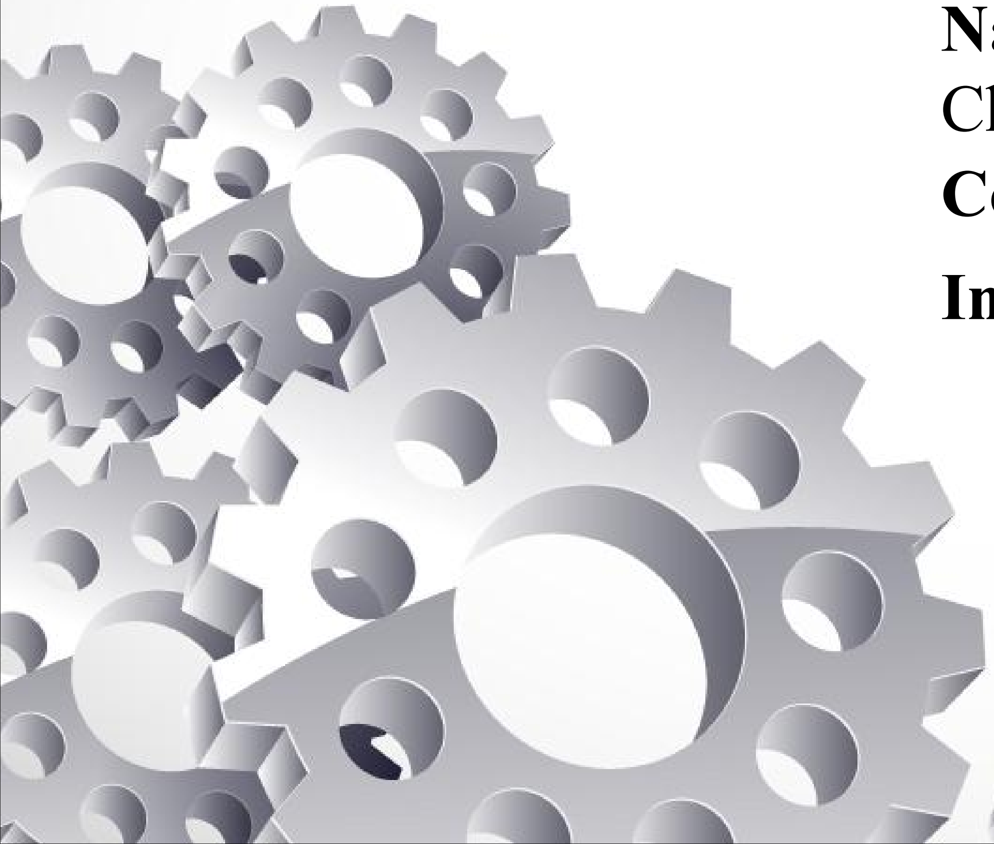
Name : J. Kavya Sri

Course Name : Interships

Name of the instructor: Priyanka Choudhary

Company: Micro IT

Internship Type: Online



CONTENT

Abstract

The Student Record Management System is a simple, command-line-based application developed in Python to efficiently manage student records. The system allows users to add, view, search, update, and delete student information based on unique roll numbers. It also provides functionality to save student data persistently in a JSON file for future access. This project focuses on providing an intuitive, text-based interface to help users manage records without the need for complex database management or graphical interfaces. It is an ideal introductory project for beginners to understand fundamental concepts like file handling, data structures, and basic CRUD (Create, Read, Update, Delete) operations in Python.

Introduction

In educational institutions, maintaining accurate and easily accessible student records is essential. Traditional paper-based record management is time-consuming and prone to errors. Digital solutions provide faster, safer, and more organized methods of handling student data.

The Student Record Management System (CLI-based) is designed to automate basic record-keeping tasks, reducing manual effort and minimizing errors. This system enables users to perform the following key functions: adding new students, viewing all student records, searching for specific students by roll number, updating existing records, deleting records, and saving all data securely in a file for later retrieval.

By using Python’s built-in libraries and straightforward logic, this project serves as an excellent learning tool for beginners to understand how to build a simple yet functional management system using the command line.

