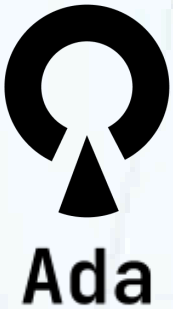
SOFTWARE TECHNICAL DOCUMENT



Ada - Programming Learning Platform

Name:- G. A. Pasindu Vidunitha (25542)
Class:- 12NSTK

04/10/2025

Version 1.0.0

REVISION HISTORY			
DATE	**VERSION**	**DESCRIPTION**	**AUTHOR**
04/10/2025	1.0.0	First full version of the technical documentation, including UI designs, ER diagram, SQL queries, routes, colour & fonts, user interactions, and detailed component breakdown.	G. A. Pasindu Vidunitha

Table of Contents

INTRODUCTION

This document gives the necessary technical requirements to develop this learning platform for programming. This platform helps programmers improve their coding skills via text-based tutorials, real-time feedback, and AI-powered support. The project serves as an educational resource for users who need to learn programming languages and concepts, as well as solve coding problems, and they are assisted by both AI tools and community members.

PURPOSE

The main purpose of this project is to develop an all-in-one learning platform. This project allows users to:

1. Access well-documented programming courses and tutorials.
2. Solve coding challenges and get instant feedback on them.
3. Ask questions from other programmers who are in this community.
4. Answer to other programmers' questions.
5. Get AI-powered code review and debugging assistance.

INTENDED AUDIENCE AND PERTINENT SECTIONS

AUDIENCE	DESCRIPTION	PERTINENT SECTIONS
Developers	Responsible for building and maintaining the platform.	System Overview, Code structure, API Endpoints, Deployment
Testers	Responsible for resolving bugs and issues associated with this project	Testing and debugging, Code structure

PROJECT SCOPE

This Project aims to create the best platform for both new and advanced programmers to develop their programming knowledge while keeping their motivation for programming. The main goal of this platform is to make coding education more accessible and engaging for people.

DESCRIPTION

PRODUCT PERSPECTIVE

This learning platform gives an interactive learning environment for programmers. This product originates from the need for an all-in-one, dynamic, and engaging platform for new programmers. It is difficult to get a solid knowledge of programming at the start of programming and to keep that motivation consistently. Expected functionalities are Side-by-side tutorials and a code editor, instant feedback, AI assistance for code review, and a gamification system that encourages students.

FEATURES

1. **Well-documented and Structured text-based tutorials**
    - Learning path for new programmers with correct guidelines
2. **Real-Time coding challenges**
    - Users can solve interactive coding challenges and get feedback on them
3. **AI-powered code review**
    - AI assistance to analyze your code and help users with errors and bugs
4. **Gamification**
    - Points, Badges, and leaderboards to motivate the users and check their progress
5. **Community Interaction**
    - Users can discuss their problems with other users
6. **Integration with External Resources**
    - User can enhance their courses, tutorials, and resources if they like.

USER OVERVIEW

- **Beginners:**
  People who are new to programming and who are looking for tutorials that are easy to understand.
- **Intermediate Users:**
  Users who have some knowledge of programming.
- **Advanced Users**
  Users who want to improve their coding knowledge further.

OPERATING ENVIRONMENT

- **Hardware platform:**
  This Platform is web-based, therefore, anyone who has a device with an internet connection and a web browser can access this platform.
- **Operating Systems:**
  This platform works on most operating systems, including Windows, macOS, and Linux.
- **Software requirements:**
  This platform needs an internet connection for functionality, and it will integrate with external APIs for additional content.

CONTRAINTS: IMPLEMENTATION / DESIGN

Describe limitations impacting development.

- **Limited Backend Database:**
  As this platform uses SQLite, there might be limitations in the database. This might be a problem when the number of users increases.
- **Real-time feedback Performance:**
  AI will take some time to give as much accurate feedback as possible, and because of that, there might be a problem when it comes to performance.
- **Cross-Browser Compatibility:**
  The platform should work on all browsers as expected without having problems with the design, etc.

ASSUMPTIONS / DEPENDENCIES

Detail all assumed factors (not known facts) that could potentially impact the technical specifications set forth. Include external factors.

- **User Internet Connection:**
  This platform assumes that users will have a stable internet connection.
- **AI assistance:**
  The successful AI-powered features depend on the machine-learning model's accuracy.
- **Third-party APIs for additional Content:**
  As the platform depends on external AI APIs to expand its learning materials, this might be a problem if third-party providers discontinue or change

SYSTEM FEATURES

SYSTEM FEATURE 1: User Authentication

DESCRIPTION AND PRIORITY	● The system allows users to create accounts, log in, and manage their profiles ● **Priority**: High
STIMULUS / RESPONSE SEQUENCES	● **Stimulus**: User provides login details or registers a new account ● **Response:** The system verifies account details and gives access to the platform
FUNCTIONAL REQUIREMENTS	● The System must validate the login process by a secure authentication method. ● Users should have a customizable profile, such as a username and custom avatar. ● User accounts should be allowed to recover passwords.

SYSTEM FEATURE 2: Courses and Learning Routes

DESCRIPTION AND PRIORITY	● This system offers a list of available courses categorized by language. ● **Priority**: High
STIMULUS / RESPONSE SEQUENCES	● **Stimulus**: The user selects a course. ● **Response:** The system displays the selected course.
FUNCTIONAL REQUIREMENTS	● The platform must display the relevant course when the user selects that course. ● Users should be able to track their progress with each course.

SYSTEM FEATURE 3: Code editor and Compiler integration

DESCRIPTION AND PRIORITY	• An integrated code editor that allows users to write code and compile it on this platform. • **Priority**: High
STIMULUS / RESPONSE SEQUENCES	• **Stimulus:** Users write code in the given code editor. • **Response:** The system compiles those codes and shows the relevant output
FUNCTIONAL REQUIREMENTS	• The code editor must support multiple programming languages. • The platform must provide real-time feedback on code errors and output. • Users should be able to submit their codes in the challenges section to check their answers.

SYSTEM FEATURE 4: Coding challenges

DESCRIPTION AND PRIORITY	• The platform provides various coding challenges for users at different difficulty levels. • **Priority**: High
STIMULUS / RESPONSE SEQUENCES	• **Stimulus:** Users select a coding challenge and submit the answer. • **Response:** The system presents the selected coding challenge, then evaluates the user's solution and gives feedback on that using AI.
FUNCTIONAL REQUIREMENTS	• This platform must allow users to submit their solutions and get detailed feedback on them.

SYSTEM FEATURE 5: AI assistance for code reviews and debugging

DESCRIPTION AND PRIORITY	● This platform uses AI to get feedback and debug Users' codes and assist them. ● **Priority**: High
STIMULUS / RESPONSE SEQUENCES	● **Stimulus:** Users submit their code for review or debugging ● **Response:** The AI assistance analyzes that and gives feedback on that.
FUNCTIONAL REQUIREMENTS	● The system must provide real-time debugging assistance for users. ● AI must give suggestions to the user's code if they ask.

SYSTEM FEATURE 6: Chat with other programmers

DESCRIPTION AND PRIORITY	● Users can ask questions and give answers ● **Priority**: High
STIMULUS / RESPONSE SEQUENCES	● **Stimulus:** Users can post answers and questions ● **Response:** The system allows users to post questions and answers, and gives a notification to the original poster when someone responds
FUNCTIONAL REQUIREMENTS	● The platform must allow for the posting of questions and answers ● Users should be able to search for them by keywords

SYSTEM FEATURE 10: Integration with External APIs for Additional Resources

DESCRIPTION AND PRIORITY	● This platform integrates with external APIs (e.g, Gemini, ChatGPT, or any other AI model) to provide extra learning materials ● **Priority**: High
STIMULUS / RESPONSE SEQUENCES	● **Stimulus:** User accesses external resources ● **Response:** The system retrieves data from the external API and displays it in the platform
FUNCTIONAL REQUIREMENTS	● Users should be able to access external learning materials according to their wishes. ● The system should handle errors with those external APIs

SYSTEM FEATURE 9: Notifications and Reminders

DESCRIPTION AND PRIORITY	● User receives notifications on responses, updates, etc ● **Priority**: Low (If the developer has time)
STIMULUS / RESPONSE SEQUENCES	● **Stimulus:** A new update, challenge, or message is available ● **Response:** The system sends a notification on that
FUNCTIONAL REQUIREMENTS	● Users should get those notifications

SYSTEM FEATURE 7: Progress Analytics and Feedback

DESCRIPTION AND PRIORITY	• The platform tracks the user's progress using courses and challenges that are completed by users • **Priority**: Low (If the developer has time)
STIMULUS / RESPONSE SEQUENCES	• **Stimulus:** User completes a course or a challenge • **Response:** The system generates a detailed progress report
FUNCTIONAL REQUIREMENTS	• Progress reports should be represented visually • The system must detect the user's progress

SYSTEM FEATURE 8: Customizable User Interface

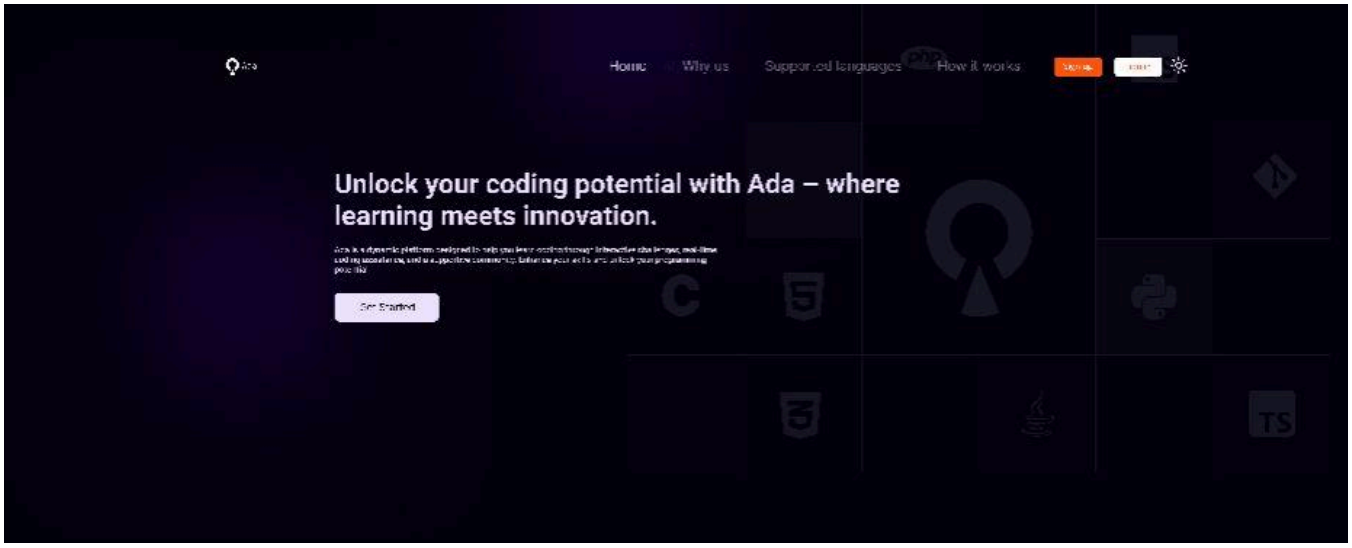
DESCRIPTION AND PRIORITY	• This platform allows users to switch from light themes to dark themes or vice versa • **Priority**: Low
STIMULUS / RESPONSE SEQUENCES	• **Stimulus:** User change their interface • **Response:** The system should update that immediately
FUNCTIONAL REQUIREMENTS	• The platform must support light themes and dark themes • The system must remember the user's decision

REQUIREMENTS OF EXTERNAL INTERFACE

USER INTERFACE BREAKDOWN WITH PURPOSE, SQL & ROUTES

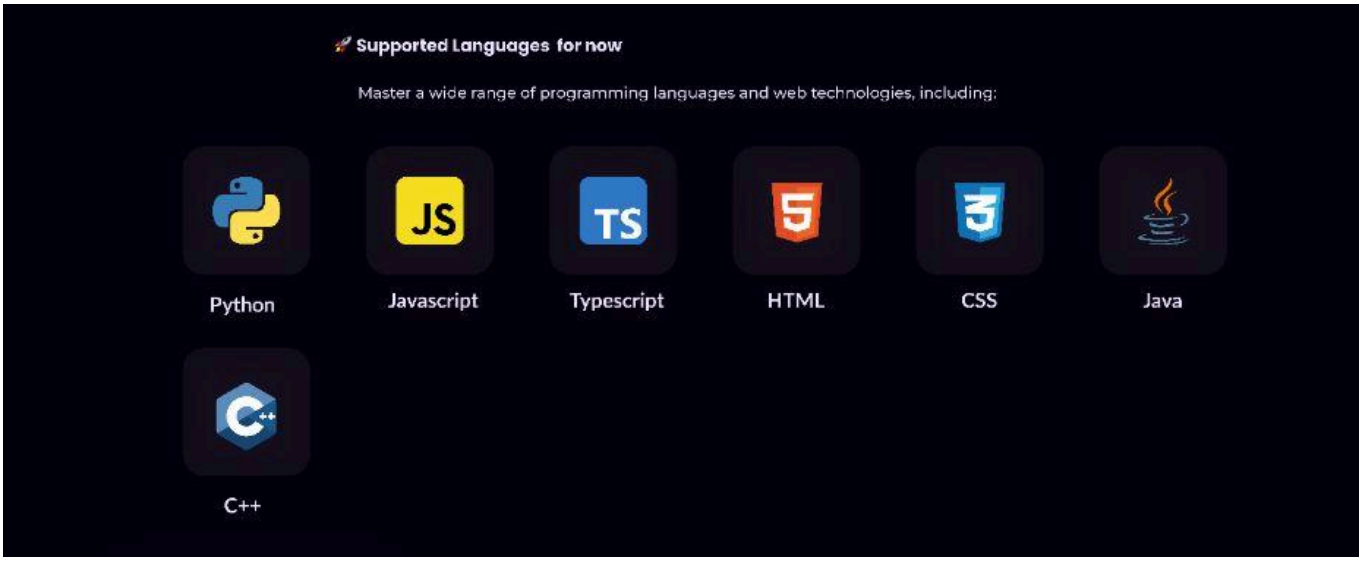
Landing Page (Desktop - Dark)

1. **Home Section**

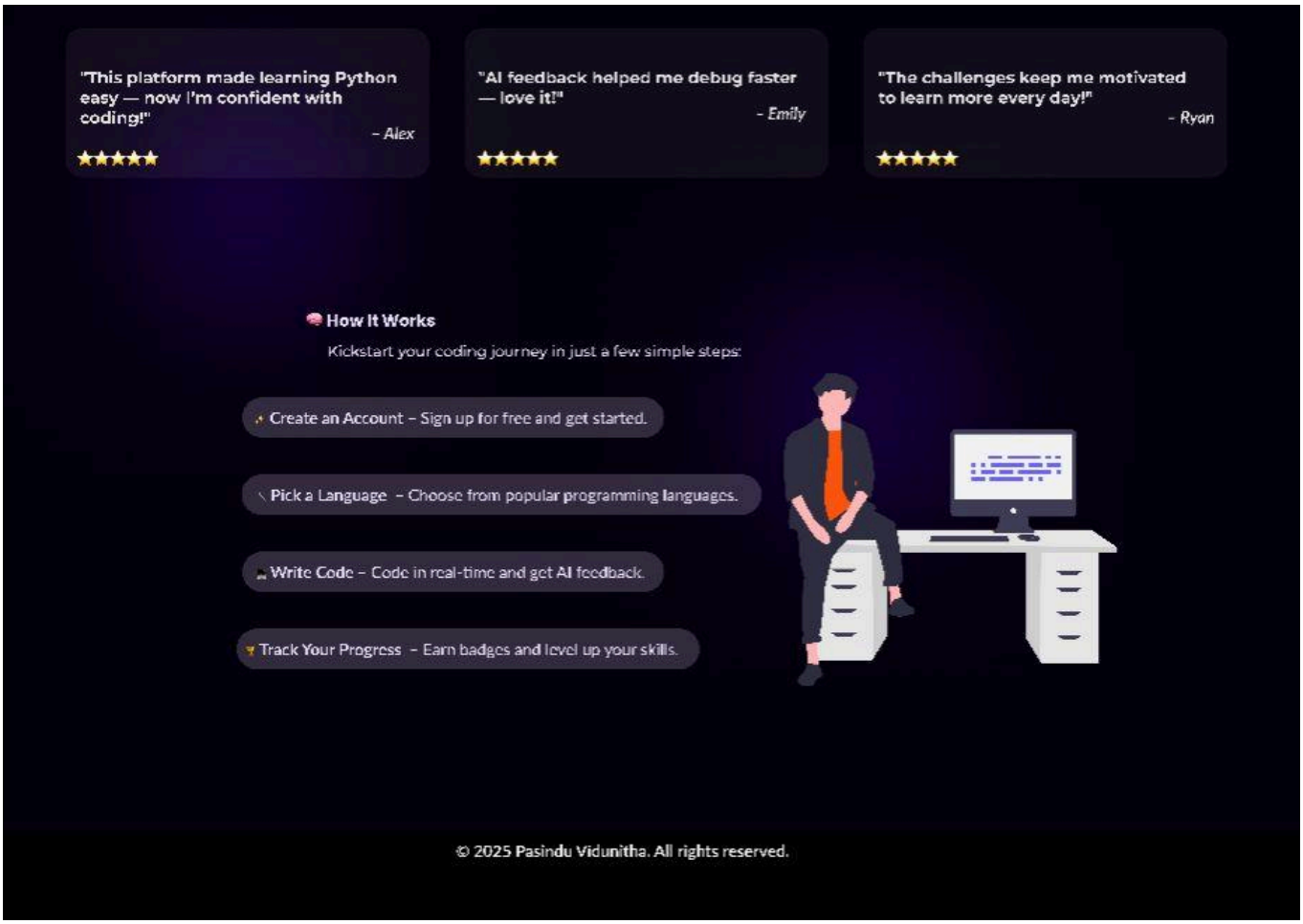


2. **Why Us Section**

3. Supported Languages Section

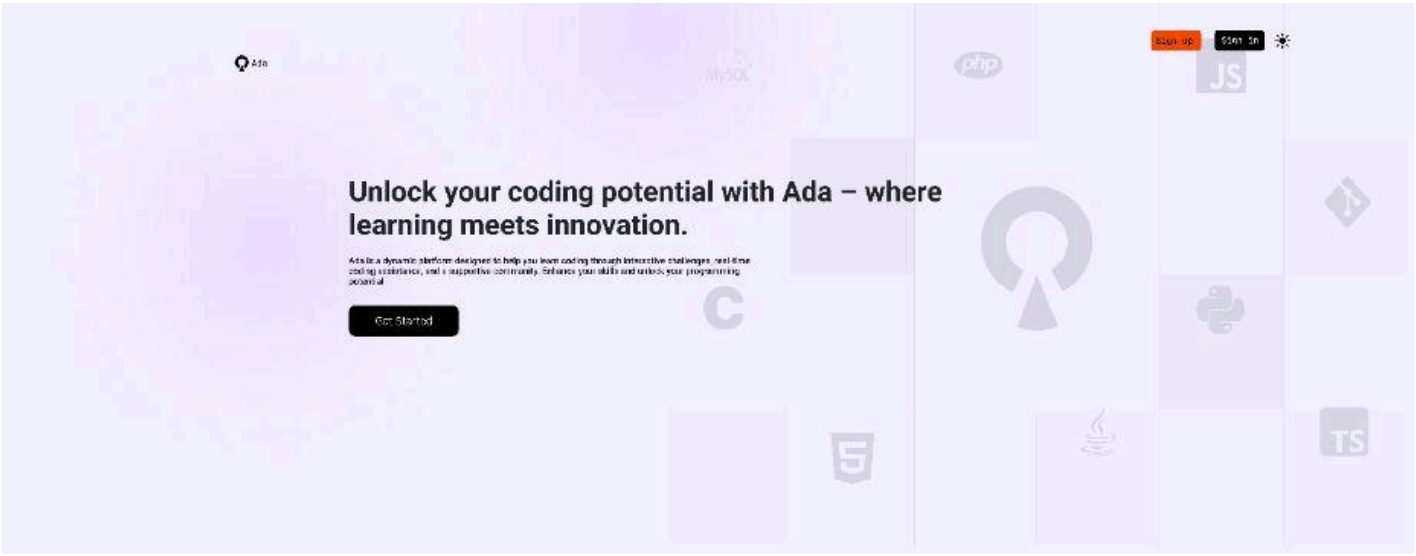


4. How it works Section, comment section, and the footer



Landing Page (Desktop - Light)

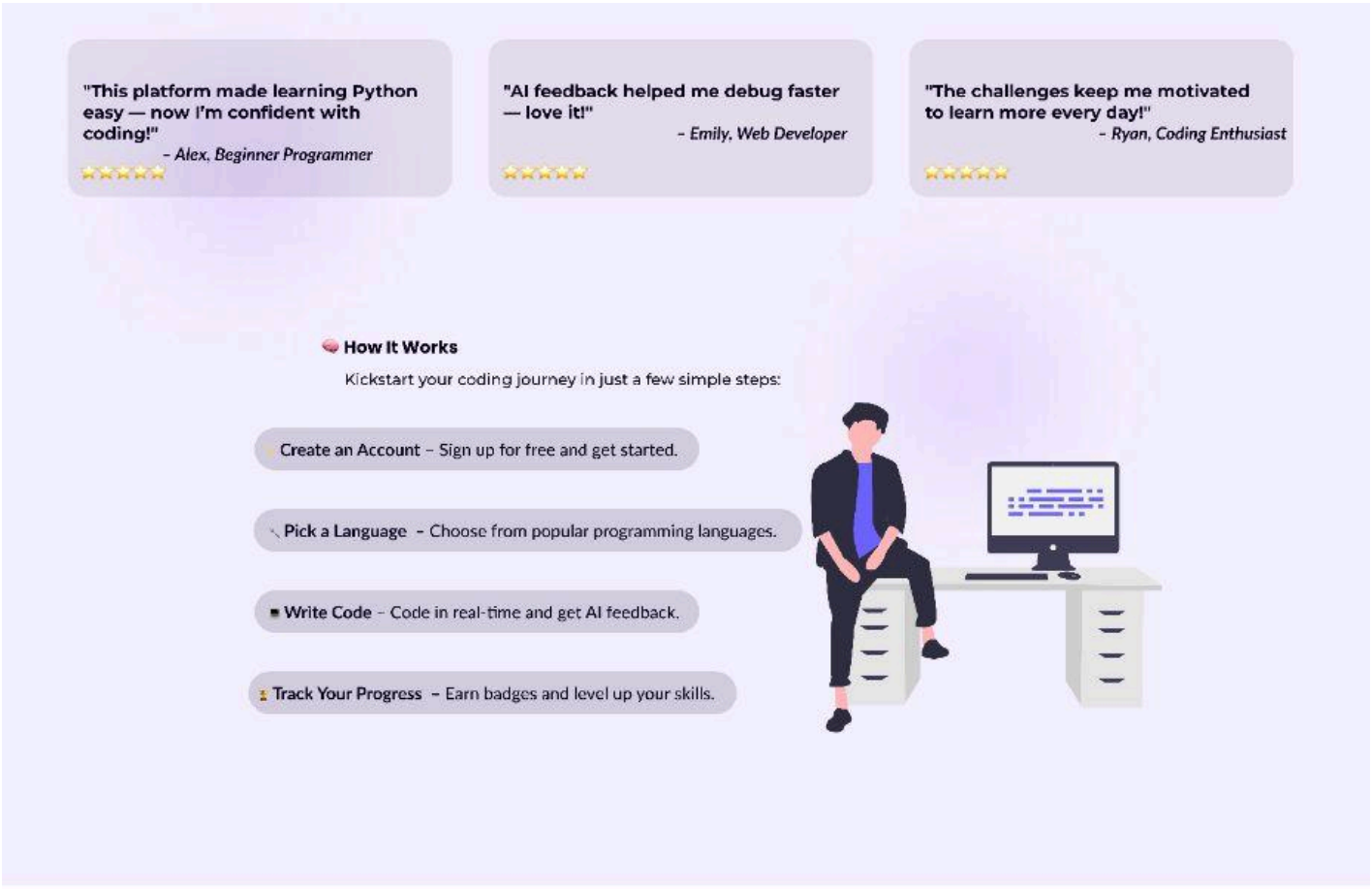
1. Home Section



2. Why Us Section

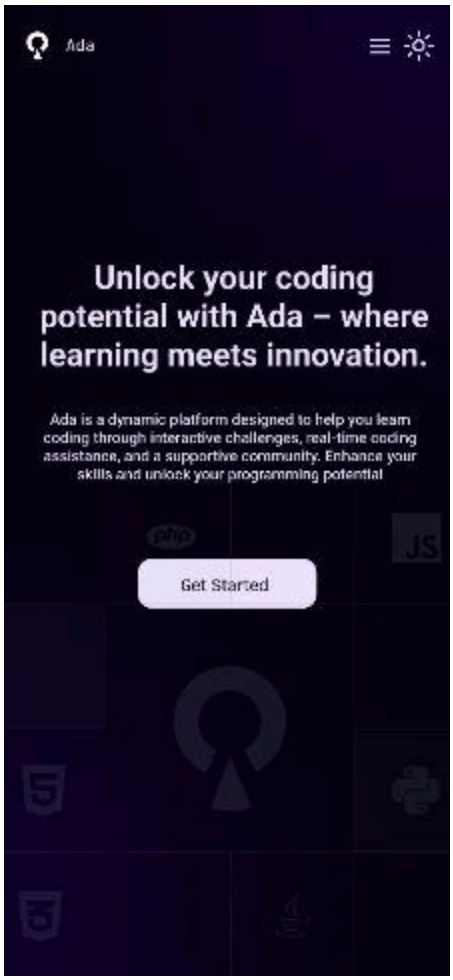
3. Supported Languages Section

4. How it works Section, comment section, and the footer



Landing Page (Mobile)

