ABOUT:

Smart contract are line of instruction or code which contains some agreement between users in order to exchange currency or assets in **blockchain technology** .This smart contract is about **buying and selling items** between parties(users) .The benefit of using blockchain technology in daily transaction over normal transaction is that it is decentralised that means there is no central authority like bank and any third party included in our transaction.

TOOLS:

In this project I used tools which are basically software platform

1**.solidity**  OOP language to write my contract which is OOP language based on javascript syntax and also has concept of C++ and PYTHON

2.This smart contract which i wrote on **remix.etherium** platform and deployed that contract with ether wallet(account) on that same platform (remix.ethereum.org) with **Injected Web 3 enviroment.**

3.**metamask extension** in order to deploy my contract through metamask account and for transfer corresponding ethers for given item to another ether wallet.

4.In order to test my contract I used **rinkeby test network** which is for testing purpose of smart contract on ether transaction network .It is similar to bank which is like [SBI,UCO,OBC and many more] But here the difference is that we can use same account number on multiple networks.

OVERVIEW:

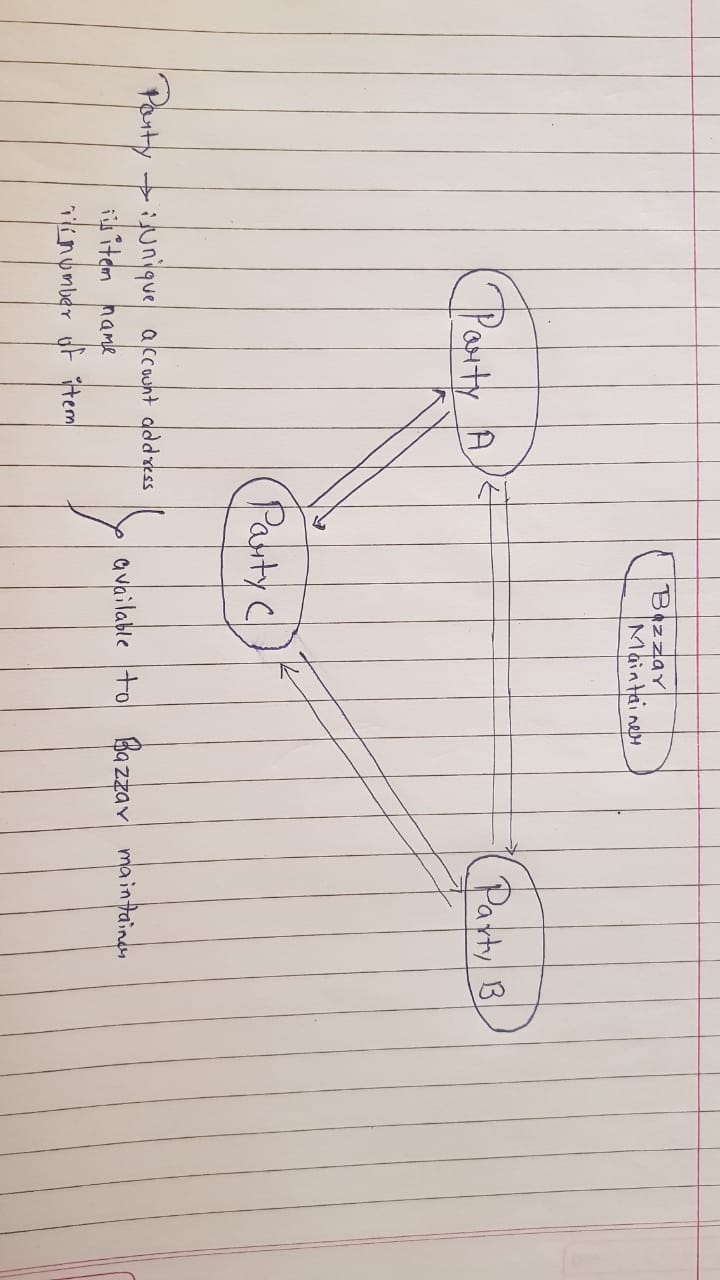
In that contract parties are identified by their unique address which is basically their account address(ether wallet) which contains ethers in order to buy and sell items between them.

In that contract which i wrote for –three parties— to allow them for exchanging their items .Parties consist of items with its name and ether wallet(similar to bank account).Each parties has their unique account address(wallet address) which is similar to this given below example. eg :[0x99cDac8d12B8Fc6C02aaf195F295980e8ff30a34].

In this contract there is bazzar maintainer which suppose to know the account address of all parties which want to join this contract and the **bazzar maintainer have fix amount of space for number of parties** .Bazzar maintainer deploy the contract corresponding to **account address of parties** .After deploying the contract it will generate transaction hash for that contract and *it will allow only those parties to take part in contract* which initially gave their account address to **bazzar maintainer** .

For verifying the contract on **rinkeby.etherscan.io** we need contract address , solidity files and compiler version which we used in writing our code.

In order to test contract I used rinkeby test network for verifying my contract as bazzar maintainer After verifying contract it allow parties(friends) which gave me their ether wallet address during deploying of smart contract to become part of this **blockchain** smart contract.



Here in the above picture I tried to show you basic demonstration of selling item between parties at some fixed ether price depend on cost.

FUNCTIONALITY OF CODE:

After verifying the contract which consist of two solidity files on **rinkeby test network .** Now the parties will come and add their details such as first name, last name, identity number, name of item, number of items for selling their products. In order to buy and sell items between parties t**hey should know wallet address of each other .**In that code I add some functionality which is **publically accessible** in such manner that without knowing their account address anyone who is not part of that smart contract can access it .Such variables are:

1. ***buytoken*** which tell how many items through that smart contract are exchanged.
2. ***number of person*** in that smart contract that tells number of users .
3. And ***information*** about parties like their id, first name and last name just by considering it as array by giving input as 0,1…n it will pop up information of parties.
4. Similar to above it also show ***parties record*** initially when they add themselves in that contract just by previous method.

Now the functionality of above code in transaction is under function named ***buynsell*** which need address payable wallet, payable ether, number of items buying, name of item and unique transaction id which buyer will set during transaction. The transaction id for each transaction will tell the details of number of item and name of item exchanged.

**NOTE:** The limit on number of user in that smart contract to exchange their items can be increase or decrease by following these below methods:

1. Add variables in contract ***address PartyD***

2. Add parameter in constructor for address of added party to be part of this deployed smart contract.

3. Add **if** condition in **addperson()** function for corresponding party address . Now the party become the part of smart contract.

4. Add one **if** condition in ***buynsell()*** function in order to transaction between parties in these decentralised network.