**Is It My Type?**

פרויקט גמר בסייבר:

נושא: AI

שם המגיש: רועי נחשון

תעודת זהות: 327757977

שמות המנחים: ירון יצחקי, תומר טלגם

תאריך הגשה: ----

# Introduction

## initiation

### Overview

The Typing Pattern Detection AI is an advanced system designed to analyze and recognize the typing patterns of computer users. It captures and saves these patterns, normalizes the data, and trains a model to identify and differentiate between different typists. The primary purpose of this AI is to detect unauthorized users typing on your computer, providing an additional layer of security and privacy.

### Project Rationale

The Typing Pattern Detection AI project was initiated to address the growing concerns related to computer security and unauthorized access. Traditional methods like passwords and biometrics have their limitations, and there was a need for an innovative approach to identify and differentiate users based on their unique typing patterns. By leveraging machine learning and artificial intelligence techniques, this project aims to provide an effective solution to detect potential intruders and protect sensitive information.

### Features and Functionality

The finished product offers the following key features:

1. Typing Pattern Capture: The AI captures and records the typing patterns of users while they interact with the computer.
2. Data Normalization: The captured typing patterns are preprocessed and normalized to remove any biases and ensure consistency in the dataset.
3. Training the Model: The normalized data is then utilized to train a machine learning model capable of recognizing and classifying different typing patterns.
4. User Identification: The trained model can identify and differentiate between authorized users and unauthorized individuals attempting to type on the computer.
5. User Interface (UI): The project includes a user-friendly UI that allows for easy interaction and configuration of the AI system.
6. Training and Testing Files: Separate files are provided for training and testing purposes, enabling the evaluation and refinement of the AI's performance.

### Motivation and Challenges

There are several reasons why this project was chosen, including:

1. Enhanced Security: By implementing a typing pattern detection system, the project aims to strengthen the security measures in place, offering an additional layer of protection against unauthorized access.
2. User-Friendly Approach: Unlike traditional security methods that often require complex passwords or biometric scans, typing pattern detection offers a more natural and user-friendly way to identify individuals.
3. Potential Real-World Applications: The technology developed in this project could find applications in various domains, such as corporate security, computer forensics, and personal privacy protection.

Challenges that may be encountered during the project include:

1. Data Collection: Gathering a diverse and representative dataset of typing patterns can be challenging, as it requires cooperation from a wide range of users.
2. Noise and Variability: Typing patterns can be influenced by various factors, including external noise, physical conditions, and typing habits, which may introduce variability and affect the accuracy of the AI model.
3. User Adaptability: The AI system needs to adapt to changes in users' typing patterns over time, accounting for natural variations and accommodating improvements or changes in typing style.

By recognizing these challenges and proactively addressing them, the project aims to develop a robust and reliable typing pattern detection AI system.

### Target Customers

The Typing Pattern Detection AI system is intended for individuals and organizations who prioritize computer security and want to add an extra layer of protection to their systems. The system can be particularly beneficial for the following customers:

1. **Individual Users**: Individuals concerned about unauthorized access to their personal computers, especially when working in public spaces or shared environments.
2. **Corporate Organizations**: Companies aiming to enhance the security of their computer networks, safeguard sensitive data, and protect against potential insider threats.
3. **Government Agencies**: Government institutions that handle classified information and require stringent security measures to prevent unauthorized access.
4. **Educational Institutions**: Schools, colleges, and universities that want to ensure secure computer usage by students, faculty, and staff members.
5. **Research Organizations**: Institutes engaged in sensitive research and development work, where protecting intellectual property is crucial.
6. **Cybersecurity Professionals**: Experts in the field of cybersecurity who wish to explore innovative techniques and solutions for intrusion detection and prevention.

The system is designed to be user-friendly and adaptable to different environments, making it accessible to both technical and non-technical users. The customization options available through the user interface allow customers to configure the system according to their specific requirements.

It is important to note that while the Typing Pattern Detection AI can provide an additional layer of security, it should not be considered a standalone solution. It is meant to complement existing security measures and best practices, such as strong passwords, firewalls, and regular system updates.

### Goals and Objectives

The Typing Pattern Detection AI system aims to achieve the following goals:

1. **Enhance Security**: The primary goal of the system is to enhance computer security by detecting unauthorized users attempting to type on the computer. It provides an additional layer of protection against potential intruders and helps prevent unauthorized access to sensitive information.
2. **User Identification**: The system aims to accurately identify and differentiate between authorized users and unauthorized individuals based on their unique typing patterns. This enables proactive measures to be taken when an unauthorized user is detected, such as triggering alerts or initiating security protocols.
3. **Ease of Use**: The system is designed to be user-friendly and accessible to both technical and non-technical users. It provides a seamless and intuitive user interface for easy interaction, configuration, and monitoring of the AI system.
4. **Adaptability**: The system strives to adapt to changes in users' typing patterns over time. It accounts for natural variations and accommodates improvements or changes in typing style without compromising accuracy.
5. **Performance and Accuracy**: The system aims to achieve high performance and accuracy in detecting and classifying typing patterns. By utilizing machine learning techniques and continuously improving the model, it strives to minimize false positives and false negatives.
6. **Flexibility**: The system offers flexibility in terms of customization and integration. Customers can tailor the system to their specific needs and integrate it with existing security infrastructure, providing a comprehensive security solution.
7. **Documentation and Support**: Comprehensive documentation and user support materials are provided to assist customers in understanding and effectively utilizing the system. This includes a detailed README file, user guides, and access to technical support for any queries or issues that may arise.

