## SCRIPT ANALYSIS

**A PROJECT REPORT**

*Submitted in partial fulfillment of the requirements*

*for the award of the degree of*

*Bachelor of Science/Commerce/Computer Application (Programme)*

**SUBMITTED BY**

**ADHIVEL V**

**2213141033052**

*Under the guidance of*

**Dr. G. Kiruthiga**

**DESIGNATION OF THE GUIDE**

## BACHELOR OF COMPUTER APPLICATIONS

****

**GURU NANAK COLLEGE**

**(AUTONOMOUS)**

**Affiliated to University of Madras**

**Accredited at ‘A++’ Grade by NAAC | An ISO 9001 2015 Certified Institution**

Guru Nanak Salai, Velachery, Chennai – 600 042.

## MARCH - 2025

**GURU NANAK COLLEGE**

**(AUTONOMOUS)**

**Affiliated to University of Madras**

**Accredited at ‘A++’ Grade by NAAC | An ISO 9001 2015 Certified Institution**

Guru Nanak Salai, Velachery, Chennai – 600 042.

**NAME OF THE PROGRAMME**

**BONAFIDE CERTIFICATE**

This is to certify that, this is a bonafide record of work done by **NAME OF THE STUDENT, REGISTER NUMBER** offor theFinal Year Project during the Academic Year 2023-24.

**PROJECT GUIDE HEAD OF THE DEPARTMENT**

**Submitted for the Project Viva Voce Examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ at GURU NANAK COLLEGE (Autonomous), Guru Nanak Salai, Velachery, Chennai - 600 042.**

**Internal Examiner External Examiner**

**Date: Date:**

**DECLARATION**

I (**STUDENT NAME) (Register** No.: ) studying III Year (Name of the Programme) at Guru Nanak College (Autonomous), Chennai hereby declare that this the Report of my Project entitled, (**TITLE OF THE PROJECT)** is the record of the original work carried out by me under the Guidance and Supervision of (NAME OF THE GUIDE) towards the partial fulfillment of the requirements of the award of the Degree of (NAME OF THE PROGRAMME)**.** I further declare that this has not been submitted anywhere for the award of Degree/Diploma or any other similar to this before.

**PLACE: CHENNAI ADHIVEL V   
DATE: 2213141033052**

**ACKNOWLEDGEMENT**

I would like to thank the **Principal Dr. T. K. Avvai Kothai and Vice Principal Dr. Anitha Malisetty** for providing the necessary resources and facilities for the completion of this project.

I extend my deepest thanks to**. R .CAROLINE KALAISELVI** whose guidance, support, and encouragement were invaluable throughout this endeavor. Her expertise and insights have been instrumental in shaping this project and enhancing its quality.

I owe my Guide (NAME OF THE PROJECT GUIDE) a debt of gratitude for his/her invaluable guidance, patience, and encouragement. His/Her mentorship has been a beacon of light, steering me through the complexities of this project and helping me realize my potential.

I also like to extend my thanks to the Faculty Members **(NAME OF ALL THE FACULTY MEMBERS),** for their valuable suggestion during the course of the study of my project.

Last but not least, I thank my **family and friends** for their unwavering encouragement and understanding during this journey.

**ABSTRACT**

This project introduces a comprehensive tool for analysing and evaluating programming code, focusing on metrics that provide insights into its structure, complexity, and quality. The tool categorizes its analysis into three key areas: **Overall Code Structure**, **Code Complexity**, and **Code Quality Metrics**, offering developers an intuitive way to assess and enhance their code.

1. **Overall Code Structure**: Assesses total lines, code lines, comment lines, and blank lines to provide a clear view of organization and readability.
2. **Code Complexity**: Examines cyclomatic complexity, maximum nesting depth, and counts of functions and classes, offering insights into logical flow and architectural depth.
3. **Code Quality Metrics**: Evaluates average line length, long lines, and code duplication scores, essential for ensuring a clean and efficient codebase.

This tool is designed to support developers, code reviewers, and project managers by highlighting inefficiencies, redundancies, and structural patterns. It enables users to make data-driven decisions to optimize code quality and maintainability. With its user-friendly interface and detailed reporting, this tool is ideal for projects of varying sizes and complexities, ensuring adherence to coding standards and promoting best practices in software development.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S/NO.** | **TITLE** | **PAGE NO.** |
| **1 INTRODUCTION 1** | | |
| **1.1** | Objectives | **2** |
| **1.2** | Project Description | **3** |
| **1.3** | key Features & Functionality | **4** |
| **1.4** | Modules in this project | **5** |
| **2 SYSTEM CONFIGURATION 6** | | |
| **2.1** | Hardware Specifications | **7** |
| **2.2** | Software Specifications | **8** |
| **2.3** | Network Specifications | **8** |
| **3 INSTALLATION GUIDE 9** | | |
| **3.1** | Prerequisites | **10** |
| **3.2** | Installation Steps | **10 - 11** |
| **3.3** | Running the Application | **12** |
| **4 SOFTWARE** **DESCRIPTION** **13** | | |
| **4.1** | Frontend Description | **14** |
| **4.2** | Backend Description | **15** |
| **5 FEASIBILITY STUDY 16 - 19** | | |
| **6 SYSTEM DESIGN 20** | | |
| **6.1** | Class Diagram | **21** |
| **6.2** | Data Flow Diagram | **22** |
| **6.3** | Sequence Diagram | **23** |
| **7 APPENDICES 24** | | |
| **7.1** | Folder Structure | **25** |
| **7.2** | Coding | **26 - 76** |
| **7.3** | Screenshots | **77 - 80** |
| **8 BOTTOMLINE 81** | | |
| **8.1** | Conclusion | **82** |
| **8.2** | Future Enhancement | **82 - 83** |
| **9 TESTING 84** | | |
| **9.1** | Issues identified & Fixes Implemented | **85** |
| **9.2** | Testing Approach | **86** |
| **9.3** | How Complexity Analyse is Calculated | **87 - 90** |
| **10 REFERENCES 91** | | |
| **10.1** | Journal References | **92** |
| **10.2** | Book References | **92** |
| **10.3** | Web References | **92** |

# INTRODUCTION

## INTRODUCTION

* 1. **OBJECTIVES:**

The primary objective of this web application is to provide a tool for analying script files, offering insights into code structure, complexity, and basic metrics. This application aims to:

* Help developers gain a better understanding of their code base.
* Identify key metrics such as the number of lines, classes, and functions.
* Measure cyclomatic complexity and maintainability of code.
* Offer additional features such as syntax highlighting, multi-language support, and real-time analysis.

## PROJECT DESCRIPTION:

The **Script Analysis Web App** is a powerful web-based tool designed to help developers analyze and evaluate their code by providing key insights into its structure, complexity, and maintainability. This tool supports multiple programming languages, including Python, Java, C++, JavaScript, Ruby, Go, Swift, PHP, and C#.

The application offers real-time analysis, allowing users to upload their script files and instantly receive detailed metrics on line count, function definitions, class structures, and cyclomatic complexity. With features like syntax highlighting, multi-language support, and downloadable reports, this tool helps developers optimize their code and improve maintainability.

Built using **HTML, CSS, and JavaScript** for the frontend and **Python (Flask)** for the backend, the app does not require a database. Instead, it processes script files dynamically, generating structured insights that can be viewed and saved by the user.

This tool is especially useful for software developers, educators, and students who want to analyze and refine their code with minimal effort.

**1.3. KEY FEATURES & FUNCTIONALITY**

* Real-Time Code Analysis: Instantly analyzes script files and provides structured insights without requiring a page refresh.
* Code Metrics: Displays essential statistics such as total lines of code, number of functions, classes, and methods.
* Complexity Analysis: Evaluates cyclomatic complexity and maintainability index, helping developers optimize their code.
* Syntax Highlighting: Enhances readability by adding color-coded syntax for different programming languages.
* Multi-Language Support: Supports various programming languages for flexible analysis.
* Error Detection: Identifies potential syntax and logical errors within the uploaded script.
* Downloadable Reports: Allows users to save the analysis results in formats like PDF or CSV for future reference.

## MODULES IN THIS PROJECT:

The **Script Analysis Web App** is structured into multiple modules, each responsible for handling a specific functionality of the application. These modules work together to analyze scripts, generate insights, and present results in an interactive interface.

#### 1. Frontend Module (HTML, CSS, JavaScript)

* Provides an interactive and user-friendly interface.
* Allows users to upload script files for analysis.
* Displays real-time analysis results, including code metrics and complexity.
* Implements syntax highlighting for better readability.
* Supports downloadable reports for user convenience.

#### 2. Backend Module (Python, Flask)

* Handles user requests and processes uploaded script files.
* Performs code analysis, extracting metrics such as line count, functions
* Computes complexity analysis, including cyclomatic complexity and maintainability index.
* Sends processed data to the frontend for visualization.

#### 3. Code Analysis Engine

* Parses and analyzes script files dynamically.
* Identifies functions, classes, methods, and their relationships.
* Evaluates code quality using industry-standard metrics.
* Supports multiple programming languages for analysis.

#### 4. Report Generation Module

* Generates downloadable reports summarizing analysis results.
* Formats reports into PDF or CSV for easy sharing and record-keeping.
* Highlights key findings, including complexity levels and maintainability scores.

# SYSTEM CONFIGURATION

## SYSTEM CONFIGURATION

* 1. **HARDWARE CONFIGURATION:**

The following minimum hardware specifications are required to run the **Script Analysis Web App** efficiently:

* **Processor:** Intel i3 or higher / AMD equivalent
* **RAM:** 4GB minimum (8GB recommended for better performance)
* **Storage:** At least 500MB of free disk space
* **Display:** Minimum resolution of 1280x720 (recommended: Full HD 1920x1080)
* **Internet Connection:** Required for accessing the web application when hosted online

## SOFTWARE CONFIGURATION:

The following software components are required for developing, running, and testing the application:

* **Operating System:** Windows, Linux, or macOS
* **Programming Language:** Python
* **Web Technologies:** HTML, CSS, JavaScript
* **Backend Framework:** Flask (Python)
* **Browser:** Google Chrome, Mozilla Firefox, Microsoft Edge (latest versions recommended)
* **Code Editor:** VS Code, PyCharm, or any text editor of choice
* **Python Libraries & Dependencies:**

1. Flask (for backend API handling)
2. Pylint (for code analysis and error detection)
3. Jinja2 (for rendering templates)

**3.3. Network Configuration :**

The **Script Analysis Web App** is a lightweight application designed to function in both **offline and online environments**. Below are the network configurations required for deployment and usage:

**Local Deployment (Offline Mode)**

* The application runs on **localhost** using Flask’s built-in development server.
* Default access is via http://127.0.0.1:5000/ or http://localhost:5000/.
* No internet connection is required unless additional external libraries or updates are needed.
* Internet required only for render template [code editor textbox]

# INSTALLATION GUIDE

## INSTALLATION GUIDE

* 1. **Prerequisites :**

Before installing the Script Analysis Web App, ensure your system meets the following requirements:

### Hardware Requirements:

* **Processor:** Intel i3 or higher
* **RAM:** 4GB (8GB recommended)
* **Storage:** At least 500MB of free space

### Software Requirements:

* **Operating System:** Windows, Linux, or macOS
* **Python:** Version 3.x installed
* **Web Browser:** Chrome, Firefox, or Edge

### Dependencies:

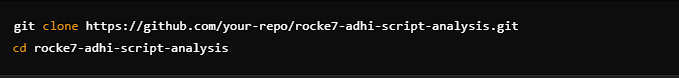
The application requires the following Python packages:

* Flask (for backend)
* Pylint (for code analysis)
* Jinja2 (for HTML templating)
  1. **Installation Steps :**

Follow these steps to set up the Script Analysis Web App on your system.

### Step 1: Clone or Download the Project

Download the project from the repository or clone it using **Git**:



### Step 2: Install Python and Virtual Environment (Optional)

### Ensure Python 3.x is installed:

For a **virtual environment** setup (recommended to avoid dependency conflicts):



**Step 3:** Install Dependencies :



* 1. **Running the Application :**

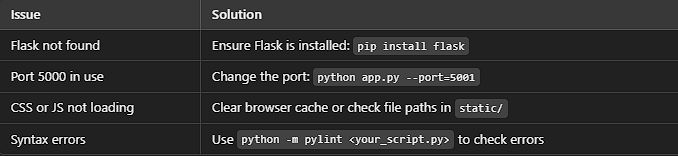
**Step 4:** Start the Flask Server (Run the **Flask application** using: )



**Step 5:** Access the Web App

* + Open your **web browser** and go to : 

* + You should see the **Get Started** page, where you can upload and analyze your scripts.
  1. **Troubleshooting & Common Issues :**

****

# SOFTWARE

# DESCRIPTION

## SOFTWARE DESCRIPTION

The **Script Analysis Web App** is developed using a combination of **frontend** and **backend** technologies to provide an interactive and efficient tool for analyzing code scripts.

### Frontend Description (HTML, CSS, JavaScript)

The frontend of the application is responsible for providing an intuitive and interactive user interface. It allows users to upload script files, view analysis results, and interact with various features.

### Technologies Used:

* **HTML (HyperText Markup Language):**
  + Provides the structure of the web pages.
  + Defines elements such as file upload, buttons, and result display sections.
* **CSS (Cascading Style Sheets):**
  + Enhances the visual design of the interface.
  + Ensures responsive layout and consistent styling across devices.
* **JavaScript (JS):**
  + Enables dynamic functionalities, such as real-time updates and interactive elements.
  + Handles event-driven actions like file uploads and API requests to the backend.

### Frontend Features:

* **User-Friendly Interface:** Designed with a clean and responsive layout.
* **File Upload & Processing:** Allows users to upload script files for analysis.
* **Real-Time Updates:** Displays analysis results dynamically without refreshing the page.
* **Syntax Highlighting:** Improves code readability using color-coded syntax.

## Backend Description (Python & Flask)

The backend of the application is responsible for handling script processing, performing code analysis, and managing communication between the frontend and the backend.

### Technologies Used:

* **Python:** The core programming language used for script processing and analysis.
* **Flask:** A lightweight web framework that manages HTTP requests, processes scripts, and returns analysis results.

### Backend Features:

* **Handles User Requests:** Accepts script files from the frontend for analysis.
* **Processes Code Analysis:** Extracts key metrics such as line count, function count, and complexity.
* **Performs Complexity Analysis:** Measures cyclomatic complexity and code maintainability.
* **Returns Results to Frontend:** Sends processed data back to the user interface for display.

