

# **A Blockchain-Based IoT-Enabled E-Waste Tracking and Tracing System for Smart Cities**

**A PROJECT REPORT**

*Submitted by*

**Hari Prasad R ( 920120104006 )**

**Sivabalan M ( 920120104028 )**

*in partial fulfilment for the award of the degree*

*of*

**BACHELOR OF ENGINEERING**



**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**BHARATH NIKETAN ENGINEERING COLLEGE,**

**AUNDIPATTY**

**ANNA UNIVERSITY : CHENNAI 600 025**

**MAY 2024**

# **ANNA UNIVERSITY : CHENNAI 600 025**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**FEATURE FREE METHOD FOR DETECTING THE PHISHING WEBSITE**” is the bonafide work of “**HARI PRASAD.R (920120104006), SIVABALAN.M (920120104028)**”, who carried out the project work under my supervision.

**SIGNATURE**

**SIGNATURE**

**HEAD OF THE DEPARTMENT**

**SUPERVISOR**

**Mr. S . OYYADEVAN, M.E,(Ph.D).,**

**Mr.S. YOGADINESH, M.E.,**

Department of Computer Science and  
Engineering,

Department of Computer Science and  
Engineering,

Bharath Niketan Engineering College,  
Aundipatty-625 536

Bharath Niketan Engineering College,  
Aundipatty-625 536

Submitted for the Project Phase Viva Voce examination held on .....

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

We thank the most graceful creator of the universe, our almighty **GOD**, who ideally supported us through this project .

At this moment of having successfully completed our project, we wish to convey our sincere thanks to the management and our Chairman **Dr.S .MOHAN** who provide all the facilities to us .

We would like to express our sincere thanks to our Principal **Dr.P.V.ARUL KUMAR, M.E, M.B.A, Ph.D.**, for letting us to do our project and offering adequate duration in completing our project .

we are also grateful to **Mr.S.OYYADEVAN, M.E,(Ph.D).**, our Head of the Department for his constructive suggestions during our project with deep sense of gratitude .

We extend our sincere thanks to our guide to **Mrs.R.MAHALAKSHMI, M.E.**, Assistant Professor, Department of Information Technology, who is our light house in the vast ocean of learning with her inspiring guideless and encouragement to complete the project .

We would like to express our gratitude to all the teaching and non -teaching staff members of Computer Science and Engineering Department and our friends for that kind help extended to us.

## **ABSTRACT**

This study revolves around the E-waste product dataset. With the proliferation of cloud services, there has been a surge in the number of data owners opting to store their encrypted data in the cloud. Concurrently, a significant number of data users engage in data retrieval activities. The approach is founded on blockchain technology. A Hybrid ECC and AES Algorithm are employed for the encryption and decryption of the dataset. Encrypted files are stored on the cloud server, and users conduct keyword-based searches using the algorithm. Users input encrypted queries based on keywords, which are then utilized for searching the encrypted cloud server. Upon locating relevant files based on the query, users can decrypt the files using specific keys, resulting in enhanced performance in terms of recall, ranking precision, and search time.

## **TABLE OF CONTENTS**

<b>CHAPTER NO</b>	<b>TITLE</b>	<b>PAGE</b>
	<b>NOABSTRACT</b>	
<b>1.</b>	<b>INTRODUCTION</b>	<b>2</b>
	1.1 General Introduction	2
	1.2 Project Objectives	5
	1.3 Problem Statement	5
<b>2.</b>	<b>SYSTEMPROPOSAL</b>	<b>6</b>
	2.1 Existing System	6
	2.2 Disadvantages	6
	2.3 Proposed System	7
	2.4 Advantage	7
<b>3.</b>	<b>LITERATURE SURVEY</b>	<b>8</b>
<b>4.</b>	<b>SYSTEM DIAGRAMS</b>	<b>16</b>
	4.1 Architecture Diagram	16
	4.2 Flow Diagram	17
	4.3 Block Diagram	18
	4.4 Flow Chart	19
	4.5 UML Diagrams	20
	4.6 Use Case Diagram	20
	4.7 Sequence Diagram	21
	4.8 ER Diagram	22
	4.9 Class Diagram	23
<b>5.</b>	<b>IMPLEMENTATION</b>	<b>25</b>
	5.1 Modules	25
	5.2 Modules Description	25
	5.3 Cyber Security Dataset	25

	5.4 ECC and AES Algorithm	26
	5.5 Query Search	27
<b>6.</b>	<b>SYSTEM REQUIREMENTS</b>	<b>30</b>
	6.1 Hardware Requirements	30
	6.2 Software Requirement	30
	6.3 Software Description	31
	6.4 Testing of Products	34
<b>7.</b>	<b>SAMPLE CODING</b>	<b>38</b>
<b>8.</b>	<b>SAMPLE SCREENSHOT</b>	<b>61</b>
<b>9.</b>	<b>FUTURE ENHANCEMENT</b>	<b>63</b>
<b>10.</b>	<b>REFERENCES</b>	<b>64</b>
<b>11.</b>	<b>CONCLUSION</b>	<b>66</b>

# CHAPTER 1

## INTRODUCTION

### 1.1 General Introduction

ECC, an alternative technique to RSA, is a powerful cryptography approach. It generates security between key pairs for public key encryption by using the mathematics of elliptic curves.

RSA does something similar with prime numbers instead of elliptic curves, but ECC has gradually been growing in popularity recently due to its smaller key size and ability to maintain security. This trend will probably continue as the demand on devices to remain secure increases due to the size of keys growing, drawing on scarce mobile resources. This is why it is so important to understand elliptic curve cryptography in context.

In contrast to RSA, ECC bases its approach to public key cryptographic systems on how elliptic curves are structured algebraically over finite fields. Therefore, ECC creates keys that are more difficult, mathematically, to crack. For this reason, ECC is considered to be the next generation implementation of public key cryptography and more secure than RSA.

It also makes sense to adopt ECC to maintain high levels of both performance and security. That's because ECC is increasingly in wider use as websites strive for greater online security in customer data and greater mobile optimization, simultaneously. More sites using ECC to secure data means a greater need for this kind of quick guide to elliptic curve cryptography.

An elliptic curve for current ECC purposes is a plane curve over a finite field which is made up of the points satisfying the equation:  $y^2 = x^3 + ax + b$ .

In this elliptic curve cryptography example, any point on the curve can be mirrored over the x-axis and the curve will stay the same. Any non-vertical line will intersect the curve in three places

With the improvement of cloud services, data proprietors are getting motivated in outsourcing their data into the cloud server to achieve better access and storage facility at a low cost. Encrypting the data before outsourcing into the cloud is considered as a general approach for protecting data privacy. Even though encryption protects the data against unauthorized access, but at the same time, it also activates *inconvenience for the authorized users* in accessing the encrypted data at large. As a result, much research is being carried out so as to quickly retrieve the information from the huge pool of data using some **keyword-based search techniques**. It has become a challenge for the researchers to give an efficient multi-keyword search model. Privacy-preserving conjunctive keyword search system over encrypted cloud data cares the update operations dynamically. The index structure is constructed on the basis of **Multi-Attribute Tree (MAT)** and an effective search procedure which is known as search MAT algorithm is introduced. In order to enhance the efficiency of the text searching the index structure based on the **Hierarchical Agglomerative Clustering tree index (HAC-tree)** is proposed. To encrypt the index of HAC tree and query vector, this method uses the secure inner product algorithm. In this, Non-candidate Pruning Depth First Algorithm is used to search the corresponding file in the tree which prunes the sub-tree which does not contain any search result to increase the relevance of the searched keyword to the cloud file, the coordinate matching along with inner product similarity is introduced. Reverse data structure to permit users to accomplish dynamic operations on document collection is proposed, which perform either inserting or deleting. The sparse matrix is used to encrypt the index matrix and query vector to enhance efficiency to provide privacy for both cloud service providers



as well as data users, the new **Oblivious Multiple Keyword Search (OMKS)** is proposed. The proposed protocol support multiple keyword searches such as conjunctive keyword search and disjunctive keyword search. In the disjunctive keyword search, it apprehends in a simple way that it sends the values of the keyword to the server in the query. In the conjunctive keyword search, the addition of all keywords values is used as the fresh keywords values involved in the calculation. By using these two searches this method achieves efficient search and matched cipher text the multi-keyword tree-based search scheme is proposed to provide security to the sensitive information of the data owners.

The document collection in the cloud environment is achieved through the hierarchical clustering method. To generate an encrypted index as well as query vectors, the vector space model is used and to achieve efficient search, DFS algorithm is used. The secure proposed algorithm is used to encrypt the query vectors. The clustering of documents is performed using bisecting k-means clustering. Context-aware search is introduced to make semantic search smart. The proposed method first introduces the **Semantic Compound Keyword Search (SCKS)** as a knowledge representation tool. Two schemes are proposed based on CG. This method converts original CG into their corresponding linear form with few modifications and it matches them to numerical vectors. Ranked multi keyword search over encrypted data in the cloud is introduced on the basis of two threat models. To resolve the problem in the privacy-preserving smart semantic search based on CGs, the proposed scheme uses PRSCG and PRSCG-TF schemes. The compound concept semantic similarity evaluation method is projected to quantify the similarity between the compound concepts. This method integrates both secure K nearest neighbour scheme and CCSS with Locality Sensitive Hashing Function, thus proposing the **Semantic Compound Keyword Search (SCKS)**. The goal of secure this scheme is to steadily recognize the K-Nearest points in the encrypted databank to a provided encrypted query. This proposed method not only achieves semantic-based

search but at the same time also performs a multi-keyword search and ranks the searched result

## **1.2 PROJECT OBJECTIVE:**

The main objective is,

To perform the, Encryption and Decryption with less data loss.

To implement the AES learning algorithm.

To enhance the performance analysis.

## **1.3 PROBLEM STATEMENT:**

To perform the, Encryption and Decryption with less data loss.

Although this method is straightforward and user, it has some severe limitations.

Time taken to done the Encryption and decryption is very low, when compared with the other techniques.

## **CHAPTER 2**

### **SYSTEM PROPOSAL**

#### **2.1 EXISTING SYSTEM**

- Employing a proxy server within a Cloud Service Provider (CSP) enhances search efficiency and reduces search time by employing a Boolean search mechanism in the proxy server.
- The primary server facilitates simultaneous access for multiple users through a Deep Learning-based Neural Network, ensuring precise results.
- A Trusted Authority is tasked with ensuring secure document retrieval for authorized users. The TA oversees dual security measures encompassing key management and the issuance of security devices.
- Ensuring secure top k ranking is attained through Euclidean distance calculations, thereby enhancing the accuracy of document retrieval.

