**Student Performance Dashboard**

**Overview**

The Student Performance Dashboard is a Python-based application designed to manage, analyze, and visualize student performance data. It leverages a database backend for data storage and provides tools for data insertion, querying, and reporting.

**Project Structure**

**StudentPerformanceDashboard/**

├── app.py # Main application logic (likely dashboard or API)

├── conn.py # Database connection utilities

├── db\_create.py # Database schema creation scripts

├── db\_insert.py # Scripts for inserting data into the database

```

**Project Steps**

**1. Requirements Gathering**

- Define the scope: Track and analyze student performance data as per the requirement.

- Identify key features: Data storage, data entry, querying, and reporting.

**2. Database Design**

- Design the schema for storing student,Programming, Soft Skills, Placements data

**- Students Table**

This table stores basic information about students enrolled in the course.

● student\_id (Primary Key): Unique identifier for each student.

● name: Full name of the student.

● age: Age of the student.

● gender: Gender of the student (e.g., Male, Female, Other).

● email: Email address of the student.

● phone: Contact number of the student.

● enrollment\_year: Year when the student enrolled in the course.

● course\_batch: Name of the batch or cohort the student belongs to.

● city: City of residence for the student.

● graduation\_year: Expected or actual graduation year for the student.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**- Programming Table**

This table stores details of students' programming performance in the course.

● programming\_id (Primary Key): Unique identifier for each programming record.

● student\_id (Foreign Key): References the student\_id in the Students Table.

● language: Programming language being evaluated (e.g., Python, SQL).

● problems\_solved: Total number of coding problems solved by the student.

● assessments\_completed: Number of assessments completed by the student.

● mini\_projects: Number of mini projects submitted by the student.

● certifications\_earned: Number of programming certifications earned by the student.

● latest\_project\_score: Score received in the most recent programming project.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**- Soft Skills Table**

This table stores data on students' performance in soft skills evaluations.

● soft\_skill\_id (Primary Key): Unique identifier for each soft skill record.

● student\_id (Foreign Key): References the student\_id in the Students Table.

● communication: Communication skills score (out of 100).

● teamwork: Teamwork skills score (out of 100).

● presentation: Presentation skills score (out of 100).

● leadership: Leadership skills score (out of 100).

● critical\_thinking: Critical thinking skills score (out of 100).

● interpersonal\_skills: Interpersonal skills score (out of 100).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**- Placements Table**

This table stores details related to students’ placement readiness and outcomes.

● placement\_id (Primary Key): Unique identifier for each placement record.

● student\_id (Foreign Key): References the student\_id in the Students Table.

● mock\_interview\_score: Score in the mock interviews (out of 100).

● internships\_completed: Total number of internships completed by the student.

● placement\_status: Placement readiness status (e.g., Ready, Not Ready, Placed).

● company\_name: Name of the company where the student got placed (if applicable).

● placement\_package: Package offered during placement (in USD or local currency).

● interview\_rounds\_cleared: Number of interview rounds cleared by the student.

● placement\_date: Date when the placement offer was received.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**-Data Set Explanation**

The dataset simulates real-world student performance metrics for a data science course:

● Relationships: Tables are connected via student\_id.

● Realism: Data is generated using Faker to mimic plausible student data.

● Preprocessing: Ensure data types align with the schema. Validate relationships.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

- Implement schema creation in `db\_create.py`.

**3. Database Connection**

- Create reusable database connection logic in `conn.py`.

- Ensure secure and efficient connection handling.

**4. Data Insertion**

- Develop scripts in `db\_insert.py` to populate the database with sample or real data.

- Validate data before insertion to maintain integrity.

**5. Application Logic**

- Implement the main application in `app.py`.

- Provide features for querying student performance, generating reports, and possibly visualizing data.

**6. Testing**

- Test database creation, data insertion, and application features.

- Handle edge cases (e.g., missing data, invalid entries).

**7. Documentation**

- Document code and usage instructions.

- Provide insights and recommendations for future improvements.

**Key Insights**

- \*\*Modular Design:\*\* Separating database connection, schema creation, and data insertion improves maintainability and scalability.

- \*\*Data Integrity:\*\* Validating data before insertion prevents errors and ensures reliable analytics.

- \*\*Reusability:\*\* Centralizing connection logic in `conn.py` avoids code duplication and simplifies updates.

- \*\*Extensibility:\*\* The structure allows for easy addition of new features, such as advanced analytics or a web interface.

- \*\*Testing:\*\* Early and thorough testing of each module reduces bugs and improves reliability.

**Technical Specifications**

**Programming Language**

- \*\*Python 3.x\*\*: The core logic, database operations, and scripts are implemented in Python.

**Database**

- \*\*MySQL\*\*: A relational database is used for storing student, programming, soft skills, and placement data.

**Libraries & Packages**

- \*\*mysql-connector-python\*\* or \*\*PyMySQL\*\*: External Python packages used for interacting with MySQL databases. (Note: Python does not include a built-in MySQL library; you must install one of these packages using pip.)

- \*\*Faker\*\*: Used for generating realistic sample data for students and related tables.

- \*\*pandas\*\* (optional, if used): For data manipulation and analysis.

- \*\*Other Standard Libraries\*\*: Such as `os`, `sys`, and `datetime` for file and date operations.

**Project Structure**

- \*\*app.py\*\*: Main application logic (dashboard, reporting, or API).

- \*\*conn.py\*\*: Handles database connection and utility functions.

- \*\*db\_create.py\*\*: Contains scripts to create the database schema and tables.

- \*\*db\_insert.py\*\*: Scripts for inserting sample or real data into the database.

**Data Model**

- \*\*Relational Schema\*\*: Four main tables (Students, Programming, Soft Skills, Placements) with foreign key relationships.

- \*\*Data Types\*\*: Integer, Text, Date, and Float fields as appropriate for each table.

**Data Generation**

- \*\*Faker Library\*\*: Used to generate realistic, random data for testing and demonstration purposes.

**Platform**

- \*\*Cross-platform\*\*: Runs on Windows, Linux, or MacOS with Python 3.x installed.

**How to Run Scripts**

- All scripts are run from the command line using Python:

```bash

python db\_create.py

python db\_insert.py

python app.py

Streamlit run app.py

**Future Improvements**

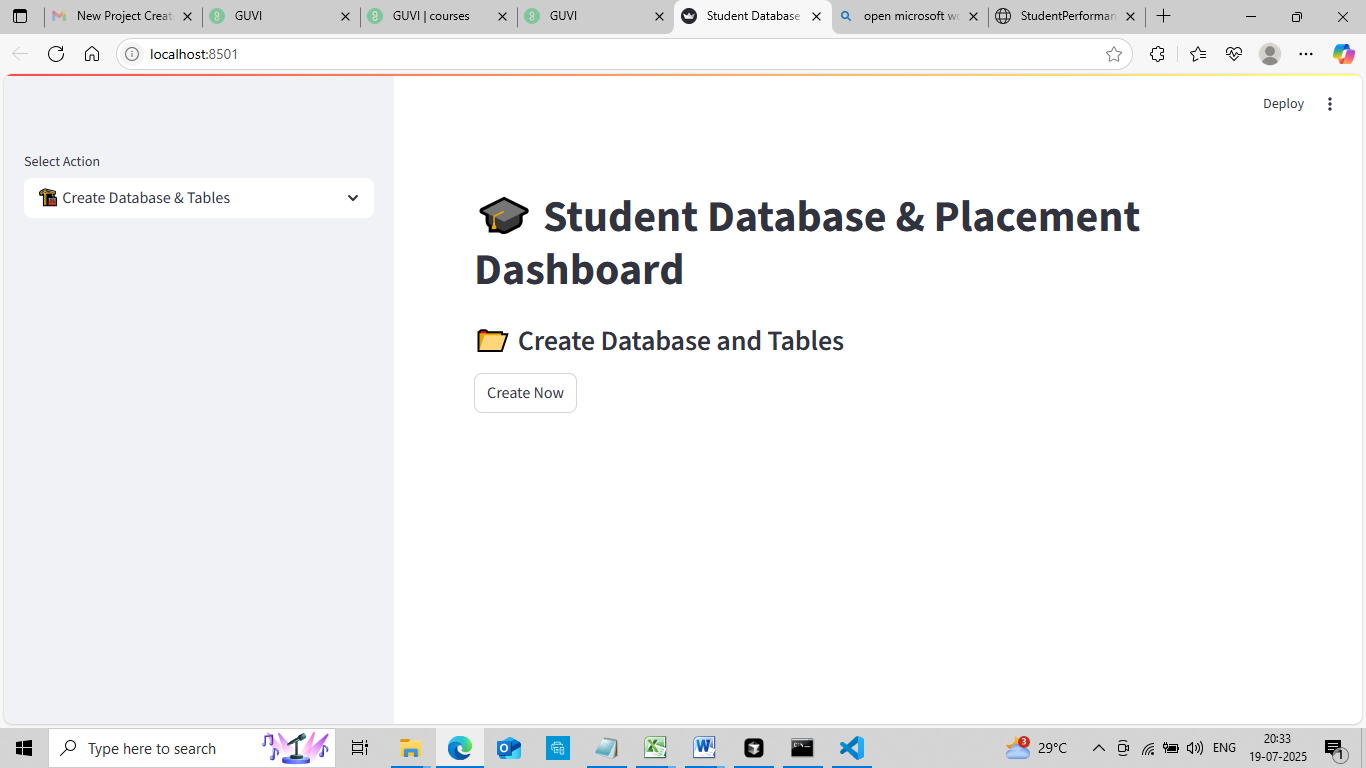
- Add a web-based dashboard for interactive data visualization.

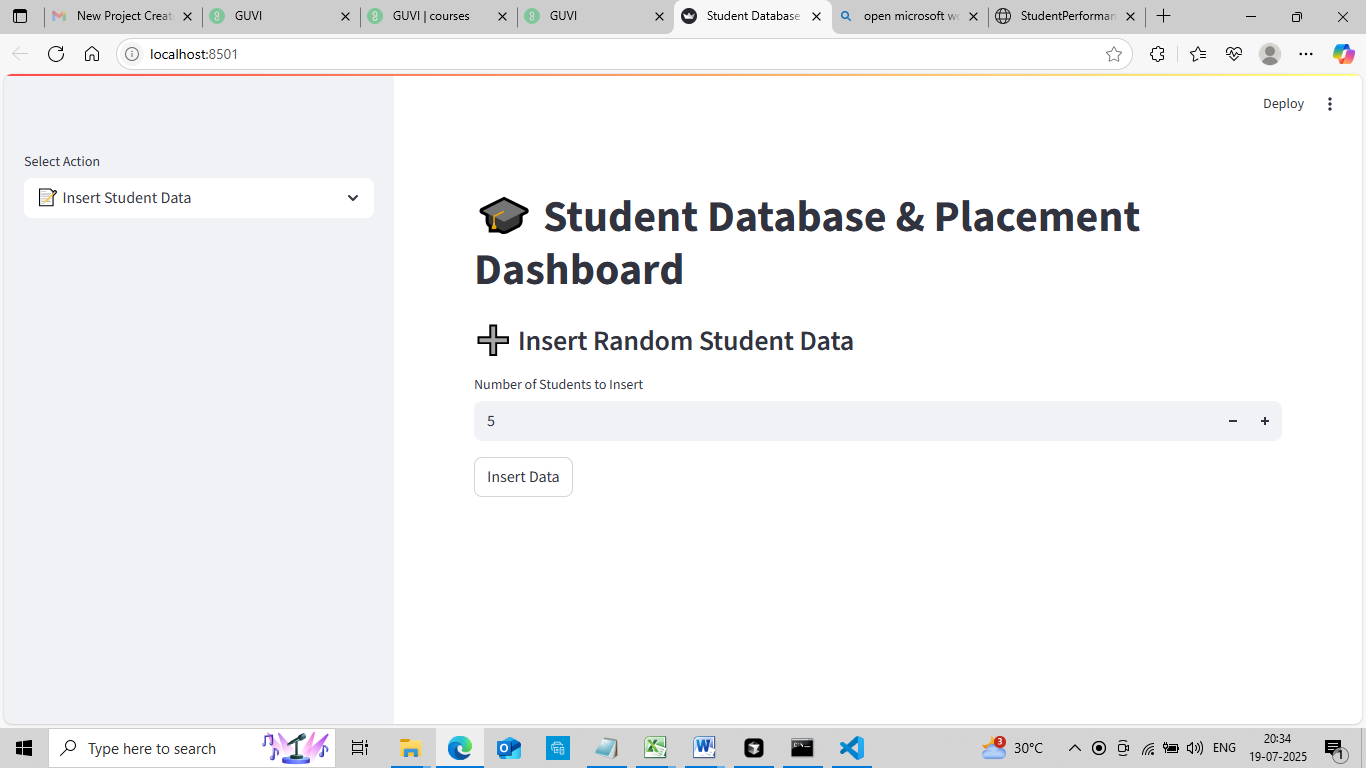
- Implement user authentication for secure access.

- Integrate advanced analytics (e.g., predictive modeling).

