CS577 - Blockchain and Cryptocurrency

Project ID 2: **Smart Farming using Blockchain Technologies**

Group No. 12

Student	Roll No.	Batch
Kolaparthi Vamsi	1801CS28	B.Tech.
M. Maheeth Reddy	1801CS31	B.Tech.
M. Nitesh Reddy	1801CS32	B.Tech.

The Project

We, the members of Group 12, have made a Decentralized Application for Food Traceability with a Micro-Finance functionality. This DApp tracks a farm product from the farmer to the consumer. Since farmers in developing countries like India are heavily dependent on bank loans, we have introduced a Micro-finance feature through which customers can provide small amount of funds to the farmers. We believe the small funds from various customers would be sufficient enough for the farmer to sustain their farming.

Steps to start the App

Step 1: **Create a private Ethereum network for the DApp**

For this, we are using Ganache. By making just one-click, Ganache automatically sets up a private network with appropriate parameters and creates several Ethereum accounts for testing.

Step 2: **Install/Update Dependencies**

Open a terminal in the project folder and execute the ***npm install*** command. This command will install or update the dependencies of the DApp.

Step 3: **Deploy the Smart Contract**

In the terminal enter this command ***truffle migrate***. This will deploy the smart contract in the contracts folder of the project, in our private network.

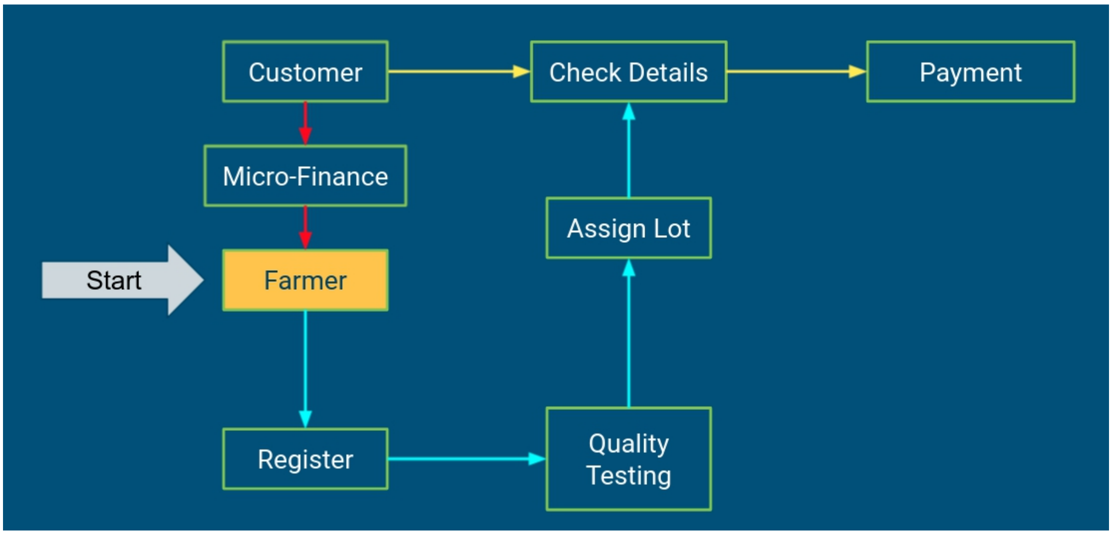
The output of ***truffle migrate*** will display a contract address. Copy this contract address into the app.js file in src folder. This file is responsible for all communications of the Web frontend with the blockchain.

Step 4: **Start the DApp**

The final step is to run the command "**npm run dev**" which will automatically open the DApp in the default web browser. The app will be running on port 3000 on localhost.

Architecture of the DApp

The process of Smart Farming begins with the Farmer uploading their personal info and product info into the blockchain. Then the Quality Testing agent will approve the product details and assign a lot number. Next, Customers will be able to choose the Farmer's product and make the payment. Finally, a Farmer, can ask his/her customers to provide some funds. Customers can optionally send the funds to the Farmers via the Micro-Finance feature.



