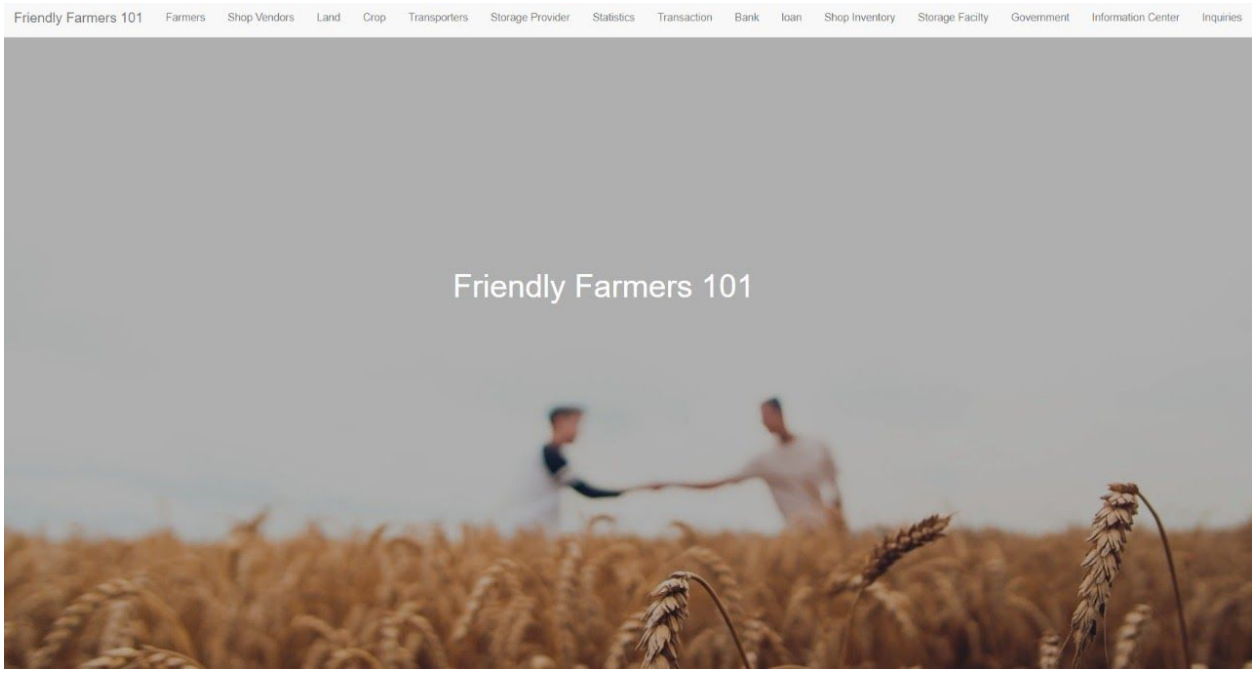
Group 34

FRIENDLY FARMERS 101
PROJECT REPORT



Team Members

Ansh Sharma, 2018130

Anunay Yadav, 2018021

Mukul Tomar, 2018296

Rishi Raj Jain, 2018304

Ruhma Mehek Khan, 2018362

GitHub Repository Link:

https://github.com/digitalPlayer1125/Friendly-Farmers-101

Week 1

We spent the first week deciding a problem statement. We wanted to come up with a project that can be useful to the people. We went about researching online platforms that do not currently exist. Our idea and motivation is listed below

The idea

To create a platform for the stakeholders involved at each stage of food production, i.e. from the production of agriculture produce to the sale of the final product in the market.

This platform will allow these stakeholders to interact with each other and will facilitate the purchase and sale of raw materials and produce respectively.

It will further allow the authorities, such as the Government to overlook the trade of agricultural produce, thereby increasing the transparency!

Our Motivation

Hoarding of the agricultural production leads to a hike in the price of the crop/end product. Hence, such a platform will help the authorities keep a track of the trade, and take down any such hoarding practices being performed at any stage.

Furthermore, our platform will help connect the stakeholders involved at each step and provide a satisfying experience to the stakeholder with an easy to use application.

A look at the process that is followed in the real scenario:

Stage	Examples
Stage 1: Assembly	Commodity buyers specializing in specific agricultural products, such commodities as grain, cattle, beef, oil palm, cotton, poultry and eggs, milk
Stage 2: Transportation	Independent truckers, trucking companies, railroads, airlines etc.
Stage 3: Storage	Grain elevators, public refrigerated warehouses, controlled-atmosphere warehouses, heated warehouses, freezer warehouses
Stage 4: Grading and classification	Commodity merchants or government grading officials
Stage 5: Distribution (Shop Vendors)	Independent wholesalers marketing products for various processing plants to retailers (chain retail stores sometimes have their own separate warehouse distribution centres)

(References: http://www.fao.org/3/w3240e/W3240E06.htm)

Week 2

Our Focus for this week was to list down our stakeholders and understand the functionalities our project would provide for each of them. Once we identified our stakeholders, we went ahead with the questions we will be answering, and then looked at this problem from the perspective of DBMS students, by further looking into the attributes and information we would need for the same.

