Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT202-007-F2024/it202-api-project-milestone-3-2024-m24/grade/ob75

Course: IT202-007-F2024
Assigment: [IT202] API Project Milestone 3 2024 m24
Student: Omar B. (ob75)

Submissions:

Submission Selection

1 Submission [submitted] 12/11/2024 5:58:21 PM

Instructions

COLLAPSE

Overview Video: https://youtu.be/-4hlb9MXrQE

1. Implement the Milestone 3 features from the project's proposal document: https://docs.google.com/document/d/1XE96a8DQ52Vp49XACBDTNCq0xYDt3kF29cO88EWVwfo/viev
2. Make sure you add your ucid/date as code comments where code changes are done
3. All code changes should reach the Milestone3 branch
4. Create a pull request from Milestone3 to dev and keep it open until you get the output PDF from this assignment.
5. Gather the evidence of feature completion based on the below tasks.
6. Once finished, get the output PDF and copy/move it to your repository folder on your local machine.
7. Run the necessary git add, commit, and push steps to move it to GitHub
8. Complete the pull request that was opened earlier
9. Create and merge a pull request from dev to prod
10. Upload the same output PDF to Canvas

Branch name: Milestone3

Group

100%

Group: API
Tasks: 2
Points: 1

COLLAPSE

Task

100%

Group: API
Task #1: Data Related to Users
Weight: ~50%
Points: ~0.50

COLLAPSE

Checklist

*The checkboxes are for your own tracking

#	Details
#1	What's the concept/association?
#2	What sort of relationship is it (one to many, many to one, many to many, etc)
#3	Note any other considerations

Task Response Prompt

Response:

The association for my API to the users is that user would be able to add artist to their list of favorite artist and be able to unadd them if they no longer want that artist to be associated with them. The sort of relationship this is is that it is a many-to-many relationship since the data of many artist can be associated to many of the the other users and users can have the same favorited artist if the choose to favorite equivalent artist.

End of Task 1

Task

100%

Group: API
Task #2: Updating Entities
Weight: ~50%
Points: ~0.50

COLLAPSE

Checklist

*The checkboxes are for your own tracking

#	Details
#1	When an update occurs either manually or from the API how does it affect associated data?
#2	Do users see the old data, new data, does data need to be reassociated, etc?

Task Response Prompt

Response:

The associated data will be updated on the users favorited list as well if the artist name, weburl, or note is changed to signify to the user that the artist they favorited before no longer has the same data as previously. The users will see new data that is associated with them and will be updated rather than the update occurring and only showing the old data to the user. The user will most likely want to unfavorite the artist and add another artist if their artist data information has changed to an artist's information that they do not favorite.

End of Task 2

End of Group: API
Task Status: 2/2

Group

100%

Group: Handle Data Association
Tasks: 3
Points: 1

COLLAPSE

Task

100%

Group: Handle Data Association
Task #1: Screenshots of the code
Weight: ~33%
Points: ~0.33

COLLAPSE

Details:

Option 1: Related pages will have a button to do association (like favorites or similar),
Option 2: a separate page will be used to associate entities to a user by some other user (like assignment of entities)
Include ucid/date comments for each code screenshot

Sub-Task

100%

Group: Handle Data Association
Task #1: Screenshots of the code
Sub Task #1: Show the related code

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Showing related code for data association and edited search_artist.php file in line 28 - 35.

Added favorite "heart Icon" into form for ease of access to toggle between favoriting and removing favorite starting line 96.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works and mention which option your application handles regarding association

Response:

The option chosen to handle association was option 1 where the user has a favorite button that will add the artist to their favorites page and get that user associated with that artist. Firstly in the logic, an assoc_check variable was created to check for the association in the database between the logged in user and the artist. It would then assign a number 1 to is_watched if the user had that artist favorited and if not it would be assigned 0 so that its known when the user has an artist favorited. The database is then called to grab all the data and to store it into the sql variable. The match of id of 1=1 is also stored inside the where variable. This then later adds this information to the database with the new table created that stores the relationship between the user and artist named `UserArtists` and they now have that relationship.

