جامعة حلوان

HELWAN UNIVERSITY

Database Course Report
2nd Year Computer Engineering

Bank Management System

20/6/2023

1. The main purpose of the project

In this project we aim to build a system that serves both clients and employees including managers, so we store data about each client and employee in the bank, data like (name, address, id, …etc.), so that we can retrieve it any time to query on it, show it, update it or even delete it.

First, we have accounts table, this table is used to connect client table with employees table using the username and password, so it actually has username, password, email and account type as attributes.

Second, we have clients table which includes all needed data about the client.

Third, we have employees table which includes all needed data about the employee.

Fourth, we have balance table which contains the balance of each client in the bank.

Fifth, we have transaction table which contains each transaction process occurred in the bank.

Sixth, issues table, it is here to store each complain that clients or employees have to tell, so we can improve our bank.

Seventh, managers table, this table purpose is to store all ids of the managers as they are employees as well.

Eighth, loan table, this table is splitted into loans and loan types.

Ninth, investments table, it’s also splitted into investments and investment types.

Finally, relative accounts table, we created this table to help the client if he/she wants to add another user to his/her account.

2. Data model

a) Domain.

Relations.

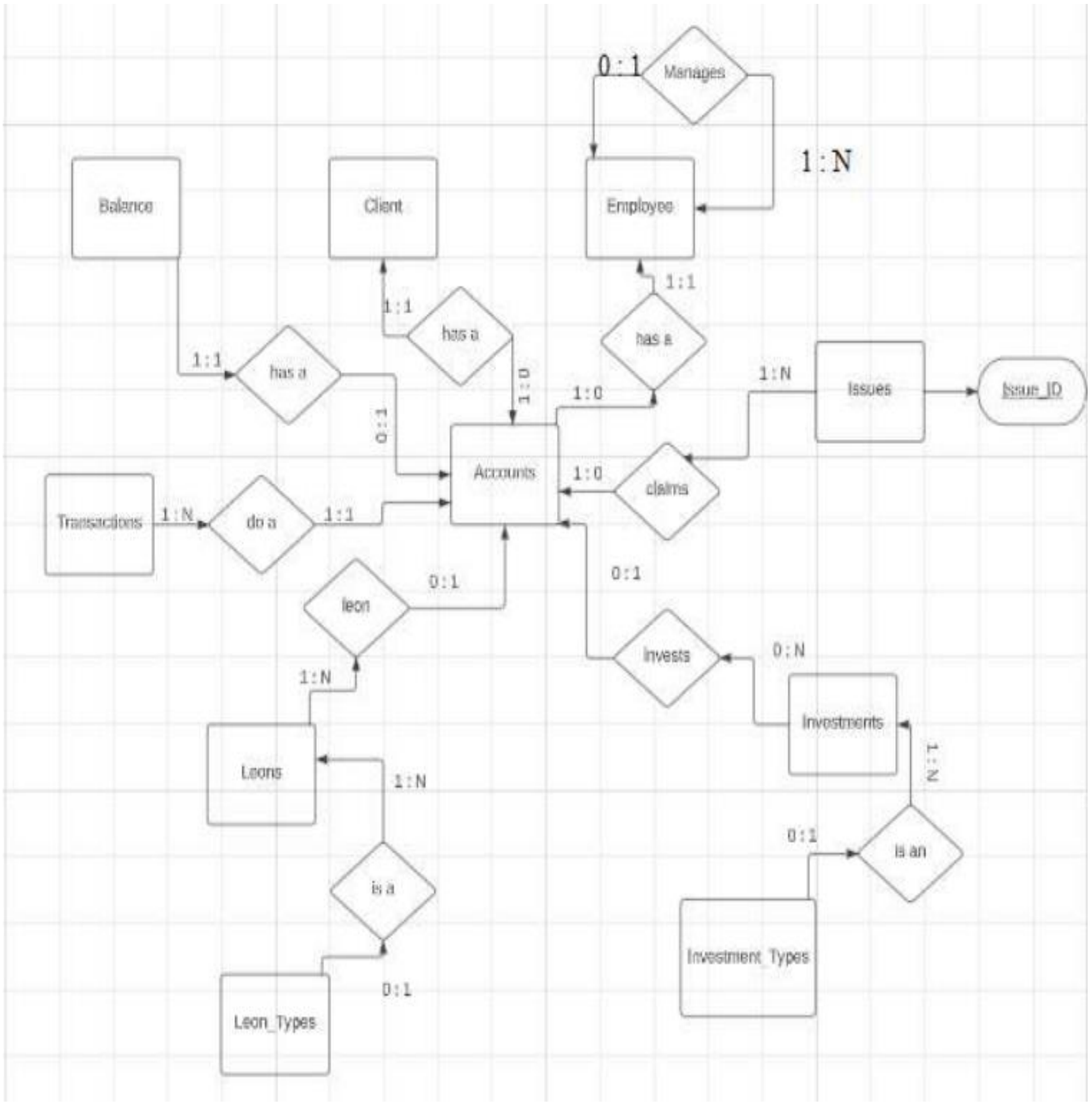
Relation name	No. of columns
accounts	4
balancetabel	2
clienttable	17
employees	17
issues	3
manager	2
transactiontable	5
leon_Types	3
leons	6
investment_types	3
investments	6
Relatives_accounts	6

