Design Documentation Deliverable:
Peer Review Website

Date	25/2/2017
Group Name/Number	Group 2

Student ID	Student Name	Signature
15147312	Jamie Mac Manus	
15178668	Rory Egan	

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**
- ❑ **Detailed Description**
- ❑ **Appendices:**
  - ❑ **Web Pages (mock ups of main pages)**
  - ❑ **Process Descriptions**
  - ❑ **Table Design**
- ❑ **Technologies**

1.0 Purpose of Website
When people enrolled in academic programs have to submit documents it is often times useful to receive a second opinion on work. This is especially true for large projects such as BSc dissertations or a PhD thesis which can weigh highly on a persons overall academic success. However, it is difficult enough to find a person who is willing to spend the time to read over the paper let alone find a person who is both willing and qualified.

Our peer review website provides a solution to this problem by providing a space online for people to both post and review projects. The process is made simple by using the website as the content is optimized to each user's preferences and area of expertise.

Users will be able to register for the website through the login page and by providing information such as their student ID, email and their area they can immediately start finding people to work with. First, the user will be redirected to the home page where they can browse tasks that have been created by other users. Here along with having already optimized content, they will have the option to add a series of filters to the content being displayed. At this stage they can start to navigate the site whether it be to view or edit their own profile or to search for a task based on its title.

1.1 User Types

1. Student:
The student is a registered member of the site that will be allowed to submit tasks, claim tasks and browse tasks. Furthermore they will be able to search for tasks with a specified title, edit their own profile and flag tasks that they feel are inappropriate. Students are encouraged to both post the tasks that they would like others to help them with as well as contribute by proof reading or reviewing the documents that others have posted.

2. Moderator:
The moderator is a student with a reputation of more than 40. They will be able to perform all the same actions as a student but they will also be able to view and remove flagged tasks as well as ban users.

1.2 Justification of Functionality
The functionality provided to regular users will be fairly basic. They will be granted the option to sign in via two form fields requiring their email address and password. Once these fields are completed the sign in button will grant users access to the main page if they are correct. From here users will be able to create new tasks, claim current tasks, search current tasks, filter current tasks and view claimed tasks. Tasks

will be displayed in a long list filtered by whatever the user has decided from a range of fields such as subject, type, date, availability etc.. Each task will be shown with title, basic task information such as subject, type etc. along with a short excerpt from the task description. Once a user clicks into a task they will be shown all task information and an option to claim the task if it is unclaimed. If the user is a moderator they will also be able to view flagged tasks, remove tasks, ban users and view currently banned users. The user can search tasks via the navbar and all tasks matching their search terms will be displayed.

1.3 Potential Ramifications

Potential ramifications of such a site on intended users/audience and how potential adverse consequences will be addressed in the proposed system.
…

The most obvious effect of creating this website is that people will be able to find suitable candidates to review their academic papers with much greater ease. If the user base were to grow large enough and users were considerate to community needs then finding a person to review your academic document would be made significantly easier by such a service. This is a result of the ubiquitous nature of the internet. Individuals are no longer restricted to relying on people close by to review their documents instead they can exploit the help of experts across the globe.

Having taken a positive outlook it seems this service would be greatly beneficial. However, there is the potential for a number of adverse consequences as a result of the very nature of such a system. The real potential downfall of this service is an issue of supply and demand. It's clear that there will be a greater demand for free proof reading and document review than there will be a supply of people willing to do this work. This issue could result in an the site simply being filled with tasks that nobody is willing to claim and complete. This will be solved in two ways. Firstly, by using a reward system where users are heavily rewarded for claiming and completing tasks. Users who perform these beneficial actions enough will become moderators and will have greater access and power than regular users. Secondly, by allowing a free and open trade of services. The website will have little to do with actual transaction itself and thus users can organise and amongst themselves the terms round which tasks are completed.

2.0 Detailed Description

Description of how proposed system will work. Include flowcharts to describe overall process; individual processed etc.
Referring to information in three Appendices - Processes, Web pages, Tables.

P1 and P2 (log in/sign up): The login/sign up page gives the user the options to either sign in with an existing account or else sign up to create a new account as shown in the page appendix. Sign up and log in functionality is achieved via php scripts interacting with the database and either verifying email/password or creating a new user entry in the database. User input is checked by Javascript as well as php to ensure a double layer of security and making user experience easier by informing them of errors in their entries before they try to submit the form. We have also set up a basic security screen for bots by setting up a honeypot. This is checked by both Javascript and php and if it is filled in the form will not be submitted. Upon successful creation of an account, the user is redirected to a thank you page, via the signup php script which informs the user that their account has been created. They are then free to log in with their user details. This will bring them to the main page.

P3 (browse + search tasks): The main page will show all current tasks in a list, which can be filtered by subject, type etc. via a php script. By default the tasks displayed by the home page to the user will be customised to the user. This will be achieved through data that is collected in the 'Click' table of the database being passed through a php algorithm.

P4 The navbar on the home page allows the user to navigate between all the pages as well as search tasks. Users enter keywords in the search bar and click the search icon which triggers a query to the database that finds titles that match the search term. All tasks with matches are displayed on screen in a search results page.

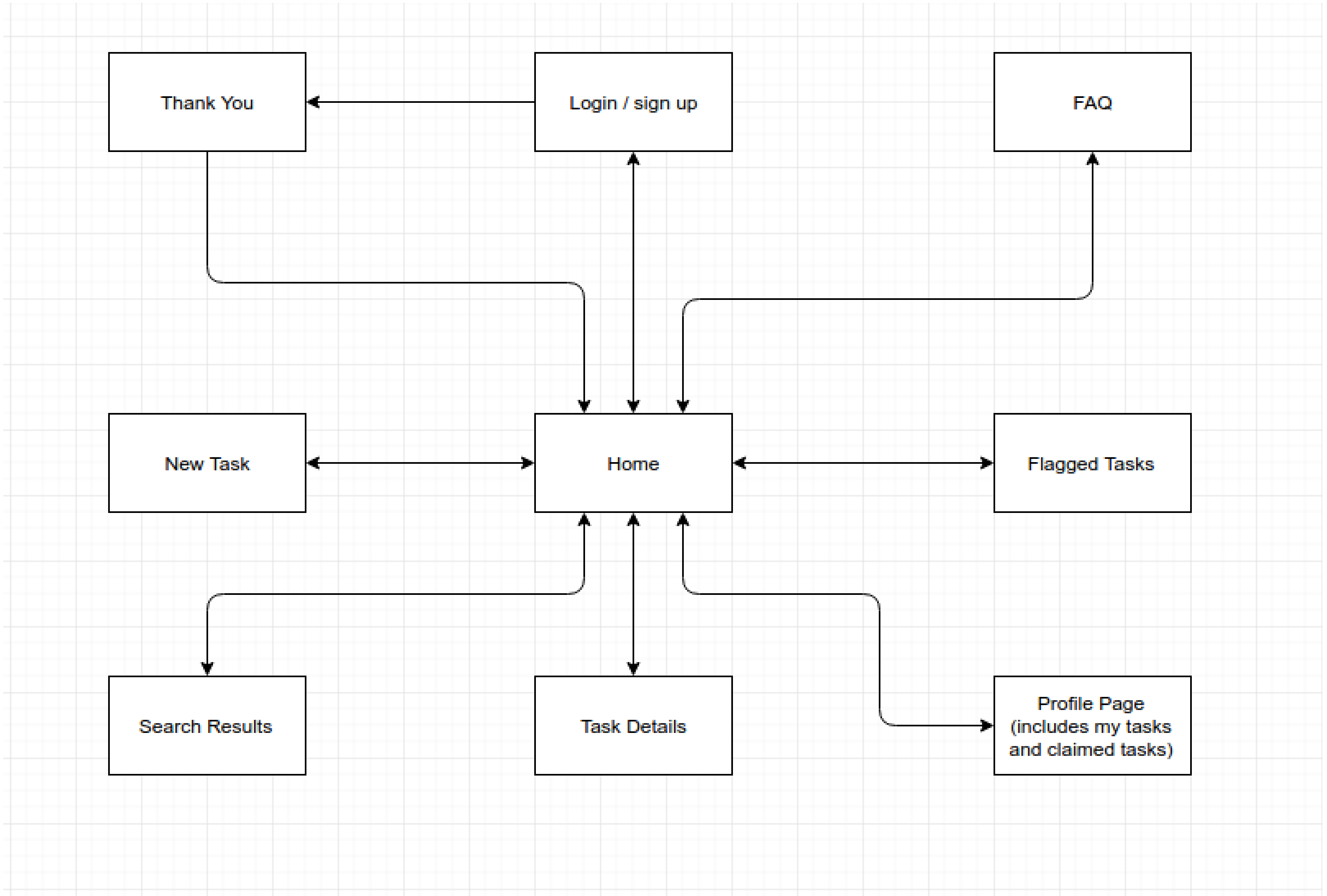
P5 + P6(flag + unpublish tasks): All users will have the option of flagging tasks when they view the Task Details page. Moderators will be able to view flagged tasks via a link in the navbar which will bring them to a page as shown in Appendices → Web Pages → Flagged Task Page. The moderator can also unpublish any tasks from this page. This will trigger a modal that requires a reason for this action and allows the moderator to confirm the action. Moderators can only unpublish tasks flagged by other users. This serves to limit their power.

P7(ban user): If the user is a moderator they will have the option of banning certain users. They will be able to click on a ban user button which will trigger a form modal requesting the user to ban along with a note about why the user was banned. The user will then have their details entered as a new entry in the banned users table in the database.

P8(claim task): Users will have the option to claim any currently unclaimed tasks. If they choose to claim a task, the task status will change from unclaimed to claimed, and the relevant row in the Task table will be updated to include the ID of the claimant. The claimant will then be able to see the task in the claimed tasks area of their profile page.

P9(new task): The new task creation is handled on the main page via clicking a new task button which pulls up a modal requiring the user to enter a form with the new task details. If the user enters any incorrect details they will receive an error message. These details are then submitted to the database via a php script and a new task is created in the database.

Page Navigation / process Overview:



4.0
Technologies

Ubuntu 16.04 LTS
This is the operating system that we will be running to complete this project.

Bootstrap
We are using Bootstrapas a framework on this project in order to make the frontend development easier through the use of all of Bootstraps predefined styles and classes. We are using a local installation installed through Bower.

Bower
We are using Bower, a package manager, to handle installations of project dependencies such as Bootstrap and Fontawesome. Dependencies are defined through the bower.json file and installed vi the command “bower install”.

Npm
We are also using npm as a package manager. It works similarly to Bower, wth dependencies listed in a package.json and installed with the command “npm install”. We are using it to install Grunt and any dependencies the grunt tasks might require.

Grunt
We are using Grunt, a Javascript task runner, in order to automate certain tasks via a Gruntfile.js. For example we are using grunt to watch and detect changes in the site styles, and create a new minified css fie automatically if any styles are modified.

PHP
This will be used for dynamic creation of content on the page and for interaction with the database. For example PHP will be used to load a list of subjects to choose from when users are registering for the site. In this process PHP queries the database for the list of subjects and then displays the subjects on the page through a dropdown menu.

HTML
This is used to provide structure to the content of the web pages. It allows us to differentiate between different types of text such as titles, headers and paragraphs.

Javascript
We are using Javascript in this project to carry out several client side features on the website, such as validation of form entries, honeypots, and carrying out changes to the site layout as the user takes certain actions eg. the navbar shrinks and changes colour as the user scrolls down the page. All Javascript is uglified to improve page loading time.

Jquery
We are using Jquery, a Javascript library in this project. It is the most used JavaScript library and is open source. JQuery allows for changes to be made to HTML pages in a similar way to JavaScript. The key difference is that JQuery requires much shorter and simplified pieces of code.

Sass
We are using sass, a CSS preprocessor for this project in order to access its superior functionality and usability compared to vanilla CSS. We are able to use nested styling, variables and can split our style sheets into many files easily to help with modularity of design and debugging. We are using Grunt tasks to create a minified CSS file from this which is then imported to our site.

CSS
This language is used to provide styling to HTML. Our CSS file is created from our sass files by a Grunt task. All of our CSS is then being minified to speed loading time by another Grunt task.

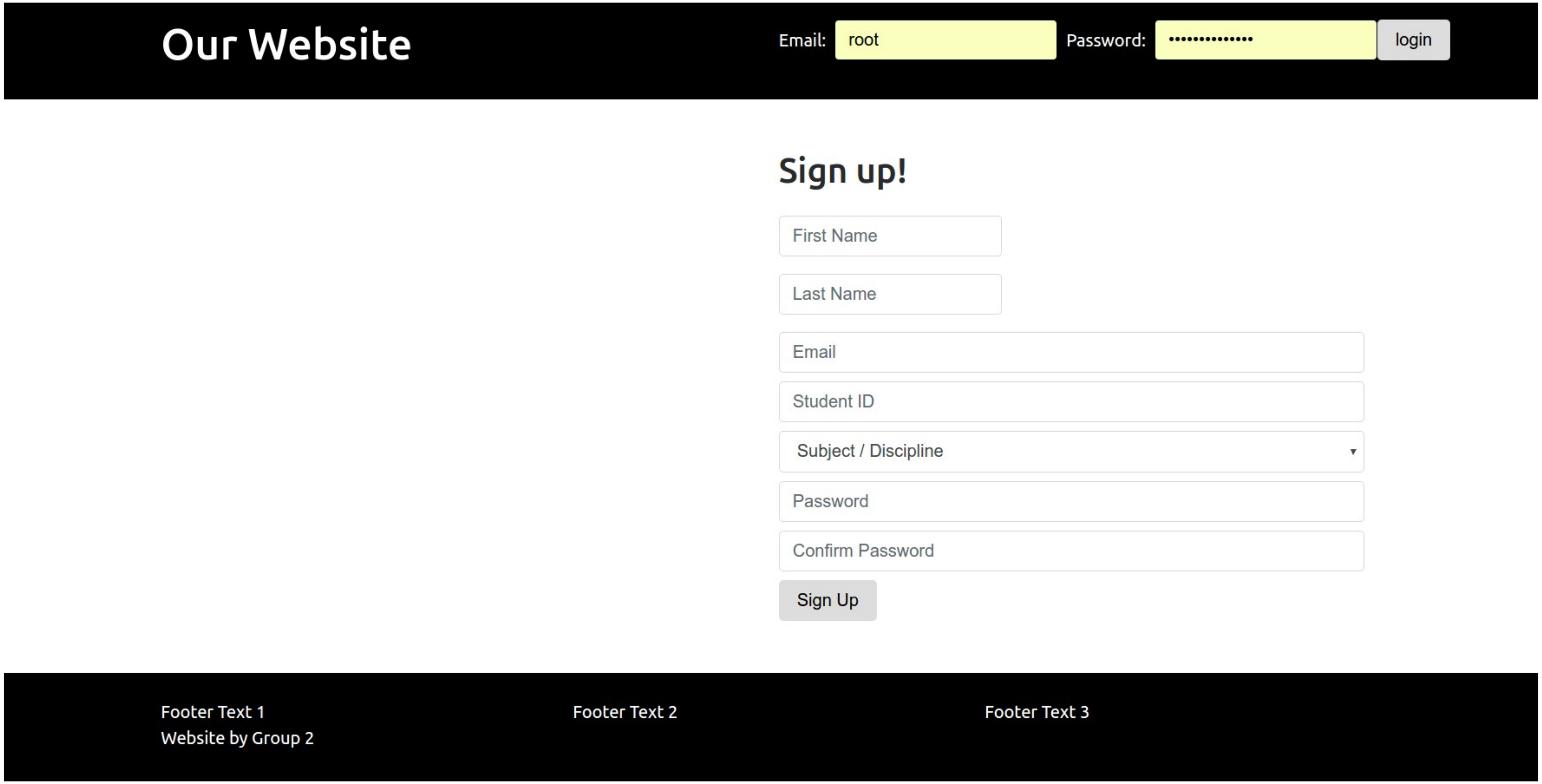
MySQL
This is a database management system that we will use to store all the crucial data belonging to the website. For example each user, and task and all their associated information will be stored using a MySQL database.

