SRI VENKATESWARAA
COLLEGE OF TECHNOLOGY

GITATHON

ETHEREUM BLOCKCHAIN AND SMART CONTRACT

NAME: MOHANA PRIYA.B

DEPARTMENT:EEE

CLUB ID:SVCTBCC-C-C1025

Introduction to Ethereum platfrom

* Ethereum is decentralized blockchain with smart contract functionality.

* Ether is the native cryptocurrency of the platfrom .

* Among crptocurrencies , ether is second only to bitcoin in market capitalization ,It is open source software .

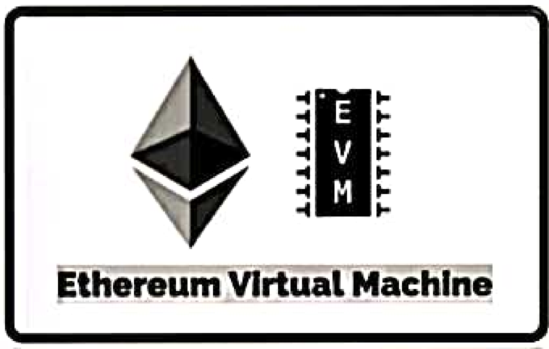
*Ethereum was conceived in 2013 by programmer vitalik Buterin .

*Additional founders of Etherum included Gavin Wood , Charless Hoskinson, Anthony Di lorio , and Joseph Lubin.

ETHER

- Ether is the native cryptocurrency of the Ethereum platform.
- It is used to pay for transaction fees and computational services on the network.
- Ether can also be traded on various cryptocurrency exchanges.

Ethereum Virtual Machine



- The Ethereum Virtual Machine (EVM) is like a computer inside the Ethereum blockchain.
- It runs programs called smart contracts that people create using a language called Solidity.
- These smart contracts control things like digital money and games on Ethereum.
- So the EVM is basically the brain that makes Ethereum apps work!

Gas -Smart Contract in block chain

* Gas Limit: Imagine it as a fuel tank for your transaction. You decide how much gas your transaction can use.

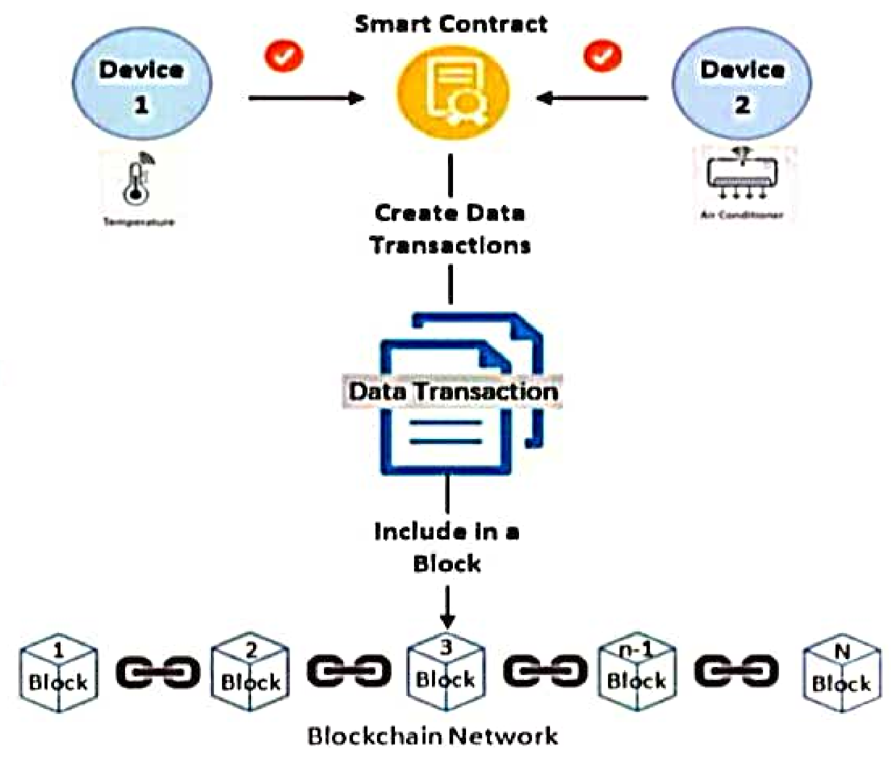
* Gas Price: This is like the cost of each unit of gas. You set this to say how much you're willing to pay per unit of gas.

