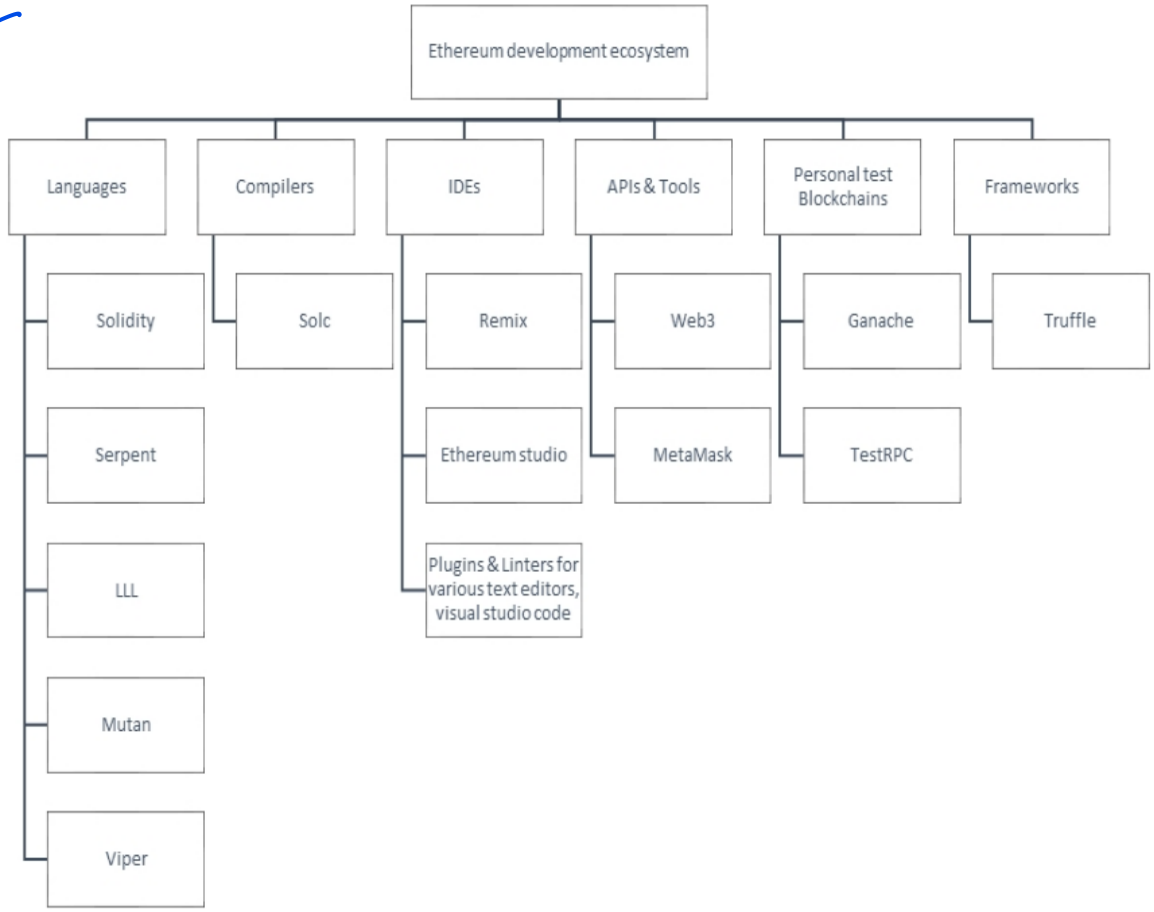
Module -5

Development Tools and Framework

There are a number of tools available for Ethereum development. The following diagram shows the taxonomy of various development tools, clients, IDEs, and development frameworks for Ethereum.



Taxonomy of Ethereum development ecosystem components

Languages

Smart contracts can be programmed in a variety of languages for Ethereumblockchain. There are five languages that can be used in order to write contracts:

- **Mutan**: This is a Go-style language, which was deprecated in early 2015 and is no longer used.
- **LLL**: This is a Low-level Lisp-like Language, hence the name LLL. This is also not used anymore.
- **Serpent**: This is a simple and clean Python-like language. It is not used for contract development anymore and not supported by the community anymore.

- **Solidity**: This language has now become almost a standard for contract writing for Ethereum.
- **Vyper**: This language is a Python-like experimental language that is being developed to bring security, simplicity, and auditability to smart contract development

Compilers

Compilers are used to convert high-level contract source code into the format that the Ethereum execution environment understands. The Solidity compiler is the most common one in use .

Solidity compiler (solc)

solc converts from a high-level solidity language into Ethereum Virtual Machine (EVM) bytecode so that it can be executed on the blockchain by EVM.

Integrated Development Environments (IDEs)

There are various IDEs available for Solidity development. Most of the IDEs are available online and are presented via web interfaces. Remix (formerly browser Solidity) is the most commonly used IDE for building and debugging smart contracts.