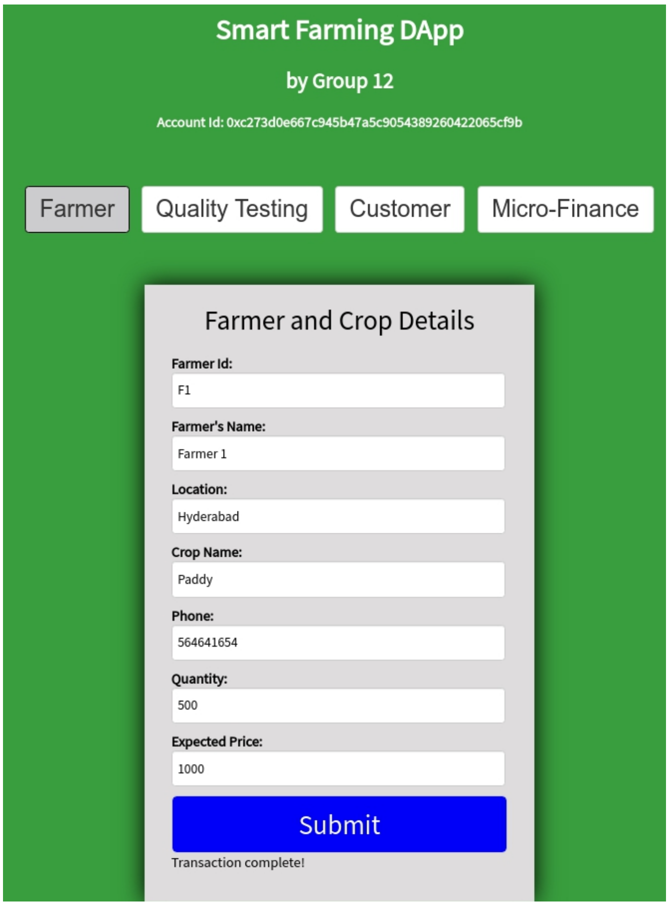
Tech Stack

For implementing this project, we have used:

- **HTML, CSS, JavaScript**: For frontend of the DApp
- **Solidity**: For writing the smart contract
- **Ganache**: For starting an Ethereum private network. The project can also be run using Geth. But in Ganache we have an option in which the configuration of the private network is done automatically. If someone wants to run the project using Geth, they will have to configure the network manually.
- **Truffle**: For compiling smart contracts using the ***truffle migrate*** command.
- **Web3.js**: In order to facilitate communication between the blockchain and the DApp
- **npm**: For installing dependencies

Description of the App

The adjacent screenshot shows the registration form for the farmer in the supply chain application. The entered details are stored directly onto the blockchain. The underlying technology uses Truffle for the deployment and Ganache as the backend blockchain. We use the Web3 Javascript provider API to interact with the blockchain.



