A Mini Project Report on

A BLOCKCHAIN BASED SECURE AND EFFICIENT VALIDATION SYSTEM FOR DIGITAL CERTIFICATES

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD

In Partial fulfilment of the requirement for the award of degree of the

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

V. Sai Bharath	**- 20RA1A0566**
P. Sai Ram	**- 20RA1A0572**
P. Sai Rohith	**- 20RA1A0568**

Under the guidance of

Dr. C. Bagath Basha

Head of the Department

Department of Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

KOMMURI PRATAP REDDY INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUH, Ghanpur(V), Ghatkesar(M), Medchal(D)-500088)

2020- 2024

KOMMURI PRATAP REDDY INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUH, Ghanpur(V), Ghatkesar(M), Medchal(D)-500088)

CERTIFICATE

This is to certify that the project work entitled **"A BLOCKCHAIN BASED SECURED AND EFFICIENT VALIDATION SYSTEM FOR DIGITAL CERTIFICATES"** is submitted by Mr. V. Sai Bharath, Mr. P. Sai Ram, Mr. P. Sai Rohith bonafied students of **Kommuri Pratap Reddy Institution of Technology** in partial fulfillment of the requirement for the reward of the Degree of Bachelor of Technology in **Computer Science and Engineering** of the **Jawaharlal Nehru Technological University Hyderabad**, during the year 2020-2024.

Internal Examiner **HOD**

External Examiner

DECLARATION

We hereby declare that this project work entitled **"A BLOCKCHAIN BASED SECURE AND EFFICIENT VALIDATION SYSTEM FOR DIGITAL CERTIFICATES"** in partial fulfillment of requirements for the award of degree of **Computer Science and Engineering** is a bonafide work carried out by us during the academic year 2020- 2024.

We further declare that this project is a result of our effort and has not been submitted for the award of any degree by us to any institute.

By

V. SAI BHARATH	(20RA1A0566)
P. SAI RAM	(20RA1A0572)
P.SAI ROHITH	(20RA1A0568)

ACKNOWLEDGEMENT

It gives us immense pleasure to acknowledge with gratitude, the help and support extended throughout the project report from the following:

We will be very much grateful to almighty our **Parents** who have made us capable of carrying out our job.

We express our profound gratitude to **Dr. P. Srinivas Rao, Principal of Kommuri Pratap Reddy Institute of Technology**, who has encouraged in completing our project report successfully.

We are grateful to **Dr. C. Bagath Basha** who is our **Head of the Department, CSE** for his amiable ingenious and adept suggestions and pioneering guidance during the project report.

We express our gratitude and thanks to the Project Coordinator **Mr. K. Srinivasa Rao, Professor** of our department for his contribution for making it success with in the given time duration.

We express our deep sense of gratitude and thanks to **Internal Guide, Dr. C. Bagath Basha, Head of the Department,** for his guidance during the project report.

We are also very thankful to our **Management, Staff Members** and all **Our Friends** for their valuable suggestions and timely guidance without which we would not have been completed it.

By

V. SAI BHARATH	**(20RA1A0566)**
P. SAI RAM	**(20RA1A0572)**
P. SAI ROHITH	**(20RA1A0568)**

Vision of the Institute

To emerge as a premier institute for high quality professional graduates who can contribute to economic and social developments of the Nation.

Mission of the Institute

Mission	Statement
IM_1	To have holistic approach in curriculum and pedagogy through industry interface to meet the needs of Global Competency.
IM_2	To develop students with knowledge, attitude, employability skills, entrepreneurship, research potential and professionally Ethical citizens.
IM_3	To contribute to advancement of Engineering & Technology that would help to satisfy the societal needs.
IM_4	To preserve, promote cultural heritage, humanistic values and Spiritual values thus helping in peace and harmony in the society.

Vision of the Department

To Provide Quality Education in Computer Science for the innovative professionals to work for the development of the nation.

Mission of the Department

Mission	Statement
DM1	Laying the path for rich skills in Computer Science through the basic knowledge of mathematics and fundamentals of engineering
DM2	Provide latest tools and technology to the students as a part of learning Infrastructure
DM3	Training the students towards employability and entrepreneurship to meet the societal needs.
DM4	Grooming the students with professional and social ethics

Program Educational Objectives (PEOs)

PEO'S	Statement
PEO1	The graduates of Computer Science and Engineering will have successful career in technology.
PEO2	The graduates of the program will have solid technical and professional foundation to continue higher studies.
PEO3	The graduate of the program will have skills to develop products, offer services and innovation.
PEO4	The graduates of the program will have fundamental awareness of industry process, tools and technologies.

Program Outcomes

PO1	**Engineering Knowledge:** Apply the knowledge of mathematics, science, Engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	**Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	**The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	**Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental context, and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9	Individual and team network: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, being able to comprehend and write effective reports and design documentation, make Effective presentations, and give and receive clear instructions.
PO11	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work,as a member and leader in a team, to manage projects and in multidisciplinary environment.
PO12	Life-Long learning: Recognize the need for, and have the preparation and able to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES

PSO1	Foundation of mathematical concepts: To use mathematical methodologies to crack problem using suitable mathematical analysis, data structure and suitable algorithm.
PSO2	Foundation of Computer Science: The ability to interpret the fundamental concepts and methodology of computer systems. Students can understand the functionality of hardware and software aspects of computer systems.
PSO3	Foundation of Software development: The ability to grasp the software development lifecycle and methodologies of software systems. Possess competent skills and knowledge of software design process.

ABSTRACT

Certificate forgery has been a persistent issue across various industries, causing concerns in education, professional certifications, and legal documentation. In the past, verifying certificates was a manual and time-consuming process, involving physical examination of paper-based documents and cross-referencing with centralized databases or authorities. Unfortunately, this approach lacked transparency, was slow, and provided opportunities for fraudulent activities. Moreover, the vulnerability of traditional paper certificates to tampering and counterfeiting raised doubts about their authenticity and reliability. Thankfully, recent technological advancements have opened up a promising solution to combat certificate forgery by utilizing blockchain technology for certificate verification. Therefore, this project proposes an innovative approach to tackle this problem head-on. By harnessing the capabilities of blockchain, this work aims to create a robust and tamper-resistant certificate verification platform. Blockchain technology offers a decentralized and immutable way of storing and managing data, making it an ideal candidate to revolutionize certificate verification. With this system, the entire process becomes more efficient and secure. Each certificate issuance is recorded in a tamper-proof manner, complete with a timestamp, which virtually eliminates the possibility of altering or deleting information. The decentralized nature of blockchain removes the need for reliance on a central authority, reducing the risk of data manipulation and fostering trust in the verification process. This means that certificates can be verified without the involvement of a single controlling entity, making the system more reliable and transparent. In addition, the proposed system addresses the challenges associated with certificate forgery and offers a secure, efficient, and trustworthy solution. By leveraging blockchain technology, this proposed system can revolutionize current practices and ensure that certificates hold their true value and authenticity, thus maintaining the integrity of various industries plagued by this long-standing problem.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Overview

A blockchain-based secure and efficient validation system for digital certificates represents a groundbreaking solution to address the challenges associated with certificate verification, authentication, and fraud prevention. This system leverages the inherent characteristics of blockchain technology, such as immutability, transparency, and decentralization, to revolutionize the way digital certificates are managed and verified.

At its core, this system operates by storing digital certificate data on a blockchain ledger, ensuring that once a certificate is issued, it cannot be tampered with or altered. Each certificate is represented as a unique digital token on the blockchain, making it easy to verify its authenticity and origin. Furthermore, the decentralized nature of blockchain ensures that there is no single point of failure, reducing the risk of data breaches and unauthorized access.

Efficiency is a key feature of this system. Traditional methods of certificate validation can be time-consuming and susceptible to errors. With blockchain, verification becomes instantaneous and highly reliable. Institutions, employers, and individuals can quickly confirm the legitimacy of a certificate by accessing the blockchain, eliminating the need for time-consuming manual checks.

Security is paramount in this system. The cryptographic nature of blockchain technology ensures that data is encrypted and protected against unauthorized access. This safeguards sensitive certificate information from potential hackers and fraudulent activity. Moreover, the transparency of blockchain allows stakeholders to track the entire history of a certificate, from issuance to validation, enhancing trust and accountability.

Additionally, this system promotes interoperability and reduces reliance on central authorities. Certificates from various sources and institutions can coexist on the same blockchain, facilitating cross-validation and making it easier for individuals to showcase their qualifications and achievements across different platforms and industries.

So, a blockchain-based secure and efficient validation system for digital certificates represents a paradigm shift in the way we manage and authenticate credentials. By harnessing the power of blockchain, this system offers a robust, tamper-proof, and transparent solution that enhances security, efficiency, and trust in the digital certificate ecosystem. It has the potential to streamline verification processes, reduce fraud, and empower individuals to take control of their digital identities and qualifications.

1.2 Research Motivation

The research motivation for developing a blockchain-based secure and efficient validation system for digital certificates stems from several critical challenges in the current landscape of credential verification and certification. These challenges collectively underscore the pressing need for innovative solutions in this domain.

Firstly, the prevalence of credential fraud and the difficulty in verifying the authenticity of digital certificates have become significant concerns. With the advent of sophisticated forgery techniques, fake diplomas and certificates have become increasingly prevalent, posing risks to employers, educational institutions, and individuals. The traditional methods of verifying these credentials are often time-consuming, error-prone, and lack transparency, making it essential to explore alternative, more secure means of authentication.

Secondly, the reliance on centralized authorities for credential verification and data storage is a prominent issue. Centralized databases and authorities are susceptible to data breaches, compromising the confidentiality and security of sensitive information. Recent instances of high-profile data breaches in educational institutions and certification bodies have underscored the vulnerability of these systems. Therefore, a decentralized approach that reduces reliance on single points of failure is imperative.

Moreover, the existing verification processes for digital certificates are often inefficient and labor-intensive. These processes entail manual checks and communication with various issuing institutions, leading to delays and administrative burdens. A blockchain-based system has the potential to streamline these processes, providing instantaneous verification and significantly reducing the associated time and costs.

1.3 Problem Statement

The problem statement for developing a blockchain-based secure and efficient validation system for digital certificates centers on the existing challenges and shortcomings in the domain of digital credential verification. These challenges collectively create a pressing need for a transformative solution:

One of the primary issues is the pervasive problem of credential fraud. In today's digital age, the ease with which fraudulent certificates and diplomas can be created and distributed online has led to a proliferation of counterfeit credentials. This poses serious risks to employers, educational institutions, and individuals who rely on the authenticity of these certificates. The lack of a reliable and tamper-proof system for verifying digital certificates creates an environment ripe for exploitation and deception.

Traditional methods of certificate verification are plagued by inefficiencies. These methods often involve manual verification processes that can be slow, error-prone, and resource-intensive. Institutions and employers are burdened with the responsibility of confirming the legitimacy of certificates through laborious and time-consuming checks, including contacting issuing authorities. This not only hampers operational efficiency but also introduces the potential for human error and delays in decision-making processes.

Furthermore, the existing systems for managing digital certificates rely heavily on centralized authorities and databases. These centralized systems are vulnerable to data breaches, as they concentrate valuable and sensitive information in one location. Recent high-profile breaches of educational institutions and certification bodies have underscored the urgent need for a more secure and resilient approach to data storage and credential verification.

In light of these challenges, there is a clear need for a novel and comprehensive solution that can effectively combat credential fraud, streamline verification processes, and enhance security. A blockchain-based validation system has the potential to address these issues by providing an immutable, decentralized, and efficient mechanism for managing and verifying digital certificates. Thus, the problem statement revolves around developing and implementing such a system to revolutionize the digital certificate ecosystem and address the aforementioned challenges.

1.4 Applications

The blockchain-based secure and efficient validation system for digital certificates has a wide range of applications across various sectors and industries due to its transformative potential in enhancing the security and reliability of digital credentials. Here are detailed applications of this system:

Education Sector: In the education sector, this system can be used to secure academic certificates, diplomas, and transcripts. Educational institutions can issue digital certificates to students, which are stored on the blockchain. Employers and other educational institutions can easily and instantly verify the authenticity of these certificates, simplifying admissions and recruitment processes. This application can also help in combating the proliferation of fake degrees and transcripts.

