# KEYLOGGER WEB APPLICATION USING PYTHON



**PYTHON PROJECT REPORT**

**FACULTY:** Dr. Gunjan Ansari

**SLOT:** B11+E11+B12+E12

**SUBJECT:** PYTHON PROGRAMMING CSE3011

**NAME & REG. NO.:**

SARTHAK JAIN 22BAI10117

VAIBHAV PAREKH 22BAI10137

SAMAKSH JAIN 22BAI10271

**1. Introduction**

In this project, a keylogger is implemented using Python, which logs keyboard inputs and stores them in a text file. The keylogger is then integrated into a web application built with Flask, providing an interface to start and stop the keylogger, view the logs, and clear them.

**2. Objectives**

* **Keylogger Implementation**: Create a Python-based keylogger that records keystrokes.
* **Web Interface**: Develop a Flask-based web interface to control the keylogger.
* **Log Management**: Provide functionality to view and clear logs from the web interface.

**3. Technology Stack**

**3.1 Python**

Python is the primary programming language used in this project. Its simplicity and extensive library support make it ideal for both scripting and web development. Python is used for:

* Keylogger Implementation: Capturing and processing keyboard inputs.
* Flask Web Application: Serving the web interface and handling HTTP requests.
* Threading: Managing concurrency, allowing the keylogger to run in the background while the web server handles incoming requests.

**3.2 pynput Library**

pynput is a Python library that allows you to control and monitor input devices such as the keyboard and mouse. In this project, pynput is used to create a keylogger by capturing keystrokes in real time.

* Keylogger Implementation: The pynput.keyboard.Listener class listens to keyboard events. Each time a key is pressed, the on\_press method is triggered, and the corresponding key is logged.

**3.3 Flask Framework**

Flask is a micro web framework for Python that provides the tools and libraries to build a web application. It is lightweight, making it an excellent choice for small to medium-sized projects where simplicity and flexibility are required.

* Routing: Flask allows the creation of URL routes that map to specific functions. These functions are responsible for rendering templates or handling user input.
* Template Rendering: Flask uses the Jinja2 template engine to generate HTML dynamically. This allows you to pass data from your Python code to the HTML templates, creating dynamic web pages.
* Form Handling: Flask provides easy handling of form submissions using methods like GET and POST.

**3.4 HTML & CSS**

HTML (Hyper Text Markup Language) and CSS (Cascading Style Sheets) are used for building the front-end of the web application.

* HTML: Defines the structure and content of the web pages. The web application consists of multiple HTML files, each responsible for a specific page like the homepage, log view, etc.
* CSS: Controls the presentation and layout of the HTML elements. It is used to style the buttons, containers, and text, ensuring the web interface is user-friendly and visually appealing.

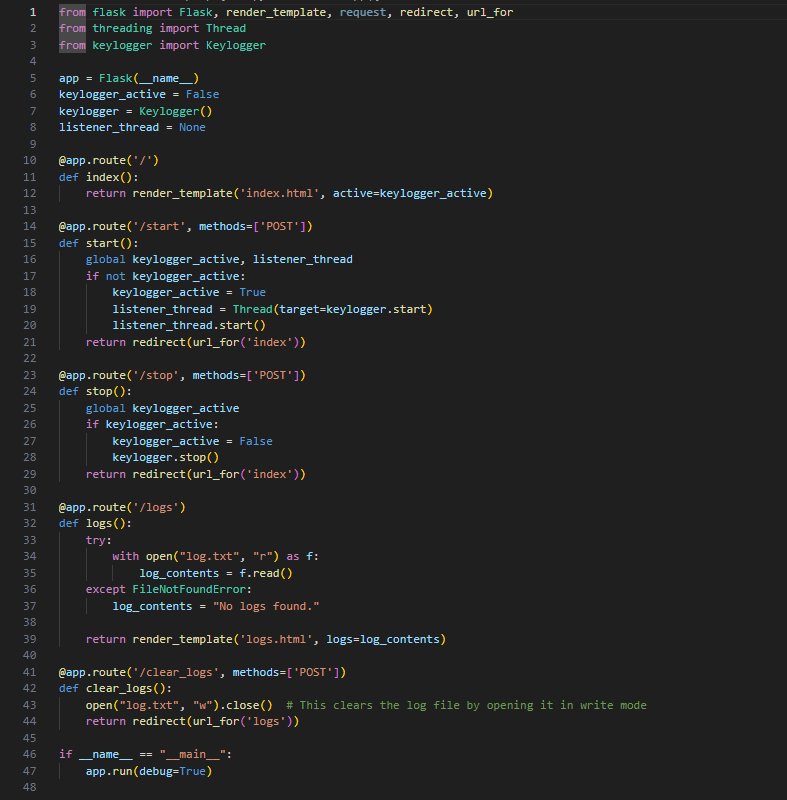
**3.5 Threading in Python**

The threading module in Python is used to manage concurrency, allowing multiple operations to run simultaneously.

