Online Judge System

An online judge system is a web-based platform where users can solve coding problems by writing and submitting code. The platform evaluates the submissions by comparing the output of the user's code with the expected output using predefined test cases. The evaluation result, or verdict, is then displayed to the user.

Expected Features:

1. User Authentication:- Login functionality using username/email and password.

2. Code Submission:- Users can submit code via file upload or an in-built text editor.

3. Problem Library:- Displays a list of available coding problems.

4. Verdict Display:- Users receive feedback on their code submissions, indicating whether the solution is correct.

High-Level Design:

1. Database Schema:

- *Problem Library (JSON structure):*
  - `Problem Title`: string
  - `Problem Statement`: string
- *Test Cases (JSON structure):*
  - `Input`: string
  - `Output`: string
  - `Problem link`: link to problem JSON document
- *Submissions (JSON structure):*
  - `Verdict`: string
  - `Submission Time`: DateTime
  - `Problem link`: link to problem JSON document
- *User Authentication (JSON structure):*
  - `User ID`: string
  - `Password`: string
  - `Email`: string
  - `Username`: string

2. Front End Structure:

- *Home Screen:*
  - Login/SignUp screen
  - Problem Library

- *Problem Screen:*
  - Text editor section
  - Language select section
  - File upload section
  - Submission log section
- *Submissions Screen:*
  - Past Submissions section (with verdicts)
  - Links to past submission code

3. Back End Structure:

- *Home Screen:*
  - Use Express.js to set up an API GET endpoint to return all problems from the database to the front end.
- *Problem Screen:*
  - Use Express.js to set up an API GET endpoint to return specific problem data from the database to the front end.
  - Set up an API POST endpoint to retrieve test cases, evaluate the solution, and return a verdict to the front end.
- *Submissions Screen:*
  - Set up an API GET endpoint to return previous submissions from the database to the front end.

4. Security Flaws and Solutions:

- *High Submission Volume:*
  - Implement a submission queue to manage many simultaneous submissions, preventing server overload.
- *Long-running Code:*
  - Implement a "Time Limit Exceeded" verdict to terminate code that runs longer than a specified duration.
- *Large Solution Files:*
  - Implement a file size limit for submissions to prevent performance slowdowns.
- *Excessive Memory Usage:*
  - Implement a "Memory Limit Exceeded" verdict to terminate code that uses more server resources than allowed.