By achieving these goals, the Typing Pattern Detection AI system aims to provide a reliable and effective solution for detecting unauthorized typing activity and ensuring computer security for individuals and organizations.

### Problems and Benefits

#### Problems:

The Typing Pattern Detection AI system addresses the following problems:

1. **Unauthorized Access**: Unauthorized individuals gaining access to computer systems can pose significant security risks, leading to data breaches, privacy violations, and potential financial or reputational damages.
2. **Weaknesses in Traditional Security Measures**: Traditional security measures like passwords or biometrics have their limitations and can be vulnerable to various attacks, such as password guessing, social engineering, or biometric spoofing.

#### Goals and Benefits:

The Typing Pattern Detection AI system aims to achieve the following benefits:

1. **Enhanced Security**: By accurately detecting unauthorized typing activity, the system provides an additional layer of security to computer systems, significantly reducing the risk of unauthorized access and potential security breaches.
2. **Proactive Intrusion Detection**: The system enables proactive measures to be taken upon detecting unauthorized typing, such as triggering alerts, locking the system, or initiating security protocols. This allows for timely response and mitigates the risk of data compromise or unauthorized activities.
3. **User-Friendly and Non-Intrusive**: Unlike traditional security measures that may require users to remember complex passwords or undergo intrusive biometric scans, the Typing Pattern Detection AI system operates in the background without causing inconvenience to the users. It seamlessly integrates into the user's workflow, requiring no additional effort or conscious authentication.
4. **Improved User Experience**: The system eliminates the need for repeated manual authentication and provides a seamless user experience. Users can access their computers without interruption, while the AI works silently in the background, ensuring their security.
5. **Cost and Time Savings**: Implementing the Typing Pattern Detection AI system can lead to cost and time savings by reducing the need for additional security measures, such as expensive biometric scanners or complex password management systems. It offers a cost-effective and efficient solution for enhancing security.

### Services Provided:

The Typing Pattern Detection AI system provides the following services:

1. **Real-time Typing Pattern Detection**: The system continuously monitors typing activity in real-time, analyzing and identifying patterns to detect potential unauthorized users.
2. **User Identification and Classification**: The system accurately identifies and classifies users based on their unique typing patterns, distinguishing between authorized and unauthorized individuals.
3. **Alerts and Notifications**: Upon detecting unauthorized typing, the system can generate alerts or notifications, enabling immediate action to be taken to prevent unauthorized access or potential security breaches.
4. **User Interface (UI)**: The system offers a user-friendly interface for easy configuration, monitoring, and management of the AI system. Users can customize settings, view logs, and access relevant information through the UI.
5. **Documentation and Support**: Comprehensive documentation, including a detailed README file and user guides, is provided to assist users in understanding and effectively utilizing the system. Additionally, technical support is available to address any queries or issues that may arise.

By addressing the problems and providing these benefits and services, the Typing Pattern Detection AI system offers an effective, user-friendly, and cost-efficient solution for enhancing computer security and mitigating the risks associated with unauthorized access.

### Review of Existing Solutions

The Typing Pattern Detection AI system distinguishes itself from existing solutions and applications in the following ways:

1. **Alternative to Passwords and Biometrics**: While traditional security measures rely on passwords or biometrics for authentication, the Typing Pattern Detection AI system offers a unique approach. It leverages the individual's typing patterns, providing a non-intrusive and user-friendly method of identification, without the need for users to remember complex passwords or undergo biometric scans.
2. **Behavior-Based Authentication**: Unlike static authentication methods, such as passwords or fingerprints, the Typing Pattern Detection AI system focuses on dynamic user behavior. By analyzing the unique patterns and rhythm of an individual's typing, it creates a personalized profile that can be used to differentiate between authorized and unauthorized users.
3. **Continuous Monitoring and Proactive Detection**: The system continuously monitors typing activity in real-time, enabling proactive detection of unauthorized users attempting to access the computer. This offers a proactive security approach compared to conventional methods that rely on periodic authentication.
4. **Adaptability to User Changes**: The Typing Pattern Detection AI system is designed to adapt to changes in users' typing patterns over time. It accommodates natural variations and allows for adjustments in typing style, ensuring accurate identification even as users' behavior evolves.
5. **Customizability and Integration**: The system provides flexibility and customization options, allowing users to configure it according to their specific needs. It can be integrated with existing security infrastructure and complement other security measures, providing a comprehensive security solution.
6. **User-Friendly Interface**: The Typing Pattern Detection AI system offers a user-friendly interface that simplifies system configuration, monitoring, and management. It provides an intuitive and accessible platform for users to interact with the AI system without requiring technical expertise.
7. **Cost-Effective Solution**: Compared to expensive biometric scanners or complex password management systems, the Typing Pattern Detection AI system offers a cost-effective solution for enhancing security. It leverages existing hardware and software resources, reducing the need for additional investments.

While there may be existing solutions that utilize typing patterns for authentication or intrusion detection, the Typing Pattern Detection AI system aims to improve upon them by providing a more robust, adaptable, and user-friendly solution. It leverages the power of machine learning and artificial intelligence techniques to offer enhanced security and proactive identification of unauthorized users typing on the computer.

### Review of Project Technology

The Typing Pattern Detection AI project utilizes a combination of established technologies and techniques. While the concept of analyzing typing patterns is not new, the specific implementation and integration of machine learning and AI make this project innovative. Here is a review of the project's technology:

1. **Machine Learning and AI**: The project leverages machine learning algorithms and AI techniques to analyze and recognize typing patterns. These technologies enable the system to learn from data, classify patterns, and make accurate predictions based on the trained model.
2. **Data Preprocessing and Normalization**: The project employs data preprocessing and normalization techniques to ensure consistency and remove biases in the collected typing pattern data. This step enhances the accuracy and reliability of the AI model.
3. **User Interface (UI)**: The project includes a user-friendly UI that provides an intuitive platform for users to interact with the AI system. The UI allows users to configure settings, view logs, and access relevant information easily.
4. **Training and Testing Files**: Separate training and testing files are provided as part of the project. These files facilitate the training of the AI model on a representative dataset and the evaluation of its performance on unseen data.

### Caveats and System Setup Restrictions

1. **Dataset Diversity**: The accuracy and effectiveness of the Typing Pattern Detection AI system heavily rely on the diversity and quality of the dataset used for training. A more diverse dataset, comprising a wide range of typing styles and scenarios, can enhance the system's ability to generalize and detect unauthorized users accurately.
2. **Hardware and Software Compatibility**: The system's performance may be influenced by the hardware and software setup of the user's computer. Compatibility issues or limitations in processing power and memory may impact the system's ability to capture and analyze typing patterns effectively.
3. **Data Privacy and Ethics**: As the system captures and analyzes user typing patterns, it is essential to address data privacy concerns and ensure compliance with ethical guidelines. User consent and data anonymization should be implemented to protect individual privacy and prevent potential misuse of personal information.
4. **User Adaptability**: The system needs to adapt to changes in users' typing patterns over time. However, it may face challenges in accommodating sudden changes, such as injuries or disabilities that significantly alter typing behavior. Users should be aware of the system's limitations and be prepared to provide alternative authentication methods if needed.
5. **False Positives and False Negatives**: Like any pattern recognition system, the Typing Pattern Detection AI may encounter false positives (identifying authorized users as unauthorized) or false negatives (failing to identify unauthorized users). It is crucial to strike a balance between minimizing false alarms and ensuring accurate detection.

Overall, while the technology utilized in the Typing Pattern Detection AI project may not be entirely new, its implementation and integration provide an innovative approach to enhance computer security. The system setup requires attention to dataset diversity, hardware and software compatibility, data privacy, and user adaptability to maximize the system's effectiveness and mitigate potential limitations.

### Delimitation of the Project

The Typing Pattern Detection AI project focuses on specific areas related to computer security, networks, and operating systems. It is important to understand the project's scope and the areas it addresses, as well as the areas it does not cover. Here are the delimitations of the project:

Areas Covered by the Project:

1. **Computer Security**: The project aims to enhance computer security by detecting unauthorized typing activity and preventing unauthorized access to systems. It focuses on the identification and classification of users based on their typing patterns.
2. **Typing Pattern Analysis**: The project involves the collection, preprocessing, and analysis of typing patterns to develop an AI model capable of recognizing and distinguishing between authorized and unauthorized users. It utilizes machine learning and AI techniques for pattern recognition and prediction.
3. **Network and Operating System Integration**: The project can be integrated into computer networks and operating systems to provide an additional layer of security. It operates at the system level, monitoring typing activity and triggering security measures when unauthorized users are detected.

Areas Not Covered by the Project:

1. **Network Protocols and Encryption**: The project does not deal with the intricacies of network protocols, encryption, or data transmission. It focuses solely on the typing pattern analysis and user identification aspects of computer security.
2. **Hardware-level Security**: The project does not address hardware-level security mechanisms, such as physical access controls or tamper-resistant devices. It operates at the software level, utilizing typing patterns as a means of authentication.
3. **Application-Specific Security**: While the project enhances computer security, it does not specifically target the security of individual applications or software programs. It provides a system-level security measure applicable to the entire computer system.
4. **Intrusion Detection and Prevention**: While the project can help detect unauthorized users attempting to type on the computer, it does not cover all aspects of intrusion detection and prevention. It focuses on the typing pattern analysis as a means of identification and prevention of unauthorized access.

It is important to note that the Typing Pattern Detection AI project should be used as a complementary security measure and not as a standalone solution. It should be integrated with other security practices and best practices, such as strong passwords, firewalls, and regular system updates, to ensure comprehensive computer security.

By understanding the delimitations of the project, users can have a clear understanding of the areas it covers and the areas where additional security measures may be required.

# Detailed description of the system (characterization)

## Detailed System Description

The Typing Pattern Detection AI system is designed to enhance computer security by analyzing and recognizing the unique typing patterns of users. The system operates at the software level, monitoring typing activity in real-time and differentiating between authorized and unauthorized individuals attempting to type on the computer. Here is a more detailed description of the system's functionality:

1. **Typing Pattern Collection**: The system captures and collects typing patterns while users interact with the computer. It records various attributes of typing behavior, such as keystroke timings, key hold durations, key release timings, and overall typing rhythm.
2. **Data Preprocessing and Normalization**: The collected typing pattern data undergoes preprocessing and normalization techniques to ensure consistency and remove biases. This step involves cleaning the data, handling missing values, and normalizing the data to a standard format for analysis.
3. **Machine Learning Model Training**: The preprocessed and normalized typing pattern data is used to train a machine learning model. The model learns to recognize the unique patterns associated with each authorized user, creating personalized typing profiles.
4. **User Identification and Classification**: Once the model is trained, the system applies it to incoming typing patterns in real-time. It compares the observed typing patterns with the trained profiles to identify and classify the user as authorized or unauthorized.
5. **Proactive Intrusion Detection**: Upon detecting unauthorized typing activity, the system can trigger proactive measures to prevent unauthorized access. This can include generating alerts or notifications, initiating security protocols, or locking the computer to protect sensitive information.
6. **User Interface (UI)**: The system provides a user-friendly interface that allows users to interact with the AI system. Through the UI, users can configure system settings, view logs of detected typing patterns, and access relevant information about system performance and security.
7. **Adaptability and Learning**: The system is designed to adapt to changes in users' typing patterns over time. It accommodates natural variations in typing behavior and allows for improvements or changes in typing style without compromising accuracy.
8. **Integration and Customization**: The Typing Pattern Detection AI system can be integrated into computer networks and operating systems to provide system-level security. It is customizable, allowing users to configure settings according to their specific requirements and integrate it with existing security infrastructure.
9. **Documentation and Support**: Comprehensive documentation and user support materials are provided to guide users in effectively utilizing the system. This includes a detailed README file, user guides, and access to technical support for any queries or issues.

The Typing Pattern Detection AI system serves as an additional layer of security, complementing existing security measures, and providing proactive identification of unauthorized users attempting to type on the computer. By leveraging typing patterns and machine learning techniques, the system offers an innovative and user-friendly approach to enhance computer security and mitigate the risks associated with unauthorized access.

## System’s Abilities

End Users:

1. **Authentication Experience**: End users benefit from a seamless and non-intrusive authentication process. They can simply type on the computer, and the Typing Pattern Detection AI system analyzes their typing patterns to authenticate their identity without requiring additional authentication steps like passwords or biometrics.
2. **System Access**: Authorized end users can gain uninterrupted access to the computer system once their typing patterns are recognized and authenticated by the system. They can perform their tasks and use the system as usual without any interference from the Typing Pattern Detection AI system.
3. **User Interface Interaction**: End users have access to the user interface provided by the Typing Pattern Detection AI system. Through this interface, they can view their typing pattern history, review any security-related notifications or alerts, and modify their own profile settings if granted permission by the system administrator.
4. **Increased Security**: By utilizing the Typing Pattern Detection AI system, end users experience an enhanced level of security. The system helps to prevent unauthorized access by detecting and differentiating between authorized and unauthorized users based on their typing patterns. This reduces the risk of unauthorized individuals gaining access to the system and helps protect sensitive information.
5. **Non-Disruptive User Experience**: The Typing Pattern Detection AI system operates in the background, analyzing typing patterns without interrupting or affecting the end users' normal workflow. End users can continue to type and use the computer system without any noticeable changes or disruptions caused by the system.

It's important to note that the capabilities granted to end users are primarily related to their authentication experience, system access, and user interface interaction. The Typing Pattern Detection AI system aims to provide end users with a convenient and secure authentication method, ensuring a smooth and uninterrupted user experience while maintaining the desired level of computer system security.

## Black-Box checks

1. **Input Validation Test**:
   * Objective: To verify that the system correctly handles different types of input and properly validates the input data.
   * Test Execution: Generate various test cases with different input data, including normal inputs, invalid inputs, and edge cases. Observe how the system handles each input, ensuring it performs appropriate validation and handles errors gracefully.
2. **Authentication Accuracy Test**:
   * Objective: To evaluate the accuracy of the Typing Pattern Detection AI system in correctly identifying authorized and unauthorized users.
   * Test Execution: Prepare a diverse set of test cases involving authorized and unauthorized users. Measure the system's accuracy in correctly identifying each user category based on their typing patterns. Compare the system's results against a ground truth to determine the authentication accuracy.
3. **False Positive/Negative Test**:
   * Objective: To assess the system's ability to minimize false positives and false negatives.
   * Test Execution: Design test cases that include authorized users with different typing behaviors and unauthorized users attempting to mimic authorized users' typing patterns. Measure the system's performance in correctly identifying authorized users and detecting unauthorized users, ensuring a balance between minimizing false positives (legitimate users flagged as unauthorized) and false negatives (unauthorized users not detected).
4. **Performance and Scalability Test**:
   * Objective: To evaluate the system's performance and scalability under varying loads and user scenarios.
   * Test Execution: Simulate different usage scenarios, including varying numbers of concurrent users and typing patterns with different complexities. Measure the system's response time, resource utilization, and accuracy under different load conditions to ensure it performs efficiently and scales well.

It's important to thoroughly document the test cases, expected outcomes, and actual results for each test. This documentation will help track the system's performance, identify any issues or areas for improvement, and ensure that the Typing Pattern Detection AI system meets the desired quality and functionality requirements.

