

A

Project Report on

" BLOCKCHAIN-BASED DECENTRALIZED AUTHENTICATION MODELING SCHEME IN EDGE AND IOT ENVIRONMENT"

A Project Report submitted to Acharya Nagarjuna University for the partial fulfillment of the Requirements of the Award of Degree of

MASTER OF COMPUTER APPLICATIONS



Submitted By

K.GOWTHAMI

REG NO.Y22MC58027

Project under the Guidance of

Mrs.M.USHA RANI MADAM

DEPARTMENT OF MCA

ST.ANN'S COLLEGE FOR WOMEN

GORANTLA,GUNTUR.



A

Project Report on

" BLOCKCHAIN-BASED DECENTRALIZED AUTHENTICATION MODELING SCHEME IN EDGE AND IOT ENVIRONMENT"

A Project Report submitted to **Acharya Nagarjuna University** for the partial fulfillment of the Requirements of the Award of Degree of

MASTER OF COMPUTER APPLICATIONS



Submitted By

K.GOWTHAMI

REG NO.Y22MC58027

Project under the Guidance of

Mrs.M.USHA RANI MADAM

DEPARTMENT OF MCA

ST.ANN'S COLLEGE FOR WOMEN

GORANTLA,GUNTUR.



ST.ANN'S COLLEGE FOR WOMEN

GORANTLA,GUNTUR

DEPARTMENT OF MCA



CERTIFICATE

This is to certify that it is a bonafied record of project work entitled” **BLOCKCHAIN-BASED DECENTRALIZED AUTHENTICATION MODELING SCHEME IN EDGE AND IOT ENVIRONMENT**” done by **K.GOWTHAMI ,REG.NO:Y22MC58027**, Student of MCA in “**ST.ANN'S COLLEGE FOR WOMEN**” during academic year 2022-2023 in partial fulfillment of the requirement of the award of the Degree “**MASTER OF COMPUTER APPLICATIONS**”, from”**ACHARYA NAGARJUNA UNIVERSITY**”.

INTERNAL GUIDE

PRINCIPAL

PRINCIPAL
St. Ann's College for Women
GORANTLA, GUNTUR

HEAD OF THE DEPARTMENT

Head of the Department
Department of MCA
St. Ann's College for Women
GORANTLA, GUNTUR-522034.

EXTERNALEXAMINER

DECLARATION

16th Feb'23

Sub: Conformation Letter

Sir/Madam,

This is to certify that **Ms. KOLLIKONDA GOWTHAMI (Y22MC58027)** student of MCA final year of **St. Ann's College for Women, Guntur**, seeking permission to do project work under our guidance and supervision. We are giving training specialized in **“BLOCKCHAIN - BASED DECENTRALIZED AUTHENTICATION MODELING SCHEME IN EDGE AND IOT ENVIRONMENT”**.

She needs to work on her project approximately for a period of 4 Months. She stay with the organization is purely temporary, and this letter is issues on her request.

Yours Sincerely,

For 1000Projects IT Technologies India Pvt. Ltd.

For 1000Projects IT Technologies India Pvt. Ltd.

Authorised Signatory

Director

1000Projects IT Technologies (India) Private Limited

DECLARATION

I **K.Gowthami**, Regd no:**Y22MC58027**, here by declare that the project entitled **“BLOCKCHAIN-BASED DECENTRALIZED AUTHENTICATION MODELING SCHEME IN EDGE AND IOT ENVIRONMENT”** has been carried by me at **1000 Projects IT technologies India Pvt Ltd** under the guidance of **Mr. Aravind Reddy**”.This project is Submitted to Department of MCA, **“Acharya Nagarjuna University”** for fulfillment for the award of **Master Of Computer Applications Degree**.

K.Gowthami

Regd No:Y22MC58027



ACKNOWLEDGEMENT

I take this opportunity to express my profound sincere gratitude to all those who helped me to carry out this project **Mr. Aravind Reddy Sr. Project Manager, 1000 Projects IT Technologies India Pvt Ltd, Guntur.** For allotting me this project in the Organization and enabling me to complete the same successfully, I express my profound gratitude for his valuable guidance and support.

My sincere gratitude to our principal **Rev.Dr.Sr.Fatima Rani.P Ganu,** and also to our college committee members for giving the encouragement.

My sincere thanks to **Mrs.M.Usha Rani Madam,** Head of the Department of MCA, **St. Ann's College for women,** who inspired me with her valuable suggestions and advice through out my **PG** studies at the college and also during my project work. I also sincere thanks to all other staff members.

My grateful thanks to **Mrs.M.Usha Rani Madam,** my internal guide for her valuable guidance and encouragement through out this project.

I thank one and all extend a helping hand in accomplishment of the project.

K.Gowthami

Regd No:Y22MC58027



ABSTRACT

The objective of this application is to manage details of current projects and future projects handled by an organization. The purpose is to keep track of employees and project details by maintaining a *centralized database*. Now a days in any organization it is main important for keep track of employee details and what are the projects assigned for the employees and they submitting the particular projects in a given time or not can be checked.

This projects main aim is to maintain the project details and also every person rights on which basic in the company is shown. In this projects contain Employee module, Project manager Module, Human resource module .By this modules we can see the each module type authorization on each module. Employee can do only view the employee details and Reports on Role location ,project location and skill location module only .PM can add the projects ,view the allocated projects and add new projects Hr can add the new employee and add requirements for the projects and allocation projects details can be accessed.



CONTENTS

TITLE	Page No
INTRODUCTION	1
1.1 BACKGROUND	
1.2 PROBLEM STATEMENT	
1.3 SCOPE AND OBJECTIVES	
SYSTEM ANALYSIS	2
2.1 STUDY OF THE PROBLEM	
2.2 REQUIREMENTS ANALYSIS	
2.3 PROPOSED SOLUTION	
2.4 FEASIBILITY STUDY	
IMPLEMENTATION	10
3.1 TECHNICAL IMPLEMENTATION	
3.2 SOFTWARE IMPLEMENTATION	
3.3 TESTING AND VALIDATION	
SOFTWARE REQUIREMENT SPECIFICATIONS	20
4.1 FUNCTIONAL REQUIREMENTS	
4.2 SOFTWARE REQUIREMENTS	
4.3 PERFORMANCE REQUIREMENTS	
4.4 SECURITY AND COMPLIANCE	
CONCLUSION	25
REFERENCES	
BIBLIOGRAPHY	

CONTENTS



CONTENTS

TITLE	Page.No
1. INTRODUCTION	1
1.1. EXISTING SYSTEM	
1.2. PRAPOSED SYSTEM	
2. LITERA TURE SURVEEY	3
3. SYSTEM ANALYSIS	5
3.1 STUDY OF THE SYSTEM	
3.2. INPUTAND OUT PUT REPRESENTATION	
3.3. PROCSS MODELS USED WITH JUSTIFICATION	
3.4. SYSTEM ARCHITECTURE	
4. IMPLEMENTATION	10
4.1. TECHNICAL FEASIBILITY	
4.2. OPERATIONAL FEASIBILITY	
4.3. ECONOMIC FEASIBILITY	
5. SOFTWARE REQUIREMENT SPECIFICATIONS	39
5.1. FUNCIONAL REQUIREMENTS	
5.2. SOFTWARE REQUIREMENTS	
5.3. HARDWARE REQUIREMENTS	
5.4. TECHNOLOGY DESCRIPTION	
6. METHODOLOGY	45
6.1. INTRODUCTION	
6.2. DFD DIAGRAMS	

6.3. ER DIAGRAMS	
7. SOFTWARE REQUIREMENT SPECIFICATIONS	53
8. SYSTEM TESTING	61
8.1. INTRODUCTION	
8.2. TESTING STRATEGIES	
9. SCREENS	66
9.1. INTRODUCTION	
10. CONCLUSION	82
11. BIBLIOGRAPHY	85

CHAPTER 1

INTRODUCTION



1. INTRODUCTION

CHAPTER 1

INTRODUCTION

1. INTRODUCTION

The traditional system designed by incorporating the Wireless Sensor Network (WSN) suffers from important issues such as centralization of information, single source of trust, and fails to provide a trusted data-auditing solution due to the involvement of Third Party Auditor (TPA). This article proposes a novel solution by leveraging the concept of blockchain and Distributed Data Storage Service (DDSS) system to mitigate the issues mentioned above. In our model, the sensor nodes are accountable for their activity because the blockchain permanently records the logs of sensor nodes. The proposed solution provides the decentralized authentication of sensor nodes and the information stored in the fully distributed system. Furthermore, a decentralized data-auditing scheme without involving the TPA is also suggested, which finally benefits the user in terms of bandwidth and money. The formal verification of the proposed protocol is done using the Scyther simulator tool, which indicates that the protocol is safe and can protect against the relevant attacks with 100% accuracy. We have estimated the proposed scheme's time complexity and computational efficiencies and compared them with the existing one. The computation and communication overhead of the proposed method is reduced by 22.143% and 12.5% compared with the Khalid et al. (2020) scheme. Lastly, the ethereum platform simulates our solution, confirming that less than USD 2 is required to execute the proposed solution

LITERATURE SURVEY



2. LITERATURE SURVEY

CHAPTER 2

LITERATURE SURVEY

2. LITERATURE SURVEY

Literature Survey: Blockchain-based Decentralized Authentication Modeling Scheme in Edge and IoT Environment

Introduction:

Authentication is a critical process in online applications, providing access to various services such as online banking, insurance, and healthcare records. Traditional authentication systems rely on a centralized server that stores user details, making them vulnerable to internal misuse by employees or external attacks by hackers. To overcome these challenges, researchers have proposed a blockchain-based decentralized authentication modeling scheme for edge and IoT environments. In this literature survey, we explore existing research related to blockchain, decentralized authentication, and their application in edge and IoT environments.

Blockchain Technology and Security:

Blockchain technology, originally developed for cryptocurrencies like Bitcoin, has gained significant attention due to its security features. The decentralized and immutable nature of blockchain provides inherent security benefits for data storage and verification. Research has focused on various aspects of blockchain security, including consensus algorithms, data privacy, encryption techniques, and smart contracts.

Decentralized Authentication:

Decentralized authentication refers to the process of authentication without relying on a single centralized authority. It eliminates the risks associated with central servers, such as single points of failure and potential data breaches. Different approaches have been explored for decentralized authentication, including the use of distributed ledger technologies like blockchain, distributed identity management systems, and peer-to-peer authentication protocols.

Blockchain-based Authentication:

Several studies have investigated the application of blockchain technology for authentication purposes. Blockchain can provide a tamper-proof and transparent authentication mechanism by storing authentication credentials and transaction history on the distributed ledger. Research has explored the integration of blockchain with different authentication methods, such as public-key cryptography, biometrics, and multi-factor authentication, to enhance security and user privacy.

Edge and IoT Environments:

Edge computing and IoT environments present unique challenges for authentication due to the distributed nature of devices and the need for real-time authentication. Research has focused on developing authentication mechanisms specifically designed for edge and IoT environments. This includes lightweight authentication protocols, secure communication frameworks, and device identity management techniques..

CHAPTER 3
SYSTEM ANALYSIS

CHAPTER 3

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

The Systems Development Life Cycle (SDLC), or Software Development Life Cycle in systems engineering, information systems and software engineering, is the process of creating or altering systems, and the models and methodologies that people use to develop these systems. In software engineering the SDLC concept underpins many kinds of software development methodologies.

EXISTING SYSTEM:

Blockchain technology supports decentralized application management where single application runs on different nodes and if any nodes crash or down due to attack then other nodes will fulfil services and Blockchain verify each node access with the help of hashcode and if hashcode verify then only user is allow to perform transaction of block storage. So Blockchain will maintain application in different nodes so services will not be disturbed if any node get failure.

DISADVANTAGES:

1. In this existing system we may not provide the security for the users data
2. Users data may be disclose to other

PROPOSED SYSTEM:

In propose paper author designed secure registration and authentication strategy, blockchain-based decentralized authentication protocol, and developed the blockchain consensus, smart contract, and implemented a whole blockchain-based authentication platform for the feasibility, security and performance evaluation. The analysis and evaluation show that the proposed BlockAuth scheme provides a more secure, reliable and strong fault tolerance decentralized novel authentication with high-level security driven configuration management.

ADVANTAGES:

1. It's more secure, reliable
2. It provide high-level security

FUNCTIONAL REQUIREMENTS:

Here are some functional requirements for a blockchain-based decentralized authentication modelling scheme in edge and IoT environments:

User Registration: Allow users to register on the blockchain platform and create their unique digital identities.

Identity Management: Enable users to manage their digital identities and control access to their personal information.

Authentication Process: Develop a decentralized authentication process that verifies the user's identity using blockchain technology and smart contracts.

Edge and IoT Integration: Integrate the blockchain platform with edge and IoT devices to enable secure authentication and communication between devices.

Security and Privacy: Ensure that the authentication process is secure and meets privacy requirements by using encryption, digital signatures, and other security measures.

Consensus Mechanism: Implement a consensus mechanism to ensure that the transactions on the blockchain platform are valid and secure.

Access Control: Enable users to control access to their personal data by setting permissions and access levels.

Transaction Monitoring: Monitor transactions on the blockchain platform to identify any suspicious activity or unauthorized access.

Auditing and Reporting: Maintain audit logs and generate reports to provide transparency and accountability for all transactions on the blockchain platform.

Scalability: Ensure that the platform is scalable and can handle a large number of users and devices.

Interoperability: Ensure that the platform is interoperable with other blockchain platforms and can integrate with existing authentication systems.

Continuous Improvement: Continuously improve the authentication process and platform functionality to ensure its effectiveness and security.

These are just some of the functional requirements for a blockchain-based decentralized authentication modelling scheme in edge and IoT environments.

MODULES

BlockAuth User Registration: using this module user can enter signup details and then DJNAGO IOT server will accept all this details and then encrypt data using ECC algorithm and then generate hash code and then call Blockchain setAuthentication function to store user signup details. For each user sign data Blockchain will generate certificate and saved it as block

BlockAuth User Verification: using this module user can enter username and password and then application will generate public and private keys on user details and then decrypt the data and if user details are valid then valid keys will be generated and then data verification will be completed and this verified data will be check with Blockchain data and if matched found then user authentication will be successful.

SOFTWARE REQUIREMENTS:

HARDWARE REQUIREMENTS:

System	:	Pentium i3/i5.
Hard Disk	:	500 GB.
Monitor	:	15'' LED
Input Devices	:	Keyboard, Mouse
Ram	:	4 GB

SOFTWARE REQUIREMENTS:

Operating system	:	Windows 8/10.
Coding Language	:	Python

CHAPTER 4

IMPLEMENTATION

CHAPTER 4

IMPLEMENTATION

4. IMPLEMENTATION

What is Python:-

Below are some facts about Python.

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

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

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

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

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

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

Advantages of Python :-

Let's see how Python dominates over other languages.

1. Extensive Libraries

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 system-dependent 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 **client-side**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbannelle**.

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 Venner¹, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it." Later on in the same interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

What is Machine Learning :-

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

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models *tunable parameters* that can be adapted to observed data; in this way the program can be

considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

Categories Of Machine Learning :-

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

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

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

Need for Machine Learning

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

the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

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

Challenges in Machines Learning :-

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

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

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

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

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

Issue of overfitting & underfitting – If the model is overfitting or underfitting, it

cannot be represented well for the problem.

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

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

Applications of Machines Learning :-

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

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

How to Start Learning Machine Learning?

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

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

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

How to start learning ML?

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

Step 1 – Understand the Prerequisites

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

(a) Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want

to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

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

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

(c) Learn Python

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

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

Step 2 – Learn Various ML Concepts

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

(a) Terminologies of Machine Learning

Model – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.

Feature – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.

• **Target (Label)** – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.

• **Training** – The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.

