**Cloud-Based Development Environment Using GitHub Codespaces and AWS Services**

*A Course Project Report Submitted in partial fulfillment of the course requirements for the award of grades in the subject of*

**CLOUD BASED AIML SPECIALITY**

**(22SDCS07A)**

by

**Rakesh Penugonda**

**2210030252**

*Under the esteemed guidance of*

**Ms. P. Sree Lakshmi**

Assistant Professor,

Department of Computer Science and Engineering



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**K L Deemed to be UNIVERSITY**

*Aziznagar, Moinabad, Hyderabad,*

*Telangana, Pincode: 500075*

April 2025

**K L Deemed to be UNIVERSITY**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

*Certificate*

This is Certified that the project entitled **“Cloud-Based Development Environment Using GitHub Codespaces and AWS Services”** which is a experimental &/ theoretical &/ Simulation&/ hardware work carried out by Rakesh Penugonda - 2210030252, in partial fulfillment of the course requirements for the award of grades in the subject of  **CLOUD BASED AIML SPECIALITY**, during the year **2024-2025**. The project has been approved as it satisfies the academic requirements.

**Ms.P.Sree Lakshmi Dr. Arpita Gupta**

**Course Coordinator Head of the Department**

**Ms. P. Sree Lakshmi**

**Course Instructor**

**CONTENTS**

Page No.

1. Introduction

2. AWS Services Used as part of the project

3. Steps involved in solving project problem statement

4. Stepwise Screenshots with brief description

5. Learning Outcomes

6. Conclusion

7. References

1. **INTRODUCTION**

*Mini Project Title*

In the evolving landscape of software development, the demand for portable, scalable, and collaborative development environments has significantly increased. To meet this demand, cloud-based development solutions have become increasingly popular. This project, titled “Cloud-Based Development Environment Using GitHub Codespaces and AWS Services”, is a practical implementation that demonstrates how developers can seamlessly practice and build applications using cloud-native infrastructure and modern developer tools.

The core concept behind this platform is to provide users with web-based coding problems where each problem is bundled as a template hosted on a GitHub repository. When a user chooses a question to solve, a GitHub Codespace is automatically created by forking that template repository into the user's own GitHub account. This allows users to code in an isolated, fully-configured environment tailored to the problem without setting up anything locally.

From a backend perspective, this platform is hosted on an AWS EC2 instance, where both frontend and backend application code is deployed. This instance serves the interface and APIs required for interaction with GitHub and manages user sessions, problem listings, and Codespace triggers. This deployment strategy ensures high availability, flexibility, and low cost while providing complete control over the server infrastructure.

This project acts as an innovative solution for students, developers, and educators, enabling them to engage with problem-solving directly from the browser in a real-world development setup—without worrying about system compatibility or software installations. It bridges the gap between problem descriptions and hands-on development by tightly integrating GitHub and AWS services in a seamless workflow.

1. **AWS Services Used as part of the project**

To bring the cloud-based coding platform to life, Amazon Web Services (AWS) was utilized to host and manage the infrastructure components required for reliable, scalable delivery. While GitHub Codespaces handled the development environment side, AWS played a key role in making the platform accessible and production-ready. Below are the primary AWS services used in the project:

1. **Amazon EC2 (Elastic Compute Cloud)**

Amazon EC2 is the backbone of the platform's hosting infrastructure. The web application, including both the frontend (React/HTML/CSS/JS) and the backend (Node.js/Express), is deployed on an EC2 instance. This virtual server allows full control over the operating system and environment configurations.

The entire platform, including the website available at [neocode.rakeshp.me](https://neocode.rakeshp.me" \t "_new), is served from the EC2 instance.Frontend build files and backend APIs were securely transferred to the instance and launched using a production-ready setup.EC2 provides high availability and flexibility to scale or upgrade as needed.

1. **AWS Security Groups (EC2 Firewall)**

Security Groups were configured to control the traffic to and from the EC2 instance. Only essential ports such as HTTP (port 80) and SSH (port 22) were allowed for web traffic and secure server access.Ensures that only authorized users (e.g., the developer) can SSH into the EC2 instance.Protects the deployed web application from unauthorized access and attacks.