Employment and Human Resources: Employers and HR departments can rely on the blockchain system to efficiently verify the qualifications and certifications of job applicants. This reduces the risk of hiring individuals with fraudulent credentials and streamlines the onboarding process. Additionally, professionals can maintain a verified record of their certifications, making it easier to switch jobs or advance in their careers.

Government and Public Records: Governments can use blockchain for securely storing and verifying vital records such as birth certificates, marriage licenses, and professional licenses. This can improve the efficiency of government services, reduce identity fraud, and enhance citizen trust in government processes.

Healthcare Industry: In healthcare, medical professionals can store their licenses, credentials, and training certificates on the blockchain. Hospitals and clinics can verify the qualifications of doctors and nurses, ensuring patient safety. Patients can also have confidence that their healthcare providers are properly credentialed.

Supply Chain Management: In industries where compliance and certification are crucial, such as food and pharmaceuticals, blockchain can be used to track and verify the authenticity of certificates related to product quality, safety, and origin. This ensures that products meet regulatory requirements and that consumers are receiving genuine, safe goods.

Online Learning and MOOCs: Online learning platforms and Massive Open Online Courses (MOOCs) can issue blockchain-based certificates of completion or achievement. Learners can easily share and validate their credentials, which adds value to their educational achievements and encourages lifelong learning.

Immigration and Visa Processes: Governments can use blockchain to verify the educational and professional qualifications of immigrants applying for visas or work permits. This streamlines the immigration process and ensures that individuals meet the required standards.

CHAPTER 2

LITERATURE SURVEY

The proliferation of industrial IoT applications and networking services has facilitated a tremendous increase in the number of connected devices. These application devices can capture real-time industrial data with a dedicated sensor unit [1]. Industrial advancement and technological guidance are behind this shift in how systems interact with physical and logical things. A centralized architecture is used to communicate real-time industrial data and evaluate the critical components of IoT, including identity management [2]. A single failure point is feasible due to this common technique [3]. A significant issue with the Internet of Things (IoT) is the difficulty in maintaining and managing many connected devices [4]. A system of networks can talk interactively through adaptive self-configuration. IoT applications can be commercialized over the 6G network. A fundamental component of the IoT, the wireless sensor network (WSN) gathers and transmits physical data using various heterogeneous models [5].

Data security is a major concern of IoT systems because they are built by connecting many IoT devices [6]. Data generated by these devices are stored in the cloud and transmitted across various networks. A cyber-attack on a smart healthcare system can substantially impact the system's ability to produce and supply electricity. In addition to financial and other types of damage, cyber-attacks on smart healthcare can cause operational failures, power outages, the theft of critical data, and complete security breaches [7]. Cyber experts face difficulties keeping tabs on everything that passes via a smart grid and recognizing potential threats and attacks. Even though machine learning has become an essential part of cybersecurity, the problem is that this field requires distinct approaches and theoretical viewpoints to handle the enormous volume of data generated and transported across numerous networks in a smart grid [8]. The attacks and threats that could be launched against this proof-of-concept environment are being determined using threat modeling. Several potential threats have been tested, including detection, tampering, repudiation, information leakage, denial of service (DoS), and extended privilege (EoP). Each of the risks and the security elements associated with them are addressed using STRIDE. STRIDE is a typical threat modeling technique for finding and classifying attack vectors [9]. Using the well-known industrial framework MITRE ATTCK, researchers can detect threats disguised as tactics, techniques, and procedures (TTP) [10].

Based on the above, blockchain technology could be one of the main solutions for IoT security issues [11]. A blockchain provides a decentralized system using a consensus mechanism and smart contracts [12]. Smart contracts are the protocols that trigger the blockchain to act according to a particular activity or situation [13]. Blockchains can be categorized into three classes: private, public, and hybrid public blockchain technology. The main feature of a blockchain is to provide security and only keep records and transactions within a single organization. A public blockchain provides access to the public using a public API. Moreover, such a model interacts with external networks such as gateway networks or cloud outsourcing. A hybrid blockchain is also called a consortium blockchain, which provides features of both a private and public blockchain.

Blockchain technology can be used to build trust and monitor node activity in IoT networks. It is challenging to integrate a blockchain into IoT applications due to its high power consumption and job outsourcing [14]. Several blockchain-based Internet of Things (IoT) applications have recently been created to address these concerns. These blocks can be used to delete old transactions and blocks from the blockchain without jeopardizing security. Pan et al. [15] created an IoT resource management prototype using blockchain technology and smart contracts to securely record all IoT transactions [15]. Deploying smart contracts involves evaluating the source code, bytes of code, and execution histories. This is how we test our computer traffic analysis deployment scenario. Ali et al. [16] investigated blockchain technology and smart contract applications in cloud storage. Tam et al. utilize a pay-as-you-go car business model. This technology's strengths are traceability and tamper-proof characteristics. Ali et al. [17] created a blockchain-based publisher–subscriber model. They designed their solution to ensure data integrity in real-time IoT processing by balancing computational resources and workload. Liu et al. delegated computationally intensive POW mining tasks to nearby edge servers in blockchain-enabled mobile IoT systems [18]. Chen et al. conducted additional research. Securing biometric data for patient authentication is a common issue. In particular, finger vein biometric data has been studied extensively. A strong verification mechanism with high levels of reliability, privacy, and security is required to better secure these data. Also, biometric data are difficult to replace, and any leakage of biometric data exposes users to serious threats, such as replay attacks employing stolen biometric data. This research offers a unique verification secure framework based on triplex blockchain-based particle swarm optimization (PSO)-advanced encryption standard (AES) approaches in medical systems for patient authentication. The discussion has three stages. First presented is a new hybrid model pattern based on RFID and finger vein biometrics to boost randomness. It proposes a new merge method that combines RFID and finger vein characteristics in a random pattern. Second, the suggested verification safe framework is based on the CIA standard for telemedicine authentication using AES encryption, blockchain technology, and PSO in steganography [19]. Finally, the proposed secure verification architecture was validated and evaluated [20]. The combination of WSN functional activities with 6G network topologies allows us to test a wide range of IoT application deployment models. Many IoT devices collect data using IPV6 across low-power wireless personal area networks and wearables (6LoWPAN) [21,22]. We were able to keep user data confidential with the help of AKA [23]. Companies that use public cloud services and large-scale data storage systems have long prioritized client data protection [24].

CHAPTER 3

EXISTING SYSTEM

3.1 Hyperledger Fabric

Hyperledger Fabric is designed for use in enterprise-level applications, and it is characterized by its modular architecture, permissioned network, and smart contract functionality, known as “chaincode”.

- The platform provides a high degree of security, privacy, and scalability, and it supports the development of custom blockchain solutions for various use cases across industries such as finance, supply chain, and healthcare.
- Hyperledger Fabric operates as a network of nodes, where each node performs a specific function, such as validating transactions, maintaining the ledger, and executing chaincode.
- Transactions are validated and ordered by a consensus mechanism, which ensures the integrity and consistency of the ledger.

3.1.2 Working of Hyperledger Fabric

Components:

Hyperledger fabric is an enterprise-level permission blockchain network. It is made up of various unique organizations or members that interact with each other to serve a specific purpose. For example, these organizations can be a bank, financial institution, or a supply chain network. Each organization is identified and they have a fabric certificate authority. These organizations are called members.

Each member of the fabric can set up one or more authorized peers to participate in the network using the fabric certificate authority. All of these peers must be authorized properly.

There is a client-side application connected to the network written with the software development kit (SDK) of any particular programming language.

Workflow:

For each and every transaction in the fabric, the following steps are followed

Creation of the proposal: Imagine a deal between a smartphone manufacturer company and a smartphone dealership. The transaction begins when a member organization proposes or invokes a transaction request with the help of the client application or portal. Then the client application sends the proposal to peers in each organization for endorsement.

Endorsement of the transaction: After the proposal reaches the endorser peers (peers in each organization for endorsement of a proposal) the peer checks the fabric certificate authority of the requesting member and other details that are needed no authenticate the transaction. Then it executes the chain code (a piece of code that is written in one of the supported languages such as Go or Java) and returns a response. This response indicates the approval or rejection of the following transaction. The response is carried out to the client.

Submission to ordering service: After receiving the endorsement output, the approved transactions are sent to the ordering service by the client-side application. The peer responsible for the ordering service includes the transaction into a specific block and sends it to the peer nodes of different members of the network.

Updating the ledger: After receiving this block the peer nodes of such organizations update their local ledger with this block. Hence the new transactions are now committed. In general, all farmers will sell their milk to 3rd part brokers or IDA staff members, and they will record each farmer milk delivery in a manual inventory report or in computer excel or centralized server. All framers may be no or less educated so brokers may alter farmer milk deliver records and make less payment to farmers and steal money.

3.2 Limitations of hyper ledger Fabric Security

Hyperledger Fabric is a permissioned blockchain framework that is designed to provide enterprise-grade security and privacy features. However, like any technology, it has its own set of disadvantages and limitations when it comes to security. Here are some of the disadvantages of Hyperledger Fabric security:

Complexity: Hyperledger Fabric is a complex system that requires a lot of expertise to set up and maintain. This complexity can make it difficult to ensure proper security measures are in place and can increase the likelihood of human error, which can result in security vulnerabilities.

Limited decentralization: Hyperledger Fabric's permissioned nature limits the number of nodes that can participate in the network. This can limit the level of decentralization and increase the risk of a single point of failure. In addition, the permissioned nature of the network can also lead to centralization of power among the permissioned nodes.

Lack of anonymity: Hyperledger Fabric's permissioned nature means that all participants in the network are identified and authenticated, which can limit anonymity. While this may be desirable in some use cases, it can be a disadvantage in others where anonymity is important.

Smart contract vulnerabilities: Hyperledger Fabric relies on smart contracts for executing transactions, and like any code, these contracts can contain vulnerabilities that can be exploited by attackers. This can lead to loss of funds or data leaks.

Limited scalability: Hyperledger Fabric's scalability is limited by its consensus mechanism, which requires all nodes to validate each transaction. This can lead to network congestion and slow transaction processing times, which can limit its use in large-scale applications

Overall, while Hyperledger Fabric is designed to provide enterprise-grade security features, it still has its own set of disadvantages and limitations. Organizations should carefully evaluate these disadvantages and consider whether Hyperledger Fabric is the best choice for their specific use case.

CHAPTER 4

PROPOSED SYSTEM

4.1Overview

Figure 4.1 shows the proposed system model. The detailed operation illustrated as follows:

Step 1. Input Certificates: This likely represents the starting point of the research. Input certificates refer to the digital certificates that need to be validated or verified. These certificates could be various types such as educational diplomas, professional certifications, or any other forms of digital credentials.

Step 2. Certificate Issuance and Verification System: This component suggests the existence of a system responsible for issuing and verifying digital certificates. In the context of the research, it's important to examine how this system currently works and identify its strengths and weaknesses.

