# Contents

# 1 Introduction

This section gives a brief overview of all the major topics, subjects and technologies needed for this application. The introduction covers blockchain technology and its features in depth, and discusses the motivation behind the research and application along with the problem statement.

## 1.1 Blockchain Technology

A blockchain is a decentralized database that is common among computer network nodes. A blockchain can store digital information and can act as a database. Blockchains are especially known for the integral function of keeping information tamper-free and decentralized and a record of transactions in cryptocurrency systems like Bitcoin [19]. The blockchain's innovation is that it generates trust without a trusted third party and ensures the accuracy and security of a data record.

In blockchain technology, transferring of digital assets is considered as a transaction that is stored on the blockchain. In the Bitcoin system, transferring ownership is called a "payment" in which the digital asset is itself "Bitcoin", which in turn is a digital currency, and the owners that can transfer and receive these digital assets are Bitcoin wallet holders identified by their unique asymmetric cryptographic keys. Specific nodes known as miners are responsible for approving and adding transactions to the blockchain. When a transaction is started, transaction details like sender, receiver, digital asset amount being transferred, and the timestamp indicating the time of the transaction are hashed together and broadcasted to the pending transaction pool. Miners put together a list of transactions to constitute a candidate block. The candidate block also contains its own timestamp, the hash value of the candidate block which is generated from the block header and the Merkle root hash value of transactions included in the block, and the hash value of the last approved block in the ledger [13]. If this hash value is equal to or greater than the required difficulty level, then this block is transmitted to the whole network, and all the transactions in the block go from pending to approve.

There are two types of blockchain: public (permissionless) and private (permissioned) blockchain. In a public blockchain, there are no restrictions, anyone can participate and become a node and do transactions. This kind of blockchain is completely transparent, as all transactions are made public to all the participating nodes. Examples of public blockchains are Bitcoin, Ethereum, etc. On the other hand, private blockchains are permissioned to gain or provide access. This facilitates the trade between privacy and transparency among the peers on the blockchain and is suitable for company use cases. An example of a private blockchain is Ripple.

## 1.2 Properties Of Blockchain

The three pillars of the blockchain have been the reason for widespread adoption of this technology

### 1.2.1 Transparency

Transparency is one of the most convincing reasons to have trust in blockchain technology. Blockchain provides a fully auditable and valid ledger of transactions. Through means of the necessary encryption and control mechanisms, blockchain guarantees confidentiality by preserving information in such a way that it cannot be updated without documenting the changes made.

### 1.2.2 Decentralization

Blockchain is a fully decentralized system, so it is not controlled by just one organization, thus shifting the power from them to everyone involved.The functioning of the blockchain is based on consensus algorithms, which helps in taking all the decisions community driven.

### 1.2.3 Immutability

Immutable is a property which allows the blockchain to be unchangeable.Anything recorded on the blockchain ledger is immutable and tamperproof, which means it cannot be changed. If there are any attempts of tampering with blockchain in an effort to alter it according to one's profit the changes are visible to everybody, anyone attempting to alter the contents of blockchain is known to all and thus the changes will not be accepted by the network.

## 1.3 Components of Blockchain

We discuss the components of blockchain architecture like block, consensus algorithm, cryptography, hashing, and digital signature.Then we also to discuss some popular blockchains which have been used for Identity Management.

### 1.3.1 Block

Block is the foundation of blockchain. A block in a blockchain is made up of transactions, the block's metadata, and the hash of the previous block, because of which blocks are chained together to form a blockchain. The first block in any blockchain is known as the genesis block.

### 1.3.2 Consensus Algorithm

Consensus is the procedure through which all the nodes involved in the network reach a common agreement about the current state of the ledger. Due to the help of consensus algorithms, trust between parties can be established and maintained without even involving a third party. There are various consensus algorithms available and some of them have been listed below:

- **Proof of Work**: The Proof of Work (PoW) protocol, requires miners to go through an intense race of trial and error to find the nonce for a block. Only blocks with a valid nonce can be added to the chain. This mathematical puzzle needs a lot of computing resources, and thus the node that solves the puzzle gets to mine the next block as mentioned in [23].
- **Proof of Stake (POS)**: Proof of Work is not a viable solution as it is very expensive. Expensive hardware is required to perform complex and fast computations and a lot of electricity. In the Proof of Stake algorithm, validators will invest some of the system coins that they own and lock them up as stake. When a block of transactions is ready to be processed, the cryptocurrency's proof-of-stake protocol will choose a validator node to review the block. The validator checks if the transactions in the block are accurate. If so, they add the block to the blockchain and receive crypto rewards for their contribution. However, if a validator proposes adding a block with inaccurate information, they lose some of their staked holdings as a penalty as explained in [20].
- **Delegated Proof of Stake (DPoS)**: [8] explains that A blockchain-based on Delegated Proof of Stake (DPoS) has a voting mechanism in which members outsource their work to a third party. In other words, they are able to vote for a few delegates that will secure the network on their behalf. The delegates are also called witnesses, and they are responsible for achieving consensus during the generation and validation of new blocks. The voting power is proportional to the number of coins each user holds. Therefore, this consensus is based on the reputation of miners.
- **Proof of Authority**: This consensus algorithm provides rights to the node to generate the next block if they have proven their authority to do so. The validators do not stake their coins, but stake their reputation.

### 1.3.3 Cryptography, hashing and digital signature

Public-key cryptography (asymmetric cryptography) is an encryption technique that has been used for many years. It is asymmetric cryptography that contains the use of 2 keys as a pair: a public key and its corresponding private key. Asymmetric cryptography is a process that allows us to encrypt and decrypt messages using two separate keys. The sender encrypts the message with the help of the public key of the receiver and sends the message. Now the receiver decrypts the received message with his own private key. Private keys are known only to the owners, while public keys are known to all. This ensures confidentiality and integrity. In Blockchain, public-key cryptography is utilized for asset ownership and for verifying the authenticity of transactions.

Hashing is a mathematical mechanism of generating a fixed size value from input data. It is made in such a way that it is not practically possible to regenerate the original input given its hash value. In Blockchain, hashing is utilized in hashing transaction and block data. For example, Bitcoin system uses SHA256 hashing algorithm, which was originally designed by United States National Security Agency (NSA). SHA256 is also being used for decades in many sectors and services that require solid security, such as in financial services, and it has proven to be secure.

Digital signature refers to digitally signing a message in order to certify authenticity and ensure integrity of the message in peer-to-peer communication, which is achieved using public-key cryptography and hashing.

### 1.3.4 Different Blockchains

- **Ethereum**: [2] talks about Ethereum, which is a decentralized, open sources' platform with the functionality of smart contracts. One of the goal of Ethereum was to build decentralized applications. It uses proof of work consensus, however Ethereum 2.0 uses Proof of Stake consensus.
- **Hyperledger**: [5] mentions that Hyperledger is also open source, the community builds stable frameworks, tools and libraries for enterprise-grade blockchain deployments.
- **uPort**: uPort is a collection of tools and protocols for building decentralized user-centric applications. It's built on open standards and open source libraries.
- **Sovrin**: [30] mentions about Sovrin as a global utility for self-sovereign identity–that is, an identity that nobody controls except its natural owner, that can't be taken away or stripped of its privacy or manipulated through unreasonable terms of service.

## 1.4 Blockchain Identity

[24] explains that Identity management is composed of processes and technologies inside an organization that is used to authenticate and authorize, identified individuals to access restricted internal systems and services of that organization or any other associated institution.

Some examples of identity management are users using his or her login credentials to identify and authenticate themselves to a software application. Another example could be a company employee authenticating themselves so as to gain access to documents or services they have been authorized to view or use. Government-issued identifications like birth certificates, passports, driver's licenses can be used to identify and authenticate a citizen of that country.

### 1.4.1 Models of Digital Identity Management

There are two types of models of digital identity management:

- **Centralized Identity**: A centralized Identity is an account that is created on a website, owned and controlled by a single entity. For example, signing up on a website or accessing a bank account using online banking. All the user's data is stored on a centralized server or database, which is problematic as these servers can be hacked. This also creates a poor user experience.
- **Federated Identity**: Since the user experience of centralized identity management was unsatisfactory and poor, many third parties made it possible for users to use the identity established in one security domain to access another domain. This federated identity management system is also known as single sign-on (SSO). The best examples of this are "Login with Google" and "Login with Facebook" functionalities. Many organizations and institutions started to outsource their identity management needs to major corporations, who have an economic interest in amassing such large databases of personal data. This, of course, raises privacy and security concerns.

  A new type or model of identity management is emerging, known as **Self-Sovereign Identity**, is possible due to the development of Blockchain and other technologies like Decentralized Identifiers, and Verifiable Credentials, which gives power back to the user.

### 1.4.2 The problem with current Identity Management Systems

Every website, service provider, and application has taken a different approach to the Internet's missing identity layer differently, this has led to a lot of confusion all across the internet. This has resulted in a lot of internet users being subject to cognitive overload, friction, high costs, privacy loss, and cyber crimes.

The internet's identity problem is a hard one to solve. There have been numerous attempts made in the past to solve this issue, various protocols and standards have been proposed in the past many years. But we are yet to come to a solution that is holistic, all the solutions proposed so far have only solved specific problems or have made only improvements to certain aspects of the problem

To better understand why the problem of digital identity is so hard to solve, it is important to consider the various differences between digital and physical identity, which are mentioned in [25].

- **Proximity**: Since there is no physical interaction between people in the online world, we can't make use of the familiar physical world signals to get an understanding of whom we are dealing with. As a result, it is difficult to recognize and remember people and organizations in the online world with a particular level of surety. Most big Organizations built various complex systems to better help them understand, remember and recognize their customers, this is difficult for a common person who has far fewer capabilities. As a result, we are mired in myriad, incompatible systems built for narrow purposes.

- **Usability**: Verifying a person's identity online is a very tedious task when using traditional identity systems. Many businesses in the past few years have made the transition from offline to online, especially during the Covid-19 pandemic. Consequently, managing identity verification remotely, Know Your Customer (KYC) requirements, signing and sharing or documents / credentials has been crucial for individuals. Users these days are required to create an account having a username and password for authentication on any online services that they might use. This becomes difficult for users to keep track of and remember multiple login information they have to use when accessing various online services across the web. Moreover, identity owners have to give out personal and sensitive information to answer questions that might be asked when a remote authentication is detected. Therefore, users are at a disadvantage of spending time to prove their identity and by providing private and sensitive information just to verify their identity in the online world. A survey report by Centrify in 2014 showed that small businesses such as those having only 500 employees lose approximately $200,000 a year in productivity due to the time spent on password management

- **Privacy**: Identity assets that an individual uses to prove who they claim to be are traditionally stored by third parties. Giving them the control over the individual's data. These third party organizations may include various websites, companies or government that maintain large warehouses of identity data. They mostly make use of private and sensitive information to verify the user's identity, information such as phone number, father's name, place of birth etc. These Sensitive information are stored in repositories without the consent or approval of the identity owners. Many of the times this data is manipulated and exploited by companies for their own commercial interests.

- **Anonymity**: In real life situations, we don't make use of identity systems. For example, we aren't required to prove our identity when we visit a shopping mall, or we don't need to log into any system when we go to a hair salon. We can perform interactions in the physical world anonymously because these interactions don't last for long. A security guard at the shopping mall does "identify" you momentarily for purposes and performs a full body scan, but that connection is short-lived and can be done anonymously, as with most interactions in the physical world. Many interactions performed in the online world can make use of ephemeral relationships to become more secure in terms of privacy.

- **Interoperability**: The main consequence of having countless data warehouses storing user's identity is that the users aren't able to carry context from system to system. People mostly tend to use different identifiers on different platforms. As a result, it makes it difficult for users to recognize and remember people they know as they move from one system to another.

- **Security**: Traditionally, service providers tend to keep some of an individual's identity information for future verification of that individual. This poses a major risk to the user's identity, as hackers who constantly attack such systems might get a hold of their personal information. Such breach of data may result in great harm to not only the user, but also to the owners of the business and business itself. According to Javelin's Identity Fraud Study in the U.S. Identity frauds have incurred a total loss of $16.8 billion in 2017, affecting over 16.7 million customers. Rather, the more alarming fact is that the number of Identity fraud cases jumped by around 8% in the following year. This has a negative relationship with economic damages. According to Herjavec Group, there are predictions made that the damages incurred by cyber-security breaches will cost the world $6 trillion in a year by 2021.

- **Scale**: On a global level, the process of identity verification and attestation poses an even greater challenge due to the institutional and international borders. When a person moves to another country, he/she has to again go through the tedious process of identity verification right from the start, i.e. the process of verification of physical documents such as passports etc. Moreover, verification of many credentials such as college transcripts, credit reports is a very slow process. Also, disconnected processes may require various other third party services for verification. For instance, educational degrees earned by individuals in India will require that particular person to seek the help of trusted third parties to prove its legitimacy in a U.S. government office.

These problems require a more generic solution rather than one that solves specific issues of the traditional identity verification systems.
[24] tells about various industries have to suffer the problems of current identity management systems:

- **Government**: The shortfall of interoperability between the various departments and levels of government takes a toll in the form of excess bureaucracy. This further leads to an increase in the time and cost of a process

- **Healthcare**: Many people across the world don't have access to proper healthcare facilities and services. Inefficient and delayed healthcare has been the result of shortfall of interoperability between various entities in the healthcare sector such as Hospitals, clinics, insurance companies, doctors, pharmacies, etc.

- **Education**: Reports estimate that around 2 Lakh fake education certificates are sold each year in the USA alone. The complexity in verifying the legitimacy of these certificates have led companies to hire unqualified individuals. Moreover, this caused brand damage to not only the universities but also to the companies that end up hiring such individuals.

- **Banking**: Traditional systems that require various details during login, such as passwords, increase the security risks of banking for users.

- **Businesses in general**: It is a tedious and high risk task for companies to store their clients' and employees' personal information. This is because any breach of personal information may rack up huge amounts of fine due to GDPR infringement. Moreover, it may result in customers losing faith in the company and consequently damage to the company's reputation.

## 1.5 Motivation

Many websites support Facebook and Google single sign-on (SSO) solutions for end-users. Unfortunately, this induces a cost in terms of privacy. Often, the business interests of such centralized service providers like Facebook and Google are not aligned with the users' interests. This may lead to the users' data being used beyond their intention when initially sharing it. Users could still seek the benefits of the SSO customer experience of not having to register at each website, without sacrificing their privacy.

