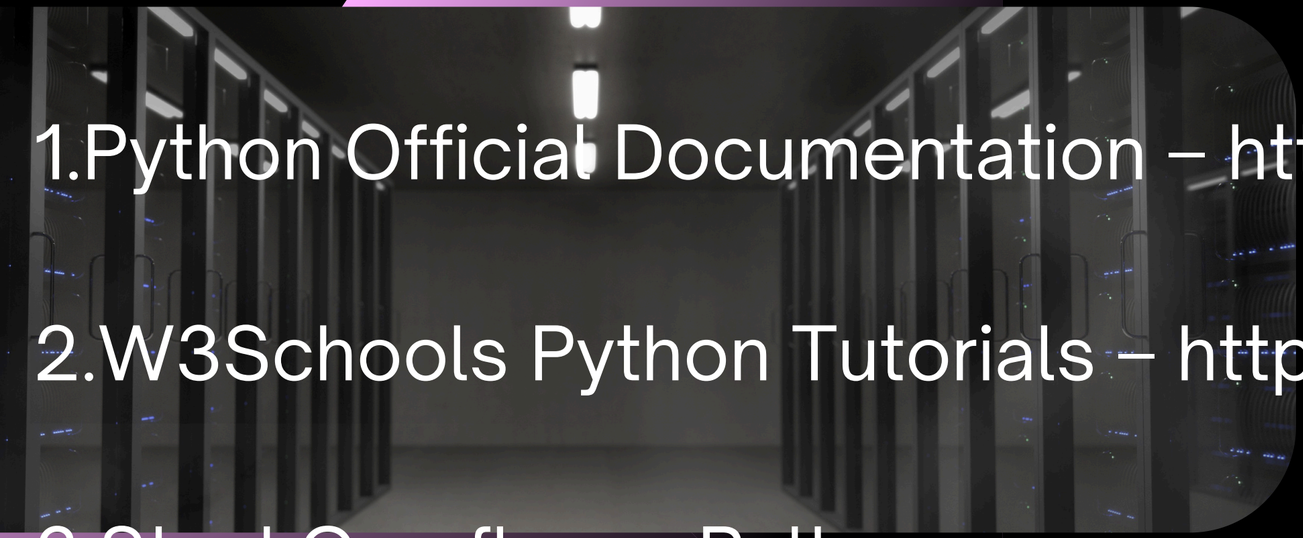
2.Add automatic GPA weighting for different subjects.

3.Implement email or notification alerts for overdue assignments.

4.Enable cloud-based storage for multi-device access.

5.Include edit and delete functionality for assignments.

REFERENCES



1.Python Official Documentation – https://docs.python.org/3/

2.W3Schools Python Tutorials – https://www.w3schools.com/python/

3.StackOverflow – Python programming queries

4.Jupyter Notebook Documentation – https://jupyter.org/

5.GitHub repositories for similar academic projects

THANK YOU!!