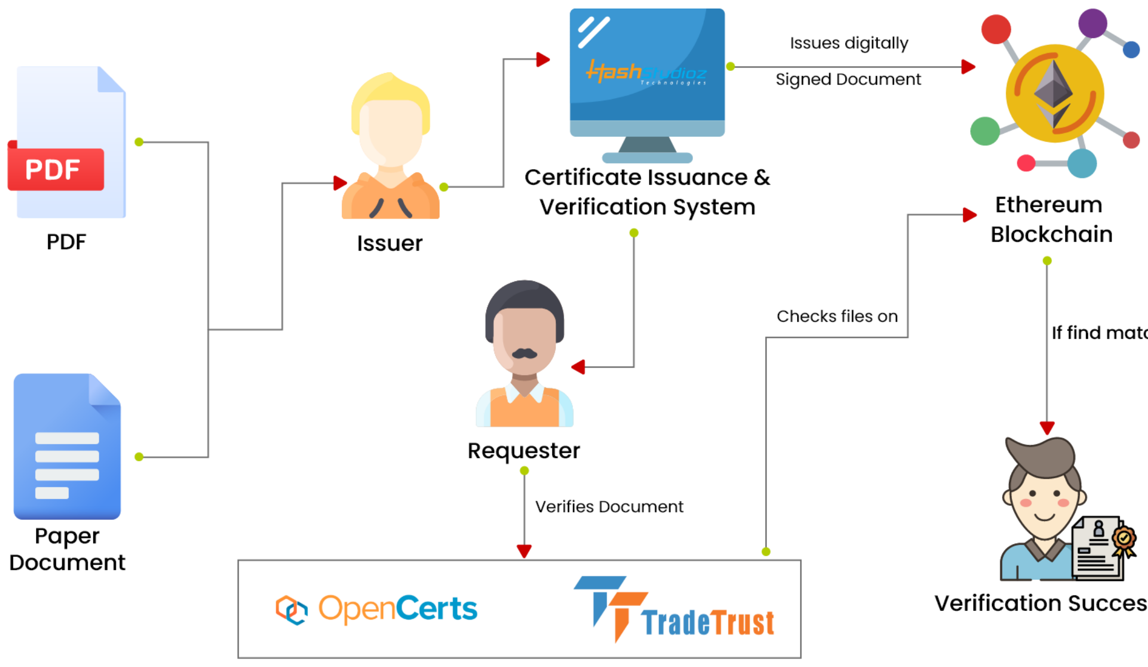


Figure 4.1. Proposed System model.

Step 3. Ethereum (Blockchain): Ethereum is mentioned as the blockchain technology that is likely being proposed or utilized in the research. Ethereum is a popular platform for developing blockchain-based applications. In this research, it's important to explain how Ethereum is used to enhance the security and efficiency of the certificate validation process.

Step 4. Requester: The "requester" is likely a user or entity that initiates the certificate validation process. In a blockchain-based system, this could be someone who wants to verify the authenticity of a digital certificate.

Step 5. Verification Results: This component indicates the outcome of the certificate validation process. It's essential to understand how the blockchain-based system generates and communicates verification results, including whether a certificate is valid or not.

4.2 Ethereum

Ethereum is a decentralized blockchain platform that allows developers to build decentralized applications (dApps) and execute smart contracts. It was launched in 2015 by Vitalik Buterin and quickly became one of the most popular blockchain platforms in the world, second only to Bitcoin in terms of market capitalization.

Ethereum's main innovation is the ability to create smart contracts, which are self-executing contracts with the terms of the agreement between buyer and seller being directly written into lines of code. These smart contracts are executed on the Ethereum Virtual Machine (EVM), which is a decentralized, Turing-complete virtual machine that runs on the Ethereum network.

The Ethereum network also has its own cryptocurrency called Ether (ETH), which is used to pay for transaction fees and computational services on the network. ETH is also used as a store of value and traded on cryptocurrency exchanges.

4.2.1 Advantages of Ethereum

Ethereum provides several advantages over other blockchain platforms and traditional systems. Here are some of the main advantages of Ethereum:

Smart Contracts: Ethereum's main innovation is the ability to create smart contracts, which are self-executing contracts with the terms of the agreement between buyer and seller being directly written into lines of code. This allows for secure and automated execution of complex agreements without the need for intermediaries or third parties.

Decentralization: Ethereum is a decentralized platform, which means that it is not controlled by any single entity or organization. This provides a level of trust and transparency, as there is no single point of failure or vulnerability.

Interoperability: Ethereum's blockchain is open-source and allows for interoperability with other blockchain platforms, making it easier to integrate with existing systems and applications.

Programmable: Ethereum's blockchain is programmable, which means that developers can create custom applications and smart contracts that meet their specific needs. This allows for more flexibility and customization than traditional systems.

Security: Ethereum's blockchain is secured through cryptographic algorithms and consensus mechanisms, making it resistant to hacking and fraud. Additionally, smart contracts on the platform are auditable and transparent, which helps to reduce the risk of fraud and corruption.

Tokenization: Ethereum enables the creation and exchange of tokens, which can represent assets, securities, or other digital assets. This makes it possible to create new business models and revenue streams that were previously not possible.

Overall, Ethereum provides a powerful and flexible platform for developers to build decentralized applications and execute complex smart contracts in a secure, transparent, and decentralized manner.

4.3 Blockchain

Blockchain is a decentralized, digital ledger technology that is used to record and store data in a secure and transparent manner. It is a distributed ledger, meaning that it is maintained by a network of computers, rather than being controlled by a single entity. Each block in the chain contains a set of transactions, and once a block is added to the chain, it cannot be altered or deleted. This makes blockchain an immutable and tamper-resistant technology that is particularly well-suited for storing and transmitting sensitive data.

Blockchain technology is perhaps best known for its use in cryptocurrencies like Bitcoin and Ethereum, but it has a wide range of other potential applications as well. These include supply chain management, identity verification, voting systems, and more. The decentralized nature of blockchain means that it has the potential to disrupt a variety of industries and business models by enabling trust and transparency in transactions and data exchange.

Concepts

There are several key concepts that are important to understand when it comes to blockchain technology:

Decentralization: Blockchain is a decentralized technology, meaning that it is not controlled by any single entity, but rather maintained by a network of participants. This increases transparency, security, and resilience.

Distributed ledger: Blockchain technology uses a distributed ledger to record and store data. Each block in the chain contains a set of transactions, and once a block is added to the chain, it cannot be altered or deleted.

Cryptography: Blockchain technology uses advanced cryptographic algorithms to secure transactions and data exchange, making it highly resistant to hacking and cyber attacks.

Consensus mechanism: In a blockchain network, participants must agree on the validity of transactions before they are recorded on the blockchain. Different blockchain networks use different consensus mechanisms to achieve this, such as Proof of Work or Proof of Stake.

Smart contracts: Smart contracts are self-executing contracts with the terms of the agreement directly written into code. They can be used to automate complex transactions and ensure that all parties involved in a transaction adhere to the terms of the contract.

Tokenization: Blockchain technology enables the creation of digital tokens that can be used to represent a variety of assets, such as currencies, commodities, or even real estate.

4.3.1 Applications of Blockchain

Blockchain technology has a wide range of potential applications across various industries. Some examples of how blockchain is currently being used, or has the potential to be used, include:

Cryptocurrencies: Blockchain technology is the foundation of cryptocurrencies like Bitcoin and Ethereum, which use blockchain to enable peer-to-peer transactions without the need for a centralized intermediary.

Supply chain management: Blockchain technology can be used to create transparent and secure supply chain systems, allowing participants to track and verify the origin and authenticity of products.

Identity verification: Blockchain technology can be used to create secure and tamper-proof digital identity systems, allowing individuals to prove their identity without the need for a centralized authority.

Voting systems: Blockchain technology can be used to create secure and transparent voting systems, ensuring the accuracy and legitimacy of election results.

Healthcare: Blockchain technology can be used to create secure and transparent healthcare systems, enabling secure sharing of patient data and facilitating drug traceability.

Finance: Blockchain technology can be used to create more efficient and secure financial systems, allowing for faster and cheaper transactions while reducing the risk of fraud and corruption.

Real estate: Blockchain technology can be used to create more transparent and secure real estate transactions, allowing for faster and more efficient transfer of ownership.

These are just a few examples of how blockchain technology is being used, and there are many other potential applications that are currently being explored.

4.3.2 Advantages of Block Chain

here are several advantages to using blockchain technology:

Decentralization: The decentralized nature of blockchain technology means that it is not controlled by any single entity, which increases transparency and security. This also means that there is no need for a centralized intermediary, reducing the risk of fraud or corruption.

Immutability: Once data has been recorded on a blockchain, it cannot be altered or deleted, which ensures that it is tamper-proof and provides a high degree of data integrity.

Security: Blockchain technology uses cryptographic algorithms to secure transactions and data exchange, making it highly resistant to hacking and cyber attacks.

Transparency: Blockchain technology provides a high degree of transparency, as all participants in the network have access to the same information, making it easier to verify and track transactions.

Efficiency: Blockchain technology can reduce the need for intermediaries in transactions, reducing the time and cost associated with processing and verifying transactions.

Trust: The security and transparency provided by blockchain technology can increase trust among participants in a network, leading to more efficient and secure transactions.

CHAPTER 5

UML DAIGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

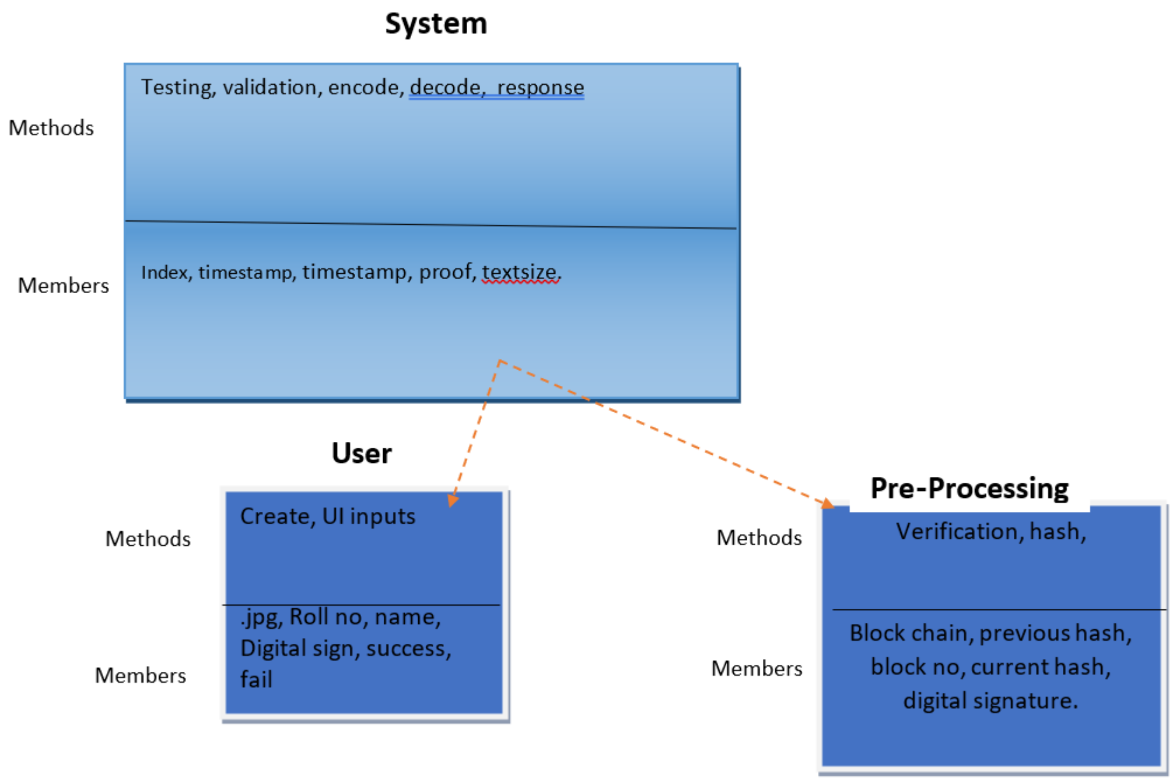
The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