1. **Steps Involved in Solving Project Problem Statement**

The goal of the project was to develop an online cloud-based coding platform that allows users to practice web development problems directly in the browser using GitHub Codespaces, while hosting the platform using AWS EC2 for scalability and reliability. To achieve this, the development process was broken down into the following steps:

**Step 1: Identifying the Problem Scope and Requirements**

The project began with identifying key requirements:

Enable users to solve frontend/web development coding problems online. Use GitHub Codespaces for isolated and cloud-based coding environments. Automatically fork problem templates from a central GitHub repository. Ensure the platform is always available and can scale as needed.

**Step 2: Designing the Workflow Architecture**

A cloud-based workflow was designed with the following structure:

All coding problems are stored in a GitHub repository as problem templates. When a user selects a problem, a GitHub Codespace is triggered by forking the template into the user’s GitHub account. Users work on the problem directly in their GitHub Codespace environment. The main platform (frontend and backend) is hosted on an AWS EC2 instance.

**Step 3: Creating and Organizing GitHub Template Repositories**

Problem templates were carefully structured in a GitHub repository with:

A pre-configured devcontainer.json for Codespaces setup. An initial boilerplate code related to each problem. Problem metadata such as titles, descriptions, and tags for frontend rendering.

**Step 4: Developing the Platform Frontend and Backend**

The platform was developed with two components:

**Frontend**: Displays a list of problems and guides users to solve them using Codespaces.

**Backend**: Handles GitHub API integration to fork repositories and trigger Codespaces, manages problem metadata, and connects the frontend with dynamic content.

**Step 5: Integrating GitHub OAuth and Codespaces API**

To allow personalized experience:

GitHub OAuth was integrated so users could log in with their GitHub account. On selecting a problem, the backend uses GitHub API to, Fork the relevant template repo to the user’s account. Open a GitHub Codespace linked to that forked repo.

**Step 6: Deploying the Full Stack Application to AWS EC2**

After successful testing:

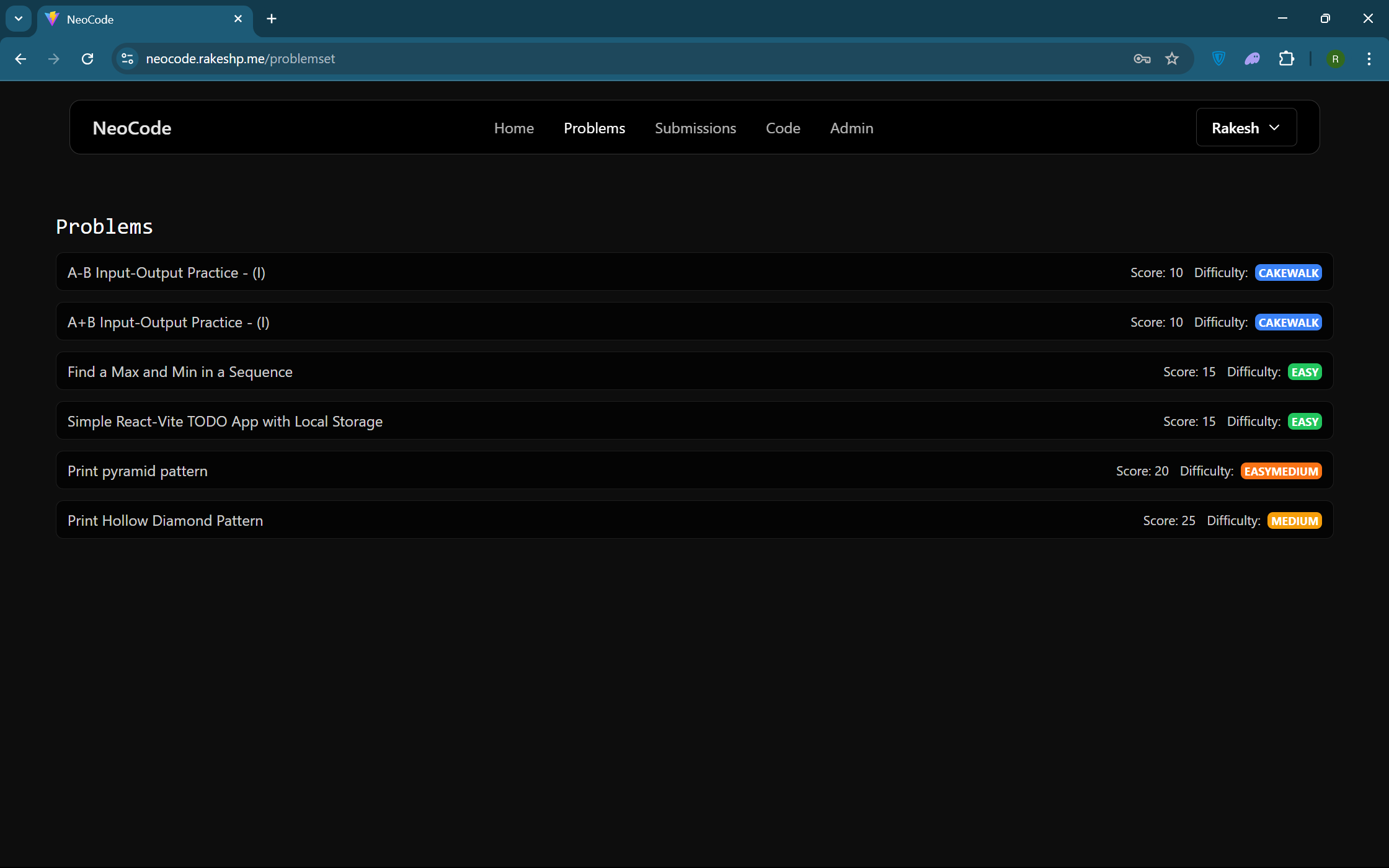
The complete web application (frontend + backend) was deployed to an Amazon EC2 instance. Production optimizations were applied such as using PM2 to run backend services and hosting static files using Nginx or directly through the Node server. Domain configuration (neocode.rakeshp.me) was mapped to the EC2 instance’s IP using Elastic IP.

