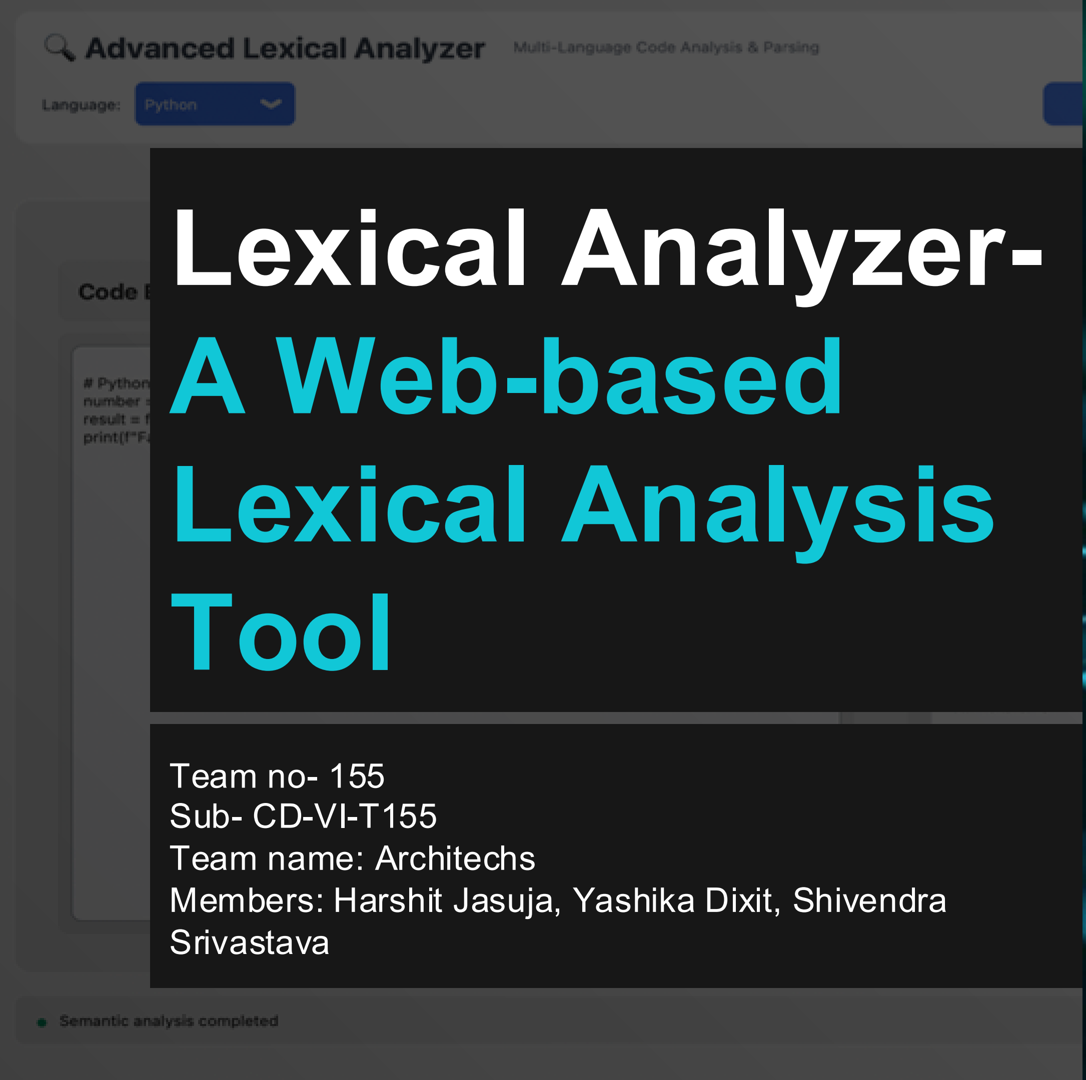


Advanced Lexical Analyzer

Multi-Language Code Analysis & Parsing

Language: Python

Code E

```
# Python
number =
result = f
print(f"F
```

Semantic analysis completed

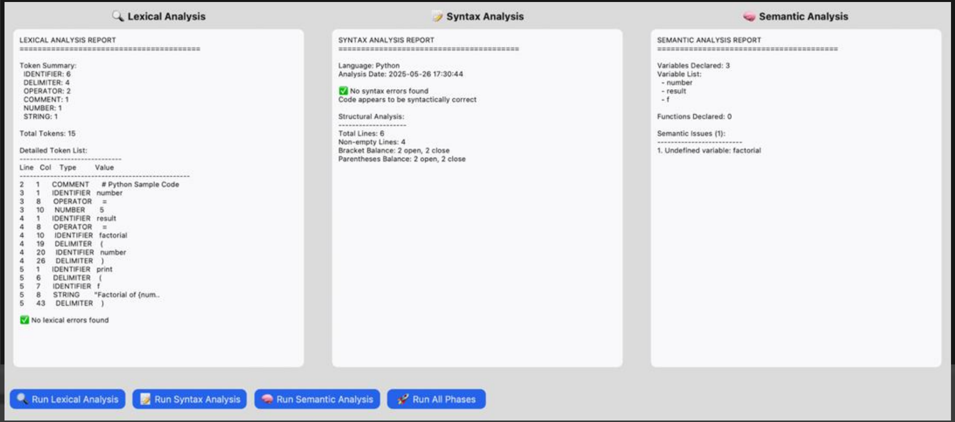
Lexical Analyzer- A Web-based Lexical Analysis Tool

Team no- 155
Sub- CD-VI-T155
Team name: Architechs
Members: Harshit Jasuja, Yashika Dixit, Shivendra Srivastava

Objective

To design and implement a robust, user-friendly lexical analyzer and code analysis tool that supports multiple programming languages, offers advanced visualization, and integrates optional AI/ML features, thereby enhancing both educational and practical understanding of compiler and code analysis concepts.Our goal is to build an **interactive, web-based lexical analyzer** that takes source code as input and performs lexical analysis in real-time.

The project aims to bridge the gap between traditional compiler theory and modern software tooling through **interactive**, real-time analysis and visualization.



🔍 Lexical Analysis

```
LEXICAL ANALYSIS REPORT