Remix : Remix is the web-based environment for the development and testing of contracts using Solidity. It is a feature-rich IDE which does not run on live blockchain; in fact, it is a simulated environment in which contracts can be deployed, tested, and debugged.

Various features, such as transaction interaction, options to connect to JavaScript VM, configuration of execution environment, debugger, formal verification, and static analysis, are available. They can be configured to connect to execution environments such as JavaScript VM, injected Web3— where Mist, MetaMask, or a similar environment has provided the execution environment—or Web3 provider, which allows connection to the locally running Ethereum client (for example, geth) via IPC or RPC over HTTP (Web3 provider endpoint). Remix also has a debugger for EVM which is very powerful and can be used to perform detailed level tracing and analysis of the EVM bytecode.

Tools and libraries

There are various tools and libraries available for Ethereum.

1. EthereumJS : At times, it is not possible to test on the testnet and mainnet is obviously not a place to test the contracts. Private net can be time-consuming to set up at times. EthereumJS' TestRPC comes in handy when quick testing is required and no proper testnet is available. It uses EthereumJS to simulate the Ethereumgeth client behavior and allows for faster development testing.

2. Ganache: Ganache is a private Ethereum blockchain environment that allows to you emulate the Ethereum blockchain so that you can interact with smart contracts in your own private blockchain . Here are some features that Ganache provides:

- uses a user-friendly graphical user interface to see transaction and blocks and relevant details.

Ganache is a useful tool for Ethereum developers who want to test and debug their contracts and applications in a simulated environment before deploying them to the main Ethereum network.

- is fully working Byzantium enabled personal blockchain which is used to provide a local testing environment for blockchains.
- Displays blockchain log output
- Provides advanced mining control
- Built-in block explorer
- Ethereum blockchain environment
- Ganache has a desktop application as well as a command-line tool
- Based on a JavaScript implementation of the Ethereum blockchain

MetaMask is a software cryptocurrency wallet used to interact with the Ethereum blockchain. It allows users to access their Ethereum wallet through a browser extension or mobile app, which can then be used to interact with other decentralized applications and Ethereum BC

3. MetaMask :MetaMask allows interaction with Ethereum blockchain via Firefox and Chrome browsers. It injects a web3 object within the running websites' JavaScript context which allows immediate interface capability for DApps. This injection allows DApps to interact directly with the blockchain.

   MetaMask also allows account management. This acts as a verification method before any transaction is executed on the blockchain. The user is shown a secure interface to review the transaction for approval or rejection before it can reach the target blockchain.
   MetaMask allows account management and also records all transactions for these accounts.

Truffle is a popular development framework for building decentralized applications (dapps) on the Ethereum blockchain. It provides a suite of tools and resources to help developers create, test, and deploy smart contracts and decentralized applications on the Ethereum network.

4. Truffle: Truffle is a development environment that makes it easier and simpler to test and deploy Ethereum contracts.
   - provides contract compilation and linking
   - provides an automated testing framework using Mocha and Chai.
   - easier to deploy the contracts to any private net, public, or testnet Ethereum blockchain.
   - asset pipeline is provided, which makes it easier for all JavaScript files to be processed, making them ready for use by a browser.

Solidity language

Solidity is a domain-specific language of choice for programming contracts in Ethereum. Its syntax is closer to both JavaScript and C.

Solidity is quite easy to use. This is the most widely used language available for programming contracts currently.

It is a statically typed language, which means that variable type checking in Solidity is carried out at compile time. Each variable, either state or local, must be specified with a type at compile time. Any validation and checking is completed at compile time and certain types of bugs, such as interpretation of data types, can be caught earlier in the development cycle instead of at runtime, which could be costly, especially in the case of the blockchain / smart contracts paradigm. Other features of the language include inheritance, libraries, and the ability to define composite data types.

Solidity is also called a contract-oriented language. In Solidity, contracts are equivalent to the concept of classes in other object-oriented programming languages.

Solidity has two categories of data types: **value types** and **reference types.**

Value types:

Boolean: This data type has two possible values, true or false.

Integers:

Keyword	Types	Details
`int`	Signed integer	`int8` to `int256`, which means that keywords are available from `int8` up to `int256` in increments of 8, for example, `int8`, `int16`, `int24`.
`uint`	Unsigned integer	`uint8`, `uint16`, ... to `uint256`, unsigned integer from 8 bits to 256 bits. The usage is dependent on the requirements that how many bits are required to be stored in the variable.

Wei is the smallest denomination of ether

Address: This data type holds a 160-bit long (20 byte) value. This type has several members that can be used to interact with and query the contracts. These are:

- Balance: The balance member returns the balance of the address in Wei.
- Send: This member is used to send an amount of ether to an address (Ethereum's 160-bit address) and returns true or false depending on the result of the transaction
- Call functions: The call, callcode, and delegatecall calls are provided in order to interact with functions that do not have ABI.
- Array value types (fixed size and dynamically sized byte arrays): Solidity has fixed size and dynamically sized byte arrays. Fixed size keywords range from **bytes1** to **bytes32**, whereas dynamically sized keywords include **bytes** and **string**.