# FEASIBILITY STUDY

## 5. FEASIBILITY STUDY

The **Feasibility Study** evaluates the viability of the **Script Analysis Web App** by analyzing its technical, operational, economic, and scheduling feasibility. This chapter ensures that the project is practical, cost-effective, and achievable within the given constraints.

### Technical Feasibility:

Technical feasibility assesses whether the available technology, tools, and infrastructure can support the successful development and deployment of the web application.

### Key Factors:

* **Technology Stack:** The application uses **HTML, CSS, JavaScript (Frontend)** and **Python Flask (Backend)**, which are widely supported and well-documented technologies.
* **Code Analysis Tools:** The project leverages tools like **Pylint** to analyze code structure and complexity.
* **Cross-Platform Compatibility:** The web application runs on all major operating systems, including Windows, Linux, and macOS.
* **Hosting & Deployment:** Can be deployed on cloud platforms such as **Heroku, AWS, or DigitalOcean**, ensuring scalability.

### Financial Feasibility:

Economic feasibility evaluates whether the benefits of the project outweigh the costs of development, deployment, and maintenance.

### Cost Considerations:

* **Development Costs:** No licensing fees required for open-source technologies (Flask, Python, JavaScript).
* **Hosting & Infrastructure:** Minimal costs if hosted on **free-tier cloud platforms** like Heroku or GitHub Pages.
* **Maintenance Costs:** Low, as the application does not require a complex database or third-party integrations.

### Operational Feasibility:

Operational feasibility examines whether the application meets user needs and is easy to use.

### Key Factors:

* **User-Friendliness:** Simple interface for uploading and analyzing scripts.
* **Automation:** Eliminates manual code analysis, improving efficiency.
* **Multi-Language Support:** Supports various programming languages, increasing usability.
* **Real-Time Feedback:** Users receive instant results without waiting.

### Market Feasibility:

Market feasibility assesses whether there is demand for the **Script Analysis Web App** and if it has a competitive advantage.

### Target Audience:

* **Software Developers** – Need tools to analyze and optimize code.
* **QA Engineers** – Require automated code analysis for testing.
* **Students & Educators** – Can use the tool for learning programming concepts.

### Market Demand:

* **Growing need for automated code review tools** due to the rising complexity of software development.
* **Competitor analysis:** Existing tools like **SonarQube** and **Codacy** focus on enterprise use, while this tool provides a **lightweight, free, and accessible alternative**.

### Legal and Ethical Feasibility:

Legal and ethical feasibility examines whether the project complies with relevant regulations and follows ethical coding practices.

### Legal Considerations:

* **Compliance with Open-Source Licenses:** Ensures that any external libraries (e.g., Flask, Pylint) comply with **MIT, Apache, or GPL licenses**.
* **Data Privacy & Security:** Since no database is used, **no sensitive user data is stored**, minimizing legal risks.
* **Copyright Protection:** The application must respect **intellectual property laws**, ensuring users only analyze their own code.

### Ethical Considerations:

* **Fair Use of Code Analysis:** The tool should not be used to analyze proprietary code without permission.
* **Transparency:** Users should be informed about how the tool processes their scripts.

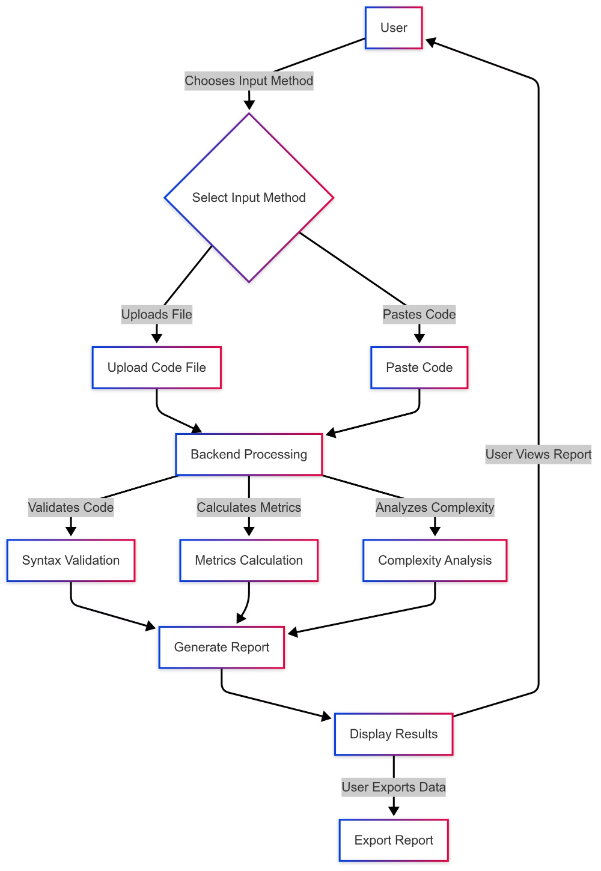
# SYSTEM DESIGN

## 6. SYSTEM DESIGN

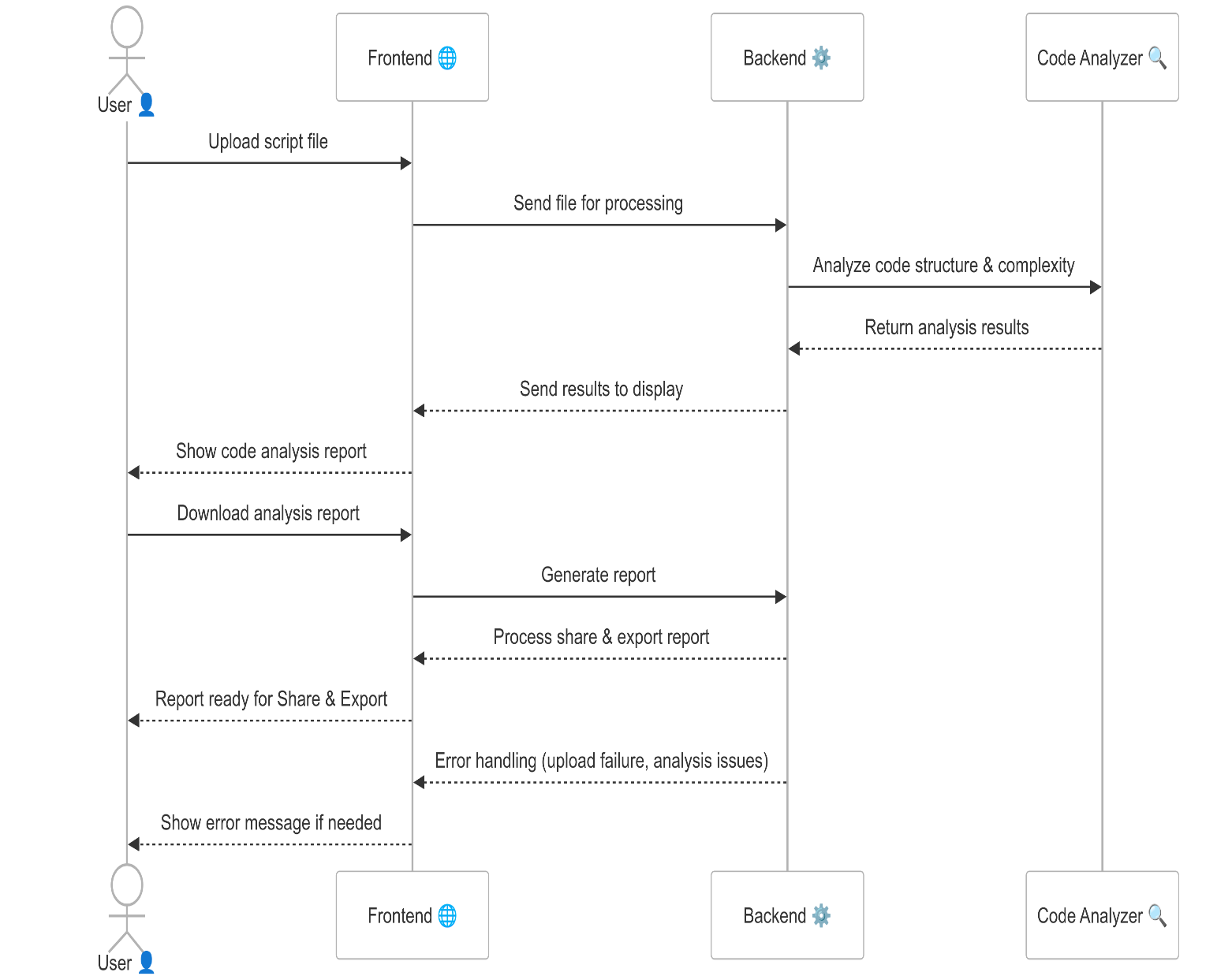
**6.1. CASS DIAGRAM:**



**6.2. Data Flow:**

****

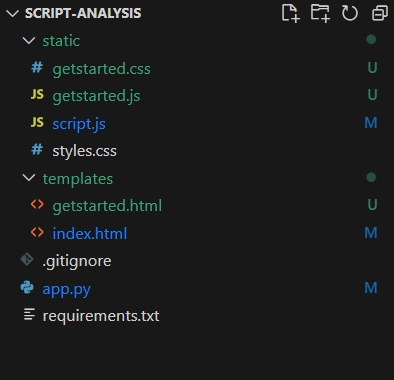
**6.3. Sequence Diagram:**

****

# APPENDICES

## APPENDICES

* 1. **FOLDER STRUCTURE:**

****

* 1. **Coding:**

**index.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Code Analysis Tool</title>

*<!-- Add CodeMirror CSS -->*

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/codemirror.min.css">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/theme/monokai.min.css">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/theme/dracula.min.css">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/theme/material.min.css">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/theme/nord.min.css">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/theme/github-dark.min.css">

    <link rel="stylesheet" href="{{ url\_for('static', filename='styles.css') }}">

</head>

<body>

    <div class="loading-overlay">

        <div class="loading-spinner"></div>

        <div class="loading-text">Analyzing your code...</div>

        <div class="loading-progress">

            <div class="loading-progress-bar"></div>

        </div>

        <div class="loading-status">Please wait while we analyze your code</div>

    </div>

    <div class="container">

        <h1>Code Analysis Tool</h1>

        <form id="uploadForm">

            <div id="uploadSection">

                <div class="top-controls">

                    <div class="language-select">

            <label for="language">Select Language:</label>

            <select id="language" name="language" required>

                            <option value="auto">Auto Detect</option>

                <option value="python">Python</option>

                <option value="javascript">JavaScript</option>

                <option value="cpp">C++</option>

                <option value="java">Java</option>

                <option value="ruby">Ruby</option>

                <option value="go">Go</option>

                <option value="swift">Swift</option>

                <option value="php">PHP</option>

                <option value="csharp">C#</option>

            </select>

                    </div>

                    <div class="file-input-container">

                        <div class="file-upload-area">

                            <label for="file" class="file-upload-label">

                                <svg width="16" height="16" viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                    <path d="M21 15v4a2 2 0 0 1-2 2H5a2 2 0 0 1-2-2v-4"></path>

                                    <polyline points="7 8 12 3 17 8"></polyline>

                                    <line x1="12" y1="3" x2="12" y2="15"></line>

                                </svg>

                                <span>Choose a file or drag it here</span>

                            </label>

                            <input type="file" id="file" name="file" accept=".txt,.java,.js,.py,.r,.c,.cpp,.cs,.jl,.go,.pl,.m,.kt,.php,.rs,.rb,.lua,.ts,.sas,.hx,.lisp,.f,.scala,.as,.asm,.clj,.coffee,.dart,.cbl,.groovy,.ex,.erl,.pas,.hs,.swift,.rkt,.ml,.scm,.cr,.elm,.fs,.tcl,.vb,.vala,.adb,.m,.objc">

                            <div class="file-info">

                                <span class="file-name" id="file-name">No file chosen</span>

                            </div>

                        </div>

                    </div>

                    <div class="result-actions" style="display: none;">

                        <div class="action-group">

                            <button class="action-btn share-btn" title="Share Results">

                                <svg width="16" height="16" viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                    <circle cx="18" cy="5" r="3"></circle>

                                    <circle cx="6" cy="12" r="3"></circle>

                                    <circle cx="18" cy="19" r="3"></circle>

                                    <line x1="8.59" y1="13.51" x2="15.42" y2="17.49"></line>

                                    <line x1="15.41" y1="6.51" x2="8.59" y2="10.49"></line>

                                </svg>

                                Share

                                <div class="dropdown-content">

                                    <a href="#" id="emailShare">

                                        <svg width="16" height="16" viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                            <path d="M4 4h16c1.1 0 2 .9 2 2v12c0 1.1-.9 2-2 2H4c-1.1 0-2-.9-2-2V6c0-1.1.9-2 2-2z"></path>

                                            <polyline points="22,6 12,13 2,6"></polyline>

                                        </svg>

                                        Email

                                    </a>

                                    <a href="#" id="whatsappShare">

                                        <svg width="16" height="16" viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                            <path d="M21 11.5a8.38 8.38 0 0 1-.9 3.8 8.5 8.5 0 0 1-7.6 4.7 8.38 8.38 0 0 1-3.8-.9L3 21l1.9-5.7a8.38 8.38 0 0 1-.9-3.8 8.5 8.5 0 0 1 4.7-7.6 8.38 8.38 0 0 1 3.8-.9h.5a8.48 8.48 0 0 1 8 8v.5z"></path>

                                        </svg>

                                        WhatsApp

                                    </a>

                                </div>

                            </button>

                            <button class="action-btn export-btn" title="Export Results">

                                <svg width="16" height="16" viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                    <path d="M21 15v4a2 2 0 0 1-2 2H5a2 2 0 0 1-2-2v-4"></path>

                                    <polyline points="7 10 12 15 17 10"></polyline>

                                    <line x1="12" y1="15" x2="12" y2="3"></line>

                                </svg>

                                Export

                                <div class="dropdown-content">

                                    <a href="#" id="exportPDF">

                                        <svg width="16" height="16" viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                            <path d="M14 2H6a2 2 0 0 0-2 2v16a2 2 0 0 0 2 2h12a2 2 0 0 0 2-2V8z"></path>

                                            <polyline points="14 2 14 8 20 8"></polyline>

                                        </svg>

                                        Export as PDF

                                    </a>

                                    <a href="#" id="exportTXT">

                                        <svg width="16" height="16" viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                            <path d="M14 2H6a2 2 0 0 0-2 2v16a2 2 0 0 0 2 2h12a2 2 0 0 0 2-2V8z"></path>