========================================

Token Summary:
 IDENTIFIER: 6
 DELIMITER: 4
 OPERATOR: 2
 COMMENT: 1
 NUMBER: 1
 STRING: 1

Total Tokens: 15

Detailed Token List:
------------------------------
Line  Col   Type        Value
--------------------------------------------------
2     1     COMMENT     # Python Sample Code
3     1     IDENTIFIER  number
3     8     OPERATOR    =
3     10    NUMBER      5
4     1     IDENTIFIER  result
4     8     OPERATOR    =
4     10    IDENTIFIER  factorial
4     19    DELIMITER   (
4     20    IDENTIFIER  number
4     26    DELIMITER   )
5     1     IDENTIFIER  print
5     6     DELIMITER   (
5     7     IDENTIFIER  f
5     8     STRING      "Factorial of {num..
5     43    DELIMITER   )

✅ No lexical errors found
```

📝 Syntax Analysis

```
SYNTAX ANALYSIS REPORT
========================================

Language: Python
Analysis Date: 2025-05-26 17:30:44

✅ No syntax errors found
Code appears to be syntactically correct

Structural Analysis:
--------------------
Total Lines: 6
Non-empty Lines: 4
Bracket Balance: 2 open, 2 close
Parentheses Balance: 2 open, 2 close
```