Literals: These used to represent a fixed value.

- Integer literals are a sequence of decimal numbers in the range of 0-9.
- String literals specify a set of characters written with double or single quotes.
- Hexadecimal literals are prefixed with the keyword hex and specified within double or single quotation marks.
- Enums: This allows the creation of user-defined types

Function types: There are two function types: internal and external functions.

- Internal functions can be used only within the context of the current contract.
- External functions can be called via external function calls. A function in solidity can be marked as a constant. Constant functions cannot change anything in the contract; they only return values when they are invoked and do not cost any gas.

Reference types:

These types are passed by reference. These are also called complex types.

- **Arrays** represent a contiguous set of elements of the same size and type laid out at a memory location. The concept is the same as any other programming language. Arrays have two members named length and push.
- **Structs**:These constructs can be used to group a set of dissimilar data types under a logical group.
- **Data location** specifies where a particular complex data type will be stored. Depending on the default or annotation specified, the location can be storage or memory. This is applicable to arrays and structs and can be specified using the storage or memory keywords.**Calldata** is another memory location that is used to store function arguments.
- **Mappings** are used for a key to value mapping.

Global variables: Solidity provides a number of global variables that are always available in the global namespace.
Control structures available in solidity language are if...else, do, while, for, break, continue, and return. They work exactly the same as other languages such as C-language or JavaScript.
Events in Solidity can be used to log certain events in EVM logs. These are quite useful when external interfaces are required to be notified of any change or event in the contract. These logs are stored on the blockchain in transaction logs.
Inheritance is supported in Solidity. The **is** keyword is used to derive a contract from another contract.
Libraries are deployed only once at a specific address and their code is called via **CALLCODE** or **DELEGATECALL**opcode of the EVM. The key idea behind libraries is code reusability.
Functions in Solidity are modules of code that are associated with a contract. Functions are declared with a name, optional parameters, access modifier, optional constant keyword, and optional return type.

Layout of a Solidity source code file

pragma can be used to specify the version of the compatible compiler as, for example, in the following:

```
pragma solidity ^0.5.0
```

Import in Solidity allows the importing of symbols from the existing Solidity files into the current global scope.

```
import "module-name";
```

Comments can be added in the Solidity source code file .Multiple line comments are enclosed in /* and */, whereas single line comments start with //

An example Solidity program is as follows, showing the use of `pragma`, `import`, and comments:

```
1  pragma solidity ^0.4.0; //specify the compiler version
2  /*
3  This is a simple value checker contract that checks the value
4  provided and returns boolean value based on the condition
5  expression evaluation.
6  */
7  import "dev.oraclize.it/api.sol";
8  contract valuechecker {
9      uint price=10;
10     //This is price variable declare and initialized with value 10
11     event valueEvent(bool returnValue);
12     function Matcher (uint8 x) returns (bool)
13     {
14         if ( x >= price)
15         {
16             valueEvent(true);
17             return true;
18         }
19     }
20 }
```

Sample Solidity program as shown in Remix IDE

Hyperledger Fabric is an open-source blockchain platform that was developed by the Linux Foundation's Hyperledger Project. It is designed for building enterprise-grade blockchain applications, with a focus on modularity, scalability, and security. Hyperledger Fabric provides a flexible and extensible architecture for building decentralized applications, known as "smart contracts." These smart contracts can be used to facilitate, verify, and enforce the negotiation or execution of a contract. In addition, Hyperledger Fabric also supports a consensus mechanism, which ensures that all participants in the network have the same view of the shared ledger.
One of the key features of Hyperledger Fabric is its permissioned network, which means that only approved participants can join the network and transact on it.

Hyperledger is an open source project that provides a framework for building enterprise-grade blockchain applications. It was initiated by the Linux Foundation and is supported by major technology companies, financial institutions, and supply chain organizations.
Hyperledger offers a variety of tools and resources for developers to build decentralized applications (dApps) with specific features and functions.

