ONLINE JUDGE - **CODECANVAS**

Problem Statement:

The Online Judge is a web-based platform that allows users to solve programming problems, submit their solutions, and receive automated evaluations. It provides a collection of coding problems with varying difficulty levels and supports multiple programming languages.

Frontend:

a. Authentication Page:

- Login form
- Registration form
- Password reset functionality

b. Home Page:

- Welcome message
- Quick statistics (e.g., total problems, user ranking etc.)
- Featured or random problem of the day like in Leetcode

  (can add more functions later)

c. Problem List Page:

- Sortable and filterable list of all available problems
- Problem difficulty indicators
- Solved/Unsolved status for logged-in users

d. Problem Statement Page:

- Detailed problem description
- Input/Output specifications
- Constraints and examples
- Code editor with language selection
- Input box for custom test cases
- Output box for results
- Submit button for final evaluation

e. User Profile Page:

- User statistics (problems solved, submissions, etc.)
- Solved problems list
- Submission history

Backend: The backend will be responsible for handling requests from the frontend, processing submissions, and managing the database. Key components include:

a. Authentication:

- User registration and login
- Password hashing and verification
- JWT token generation and validation for authentication

b. Problem Service:

- Retrieve problem lists and individual problem details
- Manage problem data and test cases

c. Submission Service:

- Handle code submissions
- Queue submissions for evaluation
- Execute submitted code in a sandboxed environment
- Compare output with expected results
- Update submission status and user statistics

d. User Service:

- Manage user profiles and statistics
- Handle user-specific data (solved problems, submissions)

Databases: we will use MongoDB, a noSQL database.

a. Users:

- id: String
- username: String
- email: String(we will have authentication based on email)
- password_hash: String

b. Problems:

- id: String
- title: String

- description: String(has the discription of the problem)
- difficulty: (Easy, Medium, Hard)
- tags: an array of strings(contains all the tags related to the problem)
- sample_input: String
- sample_output: String
- constraints: String

c. TestCases :

- id: String
- problem_id: String(reference to Problems)
- input: String
- expected_output: String
- is_sample: bool

d. Submissions:

- id: String(primary key)
- user_id: String(foreign key to Users)
- problem_id: String (reference to Problems' ObjectId)
- language: String(the programming langugae used)
- code: String
- verdict: String (Pending, Running, Accepted, Wrong Answer, Time Limit Exceeded, Memory Limit Exceeded, Runtime Error)
- execution_time: int(in milliseconds)
- memory_used: int (in KB)

Containerization with Docker:

We'll use Docker to containerize our application components. This ensures consistency across different environments and simplifies deployment.
We will run submitted codes in this containerized environment a set **strict resource limits(CPU, memory, execution time). If the code takes more time or memory then the limit then we can return the verdict of TLE or MLE respectively.**
This helps in taking care of many security flaws.