**CODELANDCS BLOCKCHAIN DEVELOPMENT SYLLABUS**

**WEEK 2**

**DAY 3**

**DECENTRALISED APPLICATIONS DEVELOPMENT**

Good afternoon everyone, today we are going to talk about Decentralised Applications development. DApps, also known as decentralized applications, are applications that run on a blockchain network, enabling the creation of trustless, secure, and transparent applications. In this lecture, we will explore the basics of DApps development, including what they are, how they work, and the technologies involved.

First, let's define what a DApp is. A DApp is an application that operates on a decentralized network. **Unlike traditional applications that run on a centralized server,** DApps are built on top of a **blockchain network,** making them decentralized and transparent. DApps can be used for a variety of purposes, including decentralized finance, gaming, supply chain management, and social media, among others.

Now, let's discuss how DApps work. DApps **consist of two main components: the frontend and the backend.** The frontend is the user interface that the user interacts with, while the backend is the code that runs on the blockchain. The backend code is executed by the nodes on the blockchain network, and the results are stored on the blockchain.

To develop a DApp, developers use a variety of **technologies, including blockchain networks, programming languages,** and **development frameworks**. Some popular blockchain networks used in DApp development include **Ethereum,** **EOS,** and **TRON**. These blockchain networks use smart contracts, which are self-executing contracts with the terms of the agreement between the buyer and the seller being directly written into lines of code. Developers write the backend code for their DApp using programming languages like **Solidity,** which is used to write smart contracts for the Ethereum network.

Another critical technology used in DApp development is development frameworks. These frameworks are used to simplify the development process and make it easier for developers to write DApps. Some popular development frameworks include **Truffle,** **Hardhat, Embark** etc which are used to create and deploy smart contracts on the Ethereum network.

Once the DApp has been developed, it needs to be deployed on the blockchain network. To deploy a DApp, developers use the frameworks aforementioned to test and deploy smart contracts on the Ethereum network. These frameworks also provide developers with the ability to interact with the smart contracts and test their functionalities.

In conclusion, DApps development is a complex process that requires a deep understanding of blockchain technology, programming languages, and development frameworks. However, the benefits of building decentralized applications are enormous, including increased security, transparency, and trustlessness. As the blockchain industry continues to grow, the demand for skilled DApp developers will only increase, making it an exciting and lucrative field to work in.

**DEVELOPMENT ENVIRONMENT SETUP OPTIONS**

As decentralized applications (dApps) have been growing in popularity over the past few years due to their unique features such as transparency, security, and decentralization. Developing dApps requires a different approach compared to traditional web or mobile application development. Some of the different dApp development environment setup options that developers can use to build dApps are as follows:

**Ethereum Virtual Machine (EVM)**

The EVM is a virtual machine that runs on the Ethereum network, which allows developers to write and deploy smart contracts that can be executed on the blockchain. The EVM is the most popular platform for developing dApps due to its widespread adoption and the large number of tools and resources available. Developers can use the EVM to develop dApps using programming languages such as Solidity, Vyper, and LLL.

**Web3.js**

Web3.js is a JavaScript library that allows developers to interact with the Ethereum network from their web applications. It provides a simple and easy-to-use API for developers to interact with smart contracts deployed on the blockchain. Developers can use Web3.js to build decentralized applications using their existing web development skills.

**Truffle Suite**

The Truffle Suite is a development environment for building and deploying dApps on the Ethereum network. It provides a set of tools that simplify the development process, such as a development framework, a testing framework, and a deployment framework. The Truffle Suite is an excellent option for developers who are new to dApp development or those who want to streamline their development process.

**Embark**

Embark is another development environment for building dApps on the Ethereum network. It provides a set of tools that enable developers to develop, test, and deploy smart contracts and decentralized applications. Embark is a good option for developers who prefer a more flexible and customizable development environment.

**Ganache**

Ganache is a personal blockchain for Ethereum development that provides a local blockchain environment for developers to test their smart contracts and dApps. It is a great tool for developers who want to test their code in a controlled environment without interacting with the live blockchain network.

**Remix IDE**

Remix IDE is a browser-based development environment for building dApps on the Ethereum network. It provides a user-friendly interface for developers to write, compile, and deploy smart contracts. Remix IDE is an excellent option for developers who prefer a browser-based development environment or those who are new to dApp development.

**Hardhat**

Hardhat is an open-source development environment used to build and deploy smart contracts on the Ethereum blockchain. It provides a comprehensive suite of tools that help developers to write, test, and deploy smart contracts more efficiently.

Some of the key features of Hardhat include**:**

* Built-in support for popular Ethereum development frameworks such as Truffle and Buidler.
* A local Ethereum network for testing and debugging smart contracts.
* Automated contract testing with Mocha and Chai.
* Contract deployment and management.
* Built-in task runner for automating common development tasks.
* Integration with popular Ethereum development tools such as Remix and MetaMask.

Hardhat is widely used in the Ethereum developer community, and it is considered a valuable tool for building and deploying smart contracts.

**KEY DEVELOPER TOOLS TO BE USED**

To create and deploy these applications, developers require specific tools and frameworks to help them code, test, and deploy their projects. Here are some key developer tools we would be using

* Solidity
* Remix
* Web3.js
* OpenZeppelin
* Hardhat
* Nodejs