Welcome to ineuron.ai

Full Stack Blockchain Development Tech Neuron

Description:

Full Stack Blockchain Development course is a live mentor-led certification program with by iNeuron. In this course you will learn the entire stack required to work in Permissionless Blockchain development. This course focuses on latest Blockchain industry standards like Ethereum Blockchain, Solidity, Decentralized Autonomous Organisations, Decentralized Finance, Non Fungible Tokens, Polygon Network, Polkadot Blockchain, Oracles along with complete development stack in Javascript and many more Blockchain concepts.

Start Date:

Doubt Clear Time: 08:00 PM to 10:00 PM (IST) Thursday

Course Time:

Features:

Course material

Course resources

On demand recorded videos

Practical exercises

Quizzes

Assignments

Course completion certificate

What we learn:

Web 1.0 vs Web 2.0 vs Web 3.0

What is Blockchain technology?

Bitcoin Blockchain

Ethereum Blockchain

Solidity

Oracles

DAO

DeFi

NFT

Layer 2 Blockchain

Truffle Suite

Hardhat

Polkadot

Requirements:

System with Internet Connection

Interest to learn

Dedication

Instructor:

>Course Introduction:

>>Course overview

>>A brief history of internet technologies

>>Web 1.0 vs Web 2.0 vs Web 3.0

>>What is Blockchain technology?

>>Why do we need Blockchain technology?

>>The connected world and the Blockchain: A disruptive computing paradigm

>>Centralized vs Decentralized networks

>>Distributed Systems overview

>Web Development:

>>What is Web Development?

>>Client-Server Architecture

>>What are APIs?

>>What is Front-end web development?

>>What is Back-end web devlopment?

>>Components of Full-Stack Web Development Applications

>HTML:

>>How do websites work?

>>Preview

>>HTML vs CSS vs Javascript

>>HTML files

>>Doctype & HTML Boilerplate

>>Spaces & Line Breaks

>>Heading Tag

>>Paragraph & Pre Tag

>>Difference between Elements, Attributes & Tags

>>Comments

>>Useful Tags

>>Nesting of Tags

>>Extensions in HTML

>>Live Server in VSCode

>>Formatting Tags

>>Article in HTML

>>Time & Address Tag

>>Quote & Cite

>>Strike

>>Progress Bar

>>Anchor Tag Styling

>>Image Tag

>>HTML Table

>>List

>>Input Tags,iframe

>>Forms

>>Video & Audio

>>iframe

>>Embed pdf

>>Maps

>>Symbols

>>Meta Tags

>>SVG

>>Emoji

>CSS:

>>CSS Introduction

>>Inline vs Internal vs External

>>Priority between Inline, Internal & External

>>Multiple Properties in Single Element

>>Types of Selectors

>>Priority between Id, Class & Element

>>Comments

>>Colors

>>Backgroud

>>Border

>>Height & Width

>>Padding

>>Margin

>>Box Model

>>Text Properties

>>Anchor Tag Styling

>>Fonts

>>Cursor

>>!Important in CSS

>>Box Shadow

>>Opacity

>>Filter

>>Gradient

>>Overflow

>>List

>>Tables

>>Box SIzing

>>Inherit & Initial

>>Object Fit

>>Pseudo Classes

>>Pseudo Elements

>>Display

>>Position

>>Z-Index

>>Floats

>>2D Transform

>>Transitions

>>Flex

>>Flex Direction & Wrap

>>Justify & Align in Flex

>>Order in Flex

>>Grow & Basis in Flex

>>Aling Items in Flex

>>Grids

>>Rows, Columns & Gap in Grids

>>Justify & Align in Grids

>>CSS Validator (Final Video)

>Javascript:

>>Introduction

>>Running Javascript in Browser

>>Console

>>Strings & Numbers

>>var, let & const

>>Data Types

>>Type Conversions

>>Arithmetic Operators

>>Assignment Operator

>>Comparision Operator

>>Logir Not, Or and And

>>Swap Numbers

>>String Handling

>>String Searching

>>Arrays

>>Objects

>>Dates

>>Maths

>>If & Else

>>Challenge - If & Else

>>Switch Case

>>Challenge - Switch Case

>>JS Loops

>>For Loops

>>Nested Loops

>>Break & Continue

>>Arrays, Strings & Objects

>>For-in

>>For-of

>>While Loops

>>Do while Loops

>>Loops Exercies

>>Functions

>>Variable Scopes in Functions

>>Nested Functions

>>Parameters & Arguments

>>How function is useful?