                                            <polyline points="14 2 14 8 20 8"></polyline>

                                            <line x1="16" y1="13" x2="8" y2="13"></line>

                                            <line x1="16" y1="17" x2="8" y2="17"></line>

                                        </svg>

                                        Export as Text

                                    </a>

                                    <a href="#" id="exportImage">

                                        <svg width="16" height="16" viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                            <rect x="3" y="3" width="18" height="18" rx="2" ry="2"></rect>

                                            <circle cx="8.5" cy="8.5" r="1.5"></circle>

                                            <polyline points="21 15 16 10 5 21"></polyline>

                                        </svg>

                                        Save as Image

                                    </a>

                                </div>

                            </button>

                        </div>

                    </div>

                </div>

                <div class="input-section">

                    <div class="editor-section">

                        <label for="codeEditor">Paste Your Code:</label>

                        <div class="editor-toolbar">

                            <button id="formatCode" title="Format Code (Ctrl+F)">

                                <svg viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                    <path d="M21 10H3M21 6H3M21 14H3M21 18H3"/>

                                </svg>

                                Format

                            </button>

                            <button id="copyCode" title="Copy Code (Ctrl+C)">

                                <svg viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                    <rect x="9" y="9" width="13" height="13" rx="2" ry="2"/>

                                    <path d="M5 15H4a2 2 0 0 1-2-2V4a2 2 0 0 1 2-2h9a2 2 0 0 1 2 2v1"/>

                                </svg>

                                Copy

                            </button>

                            <button id="clearCode" title="Clear Editor">

                                <svg viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                    <path d="M19 6v14a2 2 0 0 1-2 2H7a2 2 0 0 1-2-2V6m3 0V4a2 2 0 0 1 2-2h4a2 2 0 0 1 2 2v2"/>

                                    <line x1="10" y1="11" x2="10" y2="17"/>

                                    <line x1="14" y1="11" x2="14" y2="17"/>

                                </svg>

                                Clear

                            </button>

                            <div class="divider"></div>

                            <button id="indentCode" title="Auto Indent (Ctrl+I)">

                                <svg viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">

                                    <path d="M3 6h18M7 12h14M7 18h14M3 12l2-3m-2 3l2 3"/>

                                </svg>

                                Indent

                            </button>

                            <div class="divider"></div>

                            <select id="themeSelect" title="Select Theme">

                                <option value="monokai">Monokai</option>

                                <option value="dracula">Dracula</option>

                                <option value="github-dark">GitHub Dark</option>

                                <option value="material">Material</option>

                                <option value="nord">Nord</option>

                            </select>

                            <select id="fontSizeSelect" title="Font Size">

                                <option value="10">10px</option>

                                <option value="11">11px</option>

                                <option value="12">12px</option>

                                <option value="13">13px</option>

                                <option value="14" selected>14px</option>

                                <option value="16">16px</option>

                                <option value="18">18px</option>

                            </select>

                        </div>

                        <textarea id="codeEditor"></textarea>

                    </div>

                </div>

            </div>

            <button type="submit" class="analyze-btn">Analyze Code</button>

        </form>

        <div id="results"></div>

    </div>

*<!-- Add CodeMirror JS -->*

    <script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/codemirror.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/mode/python/python.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/mode/javascript/javascript.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/mode/clike/clike.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/mode/ruby/ruby.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/mode/go/go.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/mode/swift/swift.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.2/mode/php/php.min.js"></script>

    <script src="{{ url\_for('static', filename='script.js') }}"></script>

*<!-- Add these before your closing </body> tag -->*

    <script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.5.1/jspdf.umd.min.js"></script>

    <script src="https://html2canvas.hertzen.com/dist/html2canvas.min.js"></script>

</body>

</html>

### styles.css

### body {

### font-family: Arial, sans-serif;

### margin: 0;

### padding: 20px;

### background-color: #f4f4f4;

### min-height: 100vh;

### }

### .container {

### max-width: 1200px;

### margin: 0 auto;

### padding: 20px;

### background-color: white;

### border-radius: 8px;

### box-shadow: 0 2px 4px rgba(0,0,0,0.1);

### }

### h1 {

### text-align: center;

### color: #333;

### margin-bottom: 30px;

### }

### h2 {

### color: #2c3e50;

### border-bottom: 2px solid #eee;

### padding-bottom: 10px;

### margin-top: 30px;

### }

### h3 {

### color: #34495e;

### margin: 10px 0;

### }

### form#uploadForm {

### margin: 20px 0;

### padding: 25px;

### border: 1px solid #ddd;

### border-radius: 8px;

### background-color: #f8f9fa;

### max-width: 1000px;

### margin-left: auto;

### margin-right: auto;

### }

### .form-group {

### margin-bottom: 20px;

### }

### label {

### display: block;

### margin: 15px 0 8px;

### color: #333;

### font-weight: bold;

### }

### select {

### width: 100%;

### max-width: 300px;

### }

### */\* Keep only these specific button styles \*/*

### .upload-btn {

### display: inline-flex;

### align-items: center;

### gap: 6px;

### padding: 6px 10px;

### background-color: #4CAF50;

### color: white;

### border: none;

### border-radius: 4px;

### cursor: pointer;

### font-size: 13px;

### transition: background-color 0.3s;

### height: 28px;

### width: auto;

### min-width: 90px;

### flex-shrink: 0;

### }

### .upload-btn:hover {

### background-color: #45a049;

### }

### .upload-btn svg {

### width: 14px;

### height: 14px;

### }

### .analyze-btn {

### margin-top: 20px;

### background-color: #2196F3;

### max-width: 200px;

### display: block;

### margin-left: auto;

### margin-right: auto;

### padding: 10px 20px;

### font-size: 15px;

### border: none;

### border-radius: 4px;

### color: white;

### cursor: pointer;

### transition: background-color 0.3s;

### }

### .analyze-btn:hover {

### background-color: #1976D2;

### }

### #results {

### margin-top: 30px;

### }

### .analysis-grid {

### display: grid;

### grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));

### gap: 20px;

### margin: 20px 0;

### }

### .metrics, .code-structure {

### background-color: #f8f9fa;

### padding: 20px;

### border-radius: 8px;

### box-shadow: 0 1px 3px rgba(0,0,0,0.1);

### }

### .classes, .functions, .methods, .objects, .comments {

### padding: 20px;

### border-radius: 8px;

### margin: 15px 0;

### box-shadow: 0 1px 3px rgba(0,0,0,0.1);

### }

### .classes ul, .functions ul, .methods ul, .objects ul, .comments ul {

### list-style-type: none;

### padding-left: 0;

### max-height: 300px;

### overflow-y: auto;

### margin: 0;

### }

### .classes li, .functions li, .methods li, .objects li, .comments li {

### padding: 10px;

### margin: 5px 0;

### background-color: #fff;

### border-radius: 4px;

### font-family: monospace;

### font-size: 14px;

### border: 1px solid rgba(0,0,0,0.1);

### }

### */\* Section Colors \*/*

### .classes { background-color: #e8f5e9; }

### .functions { background-color: #f3e5f5; }

### .methods { background-color: #e3f2fd; }

### .objects { background-color: #fff3e0; }

### .comments { background-color: #fce4ec; }

### .output {

### background-color: #f8f9fa;

### padding: 20px;

### border-radius: 8px;

### margin-top: 20px;

### overflow-x: auto;

### }

### pre {

### margin: 0;

### white-space: pre-wrap;

### font-size: 14px;

### }

### .error {

### color: #dc3545;

### padding: 15px;

### border: 1px solid #dc3545;

### border-radius: 4px;

### margin: 10px 0;

### background-color: #fff;

### }

### */\* Responsive Design \*/*

### @media (max-width: 768px) {

### body {

### padding: 10px;

### }

### 

### .container {

### padding: 15px;

### }

### 

### .analysis-grid {

### grid-template-columns: 1fr;

### }

### 

### h1 {

### font-size: 24px;

### }

### 

### h2 {

### font-size: 20px;

### }

### 

### h3 {

### font-size: 18px;

### }

### }

### */\* Scrollbar Styling \*/*

### ::-webkit-scrollbar {

### width: 8px;

### height: 8px;

### }

### ::-webkit-scrollbar-track {

### background: #f1f1f1;

### border-radius: 4px;

### }

### ::-webkit-scrollbar-thumb {

### background: #888;

### border-radius: 4px;

### }

### ::-webkit-scrollbar-thumb:hover {

### background: #555;

### }

### .dropdown-content {

### display: none;

### position: absolute;

### top: 100%;

### left: 0;

### background-color: white;

### min-width: 200px;

### box-shadow: 0 4px 8px rgba(0,0,0,0.1);

### border-radius: 4px;

### z-index: 1000;

### margin-top: 5px;

### border: 1px solid #ddd;

### opacity: 0;

### transform: translateY(-10px);

### transition: opacity 0.3s ease, transform 0.3s ease;

### }

### */\* Show dropdown on button hover AND when hovering the dropdown itself \*/*

### .action-btn:hover .dropdown-content,

### .dropdown-content:hover {

### display: block;

### opacity: 1;

### transform: translateY(0);

### }

### */\* Add padding to create hover space between button and dropdown \*/*

### .action-btn::after {

### content: '';

### position: absolute;

### height: 10px;

### left: 0;

### right: 0;

### bottom: -10px;

### }

### .dropdown-content a {

### display: flex;

### align-items: center;

### gap: 8px;

### padding: 12px 16px;

### color: #333;

### text-decoration: none;

### transition: all 0.2s ease;

### }

### .dropdown-content a:hover {

### background-color: #f8f9fa;

### color: #000;

### }

### .dropdown-content a:first-child {

### border-radius: 4px 4px 0 0;

### }

### */\* .dropdown-content a:last-child {border-content: 0 0 4px 4px;} \*/*

### .dropdown-content svg {

### width: 16px;

### height: 16px;

### stroke: #666;

### transition: stroke 0.2s ease;

### }

### .dropdown-content a:hover svg {

### stroke: #333;

### }

### */\* Mobile adjustments \*/*

### @media (max-width: 768px) {

### .action-group {

### flex-direction: column;

### width: 100%;

### }

### 

### .action-btn {

### width: 100%;

### justify-content: center;

### }

### 

### .dropdown-content {

### position: static;

### transform: none;

### }

### 

### .action-btn::after {

### display: none;

### }

### 

### script.js

let editor;

document.addEventListener('DOMContentLoaded', () => {

*// Initialize CodeMirror*

    editor = CodeMirror.fromTextArea(document.getElementById('codeEditor'), {

        mode: 'python',

        theme: 'monokai',

        lineNumbers: true,

        autoCloseBrackets: true,

        matchBrackets: true,

        indentUnit: 4,

        tabSize: 4,

        lineWrapping: true,

        height: 'auto'

    });

*// File upload elements*

    const fileUploadArea = document.querySelector('.file-upload-area');

    const fileInfo = document.querySelector('.file-info');

    const fileName = document.getElementById('file-name');

    const fileInput = document.getElementById('file');

    const languageSelect = document.getElementById('language');

*// Handle file selection*

    function **handleFileSelect**(*file*) {

        if (*file*) {

            fileName.textContent = *file*.name;

            fileInfo.classList.add('show');

*// Read and display file content in the editor*

            const reader = new FileReader();

            reader.onload = function(*e*) {

                editor.setValue(*e*.target.result);

            };

            reader.readAsText(*file*);

*// Auto-detect language if needed*

            if (languageSelect.value === 'auto') {

                const detectedLanguage = detectLanguageFromFile(*file*.name);

                if (detectedLanguage) {

                    languageSelect.value = detectedLanguage;

                    editor.setOption('mode', getEditorMode(detectedLanguage));

                }

            }

        } else {

            fileName.textContent = 'No file chosen';

            fileInfo.classList.remove('show');

            editor.setValue(''); *// Clear editor if no file selected*

        }

    }

*// File input change handler*

    fileInput.addEventListener('change', (*e*) => {

        handleFileSelect(e.target.files[0]);

    });

*// Drag and drop handlers*

    ['dragenter', 'dragover', 'dragleave', 'drop'].forEach(*eventName* => {

        fileUploadArea.addEventListener(eventName, preventDefaults, false);

    });

    function preventDefaults(*e*) {

        e.preventDefault();

        e.stopPropagation();

    }

    ['dragenter', 'dragover'].forEach(*eventName* => {

        fileUploadArea.addEventListener(eventName, () => fileUploadArea.classList.add('drag-over'), false);

    });

    ['dragleave', 'drop'].forEach(*eventName* => {

        fileUploadArea.addEventListener(eventName, () => fileUploadArea.classList.remove('drag-over'), false);

    });

    fileUploadArea.addEventListener('drop', (*e*) => {

        const dt = e.dataTransfer;

        const files = dt.files;

        fileInput.files = files;

        handleFileSelect(files[0]);

    });

*// Content-based language detection for pasted code*

    editor.on('change', () => {

        const content = editor.getValue();

        if (content.length > 10 && languageSelect.value === 'auto') {

            const detectedLanguage = detectLanguageFromContent(content);

            if (detectedLanguage) {

                languageSelect.value = detectedLanguage;

                editor.setOption('mode', getEditorMode(detectedLanguage));

            }

        }

    });

*// Add loading state handling*

    function showLoading() {

        document.querySelector('.loading-overlay').style.display = 'flex';

        document.querySelector('.loading-progress-bar').style.width = '0%';

    }

    function hideLoading() {

        document.querySelector('.loading-overlay').style.display = 'none';

    }

*// Update the form submission handler*

    document.getElementById('uploadForm').addEventListener('submit', async (*e*) => {

        e.preventDefault();

        const resultsDiv = document.getElementById('results');

        resultsDiv.innerHTML = ''; *// Clear previous results*

        showLoading(); *// Show loading overlay*

        const formData = new FormData();

        const codeContent = editor.getValue();

        const selectedLanguage = document.getElementById('language').value;

        if (!codeContent.trim()) {

            hideLoading();

            resultsDiv.innerHTML = `<div class="error fade-in">Please enter some code</div>`;

            return;

        }

        try {

            const extension = getFileExtension(selectedLanguage);

            const blob = new Blob([codeContent], { type: 'text/plain' });

            const file = new File([blob], `code${extension}`, { type: 'text/plain' });

            formData.append('file', file);

            formData.append('language', selectedLanguage);

            const response = await fetch('/analyze', {

                method: 'POST',

                body: formData

            });

            const data = await response.json();

*// Start 10-second loading animation*

            document.querySelector('.loading-progress-bar').style.animation = 'progress 5s linear forwards';

            await new Promise(*resolve* => setTimeout(resolve, 5000));

            if (data.error) {

                resultsDiv.innerHTML = `<div class="error fade-in">Error: ${data.error}</div>`;

                return;

