

Design Documentation Deliverable

Date

5/31/2017

Group Name/Number

Group - 8

Student ID	Student Name	Signature
15155527	William O'Leary	William O'Leary
15160009	Aidan Cleese	Aidan Cleese
15148602	Aaron Dunne	Aaron Dunne
15182819	BERNARD STEENERS	Bernard Steeners

Please use the checklist below to ensure that your report contains all the items required.

- ☒ Purpose of the website (a brief introduction)
- ☒ User Types
- ☒ Justification of Functionality
- ☒ Potential ramifications
- ☒ Detailed Description
- ☒ Technologies
- ☒ Appendices
- ☒ List of major processes
- ☐ Database tables

UL Task Proofreading Website

1.0 Purpose of Website

The purpose of Proofreading website is to provide an online hub for students to post their work such as theses, dissertations, assignments and research papers. This work can then be proofread by students and college staff with the intention of providing feedback or help to the students. Currently if say a final year student wants advice of his final year project he must go and seek out people who can give him advice and help but searching for people in person can be quite time consuming especially if that student is under a lot of stress for his college semester as a lot of students are. We hope this website can provide the tools and services required to make this process easier and less time consuming for the student. Students who use the service will be rewarded for their efforts in a point system and they may gain moderator privileges. The aim is that the site will be used by students to improve on their work and also be used by staff to give their own opinion and feedback that maybe due to the nature of lecture and how many students attend they otherwise would not be able to give. Our hope is that the website will become a helpful social environment as people can take out tasks set by the students and moderators in which they may provide advice or their own work in order to help the poster of that task and both students or staff would be rewarded for their contribution.

1.1 User Types

There will be two types of users on the website Students and Moderators:

- 1. A student can create a profile by supplying the required information at registration. They can also edit the information on their profile and will be able to create and delete tasks, claim, comment, rate, and flag other student's tasks. End users will also rate other users with the website's reputation system, if they achieve a reputation score of 40 or over they will be promoted to a moderator.*
- 2. A moderator will have all the options an end user and more so, Moderators will be able to Ban Students, withdraw tasks deemed inappropriate and view tasks flagged by students.*

1.2 Justification of Functionality

The functionality afforded to users of the proposed site will be provided in a few simple operations. Users will be able to log in using two text boxes, one for email and the second for a password. Once both fields have been completed a sign in button will be clicked granting the user access to the system if the supplied information is correct. Should a user not have an account to access the system, he/she can click the register button to be redirected to the Registration page and create an account.

*Once logged on users can choose to search what type of resources they wish to see, in a similar way that ProDirectSoccer.com affords users to do so ******(see figure 10 in Appendix). When the user clicks on an option from the page list of all the resources related to the selection will be generated in a comparable fashion to the lists on Sharepoint.ul.ie ******(see figure 5 Appendix). Users can also use a search box to input their own parameters. One can see from figure 2 that each file is assigned a row with the file title, the date published and the author of the assignment/theses. Also, clicking on an author's name will direct the user to that profile, where all the work published from that profile can be found.*

1.3 Potential Ramifications

The potential ramifications of our site allow for graduate students or any other persons under undertaking a final year project/thesis etc. to have their work proof read by a wider intellectual community. This will hopefully quicken up the process of delivering the final drafts of a person's work, give a closer look in detail into their project, open and expand the potential changes that can be made on the topic concentrated on and allow for the possible spread of a person's work across a larger base. The website allows for contact between the students and moderators. This should enable deadlines to be set and specifications to be taken into consideration while proof reading. Tags attached to a created task should distinguish the characteristics of a person's submission. If a deadline is met a user will be awarded or deducted 10 points to their account, after 40 points has been achieved a user is upgraded to moderator status. Further so if a user is happy with the result they can award or deduct 5 point from another user. Moderators hold the ability to flag a task and remove it if perceived to be unsuitable therefore allowing the ban of a user. Once a task has been claimed it is removed from the list of readable tasks and once a task has been completed it is removed entirely from the system. The deviation from previous methods of proof reading a person's work may result in the isolation of a offline community separating new and old methods. This could be justified by ease of use of the websites services but may not solve the problem of losing potential methods and persons. The points systems may also distance potentially new members who are unsure of the websites methods of operation.

2.0 Detailed Description

When the website is launched a homepage similar to figure 1 (Appendix) will be displayed. The end user will not be able to see what is inside the website until they have either logged in or registered. Cookies will be used for returning users as logging in every time would become an inconvenience. (Flow chart, Figure 7 in appendix)

Registration is executed when a student wishes to register themselves. Students will supply information to a webform on the homepage and submit it (see figure 1 in the appendix). The input is validated and if it is correct the information is passed to the database and the new entries are added in the tables and the user is directed to their new profile page (see figure 2 in the Appendix). If incorrect information is supplied the appropriate error message is displayed. E.g. the incorrect fields will be highlighted.

A login occurs when a registered user wishes to login they supply their email and password (see figure 1 in appendix). This process is executed when these details are submitted at the login area on the homepage. The details are validated against the details in the database. If the user is authentic the process directs them to their profile page. If the supplied details were incorrect an appropriate error message will be display informing the user that their details were incorrect and to re-enter them.