#### **2.2 DISADVANTAGE**

- Encryption and decryption file on time high
- Along with that, data loss is more when compared with the other conventional methods.

#### **2.3 PROPOSED SYSTEM**

- The adoption of cloud services has led to a surge in data owners storing their encrypted data on cloud platforms, paralleled by an equivalent or greater number of data users engaged in retrieval activities.
- This approach is underpinned by blockchain technology.
- A Hybrid ECC and AES Algorithm are utilized for the encryption

and decryption of the dataset.

- Encrypted files are securely stored on cloud servers, and users conduct keyword-based searches using an algorithm.
- Users input encrypted queries based on keywords, which are then utilized for searching the encrypted cloud server.
- Upon locating relevant files based on the query, the retrieval process is initiated to access the encrypted file related to the query data.
- Users employ a specific key to decrypt files, leading to improved performance in terms of recall, ranking privacy, precision, and search time.

## **2.4 ADVANTAGE:**

- Time taken to done the Encryption and decryption is very low,when compared with the other techniques.
- Easy to retrieve the data from the cloud.
- Data loss is low,in the receiver side during the decryption process.

## CHAPTER 3

### LITERATURE SURVEY

- **Title :** Proficient Dual Secure Multi Keyword Search by Top k- Ranking based on Synonym Index and DNN in Untrusted Cloud
- **Year :** 2018
- **Author :** Aditi Gudadhe, Akanksha Parbat, Bhavana Wankhede, Brinda Darjee, Dr. Leena H.Patil
- **Methodology:**

A secure ranking based multi keyword search using semantic index is being developed. Initially, owner builds an index file by semantic representation of keywords using Term Frequency/Inverse Document Frequency (TF/IDF). Security key is provided by Trusted Authority (TA) for decrypting the obtained results at the user side. TA manages dual security processes such as managing secret key and issuing security device to the data users. User query reaches proxy server, and it checks whether any frequent keyword matches with given query by Boolean Search. If not, query enters into the main server who stores all document and index files to obtain relevant result using Deep Learning Neural Network. In deep learning neural network, the query is processed with vector space model in order to retrieve the relevant documents. Finally, user decrypts the relevant results obtained from deep neural network. The experimental result shows that our proposed model provides better performance in terms of recall, ranking privacy, precision, searching time.

#### **Advantage:**

- encryption time and accuracy

**Disadvantage:**

- It requires basic information about keyboard shortcuts used or where the keys are located.

**LITERATURE SURVEY**

**Title:** Secure Ranked Multi-Keyword Search Based on Modified Blowfish algorithm and AVL Tree in Untrusted Cloud Environment

- **Year :** 2014
- **Author :** G.Shoba, G.Anusha, V.Jeevitha, R.Shanmathi

This MB algorithm provides robustness against any intruding whereas the conventional blowfish algorithm insecure for many applications. To achieve a proficient search, every data owner's index based on AVL tree is encrypted by way of additive order and the privacy-preserving family is formed. The cloud server is then permitted to combine these indexes effectually without knowing the index content. An Iterative Deepening Depth First Search (IDDFS) procedure is used to discover the matching file for the data user request

**Advantage:**

- User query reaches proxy server, and it checks whether any frequent keyword matches with given query by Boolean Search.

**Disadvantage:**

- User query reaches proxy server, and it checks whether any frequent keyword matches with given query by Boolean Search is not matching on retrieval file.

## **LITERATURE SURVEY**

### **Title: Cyber Security Threats Detection in Internet of Things Using DeepLearning Approach**

- **Year :** 2014
- **Author :**FarhanUllah

This MB algorithm provides robustness against any intruding whereas the conventional blowfish algorithm insecure for many applications. To achieve a proficient search, every data owner's index based on AVL tree is encrypted by way of additive order and the privacy-preserving family is formed. The cloud server is then permitted to combine these indexes effectually without knowing the index content. An Iterative Deepening Depth First Search (IDDFS) procedure is used to discover the matching file for the data user request

#### **Advantage:**

- More taken time for Encryption

#### **Disadvantage:**

- User Not matched Retrieval

## **LITERATURE SURVEY**

### **Title: Encrypted multi-keyword ranked search supporting gram based search technique**

- **Year :** 2016
- **Author :** Suresh M

Scheme that not only enable document keyword search but also supports linear, gram based and semantic searches. We construct a special tree-based index structure and propose a fuzzy Search Server that creates wild card

based fuzzy keyword Set which overcome KGA (Keyword Guessing Attack) and provide efficient multi-keyword ranked search. KNN algorithm is used to encrypt the index and query. It also uses the relevance score calculation for ranking the documents.

**Advantage:**

- More taken time for Encryption

**Disadvantage:**

- It requires basic information about keyboard shortcuts used or where the keys are located.
  - Relevance score calculation for ranking the documents
- Ranking Very low

**LITERATURE SURVEY**

**Title: Smart cloud search services: verifiable keyword-based semantic search over encrypted cloud data**

- **Year :** 2016
- **Author :** Linga

**Methodology :**

For protecting data privacy, sensitive data are always encrypted before being outsourced. Although the existing searchable encryption schemes enable users to search over encrypted data, these schemes support only exact keyword search, which greatly affects data usability. Moreover, these schemes do not support verifiability of search result. In order to save computation cost or download bandwidth, cloud server only conducts a fraction of search operation or return a part of result, which is viewed as selfish and semi-honest-but-curious. So, how to enhance flexibility of encrypted cloud data while supporting verifiability of search result is a big challenge. To tackle the challenge, a smart semantic search scheme is



proposed in this paper, which returns not only the result of keyword-based exact match, but also the result of keyword-based semantic match. At the same time, the proposed scheme supports the verifiability of search result..

**Advantage:**

- Less taken time for Encryption

**Disadvantage:**

- It requires basic information about keyword-based semantic match not perfect matching.
- Relevance score calculation for ranking the documents Ranking Very low

**LITERATURE SURVEY**

**Title: Privacy preserving synonym based fuzzy multi-keyword ranked search over encrypted cloud data**

- **Year :** 2014
- **Author :** Linga
- **Methodology:**

Privacy Preserving Synonym Based Fuzzy Multi-Keyword Ranked Search over Encrypted Cloud Data, a scheme which enhances user search experience to a paramount by providing both fuzzy and synonym based multi-keyword ranked search, thereby taking encrypted search experience closer to free text search engines. The scheme additionally improves upon index generation time and search time in comparison to existing schemes by utilizing a binary tree based dynamic index. Experimental results portray the effectiveness of this proposed scheme as it reduces the search time

**Advantage:**

- As it reduces the search time.

**Disadvantage:**

- It is Convert File the data huge time taken For binary tree based dynamic index
- More taken time for Semantic Searching

**LITERATURE SURVEY**

**Title:**Anomaly detection for electricity consumption in cloud computing: framework, methods, applications, and challenges

**Year:** 2020

**Author:**Rohit Patil<sup>1</sup>, Muzamil Kacchi<sup>2</sup>, Pranali Gavali<sup>3</sup>, Komal Pimparia

**Methodology:**

To provide better electricity service for the customers and minimize the losses for the providers, a leap in the power grid is occurring, which is referred to as the smart grid. The smart grid is envisioned to increase the detection accuracy to an acceptable level by utilizing modern technologies, such as cloud computing. With the aim of obtaining achievements of anomaly detection for electricity consumption with cloud computing, we firstly introduce the basic definition of anomaly detection for electricity consumption. Next, we conduct the surveys on the proposed framework of anomaly detection for electricity consumption and propose a new framework with cloud computing.

**Advantage**

- Machine learning is fast and accurate.

**Disadvantage**

- ECC cannot handle large data for prediction.

**LITERATURE SURVEY**

**Title:**Experimental and analysis on household electronic power consumption

**Year :** 2020

**Author :** Jing

**Methodology:**

Household power consumption helps the power supply department understand the power consumption of residents and whether there will be some abnormal power consumption phenomena. Taking the individual household electric power consumption dataset as an example, this paper establishes an extensible experimental analysis framework and analyzes the data in a visual way

**Advantage**

- Machine learning is fast and accurate.

**Disadvantage**

- AES cannot handle large data for prediction.

**LITERATURE SURVEY**

**Title:** Individual Household Electric Power Consumption Forecasting using Machine Learning Algorithms

**Year:** 2020

**Author:** H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, and S. Mishra.

## **Methodology:**

Electric energy consumption is the actual energy demand made on existing electricity supply. However, the mismanagement of its utilisation can lead to a fall in the supply of electricity. It is therefore imperative that everybody should be concerned about the efficient use of energy in order to reduce consumption [1]. The purposes of this research are to find a model to forecast the electricity consumption in a household and to find the most suitable forecasting period whether it should be in daily, weekly, monthly, or quarterly. The time series data in our study is the individual household electric power consumption

## **Advantage**

- Machine learning is fast and accurate.

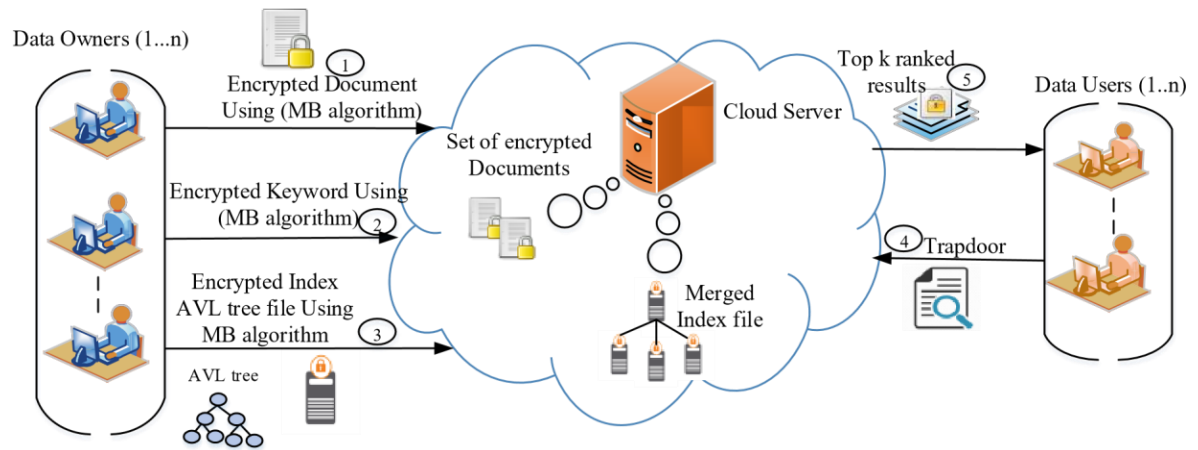
## **Disadvantage**

- ECC cannot handle large data for prediction.

