Microaggression Expressions Submission Site

Final Project Report

April 28, 2016

CS4624, Virginia Tech, Blacksburg, VA 24061

Members:
Robert Wenger
Brandon Falcone
Arunima Singh
Cyndy Ejanda

Client:
Josh Iorio
Principal faculty in the Myers-Lawson School of Construction
iorio@vt.edu

Abstract

According to Dr. Stephanie Adams, Department Head of Engineering Education at Virginia Tech, "The value of having diversity is that it brings multiple perspectives to solving problems. If everyone on the team looks the same, and has had the same experiences, then they can't think about it from another person's perspective. But when you bring people together with diverse backgrounds, races, genders, and sexual orientations, the group can approach a problem from a range of perspectives that makes for a much better solution."

A more diverse community at Virginia Tech brings multiple perspectives to solving problems. Creating a cultural climate that is supportive of everyone, can be difficult because of microaggressions. Microaggressions are the seemingly harmless snubs or insults communicated verbally or nonverbally that target an individual based on their group membership. Eliminating the occurrence of microaggressions maintains and promotes diversity at Virginia Tech. The College of Engineering is particularly homogeneous and is thus an excellent starting point for addressing microaggressions. The College consists of 85% male professors, and 15% female professors, of which 0.3% are American Indian, 21% Asian, 2% Black, 65% White, 6% Hispanic, 0.3% Multiracial, and 7% non-resident alien.

To bring about a widespread understanding of how microaggressions affect people, a website (http://microaggressions.cs.vt.edu/) was created that allows faculty members in the College of Engineering to anonymously share their personal experience with microaggressions. By submitting anonymous posts, faculty can freely express their concerns, which can then be viewed publicly. This helps bring to the surface the kinds of hurtful comments that repeatedly get dismissed in the work environment. It is important to be able to study these posts – uncovering hidden patterns, correlations, and other insights – in order to better understand the problem. That is why we have incorporated filtering and visualization tools like a word cloud, N-grams, and graphs to allow the user to better understand patterns underlying the problem. We also implemented a simple anonymity checker that analyzes the post upon submission, looking for keywords that could identify a person. This, combined with a panel that allows the administrator and moderators to manually approve and flag posts prior to publication, ensures anonymity and privacy.

Making faculty feel more included at Virginia Tech is the goal of this website. Our goal is for ALL faculty to feel accepted and included at Virginia Tech. We will accomplish this through our website by raising awareness of the damaging impact of microaggressions.

Contents

Table of Figures

Table of Tables

1. Executive Summary

The main objective of this project is to create a website so that those concerned with diversity related microaggressions can report, search, share, and study microaggressions. Josh Iorio, the principal faculty in the Myers-Lawson School of Construction, will serve as the client for this project. He will communicate with the College of Engineering to help them understand the effects of microaggressions on its targets and on the organizational climate of the department.

Microaggressions are the everyday verbal, nonverbal, and environmental slights, snubs, or insults, whether intentional or unintentional, that communicate hostile, derogatory, or negative messages to target persons based solely upon their marginalized group membership" (UCLA, 10). The scope of our project currently consists of the faculty members in Virginia Tech's College of Engineering.

The website is non-interactive and have anonymity as its key feature. Users should be able to post anonymously to the website. Any information that could reveal their identity must be flagged. We are storing all the posts in a database table. We will also dynamically collect tweets related to microaggression. The website will enable the study and analysis of microaggression based on gender, ethnicity, and department.

The core functions include submitting, storing, viewing, and searching for posts. Advanced functions include collecting tweets, creating bigrams/trigrams, creating tag clouds, and sharing the website's content on Facebook. We are using the Foundation front-end framework which is a collection of HTML, CSS, and JavaScript to build upon the website. The back-end of the project uses PHP and MySQL to store, manage, and search submitted posts and PHP and MySQL to manage tweets. We are using an instant-on approach for implementation. The site will be deployed on a server maintained by the computer science department. This machine is a virtual machine with 2 cores, 4GB RAM, and 500GB storage running 64-bit CentOS 6.7. Apache, MySQL, PHP, FTP software, and PHPMyAdmin are installed.

We are using the Twitter API to collect tweets. We are searching for specific hashtags (e.g. @microaggressive, @racist @sexist) that have content that's often related to microaggressions. Having anonymity as the key feature, we will not be asking any contact information from our users on the submission page. The post will then be reviewed by the moderator staff for acceptability. Specifically chosen faculty members will serve as the moderators, as decided by Josh Iorio. The exact number is not yet identified but we are assuming the number of moderators to be from 7 to 12. We do not have any non-disclosure agreement.

We have also considered the implementation of the non-functional requirements. We have made our website compatible with different screen resolutions on different types of devices, such as laptop, smartphones, and tablets. We have devised a way to deal with malicious posts by using Google’s reCAPTCHA. We aim to provide a better quality of service by publishing posts to the website within 24 hours. Since we are assuming all the posts to be real-life events, we will not be checking the truthfulness of the post.

2. Application Overview

2.1. Objective

What business objectives of the company will this project help achieve?

Josh Iorio, the principal faculty in the Myers-Lawson School of Construction, will serve as the client for this project. He will communicate with the College of Engineering to help them understand the effects of microaggressions on its targets and on the organizational climate of the department. By providing the College of Engineering with this project, through Josh Iorio, the professors and staff will be able to address their microaggression concerns in order to improve the current climate. Josh will analyze the posts gathered from the website to make future changes within the College of Engineering.

Who will benefit from this project?

VT College of Engineering - Microaggressions are a current problem at Virginia Tech giving professors a feeling of exclusion from others in the Department. This has a negative effect on Virginia Tech because these professors will leave to pursue their teaching careers at other universities. The College of Engineering will benefit when professors and staff are part of a more inclusive environment where they can come together to express how microaggressions make them feel excluded.

Targets of Microaggression - The targets of microaggression will benefit from less hurtful comments. By providing a platform for those who have been targeted by microaggressions to express their stories and concerns, Josh Iorio can use these posts to make changes within the Department. Professors and staff browsing the website can be more informed about the microaggressions affecting others, and can better avoid saying such statements.

2.2. User Roles and Responsibilities

Below is the list all of the users of our system:

- Virginia Tech's College of Engineering faculty members
- Faculty members from other departments at Virginia Tech
- College of Engineering at Virginia Tech
- Other people who are affected by microaggression, such as Virginia Tech students, employees, and the general public.

What are the tasks of each of the users?

1. Virginia Tech's College of Engineering faculty members and faculty members from other departments at Virginia Tech:
    a. Submit posts
    b. View approved posts
    c. View posts analytics
2. College of Engineering at Virginia Tech
    a. View submitted posts
    b. Moderate posts
    c. Manage moderators
    d. Manage website content
3. Other people who are affected by microaggressions
    a. View approved posts
    b. View post analytics

2.3. Terminology

Apache - A freely available Web server that is distributed under an "open source" license.

Bigram - A sequence of two adjacent elements in a set. In this case, two consecutive words in the submission.

HTML (**H**yper**T**ext **M**arkup **L**anguage) - A standardized system for tagging text files to achieve font, color, graphic, and hyperlink effects on World Wide Web pages.

MySQL - A database management system commonly used in web development.

PHP (**P**HP: **H**ypertext **P**rocessor) - a programming language designed to be used in web development.

PHPMyAdmin - A free and open source tool written in PHP intended to handle the administration of MySQL with the use of a web browser.

reCAPTCHA - a type of challenge-response test to establish that a computer user is human.

Trigrams - A sequence of three adjacent elements in a set. In this case, three consecutive words in the submission.

3. Requirements

4.1. Functional Requirements

4.1.1. Statement of Functionality

The main objective of this project is to create a website so that those concerned with diversity related microaggressions can report, search, share, and study. We aim to do the following:

- Define microaggression, its different types, and its impact. This information will be presented on the About page.
- Create a website where those affected by microaggressions can post their stories through an anonymous submission form. These submitted stories will be described as posts throughout this report. The posts would be non-interactive meaning that they cannot be commented on or liked. There will be a ‘share your story’ page dedicated solely for submitted posts to the website.
- Anonymity will be the key feature here. The website will flag any post if it contains any specific information like an individual’s name, academic building information, and so on. Posts submitted with flagged information will not be posted to the website. However, we will ask the person about their gender, department name (optional), and ethnicity when making a post. We will include an anonymity checker to flag all the names and abbreviations of all the academic buildings. To ensure anonymity, a moderator can flag any post that reveals the identity of an individual.
- Create a database that will store all the posts and the admin/moderator information.
- Allow people to share the website’s content on other social networking sites, such as Facebook.
- Display tweets or hashtags related to this microaggression. This content would not be anonymous.
- Devise a way to manage old posts.
- Generate statistics based on this data. Create tag cloud and pie charts.
- We will allow filtering of posts and search on our website.
- We will include bigrams or trigrams for analysis and search purposes. For example: all the posts that includes the phrase “I feel like” will be displayed.

- In order to handle malicious users, we will use Google's reCAPTCHA service to protect the website from spam and abuse.
- We would also include encryption of admin's and moderator's password to increase website security.

4.1.2. Scope

The scope of our project currently consists of the faculty members in the Virginia Tech's College of Engineering. The website will provide a platform to share, study, and search about microaggression.

4.1.3. Performance

Users should be able to post anonymously to the website. Any information that could reveal their identity must be flagged. We will store all the posts in our database table. We will also dynamically display tweets related to microaggression. The website will enable the study and analysis of microaggression based on gender, ethnicity, and department.

4.1.4. Usability

Users will not be able to interact with the posts such as being able to comment on or like a post. Anonymity is a key feature so we will not have log-in accounts for users. Apart from posting their stories on the website, users can study, analyze, and share the website's contents on other social networking site, such as Facebook.

4.2. Non-functional Requirements

The non-functional requirements are as follows:

- We need to ensure that our website is compatible with different browsers and platforms. Our website is compatible with Chrome, Firefox, and Internet Explorer. The only compatibility issue we are having is that Internet Explorer shows the header bar incorrectly, but it is still functional.
- We need to ensure that our website is compatible with different screen resolutions on different types of devices, such as laptop, smartphones, and tablets. Our website is able to be displayed properly on small devices. It is able to adjust to various screen sizes..
- Good quality of service, such as time period to publish a post to the website. This requirement will be met by the moderators chosen by our client. Having more moderators will help ensure posts are reviewed and approved quickly.

4. User's Manual

3.1 Administrators

The role of the administrator is to manage submitted posts, manage moderators, and manage website content. Managing submitted posts involves reading posts faculty members have submitted and approving or flagging each one. Managing moderators involves filling out a form to add a moderator and having knowledge of the moderator's userID to remove the moderator.

3.1.1 Managing Posts

Below are the steps that the admin would undertake for managing the posts in the admin/moderator dashboard:

1. The admin would go to the admin/moderator login webpage (http://microaggressions.cs.vt.edu/admin/login.php), and enter their username and password. The admin will select the role of 'admin' from the combobox.
2. On successful login, the admin will be directed to the admin/moderator dashboard. From this page, the admin will be presented with the number of new posts awaiting moderation. The admin can select the 'manage new posts' options on the left to manage all the new posts submitted to the website.
3. The admin will then review all the new posts. If the posts do not reveal the identity of another person, the admin will publish the post to the website. In the database, the status of the post will be updated from 'new' to 'published.' If the admin chooses to not publish the post (flag the post) the database will update the status of the post from 'new' to 'flagged.'
4. The admin can repeat step 3 for all newly submitted posts.
5. The admin can then exit the dashboard by selecting the 'logout' button. Then admin will then be redirected to the login menu.

3.1.2 Managing Moderators

Below are the steps that the admin would undertake for managing the moderators in the admin/moderator dashboard:

1. The admin would go to the admin/moderator login webpage (http://microaggressions.cs.vt.edu/admin/login.php), and enter their username and password. The admin will select the role of "admin" from the combobox.
2. On successful login, the admin will be directed to the admin/moderator dashboard. The admin can select the 'manage moderators' options on

the left. This web page will only be accessible to people with a role assigned as an ‘admin.’

3. The admin will then select one of the three options: add, delete, or update a moderator’s profile.
   3a. In the case of add/update moderator profile, the admin will enter all the required fields. The web page will include validations, and the data will be saved to the database.
   3b. In the case of deleting a moderator profile, the admin will select the moderator profile to be deleted. The web page will ask the admin for confirmation. If the admin selects ‘yes,’ the moderator’s profile will be removed from the database.
4. After the add/delete/update moderator’s profile, the web page will display a confirmation message.
5. The admin can repeat step 3 for all the other moderators.
6. The admin can then exit the dashboard by selecting the ‘logout’ button on top of the screen.

3.1.2 Managing Website Content

Below are the steps that the admin would undertake for managing the website content in the admin/moderator dashboard:

1. The admin would go to the admin/moderator login webpage (http://microaggressions.cs.vt.edu/admin/login.php), and enter their username and password. The admin will select the role of “admin” from the combobox.
2. On successful login, the admin will be directed to the admin/moderator dashboard. The admin can select the ‘manage website contents’ options on the left. This web page will only be accessible to people with a role assigned as an ‘admin.’
3. The admin will then be able to manage the twitter hashtags or the textual content on a webpage.
   3a. In the case of managing the twitter hashtags, the admin will enter the twitter hashtags in the text box. Each hashtag must be separated by the word “OR”. Please note that you might have to reload the page after submitting in order to see the changes.
   3b. In the case of managing the textual content on a webpage, the admin will click the name of the page they will like to edit from the ‘Pages and Sections’ section. The admin will be redirected to a text box containing the HTML code containing the text that appears in the selected pages. The admin will need to locate the code to remove/change in order to edit the content.
4. After making the changes to the website content, the admin will click the ‘Update’ button located under the text box.

5. The admin can then exit the dashboard by selecting the ‘logout’ button on top of the screen.

3.2 Moderators

The role of a moderator is to manage submitted posts. Managing submitted posts involves reading posts faculty members have submitted and approving or flagging each one. This is a task that is shared by both admins and moderators.

3.2.1 Managing Posts

See section 3.1.1

3.3 Website Visitors

The role of the website visitors is to submit posts and analyze/study microaggressions Submitting a post involves filling out a post form. Analyzing/Studying microaggressions involves reading and understanding bigrams/trigrams, pie charts, and tag clouds.

3.3.1 Submitting a Post

The link to submit a post page can be accessed by clicking on the “Share Your Story” links that can be found in the header bar at the top of the website.

Upon clicking the link, you will be taken to a page that will give you the rules for posting as well as the form to submit a story.

