

Project Plan

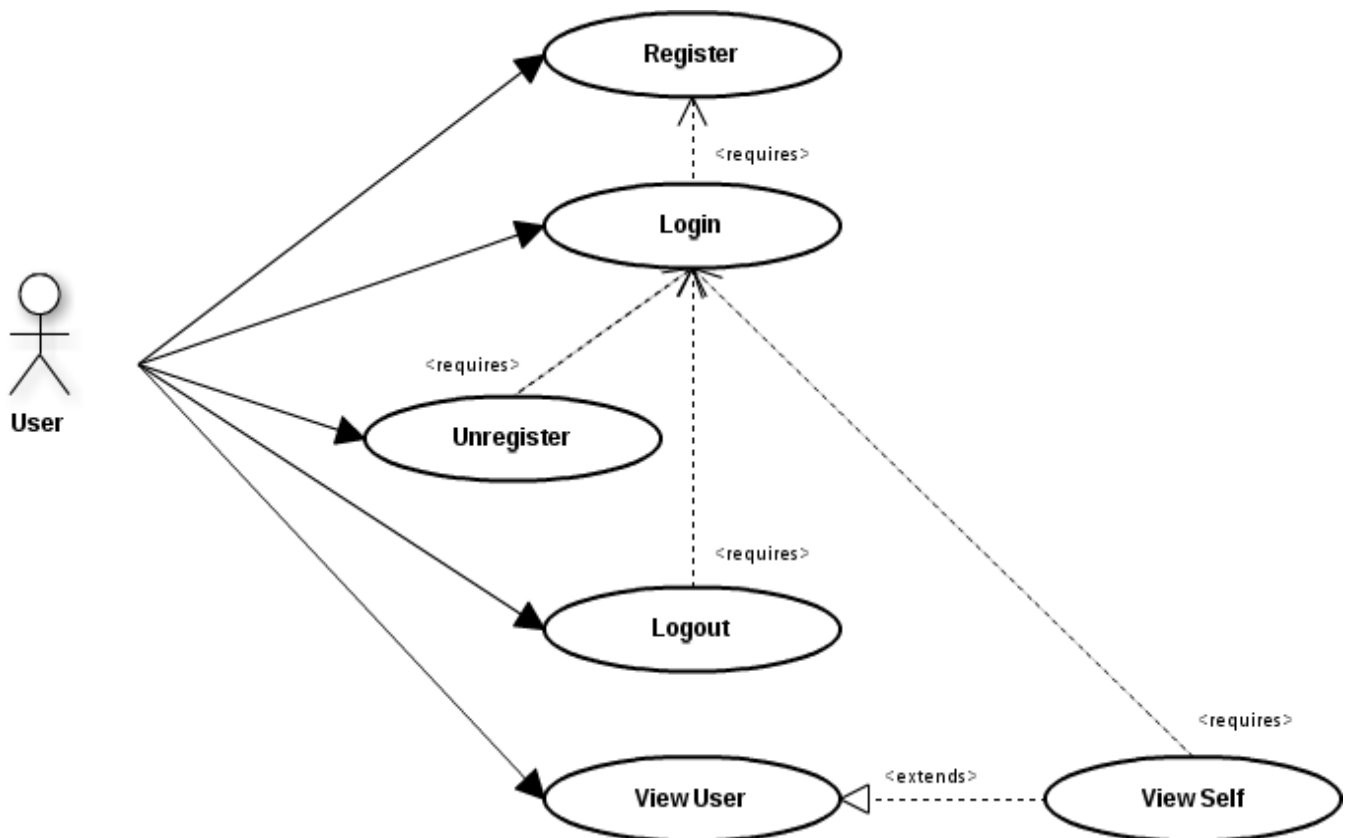
Author(s)	Adrian Gawryszewski James Barnett Samuel Clements Stoyko Veselinov Kodzhabashev John Bolton
Configuration Reference	Unknown
Last Update	18/02/13
Version Number	1.4
Document Status	Final

