Online Judge

High Level Design

1. System Overview
Online Judge system for code submission and evaluation with user authentication and problem management.

2. Features

Registration and authentication:

- Name, email, password
- Authentication using JWT and encryption using bcrypt.

Profile Management:

- Have access to profile
- Show Solved Problem, Total Submission, contest Participated.

Solution:

- Submit their solution and run using inbuilt compiler.
- If it matches the testcases output then gets accepted

Problems

- **Problem Storage**: Problem statements and test cases
- **Difficulty Levels**: Beginner (5), Easy (10), Medium (15), Hard (20) points
- **Categories**: Problems on each topic.

Contest(Optional):

- live contest and compete with others**.**

Ranking:

No of pts scored. Who solved earlier get better ranking at same no of points.

Challenges

- Malicious code
- May use submit code at same time

- Verdict can get changed if someone gets ACs and inserts malicious code

Solutions:

- **Admin**
  - Rate limiting for many users
  - Implement a msg queue, in which we store the event to execute the code file • **Docker** → Auto scalable containers with load

3. HLD

Web Server

Screen 1: Name screen

- Login / Signup

(can view problem but do not submit if not logged in)

- Login and Signup is authenticated via JWT

Screen 2: Problem List

- Tag-wise: Beginner, Easy, Medium, Hard, and topic wise.
- Can select language for submission
- Submission log (Accepted / Rejected)

Screen 3: Leaderboard (optional)

• Top problem solvers

Listing Problems

- **Frontend**: Create a simple list UI in React that displays the names of each problem and links them to individual problem pages.
- **Backend:** Define an API endpoint in Express.js that handles a GET request to fetch all problems from the database (MongoDB) and return them to the frontend.

Individual Problem

Pageflow in React: Frontend:

- Table:

Result

Problem Statement	Code Submission	
Backend:		Std I/P → O/P

- Node.js gets request to fetch the problem details from DB & returns to frontend.

Code Submission

- **Frontend:** Submit via UI in individual problem page (template)
- **Backend:**Node.js gets request from F.E. → this endpoint should execute the following steps:
  a. Fetch test cases (I/P + expected O/P)
  b. Evaluate the submitted code using Docker → share the verdict
  c. Both return verdict then also with DB suggest repeated in code

Leaderboard

Point System:

- If Hard Accepted → 20
- Medium → 15
- Easy → 10
- Beginner → 5

→ Top 10 scores shown in leaderboard

Backend: Deploy an API in Express JS to handle the GET request to fetch the problem and POST request to submit the problem

Database Design

Problem

- **_id**: string
- **name**: string
- **topic**: list
- **difficulty (Key)**: [beginner, easy, medium, hard]

Submissions

- **probId**: reference to problem document (key)
- **verdict**: string (Accepted or Rejected)
- **submitted at**: date-time
- **by userId**: S1 + U1 + U9 + U11 + U20

Test Cases

- **input**: string
- **output**: string
- **probId**: reference to problem document (key)

User Signup / Login

- **userId**: string
- **password**: string
- **email**: string (ends with .com)
- **name**: string
- **batch**
- **graduation year**

5. Technical Stack

5.1 Backend

Language: Node.js/ Express.js

Database: Mongodb for main data

Queue: Message queue for submissions

Containerization: Docker for code execution