Name: Aryan Shanker Saxena

DevDuelists: The modern way of coding right

DevDuelists is an online platform where players can 1up there coding skills by bringing innovative and optimized solutions to coding problems. DevDuelists also has an excellent online judge to validate your codes across various test cases. Following is the High-level design of the project as of writing this document.

Frontend (ReactJS/Django Templates):

➔ Login/Register Page:

This page ensures a valid user existing in the database gets access to the website. If no user exists, then He/She can create an account.

➔ Homepage:

This page requires user to be logged in (@login_required) which then exposes the user to problems.

➔ Online IDE to solve Problems:

This is a page where the user will have the problem statement. The user will code and validate the code against the test cases on this page.

➔ Discussions page:

This is a page where the multiple users can interact and discuss about the problems and solutions.

Backend (Django/ Django restframework):

➔ This section will consist of various API endpoints which will be created with the help of Django/restframework to manipulate the database and various other operations.

➔ URL paths will be created for navigating to static and dynamic webpages.

➔ Tentative endpoints:

- /login – for login
- /register – for registration
- /homepage – for a logged in user
- /problem<int:pk> - for a specific problem
- /discussions – for discussions

Database (PostgreSQL):

- ➔ The database will be created with the help of PostgreSQL, as this is one the fastest and most capable databases which works excellently with Django. Models in Django would be used to create the databases and Views will be used to manipulate the databases.
- ➔ Following is the minimum number of tables which will be used to establish the database:
  - User:
    - User ID (primary key)
    - Username
    - Password
    - Email
    - Phone number
    - Number of submissions
  - Problem:
    - Problem ID (primary key)
    - Problem name
    - Problem Description
    - Sample Testcases
    - Custom Testcases
    - Hidden Testcases
    - Number of accepted submissions
  - Code:
    - Code Submit ID (primary key)
    - Code
    - User ID (Foreign key)
    - Result
  - Discussions:
    - Discussion ID (primary key)
    - Discussion
    - User ID (Foreign key)
    - Number of Likes
    - Number of replies

Deployment (Docker and AWS):

➔ We will use services like Docker and AWS for deploying the website. AWS will help us use the database on cloud and docker will help us in deploying the website.

Vulnerabilities:

➔ Just like any other website, service etc. we do have potential areas of vulnerabilities:

- Modification of test cases without permissions.
- User security.
- Modification of other users' solutions and discussions.
- Modifying the code base.
- DDoS types of attacks especially from the competitors.

Solutions to mitigate the vulnerabilities and exploits:

➔ Following are the techniques and technologies used to counter the vulnerabilities:

- Containerization and orchestration techniques to avoid version incompatibilities and the website to keep it running.
- Using the right CORS headers.
- Writing protected routes.
- Limiting only necessary permissions to the user to manipulate the database such as writing a comment, coding, custom test cases etc.
- Writing safer code and API endpoints.
- Limiting only certain amount of submissions to avoid congestions.