Table of Contents

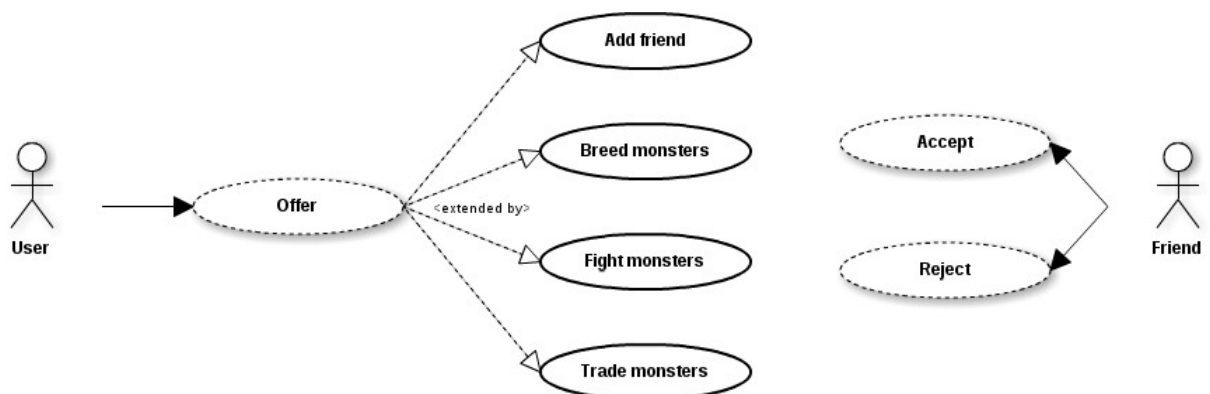
Diagrams.....	3
Account Management Use Case Diagram.....	3
Offers Use Case Diagram.....	3
Project Plan.....	4
Gantt Chart.....	4
Event Timeline.....	5
Risk Assessment.....	7
User Interaction Design.....	10
Purpose of this document.....	10
Scope.....	10
Objectives.....	10
DOCUMENT CONTENT.....	11
Account management.....	11
Register/Unregister.....	11
Login/Logoff.....	11
Offers.....	13
Friend Requests.....	13
Fight request.....	13
Breed.....	14
Trade.....	15
Notifications.....	16
Accept an offer.....	16
Other Views.....	17
Home Screen.....	17
View Friends Monsters.....	17
Profile View.....	18
REFERENCES.....	18
DOCUMENT CHANGE HISTORY.....	18

Diagrams

Account Management Use Case Diagram



Offers Use Case Diagram



Project Plan

Gantt Chart

See docs/final-docs/Gantt chart

Event Timeline

Main Event	Date	What took place and actions of project members
Arrange project team positions	08/10/12	Have a group meeting to decide positions in the team.
Introduction and details of the proposed system	08/10/12	Introduce the team members to the task ahead and give details of the system and discuss plans ahead.
First part of project documentation to be delivered	15/10/12	To allocate tasks and plan ahead so that work is collaborated on time for the first deliverable.
First deliverable	29/10/12	To deliver the work by this date.
Diagrams	29/10/12	To create a class and component diagram to give an understanding of the system structure.
Test identification	29/10/12	To identify tests that can be carried out on the finished system.
Standards meeting	29/10/12	To attend a standards meeting to discuss with the other group standards to be used.
Test Specification	29/10/12	To create the test specification for the final system so that it could be delivered.
Allocating work for coding week	29/10/12	To split up the workload so it was fairly distributed to members of the project for coding week including documentation.
Skeleton code	05/11/12	To prepare the skeleton code ready for the implementation of main methods.
Standards meeting	12/11/12	To attend a standards meeting to discuss with the other group standards to be used.
Second deliverable	12/11/12	To deliver all required documentation by this point.
Architectural and interface description	12/11/12	To put together the architectural description and interface description.
Start of coding	12/11/12	To start coding.
References	26/11/12	To put together references and appendices for the project.
Standards meeting	26/11/12	To attend a standards meeting to discuss with the other group standards to be used.

Third deliverable	03/12/12	To deliver all the required documentation up to this point which included the test specification.
Prototype demo submission	10/12/12	To research prototyping software to be used.
Integration and testing week	28/01/12	This week will be used to implement all the main methods in terms of coding and bring together the application in terms of the web interface and the back end code.
Fourth deliverable	03/02/12	To deliver the application to the end user.
Final deliverable	18/02/12	To deliver all of the work including the documentation.

