**CS22120 Software Development Life Cycle**

**Group 05 Project Plan**

**Author:** All

**Config Ref:** SE\_05\_Interaction

**Date:**

**Version:** 1.4

**Status:** Draft

1. Introduction 3

1.1. Purpose of this Document 3

1.1.1. Scope 3

1.2. Objectives 3

2. Overview of Proposed System 3

2.1. Technology Being Used 3

2.2. Use-cases 4

2.2.1. Use-case for Users 4

2.2.2. Use-case for Server 5

2.3. User Interface Design 5

2.3.1. User Interface Layout Designs 6

2.3.1.1. Menu Bar 6

2.3.1.2. Cash Pile 6

2.3.1.3. Banner 6

2.3.1.4. Create Account 7

2.3.1.5. Login 8

2.3.1.6. Homepage 9

2.3.1.7. Friend's Page 10

2.3.1.8. Battle Requests 11

2.3.1.9. Friend Requests 12

2.3.1.10. Breed Options 13

2.3.1.11. Selling Options 14

2.3.1.12. Battle Report 15

2.5. Risk Assessment Table 16

2.6. Gantt Chart 18

3. References 19

4. Document Change History 19

# 1. Introduction

## 1.1. Purpose of this Document

The purpose of this document is to describe how we plan on creating the web application from the design to the technology we plan to use.

### 1.1.1. Scope

This document specifies the user interface designs and what technology will be used.

## 1.2. Objectives

The main objective of this project plan is to show to the customer how we plan on creating the system. Showing the design of the UI will give the customer an idea of how Monster Mash will look. Describing what technology that will be used will give the customer an insight of the “back end” side of the application.

# 2. Overview of Proposed System

## 2.1. Technology Being Used

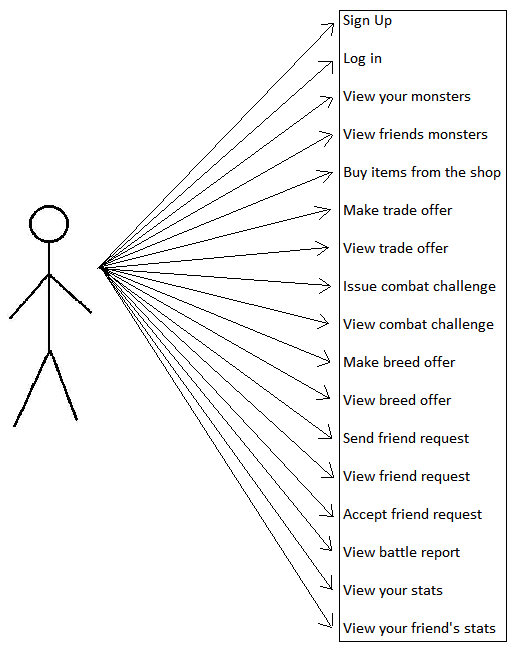
At first we were thinking of using Tomcat server application, because it supports JSP and Java Servlets and is lighter than every other server application. We decided to use Glassfish, as it is being used by all the other groups and is the only software supported by the university. Glassfish provides full Java EE support including JSP and Java Servlets, which we will be using and is easier to navigate through than alternative software.

Glassfish is an open – source application server by Sun Microsystems. It provides full support for Java EE, JavaBeans, JPA, JSF etc. Glassfish has many more administration and monitoring tools than alternative software, such as Tomcat. We will be using Java Servlets to deal with requests from the client on our server. Considering we are striving for an MVC design pattern, Servlets are the right choice for control i.e handling requests.

A Java Servlet is a Java class used to extend the capabilities of the server. Although Servlets can respond to any types of requests, they are commonly used to extend the applications hosted by web servers, so they can be thought of as Java Applets that run on servers instead of in web browsers.

## 2.2. Use-cases

### 2.2.1. Use-case for Users



### 2.2.2. Use-case for Server

## 2.3. User Interface Design

To begin the user interface design, we started with a use case diagram which had everything the user needed. From this we created a directed graph which shows all the options the user has for navigation from that page. After agreeing on the pages we will be creating, we started with a basic hand drawn design of what the game will look like. Once we had produced the hand drawn designs, it was necessary to put them in to a digital format as well as annotated versions. There will be seven pages, which include:

* Create Account
* Login
* Homepage
* Friend's Page
* Battle Requests
* Friend Requests
* Breed Options
* Selling Options
* Battle Report

### 2.3.1. User Interface Layout Designs

#### 2.3.1.1. Menu Bar

Each page (that the user is signed into (not sign up or log in page)) will have a menu bar. The menu bar will appear below the banner and will be on each page they visit. On this bar, the following will be displayed:

* **Home:** Links to the “Home” page.
* **Friend Requests:** Links to the “Friend Requests” page.
* **Breed Options:** Links to the “Breed Options” page.
* **Battle Requests:** Links to the “Battle Requests” page.
* **Selling Options:** Links to the “Selling Options” page.
* **User Name:** This will display the email of the user which is currently logged in.
* **Cash Pile:** This will display the user’s cash pile.
* **Logout:** This will end the user’s session.

