

A Project Report on

Can't Code: A web-based Coding Platform

Submitted by:

SHETH CHINMAY MILIND (1930403245004)
THAKUR TEJAS AMARSINGH (1930403245044)
SHET ANISH MILIND (1930403245073)
SATERE NISHANT DINESH (1930403245062)

Under the guidance of

Prof. G. S. Mahamunkar

in fulfilment for the award of

**DIPLOMA
IN
COMPUTER ENGINEERING
2021-22**

at



**Institute of Petrochemical Engineering
Dr. Babasaheb Ambedkar Technological University**

Lonere, Tal. Mangaon, Dist. Raigad, Maharashtra (INDIA) – 402103

CERTIFICATE

This is to certify that, the Project entitled '*Can'tCode*' is a work carried out by

SHETH CHINMAY MILIND (1930403245004)

THAKUR TEJAS AMARSINGH (1930403245044)

SHET ANISH MILIND (1930403245073)

SATERE NISHANT DINESH (1930403245062)

the students of Third Year of Diploma in Computer Engineering at Dr. BabasahebAmbedkar Technological University's Institute of Petrochemical Engineering, Lonere, during the academic year 2021 - 2022 in the fulfillment of the requirements for the award of Diploma in Computer Engineering.

Prof. Geetanjali S. Mahamunkar

Guide

Department of Computer Engineering

Prof. Santosh M. Sabale

Head

Department of Computer Engineering

Place: Lonere

Date: _____

ABSTRACT

The spread of pandemic COVID-19 has drastically disrupted every aspect of human life including education. It has created an unprecedented test on education. In many educational institutions around the world, campuses are closed and teaching-learning has moved online. Internationalization has slowed down considerably. In India, about 32 crore learners stopped to move schools/colleges and all educational activities ended. The closure of colleges and universities has widespread individual, organizational, and learning and teaching implications for students, faculty, administrators, and the institutions themselves.

Lack of access to technology or fast, reliable internet access can prevent students in rural areas and from disadvantaged families. Lack of access to technology or good internet connectivity is an obstacle to continued learning. Teachers have reported the Institute closures negatively impact student learning outcomes, that students are more likely to complete assignments if they have access to internet at home. Since lack of digital mediums among the student, despite of their geographical locations. It became much more difficult for them to acquire practical knowledge in their respective fields. Online learning only drove the students through theoretical knowledge, losing one of the most important skills like problem solving, practical programming skills, etc.

The ultimate goal of our project is to address all the problems faced by students as well as faculties too. Our project focuses developing a website which can run on almost any devices, where students with underprivileged backgrounds will gain interest in programming, learn and develop as well as practice their problem-solving skills and of course increase stats to build skills for real world challenges.

ACKNOWLEDGEMENT

Firstly, we are indebtedly grateful to our Head of Department **Prof. S. M. Sabale** for his support. We are also grateful to our guide **Prof. G. S. Mahamunkar** for her valuable guidance and support. Without their support this project would not have been initiated.

We deeply obliged to thank all the faculty members of **Department of Computer Engineering**, for their valuable guidance and constant monitoring throughout the work of our project. Also, we would like to thank our college for providing us all the necessary resources for the project.

We, especially we are very thankful to our team members, for their contribution to the project work, kind co-operation and encouragement which help to developing the project.

Thank You!

Sheth Chinmay Milind (1930403245004)

Thakur Tejas Amarsingh (1930403245044)

Shet Anish Milind (1930403245073)

Satere Nishant Dinesh (1930403245062)

Department of Computer Engineering,
Institute of Petrochemical Engineering,
Dr. Babasaheb Ambedkar Technological University,
Lonere - Raigad

TABLE OF CONTENTS

CERTIFICATE.....	ii
ABSTRACT.....	iii
ACKNOWLEDGEMENT.....	iv
1. INTRODUCTION	7
1.1 Problem Definition.....	7
1.2 Scope & Objective.....	7
1.3 Language & Tools.....	8
1.4 System Requirements.....	8
2. LITERATURE SURVEY	9
2.1 Existing Systems.....	9
2.2 Flaws in Existing Systems.....	10
2.2 Proposed System.....	10
3. REQUIREMENT ANALYSIS	11
3.1 Functional Requirements.....	11
3.2 Non-Functional Requirements.....	11
3.3 Test Criteria.....	12
4. DESIGN	13
4.1 Software Development Life-Cycle (SDLC) Model.....	13
4.2 Flow Charts.....	14
5. User Interface (UI)	19
6. Source Code Snippets	23
7. Meet the TEAM	27
CONCLUSION	29
REFERENCES	30

LIST OF FIGURES

Sr.	Fig. No.	Name of the Figure	Page No.
1.	4.1	Iterative Model (SDLC)	14
2.	4.2	Data Flow Diagram	15
3.	4.3	Class Diagram	15
4.	4.4	Entity-Relationship Diagram	16
5.	4.5	Control Flow Diagram	17
6.	4.6	Use Case Diagram	18
7.	4.7	System Architecture	19
8.	5.1	Sign-Up Page	20
9.	5.2	Login Page	20
10.	5.3	Home Page	21
11.	5.4	Question Page	21
12.	5.5.1	Coding Page	22
13.	5.5.2	Coding Page	22
14.	5.6	About-Us Page	23
15.	6.1	Python Server	24
16.	6.2	pymongo	24
17.	6.3	Postman API	25
18.	6.4.1	Database	26
19.	6.4.2	Database	26
20.	6.5.1	Sign-up Logic	27
21.	6.5.2	Sign-up Logic	27
22.	7.1	Project Meetings	29

LIST OF TABLES

Sr.	Tab. No.	Name of the Table	Page No.
1.	3.1	Test Criteria Table	13

1. INTRODUCTION

“Can't Code: A web-based Coding Platform”



1.1 Problem Definition:

- **Problem Statement:** ‘Can't Code’ is a web-based Coding Platform for aspiring students to practice and develop their programming and problem-solving skills
- **Relevance:** “Practice makes anyone Perfect”. Studies have shown that, students who practice on their practical skills are likely to have greater chances of succeeding. Also, employers hire students not only by observing their theoretical skills but also aims on practical knowledge. This resulted us into creating a platform focusing on underprivileged students to practice their programming and problem-solving skills.

1.2 Scope & Objective:

- **Scope:**
 1. **Accessibility:** As it is a web-based platform, it can be accessed anytime.
 2. **Practice:** Practice can be done on a regular basis.
 3. **Free-of-cost:** Does not charge for anything.
 4. **Training:** This can be used by students as well as institutes to train their students through assigning various assignments or problems.
- **Objectives:**
 1. **Easy-to-use:** Ease of accessing the platform.
 2. **Usability:** User Friendly Interface.
 3. **Enhance Skills:** Enhance programming skills of students.
 4. **Develop:** Problem-solving and Logical Thinking.
 5. **Gain:** Confidence to build their own projects.



1.3 Languages & Tools:

1. **Front End:** HTML, CSS, JavaScript.
2. **Back End:** MongoDB, Python, Flask, Postman.
3. **Operating System:** Ubuntu LTS 20.04.
4. **Web Browser:** Google Chrome or Mozilla Firefox.
5. **IDE used:** Visual Studio Code 1.63.0
6. **Tools:** NicePage



1.4 System Requirements:

1. **Screen Resolution:** 1280x1024 or larger
2. **Application Window:** 1024x680 or larger
3. **Operating System:** Windows 8 or later, macOS Sierra 10.12 or later, Ubuntu 14.04+
4. **Web Browser:** Google Chrome (Recommended)
5. **Internet Connection:** Required
6. **Memory:** 2GB Minimum, 4GB Recommended



2. LITERATURE SURVEY

2.1 Existing Systems:

1. HackerRank:



Hacker Rank is a site for Programmers from all over the world to solve programming problems in different CS domains like algorithms, machine learning and artificial intelligence, and to excel in different programming paradigms like functional programming. If you are not used to solving different kinds of problems, then it will appear hard. Hacker rank has around 5 levels of difficulty: Easy, Intermediate, Hard, Expert and Advanced.

2. LeetCode:



LeetCode is one of the most well-known online judge platforms that you can use to practice your programming skills by solving coding questions. It has over 1,100 different problems, support for over 18 programming languages, and an active community that is always there to help you with the solutions you come up with. If your intention is to hone your coding skills, then this online judge platform is one of the best that you can use.

