

A
Project Report
on
**PROACTIVE USER FOCUSED ML ARCHITECTURE FOR
CYBERSECURITY MONITORING CENTERS**

Submitted to JNTUK in the partial fulfilment of the requirements for the award of the
Degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING
BY

MATTUPALLI AKHIL KUMAR	21JU1A0508
DUDEKULA ABDUL RAHEEM	21JU1A0501
RUDRAPATI ABHILASH	21JU1A0503
KOTA IRMIYA	21JU1A0553

Under the Esteemed Guidance of

Dr. SK. ALTAF HUSSIAN BASHA M.Tech, Ph.D

Professor



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
KRISHNA CHAITANYA INSTITUTE OF TECHNOLOGY & SCIENCE**

(Approved by AICTE, New Delhi & Affiliated To JNTU, KAKINADA, Accredited by NAAC)

DEVARAJUGATTU, PEDDARAVEEDU MANDAL, PRAKASAM (DIST), A.P.

2021-2025

KRISHNA CHAITANYA INSTITUTE OF TECHNOLOGY & SCIENCES

(Approved by AICTE, New Delhi & Affiliated To JNTUK, KAKINADA & Accredited by NAAC)

DEVARAJUGATTU, PEDDARAVEEDU MANDAL, PRAKASAM (DIST), A.P.

2021-2025



CERTIFICATE

This is to certify that the project-II report entitled “**PROACTIVE USER FOCUSED ML ARCHITECTURE FOR CYBERSECURITY MONITORING CENTERS**”, is the confide work done by MATTUPALLI AKHIL KUMAR (21JU1A0508), DUDEKULA ABDUL RAHEEM (21JU1A0501), RUDRAPATI ABHILASH (21JU1A0503), KOTA IRMIYA (21JU1A0553) under the guidance of **Dr. SK. ALTHAF HUSSIAN BASHA M.Tech,Ph.D, Professor in partial fulfilment of the requirements for the award of BACHELOR OF TECHNOLOGY in the Department of COMPUTER SCIENCE AND ENGINEERING in KRISHNA CHAITANYA INSTITUTE OF TECHNOLOGY AND SCIENCES** affiliated to **JNTU Kakinada** during the academic year 2021- 2025.

Project Guide

Dr. SK. ALTHAF HUSSIAN BASHA M.Tech, Ph.D

Professor

Dept of CSE, KITS

Head of the Department

Dr. J.V. ANIL KUMAR M.Tech, Ph.D

Professor & HOD

Dept of CSE, KITS

External Viva voce conducted on _____

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

The present project work is the several days study of the various aspects of the project development. During this the effort in the present study, we have received a great amount of help from my **Chairman A.V. Rambabu Garu and Secretary & Correspondent Dr. A. Krishna Chaitanya Garu, Krishna Chaitanya Institute Of Technology & Sciences, Devarajugattu**, which we wish to acknowledge and thank from depth of our heart.

We thankful to our Principal **Dr. V. KRISHNA REDDY, Krishna Chaitanya Institute Of Technology & Sciences** for permitting and encouraging us in doing this project.

We deeply intended to **Dr. J.V.ANIL KUMAR M. Tech, Ph.D Professor & Head of the Department**, whose motivation and constant encouragement has led to pursue a project in the field of software development.

We are very much obliged and thankful to our internal guide **Dr. SK. ALTHAF HUSSIAN BASHA M. Tech, Ph.D Professor** for providing this opportunity and constant encouragement given by him during the course. We are grateful to him for his valuable guidance and suggestions during our project work.

We also thank to our parents who supported us for completing this project. Finally, we express our deep sense of gratitude and thanks to the Teaching and Non-Teaching Staff at our college who stood with us during the project and helped us to make it a successful venture.

Project Associates

MATTUPALLI AKHIL KUMAR
DUDEKULA ABDUL RAHEEM
RUDRAPATI ABHILASH
KOTA IRMIYA

21JU1A0508
21JU1A0501
21JU1A0503
21JU1A0553

DECLARATION

We, the students of Krishna Chaitanya Institute Of Technology & Sciences, Devarajugattu, Prakasam(Dist.), hereby declare that this project-II report entitled “**PROACTIVE USER FOCUSED ML ARCHITECTURE FOR CYBERSECURITY MONITORING CENTERS**”, being submitted to the Department of **CSE, KITS COLLEGE** affiliated to JNTU, Kakinada for the award of **Bachelor of Technology in COMPUTER SCIENCE AND ENGINEERING** is a record of confide work done by us and it has not been submitted to any other Institute or University for the award of any other degree or prize.

Project Associates

MATTUPALLI AKHIL KUMAR
DUDEKULA ABDUL RAHEEM
RUDRAPATI ABHILASH
KOTA IRMIYA

21JU1A0508
21JU1A0501
21JU1A0503
21JU1A0553

TABLE OF CONTENTS

S.No	Title	PageNo
1.	ABSTRACT	1
2.	CHAPTEE 1: INTRODUCTION	2-4
	1.1 Introduction	3
	1.2 Objective	3
	1.3 Problem Statement	4
	1.4 Algorithms and Techniques	4
3.	CHAPTER 2: LITERATURE SURVEY	5-7
4.	CHAPTER 3: SYSTEM ANALYSIS	8-10
	3.1 Existing System	9
	3.2 Proposed System	10
5.	CHAPTER 4: SYSTEM REQUIREMENTS	11-12
	4.1 Hardware Requirements	12
	4.2 Software Requirements	12
6.	CHAPTER 5: PROPOSED SOLUTIONS	13-16
	5.1 Overview	14
	5.2 Dataset Description	14
	5.3 Process	16
	5.4 Conclusion	16
7.	CHAPTER 6: SYSTEM DESIGN	17-22
	6.1 System Architecture	18
	6.2 UML Diagrams	19

6.2.1- Use Case Diagram	19
6.2.2 - Class Diagram	20
6.2.3 - Sequence Diagram	21
6.2.4- Data Flow Diagram	22
8. CHAPTER 7: SOFTWARE ENVIRONMENT	23-58
9. CHAPTER 8: IMPLEMENTATION	59-66
8.1 Modules	60
8.1.1 - Cyber Analysis	60
8.1.2 - Dataset Modification	61
8.1.3 - Data Reduction	62
8.1.4 - Risky User Detection	62
8.2 Source Code	63-66
10. CHAPTER 9: RESULTS	67-70
11. CHAPTER 10: SCREENSHOTS	71-80
12. CHAPTER 11: SYSTEMTESTING	81-86
13. CHAPTER 12: CONCLUSION AND FUTURE SCOPE	87-89
14. CHAPTER 13: REFERENCES	90-91

ABSTRACT

In order to ensure a company's Internet security, SIEM (Security Information and Event Management) system is in place to simplify the various preventive technologies and flag alerts for security events. Inspectors (SOC) investigate warnings to determine if this is true or not. However, the number of warnings in general is wrong with the majority and is more than the ability of SCO to handle all awareness. Because of this, malicious possibility. Attacks and compromised hosts may be wrong. Machine learning is a possible approach to improving the wrong positive rate and improving the productivity of SOC analysts. In this article, we create a user-centric engineer learning framework for the Internet Safety Functional Center in the real organizational context. We discuss regular data sources in SOC, their work flow, and how to process this data and create an effective machine learning system. This article is aimed at two groups of readers. The first group is intelligent researchers who have no knowledge of data scientists or computer safety fields but who engineer should develop machine learning systems for machine safety. The second groups of visitors are Internet security practitioners that have deep knowledge and expertise in Cyber Security, but do Machine learning experiences do not exist and I'd like to create one by themselves. At the end of the paper, we use the account as an example to demonstrate full steps from data collection, label creation, feature engineering, machine learning algorithm and sample performance evaluations using the computer built in the SOC production of Seyondike.

INTRODUCTION

CHAPTER – 1

INTRODUCTION

1.1 Introduction

In today's digitally connected world, cybersecurity threats have become increasingly sophisticated, targeting both individuals and organizations. Traditional security systems often lack the ability to adapt dynamically to evolving threats and user behaviors. To address this limitation, this project proposes a **User-Centric Machine Learning Framework** tailored for Cyber Security Operations Centers (CSOCs). This framework is designed to intelligently monitor user activities, detect anomalies in real-time, and generate alert messages, as visualized in the provided interface image. By combining user behavior analytics with machine learning algorithms, the system enhances situational awareness and improves proactive threat mitigation.

1.2 Objective

The main objective of this project is to develop an intelligent, user-aware framework that can:

- Continuously analyze user actions and transactions.
- Detect potential security threats or unusual behavior patterns.
- Generate alert messages based on user-centric risk levels.
- Provide an interactive dashboard for monitoring and responding to alerts in real-time.

