Introduction :

My name is Shahil Gupta. I am a 3rd year UG student of the department of Mechanical Engineering enrolled in its B. Tech course. I am from hazaribagh Jharkhand.

In my 1st year I experiment with everything from CP to web development which led to my learning more about it in my second year. Current, I am focusing on competitive programming and CS fundamentals such as OOPs, networking etc.

Project 1 : Decentralized Supply Chain Management

The project aimed to create a decentralized supply chain management system for online transaction using blockchain, mainly Ethereum Blockchain.

The motive was to enhance transparency and trust in supply chain by utilizing smart contracts, digital tokens and decentralized application.

Concepts :

1. Blockchain

Blockchain is a distributed and immutable ledger that records transactions across a network of computers in a secure and transparent manner.

Each set of transaction is called a block and is linked to the previous one forming a chain of blocks.

This chain of blocks is maintained by a decentralized network of nodes that reach agrement on validity of transaction through cryptographic algorithms.

2. Ethereum and Smart Contracts

Ethereum is a blockchain platform that goes beyond cryptocurrency and enables creation of DApps and smart contract.

- Bitcoin primarily serves as a digital currency for transactions but Ethereum provides a decentralized computing platform for executing code on Ethereum Virtual machine (EVM).

Smart Contract :

- These are self executing codes with predefined rules and conditions. (they executes on EVM)
- These contracts are deployed on blockchain (eg: Ethereum) and automatically execute when specific conditions are made.
- It eliminates the need of any intermediaries.
- Whenever a transaction is sent, EVM processes the code, performs computations and updates the blockchain state. They will reach consensus based on the contract's outcome.

Solidity :

→ High level language
→ used to write smart contracts on the ethereum platform.

Solidity code specifies how contracts should behave when interacting with users or other contracts.

Key components

(a) State variable : variable representing contract's state and are store in blockchain.

(b) Constructor : Initializes the contract's state when it is deployed.

(c) Functions : These defines the behaviour of contract and can be called by users or other contracts.

(d) Modifiers : These are used to change the behaviour of functions

(e) Events : Used to record important contract interactions and data changes.

(vi) Revert and Throw : Contracts can revert transaction by calling revert().

used for handle errors.

Defi Concepts: (Decentralized finance) Defi refers to a blockchain-based financial ecosystem that aims to recreate traditional financial services without centralized intermediaries like bank and brokers.

Q) How ethereum-based applications are build?

Ans: We require certain tools & libraries

i) Remix : Online IDE for Solidity smart contract development.

ii) Truffle : Development framework for contract compilation. and testing

iii) Hardhat : A task runner and development environment for Ethereum that simplifies smart contract testing and deployment.

iv) Metamask : A browser extension for Ethereum wallet management and DApp interaction.

Q15: What is the role of the front-end in a DApp?

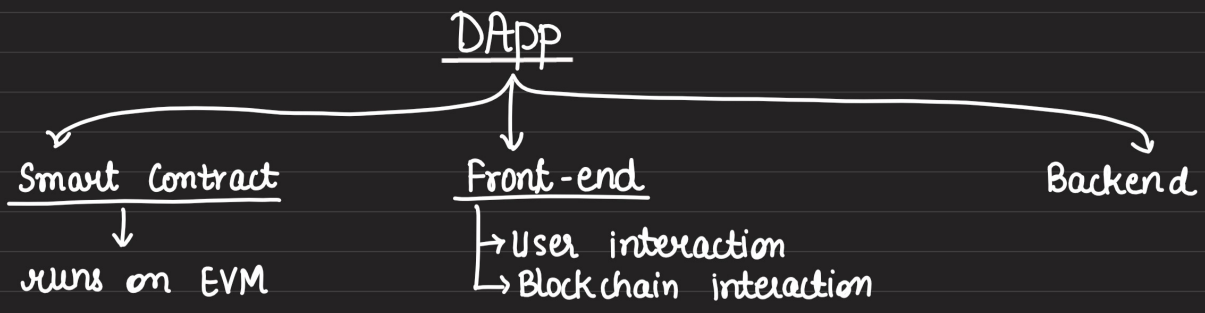
A15: The front-end of a DApp is responsible for presenting the user interface and allowing users to interact with the underlying smart contracts and blockchain. It performs several key functions:

User Interaction: It provides a user-friendly interface for users to interact with the DApp, such as submitting transactions, reading data, and managing their accounts.

Blockchain Interaction: The front-end communicates with the blockchain using libraries like Web3.js or Ethers.js to query data and send transactions to smart contracts.

Security: It ensures that user inputs are validated and that interactions with the blockchain are secure and tamper-proof.

User Experience: The front-end is responsible for creating an intuitive and responsive user experience, making it easy for users to navigate and use the DApp.



In my first project, I created a decentralized application for online transactions. The main purpose of the project is to eliminate the need for intermediaries, for ex: banks, in our daily transactions.

Let me start by explaining blockchain, which is a distributed and immutable digital ledger that records transactions over a network of nodes. Each set of transactions is known as a block, and every block is linked to the previous one, thus making a chain.

Any decentralized application has two parts. The frontend part and the blockchain part. The frontend part is designed using HTML, CSS, and JavaScript. In the frontend part, we enter the addresses and amount of transaction which is used by the blockchain part. For the blockchain part, I have used Ethereum blockchain technology to create the network of distributed nodes, which is known as the Ethereum Virtual Machine. There are different networks that are available for use; the one I used is the Goerli Ethereum test network.

Now, to validate a transaction and remove intermediaries, we need smart contracts and a digital wallet. Smart contracts are self-running codes that execute once certain conditions are met. So, suppose you want to send 10 RS to some node but you have only 5 rupees, then we cannot do the transaction. If you enter the wrong receiver address, then we cannot make the payment. So, these things are handled by the smart contract code and the digital wallet. For the digital wallet, I have used the Meta-Mask Chrome Extension, which also helps in selecting the network through which you want to make the transaction.
For testing the code, I have used the Hardhat development framework, which ensures the reliability of the decentralized application, mainly the smart contract

Project - 2 : Code-Search

Code-Search is a website where one can practice DSA problems by doing the most relavant problem. Here, one can search any topic and the top 10 problems of that particular topic shows up.

For frontend, I used HTML, CSS and Javascript ensuring user-friendly nature of the website.

I utilized BS4 and Selenium python libraries to perform web-scraping. I used them to extract data mainly the problem name, statement and solution. from external sources such as leetcode.

Once the user enters any topic, the site starts searching for the top 10 most relavant problems from the extracted data. The algorithm I used for searching is TF-IDF (Term frequency - Inverse document frequency) algorithm.

TF-IDF Algorithm :

- It is the calculation of how relevant a word is in a document when we have say N different documents.
- **Term Frequency :** Frequency of the searched term t in a document, d

$$tf(t,d) = \frac{\text{count of } t \text{ in } d}{\text{number of words in } d}$$

- **Document Frequency** $N(t)$: The number of documents in which t is present
- **Inverse document frequency :** Measure of how common the word is to the N document.

$$IDF = \frac{N}{N(t)} = \frac{\text{total no. of documents}}{\text{no. of documents containing } t}$$

for a document, $TF\text{-}IDF(t,d) = tf(t,d) \cdot \log(IDF)$