Hyperledger

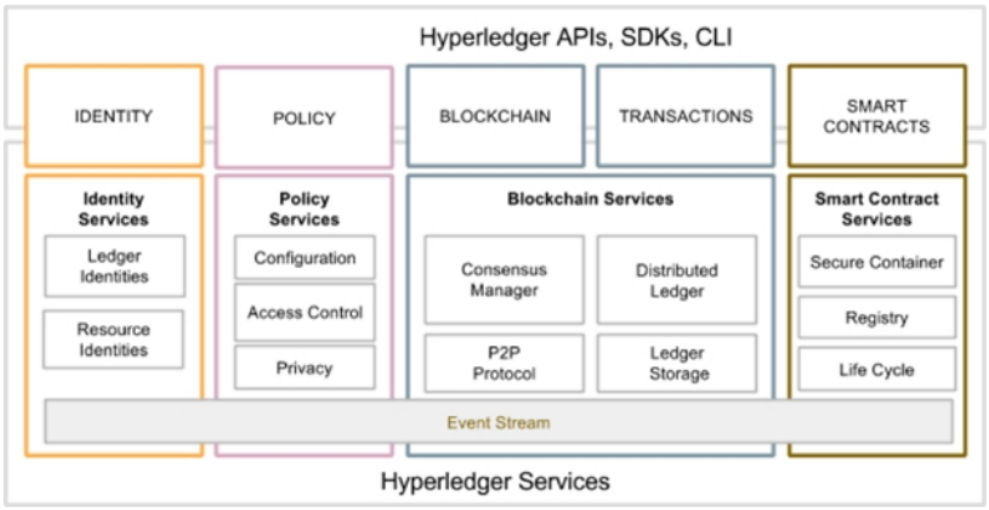
Hyperledger is a multi-project open source collaborative effort hosted by **The Linux Foundation**, created to advance cross-industry blockchain technologies.
There are two categories of projects under Hyperledger. The first is blockchain projects and the second category is relevant tools or modules that support these blockchains. Currently, there are five blockchain framework projects under the Hyperledger umbrella: Fabric, Sawtooth Lake, Iroha, Burrow, and Indy. Under modules, there are the Hyperledger Cello, Hyperledger Composer, Hyperledger Explorer, and Hyperledger Quilt.

Fabric: The fabric is a blockchain project that was proposed by **IBM and DAH** (Digital Asset Holdings). It is an open source project from the Linux Foundation, is the modular blockchain framework and de facto standard for enterprise blockchain platforms.
Fabric can be defined as a collection of components providing a foundation layer that can be used to deliver a blockchain network. There are various types and capabilities of a fabric network, but all fabrics share common attributes such as immutability and are consensus-driven.

This blockchain framework implementation is intended to provide a foundation for the development of blockchain solutions with a modular architecture. It is based on a pluggable architecture where various components, such as consensus engine and membership services, can be plugged into the system as required. It also makes use of container technology which is used to run smart contracts in an isolated contained environment.

Fabrics can also be designed either to be private or public and can allow the creation of multiple business networks. As an example, Bitcoin is an application that runs on top of its fabric . **The aim of Hyperledger Fabric is to develop a permissioned distributed ledger.**

The reference architecture
The reference architecture consists of various components that form a business blockchain.



Reference architecture - source: Hyperledger whitepaper

First is identity, that provides authorization, identification, and authentication services under membership services.

Then is the policy component, which provides policy services.

After this, ledger and transactions come, which consists of the distributed ledger, ordering service, network protocols, and endorsement and validation services. This ledger is updateable only via consensus among the participants of the blockchain network.

Finally, we have the smart contracts layer, which provides chaincode services in Hyperledger and makes use of secure container technology to host smart contracts. We will see all these in more detail in the Hyperledger Fabric section shortly.

Generally, from a components point of view Hyperledger contains various elements described here:

Consensus layer: These services are responsible for facilitating the agreement process between the participants on the blockchain network. The consensus is required to make sure that the order and state of transactions is validated and agreed upon in the blockchain network.

Smart contract layer: These services are responsible for implementing business logic as per the requirements of the users. Transaction are processed based on the logic defined in the smart contracts that reside on the blockchain.

Communication layer: This layer is responsible for message transmission and exchange between the nodes on the blockchain network.

Security and crypto layer: These services are responsible for providing a capability to allow various cryptographic algorithms or modules to provide privacy, confidentiality and non-repudiations services.

Data stores: This layer provides an ability to use different data stores for storing state of the ledger. This means that data stores are also pluggable and allows usage of any database backend.

Policy services: This set of services provides the ability to manage different policies required for the blockchain network. This includes endorsement policy and consensus policy.

APIs and SDKs: This layer allows clients and applications to interact with the blockchain. An SDK is used to provide mechanisms to deploy and execute chaincode, query blocks and monitor events on the blockchain.

the role of membership services in Hyperledger Fabric is to manage and maintain the network participants and their access to network resources, enforce network policies, maintain privacy, and support consensus and kefjy management processes.

Services:

Membership services : These services are used to provide access control capability for the users of the fabric network. The following list shows the functions that membership services perform:

- User identity verification
- User registration
- Assign appropriate permissions to the users depending on their roles

Membership services make use of a certificate authority in order to support identity management and authorization operations. This CA can be internal (Fabric CA), which is a default interface in Hyperledger Fabric or organization can opt to use an external certificate authority. Fabric CA issues enrollment certificates (E-Certs), which are produced by enrollment certificate authority (E-CA).

