

4/6/2022

# Give back, donation platform

KSU IS370 Project

KSU CSIS

441101772	فهد الكديان
441103156	خالد الرويتع
441103220	حاتم الطريري

## Contents

Introduction: .....	2
System analysis and design: .....	2
Project management:.....	2
UML Diagram: .....	3
Client class.....	3
Use case Diagram:.....	4
State machine diagram: .....	5
Network Aspect:.....	6
Result .....	7
Data structure: .....	10
Challenges: .....	11

### Introduction:

Under the supervision of Dr. Abdullah Alghofaili we started our journey to make a donation service, that is accessible to anyone who has a connection. The goal of the program is to show our understanding of the makings of network application.

### System analysis and design:

Every good project starts with the analysis of the system, our analysis concluded that our program got seven use cases. Based on these use cases we made the following UML which does not have any complex relation that because no UML survives the programing challenge And with our methodology of Extreme programming, decreases the importance of UML relationships.

### Project management:

After we finished our design and, understood our system it is time to start programming, our first sprint was about the making the associate classes such as project, donation, and project.

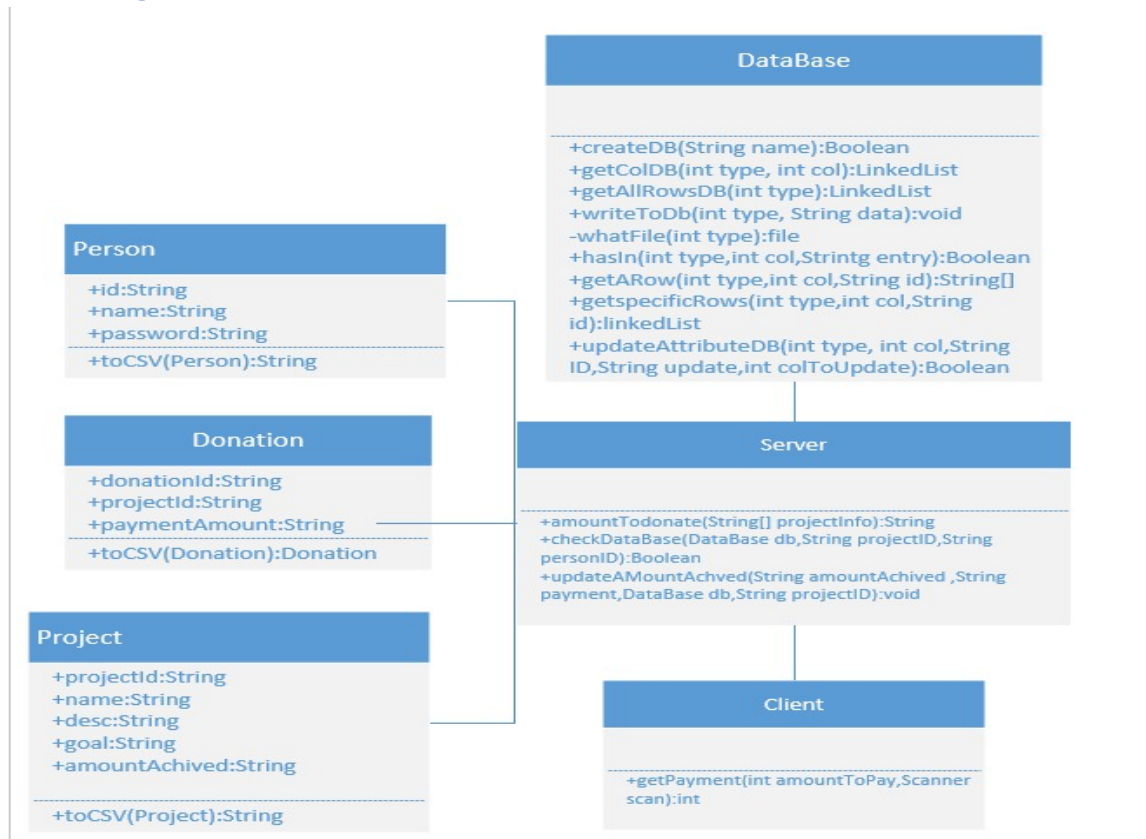
The second sprint was about database and the unit testing environment. After Every method we made sure to unit test it to perfection.

