



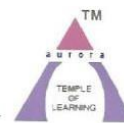
**A**  
**STUDY ON**  
**“AUTHENTICATION TO PRODUCT AND COUNTERFIETS**  
**ELIMINATION USING BLOCK CHAIN”**

**Submitted by**  
**P.BHANODAY**  
(H.T.No.2122-21-862-017)

**Project submitted in partial fulfillment for the award of the Degree of**  
**MASTER OF COMPUTER APPLICATIONS**



**By**  
**Department of Computer Applications**  
**Aurora's Post Graduate College (MBA)**  
**Punjagutta, Hyderabad.**  
**2021-2023**



## **CERTIFICATE**

This is to certify that, this project entitled “**AUTHENTICATION TO PRODUCT AND COUNTERFIETS ELIMINATION USING BLOCK CHAIN**” is submitted by **P.BHANODAY** ,H.T.No. (2122-21862-017) as part of their curriculum in the Department of Computer Applications in partial fulfilment for the award of **Master of Computer Applications**. This work has been carried out under our guidance and has not been submitted to any other University or Institution for the award of any degree/diploma/certificate.

INTERNAL GUIDE

EXTERNAL EXAMINER

HEAD OF THE DEPARTMENT

PRINCIPAL

Date: 15/09/2023

## CERTIFICATE

This is to certify that **Mr. P.BHANODAY** Bearing the **Hall Ticket No. 212221862017** from **AURORA'S PG COLLEGE** Student of **M.C.A** has successfully completed the Major Project titled "**AUTHENTICATION TO PRODUCT AND COUNTERFIETS ELIMINATION USING BLOCK CHAIN**" in our organization.

He has done the project using **Python Technology** during the period of **June 5<sup>th</sup> 2023** to **September 15<sup>th</sup> 2023**, under the guidance and supervision of **Mr. Akash** from **MANAC INFOTECH PVT LTD, Hyderabad**. He has completed the assigned project well within the time framework. He is sincere, hardworking and his conduct during period is commendable.



(Project Manager)

An IIM Alumnus Enterprise

LIBERTY: #205, Sagar View Complex, Opp. GHMC Office, Near Tankbund Ambedkar Statue. Ph: 9666607505.

DILSUKHNAGAR: 1st Floor, Above Airtel Office, Near Metro Pillar No. MSBNP-28. Ph: 9291430931.

Toll Free: - 1800-425-1839

[www.manacinfotech.com](http://www.manacinfotech.com)

## ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely privileged to have got this all along the completion of my project. All that I have done is only due to such supervision and assistance and I would not forget to thank them.

I consider myself lucky enough to get such a good project. This project would be added as an asset to my academic profile.

I would like to express my thankfulness to my Principal and Head of the Department for their constant motivation and valuable help through the project work.

I owe my deep gratitude to my project guide **Mr. Y NASIR AHMED**, who took keen interest in my project work and guided me all along, till the completion of my project work by providing all the necessary information for developing a good system. I would like to thank my friends who helped me in improving my thesis.

I am thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staff of MCA which helped me in successfully completing our project work. Also, I would like to extend our sincere esteems to all staff in the laboratory for their timely support.

Finally, I wish to thank my parents for their love and encouragement, without whom I would never have enjoyed so many opportunities.

P.BHANODAY

(2122-21-862-017)

## **DECLARATION**

I hereby declare that this Project Report titled “**AUNTHETICATION TO PRODUCT AND COUNTERFIETS ELIMINATION USING BLOCK CHIAN**”, submitted by me to the Department of Computer Applications, O.U., Hyderabad, is a bonafide work undertaken by me and it is not submitted to any other University or Institution for the award of any degree diploma/certificate or published any time before.

Signature of the Student

## **ABSTRACT**

Blockchain technologies have gained interest over the last years. While the most explored use case is financial transactions, it has the capability to agitate other markets. Blockchain remove the need for trusted intermediaries, can facilitate faster transactions and add more transparency. This paper explores the possibility to deflate counterfeit using blockchain technology. This paper provides an overview of different solutions in the anti-counterfeit area, different blockchain technologies and what characteristics make blockchain especially interesting for the use case. We have developed three different concepts and the expansion of an existing system concept, is pursued further. It is shown, that reducing counterfeits cannot be achieved by using technological means only. Increasing awareness, fighting counterfeiters on a legal level, a good alert system, and having tamper-proof packaging are all important aspects. These factors combined with blockchain technology can lead to an efficient and comprehensive approach to reduce counterfeiting

Keywords – Authentication, Blockchain, Encryption

	<b>INDEX</b>	
<b>1</b>	<b>INTRODUCTION</b>	
	1.1 Purpose of project	1
<b>2</b>	<b>REQUIREMENT</b>	
	2.1 Literature Survey	2-5
	2.2 Problem with the Existing System	6
	2.3 Proposed System	7
<b>3</b>	<b>SYSTEM ANALYSIS</b>	
	3.1 System study	8-9
	3.2 Software Requirement Specification	10-11
	3.2.1 Functional Requirements	10
	3.2.2 Non functional Requirements	11
	3.3 System Requirements	12
	3.4 Modules	13
<b>4</b>	<b>SYSTEM DESIGN</b>	
	4.1 Introduction to System Design	15
	4.1.1 Input/output Design	16-17
	4.2 Introduction to UML	18
	4.3 UML Diagrams	19-23
	4.3.1 UseCase Diagram	19
	4.3.2 Class Diagram	20
	4.3.3 Sequence Diagram	21
	4.3.4 Activity Diagram	22

	4.3.5 Collaboration	23
<b>5</b>	<b>SYSTEM IMPLEMENTATION</b>	
	5.1 About Technology	24-39
	5.2 Source code	40-43
	5.3 Output Screens	44-49
<b>6</b>	<b>SOFTWARE TESTING</b>	
	6.1 Introduction to Testing	50
	6.2 Test Cases	50-52
<b>7</b>	<b>CONCLUSION AND DISCUSSION</b>	
	7.1 Conclusion	53
	7.2 Future Enhancements	54



**CHAPTER 1**  
**INTRODUCTION**  
**“AUTHENTICATION TO PRODUCT AND**  
**COUNTERFEITS ELIMINATION USING**  
**BLOCKCHAN”**

# **1.INTRODUCTION**

## **1.1.Purpose**

Although it may seem like a far off idea, we are surrounded by a lot of counterfeits. From fashion and retail products to software, digital media, electronics, piracy, and intellectual property, reports put the cost of counterfeiting somewhere around \$600bn a year in the US alone. In fact, the International Chamber of Commerce predicts that the —negative impacts of counterfeiting and piracy are projected to drain US\$4.2 trillion from the global economy and put 5.4 million legitimate jobs at risk by 2022. In Pharmaceuticals, the counterfeit medicine market is now responsible for around 1 million deaths per year, in an industry estimated to be worth \$75bn annually.

In fact, the counterfeit medicine industry is estimated to be growing at twice the rate of legitimate pharmaceuticals, making it up to 25 times more lucrative than the global narcotics trade. Trust is a central element in all transactions. No matter if sending money or exchanging goods, it becomes difficult if there is no trust between the entities involved. It becomes even more difficult, as with many transactions, third parties are involved, such as banks. Often, not only one third-party is involved in a transaction, but multiple. An international money transfer does not only include the bank of the sender, the bank of the receiver, but also multiple intermediary entities such as clearing houses. The entities involved in the transaction do not only have to trust each other, but also the third parties.

Removing these third parties can decrease transaction cost, facilitate faster transactions and add more transparency. Bitcoin has successfully shown that removing such third-parties is possible. The cryptocurrency permits direct sending coins to a transaction partner, without the need to use banks and clearing houses.

The assets are directly transferred from one account to another. There are no intermediaries and thereby no need to trust third parties. In addition, the question if a transaction is valid is not answered by an institution, but by algorithms used. Therefore, it completely removes the need to trust any third party. The technology behind Bitcoin, the blockchain, can however not only be used for financial transactions and crypto currencies in general. The technology has potential to —redefine the digital economy [10], because it allows immutable transactions, which can be checked at all times from everyone.

This is because the information is publicly available and distributed globally. It is —chronologically updated and cryptographically sealed [11]. The full range of applicable use cases for this technology has to be seen, but tracking ownership and history of a product is surely one of them [12]. This paper explores the possibility to reduce counterfeit using blockchain technology.

# **CHAPTER 2**

## **REQUIREMENTS**

## **2. REQUIREMENTS**

### **2.1. LITERATURE SURVEY**

#### **1. Bitcoin: A Peer-to-Peer Electronic Cash System**

A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size and cutting off the possibility for small casual transactions, and there is a broader cost in the loss of ability to make non-reversible payments for nonreversible services. With the possibility of reversal, the need for trust spreads. Merchants must be wary of their customers, hassling them for more information than they would otherwise need. A certain percentage of fraud is accepted as unavoidable. These costs and payment uncertainties can be avoided in person by using physical currency, but no mechanism exists to make payments over a communications channel without a trusted party.

