**Mayo**

Table of Contents

[1. Project Overview 1](#_Toc453056358)

[2. Project Requirements 1](#_Toc453056359)

[2.1. Requirements (stated by Customer or Teacher) 1](#_Toc453056360)

[2.2. Derived Requirements 1](#_Toc453056361)

[3. Design Plans 2](#_Toc453056362)

[3.1. Design Sub-Paragraphs 2](#_Toc453056363)

[3.1.1. Design Sub-Sub-Paragraphs 2](#_Toc453056364)

[3.1.2. Design Sub-Sub-Paragraphs 2](#_Toc453056365)

[3.2. Design Sub-Paragraphs 3](#_Toc453056366)

[3.3. Design Sub-Paragraphs 3](#_Toc453056367)

[3.4. Design Sub-Paragraphs 4](#_Toc453056368)

[4. Implementation Paragraphs 5](#_Toc453056369)

[4.1. Implementation Sub-Paragraphs 5](#_Toc453056370)

[4.2. Implementation Sub-Paragraphs 5](#_Toc453056371)

[5. Testing Paragraphs 6](#_Toc453056372)

[5.1. Testing Sub-Paragraphs 6](#_Toc453056373)

[5.2. Testing Sub-Paragraphs 6](#_Toc453056374)

[Appendix A. Requirements Traceability 7](#_Toc453056375)

List of Tables

[Table 1: HTML of Web Page 2](#_Toc453056376)

[Table 2: Another Sample Table 4](#_Toc453056377)

[Table 3: Sample Fixed Font Data 5](#_Toc453056378)

List of Figures

[Figure 1: Visio Design Diagram 3](#_Toc453056379)

[Figure 2: Java UML Class Diagram 3](#_Toc453056380)

[Figure 3: Proof that Cross-references can be Selected in Word 6](#_Toc453056381)

# Project Overview

*(Author: Isaac) This application’s purpose is to unite people with diverse and unique interests and hobbies. We are hosting our project on NSCC’s network for internal testing, using SQL and PHP on a linux server. We will use googles location API in order to “match” users with other users with similar interest, in their location*

*(Author: Erich) The app has a page that allows the user to send and receive messages to users they have matched with using the discover page. This messaging page works similarly to a typical social media site, with a column showing the user’s conversations and then displaying the messages for the conversation when clicked.*

# Project Requirements

* *Page that allows users to communicate with each other*
* *Page for users to view and modify their information*
* *Page for users to view other users in their area with common interests*
* *Login / forgot password / new user creation utility pages*
* *Linux server configuration*
* *SQL database construction*
* *PHP scripting to populate webpages with information from database*

# Requirements (stated by Customer or Teacher)

# Derived Requirements

(Author: Erich)

* PHP page for sending and receiving messages
* Column that displays all ongoing conversations for the user
* Area that displays messages from a given conversation when the user selects one to open
* A text box for the user to input and then send a message
* PHP Script to pull user messages from the database to keep their messages updated
* PHP Script to insert a new message from a user into the database so that the recipient may receive it
* A Javascript function that runs on an interval to call the update script and then parse the data returned by the script into html data
* A Javascript function that runs on send to call the insert script
* A Javascript function to create a local message div to provide instant feedback to the user that their message has been sent

# Design Plans

*Provide whatever level of detail makes sense. Note the paragraph numbering, and stay in the 1.x paragraphs for Requirements, if you wish.*

# Messages

*Break down the subparagraphs in a way that makes sense. Include diagrams and tables as needed.*

# HTML/CSS

# Conversations

(Author: Erich) Conversations are shown on the left side of the screen in the messages page. These are created by querying the conversations that the current user is involved in and then building an HTML div element for each different conversation. These can be selected by the user to display the messages in the conversation in the messages area.

Table 1: Example of a Generated Div Element for a Conversation

|  |
| --- |
| <div class="messagePreview color\_LightGrey">  <p class="messagePreviewName">John Doe</p>  <p class="messagePreviewBody">Hey! I see you like to bowl, we should go  bowling sometime this week.</p>  </div> |

# Messages

(Author: Erich) Messages are displayed on the right side of the screen in the messages page. After clicking a specific conversation, the messages within it will appear here. This is where the user can send a message to someone as well.