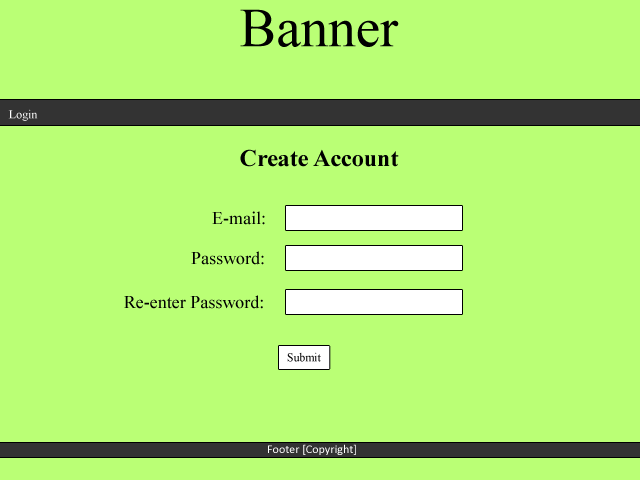
#### 2.3.1.2. Cash Pile

It is one of the requirements for the friends list to be ranked and this can be done by the highest amount of money. Each user will be sorted from richest – poorest on the homepage.

#### 2.3.1.3. Banner

Each page will have a banner which will be a design (text/image) saying 'Monster Mash'.

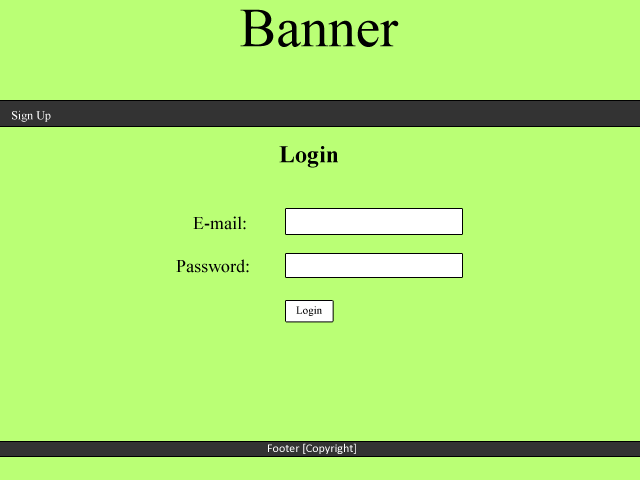
#### 2.3.1.4. Create Account



Once the user has entered their email and passwords, it will then be added to the database once the submit button is clicked.

This will direct the user to the “Login” page, to which they can log in.

#### 2.3.1.5. Login

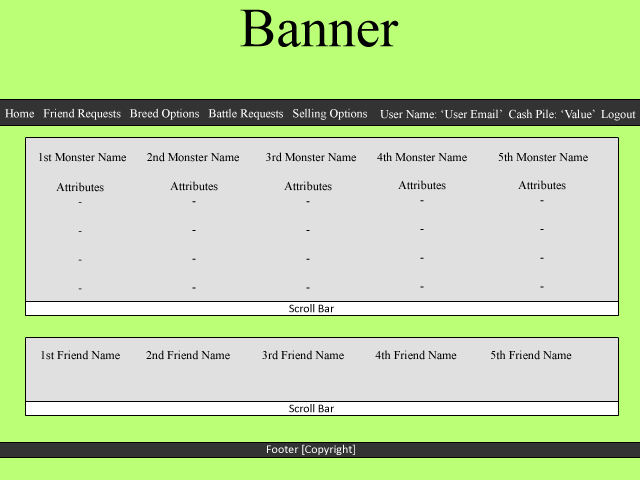


Non-registered users will click this to sign up. This will navigate the user to the “Create Account” page.

If the credentials are correct, they will be directed to their homepage. If the credentials do not match, an error message will appear.

The email and password that the user signed up with will be their credentials to login with. Regular expressions will be used for the email.

#### 2.3.1.6. Homepage

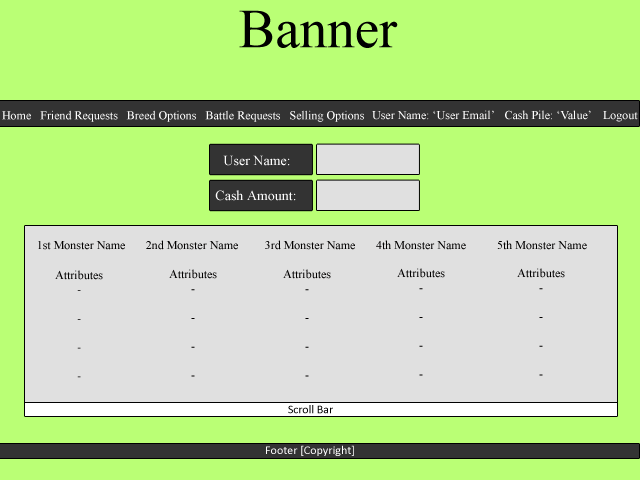


The user’s monsters will appear here and will have the monster’s name and attributes.