Create task occurs when a signed in user wishes to publish his/her project for proofreading. The student will fill out a webform and upload preview pdfs or other documents (similar to figure 3). Once completed the task will be inserted into the unclaimed table and other students and moderators will see it when browsing through the list of unclaimed tasks. (Flow chart, figure 8 in appendix)

Flagging tasks will be a simple button click with a confirm click after similar to (figure 4 in appendix). Once confirmed the task is inserted into a table only viewable by moderators and its faith with be determined by the moderators. (Flow chart, Figure 9 In appendix)

Claiming tasks will function similar to flagging tasks in that it will simply be a button click and the task will be removed from the list of unclaimed tasks and added to a table of claimed tasks the user claiming the proofread will be able to request the full document from the task creator and the claimant will have up until the tasks deadline to submit his/her

recommendations or cancel the task. Finally, if the claimant submits on time the task creator will give him a “thumbs up/thumps down” rating granting or deducting points from the claimant (see figure 4 In appendix).

(Flow chart, Figure 9 In appendix)

Student profiles will have a reputation system which will be stored in the user database, depending the Student’s actions, his/her reputation score will go up or down. This will be accomplished by checking if they miss deadlines or they get poor ratings etc. Students will also be able to edit their profile similar to figure 6 (Appendix).

Browsing tasks will be similar to figure 5 in(Appendix) in that tasks whose deadlines are closest will be sorted to the top simply by working out the days between the date of browsing to the deadline date and sorting this via a “sort-by” method. The browser will preference tasks that relate more so to the student’s field. The tasks shown will be taken from the unclaimed task table.

Moderators will be able to take down inappropriate tasks by using a delete feature. They will also be able to ban students and the student’s email address will be “blacklisted” on a table and will not be able to reregister with that email. Finally, moderators will be able to view flagged tasks in the similar fashion as a student would browse unclaimed tasks.

3.0 Technologies

Several languages and technologies will be utilised during the development of this project.

- Microsoft Windows 10 Operating System will be used as the operating system for developing the proposed website.

- Notepad++ will be used to Develop the website

- PHP will be used to generate dynamic webpages.

- HTML5 & CSS3 are languages used to generate the appearance of webpages. These languages are used together. HTML5 consists of tags which dictate where information appears on a webpage. CSS3 is used to assign styles to these tags such as colour, font, positon, margins etc. Both these languages will be used to design the appearances of the webpages.

- JavaScript will be used to add functionally and effects to the webpages. It is and object orientated language. JavaScript has the ability to validate data or produce an event on a click. It will be used as deemed fit throughout the project.

- MySQL is a relational database management system. It will be used as the primary means of creating a database for the proposed website.

- XAMPP and 000webhost will be used to test the website and iron out any problems. It is a simple, lightweight Apache distribution that makes it extremely easy for us to create a local web server for testing purposes. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server extremely easy as well. 000webhost is Server hosting site that allows us to test the website on an actual server along with XAMPP

- Gitbucket (<https://github.com/willleary6/CS4014-Project>) main version control repository.

