Title: SkyShare

Who: Mateusz Muszynski, Chenghao Xiong, Jennifer Kim, Genesse Miles

Project Description:

SkyShare is a platform that **connects space enthusiasts** together & is dedicated to showcasing the best **space photos** from all across the country! Upon creating/ logging into an account, users can access the Home page, Following page, My Photos page, the Public Map, and NASA photos.

On the Home page, the user can see their **two most recently uploaded photos** and a slideshow of photos from the other users they are following.

On the My Photos page, users can publicly **upload their coolest space photos** along with tagging the **location** they were taken. These photos are displayed on this page, too.

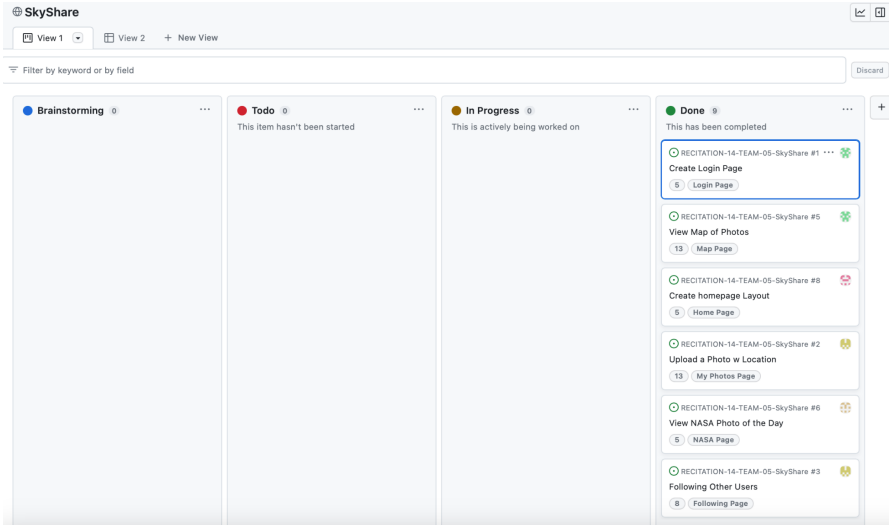
On the Following page, users can **follow their friends** by inputting their usernames and, in turn, see the photos they have posted.

On the Public Map, there are markers for the 50 US states which display a **carousel of photos taken at that state by all users on SkyShare**. For example, all user photos uploaded in Colorado are displayed in the carousel on Colorado's red marker on the map.

On the NASA page, users can view **NASA's photo of the day** along with a description of the photo (updates daily).

Project Tracker - GitHub project board:

Link to our Project Tracker: https://github.com/users/mateusz-muszynski/projects/3
Screenshot showing our project in our project tracker below:
Here is our project board (everything completed; all of our website's features work correctly!!!)



Video:

https://github.com/mateusz-muszynski/RECITATION-14-TEAM-05-SkyShare/tree/main/milestoneSubmissions: The video is called "Sky share demo.mp4". It can be viewed by downloading the file!

VCS: Link to our git repo: https://github.com/mateusz-muszynski/RECITATION-14-TEAM-05-SkyShare

Contributions:

Jennifer:

- Implemented **uploading & deleting photos** feature (connected Cloudinary API) with location, storing this info in the photos sql table
    - The user's uploaded photos display on the My Photos page (3 photos per row)
- Implemented **following & unfollowing feature**, storing this info in the followers sql table
    - Usernames the user is following's photos are displayed on the Following page (3 photos per row per Bootstrap card)
- Updated project board for weeks 2-4
- Release notes 1,2, and 3

Genesse:

- Created wireframes
- NASA API gathers and stores data in the nasa sql table
    - Implemented the **NASA photos** page; using the NASA API, this page displays NASA's Photo of the Day along with a description. This page updates daily, as NASA's Photo of the Day changes daily
- Release note 4

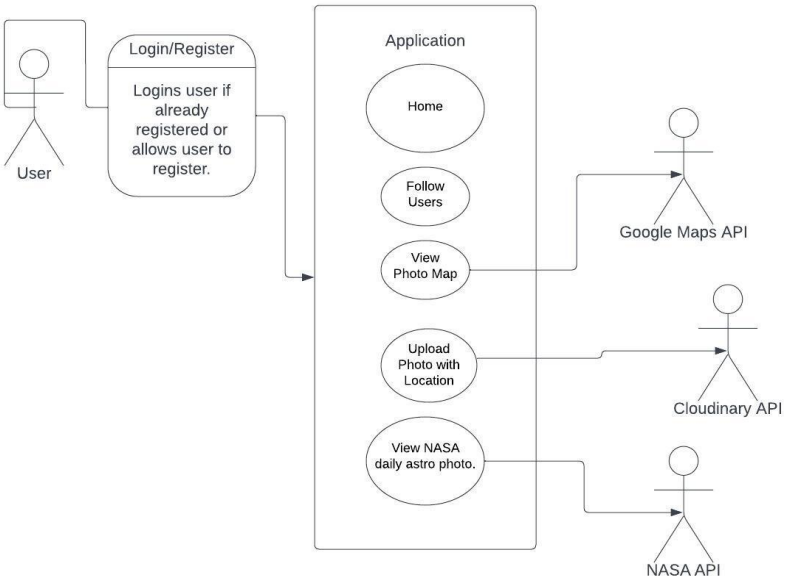
Chenghao:

- Created the **login & registration** pages, storing info in the users sql table
- Created a map method:
    - Created the states SQL table (and inserted 50 values of long/lat of each US State) and completed our **map feature** via the Google Maps API. When the user clicks on the marker on the map, a carousel of photos pop up showing all the photos uploaded at that location

Mateusz:

- Created and **designed the layout of the homepage**.
- Homepage features a user profile card which displays the user's username and underneath two of the users most recently uploaded photos.
- There is a carousel of three images which fetches a random user you follow and a random image which they have uploaded, along with a caption informing where the user took the photo.

Use Case Diagram:



Test results:

In Lab 11, we created a Test Plan of 3 UATs (doc in milestoneSubmissions folder). Here are screenshots of the 3 UAT descriptions and their test results & observations below:

#1) Our first Test Plan was for the following other users feature (screenshot from Lab 11 doc):

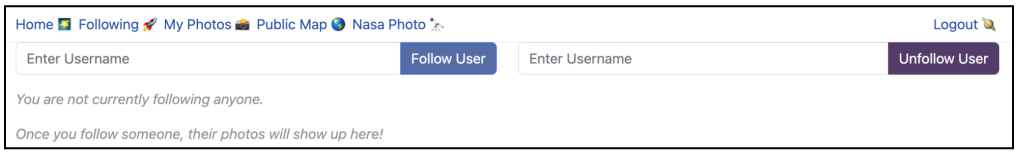
Following other users feature and its acceptance criteria

1. Users can follow other users by looking up their username. The user being followed does not have to follow the original user back. This data will be stored in the followers database
    a. On the nav bar, there will be a “following” option which will display the usernames the user is following and just the photos uploaded by those usernames
2. Test data is valid and invalid usernames
3. The user activity that will be executed in order to verify proper functionality of the feature is the user typing in a username
    a. if user enters a valid username to follow, a message will appear that they have successfully followed them and that username will be added to the followers table associated with the original user
    b. If user enters username of someone they already follow, a message will appear that they are already following them
    c. If user enters invalid username then message will appear saying something like “user does not exist”
4. The test environment is the web application.
5. User acceptance testers should know this acceptance criteria and test accordingly, verifying that upon entering a valid username, this username should show up in the followers table

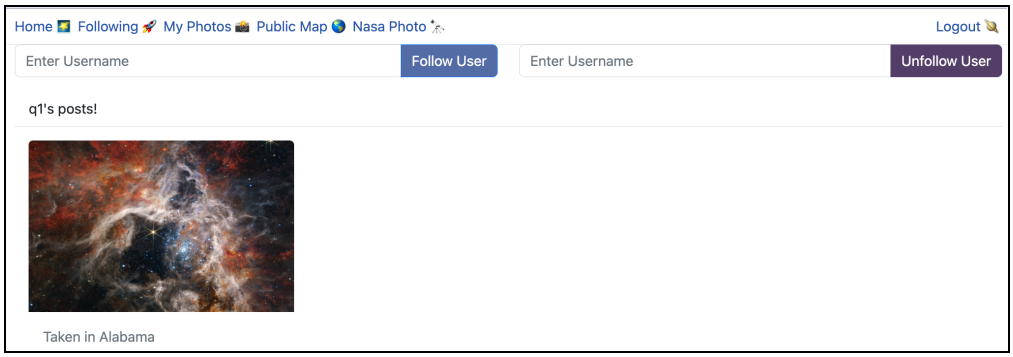
Test results: The project writeup says “You should be testing a minimum of 4 use cases.”

Case 1 [positive]) user enters a valid username to follow:

This screenshot below shows the default Following page of a newly created account:

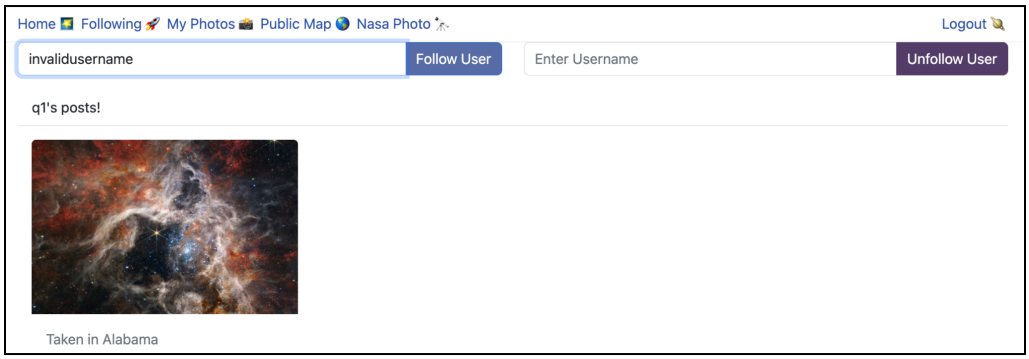


The username “q1” is a valid username. Upon typing in q1 and clicking the Follow User button, q1’s posts successfully display:



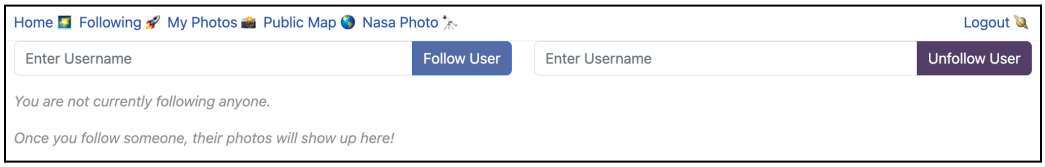
Case 2 [negative]) user enters an invalid username to follow

"invalidusername" is a username that doesn't exist. Upon typing "invalidusername" and clicking the Follow User button, the user is just redirected back to the Following page as shown here. Nothing on the page has changed, as it rightfully shouldn't, since the user can't follow other users who don't exist.



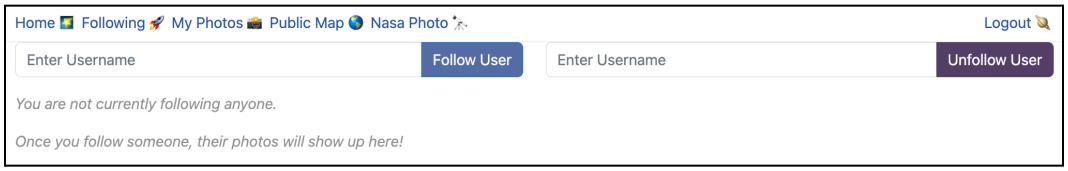
Case 3 [positive]) user enters a valid username to unfollow

So currently, the user is following q1. Upon entering q1 into the Unfollow User field and clicking on the button, q1's profile is successfully removed from the user's following page:



Case 4 [negative]) user enters an invalid username to unfollow

Similar to case 2, "invalidusername" is a username that doesn't exist. Upon typing "invalidusername" and clicking the Unfollow User button, the user is just redirected back to the Following page as shown here. Nothing on the page has changed, as it rightfully shouldn't.



Observations (directly answering the 5 questions in the writeup):

1) What are the users doing?

- The users are (un)following other users by entering their usernames.

2) What is the user's reasoning for their actions?

- For following, the reasoning is so the user can easily access the photos their friends have posted.

- For unfollowing, the reason is if the user doesn't want to have x user's posts displayed on their Following page anymore.

3) Is their behavior consistent with the use case?

- Their behavior is consistent with the use case. On the following page, the user is prompted to type usernames.

4) If there is a deviation from the expected actions, what is the reason for that?

- For both the follow and unfollow feature, the only deviations from the expected actions are when the user types in invalid usernames. The reasons could be that they just inputted the username wrong (typo), or they're just not thinking and trying to unfollow an account that does not exist *OR an account that does exist, but the user is not following them currently.*

5) Did you use that to make changes to your application? If so, what changes did you make?

- The only change we made is that we do not display messages such as "you're already following this user". This is because we originally planned to use res.render + message but instead implemented res.redirect.

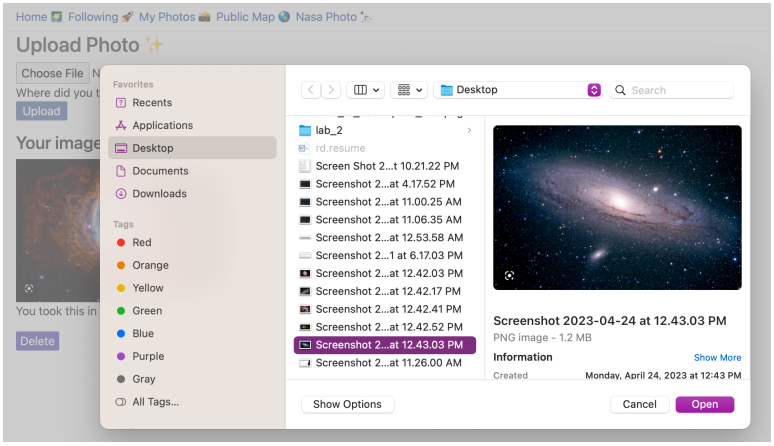
#2) Our 2nd Test Plan was for photo uploading (screenshot from Lab 11 doc):

Photo Posting Feature and its acceptance criteria

1. Users will be able to post a photo to their account which will be stored in the database
    a. The photo is public (visible to everyone user)
    b. On the nav bar, there will be a "public" option which will display all the photos uploaded by every user
2. The user activity that will be executed in order to verify proper functionality of the feature is that the user will be able to click a button to upload an image (.jpg) from their computer, and if successful this image will display on the page and stored in the database
3. The test environment is the web application
4. The test results that will be used to test the feature are if a user uploads an unsupported file such as a pdf, they will be prompted with an error message saying to only upload images. The test result will be successful when the image displays on the page
5. User acceptance testers should know this acceptance criteria and test accordingly, verifying that the image actually uploads onto the page if successful, for example

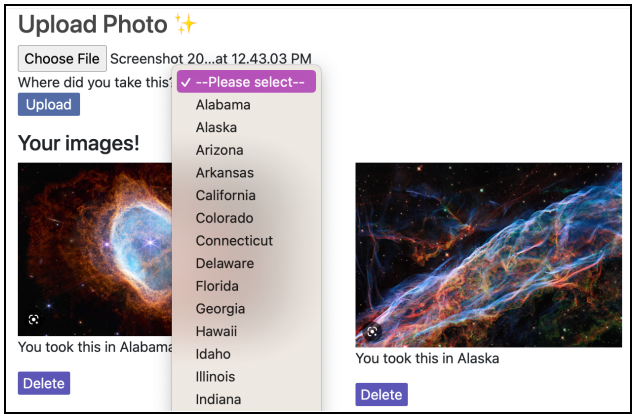
Test results: The project writeup says "You should be testing a minimum of 4 use cases."

Case 1 [positive]) upload a valid photo file (such as jpg, png)

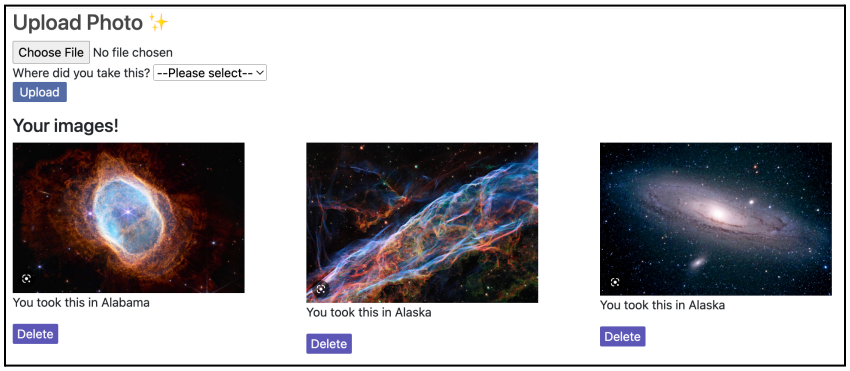


In this screenshot above, the user has clicked the "Choose File" button and has selected this PNG image to upload. They have not clicked "Open" yet.