1.3 Problem Statement

Existing cybersecurity systems often fail to personalize threat detection based on user context, which can lead to false positives or overlooked threats. Additionally, many current solutions are reactive rather than proactive. There is a pressing need for a system that understands individual user behavior, flags suspicious deviations, and provides targeted alerts. This project aims to fill that gap by implementing a user-centric machine learning solution that operates within a Cyber Security Operations Center (CSOC) environment.

1.4 Algorithms and Techniques

To achieve intelligent threat detection and personalized alerting, the system incorporates several machine learning and analytical techniques, including:

- **Anomaly Detection Algorithms** (e.g., Isolation Forest, One-Class SVM) to identify deviations from normal user behavior.
- **Supervised Learning** methods such as Decision Trees and Random Forest for classification of user transactions.
- **Natural Language Processing (NLP)** to process and generate meaningful alert messages.
- **Clustering Techniques** (e.g., K-Means) to group users with similar activity patterns and detect outliers.

LITERATURE SURVEY

CHAPTER – 2

LITERATURE SURVEY

In recent years, cybersecurity research has increasingly focused on integrating machine learning techniques for threat detection and prevention. A study by Sahu et al. (2019) explored anomaly detection in user behavior through clustering and classification models, which significantly reduced the rate of false positives in security systems. Their work highlighted the importance of behavioral baselines in distinguishing between genuine user actions and malicious activities. This foundational concept supports the proposed system's focus on a user-centric approach for more accurate threat detection.

Another study by Ghosh and Mondal (2020) emphasized the use of supervised learning techniques like Random Forest and Gradient Boosting to classify network intrusions based on labeled datasets. Their framework was able to identify both known and unknown attack patterns effectively. The proposed system builds on this concept by applying supervised learning models to user transaction data, allowing for real-time classification and alert generation for anomalous activities such as unusually high transactions.

A research paper by Alazab et al. (2021) introduced deep learning methods, particularly LSTM networks, to monitor time-series user activity logs for detecting advanced persistent threats (APTs). While deep learning offers greater accuracy, it requires extensive computational resources. Our framework prioritizes efficiency and responsiveness by utilizing lightweight machine learning models suitable for real-time implementation within cybersecurity operation centers, as seen in the system's ability to instantly issue alerts to the user via the interface.

The importance of contextual awareness in cybersecurity systems was highlighted in a study by Tankard (2018), which emphasized that understanding the user's environment and role can help in crafting more relevant and timely security alerts. This aligns with the project's "user-centric" design, where each alert message is tailored based on the user's behavior and risk score. For example, the interface displays a personalized alert such as "Karthick, your transaction money is too high. Be alert," showcasing the system's capability to analyze and notify based on individual context.

Lastly, the integration of machine learning in security operations centers (SOCs) was reviewed by Homoliak et al. (2020), who identified the need for better automation and intelligent filtering of security events. Traditional SOCs are overwhelmed with large volumes of logs and alerts, many of which are false or low-priority. The proposed framework addresses this challenge by intelligently prioritizing alerts and presenting them through a clean, user-specific interface, thereby reducing cognitive load on analysts and improving response times.

SYSTEM ANALYSIS

CHAPTER – 3

SYSTEM ANALYSIS

3.1 Existing System

Most approaches to security in the enterprise have focused on protecting the network infrastructure with no or little attention to end users. As a result, traditional security functions and associated devices, such as firewalls and intrusion detection and prevention devices, deal mainly with network level protection. Although still part of the overall security story, such an approach has limitations in light of the new security challenges described in the previous section.

Data Analysis for Network Cyber-Security focuses on monitoring and analyzing network traffic data, with the intention of preventing, or quickly identifying, malicious activity. Risk values were introduced in an information security management system (ISMS) and quantitative evaluation was conducted for detailed risk assessment. The quantitative evaluation showed that the proposed countermeasures could reduce risk to some extent. Investigation into the cost-effectiveness of the proposed countermeasures is an important future work. It provides users with attack information such as the type of attack, frequency, and target host ID and source host ID. Ten et al. proposed a cyber-security framework of the SCADA system as a critical infrastructure using real-time monitoring, anomaly detection, and impact analysis with an attack tree-based methodology, and mitigation strategies.

3.2 Proposed System

User-centric cyber security helps enterprises reduce the risk associated with fast-evolving end-user realities by reinforcing security closer to end users. User-centric cyber security is not the same as user security. User-centric cyber security is about answering peoples' needs in ways that preserve the integrity of the enterprise network and its assets. User security can almost seem like a matter of protecting the network from the user — securing it against vulnerabilities that user needs introduce. User-centric security has the greater value for enterprises. cyber-security systems are real-time and robust independent systems with high performances requirements. They are used in many application domains, including critical infrastructures, such as the national power grid, transportation, medical, and defense. These applications require the attainment of stability, performance, reliability, efficiency, and robustness, which require tight integration of computing, communication, and control technological systems. Critical infrastructures have always been the target of criminals and are affected by security threats because of their complexity and cyber-security connectivity. These CPSs face security breaches when people, processes, technology, or other components are being attacked or risk management systems are missing, inadequate, or fail in any way. The attackers target confidential data. Main scope of this project in reduce the unwanted data for the dataset.

SYSTEM REQUIREMENTS

CHAPTER – 4

SYSTEM REQUIREMENTS

4.1 Hardware Requirements

- Processor: Intel Core i5/i7 or AMD Ryzen 5/7 (or higher)
- RAM: Minimum 8GB (Recommended: 16GB or higher)
- Storage: Minimum 250GB SSD (Recommended: 512GB SSD or higher)
- GPU (if required for ML models): NVIDIA GTX 1650 or higher
- Network: High-speed internet connection for cloud interactions

4.2 Software Requirements

- Operating System: Windows 10/11, Ubuntu 20.04/22.04, or macOS
- Programming Language: Python 3.8+
- Frameworks & Libraries: TensorFlow, PyTorch, NumPy, Pandas, SciPy
- Virtualization Tools: Docker, Kubernetes, VMware/VirtualBox
- Cloud Platforms: AWS, Google Cloud, Microsoft Azure (any federated cloud setup)
- Database: PostgreSQL, MySQL, or MongoDB
- Web Framework (if using a web interface): Django/Flask
- Other Tools: Git, Jupyter Notebook, VS Code/PyCharm

PROPOSED SOLUTIONS

CHAPTER – 5

PROPOSED SOLUTIONS

5.1 Overview

The proposed solution aims to build a **User-Centric Machine Learning Framework** that enhances the capabilities of a Cyber Security Operations Center (CSOC) by focusing on individual user behavior patterns. Unlike traditional systems that treat all users uniformly, this system tailors its monitoring and alerting based on the user's historical activity and risk profile. The system detects anomalies such as unusually high transaction amounts or suspicious access patterns and generates personalized alerts in real-time. The user interface, as shown in the image, provides a clear visual alert system and ensures users are immediately informed of any potential threats.

5.2 Dataset Description

The system relies on datasets containing historical user activity logs, including:

- **Transaction history:** amount, time, location, frequency.
- **Login behavior:** IP address, device used, login time.
- **Alert records:** previously triggered warning messages and responses.

For training and evaluation, both real and synthetic data were used to simulate various user profiles and behaviors. Anomalies were injected to help the model learn what constitutes unusual behavior for different users. These datasets are labeled for supervised learning models and structured in a format that supports easy feature extraction.