* Gas Cost: It's just Gas Limit multiplied by Gas Price. This tells you how much you'll pay in total for your transaction.

* Gas Refund: If you don't use all the gas you set, you get back the unused gas.

Ethereum Gas

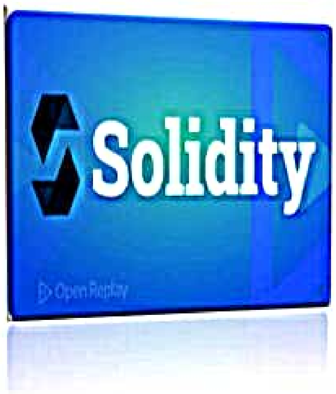
- Gas Limit: Think of it as the maximum amount of "fuel" you're willing to use for a transaction or smart contract.
- Gas Price: This is like the price you're willing to pay for each unit of fuel.
- Gas Cost: It's just the Gas Limit multiplied by the Gas Price, telling you the total cost of your transaction in Ethereum's cryptocurrency, Ether.
- Gas Refund: If you don't use all the fuel you set, you get back what's left. It's like getting a refund for unused gas in your car.
- Gas Fees: This is the actual fee you pay in Ether for using the Ethereum network. It's the Gas Limit times the Gas Price.



Structure of Ethereum in smart contracts

- Initialization: A smart contract is created and initialized with predefined code and data. It's like setting up a program with specific functions and variables.
- Functions: Smart contracts have functions, which are like actions or tasks they can perform. For example a payment function can transfer Ether from one account to another.
- State Variables: These are like the memory of the smart contract, storing data that persists between function calls. For instance a contract can have a variable to store the balance of an account.
- Events: Events are used to communicate with external applications. They're like notifications that something happened within the contract. such as a payment received or a condition met.

Solidity of features



* Data Types: Solidity has types for different kinds of information, like numbers and text.

* Functions: You can create functions in Solidity to make your contract do specific things.

* Variables: You use variables to store and work with data inside your contract.

* Control Structures: Solidity has ways to make decisions and repeat tasks, like loops and if statements.

* Inheritance: This lets you reuse code from other contracts and build on them.