Risk Assessment

This will look at various risks that can occur during the projects life. Considerations will be made and based on the risks, that can or may arise, actions will be put in place to either stop them occurring or to help recovery in such a situation.

Due to the nature of this project, there are not as many risks as there would be if this were a real life project. For example, there are no funding considerations to take into account, there are no staffing with top talent issues as the groups were chosen by the department and there are not as many software issues to take into consideration given that the university provides all the adequate software needed to successfully complete the project as required.

Risk	Action To Take
One of the major risks to the project is group members may become ill, fall behind or have a number of other personal issues to deal with that can have a direct impact on their productivity in terms of the project. If a project member falls behind there is a chance it could have an impact on their whole project productivity and therefore cause major problems in getting the project completed on time.	To combat this there are a number of things that can be done. Members who are ill shall be required to alert the project manager as soon as possible if they know it is likely that they are going to be off for some time so the work they had been allocated can be given to another member in the group and the whole project is not held back. If a member can't attend a meeting or meet a deadline they should also inform the project manager so that the project manager knows why. Time at the end of the project can also be set so that those that need to catch up on work can do so.
There is a risk that the project manager or quality assurance manager can become ill or cannot attend meetings for personal reasons.	To help stop this having severe consequences on the project work deputy roles have been given out to other members in the group. These then from attending meetings themselves should be able to pick up where those in the more senior positions left off if they are to become ill or not able to attend.
Group member turnover is another key risk. If a key programmer who has done most of the code just decides to drop out of university and there is no one else with understanding of the code that has been created it can either entirely derail the project or set the project back by quite a bit.	It's therefore essential that at least a couple of group members are working on the code so that if personnel leave for whatever reason there is someone there to carry on the implementation and update the other members of what is going on in terms of the implementation and integration of the application.
There is a risk that a project member may have issues with their own personal hardware and lose important documentation or code that they have themselves worked on.	To overcome this it is suggested that as soon as a piece of work is completed it is not only backed up on another device or online but submitted to the git repository so that the other members can sync with it and they will then themselves have

	<p>a copy of the work done. More group members having a copy of completed work minimises the risk of losing it.</p>
<p>Communication breakdown between groups is possibly one of the biggest threats to this project in terms of risk. If standards are not set by groups and a decision is made with a strict understanding from all involved then there is a major problem whereby the applications created are unlikely to be able to communicate with one another. This would then mean that the project fails to deliver on one of the requirements set out at the start of the project in server communication.</p>	<p>To deal with this effectively it is important a number of standards meetings take place so each group is kept in the loop in terms of what standards are being used and why.</p>
<p>Another threat is there could be a loss of internet connection which holds back work production on the project. For example; if the on-campus network has connection problems during coding week this may cause synchronisation errors which may lead to lack of productivity. It may also stop work being able to be accessed if it is all stored online and there are no offline copies stored to be readily worked on.</p>	<p>To overcome this the group should keep an offline copy of all work on a regular basis by syncing with the Git repository so that if there is a connection failure they can carry on doing work on an offline copy and then sync at a time when connectivity is restored.</p>
<p>Lack of understanding from certain group members can also be a problem for any group project. If there is a lack of understanding and a member has been allocated work they may not do the required work and instead do something else believing they are doing the right thing.</p>	<p>To overcome this it is important that if group members have any doubts or misunderstandings about a piece of allocated work that they speak to the project manager to come to an understanding on the issue.</p>
<p>The scope is another issue. The project requirements and features listed may be beyond that of what the development team can deliver in the time available.</p>	<p>Assign the appropriate tasks to the group members with the skills to carry out those tasks efficiently and set work in accordance to project members strengths.</p>
<p>Another risk is developing the wrong user interface. As the front of the application it is important that the user interface is as required with the functionality set in the project requirements. Failure to do so could mean that the customer is unhappy with the end product.</p>	<p>To overcome this it is important that the goals are understood and that the user interface does exactly what it is meant to do without confusing the user. A good understanding of its purpose means it is likely to be implemented correctly and give the end user a more enjoyable experience using the software. If this were a real world project then surveys could also be handed out to the public to receive feedback on the user interface design.</p>

