Software Design Specification

Created by: Rachel Tidwell
Jeffrey Heredia
Andy Thorsen
Michael Thorsen

Introduction

1.0 Introduction

The following described product is Darkspear. This SDS describes the entire design specifications for the project.

1.1 Goals and objectives

The overall purpose of Darkspear Clan finder is to organize and unite gamers of similar skill and abilities.

1.2 Statement of scope

The software specified is operated inside a GUI that allows a user to join clans based on video game preferences. The application also allows users to chat in a chat box. The application is useful for gamers (entertainment) as it will help to unify gamers based on interest and skill level.

1.3 Software context

Darkspear is intended to be used by people to find other people playing the same video game and to form groups, either for competitive or more casual playstyles. It can also be used to chat with other members of clans a person is a member of.

1.4 Major constraints

The largest factor we are considering is the time we have to complete the project.
Making sure developers don't overwrite changes that other developers make. There will be no shared IDE and we need to make sure only one developer is writing to a file at a time.

Data design

2.0 Data design

All data used will be stored in a Mysql database. This database will have approximately X tables. There exists several temporary data structures for username and password and globals for session data.

2.1 Internal software data structure

Data packets are exchanged through pythons connect function directly to the mysql database.
Data structures that are passed among components of the software are described.

2.2 Global data structure

A parent controller class which controls which page is being displayed is used globally by passing it into its children (the individual pages). In addition, an additional class designed to fetch and house the user's data will be readily available to all pages through get functions.

2.3 Temporary data structure

Each page, such as the login page and clan pages, are temporary.

2.4 Database description

player(USERNAME <string>, email <string>, password <string>, comp_player <binary>, bio <string>, fav_clan1 <string>, fav_clan2 <string>, fav_clan3 <string>, fav_clan4 <string>, fav_clan5 <string>)
- Foreign key fav_clan1 from clan(CLAN_NAME)
- Foreign key fav_clan2 from clan(CLAN_NAME)
- Foreign key fav_clan3 from clan(CLAN_NAME)
- Foreign key fav_clan4 from clan(CLAN_NAME)
- Foreign key fav_clan5 from clan(CLAN_NAME)

player_picture(USERNAME <string>, picture <link>)
- Foreign key USERNAME from player(USERNAME)

game(GAME_NAME <string>, genre <string>, max_team_size <int>, rating <string>, has_comp <binary>, is_pve <binary>, is_pvp <binary>)

game_picture(GAME_NAME <string>, picture <link>)
- Foreign key GAME_NAME from game(GAME_NAME)

game_platform(GAME_NAME <string>, PLATFORM_NAME <string>)
- Foreign key GAME_NAME from game(GAME_NAME)

clan(CLAN_NAME <string>, leader_username <string>, member_count <int>, max_members <int>, bio <string>)
- Foreign key leader_username from player(USERNAME)

clan_picture(CLAN_NAME <string>, picture <link>)
- Foreign key CLAN_NAME from clan(CLAN_NAME)

clan_officer(CLAN_NAME <string>, USERNAME <string>, rank <string>)
- Foreign key USERNAME from player(USERNAME)
- Foreign key CLAN_NAME from clan(CLAN_NAME)

player_in_clan(USERNAME <string>, CLAN_NAME <string>)
- Foreign key USERNAME from player(USERNAME)
- Foreign key CLAN_NAME from clan(CLAN_NAME)

---

plays_game(USERNAME <string>, GAME_NAME <string>)
Foreign key USERNAME from player(USERNAME)
Foreign key GAME_NAME from game(GAME_NAME)

clan_games(CLAN_NAME <string>, GAME_NAME <string>, rank <string>)
Foreign key CLAN_NAME from clan(CLAN_NAME)
Foreign key GAME_NAME from game(GAME_NAME)

f_Chat(F_chat_ID <int>, p1_username <string>, p2_username <string>, message <string>, time <date>)
Foreign key p1_username from player(USERNAME)
Foreign key p2_username from player(USERNAME)

c_Chat(C_chat_ID <int>, poster_username <string>, clan_name <string>, message <string>, time <date>, board <string>)
Foreign key clan_name from clan(CLAN_NAME)
Foreign key poster_username from player(USERNAME)

clan_event(EVENT_ID <int>, CLAN_NAME <string>, event_name <string>, description <string>, date <date>)
Foreign key CLAN_NAME from clan(CLAN_NAME)

clan _announcement(ANNOUNCEMENT_ID <int>, clan_name <string>, announcement <string>, date <date>)
Foreign key clan_name from clan(CLAN_NAME)

