CodeIt: An Online Code Judge Platform Powered by AWS Cloud

Chirag Gulati
A25305222118
Amity University Punjab
Mohali, Punjab, India
chirag.gulati@s.amity.edu

Aashish Bhandari
A25305222162
Amity University Punjab
Mohali, Punjab, India
aashish.bhandari1@s.amity.edu

Arshia Chaudhary
A25305222154
Amity University Punjab
Mohali, Punjab, India
arshia.chaudhary@s.amity.edu

Prachi Goyal
A25305222153
Amity University Punjab
Mohali, Punjab, India
prachi.goyal@s.amity.edu

***Abstract*—CodeIt is an online coding platform and an innovative and cost – effective solution for assessing programming problems similar to those in LeetCode. The platform uses AWS services and combines a Flask backend running on Amazon EC2 with a static Amazon S3 web frontend. Users are able to submit code, which we then compile, execute and validate against given predetermined test case(s), allowing us to give an accurate assessment of the code. Designed to be scalable, efficient and simple to use, the platform is coded in such a way that developers and learners have a great environment to work in. An overview of the system architecture, through the implementation, challenges faced, and future enhancements is provided in this paper.**

***Index Terms*—AWS, Online Code Judge, Amazon S3, Amazon EC2, Flask Framework, Scalability, Deployment**

I. Introduction

In the era of software development, online coding platforms serve as both a learning and a practice tool, as well as a first point of recruitment for many software companies. Developers can improve their coding skills, solve genuine issues, and race up there with competitor in platforms like LeetCode [1], HackerRank [2], and CodeForces. Accurate and scalable code execution platforms rely on well engineered robust infrastructure and seamless integration of various building blocks. Yet often such huge resources and expertise are necessary.

To tackle these, CodeIt attempts to build around the strengths of AWS [3] [4] by developing a lightweight, low cost online coding judge. Backend functionality is performed on the platform with a Flask [5] framework, while hosting of the frontend is done with Amazon S3 [3]. The dynamic pieces are handled by the backend (running on an Amazon EC2 instance [4]), which can compile and execute user submitted code and check it against a set of specified test cases.

Research on cloud based online judge systems, especially the work by Xu et al. [6] shows the benefits of modular and scalable systems design have been largely adopted by the architecture of CodeIt. By combining these concepts, CodeIt creates a fully operable interface and makes the process of entering and executing code fast and trust-proof. Additional future enhancements could include the addition of live contests and collaborative coding sessions.

II. Related Work

Online coding platforms, such as LeetCode and CodeForces, are widely used for practice and competitive programming. On the one hand, LeetCode has tons of problems to solve and a large user-base [1], however, it's interface in some way overwhelms users. Although CodeForces offers the competitive environment, it has clunky user interface, taking away the user experience [7]. With the difference being that CodeIt is built to have a simple, clean and modern user interface, allowing the users minimal distractions on their way of using the application.

The reason we chose AWS as the plateform to deploy in, in terms of cloud based, was for the scalability, reliability and cost effectiveness. The backend of the app is managed using Amazon EC2 to manage virtual servers flexibly, and support running code in multiple programming languages [4]. This was a crucial decision because running code in different languages in AWS Lambda is not possible because it runs code in one environment, and in our case, that's in Python. As of now, placing the static frontend web page is being hosted on S3 due to its simplicity and low cost, and S3 is the best solution for hosting static content at a scale [3].

III. System Design and Architecture

The system architecture of CodeIt consists of two main components: the frontend and the backend. The frontend is a simple web page hosted on Amazon S3, while the backend is a Flask application running on Amazon EC2. The communication between the frontend and backend is handled through RESTful APIs.

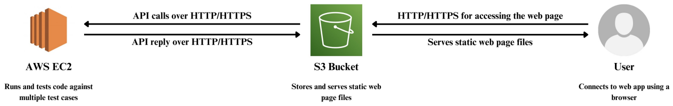

API calls over HTTP/HTTPS
API reply over HTTP/HTTPS
HTTP/HTTPS for accessing the web page
Serves static web page files
AWS EC2
Runs and tests code against multiple test cases
S3 Bucket
Stores and serves static web page files
User
Connects to web app using a browser

Fig. 1. System Architecture Flowchart

A. Frontend (S3)

The frontend of the application is built using HTML, CSS, and JavaScript. It is hosted on an S3 bucket and interacts with the backend through HTTP requests to the API endpoint.