End of Task 1

Task

100%

Group: Handle Data Association
Task #2: Screenshot of the association table(s)
Weight: ~33%
Points: ~0.33

COLLAPSE

Sub-Task

100%

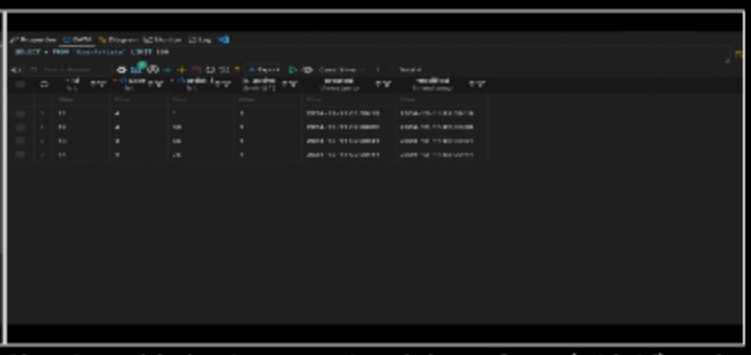
Group: Handle Data Association
Task #2: Screenshot of the association table(s)
Sub Task #1: Show the table(s) you made to handle the associations (Should have some example data)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Showing creation of table named UserArtists.



Showing table having associated data of user(with id) and the artist id.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Describe each column/association table

Response:

The id column is the id column meant for the table used for grabbing the entire rows information. The user_id column is the users id that is being related to the artist_id in the table and are placed to mark which user has a relationship with which artist. is_active column is to show whether the user is active on the page and has an account within the site. The created and modified columns are there to show the either the creation of the relation or when it was modified.

End of Task 2

Task

100%

Group: Handle Data Association
Task #3: Add related links
Weight: ~33%
Points: ~0.33

COLLAPSE

Checklist

*The checkboxes are for your own tracking

#	Details
#1	Include the heroku prod link for the page that creates the association
#2	Add the pull request link for the branch related to this feature Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

Task URLs

URL #1 https://it202-ob75-prod-62037cdcca6c.herokuapp.com/project/search_artists.php

https://it202-ob75-prod-62037cdcca6c.herokuapp.com/

URL #2 https://github.com/OmarBarrera1/ob75-IT202-007/pull/62

https://github.com/OmarBarrera1/ob75-IT202-007

End of Task 3

End of Group: Handle Data Association
Task Status: 3/3

Group

100%

Group: Current User's Association Page
Tasks: 3
Points: 2

COLLAPSE

Task

100%

Group: Current User's Association Page
Task #1: Screenshots of this page
Weight: ~33%
Points: ~0.67

COLLAPSE

Details:

Make sure the heroku dev url is visible in the address bar

Some screenshots may be repeated in subtasks, but ensure they highlight the specific subtask requirement.

Columns: 1

Sub-Task

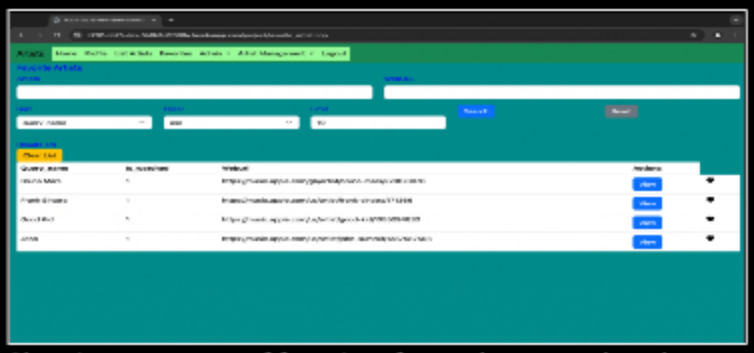
100%

Group: Current User's Association Page
Task #1: Screenshots of this page
Sub Task #1: Show the summary of the results with relevant information per entity

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Showing summary of favorites for each user and each entity.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