            }

            let resultsHTML = `

                <div class="results-container fade-in">

                    <h2>Analysis Results</h2>

                    <div class="analysis-grid">

                        <div class="metrics">

                            <h3>Basic Metrics</h3>

                            <p>Total Lines: ${data.total\_lines}</p>

                            <p>Empty Lines: ${data.empty\_lines}</p>

                            <p>Comment Lines: ${data.comment\_lines}</p>

                            <p>Import Count: ${data.import\_count}</p>

                        </div>

                        <div class="code-structure">

                            <h3>Code Structure</h3>

                            <p>Classes: ${data.code\_structure.classes}</p>

                            <p>Functions: ${data.code\_structure.functions}</p>

                            <p>Methods: ${data.code\_structure.methods}</p>

                            <p>Objects: ${data.code\_structure.objects}</p>

                            <p>Imports: ${data.code\_structure.imports}</p>

                        </div>

                    </div>

                    <div class="analysis-section complexity">

                        <h3>Code Complexity Analysis</h3>

                        <div class="metric-item">

                            <span class="metric-label">Cyclomatic Complexity</span>

                            <span class="metric-value">${data.complexity\_analysis.cyclomatic\_complexity}</span>

                            <span class="metric-info" title="Measures the number of linearly independent paths through the code">ⓘ</span>

                        </div>

                        <div class="metric-item">

                            <span class="metric-label">Maximum Nesting Depth</span>

                            <span class="metric-value">${data.complexity\_analysis.max\_nesting\_depth}</span>

                            <span class="metric-info" title="The deepest level of nested code blocks">ⓘ</span>

                        </div>

                        <div class="metric-item">

                            <span class="metric-label">Maintainability Index</span>

                            <span class="metric-value">${data.complexity\_analysis.maintainability\_index}</span>

                            <span class="metric-info" title="Higher values indicate better maintainability">ⓘ</span>

                        </div>

                    </div>

                    <div class="analysis-grid">

                        ${data.imports.length > 0 ? `

                            <div class="imports">

                                <h3>Imports Found</h3>

                                <ul>

                                    ${data.imports.map(*imp* => `<li>${imp}</li>`).join('')}

                                </ul>

                            </div>

                        ` : ''}

                        ${data.classes.length > 0 ? `

                            <div class="classes">

                                <h3>Classes Found</h3>

                                <ul>

                                    ${data.classes.map(*cls* => `<li>${cls}</li>`).join('')}

                                </ul>

                            </div>

                        ` : ''}

                        ${data.functions.length > 0 ? `

                            <div class="functions">

                                <h3>Functions Found</h3>

                                <ul>

                                    ${data.functions.map(*func* => `<li>${func}</li>`).join('')}

                                </ul>

                            </div>

                        ` : ''}

                        ${data.methods.length > 0 ? `

                            <div class="methods">

                                <h3>Methods Found</h3>

                                <ul>

                                    ${data.methods.map(*method* => `<li>${method}</li>`).join('')}

                                </ul>

                            </div>

                        ` : ''}

                        ${data.objects.length > 0 ? `

                            <div class="objects">

                                <h3>Objects Found</h3>

                                <ul>

                                    ${data.objects.map(*obj* => `<li>${obj}</li>`).join('')}

                                </ul>

                            </div>

                        ` : ''}

                    </div>

                    ${data.comments.length > 0 ? `

                        <div class="comments">

                            <h3>Comments Found</h3>

                            <ul>

                                ${data.comments.map(*comment* => `<li>${comment}</li>`).join('')}

                            </ul>

                        </div>

                    ` : ''}

                    <div class="output">

                        <h3>Analysis Output</h3>

                        <pre>${data.output || 'No issues found'}</pre>

                    </div>

                </div>

            `;

            resultsDiv.innerHTML = resultsHTML;

            resultsDiv.classList.add('fade-in');

*// Show result actions when results are displayed*

            const showResultActions = () => {

                document.querySelector('.result-actions').style.display = 'block';

            };

*// block the action buttons from submitting the form*

            const actionButtons = document.querySelectorAll('.action-btn, .dropdown-content a');

            actionButtons.forEach(*btn* => {

                btn.addEventListener('click', (*e*) => {

                    e.preventDefault();

                    e.stopPropagation();

                    return false; *// Prevent form submission*

                });

            });

*// Share functionality*

            document.getElementById('emailShare').addEventListener('click', (*e*) => {

                e.preventDefault();

                const subject = 'Code Analysis Results';

                const body = formatAnalysisResults(window.analysisData);

*// Limit the body length and clean up the content*

                const cleanBody = body

                    .replace(/[^\x00-\x7F]/g, '') *// Remove non-ASCII characters*

                    .substring(0, 2000); *// Limit length to avoid URL length issues*

*// Add a note if content was truncated*

                const truncationNote = body.length > 2000 ?

                    '\n\n[Content truncated. Please use export options for full results.]' : '';

                try {

*// Try the mailto link first*

                    window.location.href = `mailto:?subject=${encodeURIComponent(subject)}&body=${encodeURIComponent(cleanBody + truncationNote)}`;

                } catch (error) {

*// Fallback: Copy to clipboard and show instructions*

                    navigator.clipboard.writeText(body).then(() => {

                        alert('Analysis results copied to clipboard. Please paste into your email client.');

                    }).catch(() => {

*// If clipboard fails, show content in a modal or alert*

                        alert('Please copy the analysis results from the page and paste into your email client.');

                    });

                }

            });

            document.getElementById('whatsappShare').addEventListener('click', (*e*) => {

                e.preventDefault();

                const text = formatAnalysisResults(data);

                window.open(`https://wa.me/?text=${encodeURIComponent(text)}`, '\_blank');

            });

*// Export functionality*

            document.getElementById('exportPDF').addEventListener('click', async (*e*) => {

                e.preventDefault();

                const { jsPDF } = window.jspdf;

                const doc = new jsPDF();

*// Set font and size*

                doc.setFont('helvetica');

                doc.setFontSize(16);

*// Add title*

                doc.text('Code Analysis Results', 20, 20);

                doc.setFontSize(12);

*// Get the formatted content*

                const content = formatAnalysisResults(data);

*// Split content into lines that fit the page width*

                const lines = doc.splitTextToSize(content, 170);

                let y = 30;

                const pageHeight = doc.internal.pageSize.height;

*// Add lines with pagination*

                lines.forEach(*line* => {

                    if (y > pageHeight - 20) {

                        doc.addPage();

                        y = 20;

                    }

                    doc.text(line, 20, y);

                    y += 6;

                });

                doc.save('analysis-results.pdf');

            });

            document.getElementById('exportTXT').addEventListener('click', (*e*) => {

                e.preventDefault();

                const content = formatAnalysisResults(data);

                const blob = new Blob([content], { type: 'text/plain' });

                const url = window.URL.createObjectURL(blob);

                const a = document.createElement('a');

                a.href = url;

                a.download = 'analysis-results.txt';

                document.body.appendChild(a);

                a.click();

                window.URL.revokeObjectURL(url);

                document.body.removeChild(a);

            });

            document.getElementById('exportImage').addEventListener('click', (*e*) => {

                e.preventDefault();

                html2canvas(document.getElementById('results')).then(*canvas* => {

                    const link = document.createElement('a');

                    link.download = 'analysis-results.png';

                    link.href = canvas.toDataURL();

                    link.click();

                });

            });

*// Update the results display to show actions*

            const originalResultsHTML = resultsDiv.innerHTML;

            resultsDiv.innerHTML = originalResultsHTML;

            showResultActions();

        } catch (error) {

            resultsDiv.innerHTML = `<div class="error fade-in">Error: ${error.message}</div>`;

        } finally {

            hideLoading();

*// Reset progress bar animation*

            const progressBar = document.querySelector('.loading-progress-bar');

            progressBar.style.animation = 'none';

            progressBar.offsetHeight; *// Trigger reflow*

            progressBar.style.animation = '';

        }

    });

*// Format code button*

    document.getElementById('formatCode').addEventListener('click', (*e*) => {

        e.preventDefault();

        e.stopPropagation();

        try {

            const mode = editor.getMode().name;

            let code = editor.getValue();

            if (!code.trim()) {

                return; *// Don't format empty code*

            }

*// Basic formatting for different languages*

            switch (mode) {

                case 'javascript':

*// Basic JS formatting*

                    code = code.replace(/[{]/g, ' {\n    ')

                             .replace(/[}]/g, '\n}\n')

                             .replace(/;/g, ';\n')

                             .replace(/\n\s\*\n/g, '\n\n'); *// Remove extra newlines*

                    break;

                case 'python':

*// Basic Python formatting*

                    code = code.replace(/:\s\*/g, ':\n    ')

                             .replace(/\n\s\*\n/g, '\n\n')

                             .replace(/([^:]);/g, '$1\n'); *// Add newlines after statements*

                    break;

                default:

*// Generic formatting for other languages*

                    code = code.replace(/[{]/g, ' {\n    ')

                             .replace(/[}]/g, '\n}\n')

                             .replace(/;/g, ';\n')

                             .replace(/\n\s\*\n/g, '\n\n');

            }

*// Update editor content*

            editor.setValue(code);

*// Auto indent all lines*

            const totalLines = editor.lineCount();

            for (let i = 0; i < totalLines; i++) {

                editor.indentLine(i);

            }

*// Refresh the editor to update the display*

            editor.refresh();

        } catch (error) {

            console.error('Formatting failed:', error);

        }

    });

    document.getElementById('copyCode').addEventListener('click', () => {

        navigator.clipboard.writeText(editor.getValue())

            .then(() => {

                const btn = document.getElementById('copyCode');

                btn.innerHTML = '<svg>...</svg>Copied!';

                setTimeout(() => {

                    btn.innerHTML = '<svg>...</svg>Copy';

                }, 2000);

            });

    });

    document.getElementById('clearCode').addEventListener('click', () => {

        if (confirm('Are you sure you want to clear the editor?')) {

            editor.setValue('');

        }

    });

*// Update keyboard shortcuts - remove save-related shortcuts*

    editor.setOption('extraKeys', {

        'Ctrl-F': (*cm*) => {

            document.getElementById('formatCode').click();

        },

        'Ctrl-/': (*cm*) => {

            const selections = cm.getSelections();

            const mode = cm.getMode().name;

            const comment = mode === 'python' ? '#' : '//';

            const newSelections = selections.map(*selection* => {

                const lines = selection.split('\n');

                const commentedLines = lines.map(*line* => {

                    if (line.trimStart().startsWith(comment)) {

                        return line.replace(new RegExp(`^(\\s\*)${comment}\\s?`), '$1');

                    }

                    return line.replace(/^(\s\*)/, `$1${comment} `);

                });

                return commentedLines.join('\n');

            });

            cm.replaceSelections(newSelections);

        }

    });

*// Update theme select without localStorage*

    document.getElementById('themeSelect').addEventListener('change', (*e*) => {

        const theme = e.target.value;

        editor.setOption('theme', theme);

    });

*// Update font size select without localStorage*

    document.getElementById('fontSizeSelect').addEventListener('change', (*e*) => {

        const fontSize = e.target.value + 'px';

        document.querySelector('.CodeMirror').style.fontSize = fontSize;

    });

*// Reset everything on page load*

    window.addEventListener('load', () => {

        editor.setValue('');

        document.getElementById('language').value = 'auto';

        document.getElementById('themeSelect').value = 'monokai';

        document.getElementById('fontSizeSelect').value = '14';

        editor.setOption('theme', 'monokai');

        document.querySelector('.CodeMirror').style.fontSize = '14px';

    });

*// Indent code button*

    document.getElementById('indentCode').addEventListener('click', (*e*) => {

        e.preventDefault(); *// Prevent form submission*

        const totalLines = editor.lineCount();

        editor.operation(() => {

            for (let i = 0; i < totalLines; i++) {

                editor.indentLine(i);

            }

        });

        editor.refresh(); *// Refresh editor after indenting*

    });

*// Add this to prevent form submission when clicking editor toolbar buttons*

    document.querySelector('.editor-toolbar').addEventListener('click', (*e*) => {

        if (e.target.closest('button')) {

            e.preventDefault();

            e.stopPropagation();

        }

    });

*// Update language detection for pasted code*

    document.getElementById('language').addEventListener('change', (*e*) => {

        const selectedLanguage = e.target.value;

        if (selectedLanguage !== 'auto') {

            editor.setOption('mode', getEditorMode(selectedLanguage));

        }

    });

});

*// Helper function to get editor mode*

function getEditorMode(*language*) {

    const modeMap = {

        'python': {

            mode: 'python',

            mime: 'text/x-python',

            indentUnit: 4

        },

        'javascript': {

            mode: 'javascript',

            mime: 'text/javascript',

            indentUnit: 2

        },

        'cpp': {

            mode: 'clike',

            mime: 'text/x-c++src',

            indentUnit: 4

        },

        'java': {

            mode: 'clike',

            mime: 'text/x-java',

            indentUnit: 4

        },

        'ruby': {

            mode: 'ruby',

            mime: 'text/x-ruby',

            indentUnit: 2

        },

        'go': {

            mode: 'go',

            mime: 'text/x-go',

            indentUnit: 4

        },

        'swift': {

            mode: 'swift',

            mime: 'text/x-swift',

            indentUnit: 4

        },

        'php': {

            mode: 'php',

            mime: 'application/x-httpd-php',

            indentUnit: 4

        },

        'csharp': {

            mode: 'clike',

            mime: 'text/x-csharp',

            indentUnit: 4

        }

    };

    const config = modeMap[language] || { mode: 'text/plain', indentUnit: 4 };

    editor.setOption('mode', config.mime);

    editor.setOption('indentUnit', config.indentUnit);

    return config.mode;

}

*// Add this function to detect language from file extension*

function detectLanguageFromFile(*filename*) {

    const extensionMap = {

        '.py': 'python',

        '.js': 'javascript',

        '.cpp': 'cpp',

        '.hpp': 'cpp',

        '.h': 'cpp',

        '.java': 'java',

        '.rb': 'ruby',

        '.go': 'go',

        '.swift': 'swift',

        '.php': 'php',

        '.cs': 'csharp'

    };

    const ext = '.' + filename.split('.').pop().toLowerCase();

    return extensionMap[ext];

}

*// Add this helper function to get file extension*

function getFileExtension(*language*) {

    const extensionMap = {

        'python': '.py',

        'javascript': '.js',

        'cpp': '.cpp',

        'java': '.java',

        'ruby': '.rb',

        'go': '.go',

        'swift': '.swift',

        'php': '.php',

        'csharp': '.cs'

    };

    return extensionMap[language] || '.txt';