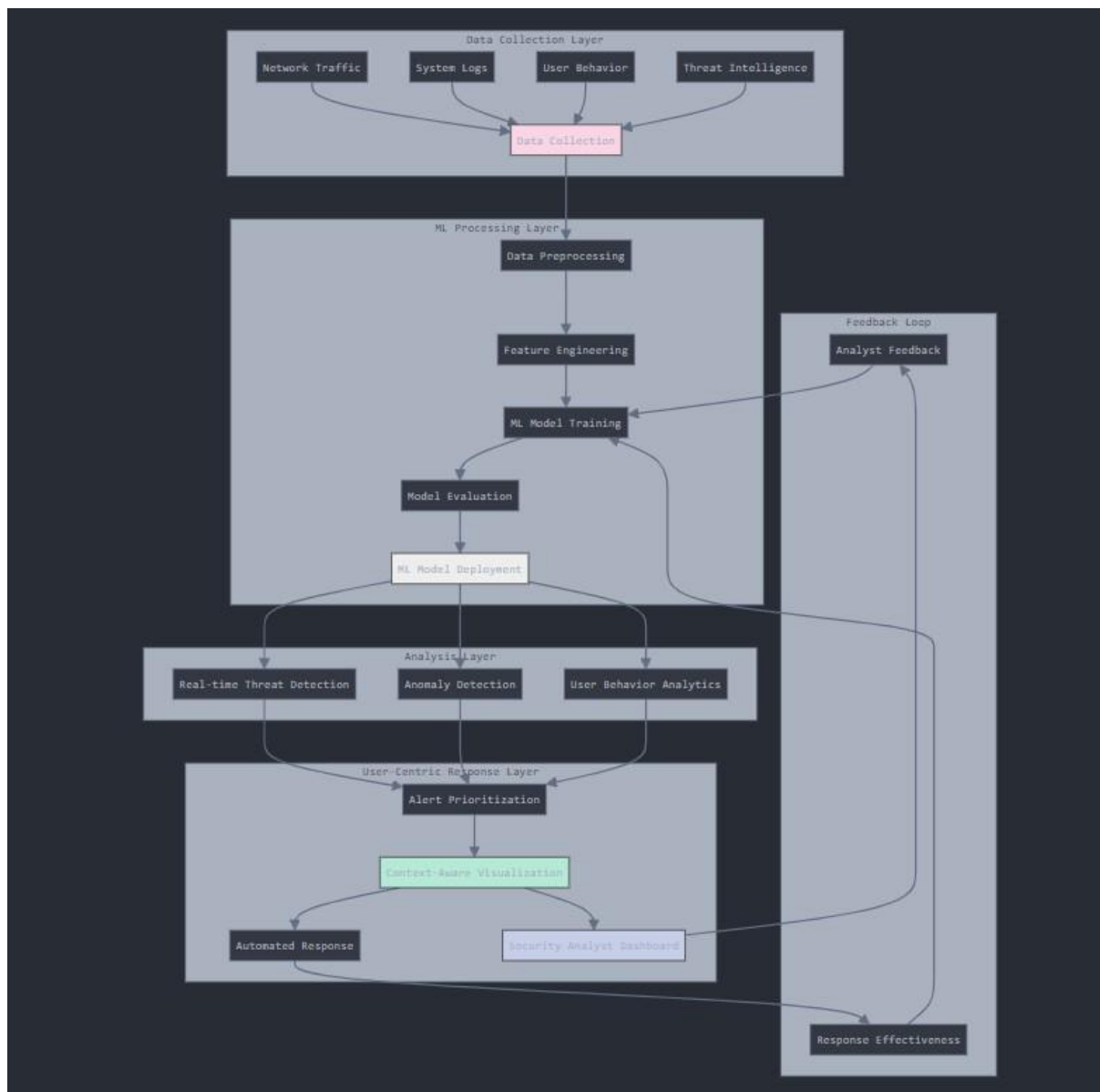


Fig : 5.1 Flowchart of Proposed System

5.3 Process

The proposed system works in the following stages:

1. **Data Collection:** User behavior data is collected from system logs and transaction records.
2. **Preprocessing:** The data is cleaned, normalized, and converted into a structured format suitable for machine learning models.
3. **Feature Extraction:** Key behavioral features (e.g., transaction frequency, login times, amount thresholds) are extracted.
4. **Model Training:** Machine learning models like Isolation Forest, Random Forest, and Decision Trees are trained to detect deviations in behavior.
5. **Anomaly Detection:** Incoming user actions are compared to the user's normal profile. If an anomaly is detected, it triggers an alert.
6. **Alert Generation:** Alerts are displayed on the "Receive Alert" screen (as shown in the image), providing the user with real-time notification of suspicious activity.

5.4 Conclusion

The proposed solution offers a powerful way to enhance cybersecurity operations by putting the user at the center of threat detection. By learning and adapting to each user's unique behavior, the system provides more accurate alerts and reduces false positives. The interface ensures alerts are visually impactful and easy to understand, helping users and security personnel respond quickly. This user-centric approach not only improves threat visibility but also builds a smarter, more responsive CSOC environment.

SYSTEM DESIGN

CHAPTER – 6

SYSTEM DESIGN

6.1 SYSTEM ARCHITECTURE

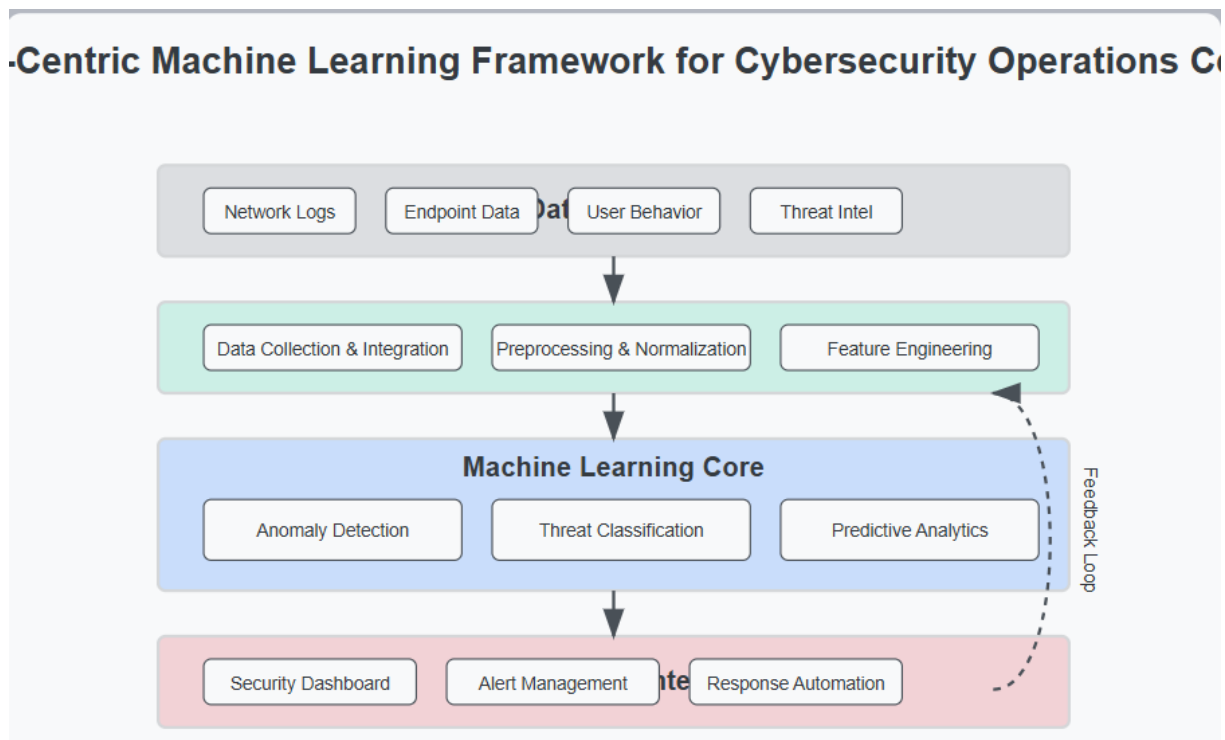
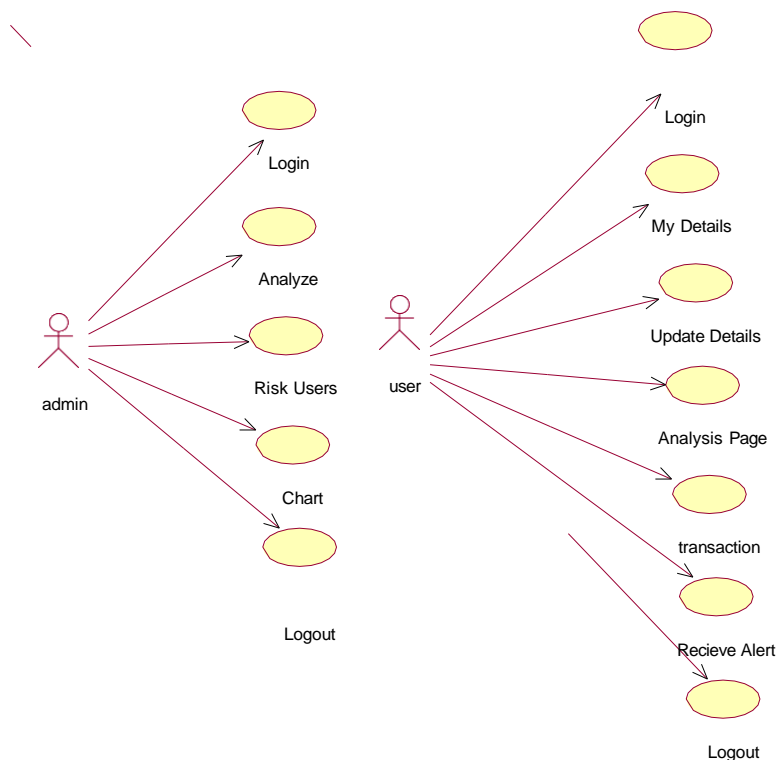


Fig 6.1 : System Architecture

6.2 UML DIAGRAMS

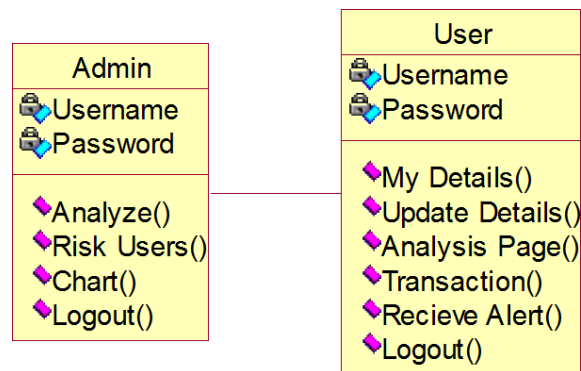
6.2.1 USE CASE DIAGRAM

The Use Case Diagram represents the system's interactions with external users. It identifies the different actors (such as users and the system) and their respective roles, ensuring clarity on how the Whisper model processes speech input and provides transcriptions.