Additionally, there are valid concerns about the security of existing authentication systems. Even certain two-factor authentication methods are not secure enough for use cases like online banking since powerful frameworks like Muraena and NecroBrowser can attack time-based one-time passwords (TOTP) generated via software (e.g., Google Authenticator) or hardware (e.g., RSA SecurID). Furthermore, the widely used SMS tokens can be intercepted with SIM swap attacks and set for replay attacks. Data breaches affecting millions of users already happened even at companies like Facebook and Google, and end-users typically have limited control over their data.

## 1.6 Problem Statement

Identity management is the process of setting and organizing the roles and access privileges of users. The identity management system helps in reducing human error, effective accessing of resources, stronger authentication systems like multifactor authentication, and provides flexibility in changing attributes related to the identity. The current identity management system is centralized, which is controlled by a single entity. User's privacy concerns are not in their best interest. Users have very restricted control over their data. The centralized system becomes a single point of failure which is prone to attack that leads to users losing their data privacy if these centralized systems are breached. Intrusive advertisements are also targeted to users, depending on their private data held in these centralized systems.

We propose a Blockchain-based decentralized Identity that provides self-sovereign identity, decentralized identifiers, verifiable credentials, and provide feature to import and export user account. The proposed system consists of 2 parts:

- Decentralized Identity enables identity owners to prove their identity to trusted entities like authentication (login) systems.
- Verifiable credentials using which a user can make a claim involving certain third-party certifications and attestations (e.g. proof of education degree, government-issued identity cards, etc.)

The main advantages of the proposed solution include the elimination of the need for a central authority for identity verification and identity data management, the reduction of time spent on identity verification, the ability to share data with permission, and the ability to verify the origin of the data while sharing. Furthermore, biometric authentication will be required to access the verifiable credentials of the user. This is much safer than using regular email and password, as the user might forget his or her password or use the same password for different applications, thus making it much weaker.

The suggested system's goal is to provide an identity management system that is remote-friendly, scalable and by design provides both privacy and security.

## 1.7 Organization of the Report

The report is organized in a way to create an understanding of the Decentralized Identity, Chapter 2 contains the literature review. Chapter 3 consists of Proposed system. Chapter 4 contains the proposed architecture followed by the implementation in chapter 5 and Results and Conclusion including Future Scope in Chapter 6 and 7 respectively.

# 2 Literature Survey

In this section, we present an overview of some existing research on Identity management and previous works concerned with enhancing the security and privacy of users using Identity management in Blockchain, followed by overview of existing work in DID and VC.

## 2.1 Blockchain for Digital Identity

Digital identity is a digital establishment of the particulars obtainable about an entity, and the identity management system ensures that only valid users are authorized to gain access to that information. The main issue arises when the centralized identity management system deprives users of the rightful ownership of their identity data online. With the growing need for security and privacy, the blockchain offers a perfect solution for delivering secure identity services. The distributed ledger technology has given a new perspective on identity management systems, and new approaches transpire, aiming to enhance decentralization, user control, and transparency [12].

### 2.1.1 Blockchain Identity as a service

Mutual authentication between the user and the partner is established without any pre-shared information or security credential shared among them. The user does not require any ID creation to the partner offering services to the user. The blockchain identity as a service where the identity service provider and its partners have access to the identity of the user which it can provide to different clients or partners, thus authenticating the user. The partner or the client can also only selectively ask for data related to the user from the identity service provider, as stated in [3].

### 2.1.2 Existing Blockchain-Based Identity Management Systems

[10] gives an overview over six of the most promising identity management systems,

- **Jolocom**: An identity management system that uses hierarchical deterministic keys (HD keys) to offer decentralized identity to its users.
- **ShoCard**: It combines identity management on the blockchain with already trusted credentials like passports or driver licenses. The identities can then be extended with further attributes.
- **Blockstack**: It's a decentralized internet where users do not need to trust remote servers and can run decentralized applications.
- **Namecoin**: An open-source technology that aims at improving decentralization security and privacy. Additionally, it also tries to improve certain structural aspects of the internet, like the domain name system (DNS) and identities.
- **uPort**: an open-source framework for delivering a decentralized identity for a self-sovereign identity. Based on the public permissionless blockchain Ethereum and utilizing its smart contracts.
- **Sovrin**: An open source framework for delivering a decentralized identity for SSI.As it is a permissioned blockchain, therefore, only trusted institutions called stewards can operate nodes while participating in the consensus process.

## 2.2 Security in Identity Management

Security is the fundamental pillar of any software application. Without proper security management, sensitive user data becomes vulnerable to attacks. Security in Identity Management is of utmost importance as sensitive documents are handled by these systems which cannot be compromised.

### 2.2.1 Concerns

Recent scandals on the abuse of personal information from social media platforms, and numerous user identity data breaches raise concerns about technical, commercial, and ethical aspects of privacy and security of user data. European Union's new General Data Protection Regulation (GDPR) is one of the largest changes in data privacy regulation and entails several key regulatory measures for both data controllers and data processors to empower and protect EU citizens' privacy. [14]

### 2.2.2 Different attacks on identity systems

- **Phishing Attacks**: Phishing attacks also no longer exist in DNS-IdM, where the authenticating process does not include redirecting HTTPS since the private key is generated and stored locally. The private key controls the uPortID, and once it is compromised, no user authentication steps are in place to protect

  With the uPortID recovery schema, the user's uPortID is transparently linked to the trustee's uPortID, meaning that it can be a vector of attack that targets ownership and identity privacy. While the controller is compromised and the trustees of uPortID are replaced, the attacker simply gets full control over the user's identity

  From the protected ownership aspect, ShoCard plays the role of the middle man, and that might lead to creating unreliability about whether the ShoCardID would still exist, while the company is non-existent. The users will not be able to make use of their certifications; hence, it puts the user's identity ownership at significant risk [12].

- **Online water army attack**: In traditional systems, it is hard to avoid the emergence of multiple identities from the same physical person. The multiple identities can cooperatively perform Sybil attacks, resulting in undesirable consequences for the system managers [15] .

### 2.2.3 Cryptographic primitives

**Hash Functions**: Hashing is used to ensure the integrity of data, and a hash function is an input independent average linear time algorithm that takes a set of variables or data and transforms it into a fixed size hash digest [14] .

**Digital Signatures**: Digital signature is a cryptographic scheme that guarantees two properties: authenticity, that the data/message created or owned by the known sender, and the non-repudiation property guarantees that the data is not altered, using a pair of keys with an asymmetric cryptographic algorithm like RSA [14] .

A fast authentication system is also presented with the implementation of a practical Byzantine fault tolerance system. The scheme can reduce the authentication frequency and improve access efficiency [6].

In the healthcare industry, there is a need for a secure data sharing infrastructure when it comes to handling health data sharing between institutions. However, there are several challenges related to privacy, security, and interoperability. Second, current systems use a centralized architecture, which requires centralized trust. Moreover, the effective integration of health data and the interoperability between healthcare systems remain a challenging task. Another challenge is that users have limited control over their personal health data [4].

Vehicular Ad Hoc Network (VANET) is a mobile network formed by vehicles, road side units, and other infrastructures that enable communication between the nodes to improve road safety and traffic control. In such a network, vehicles have an on-board unit (OBU) that periodically transmits Basic to the other vehicles around it to inform vehicles around them to inform them about its current status. BSMs contain information on a vehicle's location, speed, heading and other relevant information that can be used to manage traffic congestion and prevent collisions. However, such information can be used to reveal precise movement patterns along with driving behavior, which can lead to long term driver profiling and vehicle tracking. If messages exchanged in VANET wireless communication carry inferable Personally Identifiable Information (PII), it can introduce several privacy threats that could limit the adoption of VANET [3].

## 2.3 Privacy in Identity Management

When it comes to identity management, it is very important for the system to protect user's information. The lack of privacy in identity management systems can expose the information of users, as well as deprive them of the control they have over their information.

### 2.3.1 Concerns with Existing Systems

Traditionally, centralized identity management systems provide the convenience for entity users to manage their identities on the Internet. However, there are many challenging issues in these centralized systems. From the perspective of online users, the existing systems bear the following shortcomings:

- Personal privacy leakage: privately sensitive information (i.e., biological information, financial information, et al.) is often required to be included in the formation of an online virtual identity. The user's privacy is then exposed to the management system, which is hard to protect from leakage or manipulation.
- Inconvenient management: the same physical object has to repeatedly register online identities for different platforms, even though most information provided is the same. Meanwhile, the identities of different platforms are totally isolated, which is inconvenient for users to remember and manage [15] .

When it comes to handling health data that is shared between various institutions, there is a major need for a data-sharing infrastructure that provides privacy. The major challenge related to privacy is that health data is highly privacy-sensitive, especially as more and more data is being stored in a public cloud, raising the risks of data exposure [4].

Currently, a widely accepted approach for preserving privacy in VANET is to use pseudonyms to dissociate the vehicle identifier from the information trail. Pseudonyms (also referred to as pseudo IDs or PIDs) are the temporary identifiers that hide the real identity of the vehicle and can be used for authorizing a vehicle to be the part of the vehicular network. Even with the use of pseudonyms, it is possible to infer personal information about a driver if the same pseudonym is used for an extended period. Therefore, it is important to change these temporary identifiers from time to time in order to disconnect the information stream associated with a specific identifier. Vehicles can use pseudonyms to maintain anonymity and preserve privacy when participating in a VANET. However, for proper functioning, it is also necessary to ensure that:

- only authorized vehicles are allowed to participate in the network and
- individual vehicles are held accountable for any malicious behavior and/or misuse of network resources [18]

### 2.3.2 Solutions

Blockchain technology can be leveraged to achieve the delicate balance between privacy and accessibility of electronic health records [7]. This can be achieved using Ethereum based smart contracts for heightened access control and data privacy. Also, by employing cryptographic techniques for additional security. The idea behind privacy is not just to keep user data private and safe from malicious entities, but also to give users complete control over their own private data and which entities can use that information. Users should be able to grant and revoke permissions from any entity, thus being able to control their private data.

Access control with permission delegation mechanism allows fine granular access to secure resources. Existing architectures for permission delegation and access control are either event-based or query-based. These previous works assume a single trusted delegation service, which however is likely biased or fails to service. Also, they fail to allow users to verify delegation service operations, as such cannot be directly applied to IoT (Internet of Things) due to low power, low bandwidth, ad-hoc and decentralized nature [11]. A decentralized architecture can be used for permission delegation and access control of IoT devices. Also depending on the situation, a query-based or event-based permission delegation can be assigned.

Consent manager - When an identity owner shares a verifiable credential with a verifier, proof of the sharing agreement (consent receipt) is generated and kept on the ledger. A concept receipt is signed by both the identity owner and the verifier, and it includes their DIDs and the shared attributes names and data types without any of the private information. The proof of a concept receipt is a cryptographic hash of the receipt, which is stored on the ledger. By this mechanism, tamperproof evidences of sharing agreement of verifiable credentials are maintained, in case they are required in the future [17].

## 2.4 Using Biometrics in Identity Management

Authentication is the process of determining whether a person is who he or she claims to be. Passwords are sometimes too difficult to remember and store, instead we can leverage biometric authentication such as fingerprint which is unique to every user and can be used to authenticate him/her making it easy for users to access resources without having to remember password.

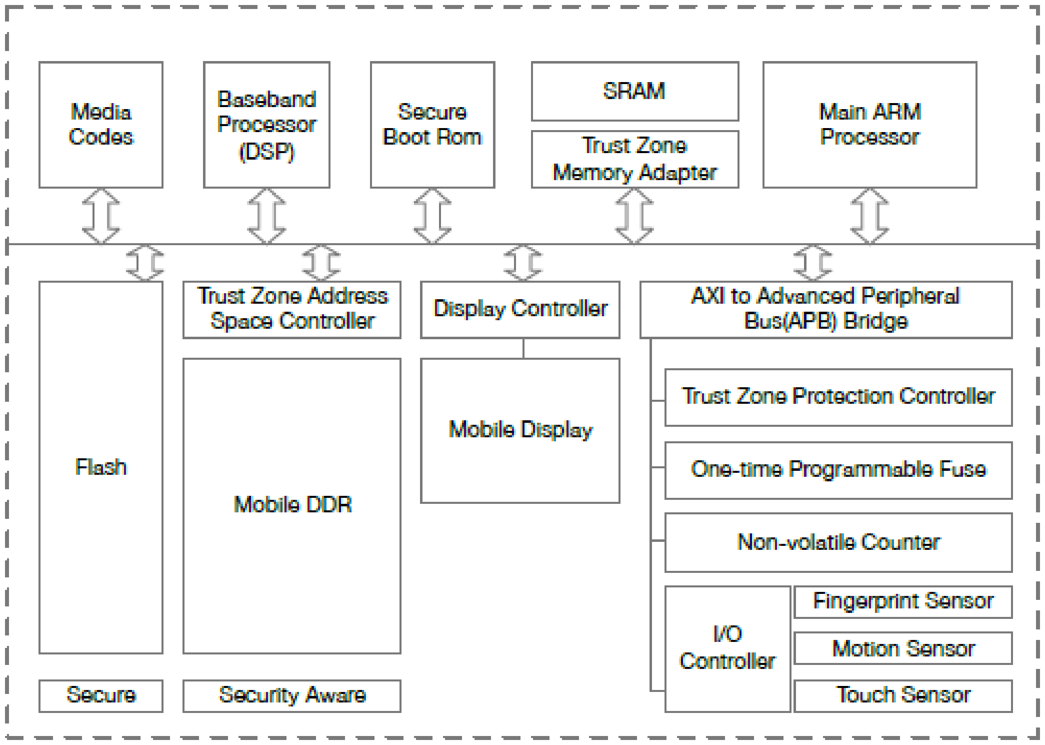


Figure 1: The general hardware layout of the fingerprint module[9]

.

[9] tells us about a finger-print-based authentication framework that includes a secure enclave (Trust

Zone), sensors, and controller as shown in the Figure 1. It also talks about other biometric authentication solutions available on the smartphone, e.g. retina scanning, voice, and face recognition.

[16] tells us about how we can use biometrics for mentally ill patients who cannot remember passwords. It explains fuzzy extractors, which attempt to derive very high entropy cryptographic keys from "fuzzy" (variable) input like biometrics. Such schemes often make use of an underlying error-correcting code in C and, while many variations are possible, can work (in very general terms)it is generated at initialization time using a cryptographically strong pseudo-random number generator with the device and human-randomness as a seed.

## 2.5 DID And Verifiable Credentials

We can use The W3C standardization efforts on verifiable credentials and decentralized identifiers for Single Sign On and Authentication mechanisms.

### 2.5.1 Decentralized Identifiers

[17] tells us about DID Auth as a possible local authentication method for an OIDC Provider and as an alternative OIDC provider discovery method through the service endpoint attribute of the DID Document obtained through DID resolution.
It explains about the DIF created specification of DID authentication for OIDC. In which, the OIDC Client does not rely on a preconfigured OIDC Provider but uses a Self issued OIDC Provider (SIOP) achieving maximum decentralization and privacy.