Account Id: 0xc273d0e667c945b47a5c9054389260422065cf9b

Farmer and Crop Details

Farmer Id: F1

Farmer's Name: Farmer 1

Location: Hyderabad

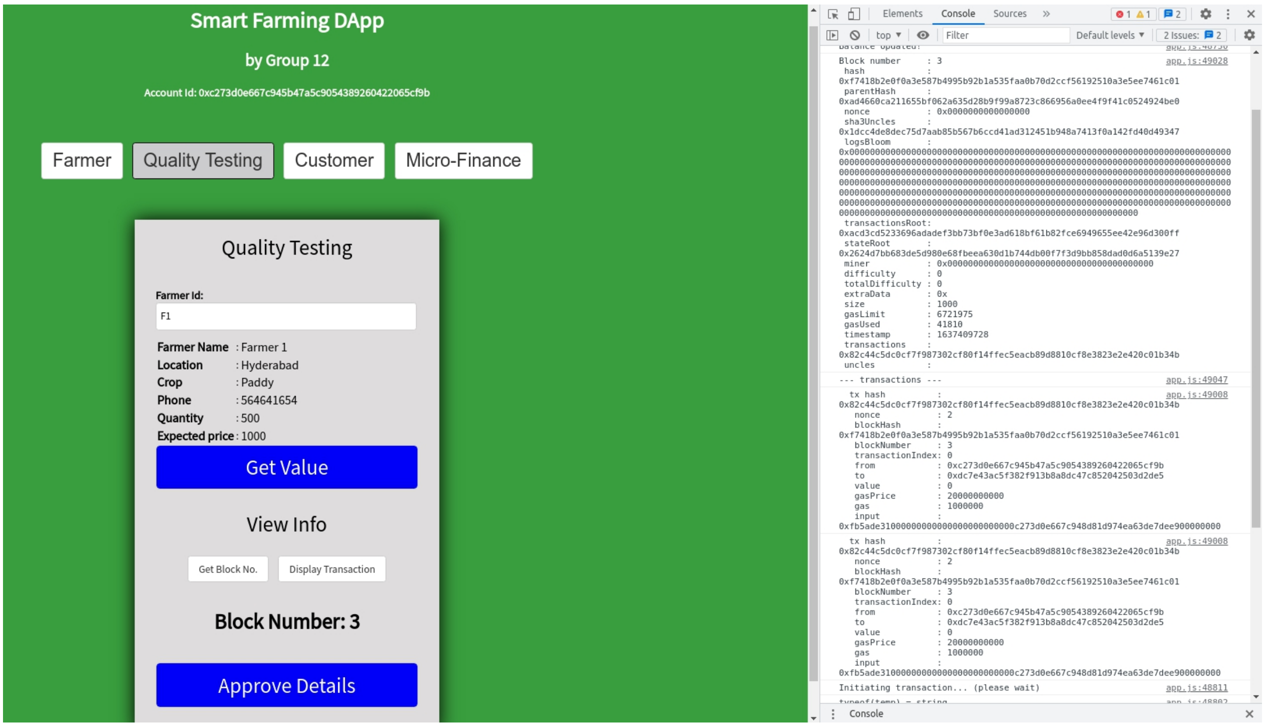
Crop Name: Paddy

Phone: 564641654

Quantity: 500

Expected Price: 1000

Transaction complete!



Farmer Id: F1

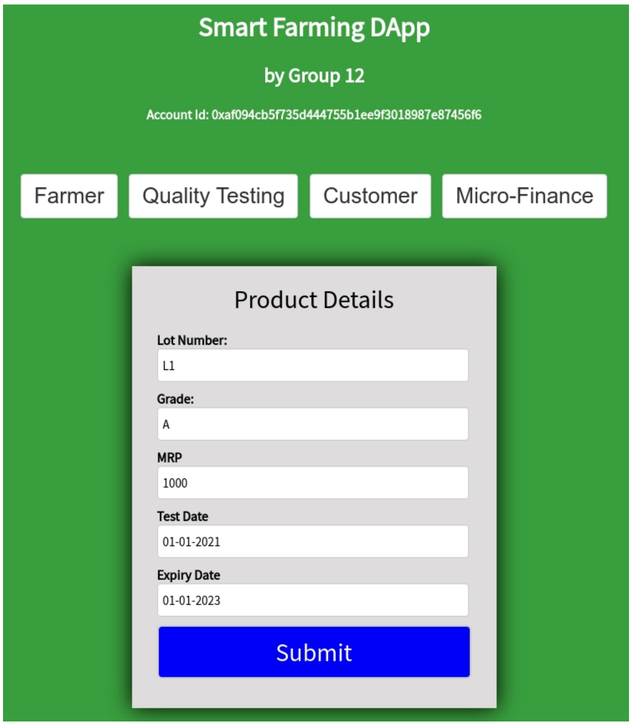
Farmer Name : Farmer 1
Location : Hyderabad
Crop : Paddy
Phone : 564641654
Quantity : 500
Expected price : 1000