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

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

CLASS DIAGRAM

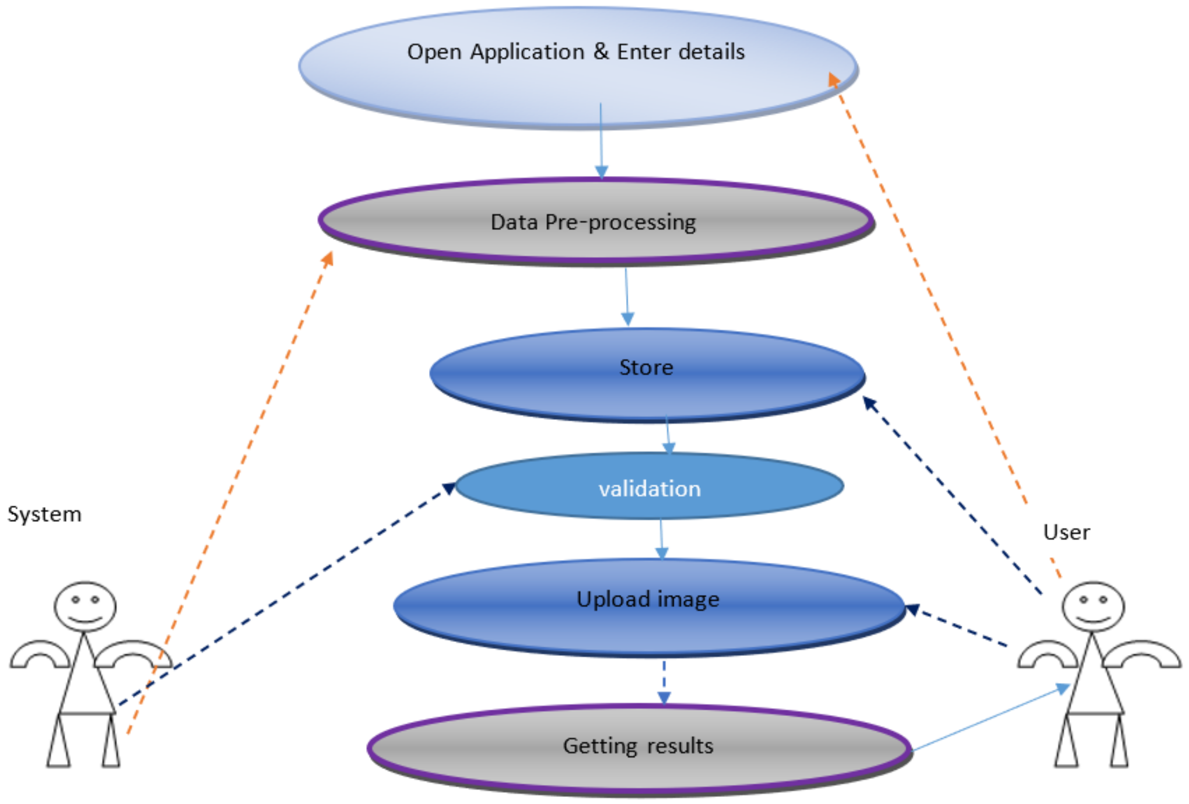
In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains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



System
Methods
Testing, validation, encode, decode, response
Members
Index, timestamp, timestamp, proof, textsize.
User
Methods
Create, UI inputs
Members
.jpg, Roll no, name, Digital sign, success, fail
Pre-Processing
Methods
Verification, hash,
Members
Block chain, previous hash, block no, current hash, digital signature.

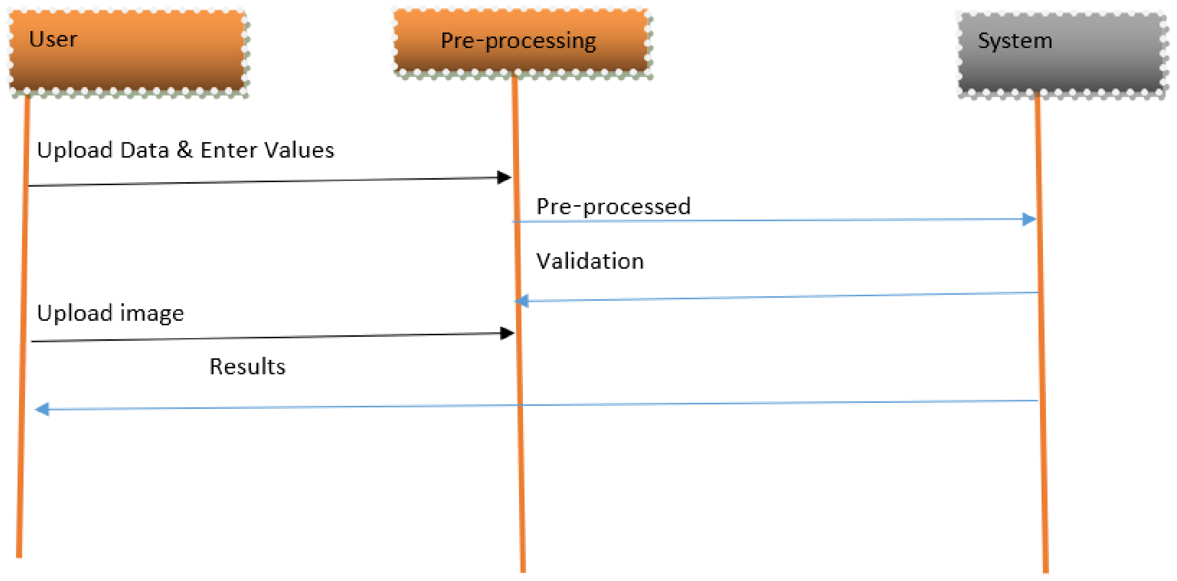
USE CASE DIAGRAM

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



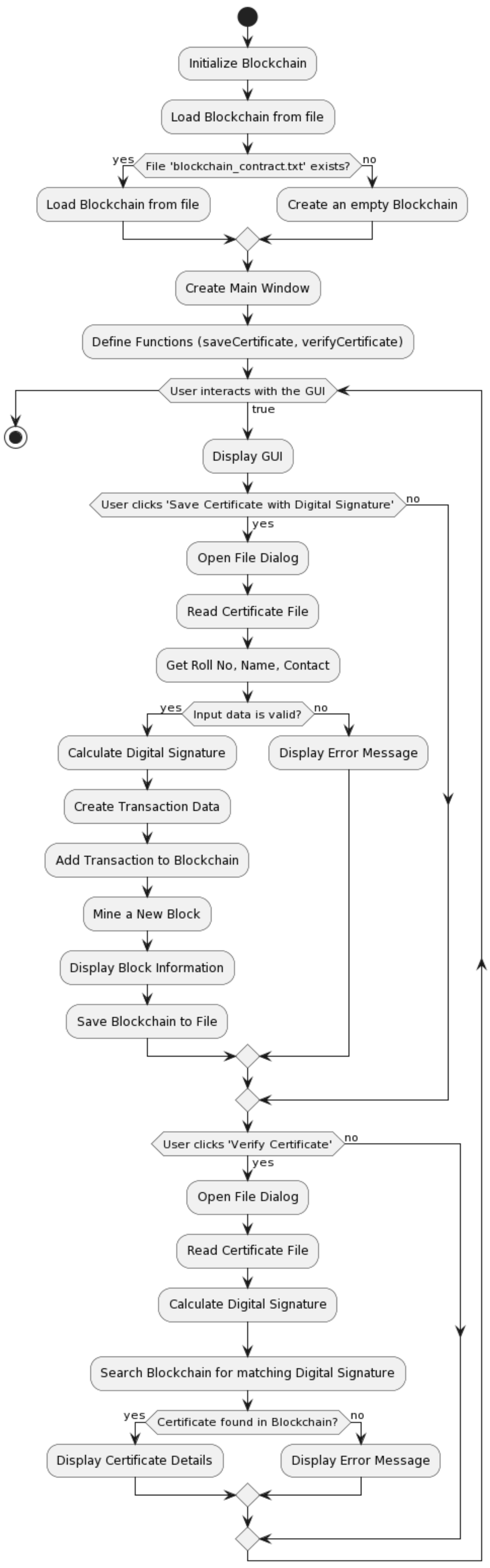
SEQUENCE DIAGRAM

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".



ACTIVITY DIAGRAM

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions.


Initialize Blockchain
Load Blockchain from file
File 'blockchain_contract.txt' exists?
yes
no
Load Blockchain from file
Create an empty Blockchain
Create Main Window
Define Functions (saveCertificate, verifyCertificate)
User interacts with the GUI
true
Display GUI
User clicks 'Save Certificate with Digital Signature'
no
yes
Open File Dialog
Read Certificate File
Get Roll No, Name, Contact
Input data is valid?
yes
no
Calculate Digital Signature
Display Error Message
Create Transaction Data
Add Transaction to Blockchain
Mine a New Block
Display Block Information
Save Blockchain to File
User clicks 'Verify Certificate'
no
yes
Open File Dialog
Read Certificate File
Calculate Digital Signature
Search Blockchain for matching Digital Signature
Certificate found in Blockchain?
yes
no
Display Certificate Details
Display Error Message

CHAPTER 6

SOFTWARE ENVIRONMENT

What is Python?

Below are some facts about Python.

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

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

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

Advantages of Python

Let's see how Python dominates over other languages.

1. Extensive Libraries

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

2. Extensible

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

3. Embeddable

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

4. Improved Productivity

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

5. IOT Opportunities

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

6. Simple and Easy

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

7. Readable

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

8. Object-Oriented

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

9. Free and Open-Source

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

10. Portable

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

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

3. Design Restrictions

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

4. Underdeveloped Database Access Layers

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

5. Simple

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

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

History of Python

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

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 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

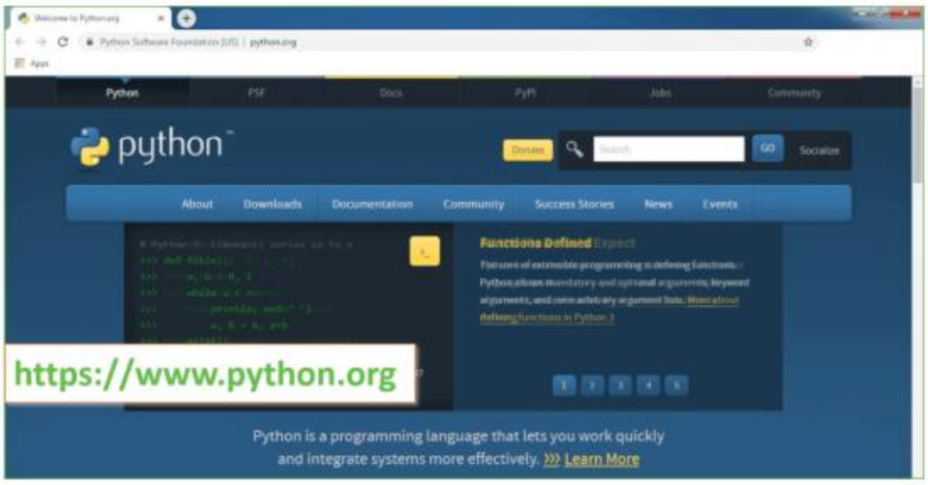
There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other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

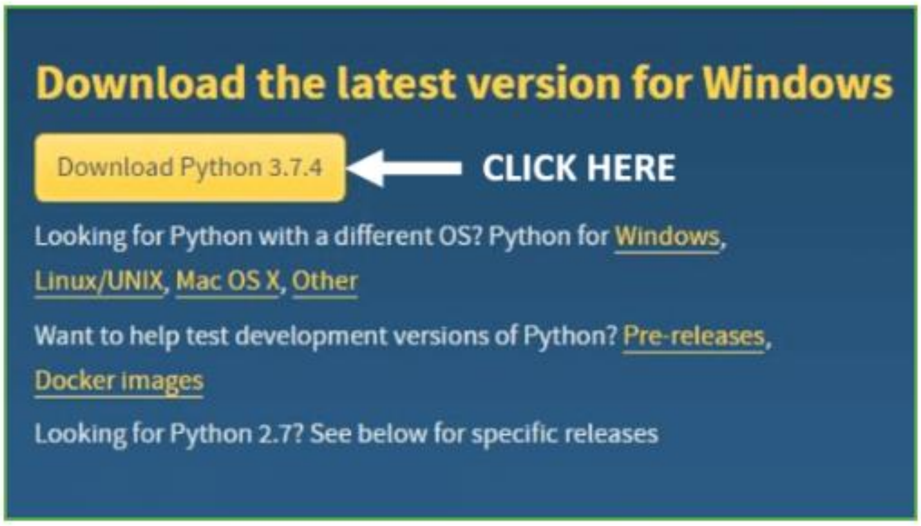
Download the Correct version into the system

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


https://www.python.org
python
About
Downloads
Documentation
Community
Success Stories
News
Events
Functions Defined
Python is a programming language that lets you work quickly
and integrate systems more effectively. >>> Learn More

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

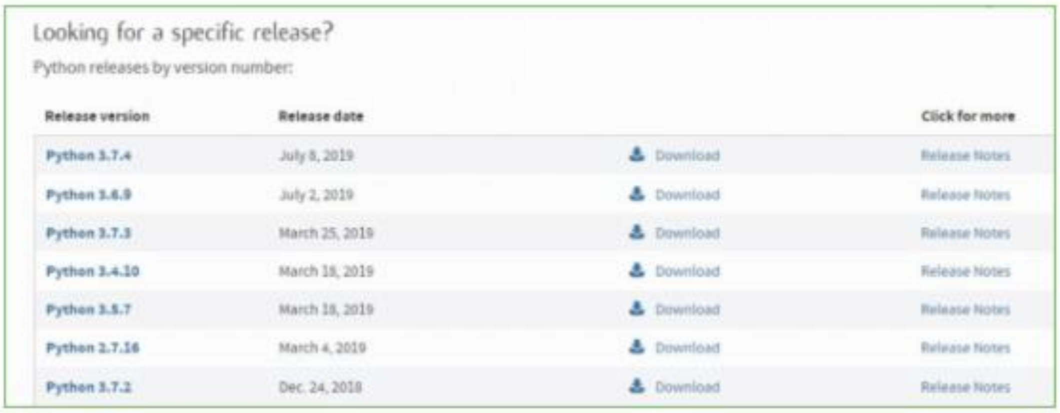
Download Python 3.7.4 ← CLICK HERE

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



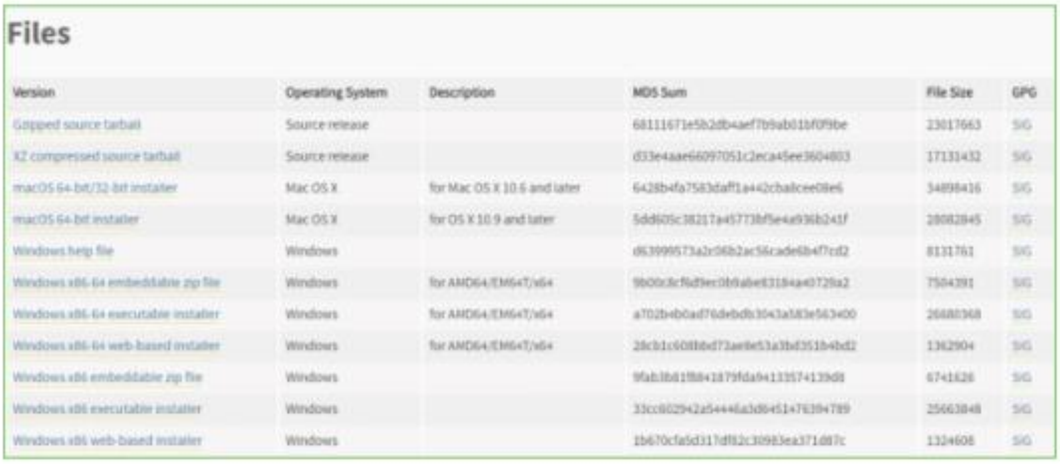
Looking for a specific release?

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 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

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

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



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01bf0f9be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aae66097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daff1a442cba8cee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b241f	28082845	SIG
Windows help file	Windows		d63999573a2c06b2ac56cade6b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b00c8cf6d9ec0b9abe83184a40729a2	7504391	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76debdb3043a583e563400	26680368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608bbd73ae8e53a3bd351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b81f8841879fda94133574139d8	6741626	SIG
Windows x86 executable installer	Windows		33cc602942a54446a3d6451476394789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	SIG

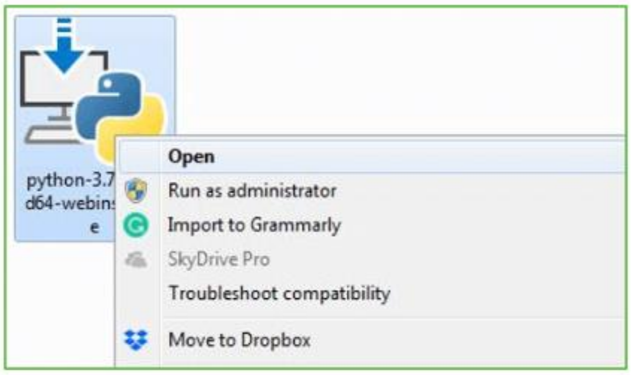
- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, 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

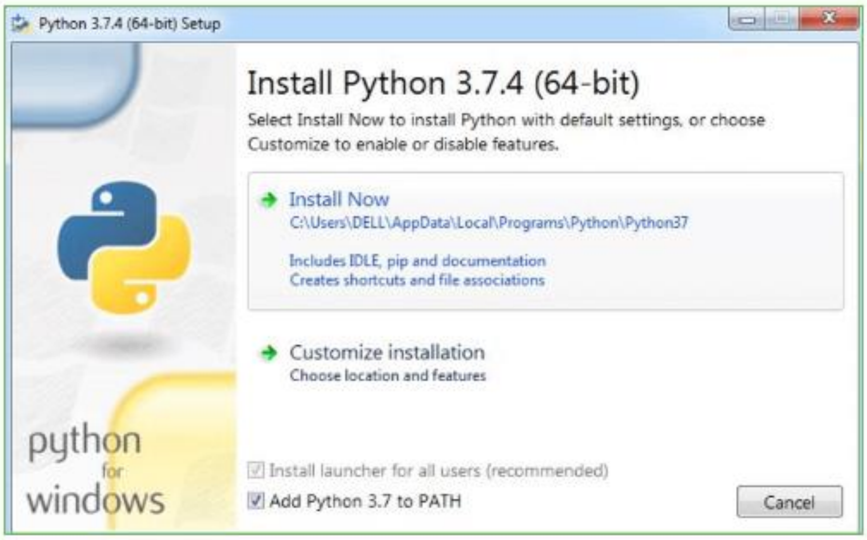
Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

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.



Python 3.7.4 (64-bit) Setup

Install Python 3.7.4 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

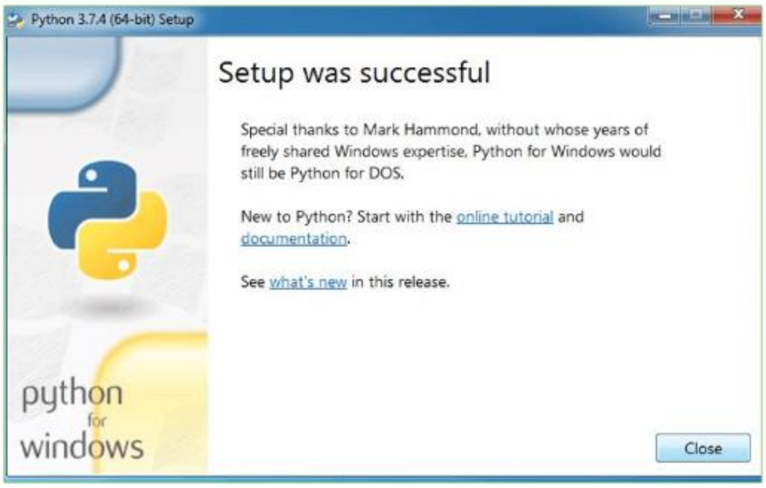
→ Install Now
C:\Users\DELL\AppData\Local\Programs\Python\Python37

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.7 to PATH

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


Python 3.7.4 (64-bit) Setup

Setup was successful

Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

New to Python? Start with the online tutorial and documentation.

See what's new in this release.

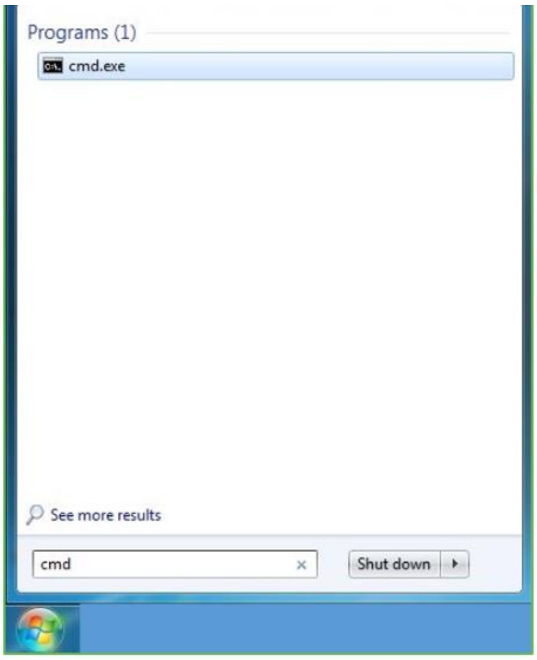
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

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

Verify the Python Installation

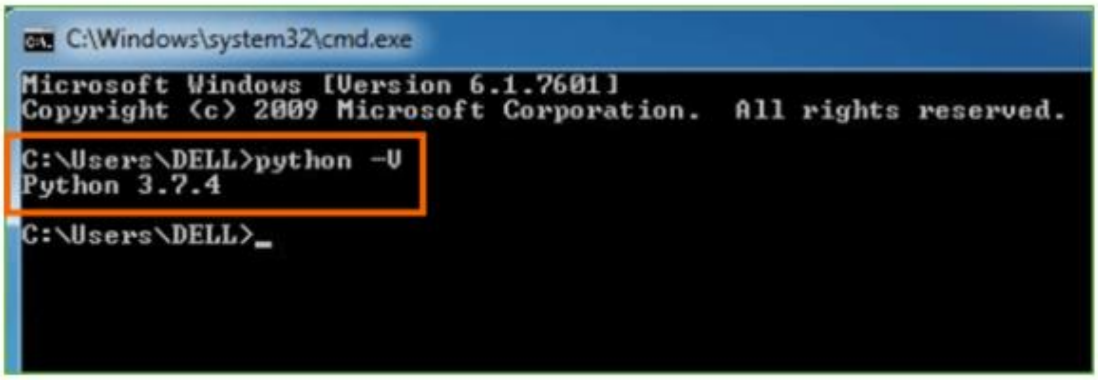
Step 1: Click on Start

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.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\DELL>python -V
Python 3.7.4

C:\Users\DELL>_
```

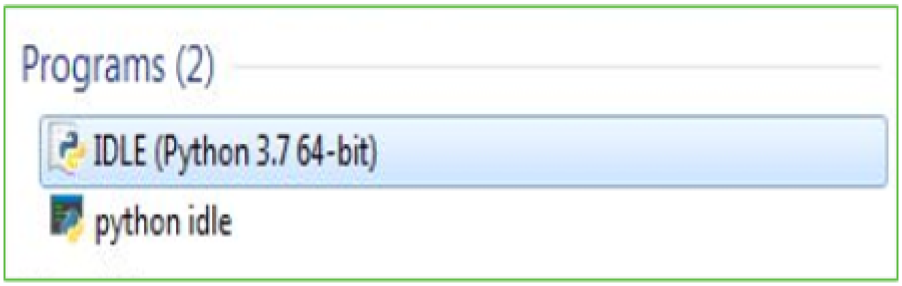
Step 5: You will get the answer as 3.7.4

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

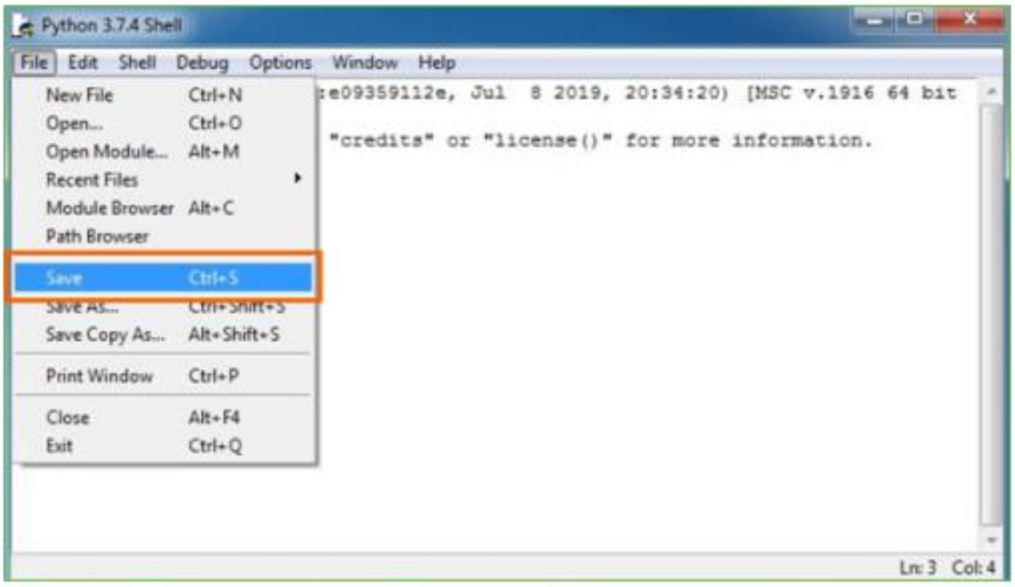
Step 1: Click on Start

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

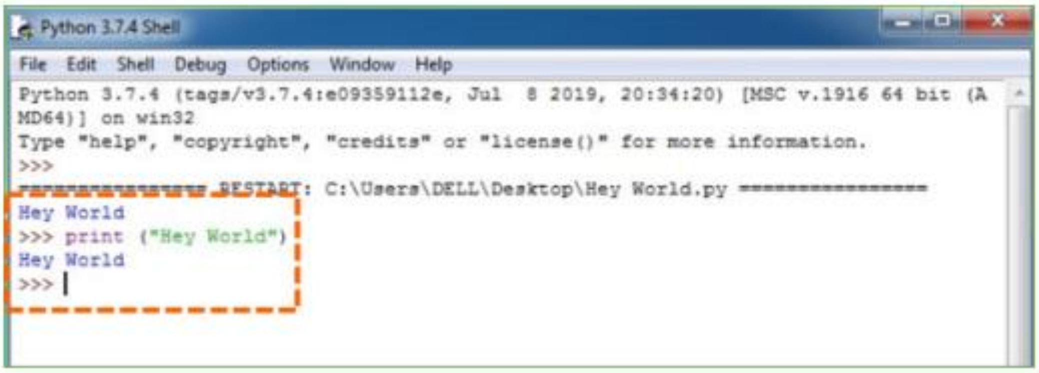


