

# The Online Judge

## ( The Online Code Judge/Submission Platform )

The Online Judge is a comprehensive platform designed for seamless **DSA problem-solving** and **contest participation**. It offers robust authentication with OTP validation, a unified dashboard for different user roles, a fully **customized problems** table, and dedicated pages for problem-solving and contest-solving. The platform includes a variety of backend routes for efficient user management, problem handling, and contest participation, ensuring a smooth and engaging experience for all users.

### Main Features

1. Authentication and Authorization -> Using Next-Auth
2. Dashboard for different users (Admin, general users, problem setters)
3. Code compilation, execution, and submission.
4. Contest with leaderboard.

### Tech Stack used:

Frontend:

TypeScript, Next.js, tailwindCSS, shadCn,

Backend: TypeScript, Express.js, Node.js, JWT, Bcrypt, Zod for data validation, Docker for containerization,

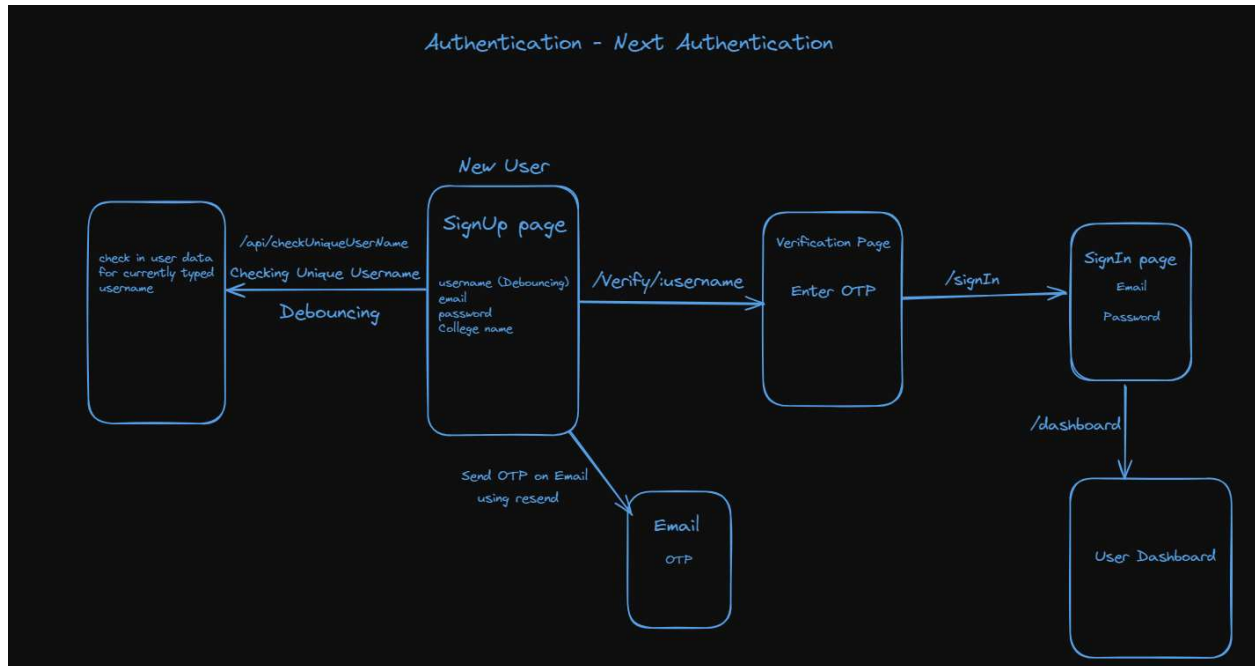
Database: MongoDB,

Models -> Users, problems, contests, problem setter applications.