>>Return in Function

>>Anonymous Functions

>>Calculator Exercise

>>Arrow Functions

>>forEach

>>maps

>>String Literals

>>Filter, Reduce & Every

>>Spread Operator

>>Challenege

>>Window & Document

>>Document Access

>>innerText & innerHTML

>>HTML Calculator

>>Query Selector

>>Styling in JS

>>Advance DOM Manipulation

>>Events

>>Basic Events

>>Time Events

>>Pop-up Boxes

>>Error Handling

>>Form Validation

>>Asynchronous JS

>>this keyword

>>useStrict

>>Hoisting

>>Local Storage

>>Session Storage

>>Cookies

>>Cookies vs Local Storage vs Session Storage

>>JSON vs Object literals

>>API

>>Fetching

>>Methods & Status Codes

>>Post Method

>>Put Method

>>Guess the Number

>>Generators

>>Regex

>The JavaScript Standard Library:

>>The JavaScript Standard

>>Sets and Maps

>>Typed Arrays and Binary Data

>>Pattern Matching with Regular Expressions

>>Dates and Times

>>Error Classes

>>JSON Serialization and Parsing

>>The Internationalization API

>>The Console API

>>URL APIs

>>Timers

>Iterators and Generators:

>>What are Iterators and Generators?

>>How Iterators Work?

>>Implementing Iterable Objects

>>Generators

>>Advanced Generator Features

>Asynchronous JavaScript:

>>What is Asynchronous JavaScript?

>>Asynchronous Programming with Callbacks

>>Promises

>>Async and await

>>Asynchronous Iteration

>Working with Web Browsers:

>>JavaScript in Web Browsers

>>Web Programming Basics

>>Events

>>Scripting Documents

>>Scripting CSS

>>Document Geometry and Scrolling

>>Web Components

>>SVG: Scalable Vector Graphics

>>Audio APIs

>>Location, Navigation, and History

>>Networking Concepts

>>Storage

>>Worker Threads and Messaging

>Node js:

>>What is Node.js?

>>Client-Server Architecture

>>Single-Threaded Model

>>Multi-Threaded Model

>>Multi-Threaded vs Event-Driven

>>What is Node.js?

>>Node.js Features

>>Node.js Installation

>>Node.js First Example

>>Blocking vs Non-blocking

>>Global Objects

>>File System

>>Callbacks

>>Events

>>Node js Architecture

>>NPM(Node Package Manager)

>>Node.js Modules

>>Node.js Modules Types

>>Core Modules

>>Local Modules

>>3rd Party Modules

>>JSON File

>>Variables

>>Operators

>>Functions

>>Objects

>>File Systems

>>Events

>>HTTP Module

>>Creating a Web Server using Node.js

>>Node.js NPM Tutorial

>>What is NPM?

>>Main Functions of NPM

>>Need For NPM

>>NPM Packages

>>NPM Installation

>>JSON File

>>Node.js Express Tutorial

>>Introduction to Express.js

>>Features of Express.js

>>Getting Started with Express.js

>>Routing Methods

>>Building RESTful API with Node.js

>>What is REST API?

>>Features of REST API

>>Principles of REST API

>>Methods of REST API

>>Building REST API with Node.js

>>Contact List MERN App

>React JS:

>>Introduction to React

>>Why should you learn React?