```
:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit

 "credits" or "license()" for more information.
```

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 ("Hey World") and Press Enter.



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (A
MD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================ RESTART: C:\Users\DELL\Desktop\Hey World.py ================
Hey World
>>> print ("Hey World")
Hey World
>>> |
```

You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python does not need semicolons at the end of the statements otherwise it won't work.

CHAPTER 7

SYSTEM REQUIREMENTS SPECIFICATIONS

Software Requirements

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regard to what the areas of strength and deficit are and how to tackle them.

- Python IDLE 3.7 version (or)
- Anaconda 3.7 (or)
- Jupiter (or)
- Google colab

Hardware Requirements

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

Operating system	:	Windows, Linux
Processor	:	minimum intel i3
Ram	:	minimum 4 GB
Hard disk	:	minimum 250GB

CHAPTER 8

FUNCTIONAL REQUIREMENTS

Output Design

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization
- Internal Outputs whose destination is within organization and they are the
- User's main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.

Output Definition

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

Input Design

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

Input Stages

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification
- Data control
- Data transmission
- Data validation
- Data correction

Input Types

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department’s communications to the system?
- Interactive, which are inputs entered during a dialogue.

Input Media

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction
- Storage and handling requirements
- Security
- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As

Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

Error Avoidance

At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

Error Detection

Even though every effort is made to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

Data Validation

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

User Interface Design

It is essential to consult the system users and discuss their needs while designing the user interface:

User Interface Systems Can Be Broadly Clasified As:

- User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
- Computer initiated interfaces

In the computer-initiated interfaces the computer guides the progress of the user/computer dialogue. Information is displayed and the user response of the computer takes action or displays further information.

User Initiated Intergfaces

User initiated interfaces fall into two approximate classes:

- Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
- Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms-oriented interface is chosen because it is the best choice.

Computer-Initiated Interfaces

The following computer – initiated interfaces were used:

- The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.
- Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

Error Message Design

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different inputs.

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 in the part of 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.

CHAPTER 9

SOURCE CODE

```
from tkinter import messagebox

from tkinter import *

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

from tkinter.filedialog import askopenfilename

from Block import *

from Blockchain import *

from hashlib import sha256

import os


main = Tk()

main.title("A Blockchain based Secure and Efficient Validation System for Digital Certificate")

main.geometry("1300x1200")


global filename


blockchain = Blockchain()

if os.path.exists('blockchain_contract.txt'):

    with open('blockchain_contract.txt', 'rb') as fileinput:

        blockchain = pickle.load(fileinput)

    fileinput.close()
```

```
def saveCertificate():
    global filename
    text.delete('1.0', END)
    filename = askopenfilename(initialdir = "certificate_templates")
    with open(filename,"rb") as f:
        bytes = f.read()
    f.close()
    roll_no = tf1.get()
    name = tf2.get()
    contact = tf3.get()
    if len(roll_no) > 0 and len(name) > 0 and len(contact) > 0:
        digital_signature = sha256(bytes).hexdigest();
        data = roll_no+"#"+name+"#"+contact+"#"+digital_signature
        blockchain.add_new_transaction(data)
        hash = blockchain.mine()
        b = blockchain.chain[len(blockchain.chain)-1]
        text.insert(END,"Blockchain Previous Hash : "+str(b.previous_hash)+"\nBlock No : 
"+str(b.index)+"\nCurrent Hash : "+str(b.hash)+"\n")
        text.insert(END,"Certificate Digital Signature : "+str(digital_signature)+"\n\n")
        blockchain.save_object(blockchain,'blockchain_contract.txt')
    else:
        text.insert(END,"Please enter Roll No")
```

```
def verifyCertificate():
    text.delete('1.0', END)
    filename = askopenfilename(initialdir = "certificate_templates")
    with open(filename,"rb") as f:
        bytes = f.read()
    f.close()
    digital_signature = sha256(bytes).hexdigest();
    flag = True
    for i in range(len(blockchain.chain)):
        if i > 0:
            b = blockchain.chain[i]
            data = b.transactions[0]
            arr = data.split("#")
            if arr[3] == digital_signature:
                text.insert(END,"Uploaded Certificate Validation Successfull\n")
                text.insert(END,"Details extracted from Blockchain after Validation\n\n")
                text.insert(END,"Roll No : "+arr[0]+"\n")
                text.insert(END,"Student Name : "+arr[1]+"\n")
                text.insert(END,"Contact No   : "+arr[2]+"\n")
                text.insert(END,"Digital Sign : "+arr[3]+"\n")
                flag = False
                break
    if flag:
        text.insert(END,"Verification failed or certificate modified")
```

```
font = ('BhaskerVille', 18, 'bold')

title = Label(main, text='A BLOCKCHAIN BASED SECURE AND EFFICIENT VALIDATION
SYSTEM FOR DIGITAL CERTIFICATES')

title.config(bg='bisque', fg='purple')

title.config(font=font)

title.config(height=3, width=85)

title.place(x=0,y=5)


font1 = ('times', 13, 'bold')


l1 = Label(main, text='Roll No :')

l1.config(font=font1)

l1.place(x=50,y=100)


tf1 = Entry(main,width=20)

tf1.config(font=font1)

tf1.place(x=180,y=100)


l2 = Label(main, text='Student Name :')

l2.config(font=font1)

l2.place(x=50,y=150)

tf2 = Entry(main,width=20)
```

```
tf2.config(font=font1)
tf2.place(x=180,y=150)
l3 = Label(main, text='Contact No :')
l3.config(font=font1)
l3.place(x=50,y=200)
tf3 = Entry(main,width=20)
tf3.config(font=font1)
tf3.place(x=180,y=200)
saveButton = Button(main, text="Save Certificate with Digital Signature", command=saveCertificate)
saveButton.place(x=50,y=250)
saveButton.config(font=font1)
verifyButton = Button(main, text="Verify Certificate", command=verifyCertificate)
verifyButton.place(x=420,y=250)
verifyButton.config(font=font1)
font1 = ('times', 13, 'bold')
text=Text(main,height=15,width=120)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=300)
text.config(font=font1)
main.config(bg='bisque')
main.mainloop()
```

CHAPTER 10

RESULTS AND DISCUSSION

This project designs a larger application to prevent forgery through blockchain-powered certificate verification. It allows users to store certificates in a blockchain and later verify their authenticity by comparing digital signatures. Here is the detailed process of implementation:

— Importing necessary libraries:

Various modules from the Tkinter library are imported to create the GUI elements.

Block and Blockchain classes are imported from custom modules, indicating that this code is part of a larger project.

The hashlib library is imported for SHA-256 hashing.

The os module is imported for handling file operations.

— Creating the main window:

An instance of the Tkinter Tk class is created to create the main window.

The window title and dimensions are set.

— Initializing blockchain and loading data:

A Blockchain object is created, representing the blockchain where certificate data will be stored.

The code checks if a file named 'blockchain_contract.txt' exists and, if so, loads the blockchain from that file using pickle.

— Define saveCertificate() function:

This function is called when the "Store Certificate in Blockchain" button is clicked.

It reads certificate data from a file, calculates its digital signature (SHA-256 hash), and adds the transaction to the blockchain.

It then displays blockchain information and saves the blockchain to a file.

— Define verifyCertificate() function:

This function is called when the "Certificate Verification" button is clicked.

It reads certificate data from a file, calculates its digital signature (SHA-256 hash), and searches the blockchain for a matching digital signature to verify the certificate's authenticity.

It displays the result of the verification.

- Creating GUI elements:

Labels, entry fields, and buttons are created for user input and interaction. These elements are positioned using the place method.

A text widget is created to display output, and a scrollbar is added to it.

- Configuring GUI elements: Font styles and sizes are configured for various GUI elements to ensure consistency and readability.
- Running the main event loop: The main event loop is started to display the GUI and handle user interactions.

Results description

Figure 1 shows the initial state of the graphical user interface (GUI) when the application is launched. It includes the following elements:

- Title: "PREVENTING FORGERY WITH BLOCK-CHAIN POWERED CERTIFICATE VERIFICATION"
- Buttons: "Store Certificate in Blockchain" and "Certificate Verification"
- Entry fields for student details (Roll Number, Student Name, Contact Information)
- A text widget for displaying information and results.
- The main window with a specific size and background color (powder blue)

Figure 2 represents the GUI after a user has filled in the student's details. It includes the following changes from Figure 1:

- The entry fields for Roll Number, Student Name, and Contact Information are populated with user input.
- The user has entered the relevant information required for storing a certificate in the blockchain.