Architectural and component-level design

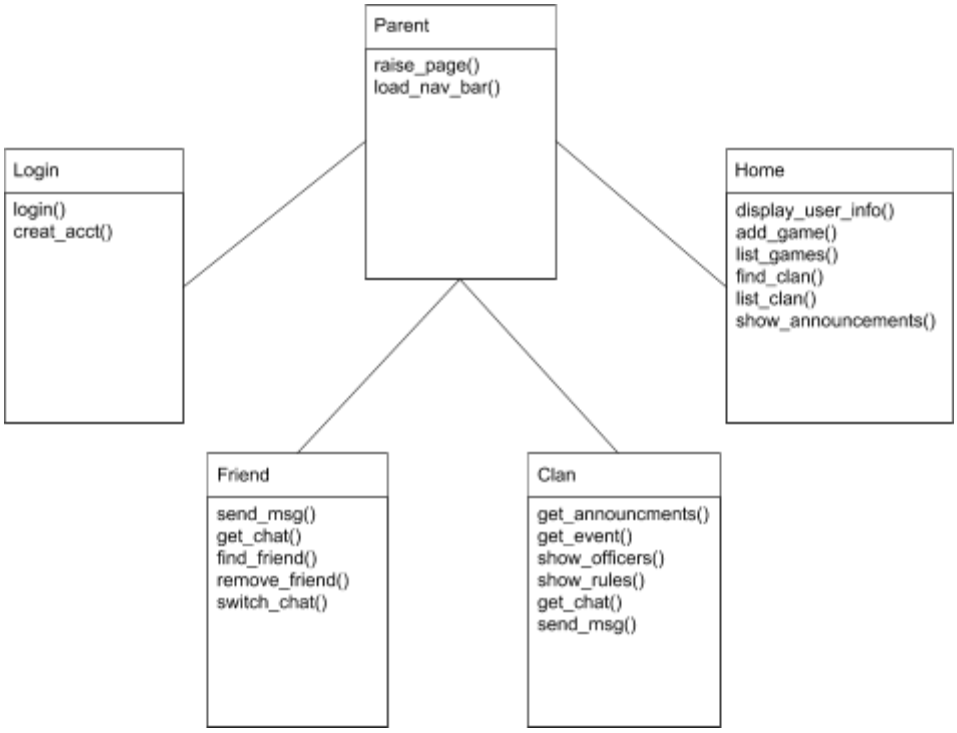
3.0 Architectural and component-level design

Our graphical user interface is designed in Python using Tkinter. Python then interfaces with a Mysql server hosted on a personal raspberry pi.

3.1 System Structure

At launch, a parent class container a controller is original launched. This controller generates all the pages used within the application itself. It then raises the login page to provide an interface for the user. Each time the user attempts to navigate between pages a request is made to the controller to raise the corresponding page to the foreground. Upon successful login a request is made to the mySQL database to gather all of that user's data. This data is then housed in a separate class. Pages that want to reference the user's data will have to make a request to the class and store the results locally.

3.1.1 Architecture diagram



3.2 Description for Components

Parent: A superclass that controls which page is currently being displayed and assists with the transition between pages.

Login: Gathers the user's credentials and then initiates the session for the correct user before redirecting to the home page.

Friend: A page that allows for a user to search and find friends. It also allows the user to send private messages between other users.

Clan: A page for hosting the clan. It presents the user with announcements and and events managed by clan leadership as well as a shared chat among all clan members.