Deployment: Frontend -> On Vercel (<https://oj-sigma.vercel.app/>),

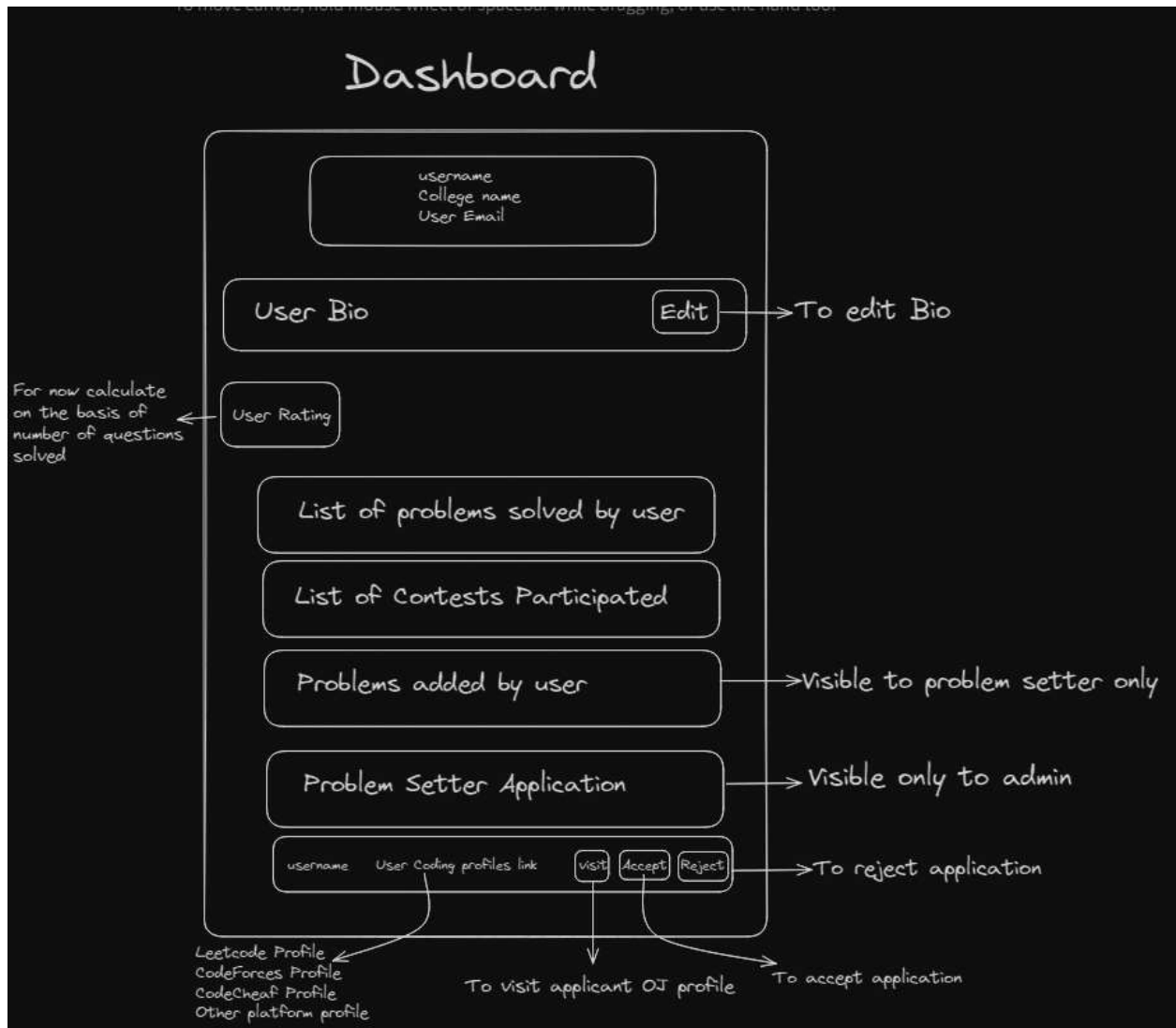
Dockerised Compiler -> On AWS,  
(<https://aws.ravikant.tech/>),

## Authentication



1. Debouncing to verify the unique username, validate email by sending OTP to user email,  
Change password. For sending emails I used -> Resend Email.
2. Routes -> /api/signup -> Create user entry on the database.  
/api/signin -> login the user.  
/api/resetPasseord -> To Change password.  
/api/verifyUniqueUsername -> checking whether a username is unique or not.  
/api/sendVerifyOTP -> sending OTP on the user email.  
/api/verifyOTP -> to verify user OTP.