Objectives

To develop a simple command-line student management system.

To perform add, view, search, update, and delete operations.

To save and load student records using file storage.

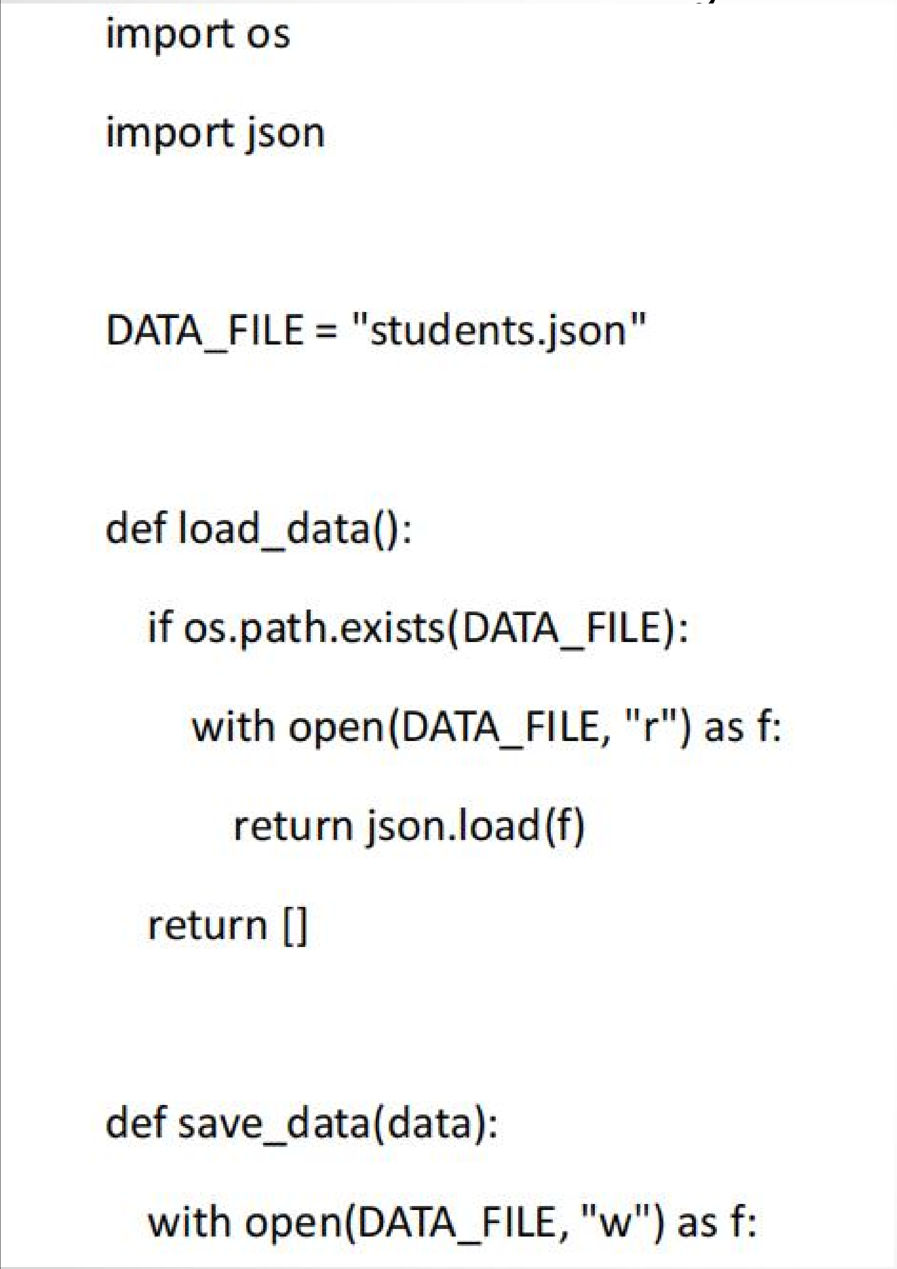
To enable easy search and quick updates by roll number.

