DAY-6

Create a School Management System (Python OOP)

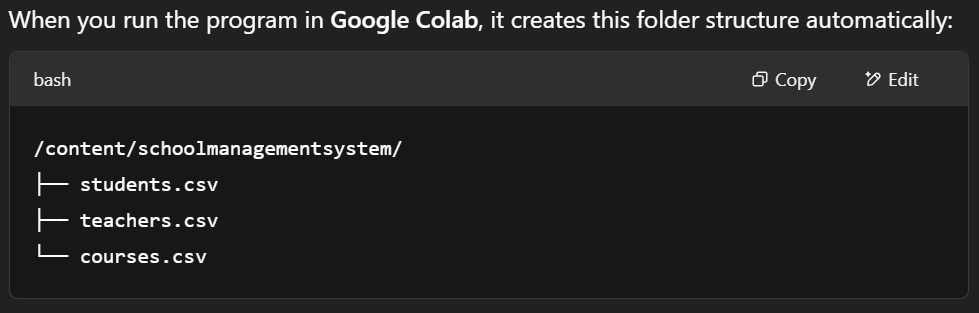
1. Introduction

This project is a School Management System built using Python's Object-Oriented Programming (OOP) concepts. It allows users to add, display, and save information about students, teachers, and courses. Data is stored in CSV files, and the program runs interactively through a text-based menu system on Google Colab.

2. Project Features

* Add new Students, Teachers, and Course
* Display lists of all Students, Teachers, and Course
* Save data into CSV files
* Simple text menu-based interface
* Fully OOP based Python implementation
* Runs directly in Google Colab

3. Folder Structure



4. How to Run the Project (Step by Step)

**Step 1: Open Google Colab**

* Go to https://colab.research.google.com
* Click **New Notebook**

**Step 2: Install and Setup**

* Create a new code cell and paste this:

import os

import pandas as pd

os.makedirs('/content/schoolmanagementsystem', exist\_ok=True)

* Press **Shift+Enter** to run it

**Step 3: Add Class Definitions**

* Create a new code cell
* Copy and paste the **class definitions code** (Person, Student, Teacher, Course, SchoolManagementSystem)

# Base Person class

class Person:

def \_\_init\_\_(self, name, age, gender):

self.name = name

self.age = age

self.gender = gender

# Student class

class Student(Person):

def \_\_init\_\_(self, name, age, gender, student\_id):

super().\_\_init\_\_(name, age, gender)

self.student\_id = student\_id

# Teacher class

class Teacher(Person):

def \_\_init\_\_(self, name, age, gender, teacher\_id):

super().\_\_init\_\_(name, age, gender)

self.teacher\_id = teacher\_id

# Course class

class Course:

def \_\_init\_\_(self, course\_name, course\_code):

self.course\_name = course\_name

self.course\_code = course\_code

# School Management System class

class SchoolManagementSystem:

def \_\_init\_\_(self):

self.students = []

self.teachers = []

self.courses = []

def add\_student(self, student):

self.students.append(student)

def add\_teacher(self, teacher):

self.teachers.append(teacher)

def add\_course(self, course):

self.courses.append(course)

def display\_all\_students(self):

if not self.students:

print("No students found.")

for s in self.students:

print(f"{s.name}, {s.age}, {s.gender}, ID: {s.student\_id}")

def display\_all\_teachers(self):

if not self.teachers:

print("No teachers found.")

for t in self.teachers:

print(f"{t.name}, {t.age}, {t.gender}, ID: {t.teacher\_id}")

def display\_all\_courses(self):

if not self.courses:

print("No courses found.")