1. **Stepwise Screenshots with Brief Description**

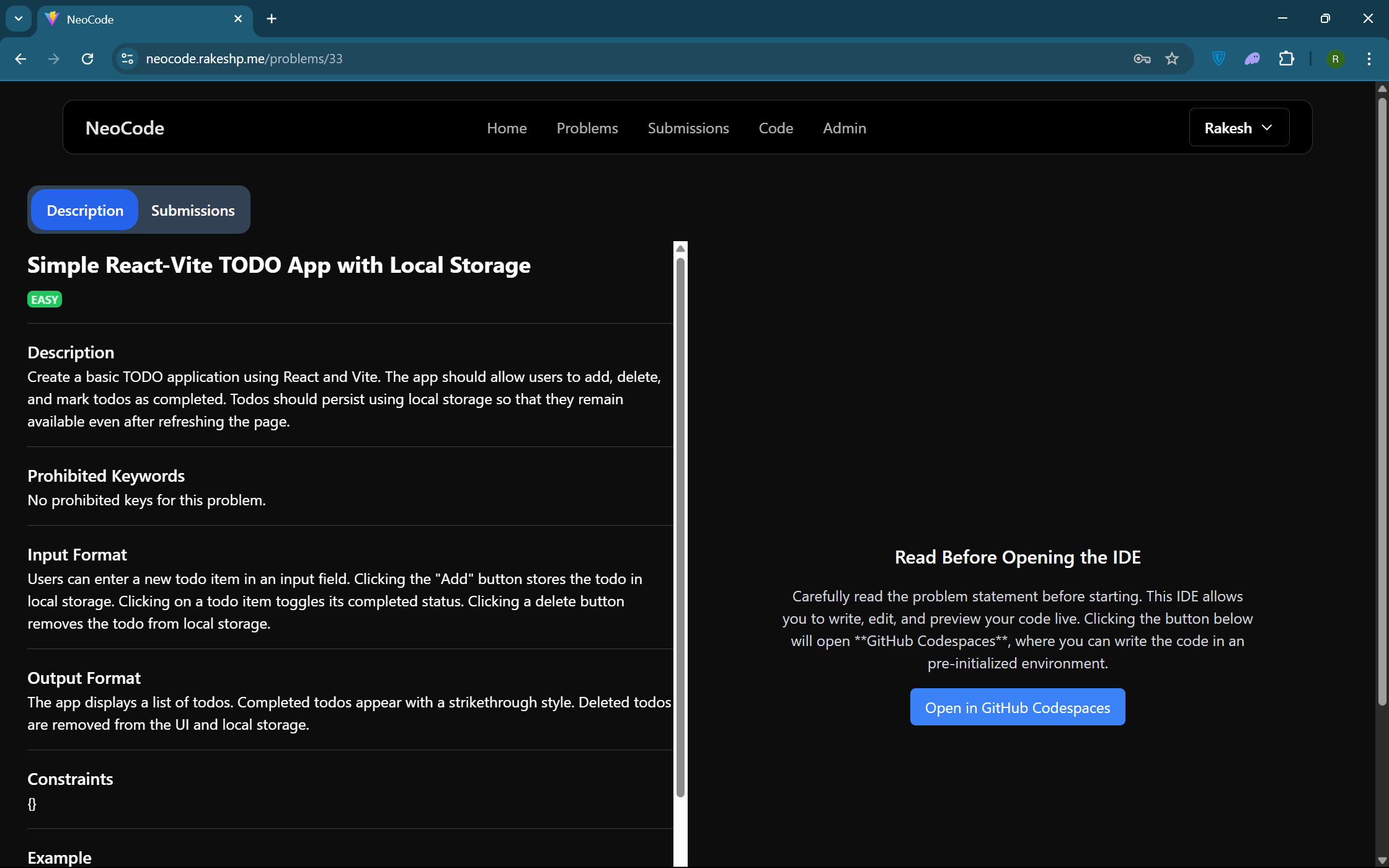
This section illustrates the step-by-step process of how a user interacts with the platform, from landing on the homepage to solving a coding problem using GitHub Codespaces. Each step is supported with a brief explanation and a placeholder for corresponding screenshots.

