

Link to GitHub repository:  [fk721/finstagram](https://github.com/fk721/finstagram)

Date of last commit to repository: December 10, 2019

Number of Team members: 1

Names and netIDs of Team members (one per line): Fahmid Kamal — fk786

Feature 1

Name of the feature: Search by poster

Team member who implemented this feature: Fahmid Kamal

The SQL query:

If the poster that we are searching for is the user that is logged in (poster == user)

```
SELECT photoID, postingdate, photoBlob, caption, photoPoster
FROM Photo
WHERE photoPoster = POSTER
ORDER BY postingdate DESC
```

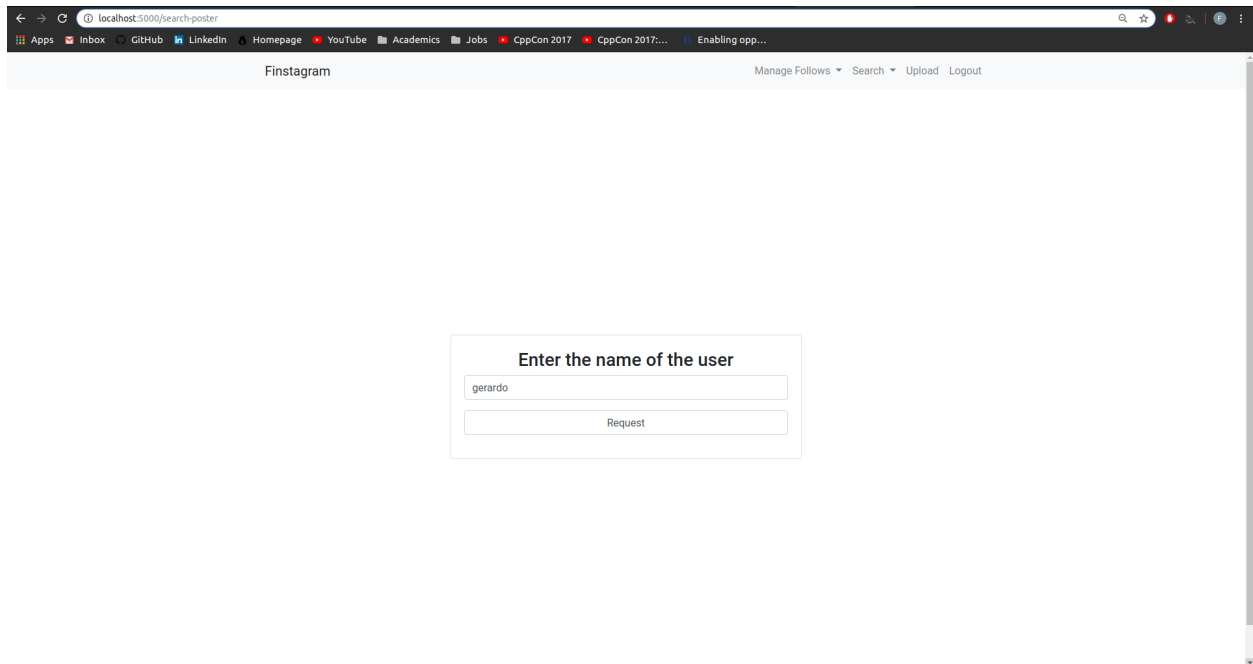
Else:

```
(SELECT P.photoID, P.postingdate, P.photoBlob, P.caption, P.photoPoster
FROM Photo P JOIN Follow F ON (P.photoPoster = F.username_followed)
WHERE followstatus = True AND allFollowers = True
AND username_follower = USER AND photoPoster = POSTER
UNION
(SELECT photoID, postingdate, photoBlob, caption, photoPoster
FROM Photo
WHERE photoPoster = POSTER AND photoID IN
(SELECT photoID
FROM BelongTo B JOIN SharedWith S ON
((B.owner_username = S.groupOwner) AND (B.groupName = S.groupName)
AND member_username = USER))
ORDER BY postingdate DESC
```

Where to find this feature: The html files associated with this feature are *search-poster.html* and *view-photos-by-poster.html*. The SQL queries can be found in *app.py*, in the function *searchPoster()* (lines 344 to 350) and the function *findSearchPoster()* (lines 352-392).

