



STUDENT MANAGEMENT SYSTEM

-CONSOLE BASED PROJECT



PURPOSE-

This Student Management System project is a console-based application developed using Python and MySQL. It provides an efficient way to manage student information by allowing users to add, update, delete, and view student records directly from a command-line interface. The system connects to a MySQL database, where student data is stored and retrieved, ensuring organized and secure data handling. The project demonstrates essential CRUD (Create, Read, Update, Delete) operations, database connectivity, offering a practical solution for educational institutions to manage student information systematically.

DATABASE-

1.Database: stud_management

This database is designed to store user information for authentication and detailed records for each student.

2.Tables

1.users Table

- **Purpose: Stores user credentials for authentication, allowing secure access to the Student Management System.**
- **Fields:**
 - **username (VARCHAR(100), Primary Key):** The unique identifier for each user.
 - **password_hash (VARCHAR(64)):** Stores the hashed password for secure login.
 - **phone_number (VARCHAR(15)):** Stores the user's phone number for contact or verification purposes.



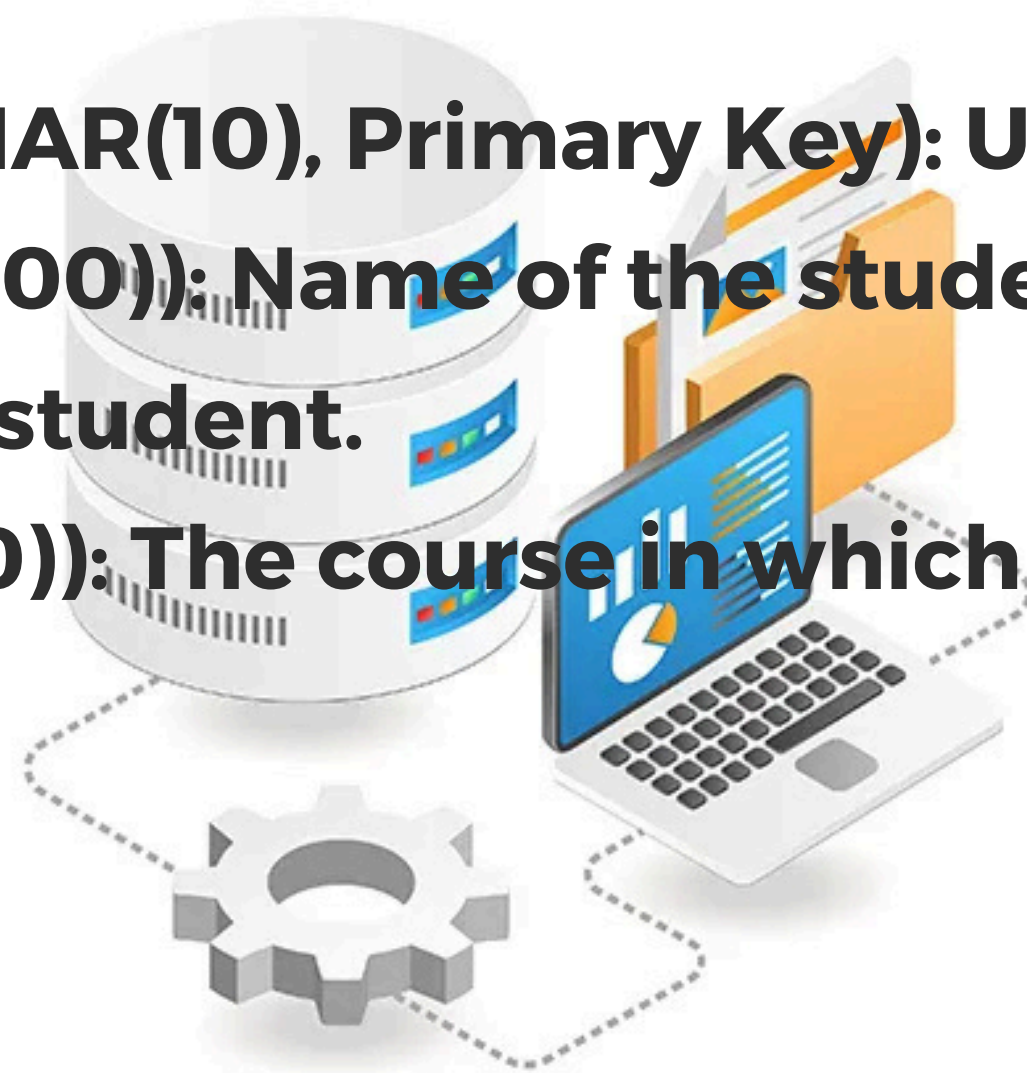
DATABASE-

2. s_students Table

Purpose: Stores detailed records of students, including personal information and enrollment details.