Home: An overview of the user's information. Includes and editable biography section as well as summaries of all clans they are in. As well as a an overview of upcoming events and recent announcements across all of their clans.

3.2.1 Component interface description.

Parent: input: desired page. Output: Contents of that page.

Login: Input: user name and password. Output: Either an error message for incorrect credentials or the home page for the corresponding user.

Friend: Input: Text message Output: the chat box

Clan: Input: text message, announcements, events. Outputs: Clan chat, clan page with appropriate fields.

Home: Input: biography. Output: Personalized home page

3.2.2 Component processing detail

Details are loaded upon login and and when updates are made. All other requests happen in O(1) time

3.2.2.1 Design Class hierarchy for component
The parent class is a superclass which each page is a descendent from this one.

3.2.2.2 Performance issues for component
Due to the limitations of the raspberry pi our server may not be able to handle the amount of input output we need to run Darkspear on multiple computer. Pi might crash frequently.

3.2.2.3 Design constraints for component
We are limited to widgets already implemented inside of tkinter

3.2.2.4 Processing detail for each operation of component
To process our chat feature we need to first load all previous chat messages with a quick select from the database, sorting by time posted. To post messages to a board, or as a personal message, the contents are sent to the database which then updates and pushes the updated version to other users in the clan.

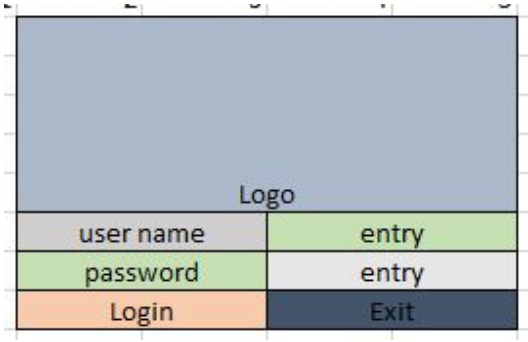
3.3 Dynamic Behavior for Component
The only dynamic aspect of the program is the chat. Everything else is static and simply filled in from information from the database.

3.3.1 Interaction Diagrams
The chat is not implemented yet, will update when chat is functioning.

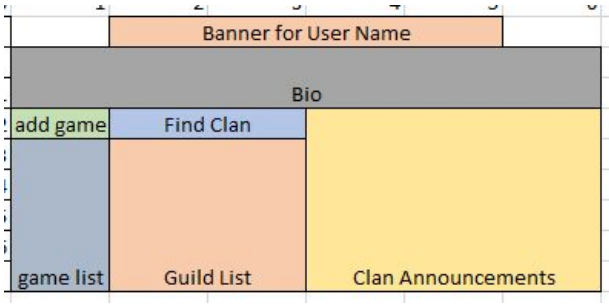
User Interface Design

4.0 User interface design
There are four pages that will be used a login page, a home page, a clan page, and a friends page.
The login page will have our logo in the background with a box at the bottom to enter login information.
The home page will have a banner at the top with your username, under that there will be a section for your bio. Under this there are three sections, the left section is a list of games you play with a button to add more games, the middle section is a list of guilds you are a part of with a button to find another clan, the right most section will show all announcements for the clans you are a part of.
The clan page will have a banner at the top with the clan name. Under that will be announcements for that particular clan. Next to that will be a list of events that clan has set up or is participating in. Under those will be a list of leaders and officers of the clan. Next to that will be a list of rules for the clan. Under all of these will be a chatbox for the clan.
The friends page will have a bar on the side that has a find friend button on the top with recent conversations you can switch to under that, the main part of the page is showing which friends are online with a chat box underneath.