>>Features of React

>>React applications

>>React App & JSX

>>Functional Components

>>Applying CSS Styles

>>Click Events

>>useState Hook

>>Lists & Keys

>>Props & Prop Drilling

>>Controlled Component Inputs

>>Project Challenge

>>useEffect Hook

>>JSON Server

>>Fetch API Data

>>CRUD Operations

>>Fetch Data Challenge

>>React Router

>>Router Hooks

>>Links

>>Flexbox Components

>>Axios API Requests

>>Custom React Hooks

>>Context API & useContext Hook

>>Build & Deploy Your React Apps

>Javascript Projects:

>>Creating shopping cart app with User Interface

>Bitcoin Blockchain:

>>History of currencies

>>Fiat currencies

>>Disadvantages of fiat currencies

>>Global financial system

>>How Central Banks work?

>>The 2008 Global Financial Crisis

>>Aftermath of 2008 recession

>>Creation of Bitcoin- A new decentralised digital currency

>>Bitcoin message hash implementation in Javascript

>>Immutable ledger practical implementation

>>Genesis block

>>Timestamp server

>>Merkel trees

>>Bitcoin as a State Transition System

>>Unspent Transaction outputs(UTXOs) Javascript implementation

>>Bitcoin whitepaper

>>What is a block?

>>Components of a Bitcoin block

>>Bitcoin Blockchain live implementation

>>Distributed Blockchain

>>Centralized vs Distributed Blockchain

>>consensus mechanism

>>Why do we need consensus mechanism in Blockchain networks?

>>Byzantine generals problem

>>Byzantine fault tolerance- A solution to Byzantine generals problem

>>BFT javascript implementation

>>Bitcoin nodes

>>Bitcoin miners

>>Blockchain mining operation

>>Mempool

>>Bitcoin difficulty adjustment

>>Bitcoin halving cycle

>>Competing chain problem

>>Maintaining immutability - Longest Chain rule

>>Block validation

>>consensus rules

>>Double Spend Validation

>>Transaction Input and Output Validation

>>Coinbase Transaction Reward Validation

>>Coinbase Maturity

>>Coinbase Transaction Block Height

>>Signature Check Counting

>>SigChecks

>>Mining incentive

>>Mining optimized hardware

>>CPU processing power

>>GPUs for mining

>>Application Specific Integrated Circuits(ASIC) miners

>>CPU vs GPU vs ASIC miners

>>Distributed peer to peer Blockchain live implementation

>>Distributed peer to peer Blockchain Javascript implementation

>>Token transaction live implementation using distributed peer to peer blockchain

>>Coinbase transaction live implementation using distributed peer to peer blockchain

>>Token and Coinbase transaction Javascript implementation

>>Bitcoin public key and private key

>>Public key and private key generation

>>Bitcoin addresses

>>Bitcoin digital signatures

>>Signing a peer to peer message with private key- Javascript implementation

>>Verifying peer to peer message using public key and digital signature-implementation

>>Signing and verifying currency transaction- implementation

>>Complete Bitcoin Blockchain implementation with transaction signatures

>Probable attacks in Bitcoin blockchain:

>>Sybil Attack

>>Race Attack

>>Finney Attack

>>Vector76 Attack

>>51% Attack

>Bitcoin Project:

>>Building a Blockchain using Javascript

>Ethereum Blockchain:

>>Module overview

>>Understanding the drawbacks of Bitcoin blockchain

>>Lack of Turing-completeness

>>Value-blindness

>>Lack of state

>>Blockchain-blindness

>>Origin of Ethereum- The programmable currency

>>The Decentralized Applications revolution and modern state of blockchain systems

>>Decentralized Applications vs Centralized Applications

>>Ethereum Accounts overview

>>Contract Accounts(CA)

>>Externally Owned Accounts(EOA)

>>Fields in Ethereum accounts

>>Ethereum Account messages

>>Ethereum Account transactions

>>Ethereum Addresses

>>Units of Ether

>>Ether Gas

>>Computing total gas cost for Ethereum transactions

>>Ethereum gas price Javascript implementation

>>Ethereum as a State Transition Function

>>Ethereum Architecture

>>Ethereum Virtual Machine(EVM)

>>EVM nodes vs mining nodes

>>EVM Bytecode

>>EVM Instruction Set

>>EVM Opcode

>>EVM Storage

>>EVM Memory

>>EVM Stack

>>Geth setup and EVM practical

>>Converting bytecode to opcode

>>Application Binary Interface(ABI)

>>Understanding end-to-end Ethereum Blockchain transaction in Javascript

>>Ethereum Smart Contracts architecture

>Ethereum 2.0:

>>Why was Ethereum 2.0 proposed?