Access Control

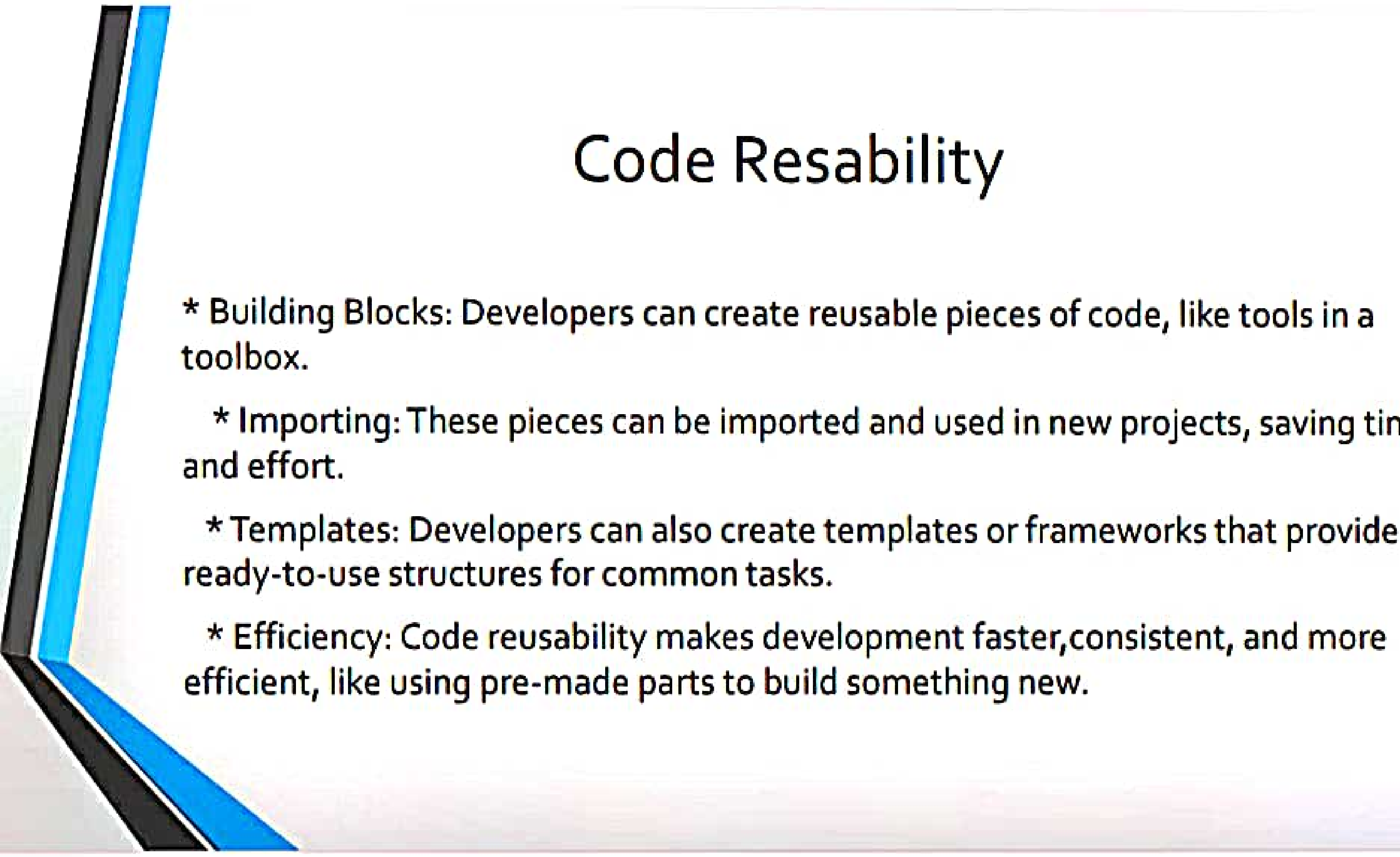
Roles: Users are assigned roles like admin, user, or guest.

Permissions: Each role has permissions, like read-only or full access.

Smart Contracts: These define who can do what, like a rulebook.

Transactions: When someone wants to do something, the smart contract checks if they have permission.

Security: This system ensures only authorized users can perform certain actions or access specific data.



