ONLINE JUDGE(OJ) USING MERN

PROBLEM STATEMENT :

A coding challenge is a competitive event in which a group of participants solve a set of coding questions within a specified timeframe.Participants who have registered beforehand compete by submitting their solutions, which are evaluated against concealed test cases. Based on the test results, participants are assigned scores.Onine Judge is a platform where we can submit our code and it returns a verdict based upon the code provided for that problem.Applications include Leetcode,Codeforces,Codechef.

OVERVIEW :

Designing a Full Stack Online Judge Using Mern Stack.Takes code from different users over the server. Evaluates it automatically as accepted or not accepted.

FEATURES:

User Registration: Participants should be able to register for future competitions by providing their personal details such as name, email,and password.

Solution Submission: Participants should be able to submit their solutions to the problems during the competitions. They can upload their code or provide a text-based solution through the platform.

Profile Management: Participants should have access to their profile, which includes personal details and their participation history. This allows them to track their progress and view their past competition Performances.

Competition Leaderboard: Participants should be able to fetch the leaderboard of a specific competition. This leaderboard will display the rankings of participants based on their scores in that particular Competition.

Practice Problems: The platform should provide practice problems that do not contribute to the scoring or rankings. These problems allow participants to hone their skills and gain experience without the pressure of competition.

Solution Evaluation and Scoring: The platform should have a mechanism to evaluate the submitted solutions against the underlying test cases and generate scores. This evaluation process should be automated to ensure fairness and accuracy in scoring

HIGH LEVEL DESIGN :

1. **Frontend Part** [*React,Tailwind CSS,JS*]

- When user enters first time to OJ Website, a UI containing Logo,Explore button,Problems,SignIn and SignUp button should be displayed.Other contents like Explore and Get Started like features will also be implemented to the Home Page.It will be visible for all users(authenticated/non-authenticated).On clicking Problem button,they will be able to see problems but can't open any problems.All features will be view-only.

- Then for authentication(**JWT**) a separate Sign-In page will be made for authenticated user.While for non-authenticated user, they can made their account by clicking to SignUp button.Thus a separate SignIn and SignUp Page had to be made for Authentication.

- Then on SignIn/SignUp , a separate user page will be made which will contain all functionality of Home page.Additionally user information will be made available and will be authorized to use functionality that non-authenticated users can't.

- When authorized user clicks on Problem button.A separate page will be open containing list of Problems.On clicking each Problem, a separate Page will be rendered containing Problem,Code Area(Containing Code language),Input,Output,Test Cases,Submit button.

- On clicking Submit button, result will be displayed as per verdict(Accepted/Rejected) in form of designed popup.Leaderboard will be also displayed.

2. **Database Part** [*MongoDB*]

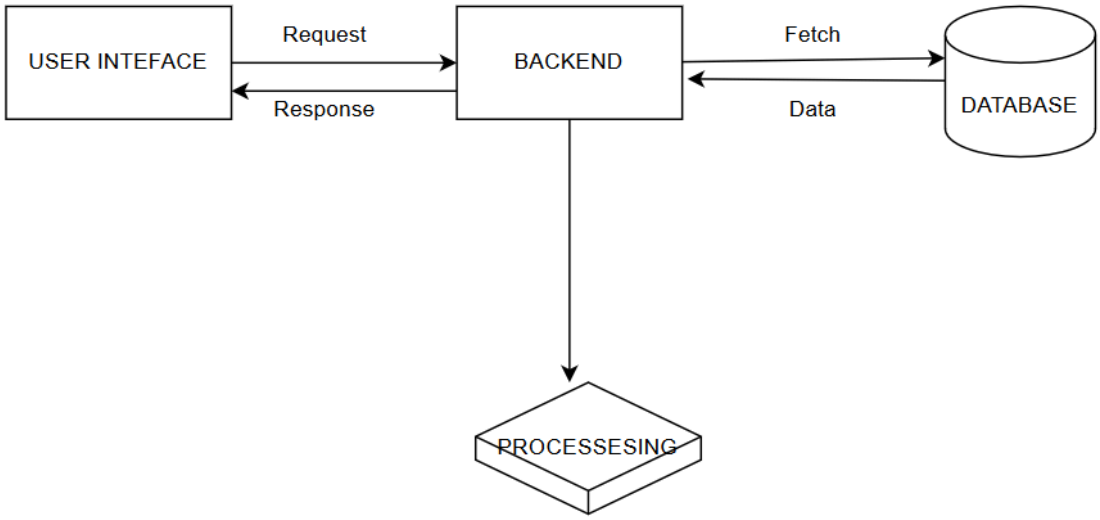
- Table 1: Problems
  Contains Pid,Pname,Pstatement,Pdifficulty.

- Table 2: Solutions
  Contains Pid(ForeignKey),verdict,submitted_at(Date&Time submitted).
- Table 3:Test Cases
  Contains input,output,Pid(ForeignKey)

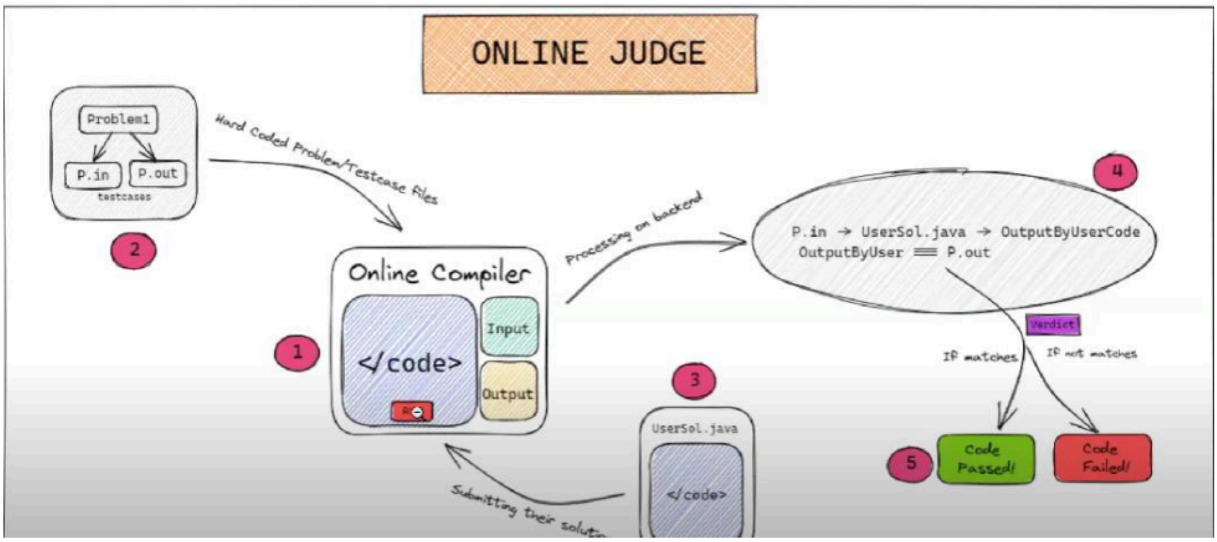
- Table 4:Login/SignUp
  Contains Userid,Password,Email,DOB,FullName.

3. **Backend Part** [*NodeJs,Express*]

- User Authentication & Authorization.
- API's to fetch data from DB.
- Also to evaluate code and save verdict.
- Docker will be used for containerization.
- All frontend and backend code will be pushed into Docker Container,Docker image will be created which will serve all dependencies for running the code.



PROCESSING IS DONE IN THIS MANNER->



ADVANTAGES & DISADVANTAGES OF TECHSTACK :

1. **MongoDB**

   Pros->
   - Schema not Required.
   - Scalability(Horizontal & Vertical)
   - Ease Of Maintenance.

   Cons->
   - Lack in ACID Properties.
   - Data Redundancy and memory Usage.
   - Limited Transactional Scope.

2. **Express**

   Pros->
   - Minimal and lightweight.
   - Middleware Support
   - Routing
   - Scalability

   Cons->
   - Lack of Structure and Convention
   - Lack of Strong Typing.
   - Limited Built-In features.

3. **React**

Pros->

- Easy to learn and use
- Creating Dynamic Web Application becomes easier.
- Reusable Components.
- Performance Enhancement
- Benefit of JS Library.

Cons->

- The high pace of Development
- JSX as a barrier.

4. **Node.js**

Pros->

- Easy Scalability
- Quick to adapt and easy to learn
- Offers long-term support for enterprises.

Cons->

- Performance reduced due to heavy computations.
- Lacks library support.

POTENTIAL RISKS :

1. Thousands Of Users submitting solutions at the same time(Thundering Herd).

2. Someone uploads a code that has a malicious event.

3. Unauthorized person gets access to manipulate the verdicts and output on the server.

SECURITY MEASURES :

1. To tackle risk1 ,we can use Rate limiting but it is generally considered as a bad practice.So, we can have a message queue in which we store the events to execute the code file at some time in the future This will be an Asynchronous Process.

2. To Tackle risk-2 , we will use Docker that basically makes containers . We can easily assign each container a set amount of memory and test whether that code

executes within that span of memory or not. It also provides a safeguard against any user trying to eat up memory with malicious code.

3. To tackle risk-3, we will be Isolating our core logic using Custom Isolation. We will implement this API using Docker.