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:

Savant Coders.

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

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
- HackerRank
- CodeChef
- Codewars
- SPOJ

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 the 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
- Participating in programming contests
- Practice programming challenges

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.	8 Days

Module 2	Resource Gathering	1.Gathering requirements. 2. Collecting problem sets. 3. Rearranging development tools.	15 Days
Module 3	Authentication	1. User Registration. 2. Sign in. 3. Log out.	10 Days
Module 4	User Profile	1. Creating "User Profile". 2. Option to "Edit Profile". 3. Option to "Set or Change User Avatar".	10 Days
Module 5	Level 1 for project "Savant Coders"	1.UI Design 2.Authorization 3.Contest Management 4.Problem Integration 5.Test Case Generation 6.Compiler Integration 7.Alpha Testing	15 Days
Module 6	Level 2 for project "Savant Coders"	1.UI Design 2.Authorization 3.Contest Management	15 Days

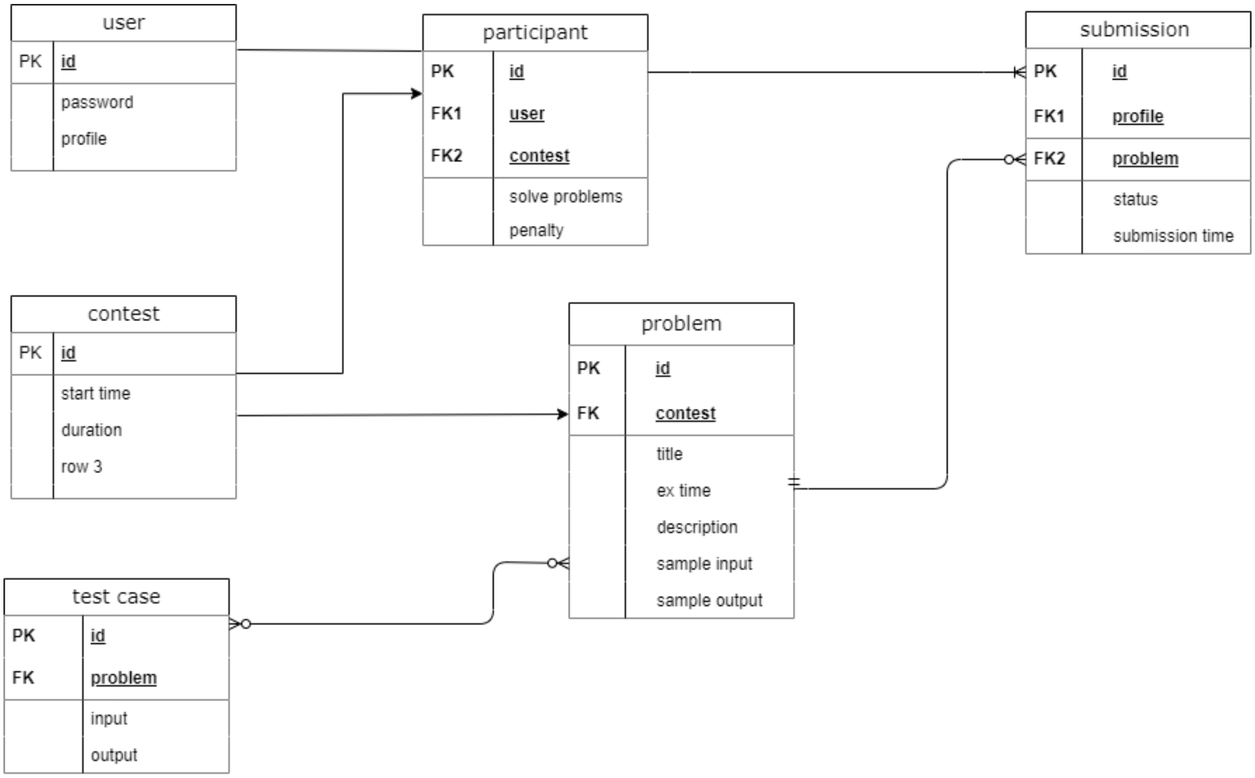
		4.Problem Integration 5.Test Case Generation 6.Compiler Integration 7.Alpha Testing	
Module 7	Level 3 for project “Savant Coders”	1.UI Design 2.Authorization 3.Contest Management 4.Problem Integration 5.Test Case Generation 6.Compiler Integration 7.Alpha Testing	15 Days
Module 8	Level 4 for project “Savant Coders”	1.UI Design 2.Authorization 3.Contest Management 4.Problem Integration 5.Test Case Generation 6.Compiler Integration 7.Alpha Testing	15 Days
Module 9	Level 5 for project “Savant Coders”	1.UI Design 2.Authorization 3.Contest Management 4.Problem Integration	15 Days