B. Backend (EC2 - Flask App)

The backend of the system runs on an EC2 instance. The Flask application handles the logic for code execution, running test cases, and returning the results to the user. The backend communicates with the frontend via a REST API.

C. Components Overview

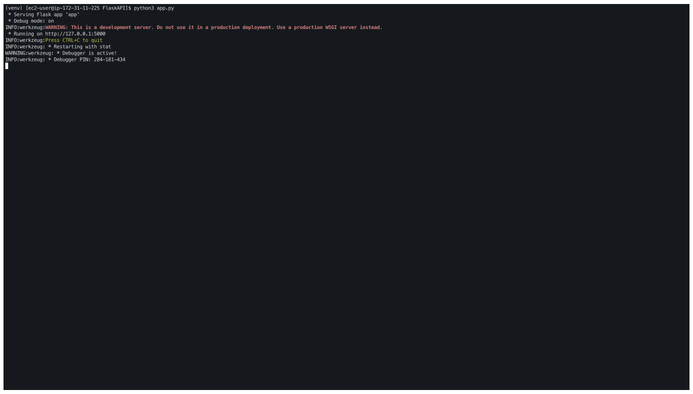
- **Frontend**: Static web pages using HTML, CSS, and JS, hosted on S3.
- **Backend**: Flask application on EC2 to handle code execution and testing.
- **AWS EC2**: Serves the backend Flask application.
- **AWS S3**: Hosts the static frontend.

IV. Implementation

A. Flask App Setup on EC2

The Flask application was set up on an Amazon EC2 instance running Amazon Linux OS. The following steps were followed:

- Connected to the EC2 instance via SSH terminal.
- Installed Flask and other necessary dependencies.
- Downloaded the app file (`app.py`) from GitHub using the `wget` command.
- Opened the necessary inbound rules on the EC2 instance to allow traffic to the app.
- The Flask app executes code submitted by users and compares it against predefined test cases, returning the results through the API.



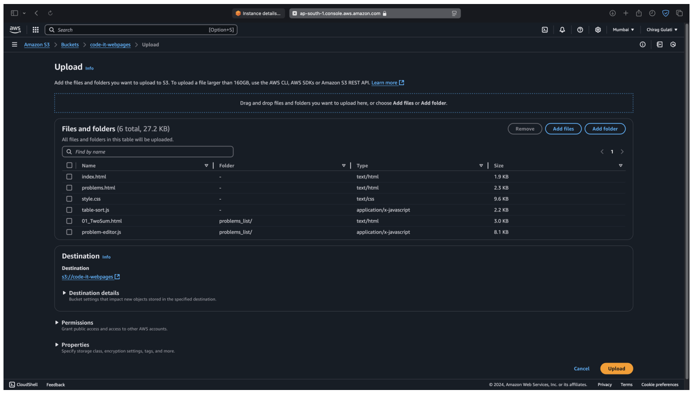
```
(venv) [ec2-user@ip-172-31-11-225 FlaskAPI]$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
INFO:werkzeug: * Restarting with stat
WARNING:werkzeug: * Debugger is active!
INFO:werkzeug: * Debugger PIN: 284-181-434
```

Fig. 2. Starting the Flask app on EC2

B. WebPages on S3 Bucket

The webpage was uploaded to the S3 bucket by enabling web hosting. The following steps were followed:

- Created an S3 bucket and enabled static website hosting.
- Uploaded the frontend files (HTML, CSS, JS) to the bucket.
- Configured the S3 bucket to serve the frontend web page.



Instance details...
ap-south-1.console.aws.amazon.com
aws
Search
[Option+S]
Mumbai
Chirag Gulati
Amazon S3 > Buckets > code-it-webpages > Upload

Upload Info

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDKs or Amazon S3 REST API. Learn more

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (6 total, 27.2 KB)

All files and folders in this table will be uploaded.