What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party. Transactions that are computationally impractical to reverse would protect sellers from fraud, and routine escrow mechanisms could easily be implemented to protect buyers. In this paper, we propose a solution to the double-spending problem using a peer-to-peer distributed timestamp server to generate computational proof of the chronological order of transactions. The

system is secure as long as honest nodes collectively control more CPU power than any cooperating group of attacker nodes

## **2. Hyperledger Blockchain Performance Metrics**

This is the first white paper from the [Hyperledger Performance and Scale Working Group](#). The purpose of this document is to define the basic terms and key metrics that should be used to evaluate the performance of a blockchain and then communicate the results. This paper also serves as a platform-agnostic resource for technical blockchain developers and managers interested in using industry-standard nomenclature.

While we appreciate that there may be discrete definitions for the terms “blockchain” and “Distributed Ledger Technology (DLT),” for the purposes of this paper we will treat both terms synonymously and use the term “blockchain” throughout.

This document provides some guidance on selecting and evaluating workloads. We expect that refinements to these definitions and new blockchain-specific metrics will warrant future revisions of this document.

In future documents, the Working Group plans to discuss workloads in greater detail and to offer additional guidance on standard procedures and emerging best practices for evaluating blockchain performance. To provide your feedback and stay informed about subsequent versions of this paper, please join us in the [Performance and Scale Working Group](#).

## **3. Protocols for public key cryptosystems**

New cryptographic protocols which take full advantage of the unique properties of public key cryptosystems are now evolving. Several protocols for public key distribution and for digital signatures are briefly compared with each other and with the conventional alternative.

## **4. Ethereum: A secure decentralised generalized transaction ledger**

The blockchain paradigm when coupled with cryptographically-secured transactions has demonstrated its utility through a number of projects, with Bitcoin being one of the most notable ones. Each such project can be seen as a simple application on a decentralised, but singleton, compute resource. We can call this paradigm a transactional singleton machine with shared-state. Ethereum implements this paradigm in a generalised manner. Furthermore it provides a plurality of such resources, each with a distinct state and operating code but able to interact through a message-passing framework with others. We discuss its design, implementation issues, the opportunities it provides and the future hurdles we envisage

## **5. Practical byzantine fault tolerance and proactive recovery**

Our growing reliance on online services accessible on the Internet demands highly available systems that provide correct service without interruptions. Software bugs, operator mistakes, and malicious attacks are a major cause of service interruptions and they can cause arbitrary behavior, that is, Byzantine faults. This article describes a new replication algorithm, BFT, that can be used to build highly available systems that tolerate Byzantine faults. BFT can be used in practice to implement real services: it performs well, it is safe in asynchronous environments such as the

Internet, it incorporates mechanisms to defend against Byzantine-faulty clients, and it recovers replicas proactively. The recovery mechanism allows the algorithm to tolerate any number of faults over the lifetime of the system provided fewer than  $1/3$  of the replicas become faulty within a small window of vulnerability. BFT has been implemented as a generic program library with a simple interface. We used the library to implement the first Byzantine-fault-tolerant NFS file system, BFS. The BFT library and BFS perform well because the library incorporates several important optimizations, the most important of which is the use of symmetric cryptography to authenticate messages. The performance results show that BFS performs 2% faster to 24% slower than production implementations of the NFS protocol that are not replicated. This supports our claim that the BFT library can be used to build practical systems that tolerate Byzantine faults.

## **6. Making byzantine fault tolerant systems tolerate byzantine faults**

This paper argues for a new approach to building Byzantine fault tolerant replication systems. We observe that although recently developed BFT state machine replication protocols are quite fast, they don't tolerate Byzantine faults very well: a single faulty client or server is capable of rendering PBFT, Q/U, HQ, and Zyzzyva virtually unusable. In this paper, we (1) demonstrate that existing protocols are dangerously fragile, (2) define a set of principles for constructing BFT services that remain useful even when Byzantine faults occur, and (3) apply these principles to construct a new protocol, Aardvark. Aardvark can achieve peak performance within 40% of that of the best existing protocol in our tests and provide a significant fraction of that performance when up to  $f$  servers and any number of clients are faulty. We observe useful throughputs between 11706 and 38667 requests per second for a broad range of injected faults.

## **7. Architecture of the hyperledger blockchain fabric**

A blockchain is best understood in the model of state-machine replication [8], where a service maintains some state and clients invoke operations that transform the state and generate outputs. A blockchain emulates a "trusted" computing service through a distributed protocol, run by nodes connected over the Internet. The service represents or creates an asset, in which all nodes have some stake. The nodes share the common goal of running the service but do not necessarily trust each other for more. In a "permissionless" blockchain such as the one underlying the Bitcoin cryptocurrency, anyone can operate a node and participate through spending CPU cycles and demonstrating a "proof-of-work." On the other hand, blockchains in the "permissioned" model control who participates in validation and in the protocol; these nodes typically have established identities and form a consortium. A report of Swanson compares the two models

The Hyperledger Project ([www.hyperledger.org](http://www.hyperledger.org)) is a collaborative effort to create an enterprisegrade, open-source distributed ledger framework and code base. It aims to advance blockchain technology by identifying and realizing a cross-industry open standard platform for distributed ledgers, which can transform the way business transactions are conducted globally. Established as a project of the Linux Foundation in early 2016, the Hyperledger Project currently has more than 50 members.

Hyperledger Fabric ([github.com/hyperledger/fabric](https://github.com/hyperledger/fabric)) is an implementation of a distributed ledger platform for running smart contracts, leveraging familiar and proven technologies, with a modular architecture allowing pluggable implementations of various functions. It is one of multiple projects currently in incubation under the Hyperledger Project. A developerpreview of the

Hyperledger Fabric (called “v0.5-developer-preview”) has been released in June 2016 ([github.com/hyperledger/fabric/wiki/Fabric-Releases](https://github.com/hyperledger/fabric/wiki/Fabric-Releases)).

## **2.2.**

## **EXISTING SYSTEM**

Many businesses rely on third-party vendors. Because the outsourced supplier has access to all of the original assets, there is a risk that they will not only make legitimate products, but also counterfeits. Outsourcers should be vetted and managed carefully. Another alternative is to not outsource the entire product to a single firm, but to divide the manufacturing of the product among several companies or to keep some of the production in-house. This ensures that no single foreign firm has all of the resources necessary to manufacture counterfeit goods.

### **Drawbacks :**

- Because the outsourced supplier has access to all of the original assets, there is a risk that they will not only make legitimate products, but also counterfeits.
- Online transactions necessitate the employment of a third party to complete the transaction, and customers must trust these third parties to complete their transactions.
- However, these third parties can sometimes commit fraud or misuse user data



## **2.3.**

# **PROPOSED SYSTEM**

Blockchain technology does not require the engagement of a third party, and verification will be carried out by a software algorithm without the need for a third party.

To avoid forging counterfeits, we are converting all product details/barcodes into digital signatures, which will be stored in a blockchain server, which supports tamperproof data storage, meaning that no one can hack or alter its data, and if its data is altered by chance, verification will fail at the next block storage, and the user will be notified.

### **Advantages:**

- Each data will be recorded in blockchain by checking old hash codes;
- If the old hash codes remain unchanged, the data will be considered original and unchanged, and new transaction data will be added to the blockchain as a new block.
- For each new data storage, the hash code of all blocks will be validated.

# **CHAPTER 3**

## **SYSTEM ANALYSIS**

## **3.SYSTEM ANALYSYS**

### **3.1. SYSTEM STUDY**

#### **FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

**Three key considerations involved in the feasibility analysis are,**

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

#### **ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

#### **TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

#### **SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the

users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

### 3.2.

## **SOFTWARE REQUIREMENT SPECIFICATION**

This Chapter describes about the requirements. It specifies the hardware and software requirements that are required in order to run the application properly. The Software Requirement Specification (SRS) is explained in detail, which includes overview of this dissertation as well as the functional and non-functional requirement of this dissertation.

### **SRS**

<b>Functional</b>	service provider browse the file and sends to end user, router sends file from source to destination by selecting shortest path, router drops the packets at nodes if they have less energy, Auditor stores all dropped packets details, Auditor may discovers the traffic pattern details, end user receives the files from router, end users can also gets dropped packets and attacker make changes of nodes energy.
<b>Non- Functional</b>	The Sender and Receiver never Find the attackers.
<b>External interface</b>	LAN , Routers
<b>Performance</b>	Browse, upload, assign energy, assign distances, view distances, view energy, view files, view attackers, verify, refresh, discover traffic pattern, get dropped packets, modify energy and exit.
<b>Attributes</b>	Anonymous Communication, Mobile Ad Hoc Networks, Statistical Traffic Analysis, Service Provider, End Users, Attackers.