Block Number: 3

```
Block number      : 3                                                  app.js:49028
 hash             :
0xf7418b2e0f0a3e587b4995b92b1a535faa0b70d2ccf56192510a3e5ee7461c01
 parentHash       :
0xad4660ca211655bf062a635d28b9f99a8723c866956a0ee4f9f41c0524924be0
 nonce            : 0x0000000000000000
 sha3Uncles       :
0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347
 logsBloom        :
0x0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000
 transactionsRoot:
0xacd3cd5233696adadef3bb73bf0e3ad618bf61b82fce6949655ee42e96d300ff
 stateRoot        :
0x2624d7bb683de5d980e68fbeea630d1b744db00f7f3d9bb858dad0d6a5139e27
 miner            : 0x0000000000000000000000000000000000000000
 difficulty       : 0
 totalDifficulty : 0
 extraData        : 0x
 size             : 1000
 gasLimit         : 6721975
 gasUsed          : 41810
 timestamp        : 1637409728
 transactions     :
0x82c44c5dc0cf7f987302cf80f14ffec5eacb89d8810cf8e3823e2e420c01b34b
 uncles           :
--- transactions ---                                                   app.js:49047
  tx hash           :                                                  app.js:49008
0x82c44c5dc0cf7f987302cf80f14ffec5eacb89d8810cf8e3823e2e420c01b34b
   nonce            : 2
   blockHash        :
0xf7418b2e0f0a3e587b4995b92b1a535faa0b70d2ccf56192510a3e5ee7461c01
   blockNumber      : 3
   transactionIndex: 0
   from             : 0xc273d0e667c945b47a5c9054389260422065cf9b
   to               : 0xdc7e43ac5f382f913b8a8dc47c852042503d2de5
   value            : 0
   gasPrice         : 20000000000
   gas              : 1000000
   input            :
0xfb5ade310000000000000000000000000c273d0e667c948d81d974ea63de7dee900000000
  tx hash           :                                                  app.js:49008
0x82c44c5dc0cf7f987302cf80f14ffec5eacb89d8810cf8e3823e2e420c01b34b
   nonce            : 2
   blockHash        :
0xf7418b2e0f0a3e587b4995b92b1a535faa0b70d2ccf56192510a3e5ee7461c01
   blockNumber      : 3
   transactionIndex: 0
   from             : 0xc273d0e667c945b47a5c9054389260422065cf9b
   to               : 0xdc7e43ac5f382f913b8a8dc47c852042503d2de5
   value            : 0
   gasPrice         : 20000000000
   gas              : 1000000
   input            :
0xfb5ade310000000000000000000000000c273d0e667c948d81d974ea63de7dee900000000
Initiating transaction... (please wait)                                app.js:48811
```

The next page is for quality testing, here we can get the farmer details by Farmer Id. These details are stored as a structure using solidity code onto the blockchain. The Farmer details are retrieved using a special data structure called mapping by Farmer Id. In the above screenshot, we can see the block details where the farmer's details are stored onto blockchain. The 'Approve Details' will approve the details of the farmer. We have included two functionalities: Get Block No. and Display Transaction. We can get the block and transaction details in which the farmer's info was uploaded in the blockchain. You can see the info in the Console of the browser.

Approve Details button click will redirect to this Product details page. This is also part of the quality testing where we enter the lot number, grade, price, test date and expiry date. These details are also stored in the blockchain as a structure.