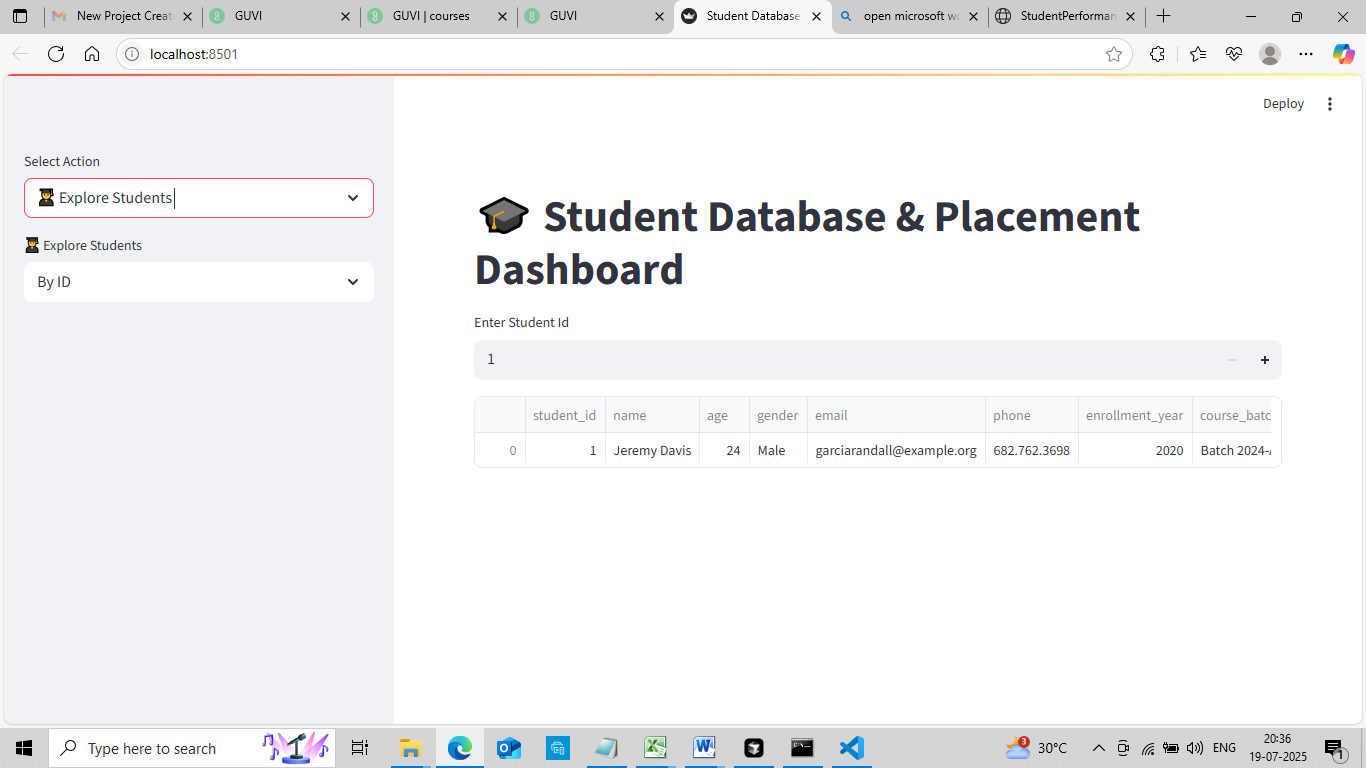
- Automate data import/export.

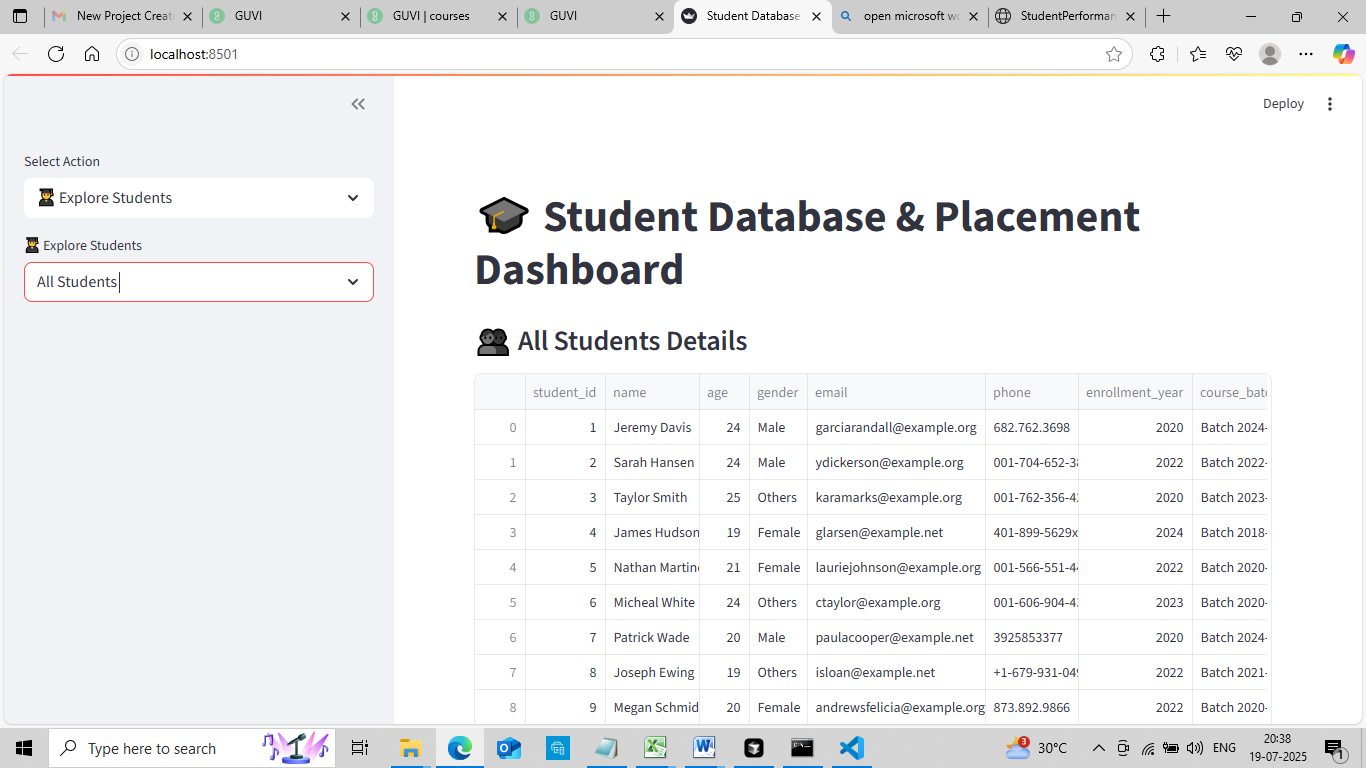
**ScreenShots :**  
**Create Database and Tables :**

