**Exercise 2: Problem Statements for Mini-projects**

1. MUST Comply: No fancy looking application is required to be built as part of this exercise. It shall be a simple console/terminal based application. Focus shall ONLY be on logic and code quality as described in the points below.
2. MUST Comply: Coding should be done adopting best practices - Behavioural/structural/creational design patterns, SOLID design principles, OOPs programming, language of candidates choice

**VIRTUAL CLASSROOM MANAEGER PROGRAMMING EXERCISE**

**Problem Statement :**Imagine you are developing the backend for an EdTech platform that aims to host virtual classrooms. Your task is to create a terminal-based Virtual Classroom Manager that handles class scheduling, student attendance, and assignment submissions**.**

**Initial State**

Number of Classrooms: 0

Number of Students: 0

Number of Assignments:0

**User Input**

1. Add Classroom: User types add\_classroom followed by the class name to create a new virtual classroom.

2. Add Student: User types add\_student followed by the student ID and the class name to enroll a student in a classroom.

3. Schedule Assignment: User types schedule\_assignment followed by class name and assignment details to add an assignment for a class.

4. Submit Assignment: User types submit\_assignment followed by student ID, class name, and assignment details to mark an assignment as submitted

**Console Output**

Classroom Addition: "Classroom [Name] has been created."

Student Addition: "Student [ID] has been enrolled in [Class Name]."

Assignment Scheduled: "Assignment for [Class Name] has been scheduled

Assignment Submission: "Assignment submitted by Student [ID] in [Class Name].

**Functional Requirements**

1. **Classroom Management**: Ability to add, list, and remove virtual classrooms.

2. **Student Management**: Ability to enroll students into classrooms, and list students in each classroom.

3. **Assignment Management**: Schedule assignments for classrooms and allow students to submit them.

**Evaluation Criteria**

1. **Code Quality:** Importance will be given to best practices, SOLID principles, and the use of appropriate design patterns.

2**. Functionality**: The terminal-based application should be fully functional and handle various classroom operations efficiently.

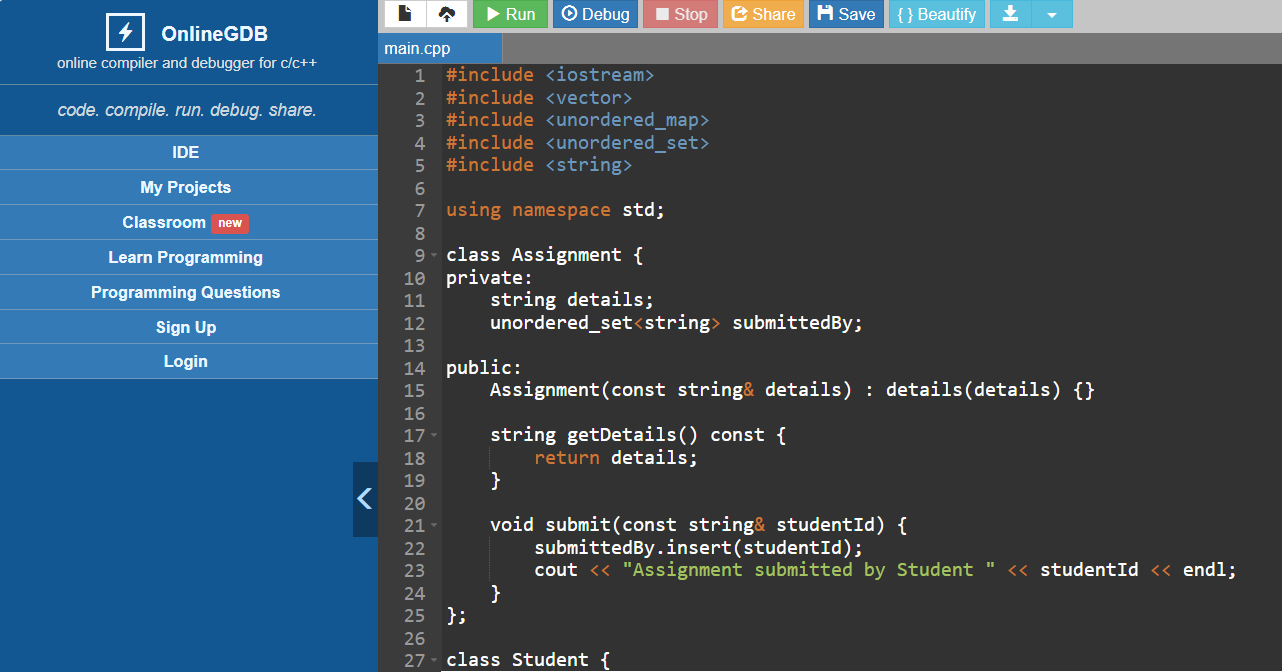
3**. Global Convention**: Adherence to coding standards for readability and maintainability.