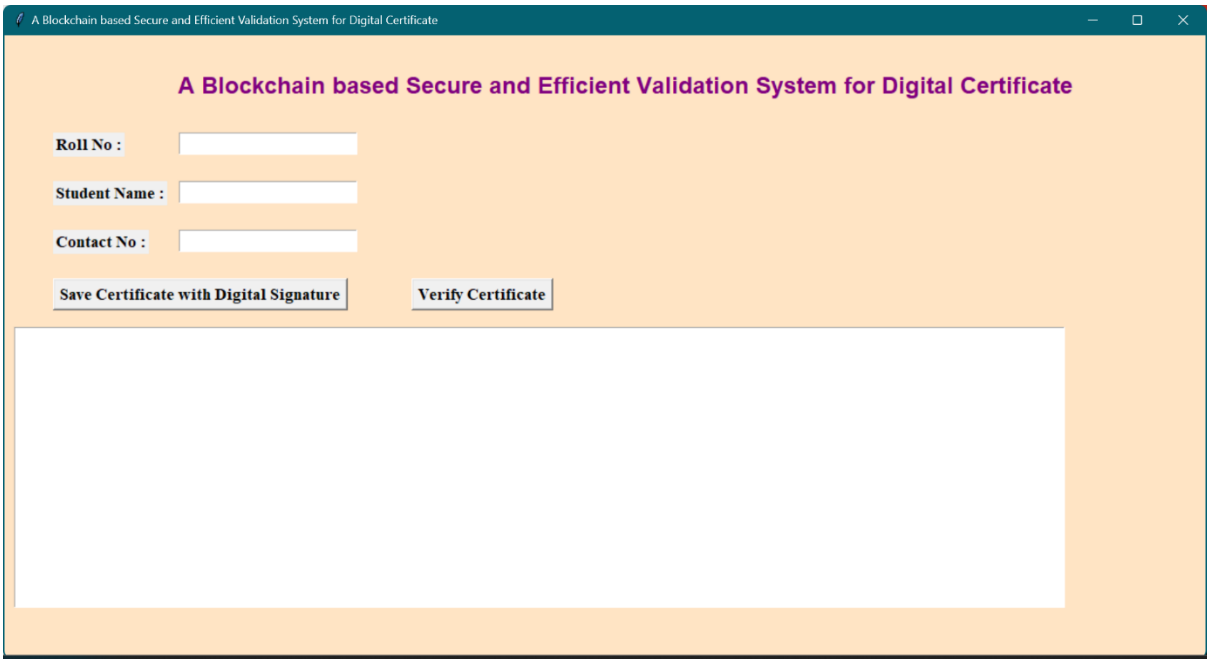
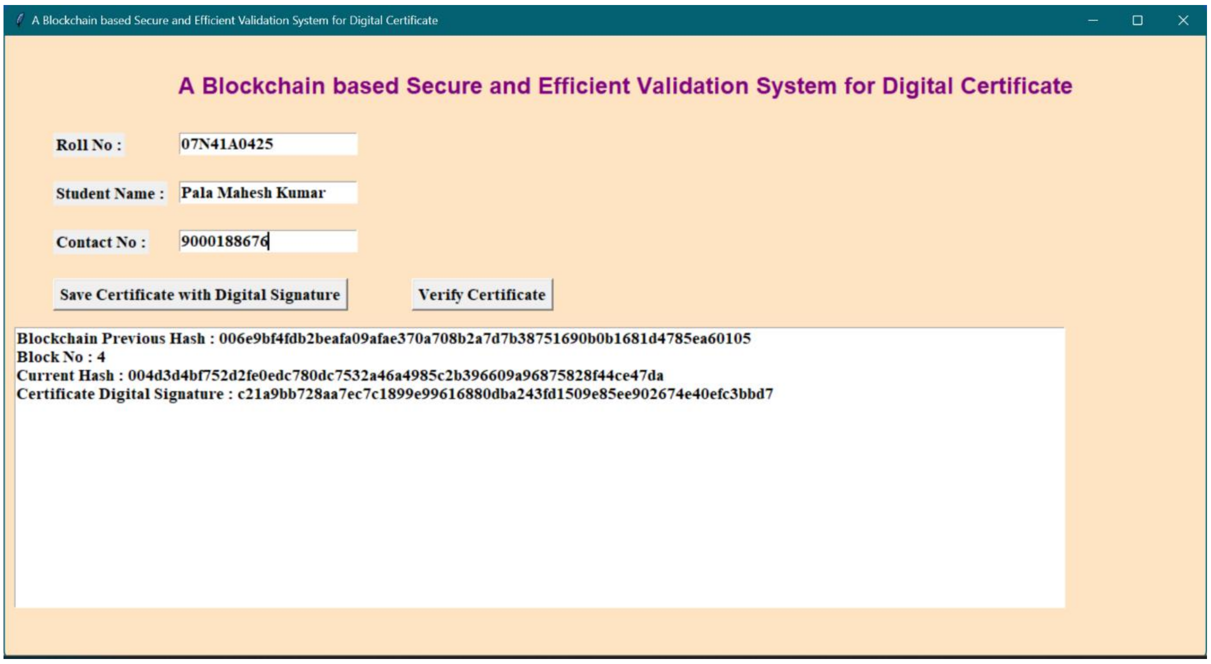

A Blockchain based Secure and Efficient Validation System for Digital Certificate
A Blockchain based Secure and Efficient Validation System for Digital Certificate
Roll No :
Student Name :
Contact No :
Save Certificate with Digital Signature
Verify Certificate

Figure 1: Main GUI application of proposed blockchain powered certificate verification for forgery prevention.


A Blockchain based Secure and Efficient Validation System for Digital Certificate
A Blockchain based Secure and Efficient Validation System for Digital Certificate
Save Certificate with Digital Signature
Verify Certificate

Roll No : 07N41A0425

Student Name : Pala Mahesh Kumar

Contact No : 9000188676

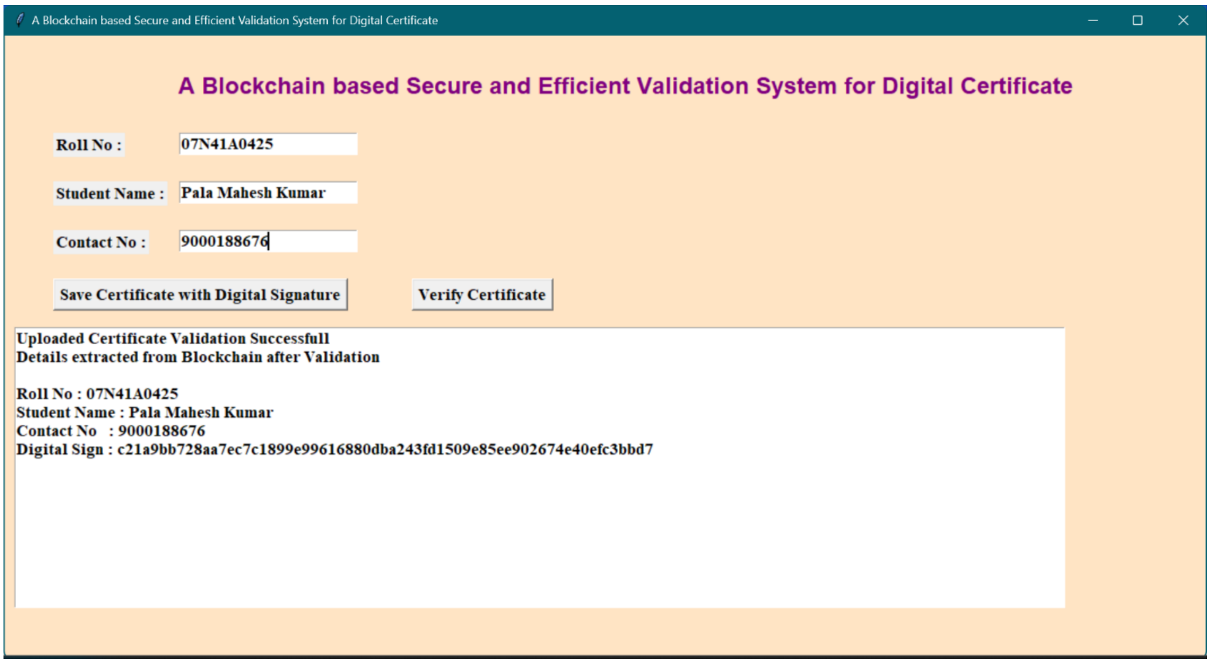
```
Blockchain Previous Hash : 006e9bf4fdb2beafa09afae370a708b2a7d7b38751690b0b1681d4785ea60105
Block No : 4
Current Hash : 004d3d4bf752d2fe0edc780dc7532a46a4985c2b396609a96875828f44ce47da
Certificate Digital Signature : c21a9bb728aa7ec7c1899e99616880dba243fd1509e85ee902674e40efc3bbd7
```

Figure 2: GUI application after entering the details of student with roll number, student name, and contact information.

Figure 3 shows the GUI after the user has clicked the "Store Certificate in Blockchain" button and the certificate has been successfully stored in the blockchain. It includes the following changes:

— Information about the stored certificate, such as its digital signature and blockchain details, is displayed in the text widget.

— The user receives confirmation that the certificate has been successfully added to the blockchain.

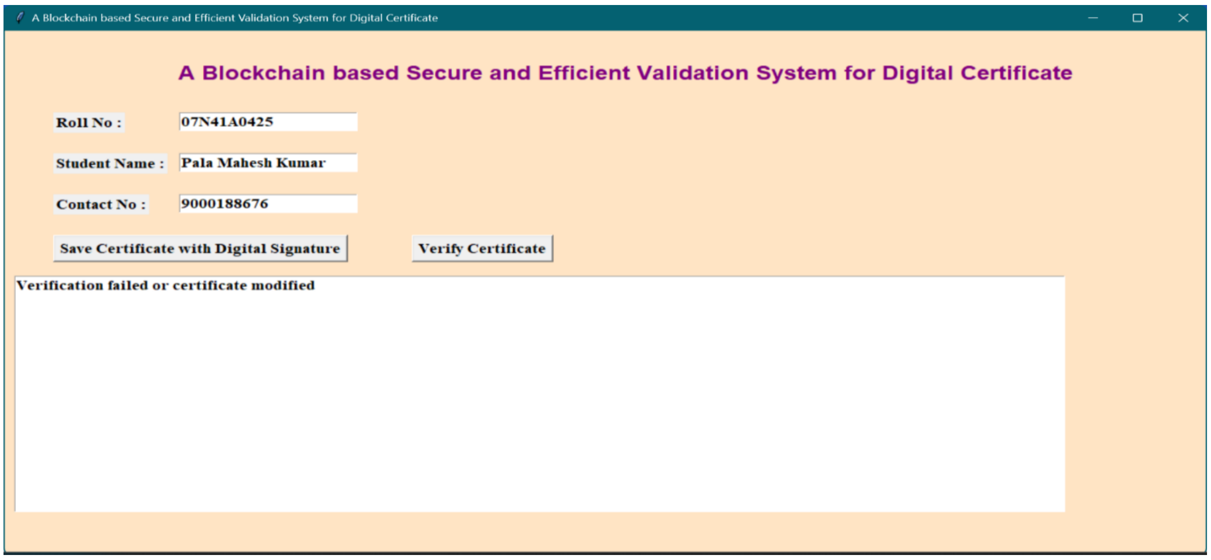

A Blockchain based Secure and Efficient Validation System for Digital Certificate
A Blockchain based Secure and Efficient Validation System for Digital Certificate
Save Certificate with Digital Signature
Verify Certificate

Roll No : 07N41A0425
Student Name : Pala Mahesh Kumar
Contact No : 9000188676

```
Uploaded Certificate Validation Successfull
Details extracted from Blockchain after Validation

Roll No : 07N41A0425
Student Name : Pala Mahesh Kumar
Contact No   : 9000188676
Digital Sign : c21a9bb728aa7ec7c1899e99616880dba243fd1509e85ee902674e40efc3bbd7
```

Figure 3: Illustration of proposed GUI application after storing the certificate with student entry details.


A Blockchain based Secure and Efficient Validation System for Digital Certificate
A Blockchain based Secure and Efficient Validation System for Digital Certificate
Save Certificate with Digital Signature
Verify Certificate

Roll No : 07N41A0425
Student Name : Pala Mahesh Kumar
Contact No : 9000188676

```
Verification failed or certificate modified
```

Figure 4: Checking the certificate verification with test template as input and displaying the validation is successfully verified with student enrolment.

In Figure 4, the user appears to be using the application to verify a certificate's authenticity that includes the following elements:

— The user has clicked the "Certificate Verification" button.

— A file dialog or input field is displayed for the user to select a test template (likely a certificate template) for verification.

— After selecting the template, the application calculates its digital signature and compares it to the blockchain's records.

