International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Machine Learning-Based Keystroke Biometrics for Enhanced Web Security

Adejumo, Samuel Olujimi[1,2] Alade, Samuel Mayowa[1,2], Oyerinde, Joshua Kehinde[3], Olatunde, Ayodeji Akano[4], Osakwe, Godwin O.[5]

[1]*Department of Cyber security, Nnamdi Azikiwe University, Awka, Nigeria*
[2]*Department of Computer Science, Nnamdi Azikiwe University, Awka, Nigeria*
[3,4]*Department of Computer Sciences, Abiola Ajimobi Technical University, Ibadan, Nigeria*
[5]*Department of Cyber Security, Southern Delta University, Ozoro, Nigeria*

***Abstract*-- The rise of sophisticated cyber threats has exposed vulnerabilities in traditional authentication methods such as passwords and multi-factor authentication. This study explores machine learning and behavioral biometrics, specifically keystroke dynamics, to enhance web authentication through continuous, user-specific monitoring. Keystroke datasets from Kaggle were preprocessed using data cleaning, normalization, feature extraction, and dimensionality reduction with Principal Component Analysis (PCA). Three anomaly detection models: Isolation Forest, One-Class SVM, and Local Outlier Factor were evaluated using accuracy, precision, recall, F1-score, and error rates. Isolation Forest showed superior detection capability and was integrated into a real-time dashboard system built with Next.js and Neo4j, enabling live monitoring, anomaly detection, and behavioral visualization. Results demonstrated effective differentiation between legitimate and anomalous keystroke behavior with low false positives. This framework offers an adaptive, efficient, and scalable solution for mitigating unauthorized access in web applications.**

***Keywords*-- Machine Learning, Keystroke Dynamics, Anomaly Detection, Web Authentication**

I. INTRODUCTION

The advancement in information technology, particularly the widespread adoption of digital platforms, has created new challenges in safeguarding user identities and data. Traditional authentication methods, such as passwords and multi-factor authentication (MFA), are increasingly vulnerable to sophisticated attacks including phishing, brute-force attempts, and adversarial artificial intelligence (Lai et al., 2023). These weaknesses expose sensitive information, compromise user privacy, and elevate financial and security risks. Prior to the emergence of advanced cyber threats, authentication systems relied mainly on static measures. While once effective, these approaches no longer provide sufficient protection in the face of evolving attacker strategies. Recent studies have shown that behavioral biometrics, particularly keystroke dynamics, can provide a promising alternative by enabling continuous user verification in real time (Kataria, 2023).

Early research explored intrusion detection and authentication models (Al-Nashif et al., 2008; Huang, 2005), and more recent efforts have applied generative adversarial networks (GANs) to improve resilience under adversarial conditions (Usama et al., 2019).

However, ML-driven web authentication still faces significant challenges, such as computational overhead, privacy risks, and adversarial attacks. The seriousness of these threats is not only measured by their frequency but by their potential consequences, including unauthorized access, financial losses, and erosion of trust in digital platforms (Rani et al., 2021).

Therefore, there is a pressing need for a robust authentication framework that leverages machine learning and behavioral biometrics to enhance security while maintaining usability. This study aims to design and implement a keystroke-based authentication system that integrates anomaly detection models to identify and prevent unauthorized access in web applications. Specifically, it evaluates supervised and unsupervised learning approaches, compares their effectiveness against adversarial conditions, and develops a scalable real-time solution for strengthening web authentication.

II. LITERATURE REVIEW

One of the most sophisticated authentication methods is behavioral biometrics which monitors user activity with respect to their identity constantly. Unlike behavioral biometrics, traditional authentication methods like passwords and PINs offer a single instance of protection (Gunuganti, 2023). In this subsection biometric methods such as keystroke identification, analysis of mouse movements, gesture identification, speech recognition and their corresponding functions in intrusion resistant authentication systems are described. Gunuganti (2023) identified different types of behavioral biometrics as:

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Keystroke dynamics: Keystroke dynamics analyze how users type, focusing on variables such as keystroke timing, dwell time (how long a key is pressed), and flight time (time between key presses). These metrics are unique to each individual, making them useful for continuous authentication. Research has shown that keystroke dynamics achieve high accuracy in user identification without requiring additional hardware.

