



Introduction to Blockchain and Algorand

LECTION 1

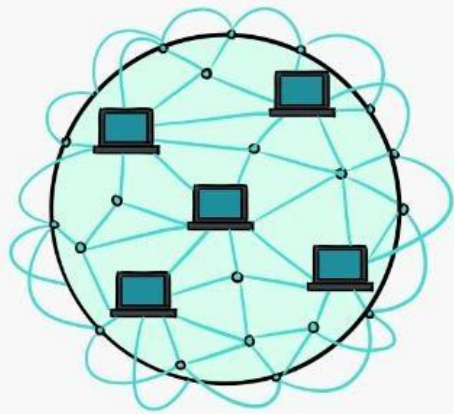
Plan

- What is blockchain. Types of blockchains (private/public).
- The difference between a blockchain and a database.
- The use of blockchain in various industries.
- Algorand blockchain. Pure Proof of Stake (PPoS).
- AVM.
- What is Dapps and smart contracts. The role of smart contracts in blockchain ecosystems.



A blockchain is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets across a business network. An asset can be tangible (house, car, money, land) or intangible (intellectual property, patents, copyrights, branding). Almost anything of value can be tracked and sold in blockchain network, which reduces risks and costs for all participants.

- IBM Blockchain



Blockchain Ledger



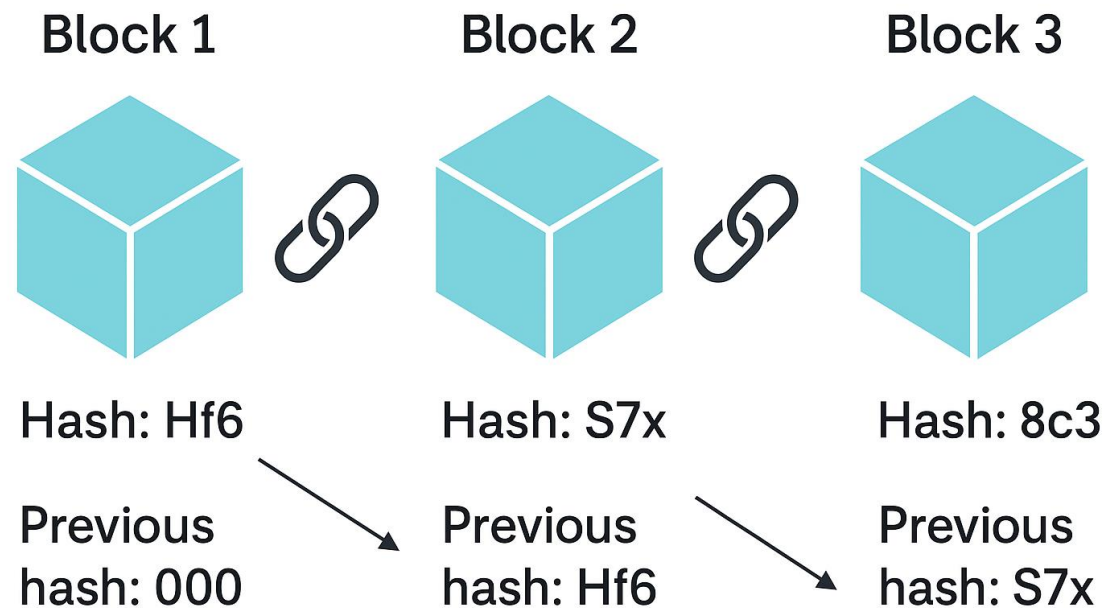
Traditional Ledger

A blockchain is a public ledger (or file) of transactional data distributed across many computers (nodes) on a network. All of these nodes work together using the same set of software and rules to validate transactions for addition to the final ledger.

- *Algorand blockchain*

Blockchain is known as a chain of blocks.

What makes it special is that each block stores knowledge about the previous one. Each block in the chain has a hash, which is like a unique digital fingerprint, representing a specific piece of information that links it to the previous block, creating a chain of blocks that is virtually tamper-proof.





Types of blockchains.