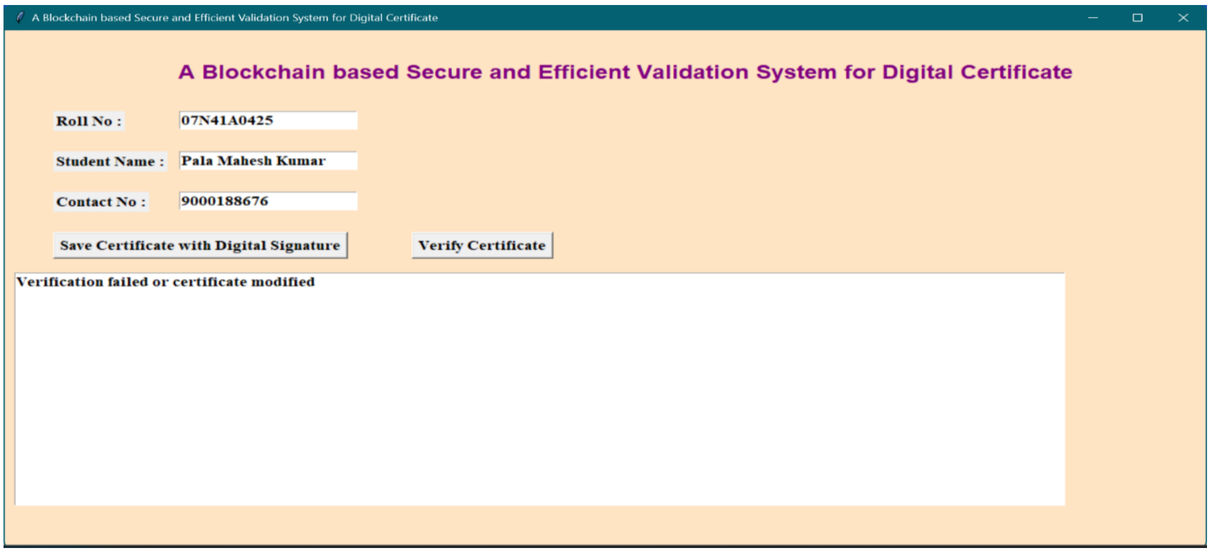
— The text widget displays a message indicating successful verification, along with student enrolment details.

Figure 5 shows the GUI when the certificate verification process has failed or when the certificate has been modified. It includes the following elements:

— The user has attempted to verify a certificate, but the digital signature does not match any records in the blockchain.

— The text widget displays a message indicating that the verification has failed or that the certificate has been tampered with.

The user is informed that the certificate is not valid.


A Blockchain based Secure and Efficient Validation System for Digital Certificate
A Blockchain based Secure and Efficient Validation System for Digital Certificate
Save Certificate with Digital Signature
Verify Certificate

Roll No : 07N41A0425

Student Name : Pala Mahesh Kumar

Contact No : 9000188676

Verification failed or certificate modified

Figure 5: GUI application with verification of certificate is failed or the certificate is modified.

CHAPTER 11

CONCLUSION AND FUTURE SCOPE

11.1 Conclusion

In conclusion, the implementation of blockchain technology in certificate verification holds immense potential for combating the persistent problem of certificate forgery across various industries. By offering a decentralized, tamper-resistant, and transparent platform, this innovative approach addresses the limitations of traditional verification methods. The immutability of blockchain ensures that certificate records remain secure and unalterable, thereby enhancing their credibility and authenticity. Additionally, the elimination of central authorities reduces the risk of data manipulation and fosters trust in the verification process. However, to fully realize the benefits of blockchain-powered certificate verification, several challenges must be overcome. These include ensuring widespread adoption of the technology, addressing scalability issues, and developing user-friendly interfaces for individuals and institutions.

11.2 Future Scope

Future research and development efforts should focus on standardizing blockchain-based certificate verification protocols and integrating them seamlessly into existing certification processes. Furthermore, exploring the use of smart contracts to automate verification and streamline administrative tasks could further enhance the efficiency of the system. As blockchain technology continues to evolve, its potential to transform certificate verification into a more secure, efficient, and reliable process remains a promising avenue for further exploration and implementation.

REFERENCES

[1]. Siam, A.I.; Almaiah, M.A.; Al-Zahrani, A.; Elazm, A.A.; El Banby, G.M.; El-Shafai, W.; El-Samie, F.E.A.; El-Bahnasawy, N.A. Secure Health Monitoring Communication Systems Based on IoT and Cloud Computing for Medical Emergency Applications. Comput. Intell. Neurosci. 2021, 2021, 8016525.

[2]. Ali, A.; Pasha, M.F.; Fang, O.H.; Khan, R.; Almaiah, M.A.; KAl Hwaitat, A. Big data based smart blockchain for information retrieval in privacy-preserving healthcare system. In Big Data Intelligence for Smart Applications; Springer International Publishing: Cham, Switzerland, 2022; pp. 279–296. [Google Scholar]

[3]. Altulaihan, E.; Almaiah, M.A.; Aljughaiman, A. Cybersecurity Threats, Countermeasures and Mitigation Techniques on the IoT: Future Research Directions. Electronics 2022, 11, 3330. [Google Scholar]

[4]. Hasnain, M.; Pasha, M.F.; Ghani, I.; Mehboob, B.; Imran, M.; Ali, A. Benchmark Dataset Selection of Web Services Technologies: A Factor Analysis; IEEE Access: Piscataway, NJ, USA, 2020; Volume 8, pp. 53649–53665. [Google Scholar]

[5]. Ali, A.; Rahim, H.A.; Pasha, M.F.; Dowsley, R.; Masud, M.; Ali, J.; Baz, M. Security, Privacy, and Reliability in Digital Healthcare Systems Using Blockchain. Electronics 2021, 10, 2034. [Google Scholar]

[6]. Almaiah, M.A.; Hajjej, F.; Ali, A.; Pasha, M.F.; Almomani, O. An AI-Enabled Hybrid Lightweight Authentication Model for Digital Healthcare Using Industrial Internet of Things Cyber-Physical Systems. Sensors 2022, 22, 1448. [Google Scholar]

[7]. Yazdinejad, A.; Dehghantanha, A.; Parizi, R.M.; Srivastava, G.; Karimipour, H. Secure Intelligent Fuzzy Blockchain Framework: Effective Threat Detection in IoT Networks. Comput. Ind. 2023, 144, 103801.

[8]. Hameed, K.; Ali, A.; Naqvi, M.H.; Jabbar, M.; Junaid, M.; Haider, A. Resource management in operating systems-a survey of scheduling algorithms. In Proceedings of the International Conference on Innovative Computing (ICIC), Lanzhou, China, 2–5 August 2016; Volume 1. [Google Scholar]

[9]. Singh, H.; Ahmed, Z.; Khare, M.D.; Bhuvana, J. An IoT and Blockchain-Based Secure Medical Care Framework Using Deep Learning and Nature-Inspired Algorithms. Int. J. Intell. Syst. Appl. Eng. 2023, 11, 183–191. [Google Scholar]

[10]. Kim, H.; Kim, S.-H.; Hwang, J.Y.; Seo, C. Efficient Privacy-Preserving Machine Learning for Blockchain Network. IEEE Access 2019, 7, 136481–136495. [Google Scholar]

[11]. Sharma, P.; Namasudra, S.; Crespo, R.G.; Parra-Fuente, J.; Trivedi, M.C. EHDHE: Enhancing security of healthcare documents in IoT-enabled digital healthcare ecosystems using blockchain. Inf. Sci. 2023, 629, 703–718. [Google Scholar]

[12]. Almadani, M.S.; Alotaibi, S.; Alsobhi, H.; Hussain, O.K.; Hussain, F.K. Blockchain-based multi-factor authentication: A systematic literature review. Internet Things 2023, 23, 100844.

[13]. Kumar, P.; Kumar, R.; Gupta, G.P.; Tripathi, R.; Jolfaei, A.; Islam, A.N. A blockchain-orchestrated deep learning approach for secure data transmission in IoT-enabled healthcare system. J. Parallel Distrib. Comput. 2023, 172, 69–83.

[14]. Yazdinejad, A.; Parizi, R.M.; Dehghantanha, A.; Choo, K.-K.R. P4-to-blockchain: A secure blockchain-enabled packet parser for software defined networking. Comput. Secur. 2020, 88, 101629.

[15]. Sharma, P.C.; Mahmood, R.; Raja, H.; Yadav, N.S.; Gupta, B.B.; Arya, V. Secure authentication and privacy-preserving blockchain for industrial internet of things. Comput. Electr. Eng. 2023, 108, 108703.

[16]. Jiang, S.; Cao, J.; Wu, H.; Yang, Y. Fairness-Based Packing of Industrial IoT Data in Permissioned Blockchains. IEEE Trans. Ind. Inform. 2020, 17, 7639–7649.

[17]. Bordel, B.; Alcarria, R.; Robles, T. A Blockchain Ledger for Securing Isolated Ambient Intelligence Deployments Using Reputation and Information Theory Metrics; Wireless Networks: New York, NY, USA, 2023; pp. 1–7. [Google Scholar]

[18]. Selvarajan, S.; Srivastava, G.; Khadidos, A.O.; Khadidos, A.O.; Baza, M.; Alshehri, A.; Lin, J.C.-W. An artificial intelligence lightweight blockchain security model for security and privacy in IIoT systems. J. Cloud Comput. 2023, 12, 38.

[19]. Lacity, M.C. Addressing Key Challenges to Making Enterprise Blockchain Applications a Reality. J. Mis. Q. Exec. 2018, 17, 3. [Google Scholar]

[20]. Sengupta, J.; Ruj, S.; Das Bit, S. A Comprehensive Survey on Attacks, Security Issues and Blockchain Solutions for IoT and IIoT. J. Netw. Comput. Appl. 2020, 149, 102481.

[21]. Pajooh, H.; Rashid, M.; Alam, F.; Demidenko, S. Multi-Layer Blockchain-Based Security Architecture for Internet of Things. Sensors 2021, 21, 772.

[22]. Peng, C.; Wu, C.; Gao, L.; Zhang, J.; Yau, K.-L.A.; Ji, Y. Blockchain for Vehicular Internet of Things: Recent Advances and Open Issues. Sensors 2020, 20, 5079.

[23]. Esposito, C.; De Santis, A.; Tortora, G.; Chang, H.; Choo, K.-K.R. Blockchain: A Panacea for Healthcare Cloud-Based Data Security and Privacy? IEEE Cloud Comput. 2018, 5, 31–37.

[24]. Patel, V. A framework for secure and decentralized sharing of medical imaging data via blockchain consensus. Health Inform. J. 2018, 25, 1398–1411.

[25]. Kim, T.M.; Lee, S.-J.; Chang, D.-J.; Koo, J.; Kim, T.; Yoon, K.-H.; Choi, I.-Y. DynamiChain: Development of Medical Blockchain Ecosystem Based on Dynamic Consent System. Appl. Sci. 2021, 11, 1612.

Project Details	
Academic Year	2020-2024
Title of the Project	A Blockchain Based Secure and Efficient Validation System for Digital Certificates
Name of the Students and Hall Ticket No:	V. Sai Bharath (20RA1A0566) P. Sai Ram (20RA1A0572) P. Sai Rohith (20RA1A0568)
Name of the Guide	Dr. C. Bagath Basha

Project PO Mapping

Name of the Course from which Principles are applied in this Project	Related Course Outcome Number	Description of the application	Page Number	Attained
Python Programming, Software Engineering (C413, C313)	C313.1, C413.1	Students described the basis for their problem statement.	01	PO2
Blockchain Technology, Python Programming, (C413, C413,)	C413.2, C413.3,	Students explained about A Blockchain Based Secure and Efficient Validation System for Digital Certificates	01-04	PO1
Python Programming, Software Engineering (C313, C413)	C313.3, C413.2	Students identified the existing system and its Drawbacks and proposed a solution to it.	08-10	PO2, PO3
Design Patterns, Software Engineering, (C313, C322)	C313.2, C322.3	Students explain the flow of the project using UML diagrams	16-20	PO3, PO5, PO9, PSO3
Python Programming, (C413, C411)	C413.2,	Students explained about python programming language and designed the modules for the solution of the problem	22-33	PO2, PO3, PO4
Software Engineering (C313)	C313.1	Students identified the hardware and software required for the project.	34-39	PO5

Python Programming (C327, C413, C413)	C327.3, C413.2, C413.3	Students developed code for the problem statement.	40-44	PO3, PO4, PO5
Software Engineering, Software Testing Methodologies (C313, C324)	C313.2, C324.3	Students performed various testings for the code they developed	45-49	PO9, PO11, PO2
Future Scope		Students explained about how they would like to further their project and develop it as their future scope.	50	PO12, PSO2
Bibliography		Listed the references from which the literature was collected	51-53	PO8, PO12
ENGLISH		Prepared the thesis and intermediate progress reports and explained to the panel. Also, continuously interact with guide and explain the progress		PO9, PO10

SIGNATURE OF STUDENT **SIGNATURE OF INTERNAL GUIDE**