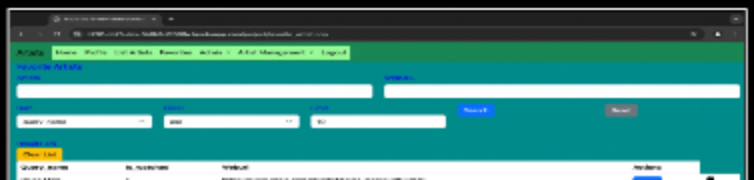
100%

Group: Current User's Association Page
Task #1: Screenshots of this page
Sub Task #2: Show the single view buttons/links, delete button/links, and delete all button/link

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Showing the clear button as Clear List, view button and the delete button is the "heart" Icon to unfavorite.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

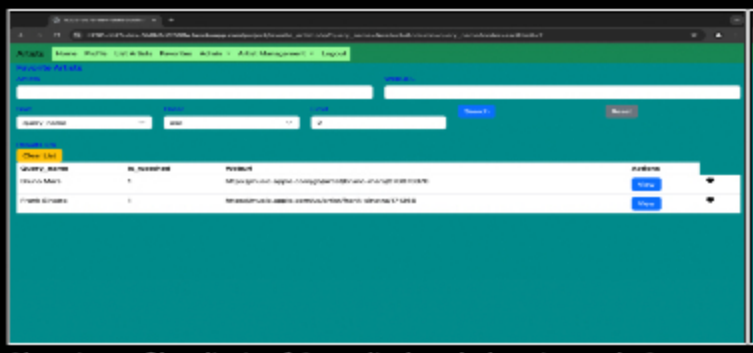
100%

Group: Current User's Association Page
Task #1: Screenshots of this page
Sub Task #3: Show variations of the number of shown items count and show the count of total number of associated items to the user

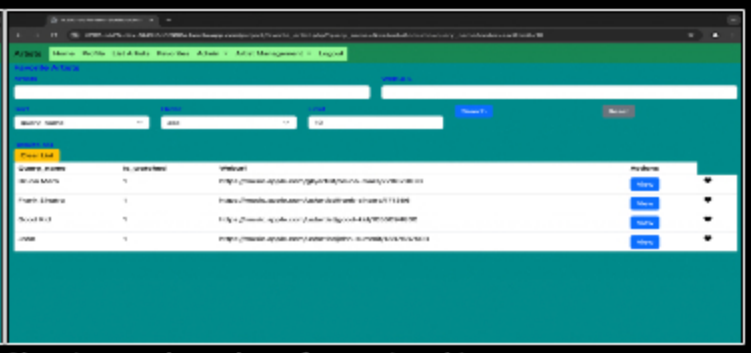
Task Screenshots

Gallery Style: 2 Columns

4 2 1



Showing a filter limit of 2 applied and showing only 2 results out of 4.



Showing total number of associated items to user as "Results"

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

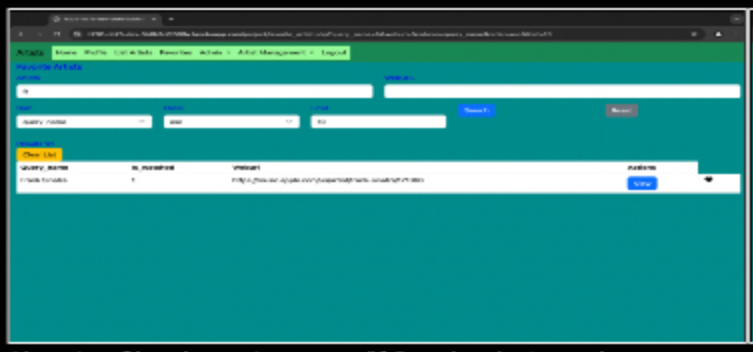
100%

Group: Current User's Association Page
Task #1: Screenshots of this page
Sub Task #4: Show variations of the filter/sort including no results (proper message should be visible)