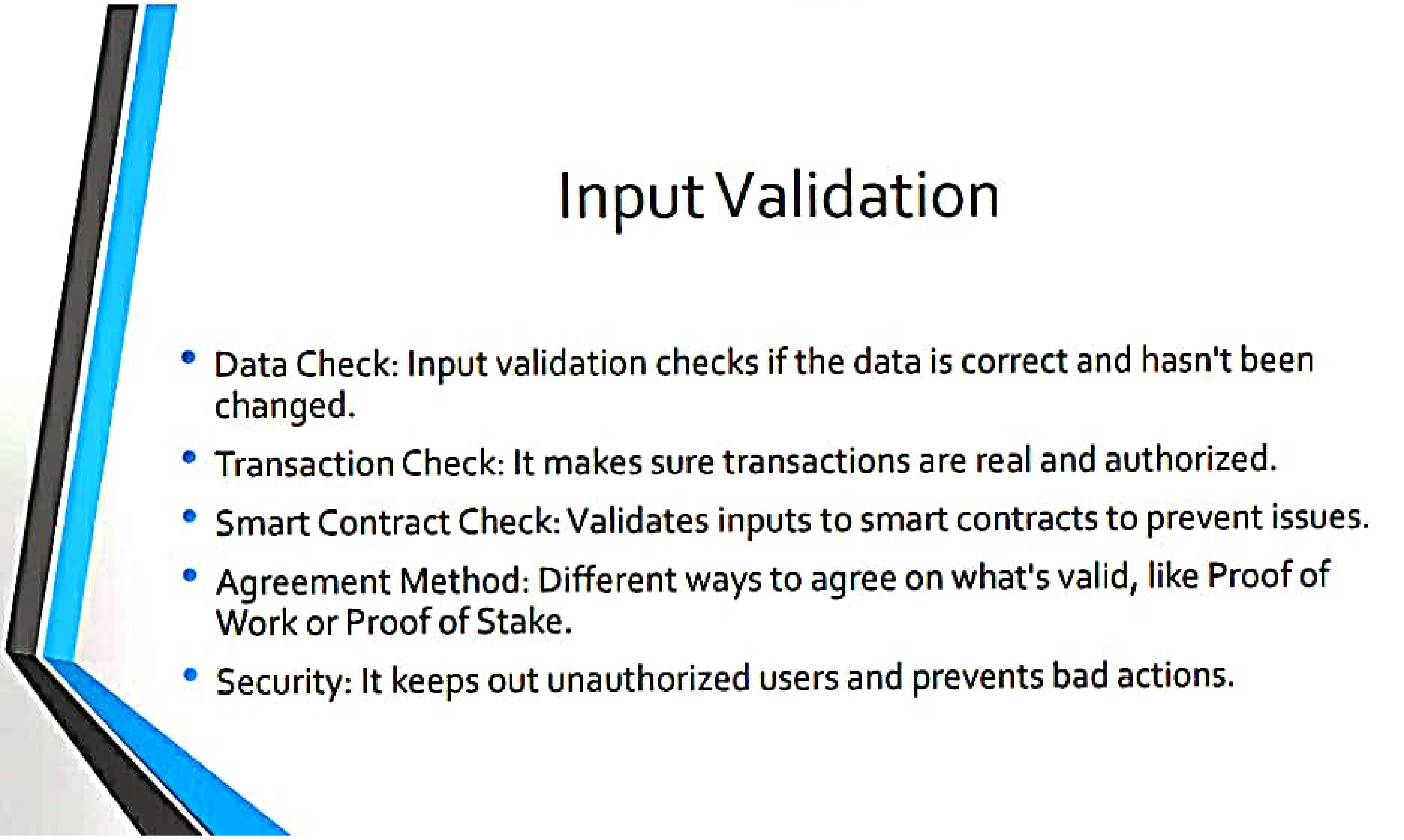
Code Resability

* Building Blocks: Developers can create reusable pieces of code, like tools in a toolbox.

* Importing: These pieces can be imported and used in new projects, saving tin and effort.

* Templates: Developers can also create templates or frameworks that provide ready-to-use structures for common tasks.

* Efficiency: Code reusability makes development faster,consistent, and more efficient, like using pre-made parts to build something new.



Input Validation

- Data Check: Input validation checks if the data is correct and hasn't been changed.
- Transaction Check: It makes sure transactions are real and authorized.
- Smart Contract Check: Validates inputs to smart contracts to prevent issues.
- Agreement Method: Different ways to agree on what's valid, like Proof of Work or Proof of Stake.
- Security: It keeps out unauthorized users and prevents bad actions.

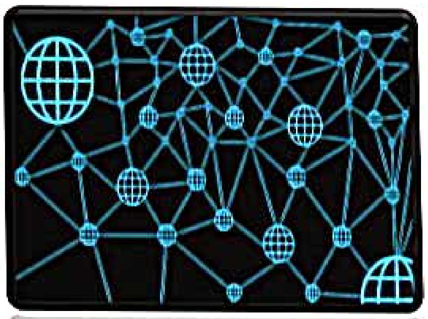
State Management

- No Changes Allowed: Once something is written in the blockchain, it stays there forever. This makes sure no one can cheat or alter records.
- Tracking Changes: When things like money or assets move around in the blockchain, it's like updating the book to show who owns what now.
- Everyone Agrees: Everyone using the blockchain agrees on what's written in the book. This agreement is reached through methods like Proof of Work or Proof of Stake.
- Smart Contracts: Smart contracts are like special pages in the book that can do things automatically, like releasing money when certain conditions are met.
- Keeping Everyone Updated: All computers in the blockchain network stay updated with the latest information in the book, so everyone sees the same accurate picture.