□	Name	Folder	Type	Size
□	index.html	-	text/html	1.9 KB
□	problems.html	-	text/html	2.3 KB
□	style.css	-	text/css	9.6 KB
□	table-sort.js	-	application/x-javascript	2.2 KB
□	01_TwoSum.html	problems_list/	text/html	3.0 KB
□	problem-editor.js	problems_list/	application/x-javascript	8.1 KB

Destination Info

Destination
s3://code-it-webpages

▶ **Destination details**
Bucket settings that impact new objects stored in the specified destination.

▶ **Permissions**
Grant public access and access to other AWS accounts.

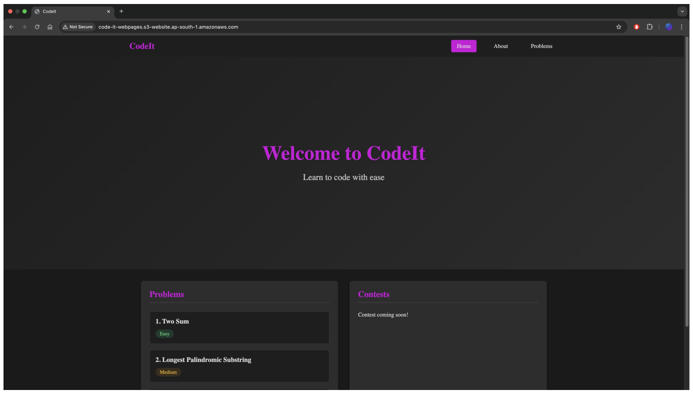
▶ **Properties**
Specify storage class, encryption settings, tags, and more.

Fig. 3. Uploading files to S3 bucket

V. Results and Testing

A. Website Screenshots

Below are some screenshots demonstrating the functionality of the website.



Welcome to CodeIt

Learn to code with ease

Problems

1. Two Sum
Easy

2. Longest Palindromic Substring
Medium

Contests

Contest coming soon!

Fig. 4. Website Home Page

B. Coding Problem Submission

A screenshot of the problem submission page, where the user enters their solution and submits it for evaluation.

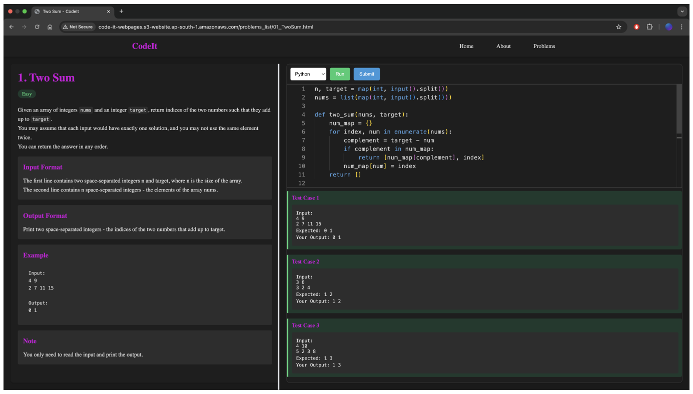
C. API Debug Logs

The following screenshot shows the debug logs of the API, detailing how the code is executed and compared against test cases.

VI. Challenges and Limitations

A. Security

One of the challenges faced during the development was ensuring the security of the backend API. Since it is accessible over the internet, we need to implement proper security measures to prevent malicious access.


Two Sum - CodeIt
Not Secure
code-it-webpages.s3-website.ap-south-1.amazonaws.com/problems_list/01_TwoSum.html
CodeIt
Home
About
Problems
Python
Run
Submit

1. Two Sum

Easy

Given an array of integers `nums` and an integer `target`, return indices of the two numbers such that they add up to `target`.
You may assume that each input would have exactly one solution, and you may not use the same element twice.
You can return the answer in any order.

Input Format

The first line contains two space-separated integers n and target, where n is the size of the array.
The second line contains n space-separated integers - the elements of the array nums.