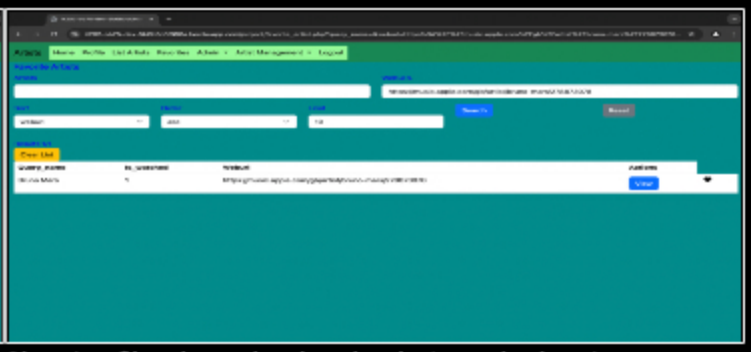
Task Screenshots

Gallery Style: 2 Columns

4 2 1

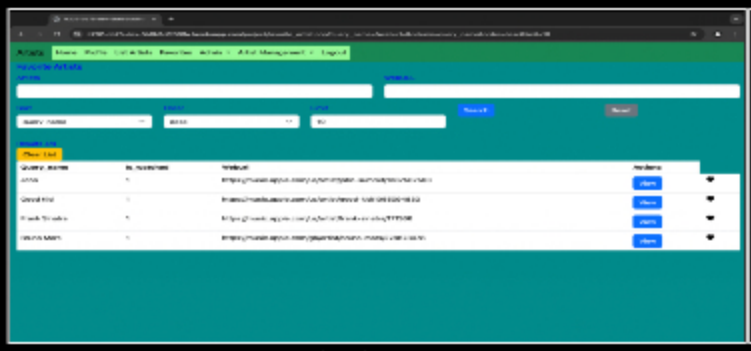


Showing filter by artist name "fr" and only 1 result

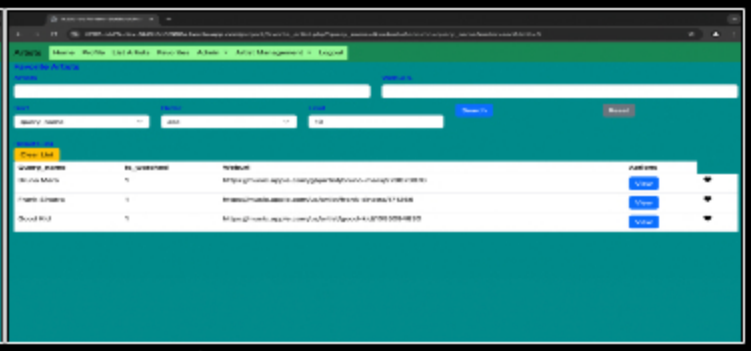


Showing filter by web url and only 1 result showing.

appearing.



Showing desc order sort for favorites.



showing limit of 3 and 3/4 results shown.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

End of Task 1

Task

100%

Group: Current User's Association Page

Task #2: Screenshot the code

Weight: ~33%

Points: ~0.67

COLLAPSE

Details:

Include ucid/date comments for each code screenshot

Columns: 1

Sub-Task

100%

Group: Current User's Association Page

Task #2: Screenshot the code

Sub Task #1: Show the code related to fetching the user's associations (including the query)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Code showing fetching the user_id to grab the user

association all below the UCID line.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works and mention how you determine the result list (include the association logic and filters)

Response:

The association check is still created here to check what is being watched by the user and is included in the query to make sure that the user has the relationship with the artist as favorited. User needs to be logged in to have the association and we get the user id of the one logged in. Then we join the query of the artists table onto the UserArtist table grabbing all those that match the id of the user and the artist id. We then store the where clause of where the user id of UserArtists is = to user_id and add that to the where variable. The results later on is determined using the where variable we created and we use it in the query to grab the assocaition id.

Sub-Task

100%

Group: Current User's Association Page

Task #2: Screenshot the code

Sub Task #2: Show the code related to the display of the results

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Showing code related to display of results Line 157-161

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

The code here shows the display of the results calculated from counting the query results of the associated entities with the user and separated the counted results of the page by a "/" and then show total results.

