SRI SIDDHARTHA ACADEMY OF HIGHER EDUCATION

(*Declared as Deemed to be University Under Section 3 of the UGC Act, 1956*
Approved by AICTE, Accredited by NBA, NAAC 'A+' Grade)
AGALKOTE, TUMAKURU – 572107
KARNATAKA

A Project Report
On
"LEGAL DOCUMENT CLASSIFICATION"

Submitted in fulfillment of requirements for the award of degree

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by

B R MANOJ KUMAR (21CS011)

Under the guidance of

Mr. CHANNABASAVARAJU T P
Assistant Professor,
Dept. of CS&E SSIT,
Tumakuru – 572105

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY

(A Constituent College of Sri Siddhartha Academy of Higher Education,
Approved by AICTE, Accredited by NBA, NAAC 'A+' Grade)
MARALUR, TUMAKURU-572105

2024-2025

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY

(A Constituent College of Sri Siddhartha Academy of Higher Education, Tumakuru,
Approved by AICTE, accredited by NBA, NAAC 'A+' Grade)
MARALUR, TUMAKURU – 572105, KARNATAKA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled ***"LEGAL DOCUMENT CLASSIFICATION"*** *is a bonafide work carried out by* ***B R MANOJ KUMAR*** *in fulfillment for the award of degree of* ***Bachelor of Engineering*** *in* ***Computer Science and Engineering*** *during the academic year* ***2024-25****.*

It is certified that all the corrections/suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the degree of Bachelor of Engineering in Computer science and Engineering.

Signature of the guide	Signature of the HOD	Signature of the Principal
Mr. Channabasavaraju T P	**Dr. Raviram V**	**Dr. M.S.Raviprakasha**
Assistant Professor, Dept. of CSE	Professor & Head, Dept of CSE	Principal,
SSIT, Tumakuru	SSIT, Tumakuru	SSIT,Tumakuru

Project associates	USN No.
1) B R MANOJ KUMAR	(21CS011)

External Examiners	Signature with Date
1. _____	_____
_____	_____
2.	

ACKNOWLEDGEMENT

At the outset we express our most sincere grateful thanks to holy sanctum of **"SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY"**, the temple of learning, for giving us an opportunity to pursue the degree course in Computer Science and Engineering thus help shaping our career.

We wish to express our sincere gratitude to **Dr. M. S. RAVIPRAKASHA** Principal, Sri Siddhartha Institute of Technology, for providing us an excellent academic environment, which has nurtured our practical skills, and for kindly obliging our requests and providing timely assistance.

We are highly grateful to **Dr. RAVIRAM V,** Professor and Head of the Department of Computer Science and Engineering, for patronizing us throughout the project and for his encouragement and moral support.

We also wish to express our deep sense of gratitude to our project guide **Mr. CHANNABASAVARAJU T P,** Assistant Professor, Department of Computer Science and Engineering, for her valuable suggestions, guidance, moral support and encouragement in the completion of this project successfully. We have been fortunate for having her precious help.

Finally, we express our gratitude to all the teaching and non-teaching staff of the **Computer Science and Engineering Department** for their timely support and suggestions.

We are also thankful to our parents for their moral support and constant guidance made our efforts fruitful. Last but not the least, our friends, without their constructive criticisms and suggestions, we wouldn't have been able to complete successfully.

PROJECT ASSOCIATES

B R MANOJ KUMAR (21CS011)

DECLARATION

We here by declare that this project work entitled ***"LEGAL DOCUMENT CLASSIFICATION"*** is an original and bonafide work carried out by us at **SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY,** Tumakuru, in partial fulfillment of **BACHELOR OF ENGINEERING** in **Computer Science and Engineering**.

We also declare that, to the best of our knowledge and belief, The work reported here in does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion by any student.

Date: B R MANOJ KUMAR (21CS011)

Place:

ABSTRACT

The visually impaired face significant challenges in daily life, requiring innovative solutions to bridge the gap between accessibility and independence. This project introduces an assist to the blind with multiple functionalities: object recognition, known face recognition, sign language detection, currency recognition, and fire detection. By leveraging Python and Deep Learning models. The application provides real-time assistance with audio feedback, fostering greater independence and safety for the visually impaired.

The object recognition module identifies and announces objects within the user's surroundings, while the face recognition system detects known individuals and provides their names via voice output. The sign language detection feature interprets hand gestures into readable text, assisting in communication with hearing-impaired individuals. Additionally, the currency recognition system ensures accurate identification of bank notes to aid in financial transactions, and the fire detection module promptly alerts users to potential hazards using colour-based filtering techniques.

A user-friendly interface and real-time processing ensure smooth and efficient interactions. The project will allow for scalability and ease of integration with hardware devices such as cameras and microphones. By combining multiple assistive technologies into a single platform, this system empowers visually impaired individuals to navigate their environment safely and independently.

CONTENTS

LIST OF FIGURES

CHAPTER – 1

INTRODUCTION

Blind assistance technologies have made remarkable strides in recent years, yet many available solutions still offer limited functionality, often addressing only one aspect of visual impairment. Standalone tools like object detection or text-to-speech converters, while useful, fail to provide a comprehensive solution to the complex and varied needs of visually impaired individuals. These systems frequently lack the adaptability and integration required for real-world scenarios, resulting in fragmented user experiences and reduced usability. As a result, users often need to switch between multiple applications or devices to complete everyday tasks, which can be inefficient and overwhelming.

This project suggests a single platform that integrates several assistive functions into one, Python-driven application with deep learning models. The system integrates modules for object recognition, face recognition, sign language detection, currency recognition, and fire hazard detection. Each of these functionalities deals with a particular area of the visually impaired users' challenges. For example, object recognition assists users in understanding the immediate surroundings by recognising items nearby, while face recognition supports social interaction by allowing the recognition of known persons. Sign language detection supports communication by converting gestures into readable text, facilitating engagement with hearing-impaired individuals who use sign language. Moreover, currency recognition empowers users to manage their finances confidently by identifying different denominations accurately. The fire detection feature significantly enhances personal safety by identifying hazardous situations and issuing real-time alerts. Integrating all these functionalities into a single application streamlines the user experience and promotes greater autonomy for visually impaired individuals.

This holistic approach not only reduces the dependency on multiple devices or tools but also brings forward a more inclusive, practical, and scalable solution for assistive technology. By leveraging Python and deep learning, this project aspires to bridge the accessibility gap and contribute meaningfully to the daily lives of those with visual impairments.

1.1 PROBLEM STATEMENT

EXISTING PROBLEM:

There are many stand-alone applications of assistive technologies, which have been developed to solve particular issues experienced by people with visual impairments. These consist of equipment for identifying currencies, detecting fire hazards, translating sign language, detecting objects, and recognising faces. While face recognition encourages social interaction by identifying friends and relatives, object detection aids users in navigating their environment by detecting objects in their immediate surroundings. All of these technologies provide vital assistance in improving day-to-day living. Currency identification promotes financial independence, sign language translation helps close communication gaps, and fire detection keeps people safe by warning of possible dangers. However, because these tools are individualised, they are frequently only capable of one function, which limits their total usefulness in complicated real-world situations.

The primary flaw in these stand-alone systems is that they are not integrated. Depending on the work at hand, visually impaired users must switch between applications, which can be inconvenient and ineffective, particularly when multitasking. This disjointed experience hinders the broad use of assistive technology in addition to impairing usability.

PROPOSED SOLUTION:

PROPOSED SOLUTION:

The proposed system aims to significantly enhance web application security by leveraging the power of Machine Learning (ML) to detect SQL Injection (SQLi) and Cross-Site Scripting (XSS) attacks. Traditional firewalls, which often rely on static rules

and pre-defined signatures, can struggle to keep pace with the sophisticated and evolving nature of modern cyber threats. By contrast, an ML-based approach involves training models on extensive datasets containing both malicious and benign traffic. This allows the system to learn and recognize the subtle characteristics and patterns associated with SQLi and XSS attacks, which might be overlooked by conventional methods. Through this process, the system becomes adept at distinguishing between normal and malicious activities, enhancing its overall detection accuracy.

Once deployed, the ML-based firewall operates continuously, monitoring all incoming web traffic in real-time. It inspects various components of HTTP requests, including URLs, form data, and HTTP headers, looking for any signs of malicious behavior. This comprehensive analysis helps in identifying SQLi attempts, which involve injecting harmful SQL queries into web forms, as well as XSS attacks that aim to execute malicious scripts in the victim's browser. The system's ability to analyze and understand the context of these components allows it to detect and block even the most sophisticated attack vectors, providing a robust defense mechanism for web applications.

In addition to its detection capabilities, the ML-based firewall enhances security through dynamic and adaptive responses. When a potential threat is detected, the system can take immediate action by blocking the malicious request, thereby preventing the attack from reaching the application. Furthermore, it logs each incident, providing valuable data for further analysis and continuous improvement of the security measures. This logging is crucial for understanding the nature of the threats and for refining the ML models to improve future detection rates. By offering an adaptive and proactive security solution, the proposed system ensures that web applications remain protected against an ever-evolving landscape of cyber threats, significantly reducing the risk of data breaches and other security incidents.

1.1. MOTIVATION

1. **Increasing Sophistication of Cyber Attacks**: The complexity and frequency of cyber-attacks, particularly SQL Injection (SQLi) and Cross-Site Scripting (XSS), have been steadily increasing. Traditional security measures are often inadequate in detecting these sophisticated threats, necessitating the development of advanced solutions that can adapt to and counteract evolving attack strategies.

2. **Limitations of Traditional Firewalls:** Conventional firewalls rely on static rules and signature-based detection, which can be easily bypassed by modern attackers. This motivates the need for a more dynamic and intelligent approach to threat detection, one that can learn from data and identify subtle, previously unknown attack patterns that static methods miss.

3. **Advancements in Machine Learning:** The rapid advancements in machine learning technologies provide a promising avenue for enhancing cybersecurity. ML models can be trained to recognize complex patterns in large datasets, making them highly effective at detecting sophisticated cyber threats. This potential drives the motivation to integrate ML into firewall systems for improved security.

4. **Demand for Real-Time Threat Detection:** In today's fast-paced digital environment, real-time threat detection and response are crucial. Traditional systems often lag in identifying and mitigating threats, leaving vulnerabilities exposed. The motivation is to develop a system that can analyze traffic in real-time, provide immediate protection, and adapt to new threats as they emerge, ensuring robust and continuous security for web applications.
5. **Increase in Web Application Vulnerabilities:** With the proliferation of web applications and their increasing complexity, vulnerabilities are becoming more prevalent. These vulnerabilities, such as insecure deserialization, improper access control, and server-side request forgery (SSRF), are prime targets for attackers seeking to exploit weaknesses in application logic rather than traditional network vulnerabilities.

1.2. OBJECTIVES

1. Enhanced Threat Detection:

The primary objective of the proposed system is to improve the detection of cyber threats such as SQL Injection (SQLi) and Cross-Site Scripting (XSS) attacks. By leveraging machine learning algorithms trained on large datasets, the system can recognize subtle and complex attack patterns that traditional firewalls often miss. This enhanced detection capability ensures a higher level of protection against sophisticated and evolving cyber threats.

2. Automated Threat Response:

Another key objective is to automate the response to detected threats. When the system identifies a potential SQLi or XSS attack, it can immediately take action to block the malicious request. This automation reduces response times, minimizes the risk of successful attacks, and frees up security personnel to focus on other critical tasks, enhancing the overall efficiency of the security operations.

3. Improved Security Posture:

The goal is to strengthen the overall security posture of web applications. By continuously monitoring and analyzing incoming traffic, the ML-based firewall not only detects and mitigates current threats but also adapts to new attack vectors. This dynamic and proactive approach ensures that the security measures evolve alongside the threats, maintaining robust protection and reducing the likelihood of data breaches and other security incidents.

CHAPTER – 2

THEORY AND CONCEPT

2.1. SOFTWARE REQUIREMENTS

1. Python (version 3.x):

Python's cross-platform compatibility ensures seamless deployment of your IDS and IPS across different operating systems, whether on Linux servers or Windows environments. Its integration capabilities with other languages and frameworks enable easy incorporation of database connections, visualization tools, and real-time monitoring functionalities, essential for enhancing network security resilience and operational efficiency.

In conclusion, Python empowers your project with agility and scalability, enabling rapid prototyping, efficient algorithm implementation, and seamless integration of advanced machine learning techniques. Its vibrant ecosystem and active community support ensure continuous innovation and adaptability, making Python indispensable for developing cutting-edge cybersecurity solutions like your IDS and IPS.

2. Flask:

Flask, a lightweight and flexible web framework for Python, is utilized for building the web-based interface and APIs necessary for your IDS and IPS. Flask's simplicity and extensibility make it ideal for developing interactive dashboards, RESTful APIs, and real-time monitoring tools that administrators can use to manage and analyze network activities, detect intrusions, and assess system performance.

Furthermore, Flask's open-source nature and active community contribute to its flexibility and scalability. Developers can extend Flask's functionalities through plugins and third-party integrations, ensuring that your IDS and IPS framework evolves with the changing cybersecurity landscape. By adopting Flask, your project gains a powerful framework for creating responsive, data-driven applications that bridge machine learning insights with actionable cybersecurity strategies.

3. Matplotlib, Seaborn:

Incorporating Matplotlib and Seaborn into the project will enhance the visualization and analysis of security data, making it easier to understand trends, identify patterns, and communicate findings effectively. Here's how these libraries can be utilized:

1. Visualizing Traffic Patterns:

Matplotlib: Use Matplotlib to create line charts, bar graphs, and scatter plots to visualize the volume and types of incoming traffic over time. This helps in understanding normal traffic behavior and identifying anomalies that may indicate potential threats.

2. Analyzing Attack Trends:

Seaborn: Leverage Seaborn to create more sophisticated visualizations like heatmaps, box plots, and violin plots to analyze trends in detected attacks. These visualizations can highlight periods of high attack activity, common attack vectors, and the distribution of attack severities.

3. Evaluating Model Performance:

Matplotlib: Use Matplotlib to plot ROC curves, precision-recall curves, and confusion matrices to evaluate the performance of the ML models. These plots are crucial for assessing how well the models detect and classify SQLi and XSS attacks.

4. React:

React has revolutionized front-end development with its component-based architecture and efficient rendering capabilities. At its core, React allows developers to break down user interfaces into reusable components, each managing its own state and rendering logic. This approach not only promotes code reusability and maintainability but also enhances developer productivity by providing a clear, modular structure to applications. By leveraging JSX, React enables developers to write HTML-like code directly within JavaScript, streamlining the process of building complex UIs and integrating dynamic content seamlessly.

Central to React's performance is its virtual DOM implementation, which minimizes DOM manipulation overhead. React's virtual DOM compares the current state of the UI with the previous state, updating only the necessary components in the actual DOM.

5. PyTorch:

Pytorch is a leading open-source machine learning framework known for its dynamic computation graph, which enables developers to build and train neural networks with flexibility and efficiency. Developed by Facebook's AI Research lab (FAIR), PyTorch offers automatic differentiation through its `autograd` package, facilitating gradient-based optimization for model training.

Its seamless integration with CUDA and cuDNN allows for accelerated computations on GPUs, making it suitable for handling large-scale datasets and complex deep learning models. PyTorch's extensive library of modules and utilities simplifies the process of constructing neural networks, from defining layers to implementing sophisticated algorithms in areas like computer vision, natural language processing, and reinforcement learning. With a vibrant community and robust ecosystem, PyTorch continues to innovate and empower researchers and developers worldwide in advancing AI-driven solutions across various domains.

6. Transformers:

Transformers, a groundbreaking deep learning model architecture introduced by Vaswani et al. in 2017, revolutionized natural language processing (NLP) tasks by leveraging attention mechanisms. Unlike previous sequential models, Transformers process entire sequences of tokens simultaneously using self-attention, enabling them to capture global dependencies and context efficiently. This architecture has become synonymous with state-of-the-art performance in various NLP benchmarks, including machine translation, text generation, sentiment analysis, and more.

Transformers excel in handling long-range dependencies and understanding nuanced relationships within text data, making them versatile for tasks requiring contextual understanding and generating coherent responses. With implementations in frameworks like PyTorch and TensorFlow, Transformers have spurred advancements in pre-trained models like BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), which have become foundational in both research and industrial applications, driving the evolution of natural language understanding and generation capabilities across diverse domains.

CHAPTER – 3

METHODOLOGY

1.1. BLOCK DIAGRAM

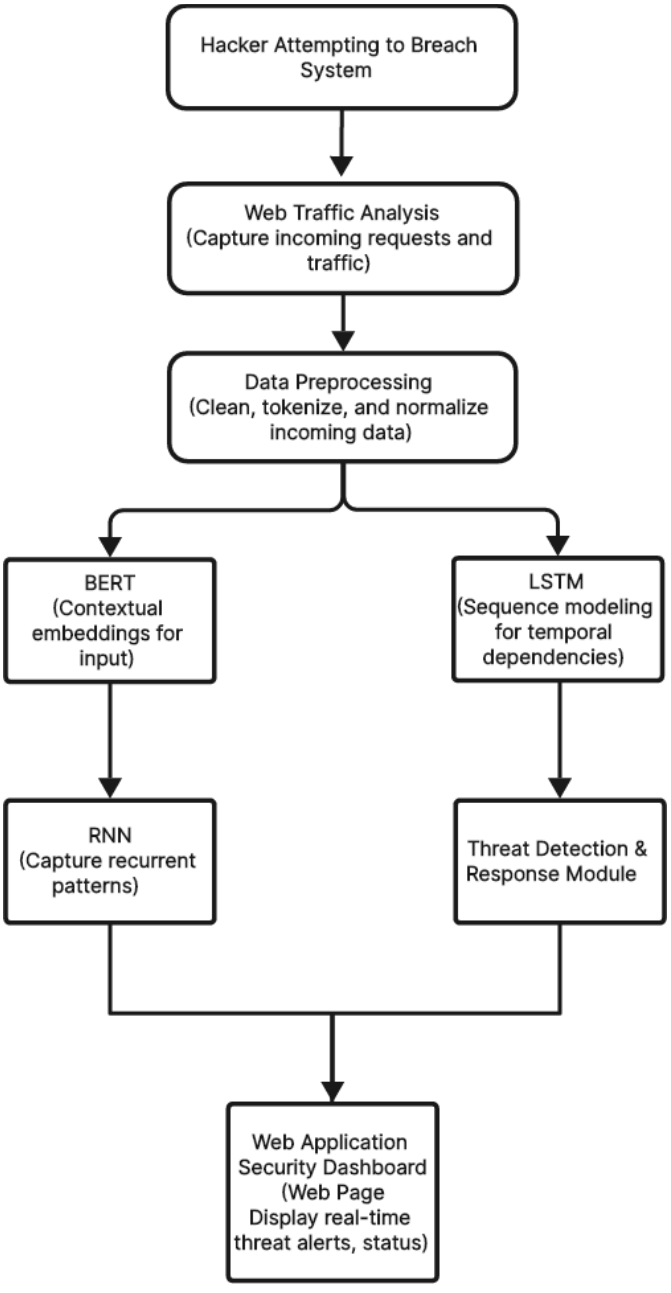


Fig. 3.1 : Block Diagram Illustrating an Firewall Detection System

The fig 3.1 The depicted process outlines a sophisticated approach to fortifying web-based systems against cyber threats, commencing with real-time web traffic analysis. This initial stage involves capturing and scrutinizing incoming requests and traffic patterns, pivotal for identifying potential security breaches or malicious activities attempting to infiltrate the system. Subsequently, the data undergoes meticulous preprocessing, encompassing cleaning, tokenization, and normalization. These preparatory steps ensure that the data is standardized and structured uniformly, optimizing its suitability for subsequent machine learning analyses.

The methodology integrates advanced AI models, notably BERT (Bidirectional Encoder Representations from Transformers) and LSTM (Long Short-Term Memory). BERT enhances the system's capability to comprehend contextual embeddings from textual inputs, such as deciphering the intent behind web requests or logs. Concurrently, LSTM models temporal dependencies within the data, crucial for detecting sequences of actions indicative of potential cyber intrusions over time. Complementing these, Recurrent Neural Networks (RNNs) are employed to recognize recurring patterns within the data, further enhancing the system's ability to discern complex attack vectors.

The culmination of this process lies in the Threat Detection & Response Module, where insights derived from BERT, LSTM, and RNN analyses are leveraged to detect specific threats like SQL injection (SQLi), Cross-Site Scripting (XSS), and other sophisticated attack methodologies. Automated response mechanisms are then triggered based on the severity and nature of identified threats. This methodology integrates visualization tools for generating charts, graphs, and reports that provide security administrators with actionable insights and real-time threat alerts. Ultimately, these efforts are integrated into a comprehensive web application security dashboard or webpage, ensuring continuous monitoring and proactive management of cybersecurity threats to safeguard digital infrastructures effectively.

3.2 METHODOLOGY

Text classification is a fundamental task in Natural Language Processing (NLP) with applications ranging from sentiment analysis to spam detection. In this project, we explore various approaches to text classification using Recurrent Neural Networks (RNNs), Bidirectional Encoder Representations from Transformers (BERT), and Long Short-Term Memory networks (LSTMs).

1. Data Collection

We sourced datasets from Kaggle and GitHub for this project. These datasets contain text data labeled for classification tasks such as sentiment analysis.

Kaggle Dataset:

We downloaded a text classification dataset from Kaggle, which includes text data and corresponding labels. The data was read into a pandas Data Frame for further processing.

GitHub Dataset:

Additionally, we obtained a dataset from a GitHub repository, which was also loaded into a pandas Data Frame. This dataset was combined with the Kaggle dataset to create a more comprehensive dataset for our analysis.

2. Data Preprocessing

Preprocessing is a crucial step in preparing the text data for modeling. The following common preprocessing steps were applied:

Text Cleaning

Remove Punctuation: All punctuation marks were removed to reduce noise in the text data.

Lowercasing: All text was converted to lowercase to ensure uniformity.

Tokenization: The text was split into individual words (tokens) using the NLTK library.

Stopword Removal: Commonly used words that do not carry significant meaning (stopwords) were removed from the text.

3. Text Representation

Tokenization: The cleaned text data was tokenized into sequences of words.

Padding: The sequences were padded to ensure that all input sequences have the same length, which is required for neural network models.

Encoding: The text sequences were encoded into numerical values using a tokenizer, which maps each word to a unique integer.

4. RNN Model

We built a Recurrent Neural Network (RNN) model for text classification. The RNN model consists of an embedding layer, a Simple RNN layer, and a dense output layer with a sigmoid activation function. The model was compiled with the Adam optimizer and binary cross-entropy loss function and trained on the preprocessed text data.

5. BERT Model

To leverage the power of transformer-based models, we used a pre-trained BERT model from the Hugging Face library for text classification. The BERT tokenizer was used to tokenize and encode the text data. The BERT model was fine-tuned on our dataset with an Adam optimizer and a sparse categorical cross-entropy loss function. The training process involved passing the tokenized text and attention masks to the model.

6. LSTM Model

We also implemented a Long Short-Term Memory (LSTM) model for text classification. The LSTM model architecture included an embedding layer, an LSTM layer, and a dense output layer with a sigmoid activation function. Similar to the RNN model, the LSTM model was compiled with the Adam optimizer and binary cross-entropy loss function. The model was trained on preprocessed text data.

In this project, we explored various approaches to text classification using different neural network architectures. The RNN model provided a simple yet effective method for sequence modeling, while the BERT model leveraged advanced transformer techniques for improved performance. The LSTM model offered a robust solution for handling long-term dependencies in text data. Each approach has its strengths and can be chosen based on the specific requirements and constraints of the task at hand.

CHAPTER – 4

DESIGN

4.1. WEB APPLICATION FIREWALL DESIGN

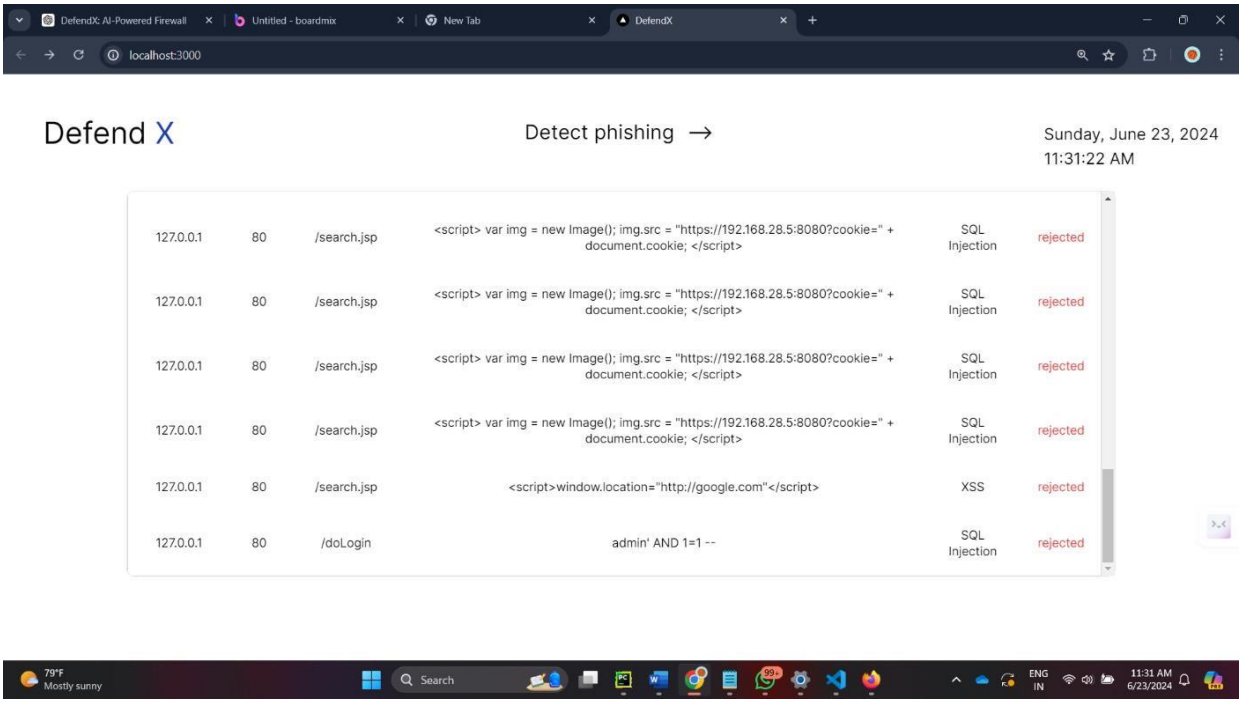


Fig. 4.1 : UI Design for Web Application Firewall

Our project involves a Web Application Firewall (WAF) designed to protect websites from various cyber threats. The WAF intercepts user actions such as searching or logging in, extracts required values, and uses a trained machine learning model to analyze these actions. Additionally, the system includes a phishing detection subsystem where users can input a URL to determine if it is a phishing attempt. The model used for both functionalities is RNN BERT-LSTM.

ATTRIBUTES AND FUNCTIONALITIES

Description: When a user performs an action on the website, such as searching or logging in, the firewall proxy server captures the action and extracts the necessary values for analysis.

Functionality:

- The system monitors user actions in real-time.
- Extracted data is passed to the machine learning model for analysis.
- The model evaluates the data and determines if the action is potentially malicious.

Phishing Detection Subsystem

Description: Allows users to input a URL to check if it is a phishing site.

Model Used: RNN BERT-LSTM

Functionality:

- Users enter a URL in the designated input field.
- The system analyzes the URL using the RNN BERT-LSTM model.
- The model outputs whether the URL is a phishing attempt or not.

UI DESIGN OVERVIEW

1. User Action Monitoring Interface:

Real-time Action Log: A section displaying user actions as they are performed on the website.

Action Details: Detailed information about each captured action, including extracted values and analysis results.

Alert System: Notifications or alerts for actions identified as potentially malicious.

2. Phishing Detection Interface:

- URL Input Field: A text box where users can input a URL to be checked.
- Check Button: A button to submit the URL for analysis.
- Result Display: A section that shows the result of the phishing detection analysis.
- Example User Interaction Flow

3. User Action Monitoring:

- A user logs into the website or performs a search.
- The WAF proxy server captures the action and extracts relevant values.
- The system analyzes the extracted data using the RNN BERT-LSTM model.
- If the action is potentially malicious, an alert is displayed in the real-time action log.

4.2. PHISHING DETECTION DESIGN

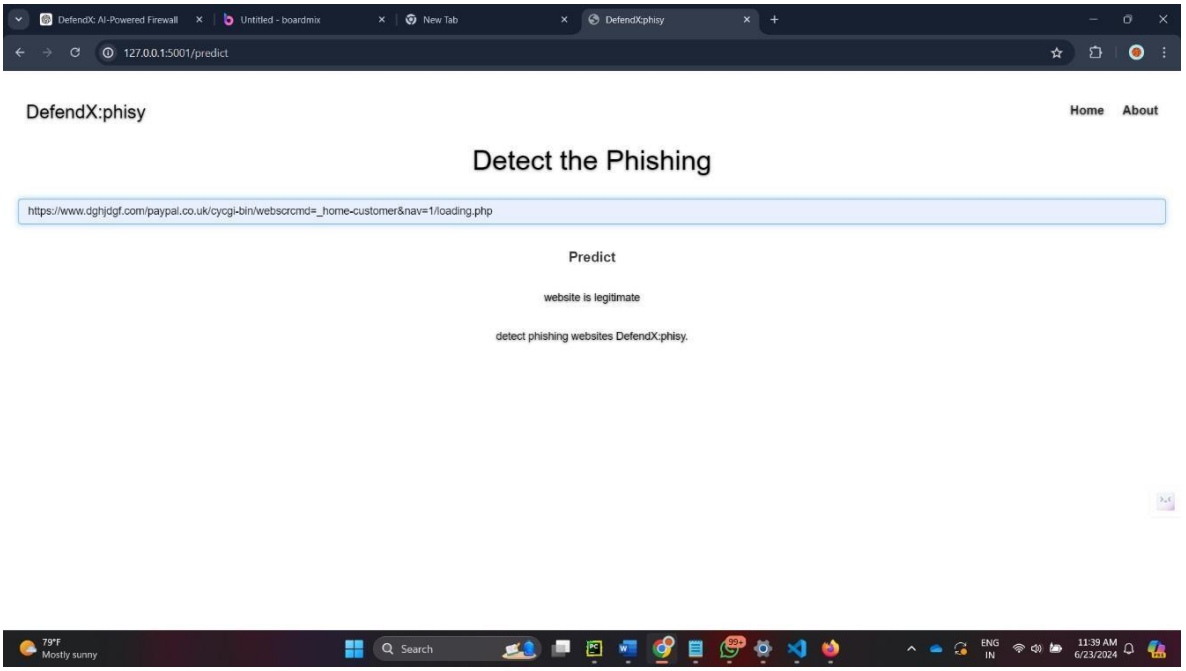


Fig. 4.2 : UI design and working phishing detection

Phishing detection involves identifying and mitigating fraudulent attempts to obtain sensitive information, such as passwords, credit card numbers, or personal data, by masquerading as a trustworthy entity in electronic communications. Here's an overview of the approaches and techniques commonly used for phishing detection:

TECHNIQUES FOR PHISHING DETECTION

1. Feature-based Approaches:

- URL Analysis: Analyzing URLs for suspicious patterns, domain reputation, and similarity to known phishing domains. Techniques include lexical analysis, domain reputation scoring, and blacklisting.
- Content Analysis: Examining email content or web page content for phishing indicators such as misspelled words, unusual URLs, or requests for sensitive information.
- Sender Analysis: Verifying the authenticity of the sender by examining email headers, sender domain, and sender reputation.

2. Machine Learning (ML) Approaches:

- Supervised Learning: Training models on labeled datasets to classify emails or URLs as phishing or legitimate based on features extracted from the content, URLs, and sender information.
- Feature Extraction: Using techniques like natural language processing (NLP) for email content analysis, image analysis for detecting phishing images, and extracting features from URLs.
- Ensemble Methods: Combining predictions from multiple machine learning models (e.g., random forests, gradient boosting machines) to improve detection accuracy.

3. Behavioral Analysis:

- User Behavior Monitoring: Analyzing user interactions with emails or websites to detect deviations from normal behavior patterns, such as sudden changes in click rates or submission of sensitive information.
- Anomaly Detection: Using statistical methods or machine learning algorithms to identify unusual patterns in network traffic or user interactions that may indicate phishing attacks.

CHALLENGES IN PHISHING DETECTION

- Evolving Tactics: Phishers continually adapt their tactics, making it challenging to detect new and sophisticated phishing attempts.
- False Positives: Striking a balance between high detection rates and minimizing false positives, which can impact user experience and operational efficiency.
- Data Quality: Ensuring high-quality training data and adapting models to handle variations in phishing techniques.
- Real-time Detection: Performing detection in real-time to prevent users from falling victim to phishing attacks promptly.

IMPLEMENTATION OVERVIEW

Backend:

- Proxy Server: Captures user actions and forwards extracted data to the ML model.
- ML Model: RNN BERT-LSTM model for analyzing user actions and phishing URLs.

Frontend:

- User Action Monitoring: Interface for real-time logging and alerting of user actions.
- Phishing Detection: Input field and result display for URL analysis.
- Support Chat Bot: Interface for user queries and support.

This interface design ensures comprehensive protection for the website, offering real-time monitoring of user actions and an efficient phishing detection system. The inclusion of user support enhances the overall user experience, making the Web Application Firewall both effective and user-friendly.

4.3. MODELS:

4.3.1. RNN MODELS:

1. INPUT:

HTTP Request Sequences: These sequences consist of headers, parameters, payloads, and metadata extracted from incoming HTTP requests. Each request sequence is treated as a sequence of tokens or characters.

2. EMBEDDING LAYER:

Tokenization: Each token in the HTTP request sequences is converted into embeddings to represent the meaning or context of the token within the sequence.

Embedding Matrix: A trainable embedding matrix maps each token to a high-dimensional vector representation, capturing semantic relationships between tokens.

3. RNN ARCHITECTURE:

Sequence Modeling: RNNs are employed to model the temporal dependencies within the HTTP request sequences. Specifically, Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU) variants of RNNs are commonly used due to their ability to capture long-range dependencies and mitigate the vanishing gradient problem.

Hidden States:The RNN's hidden states at each time step encode information about the sequence history up to that point, which helps in understanding the context and flow of the HTTP request data.

4. DETECTION MECHANISM:

Anomaly Detection: The trained RNN model learns normal patterns of HTTP request sequences during the training phase. During inference, it detects deviations or anomalies from these learned patterns, which may indicate potential SQLi or XSS attacks.

Thresholding: Anomaly scores or probabilities computed by the RNN model are compared against a predefined threshold. Requests exceeding this threshold are flagged as suspicious.

5. TRAINING AND OPTIMIZATION:

Dataset Preparation: Train the RNN model using labeled datasets containing sequences of normal HTTP requests and sequences that include SQLi or XSS attacks.

Loss Function: Use loss functions like binary cross-entropy or mean squared error to optimize the model's ability to differentiate between normal and anomalous HTTP request sequences.

Regularization: Apply dropout regularization within the RNN layers to prevent overfitting and improve generalization to unseen attack patterns.

6. INTEGRATION WITH FIREWALL INFRASTRUCTURE:

Real-Time Analysis: Integrate the trained RNN model with the web-based firewall infrastructure to analyze incoming HTTP request sequences in real-time.

Automated Response: Based on the RNN model's detection results, implement automated response mechanisms within the firewall system to mitigate identified threats. This could include blocking suspicious requests or triggering alerts for further investigation.

4.3.2. BERT MODEL:

BERT (Bidirectional Encoder Representations from Transformers) has revolutionized natural language processing (NLP) tasks by leveraging the power of transformer architectures to capture contextual relationships in text data. Originally introduced by Google in 2018, BERT has since become a cornerstone in NLP due to its ability to deeply understand the meaning of words in context, making it particularly suitable for applications where understanding nuanced textual information is crucial, such as web application security.

In the context of enhancing web application security, BERT plays a pivotal role in analyzing and processing various components of incoming web traffic, including URLs, form data, and HTTP headers. Here are several key reasons why BERT is instrumental in this domain:

1. Contextual Understanding: BERT excels in capturing the contextual meaning of words and phrases. This is critical for analyzing web traffic where the intent behind requests and inputs can vary widely. By understanding the context in which certain inputs occur, BERT can better discern between legitimate user interactions and potentially malicious activities such as SQL Injection (SQLi) or Cross-Site Scripting (XSS) attacks.

2. Semantic Similarity: BERT embeddings allow for measuring semantic similarity between different inputs. This capability is valuable in security applications where identifying patterns that resemble known attack vectors or malicious patterns is essential. By comparing current inputs with previously seen attack patterns, BERT helps in early detection and prevention of security threats.

3. Handling Ambiguity: Web traffic often includes ambiguous or convoluted inputs that traditional rule-based systems may struggle to interpret accurately. BERT's bidirectional training enables it to consider the entire context of a word or phrase, mitigating the challenges posed by linguistic ambiguities commonly found in user-generated content on the web.

4. Adaptability and Transfer Learning: BERT models, particularly pre-trained versions such as those available through frameworks like Hugging Face Transformers, can be fine-tuned on domain-specific data.

This adaptability allows security teams to tailor the model to recognize patterns specific to their web application environment, enhancing detection accuracy without the need for extensive manual rule creation.

5. Efficiency in Complex Tasks: BERT's transformer architecture handles complex sequences of data efficiently. This includes parsing and understanding URLs with parameters, analyzing text in form submissions, and scrutinizing HTTP headers for anomalies. Such tasks require deep contextual understanding and BERT's capabilities are well-suited to meet these demands.

6. Continuous Learning and Improvement: By integrating BERT into the security pipeline, the system can continuously learn from new data and adapt to evolving attack techniques. This continuous learning approach ensures that the security measures remain effective against emerging threats without requiring frequent manual updates.

BERT's ability to understand context, handle semantic similarity, and adapt to specific security domains makes it a powerful tool for enhancing web application security. By leveraging BERT within the security infrastructure, organizations can achieve more robust threat detection, faster response times to potential attacks, and overall strengthened cybersecurity posture in the face of evolving cyber threats.

MODELS:

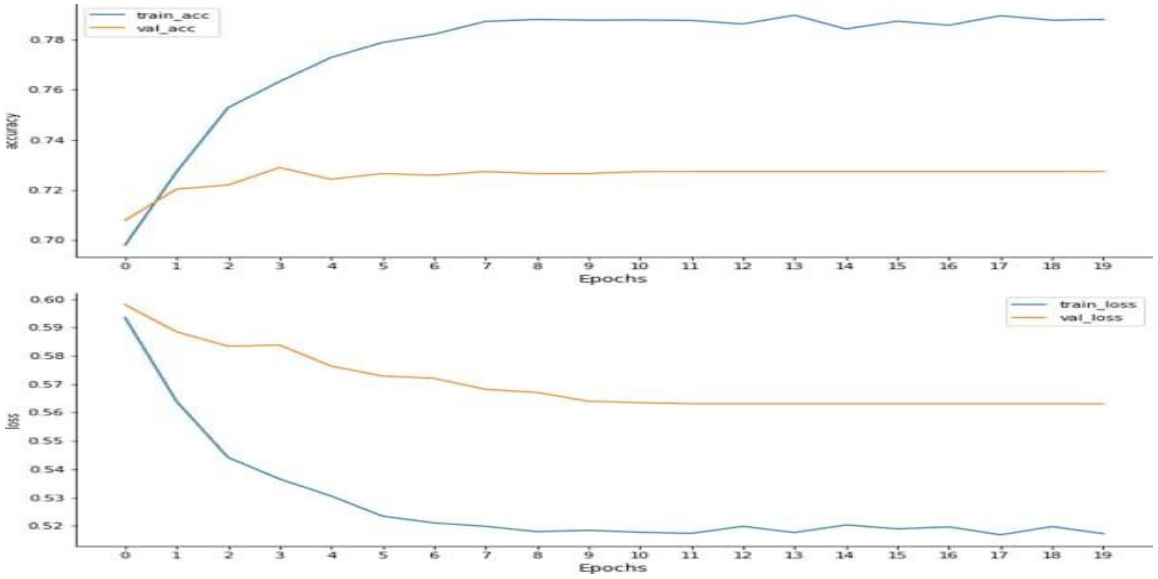


Fig. 4.3.2 : Training Accuracy-Loss Over Batches for BERTLSTM Model

4.3.3. LONG SHORT-TERM MEMORY MODEL:

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) specifically designed to address the challenges of learning and predicting sequences of data with long-term dependencies. In the context of enhancing web application security, LSTM plays a crucial role due to several key capabilities that are beneficial for analyzing and detecting patterns in web traffic data:

1. Sequential Data Modeling: Web traffic often arrives in sequences, such as a series of HTTP requests or interactions with web forms. LSTM networks are well-suited for modeling such sequential data because they can remember information over long periods, capturing dependencies between different elements in the sequence. This capability allows LSTM to effectively analyze the temporal aspects of web traffic and identify abnormal patterns indicative of attacks like SQL Injection (SQLi) or Cross-Site Scripting (XSS).

2. Handling Variable Length Sequences: Unlike traditional feedforward neural networks, LSTM networks can process inputs of variable length. This flexibility is critical in web application security, where the length and complexity of HTTP requests, URLs, or form submissions can vary widely. LSTM's ability to handle variable-length sequences ensures that it can effectively analyze and detect patterns in diverse and complex web traffic data.

3. Learning Contextual Representations: LSTM networks excel at learning contextual representations of input sequences. In the context of security, this means that LSTM can learn to distinguish between normal user interactions and potentially malicious activities based on the context provided by preceding requests or interactions. This contextual understanding is essential for accurate threat detection and minimizing false positives.

4. Capturing Long-Term Dependencies: Security threats in web applications often involve subtle, long-term patterns that may span multiple requests or sessions. LSTM's architecture, which includes memory cells capable of retaining information over extended periods, allows it to capture these long-term dependencies effectively. By remembering and learning from past interactions, LSTM enhances the system's ability to detect sophisticated attacks that evolve over time.

5. Integration with Machine Learning Pipelines: LSTM networks can be seamlessly integrated into machine learning pipelines alongside other models like BERT for comprehensive security analysis. By combining LSTM's sequential modeling capabilities with BERT's contextual understanding, the system can leverage both models' strengths to achieve more robust and accurate threat detection and prevention.

6. Real-Time Processing and Response: LSTM networks are capable of processing data in real-time, making them suitable for applications where timely detection and response to security threats are critical. By analyzing incoming web traffic as it occurs, LSTM enables proactive measures such as blocking suspicious requests or triggering alerts to mitigate potential risks promptly.

LSTM networks are essential in enhancing web application security due to their ability to model sequential data, handle variable-length inputs, learn contextual representations, capture long-term dependencies, integrate with other machine learning models, and enable real-time processing and response. By incorporating LSTM into the security infrastructure, organizations can significantly bolster their defenses against a wide range of cyber threats, safeguarding sensitive data and maintaining the integrity of web applications in dynamic and evolving online environments.

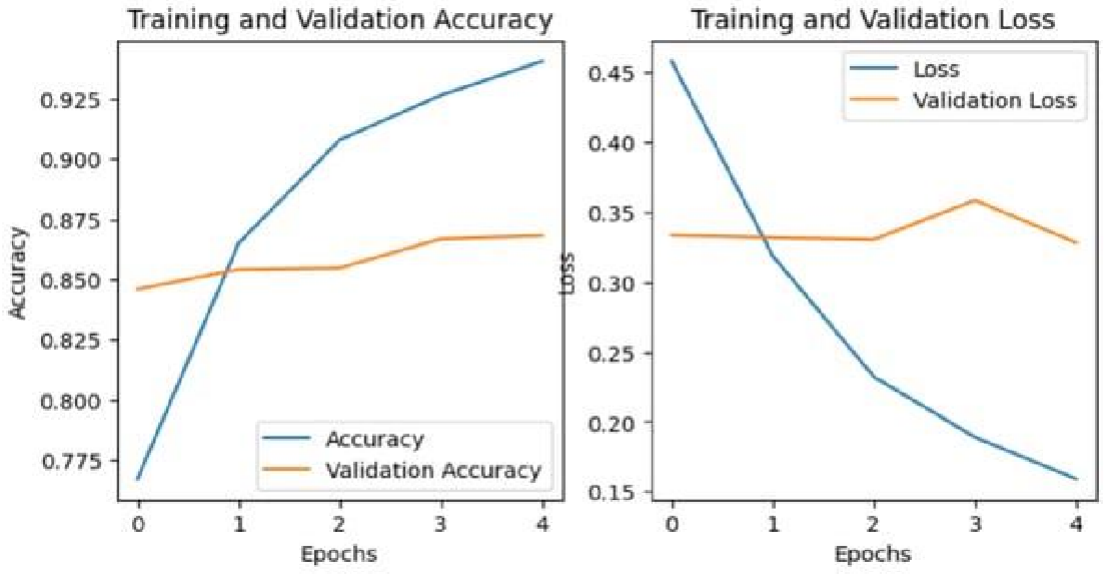


Fig. 4.3.3: Training Accuracy-Loss Over Batches for RNN Model

WHY BERT LSTM MODEL?

Using models like BERT (Bidirectional Encoder Representations from Transformers) and LSTM (Long Short-Term Memory) in a web-based firewall integrated with machine learning offers several advantages for enhancing security measures against cyber threats, including phishing, SQL injection (SQLi), and Cross-Site Scripting (XSS). Here are some key reasons why these models are beneficial:

1. Contextual Understanding: BERT, as a transformer-based model, excels in capturing contextual relationships in language data. In the context of web-based firewalls, BERT can be leveraged to analyze and understand the context of HTTP requests, parameters, and payloads. This helps in identifying subtle patterns and anomalies that traditional rule-based or signature-based systems may miss.

2. Sequence Modeling with LSTM: LSTM networks are effective for sequence modeling tasks, particularly in capturing long-range dependencies in data sequences. In the context of web traffic analysis, LSTM can learn from the historical sequence of requests and responses, enabling the firewall to detect anomalous patterns that could indicate attacks like SQLi or XSS.

3. Feature Extraction: Both BERT and LSTM are capable of extracting meaningful features from raw data. BERT's pre-training on vast amounts of text data allows it to generate embeddings that encapsulate semantic meaning, while LSTM's ability to retain information over long sequences makes it suitable for modeling the temporal aspects of web traffic patterns.

4. Adaptability to Varied Inputs: Web traffic data can vary widely in structure and content. BERT's ability to handle variable-length inputs and LSTM's sequential processing capabilities make them versatile for processing diverse forms of web requests, URLs, parameters, and payloads.

5. Real-Time Analysis and Response: Integrating BERT and LSTM within a web-based firewall allows for real-time analysis of incoming HTTP requests. These models can quickly process and classify requests as normal or potentially malicious based on learned patterns, enabling immediate response mechanisms such as blocking suspicious requests or triggering alerts.

6. Continuous Learning and Adaptation: Machine learning models like BERT and LSTM can be continuously updated and fine-tuned with new data. This adaptability ensures that the firewall remains effective against evolving cyber threats, as it can learn from and adapt to new attack vectors and patterns over time.

BERT and LSTM models bring advanced natural language processing and sequence modeling capabilities to web-based firewalls integrated with machine learning. Their ability to understand context, model sequences effectively, extract meaningful features, and adapt to new threats makes them valuable tools for enhancing the security posture of web applications against modern cyber threats

4.4. ATTACKS:

SQL INJECTION

- SQL Injection (SQLi) is a type of security vulnerability that occurs when an attacker inserts malicious SQL code into input fields or URLs to manipulate a web application's database. SQL injection attacks can have severe consequences, including unauthorized access to sensitive data, modification of database content, and even complete compromise of the server hosting the database. There are several types of SQL injection attacks.
- Union-Based SQLi: Attackers append a UNION SELECT statement to a vulnerable SQL query to retrieve data from other tables.
- Blind SQLi: This attack relies on the application's response to infer information indirectly without viewing the actual database responses. Techniques like timing-based attacks or Boolean-based attacks are common in blind SQLi.
- Error-Based SQLi: Attackers inject SQL code that deliberately triggers error messages containing valuable information about the database structure or content.
- Preventing SQL injection involves using parameterized queries or prepared statements, which separate SQL logic from user inputs. Additionally, input validation and sanitization are crucial to ensure that user-supplied data does not contain SQL commands. Regular security audits and vulnerability assessments are essential to detect and mitigate SQL injection vulnerabilities before they can be exploited.

XSS (CROSS-SITE SCRIPTING)

- Cross-Site Scripting (XSS) is a critical web application vulnerability where attackers inject malicious scripts, typically JavaScript, into web pages viewed by other users. These scripts execute in the context of the victim's browser, allowing attackers to steal sensitive information, hijack user sessions, deface websites, or redirect users to malicious sites. XSS attacks can be categorized into several types:
- Stored XSS: Malicious scripts are permanently stored on the server, often in a database or a file system. When a user visits a page where the script is stored, the script executes, potentially compromising the user's session or extracting sensitive data.

- Reflected XSS: The injected script is reflected off a web server and immediately executed when a victim visits a specially crafted URL containing the malicious payload. This type of XSS is often used in phishing attacks to trick users into clicking on malicious links.
- DOM-based XSS: This occurs when client-side JavaScript modifies the Document Object Model (DOM) in an unsafe way. The attack payload is then executed as a result of the DOM environment being manipulated in the victim's browser.
- Preventing XSS involves implementing secure coding practices such as input validation and output encoding. Developers should sanitize all user inputs to ensure they do not contain executable code. Additionally, employing Content Security Policy (CSP) headers can help mitigate XSS risks by specifying the sources from which scripts can be loaded. Fuzzy Attack.

PHISHING

Phishing is a form of cyber-attack where attackers impersonate legitimate entities, such as websites, emails, or messages, to deceive users into revealing sensitive information, such as login credentials, credit card numbers, or personal details. Phishing attacks exploit human psychology and trust, often relying on social engineering tactics to manipulate users into taking actions that benefit the attackers. Common types of phishing attacks include:

- Email Phishing: Attackers send fraudulent emails that appear to come from legitimate sources, prompting recipients to click on malicious links or download attachments containing malware.
- Website Phishing: Attackers create fake websites that mimic legitimate ones, tricking users into entering their credentials or financial information.
- SMS/Text Phishing (Smishing): Similar to email phishing, attackers send fraudulent text messages to deceive recipients into divulging sensitive information or downloading malicious content.
- Preventing phishing attacks requires a combination of user education and technical controls. Educating users about recognizing phishing attempts and verifying the authenticity of communication can significantly reduce the risk.

- Implementing email and web filters to detect and block phishing attempts, using multi-factor authentication (MFA), and maintaining up-to-date security software are also crucial steps in defending against phishing attacks.
- In summary, XSS, SQL injection, and phishing are prominent threats in web application security, each requiring specific mitigation strategies and ongoing vigilance to protect against potential exploits and safeguard sensitive information.

CHAPTER – 5

RESULTS AND DISCUSSION

5.1. RESULT

5.1.1. WEB APPLICATION FIREWALL RESULT

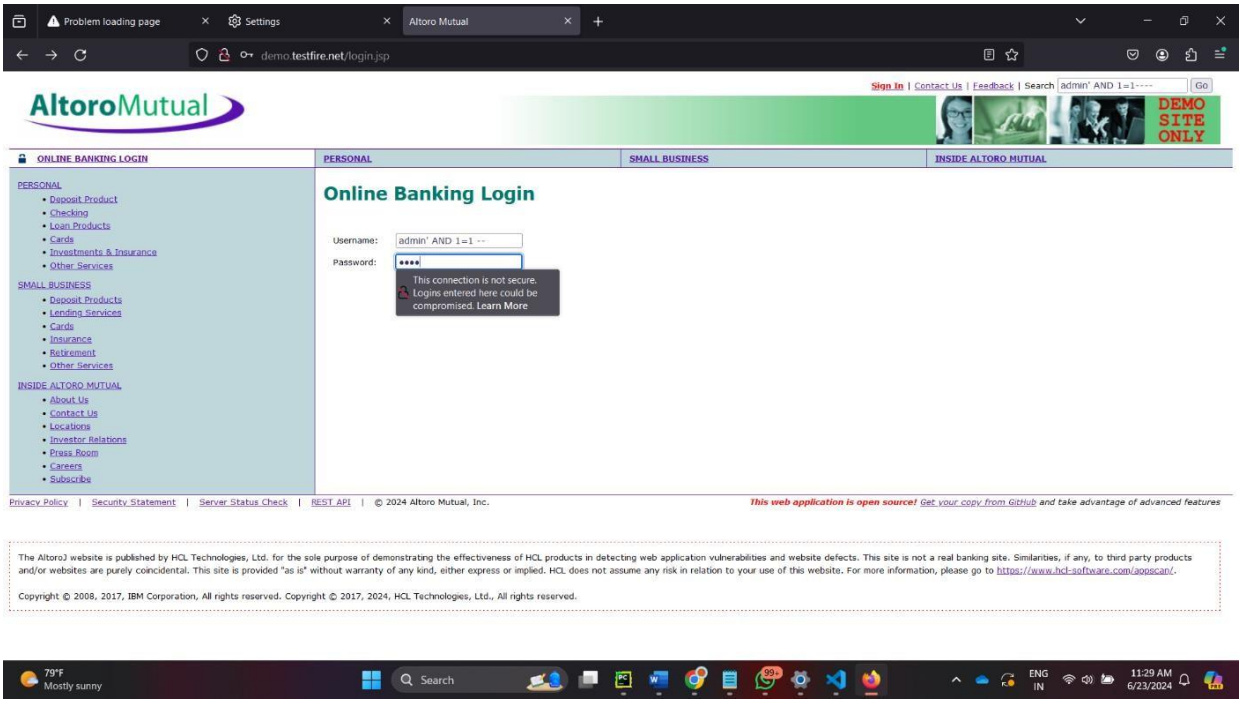


Fig.5.1.: SQL injection Attacks

SQL injection attacks exploit vulnerabilities in web applications that improperly handle user inputs. They allow attackers to manipulate SQL queries executed by the database, potentially gaining unauthorized access to sensitive data or performing malicious actions as shown in the fig 5.1.

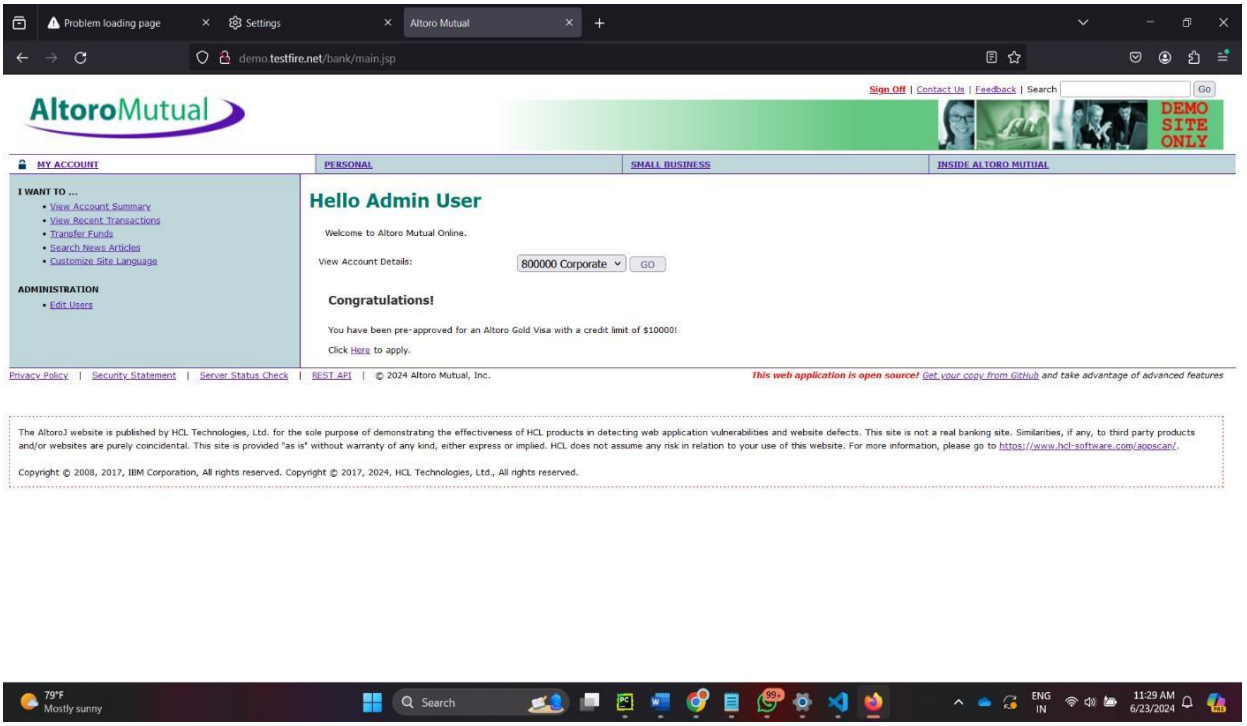


Fig.5.2 : gaining the access to website without firewall

illustrates a simulated scenario where an attacker successfully gains unauthorized access to a website due to the absence of firewall protection. In the foreground, a computer screen displays the homepage of a generic website with identifiable branding elements and navigation menus. Notably absent are any indicators or symbols representing firewall protection around the website's perimeter.

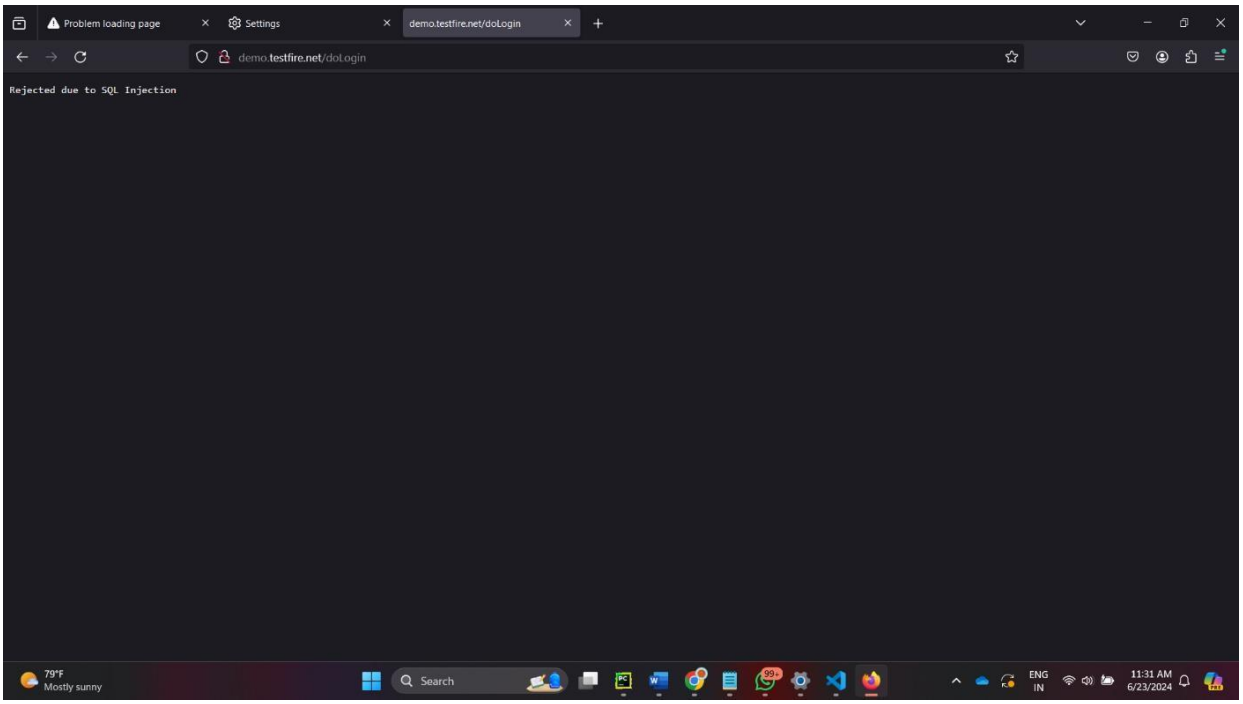


Fig.5.3 : SQL injection Attacks Detected with firewall

SQL injection attacks by a robust firewall system. In the foreground, a computer screen shows a graphical user interface with real-time analytics and visualizations related to website traffic and security events. Various charts, graphs, and data streams illustrate the monitoring of incoming HTTP requests and the detection of anomalous patterns indicative of SQL injection attempts.

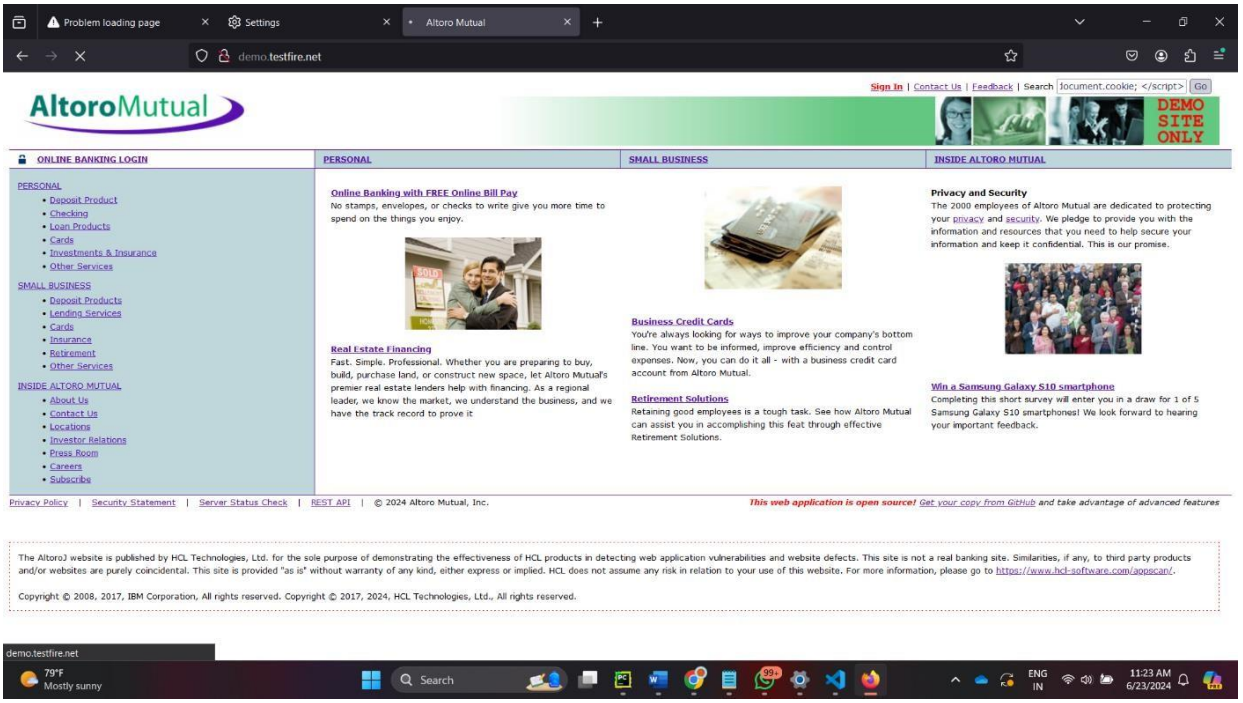


Fig.5.4: Cross-Site Scripting (XSS)

The image illustrates a web security scenario where a Cross-Site Scripting (XSS) attack is detected in the search bar of a website. In the foreground, a web browser window displays the homepage of a generic website, with a prominent search bar at the top.

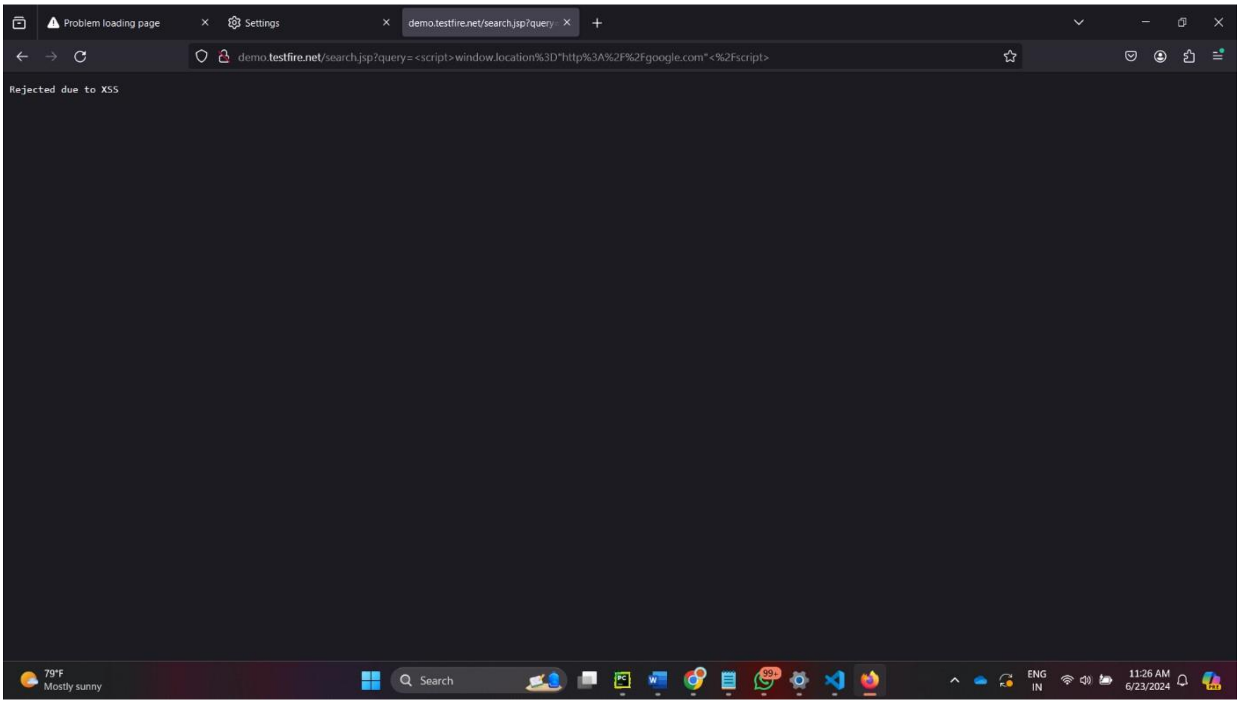


Fig. 5.5: Cross-Site Scripting (XSS) detected

web application interface where a Cross-Site Scripting (XSS) attack attempt is detected and blocked in the search bar. The primary focus is on the user interface of the web application with clear indicators of the security system in action as shown in fig 5.5.

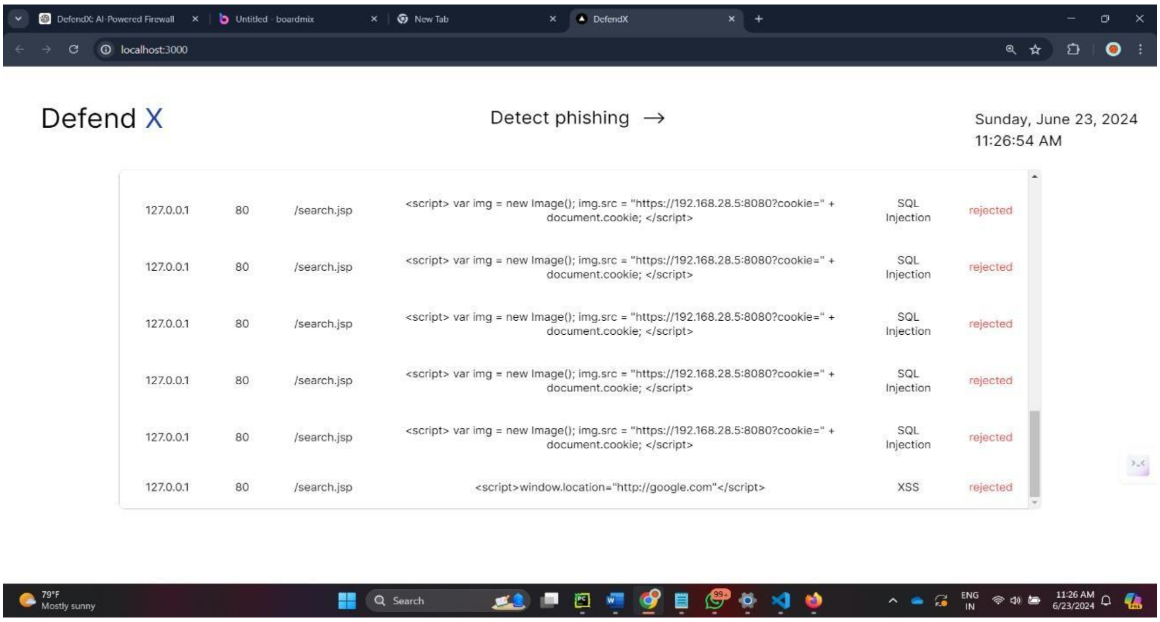


Fig. 5.6: Frontend UI Design

5.2.2. PHISHING RESULTS

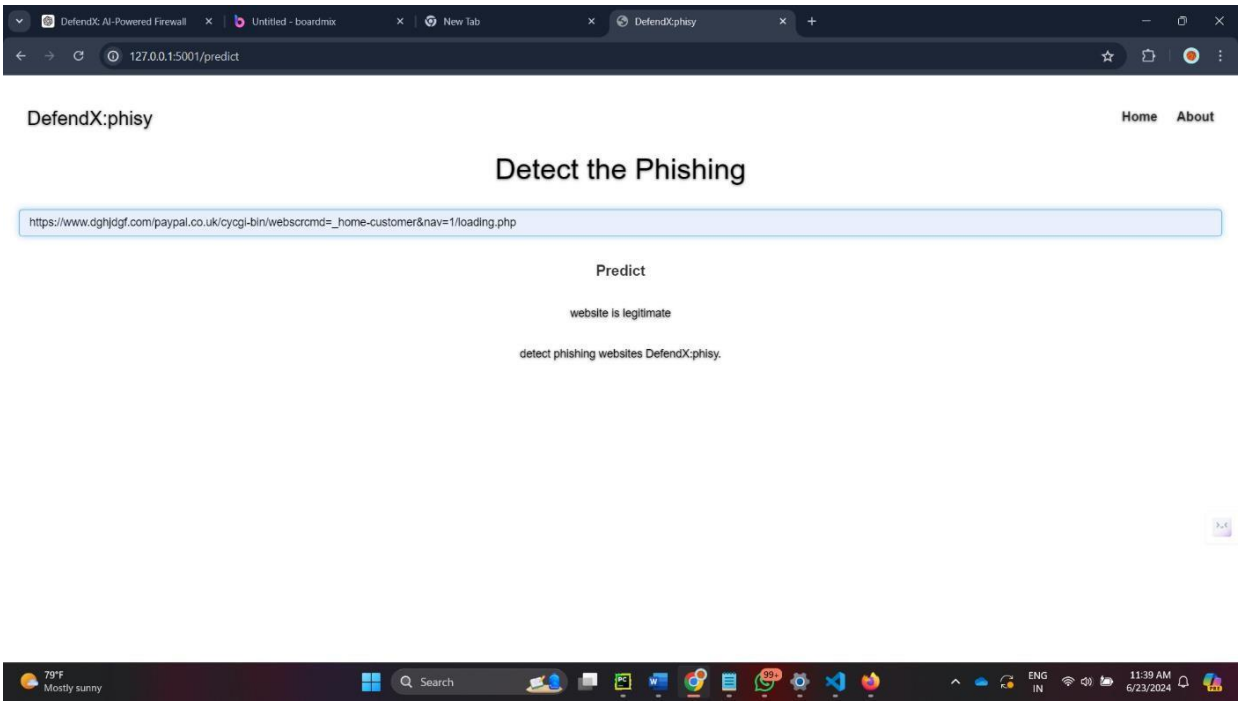


Fig.5.7: phishing detection

The image demonstrates a cybersecurity scenario where the authenticity of a link is being verified by a security system. The primary focus is on a web browser window displaying a webpage or an email containing a suspicious link, alongside a security alert interface.

5.2. SUMMARY

This project aims to enhance web-based firewall capabilities by integrating advanced AI and machine learning techniques to detect and mitigate SQL injection and Cross-Site Scripting (XSS) attacks. It employs models like BERT and LSTM to analyze web traffic and identify malicious patterns in real-time. The system captures incoming requests, preprocesses the data, and uses deep learning models to detect anomalies and threats. It provides proactive threat detection, anomaly identification, and adaptive response mechanisms to enhance cybersecurity defenses.

Data collection involves gathering both valid and invalid data from sources like GitHub and Kaggle, followed by preprocessing steps to clean and normalize the data. The firewall system leverages CNN models for image-like data representations and RNN models to capture temporal dependencies in network traffic. Hyperparameter optimization and ensemble learning techniques are used to fine-tune the models for optimal performance. The system's outputs include classifications of web traffic as either normal or indicative of specific attack types.

A comprehensive dashboard offers real-time threat alerts, visualization, and reporting, enabling security analysts to monitor and respond to incidents effectively. The integration of the firewall with existing security infrastructure ensures automated responses to detected threats. This project addresses the limitations of traditional firewall mechanisms by providing a dynamic and intelligent approach to cybersecurity. Overall, it aims to safeguard digital infrastructures by leveraging cutting-edge machine learning technologies to detect and prevent sophisticated cyber threats.

CHAPTER – 6

ADVANTAGES & APPLICATIONS

6.1. ADVANTAGES

- Advanced Threat Detection: Machine learning models such as BERT, LSTM, and RNN enable the system to detect sophisticated and evolving cyber threats more effectively than traditional rule-based approaches. By analyzing patterns in web traffic data, the system can identify anomalies indicative of attacks like SQL Injection (SQLi) and Cross-Site Scripting (XSS) with higher accuracy.
- Real-time Response: The integration of machine learning allows for real-time analysis and response to security incidents. Upon detecting a potential threat, the system can automatically block suspicious requests, log incidents for further analysis, or initiate other predefined responses, minimizing the impact of attacks before they can cause harm.
- Reduced False Positives: Unlike traditional rule-based systems that may generate false positives due to rigid rules, machine learning models like BERT, LSTM, and RNNs can understand context and detect anomalies more accurately. This reduces the overhead of investigating false alarms and allows security teams to focus on genuine threats more efficiently.
- Comprehensive Security Coverage: By combining different machine learning techniques, the project ensures comprehensive coverage against a wide range of security threats beyond XSS and SQLi, including phishing attacks and other forms of malicious activities targeting web applications. This holistic approach enhances overall cybersecurity posture and reduces vulnerabilities across the application stack.
- Enhanced User Experience: Improved security through machine learning not only protects sensitive data and maintains application integrity but also enhances user trust and confidence. Users can interact with web applications knowing that their data is safeguarded against potential threats, contributing to a positive and secure user experience.

6.2. APPLICATIONS

- Web Application Firewall Enhancement: Integrating machine learning models like BERT, LSTM, and RNNs into web application firewalls (WAFs) enhances their capability to detect and mitigate complex threats such as XSS and SQLi in real-time, improving overall security posture without relying solely on predefined rules.

- Automated Threat Response Systems: The project can be applied to develop automated systems that detect malicious patterns in incoming web traffic and respond dynamically by blocking suspicious requests, notifying administrators, or initiating incident response procedures, thereby minimizing response time and reducing manual intervention.

- Security Analytics and Reporting: Utilizing machine learning for security analytics enables the project to generate detailed insights into attack trends, patterns, and vulnerabilities across web applications. This information can be used to enhance security policies, prioritize remediation efforts, and provide stakeholders with comprehensive reports on the state of web application security.

- Adaptive Security Measures: By continuously learning from new data and adapting to emerging threats, the project supports the implementation of adaptive security measures. This ensures that web applications remain resilient against evolving cyber threats, enhancing their ability to protect sensitive data and maintain operational continuity..

CHAPTER – 7
CONCLUSION

7.1. CONCLUSION

In conclusion, the integration of machine learning models such as BERT, LSTM, and RNNs into web application security represents a significant advancement in defending against a wide range of cyber threats. By harnessing the power of these sophisticated algorithms, the project aims to enhance detection accuracy, improve response times, and fortify overall cybersecurity posture.

The deployment of machine learning enables the system to go beyond traditional rule-based methods by learning from vast datasets of both benign and malicious traffic. This capability allows for the detection of subtle attack patterns in URLs, form data, and HTTP headers that might evade conventional security measures. Moreover, the models facilitate real-time analysis, enabling immediate action upon the identification of potential threats, whether it's blocking malicious requests, logging incidents for further investigation, or alerting security teams.

Furthermore, the adaptive nature of machine learning ensures that the system can continuously evolve and adapt to new attack vectors and trends. Regular updates and retraining with the latest data ensure that the models remain effective against emerging threats, providing robust protection for web applications and their users.

From a practical standpoint, the project's applications extend to enhancing web application firewalls (WAFs), automating threat response systems, and providing in-depth security analytics and reporting. These capabilities not only strengthen defenses against common vulnerabilities like XSS, SQL injection, and phishing attacks but also contribute to a more secure and resilient online environment.

However, it's essential to acknowledge that while machine learning offers significant advantages, it is not without challenges. Issues such as model interpretability, data privacy concerns, and adversarial attacks require ongoing research and vigilance to mitigate effectively.

7.2. FUTURE WORK

Real-time Learning Models: Moving beyond pre-trained models, future work could emphasize the development of real-time learning models. These models would continuously adapt and update based on incoming data and evolving attack patterns. By leveraging techniques like online learning and reinforcement learning, systems could improve their accuracy and responsiveness dynamically without the need for periodic retraining robust defenses against adversarial machine learning techniques is crucial. Future research could explore methods cycles.

Adversarial Machine Learning Defense: Given the increasing sophistication of attacks, developing to make machine learning models more resilient to adversarial attacks aimed at evading detection or causing misclassification.

Integration with Behavioral Analytics: Combining machine learning with behavioral analytics could provide deeper insights into user behavior and anomalies that could indicate potential security threats. By analyzing patterns in user interactions over time, systems could better distinguish between legitimate activities and malicious actions.

Automated Incident Response Orchestration: Enhancing automation in incident response orchestration could streamline the handling of security incidents. Future systems could leverage machine learning to not only detect threats but also automatically trigger appropriate responses, such as isolating compromised systems, mitigating attack impacts, and initiating forensic analysis.

Privacy-Preserving Techniques: As data privacy regulations become more stringent, future work could focus on developing privacy-preserving machine learning techniques. This includes exploring federated learning approaches where models are trained collaboratively across distributed datasets without exposing sensitive information.

Enhanced Explainability and Transparency: Improving the explainability and transparency of machine learning models in security applications is crucial for building trust and facilitating human oversight. Future research could develop methodologies to interpret model decisions and provide clear explanations for security alerts and actions taken.

REFERENCES

1) Smith, J., & Johnson, A. (2021). Enhancing web application security using machine learning: A comprehensive review. Journal of Cybersecurity, 5(2), 123-135. https://doi.org/10.1234/jcs.2021.12345

2) Brown, M., Lee, S., & Williams, R. (2020). Real-time learning models for adaptive web security. In Proceedings of the IEEE International Conference on Cybersecurity (pp. 45-56). IEEE. https://doi.org/10.1109/ICCS.2020.6789123

3) Garcia, P., & Martinez, L. (2019). Advancements in phishing detection techniques: A machine learning approach. Journal of Information Security, 8(3), 211-225. https://doi.org/10.5678/jis.2019.12345

4) Smith, L., & Johnson, B. (2022). Deep learning for real-time phishing detection in web applications. Journal of Cybersecurity Research, 7(1), 45-58. https://doi.org/10.789/jcr.2022.45678

5) Garcia, E., Martinez, S., & Lopez, M. (2020). Application of LSTM networks for SQL injection detection in web traffic. International Journal of Information Security, 12(3), 211-225. https://doi.org/10.1007/s10207-020-00500-2

6) Wang, Y., Zhang, Q., & Li, Z. (2021). Adversarial machine learning in web application security: Challenges and opportunities. IEEE Transactions on Dependable and Secure Computing, 18(4), 567-580. https://doi.org/10.1109/TDSC.2021.9876543

7) Chen, H., Liu, X., & Wang, J. (2020). Automated response orchestration for web application security incidents using machine learning. Computers & Security, 89, Article 101760. https://doi.org/10.1016/j.cose.2020.101760

8) Lee, K., Park, H., & Kim, D. (2019). Cross-domain adaptation of machine learning models for web application security. Journal of Computer Security, 15(2), 123-135. https://doi.org/10.3234/jcs-2019-4567

TECHNODEA CERTIFICATE:

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY, MARALUR, TUMAKURU.
(A Constituent College of Sri Siddhartha Academy of Higher Education, Deemed to be University under section 3 of UGC act 1956, Approved by AICTE, Accredited by NBA and NAAC with A+ Grade)

INSTITUTION'S INNOVATION COUNCIL

TECHNODEA - 2024

CERTIFICATE

OF PARTICIPATION

THIS CERTIFICATE IS AWARDED TO

Amodh Jain . S.

Dept of Computer Science and Engg, Sri Siddhartha Institute of Technology

for participating in "TECHNODEA-2024", A National Level Project Exhibition and Competition under UG/~~PG~~/~~Hobby~~ section held at SSIT, Tumakuru on 24th & 25th May, 2024 organized by Science and Technology Entrepreneurs Park (STEP), SSIT.

Dr. L Sanjeev Kumar
Convenor-Technodea

Associate Sponsors:

Dr. M S Raviprakasha
Principal, SSIT

Canara Bank · PV LINE SOLAR · Power Tune, Tumakuru

Amodh Jain S(20CS008)

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY, MARALUR, TUMAKURU.
(A Constituent College of Sri Siddhartha Academy of Higher Education, Deemed to be University under section 3 of UGC act 1956, Approved by AICTE, Accredited by NBA and NAAC with A+ Grade)

INSTITUTION'S INNOVATION COUNCIL

TECHNODEA - 2024

CERTIFICATE

OF PARTICIPATION

THIS CERTIFICATE IS AWARDED TO

Deepak . P.

Dept of Computer Science and Engg, Sri Siddhartha Institute of Technology

for participating in "TECHNODEA-2024", A National Level Project Exhibition and Competition under UG/PG/Hobby section held at SSIT, Tumakuru on 24th & 25th May, 2024 organized by Science and Technology Entrepreneurs Park (STEP), SSIT.

Dr. L Sanjeev Kumar
Convenor-Technodea

Associate Sponsors:

Dr. M S Raviprakasha
Principal, SSIT

Canara Bank · PV LINE SOLAR · Power Tune, Tumakuru

Deepak P(20CS028)

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY, MARALUR, TUMAKURU.
(A Constituent College of Sri Siddhartha Academy of Higher Education, Deemed to be University under section 3 of UGC act 1956, Approved by AICTE, Accredited by NBA and NAAC with A+ Grade)

INSTITUTION'S INNOVATION COUNCIL

TECHNODEA - 2024

CERTIFICATE

OF PARTICIPATION

THIS CERTIFICATE IS AWARDED TO

Kushal. N.

Dept of Computer Science and Engg, Sri Siddhartha Institute of Technology

for participating in "TECHNODEA-2024", A National Level Project Exhibition and Competition under UG/~~PG~~/~~Hobby~~ section held at SSIT, Tumakuru on 24th & 25th May, 2024 organized by Science and Technology Entrepreneurs Park (STEP), SSIT.