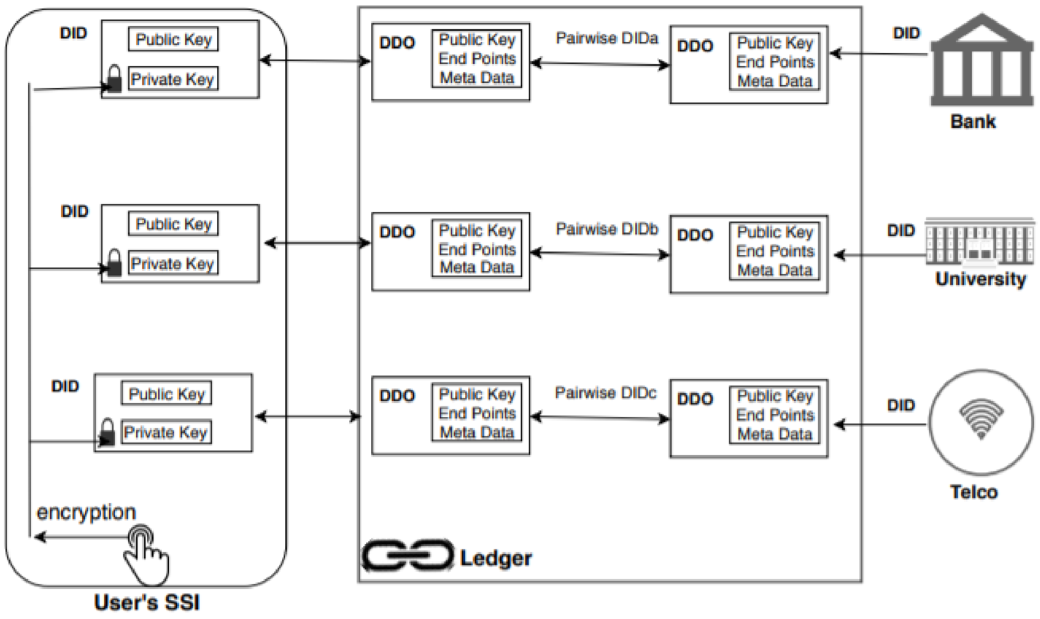


Figure 2: Self-sovereign identity stored on user device and pairwise decentralized identifiers.[13]

[13] explains how SSI consists of multiple decentralized identifiers (DID), one for each relation the identity owner has with other identity owners as shown in figure 2.Each DID is globally unique and includes a cryptographically verifiable PKI (public, private key pair).Each DID resolve to a DID document, a DID descriptor object (DDO) which is stored on the Blockchain.

The details of cryptographic key pairs (public and encrypted private keys) of DIDs that belong to the user's self-sovereign identity are stored on user's devices, such as mobile phones. Public keys are stored in non-encrypted form, corresponding private keys are stored in encrypted form. A private key is encrypted in a way such that it can only be decrypted using a biometric signature of the identity owner, as a fingerprint, facial feature, an iris or a retina.

### 2.5.2 Verifiable Credentials

[17] mentions that Verifiable credentials are machine-readable, privacy respecting, cryptographically secure digital credentials of identity owners. Verifiable credentials support self-sovereign identity, such that identity owners accumulate credentials into an identity account and use the credentials to prove who they are. Verifiable credentials usually involve a third-party attestation, but they can also be self-attested.

[13]mentions that VCs support selective disclosure, so end users can prove claims about their identity without revealing more information than they intend and need for performing a specific action. VCs can express any information that a physical credential contains, but the usage of digital signatures from both the issuer and holder make them tamper-evident and more trustworthy to the verifier

## 3 Proposed System

We offer a Blockchain-based digital identity system that incorporates attribute-based data sharing, self-sovereign identity, decentralized identifiers, verified credentials, and allows identity owners to leverage existing trust connections. The system allows identity owners to show that they are who they say they are (authentication, i.e., login systems) and to make a specific claim based on third-party certifications (attestation, i.e., proof of education degree certificates). The suggested system's goal is to provide a framework that is remote-friendly, scalable, internationally usable, and by design provides both privacy and security.

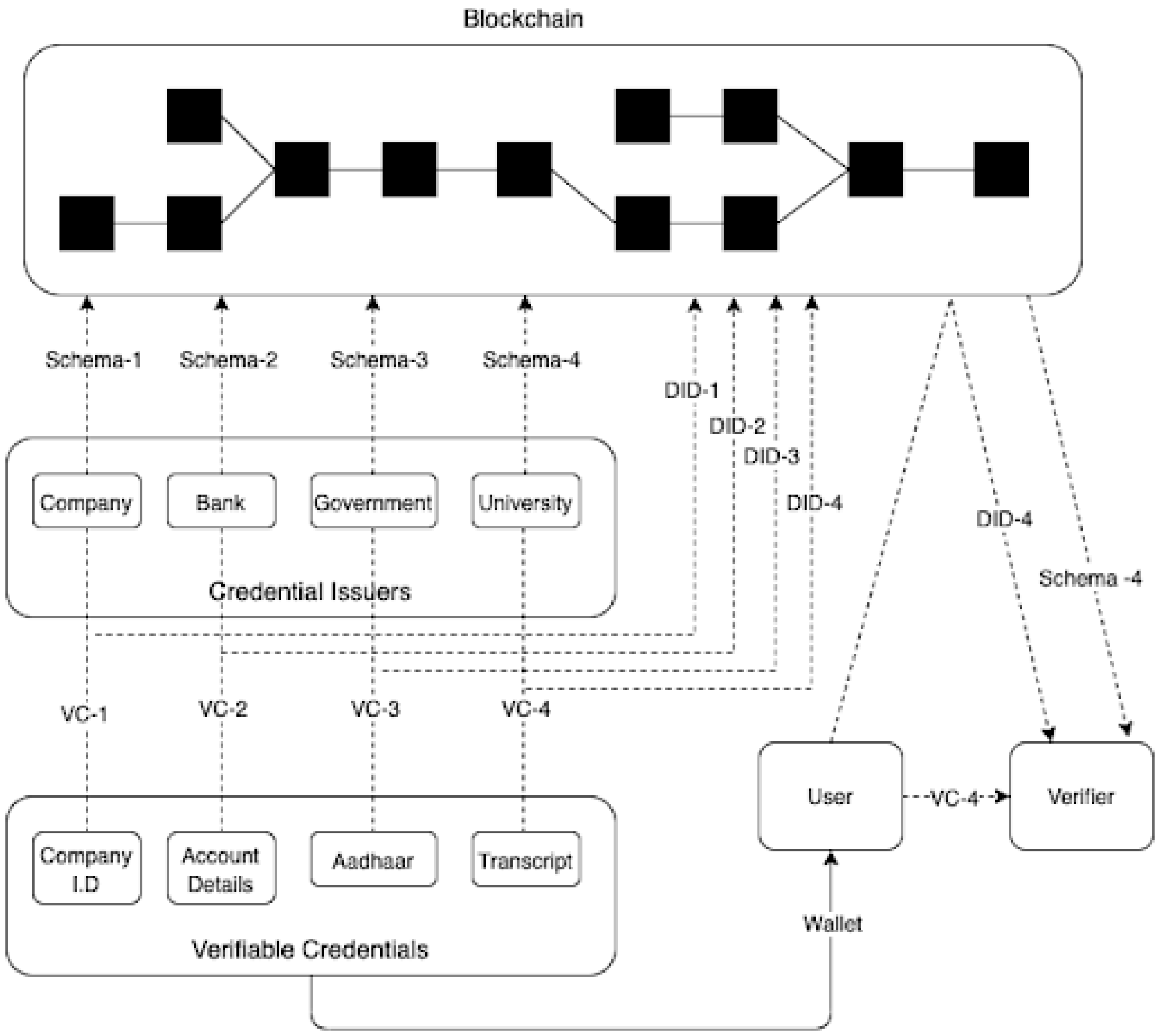


Figure 3: Proposed system. [13]

Figure 3 depicts the suggested solution's total workflow. Individuals and institutions are given a digital identity in the system, which is managed by Blockchain on a decentralized platform [13]. In the public ledger, the system does not keep any private user identifying data, even encrypted versions. It just has the evidence of transactions for the transactions.

## 3.1 Decentralized Identifiers

The Decentralized Identifiers (DIDs) defined in this specification are a new type of globally unique identifier. They are designed to enable individuals and organizations to generate their own identifiers using systems they trust [13]. Individuals, organizations, and things are given a standard, cryptographically verifiable, globally unique, and permanent identity via a decentralized identifier (DID). DIDs are totally controlled by the identity owner and are not reliant on central authority.

Since the generation and assertion of Decentralized Identifiers is entity-controlled, each entity can have as many DIDs as necessary to maintain their desired separation of identities, personas, and interactions. The use of these identifiers can be scoped appropriately to different contexts. They support interactions with other people, institutions, or systems that require entities to identify themselves, or things they control, while providing control over how much personal or private data should be revealed, all without depending on a central authority to guarantee the continued existence of the identifier.

DID employ public-key cryptography because each DID have an asymmetric key pair consisting of a public and a private key. The DID's private key is used to administer the DID's control. DIDs allow one identity owner to communicate with another identity owner through an encrypted private channel for the rest of their lives. DIDs are used by identity owners to identify themselves. Each DID is resolved to a DID document (DID descriptor object), which contains the cryptographic keys for the DID, publicly available metadata (if any) about the DID owner, and resource pointers for finding endpoints to initiate interactions with the DID owner [**Sporny2021a**].

### 3.1.1 Self-sovereign identity

Self-sovereign identity (SSI) is a way of having digital identity in which every user of the system owns and controls their own digital identity. The centralized authorities have no role in this case and accessibility of the data cannot be violated. The essential factors to be considered before usage of the SSI model.

- Digital identity owners have complete control on the identity data provided to the system.
- The system ensures the integrity, security, and secrecy of the owner's identity and central authority is not involved in the trust of the digital identity.
- Provides complete data accessibility. This means that identity owners can use their data wherever they wish (for instance, in accessing an online service.)
- Any changes to the data are visible to the respective roles, and the system maintains that transparency [13].

A DID is a simple text string consisting of three parts: 1) the did URI scheme identifier, 2) the identifier for the DID method, and 3) the DID method-specific identifier as shown in figure 4.

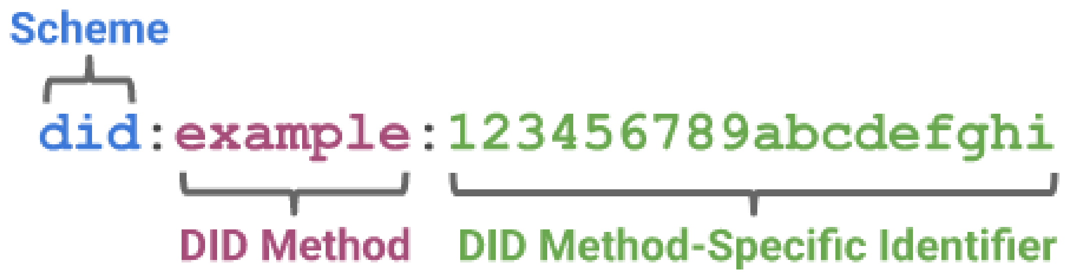
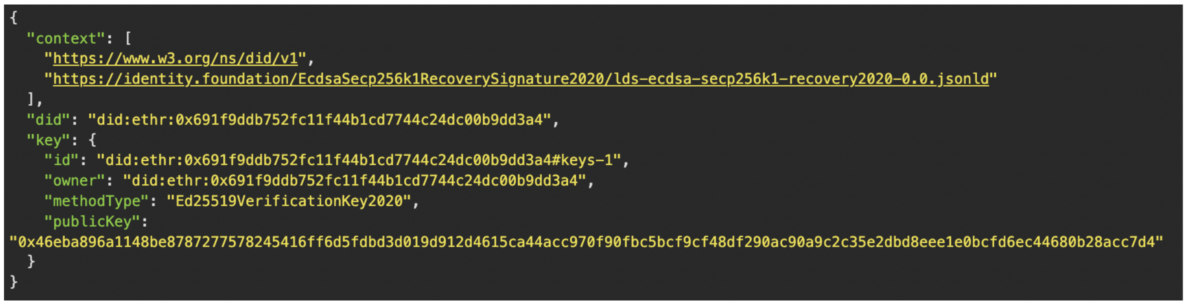

Scheme
did:example:123456789abcdefghi
DID Method
DID Method-Specific Identifier

Figure 4: DID Scheme Example [**Sporny2021a**]

The example DID above resolve to a DID document as shown in figure 5. A DID document contains information associated with the DID, such as ways to cryptographically authenticate a DID controller.



```
{
  "context": [
    "https://www.w3.org/ns/did/v1",
    "https://identity.foundation/EcdsaSecp256k1RecoverySignature2020/lds-ecdsa-secp256k1-recovery2020-0.0.jsonld"
  ],
  "did": "did:ethr:0x691f9ddb752fc11f44b1cd7744c24dc00b9dd3a4",
  "key": {
    "id": "did:ethr:0x691f9ddb752fc11f44b1cd7744c24dc00b9dd3a4#keys-1",
    "owner": "did:ethr:0x691f9ddb752fc11f44b1cd7744c24dc00b9dd3a4",
    "methodType": "Ed25519VerificationKey2020",
    "publicKey":
"0x46eba896a1148be8787277578245416ff6d5fdbd3d019d912d4615ca44acc970f90fbc5bcf9cf48df290ac90a9c2c35e2dbd8eee1e0bcfd6ec44680b28acc7d4"
  }
}
```

Figure 5: DID Document. [**Sporny2021a**]

## 3.2 Verifiable credentials

Credentials are digital documents that are used to confirm the identity of a particular individual, giving them the right to access documents pertaining to a particular field. These are documents that the individual makes use of on a day-to-day basis. They include driver's license, college transcripts, Aadhaar cards. Moreover, verifiable credentials are those credentials that are Machine-readable, privacy-preserving and cryptographically secure. Verifiable credentials support Self-sovereign identity, in which identity owners can collect and store credentials in an identity account or wallet and make use of them to prove who they claim to be. Verifiable credentials are mostly verified by a third party organization, but they can also be self-attested. Attestation is achieved by making use of the concept of digital signatures. The individual first signs his/her records using their private key, which is then converted into a verifiable credential by an individual having a DID [13]. A verifier can then verify a person's credential using the issuer's/attester's public key. The verifier makes sure to trust only those credentials that are signed by a reliable issuer.

The three essential components of verifiable credentials are:

- It is machine verifiable.
- It is secure and can't be tampered.
- Has been issued by a competent and reliable organization.

#### QR Code

Users can share his or her verifiable credentials, which have been issued by an issuer, to the service provider by scanning the QR code presented by the provider and then authenticating the transaction with his/her biometrics, thus verifying their claim.

Users can also gain access to their verifiable credentials, by initially authenticating themselves to the issuer and then scanning the QR code presented by the issuer. This way the user gets access to their verifiable credential which will then be signed by the issuer using their DID.

#### Biometric Authentication

Biometric information is unique and safe for every individual, and replication of biometric templates is challenging. Furthermore, it removes the factor of remembering the security password. Biometric data also solves the problem of repetition of passwords. This type of authentication makes its application systems more secure compared to traditional methods such as passwords.

We will be using the biometric authentication for the user app, where the user will need to authorize the sharing of verifiable credentials.

## 3.3 Sample Use Case for VJTI

In this section, we describe the proposed framework with a sample use case. The framework enables institutions to issue digitally signed documents to identity owners. The signed documents are in the form of verifiable credentials. Verifying parties and service providers are able to verify that the documents are original, not mutated, and signed by the issuers with the help of digital signatures from the blockchain.

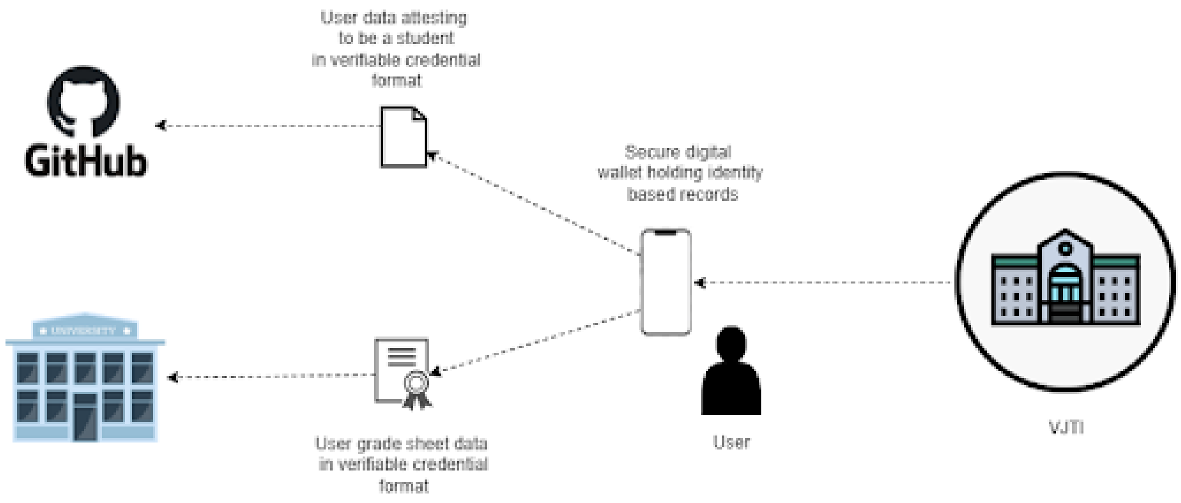

User data attesting to be a student in verifiable credential format
GitHub
Secure digital wallet holding identity based records
UNIVERSITY
User grade sheet data in verifiable credential format
User
VJTI

Figure 6: An example use case for VJTI.

Figure 6 shows an example use case regarding a student's identity, his or her mark sheets, and transcripts. In the example, a university (for example VJTI) can issue identity cards, mark sheets, and transcripts in the form of verifiable credentials. A student after authenticating itself can access these verifiable credentials using their digital wallet. The same student can use these verifiable credentials at multiple locations for example for accessing pro features for free on github.com, students can submit their identity issued by the university which can be verified by the application, here github.com. Students can also submit their issued mark sheets and transcripts to different institutions or universities as proof which is required for further education purposes.

## 4 Architecture

This proposed architecture section covers the architectural overview of the application. It also describes various components of the application. Various goals to be achieved are also covered in this section.

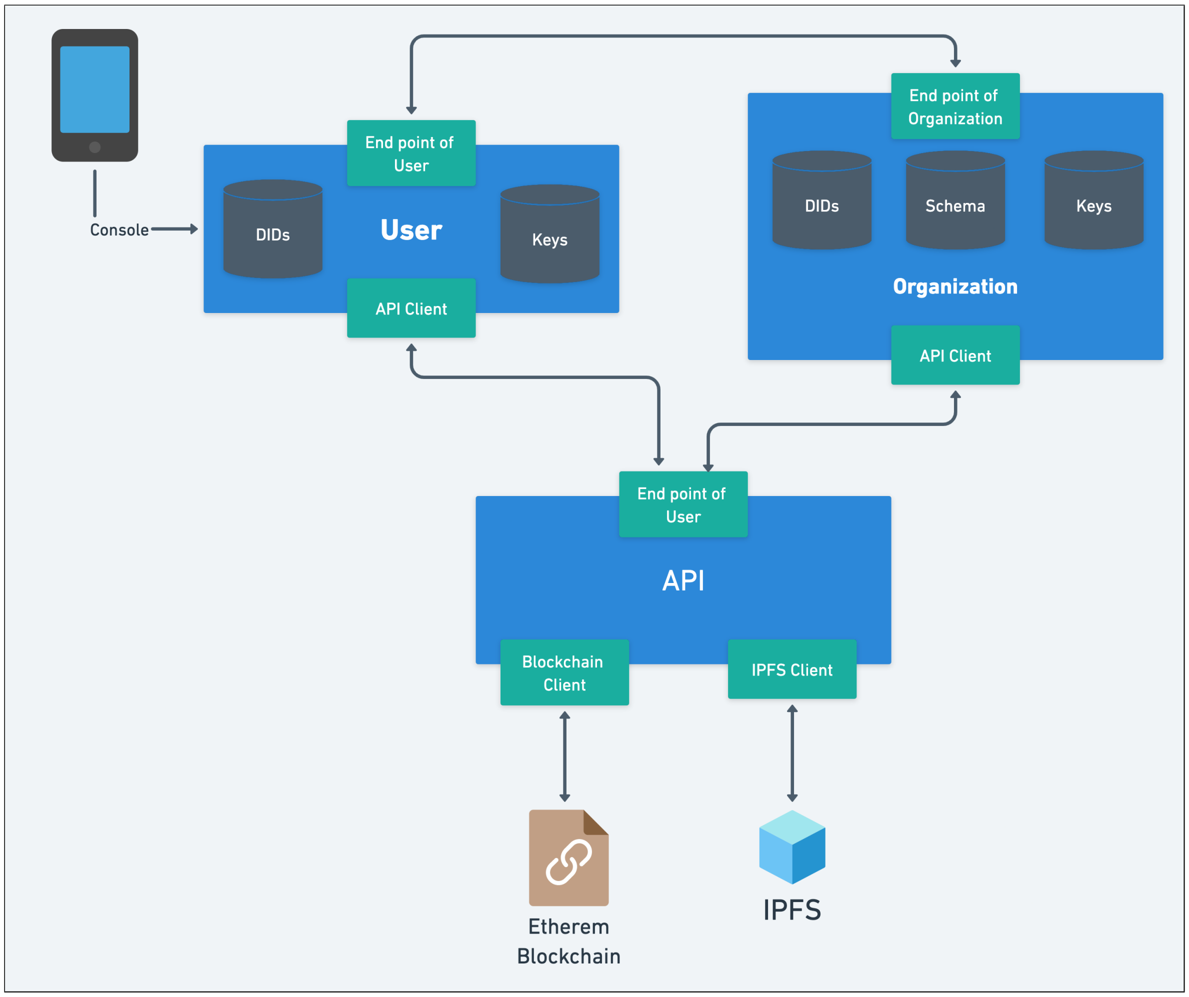


Figure 7: Application Architecture

The architecture diagram shown in the above figure shows various components and entities in the application.

- **Client App**: Client App is a digital wallet application which uses biometric functions to authenticate a user. All user's public keys and DIDs can be accessed using this client app.
- **User**: User entity consists of a DID and public keys. DID is the unique user ID. Public keys and policies are used to access the user Data. An API client is required to connect to the API. Client app is used to interact with all these processes.
- **Organization**: Organization entity is similar to user entity, except there is not a client app. Organization directly integrates with the API.
- **API**: The API connects the user, via Client App, and the Organization to the Ethereum Blockchain and IPFS. With help of API, onboarding into application ecosystem is made simple and easy.
- **Ethereum Blockchain**: The Ledger entity consists of all the transactions made by a particular user. The Ledger records information about events where users share their identity with others. It also stores the metadata of and the IPFS hexcode for credentials.

- **IPFS**: Interplanetary File System (IPFS), is used to store the actual Credential Data and Credential Schema Data. IPFS returns a hexcode corresponding to the data, that is in turn stored on the Ethereum Blockchain.

## 4.1 Components of Application

Here we give an overview of the architecture of the 2 main components of our proposed system i.e. DID (Decentralized Identifiers) and VC (Verifiable Credentials)

### 4.1.1 Decentralized Identifiers - DID

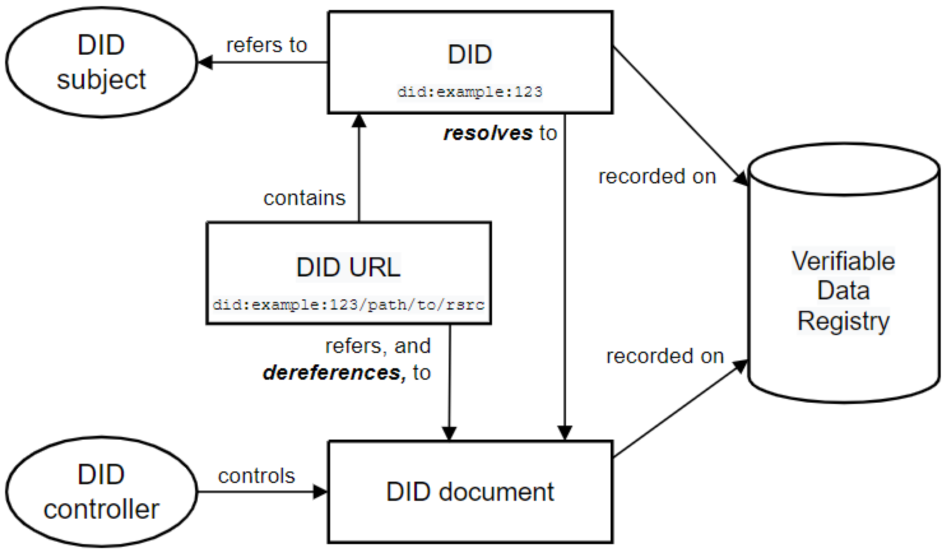


Figure 8: Overview of DID architecture and the relationship of the basic components [**Sporny2021a**]

**DIDs and DID URLs**: A Decentralized Identifier, or DID, is a URI composed of three parts: the scheme, a method identifier, and a unique, method-specific identifier specified by the DID method. DIDs are resolvable to DID document. A DID URL extends the syntax of a basic DID to add other standard URI components such as path, query, and fragment in order to locate a particular resource—for example, a cryptographic public key inside a DID document, or a resource external to the DID document.

**DID subject**: The subject of a DID is, the entity identified by the DID. The DID subject might also be the DID controller. Anything can be the subject of a DID: person, group, organization, thing, or concept.

**DID controllers**: The controller of a DID is the entity (person, organization, or autonomous software) that has the capability—as defined by a DID method—to make changes to a DID document. This capability is typically asserted by the control of a set of cryptographic keys used by software acting on behalf of the controller, though it might also be asserted via other mechanisms. Note that a DID might have more than one controller, and the DID subject can be the DID controller, or one of them.

**Verifiable data registries**: In order to be resolvable to DID document, DIDs are typically recorded on an underlying system or network of some kind. Regardless of the specific technology used, any such system that supports recording DIDs and returning data necessary to produce a DID document is called a verifiable data registry. Examples include distributed ledgers, decentralized file systems, databases of any kind, peer-to-peer networks, and other forms of trusted data storage.

**DID document**: DID document contain information associated with a DID. They typically express verification methods, such as cryptographic public keys, and services relevant to interactions with the DID subject.

**DID methods**: DID methods are the mechanism by which a particular type of DID and its associated DID document are created, resolved, updated, and deactivated.

**DID resolvers and DID resolution**: A DID resolver is a system component that takes a DID as input and produces a conforming DID document as output. This process is called DID resolution. The steps for resolving a specific type of DID are defined by the relevant DID method specification.

**DID URL dereference and DID URL dereferencing**: DID URL dereference is a system component that takes a DID URL as input and produces a resource as output. This process is called DID URL dereferencing [**Sporny2021a**].

### 4.1.2 Verifiable Credentials - VC

The following sections describe the core data model concepts of verified credentials, such as claims, credentials, and presentations, which form the foundation of this specification.

#### Claims

A claim by definition is a statement that is made about a subject. Whereas, a subject is an object about which claims can be asserted. Claims are conveyed using subject-property-value relationships.

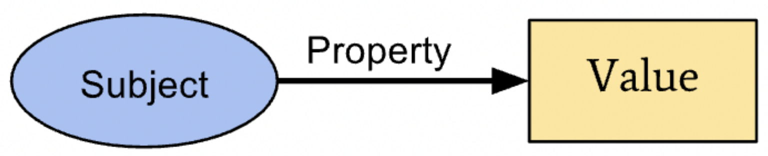


Figure 9: The basic structure of a claim [21]

Numerous statements can be expressed with the help of the above data model. For instance, A person graduating from a university can be expressed as shown in the below figure.

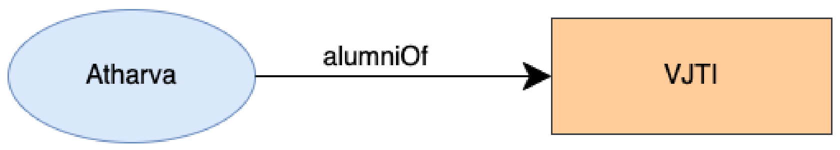


Figure 10: A basic claim, expressing that Atharva is an alumnus of ”VJTI” [21]
.

Claims made by individuals can be merged together to express information about a subject in the form of a graph. The below figure extends the claim made in the previous example by adding information claiming that Atharva knows Pratik, who is employed as a Professor at VJTI.