## Dashboard

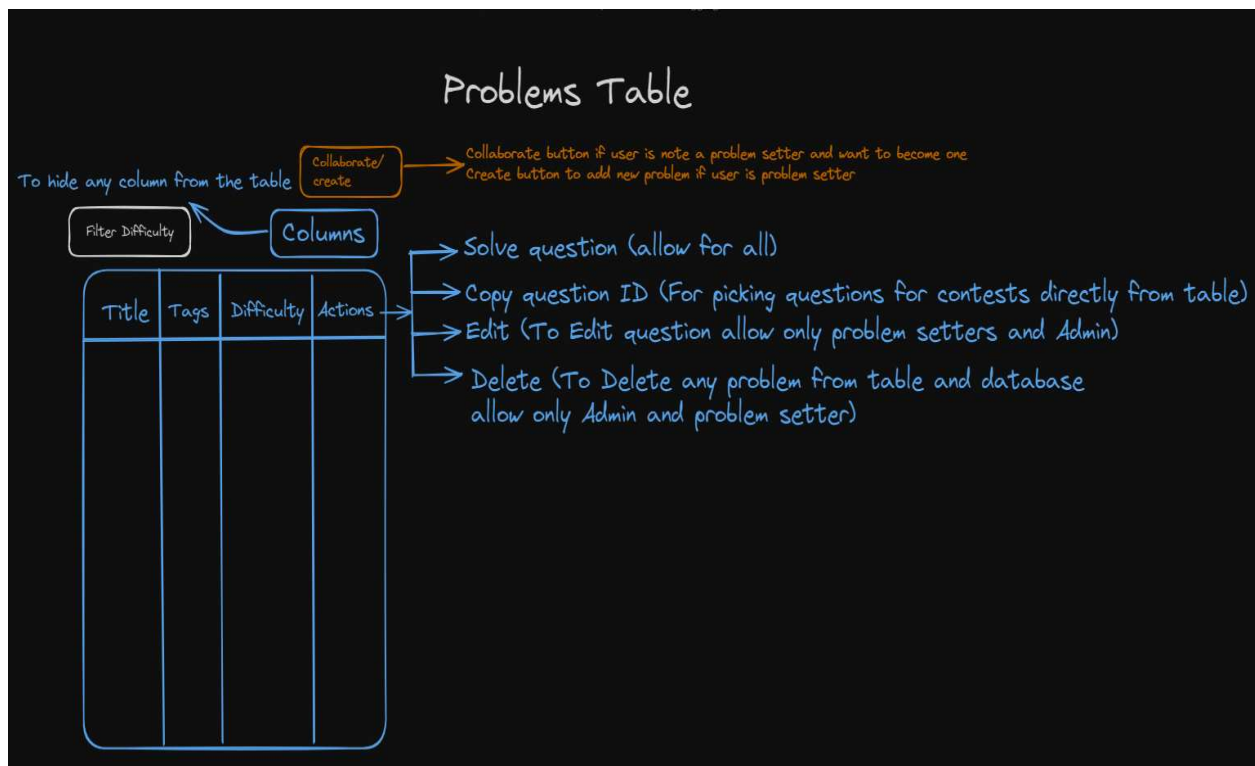


1. Single dashboard of all types of users like normal users, problem setters, and admin.
2. Routes -> `/api/updateBio` -> To save updated user bio on the database.  
`/api/problemAddedByUser/${user.username}` -> fetch problem added by user from database if user is a problem setter.

`/api/getContest/${contest.contestId}` -> fetch contests given by user.

The question solved by the user is present in the user model only.

## Problems Table



1. A fully Customised table, hides any column for better practices.

2. Routes -> `/solve/[question_ID]` -> Question-solving page.

`/api/edit/[question_ID]` -> Edit question (Allow on problem setters and admin).

`/api/delete/[question_ID]` -> Delete question (Allow only problem setters and admin).

`/problemsetterform` -> Form to become a problem setter at OJ if you are not a problem setter.

`/create` -> To add new problems if you are already a problem setter.

## Problem-Solving Page

The screenshot shows a web interface for a coding problem. The problem is titled "Minimum Window Substring". The interface includes a countdown timer at the top, a problem statement on the left, and a code editor on the right. Annotations with arrows point to various features:

- Countdown Timer (Every Student set time according to their pace)**: Points to the timer at the top.
- Copy code to clipboard**: Points to the "Copy code" button in the code editor.
- Select languages Java, C++, C, JavaScript, Python**: Points to the language selection dropdown in the code editor.
- Problem statement with Only Two Testcases rest test-cases hide from the user, Question concept name, Constraints, Question created At**: Points to the problem statement area on the left.
- This online compiler hosted on AWS server**: Points to the code editor area.
- Question Submit and added to user Solved Problems table**: Points to the "Submit" button.
- Run user code on testcases**: Points to the "Run" button.