}

*// Add this function to detect language from content*

function detectLanguageFromContent(*content*) {

    const patterns = {

        python: {

            keywords: /\b(def|class|import|from|if|for|while|try|except|with|async|await)\b/,

            syntax: /:\s\*$/m,

            imports: /^(?:from\s+\w+(?:\.\w+)\*\s+import|\s\*import\s+\w+)/m

        },

        javascript: {

            keywords: /\b(function|const|let|var|if|for|while|try|catch|class|import|export)\b/,

            syntax: /[{};]/,

            imports: /^(?:import\s+.\*\s+from\s+[\'"].\*[\'"]|require\s\*\([\'"].\*[\'"]\))/m

        },

        cpp: {

            keywords: /\b(class|struct|namespace|template|public|private|protected)\b/,

            syntax: /::|->|<>/,

            includes: /#include\s\*[<"]/

        },

        java: {

            keywords: /\b(public|private|protected|class|interface|extends|implements|package)\b/,

            syntax: /;$/m,

            imports: /^import\s+[\w.]+(?:\s\*\\*)?;/m

        },

        ruby: {

            keywords: /\b(def|class|module|require|include|attr\_accessor)\b/,

            syntax: /end$/m,

            requires: /^require\s+[\'"].\*[\'"]/

        },

        go: {

            keywords: /\b(func|type|struct|interface|package|import|go|chan|defer)\b/,

            syntax: /:\=|<-/,

            imports: /^import\s+(?:\([^)]+\)|"[^"]+")/m

        },

        swift: {

            keywords: /\b(class|struct|enum|protocol|extension|guard|let|var)\b/,

            syntax: /->|@/,

            imports: /^import\s+\w+/m

        },

        php: {

            keywords: /\b(function|class|namespace|use|public|private|protected)\b/,

            syntax: /\$\w+|<?php/,

            imports: /^(?:require|include)(?:\_once)?\s\*\([\'"].\*[\'"]\)/m

        },

        csharp: {

            keywords: /\b(class|namespace|using|public|private|protected|async|await)\b/,

            syntax: /;$/m,

            imports: /^using\s+[\w.]+;/m

        }

    };

*// Score each language based on matches*

    const scores = Object.entries(patterns).map(([*lang*, *pattern*]) => {

        let score = 0;

        const contentSample = content.slice(0, 1000); *// Check first 1000 chars*

        if (pattern.keywords.test(contentSample)) score += 2;

        if (pattern.syntax.test(contentSample)) score += 1;

        if (pattern.imports?.test(contentSample)) score += 3;

        return { language: lang, score };

    });

*// Return the language with highest score*

    const bestMatch = scores.reduce((*max*, *curr*) =>

        curr.score > max.score ? curr : max

    );

    return bestMatch.score > 2 ? bestMatch.language : null;

}

*// Update the formatAnalysisResults function to be more email-friendly*

function formatAnalysisResults(*data*) {

    if (!data) return 'No analysis data available.';

    try {

        return `

Code Analysis Results

--------------------

BASIC METRICS

Total Lines: ${data.total\_lines}

Empty Lines: ${data.empty\_lines}

Comment Lines: ${data.comment\_lines}

Import Count: ${data.import\_count}

CODE STRUCTURE

Classes: ${data.code\_structure.classes}

Functions: ${data.code\_structure.functions}

Methods: ${data.code\_structure.methods}

Objects: ${data.code\_structure.objects}

Imports: ${data.code\_structure.imports}

${data.imports.length > 0 ? `IMPORTS FOUND\n${data.imports.map(*imp* => `- ${imp}`).join('\n')}\n` : ''}

${data.classes.length > 0 ? `\nCLASSES FOUND\n${data.classes.map(*cls* => `- ${cls}`).join('\n')}\n` : ''}

${data.functions.length > 0 ? `\nFUNCTIONS FOUND\n${data.functions.map(*func* => `- ${func}`).join('\n')}\n` : ''}

${data.methods.length > 0 ? `\nMETHODS FOUND\n${data.methods.map(*method* => `- ${method}`).join('\n')}\n` : ''}

${data.objects.length > 0 ? `\nOBJECTS FOUND\n${data.objects.map(*obj* => `- ${obj}`).join('\n')}\n` : ''}

${data.comments.length > 0 ? `\nCOMMENTS FOUND\n${data.comments.map(*comment* => `- ${comment}`).join('\n')}\n` : ''}

ANALYSIS OUTPUT

${data.output || 'No issues found'}`;

    } catch (error) {

        console.error('Error formatting results:', error);

        return 'Error formatting analysis results. Please try exporting as PDF or TXT instead.';

    }

}

### getstarted.html

 <!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Get Started - Code Analysis Tool</title>

    <link rel="stylesheet" href="{{ url\_for('static', filename='getstarted.css') }}">

</head>

<body>

    <nav class="navbar">

        <div class="nav-container">

            <a href="/" class="logo">Code Analysis Tool</a>

            <div class="nav-links">

*<!-- <a href="/" class="nav-link">Home</a> -->*

                <a href="/getstarted" class="nav-link active">Get Started</a>

            </div>

        </div>

    </nav>

    <div class="hero-section">

        <h1>Welcome to Code Analysis Tool</h1>

        <p>Analyze your code with powerful metrics and insights</p>

    </div>

    <div class="container">

        <div class="steps-section">

            <h2>How to Use the Tool</h2>

            <div class="step-card">

                <div class="step-number">1</div>

                <div class="step-content">

                    <h3>Select Your Programming Language</h3>

                    <p>Choose from multiple supported languages or use auto-detect:</p>

                    <ul>

                        <li>Python</li>

                        <li>JavaScript</li>

                        <li>Java</li>

                        <li>C++</li>

                        <li>And more...</li>

                    </ul>

                </div>

            </div>

            <div class="step-card">

                <div class="step-number">2</div>

                <div class="step-content">

                    <h3>Upload Your Code</h3>

                    <p>Two ways to input your code:</p>

                    <ul>

                        <li>Upload a file directly</li>

                        <li>Copy and paste your code into the editor</li>

                    </ul>

                    <div class="tip">Tip: You can drag and drop files directly into the upload area</div>

                </div>

            </div>

            <div class="step-card">

                <div class="step-number">3</div>

                <div class="step-content">

                    <h3>Analyze Your Code</h3>

                    <p>Get comprehensive analysis including:</p>

                    <ul>

                        <li>Code Metrics</li>

                        <li>Complexity Analysis</li>

                        <li>Structure Overview</li>

                    </ul>

                </div>

            </div>

            <div class="step-card">

                <div class="step-number">4</div>

                <div class="step-content">

                    <h3>Review Results</h3>

                    <p>Understand your code better with:</p>

                    <ul>

                        <li>Visual Metrics</li>

                        <li>Detailed Reports</li>

                        <li>Export Options</li>

                    </ul>

                </div>

            </div>

        </div>

        <div class="features-grid">

            <div class="feature-card">

                <div class="feature-icon"></div>

                <h3>Basic Metrics</h3>

                <p>Analyze code structure, line counts, and basic statistics</p>

                <button class="learn-more" data-section="basic-metrics">Learn More</button>

            </div>

            <div class="feature-card">

                <div class="feature-icon"></div>

                <h3>Code Structure</h3>

                <p>Identify classes, functions, methods, and code organization</p>

                <button class="learn-more" data-section="code-structure">Learn More</button>

            </div>

            <div class="feature-card">

                <div class="feature-icon"></div>

                <h3>Complexity Analysis</h3>

                <p>Measure cyclomatic complexity and code maintainability</p>

                <button class="learn-more" data-section="complexity">Learn More</button>

            </div>

        </div>

        <div class="info-sections">

            <section id="basic-metrics" class="info-section">

                <h2>Basic Metrics</h2>

                <ul>

                    <li>Total Lines of Code</li>

                    <li>Empty Lines</li>

                    <li>Comment Lines</li>

                    <li>Import Statements</li>

                </ul>

                <div class="example-code">

                    <pre><code>

        # Import statement

        import os

        import sys

        # This is a comment line

        def hello():

            print("Hello, World!")

        # Another comment

        hello()

                        </code></pre>

                </div>

            </section>

            <section id="code-structure" class="info-section">

                <h2>Code Structure Analysis</h2>

                <ul>

                    <li>Class Definitions</li>

                    <li>Function Declarations</li>

                    <li>Method Implementations</li>

                    <li>Object Instantiations</li>

                </ul>

                <div class="example-code">

                    <pre><code>

            # Class Definition

            class Animal:

            # Method Implementation

                def \_\_init\_\_(self, name):

                    self.name = name

            # Method Implementation

                def speak(self):

                    return "Some sound"

            # Function Declaration

            def greet():

                print("Hello!")

            dog = Animal("Buddy")  # Object Instantiation

            print(dog.speak())  # Calling a method

            greet()  # Calling a function

                    </code></pre>

                </div>

            </section>

            <section id="complexity" class="info-section">

                <h2>Complexity Analysis</h2>

                <ul>

                    <li>Cyclomatic Complexity</li>

                    <li>Maximum Nesting Depth</li>

                    <li>Maintainability Index</li>

                </ul>

                <div class="example-code">

                    <pre><code>def complex\_function(x):

    if x > 0:

        for i in range(x):

            if i % 2 == 0:

                print("Even")

            else:

                print("Odd")</code></pre>

                </div>

            </section>

        </div>

        <div class="cta-section">

            <h2>Ready to Analyze Your Code?</h2>

            <p>Get started with our powerful code analysis tool</p>

            <a href="/" class="cta-button">Start Analyzing</a>

        </div>

    </div>

    <footer>

        <div class="footer-content">

            <p>&copy; 2024 Code Analysis Tool. All rights reserved.</p>

        </div>

    </footer>

    <script src="{{ url\_for('static', filename='getstarted.js') }}"></script>

</body>

</html>

### getstarted.css

:root {

    --primary-color: #2196F3;

    --secondary-color: #1976D2;

    --text-color: #333;

    --bg-color: #f4f4f4;

    --card-bg: #ffffff;

}

body {

    margin: 0;

    padding: 0;

    font-family: 'Arial', sans-serif;

    line-height: 1.6;

    color: var(--text-color);

    background-color: var(--bg-color);

}

.navbar {

    background-color: var(--card-bg);

    box-shadow: 0 2px 4px rgba(0,0,0,0.1);

    position: fixed;

    width: 100%;

    top: 0;

    z-index: 1000;

}

.nav-container {

    max-width: 1200px;

    margin: 0 auto;

    padding: 1rem;

    display: flex;

    justify-content: space-between;

    align-items: center;

}

.step-card {

    background: white;

    border-radius: 8px;

    padding: 2rem;

    margin: 2rem 0;

    box-shadow: 0 2px 8px rgba(0,0,0,0.1);

    display: flex;

    gap: 2rem;

    position: relative;

    transition: transform 0.3s ease;

}

.step-card:hover {

    transform: translateY(-5px);

}

.step-number {

    background: #2196F3;

    color: white;

    width: 40px;

    height: 40px;

    border-radius: 50%;

    display: flex;

    align-items: center;

    justify-content: center;

    font-size: 1.5rem;

    font-weight: bold;

}

.step-content {

    flex: 1;

}

.step-content h3 {

    color: #2196F3;

    margin-bottom: 1rem;

}

.step-content ul {

    list-style: none;

    padding: 0;

}

.step-content li {

    margin: 0.5rem 0;

    padding-left: 1.5rem;

    position: relative;

}

.step-content li::before {

    content: "→";

    position: absolute;

    left: 0;

    color: #2196F3;

}

.tip {

    background: #e3f2fd;

    padding: 1rem;

    border-radius: 4px;

    margin-top: 1rem;

    color: #1976D2;

}

.step-image {

    max-width: 100%;

    border-radius: 4px;

    margin-top: 1rem;

}

@media (max-width: 768px) {

    .step-card {

        flex-direction: column;

        gap: 1rem;

        padding: 1.5rem;

    }

    .step-number {

        width: 32px;

        height: 32px;

        font-size: 1.25rem;

    }

}

body.dark-theme .step-card {

    background: #2d2d2d;

}

body.dark-theme .tip {

    background: #1a1a1a;

    color: #64B5F6;

}

body.dark-theme .step-content h3 {

    color: #64B5F6;

}

body.dark-theme .step-content li::before {

    color: #64B5F6;

}

### getstarted.js

document.addEventListener('DOMContentLoaded', () => {

*// Get all learn more buttons*

    const learnMoreButtons = document.querySelectorAll('.learn-more');

*// Get all info sections*

    const infoSections = document.querySelectorAll('.info-section');

*// Add click handlers to buttons*

    learnMoreButtons.forEach(*button* => {

*button*.addEventListener('click', () => {

*// Get the section to show*

            const sectionId = *button*.getAttribute('data-section');

*// Hide all sections first*

            infoSections.forEach(*section* => {

*section*.classList.remove('active');

            });

*// Show the selected section*

            const sectionToShow = document.getElementById(sectionId);

            if (sectionToShow) {

                sectionToShow.classList.add('active');

*// Smooth scroll to the section*

                sectionToShow.scrollIntoView({

                    behavior: 'smooth',

                    block: 'start'

                });

            }

        });

    });

*// Add scroll reveal animation for feature cards*

    const featureCards = document.querySelectorAll('.feature-card');

    const observerOptions = {

        threshold: 0.1,

        rootMargin: '0px 0px -50px 0px'

    };

    const observer = new IntersectionObserver((*entries*) => {

*entries*.forEach(*entry* => {

            if (*entry*.isIntersecting) {

*entry*.target.style.opacity = '1';

*entry*.target.style.transform = 'translateY(0)';

            }

        });

    }, observerOptions);

    featureCards.forEach(*card* => {

*card*.style.opacity = '0';

*card*.style.transform = 'translateY(20px)';

*card*.style.transition = 'all 0.5s ease';

        observer.observe(*card*);

    });

});

**app.py**

*from* flask *import* Flask, request, jsonify, render\_template, session, redirect, url\_for

*import* os

*import* subprocess

*import* re

*import* ast

*from* radon.complexity *import* cc\_visit

*from* radon.metrics *import* mi\_visit

*from* radon.raw *import* analyze

*import* secrets

app = Flask(\_\_name\_\_)

app.secret\_key = secrets.token\_hex(16)  *# Required for session management*

*# Function to analyze code based on file type*

def analyze\_code(*file\_path*, *language*):

*try*:

*# Read file contents first*

*with* open(file\_path, "r", *encoding*="utf-8") *as* f:

            lines = f.readlines()

            content = ''.join(lines)

            total\_lines = len(lines)

*# Common patterns for object-oriented features*

        method\_pattern = r'\s\*def\s+\w+\s\*\(.\*\)'

        object\_pattern = r'\w+\s\*=\s\*\w+\('

*# Set up language-specific patterns and try analysis*

*if* language == "python":

            comment\_symbol = "#"

            class\_pattern = r'class\s+\w+'

            function\_pattern = r'def\s+\w+'

            method\_pattern = r'\s+def\s+\w+\s\*\(.\*\)'

            object\_pattern = r'\w+\s\*=\s\*\w+\('

            import\_pattern = r'^(?:from\s+\w+(?:\.\w+)\*\s+import|\s\*import\s+\w+(?:\s\*,\s\*\w+)\*)'

*try*:

                result = subprocess.run(["pylint", file\_path], *capture\_output*=True, *text*=True)

                analysis\_output = result.stdout or "No issues found"

*except* FileNotFoundError:

                analysis\_output = "Pylint not installed. Please install pylint for detailed Python analysis."