Blockchain services

Blockchain services are at the core of the Hyperledger Fabric. Components within this category are as follows.

1. **Consensus services**

A consensus service is responsible for providing the interface to the consensus mechanism. This serves as a module that is pluggable and receives the transaction from other Hyperledger entities and executes them under criteria according to the type of mechanism chosen. Consensus is pluggable and currently, there are two types of ordering services available in Hyperledger Fabric:
SOLO: This is a basic ordering service intended to be used for development and testing purposes.
Kafka: This is an implementation of Apache Kafka, which provides ordering service. It should be noted that currently Kafka only provides crash fault tolerance but does not provide byzantine fault tolerance

2. **Distributed ledger**

Blockchain and world state are two main elements of the distributed ledger. Blockchain is simply a cryptographically linked list of blocks and world state is a key-value database. This database is used by smart contracts to store relevant states during execution by the transactions.

3. **The peer to peer protocol**

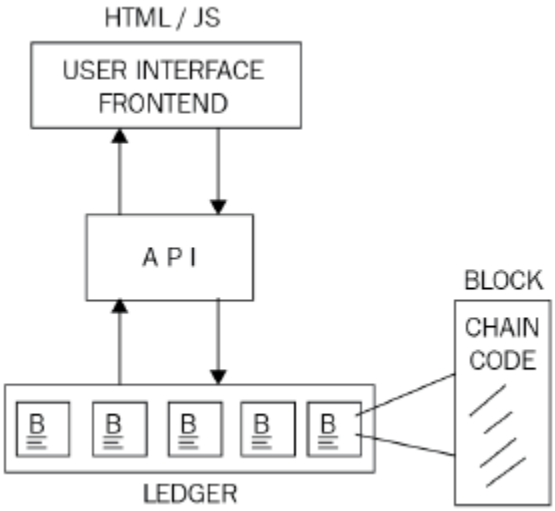
The P2P protocol in the Hyperledger Fabric is built using google RPC (gRPC). It uses protocol buffers to define the structure of the messages. Messages are passed between nodes in order to perform various functions.
There are four main types of messages in Hyperledger Fabric: discovery, transaction, synchronization, and consensus. Discovery messages are exchanged between nodes when starting up in order to discover other peers on the network. Transaction messages are used to deploy, invoke, and query transactions, and consensus messages are exchanged during consensus.
Synchronization messages are passed between nodes to synchronize and keep
the blockchain updated on all nodes.

4. **Ledger storage**

In order to save the state of the ledger, by default, LevelDB is used which is available at each peer.

5. Chaincode services

These services allow the creation of secure containers that are used to execute the chaincode. Components in this category are as follows:



- Secure container: Chaincode is deployed in Cocker containers that provide a locked down sandboxed environment for smart contract execution. Currently, Golang is supported as the main smart contract language, but any other mainstream languages can be added and enabled if required.
- Secure registry: This provides a record of all images containing smart contracts.

Alternative Blockchains

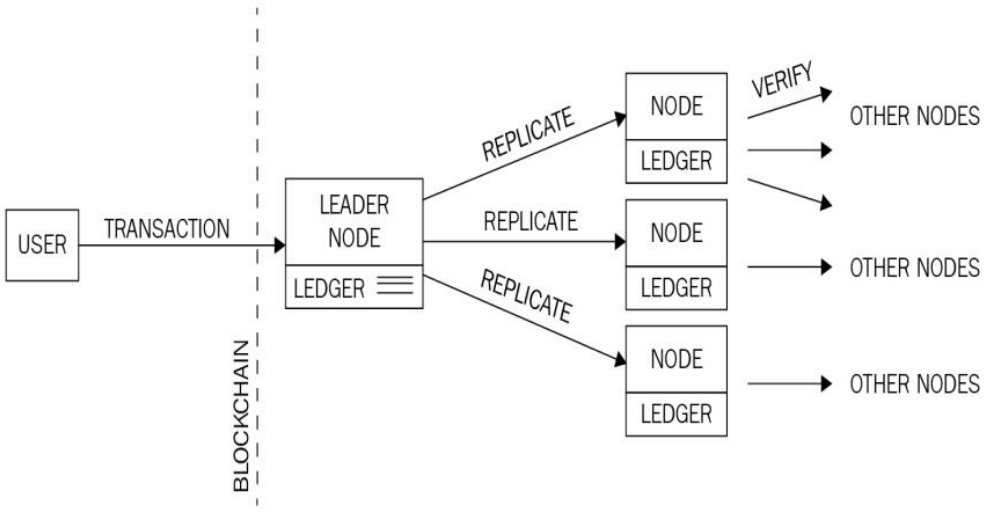
Kadena, Ripple, Stellar, Quorum are the various blockchains

1. Kadena:

   Kadena is a private blockchain that has successfully addressed scalability and privacy issues in blockchain systems.A new Turing incomplete language, called Pact, has also been introduced with Kadena that allows the development of smart contracts. A key innovation in Kadena is its Scalable BFT consensus algorithm, which has the potential to scale to thousands of nodes without performance degradation.

   confidentiality is another significant aspect of Kadena that enables privacy of transactions on the blockchain. This security service is achieved by using a combination of key rotation, symmetric on-chain encryption, incremental hashing, and Double Ratchet protocol.

