Backend Engineering Launchpad – Case Study

[Backend] CodeRank

Author: Airtribe

Background & Objective

In the era of cloud computing and online development environments, there's a growing need for efficient and secure online code execution systems. These systems allow users to write, test, and execute code in various programming languages without the need to set up a local development environment. This project involves designing and implementing a backend system for an online code execution platform.

Key Features

1. **Language Support**: Implement support for multiple programming languages. This includes setting up execution environments for languages like Python, Java, JavaScript, C++, etc.

2. **Code Execution API**: Develop RESTful APIs that allow users to submit code snippets and receive the output or execution results.

3. **Security Measures**: Incorporate security measures to prevent malicious code from affecting the server or accessing sensitive data.

4. **Concurrency Handling**: Design the system to handle multiple code execution requests simultaneously without performance degradation.

5. **Timeout and Error Handling**: Implement mechanisms to handle execution timeouts and runtime errors, providing meaningful feedback to the user.

6. **Resource Management**: Implement a system to manage and allocate resources like memory and processing time for each code execution request.

Technical Requirements

1. The backend should be implemented with a focus on RESTful API design.

2. Use containerization (like Docker) for setting up isolated environments for code execution.

3. Implement a database for storing user data and code snippets.

4. Ensure robust authentication and authorization for API access.

5. Include rate limiting and other security measures to prevent abuse of the service.

Assessment Criteria

1. **Functionality:** Does the system work as intended? Does it meet all the requirements stated above?
2. **Code Quality:** Is your code clean, organized, well-commented, and following best practices?
3. **Design and Structure:** Is the system well-designed? Does it demonstrate a good understanding of system design principles and patterns?
4. **Documentation:** Is your report comprehensive and clear? Does it effectively explain the choices made and how to use the project?
5. **Presentation:** Do you effectively demonstrate and explain your system and the decisions made during its development?

Deliverables

1. The final, functional product.
2. README file outlining how to use the system, API documentation, the design decisions and other necessary information.
3. Public link of the Github repository
4. Explainer video demonstrating your project work