Fields:

1. **roll_number (VARCHAR(10), Primary Key):** Unique ID assigned to each student.
2. **s_name (VARCHAR(100)):** Name of the student.
3. **age (INT):** Age of the student.
4. **course (VARCHAR(50)):** The course in which the student is enrolled.



TABLES

The screenshot displays the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons for file operations and database management. The left sidebar, titled 'Navigator', shows a tree view of the 'stud_management' schema, including tables, columns, indexes, foreign keys, triggers, views, stored procedures, and functions. The 'users' table is currently selected. The main editor window, titled 'stud_management - Schema', shows the SQL script for creating the 'users' and 'students' tables. The 'users' table has columns for 'username' (VARCHAR(100), PRIMARY KEY), 'password_hash' (VARCHAR(64)), and 'phone_number' (VARCHAR(15)). The 'students' table has columns for 'roll_number' (VARCHAR(10), PRIMARY KEY), 'name' (VARCHAR(100)), 'age' (INT), and 'course' (VARCHAR(50)). The right sidebar, titled 'SQLAdditions', contains a message: 'Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.'

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: stud_management - Schema SQL File 6* x

SCHEMAS

Filter objects

stud_management

Tables

students

Columns

Indexes

Foreign Keys

Triggers

users

Views

Stored Procedures

Functions

sys

1 • CREATE DATABASE stud_management;

2 • USE stud_management;

3

4 • CREATE TABLE users (

5 username VARCHAR(100) PRIMARY KEY,

6 password_hash VARCHAR(64),

7 phone_number VARCHAR(15)

8);

9

10 • CREATE TABLE students (

11 roll_number VARCHAR(10) PRIMARY KEY,

12 name VARCHAR(100),

13 age INT,

14 course VARCHAR(50)

15);

16

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

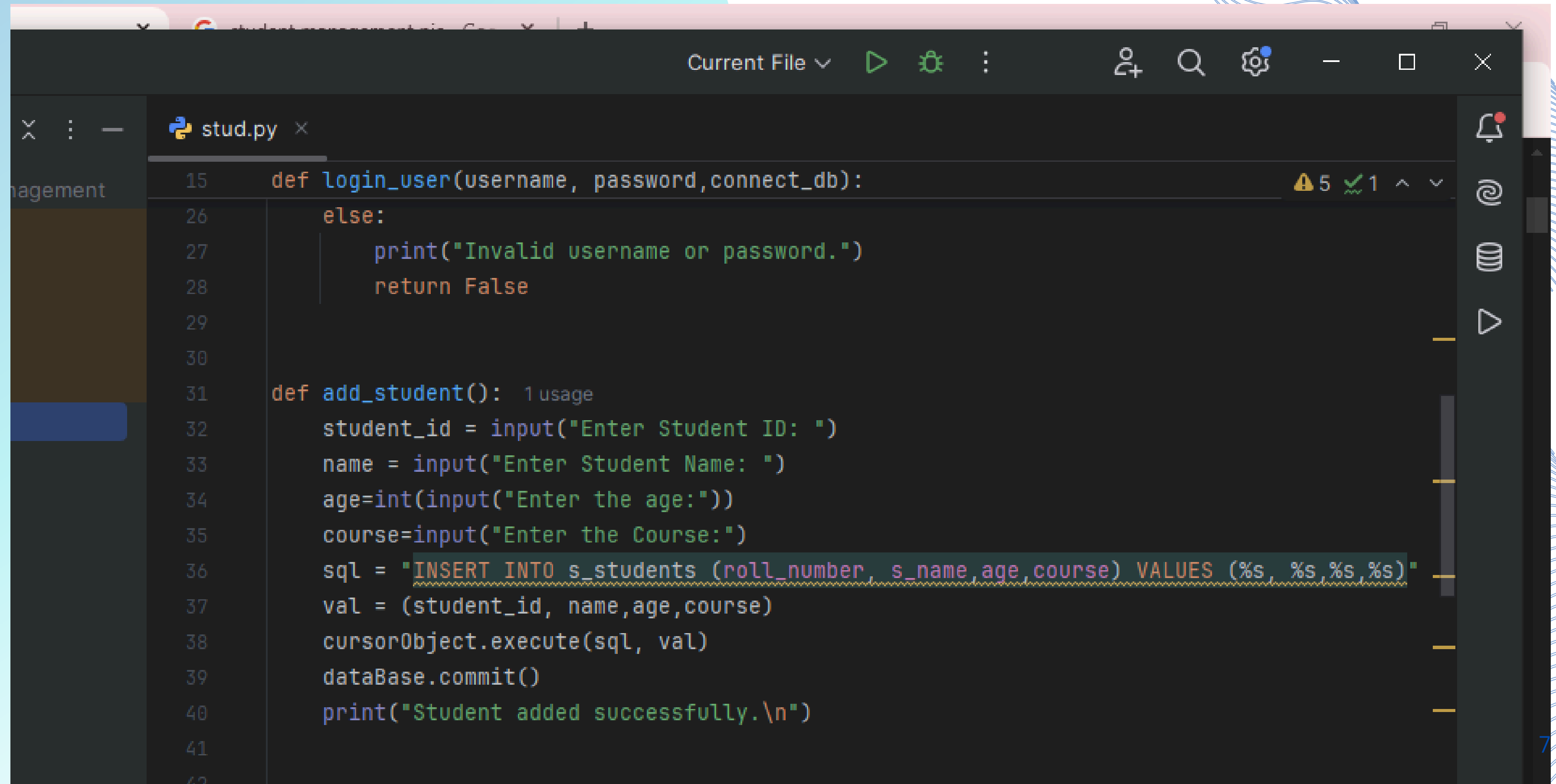
Administration Schemas Context Help Snippets