## Time Management

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| פעילות | זמן התחלה מתוכנן | זמן סיום מתוכנן | זמן התחלה בפועל | זמן סיום בפועל | הערות |
| יזום | 1.3.23 | 8.3.23 | 3.3.23 | 18.3.23 | איחרתי וסיימתי פחות מהרצוי |
| אפיון | 12.3.23 | 15.3.23 | 18.3.23 | 20.3.23 | איחרתי אך צמצמתי פערים |
| מדידת זמנים של כתיבה | 22.3.23 | 30.3.23 | 22.4.23 | 3.4.23 | התעכבתי |
| מודל | 4.4.23 | 18.4.23 | 4.4.23 | 20.4.23 | התעכבתי |
| סגירת פינות | 21.4.23 | 25.4.23 | 22.4.23 | 26.4.23 | התעכבתי |
| גרסה ראשונית | 27.4.23 | 13.5.23 | 28.4.23 | 10.5.23 | הקדמתי |
| גרסה סופית | 19.5.23 | 23.5.23 | 12.5.23 | 2.6.23 | איחרתי |
| סגירת תיק פרויקט | 25.5.23 | 27.5.23 |  |  |  |
| מצגת הסבר | 29.5.23 | 30.5.23 |  |  |  |

## Risk Management

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| הסיכון | פירוט הסיכון | רמת הסיכון | תיאור דרכים להתמודדות | מה בוצע בפועל | תאריך |
| עמידה בלוז | פרויקט לא גמור | קשה | הקדמת לוזים  תיעדוף קידוד | הקידוד הפך להיות priority | 1.5.23 |
| חוסר אמינות של המודל | מודל לא עובד ולא אמין | בינונית | קריאת מאמרים  התייעצות | אבא שלי הביא בחור שעוסק בעבודה שלו בAI | 3.5.23 |
| באגים ויעילות | סגירת פינות שהתכנה תרוץ חלק יותר | קלה | דיבאגינג  חזרה על הקוד | חזרתי על הקוד וייעלתי שורות שהיו סתם | 3.6.23 |

# In-Depth Breakdown of Client-Side Capabilities in the Typing Pattern Detection AI System

## Authentication Experience:

* **Essence**: This ability enables end users to authenticate their identity through their typing patterns without additional authentication steps.
* **Actions**:
  + Collect and analyze typing patterns in real-time.
  + Compare the observed typing patterns with the user's registered typing profile.
  + Determine the authenticity of the user based on the similarity of typing patterns.
* **Necessary Objects**:
  + Typing pattern data collected from the user.
  + Registered typing profiles of authorized users.

## System Access:

* **Essence**: This ability allows authorized end users to gain uninterrupted access to the computer system once their typing patterns are recognized and authenticated.
* **Actions**:
  + Verify the authenticity of the user's typing patterns.
  + Grant access privileges to authorized users.
  + Monitor typing patterns for any changes or anomalies during the user's session.
* **Necessary Objects**:
  + Typing pattern data of authorized users.
  + Access control settings and privileges.

## User Interface Interaction:

* **Essence**: This ability enables end users to interact with the user interface provided by the Typing Pattern Detection AI system.
* **Actions**:
  + View and access the user interface elements.
  + Navigate through different sections and features of the user interface.
  + Modify personal profile settings if permitted.
* **Necessary Objects**:
  + User interface components and elements.
  + User profile settings and preferences.

## Increased Security:

* **Essence**: This ability enhances the security of the computer system by leveraging typing patterns for user authentication.
* **Actions**:
  + Detect and differentiate between authorized and unauthorized users based on their typing patterns.
  + Generate alerts or trigger security measures upon detecting unauthorized typing activity.
  + Prevent unauthorized access to the system.
* **Necessary Objects**:
  + Typing pattern data for authorized and unauthorized users.
  + Security protocols and measures.

## Non-Disruptive User Experience:

* **Essence**: This ability ensures that the Typing Pattern Detection AI system operates in the background without causing disruptions to the end users' workflow.
* **Actions**:
  + Analyze typing patterns without interrupting the user's normal typing experience.
  + Maintain a seamless authentication process without additional manual steps.
  + Minimize any noticeable changes or disruptions caused by the system.
* **Necessary Objects**:
  + System processes and algorithms for real-time analysis.
  + User feedback and usability testing.

These client-side capabilities in the Typing Pattern Detection AI system provide a seamless and secure authentication experience, enable system access for authorized users, facilitate user interface interaction, enhance overall security, and ensure a non-disruptive user experience. By leveraging typing patterns, the system strengthens computer security and delivers a user-friendly authentication process for end users.

# Structure / architecture of the project

## hardware description

Only a single computer is needed. Nothing more.

1. **Computer System**:
   * This forms the main hardware component on which the Typing Pattern Detection AI system will be deployed.
   * It consists of a CPU, GPU, memory, storage devices, and input/output peripherals.
2. **Database Server**:
   * This component hosts the existing database of users, which contains their typing pattern data.
   * The database server provides storage and retrieval capabilities for user data.
3. **User Data Integration**:
   * The user data integration process involves combining the existing user database with the main user data using the "create\_train\_file.py" Python script.
   * This script retrieves user data from the existing database and merges it with the main user data for further processing.
4. **Data Normalization**:
   * The combined user data, obtained from the integration process, is then passed through a normalization process.
   * The normalization aims to standardize the data, ensuring consistency and compatibility for subsequent processing.
5. **Model Training**:
   * The normalized data is used to train a machine learning model that will analyze and identify typing patterns.
   * The "train\_model.py" Python script is responsible for training the model using the normalized data.
   * The trained model captures the patterns and characteristics of authorized users' typing behavior.
6. **User Interface**:
   * The "user\_interface.py" Python script provides a graphical user interface (UI) for end users to interact with the Typing Pattern Detection AI system.
   * The UI allows users to input their typing patterns for authentication and provides visual feedback on the authentication process.
7. **Model Testing**:
   * The "test\_model.py" Python script utilizes the trained model to analyze and validate the typing patterns of users.
   * It compares the user's input with the patterns stored in the model to determine the authenticity of the user.

The connections between these components involve the flow of data and execution of scripts. The "create\_train\_file.py" script combines the existing user database with the main user data, which is then passed to the "train\_model.py" script for model training. The trained model is saved for later use by the "user\_interface.py" script, which provides the UI for user interaction. The "test\_model.py" script uses the trained model to analyze and verify the typing patterns entered through the UI.

Overall, this architecture allows for the integration of existing user data, training of a machine learning model, and providing a user-friendly interface for authentication using typing pattern analysis.

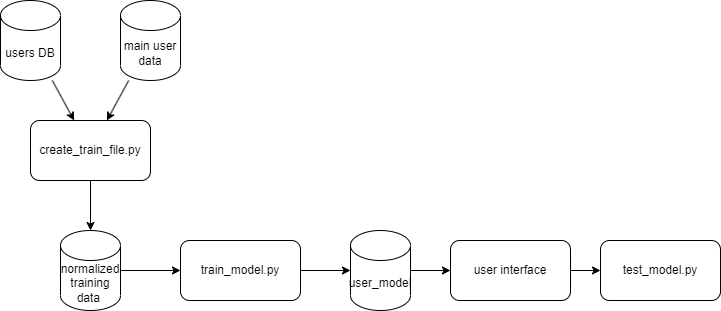
## Description of Relevant Technology

The Typing Pattern Detection AI system utilizes various technologies for its implementation. Here is a description of the relevant technologies:

1. **Programming Language**:
   * The system is developed using the Python programming language.
   * Python provides a wide range of libraries and frameworks for data processing, machine learning, and user interface development.
   * Its simplicity, readability, and extensive community support make it an ideal choice for implementing AI-based systems.
2. **Operating System**:
   * The system can be developed and deployed on multiple operating systems, including Windows, macOS, and Linux distributions.
   * The choice of the operating system depends on the specific requirements and preferences of the users and developers.
3. **Communication**:
   * The Typing Pattern Detection AI system may involve communication between different components.
   * The communication can be achieved through inter-process communication (IPC) mechanisms provided by the operating system, such as sockets, pipes, or shared memory.
   * Additionally, the system may utilize communication protocols, such as HTTP or WebSocket, for communication between the user interface and the backend components.

The selected technologies, such as Python programming language and the choice of operating system, enable the system to effectively process and analyze typing patterns, train models, and provide a user-friendly interface. The use of inter-process communication mechanisms and protocols allows for seamless communication between different components of the system.

Flow Of Information



## Description of the Main Algorithms in the Project

In the Typing Pattern Detection AI project, the main goal is to develop an algorithm that can analyze and identify typing patterns for user authentication. The algorithmic problem involves finding an effective approach to recognize and differentiate between authorized and unauthorized users based on their typing behavior. Here is an overview of the formulation, existing algorithms, and the chosen solution in the project:

1. **Formulation and Analysis of the Algorithmic Problem**:
   * The algorithmic problem in this project revolves around developing a model that can accurately classify and authenticate users based on their typing patterns.
   * It requires analyzing various features of typing behavior, such as key press timings, key hold durations, and inter-key intervals, to capture the unique characteristics of individual users.
2. **Description of Existing Algorithms to Solve the Problem**:
   * Initially, the project explored the use of popular machine learning algorithms such as K-Nearest Neighbors (KNN) and Decision Forests.
   * KNN algorithm attempts to classify users based on the similarity of their typing patterns to previously observed patterns in the training data.
   * Decision Forests (e.g., Random Forest) employ an ensemble of decision trees to classify users based on various features of their typing behavior.
3. **Reviewing the Chosen Solution**:
   * Despite attempting the use of KNN and Decision Forest algorithms, they did not yield satisfactory results in accurately identifying and authenticating users based on their typing patterns.
   * As a result, the project moved towards exploring other solutions and experimented with the One-Class Support Vector Machine (SVM) algorithm.
   * However, the One-Class SVM approach also did not meet the desired performance criteria.
4. **Development of Original Solution: Keras Models Sequential**:
   * After exploring existing algorithms, the project ultimately decided to develop an original solution using the Keras library's Sequential model.
   * The Sequential model in Keras provides a flexible and intuitive way to build neural networks for various machine learning tasks.
   * By designing a custom architecture using Sequential, the project could leverage deep learning techniques to capture and analyze intricate patterns in typing behavior.

The choice of using Keras Models Sequential was justified based on the need for a more sophisticated and adaptable approach to accurately identify and authenticate users based on their typing patterns. This approach allows for the creation of customized neural network architectures tailored to the specific requirements of the Typing Pattern Detection AI system.

By adopting a novel approach with the Keras Models Sequential, the project aims to achieve improved accuracy and performance in detecting typing patterns and authenticating users.