1. **Home Section**

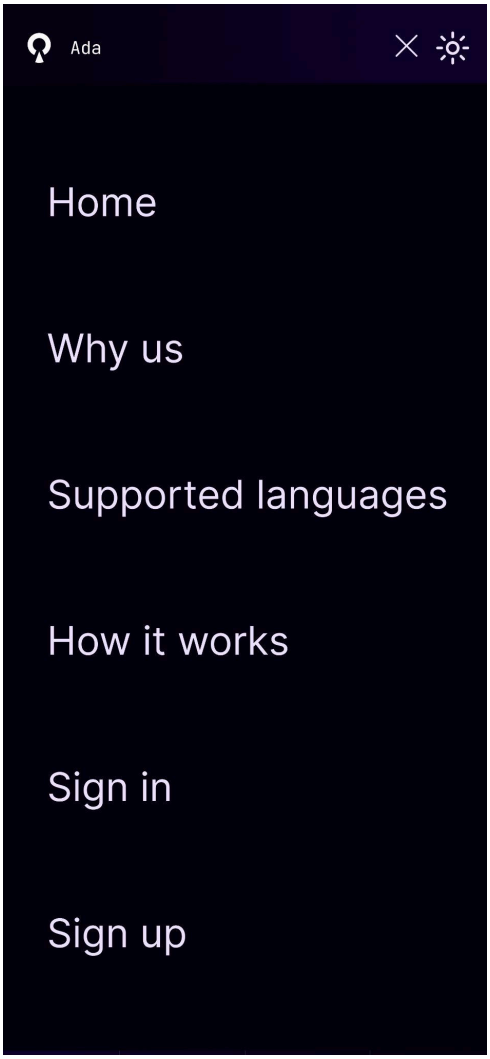


2. **Why Us Section**

3. **Supported Languages Section & comment section**

4. **How it works Section**

5. **Menu Page**

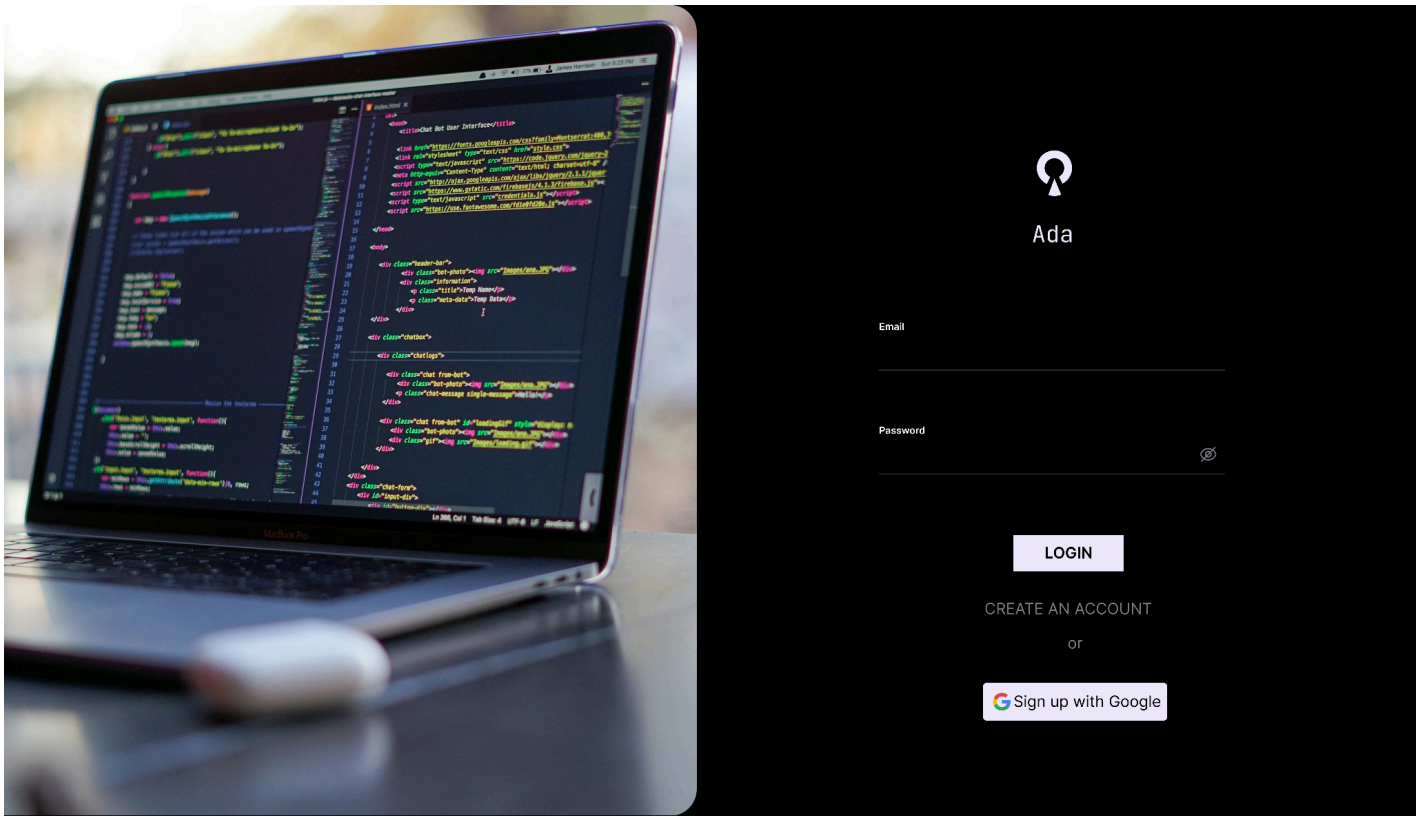


Routes & Function Signatures:

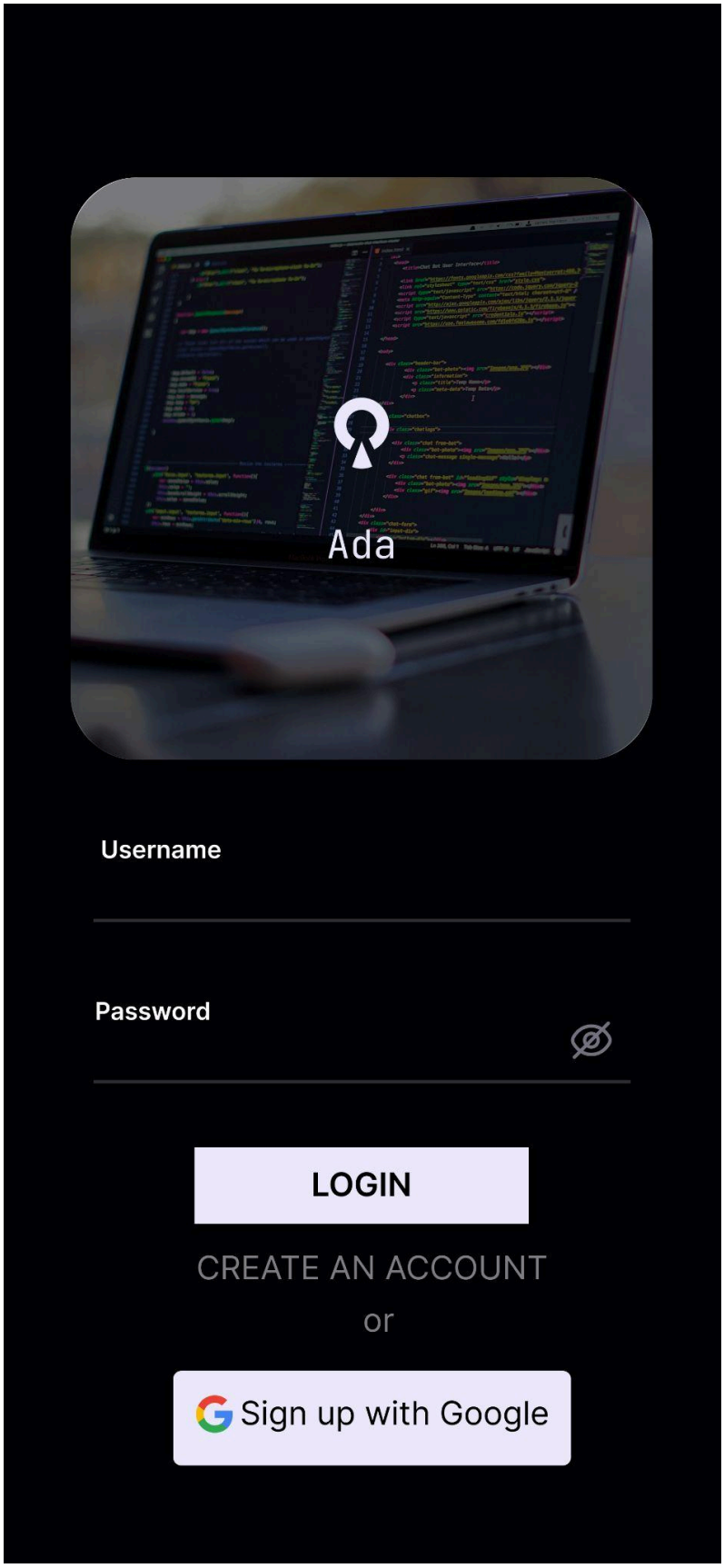
```
@app.route('/')
def landing():
    pass
```

Login Page

Dark theme Desktop UI



Dark theme Mobile UI



Purpose: This allows users to log in with Google or create accounts in the system.

SQL Queries:

- Normal Login

```
SELECT user_id, profile_image, theme_preference FROM User WHERE email= ? AND password
= ? AND auth_provider = "manual"
```

- Log in with a Google account.

```
SELECT user_id, profile_image, theme_preference FROM User WHERE google_id= ? AND
auth_provider = "google"
```

- If the user doesn't exist (This is for those who are using Google accounts)

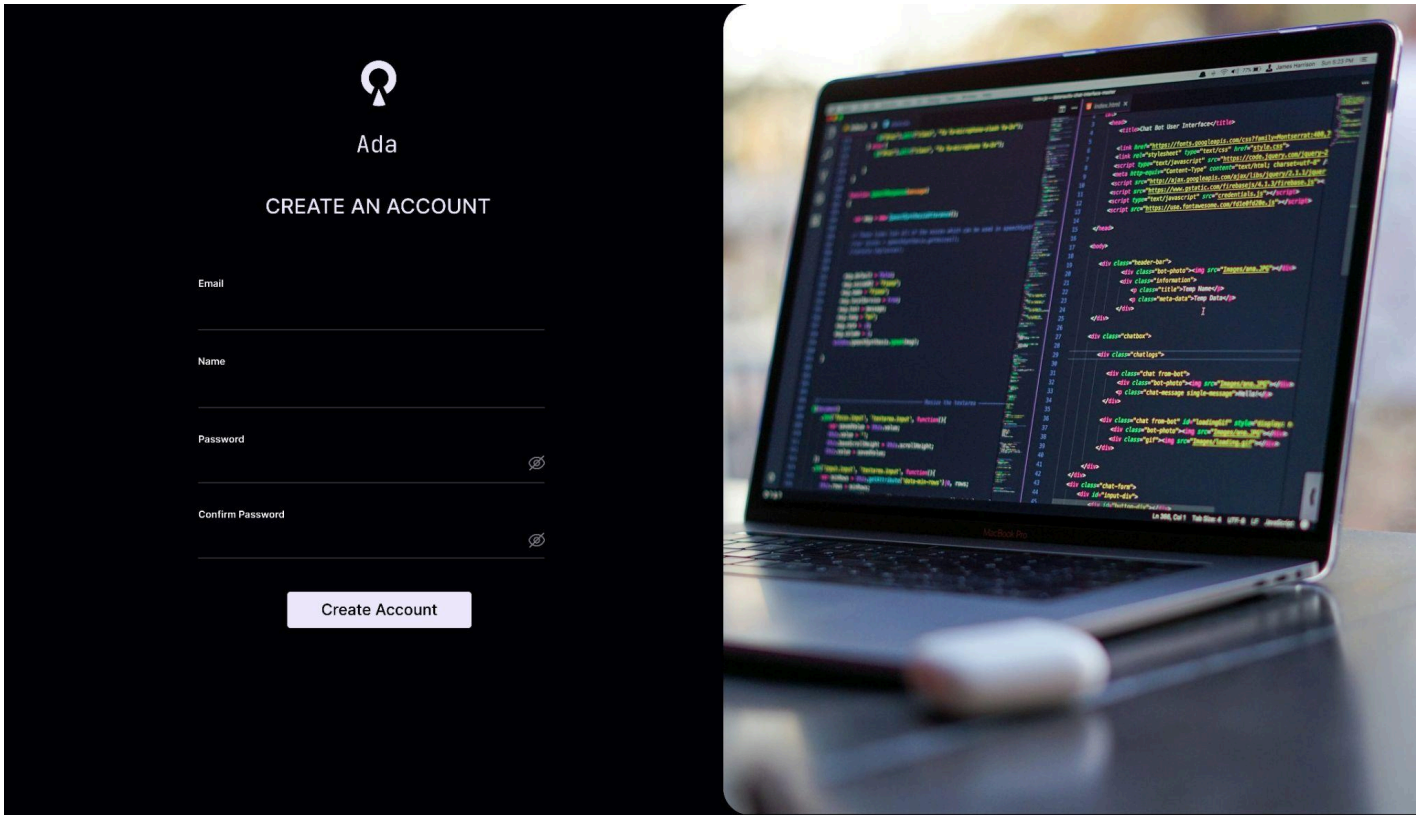
```
INSERT INTO User (user_id, email, full_name, profile_image, auth_provider, google_id,
theme_preference) values (?,?,?,?,"google",?,"dark")
```

Routes & Function Signatures:

```
@app.route('/login', methods=['GET', 'POST'])
def login():
    pass
```

Sign Up page

Dark theme Desktop UI



Purpose: To create new accounts for new users

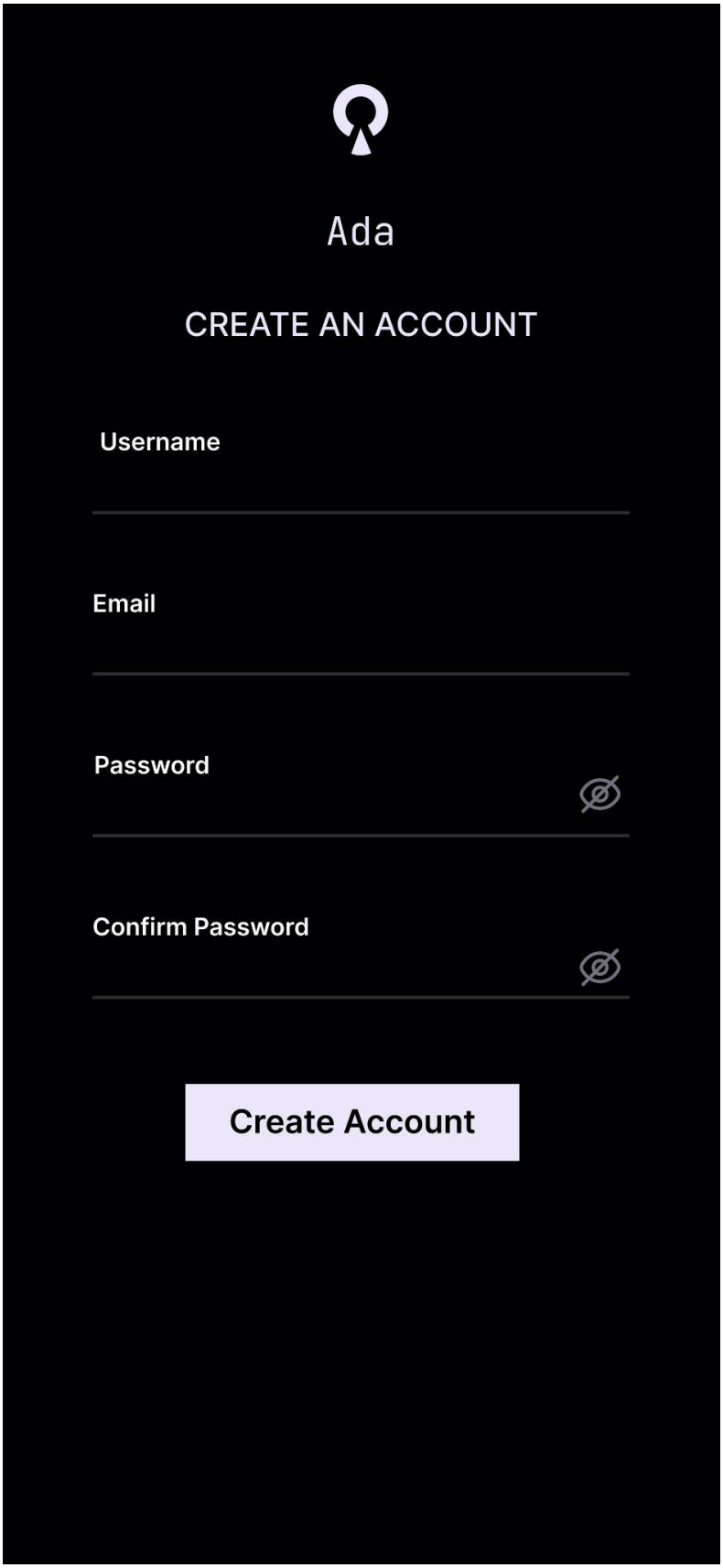
SQL Query:

```
INSERT INTO User (user_id, email, full_name, profile_image, auth_provider, password, theme_preference) VALUES (?, ?, ?, "default_img.png", "manual", ?, "dark")
```

Routes & Function Signatures:

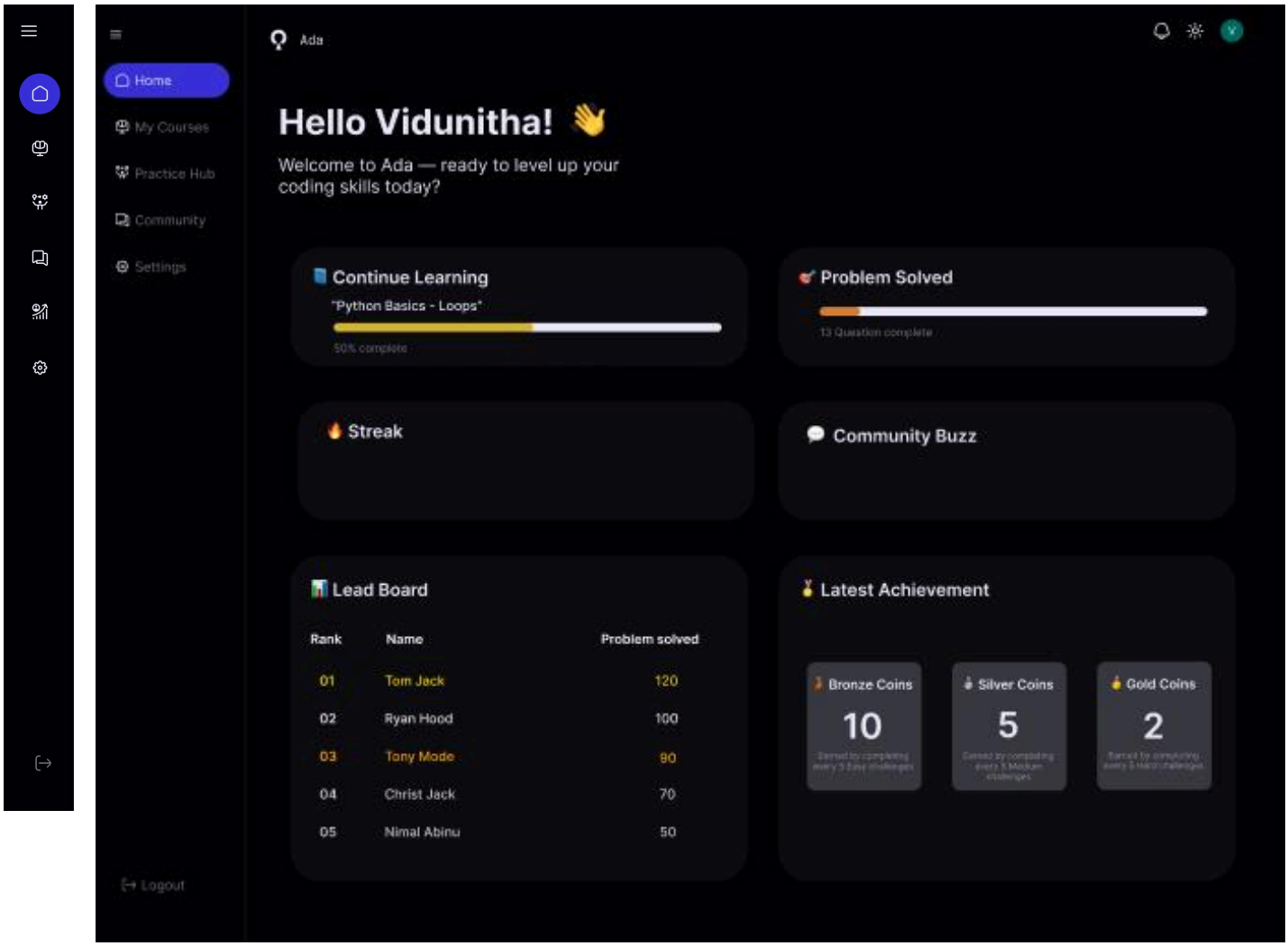
```
@app.route('/signup', methods=['GET', 'POST'])
def signup():
    pass
```

Dark theme Mobile UI



Dashboard

Dark theme Desktop UI



Purpose: To display the User's name, the user's progress and etc.

SQL Queries:

1. User Welcome Section - Get the Full name of the logged user
   (SAVE THE USER_ID, AS WE NEED THAT)

```
SELECT full_name FROM User WHERE user_id = ?;
```

2. Continue Learning Progress - Get the user's current Course and the percentage of it completed

```
SELECT c.course_name,
ROUND(COUNT(ul.lesson_id) * 100.0 /
```

```
(SELECT COUNT(*) FROM Lesson WHERE course_id = c.course_id), 0) AS progress_percent FROM
user_lesson ul
JOIN Lesson l ON ul.lesson_id = l.lesson_id
JOIN Course c ON l.course_id = c.course_id WHERE ul.user_id = ?
GROUP BY c.course_id ORDER BY ul.completed_at DESC LIMIT 1;
```

3. Count the number of challenges that have cobeen mpleted by the user

```
SELECT COUNT(*) AS problems_solved FROM Challenge_attempt WHERE user_id = ?;
```

4. Streak Tracking - Count the challenges completed per day

```
SELECT DATE(completed_at) AS day, COUNT(*) AS challenges_done FROM Challenge_attempt
WHERE user_id = ? GROUP BY DATE(completed_at) ORDER BY DATE(completed_at) DESC;
```

5. Community Buizz - Retrieve the most recent question asked in the community.

```
SELECT question FROM Chat ORDER BY created_at DESC LIMIT 1;
```

6. Display the top 5 users with the highest number of solved challenges.

```
SELECT u.full_name, COUNT(ca.id) AS problems_solved FROM Challenge_attempt AS ca
JOIN User u ON ca.user_id = u.user_id GROUP BY ca.user_id ORDER BY problems_solved DESC
LIMIT 5;
```

7. Coin Rewards (Based on difficulty levels in Challenges)
   Bronze Coins (Every 5 Easy Challenges):

```
SELECT COUNT(*) / 5 AS bronze_coins FROM Challenge_attempt  AS ca JOIN Challenge c ON
ca.challenge_id = c.challenge_id WHERE ca.user_id = ? AND c.difficulty_level = 'easy';
```

Silver Coins (Every 5 Medium Challenges):

```
SELECT COUNT(*) / 5 AS silver_coins FROM Challenge_attempt AS ca JOIN Challenge c ON
ca.challenge_id = c.challenge_id WHERE ca.user_id = ? AND c.difficulty_level = 'medium';
```

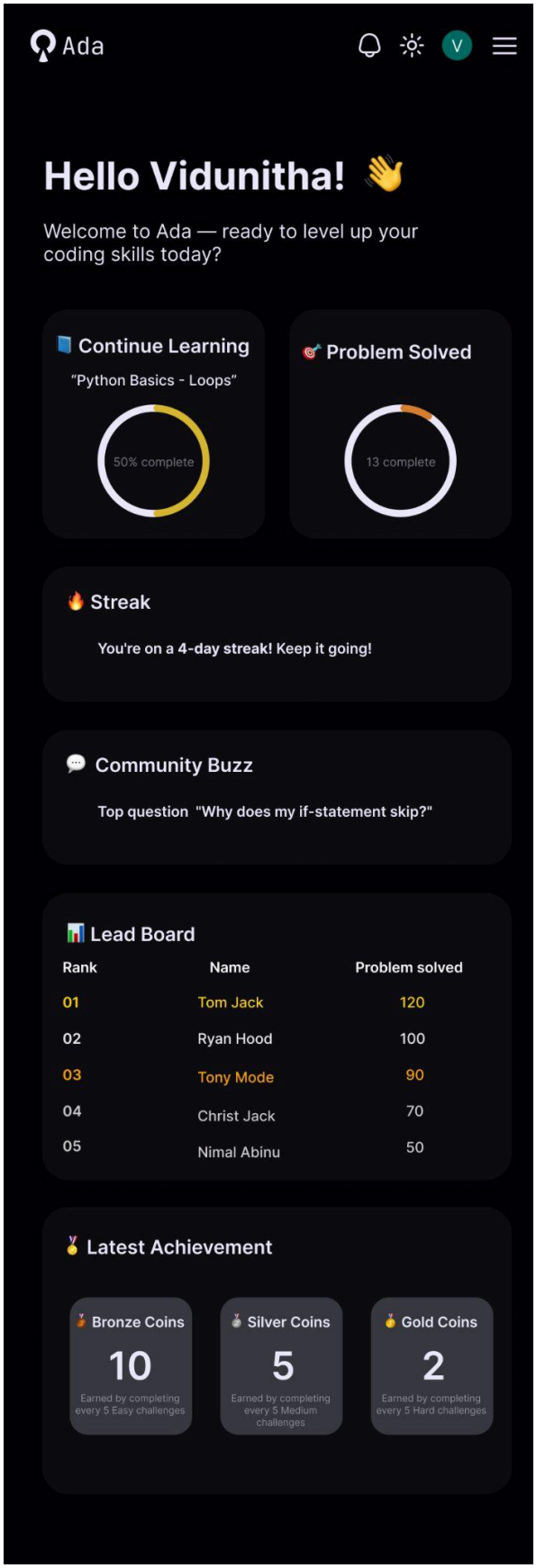
Gold Coins (Every 5 Hard Challenges):

```
SELECT COUNT(*) / 5 AS gold_coins FROM Challenge_attempt AS ca JOIN Challenge c ON
ca.challenge_id = c.challenge_id WHERE ca.user_id = ? AND c.difficulty_level = 'hard';
```

Routes & Function Signatures:

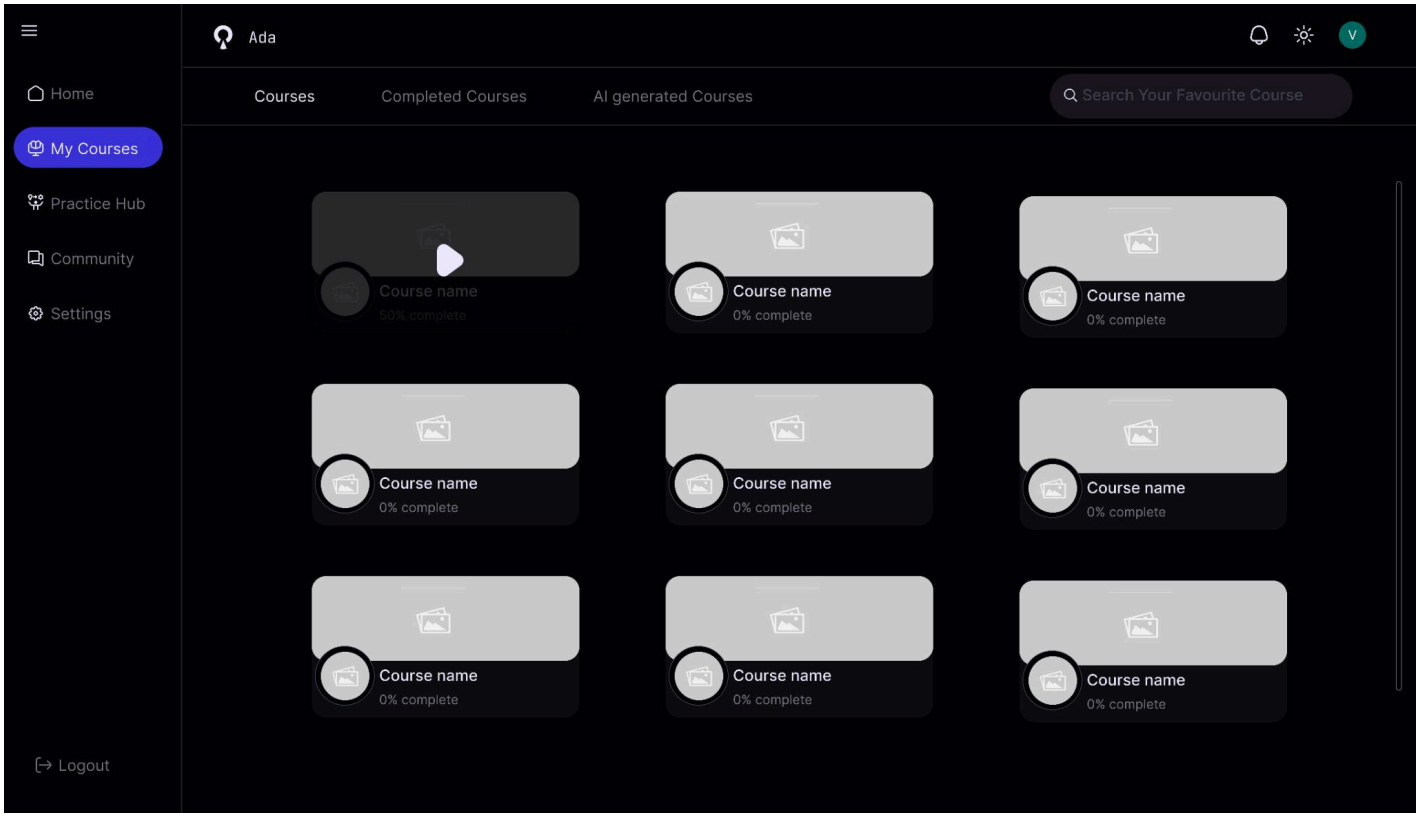
```
@app.route('/dashboard')
def dashboard():
    pass
```

Dark theme Mobile UI



Course Page

Dark theme Desktop UI



Purpose: To show the courses that users haven't started, and have started but not completed.

