|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
| **Team Dec 14-13 CySwap Project Plan** |
| Client: Merry RankinAdviser: Professor MitraTeam Members:Fabian BriesmooreJared CookKyle JohnsonAdam SundermanJake Roman |

Table of Contents

[Problem/need Statement 3](#_Toc405877675)

[Work Plan 3](#_Toc405877676)

[System Block Diagram 5](#_Toc405877748)

[Operating Environment 6](#_Toc405877750)

[User Interface Description 7](#_Toc405877751)

[Template Pieces 7](#_Toc405877752)

[Pages 8](#_Toc405877753)

[Home Screen 8](#_Toc405877754)

[Category Screen 9](#_Toc405877755)

[Posting Screen 10](#_Toc405877756)

[Contact Seller Popup 11](#_Toc405877757)

[Report Screen 11](#_Toc405877758)

[Create a Post Screen 12](#_Toc405877759)

[Functional Requirements 13](#_Toc405877760)

[Non-Functional Requirements 14](#_Toc405877761)

[Software Specification 14](#_Toc405877762)

[Design Decisions 15](#_Toc405877763)

[Testing Process and Results 16](#_Toc405877764)

[Appendix I- User Guides 17](#_Toc405877765)

[General User 17](#_Toc405877766)

[Administrator User 18](#_Toc405877767)

[Appendix II - Versions 19](#_Toc405877768)

[Version 1 19](#_Toc405877769)

[Version 2 19](#_Toc405877770)

[Appendix III – Atlassian Software 20](#_Toc405877771)

Problem/need Statement

The Government of the Student Body is aiming to develop a website that allows Iowa State Students to buy and sell goods. The site will allow for multiple types of items to be sold including: textbooks, furniture, clothes, tickets, places for rent, etc. Each type of post will need to be organized in ways that make sense for the type of item, for example, books will be searchable with ISBN number and the website should fill in all relevant information accordingly. The site will not handle the monetary transaction between students. It should instead be a way of connecting students. However, it should use a formalized procedure for contacting and completing transactions to allow for detailed reporting features. To build a strong sense of community, the web-site should be accessible through the use of netid (@iastate.edu emails).

# Work Plan

## Team Member Roles

|  |  |
| --- | --- |
| **Team Member** | **Role** |
| Fabian Briesmoore | Key Idea Holder |
| Jared Cook | Webmaster |
| Kyle Johnson | Team Leader |
| Adam Sunderman | Communicator |
| Jake Roman | Quality Assurance |

## Risks

It is possible that a team member will leave our project. This risk will require us to maintain thorough documentation through the development of this project.

We will have to be prepared to handle large amount of traffic. We will likely need to be able to process significant traffic during the start of each semester, when students are buying textbooks.

Not making the system easily maintainable would lead to the system not being used in the future.
Developing the system using Laravel and PHP where our team has little experience creates the risk of not having the time to implement every proposed feature by our client.
Developing the system to be extremely expandable where our client will not have to look at code is a challenging task and creates more program vulnerabilities.

## Deliverables

|  |  |
| --- | --- |
| **Deliverable** | **Date** |
| Design Document Version 1 | 03/14/14 |
| Project Plan Version 2 | 04/04/14 |
| Design Document Final Version | 04/25/14 |
| Spring Project Plan Final Version | 04/25/14 |
| Spring Presentation | 04/29/14 |
| CySwap Version 1 | 09/20/14 |
| Final Poster | 12/01/14 |
| Revised Final Project Plan | 12/08/14 |
| Presentation for Industry Review | 12/12/14 |
| Demo for Industry Review | 12/12/14 |
| CySwap Version 2 | 12/20/14 |

## Project Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| **Task Name** | **Start Date** | **End Date** | **Duration** |
| **Course Deliverables** | **03/06/14** | **04/29/14** | **39** |
| Design Document Version 1 | 03/06/14 | 03/14/14 | 7 |
| Project Plan Version 2 | 03/27/14 | 04/04/14 | 7 |
| Design Document Final Version | 04/14/14 | 04/25/14 | 10 |
| Project Plan Final Version | 04/14/14 | 04/25/14 | 10 |
| Presentation | 04/14/14 | 04/29/14 | 12 |
| **Product Development** | **03/06/14** | **12/20/14** | **210** |
| Create Iteration Plan | 03/06/14 | 03/19/14 | 10 |
| Gather Requirements | 03/19/14 | 04/15/14 |  |
| Version 1 | 04/16/14 | 09/20/14 | 27 |
| Write User Stories | 04/16/14 | 03/22/14 | 5 |
| Develop Code | 03/22/14 | 09/13/14 | 60 |
| Test Code | 09/14/14 | 09/20/14 | 5 |
| Version 1 Release | 09/20/14 | 09/20/14 | 1 |
| Version 2 | 09/21/14 | 12/20/14 | 40 |
| Write User Stories | 09/22/14 | 10/01/14 | 9 |
| Develop Code | 10/02/14 | 12/08/14 | 45 |
| Test Code | 12/09/14 | 12/20/14 | 10 |
| Version 2 Release | 12/20/14 | 12/20/14 | 1 |

