Cosaic

Team #38

Project Final Report

By
Ahmed Darkazanli
Albert Ong
Anh Phan (Chloe)

Table of Contents

Project Requirements **3**
- Project Description 3
- System Environment 3
  - Hardware and Software used 4
  - RDBMS 4
  - Application Languages 4
- Functional Requirements 4
- Non-functional Issues 6
  - Graphic User Interface (GUI) 6
  - Security 12
  - Access Control 12

Project Design **13**
- Schemas 14
- Non-Trivial Functional Dependencies 14
- Table-Entities 15
- Table-Relationships 18

Implementation **22**
- Setup procedure (Linux) 22
- Setup procedure (MacOS) 23

Project Conclusion **24**
- Statements 24
- Improvements 25

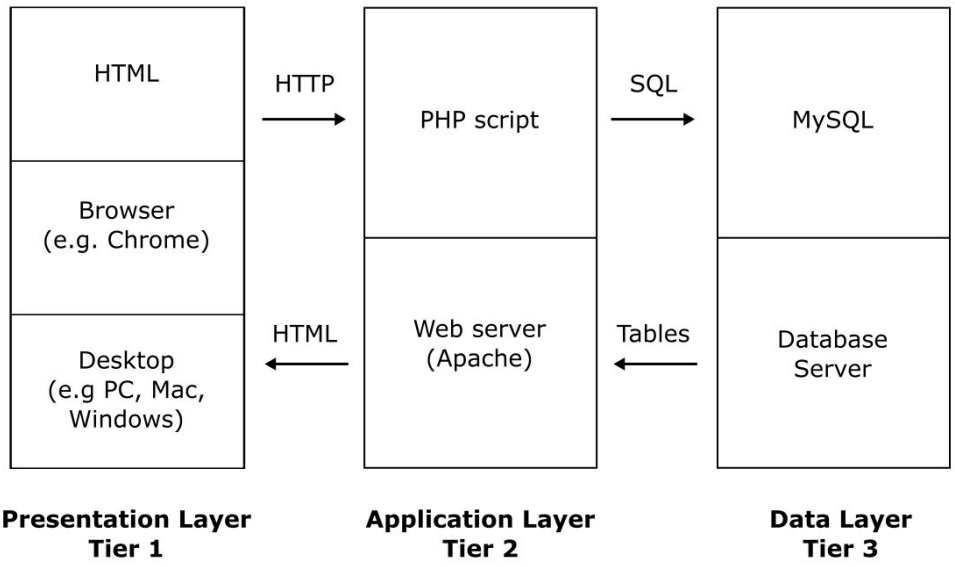
Project Requirements

Project Description

Cosaic is a social networking application that allows users to share their daily lives through online image posts. When users login to Cosaic, they will be able to view their previous posts on their own profile page as well as update their status, post pictures, and add captions and hashtags. Additionally, Cosaic will allow users to connect with other users by searching for their profile name and viewing their posts. Cosaic is designed to be a fun, casual platform that allows users to share their story with the world and view the stories of others.

The stakeholders of this project will include the members of the project team (Ahmed Darkazanli, Albert Ong, and Chloe Phan), project manager (Professor Mike Wu), and every user who loves to share their interests and activities and regularly follow their favorite people. The individuals who most enjoy interacting through social media will be the end-users that our application is targeted at. Therefore, our team must ensure that our software is functional, secure, and bug-free in order to provide the best user experience and maintain a positive image of the people working on the project.

System Environment



Hardware and Software used

- Apache web server
- Mac/PC

RDBMS

- MySql

Application Languages

- HTML/CSS
- PHP,
- SQL

Functional Requirements

- Describe users:
    - End users will be people who are interested in using social networking services to share photos and captions that describe their lives.
- How users can access the system:
    - Users will need a PC or laptop to connect to the browser such as Safari, Google Chrome, or Internet Explorer. Users will be required to sign up for a new account or log in if they have already created an account previously with Cosaic.
- Describe each functionality/features, functional processes and I/O(s).
    - View profile
        - After users log in their account, they will initially be shown a list of all their posts on the home page with a shortened version of their caption (one that fits within one line). Users will be able to view their profile details, including their profile picture, name, username and bio.
    - Edit profile
        - Users can edit their profile picture and bio by clicking the “edit profile” link below their profile picture, and either uploading a new picture or a new bio.
    - Create Posts
        - On the navbar of the Cosaic’s website, there is an option to create a post. By clicking on the “+” button, a status popup will appear on the screen and prompt users to upload a photo. Users can add their information that they want to share in the “caption” box. The message can be anything what’s going on in their lives, what’s changed, or simply how they’re

feeling or what they're thinking. Users can post their message on their homepage by simply clicking on the "Save" button at the bottom of the popup.

