Project 2

Photogram Web Application

Due: ~~April 26, 2020~~ May 17, 2020 at 11:55PM

Group Assignment

Note: Project 2 is a group project and groups should consist of at least 2 persons and no more than 4 persons.

Please ensure that you let me know by email (yannick.lynfatt@uwimona.edu.jm) who your group members are by the latest, Friday, ~~April 5~~ April 24.

Background

For this project we will create a fictional Instagram clone called Photogram that we can share photos with our friends and family.

Database Schema

- Posts
    - id
    - user_id
    - photo
    - caption
    - created_on

- Users
    - id
    - username
    - password
    - firstname
    - lastname
    - email
    - location
    - biography
    - profile_photo
    - joined_on
- Likes
    - id
    - user_id
    - post_id
- Follows
    - id
    - user_id
    - follower_id

Note: Ensure that you define the appropriate models and create your database migrations.

Key Functionality

You should be able to register for an account on our Photogram web application. Once a user has an account, they should be able to login and upload photos to their Photogram feed. Each photo, should have an image and a caption. A user should also be able to "Like" another users posts and also follow a user.

Part 1

The aim of the first part of the project is to build the API for your Photogram application. This includes creating routes (endpoints) for the user registration and login system as well as the ability to add and view your posts and follow and like. See *Table 1* and accompanying note.

The API routes (endpoints)

Table 1: API Routes (endpoints)

HTTP Method	Route	Description
POST	/api/users/register	Accepts user information and saves it to the database
POST	/api/auth/login	Accepts login credentials as username and password
GET	/api/auth/logout	Logout a user
POST	/api/users/{user_id}/posts	Used for adding posts to the users feed
GET	/api/users/{user_id}/posts	Returns a user's posts
POST	/api/users/{user_id}/follow	Create a Follow relationship between the current user and the target user.
GET	/api/posts	Return all posts for all users
POST	/api/posts/{post_id}/like	Set a like on the current Post by the logged in User

Note: More details about the API and examples of the expected JSON responses can be viewed at: https://photogram.docs.apiary.io.

Test to ensure that your API works by using either the Postman REST Client (http://getpostman.com/) or the Curl command line tool to make requests to your API routes (endpoints).

Part 2

You are required to use VueJS to build the front end of your web application that will interact with the API you built in Part 1. You should also ensure the following is in place:

1. You should use the VueRouter library to create routing for your frontend and implement some Vue components to represent the different pages. *See Table 2 below for a list of routes.*
2. A User should be able to Register for an account and Login to the website. *See Figure 2 and 3.*
3. You should generate and send the appropriate Authorization header with each request to your API (except the login and register API routes) using a JWT e.g. **`Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ`**
4. When a user successfully logs in they should see posts from all users. They can also access this same view by clicking the 'Explore' link in the menu. *See Figure 4.*
5. When on the 'Explore' page, a user should be able to view a specific users profile by clicking on their username in the post.
6. When viewing another users' profile (*See Figure 5*), you should be able to 'Follow' that user by clicking the 'Follow' button on the use

profile. This should update the follower count on the page and change the text on the button to say 'Following'. *See Figure 6.*

7. The user can then view their own profile by clicking 'My Profile' from the menu. *See Figure 5.*
8. The user should be able to choose to add a new post to their own Photogram feed at which point they will upload an image and give a description for the photo they would like to add to their feed. *See Figure 7.*
9. You should display a success message if the user successfully adds a new post or failure otherwise.

Frontend Routes

Table 2: Frontend Routes

Route	Description
`/`	`Display the homepage of the web application.`
`/register`	`Accepts user information and saves it to the database`
`/login`	`Accepts login credentials as username and password`
`/logout`	`Logout a user`
`/explore`	`View/Explore all posts by all users`
`/users/{user_id}`	`View user profile info as well as all Posts by that user`
`/posts/new`	`Allow the user to add a new post`

Deploy the application to Heroku

The finished application should be deployed to Heroku. If you haven't already done so, ensure that you sign up for an account on the Heroku website (https://heroku.com) and then do the following.