## Description of the Development Environment

The development of the Typing Pattern Detection AI system requires specific tools and libraries to facilitate the coding and testing process. Here is a list of the development tools needed for development:

1. **Python**: The development environment is based on the Python programming language, which provides the necessary syntax and features for implementing the system.
2. **Text Editor/IDE**: I wrote my code using Pycharm.
3. **Libraries and Dependencies**: Several libraries and dependencies are needed for the development of the Typing Pattern Detection AI system. These libraries can be installed using the requirements file, which contains the following libraries:
   * random: Used for generating random numbers or samples.
   * h5py: A package to interact with the HDF5 file format.
   * ast: Provides tools for working with Abstract Syntax Trees in Python.
   * os: Provides a way to interact with the operating system, e.g., file handling and directory operations.
   * numpy: A powerful library for numerical computing in Python.
   * scikit-learn: A machine learning library that provides various algorithms and tools for data analysis.
   * pynput: Enables monitoring and controlling input devices, such as keyboards and mice.
   * tkinter: A standard Python GUI toolkit for creating graphical user interfaces.
   * keras: A high-level neural networks library built on top of TensorFlow, used for building and training models.
   * tensorflow: An open-source machine learning framework used for deep learning tasks.

## Description of the Testing Environment

The testing environment for the Typing Pattern Detection AI system requires the following tools and resources:

1. **Testing Data**: A collection of test data representing various typing patterns is needed to evaluate the accuracy and performance of the system. This data can be obtained from real-world users or generated synthetically for controlled testing.
2. **Test Scripts**: Test scripts are created to automate the testing process. These scripts simulate user inputs and evaluate the system's response, checking if the typing patterns are correctly identified and authenticated.

By utilizing the above development tools and setting up a dedicated testing environment, the Typing Pattern Detection AI system can be thoroughly tested for its accuracy, performance, and robustness.

## Description of the Communication Protocol

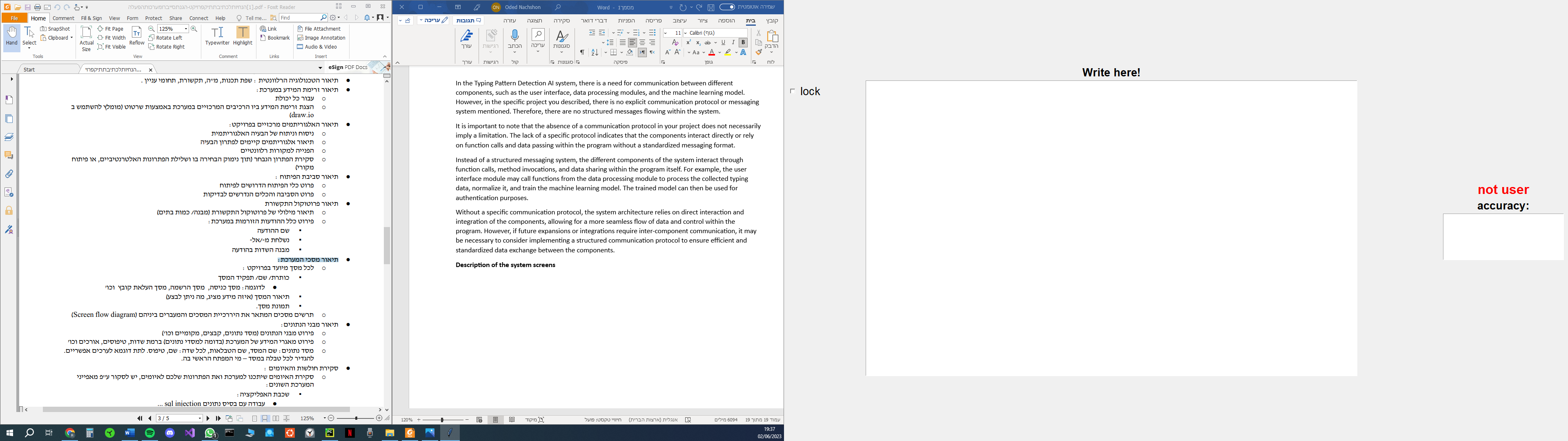
In the Typing Pattern Detection AI system, there is a need for communication between different components, such as the user interface, data processing modules, and the machine learning model. However, in the specific project you described, there is no explicit communication protocol or messaging system mentioned. Therefore, there are no structured messages flowing within the system.

It is important to note that the absence of a communication protocol in your project does not necessarily imply a limitation. The lack of a specific protocol indicates that the components interact directly or rely on function calls and data passing within the program without a standardized messaging format.

Instead of a structured messaging system, the different components of the system interact through function calls, method invocations, and data sharing within the program itself. For example, the user interface module may call functions from the data processing module to process the collected typing data, normalize it, and train the machine learning model. The trained model can then be used for authentication purposes.

Without a specific communication protocol, the system architecture relies on direct interaction and integration of the components, allowing for a more seamless flow of data and control within the program. However, if future expansions or integrations require inter-component communication, it may be necessary to consider implementing a structured communication protocol to ensure efficient and standardized data exchange between the components.

## Description of the system screens



The Typing Pattern Detection AI system features a single window user interface with different components that facilitate user interaction and display system outputs. Here is a breakdown of the system screens and their respective components:

1. **Main Window**:
   * The main window serves as the primary interface for the Typing Pattern Detection AI system.
   * It contains the following components:
   * **Output Box (Right Side)**:
     + The output box, located on the right side of the window, displays the AI's prediction values.
     + It provides real-time feedback on the system's assessment of the user's typing pattern, indicating the likelihood of authentication or unauthorized access.
   * **AI Decision Text (Right Side)**:
     + Adjacent to the output box, there is a text section that conveys the AI's decision.
     + This text provides additional information or a specific decision made by the AI regarding the user's typing pattern.
   * **Input Window (Middle)**:
     + Positioned in the middle of the window, the input window is where the user types to demonstrate their typing pattern.
     + Users can interact with the system by typing text within this input window, allowing the AI to analyze their typing behavior.
   * **Lock Screen Checkbox (Left Side)**:
     + On the left side of the window, there is a checkbox option that allows the user to decide whether to enable the screen lock feature.
     + If selected, the system will automatically lock the screen after a certain number of false identification attempts, typically 10.