- Tag users in posts
  - Users can tag other users in posts by putting the "@" symbol before their username within the caption. Once posted, it will present a link to that users profile.
- Delete Posts
  - Users can delete their posts by clicking the "edit" button underneath each post and selecting "delete" on the popup that follows. After deleting the posts, users will see their home page with an updated view.
- Edit Posts
  - Users can edit the caption of any post by choosing "edit" button underneath the post. The user will then be able to enter in a new caption in an input field in the popup that follows. To successfully edit the chosen post, users will need to click on the "Save" button at the bottom at the popup to publish their changes.
- View other users posts
  - Users can search for other people who have an account on Cosaic by inputting another users' first name, last name, or username in the search bar and selecting the user in the dynamic dropdown. If the user is still not visible in the drop down, a user can hit "search" and it will show the search results in a separate page. Because ids on Cosaic are unique for each user and each account, users will immediately be brought to the searched user's home page. The searched users' homepage layout is the same style as users' homepage, but there is no option to Create Posts, Delete Posts, and or Posts.
- Comment on posts
  - Users will be able to comment on posts added by themselves or others. They must simply click on the posts, then type their response within the input field shown in the modal.
- Like posts
  - Each post will have a thumbs up button at the bottom of the posts.Each user can have maximum one like for each post. A user can just click on this button and the page will refresh showing the new changes. This will work on either the user's post or on other users' posts.

Non-functional Issues

Graphic User Interface (GUI)

The fundamental graphic user interface (GUI) of Cosaic will consist of HTML files, which contain content such as dividers, buttons, and input fields, combined with CSS files which are used to style the documents for presentation aesthetics. An integrated development environment (IDE), such as Microsoft Visual Studio Code, will be used to write, edit, and maintain these documents. Additionally, graphic design assets, including images, logos, icons, and other branding necessities, will be drawing using a vector based imaging program such as Adobe Illustrator or Inkscape. If further image editing capabilities are required to complete the project, then tools such as Adobe Photoshop and GIMP may be deployed.

The user interface of Cosiac will abide by the fundamental graphic design principles of contrast, repetition, alignment, and proximity to ensure and aesthetically pleasing user experience. The main page will double as the login page and include the Cosaic logo, input fields for a user's email and password, the login button, as well as a button that links to the signup page. The signup page will have a similar layout to the login page, with the most significant distinction being functionality rather than aesthetics. After logging in, a user will be presenting with their user profile page, which will include a search bar, logout button, profile description, post button, and image posts alongside image captions. The logout button will simply return the user to the login screen while the search bar will allow the user to search for the profiles of other users on the platform. The post button will redirect the user to another page that allows them to upload an image and write a caption to said image that will be posted to their own profile.

Because the main feature of Cosaic is the ability to share photos will the public, the majority of the screen will be occupied by the images of image posts while the minority will be occupied by the remaining features. When viewing either their own profile or ther profile of another user, users will be able to scroll down the page and view posts chronologically by earliest post first. The use of repetition will be clearly defined in this aspect of the user interface because image files will be repeated in the same manner while ordered in a vertical fashion.

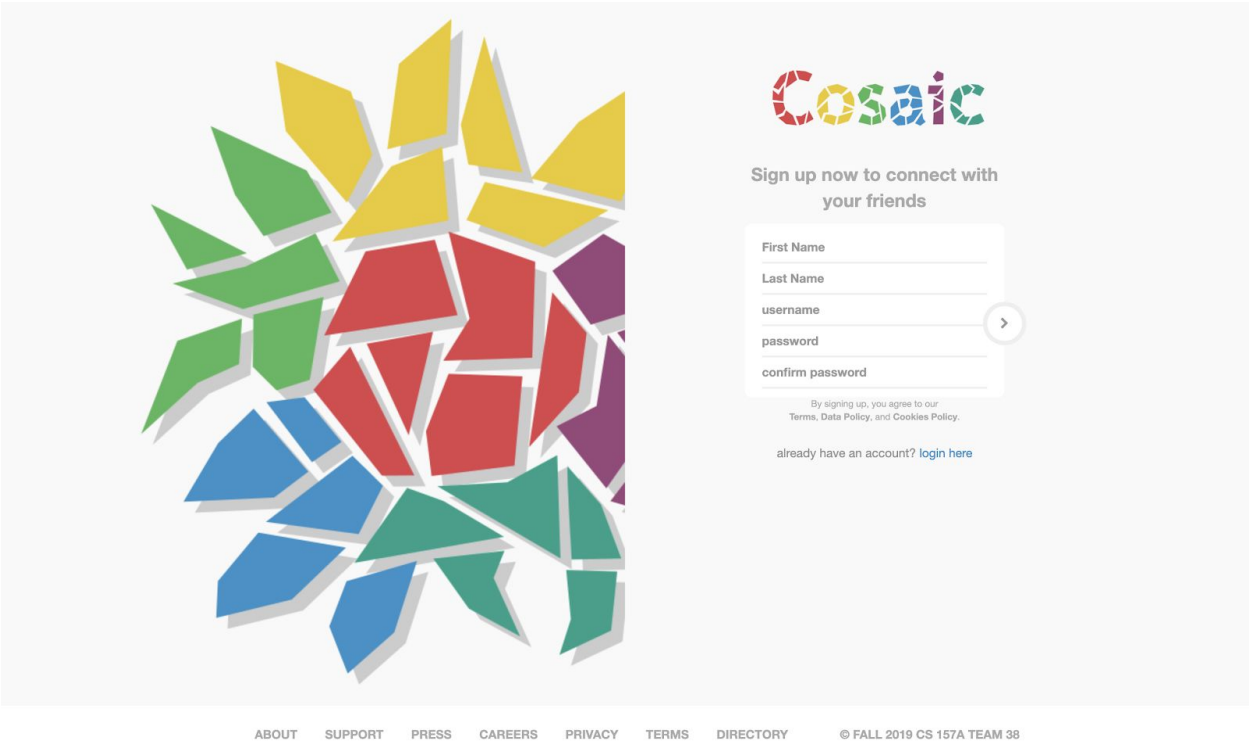


Figure 1. Signup/Login.

Chloe worked on the authentication system by creating input fields, sanitizing data, and inserting the input into MySQL, and then tracking the user's session between pages. Albert improved the UI to give it a more professional appearance.

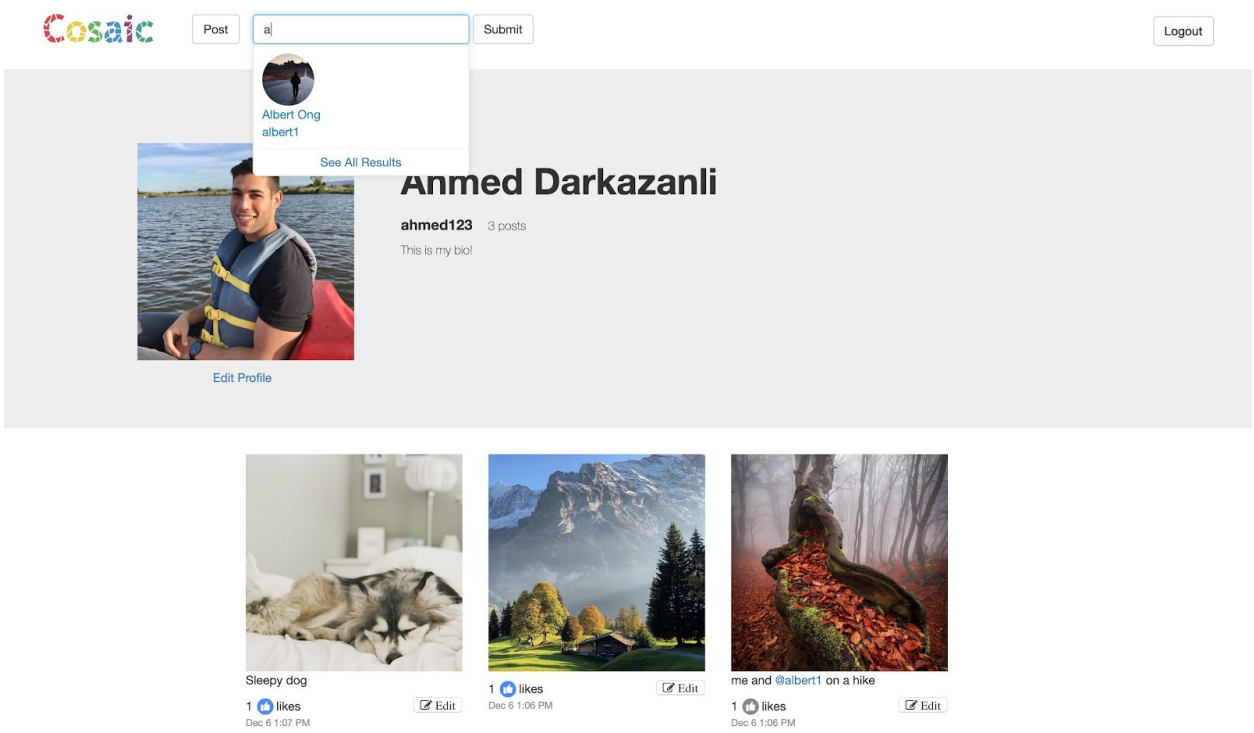


Figure 2. Home Page (with dynamic searching)

Ahmed worked on creating a fully functional navbar with a search bar that allows a user to search for other users without the whole page refreshing. Albert worked on inserting and displaying posts from the database, as well as displaying a functional like button, timestamp and a caption underneath. Ahmed improved the appearance of posts by cropping them to fit in containers, and converting any username in the caption into a hyperlink that links to the corresponding user's profile. Chloe worked on displaying the user's profile, editing the profile picture and bio, and the functionality to edit and delete posts, all of which updating the database accordingly. Albert further improved the UI.

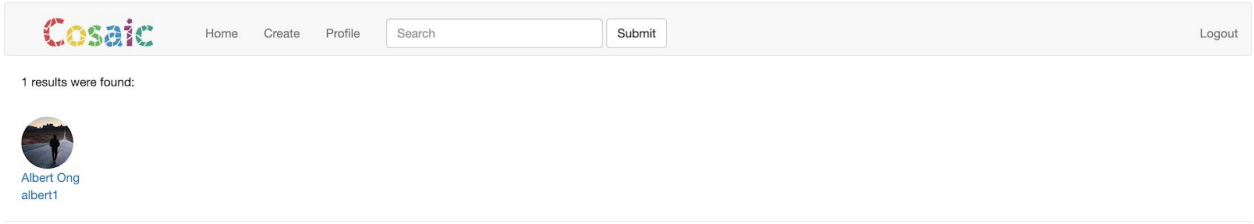


Figure 3. Search Results Page

Ahmed created the search results page which retrieves users either by their username, first name, or last name.

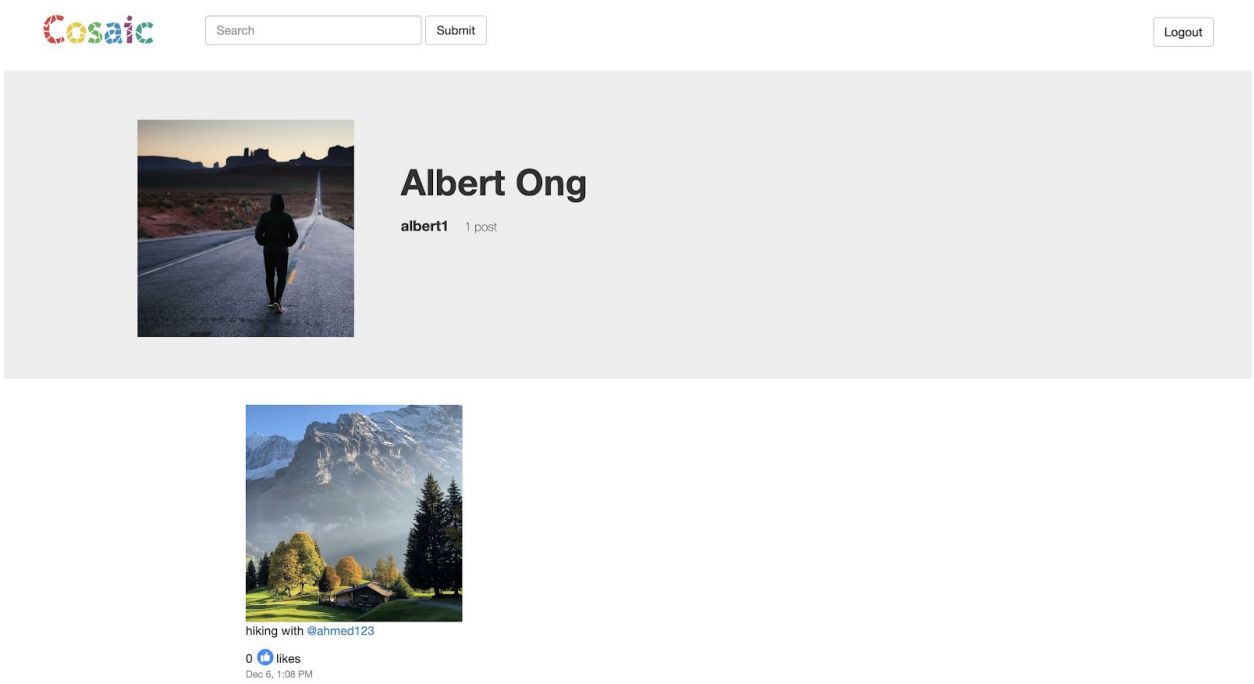


Figure 4. Viewing Other Users Profile

Ahmed coded the program to display the profile of a user with the username specified in the GET request (executed when clicking on a hyperlink). The logic for displaying the profile page is similar to that of the homepage, with the exception of updating profiles and editing/deleting posts.

Figure 5. About Page

Albert created an About page and designed it to be consistent with the appearance of the rest of the website

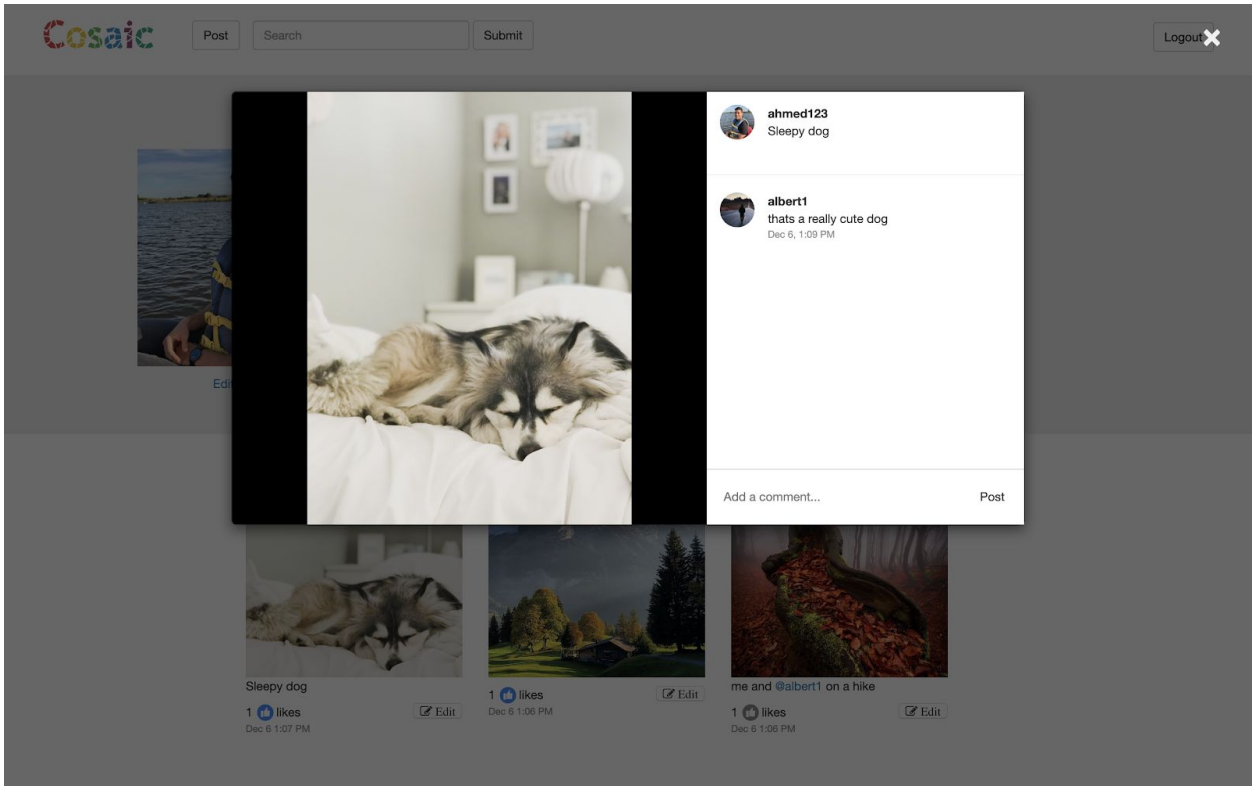


Figure 6. Adding comments

Ahmed worked on displaying a modal upon clicking posts, and being able to comment on the pictures as shown on the right. Any user can add any comment to any post, and it will update the database accordingly. The comments are appended dynamically using AJAX so that the whole page will not refresh upon form submission.

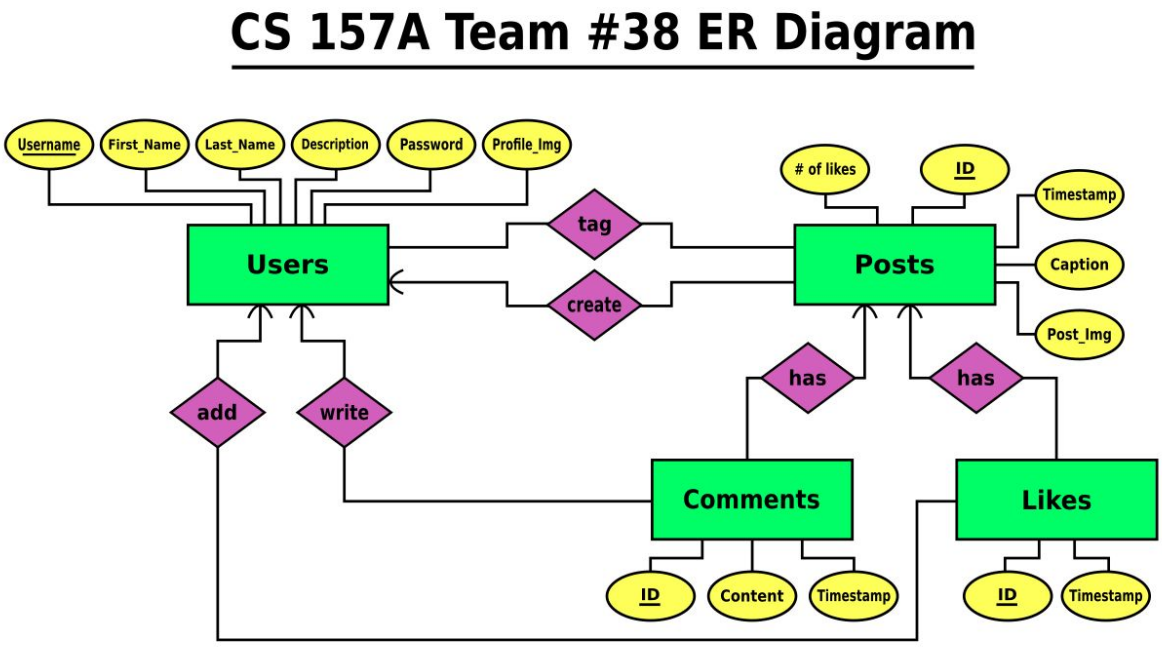
Security

Accounts will be identified by their respective username and passwords, therefore security measures should be implemented to ensure the safety of the platform. One method of security that will be implemented is SQL injection, which will involve filtered data using mysql[i]_real_escape_string() command to ensure that dangerous characters will not be passed into the SQL query. Additional, user input will be sanitized by filtering the user's data by context. Furthermore, passwords will be secured using some type of cryptographic function, such as RIPEMD, and utilize salting, the practice of adding random data before and after user input to confuse attackers. To avoid cross-site scripting (XSS), our project will filter output to the browser through the htmlentities() command. Finally, the project will avoid session fixation and hijacking by regenerating session IDs and by utilizing secure socket layers (SSL).

Access Control

A user's profile along with all images and captions posted to said profile will be visible to other Cosaic users as well as the general public. However, control over an individual user profile will be limited to the user themself. The system of access control that has been described is designed to ensure that all users can read the profiles of others, and the owner of a user account will be able to read/write the content of their own profile.

Project Design



Cosaic's Entity Relationship Diagram has four entities:

1. The User entity: has 5 attributes that define it which are <u>username</u>, first_name, last_name, description, password, profile_img.
2. The Post entity: has the following properties: <u>ID</u>, number of likes, timestamp, a caption, and a post image.
3. The Comment Entity: has three attributes which are <u>ID</u>, content, and timestamp.
4. The Like Entity: has two attributes which are <u>ID</u> and timestamp

Cosaic's Entity Relationship Diagram has six relationships:

1. Tags: the user entity has a many-to-many "tags" relationships with the post entity since we know that users can be tagged in multiple posts, and a post can tag multiple users.
2. Creates: the user entity then has a many-to-one "creates" relationship with the post entity since one user can create many posts, but a post cannot be created by many users.
3. Writes: the user entity then has a many-to-one "write" relationship with the comment entity since one user can write many comments, but one comment can only belong to one user.
4. Adds: the user entity then has a one-to-many "add" relationship with the like entity since one user can add many likes, but a like can belong to only one user.
5. Has_Like: the like entity has a many-to-one "has_like" relationship with the post entity since one post can have many likes, but a specific like can only belong to only one post.

6. Has_Comment: the comment entity also has a many-to-one “has_comment” relationship with the post entity as one post can have many comments, but a specific comment can belong to only one post.

Cosaic Database Constraints:

- All primary keys must not contain null values.
- Users table: the first_name, last_name, password fields must not be null.
- Posts table: the num_of_likes attribute must be greater than 0, timestamp and post_img fields must not be null.
- Likes table: timestamp must not be null.
- Comments table: the content and timestamp must not be null.

Schemas

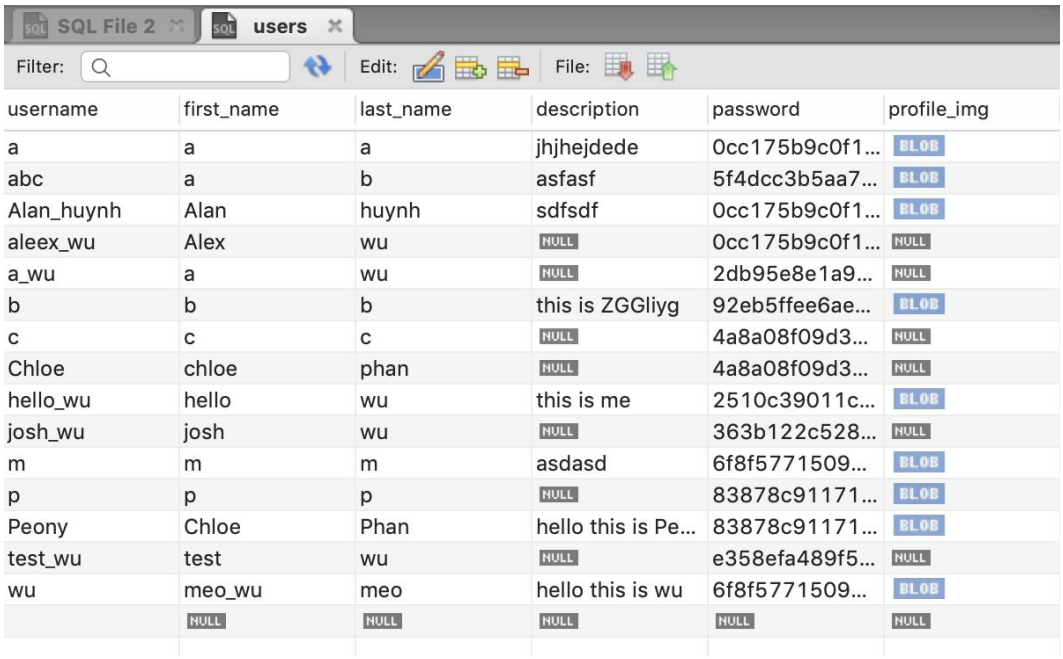
Users (<u>username</u>, first_name, last_name, description, password, profile_img)
Posts (<u>id</u>, likes, timestamp, caption, post_image)
Likes (<u>id</u>, timestamp)
Comments (<u>id</u>, content, timestamp)
Tag (<u>username</u>, <u>post_id</u>)
Creates (<u>username</u>, <u>post_id</u>)
Has_Like (<u>like_id</u>, <u>post_id</u>)
Has_Comment (<u>comment_id</u>, <u>post_id</u>)
Adds (<u>username</u>, <u>like_id</u>)
Writes (<u>username</u>, <u>comment_id</u>)

Non-Trivial Functional Dependencies

username → first_name, last_name, description, pasword, profile_img
post_id → likes, post_timestamp, caption, post_image
like_id → content, like_timestamp
comment_id → content, comment_timestamp

Table-Entities

Users



username	first_name	last_name	description	password	profile_img
a	a	a	jhjhejdede	0cc175b9c0f1...	BLOB
abc	a	b	asfasf	5f4dcc3b5aa7...	BLOB
Alan_huynh	Alan	huynh	sdfsdf	0cc175b9c0f1...	BLOB
aleex_wu	Alex	wu	NULL	0cc175b9c0f1...	NULL
a_wu	a	wu	NULL	2db95e8e1a9...	NULL
b	b	b	this is ZGGliyg	92eb5ffee6ae...	BLOB
c	c	c	NULL	4a8a08f09d3...	NULL
Chloe	chloe	phan	NULL	4a8a08f09d3...	NULL
hello_wu	hello	wu	this is me	2510c39011c...	BLOB
josh_wu	josh	wu	NULL	363b122c528...	NULL
m	m	m	asdasd	6f8f5771509...	BLOB
p	p	p	NULL	83878c91171...	BLOB
Peony	Chloe	Phan	hello this is Pe...	83878c91171...	BLOB
test_wu	test	wu	NULL	e358efa489f5...	NULL
wu	meo_wu	meo	hello this is wu	6f8f5771509...	BLOB
	NULL	NULL	NULL	NULL	NULL

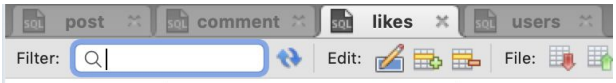
Comments

post | comment | likes | users

Filter: Edit: File:

id	content	create_at
1	“This is nice”	2019-03-01 11:57:43
2	“ How are you?”	2019-04-03 10:45:19
3	“That pizza! nice”	2019-04-07 11:26:23
4	“ Great pic!”	2019-04-10 04:25:12
5	jacket	2019-04-26 11:34:39
6	“yummy!”	2019-05-15 09:15:19
7	“have funs”	2019-05-29 20:57:34
8	cool	2019-05-30 05:23:56
9	“that food is nom nom”	2019-06-15 11:57:13
10	“Hey, is it the restaurant on Story rd?”	2019-07-10 14:34:16
11	“where is this place?”	0000-00-00 00:00:00
12	“address plzzzz”	0000-00-00 00:00:00
13	“yummy !!”	2019-09-27 10:14:35
14	“that food is nom nom”	2019-11-10 12:46:15
15	“nice jacket!”	2019-11-10 12:46:15
	NULL	NULL

Likes



id	create_at
1	2019-01-01 00:00:01
2	2019-04-03 10:45:19
3	2019-04-07 11:26:23
4	2019-04-10 04:25:12
5	2019-04-26 11:34:39
6	2019-05-15 09:15:19
7	2019-05-15 23:15:34
8	2019-05-29 20:57:34
9	2019-05-30 05:23:56
10	2019-06-15 11:57:13
11	2019-07-10 14:34:16
12	0000-00-00 00:00:00
13	0000-00-00 00:00:00
14	2019-09-27 10:14:35
15	2019-11-10 12:46:15
	NULL

Posts

id	likes	timestamp	caption	post_image
1	0	2019-11-11 2...	hahahahaha	BLOB
2	0	2019-11-11 2...	ertyuio	BLOB
3	0	2019-11-11 2...	njamd	BLOB
4	0	2019-11-11 2...	this is cool	BLOB
5	0	2019-11-11 2...	this is cool	BLOB
6	0	2019-11-11 2...	this is cool	BLOB
7	0	2019-11-11 2...	asdasd	BLOB
8	0	2019-11-12 0...	this is cool	BLOB
9	0	2019-11-12 1...	Lala	BLOB
10	0	2019-11-12 1...	lklklklk	BLOB
11	0	2019-11-12 1...	Chloe	BLOB
12	0	2019-11-12 1...	this is first pic	BLOB
13	0	2019-11-12 1...	new new new	BLOB
14	0	2019-11-12 1...	this is second...	BLOB
15	0	2019-11-12 1...	asdasd	BLOB
16	0	2019-11-12 1...	this is. niece	BLOB
17	0	2019-11-12 1...	sfsdfsdf	BLOB
18	0	2019-11-12 1...	asdsa	BLOB
20	0	2019-11-12 1...	dfgdfgdfg	BLOB
22	0	2019-11-18 2...	aasd	BLOB
23	0	2019-11-21 1...	this is a bear post	BLOB
24	0	2019-11-21 1...	sdfghjk	BLOB
25	0	2019-11-21 1...	cosaic	BLOB
26	0	2019-12-06 2...	asd	BLOB
27	0	2019-12-06 2...		BLOB
	NULL	NULL	NULL	NULL

Table-Relationships

Create

username	post...
Rosie	1
Greenface	2
Mike	3
Mickey	4
Kellywang	5
Greenface	6
Mike	7
Harrypotter	8
Haileywu	9
Harrypotter	10
Noah78	11
Johnisme	12
Sanfrankisco	13
Peony87	14
Mike	15

creates

Tag

username	id
Chloe	29
abc	2
a_wu	3
b	17
c	15
hello_wu	30
p	22
peony	13
aleex_wu	1
Chloe	2
Alan_huynh	5
test_wu	10
josh_wu	17
m	20
p	25
	NULL

Has_Comments

post_id	comment...
4	1
2	2
6	3
4	4
3	5
4	6
7	7
6	8
6	9
2	10
4	11
12	12
15	13
3	14
6	15

has_comments

Has_Likes

	LikeID	PostID
▶	1	8
	2	4
	3	13
	4	5
	5	5
	6	9
	7	27
	8	6
	9	4
	10	9
	11	31
	12	8
	13	1
	14	3
	15	7

has_like 1 ×

Adds

	username	like_id
▶	Harrypotter	1
	Rosie	2
	Greenface	3
	Mike	4
	Noah78	5
	Rosie	6
	Harrypotter	7
	Mike	8
	Sanfrankisco	9
	Rosie	10
	Johnisme	11
	Noah78	12
	Haileywu	13
	Kellywang	14
	Greenface	15

adds

Writes

username	comment...
Greenface	1
Harrypotter	2
Kellywang	3
johnisme	4
Mike	5
Mickey	6
Kellywang	7
Rosie	8
Greenface	9
Mike	10
Harrypotter	11
Alexander	12
Dogandcat	13
Dogandcat	14
Greenface	15

writes

Implementation

The database application system was implemented using a basic 3-tier architecture, where we have an Apache Web Server serving php files on the same domain as the MySQL server. To connect to a database instance, the PHP script will call mysqli_connect() along with the parameters: host name, username, password, and name of database. So long as this connection is open, our application is able to access the database and perform queries that either define or manipulate the data within it using mysqli_query().

Setup procedure (Linux)

1. Install Apache, PHP, and MySQL
   ```
   sudo apt-get install apache2
   sudo apt-get install php libapache2-mod-php
   sudo apt-get install mysql-server
   ```

2. Restart Apache once the packages are installed
   ```
   sudo systemctl restart apache2
   ```

3. Install Git
   ```
   sudo apt install git
   ```

   Check git version to verify installation
   ```
   git --version
   ```

4. Clone the repository from Github to the directory `var/www/html`
   ```
   cd /var/www/html
   git clone https://github.com/CS-157A-Team-38/CS157A-Team-38.git
   ```

5. Start apache and open cosaic in your browser
   ```
   sudo systemctl start apache2
   ```

   Enter the following address into your internet browser
   ```
   localhost/CS157A-Team-38
   ```

   See figure 1. to see what the initial page should look like

Setup procedure (MacOS)

1. Start Apache Web Server
   ```
   sudo apachectl start
   ```

2. Activate PHP

   Open httpd.conf file
   ```
   sudo nano /etc/apache2/httpd.conf
   ```

   Press **control-w** and type **php** then hit enter. Remove '#' from following line and save changes to activate php7
   ```
   #LoadModule php7_module libexec/apache2/libphp7.so
   ```

3. Restart Apache
   ```
   sudo apachectl restart
   ```

4. Install MySQL from https://dev.mysql.com/downloads/mysql

5. Install Git
   ```
   sudo apt install git
   ```

   Check git version to verify installation
   ```
   git --version
   ```

6. Clone the repository from Github to the directory `Library/WebServer/Documents`
   ```
   cd /Library/WebServer/Documents
   git clone https://github.com/CS-157A-Team-38/CS157A-Team-38.git
   ```

7. Start apache and open cosaic in your browser
   ```
   sudo apachectl start
   ```

   Enter the following address into your internet browser
   ```
   localhost/CS157A-Team-38
   ```

   See figure 1. to see what the initial page should look like

Project Conclusion

Statements

Ahmed

I learned how to debug php programs, resolve merge conflicts, and use AJAX with JQuery to create very interactive web pages. Things like displaying modals, adding new comments to posts, displaying search results in the search bar without refreshing the entire page can be achieved using tools like AJAX. I also learned about how apache works under the hood to handle POST/GET requests sent by the browser. Of course, I also learned the proper way to use Relational Database Management Systems such as MySQL to store and handle application data efficiently and consistently.

Albert

Over the course of this project, I learned how to configure Apache, PHP, and MySQL within a Linux environment and how to deploy these tools to create professional style web applications. Additionally, I learned how to utilize Git and GitHub to collaboratively manage a long-term software project and to maintain appropriate version control using Git commands. Furthermore, I was able to once again reinforce my skill with graphic design, especially when creating and displaying the graphics that would be used in the user interface. This project emphasized the importance of hard skills, such as programming languages and database management, as well as soft skills, such as teamwork and communication, which made the entire experience a valuable lesson.

Chloe

Thanks to being a member of the Cosaic's software developers team, I have learned how to set up the IOS environment using Apache. Not only that, I understand and practice MySQL to build the database through MySQL workbench and execute the query to manage my database into the project's high-level requirements. I also have more chances to practice PHP, JavaScript, and jQuery to build the backend with features such as creating/editing the profile's picture/images/descriptions, create comments for post and login/signup authentication. This project gives me more opportunities to be more creative in the design front end using HTML and CSS. I also learned how to translate ERD's schema into the database's logic for executing features' needs. Cosaic gives me more hands-on experiences on the foundation concept of using GET/POST request by the browser. Cosaic also gives me a great teamwork environment that each teammate makes a great contribution to the project's process. I learned and experienced

much about how to cooperate with teammates to test and integrate the huge amount of code through Github.

Improvements

An area of the application that could be improved in the future would be adding a dynamic dropdown menu when tagging users. Typing '@' would automatically reveal a dropdown of potential users you would like to add to your post, and adding more letters after it will narrow the selection down even more. Additionally, tagging users currently only hyperlinks to the tagged users profile in the description below an image post. An extension of the tagging feature would be display a properly formatted hyperlink of tagged users in the post description presented in a post's modal and the ability to tag users within the comments of a post.

An improvement to the like button would be the ability to display which users liked a post, rather than simply displaying the number of users who like said post. Equivalent to Instagram's functionality, such a feature would allow the user to click on the number of likes per photo and cause a modal appears with the list of users who liked it. Furthermore, liking could be extended to include comments and posts, rather than the current version of the project which only supports liking posts. Liking a comment to a like would include the same functionality as liking a post, with the number of likes being displayed below the comment along with the ability to see which users liked said comment.