Note: *You will also need the Heroku CLI. This should already be installed on Cloud9. If you are instead working on your local machine, then you must install the CLI tool. See instructions at* https://devcenter.heroku.com/articles/heroku-cli *.*

```
heroku login
heroku apps:create
git push heroku master
```

You will also need to ensure that you have a Database setup on Heroku for your application. To create a provision the Postgres database add-on and create a PostgreSQL database on Heroku, follow the instructions at the following link:

https://devcenter.heroku.com/articles/heroku-postgresql#provisioning-the-add-on

Note: You will be using the **hobby-dev** plan when creating your database.

```
heroku addons:create heroku-postgresql:hobby-dev
heroku config -s
```

You should then see something like the following:

```
postgres://
yecsapnzlttgcb:8860d3512549e3ebba04aa68ede74496e30041ff
458ea1d46d7c4ed7f5402af0@ec2-54-221-244-196.compute-1.a
mazonaws.com:5432/d9d6gbbcjoc71g
```

This is the URI that you will use in your **`SQLALCHEMY_DATABASE_URI`** config option in your Flask Application. Ensure that you change driver (at the start of the URI) from **`postgres`** to **`postgresql`** for Flask SQLAlchemy.

You will also need to ensure you make a modification to your Heroku **`Procfile`**. This will allow you to run the migration you created earlier (an any other future migrations) to create/update your Heroku Database. The updated **`Procfile`** will look similar to this:

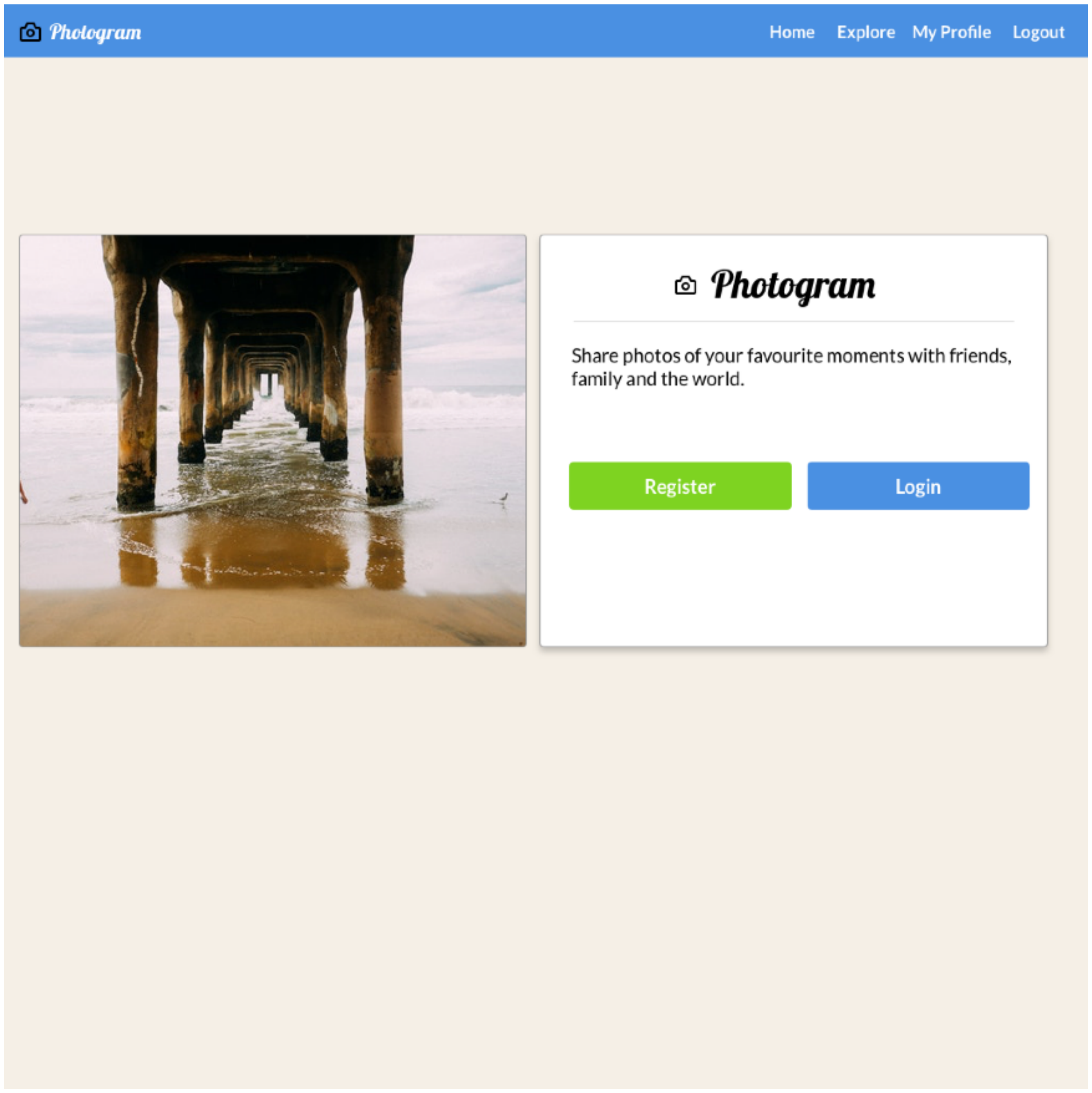
```
release: python flask-migrate.py db upgrade --directory migrations
web: gunicorn -w 4 -b "0.0.0.0:$PORT" app:app
```

Submission

Submit your code via the "Project 2 Submission" link on OurVLE. You should submit the following links:

1. Your Github repository URL for your Flask app e.g. https://github.com/{yourusername}/info3180-project2
2. Your URL for your Heroku app e.g. https://{yourappname}.herokuapp.com

Appendix



Share photos of your favourite moments with friends, family and the world.

FIGURE 1. HOME PAGE

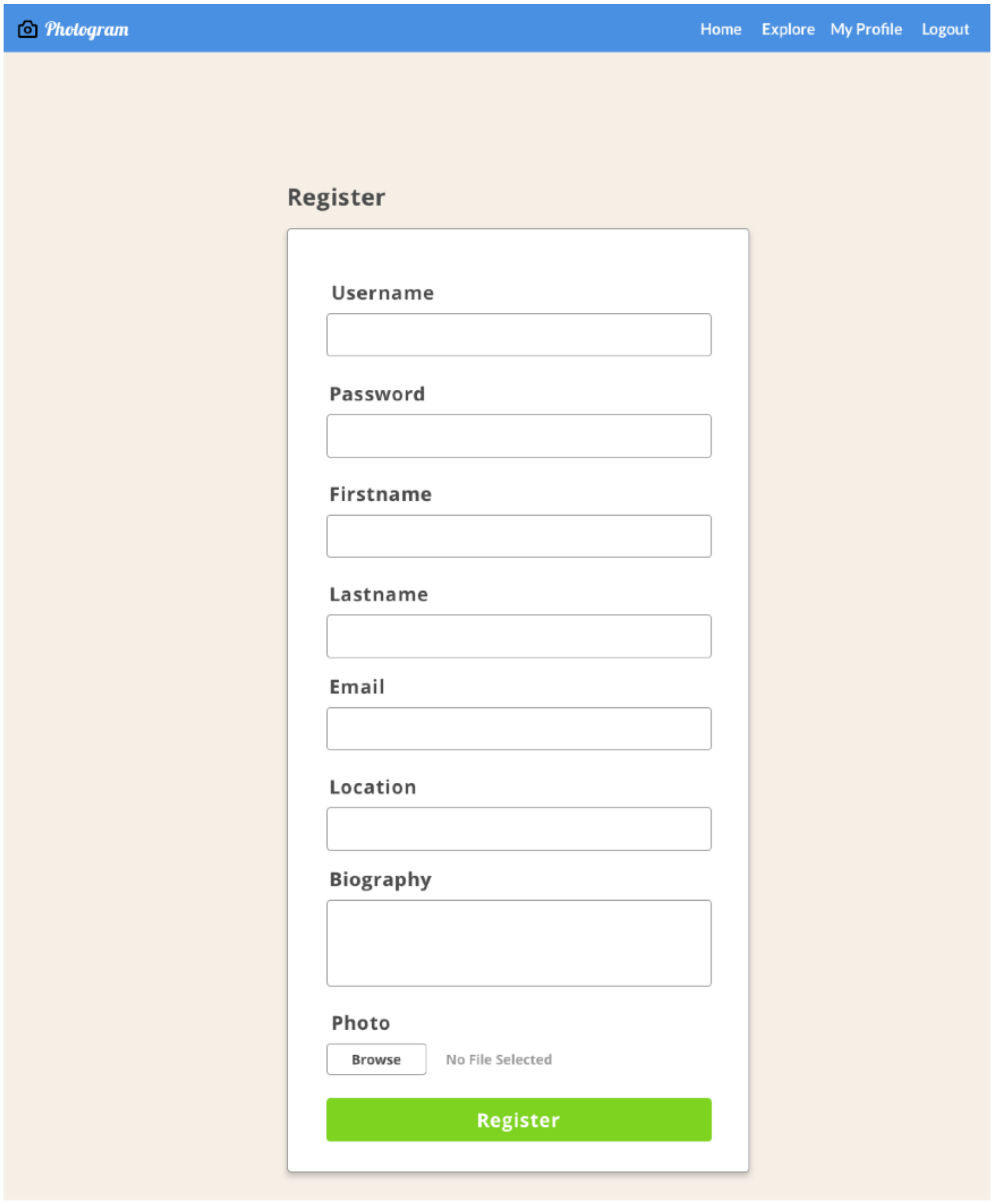


FIGURE 2. USER REGISTRATION

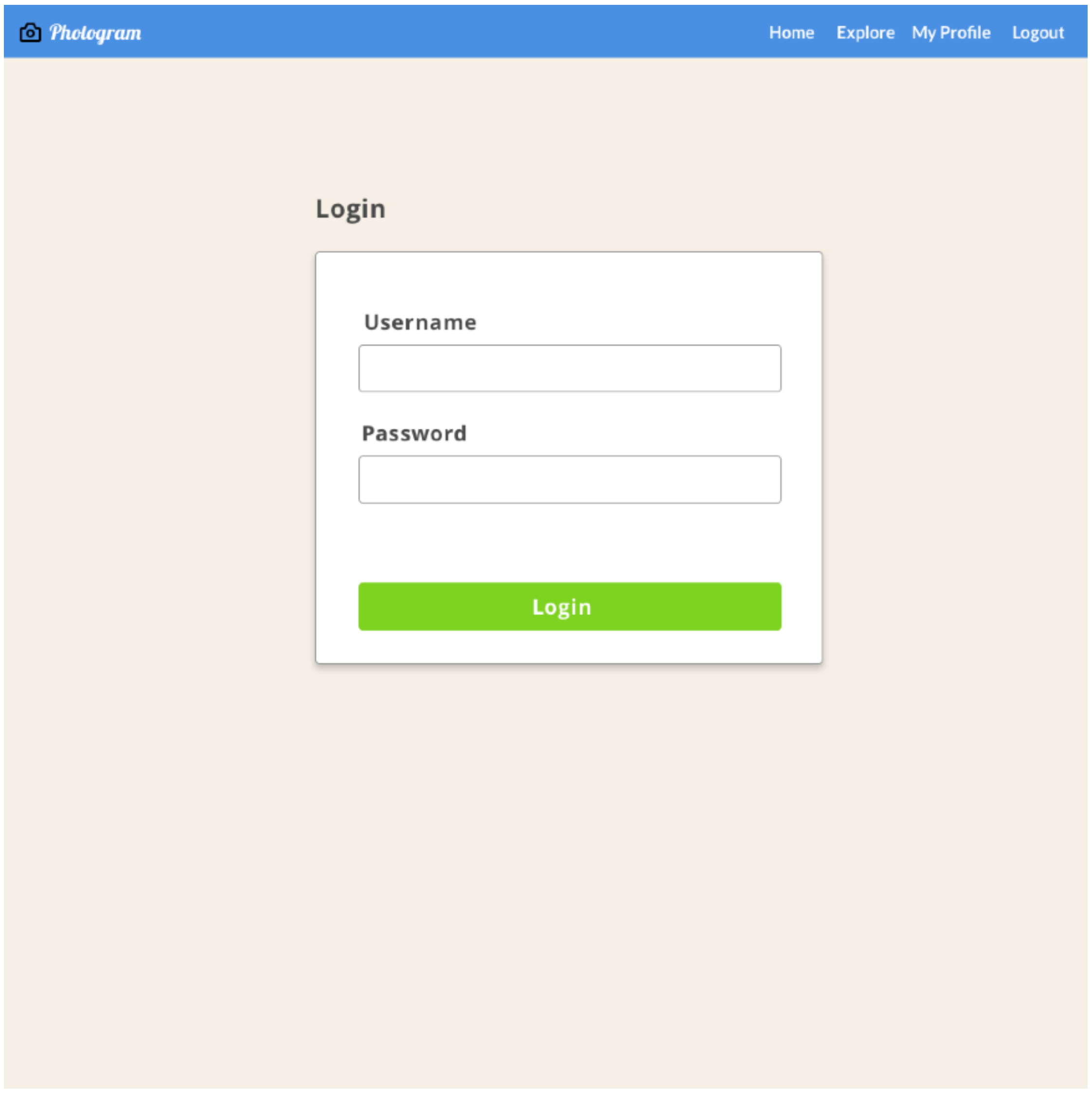
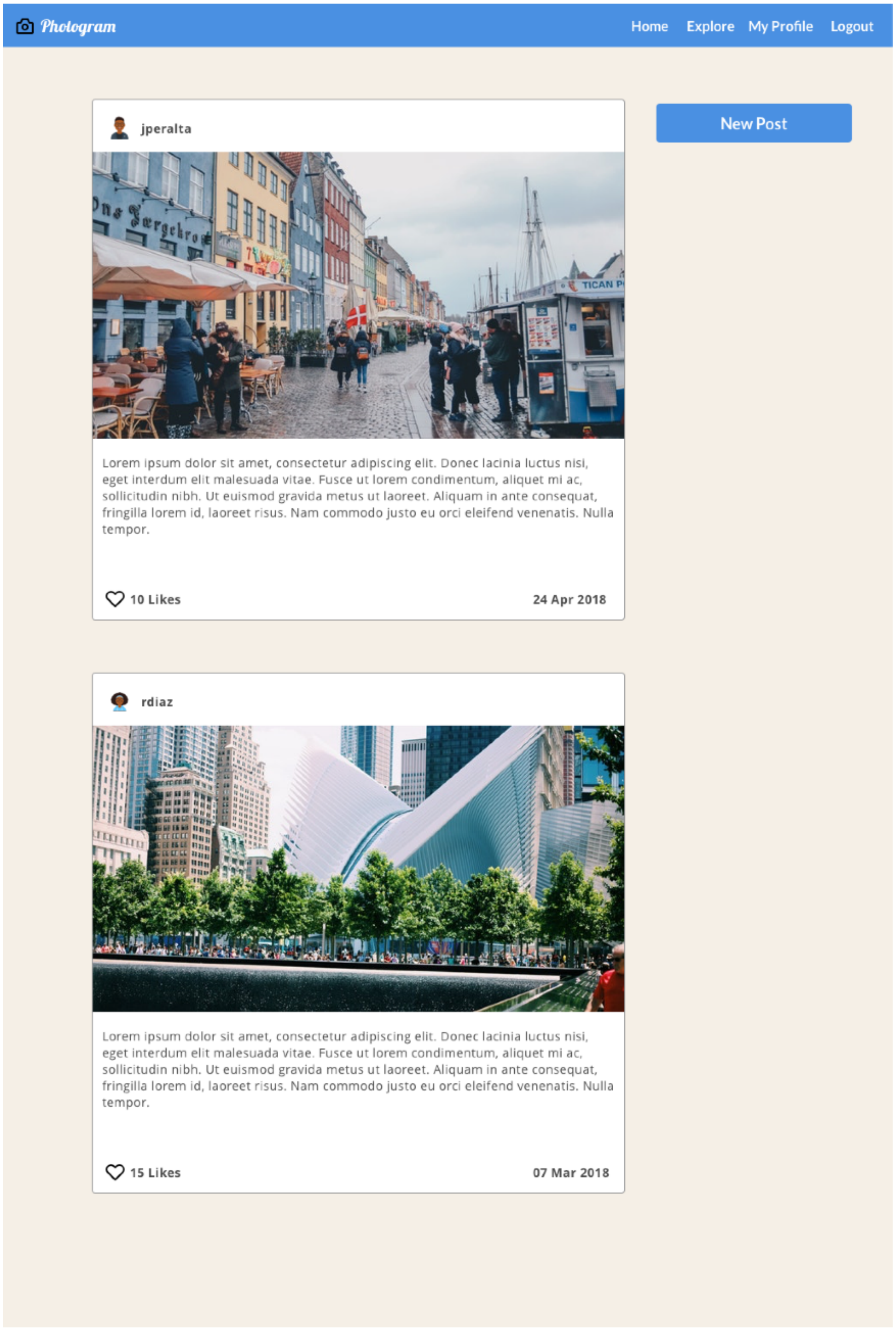


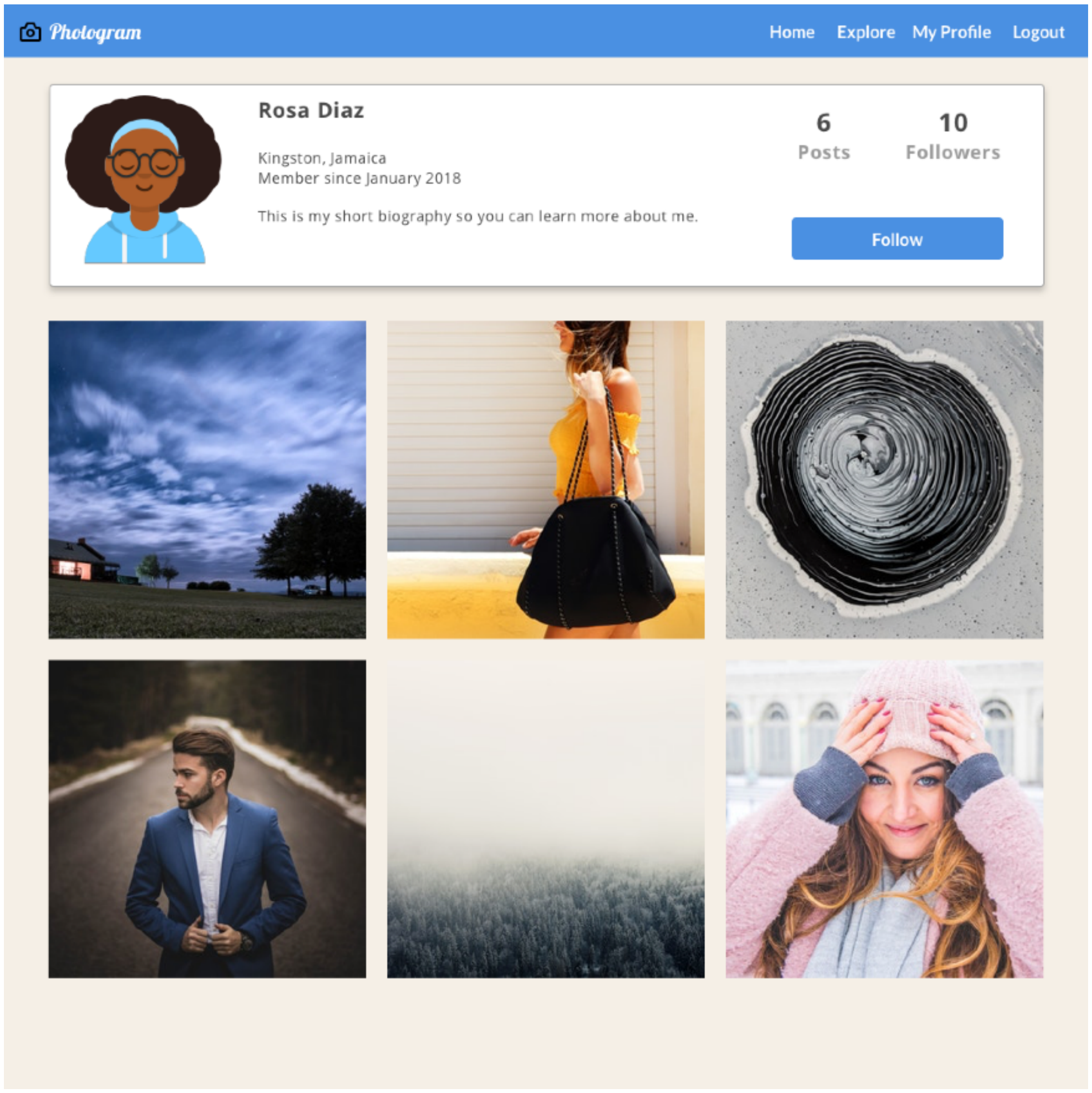
FIGURE 3. LOGIN



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec lacinia luctus nisi, eget interdum elit malesuada vitae. Fusce ut lorem condimentum, aliquet mi ac, sollicitudin nibh. Ut euismod gravida metus ut laoreet. Aliquam in ante consequat, fringilla lorem id, laoreet risus. Nam commodo justo eu orci eleifend venenatis. Nulla tempor.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec lacinia luctus nisi, eget interdum elit malesuada vitae. Fusce ut lorem condimentum, aliquet mi ac, sollicitudin nibh. Ut euismod gravida metus ut laoreet. Aliquam in ante consequat, fringilla lorem id, laoreet risus. Nam commodo justo eu orci eleifend venenatis. Nulla tempor.