• **Prediction** – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

• **Supervised Learning** – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.

• **Unsupervised Learning** – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.

• **Semi-supervised Learning** – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.

• **Reinforcement Learning** – This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

Advantages of Machine learning :-

1. Easily identifies trends and patterns -

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

2. No human intervention needed (automation)

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

3. Continuous Improvement

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

4. Handling multi-dimensional and multi-variety data

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

5. Wide Applications

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

Disadvantages of Machine Learning :-

1. Data Acquisition

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

2. Time and Resources

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

3. Interpretation of Results

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

4. High error-susceptibility

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

Python Development Steps :-

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system.

Python version 1.0 was released in January 1994. The major new features included in

this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it." Some changes in Python 7.3:

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

Purpose :-

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

Python

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

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative,

functional and procedural, and has a large and comprehensive standard library.

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

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

Modules Used in Project :-

Tensorflow

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

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

Numpy

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

It is the fundamental package for scientific computing with Python. It contains various

features including these important ones:

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

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

Pandas

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

Matplotlib

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

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

Scikit – learn

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

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

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

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

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

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

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace. The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac :

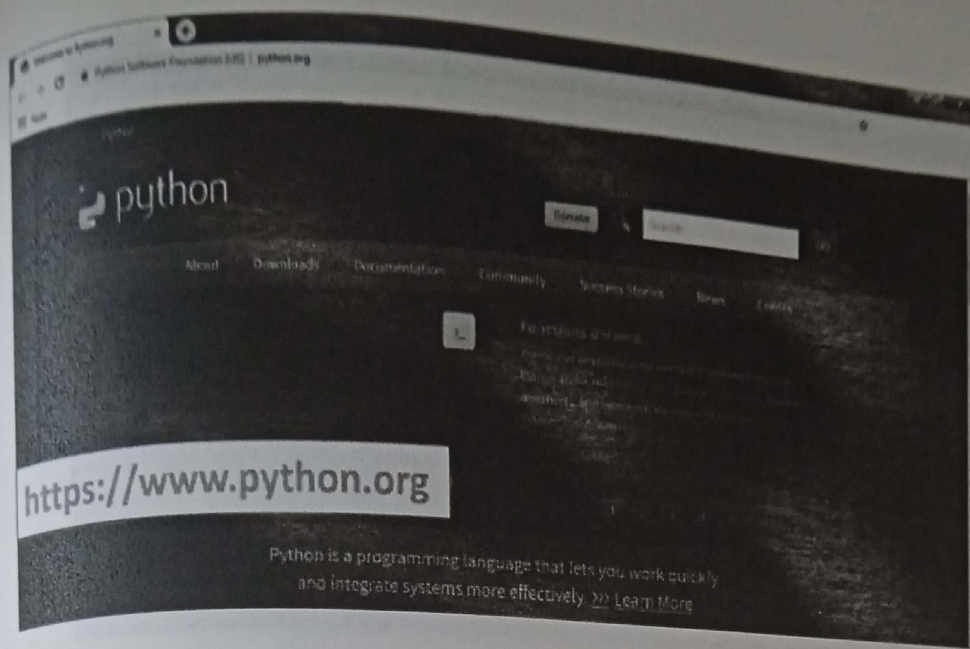
There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other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

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

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

Download the Correct version into the system

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



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

Step 2: Click on the Download Tab.

Download the latest version for Windows

Download Python 3.7.4 ← **CLICK HERE**

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)








Want to help test development versions of Python? [Pre-releases](#), [Docker images](#)

Looking for Python 2.7? See below for specific releases

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

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	 Download	Release Notes
Python 3.6.9	July 2, 2019	 Download	Release Notes
Python 3.7.3	March 25, 2019	 Download	Release Notes
Python 3.4.10	March 18, 2019	 Download	Release Notes
Python 3.5.7	March 18, 2019	 Download	Release Notes
Python 3.7.16	March 4, 2019	 Download	Release Notes
Python 3.7.2	Dec 24, 2018	 Download	Release Notes

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

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

Version	Operating System	Description	MD5 Sum	File Size	GPG
Clipped source tarball	Source release		58111671e5b2b4ae77b9abd1bf0f9be	23017563	SIG
XZ compressed source tarball	Source release		033e4ae96097051c2eca45ee3904903	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.4 and later	6428b4fa7583da91e442c8a1cee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5d9605c38217445773b5e4a936b241f	28082845	SIG
Windows help file	Windows		053998573a2c66b2ac58cade6b47r02	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64 (EM64T)/x64	9b00c8cfd9ec0bfabe81154a40727a2	7504391	SIG
Windows x86-64 executable installer	Windows	for AMD64 (EM64T)/x64	a702b4b0ad76deb0c3041a583e563e00	26685348	SIG
Windows x86-64 web-based installer	Windows	for AMD64 (EM64T)/x64	28cb116081bdf3aee51a3d6351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b8198418791da9413357413903	6741626	SIG
Windows x86 executable installer	Windows		33ca902942a54446a3d0451476394789	25683848	SIG
Windows x86 web-based installer	Windows		1b679cfa5d117d92c30883ea371d57c	1324608	SIG

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

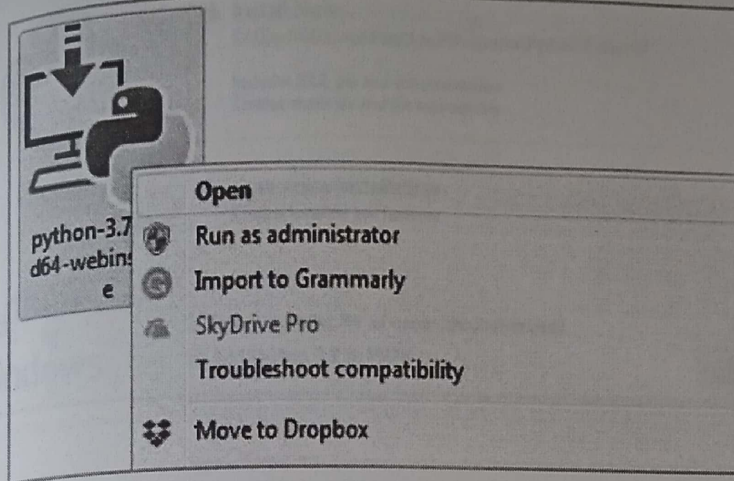
• To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

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

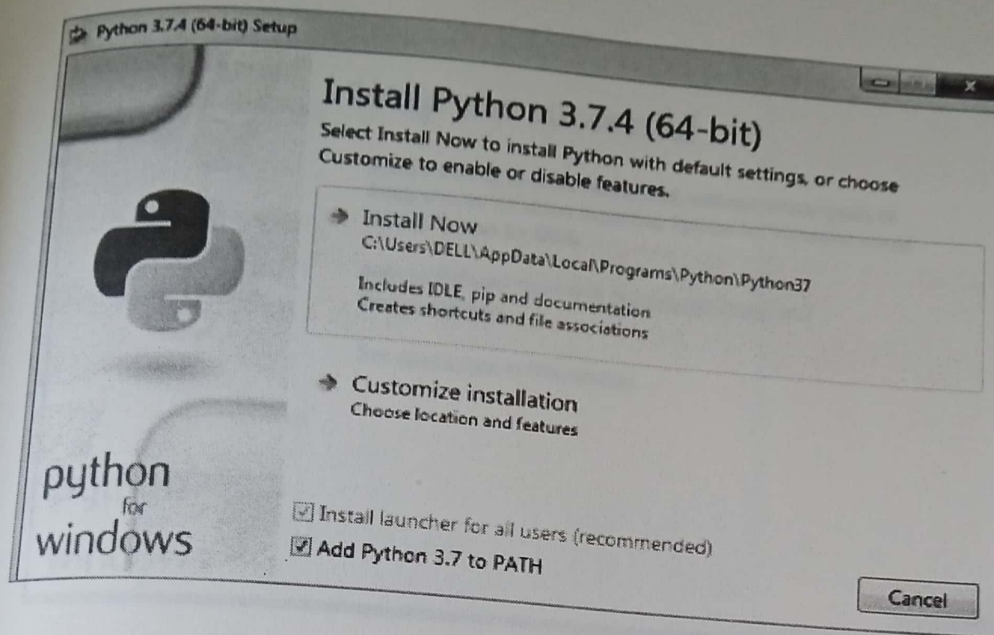
Note: To know the changes or updates that are made in the version you can click on

the Release Note Option.
Installation of Python

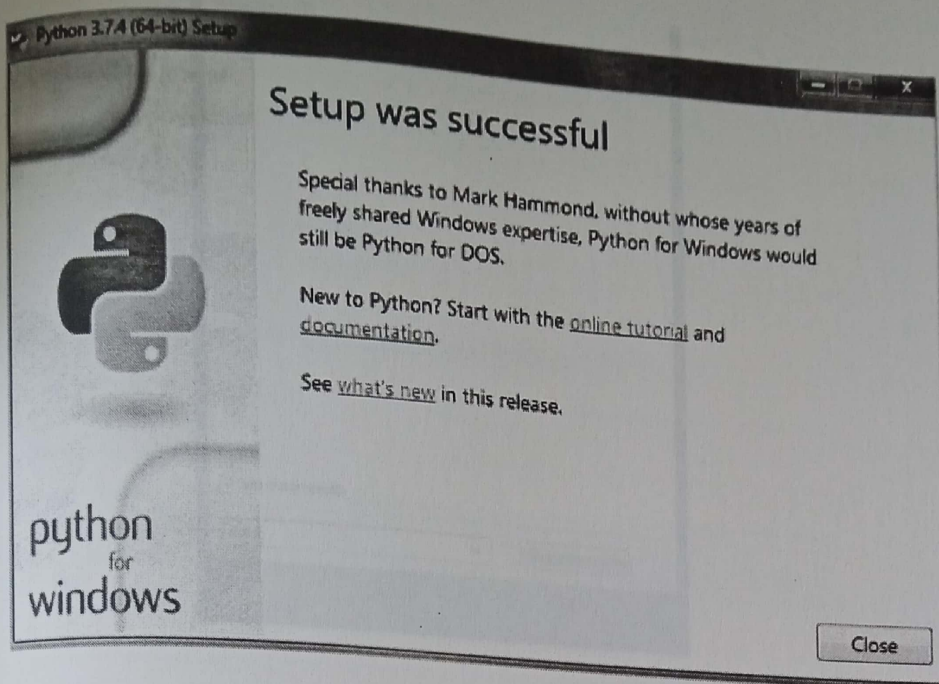
Step 1: Go to Download and Open the downloaded python version to carry out the installation process.



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



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



Step 3: Click the Command prompt option.

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

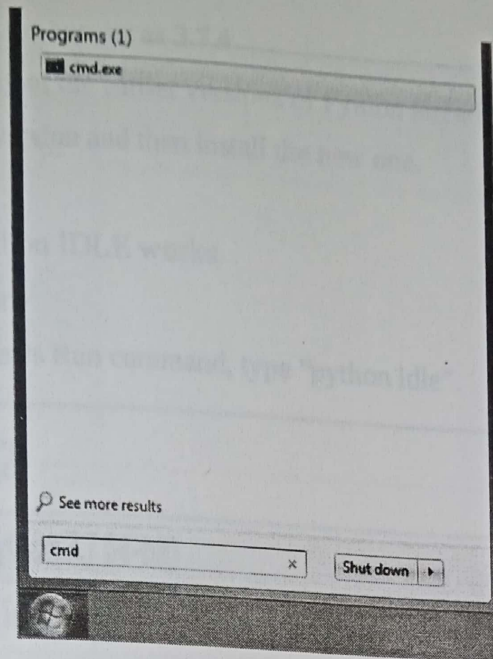
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

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

Verify the Python Installation

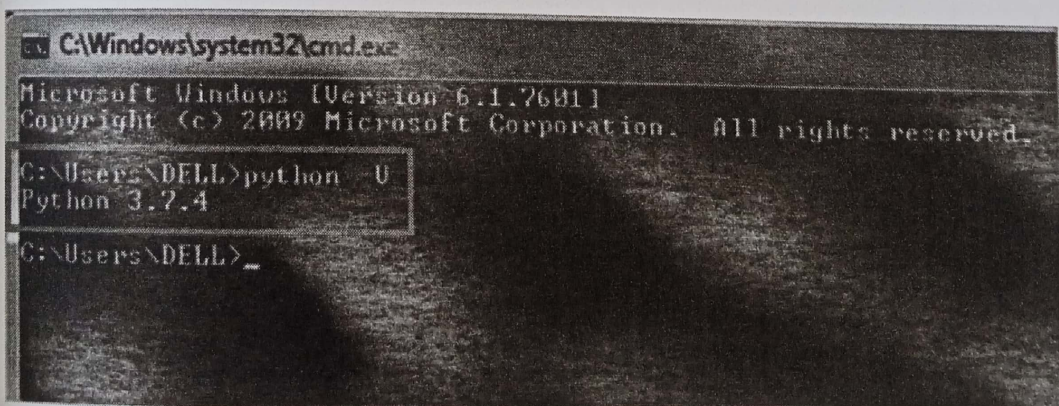
Step 1: Click on Start

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



Step 3: Open the Command prompt option.

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



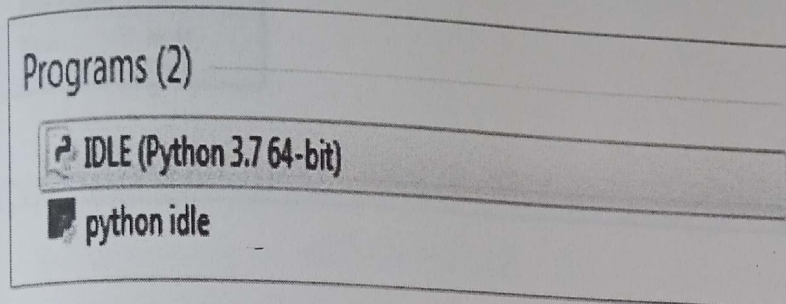
Step 5: You will get the answer as 3.7.4

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

Check how the Python IDLE works

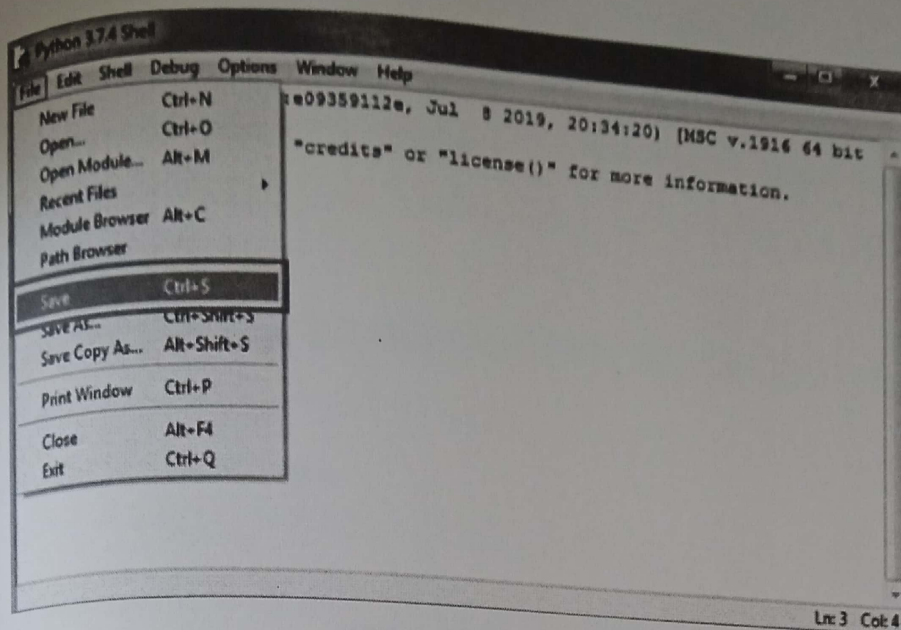
Step 1: Click on Start

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



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

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



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

Step 6: Now for e.g. enter print

SOFTWARE REQUIREMENT SPECIFICATION

Software Specification:

Software specification provides a high-level overview to the web server software requirements that will be used to be installed on the server which provides optimal functioning for the application. These software and hardware requirements provide a compatible support to the organization system in developing applications.