<p>Plagiarism is a massive risk to the group project as one person committing such an offence could have consequences on the project as a whole and cause problems. This also applies to a real world project whereby committing plagiarism could not only have productivity implications but financial and legal implications as well.</p>	<p>To overcome this it is essential that all code is checked and verified and that each user submits work clearly identifying the author or creator of the work so if plagiarism is an issue it is easy to identify the person responsible. Git also helps with this as submits are recorded so its easy to backtrack and see who submitted the work.</p>
---	---

User Interaction Design

Purpose of this document

This document describes the user interaction design for the application.

Scope

This document covers the possible interactions a user may have with the end product.

Objectives

The objectives of this document are to:

- Describe all possible user interactions with the end product, without any reference to technical aspects of the product.

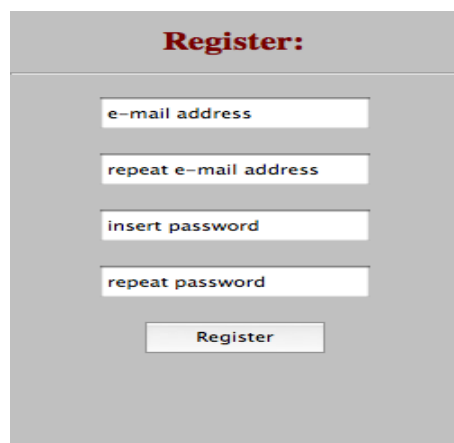
DOCUMENT CONTENT

Account management

When a user starts playing the ‘Monster Mash’ game, they will first have to register and then login in order to start playing. The homepage for the game will display options for the user to provide an email and password to login, or a button to register.

Register/Unregister

The first step you need to make in order to play the game is to create an account. When you click on the “register” button, you will be taken to another page where you will be able to fill in your information, including an email and password. When this has been filled in, the user will be able to click a button at the bottom of the page to continue. A page will then be displayed telling the user if the action was successful or not.



A registration form titled "Register:" in red text. It contains four input fields: "e-mail address", "repeat e-mail address", "insert password", and "repeat password". Below these fields is a "Register" button.

Users can remove their account with the “unregister” button placed on MyProfile site. Again, a page will then be displayed telling the user if the action was successful or not.



A profile page titled "Profile" in large, bold, black text. Below the title, it says "Username: User". At the bottom of the page is an "Unregister" button.

Login/Logoff

A login form will be displayed on the homepage. The user will need to enter the correct account details, and will then be taken to their account page. If the account details are incorrect, they will be

returned to the home page and shown a message explaining this. Underneath the form will be an option to register.

<div>Log in:</div> <div>Email</div> <div>Password</div> <div>Login</div>	<div>Register:</div> <div>Register</div>
--	--

Once the user has logged into their account, they will be able to see a logoff button. Pressing it will immediately take them out of the game and back to the home page of the game.

MyProfile	Notofication	Monsters	Highscores	Log out
-----------	--------------	----------	------------	---------

Offers

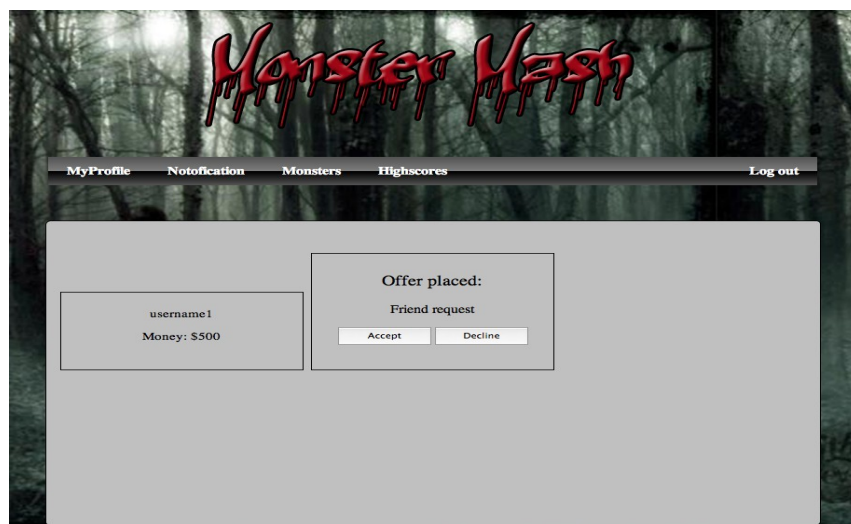
There are several types of user interaction that can be described in terms of “offers”, where one user offers a request to another user, who then has the chance to either accept or reject the offer. There are four interactions that work this way:

Friend Requests

To send friend request to another player we need to insert friends name (email address) in a text box, placed below friends list and press “Add” button. After that a friend request will occur in a notifications panel.

FRIENDS:	\$
user1@email.com	\$50
user2@email.com	\$150
user3@email.com	\$250
<input type="text" value="user@email.com"/>	<input type="button" value="Add"/>

When click on friend request in notification panel you will be redirected to new window, where you will be able to accept or decline friends request.



If offer is accepted a new friend occurs in your friend table. From now on you can send offers to this player.

Fight request

To sent a fight request to another player you have to press “fight” button placed under your friend name in friends table.

FRIENDS:	\$
<div> <div>username@email.com</div> <div>\$ 900</div> <div>Fight</div> <div>Insepect</div> </div>	
<input type="text"/> <div>Add</div>	

You will be redirected to another page, where you have to choose one monster from your and friend's farm and specify a prize.

<div>me@email.com</div> <div>Money: \$500</div> <div><input checked="" type="checkbox"/> Monster 1</div> <div><input type="checkbox"/> Monster 2</div>	<div>Offer placed:</div> <div>Fight request</div> <div>Prize: <input type="text" value="100"/></div> <div>Sent fight request</div>	<div>your_friend@email.com</div> <div>Money: \$500</div> <div><input type="checkbox"/> Monster 1</div> <div><input checked="" type="checkbox"/> Monster 2</div>
--	--	---

Your opponent has to accept your offer, otherwise any action won't be taken. If accepted monsters proceed to fight. After the fight a winner receives money reward. A report from a fight occurs on a notifications page.

As a result of any fight your monster can be injured or die. Injury may reduce the monsters attributes, and death will remove it from the owners list of owned monsters.

Breed

To send a breed request to your friends you need to go to your monster farm. Under every Monster record is placed a “Offer to breed” button.

MONEY: \$\$\$

<p>Monster 1</p> <p>Stats Offer to breed</p>	<p>Monster 2</p> <p>Stats Offer to breed</p>	<p>Monster 3</p> <p>Stats Offer to breed</p>
<p>Monster 4</p> <p>Stats Offer to breed</p>	<p>Monster 5</p> <p>Stats Offer to breed</p>	<p>Monster 6</p> <p>Stats Offer to breed</p>

FRIENDS:	\$
Player 1	\$ 900
Player 2	\$ 800
Player 3	\$ 2503
Player 4	\$ 321
Player 5	\$ 621
Player 6	\$ 123
<input type="text"/>	Add

When you click it, you will be redirected to another page, where monster with stats are displayed and you will be asked to enter a price. Then you have to press a “Send breed offer” and it will be sent to every one from your friends list.

<p>me@email.com</p> <p>Money: \$500</p> <p>Monster 1</p> <p>Stats:</p>	<p>Offer placed:</p> <p>Breed request</p> <p>Prize: <input type="text"/></p> <p>Sent breed request</p>
--	--

If anyone will accept your offer a person who accepted gets a new monster and other side gets a money reward. If offer is declined nothing happens. In both cases appropriate message occurs on a notification page.

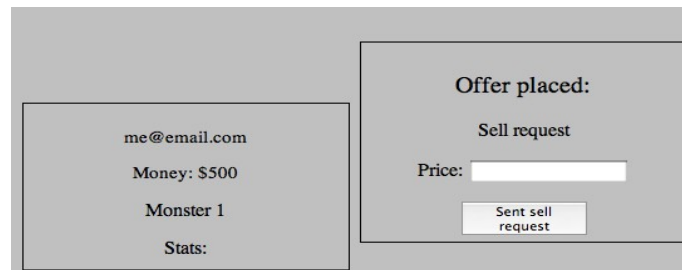
Trade

This game allows the user to trade our monsters to other players. We might decide we want to sell one of our creatures in order to earn some extra money etc. To send a sell request we need to go to our monster farm and press the ”sell” button under the chosen monster.