Apache
This is the web server we will be using to host our web page. (localhost).

GitHub
https://github.com/RoryEgan/cs4014-project/
GitHub is the version control system that we will be using to complete the project. Rory created a repository which Jamie forked.

Appendix 1: Web Pages

Login Page:



Our Website

Email: root

Password: ••••••••••••••

login

Sign up!

First Name

Last Name

Email

Student ID

Subject / Discipline ▾

Password

Confirm Password

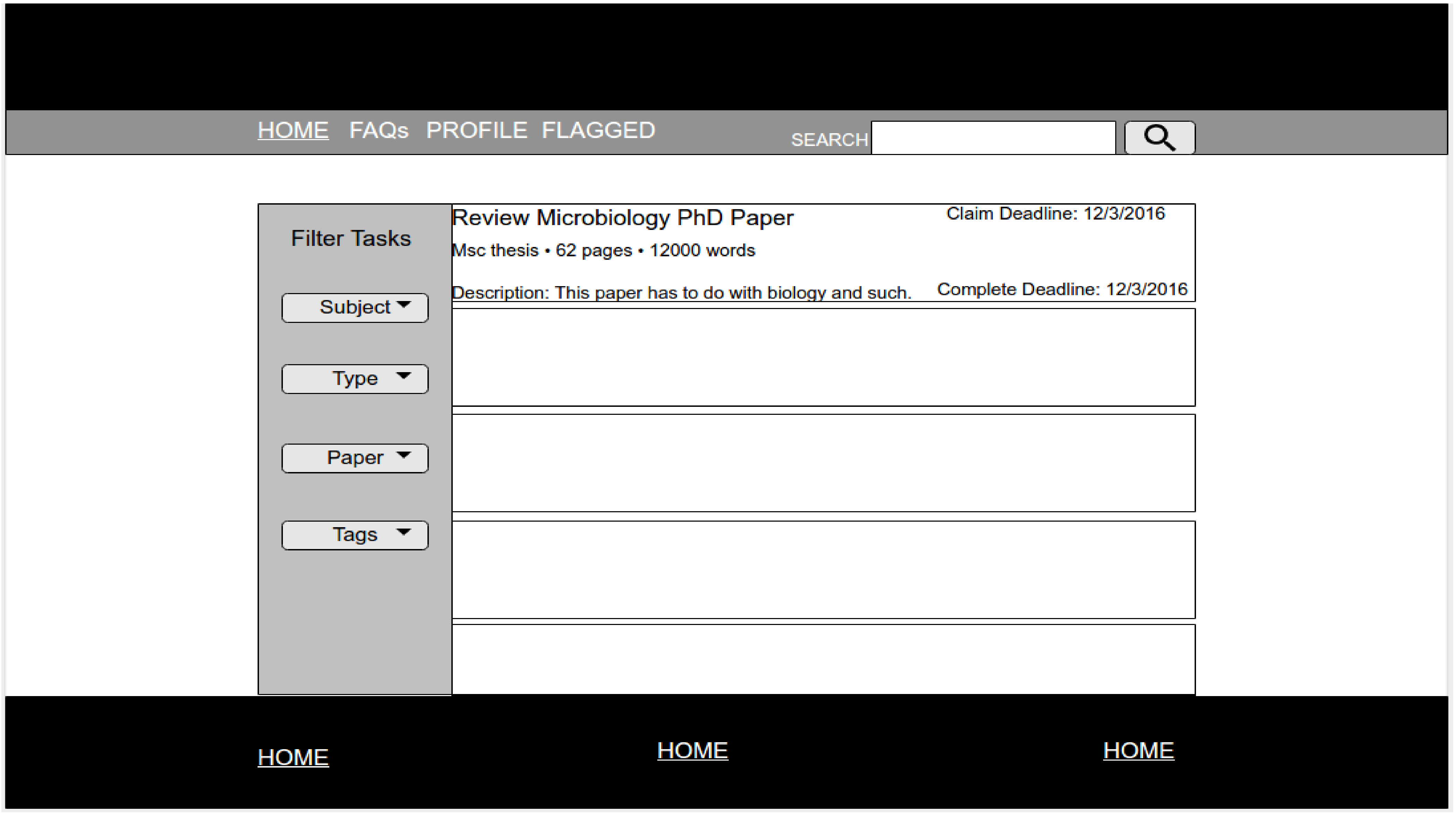
Sign Up

Footer Text 1
Website by Group 2

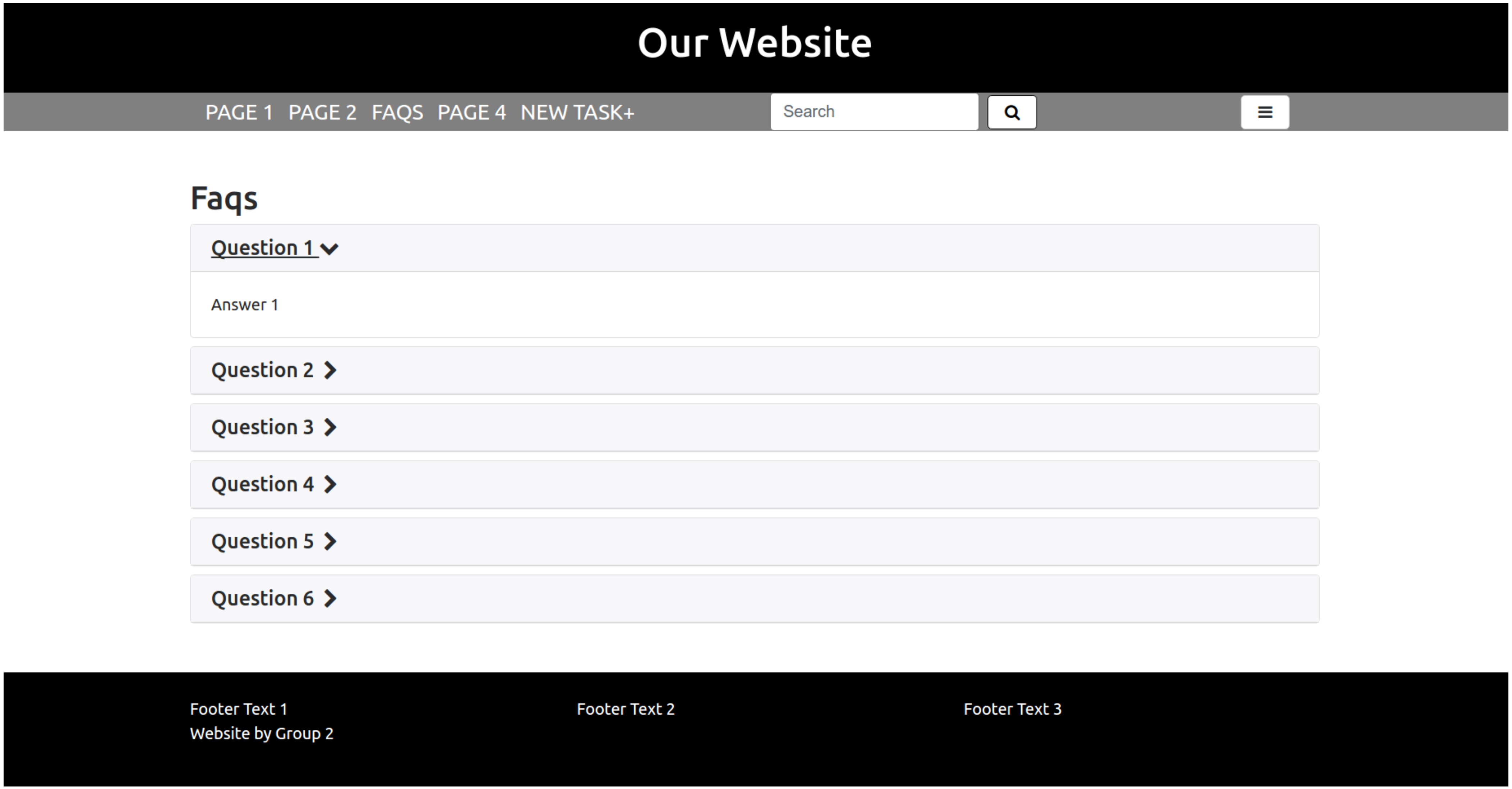
Footer Text 2

Footer Text 3

Home Page:


HOME
FAQs
PROFILE
FLAGGED
SEARCH
Filter Tasks
Subject
Type
Paper
Tags
Review Microbiology PhD Paper
Claim Deadline: 12/3/2016
Msc thesis • 62 pages • 12000 words
Description: This paper has to do with biology and such.
Complete Deadline: 12/3/2016
HOME
HOME
HOME

FAQs page



Our Website

PAGE 1 PAGE 2 FAQS PAGE 4 NEW TASK+

Search

Faqs

Question 1

Answer 1

Question 2

Question 3

Question 4

Question 5

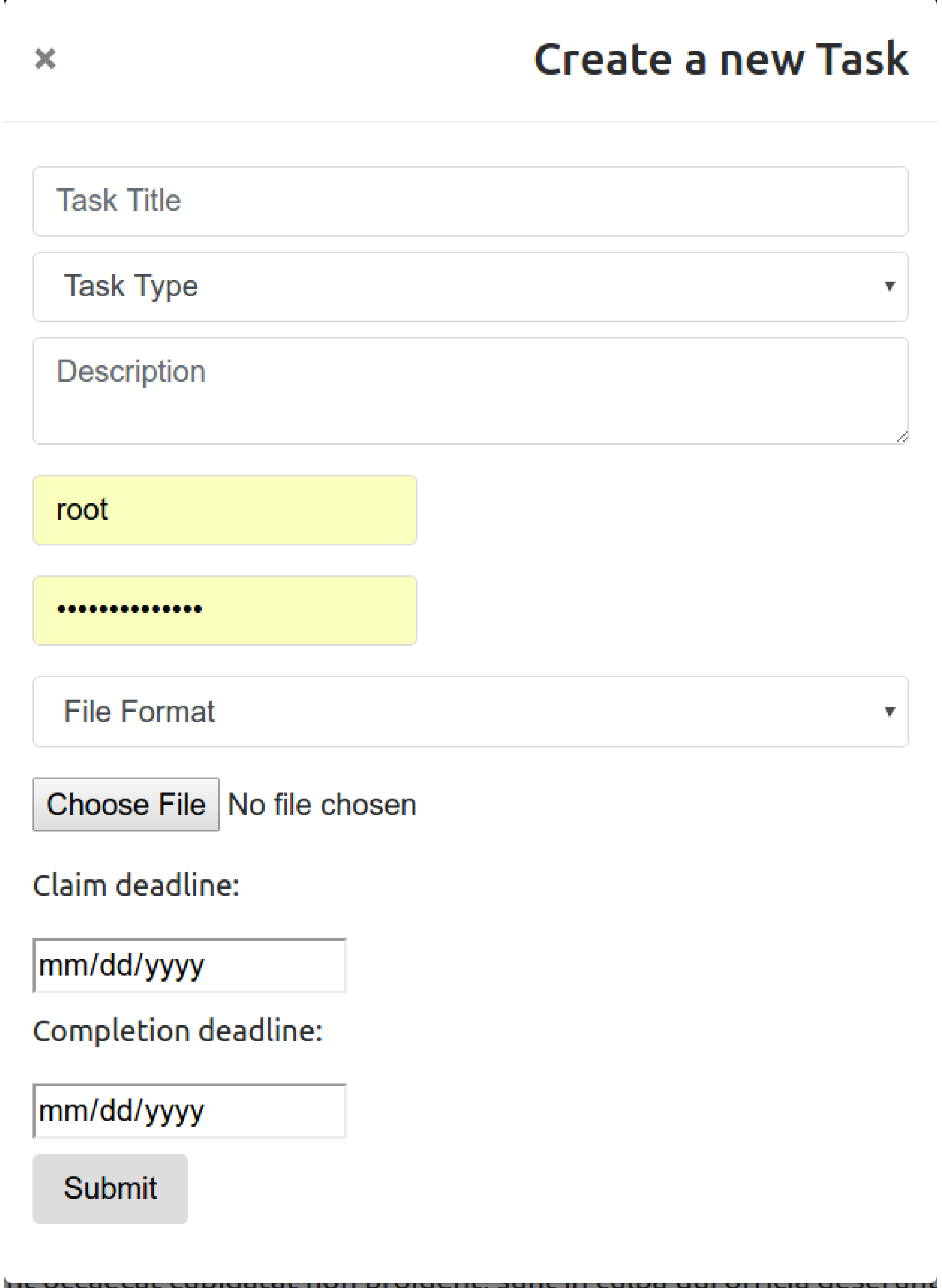
Question 6

Footer Text 1
Website by Group 2

Footer Text 2

Footer Text 3

New Task Modal:



× Create a new Task

Task Title

Task Type

Description

root

••••••••••••••

File Format

Choose File No file chosen

Claim deadline:

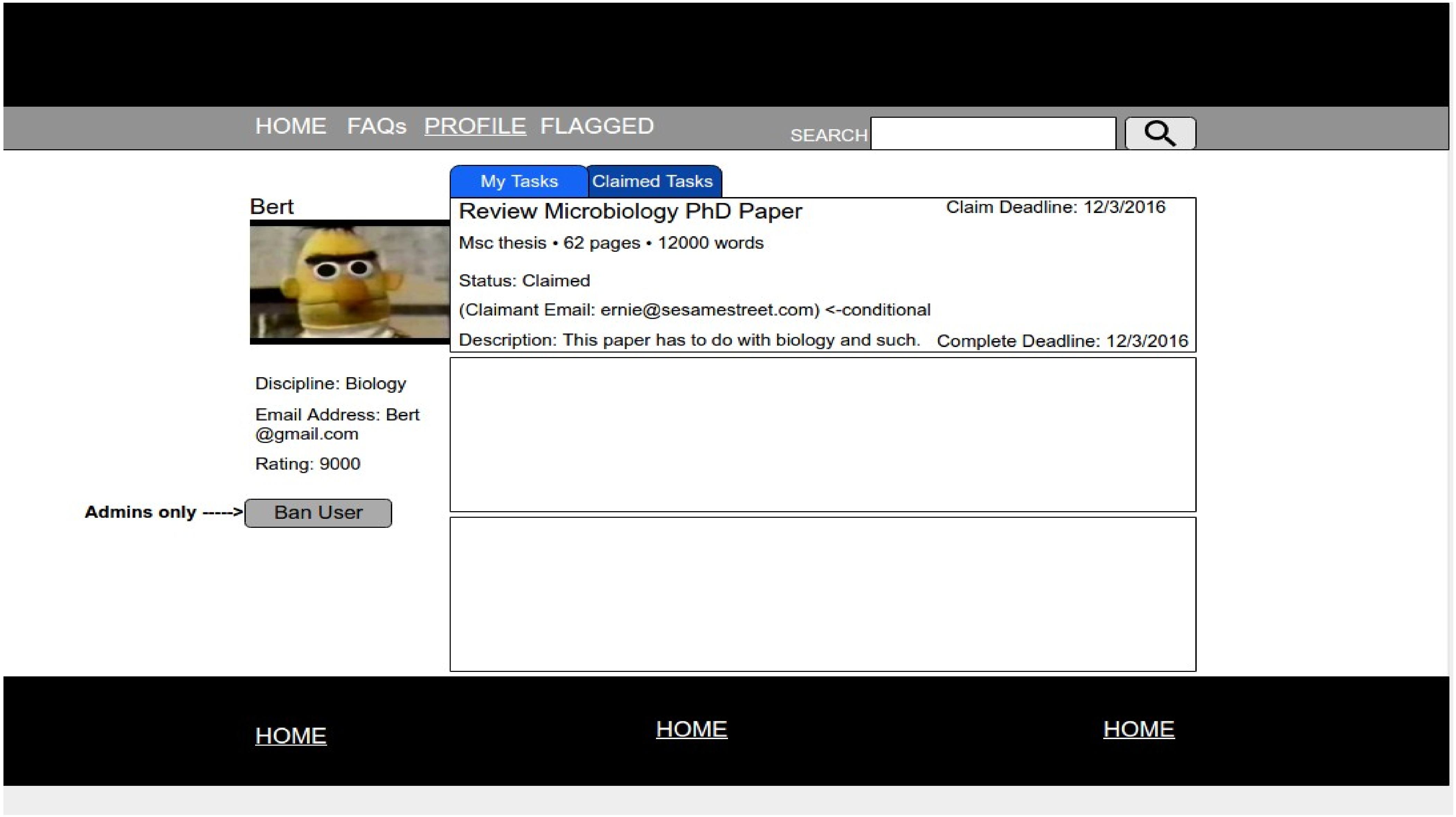
mm/dd/yyyy

Completion deadline:

mm/dd/yyyy

Submit

Profile Page:(my tasks selected)



HOME FAQs PROFILE FLAGGED SEARCH

My Tasks | Claimed Tasks

Bert

Review Microbiology PhD Paper

Claim Deadline: 12/3/2016

Msc thesis • 62 pages • 12000 words

Status: Claimed

(Claimant Email: ernie@sesamestreet.com) <-conditional

Description: This paper has to do with biology and such. Complete Deadline: 12/3/2016

Discipline: Biology

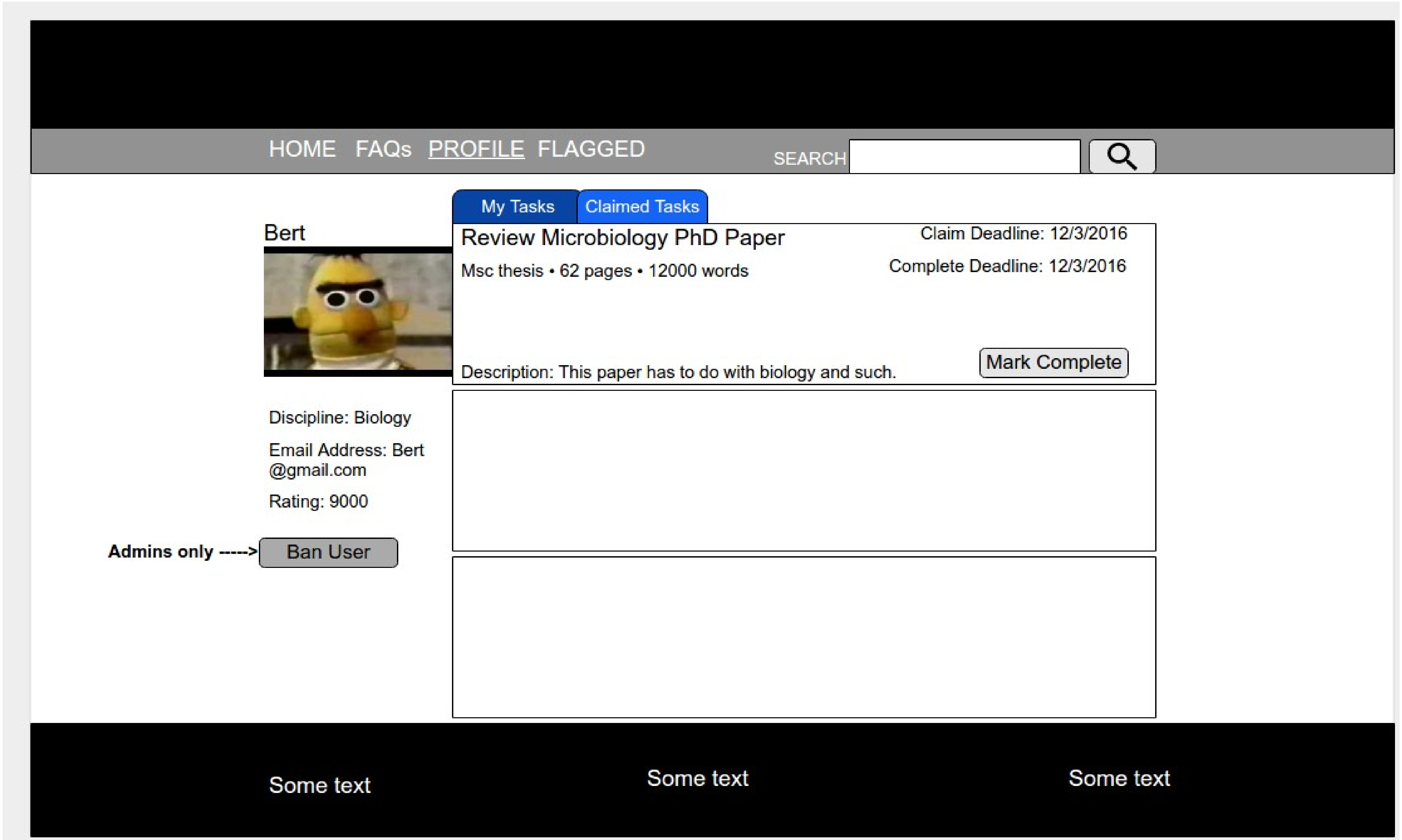
Email Address: Bert @gmail.com

Rating: 9000

Admins only -----> Ban User

HOME HOME HOME

Profile Page: (claimed tasks selected)