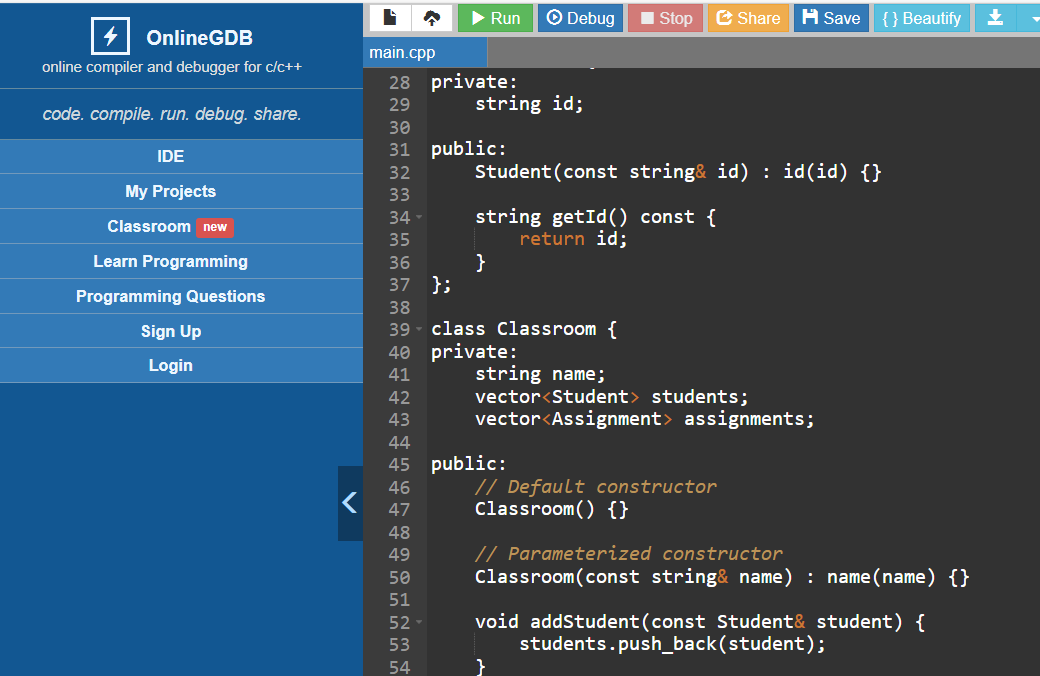
4**. Gold Standards**: The code should include logging, exception handling, and transient error handling.

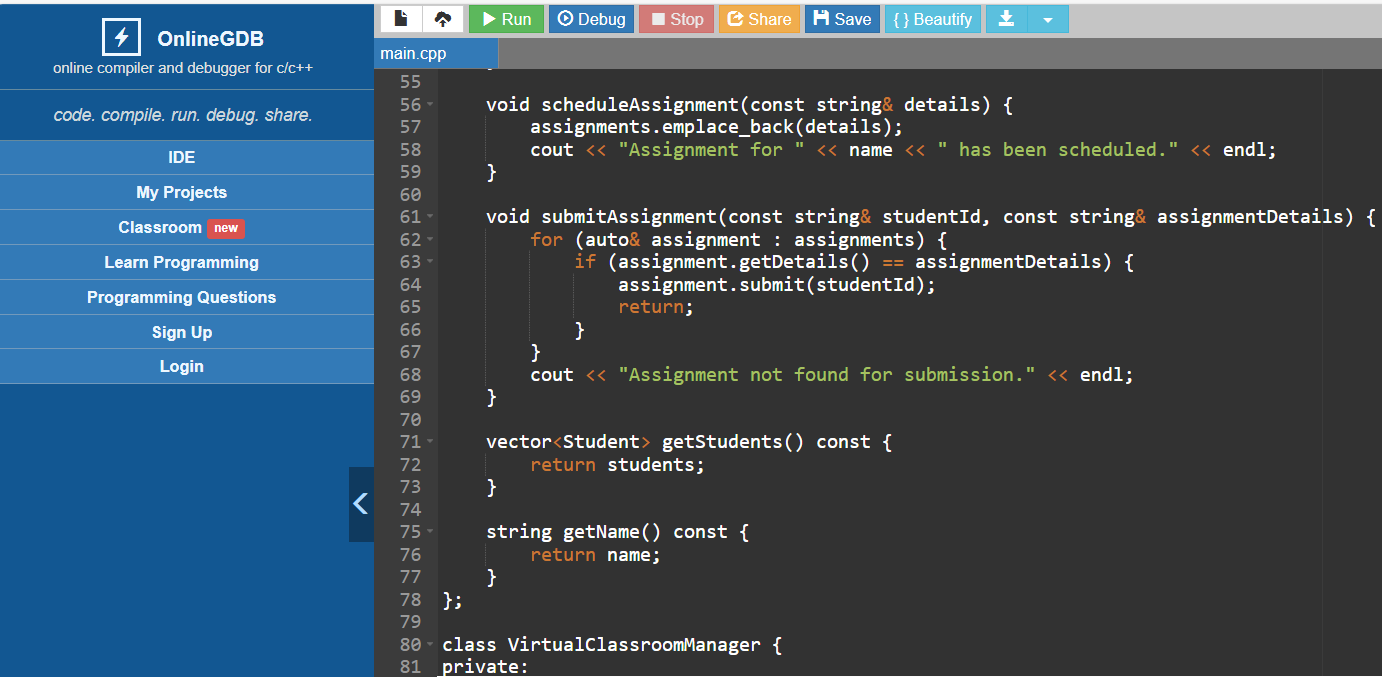
5**. Code Walkthrough**: Candidates should be able to fully walk us through their code and the decisions made during development.