>>Energy usage in Proof of Work

>>Gas costs in Ethereum 1.0

>>Potential scalability issues

>>Moving from Proof of Work to Proof of Stake

>>Proof of Stake in Ethereum 2.0

>>Validators

>>Staking

>>Attestation

>>Crosslinks

>>Finality

>>consensus clients

>>Execution clients

>>Sharding

>>Shard chains

>>Beacon chain

>>Data rollup in Ethereum 2.0

>>Forking in Blockchain

>>Hard Fork

>>Soft Fork

>>The DAO attack and Ethereum Hard Fork

>Solidity:

>>What is Solidity?

>>Why should you learn Solidity programming?

>>Introduction to Smart Contracts

>>Solidity Installation

>>Remix IDE

>>Installing Solidity in npm / Node.js

>>Layout of a Solidity Source File

>>SPDX License Identifier

>>Pragmas

>>Comments in Solidity

>>Structure of a Smart Contract

>Solidity Value Types:

>>Solidity datatypes

>>Booleans

>>Integers

>>Address Type

>>Address Literals

>>Contract Types

>>Byte Type

>>String Types

>>Enums in Solidity

>Solidity Reference Types:

>>Data locations- storage,memory and callback

>>Solidity Arrays

>>Fixed Arrays

>>Dynamic Arrays

>>Bytes and Strings as Arrays

>>Array Slicing

>>Structs

>>Mapping Types

>Solidity Units and Global Variables:

>>Ether Units

>>Time Units

>Solidity Control Structures:

>>If statement

>>If/else statement

>>Nested if/else statements

>>Solidity Loops

>>For loop

>>While loop

>>Do-while loop

>>Break statement

>>Continue statement

>ABI Encoding and Decoding Functions:

>>ABI encoder

>>ABI decoder

>Cryptographic Functions:

>>Keccack256

>>SHA256

>>Ripemd160

>>Ecrecover

>Smart Contracts:

>>Creating Smart Contracts

>>Constructor

>>Scope visibility

>>State variable visibility

>>Functions

>>Function visibility

>>Getter functions

>>Setter functions

>>Function modifiers

>>Return variables and returning multiple values

>>Immutable state variables

>>Payable functions

>>Fallback functions

>>View functions

>>Pure functions

>>Function overloading

>>Function overriding

>>Solidity Events

>>Block and Transaction details

>>Solidity Inheritance

>>Single Inheritance

>>Multiple Inheritance

>>Heirarchical Inheritance

>>Multilevel Inheritance

>>Abstract Contracts

>>Solidity Interfaces

>>Solidity Libraries

>Solidity Programming Applications:

>>Ether Wallet

>>Multi Sig Wallet

>>Iterable Mapping

>>ERC20

>>ERC721

>>Uni-directional Payment Channel

>>Bi-directional Payment Channel

>>NFT Auction

>>Crowd Fund

>>Time Lock

>Common Ethereum Blockchain Hacks and Loopholes:

>>Re-Entrancy Attack

>>Self Destruct

>>Accessing Private Data

>>Denial of Service

>>Phishing with tx.origin

>>Hiding Malicious Code with External Contract

>>Honeypot

>>Front Running

>>Block Timestamp Manipulation

>>Signature Replay

>>Bypass Contract Size Check

>Introduction to Blockchain Development Frameworks:

>>Introduction to Smart Contract Development in Production

>>Web3 libraries for Javascript

>>Smart Contract development tools

>>Web3 Providers

>>Wallets

>Truffle Suite:

>>Truffle overview

>>Truffle Installation

>>Creatin a new project in Truffle

>>Exploring project directories in Truffle

>>Compiling Smart Contracts

>>Building Artifacts

>>Handling Dependencies

>>Reading and writing Smart Contract data

>>Smart Contract Transactions in Truffle

>>Function calls in Truffle

>>Abstractions

>>Executing Contract functions

>>Making Transactions

>>Processing Transaction results

>>Catching events

>>Add a new contract to the network

>>Sending ether to a contract

>>Invoking overloaded methods

>>Using enumerations

>>Preserving Files and Content to Storage Platforms

>>Inter Planetary File System(IPFS)