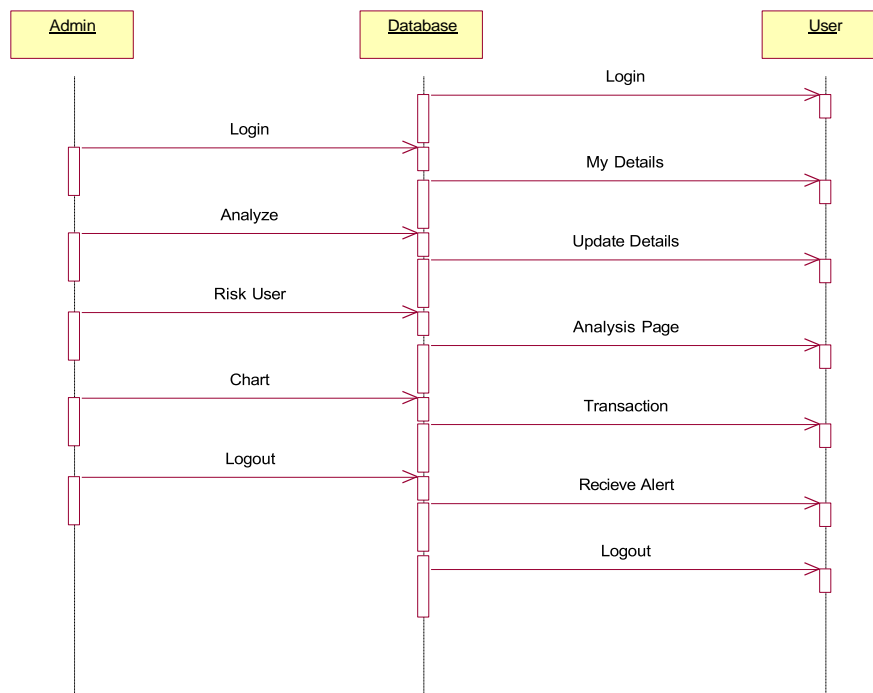
6.2.2 CLASS DIAGRAM

The Class Diagram depicts the structure of the transcription system in terms of its classes, attributes, and relationships. It outlines how different components, such as the audio processor, transcription engine, and text formatter, interact within the system.

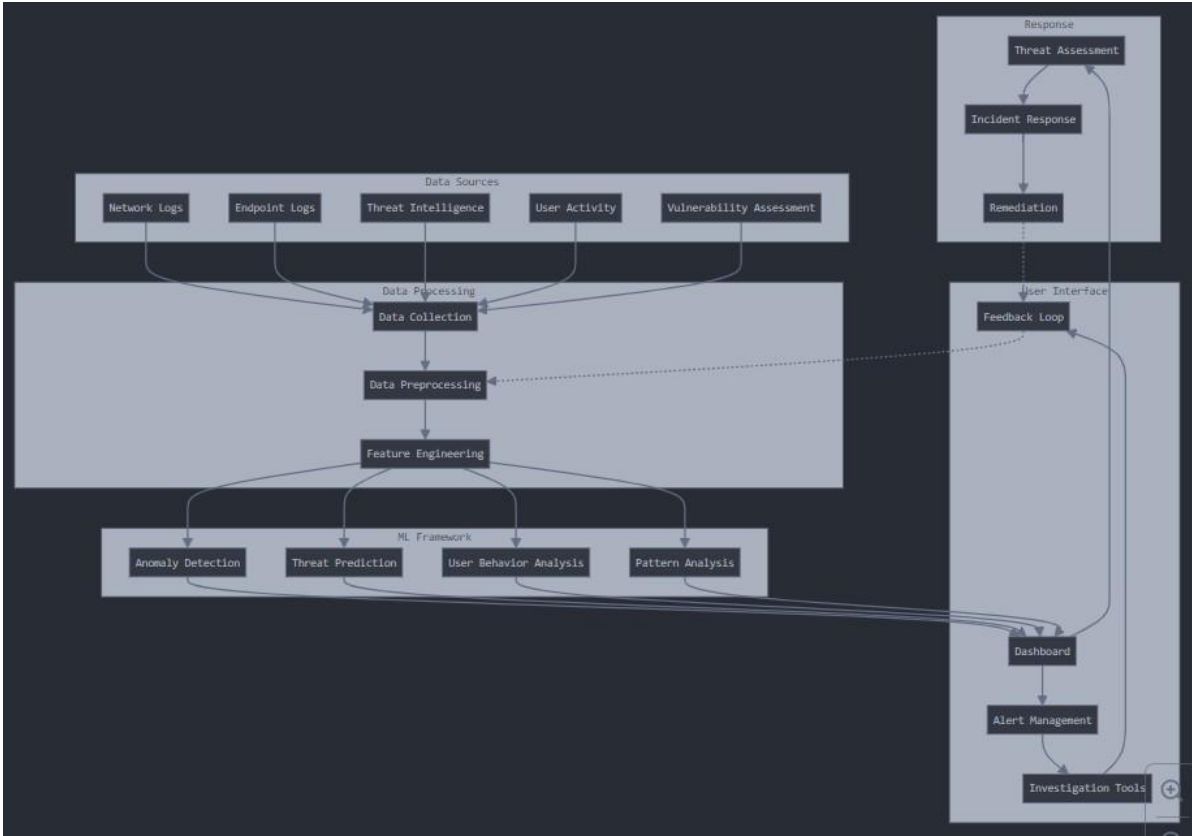


6.2.3 SEQUENCE DIAGRAM

The Sequence Diagram illustrates the step-by-step flow of interactions between system components over time. It demonstrates how an audio file is processed, sent to the Whisper model for transcription, and returned as text output.



6.2.4 DATA FLOW DIAGRAM



SOFTWARE ENVIRONMENT

CHAPTER – 7

SOFTWARE ENVIRONMENT

What is Python :

Below are some facts about Python.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber... etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)

- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

Advantages of Python :-

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it contains code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

When working with Java, you may have to create a class to print 'Hello World'. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

6. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. This further aids the readability of the code.

7. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

8. Free and Open-Source

Like we said earlier, Python is freely available. But not only can you [download Python](#) for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

9. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

10. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

Advantages of Python Over Other Languages :

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third- party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform

data analysis and [machine learning](#), automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since [Python](#) is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle.

The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is dynamically-typed. This means that you don't need to declare the type of variable while writing the code. It uses duck-typing. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java DataBase Connectivity) and ODBC (Open DataBase Connectivity), Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

History of Python : -

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in

the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners¹, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it." Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin- end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

What is Machine Learning : -

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of *building models of data*.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models *tunable parameters* that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with

some broad categorizations of the types of approaches we'll discuss here.

Categories Of Machine Learning :-

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into *classification* tasks and *regression* tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as *clustering* and *dimensionality reduction*. Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't

surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, “to make decisions, based on data, with efficiency and scale”.

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programming logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

Challenges in Machines Learning :-

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are –

Quality of data – Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

Time-Consuming task – Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

Lack of specialist persons – As ML technology is still in its infancy stage, availability of expert resources is a tough job.

No clear objective for formulating business problems – Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

Issue of overfitting & underfitting – If the model is overfitting or underfitting, it cannot be represented well for the problem.

Curse of dimensionality – Another challenge ML model faces is too many features of data points. This can be a real hindrance.

Difficulty in deployment – Complexity of the ML model makes it quite difficult to be deployed in real life.

Applications of Machines Learning :-

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be

solved with traditional approach. Following are some real-world applications of ML –

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

How to Start Learning Machine Learning?