Upon clicking "Open", they are prompted to select where they took the photo and must select from the dropdown:

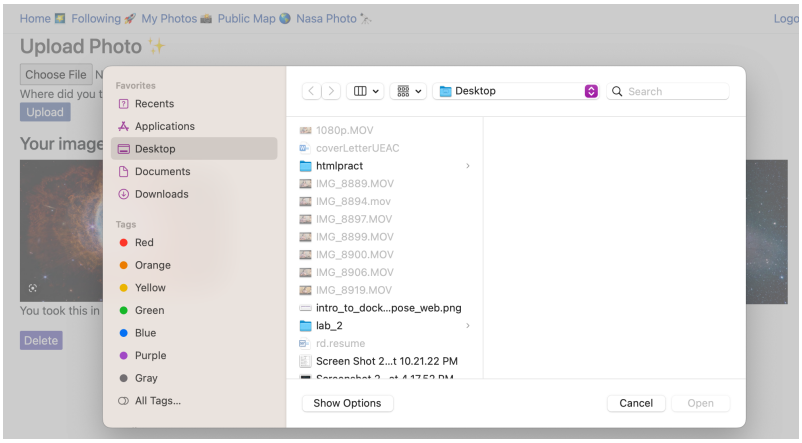


Once they select the location & click the blue Upload button, that photo displays as shown:



Case 2 [negative]) attempt to upload an invalid file that is not a photo

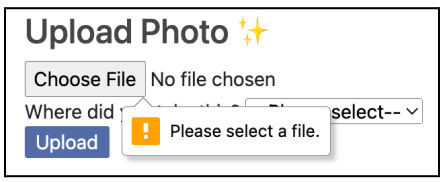
So upon the user clicking the 'Choose File' button, they are prompted with files they can upload from their device as displayed below:



As shown, non-image files are greyed out and cannot be chosen to upload. So, it is only possible that the user uploads photo files.

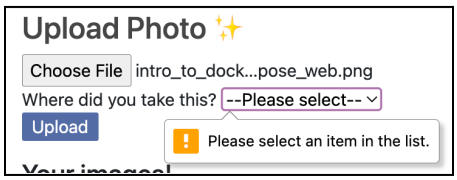
Case 3 [negative]) attempt to click the "Upload" button without selecting a photo to upload

When the user clicks the blue 'Upload' button without selecting a file, they are prompted with a message:



Case 4 [negative]) attempt to click the "Upload" button without selecting a location

Even if the user has selected a photo file, if they haven't selected a location, they are prompted with this message, and their photo will not be uploaded until they've select the location:



Observations (directly answering the 5 questions in the writeup):

1) What are the users doing?

- The users are uploading photos along with the location.

2) What is the user's reasoning for their actions?

- The reasoning is so the user can store space photos associated with their account and have them displayed on the My Photos page.

3) Is their behavior consistent with the use case?

- Their behavior is consistent with the use case. The purpose of the My Photos page is so the user can upload their photos.

4) If there is a deviation from the expected actions, what is the reason for that?

- The deviations are if the user…
  - …tries to upload a non-photo file. The reason is the user may not know only photo files are valid.
  - …clicks the blue "Upload" button without having the 2 necessary fields filled out. The reason is the user just forgot to fill in those fields.

5) Did you use that to make changes to your application? If so, what changes did you make?

- We added a delete photo button.