3. CodeWars:



CodeWars provides a large collection of coding challenges submitted and edited by their own community. You can solve the challenges directly online in their editor in one of several languages. You can view a discussion for each challenge as well as user solutions.

2.2 Flaws in Existing System:

1. **Developer-Oriented**
2. **Costly**
3. **Frustrating**



After studying this existing system, it concluded that these sites are not suitable for beginners as these sites are developer-oriented rather than being student-oriented. People need to pay for solving premium problems, and thereby limiting the learning. After solving a few problems, the level of problems drastically changes and some people finds it frustrating.

2.3 Proposed System:

As a solution to flaws in existing system, we came up with '*Can'tCode*' that is:

1. **Student-Oriented**
2. **Free & Open-Source**
3. **Consistent Difficulty of Problems**



'*Can'tCode*' provides a potential solution to the problems mentioned above. It is more of a student-oriented platform. The Existing platform provide a wide range of programs to practice, but they are more difficult and confusing as they are developer oriented. Whereas, using '*Can'tCode*' can be very friendly to the beginners as it is free-of-cost. Also, the difficulty levels in '*Can'tCode*' increase gradually, helping the user to learn better at a suitable pace!



3. REQUIREMENT ANALYSIS

3.1 Functional Requirements:

i. Registration:

If a user wants to practice programming and improve programming skills, then he/she must be registered, an unregistered user can't use the website. All the data of users is maintained securely.

ii. Login:

User logs in to the system by entering valid user id and password to solve the programming questions and then can use all facilities of the system.

iii. Solve Question:

The system will provide user list of question from database. Short description and long description are provided with the questions.

iv. Test cases result:

The user has to pass all the test cases.

v. Logout:

After the solving questions user can logout of system.



3.2 Non-Functional Requirements:

a) Security: The system's back-end servers shall only be accessible to authenticated administrator. Sensitive data will be encrypted before being sent over internet.

b) Reliability: The system's architecture is made having a focus on students.

c) Availability: The system should be available at all times; the user can access it using a desktop or a laptop. It means 24x7 availability.

d) Portability: The application is a web app available and runnable on web browsers.

3.3 Test Criteria:

Sr.	Functional Test Cases	Actual Output	Expected Output
1.	Verify if a user is able to register with appropriate credentials and login with same	User is registered with system.	User registered successfully
2.	Verify if clicking on login button will redirect to home page	Clicking on particular link is redirecting to proper page	Redirection on proper page
3.	Verify if user is able to access about us, questions, logout from the home page	User can access all the three pages	Accessing all three pages
4.	Clicking on question is redirecting to the IDE page where user is able to solve programming questions	User can solve the programming questions	Solving the programming questions
5.	After typing our program into IDE and clicking on submit button it should display proper testcases	Testcases are visible	It should display testcases

Table. 3.1: Test Criteria Table



4. DESIGN

4.1 Software Development Life-Cycle (SDLC) Model:

The SDLC Model on which 'Can'tCode' is based on is 'Iterative Model'.

- **Iterative Model:**

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).

The following illustration is a representation of the Iterative and Incremental model –

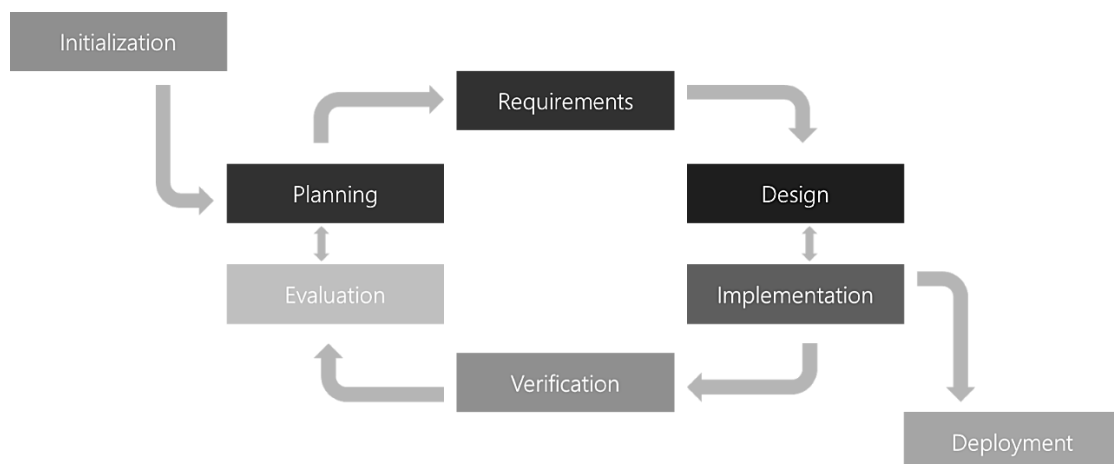


Fig. 4.1: Iterative Model (SDLC)