Listed below are our stakeholders

Farmers:

Farmers: Farmers are our primary stakeholders. The application ensures transparency, thus satisfaction to the farmer. This will improve the current scenario. Every farmer can search for the storage providers, transporters, authorities, processing units, distributors, packaging units to meet his concerns.

Banks:

By banks, one is meant to infer different sources that act as money lenders. The role is major as cash is what is dealt through.

Storage facility providers:

These are the people who provide units to store the produce before it reaches the markets

Transportation facility providers:

The role these play is to transport the food from one place to the other, thereby bringing the product to the consumers

Shop vendors:

The distributors which mark the end stage of the above described process and sell the final product to the consumer

Authorities:

The project will help bring in transparency and enable the authorities to track the flow of agricultural produce.

The key questions we decided for each stakeholder are listed below:

Key questions for each stakeholder:

1. Farmer:
    a. What is the price of crop XYZ in the nearby area?
    b. What are the available loan rates?
    c. What are the nearest transport facilities?
    d. What all do the nearby storage facilities offer?

2. Banks:
    a. The number of loans that have been given out?
    b. What is the number of online transactions?
    c. What are the rates offered by local banks?
    d. What is the total amount of all the loans given?
    e. What is the total amount of all the pending loans?

3. Transporters:
    a. Check the status of authorization request?
    b. What are the prices being offered by other transporters for X units of weight?
    c. How many resources are left for a particular transporter?
    d. What is the distance between <destination> and <source>?
    e. How much weight can my resources transport?

4. Authorities:
    a. The number of non-authorized units operating(output the records)?
    b. The number of pending authorizations?
    c. The total cost of incomplete transactions,
    d. Rates being offered by farmers in “xyz locality” for “abc crop”,
    e. The number of authorized shop vendors in “xyz locality”.

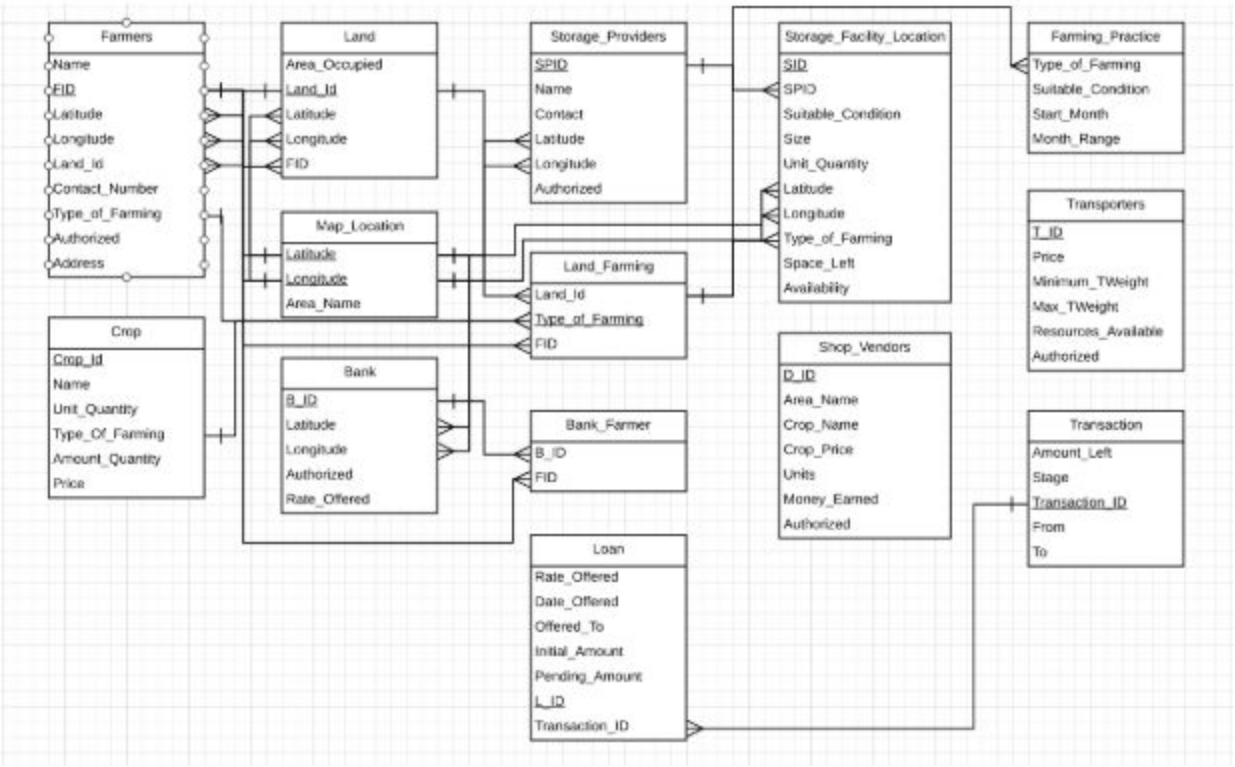
5. Shop Vendors:
    a. What price are other distributors in my locality offering for “xyz” crop?
    b. Check authorization status.
    c. What are the rates at which nearby banks are offering the loans?
    d. Display my inventory?