User Interaction: In the desktop version, when the user hovers over a course, it will change to the first course card.
All the course details are passed in JSON format to the frontend, and when the user clicks on a specific course, Relevant JSON data is sent to the backend, saved for future use.

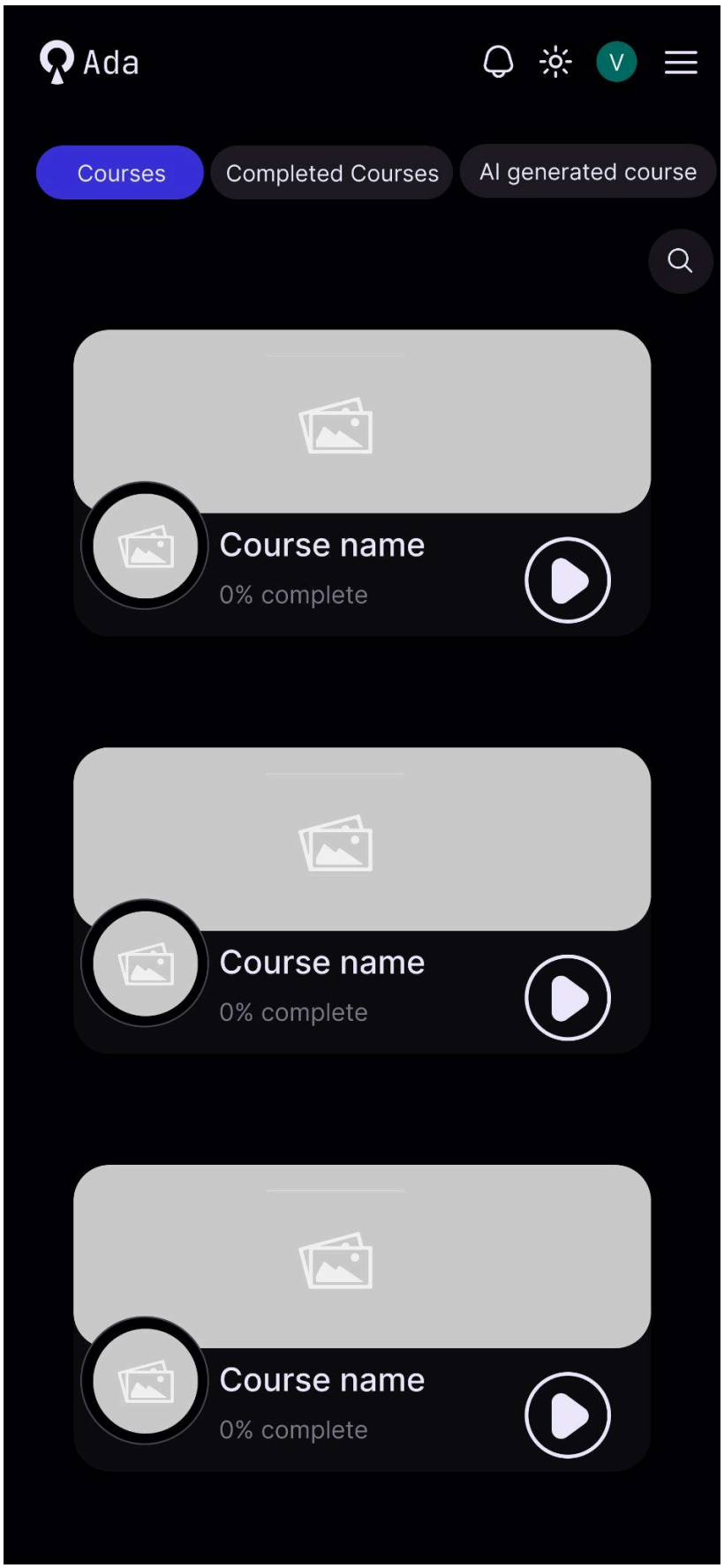
SQL Query:

```
SELECT c.course_id, c.course_name, c.language_image, c.course_image,
  ROUND(COUNT(ul.lesson_id) * 100.0 /
        (SELECT COUNT(*) FROM Lesson WHERE course_id = c.course_id), 0) AS
progress_percent FROM Enrollment AS e
JOIN Course AS c ON e.course_id = c.course_id
LEFT JOIN Lesson l ON c.course_id = l.course_id
LEFT JOIN user_lesson ul ON ul.lesson_id = l.lesson_id AND ul.user_id = e.user_id
WHERE e.user_id = ?
GROUP BY c.course_id HAVING progress_percent < 100;
```

Routes & Function Signatures:

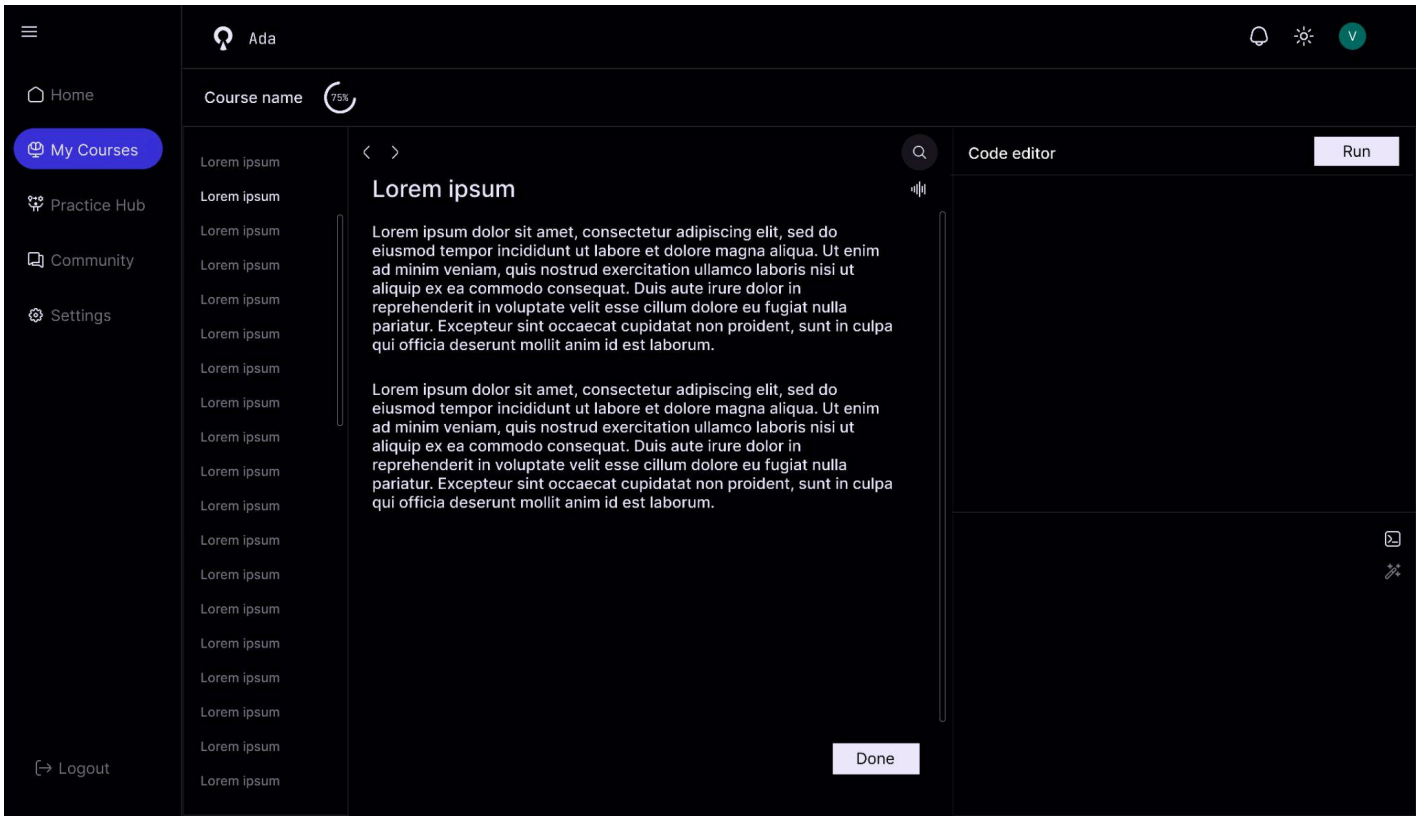
```
@app.route('/my_courses')
def courses():
    pass
```

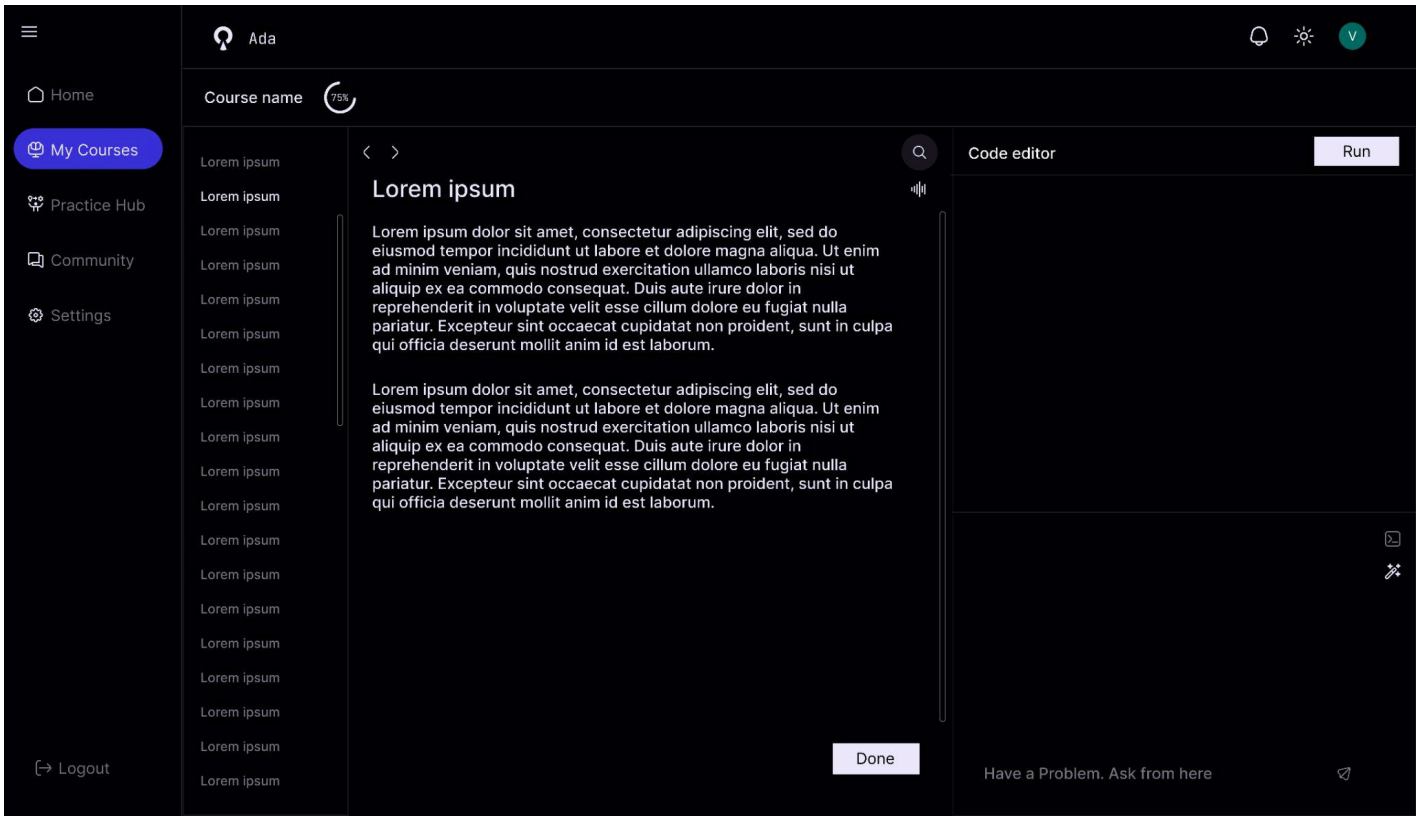
Dark theme Mobile UI



Lesson Page

Dark theme Desktop UI





Purpose: This page shows the lessons which are in related to the course that is selected by the user.

User Interactions:

- On the bottom right side, there is a box where the user can switch from AI assistance to a code terminal.
- When the user clicks on the done button, it is marked as "completed" in the *lesson table*.
- "Run" button to run the code that is typed by the user.
- When the user clicks on the "sound wave" button, the user will be able to hear the text that is displayed on the page.
- The search button is used to type keywords to find things in the text.
- In the left corner menu, which is between the text and the main menu, the user will be able to see all the lessons that are related to the selected course.
- From the back and front arrows, the user can go to the next and previous lessons.

SQL Queries:

1. Get all the lessons related to the selected course

```
SELECT l.lesson_id, l.lesson_title, l.content FROM Lesson AS l WHERE l.course_id = ?
ORDER BY l.lesson_order;
```

2. Display the selected lesson title and the content.

```
SELECT l.lesson_title, l.lesson_content FROM lesson as l WHERE l.lesson_id = ?;
```

   Frontend receives all lesson details in JSON format → User clicks on a specific lesson → The selected lesson ID is sent to the backend → Backend retrieves the lesson and stores it in a variable for further use.

3. Mark as completed when the user clicks on the "Done" Button

```
UPDATE user_lesson SET status = 'completed', completed_at =  CURRENT_TIMESTAMP WHERE
user_id = ? AND lesson_id = ?;
```

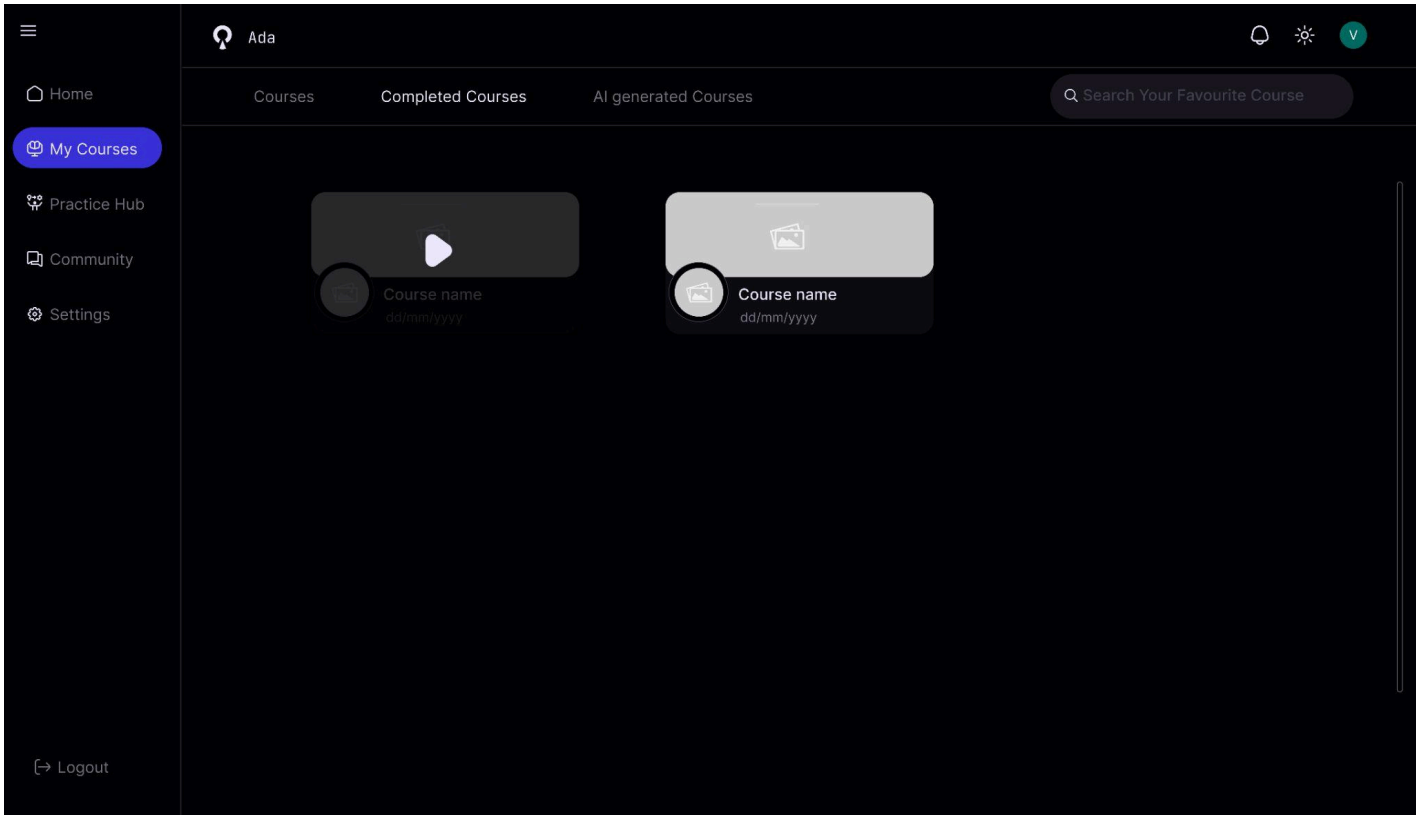
HINT: Don't use a separate table to save AI chat; save it in local storage in the browser

Routes & Function Signatures:

```
@app.route('/my_courses/<course_id>/lesson/<lesson_id>')
def view_lesson(course_id, lesson_id):
    pass
```

Completed Course Page

Dark theme Desktop UI



Purpose: To show courses that have been completed by the user; if they like, they can do those courses again. It shows the course title and the completion date.

User Interactions:

- In the desktop version, when the user hovers over a course, it will change to the first course card.

SQL Query:

```
SELECT c.course_id , c.course_name, e.completed_at FROM Enrollment AS e
JOIN Course c ON e.course_id = c.course_id WHERE e.user_id = ? AND e.completed_at IS
NOT NULL ORDER BY e.completed_at DESC;
```

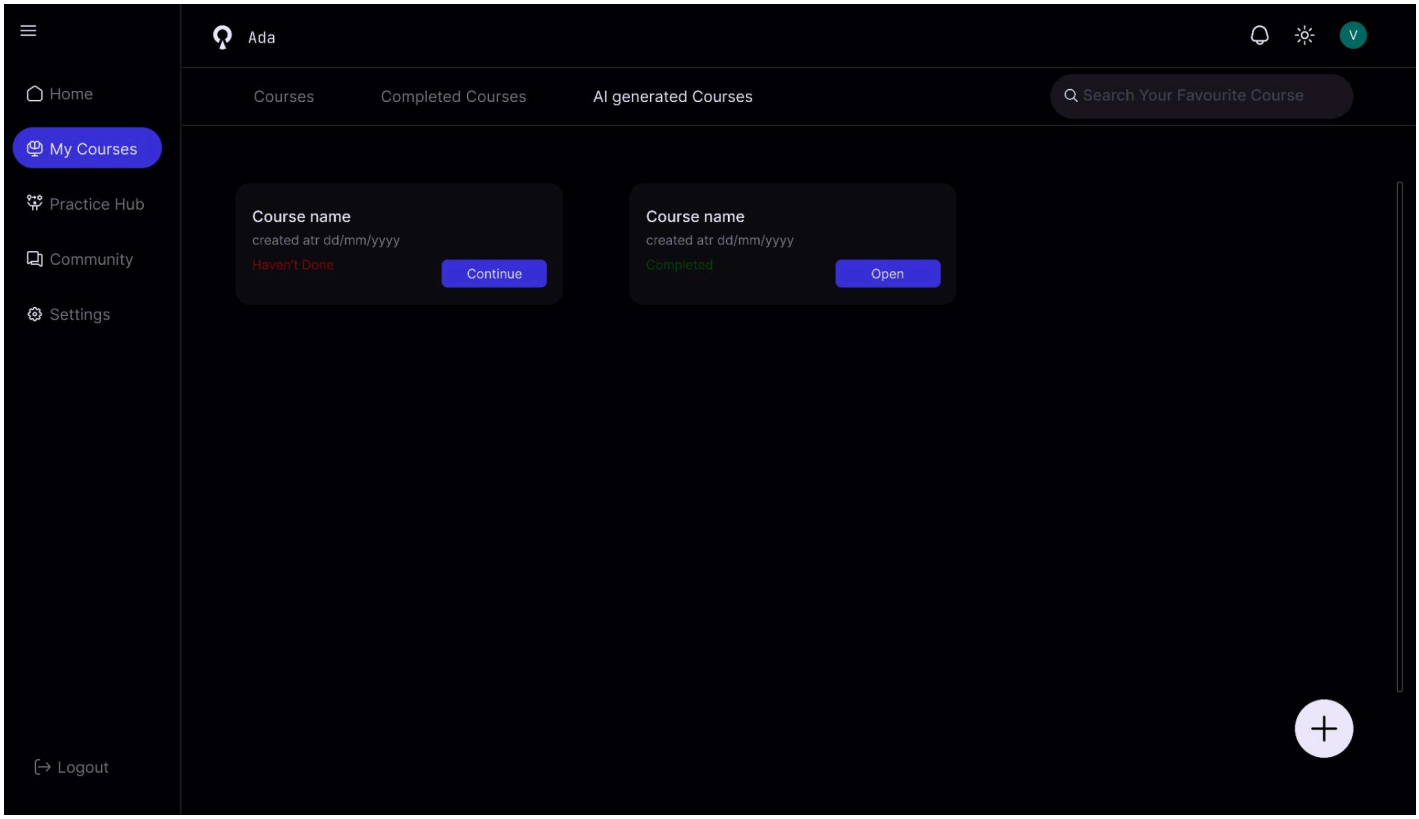
NOTE: The completed_at field is automatically updated in the backend once all the lessons are completed by the users in a course

Routes & Function Signatures:

```
@app.route('/my_courses/completed')
def completed_courses():
    pass
```

AI Genrated Course Page

Dark theme Desktop UI



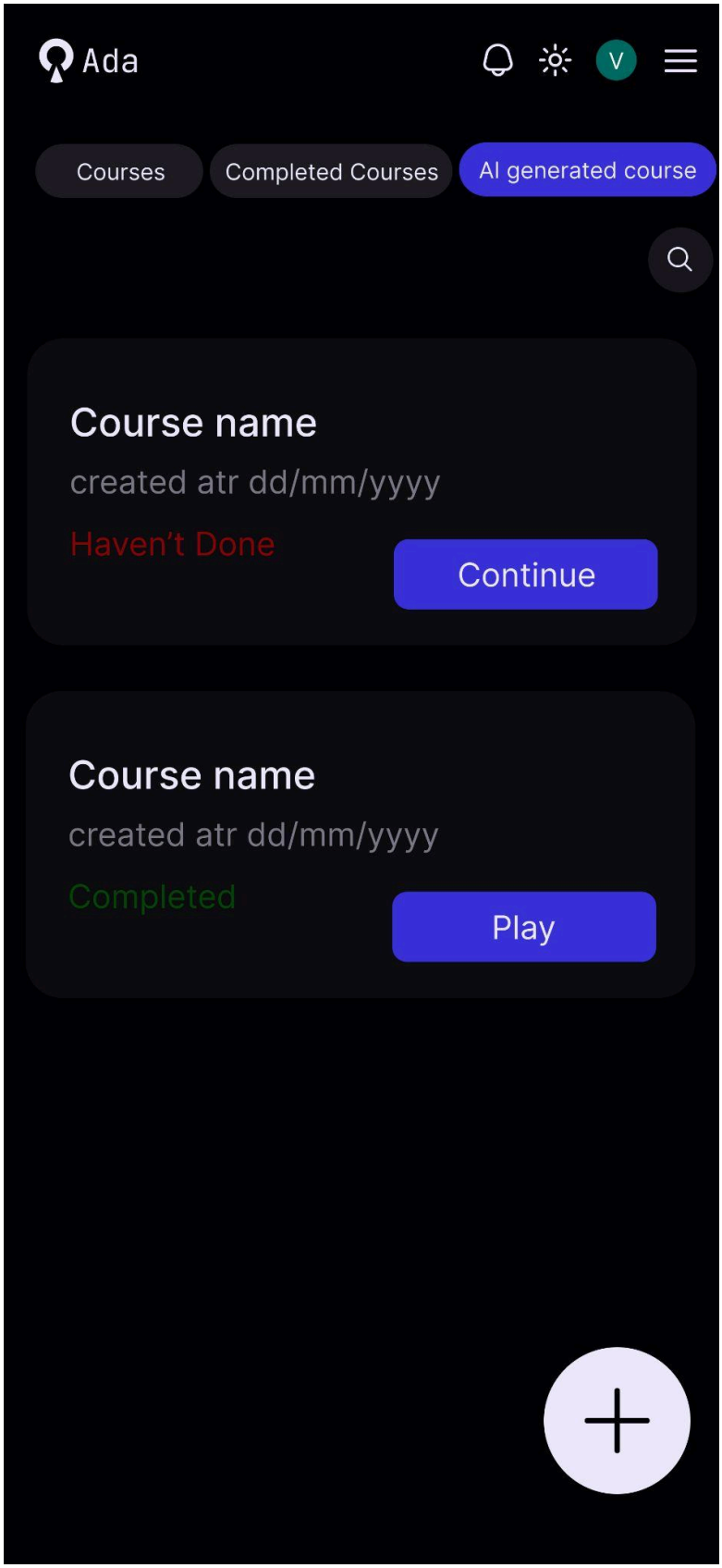
Purpose: To show AI-generated courses that are created by users
User Interactions: All the AI courses' details are passed in JSON format to the frontend, and when the user clicks on a specific course, Relevant JSON data is sent to the backend, saved for future use.
SQL Query:

```
SELECT ai.title, ai.user_id , status, generated_at FROM Ai_resource AS ai WHERE
ai.user_id = ?;
```

Routes & Function Signatures:

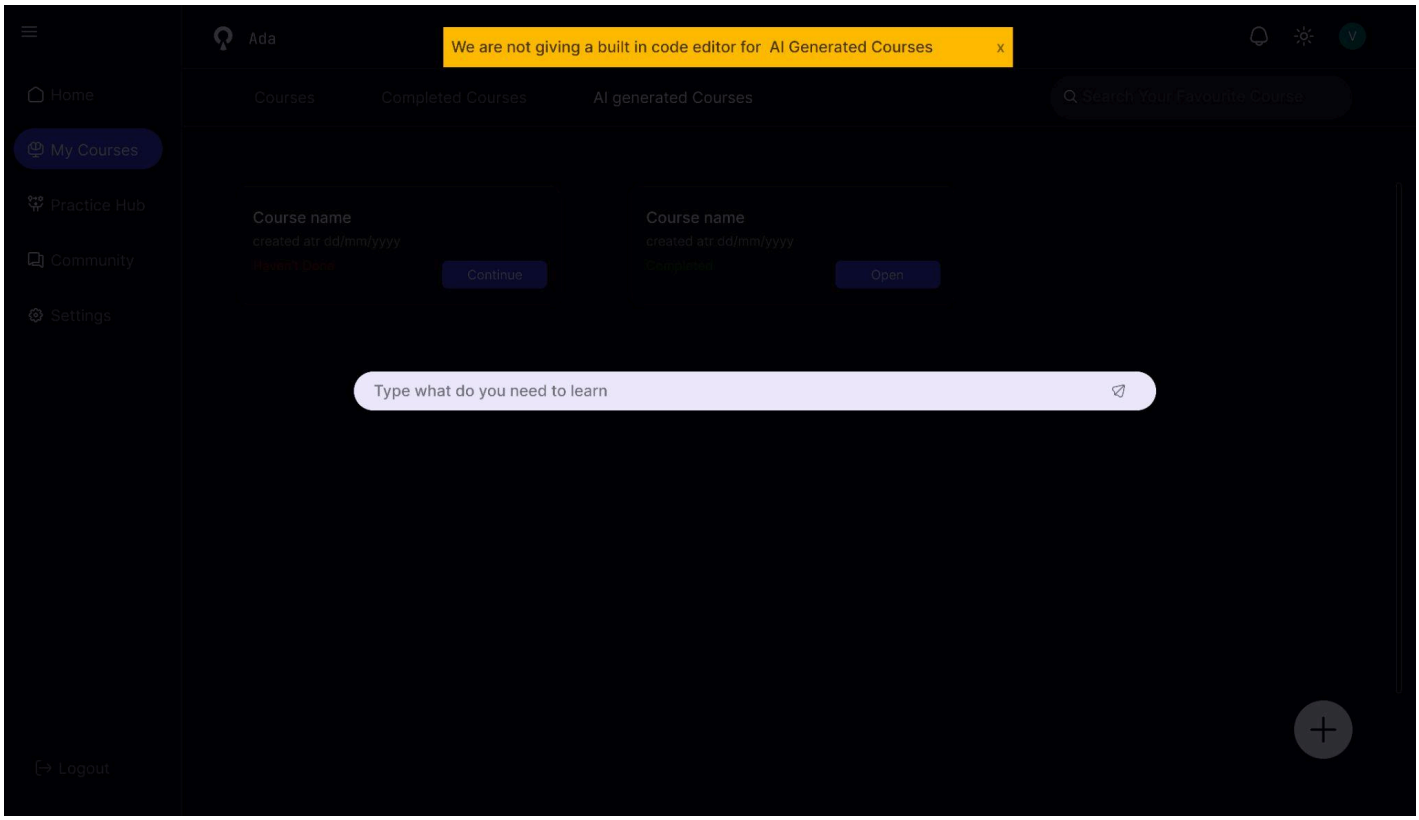
```
@app.route('/my_courses/ai_generated')
def ai_courses():
    pass
```

Dark theme Mobile UI



AI Course Request Panel– Pop-up window

Dark theme Desktop UI



Purpose: Users can request new courses from the AI

User Interactions:

- When the user clicks on the plus button text input box appears, and the rest of the part will become darker to highlight the text input box, and the user can go back to the normal screen by clicking the background.
- The user input passes to the third-party AI, and it generates the title and the course and saves it into the database.