Table: 3.1 Summaries of SRS

### **3.2.1. Functional Requirements**

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. In this system following are the functional requirements:-

- The service provider browses the file and sends to the particular end users via router.
- .The service provider can also assign energy and assigns distance for nodes.
- The router will send the all files from source to particular destination, by selecting shortest path & node energy.
- The packet dropper in router may drops the packets from file, suppose if node has the less energy than file size.
- The router can also view distances, view energy, view files, view attackers, verify, refresh.
- The Auditor stores the dropped packet details on it and also it can discovers the traffic pattern details.
- The Remote user has to receive the file from router by without changing the file Contents.
- If packets are dropped from file then end users will gets dropped packets from point to point manager.
- Attacker is one who makes changes the energy sizes of nodes in router.
- The Attributes are Anonymous Communication, Mobile Ad Hoc Networks, Statistical Traffic Analysis, Service Provider, End Users and Attackers.

### **3.2.2. Non – Functional Requirements**

Non – Functional requirements, as the name suggests, are those requirements that are not directly concerned with the specific functions delivered by the system. They may relate to emergent system properties such as reliability response time and store occupancy. Alternatively, they may define constraints on the system such as the capability of the Input Output devices and the data representations used in system interfaces. Many non-functional requirements relate to the system as whole rather than to individual system features. This means they are often critical than the individual functional requirements. The following non-functional requirements are worthy of attention.

#### **The key non-functional requirements are:**

- Security: The system should allow a secured communication between Service provider and Router and Receiver.
- Energy Efficiency: The Time consumed by the Router, transfer the File's to the Receiver.
- Reliability: The system should be reliable and must not degrade the performance of the existing system and should not lead to the hanging of the system.

### 3.3.

## **SYSTEM REQUIREMENTS**

This section elaborates on the functional requirements of the application. The SRS itself can be divided into module, each module having specifications. In order to carry out the project, the following hardware and software is required,.

### **SYSTEM SPECIFICATION:**

### **HARDWARE REQUIREMENTS:**

- ❖ **System** : Pentium IV 2.4 GHz.
- ❖ **Hard Disk** : 40 GB.
- ❖ **Floppy Drive** : 1.44 Mb.
- ❖ **Monitor** : 14' Colour Monitor.
- ❖ **Mouse** : Optical Mouse.
- ❖ **Ram** : 512 Mb.

### **SOFTWARE REQUIREMENTS:**

- ❖ **Operating system** : Windows 7 Ultimate.
- ❖ **Coding Language** : Python.
- ❖ **Front-End** : Python.
- ❖ **Designing** : Html,css,javascript.
- ❖ **Data Base** : MySQL.

## **3.4. MODULE DESIGN**

1. Save Product with Blockchain Entry:

2. Retrieve Product Data

3. Authenticate Scan

- 1) Save Product with Blockchain Entry: In this module user will enter product details and then upload product bar code image and then digital signature will be generated on uploaded barcode and then this transaction details will be store in Blockchain. Before storing transaction Blockchain will verify all old transaction and upon successful verification new transaction block will be store
- 2) Retrieve Product Data: Using this module user can search existing product details by entering product id
- 3) Authenticate Scan: Here in this module we don't have any scanner so we are uploading original or fake bar code images and then Blockchain will verify digital signature of uploaded bar code with already store bar codes and if match found then Blockchain will extract all details and display to user else authentication will be failed.



# **CHAPTER 4**

## **SYSTEM DESIGN**

## **4.SYSTEM DESIGN**

### **4.1. Introduction:**

System design is transition from a user oriented document to programmers or data base personnel. The design is a solution, how to approach to the creation of a new system. This is composed of several steps. It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. Designing goes through logical and physical stages of development, logical design reviews the present physical system, prepare input and output specification, details of implementation plan and prepare a logical design walkthrough.

The database tables are designed by analyzing functions involved in the system and format of the fields is also designed. The fields in the database tables should define their role in the system. The unnecessary fields should be avoided because it affects the storage areas of the system. Then in the input and output screen design, the design should be made user friendly. The menu should be precise and compact.

**INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

**OBJECTIVES**

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the □ Future.

- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

## **4.2.**

# **INTRODUCTION TO UML**

UML UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

## **GOALS:**

The Primary goals in the design of the UML are as follows:

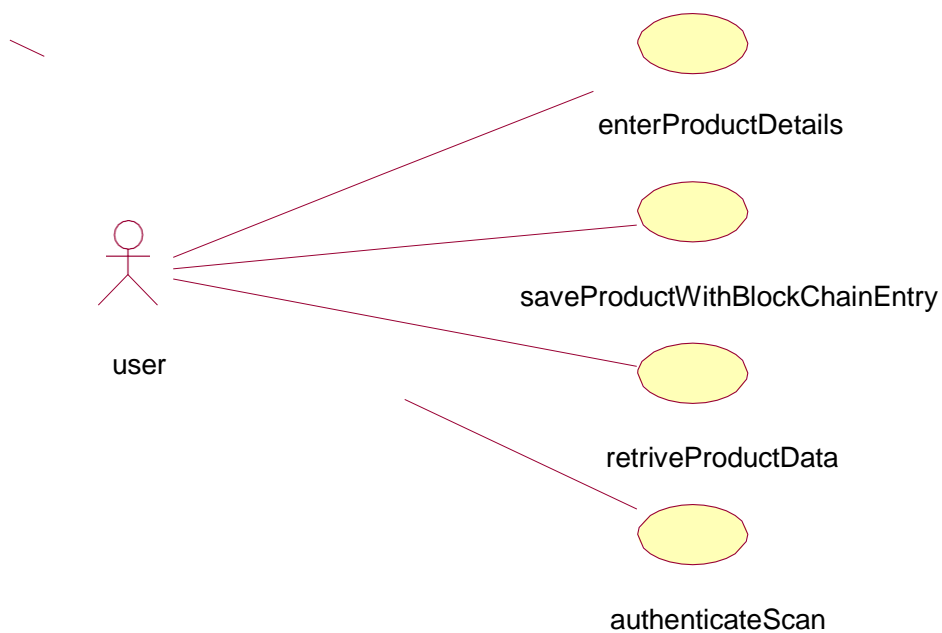
1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

## **4.3.**

# **UML DIAGRAMS**

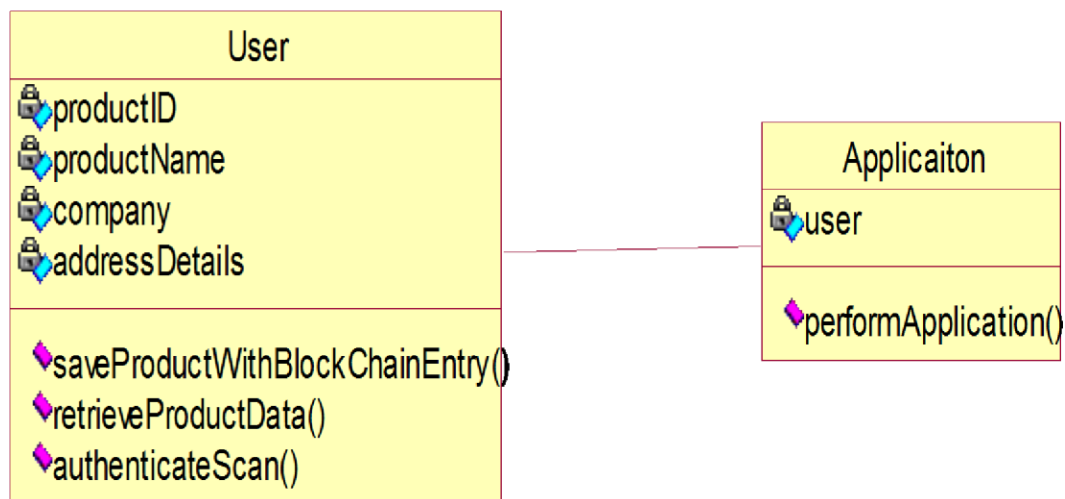
### **4.3.1. USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



### **4.3.2. CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



### **4.3.3. SEQUENCE DIAGRAM:**

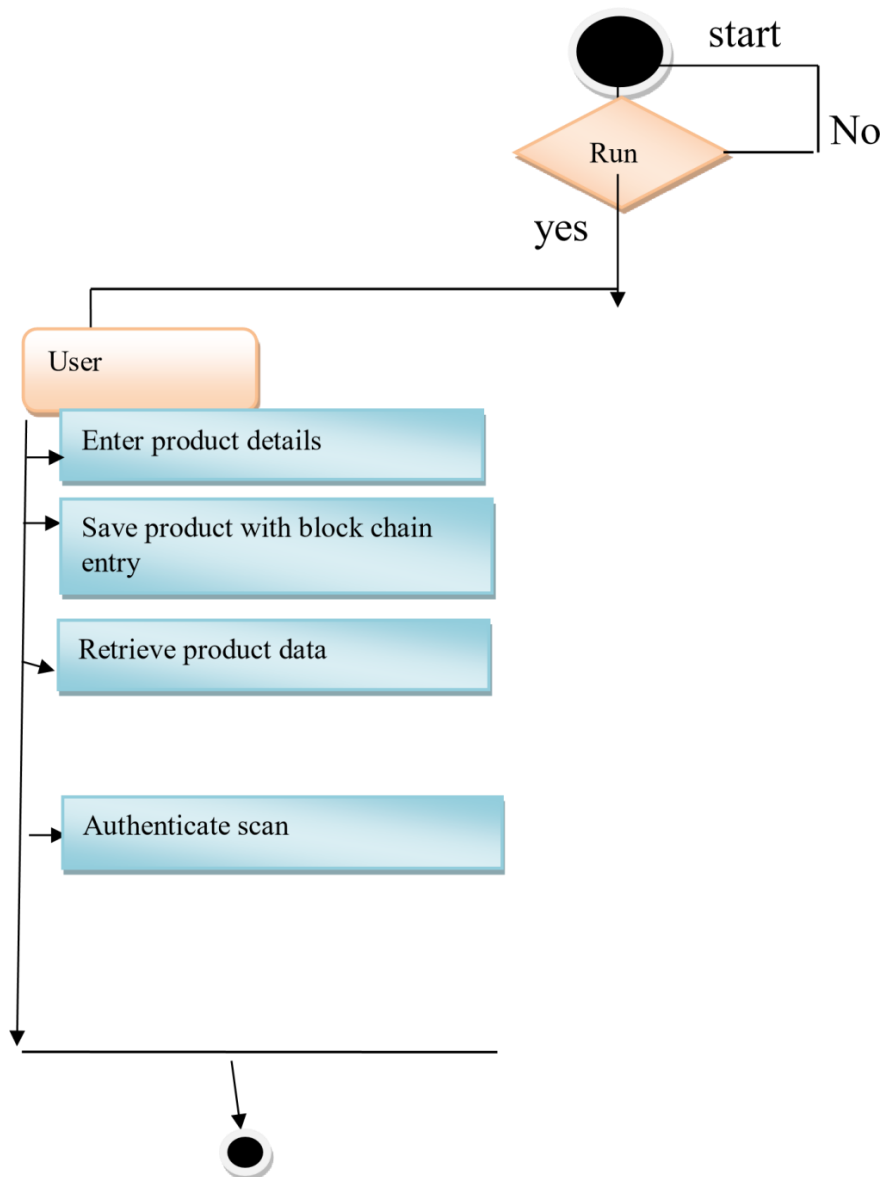
A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.





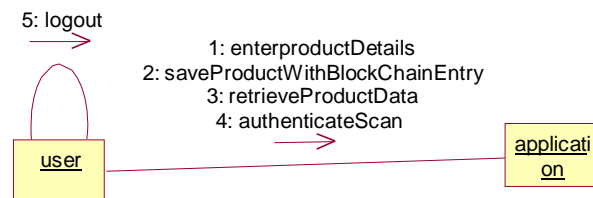
#### **4.3.4. ACTIVITY DIAGRAM:**

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent



### **4.3.5. COLLABORATION**

In UML diagrams, a collaboration is a type of structured classifier in which roles and attributes co-operate to define the internal structure of a classifier. You use a collaboration when you want to define only the roles and connections that are required to accomplish a specific goal of the collaboration.



# **CHAPTER 5**

## **SYSTEM IMPLEMENTATION**

# **5.SYSTEM IMPLEMENTATION**

## **5.1. Overview of software development tools**

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 contain 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.

#### **6. Simple and Easy**

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.

#### **7. 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.

#### **8. 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.

#### **9. 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.

#### **10. 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 systemdependent features.

## **11. 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 **clientside**. 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 Venners<sup>1</sup>, 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 Machine 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 multivariety, 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 highlevel 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>

## **5.2 SOURCE CODE**

```
Main.py
from tkinter import messagebox
from tkinter import * from
tkinter import simpledialog
import tkinter
from tkinter import filedialog
from tkinter.filedialog import askopenfilename
from Block import * from Blockchain import *
from hashlib import sha256 import os import
datetime import webbrowser
main =
Tk()
main.title("Authentication of Product & Counterfeits Elimination Using
Blockchain")
main.geometry("1300x1200")

global filename
blockchain = Blockchain() if
os.path.exists('blockchain_contract.txt'): with
open('blockchain_contract.txt', 'rb') as fileinput:
    blockchain = pickle.load(fileinput)
fileinput.close()
def addProduct():
global filename
text.delete('1.0', END)
    filename = askopenfilename(initialdir = "original_barcodes")
with open(filename,"rb") as f:
    bytes = f.read()
    f.close() pid = tf1.get() name = tf2.get() user = tf3.get()
address = tf4.get() if len(pid) > 0 and len(name) > 0 and len(user) > 0
and len(address) > 0:
    current_time = datetime.datetime.now()
digital_signature = sha256(bytes).hexdigest(); data
=
pid+"#"+name+"#"+user+"#"+address+"#"+str(current_time)+"#"+digital_signature
blockchain.add_new_transaction(data) hash = blockchain.mine()
    b = blockchain.chain[len(blockchain.chain)-1]
text.insert(END,"Blockchain Previous Hash :
"+str(b.previous_hash)+"\nBlock No : "+str(b.index)+"\nCurrent Hash :
"+str(b.hash)+"\n") text.insert(END,"Barcode Blockchain
Digital Signature :
"+str(digital_signature)+"\n\n")
    blockchain.save_object(blockchain,'blockchain_contract.txt')
tf1.delete(0, 'end') tf2.delete(0, 'end')
tf3.delete(0, 'end') tf4.delete(0, 'end') else:
text.insert(END,"Please enter all details")
def
authenticateProduct():
text.delete('1.0', END)
    filename = askopenfilename(initialdir = "original_barcodes")
with open(filename,"rb") as f:
    bytes = f.read()
    f.close()
```

```

        digital_signature = sha256(bytes).hexdigest();
flag = True        for i in
range(len(blockchain.chain)):        if i > 0:
b = blockchain.chain[i]                data =
b.transactions[0]                arr =
data.split("#")                if arr[5] ==
digital_signature:                output = ''
        text.insert(END,"Uploaded Product Barcode Authentication
Successfull\n")
        text.insert(END,"Details extracted from Blockchain after
Validation\n\n")                text.insert(END,"Product ID
:
"+arr[0]+"\\n")                text.insert(END,"Product Name
:
"+arr[1]+"\\n")                text.insert(END,"Company/User
Details :
"+arr[2]+"\\n")                text.insert(END,"Address Details
:
"+arr[3]+"\\n")                text.insert(END,"Scan Date &
Time :
"+arr[4]+"\\n")                text.insert(END,"Product Barcode
Digital Sign :
"+arr[5]+"\\n")

        output='<html><body><table border=1>'
        output+='<tr><th>Block No</th><th>Product ID</th><th>Product
Name</th><th>Company/User Details</th><th>Address Details</th><th>Scan Date &
Time</th>'

        output+='<th>Product Barcode Digital Signature</th></tr>'

output+='<tr><td>'+str(i)+'</td><td>'+arr[0]+'</td><td>'+arr[1]+'</td><td>'+a
rr[2]+'</td><td>'+arr[3]+'</td><td>'+arr[4]+'</td><td>'+arr[5]+'</td></tr>'
f = open("output.html", "w")
        f.write(output)
        f.close()
        webbrowser.open("output.html",new=1)
flag = False        break        if flag:
        text.insert(END,"Uploaded Product Barcode Authentication Failed")
def searchProduct():
text.delete('1.0', END)
pid = tf1.get()        flag =
True        if len(pid) > 0:
        for i in range(len(blockchain.chain)):
                if i > 0:
                        b = blockchain.chain[i]
data = b.transactions[0]
arr = data.split("#")                if
arr[0] == pid:
        output = ''

                text.insert(END,"Product Details extracted from
Blockchain using Product ID : "+pid+"\\n\\n")
text.insert(END,"Product ID : "+arr[0]+"\\n")
text.insert(END,"Product Name :
"+arr[1]+"\\n")                text.insert(END,"Company/User
Details :
"+arr[2]+"\\n")                text.insert(END,"Address Details
:
"+arr[3]+"\\n")                text.insert(END,"Scan Date &
Time :
"+arr[4]+"\\n")                text.insert(END,"Product Barcode
Digital Sign :