FIGURE 4: EXPLORE PAGE. THIS DISPLAYS POSTS BY ALL USERS



Rosa Diaz

Kingston, Jamaica
Member since January 2018

This is my short biography so you can learn more about me.

6 Posts

10 Followers

FIGURE 5: USER PROFILE WITH PHOTOS

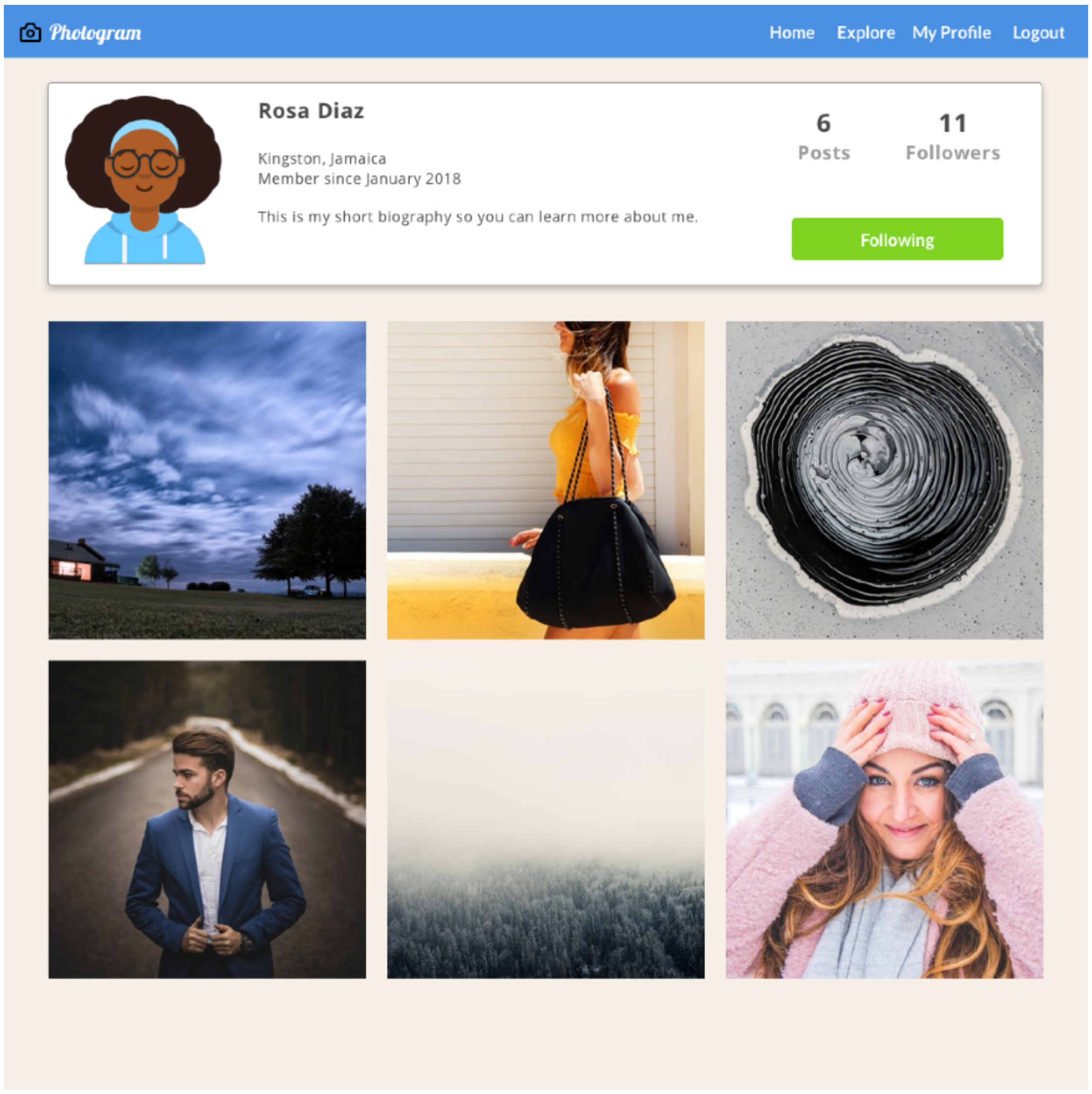


FIGURE 6: WHEN FOLLOW BUTTON IS CLICKED IT CHANGES TO "FOLLOWING" AND THE FOLLOWER COUNT INCREASES.