   Scalable BFT consensus protocol ensures that adequate replication and consensus has been achieved before smart contract execution.



Consensus mechanism in Kadena

Once the consensus is achieved, a smart contract execution can start and takes a number of steps, as follows:

1. First, the signature of the message is verified.
2. Pact smart contract layer takes over.
3. Pact code is compiled.
4. The transaction is initiated and executes any business logic embedded within the smart contract. In case of any failures, an immediate rollback is initiated that reverts that state back to what it was before the execution started.
5. Finally, the transaction completes and relevant logs are updated.

2. Stellar :
   Stellar is a payment network based on blockchain technology and a novel consensus model called Federated Byzantine Agreement (FBA). FBA works by creating quorums of trusted parties. Stellar Consensus Protocol (SCP) is an implementation of FBA.

It has four main properties:

- Decentralized control: This allows participation by anyone without any central party
- Low latency: This addresses the much-desired requirement of fast transaction processing
- Flexible trust: This allows users to choose which parties they trust for a specific purpose
- Asymptotic security: This makes use of digital signatures and hash functions for providing the required level of security on the network

3. Quorum :
   This is a blockchain solution built by enhancing the existing Ethereumblockchain. There are several enhancements such as transaction privacy and a new consensus mechanism that has been introduced in Quorum. Quorum has introduced a new consensus model known as QuorumChain, which is based on a majority voting and time-based mechanism. Another feature called Constellation is also introduced which is a general-purpose mechanism for submitting information and allows encrypted communication between peers. Furthermore, permissions at

node level is governed by smart contracts. It also provides a higher level of performance compared to public Ethereumblockchains.

4. Ripple:
   Introduced in 2012, Ripple is a currency exchange and real-time gross settlement system. In Ripple, the payments are settled without any waiting as opposed to traditional settlement networks, where it can take days for settlement. It has a native currency called Ripples (XRP). It also supports non-XRP payments.

The Ripple network is composed of various nodes that can perform different functions based on their type:

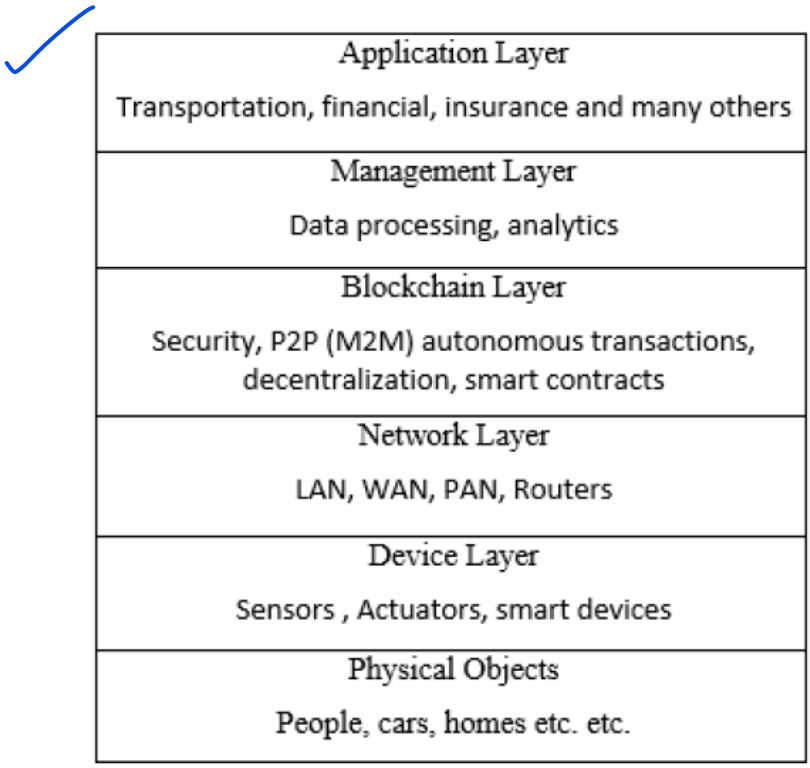
- User nodes: These nodes use in payment transactions and can pay or receive payments.
- Validator nodes: These nodes participate in the consensus mechanism. Each server maintains a set of unique nodes, which it needs to query while achieving consensus. Nodes in the Unique Node List (UNL) are trusted by the server involved in the consensus mechanism and will accept votes only from this list of unique nodes.