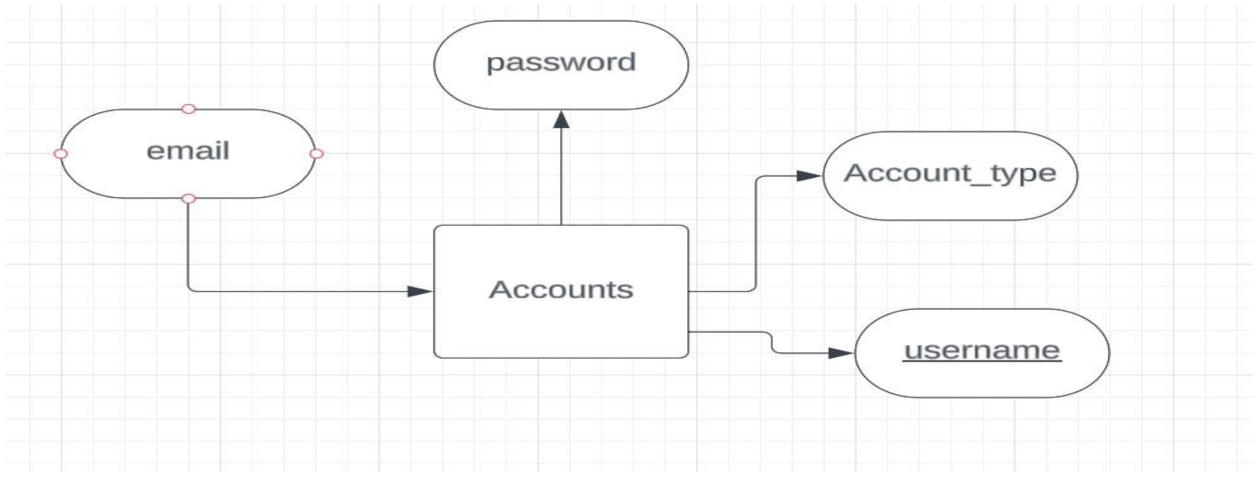
Columns.

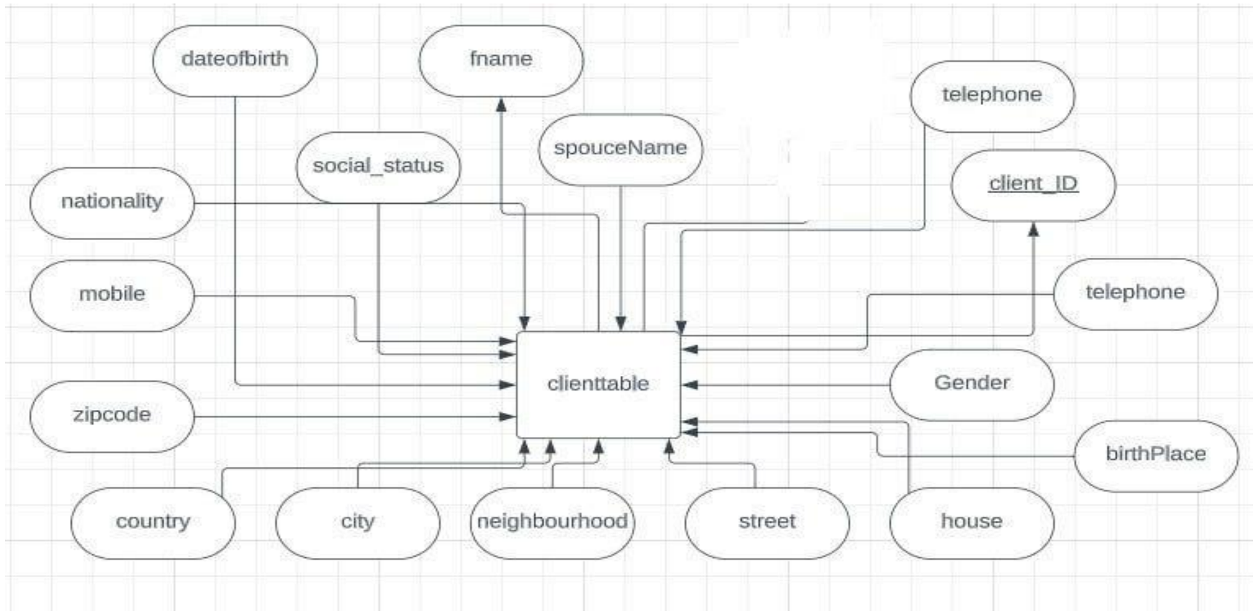
Column name	Column type	Belongs to
username	varchar(30)	accounts, balancetable, clienttable, employees, leons, issues ,investment_types, relatives_accounts
password	varchar(30)	accounts
accountType	int	accounts
email	varchar(50)	accounts
balance	float	balancetable
id_client	int	clienttable, relatives_accounts
fname	varchar(100)	clienttable, employees
dateOfBirth	varchar(15)	clienttable, employees
gender	char(10)	clienttable, employees
telephone	varchar(15)	clienttable, employees
house	int	clienttable, employees