```

```

"+arr[5]+"\\n")
        output='<html><body><table border=1>'
output+='<tr><th>Block No</th><th>Product ID</th><th>Product
Name</th><th>Company/User Details</th><th>Address Details</th><th>Scan Date &
Time</th>'
        output+='<th>Product Barcode Digital Signature</th></tr>'

output+='<tr><td>'+str(i)+'</td><td>'+arr[0]+'</td><td>'+arr[1]+'</td><td>'+a
rr[2]+'</td><td>'+arr[3]+'</td><td>'+arr[4]+'</td><td>'+arr[5]+'</td></tr>'
f = open("output.html", "w")
        f.write(output)
        f.close()
        webbrowser.open("output.html",new=1)
flag = False
        break
        if flag:
            text.insert(END,"Given product id does not exists")

font = ('times', 15, 'bold')
title = Label(main, text='Authentication of Product & Counterfeits
Elimination Using Blockchain') title.config(bg='bisque',
fg='purple1') title.config(font=font)
title.config(height=3, width=120) title.place(x=0,y=5)

font1 = ('times', 13, 'bold')
l1 = Label(main, text='Product ID
:') l1.config(font=font1)
l1.place(x=50,y=100)

tf1 = Entry(main,width=20)
tf1.config(font=font1) tf1.place(x=240,y=100)
l2 = Label(main, text='Product Name
:') l2.config(font=font1)
l2.place(x=50,y=150)
tf2 =
Entry(main,width=20)
tf2.config(font=font1)
tf2.place(x=240,y=150)

l3 = Label(main, text='Company/User Details :')
l3.config(font=font1) l3.place(x=50,y=200)

tf3 = Entry(main,width=60)
tf3.config(font=font1) tf3.place(x=240,y=200)

l4 = Label(main, text='Address Details :')
l4.config(font=font1) l4.place(x=50,y=250)
tf4 =
Entry(main,width=80)
tf4.config(font=font1)
tf4.place(x=240,y=250)

saveButton = Button(main, text="Save Product with Blockchain Entry",
command=addProduct) saveButton.place(x=50,y=300)
saveButton.config(font=font1)

```

```

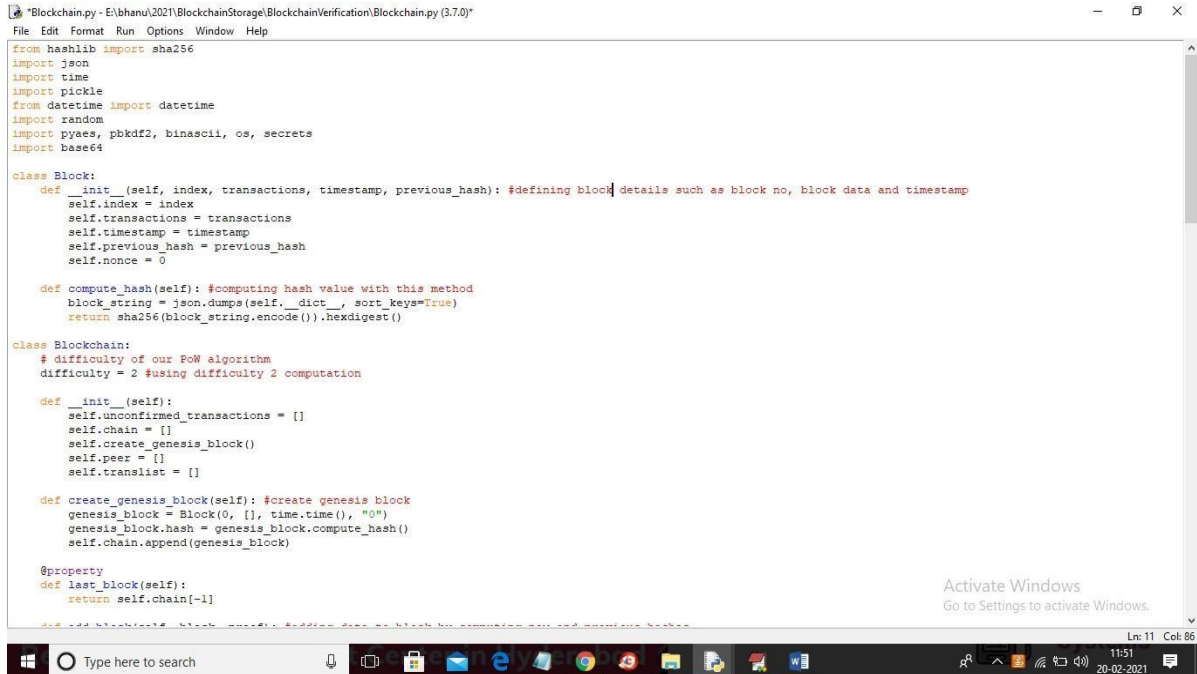
searchButton = Button(main, text="Retrieve Product Data",
command=searchProduct) searchButton.place(x=370,y=300)
searchButton.config(font=font1)

scanButton = Button(main, text="Authenticate Scan",
command=authenticateProduct) scanButton.place(x=590,y=300)
scanButton.config(font=font1)
font1 = ('times', 13, 'bold')
text=Text(main,height=15,width=120)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=350) text.config(font=font1)

main.config(bg='cornflower blue') main.mainloop()

