Project 2

Photogram Web Application

Due: April 26, 2024 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 5 persons.

If you have not already done so, please ensure that you let me know by email (yannick.lynfatt@uwimona.edu.jm) who your group members.

Background

For this project we will create a fictional Instagram clone called Photogram that we can share photos with our friends and family. Users can also like photos and follow other users.

You may use the Starter code at: https://github.com/uwi-info3180/info3180-vuejs-flask-starter if you need to.

Database Schema

Posts

Column Name	Data Type
id	Integer
caption	String
photo	String
user_id	Integer
created_on	DateTime

Likes

Column Name	Data Type
id	Integer
post_id	Integer
user_id	Integer

Follows

Column Name	Data Type
id	Integer
follower_id	Integer
user_id	Integer

Users

Column Name	Data Type
id	Integer
username	String
password	String
firstname	String
lastname	String
email	String
location	String
biography	String
profile_photo	String
joined_on	DateTime

Note: Ensure that you define the appropriate models and create your database migrations and that these are included in your Git repository.

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/v1/register	Accepts user information and saves it to the database
POST	/api/v1/auth/login	Accepts login credentials as username and password
POST	/api/v1/auth/logout	Logout a user
POST	/api/v1/users/{user_id}/posts	Used for adding posts to the users feed
GET	/api/v1/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/v1/posts	Return all posts for all users

HTTP Method	Route	Description
POST	/api/v1/posts/{post_id}/like	Set a like on the current Post by the logged in User

Note: More details about the API and some 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 error message on 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