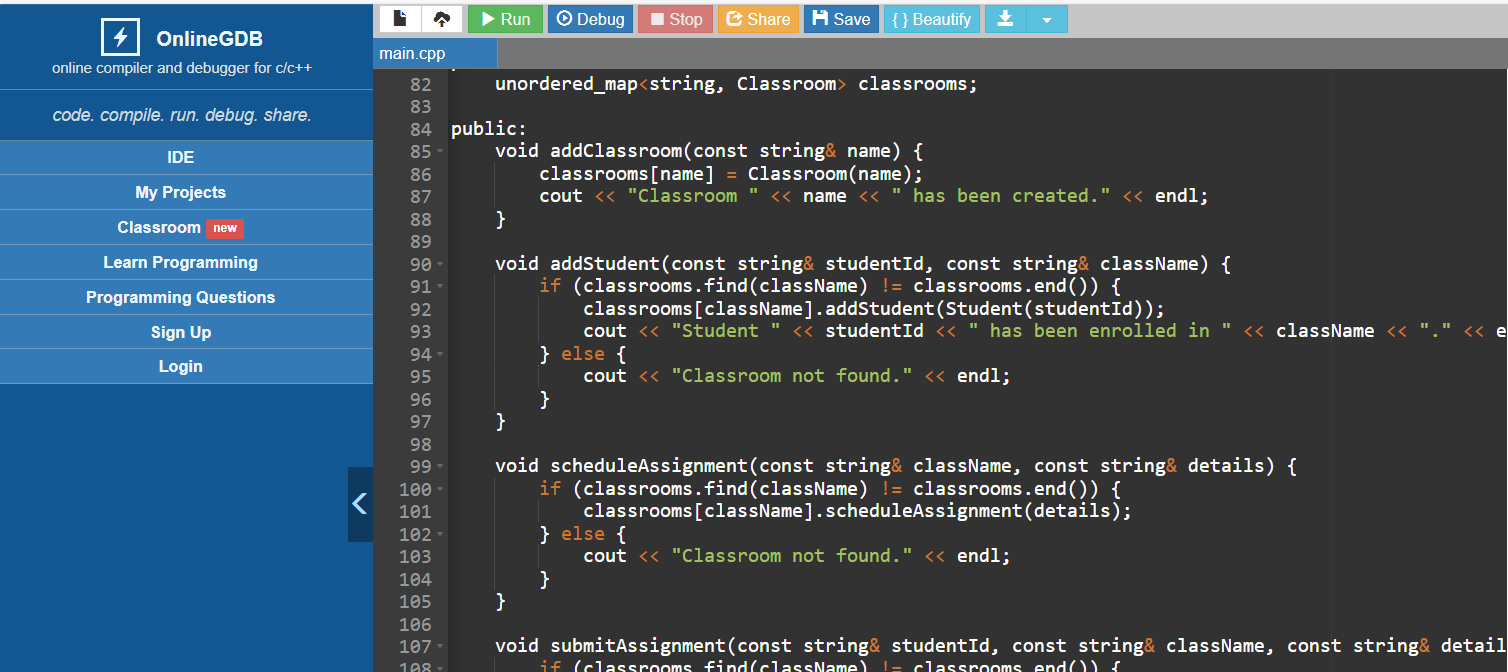
The exercise has been designed to echo the real-world complexities that come with managing an educational platform... It's an engaging problem that evaluates a candidate's ability to model relationships between entities like students and classrooms, and manage state, all within the constraints of a terminal-based application.