1. To Code your approaches and execute them through the OJ Compiler.
2. Set a timer according to your speed.
3. Routes -> `/api/getProblem/[problem_ID]` -> To fetch the problem from the database.

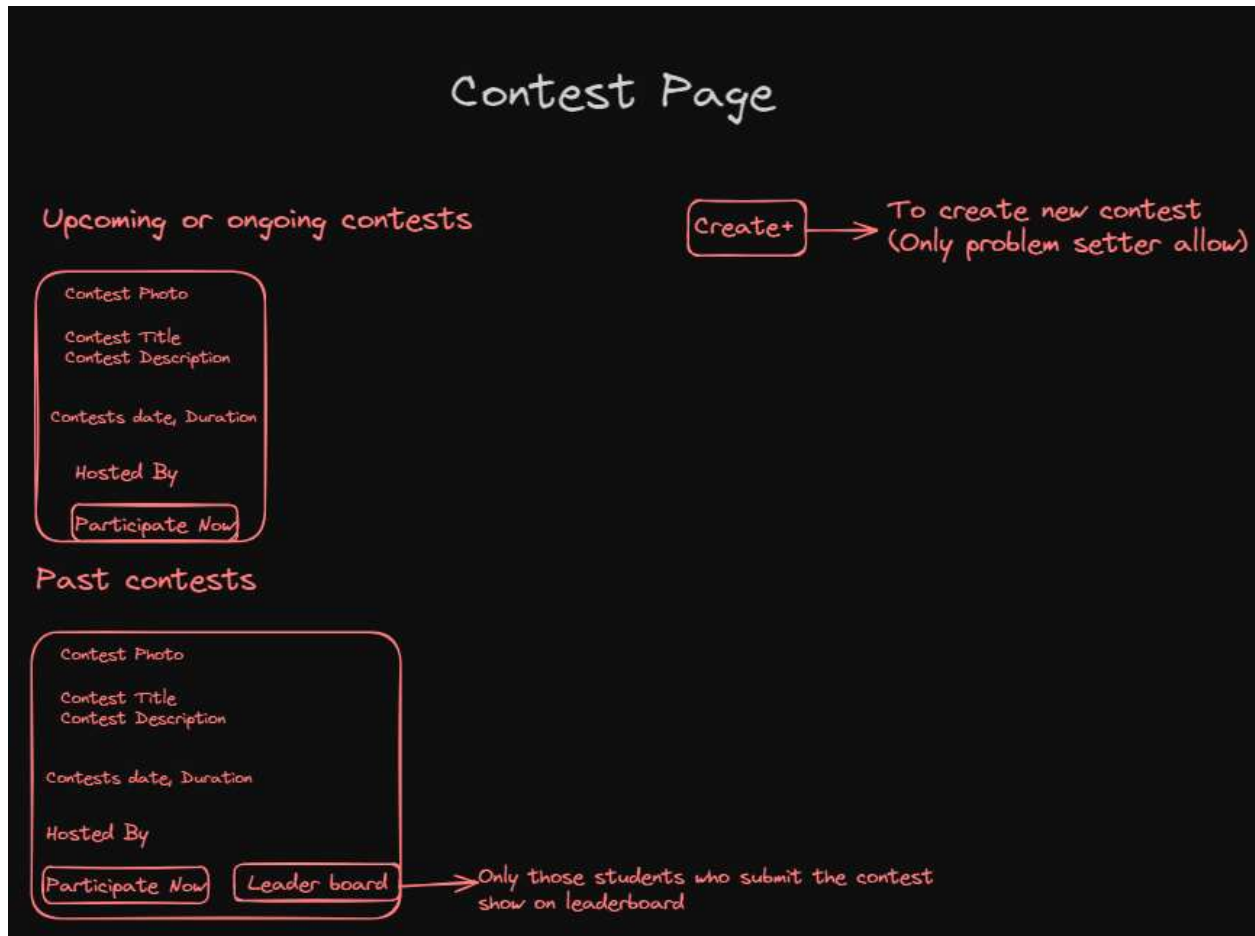
On clicking the run button

<https://aws.ravikant.tech/execute> -> To Compile user's code

On clicking the submit button

`/api/AddSubmittedProblemsToUser` -> to add the submitted problem to the user's problems solved table,