To improve Python programming and file handling skills.

Project 1: Student Record Management System

- **Type:** Python CLI Application
- **Purpose:** Manage student records easily
- **CRUD operations:** Add, View, Search, Update, Delete
- **Data storage:** JSON file
- **Description:** A simple command-line app to manage student records.

Student Record System - Python Code



```python
import os
import json


DATA_FILE = "students.json"


def load_data():
    if os.path.exists(DATA_FILE):
        with open(DATA_FILE, "r") as f:
            return json.load(f)
    return []


def save_data(data):
    with open(DATA_FILE, "w") as f:
        json.dump(data, f, indent=4)


def add_student(data):
    roll = input("Enter Roll Number: ")
    name = input("Enter Name: ")
    marks = float(input("Enter Marks: "))
    data.append({"roll": roll, "name": name, "marks": marks})
    print("Student added successfully!")
```

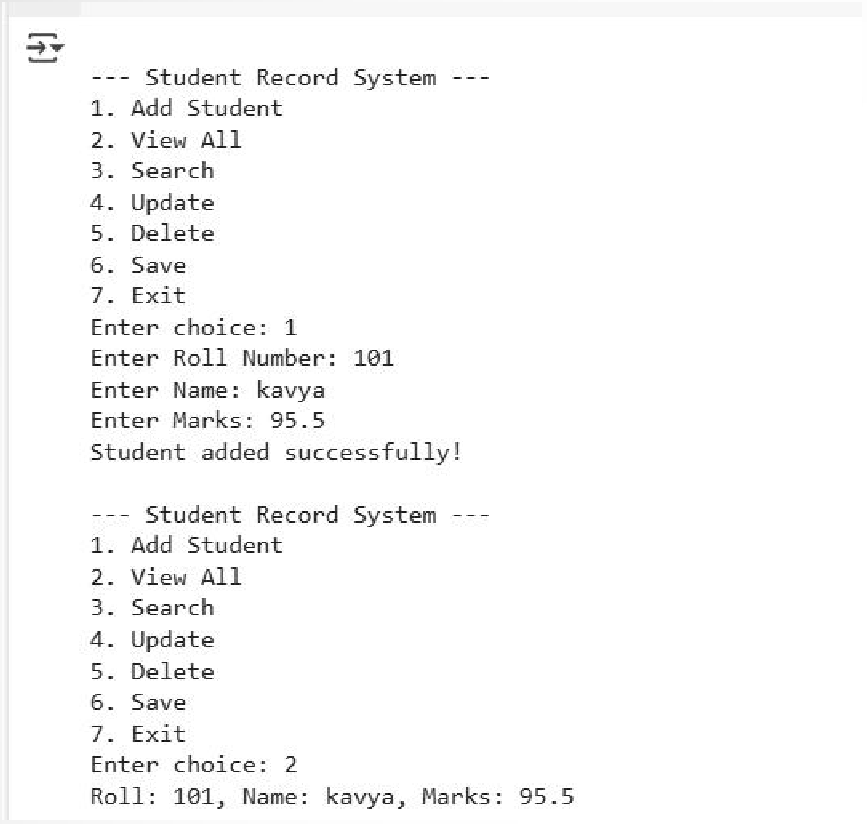
```
def view_students(data):
    if not data:
        print("No records found.")
        return
    for stu in data:
        print(f"Roll: {stu['roll']}, Name: {stu['name']}, Marks: {stu['marks']}")


def search_student(data):
    roll = input("Enter Roll Number to search: ")
    for stu in data:
        if stu['roll'] == roll:
            print(f"Found: {stu}")
            return
    print("Student not found.")
```

```
def update_student(data):
    roll = input("Enter Roll Number to update: ")
    for stu in data:
        if stu['roll'] == roll:

                stu['name'] = input("Enter new name: ")
                stu['marks'] = float(input("Enter new marks: "))
                print("Record updated.")
                return
       print("Student not found.")
```