Sub-Task

100%

Group: Current User's Association Page

Task #2: Screenshot the code

Sub Task #3: Each record should have a button/link for single view

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Code showing Line 113 - 118 shows button link to single view.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

The same view link that is used for the list page is just copied and pasted to use on the table here for users to access the view link to get a better description of the entity. The view is placed onto the page as a button to be pressed on and is generated alongside the results of the favorited.

Sub-Task

100%

Group: Current User's Association Page

Task #2: Screenshot the code

Sub Task #4: Each record should have a button/link for delete (this may be an admin-only thing but should be present for the specific role) Note: this is to delete the relationship and not the specific entity

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Code showing the toggle for adding or removing an entity to association but in this case will just remove in the favorites page.

Code showing for the "toggleWatched to be called and delete line 27 - 36.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

The link to delete here is done in the same way as associating the artist to the user in which there is a "heart" icon to unfavorite the artist and essentially delete them from the page but not delete the entity, just relationship. This is added onto render_table when the render table is called and it then calls the toggleWatched file which insert and deletes the relationships between artist and users. The delete here is the more important part of the page since once deleted or unfavorited, the user will need to go back to the list page of artists and add them again. The delete query here deletes the relationship where the artist_id is matched and where the user_id is mathced to properly remove the correct relationship and not all. A message is then shown to the user if successful and if not it shows an error message.

Sub-Task

100%

Group: Current User's Association Page

Task #2: Screenshot the code

Sub Task #5: Show the logic for deleting all associations for the user (this may be admin-only but should be present for the specific role)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Code showing clearWatched file clearing all the user's associations to the user.

Clear_watched file being set to a Clear List button to clear.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

The clearWatched file gets the user_id and only allows for admin users to be able to access this button. We open the database query and delete all records from the UserArtists table since its not specified where the user_id matches that of the one clicking the button. The list becomes cleared of the records and the user is met with a message that the list was cleared. If not, they are met with a friendly error message. This function is called in the button Clear List and is set in the favorite_aritsts file in the form to display to the user.

Sub-Task

Group: Current User's Association Page

Task #2: Screenshot the code

100%

Sub Task #6: Show the logic related to the count of all associated items to the user (even the ones not shown in the filtered results)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

```
//UCID - ob72 - 12/12/2024
$total = 0;
$sql = "SELECT COUNT(DISTINCT sa.id) AS c FROM `Shazam-Artists` AS sa
JOIN `UserArtists` ua ON ua.artist_id = sa.id
where";

try {
    $db = getDB();
    $stmt = $db->prepare($sql);
    if (isset($params["user_id"])) {
        unset($params["user_id"]);
    }
    //
    $stmt->execute($params);
    $r = $stmt->fetch();
    if ($r) {
        $total = (int)$r["c"]; // cast to int in case it's a string
    }
} catch (PDOException $e) {
    flush("Error fetching count", "danger");
    error_log("Error fetching count " . var_export($e, true));
    error_log("Query: $sql");
    error_log("Params: " . var_export($params, true));
} // else is safe to continue
```

Code showing counting all the user associated items.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

We count the artist in the table Shazam-Artists and join that onto the UserArtists based on their artist id to be able to count how many of the artist are matched and are favorited by the user. The total is defaulted at 0 but get appended to if the user contains associations in their table and counts up. The query is executed and called onto the r variable that fetches the results. If there are results, we change total to be the total number of new results.

Sub-Task

100%

Group: Current User's Association Page

Task #2: Screenshot the code

Sub Task #7: Show the logic related to the count of the items on the page (this value should change based on the filter applied)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

```
    <div class="row">
        <div class "col">
            Results <?php echo count($results) . "/" . $total; ?>
        </div>
    </div>
    <!-- UCID - ob72 - 12/12/2024 -->
```

Code showing counting the result on page and displaying it to the user.

Caption(s) (required)

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

This counts the total number of results displayed on the page and the total number of associations to the user based on the sorting/filter that is done to the page. The left side represents the count that is displayed and the right represents the total number of associations.