Up Till now, the concepts of a claim and a graph of information have been discussed. Further, we need

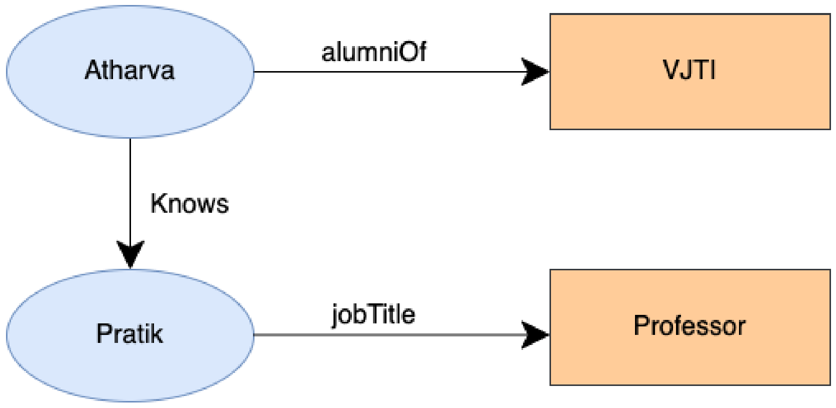


Figure 11: Multiple claims can be combined to express a graph of information [21]

.

to be able to trust the claims made. This will require additional information to be added to graph [21].

**Credentials**

Credentials are defined as a combination of one or more claims made by a particular entity. Identifiers and Information describing properties of credentials such as details of the issuer, expiry date and date of production of the credential, public key used for verification etc. may also be mentioned within the credential. Verifiable credentials contain tamper-proof claims and metadata that can prove who the issuer is using various cryptography methods.

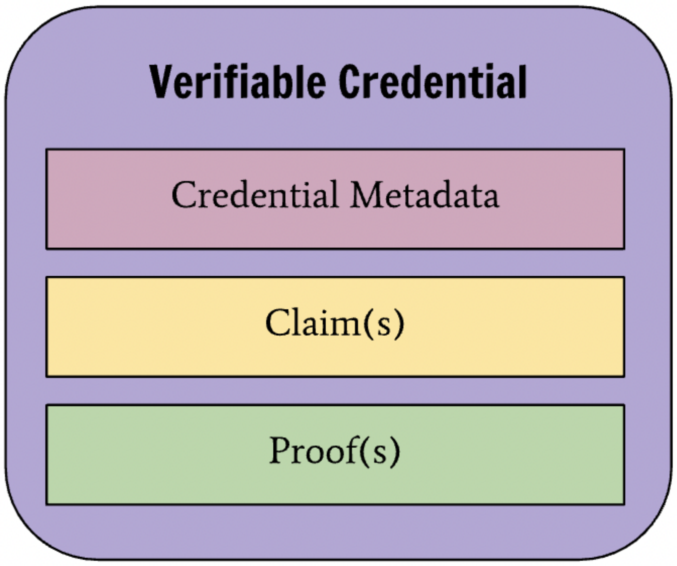


Figure 12: Basic components of a verifiable credential [21]

.

Examples of verifiable credentials include digital student identification cards, digital college transcripts, and digital educational certificates.

The above figure displays various components of verifiable credentials but doesn't go into much detail about how the claims are organized and connected to form the information graphs, which are further organized to form verifiable credentials [21].

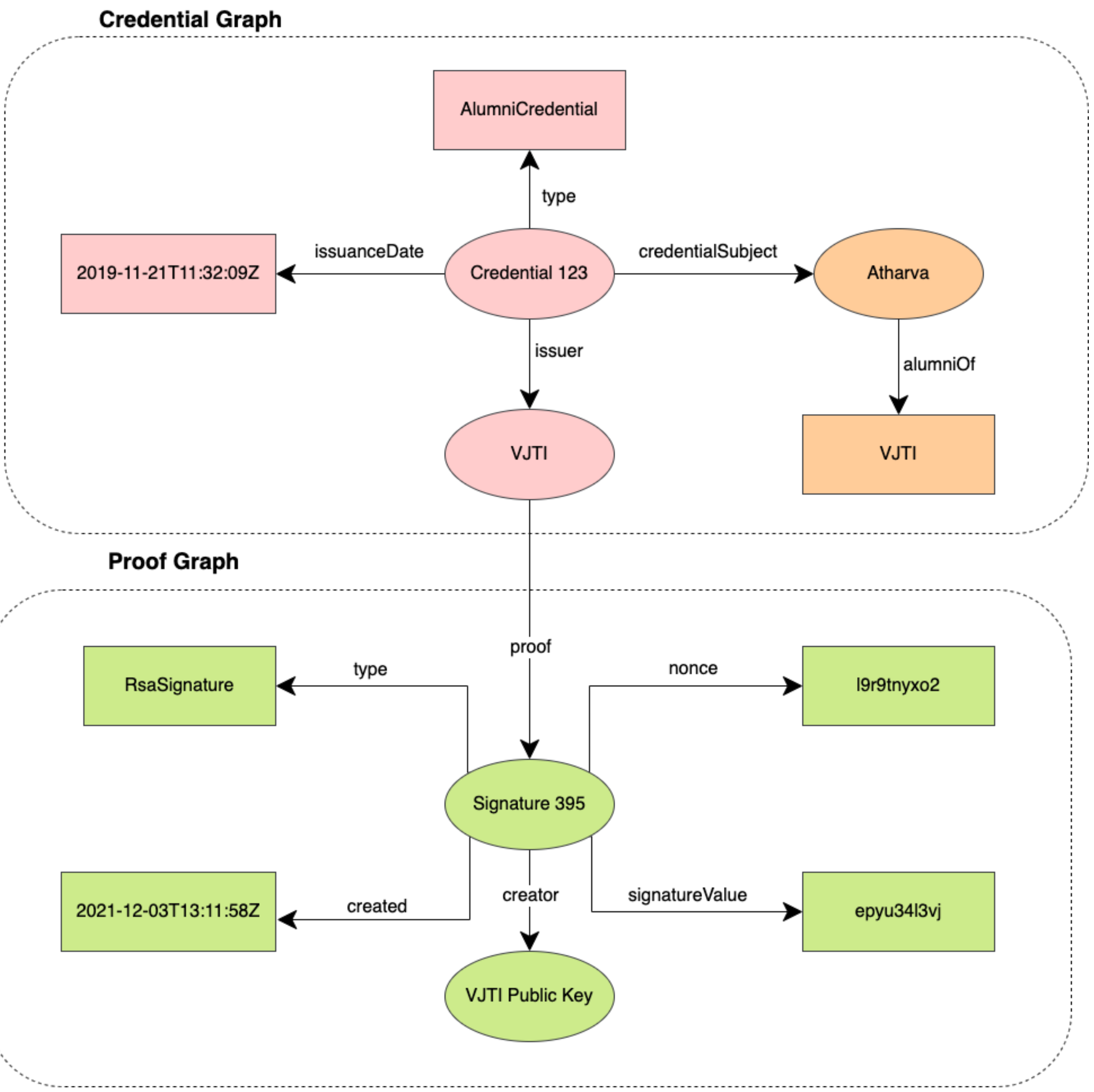


Figure 13: Information graphs associated with a basic verifiable credential [21]
.

The above figure gives a much better understanding and depiction of what makes a verifiable credential. The first graph in the shows the contents of a verified credential, containing credential metadata and claims. Whereas, the second graph shows how digital signatures make up the digital proof [21].

### 4.1.3 Mobile Wallet App

A smartphone wallet application is also needed to identify and authenticate a user. The mobile wallet can use the inbuilt biometric authentication system to verify the identity. This way, the user doesn't have to remember a password, which is a cumbersome task. By using biometric authentication, an additional layer of security is added. A pairwise-connection between the SSI IDP and the holder can be established via scanning a QR code displayed on the SSI IDP's website with an SSI smartphone wallet app. For starting the authentication, we could use HTTP cookies or a challenge response-based authentication scheme via scanning a QR, provided that no previous session information about the DID of the user exists [17]. The user is requested to provide proof to the SSI OIDC provider, which he/she can provide by giving consent using biometric authentication functionality provided on a smartphone.

Even to add new verifiable credentials to the data vault of the user, an SSI smartphone wallet application is required. The user is provided with a QR code that they could scan and then give consent by providing using biometric identification functions. The user could also be provided with a deep link to the mobile wallet application, which he or she can use to authenticate themselves and get access to the verifiable credentials.

## 4.2 Goals to be achieved

- **Transparency**: The private data of the user is secure, but there is total transparency to the user on how and where their data is being used, as they can grant and revoke permissions easily.
- **Security**: Biometric authentication along with the public key infrastructure of blockchain is used to provide high security. This way, only the user and the entities to whom the user has given permission can access the data.
- **Traceability**: User has complete control over their data and can grant and revoke permissions regarding their private data.
- **Ease of Use**: Since the user needs to use biometric authentication to use the application, he/she does not need to remember different passwords or reuse the same password [16].

# 5 System Implementation

This section covers the implementation of the proposed system and the architecture. It initially covers a sample use case, then goes through the tech stack used for the system implementation. Further, a demo implementation of the system is described. Finally, other features that the application offers are discussed in-depth.

## 5.1 Use Case

The Use case considered here is that of a student who wants to prove their identity on a website such as GitHub, which provides premium services to students. The student in this use case proves to the receiver, here GitHub, that he is a student by first receiving his verifiable credential from his college and then sharing it to the GitHub website for verification. Therefore, for the entirety of the section, we will assume the student to be a user, the College to be the issuer and GitHub to be the receiver. In the use case, the college website issues transcript as the verifiable credential. The student is then made the owner of the credential, giving only the user to control who has access to the credential. The student sends this credential information to GitHub website as a proof that the user is in fact a student.

This is only one use case, since the implementation of API service and mobile application, many organizations and users can be onboarded on to the application ecosystem. This leads to various different use cases.

## 5.2 Tech stack

A tech stack is a collection of technologies used to build an application. In this section, the technologies used to build this application are discussed.

### 5.2.1 React

ReactJS is an open-source JavaScript library that is used for building user interfaces specifically for single-page applications. It's used for handling the view layer for web and mobile apps. React also allows us to create reusable UI components [22]. React allows developers to create large web applications that can change data, without reloading the page. The main purpose of React is to be fast, scalable, and simple. It works only on user interfaces in the application. This corresponds to the view in the MVC template. It can be used with a combination of other JavaScript libraries or frameworks, such as Angular JS in MVC.

ReactJS was made use of in developing the frontend of the issuer website and receiver website.

### 5.2.2 React Native

React Native is a JavaScript framework for writing real, natively rendering mobile applications for iOS and Android. It's based on React, Facebook's JavaScript library for building user interfaces, but instead of targeting the browser, it targets mobile platforms. Through this, web developers can now write mobile applications that look and feel truly "native," all from the comfort of a JavaScript library that is already known. Plus, because most of the code you write can be shared between platforms, React Native makes it easy to simultaneously develop for both Android and iOS [31]. React Native was used in developing the mobile app, where users can connect to the main API. The Users have the ability to share credentials with receivers, as well as to revoke their access. We have further leveraged the biometric capabilities of the mobile phone through this app to make it easy for the users to grant permission and verify their identity for the various operations that the mobile app may perform on their behalf.

### 5.2.3 Express

Express is a minimal and flexible Node.js web application framework that provides a robust set of features to develop web and mobile applications. It facilitates the rapid development of Node based Web applications. Following are some core features of Express framework  Allows setting up middlewares to respond to HTTP Requests, defines a routing table which is used to perform different actions based on HTTP Method and URL and Allows to dynamically render HTML Pages based on passing arguments to templates [29].

Express in NodeJS was made use of in developing the Backend API of the issuer website and receiver website.

### 5.2.4 IPFS

IPFS is a distributed system for storing and accessing files, websites, applications, and data.We have made use of IPFS to store the schema of credentials created by issuers, as well as to store the credentials of all users.

### 5.2.5 Ganache

Ganache is used for setting up a personal Ethereum Blockchain for testing your Solidity contracts. It provides more features when compared to remix [27]. Ganache was used for testing purpose by deploying the Ethereum contract on a personal blockchain

### 5.2.6 Ethereum

Ethereum is a decentralized global software platform powered by blockchain technology. It is most commonly known for its native cryptocurrency, ether, or ETH. Ethereum can be used by anyone to create any secured digital technology they can think of. It has a token designed for use in the blockchain network, but it can also be used by participants as a method to pay for work done on the blockchain [26].

Ethereum is designed to be scalable, programmable, secure, and decentralized.

### 5.2.7 MongoDB

MongoDB is an open-source document database and leading NoSQL database. MongoDB is written in C++.MongoDB is a cross-platform, document oriented database that provides, high performance, high availability, and easy scalability. MongoDB works on concept of collection and document [28].

MongoDB was used as the database for the issuer website and the receiver website.

## 5.3 Workflow

In this section, the workflow for the sample use case is covered in detail. The use case covers a user signing up on our mobile application. The receiving a verifiable credential from its college organization. Finally, the users shares the received credential using the mobile app.

### 5.3.1 Creating DID

#### Wallet User

Every user will be using the app and will be identified by their DID.The app have various sections like View Did document, get credential, sent credential and export data.