Route	Description
`/posts/new`	`Allow the user to add a new post`

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

Appendix

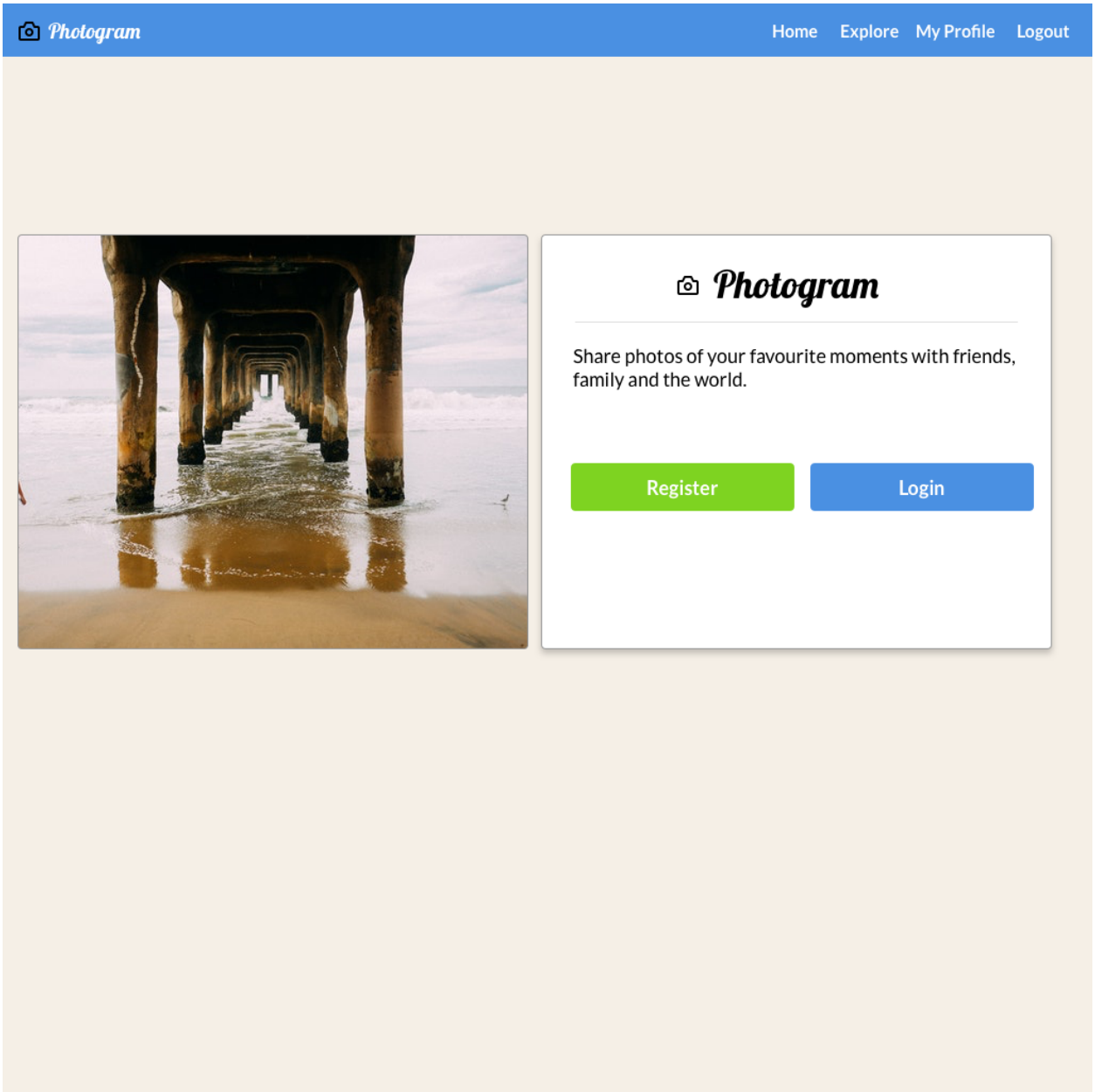

FIGURE 1. HOME PAGE

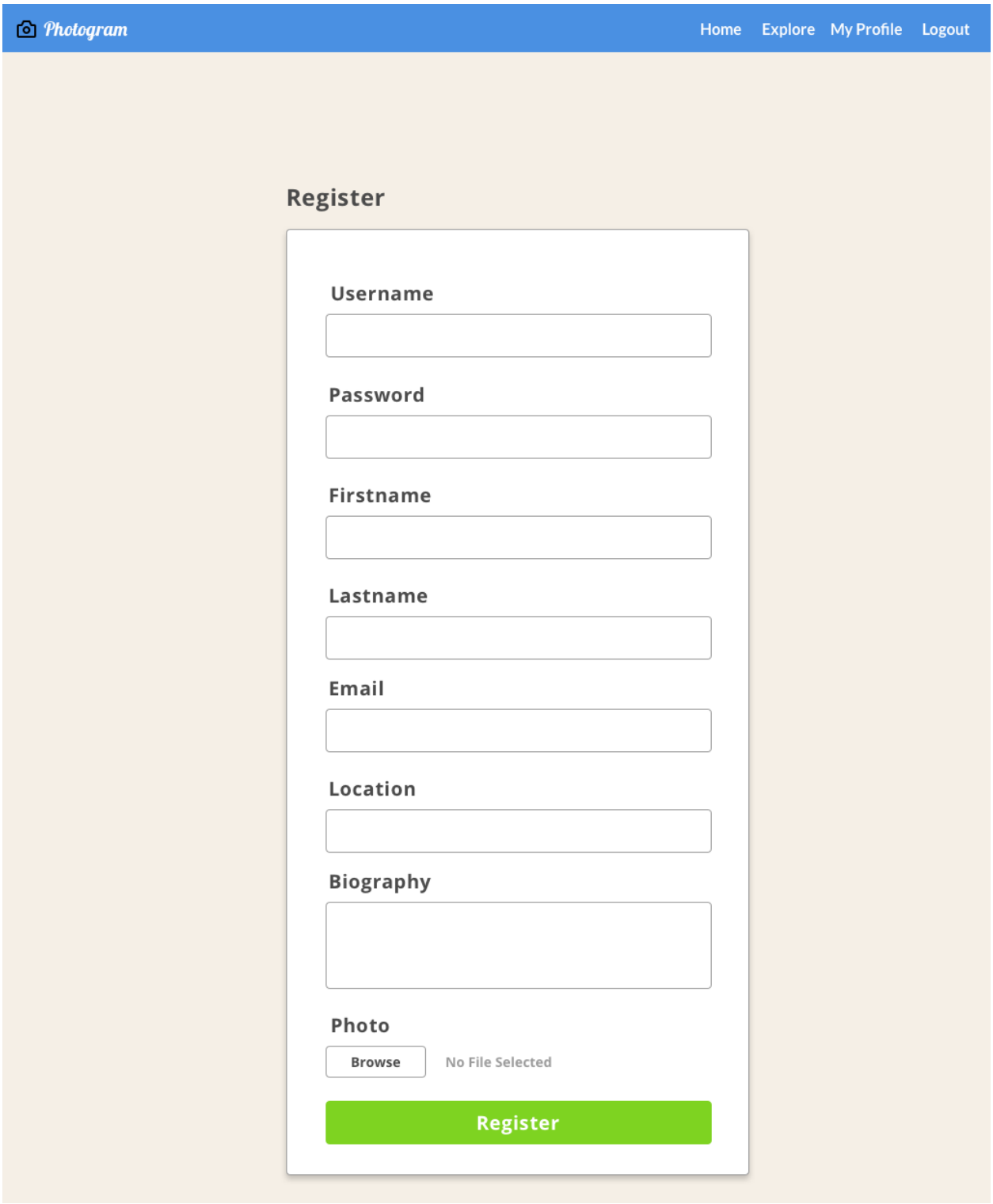

FIGURE 2. USER REGISTRATION