4.2 Flow Charts:

1. Data Flow Diagram:

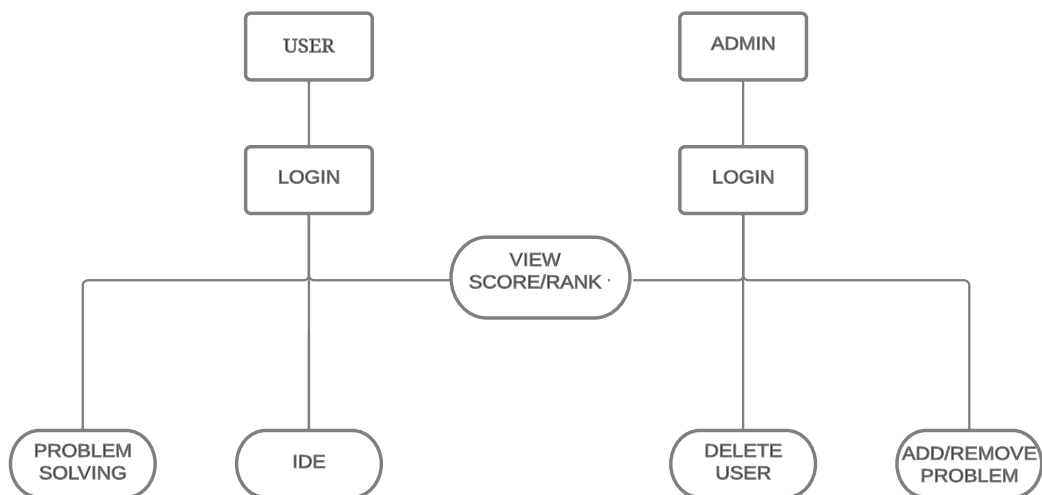


Fig. 4.2: Data Flow Diagram

2. Class Diagram:

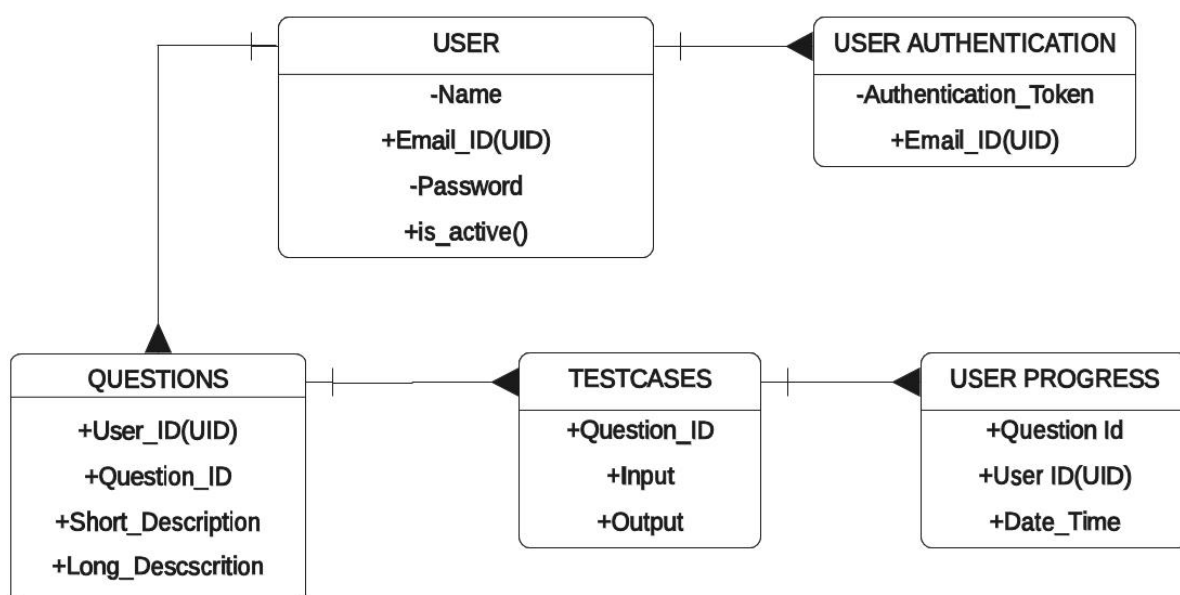


Fig. 4.3: Class Diagram

3. Entity-Relationship Diagram:

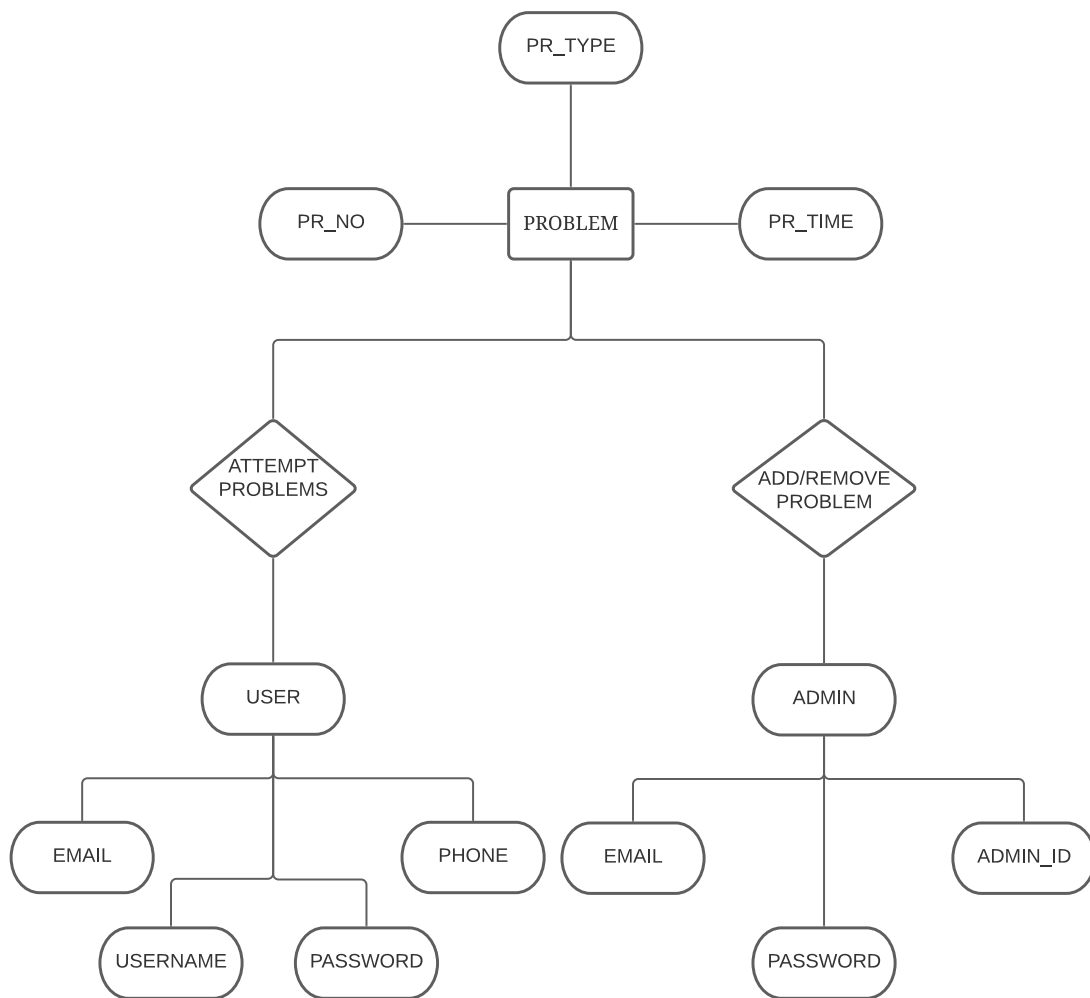


Fig. 4.4: Entity-Relationship Diagram

4. Control-Flow Diagram:

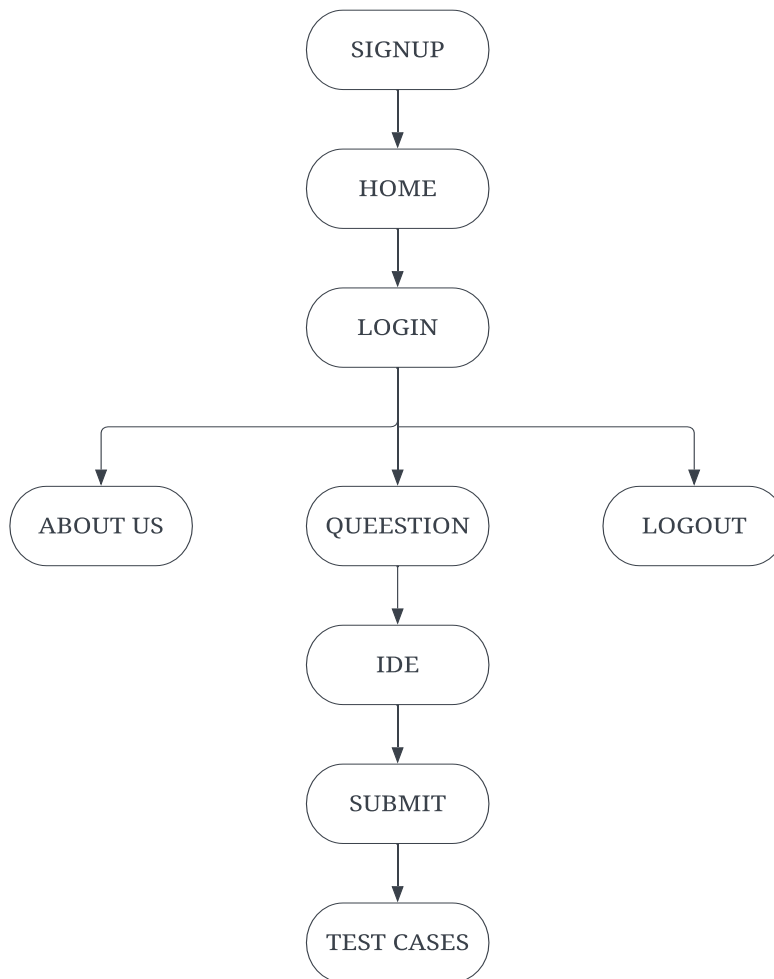


Fig. 4.5: Control-Flow Diagram

5. Use-Case Diagram:

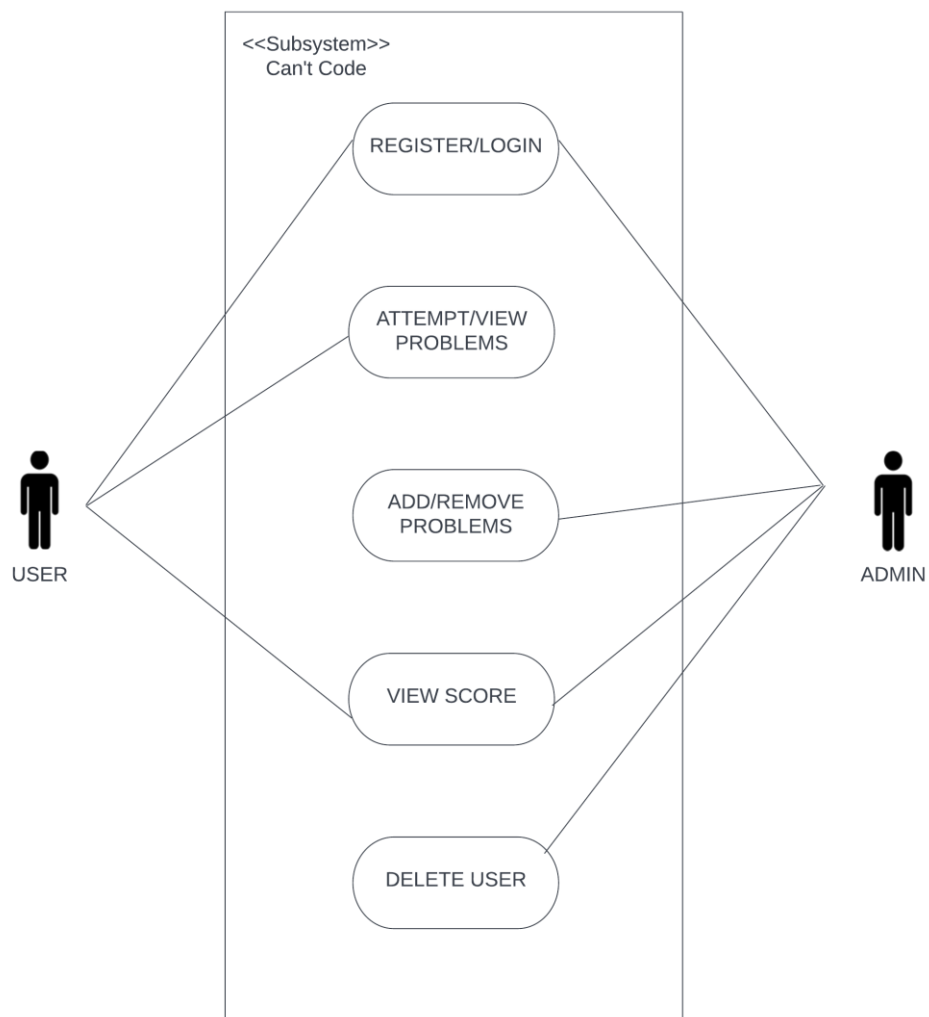


Fig. 4.6: Use-Case Diagram

6. System Architecture:

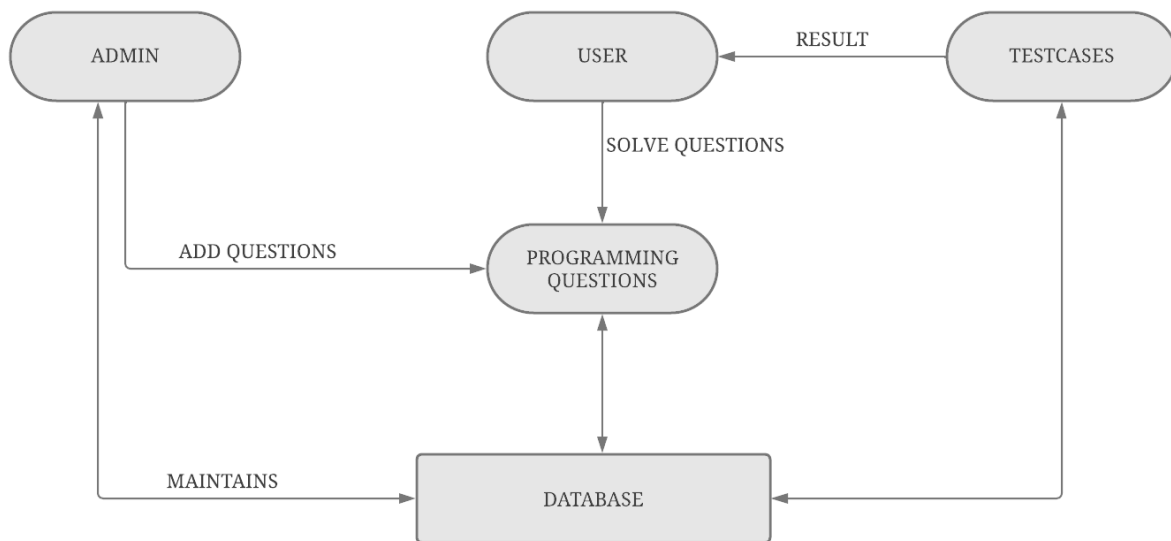
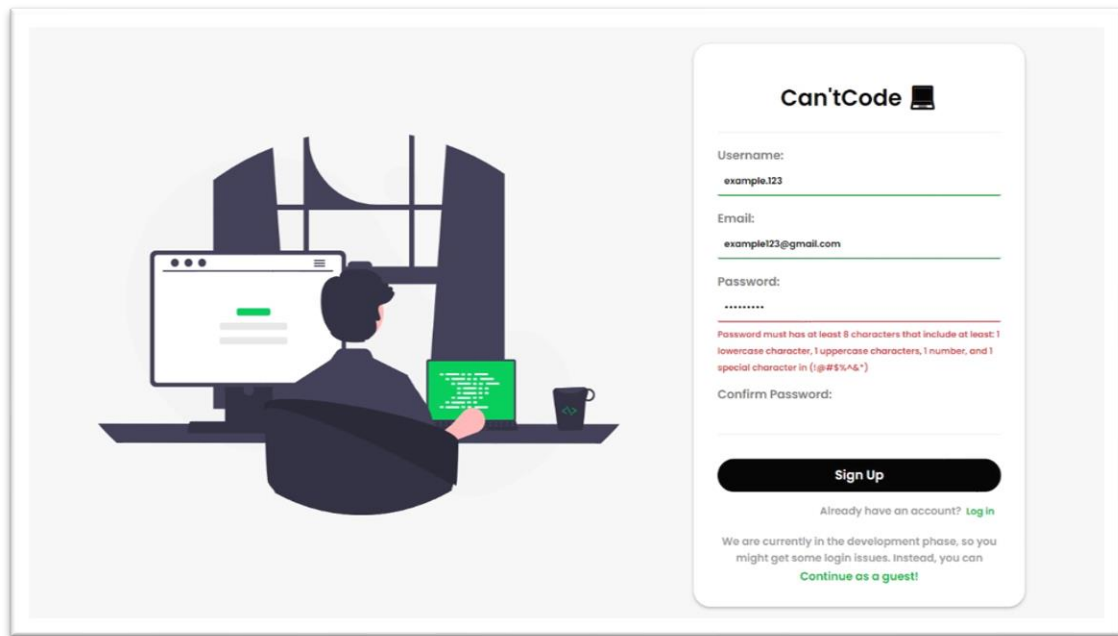


Fig. 4.7: System Architecture



5. USER INTERFACE (UI)

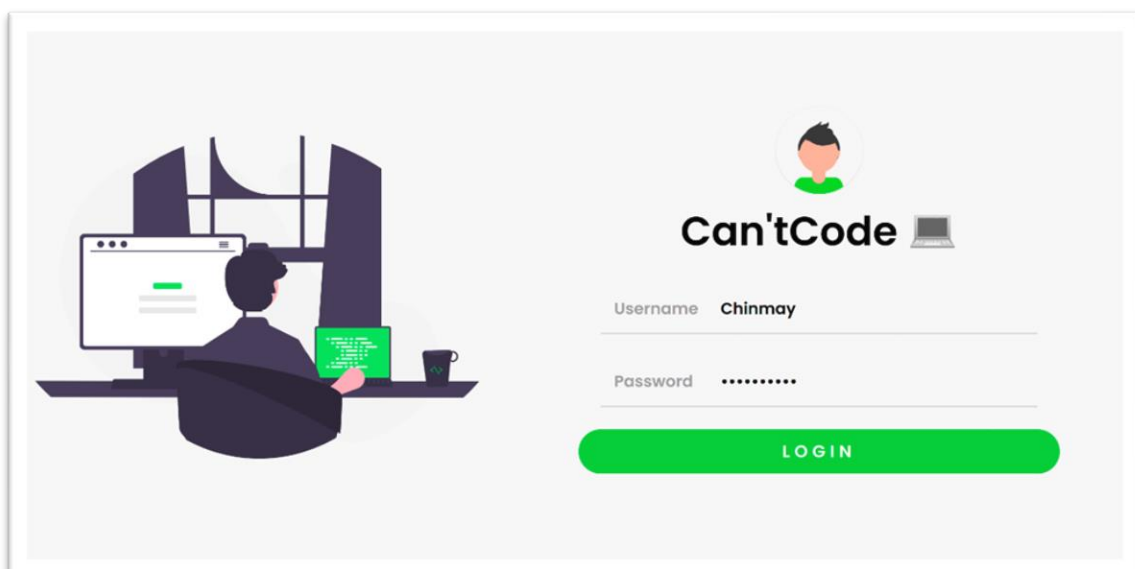
1. Sign-Up/Registration Page:



The image shows a UI mockup for a sign-up page. On the left, there is an illustration of a person sitting at a desk with a computer monitor and a laptop, with a window in the background. On the right, there is a white card with the 'Can'tCode' logo at the top. The card contains the following fields: 'Username:' with the value 'example.123', 'Email:' with the value 'example123@gmail.com', and 'Password:' with masked characters '.....'. Below the password field, there is a red error message: 'Password must have at least 8 characters that include at least: 1 lowercase character, 1 uppercase character, 1 number, and 1 special character in (!@#\$%^&*~)'. Below this is a 'Confirm Password:' field. At the bottom of the card is a black 'Sign Up' button. Below the button, there is a link 'Already have an account? Log in' and a message: 'We are currently in the development phase, so you might get some login issues. Instead, you can Continue as a guest!'.

