**Script Analyzer for ALLOT**

**Overview**

The Script Analyzer is a sophisticated tool engineered to scrutinize C++ scripts for various parameters such as indentation, naming conventions, modularization, consistency, and excess whitespace. It generates an exhaustive analysis report and has the capability to dispatch this report via email. The tool is seamlessly integrated with a Flask web application, facilitating effortless file upload and analysis.

**Features**

* **File Upload**: Empowers users to upload C++ scripts for in-depth analysis.
* **Analysis**: The tool meticulously analyzes the uploaded files for indentation, naming conventions, modularization, consistency, and excess whitespace.
* **Email Notification**: Upon completion of the analysis, the tool dispatches an email notification with the comprehensive analysis report to a designated recipient email address.
* **Web Interface**: The Flask web application offers an intuitive interface for uploading files and perusing analysis results.

**How to Use**

* **Uploading Files**: Navigate to the web interface, click on the “Choose File” button, select a C++ script file (.cpp), input a recipient email address, and click the “Submit” button to upload the file.
  + Please note that only files with the ".cpp" extension will only be accepted for this version of the Script Analyzer.
* **Analysis**: The tool will conduct a thorough analysis of the uploaded file across various parameters and generate a detailed analysis report with logs and Summary via Email.
* **Email Notification**: Upon completion of the analysis, the tool will dispatch an email notification to the specified recipient email address. The email will encompass the analysis report as an attachment.
  + Please note presently this will only accept thinkpalm email ids as Recipient Email ids in the GUI
  + The email notification includes both the analysis report and the logs.
* **Accessing Analysis Results**: The web interface will exhibit a success message indicating that the file was successfully uploaded Analyzed and Email sent. Users can also peruse the analysis report in the email dispatched to the recipient email address.

**Detailed Analysis by Script Analyzer.py**

* **Line Count Verification:**

Checks the total number of lines in the file to ensure it does not exceed a specified limit (e.g., 1500 lines).

* **Indentation Consistency Inspection:**

Validates if the code adheres to standard conventions for indentation. It ensures that only spaces are used for indentation, and tabs are converted to spaces as a standard coding practice.

* **Naming Standards Assessment:**

Confirms that variables, functions, classes, and other identifiers comply with the specified naming conventions.

* + All symbols should have a prefix identifying their module.
  + Variables and functions should start with a lowercase letter
  + Types and Classes should start with an uppercase letter.
  + Constants should be written in all uppercase letters.
  + Global variables should start with 'g\_'.
  + Members should start with 'm\_'
  + Pointers should start with 'p'.

These conventions ensure consistent and meaningful naming throughout the codebase.

* **Module Structure Evaluation:**

Inspects the code to ascertain it is properly modularized, with a clear demarcation of functionality into modules or classes. It checks for appropriate use of headers and namespaces.

* **Code Uniformity Check:**

Checks for uniformity in coding style and formatting throughout the file. It ensures that the code is written in a consistent manner, enhancing its readability and maintainability.

* **Whitespace Reduction Analysis:**

Identifies and eliminates excess whitespace in the code. It ensures that there are no unnecessary spaces or blank lines, thereby improving code readability.

* **File Format Consistency Verification:**

Ensures that the file format is consistent and adheres to the specified standards.

These analyses help maintain code quality, readability, and consistency, ensuring that the C++ codebase is well-structured and adheres to best practices.

In conclusion, Script\_Analyzer.py offers a comprehensive analysis of C++ script files, focusing on pivotal aspects of code quality and adherence to coding standards. It is an invaluable tool for developers and code reviewers to ensure code quality and compliance with coding standards.

**Limitations:**

* **Single File Upload:** Only one file can be uploaded for analysis at a time.
* **Analysis Time:** The analysis process, including sending out the email with logs, typically takes under a minute for a large file, and even less for smaller files.
* **No File Size Limit:** There is no limit to the size of the file that can be uploaded for analysis.
* **File Extension Requirement:** The uploaded C++ file must have the ".cpp" extension.
* **Fixed Analysis Settings:** Currently, users cannot change the analysis settings. Future updates may allow customization based on project requirements.
* **No Result Display on GUI:** There is no provision to view the analysis results directly on the GUI. This may be considered for future enhancements.
* **Automatic Deletion of Uploaded Files:** Uploaded files and logs are automatically deleted from the server once the email containing the logs is sent to the recipient email mentioned in the GUI.
* **Update Policy:** The tool will be updated based on issues raised by end users and to incorporate new features or improvements.

**Feedback and Support:**

If you encounter any issues, have suggestions, or require assistance, please reach out to me at [manu.m@thinkpalm.com](mailto:manu.m@thinkpalm.com) or via Teams.