Public vs Private

- A public blockchain is a type of blockchain that is open to for all and can be accessed and verified by any network participant. Its key features and benefits include transparency, security and decentralisation. Examples of public blockchains include Ethereum, Bitcoin, and Algorand.
- A private blockchain is a type of blockchain that is limited to a group of participants who have been granted permission to access the network. Its key features include privacy and control. However, the disadvantage is that they are vulnerable to security threats, opaque, can be expensive to maintain, and centralised.

The difference between public and private blockchains

	Public	Private
Availability	Anyone is free to join and participate in the main activities of the of the network blockchain, including reading, writing, adding blocks, and auditing network activity.	To the networks can join only selected and verified members.
Transparency	Transparent, because all transactions are visible to anyone in the network.	Private, as only authorised users can view data and transactions on the network.
Control	Decentralised and managed by a community of users with no single point of control. Once the blocks are verified, records cannot be edited or deleted.	Centralised and controlled by a single entity or organisation. The operator has the right to override, edit, or delete records in the blockchain.



The difference between databases and blockchain

- A conventional database simply stores and retrieves data.
- The blockchain stores data, receives data, connects to peer-to-peer devices, checks new data against existing rules, and transmits this information across the network, and it does so constantly.
- A database typically structures its data in a table, while a blockchain, as its name suggests, structures its data into chunks (blocks) that are linked to each other.

The difference between databases and blockchain

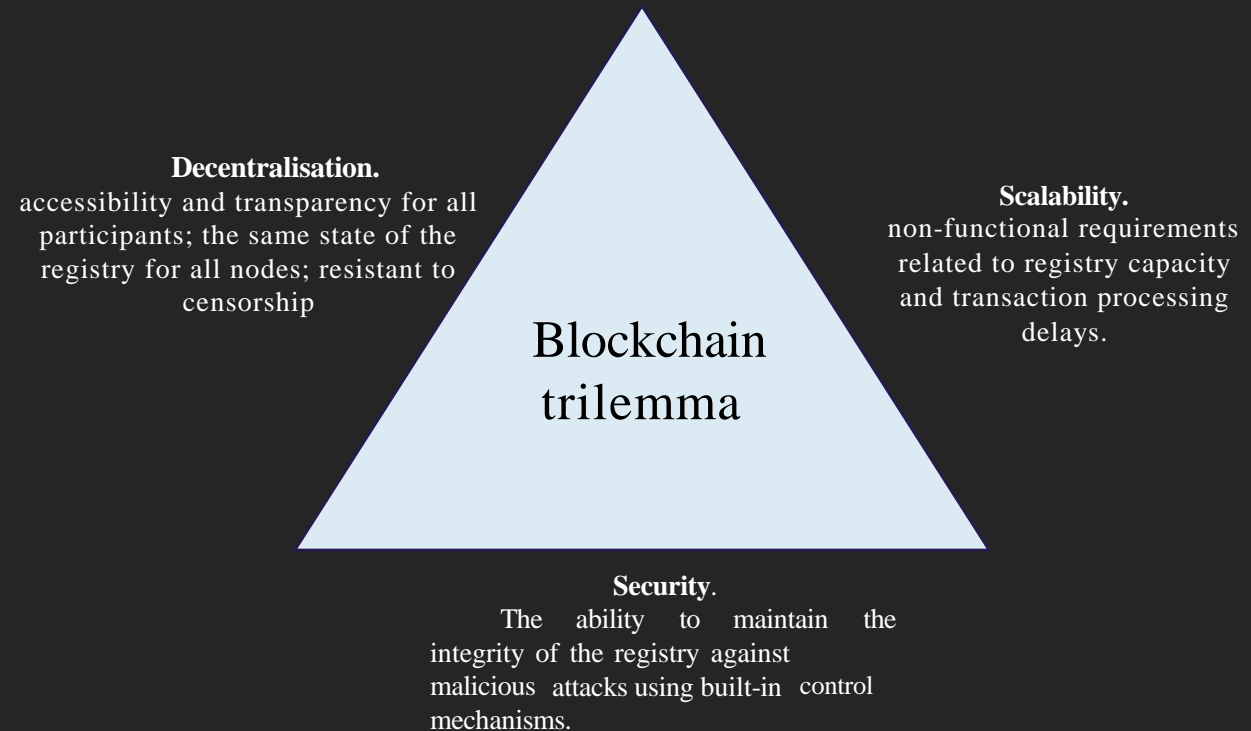
	Databases	Blockchain
Centralisation and decentralisation	The database is usually centralised, i.e. all data is stored and managed by a single centralised organisation or entity.	Blockchain is a decentralised technology, where data are distributed and managed by different nodes or participants in the network without central control.
Consensus mechanism	Data control and integrity is usually performed by a centralised system or database administrator.	In a blockchain, a consensus mechanism allows network participants to reach agreement on the state of the network and transactions, which ensures data integrity.
Data integrity	In the database, data can be modified or deleted by a centralised authority in accordance with access rights and security policies.	Data in the blockchain is immutable. Once the data is added to the blockchain, it cannot be changed or deleted without the consent of the majority of network participants, which ensures data integrity.