CHAPTER 5

SOFTWARE REQUIREMENT SPECIFICATION

HARDWARE REQUIREMENTS:

The hardware requirement specifies each hardware of the software elements and the hardware details of the system. These hardware requirements include configuration details.

- Processor : Pentium IV 2.4 GHz
- Hard Disk : 100 GB
- Monitor : 15 VGA Color
- Mouse : Logitech
- RAM : 1 GB

5. SOFTWARE REQUIREMENT SPECIFICATION

5.1 Requirements Specification:

Requirement Specification provides a high secure storage to the web server efficiently. Software requirements deal with software and hardware resources that need to be installed on a server which provides optimal functioning for the application. These software and hardware requirements need to be installed before the packages are installed. These are the most common set of requirements defined by any operation system. These software and hardware requirements provide a compatible support to the operation system in developing an application.

5.1.1 HARDWARE REQUIREMENTS:

The hardware requirement specifies each interface of the software elements and the hardware elements of the system. These hardware requirements include configuration characteristics.

- System : Pentium IV 2.4 GHz.
- Hard Disk : 100 GB.
- Monitor : 15 VGA Color.
- Mouse : Logitech.
- RAM : 1 GB.

5.1.2 SOFTWARE REQUIREMENTS:

The software requirements specify the use of all required software products like data management system. The required software product specifies the numbers and version. Each interface specifies the purpose of the interfacing software as related to this software product.

- Operating system : Windows XP/7/10
- Coding Language: Python 3.7

first install node-v 12 software available in this folder

then install below packages

```
pip install web3==4.7.2
```

```
pip install Django==2.1.7
```

After above steps and now follow screen shots to run code

no need to do any changes in hello-eth folder as i already done it

to start cloud server go inside 'CloudServer' folder and then double click on run.bat file

```
C:\Users\del\Desktop\BlockAuthProject\hello-eth\node_modules\.bin
```

```
truffle develop
```

```
truffle migrate
```

5.2 FUNCTIONAL REQUIREMENTS:

The functional requirement refers to the system needs in an exceedingly computer code engineering method.

The key goal of determinant “functional requirements” in an exceedingly product style and implementation is to capture the desired behavior of a software package in terms of practicality and also the technology implementation of the business processes.

5.3 NON FUNCTIONAL REQUIREMENTS

All the other requirements which do not form a part of the above specification are categorized as Non-Functional needs. A system perhaps needed to gift the user with a show of the quantity of records during info. If the quantity must be updated in real time, the system architects should make sure that the system is capable of change the displayed record count at intervals associate tolerably short interval of the quantity of records dynamic. Comfortable network information measure may additionally be a non-functional requirement of a system.

The following are the features:

- Accessibility
- Availability
- Backup
- Certification
- Compliance
- Configuration Management
- Documentation
- Disaster Recovery
- Efficiency(resource consumption for given load)
- Interoperability

5.4 PERFORMANCE REQUIREMENTS

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the

system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

- The system should be able to interface with the existing system
- The system should be accurate
- The system should be better than the existing system

The existing system is completely dependent on the user to perform all the duties.

5.5 Feasibility Study:

Preliminary investigation examines project feasibility; the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility

Economical Feasibility

5.5.1 Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?

Are there technical guarantees of accuracy, reliability, ease of access and data security?

5.5.2 Operational Feasibility

User-friendly

Customer will use the forms for their various transactions i.e. for adding new routes, viewing the routes details. Also the Customer wants the reports to view the various transactions based on the constraints. These forms and reports are generated as user-friendly to the Client.

Reliability

The package will pick-up current transactions on line. Regarding the old transactions, User will enter them in to the system.

Security

The web server and database server should be protected from hacking, virus etc

Portability

The application will be developed using standard open source software (Except Oracle) like Java, tomcat web server, Internet Explorer Browser etc these software will work both on Windows and Linux o/s. Hence portability problems will not arise.

Availability

This software will be available always.

Maintainability

The system uses the 2-tier architecture. The 1st tier is the GUI, which is said to be front-end and the 2nd tier is the database, which uses My-Sql, which is the back-end.

The front-end can be run on different systems (clients). The database will be running at the server. Users access these forms by using the user-ids and the passwords.

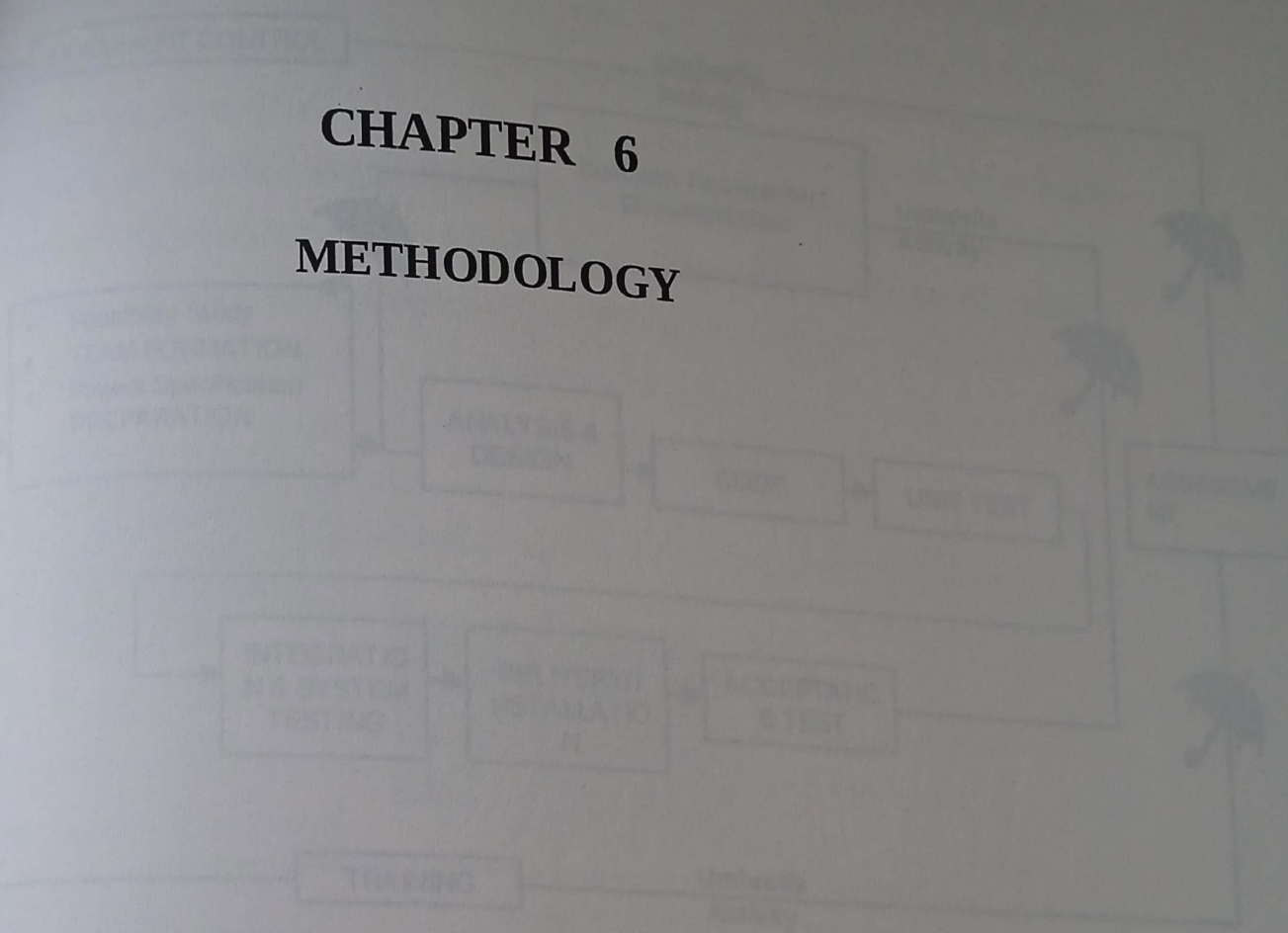
5.5.3 Economic Feasibility

The computerized system takes care of the present existing system's data flow and procedures completely and should generate all the reports of the manual system besides a host of other management reports.

It should be built as a web based application with separate web server and database server. This is required as the activities are spread throughout the organization customer wants a centralized database. Further some of the linked transactions take place in different locations.

CHAPTER 6

METHODOLOGY



6. Methodology

SDLC (Software Development Life Cycle) – Umbrella Model

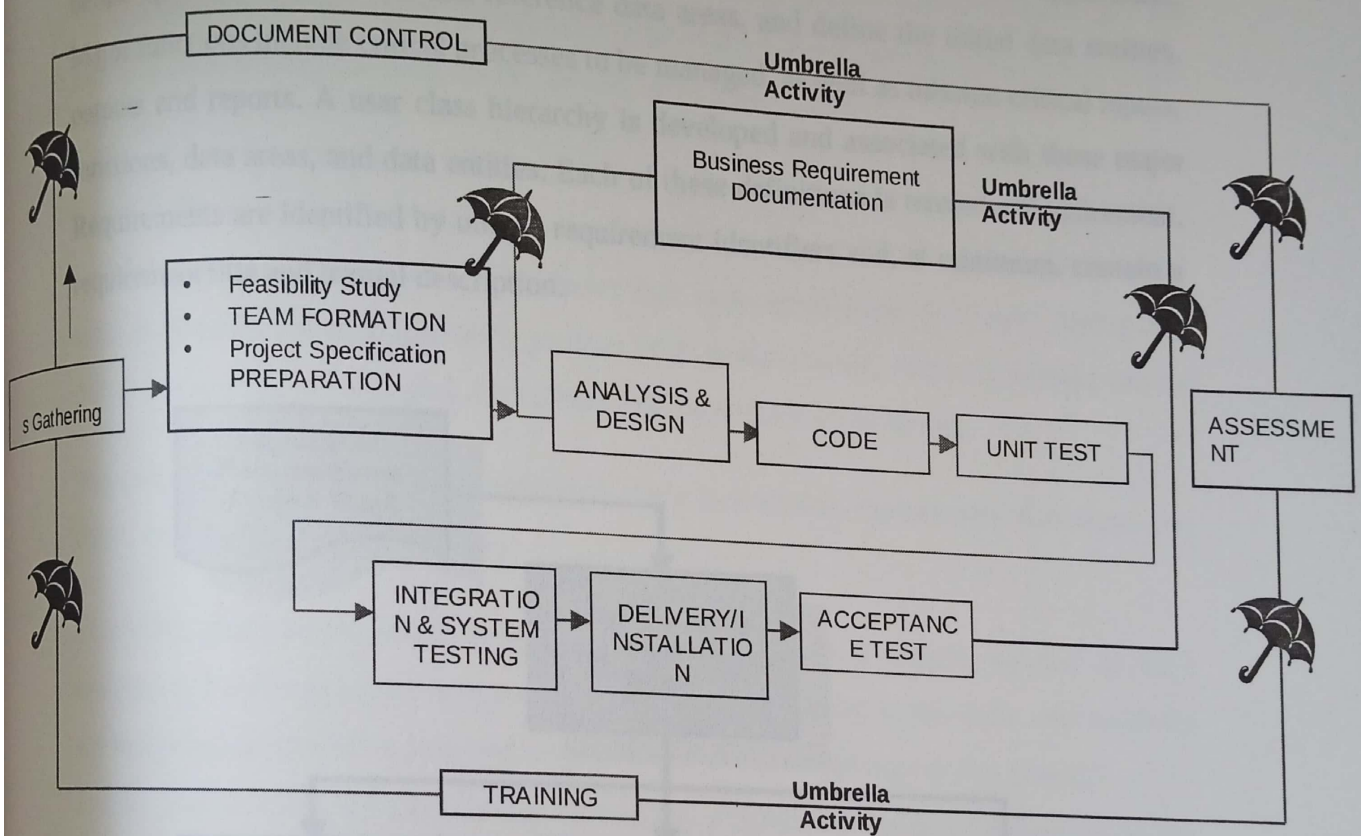


Fig no. 6.1 Umbrella model

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

Requirements Gathering Stage

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.

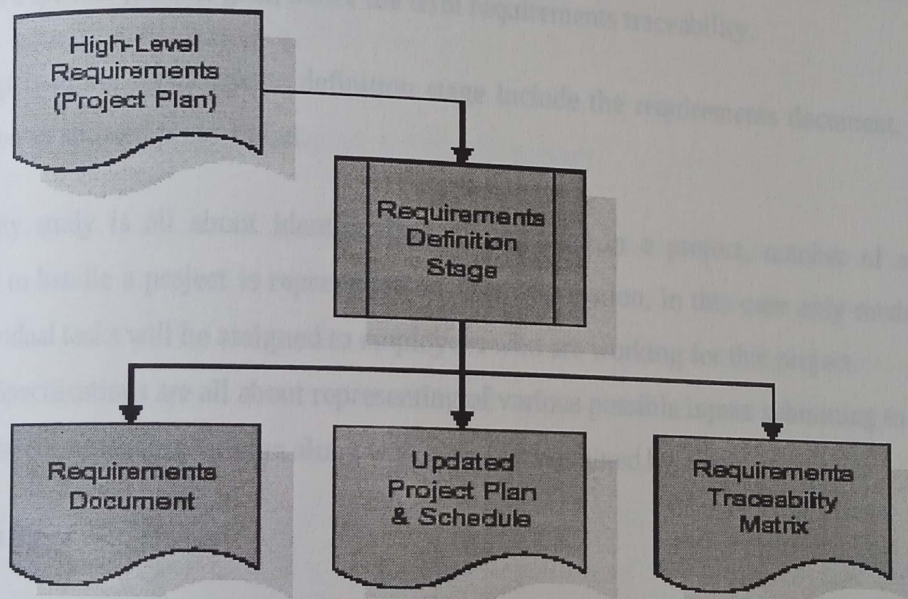


Fig no. 6.2 Requirements Gathering stage

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are not included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

Feasibility study is all about identification of problems in a project, number of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.

Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

Analysis Stage

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.

