Group G Design Document

Due: 9/15/13

By: Anthony McBride, Amber Maynard, and William Suter

CMSC 495, Current Trends and Projects in Computer Science

1. **Event-trace diagram:**
2. Program Starts. The program starts and creates a directory for the database is if one does not exists.



1. The user adds a person profile. This creates a new directory in the database for a specific person.



1. The user adds an event to the application. A specific person is selected, and then the user is prompted to enter information (date, event details) to be added to the database.



1. The user edits or deletes an event within the application. The user selects a specific person, and then the user selects to edit or delete an event associated with the selected person.



1. The user adds an item to a specific person’s wish list. The user selects a person, and then adds an item to the user’s wish list.



1. The user displays the wish list or the special events associated with a specific person.



1. The program stops.



1. Error Scenarios. The following error scenarios exist:
2. There are no people, so there are no special events to be displayed.
3. A person has no special events or wish list information associated with them, so there is no information to display, edit, or delete.

All error scenarios check the database for any information, and if none exists then an error message is displayed in the GUI.



1. **Class Design**
2. Input System:

This is the main class which initializes the main GUI. It represents the major hub that the user will interact with.

Class UserGui(){

JButton edit or delete event;

JButton add person;

JButton add event;

JButton add to wish list;

JButton display wish list;

JButton display events;

JComboBox names;

JFrame main frame;

Void main(){

Create database (if it does not already exist)

Initialize GUI.  
 }

Void create database(){

This will create the database if it does not already exist.

}

Void initialize GUI(){

This will initialize the user GUI, creating a main GUI hub for the

User to interact with. Some of the features are:

1. Add Person
2. Add Event
3. Add Item to Wish List
4. Edit/Delete event
5. Display Event
6. Display wish list
7. Select a loved one from a combo box.

}

Void Action Performed (Action){

Depending on the action selected by the user, the program will

Perform one of the following actions:

1. Display the events for a person in a text area.
2. Display the wish list for the selected person in the text area.
3. Launch a new window that will allow the user to edit an event.
4. Launch a new window that will allow the user to add an event.
5. Launch a new window that will allow the user to add a person.
6. Launch a new window that will allow the user to add an item to a specific persons wish list.

}

}

1. Add person subsystem:

This allows the user to add a loved one to the database. Each person added has their own special events and a wish list associated with them.

Class add person(){

JButton finish;

JButton cancel;

JTextArea name;

Void initialize(){

This method initializes the add person GUI.

}

Void action performed(Action){

Two options are available:

1. Cancel and close window.
2. Finish, and take the new information from the text area and add it to the database.

}

}

1. Add event subsystem:

This class allows the user to add an event to the database.

Class add event{

String name, file name;

JTextField event;

JTextField date;

JButton finish;

JButton cancel;

Void initialize(String name){

This class initializes the GUI window for adding an event. It also

Creates a string that will be used to reference the text file in the

Database that will be appended.

}

Void action performed(Action){

The user will have the option of adding the event to the database,

Or cancelling and closing the window.

}

}

d: Add wish list subsystem:

This sub system allows the user to add an item to a specific person’s wish list.

Class Add to wish list (){

String name, file name;

JTextField item;

JButton finish;

JButton cancel;

Void initialize(String name){

This class initializes the GUI window for adding an event. It also

Creates a string that will be used to reference the text file in the

Database that will be appended.

}

Void action performed(Action){

The user will have the option of adding the event to the database,

Or cancelling and closing the window.

}

}

e: Edit or delete event subsystem:

This sub system allows the user to edit or delete a specific event

Class edit or delete event (){

String name, file name;

JTextField itemt;

JButton finish;

JButton cancel;

Void initialize(String name){

This class initializes the GUI window for adding an event. It also

Creates a string that will be used to reference the text file in the

Database that will be appended.

}

Void action performed(Action){

The user will have the option of deleting or editing an event,

As well as cancelling the operation.

}

}

f: Unresolved risks and mitigation: There are many risks associated with this project, as well as strategies that can mitigate those risks. The following table explains the risks and how they may be mitigated.

|  |  |  |
| --- | --- | --- |
| **Problem/Risk** | **Risk Level** | **Mitigation Strategy** |
| Time Constraints | Extremely High | Every member to provide information |
| Response | Extremely High | Request information from all members. |
| Client configuration | Extremely High | Set minimums when implementing the program to ensure users will be more likely to be able to run the program. |