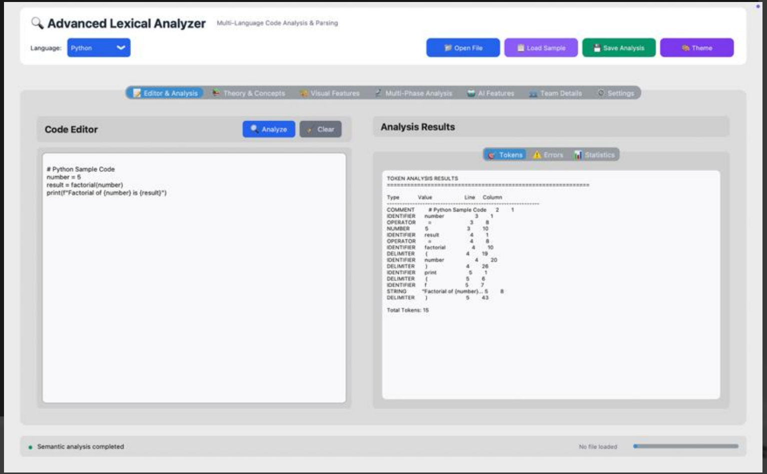
🧠 Semantic Analysis

```
SEMANTIC ANALYSIS REPORT
========================================

Variables Declared: 3
Variable List:
  - number
  - result
  - f

Functions Declared: 0

Semantic Issues (1):
-------------------------
1. Undefined variable: factorial
```



🔍 Advanced Lexical Analyzer Multi-Language Code Analysis & Parsing

Language: Python

```
# Python Sample Code
number = 5
result = factorial(number)
print(f"Factorial of {number} is {result}")
```

```
TOKEN ANALYSIS RESULTS
============================================================

Type        Value               Line    Column
------------------------------------------------------------
COMMENT     # Python Sample Code    2       1
IDENTIFIER  number              3       1
OPERATOR    =                   3       8
NUMBER      5                   3       10
IDENTIFIER  result              4       1
OPERATOR    =                   4       8
IDENTIFIER  factorial           4       10
DELIMITER   (                   4       19
IDENTIFIER  number              4       20
DELIMITER   )                   4       26
IDENTIFIER  print               5       1
DELIMITER   (                   5       6
IDENTIFIER  f                   5       7
STRING      "Factorial of {number}... 5       8
DELIMITER   )                   5       43

Total Tokens: 15
```

● Semantic analysis completed

No file loaded

Problem Statement & Proposed Solution

Problem Statement:

Traditional lexical analyzers and code analysis tools are often limited to a single language, lack modern interfaces, and provide minimal visualization or AI-powered insights. This restricts their effectiveness for learning, comparison, and advanced code exploration, especially in educational settings.

Proposed Solution:

Develop a cross-platform, modular application that:

- Supports lexical, syntax, and semantic analysis for Python, JavaScript, Java, and C++.
- Offers interactive visualizations (AST, parse trees, token frequency).
- Integrates optional AI/ML models for code completion and error prediction.
- Provides a modern, customizable GUI with real-time feedback and accessibility features.

Updated Workflow / Architecture

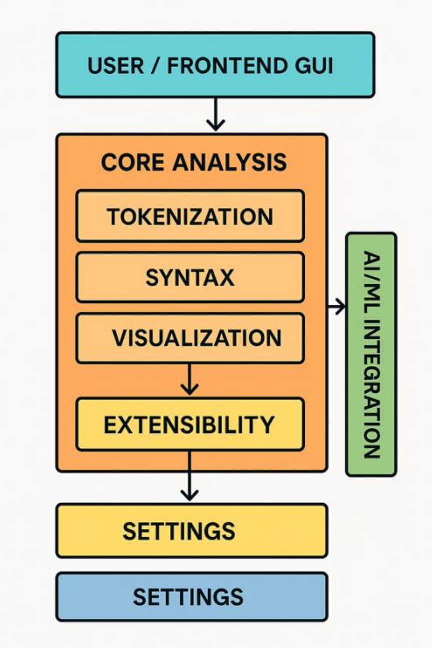
- Frontend: Modern GUI with tabbed navigation, real-time analysis, and customization.

- Backend: Language-aware tokenization, analysis modules, and error reporting.

- Visualization: AST, parse trees, and token frequency charts using Matplotlib & NetworkX.