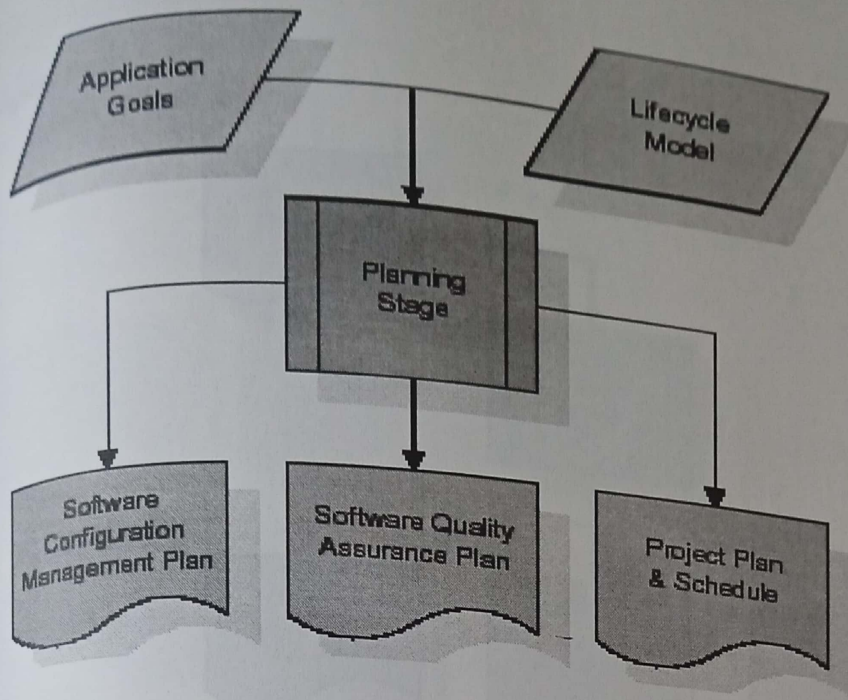


Fig no. 6.3 Analysis stage

The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

Designing Stage

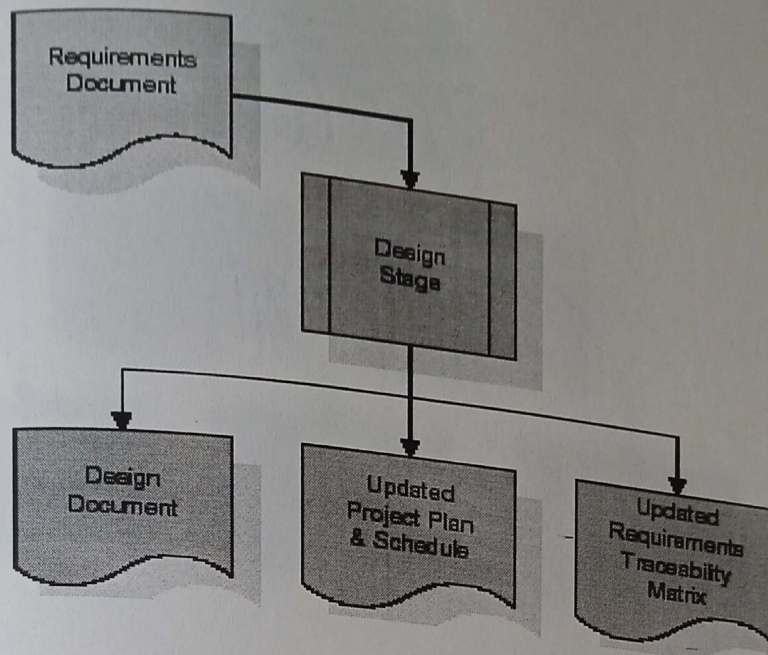


Fig no. 6.4 Designing stage

When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

Development (Coding) Stage

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.

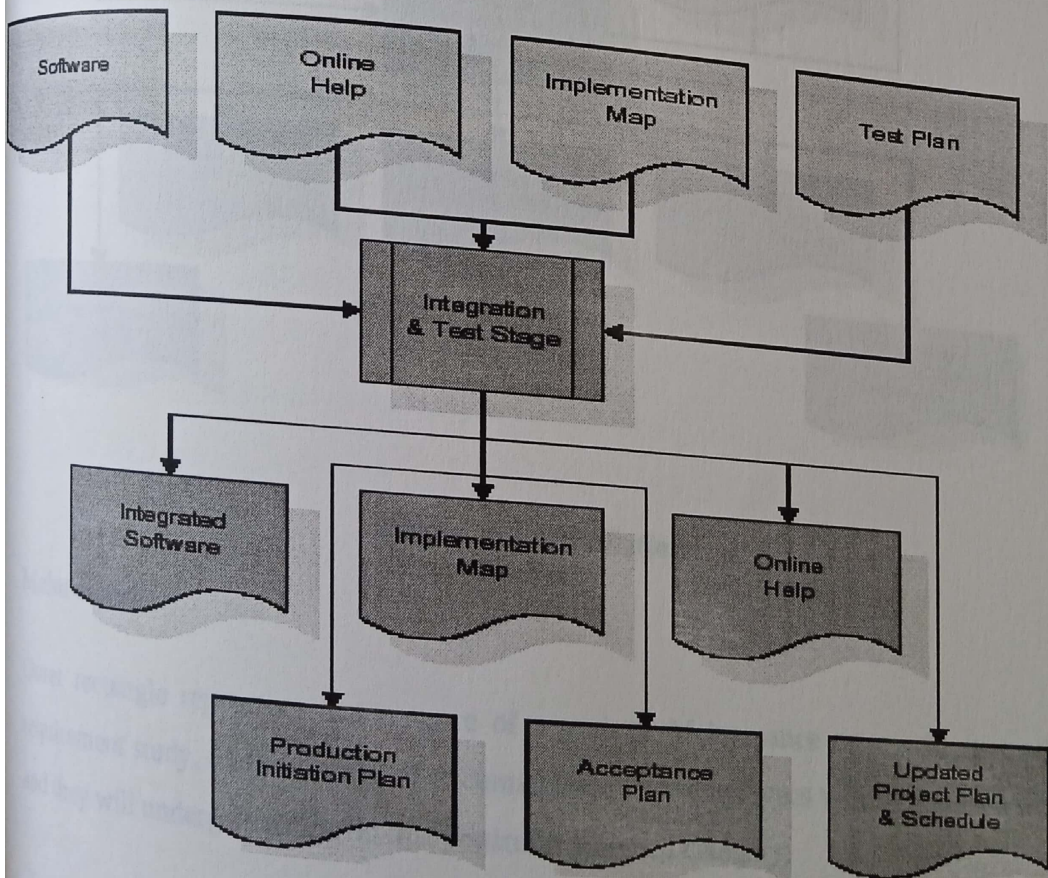
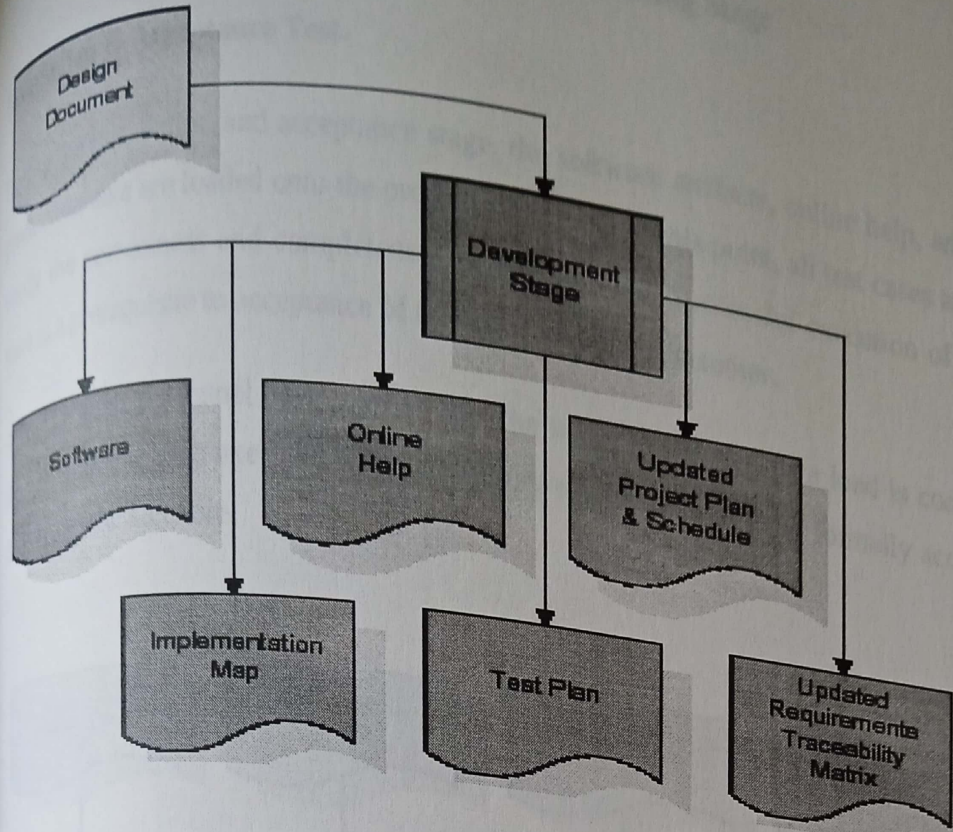


Fig no. 6.6 Integration and Testing Stage

Installation & Acceptance Test

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.

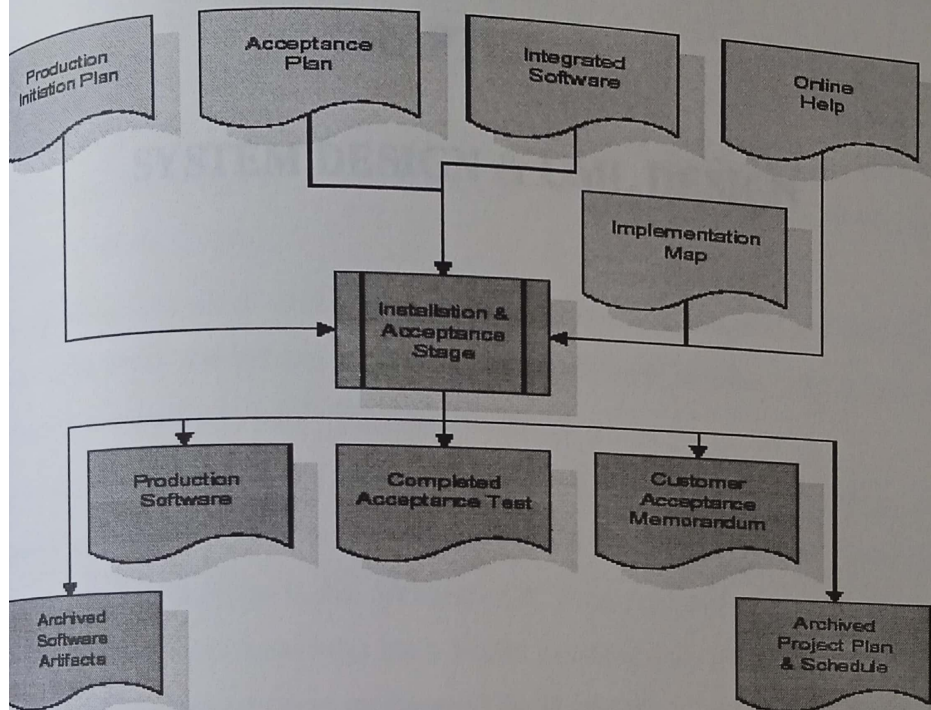


Fig no. 6.7 Installation

Maintenance

After the maintenance of a project, the Maintenance team will start with a requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category.

SYSTEM ARCHITECTURE

The design phase is the stage in which the system is designed to meet the requirements. This phase involves the selection of the system architecture, the design of the system components, and the design of the system interfaces. The design phase is the most critical phase in the system development process, as it determines the overall structure and functionality of the system. The design phase is also the most expensive phase, as it involves the development of the system architecture and the design of the system components.

CHAPTER 7

SYSTEM DESIGN & UML DESIGN

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy its requirements. It is a multi-disciplinary activity that involves the collaboration of system analysts, software engineers, and hardware engineers. The system design process is iterative and incremental, as it allows for the refinement of the system design as more information becomes available. The system design process is also a collaborative process, as it involves the input and feedback of all stakeholders involved in the system development process.

7. System Design

7.1 SYSTEM ARCHITECTURE

The purpose of the design phase is to arrange an answer of the matter such as by the necessity document. This part is that the opening moves in moving the matter domain to the answer domain. The design phase satisfies the requirements of the system. The design of a system is probably the foremost crucial issue warm heartedness the standard of the software package. It's a serious impact on the later part, notably testing and maintenance.

The output of this part is that the style of the document. This document is analogous to a blueprint of answer and is employed later throughout implementation, testing and maintenance. The design activity is commonly divided into 2 separate phases System Design and Detailed Design.

System Design conjointly referred to as top-ranking style aims to spot the modules that ought to be within the system, the specifications of those modules, and the way them move with one another to supply the specified results.

At the top of the system style all the main knowledge structures, file formats, output formats, and also the major modules within the system and their specifications square measure set. System design is that the method or art of process the design, components, modules, interfaces, and knowledge for a system to satisfy such as needs. Users will read it because the application of systems theory to development.

Detailed Design, the inner logic of every of the modules laid out in system design is determined. Throughout this part, the small print of the info of a module square measure sometimes laid out in a high-level style description language that is freelance of the target language within which the software package can eventually be enforced.

In system design the main target is on distinguishing the modules, whereas throughout careful style the main target is on planning the logic for every of the modules.

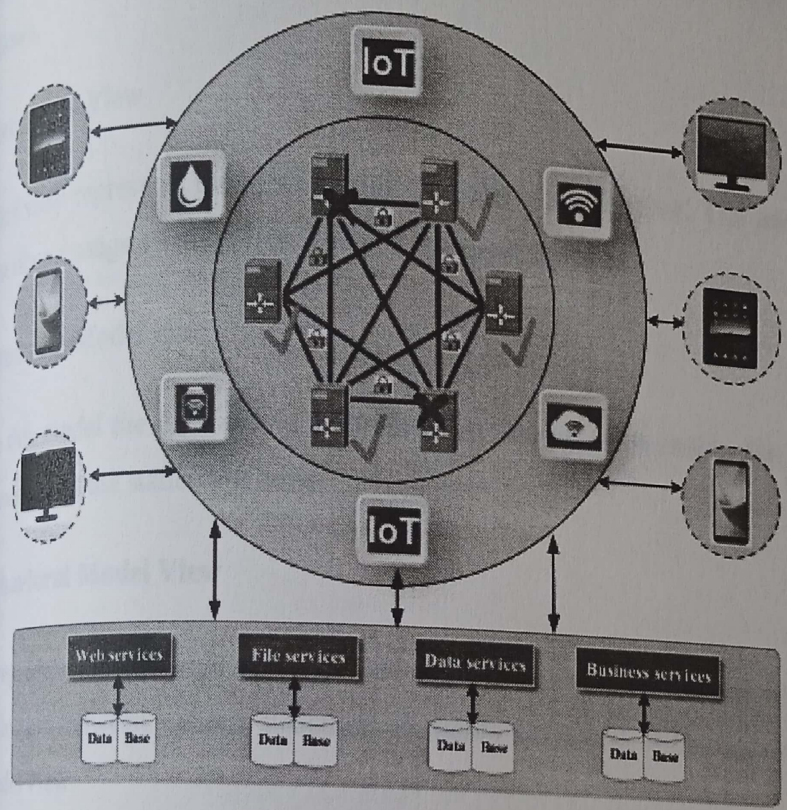


Figure 7.1: Architecture diagram

7.3 UML DIAGRAMS

The Unified Modeling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

User Model View

This view represents the system from the user's perspective. The analysis representation describes a usage scenario from the end-users perspective.

Structural Model view

In this model the data and functionality are arrived from inside the system. This model view models the static structures.

Behavioral Model View

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

Implementation Model View

In this the structural and behavioral as parts of the system are represented as they are to be built.