Table 1: Example of a Generated Div Element for a Message

|  |
| --- |
| <div class="message incoming">  <span class="color\_DarkGrey">We have a lot in common, lets hang out?</span>  </div> |

# PHP

(Author: Erich) PHP Scripts are used by the messages page to query the database for any new messages for the current user every 2.5 seconds. This allows a user to receive a message without needing to refresh the page. They are also used to insert each send message to the database so that the recipient may query it from the database next time their client does a query.

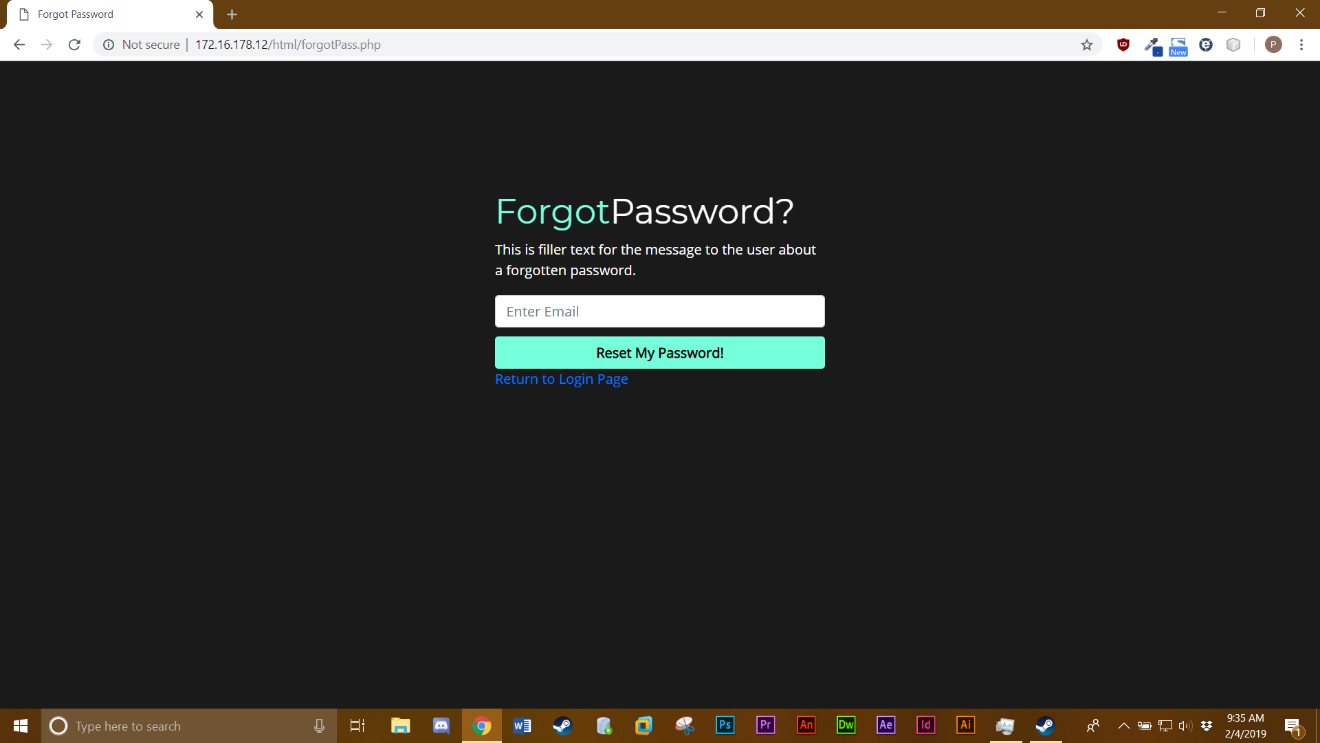