The user’s friends will appear here and will be sorted by wealthiest (by “cash pile”) to poorest. The user will be able to click on a friend which will direct them to the selected friend’s page.

For cases where the user has a lot of monsters/friends, a scroll bar is used so they can view the objects that don’t fit on the screen.

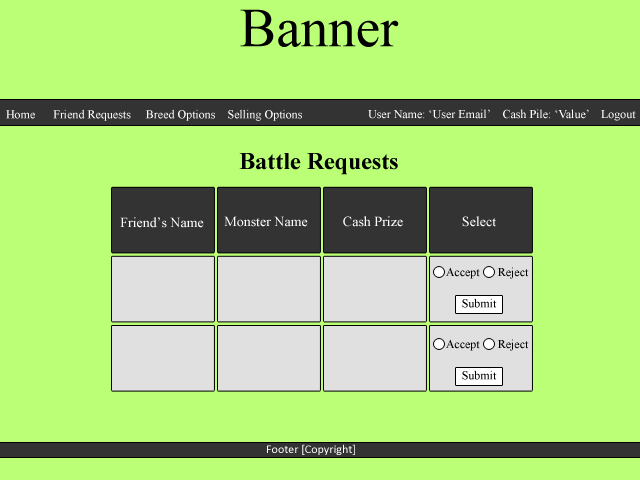
#### 2.3.1.7. Friend's Page



The monster’s belonging to the friend will appear here with the name and attributes of that monster.

The selected friend’s user name will show here as well as their cash amount.

#### 2.3.1.8. Battle Requests



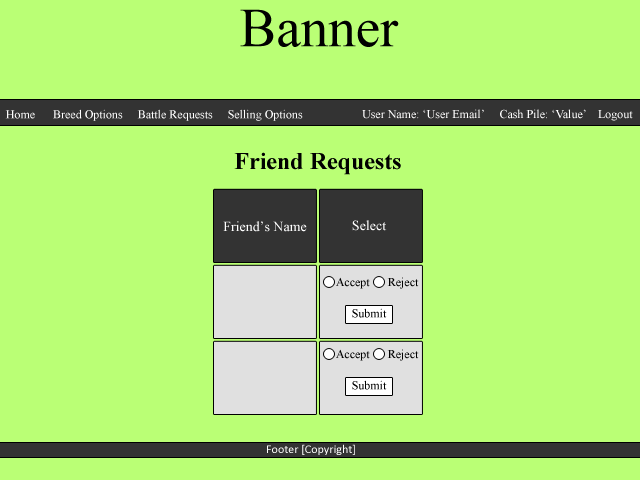
The monster that the friend is requesting to battle with will appear in this column.

The name of the friend requesting for battle will appear in this column.

Each victor will receive prize money; which will appear in this column.

The user will have the option to choose to accept or reject the request.

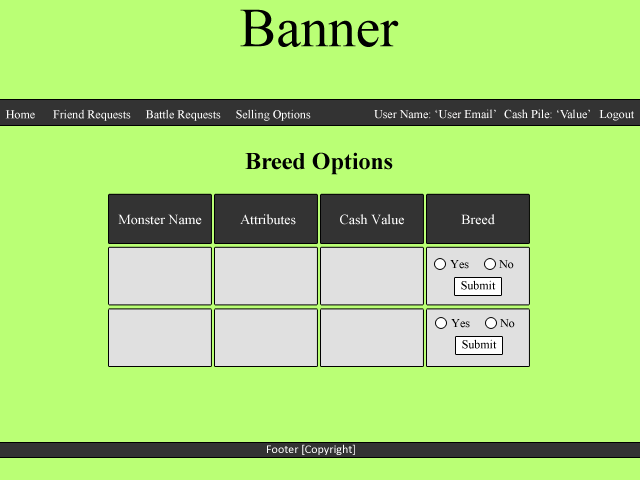
#### 2.3.1.9. Friend Requests



Each request to friend will appear here, with that friend’s name.

The user will be able to accept or reject the request to friend. Accepting will add that friend to their friends list.

#### 2.3.1.10. Breed Options



The user will be able to choose what monster they want to breed with.

The cash value to breed with will appear here. The user will pay this value if they choose to breed with that monster.

The name of the monster that is on offer to be bred will be shown in this column.

The monster’s attributes will appear here.