>>Filecoin

>>Textile Buckets

>>Running Migrations

>>Initial Migration

>>Truffle Deployer

>>Network considerations

>>Truffle Deployer API

>>Integrating Truffle with Metamask

>>Using Truffle Dashboard

>>Using truffle Debugger

>>Truffle Develop and Truffle Console

>>Writing and executing external scripts

>>Testing Smart Contracts

>>Writing Automated Tests in Javascript

>>Writing Automated Tests in Solidity

>>Truffle Build Process

>>Truffle Boxes

>>Ethereum Name Service

>>Truffle Event System

>>Network Configuration and Dapp Deployment

>>Ganache- Ethereum Client for Truffle Suite

>>Installing Ganache

>>Ganache Workspaces

>>Ganache Ethereum Workspace

>>Understanding Workspace Default Configuration in Ganache

>>Managing Ganache configurations and settings

>>Configuring Truffle to connect to Ganache

>>Managing Truffle projects in Ganache

>>Exploring the Contracts page

>>Exploring the Transactions page

>>Linking and unlinking a Truffle project

>>Ganache Workspaces

>>Creating Workspaces

>>Deleting Workspaces

>>Editing Workspaces

>>Ethereum Workspace

>>Loading Existing Workspaces

>>Switching Workspaces

>Hardhat:

>>Introduction To Hardhat - Ethereum development environment for professionals

>>Hardhat Installation

>>Creating a Hardhat project

>>Configuring Ethereum Networks

>>Configuring the compiler

>>Compiling your contracts

>>Artifacts

>>Writing deployment scripts

>>Deploying the Contracts

>>Testing Smart Contracts

>>Running tests with Ganache

>>Running tests on Visual Studio Code

>>Running multiple tests in parallel

>>Running tasks

>>Hardhat Console

>>Creating custom tasks

>>Hardhat Runtime Environment(HRE)

>>Hardhat Plugins

>>Optimizing Plugins

>>Verbose Logging for debugging

>>Solutions to common runtime problems

>Web3.js:

>>Introduction to Web3.js

>>Why should you learn Web3.js?

>>Applications of Web3.js

>>Installing Web3.js using NPM

>>Web3 modules

>>Creating a new Web3 instance

>>Introduction to Web3 Providers

>>Setting up a Web3 Provider

>>Batch request

>>Extending Web3 modules

>>Introduction to Web3.eth

>>Checksum addresses overview

>>Fetching default blockchain details

>>Transaction methods

>>Block Node methods

>>Subscriber Methods

>>Web3.js Smart Contract objects and methods

>>User wallet and account methods

>>Interacting with Ethereum node accounts using web3.eth.personal

>>Working with ABI in web3.js

>>Commonly used utilities in web3.js

>>Hardhat automated testing with Web3.js and Truffle

>Ethers.js:

>>What is Ethers?

>>Ethers.js Features

>>Installing Ethers.js using NPM

>>Connecting to Ethereum: MetaMask

>>Connecting to Ethereum: RPC

>>Building blocks of Ethers.js- Signers,Providers and Contracts

>Ethers.js Providers:

>>What are Providers?

>>Ethers.js provider API overview

>>Provider Account methods

>>Blocks Methods

>>Ethereum Naming Service (ENS) Methods

>>EnsResolver

>>Logs Methods

>>Network Status Methods

>>Transactions Methods

>>Event Emitter Methods

>>Inspection Methods

>>BaseProvider

>>JsonRpcProvider

>>JsonRpcSigner

>>JsonRpcUncheckedSigner

>>StaticJsonRpcProvider

>>Node-Specific Methods

>>API Providers

>>EtherscanProvider

>>InfuraProvider

>>AlchemyProvider

>>CloudflareProvider

>>PocketProvider

>>AnkrProvider

>>Other Providers

>>FallbackProvider

>>IpcProvider

>>JsonRpcBatchProvider

>>UrlJsonRpcProvider

>>Web3Provider

>>WebSocketProvider

>Ethers.js Signers:

>>What are Signers?