#3) Our 3rd Test Plan was for the login/ registration feature (screenshot from Lab 11):

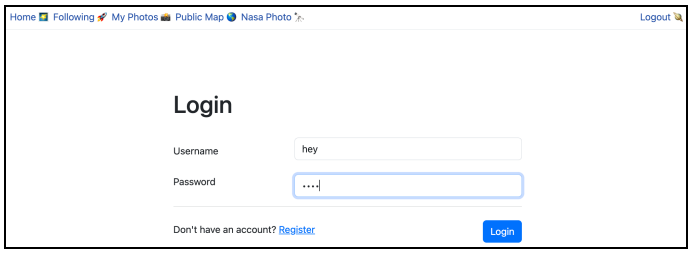
Login/Registration Feature and its acceptance criteria

1. Users will be able to register a specific username and password that is linked to their account. This username and password will be added to a database, stored in users table in users_db database. They will then be able to log back into their account using that information.
2. Test data that'll be used to test the feature include invalid usernames and passwords on the login page and registering for an account with a username that already exists; in those cases users will be prompted with specific information about why their request was unsuccessful
3. The environment being used to test the feature is using the web application
4. The test results are: if user logins in successfully they are redirected to the home page. Else, they enter the registration page if the username they tried logging in w does not exist. Or they are prompted with something like "Incorrect username or password". If the user registers successfully they are redirected to the login page, else they stay on the registration page
5. User acceptance testers should know this acceptance criteria and test accordingly, verifying the results of each possibility

Test results: The project writeup says "You should be testing a minimum of 4 use cases."

Case 1 [positive]) log in with valid username and correct password

Correct credentials are: username: hey password: heyy

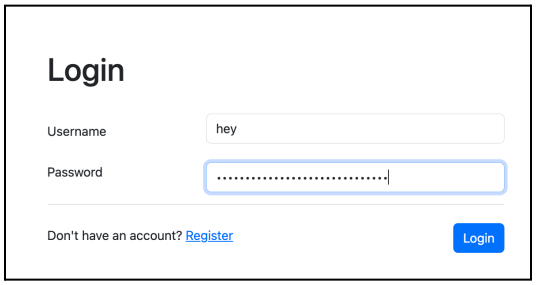


In this screenshot above, the user has entered hey as the username and heyy as the password. They have not clicked the Login button yet.

Once they click the Login button, they are successfully redirected to the Home page shown here:

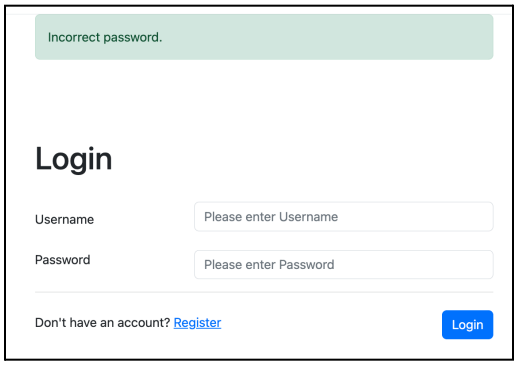
Case 2 [negative]) attempt to log in with incorrect password

Correct credentials are: username: hey password: heyy



In this screenshot above, the password the user has entered is clearly not “heyy”. They have not clicked the Login button yet.

Upon clicking the Login button, they are prompted with a message and redirected back to the Login page to try again:

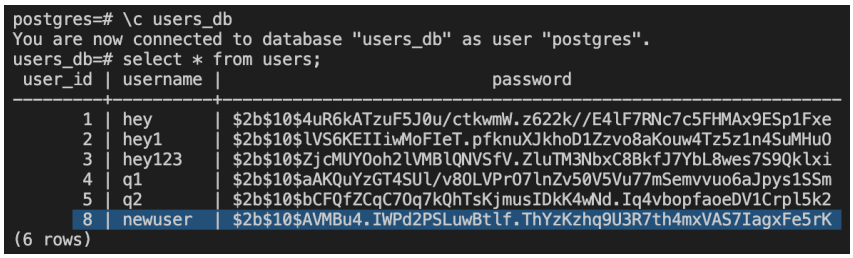


Case 3 [positive]) registering with a new username and new password

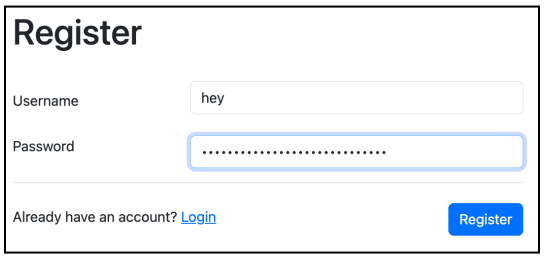
In this screenshot above, the user is registering for a new account. The username is “newuser” and the password is “newuser”. The username “newuser” is unique. They have not clicked the register button yet.