Blockchain – Outside of Currencies

Digital currencies were the first-ever application of blockchain technology

Internet of Things

IoT can be defined as a network of computationally intelligent physical objects (any object such as cars, fridges, industrial sensors, and so on) that are capable of connecting to the internet, sensing real-world events or environments, reacting to those events, collecting relevant data, and communicating it over the internet.
As per the definition of IoT, four functions come to light as being performed by an IoT device. These include sensing, reacting, collecting, and communicating. All these functions are performed by using various components on the IoT device.
Sensing is performed by sensors. Reacting or controlling is performed by actuators, the collection is a function of various sensors, and communication is performed by chips that provide network connectivity.
The usual IoT model is based on a centralized paradigm where IoT devices usually connect to a cloud infrastructure or central servers to report and process the relevant data back. This centralization poses certain possibilities of exploitation including hacking and data theft. According to IBM, blockchain for IoT can help to build trust, reduce costs, and accelerate transactions. Additionally, decentralization, which is at the very core of blockchain technology, can eliminate single points of failure in an IoT network.



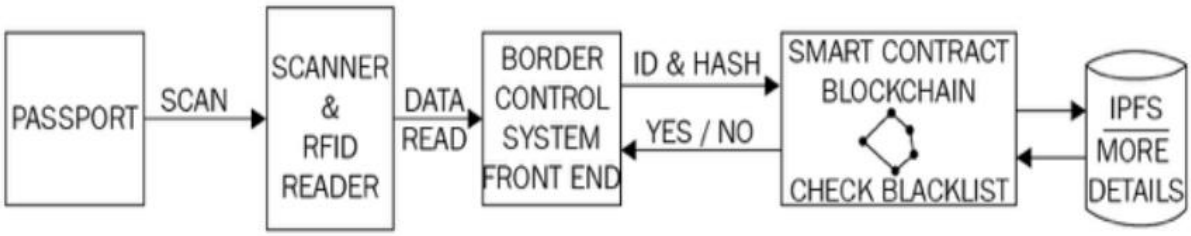
Blockchain-based IoT model

Physical object layer
These include any real-world physical objects. It includes people, animals, cars, trees, fridges, trains, factories, homes, and that is required to be monitored and controlled can be connected to the IoT.
Device layer
This layer contains things that make up the IoT such as sensors, transducers, actuators, smartphones, smart devices, and Radio-Frequency Identification (RFID) tags.
Network layer
This layer is composed of various network devices that are used to provide Internet connectivity between devices and to the cloud or servers that are part of the IoT ecosystem. These devices can include gateways, routers, hubs, and switches. This layer can include two types of communication.
Management layer: This layer provides the management layer for the IoT ecosystem . It can consist of only software related to analytics and processing.
Blockchain layer:It take care of Security and control can be moved to the. It can also result in cost saving which is due to easier device management by using a blockchain based decentralized approach. The IoT network can be optimized for performance by using blockchain. There will be no need to store IoT data centrally for millions of devices because storage and processing requirements can be distributed to all IoT devices on the blockchain. This can result in completely removing the need for large data centers for processing and storing the IoT data.
Blockchain-based IoT can also thwart denial of service attacks where hackers can target a centralized server or data center more efficiently, but with blockchain's distributed and decentralized nature, such attacks are no longer possible.
Application layer : This layer includes applications running on top of the IoT network. This layer can consist of many applications depending on the requirements such as transportation, healthcare, financial, insurance, or supply chain management.

Other Applications of Blockchain

1. Government: Government or electronic government is a paradigm where information and communication technology are used to deliver public services to citizens.Many governments are researching the possibility of using blockchain technology for managing and delivering public services including but not limited to identity cards, driving licenses, secure data sharing among various government departments and contract management. Transparency, auditability, and integrity are attributes of blockchain that can go a long way in effectively managing various government functions.

   Border control: Machine-readable travel documents and specifically biometric passports have paved the way for automated border control. Currently, there is no mechanism available to blacklist or revoke a suspected passport immediately and broadcast it to the border control ports worldwide. Blockchain can provide a solution to this problem by maintaining a blacklist in a smart contract which can be updated as required and any changes will be immediately visible to all agencies and border control points thus enabling immediate control over the movement of a suspected travel document.



Automated border control using blockchain

   Voting Voting in any government is a key function and allows citizens to participate in the democratic election process.Blockchain-based voting systems can resolve issues by introducing end-to-end security and transparency in the process. Security is provided in the form of integrity and authenticity of votes by using public key cryptography which comes as standard in a blockchain. Moreover, immutability guaranteed by blockchain ensures that votes cast once cannot be cast again. This can be achieved through a combination of biometric features and a smart contract maintaining a list of votes already cast. For example, a smart contract can maintain a list of already casted votes with the biometric ID (for example a fingerprint) and can use that to detect and prevent double casting. Secondly, Zero-Knowledge Proofs (ZKPs) can also be used on the blockchain to protect voters' privacy on the blockchain.

