# PROJECT AND TEAM INFORMATION

## Project Title

(Try to choose a catchy title. Max 20 words).

Write project title here

Student / Team Information

|  |  |
| --- | --- |
| *Team Name: Team #* | *Avengers* |
| **Team member 1 (Team Lead)**  *(Last Name, name: student ID: email, picture):* | *Parker, Peter – 123456789* [*pparker123@xyz.com*](mailto:pparker123@xyz.com) |
| **Team member 2**  *(Last Name, name: student ID: email, picture):* | *Snow, John – 987654321* [*Jsnow789@xyz.com*](mailto:Jsnow789@xyz.com) |

# PROPOSAL DESCRIPTION (10 pts)

## Motivation (1 pt)

(Describe the problem you want to solve and why it is important. Max 300 words).

Being students with a strong passion for programming and software development, we view compiler design as an integral part of computer science that connects theory to practical application. This project is a chance for us to extend learning beyond the classroom and experiment practically how a programming language is processed and run.

Creating our own in-house custom compiler helps us learn the nuances of how code is analyzed, translated, and optimized. There are popular compilers available, yet creating one from scratch makes us see the logic and form behind programming languages working. Through this project, we hope to hone critical problem-solving skills, deepen our knowledge of language processing, and get hands-on experience with system design.

One of the prime motivations for this project is to implement theoretical concepts in a useful manner. Compilers are an integral component of programming, but most of us, students only get to work with them at a superficial level. Through the process of designing and creating one ourselves, we get to feel the challenges and intricacies firsthand of transforming source code into executable programs.

Moreover, this project is a rich collaborative learning opportunity. Team work on this project facilitates our learning through collaboration, critical thinking, and problem-solving. Through collaborative idea sharing and overcoming challenges, we learn more about compiler design while sharpening our analytical and debugging skills. This experience also enables us to acquire important soft skills such as communication, coordination, and adaptability—essential in real-world software development. Aside from the technicalities, this project enhances our capacity to work well in a team, which will equip us for future professional challenges in the technology sector.

## State of the Art / Current solution (1 pt)

(Describe how the problem is solved today (if it is). Max 200 words).

The majority of contemporary compilers for languages such as C, Java, and Python rely on sophisticated frameworks such as LLVM, GCC, or Flex & Bison. All these compilers execute numerous phases such as lexing, parsing, semantic analysis, optimization, and code generation. Though they are effective, they are sophisticated and complicated for students to learn to build compilers.

Lexical analysis is done usually with Flex, and parsing with Bison, both of which obfuscate much detail, so it's hard to develop a compiler from scratch. Real-world compilers also have optimizations and advanced error reporting, which may be overwhelming for beginners.

One of the biggest challenges to learning compiler construction is that there are not many starting-point resources that break down the process step by step. Most approaches overemphasize theory or rely on tools, decreasing hands-on time. Building a simple compiler from scratch instructs students on how programming languages are processed and decoded to machine code.

Existing solutions are either high-end real-world compilers or obfuscate critical processes through automation. The availability of a simple compiler helps students work directly with lexing, parsing, and code generation, providing them with a good idea of language processing and system design.

## 

## Project Goals and Milestones (2 pts)

(Describe the project general goals. Include initial milestones as well any other milestones. Max 300 words).

*This project aims to create and execute a compiler tailored for a custom programming language using C++. The compiler will include features like lexical analysis, parsing, semantic analysis, generating machine code, and executing it, all while incorporating error handling and optimization.*

***Week 1: Research & Planning***

* *Study compiler design concepts (lexical analysis & parsing techniques)*
* *Analyze existing compilers (LLVM, GCC) for best practices.*
* *Finalize the tools (Flex/Bison, C++ libraries), planning out the architecture and assign the team responsibilities.*

***Week 2: Lexical Analysis (Lexer) Implementation***

* *Develop a lexer to tokenize input code, handling keywords, identifiers, literals, operators & punctuation.*
* *Implement error detection and validate tokenization with sample inputs.*

***Week 3: Syntax Analysis (Parser) Development***

* *Implement a parser using chosen parsing technique (LL (1) or LR).*
* *Define grammar rules for statements, expressions, loops & functions.*
* *Generate Abstract Syntax Tree (AST).*
* *Validate correct and incorrect syntax cases.*

***Week 4: Semantic Analysis & Symbol Table***

* *Develop semantic rules (like: type checking, function resolution).*
* *Implement a symbol table to store variable/function details.*
* *Handle scope resolution, undeclared variables, and type mismatches.*

***Week 5: Intermediate Representation (IR) & Code Generation***

* *Convert AST into an intermediate representation.*
* *Implement basic optimizations like constant folding and dead code elimination.*
* *Generate simple machine-level code or bytecode.*

***Week 6: Execution Engine & Error Handling***

* *Implement an execution engine to interpret generated code.*
* *Develop error handling mechanisms for syntax, semantic, and runtime errors.*

***Week 7: Advanced Features & Testing***

* *Enhancing functions, loops, conditionals, and recursion.*
* *Optimize performance and memory usage.*

***Week 8: Documentation & Final Review***

* *Detailed documentation.*
* *Conduct final testing & debugging.*

## Project Approach (3 pts)

(Describe how you plan to articulate and design a solution. Including platforms and technologies that you will use. Max 300 words).

Write your answer here

## System Architecture (High Level Diagram)(2 pts)

(Provide an overview of the system, identifying its main components and interfaces in the form of a diagram using a tool of your choice).

Write your answer here

## Project Outcome / Deliverables (1 pts)

(Describe what are the outcomes / deliverables of the project. Max 200 words).

*This project aims to create a* ***custom programming language*** *with a fully functional* ***compiler*** *capable of translating source code into executable instructions.*

***Expected Deliverables:***

* ***Lexical and Syntax Analysis****: A lexer and parser to tokenize the input and validate syntax based on predefined grammar rules.*
* ***Abstract Syntax Tree (AST) & Symbol Table****: Structures to represent program hierarchy and manage variables, functions, and scopes efficiently.*
* ***Intermediate Representation (IR) & Code Generation****: Conversion of parsed code into an intermediate form, followed by translation into machine code or bytecode.*
* ***Error Handling & Optimization****: Mechanisms for detecting syntax and semantic errors, along with performance optimizations for efficient execution.*
* ***Comprehensive Documentation****: A guide detailing language syntax, compiler structure, setup instructions, and usage.*
* ***Test Cases & Sample Programs****: Demonstrations showcasing language capabilities, along with rigorous testing to ensure correctness and reliability.*

*The final outcome will be a* ***working compiler*** *that processes programs written in the designed language, producing executable output while adhering to best practices in compiler design.*

## Assumptions

( Describe the assumptions ( if any ) you are making to solve the problem. Max 100 words )

Your answer here

## References

(Provide a list of resources or references you utilised for the completion of this deliverable. You may provide links).

Your answer here