WEEK 3

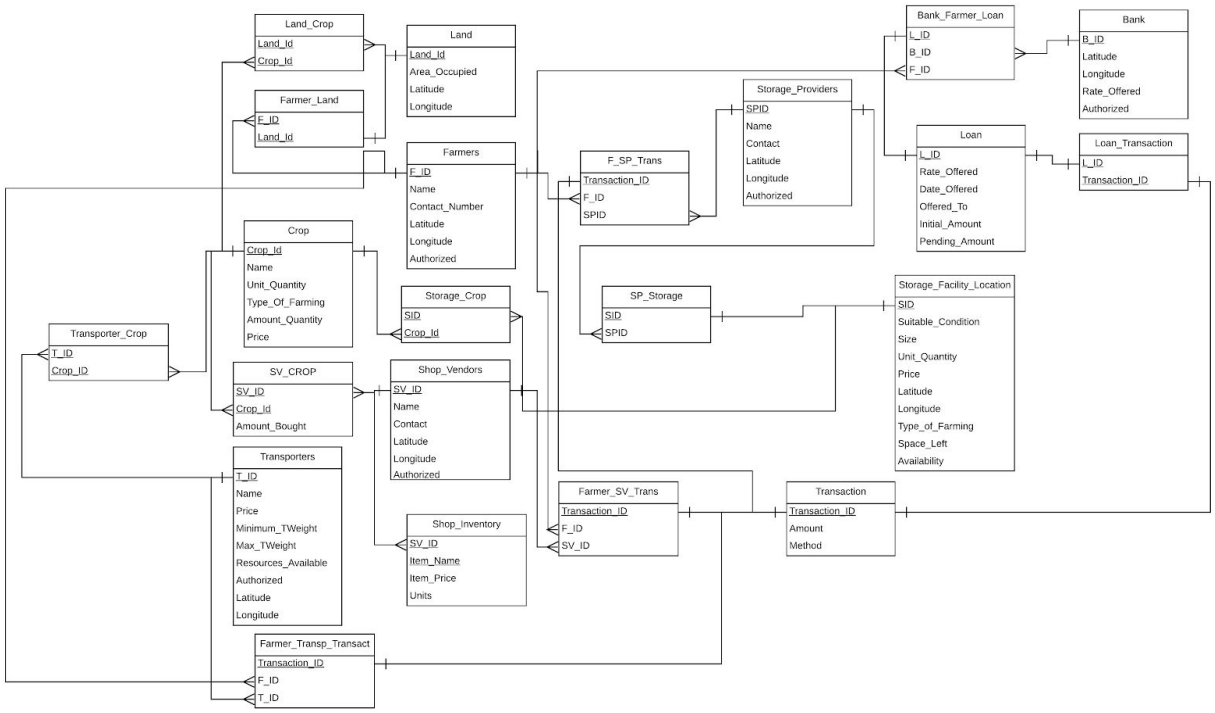
Once we had decided our stakeholders, we went on to working on our schema. Understanding what all attributes and information to store, the format for them, the relation between the various entities. We further defined the Foreign key-Primary key relationship, Referential Integrity constraints, Domain Integrity constraints and range values, etc.

Though we worked on our tables in the 3rd week, we later used the concepts learnt in the course, such as Normalization, to improve our schema. Currently we have tried to maintain a BCNF normal form. Below is the ER diagram of our final schema.

Initial



Final



Week 4

The task for this week was to populate our tables.

We wrote insert queries for each table so that all of them had sizable data.

Week 5

This week was reserved for mid-project evaluation.

Week 6

We started working on basic SQL queries. We used relational algebraic operations such as SELECT, NATURAL JOIN, INNER JOIN etc. We also worked on UPDATE and DELETE queries to allow updation and deletion of values from the database.

Index Creation

We have created the indexes of the following tables using their primary keys:

```
query_db("CREATE INDEX IF NOT EXISTS idx_farmer ON farmer (fid);")

    query_db("CREATE INDEX IF NOT EXISTS idx_land ON land (lid);")

    query_db("CREATE INDEX IF NOT EXISTS idx_crop ON crop (cid);")

      query_db("CREATE INDEX IF NOT EXISTS idx_shopvendor ON shopvendor
(svid);")

     query_db("CREATE INDEX IF NOT EXISTS idx_transporter ON transporter
(tid);")

     query_db("CREATE INDEX IF NOT EXISTS idx_storageprov ON storageprov
(spid);")

    query_db("CREATE INDEX IF NOT EXISTS idx_transactions ON transactions
(transid);")

    query_db("CREATE INDEX IF NOT EXISTS idx_bank ON bank (bid);")

    query_db("CREATE INDEX IF NOT EXISTS idx_loan ON loan (lid);")

         query_db("CREATE INDEX IF NOT EXISTS idx_storagefacloc ON
storagefacloc (sid);")

    query_db("CREATE INDEX IF NOT EXISTS idx_farmer ON farmer (fid);")
```

Relational queries :

a. Check authorization status of shopvendor.
   1.Input → ID
   2. Relational - $\prod_{authorized}(\sigma_{svid='ID'}(shopvendors))$
b. What are the prices being offered by other transporters for X units of weight?
   1.INPUT - > X units of weight
   2. Relational -> $\prod_{tid,Price*X}(\sigma_{mintwht <= x \text{ and } maxtwht >= x}(transporters))$
c. What is the number of online transactions?
   i. Input → None
   ii. Relational - $\sigma_{stage='online'}(count(transaction))$
d. What are the available loan rates?
   i. Input → None
   ii. Relational - $\prod_{rateoffr}(bank)$
e. Number of loans given by a bank?
   i. Input → bid as x
   ii. Relational → $\sigma_{bid='x'}(count(bfloan))$
f. Total amount of loans given out?
   i. Input → None
   ii. $\sigma_{sum(iniamt)}(loan)$
g. Prices offered for crop “xyz”?
   i. Input → crop name as xyz
   ii. $\prod_{price}(\sigma_{cname=''xyz''}(crop))$
h. What all do the nearby storage facilities offer?
   i. Input → latitude as A and longitude as B
   ii. $(\sigma_{lat>A-20 \text{ and } lat<A+20 \text{ and } long>B-20 \text{ and } long<B+20}(storagefacloc))$

i. Display my inventory?
   i. Input → svid as x
   ii. $\sigma_{svid = 'x'}(shop\_inv)$

j. Rates being offered by farmers in “xyz locality” for “abc crop”,
   i. Input → xyz locality, abc crop
   ii. T1 = $farmer \bowtie_{farmer.fid=farmerland.fid} farmerland$
   T2 = $T1 \bowtie_{farmerland.lid=land.lid} land$

$T3 = T2 \bowtie_{land.lid=landcrop.lid} landcrop$

$T4 = T3 \bowtie_{landcrop.cid=crop.cid} crop$

$\prod_{cid,\ cname,\ units,\ price}(\sigma_{cname=”abc” \text{ and } faddress=”xyz”}(T4))$

Below are some of our queries.

```
def insertintotrasaction():
    insert('transactions', ('transid', 'amount', 'method'), ('TR_101', 15000.00, 'Online'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_102', 120000.00, 'Cash'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_103', 100000.00, 'Online'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_104', 800000.00, 'Online'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_105', 200000.00, 'Cash'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_106', 100000.00, 'Online'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_107', 400000.00, 'Cash'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_108', 500000.00, 'Online'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_109', 600000.00, 'Cash'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_110', 200000.00, 'Cash'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_111', 400000.00, 'Online'))
    insert('transactions', ('transid', 'amount', 'method'), ('TR_112', 180000.00, 'Cash'))

def insertintoloan():
    insert('loan', ('lid', 'rateoffr', 'dateoffr', 'offrto', 'iniamt', 'pendamt'), ('L_1586', 10.35, '22-04-10', 'F_102', 13500.00, 11000.0
    insert('loan', ('lid', 'rateoffr', 'dateoffr', 'offrto', 'iniamt', 'pendamt'), ('L_2000', 8.00, '10-04-10', 'F_104', 50000.00, 49000.00
    insert('loan', ('lid', 'rateoffr', 'dateoffr', 'offrto', 'iniamt', 'pendamt'), ('L_2314', 9.35, '18-04-10', 'F_105', 20500.00, 20500.00

def insertintofarmer():
    insert('farmer', ('fid', 'fname', 'fcontact', 'authorized','lat','long'), ('F_102','Ramu',9997712345,1,28.613459,77.176208))
    insert('farmer', ('fid', 'fname', 'fcontact', 'authorized','lat','long'), ('F_104','Sahu',9412345678,0,28.603212,77.188439))
    insert('farmer', ('fid', 'fname', 'fcontact', 'authorized','lat','long'), ('F_105','Sid',7771122333,1,28.596580,77.181745))

def insertintoland():