Gas Optimization

- Gas : The Gas is like fuel for blockchain transactions and smart contracts.
- Gas Cost: The cost of a transaction or contract depends on how much gas it uses and the price of gas. Gas price is like the cost per unit of gas.
- Estimate Gas: Check how much gas your action will need before doing it, so you don't waste gas.
- Manage Gas Prices: Sometimes, you can choose when to do your action based on gas prices to save money.

Event Decentrilized



* Events : The Events are like notifications from smart contracts.

- They tell other parts of the blockchain when something important happens.
- Declaring an Event: To declare an event, you give it a name and say what information it will include.
- This sends the notification with all the details.

Logging Events to the Blockchain

* Purpose of Logging: Logging events is like keeping a record of important events that happen within a smart contract.

* These events can be anything from a user making a transaction to a change in contract state.

* Event Declaration: Before logging events, you need to declare them in the smart contract. This includes defining the event's name and any relevant parameters it will include.

* Logging Events: When a specific action or event occurs that you want to record, you emit the corresponding event within the smart contract.

Logging Events to the Blockchain

* Purpose of Logging: Logging events is like keeping a record of important events that happen within a smart contract.

* These events can be anything from a user making a transaction to a change in contract state.

* Event Declaration: Before logging events, you need to declare them in the smart contract. This includes defining the event's name and any relevant parameters it will include.

* Logging Events: When a specific action or event occurs that you want to record, you emit the corresponding event within the smart contract.

Advanced Solidity

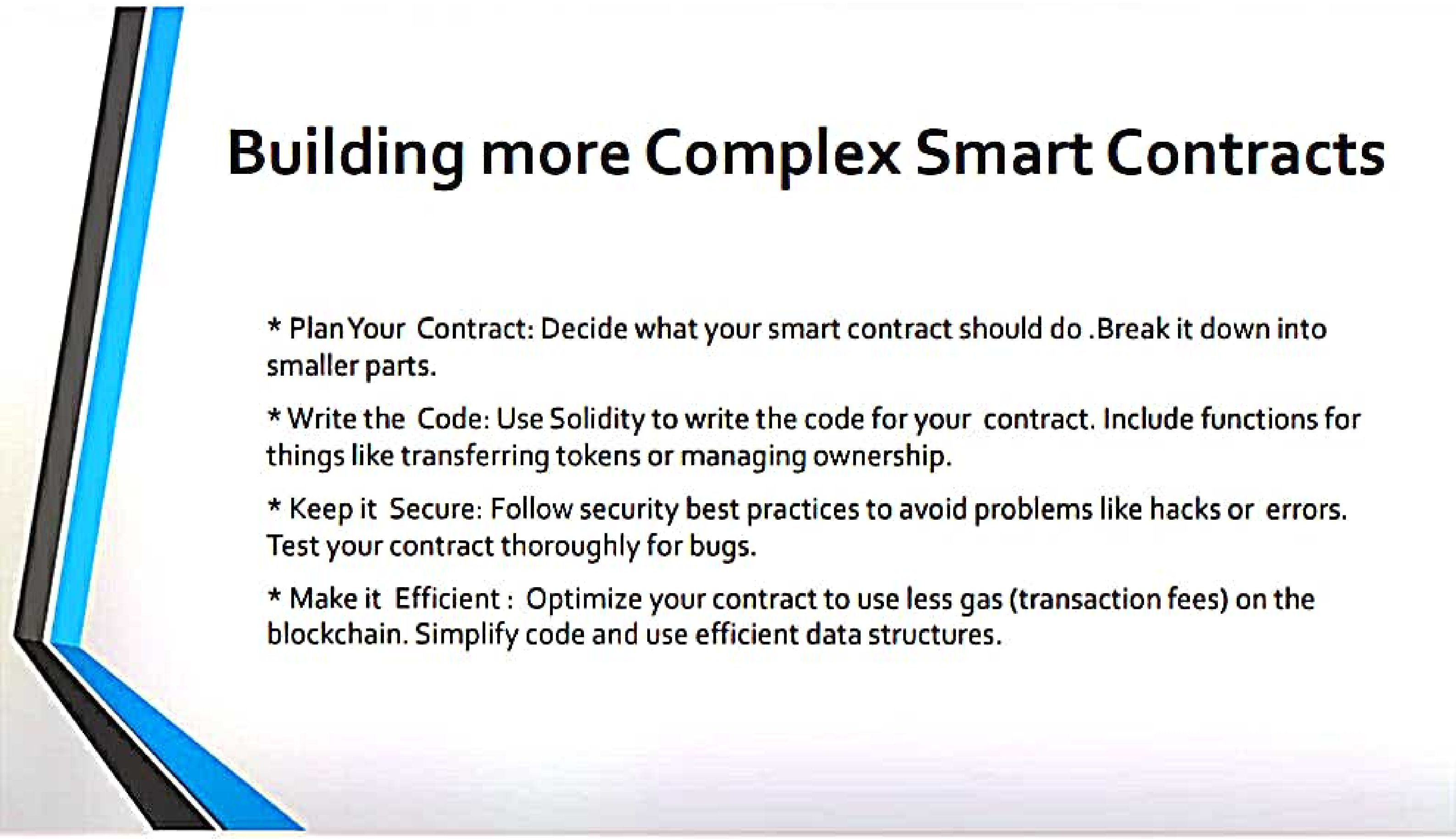
- Solidity Basics: Solidity is the language used to create smart contracts on Ethereum and similar blockchains.
- Advanced Features: Advanced Solidity means using special tools like.
- Interfaces and Abstract Contracts: Defining common rules for contracts to follow.
- Modifiers: Adding extra conditions to functions.
- Events: Recording important actions for transparency.