- AI/ML Integration: Optional transformer-based models for code intelligence.

- Extensibility: Modular design for adding languages or features.



Key Features

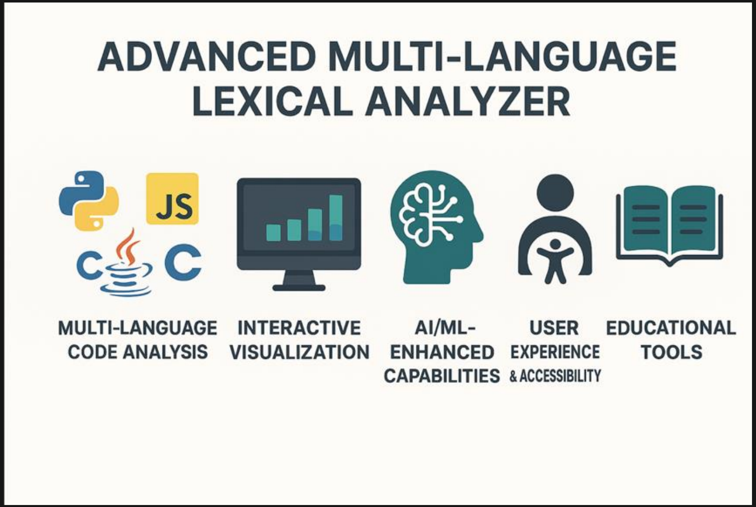
1. **Multi-Language Code Analysis:**
- Lexical, syntax, and semantic analysis for Python, JavaScript, Java, and C++.
- Detailed token tables and error reporting.

2. **Interactive Visualization:**
- Abstract Syntax Tree (AST) diagrams.
- Parse tree generation.
- Token frequency and complexity charts.

3. **AI/ML-Enhanced Capabilities:**
- Optional code completion suggestions.
- Automated error prediction using transformer models.

4. **Educational Tools:**
- LALR(1) parsing table generator
- Comprehensive documentation and tooltips

5. **User Experience & Accessibility:**
- Modern, customizable GUI (themes, fonts, high-contrast modes).
- Real-time analysis and sample code loading.
- Keyboard shortcuts and in-app guidance.


ADVANCED MULTI-LANGUAGE LEXICAL ANALYZER
JS
C
MULTI-LANGUAGE CODE ANALYSIS
INTERACTIVE VISUALIZATION
AI/ML-ENHANCED CAPABILITIES
USER EXPERIENCE & ACCESSIBILITY
EDUCATIONAL TOOLS

Technologies Used

- Frontend: Python, CustomTkinter (GUI)-

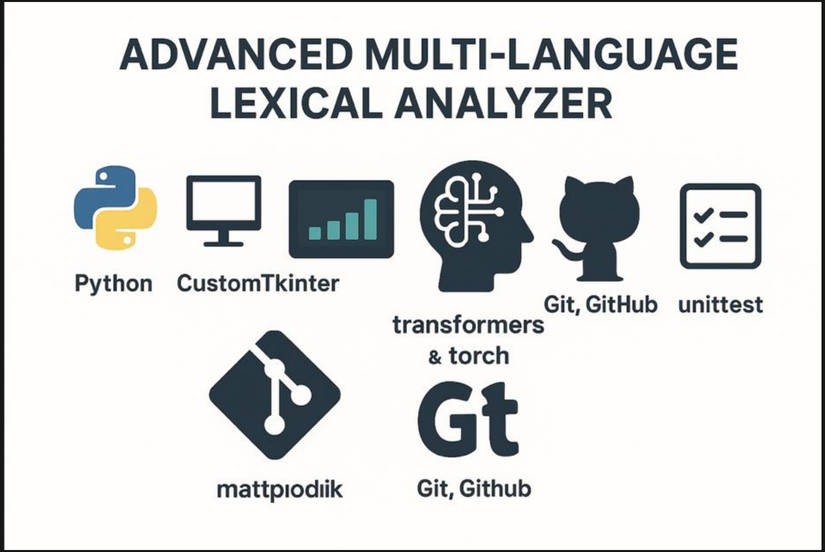
- Visualization: Matplotlib, NetworkX

- Backend: Python Standard Library (ast, re, etc.)