```

## 5.3.OUTPUT SCREENSHOTS



```
*Blockchain.py - E:\bhanu\2021\BlockchainStorage\BlockchainVerification\Blockchain.py (3.7.0)*
File Edit Format Run Options Window Help
from hashlib import sha256
import json
import time
import pickle
from datetime import datetime
import random
import pyaes, pbkdf2, binascii, os, secrets
import base64

class Block:
    def __init__(self, index, transactions, timestamp, previous_hash): #defining block details such as block no, block data and timestamp
        self.index = index
        self.transactions = transactions
        self.timestamp = timestamp
        self.previous_hash = previous_hash
        self.nonce = 0

    def compute_hash(self): #computing hash value with this method
        block_string = json.dumps(self.__dict__, sort_keys=True)
        return sha256(block_string.encode()).hexdigest()

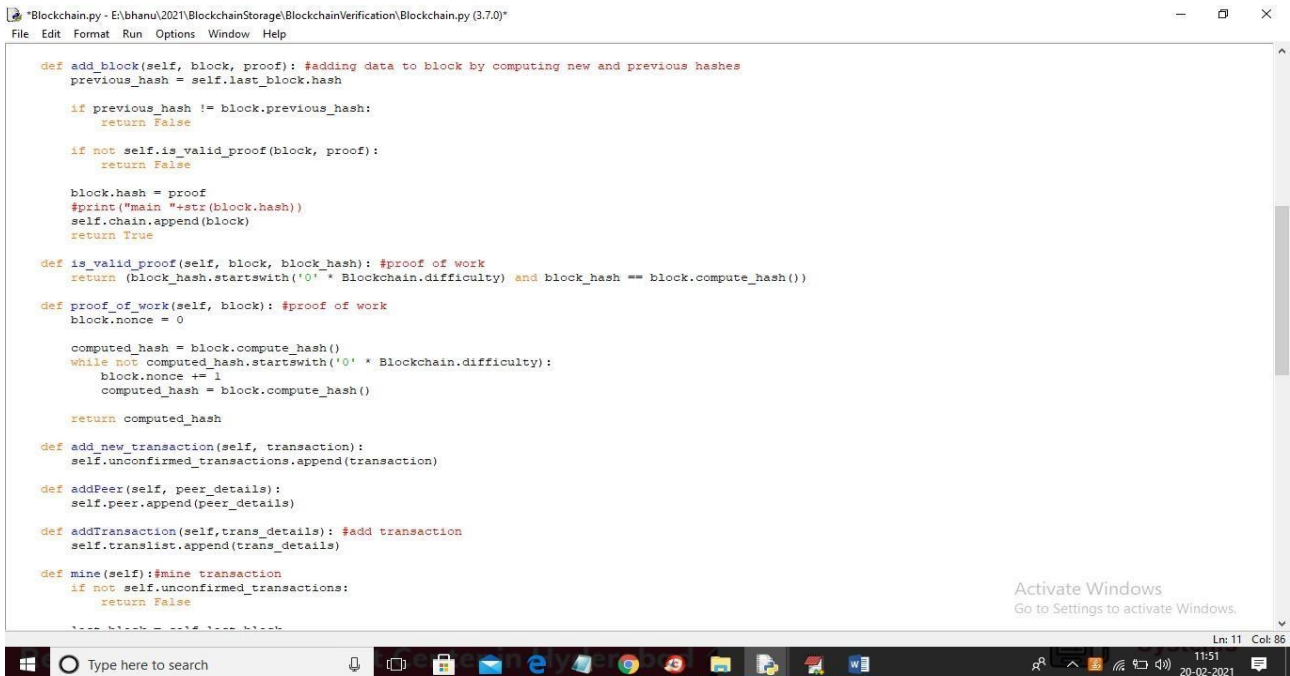
class Blockchain:
    # difficulty of our PoW algorithm
    difficulty = 2 #using difficulty 2 computation

    def __init__(self):
        self.unconfirmed_transactions = []
        self.chain = []
        self.create_genesis_block()
        self.peer = []
        self.translist = []

    def create_genesis_block(self): #create genesis block
        genesis_block = Block(0, [], time.time(), "0")
        genesis_block.hash = genesis_block.compute_hash()
        self.chain.append(genesis_block)

    @property
    def last_block(self):
        return self.chain[-1]

# Add new blocks to the blockchain. Add new data as block by computing new and previous hashes
```



```
*Blockchain.py - E:\bhanu\2021\BlockchainStorage\BlockchainVerification\Blockchain.py (3.7.0)*
File Edit Format Run Options Window Help

def add_block(self, block, proof): #adding data to block by computing new and previous hashes
    previous_hash = self.last_block.hash

    if previous_hash != block.previous_hash:
        return False

    if not self.is_valid_proof(block, proof):
        return False

    block.hash = proof
    #print("Main "+str(block.hash))
    self.chain.append(block)
    return True

def is_valid_proof(self, block, block_hash): #proof of work
    return (block_hash.startswith('0' * Blockchain.difficulty) and block_hash == block.compute_hash())

def proof_of_work(self, block): #proof of work
    block.nonce = 0

    computed_hash = block.compute_hash()
    while not computed_hash.startswith('0' * Blockchain.difficulty):
        block.nonce += 1
        computed_hash = block.compute_hash()

    return computed_hash

def add_new_transaction(self, transaction):
    self.unconfirmed_transactions.append(transaction)

def addPeer(self, peer_details):
    self.peer.append(peer_details)

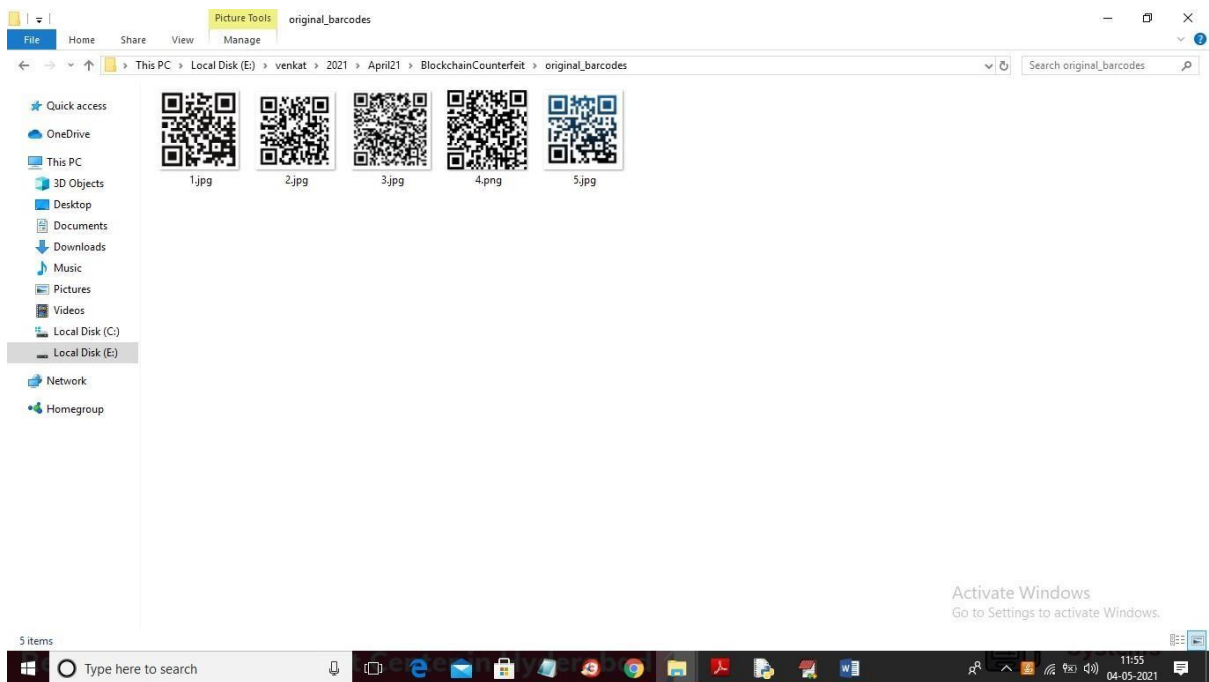
def addTransaction(self, trans_details): #add transaction
    self.translist.append(trans_details)

def mine(self): #mine transaction
    if not self.unconfirmed_transactions:
        return False

    new_block = self.last_block
```

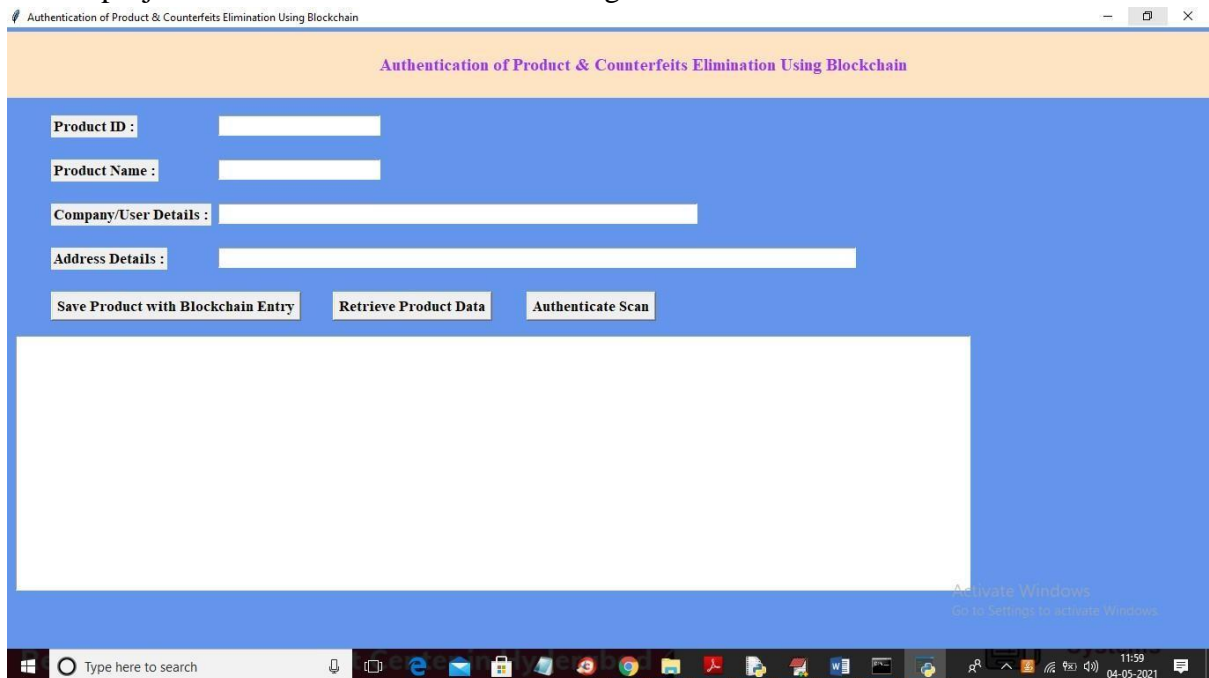


To implement this project we have taken some bar codes and this bar codes stored inside 'original\_barcode' and you can use those or you own 'barcode' to upload to Blockchain and below is the bar codes screen shots

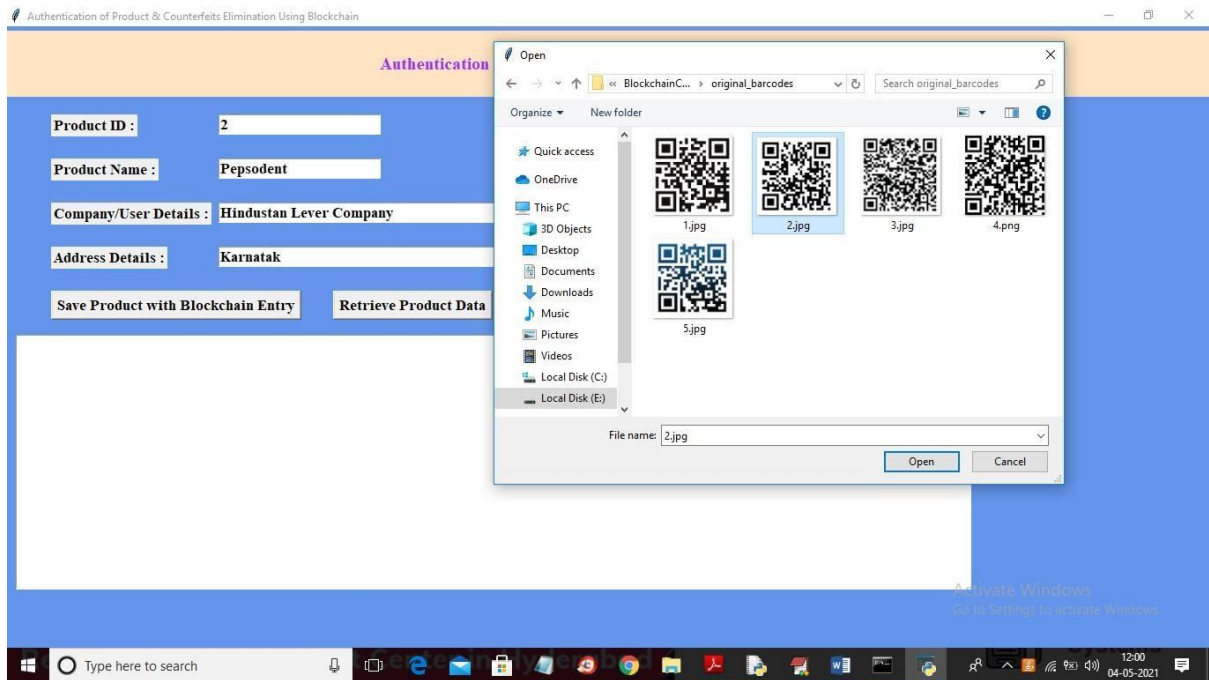


Will use above bar codes to store product details in Blockchain  
SCREEN SHOTS

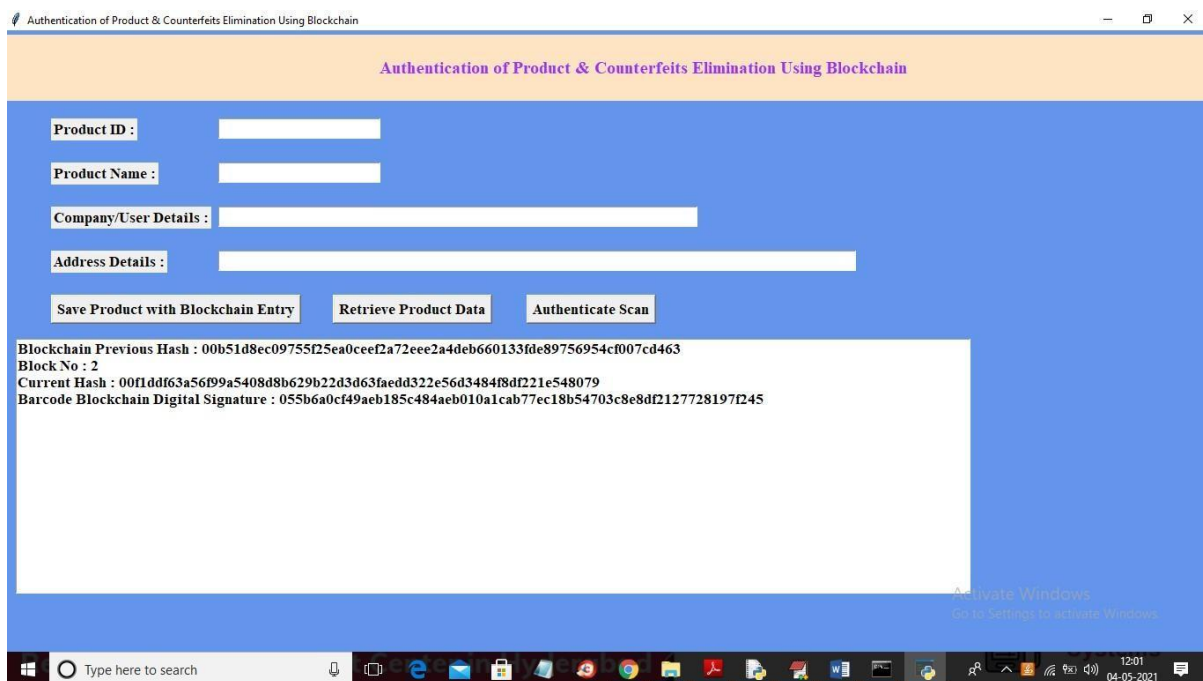
To run project double click on 'run.bat' file to get below screen



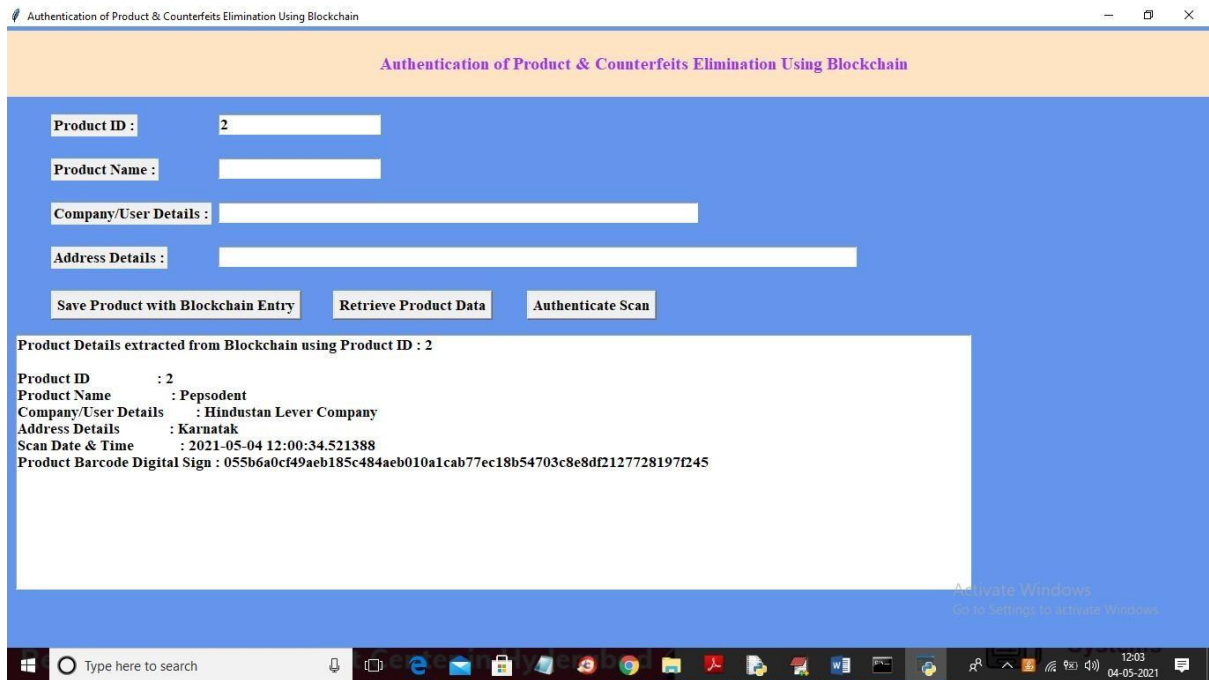
In above screen enter product details and then click on 'Save Products with Blockchain Entry' button to store product details in Block



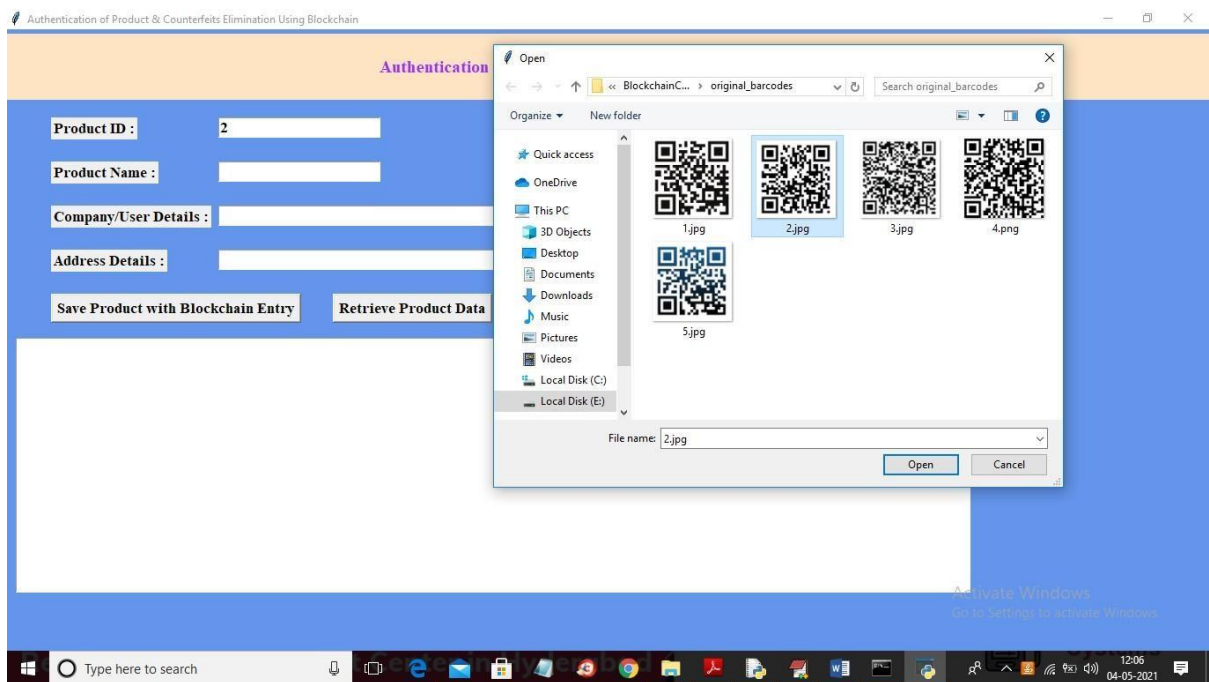
In above screen I entered product details and then selecting and uploading associated BARCODE image and then click on 'Open' button to get below result