  
**Insert Data in Table :**

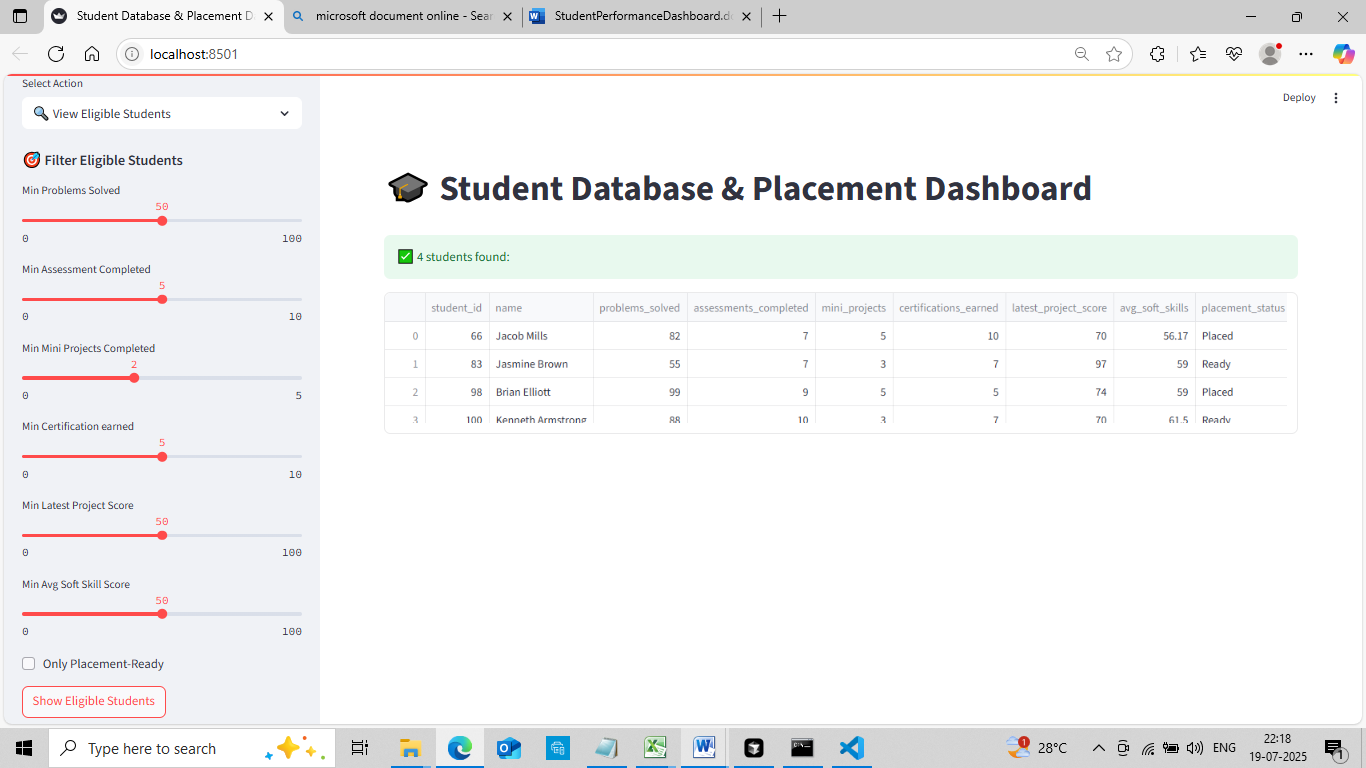


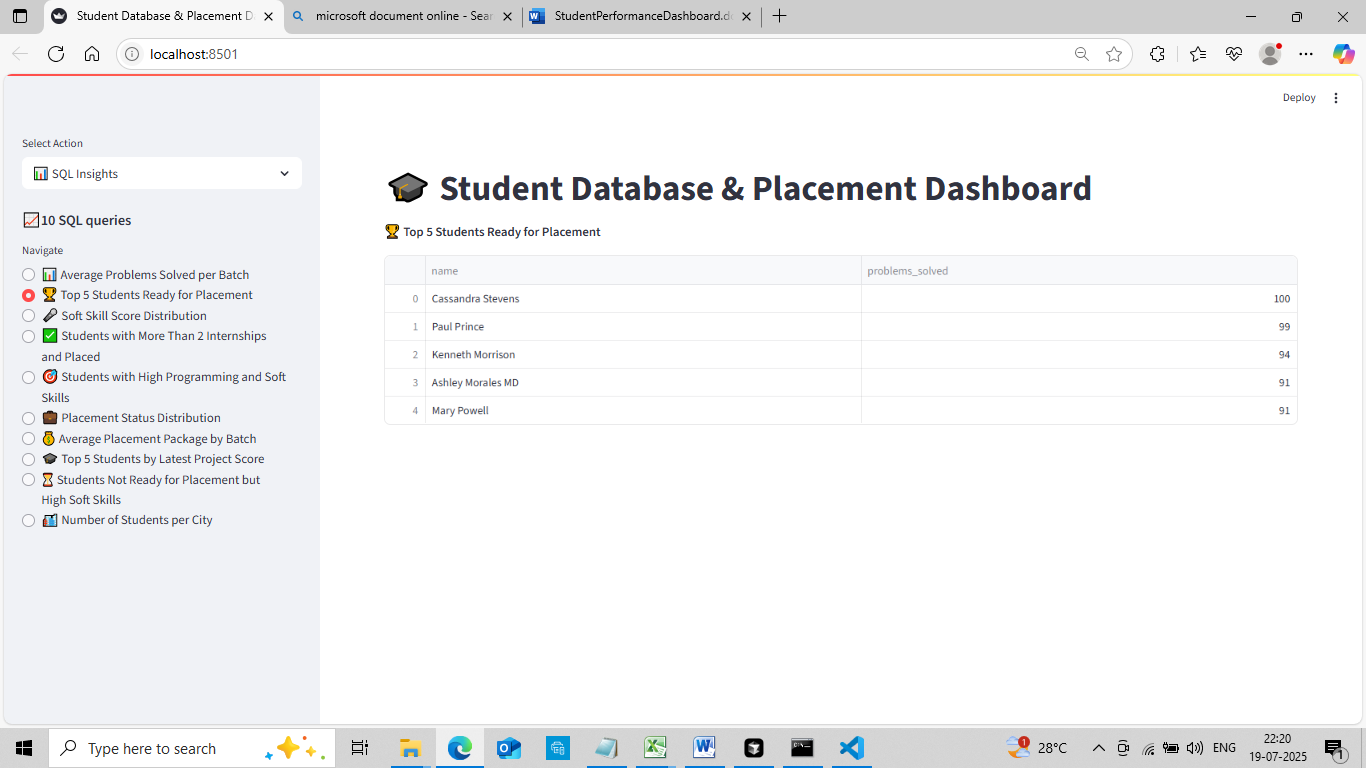
**Explore Student details by Id:**