Fig. 5.1: Sign-Up Page

2. Login Page:



The image shows a UI mockup for a login page. On the left, there is an illustration of a person sitting at a desk with a computer monitor and a laptop, with a window in the background. On the right, there is a white card with a user profile icon at the top. Below the icon is the 'Can'tCode' logo. The card contains the following fields: 'Username' with the value 'Chinmay' and 'Password' with masked characters '.....'. At the bottom of the card is a green 'LOGIN' button.

Fig. 5.2: Log-in Page

3. Home Page:

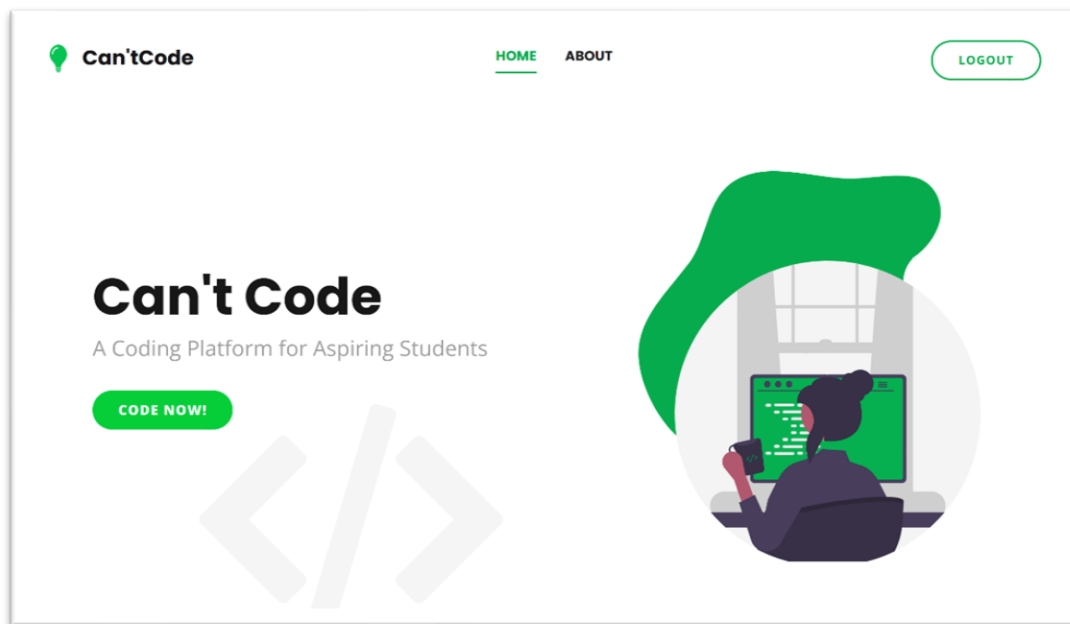


Fig. 5.3: Home Page

4. Question Page:

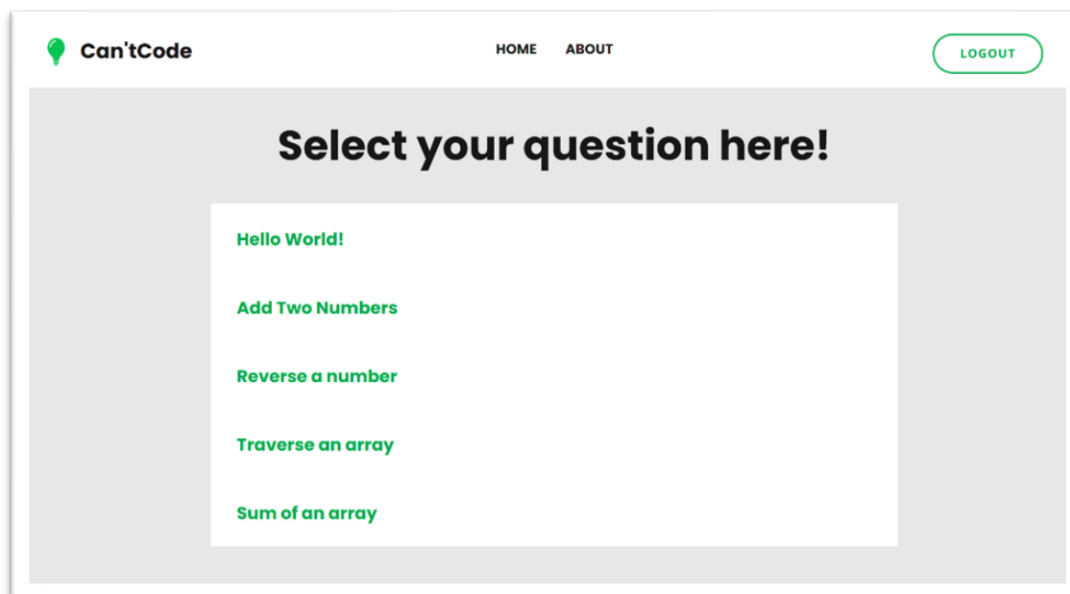


Fig. 5.4: Question Page

5. Coding Page:

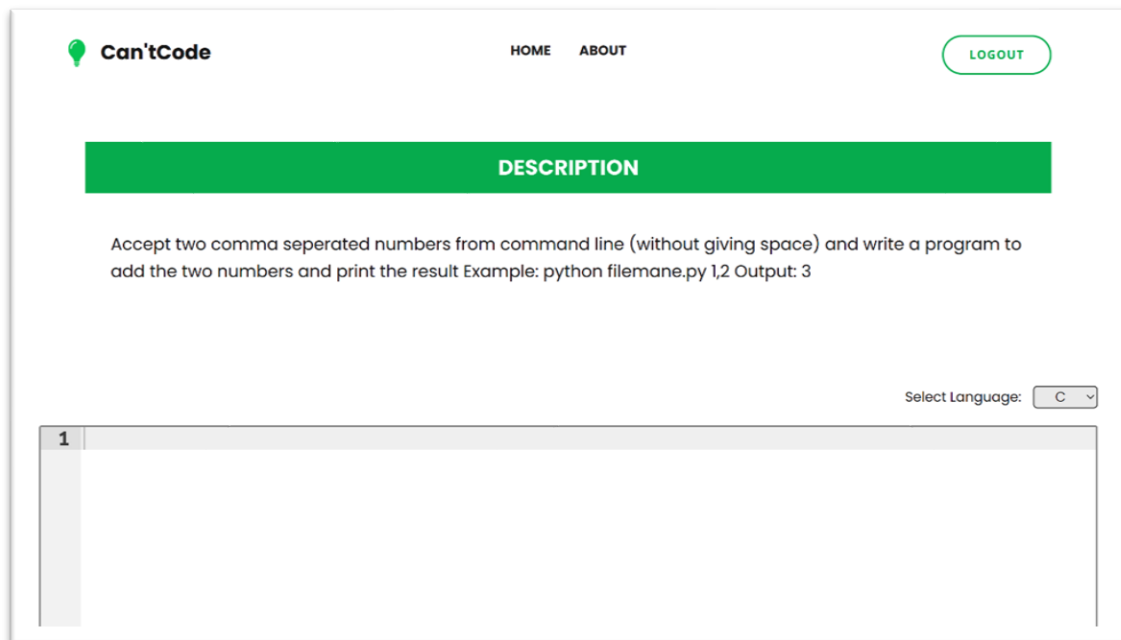


Fig. 5.5.1: Coding Page

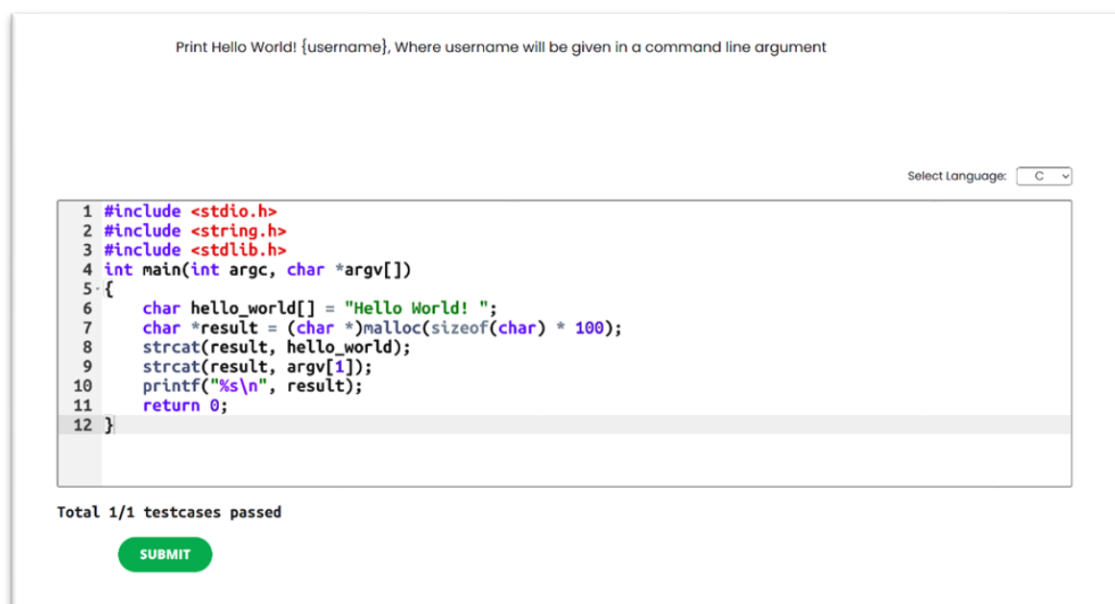


Fig. 5.5.2: Coding Page

6. About-Us Page:

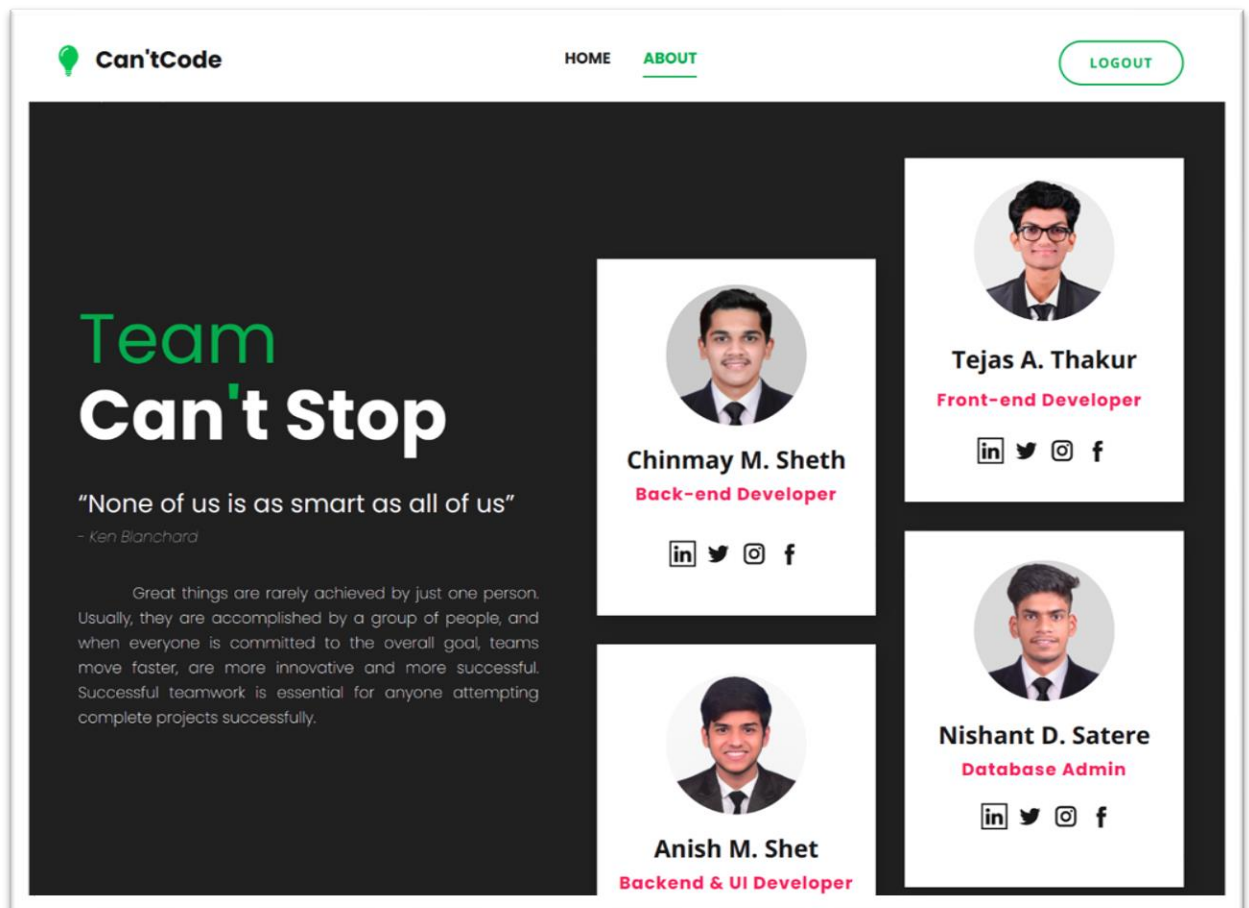


Fig. 5.6: About-Us Page



6. SOURCE CODE SNIPPETS

1. Python Server:

```
chinmay@chinmay-ThinkPad-E14: ~/CCV2/CCV2/APP
venv) chinmay@chinmay-ThinkPad-E14:~/CCV2/CCV2$ cd APP
venv) chinmay@chinmay-ThinkPad-E14:~/CCV2/CCV2/APP$ ls
i.out app.py controller dao program.c program.cpp program.py __pycache__ tmp.c utils venv
venv) chinmay@chinmay-ThinkPad-E14:~/CCV2/CCV2/APP$ python3 app.py
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5678/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 105-606-858
.27.0.0.1 - - [21/May/2022 20:44:01] "GET / HTTP/1.1" 404 -
.27.0.0.1 - - [21/May/2022 20:44:01] "GET /favicon.ico HTTP/1.1" 404 -
]
```

Fig. 6.1: Python Server

2. pymongo:

```
APP > utils > pymongo_fun.py > ...
1 |from pymongo import MongoClient
2
3
4 def get_mongodb_collection(collection_name):
5     mongo_client = MongoClient("mongodb://127.0.0.1:27017/")
6     db = mongo_client["cant_code"]
7     # print(db[collection_name])
8     return db[collection_name]
9
```

