PROJECT AND TEAM INFORMATION

Project Title

Mini Compiler: A Web-Based Custom Language Compiler with React & Flask

Student / Team Information

Team Name	***CodeMasters***
Team member 1 (Team Lead)	*Chandra, Vinod – 22011515* *itsvinod14@gmail.com*
Team member 2	*Rana, Rahul – 220113260* *rahulrana89546@gmail.com*
Team member 3	*Chauhan, Divyansh – 22011456* *divyanshchauhan349@gmail.com*

PROPOSAL DESCRIPTION (10 pts)

Motivation (1 pt)

Most existing compilers are designed for predefined programming languages like C, C++, or Python. However, learning compiler design is more effective when students can define ***their own syntax and grammar rules****. This project aims to develop a* ***Mini Compiler*** *that allows users to* ***define and compile a custom language*** *instead of relying on fixed syntax rules. The project will help students understand how* ***compilers process tokens, parse syntax, and generate intermediate code for any language of their choice****.*

State of the Art / Current solution (1 pt)

Traditional compilers like GCC, Clang, and JVM support ***fixed programming languages****. Some online compilers allow syntax checking and execution, but they lack* ***support for defining a custom language****. Few tools provide an* ***interactive breakdown of lexical analysis, parsing, and code generation****. This Mini Compiler will bridge this gap by allowing users to* ***experiment with their own language syntax and test compilation in real-time via a web-based UI****.*

Project Goals and Milestones (2 pts)

Phase 1*: Develop the React frontend with a* ***code editor and UI styling****.*
Phase 2*: Implement Flask APIs for* ***handling custom language rules****.*
Phase 3*: Build the* ***lexer and parser*** *to support* ***user-defined syntax****.*
Phase 4*: Implement* ***custom language code generation*** *and error handling.*
Phase 5*: Integrate frontend and backend using* ***REST API****.*
Phase 6*: Allow users to* ***define grammar rules dynamically****.*

Project Approach (3 pts)

*1. **Frontend (React + CSS)***

- *A **code editor (Monaco Editor)** for writing code in the **custom language**.*
- *A section to allow users to **define grammar rules dynamically**.*
- *A **"Compile" button** to send the custom language code to the backend.*
- *A section to **display errors, tokens, or intermediate code output**.*

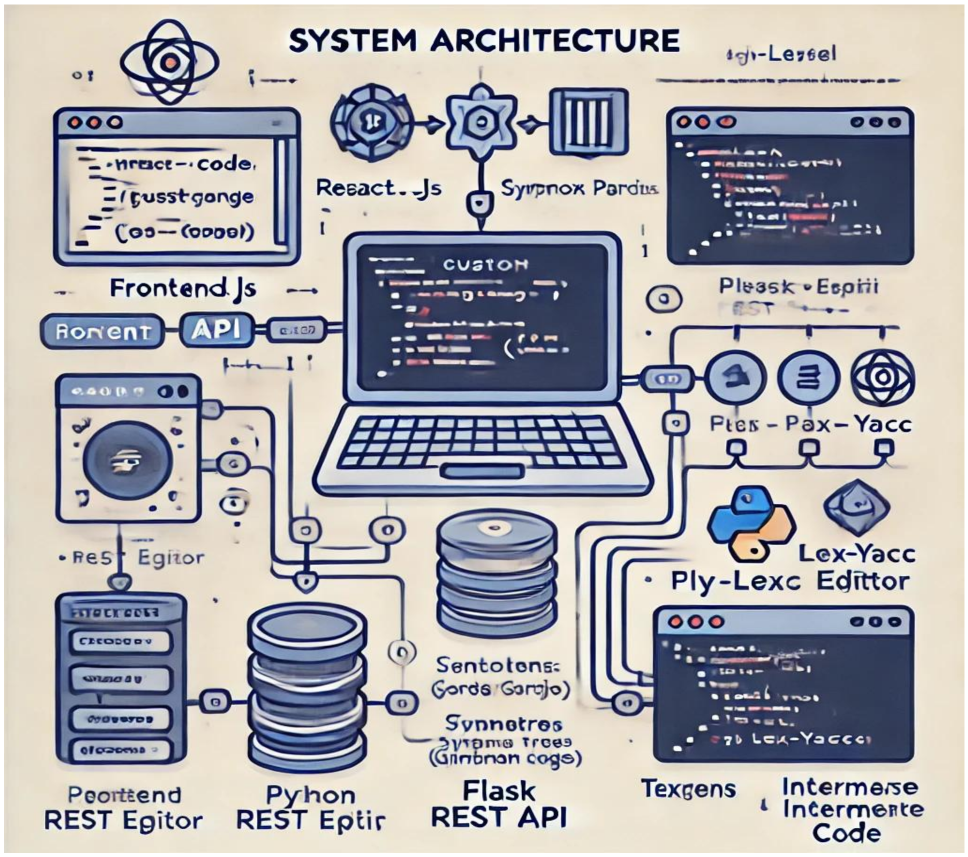
*2. **Backend (Flask + Python)***

- ***Lexical Analysis**: Tokenizes the user-defined language using **PLY (Python Lex-Yacc)**.*
- ***Syntax Parsing**: Parses tokens based on **user-defined grammar rules**.*
- ***Code Generation**: Converts parsed input into an **intermediate representation**.*
- ***API Communication**: Exposes a /compile API to handle custom language compilation requests.*

*3. **Workflow***

- *Users write code in the **custom language** and define syntax rules in the React UI.*
- *React frontend **sends the code & grammar rules** to Flask via API.*
- *Flask backend **processes the code according to the user's custom rules** and returns results.*
- *React frontend **displays the output**, including lexical analysis, parsing, and errors.*

System Architecture (High Level Diagram) (2 pts)



Project Outcome / Deliverables (1 pts)

1. *Web-based Mini Compiler that supports user-defined syntax.*
2. *Interactive lexical analysis and syntax parsing visualization.*
3. *REST API for compiling custom language code dynamically.*
4. *Intermediate code generation based on custom grammar rules.*
5. *Error handling and debugging assistance for custom languages.*

Assumptions

1. *Users will define the syntax and grammar rules for their custom language.*
2. *The compiler will support basic arithmetic, variables, and control structures for initial implementation.*
3. *The output will be in intermediate representation (not directly executable machine code).*

References

Python Lex-Yacc (PLY) - *https://www.dabeaz.com/ply/*
Flask Framework - *https://flask.palletsprojects.com/*
React.js Documentation - *https://reactjs.org/*
Monaco Editor (VS Code Editor for Web) - *https://microsoft.github.io/monaco-editor/*