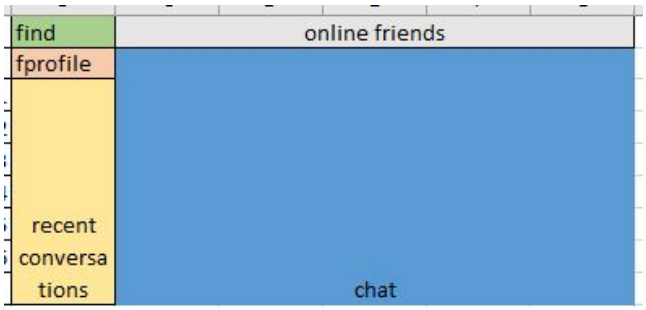
4.1 Description of the user interface
Login page



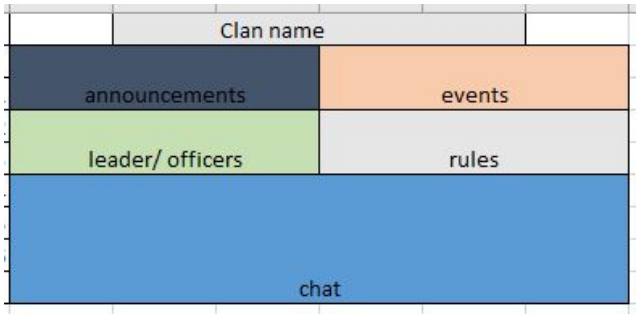
Home page



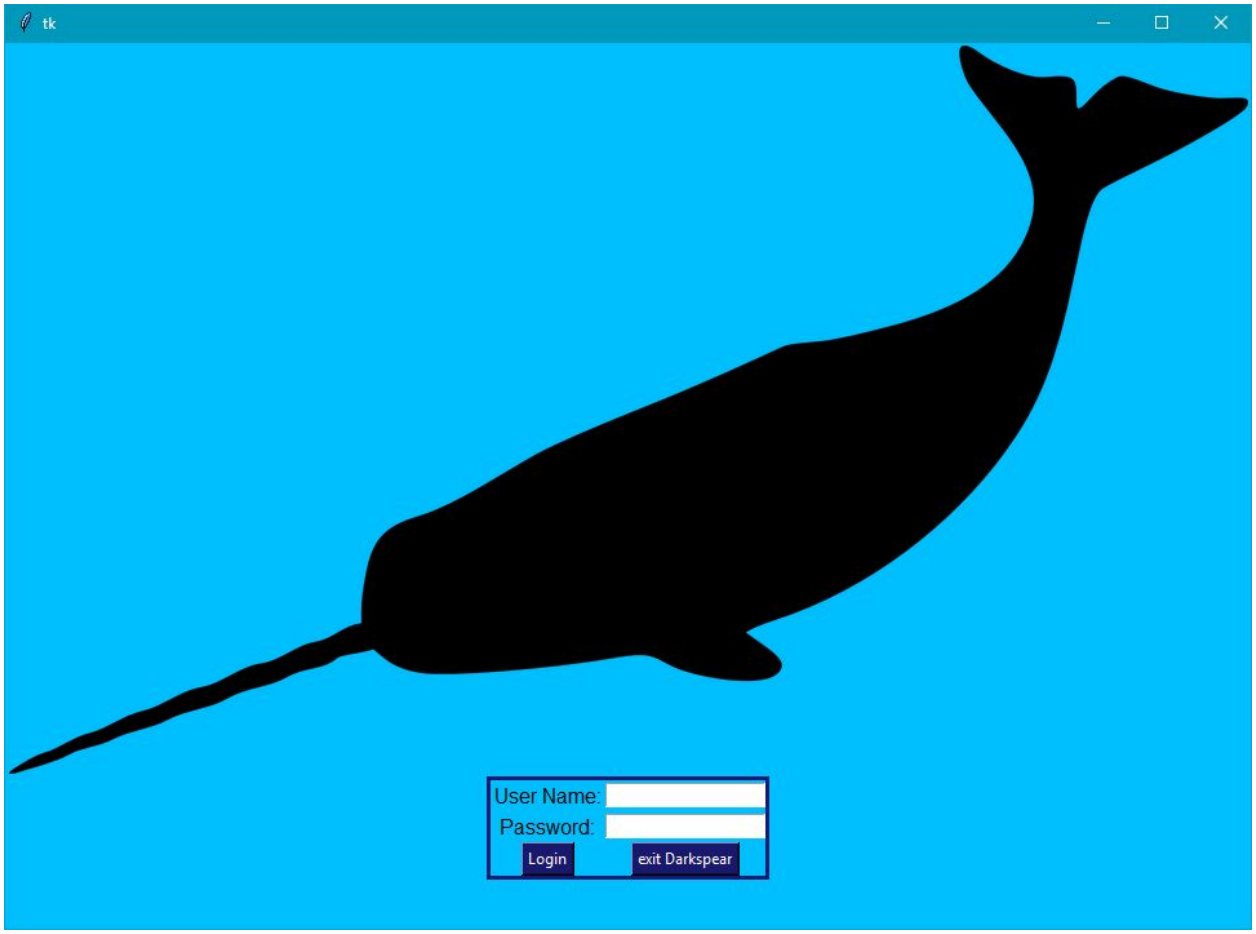
Friend page

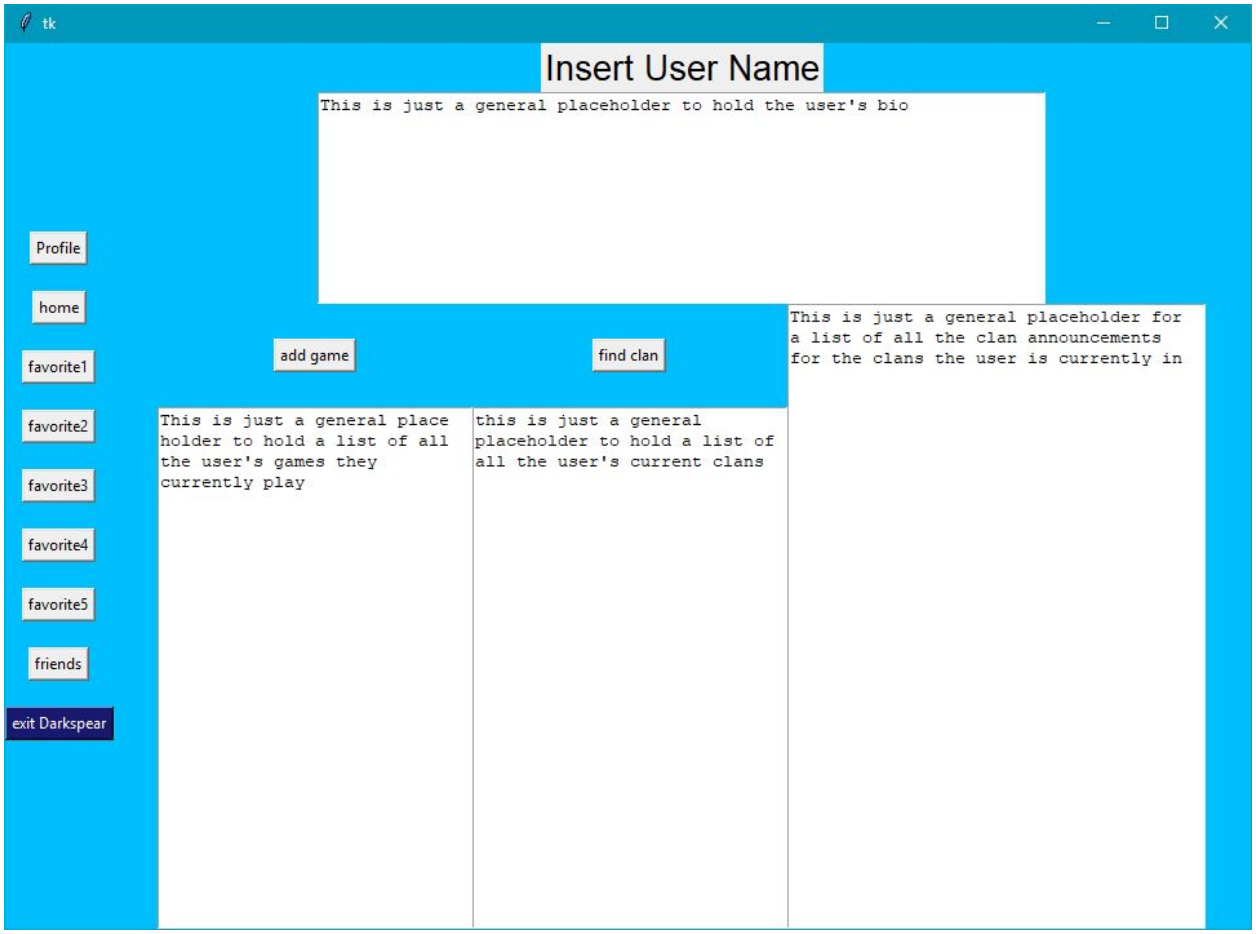


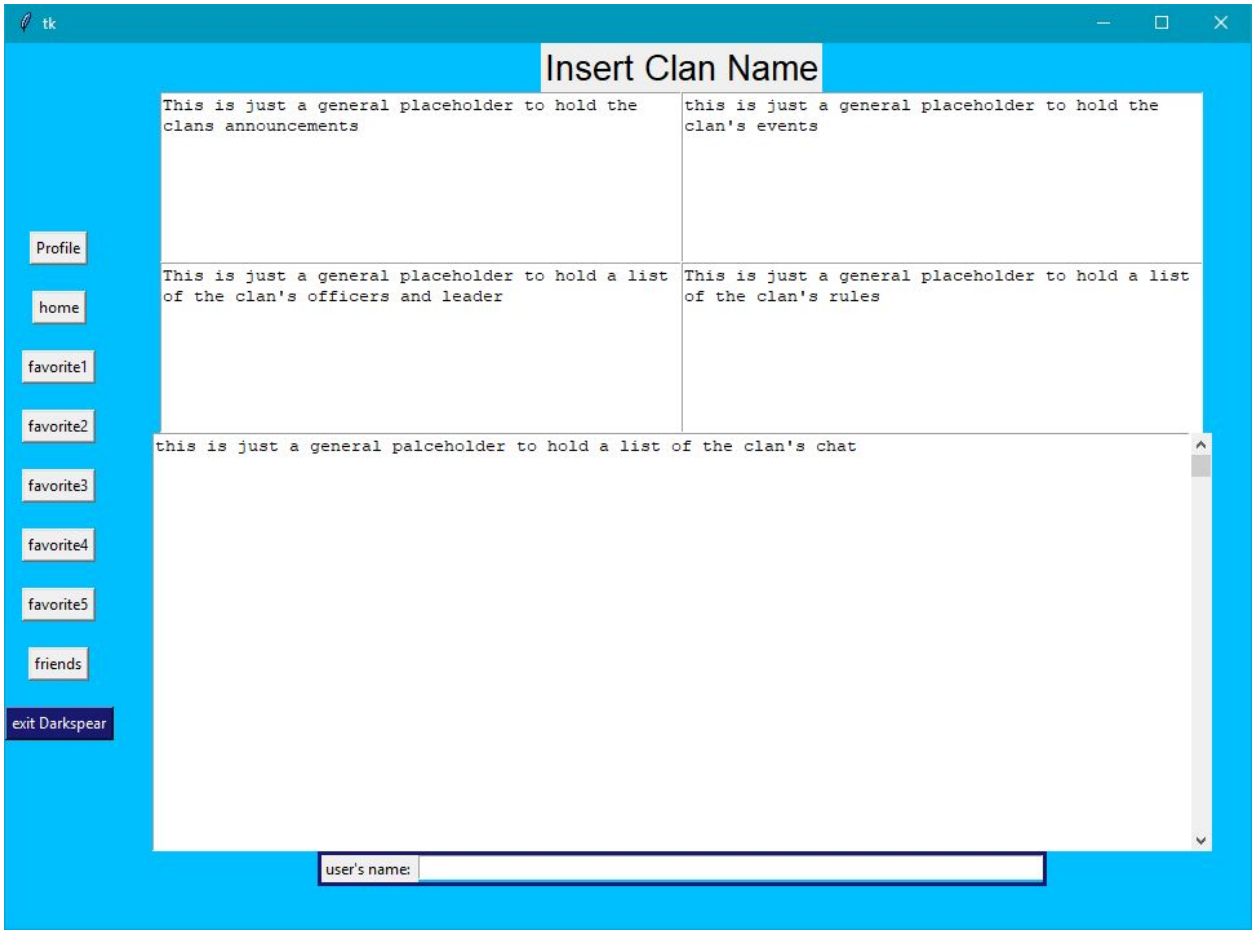
Clan page



4.1.1 Screen images


tk
User Name:
Password:
Login
exit Darkspear


tk
Insert User Name
This is just a general placeholder to hold the user's bio
Profile
home
favorite1
favorite2
favorite3
favorite4
favorite5
friends
exit Darkspear
add game
find clan
This is just a general placeholder for a list of all the clan announcements for the clans the user is currently in
This is just a general place holder to hold a list of all the user's games they currently play
this is just a general placeholder to hold a list of all the user's current clans



4.1.2 Objects and actions

4.1.2.1 Login screen

1. Username Field: a field to enter your username.
2. Password Field: a field to enter your password.
3. Login Button: a button to submit information entered to authenticate login.
4. Exit Button: a button to exit the application.

4.1.2.2 Home screen