Fig. 6.2: pymongo



3. Postman API:

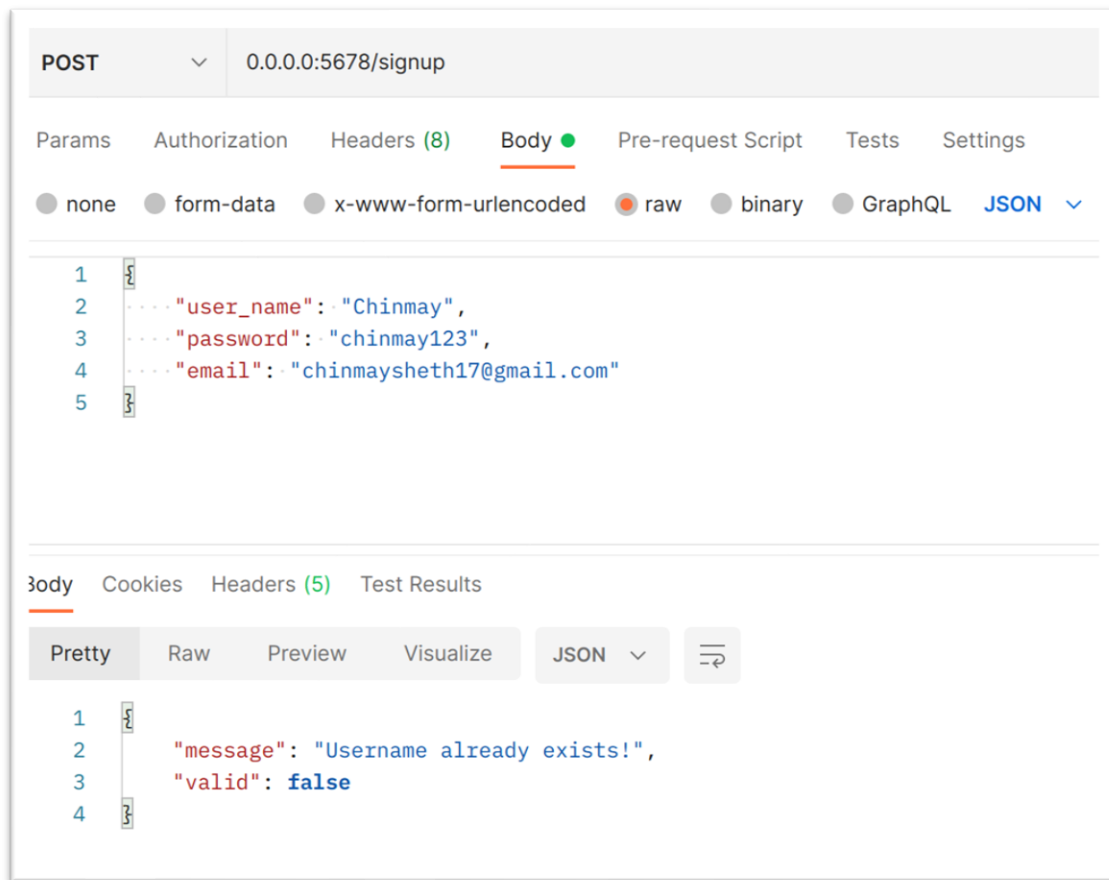
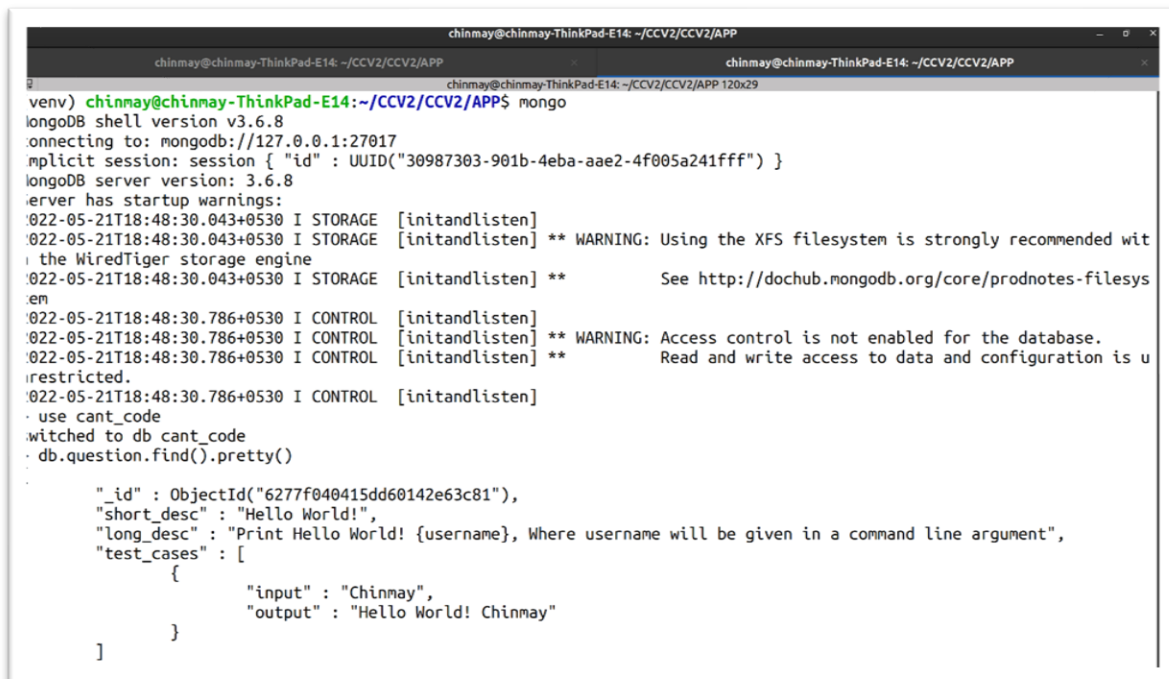


Fig. 6.3: Postman API



4. Database:



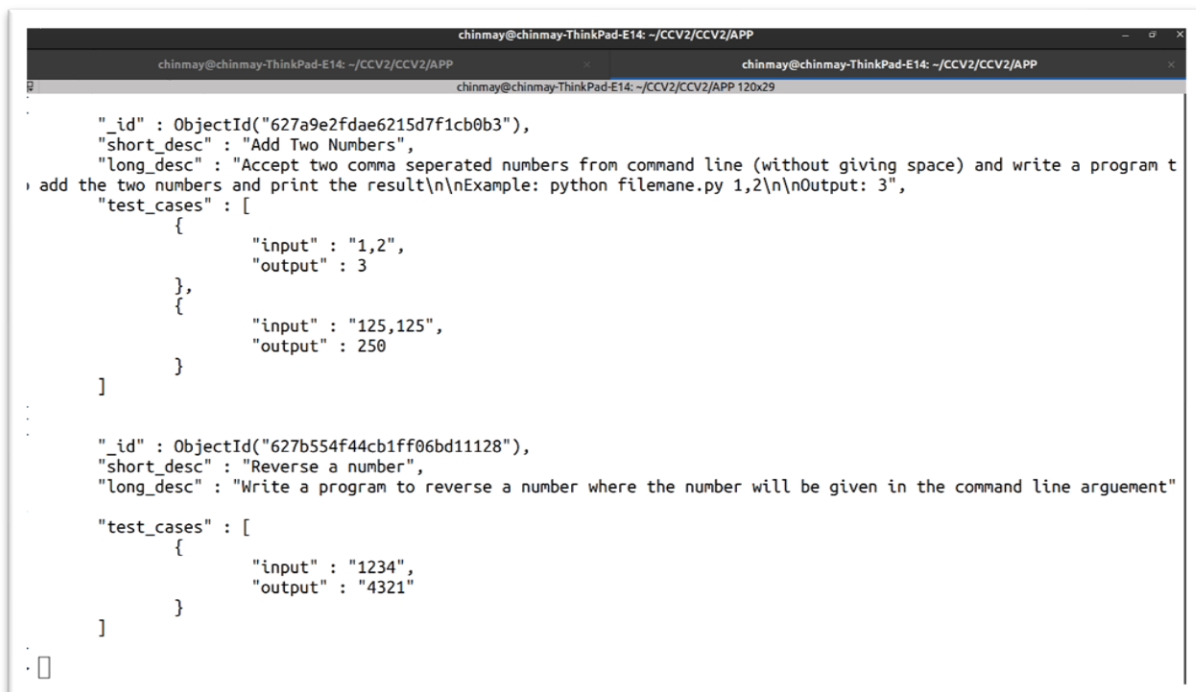
```

chinmay@chinmay-ThinkPad-E14: ~/CCV2/CCV2/APP
venv) chinmay@chinmay-ThinkPad-E14:~/CCV2/CCV2/APP$ mongo
longoDB shell version v3.6.8
connecting to: mongodb://127.0.0.1:27017
implicit session: session { "id" : UUID("30987303-901b-4eba-aae2-4f005a241fff") }
longoDB server version: 3.6.8
server has startup warnings:
022-05-21T18:48:30.043+0530 I STORAGE [initandlisten]
022-05-21T18:48:30.043+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
022-05-21T18:48:30.043+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem
022-05-21T18:48:30.786+0530 I CONTROL [initandlisten]
022-05-21T18:48:30.786+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
022-05-21T18:48:30.786+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
022-05-21T18:48:30.786+0530 I CONTROL [initandlisten]
use cant_code
switched to db cant_code
db.question.find().pretty()

  "_id" : ObjectId("6277f040415dd60142e63c81"),
  "short_desc" : "Hello World!",
  "long_desc" : "Print Hello World! {username}, Where username will be given in a command line argument",
  "test_cases" : [
    {
      "input" : "Chinmay",
      "output" : "Hello World! Chinmay"
    }
  ]