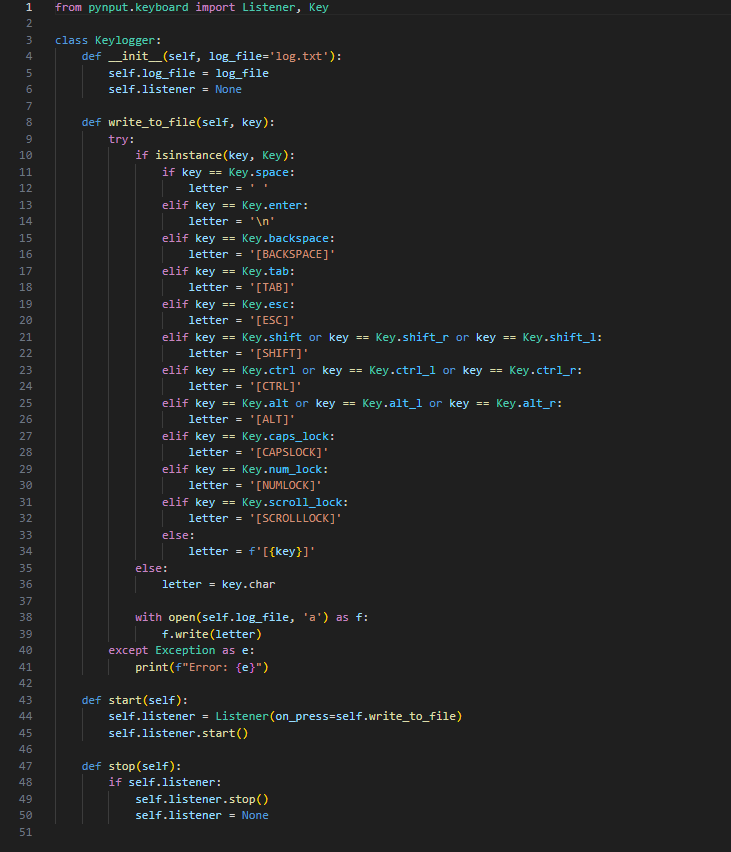
* Keylogger Thread: The keylogger runs in a separate thread to ensure it does not block the main Flask application. This means that while the keylogger is capturing keystrokes, the web application remains responsive and can handle user requests like viewing logs or stopping the keylogger.

The keylogger is implemented using the ***pynput*** library. It captures keystrokes and writes them to a text file (log.txt). The keylogger runs in a separate thread to allow the Flask server to handle requests concurrently. The web application made by FLASK provides a user interface to control the keylogger. It includes routes to start and stop the keylogger, view the logs, and clear the logs. The web interface is styled using CSS. The interface includes buttons for starting and stopping the keylogger and viewing and clearing logs. The buttons are centrally aligned, and the logs are displayed within a scrollable container to ensure the content stays within the screen boundaries.

**FLASK CODE:**



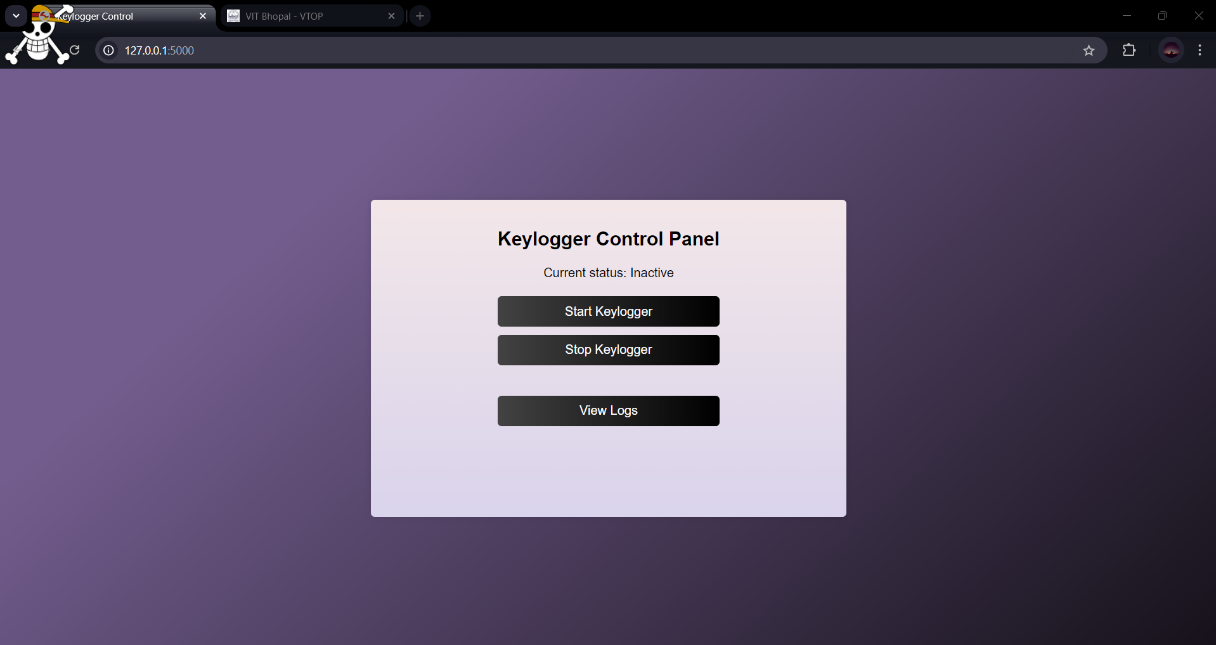
**PYTHON MODULE CODE (CREATION OF KEYLOGGER):**