The user first has to create their DID, which is done by typing in their name as shown in the figure below. This data is sent to the main API service. Upon creating the API, a message containing the user is then displayed on the mobile app

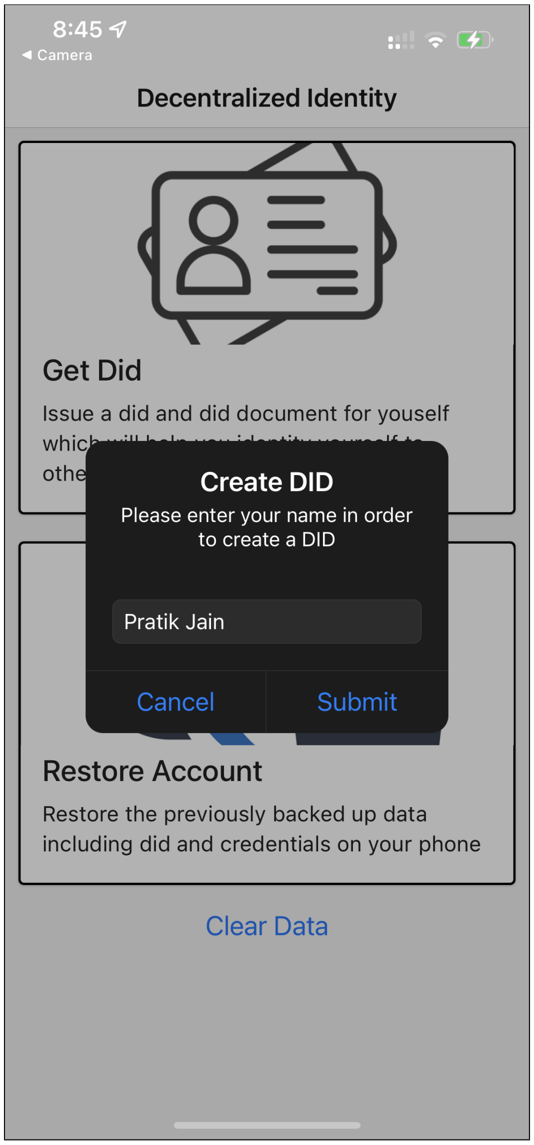
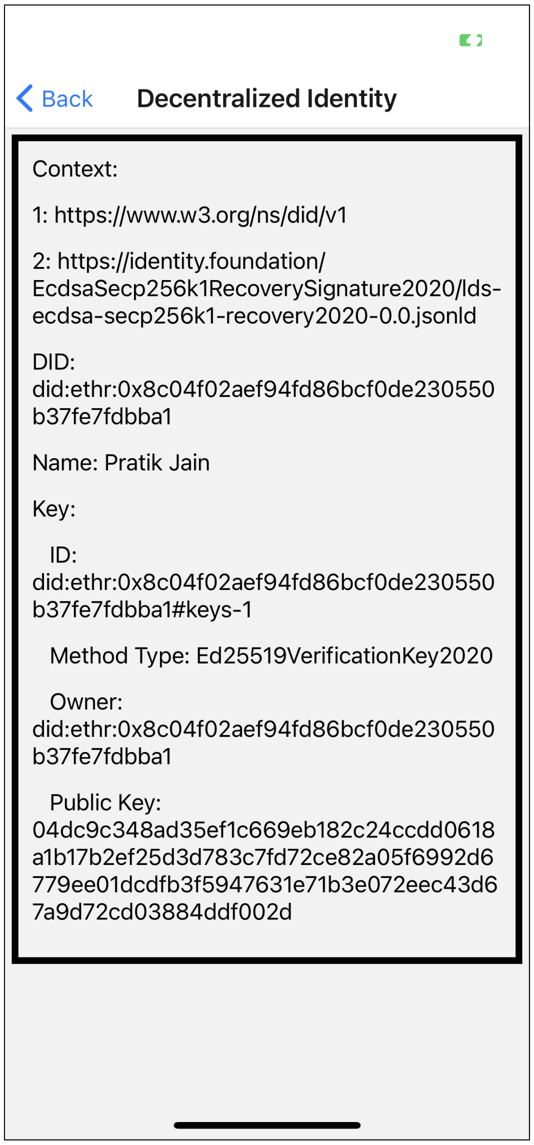

8:45
Camera
Decentralized Identity
Get Did
Issue a did and did document for youself
which
othe
Create DID
Please enter your name in order to create a DID
Pratik Jain
Cancel
Submit
Restore Account
Restore the previously backed up data including did and credentials on your phone
Clear Data

Figure 14: Wallet App Creating DID

Once the DID is created, it is stored on the blockchain. The user can view this DID document by send a *GET* request to the main API service. The main API service sends back the DID document. The DID document for the user can be viewed in the diagram below.



Context:

1: https://www.w3.org/ns/did/v1

2: https://identity.foundation/EcdsaSecp256k1RecoverySignature2020/lds-ecdsa-secp256k1-recovery2020-0.0.jsonld

DID: did:ethr:0x8c04f02aef94fd86bcf0de230550b37fe7fdbba1

Name: Pratik Jain

Key:

ID: did:ethr:0x8c04f02aef94fd86bcf0de230550b37fe7fdbba1#keys-1

Method Type: Ed25519VerificationKey2020

Owner: did:ethr:0x8c04f02aef94fd86bcf0de230550b37fe7fdbba1

Public Key: 04dc9c348ad35ef1c669eb182c24ccdd0618a1b17b2ef25d3d783c7fd72ce82a05f6992d6779ee01dcdfb3f5947631e71b3e072eec43d67a9d72cd03884ddf002d

Figure 15: Wallet App DID Document Display Screen

## Organizations

For any organization that wants to make an exchange of credentials with the user through the main API, they are required to first register themselves with the main API. The registration process is done by sending a *GET* request to the /createDID endpoint containing the following information:

- Ethereum Address
- Public Key
- Name of the Organization

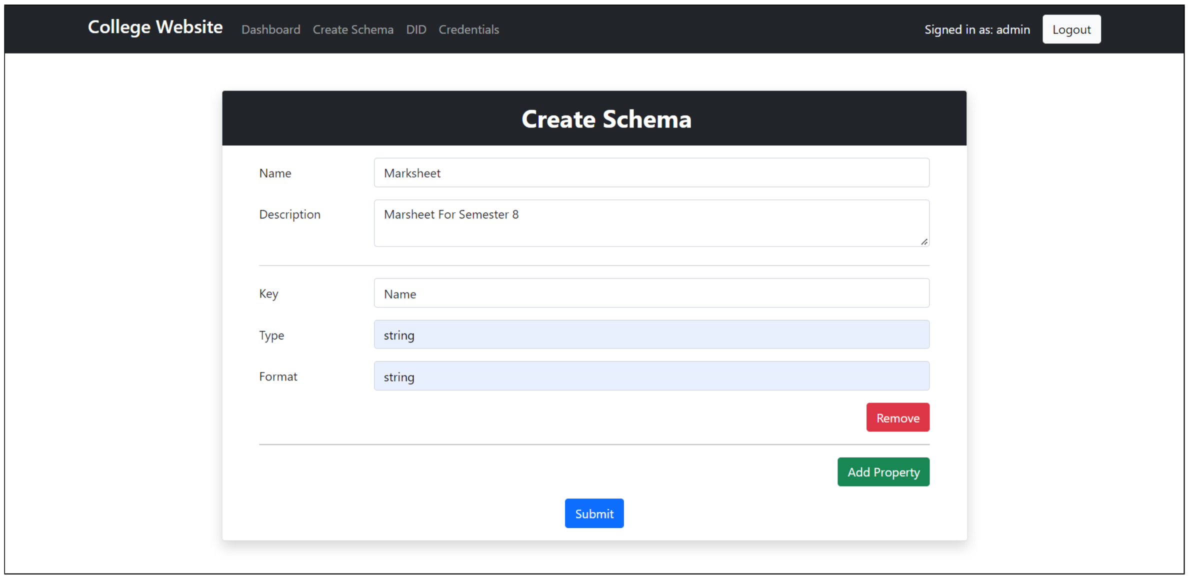
Based on the data provided by the organization, the main API then creates a DID for the organization and sends it as a part of the response

### 5.3.2 Creating Schema

Admin can create new schemas for credentials the issuer wants to issue.The Schema is the format for a particular credential. With Schema standardization across credentials and organizations can be created. It contains the following Information:

- Name: This attribute holds the name of the schema
- Description: This attribute defines the purpose and meaning of a schema
- Attribute: This is the data contained in the credential that is specific to a user

Figure below shows creation of a schema for the issuer website. The figure is a for demonstration purposes only. For creating the schema, the organization needs to call the API *createSchema* while providing their DID, name for the schema, description of the schema, and the properties of the given schema.
The API requests add the credential schema to IPFS and creates a unique DID to identify the schema.



| | |
|---|---|
| Name | Marksheet |
| Description | Marsheet For Semester 8 |
| Key | Name |
| Type | string |
| Format | string |

Figure 16: Issuer Website Admin Creating Schema

.

### 5.3.3 Issuing Credentials

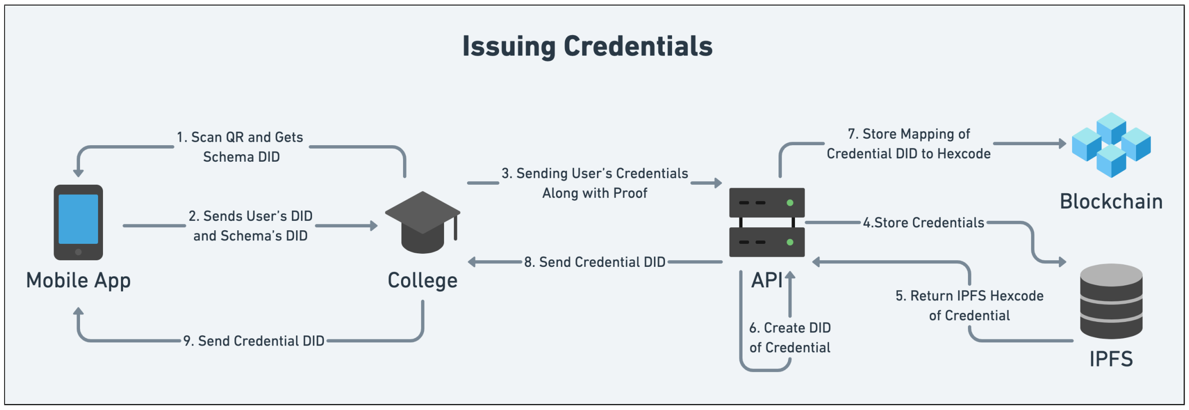

Issuing Credentials
1. Scan QR and Gets Schema DID
2. Sends User's DID and Schema's DID
3. Sending User's Credentials Along with Proof
4.Store Credentials
5. Return IPFS Hexcode of Credential
6. Create DID of Credential
7. Store Mapping of Credential DID to Hexcode
8. Send Credential DID
9. Send Credential DID
Mobile App
College
API
Blockchain
IPFS

Figure 17: Issuing Credentials Workflow

#### Scan QR Code and Gets Schema DID

The first step to issuing credentials is for the user to scan the QR code presented on the issuer website. The QR code in the figure below encodes the following information

- **URL**: It is the URL to which the mobile app sends the required data.
- **Schema DID**: This is the Schema using which the credential is to be issued.
- **User unique ID**: This is the unique ID of the User's account on the Issuer website. This helps to fetch the user data from Issuer's database

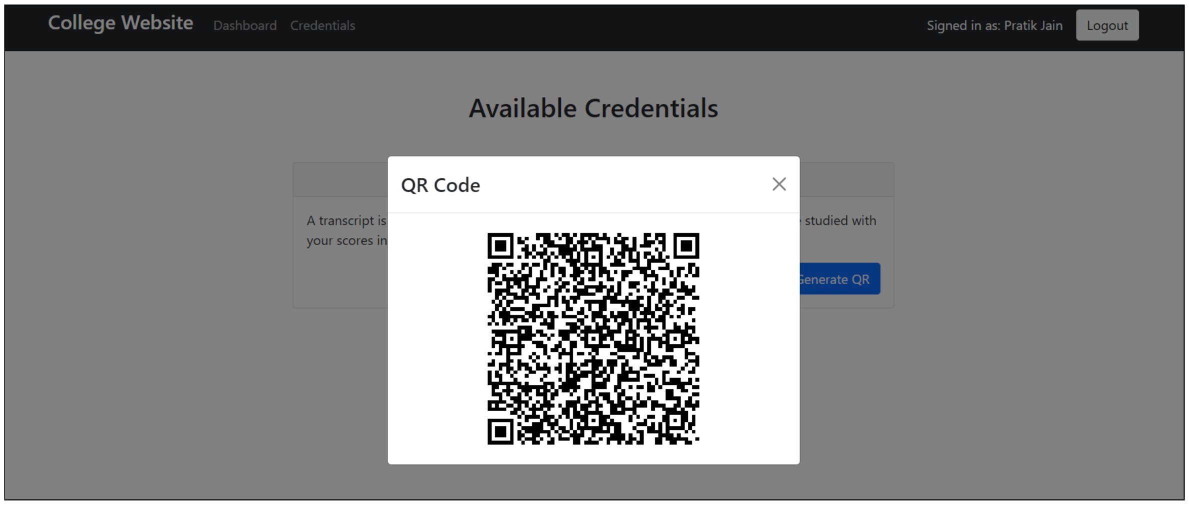


Figure 18: Issuer Website User QR Code for Credential

#### Sends User's DID and Schema's DID

Once the User Scans the QR. The mobile app makes a request to the URL encoded in the QR code, containing the User's DID and Schema DID.

### Sending User's Credentials along with Proof

Upon receiving a request from the mobile app for creating the User's credentials, the issuer creates the Credential following the schema of the Specified Schema DID. The Issuer then sends the Credential along with its proof to the *addCredential* route of the Main API.

### Store Credentials on IPFS and getting Its Hexcode

The Main API firsts verifies the proof, confirming the identity of the issuer. Then sends the Credentials to IPFS to store it, which in turn sends the Hexcode corresponding to the Credential.

### Create DID of Credential

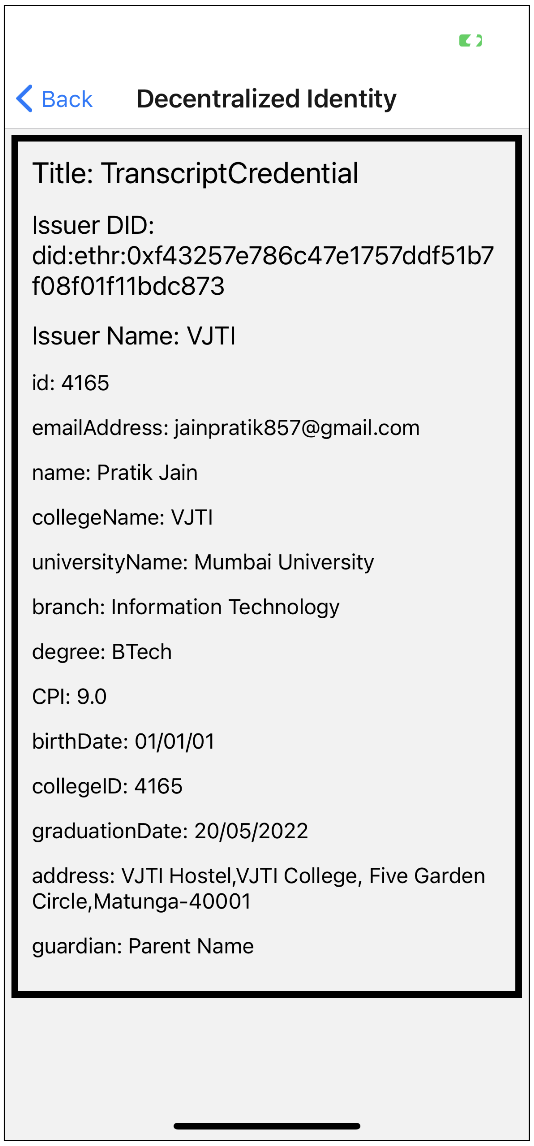
The Main API creates the DID for the Credential it has generated.

### Store Mapping of Credential DID to Hex Code

The Mapping of Credential DID to its Hexcode is stored on the Blockchain. This is done so that the API is able to retrieve the Credentials from IPFS for a particular Credential DID for future situations. The user is also set as the owner of the credential, giving the user complete control over who has access to the credential.