    insert('land', ('lid', 'areaocc', 'lat', 'long'), ('LD_1321',44.12,28.605548,77.199597))
    insert('land', ('lid', 'areaocc', 'lat', 'long'), ('LD_5412',12.89,28.603212,77.203631))
    insert('land', ('lid', 'areaocc', 'lat', 'long'), ('LD_3498',23.01,28.598238,77.207236))

def insertintoshopv():
    insert('shopvendor', ('svid','svname','scontact','lat','long','authorized'), ('SV_191','Shop Benndor', 9898989898, 28.605133,77.202709,
    insert('shopvendor', ('svid','svname','scontact','lat','long','authorized'), ('SV_192','Shop Vendoe', 9898989898, 28.604116,77.204254,1
    insert('shopvendor', ('svid','svname','scontact','lat','long','authorized'), ('SV_193','Shop LOL', 9898989898, 28.598012,77.204812,0))

def insertintocrop():
    insert('crop', ('cid','cname','units','typeoffarming','quantity','price'), ('C_101','rice','kg','commercial farming',1.6,30))
    insert('crop', ('cid','cname','units','typeoffarming','quantity','price'), ('C_103','cotton','kg','Extensive farming',34.2,10))
    insert('crop', ('cid','cname','units','typeoffarming','quantity','price'), ('C_102','Chicken','kg','poultry farming',39.7,120))
```

```
    s = ("select price from crop where cname='{}' and cid in (select cid from landcrop where lid in (select lid from land as L where L.lat
    result = query_db(
        s
        )
    return result

def farmer_available_lrates():
    return query_db("select distinct rateoffr from bank")

def farmer_nearby_transport_fac(lat,long):
    lat_max = lat + 20
    lat_min = lat - 20
    lon_min = long - 20
    lon_max = long + 20
    s = ("select * from transporter where transporter.lat between {} and {} and transporter.long between {} and {}").format(lat_min,lat_max
    result = query_db(
        s
        )
    return result

def farmer_nearby_storage_fac(lat,long):
    lat_max = lat + 20
    lat_min = lat - 20
    lon_min = long - 20
    lon_max = long + 20
    s = ("select * from storagefacloc as D where D.lat between {} and {} and D.long between {} and {}").format(lat_min,lat_max,lon_min,lon_
    result = query_db(
        -
```

```
def update_authorized_farmer(val, farmer_id):
    s = ('update farmer set authorized = {} where fid = "{}"').format(val, farmer_id)
    query_db(s)


def update_contact_farmer(contact_no, farmer_id):
    s= ('update farmer set fcontact={} where fid ="{}"').format(contact_no, farmer_id)
    query_db(s)
def update_shopinv_amount(units,svid,item_name):
    quant = units
    s = ('select units from shop_inv where svid ="{}" and item_name="{}"').format(svid,item_name)
    res = query_db(s)
    quant += res[0][0]
    s= ('update shop_inv set item_units={} where svid ="{}" and item_name="{}"').format(quant,svid,item_name)
    query_db(s)

def  update_authorized_bank(val, bank_id):
    s = ('update bank set authorized = {} where bid = "{}"').format(val, bank_id)
    query_db(s)

def update_rateoffr_bank(rate, bank_id):
    s = ('update bank set rateoffr = {} where bid = "{}"').format(rate, bank_id)
    query_db(s)

def update_authorized_transporter(val, transporter_id):
    s = ('update transporter set authorized = {} where tid = "{}"').format(val, transporter_id)
    query_db(s)

def update_resavl_transporter(resval_val, transporter_id):
    s = ('update transporter set resavl ={} where tid = "{}"').format(resval_val, transporter_id)
    query_db(s)

def update_mintht_maxtht( transporter_id, mintht1=0, maxtht1=0 ):
    s = ('update transporter set mintwht={}, maxtwht = {} where tid = "{}"').format(mintht1, maxtht1, transporter_id)
    query_db(s)
def update_price_transporter(p1, transporter_id):
```

Week 7

We decided to integrate flask and SQL, and worked on writing embedded SQL queries to provide the functionalities. We also explored aggregate functions such as stdev so as to generate statistics for maintaining transparency & making the portal more informative. We have incorporated them into our queries to provide the required support to our application features.

Some of the queries we wrote our:

```
def storage_provider_auth():
    s = ("select count(*) from storageprov where authorized=0")
    result = getresult(s)
    l = [0,0]
    l[0] = result[0][0]
    s = ("select count(*) from storageprov  where authorized=1")
    result = getresult(s)
    l[1] = result[0][0]
    pieChart(["authorized","not_authorized"],l,"storage_provider_auth")
    return "storage_provider_auth_pie.png"
def shopvendor_auth():
    s = ("select count(*) from shopvendor where authorized=0")
    result = getresult(s)
    l = [0,0]
    l[0] = result[0][0]
    s = ("select count(*) from shopvendor  where authorized=1")
    result = getresult(s)
    l[1] = result[0][0]
    total = l[0] + l[1]
    pieChart(["authorized","unauthorized"],l,"shopvendor_auth")
    return "shopvendor_auth_pie.png"
```

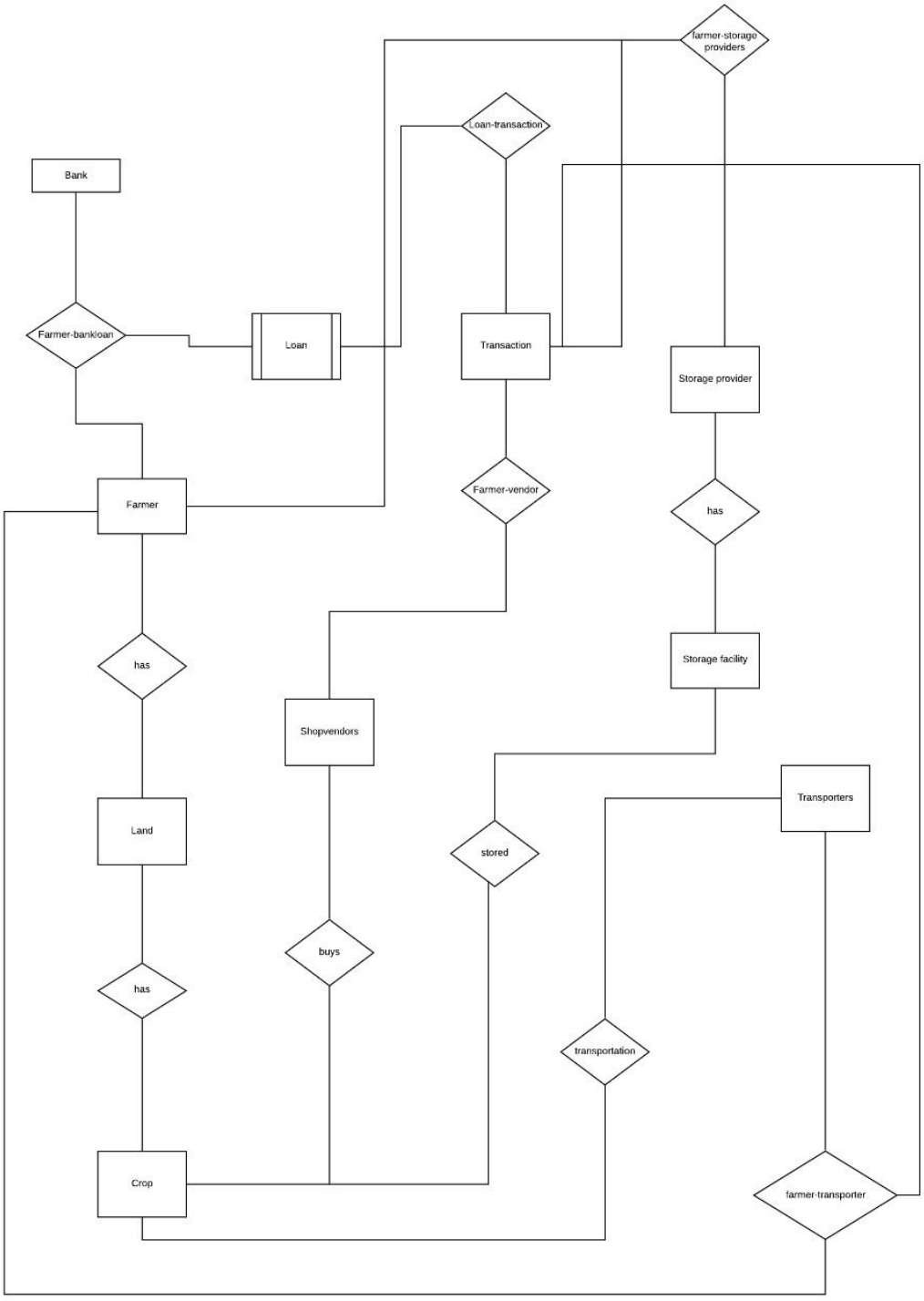
```
def bank_total_pending(BID):
    s = ("select SUM(pendamt) from bankloan,loan where bankloan.bid='{}' and loan.lid=bankloan.lid").format(BID)
    result = query_db(
        s
        )
    return result

def bank_total_lgiven():
    s = "select SUM(iniamt) from loan;"
    result = query_db(
        s
        )

def auth_total_inc_trans():
    s = "select SUM(transactions.amount) from transactions where transactions.method!='Complete'"
    res = query_db(s)
    return res;
```

Week 8

This week's target was to work on the ER diagram.



Week 9-12

Since the base of our project was ready. We spent the rest of our weeks working on our previous ideas.

We planned to have a user friendly UI, to make our project friendly and easy to use for the stakeholders.

We added a password to the authority page to restrict access.

We also added more queries, and worked on integrating the frontend and backend.

Queries

1. Farmer:
    a. What is the price of crop XYZ in the nearby area?
        i. Input → Crop name as XYZ
        ii. SELECT price from crop where cname=”xyz” and cid in (SELECT cid from landcrop where lid in (Select lid from land as l where l.lat between A-20 and A+20 and l.long between B-20 and B+20))
    b. What are the available loan rates?
        i. Input → None
        ii. Query → SELECT DISTINCT rateoffr from bank
    c. What are the nearest transport facilities?
        i. Input → Lat as A, Long as B (Of the farmer while executing the query)
        ii. Query → Select transporters.lat, transporters.long from transporters where transporters.lat between A-20 and A+20 and transporters.long between B-20 and B+20;

    d. What all do the nearby storage facilities offer?

i. Input → Lat as A, Long as B (Of the farmer while executing the query)
ii. Query → Select * from storagefacloc as D where D.lat between A-20 and A+20 and D.long between B-20 and B+20;

Modify → authorized, fcontact, faddress

Authorized:

Input: boolean val, farmer_id
Update farmer Set authorized=val Where bid=bank_id;

Contact:

Input: contact_farmer int(10), farmer_id
Update farmer Set fcontact=contact_farmer Where fid=farmer_id;

Address:

Input: contact_address, farmer_id
Update farmer Set faddress=contact_address Where fid=farmer_id;

2. Banks:
   a. The number of loans that have been given out?
      Input → BID of bank executing query as X
      SELECT count(*) from bfloan as l where l.bid=X

   b. What is the number of online transactions?
      i. Input → None
      ii. Query → SELECT count(*) from transaction as t1 where t1.stage='online'
   c. What are the rates offered by local banks?
      i. Input → Lat as A, Long as B (Of the bank while executing the query)
      ii. Query → Select bank.rateoffr from bank where bank.lat between A-20 and A+20 and bank.long between B-20 and B+20;
   d. What is the total amount of all the loans given?
      i. Input → None
      ii. Query → SELECT SUM(iniamt) FROM loan;
   e. What is the total amount of all the pending loans?
      i. Input → Bank ID

ii. Query → Select SUM(pendamt) from bankloan,loan where bankloan.bid=BANK_ID and loan.lid=bankloan.lid

Modify → authorized, rateoffr
Authorized :
Input : boolean val,bank_id
Update Bank Set authorized=val Where bid=bank_id;
rateoffr :
Input : rate,bank_id
Update Bank Set rateoffr = rate Where bid=bank_id;

3. Transporters:
   a. Check the status of authorization request?
      Input → T_ID of transporter executing the request
      Query → SELECT authorized from transporters as T where T.tid = T_ID
   b. What are the prices being offered by other transporters for X units of weight?
      i. Input → X units of weight
      ii. Query →SELECT tid,Price*X from (SELECT tid,Price from transporters where mintwht <= X and maxtwht >= X)
   c. How many resources are left for a particular transporter?
      i. Input → Tname
      ii. Query → Select SUM(transporter.resavl) from transporter where transporter.tname=Tname;
   d. What is the distance between <destination> and <source>?
      i. Input → source, destination, id
      ii. Query → SELECT lat,long from <table_name> where <id>=<id>;
   e. How much weight can my resources transport?
      Input → Transporter_ID
      i. Query → Select maxtwht from transporters as A where A.ID= Transporter_ID;

Modify →
Authorized :

Input : boolean val,transporter_id
Update Transporters Set authorized=val Where transid=transporter_id;

Resavl :

Input : resval val,transporter_id
Update Transporters Set reval=val Where transid=transporter_id;

mintht,maxtht :

Input :mintht1,maxtht1,transporter_id
Update Transporters Set mintht=mintht1,maxtht=maxtht1 Where transid=transporter_id;

Price:

Input : price1,transporter_id
Update Transporters Set price=price1 Where transid=transporter_id;

4. Authorities:
    a. The number of non-authorized units operating(output the records)?
    SELECT count(*) from bank as b where b.authorized = false
    SELECT count(*) from farmer as b where b.authorized = false
    SELECT count(*) from shopvendors as b where b.authorized = false
    SELECT count(*) from storageprov as b where b.authorized = false

    b. The number of pending authorizations?
        i. Input →None
        ii. Query → SELECT count(authorized) from (SELECT authorized from shopvendors UNION ALL SELECT authorized from storageprov UNION ALL SELECT authorized from transporters) where authorized=false
    c. The total cost of incomplete transactions,
        i. Input → None
        ii. Query → Select SUM(transaction.amount) from transaction where transaction.stage!='Complete';
    d. Rates being offered by farmers in "xyz locality" for "abc crop",
        i. Input → xyz locality, abc crop
        ii. Query → Select unique c.cid, c.cname, c.units, c.price from crop c inner join landcrop lc on c.cid=lc.cid inner join land l on lc.lid=l.lid inner join farmerland fl on l.lid=fl.lid inner join farmer f on fl.fid=f.fid where locate(xyz, f.faddress)>0 and c.cname=abc;

e. The number of authorized shop vendors in “xyz locality”.
    i. Input → xyz locality
    ii. Query → Select count(*) from shopvendors as A where locate(xyz, A.svaddress) >0;

5. Shop Vendors:
    a. What price are other distributors in my locality offering for “xyz” crop?
    Input → Crop name “XYZ”,lat,long
    Query → Select item_price from shop_inv where item_name= “xyz” and svid in (Select svid from shopvendors as A where A.lat between l-20 and l+20 and A.long between long-20 and long+20);

    b. Check authorization status.
        i. Input → ID
        ii. Query → Select authorized from shopvendors where svid=ID
    c. What are the rates at which nearby banks are offering the loans?
        i. Input → Lat as A, Long as B (Of the distributor while executing the query)
        ii. Query → Select bank.rateoffr from bank where bank.lat between A-20 and A+20 and bank.long between B-20 and B+20;
    d. Display my inventory?
        i. Input → shopvendor id
        ii. Query → Select * from shop_inv where shop_inv.svid=svid;

Modify →

Authorized :

Input: boolean val, sv_id
Update shopvendor Set authorized=val Where svid=sv_id;

Lat, Long:

Input: lat_a, long_a, sv_id
Update shopvendor Set lat=lat_a, long=long_a Where svid=sv_id;

Price:

Input: decimal price_value, crop_id

Update crop Set price=price_value Where cid=crop_id;

Item Price:

Input: decimal item_price_value, shop_inv_id
Update shop_inv Set item_price=item_price_value Where svid=shop_inv_id;

Item Price:

Input: decimal units_value, shop_inv_id
Update shop_inv Set units=units_value Where svid=shop_inv_id;

Course Learnings and Concepts Used

As we progressed with the course material we kept making changes to our project to utilise the concepts learnt. Some of them are:

Using the concept of Relational Schema and ER diagram

Normalization to make improve our database design

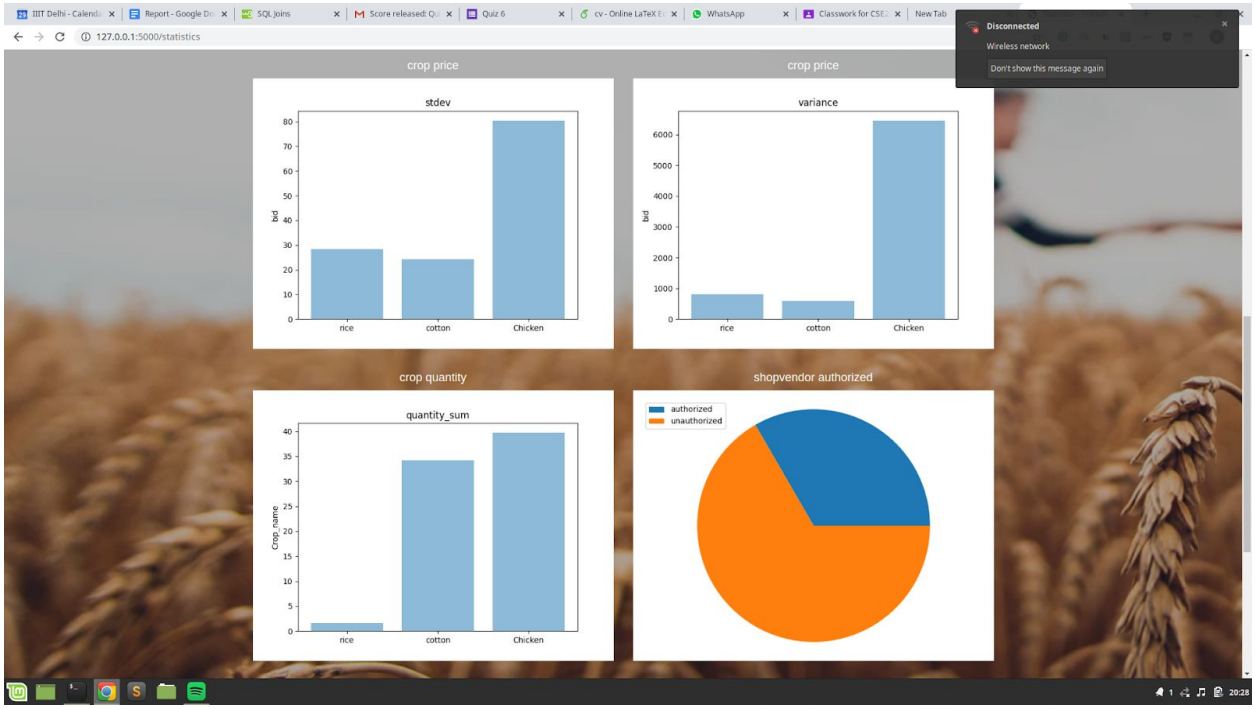
Use of advanced SQL queries such as aggregation, nested queries, etc.

Embedded SQL queries

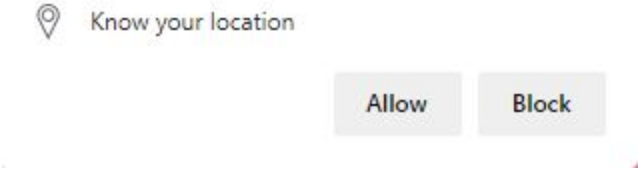
Integration of Flask (python) and SQL

Additional Features/Bonus:

- Apart from the features previously decided, we realised that a visualisation of data would enable our stakeholders to get a clear picture and hence also added bar graphs and pie charts to enable the stakeholders to view the statistics.



Instead of asking users for their address, as we previously planned, we decided to collect their locations from the browser. We used python's geopy module for conversion of the collected coordinates to addresses.



This enabled us to provide more accurate answers to queries relating to distances between two units (such as storage facility and shop vendors).

Further Developments:

We were able to develop a project that tackles and provides a solution to our initial problem statement.

However, as developers, we also kept in mind to keep our project flexible in order to incorporate additional features. Our project can hence be extended in the following ways:

- Some agricultural produce is processed and packaged before going into markets. This brings in two more stakeholders, i.e. the processing units and the packaging units. Our project can be developed further to include them as well.
- Role of banks:

  For the purpose of this project, we defined the role of banks as loan providing units. However, our project can be expanded, and probably interlinked with other platforms which manage other roles of banks as well.

Individual Contributions:

Anunay: Ideation, E-R Model, Queries, Front End, Back End

Ansh: Ideation, E-R Model, Queries, Back End, Statistics (Graphs)

Mukul: Queries, Ideation, Report, Populating the tables, Defining the schema (schema.sql)(Back End)

Rishi Raj Jain: Ideation, Queries, Integration, Front End, Back End, Design

Ruhma: Ideation, Queries, Report, Back End