SQL Query:

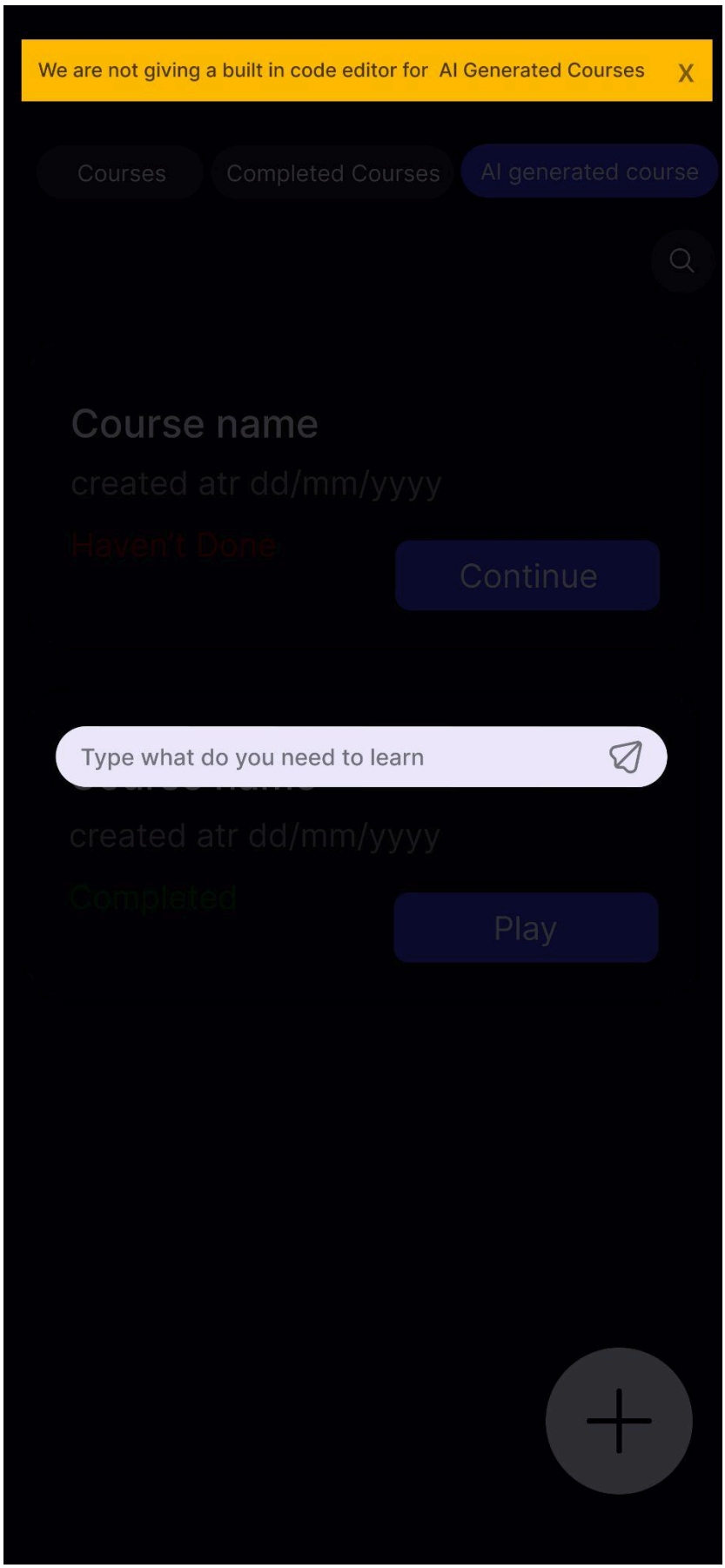
1. Insert a new AI-generated course

```
INSERT INTO Ai_resource ( resource_id, user_id, title, content generated_at, status )
VALUES (?, ?, ?, ?, CURRENT_TIMESTAMP, "not_completed");
```

Routes & Function Signatures:

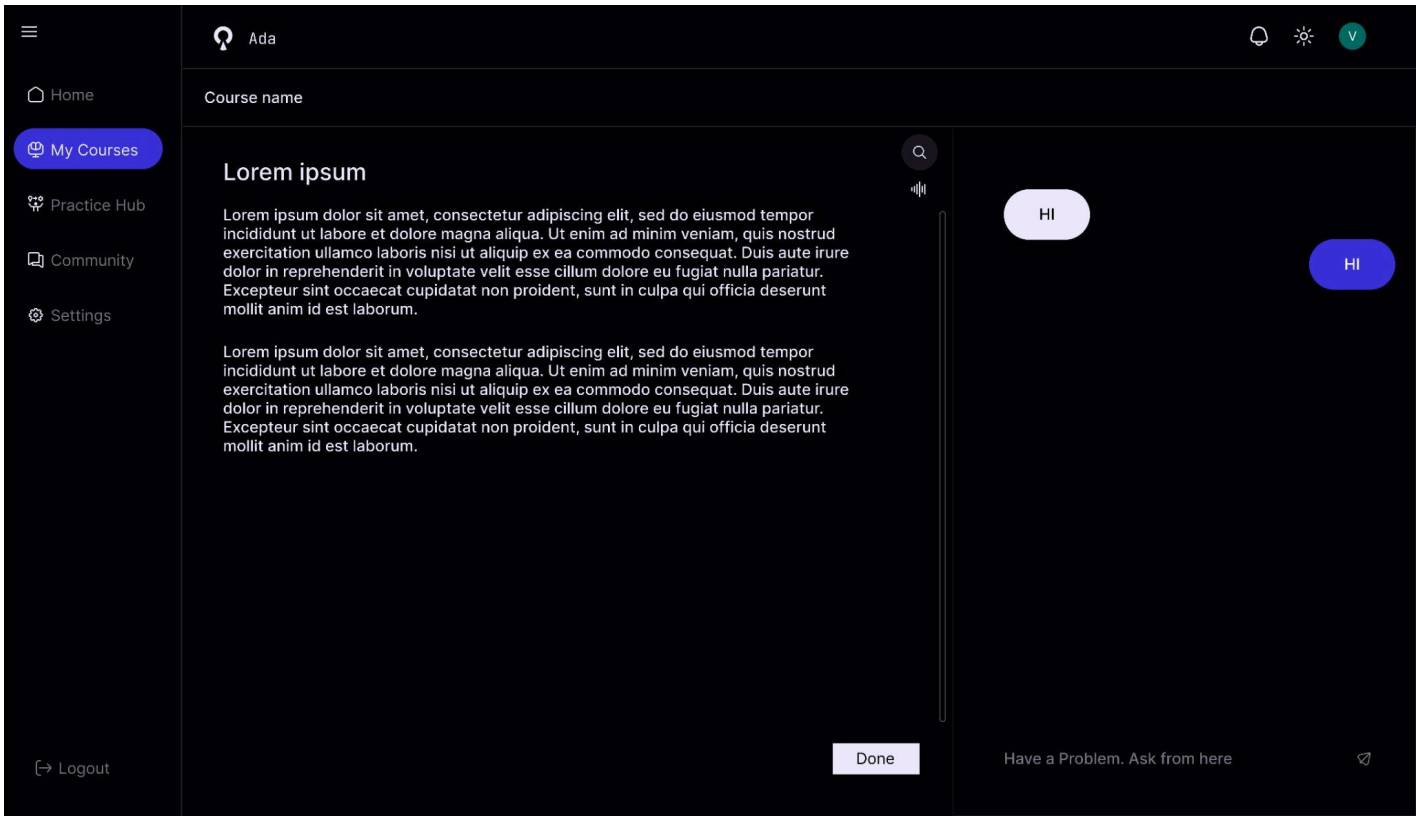
```
@app.route('/my_courses/ai_generated')
def ai_courses():
    pass
```

Dark theme Mobile UI



AI Courses Content Page

Dark theme Desktop UI



Purpose: This page shows AI-generated course content, but no built-in code editor for this, On the left side, the user can chat with the AI assistant

User Interactions:

- When the user clicks on the done button, it is marked as "completed" in the *lesson table*.
- When the user clicks on the "sound wave" button, the user will be able to hear the text that is displayed on the page.
- The search button is used to type keywords to find things in the text.

NOTE: NO NEED TO SAVE USER AND AI CHAT IN A TABLE, SAVE IT IN LOCAL STORAGE

SQL Queries:

1. Get all the challenge details

```
SELECT title, content FROM Ai_resource WHERE user_id = ? AND resource_id = ?;
```

2. When the user clicks on the "Done" button

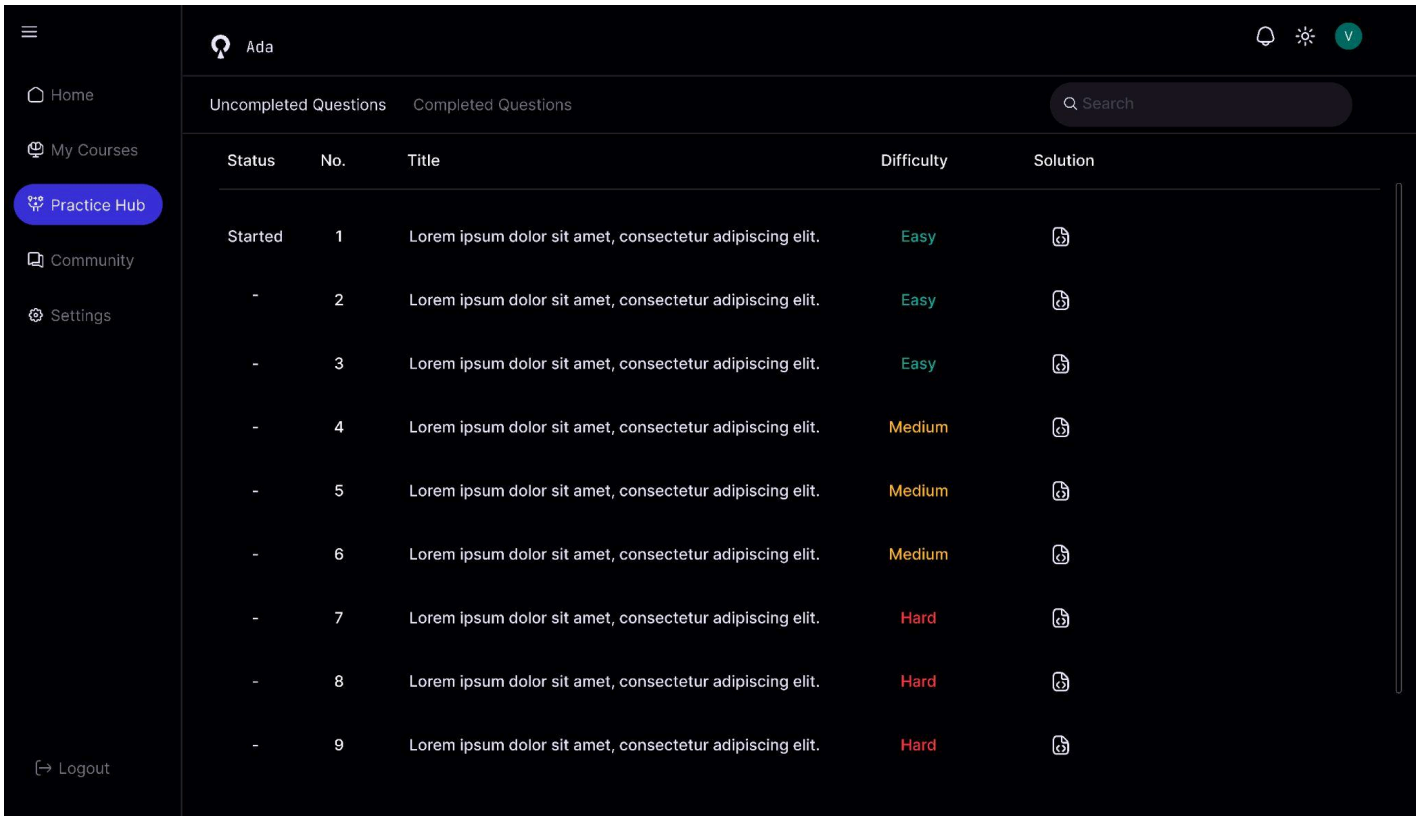
```
UPDATE Ai_resource SET status = 'completed' WHERE user_id = ? AND resource_id = ?;
```

Routes & Function Signatures:

```
@app.route('/my_courses/ai_generated/<resource_id>')
def ai_course_detail(resource_id):
    pass
```

Uncompleted Challenges Page

Dark theme Desktop UI



Purpose: This display displays all the challenges that the user hasn't started, and has started but hasn't completed

User Interactions:

- Users can click on those questions, a new window will open
- Users can click on the solution icon to see the solution for the relevant challenge, which pops up a screen with the solution

SQL Query:

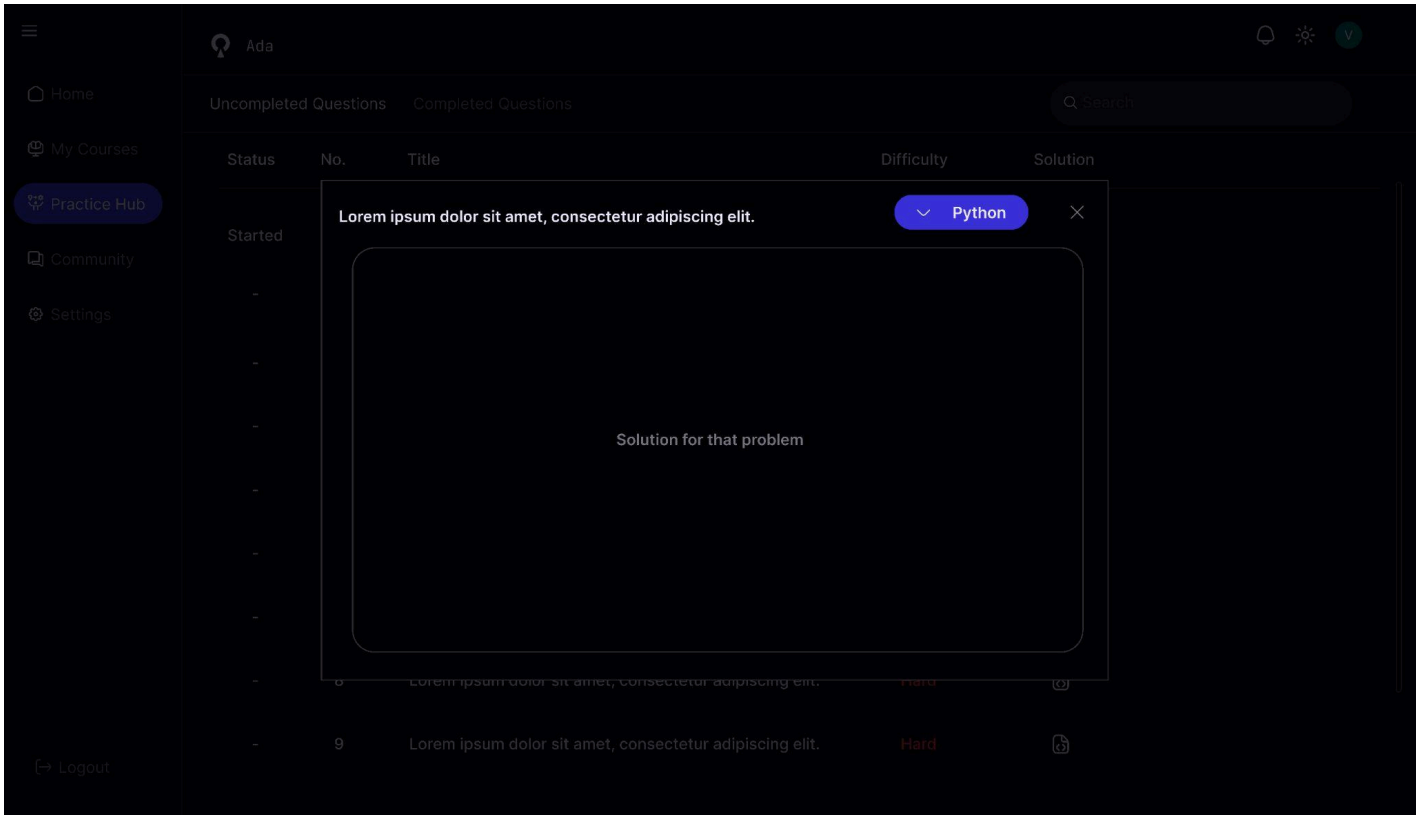
```
SELECT ch.question, ch.status, ch.difficulty_level, ch.challenge_id, ch.number FROM
Challenge  AS ch WHERE ch.user_id = ?;
```

Routes & Function Signatures:

```
@app.route('/practice_hub/uncompleted')
def uncompleted_challenges():
    pass
```

Pop-up window in Uncompleted Question Page

Dark theme Desktop UI



Purpose: This is a pop-up window that appears when the user clicks the solution button on a challenge.

User Interactions:

- Users can change the programming language to get the solution according to their needs.
- Users can close the window by clicking the “x” button.

SQL Query:

1. Displays relevant solutions for the selected challenge and the selected language

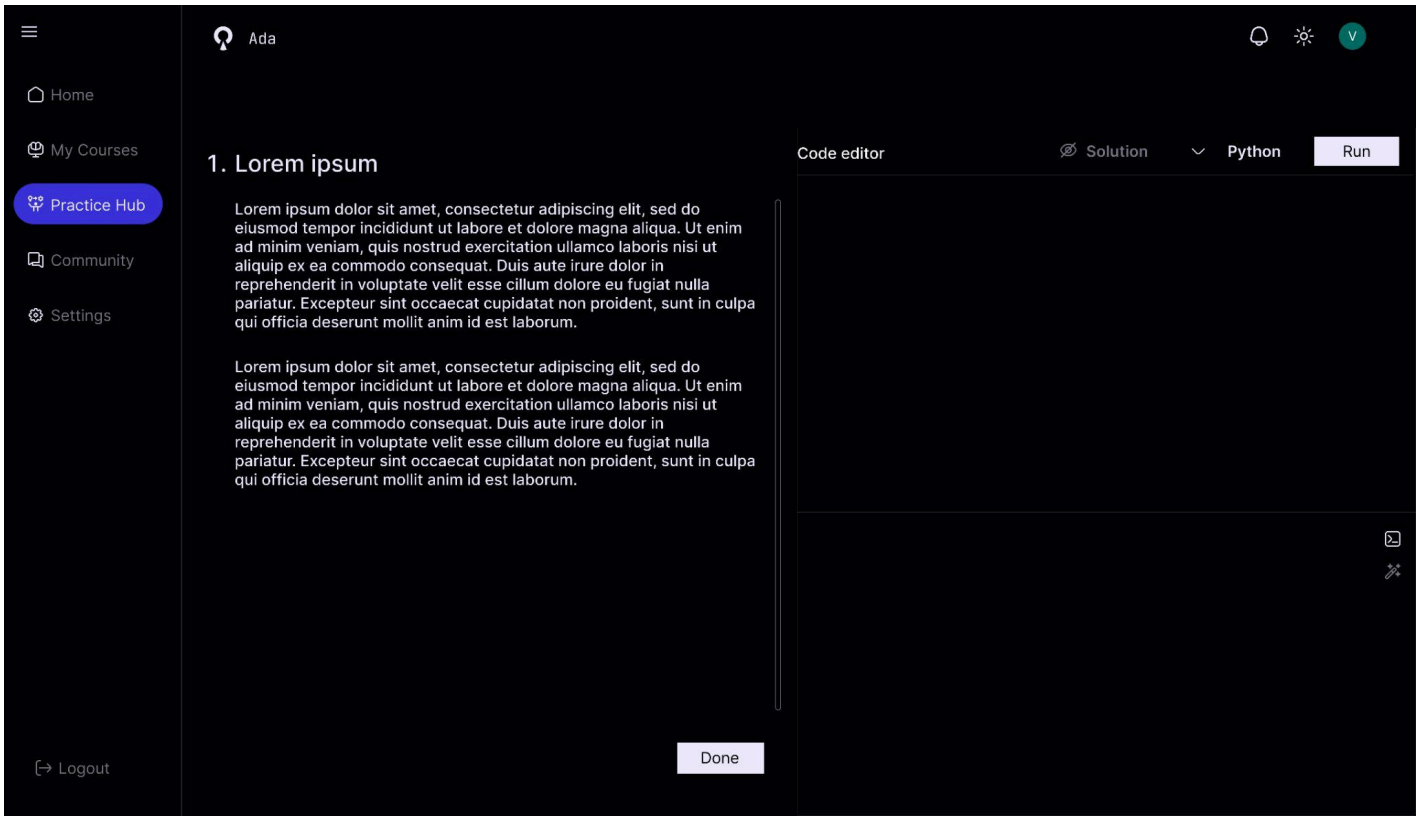
```
SELECT s.answer FROM Solution as s WHERE s.challenge_id = ? AND s.language = ?;
```

Routes & Function Signatures:

```
@app.route('/practice_hub/uncompleted')
def uncompleted_challenges():
    pass
```

Code Editor page for the Challenges

Dark theme Desktop UI



Purpose: This page displays the code editor, the challenge title and its content and the terminal, and the AI assistance similar to *the lesson page*.

User interactions:

- Users can run their code, and they can change their programming languages.
- When the user clicks on the "Done" button, it marks as compelled the challenge if the answer is correct. (Use an AI model to check if the user's answer is correct or not.)
- When the user clicks on the solution button, they can see the solution in a pop-up window similar to the previous page (**Pop-up window in Uncompleted Question Page)**

SQL Queries:

1. Display the Challenge title and the content.

```
SELECT ch.question, ch.challenge_title, ch.number FROM Challenge AS ch WHERE
ch.user_id = ? AND ch.challenge_id = ?;
```

2. Mark as completed the challenge if the answer is correct

```
UPDATE User_challenge SET completed_at = CURRENT_TIMESTAMP WHERE user_id = ? AND
challenge_id = ?;
UPDATE Challenge SET status = 'completed' WHERE challenge_id = ? AND user_id = ?;
```

3. Displays the Solution

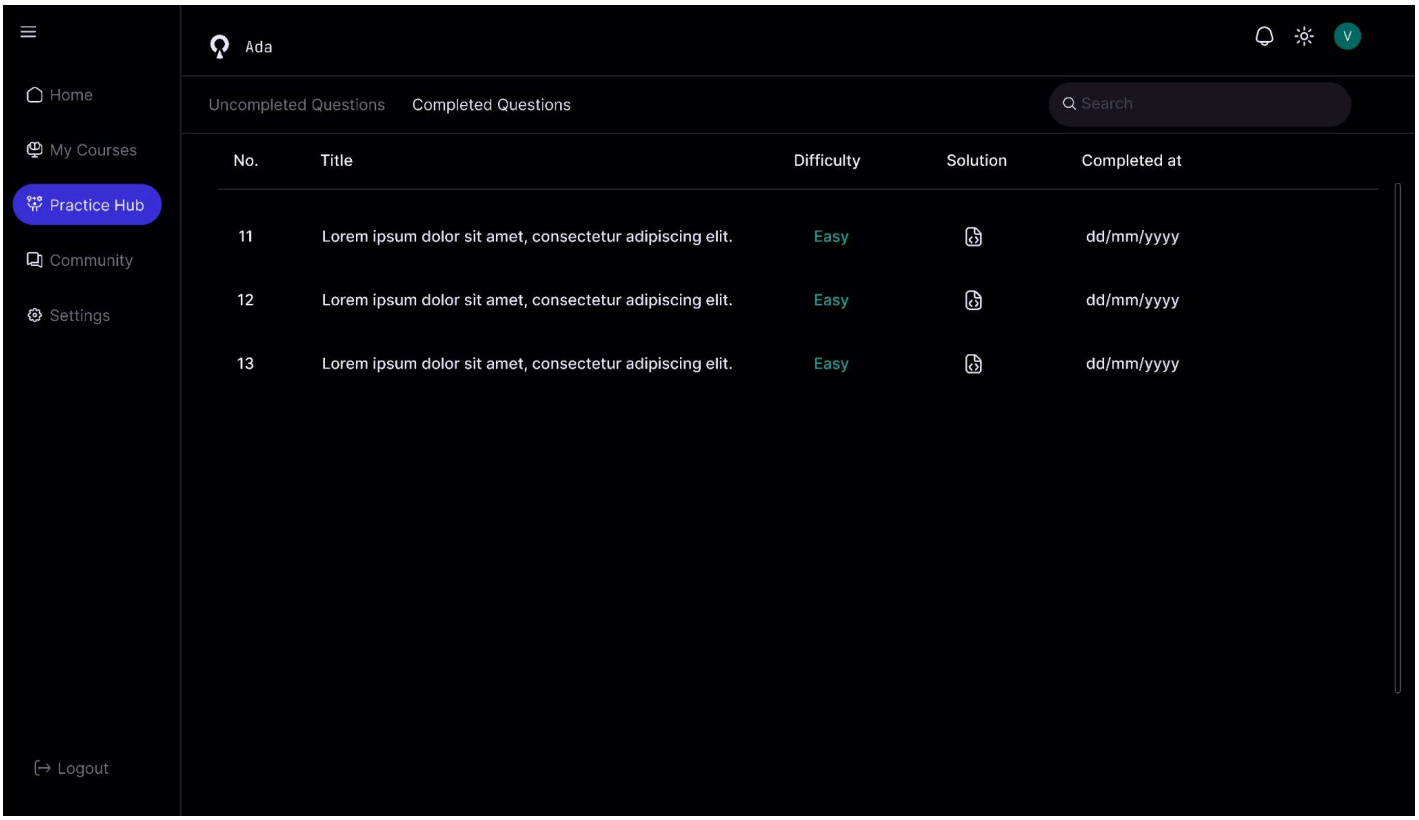
```
SELECT answer FROM Solution WHERE challenge_id = ? AND language = ?;
```

Routes & Function Signatures:

```
@app.route('/practice_hub/<challenge_id>')
def challenge_editor(challenge_id):
    pass
```

Completed Challenges Page

Dark theme Desktop UI



Purpose: This page displays challenges that are complicated by the user. If they like, they can do those things again

-**User Interactions:**

- When the user clicks on the solution button, they can see the solution in a pop-up window similar to the previous page (**Pop-up window in Uncompleted Question Page)**
- They can click on the challenge and do it again.