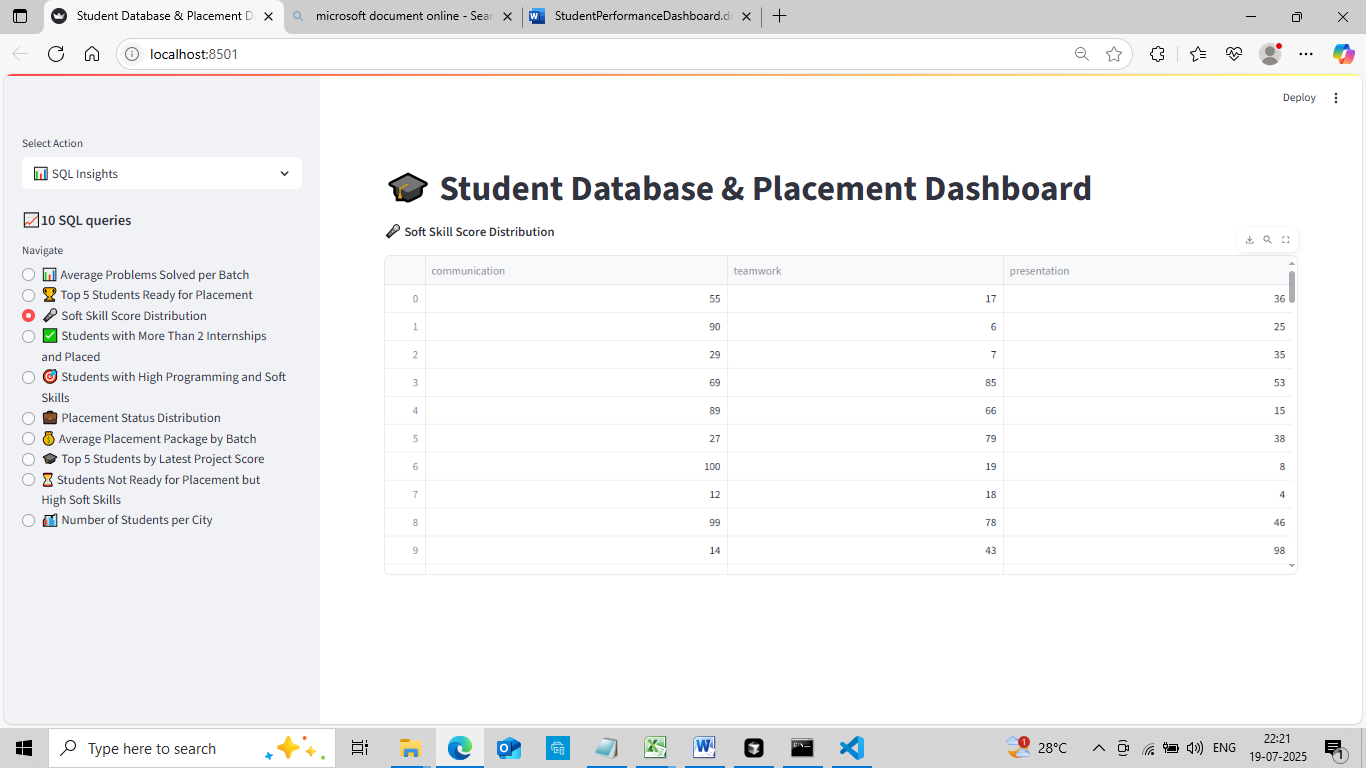
  
**Explore All Students :**

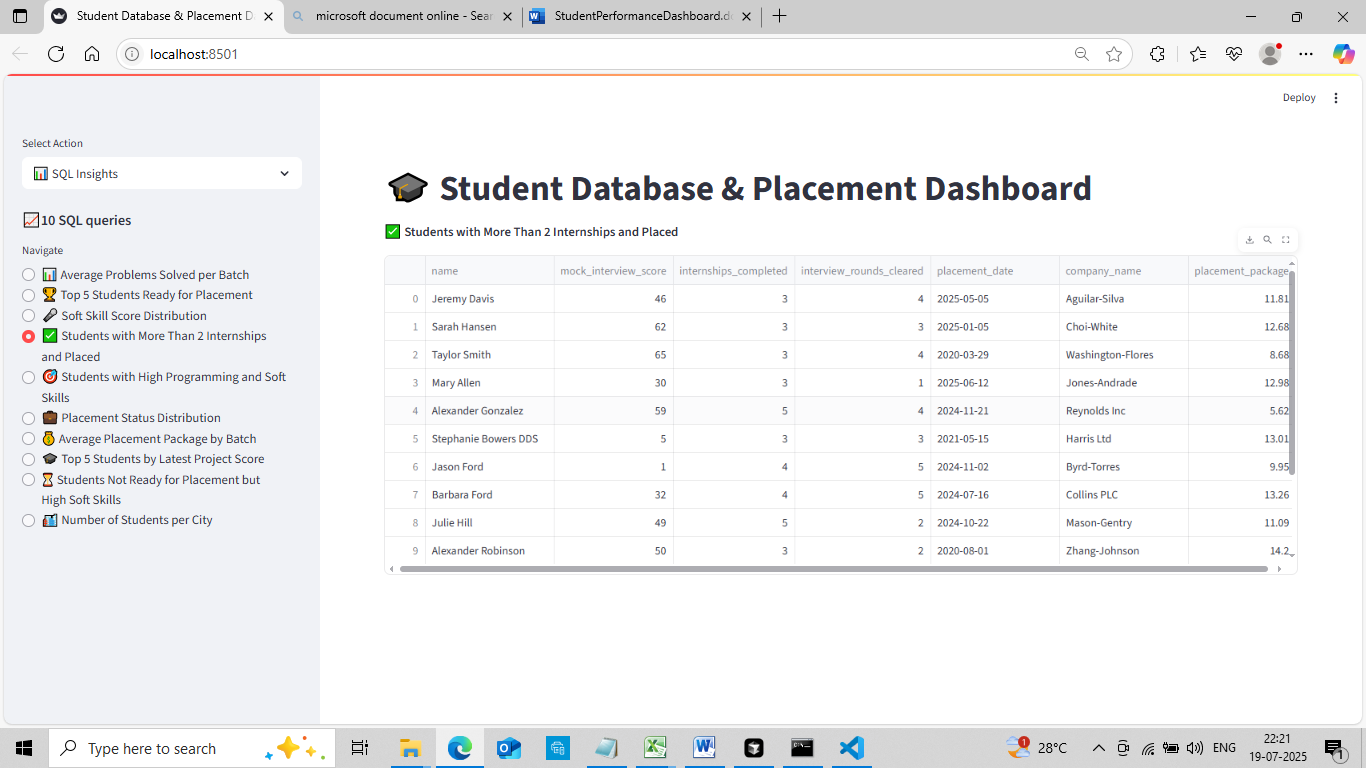
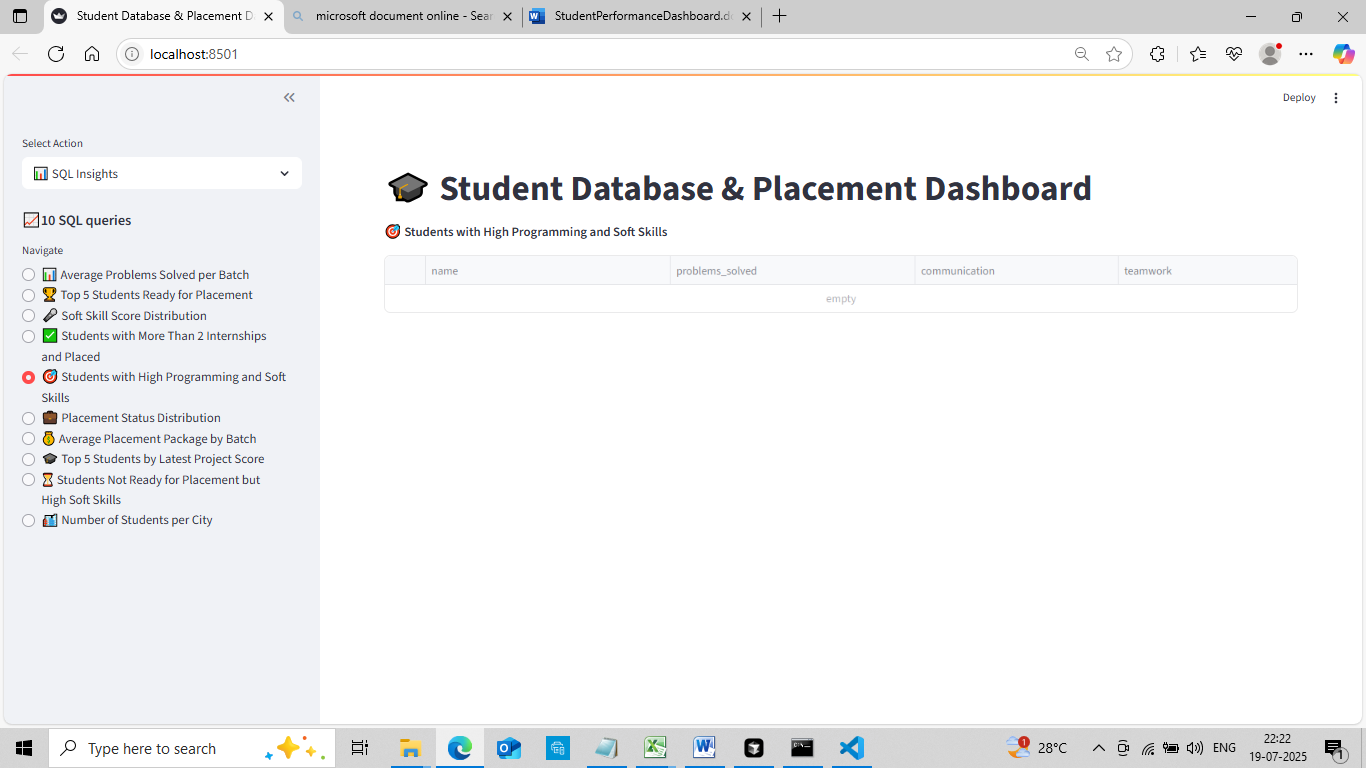