The third sprint was about server and client side, which got thrown out because, we overlooked a major point, which is that no modern app especially ones that deals with real money will allow the client to deal with any logic/calculation.

For the fourth sprint we specifically agreed that all logic should be handled at the server side and that was good decision since the database does not have any tool to keep the integrity of the database safe. What we mean is the database will accept info information from the server corrupted or not.

\*Even after acknowledging this problem, the comfort of dealing with some of the logic in the client side won over the security aspect.

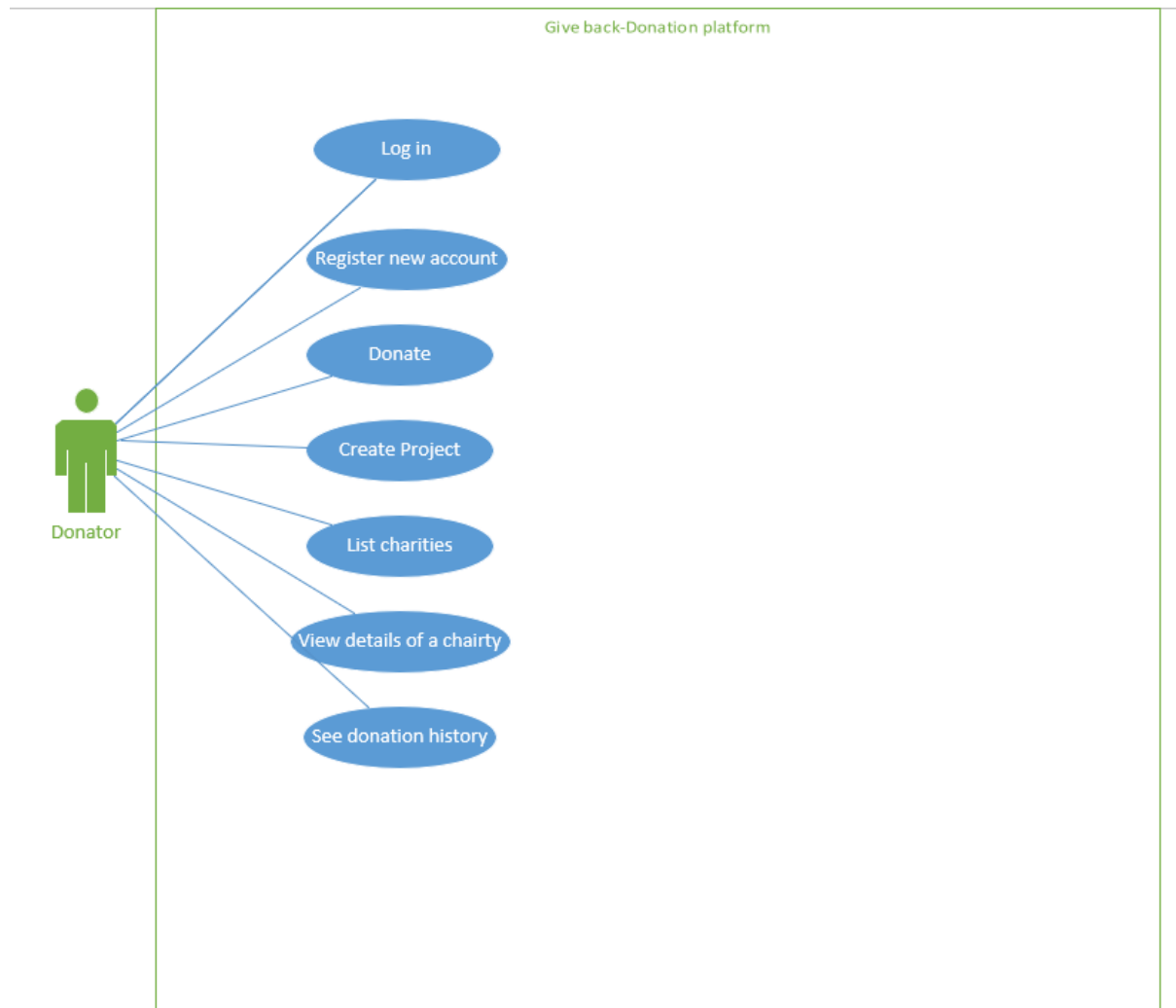
## UML Diagram:



## Client class

Is responsible for sending request and data, from user to client. Sever is an intermediate between the Client and Database .it works to keep data confidential and integrity . the other classes work as a supporter for server.