### Send Credential DID from API to Mobile App

Now that the credential is stored and its DID generated, the API can send the credential's DID to the Issuer along with a success message. The Issuer then passes on this information back to the user via the mobile app. Now, the user can use the credential DID to view the entire credential on the mobile app by just sending a request to the main API containing the credential's DID.
A sample credential is shown in the figure below. This credential is created after scanning the QR code shown in the previous diagram.



Title: TranscriptCredential

Issuer DID:
did:ethr:0xf43257e786c47e1757ddf51b7f08f01f11bdc873

Issuer Name: VJTI

id: 4165

emailAddress: jainpratik857@gmail.com

name: Pratik Jain

collegeName: VJTI

universityName: Mumbai University

branch: Information Technology

degree: BTech

CPI: 9.0

birthDate: 01/01/01

collegeID: 4165

graduationDate: 20/05/2022

address: VJTI Hostel,VJTI College, Five Garden Circle,Matunga-40001

guardian: Parent Name

Figure 19: Wallet App Credential Display Screen

### 5.3.4 Sharing Credentials

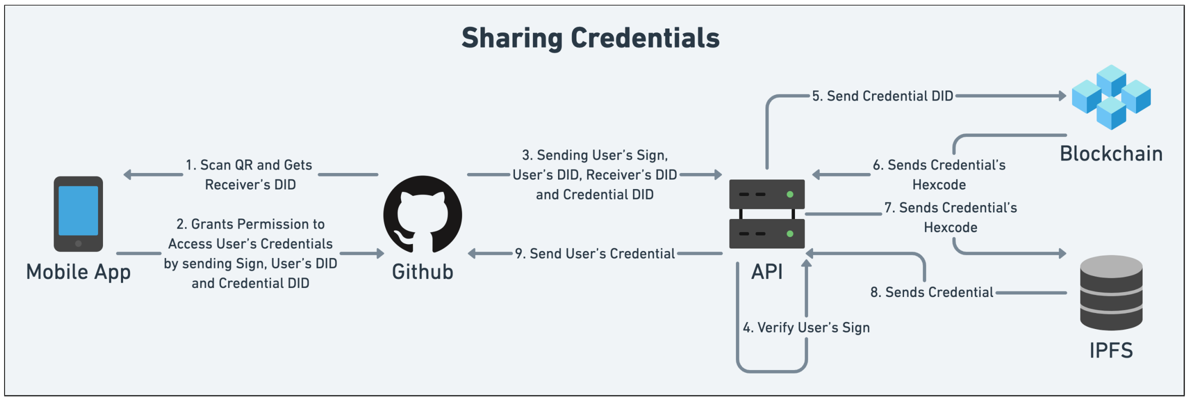
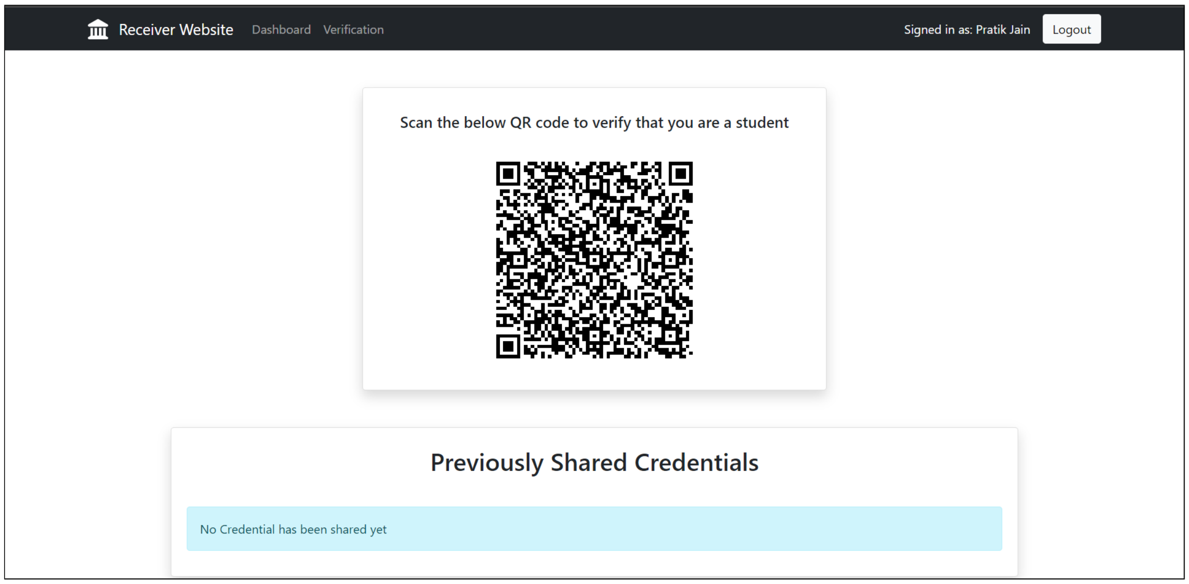

Sharing Credentials
Mobile App
1. Scan QR and Gets Receiver's DID
2. Grants Permission to Access User's Credentials by sending Sign, User's DID and Credential DID
Github
3. Sending User's Sign, User's DID, Receiver's DID and Credential DID
9. Send User's Credential
API
4. Verify User's Sign
5. Send Credential DID
6. Sends Credential's Hexcode
7. Sends Credential's Hexcode
8. Sends Credential
Blockchain
IPFS

Figure 20: Sharing Credentials Workflow

#### Scan QR and Get Receiver's DID

The user scans the QR code shown on the website of the receiver, in this case GitHub. By scanning the QR code, the user gets the Receiver's DID, its user ID on receiver website and a URL on which the user will send the required data.



Scan the below QR code to verify that you are a student

Previously Shared Credentials

No Credential has been shared yet

Figure 21: Receiver Website Share Credential QR Code

**Grants Permission to Access User's Credentials**

Data to grant permission to access User's Credentials are sent on the URL received after scanning the QR code. The data that is sent includes hash, sign of the hash, User's DID, and Credential's DID. All this data is automatically sent to the Receiver organization after the user selects which document to share, as shown in figure 22.

**Sending Received Information**

The organization receiving the user credential just forwards all this information along with its own DID to the API service.

**Verify Signature**

The API service first verifies the signature of the user to determine whether the user has given permission to access the credential.

**Send Credential DID**

After the signature has been successfully verified, the Credential DID is sent to the blockchain along with the Receiver DID, and User DID. After the owner of the credential is verified, as the given user, then this adds the Receiver's DID to a list of DIDs that have access to the given Credential DID.

**Receive Credential Hexcode**

The blockchain stores the mapping from Credential DID to Credential Hexcode, which is generated when the credential is added to IPFS. This Hexcode is sent back to the API service.

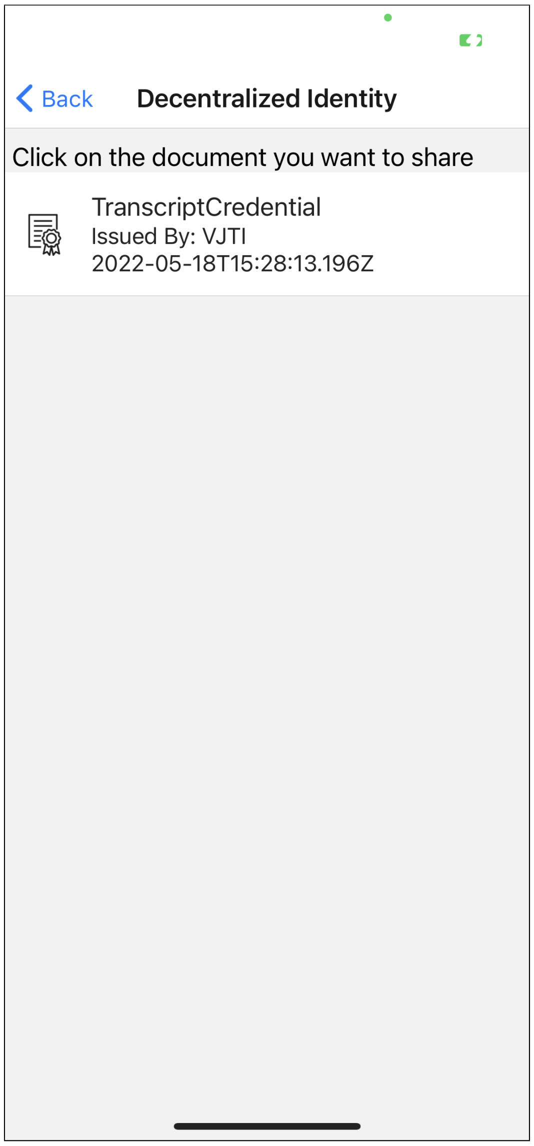
**Send Hexcode to IPFS and Receive Credential**

After receiving the Hexcode from the blockchain, this Hexcode is used to fetch data from IPFS. IPFS returns the data corresponding to the Hexcode back to the service, which is then converted to a JSON object.

**Send Credential**

The created JSON object is sent back to the receiver as a response to the API call made in step number three. The user is notified on successful sharing of the credential.

The user can revoke access to the credential anytime through the mobile app.



Click on the document you want to share

TranscriptCredential
Issued By: VJTI
2022-05-18T15:28:13.196Z

Figure 22: Wallet App Credential Share select screen

## 5.4 Other Features

There are multiple features that are offered in our application, which are mentioned below. These features add ease of use for the user by providing a lot of necessary functionality.

### 5.4.1 Import/Export Account

The user is provided with the feature to import or export their account. So in case if user's device changes, he/she does not need to go through all the hassle to get access to their credentials again by scanning QR codes at various websites.

The user can export their data by selecting the export data option. The user will be prompted to add a password to encrypt their data, as shown in the figure below. So no other entity gets access to the user's sensitive data. The encrypted data is sent to the API service, which then adds this data to IPFS. The received Hexcode corresponding to the encrypted data is then added to the blockchain. In the blockchain, this hexcode is then mapped to the user's did and the user is notified on success.

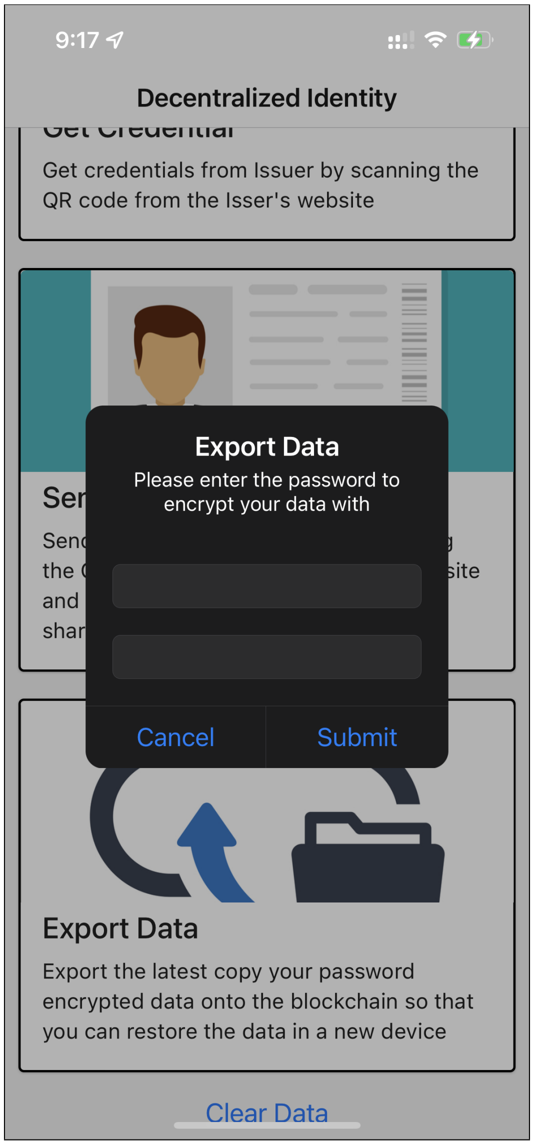

9:17
Decentralized Identity
Get Credential
Get credentials from Issuer by scanning the QR code from the Isser's website
Export Data
Please enter the password to encrypt your data with
Cancel
Submit
Export Data
Export the latest copy your password encrypted data onto the blockchain so that you can restore the data in a new device
Clear Data

Figure 23: Wallet App Account Export Screen

When the user needs to import the data, the user can select the import data option. By selecting this option, the user is prompted to fill in the DID for the account that needs to be imported and the password that was used to encrypt the data, as shown in figure 24. The DID is then sent to the API service, which in turn fetches the mapped IPFS hexcode. The encrypted data is then fetched from IPFS and sent to the mobile application. On the application, the password that the user had entered is used to decrypt the data. If decryption is successful, then the data is imported successfully.