HOME FAQs PROFILE FLAGGED SEARCH

My Tasks | Claimed Tasks

Bert

Discipline: Biology

Email Address: Bert @gmail.com

Rating: 9000

Admins only -----> Ban User

Review Microbiology PhD Paper

Claim Deadline: 12/3/2016

Msc thesis • 62 pages • 12000 words

Complete Deadline: 12/3/2016

Mark Complete

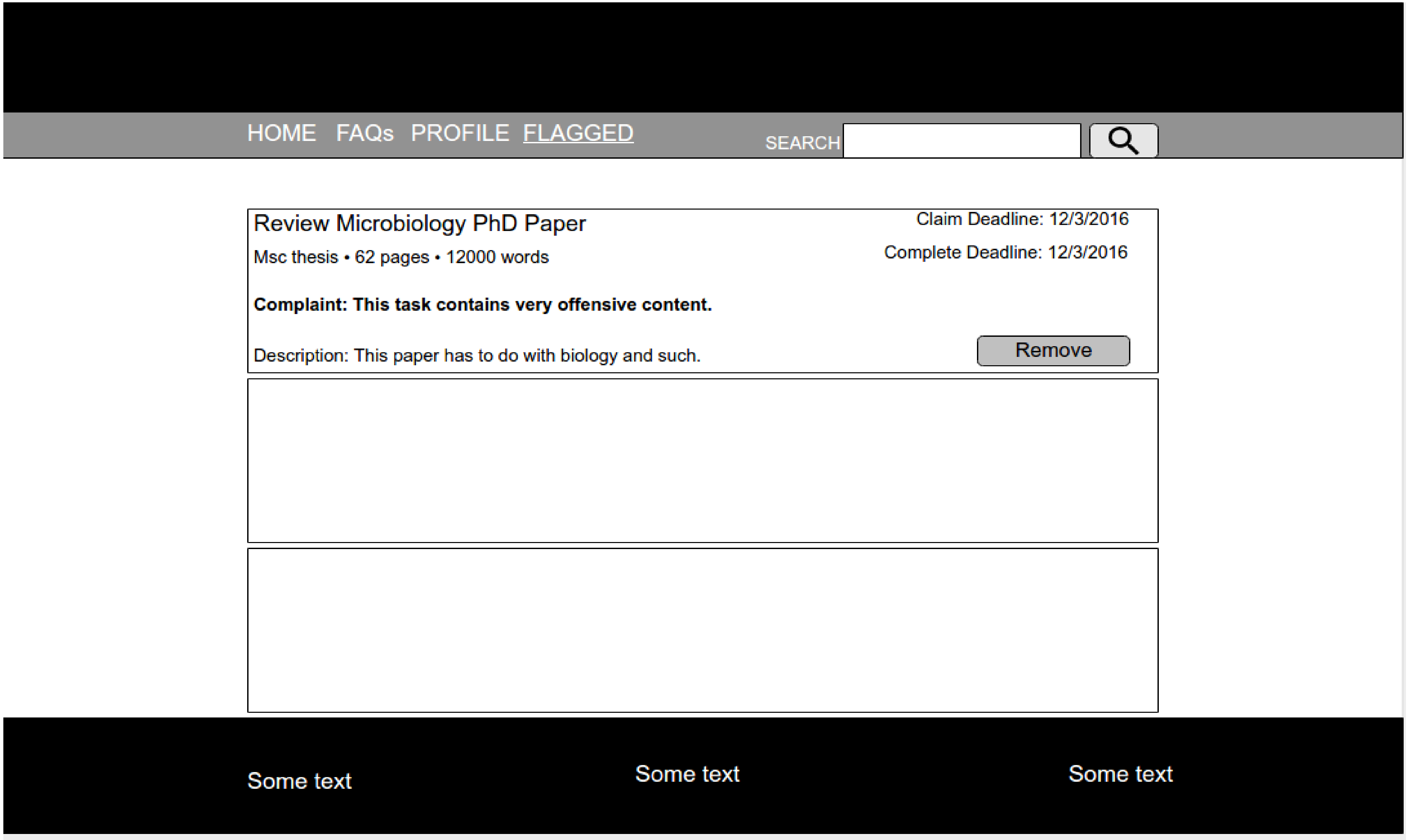
Description: This paper has to do with biology and such.

Some text

Some text

Some text

Flagged Tasks:



Review Microbiology PhD Paper

Msc thesis • 62 pages • 12000 words

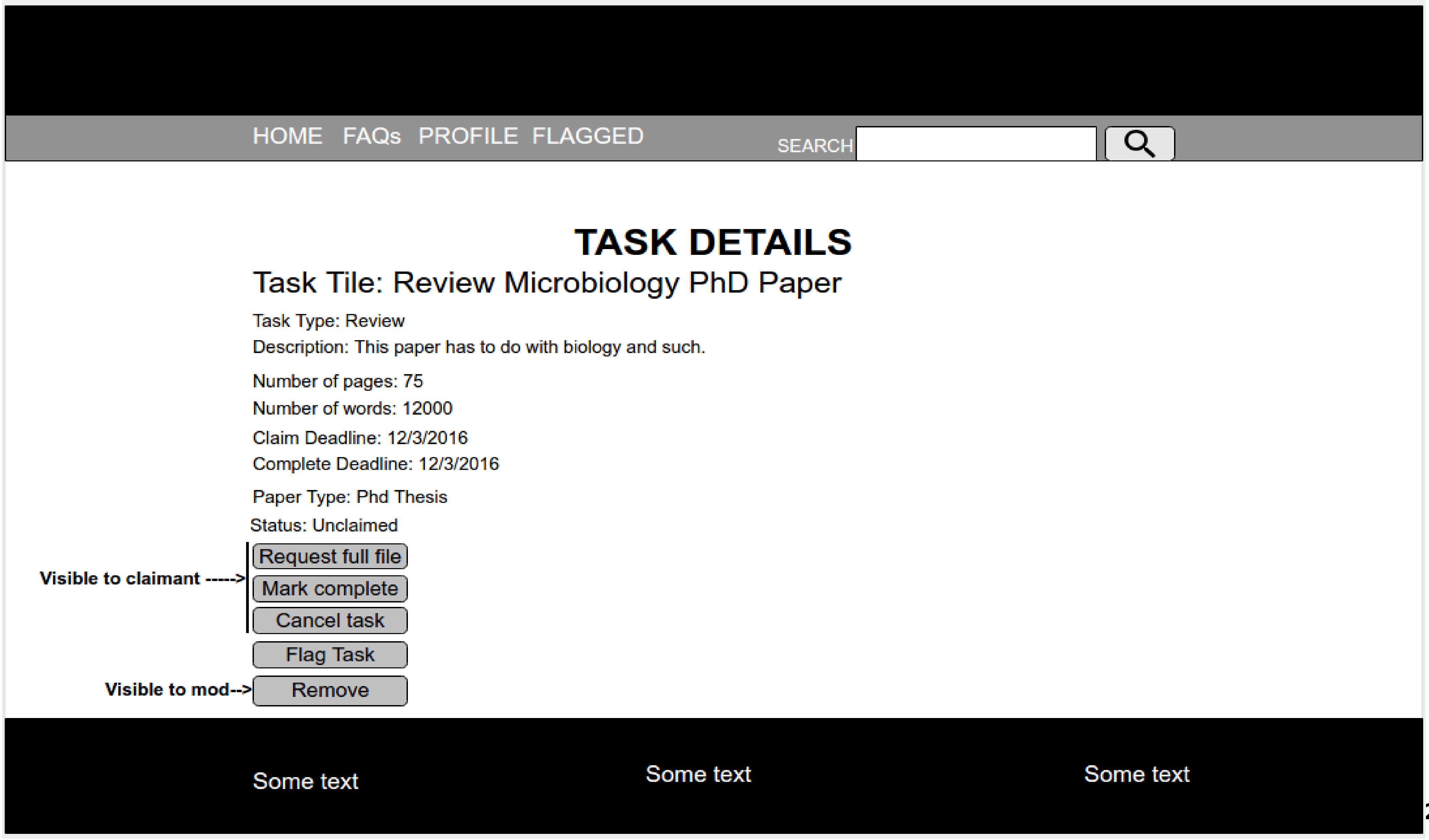
Claim Deadline: 12/3/2016

Complete Deadline: 12/3/2016

Complaint: This task contains very offensive content.

Description: This paper has to do with biology and such.

Task Details Page:



TASK DETAILS

Task Tile: Review Microbiology PhD Paper

Task Type: Review

Description: This paper has to do with biology and such.