## Use case Diagram:

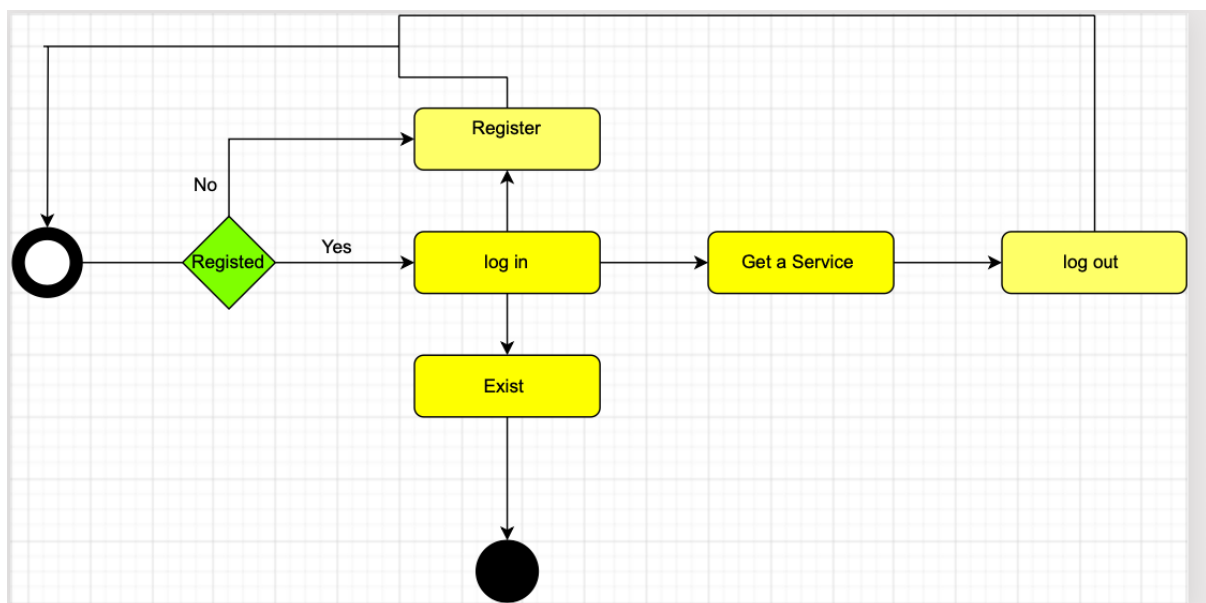


This picture shows that our program has only one external actor.

## Use Cases

1. Log In :  
User will enter the password and user name then he will have access if the account is valid .
2. Register :  
User will Enter account information then the system will create an account if the ID does not exist .
3. Donation :  
User will search for specified project . system will return the project information a the range of payment .finally the user will enter payment amount then the system will return the ID for the donation .
4. Create charity :  
System will show the available charities for the user
5. View Details Charity :  
User will enter project ID then System will print it's information
6. See History  
System will print all the donation that the user has made .

## State machine diagram:



This diagram reflects the journey of the user. it starts with creating an account so, he can use the services we have.

### Network Aspect:

- **Server Class**

we created new object of ServerSocket as serverSocket with port number (5555) to wait for clients to come over the network, (the system is waiting for clients to come over the network).

we created an object of Socket as socket to accept the connection coming from the client (if the client is initialized then the system will display a message that client is created).

To read the input from client we used DataInputStream object as sdis.

To send the messages from server to client we used DataOutputStream object as sdout.