PYTHON CODE: CORE FUNCTIONS

- **ADD STUDENT:** TAKES INPUT AND INSERTS DATA INTO S_STUDENTS.
- **UPDATE STUDENT:** MODIFIES EXISTING STUDENT DATA BY ID.
- **DELETE STUDENT:** DELETES A STUDENT'S RECORD BY ID.
- **VIEW STUDENTS:** RETRIEVES AND DISPLAYS ALL STUDENT RECORDS.



ADD STUDENT FUNCTION

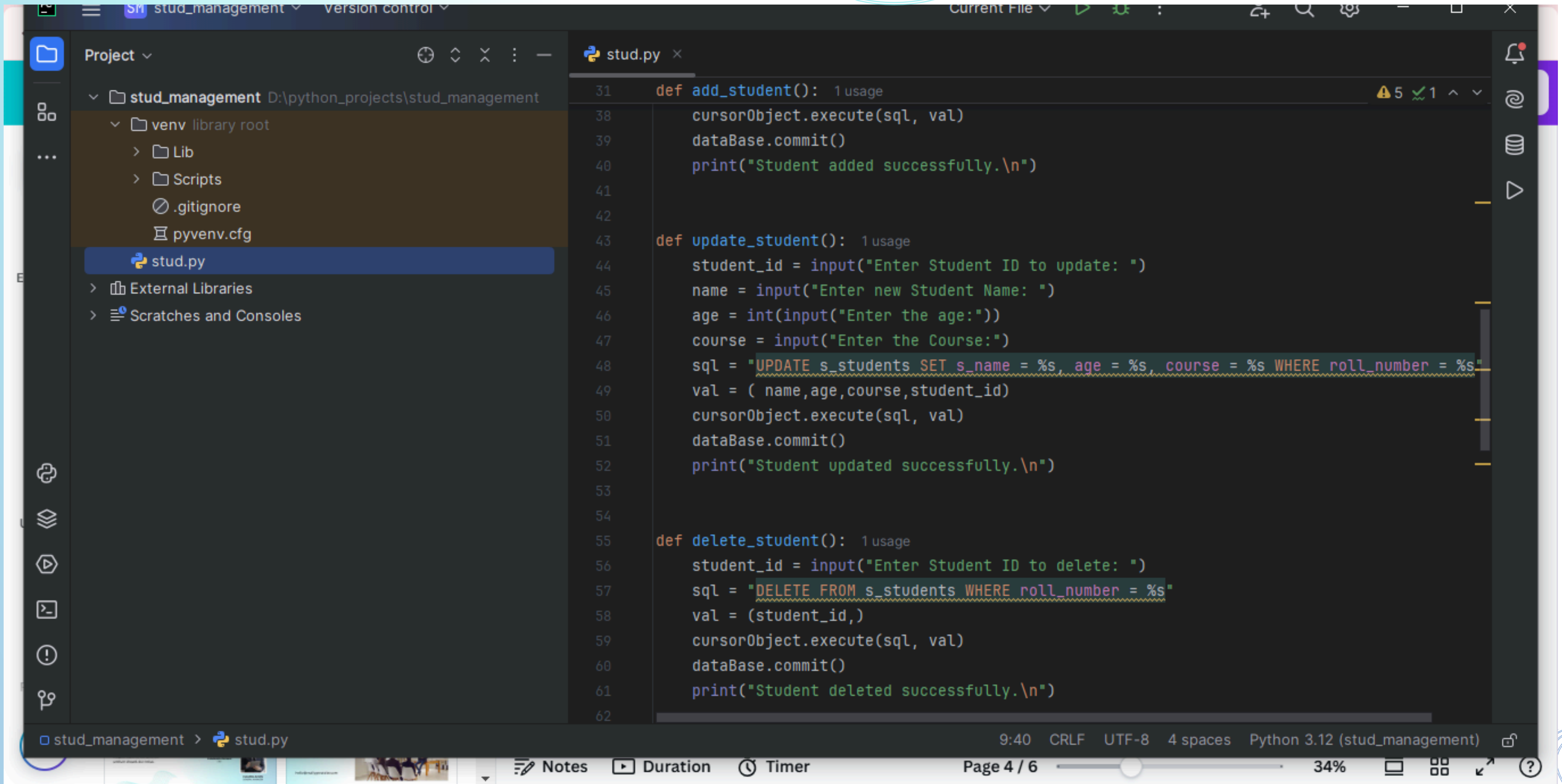


The image shows a code editor window with a dark theme. The title bar at the top says 'Current File' and has icons for running, settings, and search. The editor has a sidebar on the left with a file explorer showing 'management' and 'stud.py'. The main area displays the code for the 'add_student' function. The code is as follows:

```
15 def login_user(username, password, connect_db):
26     else:
27         print("Invalid username or password.")
28         return False
29
30
31 def add_student(): 1 usage
32     student_id = input("Enter Student ID: ")
33     name = input("Enter Student Name: ")
34     age = int(input("Enter the age:"))
35     course = input("Enter the Course:")
36     sql = "INSERT INTO s_students (roll_number, s_name, age, course) VALUES (%s, %s, %s, %s)"
37     val = (student_id, name, age, course)
38     cursorObject.execute(sql, val)
39     dataBase.commit()
40     print("Student added successfully.\n")
41
42
```

The SQL statement on line 36 is highlighted with a yellow background. The editor also shows a status bar at the bottom right with a warning icon, '5', a checkmark, '1', and a dropdown arrow.

UPDATEFUNCTION-



The screenshot shows a code editor with a project named 'stud_management' and a file named 'stud.py'. The code defines three functions: 'add_student()', 'update_student()', and 'delete_student()'. The 'update_student()' function is highlighted, showing its implementation for updating a student's name, age, and course based on their roll number.

```
31 def add_student(): 1 usage
38     cursorObject.execute(sql, val)