>>Wallet Signer

>>JsonRPC Signer

>>Signer class and member functions

>>Ethers.js Wallet class and member functions

>>VoidSigner

>>Interacting with Externally Owned Accounts(EOA)

>Smart Contract Interaction:

>>Creating new Smart Contract instance

>>Contract Properties

>>Contract Methods

>>Events

>>ContractFactory

>>Creating ContractFactory Instances

>>ContractFactory Interface Properties

>>ContractFactory Methods

>>Meta-Class

>>Deploying a Contract

>>Connecting to a Contract

>>Properties

>>Methods

>>Events

>>Meta-Class Methods

>>Meta-Class Filters

>>Hardhat automated testing with Ether.js and Waffle

>Ethereum Blockchain Projects:

>>Building cryptocurrency with ICO

>>Building decentralized ecommerce website

>>Building decentralized voting application

>>Decentralized music sharing app

>>Token contract swap application

>>Full stack email dapp

>Oracles:

>>What is a Blockchain Oracle?

>>Solving the Oracle problem

>>Decentralized Oracles

>>Types of Blockchain Oracles

>>Applications of Blockchain oracles

>Chainlink overwiew:

>>Introduction to Chainlink

>>Understanding the Chainlink Ecosystem

>>Chainlink Features

>>Chainlink Applications as Decentralized Oracles

>>Chainlink Architecture

>>ERC677 Standard

>>The LINK token

>>Decentralized Data Model

>>Chainlink Off-chain Reporting

>>Chainlink Whitepaper

>Data Feeds:

>>Introduction to Data Feeds

>>Using Data Feeds

>>Fetchin Historical Cryptocurrency Price Data

>>Chainlink Feed Registry

>>Using ENS with Data Feeds

>>Contract Addresses

>>Ethereum Data Feeds

>>Binance Smart Chain Data Feeds

>>Polygon (Matic) Data Feeds

>>Gnosis Chain (xDai) Data Feeds

>>HECO Chain Data Feeds

>>Avalanche Data Feeds

>>Fantom Data Feeds

>>Arbitrum Data Feeds

>Chainlink VRF:

>>Introduction to Chainlink VRF(Verifiable Random Function)

>>Applications of randomness in Blockchain

>>Generating randomness

>>Some security considerations in Chainlink VRF

>>Smart Contract Integration

>Custom Data Feeds:

>>Using any API

>>Make a GET Request

>>Multi-Variable Responses

>>Large Responses

>>Make an Existing Job Request

>>Find Existing Jobs

>>Contract Addresses

>Chainlink Keepers:

>>Automating Smart Contracts

>>Introduction to Chainlink Keepers

>>Keepers Architecture

>>Keepers-compatible Contracts

>>Register an Upkeep

>>Manage your Upkeeps

>>Utility Contracts

>>EthBalanceMonitor

>>Supported Networks

>>Chainlink Keepers Economics

>Oracle Projects:

>>Live cryptocurrency trading using chainlink

>>Insurance Dapp using chainlink

>The Graph:

>>The Graph Protocol

>>The Graph architecture

>>Edge and Node

>>Everest Registry

>>Graph Protocol

>>The Graph vs Etherscan

>>Graph-cli Installation

>>Creating new subgraphs

>>Writing subgraphs

>>Publishing a Subgraph to the Decentralized Network

>GraphQL API:

>>Queries

>>Sorting

>>Pagination

>>Filtering

>>Time-travel queries

>>Fulltext Search Queries

>>Validation

>>Schema

>>Entities

>>Signalling

>>Curation

>>Delegators

>>Consumers

>>Deploying subgraphs

>>Subgraph logging

>>Graph protocol testnet using docker compose

>>Ethereum node monitoring using The Graph, Prometheus and Grafana

>The Graph Networking:

>>Introduction to indexers

>>Revenue streams

>>Distribution

>>Allocation life cycles

>>Querying and indexing subgraphs

>>IPFS Hash convertor

>AssemblyScript API for The Graph:

>>Installing AssemblyScript API

>>API Reference

>>Versions

>>Built-in Types

>>Store API

>>Ethereum API

>>Logging API

>>IPFS API

>>Crypto API

>>JSON API

>>Type Conversions Reference

>>Data Source Metadata

>>Entity and DataSourceContext

>The Graph Unit Testing:

>>Installing dependencies

>>WSL (Windows Subsystem for Linux)

>>Usage

>>CLI options

>>Docker

>>System Configuration

>>Demo subgraph

>>Asserts

>>Writing a Unit Test

>>Common test scenarios

>>Hydrating the store with a certain state

>>Calling a mapping function with an event

>>Calling all of the mappings with event fixtures

>>Mocking contract calls

>>Asserting the state of the store

>>Interacting with Event metadata

>>Asserting variable equality

>>Asserting that an Entity is not in the store

>>Printing the whole store (for debug purposes)

>>Expected failure

>>Logging

>>Testing derived fields

>>Testing dynamic data sources

>>Test Coverage

>Project:

>>Building a Full-stack Blockchain Application using Ethereum, Polygon,Next.js and Gra

>Decentralized Autonomous Organisations(DAO):

>>What are DAOs?

>>Why do we need DAOs?

>>DAO membership

>>Token-based membership

>>Share-based membership

>>How do DAOs work?

>>Properties of DAOs

>>Ethereum and DAOs

>>Understanding Governance Mechanisms

>>DAOs and the principal-agent problem

>>Building Decentralized Autonomous Organisations

>>Defining the DAO purpose

>>Building the DAO voting mechanism

>>Creating the governance token

>>DAO fund management

>>Initial Coin Offering (ICO)

>>Creating a DAO on Aragon

>>Creating a DAO using Snapshot

>>Building a DAO using DAOstack Alchemy

>Creating a Custom DAO Project:

>>Understanding custom DAOs

>>Finding the purpose for our Custom DAO

>>Designing the voting architecture

>>Implementing the voting architecture in Solidity

>>Designing the components of the governance token(DAO cryptocurrency)

>>Creating the governance token in Solidity

>>Fund Management for our custom DAO

>>Designing the Multi-signature wallet for Fund Management

>>Creating the Multi-signature wallet in Solidity

>>Testing DAO Smart Contracts

>>Deploying the DAO to testnet

>Decentralized Finance(DeFi):

>>The Traditional Financial Institutions

>>Centralization & Transparency

>>The Banks

>>General Public Accessibility

>>Decentralized Finance

>>The DeFi Ecosystem

>>How does DeFi work?

>>DeFi Categories

>>Decentralized Stablecoins

>>Lending and Borrowing

>>Decentralized Exchanges

>>Derivatives

>>Fund Management

>>Lottery

>>Decentralized payments systems

>>Insurance

>>Yield Farming

>>Liquidity Mining

>>Airdrops

>>Decentralized Prediction Markets

>Famous DeFi Protocols:

>>Aave

>>yEarn

>>Compound

>>Uniswap

>>Sushiswap

>>Maker

>>Numerai

>>Curve Finance

>>Alpha Finance

>DeFi projects:

>>Understanding DeFi Project Architecture

>>Components of Full-Stack DeFi applications

>>Designing DeFi project workflows

>>Building a Decentralized lottery system

>>Building a Decentralized borrowing and lending platform

>>Building a Decentralized stablecoin

>Non Fungible Tokens (NFT):

>>What is a Non Fungible Token(NFT)?

>>How does a NFT work?

>>Fungible Tokens vs Non Fungible Tokens

>>Exploring uses of NFTs

>>NFT as an internet of assets

>>NFT as a store of value

>>The Metaverse and NFT's role in it

>NFT Platforms:

>>What are NFT Platforms/Marketplaces?

>>CryptoKitties

>>Opensea

>>Rarible

>>Decentraland

>>Binance NFT

>>Enjin Marketplace

>>Axie Marketplace

>>Foundation

>>Nifty Gateway

>>Mintable

>>Theta Drop

>NFT Transaction Fees:

>>Gas Fees in NFT

>>What are one-time Gas Fees NFT?