*elif* language == "javascript":

            comment\_symbol = "//"

            class\_pattern = r'class\s+\w+'

            function\_pattern = r'function\s+\w+'

            method\_pattern = r'(async\s+)?[\w.]+\s\*\(.\*\)\s\*{'

            object\_pattern = r'(const|let|var)\s+\w+\s\*=\s\*new\s+\w+'

            import\_pattern = r'^(?:import\s+.\*\s+from\s+[\'"].\*[\'"]|require\s\*\([\'"].\*[\'"]\))'

*try*:

                result = subprocess.run(["jshint", file\_path], *capture\_output*=True, *text*=True)

                analysis\_output = result.stdout or "No issues found"

*except* FileNotFoundError:

                analysis\_output = "JSHint not installed. Please install jshint for detailed JavaScript analysis."

*elif* language == "cpp":

            comment\_symbol = "//"

            class\_pattern = r'class\s+\w+'

            function\_pattern = r'\w+\s+\w+\(.\*\)'

            import\_pattern = r'#include\s\*[<"].\*[>"]'

*try*:

                result = subprocess.run(["cppcheck", file\_path], *capture\_output*=True, *text*=True)

                analysis\_output = result.stdout or "No issues found"

*except* FileNotFoundError:

                analysis\_output = "Cppcheck not installed. Please install cppcheck for detailed C++ analysis."

*elif* language == "java":

            comment\_symbol = "//"

            class\_pattern = r'class\s+\w+'

            function\_pattern = r'\w+\s+\w+\(.\*\)'

            method\_pattern = r'(public|private|protected)?\s+\w+\s+\w+\s\*\(.\*\)'

            object\_pattern = r'\w+\s+\w+\s\*=\s\*new\s+\w+'

            import\_pattern = r'import\s+[\w.]+(?:\s\*\\*)?;'

*try*:

                result = subprocess.run(["javac", file\_path], *capture\_output*=True, *text*=True)

                analysis\_output = result.stdout or "No issues found"

*except* FileNotFoundError:

                analysis\_output = "Java compiler not found. Please install JDK for Java analysis."

*elif* language == "ruby":

            comment\_symbol = "#"

            class\_pattern = r'class\s+\w+'

            function\_pattern = r'def\s+\w+'

            import\_pattern = r'^(?:require|require\_relative|load)\s+[\'"].\*[\'"]'

*try*:

                result = subprocess.run(["ruby", "-wc", file\_path], *capture\_output*=True, *text*=True)

                analysis\_output = result.stdout or "No issues found"

*except* FileNotFoundError:

                analysis\_output = "Ruby not installed. Please install Ruby for detailed analysis."

*elif* language == "go":

            comment\_symbol = "//"

            class\_pattern = r'type\s+\w+\s+struct'

            function\_pattern = r'func\s+\w+'

            import\_pattern = r'import\s+(?:\([^)]+\)|"[^"]+")'

*try*:

                result = subprocess.run(["go", "vet", file\_path], *capture\_output*=True, *text*=True)

                analysis\_output = result.stdout or "No issues found"

*except* FileNotFoundError:

                analysis\_output = "Go not installed. Please install Go for detailed analysis."

*elif* language == "swift":

            comment\_symbol = "//"

            class\_pattern = r'class\s+\w+'

            function\_pattern = r'func\s+\w+'

            import\_pattern = r'import\s+\w+'

*try*:

                result = subprocess.run(["swiftc", file\_path], *capture\_output*=True, *text*=True)

                analysis\_output = result.stdout or "No issues found"

*except* FileNotFoundError:

                analysis\_output = "Swift compiler not found. Please install Swift for detailed analysis."

*elif* language == "php":

            comment\_symbol = "//"

            class\_pattern = r'class\s+\w+'

            function\_pattern = r'function\s+\w+'

            import\_pattern = r'(?:require|include)(?:\_once)?\s\*\([\'"].\*[\'"]\)'

*try*:

                result = subprocess.run(["php", "-l", file\_path], *capture\_output*=True, *text*=True)

                analysis\_output = result.stdout or "No issues found"

*except* FileNotFoundError:

                analysis\_output = "PHP not installed. Please install PHP for detailed analysis."

*elif* language == "csharp":

            comment\_symbol = "//"

            class\_pattern = r'class\s+\w+'

            function\_pattern = r'\w+\s+\w+\(.\*\)'

            import\_pattern = r'using\s+[\w.]+;'

*try*:

                result = subprocess.run(["csc", file\_path], *capture\_output*=True, *text*=True)

                analysis\_output = result.stdout or "No issues found"

*except* FileNotFoundError:

                analysis\_output = "C# compiler not found. Please install .NET SDK for C# analysis."

*else*:

*return* {"error": "Language not supported yet"}

*# Enhanced Analysis*

        empty\_lines = sum(1 *for* line *in* lines *if* line.strip() == "")

        comment\_lines = sum(1 *for* line *in* lines *if* line.strip().startswith(comment\_symbol))

        class\_count = sum(1 *for* line *in* lines *if* re.search(class\_pattern, line))

        function\_count = sum(1 *for* line *in* lines *if* re.search(function\_pattern, line))

        method\_count = sum(1 *for* line *in* lines *if* re.search(method\_pattern, line))

        object\_count = sum(1 *for* line *in* lines *if* re.search(object\_pattern, line))

        import\_count = sum(1 *for* line *in* lines *if* re.search(import\_pattern, line))

*# Find methods, classes, functions and comments*

        methods = []

        objects = []

        classes = []

        functions = []

        comments = []

        imports = []

*for* line *in* lines:

*# Extract method names*

            method\_match = re.search(method\_pattern, line)

*if* method\_match:

                method\_name = method\_match.group().strip()

                methods.append(method\_name)

*# Extract class names*

            class\_match = re.search(class\_pattern, line)

*if* class\_match:

                class\_name = class\_match.group().strip()

                classes.append(class\_name)

*# Extract function names*

            function\_match = re.search(function\_pattern, line)

*if* function\_match:

                function\_name = function\_match.group().strip()

                functions.append(function\_name)

*# Extract comments*

*if* line.strip().startswith(comment\_symbol):

                comment\_text = line.strip()

                comments.append(comment\_text)

*# Extract object creation*

            object\_match = re.search(object\_pattern, line)

*if* object\_match:

                object\_name = object\_match.group().strip()

                objects.append(object\_name)

*# Extract imports*

            import\_match = re.search(import\_pattern, line)

*if* import\_match:

                import\_statement = import\_match.group().strip()

                imports.append(import\_statement)

*# Add complexity analysis*

        complexity\_metrics = analyze\_code\_complexity(content, language)

*# Add complexity metrics to the result*

        result = {

            "output": analysis\_output,

            "total\_lines": total\_lines,

            "empty\_lines": empty\_lines,

            "comment\_lines": comment\_lines,

            "class\_count": class\_count,

            "function\_count": function\_count,

            "method\_count": method\_count,

            "object\_count": object\_count,

            "import\_count": import\_count,

            "methods": methods,

            "objects": objects,

            "classes": classes,

            "functions": functions,

            "comments": comments,

            "imports": imports,

            "code\_structure": {

                "classes": class\_count,

                "functions": function\_count,

                "methods": method\_count,

                "objects": object\_count,

                "imports": import\_count

            },

            "complexity\_analysis": {

                "cyclomatic\_complexity": complexity\_metrics['cyclomatic\_complexity'],

                "max\_nesting\_depth": complexity\_metrics['max\_nesting\_depth'],

                "maintainability\_index": complexity\_metrics['maintainability\_index'],

            }

        }

*return* result

*except* Exception *as* e:

*return* {"error": str(e)}

*# Add this function to validate file extensions*

def allowed\_file(*filename*, *language*):

    extensions = {

        'python': ['.py'],

        'javascript': ['.js'],

        'cpp': ['.cpp', '.hpp', '.h'],

        'java': ['.java'],

        'ruby': ['.rb'],

        'go': ['.go'],

        'swift': ['.swift'],

        'php': ['.php'],

        'csharp': ['.cs']

    }

*return* '.' in filename and \

           filename.rsplit('.', 1)[1].lower() in [ext[1:] *for* ext *in* extensions.get(language, [])]

*# Add this function for auto-detecting language*

def detect\_language(*filename*):

    extension\_map = {

        '.py': 'python',

        '.js': 'javascript',

        '.cpp': 'cpp',

        '.hpp': 'cpp',

        '.h': 'cpp',

        '.java': 'java',

        '.rb': 'ruby',

        '.go': 'go',

        '.swift': 'swift',

        '.php': 'php',

        '.cs': 'csharp'

    }

*if* '.' in filename:

        ext = '.' + filename.rsplit('.', 1)[1].lower()

*return* extension\_map.get(ext)

*return* None

@app.route("/analyze", *methods*=["POST"])

def analyze():

*if* "file" not in request.files:

*return* jsonify({"error": "No file uploaded"})

    file = request.files["file"]

*if* file.filename == '':

*return* jsonify({"error": "No file selected"})

*# Get language from form or auto-detect from file*

    language = request.form.get("language")

*if* not language or language == "auto":

        detected\_language = detect\_language(file.filename)

*if* not detected\_language:

*return* jsonify({"error": "Could not detect language from file extension"})

        language = detected\_language

*if* not allowed\_file(file.filename, language):

*return* jsonify({"error": f"Invalid file type for {language}"})

*try*:

*# Create uploads directory if it doesn't exist*

        os.makedirs("uploads", *exist\_ok*=True)

        file\_path = os.path.join("uploads", file.filename)

        file.save(file\_path)

        result = analyze\_code(file\_path, language)

        os.remove(file\_path)  *# Cleanup after analysis*

*return* jsonify(result)

*except* Exception *as* e:

*return* jsonify({"error": f"Analysis failed: {str(e)}"})

@app.route('/')

def index():

    """Route handler for the home page"""

*# Check if this is the user's first visit*

*if* not session.get('has\_visited'):

*# First visit - mark as visited and redirect to getstarted*

        session['has\_visited'] = True

*return* redirect(url\_for('getstarted'))

*# Returning visitor - show the main page*

*return* render\_template('index.html')

@app.route('/getstarted')

def getstarted():

    """Route handler for the get started page"""

*return* render\_template('getstarted.html')

@app.route('/reset-session')

def reset\_session():

    """Helper route to reset the session (for testing)"""

    session.clear()

*return* redirect(url\_for('index'))

def analyze\_code\_complexity(*content*, *language*):

    """Analyze code complexity metrics"""

*try*:

*# Basic complexity metrics*

        complexity\_metrics = {

            'cyclomatic\_complexity': 0,

            'max\_nesting\_depth': 0,

            'maintainability\_index': 0,

        }

*if* language == "python":

*# Use radon for Python complexity analysis*

            results = cc\_visit(content)

            complexity\_metrics['cyclomatic\_complexity'] = sum(result.complexity *for* result *in* results)

            complexity\_metrics['maintainability\_index'] = mi\_visit(content, *multi*=True)

*# Calculate max nesting depth*

            tree = ast.parse(content)

            complexity\_metrics['max\_nesting\_depth'] = get\_max\_nesting\_depth(tree)

*else*:

*# Basic complexity analysis for other languages*

            lines = content.split('\n')

            current\_depth = 0

            max\_depth = 0

*# Language-specific patterns for nesting detection*

            nesting\_patterns = {

                'javascript': ['{', '}'],

                'java': ['{', '}'],

                'cpp': ['{', '}'],

                'csharp': ['{', '}'],

                'php': ['{', '}'],

                'ruby': ['do|{|begin|if|unless|case', 'end|},'],

                'swift': ['{', '}'],

                'go': ['{', '}'],

            }

            open\_pattern, close\_pattern = nesting\_patterns.get(language, ['{', '}'])

*for* line *in* lines:

                line = line.strip()

*if* re.search(f'.\*{open\_pattern}.\*', line):

                    current\_depth += 1

                    max\_depth = max(max\_depth, current\_depth)

*if* re.search(f'.\*{close\_pattern}.\*', line):

                    current\_depth = max(0, current\_depth - 1)

            complexity\_metrics['max\_nesting\_depth'] = max\_depth

*# Calculate basic cyclomatic complexity*

            decision\_patterns = [

                r'\bif\b', r'\bwhile\b', r'\bfor\b', r'\bforeach\b',

                r'\bcase\b', r'\bcatch\b', r'\b\|\|\b', r'\b&&\b'

            ]

            complexity = 1  *# Base complexity*

*for* pattern *in* decision\_patterns:

                complexity += len(re.findall(pattern, content))

            complexity\_metrics['cyclomatic\_complexity'] = complexity

*# Calculate difficulty score (Halstead difficulty)*

            operators = len(re.findall(r'[+\-\*/=<>!&|^~%]|\b(if|else|while|for|return)\b', content))

            operands = len(re.findall(r'\b[a-zA-Z\_]\w\*\b', content))

            unique\_operators = len(set(re.findall(r'[+\-\*/=<>!&|^~%]|\b(if|else|while|for|return)\b', content)))

            unique\_operands = len(set(re.findall(r'\b[a-zA-Z\_]\w\*\b', content)))

*if* unique\_operands != 0:

                difficulty = (unique\_operators / 2) \* (operands / unique\_operands)

                complexity\_metrics['difficulty\_score'] = round(difficulty, 2)

*return* complexity\_metrics

*except* Exception *as* e:

*return* {

            'cyclomatic\_complexity': 0,

            'max\_nesting\_depth': 0,

            'maintainability\_index': 0,

            'error': str(e)

        }

def get\_max\_nesting\_depth(*node*, *current\_depth*=0):

    """Calculate maximum nesting depth for Python AST"""

    max\_depth = current\_depth

*for* child *in* ast.iter\_child\_nodes(node):