#### 2.3.1.11. Selling Options



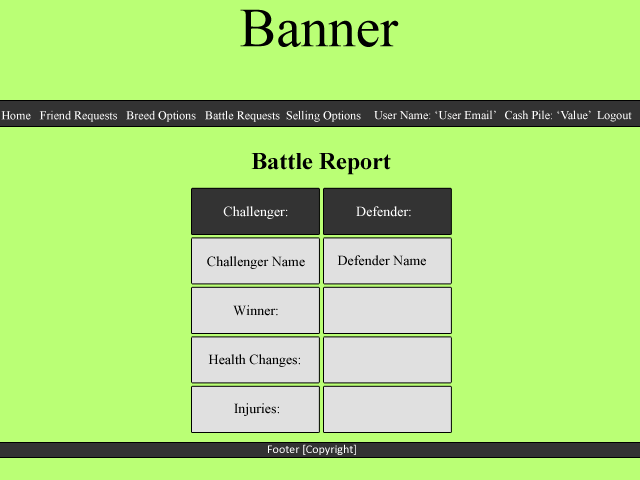
The user can add a cash value to the monster which will represent the sale price.

The user will be able to choose to sell the monster from this field.

The monster’s attributes will appear here.

The monster’s name will appear in this column.

#### 2.3.1.12. Battle Report



Monster injuries from the battle will appear here.

The remaining health of the monster will appear here.

The winner of the battle will be shown here.

The names of the contestants will appear here.

These headings represent which side the challenger and defender are.

## 2.5. Risk Assessment Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk** | **Risk Severity** | **What Will It Affect?** | **Measures Put In Place** | **Other Notes** |
| Illness | Medium-High | Project milestones and group centralisation. | Alert group as soon as possible so group Could act accordingly. |  |
| Natural Disaster | Low-High | Depending on the natural disaster, progress may be affected in different ways. e.g. flooding may change team members’ priorities. | Contact numbers distributed to group and try to a communication channel open with group to deal with problems. |  |
| Small coding error and unable to track source of error | Low | Ability to proceed with implementation.  Increased stress to | Use version management to roll back code to last working commit of code. |  |
| Deletion of local git repository | Low | Could affect progress of coding/ loss of code. | Commit updates frequently to avoid losing too much code.  Can re-clone online repository to local system. |  |
| Deletion of online git repository | Medium-High | Could lose whole project work. | Make sure each group member has an up-to-date local clone of the git repository to re-upload to the online repository/recreate a new one. | To delete the whole online git repository, the version management controller must delete it and manually and confirm deletion by entering the name of the repository to be deleted. |
| Code incompatibility | Low-High | Interaction of code between group members’ work could be hit and cause program-wide problems. | Make sure coders meet up frequently and work on code together along with QA manager.  Keep all work as centralised as possible by having frequent group meetings in which to do work/assign task at. | If work becomes decentralised, code incompatibility could become a big problem. |
| Server-server interaction problems | Medium-High | Servers may not be able to communicate with each other. | Make sure frequent meetings between allocated members from other groups are arranged to discuss server-server interaction protocols. | Keep the program as simple as possible but making sure that the program meets all of the requirements.  By keeping it as simple as possible, it is less likely that server-server interaction problems will occur. |
| Loss of project direction | Medium | Wrong tasks being allocated so wrong work is produced for delivery. | Frequent group meetings, checking requirements specification and appropriate documents to find check if the right goals are being worked towards at the right time. |  |
| Individual circumstances | Low-High | Could affect work motivation/priorities as well as group dynamic depending on situation. | Group supports each other appropriately having meetings to decide what to do if needed. Handle delicately. | May not become aware of individual’s circumstances straight away but this is expected. |
| Browser compatibility | Low | Client requires program to run on all installed browsers in the Delphinium and Solarium. | Keep interface with browser simple and validate to make sure it is compatible. |  |

## 2.6. Gantt Chart

For the same reasons that the task allocation table cannot be presented as a readable fashion, see the **“Project Gantt Chart 27-10-2012.pdf”** document for the Gantt Chart. This will be sent along with the Project Plan and the Task Allocation Table.

# 3. References

N/A

# 4. Document Change History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | CCF No. | Date | Changes made to document | Changed By |
| 1 | N/A | 26/10/12 | This is the first draft | All (each member wrote their part) |
| 1.1 | N/A | 26/10/12 | Table of contents, Introduction and formatting was added. | sjm16 |
| 1.2 | N/A | 27/10/12 | Adding of risk assessment table and note of Task Allocation and Gantt Chart will be sent along with this Project Plan. | sjm16 |
| 1.3 | N/A | 09/12/12 | Removal of Battle Report Seq. Diagram | sjm16 |
| 1.4 | N/A | 30/01/13 | Update to UI Designs and added Task Allocation to the document | sjm16 |