# CHAPTER 4

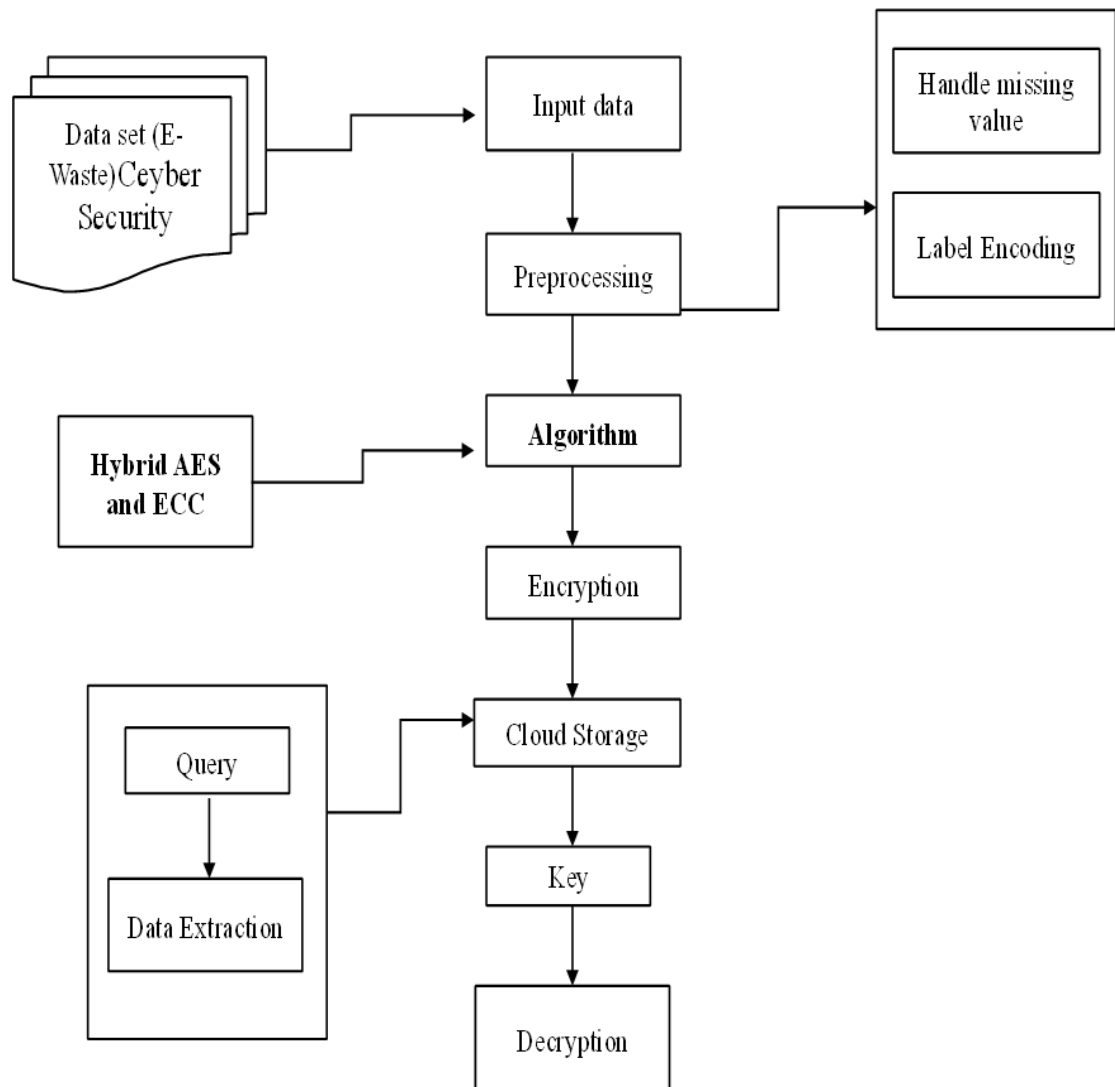
## SYSTEM DIAGRAMS

### 4.1 ARCHITECTURE DIAGRAM

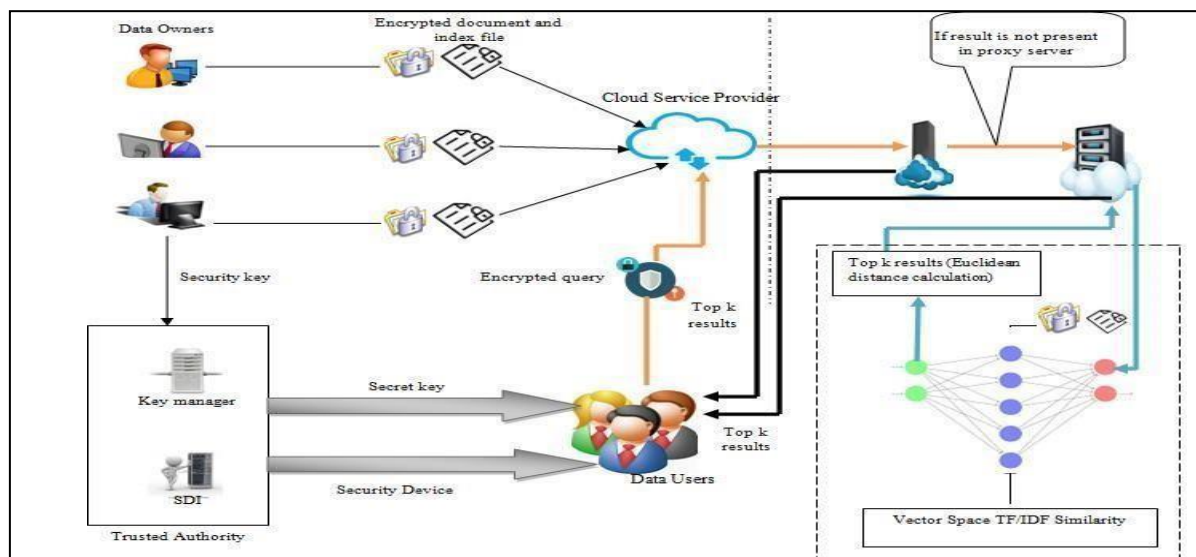
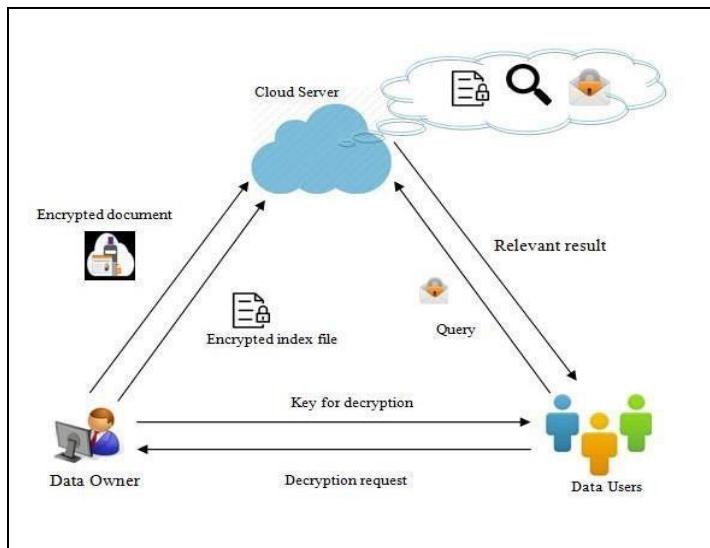


**Figure 1**

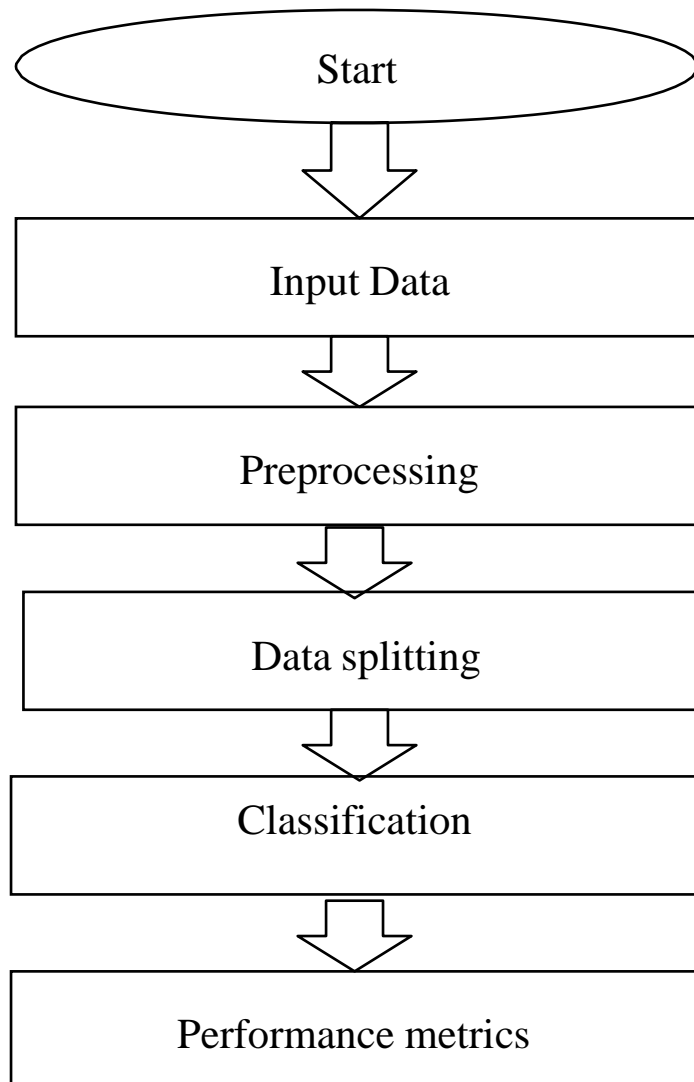
**4.2FLOW DIAGRAM:**



### 4.3 BLOCK DIAGRAM



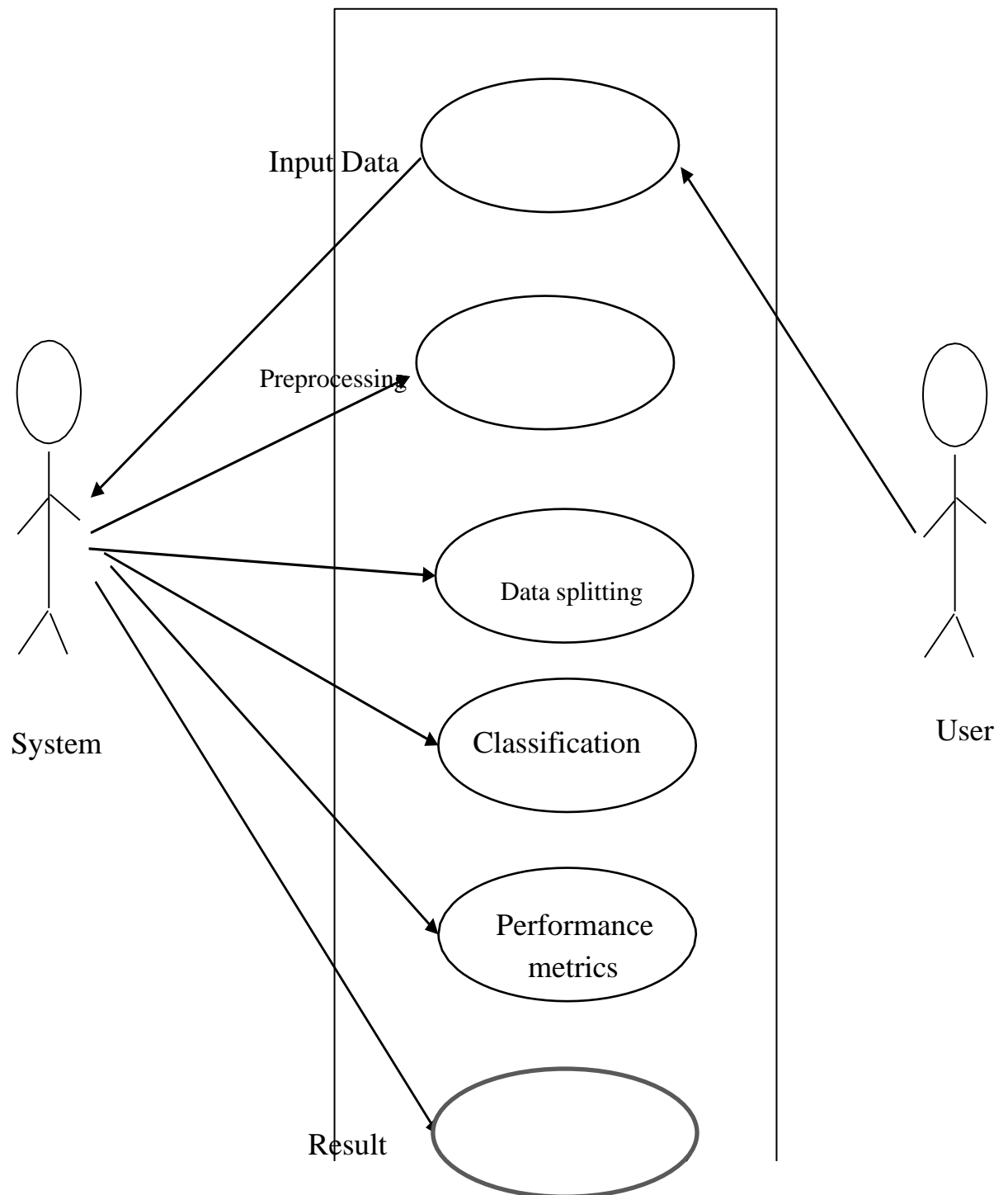
#### 4.4 FLOW DIAGRAM



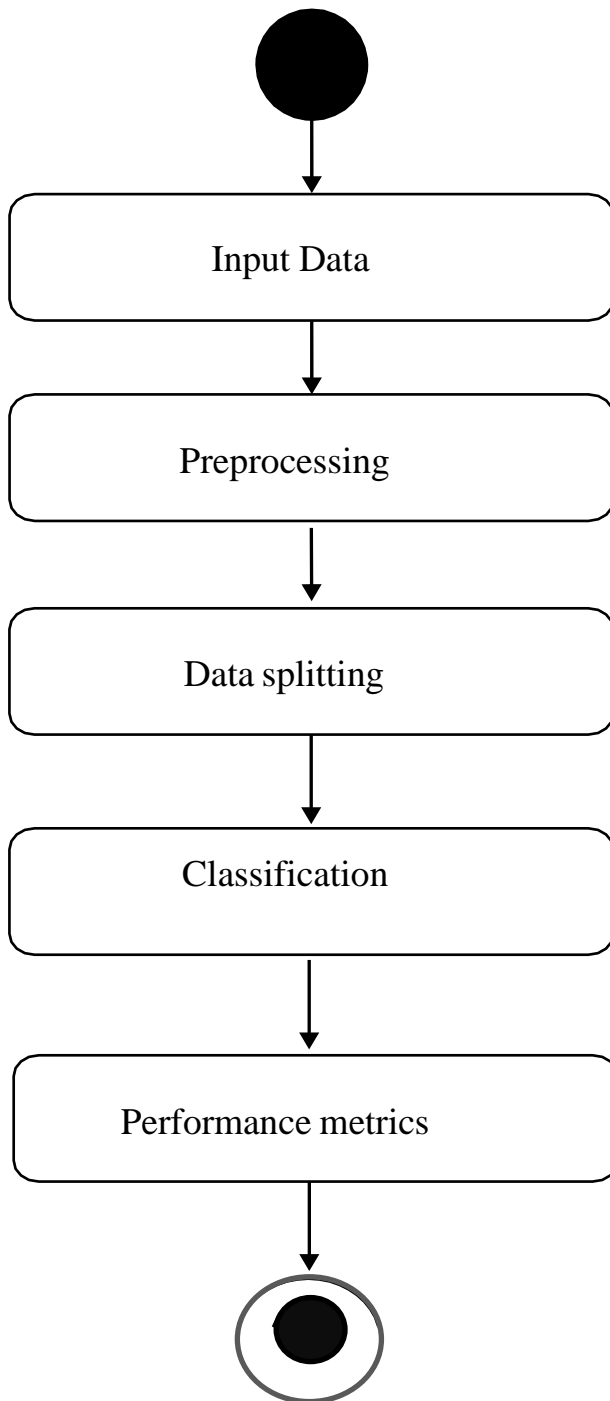


## 4.5 UML DIAGRAMS:

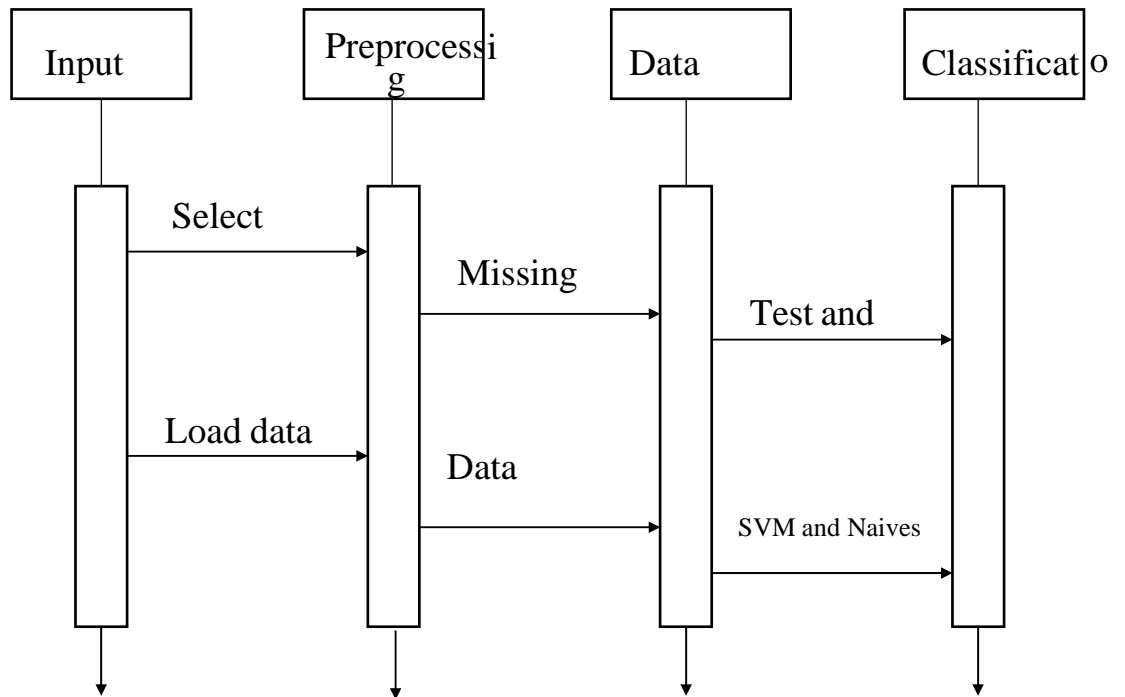
### USE CASE DIAGRAM:



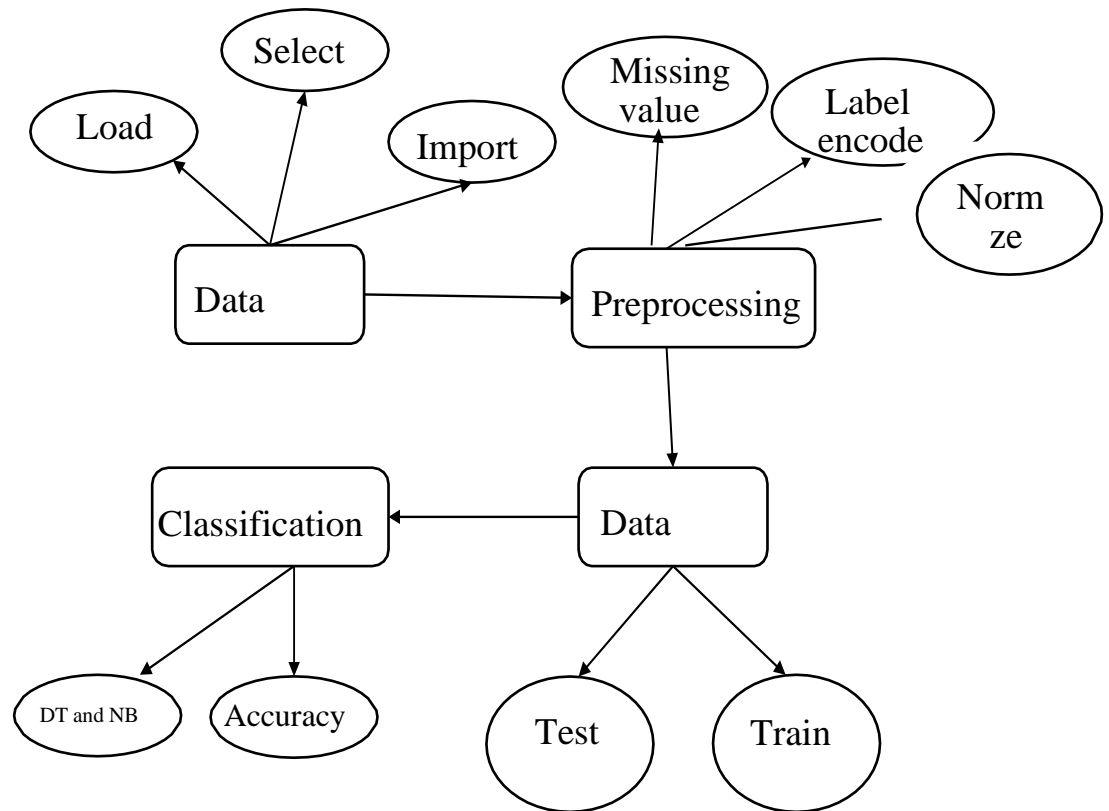
#### 4.6 SEQUENCE DIAGRAM:



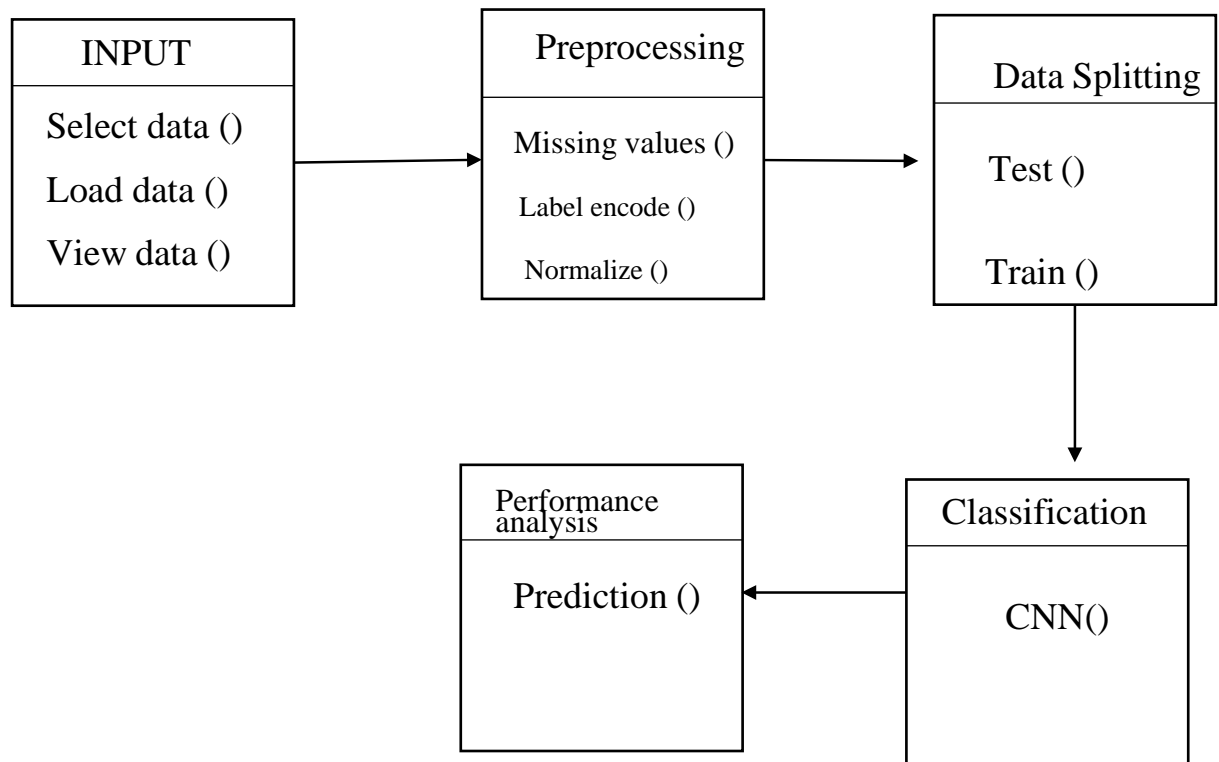
#### 4.7 SEQUENCE DIAGRAM:



#### 4.8 ER DIAGRAM:



#### 4.9 CLASS DIAGRAM:



## CHAPTER 5

### IMPLEMENTATION

#### 5.1 ONMODULES

- Cyber Security Dataset
- AES Algorithm
- Block chain
- Query search
- Key Generation
- Cloud Storage
- Performance Metrics

### MODULE DESCRIPTION

#### 5.2 CYBER SECURITY DATASET

- Cyber security is the use of technologies, processes, and controls to defend against cyber-attacks on systems, networks, programs, devices, and data.
- Its goal is to reduce the risk of cyber-attacks and to protect against unauthorized use of systems, networks, and technologies.
- Cyber security Protocols Reference and Keywords
- In this step, we have to load the data with the help of panda's packages.

#### 5.3 BLOCKCHAIN

**Blockchain defined:** Blockchain is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets in a business network. An *asset* can be tangible (a house, car, cash, land) or intangible (intellectual property, patents, copyrights, branding). Virtually anything of value can be tracked and traded on a blockchain network, reducing risk and cutting costs for all involved. Block chain is a system of recording information in a way that makes it difficult or impossible to change, hack, or cheat the system. A block chain is essentially a digital ledger of transactions that is

uplicated and distributed across the entire network of computer systems on the block chain. Each block in the chain contains a number of transactions, and every time a new transaction occurs on the blockchain, a record of that transaction is added to every participant's ledger. The decentralised database managed by multiple participants is known as Distributed Ledger Technology (DLT).Block chain is a type of DLT in which transactions are recorded with an immutable cryptographic signature called a [hash](#).

## 5.4 ECC and AES Algorithm

- AES Algorithm based on Public and Private Key
- AES based on Encrypted data
- The quick explanation is that keys using Elliptic Curve Cryptography (ECC) are asymmetric (public and private), whereas AES-256 uses a symmetric cypher (key)
- ECC and AES based on it Public and Private key

## Key Generation

- Hybrid AES and ECC based on **128 bit key** Generated For Encrypted data Wise
- A encryption system is designed by combining the characteristics of the **AES** and ECC
- Which Can solve Security Problem itself
- Efficiently realize the information, data encryption, signature, and identity verification.

## CLOUDME

- Encrypted File Will be Stored in data for Security purpose
- A cloud computing model in which data is stored on the Internet via a cloud computing provider who manages and operates data storage as a service
- Cloud Server will be used on Cloud me Software

## 5.5 Query Search

- Query based on semantic Searching
- Easy Find out the Fault Cyber Security through to Query Wise

### **SPLITTING DATASET INTO TRAIN AND TEST DATA**

- Data splitting is the act of partitioning available data into two portions, usually for cross-validator purposes.
- One Portion of the data is used to develop a predictive model and the other to evaluate the model's performance.
- Separating data into training and testing sets is an important part of evaluating data mining models.
- Typically, when you separate a data set into a training set and testing set, most of the data is used for training, and a smaller portion of the data is used for testing.
- To train any machine learning model irrespective what type of dataset is being used you have to split the dataset into training data and testing data.

## **CLASSIFICATION**

Classification is the problem of identifying to which of a set of categories, a new observation belongs to, on the basis of a training set of data containing observations and whose categories membership is known.

**Support Vector Machine** is one of the most powerful methods in machine learning algorithms. It can find a balance between model complexity and classification ability given limited sample information. Compared to other machine learning methods, the SVM has many advantages in that it can overcome the effects of noise and work without any prior knowledge. The SVM is a non-probabilistic binary linear classifier that predicts an input to one of two classes for each given input. It optimizes the linear analysis and classification of hyper plane formation techniques.



**Round Robin** algorithm works on the principle of round-robin, where an equal share of an object is given to each person in turns. Mostly used for multitasking, this is the oldest and simplest scheduling algorithm that offers starvation-free execution. Each ready task has to run turn by turn in a cyclic queue for a limited time period in round-robin (RR).

**K-Nearest Neighbour** is a supervised machine learning algorithm. The algorithm can be used to solve both classification and regression problem statements. The number of nearest neighbours to a new unknown variable that has to be predicted or classified is denoted by the symbol 'K'.

The PC algorithm is **the state-of-the-art constraint based method for causal discovery**. However, runtime of the PC algorithm, in the worst-case, is exponential to the number of nodes (variables), and thus it is inefficient when being applied to high dimensional data, e.g., gene expression datasets.

## **PREDICTION**

Predictive analytics algorithms try to achieve the lowest error possible by either using “boosting” or “bagging”.

**Accuracy** – Accuracy of classifier refers to the ability of classifier. It predict the class label correctly and the accuracy of the predictor refers to how well a given predictor can guess the value of predicted attribute for a new data.

**Speed** – Refers to the computational cost in generating and using the classifier or predictor.

**Robustness** – It refers to the ability of classifier or predictor to make correct predictions from given noisy data.

**Scalability** – Scalability refers to the ability to construct the classifier or predictor efficiently; given large amount of data.

**Interpretability** – It refers to what extent the classifier or predictor understands.

## RESULT GENERATION

The Final Result will get generated based on the overall classification and prediction. The performance of this proposed approach is evaluated using some measures like,

### Accuracy

**Accuracy** of classifier refers to the ability of classifier. It predicts the class label correctly and the accuracy of the predictor refers to how well a given predictor can guess the value of predicted attribute for a new data.

$$AC = \frac{TP+TN}{TP+TN+FP+FN}$$

### Precision

**Precision** is defined as the number of true positives divided by the number of true positives plus the number of false positives.

$$\text{Precision} = \frac{TP}{TP+FP}$$

### Recall

**Recall** is the number of correct results divided by the number of results that should have been returned. In binary classification, recall is called sensitivity. It can be viewed as the probability that a relevant document is retrieved by the query.

$$\text{Recall} = \frac{TP}{TP+FN}$$

### Confusion matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing.

## **CHAPTER 6**

### **6. SYSTEM REQUIREMENTS**

The system requirement is the first step in the requirements analysis process. It lists the requirements of a particular software system including functional, performance and security requirements. The requirements also provide usage scenarios from a user, an operational and an administrative perspective. The purpose of software requirements specification is to provide a detailed overview of the software project, its parameters and goals. This describes the project target audience and its user interface, hardware and software requirements. It define show the client, team and audience see the project and its functionality.

### **HARDWAREANDSOFTWARESPECIFICATION**

#### **6.1 HARDWARESPECIFICATION:**

- System : Pentium IV 2.4 GHz
- Hard Disk : 200 GB
- Mouse : Logitech.
- Keyboard : 110 keys enhanced
- Ram : 4GB

#### **6.2 SOFTWARESPECIFICATION:**

- O/S : Windows 7.
- Language : Python
- Front End : Anaconda Navigator - Spyder

## 6.3 SOFTWARE DESCRIPTION

### Python

Python is one of those rare languages which can claim to be both *simple* and powerful. You will find yourself pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than the syntax and structure of the language you are programming in. The official introduction to Python is Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. I will discuss most of these features in more detail in the next section.

### Features of Python

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

### Easy to Learn

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

## **Free and Open Source**

Python is an example of a FLOSS (Free/Libre and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

## **High-level Language**

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

## **Portable**

Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC!

You can even use a platform like Kivy to create games for your computer and for iPhone, iPad, and Android.

## Interpreted

A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer(binary code i.e. 0s and 1s) using a compiler with various flags and options.

When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.

Python, on the other hand, does not need compilation to binary. You just run the program directly from the source code. Internally, Python converts the source code into an intermediate form called bytecodes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

## Object Oriented

Python supports procedure-oriented programming as well as object-oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

## Extensible

If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program. **Embeddable**

You can embed Python within your C/C++ programs to give scripting capabilities for your program's users.

## **Extensive Libraries**

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the Batteries Included philosophy of Python.

Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

## **6.4 SYSTEM TESTING (Testing of Products)**

System testing is the stage of implementation, which immediately ensures that the system works accurately and efficiently before the live operation commences. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet-un discovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved.. A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to perform, a test can be conducted to demonstrate each function is fully operational and fully exercised.

## **UNIT TESTING:**

Unit testing is the testing of each module and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as ‘module testing’.

The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each mode list found to be working satisfactorily as regard to the expected out put from the module. There are some validation checks for the fields. For example, the validation check is done for verifying the data given by the user where both format and validity of the data entered is included. It is very easy to find error and debug the system.

## **INTEGRATION TESTING:**

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data.

The need for the integrated test is to find the overall system performance.

There are two types of integration testing. They are:

- I. Top-Down Integration Testing
- II. Bottom-Up Integration Testing

## **TESTING TECHNIQUES/STRATEGIES:**

### **WHITE BOX TESTING:**

White Box testing is a test case design method that uses the control structure of the procedural design to drive cases. Using the white box testing methods, we Derived test cases that guarantee that all independent paths with in a module have been exercised at least once.



## **BLACK BOX TESTING:**

- Black box testing is done to find incorrect or missing function
- Interface error
- Errors in external database access □ Performance errors.
- Initialization and termination errors

In 'functional testing', is performed to validate an application conforms to its specifications of correctly performs all its required functions. So this testing is also called 'black box testing'.

It tests the external behaviour of the system. Here the engineered product can be tested knowing the specified function that a product has been designed to perform, tests can be conducted to demonstrate that each function is fully operational.

## **SOFTWARE TESTING STRATEGIES**

### **VALIDATION TESTING:**

After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many, But a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer

### **USER ACCEPTANCE TESTING:**

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes when ever required.

## **OUTPUT TESTING:**

After performing the validation testing, then ex-step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format.

The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system

## CHAPTER 7

### SOURCE CODE

```
importtkinter
importtkinter.messagebox
importsqlite3
fromtkinterimport *
importtkinterastk
fromrandomimport *
importstring

entry_1 = None;
entry_2 = None;
entry_3 = None;
classForFrames(tk.Tk):
def__init__(self, *args, **kwargs):
    tk.Tk.__init__(self, *args, **kwargs)
    container = tk.Frame(self)
    container.pack(side="top", fill="both", expand=True)
    container.grid_rowconfigure(0, weight=1)
    container.grid_columnconfigure(0, weight=1)
self.frames = {}

    forFin (Registerform,Login):
        page_name = F.__name__
        frame = F(parent=container, controller=self)
        self.frames[page_name] = frame
        frame.grid(row=0, column=0, sticky="nsew")
self.show_frame("Registerform")

defshow_frame(self, page_name):
```

```

frame = self.frames[page_name]
frame.tkraise()

```

```

class Registerform(tk.Frame):

```

```

    def __init__(self, parent, controller):

```

```

        tk.Frame.__init__(self, parent)

```

```

        self.controller = controller

```

```

        # convert registered userinfo to json file

```

```

    def regPress():

```

```

        usern = entry_1.get()

```

```

        passw = entry_2.get()

```

```

        conn = sqlite3.connect('users.db')

```

```

        c = conn.cursor()

```

```

        if entry_2.get() == entry_3.get() and not len(entry_1.get()) == 0:

```

```

            c.execute("CREATE TABLE IF NOT EXISTS 'entries' (username TEXT,
password TEXT)")

```

```

        c.execute("INSERT INTO entries(username,password)VALUES(?,?)",(usern,passw))

```

```

        MsgBox = tkinter.messagebox.showinfo("Success", "Registered, click OK to
login")

```

```

        if MsgBox == 'ok':

```

```

            controller.show_frame("Login")

```

```

        conn.commit()

```

```

        if entry_2.get() != entry_3.get():

```

```

            tkinter.messagebox.showinfo("Failed", "Passwords don't match")

```

```

        elif len(entry_1.get()) == 0:

```

```

            tkinter.messagebox.showinfo("Failed", "Please enter a username")

```

```

registerframe1 = Frame(self)
registerframe1.pack(fill=X)

registerframe2 = Frame(self)
registerframe2.pack(fill=X)

registerframe3 = Frame(self)
registerframe3.pack(fill=X)

registerframe6 = Frame(self)
registerframe6.pack(fill=X)

label_1 = tk.Label(registerframe1, text="Username")
label_2 = tk.Label(registerframe2, text="Password")
label_3 = tk.Label(registerframe3, text="Password confirmation")
label_1.pack(side=LEFT, padx=5, pady=5)
label_2.pack(side=LEFT, padx=5, pady=5)
label_3.pack(side=LEFT, padx=5, pady=5)

entry_1 = Entry(registerframe1, width=50)
entry_2 = Entry(registerframe2, width=50, show='*')
entry_3 = Entry(registerframe3, width=50, show='*')

entry_1.pack(side=RIGHT, padx=100)
entry_2.pack(side=RIGHT, padx=100)
entry_3.pack(side=RIGHT, padx=100)

defrandompw():

```

```

characters = string.ascii_letters + string.digits
pwmessage = "".join(choice(characters) for x in range(randint(8, 12)))
print (pwmessage)
registerframePW = Frame(self)
registerframePW.pack(fill=X)

label_PW = tk.Label(registerframePW, text="This is your password. Please,
never share it !")
label_PW.pack()

entryText = tk.StringVar()
entry_PW = Entry(registerframePW, width=50, textvariable=entryText)
entryText.set(pwmessage)
entry_PW.pack()

#### nupud
button1 = tk.Button(self, text="Register", command=regPress)
button2 = tk.Button(self, text="Already have an account? Login",command=lambda:
controller.show_frame("Login"))
button3 = tk.Button(self, text="Create a random password",command=randompw)
button2.pack(side=BOTTOM)
button1.pack(side=TOP,padx=5,pady=5)
button3.pack(side=BOTTOM)
class Login(tk.Frame):
    def __init__(self, parent, controller):

```

```

tk.Frame.__init__(self,parent)

self.controller = controller


#database
def LogPress():
    usern = entry_1.get()
    passw = entry_2.get()
    if usern == "" or passw == "":
        tkinter.messagebox.showinfo("Failed", "Please enter username and password")

    conn = sqlite3.connect('users.db')
    c = conn.cursor()
    c.execute("SELECT * FROM entries WHERE username = ? and password =
    ?",(usern,passw))
    if c.fetchall():
        tkinter.messagebox.showinfo(title = "Successfully logged in", message = "Welcome!!! ")
    else:
        tkinter.messagebox.showerror(title = "Error", message = "incorrect username or password")
    c.close()


registerframe4 = Frame(self)
registerframe4.pack(fill=X)


registerframe5 = Frame(self)
registerframe5.pack(fill=X)

```

```

label_1 = tk.Label(registerframe4, text="Username")
label_2 = tk.Label(registerframe5, text="Password")

label_1.pack(side=LEFT, padx=5, pady=5)
label_2.pack(side=LEFT, padx=5, pady=5)

entry_1 = Entry(registerframe4, width=50)
entry_2 = Entry(registerframe5, width=50, show='*')
entry_1.pack(side=RIGHT, padx=100)
entry_2.pack(side=RIGHT, padx=100)

button1 = tk.Button(self, text="Login", command=LogPress)
button1.pack(side=TOP)

    button2 = tk.Button(self, text="Don't have an account?", command=lambda:
controller.show_frame("Registerform"))
    button2.pack(side=BOTTOM)

def close_window(self):
    self.master.destroy()

if __name__ == "__main__":

    app = ForFrames()
    app.geometry("700x250")

```



```
app.mainloop()
```

## #IMPORTING LIBRARIES

```
import datetime
```

```
import hashlib
```

```
import json
```

```
from tinyec import registry
```

```
from Crypto.Cipher import AES
```

```
import secrets
```

```
import hashlib, binascii
```

```
import pandas as pd
```

```
import numpy as np
```

```
import os
```

## #CREATING BLOCKCHAIN CLASS

```
class Blockchain:
```

```
    def __init__(self):
```

```
        self.chain = []
```

```
        self.create_block(proof=1, previous_hash='0')
```

```
    def create_block(self, proof, previous_hash):
```

```
        block = {'index': len(self.chain) + 1,
```

```
                  'timestamp': str(datetime.datetime.now()),
```

```
                  'proof': proof,
```

```
                  'previous_hash': previous_hash}
```

```
        self.chain.append(block)
```

```
        return block
```

```

def print_previous_block(self):
    return self.chain[-1]

def proof_of_work(self, previous_proof):
    new_proof = 1
    check_proof = False

    while check_proof is False:
        hash_operation = hashlib.sha256(
            str(new_proof**2 - previous_proof**2).encode()).hexdigest()
        if hash_operation[:4] == '0000':
            check_proof = True
        else:
            new_proof += 1

    return new_proof

def hash(self, block):
    encoded_block = json.dumps(block, sort_keys=True).encode()
    return hashlib.sha256(encoded_block).hexdigest()

def chain_valid(self, chain):
    previous_block = chain[0]
    block_index = 1

    while block_index < len(chain):
        block = chain[block_index]
        if block['previous_hash'] != self.hash(previous_block):

```

```
return False
```

```
previous_proof = previous_block['proof']  
proof = block['proof']  
hash_operation = hashlib.sha256(  
    str(proof**2 - previous_proof**2).encode()).hexdigest()
```

```
if hash_operation[:4] != '0000':
```

```
    return False
```

```
previous_block = block
```

```
block_index += 1
```

```
return True
```

