Civil Supplies Chain- A Blockchain Adapted Method for Civil Supplies Department

About The Project

The Civil Supplies Department discharges the important responsibilities of Public Distribution, enforcement of markets discipline and promotion of consumer awareness and protection of their interest. In the 60s and 70s it won many accolades for the pioneering achievements in the implementation of Universal Rationing System.The Department of Civil Supplies functions under the Department of Food, Civil Supplies and Consumer Affairs of the Government of Kerala.

Kerala State has the privilege of having the best system of Public Distribution in the whole country. The system is evenly spread over the whole state without any distinction between urban or rural. To ensure equitable distribution of the food grains at a fairly low cost to all people especially to the weaker sections of the population. There is a very good net work of wholesale and retail outlets for the distribution of rationed articles under the Public Distribution System. In each Taluk, with every AWD and KWD, a set of ARDs are permanently linked. Under this system, the requirement of each AWD and KWD in the State is fixed and this facilitates the District Supply Officer who is the Departmental Head of the District to make allocation of the Rationed Articles to the Taluks in each district and the Taluk Supply Officer in turn to make allocation to the AWDs and KWDs in his Taluk. State Average of Ration cards per Ration shop is 400. On the supply side, there are 14203 retail outlets at present, in the state.

PROBLEM

The documentation process in Civil Supplies Department is mostly done by both by paper and centralised server. The server failure issues are very often in some places, and these issues causes interruption for the working of Ration Shops, as each retailer have to record each purchase by consumers to their card, and server using a POS machine.

A ration card is a document which is used as address proof and income proof. So poor handling of ration card data using paper documents will cause many troubles. A centralised server may cause data breach also.

Solution

I am using blockchain technology to counter the problems faced by centralised server and paper. In this method we store the data in a decentralised ledger using Hyperledger fabric. Each data is stored in blockchain, so the data will be safe and tamper-proof. And we can verify the authenticity of these data as well.

There are 4 types of ration cards are there in Kerala. They are

1. Antyodaya Anna Yojana (AAY cards – Yellow colour)
2. Priority House Holds (PHH cards – Pink colour)
3. Non-Priority with State Subsidy (NPSS cards – Blue colour)
4. Non-Priority Non-Subsidy (NPNS cards – White colour)

The categorization of these cards is based on several criterias. But, since it is a small project, I am basing these grouping only upon total family income.

In Kerala, the first choice for the head of the family in ration card is for the eldest women in the family. If there is no women in the family, or there is no women with age greater than 18 years, only then a man can become the head of the family. Then the chance is for the eldest man. If there is no men in the family, and there is no women who is older than 18 years, then the eldest women can become the family head, even if she is less than 18 years.

Workflow

Here in this project, the working of the department is divided into 3 organizations, They are Commisioner, Nodal Officer, and Ration Retailer.

Commisioner

The commisioner organization is entitled with most important operations in the network. They are the one who is responsible for the operations on the other two organizations. i.e. creation, updation, and deletion of these organization accounts.

And also only the commisioner can do operations under the jurisdiction of nodal officers, which involve multiple nodal officers.

Nodal Officer

Nodal officer is responsible for the procedures such as creating ration card, adding consumer to a ration card etc. A nodal officer account have to keep track of ration cards and the consumers which have under it’s authority. It is essential for them, because it helps to calculate the accurate number of goods to make purchase.

Ration Retailer

A ration retailer does the goods purchase for the consumers, and record the consumer purchase data in blockchain.

Workflow Diagram

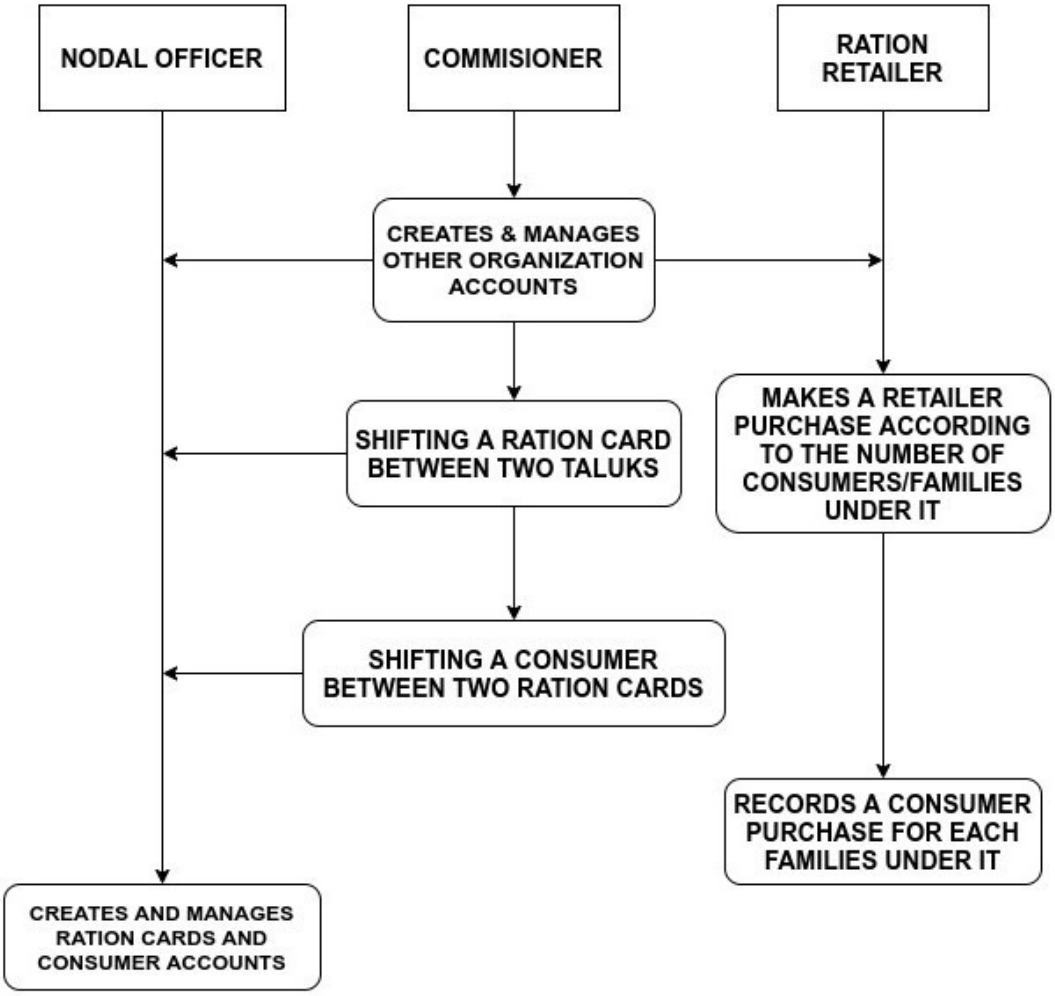
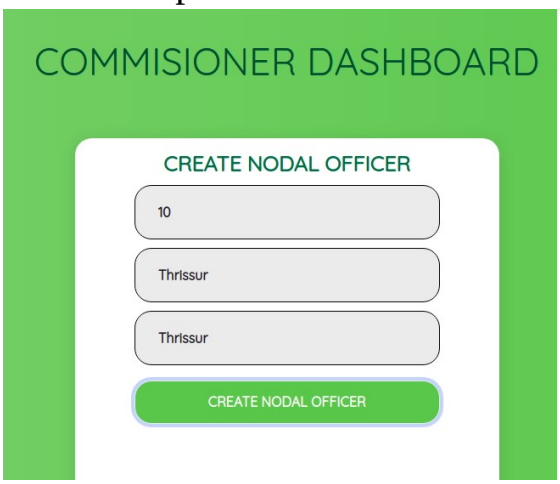


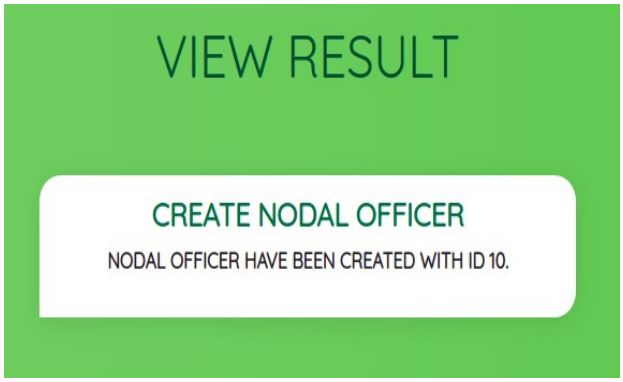
FIG 1 : Work flow diagram of the civil supplies chain project

Smart Contract Functions

Commisioner Functions

- Create Nodal Officer
  - Input **:** Nodal officer ID, district, taluk
  - Output **:** Saves the data with key as nodal officer ID





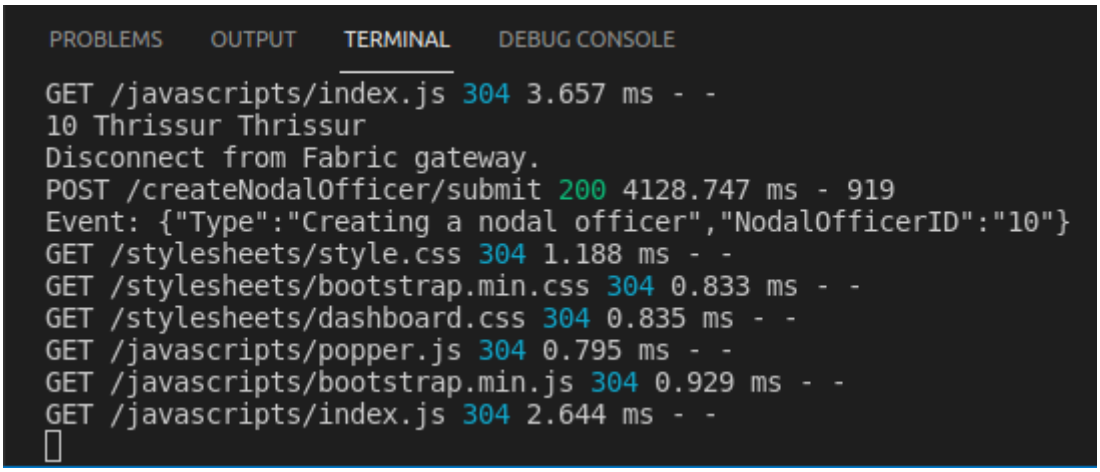
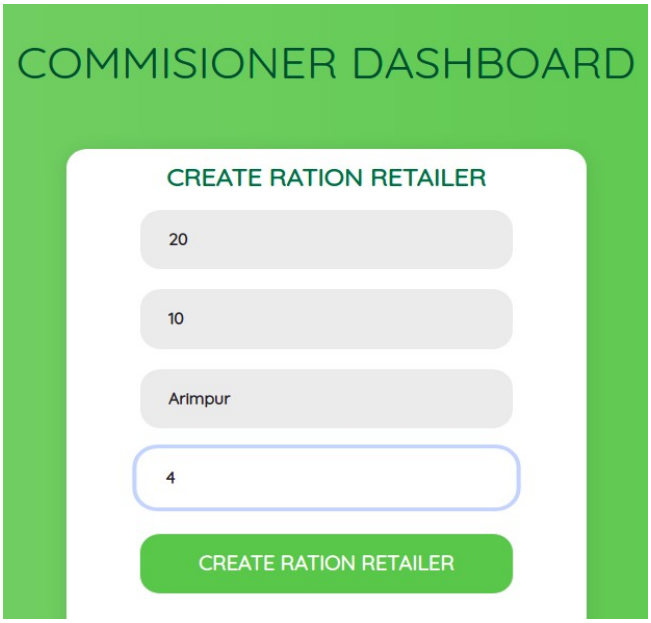
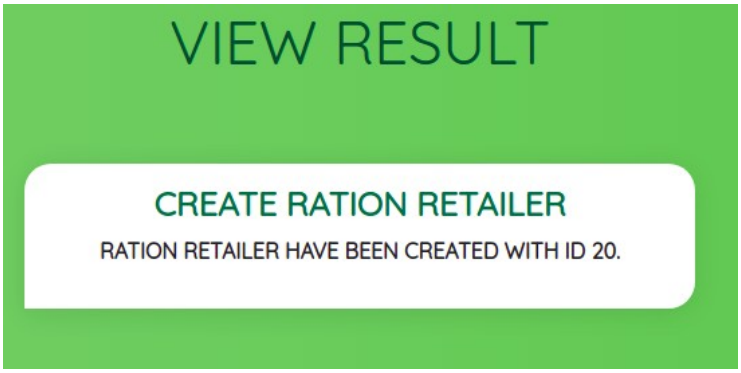


FIG 2 : Working of create nodal officer function in the dapp

- Update Nodal Officer
  - Input **:** Nodal officer ID, new district, new taluk
  - Output **:** Updates the values.

- Delete Nodal Officer
  - Input : Nodal officer ID
  - Output : Delete the value.
- Create Ration Retailer
  - Input : Ration retailer ID, nodal officer ID, LSG (Local Self Governance) body, ward number
  - Output : Saves the data with key as ration retailer ID. Initially sets the number of yellow card families, yellow card consumers, pink card consumers, blue card consumers, white card consumers, homes with electricity and homes without electricity as 0.





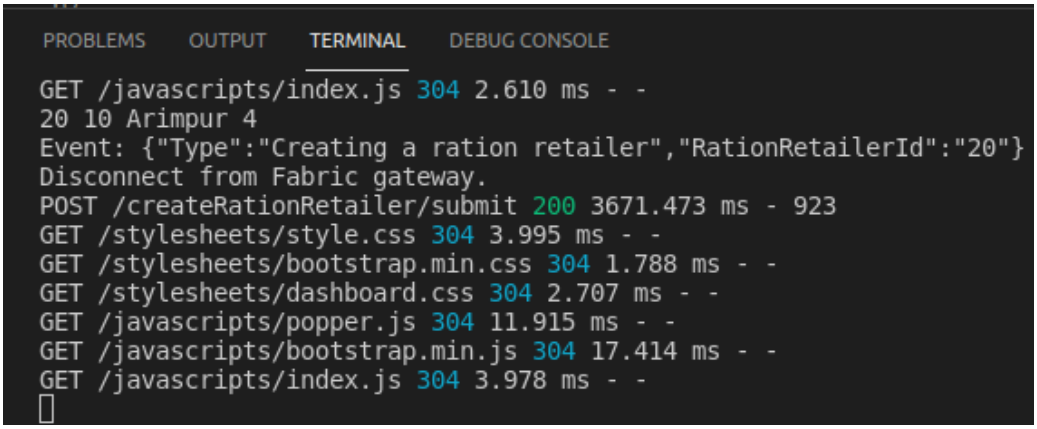
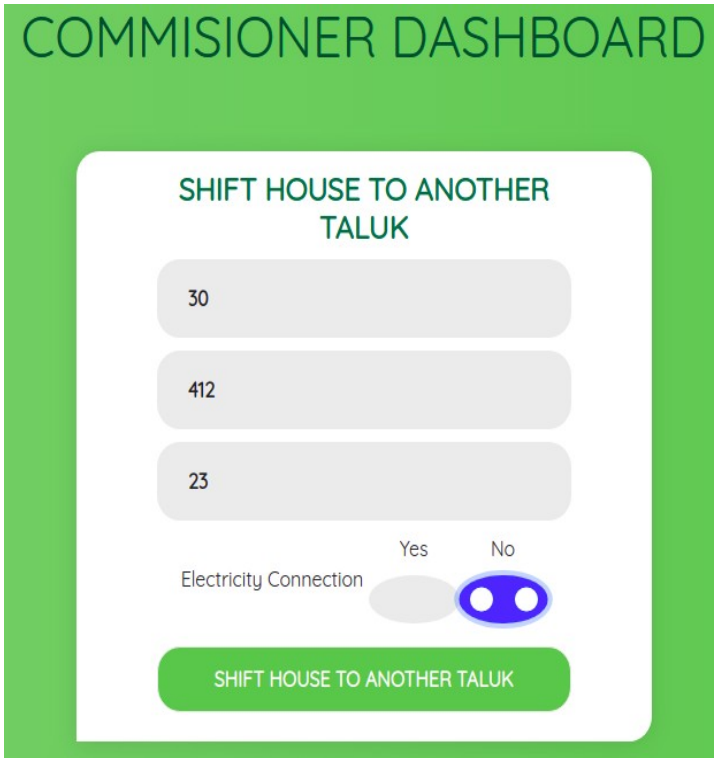
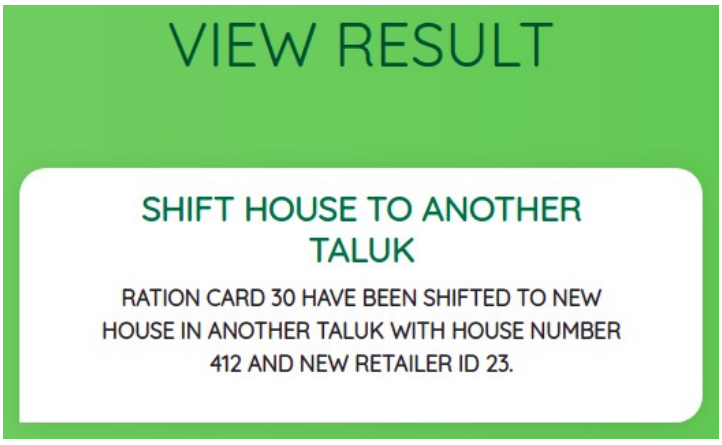


FIG 2 : Working of create ration retailer function in the dapp

- Update Ration Retailer
  - Input : Ration retailer ID, new nodal officer ID, new LSG body, new ward number
  - Output : Updates the value.
- Delete Ration Retailer
  - Input : Ration retailer ID
  - Output : Delete the value.

- Shift House To Another Taluk
  - Input : Ration card number, new house number, new ration retailer ID, is new home electrified
  - Output : Updates the ration retailer ID, nodal officer ID, house number, home electrification status with new value. Updates the consumers, and family numbers at both retailer's data

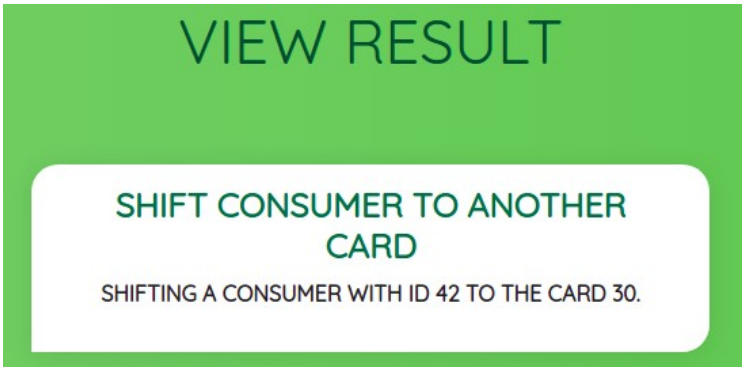




```
[Object: null prototype] { rationCardNumber: '30' }
Disconnect from Fabric gateway.
POST /changeElectricityConnection/submit 200 3965.174 ms - 946
Event: {"Type":"Changing electricity connection status for a ration card","RationCardNumber":"30"}
GET /stylesheets/style.css 304 0.981 ms - -
GET /stylesheets/bootstrap.min.css 304 1.353 ms - -
GET /stylesheets/dashboard.css 304 15.007 ms - -
GET /javascripts/popper.js 304 3.305 ms - -
GET /javascripts/bootstrap.min.js 304 6.173 ms - -
GET /javascripts/index.js 304 5.406 ms - -
```

FIG 2 : Working of shift family to another taluk function in the dapp

- Shift User To Another Ration Card
  - Input : Consumer number, new ration card number.
  - Output : Updates ration card number, ration retailer ID, nodal officer Id at consumer data. If that consumer was the head of the current family, determine the next head of the family based on gender and age. Check if the new member is eligible to become the family head. If yes, then replace the current leader with the new one.

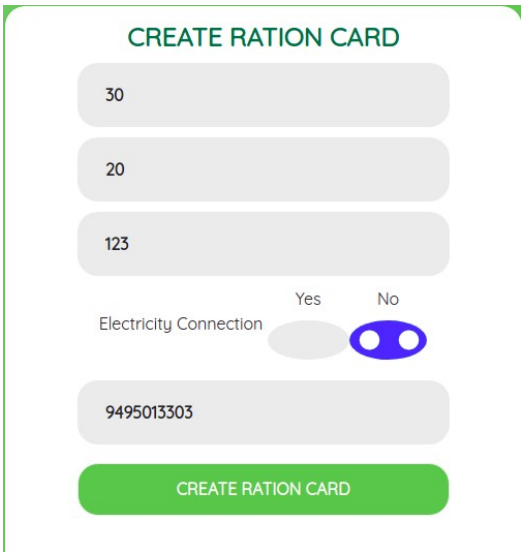


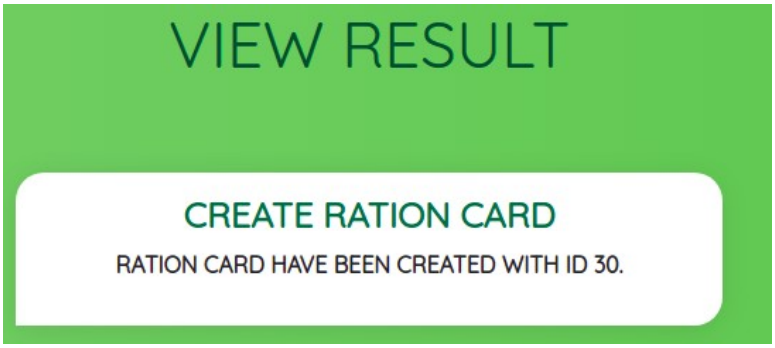

```
[Object: null prototype] {
  consumerNumber: '42',
  newRationRetailerId: '30'
}
Event: {"Type":"Shifting one consumer to another family","ConsumerNumber":"42"}
Disconnect from Fabric gateway.
POST /shiftConsumerToAnotherRationCard/submit 200 3999.475 ms - 932
GET /stylesheets/style.css 304 1.566 ms - -
GET /stylesheets/bootstrap.min.css 304 0.931 ms - -
GET /stylesheets/dashboard.css 304 0.748 ms - -
GET /javascripts/popper.js 304 0.800 ms - -
GET /javascripts/bootstrap.min.js 304 0.736 ms - -
GET /javascripts/index.js 304 0.814 ms - -
```

FIG 2 : Working of shifting consumer to another card function in the dapp

Nodal Officer Functions

- Create Ration Card
  - Input : Ration card number, ration retailer ID, house number, is home electrified status, mobile number
  - Output : Create the data with ration card number as key. Initially total family members, and total family income will be 0. The family head place will be null, since there is no members included. And the type of the card will also become null since there is no family members, and no family income.





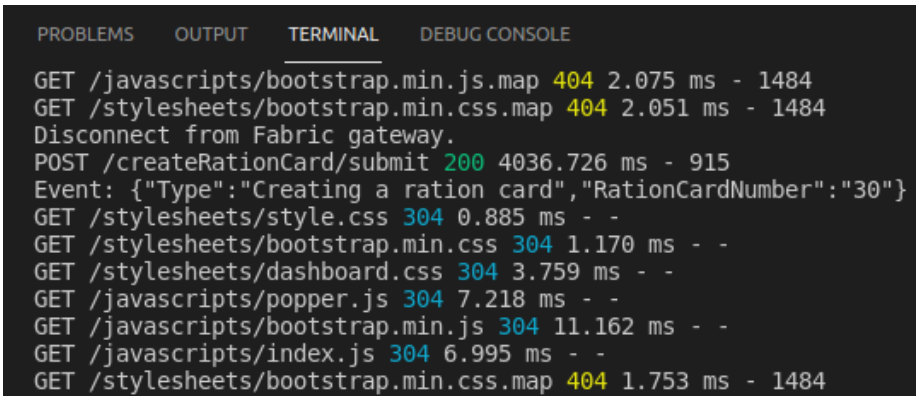
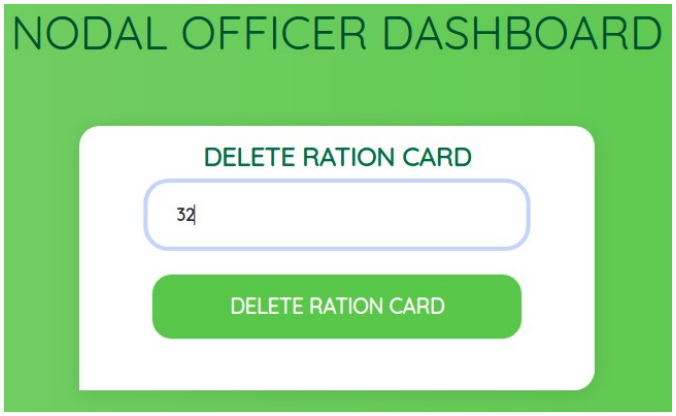
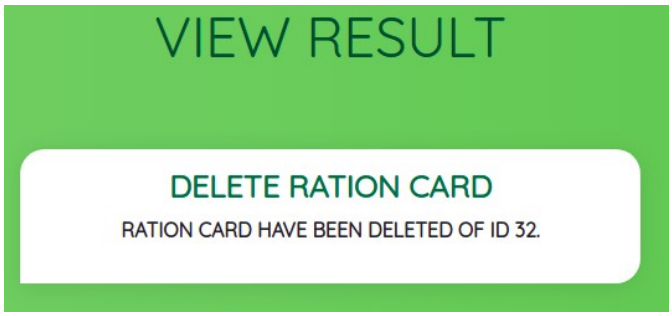


FIG 2 : Working of create ration card function in the dapp

- Delete Ration Card
  - Input : Ration card number
  - Output : Deletes the data, the number of consumers from this family is decreased from the ration retailer data.





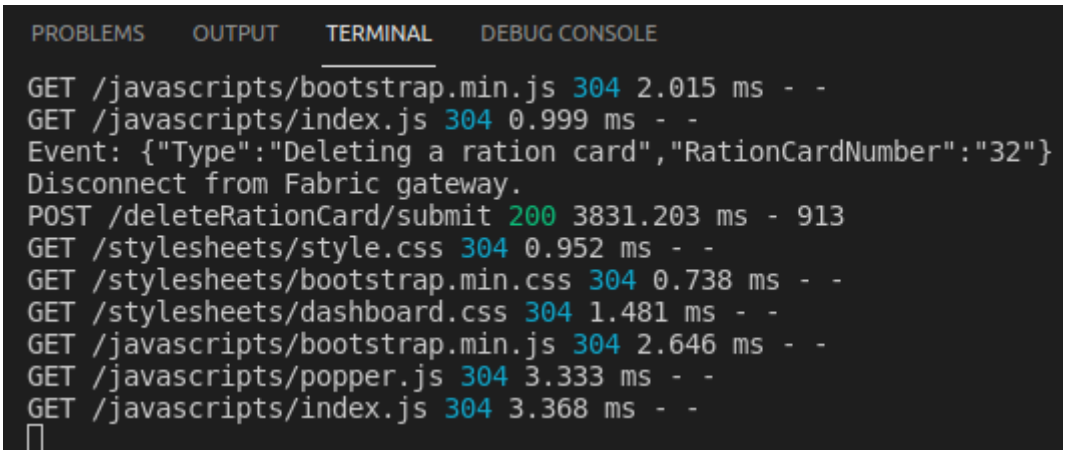
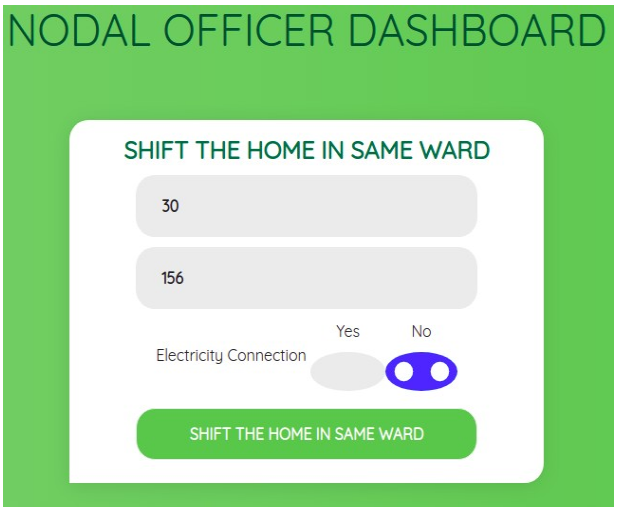
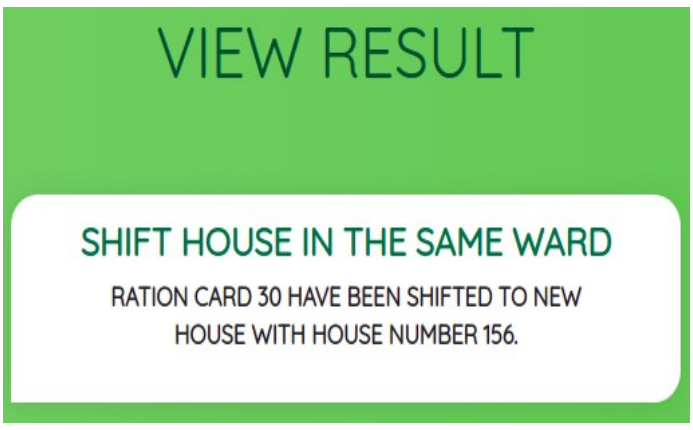


FIG 2 : Working of delete ration card function in the dapp

- Shift House In Same Ward
  - Input : Ration card number, new house number, new electricity connection status
  - Output : Just changes the house number and electricity connection status for the ration card.

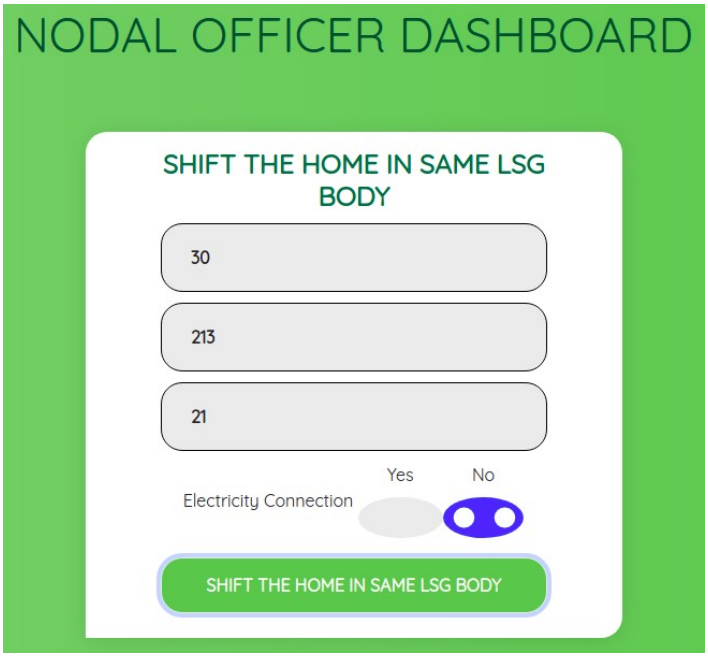


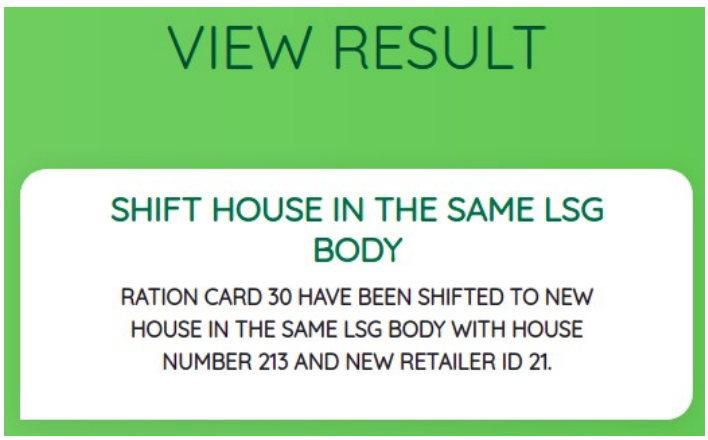


```
Disconnect from Fabric gateway.
POST /shiftFamilyInSameWard/submit 200 3790.370 ms - 952
Event: {"Type":"Shifting house in same ward","RationCardNumber":"30"}
GET /stylesheets/style.css 304 0.671 ms - -
GET /stylesheets/bootstrap.min.css 304 0.703 ms - -
GET /stylesheets/dashboard.css 304 0.741 ms - -
GET /javascripts/popper.js 304 1.465 ms - -
GET /javascripts/bootstrap.min.js 304 5.479 ms - -
GET /javascripts/index.js 304 4.053 ms - -
```

FIG 2 : Working of shift house in same ward function in the dapp

- Shift House In Same LSG Body
  - Input : Ration card number, new house number, new raion retailer ID, new electricity connection status
  - Output : Ration retailer ID, house number, electricity connection status of the ration card is changed. Consumer/Family details of both old and new ration retailers is updated.

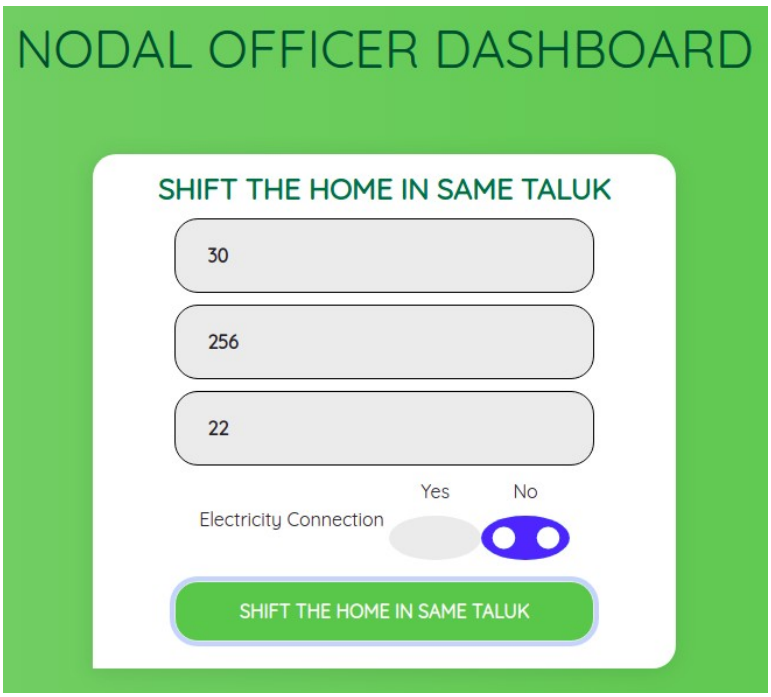




```
[Object: null prototype] {
  rationCardNumber: '30',
  newHouseNumber: '213',
  newRationRetailerId: '21',
  isNewHomeElectrified: 'no'
}
Disconnect from Fabric gateway.
POST /shiftFamilyInSameLSGBody/submit 200 3803.460 ms - 1000
Event: {"Type":"Shifting house in LSG body","RationCardNumber":"30"}
GET /stylesheets/style.css 304 1.769 ms - -
GET /stylesheets/bootstrap.min.css 304 6.771 ms - -
GET /stylesheets/dashboard.css 304 11.092 ms - -
GET /javascripts/popper.js 304 12.962 ms - -
GET /javascripts/bootstrap.min.js 304 9.016 ms - -
GET /javascripts/index.js 304 9.257 ms - -
```

FIG 2 : Working of shift home in same LSG body function in the dapp

- Shift House In Same Taluk
  - Input : Ration card number, new house number, new raion retailer ID, new electricity connection status
  - Output : Ration retailer ID, house number, electricity connection status of the ration card is changed. Consumer/Family details of both old and new ration retailers is updated.



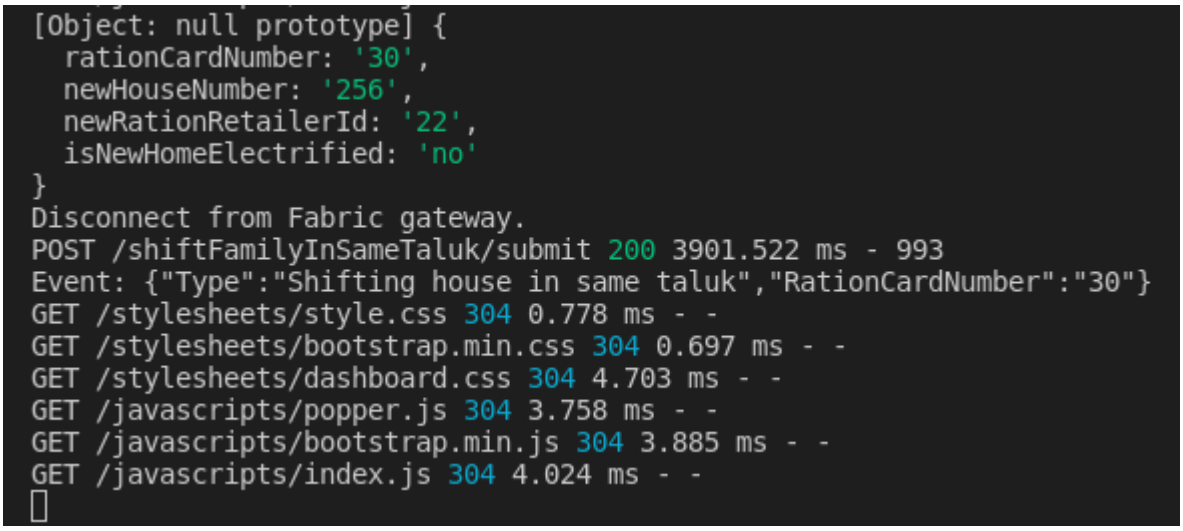
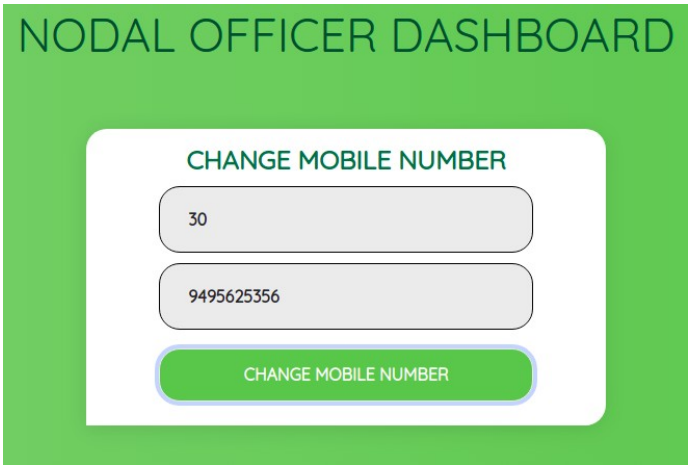
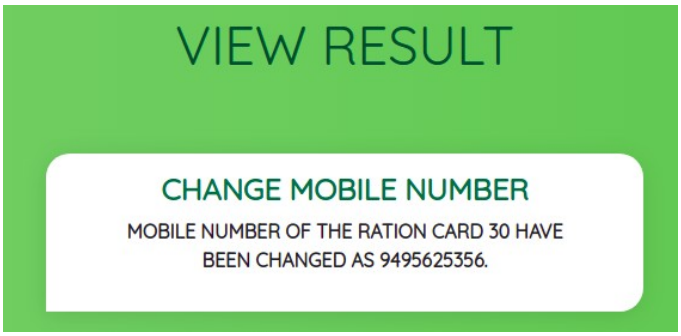


FIG 2 : Working of shift house in same taluk function in the dapp

- Change Mobile Number
  - Input : Ration card number, new mobile number
  - Output : Mobile number is changed for ration card data





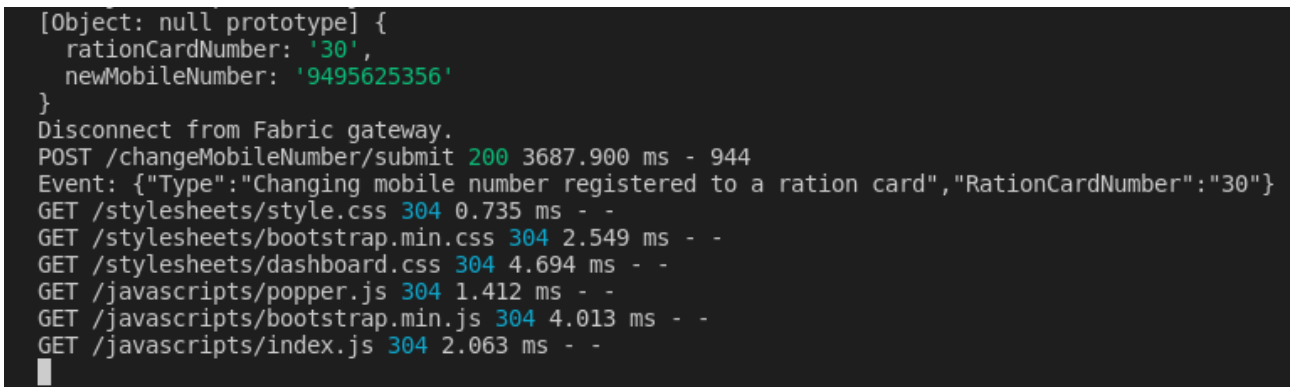
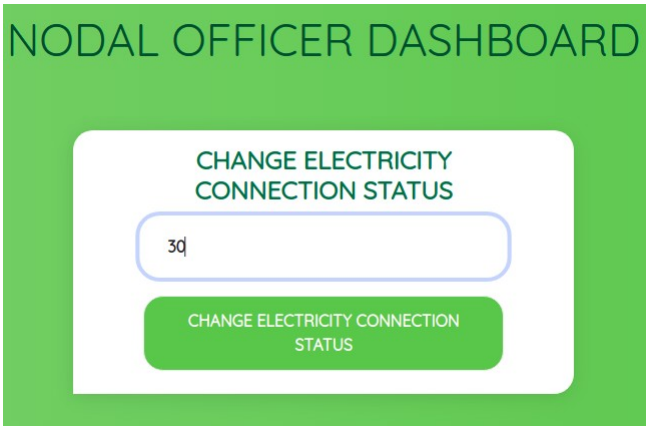
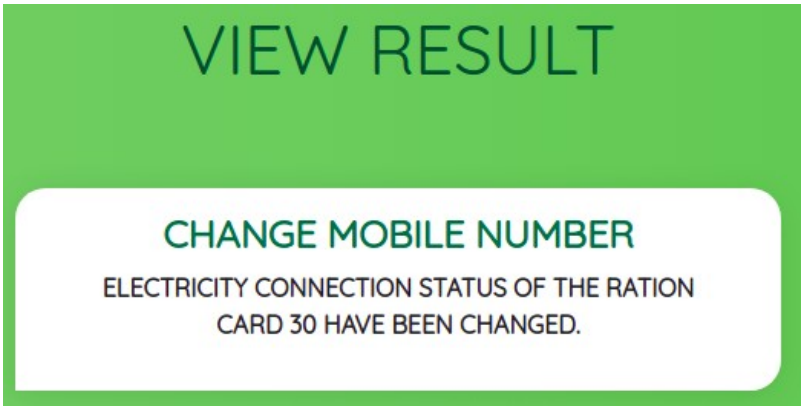


FIG 2 : Working of change mobile number function in the dapp

- Change Home Electrification Status
  - Input : Ration card number
  - Output : Electricity connection status of the ration card is reversed.





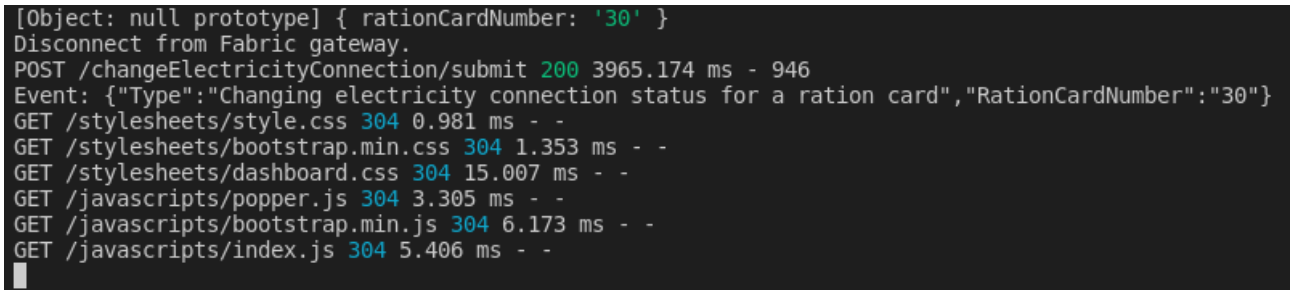
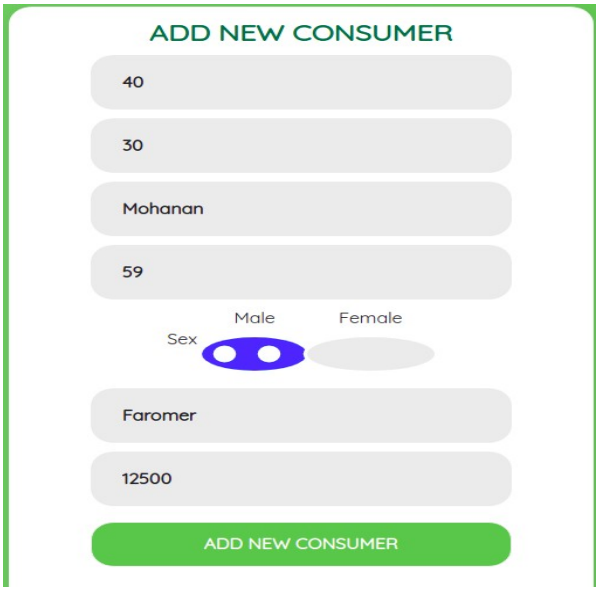
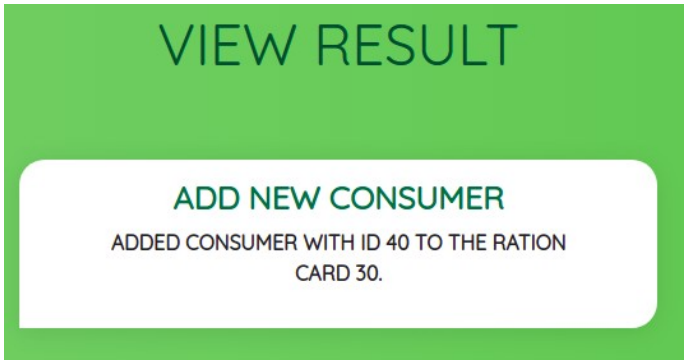


FIG 2 : Working of changing electricity connection status function in the dapp

- Add Consumer
  - Input : Consumer number, ration card number, name, age, sex, occupation, individual income
  - Output : Consumer account is created with the given parameters. If the new consumer is eligibile to become the family head, then the current family head is replaced with the new consumer. If adding the individual income to the total family income changes the type of the card, then the card type is updated with the new value. Retailer’s data about consumers/families is updated.





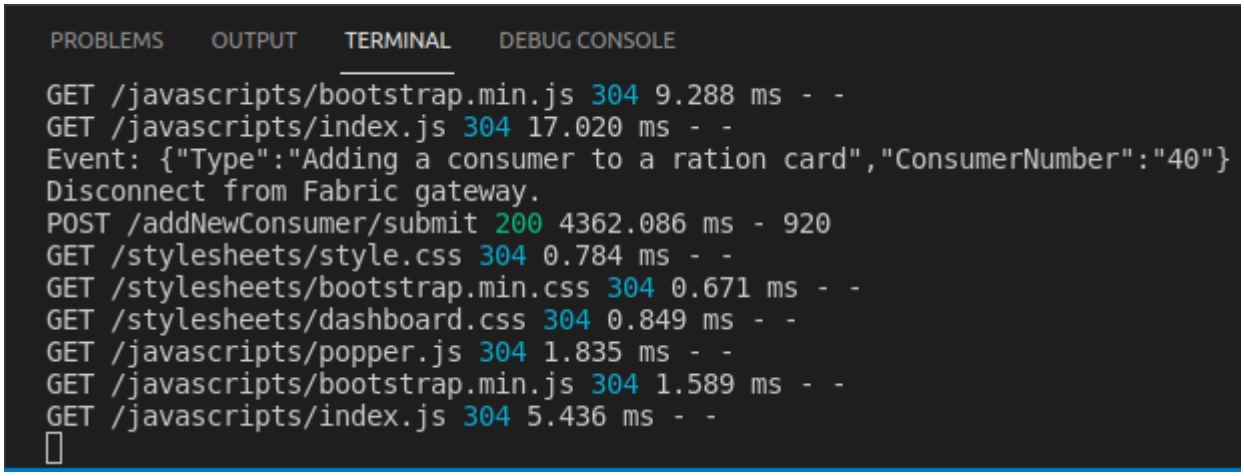
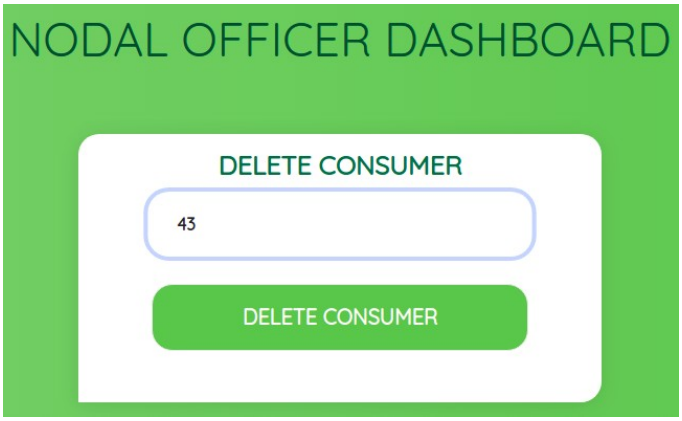
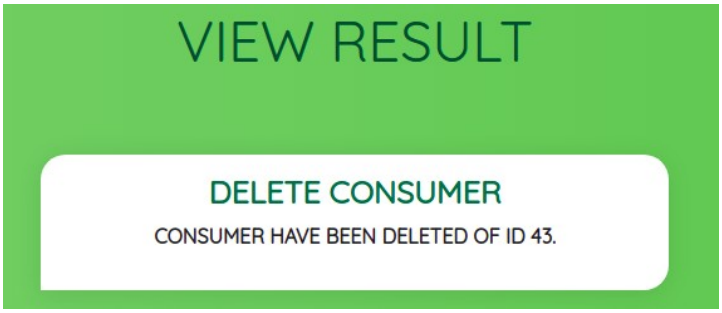


FIG 2 : Working of add consumer function in the dapp

- Delete Consumer
  - Input : Consumer number
  - Output : Deletes the consumer data. If the deleted user was the family head, then new family head is determined and data updated. If deleting consumer causes the ration card type to change, then the ration card type is updated. Ration retailers data about consumers/families is updated.





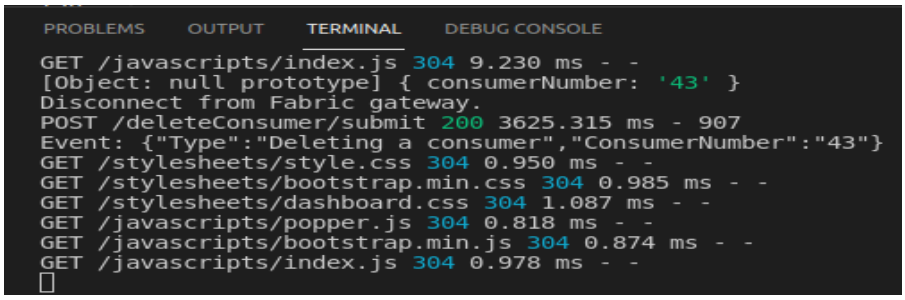
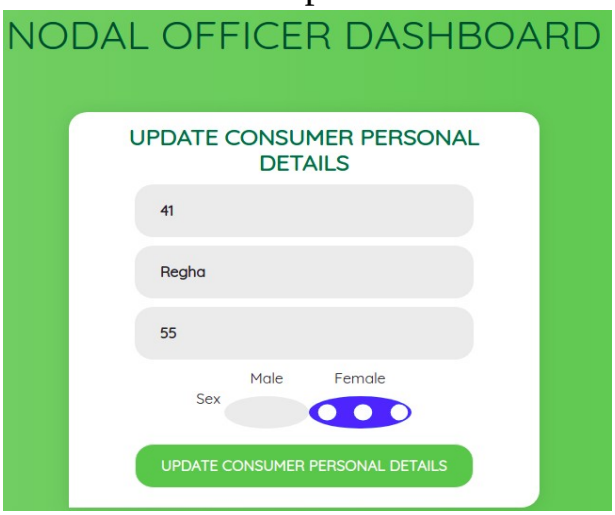


FIG 2 : Working of delete consumer function in the dapp

- Update Consumer Personal Details
  - Input : Consumer number, new name, new age, new sex
  - Output : The old data is replaced with the new one. If changing the data o new one causes to remove the eligibility to become the family head or causes to eligible to become the family head, then the family head is replaced.



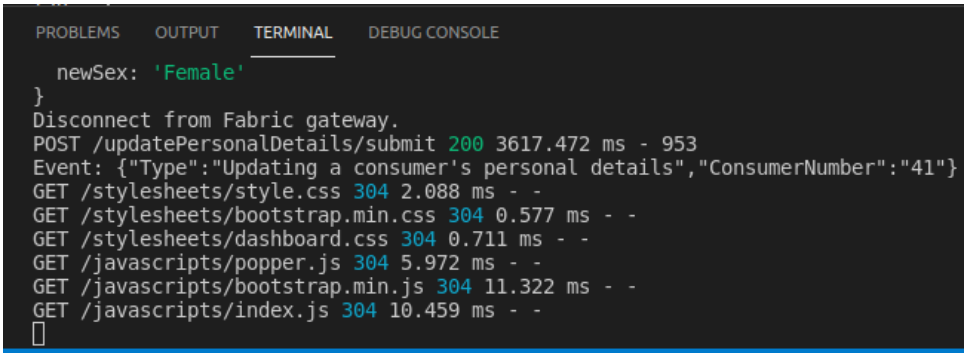
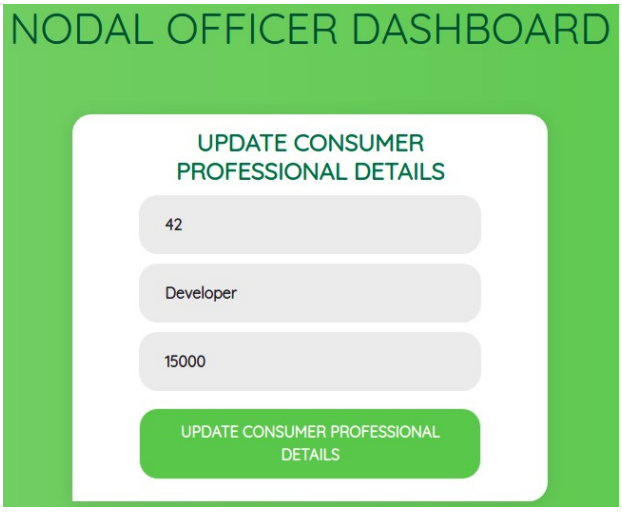


FIG 2 : Working of update personal details function in the dapp

- Update Consumer Professional Detailes
  - Input : Consumer number, new occupation, new individual income
  - Output : The old data is replaced with the new one. If changing the individual income makes the total family income changes such that the card type becomes different, then the card type detailes are updated, and retailer’s data will also become updated.



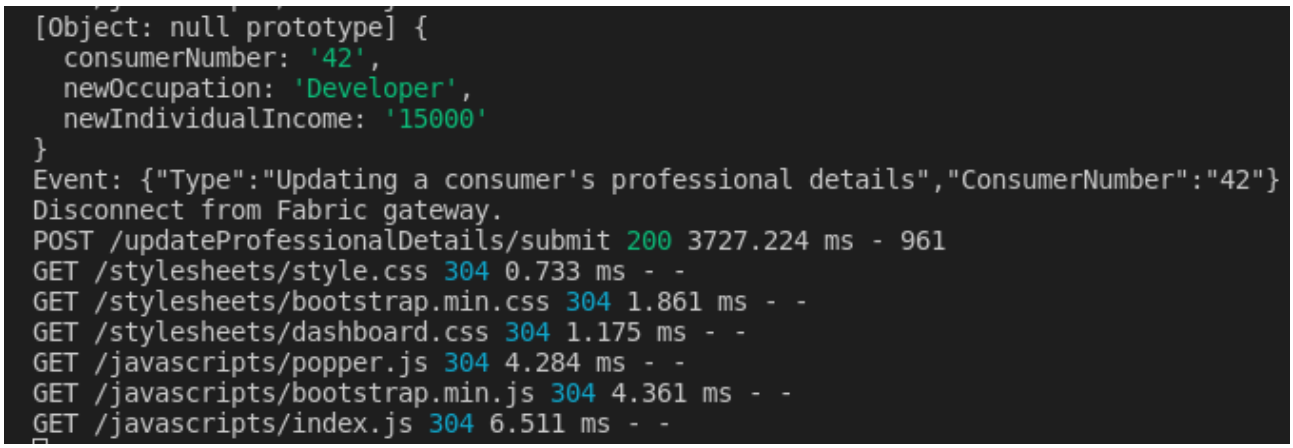
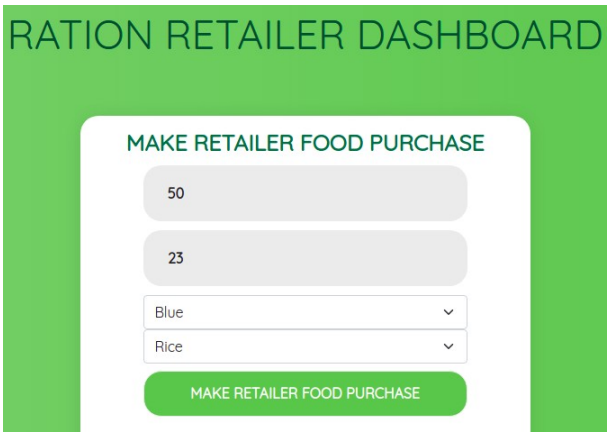


FIG 2 : Working of update professional details function in the dapp

Ration Retailer Functions

- Make Retailer Food Item Purchase
  - Input : Retailer purchase, number, ration retailer ID, ration card colour, and Item name
  - Output : Creates retailer purchase data of food items according to the four type of ration cards, numer of ration cards/number of consumers



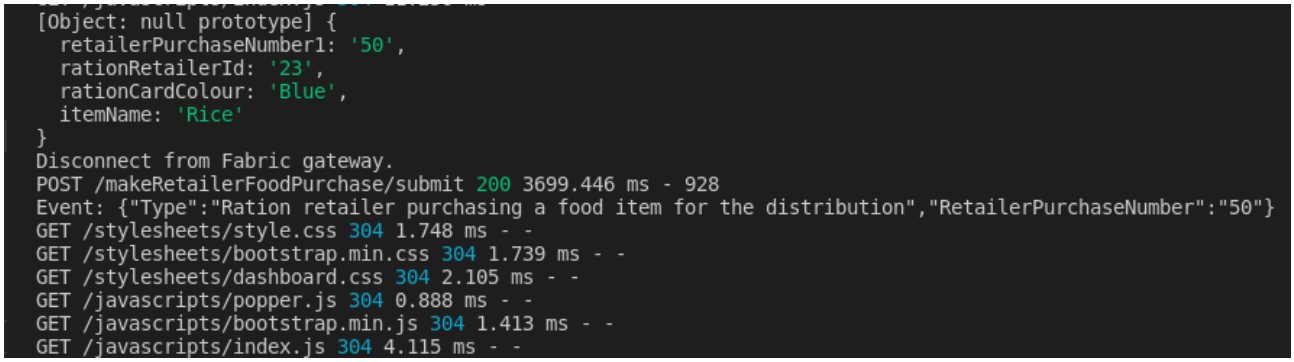


FIG 2 : Working of retailer food item purchase function in the dapp

- Make Retailer Kerosine Purchase
  - Input : Retailer purchase number, ration retailer ID
  - Output : Creates retailer purchase data of kerosine according to the number of ration cards with and without electricity connection

```
[Object: null prototype] {
  retailerPurchaseNumber: '53',
  rationRetailerId: '23'
}
Event: {"Type":"Ration retailer purchasing kerosine for the distribution","RetailerPurchaseNumber":"53"}
Disconnect from Fabric gateway.
POST /makeRetailerKerosinePurchase/submit 200 3335.967 ms - 930
GET /stylesheets/style.css 304 5.037 ms - -
GET /stylesheets/bootstrap.min.css 304 4.688 ms - -
GET /stylesheets/dashboard.css 304 3.784 ms - -
GET /javascripts/popper.js 304 3.648 ms - -
GET /javascripts/bootstrap.min.js 304 4.410 ms - -
GET /javascripts/index.js 304 4.486 ms - -
```

FIG 2 : Working of make retailer kerosine purchase function in the dapp

- Delete Retailer Putchase
  - Input : Retailer purchase number
  - Output : Deletes the data
- Make Consumer Purchase
  - Input : Retailer purchase number, retailer purchase number, ration card number
  - Output : Creates a consumer purchase data according to the given parameters.

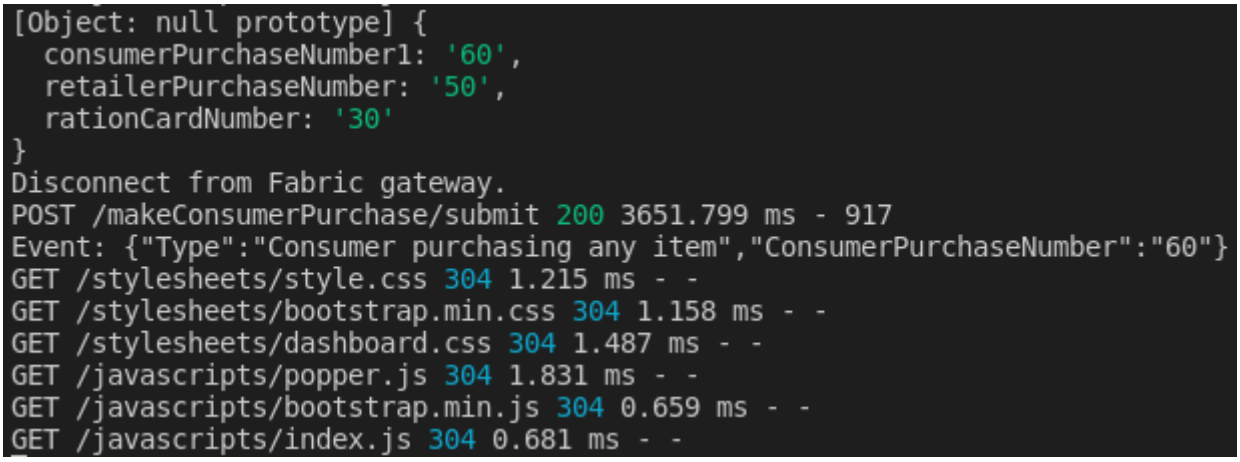


FIG 2 : Working of make consumer purchase function in the dapp

- Delete Consumer Purchase
  - Input : Retailer purchase number
  - Output : Deletes the data