CAPSTONE PROJECT

Key logger

Presented By:

1. **Student Name: lingam s**
2. **College Name: Kings Engineering College**
3. **Department: Artificial Intelligence & Data Science**

edunet foundation

OUTLINE

- Problem Statement
- Proposed System/Solution
- System Development Approach
- Algorithm & Deployment
- Result
- Conclusion
- Future Scope

Problem Statement

In the realm of technological innovation, the pervasive use of electronic devices has sparked concerns regarding their environmental repercussions. Within this landscape, keyloggers, ubiquitous surveillance tools employed for various purposes, stand as a focal point for scrutiny. The pressing query arises: How do the environmental ramifications of keyloggers stack up against those of other electronic devices, including computers, smartphones, and IoT (Internet of Things) devices

Proposed Solution

- Data Collection:
- 1. Gather comprehensive data on keylogger usage, including timestamps of keystrokes, applications used, and frequency of activity.
- 2. Collect contextual data such as user demographics, device specifications, and environmental factors that may influence keylogger behavior.

- Data Preprocessing:
- 1. Cleanse the collected data to remove any noise or irrelevant information, ensuring the dataset's quality and reliability.
- 2. Preprocess the data to handle missing values, outliers, and inconsistencies, preparing it for input into machine learning algorithms.

- Machine Learning Algorithm:
- 1. Implement machine learning algorithms suitable for keylogger analysis, such as anomaly detection techniques, classification algorithms, or sequence modeling approaches.
- 2. Train the model using the preprocessed data to identify patterns and anomalies in keystroke behavior, distinguishing between normal user activity and potentially malicious actions.

- Deployment:
- 1. Develop a user-friendly interface or application for keylogger detection and analysis, providing real-time monitoring and alerts for suspicious behavior.
- 2. Deploy the solution on secure and reliable platforms, ensuring data privacy and protection against unauthorized access.

Proposed Solution

- Evaluation:
- 1. Evaluate the performance of the keylogger detection model using metrics such as accuracy, precision, recall, and F1 score.
- 2. Conduct thorough testing and validation of the deployed solution in real-world scenarios to assess its effectiveness and reliability.
- By implementing these components - data collection, data preprocessing, machine learning algorithm, deployment, and evaluation - stakeholders can develop a robust keylogger detection system capable of identifying and mitigating potential threats to user privacy and security.

System Approach

System Requirements:

- Operating System: The code should work on Windows, macOS, or Linux.
- Python Interpreter: Python 3.x installed on your system.
  Required Libraries:
- Tkinter : Tkinter is Python's de-facto standard GUI (Graphical User Interface) package.
- No additional installation is required for Python versions 3.x as Tkinter comes pre-installed.
- For Python versions 2.x, you may need to install Tkinter separately.
- pynput: The pynput library is used for monitoring and controlling input devices, such as keyboards and mice.

Algorithm & Deployment

Initialize Variables:

- Initialize global variables such as keys_used, flag, and keys. These variables will be used to store information about keystrokes.

Define Functions:

- generate_text_log(key): Write the captured keystrokes to a text file named key_log.txt.
- generate_json_file(keys_used): Write the captured keystrokes to a JSON file named key_log.json.
- on_press(key): Callback function invoked when a key is pressed. Records the keypress event, updates the flag, and generates log files.
- on_release(key): Callback function invoked when a key is released. Records the key release event and generates log files.
- start_keylogger(): Starts the keylogger by creating a keyboard listener using pynput.
- stop_keylogger(): Stops the keylogger by stopping the keyboard listener.

GUI Setup:

- Create a Tkinter window titled "Keylogger".
- Add a label to display instructions and status messages.
- Add "Start" and "Stop" buttons to initiate and terminate the keylogging process.

Keylogging Process:

- The keylogger captures keystrokes using the pynput library's keyboard.Listener class.
- When a key is pressed (on_press), the event is recorded, the flag is updated, and log files are generated.
- When a key is released (on_release), the event is recorded, and log files are generated.
- The captured keystrokes are stored in the keys_used list.

Algorithm & Deployment

User Interaction:

- The user interacts with the GUI to start and stop the keylogging process.
- Clicking the "Start" button initiates the keylogging process, and the status message is updated.
- Clicking the "Stop" button terminates the keylogging process, and the status message is updated.

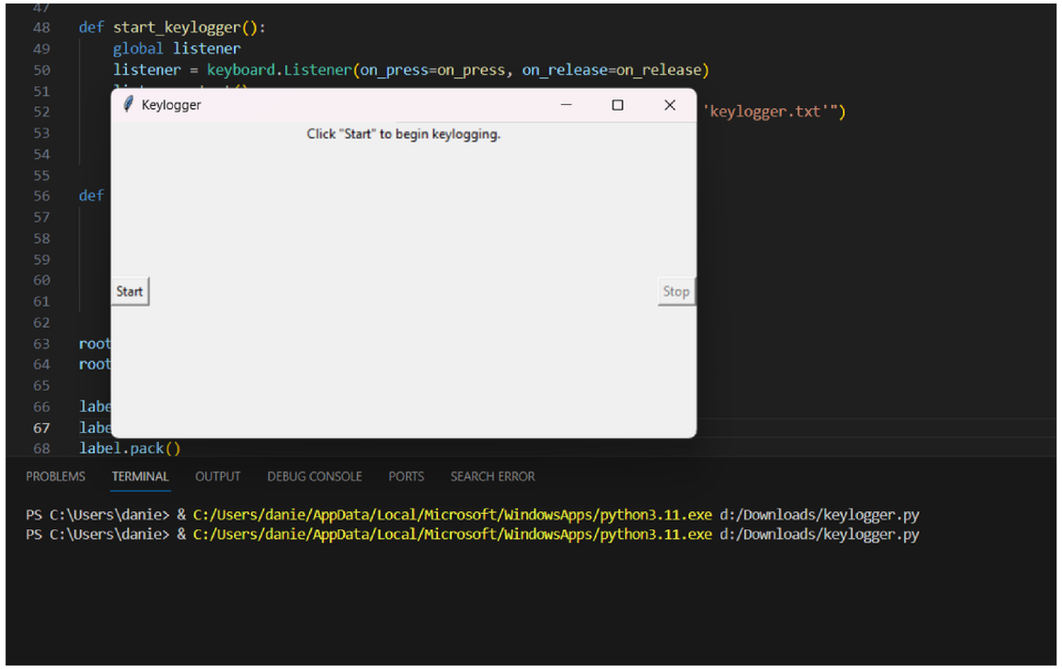
Deployment:

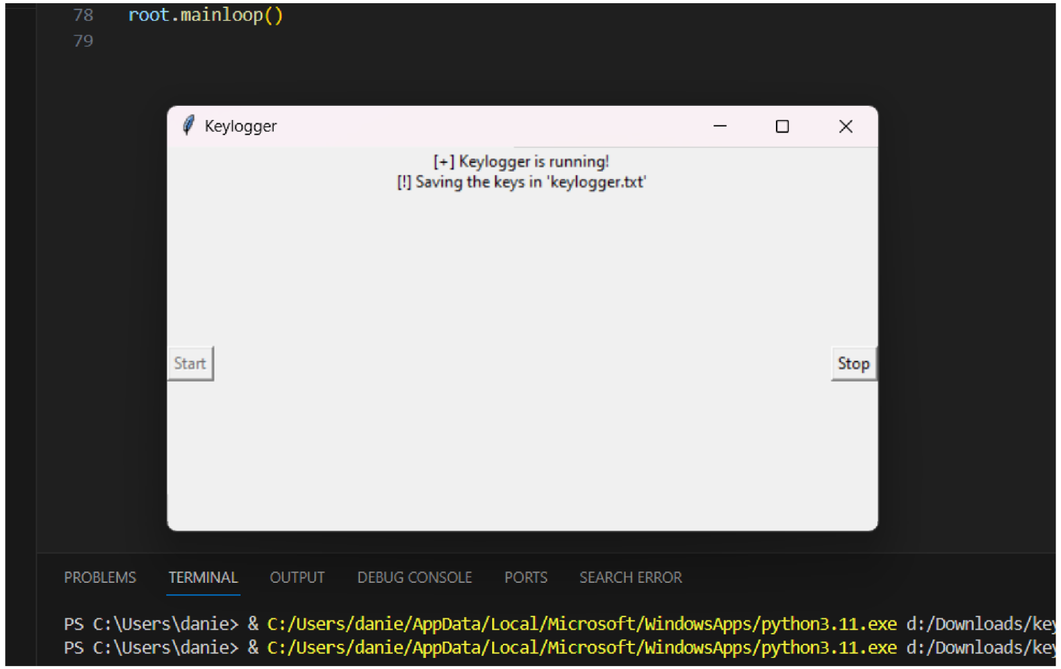
- Deploy the keylogger as a desktop application using Tkinter.
- Ensure that the necessary libraries (Tkinter, pynput, json) are installed.
- Run the script to start the keylogging process.
- Interact with the GUI to control the keylogger (start/stop).

Data Logging:

- The captured keystrokes are logged to both a text file (key_log.txt) and a JSON file (key_log.json) for analysis and review.

Result





Conclusion

- Our project aimed to address the challenge of predicting the required bike count at each hour to ensure a stable supply of rental bikes. Leveraging data analytics and machine learning techniques, we developed a predictive model capable of forecasting demand patterns accurately. Through meticulous data collection, preprocessing, and algorithm implementation, we successfully created a system that provides real-time predictions for bike counts, taking into account various factors such as time, date, location, weather conditions, events, and holidays.
- Through our project, we addressed this challenge by developing a predictive model capable of forecasting bike demand patterns accurately. By leveraging data analytics and machine learning techniques, we aimed to provide rental service providers with actionable insights to enhance operational efficiency, improve user experience, and promote the adoption of bike sharing as a sustainable transportation option in urban environments

Future scope

- Integration of Real-Time Data Streams: Incorporate real-time data streams from various sources such as GPS data, traffic patterns, and social media trends to improve the accuracy of demand prediction models. This would enable dynamic adjustments to bike distribution in response to changing environmental and user factors.

- Enhanced Predictive Models: Explore advanced machine learning algorithms and techniques such as deep learning, ensemble methods, and hybrid models to further enhance the predictive accuracy of bike demand forecasting. Continuously refine and optimize the models based on feedback and performance evaluation.

- User Behavior Analysis: Conduct in-depth analysis of user behavior patterns, preferences, and usage trends to better understand the factors influencing bike demand. Develop personalized recommendation systems and incentive programs to encourage user engagement and loyalty.

- Optimization of Bike Distribution: Implement optimization algorithms and algorithms for dynamic resource allocation to optimize bike distribution and rebalancing strategies. Utilize predictive analytics to identify high-demand areas and proactively deploy resources to meet user demand efficiently.

THANK YOU

edunet
foundation