Mouse Movement Analysis: Mouse movement analysis tracks cursor speed, trajectory, acceleration, and click patterns to distinguish between legitimate users and impostors. Since each user has a unique way of interacting with a computer mouse or touchpad, machine learning models can use this data for authentication purposes. This method is particularly effective in real-time authentication without disrupting the user experience.

Gesture Recognition: Gesture recognition identifies users based on hand movements, touch gestures, and motion patterns. This technique is widely used in mobile authentication, where users interact with touchscreen devices. It integrates behavioral and physiological biometrics, making authentication more secure and resistant to spoofing attacks.

Voice Analysis: Voice analysis evaluates pitch, tone, pronunciation, and speech rhythm to authenticate users. This biometric technique is commonly used in voice assistants and call center authentication systems. By leveraging machine learning models, voice recognition can achieve high accuracy and be integrated into multi-factor authentication frameworks.

A. Related Works

Incorporating cybersecurity advancements through machine learning (ML), the development of system-driven intrusion monitoring systems has grown in use and importance. While the field of research has not achieved everything, existing models and practices have achieved a degree of success. Several studies have performed and focused on detecting breaches through machine learning, notably using anomaly-based techniques. Some models, however, have excessive computational cost and low effectiveness, posing problems in application outside of the laboratory. Although there is improvement in the architecture of deep learning, other issues like real time detection and scaling still have not been fully solved.

Moreover, the consideration of pre-authentication behavioral biometrics, with an emphasis on keystroke dynamics, has increased among researchers. Notably, combining these biometrics with ML-based intrusion detection has not been exploited fully.

This approach is limited by the absence of large-scale datasets and monitored real-life environments necessary to implement these solutions.

In these contexts, a number of methods on intrusion detection have been developed and tested independently for their accuracy in supervised environments. Many systems do not perform well when placed in a dynamic and hostile environment. Constantly changing attack patterns create a challenge for systems that are static, exposing them to Adversarial Machine Learning. In an attempt to fill in the gaps, this study focuses on the improvement of verification by using intrusion detection with behavior biometrics based on machine learning. With anomaly detection, adaptive learning, and data processing in real-time, the system will be able to overcome existing constraints and enhance the development of authentication models that are extensible, strong, and less intrusive.

Huang et al. (2006) proposed a distributed authentication system with intrusion tolerance, enhancing security by utilizing redundant proxy servers and shared authentication servers. Their approach demonstrated improvements in mitigating unauthorized access attempts by ensuring system redundancy.

Huang (2005) introduced a Scalable Intrusion Tolerance Architecture (SITAR) designed to prevent dictionary attacks through a combination of redundancy and distributed intrusion detection techniques. This approach significantly reduced unauthorized access in networked systems.

Davuluri (2023) developed a ML-based authentication system utilizing mouse dynamics. The system successfullymachieved high accuracy in detecting unauthorized access by analyzing users' unique mouse movement behaviors.

Pyla et al. (2024) implemented a ML-based Network Intrusion Detection System (NIDS), which demonstrated improved accuracy and efficiency in detecting cyberattacks using logistic regression. The study highlighted how machine learning models can enhance network security by identifying anomalies in real time.

Al-Nashif et al. (2008) introduced a Multi-Level Intrusion Detection System (ML-IDS) that enhances detection rates by applying three levels of traffic analysis. Their system improved response times and accuracy in identifying potential threats within a network.

Kataria (2023) designed a ML-based IDS with adaptive learning capabilities, which achieved superior detection accuracy while significantly reducing false positives. The system adapted dynamically to new attack patterns, improving its overall effectiveness in real-world scenarios.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Talpini et al. (2023) proposed an uncertainty-aware ML-based IDS, improving trustworthiness by minimizing overconfident predictions in intrusion detection models. Their approach ensured that false alarms were reduced, making the system more reliable for enterprise security applications.

III. Methodology

This section presents a comprehensive examination of the methods, techniques, tools, and procedures employed in the development and implementation of the ML-based keystroke biometrics system for enhanced web security. Starting with an in-depth analysis of the system architecture, it explores the complexities of data collection, preprocessing, model selection, and training for keystroke anomaly detection in web environments. Additionally, the methodology incorporates keystroke dynamics patterns as a core behavioral biometric into the authentication framework to enhance web application security. It examines the real-time implementation of keystroke anomaly detection models and their integration into the web authentication system to ensure adaptive learning and resilience against cyber threats targeting online access. The section concludes with a discussion on the evaluation metrics, performance assessment, and deployment strategies used to validate the effectiveness of the system for web security.

