Matthew King, Joe Canero, Gary Patricelli, Dan Thielke

Dr. Monisha Pulimood

CSC260-02

13 November 2012

**Collaborative Project Stage III**

Elaboration of Design

**Part 1: Design Class Diagram**

**Part 2: System Sequence Diagrams**

New User Creates New User Profile



Existing User Adds an Account from System Ready State



Existing User Selects the Option to Print All Data



Remove User Profile



User Searches for a Specific Account



User Decides to Remove an Account



**Part 3: Detailed System State Charts**

New User Creates a User Profile



Existing User Logs into System Ready State



User Decides to Print All Data



Existing User Decides to Remove an Account



Existing User Decides to Add a New Account



Existing User Removes User Profile (All User Data)



**Part 4: User Interface Design**

Our user interface design will be completely text based implemented via CLI and will utilize all of the eight golden rules for designing interfaces. Due to CLI limitations, we must prompt the user for each individual input for each action. After the user logs into the system ready state, either via log in or profile creation, the command line will present several functionality options, each with a corresponding input token. In order to access a particular functionality, the user must enter its input token, which will be of the char datatype. Each submenu will list a back button to access the previous menu, thereby undoing the most recent action by the user.

1. Strive for consistency
   1. The interface will appear the same for every use. There will be a list of possible functions available to the user on startup. For each version change, we intend for the user interface to remain the same to not confuse users.
2. Enable frequent users to use shortcuts
   1. The program will just have the shortcuts designated to access all of the functions. Additional advanced shortcuts will not be available.
3. Offer informative feedback
   1. The program will indicate error messages when things go wrong. It will explain what is acceptable input and how to operate the program. After completing a task, the program will demonstrate that the task has been completed.
4. Design dialogs to yield closure
   1. When a task is completed, the program will say that the task is completed. For example, if a user adds an account, the program will display a message saying that the account has been added.
5. Offer simple error handling
   1. The program will check for all invalid input and make sure the user knows what valid input includes. It will allow users to re-enter data when an error occurs or otherwise when necessary.
6. Permit easy reversal of actions
   1. The program will not exactly have reversal of actions, however it will double check to confirm that the user really wants to perform an action.
7. Support internal locus of control
   1. Every action taken by the program will be directly under the control of the user through the use of text-based commands. The program outputs verification of the user’s actions.
8. Reduce short-term memory load
   1. The program accomplishes this by listing all of the commands available and the hotkey required to use them. It will explain how to enter data correctly and will also display a list of accounts that the user owns.