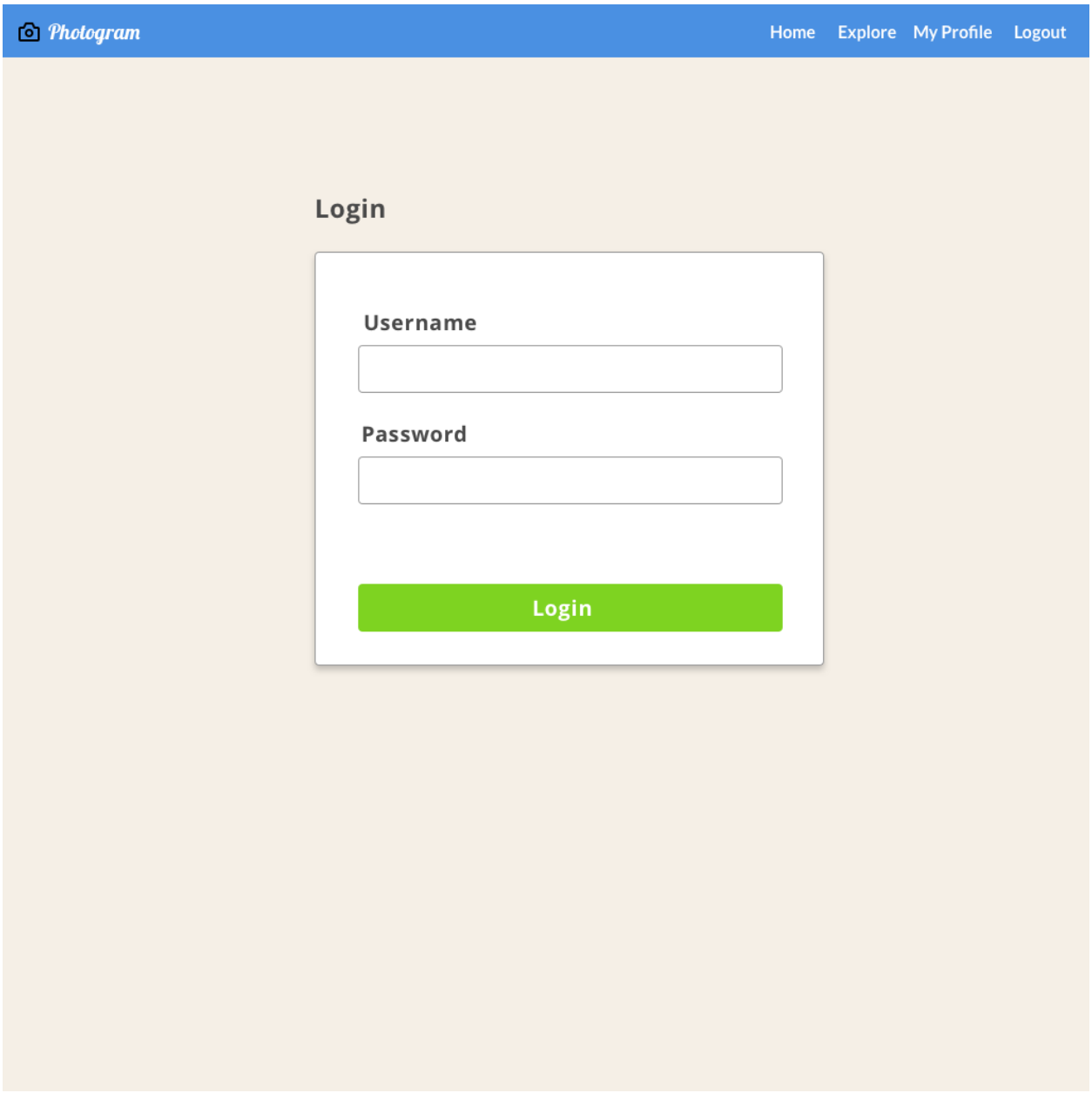

FIGURE 3. LOGIN

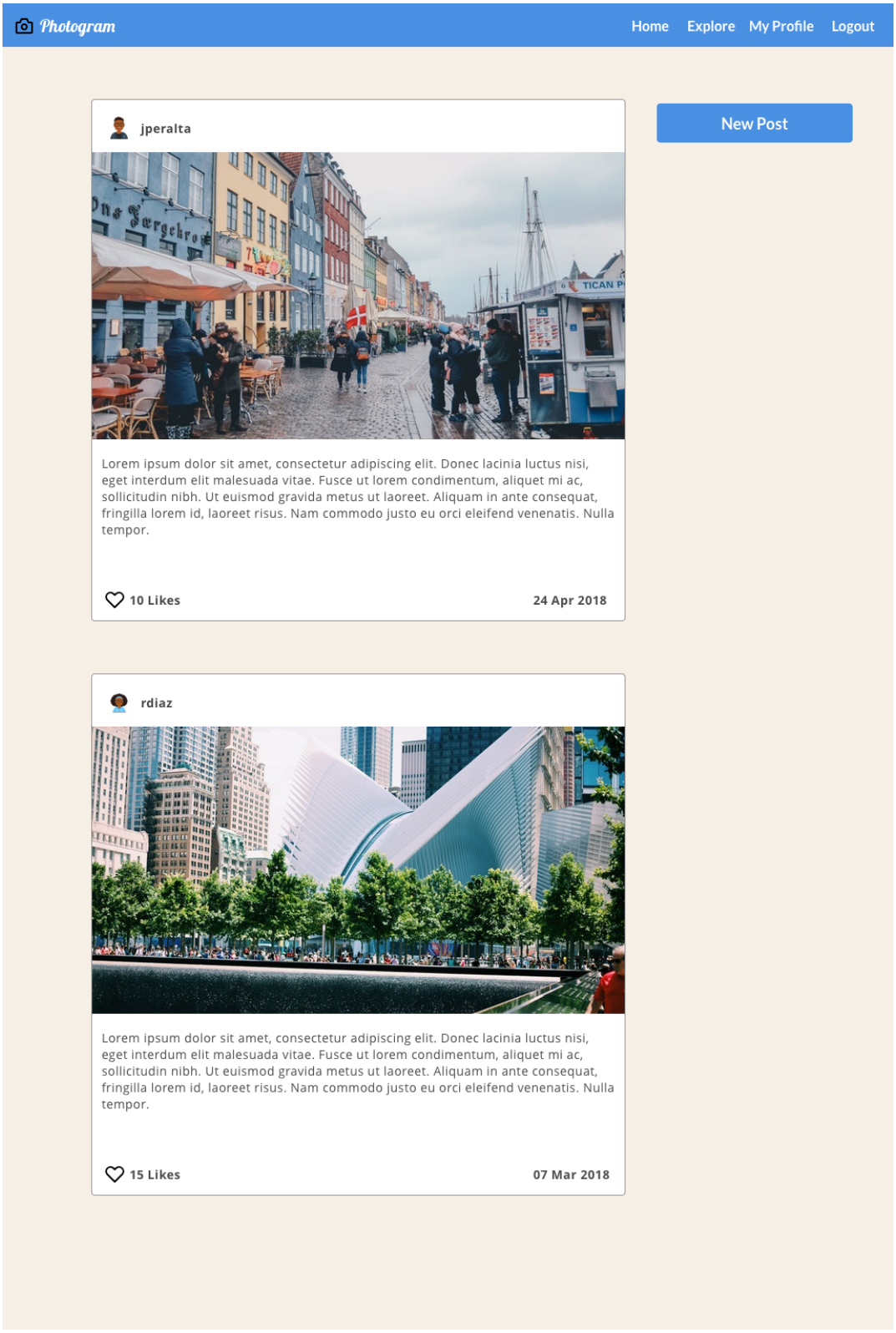

FIGURE 4: EXPLORE