## #ECC ENCRYPTION AND DECRYPTION WITH AES

```
def encryption_AES(msg, secretKey):
```

```
    aesCipher = AES.new(secretKey, AES.MODE_GCM)
```

```
    ciphertext, authTag = aesCipher.encrypt_and_digest(msg)
```

```
    return (ciphertext, aesCipher.nonce, authTag)
```

```
def decryption_AES(ciphertext, nonce, authTag, secretKey):
```

```
    aesCipher = AES.new(secretKey, AES.MODE_GCM, nonce)
```

```
    plaintext = aesCipher.decrypt_and_verify(ciphertext, authTag)
```

```
    return plaintext
```

```
def ecc_to_256_bitkey(point):
```

```
    sha = hashlib.sha256(int.to_bytes(point.x, 32, 'big'))
```

```
    sha.update(int.to_bytes(point.y, 32, 'big'))
```

```
    return sha.digest()
```

```
curve = registry.get_curve('brainpoolP256r1')
```

```
defECC_Encryption(msg, pubKey):
```

```
    ciphertextPrivKey = secrets.randbelow(curve.field.n)
```

```
    sharedECCKey = ciphertextPrivKey * pubKey
```

```
    secretKey = ecc_to_256_bitkey(sharedECCKey)
```

```
    ciphertext, nonce, authTag = encryption_AES(msg, secretKey)
```

```
    ciphertextPubKey = ciphertextPrivKey * curve.g
```

```
    return (ciphertext, nonce, authTag, ciphertextPubKey)
```

```
defECC_Decryption(storedMsg, privKey):
```

```
    (ciphertext, nonce, authTag, ciphertextPubKey) = storedMsg
```

```
    sharedECCKey = privKey * ciphertextPubKey
```

```
    secretKey = ecc_to_256_bitkey(sharedECCKey)
```

```
    plaintext = decryption_AES(ciphertext, nonce, authTag, secretKey)
```

```
    returnplaintext
```

```
#
```

---

```
blockchain = Blockchain()
```

```
previous_block = blockchain.print_previous_block()
```

```
previous_proof = previous_block['proof']
```

```
proof = blockchain.proof_of_work(previous_proof)
```

```
previous_hash = blockchain.hash(previous_block)
```

```
block = blockchain.create_block(proof, previous_hash)
```

```
#LOADING DATASET
```

```
df=pd.read_csv('dataset.csv')
```

```
df=df.iloc[:10]
```

```

lak = df.to_numpy().flatten()

encrypt = []
decrypt = []
for j in lak:
    j = str(j)
    msg = str.encode(j)
    privKey = secrets.randbelow(curve.field.n)
    pubKey = privKey * curve.g

    encryptedMsg = ECC_Encryption(msg, pubKey)
    encrypt.append(encryptedMsg)

response = {'message': encryptedMsg,
            'index': block['index'],
            'timestamp': block['timestamp'],
            'proof': block['proof'],
            'previous_hash': block['previous_hash']}
response2 = {'chain': blockchain.chain,
            'length': len(blockchain.chain)}
valid = blockchain.chain_valid(blockchain.chain)

if valid:
    print('The Blockchain is valid.')
    storedMsg=response["message"]
    #print(storedMsg)
    decryptedMsg = ECC_Decryption(storedMsg, privKey)

```

```

decryptedMsg = decryptedMsg.decode('utf-8')
decrypt.append(decryptedMsg)

print("decrypted msg:", decryptedMsg)
else:
    print( 'The Blockchain is not valid.')

```

"Blockchain Encryption and decryption "

```

defAES_Encryption(msg, secretKey):
    aesCipher = AES.new(secretKey, AES.MODE_GCM)
    ciphertext, authTag = aesCipher.encrypt_and_digest(msg)
    return (ciphertext, aesCipher.nonce, authTag)

defAES_Decryption(ciphertext, nonce, authTag, secretKey):
    aesCipher = AES.new(secretKey, AES.MODE_GCM, nonce)
    plaintext = aesCipher.decrypt_and_verify(ciphertext, authTag)
    returnplaintext

defECC_bit_key_generation(point):
    sha = hashlib.sha256(int.to_bytes(point.x, 32, 'big'))
    sha.update(int.to_bytes(point.y, 32, 'big'))
    returnsha.digest()

curve = registry.get_curve('brainpoolP256r1')

defECC_Encryption(msg, pubKey):
    ciphertextPrivKey = secrets.randbelow(curve.field.n)
    sharedECCKey = ciphertextPrivKey * pubKey

```

```

secretKey = ECC_bit_key_generation(sharedECCKey)
ciphertext, nonce, authTag = AES_Encryption(msg, secretKey)
ciphertextPubKey = ciphertextPrivKey * curve.g
return (ciphertext, nonce, authTag, ciphertextPubKey)

```

```

defECC_Decryption(encryptedMsg, privKey):
    (ciphertext, nonce, authTag, ciphertextPubKey) = encryptedMsg
    sharedECCKey = privKey * ciphertextPubKey
    secretKey = ECC_bit_key_generation(sharedECCKey)
    plaintext = AES_Decryption(ciphertext, nonce, authTag, secretKey)
    returnplaintext

```

```

# 

---


df1 = pd.read_csv("dataset.csv")
df1=df1.iloc[:10]
df1.shape

column_names = list(df.columns)

result = df.values

print("Encrypting and Decrypting the CSV file...")
empty = []
empty_decoded = []
foriinresult:
    forjini:
        a = str(j)
        en = a.encode()
        s = ECC_Encrytion(en, pubKey)

```

```

b = binascii.hexlify(s[0])
encoded_text = b.decode('utf-8')
empty.append(encoded_text)
#print(f"Encoded Text : {encoded_text}")

de = ECC_Decryption(s, privKey)
decoded_text = de.decode('utf-8')
empty_decoded.append(decoded_text)
#print(f"Decoded Text : {decoded_text}")

encrypted_df = pd.DataFrame(np.array(empty).reshape(10,45),columns = column_names)

print("Encryption Completed and written as encryption.csv file")
encrypted_df.to_csv(r'encrypted.csv',index = False)

print("decryption Completed and written as Decryption.csv file")

decrypted_df = pd.DataFrame(np.array(decrypt).reshape(10,45),columns =df.columns)
decrypted_df.to_csv(r'decryption.csv',index = False)

decrypted_df.head()

"Import Libraries "

import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split

```



```
from sklearn.metrics import accuracy_score
from sklearn import metrics
```

```
print("=====")
print("Block chain in the 5G/6G technology Dataset")
print(" Process - Block chain in the 5G/6G technology Attack Detection")
print("=====")
```

```
##1.data slection_____
```

```
#def main():
```

```
dataframe=pd.read_csv("dataset.csv")
```

```
print("_____")
```

```
print()
```

```
print("Data Selection")
```

```
print("Samples of our input data")
```

```
print(dataframe.head(10))
```

```
print("_____")
```

```
print()
```

```
#2.pre processing_____
```

```
#checking missing values
```

```
print("_____")
```

```
print()
```

```
print("Before Handling Missing Values")
```

```
print()
print(dataframe.isnull().sum())
print("_____")
print()
```

```
print("_____")
print("After handling missing values")
print()
dataframe_2=dataframe.fillna(0)
print(dataframe_2.isnull().sum())
print()
print("_____")
```

```
#label encoding
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
print("_____")
print("Before Label Handling ")
print()
print(dataframe_2.head(10))
print("_____")
print()
```

```
#3.Data splitting_____
```

```
df_train_y=dataframe_2["label"]
df_train_X=dataframe_2.iloc[:,20]
from sklearn.preprocessing import LabelEncoder
```

```
number = LabelEncoder()
```

```
df_train_X['proto'] = number.fit_transform(df_train_X['proto'].astype(str))
```

```
df_train_X['service'] = number.fit_transform(df_train_X['service'].astype(str))
```

```
df_train_X['state'] = number.fit_transform(df_train_X['state'].astype(str))
```

```
#df_train_X['attack_cat'] = number.fit_transform(df_train_X['attack_cat'].astype(str))
```

```
print("=====")
```

```
print(" Preprocessing")
```

```
print("=====")
```

```
df_train_X.head(5)
```

```
x=df_train_X
```

```
y=df_train_y
```

```
##4.feature selection_____
```

```
##kmeans
```

```
from sklearn.datasets import make_blobs
```

```
from sklearn.cluster import KMeans
```

```
import matplotlib.pyplot as plt
```

```
x, y_true = make_blobs(n_samples=175341, centers=4, cluster_std=0.30, random_state=0)
```

```
plt.scatter(x[:, 0], x[:, 1], s=20);
```

```
kmeans = KMeans(n_clusters=3)
```

```
kmeans.fit(x)
```

```
y_kmeans = kmeans.predict(x)
```

```

plt.scatter(x[:, 0], x[:, 1], c=y_kmeans, s=20, cmap='viridis')

centers = kmeans.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5);

plt.title("k-means")
plt.show()

# _____
x_train,x_test,y_train,y_test = train_test_split(df_train_X,y,test_size = 0.20,random_state =
42)

from sklearn.ensemble import RandomForestClassifier

rf= RandomForestClassifier(n_estimators = 100)
rf.fit(x_train, y_train)
rf_prediction = rf.predict(x_test)
Result_3=accuracy_score(y_test, rf_prediction)*100
from sklearn.metrics import confusion_matrix

print()
print("_____")
print("Random Forest")
print()
print(metrics.classification_report(y_test,rf_prediction))
print()
print("Random Forest Accuracy is:",Result_3,'%')
print()
print("Confusion Matrix:")

```

```

cm2=confusion_matrix(y_test, rf_prediction)
print(cm2)
print("_____")
print()
import matplotlib.pyplot as plt
import seaborn as sns
sns.heatmap(cm2, annot = True, cmap = 'plasma',
            linecolor = 'black', linewidths = 1)
plt.show()

```

```

# _____

```

```

from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier(criterion = "gini", random_state = 100, max_depth=3,
min_samples_leaf=5)
dt.fit(x_train, y_train)
dt_prediction=dt.predict(x_test)
print()
print("_____")
print("Decision Tree")
print()
Result_2=accuracy_score(y_test, dt_prediction)*100
print(metrics.classification_report(y_test,dt_prediction))
print()
print("DT Accuracy is:",Result_2,'%')
print()
print("Confusion Matrix:")

```

```

from sklearn.metrics import confusion_matrix
cm1=confusion_matrix(y_test, dt_prediction)
print(cm1)
print("_____")
print()
import matplotlib.pyplot as plt
import seaborn as sns

sns.heatmap(cm1, annot = True, cmap = 'plasma',
            linecolor = 'black', linewidths = 1)
plt.show()
#ROC graph

#_____

from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

from sklearn.ensemble import GradientBoostingClassifier
gradient_booster = GradientBoostingClassifier(learning_rate=0.1)
gradient_booster.get_params()

gradient_booster.fit(x_train,y_train)
gb_prediction = gradient_booster.predict(x_test)

print(classification_report(y_test,gradient_booster.predict(x_test)))

Result_2=accuracy_score(y_test, gb_prediction)*100
print()
print("gradient_booster Accuracy is:",Result_2,'%')