SQL Query:

1. Display the completed challenges

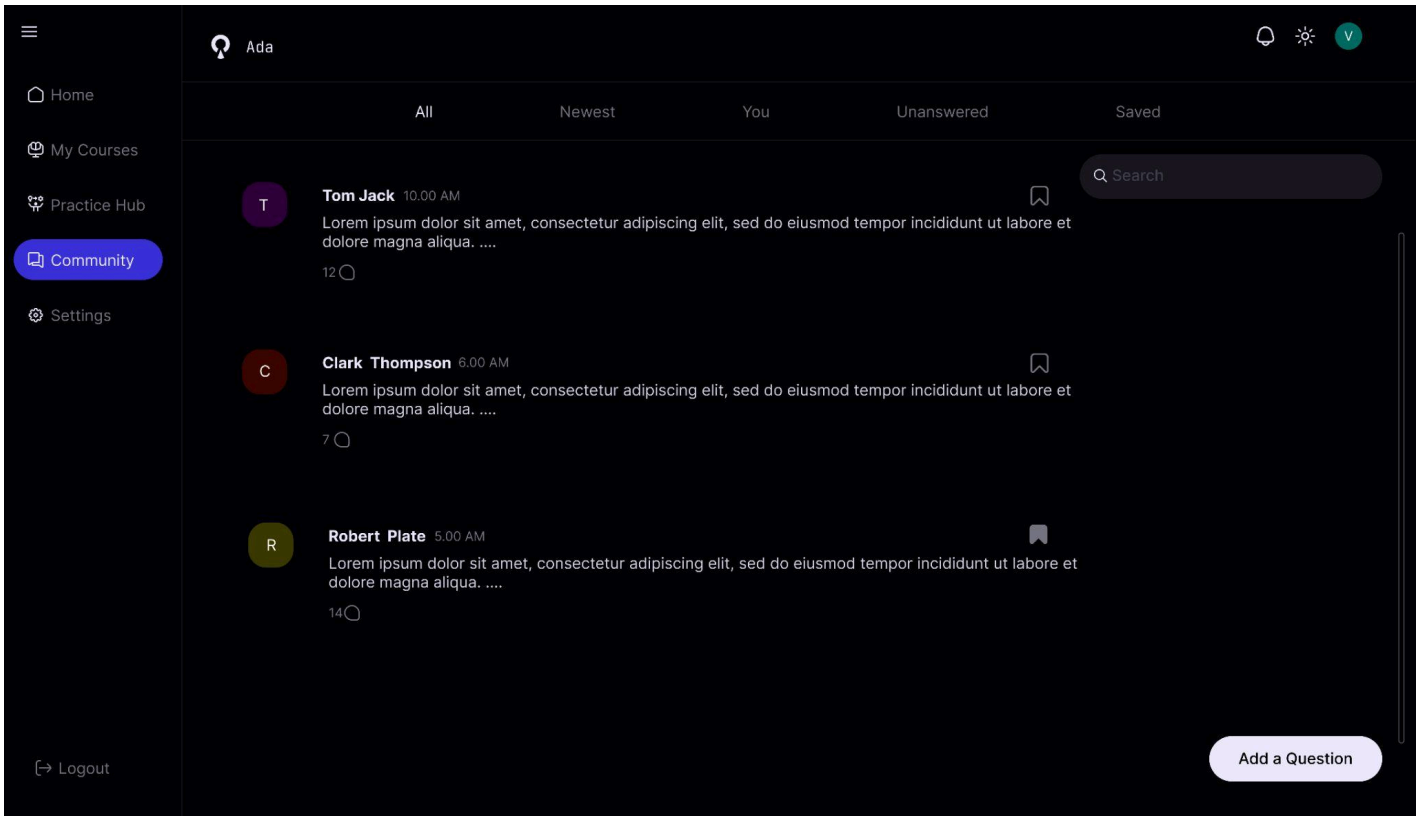
```
SELECT  ch.challenge_id, ch.challenge_title, ch.question, ch.difficulty_level,
ca.completed_at, ch.number FROM Challenge_attempt AS ca JOIN Challenge AS ch ON
ca.challenge_id = ch.challenge_id WHERE ca.user_id = ? AND ca.status = 'completed';
```

Routes & Function Signatures:

```
@app.route('/practice_hub/completed')
def completed_challenges():
    pass
```

Community Page - All Chat

Dark theme Desktop UI



Purpose: This page displays all community questions and discussions that have happened throughout time.

User interactions: Users can save community discussions if they like.

SQL Query:

1. Represent community discussion ordered by time, the latest discussions at the top oldest discussions at the bottom

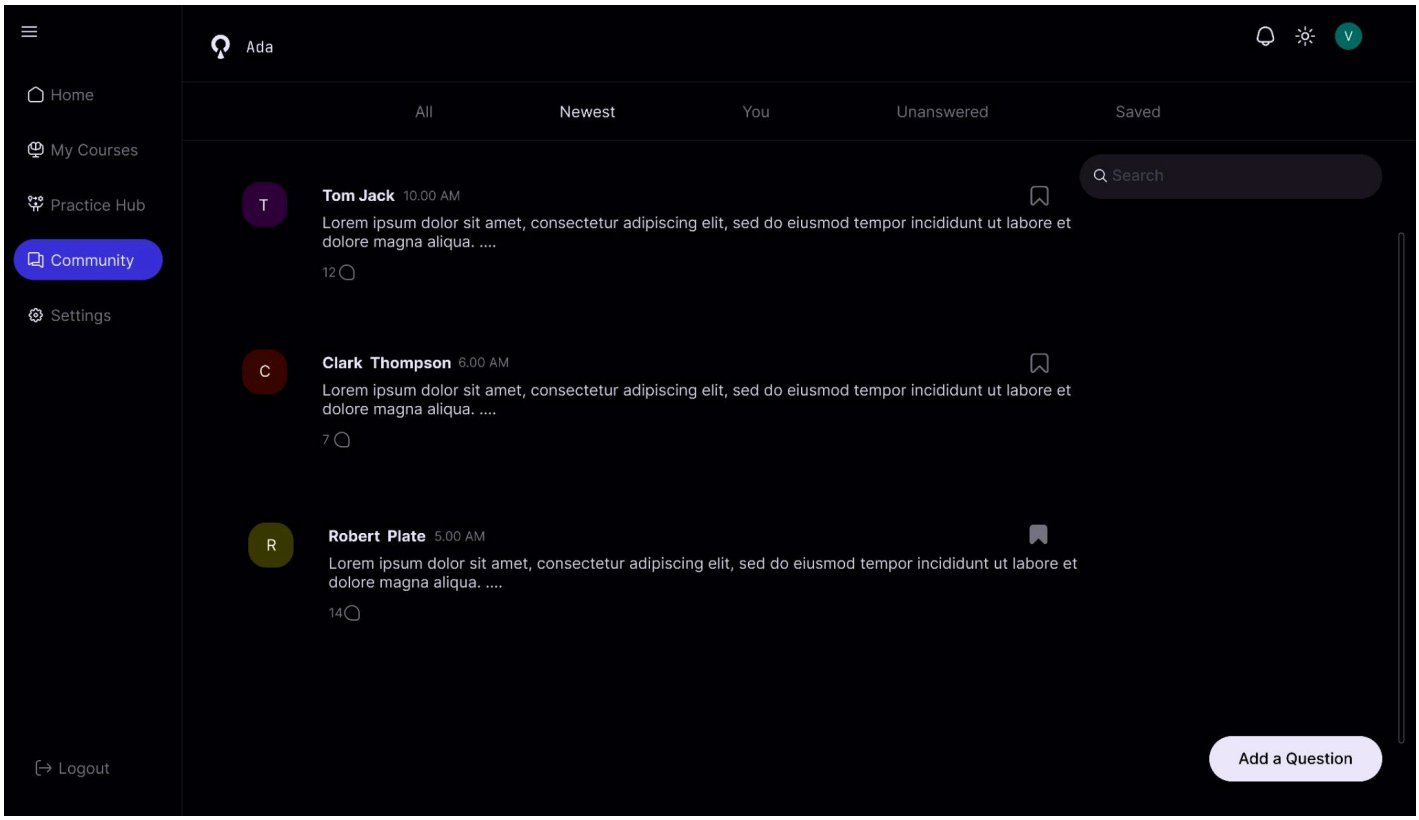
```
SELECT ch.chat_id, u.full_name, u.profile_image, ch.question, ch.created_at,
ch.is_saved, (SELECT COUNT(*) FROM Reply AS r WHERE r.chat_id = ch.chat_id) AS
nu_reply FROM Chat AS ch JOIN User AS u ON ch.user_id = u.user_id ORDER BY
ch.created_at DESC;
```

Routes & Function Signatures:

```
@app.route('/community')
def community_all():
    pass
```

Community Page - Newest

Dark theme Desktop UI



Purpose: This page displays all community questions and discussions that have happened in a day.

SQL Query:

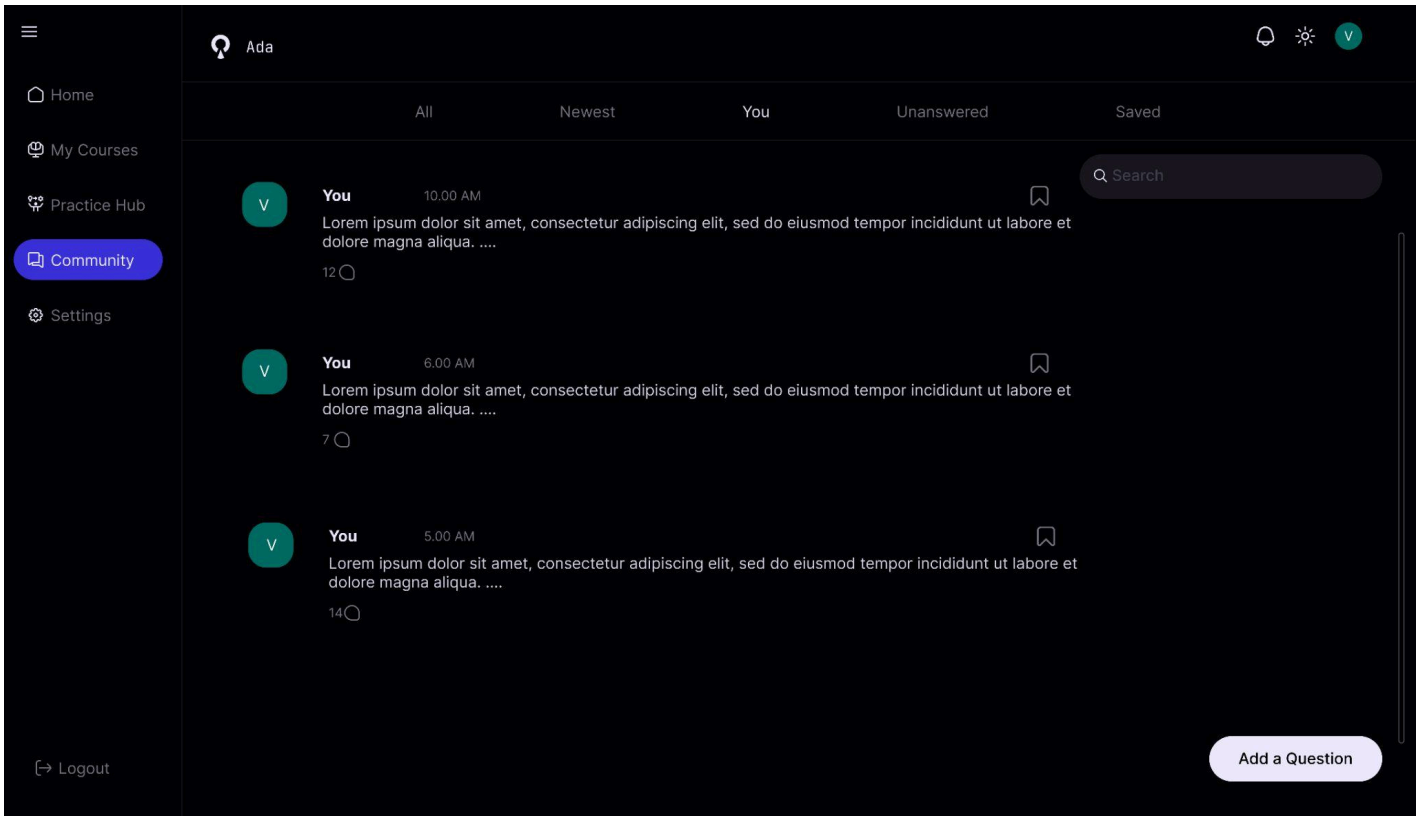
```
SELECT ch.chat_id, u.full_name, u.profile_image, ch.question, ch.created_at,
ch.is_saved,
(SELECT COUNT(*) FROM Reply AS r WHERE r.chat_id = ch.chat_id) AS nu_reply FROM Chat
AS ch JOIN User AS u ON ch.user_id = u.user_id
WHERE ch.created_at ⩾ DATETIME('now', '-1 day') ORDER BY ch.created_at DESC;
```

Routes & Function Signatures:

```
@app.route('/community/newest')
def community_newest():
    pass
```

User's Question Page

Dark theme Desktop UI



Purpose: This page displays all community questions and discussions that have been done by the user.

SQL Query:

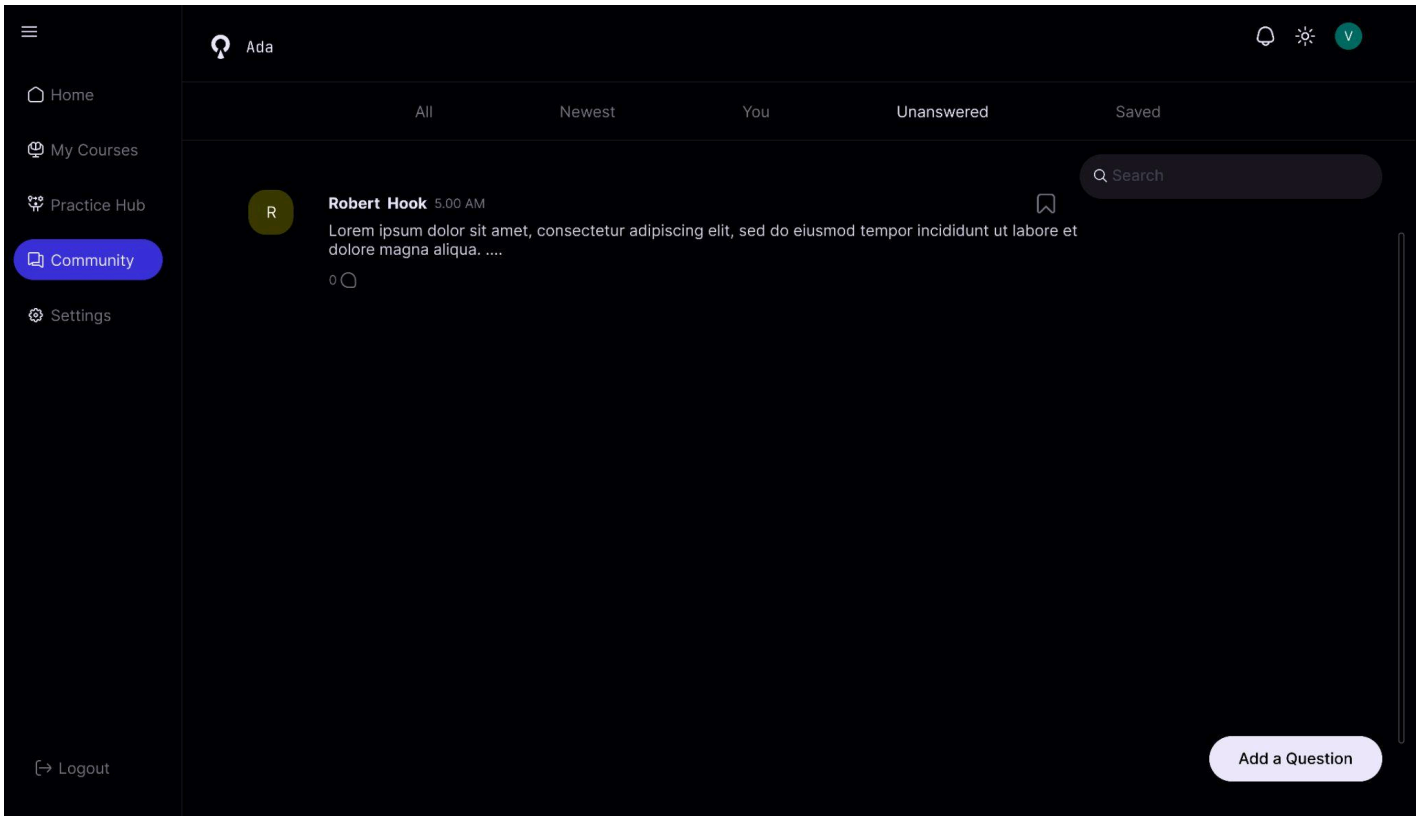
```
SELECT ch.chat_id, u.full_name, u.profile_image, ch.question, ch.created_at,
ch.is_saved,
(SELECT COUNT(*) FROM Reply AS r WHERE r.chat_id = ch.chat_id) AS nu_reply FROM Chat
AS ch JOIN User AS u ON ch.user_id = u.user_id WHERE ch.user_id = ? ORDER BY
ch.created_at DESC;
```

Routes & Function Signatures:

```
@app.route('/community/you')
def your_question():
    pass
```

Not Answered Chat Page

Dark theme Desktop UI



Purpose: This page displays all the unanswered questions that are asked by users.

SQL Query:

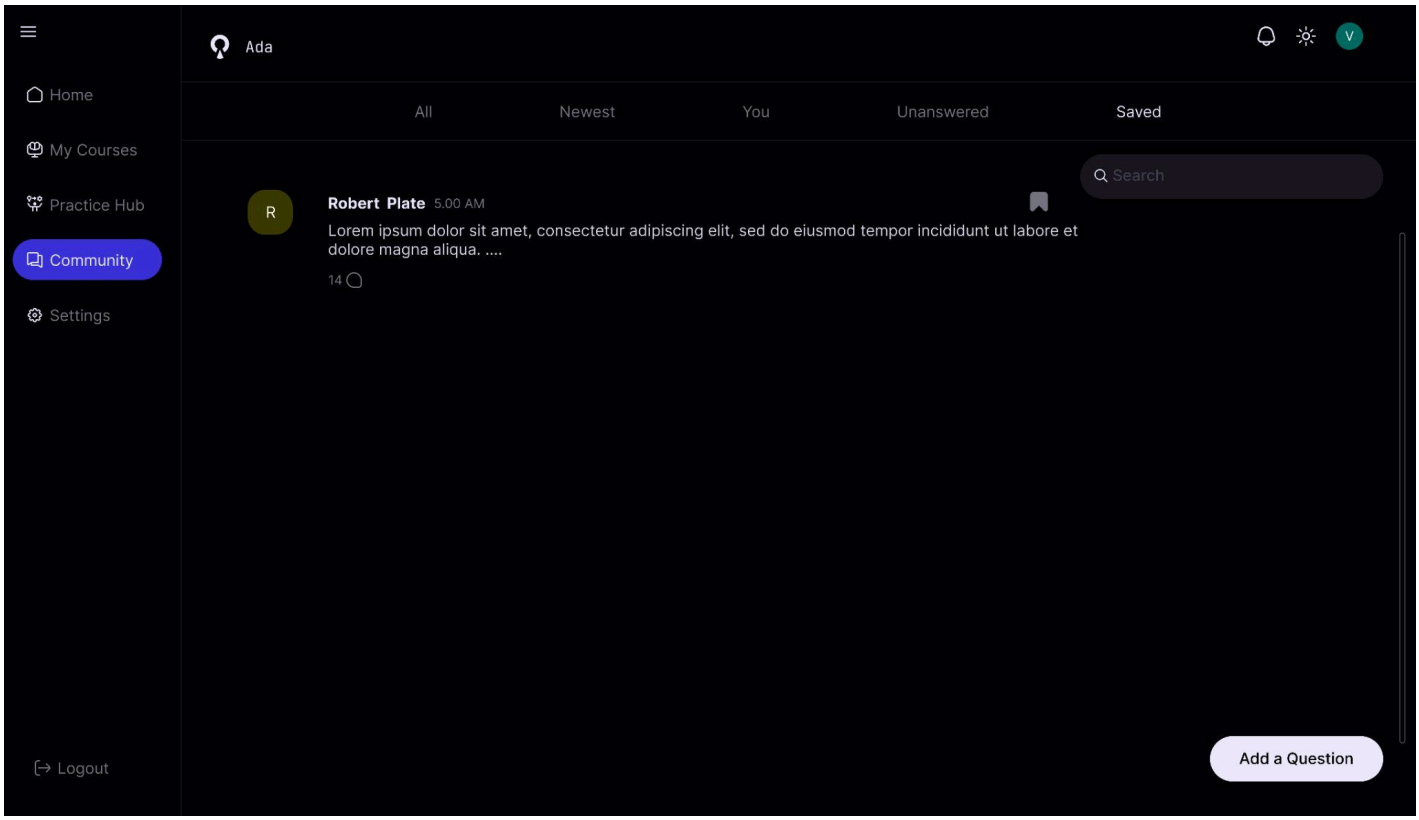
```
SELECT ch.chat_id, u.full_name, u.profile_image, ch.question, ch.created_at,
ch.is_saved, COUNT(r.reply_id) AS nu_reply FROM Chat AS ch JOIN User AS u ON
ch.user_id = u.user_id LEFT JOIN Reply AS r ON ch.chat_id = r.chat_id GROUP BY
ch.chat_id, ch.question, ch.created_at, u.full_name HAVING no_reply = 0 ORDER BY
ch.created_at DESC;
```

Routes & Function Signatures:

```
@app.route('/community/unanswered')
def community_unanswered():
    pass
```

Saved Chat Page

Dark theme Desktop UI



Purpose: This page displays all the saved questions and discussions.

User Interactions: Users can save or unsave a chat by clicking the bookmark icon

SQL Queries:

1. Displays saved chat

```
SELECT ch.chat_id, u.full_name, u.profile_image, ch.question, ch.created_at,
(SELECT COUNT(*) FROM Reply AS r WHERE r.chat_id = ch.chat_id)
AS nu_reply FROM Chat AS ch JOIN User AS u ON ch.user_id = u.user_id
WHERE ch.saved = 1 ORDER BY ch.created_at DESC;
```

2. Save a Chat

```
UPDATE Chat SET is_saved = 1 WHERE chat_id = ? AND user_id = ?;
```

3. Unsave a Chat

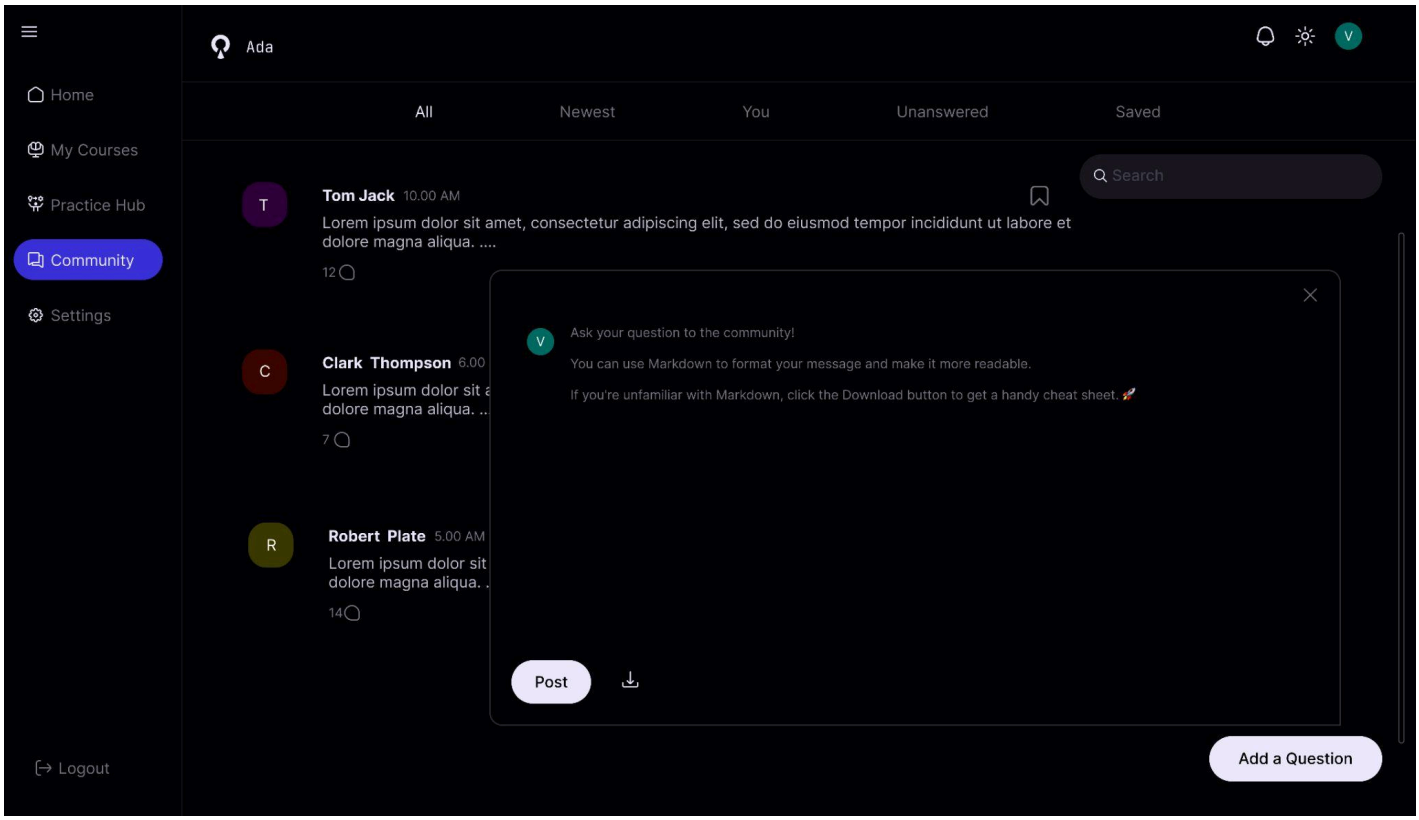
```
UPDATE Chat SET is_saved = 0 WHERE chat_id = ? AND user_id = ?;
```

Routes & Function Signatures:

```
@app.route('/community/saved')
def community_saved():
    pass
```

Pop-up window to Post Questions

Dark theme Desktop UI



Purpose: This pop-up window allows users to post their questions.

User Interactions:

- When the user clicks on the "Add a Question" Button, this window will appear.
- Users can use Markdown to write more stylishly, and they can learn to write Markdown.
- Users can download a cheat shit on markdonw through the download button.
- Users can click "x" to exit from this window.
- Users can click the "post" button to post it.
- Clicking the "post" button submits the question to the backend route `/community/post`.