5.3.1 USE CASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the

different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

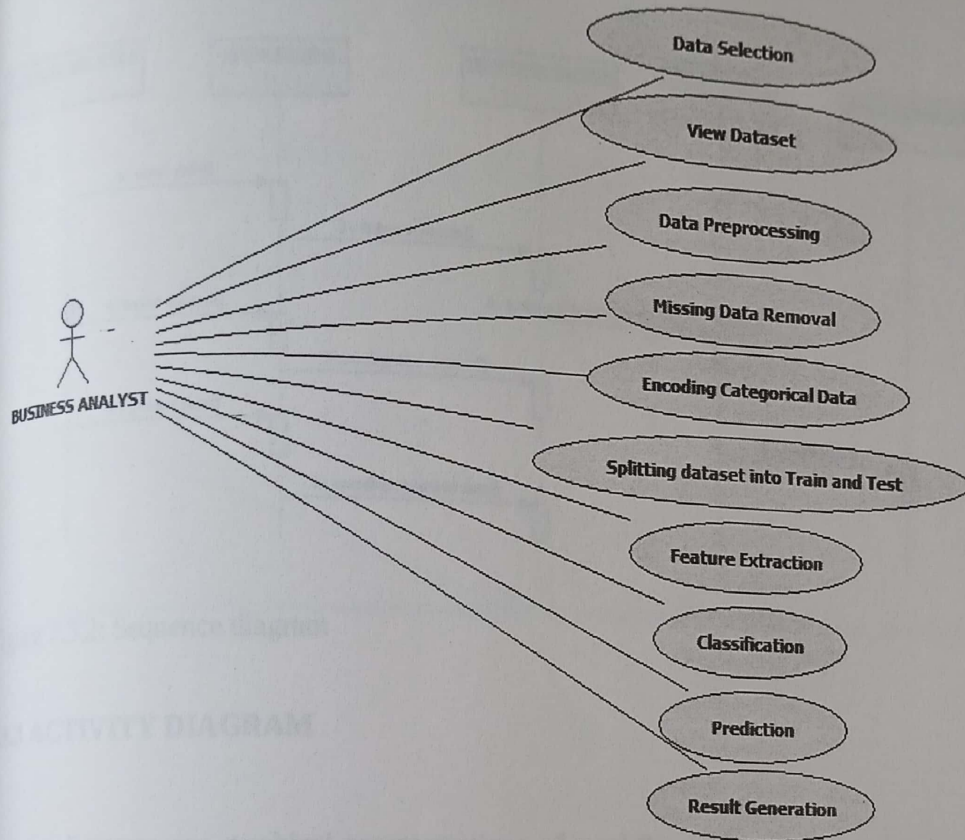


Figure 7.3.1 Use Case Diagram

5.3.2 SEQUENCEDIAGRAM

A sequence diagram 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. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged

between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

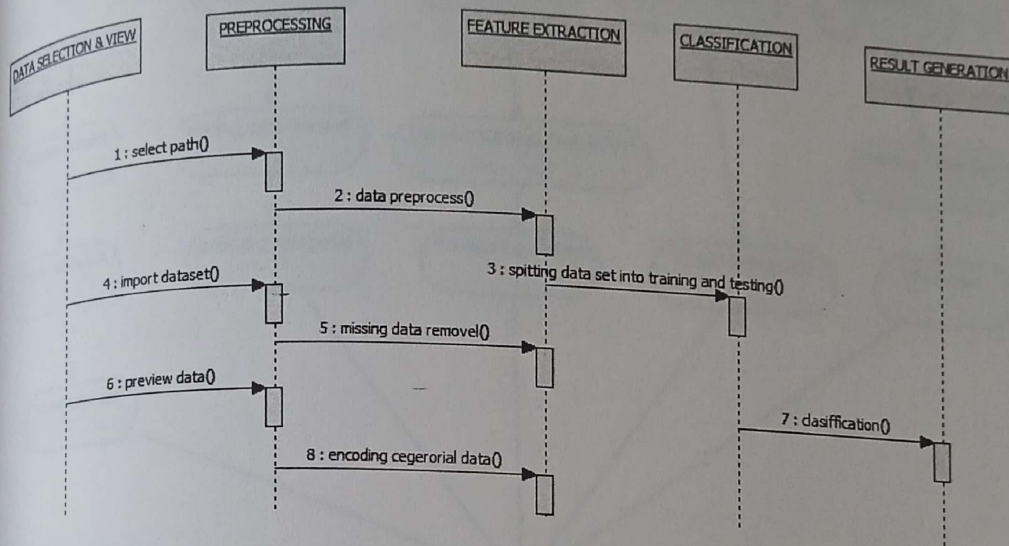


Figure 7.3.2: Sequence diagram

5.3.3 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

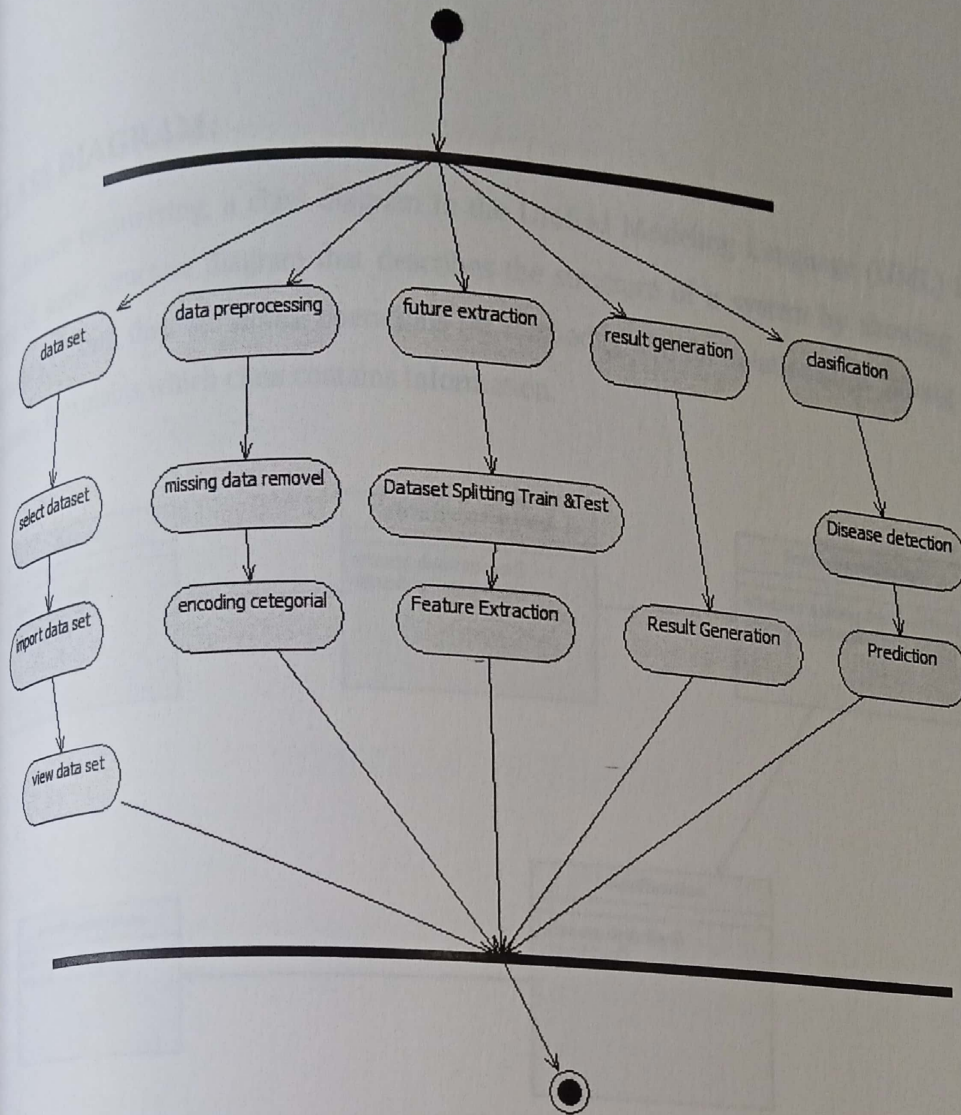


Figure 7.3.4: Class Diagram

Figure 7.3.3: Activity Diagram

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.

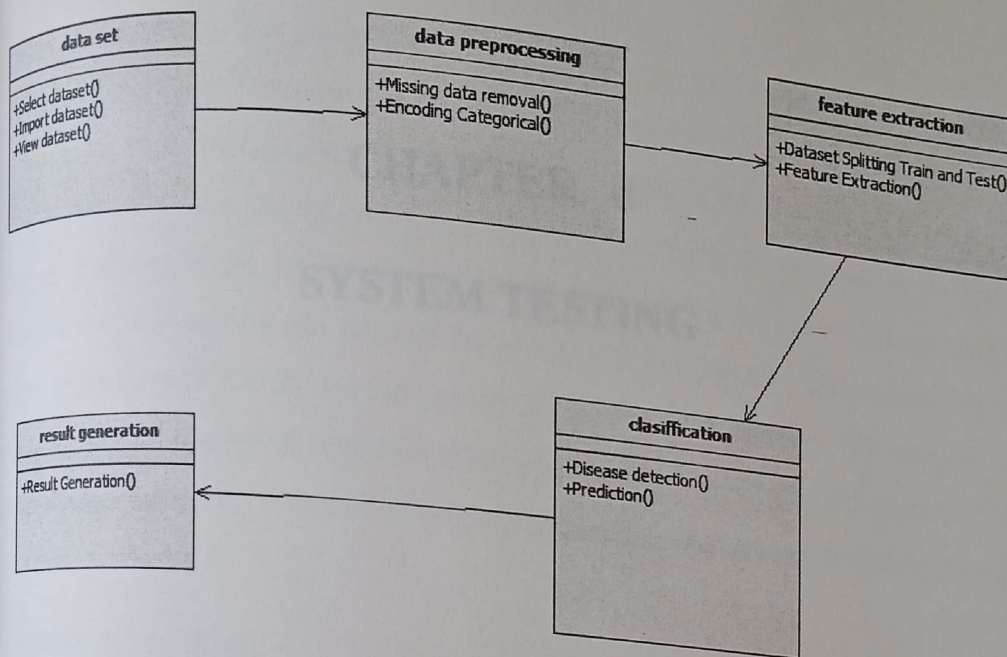


Figure 7.3.4: Class Diagram

CHAPTER 8
SYSTEM TESTING

8. TESTING

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. The following is the description of the testing strategies, which were carried out during the testing period.

8.1 SYSTEM TESTING

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

8.2 MODULE TESTING

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

8.3 INTEGRATION TESTING

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system

8.4 ACCEPTANCE TESTING

When that user find no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

8.5 TEST CASES:

Test Case Id	Test Case Name	Test Case Desc.	Test Steps			Test Case Status	Test Priority
			Step	Expected	Actual		
01	Upload the tasks dataset	Verify either file is loaded or not	If dataset is not uploaded	It cannot display the file loaded message	File is loaded which displays task waiting time	High	High
02	Upload patients dataset	Verify either dataset loaded or not	If dataset is not uploaded	It cannot display dataset reading process completed	It can display dataset reading process completed	low	High
03	Preprocessing	Whether preprocessing on the dataset applied or not	If not applied	It cannot display the necessary data for further process	It can display the necessary data for further process	Medium	High
04	Prediction Random Forest	Whether Prediction algorithm applied on	If not applied	Random tree is not generated	Random tree is generated	High	High

	the data or not					
Recommendation	Whether predicted data is displayed or not	If not displayed	It cannot view prediction containing patient data	It can view prediction containing patient data	High	High
Noisy Records Chart	Whether the graph is displayed or not	If graph is not displayed	It does not show the variations in between clean and noisy records	It shows the variations in between clean and noisy records	Low	Medium

TABLE 8.5.1 TESTCASES

CHAPTER 9

SCREEN SHOTS

9. SCREEN SHOTS

Blockchain-based Decentralized Authentication Modelling Scheme in Edge and IoT Environment

In any online application authentication is the key process to access its services such as Online Banking, insurances access, health record access and many more. All existing online applications are dependent on single centralized server which contains user details and this details will be check while authenticating users but this centralized servers will be managed by internal employees and this employees can misuse user's database or attackers may hack this server database to steal user information. To secure user details existing servers can make use of encryption technologies but by monitoring keys and decryption process internal employees may recover keys and decrypt data.

Sometime some malicious hackers may crash server by sending enormous request and in such situation server will be dead and services to normal users will be disturbed. Currently all existing servers provides various types of authentication such as digital certificate using public and private keys authentication, biometric authentication and username and password based authentication and all this authentication techniques were carried out by single centralized server and can be hacked or alter by malicious users.

To overcome from this problem author of this paper introducing Blockchain technology to decentralized and secure current authentication techniques. As Blockchain servers does not require any human management and form a decentralized group of networks where each and every node or IOT device will share user details and if any device down then user authentication can be carried out form other working IOT device. In Blockchain each node will store data as block/transaction and associate each block with unique hash code

and this hash code will connect all previous and new data block as chain. Before storing new data block all nodes will perform verification of hash codes of all block to check any data block is altered or not and if not altered then data is secure and due to this verification tampering of data in Blockchain is impossible. In propose paper each IOT device server will form a group of network and user can perform authentication from any decentralized IOT devices.

So by using Blockchain with IOT devices users are not depend on single centralized server and their data cannot be tamper due to Blockchain inbuilt hash verification technique.

To build Blockchain based authentication we need to designed smart contract which will be deployed on Blockchain Ethereum tool and application will call this Blockchain smart contract to store user registration details and then this details can be access for verification. All user registration data will be encrypted using ECC (elliptic curve cryptography) algorithm. Below screen showing smart contract developed for authentication process.

In above screen we designed smart contract with two functions for storing and getting authentication details. To deploy this contract first double click on 'deployContract.bat' file from 'hello-eth-blockchain-master' folder to get below screen


```
1 pragma solidity >= 0.8.11 <= 0.8.11;
2
3 contract BlockAuth {
4     string public data;
5
6     function setAuthentication(string memory d) public {
7         data = d;
8     }
9
10    function getAuthentication() public view returns (string memory) {
11        return data;
12    }
13
14    constructor() public {
15        data = "empty";
16    }
17 }
```

In above screen we designed smart contract with two functions for setting and getting authentication details. To deploy this contract just double click on 'deploycontract.bat' file from 'hello-eth/node-modules/.bin' folder to get below screen


```
File Edit Format Run Options Window Help
from django.shortcuts import render
from django.contrib import messages
from django.http import HttpResponseRedirect
from django.core.files.storage import FileSystemStorage
import os
from datetime import date
import json
from web3 import Web3, HTTPProvider
import time
from ecdsa.util import generate_eth_key, generate_key
from ecdsa import encrypt, decrypt
from hashlib import sha1
import base64
import pickle

global details
details = {}
global secret_key, private_key, public_key

#function to generate ECG public, private and secret key
def generateKeys():
    global secret_key, private_key, public_key
    if os.path.exists('public.pkl'):
        f = open('public.pkl', 'rb')
        public_key = pickle.load(f)
        f.close()
    f = open('private.pkl', 'wb')
    private_key = generate_eth_key()
    secret_key = generate_key()
    public_key = secret_key.public_key.to_hex()
    f = open('public.pkl', 'wb')
    pickle.dump(public_key, f)
    f.close()
    f = open('private.pkl', 'wb')
    pickle.dump(private_key, f)
    f.close()
```

Activate Windows
Go to Settings to activate Windows.

Type here to search



Ln: 21 Col: 57

12:39
10-01-2022


```
File Edit Format Run Options Window Help
generateKeys()
def saveDataToBlockchain(currentData, flag): #calling to save data in blockchain
    global details
    blockchain_address = 'http://127.0.0.1:9545'
    web3 = Web3(HTTPProvider(blockchain_address))
    web3.eth.defaultAccount = web3.eth.accounts[0]
    compiled_contract_path = 'BlockAuth.json'
    deployed_contract_address = '0x1c0c4fb431cd769f32c3a6d448c0197d4d405b' #block chain contract address where BlockAuth contract deployed
    with open(compiled_contract_path) as file:
        contract_json = json.load(file) # load contract info as JSON
        contract_abi = contract_json['abi'] # fetch contract's abi - necessary to call its functions
    file.close()
    contract = web3.eth.contract(address=deployed_contract_address, abi=contract_abi) #build web3 object on given contract address
    readDetails()
    if flag == False:
        details=currentData
    else:
        details = currentData
    msg = contract.functions.setAuthentication(details).transact() #now call setAuthentication function to store details in Blockchain
    tx_receipt = web3.eth.waitForTransactionReceipt(msg)

def readDetails(): #calling to read data from blockchain
    global details
    blockchain_address = 'http://127.0.0.1:9545' #Blockchain connection IP
    web3 = Web3(HTTPProvider(blockchain_address))
    web3.eth.defaultAccount = web3.eth.accounts[0]
    compiled_contract_path = 'BlockAuth.json' #BlockAuth contract code
    deployed_contract_address = '0x1c0c4fb431cd769f32c3a6d448c0197d4d405b' #hash address to access industrial contract
    with open(compiled_contract_path) as file:
        contract_json = json.load(file) # load contract info as JSON
        contract_abi = contract_json['abi'] # fetch contract's abi - necessary to call its functions
    file.close()
    contract = web3.eth.contract(address=deployed_contract_address, abi=contract_abi) #now calling contract to access data
    details = contract.functions.getAuthentication().call() #getting authentication data from Blockchain
    if len(details) > 0:
        if 'empty' in details:
            details = details[5:len(details)]
```

In above screen read red colour comments to know how Blockchain smart contract will be called to perform authentication.