```

```

print()
print("Confusion Matrix:")
from sklearn.metrics import confusion_matrix
cm1=confusion_matrix(y_test, dt_prediction)
print(cm1)
print("_____")
print()
#_____

```

"Navie Bayies "

```

from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(x_train, y_train)

```

# Predicting the Test set results

```

y_pred = classifier.predict(x_test)

```

# Making the Confusion Matrix

```

from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

```

```

print("Navie Bayies Accuracy is:",Result_2,'%')

```

```

print()

```

```

print("Confusion Matrix:")

```

```

from sklearn.metrics import confusion_matrix
cm1=confusion_matrix(y_test, y_pred)

```

```

print(cm1)

```

```

print("_____")

```

```

print()

```

```

# -----

from easygui import *
Key = "Enter the DOS Id to be Search"

# window title
title = "DOS Fault Id "
# creating a integer box
str_to_search1 = enterbox(Key, title)
input = int(str_to_search1)

import tkinter as tk
if (rf_prediction[input] == 0):
    print("Non Attack ")
    root = tk.Tk()
    T = tk.Text(root, height=20, width=30)
    T.pack()
    T.insert(tk.END, "Non Attack ")
    tk.mainloop()
elif (rf_prediction[input] == 1):
    print("Attack ")
    root = tk.Tk()
    T = tk.Text(root, height=20, width=30)
    T.pack()
    T.insert(tk.END, "Attack ")
    tk.mainloop()
import smtplib as smtp

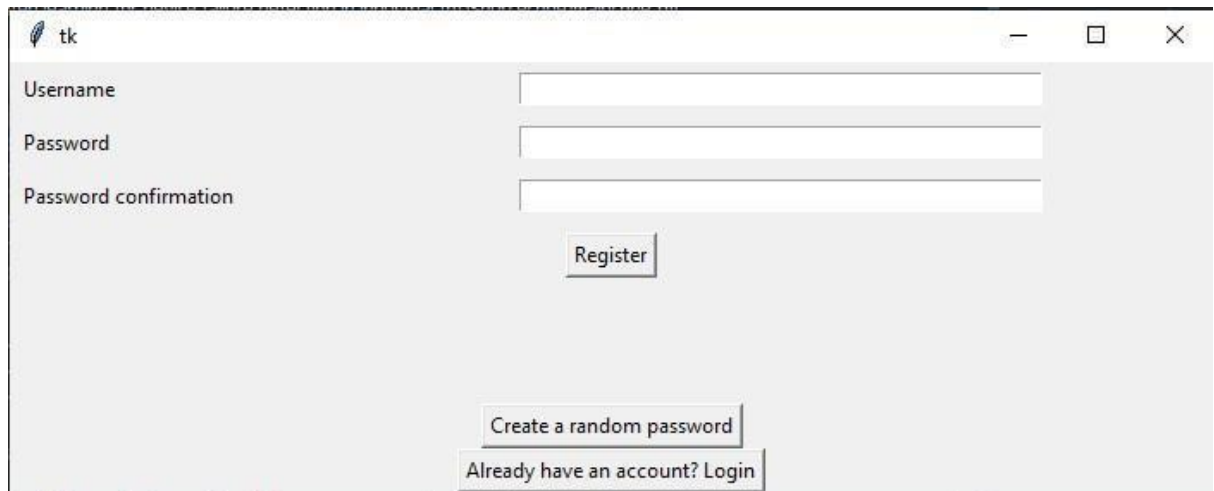
```



```
connection = smtp.SMTP_SSL('smtp.gmail.com', 465)

email_addr = 'sathyakumar17112022@gmail.com'
email_passwd = 'ncfplwzacztnxyp'
connection.login(email_addr, email_passwd)
connection.sendmail(from_addr=email_addr,
to_addrs='sathyakumar17112022@gmail.com', msg="Attack kindly prevent system ")
connection.close()
```

## SCREENSHOT



A screenshot of a Tkinter window titled "tk" with standard window controls (minimize, maximize, close). The window contains a registration form with the following elements:

- Labels: "Username", "Password", and "Password confirmation" are positioned on the left side.
- Input fields: Three text entry boxes are aligned to the right of the labels.
- Buttons: A "Register" button is centered below the input fields. At the bottom of the window, there are two buttons: "Create a random password" and "Already have an account? Login".



A second screenshot of the same Tkinter window, now containing user input:

- The "Username" field contains the text "mutheswari".
- The "Password" field contains seven asterisks "\*\*\*\*\*".
- The "Password confirmation" field contains seven asterisks "\*\*\*\*\*" followed by a vertical cursor line.
- The "Register" button remains centered below the fields.
- The buttons "Create a random password" and "Already have an account? Login" are still present at the bottom.

```
Console 1/A x
The Blockchain is valid.
decrypted msg: 0
The Blockchain is valid.
decrypted msg: 3
The Blockchain is valid.
decrypted msg: 1.623129
The Blockchain is valid.
decrypted msg: tcp
The Blockchain is valid.
decrypted msg: -
The Blockchain is valid.
decrypted msg: FIN
The Blockchain is valid.
decrypted msg: 8
The Blockchain is valid.
```

```
Random Forest Accuracy is: 98.94208560266902 %
Confusion Matrix:
[[10891  278]
 [   93 23807]]
-----
```

```

      0      1.00      0.89      0.94      11169
      1      0.95      1.00      0.98      23900

accuracy
macro avg      0.98      0.95      0.97      35069
weighted avg    0.97      0.97      0.97      35069

DT Accuracy is: 96.60098662636517 %
Confusion Matrix:
[[ 9977  1192]
 [    0 23900]]
-----
```

```
gradient_booster Accuracy is: 97.51632495936582 %
Confusion Matrix:
[[ 9977  1192]
 [    0 23900]]
```

## **CHAPTER 9**

### **FUTURE ENHANCEMENT**

- It is based on Future Hybrid algorithm for ECC and AES Algorithm
- You may increase the Efficiency of the Algorithm.
- we Can future implementation on testing software for IOT Security

## CHAPTER 10

### REFERENCES

- [1] L. Zhang, Y. Zhang and H. Ma, “Privacy-Preserving and Dynamic Multi-Attribute Conjunctive Keyword Search Over Encrypted Cloud Data”, *IEEE Access*, vol. 6, pp. 34214-34225, 2018.
- [2] Z. Xiang yang, D. Hua, Y. Xun, Y. Geng, and L. Xiao, “MUSE: An Efficient and Accurate Verifiable Privacy Pre-serving Multi-keyword Text Search over Encrypted Cloud Data”, *Security and Communication Networks*, vol. 2017, pp. 1-17, 2017.
- [3] L. Chen, L. Qiu, K-C. Li, W. Shi, and N. Zhang, "DMRS: an efficient dynamic multi-keyword ranked search over encrypted cloud data", *Soft Computing*, vol. 21(16), pp. 4829–4841, 2017.
- [4] R. Zhang, R. Xue, L. Liu, and L. Zheng, “Oblivious Multi-Keyword Search for Secure Cloud Storage Service”, 2017 IEEE 24th International Conference on Web Services, pp. 269-276, 2017.
- [5] P. K. Samantaray, N. K. Randhawa, and S. L. Pati, “An Efficient Multi-keyword Text Search Over Outsourced Encrypted Cloud Data with Ranked Results”, *Computational Intelligence in Data Mining*, pp. 31-40, 2018.
- [6] Z. Fu, F. Huang, K. Ren, J. Weng, and C. Wang, “Privacy-Preserving Smart Semantic Search Based on Conceptual Graphs Over Encrypted Outsourced Data”, *IEEE Transactions On Information Forensics And Security*, vol. 12(8), pp. 1874-1884, 2017.

- [7] B. Lang, J. Wang, M. Li, and Y. Liu, “Semantic-based Compound Keyword Search over Encrypted Cloud Data”, *IEEE Transactions On Services Computing*, 2018.
- [8] Z. Fu, L. Xia, X. Sun, A. X. Liu, and G. Xie, “Semantic-aware Searching over Encrypted Data for Cloud Computing”, *IEEE Transactions on Information Forensics and Security*, vol. 13(9),pp. 2359-2371, 2018.
- [9] Z. Wu, and K. Li, “VB Tree: forward secure conjunctive queries over encrypted data for cloud computing”, *The VLDB Journal*, pp. 1–22, 2018.
- [10] Y. Yang, X. Liu, and R. H. Deng, “Multi-user Multi-Keyword Rank Search over Encrypted Data in Arbitrary Language”, *IEEE Transactions on Dependable and Secure Computing*, 2018.
- [11] X. Ding, P. Liu, and H. Jin, “Privacy-Preserving Multi-keyword Top- $k$  Similarity Search Over Encrypted Data”, *IEEE Transactions on Dependable and Secure Computing*, 2018.
- [12] Y. Li, F. Zhou, Y. Qin, M. Lin, and Z. Xu, “Integrity-verifiable conjunctive keyword searchable encryption in cloud storage”, *International Journal of Information Security*, vol. 17(5), pp. 549–568, 2018.
- [13] C. Guo, X. Chen, Y. Jie, Z. Fu, M. Li, and B. Feng, “Dynamic Multi-phrase Ranked Search over Encrypted Data with Symmetric Searchable Encryption”, *IEEE Transactions On Services Computing*, 2018.
- [14] Raghavendra S, Girish S, Geeta C. M., R. Buyya, Venugopal K. R., S. S. Iyengar, and L. M. Patnaik, “Split keyword fuzzy and synonym search over encrypted cloud data”, *Multimedia Tools and Applications*, vol. 77(8), pp. 10135–10156, 2018.

- [15] A. V. Vora, and S. Hegde, “Keyword-based private searching on cloud data along with keyword association and dissociation using cuckoo filter”, *International Journal of Information Security*, pp. 1–15, 2018.
- [16] H. Wang, X. Dong, and Z. Cao, “Secure and efficient encrypted keyword search for multi-user setting in cloud computing”, *Peer-to-Peer Networking and Applications*, pp. 1–11, 2018.

## CONCLUSION

- Hybrid ECC and AES Algorithm using the Encrypted and Decrypted the dataset
- Encrypted File will be Stored in Cloud Server and User based on Semantic Searching method Algorithm.
- User based keyword Entering is done to retrieve the corresponding data file from the cloud storage.
- Finally Retrieve the Related File based on Query.
- This will easily Find out Cyber security Problem like, the fault file will be detected.