Output Format

Print two space-separated integers - the indices of the two numbers that add up to target.

Example

```
Input:
4 9
2 7 11 15

Output:
0 1
```

Note

You only need to read the input and print the output.

```python
n, target = map(int, input().split())
nums = list(map(int, input().split()))

def two_sum(nums, target):
    num_map = {}
    for index, num in enumerate(nums):
        complement = target - num
        if complement in num_map:
            return [num_map[complement], index]
        num_map[num] = index
    return []

```

Test Case 1

```
Input:
4 9
2 7 11 15
Expected: 0 1
Your Output: 0 1
```

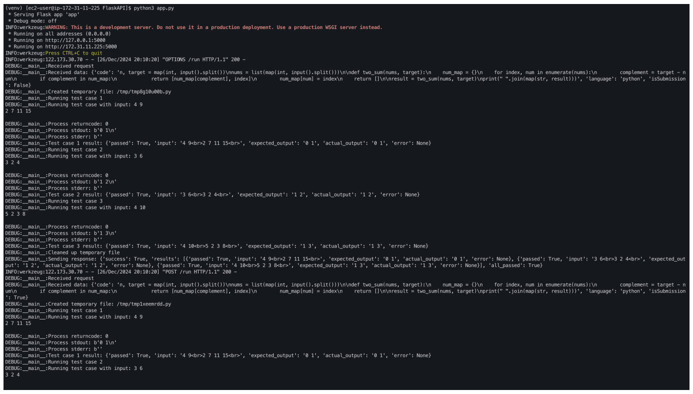
Test Case 2

```
Input:
3 6
3 2 4
Expected: 1 2
Your Output: 1 2
```

Test Case 3

```
Input:
4 10
5 2 3 8
Expected: 1 3
Your Output: 1 3
```

Fig. 5. Problem Submission with Answer



```
(venv) [ec2-user@ip-172-31-11-225 FlaskAPI]$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: off
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.31.11.225:5000
INFO:werkzeug:Press CTRL+C to quit
INFO:werkzeug:122.173.30.70 - - [26/Dec/2024 20:10:20] "OPTIONS /run HTTP/1.1" 200 -
DEBUG:__main__:Received request
DEBUG:__main__:Received data: {'code': 'n, target = map(int, input().split())\nnums = list(map(int, input().split()))\n\ndef two_sum(nums, target):\n    num_map = {}\n    for index, num in enumerate(nums):\n        complement = target - n
um\n        if complement in num_map:\n            return [num_map[complement], index]\n        num_map[num] = index\n    return []\n\nresult = two_sum(nums, target)\nprint(" ".join(map(str, result)))', 'language': 'python', 'isSubmission
': False}
DEBUG:__main__:Created temporary file: /tmp/tmp8g10u00b.py
DEBUG:__main__:Running test case 1
DEBUG:__main__:Running test case with input: 4 9
2 7 11 15

DEBUG:__main__:Process returncode: 0
DEBUG:__main__:Process stdout: b'0 1\n'
DEBUG:__main__:Process stderr: b''
DEBUG:__main__:Test case 1 result: {'passed': True, 'input': '4 9<br>2 7 11 15<br>', 'expected_output': '0 1', 'actual_output': '0 1', 'error': None}
DEBUG:__main__:Running test case 2
DEBUG:__main__:Running test case with input: 3 6
3 2 4

DEBUG:__main__:Process returncode: 0
DEBUG:__main__:Process stdout: b'1 2\n'
DEBUG:__main__:Process stderr: b''
DEBUG:__main__:Test case 2 result: {'passed': True, 'input': '3 6<br>3 2 4<br>', 'expected_output': '1 2', 'actual_output': '1 2', 'error': None}
DEBUG:__main__:Running test case 3
DEBUG:__main__:Running test case with input: 4 10
5 2 3 8

DEBUG:__main__:Process returncode: 0
DEBUG:__main__:Process stdout: b'1 3\n'
DEBUG:__main__:Process stderr: b''
DEBUG:__main__:Test case 3 result: {'passed': True, 'input': '4 10<br>5 2 3 8<br>', 'expected_output': '1 3', 'actual_output': '1 3', 'error': None}
DEBUG:__main__:Cleaned up temporary file
DEBUG:__main__:Sending response: {'success': True, 'results': [{'passed': True, 'input': '4 9<br>2 7 11 15<br>', 'expected_output': '0 1', 'actual_output': '0 1', 'error': None}, {'passed': True, 'input': '3 6<br>3 2 4<br>', 'expected_out
put': '1 2', 'actual_output': '1 2', 'error': None}, {'passed': True, 'input': '4 10<br>5 2 3 8<br>', 'expected_output': '1 3', 'actual_output': '1 3', 'error': None}], 'all_passed': True}
INFO:werkzeug:122.173.30.70 - - [26/Dec/2024 20:10:20] "POST /run HTTP/1.1" 200 -
DEBUG:__main__:Received request
DEBUG:__main__:Received data: {'code': 'n, target = map(int, input().split())\nnums = list(map(int, input().split()))\n\ndef two_sum(nums, target):\n    num_map = {}\n    for index, num in enumerate(nums):\n        complement = target - n
um\n        if complement in num_map:\n            return [num_map[complement], index]\n        num_map[num] = index\n    return []\n\nresult = two_sum(nums, target)\nprint(" ".join(map(str, result)))', 'language': 'python', 'isSubmission
': True}
DEBUG:__main__:Created temporary file: /tmp/tmp1xeemrdd.py
DEBUG:__main__:Running test case 1
DEBUG:__main__:Running test case with input: 4 9
2 7 11 15

DEBUG:__main__:Process returncode: 0
DEBUG:__main__:Process stdout: b'0 1\n'
DEBUG:__main__:Process stderr: b''
DEBUG:__main__:Test case 1 result: {'passed': True, 'input': '4 9<br>2 7 11 15<br>', 'expected_output': '0 1', 'actual_output': '0 1', 'error': None}
DEBUG:__main__:Running test case 2
DEBUG:__main__:Running test case with input: 3 6
3 2 4
```