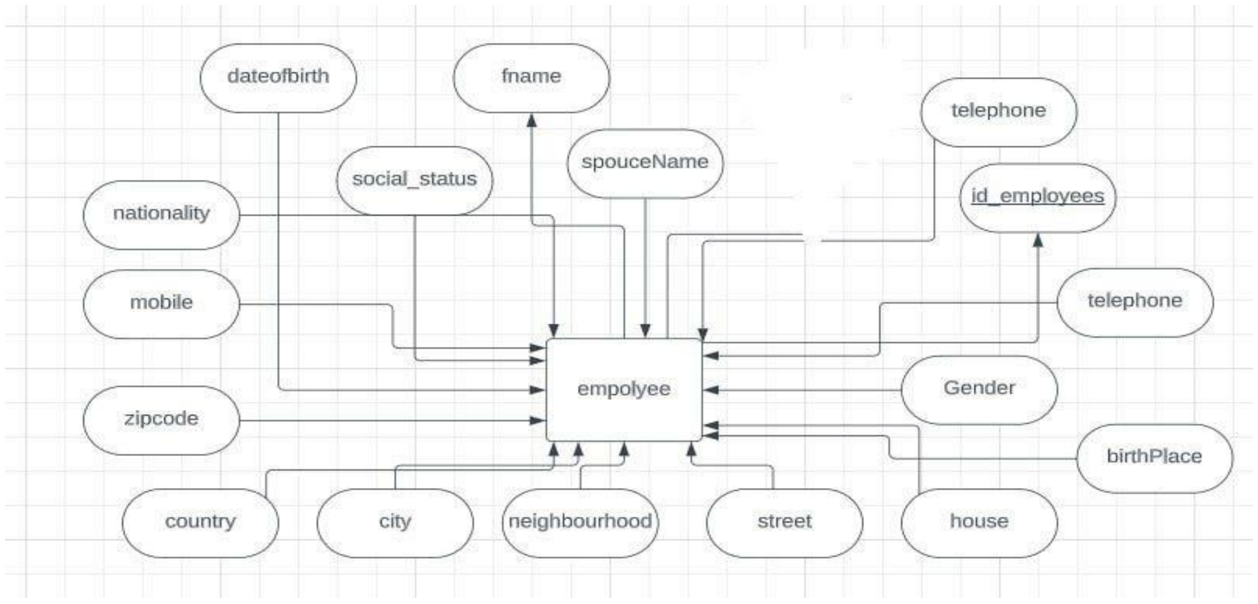
street	varchar(30)	clienttable, employees
neighbourhood	varchar(30)	clienttable, employees
city	varchar(20)	clienttable, employees
country	varchar(20)	clienttable, employees
zipcode	varchar(10)	clienttable, employees
mobile	varchar(30)	clienttable, employees
nationality	varchar(50)	clienttable, employees
social_status	char(10)	clienttable, employees
spouceName	varchar(30)	clienttable, employees
birthPlace	varchar(50)	clienttable, employees
id_employees	int	employees
id_manager	int	manager
id_emp	int	manager
id_bankBalance	int	transactiontable
username1	varchar(30)	transactiontable
username2	varchar(30)	transactiontable
amount	int	transactiontable
dateOfTransaction	varchar(15)	transactiontable
leon_ID	int	leon_types, leons
leon_amount	int	leon_types
leon_no_of_dues	int	leon_types
leon_number	int	leons
due_amount	int	leons
start_date	varchar(15)	leons
end_date	varchar(15)	leons
investment_ID	int	investment_types, investments
investment_ammount	int	investment_types
investment_profit_percentage	float	investment_types
investment_number	int	investments
startdate	varchar(15)	investments
enddate	varchar(15)	investments
issue_ID	int	issues
issue_details	varchar(200)	issues
relative_ID	int	relatives_accounts
relative_name	varchar(100)	relatives_accounts
max_ammount	int	relatives_accounts
max_withdraw	int	relatives_accounts

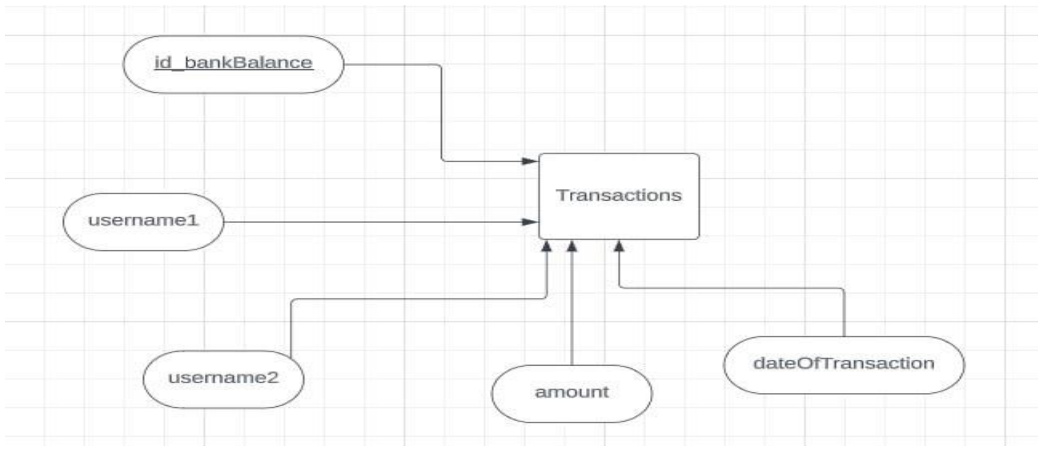
b) Conceptual data model diagram.

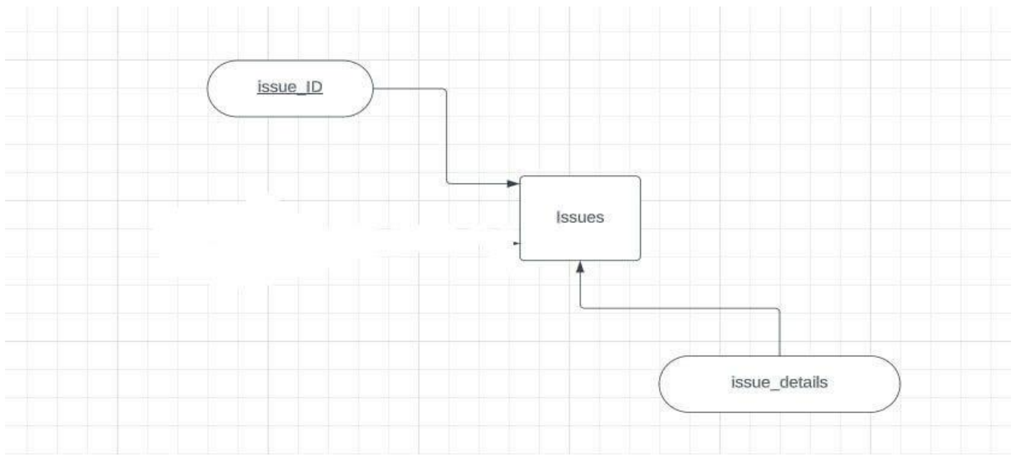


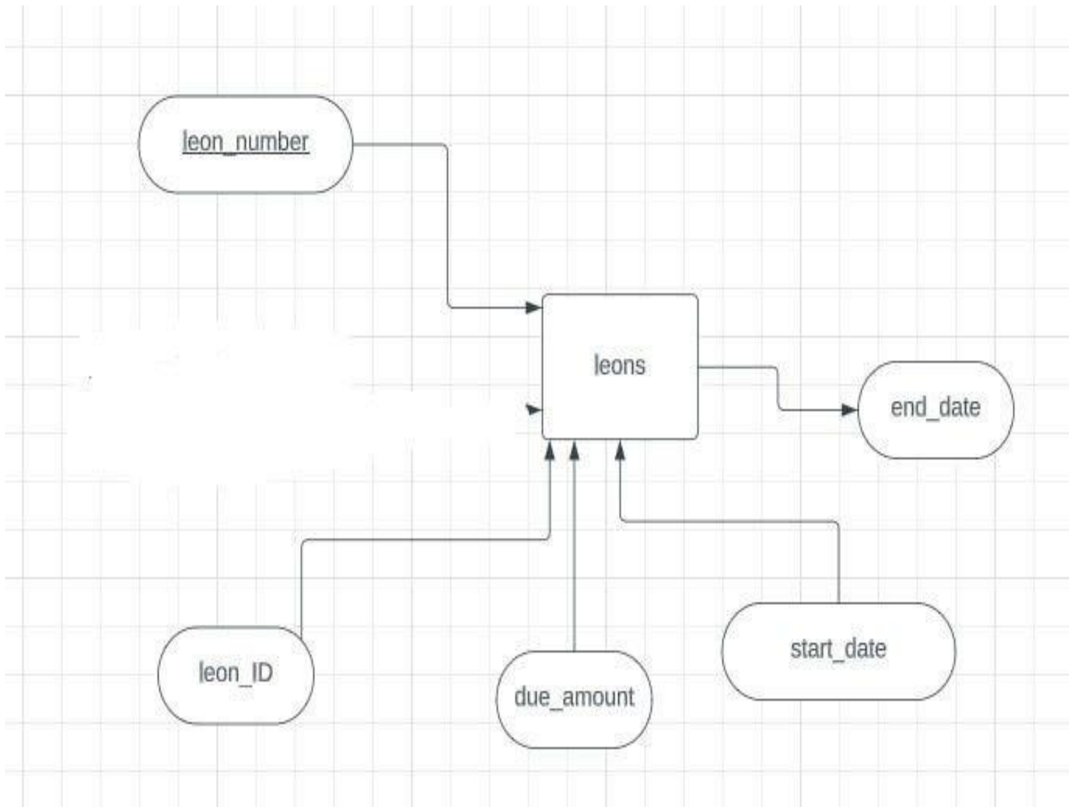

password
email
Account_type
Accounts
username

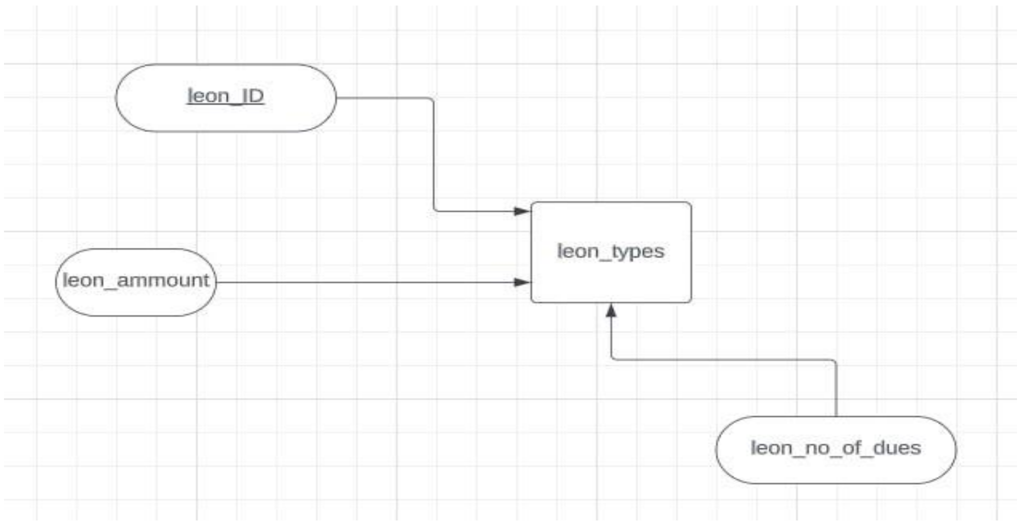

dateofbirth
fname
telephone
social_status
spouceName
nationality
client_ID
mobile
telephone
clienttable
Gender
zipcode
birthPlace
country
city
neighbourhood
street
house

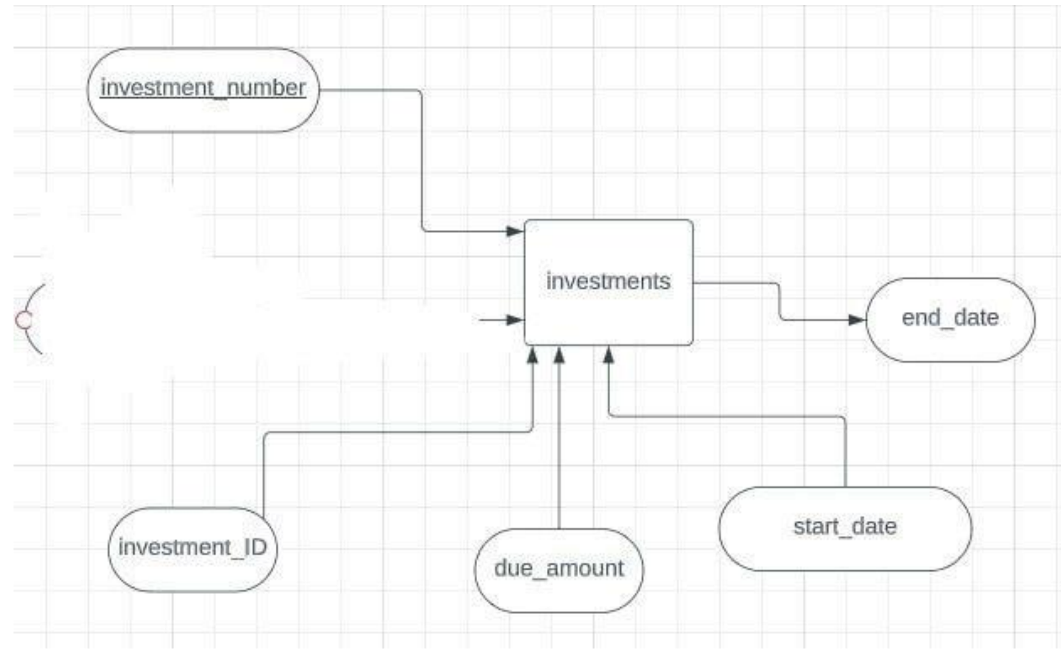

dateofbirth
fname
telephone
social_status
spouceName
nationality
id_employees
mobile
telephone
empolyee
Gender
zipcode
birthPlace
country
city
neighbourhood
street
house

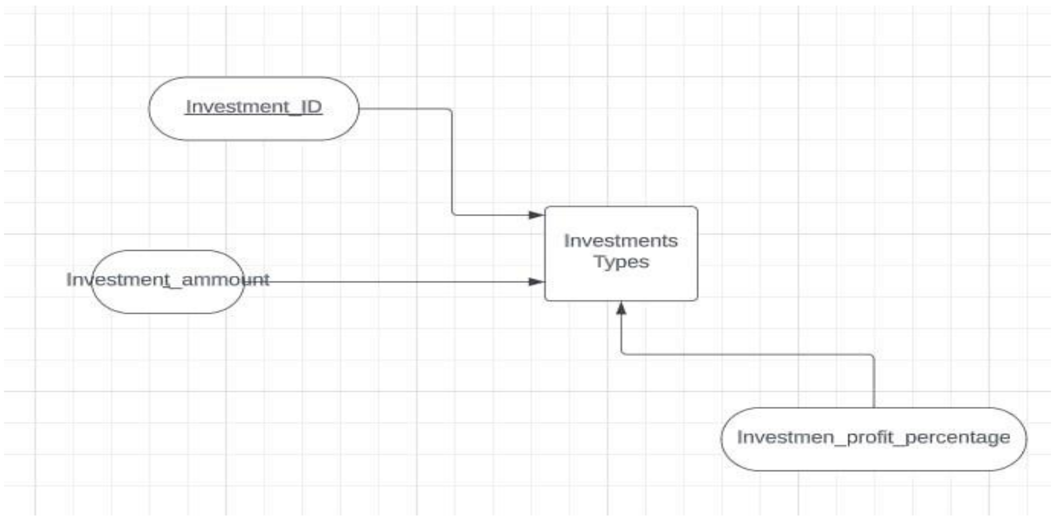

id_bankBalance
Transactions
username1
username2
amount
dateOfTransaction


issue_ID
Issues
issue_details

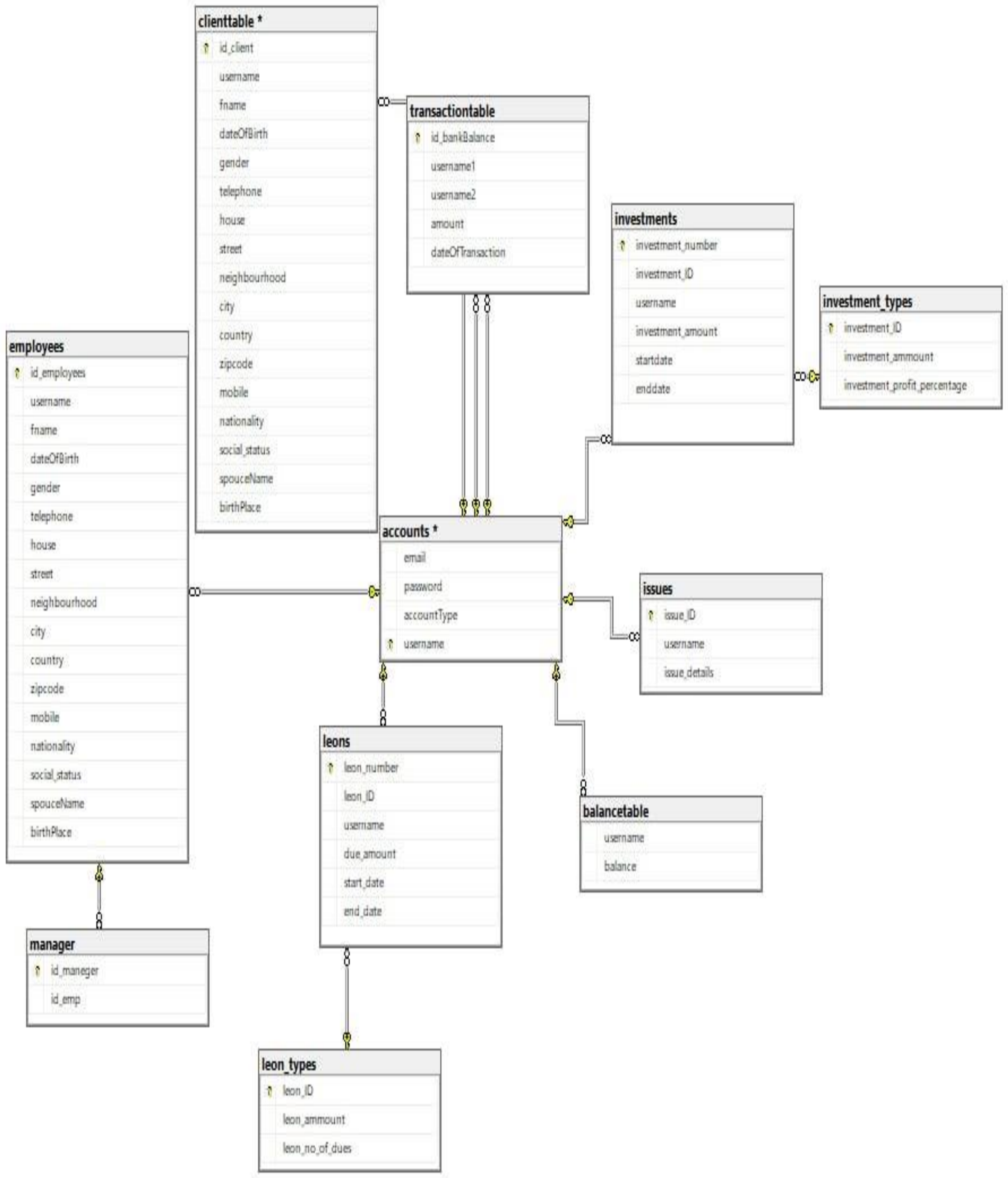

leon_number
leons
end_date
leon_ID
due_amount
start_date


leon_ID
leon_ammount
leon_types
leon_no_of_dues


investment_number
investments
end_date
investment_ID
due_amount
start_date


Investment_ID
Investments Types
Investment_ammount
Investmen_profit_percentage

3. Logical and physical modeling



4. Database implementation

Accounts table:

```
CREATE TABLE  accounts (
  email varchar(50) UNIQUE,
  password varchar(30) NOT NULL,
  accountType int NOT NULL,
  username varchar(30) primary key,
);
```

Clients table:

```
CREATE TABLE clienttable (
  id_client int identity(1,1)  primary key,
  username varchar(30) not null,
  fname varchar(100) not null,
  dateOfBirth varchar(15),
  gender char(10) not null,
  telephone varchar(15) not null,
  house int not null,
  street varchar(30) not null,
  neighbourhood varchar(30) not null,
  city varchar(20) not null,
  country varchar(20) not null,
  zipcode varchar(10) not null,
  mobile varchar(30) not null,
  nationality varchar(50) not null,
  social_status char(10) not null,
  spouceName varchar(30) not null,
  birthPlace varchar(50),
  CONSTRAINT FK_clienttable_accounts_email  FOREIGN KEY (username)
REFERENCES accounts (username)
);
```

Employees table:

```
CREATE TABLE employees (
  id_employees int identity(1,1) primary key,
  username varchar(30) not null,
  fname varchar(100) not null,
  dateOfBirth varchar(15),
  gender char(10) not null,
  telephone varchar(15) not null,
  house int not null,
  street varchar(30) not null,
  neighbourhood varchar(30) not null,
  city varchar(20) not null,
  country varchar(20) not null,
  zipcode varchar(10) not null,
  mobile varchar(30) not null,
  nationality varchar(50) not null,
  social_status char(10) not null,
  spouceName varchar(30) not null,
  birthPlace varchar(50),
  CONSTRAINT FK_employees_accounts_email  FOREIGN KEY (username) REFERENCES
accounts (username)
);
```

Managers table:

```
CREATE TABLE   manager (
  id_maneger int identity(1,1) primary key,
  id_emp int foreign key references employees(id_employees),
);
```

Balance table:

```
CREATE TABLE  balancetable (
  username varchar(30),
  balance float  NOT NULL ,
  CONSTRAINT FK_BalanceTable_accounts_email FOREIGN KEY (username)
REFERENCES accounts (username),
);
```

Transaction table:

```
CREATE TABLE transactiontable (
  id_bankBalance int identity(1,1) primary key  ,
  username1 varchar(30) not null,
  username2 varchar(30) not null,
  amount int NOT NULL,
  dateOfTransaction varchar(15) NOT NULL,
  CONSTRAINT FK_transactiontable_transactiontable_accountNo FOREIGN KEY
(username1) REFERENCES accounts (username),
  CONSTRAINT FK_transactiontable_transactiontable_accountNoRecipient
FOREIGN KEY (username2) REFERENCES accounts (username),
);
```

Loans tables:

```
  create table leon_types
  (
   leon_ID int identity(1,1) primary key,
   leon_ammount int not null,
   leon_no_of_dues int not null --الاقساط
  );

  create table leons
  (
  leon_number int identity(1,1) primary key,
  leon_ID int not null,
  username varchar(30) not null,
  due_amount int not null,       -- derived from amount  & no. of dues
  start_date date not null,
  end_date date not null,
  constraint fk_leons_idclient foreign key (username) references
accounts(username),
  Constraint fk_leons_leonid foreign key (leon_ID) references
leon_types(leon_ID),
    );
```

Investment tables:

```
create table investment_types
  (
   investment_ID int identity(1,1) primary key,
   investment_ammount int not null,
   investment_profit_percentage float not null
  );

 create table investments
  (
  investment_number int identity(1,1) primary key,
  investment_ID int not null,
  username int not null,
  startdate date not null,
  enddate date not null,
  Constraint fk_investments_idclient foreign key (username) references
clienttable(username),
  constraint fk_investments_investmenttypes foreign key (investment_ID)
references investment_types(investment_ID)
  );
```

Issues table:

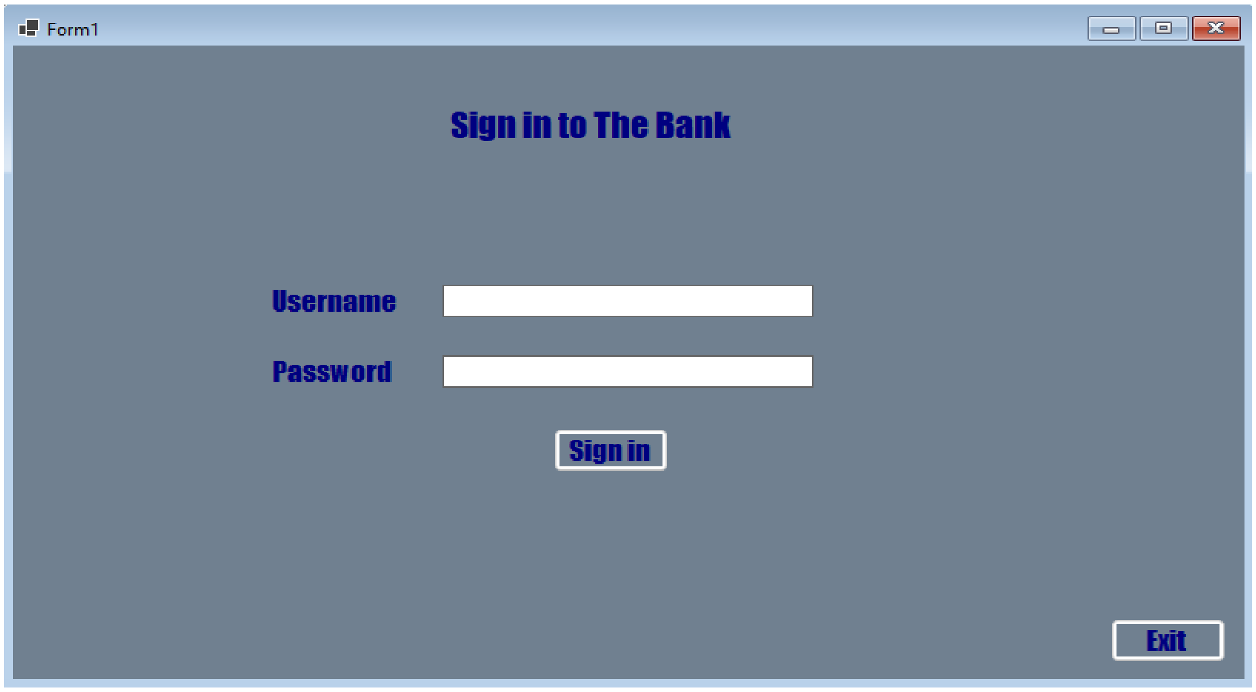
```
create table issues
  (
   issue_ID int identity(1,1) primary key,
   username varchar(30) not null,
   issue_details varchar(200) not null,
  constraint fk_issues_idclient foreign key (username) references
accounts(username),
    );
```

Relatives table:

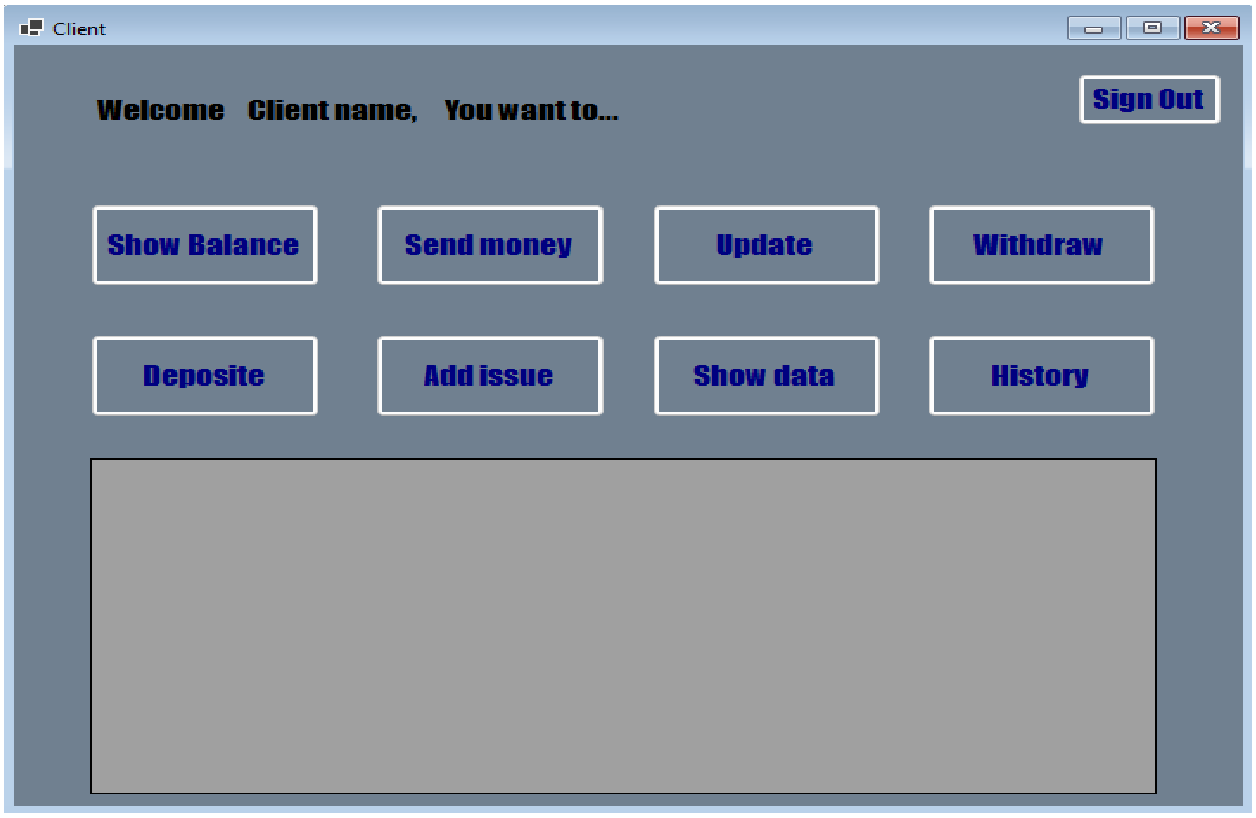
```
create table relatives_accounts
  (
  relative_ID int identity(1,1) primary key,
  relative_name varchar(100) not null,
  username varchar(30) not null,
  max_ammount int not null,
  max_withdraw int not null,
  id_client int not null,
  constraint fk_relatives_clientid foreign key (id_client) references
clienttable(id_client),
  constraint fk_relatives_username foreign key (username) references
accounts(username)
 );
```

5. Application implementation

The application starts with the sign in form.

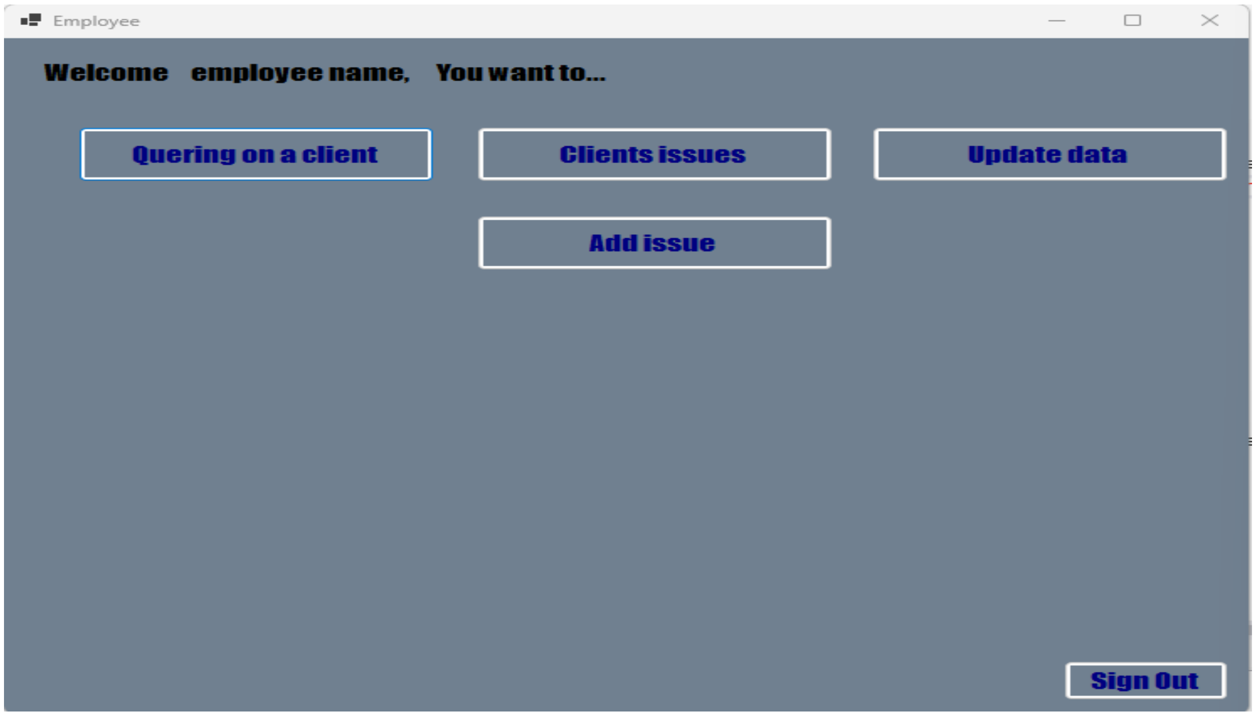


This form used to sign into the bank as a client, employee or manager, if you signed in as a client you will be directed to:



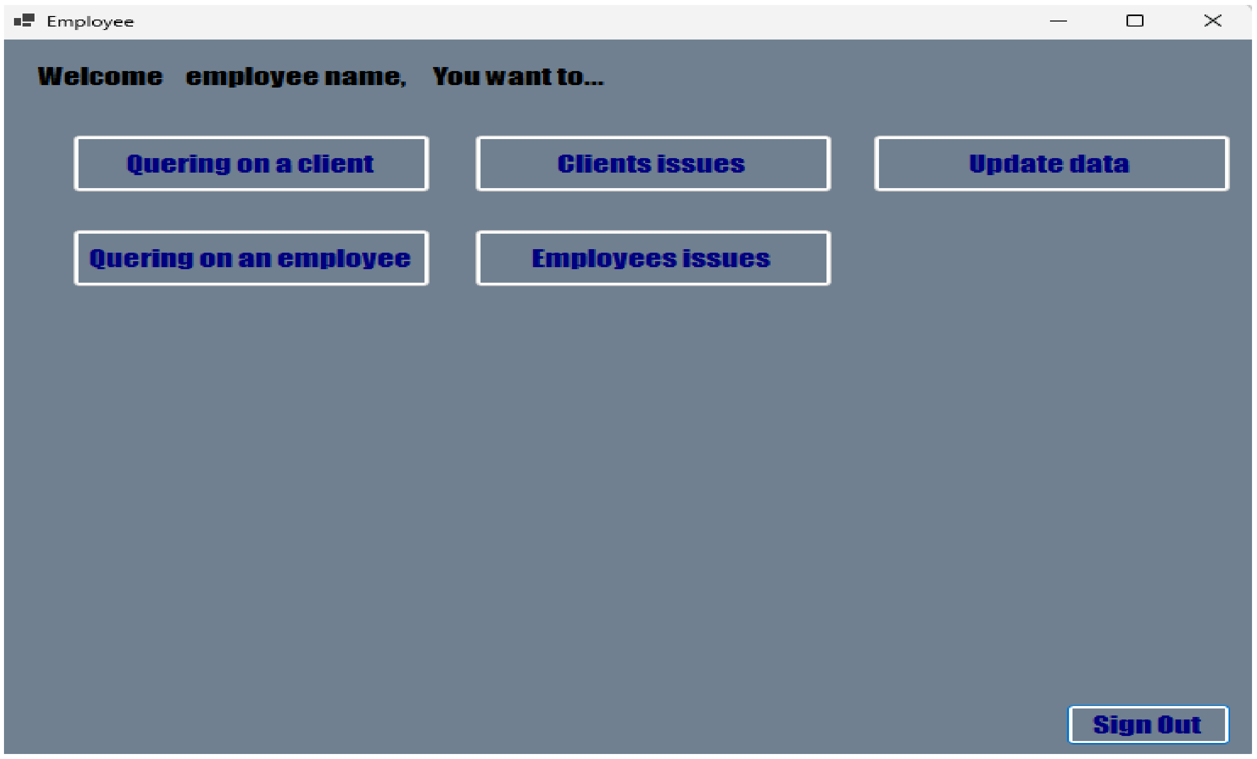
As a client you can see your balance, send money to other accounts, update your profile, withdraw, deposit, add issue, show your data, show your transaction history.

if you signed in as an employee you will be directed to:



Cause as you as an employee you can add, update clients , show clients problems or add issue yourself.

And as manager:



You can do anything employees can in addition to adding and removing employees and clients and show issues of all users.

And last but not least these are some queries that we used in our system:

1- To check if you are manager:
```
select * from accounts, employees, manager where
(accounts.username=employees.username and accounts.username=@username)
and (manager.id_emp=employees.id_employees)
```
2- To check either you are employee or client:
```
select * from accounts,balancetable where accounts.username = @username
and accounts.username = balancetable.username
```
3- To show client data:
```
select distinct id_client,fname, dateOfBirth, gender, telephone, house,
street, neighbourhood, city, country, zipcode, mobile, nationality,
social_status, spouceName, birthPlace, clienttable.username, email,
password from clienttable, accounts where accounts.accountType=2 and
clienttable.username = accounts.username
```
4- To withdraw money:
```
update balancetable set balance = @balance1 - @balance2 where username
= @username
```
5- To insert into transaction table:
```
insert into transactiontable values('" + Program.usern + "', '" +
Program.usern + "','" + y + "','no date'
```