**HLD Document → Online Judge**

**UI description→**

**Opening Sign in page**

* It will start with a basic sign in page which will give the functionality of sign up and log in for a user and enable him/her to make an account.
* We will store details of user in database

**Landing Homepage**

* Home Page showing number of problems solved and on the basis of number of problems solved the rating of the user will be given.
* Topic wise problem given in practice section.
* Submit Button and discussion section with each question.

**Leaderboard**

* A rating wise list of all the registered candidates which shows the rank of a particular candidate among all the people out there on the website.
* Also show the status when many users are using a site simultaneously showing ranking in a contest.

**Server description→**

Made a backend server using express.js to perform serving files, handling requests, and handling HTTP methods.

**DataBase description→**

**Database for login details(private)**

* Username
* Email id
* password

**Database for User Profile(displayed)**

* Username
* No of problems solved
* Rating
* Ranking on leaderboard

**Database for Problems**

* Problem ID
* Problem statement
* Sample Test Cases
* Hidden Test Cases
* Answers of test cases

**Database for Submission**

* Solution file
* Compilation verdict→compilation successful or compilation error
* Users Output
* Users verdict →TLE, MLE, passed, failed

**Backend compiling and verdict description→**

* The solution file will be saves as filename and we will be using our terminal to execute output.
* Using child process module in nodejs I have run a program through the javascript file only and store the file in output .txt
* By comparing the two files we will find our verdict using file system and path modules.
* Updating the result on rating and leaderboard.

**Security and Authentication description→**

We will be using JWH tokens for securing our database from XSS and noSQL injection and we will also be checking for any unauthorized access.

**Deployment and Scalability→**

Deploy the OJ into the cloud using AWS and for scaling we wlll use horizontal or vertical scaling and handle simultaneous many request.

**Containerisation using Docker→**

We will create a container using docker and it will contain all the requirements that will be required to run the application.