>>Recurring Gas Fees

>>Actions in Gas Fees

>>Check Ethereum Gas Fee

>>Create and Sell NFTs without Gas Fees

>>NFT Marketpalce Fees

>NFT Programming:

>>Getting data for generating NFTs

>>Assigning trait rarity for digital assets

>>Classifying traits

>>Defining image traits

>>Validating uniqueness

>>Trait Counting

>>Generate the Images

>>Understanding NFT metadata

>>Uploading NFT images to IPFS

>>Generate NFT metadata

>>Upload the metadata to IPFS

>>Environment Setup for Smart Contract deployment

>>Creating Alchemy account

>>Writing NFT smart contract

>>Integrating Metamask, Alchemy and your Project

>NFT project:

>>Building a complete NFT Marketplace with User Interface

>Polygon Blockchain(MATIC):

>>Introduction to Polygon Blockchain

>>Why should you use Polygon network?

>>Layer 1 vs Layer 2 Blockchains

>>Features of Polygon Blockchain

>>Polygon Architecture

>>Zero-Knowledge cryptography

>>Zero-Knowledge rollups

>Polygon Network:

>>Introduction to Polygon Mainnet and Testnet

>>Mapped Tokens

>>Matic Gas Token

>>Genesis Contracts

>>Minimum Technical Requirements

>>Snapshot Instructions for Heimdall and Bor

>>Full Node Binaries

>>Full Node Deployment

>>Polygon Wallets

>>Arkane

>>Formatic

>>Metamask

>Polygon-Ethereum Bridge:

>>Introduction to Polygon POS bridge

>>Matic.js

>>Installing matic.js using NPM

>>Polygon Web3.js Setup

>>Polygon Ethers.js Setup

>>Supported libraries

>>Web3js setup

>>Ethers setup

>>Matic.js POS

>>Matic.js POSClient

>>Matic.js ERC20

>>Matic.js ERC721

>>Matic.js ERC1155

>>isCheckPointed

>>isDeposited

>>deposit ether

>>FxPortal

>>Set ProofApi

>Advanced Concepts:

>>ABIManager

>>Plugins

>>ExitUtil

>>PoS Bridge

>>Using Polygon Edge

>>Instantiating Polygon Edge

>>Deposit and Checkpoint Event Tracking

>>Deployment Details

>>Mapping Assets using POS

>>Tools

>>Wallet Widget

>>Submit Mapping Request

>>Polygon Mintable Assets

>>IPFS - Filecoin

>>Using IPFS

>>Using Filecoin

>>Mint with NFT.storage on Polygon

>>Polygon with Oracles

>>Chainlink integration

>Polygon Projects:

>>Retail supply chain Application using Polygon Network

>>Building a Social media Dapp on Polygon

>Polkadot:

>>Polkadot Overview

>>Polkadot Whitepaper

>>Polkadot Architecture

>>Parachains

>>Parathreads

>>Substrate Installation

>Substrate Fundamentals:

>>Runtime environment and setup

>>Extrinsics

>>Account Abstractions

>>Transaction Pool

>>Session Keys

>>Transaction Weight

>>Execution

>>Off-Chain Features

>Runtime Development:

>>Frames

>>Macros

>>Metadata

>>Storage

>>Origins

>>Events and Errors

>>Weights and Fees

>>Benchmarking

>>Debugging

>>Testing

>>Randomness

>>Chain Specification

>>Upgrades

>>Pallet Coupling

>>Custom RPCs

>>Smart Contract Toolkits

>Development Integration:

>>Polkadot-JS

>>Client Libraries

>>Substrate Connect

>Development Tools:

>>SR tool

>>Subxt

>>Tx Wrapper

>>Sub Flood

>>Substrate Archive

>>Sidecar

>>Polkadot Launch

>Advanced topics in Polkadot:

>>Account Info

>>SCALE Codec for Substrate

>>Consensus

>>Block Import

>>Executor

>>Cryptography

>>Storage

>>SS58 Address Format

>>Hash Collections