- AI/ML (optional): transformers, torch (CodeGPT, CodeBERTa)

- Testing: unittest, manual validation

- Version Control: Git, GitHub

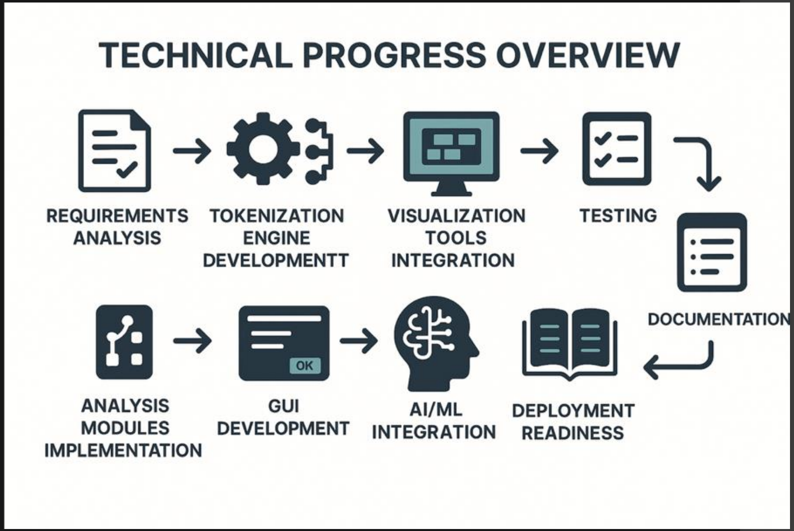


Technical Progress Overview

The project advanced smoothly through all planned development phases. Each core module—**tokenization, analysis, visualization, and AI/ML integration**—was implemented, tested, and refined for reliability and usability. The GUI was made fully responsive and customizable, while thorough validation ensured cross-platform compatibility. Comprehensive documentation and user support were also completed, resulting in a robust, user-friendly, and extensible tool ready for deployment.

Key highlights:

- All core modules (tokenization, analysis, visualization) implemented and validated
- Responsive, customizable GUI with real-time analysis
- AI/ML integration operational and robust
- Comprehensive testing and documentation completed
- Ready for deployment and further extension


TECHNICAL PROGRESS OVERVIEW
REQUIREMENTS ANALYSIS
TOKENIZATION ENGINE DEVELOPMENTT
VISUALIZATION TOOLS INTEGRATION
TESTING
DOCUMENTATION
ANALYSIS MODULES IMPLEMENTATION
GUI DEVELOPMENT
OK
AI/ML INTEGRATION
DEPLOYMENT READINESS

Role-wise Contributions

Team Member	Contribution Area
Harshit Jasuja	Project Lead, Architecture, Tokenization, AI/ML Integration
Yashika Dixit	GUI Design, User Support, Documentation
Shivendra Srivastava	Testing, Visualization Modules, Automation

Challenges Faced & Key Learnings

Challenges Faced:

- Ensuring cross-platform GUI consistency
- Handling multi-language tokenization complexity
- Optimizing visualization for large data sets
- Managing optional AI/ML dependencies
- Balancing educational usability with technical accuracy

Key Learnings:

- Importance of modular, extensible design
- Effective collaboration and iterative testing
- Integrating modern AI/ML with traditional software tools
- User-centered design enhances adoption and impact-

Learning Outcomes

- Deepened understanding of compiler construction and code analysis
- Practical experience with GUI development and visualization
- Exposure to AI/ML integration in software tools
- Enhanced teamwork and project management skills

Future Scope

- Support for additional programming languages
- Deeper semantic and runtime analysis
- Cloud-based collaboration and analysis features
- Community-driven plugins and extensions