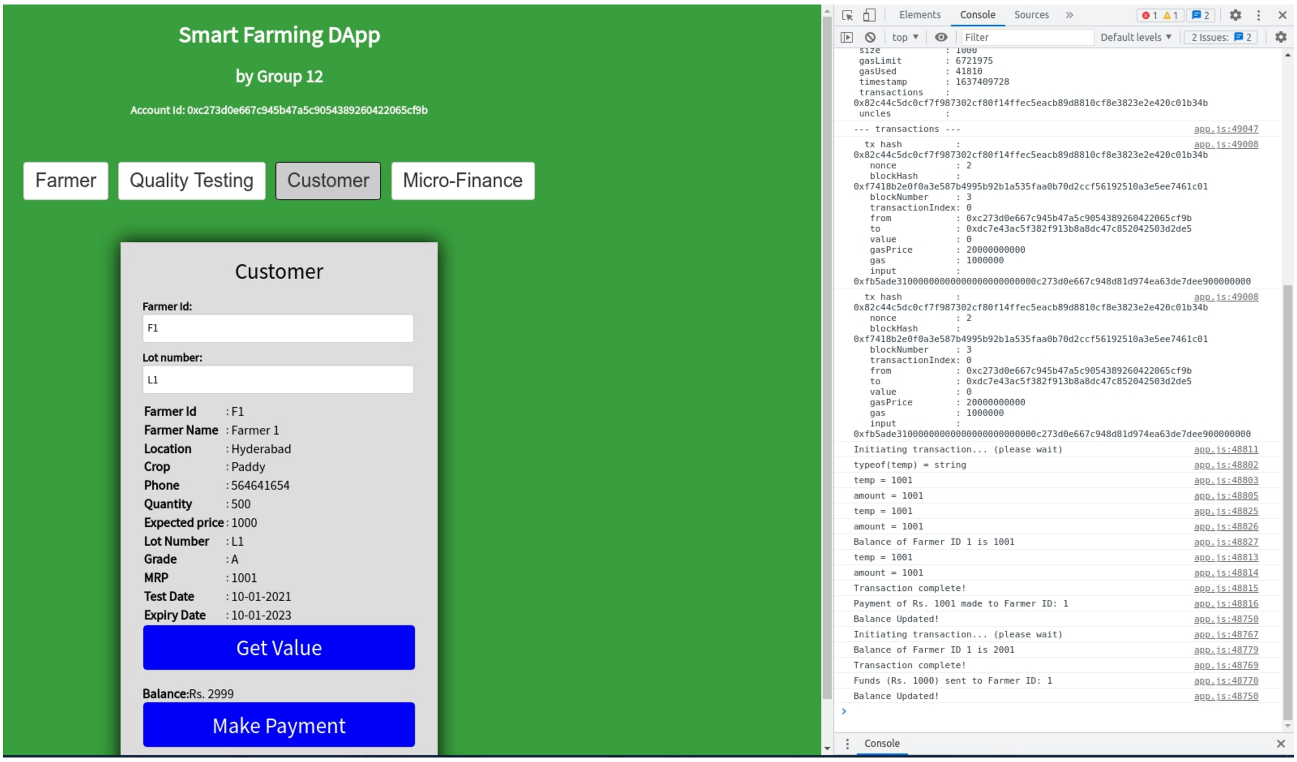
Lot Number: L1

Grade: A

MRP 1000

Test Date 01-01-2021

Expiry Date 01-01-2023



Account Id: 0xc273d0e667c945b47a5c9054389260422065cf9b

Customer

Farmer Id:

F1

Lot number:

L1

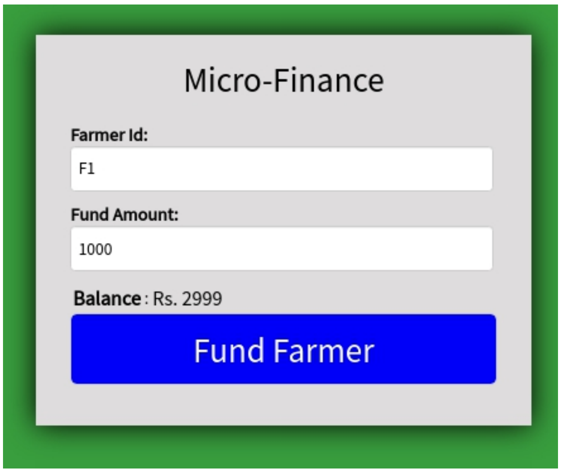
Farmer Id	: F1
Farmer Name	: Farmer 1
Location	: Hyderabad
Crop	: Paddy
Phone	: 564641654
Quantity	: 500
Expected price	: 1000
Lot Number	: L1
Grade	: A
MRP	: 1001
Test Date	: 10-01-2021
Expiry Date	: 10-01-2023