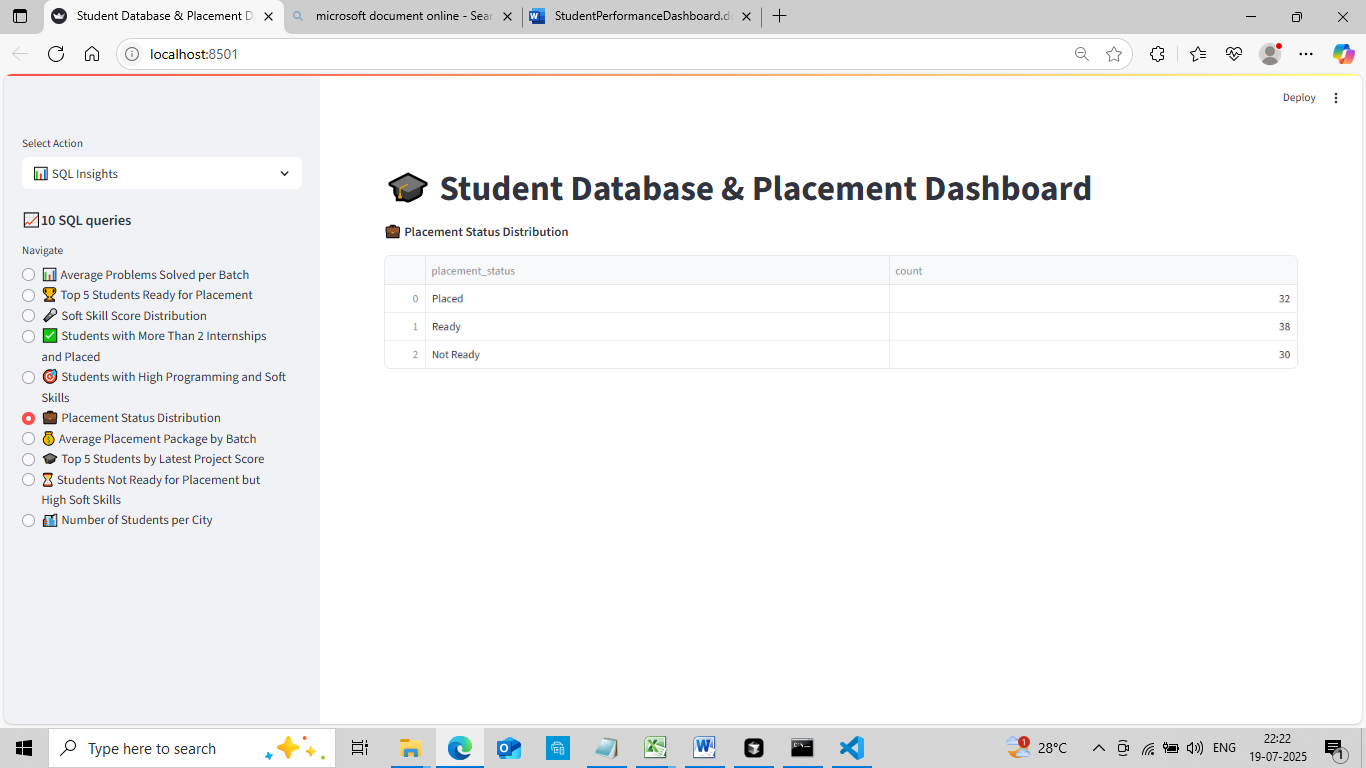
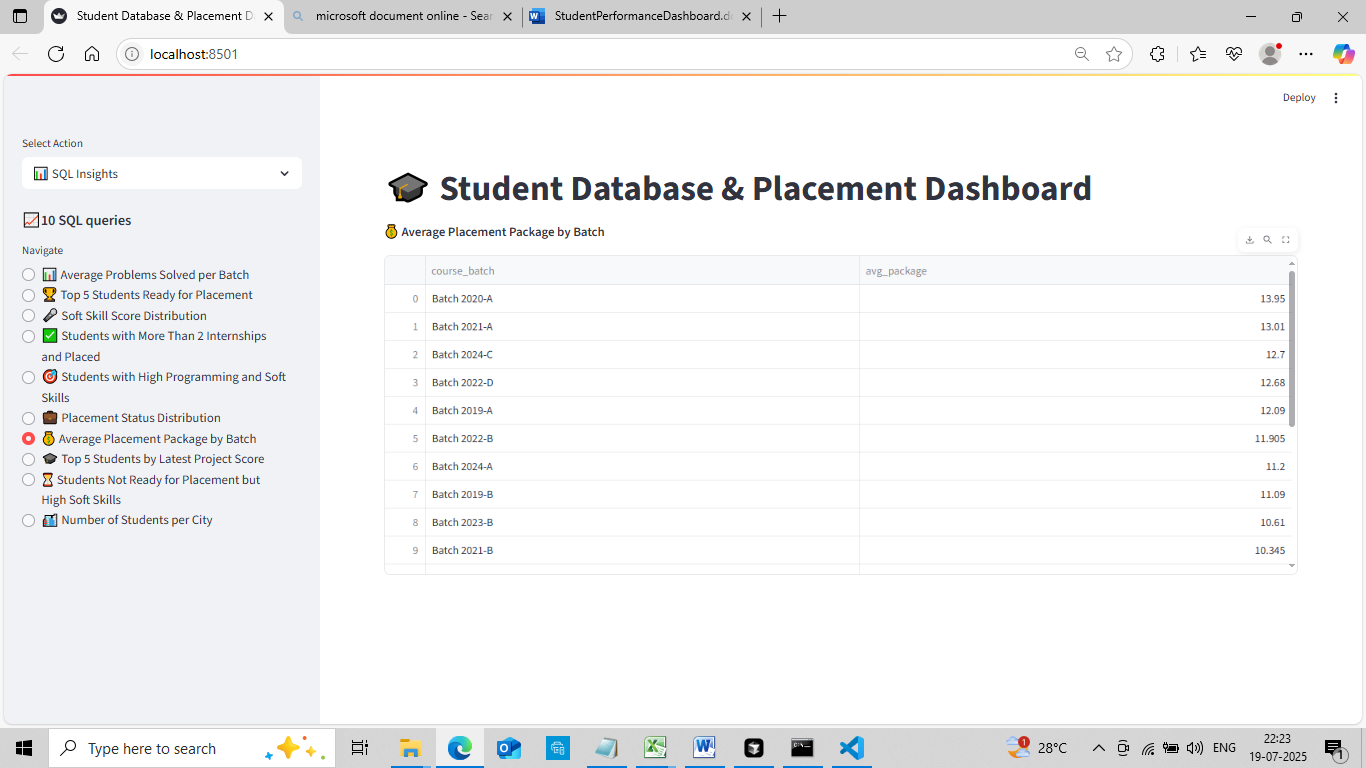
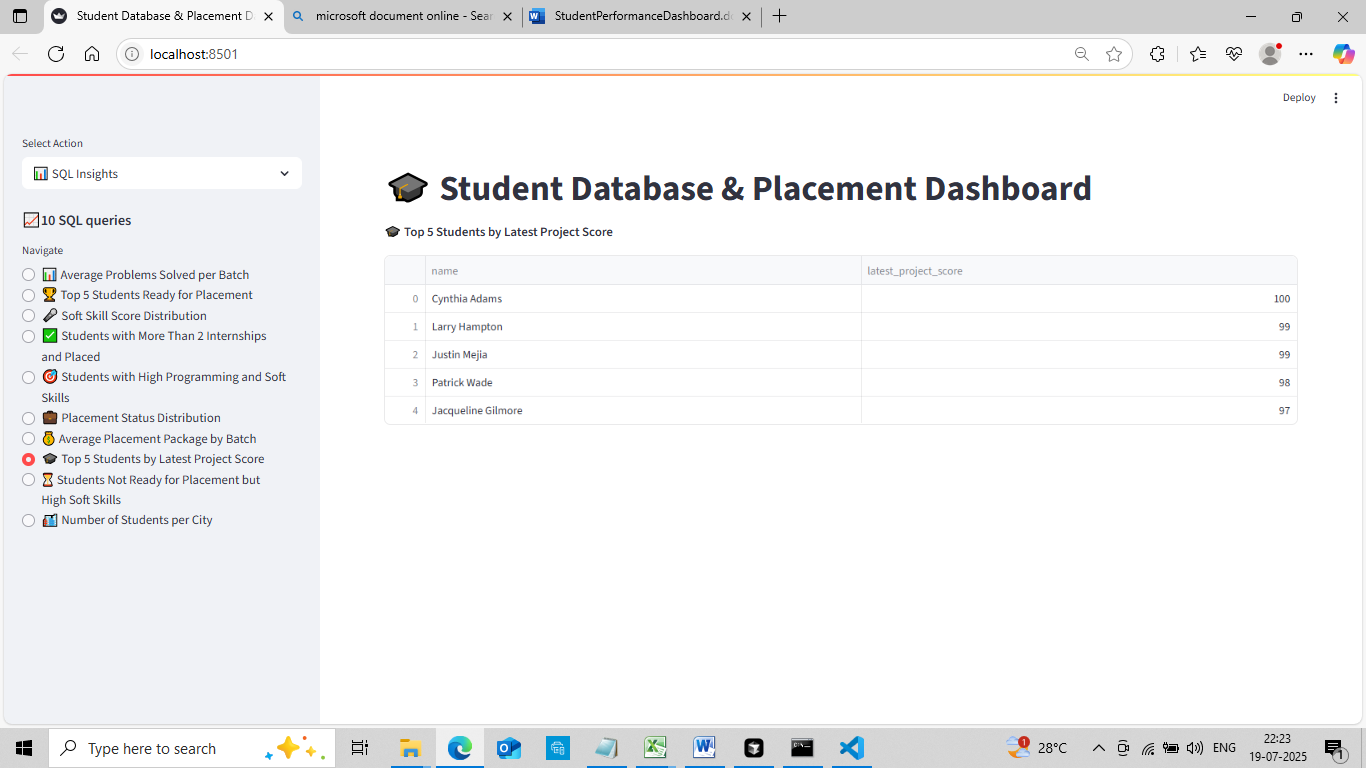
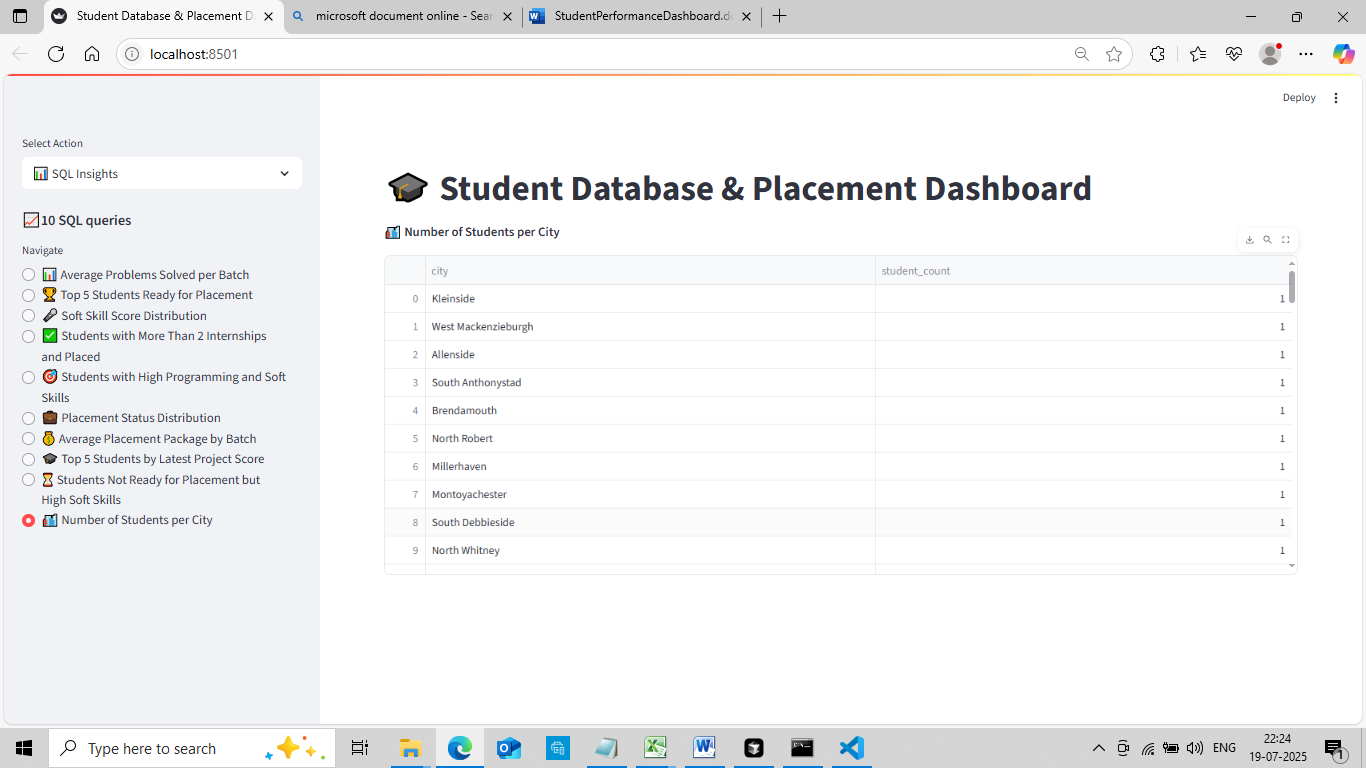
