DBMS PROJECT Group No 41
SELF EVALUATION TABLE

DBMS PROJECT Group No 41

Single-Source Capital Management System

Week 1

Our team members are :-

1. Samarth Chauhan (front and back end of site)
2. Ritvik Gupta (mysql and flask)
3. Ritwik Gupta (flask and integrity with other platform)
4. Yash Kanojia (mysql and flask)
5. Kartikay Sapra (flask and integrity with other platform)

“Monergize”

Project Description:

The capital management system which will deal with management and transfer of digital currency and has the following stakeholders. This unique database management application can solve the problems of several categories of organisations at once. This system is a unified capital management system, one source for all kinds of financial services.

Our Web-Application provides a centralized control to the government on all the Customers, Banks, Businesses and other financial institutions in the economy.

Purpose:

Finance related activities have always been time consuming and involved multiple steps which had to be performed sequentially. Monergize introduces a single source platform which will fulfil all your finance related transactions from buying items from a local retailer to paying salaries to employees. The intention is to provide this system to the Government which will monitor all the transactions in the economy.

Week 2

The StakeHolders

The following are the stakeholders identified by us along with their corresponding use cases :

Bank Customers:-

Key Questions Addressed: (achieve all these in a time and cost efficient manner)

1. How to Open Savings/Current Account with the Bank?
2. On which platform to perform Online/Offline Money transactions?
3. Where to apply for loan?
4. How to ensure the privacy of personal financial data?
5. How to manage Fixed Deposits set up with the bank?

Government:-

Key Questions Addressed:

1. Show the daily Stock Market Analysis to the users
2. How to ensure that the relevant data is shared with the users?
3. How to manage revenue coming from various sources?
4. How to keep track of all Banks under the government?
5. How to keep the salary record of the Officers/Employees?

MNCs:-

Key Questions Addressed:

1. How to manage the assets of the company?
2. How to manage the data related to the employees?
3. How to ensure the privacy of the company's data?
4. How to keep the salary record of the Employees?
5. Recommendation on which bank should the company take loan from?
6. Show analysis of company's revenue and profits.

Business Owners/Startups :-

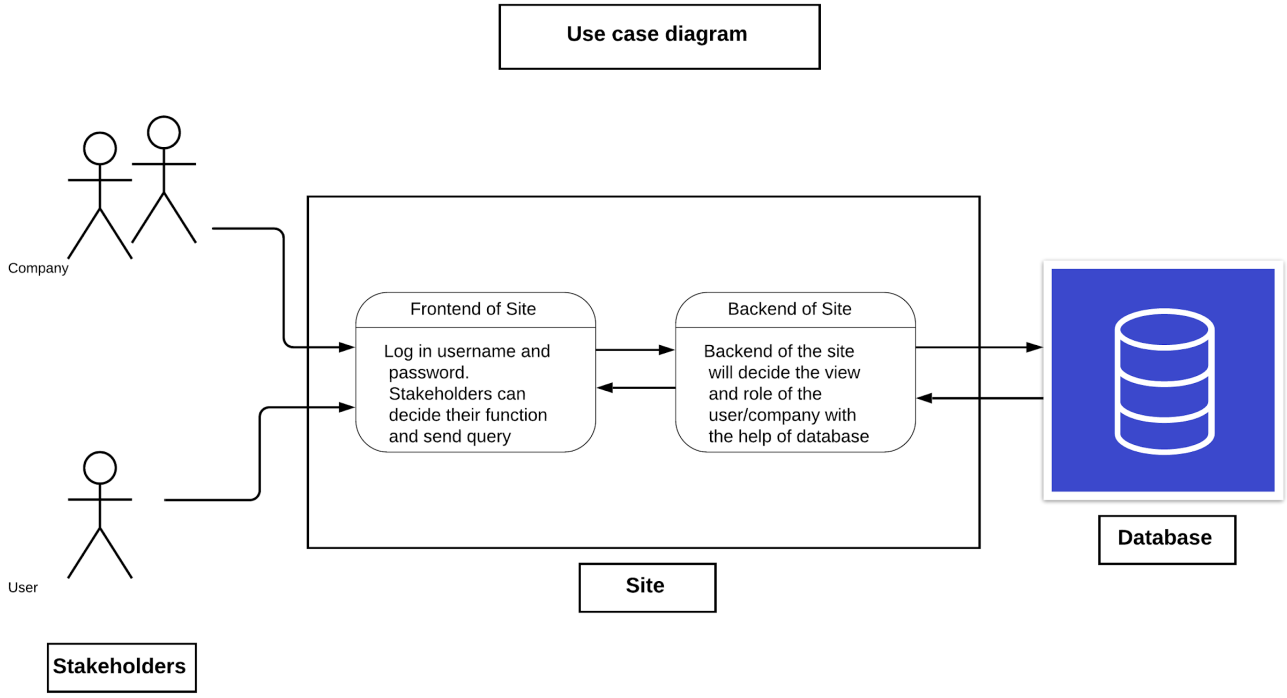
1. How to keep track of all the products which are in the inventory?
2. How to manage the records of the transactions related to the business?
3. How to keep a record of salary payments to the Employees?
4. How to apply for a loan to the bank?
5. How to track progress of your business compared to others in the market?

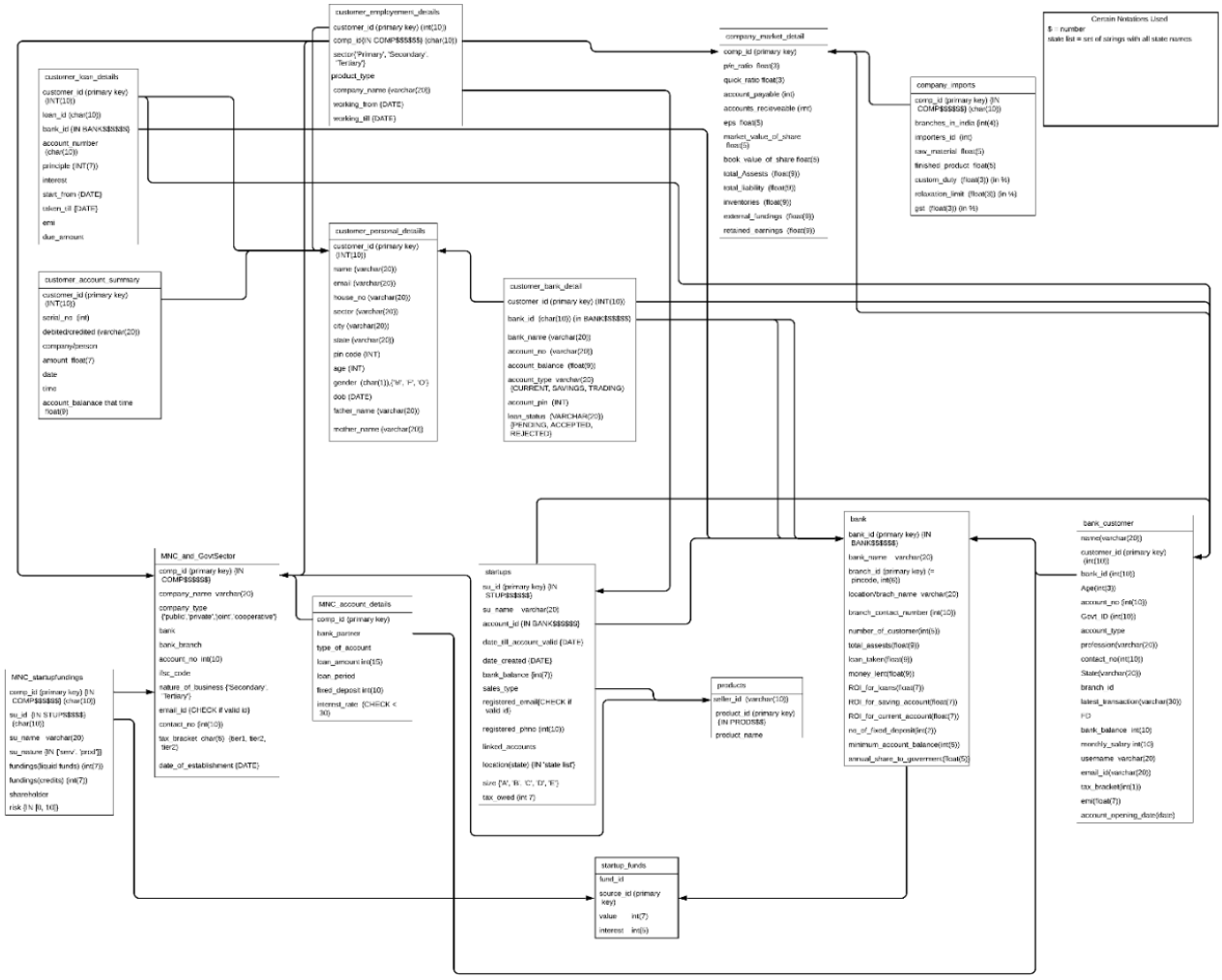
Banks:-

1. Where to find the loan applications pending for approval?
2. How to manage all the Savings accounts?
3. How to manage all the Business accounts?
4. Management of the various branches of the bank?
5. How to manage Revenue Database for the Bank?

Week 3

Schema of the DBMS





Above contains table that will be used in the database and the relation between them

If there is any problem with resolution or blur table view you go to

[Database Schema - Entity Relationship Diagram Example (UML Notation).png]

Week 4

Created the tables in mysql as per the schema. Population of the data was done using an online data generator and various web sources.

Contributions till week 4:

All the team members sat together and decided the domain and made the schema for the Database.

Contributions:

(1) Samarth Chauhan : Database creation and online management, table population. Schema creation.

(2) Kartikay Sapra : Transaction Management, Market Details and Logic behind transaction and making the right financial decisions.

(3) Ritvik Gupta : Table population, schema creation, w-in-p write up.

(4) Ritwik Gupta : Merging between tables, logic behind connection of attributes of tables.

(5) Yash Kanojia : Table population, w-in-p writeup, ER diagram.

Week 5

- Mid project Evaluation
- Completion of population of data in all the tables
- Referential Integrity
- Connections among table clearly satisfied

Week 6 and 7

Refer PDF attached in the shared folder for relational algebraic queries.

https://drive.google.com/file/d/1jkXV2qnVB5YvVlmXEabZW0FgfnyGUAjp/view

PL/SQL Queries:

We have used embedded sql queries everywhere for using our mysql database with Flask.

Format of a general query:

cur = mysql.connection.cursor()

cur.execute(query)

details=cur.fetchone()

mysql.connection.commit()

Following are a few of the queries that we have written.

1) Accept Loan Application:

```
cur.execute("SELECT l.principal,l.max_period,b.interest,b.bank_id FROM
bank.loan_application_data l LEFT JOIN bank.bank_loan_detail b ON l.Loan_id =
b.loan_id WHERE l.application_id=%s;",[int(application_id)])
```

2) Request Customer List:

```
cur.execute("SELECT
s.customer_id,s.name,s.email,s.house_no,s.sector,s.city,s.city,s.state,s.pin_code,s.ag
e,s.gender,s.dob,m.bank_id,m.account_no,m.account_balance,m.account_type,m.account_pin
,d.password FROM customer_personal_detail s LEFT JOIN customer_bank_details m ON
s.customer_id = m.customer_id LEFT JOIN customer_login_detail d ON d.customer_id =
```

```
m.customer_id WHERE m.bank_id=%s AND s.customer_id LIKE
%s;",[str(bankid),'%'+str(term)+"%"])
```

3) Request Customer Details:

```
cur.execute("SELECT
s.customer_id,s.name,s.email,s.house_no,s.sector,s.city,s.city,s.state,s.pin_code,s.ag
e,s.gender,s.dob,m.bank_id,m.account_no,m.account_balance,m.account_type,m.account_pin
,d.password,e.Company_ID,e.Company_Name,e.Sector,e.Working_From,e.Working_Till FROM
customer_personal_detail s LEFT JOIN customer_bank_details m ON s.customer_id =
m.customer_id LEFT JOIN customer_login_detail d ON d.customer_id = m.customer_id LEFT
JOIN customer_employment_details e ON e.customer_id = s.customer_id WHERE
s.customer_id LIKE %s;",['%'+str(term)+"%"])
```

4) Bank Recommendations:

```
cur.execute("SELECT bank_id, bank_name, branch_id, branch_name, branch_contact_number,
number_of_customer, ROI_for_loans, ROI_for_savings, ROI_for_current, min_acc_balance
from (select bank_id, bank_name, branch_id, branch_name, branch_contact_number,
number_of_customer, ROI_for_loans, ROI_for_savings, ROI_for_current, min_acc_balance
from bank.bank where annual_share_govt>20000) as t1 where number_of_customer > 200")
```

5) Company Comparison:

```
cur.execute("(select * from bank.company_market_detail where pe_ratio = (select
max(pe_ratio) from company_market_detail)) union (select * from
bank.company_market_detail where eps = (select max(eps) from company_market_detail))")
```

6) Lowest Priced Products:

```
cur.execute("select f.seller_id, f.product_id,f.product_name,f.category,f.price from (
select category, min(price) as minprice from products group by category) as x inner
join products as f on f.category = x.category and f.price = x.minprice")
```

7) Best Seller Products:

```
cur.execute("select f.seller_id, f.product_id,f.product_name,f.category,f.price from (
select category, max(sold) as maxsold from products group by category) as x inner join
products as f on f.category = x.category and f.sold = x.maxsold")
```

8) Current Loan Application:

```
cur.execute("SELECT * FROM bank.loan_application_data l LEFT JOIN bank_loan_detail b
ON l.Loan_id=b.loan_id WHERE l.Customer_id=%s;",[Customer_ID])
```

9) Request Customer List By Name:

```
 cur.execute("SELECT
s.customer_id,s.name,s.email,s.house_no,s.sector,s.city,s.city,s.state,s.pin_code,s.ag
e,s.gender,s.dob,m.bank_id,m.account_no,m.account_balance,m.account_type,m.account_pin
,d.password FROM customer_personal_detail s LEFT JOIN customer_bank_details m ON
s.customer_id = m.customer_id LEFT JOIN customer_login_detail d ON d.customer_id =
m.customer_id WHERE m.bank_id=%s AND s.name LIKE %s;",[str(bankid),'%'+str(term)+'%'])
```

10) Recommend Banks based on Savings:

```
cur.execute("SELECT bank_id, bank_name, branch_id, branch_name, branch_contact_number,
number_of_customer, ROI_for_loans, ROI_for_savings, ROI_for_current, min_acc_balance
from bank.bank where ROI_for_savings >= (SELECT AVG(roi_for_savings) from bank.bank)
and no_of_fds >=(select avg(no_of_fds) from bank.bank)")
```

11) Recommend Most Friendly Banks

```
      cur.execute("SELECT bank_id, bank_name, branch_id, branch_name,
branch_contact_number, number_of_customer, ROI_for_loans, ROI_for_savings,
ROI_for_current, min_acc_balance from bank.bank where ROI_for_loans <= (SELECT
AVG(roi_for_loans) from bank.bank)")
```

12) Bank User details:

```
   cur.execute("SELECT bank_id,bank_name,account_no,account_balance,account_pin FROM
bank.customer_bank_details WHERE Customer_id = %s;",[int(id)])
```

13) Make Transactions:

```
   cur.execute("UPDATE bank.customer_bank_details SET account_balance =
(account_balance - %s) WHERE account_no = %s;",[int(amount),str(acc_from)])
```

14) Enquire Loan:

```
   cur.execute("SELECT * FROM bank.bank_loan_detail WHERE loan_type=%s AND
max_period>%s AND bank_id=%s;",[str(type),int(period),str(bank_id)])
```

15) Recommend Most Trusted Banks:

```
       cur.execute("SELECT bank_id, bank_name, branch_id, branch_name,
branch_contact_number, number_of_customer, ROI_for_loans, ROI_for_savings,
ROI_for_current, min_acc_balance from (select bank_id, bank_name, branch_id,
branch_name, branch_contact_number, number_of_customer, ROI_for_loans,
ROI_for_savings, ROI_for_current, min_acc_balance from bank.bank where
annual_share_govt>20000) as t1 where number_of_customer > 200")
```

16) Update Loan Summary:

```
   cur.execute("INSERT INTO bank.bank_loan_detail VALUES(%s, %s, %s, %s, %s)",
[id_val, str(bank_id), str(loan_type), int(interest), int(max_period)])
```

17) Calculate Custom Duty:

```
   cur.execute("SELECT (raw_material + finished_product)*(custom_duty + gst -
relaxation_limit)/100 FROM bank.company_imports WHERE COMP_ID=(%s)", [comp_id])
```

18)

```
   cur.execute("INSERT INTO bank.customer_personal_detail VALUES(%s, %s, %s, %s, %s,
%s, %s, %s, %s, %s, %s, %s, %s);", [id_val, user_detail[0], user_detail[1],
user_detail[2], user_detail[3], user_detail[4], user_detail[5], user_detail[6],
user_detail[7], user_detail[8], user_detail[9], user_detail[10], user_detail[11]])
```

19)

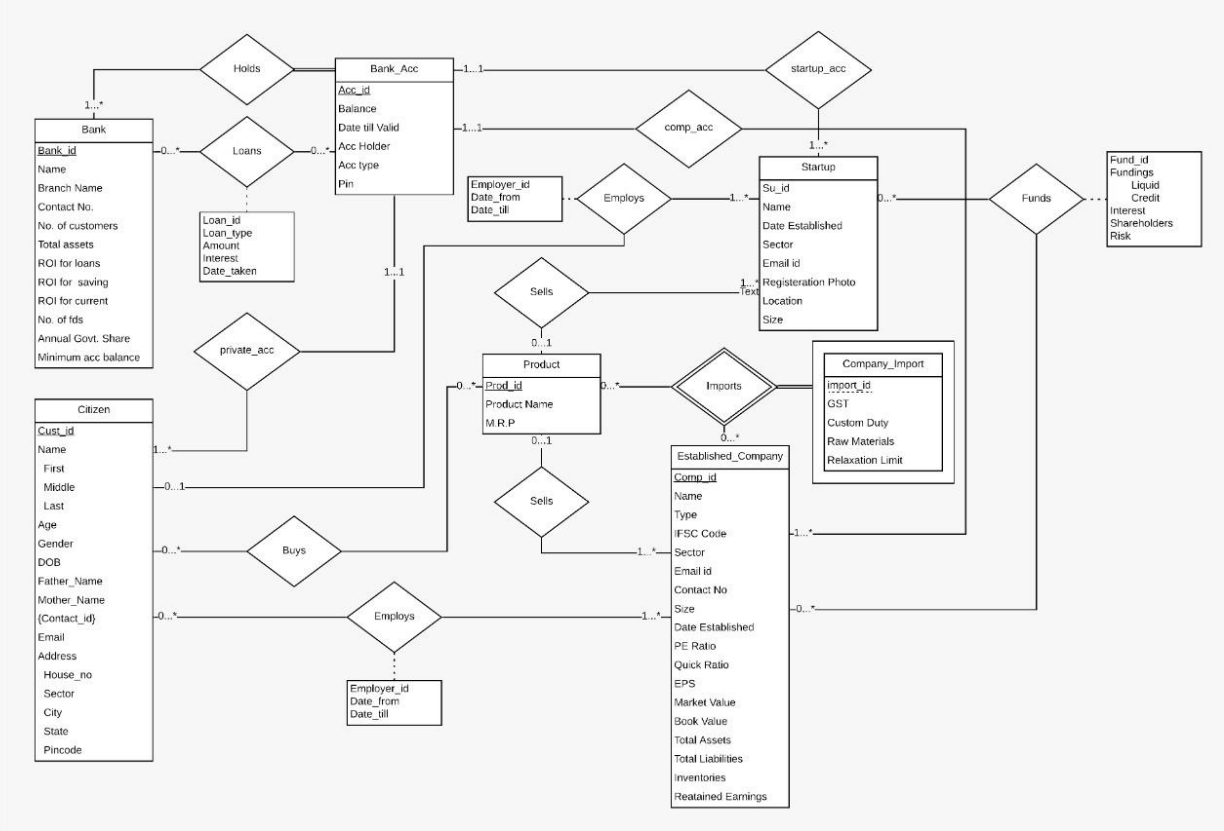
```
   cur.execute("SELECT * FROM bank.bank_loan_detail WHERE loan_type=%s AND
max_period>%s AND bank_id=%s;",[str(type),int(period),str(bank_id)])
```