		5.Test Case Generation 6.Compiler Integration 7.Alpha Testing	
Module 10	Final Test Phase	1. Beta testing. 2. Bug Fixing.	8 Days
Module 11	Product Release	1. Website Publishing. 2. Monitor Server performance. 3. Address any interference.	7 Days
Module 12	Future Support	1. Implement bug fixes and new features. 2. Collect bug reports.	Continuous

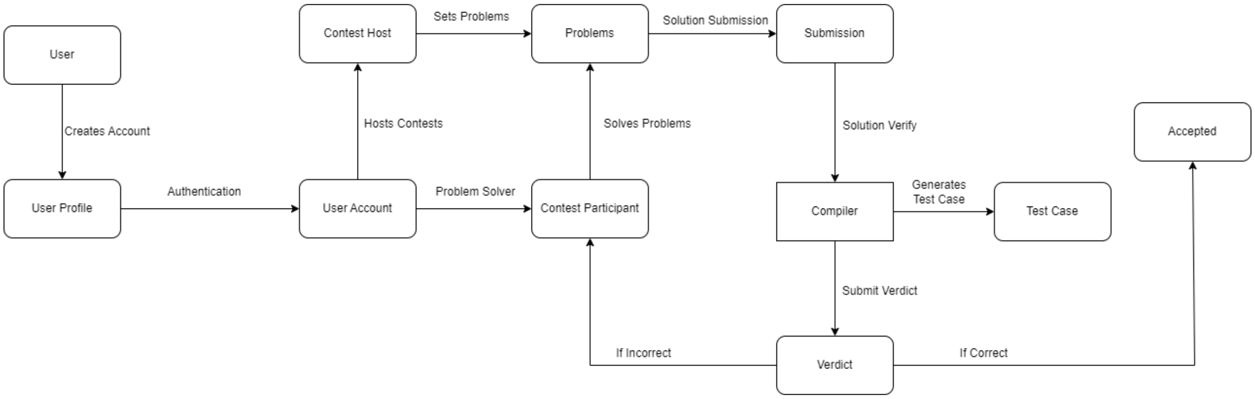
Methodology

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

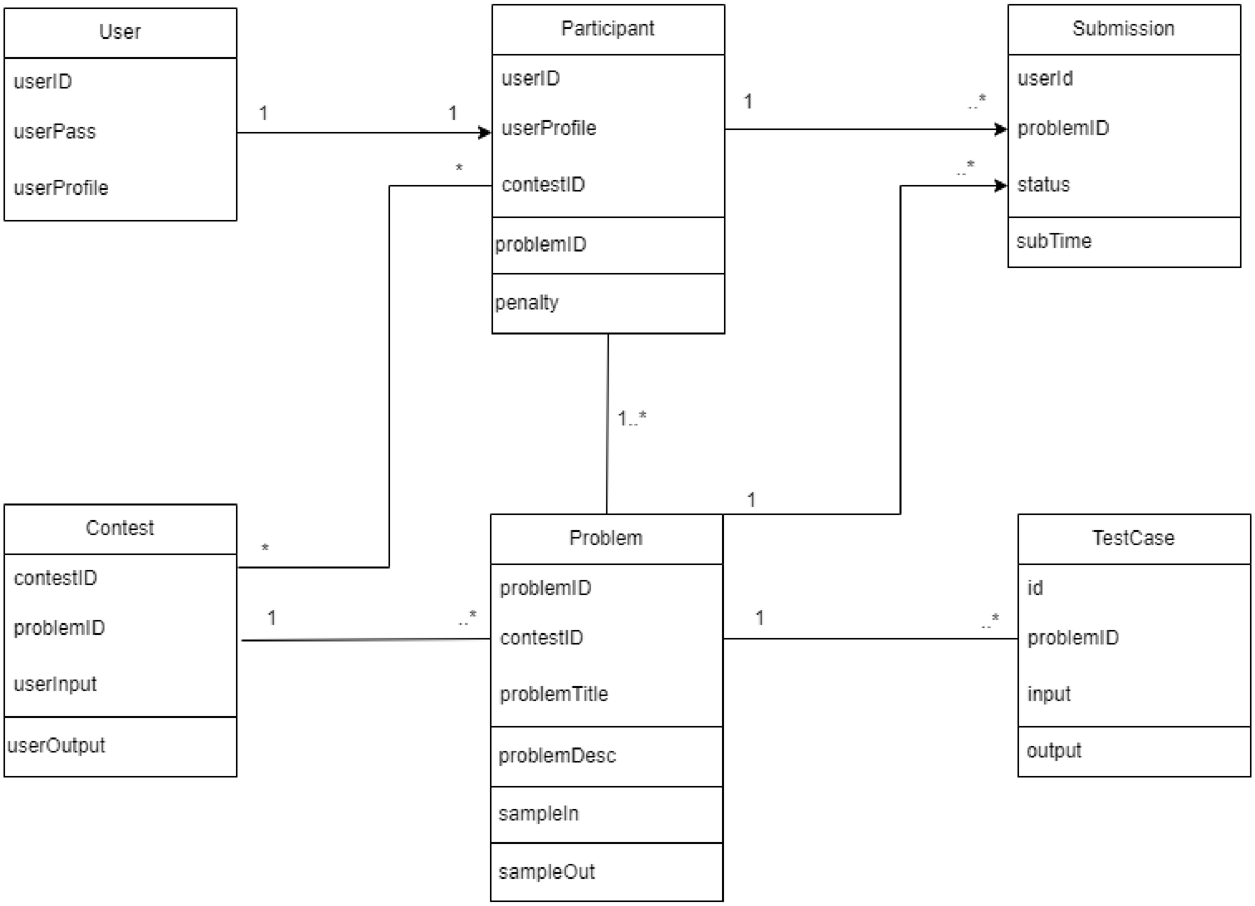
ER Diagram:



DFD Diagram:



UML Diagram:


User
userID
userPass
userProfile
1
1
Participant
userID
userProfile
contestID
problemID
penalty
*
1
..*
Submission
userId
problemID
status
subTime
..*
1..*
1
Contest
contestID
problemID
userInput
userOutput
*
1
..*
Problem
problemID
contestID
problemTitle
problemDesc
sampleIn
sampleOut
1
..*
TestCase
id
problemID
input
output

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.

Module 2: Resource Gathering

- **Sprint**
    - ⇨Gathering requirements.
    - ⇨Collecting problem sets.
    - ⇨Collecting developments tools.
    - ⇨Rearranging developments tools.

Module 3: User Profile

➢ Sprint

- ⇨Create “User Name”.
- ⇨Create “Password”.
- ⇨Create “User Avatar”.
- ⇨Create “Profile”.

➢ Backlog

- ⇨Gmail Account

Module 4: Level-1 of “Savant Coders”

➢ Sprint

- ⇨UI Design
- ⇨Authorization
- ⇨Contest Management
- ⇨Problem Integration
- ⇨Test Case Generation
- ⇨Compiler Integration
- ⇨Alpha Testing

➢ Backlog

- ⇨IDE (Integrated Development Environment).
- ⇨MySQl for database management.

