Prepared By:

Mohammad Enan Al Harun Sahan,

Reg No. 20101095

Section: B2

MD. Asadujjaman Noor,

Reg No. 20101101

Section: B2

Sheikh Nafez Sadnan,

Reg:20101106

Section: B2

Presented to:

Fahad Ahmed,

Lecturer, CSE, UAP.

Group ID:

B2-G5

Project Title:

Code Samlao.

Project Repository Link:

https://github.com/asadcop/cponlinejudge

Project Team Leader:

- MD. Asadujjaman Noor
  Reg No. 20101101

Project members:

- Mohammad Enan Al Harun Sahan,
  Reg No. 20101095

- Sheikh Nafez Sadnan,
  Reg:20101106

Instructor:

- Fahad Ahmed
  Lecturer,
  Department of Computer Science and Engineering,
  University of Asia Pacific, Dhaka.

Motivation:

As an adept programmer we have breezed through a lot of coding tutorial and participated in different contests. There are many popular programming challenge platforms worldwide. Such as,

- **Codeforces** [1]**:** Codeforces is an online judge sponsored by TON which, hosts contests and has sets of problems for users to solve and practice. Codeforces has a rating system to gauge the performance of a user compared to another user. This site also a dedicated community of programmers who share and discuss their knowledge and expertise on programming.

- **HackerRank** [2]**:** This is a platform that provides users with programming challenges and contests to improve their skills as a programmer. HackerRank offers a variety of programming languages, domains, and skill levels for users to choose from. HackerRank offers resources to prepare for coding interviews. It is a great website for aspiring programmers looking for a job in the field of Computer Science.

- **CodeChef** [3]**:** Codechef offers its users opportunity to participate in coding challenges, contests and hackathons. It provides a wide range of problems that are designed to improve the problem-solving skills of the users. It provides two types of membership to a user. Free and Pro. Both types of users can learn from provided contents but pro users have access to quizzes, practical projects, guided video solutions and etc.

- **Codewars** [4]**:** According to the Codewars website, it helps its users to go from beginner to expert and beyond. The opportunities Codewars provides its users are are:

    - Get new perspectives
    - Learn new languages
    - Compete with peers
    - Build self-confidence
    - Become a mentor

- **SPOJ** [5]**:** SPOJ or Sphere online judge is also a website that contains thousands of programming challenges for users to solve. It supports over 45 programming languages, and users can solve challenges to hone their coding skills as well as earn ranks and badges. It also offers a feature called "spojtoolkit" that provides various tools and libraries to help users solve coding challenges more efficiently.

Due to their popularity, hosting instant contests are difficult and sometimes applicants are put on a waitlist. We wish to create a platform that will work as a solution for this issue and provide opportunities to the people that urgently needs it.

Imagine a teacher who is need to host a contest often to evaluate his/her students' progress or increase their enthusiasm in programming. Our platform will provide them with instant and easy access whilst fulfilling the requirement.

Problem Statement:

We will be building a platform for programming contest. This platform will contain features such as:

- **Hosting contests:** Contests are collection of problems that participants can solve in order to gain points/scores. Teachers or professionals will be able to host contests in order to judge their students or colleagues.
- **Participating in programming contests:** A contest needs participant. Once any user or organization hosts a contest, selected participant will be eligible to take part in it through our website.
- **Practice programming challenges:** Users will be able to practice challenges and hone their skill without having to participate in contests. This will help refine his/her skills and boost confidence. This will encourage users to participate in contests more and sharpen own skills through competition.

Objectives:

Our objective is to provide an online platform for enthusiastic programmers to test their skills by solving challenges and participate in contests to prove their proficiency.

Project output:

Our primary target is the nurturing of aspiring programmers. Here are some of the project outputs for our project that should help them:

- An online platform
- Problem management System
    - Contest creation and management
    - Problem submission and evaluation system
    - Leaderboard system
    - Analytic system
- User Guide
    - Notification
    - User friendly UI
    - Documentation and support

Effect on Society:

There is no negative side of having a programming contest platform. Rather it has multiple positives aspects to consider. At the current age of AI and technology this website will:

- **Encourages Learning:** This is a competitive platform. To keep improving and achieve success users will have to engage themselves in more learning practices.
- **Skill development:** Learning broadens the participant's mind. This platform will the increase the pool of talented programmers available to organizations.
- **Fosters competitive spirits:** Programming contest is a competitive platform. Competing against real humans will motivate participants to perform at their best. Which as a result will lead to better products and services as developers strive to create the best possible solutions.
- **Promotes Collaboration and teamwork:** Some contests often require participants to participate as a team due the difficulty of the problems. This will help them co-ordinate with other better in their future thus increasing their potential as developers.
- **Provides networking opportunity:** Since this is an online platform it will lead to more connectivity thus increasing the opportunity of connectivity. Participants will be able to connect with each other, motivate others, help learning etc. This is beneficial to society as it can help to create more robust and connected tech community.

Requirement Analysis:

Basic Requirements:

Performance: Our target is to make a quick responsive website that will confirm the users the results within a few seconds.

Information: The information collected will be user provided. Their email will be collected to create and store data for the account. Their contests data will be collected from the code or program they submit.

Economy: Using a less responsive server for contest that doesn't need to provide immediate leaderboard might reduce the cost. Most profits will be expected to come through ad revenue or private contests. Estimated development time is 3 months.

Control: The privacy requirements for the users are just their email and passwords. Contact info. Will be collected if the private contest requires.

Efficiency: Non frequently used programming languages won't be available for users to increase efficiency in evaluation.

Service: The service will be interactive. Our target audience are programmers. There will be three types of users in this system.

1. Regular users,
2. Contest hosts and
3. Admin panel.

Admin panel will be maintaining the backend and frontend.

Functional Requirements:

An online judge for competitive programming should have the following functional requirements:

- **Authentication:** The system should have measures in place to prevent code injection, hacking, or other malicious activities. This starts with an authentication system in place. Sectors including authentication are:
    - Registration
    - Log in
    - Profile

- **User data management:** Profile will contain users all required info. History portion will contain their competitive history, solved problems list and progress in any contest they participated in.

- **Code execution and compilation:** Live updates are one the core functions for an online judge. The system should be able to compile and execute user-submitted code in real-time.

- **Input/Output handling:** The system should be able to handle both standard input and output, as well as input and output files.

- **Test case management:** The system should allow the creation, management and execution of test cases for problem submissions.

- **Problem statement and resource management:** The system should allow for the creation and management of problem statements and resources such as sample input and output files.

- **Time and memory limits:** The system should enforce time and memory limits to prevent infinite loops or excessively long computations.

- **Verdict generation:** The system should be able to generate verdicts such as "Accepted", "Wrong Answer", "Time Limit Exceeded", etc. based on the results of code execution and comparison with expected results.

- **Reporting and feedback:** The system should have mechanisms for reporting bugs, issues, and providing feedback to the users.

Technical Requirement:

We have decided to opt in agile methodology. This methodology refers to breaking down the project into small, manageable tasks and delivering working software in iterative sprints. We determined it will be well suited for our system development since it allows flexibility and encourage close collaboration between developers and stakeholders.

Here is the design pattern of the methodology we decided to follow:

Phase	**Name**	**Activity**	**Time**
Module 1	Pre-production	1. Forming the development team. 2. Discussion on the concept. 3. Deciding on a preliminary design for the project.	Week 1
Module 2	Resource Gathering	1.Gathering requirements. 2. Collecting problem sets. 3. Rearranging development tools.	Week 1-2
Module 3	User Profile	1. Creating “User Profile”. 2. Option to “Edit Profile”.	Week 2-3

		3. Option to "Set or Change User Avatar". 4. Feedback Collection.	
Module 4	Level 1 for project "Code Samlao"	1.UI Design. 2.Authorization. 3.Contest Management. 4.Problem Integration. 5.Test Case Generation. 6.Compiler Integration. 7.Alpha Testing. 8. Feedback Collection.	Week 3-5
Module 5	Level 2 for project "Code Samlao"	1. UI redesign. 2. Github repository checking. 3. Database creation. 4. Compiler Integration 5. Alpha Testing 6. Feedback Collection.	Week 4-5
Module 6	Level 3 for project "Code Samlao"	1.UI redesign 2. Database connection	Week 5-7

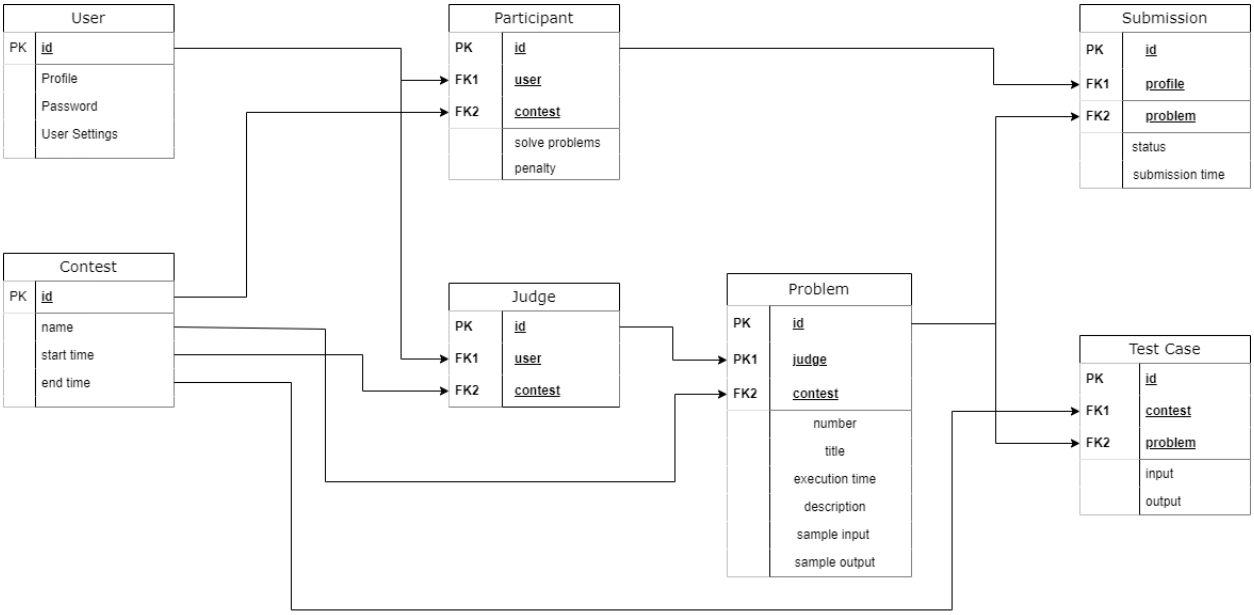
		establishment. 3.Sign up confirmation 4. Contest Management. 5. Problem Integration. 6.Scoreboard implementation. 7.Alpha Testing. 8.Feedback Collection.	
Module 7	Level 4 for project “Code Samlao”	1.UI Design. 2.Project app design. 3.Judge panel design. 4.Problem Integration 5.Test Case Generation 6.Compiler Integration 7.Alpha Testing 8.Feedback collection.	Week 7-8
Module 8	Level 5 for project “Code Samlao”	1.UI Design. 2.Database management. 3.Contest Management. 4.Managing leaderboard.	Week 8-10