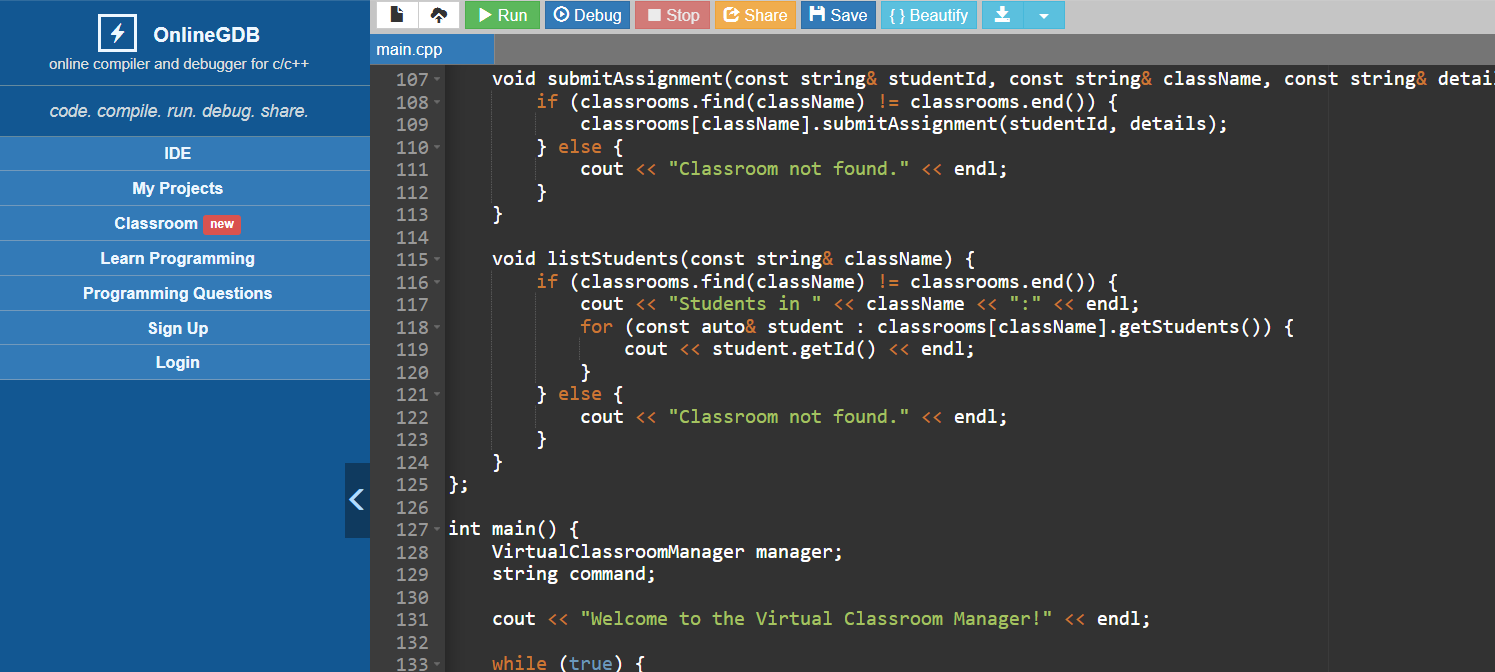
**CODE:**

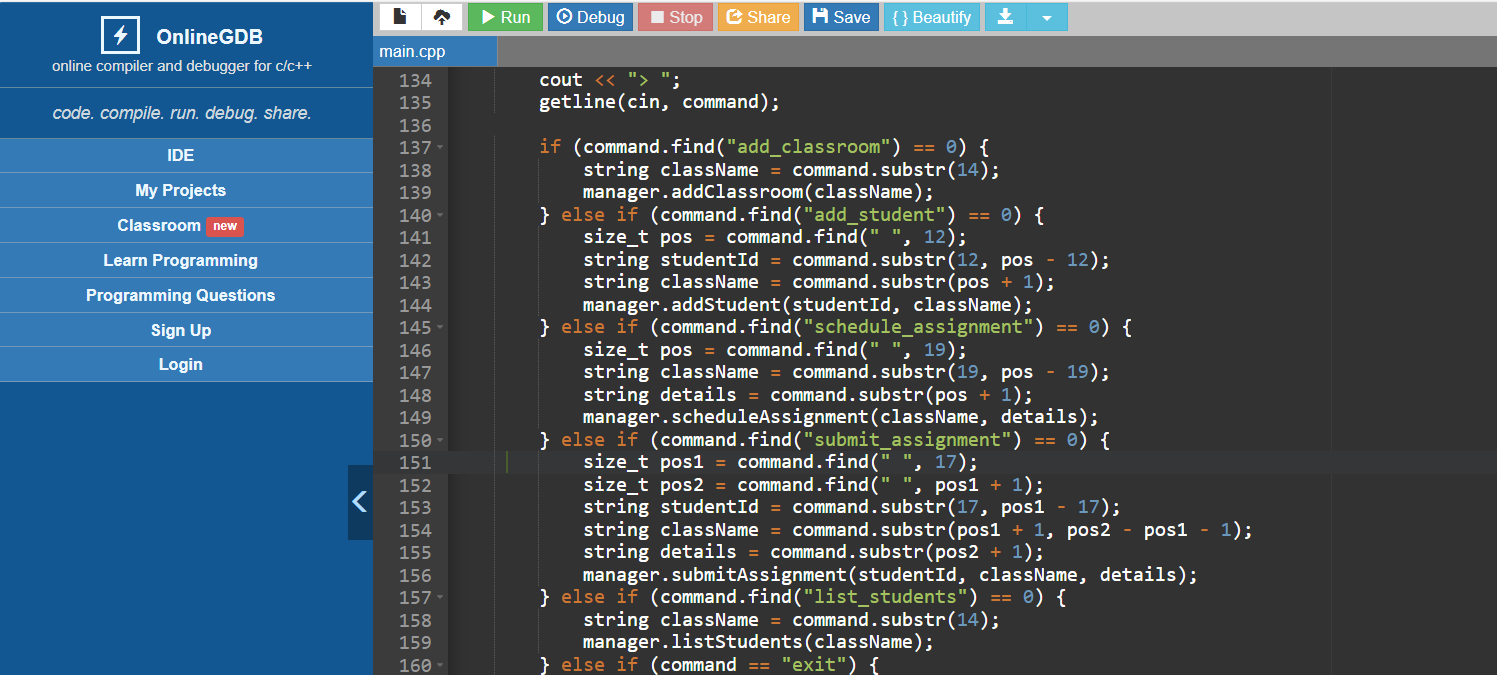
****

