Corey Parks – Final Project Proposal

CS 240

11/10/2016

Project Summary:

* **Proposed Project Title: C++ Finance Database**
* **Longer Description of project:**
  + My project will be a command line driven bank account database. The program will allow users to create an “account” with the bank databse.
  + Username and Password: The user will create a username and password that will be stored. When the user logs into the system, it will prompt them for this information. Unless the information is correct, it will not let them into the system.
  + Financial Management: Once logged in, the user will be able to deposit, withdrawl, and transfer funds.
  + Funds Transfer: If the user is attempting a funds transfer, it will require the username and password of both accounts.
* **Intended User:**
  + The intended user for this project is anyone who would like a bank account.
* **What problem is project trying to solve:**
  + The project is trying to streamline a bank account system for any given person.
* **What technologies will you need:** 
  + The project will be written in C++, and will require use of the sqlite library. I may not need to use a database program, a simple text file may do the job.

**Use Case Analysis:**

Here is a rough transcript of what the user interaction will look like:

Welcome to SC240 Banking:

What would you like to do?

1.Login

2.Create an Account

3. Quit

> 1

Username: >

Password: >

//If username and password are incorrect or do not exist, give an error message.

Username or password is incorrect.

//If username and password are correct, it logs them in to their account.

Welcome back! You have a current balance of $xxx.xx

Please choose from the following:

1. Deposit
2. Withdrawl
3. Transfer Funds
4. Logout

>1

Please insert funds (type integer value)

>2

Please remove funds(Type integer value)

//If the user tries to take more than available, give an error message.

>3

Please choose from the following:

1. Transfer from account
2. Transfer to account
3. Return to Previous Menu

Note: In order to transfer funds, username and password of second account is required.

>1.

Account 2 Username:

Account 2 Password:

How much will you be transferring out?

//If the user tries to transfer more than available, give and error message. If the password and username are incorrect, create an error message.

>2.

Account 2 Username:

Account 2 Password:

How much will you be transferring in?

//If the user tries to transfer in more than is available, print an error message. If the password and username are incorrect, give an error message.

//We are now returning to the first screen and creating a new account.

Please enter a username:

Please enter a password:

Thank you! Please login with you information to access your account. //Returns to the login screen.

//If the username already exists, give an error message.

**Data Design:**

What data is the program really about?

* **The data the program is dealing with are floating point (for account balanaces) and strings for usernames and passwords.**

What is the best way to represent the data?

* **Could I get a little more information on this question? Are we being asked for Data representation in terms on Computer Memory?**

Will the data need to be persistent? How will you make that happen?

* **Yes, the data will need to be persistent in between runs. As of right now, all of the data will be saved in an external database file that will act as a “server” for the information.**

Will the data need to be aggregated into a larger structure? How?

* **Aggregation models are like a “has-a” relationship. (A department has teachers, the car has an engine). My program will need aggregation. The database has accounts. We can add parts as member variables, either in the form of references or pointers. I will need to implement the simplest relationship type that meets the needs of the program, not what seems right in real life. More info here:** [**http://www.learncpp.com/cpp-tutorial/103-aggregation/**](http://www.learncpp.com/cpp-tutorial/103-aggregation/)

**UI Design:**

Since my program will be command line driven, I think that the example above in Use-Case Analysis can serve for this part.

**Algorithm:**

Define Date Members – what are the key data elements of the class?

**The data members I will need for this program will be the “accounts” This will be user to create usernames and passwords. I will also need a way to retrieve this information.**

Describe the initializer – what will create and populate the data members. Will you read in parameters? Have default values?

**I think I will need to read in parameters. For example, when the program runs, and the class is insatiated, it will prompt the user for information. If they are creating a new account, the username and password will be passed as a parameter/argument to be stored in the database. No default values.**

Define access methods for all data members. Build appropriate getters and setters.

**Could use some help here. I could set the information to null, and just use the getters as whatever the user input is for name and password?**

Identify any methods your class will need beyond access modifiers

**We will need to deal with the bank accounts – withdrawing, depositing, and transferring.**

**WITHDRAW:**

**/\***

***Find the selected account’s balance information in the database/data file***

***Print: How much do you want to take out?***

***Input: Floating point input***

***Error Handling: Make sure it is a float***

***If it isn’t: Print: What’s your problem?***

***Error Handling: If there isn’t enough***

***Print: Error***

***Print: You withdrew X amount.***

***\*/***

**DEPOSIT:**

***/\****

***Find the selected account’s balance information in the database/data file***

***Print: how much deposit?***

***Input: Floating point***

***Error Handling: Make sure it is a float***

***If it isn’t: Print: Wat?***

***Print: Your deposit of X is completed***

***\*/***

**TRANSFER IN:**

**/\***

***Print: What is the other account username?***

***Input:***

***Code: Find the username in the database file***

***If it doesn’t exist, error message***

***Print: What is the password?***

***Input: Password***

***Find the joining password to the username***

***If it is wrong error message.***

***Print: How much do you want to transfer in?***

***Input: Floating point***

***If it isn’t floating point or is more than the account has: error message***

***Print: Your transfer of X is complete.***

***\*/***

**TRANSFER OUT:**

**/\***

***Print: What is the other account username?***

***Input:***

***Code: Find the username in the database file***

***If it doesn’t exist, error message***

***Print: What is the password?***

***Input: Password***

***Find the joining password to the username***

***If it is wrong error message.***

***Print: How much do you want to transfer out?***

***Input: Floating point***

***If it isn’t floating point or is more than the account has: error message***

***Print: Your transfer of X is complete.***

\*/

**We will need some sort of string comparison to check the username and password is correct.**

**Similar to above, I need to find a way to check to see if a username is already taken.**

**We will need to scan the database of accounts information (usernames, passwords, and account information)**