Using the Submission Form:

1. Select your gender and ethnicity which are both required information
2. Select your department (not required)
3. Enter your story in the “Tell your story” text box(required)
4. Once you are done, click the submit button
   a. If your post contains keywords that could identify a person’s identity (either yourself or someone else’s), the website will popup a notification informing about this. You will be given the option to edit your post or submit the current one as it is. Note that if you chose to submit the post that contains keywords that could potentially identify someone’s identity, the website will flag it. Posts submitted with flagged information will not be posted until after the moderator reviews them.
5. Before clicking submit, users will need to click the checkbox indicated that they you have read the rules for posting located above the post form. They will also need to click on the reCAPTCHA checkbox establishing they are human.
6. Once you have successfully submitted the post, it will be reviewed by the moderator staff for acceptability within a few days.

3.3.2 Analyzing/Studying Microaggressions

The “About” and “Resources” page provides a general overview about microaggressions. The “Analytics” page contains tools for filtering submitted posts in order to study a specific gender, ethnicity, or department. It also contains tag clouds, bigrams and trigrams, and pie charts as visual aids for analyzing submitted post data. Tag clouds are also displayed on the “Home” page and “Posts” page. The “Posts” page also displays recently submitted tweets with hashtags such as #microaggression.

3.3.2.1 Filtering Posts

The “Posts” page also gives you the ability to filter the posts to allow you to focus on certain topics.

Gender:
To view posts about a specific gender, click on the Gender drop-down box and select the gender you want. Click the ‘Filter’ button to begin the filtering process to display only the posts related to the gender you chose. If you would like to undo the filtering, simply select the option “Gender” from the dropdown box and click ‘Filter’ again.

Ethnicity:
To view posts about a specific ethnicity, click on the Ethnicity drop-down box and select the ethnicity you want. Just like for the gender option, click the ‘Filter’ button after making a selection. If you would like to undo the filtering, simply select the option “Ethnicity” from the dropdown box and click ‘Filter’ again.

Department:
To view posts about a specific department, click on the Department dropdown box and select the department you want. Click the ‘Filter’ button to begin the filtering process to display only the posts related to the department you chose. If you would like to undo the filtering, simple select the option “Department” from the dropdown box and click ‘Filter’ again. Please note that we do not require the users to indicate their department when submitting a post so some posts might not have a department name tied to them.

3.3.2.2 Filtering Data

The “Analytics” has a filtering tool to display analytics about a specified gender, ethnicity, or department. The filtering procedure is the same as the procedure specified in section 3.3.2.1. However, the filtering process updates the tag cloud, bigrams and trigrams, and pie charts that reflect the filtering options.

3.3.2.3 Analyzing Tag Clouds

Tag clouds appear on numerous pages such as the “Home” page, “Posts” page, and “Analytics” page. The tag cloud on the “Analytics” page displays frequently occurring words based on the specified filter options. By default the tag cloud displays frequently occurring words across all submitted posts. The tag clouds displayed on the “Home” page and “Posts” page also display the frequently occurring words across all submitted posts.

3.3.2.4 Analyzing Bigrams and Trigrams

The “Analytics” page contains bigrams and trigrams which display the most frequently occurring sequences of words that appear in submitted posts. The filtering tools can display bigrams and trigrams based on the specified filtering options.

5. Developer’s Manual

5.1. Server Information

The site is currently deployed on a server maintained by the computer science department. This machine is a virtual machine with 2 cores, 4GB RAM, and 500GB storage running 64-bit CentOS 6.7. Apache, MySQL, PHP, FTP, and PHPMyAdmin are installed.

5.2. Making Changes to the Website

Developers who want to make changes to the website will need to understand the structure of the website (described in 4.5.1.2). Developers will also need knowledge and understanding of the core technologies used within the website such as PHP, MySQL, Foundation Framework, HTML, CSS, jQuery, and JavaScript. You will find sections 4.3.1.1, 4.3.1.2, 4.3.1.3, 4.3.1.5, 4.3.2, and 4.4.1.1 to be particularly helpful in understanding how each feature has been implemented.

5.3. Design

5.3.1. Front-end Functionality

5.3.1.1. Common Features

An important feature of the website is the ability to make anonymous posts. To accomplish this, all posts will be analyzed by the anonymity checker to detect certain keywords. Upon clicking the submit button, if

the post contains a building name (abbreviation or full name), the anonymity checker will alert the users that the post they are about to submit contains keywords that could identify them or someone else. We then give the user an option to edit the post or proceed with the submission of the post.

A moderator will be used to review all posts that are made and remove posts that could reveal the identity of others. The moderator will log in using a password allowing him to remove such posts. The moderator will also be able to review the “flagged” posts that have been submitted and submit the ones that are anonymous. The faculty involved in this project will be the moderators. The exact number is not yet identified but we are assuming the number of moderators to be from 7 to 12.

The “Home” page will display a tag cloud containing frequently used words contained in recently submitted posts. These recently submitted posts will be displayed in a photo carousel that automatically sequences through. The most recent posts will be determined by comparing the post’s submit date and time.

The website will also contain a Twitter widget that displays tweets related to microaggression. We also have a data analytics page that enables the user to study the data visually through pie graphs or through ngrams.

Having anonymity as the key feature, we will not be asking any contact information from our user. After the user submits a post, they would be directed to a ‘Thank you for your submission’ page. The post will then be reviewed by the moderator staff for acceptability.

We will handle malicious users by implementing a script that limits the number of posts per IP to be fifteen posts in a fifteen minute time period. We will also use Google’s reCAPTCHA service to protect the website from spam and abuse. We will include encryption of admin’s and moderator’s password to make the website more secure.

5.3.1.2. Sitemap and Website Structure

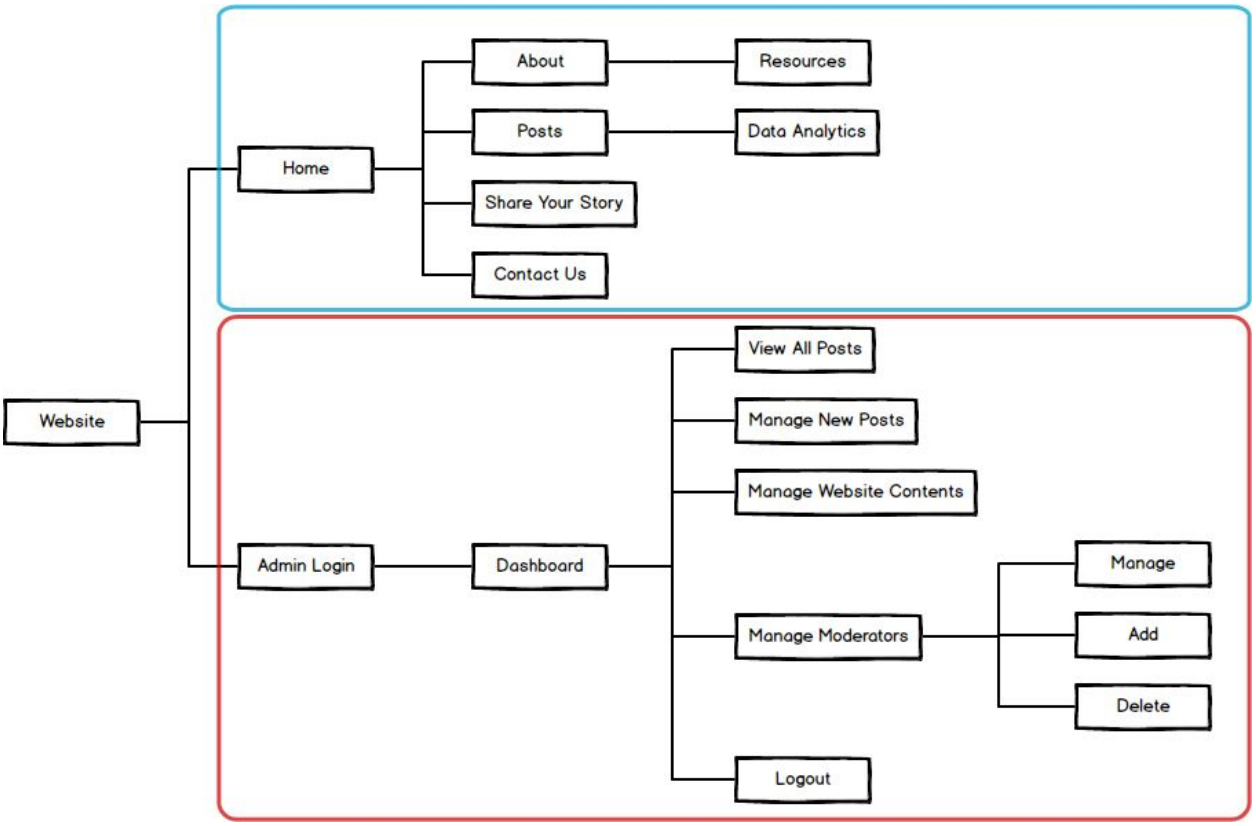


Figure 1: Final Sitemap

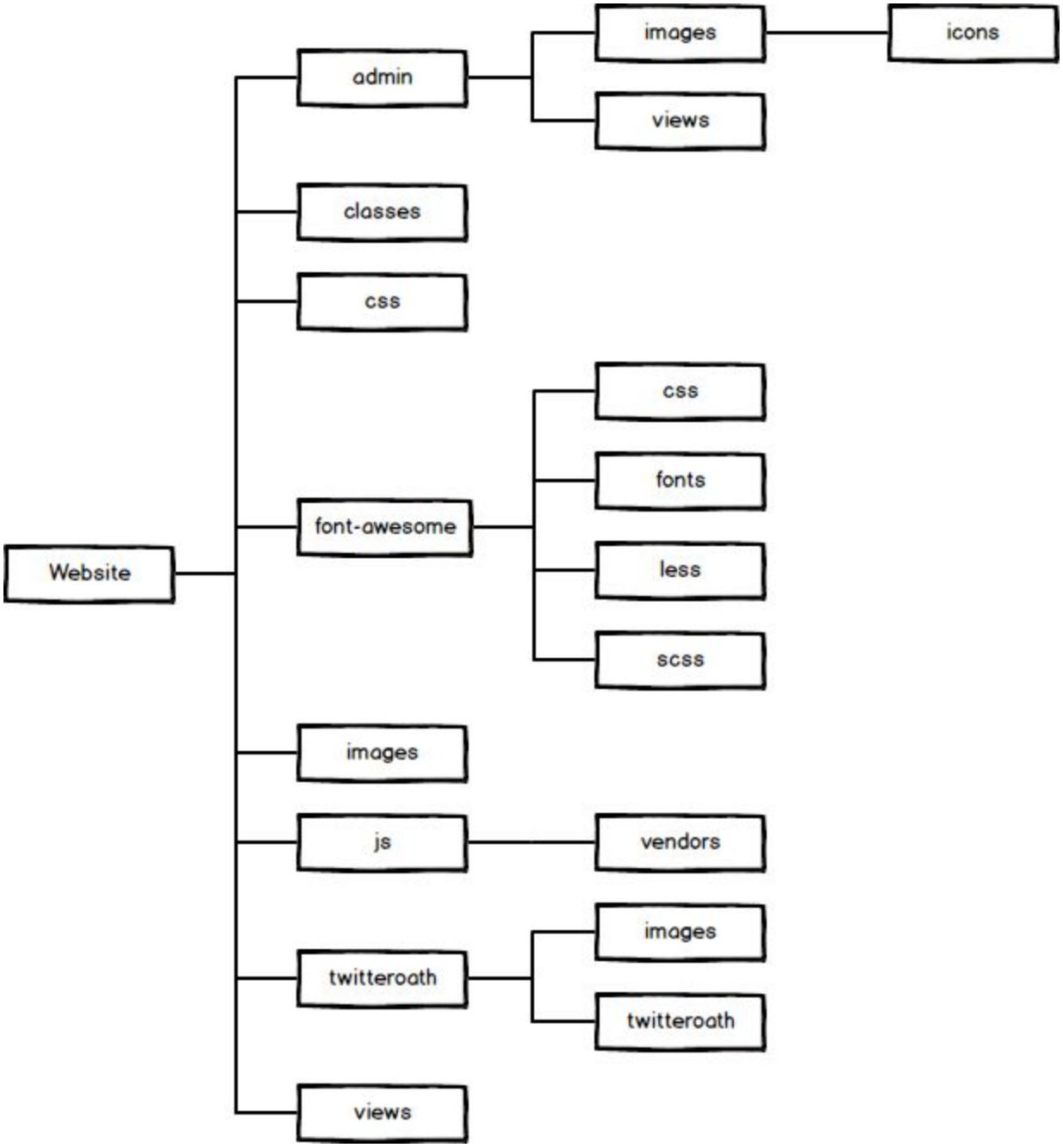


Figure 2: Folder Structure

5.3.1.3. Description of Website Structure

(Refer to Figure 11: Final Sitemap) The website is divided into two categories. The main website (boxed in blue) is accessible by the public, and the administration panel (boxed in red) is accessible only by the admin and moderators.

Main Website:

The “Home” page displays a few of the most recent posts, as well as a smaller version of the tag cloud..

The ‘Posts’ web page allows access to all of the posts that have been submitted. The posts will be displayed so that the most recent post is at the top of the webpage followed by older posts. In order to avoid having all posts displayed at the same time, we will include functionality that allows users to flip through pages, where a page will include 10-20 posts. This page also displays the Twitter widget and tag cloud.

The ‘Submit a Post’ web page allows users to make anonymous posts. Users will fill out a form that involves selecting choices from a combo box pertaining to their gender, ethnicity, department, and issue of their post, as well as typing their story into a text box.

The ‘About’ web page will be devoted to educational purposes to explain the purpose of the website and about micro-aggressions.

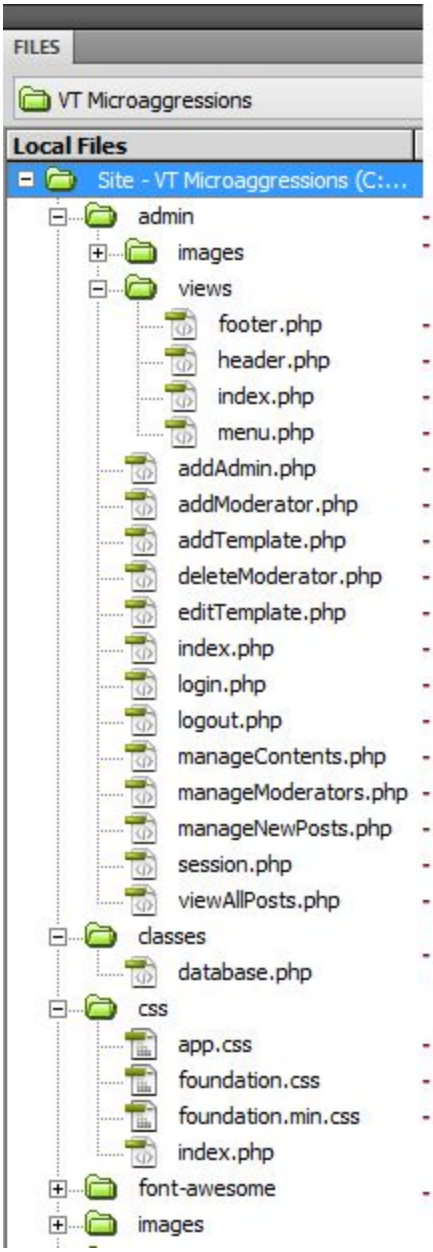
The ‘Resources’ page will include resources.

The ‘Thank you for your post’ page. Users will be redirected to this page after they submit a post.

The ‘Analytics’ page will include tools to help the users study the submitted posts. This page displays a table for bigrams and trigrams, a tag cloud, and pie charts.

The ‘Contact Us’ page contains a form that allows anyone to contact the admin or moderators.

Description of files and folders:



- contains files for admin panel
- contains icons that were used for the login page
- contents of footer container goes here
- contents of header container goes here
- just a place holder
- contains menu links
- allows the admin to add another admin
- allows the admin to add moderators
- allows the admin to add a website template
- allows the admin to delete a moderator
- allows the admin to edit parts of the website
- main dashboard page
- code for logging in
- code to end a session
- page for admin to modify page sections and twitter hashtags
- allows admin to manage moderators
- allows admin and moderators to manage posts
- code for current session; helpful in determining who is logged in
- allows admin and moderators to view posts
- contains code for managing the database and handling queries it also contains functions to make query requests easier
- contains custom styles; mostly not related to frontend structure
- contains default stylesheet
- minimized version of default stylesheet
- used for displaying textual "icons" throughout the website

Description of files and folders (cont):

Figure 3: Description of Files and Folders

5.3.1.4. User Interface Design

5.3.1.4.1. Wireframes

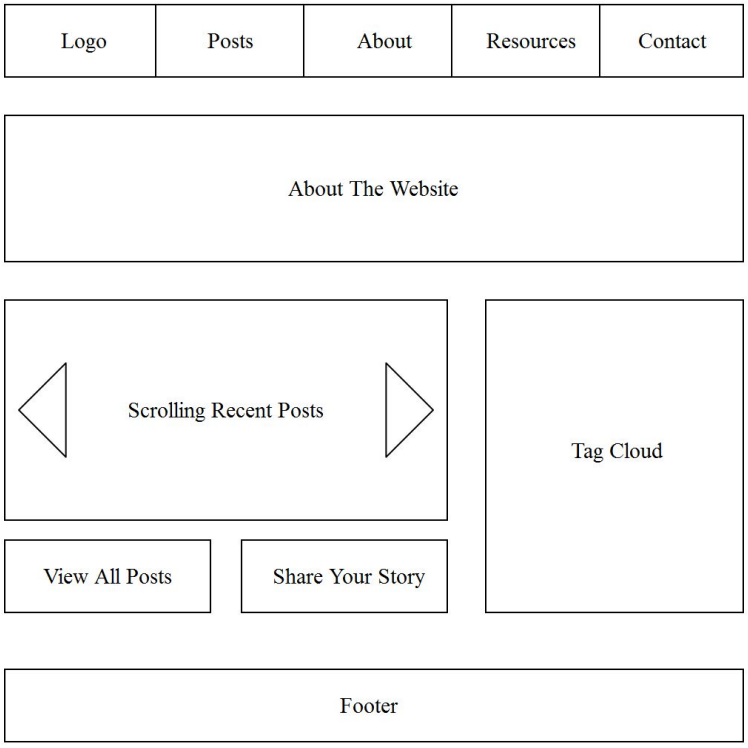


Figure 4: The 'Home' page

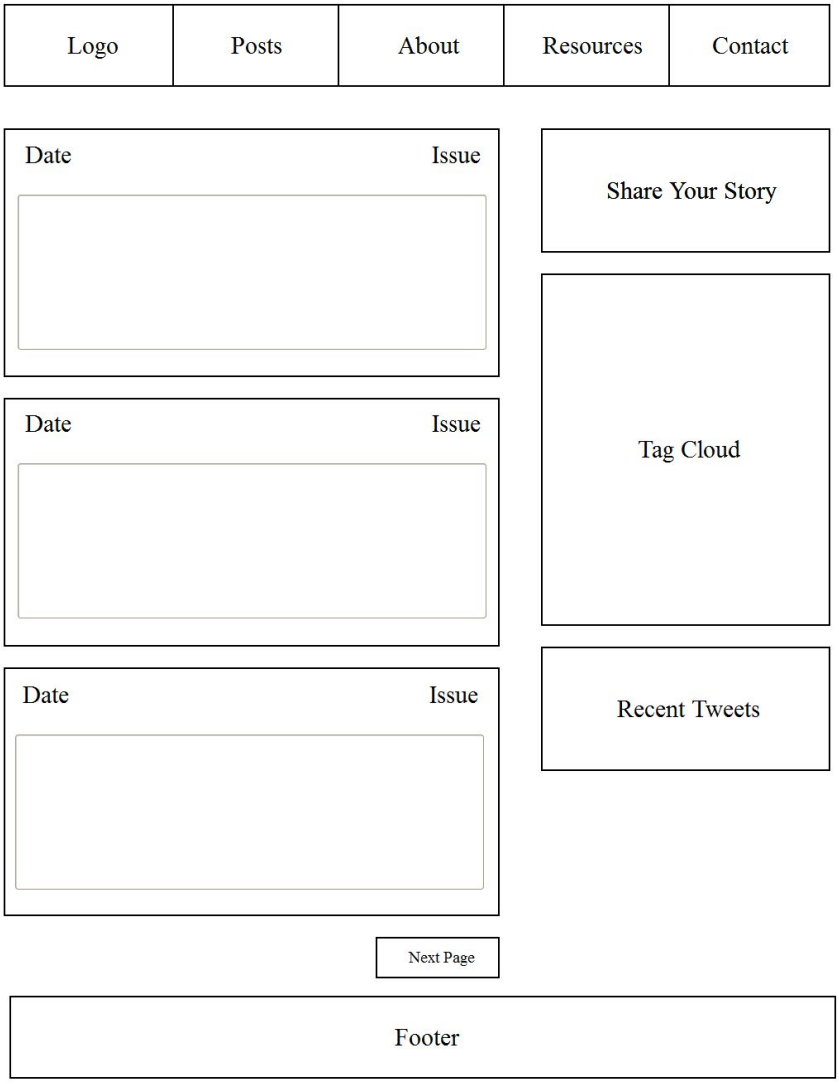


Figure 5: The 'Posts' page

Figure 6: The 'Make A Post' page

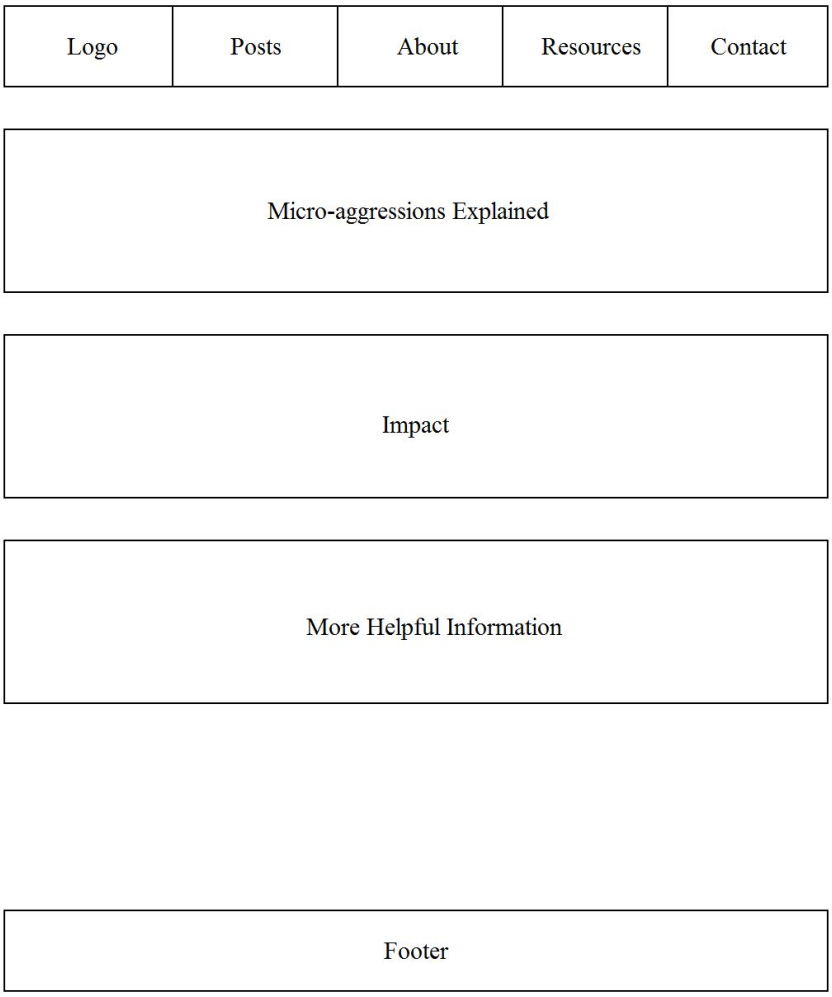


Figure 7: The 'About' page

5.3.1.4.2. Initial Mockups

Figure 8: Initial Mockups

5.3.1.4.3. Final Mockup

Microaggressions Expressions Home | About | Posts | Submit A Post | Contact Us search

About Us

Lorem ipsum dolor sit amet, ultricies dolor vestibulum felis lectus tempor cras, non leo varius viverra id. Vitae lobortis, in suspendisse dolor metus lectus maecenas, sem dolor cras sed, nulla et ipsum eget at vestibulum, morbi semper. Lorem ipsum dolor sit amet, ultricies dolor vestibulum felis lectus tempor cras, non leo varius viverra id. Vitae lobortis, in suspendisse dolor metus lectus maecenas, sem dolor cras sed, nulla et ipsum eget.

February 2, 2016 Gender

Lorem ipsum dolor sit amet, ultricies dolor vestibulum felis lectus tempor cras, non leo varius viverra id. Vitae lobortis, in suspendisse dolor metus lectus maecenas, sem dolor cras sed, nulla et ipsum eget at vestibulum, morbi semper. Lorem ipsum dolor sit amet, ultricies dolor vestibulum felis lectus tempor cras, non leo varius viverra id. Vitae lobortis, in suspendisse dolor metus lectus maecenas, sem dolor cras sed, nulla et ipsum eget at vestibulum, morbi semper.

View All Posts | Submit A Post

Microaggressions Expressions Home | About | Posts | Submit A Post | Contact Us search

Rules For Posting

Lorem ipsum dolor sit amet, ultricies dolor vestibulum felis lectus tempor cras, non leo varius viverra id. Vitae lobortis, in suspendisse dolor metus lectus maecenas, sem dolor cras sed, nulla et ipsum eget at vestibulum, morbi semper. Lorem ipsum dolor sit amet, ultricies dolor vestibulum felis lectus tempor cras, non leo varius viverra id. Vitae lobortis, in suspendisse dolor metus lectus maecenas, sem dolor cras sed, nulla et ipsum eget.

Submission Form

Gender: Please Select

Ethnicity: Please Select

Department: Please Select

Issue: Please Select

Message:

Share your story...

Submit

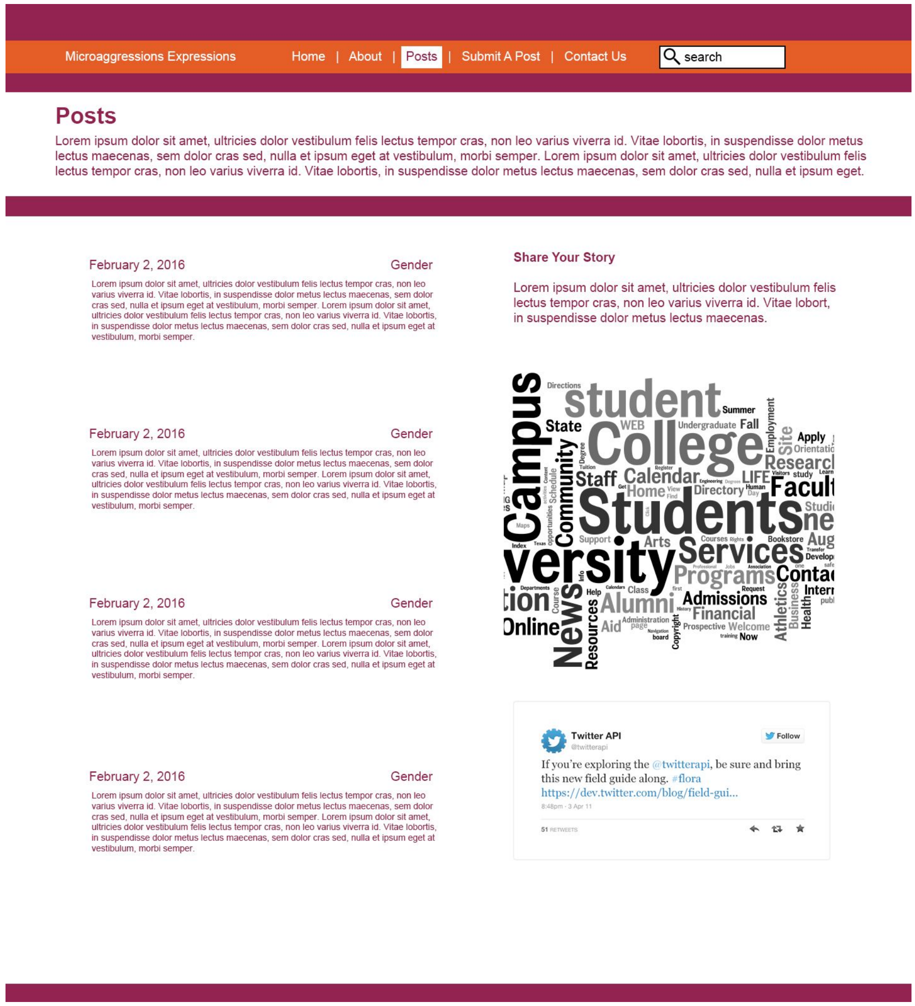
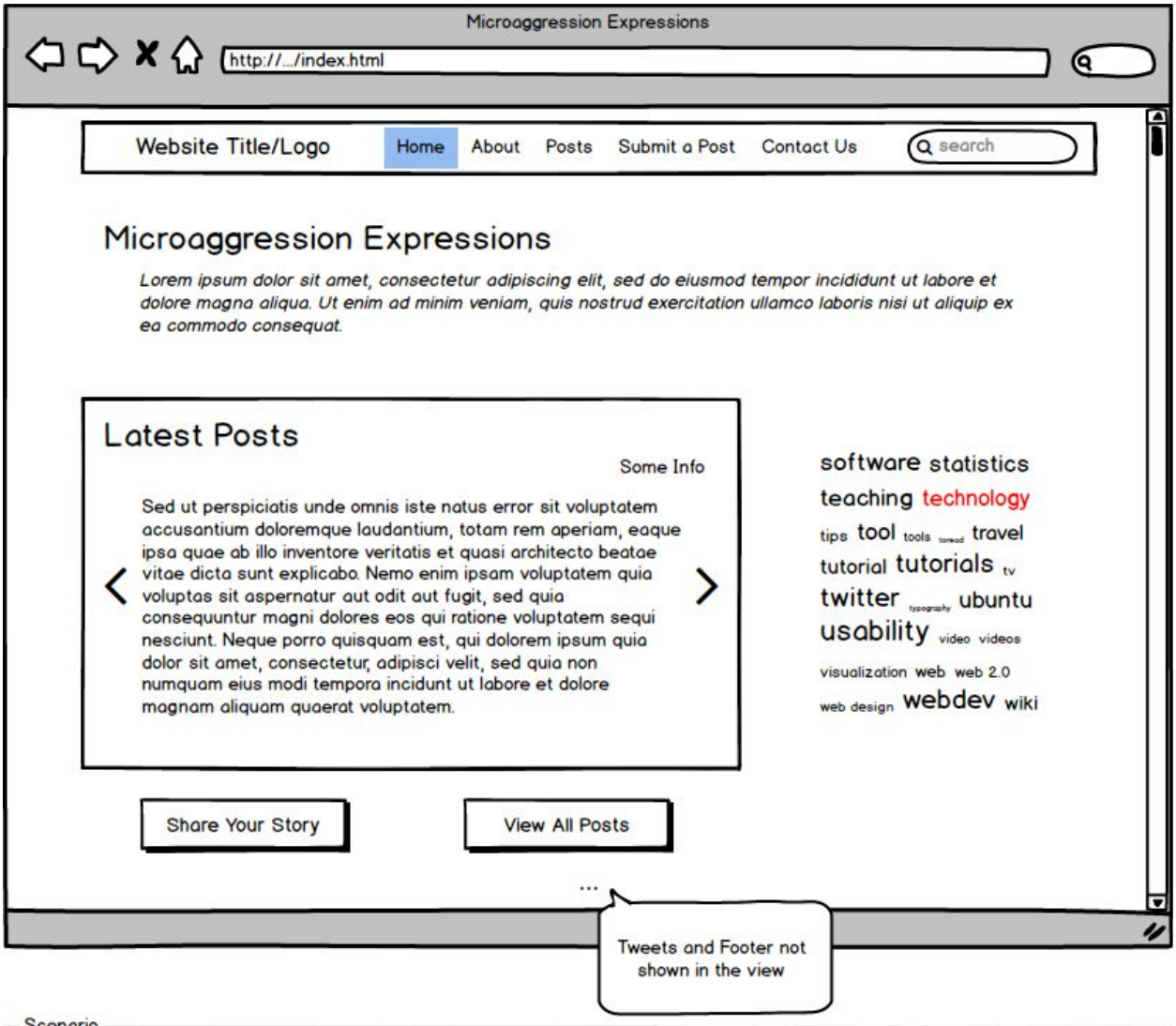


Figure 9: Final Mockup

5.3.1.4.4. Storyboards



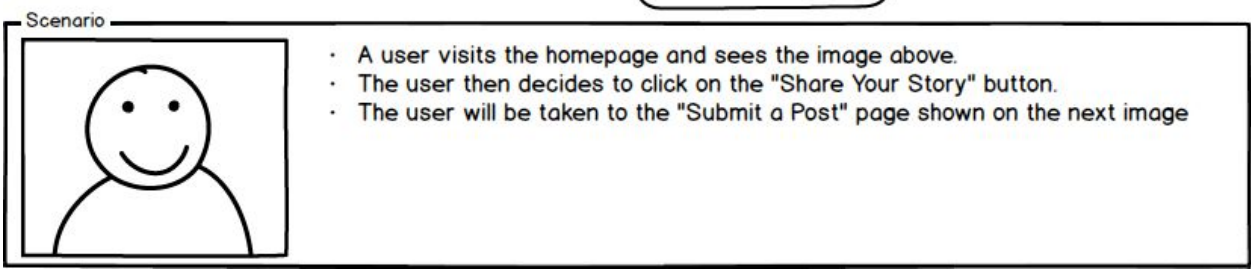


Figure 10: Home Page Storyboard

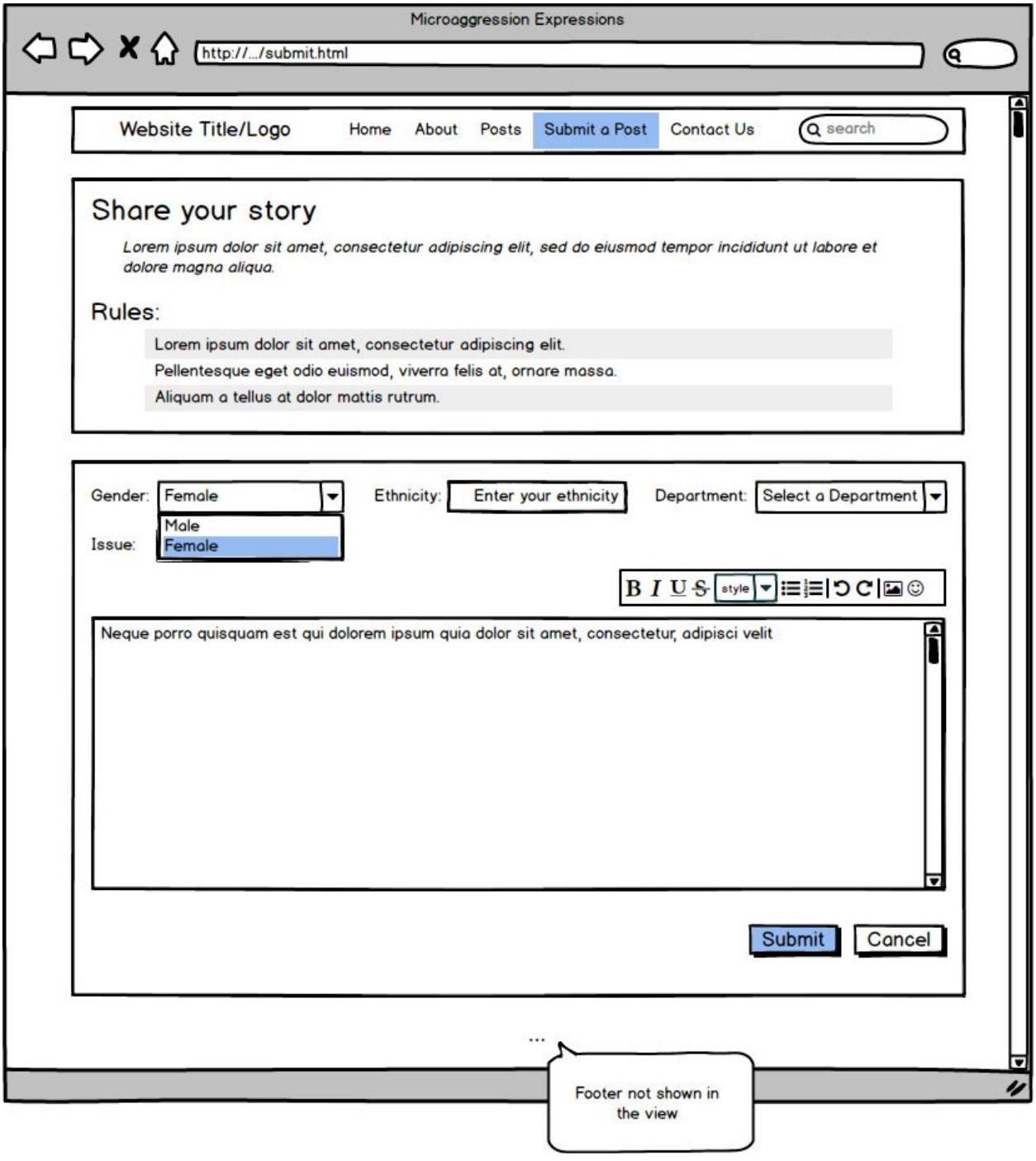


Figure 11: Share Your Story Storyboard

5.3.1.5. Front-end Development

We will use Foundation front-end framework which is a collection of HTML, CSS, and JavaScript to build upon the website. Part of our non-functional requirements is to make the website cross-browser, cross-platform, and cross-device compatible. See 4.4.1.1 for more information.

5.3.2. Back-end Functionality

The back-end of the project uses MySQL to store admin and moderator details, post data and metadata, and website contents. We have used PHP and MySQL to store and access the content in our database. We have five database tables as shown below:

Table Name	Primary Key	Other fields
Admin	UserID	Password, First Name, Last Name, Mobile, Email, Department
Moderator	UserID	Password, First Name, Last Name, Mobile, Email, Department
Posts	Post Number	Post, IP, Time Stamp, Date, Status(Published, Flagged, New), Gender, Ethnicity, Department, Type
Templates	Name	Data
Tweets	Tweet Number	Hashtag

Table 1 : Database Tables

The admin and moderator tables are used to store the login details and information about the admin and the moderators. The password has been encrypted to ensure security.

The posts table stores all the fields from the ‘submit a post’ form from the website. It also has a status field of type enum that can take values new, published, and flagged. All the recently submitted posts will have a status ‘new.’ The admin/moderator will review the newly submitted posts and update its status to published by selecting the publish post option in the manage new posts. If the post reveals the identity of an individual or a location, it will be marked as flagged and it will not be displayed on the website.

The templates table allows the admin to change the website’s contents without having to deal with the coding. The name field refers to the distinct and unique sections of the websites that the admin can edit in the future. The associated data field will store the website’s content.

The tweets table are used to store the hashtags that the admin would wish to display on the website. For example, all the tweets with #microaggressions or #sexist will be displayed on the view posts page of

the website.

The PHP backend will manage accepting the data from the submission form, flagging posts containing identifying information, inserting posts into the database, searching submitted posts, logging in to an administrative control panel, managing the website's contents, and adding or deleting moderators that is accessible only to the admin.

5.3.3. Use Cases

Use Case I - Submitting a post
Actor: Engineering faculty member at Virginia Tech
Stakeholder: Engineering Department at Virginia Tech
Primary Actor: Engineering faculty member
Secondary Actors: Other people affected by Micro-aggression, Virginia Tech
Preconditions: The actor has successfully filled out the form to submit a post
Main success scenarios: After completing the form, the primary actor clicks the "submit" button. The website processes the post and checks for anything that could identify an individual (for example: names, building names, room number, etc). The post has successfully passed the anonymity checker and has been uploaded to the database for moderation.

Alternative path(s):

The anonymity checker has determined that the actor included someone's name in the post. The website then notifies the actor and asks if he/she wants to modify the post or continue submitting this post even though it might reveal someone's identity. The actor decides to modify and resubmit the post. The website once again checks the post to verify for anonymity. The post has successfully passed the anonymity checker and has been uploaded to the database for moderation.

Use Case II - Moderating a post
Actor: Website moderator or admin
Stakeholder: Engineering faculty member
Primary Actor: Website moderator or admin
Secondary Actor: Other people affected by microaggression, Virginia Tech
Preconditions: Admin or moderator was able to successfully login on the Administration page of the website. A new post is waiting to be moderated.

Main success scenarios: The new post did not contain any sensitive information that could identify an individual. The moderator then clicks the “approve” button to add the new post into the website and make it viewable to the public.
Alternative paths:
The new post contained identifying information. The moderator removes this information, then approves the post.

Use Case III - Searching the Site
Actor: Any visitor to the site
Stakeholder: Visitor
Primary Actor: Visitor
Preconditions: The website contains entries of microaggressions.
Main success scenarios: The website returns a set of posts previously submitted by users. This set is relevant to the search and ordered appropriately.
Alternative paths: No relevant results are returned.

Use Case IV - Adding a moderator
Actor: Website admin
Stakeholder: Moderators
Primary Actor: Website admin
Preconditions: Admin was able to successfully login on the Administration page of the website.
Main success scenarios: Admin fills out the details of the moderator. After correct validation, the moderator profile gets added to the database.
Alternative paths: The moderator profile may already exist in the database. The admin will choose another userID for the moderator or cancel the adding of that moderator.

Use Case V - Deleting a Moderator
Actor: Website admin
Stakeholder: Moderators
Primary Actor: Website admin
Preconditions: Admin was able to successfully login on the Administration page of the website.
Main success scenarios: Admin fills out the userID of the moderator. After correct validation, the moderator profile gets added to the database.
Alternative paths: The moderator profile may not exist in the database. The admin will choose another userID or cancel the deletion of that moderator.

Use Case VI - Editing the website's contents
Actor: Website moderator or admin
Stakeholder: Engineering faculty member
Primary Actor: Website moderator or admin
Secondary Actor: Other people affected by microaggression, Virginia Tech
Preconditions: Admin or moderator was able to successfully login on the Administration page of the website.
Main success scenarios: Admin or moderator selects the option for changing the desired section of the website's contents. After making the changes, they select save. All changes are reflected on the website successfully.
Alternative paths: The website's contents did not get updated. The admin or the moderator may choose to retry or cancel the update.

5.4. Implementation

5.4.1. Overview

5.4.1.1. Description of Implementation

Server details: [See 4.1: Server Information for server details and login information] Apache, MySQL, PHP, PHPMyAdmin, FTP, and SendMail are installed

The URL for the site will, for now, is http://microaggressions.cs.vt.edu. The installation procedure will be as follows:

1. MySQL and PHP must be installed
2. The database must be moved from the previous install. If installed from scratch, the database must be recreated.
3. A configuration file included with the code must be changed to reflect the correct MySQL username and password, the site's root URL, and other relevant site details.

Front-end: We are using Foundation Framework[4]. This is a responsive front-end framework that provides a responsive grid and HTML and CSS UI components, templates, and code snippets, including typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. It is maintained by ZURB and is an open source project. We have also incorporated PHP along with this framework.

Twitter Widget: We are using a modified version of Tom Elliott's JQuery Twitter Feed[5] to display publicly posted tweets about microaggressions. This uses the Twitter Search API and requires a consumer key and access token to work.

Facebook Sharing: We are using Facebook's API[2] for sharing and this widget requires a Facebook App ID and tokens for authentication.

Anonymity Checker: We are using a simple JavaScript code for the anonymity checker. Currently, the checker checks every word in the post and matches it with a list of "flagged" keywords (such as building names and abbreviation) storied in an array.

Analytics: Currently, the codes for bigrams, trigrams, pie charts, and word cloud are embedded on the page where they appear instead of a separate JavaScript file. This is because these elements need to be connected to the database to get the data in real time (especially since we have incorporated a filtering feature).

Bigrams and Trigrams: We are using a modified version of Rob W.'s N-gram script[6]

Pie Charts: We are using Google Visualization API[7] to display the data in visual form

Word Cloud: We are using the D3 word cloud API created by Jason Davies[8]

Search: We are using the SQL "LIKE" condition to search throughout the posts

5.4.1.2. Points of Contact

Role	Name
Project Client	Josh Iorio
Database Implementation	Arunima Singh
Front end HTML/CSS	Cyndy Ejanda
Graphic Design	Brandon Falcone

Writing	Brandon Falcone
PHP Backend	Robert Wenger

Table 2 : Points of Contact

5.4.1.3. Major Tasks

We will be using an incremental model that follows the waterfall model in a sequential manner. We have defined our iteration period to be two weeks. Each iteration will have many releases or increments. Our first increment will include the core functionalities. Successive increments will include the addition of advanced functionalities or fixing the core and advanced functionalities. The final increment will include the microaggressions website.

Task 1: Select the mockup
Complete the final design of the website including the structure and aesthetics of the web pages. Photoshop editing software will be used to create the mockup representing the idealized look of the website.

Task 2: Create the web pages
Foundation framework will be used to design and build the web pages. This will be used in combination with HTML, CSS, and Javascript for front-end development. Our website will include the following webpages:

- ❏ Home
- ❏ About
  - ❏ Resources
- ❏ Posts
  - ❏ Analytics
- ❏ Submit a post
  - ❏ Thank you for your post
- ❏ Contact Us
- ❏ Search Results

We will also have the following pages for the administration panel:

- ❏ Login
- ❏ Dashboard

- ❏ Manage New Posts
- ❏ Manage Website Contents
- ❏ Manage Moderators
  - ❏ Manage
  - ❏ Add
  - ❏ Delete
- ❏ View All Posts

Task 3: Create the database

We will use PhpMyAdmin to store the admin and moderator information, and all data relating to posts including the text-only post, metadata about the post, and the website's contents. We will create five database tables: admin, moderator, posts, templates, and tweets. All of the fields in the admin, moderator, templates, and tweets tables cannot contain any null values. The posts table can have null entries for the department field. The 'status' field in the posts is of type enum with values: published, flagged, or new. The templates table will be used to store the website's contents and the tweets table will be used to change the hashtags.

Task 4: Create an admin/moderator login system

PHP and HTML will be used to create a login system, allowing the admin and moderators to sign in to perform administrative functions. The username and password will be compared to the associated values in the database table 'admin' or 'moderator' respectively. The admins are in charge of adding or removing moderators. The admin and moderators have the ability to classify posts that are deemed anonymous and the posts that have been submitted with flags by our anonymity checker. The admin and moderators can also edit the website's contents through the dashboard.

Task 5: Submit a post

The PHP server-side scripting language will be used for managing the acceptance of data from the submission form. PHP will be used in junction with MySQL queries to insert posts into the database. Before inserting the post to the database, the post will be validated. We will include validators in the 'submit a post' HTML page. We will also include a check box to ensure the user has read the rules for submitting a post. After a post has been submitted successfully, the user will be directed to a 'Thank you for your submission' page. We will then store the post to the database, where the status of the post would be 'new.'

Task 6: Create the anonymity checker
JavaScript will be used to create an anonymity checker to flag posts that contain identifying information. This will help ensure that posts will not identify those described within the post. The anonymity checker will analyze the text within a post to flag words such as names and abbreviations of all academic buildings at Virginia Tech. Each word will be compared to a list of academic buildings to determine whether the post should be flagged. If there are flags present after the user tries to submit a post, the user will be prompted to revise the post and re-submit, or else their post will be not published to the website. In the scenario where the user made use of a proper noun, the post would be checked by the moderator manually. Hence, we will guarantee anonymity of the post under any circumstances.

Task 7: Allow moderators to flag or publish posts
If the post reveals the identity of any person or place, the moderator will update the status of the post to be ‘flagged.’ The flagged posts will not be uploaded to the website. If the post meets our anonymity requirements, it will be published to the website.

Task 8: Display tweets
We will use the Twitter API to collect tweets. The Twitter API will be queried periodically to gather relevant tweets that are examples of microaggressions. These queries will be performed every time the view posts page of the website would be loaded. It will display the ten most recent tweets with hashtags set by the admin. For example, the ten most recent tweets with #microaggressions will be displayed.

Task 9: Share website contents to Facebook
One feature of the website is the ability to share posts on other social networking sites. There will be a linked share icon located on all the relevant website pages which will allow anyone to share the information in the web page to Facebook. We will use Facebook API to implement this feature.

Task 10: Filtering of posts
We will include filtering of posts based on gender, ethnicity, and department. We will use PHP backend and MySQL to display the required results.

Task 11: Create a tag cloud

D3 word cloud API will be used to create tag clouds that will be displayed on the ‘Home’ page, ‘Posts’ page, and ‘Analytics’ page. Words in the tag cloud will come from the most frequently occurring words contained in recently submitted posts. All the published posts will be scanned to collect the words.

Task 12: Enable N-grams analysis
We will use a script to get the most frequent bigrams and trigrams from all the published posts. We will also devise a way to not include the stop words in our analysis, such as ‘is the’ etc.

Task 13: Generate Pie-Graphs
We will use Google Visualization API to create pie graphs which would depict the number of posts based on gender and ethnicity. As we would filter the posts based on gender, ethnicity, and/or department, the associated pie graphs would also get updated.

Task 14: Allow search on the website
The site will provide search functionality for the posts in the database with status ‘published.’ After the user submits their search string, the PHP backend will query the database by using the MySQL ‘like’ to find all the posts that contains the user specified substring. Once the results are returned, the backend will divide the results into pages of 20 posts each and display the first page of results. If no such posts are found then it would display the ‘no results found’ message.

Task 15: Handle malicious software and submissions
We will use Google’s reCAPTCHA service to handle malicious and spam posts.

Task 16: Test and Deployment
The team members will perform usability tests by simulating the actions of real users of the website. By going through the motions of submitted a post, searching for posts, and navigating the website, team members will identify any existing errors and correct them in an iterative process. This task will be performed by all team members. The client and professor will also be asked to test features in order to identify issues.

5.4.1.4. Implementation Schedule

Completed Tasks as of February 24, 2016

- Identify the project requirements
- Brainstorm the design of the website
- Design the skeleton of the website
- Configure the web host
- Gather resources and content for the “About”, “Resources”, and “Contact” pages

February 24, 2016 - March 4, 2016:

- Enable users to submit posts anonymously
  - Task 1: Select the mockup
  - Task 3: Create the database
  - Task 5: Submit a post
- Set up the website
  - Task 2: Create the web pages

March 14, 2016 - March 29, 2016:

- Finish initial website with basic functionality
  - Task 6: Create the anonymity checker
- Set up the moderator system
  - Task 4: Create an admin/moderator login system
  - Task 7: Allow moderators to flag or publish posts
  - Task 8: Allow moderators to approve flagged posts

March 30, 2016 - April 15, 2016:

- Start working on advanced functionality
  - Task 9: Collect Tweets
  - Task 10: Share posts to Twitter and Facebook
  - Task 11: Create a tag cloud
  - Task 12: Allow search on the website
  - Task 13: Handle malicious software and submissions

April 16, 2016 - April 28, 2016:

- Finalize the advanced features

April 29, 2016 - May 3, 2016:

- Task 14: Tests and deployment

5.4.1.5. Security and Privacy

The anonymity of posts is an important feature of our project. The website will include several measures to ensure the identity of others is kept hidden. All the posts would be submitted anonymously to the website.

To prevent the person posting from being identified, the site will not ask about their name, email, or any personal information that could be traced back to them. Only general questions about the person posting will be asked to get information on their gender, ethnicity, and their department. The department field will be optional.

The content of posts can sometimes reveal the identity of others, especially if they identify a person by name or location. An anonymity checker will be used to alert the user if their post consisted of the names or abbreviations of any of the academic buildings at Virginia Tech. The person posting will be prompted to revise their post and remove these words if detected by the anonymity checker. Users will be allowed to submit posts that does not contain any flagged keywords, which will then require the approval of a moderator before being published to the website.

We will also ensure the security of our website by including encryption of admin's and moderator's password. We will handle malicious users by including Google's reCAPTCHA to our submission form.

5.4.1.5.1. System Security Features

To ensure website and user security, we aim to implement the following:

- We will handle malicious posts by using Google's reCAPTCHA service to protect the website from spam and abuse.
- We will include encryption of admin's and moderator's password to make the website more secure.

- Allow moderators to flag or publish posts
  If the post reveals the identity of any person or place (such as Room 1040 in Torg), the moderator will update the status of the post to be 'flagged.' The flagged posts will not be uploaded to the website. If the post meets our anonymity requirements, it will be published to the website.
- The anonymity checker
  JavaScript will be used to create an anonymity checker to flag posts that includes the names or abbreviations of any of the academic buildings at Virginia Tech. The user will be prompted to review their post since it consists of flagged keywords.

5.4.1.6. Problem Size

Microaggressions are a unique kind of insult in that most of the time such statements are so small and often ignored or downplayed. Because of this, it's unknown as to how many of these microaggressions occur on a daily basis. With this website, victims of microaggressions can share their stories about how microaggressions have personally affected them, instead of being downplayed. This could potentially attract the attention of a large portion of professors and staff in the College of Engineering. It's expected that not all professors will need or want to use this website; it's more likely that there will be a dozen frequent users, with the number of infrequent users estimated to be about 50.

5.4.2. Implementation Support

5.1.2.1. Hardware

The site will be deployed on a server maintained by the computer science department. This machine is a virtual machine with 2 cores, 4GB RAM, and 500GB storage running 64-bit CentOS 6.7. Apache, MySQL, PHP, FTP, SendMail, and PHPMyAdmin are installed.

The URL for the site will, for now, is http://microaggressions.cs.vt.edu. The installation procedure will be as follows:

1. MySQL and PHP must be installed

2. The database must be moved from the previous install. If installed from scratch, the database must be recreated.
3. A configuration file included with the code must be changed to reflect the correct MySQL username and password, the site's root URL, and other relevant site details.

Laptops will be used by each member of the project for carrying out the development of the website. Aside from laptops and the server, no other hardware will be needed.

5.1.2.2. Software

Programming Tools

- Foundation Front-End Framework
- Twitter API
- Facebook API
- Google Visualization API
- D3 Word Cloud API

Programming/Markup Languages

- PHP
- MySQL
- HTML/CSS
- Javascript
- Jquery

Operating Systems

- OS X and Windows for development
- Centos 6.7 for the server

Software

- WinSCP
- Notepad++
- Photoshop
- Adobe Dreamweaver

5.4.3. Documentation

The required documentation is the elaborate project description that focuses on the requirements, timeline, scope, client information, stakeholders, project

deliverables, the implementation plan, analysis and handling of risks, testing, project security, performance, and deployment.

5.4.3.1. Performance Monitoring

After implementing the many tasks of the project, testing will be done on the more complicated tasks to ensure features work as expected. The team members will perform usability tests by simulating the actions of real users of the website. By going through the motions of submitted a post, searching for posts, and navigating the website, team members will identify any existing errors and correct them in an iterative process. Implementation is a success when no more errors are detected after multiple iterations.

For performance monitoring, we aim for the moderator to classify all the newly submitted posts as flagged or not within 24 hours. The non-flagged posts will then be published to the website. We also need to device a way of handling and displaying the most recent posts. This can be achieved by comparing the post's submit date and timestamp. We will have ten most recent tweets and posts displayed per page.

5.4.4. Implementation Approval

5.4.4.1. Risks and Contingencies

In the event of an implementation failure of a certain task, the functionality of the task will be scrapped from the website. This will only occur for tasks that involve advanced functionality due to their difficulty in implementation or failure to meet the final deadline. A contingency plan will not be set for tasks involving core functionality, as these tasks are imperative to the website.

5.4.4.2. Acceptance Criteria

The website would provide a medium to post stories related to microaggression, which would then be stored in the database. The website should be able to efficiently validate all the posts. The moderator should also classify the posts as flagged or publish them to the website if the post met the anonymity criteria. This should be done within 24 hours. We need to be able to display tweets dynamically to the

website. The website should allow sharing of website's contents to Facebook. The website should also allow data analytics, such as display tag cloud, pie graphs, N-gram analysis, and allow filtration of posts based on gender, ethnicity, and department. Furthermore, the results of filtration must also be reflected on the tag cloud and pie charts. Most importantly, the website should ensure the anonymity of its users.

5.5 Prototype

5.5.1. Description of Prototype

Our prototype consists of the click-through website which simulates the core functionality needed for users to navigate the website, submit a post, and view posts, as well as the functionality for moderators to log in, view posts, and approve or flag posts. The scope for this prototype is limited to the user's ability to accomplish specific tasks as well freely navigating the website. The prototype consists of the pages to be included in the website as well as a framework for the content within each page.

5.5.2. Progress Report

5.5.2.1. Work Completed

The website is now fully functional and live.

Main website: http://microaggressions.cs.vt.edu/

Admin panel: http://microaggressions.cs.vt.edu/admin

Page	Work Completed	Future Work
Home	Displays most recent posts in carousel, tag cloud, sections can be modified via Admin panel.	None
About	Contents, formatting, added a video, sections can be modified via Admin panel	None
Resources	Updated list of resources, sections can be modified via Admin panel.	None

Posts	Displays published posts, filtering works, displays live tweets, tag cloud, intro and twitter hashtags can be modified via Admin panel, twitter.	None
Data Analytics	Bigrams, trigrams, removed stop words, pie charts, tag cloud, filtering works, intro can be modified via Admin panel.	None
Share Your Story	Users can submit a story, data validation of submitted post, reCaptcha for anti spam, simple anonymity checker that detects building names, ensures users read the posts before submission.	The anonymity checker needs to be expanded to detect names and limit for the number of posts per IP needs to be implemented. The intro for this page can be modified via Admin panel.
Contact Us	Contact form is fully functional. Mail delivery is working.	The anti spam feature still needs to be implemented.
Admin Panel	Managing posts, managing admin and moderators, managing certain sections of the main website are functional, Password encryption is also working.	A forgot password feature, enable multi deletion and publishing, add a better content management system.

Table 3: Breakdown of work completed per webpage

5.5.2.1.1. Derivations from the Project Plan

Our website will include the ability for moderators to obtain a forgotten password. When clicking on the “Forgot Password” button on the moderator login page, a pop-up window will appear for instructions on resetting their password.

We also included a “Submission Successful” page which will appear after users submit a post. Clicking on the submit button after filling out the post form will redirect the user to this page indicating their post was successfully submitted.

On the “View All Posts” admin page, there is an option to filter posts by gender, ethnicity, department, and issue.

Additionally, there is the ability to sort the posts by date as well as gender, ethnicity, department, and issue.

6. Refinement

6.1. Refinement Details

Link to the website:
http://microaggressions.cs.vt.edu/

Link to the admin/moderator panel:
http://microaggressions.cs.vt.edu/admin/login.php

The About page is designed to have a grid structure to improve the look and feel. The headers on this page are titled “About”, “What are Micro-Aggressions?”, “The Effects of Microaggressions”, and “Examples of Commonly Used Micro-Aggressions”. We decided to include the objective of the project in the “About” section. We have also included videos on microaggression in the “About” page. These YouTube videos are made by another group whom we have given credit to in the “Resources” page.

On the Post page, we included a header block of text which explains the different tools you can use to search and filter the submitted posts as well as a direct link to the post submission form.

The “Submit A Post” of page has been renamed to “Share Your Story.” The header block of text explains important information about posting such as when posts get published, and how to avoid submitting a flagged post. When filling out the form, the gender and ethnicity fields are required and the department field is optional. We have also included a check box for attesting to having read the rules for submitting a post. For security purposes, we have included Google’s reCAPTCHA to handle malicious posts.

The Contact page was refined to contain a message box for users to contact the admins, and subject line. It also requires the user to enter their email address in order to receive a response email.

6.2. Sample Outputs & Results

This section includes the sample outputs and results for the tasks that we have implemented so far.

Sample outputs for Task 5: Submit a post

All the posts submitted via the “share your story” webpage gets stored to the database. The submit form also has validations. On successful submission, the webpage displays a “Thank you” message to the user. For purposes of testing, we are also displaying the MySQL query to the webpage, but the final webpage would not include this query. Below are the screenshots depicting the post submission process:

Step 1: User fills out the post submission form.

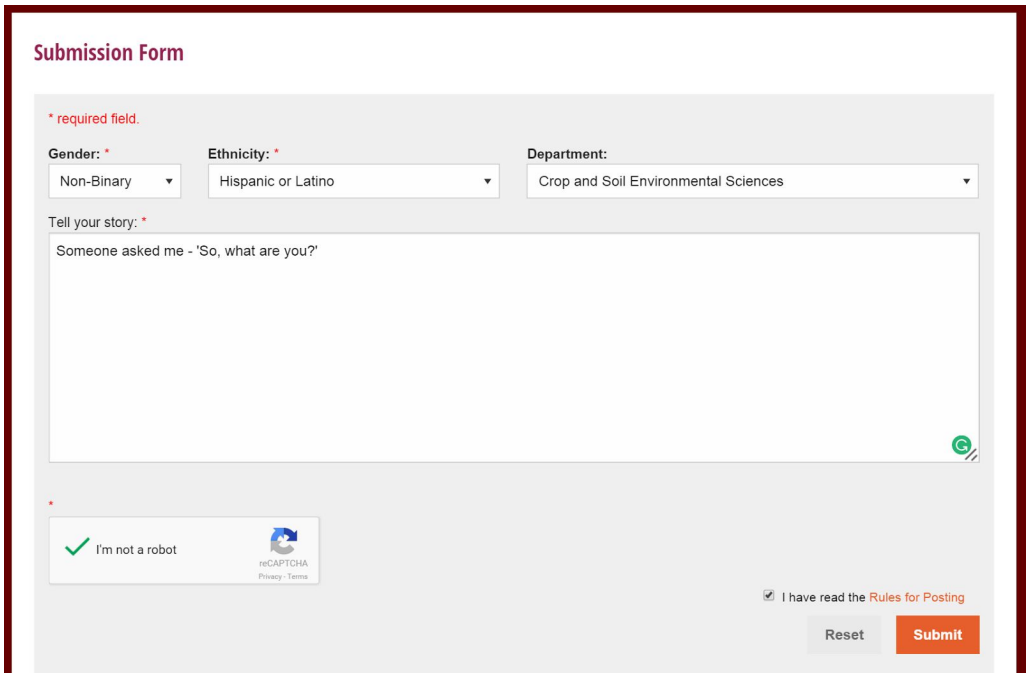


Figure 12: The form for submitting a post

Step 2: On successful submission, the webpage displays a ‘Thank you’ message.

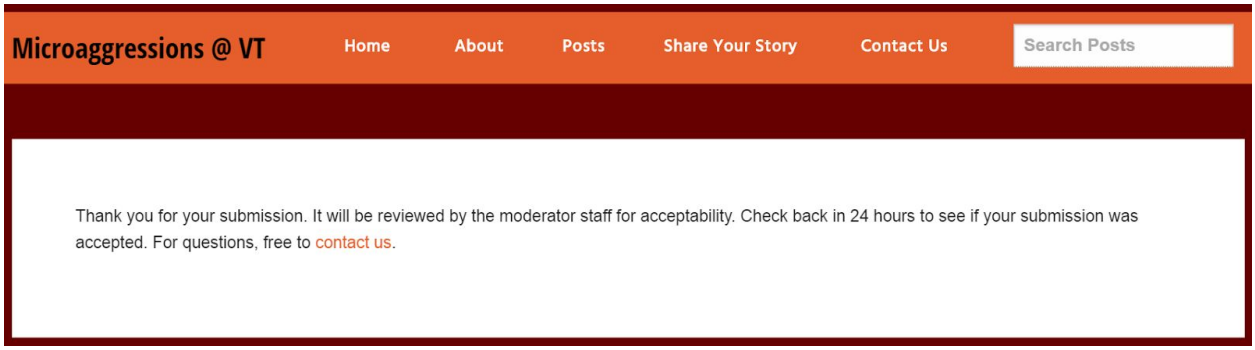


Figure 13: Confirmation message for post submission

Step 3: The post gets saved to the database.

Edit Copy Delete	72	testing recaptcha	24.219.193.121	2016-04-27 23:29:30	new	Female	American Indian or Alaska Native	Biological Systems Engineering
Edit Copy Delete	73	Someone asked me - 'So, what are you?'	73.31.41.47	2016-04-28 19:50:59	new	Non-Binary	Hispanic or Latino	Crop and Soil Environmental Sciences

Figure 14: The 'posts' table in the database

Sample Output for Task 6: Create the anonymity checker

The anonymity checker will display an alert message when the user hits submit if and only if the post contains any of the academic buildings names at Virginia Tech.

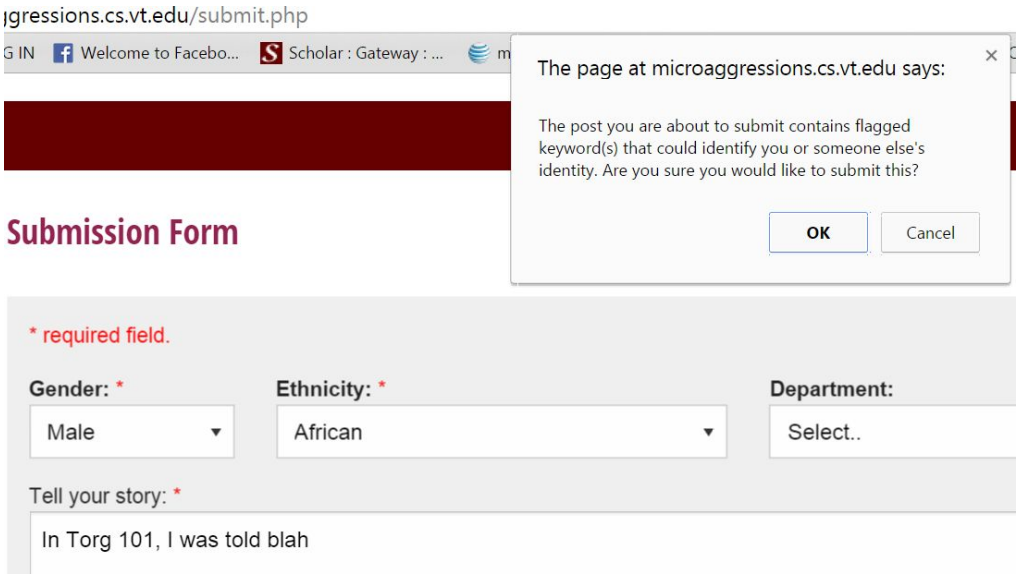


Figure 15: Anonymity checker

Sample Output for Task 7: Allow moderators to flag or publish posts

The admin/moderator can approve or flag posts if the posts would reveal the identity of a person or a specific location. If the admin selects the 'approve' button, it gets published to the website. Flagged posts will not be displayed on the website.

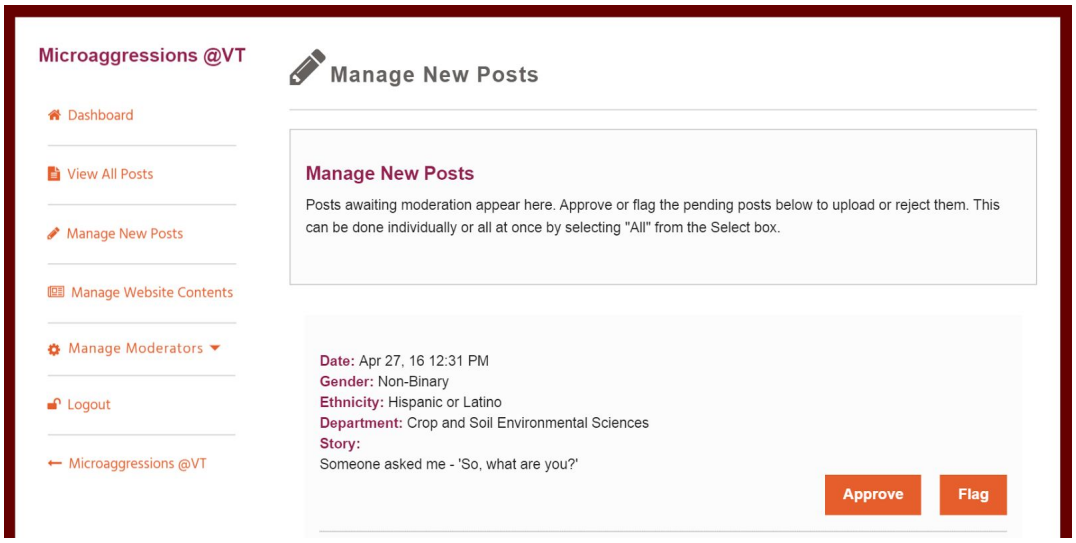


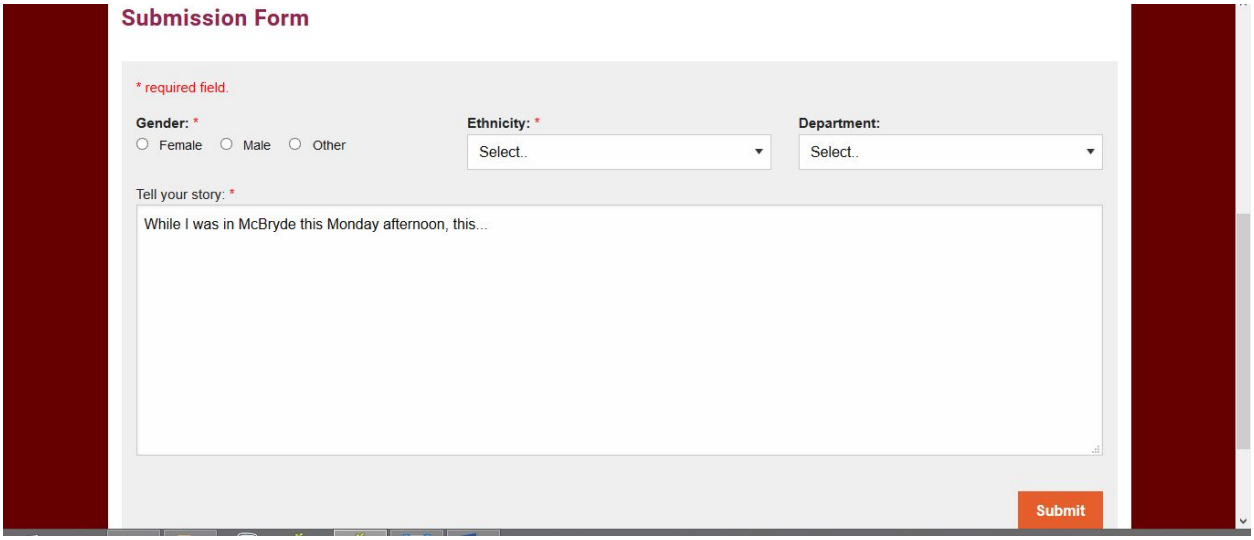
Figure 16: Manage New Posts

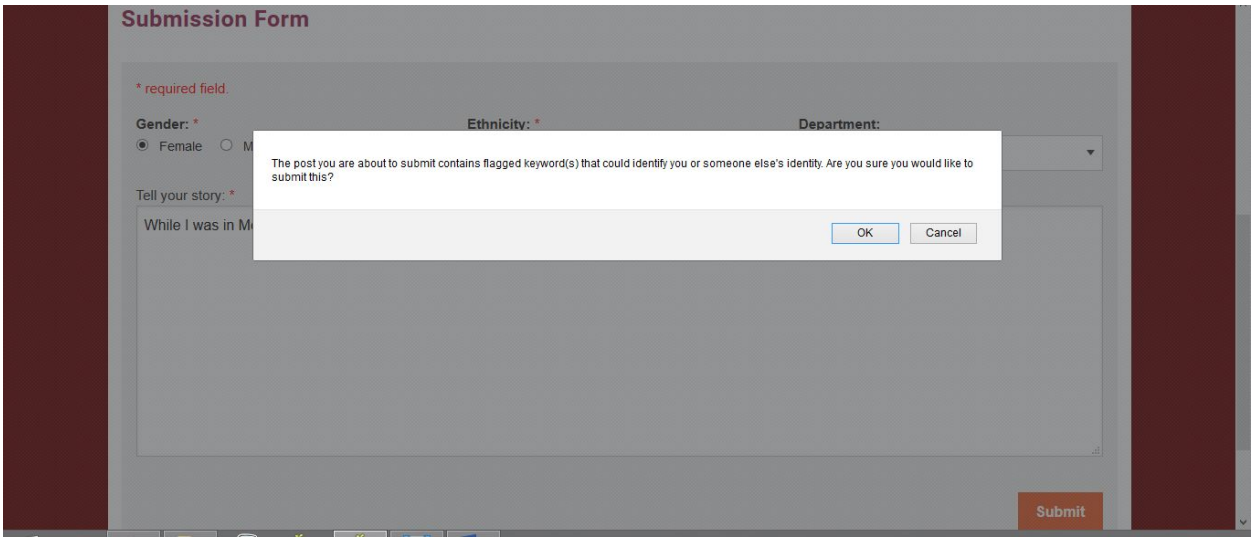
7. Testing

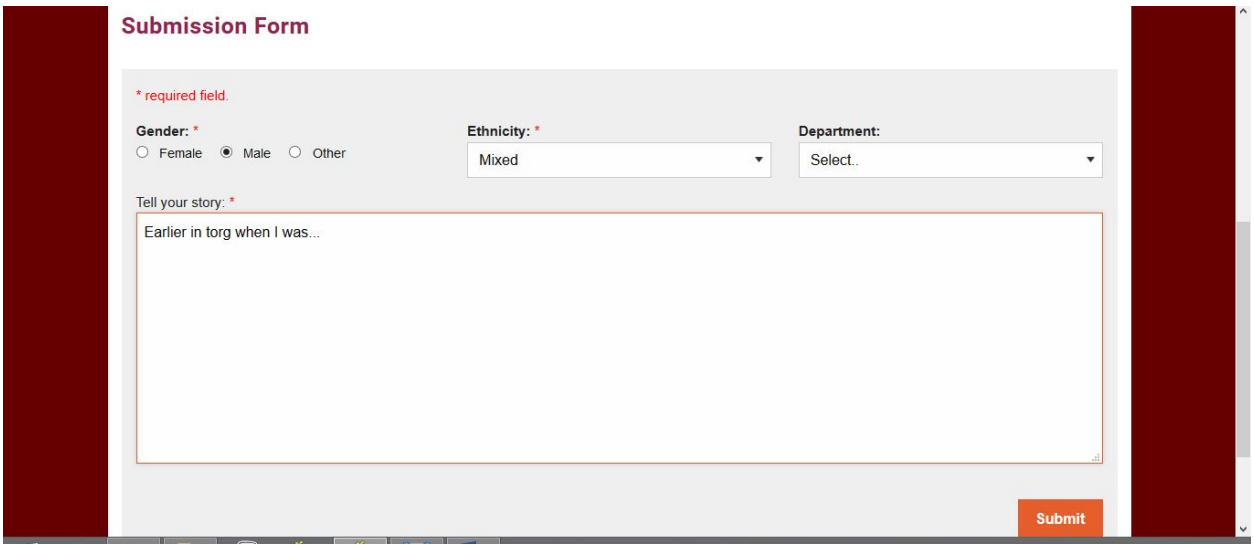
7.1. Unit Testing

7.1.1. Anonymity Checker Test

Currently, the anonymity checker can detect all the major buildings in Virginia Tech including their abbreviations. We tested this feature by creating a post that contains the name of a building and also by creating a post that does not contain any building information. Below are screenshots of what happens when you submit a post that contains building information.


Submission Form
* required field.
Gender: *
Female
Male
Other
Ethnicity: *
Select..
Department:
Select..
Tell your story: *
While I was in McBryde this Monday afternoon, this...
Submit


Submission Form
* required field.
Gender: *
Female
Ethnicity: *
Department:
The post you are about to submit contains flagged keyword(s) that could identify you or someone else's identity. Are you sure you would like to submit this?
OK
Cancel
Tell your story: *
While I was in M
Submit


Submission Form
* required field.
Gender: *
Female
Male
Other
Ethnicity: *
Mixed
Department:
Select..
Tell your story: *
Earlier in torg when I was...
Submit

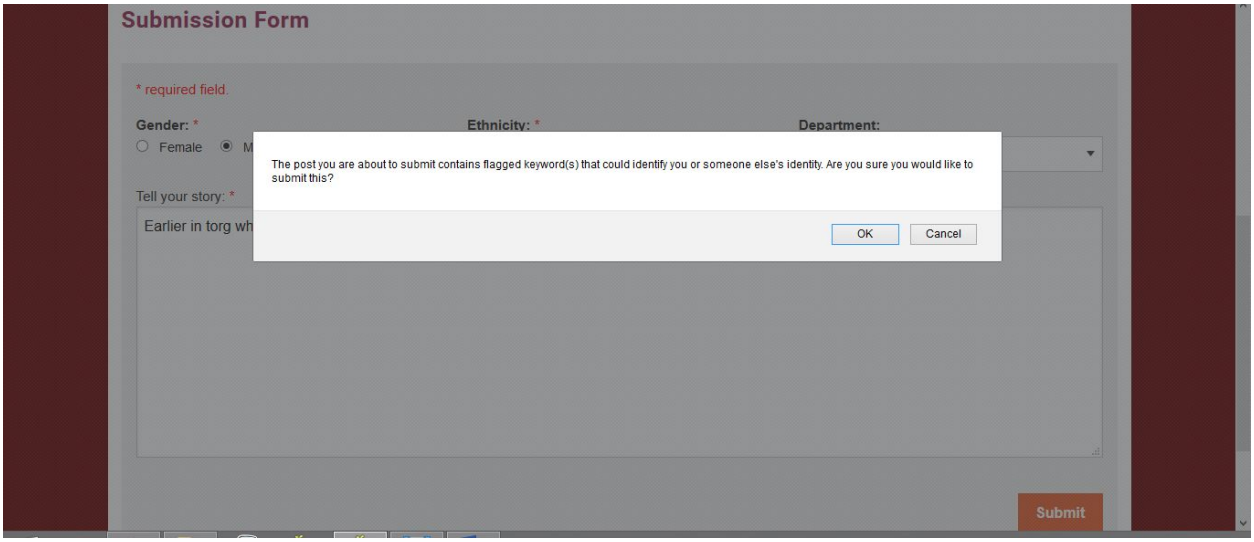


Figure 17: The anonymity checker

7.1.2. Combobox Consistency Test

Filtering posts is an important feature of our website that allows users to display posts about a specific gender, ethnicity, or department by selecting choices from comboboxes. These comboboxes can be located on the “Posts” page, “Share Your Story” page, and “View All Posts” admin page. The contents within each of these comboboxes must be the same across all pages. It was determined that fields were not consistent. During development, different genders and ethnicities were added and removed, but the changes were only made to the comboboxes of one page, and not all of them.

7.1.3. New Moderator Test

We performed a unit test on the add moderator functionality. A newly created moderator should have its data stored correctly with the ability to log in.

7.2. Integration Testing

We tested the combination of software modules involved in submitting a post. This will determine if the data in the post form is being correctly stored in the database. Likewise, it will determine if the posts are displayed correctly on the posts page. The software involved includes PHP to fill out the ‘submit a post’ form, MySQL to submit the post, PhpMyAdmin to view database contents, and PHP to display the post’s contents in the admin/moderator dashboard before publishing it to the website. Screenshots are

taken of the post form prior to being submitted, the corresponding data from the MySQL database after the post was submitted, and the corresponding post displayed on the “Posts” page after moderator approval. A sample output from our test is displayed below.

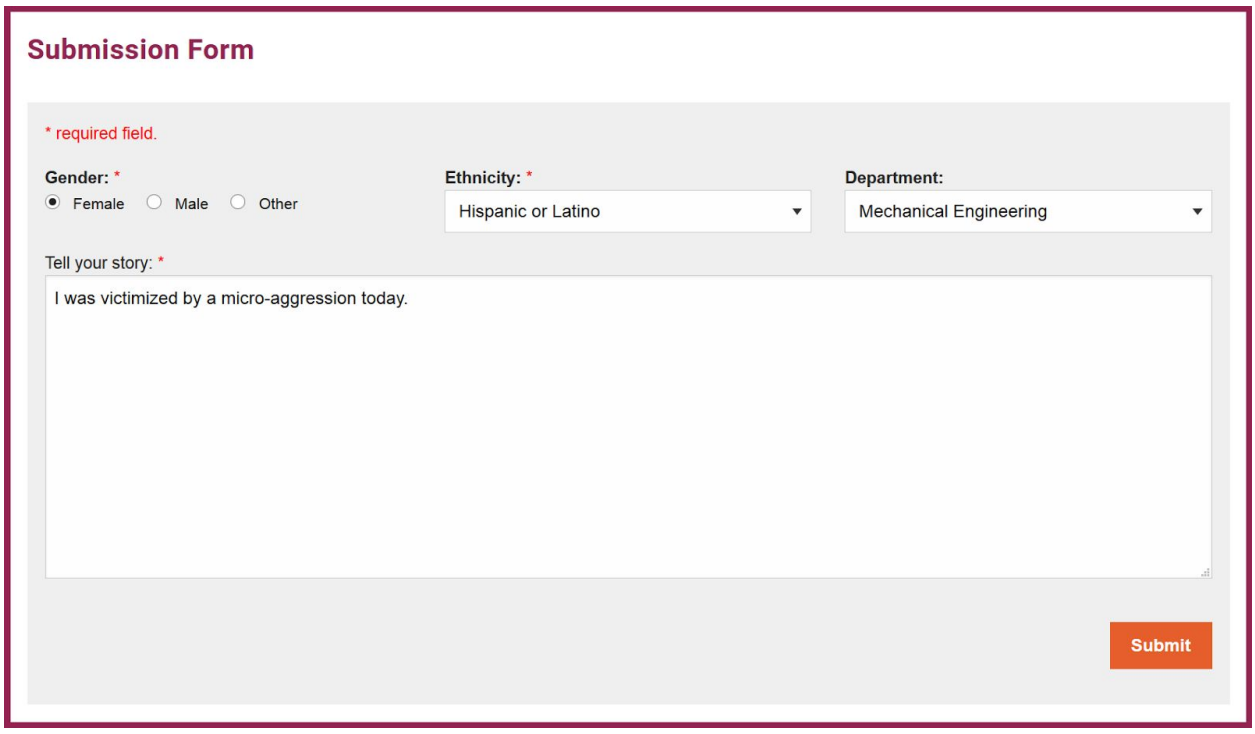


Figure 18: The sample post form input

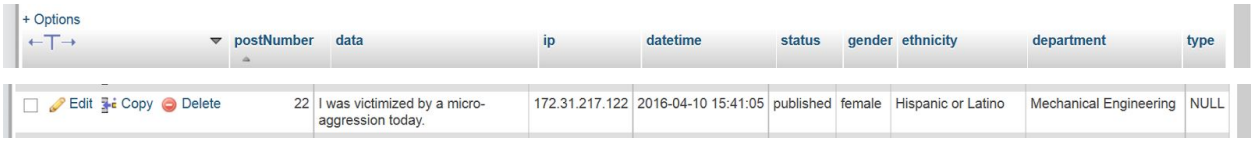


Figure 19: The database data from the post

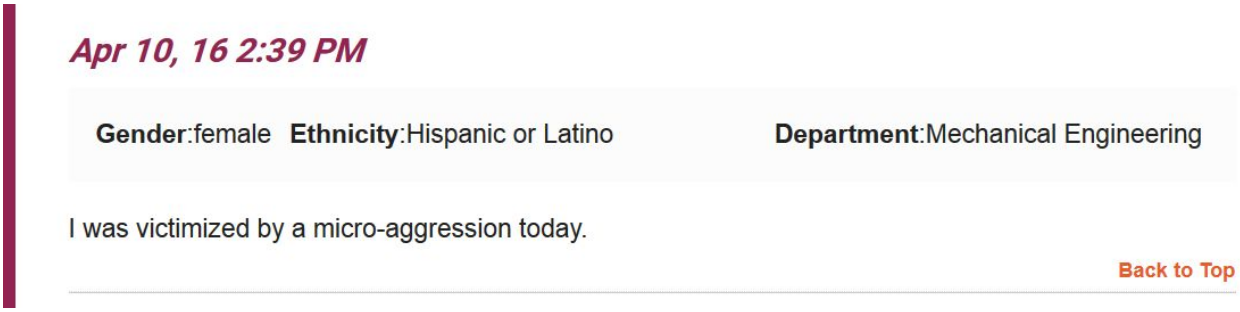


Figure 20: The post as displayed on the Posts page

7.3. Usability Testing

During the usability tests, we gathered critical incident data and think aloud data from our evaluators. This includes any errors the evaluators experienced and their auditory reactions while completing certain tasks. We asked the evaluators to think aloud and describe their thought process throughout their interaction with the website. After completing certain tasks, we asked the evaluators several questions about what they would have liked to see or do differently with the website.

7.3.1. Evaluator Tasks

The tasks involved in the usability tests will help determine if actual users are able to use and understand the website like we anticipated. Evaluators were assigned a worker role, either faculty member or evaluator, and instructed to complete tasks pertinent to each role. The tasks and the interactions users had while completing the task are outlined below.

Faculty member tasks:

1. Fill out a post form
2. Fill out a post from with flagged text
3. Fill out a contact us form
4. Locate the oldest submitted post

Moderator/Admin tasks:

1. Moderate pending posts
2. Add a moderator
3. Freely navigate the admin pages

7.3.1.1. Evaluator Tasks Results

Evaluator: Ben
Task 1 Results: He used the header bar to navigate to the “Posts” page. He looked around for the post form and followed the link on the right side of the page that took him to the “Share Your Story” page. He was confident that he was on the right page to fill out the form. He first read the rules for posting before beginning to fill out the form. He selected his gender, ethnicity, and department. Then he entered into the textbox his

story. He reviewed his post one final time to make sure everything was correct before hitting submit. The anonymity checker did not detect any errors. He briefly read the “Thank You” message.
Task 2 Results: Since the first task did not result in the moderator detecting unanonymous words, he was instructed to write a post that contained the name of a building on Virginia Tech campus. He filled out the post form just like he did in task 1. The user hit submit and was altered by the anonymity checker. He hit “cancel” on the pop-up window, and removed the building name from this post. He then clicked “Submit” and did not read the “Thank You” message.
Task 3 Results: He used the header bar to navigate to the “Contact Us” page. He quickly entered his name, email, and selected a random option from the combobox. He filled out the textbox with his message and hit submit. He had no difficulty completing this task.
Task 4 Results: He used the header bar to navigate to the “Posts” page. He noticed that, by looking at the first few posts, they were arranged in chronological order. He scrolled to the bottom of the page and clicked on the next page where he located the oldest post.
Task 5 Results: He clicked on the “Manage New Posts” admin page. He read the first post at the top of the list and clicked on the “Approve” button next to it. He correctly approved the post, but the screen glitched out into an unusable state where only certain text was readable. He noticed that the website did not load correctly, so he decided to refresh the page.
Task 6 Results: He clicked on the “Manage Moderators” link which opened up 3 sub-links to click on. He clicked on the “Add Moderator” sub-link. He quickly filled out the form with no difficulties.
Task 7 Results: The used navigated to the “View All Posts” admin page and noticed that there was not an indicator as to which of the posts had a flagged status and which had an approved status. The user navigated to the “Home” page by clicking on the “Microaggressions@VT” link. After navigating to the “Posts” page, he spent a fair amount of time reading through the tweets. He was very pleased with this feature.

Evaluator: Zack
Task 1 Results: He used the “Share Your” story button underneath the carousel to navigate to the post form. He quickly skimmed through the rules for posting before filling out the form. He selected his gender, ethnicity, and department. He entered his story in the text box and pressed submit. The anonymity checker reported that there was unanonymous content in his post when in fact there was not. This created confusion for the user who was unsure about what was causing

the problem. After clicking “ok” in the anonymity checker’s checkbox, he did not read the “Thank You” message. After submitted the post, he navigated to the “Home” page to see his post on the carousel. He failed to understand that the post would need to be moderated for approval before being uploaded to the website. This was attributed to his failure to read the Rules for Posting as well as the Thank You message.
Task 2 Results: See Task 1 Results.
Task 3 Results: He clicked on the “Contact Us” link at the top of the website to navigate to the contact form. He filled out his name and email, selected the issue of this message, and filled out his message. He clicked the submit button. No errors were reported.
Task 4 Results: The user clicked on the “Home” link at the top of the page to navigate to the “Home” page. He then clicked on the “View All Posts” button that he noticed early in the user test to navigate to the “View All Posts” page. He then scrolled through all of the user submitted posts on the first page, and then clicked on page 2. The successfully located the first post that was submitted.
Task 5 Results: The user started from the “Dashboard” admin page. He clicked on the “Manage New Posts” page. He clicked the “approve” button next to the first post he saw. After approving the post, the page refreshed causing text to shift to the right and the organized structure of the page was temporarily lost. At this point the page was unusable and the user was forced to refresh the page. After refreshing, the page went back to normal. The post that he previously approved was removed from the list of posts to manage.
Task 6 Results: He clicked on the Moderator link which opened up 3 sub links. He then clicked on the “Add Moderator” sublink. He was very familiar with filling out a basic form and completed this task with ease. He clicked submit.
Task 7 Results: When looking at the “View All Posts” page, the user suggested adding a field to each post in the list that described whether the post was approved or flagged. Additionally, he suggested adding a field to each post indicating which moderator was responsible for approving/flagging the post.

7.3.2. Questions for Evaluators

Evaluators were asked several questions about their experience using the website and completing the tasks. The questions and responses are outlined below.

Questions:

1. “Was the website easy to navigate?”
2. “Was the anonymity checker helpful?”
3. “Is there anything you would like to see improved?”

Question 1 Responses:
When asked, “Was the website easy to navigate?”, evaluators responded that they were satisfied with navigation when completing tasks. Users had no issues with the organization used and structure of the pages.

Question 2 Responses:
When asked, “Was the anonymity checker helpful?”, evaluators responded that the pop up message was confusing and not helpful, especially when the anonymity checker incorrectly flagged certain words. Suggestions were made to make the anonymity checker’s pop up message more informative so the user knows what information to correct in the post.

Question 3 Responses:
When asked, “Is there anything you would like to see improved?”, evaluators responded that the important information in the rules for posting should be made more visible. This includes making the text bigger and bolder. On the “View All Posts” admin page, evaluators suggested showing which moderator approved/flagged each post including a reason for why the post would be flagged.

7.3.3. Important Findings

Users did not read the rules for posting. Users claimed that the text was too small to read in certain parts of the website, such as the rules for posting and the “thank you” message that appears after a post was submitted. The size of the font could be a reason for users not reading certain text. This could also be caused by the small screen size of the computer used for the testing. It’s important to note that the website does accommodate variable sized screens.

There is a bug in the process of approving posts on the “Manage New Posts” admin page. Clicking “Approve” on pending posts has an unwanted side effect where the content on the page is arranged in a cluttered manner.

7.4. Stress Testing

We performed many system stress tests to ensure our website does not crash and functions as intended.

7.4.1. Character Count Stress Test

When it comes to filling out the post form, we made the assumption that users would be conservative with the amount of text being entered in the input textbox so we did not include a feature that would limit the number of characters. Furthermore, we did not consider the possibility of malicious users entering text in the textbox that contained, for example, more than a ten-thousand characters. We performed multiple stress tests by submitting posts that contained up to 100,000 characters. The website was able to handle a post of this magnitude. After conducting this test, we considered if we should set a limit to the number of characters contained in a post. However, the client deemed that this was not necessary.

7.4.2. Character Sequence Stress Test

We tested what would happen if users submitted a post that contained atypical characters. We used a pre-existing text file found online that consisted of interesting sequences of characters for testing purposes. We copied the characters contained in the file and pasted it into the input textbox as part of the post form. The website and database was able to handle this input and was also able to display the contents of the post on the “Posts” page just like any other post.

The file containing the characters used for the stress test:
https://www.cl.cam.ac.uk/~mgk25/ucs/examples/UTF-8-test.txt

7.5. Security/Privacy Testing

The login system for the admin as well as the moderators have encrypted passwords. In our project, the admin will be our client, Josh Iorio. After consulting with him, we have concluded that we’ll not be implementing a ‘forgot password’ mechanism. In case the moderators forget their password, our admin will delete that moderator and recreate his profile by selecting the ‘add moderator’ option in the dashboard. Consequently, we will not block a moderator’s profile on successive login failures. We have tested our admin/moderator login system by using incorrect username and/or password combination, and we were redirected to the login page. We also tried accessing other pages of the admin/moderator dashboard by not logging and simply using the URL, once again we were redirected to the login page.
The ‘manage moderators’ option in the admin/moderator dashboard is only visible to the person logged in with the role of an ‘admin.’ We logged in as a moderator and we

were denied the access to the ‘manage moderators’ option.
In order to ensure that posts remain anonymous, we are not collecting any user information that would reveal their identity. Hence, we do not have any security features for the front end. However, we will be limiting the number of posts per IP to be fifteen per fifteen minutes. This helps us in handling the malicious posts.

7.6. Progress to Date

The Tag Cloud is displayed on the “Home” page and “Posts” page which shows the most frequently occurring words in user submitted post. A different Tag Cloud is displayed on the newly created “Analytics” page which shows the most frequently occurring words based on filtering options.

The “Analytics” page has been created. This page displays bigrams and trigrams of user submitted posts, as well as the Tag Cloud mentioned previously. A pie chart illustrates the proportion of posts that have been submitted by males and females. Filtering works on the analytics page such that the digrams, trigrams, tag cloud, and pie chart reflect the filtering options.

Search on the website has been implemented. Entering a word in the search bar located at the top of the website displays a list of results where the searched word occurred in submitted posts. The number of posts containing the searched word is displayed as well each of the matching posts.

Tweet collection has been implemented. Tweets related to the hashtag #microaggression are displayed in the “Posts” page. Tweets are displayed on the website and not saved to the database.

The “Dashboard” admin page displays the number of new posts awaiting moderation, the number of posts that have been approved by moderators, and the number of posts that have been flagged by moderators.

The “Manage New Posts” admin page displays all posts awaiting moderation. Moderators are able approve or flag each post. Posts that the moderator has approved are added to the website as well as to the list of submitted posts on the “View All Posts” page. Flagged posts will also be added to the “View All Posts” page.

The “View All Posts” admin page now displays all posts that have been submitted. This includes flagged posts.

The sidebar for the pages accessible by the moderators has been edited to include sub-tabs under the “Manage Moderator” tab. The “Add Moderator” sub-tab directs you

to the page for adding a moderator. The “Delete Moderator” sub-tab directs you to the page for removing a moderator. There is also a sub-tab titled “Manage” which provides more information on managing moderators, as well as a list of all moderators.

The “Contact Us” page is complete. We have created the contact form, and the server successfully emails the contact listed in the settings file by using the PHP ‘mail’ function.

A new feature has been implemented so an admin is able to edit the content on the website. This content refers to the text that appears in the “About”, “Contact Us” and “Resources” pages. In addition, the administrator has the ability to dynamically choose what hashtags to include in the tweet collection. The purpose of this feature it to allow administrators to make changes to the website’s content throughout the existence of the website.

ReCaptcha has been implemented on the “Share Your Story” page to establish that the poster is human and now spam.

8. Lessons Learned

8.1. Timeline/Schedule

The first meeting with our client provided us with a better understanding of the impact this project would have on the University’s Engineering department. We discussed how the website would be implemented to accomplish this goal given the short deadline. Our ambitious group decided on a number of features to include in the website with the possibility that the more advanced features would not be implemented in the given time. Eventually we reached a point where we were falling behind the schedule we had created, and needed to push back the implementation of certain advanced functionality. It was very important that we introduced the possibility that certain “extra” features would be unimplemented at the beginning of this project in order to meet expectations for core features.

8.2. Problems

A problem we encountered involved the assurance that all submitted posts would remain anonymous. The anonymity of posts was the most important aspect of the website, as described by our client. To meet this goal, we came up with a mechanism, called an anonymity checker, that would read a person’s post and flag certain words that revealed one’s identity or the identity of others before the post was submitted. This

allowed the user and the website to know that the post contained sensitive information which would result in the post not being uploaded to the site. However, this turned out to be more complicated than anticipated. We determined that certain words could not appear in the post such as proper nouns, room numbers, and building abbreviations. It was a difficult task to detect sensitive information without errors due to many factors such as, proper nouns of people and places that were left lowercase, the appearance of the word “Room” next to other words that were numbers, and abbreviated building names. The anonymity checker would not be a foolproof method for ensure that post remained anonymous.

Another problem we encountered was trying to avoid having website features that could be a microaggression in and of itself. When filling out a post form, users are asked to select their gender from only 2 options, male and female. This was in fact a microaggression because it assumes that the user chooses to identify as either male or female and completely ignores the plethora of other existing genders.

8.3. Solutions

Our solution to the anonymity checker problem was solved by a new method of approving posts. All submitted posts would be approved by the moderator before being uploaded to the website. By having a real person read all submitted posts for sensitive information, we ensured the anonymity of everyone involved.

To correct the microaggression that was part of our post form, we added a third gender option, non-binary. Non-binary is a better word to use than “other” in identifying a person’s gender. This gives user the ability to choose their gender respectfully while maintaining the ability to filter posts using a pre-determined lists of genders.

8.4. Future Work

The remaining work to be done on the website consists of the advanced functionality of the website as well as some additions and tweaks that need to be made to core features.

- Anonymity Checker - Currently, the checker only detects building names and abbreviations, in the future, adding the ability to recognize names and additional keywords would be helpful. In terms of scalability, the current implementation will not be as efficient if we are going to deal with posts with a lot of text. We need a better and more efficient algorithm.
- Search - The search feature works well and can efficiently search through the post but not throughout the entire website.
- Security

    - Share Your Story - in order to limit spam and other attacks, we need to limit the number of posts per IP address. Currently, we only have reCaptcha.
    - Contact Us - the contact form needs reCaptcha and other anti spam features as well
- Sharing - Add the ability to share individual posts in social networking websites
- Admin Panel
    - Implement Forget Password feature
    - Add a better way to manage the contents of the website without making the client deal with any HTML code at all. Possibly implement a WYSIWYG (What You See Is What You Get) editor.
- Analytics - Implement a better n-gram algorithm that displays a more meaningful data. Apply natural language processing and machine learning.
- Others - The user tests provided useful information for refining the website. The rules for posting will need to be adjusted for clarity. The post form will include more gender options in the gender combo box. User suggestions will be considered further for implementation in the near future. More usability tests will be conducted in the future when features of the website near completion. There will be improvements to the usability tests that focus on understanding what is presented to the users.

9. Acknowledgments

9.1. Client Acknowledgements

Name: Josh Irorio
Email: iorio@vt.edu
Title: Principal Faculty in the Myers-Lawson School of Construction

Our client, Josh Iorio, assisted in guiding us through the development of the website. He also helped design our poster for VTURCS.

9.2. Professor Acknowledgements

Name: Edward Fox
Email: fox@vt.edu
Title: CS4626 Professor

Gratitude goes to Professor Fox for helping to provide us with a server. He made the process very easy for us to set up the server.

9.3. CS Department Acknowledgments

The CS Department deserves recognition for supplying us with the server on which all website content and data will be stored. 500GB was allotted to us for use on this project.

10. Works Cited

1. Garibay, Juan C. "Diversity in the Classroom Booklet." *UCLA Diversity & Faculty Development*. UCLA Diversity & Faculty Development, 2014. Web. 3 Feb. 2016. <https://faculty.diversity.ucla.edu/resources-for/teaching/diversity-in-the-classroom-booklet>.
2. "Share Button - Social Plugins - Documentation - Facebook for Developers." *Facebook Developers*. Web. 10 May 2016. <https://developers.facebook.com/docs/plugins/share-button>.
3. "Foundation Framework." Foundation Framework for Sites. Web. 05 May 2016. <http://foundation.zurb.com/sites/docs/>.
4. "JavaScript & AJAX | Web Dev Door." Web Dev Door JavaScript AJAX Category. Web. 05 May 2016. <http://www.webdevdoor.com/javascript-ajax/custom-twitter-feed-integration-jquery>.
5. "Rob W.'s N-gram." Rob W.'s N-gram. Web. 05 May 2016. <http://fiddle.jshell.net/WsKMx/>. http://stackoverflow.com/users/938089/rob-w
6. "Google Visualization API Reference." Google Developers. Web. 05 May 2016. <https://developers.google.com/chart/interactive/docs/reference>.
7. "D3 Word Cloud API Created by Jason Davies." *GitHub Jasondavies/d3-cloud*. Web. 05 May 2016. <https://github.com/jasondavies/d3-cloud>.