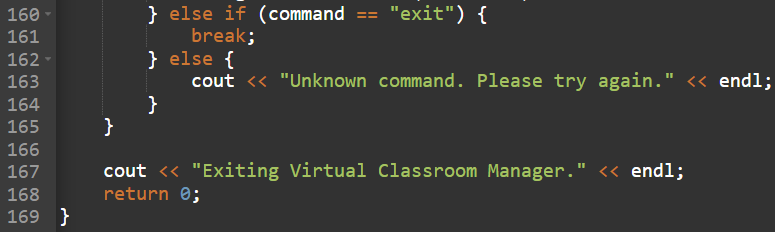
****

****

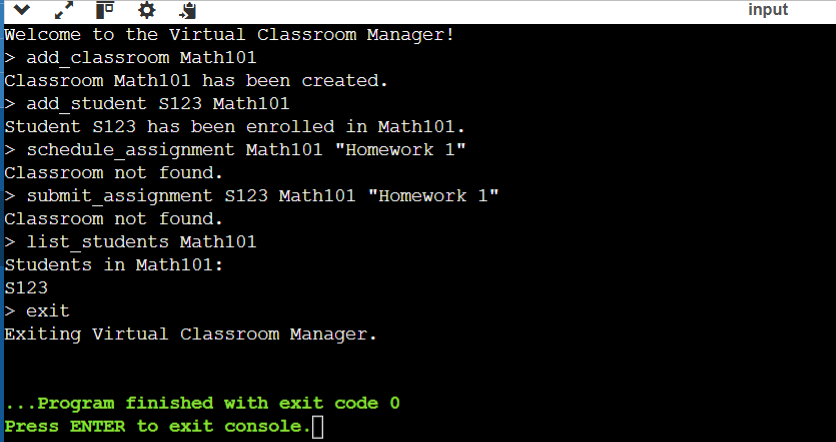
****

****

****

****

**OUTPUT:**

****