A. Data Preprocessing for Kaggle Keystroke Dynamics Datasets

This study utilizes Kaggle keystrokes dynamics datasets containing keystroke dynamics patterns to enhance the security of web application authentication. These behavioral biometric datasets were collected to establish user behavior profiles and detect anomalies that may indicate unauthorized access attempts in online environments. The keystroke dataset includes features such as key press duration, typing speed, key hold time, and inter-key delay. These datasets were processed to remove inconsistencies, filter noise, and extract relevant features for training machine learning models specifically designed for keystroke biometrics.

The datasets were loaded into Python in CSV format using Pandas, and preprocessing steps such as handling missing values, scaling feature values, and normalization were carried out to improve data quality and enhance the performance of classification algorithms for behavioral anomaly detection. Feature extraction was performed using Principal Component Analysis (PCA) and Time-Series Segmentation to reduce dimensionality while preserving crucial user behavior patterns relevant to keystroke dynamics.

The cleaned and preprocessed datasets were then split into training and testing sets using an 80:20 ratio, ensuring a balanced dataset for training machine learning models aimed at enhancing web security through keystroke biometrics.

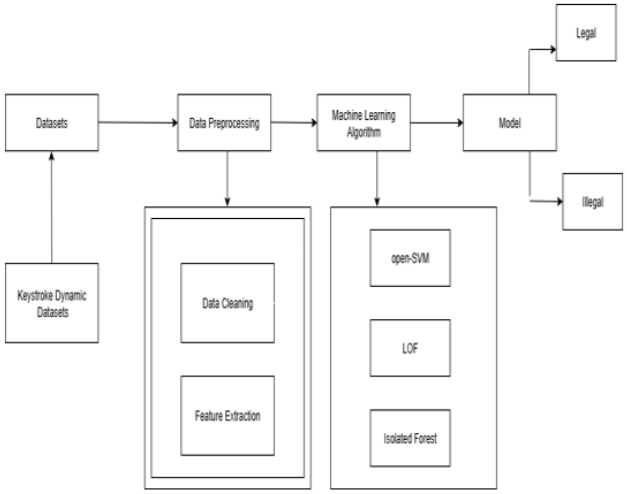

Figure 1: Model Building Architectural Diagram

B. Dataset

The dataset utilized in this study consists of Kaggle keystrokes dynamics datasets collected to enhance web security. The details of the dataset are summarized below.

Table I
Dataset used

Datasets	Number of records	Features Extracted	Data Formats
Kaggle keystrokes dynamics datasets	10,000	Key press duration, Typing speed, Inter-key Delay	CSV

The keystroke dynamics dataset records user typing patterns, including key press duration, typing speed, inter-key delay, and dwell time. These biometric features are used to differentiate between legitimate users and potential intruders. The dataset was collected from multiple users performing controlled typing tasks to ensure diverse behavioral patterns. Users interacted with predefined tasks to simulate real-world authentication scenarios, ensuring the dataset is representative of various user behaviors.

C. Data Preprocessing

This section details the data preprocessing methods used to transform raw keystroke data into a structured format suitable for machine learning models.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

1) *Data Cleaning:* Data cleaning is the initial step in the proposed model. The goal of this task is to remove inconsistencies, handle missing values, and normalize input features to enhance the performance of the authentication model. The specific preprocessing steps include: (1) Normalization: Standardizing keystroke features to ensure consistency in data representation.(2) Outlier Removal: Identifying and eliminating extreme values that could distort model training. (3) Missing Value Handling: Imputing missing data using statistical methods to maintain dataset integrity. (4) Feature Scaling: Applying techniques such as Min-Max Scaling and Standardization to optimize model performance. The preprocessed dataset was then split into training and testing sets to ensure effective model generalization and performance evaluation.
2) *Feature Extraction:* This section describes the feature extraction techniques used to transform the cleaned keystroke data into meaningful features for machine learning models. The extracted features include: (1) Key Press Duration: The amount of time a key is held down before release. (2) Inter-Key Delay: The time interval between releasing one key and pressing the next. (3) Typing Speed: The rate at which a user types, measured in characters per second. (4) Dwell Time: The duration for which a key remains pressed.

D. Model Detection of Anomaly Behavior and Comparative Evaluations of Machine Learning Algorithms

The development of the models and a comparison study of the performance evaluation results are discussed in this section.

1) *Building the Model for Detecting Anomalous Behavior:* To select the best ML algorithms, the dataset was split using different training-to-testing ratios, such as 90:10,80:20, and 70:30, and evaluated using classification performance metrics. The classifier with the best performance was chosen to build the anomaly detection system.

The system was implemented using Python libraries, including Scikit-learn, NumPy, Pandas, and TensorFlow, ensuring optimal model performance. The best-performing classification model was used as the golden model for anomaly detection. The anomaly detection model was trained using extracted keystroke and mouse movement features. The input feature vector X consisted of various behavioral parameters, and the model predicted whether an authentication attempt was normal or anomalous. The binary classification function is represented as:

$$y = f(X)$$

Equation(i)

where:
y is the prediction (0 for normal, 1 for anomaly)
X represents the extracted features from keystroke dynamics and mouse movements
f is the trained ML model.

The loss function used for optimization was the binary cross-entropy, defined as:

$$L = -N1i = 1\sum N[yi\text{loglog}(yi) + (1 - yi)\text{loglog}(1 - yi)]$$

Equation(ii)

Where:
yi is the actual label (0 or 1)
$y^{i\backslash hat\{y\}i}$ is the predicted probability of anomaly
N is the number of training samples.

One-Class Support Vector Machine (OC-SVM)

OC-SVM is a boundary-based anomaly detection algorithm. It tries to find the smallest hypersphere or margin that contains most of the data points, assuming that anomalies lie outside this boundary.

Decision function:

Isolation Forest (IF)

Isolation Forest is an ensemble method specifically designed for anomaly detection. It isolates anomalies by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. Anomalies are isolated faster because they require fewer splits.

Anomaly score formula:

$$s\ (x, n) = 2\ ^{-E|(h(x))}$$

Equation (iii)

Where:
$S\ (x, n)$ is the anomaly score for data point
$E(h(x))$ is the average path length of xxx in an ensemble of isolation trees
$c(n)$ is the average path length of unsuccessful searches in Binary Search Trees, used for normalization:

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

$$\frac{c(n) = 2H(n-1) - n}{n2(n-1)}$$

Equation(iv)

$$Where: H(i) = In(i) + 0.5772(Euler's constant)$$

$$f(x) = sign(\langle w, \phi(x) \rangle - \rho)$$

Equation(v)

Where:
x is the input feature vector
$\phi(x)x$ into a higher-dimensional space w is the weigh vector
ρ is the offset

Radial Basis Function (RBF):
$K(xi,xj) = \exp(-\gamma \| xi - xj \| 2)$
Euation(vi)

Polynomial:

$K(xi,xj) = (\langle xi,xj \rangle + c)d$
Equation(vii)

Local Outlier Factor (LOF)

LOF measures the local density deviation of a given data point with respect to its neighbors. A point is considered anomalous if it has a substantially lower density than its neighbors.

LOF formula:

$$LOFk(x) = | Nk(x) | 1y \in Nk(x) \sum lrdk(x) lrdk(y)$$ Equation(viii)

Where:
$Nk(x)$ is the set of k-nearest neighbors of point x.
$lrdk(x)$ is the local reachability density of point x, which is the inverse of the average reachability distance from the neighbors.

A score greater than 1 implies a potential anomaly; a score $\approx$ 1 indicates normality.

Each model was evaluated using metrics such as Precision, Recall, F1-score, and AUC-ROC. The model with the best balance between detection accuracy and computational performance was selected for real-time integration into the dashboard.

1) Developing a Real-Time Device and Analysis Dashboard System,

In addition to capturing keystroke input, the system includes a real-time device monitoring and analysis dashboard that supports advanced visualization of behavioral patterns and anomaly scores. This dashboard provides administrators with a comprehensive view of ongoing sessions, connected devices, user behaviors, and detected anomalies. Each connected session is linked with metadata such as device type, IP address, browser fingerprint, operating system, and geolocation (where applicable). These details are visualized in the dashboard and correlated with keystroke sessions stored in the Neo4j graph database. Neo4j's schema-free, relationship-focused design makes it ideal for associating sessions with devices and evaluating behavioral trends across time or user accounts.

The dashboard provides analytical insights including: (1) Time-series plots of user keystroke metrics (latency, dwell, flight). (2) Real-time anomaly score graphs. (3) Historical summaries of each user's behavior. (4) Device usage correlation and multi-session detection.

The backend, built in Node.js and Express.js, manages live data streams and sends them to the dashboard frontend. This facilitates real-time updates as users type or switch devices. The anomaly detection engine evaluates each input batch using Isolation Forest, Local Outlier Factor, and One-Class SVM, then transmits results to the dashboard for immediate visualization. To maintain integrity and privacy, access to this analytics dashboard is role- restricted and protected using token-based authentication and encrypted API endpoints. This ensures that behavioral monitoring remains both accurate and secure, allowing system administrators to detect potential threats and analyze risky behavior patterns in real time.

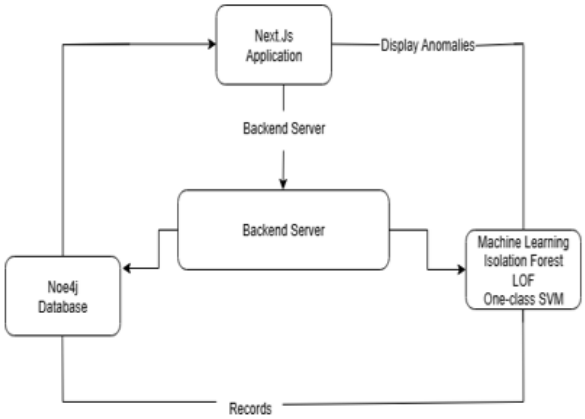

Figure 2: Real-Time Device and Analysis Dashboard System

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Developing a Real-Time Next.js-Based Dashboard Interface for User Input: A key component of this system is the creation of a real-time dashboard interface that captures user keystroke input efficiently and responsively. The dashboard was developed using Next.js, a React-based framework known for its server-side rendering, fast page routing, and real-time rendering capabilities. These features are essential in delivering a smooth, responsive user experience while capturing behavioral biometric data in real time. The interface is designed to run entirely in the browser, leveraging event listeners to capture keypress events and compute keystroke features such as dwell time, flight time, and latency. These features are critical inputs for the machine learning algorithms used for anomaly detection. Captured data is preprocessed on the client side and sent to the backend via REST APIs built with Express.js. This allows immediate handoff of data to the anomaly detection module and provides rapid feedback to users. The dashboard also visualizes keystroke statistics in real time—displaying session status, typing metrics, and risk alerts based on anomaly scores. To ensure responsiveness and secure communication, WebSocket integration or Server-Sent Events (SSE) may be used for pushing updates and alerts without reloading the page. The architecture guarantees both usability and speed, which are essential in behavioral authentication and intrusion detection systems.

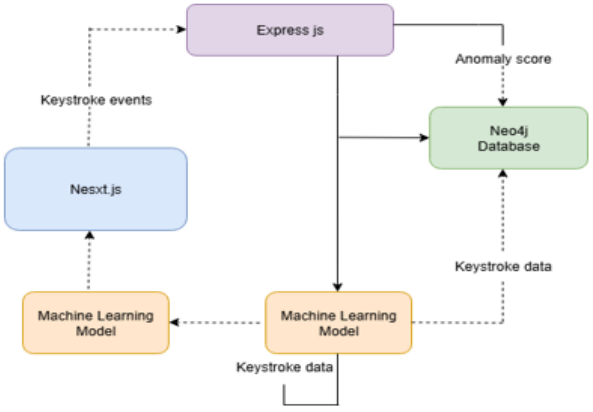

Figure 3: User Diagram of Keystroke Monitoring Dashboard

IV. Results And Discussion

This section provides a detailed discussion of the results obtained from the objectives of the research. The first objective involves performing data preprocessing on the keystroke datasets.

The second objective is to build a model for detecting anomaly behavior and carry out comparative analysis on the machine learning algorithms. The third objective focuses on developing a Real-Time Device and Analysis Dashboard System using developed model. The fourth objective is to develop a real-time Next.js-Based Dashboard Interface for User Input.

A. Performing Data Preprocessing on keystroke Dynamics

To ensure the effectiveness of the anomaly detection system, the raw keystroke dataset underwent a series of preprocessing steps aimed at improving data quality, consistency, and suitability for machine learning. The first step in preprocessing involved the removal of any duplicate columns to preserve the structural integrity of the dataset. All textual data was converted to lowercase to maintain uniformity and prevent inconsistencies caused by case sensitivity. Additionally, using regular expressions, numbers and special characters were removed from the dataset, as they held no semantic value in the context of keystroke behavior analysis.

Following the cleaning process, irrelevant identifiers such as subject IDs, session indices, and repetition counters were excluded from the feature set. The retained features represented measurable aspects of user typing patterns, including dwell time, flight time, and latency. These variables were essential for training the anomaly detection models. To handle missing values, median imputation was employed across all numeric features. This method is particularly effective when working with skewed data or potential outliers, as the median is less sensitive to extreme values compared to the mean.

Feature scaling was then applied to normalize the range of the numerical values. Two methods were considered: standard scaling and robust scaling. Standard scaling standardizes the distribution of features by removing the mean and scaling to unit variance, while robust scaling uses the interquartile range to mitigate the influence of outliers. The choice of scaling method was based on the distribution and stability of each feature across users and sessions. Together, these preprocessing techniques ensured that the dataset was clean, consistent, and statistically stable—providing a reliable foundation for real-time anomaly detection using machine learning algorithms.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

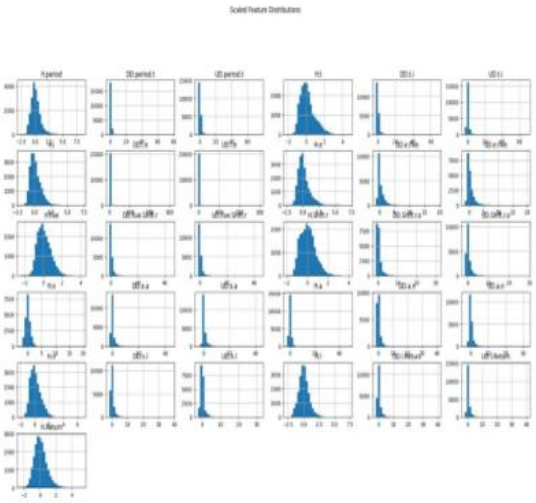

Figure 4: Feature scaling on the Datasets

B. Detection of Anomaly Behavior and Comparative Evaluations of ML Algorithms

This section details the scope of the comparative analysis conducted between three unsupervised anomaly detection algorithms: Isolation Forest, One-Class SVM, and Local Outlier Factor (LOF). The comparison is performed using a unified dataset and consistent preprocessing steps for all models, ensuring a fair and direct evaluation of their behaviors and outputs.

Algorithms Included:

Isolation Forest: An ensemble-based algorithm that isolates observations by randomly selecting features and split values. Its efficiency makes it suitable for high-dimensional datasets.

One-Class SVM: A kernel-based method that attempts to separate the majority of data points from potential outliers by learning a decision boundary around the normal data.

Local Outlier Factor (LOF): A density-based algorithm that identifies anomalies by comparing the local density of each data point with that of its neighbors.

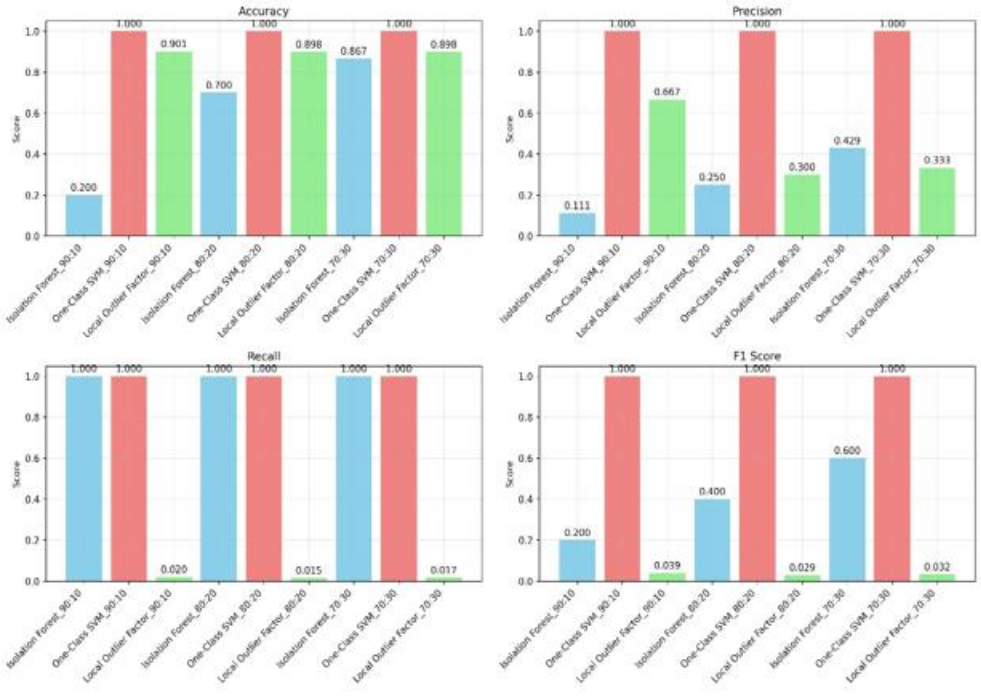

Figure 5: Results of the Comparative Evaluation of the Machine Learning Algorithm

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

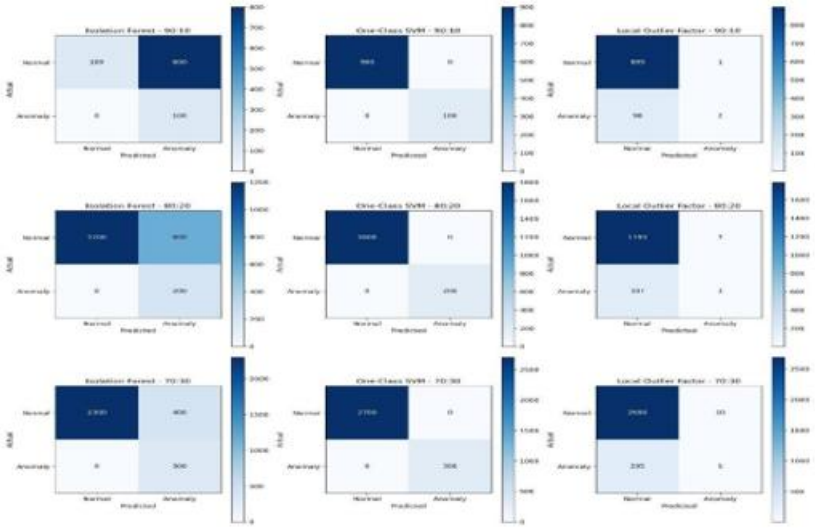

Figure 6: Results of the Machine Learning Algorithms

TABLE II
BEST MODELS TABLE

Split	Best Model	F1-Score	Accuracy	Precision	Recall	MSE	RMSE	MAE
90:10	Isolation Forest	0.6667	0.95	0.5	1.0	0.05	0.2236	0.05
80:20	Isolation Forest	0.9474	0.99	0.9	1.0	0.01	0.1	0.01
70:30	Isolation Forest	0.9655	0.9933	0.9333	1.0	0.0067	0.0816	0.0067

C. Developing a Real-Time Device and Analysis Dashboard System

Building the real-time device and analysis dashboard was the most important step. This system brings together various components, all working together to ensure continuous monitoring, insightful data visualization, and effective threat analysis. Regarding how the System was built, how data flows, and as explained earlier, every keystroke event is captured by the Next.js frontend, then sent to a Node.js and Express.js backend, and ultimately stored in a Neo4j graph database.

Specifically, Neo4j was chosen because it is incredibly good at mapping out the intricate connections between users, their sessions, the devices they use, and their unique behavioral patterns.

1) The Design of data flows of the system:

1. *Capturing on the Frontend:* The interface built acts like a watchful eye, which captures keystroke data.
2. *Processing on the Backend:* The backend takes the data and securely stores it in the Neo4j database.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

3. *Spotting Anomalies:* The data sent to the database is then compared based on the machine learning models