Appendix

Figure 1

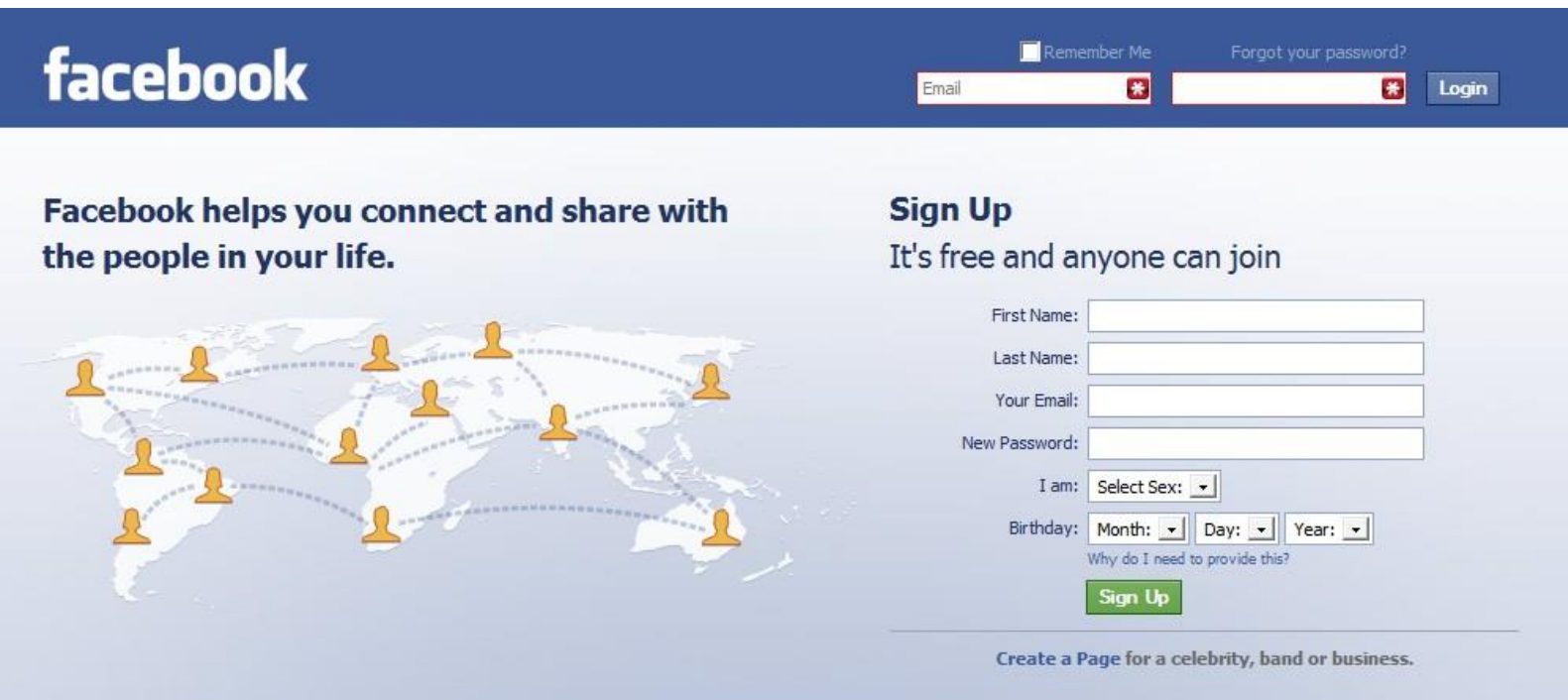


Figure 2

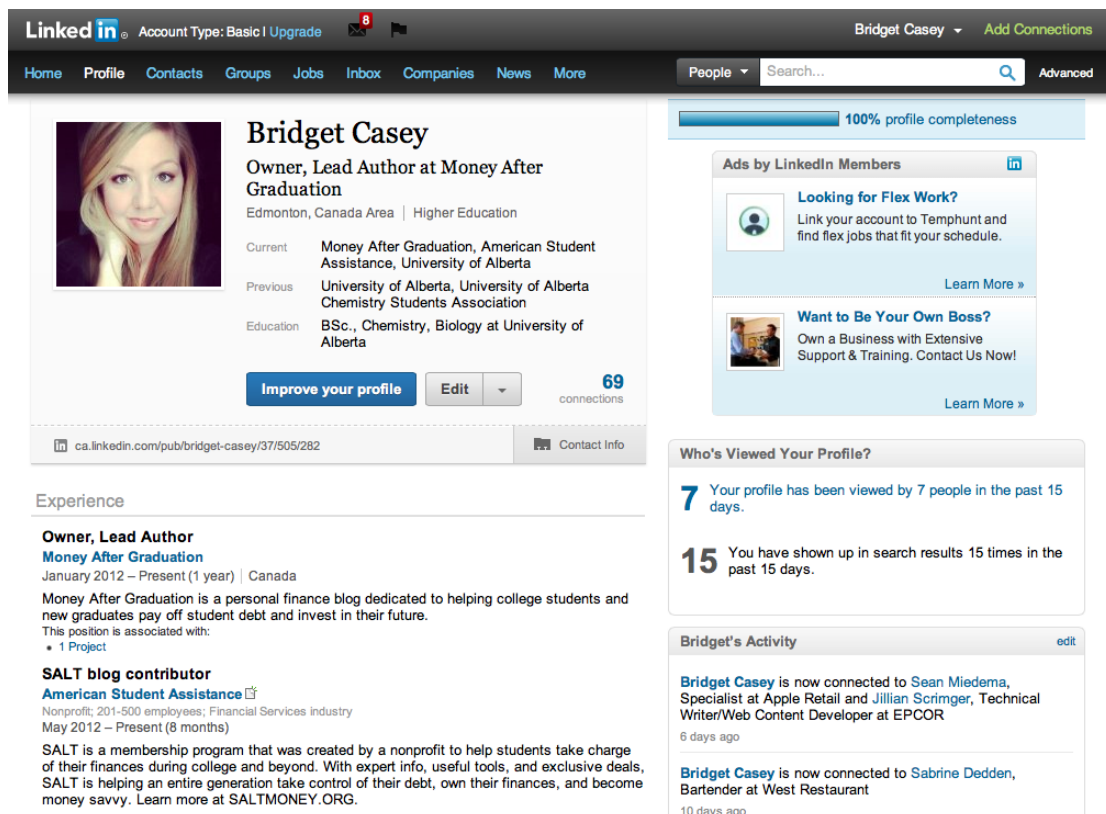




Figure 3

When? Today  2:30 pm  [Add end time](#)

What are you planning?

Where?

[Add street address](#)


More info?



Who's invited? [Select Guests](#)

- ☒ Invite Members of the host group **Front Range Blogger Meetup: The Facebook Group**
- ☒ Anyone can view and RSVP (public event)
- ☒ Show the guest list on the event page



[Create Event](#)

Figure 4

 **How to pick an avatar on YouTube**

 **YouTube Help**  [Subscribe](#) 992,215

55,301

[+ Add to](#) [Share](#) [... More](#)  328  189

Published on Aug 28, 2016
Learn how to change your profile picture on YouTube.

[Report](#) [Transcript](#) [Statistics](#)

[SHOW MORE](#)