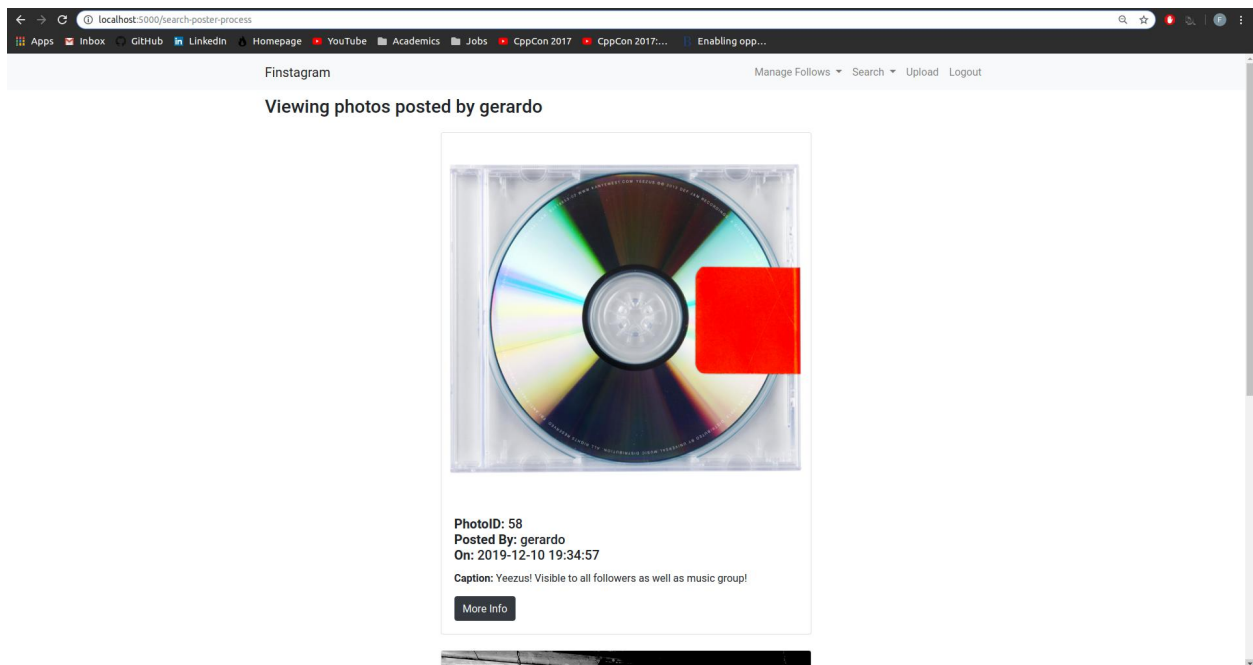
Screenshots: Note: This feature does not modify the database, therefore I will not be posting the before/after photos of the database since the database does not change.

The form:



The screenshot shows a web browser window with the address bar displaying 'localhost:5000/search-poster'. The browser's tab bar includes 'Apps', 'Inbox', 'GitHub', 'LinkedIn', 'Homepage', 'YouTube', 'Academics', 'Jobs', 'CppCon 2017', 'CppCon 2017:...', and 'Enabling app...'. The page header for 'Finstagram' includes links for 'Manage Follows', 'Search', 'Upload', and 'Logout'. The main content area features a form titled 'Enter the name of the user'. This form contains a text input field with the text 'gerardo' and a button labeled 'Request'.

The Result:



Feature 2

Name of the feature: Search by tag

Team member who implemented this feature: Fahmid Kamal

The SQL query:

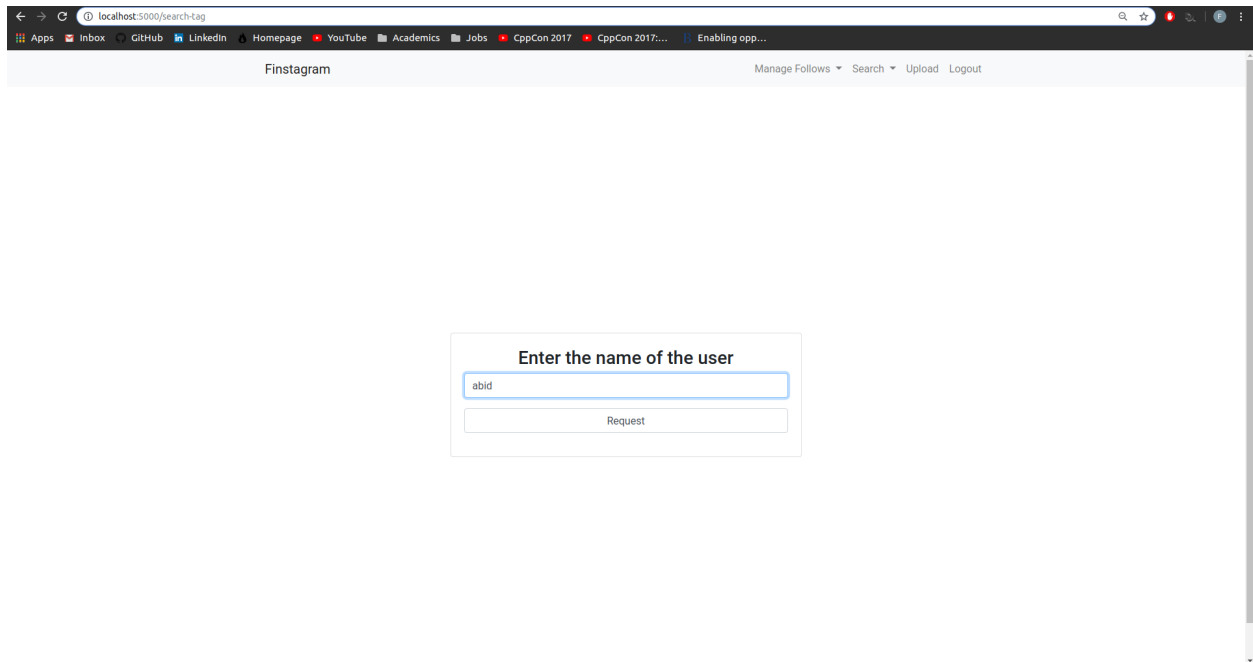
Given the user, and a tagged-user, we can use the following query:

```
SELECT * FROM
((SELECT P.photoID, P.postingdate, P.photoBlob, P.caption, P.photoPoster
FROM Photo P JOIN Follow F ON (P.photoPoster = F.username_followed)
WHERE followstatus = True AND allFollowers = True
AND username_follower = USER)
UNION
(SELECT photoID, postingdate, photoBlob, caption, photoPoster
FROM Photo
WHERE photoID IN
(SELECT photoID
FROM BelongTo B JOIN SharedWith S ON
((B.owner_username = S.groupOwner) AND (B.groupName = S.groupName))
AND member_username = USER))
UNION
(SELECT photoID, postingdate, photoBlob, caption, photoPoster
FROM Photo
WHERE photoPoster = USER)
ORDER BY postingdate DESC) AS T
WHERE T.photoID IN
(SELECT photoID
FROM Tagged
WHERE username = TAGGED-USER AND tagstatus = True)
```