Blockchain

- A public ledger of transactional data represented as a chain of blocks distributed through a system of many nodes.
- A set of rules for agreeing on the next block: consensus. The network selects a proposer block, which then distributes the block to network nodes. Nodes check the block for the correctness of the transaction and the right to offer. If correct, the blockchain is added.
- Publicly verifiable (transparent), does not require permission and is protected from unauthorised access.

The blockchain trilemma

- The blockchain trilemma refers to the problem of achieving the three most important aspects of blockchain technology: security, scalability and decentralisation.
- The trilemma implies that optimising one aspect often compromises the others, making it difficult to achieve all three simultaneously.



The main of any blockchain is the ability not to be controlled by any one node. This is one of the most important and, at the same time, the most sought- after features, as it allows new types of business to exist.

There is no state control.



Blockchain application areas

Finance

- Cryptocurrencies and digital assets.
- International transfers and payments.
- Electronic payment systems.

Logistics and deliveries

- Supply chain management.
- Improving transport services.
- Ensuring easy reverse flow of money during payments and documenting financial transactions.

Internet of Things (IoT)

- Secure data exchange between devices.
- Process tracking and automation.
- Creation of decentralised management and monitoring systems.

Blockchain application areas

Medicine

- Electronic medical records and patient identification.
- Tracking of medicines and medical equipment.
- Medical data and research management.

Real estate

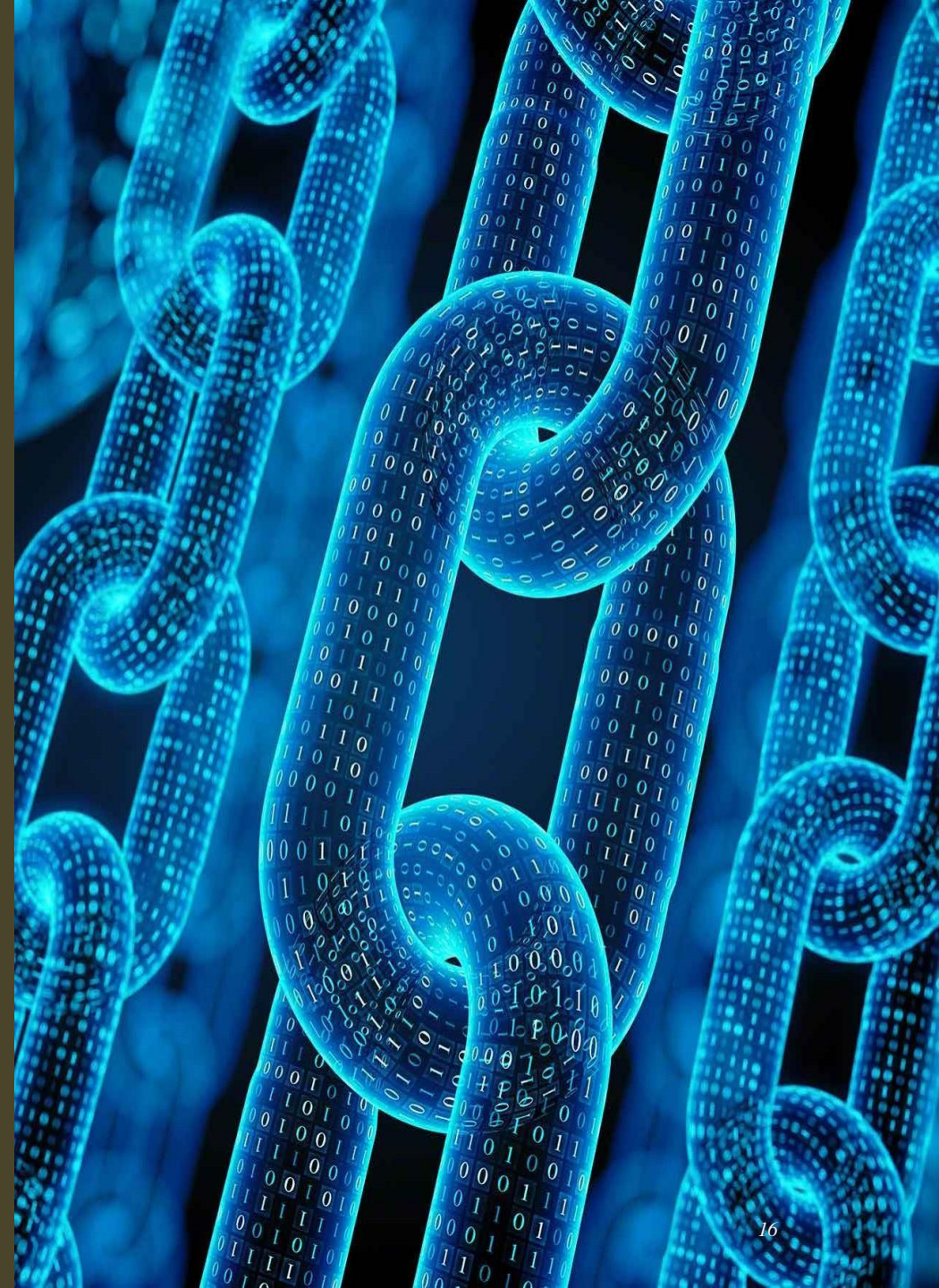
- Property registration and property rights management.

Voting.

- Electronic voting.
- Increase the transparency and security of electoral processes.

Blockchain application areas

- Copyright protection
- Crowdfunding
- Registration of land ownership
- Insurance
- Tourism
- Education
- Media and entertainment





Algorand is a decentralised network created to solve the blockchain trilemma: simultaneously achieving speed, security and decentralisation. Algorand was launched in June 2019 by computer scientist and MIT professor Silvio Micali. Algorand is an open-source blockchain network anyone can build on.

Algorand uses a Proof-of-Stake (PoS) consensus mechanism and distributes validator rewards to all holders of its own ALGO cryptocurrency.

Features of Algorand

1. **Pure Proof of Stake (PPoS) protocol:** Algorand uses a consensus-based PPoS mechanism that allows for high transaction speeds and a high degree of security. Each Algorand token holder can participate in the decision-making process by using their tokens as votes.
2. **Scalability:** Algorand is designed with scalability in mind. The protocol can process thousands of transactions per second (TPS), making it one of the fastest blockchains.
3. **Security:** Algorand's consensus algorithm guarantees the security of the network, even with a large number of participants. This is achieved by randomly selecting participants to create new blocks and verify transactions.
4. **Decentralisation:** Algorand strives to ensure the highest level of decentralisation, avoiding the problems associated with centralised actors or groups. This makes the network resistant to censorship and manipulation.
5. **Openness:** Algorand is completely open and does not require permissions. Anyone anywhere in the world who owns Algos can participate in the consensus.

Features of Algorand

1. **Algo:** Like every blockchain, Algorand has its own currency, Algo, which plays a crucial role in incentivising good network behaviour. If you own Algos, you can sign up to participate in the consensus, which means you will participate in the process of proposing and voting for new blocks. Algos are also used to pay transaction fees on the network.
2. **Low transaction cost:** the minimum transaction fee is only 1000 microAlgos or 0.001 Algos. The Algorand network does not have the concept of gas fees like Ethereum.
3. **Development toolkit:** Developers can write smart contracts in Python and use one of four SDKs (Python, JavaScript, Golang, Java) to connect to network resources or applications.

Protocol and consensus mechanism

A consensus protocol is a set of rules and procedures that define how network participants reach agreement in a coordinated manner on the current state of the system and the correctness of transactions. Consensus protocols are based on the idea of ensuring data integrity and conflict resolution in decentralised systems.

A consensus mechanism is a specific implementation of a consensus protocol that is used in a particular system or network to reach agreement among its participants. There are different consensus mechanisms that can use different algorithms, techniques, and approaches. Examples of consensus mechanisms are "Proof of Work", "Proof of Stake" etc.



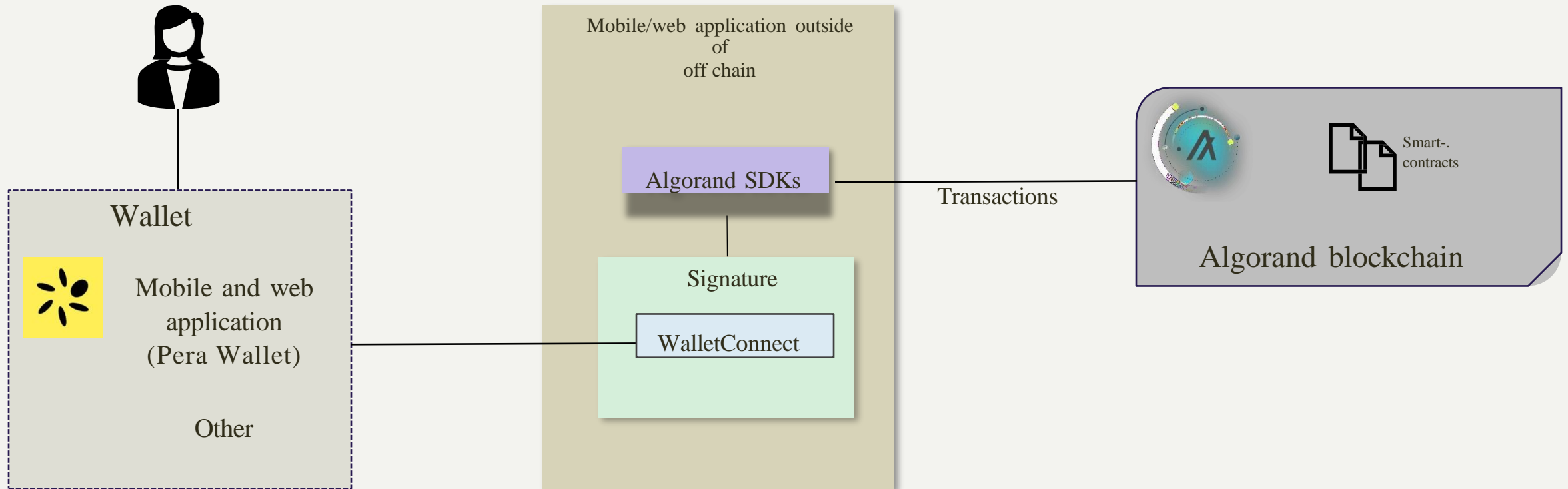



What are decentralized applications?

Decentralised applications, or dApps, are programs that run on a decentralised computing system, such as blockchain. They are mostly or completely decentralised.

For most applications, smart contracts will be only a part of the dApp architecture. Usually, developers create blockchain-based functionality in the dApp and some external interface to interact with smart contracts.

Architecture of a decentralised application in Algorand





What are smart contracts?

Smart contracts are software codes stored on a blockchain that have the ability to automatically execute the agreements recorded in them. They allow the parties to a transaction to automate and securely fulfil the terms without the need for intermediaries.

Smart contracts don't contain legal wording, terms, or agreements - just code that executes actions when certain conditions are met.

The life cycle of a smart contract





The life cycle of a smart contract refers to the sequence of stages that a smart contract goes through from its creation to its termination or completion.

This life cycle typically includes the following stages

- design,
- development,
- testing,
- deployment,
- execution,
- audit,
- modifications
- completion.

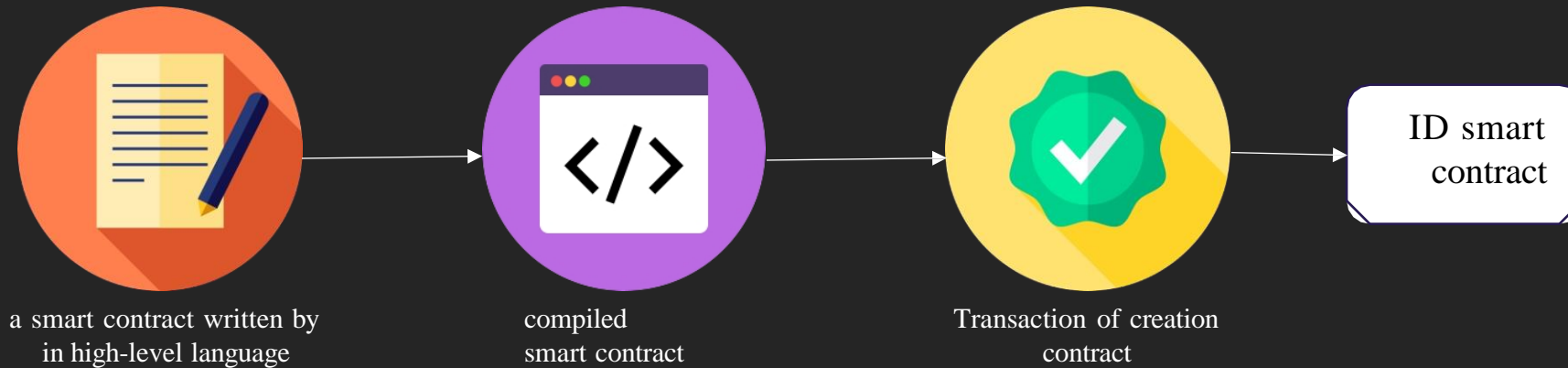
Each of these stages involves certain actions and requirements adapted to the needs and conditions of the smart contract.

Smart contracts are typically written in a high-level language, the type of which is characterised by the respective blockchain network. In order for a smart contract to be deployed, it must be compiled into low-level code that can be understood by a blockchain virtual machine.

Blockchain	Language.
Ethereum	Solidity/Vyper
Algorand	PyTeal/Reach
Hyperledger Fabric	JavaScript/Java/Go

Once compiled from code written in a high-level language, the smart contract can be deployed on the blockchain platform using a special transaction to create the contract.

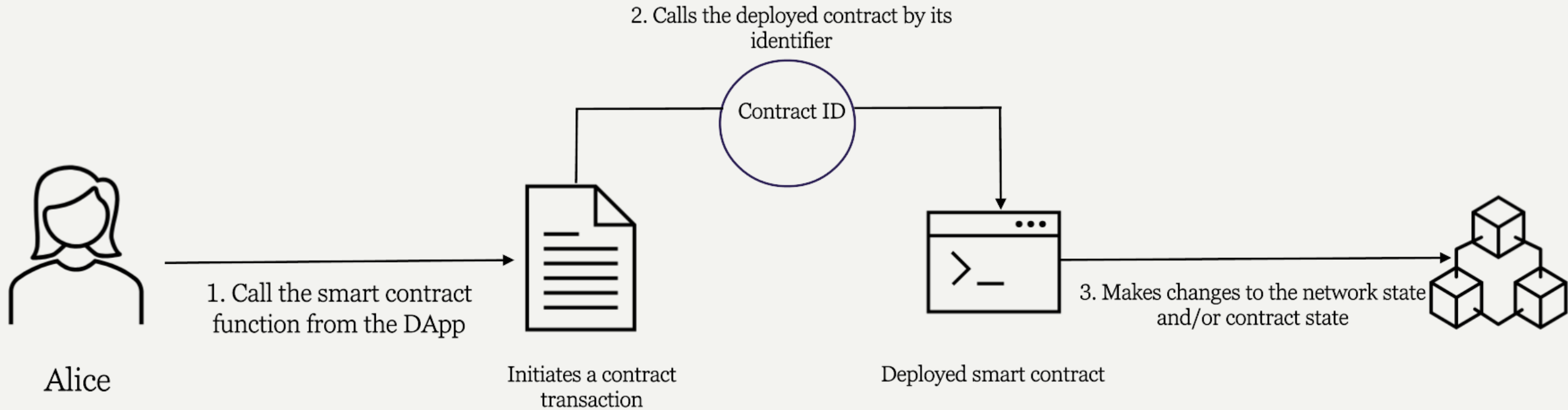
A unique identifier will be returned from this transaction. You can use it to to interact with the extended contract.



Smart contract:

- is launched only when it is called by a transaction.
- never runs in the background by itself.
- may trigger another contract, which may trigger another contract.

But the first call to a smart contract is made by a transaction from an external object (user).



Creation of smart contracts for the Algorand blockchain. Development environment



Setting up your environment

You can choose from three options when setting up your development environment:

- **Algorand Sandbox:** The most commonly used option is the Algorand sandbox setup. The sandbox allows developers to create local private networks. In addition, you can quickly delete a network, reset its state, or create a new network.
- **Third-party API services:** It is possible to use third-party API services to access Algorand's own REST APIs for the main network, test network, and beta network. This is a great choice if you don't want to set up a local network using Docker, but just want to experiment with Algorand development first.
- **Custom node:** It is possible to run your own Algorand node that contains a complete implementation of the Algorand software. This solution is more complex to configure and less flexible. Unlike the Algorand Sandbox, you can't throw away the network and set up a new one whenever you want. Setting up an Algorand node takes much more time than setting up a LAN using Sandbox tools.



Algorand network

Algorand has three public networks: MainNet, TestNet and BetaNet.

- Mainnet: the main network used to create real transactions and deploy smart contracts. All transactions on the Mainnet are final and have real economic value.
- Testnet: the test network is designed for testing and development. It simulates the Mainnet environment, but without the real economic value of transactions. Developers can test their applications without risking real assets.
- Betanet: serves as an early testbed for new features and updates that are not yet ready for Mainnet or Testnet. It is mainly used by developers who want to experiment with the latest Algorand protocols and updates.



Algorand Sandbox

The Algorand Sandbox is a tool designed to provide a simplified and easy-to-use environment for development and testing on the Algorand blockchain. It uses Docker to create isolated containers that mimic the Algorand network, allowing developers to experiment and build applications without having to set up a full Algorand node or connect to an existing network.

Main features of Algorand Sandbox

- For training: the sandbox is designed for training, not for creating real applications.
- Isolated environment: with Docker containers, the sandbox provides an isolated environment where you can develop and test your applications without affecting the active network (Mainnet, Testnet...).
- Pre-configured tools: sandbox comes with pre-configured tools such as 'goal' (an Algorand command-line tool) and other utilities needed for development.



Setting up your environment

The necessary software

Visual Studio Code, Docker (Docker Desktop for Windows), Linux Distribution for Windows

Optional: Windows Terminal

Setting up the Algorand Sandbox

1. Open the terminal (Ubuntu terminal on Windows)
2. Open the required local directory and run the following commands:

```
mkdir yourProjectName cd
```

```
yourProjectName
```

```
git clone https://github.com/algorand/sandbox.git
```

```
cd sandbox
```

```
./sandbox up
```

For [Ubuntu and macOS](#)

For [Windows](#)

Main teams

Sandbox teams:

up [config] -> start the sandbox environment; down ->

tear down the sandbox environment;

reset -> reset containers to their original state;

enter [algod||conduit||indexer||indexer-db] -> enter the sandbox container;

copyTo <file> -> copy <file> to algod. Useful for offline transactions, offline work
LogicSigs and TEAL.

copyFrom <file> -> copy <file> from algod. Useful for offline transactions, offline work
LogicSigs and TEAL.