Arthur Samuel coined the term “Machine Learning” in 1959 and defined it as a “Field of study that gives computers the capability to learn without being explicitly programmed”.

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to [Indeed](#), Machine Learning Engineer Is The Best Job of 2019 with a 344% growth and an average base salary of \$146,085 per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let's get started!!!

How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need a Ph.D. degree in these topics to get started but you do need a basic understanding.

(a) Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!!

Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

(c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is [Python](#)! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as [Keras](#), [TensorFlow](#), [Scikit-learn](#), etc.

So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as [Fork Python](#) available Free on GeeksforGeeks.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It's best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- **Model** – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
- **Feature** – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
- **Target (Label)** – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature

section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.

- **Training** – The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- **Prediction** – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

- **Supervised Learning** – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- **Unsupervised Learning** – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
- **Semi-supervised Learning** – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- **Reinforcement Learning** – This involves learning optimal actions through trial and error. So the next action is decided by learning

behaviors that are based on the current state and that will maximize the reward in the future.

Advantages of Machine learning :-

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

2. No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

3. Continuous Improvement

As [ML algorithms](#) gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

Disadvantages of Machine Learning :-

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

Python Development Steps : -

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system. Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it." Some changes in Python 7.3:

- Print is now a function

- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e. int. long is int as well.
- The division of two integers returns a float instead of an integer. "/" can be used to have the "old" behaviour.
- Text Vs. Data Instead Of Unicode Vs. 8-bit

Purpose :-

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Modules Used in Project :-

Tensorflow

TensorFlow is a [free](#) and [open-source software library for dataflow and differentiable programming](#) across a range of tasks. It is a symbolic math library, and is also used for [machine learning](#) applications such as [neural networks](#). It is used for both research and production at [Google](#).

TensorFlow was developed by the [Google Brain](#) team for internal Google use. It was released under the [Apache 2.0 open-source license](#) on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and [IPython](#) shells, the [Jupyter](#) Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the [sample plots](#) and [thumbnail gallery](#).

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Install Python Step-by-Step in Windows and Mac :

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. [Download the Python Cheatsheet here](#). The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web **browser**. OR Click on the following **link:** <https://www.python.org>



Now, check for the latest and the correct version for your operating system.








Step 2: Click on the Download Tab.



Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	 Download	Release Notes
Python 3.6.9	July 2, 2019	 Download	Release Notes
Python 3.7.3	March 25, 2019	 Download	Release Notes
Python 3.4.10	March 18, 2019	 Download	Release Notes
Python 3.5.7	March 18, 2019	 Download	Release Notes
Python 2.7.16	March 4, 2019	 Download	Release Notes
Python 3.7.2	Dec. 24, 2018	 Download	Release Notes

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

Files					
Version	Operating System	Description	MD5 Sum	File Size	GPU
Gzipped source tarball	Source release		68111671e5b2db4ae77b9ab01b7079be	23017663	3xG
XZ compressed source tarball	Source release		d33e4aae6097051c2eca45ee3604803	17131432	3xG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6a28b4fa7583daf1a442cbafce0f8e6	34898416	3xG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773b05e4a936b2a3f	28882845	3xG
Windows help file	Windows		063999573a2c082ac58ade6b47cd2	8131761	3xG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b09c3bfad8e3b9afae3184a40729a2	7504391	3xG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76d8bdc3c3a583e563400	2688398	3xG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c90ff8bd71a8e51a3b4351b4bd2	1362904	3xG
Windows x86 embeddable zip file	Windows		9fab18d18b41379fda941257413bd0	6741628	3xG
Windows x86 executable installer	Windows		33c0c2942a54446ad8d4d147e3b4789	25663848	3xG
Windows x86 web-based installer	Windows		1b670cfafcd117d82c3093ea371d87c	1324608	3xG

- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.

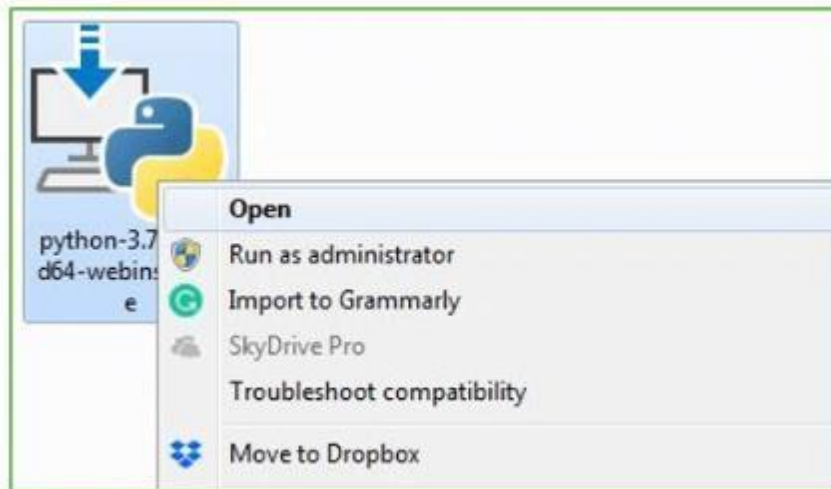
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.



Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



Step 3: Click on Install NOW After the installation is successful. Click on Close.



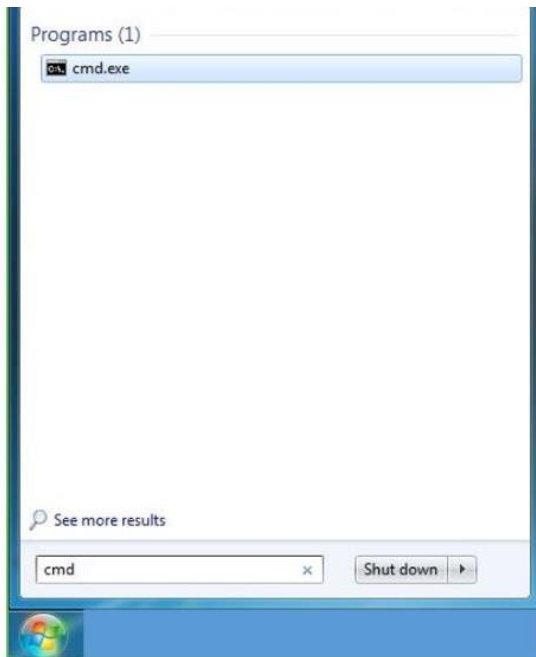
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

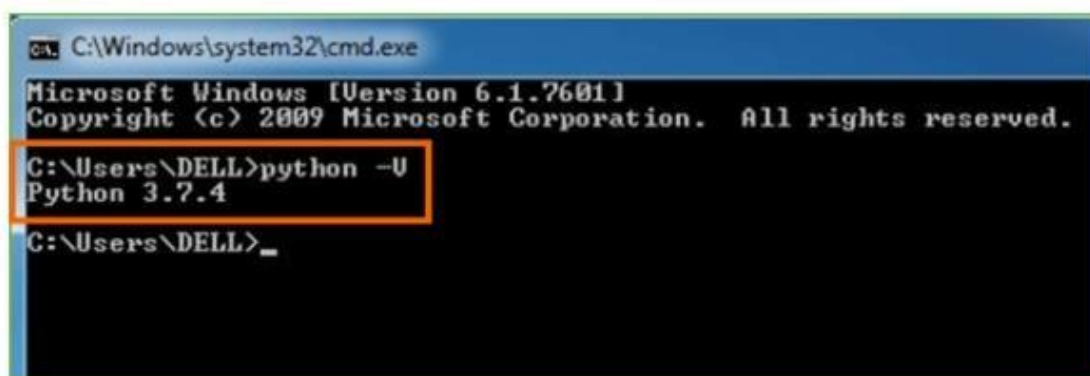
Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed.
Type python -V and press Enter.



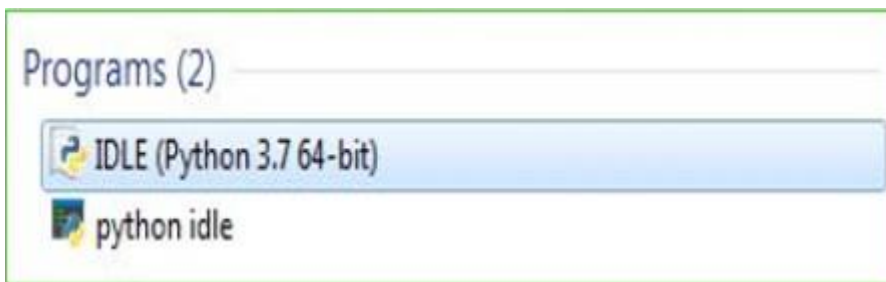
Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

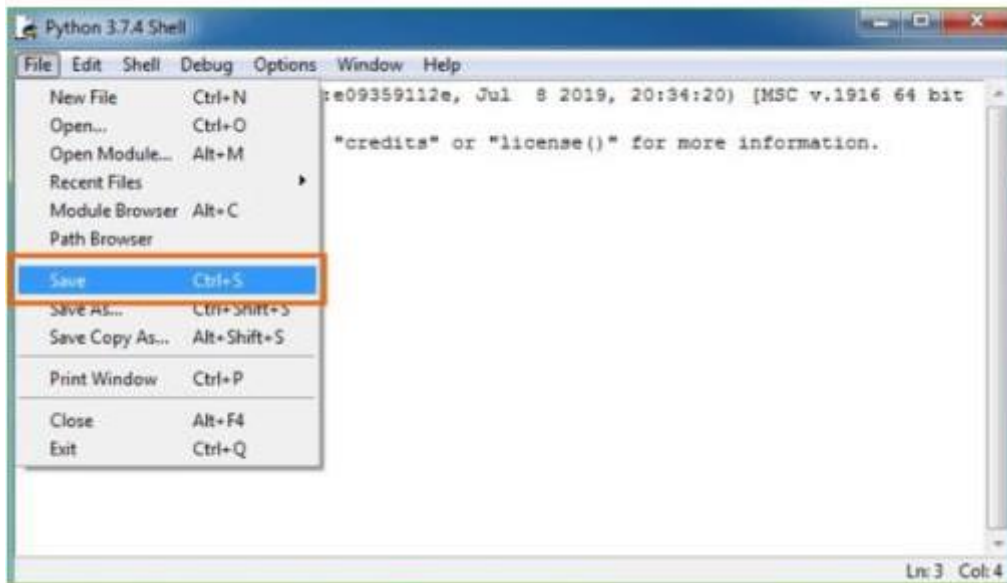
Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.



Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save



Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. enter print

IMPLEMENTATION

CHAPTER – 8

IMPLEMENTATION

The implementation of this user-centric machine learning framework focuses on delivering real-time threat alerts based on dynamic user behavior analysis. The system collects transaction and activity data from users and processes it through various machine learning models trained to detect anomalies, high-risk behaviors, or suspicious patterns. As seen in the interface image, the system immediately alerts the user with contextual messages, such as notifying when a transaction amount exceeds the usual limit. The backend is powered by supervised learning algorithms like Decision Trees and Random Forests, as well as anomaly detection models like Isolation Forests to flag unusual activities. The frontend is built using web technologies to display alerts, manage user profiles, and allow administrators to monitor threat levels across users. This seamless integration between the machine learning engine and user interface ensures both security analysts and end-users are kept informed and protected in real-time.

8.1 MODULES

CYBER ANALYSIS

Cyber threatanalysis is a process in which the knowledge of internal and external information vulnerabilities pertinent to a particular organization is matched against real-world cyber-attacks. With respect to cyber security, this threat-oriented approach to

combating cyber-attacks represents a smooth transition from a state of reactive security to a state of proactive one. Moreover, the desired result of a threat assessment is to give best practices on how to maximize the protective instruments with respect to availability, confidentiality and integrity, without turning back to usability and functionality conditions.

CYPER ANALYSIS. A threat could be anything that leads to interruption, meddling or destruction of any valuable service or item existing in the firm's repertoire. Whether of "human" or "nonhuman" origin, the analysis must scrutinize each element that may bring about conceivable security risk.

DATASET MODIFICATION

If a dataset in your dashboard contains many dataset objects, you can hide specific dataset objects from display in the Datasets panel. For example, if you decide to import a large amount of data from a file, but do not remove every unwanted data column before importing the data into Web, you can hide the unwanted attributes and metrics,

To hide dataset objects in the Datasets panel, To show hidden objects in the Datasets panel, To rename a dataset object, To create a metric based on an attribute, To create an attribute based on a metric, To define the geo role for an attribute, To create an attribute with

additional time information, To replace a dataset object in the dashboard

DATA REDUCTION

Improve storage efficiency through data reduction techniques and capacity optimization using data deduplication, compression, snapshots and thin provisioning. Data reduction via simply deleting unwanted or unneeded data is the most effective way to reduce a storing's data

RISKY USER DETECTION

False alarm immunity to prevent customer embarrassment, High detection rate to protect all kinds of goods from theft, Wide-exit coverage offers greater flexibility for entrance/exit layouts, Wide range of attractive designs complement any store décor, Sophisticated digital controller technology for optimum system performance

8.2 SOURCE CODE

```

from itertools import count

from MySQLdb import Date
from django.contrib import messages
from django.db.models import Count

from django.shortcuts import render, redirect, get_object_or_404

# Create your views here.
from admins.models import Sendquery
from cyber_alert import forms
from cyber_alert.forms import AdminForm, GiverForm
from cyber_alert.models import GiverTransaction, AdminRegister

def admin_login(request):
    if request.method == "POST":
        name = request.POST.get('name')
        password = request.POST.get('password')
        try:
            check = AdminRegister.objects.get(name=name, password=password)
            request.session['name'] = check.id

            return redirect('giver_transaction')
        except:
            pass

    return render(request, "admin_login.html")

def admin_register(request):
    if request.method == "POST":
        forms = AdminForm(request.POST)
        if forms.is_valid():
            forms.save()
            messages.success(request, 'You have been successfully registered')
            return redirect('admin_login')
    else:
        forms = AdminForm()

    return render(request, 'admin_register.html', {'form': forms})

def giver_transaction(request):
    sd= ''
    aas= ''

```

```

sw=''
q=''
name = request.session['name']
obj = AdminRegister.objects.get(id=name)
if request.method == "POST":
    name = request.POST.get('name')
    aadhar = request.POST.get('aadhar')
    address = request.POST.get('address')
    mobile = request.POST.get('mobile')
    bank = request.POST.get('bankname')
    account = request.POST.get('accountno')
    branch = request.POST.get('branchname')
    amount = request.POST.get('amount')
    ifsc = request.POST.get('ifsc')
    micr = request.POST.get('micr')
    date = request.POST.get('date')
    time = request.POST.get('time')
    transaction = request.POST.get('transactionid')
    sd = date.split("-")

    GiverTransaction.objects.create(userid=obj, day=sd[0], month=sd[1], year=sd[2], name=name, aadhar=aadhar, address=address, mobile=mobile, bankname=bank, accountno=account, branchname=branch, amount=amount, ifsc=ifsc, micr=micr, date=date, time=time, transactionid=transaction)

    return render(request, 'giver_transaction.html', {'form':sd, 'we':q})

def analyze_page(request):
    name = request.session['name']
    admin_obj = AdminRegister.objects.get(id=name)
    to_name = admin_obj.name
    obj = GiverTransaction.objects.filter(name=to_name, )

    return render(request, 'analyze_page.html', {'objv': obj})

def viewer(request, chart_type):
    chart =
    GiverTransaction.objects.values('month').annotate(dcount=Count('month'))

    return
    render(request, "viewer.html", {'form':chart, 'chart_type':chart_type})

def update(request):

```

```

name = request.session['name']
obj = AdminRegister.objects.get(id=name)
if request.method == "POST":
    Admin_Id = request.POST.get('adminid', '')
    Name = request.POST.get('name', '')
    Email = request.POST.get('email', '')
    Password = request.POST.get('password', '')
    Phone_Number = request.POST.get('phoneno', '')
    Address = request.POST.get('address', '')

    obj = get_object_or_404(AdminRegister, id=name)
    obj.adminid = Admin_Id
    obj.name = Name
    obj.email = Email
    obj.password = Password
    obj.phoneno = Phone_Number
    obj.address = Address
    obj.save(update_fields=["adminid", "name", "email", "password",
"phoneno", "address" ])
    return redirect('admin_login')
return render(request, 'update.html', {'objc':obj})

def logout_page(request):
    return redirect(admin_login)

def mydetails(request):
    name = request.session["name"]
    obj= AdminRegister.objects.get(id=name)
    if request.method == "POST":
        Admin_Id = request.POST.get('adminid','')
        Name = request.POST.get('name', '')
        Email = request.POST.get('email', '')
        Password = request.POST.get('password', '')
        Phone_Number = request.POST.get('phoneno', '')
        Address = request.POST.get('address', '')

        obj= get_object_or_404(AdminRegister, id=name)
        obj.adminid = Admin_Id
        obj.name = Name
        obj.email = Email
        obj.password = Password
        obj.phoneno = Phone_Number
        obj.address = Address

    return render(request, 'mydetails.html', {'objc': obj})

def show(request):
    return render(request, 'show.html' )