In propose paper author describe about only user registration and authentication technique so we build online python DJNAGO web application with two modules

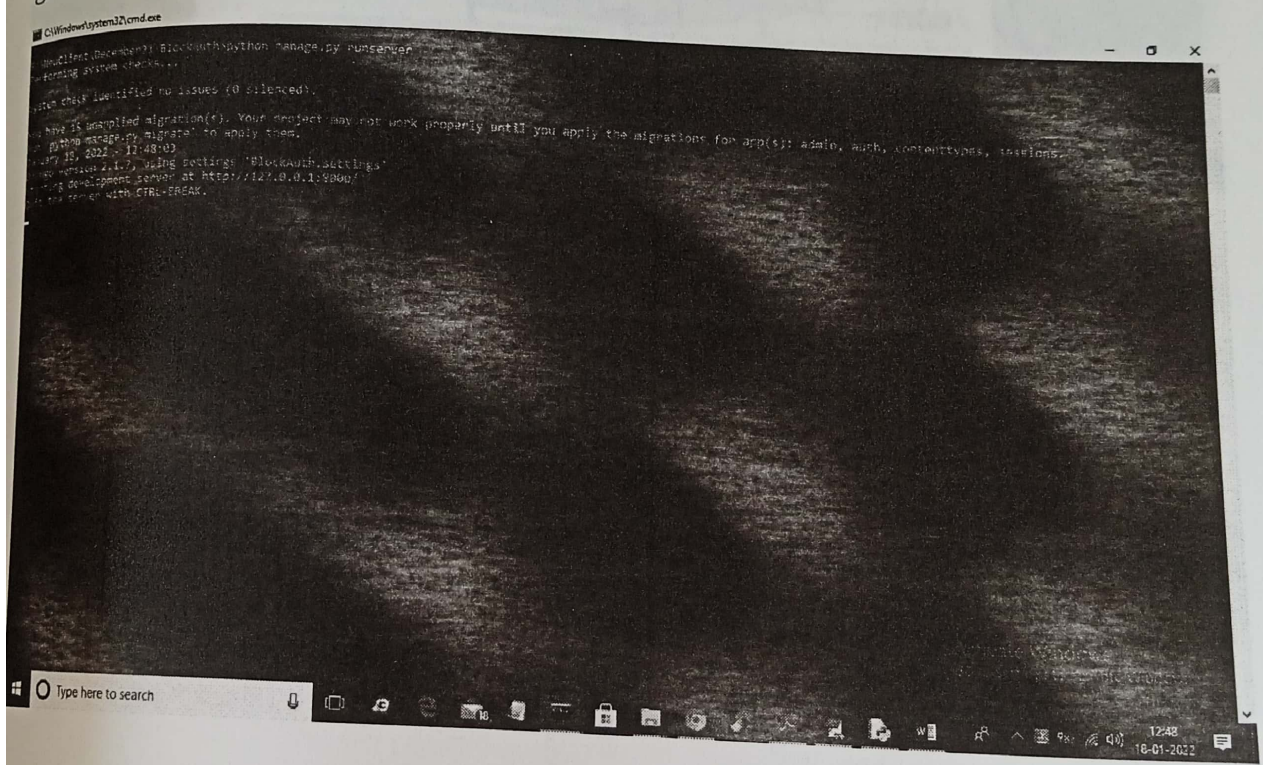
- 1) BlockAuth User Registration: using this module user can enter signup details and then DJNAGO IOT server will accept all this details and then encrypt data using ECC algorithm and then generate hash code and then call Blockchain setAuthentication function to store user signup details. For each user sign data Blockchain will generate certificate and

saved it as block

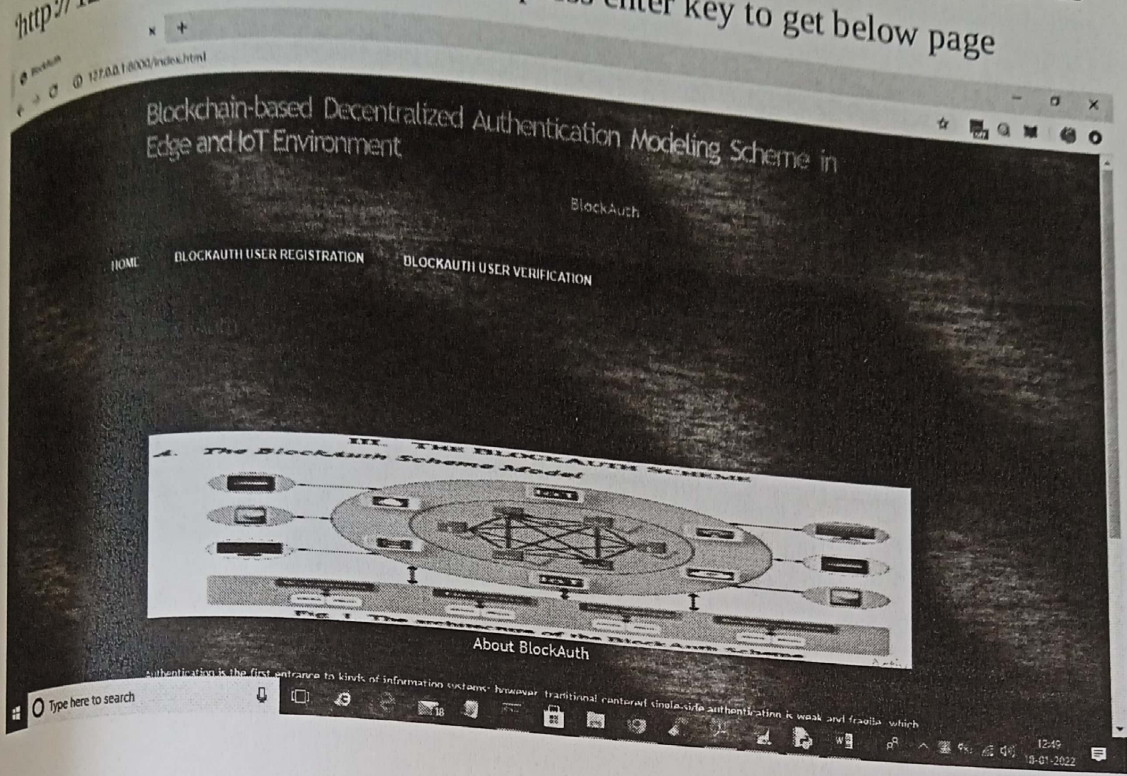
2) BlockAuth User Verification: using this module user can enter username and password and then application will generate public and private keys on user details and then decrypt the data and if user details are valid then valid keys will be generated and then data verification will be completed and this verified data will be check with Blockchain data and if matched found then user authentication will be successful.

SCREEN SHOTS

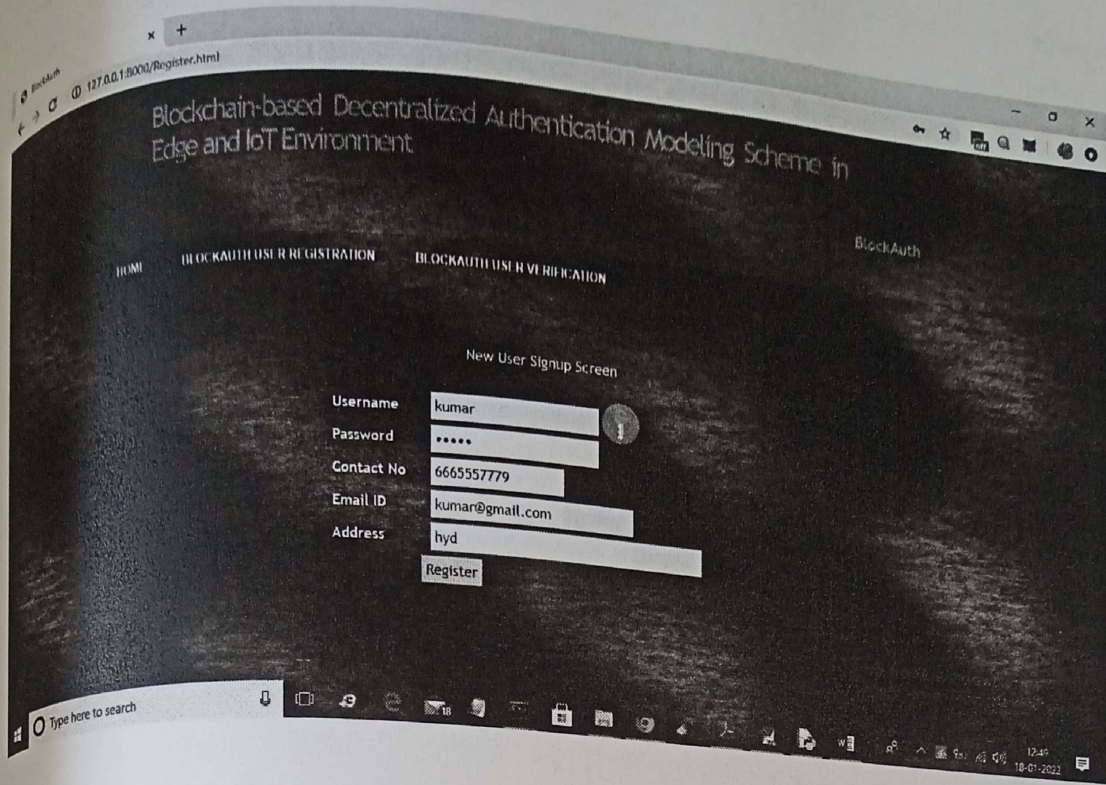
To run project double click on 'run.bat' file to start DJANGO server and to get below screen



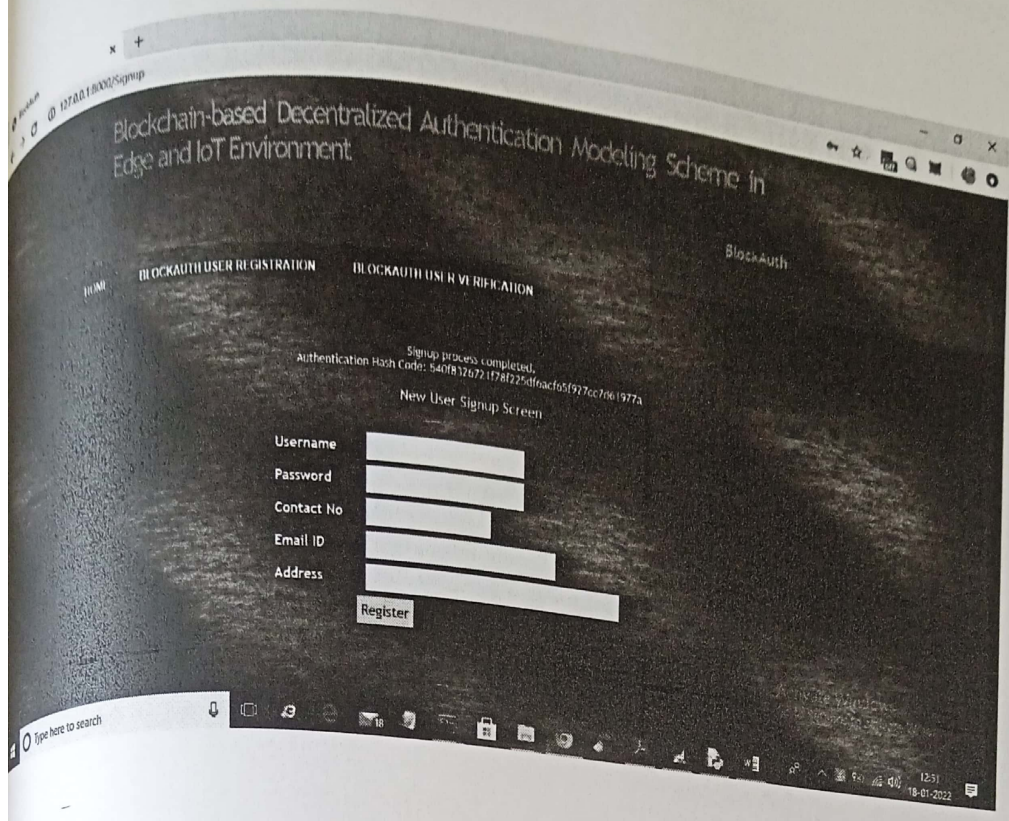
In above screen server started and now open browser and enter URL as 'http://127.0.0.1:8000/index.html' and press enter key to get below page



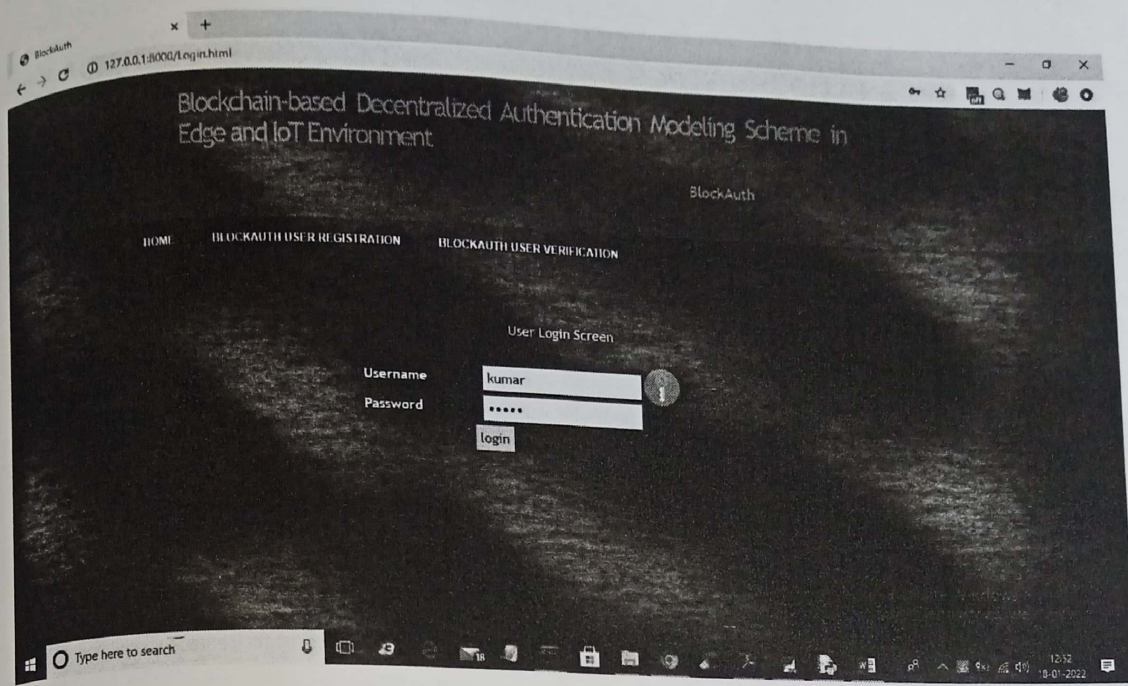
In above screen click on 'BlockAuth User Registration' link to get below screen