   Citizen identification (ID cards): A blockchain-based online digital identity allows control over personal information sharing. Users can see who used their data and for what purpose and can control access to it. This is not possible with the current infrastructures which are centrally controlled. The key benefit is that a single identity issued by the government can be used easily and in a transparent manner for multiple services via a single government blockchain. In this case, the blockchain serves as a platform where a government is providing various services such as pensions, taxation, or benefits and a single ID is being used for accessing all these services. Blockchain, in this case, provides a permanent record of every change and transaction made by a digital ID, thus ensuring integrity and transparency of the system. Also, citizens can notarize birth certificates, marriages, deeds,

and many other documents on the blockchain tied with their digital ID as a proof of existence.

2. **Health**: Blockchain provides an immutable, auditable, and transparent system that traditional peer-to-peer networks cannot. With the adaptability of blockchain in the health sector, several benefits can be realized, ranging from cost saving, increased trust, faster processing of claims, high availability, no operational errors due to complexity in the operational procedures, and preventing the distribution of counterfeit medicines.
3. **Finance**: Blockchain in finance is the hottest topic in the industry currently, and major banks and financial organizations are researching to find ways to adapt blockchain technology primarily due to its highly-desired potential to cost-save.
4. **Insurance**: In the insurance industry, blockchain technology can help to stop fraudulent claims, increase the speed of claim processing, and enable transparency. Imagine a shared ledger between all insurers that can provide a quick and efficient mechanism for handling intercompany claims. Also, with the convergence of IoT and blockchain, an ecosystem of smart devices can be imagined where all these things can negotiate and manage their insurance policies controlled by smart contracts on the blockchain.Blockchain can reduce the overall cost and effort required to process claims. Claims can be automatically verified and paid via smart contracts and the associated identity of the insurance policyholder.
5. **Post-trade settlement**: the post-trade settlement process usually takes two to three days and has a dependency on central clearing houses and reconciliation systems. With the shared ledger approach, all participants on the blockchain can immediately see a single version of truth regarding the state of the trade. Moreover, the peer-to-peer settlement is possible, which results in the reduction of complexity, cost, risk, and the time it takes to settle the trade. Finally, intermediaries can be eliminated by making use of appropriate smart contracts on the blockchain. Also, regulators can also see view the blockchain for auditing and regulatory requirements.
6. **Financial crime prevention**: Know Your Customer (KYC), and Anti Money Laundering (AML) are the key enablers for the prevention of financial crime.In the case of KYC, currently, each institution maintains their own copy of customer data and performs verification via centralized data providers. This can be a timeconsuming process and can result in delays in onboarding a new client. Blockchain can provide a solution to this problem by securely sharing a distributed ledger between all financial institutions that contain verified and true identities of customers. This distributed ledger can only be updated by consensus between the participants thus providing transparency and auditability. This can not only reduce costs but also enable meeting regulatory and compliance requirements in a better and consistent manner.
7. **Media**: Critical issues in the media industry revolve around content distribution, rights management, and royalty payments to artists.Blockchain can provide a network where digital music is cryptographically guaranteed to be owned only by the consumers who pay for it. This payment mechanism is controlled by a smart contract instead of a centralized media agency or authority. The payments will be automatically made based on the logic embedded within the smart contract and number of downloads. Moreover, illegal copying of digital music files can be stopped altogether because everything is recorded and owned immutably in a transparent manner on the blockchain. A music file, for example, can be stored with owner information and timestamp which can be traced throughout the blockchain network. Furthermore, the consumers who own a legal copy of some content are cryptographically tied to the content they

have, and it cannot be moved to another owner unless permissioned by the owner. Copyrights and transfers can be managed easily via blockchain once all digital content is immutably recorded on the blockchain. Smart contracts can then control the distribution and payment to all concerned parties

Healthcare: Blockchain can be used to securely store and manage electronic medical records (EMRs). By using blockchain, healthcare organizations can ensure that EMRs are accurate, complete, and secure, while also making it easier for patients to control access to their personal health information.

Supply Chain Management: Blockchain can be used to track the movement of goods and materials through a supply chain. It can provide a transparent and secure record of all transactions, making it easier to detect and prevent fraud, reduce the risk of errors, and improve the overall efficiency of the supply chain.

Real Estate: Blockchain can be used to create digital registries of property ownership and transactions, making it easier for buyers, sellers, and governments to verify the ownership of a property. This can help to reduce fraud, increase transparency, and make it easier to transfer ownership of real estate assets.

Digital Identity Management: Blockchain can be used to securely store and manage personal information, such as name, date of birth, and biometric data. This information can then be used to create a tamper-proof digital identity that can be used across a variety of applications, such as voting systems, financial services, and online marketplaces.