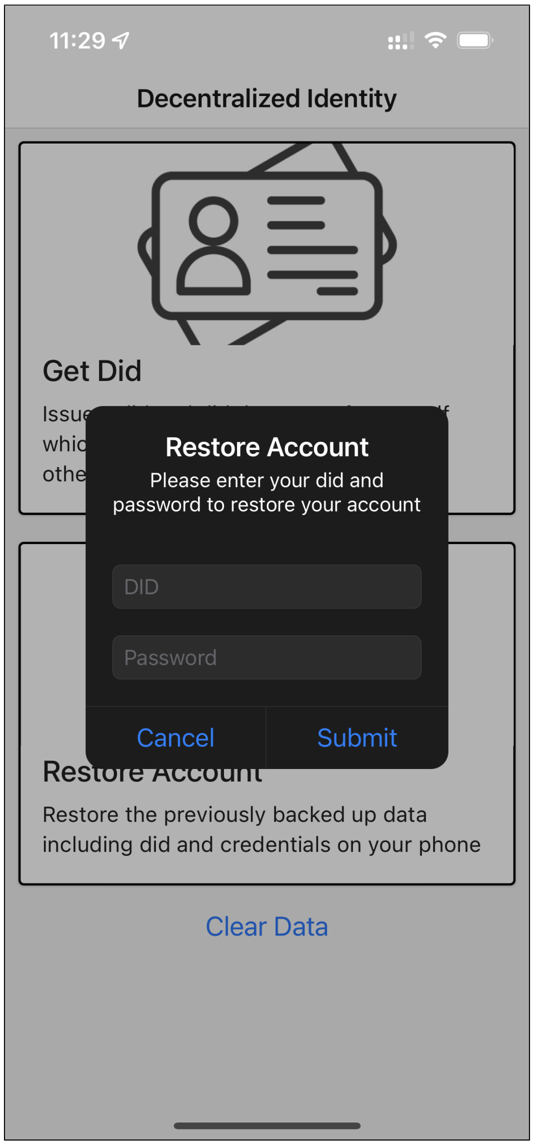
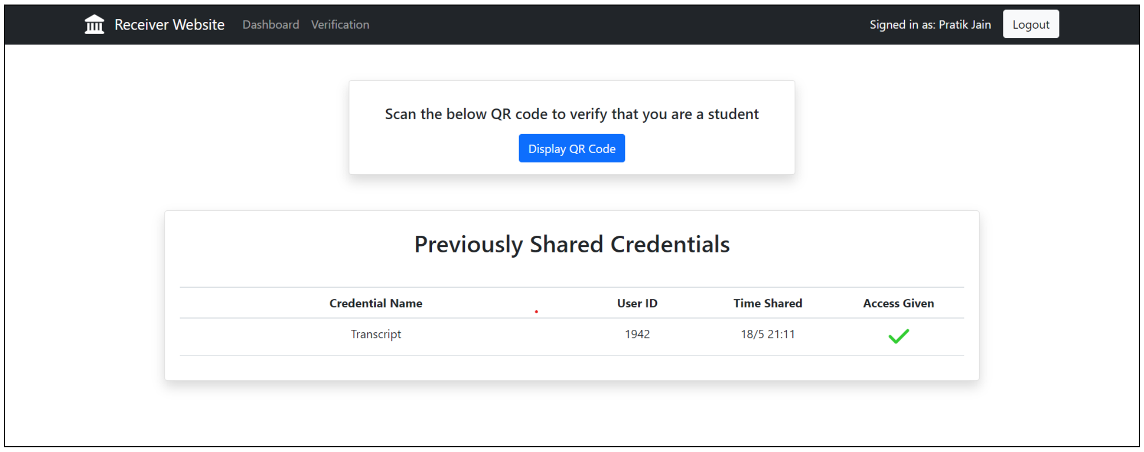

11:29
Decentralized Identity
Get Did
Restore Account
Please enter your did and password to restore your account
DID
Password
Cancel
Submit
Restore Account
Restore the previously backed up data including did and credentials on your phone
Clear Data

Figure 24: Importing account details from existing account

### 5.4.2 Access Revoke

The user has complete control over its credentials. The user can share credentials to any organization or entity it wants, and can also revoke access whenever it deems necessary.
The below figure represents a receiver which has access to user's credentials. The figure is for demonstration purpose only. The receiver organization, in this case GitHub, can make an API *GET* request to *getCredential* route of the main API service. The function rehttps://www.overleaf.com/project/6283d4858294371e647ef4be credential DID, receiver DID, unique hash, and sign of the hash using receiver's private key. This API request will return the credential if the receiver has access to it, or it will return access denied.



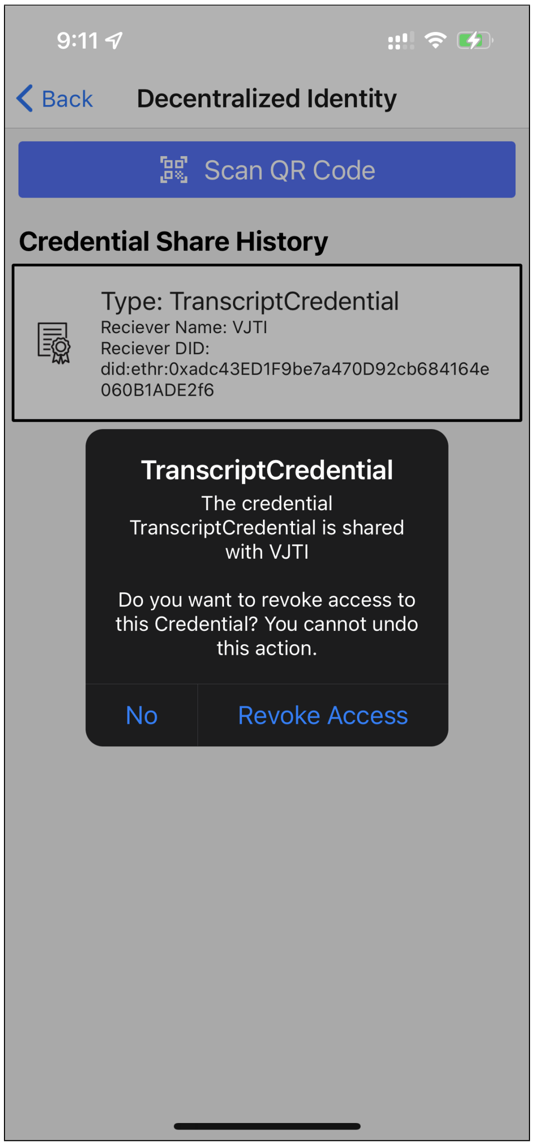
Scan the below QR code to verify that you are a student

Previously Shared Credentials

| Credential Name | User ID | Time Shared | Access Given |
|---|---|---|---|
| Transcript | 1942 | 18/5 21:11 | ✓ |

Figure 25: Receiver Website Shared Credential History

On the mobile application, the user is shown the list of credential share history. It contains the list of organizations the user has shared what credentials to. With one tap, the user can revoke access of any organization on any credential that the user owns. The user is shown a success message on successful removal of access.



Credential Share History

Type: TranscriptCredential
Reciever Name: VJTI
Reciever DID:
did:ethr:0xadc43ED1F9be7a470D92cb684164e060B1ADE2f6

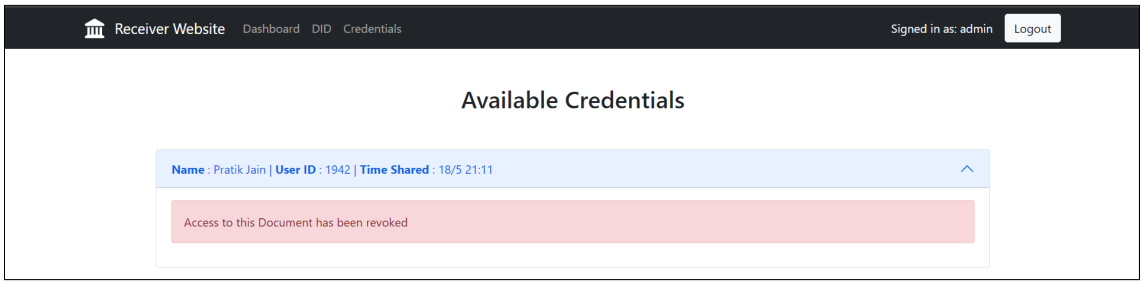
TranscriptCredential

The credential TranscriptCredential is shared with VJTI

Do you want to revoke access to this Credential? You cannot undo this action.

Figure 26: Wallet App Access Revoke for Credential

Once the access to the credential has been revoked, the receiver organization won't have access to the credential document again. The *getCredential* API route will give access denied now. The user will need to go through the whole process of sharing the credential with the receiver, as was shown in the sharing credential workflow, to give the receiver access to the credential again.



Available Credentials

**Name** : Pratik Jain | **User ID** : 1942 | **Time Shared** : 18/5 21:11

Access to this Document has been revoked

Figure 27: Receiver Website Credential Access Revoked

## 6 Result and Discussion

We developed an application prototype for decentralized identity management and verifiable credentials using blockchain. Important features such as issuing, sharing, and revoking access to credentials were achieved using IPFS and blockchain. We implemented a three tier architecture to provide greater accessibility and usability.

All entities, user or organization, can easily on board onto our application. For a user to onboard, the user only needs to install our application. For an organization, they'll need to integrate and use the APIs provided in our application. The organization has complete control over which APIs to use and integrate, but for all the features, every API needs to be assimilated.

The three tier architecture contains the mobile application that we developed, for facilitating easy onboarding of user to our ecosystem. The APIs used to interact with blockchain and IPFS. It also allows easy integration and onboarding of organizations on our application. The third and final tier refers to the Ethereum blockchain and IPFS, which is used to hold user and credential information.

Blockchain itself cannot be considered for storing complete credential data, as the gas fees for storing data on blockchain is very high and only a limited amount of data can be stored in a particular block on the Ethereum blockchain. Therefore, IPFS is used along with blockchain. On IPFS the actual credential data is stored, while on the blockchain only the metadata is stored. Even though blockchain improves transparency, there should exist trust between entities.

# 7 Conclusion and Future Work

## 7.1 Conclusion

Our work explored the concepts and implementation of decentralized identity and verifiable credentials using the Ethereum blockchain. Current Identity management systems are centralized and there are concerns with respect to users privacy and the question of security of users personal data. Blockchain provides a solution to this problem in the way of decentralized identity. The user has complete control over their credentials and identity. Since blockchain is decentralized and immutable, the control of the data of a particular user is only in their hand.

An organization can easily issue a verifiable credential for a user which is stored on IPFS, a decentralized file system. The credential can only be accessed by the user, and the entities the user gives permission to access it. The access permissions to a credential are stored on the blockchain and can only be updated by the user. This way decentralized identity with complete control over the credential data is achieved using blockchain.

The code for the whole project can be viewed on: `https://github.com/BTechProject2022`

## 7.2 Future Work

Future work for this involves integration of zero knowledge proofs. This way, only the necessary data stored in a credential needs to be shared. This will offer even more control over the data to the user. Also, monetization of APIs need to be conducted. But this monetization will be only be applied to the APIs related to issuing and sharing credentials. Furthermore, sharing credentials between users via mobile app. This provides ease of sharing credentials between individual users.

## References

[1] Ronald L Rivest, Adi Shamir, and Leonard Adleman. “A method for obtaining digital signatures and public-key cryptosystems”. In: *Communications of the ACM* 21.2 (1978), pp. 120–126.

[2] Gavin Wood et al. “Ethereum: A secure decentralised generalised transaction ledger”. In: *Ethereum project yellow paper* 151.2014 (2014), pp. 1–32.

[3] Jong-Hyouk Lee. “BIDaaS: Blockchain based ID as a service”. In: *IEEE Access* 6 (2017), pp. 2274–2278.

[4] Xueping Liang et al. “Integrating blockchain for data sharing and collaboration in mobile healthcare applications”. In: *2017 IEEE 28th annual international symposium on personal, indoor, and mobile radio communications (PIMRC)*. IEEE. 2017, pp. 1–5.

[5] Elli Androulaki et al. “Hyperledger fabric: a distributed operating system for permissioned blockchains”. In: *Proceedings of the thirteenth EuroSys conference*. 2018, pp. 1–15.

[6] Zhonglin Chen et al. “A security authentication scheme of 5G ultra-dense network based on block chain”. In: *IEEE Access* 6 (2018), pp. 55372–55379.

[7] Gaby G Dagher et al. “Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology”. In: *Sustainable cities and society* 39 (2018), pp. 283–297.

[8] *Delegated Proof of Stake Explained*. https://academy.binance.com/en/articles/delegated-proof-of-stake-explained. Accessed: 2021-12-03. Nov. 2018.

[9] Zhimin Gao et al. “Blockchain-based identity management with mobile device”. In: *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems*. 2018, pp. 66–70.

[10] Julian Roos. “Identity management on the blockchain”. In: *Network* 105 (2018).

[11] Gauhar Ali et al. “Blockchain based permission delegation and access control in Internet of Things (BACI)”. In: *Computers & Security* 86 (2019), pp. 318–334.

[12] Jamila Alsayed Kassem et al. “DNS-IdM: A blockchain identity management system to secure personal data sharing in a network”. In: *Applied Sciences* 9.15 (2019), p. 2953.

[13] Mehmet Aydar, Serkan Ayvaz, and Salih Cemil Cetin. “Towards a Blockchain based digital identity verification, record attestation and record sharing system”. In: *arXiv preprint arXiv:1906.09791* (2019).

[14] Benedict Faber et al. “BPDIMS: A blockchain-based personal data and identity management system”. In: (2019).

[15] Zheng Zhao and Yuan Liu. “A Blockchain based Identity Management System Considering Reputation”. In: *2019 2nd International Conference on Information Systems and Computer Aided Education (ICISCAE)*. IEEE. 2019, pp. 32–36.

[16] Carlisle Adams. “A privacy-preserving Blockchain with fine-grained access control”. In: *Security and Privacy* 3.2 (2020), e97.

[17] Zoltán András Lux et al. “Distributed-Ledger-based Authentication with Decentralized Identifiers and Verifiable Credentials”. In: *arXiv e-prints* (2020), arXiv–2006.

[18] Sonia Alice George, Arunita Jaekel, and Ikjot Saini. “Secure identity management framework for vehicular ad-hoc network using blockchain”. In: *2020 IEEE Symposium on Computers and Communications (ISCC)*. IEEE. 2020, pp. 1–6.

[19] Luke Conway. *Blockchain Explained*. https://www.investopedia.com/terms/b/blockchain.asp. Accessed: 2021-12-03. Nov. 2021.

[20] Lyle Daly. *What Is Proof of Stake (PoS) in Crypto?* https://www.fool.com/investing/stock-market/market-sectors/financials/cryptocurrency-stocks/proof-of-stake/. Accessed: 2021-12-03. Sept. 2021.

[21] David Chadwick Manu Sporny Dave Longley. *Verifiable Credentials Data Model v1.1*. https://www.w3.org/TR/vc-data-model/. Accessed: 2021-12-03. Nov. 2021.

[22] Nitin Pandit. *What And Why React.js*. https://www.c-sharpcorner.com/article/what-and-why-reactjs/. Accessed: 2022-05-19. Feb. 2021.

[23] *PROOF-OF-WORK (POW)*. https://ethereum.org/en/developers/docs/consensus-mechanisms/pow/. Accessed: 2021-12-03. Nov. 2021.

[24] Tykn. *Blockchain Identity Management: The Definitive Guide (2021 Update)*. https://tykn.tech/identity-management-blockchain/. Accessed: 2021-12-03. May 2021.

[25] Phillip J Windley. "Sovrin: An Identity Metasystem for Self-Sovereign Identity". In: *Frontiers in Blockchain* (2021), p. 30.

[26] JAKE FRANKENFIELD. *Ethereum*. https://www.investopedia.com/terms/e/ethereum.asp. Accessed: 2022-05-19. May 2022.

[27] *Ethereum - Ganache for Blockchain*. https://www.tutorialspoint.com/ethereum/ethereum_ganache_for_blockchain.htm. Accessed: 2022-05-19.

[28] *MongoDB Tutorial*. https://www.tutorialspoint.com/mongodb/index.htm. Accessed: 2022-05-19.

[29] *Node.js - Express Framework*. https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm. Accessed: 2022-05-19.

[30] *Sovrin identity for all*. https://w3c-ccg.github.io/sovrin/. Accessed: 2021-12-03.

[31] *What Is React Native?* https://www.oreilly.com/library/view/learning-react-native/9781491929049/ch01.html. Accessed: 2022-05-19.