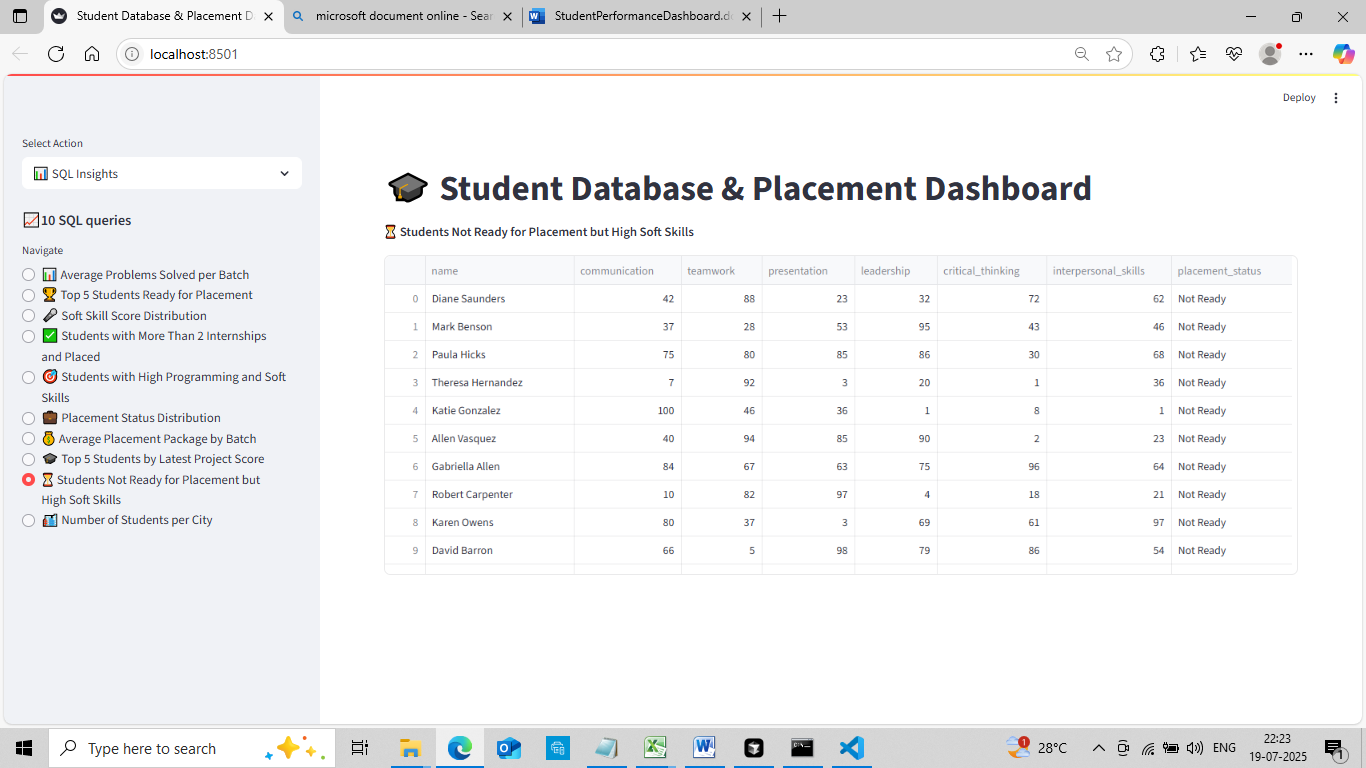


View Eligible Students:

  
  
**10 SQL queries Insights :**





**Conclusion**

This project provides a solid foundation for managing and analyzing student performance data. Its modular design ensures ease of maintenance and future expansion.