for c in self.courses:

print(f"{c.course\_name}, Code: {c.course\_code}")

def save\_students\_to\_file(self):

data = [{'Name': s.name, 'Age': s.age, 'Gender': s.gender, 'Student ID': s.student\_id} for s in self.students]

df = pd.DataFrame(data)

df.to\_csv('/content/schoolmanagementsystem/students.csv', index=False)

def save\_teachers\_to\_file(self):

data = [{'Name': t.name, 'Age': t.age, 'Gender': t.gender, 'Teacher ID': t.teacher\_id} for t in self.teachers]

df = pd.DataFrame(data)

df.to\_csv('/content/schoolmanagementsystem/teachers.csv', index=False)

def save\_courses\_to\_file(self):

data = [{'Course Name': c.course\_name, 'Course Code': c.course\_code} for c in self.courses]

df = pd.DataFrame(data)

df.to\_csv('/content/schoolmanagementsystem/courses.csv', index=False)

* Run the cell

**Step 4: Add Data Entry Functions**

* Create another new code cell
* Paste the **add\_student\_manually, add\_teacher\_manually, add\_course\_manually** functions

def add\_course\_manually(sms):

course\_name = input("Enter course name: ")

course\_code = input("Enter course code: ")

course = Course(course\_name, course\_code)

sms.add\_course(course)

sms.save\_courses\_to\_file()

print(f"Course '{course\_name}' added successfully!")

def add\_teacher\_manually(sms):

name = input("Enter teacher's name: ")

age = int(input("Enter teacher's age: "))

gender = input("Enter teacher's gender: ")

teacher\_id = input("Enter teacher ID: ")

teacher = Teacher(name, age, gender, teacher\_id)

sms.add\_teacher(teacher)

sms.save\_teachers\_to\_file()

print(f"Teacher '{name}' added successfully!")

def add\_student\_manually(sms):

name = input("Enter student's name: ")

age = int(input("Enter student's age: "))

gender = input("Enter student's gender: ")

student\_id = input("Enter student ID: ")

student = Student(name, age, gender, student\_id)

sms.add\_student(student)

sms.save\_students\_to\_file()

print(f"Student '{name}' added successfully!")

* Run the cell

**Step 5: Add the Interactive Menu**

* Create one final code cell
* Paste the **menu code**

# System ka object banao

sms = SchoolManagementSystem()

# Menu loop

while True:

print("\n--- School Management System ---")

print("1. Display Students")

print("2. Display Teachers")

print("3. Display Courses")

print("4. Add Student")

print("5. Add Teacher")

print("6. Add Course")

print("7. Exit")

choice = input("Enter choice: ")

if choice == '1':

sms.display\_all\_students()

elif choice == '2':

sms.display\_all\_teachers()

elif choice == '3':

sms.display\_all\_courses()

elif choice == '4':

add\_student\_manually(sms)

elif choice == '5':

add\_teacher\_manually(sms)

elif choice == '6':

add\_course\_manually(sms)

elif choice == '7':

print("Goodbye!")

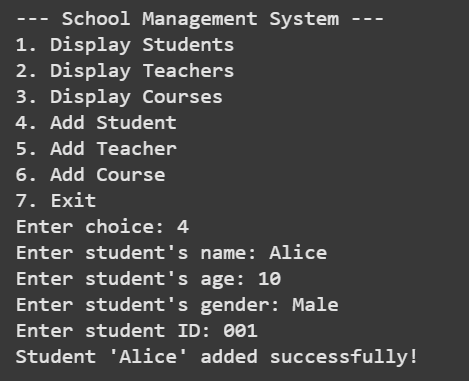
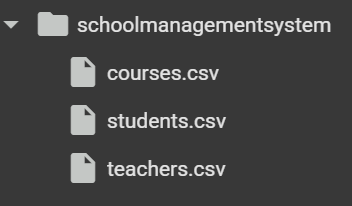
break

else:

print("Invalid choice.")

* Run the cell — your program is now live!

Output:

**Step 6: Use the System**

* Enter menu options like 1, 2, 3, etc. to:
  + Display Student
  + Display Teacher
  + Display Course
  + Add Student
  + Add Teachers
  + Add Courses
  + Exit
* All data will be saved automatically into the schoolmanagementsystem folder in Colab.

5. Conclusion

This project demonstrates how Object-Oriented Programming in Python can be used to efficiently manage and store school data. It also showcases how to run Python projects on Google Colab, manage file systems, and save data using CSV files.