System Block Diagram

****

Figure 1-System Block Diagram

The server communicates with the database to get user data (such as account details and their postings). This data, along with other data (possibly from other databases) is used to generate a page. The pages will likely be generated using php. The server is also responsible for authenticating the user. The client will communicate to the server through a session.

## System Description

We aim to develop a web based application to allow users to view and post listings for several different types of items for sale with fellow Iowa State students. Posts will be stored in the server’s database for future access with any potential buyers. The users will be able to find posts relevant to them through search categories (books, tickets, furniture, etc.) as well as search filters (price, popularity, etc.). These categories will be configurable by privileged users using web pages made available to system administrators.

The transactions will be handled independently from our system. However, the system will implement a formalized process for making contact with a seller to ensure accurate reporting statistics within our system as well as help automate post management (removing completed transactions and keeping track of item interest and popularity). Upon viewing a post, a user will be able to contact the seller via a “Contact Seller” button. They will then fill in a form that will allow the system to complete an auto-generated email to the seller with the offer. If a post has not been closed a week after contact, our system will auto-generate and send an email to inquire if the transaction had been completed, and update internal information accordingly. Once a transaction is complete, automatically generated emails will ask the buyer and seller to submit feedback on each other. The system will keep track of the successful transaction in the database for reporting purposes.

When users post an item, the system will allow them to choose a category. Upon selecting a category, the system will use an appropriate form to collect the necessary post information in addition to helping automate the process for items with standardized data. An example of this is the use of ISBN numbers for books to help auto-complete the majority of the form for the user. Upon successful form completion, the item will have a posting generated and stored within our database. The post will be accessible for any users searching for the item.

Finally, The Government of the Student Body will be able to use the system with additional functionality and will be given further information about the system. As administrators, they will be given access to manage users. This capability includes being able to banning and suspending users. Additionally, administrators will be able to view reports of users that have been reported by other users. The categories that the users can view and add posts to will be able to be added and removed. Administrators will also be able to update static content on the website (i.e. Terms of Use).

# Operating Environment

Our website functionality will be supported on the most popular browsers (Mozilla Firefox, Google Chrome, Internet Explorer, Safari) and mobile friendly for phones running Android or iOS. Though functional on other platforms, Firefox and Chrome are the only platforms that styling is supported on. We will have a MySQL database that will store information about users, their posts, and other necessary information. Our site will be running on a virtual LAMP Server (Linux, Apache, MySQL, PHP server) hosted by ITS.

# User Interface Description

## Template Pieces

*Header*

* The header will consist of two rows. The top row will contain the logo, site name, and search bar.
* The second row (directly under the top) has links to other pages. These links include a link to the home page, a link to the categories page, a link to post an item. A user that isn’t logged in will be provided a link to login. Following login, links to logout and view the profile of the user being logged in will be available. Administrators will additionally be able to access a page with administrator links.

*Footer*

* The footer will contain static content. This includes Terms of Use, About Us, Contact Us, and Safety and Tips.

Pages

These mockups are what started our design process. We created these after several meetings with the client to determine their preferences. These mockups have minor differences from the current application as updates to the design were made during development.

### Home Screen



Figure 2- CySwap Home Page

### Category Screen



Figure 3- CySwap Category Page shows brief descriptions of several postings within a certain category.

### Posting Screen



Figure 4- CySwap Posting Screen shows an individual posting.

### Contact Seller Popup



Figure 5- Contact Seller Popup will appear when the ‘Contact Seller’ button from the previous screen is clicked.

### Report Screen



Figure 6- CySwap Report Screen is where a user can report behavior that is a violation of the site’s terms of use.

### Create a Post Screen



Figure 7- CySwap Create a Post Screen is where a user will post an item to sell.