PAGE. THIS DISPLAYS POSTS BY ALL USERS

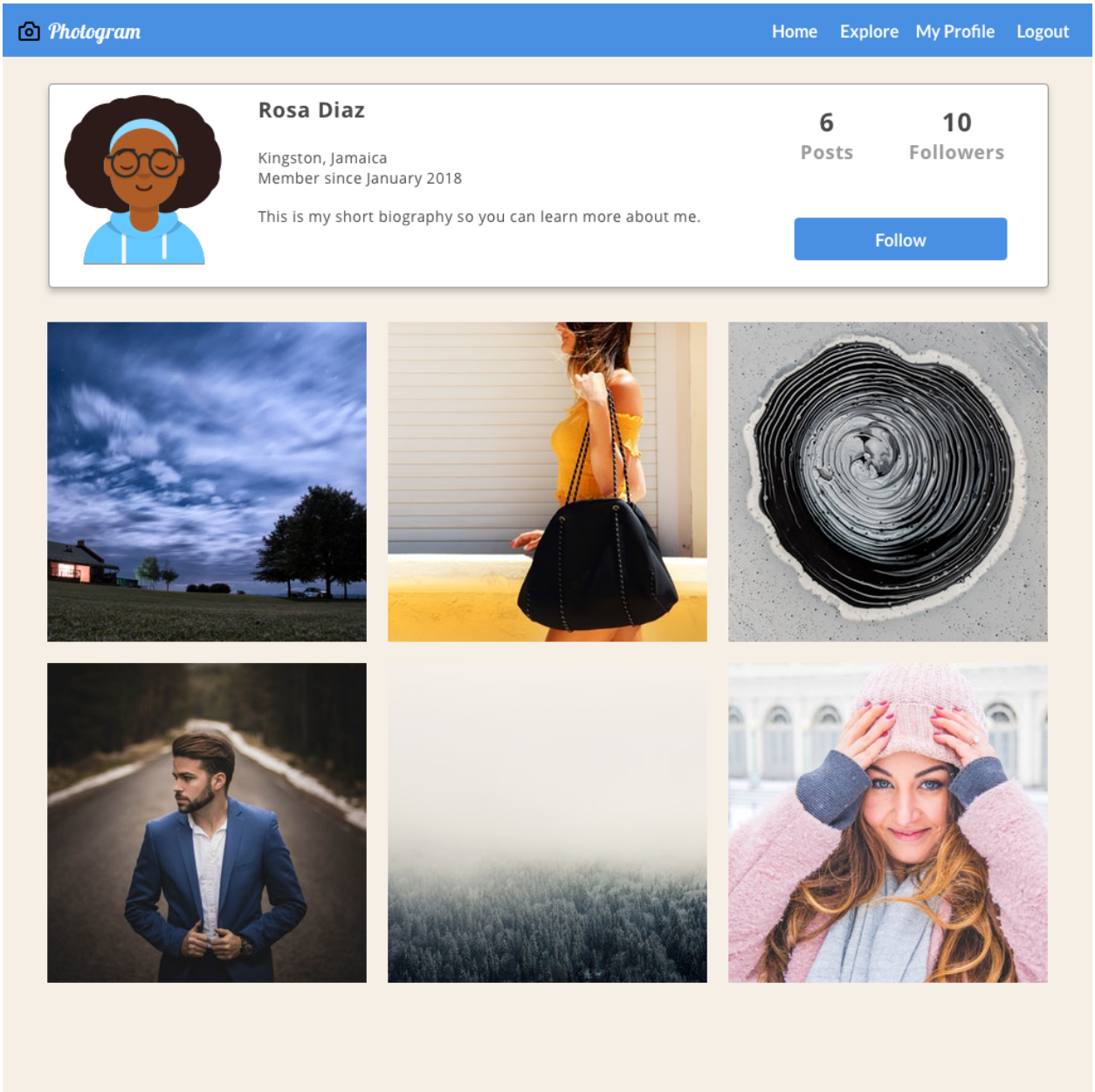

FIGURE 5: USER PROFILE WITH PHOTOS

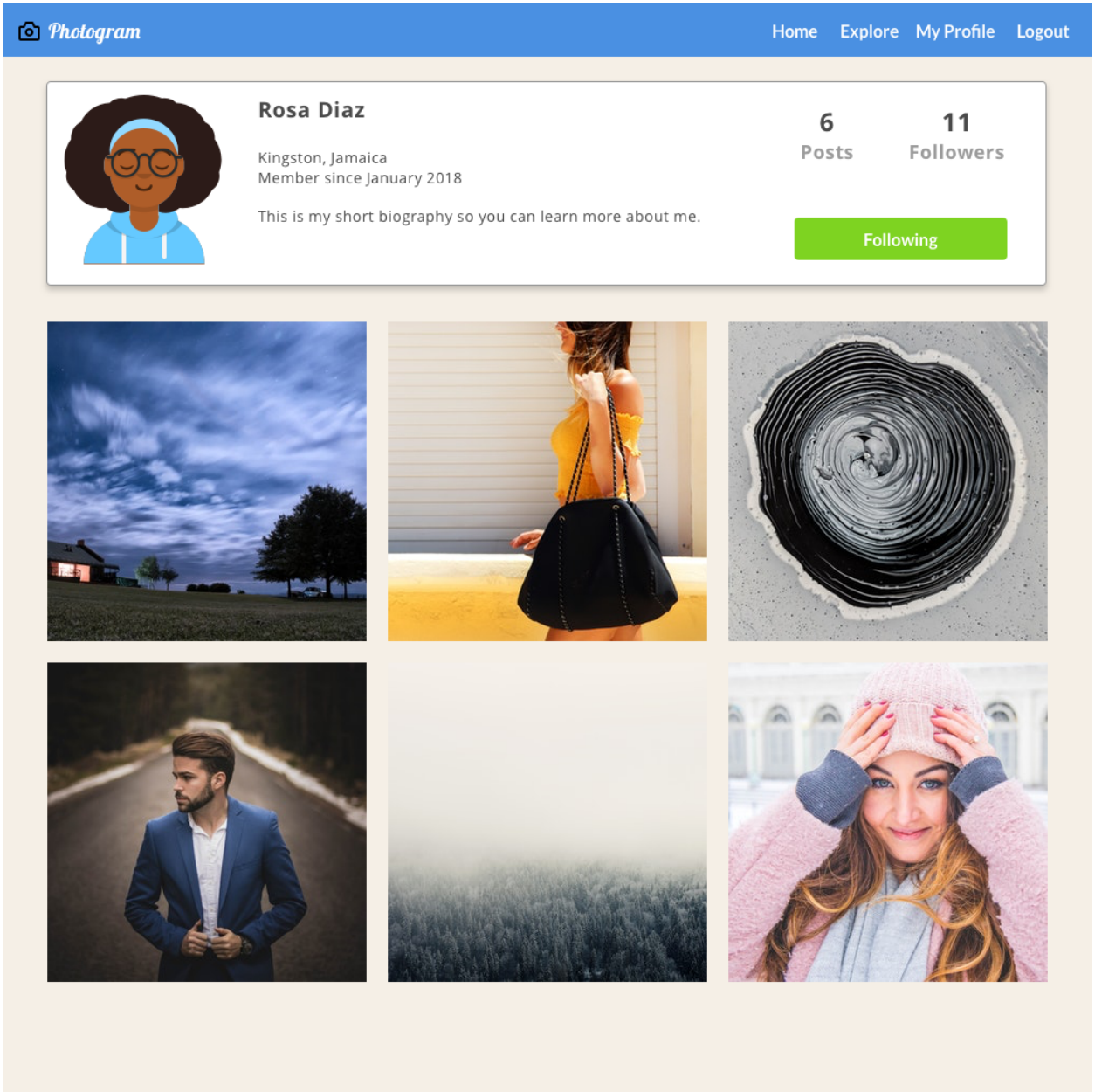

FIGURE 6: WHEN FOLLOW BUTTON IS CLICKED IT CHANGES TO "FOLLOWING" AND THE FOLLOWER COUNT INCREASES.