```

```
def receivealert(request):  
    name = request.session['name']  
    admin_obj = AdminRegister.objects.get(id=name)  
    to_name = admin_obj.name  
    obj=Sendquery.objects.filter(name=to_name)  
  
    return render(request, 'receivealert.html',{'de':obj})
```

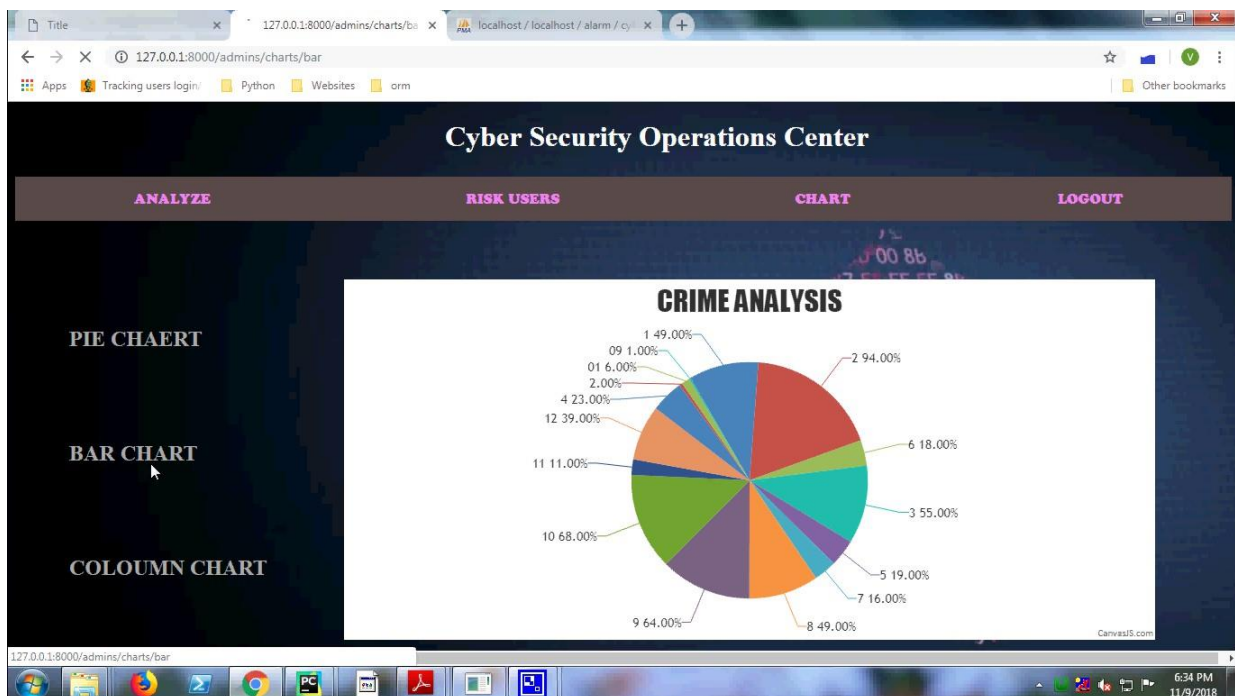

RESULTS

CHAPTER – 9

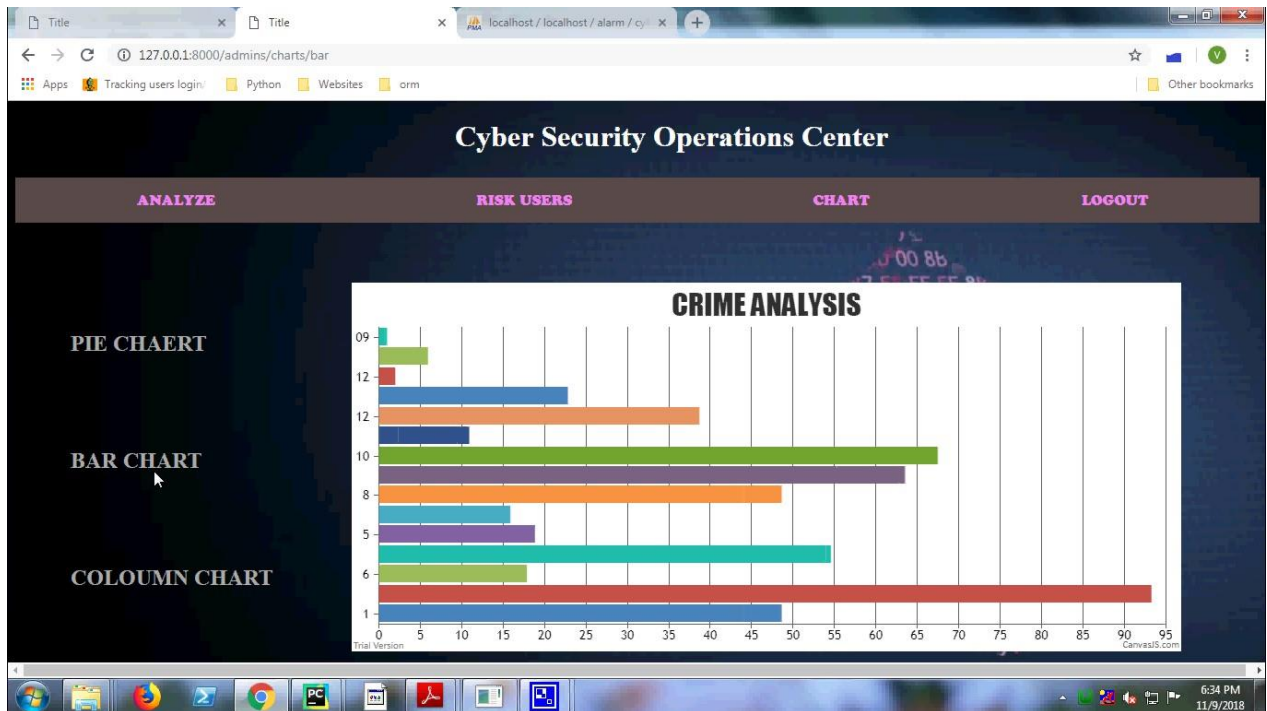
RESULTS



View Graph Analysis



Bar Chart



Coloumn Chart



Risk User Alert



User



SCREENSHORTS

CHAPTER – 10

SCREENSHORTS

Home Page



Registration Page





Login Screen



User Home Page

A User-Centric Machine Learning Framework for Cyber Security Operations Center

MY DETAILS **UPDATE DETAILS** **ANALYZE PAGE** **TRANSACTION** **RECEIVE ALERT** **LOGOUT**

Admin Id: 1002

Name: karthick

Email: karthick@gmail.com

Password: 1234567890

Phone Number: 6387398653

Address: jay nagar , anna salai ch - 2C

UPDATE

Update Details

A User-Centric Machine Learning Framework for Cyber Security Operations Center

MY DETAILS **UPDATE DETAILS** **ANALYZE PAGE** **TRANSACTION** **RECEIVE ALERT** **LOGOUT**

Admin Id: 1002

Name: karthick

Email: karthick@gmail.com

Password: 1234567890

Phone Number: 6387398664

Address: jay nagar , anna salai ch - 2C

UPDATE

Give Transaction Details

A User-Centric Machine Learning Framework for Cyber Security Operations Center

MY DETAILS **UPDATE DETAILS** **ANALYZE PAGE** **TRANSACTION** **RECEIVE ALERT** **LOGOUT**

User Name :

Aadhar Card No :

Address :

Mobile No :

Bank Name :

Account No :

Branch Name :

Amount :

IFSC Code :

MICR Code :

No Cash! Go Digital

A User-Centric Machine Learning Framework for Cyber Security Operations Center

MY DETAILS **UPDATE DETAILS** **ANALYZE PAGE** **TRANSACTION** **RECEIVE ALERT** **LOGOUT**

User Name :

Aadhar Card No :

Address :

Mobile No :

Bank Name :

Account No :

Branch Name :

Amount :

IFSC Code :

MICR Code :

