# PROJECT AND TEAM INFORMATION

## *Student / Team Information*

| | |
|---|---|
| *Team Name:* | *Four-Loop* |
| **Team member 1 (Team Lead)**<br>*(Last Name, name: student ID: email, picture):*<br>*Vaibhav Bhandari- 220211995*<br>vaibhav.bhandari1103@gmail.com |  |
| **Team member 2**<br>*(Last Name, name: student ID: email, picture):*<br>*Piyush Chamoli- 220111074*<br>*piyushchamoli6@gmail.com* |  |
| **Team member 3**<br>*(Last Name, name: student ID: email, picture):*<br>*Mohd Aman*– 22011794<br>*getwithaman22@gmail.com* |  |
| **Team member 4**<br>*(Last Name, name: student ID: email, picture):*<br>*Ashutosh Dobriyal- 220112385*<br>*ashutoshdobriyal007@gmail.com* |  |

# PROPOSAL DESCRIPTION (10 pts)

## Motivation (1 pt)

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

*Security vulnerabilities in code, such as SQL injection, buffer overflows, and improper memory management, lead to severe cybersecurity threats. Many developers unknowingly introduce security flaws that can be exploited by attackers. Existing solutions, such as static code analyzers, are often complex, expensive, or require expert knowledge. To address this, our project proposes SafeCompile, an AI-powered compiler that detects security vulnerabilities in source code during compilation and provides real-time suggestions for secure alternatives. This will enhance software security, prevent cyberattacks, and promote secure coding practices, particularly for developers without extensive security expertise.*

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

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

*Currently, secure coding practices are enforced using static analysis tools such as SonarQube, Fortify, and Coverity. However, these tools require manual configuration, integration with existing workflows, and are mostly used post-development. Traditional compilers like GCC and Clang primarily focus on syntax and performance optimizations but do not actively detect security flaws during compilation. This gap leads to vulnerabilities being caught late in the software development lifecycle, increasing the cost and effort needed to fix them.*

## Project Goals and Milestones (2 pts)

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

*- Phase 1: Research and collect security vulnerability datasets (e.g., OWASP Top 10).*
*- Phase 2: Develop a lexical and syntax analyzer integrated with security pattern recognition.*
*- Phase 3: Implement a machine learning model for detecting common vulnerabilities.*
*- Phase 4: Integrate secure code suggestions and automatic fixes within the compiler.*
*- Phase 5: Test the compiler with real-world codebases and refine detection accuracy.*
*- Final Phase: Deploy a working prototype and evaluate effectiveness compared to existing security tools.*

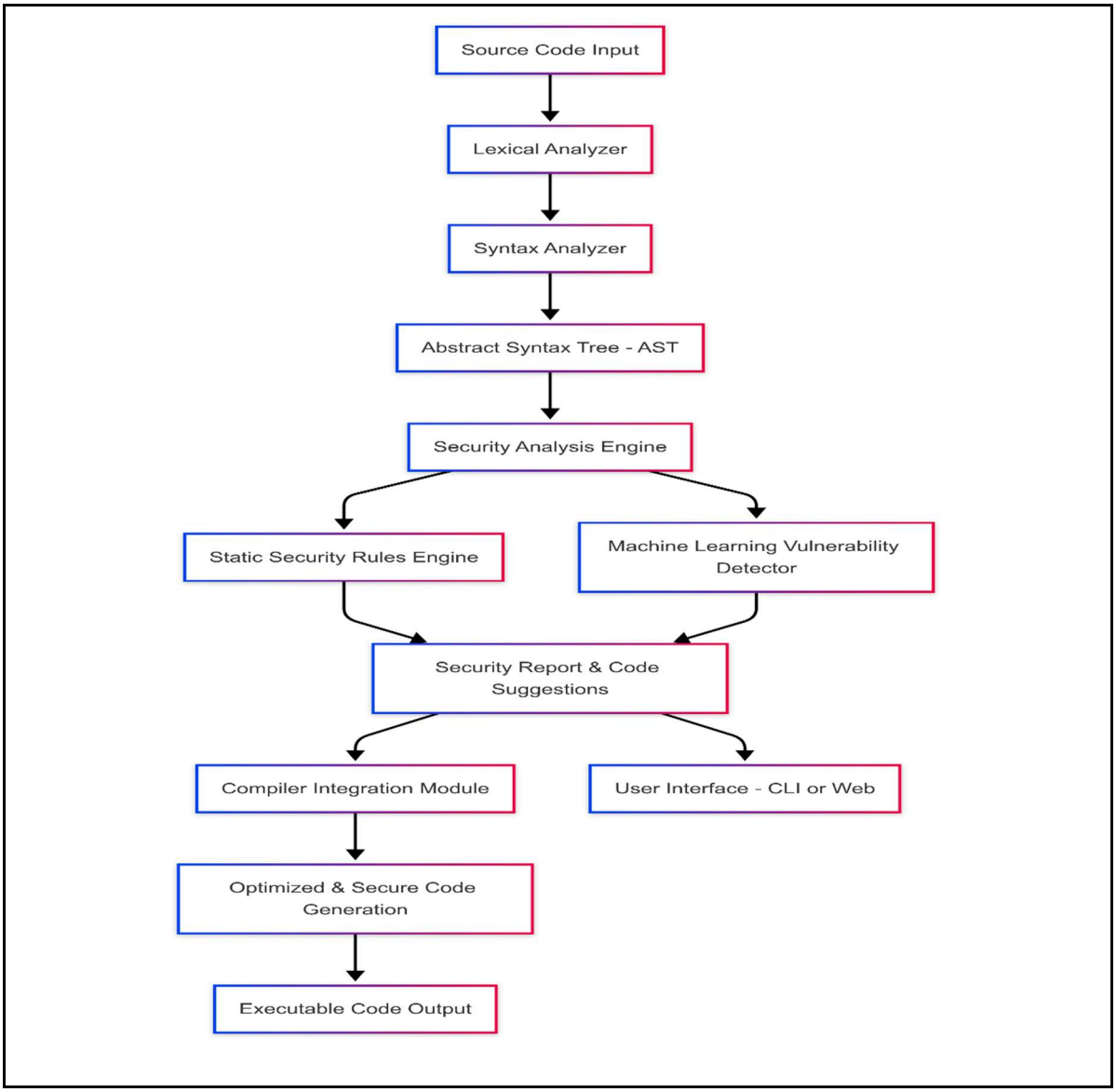
## Project Approach (3 pts)

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

*Our project will be built using Python and LLVM to extend compiler functionalities. The key components include:*
*- Lexical & Syntax Analyzer: Identifies security vulnerabilities in code syntax.*
*- Machine Learning Model: Trained on secure and insecure code samples to detect patterns.*
*- Security Rule Engine: Contains predefined security guidelines based on OWASP, CWE, and NIST standards.*
*- Compiler Integration: Implements real-time security checks within the compilation process.*
*- User Interface: Provides real-time suggestions and explanations for detected vulnerabilities.*
*- Technologies Used: Python, LLVM, TensorFlow (for ML model), and a custom rule-based security engine.*

## 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).*



## Project Outcome / Deliverables (1 pts)

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

*- A functional Secure Code Compiler capable of detecting security flaws during compilation.*
*- A web-based demo showcasing real-time secure code recommendations.*

## Assumptions

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

*Developers will adopt secure coding practices if security flaws are detected early.*
*- AI models can accurately classify secure vs. insecure code patterns.*

## References

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

*- OWASP Top 10 Security Risks (https://owasp.org/www-project-top-ten/)*
*- LLVM Compiler Infrastructure (https://llvm.org/)*
*- CWE (Common Weakness Enumeration) Database (https://cwe.mitre.org/)*
*- NIST Secure Coding Guidelines (https://csrc.nist.gov/publications/)*