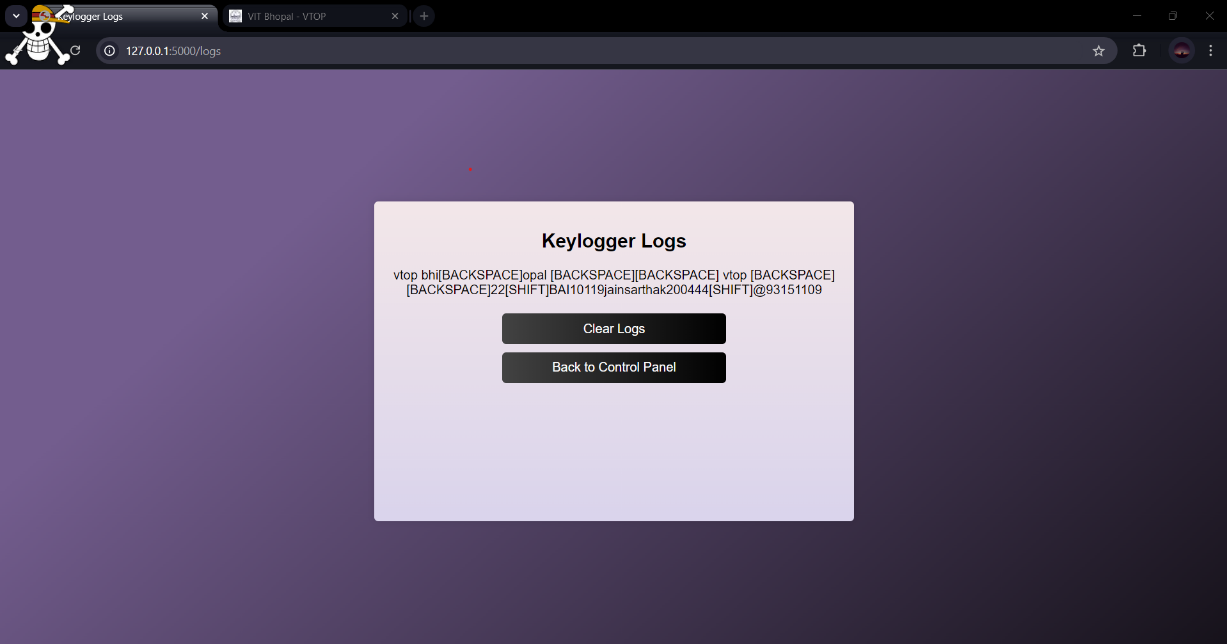


**4. Testing**

Testing is a crucial part of the development process. The application was tested locally in the following scenarios:

* **Start/Stop Keylogger**: The keylogger can be started and stopped via the web interface, and it correctly captures keystrokes when active.
* **View Logs**: The logs are displayed in the web interface, allowing the user to see the captured keystrokes in real time.
* **Clear Logs**: The clear logs function successfully empties the log file, removing all recorded keystrokes.





**5. Challenges and Solutions**

* **Concurrency**: Running the keylogger while handling web requests required careful management of threads. This was resolved using Python's threading module.
* **UI Responsiveness**: Ensuring the web interface remains responsive and accessible, especially with long log files, was handled by using CSS for scrollable containers.

**6.Applications**

* Parents can use a keylogger to monitor their children's computer usage. It helps in ensuring that children are not exposed to inappropriate content online or engaging in unsafe communication.
* Employers can use keyloggers to monitor employee activity on company-owned devices. This can help ensure that company resources are being used appropriately and that employees are not engaging in unauthorized activities.
* Organizations might use keyloggers as part of a broader security auditing process to detect unauthorized access or suspicious activities.
* Keyloggers can be used in research settings to study user behaviour, such as typing patterns, which can be useful in fields like psychology or human-computer interaction.
* In cases where unsaved work is lost due to a software crash, a keylogger might help recover some of the lost information by providing a record of the keystrokes.
* Keyloggers can be used in cybersecurity training to demonstrate the risks associated with unsecured devices and the importance of using secure, encrypted channels.
* Keyloggers can assist in recovering forgotten passwords if the user needs to remember what they typed previously.

**7. Conclusion**

The project successfully demonstrates a combination of Python scripting and web development using Flask. The keylogger works as intended, and the web interface provides a user-friendly way to manage the keylogger and logs. This project can be further extended by adding more features like live updates to logs or more detailed control over keylogging behaviour.