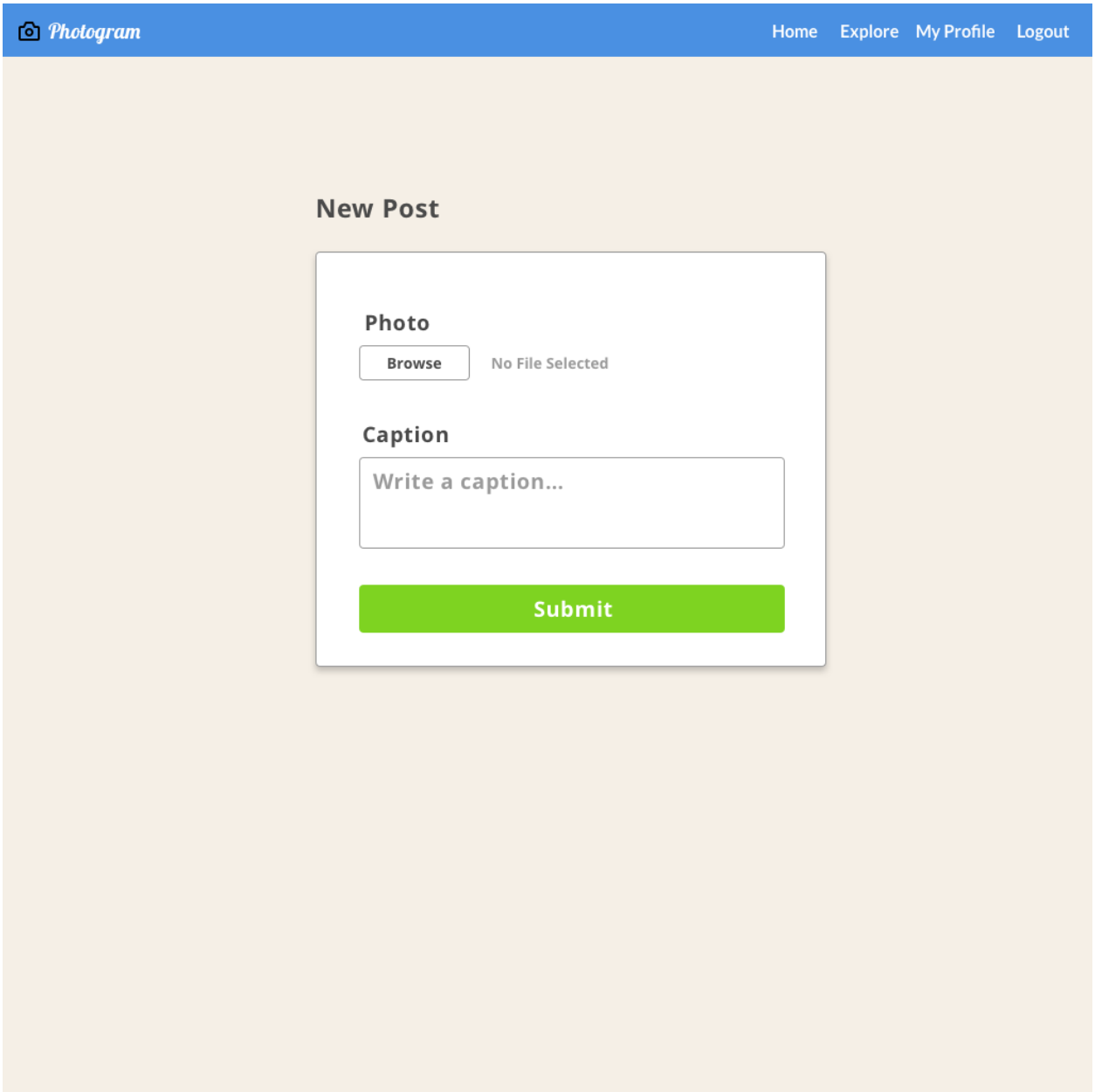

FIGURE 7: FORM FOR ADDING A NEW POST