Upon clicking the register button, they will be redirected to the login page so they can login with their newly created account.

Here is a screenshot of the users table showing “newuser” was successfully added to the users table with a hashed password:

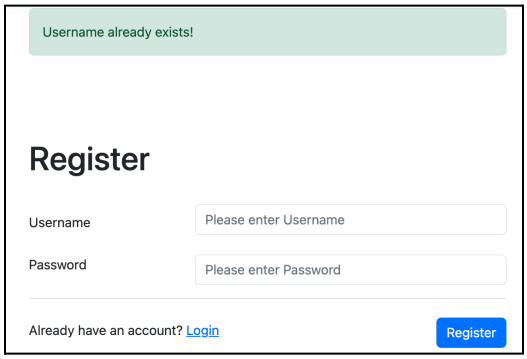


Case 4 [negative]) attempt to register with already existing username



In the screenshot above, the user is trying to register with the username “hey”, which is an already existing username. They have not clicked the Register button yet.

Upon clicking the button, they are prompted with a message and are redirected back to the register page so they can try again:



Here is a screenshot of the users table that shows “hey” was not added a second time into the users table, since that username already exists:

```
users_db=# select * from users;
 user_id | username |                          password
---------+----------+------------------------------------------------------------
       1 | hey      | $2b$10$4uR6kATzuF5J0u/ctkwmW.z622k//E4lF7RNc7c5FHMAx9ESp1Fxe
       2 | hey1     | $2b$10$lVS6KEIIiwMoFIeT.pfknuXJkhoD1Zzvo8aKouw4Tz5z1n4SuMHuO
       3 | hey123   | $2b$10$ZjcMUYOoh2lVMBlQNVSfV.ZluTM3NbxC8BkfJ7YbL8wes7S9Qklxi
       4 | q1       | $2b$10$aAKQuYzGT4SUl/v8OLVPr07lnZv50V5Vu77mSemvvuo6aJpys1SSm
       5 | q2       | $2b$10$bCFQfZCqC7Oq7kQhTsKjmusIDkK4wNd.Iq4vbopfaoeDV1Crpl5k2
       8 | newuser  | $2b$10$AVMBu4.IWPd2PSLuwBtlf.ThYzKzhq9U3R7th4mxVAS7IagxFe5rK
(6 rows)
```

Observations (directly answering the 5 questions in the writeup):

1) What are the users doing?

- The users are logging in or registering for an account by entering a username and password.

2) What is the user's reasoning for their actions?

- For logging in, the reasoning is accessing their account and their uploaded photos, the photos uploaded by those they are following, the public map which displays all photos uploaded by every user on SkyShare in each of the 50 states, and viewing NASA's photo of the day.
- For registering, the reasoning is to create an account to have access to all the features described above.

3) Is their behavior consistent with the use case?

- Their behavior is consistent with the use case. Upon entering the site, the user is prompted to log in or register; they cannot access the site's features without logging in.

4) If there is a deviation from the expected actions, what is the reason for that?

- For registering, a deviation from the expected action is if the (new) user attempts to register for an account with a username that already exists.
    - The reason for this is the (new) user just isn't aware that their desired username is already taken. They will need to select a different username to register for an account, as the site will not let them register with an existing username.
- For login, a deviation is if the user…
    - …enters a username that doesn't exist. In that case, they are redirected to the register page.
        - The reason for this is the user may be trying to register instead of logging in, or they may just incorrectly input their username by accident.
    - … enters the incorrect password. In that case, they are prompted with the message "incorrect password" and are redirected to the login page to try again.
        - The reason for this is the user just incorrectly typed in their password.
- For both register and login, if the user leaves at least one of the username and password fields blank, they will not be able to continue.

5) Did you use that to make changes to your application? If so, what changes did you make?

- We didn't make changes to this feature.

Deployment: This was the link to our deployed website: http://recitation-014-team-05.eastus.cloudapp.azure.com:3000/login

During office hours, Nathan said since we deployed but deleted the VM as instructed in lab 13, we can just include a screenshot of our once deployed website. *This screenshot is also in our milestoneSubmissions folder AND in our README:*