# Functional Requirements

|  |  |
| --- | --- |
| The system shall allow users to login and logout via Iowa State’s NetID system. | FR-1 |
| The system shall allow users to post items for sale and edit a previously post. | FR-2 |
| The system shall allow users to search for posts based on specified criteria. | FR-3 |
| The system shall allow users to view postings. | FR-4 |
| The system shall allow sellers to mark transactions as completed. | FR-5 |
| The system shall allow users to contact sellers to initiate discussion on starting a transaction. | FR-6 |
| After a transaction has been completed, the system shall allow users (buyer and seller in a specific transaction) to leave positive or negative feedback on one another. | FR-7 |
| The system shall allow users to report postings or users that violate the posting code of conduct. | FR-8 |
| The system shall display a section of Safety Tips, a Contact Us, and an About Us. | FR-9 |
| The system shall display an agree to Terms of Use page if the user has not yet agreed to them when the user logs in, tries to post an item, contact seller or report. | FR-10 |
| The system shall allow moderators or the admin to view reported postings in F.I.F.O. order. | FR-11 |
| The system shall allow moderators or the admin to remove or edit a posting. | FR-12 |
| The system shall allow moderators or the admin to suspend or ban account privileges for users who abuse the website’s posting code of conduct. | FR-13 |
| The system shall allow the admin to assign and remove moderator privileges from NetID accounts. | FR-14 |
| The system shall allow the admin to edit the content on the following pages: Terms of Use, Contact Us, About Us, and Safety | FR-15 |
| The system shall remove posts that are 30 days old. | FR-16 |
| The admin can add a new category and its associated fields. | FR-17 |
| The admin can remove a category and all postings in that category are migrated to the miscellaneous category. | FR-18 |

# Non-Functional Requirements

|  |  |
| --- | --- |
| The system should be seen as visually appealing by a number greater than 75% of surveyed students. | NFR-1 |
| During high traffic timeframes (beginning of semester) response times should be under 1 second on average. (assuming no network delay on client side) | NFR-2 |
| The site should not be able to have its information maliciously accessed through user input, direct server requests and cookie jacking. | NFR-3 |
| Must be easily maintainable after project completion. Proposed maintenance features include:* + The ability to update static content on the website.
	+ The ability to add or remove categories
	+ The ability to promote or demote a moderator
 | NFR-4 |

# Software Specification

Languages: PHP, JavaScript, HTML, CSS, and mySQL.

Libraries: Laravel (PHP framework), JQuery

Database Server: mySQL

Development Tools: Atlassian’s Jira, mySQL Workbench, WAMP server, and Sublime Text Editor

Testing Tools: Selenium SE automation tests and Java Eclipse

# Design Decisions

We determined to use PHP after examining the alternative option of using Java with Spring MVC. Our deciding factor was that PHP would have a smaller learning curve.

We determined that MySQL was the best choice for our database because of our prior knowledge with the system.

We determined that Bootstrap would be used for our web site because of Jared's previous experience with Bootstrap and its popularity and documentation. It was also chosen because it would lead to faster site development.

Laravel our PHP framework was chosen based on our research on PHP frameworks. It was found to have the smallest learning curve. In addition, Laravel was found to be the most popular PHP framework in 2013. The main article we used when making this decision can be found here: http://www.sitepoint.com/best-php-frameworks-2014/

Figure : PHP Framework popularity graph that helped us determine which Framework to work with.

For testing the site, we decided to use Selenium SE for the ease of use in creating test cases and running tests to ensure functionality is working correctly.

# Testing Process and Results

 During the work on this project, there were several testing processes that took place. The first testing took place on the Local environment utilizing the WAMP server, using the WAMP server allowed us to test on a private server environment similar to what would be used in the final site. While on the local environment, we used Selenium SE to make automation regression tests to ensure that when a new feature was added to the site, the previous working functionality was not affected. These regression tests are repeated and updated with new tests when new content was added. We also have acceptance tests for every feature of the site, these acceptance tests need to be met before a feature can be marked completed. The next step in the testing process is uploading from your local host environment into our development locker, each member of the team has access to their own development locker. Once the content is uploaded to the development locker, we are able to simulate how the site will run on the live server. After upload is complete, we used our automated tests from before to test the usability of the site on the new server environment.

Figure : Selenium SE automated regression test result page displaying output.

 Once we have determined that all functionality has worked properly on the development locker, the next step was to upload the content to the staging site. Our staging site is where we can upload our site and test it on a public server. On our staging site, we are able to send our URL to users to determine usability and provide outside feedback of our site.