*if* isinstance(child, (ast.If, ast.For, ast.While, ast.Try, ast.With)):

            child\_depth = get\_max\_nesting\_depth(child, current\_depth + 1)

            max\_depth = max(max\_depth, child\_depth)

*else*:

            child\_depth = get\_max\_nesting\_depth(child, current\_depth)

            max\_depth = max(max\_depth, child\_depth)

*return* max\_depth

*if* \_\_name\_\_ == "\_\_main\_\_":

    os.makedirs("uploads", *exist\_ok*=True)

    app.run(*debug*=True)

### requirements.txt

Flask==3.0.0

radon==6.0.1

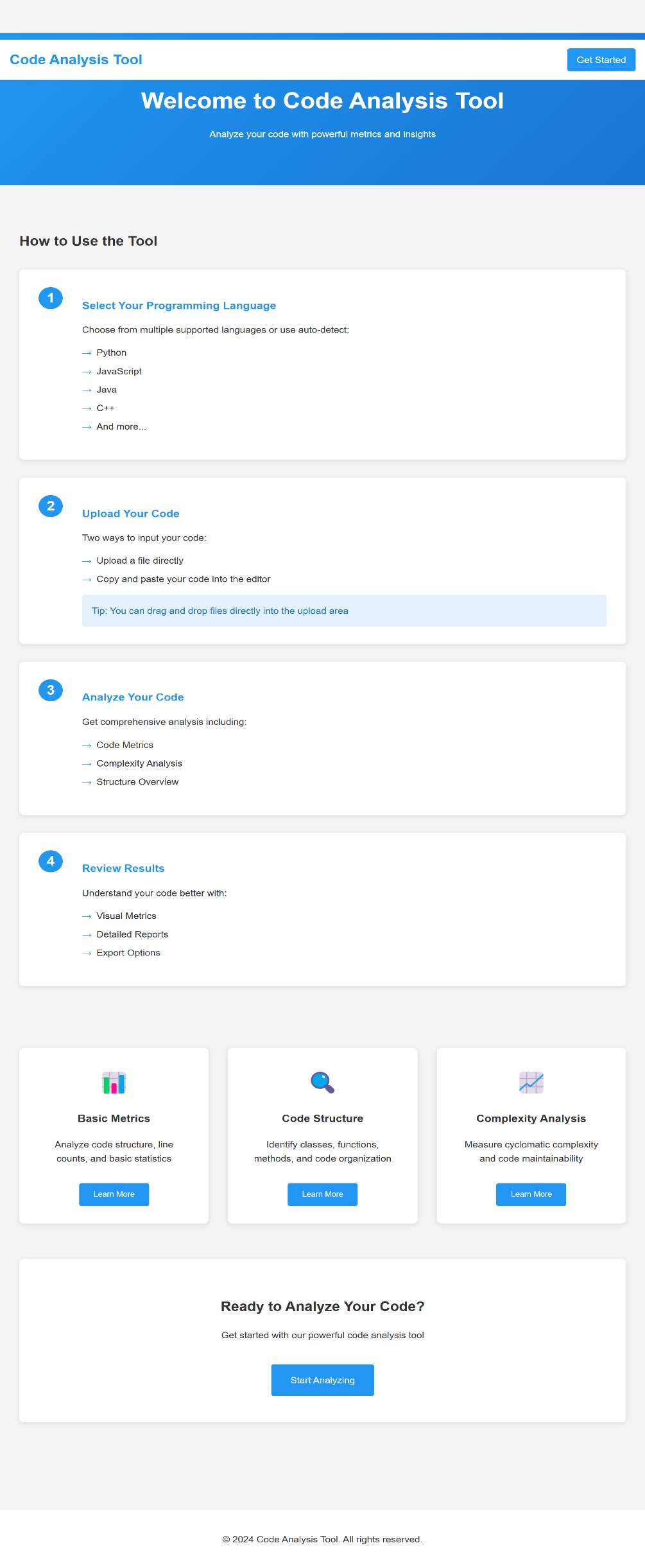
pylint==3.0.3

Werkzeug==3.0.1

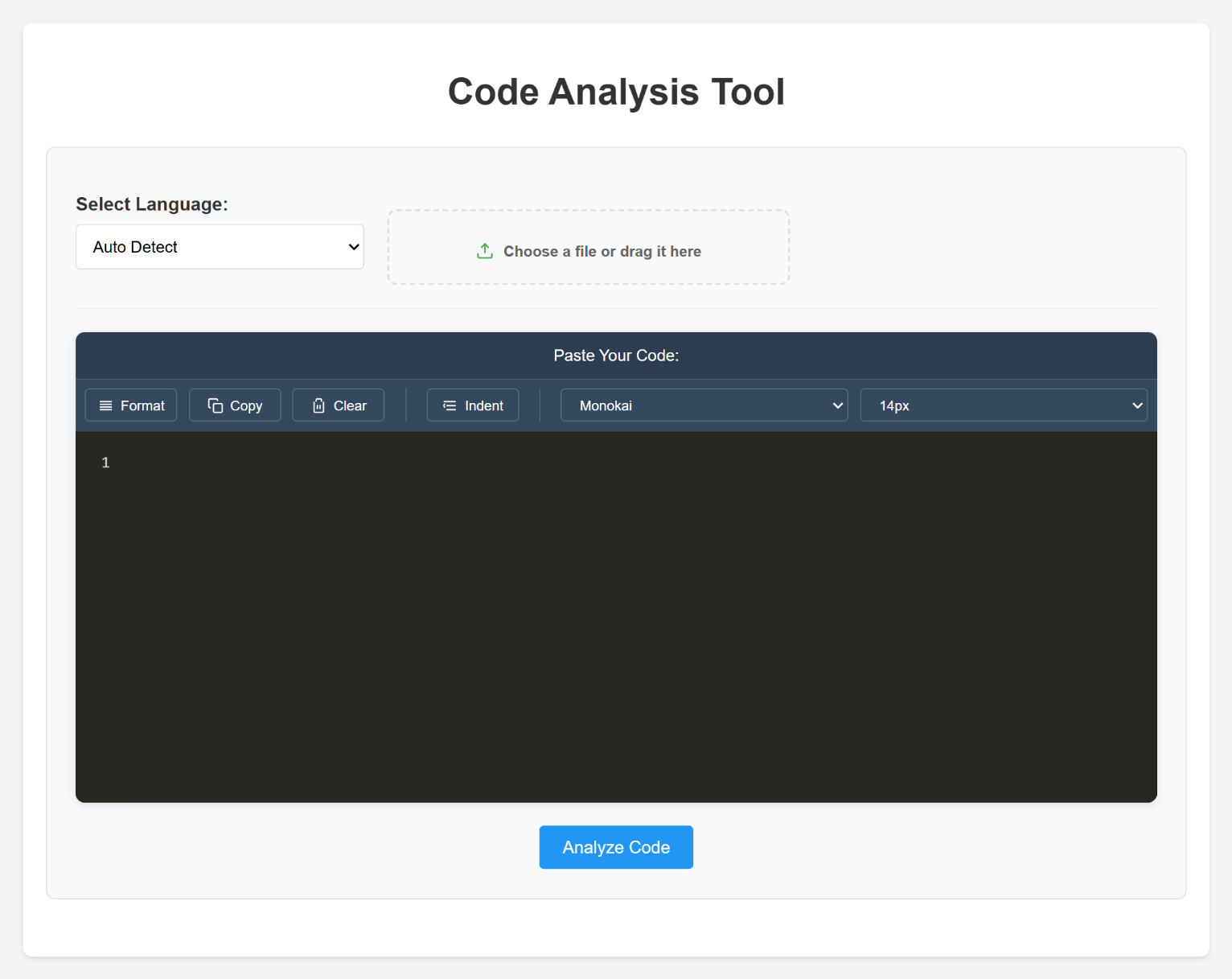
jinja2==3.1.2

* 1. **SCREENSHOTS**

**OUTPUT : Get-started page**

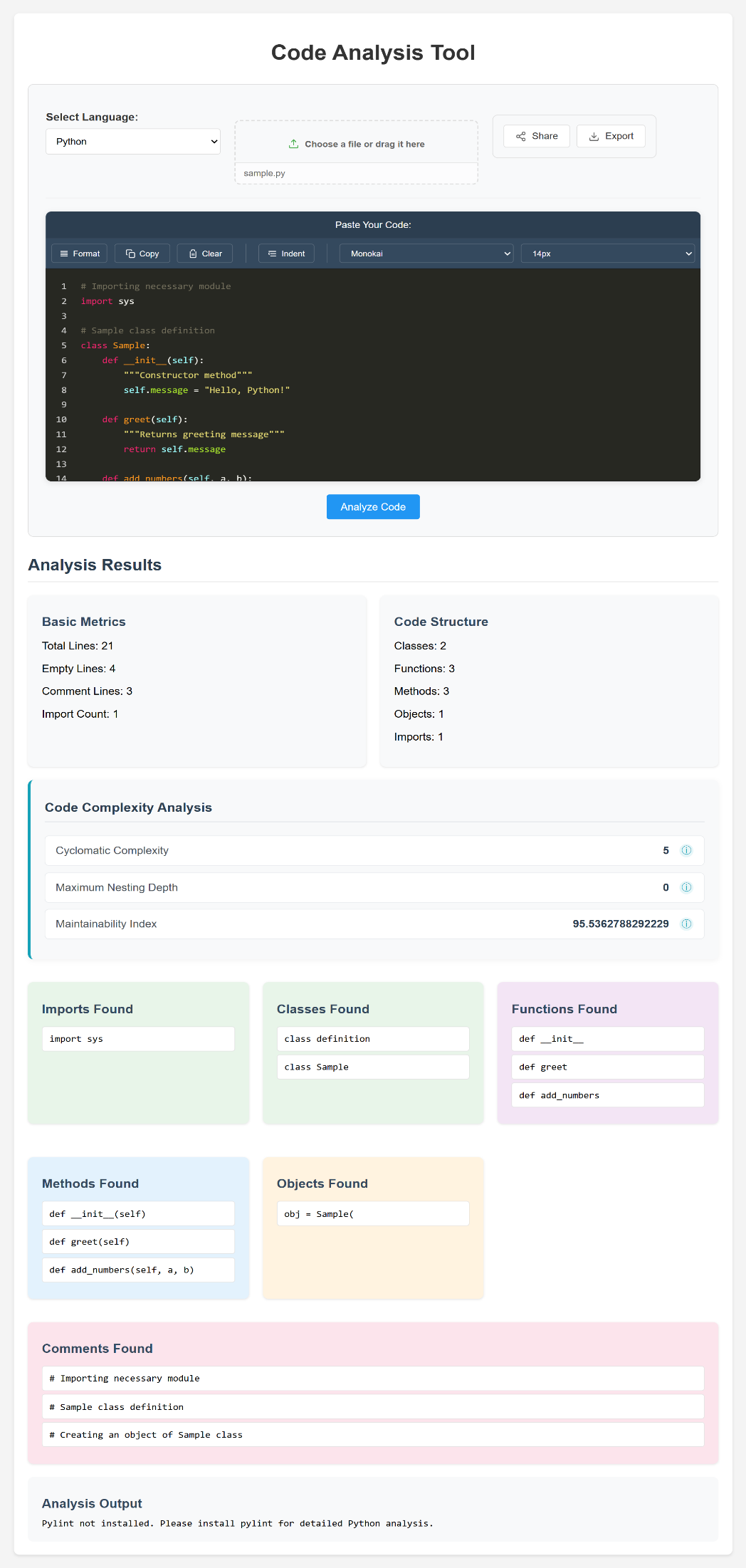


**OUTPUT : index page**

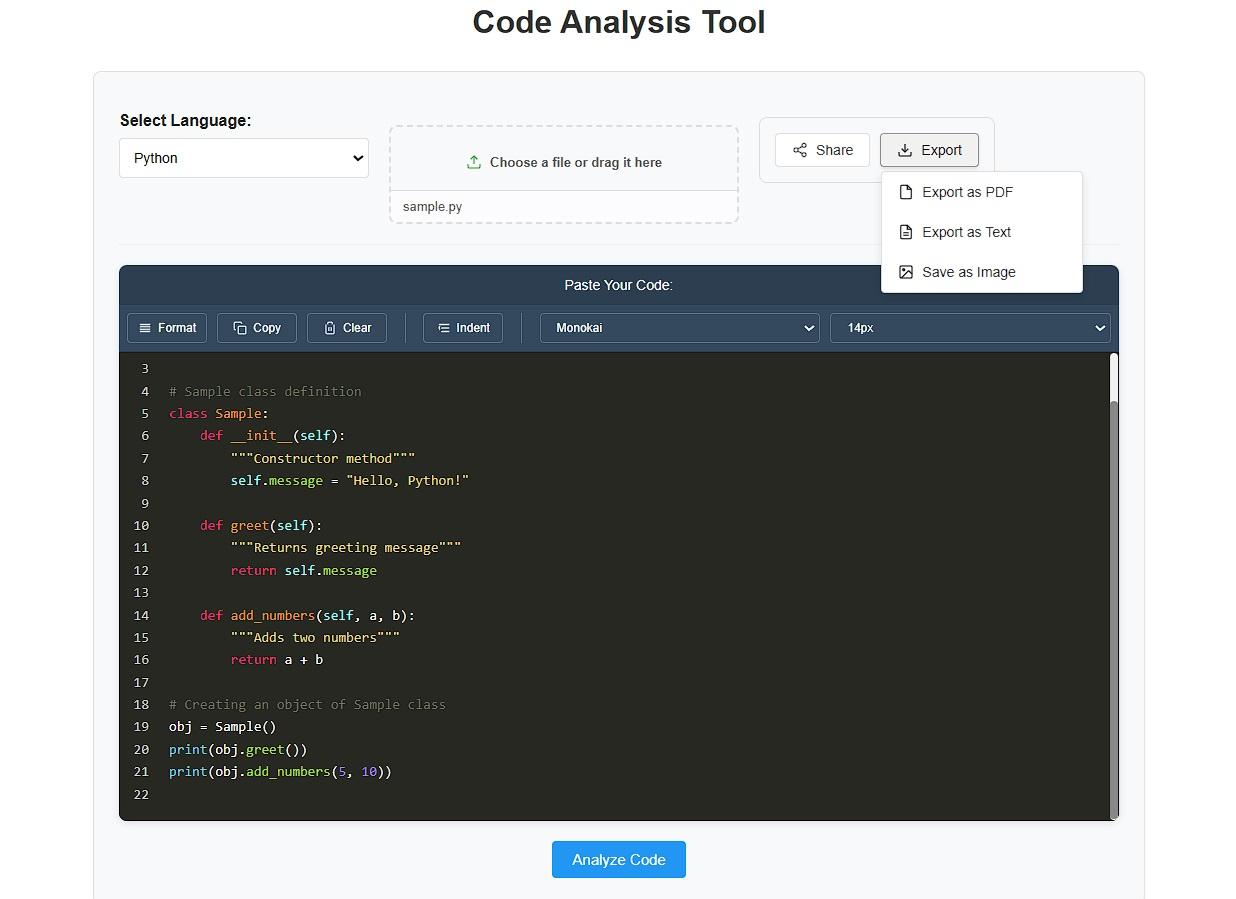
****

9

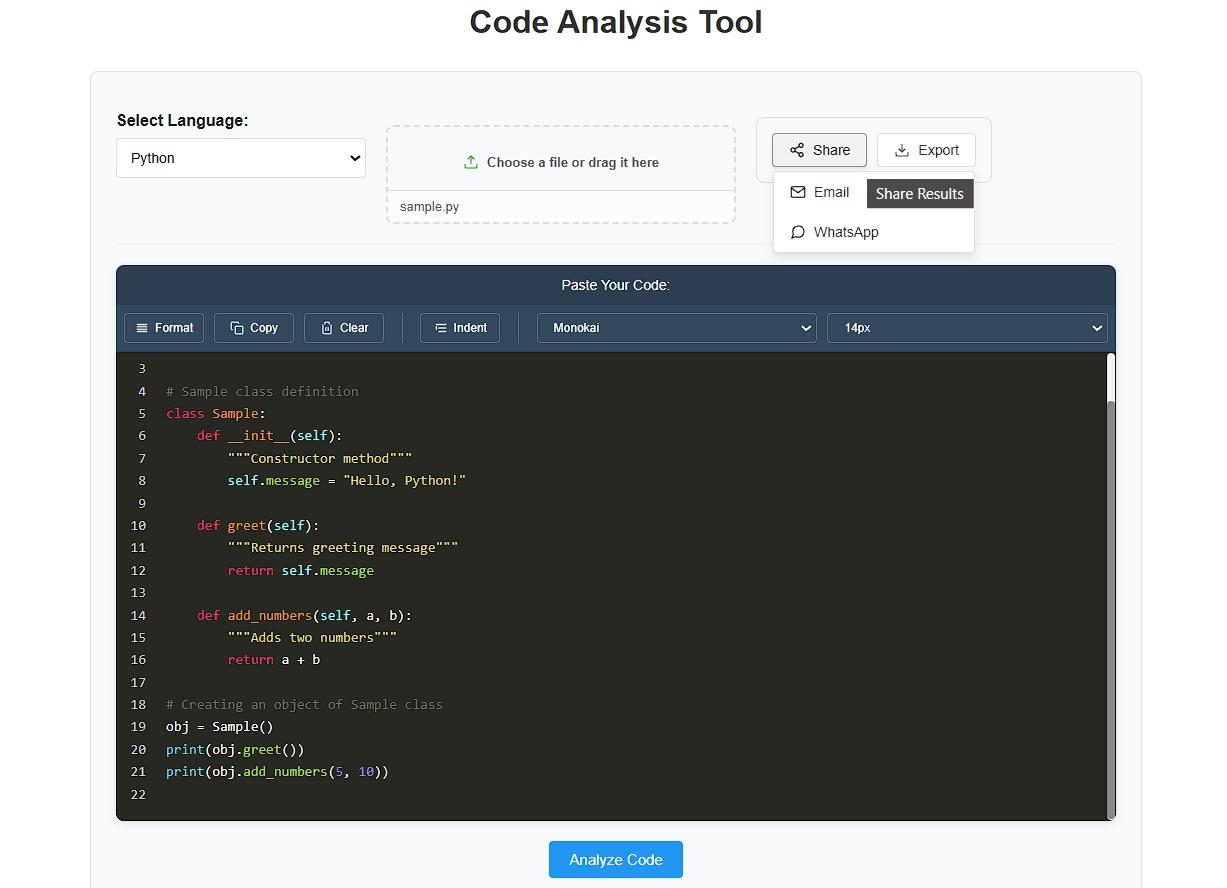
**OUTPUT : result**



**OUTPUT : Export**

****

**OUTPUT : share**



BOTTOMLINE

**8: BOTTOMLINE**

This chapter provides a final summary of the **Script Analysis Web App**, highlighting key takeaways and future directions for improvement.

**8.1 Conclusion :**

The **Script Analysis Web App** successfully provides a comprehensive tool for analyzing source code across multiple programming languages. By offering features such as **code metrics, complexity analysis, structure breakdown, syntax highlighting, and real-time error detection**, the application helps developers better understand their code and improve maintainability.

The integration of **Flask, Radon, Pylint, and Jinja2** enables accurate and efficient analysis while maintaining a lightweight, user-friendly interface. Rigorous testing ensured that the system is stable, responsive, and capable of handling various script sizes.

This project contributes to **better software quality, improved coding standards, and enhanced debugging processes**, making it a valuable tool for developers, educators, and software engineers.

**8.2 Future Enhancements :**

To further improve the **Script Analysis Web App**, the following enhancements are planned:

**Expanded Language Support**

* Extending support for additional programming languages such as **Kotlin, TypeScript, Rust, and R**.
* Enhancing parsing capabilities for complex scripts with multiple interdependent files.

**Advanced Code Analysis**

* Implementing **AI-based code quality suggestions** to improve maintainability.
* Adding **security vulnerability detection** to identify potential security risks.
* Providing **code duplication detection** to help maintain cleaner codebases.

**User Experience Improvements**

* Introducing **dark mode and customizable themes** for better readability.
* Allowing **real-time collaborative code analysis**, where multiple users can review a script together.
* Adding an **export feature for reports** in different formats (PDF, JSON, Excel).

# TESTING

## 9. TESTING

Testing is an essential phase in the development of the **Script Analysis Web App**, ensuring that the system functions correctly, delivers accurate analysis, and performs efficiently. The testing process covers unit testing, integration testing, functional testing and performance testing.

### 9.1 Testing Approach

### Unit Testing

* Tested individual components such as **code metrics analysis, complexity calculation,**  using Python
* Verified that the *radon* and *Pylint* packages provided correct outputs for different script inputs.

### Integration Testing

* Ensured seamless communication between the **frontend (HTML, CSS, JavaScript)** and **backend (Flask, Jinja2, Werkzeug)**.
* Validated that script files uploaded via the web interface were correctly processed and return results.

### Functional Testing

* Tested core functionalities, including:
  + Script upload and validation.
  + Complexity and maintainability analysis.
  + Code structure breakdown (functions, classes, method detection & etc ).
  + Syntax highlighting and report generation.
* Verified system behavior with valid and invalid inputs (e.g., empty files, unsupported file formats).

### Performance Testing

* Evaluated system responsiveness by testing with **small (1 lines), medium (500 lines), and large (2000+ lines) scripts**.
* Measured processing time and ensured performance optimizations were in place for handling larger scripts efficiently.