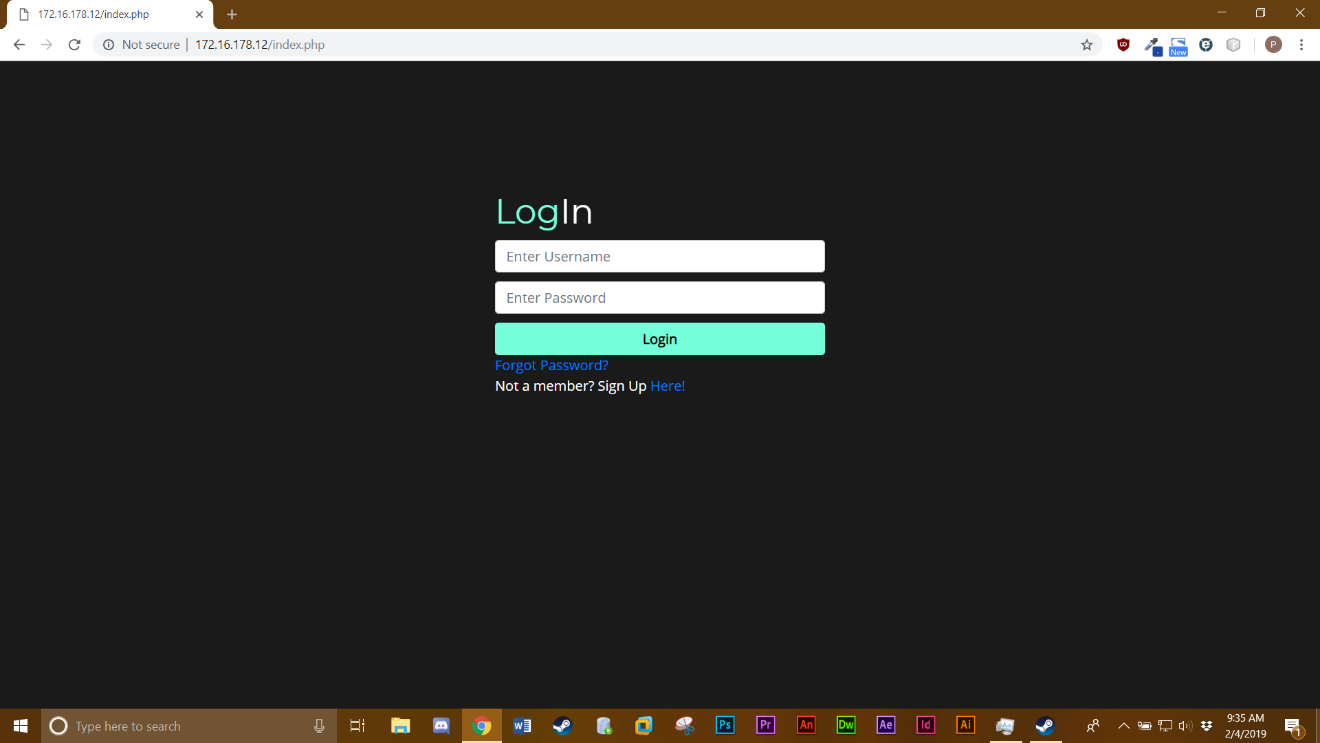
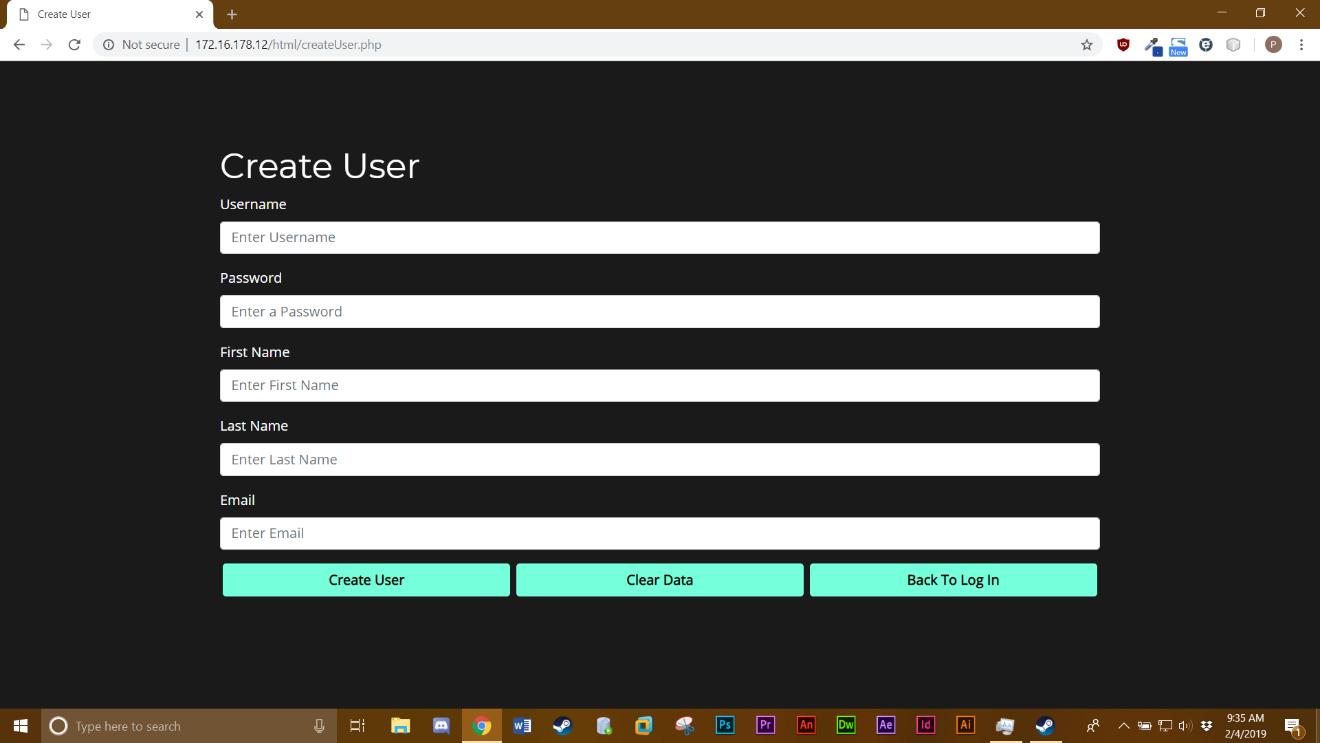
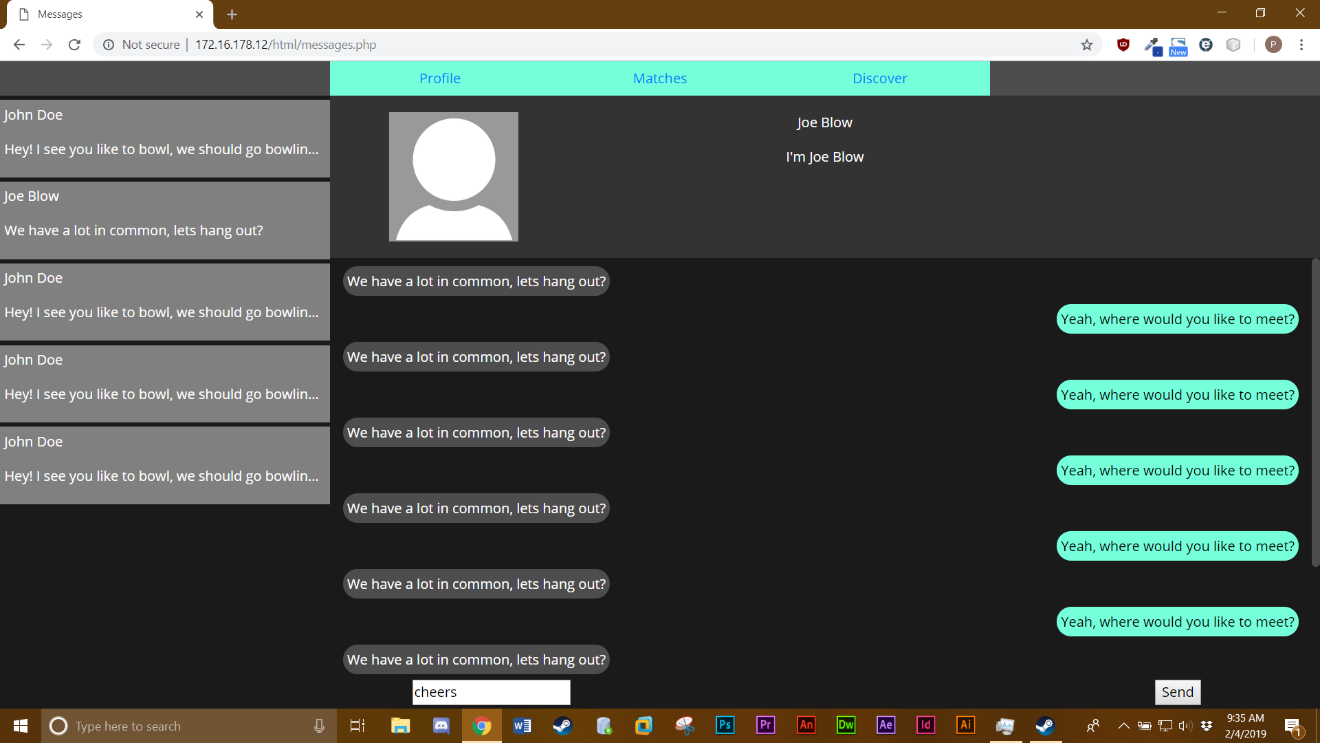
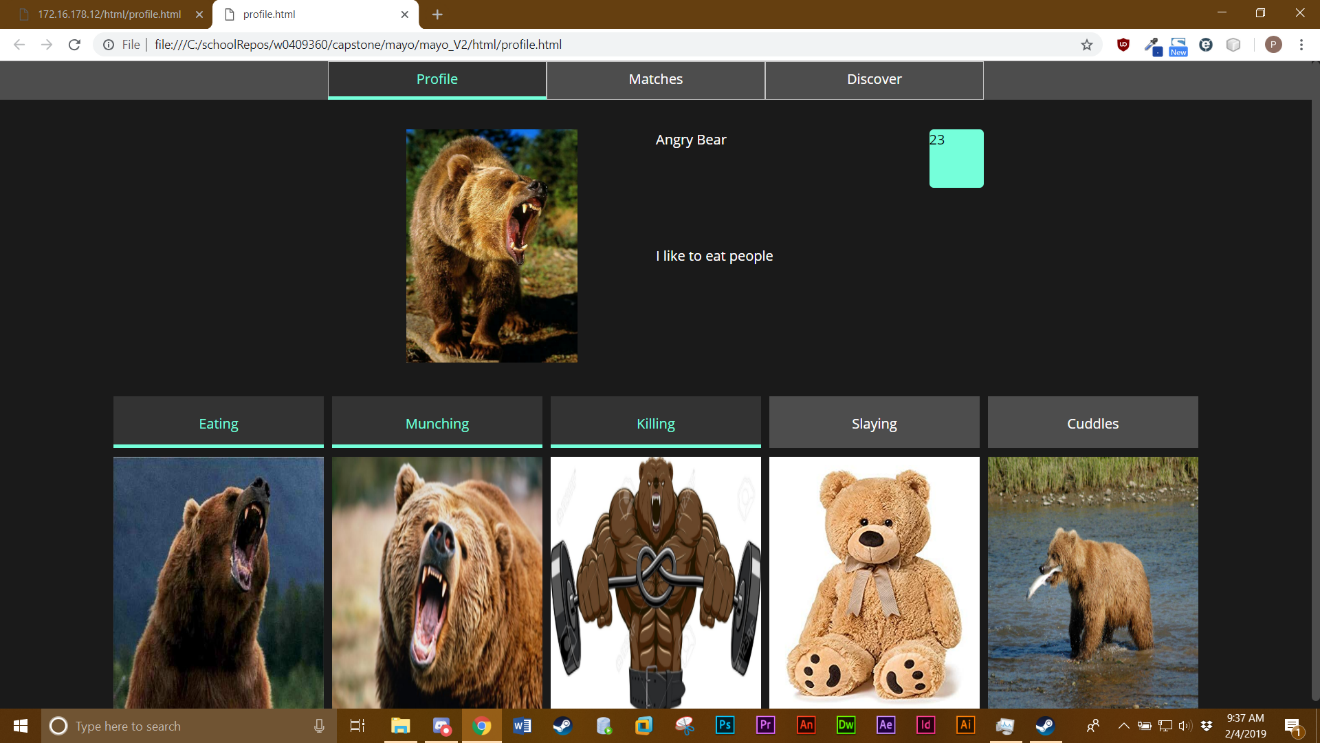
Table 1: HTML of Web Page