20)

```
   cur.execute("UPDATE bank.customer_bank_details SET loan_status = 'PENDING' WHERE
(application_id = %s);",[(Customer_id)])
```

Week 8

ER MODEL:



Link:
https://www.lucidchart.com/invitations/accept/9ed95457-f170-42f0-86ac-1108eca42303

Week-9

- Started Frontend Development work in Flask Framework
- Developed Basic Layout of the application
- Made all the Web pages
- Connecting frontend with the database
- Deciding the Innovative features of our application
- Completion of the Bank and Customer related part.

Week 10

BONUS FEATURE

a) Find the Right Investment Plan Feature:

   i) Chooses the appropriate Investment Plan for the user.

   ii) Takes parameters profit, risk, time, capital as input and generates the correct plan

   iii) Plans: Gold, SENSEX, NIFTY, NPS, PPF, FD, Real Estate, Mutual Funds, Bonds

b) Graphical Data Generation:

   i) Graphical Analysis for banks, companies, products, stocks.

   ii) Data from database projected as bar graphs using Python(pygal library)

   iii) For the ease of analysis and more presentable data

c) Live Data Display:

   i) International Market, Cryptocurrency Data, Currency Exchange

ii) Used Web APIs to obtain the data

d) QR Code Generation:

i) Each Bank Customer obtains its unique QR Code

ii) It is an extra-identification feature

Week 11

- Completing the Implementation of Bonus Features
- Completion of Startups and MNCs related part of the project.
- Merging all the different branches of the project made by the team
- Debugging of the whole final project
- Improvements in the User Interface

Week 12

Overall Contribution:

NAME	CONTRIBUTION
Kartikay Sapra	>Features for User, MNC, Bank >Worked across the Flask-MySQL-HTML setup >Ideation and Implementation of the bonus feature parts: Live Data, Investment Plan >PL/SQL Queries >Integration of branches at intermediate steps >Applying basics of stock market and finance >using advanced aggregation functions of SQL
Samarth Chauhan	>Creation of the complete setup/ideation-to-implementation >UI Enhancement >Features for User, MNC, Bank >Worked across the Flask-MySQL-HTML setup >PL/SQL Queries >Integration of branches at intermediate steps > Ideation and Implementation of bonus features part: QR Code, Loan EMI reminder >using advanced aggregation functions of SQL
Ritvik Gupta	>Features for User, MNC, Bank

	>Worked across the Flask-MySQL-HTML setup >PL/SQL Queries >Integration of branches at intermediate steps >Ideation and Implementation of the bonus feature part: Graphical Analysis >using advanced aggregation functions of SQL
Ritwik Gupta	>Database handling/core of data >Normalisation to an optimum level >Indexing >Ensuring Referential Integrity >ER Model creation > Converting SQL Queries to algebraic statements >PL/SQL Queries
Yash Kanojia	>Data Entry in logical format >Indexing >Rectifying Errors in database >Working in MySQL-Workbench >Documentation >Ensuring Referential Integrity