More sandbox commands: [Algorand Sandbox Usage](#)



Sandbox containers

The Algorand sandbox uses Docker containers to simulate various components of the Algorand network:

- *algod Container:* The algod container runs the Algorand daemon, which is the main software responsible maintaining the Algorand blockchain. It manages block creation, verification, and consensus protocols. Developers use the algod container to locally simulate a mainnet or testnet environment.
- *indexer Container:* The indexer container runs the Algorand Indexer service, which provides advanced querying capabilities for recorded blockchain data. The Indexer indexes blockchain data, making it easier to query for information such as history transactions, asset balances and other status data.

Examples of commands

- Start the node with the standard configuration: `./sandbox up`
 - Enter the Algod container (you can use the same command to enter other containers, such as the indexer or indexer-db containers): `./sandbox enter algod`
 - Stop the sandbox: `./sandbox down`
 - Copy the file to the sandbox: `./sandbox copyTo "example.teal"`
- *If you placed the example.teal file in another folder, be sure to pass the absolute path to that file.
- This command will now create a copy of the example.teal application and place it in the algod.

An example of a workflow

./sanbox up (from sandbox path)

```
● olga@DESKTOP-QNI4VUN:~/pyTealStudy/sandbox$ ./sandbox up
```

Bringing up existing sandbox: 'release'

see sandbox.log for detailed progress, or use -v.

* docker containers started!

* waiting for services to initialize.

* services ready!

algod version

3298870427649

3.20.1.stable [rel/stable] (commit #6a6a15de)

go-algorand is licensed with AGPLv3.0

source code available at <https://github.com/algorand/go-algorand>

Indexer version

Dev Build compiled at 2024-01-01T19:42:34+0000 from git hash 3e9446aeb41010c514de43e8c2e1cd5e629f8773

Postgres version

postgres (PostgreSQL) 13.13

Example of a workflow

./sandbox goal account list

```
olga@DESKTOP-QNI4VUN:~/pyTealStudy/sandbox$ ./sandbox goal account list
[offline]      FXA6NGWY6MR2Y5Q7ZPFH7CC7OWHX5JAIED22235KH6RPK55GHUB5JH4054      FXA6NGWY6MR2Y5Q7ZPFH7CC7OWHX5JAIED22235KH6RPK55GH
UB5JH4054      999999999481000 microAlgos      [created app IDs: 1145, 1147, 1151, 1152, 1160, 1335, 1608, 1946]      [opted in
app IDs: 1152, 1160, 1335, 1608]
[online]       JLFS5K27S74MN2HF4C567UBK64LCJNPVE4DNKWX4W3ULGTNUVZH5QH2ML4      JLFS5K27S74MN2HF4C567UBK64LCJNPVE4DNKWX4W3ULGTNUV
ZH5QH2ML4      3999999999895000 microAlgos      [opted in app IDs: 1160, 1335, 1608]
[offline]      UW3N2WTVLZHOAZCNZCYTBOH4DSA7YSSNXYWVXZDIWGH34NII4ZDIUFZCGA      UW3N2WTVLZHOAZCNZCYTBOH4DSA7YSSNXYWVXZDIWGH34NII4
ZDIUFZCGA      4000000000000000 microAlgos      _
```

The command returns a list of all accounts available in the Algorand Sandbox environment. In particular, it provides information such as account addresses and corresponding balances (the balance is usually displayed in microAlgos (1 Algo= 1 000 000 microAlgos)).

Additional materials

[How does a blockchain work](#) (video)

[What is Algorand?](#)

[ALGORAND'S CORE TECHNOLOGY \(in a nutshell\)](#): An article by Algorand founder Silvio Micali explains Algorand's core technology and its motivation in more detail.

[Introducing Sandbox: The quick way to get started on Algorand](#) (lecture)