In above screen Blockchain generated new Block with id 2 and we can see Blockchain hash code of old and new transaction with uploaded bar code digital signature and all this details will saved inside Blockchain and now to search product details click on 'Retrieve Product Data' button to get below details



In above screen I entered product id as 2 and then click on 'Retrieve Product Data' button to get above details. Now click on 'Authenticate Scan' button to upload product Barcode and then Blockchain will match this uploaded Barcode signature with available stored signatures and if match found then authentication will be successful else failed

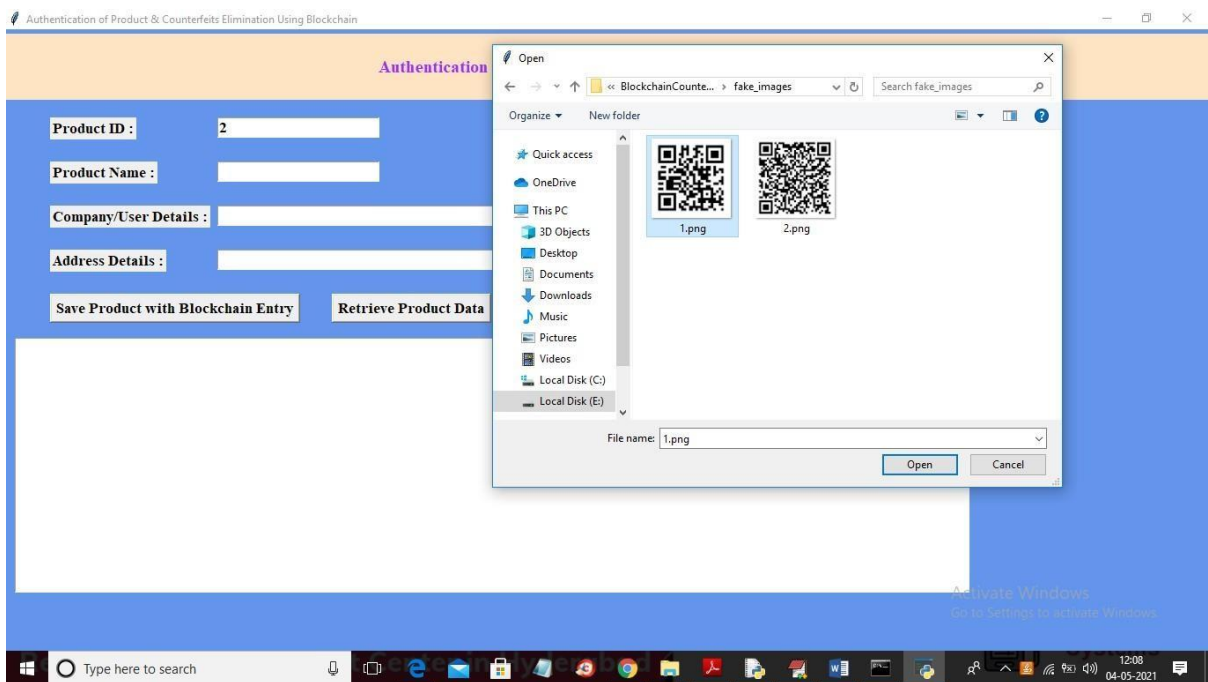


In above screen I am selecting and uploading '2.jpg' file and then click on 'Open' button to get below result

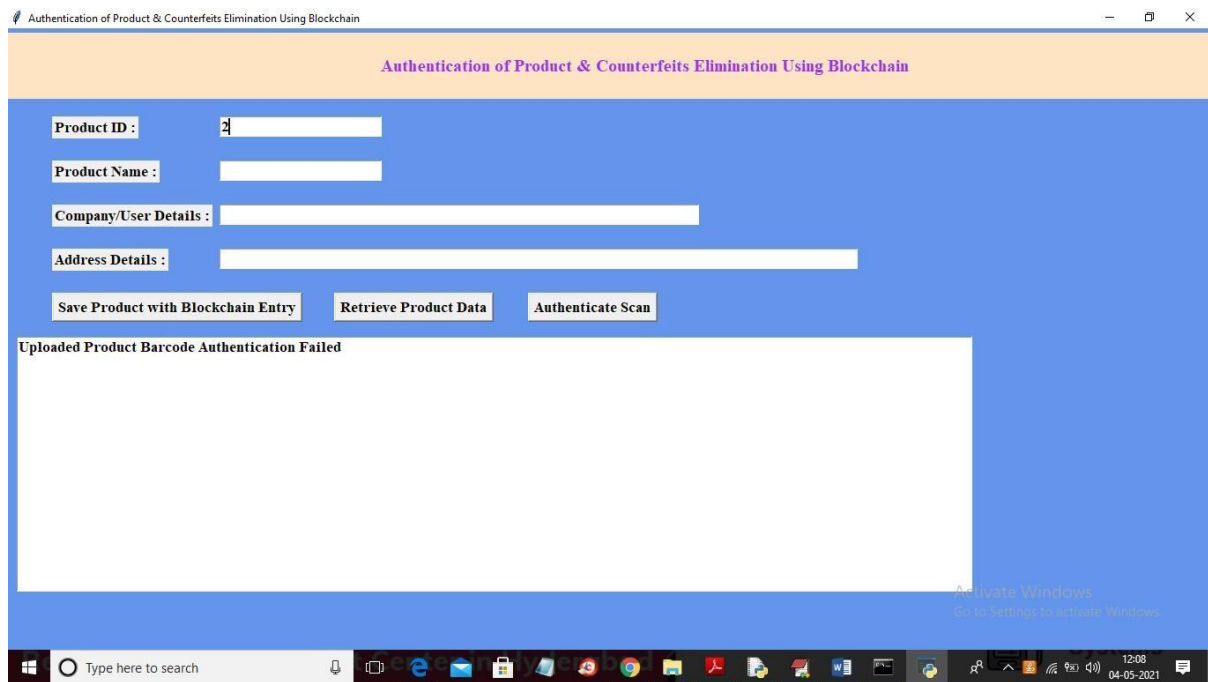
Block No	Product ID	Product Name	Company/User Details	Address Details	Scan Date & Time	Product Barcode Digital Signature
2	2	Pepsodent	Hindustan Lever Company	Karnatak	2021-05-04 12:00:34.521388	055b6a0cf49aeb185c484aeb010a1cab77ec18b54703c8e8df2127728197f245



In above screen in browser author can see all authentication details uploaded product bar code. Now check with fake barcode by uploading from 'fake bar code' folder



In above screen uploading barcode from fake folder and below is the result



In above screen in text area we can see uploaded bar code authentication failed.

# **CHAPTER 6**

## **SOFTWARE TESTING**

## **6.** **SOFTWARE TESTING**

### **6.1. Introduction to testing :**

#### **What do you mean by software testing?**

Testing involves operation of a system or application under controlled conditions and evaluating the results. The controlled conditions should include both normal and abnormal conditions. Testing should intentionally attempt to make things go wrong to determine if things happen when they shouldn't or things don't happen when they should. It is oriented to 'detection'.

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.

### **6.2. 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 satisfactory, as shown by successful 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.

# **CHAPTER 7**

# **CONCLUSION**

## **7. CONCLUSIION AND DISCUSSION**

### **7.1. CONCLUSION**

With this system, the products journey from manufacturing to customer can be recorded, and the customer is assured that the scans weren't faked. Manufacture is able to prove their product is authentic and is also able to track their product's pathway. The setup is easy to implement and requires less operation cost. Manufacturer can also adopt RFID or NFC tokens instead of QR codes to further strengthen their system.

## **7.2. FUTURE SCOPE :**

- Multiple techniques to reducing counterfeits were examined in this thesis. These improvements were considered, and their impact on minimising counterfeits was assessed, in order to be less reliant on external variables.
- Due to time constraints and the fact that several other system changes were also required, it was not possible to implement all of the suggested changes.
- The finalisation of these implementations for the proposed system, as well as the potential of running pilots, are among the next steps. The concept for reducing counterfeits in the humanitarian supply chain is currently being developed, as is the execution.