Fig. 6. API Debug Logs

B. Manual Test Case Management

Currently, the system uses a manual process for adding questions and test cases. This could be streamlined with a better DBMS integration for easier management.

C. Language Support

The system currently supports a limited set of languages. Adding more languages would require additional backend configuration and support.

VII. Future Work

A. Security Enhancement

One of the goals for future development is to enhance the security of the API. This can be achieved by integrating API Gateway for better control and monitoring of requests.

B. OAuth 2.0 Authentication

A more secure login mechanism, such as OAuth 2.0, will be implemented for user authentication and authorization.

C. DBMS Integration

A proper DBMS (Database Management System) will be integrated to allow easy management of coding problems and test cases. This will simplify the process of adding new questions and automating test case evaluation.

D. Live Contests and Competitions

Future improvements also include the addition of live coding contests and competitive programming features, where users can participate in real-time coding challenges.

VIII. Conclusion

The CodeIt platform provides a simple and efficient online coding judge system for users to practice and test their programming skills. By utilizing AWS services such as EC2 for backend hosting and S3 for frontend hosting, the platform ensures scalability and reliability. The system also offers a modern, user-friendly interface with minimal clutter, making it ideal for coding practice without unnecessary distractions. Future improvements will focus on enhancing security, expanding language support, and integrating a more robust database system for better management of coding challenges.

References

[1] LeetCode. Available: https://leetcode.com/

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

[3] Amazon S3. Available: https://aws.amazon.com/s3/

[4] Amazon EC2. Available: https://aws.amazon.com/ec2/

[5] Flask Documentation. [Online]. Available: https://flask.palletsprojects.com/

[6] W. Xu, J. Zhang, Y. Wang, and Y. Liu, “Design and Implementation of a Cloud-based Online Judge System,” in *Proceedings of the 2017 International Conference on Computer Science and Software Engineering*, pp. 101–105, Dec. 2017.

[7] CodeForces. Available : https://codeforces.com