KALVIUM ASSIGMENT

By…
Kunwar Ranjeet

Pull Request Report

Overview

This report provides a detailed overview of the enhancements and features added to the project. The contributions include support for three additional languages (C++, Java, Python), additional test cases, and the implementation of a simple frontend.

Contributions

1. **Support for Additional Languages**
    - Implemented support for three new programming languages: **C++**, **Java**, and **Python**.
    - Each language implementation includes appropriate integration and testing to ensure functionality and compatibility.
2. **Additional Test Cases**
    - Added extensive test cases to cover edge scenarios and improve the robustness of the existing functionality.
    - Each new language has dedicated test cases to validate various functionalities.
3. **Frontend Implementation**
    - Developed a simple yet well-designed frontend to run code against any supported language.
    - The frontend includes user-friendly interfaces for selecting languages, inputting code, and viewing output.

Detailed Changes

1. Support for Additional Languages

Languages Added:

- **C++**
    - **Implementation Details**:
        - Integrated C++ compiler and runtime environment.
        - Added C++ code execution logic.
    - **Key Features**:
        - Supports standard C++ syntax.
        - Handles input/output operations.
    - **Tests**:
        - C++ Test 1: Tests basic syntax and operations.
        - C++ Test 2: Tests input/output handling.

- **Java**
  - **Implementation Details**:
    - Integrated Java compiler and runtime environment.
    - Added Java code execution logic.
  - **Key Features**:
    - Supports standard Java syntax.
    - Handles input/output operations.
  - **Tests**:
    - Java Test 1: Tests basic syntax and operations.
    - Java Test 2: Tests input/output handling.
- **Python**
  - **Implementation Details**:
    - Integrated Python interpreter.
    - Added Python code execution logic.
  - **Key Features**:
    - Supports standard Python syntax.
    - Handles input/output operations.
  - **Tests**:
    - Python Test 1: Tests basic syntax and operations.
    - Python Test 2: Tests input/output handling.

Tests:

- For each language, detailed tests have been added to ensure all functionalities are covered:
  - **C++ Tests**:
    - Test1: Printing Hello World.
    - Test2: Tests input/output handling.
  - **Java Tests**:
    - Test1: Printing Hello World.
    - Test2: Tests input/output handling.
  - **Python Tests**:
    - Test1: Printing Hello World.
    - Test2: Tests input/output handling.

2. Additional Test Cases

- **Test Case1**: Validates string manipulation functions across all supported languages.
- **Test Case2**: Tests arithmetic operations with edge cases such as large numbers and division by zero.
- **Test Case3**: Ensures proper handling of nested loops and complex control structures.
- Each test case includes expected outputs and validation steps to ensure accuracy.

3. Frontend Implementation

Frontend Features:

- **Language Selection**: Dropdown menu to select from available languages (C++, Java, Python, etc.).
- **Code Input**: Text area for users to input their code.
- **Execution**: Button to run the code and display output.
- **Output Display**: Section to show the result of the code execution.
-

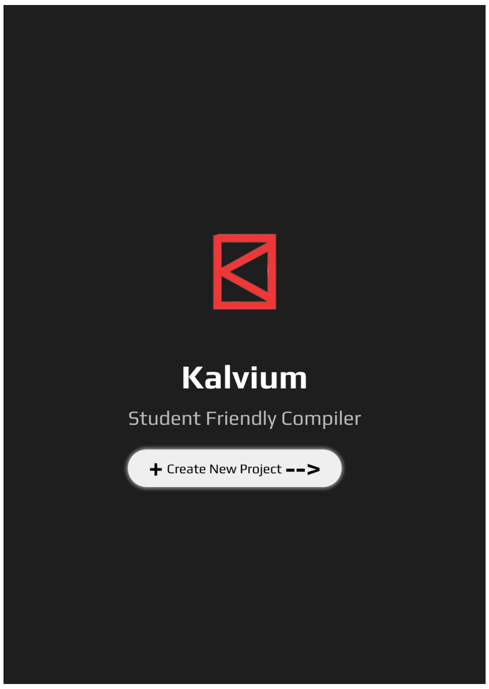
Frontend Design:

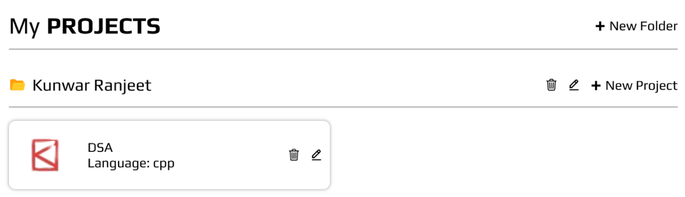
- The design follows modern UI/UX principles to provide an intuitive and efficient user experience.
- The code is organized in a `frontend` folder and adheres to the project's coding standards.

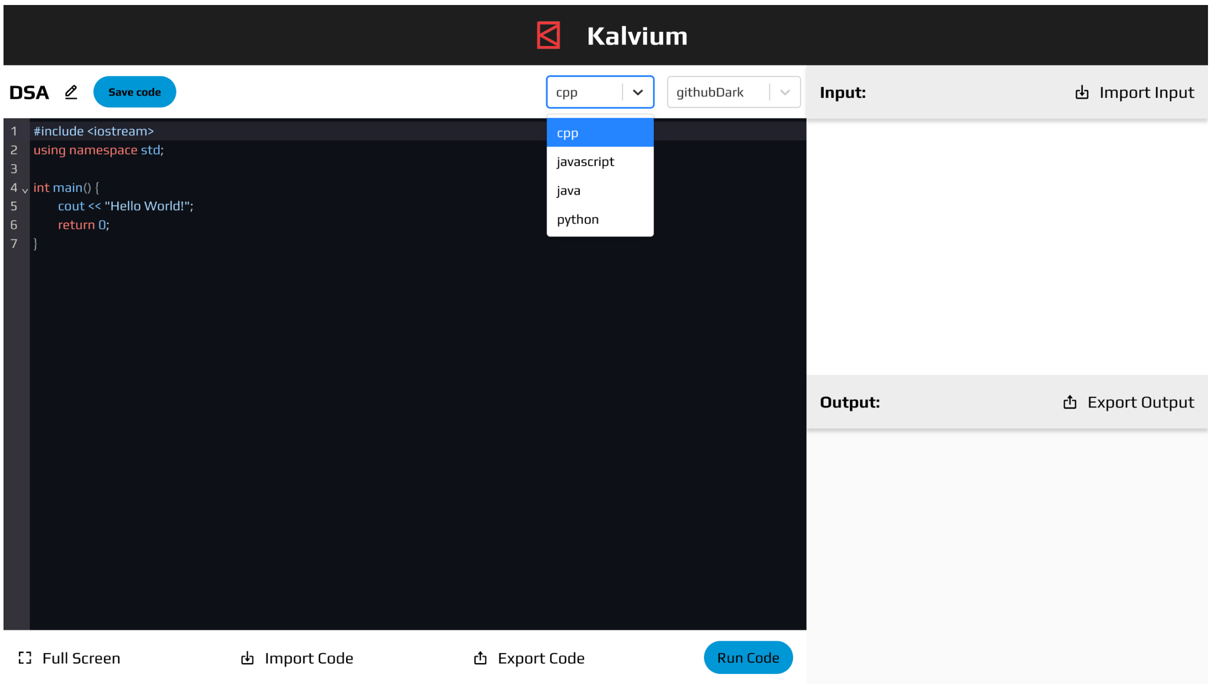
Guidelines Adherence

- **Coding Standards**: Followed the project's coding standards and style guides throughout the implementation.
- **Documentation**: Updated relevant documentation to reflect the new changes and provide usage instructions.
- **Consistency**: Ensured all changes maintain consistency with the existing codebase.
- **Testing**: Thoroughly tested all new features and languages to ensure they integrate seamlessly with the existing system.

Result

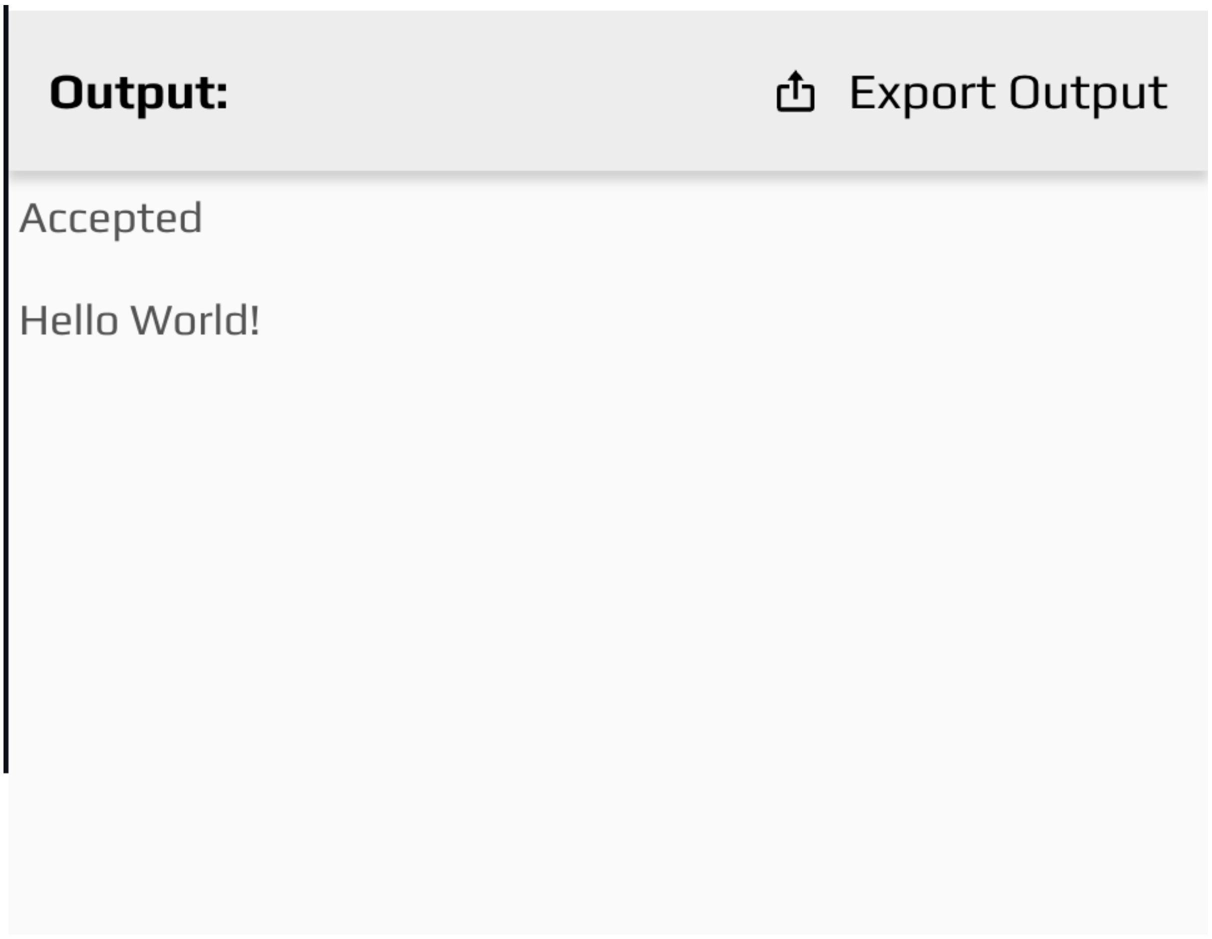






```cpp
#include <iostream>
using namespace std;

int main() {
    cout << "Hello World!";
    return 0;
}
```



```
Accepted

Hello World!
```

Conclusion

These contributions significantly enhance the project's capabilities, providing broader language support, improved testing coverage, and a user-friendly frontend interface. The detailed documentation and adherence to coding standards ensure that these additions are maintainable and easy for other developers to understand.