1. Add game button: a button to add a game to the list of games you play.
2. Find Clan Button: a button to find a clan to join.

4.1.2.3 Clan page

1. Chat box: a box with messages related to the clan, can be scrolled through.
2. Chat box entry field: a field to add a message to the chat box.

4.1.2.4 Friends page

1. Chat box: a box with messages sent and received by a friend, can be scrolled through.
2. Chat box entry field: a field to add a message to the current chat box.
3. Find button: a button to start the search for a new friend.
4. Fprofile button: a button that lets you see a friends profile.

5. Recent conversations: clicking a name here lets you change to a message box related to the name you clicked on.

4.1.2.5 Home, clan, and friends page shared items

1. Profile button: changes current window to your profile page.
2. Home button: changes your current window to the home page
3. Friends button: changes your current window to your friend page.
4. Favorite buttons: changes your current window to a favorites page.

4.2 Interface design rules

1) Strive for consistency
2) Enable frequent uses to use shortcuts
3) Offer informative feedback
4) Design dialog to yield closure
5) Offer simple error handling
6) Permit easy reversal of action
7) Support internal locus of control
8) Reduce short-term memory load

4.3 Components available

Components for the GUI include a login page, a home page, a clan page for each clan, a friends page, and a profile page for each user..

Restrictions, Limitations and Mistakes

5.0 Restrictions, limitations, and constraints

A restriction of our software is creating the interface for the comments and chat functions. We also need to assure that no files are being overwritten or altered at the same time creating inconsistencies in code.

Testing Issues

6.0 Testing Issues

Test create new user trigger.
Test clan search feature.
Test clan server creation.
Test clan chat feature.
Test joining clans.

6.1 Classes of tests

Component Testing, testing each small component individually before combining the whole thing.
Exploratory Testing, upon completion use the product and note any issues that come up.

GUI Testing, testing that all aspects of the GUI are lined up and orientated correctly.
Monkey testing, testing random inputs from the user.

6.2 Expected software response

A new user is added to the database with correct user_id number.
The correct query is passed to the database and a list of suitable clans are returned.
A new clan is added to the database and an editable page is created to host their activity.
The desired text is correctly placed in the correct text channel.
User is added to the clan member list and granted access to their clan page.

6.3 Performance bounds

Because the software will use local hosting and not be accessible to the outside world, only one user will be able to access it at a time. However, if the app was actually published, the limitations would depend on the hosting service used to host the site.

6.4 Identification of critical components

The clan search feature needs to only return relevant clans, as well as clans that play at the appropriate level of the user (competitive/casual, if competitive what rank?).
The clan chat feature, finding a clan is useful but if they have no point in communicating, then the entire process of finding them is pointless. The text not only needs to be rendered in a timely fashion, but also in the correct channel so that the correct users see it.