NOTE: This input box should support markdown

SQL Query:

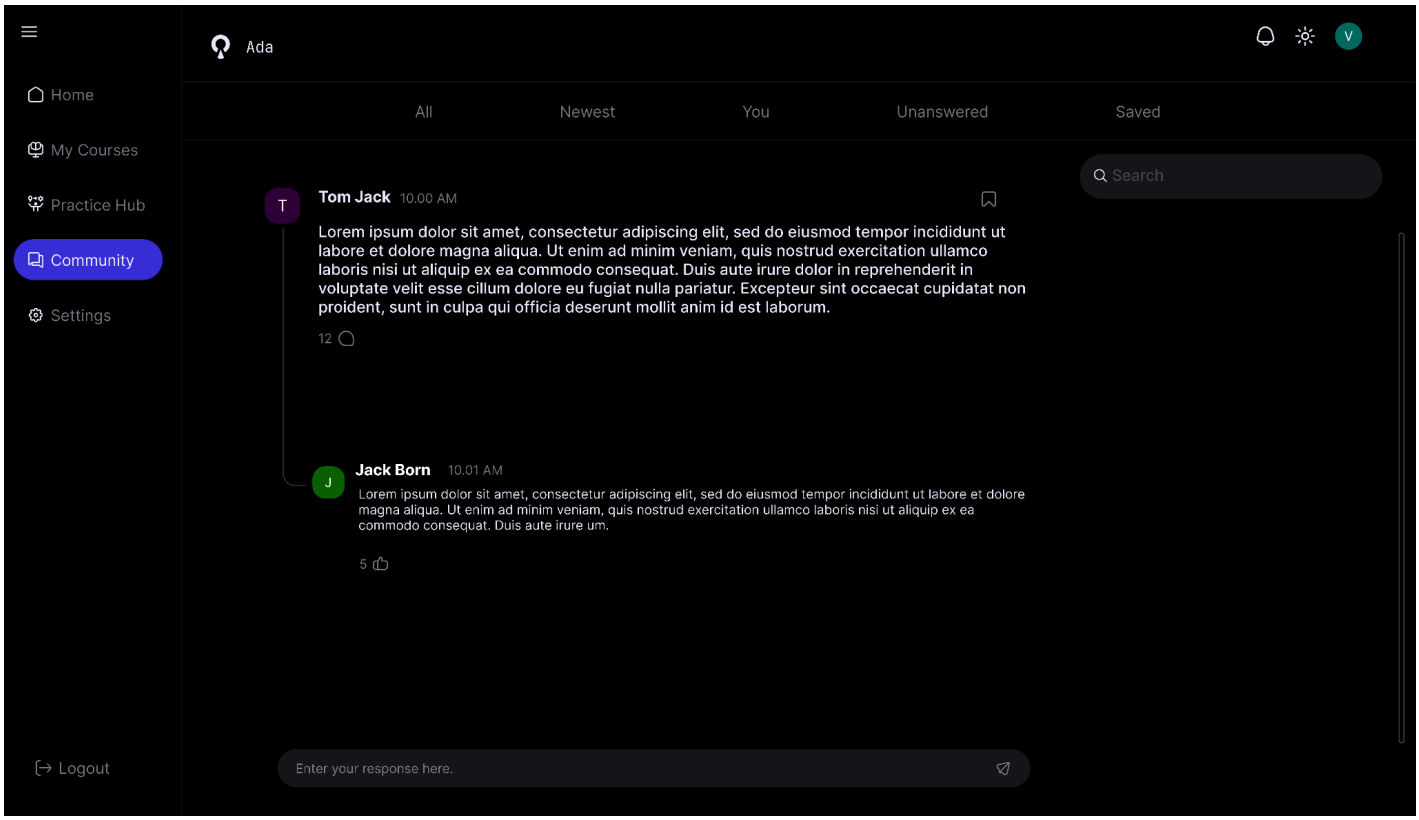
```
INSERT INTO Chat (chat_id, user_id, question, created_at, is_saved) VALUES (?, ?, ?, CURRENT_TIMESTAMP, 0);
```

Routes & Function Signatures:

```
@app.route('/community/post', methods=['POST'])
def post_question():
    pass
```

Answers Page

Dark theme Desktop UI



Purpose: This page displays the selected question and relevant replies for that question.

User Interaction:

- The user can put a like on a reply.
- The user can write replies or a reply to that question

SQL Queries:

1. Displays the question and replies to it, Here the oldest message appears first and the newest message appears last

```
SELECT ch.chat_id, ch.question AS chat_question, ch.created_at AS chat_time,
u.full_name AS asked_by, u.profile_image AS asked_by_image, r.reply_id, r.reply,
r.created_at AS reply_time, ru.full_name AS replied_by, ru.profile_image AS
replied_by_image, r.likes,
(SELECT COUNT(*) FROM Reply AS r2 WHERE r2.chat_id = ch.chat_id) AS total_reply_count
FROM Chat AS ch JOIN User AS u ON ch.user_id = u.user_id LEFT JOIN Reply AS r ON
ch.chat_id = r.chat_id LEFT JOIN User AS ru ON r.user_id = ru.user_id WHERE
ch.chat_id = ? ORDER BY r.created_at ASC;
```

2. Answer to that question

```
INSERT INTO Reply (reply_id, chat_id, user_id, reply, created_at, like) VALUES (?, ?,
?, ?, CURRENT_TIMESTAMP, 0)
```

3. Get the number of likes per reply

```
SELECT r.reply_id, COUNT(rl.like_id) AS like_count FROM Reply AS r
LEFT JOIN Reply_like AS rl ON r.reply_id = rl.reply_id GROUP BY r.reply_id;
```

4. Add a like (One Like Per User Per Reply)

```
INSERT INTO Reply_like (like_id, user_id, reply_id, liked_at) VALUES (?, ?, ?,
CURRENT_TIMESTAMP);
```

5. Remove a like

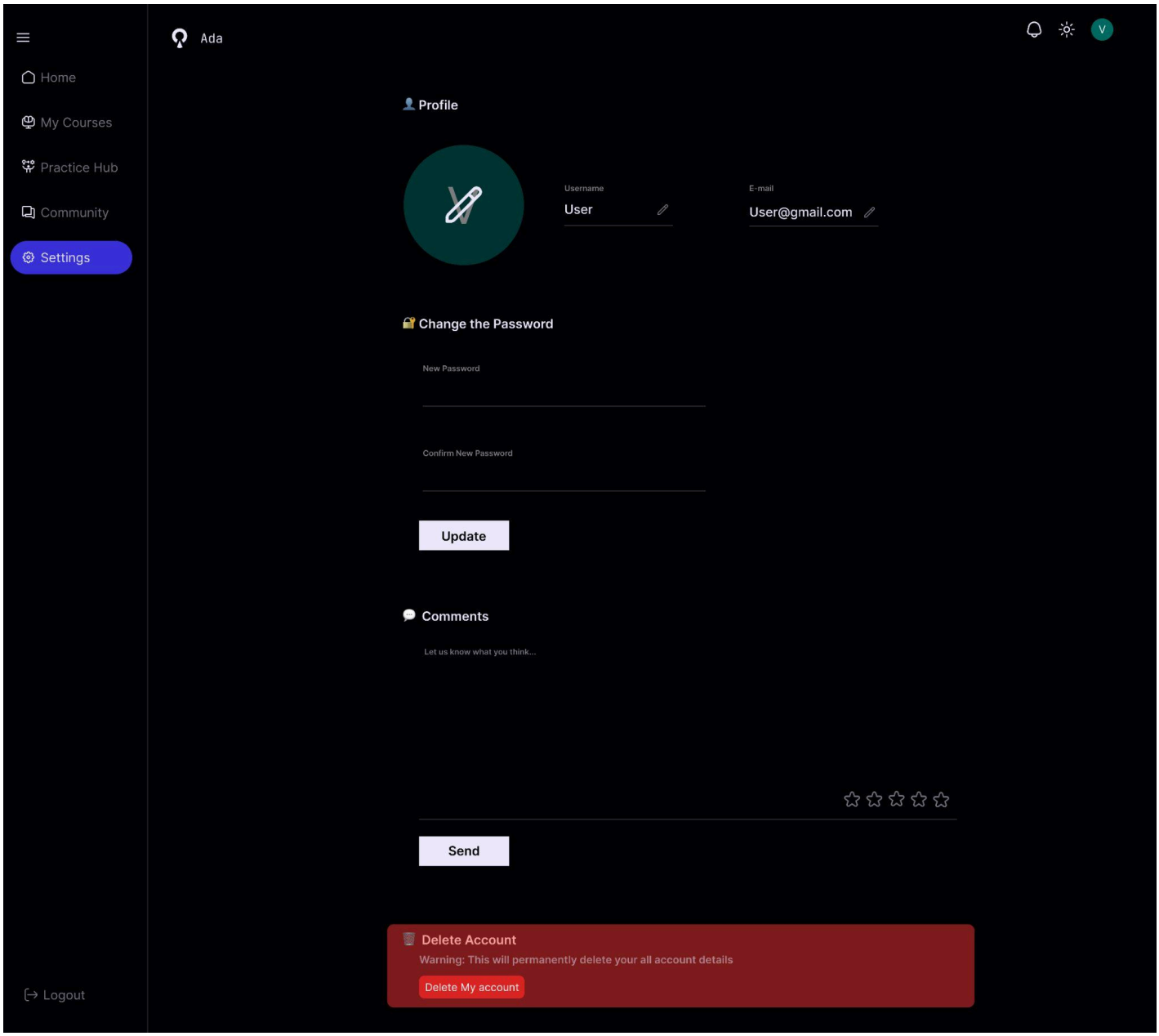
```
DELETE FROM Reply_like
WHERE user_id = ? AND reply_id = ?;
```

Routes & Function Signatures:

```
@app.route('/community/chat/<chat_id>')
def view_chat(chat_id):
    pass
```

Setting Page

Dark theme Desktop UI



Purpose: The User can change their profile pictures, username, email, passwords, and they can put a comment on this product. And users can delete their accounts

SQL Queries:

1. Change the Profile Picture

```
UPDATE User SET profile_image = ? WHERE user_id = ?;
```

2. Change the username (Full Name)

```
UPDATE User
SET full_name = ?
WHERE user_id = ?;
```

3. Change Email

```
UPDATE User
SET email = ?
WHERE user_id = ?;
```

4. Change the Password

```
UPDATE User
SET password = ?
WHERE user_id = ?;
```

NOTE: Hash the password

5. Add a comment on the product

```
INSERT INTO User_feedback (feedback_id, user_id, feedback, star)
VALUES (?, ?, ?, ?);
```

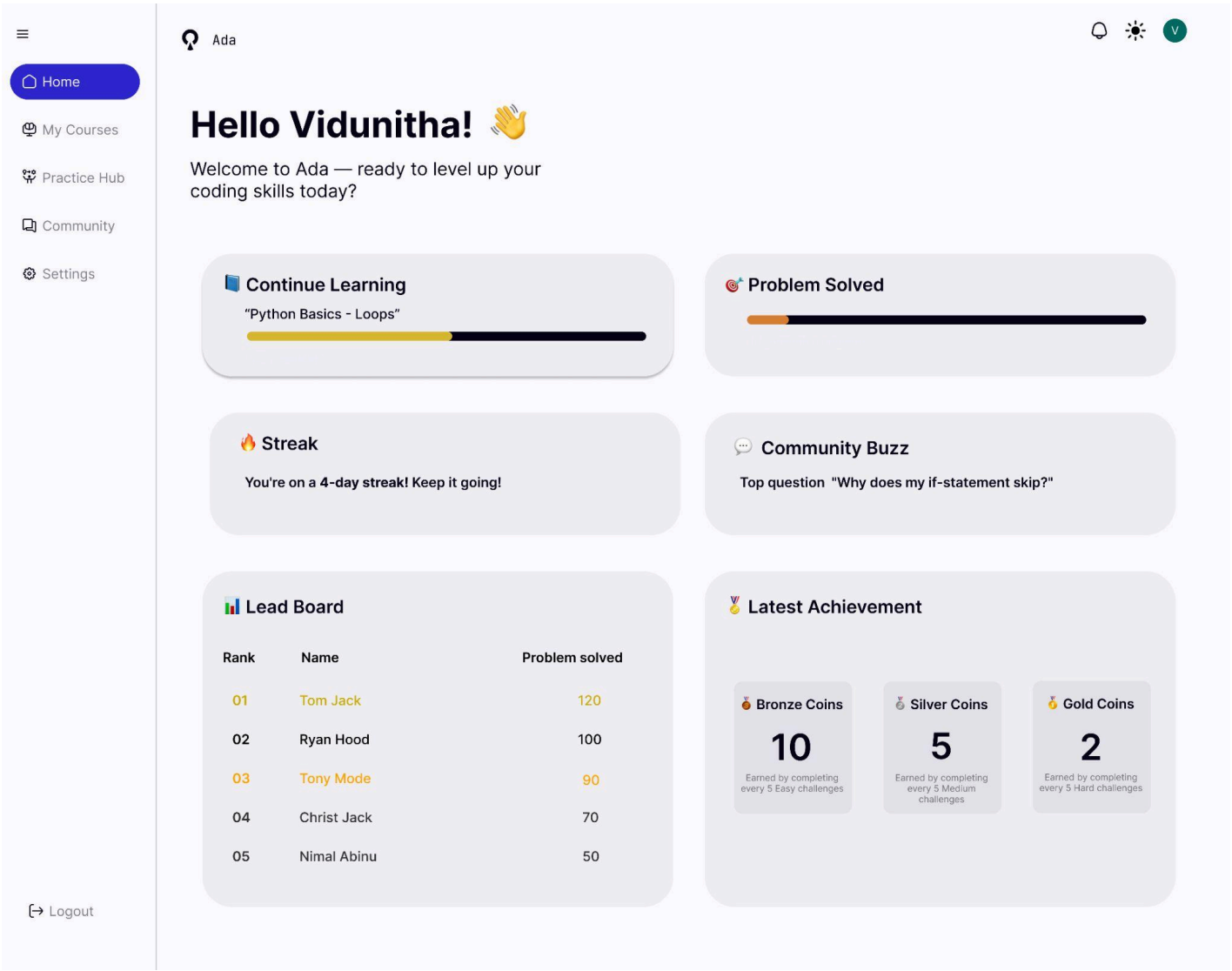
6. Delete the Account

```
DELETE FROM User
WHERE user_id = ?;
```

Routes & Function Signatures:

```
@app.route('/settings', methods=['GET', 'POST'])
def settings():
    pass
```

Sample White theme UI



NOTE: There are sample UIs for the Mobile version on the above pages

SOFTWARE INTERFACES

Core Technologies and Frameworks

LAYER	TECHNOLOGY
Frontend	HTML + CSS + Jinja
	JavaScript (when necessary)
	React (Optional)
Backend	Python + flask
Templating	Jinja
Database	SQLite + SQLite Studio

Libraries and Tools

Tool / Library	Purpose	Status
Flask	Web backend framework	In use
Jinja	Templating engine for HTML	In use
Monaco Editor	In-browser code editor	In use
SQLite	Lightweight local database	In use
Tailwind CSS	Utility-first CSS framework for UI styling	Optional
React	To increase the speed of the app	Optional
bcrypt	Password hashing for secure login	Planned
OpenAI API	AI-powered code feedback (future feature)	Planned
Judge0 API	Remote code execution engine	Planned

NOTE: These Libraries and tools can change within the development process, and there will be more libraries.

Operating Environment

- Operating System Supported:
    - **Development**: Windows/macOS/Linux
    - **Deployment**: Any system capable of running Python/Flask apps

Development Environment

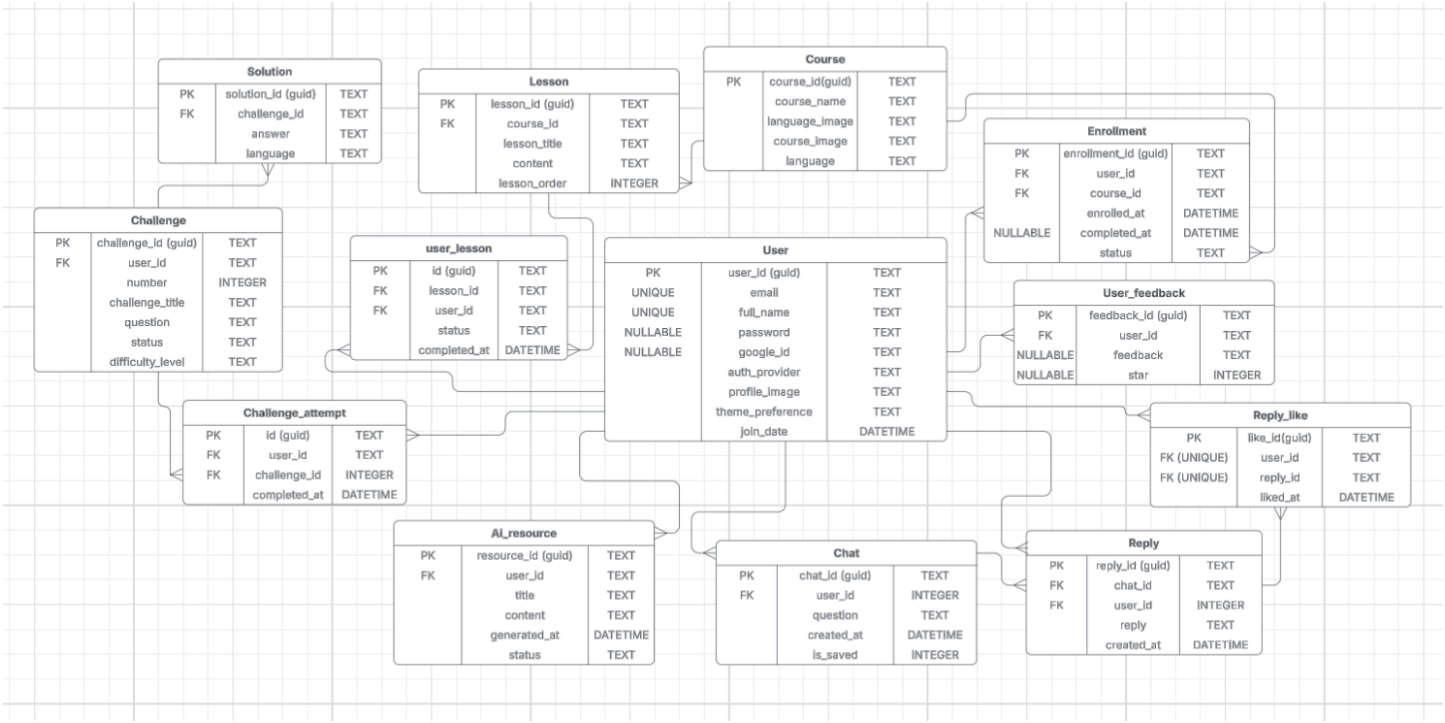
- **Code Editor:** Visual Studio Code
- **Python Version**: Python 3.x (latest stable version)
- **Package Manager**: `pip` (To install libraries)
- **Virtual Environment**: `venv` (optional)
- **Browser**: Chrome, Firefox, or Microsoft Edge (Test for frontend)
- **Command Line**: Terminal / Command Prompt / VS Code Terminal
- **SQLite Studio**: For database design

ADDITIONAL NONFUNCTIONAL REQUIREMENTS

SECURITY

- User Passwords should be stored securely.
- API keys will be stored in environment variables and never exposed to the frontend.
- Form data is validated to avoid injection attacks.
- HTTPS must be used to communicate between the client and server when deployed.

ER DIAGRAM FOR THE APPLICATION



COLOUR AND FONTS

Dark theme Colour Palette - Web Application

Text	Background	Primary	Secondary	Cards
#EBE9FC	#010104	#3A31D8	#020024	#0D0D11

White theme Colour Palette - Web Application

Text	Background	Primary	Secondary	Cards
#050316	#fbfbfe	#2F27CE	#dddbff	#EEEEF2

Dark theme Colour Palette - Landing Page

Text	Background	Primary	Secondary
#ede3ff	#05000E	#0D0024	#fd550e

White theme Colour Palette - Landing Page

Text	Background	Primary	Secondary
#09001a	#f5f0ff	#e8dbff	#f24a02

I use a Blue colour theme for the application, because blue and violet tones are psychologically associated with **trust, relaxation, less stress, intelligence, and stability,** and it gives a professional look to the product, which is best for a Developer-Focused Interface. Most applications use blue colour themes, some examples are VS Code, GitHub

Fonts I use for the Web Application

1. Konkhmer Sleokchher:- For the greeting part in the dashboard
2. Inter:- For all other texts except above

Fonts I use for the Landing Page

1. Roboto:- Landing Page "Unlock your coding potential with Ada …" and "Ada is a dynamic platform designed to help you learn coding …"
2. Inter:- Navigation Bar and sign-in button, sign-up button
3. Krub:- "Get started" button
4. Poppins:- Heading of the Why Us section. supported language section, how it works section
5. Montserrat:- description in those sections
6. Lato:- Other texts except above

IMPLEMENTATION AND ITERATIVE DEVELOPMENT

Project Link :

https://github.com/AlgorithmicPV/Ada-Learning-Platform.git

ITERATIVE DEVELOPMENT SUMMARY

This project has been developed using an iterative development methodology. I divided the project into nine sprints. Each sprint focuses on building and refining features, followed by testing, feedback and improvements. This method helps me to improve the quality of my application through feedback, manage my time efficiently and increase the flexibility of my application.

Sprint	Work
Sprint 1	Planning
Sprint 2	Landing Page
Sprint 3	Database Setup
Sprint 4	Authentication
Sprint 5	Core Learning Features
Sprint 6	Community and Practice Hub
Sprint 7	Dashboard & Settings
Sprint 8	UI/UX Refinements
Sprint 9	Testing & Documentation

Sprint 1 - Planning

What I did

- Brainstormed some ideas for this digital outcome, and got some feedback on ideas that I thought. I got a few ideas, to make a web application for the business AI chat, a travelling web application and a Learning web application for new programmers.
- From my feedback, I decided to create a learning web application, which includes lessons, challenges, a custom AI-generated course, an in-built code editor and community features.
- Then, I designed the initial wireframes for my UI and ER diagram.
- I chose the colour plates and fonts for the application.
- Also, I wrote rough SQL queries for each UI, which can be helpful for my development process.

Why

- Planning was helpful for me, because it gave me a clear direction before the development. By identifying key features early, I was able to stay focused. Wireframes helped me to design the UI quickly without taking time for the design. The ER diagram helped me to design the database soon, supporting

data integrity before implementation. Overall, planning helped me to move into the development process with confidence and saved time later in the project.

Sprint 2 - Landing Page and the frontend of the Dashboard

What I did

- The landing page was created with light and dark themes, enabling users to choose according to their preference or accessibility needs, and it created a professional, attractive first impression by adding branding elements, including a logo, custom fonts, and animated visuals.
- I also created the landing page to be mobile responsive.
- Created the frontend of the dashboard, but could not create the backend as the data was not enough at that moment

Why

- Users will be first attracted by the landing page, and therefore, it should be visually appealing, modern, and accessible. I believe Dark/light themes can improve usability for different lighting conditions.
- Dashboard gives a rough idea to the user where they are.

Sprint 3 - Database Setup

What I did

- Designed all the tables with correct connections using sqliteStudio.
- Populated the tables with data using a Python Script

Why

- A structured database confirms that the learning platform stores lessons, progress, and user activity reliably.

Sprint 4 - Authentication

What I did

- I built signup and login functionality using Argon2 password hashing for strong security.
- Added Google login integration for convenience.
- Implemented validation (reject empty email/password, etc).

Why

- Authentication is the foundation for user trust and privacy. Argon2 is stronger than SHA256/MD5 and prevents brute-force attacks.

Sprint 5 - Core Learning Features

What I did

- I made the code editor with compiler support
- Added AI chatbot
- Implemented progress tracking
- Created a user interface for all the pages related to the My Course section with all the functionality

- Made all the UIs mobile responsive
- Search functionality

Why

- As learning features are the core purpose of the platform. Without these, the application would not meet its outcome.

Sprint 6 - Community and Practice Hub

What I did

- I made the code editor with compiler support for the practice hub
- Added an AI chatbot for the practice hub section
- Implemented progress tracking
- Created a user interface for all the pages related to the Community section and Practice Hub section, with all the functionality
- Made all the UIs mobile responsive
- Search functionality for both sections
- Added functionalities for the community hub section, such as adding new questions, answering posted questions, saving questions, and liking answers.

Why

- Social features increase collaboration and engagement. The Practice Hub ensures learners don't just skip content but track their progress.

Sprint 7 - Backend of the Dashboard & Settings

What I did

- I made the backend of the Dashboard
- Settings allow updating username, email, password, profile picture, and deleting the account.

Why

- Personalisation and control are essential for user engagement and ethical data management (deletion).

Sprint 8 - UI/UX Refinements

What I did

- Improved responsiveness for mobile devices.
- Built 404 and 403 error pages with animations.
- Improved profile image handling and navigation bar usability

Why

- Polishing UI ensures the app is not only functional but enjoyable. Error pages prevent confusion.

Sprint 9 - Testing & Documentation

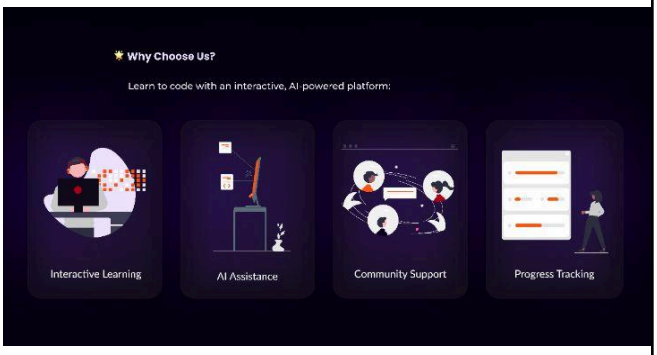
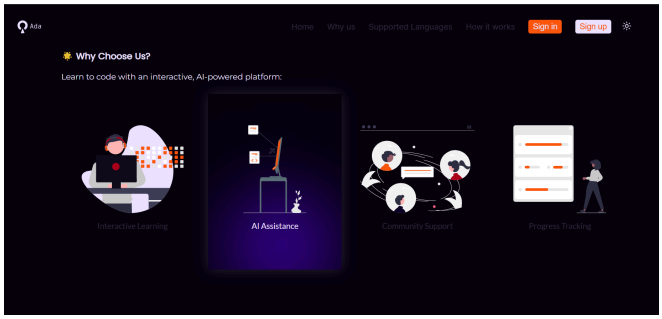
What I did

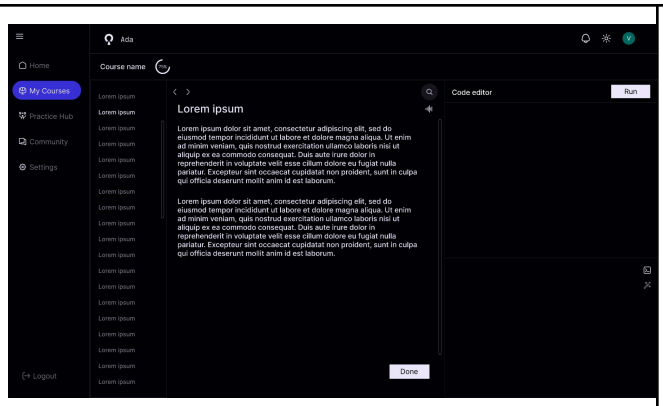
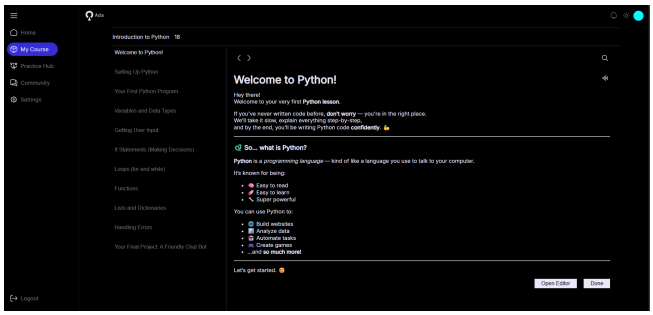
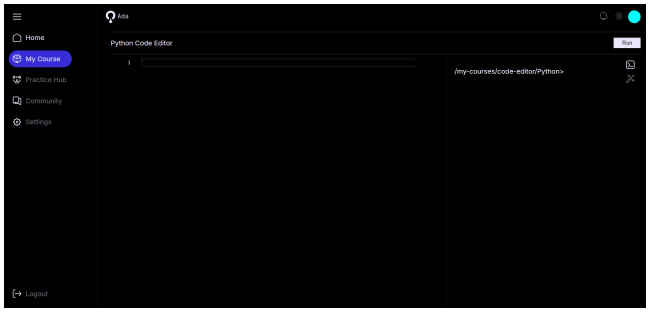
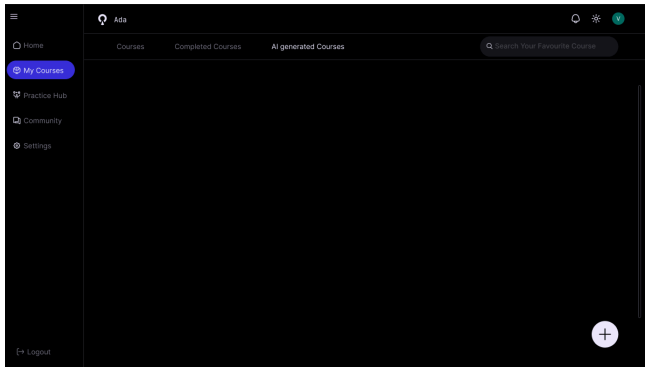
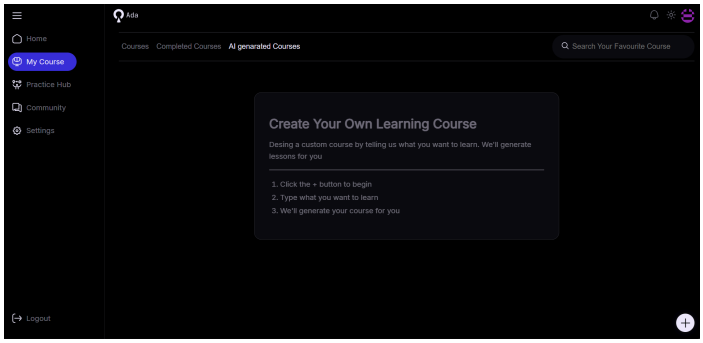
- Added Python docstrings and inline comments to functions and Flask routes.
- Refactored into Flask blueprints, reducing code repetition.
- Browser tested on Chrome, Edge..
- Done other Application testing

Why

- Confirmed app worked across different browsers. Linted code with Flake8 and Pylint.