Number of pages: 75

Number of words: 12000

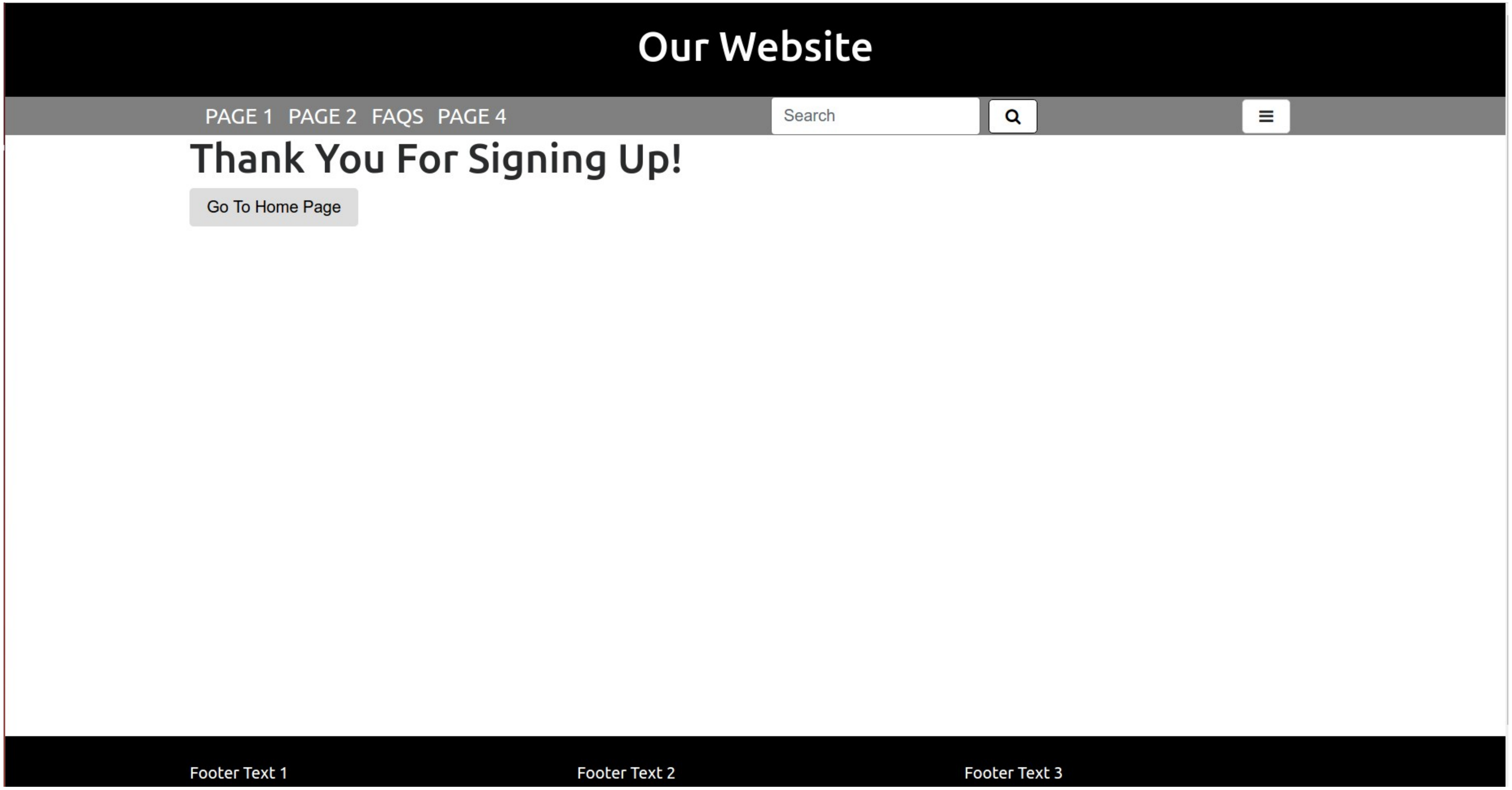
Claim Deadline: 12/3/2016

Complete Deadline: 12/3/2016

Paper Type: Phd Thesis

Status: Unclaimed

Thank You Page: (Note: This page will be styled further and may contain email verification functionality if we have time to spare.**)**



Thank You For Signing Up!

Appendix 2: Tables

Note: For each attribute the name, datatype and length are shown in the images and thus they are not mentioned in the description.

Table Name: User
Primary ID: StudentID
Description:

- **UserID** Unique identifier for users where valid values are 0 to 99999999999.
- **Subject_SubjectID** Foreign key that references the unique identifier for subjects. No action is taken when we update / delete subjects (subjects will be a fixed list so this should not happen).
- **ForeName** Students first name.
- **LastName** Students last name.
- **EmailAddress** The email address entered by the user. This attribute must be unique.
- **StudentID** This is the users student Id as per his/her college
- **Password** The users password.
- **Reputation** Reputation of the user.
- **IsMod** Whether the user is a moderator or not.

Table Name: TaskTag
Primary ID: TaskTagID
Description:

- **TaskTagID** Identifies the individual tags that belong to a particular task. It is auto incremented. This table can contain at most 4 rows.
- **Task_TaskID** This is a foreign key that references the the Task table. When a Task is deleted its TaskTags are also deleted (ON DELETE CASCADE).
- **Tag_TagID** This is a foreign key that references the Tag table. Default behaviour on delete and update (NO ACTION).

Table Name: Task
Primary ID: TaskID
Description:

- **TaskID** Identifies individual tasks (which can be created by users).
- **User_StudentID** Foreign key to indicate the user that created the task. Constraint: when user is deleted the task is deleted.
- **Subject_SubjectID** Foreign key to indicate the task's subject. Constraint: default action on delete/update.
- **Status_StatusID** Foreign key to indicate the task's status. Constraint: default action on delete/update.
- **Title** The title of the task.
- **Description** Description of the task.
- **NumPages** The number of pages in the document being submitted
- **NumWords** The number of pages in the document being submitted.
- **ClaimantID** The ID of the claimant if the task has been claimed.

Table Name: Tag
Primary ID: TagID
Description:

- **TagID** Unique identifier for tags.
- **Value** The tag's actual value (eg. science, engineering).

Table Name: Subject
Primary ID: SubjectID
Description:

- **SubjectID** The primary key for subjects. Each task and each user has a subject.
- **SubjectName** The name of the subject (eg physics).

Table Name: Status
Primary ID: StatusID
Description:

- **StatusID** The primary key for statuses. Each task has a status at given period of time.
- **StatusVal** The string that represents the actual status (eg unclaimed, claimed, completed).

Table Name: Format
Primary ID: FormatID
Description:

- **FormatID** The primary key for Format. Each document has a Format.
- **FormatVal** The actual format (eg .pdf, .odt etc)

Table Name: Flag
Primary ID: FlagID
Description:

- **FlagID** The unique identifier for a Flag. Each task can get flagged by a user.
- **Task_TaskID** The ID of the task that was flagged.
- **Complaint** The reason given for the flag.