**Step 1: Homepage Display – Problem List Interface**

The homepage of the platform lists all available coding problems.

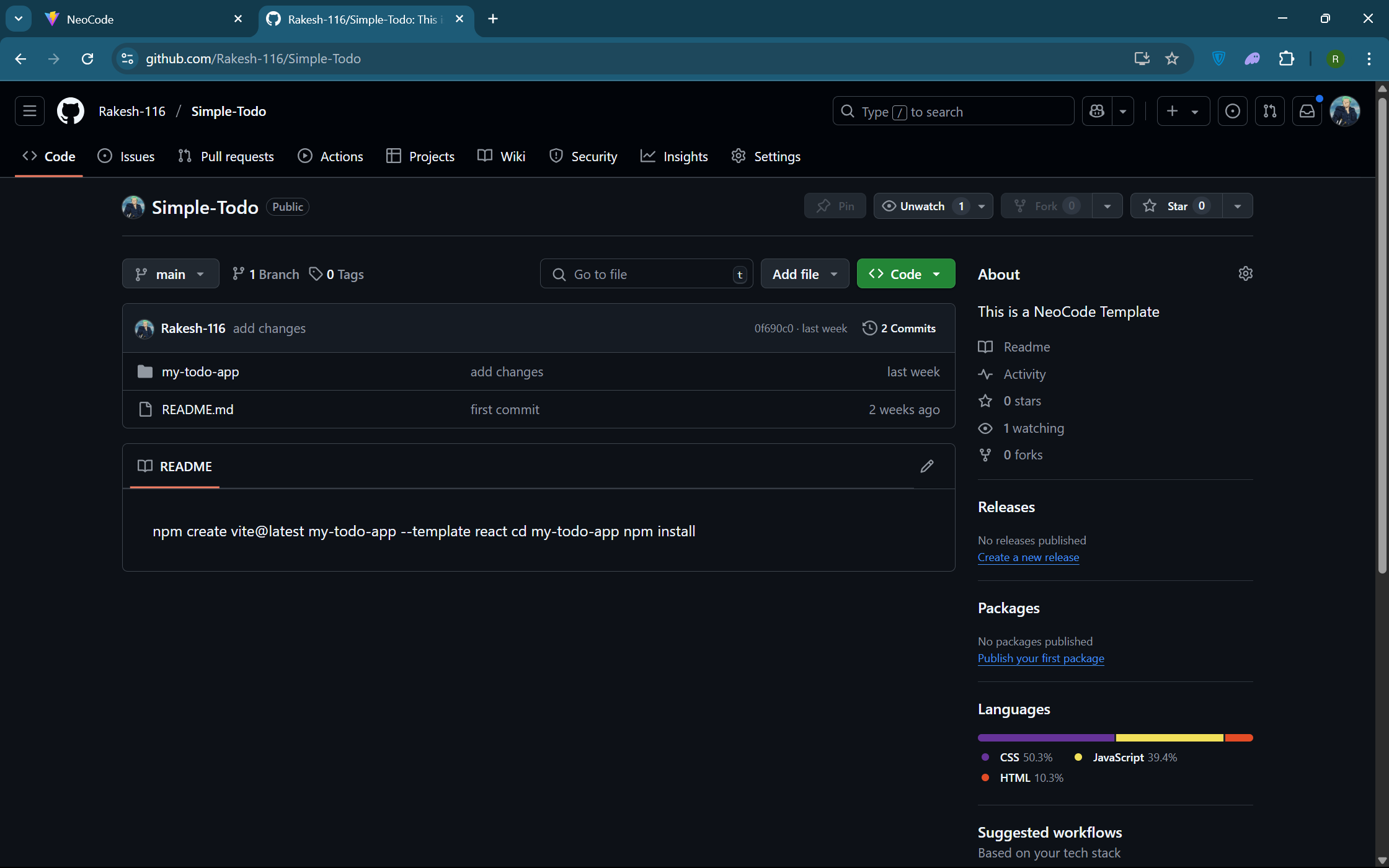


Each problem entry includes a title, short description, difficulty level, and a “Solve in Codespace” button. Users can browse and select any question they want to attempt.



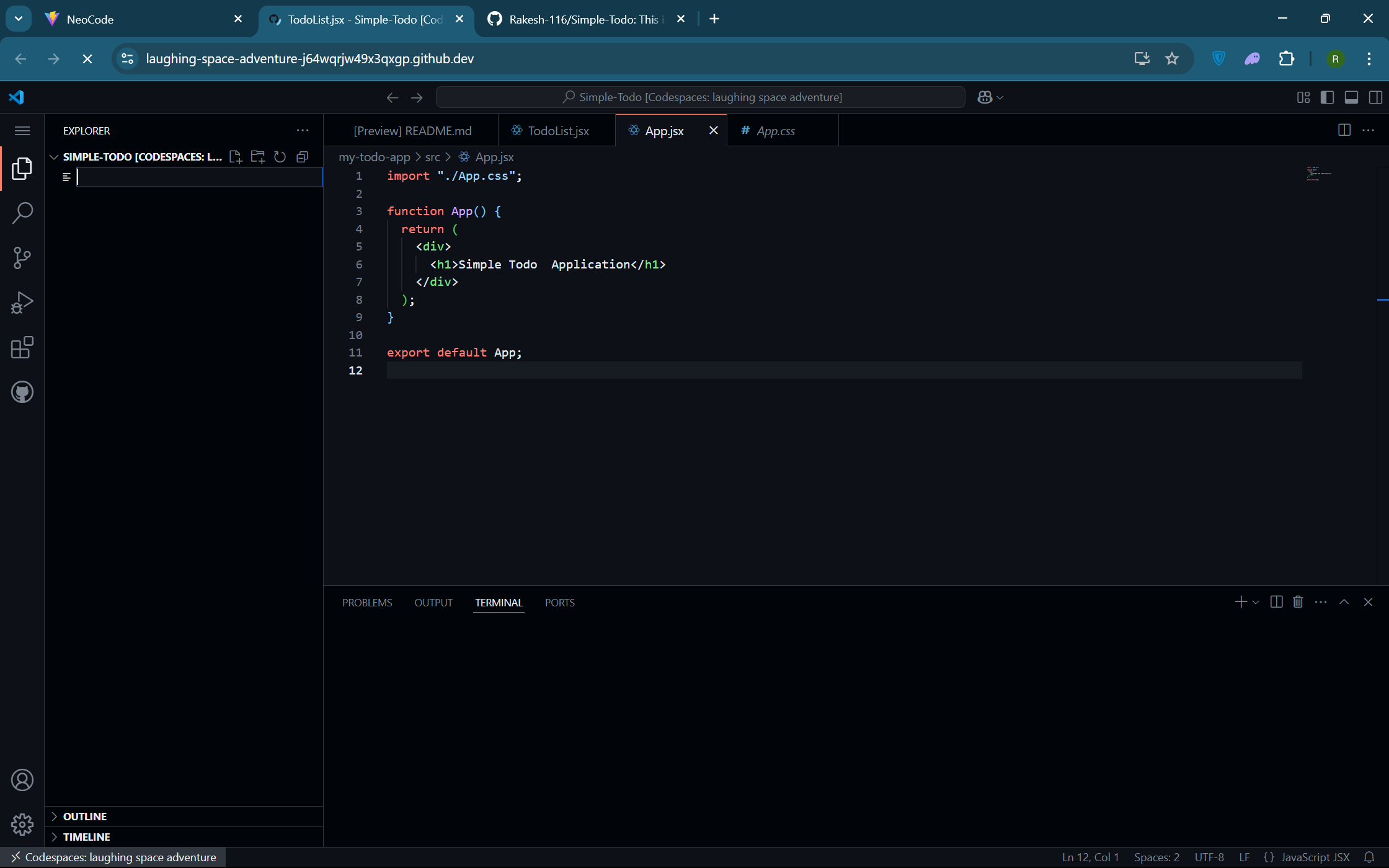
**Step 2: Selecting a Problem and Forking the Template Repo**

When a user selects a problem, the backend automatically forks the template GitHub repository for that problem into the user’s GitHub account. This fork contains pre-configured boilerplate code and Codespace setup (devcontainer.json, etc.).



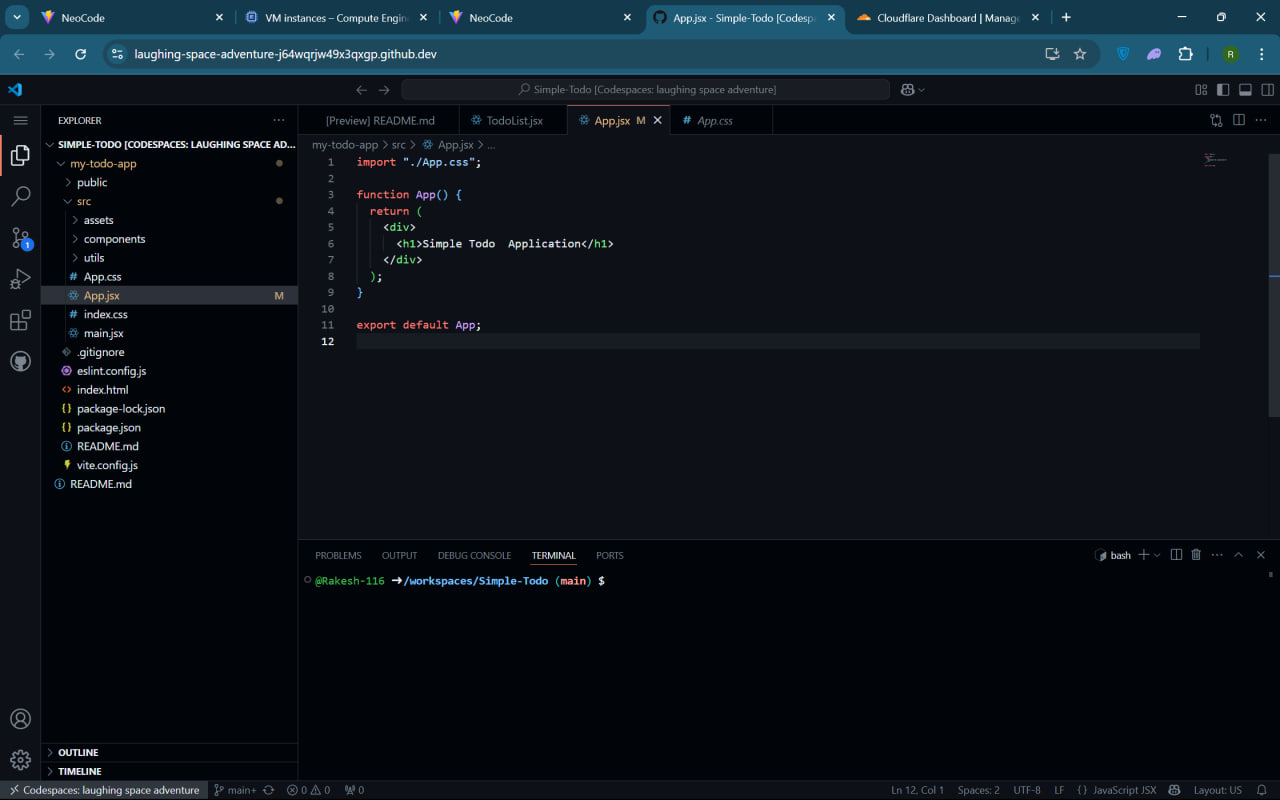
**Step 4: Launching GitHub Codespace**

After forking, the platform uses GitHub's Codespaces API to open the repo in a new Codespace environment. Users are redirected to the live Codespace where they can begin coding immediately—no local setup required.



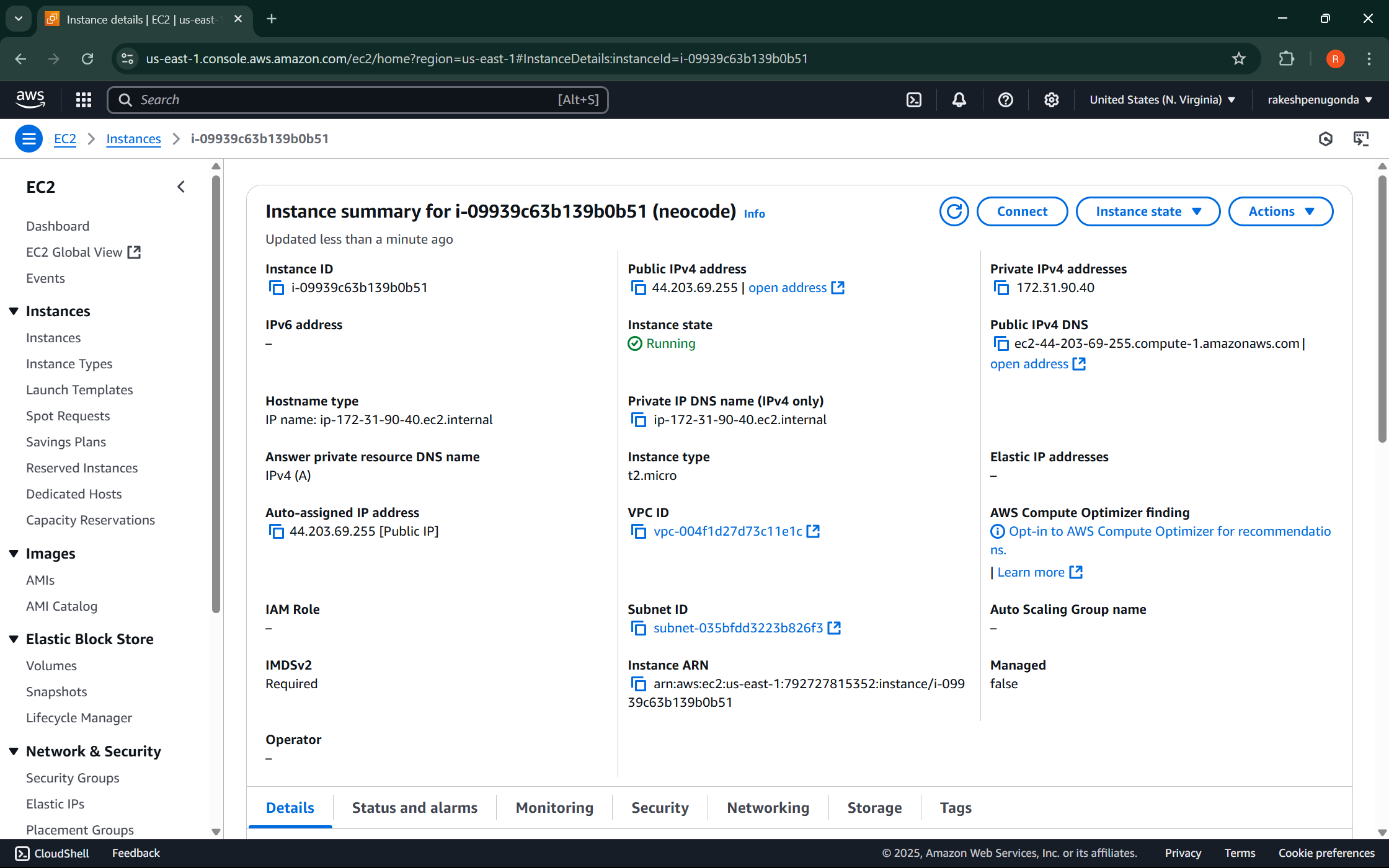
**Step 5: Inside the Codespace – Editing the Solution**

Users can now solve the coding challenge directly in the GitHub Codespace. Features like live preview (for frontend), terminal access, and extensions are available within the browser.



**Step 6: Deployed Platform on AWS EC2**

The full stack application (frontend + backend) was deployed to an EC2 instance. This instance runs continuously to serve users accessing <https://neocode.rakeshp.me.> You can optionally show screenshots of your file upload process to the EC2 server or terminal view of server running.



**Section 5: Learning Outcomes.**

This project provided a hands-on opportunity to work with modern, cloud-native tools to build a real-world, scalable application. Throughout the development of the online coding platform, several key technical and conceptual skills were learned and applied:

Learned how to configure and launch GitHub Codespaces using devcontainer.json files and template repositories. Understood how Codespaces provide isolated development environments directly in the browser without requiring local setup. Explored the GitHub API for automating repository forking and initiating Codespace sessions programmatically.

Built backend logic using Node.js/Express to automate GitHub actions like forking repos and managing problem templates. Developed APIs for dynamic problem retrieval, Codespace launching, and GitHub profile linking. Implemented secure backend communication with GitHub's REST API.

Designed an intuitive and clean user interface to display coding problems, initiate Codespaces, and manage session state. Used modern frontend libraries (likely React or plain JS) to build a responsive and user-friendly experience.

Gained hands-on experience launching an EC2 instance, configuring the environment, and deploying both frontend and backend applications. Learned how to manage firewall rules with AWS Security Groups and assign an Elastic IP for consistent access. Used tools like SCP/WinSCP to transfer files and PM2/Nginx to serve applications.

Set up a custom domain (neocode.rakeshp.me) using Cloudflare and mapped it to the EC2 server via DNS settings. Explored additional Cloudflare features like SSL, caching, and performance enhancements.

Learned how to build an end-to-end workflow where users can go from selecting a coding problem to solving it live in a containerized browser-based IDE. Understood the synergy between frontend/backend code, GitHub services, and AWS infrastructure.

**Section 6: Conclusion.**

The project “Cloud-Based Development Environment Using GitHub Codespaces and AWS Services” successfully demonstrates how modern cloud technologies can be combined to create a powerful, real-time, and scalable coding platform for developers and learners. By integrating GitHub Codespaces, the platform eliminates the need for local development setup, offering users an instant, browser-based IDE experience tailored to individual coding problems.

Through the use of template repositories, problem-specific boilerplate code is made instantly available, enabling users to start coding with minimal friction. The backend, deployed on an AWS EC2 instance, efficiently manages authentication, template forking, and Codespaces initiation, while the frontend provides an intuitive interface for problem selection and interaction.

Deploying the application on AWS EC2 ensured high availability and control, while Cloudflare was used to manage domain DNS resolution and enhance performance and security. This combination of tools and services provided a seamless development and deployment pipeline, from code preparation to execution.

The project provided valuable insight into cloud-based development environments, API automation, secure authentication, and DevOps deployment practices. It also showcased the real-world potential of integrating cloud infrastructure with developer tools to create collaborative and accessible learning platforms.

Overall, the project not only achieved its functional objectives but also served as a strong learning experience in the field of cloud computing, web development, and developer experience engineering.

**Section 7: References**

1. GitHub Codespaces Documentation - <https://docs.github.com/en/codespaces>
2. GitHub REST API Reference - <https://docs.github.com/en/rest>
3. AWS EC2 (Elastic Compute Cloud) Documentation - <https://docs.aws.amazon.com/ec2/>
4. AWS Elastic IP Documentation - <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html>
5. Cloudflare DNS Documentation - <https://developers.cloudflare.com/dns/>
6. Node.js Official Documentation - <https://nodejs.org/en/docs/>
7. PM2 Process Manager for Node.js - <https://pm2.keymetrics.io/docs/usage/quick-start/>
8. Frontend Deployment with Nginx on EC2 - <https://docs.nginx.com/nginx/admin-guide/web-server/>
9. Domain Setup on Cloudflare - <https://developers.cloudflare.com/fundamentals/get-started/setup/>