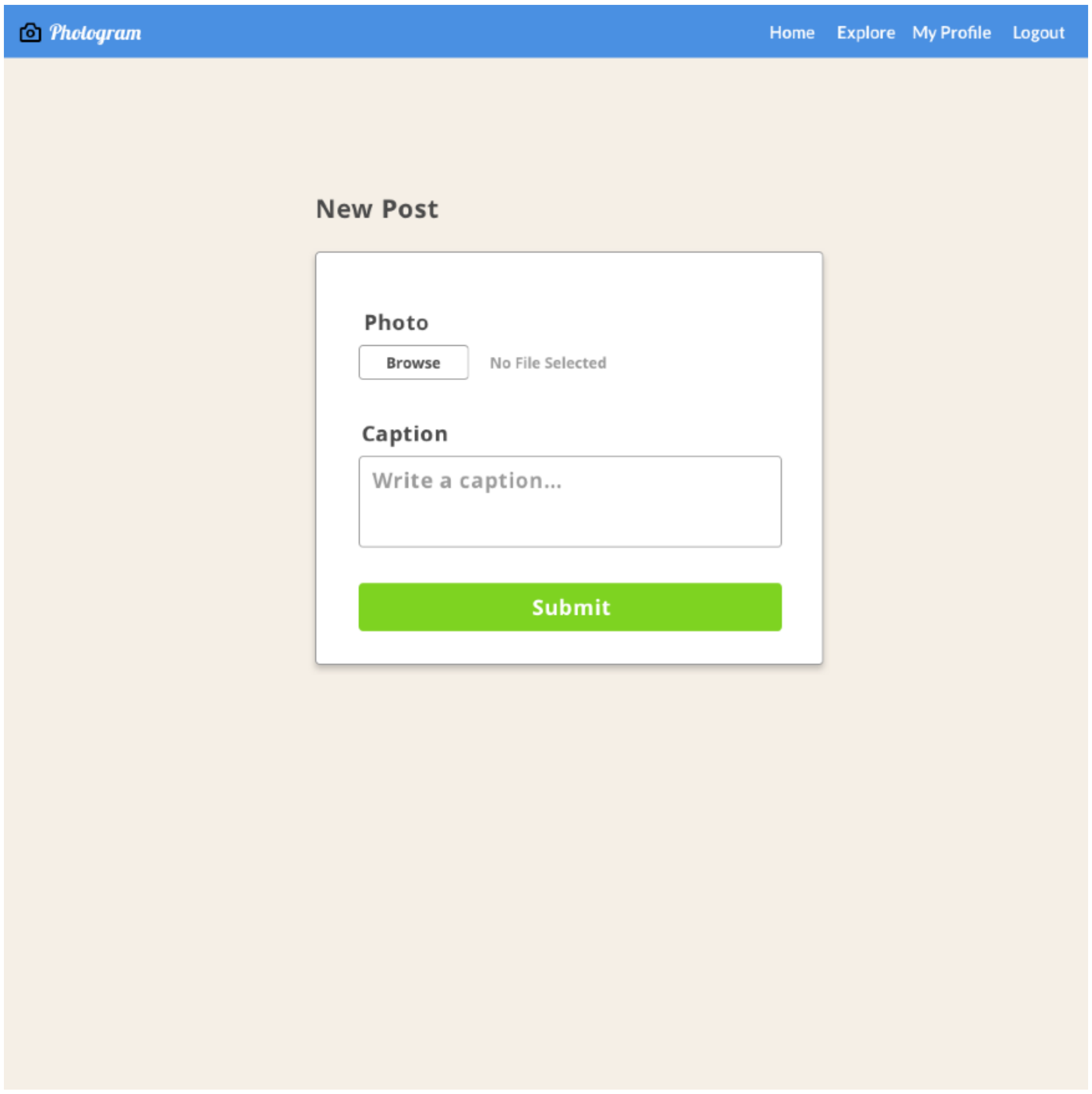


FIGURE 7: FORM FOR A NEW POST