⇨API.

⇨Github for version control.

Module 5: Level-2 of “Savant Coders”

➢Sprint

⇨UI Design

⇨Authorization

⇨Contest Management

⇨Problem Integration

⇨Test Case Generation

⇨Compiler Integration

⇨Alpha Testing

➢Backlog

⇨IDE (Integrated Development Environment).

⇨MySQl for database management.

⇨API.

⇨Github for version control.

Module 6: Level-3 of “Savant Coders”

- **Sprint**
  - ⇨UI Design
  - ⇨Authorization
  - ⇨Contest Management
  - ⇨Problem Integration
  - ⇨Test Case Generation
  - ⇨Compiler Integration
  - ⇨Alpha Testing

- **Backlog**
  - ⇨IDE (Integrated Development Environment).
  - ⇨MySQl for database management.
  - ⇨API.
  - ⇨Github for version control.

Module 7: Level-4 of “Savant Coders”

- **Sprint**
  - ⇨UI Design
  - ⇨Authorization
  - ⇨Contest Management

- Problem Integration
- Test Case Generation
- Compiler Integration
- Alpha Testing

Backlog

- IDE (Integrated Development Environment).
- MySQl for database management.
- API.
- Github for version control.

Module 8: Level-5 of “Savant Coders”

Sprint

- UI Design
- Authorization
- Contest Management
- Problem Integration
- Test Case Generation
- Compiler Integration
- Alpha Testing

Backlog

- IDE (Integrated Development Environment).

⇨MySQl for database management.
⇨API.
⇨Github for version control.

Module 9: Level-6 of "Savant Coders"

➢ Sprint

⇨UI Design
⇨Authorization
⇨Contest Management
⇨Problem Integration
⇨Test Case Generation
⇨Compiler Integration
⇨Alpha Testing

➢ Backlog

⇨IDE (Integrated Development Environment).
⇨MySQl for database management.
⇨API.
⇨Github for version control.

Module 10: Final Test Phase

- **Sprint**
  - Beta Testing
  - Address any issues and bugs.
- **Backlog**
  - User
  - Errors.

Module 11: Product Release

- **Sprint**
  - Website Publishing.
  - Monitor Server Performance.
  - Address bugs and reports.
- **Backlog**
  - Users

Module 12: Future Support

➢ Sprint

- ⇨Implement Bug Fixes.
- ⇨Implement New Features.
- ⇨Collect bug reports

➢ Backlog

- ⇨System Developers

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})
```

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, CO4	PO-(a), PO-(b)
K3	Engineering Discipline	Github, Grant Chart, Codeblocks, HTML viewer, Browser	CO7, CO8, CO9	PO-(a), PO-(b), PO-(i)
K4	Specialist Knowledge	Figma for HTML UI/UX Design	CO1, CO3	PO-(k), PO-(e)
K5	Engineering Design	Python, HTML, Bootstrap, JavaScript, DB Browser	CO3, CO4	PO-(c)
K6	Engineering Practice	VS Code, Pycharm, Github	CO4, CO9	PO-(k)

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, CO3	PO-(a), PO-(b)
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-(d), PO-(g)
P4	Familiarity of issues	Contest with lot of participants can slow down the server	CO7, CO8	PO-(h)

		decreasing the server performance.		
P7	Interdependence	Interdependency subsystem like • Data Collection • Problem organization • Generate results • Working IDE	CO4 CO6, CO9	PO-(k), PO-(i)

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	CO3, CO4	PO-(a)
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.	CO1, CO2, CO3	PO-(b), PO-(d), PO-(i)
A4	Consequences for society and the environment	Will help to nurture aspiring programmers. Improving their skill and increase the availability of proficient developers.	CO6	PO-(h)