My Fancy Bank Documentation

Yufeng Chen, yufeng72@bu.edu

- Main Class List

`Account.java`: the abstract class to represent a bank account. It can be extended to implement more accounts like student account or company account. It has variables like balance and account type, and some abstract methods for its subclass to implement.

`AccountChecking.java`: extends Account, class to represent a checking account.

`AccountSaving.java`: extends Account, class to represent a basic saving account. It can be extended to implement more kind of saving accounts like CD saving account or IRA saving account.

`BankDataBase.java`: I use this class as a database to store all bank data. All client operations will go through this class and it will generate a record for every transaction like registration, deposit/withdraw or transfer.

`Client.java`: class to represent a client. A client can have several accounts of different account type, and supports all kinds of transactions: deposit, withdraw, transfer, manage loans, and manage accounts using the ATM interface.

`Currency.java`: class to support multiple currencies.

`Loan.java`: class to represent a loan. A client can maintain several loans.

`TransactionRecord.java`: class to represent a transaction record, used for the database to record all clients' valid transactions. The Manager can view these records through Bank Management GUI.

`FrameATM.java`: ATM GUI designed for users. Users can register, login, deposit, withdraw, transfer, request a loan, repay a loan, view accounts, and add new accounts here. I design several sub frames for these operations each. All client operation logic are implemented in these frame classes.

`FrameBankManager.java`: Bank Management GUI designed for the manager to view all clients, their accounts, their loans and all transactions the bank database generated. All manager operation logic are implemented in this class.

`FrameTest.java`: because no data storage is applied for this design, I write this class as a backend to maintain all data in memory when testing. As result you can open and close ATM GUI or Bank Management GUI several times when testing as long as you keep this frame open.

`Test.java`: entry class to test all my code.

- Statements:

The bank rules provided in the pdf is not very clear so I add some statements myself.

1. To make test easier, I didn't set login password limit, so you can just use any simple String as password (but make sure you re-enter the password right when registration!). For the same reason I didn't implement transaction password which is required in real life. Other information like address and phone number is not necessary in this so I didn't record them at all (but I do add input for these information when registration to make the process looks more real XD).

2. I didn't implement account delete function because it is associated with balance and it is not required for this assignment.

3. I think the bank manager shouldn't do anything to the bank data so in my design the manager can only view some data in the database like a client's accounts and loans, and cannot change them. The manager is not able to see the client's login password because he/she doesn't need to know that to manage the bank.

4. Only deposit, withdraw and transfer support multiple currencies. My implementation supports three currencies: USD, EUR and CNY. For easier management, the balance in the account is saved as USD in the database. The amount occurred in the transaction record which will be presented to the manager is exchanged to USD too so that the manager can have a more clear sense of the amount value.

5. For every registered client (with a unique username), we will create a checking account (with balance $100) and a saving account (with balance $0) as default accounts right after registration. That means you need to pay $100 to register as a client.

6. I used account ID instead of account number in real life as a unique identification. When an account is created, its unique account ID [username + "Acc" + account number] is created too and not changeable. For example, if my username is Allen, then my default accounts should have ID "AllenAcc1" and "AllenAcc2", and my next new account's ID will be "AllenAcc3". You will need to enter that ID if you want to transfer money to someone.

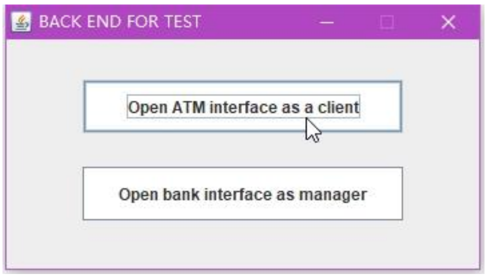
7. You can only open one sub window at the same time. When you click on the parent window when there exist a sub window, nothing will happen.