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <h1>My First Heading</h1>  <p>My first paragraph.</p>  </body>  </html> |

# Javascript

*(Author: Erich) Javascript is used alongside the PHP scripts on the messages page to call the script with the correct variables (user\_id) and also is how the script is called every 2.5 seconds. The Javascript is also responsible for using the returned string from the PHP script to create HTML div elements to display new messages on the screen.*

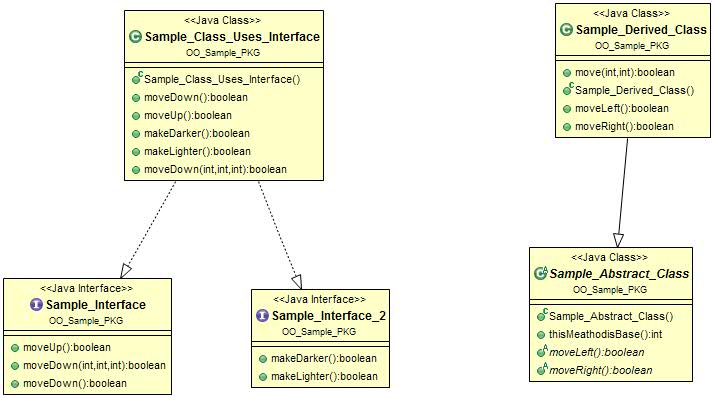
# In Progress interfaces:



# Design Sub-Paragraphs

*Include diagrams, ….*

Figure 2: Java UML Class Diagram

**

# Design Sub-Paragraphs

*Again, break down the subparagraphs in a way that makes sense. Include diagrams and tables as needed.*

Table 2: Another Sample Table

|  |  |  |
| --- | --- | --- |
| **Heading 1** | **Heading 2** | **Heading 3** |
|  |  |  |
|  |  |  |

# Implementation Paragraphs

Implementation uses Java and SQLite, or Windows Server 2008 with IIS installed, or whatever.

# Implementation Sub-Paragraphs

*Again, break down the subparagraphs in a way that makes sense. Include diagrams and tables as needed.*

# Implementation Sub-Paragraphs

*Again, break down the subparagraphs in a way that makes sense. Include diagrams and tables as needed.*

Table 3: Sample Fixed Font Data

|  |
| --- |
| REM \*\*\* Sample Batch File that only has comments \*\*\*  REM  REM Sample code in currier new font  REM And placed inside a 1x1 table (just one cell)  REM Will line up  REM Nicely  REM ..and be readable (well, more readable than if it wasn’t lined up).  REM  REM Use this method for scripts,  REM Configuration files  REM And any other data file that is convenient to read  REM With the text lined up |

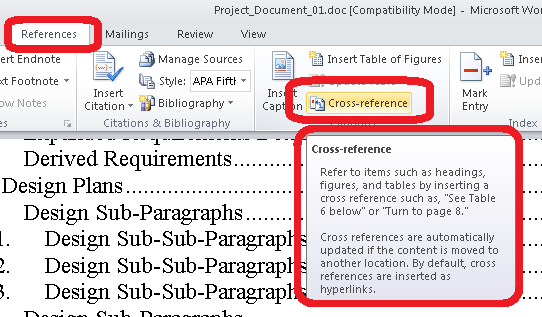
# Testing Paragraphs

*Again, break down the subparagraphs in a way that makes sense. Include diagrams and tables as needed.*

# Testing Sub-Paragraphs

*Again, break down the subparagraphs in a way that makes sense. Include diagrams and tables as needed.*

Figure 3: Proof that Cross-references can be Selected in Word



# Testing Sub-Paragraphs

*Again, break down the subparagraphs in a way that makes sense. Include diagrams and tables as needed.*

# Requirements Traceability

*Introduce the appendix and what it contains.*

|  |  |  |  |
| --- | --- | --- | --- |
| Paragraph Defined | Requirements Text | Paragraph Implemented | Paragraph Tested |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |