OJ - HLD:

Frontend : React.js mainly
Backend : Django

1.Apps:

(i)Problem:
- problemStatement (string)
- id (by default in django)
- sampleTestCase (visible to user to run and observe)
- privateTestCases (list of TestCases not visible to user)
- totalAccepted (int)
- totalRejected (int)
- difficultyLevel (Easy,Medium,Hard)
- deleteProblem (only accessible for authors and admin)

(iii)Solution:
- problem (so that each submission is mapped to the problem)
- user (whose solution is this)
- verdict (string)
- timeOfSubmission (auto dateTimeFeild)

(i)User:
- userName (string)
- id (by default in django)
- firstName,LastName (string)
- email (string)
- submissionHistory (set of their solutions)
- conestsParticipated (optional)
- regesteredContests (currently registered contests to be participated)
- Score (based on the performance in contests).

(iv)TestCase:
- problem
- Input (string)
- Output (string)

(v) Contest: (optional)
- problemSet (list of problems)
- Authors (who created this contest)
- Participants (users)
- leaderboard (refreshes every 5 min)
- evaluationOfScores (for each user)
- timer (like 2 hrs or 2.5 hrs)
- Expiration (whether the contest is over or not)

2.Features and Functions:

(i) Register and Login/Logout:
- if user is not registered, will go to the register page

users can login/logout at any time.
Using JWT for this.

(ii)ProfilePage:

For each user we will be having a profile page with his public information visible to all users.

(iii)ProblemSetPage:

Shows the list of all problems that are there in the database. Useful for practice.

(iv)ProblemPage:

Note that the submission and problem happens in the same page itself.
When a user clicks the problem the url shifts to the ProblemPage of that particular problem and in that page we will be having:

- Run,submit buttons
- input text which is basically an editable space and after pressing the restore key on it will show the default public testcase.
- After the code runs/submitted the verdict is shown within that page itself.
- Thinking of making a problem statement on the left and an IDE on the right (similar to leetcode and geeksforgeeks).
- After clicking run/submit it will be sent to the backend and there it will compile the code and check the correctness of the outputs and based on that it will return the verdict.
- In specific to the user if accepted his problem count increases in the difficulty level of that problem.
- For the problem the total accepted or total rejected increases based on the verdict to show the accuracy.

(v)contestPage:

When the contest starts only the registered participants are allowed and the leaderboard in the last second of the endtime will be considered for evaluation.

- Contest app was created so that in future if we want to improve the scoring system or add some other features for the contest it will be very comfortable to make changes.
- Contains a set of problems which will go to its specific problem page.
- scoreEvaluation for each user based on their submission.
- After the time is over the contest should be considered as expired (completed) and those problems will be included in the problemSet page as well.

(vi)PreviousContests page:

contains the list of contests which are completed showing the key details like date,name of the contest,total participation.
If the user accessing this page is a registered member then we will also show that the user has participated in this particular contest or not.

NOTE: for all of this we will just create the framework for each type of page so that it will be efficient and the rendering cost will be reduced.