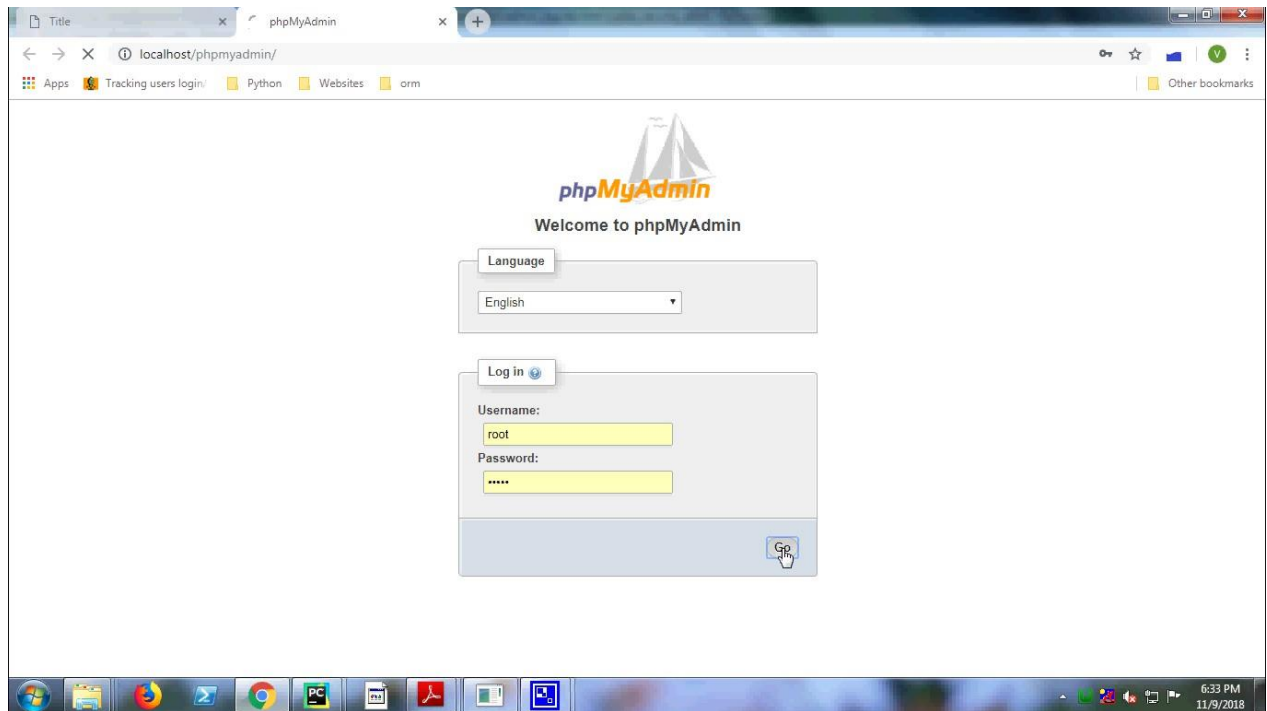
Date :

Time :

Transaction Id :

No Cash! Go Digital

View Details In data Base



Analyze Page



SOC ADMIN Login Page



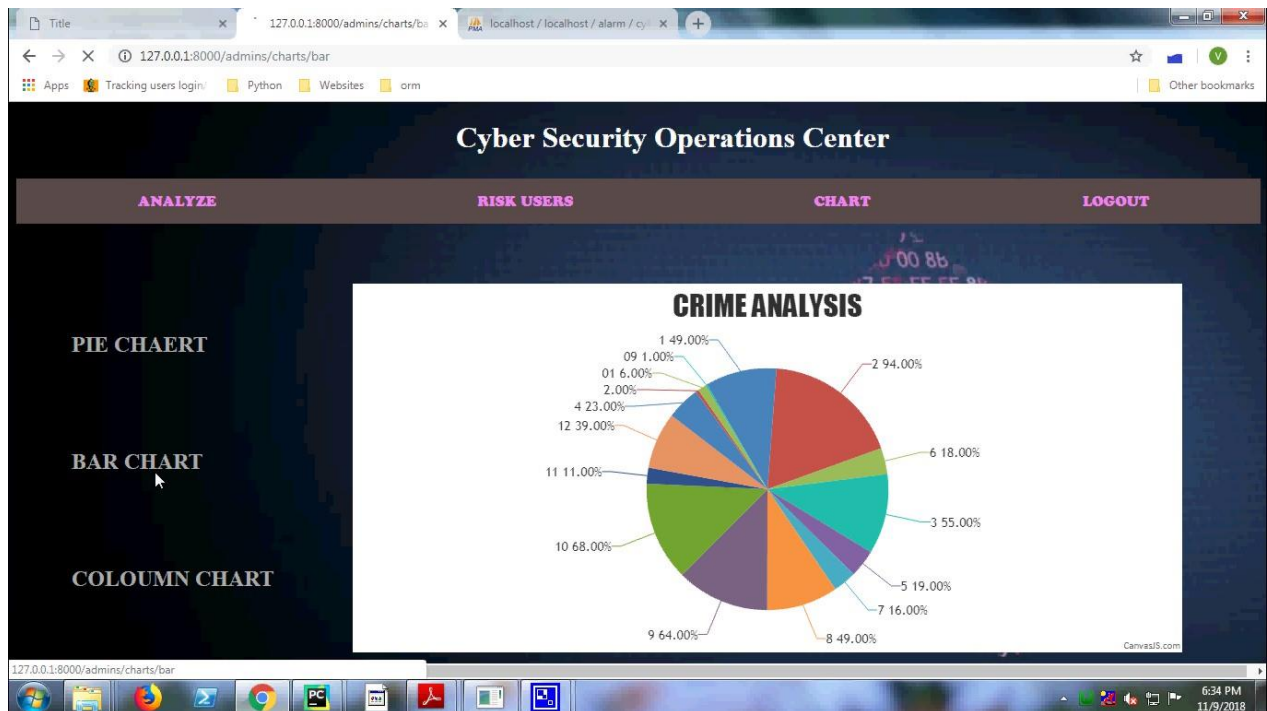
Admin Analyze Page



View Risk Users



View Graph Analysis



Bar Chart



Coloumn Chart



Risk User Alert



User



SYSTEM TESTING

CHAPTER – 11

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before

functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CONCLUSION & FUTURE SCOPE

CHAPTER - 12

CONCLUSION AND FUTURE SCOPE

We provide a user-centered computer learning system that affects large data from various security logs, awareness information, and inspector intelligence. This method provides complete configuration and solution for dangerous user detection for the Enterprise System Operating Center. Select machine learning methods in the SOC product environment, evaluate efficiency, IO, host and users to create user-centric features. . Even with simple mechanical learning algorithms, we prove that the learning system can understand more insights from the rankings with the most unbalanced and limited labels. More than 20% of the neurological model of modeling is 5 times that of the current rule-based system. To improve the detection precision situation, we will examine other learning methods to improve the data acquisition, daily model renewal, real time estimate, fully enhance and organizational risk detection and management. As for future work, let's examine other learning methods to improve detection accuracy

FUTURE SCOPE:

The proposed user-centric machine learning framework holds significant potential for future expansion and real-world deployment within advanced Cyber Security Operations Centers (CSOCs). In the

future, the system can be enhanced with deep learning models for even more accurate anomaly detection and behavior prediction. Integration with real-time threat intelligence feeds and external monitoring tools will allow the framework to respond to global cybersecurity threats proactively. Additionally, incorporating biometric-based user profiling and risk scoring can further personalize the system's alert mechanism. The alert interface, as shown in the image, can be extended to support multi-channel notifications (email, SMS, app notifications) to ensure timely user response. Furthermore, the system can evolve into a fully autonomous cybersecurity assistant that not only detects but also recommends or initiates preventive actions based on learned behaviors and threat patterns.

REFERENCES

CHAPTER – 13

REFERENCES

1. □ Ahmad, I., Basher, M., Iqbal, M. J., & Rahim, A. (2018). *Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection*. IEEE Access, 6, 33789-33795.
2. □ Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). *A deep learning approach to network intrusion detection*. IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1), 41-50.
3. □ Sommer, R., & Paxson, V. (2010). *Outside the closed world: On using machine learning for network intrusion detection*. IEEE Symposium on Security and Privacy, 305–316.
4. □ Chandola, V., Banerjee, A., & Kumar, V. (2009). *Anomaly detection: A survey*. ACM Computing Surveys (CSUR), 41(3), 15.
5. □ Moustafa, N., & Slay, J. (2015). *UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)*. In 2015 Military Communications and Information Systems Conference (MilCIS), 1–6.
6. □ Sarker, I. H., Kayes, A. S. M., Badsha, S., Alqahtani, H., Watters, P., & Ng, A. (2021). *Cybersecurity data science: An overview from machine learning perspective*. Journal of Big Data, 8(1), 1–29.
7. □ Islam, M. R., Tian, H., & Wang, Z. (2019). *User-centered anomaly detection in web applications using behavioral profiling*. Future Generation Computer Systems, 91, 563–573.
8. □ Garfinkel, S., & Rosenblum, D. S. (2020). *Bringing science to cybersecurity*. Communications of the ACM, 63(2), 28–30.
9. □ Brown, G., Cowie, M., Dykstra, J., & Booher, D. (2019). *The role of user behavior analytics in insider threat detection*. In IEEE Security & Privacy, 17(5), 62–70.
10. □ OWASP Foundation. (2023). *Machine Learning Security*. Retrieved from <https://owasp.org>