**9.2 Issues Identified & Fixes Implemented**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **Issue** | **Identified Problem** | **Solution Implemented** | | **Slow processing of large scripts** | Execution delays when analyzing large code files | Optimized Radon and Pylint calls, improved Flask request handling | | **Application crash on invalid file uploads** | Unexpected termination when unsupported formats were uploaded | Added exception handling for invalid file types | | **Incorrect code metric calculations** | Some scripts returned incorrect function and class counts | Improved parsing logic for better multi-language support | | **Unresponsive UI for large scripts** | Page appeared frozen while processing large files | Implemented a loading indicator for better user experience | |

**9.3 How Complexity Analyse is Calculated:**

**1. Cyclomatic Complexity (CC):**

Cyclomatic Complexity (CC) measures the number of independent paths in a program. It helps determine how complex a function is, indicating how many different ways the code can execute.

The formula for Cyclomatic Complexity (CC) is:

|  |
| --- |
| **CC=E−N+2P** |

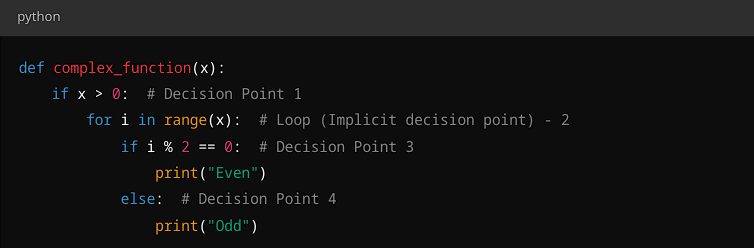
Where:

E = Number of edges (transitions in the control flow graph)

N = Number of nodes (decision points + start/end)

P = Number of connected components (usually 1 for a single function)

Step-by-Step Breakdown of complex\_function(x) :



Decision Points Identified:

1. if x > 0 (1st decision point)

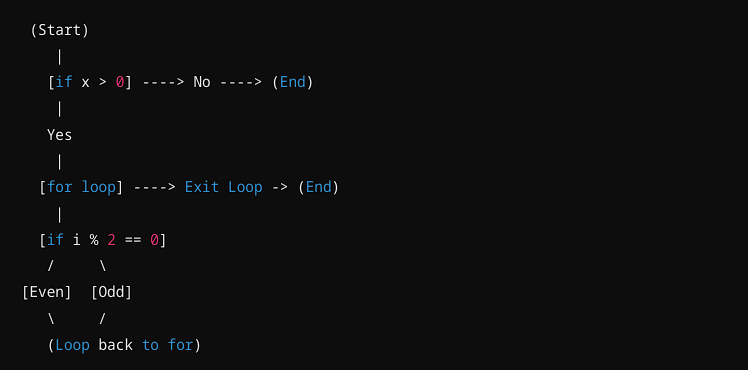
2. for i in range(x) (Loop adds complexity, considered as an implicit branch)

3. if i % 2 == 0 (2nd decision point)

4. else branch (3rd decision point)

Control Flow Graph (CFG) :

A Control Flow Graph (CFG) represents the program’s flow using nodes and edges:



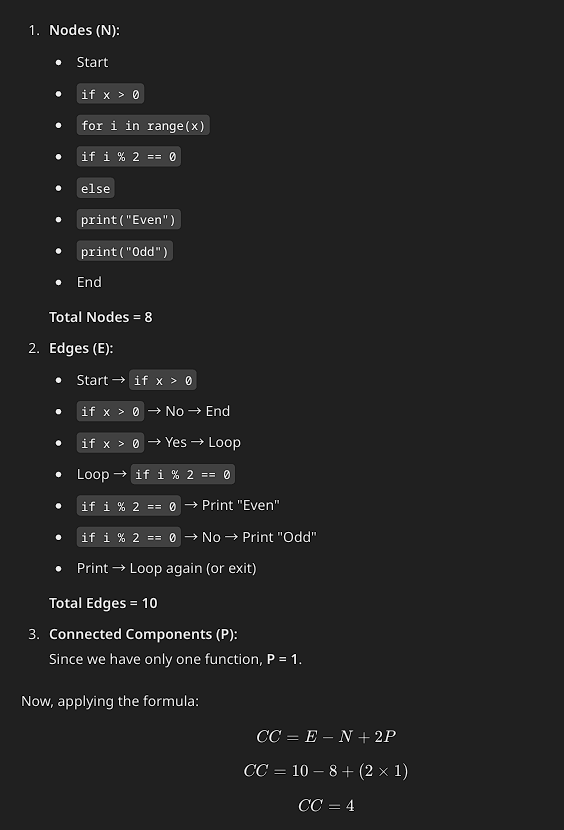
1. if x > 0 (First decision point)

2. for i in range(x) (Loop introduces a decision)

3. if i % 2 == 0 (Another decision point)

4. else branch (Implicitly part of the if, but counted as an additional path)

**Applying the Cyclomatic Complexity Formula :**



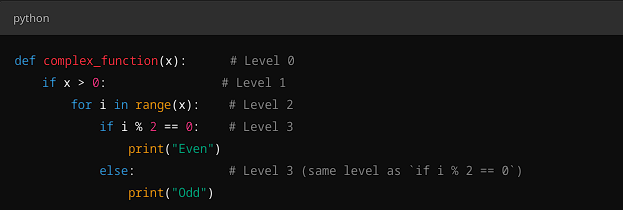
**Final Answer:**

* + - else If your tool counts loops as decision points, then CC = 4.
    - If your tool ignores loop conditions in CC calculation, then CC = 3

**2. Nesting Depth :**

Nesting Depth (MND) measures the deepest level of nested structures (such as loops and conditionals) within a function. It helps assess code complexity by determining how many layers of indentation are needed to follow the logic.

**Step-by-Step Analysis of Nesting Depth :**

****

**How Nesting Depth is Counted :**

* Level 0 → def complex\_function(x): The function itself.
* Level 1 → if x > 0: First conditional statement.
* Level 2 → for i in range(x): Loop inside the if statement.
* Level 3 → if i % 2 == 0: Condition inside the loop.
* Level 3 → else: At the same level as if i % 2 == 0, so it does not increase depth further

Why Not 4?

* Some might mistakenly count else as an additional level, but it is at the same level as if (i % 2 == 0), so it does not contribute to depth.
* No additional nested condition inside if (i % 2 == 0) or else , so depth does not increase further

# REFERENCES

## 10. REFERENCES

This chapter provides references used in the development, research, and implementation of the **Script Analysis Web App**. The sources include journals, books, and web-based resources that contributed to understanding programming analysis, complexity measurement, and software architecture.

**10.1 Journal References**

* McCabe, T. J. (1976). *A Complexity Measure*. IEEE Transactions on Software Engineering, SE-2(4), 308-320.
* Chidamber, S. R., & Kemerer, C. F. (1994). *A Metrics Suite for Object-Oriented Design*. IEEE Transactions on Software Engineering, 20(6), 476-493.
* Halstead, M. H. (1977). *Elements of Software Science*. Elsevier.

**10.2 Book References**

* Sommerville, I. (2015). *Software Engineering (10th Edition)*. Pearson Education.
* Pressman, R. S. (2019). *Software Engineering: A Practitioner's Approach (9th Edition)*. McGraw-Hill.
* Martin, R. C. (2009). *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall.

**10.3 Web References**

* Flask Framework Documentation: https://flask.palletsprojects.com/
* Radon Package Documentation: <https://radon.readthedocs.io/>
* Pylint Documentation: https://pylint.pycqa.org/
* Jinja2 Documentation: https://jinja.palletsprojects.com/
* Cyclomatic Complexity: https://www.geeksforgeeks.org/cyclomatic-complexity/