Figure 5

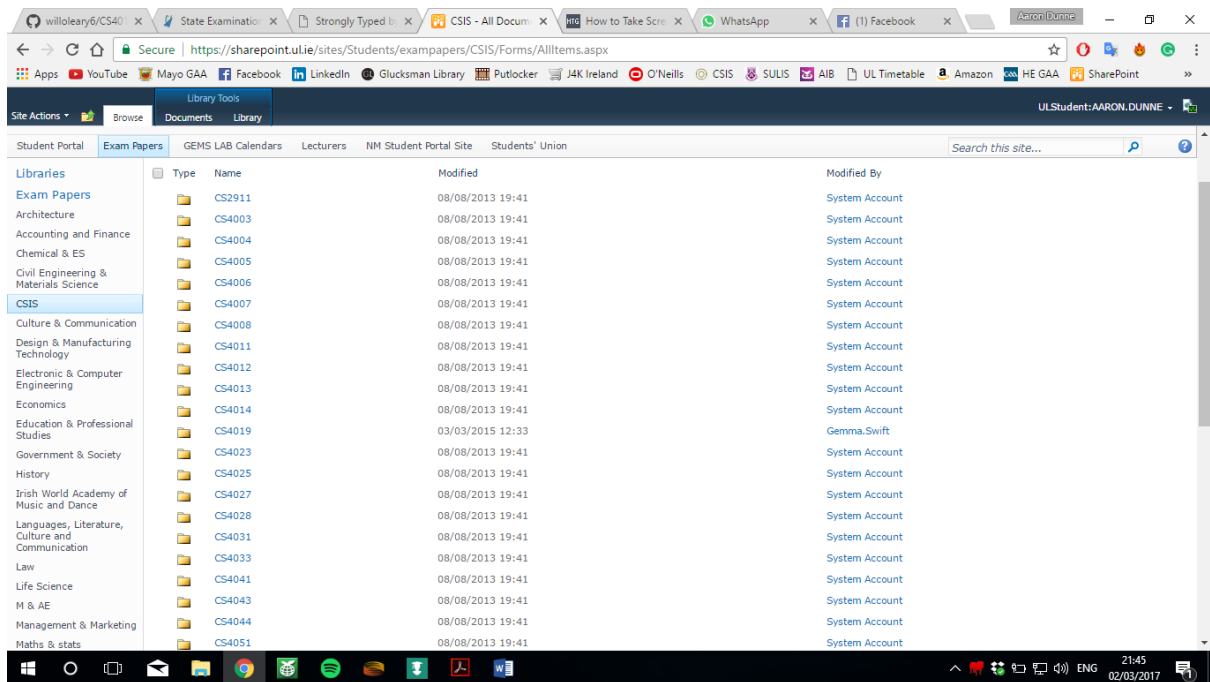


Figure 6

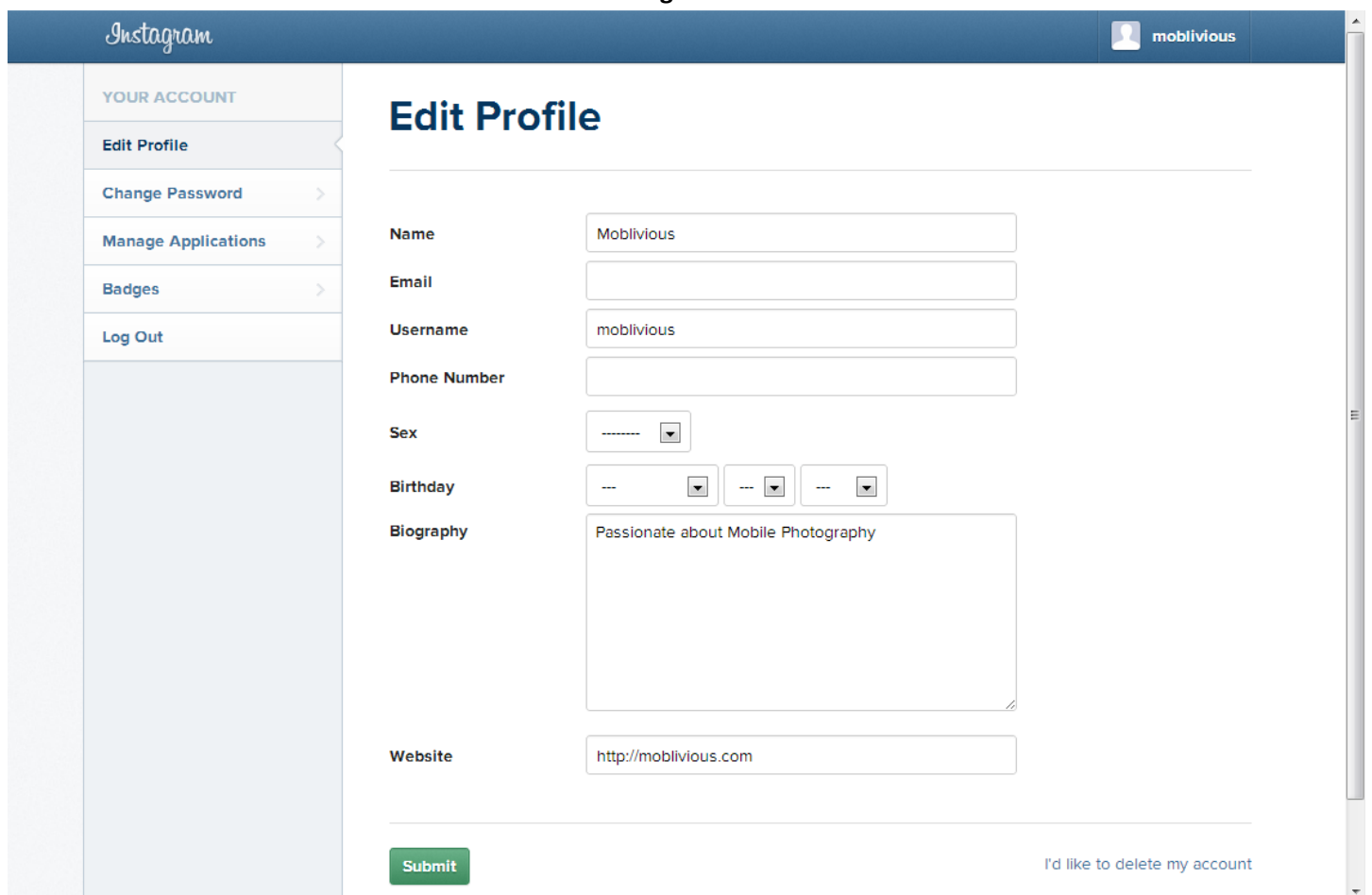


Figure 7

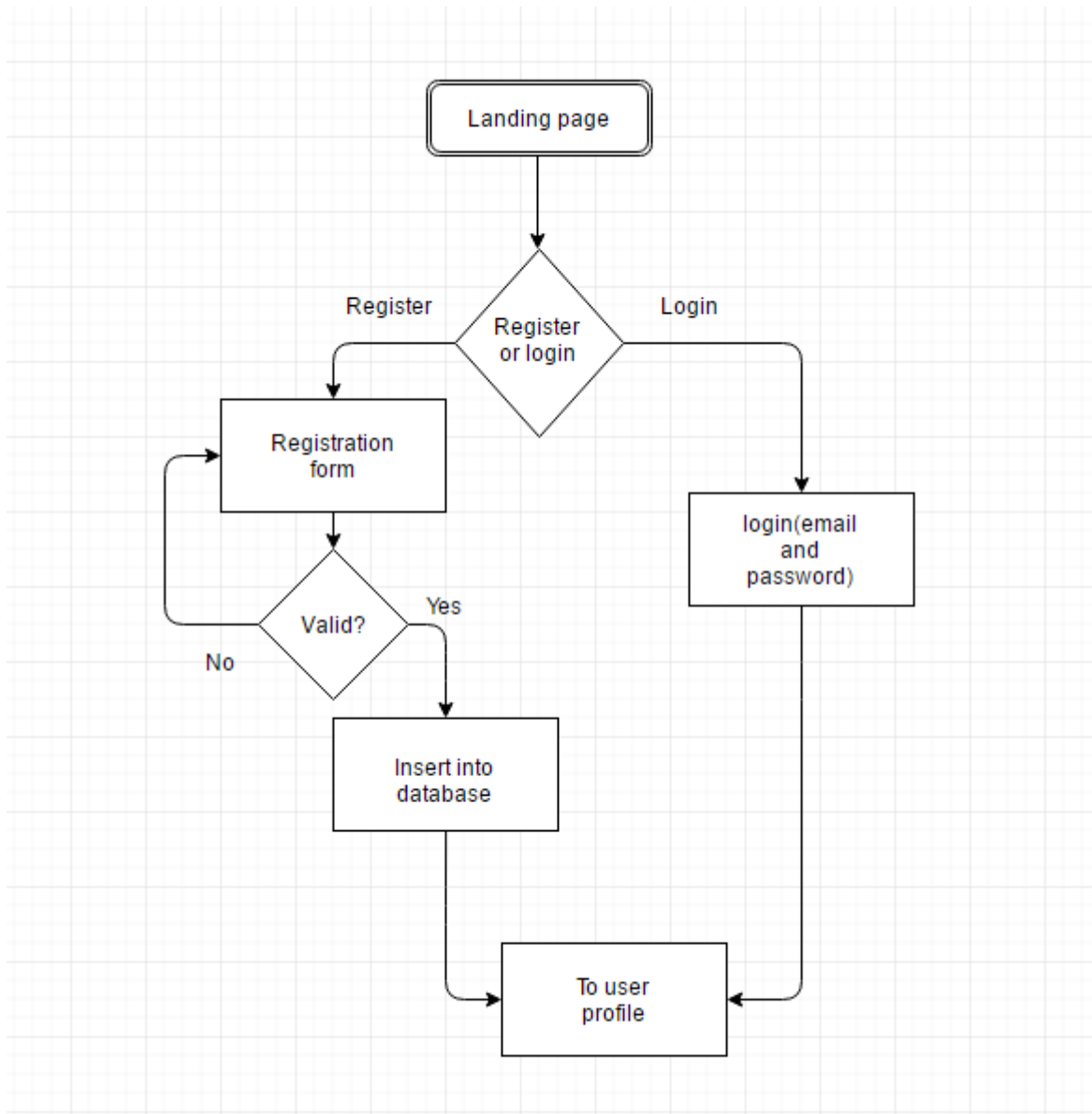


Figure 8

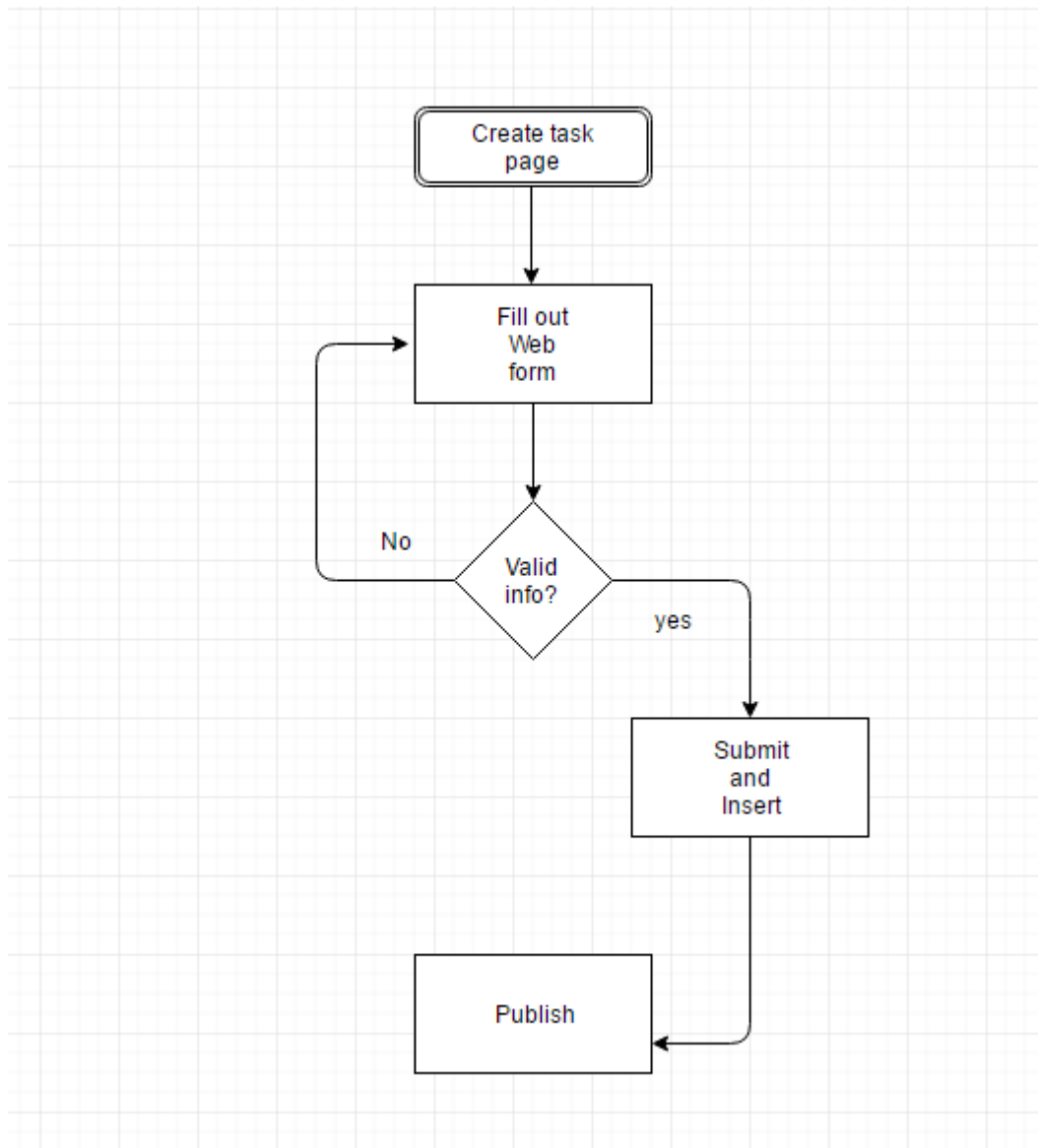


Figure 9

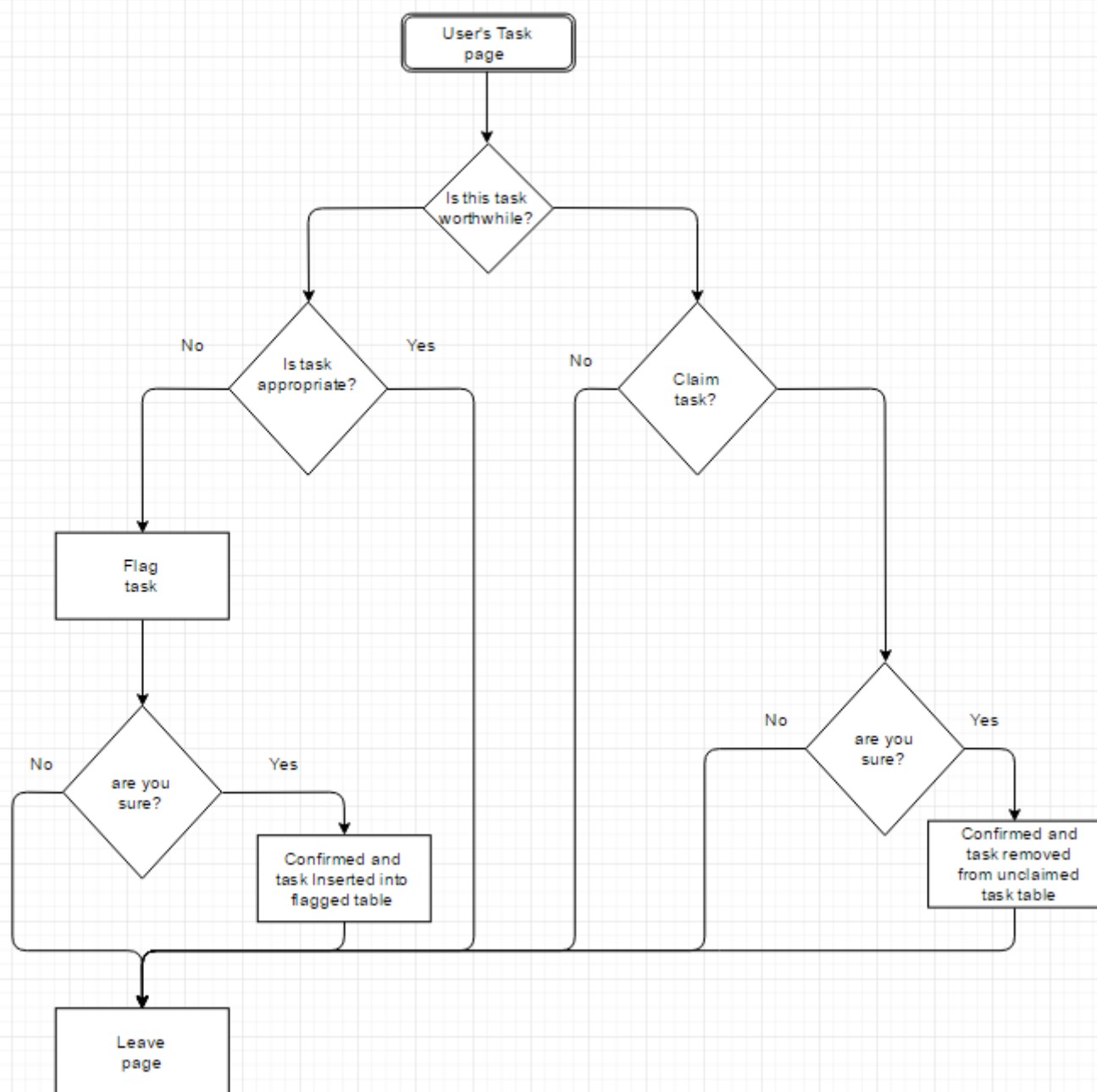
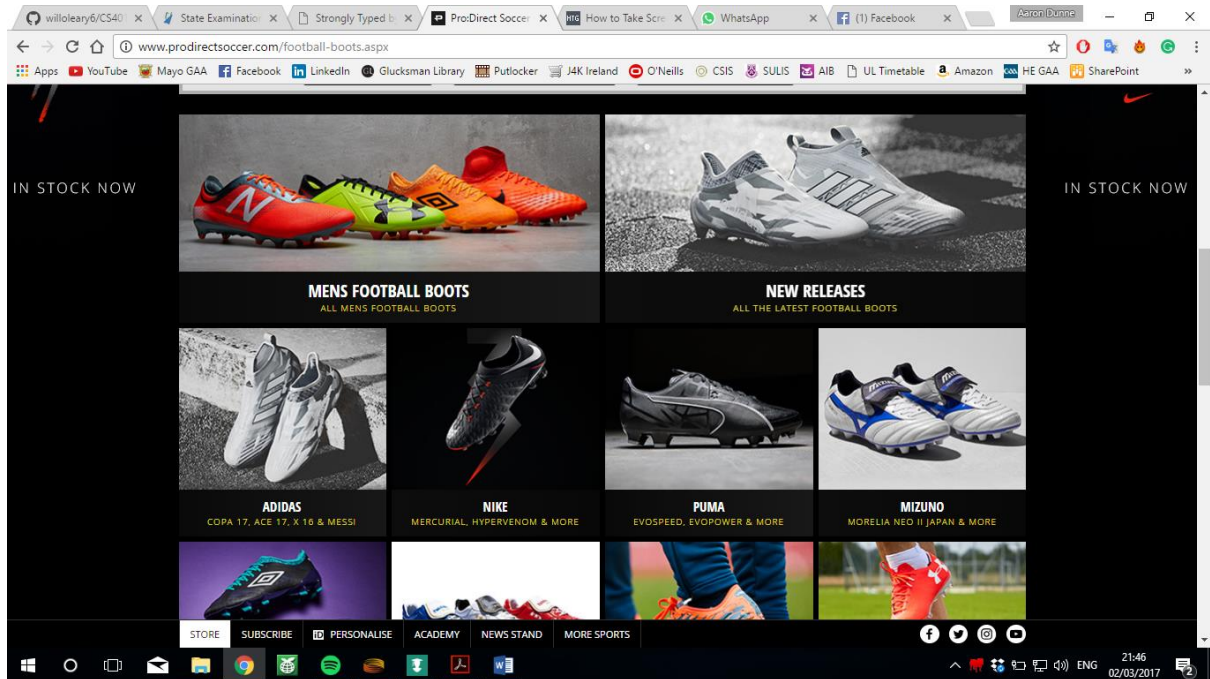


Figure 10



List of processes

Process Title	<i>Registration</i>
Brief Description	<i>Students wishing to join and view the website will fill out a webform with their information and Submit said information which is then stored in the database, granting them access and a student profile.</i>
inputs	<i>Student's information such as email, ID number etc.</i>