<p>Monster 1</p> <p>Stats Offer to breed</p> <p>Sell</p>	<p>Monster 2</p> <p>Stats Offer to breed</p> <p>Sell</p>	<p>Monster 3</p> <p>Stats Offer to breed</p> <p>Sell</p>
<p>Monster 4</p> <p>Stats Offer to breed</p> <p>Sell</p>	<p>Monster 5</p> <p>Stats Offer to breed</p> <p>Sell</p>	<p>Monster 6</p> <p>Stats Offer to breed</p> <p>Sell</p>

FRIENDS:	\$
Player 1	\$ 900
Player 2	\$ 800
Player 3	\$ 2503
Player 4	\$ 321
Player 5	\$ 621
Player 6	\$ 123
<input type="text"/>	Add

System redirects us to another page, where we decide how much we expect to earn on this transaction and press “sent sell request”. This will sent sell request to all players from our friends list.

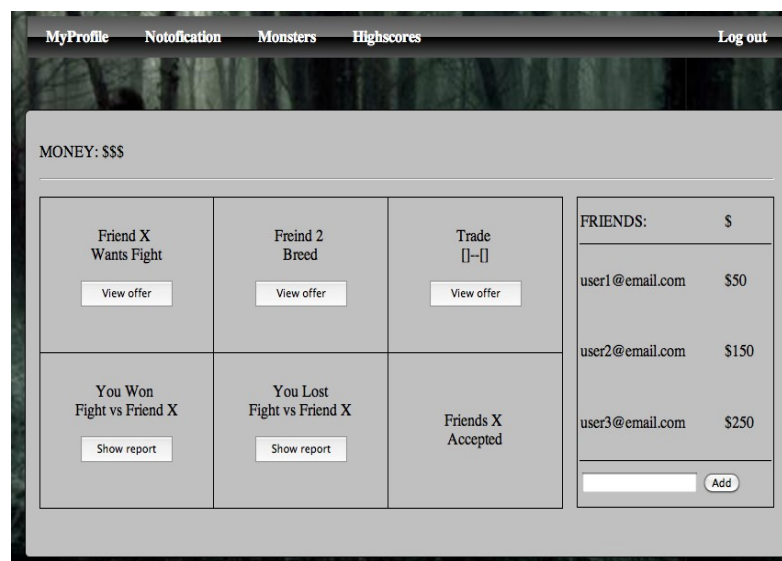


The screenshot shows a user interface with two main panels. The left panel displays user information: 'me@email.com', 'Money: \$500', 'Monster 1', and 'Stats:'. The right panel is titled 'Offer placed:' and contains a 'Sell request' section with a 'Price:' input field and a 'Sent sell request' button.

If offer is accepted, we get money and the other side receives a monster. If offer is declined nothing happens. In both cases an appropriate message occurs on our notification page.

Notifications

Once people have made offers to you, they will appear on the notifications page. Clicking on a “View offer” button will take the user to the 'Accept offer'. page.

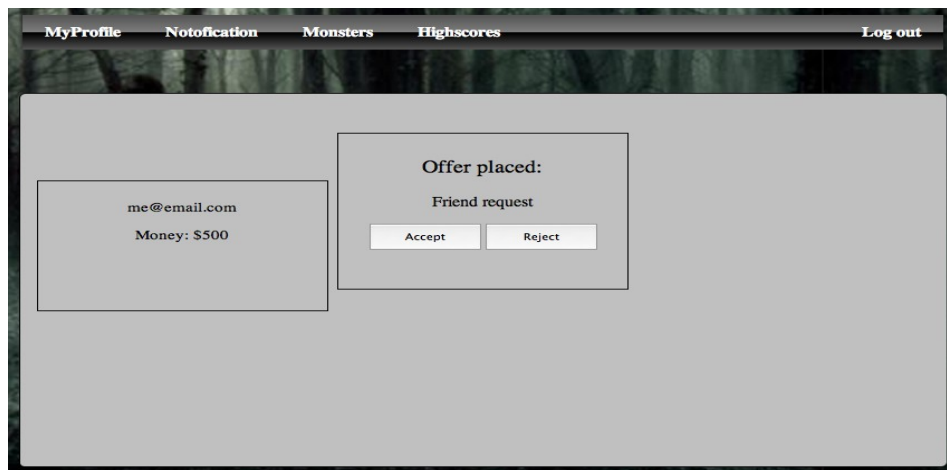


The screenshot shows a notifications page with a navigation bar at the top containing 'MyProfile', 'Notification', 'Monsters', 'Highscores', and 'Log out'. Below the navigation bar, the page is divided into several sections. On the left, there are three boxes: 'Friend X Wants Fight' with a 'View offer' button, 'Freind 2 Breed' with a 'View offer' button, and 'Trade []-[]' with a 'View offer' button. Below these are three more boxes: 'You Won Fight vs Friend X' with a 'Show report' button, 'You Lost Fight vs Friend X' with a 'Show report' button, and 'Friends X Accepted'. On the right, there is a 'FRIENDS:' section with a list of friends and their associated amounts: 'user1@email.com' for \$50, 'user2@email.com' for \$150, and 'user3@email.com' for \$250. At the bottom of this section is an 'Add' button.

We can also check reports from our last fights (won and lost) by clicking on the “Show report” button.

Accept an offer

A user can select an offer, which will then show this screen, displaying details of the offer to be accepted. The details will be based on whether the offer is a friend request, or a monster trade/breed/fight. Either the friends or the monsters involved will be shown on the left and right, as a small card containing their stats. In the centre will be buttons to accept and reject the offer.



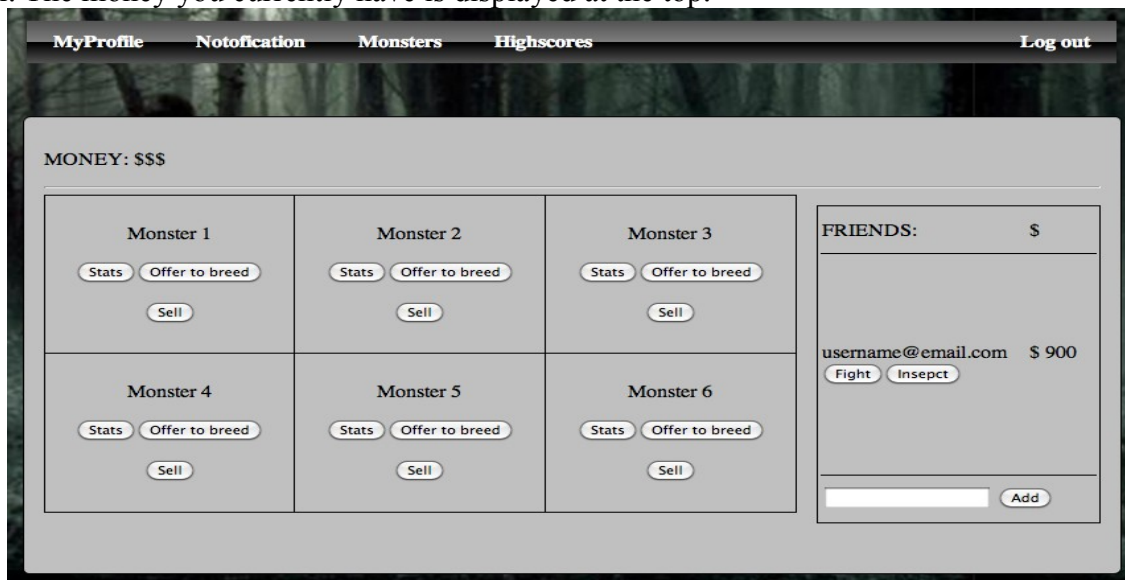
When the user continues, they will be taken back to the previous page and shown a message telling them that the offer has been accepted or rejected.

Other Views

Home Screen

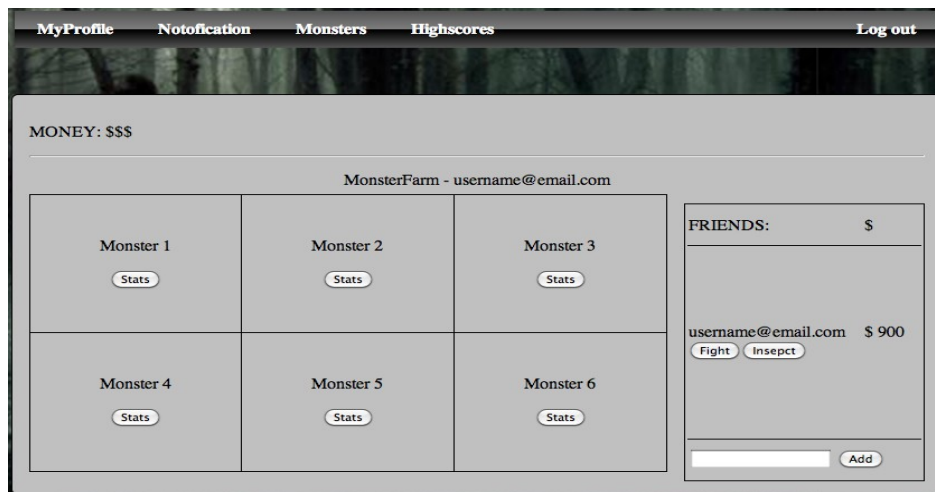
After you successfully login to a system you will be directed to a My Farm page.

From the main home screen you can see a panel of your monsters, a list of your friends on the far right and an option to add friends. You can also logout using the logout button at the top right corner. The money you currently have is displayed at the top.



View Friends Monsters

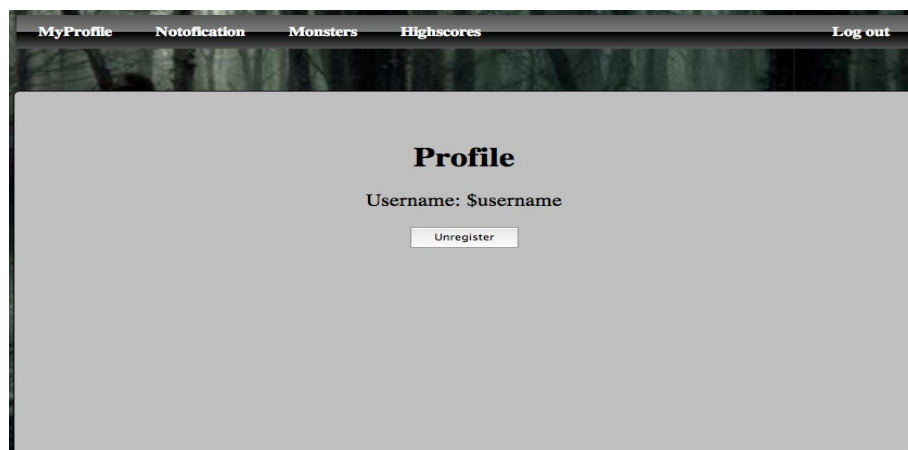
You can inspect you friend's monster farm and check what monster he/she has simply by clicking on the "Inspect" button under your friend's name in a friend's list table.



On this page will be a status bar that will tell you the username of your friend so you know you are on the right page. Below that you will see the main panel which will display the monsters that your friend has and the stats for each monster.

Profile View

On this page you can see your username and a “Unregister” button. You can use it to remove your account. You can also logout using the logout button at the top right corner.



REFERENCES

This section is an automatic bibliography.

DOCUMENT CHANGE HISTORY

Version	CCF No.	Date	Sections changed	Changed by
1.1	N/A	31/10/12	Corrected document to be inline with the General Documentation Standards	Samuel Clements

Version	CCF No.	Date	Sections changed	Changed by
1.2	N/A	15/02/13	Put all required documents into one file so that it can be submitted once the Gantt chart has been added.	James Barnett
1.3		17/02/13	All section updated/added print screens	Adrian Gawryszewski
1.4		18/02/13	Spelling + grammar changes	John Bolton