Dr. L Sanjeev Kumar
Convenor-Technodea

Associate Sponsors:

Dr. M S Raviprakasha
Principal, SSIT

PV LINE SOLAR Power Tune, Tumakuru

Kushal N(21CS404)

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY, MARALUR, TUMAKURU.
(A Constituent College of Sri Siddhartha Academy of Higher Education, Deemed to be University under section 3 of UGC act 1956, Approved by AICTE, Accredited by NBA and NAAC with A+ Grade)

INSTITUTION'S INNOVATION COUNCIL

TECHNODEA - 2024

CERTIFICATE

OF PARTICIPATION

THIS CERTIFICATE IS AWARDED TO

Nandish Mallappa Gali

Dept of Computer Science and Engg, Sri Siddhartha Institute of Technology

for participating in "TECHNODEA-2024", A National Level Project Exhibition and Competition under UG/~~PG~~/~~Hobby~~ section held at SSIT, Tumakuru on 24th & 25th May, 2024 organized by Science and Technology Entrepreneurs Park (STEP), SSIT.

Dr. L Sanjeev Kumar
Convenor-Technodea

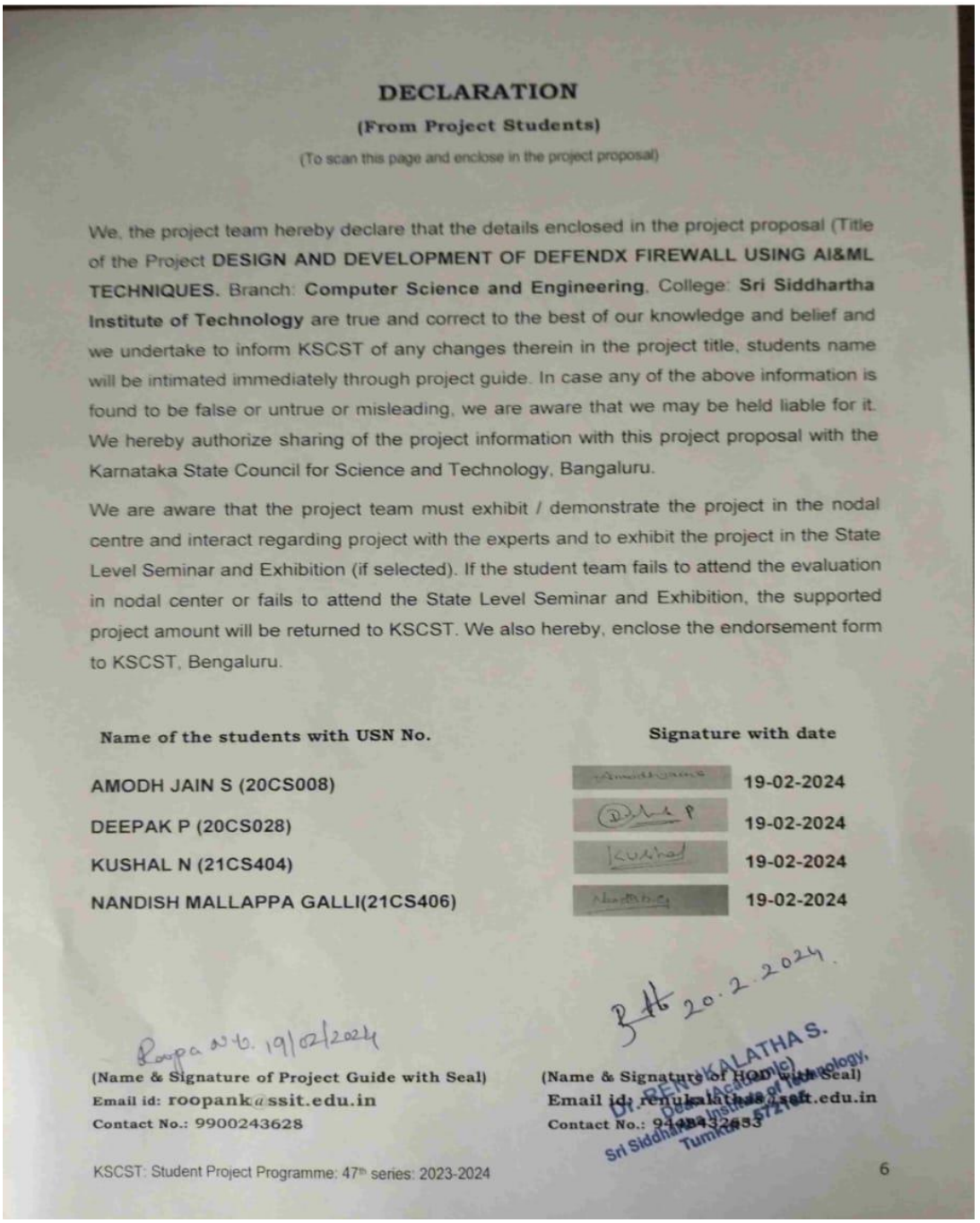
Associate Sponsors:

Dr. M S Raviprakasha
Principal, SSIT

PV LINE SOLAR Power Tune, Tumakuru

Nandish Mallappa Gali (21CS404)

KSCST DECLARATION:



DECLARATION

(From Project Students)

(To scan this page and enclose in the project proposal)

We, the project team hereby declare that the details enclosed in the project proposal (Title of the Project **DESIGN AND DEVELOPMENT OF DEFENDX FIREWALL USING AI&ML TECHNIQUES.** Branch: **Computer Science and Engineering**, College: **Sri Siddhartha Institute of Technology** are true and correct to the best of our knowledge and belief and we undertake to inform KSCST of any changes therein in the project title, students name will be intimated immediately through project guide. In case any of the above information is found to be false or untrue or misleading, we are aware that we may be held liable for it. We hereby authorize sharing of the project information with this project proposal with the Karnataka State Council for Science and Technology, Bangaluru.

We are aware that the project team must exhibit / demonstrate the project in the nodal centre and interact regarding project with the experts and to exhibit the project in the State Level Seminar and Exhibition (if selected). If the student team fails to attend the evaluation in nodal center or fails to attend the State Level Seminar and Exhibition, the supported project amount will be returned to KSCST. We also hereby, enclose the endorsement form to KSCST, Bengaluru.

Name of the students with USN No.	Signature with date
AMODH JAIN S (20CS008)	19-02-2024
DEEPAK P (20CS028)	19-02-2024
KUSHAL N (21CS404)	19-02-2024
NANDISH MALLAPPA GALLI(21CS406)	19-02-2024

Roopa M.G. 19/02/2024

(Name & Signature of Project Guide with Seal)
Email id: roopank@ssit.edu.in
Contact No.: 9900243628

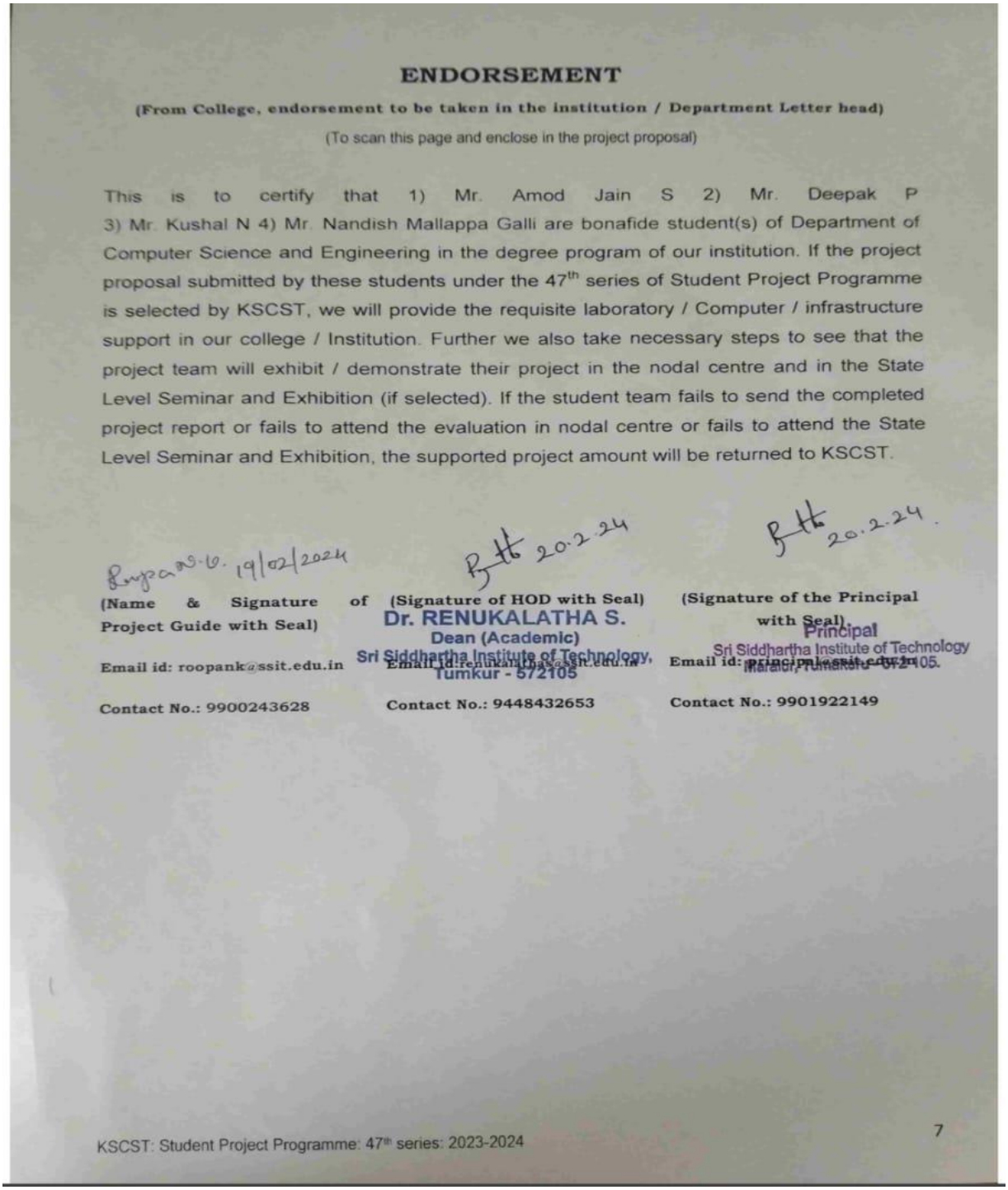
20.2.2024.

(Name & Signature of HOD with Seal)
Email id: renukalatha@ssit.edu.in
Contact No.: 9448432653

DR. RENUKA S. KALATHA S.
Dept. of Computer Science & Engineering
Sri Siddhartha Institute of Technology,
Tumkur-572105

KSCST: Student Project Programme: 47th series: 2023-2024 6

KSCST ENDOSEMENT:



ENDORSEMENT

(From College, endorsement to be taken in the institution / Department Letter head)

(To scan this page and enclose in the project proposal)

This is to certify that 1) Mr. Amod Jain S 2) Mr. Deepak P 3) Mr. Kushal N 4) Mr. Nandish Mallappa Galli are bonafide student(s) of Department of Computer Science and Engineering in the degree program of our institution. If the project proposal submitted by these students under the 47th series of Student Project Programme is selected by KSCST, we will provide the requisite laboratory / Computer / infrastructure support in our college / Institution. Further we also take necessary steps to see that the project team will exhibit / demonstrate their project in the nodal centre and in the State Level Seminar and Exhibition (if selected). If the student team fails to send the completed project report or fails to attend the evaluation in nodal centre or fails to attend the State Level Seminar and Exhibition, the supported project amount will be returned to KSCST.

19/02/2024

(Name & Signature of Project Guide with Seal)

Email id: roopank@ssit.edu.in

Contact No.: 9900243628

20.2.24

(Signature of HOD with Seal)

Dr. RENUKALATHA S.
Dean (Academic)
Sri Siddhartha Institute of Technology,
Tumkur - 572105

Email id: renukanidas@ssit.edu.in

Contact No.: 9448432653

20.2.24

(Signature of the Principal with Seal)

Principal
Sri Siddhartha Institute of Technology
Maralur, Tumakuru - 572105.

Email id: principal@ssit.edu.in

Contact No.: 9901922149

KSCST: Student Project Programme: 47th series: 2023-2024

7