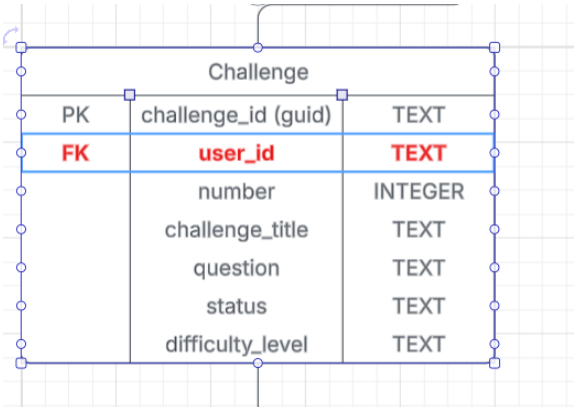
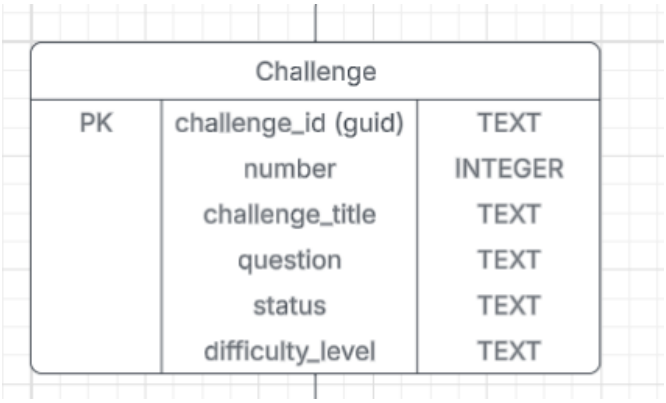
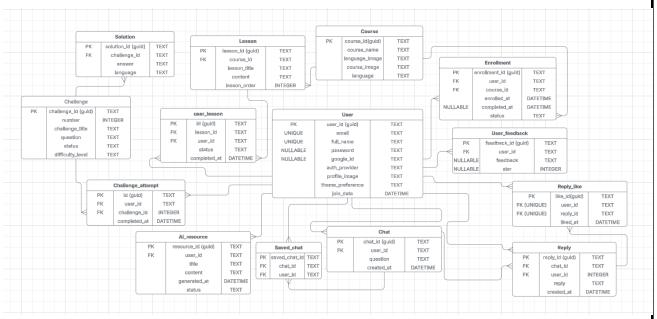
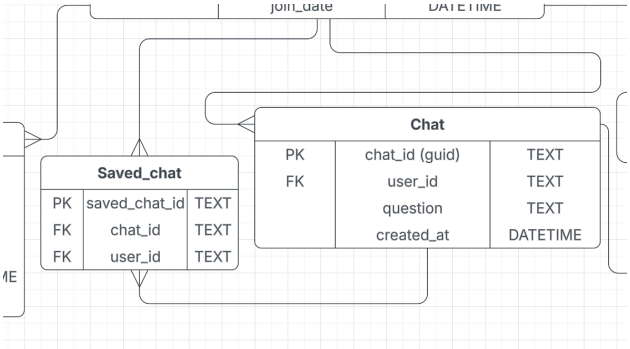
Iterative Improvements Table (Web)

<table>
<tr><th>No.</th><th>Date</th><th>Area</th><th>Description of change</th><th>Reason for change</th></tr>
<tr><td>1</td><td>06/08/2025</td><td>Frontend - Landing Page - Why Us Section</td><td>Make the borders a little bit rounded, not a lot, and add a gradient overlay that moves with the mouse pointer, and add a box shadow to each card</td><td>It gives a modern look to the landing page</td></tr>
<tr><td></td><td colspan="3">Before:-
 
</td><td>After:-
 
</td></tr>
<tr><td>2</td><td>06/23/2025</td><td>Frontend - Web app - My course-lesson page</td><td>Remove the code editor part from the lesson page and keep it on a separate page.</td><td>With the code editor, it is difficult to manage the layout for the mobile version; mobile users cannot utilise the code editor effectively. Even on larger screens, it is challenging to maintain user-friendly elements, and users may feel overwhelmed by the excessive amount of content on a single page.</td></tr>
<tr><td></td><td colspan="3">Before:-</td><td>After:-</td></tr>
</table>

				 
3	08/02/2025	AI-generated courses page	Add a user guide for the AI-generated courses page	Due to the feedback from Sprint 1, I think it is better now because it increases user friendliness. (This only shows when the user hasn't created any courses)
	Before:- 			After:- 

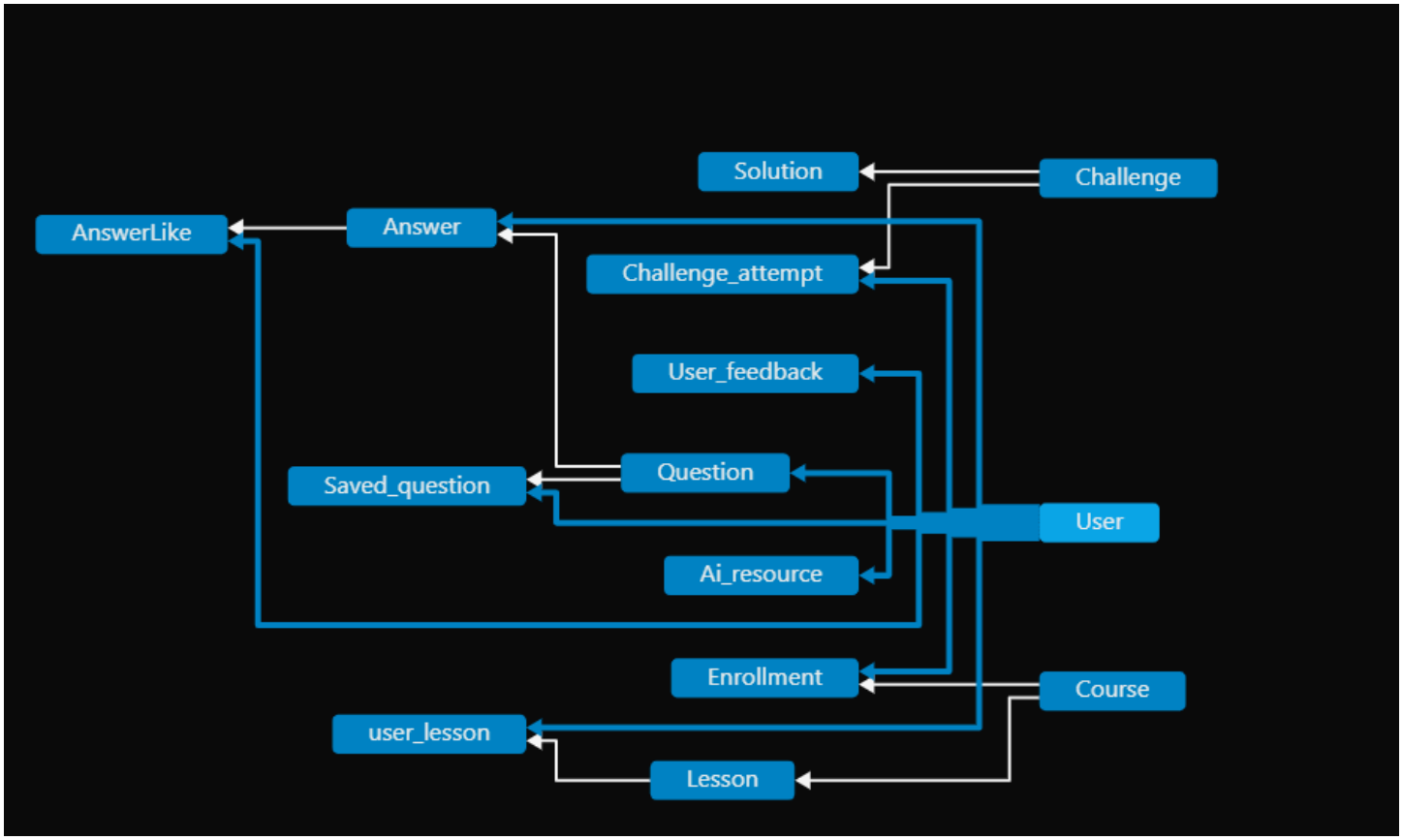
Iterative Improvements Table (Database)

No.	Date	Area	Description of change	Reason for change
1	05/05/2025	Database - Challenge table	Delete the ***user_id*** from the ***Challenge***.	Because it was a mistake, we don't need that, as we have the ***challenge_attempt*** table to connect the ***Challenge table*** and the ***User*** table.

	Before  			After 
2	07/02/2025	Database - Saved_chat table	Add a new table called “Saved_chat”.	Previously, saved_chat was in the “Chat” table as a column. This is not working as many users can select one discussion and one discussion can be selected by many users; therefore, put that into a separate table.
	Before:- 			After:- 

3	07/07/2025	Database - Name of the Reply table	Change the name of the Reply table to Answer table	Because the Community section is more like a Q&A website, not a chat application, therefore Answer table makes more sense, as this table stores answers for relevant questions.
4		Database - Name of the Chat table	Change the name of the Chat table to Question table	Because the Community section is more like Q & A, and I think the Question table is more sensible rather than saying it as a Chat Table
5		Database- Name of the Reply_like	Change the name of the Reply_like table to AnswerLike table.	Same reason above

FInal ER diagram



Testing

NOTE: Tested in different modern browsers, which include Chrome and Microsoft Edge

Feature/ Module: Login Section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Show the Image of the laptop on the login page	Load the login screen	Shows the image of the laptop	Showed the image of the laptop	Pass	-
002	Show the logo of the app on the Login page	Load the login screen	Shows the Application logo on the login page	Showed the application logo on the login page	Pass	-

Feature/ Module: Login Section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
003	Email input field visibility	Load the login screen	The email field appears with placeholder text	The email field is visible with the placeholder	Pass	-
004	Password input field visibility	Load the login screen	The password field appears with the placeholder	The password field appears with the placeholder	Pass	-
005	Placeholder animation (email field)	Click inside the email field	Placeholder moves up and shrinks	Placeholder animation works	Pass	-
006	Placeholder animation (password field)	Click inside the password field	Placeholder moves up and shrinks	Placeholder animation works	Pass	-
007	Google Login button styling	Load the login screen	The Google login button appears correctly	Google login button styled properly	Pass	-
008	Login button visibility and style	Load the login screen	The login button is visible and styled	Button displayed properly	Pass	-
009	Input focus effect	Click in the input field	Bottom border glows	Visual effect appears as expected	Pass	-
010	Hover effect on “Create an account” text and “or” text	Hover the mouse over both texts	Changes the text colour smoothly	Colour changes as expected	Pass	-
011	Hover effect on the Login button	Hover the mouse over the login button	Button scales up slightly	Button enlarges with animation	Pass	-
012	Hover effect on the	Hover the	Button scales	Button	Pass	-

Feature/ Module: Login Section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
	Google login button	mouse over the Google login page	up slightly	enlarges with animation		
013	Toggle show password icon	Click on the eye icon	The password becomes visible	Password shown	Pass	-
014	Toggle hide password icon	Click on the non-eye icon	The password becomes hidden	Password Hidden	Pass	-
015	Responsive layout (mobile)	Resize screen to mobile view	Layout adapts correctly	The layout should adapt correctly with no overflow	Pass	-
016	Log in with valid inputs	Email: admin@gmail.com Password: 123456789	The dashboard will appear	Dashboard appeared	Pass	-
017	Log in with invalid inputs	Email: wrong@gmail.com Password: 1234	An error message will appear saying “Authentication failed”	An error message appeared saying “Authentication failed”	Pass	-
018	Log in with empty email and password fields	Spaces and No value	An error message will appear saying “Email and password required”	An error message appeared saying “Email and password required”	Pass	-
019	Log in with a valid	Email:	An error	An error	Pass	-

Feature/ Module: Login Section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
	email and a wrong password	admin@gmail.com Password: 1234	message will appear saying "Authentication failed"	message will appear saying "Authentication failed"		
020	Log in with an invalid email and an existing password	Email: wrong@gmail.com Password: 123456789	An error message will appear saying "Authentication failed"	An error message will appear saying "Authentication failed"	Pass	-
021	function of the Google Login button	Click on that button	The page will redirect to the Google login	As expected	Pass	-
022	Function of the Create an account button	Click on it	Redirect to the sign-up page	As expected	Pass	-
023	Type 3 characters to check the limitation in the email field	type "ddd"	An error message will appear saying "Email must be between 3 and 150 characters.	As expected	Pass	-
024	Type 51 characters to check the limitation in the email field	Type "d x 51"	An error message will appear saying "Email must be between 3 and 150 characters.	As expected	Pass	-
025	Type 1200 characters to check the limitation in the password field	Type "d x 1200"	An error message will appear	As expected	Pass	-

Feature/ Module: Login Section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
			saying “Password must be between 3 and 150 characters.			

*** All the validations were done with and without the required tag in HTML**
NOTE: For the passwords, users can type spaces

Feature/ Module: Landing page section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	The logo displays correctly	Load page	The logo is visible and positioned in the top left of the navigation bar	The logo was visible and positioned in the top left of the navigation bar	Pass	-
002	The Navigation Bar appears	Load page	The navigation bar appears	The navigation bar appeared	Pass	-
003	The navigation bar is stuck at the top of the screen	Scroll the page	The navigation bar was placed at the top when scrolling	The navigation bar is placed at the top when scrolling	Pass	-
004	Navigation bar links scroll to sections	Click "Why Us" / "Contact"	Smoothly scrolls to the section	Smoothly scrolled to the section	Pass	-
005	Navbar hover animations work	Hover over nav links	Colour changes and underlines	Colour changed and underlined texts	Pass	-

Feature/ Module: Landing page section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
006	Theme toggle appears	Load page	Dark/light mode toggle is visible depending on the user's previous choice	The dark/light mode toggle was visible depending on the user's previous choice	Pass	-
007	Theme switch functionality	Click toggle	The theme changes between light and dark smoothly	The theme changed between light and dark smoothly	Pass	-
008	Image loading	Load page	All images display properly	All images displayed properly	Pass	-
009	Animation of the SVG image in the hero section	Load page	SVG image's animation works	SVG image's animation worked	Pass	-
010	Buttons hover on the landing page	Hover over the buttons	Buttons scale up slightly	Buttons enlarged with animation	Pass	-
011	The Cursor changes when hovering over the buttons	Hover over the buttons	Cursor changes to pointer	Cursor changed to pointer	Pass	-
012	Animation appeared when hovering over the cards in the Why Choose use section	Hover over the cards	A glow effect follows the mouse, and the box shadow in the cards appears	A glow effect followed the mouse, and the box shadow in the cards appeared	Pass	-
013	Animation when hovering over the	Hover over the	Language blocks scale	Language blocks	Pass	-

Feature/ Module: Landing page section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
	language blocks in the Supported Languages section	languages block	up slightly	enlarged with animation		
014	Animation of moving comment boxes	Page Load	Comment boxes move across the screen continuously	Comment boxes moved across the screen continuously	Pass	-
015	Stop the animation when hovering over the comment boxes	Hover over the comment box	Stops the animation of the comment boxes	Stopped the animation of the comment boxes	Pass	-
016	Hover over the step blocks in the How it works section to test if those blocks enlarge	Hover over the step blocks	Step blocks scale up slightly	Step blocks enlarged with animation	Pass	-
017	Responsive layout S(mobile)	Resize screen to mobile view	Layout adapts correctly	Layout should adapt correctly with no overflow	Pass	-
018	Cards in the Why Us section carousel display correctly on mobile view	Open the app on the mobile screen	cards appear centred, image + text visible	The card displayed as expected in mobile view	Pass	-
019	Mobile menu opens on hamburger click	Click the hamburger icon on mobile	Side menu slides open with nav links	Menu displayed correctly	Pass	-
020	Navigation links are visible in the mobile menu	Open menu	All links (Home, Why us, Sign in…) are visible and clickable	Links appear correctly	Pass	-
021	The menu closes on	Click the	Side menu	Menu closed	Pass	-

Feature/ Module: Landing page section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
	clicking the X button	close icon (X)	disappears	successfully		
022	Mobile menu supports theme toggle	Click the theme toggle icon inside the menu	Theme switches properly	Toggle works inside the menu	Pass	-
023	Clicking the nav links scrolls to correct sections	Click "Why Us" in the mobile menu	Scrolls to "Why Us" section	Smooth scroll occurred	Pass	-
024	The menu adapts correctly in light and dark modes	Switch themes	Menu styles update to match the active theme	Styles consistent with the theme	Pass	-

Feature/ Module: Sign up section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	The signup page displays correctly on desktop	Load the page in desktop view	Page displays with image and input fields properly aligned	As expected	Pass	-
002	The signup page displays correctly on mobile	Load the page in mobile view	Layout stacks vertically, image resizes properly, no overflow1	As expected	Pass	-
003	The signup form input fields appear	Load page	All required fields are visible	As expected	Pass	-
004	Placeholder text appears inside input	Load page	Fields show placeholders	As expected	Pass	-

Feature/ Module: Sign up section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
	fields		like "Email", “Name”, etc			
005	Input focus effect	Click in all input fields	Bottom border glows	Visual effect appears as expected	Pass	-
006	Show the Image of the laptop on the login page	Load page	Shows the image of the laptop	Showed the image of the laptop	Pass	-
007	Show the logo of the app on the Login page	Load page	Shows the Application logo on the login page	Showed the application logo on the Signup page	Pass	-
008	Placeholder animation (all fields)	Click inside all fields	Placeholder moves up and shrinks	Placeholder animation works	Pass	-
009	Toggle the show password icon (both the password field and the confirm password)	Click on the eye icon	The password becomes visible	Password shown	Pass	-
010	Toggle hide password icon (both the password field and the confirm password)	Click on the non-eye icon	The password becomes hidden	Password Hidden	Pass	-
011	Create Account button visibility and style	Load the signup screen	The signup button is visible and styled	Button displayed properly	Pass	-
012	Hover effect on the Create Account button	Hover the mouse over the login button	Button scales up slightly	Button enlarges with animation	Pass	-
013	Try to make an account with empty fields	No values in the input	Shows an error saying	As expected	Pass	-

Feature/ Module: Sign up section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
		fields	“All fields are required!”			
014	If the client changes or removes the input attributions like name, id	Change and remove input attribution through DevTool	Shows an error saying “All fields are required!”	As expected	Pass	-
015	Email validation	Type a non-Email value for the email	Shows an error saying “Invalid email!”	As expected	Pass	-
016	Check if the system takes the existing email	type “admin@gmail.com”	Shows an error message saying “Email is already in use.”	As expected	Pass	-
017	Check the character limitation in the email	Type “l x 51 @gmail.com”	Shows a warning message saying “Display email is too long..”	As expected	Pass	-
018	Password matching	Type two different passwords 123456 654321	An error message saying “Passwords do not match”	As expected	Pass	-
019	Type less than 6 characters for the password	type 1234	An error message saying “Password is too short”	As expected	Pass	-
020	Type more than 1024	type “Lorem	An error	As expected	Pass	-

Feature/ Module: Sign up section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
	characters for passwords	Ipsum Generator" * 1200	message saying "Password is too long…"			
021	Check the character minimum limitation for the name	Type less than three characters "LLL"	An error message saying "Name is too short. "	As expected	Pass	-
021	Check the character maximum limitation for the name	Type more than 50 characters "L" x 50	An error message saying "Name is too long"	As expected	Pass	-
022	Type correct sign-up details, and try to log in to the web app, to check whether the DB works	Email: "test@gmail.com" Name: test Password: 123456789 Password: 123456789	Redirect to login page	As expected	Pass	-
023	Check the image in position	Reload the page	Image is appeared	As expected	Pass	-

Feature/ Module: Dashboard						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Verify background color updates correctly when switching between dark mode and light mode	Toggle theme switch	Background color changes to match selected mode	Worked as expected	Pass	-
002	Redirect unauthenticated users	Copy the dashboard URL and open it in a different browser	Redirects to /login screen	Worked as expected	Pass	-
003	Verify hamburger menu button functionality	Tap the hamburger menu icon	Verify the element resizes correctly according to the expected behavior	Worked as expected	Pass	-
004	Responsive grid	Switch to the developer tools and confirm that clicking the Toggle Fullscreen icon	Layout reflows smoothly on all screen sizes with no overlap.	Worked as expected	Pass	-
005	User menu actions	Profile/Settings/Logout	functions as expected	Worked as expected	Pass	-
006	Shows the title bar as Dashboard	Load the page	Shows as expected	Shows as expected	Pass	-
007	Show the latest course if the user has selected	Select a course from the My course section	Show the percentage of completion	As expected	Pass	-

Feature/ Module: Dashboard						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
008	Check the progress bar in the Continue Learning widget	Complete some lessons in My course Section	Percentage increases, and the progress bar also	As expected	Pass	-
009	Check the Learning widget at the first login	First login	Shows “No Course selected”	As expected	Pass	-
010	Check the number of problems solved in the Problem Solved widget	Reload the page, and first login	The progress bar is empty, and the down test is 0 questions completed	As expected	Pass	-
011	Check the progress bar in the Problem Solved widget to see if it increases when the user completes one	Complete some questions	The progress bar increases, and the number under the progress bar increases	As expected	Pass	-
012	To check when the user clicks on the Continue Learning, redirects to the relevant course	Click on that	Redirect to the relevant Course	As expected	Pass	-
013	Check whether the course details are changing when the user changes the course	Select a different course “Introduction to JavaScript”	Course details in the Continuing Learning widget change	As expected	Pass	-
014	Check the strike	Stay two days consistently using (adding question, doing	Change the “No current streak….” in the Streak widget, get	As expected	Pass	-

Feature/ Module: Dashboard						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
		challenges …)	the value 2			
015	Check whether the system shows the question that has the highest number of answers	Select one question, and give answers. Select the question called “Test 2” for this	It appears in the Community Buzz widget as “Top Question “test 2”	As expected	Pass	-
016	Check whether the leaderboard works or not	Creates two different accounts and does some challenges vidunothap@gmail.com - 2 challenges Pasindu Vidunitha - 1 challenges	Shows both names with relevant rank numbers, and font colours	As expected	Pass	-
017	Check whether users can get Bronze Coins, Silver Coins and Gold Coins	Do 5 from each	Increases by 1 for each 5 challenges 5 easy = 1 Bronze coin 5 medium = 1 silver coin 5 hard = 1 Gold coin	As expected	Pass	-
018	Check the cursor changes to a pointer from the default when hovering over each widget	Hover over widgets	Changes to the pointer from the cursor	As expected	Pass	-
019	Theme toggling	Click on the theme icon	change from dark to light and light to	As expected	Pass	-

Feature/ Module: Dashboard						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
			dark			
020	Mobile responsiveness	Through inspect, changes to the mobile view	Layout should be adjusted	As expected	Pass	-

Feature/ Module: My Course section - All Courses page						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Shows the sub navigation bar	Load the page	Shows the sub navigation bar at the top	As expected	Pass	-
002	Sub Navigation bar functionality	Click on the Courses, Completed Courses, and AI-generated Courses	Change the page according to the menu, and change the colour of the text that was clicked from lighter to darker	As expected	Pass	-
003	Checks the animation of the Course card	Hover over the card	A play button appears, and the background becomes darker	As expected	Pass	-
004	Check the functionality of the Course cards	Click on the cards.	Redirect to relevant courses	As expected	Pass	-

Feature/ Module: My Course section - All Courses page						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
005	Search bar functionality (search bar for both All Courses and Completed Courses; therefore, it gives the same result)	Type and search	Shows the relevant courses	As expected	Pass	-
006	Theme switching	Click on the theme icon	Change the theme	As expected	Pass	-
007	Mobile responsive	Through the inspect, change the device	Adjusts the layout	As expected	Pass	-

Feature/ Module: My Course section - Completed Courses						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Checks whether there are any courses, without completing them	Click on the completed Course in the sub navigation bar	Empty Page	As expected	Pass	-
002	Checks whether completed courses are showing in the Completed Courses	Complete the introduction to Python course	The relevant completed course is there	As expected	Pass	-
003	Theme switching	Click on the theme icon	Change the theme	As expected	Pass	-
004	Mobile responsive	Through the inspect, change the device	Adjusts the layout	As expected	Pass	-

Feature/ Module: My Course section - Lesson Page						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Checks the code blocks' style in the lesson page	Click on a course	The code blocks should have a different font family, font colour, and a different background colour	As expected	Pass	-
002	Functionality of the arrow keys in the lesson page	Click on the buttons	Able to go to the next lesson and the previous lesson	As expected	Pass	-
003	Search functionality in the lesson page	Type something in the search bar	Keywords are highlighted	As expected	Pass	-
004	Functionality of the Done Button in the lesson page	Click on the button	Move to the next lesson; if it is the last lesson, it stays the same, and the percentage increases	As expected	Pass	-
005	Read aloud the capability of the lesson page	Click the button that has a wave icon	Start to read the text on the screen, and when you click it again, stop reading	As expected	Pass	-
006	Functionality of the Open editor button	Opens the code editor with the relevant programming language.	can see the code editor	As expected	Pass	-

Feature/ Module: My Course section - Lesson Page						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
007	Theme switching	Click on the theme icon	Change the theme	As expected	Pass	-
008	Mobile responsive	Through the inspect, change the device	Adjusts the layout	As expected	Pass	-

Feature/ Module: My Course section - Code Editor						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Theme switching	Click on the theme icon	Change the theme	As expected	Pass	-
002	Mobile responsive	Through the inspect, change the device	Adjusts the layout	As expected	Pass	-
003	Programming language name	Relaod	Correct programming language name	As expected	Pass	-
004	Check the compiler	Run some codes in different languages	Get output	As expected	Pass	-
005	Character limitations in the code	Type more than 15000 characters	A warning message “Input too large (max	As expected	Pass	-

Feature/ Module: My Course section - Code Editor						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
			15,000 chars). “			
006	Character limitation in the AI chat	Type more than 1500 characters	A warning message” message too long…”	As expected	Pass	-
007	Use an input value and test the compliler	type”x = input(">>") print(x)[redacted] “ in the code edit, and type the value “d” in the input	print d in the output	As expected	Pass	-

Feature/ Module: My Course - AI generated Course						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Check that the instructions are showing when there is no created courses	Reload	Showing instructions	as expected	Pass	-
002	Plus mark functionality	Click on it	The background becomes darker, the warning message shows that code editor is not	as expected	Pass	-