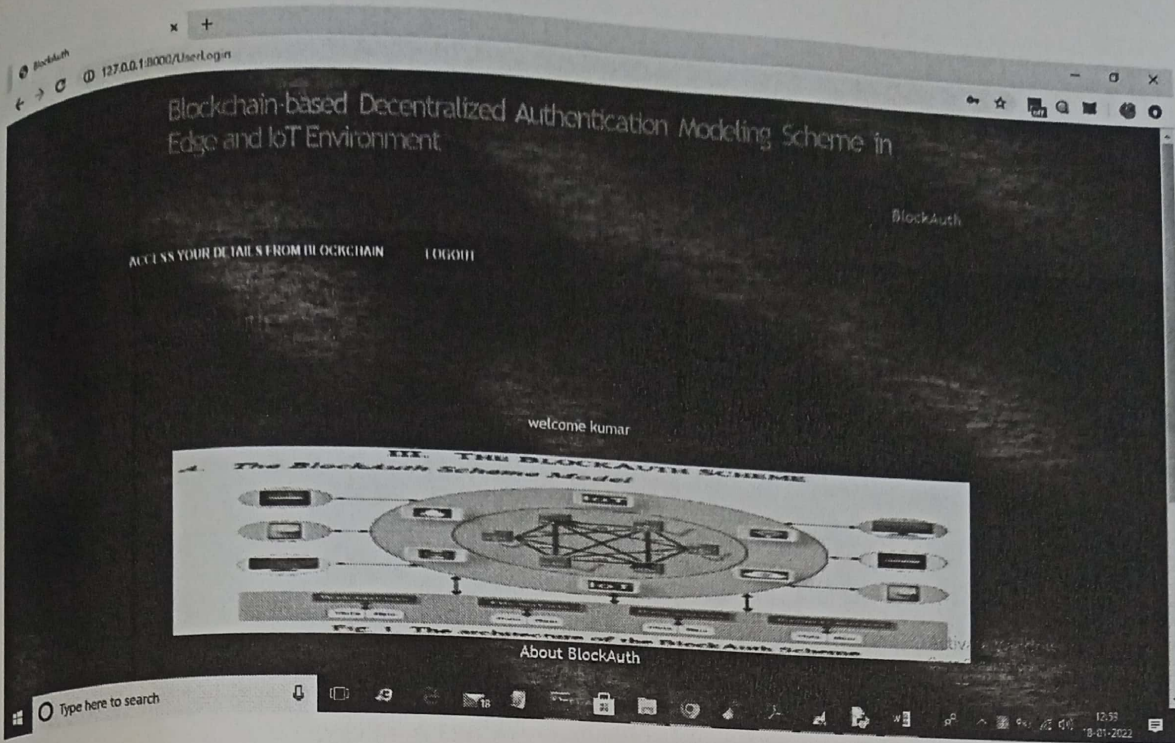
In above screen user can enter signup request details and then this details will be stored in Blockchain using technique describe in above paragraphs. Now click 'Register' button to get below output



In above screen in white colour text we can see user signup process completed and we can see authentication hash code where this block of data stored. Now click on 'BlockAuth User Verification' link to get below login screen

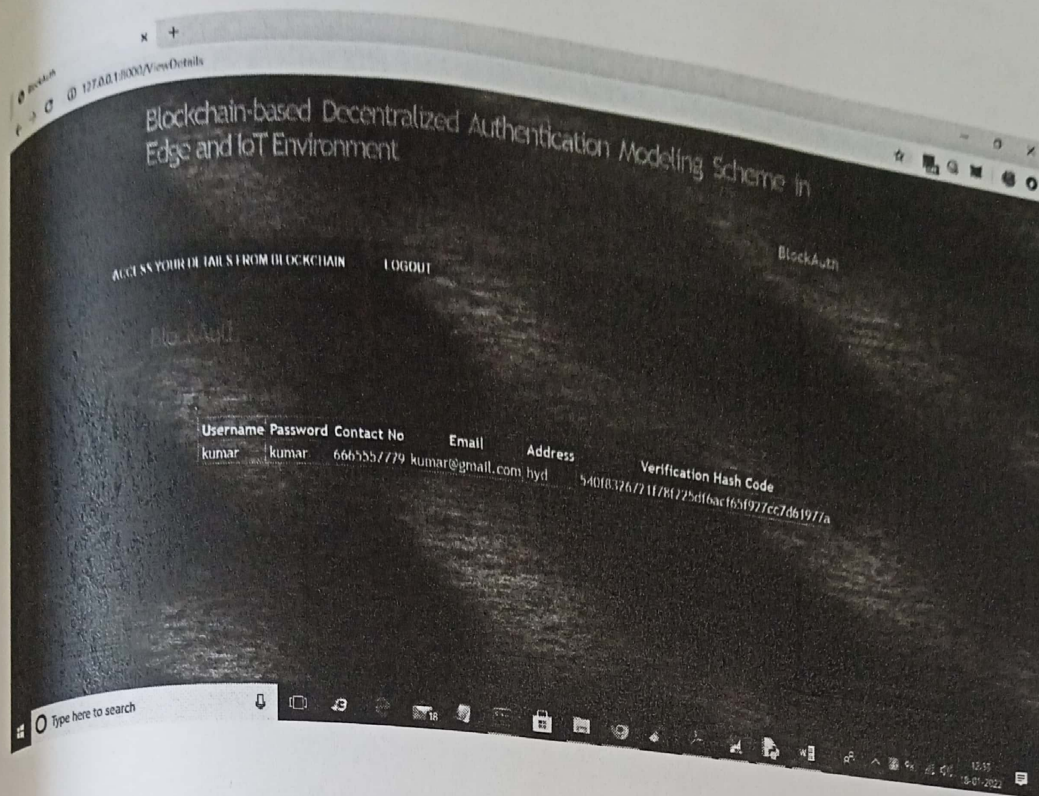


In above screen user will enter his authentication data and press 'Login' button to get below screen and if user details incorrect then authentication will be failed



In above screen user authentication successful so we got 'welcome' message and user can click on 'Access Your Details from Blockchain' link to recover his data stored in Blockchain

As seen how we can perform authentication and registration process using Blockchain technology. Now please don't close Blockchain deployed contract black console screen as it's a Blockchain server so you need to let it running and below I am showing the Blockchain deployed screen which must be running



In above screen user can see all his details which is given at signup time and Blockchain returned verification hash code address where this data is saved. So in above screens we have seen how we can perform authentication and registration process using Blockchain technology.

Note: please don't close Blockchain deployed contract black console screen as it's a Blockchain server so you need to let it running and below I am showing that Blockchain deployed screen which must be running

10. CONCLUSION

In conclusion, the proposed paper introduces a blockchain-based decentralized authentication solution for edge and IoT environments to address the vulnerabilities of existing centralized authentication systems. By leveraging blockchain technology, the system aims to enhance the security, reliability, and scalability of authentication processes.

The traditional centralized authentication systems are prone to various risks, such as unauthorized access, data breaches, and server crashes. These risks can be mitigated by decentralizing the authentication process using blockchain technology. In a blockchain network, each IoT device or node stores user details, and if one device fails, authentication can still be carried out from other working devices. This decentralized approach reduces the reliance on a central server and enhances system resilience.

CHAPTER 10

CONCLUSION

The use of blockchain in authentication introduces several key advantages. First, the data stored in the blockchain is secured through encryption and hash verification techniques, making it nearly impossible to tamper with or alter the data. Additionally, the distributed nature of blockchain eliminates the need for human management, reducing the risk of internal employees misusing user databases. Furthermore, blockchain-based authentication can support various authentication techniques such as digital certificates, biometrics, and username/password-based authentication.

To implement the proposed blockchain-based authentication system, a smart contract is designed and deployed on the Ethereum blockchain. This smart contract handles user registration details, ensuring secure storage and accessibility for verification purposes. The registration data is encrypted using elliptic curve cryptography (ECC), adding an additional layer of security to the system.

Overall, the use of blockchain technology in edge and IoT environments provides a decentralized and secure authentication solution, reducing the vulnerabilities associated with traditional centralized systems.

10. CONCLUSION

In conclusion, the proposed paper introduces a blockchain-based decentralized authentication modeling scheme in edge and IoT environments to address the limitations and vulnerabilities of existing centralized authentication systems. By leveraging blockchain technology, the authors aim to enhance the security, reliability, and availability of authentication processes.

The traditional centralized authentication systems are prone to various risks, including unauthorized access, data breaches, and server crashes. These risks can be mitigated by decentralizing the authentication process using blockchain technology. In a blockchain network, each IoT device or node shares user details, and if one device fails, authentication can still be carried out from other working devices. This decentralized approach reduces the reliance on a single centralized server and enhances system resilience.

The use of blockchain in authentication introduces several key advantages. First, the data stored in the blockchain is secured through encryption and hash verification techniques, making it nearly impossible to tamper with or alter the data. Additionally, the distributed nature of blockchain eliminates the need for human management, reducing the risk of internal employees misusing user databases. Furthermore, blockchain-based authentication can support various authentication techniques such as digital certificate, biometrics, and username/password-based authentication.

To implement the proposed blockchain-based authentication system, a smart contract is designed and deployed on the Ethereum blockchain. This smart contract handles user registration details, ensuring secure storage and accessibility for verification purposes. User registration data is encrypted using elliptic curve cryptography (ECC), adding an extra layer of security to the system.

Overall, the use of blockchain technology in edge and IoT environments provides a decentralized and secure authentication solution, reducing the vulnerabilities associated

with centralized servers. By leveraging the benefits of blockchain, online applications can offer enhanced protection of user data, ensuring privacy, integrity, and availability of services. Future research in this area could focus on real-world implementation and performance evaluation of the proposed scheme to validate its effectiveness in practical applications..

Future Enhancements:

It is not possible to develop a system that makes all the requirements of the user. User requirements keep changing as the system is being used. Some of the future enhancements that can be done to this system are:

- As the technology emerges, it is possible to upgrade the system and can be adaptable to desired environment.
- Based on the future security issues, security can be improved using emerging technologies like single sign-on.

REFERENCES

References

ITU-RM.2083-0, IMT Vision Framework and Overall Objectives of the Future Development of IMT for 2020 and Beyond, International Telecommunication Union, Geneva, Switzerland, 2015.

J. Cao, P. Yu, M. Ma, and W. Gao, "Fast authentication and data transfer scheme for massive NB-IoT devices in 3GPP 5G network," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1561–1575, 2019.

[View at: Publisher Site | Google Scholar](#)

J. Cao, P. Yu, X. Xiang, M. Ma, and H. Li, "Anti-quantum fast authentication and data transmission scheme for massive devices in 5G NB-IoT system," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9794–9805, 2019.

[View at: Publisher Site | Google Scholar](#)

J. Li, M. Wen, and T. Zhang, "Group-based authentication and key agreement with dynamic policy updating for MTC in LTE-A networks," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 408–417, 2016.

[View at: Publisher Site | Google Scholar](#)

D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Blockchain for 5G and beyond networks: a state of the art survey," *Journal of Network and Computer Applications*, vol. 166, Article ID 102693, 2020.

[View at: Publisher Site | Google Scholar](#)

J. Xing zhong, X. Qingshui, M. Haifeng, C. Jiageng, and Z. Haozhi, "The research on identity authentication scheme of internet of things equipment in 5G network environment," in *Proceedings of the International Conference on Communication*

Technology, ICCT, pp. 312–316, Xi'an, China, October 2019.

View at: [Publisher Site](#) | [Google Scholar](#)

I. Psaras, “Decentralised edge-computing and IoT through distributed trust,” in *Proceedings of the MobiSys 2018—16th ACM International Conference on Mobile Systems, Applications, and Services*, pp. 505–507, Munich, Germany, June 2018.

View at: [Publisher Site](#) | [Google Scholar](#)

S. Behrad, E. Bertin, S. Tuffin, and N. Crespi, “A new scalable authentication and access control mechanism for 5G-based IoT,” *Future Generation Computer Systems*, vol. 108, pp. 46–61, 2020.

View at: [Publisher Site](#) | [Google Scholar](#)

N. K. Pratas, S. Pattathil, C. Stefanovic, and P. Popovski, “Massive machine-type communication (mMTC) access with integrated authentication,” in *Proceedings of the 2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, Paris, France, May 2017.

View at: [Publisher Site](#) | [Google Scholar](#)

M. Nasimi, M. A. Habibi, B. Han, and H. D. Schotten, “Edge-assisted congestion control mechanism for 5G network using software-defined networking,” in *Proceedings of the 2018 15th International Symposium on Wireless Communication Systems (ISWCS)*, pp. 1–5, Lisbon, Portugal, August 2018.

View at: [Publisher Site](#) | [Google Scholar](#)

S. Hong, “P2P networking based internet of things (IoT) sensor node authentication by Blockchain,” *Peer-to-Peer Networking and Applications*, vol. 13, no. 2, pp. 579–589, 2020.

View at: [Publisher Site](#) | [Google Scholar](#)

Z. Nezami, K. Zamanifar, K. Djemame, and E. Pournaras, “Decentralized edge-to-cloud

load-balancing: service placement for the Internet of Things," 2020,
<http://arxiv.org/abs/2005.00270>.

View at: [Google Scholar](#)

M. El-Hajj, A. Fadlallah, M. Chamoun, and A. Serhrouchni, "A survey of internet of things (IoT) authentication schemes," *Sensors*, vol. 19, no. 5, pp. 1141–1143, 2019.

View at: [Publisher Site](#) | [Google Scholar](#)

L. Kou, Y. Shi, L. Zhang, D. Liu, and Q. Yang, "A lightweight three-factor user authentication protocol for the information perception of IoT," *Computers, Materials & Continua*, vol. 58, no. 2, pp. 545–565, 2019.

View at: [Publisher Site](#) | [Google Scholar](#)

C. Ellison and B. Schneier, "Ten risks of PKI: what you are not being told about public key infrastructure," *Public Key Infrastructure: Building Trusted Applications and Web Services*, vol. 14, no. 1, pp. 299–306, 2004.

View at: [Publisher Site](#) | [Google Scholar](#)

Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "When mobile blockchain meets edge computing," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 33–39, 2018.

View at: [Publisher Site](#) | [Google Scholar](#)

S. Guo, X. Hu, S. Guo, X. Qiu, and F. Qi, "Blockchain meets edge computing: a distributed and trusted authentication system," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1972–1983, 2020.

View at: [Publisher Site](#) | [Google Scholar](#)

R. Almadhoun, M. Kadadha, M. Alhemeiri, M. Alshehhi, and K. Salah, "A user authentication scheme of IoT devices using blockchain-enabled fog nodes," in *Proceedings of the 2018 IEEE/ACS 15th International Conference on Computer Systems and*

Applications (AICCSA), pp. 1–8, Aqaba, Jordan, October-November 2018.

[View at: Publisher Site | Google Scholar](#)

T. Salman, M. Zolanvari, A. Erbad, R. Jain, and M. Samaka, "Security services using blockchains: a state of the art survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 858–880, 2019.

[View at: Publisher Site | Google Scholar](#)

T. Hewa, A. Braeken, M. Ylianttila, and M. Liyanage, "Blockchain based Automated Certificate Revocation for 5G IoT," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Dublin, Ireland, June 2020.

[View at: Google Scholar](#)

X. Jia, N. Hu, S. Su et al., "IRBA: an identity-based cross-domain authentication scheme for the internet of things," *Electronics*, vol. 9, no. 4, p. 634, 2020.

[View at: Publisher Site | Google Scholar](#)

K. Kaur, S. Garg, G. Kaddoum, F. Gagnon, and S. H. Ahmed, "Blockchain-based lightweight Authentication mechanism for vehicular fog infrastructure," in *Proceedings of the 2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, Shanghai, China, May 2019.

[View at: Publisher Site | Google Scholar](#)

R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan, "Internet of things (IoT) security: Current status, challenges and prospective measures," in *Proceedings of the 2015 10th International Conference for Internet Technology and Secured Transactions, ICITST 2015*, pp. 336–341, London, UK, December 2015.

[View at: Publisher Site | Google Scholar](#)

A. Esfahani, G. Mantas, R. Maticsek et al., "A lightweight Authentication mechanism for

M2M communications in industrial IoT environment," IEEE Internet of Things Journal, vol. 6, no. 1, pp. 288–296, 2019.

View at: Publisher Site | Google Scholar

J. Ni, X. Lin, and X. S. Shen, "Efficient and secure service-oriented authentication supporting network slicing for 5G-enabled IoT," IEEE Journal on Selected Areas in Communications, vol. 36, no. 3, pp. 644–657, 2018.

View at: Publisher Site | Google Scholar

H. Gross, M. Hölbl, D. Slamanig, and R. Spreitzer, "Privacy-Aware Authentication in the Internet of Things," in Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), M. Reiter and D. Naccache, Eds., vol. 9476, pp. 32–39, Springer International Publishing, Cham, Switzerland, 2015.

View at: Google Scholar

C. Lai, H. Li, X. Liang et al., "CPAL: a conditional privacy-preserving authentication with access linkability for roaming service," IEEE Internet of Things Journal, vol. 1, no. 1, pp. 46–57, 2014.

View at: Publisher Site | Google Scholar

M. T. Hammi, B. Hammi, P. Bellot, and A. Serhrouchni, "Bubbles of Trust: a decentralized blockchain-based authentication system for IoT," Computers & Security, vol. 78, pp. 126–142, 2018.

View at: Publisher Site | Google Scholar

Z. Bao, W. Shi, D. He, and K.-K. R. Chood, "IoTChain: a three-tier blockchain-based IoT security architecture," 2018, <http://arxiv.org/abs/1806.02008>.

View at: Google Scholar

U. Khalid, M. Asim, T. Baker, P. C. K. Hung, M. A. Tariq, and L. Rafferty, "A decentralized lightweight blockchain-based authentication mechanism for IoT systems," *Cluster Computing*, vol. 23, no. 3, p. 2067, 2020.

View at: [Publisher Site](#) | [Google Scholar](#)

H. Liu, P. Zhang, G. Pu, T. Yang, S. Maharjan, and Y. Zhang, "Blockchain empowered cooperative authentication with data traceability in vehicular edge computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4221–4232, 2020.

View at: [Publisher Site](#) | [Google Scholar](#)

Z. Cui, F. Xue, S. Zhang et al., "A hybrid BlockChain-based identity authentication scheme for multi-WSN," *IEEE Transactions on Services Computing*, vol. 13, no. 2, p. 1, 2020.

View at: [Publisher Site](#) | [Google Scholar](#)

N. Shahin, R. Ali, S. Y. Nam, and Y.-T. Kim, "Performance evaluation of centralized and distributed control methods for efficient registration of massive IoT devices," in *Proceedings of the 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*, vol. 2018, pp. 314–319, Prague, Czech Republic, July 2018.

View at: [Publisher Site](#) | [Google Scholar](#)

O. Alphand, M. Amoretti, T. Claeys et al., "IoTChain: a blockchain security architecture for the Internet of Things," in *Proceedings of the IEEE Wireless Communications and Networking Conference, WCNC*, pp. 1–6, Barcelona, Spain, April 2018.

View at: [Publisher Site](#) | [Google Scholar](#)

C. Lin, D. He, N. Kumar, X. Huang, P. Vijayakumar, and K.-K. R. Choo, "HomeChain: a blockchain-based secure mutual authentication system for smart homes," *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 818–829, 2020.

View at: [Publisher Site](#) | [Google Scholar](#)

M. A. Jan, W. Zhang, M. Usman, Z. Tan, F. Khan, and E. Luo, "SmartEdge: an end-to-end encryption framework for an edge-enabled smart city application," *Journal of Network and Computer Applications*, vol. 137, pp. 1–10, 2019.

View at: [Publisher Site](#) | [Google Scholar](#)

M. Shen, H. Liu, L. Zhu et al., "Blockchain-assisted secure device authentication for cross-domain industrial IoT," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 942–954, 2020.

View at: [Publisher Site](#) | [Google Scholar](#)

M. Ma, G. Shi, and F. Li, "Privacy-oriented blockchain-based distributed key management architecture for hierarchical access control in the IoT scenario," *IEEE Access*, vol. 7, pp. 34045–34059, 2019.

View at: [Publisher Site](#) | [Google Scholar](#)

M. A. Xiaoting, M. A. Wenping, and L. I. U. Xiaoxue, "A cross domain authentication scheme based on blockchain technology," *Acta Electronica Sinica*, vol. 46, no. 11, pp. 2571–2579, 2018.

View at: [Google Scholar](#)

Y. Chen, G. Dong, J. Bai, Y. Hao, F. Li, and H. Peng, "Trust Enhancement Scheme for Cross Domain Authentication of PKI System," in *Proceedings of the 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 103–110, Guilin, China, October 2019.

View at: [Publisher Site](#) | [Google Scholar](#)

J. Cui, L. Wei, J. Zhang, Y. Xu, and H. Zhong, "An efficient message-authentication scheme based on edge computing for vehicular ad hoc networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1621–1632, 2019.

View at: Publisher Site | Google Scholar

S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 160–168, 2019.

View at: Publisher Site | Google Scholar

Y. Jiang, C. Wang, Y. Wang, and L. Gao, "A cross-chain solution to integrating multiple blockchains for IoT data management," *Sensors*, vol. 19, no. 9, pp. 2042–2060, 2019.

View at: Publisher Site | Google Scholar

G.-H. Hwang, P.-H. Chen, C.-H. Lu et al., "A multi-chain architecture with distributed auditing of sidechains for public blockchains," in *Proceedings of the Blockchain-ICBC*, vol. 10974, pp. 47–60, Springer International Publishing, Honolulu, HI, USA, September 2018.

View at: Google Scholar

A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology*, vol. 196 LNCS, pp. 47–53, Springer Berlin Heidelberg, Berlin, Heidelberg, 1985.

View at: Google Scholar

F. Yuan and Z. Cheng, "Overview on SM9 identity-based cryptographic algorithm," *Journal of Information Security Research*, vol. 2, no. 11, pp. 1008–1027, 2016.

View at: Google Scholar

A. Back, M. Corallo, and L. Dashjr, "Enabling blockchain innovations with pegged sidechains," pp. 1–25, 2014, <http://newspaper23.com/ripped/2014/11/>, <http://www.blockstream.com.sidechains.pdf>.

View at: Google Scholar

A. Garoffolo, D. Kaidalov, and R. Oliynykov, "Zendoo: a zk-SNARK verifiable cross-

chain transfer protocol enabling decoupled and decentralized sidechains," 2020,
<http://arxiv.org/abs/2002.01847>.

View at: Google Scholar

S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," 2009,
<https://bitcoin.org/en/bitcoin-paper>.

View at: Publisher Site | Google Scholar

```
#!/usr/bin/env python
import os
import sys
import django
from django.conf import settings
from django.core.management import execute_from_command_line

if __name__ == '__main__':
    os.environ.setdefault('DJANGO_SETTINGS_MODULE',
        'your_app.settings')
    django.setup()
    execute_from_command_line(sys.argv)
```

SAMPLE CODE

```
except ImportError:
    raise ImportError(
        "Couldn't import Django. Are you sure it's installed and
        available on your PYTHONPATH environment variable? Did you
        forget to activate a virtual environment?"
    )
) from exc
execute_from_command_line(sys.argv) from django.core.management import
```


Sample Code

```
#!/usr/bin/env python
```

```
import os
```

```
import sys
```

```
if __name__ == '__main__':
```

```
    os.environ.setdefault('DJANGO_SETTINGS_MODULE',  
    'BlockAuth.settings')
```

```
    try:
```

```
        from django.core.management import execute_from_command_line
```

```
    except ImportError as exc:
```

```
        raise ImportError(
```

```
            "Couldn't import Django. Are you sure it's installed and "
```

```
            "available on your PYTHONPATH environment variable? Did you "
```

```
            "forget to activate a virtual environment?"
```

```
        ) from exc
```

```
        execute_from_command_line(sys.argv) from django.shortcuts import
```

```
render
from django.template import RequestContext
from django.contrib import messages
from django.http import HttpResponse
from django.core.files.storage import FileSystemStorage
import os
from datetime import date
import json
from web3 import Web3, HTTPProvider
import time
from ecies.utils import generate_eth_key, generate_key
from ecies import encrypt, decrypt
from hashlib import sha1
import base64
import pickle
import LZW
from LZW import losslessCompress, losslessDecompress #====importing
lossless LZW compression and decompression function
import zlib
import sys
import matplotlib.pyplot as plt
import numpy as np

generateKeys()

global details
details = ""

global secret_key, private_key, public_key
```



```
#function to generate ECC public, private and secret keys
def generateKeys():
```

```
    global secret_key, private_key, public_key
```

```
    if os.path.exists('public.pckl'):
```

```
        f = open('public.pckl', 'rb')
```

```
        public_key = pickle.load(f)
```

```
        f.close()
```

```
        f = open('private.pckl', 'rb')
```

```
        private_key = pickle.load(f)
```

```
        f.close()
```

```
    else:
```

```
        secret_key = generate_eth_key()
```

```
        private_key = secret_key.to_hex() # hex string
```

```
        public_key = secret_key.public_key.to_hex()
```

```
        f = open('public.pckl', 'wb')
```

```
        pickle.dump(public_key, f)
```

```
        f.close()
```

```
        f = open('private.pckl', 'wb')
```

```
        pickle.dump(private_key, f)
```

```
        f.close()
```

```
generateKeys()
```

```
def saveDataBlockChain(currentData,flag): #calling to save data in
```

```

blockchain
global details
global contract
blockchain_address = 'http://127.0.0.1:9545'
web3 = Web3(HTTPProvider(blockchain_address))
web3.eth.defaultAccount = web3.eth.accounts[0]
compiled_contract_path = 'BlockAuth.json'
deployed_contract_address =
'0xc4e71a4aF105dc0eF422322AE80a611Ee9D7Ed71' #block chain contract
address where BlockAuth contract deployed
with open(compiled_contract_path) as file:
    contract_json = json.load(file) # load contract info as JSON
    contract_abi = contract_json['abi'] # fetch contract's abi - necessary to
call its functions
file.close()
contract = web3.eth.contract(address=deployed_contract_address,
abi=contract_abi) #build web3 object on given contract address
readDetails()
if flag == False:
    details+=currentData
else:
    details = currentData
msg = contract.functions.setAuthentication(details).transact() #now call
setAuthentication function to store details in Blockchain
tx_receipt = web3.eth.waitForTransactionReceipt(msg)

```

```

def readDetails(): #calling to read data from blockchain
    global details
    blockchain_address = 'http://127.0.0.1:9545' #Blockchain connection IP
    web3 = Web3(HTTPProvider(blockchain_address))
    web3.eth.defaultAccount = web3.eth.accounts[0]
    compiled_contract_path = 'BlockAuth.json' #blockauth contract code
    deployed_contract_address =
'0xc4e71a4aF105dc0eF422322AE80a611Ee9D7Ed71' #hash address to access
    industrail contract
    with open(compiled_contract_path) as file:
        contract_json = json.load(file) # load contract info as JSON
        contract_abi = contract_json['abi'] # fetch contract's abi - necessary to
call its functions
    file.close()
    contract = web3.eth.contract(address=deployed_contract_address,
abi=contract_abi) #now calling contract to access data
    details = contract.functions.getAuthentication().call() #getting
authentication data from Blockchain
    if len(details) > 0:
        if 'empty' in details:
            details = details[5:len(details)]
    return details

def Signup(request):
    global details
    global secret_key, private_key, public_key

```



```

if request.method == 'POST':
    username = request.POST.get('username', False)
    password = request.POST.get('password', False)
    contact = request.POST.get('contact', False)
    email = request.POST.get('email', False)
    address = request.POST.get('address', False)
    data = username+"$"+password+"$"+contact+"$"+email+"$"+address
    enc = encrypt(public_key, data.encode())
    verification_code = sha1(enc).hexdigest()
    enc = str(base64.b64encode(enc),'utf-8')
    enc = enc+"#"+verification_code+"\n"
    #=====
    #getting size of plain encrypted data
    plain_size = sys.getsizeof(enc)
    #performing compression on encrypted data
    compressData = losslessCompress(enc)
    compress_data = zlib.compress(enc.encode())
    #getting memory size of compressed data
    compress_size = sys.getsizeof(compress_data)
    saveDataBlockChain(enc,False)
    height = [plain_size,compress_size]
    bars = ('Blockchain Plain Storage Size','Blockchain Compress Storage
Size')
    y_pos = np.arange(len(bars))
    #plotting graph with required memory
    plt.bar(y_pos, height)

```

```

plt.xticks(y_pos, bars)
plt.title("Blockchain Memory Required Graph for Plain & LZW
Compressed Data")
plt.show()
context= {'data': 'Signup process completed.<br/>Authentication Hash
Code: '+verification_code+"<br/>Plain Storage Memory :
"+str(plain_size)+"<br/>Compress Storage Memory : "+str(compress_size)}
return render(request, 'Register.html', context)

```

```

def ViewDetails(request):
    global secret_key, private_key, public_key, details
    if request.method == 'GET':
        user = ""
        with open("session.txt", "r") as file:
            for line in file:
                user = line.strip('\n')
        file.close()
        font = "<font size=3 color=white>"
        output="<table border=1
align=center><tr><th>"+font+"Username</th><th>"+font+"Password</th><t
h>"+font+"Contact
No</th><th>"+font+"Email</th><th>"+font+"Address</th>"
        output+='<th>'+font+'Verification Hash Code</th></tr>'
        readDetails()

```

```

arr = details.split("\n")
for i in range(len(arr)):
    if len(arr[i]) > 0:
        values = arr[i].split("#")
        enc = base64.b64decode(values[0])
        decryptData = decrypt(private_key, enc)
        decryptData = decryptData.decode()
        array = decryptData.split("$")
        if array[0] == user:

output+="<tr><td>"+font+array[0]+"</td><td>"+font+array[1]+"</td><td>"+font+array[2]+"</td><td>"+font+array[3]+"</td><td>"+font+array[4]+"</td></tr>"+font+values[1]+"</td></tr>"
context= {'data1':output }
return render(request, 'ViewDetails.html', context)

def Register(request):
    if request.method == 'GET':
        return render(request, 'Register.html', {})

def Logout(request):
    if request.method == 'GET':
        return render(request, 'index.html', {})

def index(request):
    if request.method == 'GET':

```



```

return render(request, 'index.html', {})

def Login(request):
    if request.method == 'GET':
        return render(request, 'Login.html', {})

def authenticateUser(username,password):
    global secret_key, private_key, public_key, details
    flag = False
    readDetails()
    arr = details.split("\n")
    for i in range(len(arr)):
        if len(arr[i]) > 0:
            values = arr[i].split("#")
            enc = base64.b64decode(values[0])
            decryptData = decrypt(private_key, enc)
            decryptData = decryptData.decode()
            print(decryptData)
            array = decryptData.split("$")
            if array[0] == username and array[1] == password:
                flag = True
                break
    return flag

def UserLogin(request):

```

```
if request.method == 'POST':  
    username = request.POST.get('username', False)  
    password = request.POST.get('password', False)  
    if authenticateUser(username,password) == True:  
        file = open('session.txt','w')  
        file.write(username)  
        file.close()  
        context= {'data':'welcome '+username}  
        return render(request, 'UserScreen.html', context)  
    else:  
        context= {'data':'Invalid Login details'}  
        return render(request, 'Login.html', context)
```