State variables and constants

- State Variables: The variables are used to store and maintain data across multiple function calls within a smart contract.
- Example: If you're building a token contract, a state variable could store the balances of different users.
- Mutable: State variables can be changed by functions within the contract, and their values are persisted on the blockchain.
- Constants Purpose: Constants are used to define values that remain the same throughout the life of the contract and cannot be changed.
- Example: In a contract, you might have a constant for the total supply of tokens, which doesn't change once set.

Payable and Non Payable functions

- Payable Functions: The Payable functions in smart contracts can receive cryptocurrency (like Ether in Ethereum) as part of the function call.
- Use Case: A payable function might be used to accept payments, transfer funds, or participate in crowdfunding campaigns where users send cryptocurrency to the contract.
- Example: A crowdfunding contract may have a payable function to accept contributions from backers.
- Non-Payable Functions: The Non-payable functions are unable to receive cryptocurrency. They are used for operations that do not involve sending or receiving funds.
- Use Case: Non-payable functions are commonly used for reading data from the blockchain, updating contract states, or performing computations without involving payments.



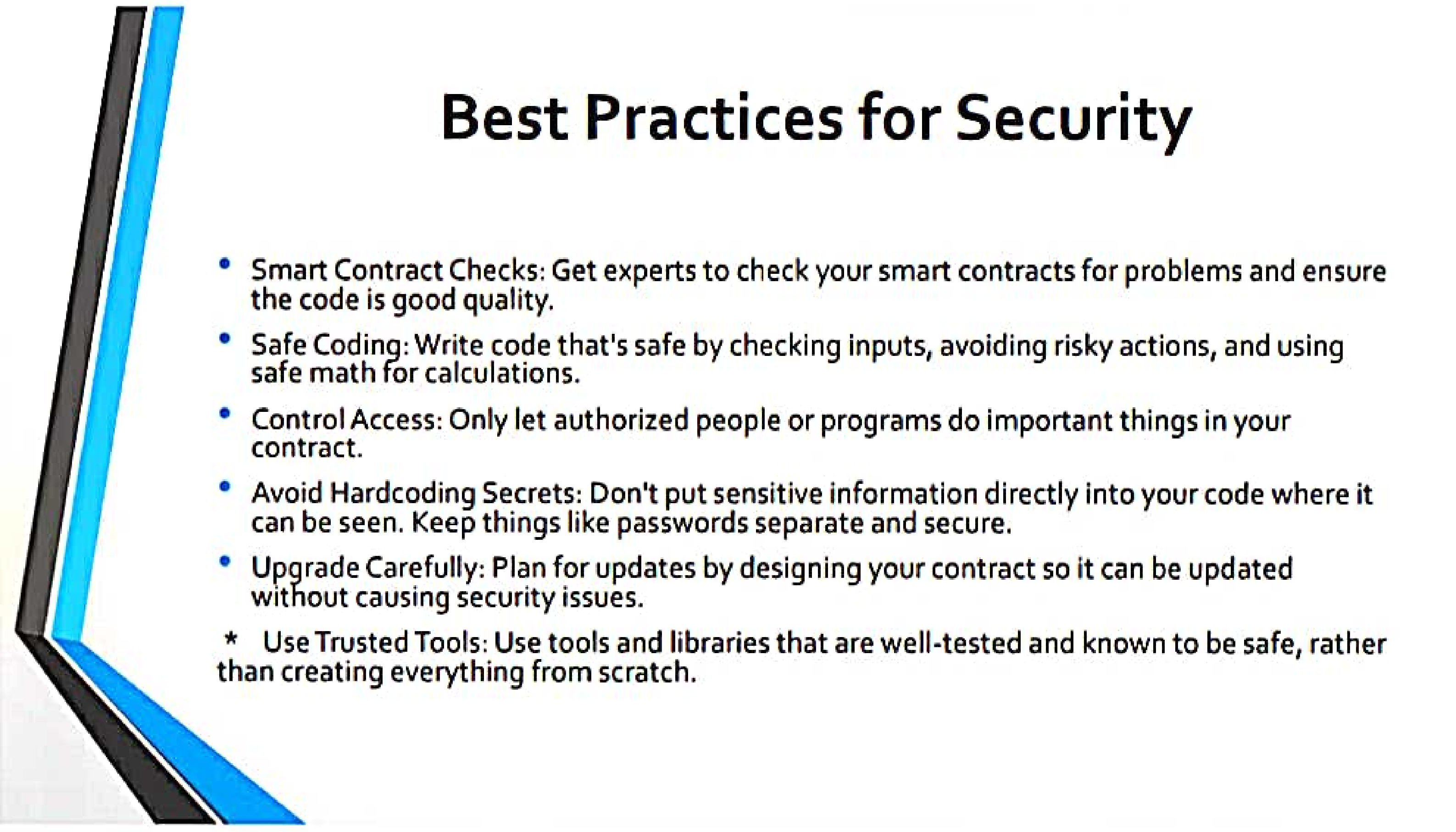
Building more Complex Smart Contracts

* Plan Your Contract: Decide what your smart contract should do .Break it down into smaller parts.

* Write the Code: Use Solidity to write the code for your contract. Include functions for things like transferring tokens or managing ownership.