Feature/ Module: My Course - AI generated Course						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
			providing, and the input bar appear			

Feature/ Module: Practice Hub Section - Uncomplete Questions						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Started task tracking	Start “Sum of List”	Status = Started	As expcted	Pass	-
002	Not started tasks	Leave “Prime Number Generator” untouched	Status = blank	Status shows blank	Pass	-
003	Mixed task states	Have both started & not started	Correct mix displayed	Matches expected states	Pass	-

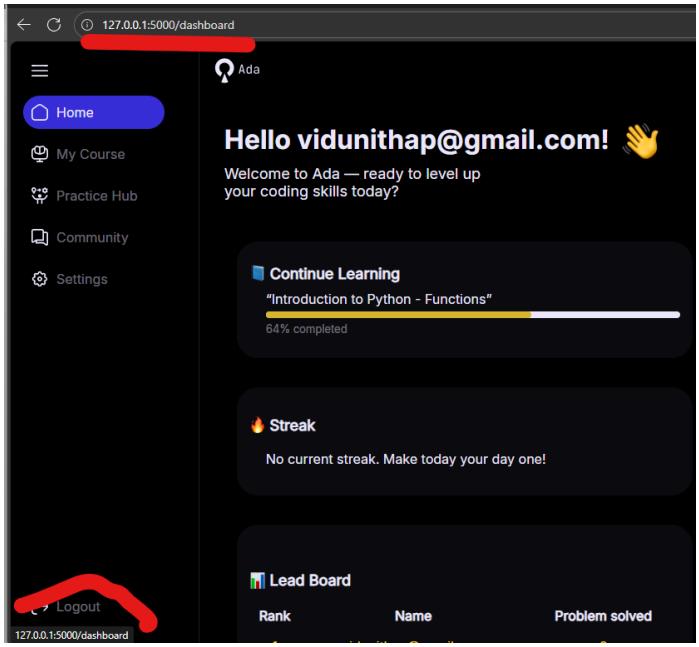
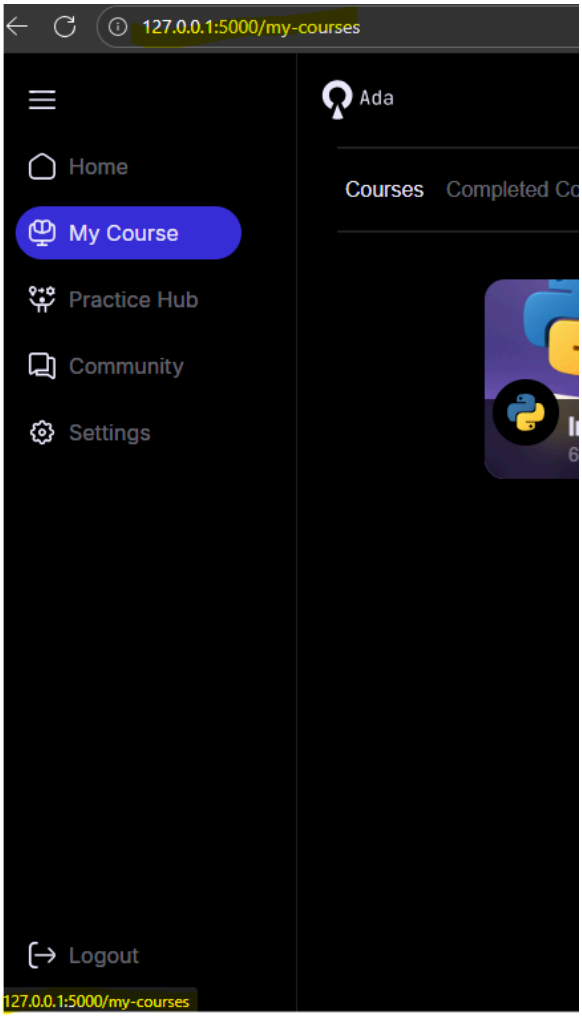
Feature/ Module: Practice Hub Completed Tasks						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Verify completed task appears	Complete “Print Hello World”	Task marked completed	Task displayed in list	Pass	-
002	Multiple completions	Complete 2 tasks	Both show in list	Both listed with timestamps	Pass	-
003	Timestamp validation	Check completion	Shows correct	Date/time correct	Pass	-

Feature/ Module: Practice Hub Completed Tasks						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
		time	date/time			

Feature/ Module: Community Section						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	All the question are there	Reload	With number of answers, posted user, profile imae are ther	As excpected	Pass	
002	Functional test (normal input)	Enter a valid question (100 chars) Question posted successfully Question posted successfully	Question posted successfully	””	Pass	-
003	Boundary testing (minimum, maximum, below, above, empty spaces) both question and answers fields	29 chars (too short), 30 chars (min valid), 30,000 chars (max valid), 30,001 chars (too long), " " (only spaces)	Accept valid (30–30,000). Reject invalid (<30, >30,000, empty)	””	Pass	-

Feature/ Module: Settings (Profile & Password)						
Test ID	Test Description	Input / How are you testing it	Expected Output	Actual Output	Pass/ Fail	Notes
001	Profile details display	View settings	Show username & email	Username/email displayed	Pass	
002	Password change valid	Enter old + new password	Password updated	Update success	Pass	-
003	Password change invalid	Wrong current password	Error message shown	Error shown	Pass	
004	Delete account	Click delete	Account removed	Account removed	Pass	-
005	Feedback adding	put a feedback and send	get the success message	same	Pass	
006	Boundary testing (all fields)	Tested minimum/maximum lengths, empty fields, and invalid formats across username, email, and password	Valid inputs accepted, invalid rejected	same	Pass	-

Data Intergirty Testing

Test ID: NAV-HOME-01 Goal: Ensure the Home link points to the dashboard. Method: Hovered on *Home* in the side navigation. Evidence: Browser status bar shows `127.0.0.1:5000/dashboard` (see screenshot). Result: Passed – Home correctly directs to the dashboard page.	
Test ID: NAV-COURSE-01 Goal: Ensure the My Course link goes to the correct page. Method: Hovered on *My Course* in the side navigation. Evidence: Browser status bar shows `127.0.0.1:5000/my-courses` (see screenshot). Result: Passed – Link correctly loads the *My Course* page.	

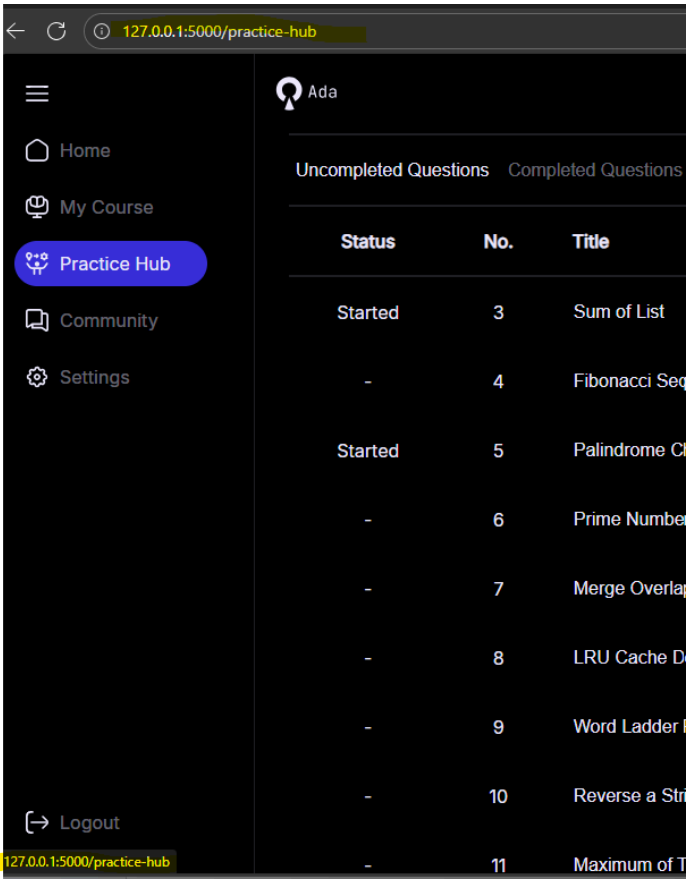
Test ID: NAV-PRACTICE-01

Goal: Ensure the Practice Hub link loads correctly.

Method: Hovered on *Practice Hub* in the side navigation.

Evidence: Browser status bar shows `127.0.0.1:5000/practice-hub` (see screenshot).

Result: Passed – Link correctly opens the Practice Hub page.



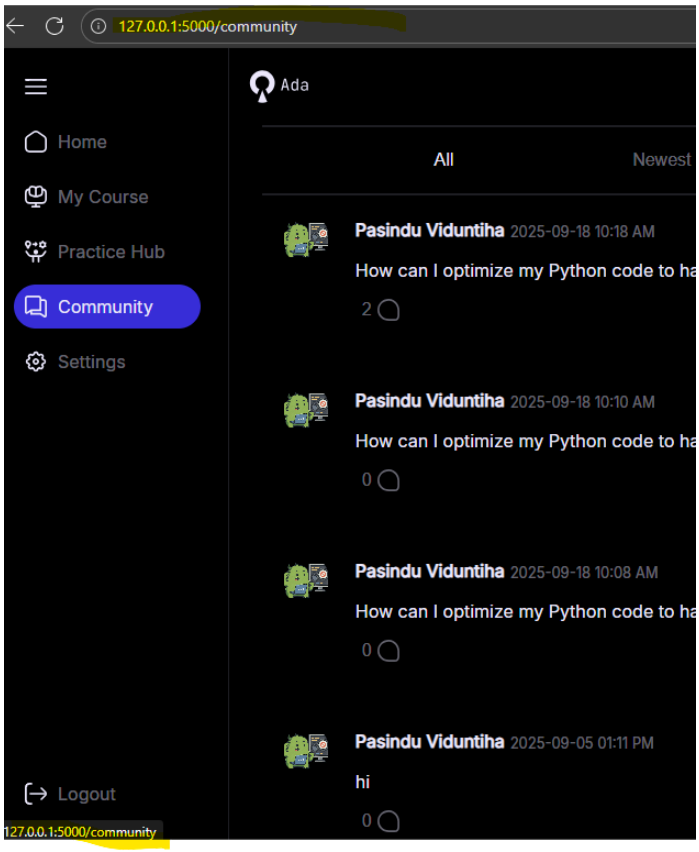
Test ID: NAV-COMM-01

Goal: Ensure the Community link loads correctly.

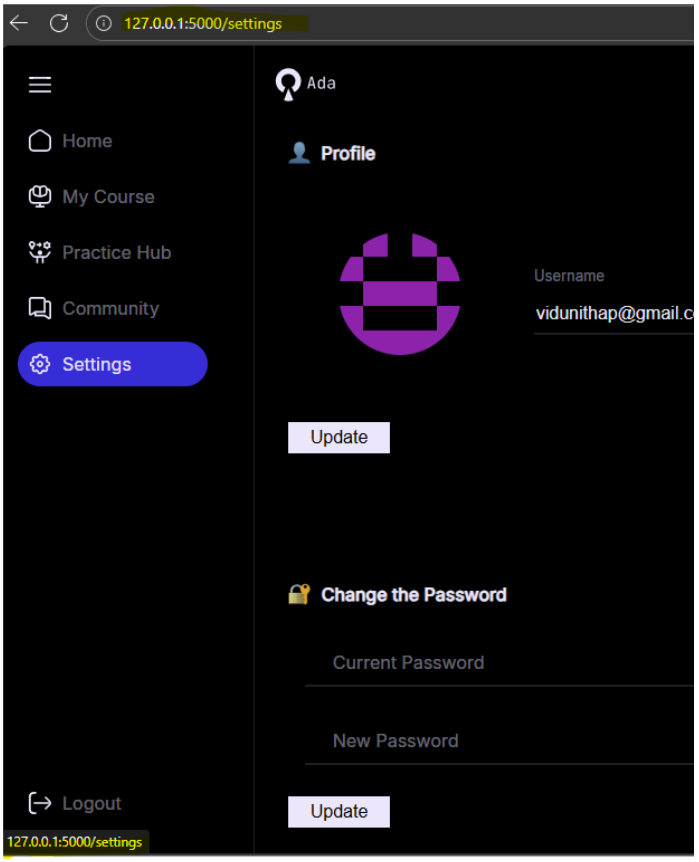
Method: Hovered on *Community* in the side navigation.

Evidence: Browser status bar shows `127.0.0.1:5000/community` (see screenshot).

Result: Passed – Link correctly opens the Community page.

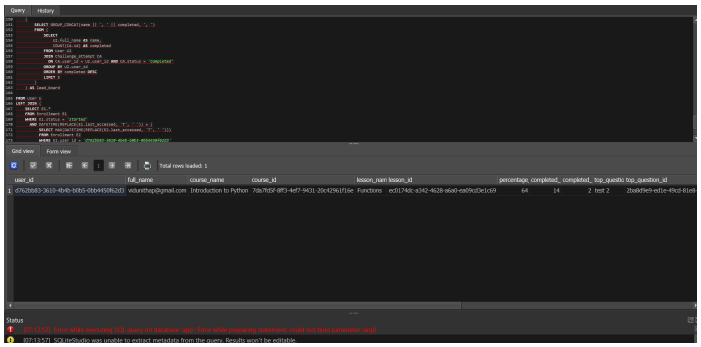


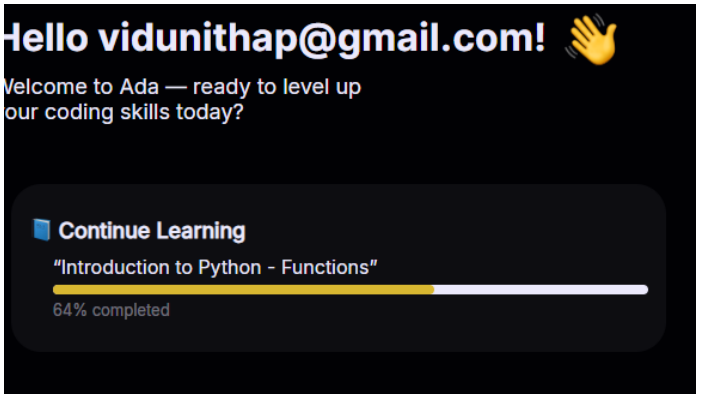
Test ID: NAV-SET-01
Goal: Ensure the Settings link loads correctly.
Method: Hovered on *Settings* in the side navigation.
Evidence: Browser status bar shows `127.0.0.1:5000/settings` (see screenshot).
Result: Passed – Link correctly opens the Settings page.

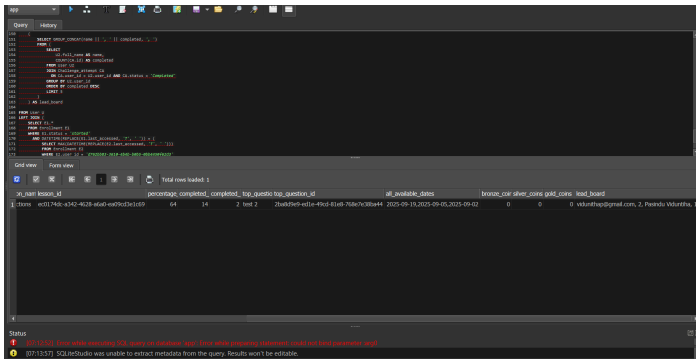


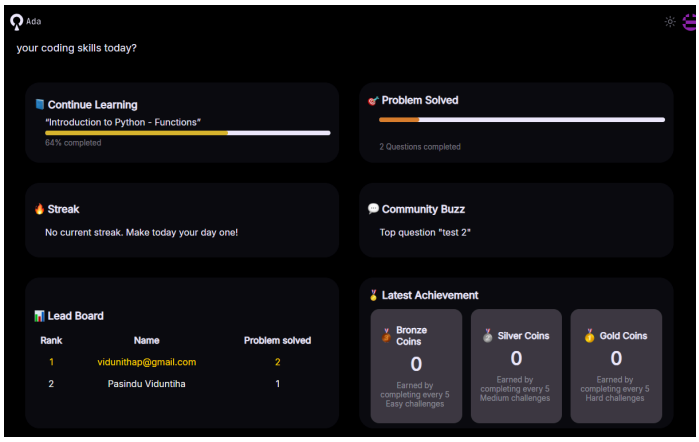
Test ID: DB-COURSE-01
Goal: Ensure course progress in the UI matches the database.
Method: Ran SQL query for user `d762bb83-3610-4b4b-b0b5-0bb4450f62d3` to fetch current course and progress.
Evidence:

- SQL shows *Introduction to Python – Functions* with 64% completed.
- Dashboard UI shows the same course with 64% completed (see screenshot).
  Result: Passed – Database values are correctly displayed on the dashboard.



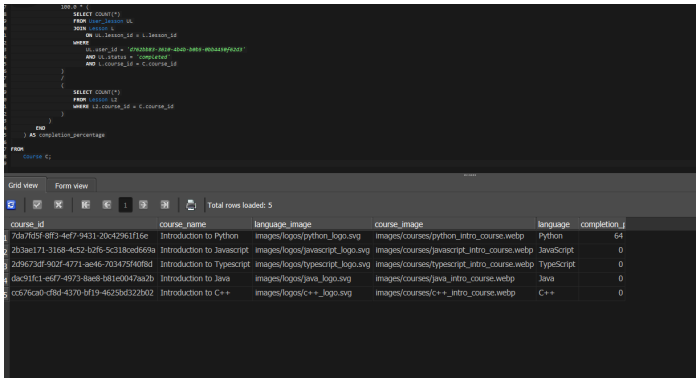






Test ID: DB-COURSE-02
Goal: Ensure all available courses and progress match database records.
Method: Ran SQL query to fetch course list + completion % for user `d762bb83-3610-4b4b-b0b5-0bb4450f62d3`.
Evidence:

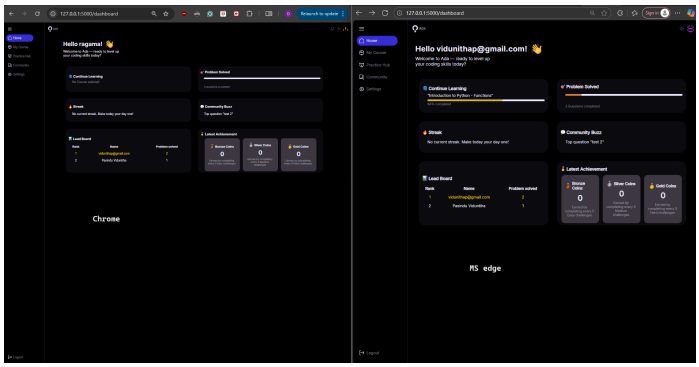
- SQL shows courses: *Introduction to Python, Introduction to JavaScript, Introduction to TypeScript, Introduction to Java, Introduction to C++* with progress values.
- UI displays the exact same courses with matching progress indicators (see screenshot).
  Result: Passed – Database course data is accurately displayed on the frontend.



course_id	course_name	language_image	course_image	language	completion_p
7da7fd5f-8ff3-4ef7-9431-20c42961f16e	Introduction to Python	images/logos/python_logo.svg	images/courses/python_intro_course.webp	Python	64
2b3ae171-3168-4c52-b2f6-5c318ced669a	Introduction to Javascript	images/logos/javascript_logo.svg	images/courses/javascript_intro_course.webp	JavaScript	0
2d9673df-902f-4771-ae46-703475f40f8d	Introduction to Typescript	images/logos/typescript_logo.svg	images/courses/typescript_intro_course.webp	TypeScript	0
dac91fc1-e6f7-4973-8ae8-b81e0047aa2b	Introduction to Java	images/logos/java_logo.svg	images/courses/java_intro_course.webp	Java	0
cc676ca0-c8d-4370-bf19-4625bd322b02	Introduction to C++	images/logos/c++_logo.svg	images/courses/c++_intro_course.webp	C++	0

Test ID: WEB-BROWSER-01
Goal: Ensure dashboard data displays consistently across browsers.
Method: Opened the dashboard in Chrome and Microsoft Edge with the same user.
Evidence: Screenshot comparison shows:

- Same user email `vidunithap@gmail.com` displayed.
- Same leaderboard data (Rank 1 with 2 problems solved).
- Same achievements (0 bronze, 0 silver, 0 gold).
- Same streak status = “No current streak”.
  Result: Passed – Data is consistent across Chrome and Edge, confirming correct web rendering and no browser-specific data issues.



1. Chrome
2. MS edge

Feedback Integration

User testing included two technical and two non-technical testers. I have recorded their exact feedback in quotes and described how I responded.

Technical person (My father) - 1

Feedback: *“The app should not allow simple passwords.”, “Add an email verification for the system”*
Rating: ★★★★★★★★☆☆ 8/10
My response: This is a great idea. I will try to do this in the next sprint if I have enough time to do so; otherwise, I will keep this idea for future improvements. Secondly, he said that adding an email verification to the authentication is also another great idea, and I am considering putting this idea into the future improvements section, because of the limited time.

Technical person (Kyle - A classmate) - 2

Feedback: *“Make the navigation bar draggable and reduce its width.”, “Tells the user what was wrong “ (in the login page)*

Rating: ★★★★★★★★★☆ 9/10

My response: This is a good UI idea, but with my limited time, I don't think that I will be able to do this; however, I will keep this idea for future improvements. In the sign-up page, he said that it tells what the user has gone wrong, for example, the Username is not correct, or the Password is not correct. I thought about this when I was developing the signing page, but I thought that this would explore user information on the client side. If someone is trying to hack a user account, it will be easy for them to get the user data. Therefore, I will keep my current error message. Furthermore, he said that a hovering effect should be applied to the widgets in the dashboard to make they clickable, and he said that a tick should be placed in front of each completed lesson to inform the user that they have completed the lesson. I feel like this is a great UI improvement, but first of all, I have to complete all the core features of my application.

Non-technical person (Tathsilu - A Sri Lankan Friend) - 1

Feedback: *"Add a user guide to the AI-generating courses page, because it takes a little time to understand the system.",*
"Add the title Welcome to the login page",
"Keep the code editor in the lesson page, so we do not need to go back and forward"

Rating: ★★★★★★★★☆☆ 8/10

My response: The first impression of the application is nice. He said to change the title to "Welcome" on the login page instead of keeping the name of the application. I think it is good feedback because I checked a lot of modern login pages, most of the UIs have "Welcome " or some kind of greeting message; therefore, I am hoping to change it in the next sprint. Also, he said, keep the code editor on the lesson page. This was my previous UI idea, but as I need a responsive design, it is hard to keep all the widgets on the same page, and it is getting messy, and the widgets would be smaller, which affects user experience badly; therefore, I will neglect this idea. Furthermore, he said that he would add a user guide to the AI-generating courses page, because he said that it takes a little time to understand the system. To make the application more user-friendly, I did this in the 8th sprint. Also, he said that he will add a restriction for AI-generated courses, as current people can ask, how to cook, and AI is generating a course on that, but it is not relevant to my application theme; therefore, I added a restriction to the AI model in the 8th sprint to generate only technical-based courses.

Non-technical person (My mother) - 2

Feedback: "Add a brief explanation about the app on the landing page."

Rating: ★★★★★★★★★★ 10/10

Feedback: She said that the app was a superb tool for learning programming languages, and it was very user-friendly. For the improvements, she said that adding a brief explanation about the app in the Landing page, although I have added a brief explanation in the landing page, maybe it is not noticeable to users, so I will consider increasing the font size.

Overall, there is much positive feedback from both sides (technical and non-technical). I will try to add as many of these improvements to the application in the next sprint, but firstly, I should complete the core REST features of the application in the next sprint.

PROGRAMMING CONVENTIONS AND STANDARDS

To ensure my program is maintainable, consistent and easy to collaborate with others, I used Python programming conventions and standards throughout my Python code. I followed the PEP8 style guide. I used snake case for naming conventions (eg,- user_id). I tried my best to reduce the repetition of the code, using custom helper functions (eg, db_execute(), validate_email_address()). I used clear, meaningful variable names in my code. For the error handling, I used try/except or if statements. I used Flake8 and Pylint tools to check

my mistakes in my Python code. Below, I have shown an example of my Python code to show the indentation, descriptive naming, and error handling.

```python
def validate_email_address(email):
    try:
        email_info = validate_email(email, check_deliverability=False)
        if email_info:
            return "valid"
        else:
            return "invalid"
    except EmailNotValidError:
        return "invalid"
```

DESIGN CHOICE JUSTIFICATION

During development, I tried various technologies and methods before deciding on the suitable options for my application. These choices were made based on efficiency, maintenance and usability. At the beginning of this project, my backend code was in one Python file. Later, as the project became more complex and bigger, I chose to divide the code into blueprints. This helped me a lot because I can focus on specific parts when I have an error, and it will also help with future maintenance. Initially, I handled SQL joins with Python, but this increased the code complexity, reduced the readability of the code, and reduced the performance of the application. Then I decided to use joins directly in SQL code. I used a blue colour-based theme for my whole application because it gives trust, relaxation, less stress, intelligence and stability according to the psychological resources. This is best for a developer-focused interface; this aligns with other developer tools like VS Code, GitHub. In the early stages, I wrote a long block of inline code, which increased the repetition and was hard to maintain; therefore, I created a reusable helper function like db_execute().

HCI CONVENTIONS JUSTIFICATION

When designing the interface, I applied HCI conventions to make the system clear, accessible, and easy to use across different devices (PCs and Mobile devices). I kept my designs very minimalistic and clean, as evidenced by my feedback. I added a light theme and a dark theme to my application for user preference. I used different kinds of fonts for the landing page to keep the user, but after logging in to the application, I used the “Inter” font throughout the application. I used a consistent font throughout the application to keep my designs minimalistic and reduce overwhelming feelings for the end users. By following these HCI guidelines, I ensure that my application is both user-friendly and accessible. I used icons and language that most users are familiar with (eg, in the navigation bar, I used Home, My Course, Settings, etc). After the user signed up for the application, I used the same format for the application (base.html). I ensured to stick to my colour palette always. This shows that I applied HCI conventions such as consistency, minimalism, and flexibility, which align with Nielsen’s usability heuristics.

IMPLICATIONS OF THE DIGITAL OUTCOME

Database Implications

Ethical – Protecting User Accounts

Meaning: Protecting user accounts means keeping login details private and secure, so users' personal information is not stolen or exposed to the outside world.
How I expressed it: If passwords are stored in plain text, they can be hacked easily, which is unfair and affects user privacy also. To reduce these risks, I used Argon2 password hashing, which makes the passwords more secure. I chose this because this is one of the best ways to protect passwords compared to other methods like hashing and encrypting. I have not yet added stronger password rules due to time limits, but that would be a future improvement to make the system more secure.

Ethical – Account Deletion

Meaning: Account deletion relates to respecting the users' choice to leave a system. Ethically, if a user wants to delete their account, that means all their personal information should be removed.
How I expressed it: Some systems are doing only 'soft delete' accounts, meaning the data is hidden, but still keeping users' information in databases. I chose to use permanent deletion, so when a user deletes their accounts, all the data relevant to that user will be deleted. This respects their decision and supports privacy. In the future, I could add a short recovery period (like 7 days) in case the deletion was a mistake, but the main principle is to respect the user's choice.

Web Implications

Legal – Copyright and Attribution

Meaning: Copyright means respecting ownership of creative work, such as icons, images, and written content. Using someone else's work without permission can lead to legal issues.
How I expressed it: I used icons from Flaticon, which are free to use with attribution. To follow this rule, I added credit to them in my project documentation and my landing page (footer). I also used AI-generated images and AI course content, which do not have copyright restrictions. This shows my application respects copyright law and avoids plagiarism.

Social – Usability and Accessibility

Meaning: A social implication is how easy the system is to use for different kinds of users. If the user interface is confusing, users may struggle to use the application.
How I expressed it: Feedback from non-technical users helps a lot, because they don't have much development experience, but for my application, most users have basic knowledge of programming, but I used people who do not have any technical background, because I need to make the application as easy to use as possible. Feedback from these non-technical users helped me a lot to create an accessible, user-friendly application for a wider range of users.

CREDITS

- Icons designed by Flaticon (www.flaticon.com)
- Some images generated using ChatGPT (OpenAI)
- Course content generated with the help of ChatGPT AI