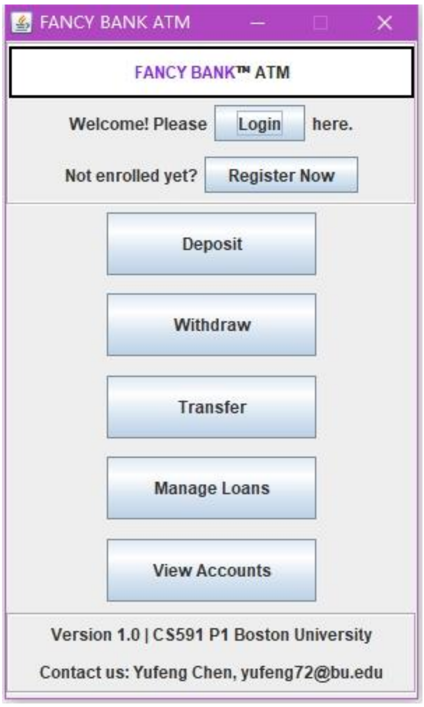
8. Withdraw and transfer made from a checking account will be charged a fee. A loan's repayment will increase every month.

9. Because there is no data storage in this assignment, I set all loans' time period as one month for you to test transaction "repay a loan". But I didn't set time period for the saving accounts to calculate interest (only implemented some methods in the class).

10. I think it's the bank manager's job to calculate how much the bank earned when he/she decides to do so (maybe after all loans have been repaid) according to all records, so I didn't calculate it in the database.

- Some Running examples:


BACK END FOR TEST
Open ATM interface as a client
Open bank interface as manager

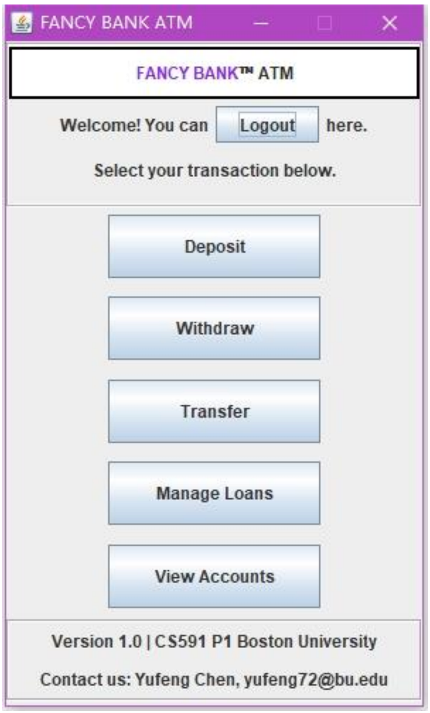


Welcome! Please Login here.

Not enrolled yet? Register Now

Version 1.0 | CS591 P1 Boston University

Contact us: Yufeng Chen, yufeng72@bu.edu



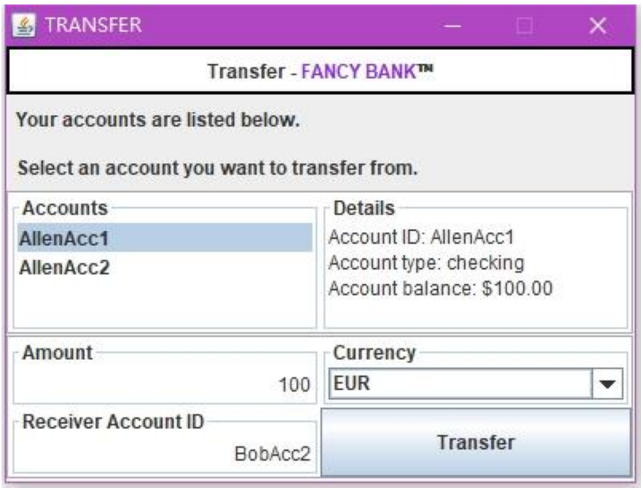
Welcome! You can Logout here.

Select your transaction below.

Version 1.0 | CS591 P1 Boston University

Contact us: Yufeng Chen, yufeng72@bu.edu

Login: open ATM GUI, click login button to open a new login window. Enter username and password then click login. After logged in you can click the new logout button to log out.



Your accounts are listed below.

Select an account you want to transfer from.

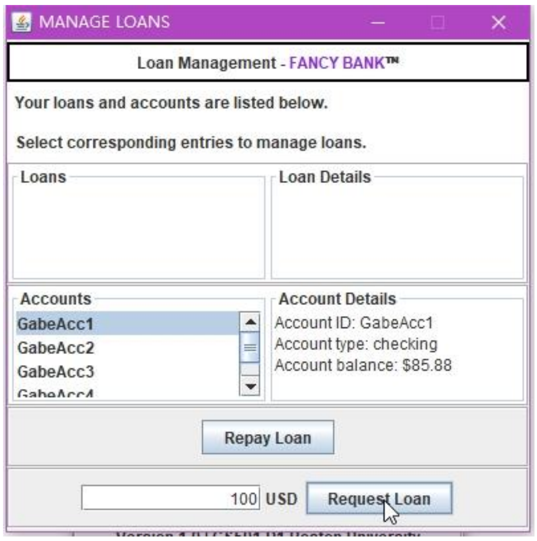
Account ID: AllenAcc1
Account type: checking
Account balance: $100.00

Amount: 100

Currency: EUR

Receiver Account ID: BobAcc2

Transfer: click transfer button on the ATM GUI to open a new transfer window. Select your account, enter amount, enter receiver account ID, choose currency, then click transfer button.



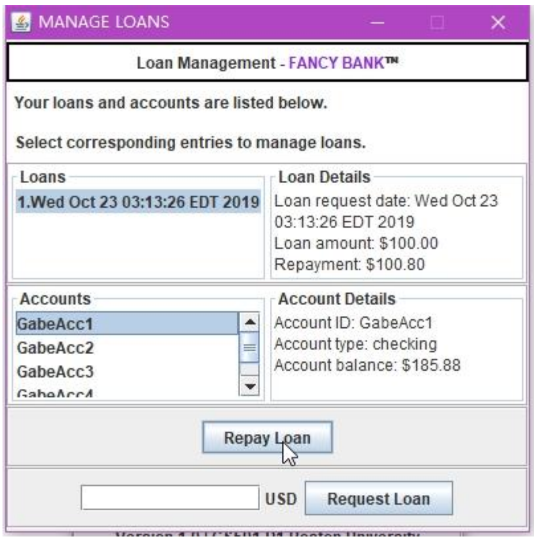
Your loans and accounts are listed below.

Select corresponding entries to manage loans.

GabeAcc1
GabeAcc2
GabeAcc3

Account ID: GabeAcc1
Account type: checking
Account balance: $85.88

100 USD



Your loans and accounts are listed below.

Select corresponding entries to manage loans.

1.Wed Oct 23 03:13:26 EDT 2019

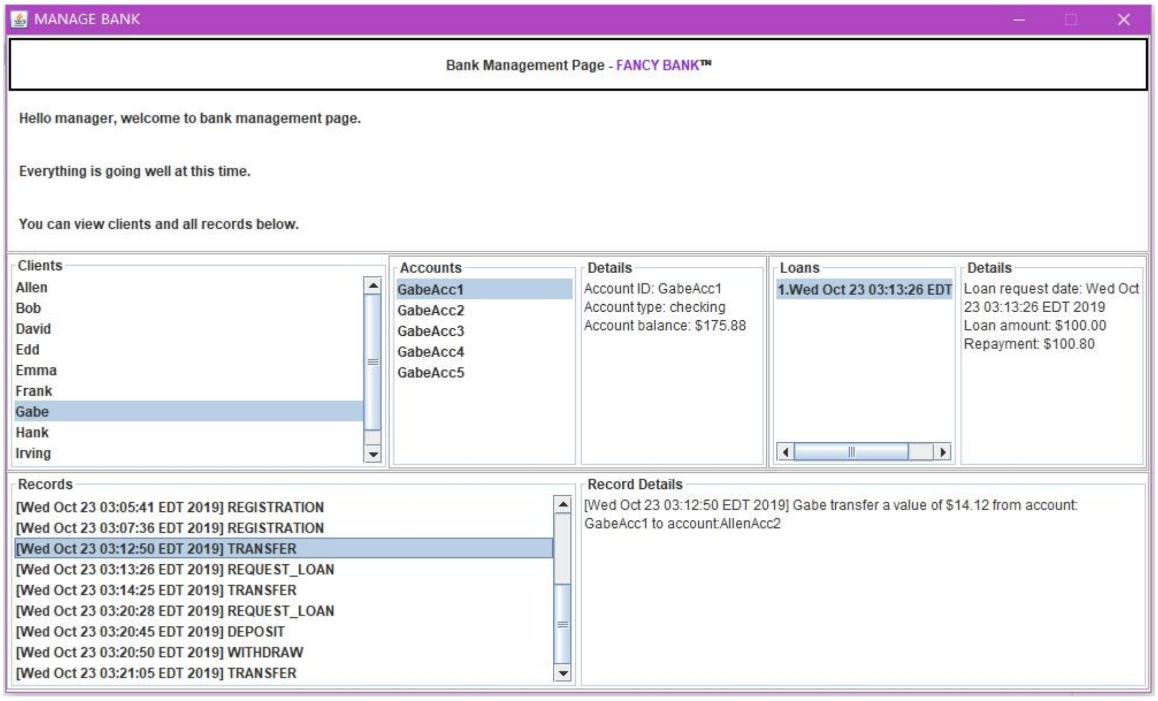
Loan request date: Wed Oct 23 03:13:26 EDT 2019
Loan amount: $100.00
Repayment: $100.80

GabeAcc1
GabeAcc2
GabeAcc3

Account ID: GabeAcc1
Account type: checking
Account balance: $185.88

Request a loan: click manage loan button on the ATM GUI to open a new loan management window. Select your account, enter amount, then click request loan button.

Repay a loan: similar as above, this time select a loan, then select one of your account, then click repay loan button.



Hello manager, welcome to bank management page.

Everything is going well at this time.

You can view clients and all records below.

Allen
Bob
David
Edd
Emma
Frank
Gabe
Hank
Irving

GabeAcc1
GabeAcc2
GabeAcc3
GabeAcc4
GabeAcc5

Account ID: GabeAcc1
Account type: checking
Account balance: $175.88

1.Wed Oct 23 03:13:26 EDT

Loan request date: Wed Oct 23 03:13:26 EDT 2019
Loan amount: $100.00
Repayment: $100.80

```
[Wed Oct 23 03:05:41 EDT 2019] REGISTRATION
[Wed Oct 23 03:07:36 EDT 2019] REGISTRATION
[Wed Oct 23 03:12:50 EDT 2019] TRANSFER
[Wed Oct 23 03:13:26 EDT 2019] REQUEST_LOAN
[Wed Oct 23 03:14:25 EDT 2019] TRANSFER
[Wed Oct 23 03:20:28 EDT 2019] REQUEST_LOAN
[Wed Oct 23 03:20:45 EDT 2019] DEPOSIT
[Wed Oct 23 03:20:50 EDT 2019] WITHDRAW
[Wed Oct 23 03:21:05 EDT 2019] TRANSFER
```

[Wed Oct 23 03:12:50 EDT 2019] Gabe transfer a value of $14.12 from account: GabeAcc1 to account:AllenAcc2

Manager: open Manager GUI, just click any entry you want to view (a client, a client's account, a client's loan, a transaction record…), the details will be printed in the corresponding area.

Other operations like register, deposit/withdraw and add account are similar with operations above.