 After distributing of site to various users, we received an abundance of feedback. Some of the feedback was very helpful and some was not all that helpful, but overall the feedback we received from the user testing helped us determine what needed to be changed in our site per the next version release.

# Appendix I- User Guides

### General User

#### Login

1. Click on the Login link on the right side of the navigation bar.
2. Input your netid and netid password.
3. If you have not logged in before, you will see CySwap’s Terms of Use. Please read and click I agree at the bottom of the page.

#### Post an Item

1. Ensure you are logged into the CySwap.
2. Click on the Post an Item link in the navigation bar.
3. Select the category that you would like to post an item in.
4. Fill out all required fields.
5. Click submit post.

#### Contact Seller

1. Ensure you are logged into CySwap.
2. Navigate to a posting that you are interested in buying.
3. Click content seller in the left content pane.
4. Edit the default message text to say what you would like to communicate.
5. Click the Send Email button.

#### Mark as complete

1. Ensure you are logged into CySwap.
2. Navigate to the posting you posted that you would like to complete.
3. Click Close Post in the left content pane.
4. If this item was sold, enter the NetID of the buyer. Otherwise click Delete Post
5. Click Submit

#### My Profile

1. Ensure you are logged into CySwap.
2. Click on the Profile link in the navigation bar.
3. On this page you can view your postings and what your current rating is (upvotes and downvotes).

#### Report a Posting

1. Ensure you are logged into CySwap.
2. Navigate to a posting that you are interested in buying.
3. Click the Report button at the bottom of the posting.
4. Enter a description of why you are reporting the post and click the Submit Report button.

### Administrator User

#### Manage Users

1. Ensure you are logged into CySwap.
2. Click on the Admin Area link in the navigation bar.
3. Click on the Manage Users link.
4. Enter the netid of the user you would like to manage.
5. From here you can suspend the user, ban the user or clear all suspensions and bans (pardon).

#### View Reports

1. Ensure you are logged into CySwap.
2. Click on the Admin Area link in the navigation bar.
3. Click on the View Reports link.
4. Reports are displayed if they are unresolved and in first in first out order.
5. On the left content pane, there are filter options.
6. You can resolve a report by clicking the close issue button on the report.

#### Add Category

1. Ensure you are logged into CySwap.
2. Click on the Admin Area link in the navigation bar.
3. Click on the Add Category link.
4. Enter the category name, if the item has a condition field and as many additional fields as you would like.
5. Click the create category button.

#### Remove Category

1. Ensure you are logged into CySwap.
2. Click on the Admin Area link in the navigation bar.
3. Click on the Remove Category link.
4. Select which category you would like to remove from the dropdown.
5. Click the Remove Category button.

#### Update Content

1. Ensure you are logged into CySwap.
2. Click on the Admin Area link in the navigation bar.
3. Click on the Update Content link.
4. From this page you can update the content on Terms of Use, Safety and Tips, About Us and Contact Us.

# Appendix II - Versions

## Version 1

We treated the first version of CySwap as a preliminary prototype. This version’s functionality included the capability for users to post items, view posted items, login with netid authentication, contact sellers, mark transactions as complete, view categories, and view static content. This version was useful in getting input from our client and implementing functionality that we currently use.

Despite the functionality in Version 1, many modules were not expandable or generic and needed to be restructured to accommodate the requirements made by our client. However this initial version was necessary to reach our final product from providing something that could be seen or used by the client.

## Version 2

When we started developing version 2, we realized our database setup was not designed with some of our client’s requests in mind. Furthermore we noticed that our databases had some redundancy and could be better structured overall. This resulted in our group developing our final version, which is based off of our first version.

One of our client’s major requests was to allow them to add and remove categories. In addition to restructuring our databases, we made a significant portion of our php code more generic. Our first version had static categories of Textbooks and Miscellaneous. There wasn’t the option to add and remove categories in our first version.

Additionally, administrator management was a major request made by our client. Our client did not want to deal with code to manage users and to modify terms of use and other static content. Our final version includes all of this functionality.

# Appendix III – Atlassian Software

When we were looking for task tracking software, we came across a software tool called Jira. Jira allowed for organization development tasks, assign them and represent a developmental process. A team’s current development progress can be seen at a glance.

The only roadblock in the way of us using this software is that it is not a free product. We emailed Atlassian asking if they would grant us access to the software for a year for free. Atlassian responded by giving us access to their entire set of software solutions without user limits for free! We estimated this value at $138,000.



Figure 10: Jira Screenshot. Shows Tasks to do, in progress, ready for QA and completed.