Table Name: Document
Primary ID: DocumentID
Description:

- **DocumentID** The unique identifier for a specified document. Each task has a document attached to it.
- **DocumentURL** The url of the document stored on the server.
- **Type** The type of document that the task involves (eg phd thesis, dissertation, project report.
- **Task_TaskID** The task that the document belongs to.
- **Format_FormatID** The document format (.pdf, .odt etc)

Table Name: Deadline
Primary ID: Task_TaskID
Description:

- **Task_TaskID** The task that the deadlines belong to.
- **Claim** The deadline date for claiming the task
- **Completion** The deadline date for completing the task

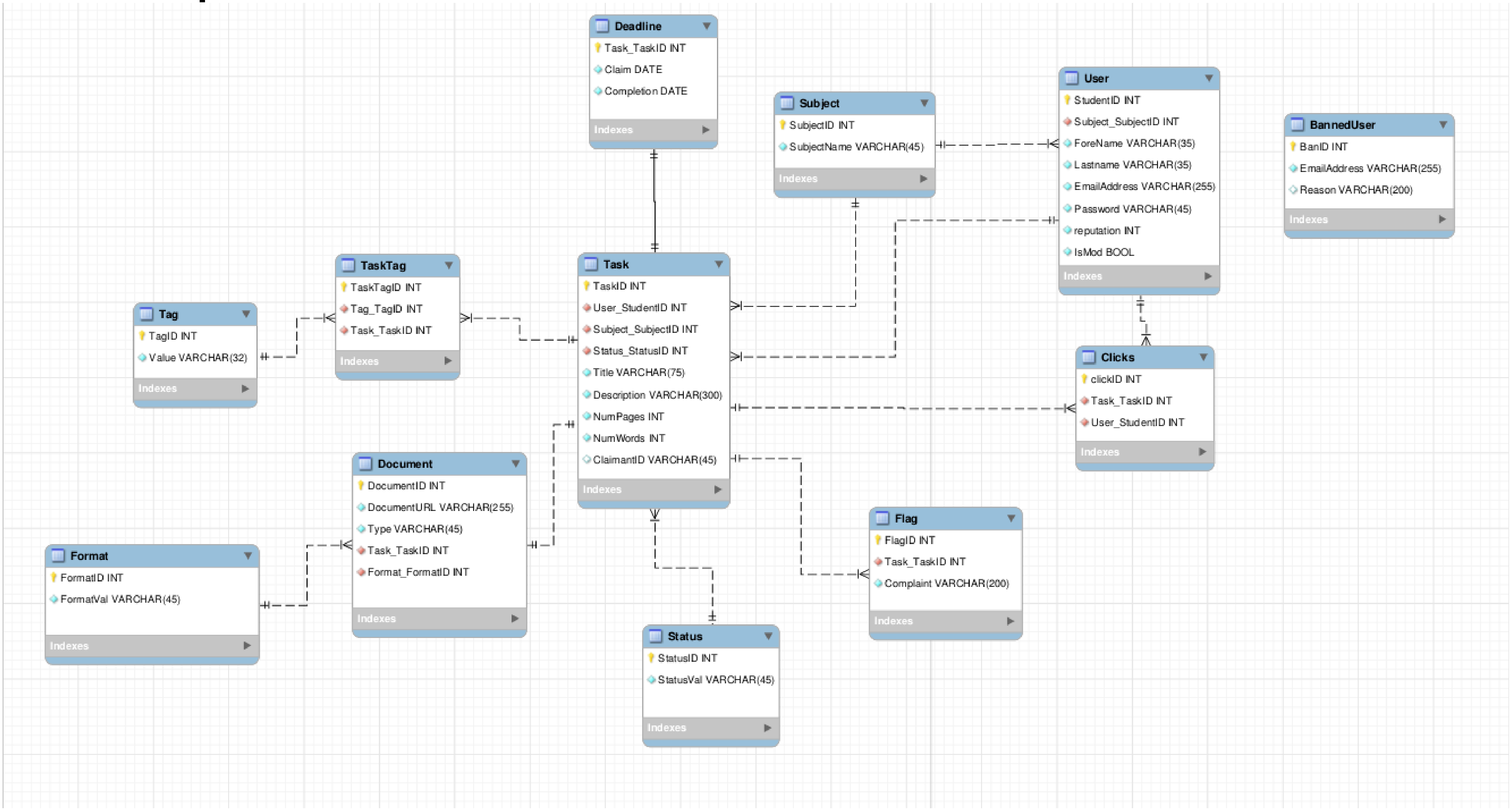
Table Name: Click
Primary ID: clickID
Description:

- **ClickID** The unique identifier for specific clicks. A click is when a user clicks on a tasks to view the tasks details. This will be used to optimize the users suggested tasks and to allow a 'users that viewed this task also viewed' functionality.
- **Task_TaskID** The task that the user clicked on.
- **User_StudentID** The user that clicked on the task.

Table Name: BannedUser
Primary ID: BanID
Description:

- **BanID** Unique identifier for each instance of a banned user.
- **EmailAddress** The email address of the user who was banned.
  Although the banned user will be deleted from the database it is important that a user can't make a new account with the same email address so before a new user creates an account we ensure that the email address provided is not one of those belonging to a banned user.
- **Reason** The reason for banning this user given by a moderator.

Relationships Between Tables:



Description of Table Relationships:

- Each user can have many tasks.
- Each user can have many clicks (when a user clicks on a specific task).
- Each subject can belong to many users or tasks.
- Each task has a single status (at any given time).
- A task can be flagged multiple times.
- A task has one set of deadlines.
- A task has a single document attached to it.
- Each format can belong to multiple documents.
- Tasks and Tags have a many to many relationship. This is not supported by mysql thus we have the intermediary table TaskTags. TaskTags stores the tasks that have certain tags.

Appendix 3: Processes

Process Number	P1
Process Title	Sign Up
Brief Description	Allow users to register a new account
Inputs	Email address, name, password, subject, student ID
Detailed Description	Users enter their information, it is validated at the client and server side, it is inserted into the database.
Output	redirects to thank you page / error message output.

Process Number	P2
Process Title	Sign In
Brief Description	Allow user to enter their email and password and sign into the site
Inputs	Email address, password.
Detailed Description	User information is entered, it is validated at the server side, a select query is sent to the database to check if a user with the specified email and password exists. If something is returned then the user is logged in. If not then an error message is presented.
Output	Error message or redirected to home page.

Process Number	P3
Process Title	Browse Tasks
Brief Description	On the home page the user can browse tasks that are unclaimed
Inputs	None (or filters for subject, type of task, tags)
Detailed Description	Tasks will be loaded from the database using a select query and will be organised based on the user using a number of factors such as the user's browse history, subject of study, and any filters they have applied.
Output	The tasks that populate the page.

Process Number	P4
Process Title	Search tasks
Brief Description	User searches for a task based on title
Inputs	String thats similar to a title of a task they wish to recieve
Detailed Description	User enters a string that we use to query the database using the SELECT clause a LIKE statement.
Output	A list of similar tasks.

Process Number	P5
Process Title	Flag Task
Brief Description	Users flag a task they feel is inappropriate by clicking the flag button and entering a complaint.
Inputs	A message that specifes why they flagged the task.
Detailed Description	User clicks on the flag button on the task details page. Modal pops up that allows them to specify a reason and then allows them to flag the task. This information is inserted into a the 'Flag' table using an sql insert statement.
Output	Error message or redirected to home page.

Process Number	P6
Process Title	Unpublish task (Moderator)
Brief Description	Moderator browses Flagged task page and un publishes task they feel is inappropriate
Inputs	none
Detailed Description	Flagged tasks page populated with the tasks that have associated flags is loaded using a select (and join) query. Moderator browses tasks. Moderator selects task to un-publish. Task deleted from database.
Output	Page with unpublished task removed.

Process Number	P7
Process Title	Ban user (Moderator)
Brief Description	Moderator clicks ban user on user profile and gives a reason.
Inputs	Reason for banning
Detailed Description	When user is banned a new row is added to the 'BannedUser' table and the users info is deleted from the 'user' table (along with any corresponding tasks).
Output	none

Process Number	P8
Process Title	Claim Task
Brief Description	Click claim task button on task details page.
Inputs	none
Detailed Description	When the button is clicked the status of the task is changed and the claimantID is added to the row for that particular task using an sql update statement.
Output	Modified task details page

Process Number	P9
Process Title	New Task
Brief Description	Click new task button on main page.
Inputs	All task details eg. title, subject, date etc.
Detailed Description	When the new task button is clicked a form modal is brought up requesting all task details. Once this form is submitted a new task entry is created in the task table in the database with all the form information.
Output	Modified main tasks list