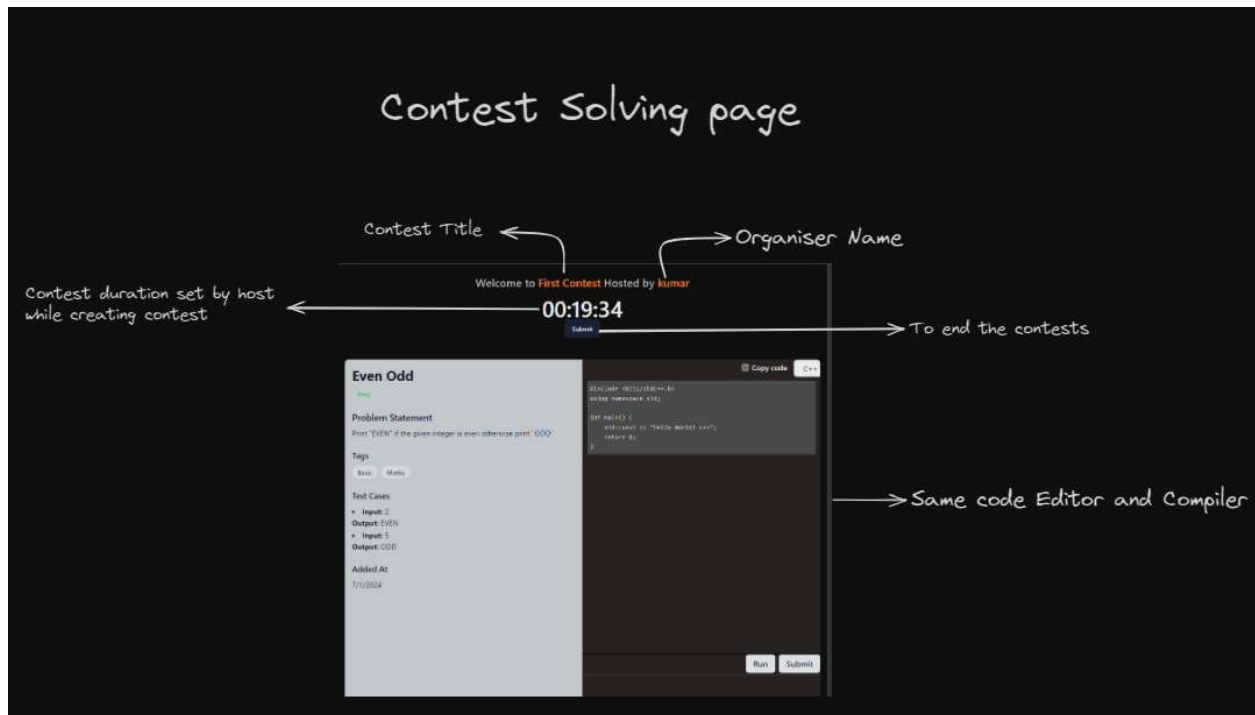
## All Contest Listing Page



1. We are listing all contests both completed, upcoming, and ongoing contests here.
2. Routes -> on clicking participate Now -> /contest/[contest\_ID] -> solving contests.  
On clicking the leaderboard we get a list of all participants with their ranks based on their marks.
3. Marks are calculated based on the number of test cases passed for a particular problem each test case contain equal marks.

4. Create button -> Only problem setter and admin can see this button to create a new contest.

## Contest Solving Page



1. The contest host sets the timer.
2. Routes -> On clicking the run button
3. <https://aws.ravikant.tech/execute> -> To Compile user's code
4. On clicking the submit button
5. `/api/AddSubmittedProblemsToUser` -> to add the submitted problem to the user's problems solved table,  
`/api/addProblemOfContestGivenByuser` -> to add the submitted problem to the user's contest solved table.
6. On clicking submit -> End Contest and save to the user contest solved table.