* Keep it Secure: Follow security best practices to avoid problems like hacks or errors. Test your contract thoroughly for bugs.

* Make it Efficient : Optimize your contract to use less gas (transaction fees) on the blockchain. Simplify code and use efficient data structures.



Best Practices for Security

- Smart Contract Checks: Get experts to check your smart contracts for problems and ensure the code is good quality.
- Safe Coding: Write code that's safe by checking inputs, avoiding risky actions, and using safe math for calculations.
- Control Access: Only let authorized people or programs do important things in your contract.
- Avoid Hardcoding Secrets: Don't put sensitive information directly into your code where it can be seen. Keep things like passwords separate and secure.
- Upgrade Carefully: Plan for updates by designing your contract so it can be updated without causing security issues.

* Use Trusted Tools: Use tools and libraries that are well-tested and known to be safe, rather than creating everything from scratch.

Introduction to Truffle Sutie

- Truffle Framework: It's a toolbox for building Ethereum projects. You can create, test, and deploy smart contracts easily.
- Ganache: It's like a practice Ethereum network on your computer. You can test your contracts without spending real money.
- Truffle Boxes: These are starter packs for projects. They come with basic setups, so you don't have to start from scratch.
- Truffle Contract: Helps your code talk to smart contracts. It makes it easier to connect your front-end (like a website) to your smart contracts.
- Truffle Testing: You can write tests to check if your contracts work as expected. This helps catch bugs early.

THANK YOU!!!