Detailed description	<i>When a new student visits the website he/she must register to gain access to the rest of the website, they will be required to submit their name, id number, email, and a password. On clicking submit the Students information will be handled by a php program and Inserted into the Student details table. The student will then be taken to his/her user profile.</i>
Output	<i>Creates Student profile and redirects student too it</i>

Process Title	<i>Login</i>
Brief Description	<i>Returning Students will fill out a small form which includes their email and password to be verified by the system and granted access to their account</i>
inputs	<i>Student's email and password</i>
Detailed description	<i>Once a student submits his/her information php will be used to input a sql query to the database checking if the information provided is legitimate, once confirmed the user will be redirected to their profile page.</i>
Output	<i>Grants user access back to his account.</i>

Process Title	<i>Create task</i>
Brief Description	<i>Students will create tasks to be proofread by filling out a form similar to figure 3(appendix) and this will be inserted into the unclaimed task table and viewable to all other students.</i>
inputs	<i>Information about the task such as preview documents and text.</i>
Detailed description	<i>When a student wants to create a new task he/she will click a button bringing them to the create task page where they will be presented with a web form requiring information such as a title, Description, Deadline etc. once all information is filled out the website will employ a php program to take that information and insert it into the unclaimed tasks table and students browsing unclaimed task will see this task on the browsing page.</i>
Output	<i>Publishes a student's task to the website for proofreading by other students.</i>