Where to find this feature: The html files associated with this feature are *search-tag.html* and *view-photos-by-tag.html*. The SQL queries can be found in *app.py*, in the function *searchTag()* (lines 293 to 299) and the function *findSearchTags()* (lines 301-341).

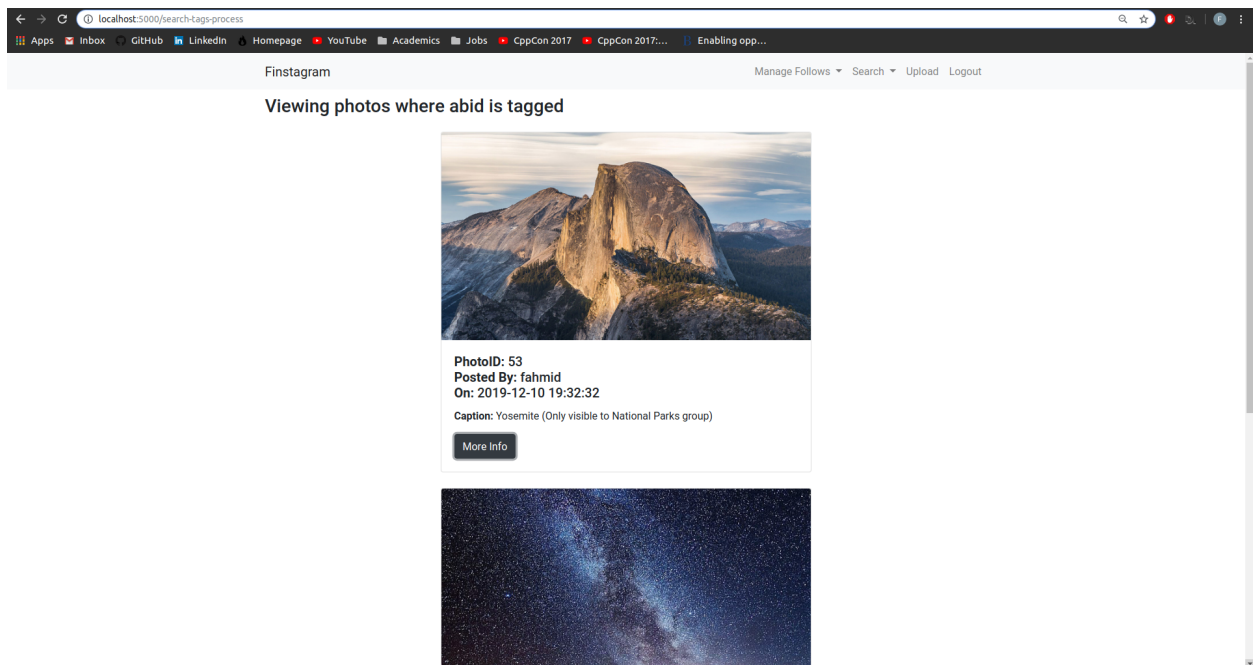
Screenshots: Note: This feature does not modify the database, therefore I will not be posting the before/after photos of the database since the database does not change.

The form:



The screenshot shows a web browser window with the address bar displaying 'localhost:5000/search-tag'. The browser's tab bar includes 'Apps', 'Inbox', 'GitHub', 'LinkedIn', 'Homepage', 'YouTube', 'Academics', 'Jobs', 'CppCon 2017', 'CppCon 2017...', and 'Enabling app...'. The page title is 'Finstagram'. The navigation bar contains 'Manage Follows', 'Search', 'Upload', and 'Logout'. The main content area features a form titled 'Enter the name of the user'. The form has a text input field containing the text 'abid' and a 'Request' button below it.

The Result:



Viewing additional photo information on PhotoID 53:

