Group Project

Title

User Blog Website
prepared by
Oatmeal Raisin Web Development Group

Course

CS 4880

Section

Summer19

Due Date

August 07, 2019

Team Members

Milush Yanev
Alvin Heng
Reed Schrier
Yan Huang
Gabriel Bueno

Table of Contents

1. Problem Description

Our company got hired by Mr. Edwin Rodriguez to create a User Blog Webservice that will connect people and their views of any subjects. The website service will be a User Blog Website for any members with a valid email address. It will be designed to bring users together by creating different blogs with no restrictions about. More specifically this s

ervice is designed to allow an admin to manage communication inside a group of bloggers and articles posted on the website. Per asking from Mr. Rodriguez the admin part of the website will be disabled in the beginning and he reserves the right to ask us to design in within a specific time period, at no additional charge. The software will facilitate a username, which will be correlated with a valid email address entered. The maximum number of users is not defined but there is an option to extend or shrink it he after the website is deployed. The system will contain a relational database containing username, email, password, and number of blogs. There must be a Login Page where a user can enter his/her credentials and use our services. In order to do so he/she must complete the sign-up process. We are responsible for creating a way to verify if user had already been registered or not, password or email is correct. There must be a Sign-Up page where new users can register and proceed to using the website. A login verification must be completed in order for the system to know if the user is logged in or not at all time. The team had agreed to create a comment option for users to be able to communicate with each other by leaving comments under posts. There will be an option for a user to dislike or like a certain post. Every user has his own Profile page, where he can add his name and biography if he wants. This option is only optional, and the service will continue with or without in. The user will be able to see his posts on a

separate box from the personal one. The background theme should be the same as the webservice – Oatmeal and Raisins in any variation and form. Our company is responsible for creating the database and merge through a company that will be able to store it. A proper domain name will have to be purchased and linked to our service for external usage.

2. Requirements

There are several Requirements that must be met for the project to be completed. In order for the audience to better understand the webservice, the functional requirements will be stated first.

2.1 Functional

The first and most important is the Login User and verification - All users must have an account in order to create blogs, post a comment, and like or dislike posts. All accounts are created using an email and a password with a minimum amount of characters. Emails must be verified for the account to be usable. Users without an account can only view public posts and blogs. Sign up page must be created to create and store the user's information in the database for a future log in. A Profile page must be created. Each user with account will have a profile of their own where it shows their personal information, the number of posts and comments that the user has posted. The users will have the option to use their personal information from their profile. Next step is creating a Blog page - Users can create multiple blogs that must have at least one post. Blogs will be shown and listed in the user's profile page. Each Blog will have Posts - Users can post text with a maximum of 200 characters. All posts must be a part of a blog they have created. A comment option must be created - Users with an account will be able to comment on posts with a maximum of 200 characters without being able to format the text. Users without an account will not be able to view comments, posts, etc. until an account is created. A Like Button must exists - Users with an account will be able to like other users' posts.

A dislike Button must exist - Users with an account will be able to dislike other users' posts. A count of likes and dislikes will be provided for users. Back button will provide the client the option to go back

2.2 Non-Functional

The Non-Functional Requirements are Blog Article Count – The owner has the right to ask the development team to display or not such count which occurs in the back of the system. Database Storage will be provided from an external company such as Google, AWS, etc. Accessibility will be open for a maintenance team. All Documentation will be inside the files. The capacity of users will be set up to a certain amount with the option to change at any time. Modifiability will leave the option for modifications of daily basis without interfering users. A back up can be conducted on certain schedule.

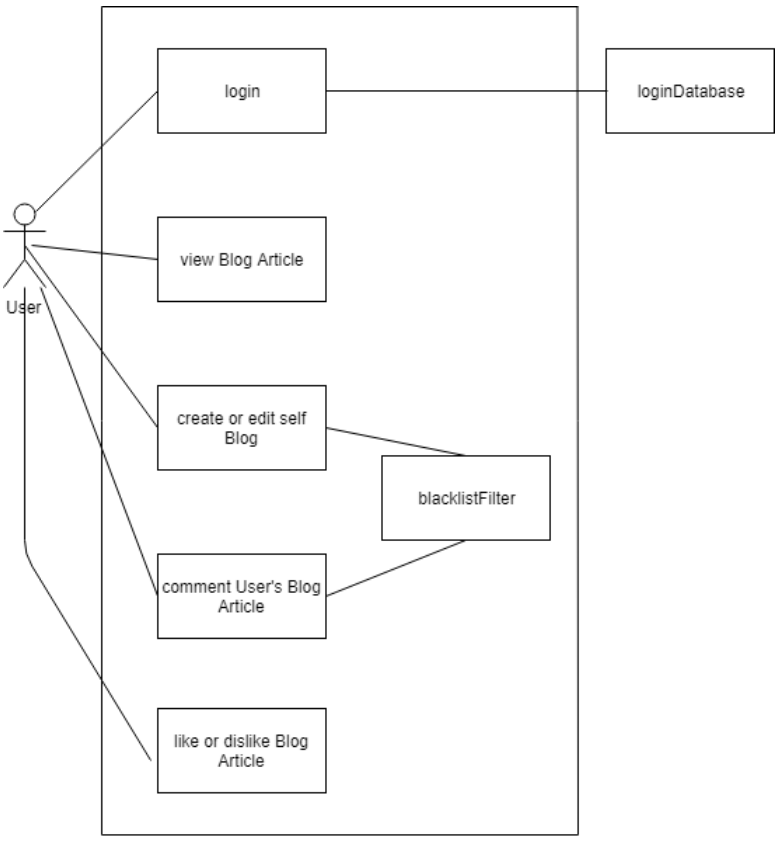
3. Overall solution Approach

In order to satisfy the client's requirements, Oatmeal Raisin Web Development Group will have to use Maria database services. The reason for this is that it is more reliable, cheaper and easier to maintain. In the original contract, the company will create a web service that will hold up to a thousand people with the option of expanding that only after future negotiations. For the frontend, the company will create HTML files from scratch and bootstrap from a library, Django for middleware, and MariaDB for backend. The process of creating the project was interesting and intense. First a proper Service Requirements Specification document had to be created by YanMan "Lilli" Huang and Milush. After agreeing on terms with the client, the process began. Second, we created a local database to test if it is storing correctly. The next step was to use Django Database services with Python, create multiple calls and see the result. The process of

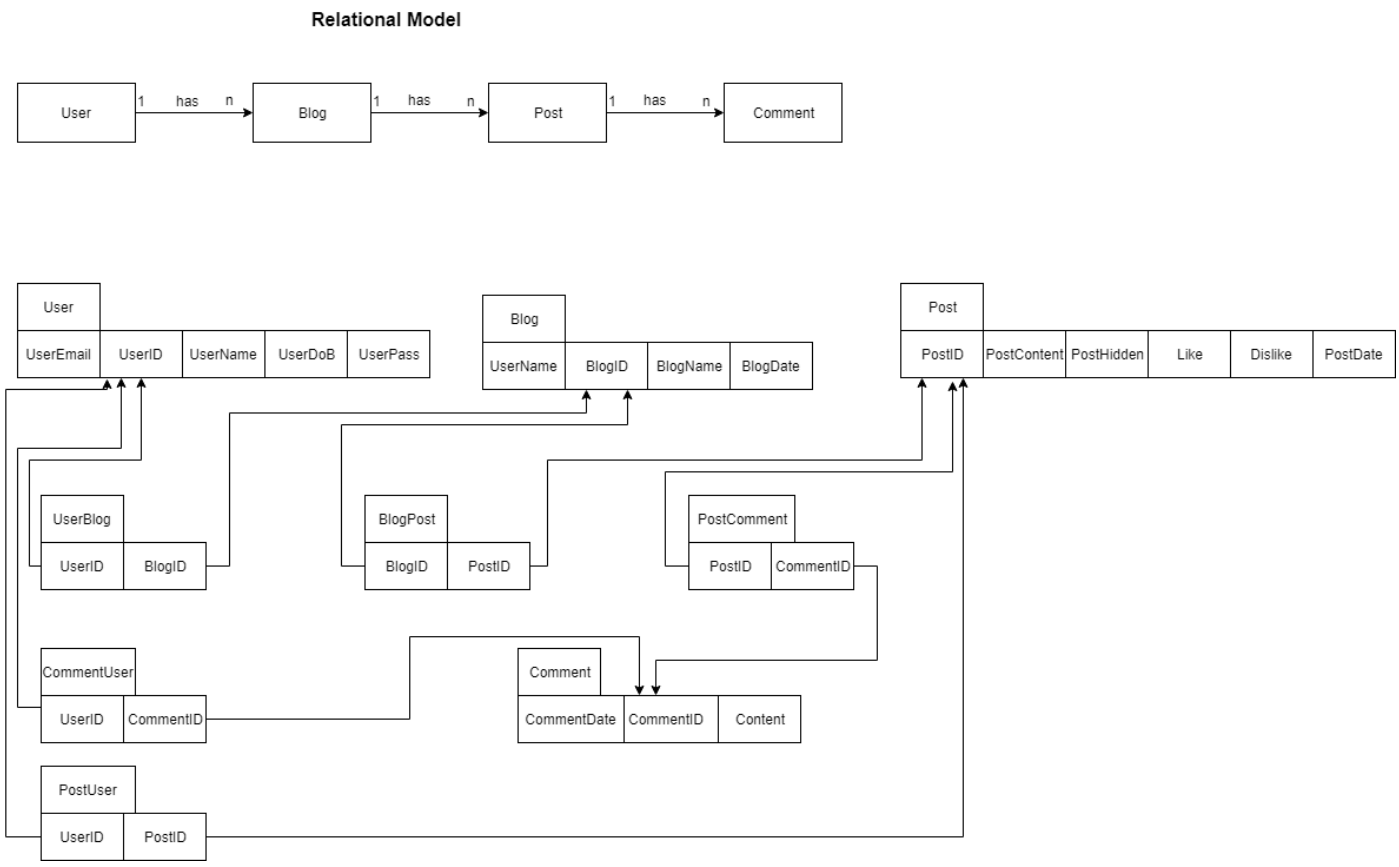
putting all this together was interesting and challenging but it was perfectly executed from Reed and YanMan "Lilli" Huang. One of the most important part was agreeing on the ER and Relational Models which was done in a day. After doing this he created a way for all the members using a proxy and working side by side with them until the desired result was accomplished. The next step was to purchase a domain and relate our work with it. Again, Reed was able to do it in timely manner. After everything was set up correctly our job was to make sure that everything, we promised will be delivered so we split up the work. YanMan "Lilli" Huang and Reed were responsible for Blog and Posts part. Milush and Alvin were supposed to create the front end of the project and make sure UI is working. Gabriel's Job was to create the Profile. To ensure better communication Alvin and Milush decided to work together in executing step by step by only Alvin pushing working code from his side. This helped with time efficiency because communication was conducted all the time using discord and sharing work. After making sure that the database is working with the team's models, the first task was to create a sign-up page. They did that using html plus CSS and making sure that the references are passed correctly. It took them a while to figure this step but in two days they were done. The implementation of the login option was easy because it used the same model. After this was done, Reed made sure that authentication occurs and created a way to check for it every time something needs to be done. YanMan "Lilli" Huang and Reed created a perfect model for Blog option, where user relates to the current blog, the current blog with the current post, the current post with the comment and like/dislike. They used the model discussed with our client of how the connection between classes must be done inside the database. This was the breakthrough of our problem and the rest was tedious but doable. Gabriel created the profile where he connected user's email with their biography, name and blogs posted. Alvin and Milush were responsible for

making the Profile page working correctly and make it according to the idea of how the website should work. The last week Milush was responsible to make sure that everything connected with looks, looks presentable by upgrading Posts, Blog Page, Add Comment, Add Posts page with the correct html/CSS documentation. He shared his ideas with Alvin and after they fixed the issues, the code was pushed by Alvin. As for the middleware Reed and YanMan "Lilli" Huang made sure that code looks presentable, by compressing it and deleting the repeating code or not used one. Milush was responsible for creating the final report. To ensure better results we split the team into little teams and communicated at least once per day to see what every team did and what needs to be done.

4.Data Model



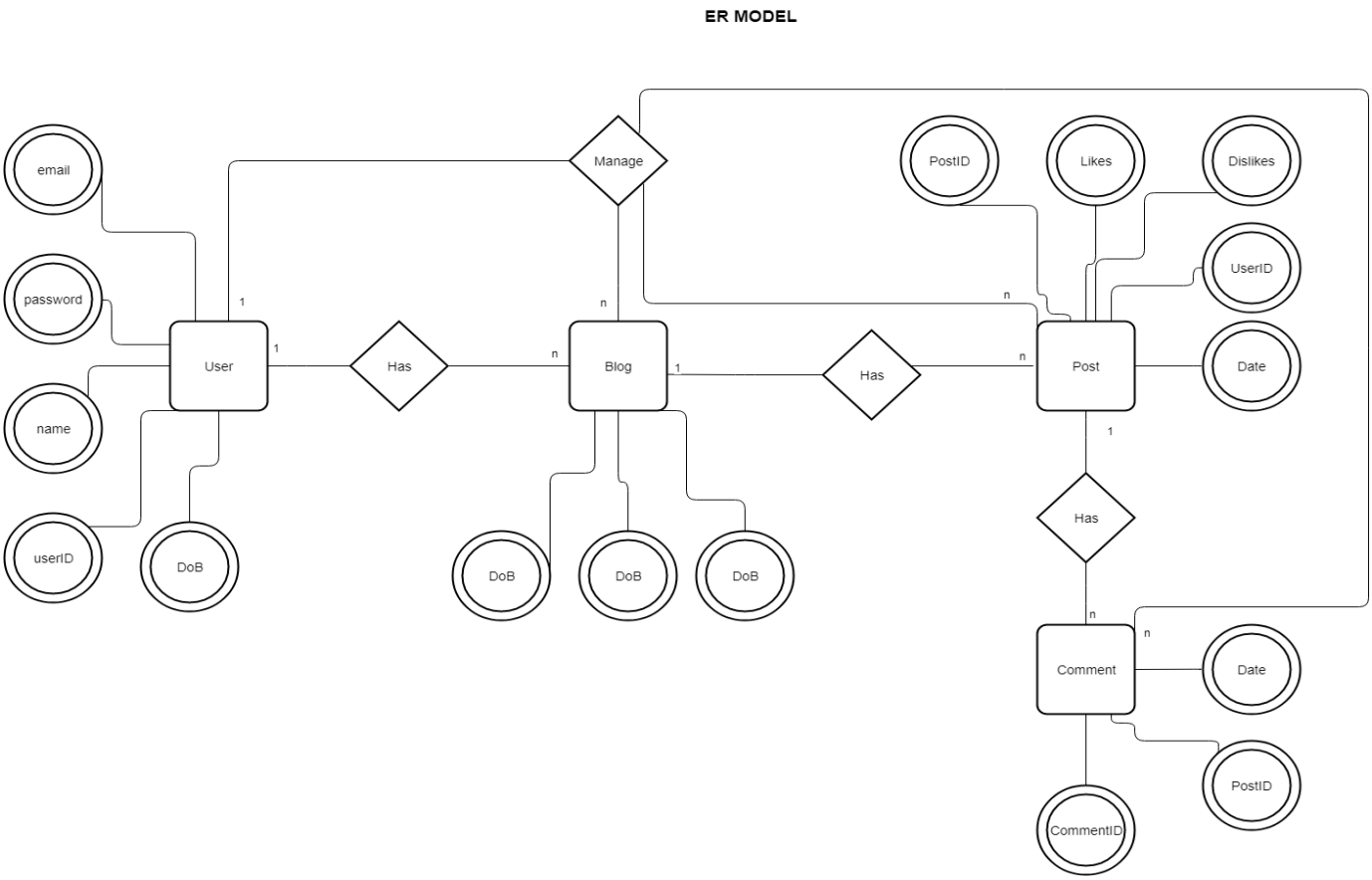
The first diagram Represents a Use Case Diagram or how the User is connected to service. In our case the User has access to Login, Sign Up, view Blog, view Posts, comment, like, dislike, sign out. The database stores everything after the user is logged in and later on pushes the information so the user can be authenticated.



Relational Model breaks down how our data model works. Every user has many blogs. Every Blog has many Posts. Every Post has many comments and like/dislike.In order to make sure that there is no cross data collection as seen from table there is a middle connection between connecting two classes. As an example, we can use the User and Blog. In order to see which user has which Blog we are using a middle connection called UserBlog, where it is connecting the UserID and the BlogID. This we will have a proper connection and never have issues.

The Entity Relation Table shows how classes are related and their attributes. As an example, we will take entity User. It has Exists, Password, Name, UserID, DoB as attributes and it has relation with Blog, Post and Comment. Circles represents attributes of an Entity; Diamond

is the relation set and rectangle is the Entity set. Numbers represents sets (ex. 1 is one and n is many-> Entity User (1) has many Blogs (n)).



5. Development Summary

Lines of Code	**1595**
Comment Lines	**157**

Blank Lines (White space)	**153**
Total Lines of Source File	**1905**

6. Design and Architecture

As a team, Oatmeal and Raisin Web Development group decided to use Oatmeal and Raisin cookies, Oatmeal or Raisins as theme subject regarding the project. An agreement was made with our client and the design process was started 2 weeks before the deadline. Milush and Alvin were responsible for the design of the website. Every decision was made after consulting the other team members and only after an agreement was made.

For the Login and Sign up Page the team decided to use a beautiful background picture of oatmeal and raisin cookies and a glass of milk, which represents childhood memories, which most of us have after growing up. For the color, we decided to dim it a little bit so it can be more pleasant for the eye, especially after there will be other components on the top of it. The focus of the website will be at the glass of milk with the cookie on top of it. Furthermore, to express better the theme, there are several oatmeal and raisin cookies on the left side. This will help the user to blend in our world and feel the culture we are trying to create.

The team were going to use the same background for the whole website but after YanMan "Lilli" Huang recommendation of different backgrounds with the same theme, we proceeded with the execution of the idea and the result was beautiful. A decision was made to stick with three different background pictures – one for Sign Up/Login, a completely different one for the Profile part and a third one for Blogs/Comments/Posts.

For the Profile background the decided to use an amazing picture that represents the beauty behind the golden color of wheat. The focus is on a light brown spoon filled with oat. The spoon lays on a tablecloth and there is wheat on the right side of it, which brings the user back to nature and its beautiful colors. As for the content, we are going to use to transparent boxes filled with different sorts of information. Transparency was chosen because we wanted for the user to be able to see and feel the theme color. The background is slightly dimmed, barely noticeable, for better readability.

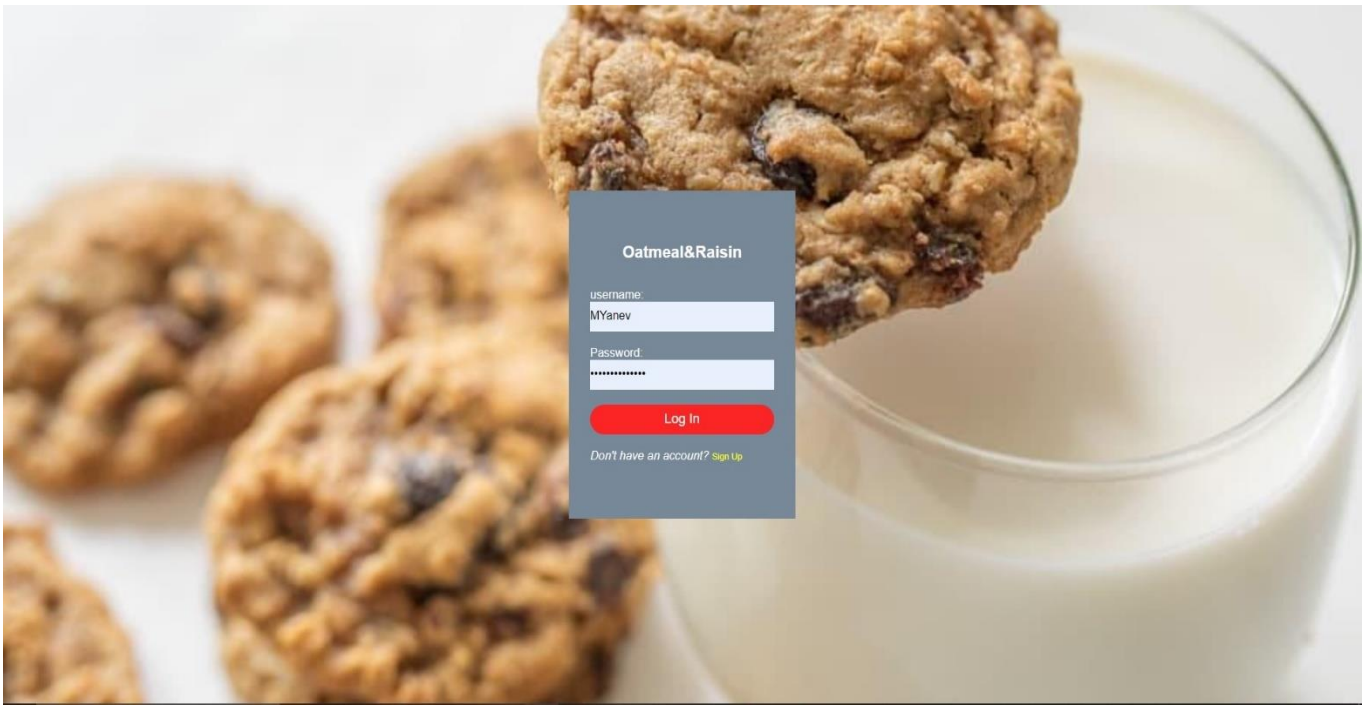
Lastly the theme for Blog, Post, Comment will be oatmeal and raisin cookies served on a black plate. The black plate is bringing out the other colors and it is stylish. It sits on the top of a red tablecloth and the user can see other deserts in the back, but they are blurred so the focus is on the oatmeal and raisin cookies. The combination between red, black and gold is astonishing. We choose a little darker component for this part of the website because we want the user to feel

where exactly he is on the website platform at every single moment, even without looking at the content.

The development team is going to provide the owner with information of how he can change the background pictures at any time.

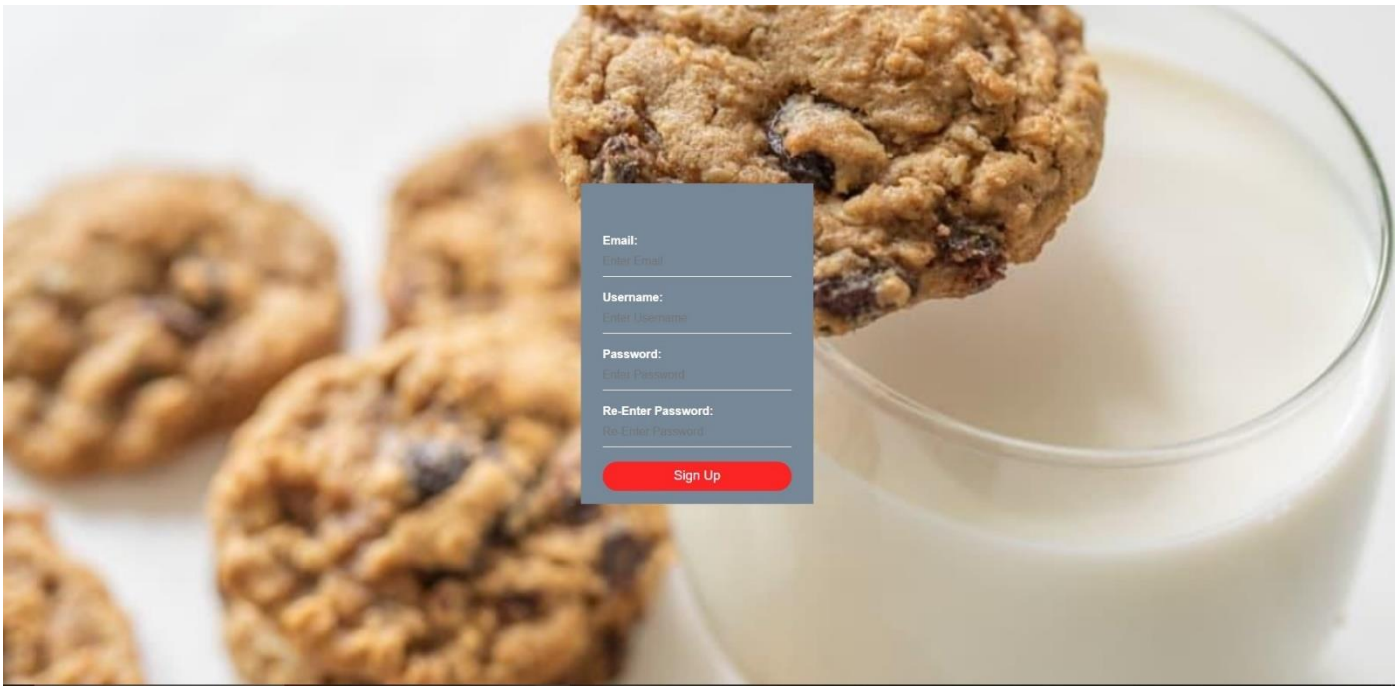
The next process is how every page is going to look with content on the top of the background. We are using several techniques in order to achieve the desired result such as navbar, boxes and passing variables and classes from database. The navbar will have only two boxes – one for the home page on the left top corner and one for sign out on the top right corner. It will be the same for every page the user is on. This is made for simplicity – if a user wants to go the home page or exit at any given moment.

The Login page has a box which is centralized. The content of the Box is username, password, Login button, Sign up Button and two strings. The first thing the user will see is a string with the name of the website. This string can be change to whatever the owner wants but we had chosen the website's name. Next step is the username and password. There are two text fields for each option, which after the click of the login button, will directly correspond with the database and an authentication process will be conducted in the background. If a user had entered wrong information, he/she will simply not be able to continue. If there is a new user trying to use the webservices, there is a sign-up option. It is displayed just after a string with a message "Don't have an account?". We used a direct passing technique to achieve the best result. When user clicks of it, it will be directly transferred to a new page with sign up information. After going over a lot of color options we conducted that the login button should be red with white text inside.



Next step is if the user clicks on Sign Page. We had created a different page that will get and store any new user's information. As stated above it will use the same background page and same positioned box. There is slightly different process when it comes to the sign-up page. Upon our agreement with the client in the Software Service Requirements Document, an email should be associated with every user. The first field that must be completed is the email box, second, we have the username and lastly two boxes for password. The reason for two boxes for the password option is that we are helping the user in case he presses something by accident. We are going to compare both passwords and if tests are successful the profile will be created. All this information is stored in the database and it is ready to be used once the Sign-Up button is pressed. As stated above the color chosen for the button will be red as well and the reason is the background and box colors. Functionality is the same as the Login Button – it will transfer you to

a page and in this case to the Login Page, so the user can re-enter his credentials in order to continue with the usage of the web service.



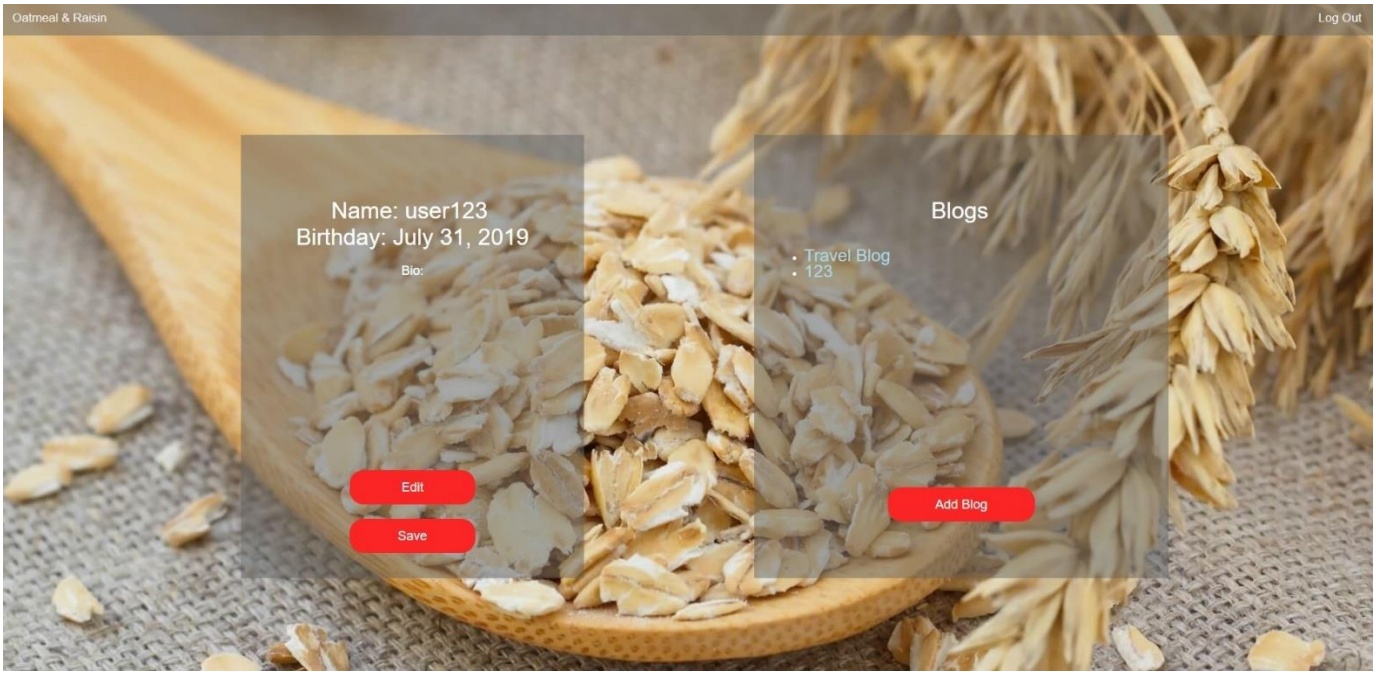
After verification has been passed the user will be transferred to the Profile page. As mentioned above we have a special background picture just for the Profile Page. The team decided that there will two main boxes displayed on the profile page. One will be with the user's information such as name, birthday and biography box. There are default values that will be stored in the name and when the user edits them, the information will be presented. The biography part is created using a box inside the box. A no color was chosen for this specific box because it looks amazing without any visible borders. The function for the name option is there because the user can choose what his name will be under the posts or comments, he/she does. If a change needs to be done the user must click on the Edit button and then Save in order to save the

current input. Everything is stored in the database and will be accessible after logout and login back in.

There is a second box inside the profile page on the right side of the user's one. This box will hold the Blogs information made by the current user. We had used a simple string text passed for the Blog message. As for where the Blogs will be displayed, we have a not visible box with scroll option, in case user has more blogs than what the blog box can fit. The box is getting information from user's id and blog's id corresponding to the current user and it is displaying the blog name. All the blogs created will be displayed using bullet points and a user will be able to click on any specific blog. The light blue color of the name represents a link to the specific blog. We did this by transferring the user to the Blog page for the current selection. This is conducted again by getting information from the database – user id plus blog id. Last there is an option of Adding a new Blog. This has been done by creating a button which transfers the user to the add blog page and he/she will be able to do the operation.

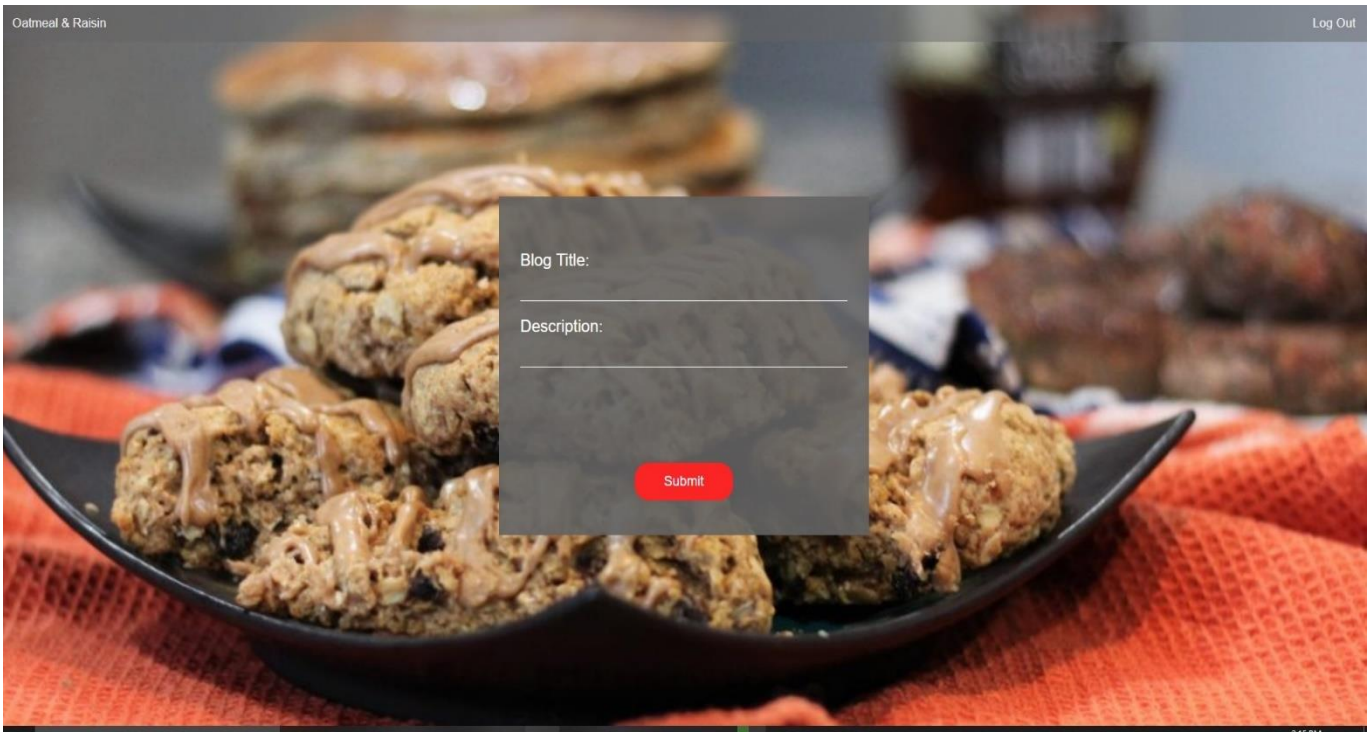
Lastly you see the implementation of the navbar mentioned before. It is a rectangular box on the top of the screen with two options attached. Top left corner is to go back to home page and top right corner is to sign out. Both buttons hold a link that when button is clicked the user will be transferred to.

For color of the boxes and the navbar, the team had chosen a transparent color because we want the user to be able to see the background picture. This is made with the idea of blending in with the theme and feel like you are apart of the web service.



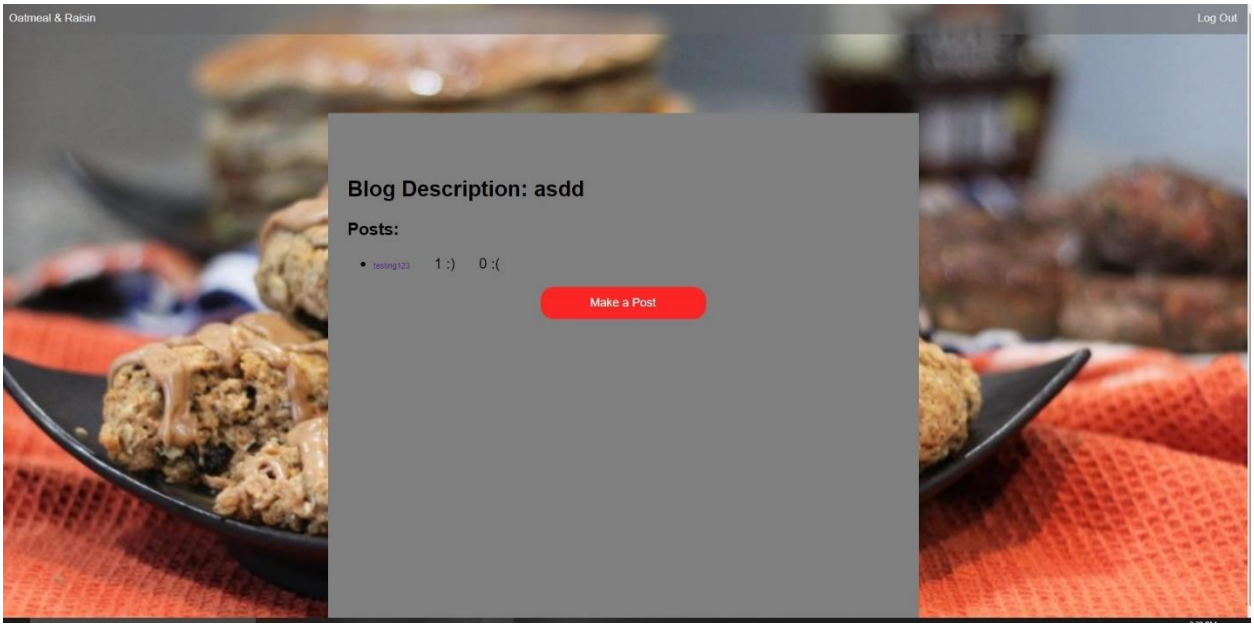
If Add Blog button is clicked the user will be transferred to the page of addition of a new blog. As mentioned above this page will have a darker and different background image. There will be a centralized square box with two options inside. As before we are using white font for all the input and darker box background. We had chosen to make the transparency barely visible because of the darker background. Upon several tests we concluded that if we use same transparency as before, there is an interference with visibility of the text and aesthetically doesn't look good. The team had chosen to underline the entered text. It was done after several options and this is the best visible way according to us. The box content has blog title and description field. Both will pass information to the database using two parameters from our models' class, stored in the database – blog name and blog description. When submit is clicked the blog will be saved in the database and it can be accessed momentarily. As before we are using the same

navigation bar on the top. The only difference is the color, we had chosen to use the same darker transparency for better visibility.



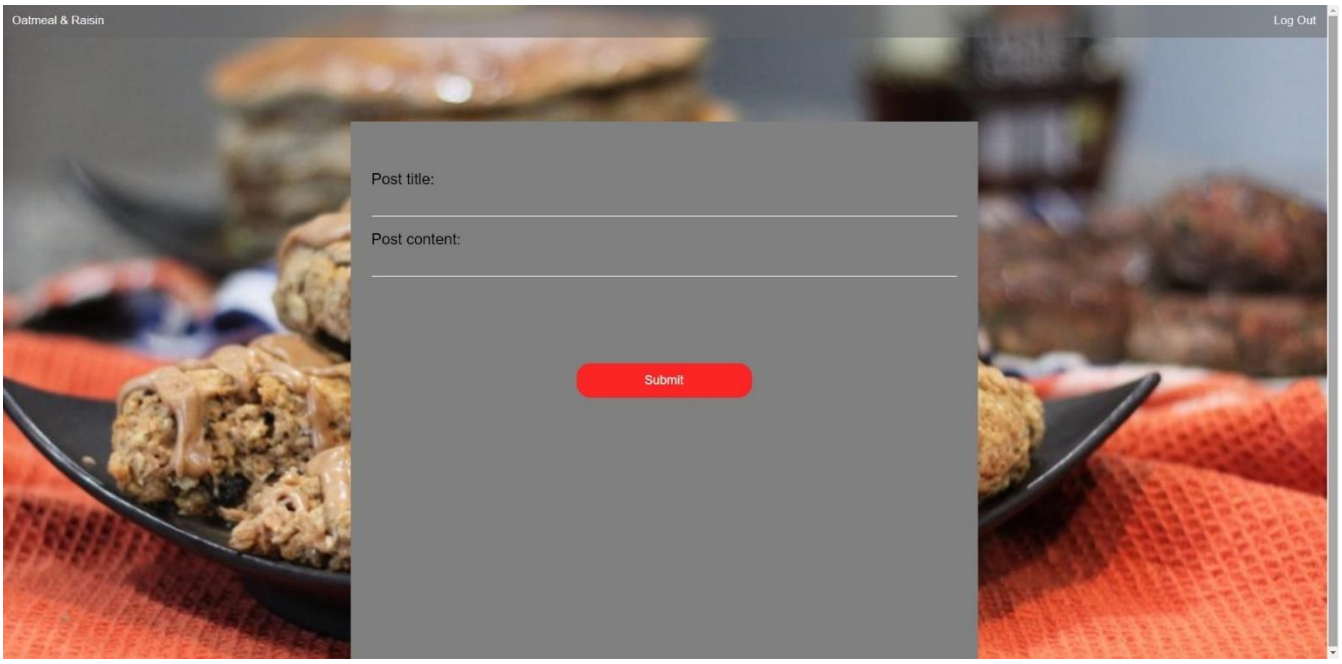
If clicked on the blue link from the Blog's box, the user will be transferred to a new screen when he/she will be able to see the current information about the blog. As stated, before we are going to use the same background picture as the add Blog one, with the same transparency. The reason is better readability. As usual we have our navigational bar on the top, with same functionalities – home page and sing out option. On this page we decided to use a taller box size, because in the long run the user will have a lot of blogs. The box has an inside scroll bar box with an absolute visibility and a scroll bar option when it gets full. As stated, before the full visibility is chosen because of a better aesthetic. On the top of the main blog box

we had passed a string “Blog description:” and right after that the description will be displayed by getting the necessary information from the database. Next step is the passed string “Post:”. Under posts we have the mentioned inside box. The content of the box will be a bullet menu, which has post name and number of likes (smiley face representation) and dislike (sad face representation.). The post name is blue because it is a link to the current post, where users can see all the different information regarding this post and can like, dislike, comment, etc. Lastly, we have the Make Post button, which if pressed the user will be transferred to a page to create a new post.



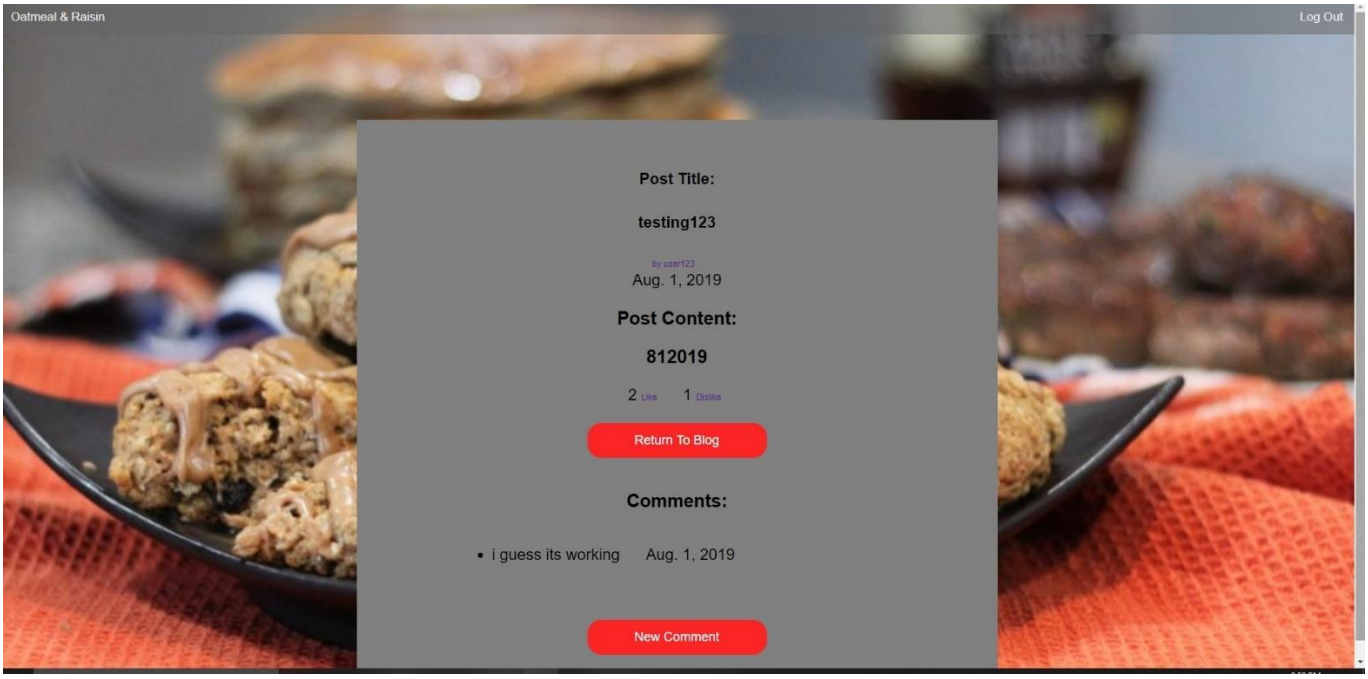
If “Make a Post” is pressed the user will be transferred to a page to create a new desired post. We are using the same template as the make blog one. The box content has post title and description field. Both will pass information to the database using two parameters from our models’ class, stored in the database – post name and post description. When submit is clicked the blog will be saved in the database and it can be accessed momentarily. The user is

momentarily transferred to the blog description page and he/she can see the new post. This is proof that the database is executing properly. As before we are using the same navigation bar on the top. The only difference is the color, we had chosen to use the same darker transparency for better visibility.



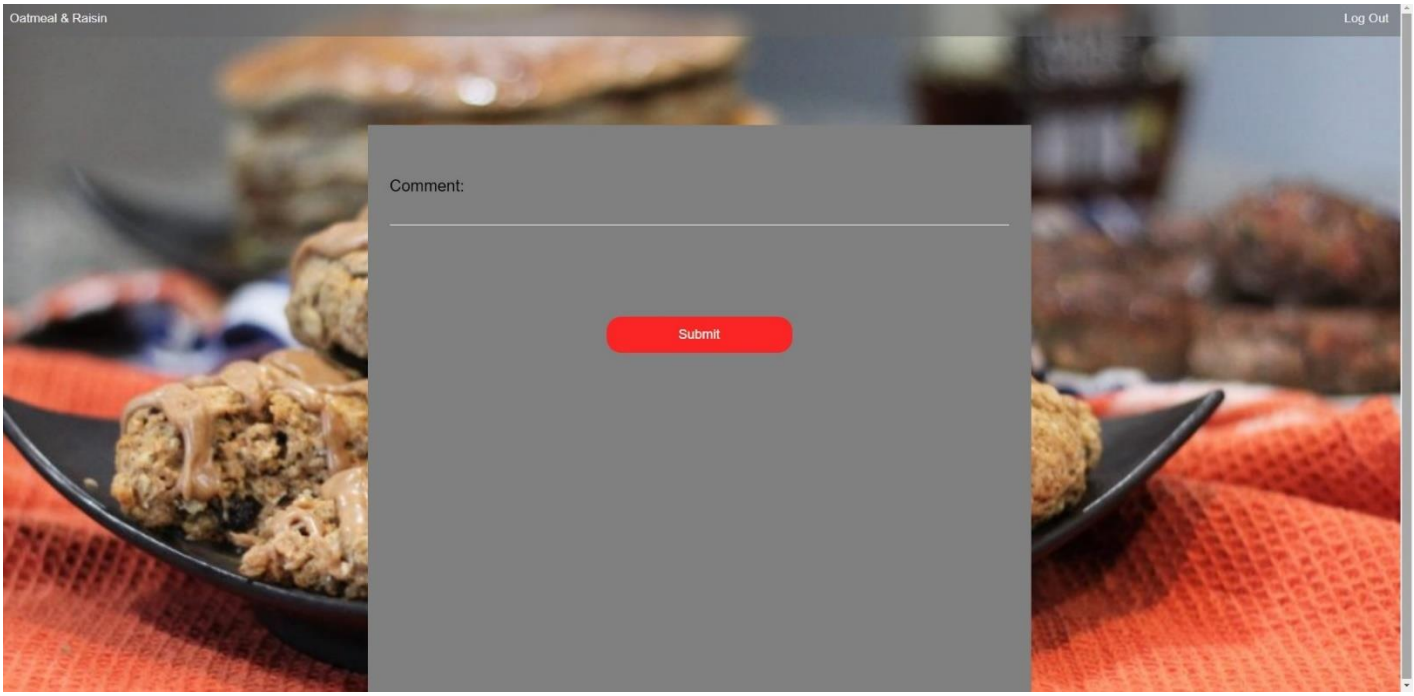
If the blue link for the post content is clicked the user will be transferred to a new page where she/he can see all the necessary information about it. As before we are using the same template with one box inside. This is the page where almost every functionality is shown in one place. On the top of the box we see a passed string “Post Title:”, followed by passing the post title from the database. Next we see the user posted the post and it is colored blue, which is an indication that it is a link. If pressed on the user, the current user will be transferred to the user’s profile page. Continuing with the date the post had been posted. The date is stored in database upon posting and it is simply called here. Then we have a string “Post Content:” where the

content is passed from the data base. Furthermore, we see the like/dislike option and numbers that indicates the number it was liked/disliked. It is live interactable option and after every click the user will see count changing accordingly. We had created a button Return Blog, for the user to go back to blog when done. The next step is to display the comments. There is a string "Comments:" passed and under it we had implemented the same bullet style scroll box as before. Inside of the box we can see the current comment and the date it was created. This information comes directly from the database. If there is a lot of comments, a scroll bar option will appear in order to display all of them. Lastly, we have button Make Comment, which will transfer the user to a new page to create comment for the current post.



The final part of the website is the Make Comment Option. When this option is chosen, the user will be redirected to a new window, where she/he can create a comment regarding a post. We used the same template as the make post one. The only difference is the content. Here

we have a passed string “Comment” and an underline text field where the input will be put. After user is done, she/he will press the Submit button and comment will be posted. It is stored in the database and will be called after the procedure is finished. When submit is pressed the user will be redirected to the post page, where she/he will see the new comment in the comment box.



7. Implementation Details

- HTML
- CSS
- Django
- JavaScript
- Python
- Bootstrap library
- MySQLs
- Proxy
- GitHub
- Domain

For us to create the webservice and its successful execution we started with installing Django and Python. We needed time to see their documentation. Next step was creating a basic internal website using a python and Django. In order for that we needed to set up a proper proxy. The database implementation was the hardest part and it was created by Reed. Next step was to create the models in python that will be used when it comes to database usage. After this the database was connected to an external service which is MariaDB and then a domain was purchased so the website will be visible from everybody. After this was done the next step was to proper use the CSS classes with HTML and Settings in Django, using Python coding. Code was pushed in GitHub on daily bases mainly by Alvin and Reed to avoid further complications. Reed had the final push – when everything was tested, and it was running.

8. Testing Data

Login	Works
Login with wrong username or password	Cannot login, reloads login page
Login with missing password	Pop up says you need to fill in the field
Sign up	Works
Sign up with an existing username	Integrity error at /signup
Sign up and passwords don't match	Cannot sign up, reloads sign up page
Viewing a profile while not logged in	-Can see bio/first name/last name -Can go to any blog on profile
Viewing a blog while not logged in	-Can see and click to go to any post -Can see the amount of likes and dislikes -Cannot like or dislike
Viewing a post of a blog while not logged in	-Can see post title -Can see the author of the post -Can see the date of the post -Can see the contents of the post -Can see likes and dislikes -Can see comments -Cannot make a comment on a post
Viewing your own profile	-Can edit bio -Can create a new blog -Can click on blog to view posts
Edit bio and submitting without changing anything	Takes you back to profile page and the data that was previously saved, remains
Viewing a blog	-Can add a post -Can view previous posts -Can see likes and dislikes
Submitting a new blog and nothing is typed	Pop up saying "Please fill out this field"
Viewing a post	-Post title -Date post was made

	-post content -comments -Can like and dislike your own post
Submitting a new post and nothing is typed	Pop up saying “Please fill out this field”
Like and dislike	-Unlimited amount of likes and dislikes -Once a like is made, page reloads with new number of likes
Comment	-works -displays comment and date posted
Submitting a new comment and nothing is typed	Pop up saying “Please fill out this field”
Return to blog button on post page	Works
Oatmeal Raisin home button	-Works on Blog page -Doesn’t work on addblog page -Doesn’t work on addpost page -Doesn’t work on post page -Doesn’t work on addcomment page
Return to blog button	Works
Logout	Works on all pages
Logged in and viewing someone else's page	-Cannot edit another user’s bio/info -Cannot add to another users list of blogs and posts
Add blog/post/comment	-only logged in users can do them -all fields where users enter text must tell user max character of that field

9. Conclusions

Overall, we are pleased with the project and the work we had put in. Most of us came into this project without any knowledge about Django, Python, Html, databases in general, servers, etc. but at the end we made it happen. The hardest thing was to learn everything in such a short period and make it works. We created a fully operating website, which is using database, which is stored in a third party's server. Despite of this there is still much work to be done and if more time was given, the results were going to be way better. The actual work was done the last 10 days simply because of trying to figure out how everything works, simply because the team was learning and doing simultaneously.

10. Suggestions for future work

We must re-do this whole process in order to absorb all the information the right way. We have the basics right now and we can create way better service if more time is presented. There are way better ways of website interactions that we need to learn. Mainly for the UI – things like displaying the content using percentage of screen and not pixel scale. Also, we need to master the button functionality because we had hard time making it work. Another thing we must learn is how to pass properly CSS files and use one main CSS and just pass the differences in the other files. For some reason some of it is working and other is not. We must master Django and Python if we need to get better results in future. Another big deal is the step process – there is so much documentation out there, we need to learn how to use it properly.

11. Appendices
11.1 A Copy of the SRS:

Software Requirements Specification

July 09, 2019

User Blog Website
prepared by
Oatmeal Raisin Web Development Group

Team Members:
Reed Schrier
Yan Huang
Milush Yanev
Alvin Heng
Gabriel Bueno

Submitted in partial fulfillment of the requirements of CS 4800 Software Engineering

Any comments in double brackets such as “<<,” “//,” or “#” are not a part of the of this Software Requirements Specifications document but are comments upon it in order to help the reader better understand the point being made about methods used in the program.

Please refer to the Software Requirement Specification Document for details on the purpose and rules for each section of it.

The work is based upon the submissions of the Summer 2019 CS 4800.

People who are working to complete the execution of the Software Requirements Specification Document are Reed Schrier (Team Captain), Yan Huang, Milush Yanev, Alvin Heng, Gabriel Bueno.

Table of Contents

1.0 Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the User Blog Website Service that Oatmeal Raisin Web Development Group will create. It will explain the purpose and features of the website, what the users will be able to do, how it is build and how it will react to external stimuli. This document is intended for both the client and the development company and it will be only approved if both sides agree on the terms and conditions stated inside of it.

1.2 Scope of Project

The website service will be a User Blog Website for any members with a valid email address. It will be designed to bring users together by creating different blogs with no restrictions about subjects except legal exclusions such as offensive themes such as inappropriate sexual subjects, racism, terrorism etc. Inappropriate language or themes used inside the website will give the administrator the option to ban the user. Other users will also be able to report another user for inappropriate content.

More specifically this service is designed to allow an admin to manage communication inside a group of bloggers and articles posted on the website. The software will facilitate a username, which will be correlated with a valid email address entered. The maximum number of users will be 1 million with the option to be extended under different conditions later. The

system will contain a relational database containing username, email, password, and number of blogs.

1.3 Glossary

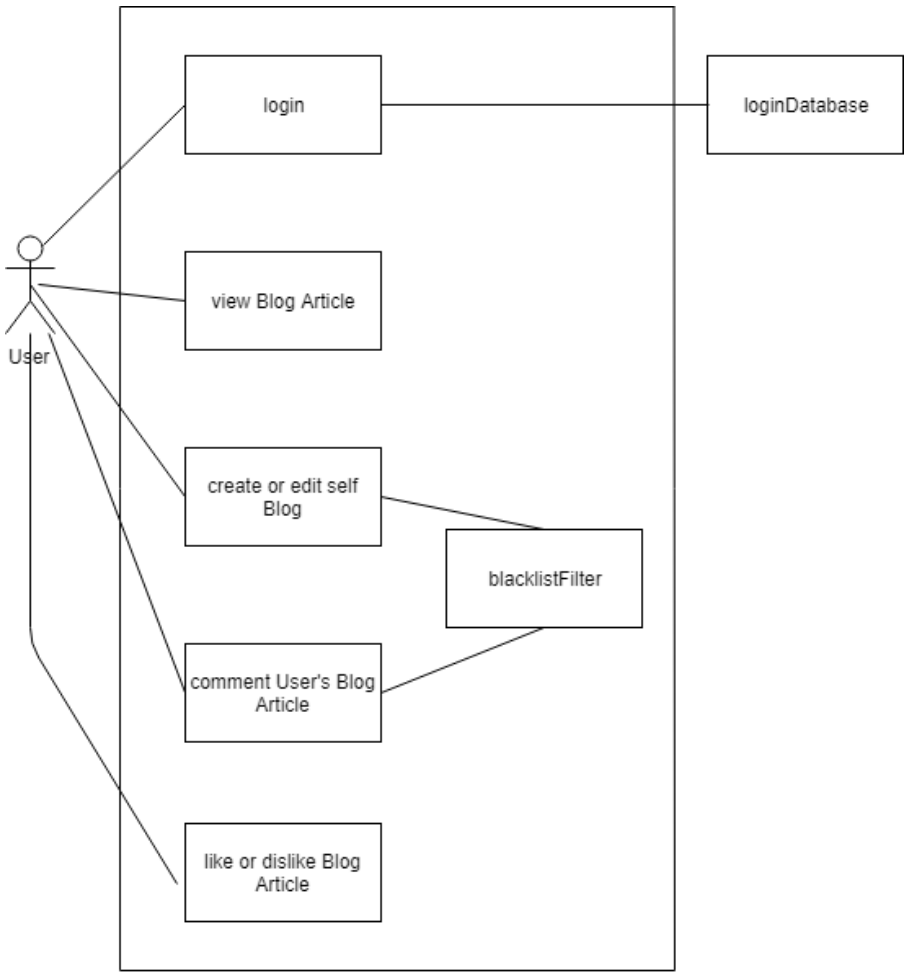
Term	Definition
Active Blog Article	An active document created by a user, which can be edited or deleted by administrator
User	A person with a valid email address. It has the option of post, edit own posts, comment, like or dislike other's posts.
Admin	A member of the support team who has full privileges to add, edit, delete posts, comments, or users.
Active Post	A written opinion by a user, containing a maximum of 10,000 characters.
Comment	A written opinion by a user, containing a maximum of 1,000 characters.
Like/Dislike Option	An option for users to express feelings about another user's post
Software Requirements Specification	A document that completely describes all the functions of a proposed system and the constraints under which it must operate.
Client	Any person who is interested in this project and is not a Developer.
Author	A member of the Development team, who is submitting this article to be reviewed by the Client. A person with whom all communication is made.

1.4 Overview of Document

The next chapter gives an overview of the functionality of the website. It uses text examples, followed by a simple use case diagrams in order to support them. This is made specially for the Client to better visualize the product and its functionalities.

2.0 Problem Description

2.1 System Environment



The Website Blog System has two main actors and one cooperating systems. The user and Admin have to login for the system to check credentials. In order to do so, the information will be checked with the cooperating system, in our case a LoginDatabase. The LoginDatabase will have information about user, email, password, isAdmin and banned user. The banned user will not be able to use the website services anymore. When the regular user verifies its credentials, there will be an option to view active blogs, comment on other's blogs, like or dislike other's posts. The administrator will have the option to delete a post or comment, ban or delete a user. If the admin chooses to ban the user, he/she will no longer be able to use the

website services. All the necessary information will be stored in a database, operated by a third company. The client will be able to decide what to do with the database information.

2.2 Functional Requirements Stated by Client

The client wants to create a website service for minimum of users that will be able to create, edit, like, dislike a blog Article. In order to do so the user must have a valid email address, create a username and password. Any comments made by the user cannot be longer than a thousand characters. There is also no password restriction besides needing to have a minimum length of six characters.

3.0 Approach

In order to satisfy the client's requirements, Oatmeal Raisin Web Development Group will have to use Maria database services. The reason for this is that it is more reliable, cheaper and easier to maintain. In the original contract, the company will create a web service that will hold up to one million people with the option of expanding that only after future negotiations. For the frontend, the company will use JavaScript, Django for middleware, and MariaDB for backend. The maximum length needed for competition of the project will be four weeks and

there will be weekly progress update. In order to satisfy the client's requirements, the company will need to have a personal access to the client at any time for regular business hours.

4.0 Requirements

4.1 Functional Requirements

Title: Account/Login

Desc: All users must have an account in order to create blogs, post a comment, and like or dislike posts. All accounts are created using an email and a password with a minimum amount of characters. Emails must be verified for the account to be usable. Users without an account can only view public posts and blogs.

Title: Profile Page

Desc: Each user with account will have a profile of their own where it shows their personal information, the number of posts and comments that the user has posted. The users will have the option to hide their personal information from their profile. The user will be able to customize their profile theme color.

Title: Blogs

Desc: Users can create multiple blogs that must have at least one post. Blogs will be shown and listed in the user's profile page.

Title: Posts

Desc: Users can post text, embedded images and/or videos with the ability to format their text with different colors and styles with a maximum of 10,000 characters. Users can also be able to edit their posts after posting. All public posts are visible to anyone including user without an account. All posts must be a part of a blog they have created.

Title: Comments

Desc: Users with an account will be able to comment on posts with a maximum of 1,000 characters without being able to format the text. Users without an account will only be able to view comments without being able to post any comments on the posts.

Title: Like

Desc: Users with an account will be able to like other users' posts.

Title: Dislike

Desc: Users with an account will be able to dislike other users' posts.

Title: Back Button

Desc: Users with an account will be able to go back to the previous selection.

Title: Change Profile Picture

Desc: Users with an account will be able to change their profile picture.

Title: Change Theme Color

Desc: Users with an account will be able to change theme color.

4.2 Non-Functional Requirements

- Blog Article Count
- Database Storage
- Accessibility
- Backup
- Documentation
- Privacy
- Modifiability
- Capacity of Users
- Security of User Database
- Platform Compatibility

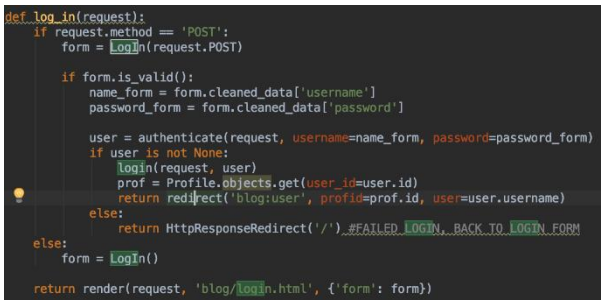
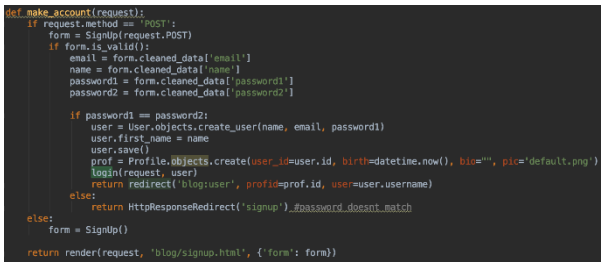
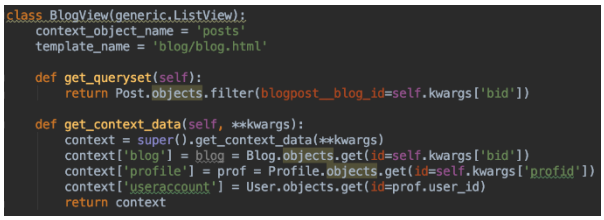
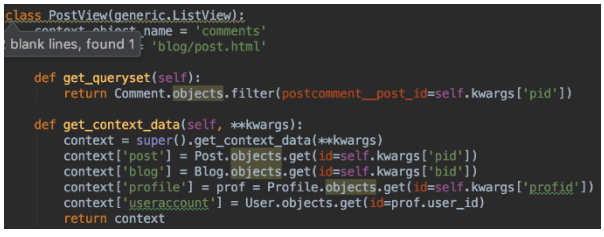
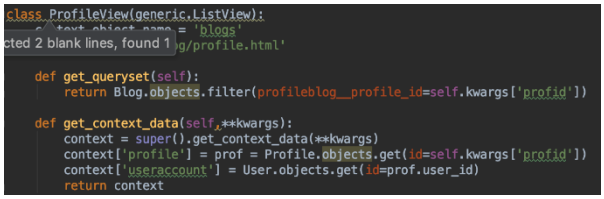
5.0 Contract Agreement

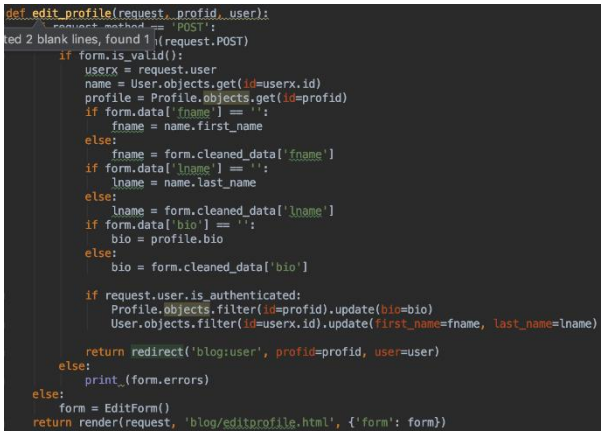
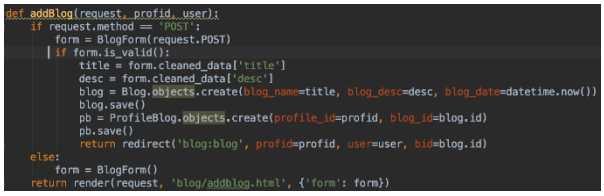
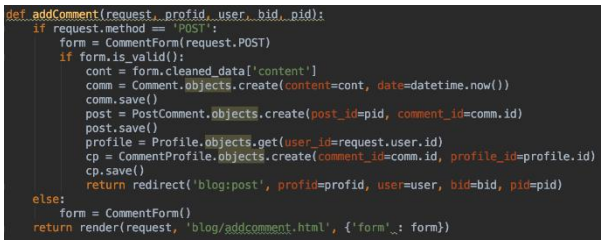
5.1 Agreement Between Parties

_____	_____
Company Manager Signature	Client's Signature
_____	_____
Company Acting Manager's Name	Client's Last, First Name
_____	_____
Effective Date	Effective Date

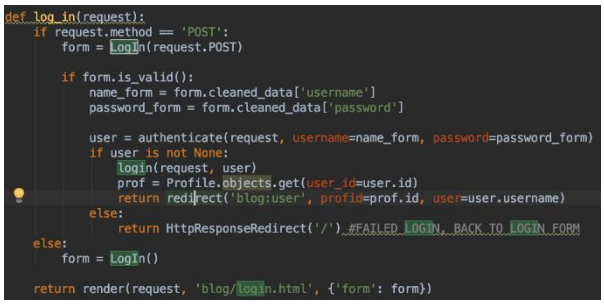
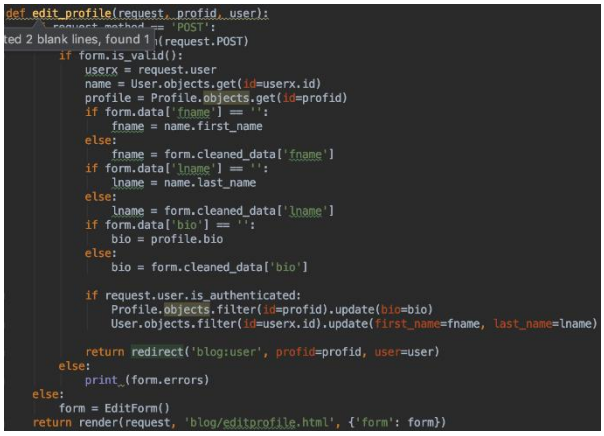
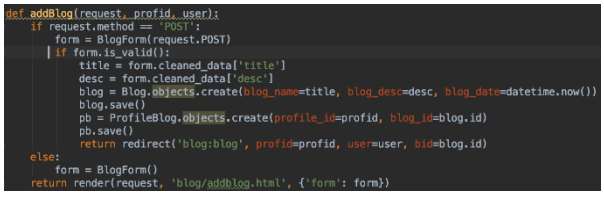
11.2 Testing Data Graphs and Tables

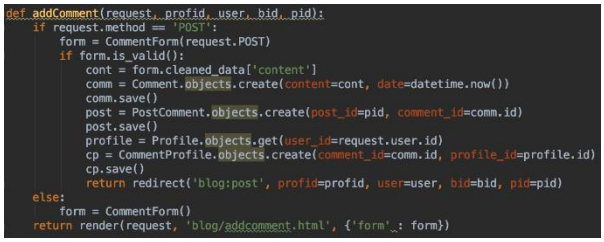
Yes table:

	Login: Yes
	Signup: Yes
	Blog: Yes
	Post: Yes
	Profile: Yes

	Logout: Yes
	Edit profile: Yes
	Add blog: Yes
	Add post: Yes
	Add comment: Yes
	Like and Dislike: Yes

No table

	Login with wrong password: No Login with missing password: No
	Signup with existing username: No Signup with passwords not matching: No
	Edit profile while logged out: No
	Add Blog while logged out: No
	Add Post while logged out: No

	Add comment while logged out: No
	Like and dislike while logged out: No