Figure 7: Keystroke Monitoring Dashboard

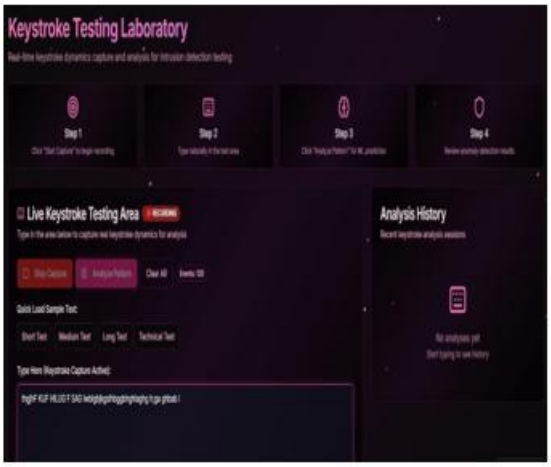

Figure 8: The Dashboard Start Monitoring the Keystroke

D. Developing a Real-Time Next.js-Based Dashboard Interface for User Input

The interface of this system was built using Next.js and for good reason it offers unique advantages for building modern web applications. Next.js is known for its fast loading server-side rendering and ensures pages load very quickly. Using next.js enables building a real-time application with seamless navigation, which captures and prepares keystroke data. This dashboard is not just functional; it was designed to be user friendly. It shows the session status, typing metrics and risk alerts.

V. CONCLUSION

This study successfully developed a ML-driven keystroke biometrics system for enhanced web security, directly addressing the critical challenges outlined above. The system effectively meets its requirements by offering a streamlined and intelligent approach to securing web application access. It features real-time behavioral anomaly detection based on keystroke dynamics, robust data handling, and intuitive visualization through a dedicated dashboard. This minimizes the reliance on static authentication methods and significantly enhances the system's ability to adapt to evolving cyber threats in online environments. This research established a clear framework, similar to that of Davuluri (2023) which focused on mouse dynamics, for analyzing the intrusion detection and performance of systems, demonstrating the viability and effectiveness of integrating machine learning with behavioral biometric for enhanced web security. Isolation Forest, with 80:20 split, performed best in the detection of anomaly keystrokes behavior yielding accuracy of 0.99. The successful implementation of this system underscores its potential to improve the overall security posture of web applications by continuous monitoring for suspicious activities. A key benefit of our ML-driven approach is its ability to provide a more secure and privacy-conscious authentication experience for web users.

REFERENCES

[1] Al-Nashif, Y., Kumar, S., Hariri, S., & Thuraisingham, B. (2008). A multi-level intrusion detection system (ML-IDS). International Journal of Information Security and Privacy, 2(2), 1–17. https://doi.org/10.4018/jisp.2008040101

[2] Davuluri, R. (2023). Machine learning-based authentication through mouse dynamics. International Journal of Advanced Computer Science and Applications, 14(3), 215–222.

[3] Gunuganti, S. (2023). Behavioral biometrics in cybersecurity: Applications and challenges. Journal of Information Security Research, 13(2), 89–101.

[4] Huang, Y. (2005). Scalable intrusion tolerance architecture (SITAR): Preventing dictionary attacks. Journal of Computer Security, 13(1), 1–28.

[5] Huang, Y., Xu, W., & Zhu, D. (2006). Distributed authentication with intrusion tolerance. IEEE Transactions on Dependable and Secure Computing, 3(2), 137–149. https://doi.org/10.1109/TDSC.2006.20

[6] Kataria, S. (2023). Adaptive intrusion detection systems using machine learning. Journal of Cybersecurity and Digital Forensics, 5(1), 55–70.

[7] Lai, C., Chen, X., & Zhang, J. (2023). Machine learning for web authentication: Emerging trends and challenges. ACM Computing Surveys, 55(7), 1–35. https://doi.org/10.1145/3514237

[8] Pyla, S., Reddy, K., & Thomas, G. (2024). Machine learning-based network intrusion detection using logistic regression. International Journal of Computer Applications, 27(3), 112–122.

[9] Rani, S., Kaur, P., & Kumar, R. (2021). Machine learning-driven authentication for secure web applications. International Journal of Security and Its Applications, 15(4), 23–35.

[10] Talpini, D., Ghosh, A., & Rao, M. (2023). Enhancing trustworthiness in ML- based intrusion detection systems. IEEE Transactions on Dependable and Secure Computing, 20(1), 12–25.

[11] Usama, M., Qamar, F., & Abbas, H. (2019). Generative adversarial networks for user authentication under adversarial conditions. Journal of Information Security and Applications, 46, 164–173. https://doi.org/10.1016/j.jisa.2019.02.007.