Balance: Rs. 2999

```
  size          : 1000
  gasLimit      : 6721975
  gasUsed       : 41810
  timestamp     : 1637409728
  transactions  :
0x82c44c5dc0cf7f987302cf80f14ffec5eacb89d8810cf8e3823e2e420c01b34b
  uncles        :
--- transactions ---                                          app.js:49047
   tx hash          :                                         app.js:49008
0x82c44c5dc0cf7f987302cf80f14ffec5eacb89d8810cf8e3823e2e420c01b34b
    nonce           : 2
    blockHash       :
0xf7418b2e0f0a3e587b4995b92b1a535faa0b70d2ccf56192510a3e5ee7461c01
    blockNumber     : 3
    transactionIndex: 0
    from            : 0xc273d0e667c945b47a5c9054389260422065cf9b
    to              : 0xdc7e43ac5f382f913b8a8dc47c852042503d2de5
    value           : 0
    gasPrice        : 20000000000
    gas             : 1000000
    input           :
0xfb5ade310000000000000000000000000c273d0e667c948d81d974ea63de7dee900000000
   tx hash          :                                         app.js:49008
0x82c44c5dc0cf7f987302cf80f14ffec5eacb89d8810cf8e3823e2e420c01b34b
    nonce           : 2
    blockHash       :
0xf7418b2e0f0a3e587b4995b92b1a535faa0b70d2ccf56192510a3e5ee7461c01
    blockNumber     : 3
    transactionIndex: 0
    from            : 0xc273d0e667c945b47a5c9054389260422065cf9b
    to              : 0xdc7e43ac5f382f913b8a8dc47c852042503d2de5
    value           : 0
    gasPrice        : 20000000000
    gas             : 1000000
    input           :
0xfb5ade310000000000000000000000000c273d0e667c948d81d974ea63de7dee900000000
Initiating transaction... (please wait)                       app.js:48811
typeof(temp) = string                                         app.js:48802
temp = 1001                                                   app.js:48803
amount = 1001                                                 app.js:48805
temp = 1001                                                   app.js:48825
amount = 1001                                                 app.js:48826
Balance of Farmer ID 1 is 1001                                app.js:48827
temp = 1001                                                   app.js:48813
amount = 1001                                                 app.js:48814
Transaction complete!                                         app.js:48815
Payment of Rs. 1001 made to Farmer ID: 1                      app.js:48816
Balance Updated!                                              app.js:48750
Initiating transaction... (please wait)                       app.js:48767
Balance of Farmer ID 1 is 2001                                app.js:48779
Transaction complete!                                         app.js:48769
Funds (Rs. 1000) sent to Farmer ID: 1                         app.js:48770
Balance Updated!                                              app.js:48750
>
```

This is the Customer page where the customer can check the customer details and status of the quality testing of his agriculture produce. The customer has to enter the farmer id and Lot number to see the details. These customer details are retrieved from the blockchain. If thecustomer is okay with the price of the product, they can make the payment. The updated Customer balance is shown in the App and the updated Farmer balance can be seen in the Console.

The micro-finance form shown here enables any user to fund a farmer. The funding is done by providing the Farmer's ID and the amount.



Micro-Finance

Farmer Id:

F1

Fund Amount:

1000

Balance : Rs. 2999

----------x---------x---------