39     DataBase.commit()
40     print("Student added successfully.\n")
41
42
43 def update_student(): 1 usage
44     student_id = input("Enter Student ID to update: ")
45     name = input("Enter new Student Name: ")
46     age = int(input("Enter the age:"))
47     course = input("Enter the Course:")
48     sql = "UPDATE s_students SET s_name = %s, age = %s, course = %s WHERE roll_number = %s"
49     val = ( name,age,course,student_id)
50     cursorObject.execute(sql, val)
51     DataBase.commit()
52     print("Student updated successfully.\n")
53
54
55 def delete_student(): 1 usage
56     student_id = input("Enter Student ID to delete: ")
57     sql = "DELETE FROM s_students WHERE roll_number = %s"
58     val = (student_id,)
59     cursorObject.execute(sql, val)
60     DataBase.commit()
61     print("Student deleted successfully.\n")
62
```

The status bar at the bottom indicates the file is 'stud.py' in the 'stud_management' project, using Python 3.12, with 4 spaces for indentation and UTF-8 encoding. The page number is 4 out of 6.

CODE OVERVIEW-

This Python script implements a console-based Student Management System with MySQL database integration. It establishes a connection using `mysql.connector` and authenticates users via a login function that validates username and password from the users table. Once authenticated, the system enables CRUD operations on student records stored in the `s_students` table. Specifically, it uses parameterized SQL queries in functions for adding, updating, deleting, and retrieving student records, each using SQL commands (`INSERT`, `UPDATE`, `DELETE`, `SELECT`) and committing changes via `dataBase.commit()` to ensure ACID compliance. The main function contains an interactive loop presenting a menu, where user input dictates which operation to execute. Finally, the database connection is closed upon program termination to release resources securely.

THANK YOU!

AAYUSHI SHARMA