- **Client Class**

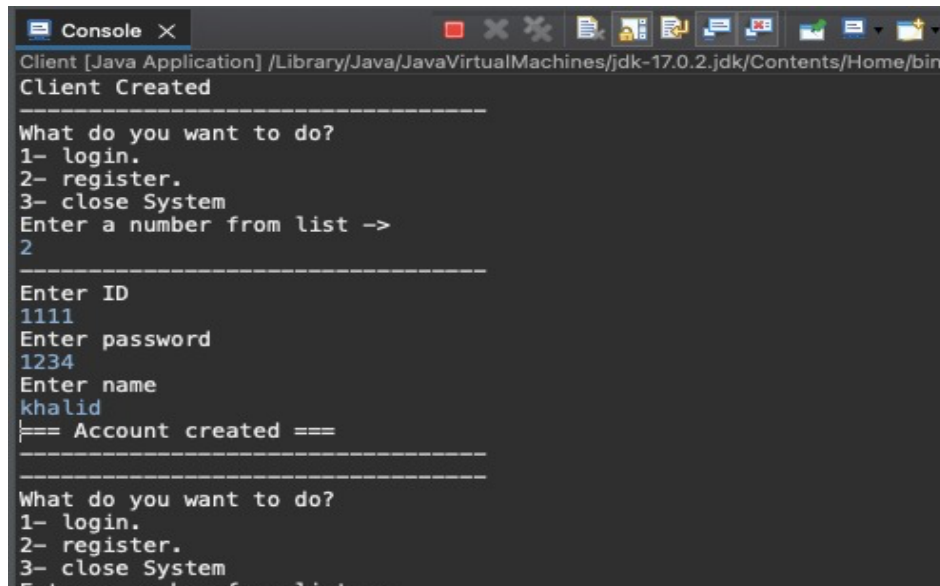
we created a client socket as socket to communicate with the Server with the same port number (5555).

To read the message coming from the server we used DataInputStream object as cdis , and we directed it into a print statement such System.outprintln(cdis.readUTF());.

To send messages to server over the network we used DataOutputStream object as cdout.

## Result

In this section we represent the result that the user gets when he interacts With the system from client side.



```

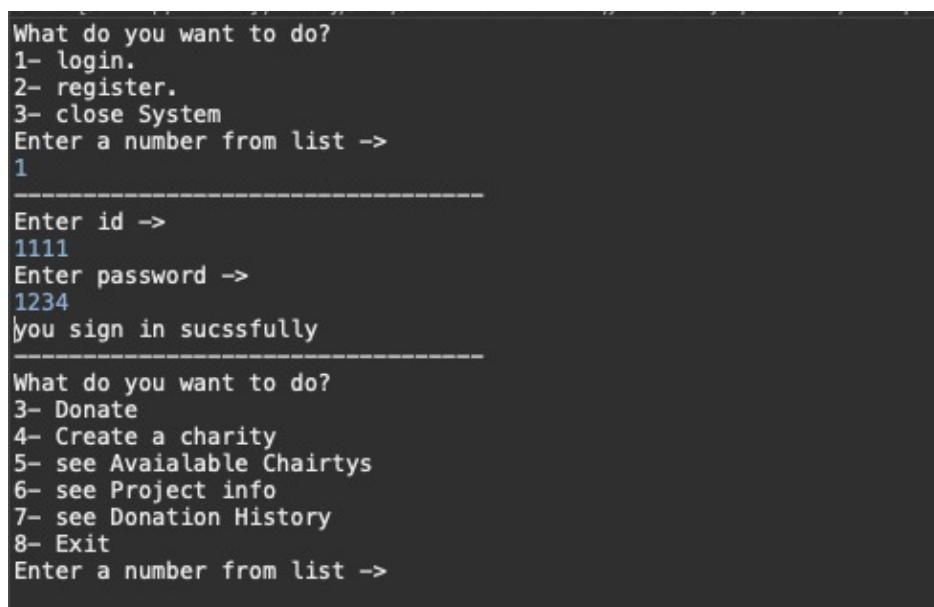
Client [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.2.jdk/Contents/Home/bin/
Client Created
=====
What do you want to do?
1- login.
2- register.
3- close System
Enter a number from list ->
2
=====
Enter ID
1111
Enter password
1234
Enter name
khalid
=== Account created ===
=====
What do you want to do?
1- login.
2- register.
3- close System
Enter a number from list ->

```

Case Account Register new account,

As we explained in the introduction, the user starts with creating an account.

