Assignment Name: Layered Architecture Model of an Online Judge system:

Creating a layered Architecture model for an Online Judge system involves understanding its structure and components. Here's how you can create and organize yo my handwritten drawing:

Steps to Create Handwritten Diagram:

1. Choose layers:
   • Presentation layer
     → Web or mobile Interface
     → Displays problem, user submissions and results.

   • Application layers:
     → Handles problem submission, solution evaluation and user requests.

   • Business logic layer:
     → Core functionalities like compiling code, running test cases, and scoring.

   • Data Analysis Layer:
     → Interacts with databases for user data, problem sets and submission history.

2. Identity Components:

→ Authentication & Authorization

→ Code Submission & Execution queue.

→ Compiler, test case management, Scoring Algorithm

→ user profile Data, submission log

→ cloud platforms, Sandboxes, Databases.

3. Draw the Diagram:

→ use rectangular box on layers to represent each layer.

→ Add Arrows to indicate data flow between layers.

→ label Components with each layer.

4. Test Case flow example:

→ User submits code → Application queues the Code → Business logic Compiles and tests it → Result stored in the database → presentation display the results

OJ Architecture

```
          ┌─────────────────────────┐
          │    Presentation layer    │
          └─────────────────────────┘
                      │
                      ▼
              ┌───────────────┐
              │      API       │
              └───────────────┘
           ╱         │          ╲
          ╱          │           ╲
  ┌──────────────┐  ┌──────────────────┐  ┌──────────────┐
  │  Submission   │  │ Code Execution    │  │  Submission   │
  │  Database     │  │ Engine            │  │  queue        │
  │               │  │   Sandbox         │  │               │
  │               │  ├──────────────────┤  └──────────────┘
  │               │  │ Compiler/         │
  │               │  │   Interpreter     │
  └──────────────┘  └──────────────────┘
```