Process Title	<i>Claim task</i>
Brief Description	<i>When a student see's a task he/she wishes to claim, he will click a button followed by a confirm button to claim the task, this will remove the task from the unclaimed tasks list and he will be able to give notes, comments etc. to tasks owner on how to improve their work</i>
inputs	<i>Two clicks by the student</i>
Detailed description	<i>When a student claims a task, php will be used to copy said task and insert it into a table which contains all the tasks information and the details of the claimant and the owner. The task will also be deleted from the unclaimed task table as not too have two or more claimants for a single task. Claimants will have a section in their user profile page showing claimed tasks with features to write notes and comments to the owner. Similarly owners will also have a section in their user profile where they can receive claimants notes and comments.</i>
Output	<i>Allows students to claim task's and give notes on said tasks to their owners.</i>

Process Title	<i>Flag task</i>
Brief Description	<i>When a Student see's a task he/she feels is inappropriate they can click a button similar to figure 4 (appendix) and this task will be inserted in a "flagged task" list viewable by moderators who can either unpublish the task or reinstate it.</i>
inputs	<i>Simple click and confirm operation from a student</i>
Detailed description	<i>Once the flag button has been confirmed php will be used to copy its information and insert it into a purgatory of sorts in "flagged task" table which moderators will have a an option of viewing and deciding its faith from, either by deleting it entirely or reinstating it in the unclaimed task page.</i>
Output	<i>Allows the community to police what tasks are published on the site.</i>

Process Title	<i>Rate claimant</i>
Brief Description	<i>After a claimant's proofreading submission, the task owner will be asked to give the claimant either a "thumbs up" or "thumbs down" which will either increase or deduct 5 points from the claimant's reputation score.</i>
inputs	<i>Option of one of two buttons to click by the owner of the task</i>
Detailed description	<i>Once an option is clicked, php will be used to edit the user's reputation score held in the user_details table either adding or subtracting 5 points from that score.</i>
Output	<i>Allows claimants to build a reputation as competent proof-readers.</i>

Process Title	<i>Promotion to moderator</i>
Brief Description	<i>Once a student hits 40 reputation points or over they will be promoted to moderator of the site.</i>
inputs	<i>Wholesome efforts on the website helping other users with their task's resulting in a promotion.</i>
Detailed description	<i>Every time a student completes a claim and is rates by a task's owner a small php program will be used to check if his score is high enough (40 point or over) to be promoted to a moderator, if the case is correct the user will be promoted and his details will be edited on the user_details table changing his user-type to moderator.</i>
Output	<i>Allows exemplary members of the community to have a larger role in the website.</i>

Process Title	<i>Banning users</i>
Brief Description	<i>When a Student is deemed too inappropriate or abusive on the website moderators can ban him from the site.</i>
inputs	<i>Click of a button only available to moderators.</i>
Detailed description	<i>Once the button has been clicked the banned user's details will be copied to a "black list" table in the data base and if he/she tries to register again they will not be allowed access.</i>
Output	<i>Allows moderators to remove abusive students from the website.</i>

Process Title	<i>Editing a user's profile</i>
Brief Description	<i>If a user wished to edit his/her information on the website they can fill out a form similar to figure 6 (appendix)</i>
inputs	<i>Filling out a webform allowing the user to edit his/her information.</i>
Detailed description	<i>If the user wishes to edit his/her information they will find an option on their profile allowing them to reenter their information. Php code will then use the edit function to change what ever details the user requested be edited</i>
Output	<i>Allows users to edit their information.</i>

Database Tables

User details

Field name	Data type	Example
User_id (primary)	Int	1
First_name	varchar	john
Last_name	varchar	jackson
Student/staff_id	int	1343456
email	varchar	johnjackson@gmail.com
major_subject	varchar	Computer science
password	varchar	Pass123
reputation_score	int	35
User_type	tinyint	1

Description: Contains all the details of a user.

Unclaimed tasks

Field name	Data type	Example
Task_id(primary)	int	43
owner_id	int	1
Title	varchar	Thesis on file compression
Text_description	varchar	"my thesis is about....."
Task_type	varchar	Phd thesis
Attached_files	BLOB	"previewOfThesis.pdf"
No. of pages	int	12
No. of words	int	10000
Deadline	datetime	"25 of November 2017 17:00:00"
tags	varchar	"file","compression"

Description: table with all the details of an unclaimed task.

claimed tasks

Field name	Data type	Example
Task_id(primary)	int	43
owner_id	int	1
Claimant_id	int	2
Title	varchar	Thesis on file compression
Text_description	varchar	"my thesis is about....."
Task_type	varchar	Phd thesis
Attached_files	BLOB	"previewOfThesis.pdf"
No. of pages	int	12
No. of words	int	10000
Deadline	datetime	"25 of November 2017 17:00:00"
tags	varchar	"file","compression"

Description: table with all the details of a claimed task, including the id of the claimant.

Flagged tasks

Field name	Data type	Example
Task_id(primary)	int	43
owner_id	int	1
Flaggers_id	int	3
Title	varchar	Thesis on file compression
Text_description	varchar	"my thesis is about....."
Task_type	varchar	Phd thesis
Attached_files	BLOB	"previewOfThesis.pdf"
No. of pages	int	12
No. of words	int	10000
Deadline	datetime	"25 of November 2017 17:00:00"
tags	Varchar	"file","compression"
Reason_for_flagging	Varchar	"Not relevant"

Description: table with all the details of a flagged task, including the id of the flagger and reason why it was flagged.

Black-Listed users

Field name	Data type	Example
User_id (primary)	Int	1
First_name	varchar	john
Last_name	varchar	jackson
Student/staff_id	int	1343456
email	varchar	johnjackson@gmail.com
major_subject	varchar	Computer science
password	varchar	Pass123
reputation_score	int	35
User_type	tinyint	1
Reason for banning	varchar	"Was not into dank memes"

Description: table with all the details of a banned user, including the reason for banning.