The client will send this information to the sever then the server will insert it to the data base if the ID does not exist.



```

What do you want to do?
1- login.
2- register.
3- close System
Enter a number from list ->
1
=====
Enter id ->
1111
Enter password ->
1234
you sign in succssfully
=====
What do you want to do?
3- Donate
4- Create a charity
5- see Avaialable Chairtys
6- see Project info
7- see Donation History
8- Exit
Enter a number from list ->

```

Case Log in,



Client sends this information to the Server. the sever will validate the password and the ID. If it exists, then the user will have the access to the service

```

=====
What do you want to do?
3- Donate
4- Create a charity
5- see Avaialable Chairtys
6- see Project info
7- see Donation History
8- Exit
Enter a number from list ->
4
=====
Enter name
AHSAN
Enter projectDese
0
Enter goal
1000
charity has been made
charity id:952
=====

```

Case create Project(charity),  
the client sends the information to the sever. The server will create ID number for the charity then insert it to the data base.

```

=====
What do you want to do?
3- Donate
4- Create a charity
5- see Avaialable Chairtys
6- see Project info
7- see Donation History
8- Exit
Enter a number from list ->
3
=====
Enter project ID
952
Enter the amount you want to Pay
Note, The range of payment is between 1 to 1000
*Enter 0 to stop
100
Donation has been made
Donation id: 25
=====

```

Case Donate,  
the client will send to the sever the charity ID that the user wants to donate in. the sever will check the data base and return the achieved amount to the client. user will enter the amount (amount paid <amount achieved goal)

```

5- see Avaialable Chairtys
6- see Project info
7- see Donation History
8- Exit
Enter a number from list ->
5
-----
Project id      Project name    Project desc    Project goal    Proj
286            khalid         0               1000            -201
951            ahsan          0               10000           0
411            ahsan          0               1000            0
952            AHSAN          0               1000            100
-----
What do you want to do?
3- Donate
4- Create a charity
5- see Avaialable Chairtys
6- see Project info
7- see Donation History
8- Exit
Enter a number from list ->

```

Case List charities,  
the sever will retrieve all the charity for the database in a list. Then the sever will start sending the list node by node.

```

5- see Avaialable Chairtys
6- see Project info
7- see Donation History
8- Exit
Enter a number from list ->
6
-----
Enter project ID
951
Project id      Project name    Project desc    Project goal    Proj
951            ahsan          0               10000           0
-----
What do you want to do?
3- Donate
4- Create a charity
5- see Avaialable Chairtys
6- see Project info
7- see Donation History
8- Exit
Enter a number from list ->

```

Case view details of charity,  
client will send a charity ID to the sever. server will retrieve the charity information form the data base then return it to the client.

```
5- see Avaialable Chairtys
6- see Project info
7- see Donation History
8- Exit
Enter a number from list ->
7
-----
Donation ID      person ID      Project ID      payment amont
25              1111          952             100
-----
What do you want to do?
3- Donate
4- Create a charity
5- see Avaialable Chairtys
6- see Project info
7- see Donation History
8- Exit
Enter a number from list ->
```

Case See donation history,  
client will send the ID that the user has log in with. The sever will insert all the donations that related to user in a list. At the end server will send the list node by node

---

### Data structure:

We had many ideas on how to deal with data and storage in general.  
first idea we did have was to use SQL like service, and host it on the cloud, but the idea didn't hold ground since it required the client to download the appropriate tools to run the program, and it was hard. So, we went with the next best thing which was CSV file format. To avoid any problems we agreed that the primary key should be the first column and that the CSV structure will be header less since we didn't need to deal with any other data base.

All our data are strings since string is easier to deal with and if needed using proper method you can convert strings to int if they are compatible or also you can return them to their ASCII values.

The coding of the data base was done as a separate system all the methods are done assuming it will interact with foreign, but TRUSTED system. This was done on purpose in case the team wanted to reuse the code or reuse the entire class for future updates on this project or other, it is 100% possible since the only things that need to change is the class variables and whatFile (int type) method.

## Challenges:

- 1- In donation case we face problem that we have a ceiling for donation, and we should stop the donation process if it achieved.
- 2- In donation case there is possibility to enter negative number which will sabotage The calculations and we handle it by putting conditions.
- 3- The initiation of the database and how to extract the data, add and modify it.
- 4- Dealing with bad user after all Garbage in garbage out.
- 5- Keeping the integrity of the database safe. Since the database accepts everything.
- 6- We made the cardinal sin of project management, which is not having a project charter and to be exact is not defining the **scope** this have made us lose a lot of time in devolving things that were unnecessary at best and harmful at worst.
- 7- Defining relationships in UML diagram, as I have stated before no UML will stand real world implementation of the program, especially the relationships. hours wasted deciding between aggregation/composition when at the end of the day it was not important.