		5.Compiler Integration 6.Submission verification. 7.Alpha Testing 8.Feedback Collection	
Module 9	Final Test Phase	1. Beta testing. 2. Addressing any issues and bugs. 3. Feedback Collection.	Week 10
Module 10	Product Release	1. Website Publishing. 2. Monitor Server performance. 3. Feedback Collection. 4. Address any interference.	Week 11
Module 11	Future Support	1. Implement bug fixes and new features. 2. Collect bug reports and feedbacks.	Continuous

Methodology

Here are different diagrams showing our approach to the system and its operation:

ER Diagram:



Our web system contains 7 entities.

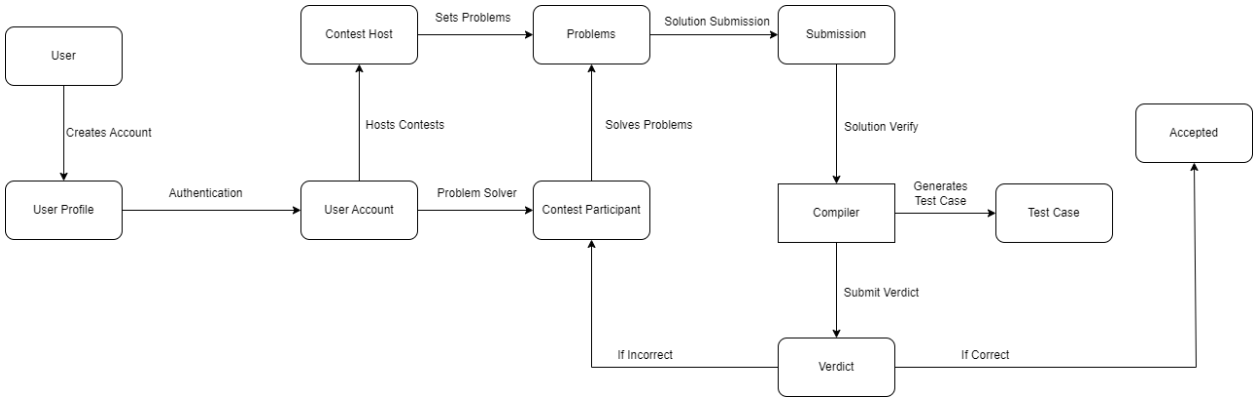
- **User:** User are the one who will interact with the system. There will be three types of users:
  - Participant
  - Judge
  - System Developers/Admin

  Each user will have a specific id that will be used as primary key in the database. They will also a profile which they will be able to

edit using User Settings. They will have to use password to secure their account.

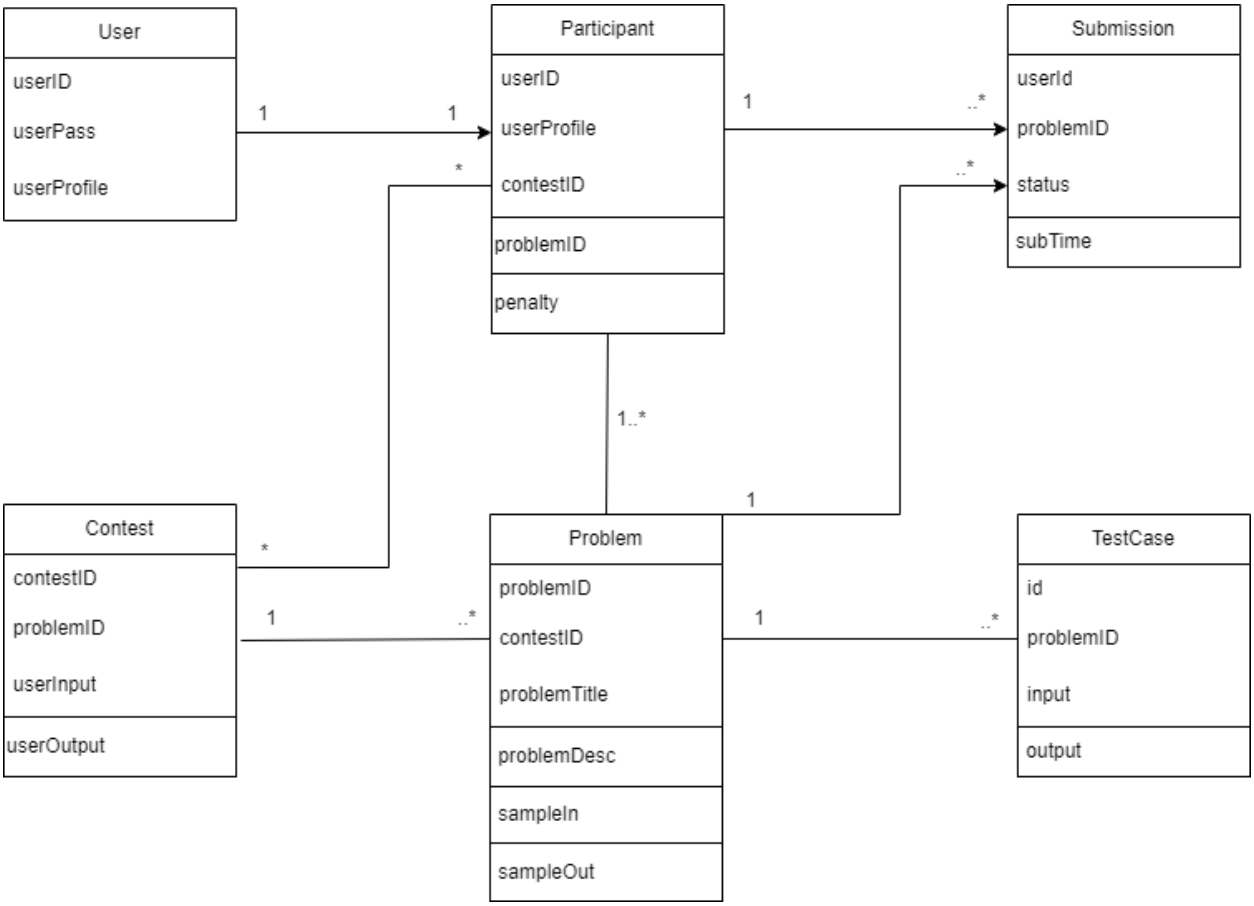
- **Participant:** Participants are a type of users. They will take part in contests and solve problem. They will also have a special id to be used as primary key. In the participant class, they will have their user id and contest id as foreign key.
- **Contest:** Contest entity will have designated id assigned to them. The id will be used as a primary key. It will also have entity such as name, start time and end time.
- **Judge:** Judges are the type of users who will be setting problems for participants to solve. They will have id as their primary key. And have user id and contest id as foreign key.
- **Problem:** Each contest will have problems assigned by judges. Here the primary keys are problem id and foreign keys are judge id and contest id. Each problem will have:
    - Specific number
    - Title
    - Execution time
    - Description
    - Sample input and
    - Sample output.
- **Test Cases:** Test Cases will have designated id as their primary keys and contest id and problem id as their foreign key. Test cases are connected to input and output.
- **Submission:** Each submission will have specific id for their primary key. Submission will have participant id and contest id as foreign key.

DFD Diagram:



This Data Flow Diagram illustrates how the interaction between and system will happen. Initially a user creates an account providing required information which generates a personal user profile. Once the authentication procedure is complete the user will have an account to himself. Contest hosts/judges will set problems which contest participants will solve. The problem solves submitted by the participants will be verified using the solution provided by the judge. Given solutions will be checked by running test cases using compiler. Then the verdict will be generated. If the verdict is incorrect the participants will be informed and if it is correct the submission will be accepted.

UML Diagram:



This diagram represents our programming contest judge web system. There are users with profile, id and pass acting as participant. A user can act as only one participant in a contest. Participants are identified specific features such as userID, userProfile, contestID, problemID. Penalty is given to the participant for failed submission. One participant can do multiple submission. Also, there can be many contests and many participants.

Participants will be solving problem which will be verified by generating TestCase. There can be one or many test cases for a single problem.

Software Process Model:

Our followed model is agile methodology. Here is how we divided our necessary tasks into sprints:

Module 1: Pre-production

➢ Sprint

- ⇨Forming a development team.
- ⇨Proposing concept for the system.
- ⇨Designing a preliminary function for the project.
- ⇨Selecting requirements.

➢ Time

- ⇨**Week 1**

Steps	Activity
Analysis	• Project process discussion. • Finding out approximate cost. • Finding out approximate time required.
Design	• Making a temporary diagram for the whole system.

Code	-
Test	-
Feedback	-

➢ Backlog

⇨Different types of diagrams containing the project plan.

Module 2: Resource Gathering

➢ Sprint

⇨Gathering requirements.
⇨Collecting problem sets.
⇨Collecting developments tools.
⇨Rearranging developments tools.

➢ Time

⇨**Week 1 – Week 2**

Steps	**Activity**
Analysis	• Figuring out tools to use.
Design	• Installing required tools. • Github repository setup.
Code	-

Test	-
Feedback	-

➢ Backlog

⇨Github repository for online workspace.
⇨Coding environment for every contributor.

Module 3: User Profile

➢ Sprint

⇨Create “User Name”.
⇨Create “Password”.
⇨Create “User Avatar”.
⇨Create “Profile”.
⇨Feedback Collection.

➢ Time

⇨**Week 2 – Week 3**

Steps	Activity
Analysis	• Contents to be in a user profile. • How will user sign in and log out interaction play out.

Design	• HTML design for create account and home page. • Opening new database.
Code	• Establishing CSS and HTML. • Connecting views.py with html.
Test	• Alpha testing done.
Feedback	• UI is too bland.

➢ Backlog

⇨Temporary overview of the whole system.

⇨Authentication system for the users.

Module 4: Level-1 of "Code Samlao"

➢ Sprint

⇨UI design
⇨Authorization
⇨Contest system design.
⇨Problem Integration
⇨Test Case Generation
⇨Compiler Integration
⇨Alpha Testing

⇨Feedback Collection.

➢ Feedback Implementation

⇨Updated CSS.
⇨Updated HTML.

➢ Time

⇨**Week 3 - Week 5**

Steps	Activity
Analysis	• Finding out all the apps to sort the project. • Separating apps by their purpose
Design	• Setting up views.py for each app. • Configuring logic in each views.py.
Code	• Installing apps in the project folder.
Test	• Alpha testing done.
Feedback	• Issue setting up the project from github.

➢ Backlog

- ⇨ Authentication system with proper UI.
- ⇨ Interactable system.

Module 5: Level-2 of “Code Samlao”

➢ Sprint

- ⇨ UI redesign
- ⇨ Github repository checking.
- ⇨ Database creation.
- ⇨ Compiler Integration
- ⇨ Alpha Testing
- ⇨ Feedback Collection.

➢ Feedback Implementation

- ⇨ Updating setup instruction in readme.

➢ Time

- ⇨ **Week 4 – Week 6.**

Steps	Activity
Analysis	• Assigning subsystem of the whole system to project members. • Github instructions unclear.
Design	• Updating CSS and HTML.
Code	• Forking projects into personal repository. • Configuring views.py in the apps folder.
Test	• Alpha testing done.
Feedback	• Database error after logging in.

➢ Backlog

⇨Platform to store data.

⇨System able to collect data.

Module 6: Level-3 of "Code Samlao"

➢ Sprint

⇨UI redesign

⇨Database connection establishment.

⇨Sign up confirmation

- ⇨Contest Management.
- ⇨Problem Integration.
- ⇨Scoreboard implementation.
- ⇨Alpha Testing.
- ⇨Feedback Collection.

➢ Feedback Implementation

- ⇨Setting up xampp.
- ⇨Setting up apache server.
- ⇨Updating database.

➢ Time

- ⇨**Week 5 – Week 7**

Steps	**Activity**
Analysis	• SQL implementation analysis. • Database connection with system.
Design	• Sign up page html update. • CSS updated.
Code	• SQL for data table changed. • Model.py connection with database.

Test	• Alpha testing done.
Feedback	• Score board isn't updating. • Contestants are not receiving points.

➢ Backlog

⇨Viewable live scoreboard.

⇨Leaderboard system.

⇨Updated UI with gradient colors.

Module 7: Level-4 of "Code Samlao"

➢ Sprint

⇨UI Design
⇨Project app design.
⇨Judge panel design.
⇨Problem Integration
⇨Test Case Generation
⇨Compiler Integration
⇨Alpha Testing
⇨Feedback collection.

➢ Feedback Implementation

⇨Reestablishing scoreboard connection.
⇨Resolving issues in model.py
⇨Updating database

➢ Time

⇨**Week 7 – Week 8**

Steps	Activity
Analysis	• Testing solution through compiler. • Discussion about problem submission.
Design	• Updating HTML and CSS.
Code	• Configuring views.py • Configuring urls.py • Configuring models.py
Test	• Alpha testing done
Feedback	• Submission Error • Password confirmation email not being sent.

➢ Backlog

⇨ UI for uploading problem.

⇨ UI for submission page.

⇨ Interactable contest system.

Module 8: Level-5 of “Code Samlao”

➢ Sprint

- ⇨UI Design
- ⇨Database management.
- ⇨Contest Management.
- ⇨Managing leaderboard.
- ⇨Compiler Integration
- ⇨Submission verification.
- ⇨Alpha Testing
- ⇨Feedback Collection

➢ Feedback Implementation

- ⇨Fixing database connection.
- ⇨Fixed verdict generation for submission.
- ⇨Updating database.
- ⇨Solving issue in app folder.

➢ Time

- ⇨ **Week 8 – Week 10**

Steps	Activity
Analysis	• Authentication system. • Problem submission verdict. • Profile management.

Design	• Updating system overview.
Code	• Merging submissions. • Changing CSS and HTML. • HTML and views.py connection. • Change in models.py.
Test	• Alpha testing done.
Feedback	• No error detected.

➢ Backlog

⇨Live verdict of submission.
⇨Live score update.
⇨Improved UI.
⇨User profile options.

Module 9: Final Test Phase

➢ Sprint

⇨Beta Testing
⇨Address any issues and bugs.
⇨Feedback Collection.

➢ Time

⇨**Week 10**

Steps	Activity
Analysis	• Contest results. • Point table update.
Design	• Updating CSS and HTML.
Code	-
Test	• Beta testing done.
Feedback	• No error detected.

➢ Backlog

⇨Final project.

Module 10: Product Release

➢ Sprint

⇨Website Publishing.
⇨Monitor Server Performance.
⇨Address bugs and reports.
⇨Feedback Collection.

➢ Feedback implementation

⇨Resolving issue in view.py logic.
⇨HTML updated.

➢ Time

⇨ **Week 11**

Steps	Activity
Analysis	• Website performance. • User interaction.
Design	-
Code	-
Test	• Beta testing done.
Feedback	• Server became overloaded.

➢ Backlog

⇨ Active website.

⇨ Active server.

Module 11: Future Support

➢ Sprint

⇨ Implement Bug Fixes.
⇨ Implement New Features.
⇨ Collect bug reports.
⇨ Feedback collection.

➢ Feedback implementation

⇨Contacting server management.

Steps	Activity
Analysis	• System performance checkup.
Design	-
Code	-
Test	-
Feedback	• Continuous.

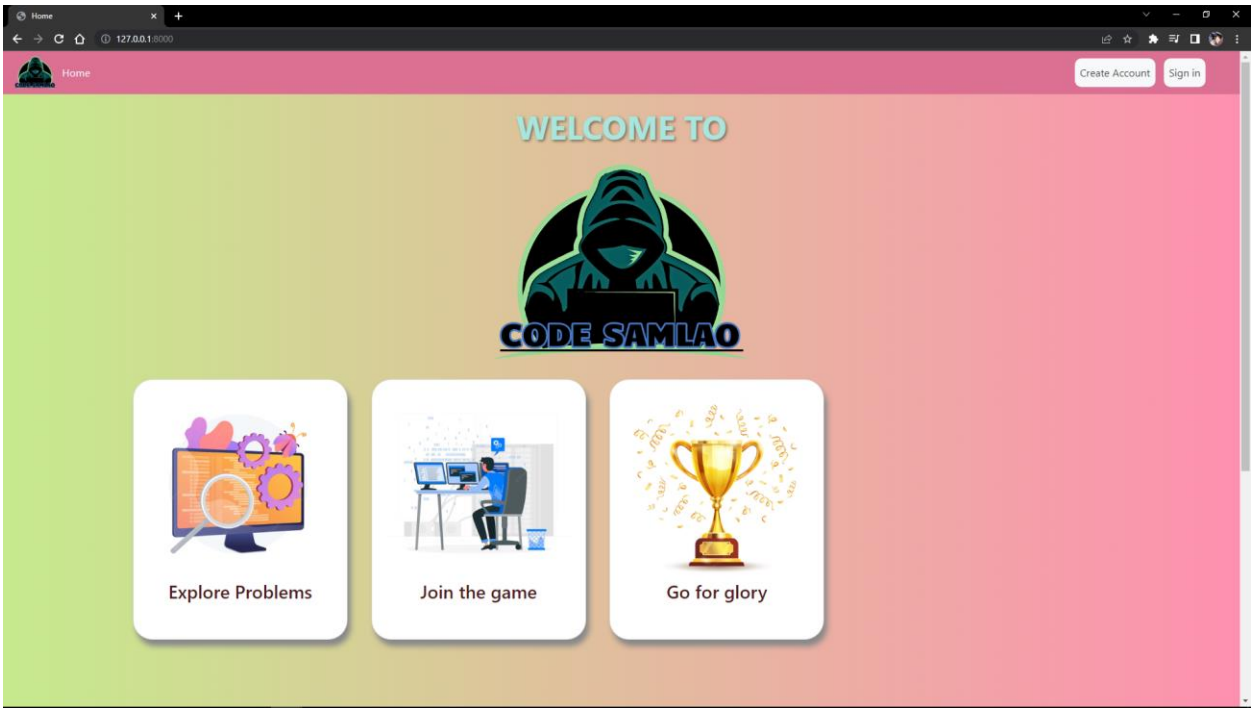
➢ Backlog

⇨Useable system.

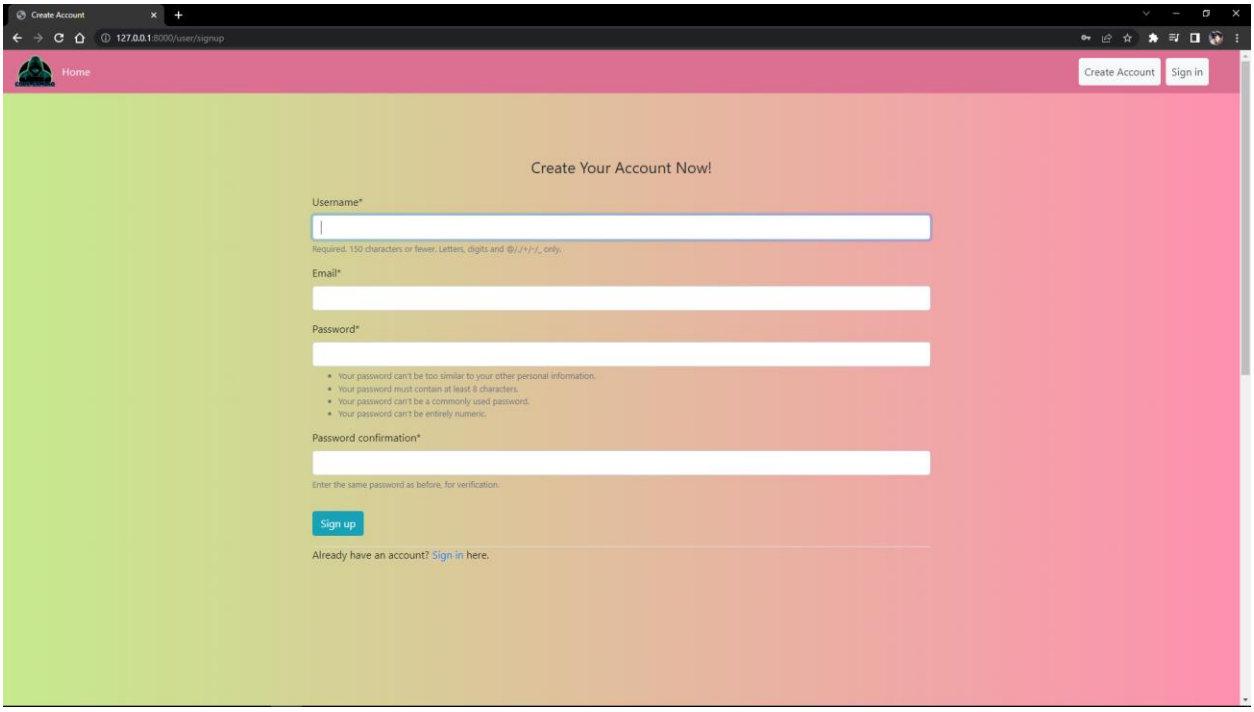
⇨Working computer programming contest judge.

Final Result of our project:

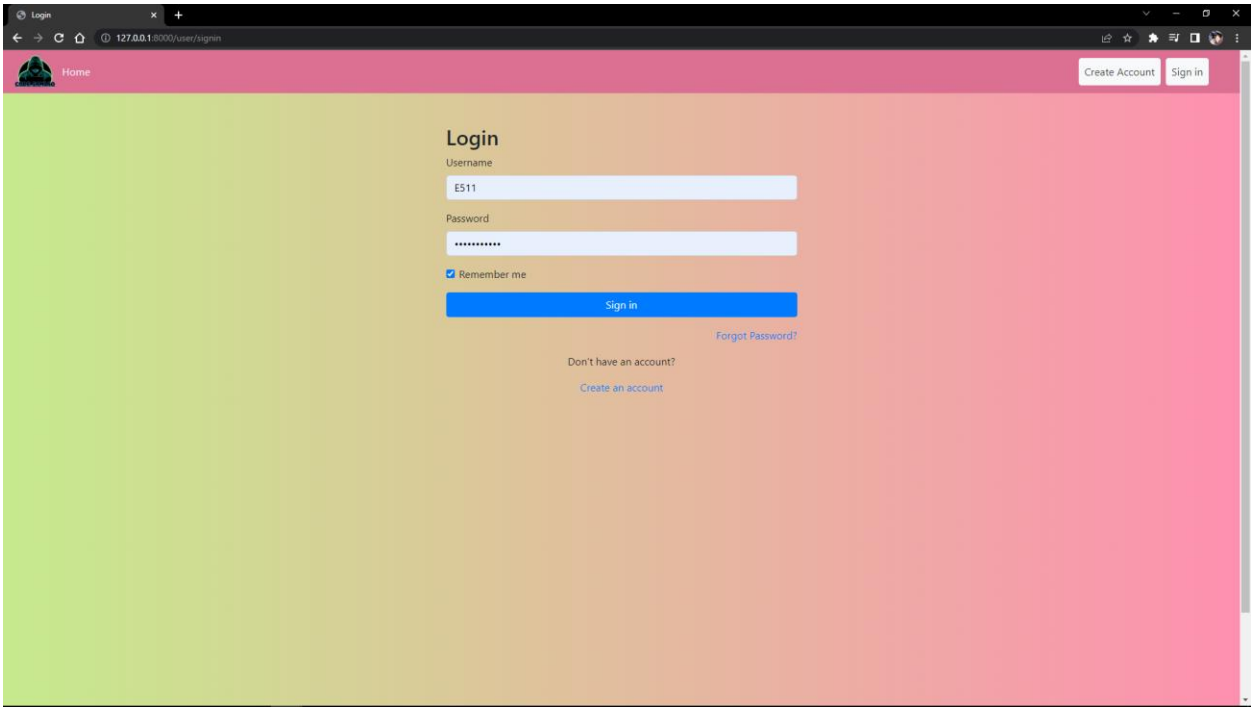
Homepage:



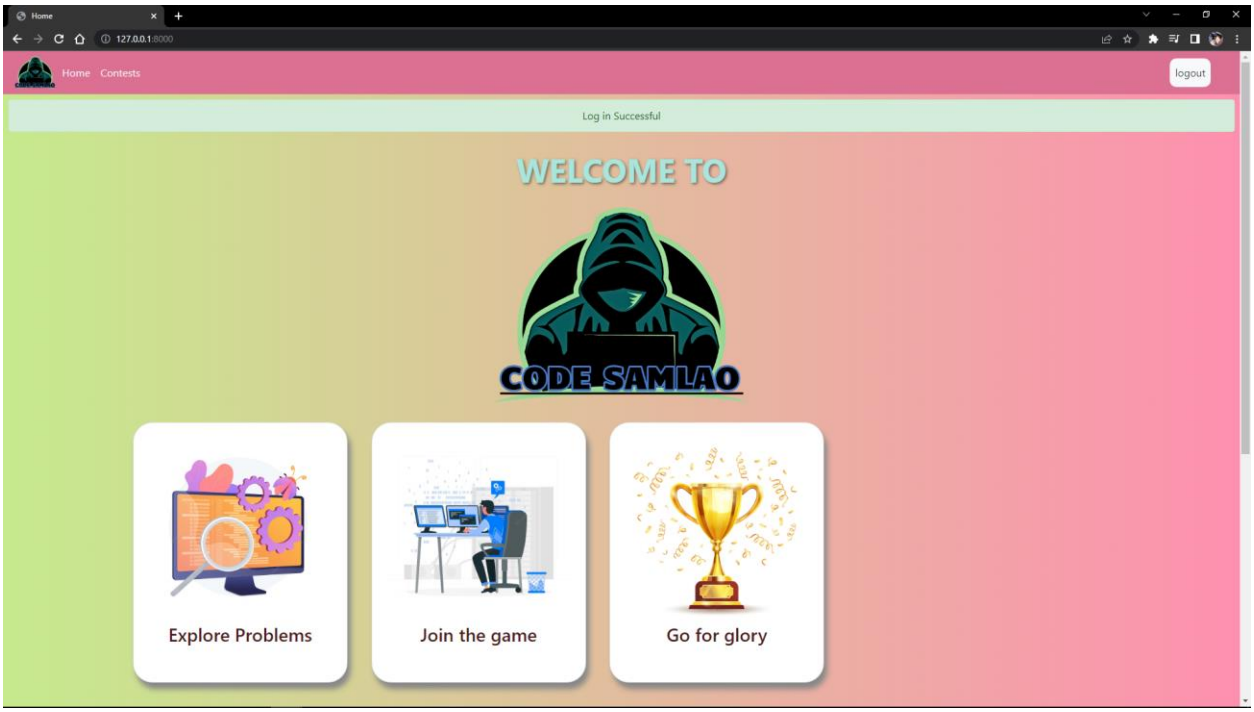
Signing up to create a new account:



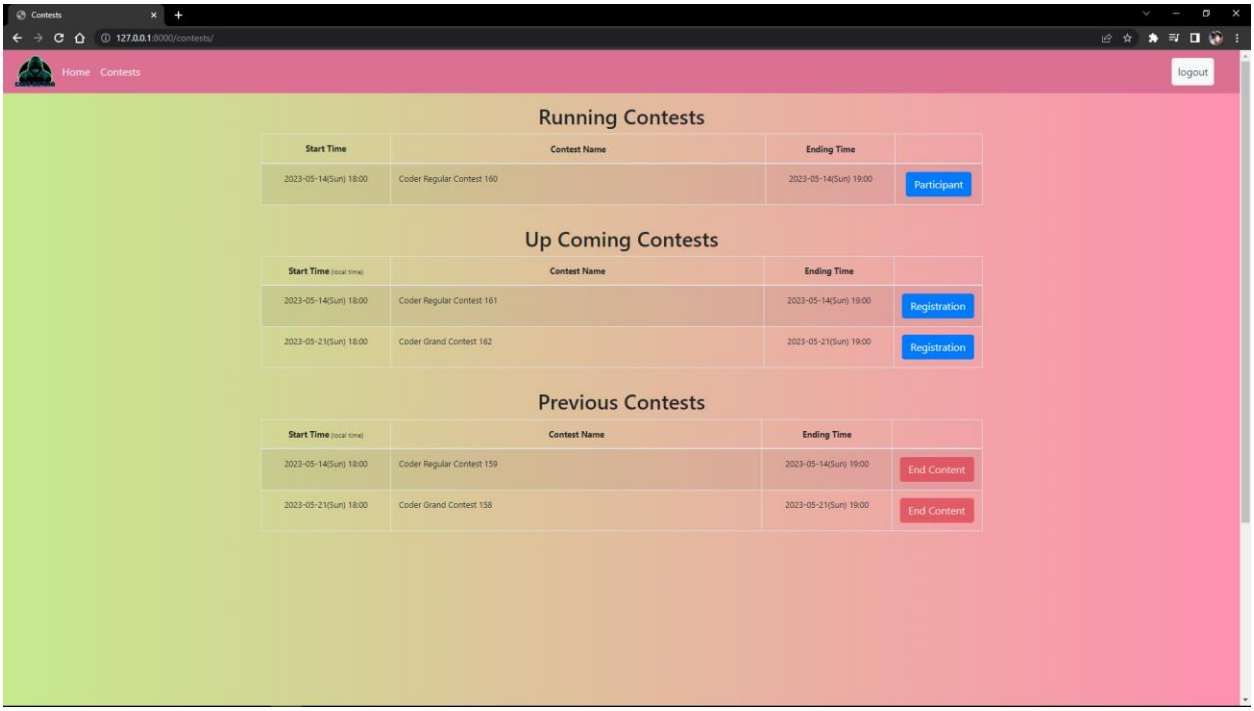
Log in with account credentials:



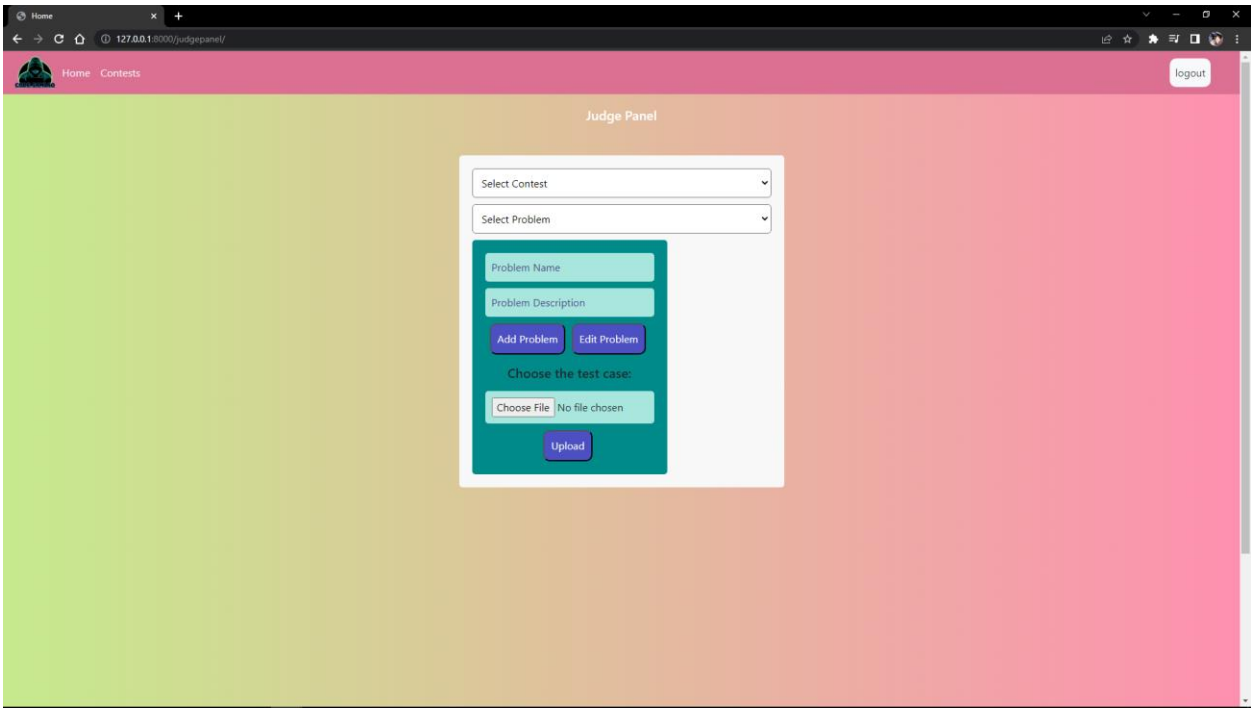
Home page after first time logging in:



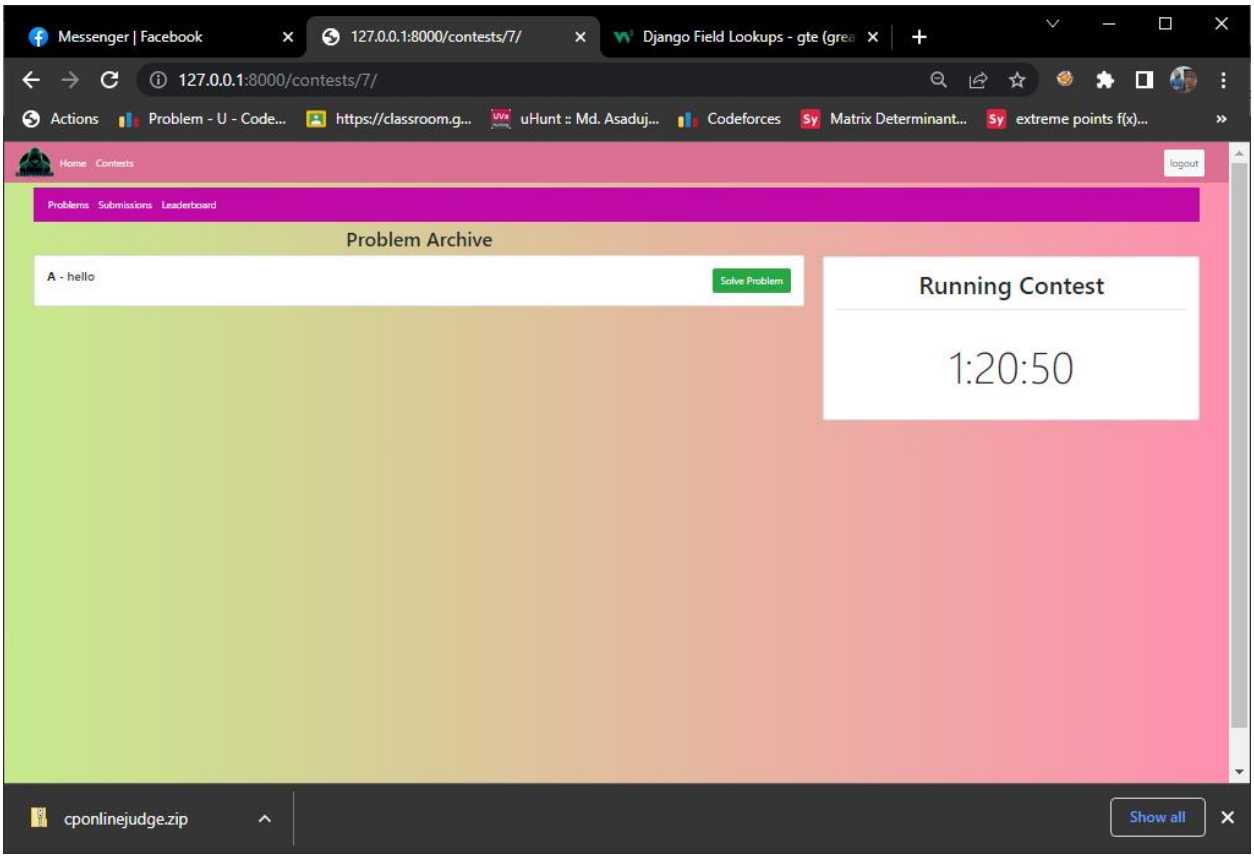
Contest page:



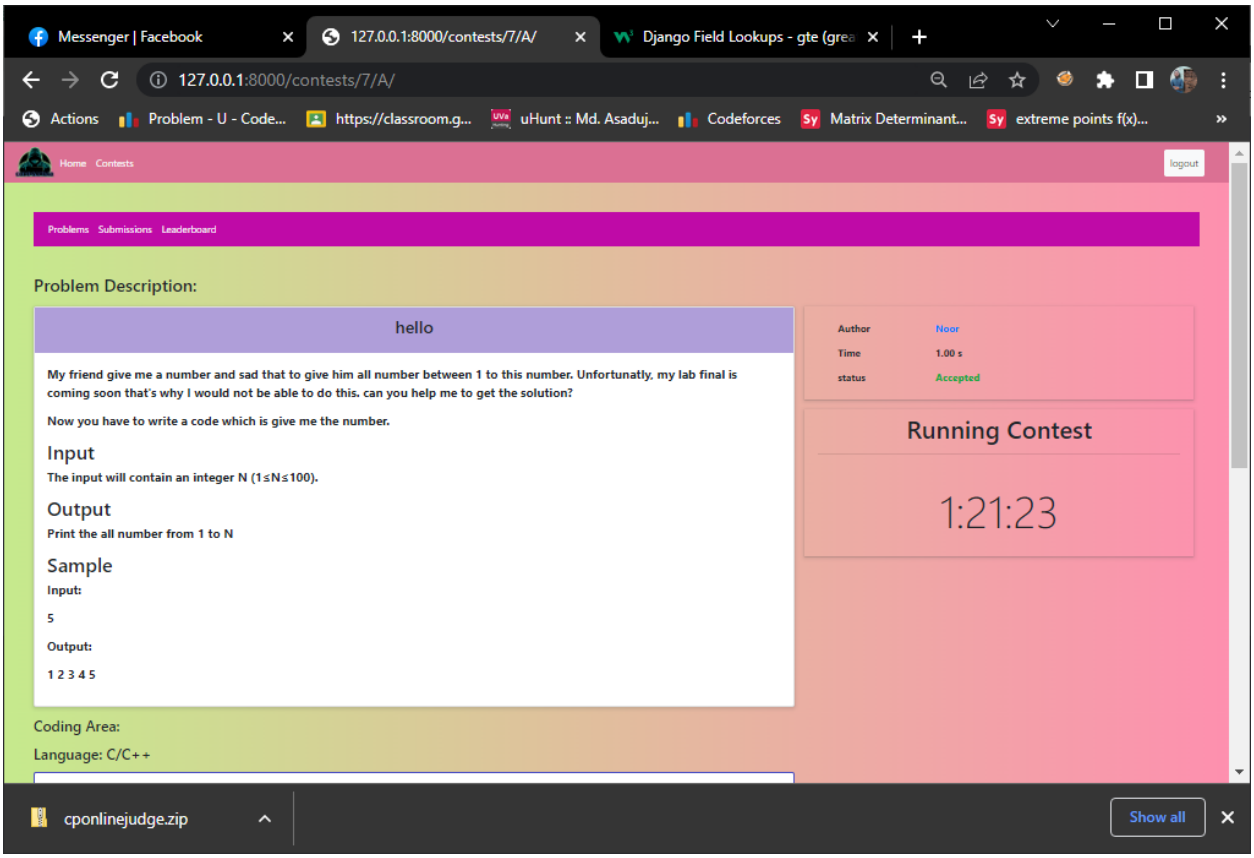
Judge panel for uploading and editing problems:



Running Contest:



Problem and Code submission page:

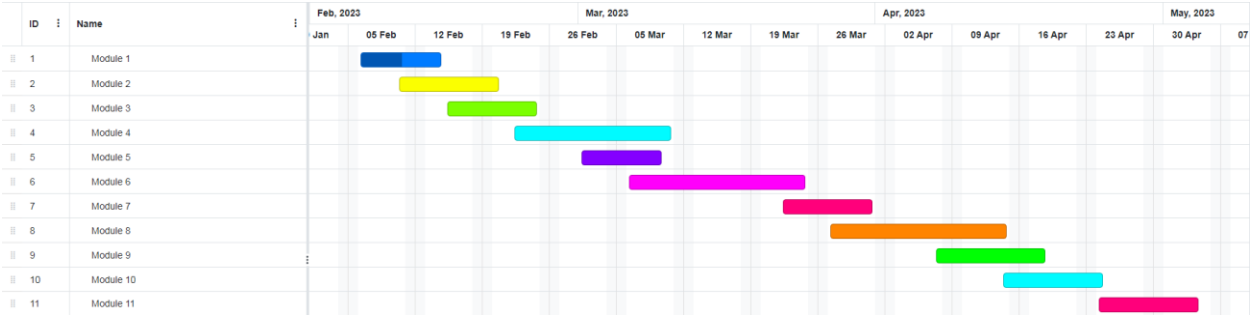


Project Management:

Project Timeline

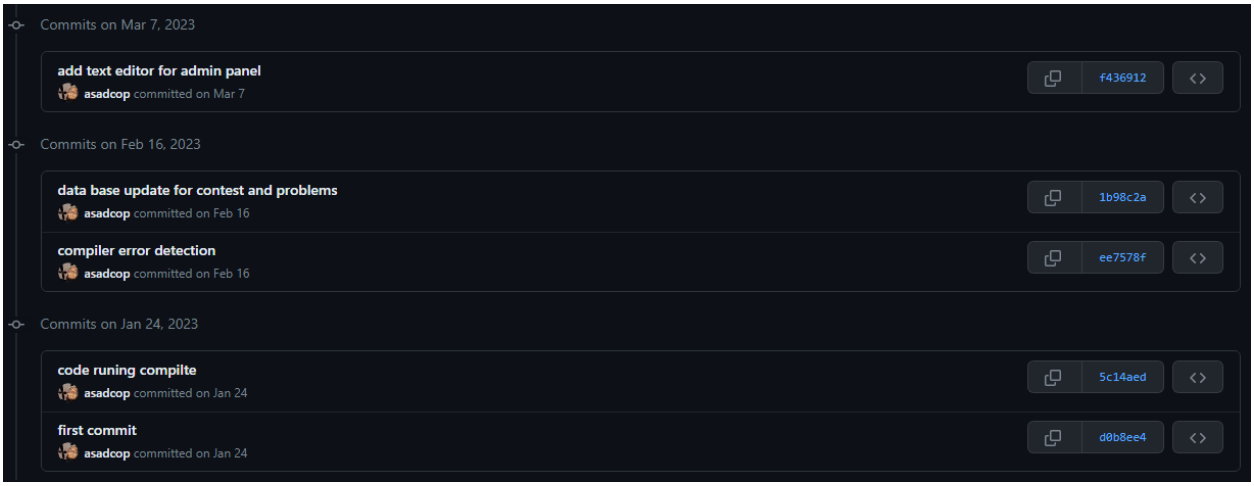
Our project started at February 05, 2023 and development ended at April 23,2023.

Each module lasted a minimum of 1 week. Some modules were 2-3 weeks long. Our tasks were divided among contributors and some modules continued parallelly. The project development lasted for 11 weeks. After the development, the continuous support module kicked in. Total work process was separated in 10 modules. And the final module 11, is for continuous future support. Here is the grant chart illustrating our project timeline.

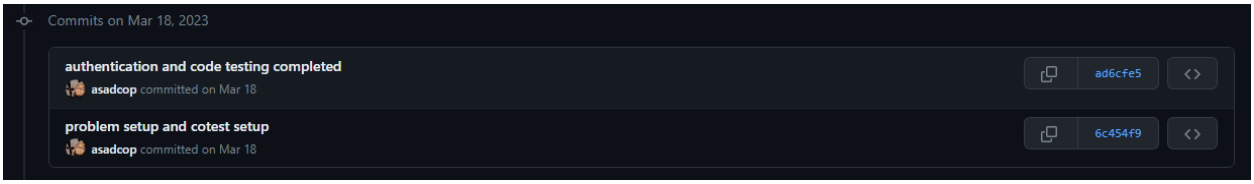


Snapshots of Version Control System Commit:

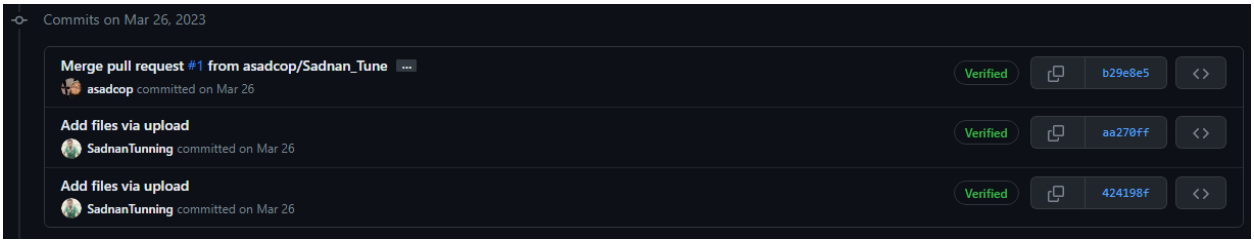
Environment Setup:



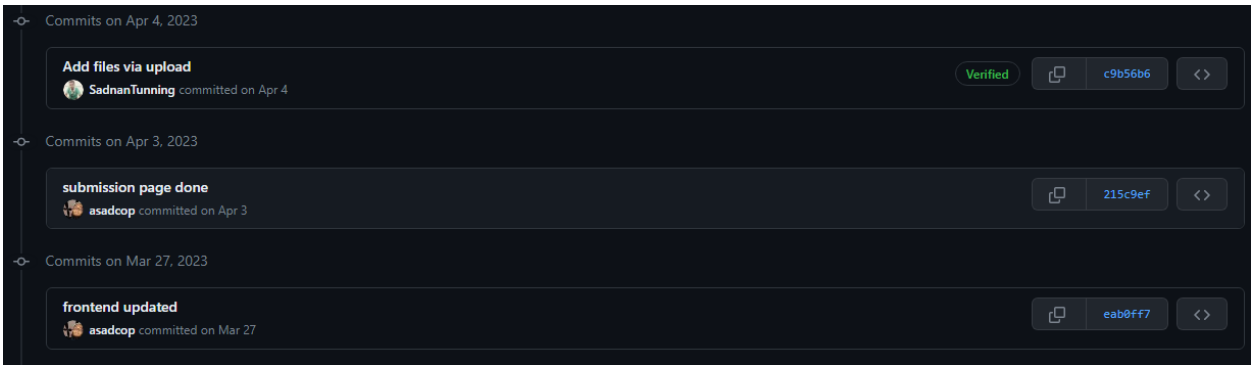
Authentication:



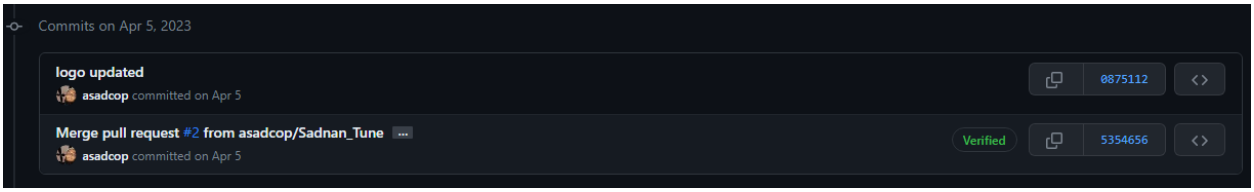
HTML update and merging:



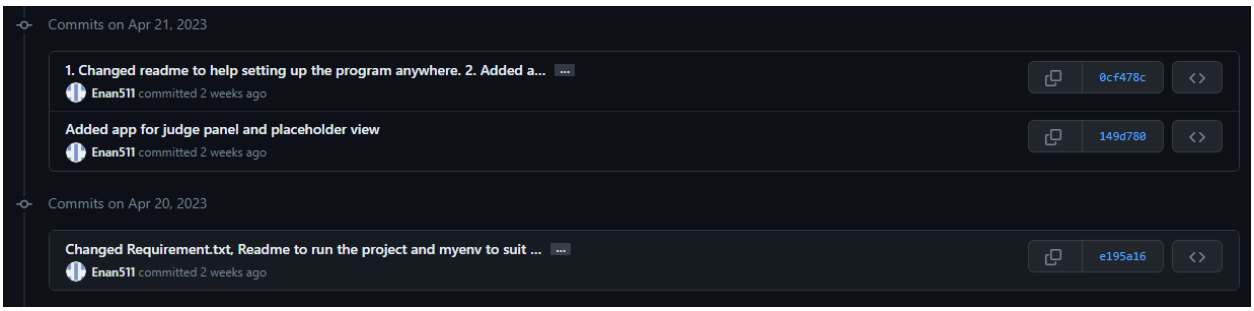
Updating Frontend:



Merging and updating logo:



Updating instructions:



Adding judge panel and merging:

Commits on Apr 26, 2023

Merge pull request #3 from Enan511/main ...
asadcop committed last week — Verified — d4d7791

Commits on Apr 25, 2023

Added judgepanel.html
Enan511 committed last week — 3295f47

Merging with main branch to update design of HTML and connect leaderboard:

Commits on Apr 30, 2023

Merge branch 'main' of https://github.com/asadcop/cponlinejudge
asadcop committed 5 days ago — 98779c3

leaderboard complete with testing
asadcop committed 5 days ago — 96de9fc

Updated HTML Gradient
Enan511 committed 5 days ago — 7585e4c

Edited welcome design in home.html and fixed a few spelling mistakes ... ...
Enan511 committed 5 days ago — 8b5f99e

Submission filter and merging:

Commits on May 4, 2023

Merge pull request #4 from Enan511/main ...
Enan511 committed 18 hours ago — Verified — 834e21e

Commits on May 1, 2023

multiple submissions filters done
asadcop committed 5 days ago — 2908f8d

Fixing CSS and automation checkup with scripting:

Commits on May 5, 2023

Merge branch 'main' of https://github.com/asadcop/cponlinejudge
asadcop committed 1 hour ago — 1a845e7

selenium and report update
asadcop committed 1 hour ago — 40ae27f

Merge pull request #5 from Enan511/main ...
Enan511 committed 11 hours ago — Verified — 60d3d1b

Fixed CSS issue in Log in page
Enan511 committed 11 hours ago — 067c0ba

final update
asadcop committed 12 hours ago — e04fabe

Scripting and Automation:

Scripting:

```
# pip install pyhtmlreport
from pyhtmlreport import Report
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time
report = Report()
driver =webdriver.Chrome()
report.setup(
report_folder=r'C:\\Users\\LS\\Downloads',
module_name='Report',
release_name='Release 1',
selenium_driver=driver
)
driver.get('http://127.0.0.1:8000/')
test_number=0
try:
#test1
    test_number=test_number+1
    report.write_step(
        'Create Account testing',
        status=report.status.Start,
        test_number=test_number
        )
    driver.find_element(By.XPATH, '//a[contains(@href,
"user/signup")]').click()
    driver.find_element(By.NAME, "username").send_keys("test")
    driver.find_element(By.NAME, "email").send_keys("test@test.com")
    driver.find_element(By.NAME, "password1").send_keys("test")
    driver.find_element(By.NAME, "password2").send_keys("test")

    time.sleep(2)
    driver.find_element(By.XPATH, "//button[text()='Sign up']").click()
    time.sleep(4)



    print("Thank You")
    # Test Steps
    report.write_step(
    'Sign up successfully',
    status=report.status.Pass,
    screenshot=True
    )
except AssertionError:
    report.write_step(
    'Fall to Sign up',
    status=report.status.Fail,
```

```
    screenshot=True
    )
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )



try:
# Test2
    test_number=test_number+1
    report.write_step(
    'Long in testing',
    status=report.status.Start,
    test_number=test_number
    )
    driver.find_element(By.XPATH, '//a[contains(@href,
"user/signin")]').click()
    driver.find_element(By.NAME, "username").send_keys("aaa")
    print("find username field\n")
    driver.find_element(By.NAME, "password").send_keys("aa")
    print("find password field\n")
    time.sleep(2)
    driver.find_element(By.XPATH, "//button[text()='Sign in']").click()
    time.sleep(4)

    assert "logout" in driver.page_source
    time.sleep(2)
    print("Thank You")
    # Test Steps
    report.write_step(
    'Long in successfully',
    status=report.status.Pass,
    screenshot=True
    )
except AssertionError:
    report.write_step(
    'Fall to long in',
    status=report.status.Fail,
    screenshot=True
    )
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )
try:
# Test3
    test_number=test_number+1
    report.write_step(
    'Long in testing',
    status=report.status.Start,
    test_number=test_number
```

```
    )
    driver.find_element(By.XPATH, '//a[contains(@href,
"user/signin")]').click()
    driver.find_element(By.NAME, "username").send_keys("test")
    print("find username field\n")
    driver.find_element(By.NAME, "password").send_keys("test")
    print("find password field\n")
    time.sleep(2)
    driver.find_element(By.XPATH, "//button[text()='Sign in']").click()
    time.sleep(4)

    assert "logout" in driver.page_source
    time.sleep(2)
    print("Thank You")
    # Test Steps
    report.write_step(
    'Long in successfully',
    status=report.status.Pass,
    screenshot=True
    )
except AssertionError:
    report.write_step(
    'Fall to long in',
    status=report.status.Fail,
    screenshot=True
    )
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )

try:
# Test4
    test_number=test_number+1
    report.write_step(
    'contest page testing',
    status=report.status.Start,
    test_number= test_number
    )
    driver.find_element(By.XPATH, '//a[contains(@href, "contests")]').click()

    time.sleep(2)
    print("Thank You")
    # Test Steps
    report.write_step(
    'switch contests page successfully',
    status=report.status.Pass,
    screenshot=True
    )
except AssertionError:
    report.write_step(
    'Fall to swich',
    status=report.status.Fail,
    screenshot=True
    )
```

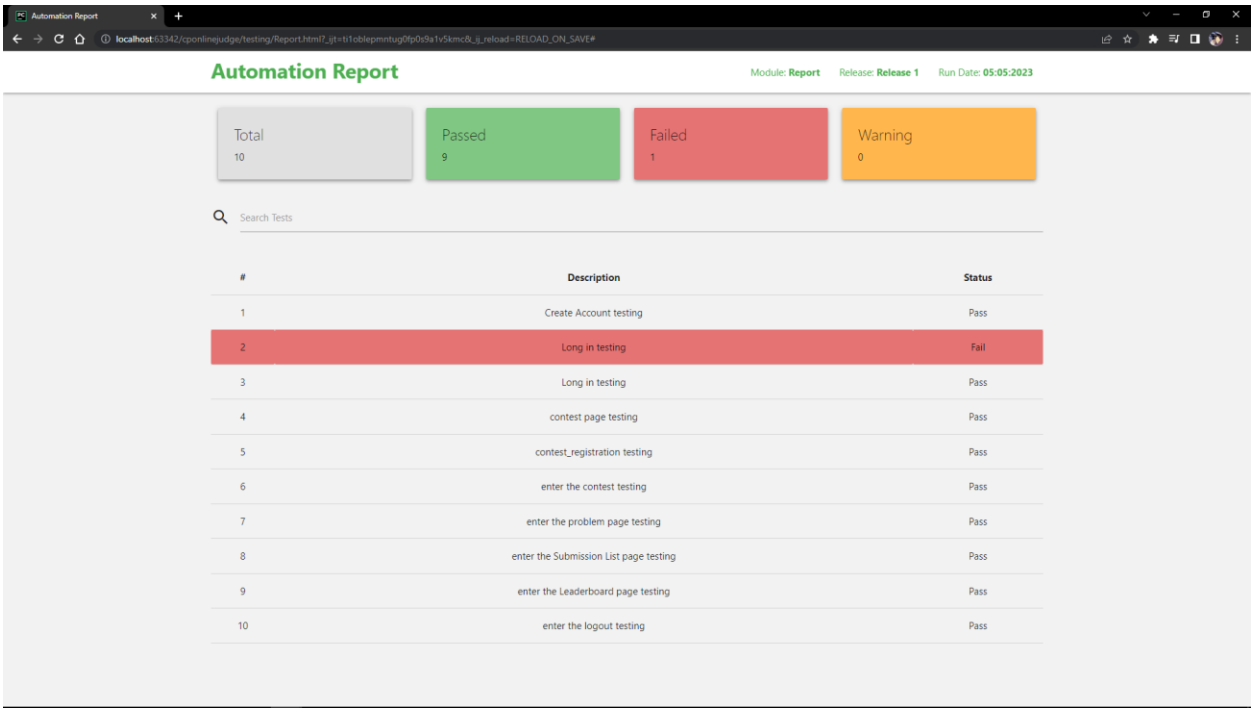
```
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )


try:
# Test5 contest_registration
    test_number=test_number+1
    report.write_step(
    'contest_registration testing',
    status=report.status.Start,
    test_number= test_number
    )
    driver.find_element(By.XPATH, '//a[contains(@href,
"contests/registration/6")]').click()

    time.sleep(2)
    print("Thank You")
    # Test Steps
    report.write_step(
    'contest_registration successfully',
    status=report.status.Pass,
    screenshot=True
    )

    driver.find_element(By.XPATH, '//a[contains(@href,
"user/signout")]').click()
    driver.find_element(By.XPATH, '//a[contains(@href,
"user/signin")]').click()
    driver.find_element(By.NAME, "username").send_keys("aaa")
    print("find username field\n")
    driver.find_element(By.NAME, "password").send_keys("aaa")
    print("find password field\n")
    time.sleep(2)
    driver.find_element(By.XPATH, "//button[text()='Sign in']").click()
    time.sleep(2)

except AssertionError:
    report.write_step(
    'Fall to contest_registration',
    status=report.status.Fail,
    screenshot=True
    )
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )


try:
# Test6 entert the contest page
    test_number=test_number+1
```

```
    report.write_step(
    'enter the contest testing',
    status=report.status.Start,
    test_number= test_number
    )
    driver.find_element(By.XPATH, '//a[contains(@href, "contests")]').click()
    time.sleep(4)
    driver.find_element(By.XPATH, '//a[contains(@href,
"contests/7/")]').click()

    time.sleep(2)
    print("Thank You")
    # Test Steps
    report.write_step(
    'enter the contest page successfully',
    status=report.status.Pass,
    screenshot=True
    )
except AssertionError:
    report.write_step(
    'Fall to enter the contest page',
    status=report.status.Fail,
    screenshot=True
    )
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )


try:
# Test7 entert the problem page
    test_number=test_number+1
    report.write_step(
    'enter the problem page testing',
    status=report.status.Start,
    test_number= test_number
    )
    driver.find_element(By.XPATH, '//a[contains(@href,
"contests/7/A")]').click()

    time.sleep(2)
    print("Thank You")
    # Test Steps
    report.write_step(
    'enter the problem page successfully',
    status=report.status.Pass,
    screenshot=True
    )
except AssertionError:
    report.write_step(
    'Fall to enter the problem page',
    status=report.status.Fail,
    screenshot=True
    )
```

```
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )

try:
# Test8 entert the Submission List page
    test_number=test_number+1
    report.write_step(
    'enter the Submission List page testing',
    status=report.status.Start,
    test_number= test_number
    )
    driver.find_element(By.XPATH, '//a[contains(@href,
"contests/7/submissions")]').click()

    time.sleep(2)
    print("Thank You")
    # Test Steps
    report.write_step(
    'enter the Submission List page successfully',
    status=report.status.Pass,
    screenshot=True
    )
except AssertionError:
    report.write_step(
    'Fall to enter the Submission List page',
    status=report.status.Fail,
    screenshot=True
    )
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )

try:
# Test9 entert the Leaderboard page
    test_number=test_number+1
    report.write_step(
    'enter the Leaderboard page testing',
    status=report.status.Start,
    test_number= test_number
    )
    driver.find_element(By.XPATH, '//a[contains(@href,
"contests/7/leaderboard")]').click()

    time.sleep(2)
    print("Thank You")
    # Test Steps
    report.write_step(
    'enter the Leaderboard page successfully',
    status=report.status.Pass,
    screenshot=True
```

```
    )
except AssertionError:
    report.write_step(
    'Fall to enter the Leaderboard page',
    status=report.status.Fail,
    screenshot=True
    )
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )

try:
# Test10 logout
    test_number=test_number+1
    report.write_step(
    'enter the logout testing',
    status=report.status.Start,
    test_number= test_number
    )
    driver.find_element(By.XPATH, '//a[contains(@href,
"user/signout")]').click()


    time.sleep(2)
    print("Thank You")
    # Test Steps
    report.write_step(
    'logout successfully',
    status=report.status.Pass,
    screenshot=True
    )
except AssertionError:
    report.write_step(
    'Fall to logout',
    status=report.status.Fail,
    screenshot=True
    )
except Exception as e:
    report.write_step(
    f'Something went wrong during execution!</br>{e}',
    status=report.status.Warn,
    screenshot=True
    )




finally:
    report.generate_report()
    driver.quit()
```

Automation Report:



Automation Report

Module: **Report** Release: **Release 1** Run Date: **05:05:2023**

Total	Passed	Failed	Warning
10	9	1	0

Search Tests

#	Description	Status
1	Create Account testing	Pass
2	Long in testing	Fail
3	Long in testing	Pass
4	contest page testing	Pass
5	contest_registration testing	Pass
6	enter the contest testing	Pass
7	enter the problem page testing	Pass
8	enter the Submission List page testing	Pass
9	enter the Leaderboard page testing	Pass
10	enter the logout testing	Pass

Finance Management:

Here is a table illustrating the cost required for the project.

Activity	**Time Cost**	**Estimated expense in Taka**
Information gathering	160 Hours	50000
Pre-production setup	170 Hours	50000
Setting up user experience	250 Hours	100000
Database setup	270 Hours	50000
Server management	Project lifetime	200000+
UI development	320 Hours	125000
Documentation	50 Hours	25000
Approximate Project Costs	1220 and more	600000

Project Development Resource:

This includes the list of developing resources used in the system development.

- ❖ Django Documentation.
- ❖ MySQL API.
- ❖ TinyMCE API.
- ❖ “Multiprocessing” Package for python.
- ❖ “Subprocess” module for python.

This is a sample for our compiler that takes submitted codes and generates verdict.

```
from django.shortcuts import render,HttpResponse
import subprocess,os
from django.contrib.auth.decorators import login_required
from django.conf import settings
import multiprocessing as multi
import time
import threading
from contest.models import TestCase
# Create your views here.
```

```
def codecheck(send,input,file_name):

    Input=input
    p = subprocess.Popen([file_name], stdin=subprocess.PIPE,
stdout=subprocess.PIPE)

    stdout, _ = p.communicate(input=Input)
    send.send(stdout)



@login_required
def runcode(request):

    if request.method=='POST':
        problemid=request.POST['pk']
        code=request.POST['code']
        file_name=os.path.join(settings.MEDIA_ROOT_CODE,
request.user.username)
        file=str(file_name+".cpp")
        #insert the code into to file
        with open(file, "w") as f:
            f.write("".join(code))

        #get testcase
        test_casees=TestCase.objects.filter(problems=problemid)
```

```
        try:
            submitionstate=""
            process=subprocess.Popen(['g++',file],stderr=subprocess.PIPE)
            _,stderr=process.communicate()
            if "error" in stderr.decode():
                submitionstate="compilar error"
                print("compilar error")
            else:
                subprocess.run(["g++", "-o",file_name , file])

                recv, send = multi.Pipe(False)
                for testcase in test_casees:

                    output=testcase.output.read()
                    input=testcase.input.read()

                    p = threading.Thread(target=codecheck,
args=(send,input,file_name))
                    p.start()
                    runtime=float(testcase.problems.execution_time)
                    time.sleep(runtime)
                    ret = recv.poll()
                    if ret == False:
                        submitionstate='TLE'
                        print("TLE")
                        p.kill()
```

```
                break
            else :
                recive=recv.recv()
                if recive!=output:
                    print(recive)
                    submitionstate="Wrong"
                    print("Wrong")
                    break
        else:
            submitionstate="Accepted"
            print("Accepted")
        p.close()
        send.close()
        recv.close()
    except:
        None
return render(request, "submition.html",{'submitionstate':submitionstate})
```

Conclusion and Future Learning:

Participating in this project helped me learn and lot. It has taught us:

- The necessity of co-operation.
- The importance of communication between contributors.
- Discipline.
- Importance of time management.
- Connection between programming languages.

This project assisted us in improving our proficiency. We learned different tactics and implementation of those in the field of programming. Our project influenced us to get in touch with more advanced development tools and also improved our efficiency in solving problems. We learned how important it is important to keep up with the latest developments in the programming world by staying up-to-date with new programming languages, frameworks, and tools. By continuously learning and challenging ourselves, we can improve our competitive programming skills and become a more well-rounded programmer.

To conclude, this has been a valuable lesson for all of us. We got the opportunity to increase our depth of knowledge and experience. Our project will immensely programmers to improve their coding skills and develop their problem-solving capability. Our project features an interactive user-friendly interface which will attract more users. Realtime scoring and live update will assist aspiring programmers by keeping them on their toes.

References

Appendix A

CEP Mapping

How Ks are addressed through the project and mapping among Ks, COs and POs

Ks	Attribute	How Ks are addressed through the project	COs	POs
K1	Complex Engineering	VS Code, Pycharm, Python Libraries for Web Development	CO1, CO2	PO-(a), PO-(b)
K3	Engineering Discipline	Github, Grant Chart, Codeblocks, HTML viewer, Browser	CO1, CO2	PO-(a), PO-(b),
K4	Specialist Knowledge	Figma for HTML UI/UX Design	CO1, CO2	PO-(a), PO-(b)
K5	Engineering Design	Python, HTML, Bootstrap, JavaScript, DB Browser	CO3	PO-(c)
K6	Engineering Practice	VS Code, Pycharm, Github	CO9	PO-(k)
K8	Research Literature	The Programming Journal	CO5	PO-(d)

How Ps are addressed through the project and mapping among Ps, COs and POs

Ps	**Attribute**	**How Ps are addressed through the project**	**COs**	**POs**
P1	Depth of knowledge required	This system requires mathematical knowledge, engineering fundamentals to set question for contestants to solve (K2). Identify, formulate and analyze complex engineering problems to reach conclusion (K3). Problem will be set by professionals (K4).	CO1, CO2, CO5	PO-(a), PO-(b), PO-(d)
P2	Range of conflicting requirements	The server needs to be fast and responsive. Provide the result, point out the error and update the leader board almost instant. But in test or contests some time the result will only show if it can be accepted and update the leaderboard after the contest is over.	CO6, CO7	PO-(c), PO-(g)
P4	Familiarity of issues	Contest with lot of participants can slow	CO7, CO8	PO-(i), PO-(j)

		down the server decreasing the server performance.		
P7	Interdependence	Interdependency subsystem like • Data Collection • Problem organization • Generate results • Working IDE	CO4, CO9	PO-(e), PO-(k)

How As are addressed through the project

As	**Attribute**	**How As are addressed through the project**	**COs**	**POs**
A1	Range of Resources	This project needs database management, Fast and Responsive server, Backend technology, Problem setters	CO8	PO-(j)
A2	Level of interaction	It requires constant interaction between server and the user to fulfill its purpose. Needs to provide fast output and update the leaderboard real time. Also hide/freeze the leaderboard upon the hosts wish.	CO8	PO-(j)
A4	Consequences for society and the environment	Will help to nurture aspiring programmers. Improving their skill and increase the availability of proficient developers.	CO8	PO-(j)

References

[1] "Codeforces," [Online]. Available: https://codeforces.com/.

[2] "HackerRank," [Online]. Available: https://www.hackerrank.com/.

[3] "CodeChef," [Online]. Available: https://www.codechef.com/.

[4] "Codewars," [Online]. Available: https://www.codewars.com/.

[5] "SPOJ," [Online]. Available: https://www.spoj.com/.