The main window of the system provides a clear and intuitive interface for users to observe the AI's predictions, input their typing patterns, and control the screen lock functionality. The arrangement of the components ensures easy access and visibility of important information, promoting a user-friendly experience.

## Description of the Data Structures

The Typing Pattern Detection AI system utilizes various data structures to store and process different types of information. Here is a breakdown of the data structures involved:

1. **Initial Users Database** (txt file):
   * This database contains the typing patterns of multiple users and serves as the initial reference for user authentication.
   * The structure of each record in the database is an array with the following fields:
     + Key 1: The first key pressed by the user.
     + Key 2: The second key pressed by the user.
     + Timing Float 1: A float value representing the timing or duration of the first key press.
     + Timing Float 2: A float value representing the timing or duration of the second key press.
     + Timing Float 3: A float value representing the timing or duration between the first and second key presses.
     + Timing Float 4: A float value representing the timing or duration between the second and third key presses.
   * Example values for a record in the initial users database: **[164, 9, 0.219, 0.124, 0.11, -0.109]**
2. **Main User Database** (txt file):
   * This database contains the typing patterns of the main user who is currently using the system.
   * Similar to the initial users database, each record in this database has the following structure:
     + Key 1: The first key pressed by the main user.
     + Key 2: The second key pressed by the main user.
     + Timing Float 1: A float value representing the timing or duration of the first key press.
     + Timing Float 2: A float value representing the timing or duration of the second key press.
     + Timing Float 3: A float value representing the timing or duration between the first and second key presses.
     + Timing Float 4: A float value representing the timing or duration between the second and third key presses.
   * Example values for a record in the main user database: **[78, 79, 0.14441466331481934, 0.12945318222045898, 0.0945744514465332, -0.04984021186828613]**
3. **Normalized Data** (h5 file):
   * After processing and normalizing the typing patterns, the data is saved in an h5 file format.
   * The structure and organization of the normalized data may depend on the specific requirements and techniques used for normalization.
4. **Trained Model** (h5 file):
   * The trained machine learning model, which is used for identifying and authenticating typing patterns, is saved in an h5 file format.
   * The exact structure and organization of the model depend on the chosen architecture and the framework used for training, such as Keras.

It is important to note that the specific details and structure of the databases and files may vary based on the implementation and requirements of the Typing Pattern Detection AI system. The provided examples give a general idea of the fields and values involved in the data structures but may not reflect the entire complexity of the system's data representation.

## Overview of Weaknesses and Threats

In the Typing Pattern Detection AI system, it is essential to consider potential weaknesses and threats that may impact the system's security and functionality. However, my system is not web-connected and does not involve external network communication, some of the common vulnerabilities and threats related to web applications and network protocols are not be applicable.

# The Implementation Of The Project תחתית הטופס

**Overview of System Modules and Interrelationships**

In the implementation of the Typing Pattern Detection AI system, there are several modules and imported libraries that play a role in the system's functionality. Here is an overview of these modules and their interrelationships:

**Imported Modules/Departments:**

1. **random**:
   * This module is imported to enable randomization in the system. It is used to slice the data randomly, ensuring randomness in the training and testing datasets.
2. **h5py**:
   * The h5py module is used for working with the Hierarchical Data Format (HDF5) files. It allows the system to save the normalized data and trained model as h5 files, providing efficient storage and retrieval capabilities.
3. **ast**:
   * The ast module is imported to facilitate the literal evaluation of lines in the data files. It is used to convert string representations of arrays into actual array objects for further processing.
4. **os**:
   * The os module provides functionalities related to the operating system. It is utilized in the system to perform various tasks such as locking the screen after a certain number of false identifications and accessing files on the local system.
5. **numpy**:
   * The numpy library is imported to enable efficient handling and manipulation of multi-dimensional arrays. It is used in the system to arrange and organize the data in a structured format for further analysis and processing.
6. **scikit-learn**:
   * The scikit-learn library is utilized in the system for various machine learning tasks. It provides functionalities for data preprocessing, shuffling of data, and training of machine learning models.
7. **pynput**:
   * The pynput module is imported to enable recording of keyboard events. It allows the system to capture the typing patterns of the user and process them for analysis and authentication purposes.
8. **tkinter**:
   * The tkinter library is used for developing the user interface of the system. It provides GUI components and functionalities, allowing users to interact with the system through a graphical interface.
9. **keras**:
   * The keras library is imported to enable the loading and training of the machine learning model. It provides high-level APIs and utilities for building and training neural networks.
10. **tensorflow**:
    * The tensorflow library is imported to support the functionalities of keras. It is an open-source machine learning framework that provides a computational backend for building and training deep learning models.

These imported modules and libraries enhance the system's capabilities by providing various functionalities required for data manipulation, machine learning, user interface development, and system interaction.

It is important to note that the system does not include additional self-developed modules or departments beyond these imported libraries.