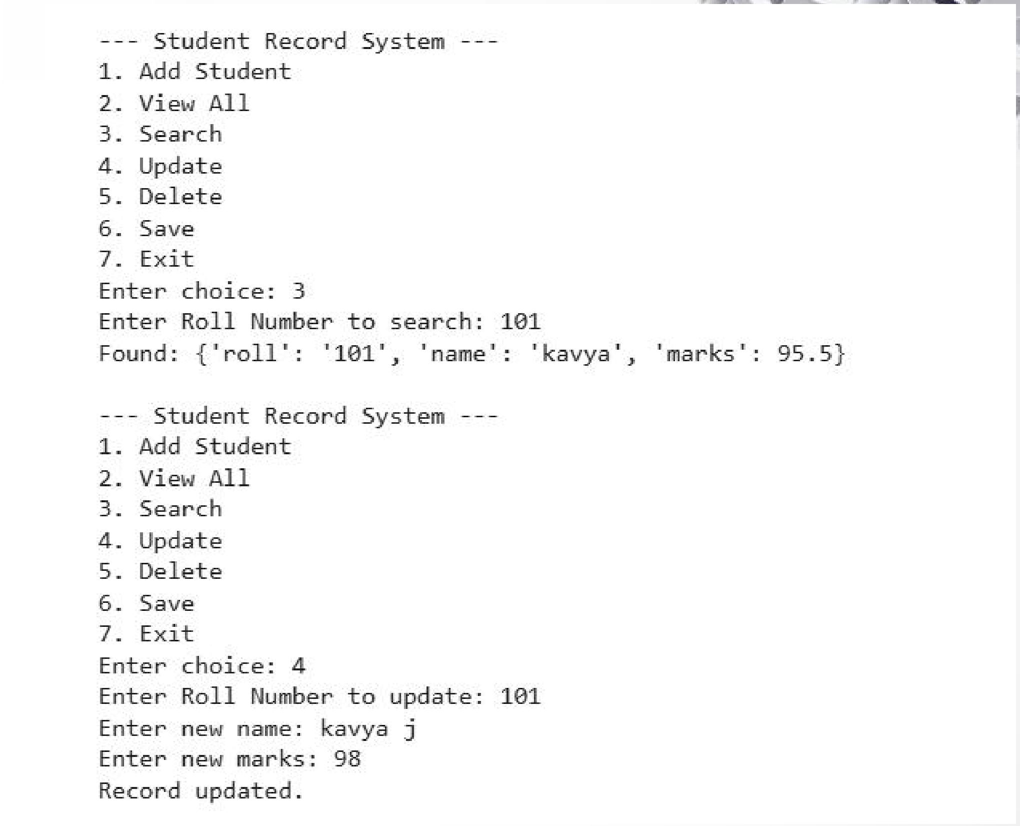
```
def delete_student(data):
    roll = input("Enter Roll Number to delete: ")
    for i, stu in enumerate(data):
        if stu['roll'] == roll:
            del data[i]
            print("Record deleted.")
            return
    print("Student not found.")
```

```
def main():
    data = load_data()
    while True:
        print("\n--- Student Record System ---")
        print("1. Add Student\n2. View All\n3. Search\n4. Update\n5. Delete\n6. Save\n7. Exit")
        choice = input("Enter choice: ")
        if choice == '1':
            add_student(data)
        elif choice == '2':
            view_students(data)
        elif choice == '3':
            search_student(data)
        elif choice == '4':
            update_student(data)
        elif choice == '5':
            delete_student(data)
        elif choice == '6':
            save_data(data)
            print("Data saved.")
        elif choice == '7':
            break
        else:
            print("Invalid choice.")


if __name__ == "__main__":
    main()
```

