CS1132 Data Structures - Final Project  
“The Social Network”

Muaz Rahman

Professor Dr. Anasse Bari  
December 8, 2013

**Table of Contents**

1. Project Abstract
2. Introduction and Motivation
3. Requirement Analysis
4. Design
5. Implementation Details
6. Project Timeline
7. Conclusion

**I. Project Abstract**

The idea behind the creation of the “social network” was to be able to successfully design and implement an interface that will mimic a small Facebook prototype in which the user will be given a variety of tools and abilities that comes with any social networking application. Our project incorporates a plethora of different capabilities including a login screen, being able to manage people and friends, sending personal messages, as well as managing a personal profile to name a few. Linked Lists was the main data structure that we used for this project, along with various other smaller structures such as arrays, pointers, memory allocation, and file input/output among others. As we further go through this report, you will be shown in detail the various capabilities and methodologies and structures that work behind the scenes to empower this application to run successfully and smoothly.

Lets begin.

**II. Introduction and Motivation**

The main motivation behind creating this mini “social network” prototype came from one of our in-class assignments in which we had to implement linked lists to be able to add, delete, and view friends and things of that nature. After successfully completing that assignment, we wanted to create something of that nature but more on a grand-scale in which the user will be allowed a variety of tools and applications that they may utilize. We also wanted this social network application to be as similar as possible in terms of design and aesthetics to the various social networking sites we see in our every day online use.

**III. Requirement Analysis**

In terms of requirements of this project, our professor did not restrict us to any one idea or any one data structure. Most of the idea brainstorming came from our groups solely. We submitted a proposal before beginning this project, and the requirements we came up with are the following. Note that these requirements do not completely resemble what our final project came out to be. A basic outline is what it’s more or less meant to be. Take a look.

Basic Functionalities:

* Add Personal Profile (categories: name, birthday, email, occupation, school, hobbies, etc.)
* Add Friends.
* Search for Friends.
* Delete Friends.
* View All Friends.
* Export Friends and Personal Profile from File.
* Import Friends and Personal Profile from File.
* Import “People” List (non-friends) into application.

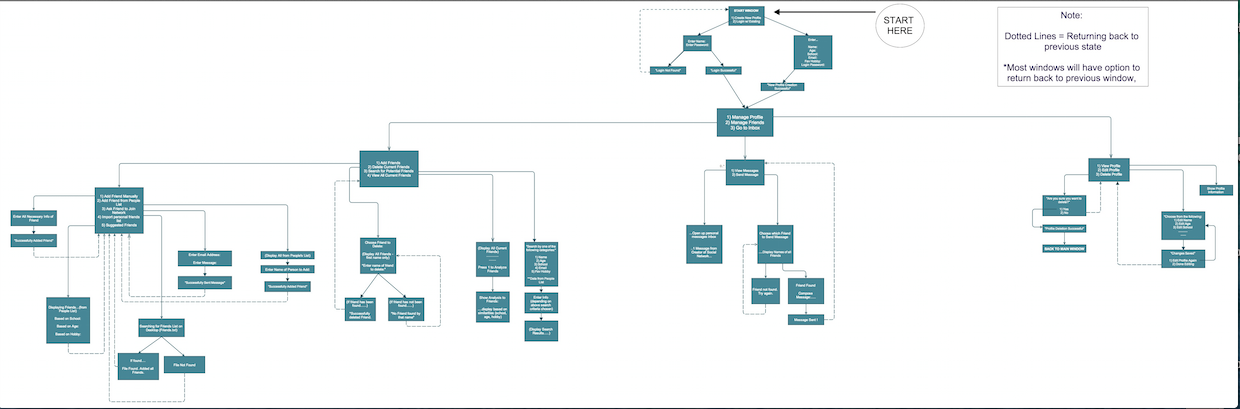
Additional Functionalities:

* Analyze personal profile to suggest to user:
  + What additional info is missing/they may want to add to their profile.
  + Suggest to them friends (from “people” list that was imported into program based on similarities)
* Analyze friends/peoples list to depict to user various forms of data:
  + Friends/people with same birthday day/month
  + Friends/people with same hobbies and/or occupation and/or school and/or etc.
* Personal Messaging Inbox System
  + User can “message” their friends.
  + All messages will be stored under that person/friends personal inbox that can be exported into a file.
  + User can also check their inbox
* Set-up user login system
  + User can choose which person to log in as (depending on the “people” list that can be imported into the application) – user needs correct password to log in.
  + If user cannot find profile on list, then user can create new profile with unique password.

**IV. Design**

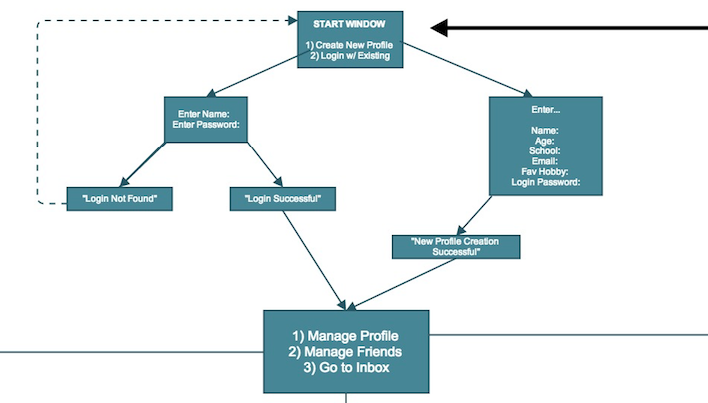
This section will go over the overall design of our projects using graphs and diagrams.

Overall Diagram (to show the extent of the system):

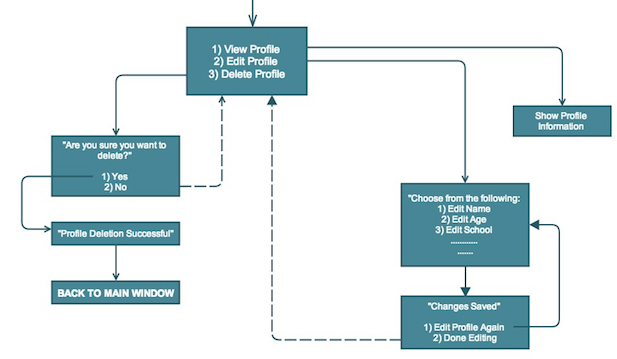


We will go more in detail of the above diagram in the following pages. Look below.

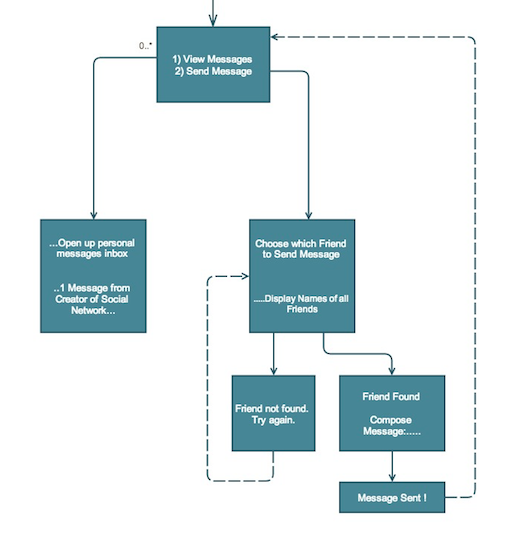
User Login System:



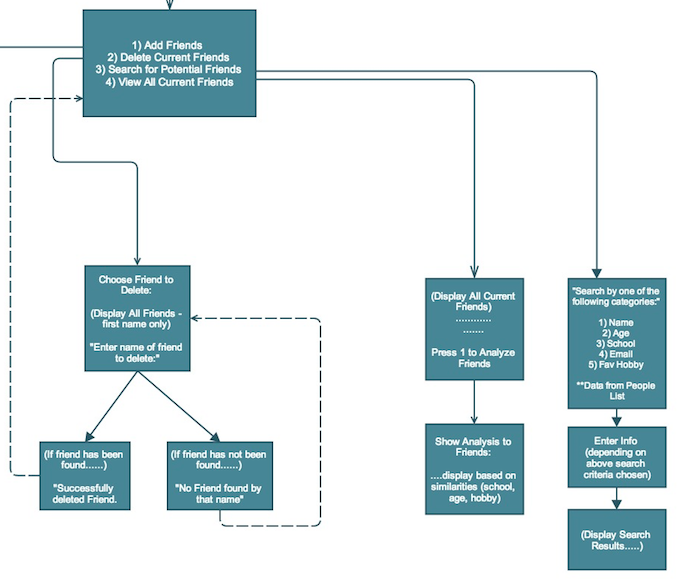
Manage Profile System:



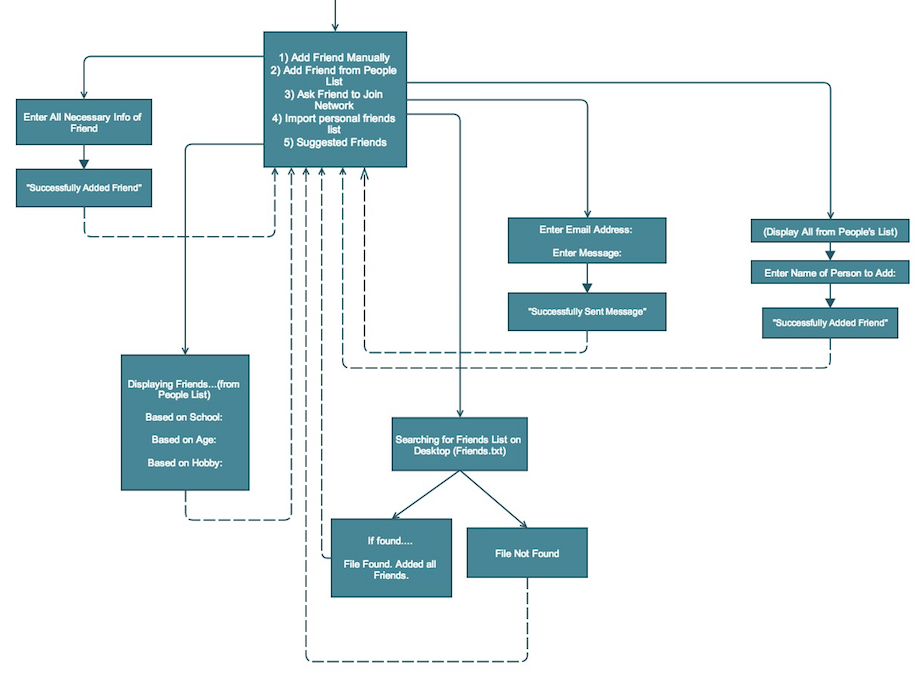
Manage Inbox System:



Manage Friends/People System:



Manage Friends/People System Cont’d:



**V. Implementation Details**

User Login System:

To create this system, we basically set up a text file input/output regeneration system that calls upon this text file every time the user decides to login. The login will take in a username and password argument and pass it on to the check Login function that will subsequently cross reference that with the text file and if it matches then the user will be allowed to login. If unsuccessful, the program will generate an unsuccessful message to the user and he or she will be redirected to the beginning state of the program. The user can from here on decide to go ahead and create an entirely new profile in which they will be asked to create a username and password that they can use to login to the system anytime from that point on.

Personal Profile Management System:

The manage profile option allows the user to first create a profile with their personal information stored in a struct. The information given by the user is also stored in a text file. From here there the user is given three options: view, edit, and delete. When viewing, we simply uses fscanf to scan the text file for the listed information and then prints it. When editing the profile, the user can pick which aspect of their profile they would like to change. These change or changes will transfer over to the textfile. For example, if the user wants to edit their name, then in order to login as an existing user, they will have to use the new name, however the rest of their information that wasn’t edited will still stay the same. If the user wants to delete their profile, all their information within the text file will be erased. This also means they will no longer be in the system which means they won't have access to their friends list.

Personal Messaging Management System:

The personal messaging system is implemented using a text file. The user has two options, viewing the messages and sending the messages. If the user choses to view a message, he or she will be able to see a predefined message from the creators of the network, welcoming the user. Once the message has been read, a for loop ran in the background will mark the message as “Read,” so that when the user comes back to the “View Messages” section, it will tell the user that he/she have “0 Unread Messages.” If the user chooses to send out a message, a list of all friends will be displayed, where the user is able to choose who the message will be sent to. Once a friend is chosen, the message input is stored in a text file under the specified friends name.

Manage Friends System:

The friends of a specific user is all stored in a linked list. The user can add, delete, and view all their friends. They can even analyze their friends to see what similarities are in between them. They can also import as well as export their friends if they choose to. The user may also add friends from the people list which will be discussed later. It is essentially a list of people that the user may decide to add if they would like to, to their friends list. The user can also decide to “email” a friend and send them a custom message to be able to ask them to join the network, if they so choose to.

Manage People System:

The manage people system in this project was implemented because we essentially wanted to allow the user to generate a list of people for them from which they can also add as friends. At first all these people and their entire profiles are not shown, simply because the user has not added them as friends. But once they are added, the user can directly view their profile. The user can also search through these friends in a manner in which they can directly view all the names of the peoples that correspond to this certain search criteria.

**VI. Project Timeline**

The timeline was relatively short and to the point. View below:

Week1: Generate Idea, Brainstorm.

Week2: Create Outline, Submit Proposal.

Week3: Begin Programming.

Week4: Continue Programming.

Week5: Debug Program.

Week6: Debug/Test/Finalize Program.

**VII. Conclusion**

This project helped to solidify concepts learned throughout the semester. Group members were able to utilize their knowledge by building separate modules, uniting them at the end to make a complete system. Some of the challenges faced included reading/writing files effectively, data mining and data manipulation, algorithms for various analysis methods, and the flow of program maintaining the unity between the three group members and their codes. The group was able to successfully debug all the issues faced, and produce a working prototype. In the near future the group would like to improve upon the inbox messaging features, being able to connect to the internet, along with being able to implement the network on a website incorporating graphic design.