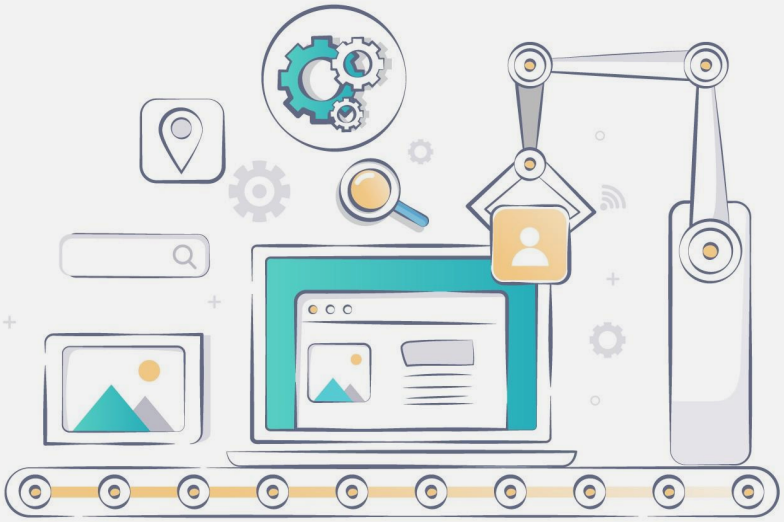
May 2024



Devankar Raj
devankarraj@gmail.com
Github
LinkedIn

High-Level Design
Online Judge

The main aim is to provide an overview of the system architecture. The task is to build an Online Judge using Django from scratch. The final product will allow users to submit their code in multiple programming languages, execute them and generate a verdict in real-time. The system will be constructed with a focus on ensuring robust security and scalability.

Main Components

1) **Frontend:** HTML, CSS and Python Templates
2) **Backend:** Django framework to handle the logic and database interactions using MVT.
3) **Database:** dbsqlite3 of django
4) **Other components**: Custom API for isolation using containerisation, Docker and other DevOps tools.

Key Functionality

1. **User Authentication and Authorization:**
    a) User registration page containing basic details
    b) User login page
        i) Username
        ii) Password
    c) Permissions based on role (e.g., regular users, admin, problem setters).
2. **Database Design:**
    a. User_id to identify which user is currently active
    b. Customized Home Page based on gender and role
    c. Problem Table:
        i. Problem Id
        ii. Problem name
        iii. Problem description
        iv. Difficulty level → Easy, Medium, Hard
        v. Input and Output (base case)

d. Testcase Table:
    i. Problem Id → to which this testcases corresponds to
        1. **Use foreign key to connect the problems and test case**
    ii. Test cases → array of actual Output corresponding to each Input
        1. Test case number
        2. Input and Output → pre-defined

e. Attempt Table:
    i. Attempt number
    ii. Problem Id
    iii. User Id
    iv. Language used
    v. Date and Time of submission
    vi. Time taken to run the code
    vii. Memory consumed by the code
    viii. Final Verdict
        1. Returns "**Accepted**" if the output from user's code matches with the pre-defined output array
        2. Make changes in the attempts database
        3. Update the leaderboard (optional)

f. Leaderboard and filters to sort the problem based on difficulty and tags (optional)

g. Hints to problem (optional)

h. Streak of solving problems (optional)

02

3. WebServer Design:

a. UI Screens Involved:
    i. Login Page
    ii. Registration Screen
    iii. Home Page (before & after)
    iv. Dashboard with list of problems & difficulty
    v. Profile (optional)
    vi. Problem Screen → Shows problem statement with Input and Output (Base Case) with difficulty of problem

vii. Submission Screen

1. list of past submissions containing username, time of submission, verdict, time taken, space occupied and language used.

b. Editor to write code along with submit & run button.

i. After clicking the submit button, feed the Input to the user's code and compare the output received with the pre-defined Output in the solution table. Generate the verdict and update the attempts table.

ii. Run Button - to test code on custom input

4. Security:

a. Custom API for isolation

b. Docker

5. Deployment on AWS

a. Using Devops tools