```

Fig. 6.4.1: Database



```

chinmay@chinmay-ThinkPad-E14: ~/CCV2/CCV2/APP
chinmay@chinmay-ThinkPad-E14:~/CCV2/CCV2/APP$ mongo
longoDB shell version v3.6.8
connecting to: mongodb://127.0.0.1:27017
implicit session: session { "id" : UUID("30987303-901b-4eba-aae2-4f005a241fff") }
longoDB server version: 3.6.8
server has startup warnings:
022-05-21T18:48:30.043+0530 I STORAGE [initandlisten]
022-05-21T18:48:30.043+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
022-05-21T18:48:30.043+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem
022-05-21T18:48:30.786+0530 I CONTROL [initandlisten]
022-05-21T18:48:30.786+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
022-05-21T18:48:30.786+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
022-05-21T18:48:30.786+0530 I CONTROL [initandlisten]
use cant_code
switched to db cant_code
db.question.find().pretty()

  "_id" : ObjectId("627a9e2fdae6215d7f1cb0b3"),
  "short_desc" : "Add Two Numbers",
  "long_desc" : "Accept two comma seperated numbers from command line (without giving space) and write a program to add the two numbers and print the result\n\nExample: python filemane.py 1,2\n\nOutput: 3",
  "test_cases" : [
    {
      "input" : "1,2",
      "output" : 3
    },
    {
      "input" : "125,125",
      "output" : 250
    }
  ]
}

  "_id" : ObjectId("627b554f44cb1ff06bd11128"),
  "short_desc" : "Reverse a number",
  "long_desc" : "Write a program to reverse a number where the number will be given in the command line argument"
  "test_cases" : [
    {
      "input" : "1234",
      "output" : "4321"
    }
  ]
}

```

Fig. 6.4.2: Database

6. Sign-up Logic:

User registration form with JavaScript

Validation & Database (MongoDB) linking.



```

user2.py U x
APP > dao > user2.py > ...
1  from curses import flash
2  from utils.mongodb_fun import get_mongodb_collection
3
4  def create_user(user_data):
5      user_collection = get_mongodb_collection("user")
6      # gives info about collection
7      user_data["_id"] = user_data["user_name"]
8      # id is never first generated when creating a new user
9
10     #print(user_collection.find({"user_name": user_data["user_name"]}))
11     if len(list(user_collection.find({"user_name": user_data["user_name"]}))) > 0:
12         print("Exists!")
13         return {"message": "Username already exists!", "valid":False}
14     else:
15         user_collection.insert_one(user_data)
16         return {"message": "Account created Successfully!", "valid":True}
17
18 def update_user(id, user_data):
19     user_collection = get_mongodb_collection("user")
20     user = user_collection.find_one({"_id": id})
21     # print(user["user name"])

```

Fig. 6.5.1: Sign-up Logic

```

user2.py U x
APP > dao > user2.py > ...
23     if user["user_name"] != user_data["user_name"] or user["_id"] != user_data
    ["_id"]:
24         print("Cannot Update Record!")
25         return "Cannot update Record!"
26     else:
27         user_collection.update_one({"_id": id}, {"$set": user_data})
28         return "Record updated Successfully!"
29
30 # GET METHOD used to fetch the user data
31 def get_user(id):
32     user_collection = get_mongodb_collection("user")
33     return user_collection.find_one({"_id": id})
34
35 def validate_user(user_name, password):
36     user_collection = get_mongodb_collection("user")
37     user = user_collection.find_one(
38         {"user_name": user_name, "password": password})
39     if user:
40         return {"name" : user["user_name"], "valid":True}
41     else:
42         return {"message" : "Cannot Login. Invalid Credentials!", "valid":False}

```

Fig. 6.5.2: Sign-up Logic

7. Meet the TEAM

Team Can't Stop 🔥

"None of us is as smart as all of us"

– Ken Blanchard

The name CAN'T stands for:



C – chinmay



A – anish



N – nishant



'T - tejas

Can't Stop

"Great things are rarely achieved by just one person. Usually, they are accomplished by a group of people, and when everyone is committed to the overall goal, teams move faster, are more innovative and more successful. Successful teamwork is essential for anyone attempting complete projects successfully"



7.1 Team Can't Stop during their project work:

Team Can't Stop discussing and working on the project 'Can't Code', in virtual meetings using Discord:

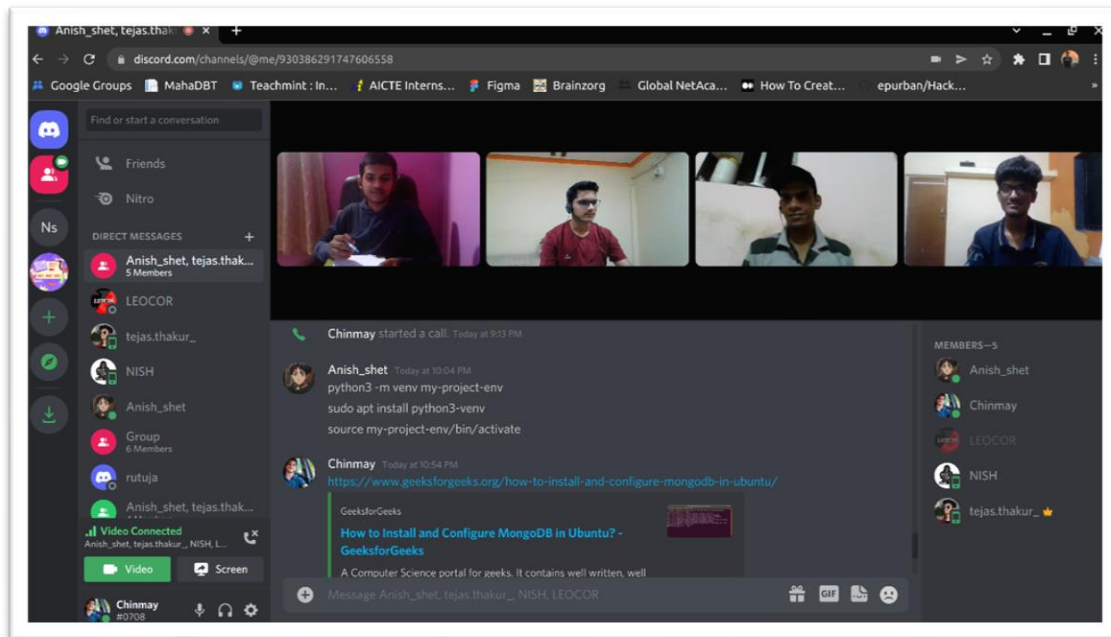


Fig. 7.1: Project Meetings

7.2 Roles of team members:

1. **Chinmay Milind Sheth:** Team Leader & Back-end Developer
2. **Anish Milind Shet:** Back-end & UI Developer
3. **Nishant Dinesh Satere:** Database Administrator
4. **Tejas Amarsingh Thakur:** Front-End Developer & Documentation



Can't Code

A CREATION BY

Team Can'tStop

CONCLUSION

Can'tCode is a web-based programming platform which can help students to achieve their skills in the Computer Field, they can upgrade their problem-solving skills. In the Era of Internet, where tech giants select students based on their coding skills and problem solving and everything in this world is eventually tied to the web.

So, our idea started with this, where our dream was to create a coding platform which is specially built for students, it can be accessible to them anytime, anywhere. They should not be charge for developing their skills too. And eventually they can gain confidence to build their own real-world projects.

Our application is based on test concept of test cases. A test case consists of an output to the code and an expected output. Once users submit the code, it is run against all the test cases. The output from the user's code is compared with the expected output to see whether the test cases has passed or failed.

We are pretty sure that using Can'tCode will significantly contribute to the students' learning process.

REFERENCES

YouTube: <https://www.youtube.com/c/CodeWithHarry>

Flask: <https://flask.palletsprojects.com/en/2.1.x/>

MongoDB: <https://www.mongodb.com/docs/drivers/pymongo/>

Postman API: <https://learning.postman.com/docs/>

Python: <https://docs.python.org/3/><https://docs.python.org/3/>