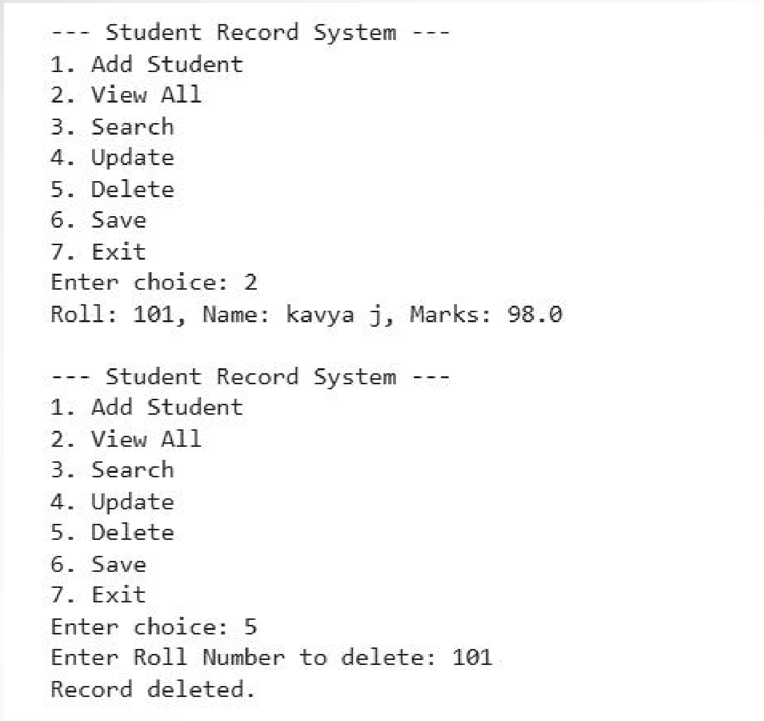
- **OUTPUT**



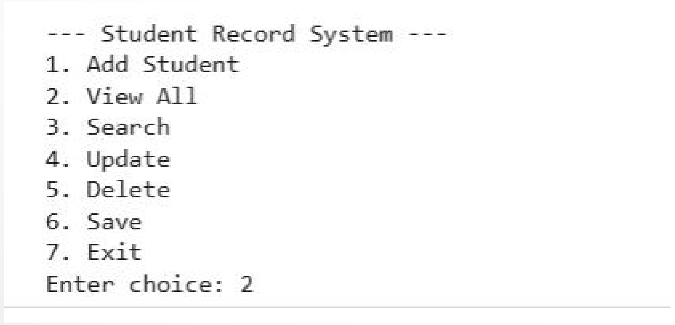
```
--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 1
Enter Roll Number: 101
Enter Name: kavya
Enter Marks: 95.5
Student added successfully!

--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 2
Roll: 101, Name: kavya, Marks: 95.5
```



```
--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 3
Enter Roll Number to search: 101
Found: {'roll': '101', 'name': 'kavya', 'marks': 95.5}

--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 4
Enter Roll Number to update: 101
Enter new name: kavya j
Enter new marks: 98
Record updated.
```



```
--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 2
Roll: 101, Name: kavya j, Marks: 98.0

--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 5
Enter Roll Number to delete: 101
Record deleted.
```



```
--- Student Record System ---
1. Add Student
2. View All
3. Search
4. Update
5. Delete
6. Save
7. Exit
Enter choice: 2
```

Advantages

- Very beginner-friendly.
- Easy to understand Python basics like lists, dictionaries, file operations, and functions.
- Can be expanded with additional features later (like sorting, validation, etc.).

Applications

Small Schools / Institutions

For managing small sets of student records quickly without needing complex software.

Personal Academic Tracking

Students or teachers can track individual performance records.

Python Learning Projects

Excellent for beginners to understand Python’s file handling, lists, and dictionaries.

Foundations for Advanced Systems

Can be expanded to GUI applications or full databases (like SQLite).

Challenges Faced:

Data Validation:

- Ensuring correct input types (e.g., numbers for marks).

Unique Roll Numbers:

- Preventing duplicate roll numbers required additional checking.

Error Handling:

- Managing cases when files do not exist or input is invalid.

User Experience:

- Making the interface clear and easy to use in a text-only environment.

Summary

- A simple CLI-based application to manage student records.
- Supports adding, viewing, searching, updating, deleting, and saving student data.
- Uses JSON files for data storage to ensure records persist between program runs.
- A practical project to understand Python basics, file handling, and data structures.

THANK YOU!