Sub-Task

100%

Group: Current User's Association Page

Task #2: Screenshot the code

Sub Task #8: Show the logic related to filter/sort (limit should be constrained to 1-100 otherwise default to 10)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Showing the filter/sort logic and limit being defaulted to 10 when not in range 1-100.

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

The same limit done previously in the list and sorting page is created here to be able to carry over the limit on how many results should appear. The limit is changed based on the int number that the user enters into the limit field and is only within the range of 1-100. If not in that range, then the limit is defaulted to 10 for the field. The sql variable then appends the limit to the query before execution to display the proper amount of results the user is requesting to see and if not changed, it will just show 10.

End of Task 2

Task

100%

Group: Current User's Association Page

Task #3: Add related links

Weight: ~33%

Points: ~0.67

COLLAPSE

Checklist

*The checkboxes are for your own tracking

#	Details
#1	Include the heroku prod link for the page that creates the association
#2	Add the pull request link for the branch related to this feature Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

Task URLs

URL #1 https://it202-ob75-prod-62037cdcca6c.herokuapp.com/project/favorite_artist.php

URL #2 https://github.com/OmarBarrera1/ob75-IT202-007/pull/63

End of Task 3

End of Group: Current User's Association Page
Task Status: 3/3

Group

0%

Group: All Users Association Page (likely an admin page)
Tasks: 3
Points: 2

COLLAPSE

Task

0%

Group: All Users Association Page (likely an admin page)
Task #1: Screenshots of this page
Weight: ~33%
Points: ~0.67

COLLAPSE

Details:
Make sure the heroku dev url is visible in the address bar

Some screenshots may be repeated in subtasks, but ensure they highlight the specific subtask requirement.

Columns: 1

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #1: Screenshots of this page
Sub Task #1: Show the summary of the results with relevant information per entity

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Sub-Task

0%

Group: All Users Association Page (likely an admin page)

Task #1: Screenshots of this page

Sub Task #2: Show the single view buttons/links, delete button/links

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Sub-Task

0%

Group: All Users Association Page (likely an admin page)

Task #1: Screenshots of this page

Sub Task #3: Show the username related to the specific entity and that it's clickable

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #1: Screenshots of this page
Sub Task #4: Show variations of the number of shown items count and show the count of total number of associated items

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #1: Screenshots of this page
Sub Task #5: Show variations of the filter/sort including no results (proper message should be visible)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

End of Task 1

Task

0%

Group: All Users Association Page (likely an admin page)
Task #2: Screenshot the code
Weight: ~33%
Points: ~0.67

COLLAPSE

Details:

Include ucid/date comments for each code screenshot

Columns: 1

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #2: Screenshot the code
Sub Task #1: Show the code related to fetching all associations (including the query)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works and mention how you determine the result list (include the association logic and filters)

Response:

Missing Response

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #2: Screenshot the code
Sub Task #2: Show the code related to the display of the results

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works and mention the logic for handling the username requirements

Response:

Missing Response

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #2: Screenshot the code
Sub Task #3: Each record should have a button/link for single view

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #2: Screenshot the code
Sub Task #4: Each record should have a username field that is clickable to go to the user's profile page

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #2: Screenshot the code
Sub Task #5: Each record should have a button/link for delete (this may be an admin-only thing but should be present for the specific role) Note: this is to delete the relationship and not the specific entity

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #2: Screenshot the code
Sub Task #6: Show the logic related to the count of all associated items (even the ones not shown in the filtered results)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #2: Screenshot the code
Sub Task #7: Show the logic related to the count of the items on the page (this value should change based on the filter applied)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: All Users Association Page (likely an admin page)
Task #2: Screenshot the code
Sub Task #8: Show the logic related to filter/sort (should include a partial match for username) (limit should be constrained to 1-100 otherwise default to 10)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

End of Task 2

Task

0%

Group: All Users Association Page (likely an admin page)
Task #3: Add related links
Weight: ~33%
Points: ~0.67

COLLAPSE

Checklist

*The checkboxes are for your own tracking

#	Details
☐ #1	Include the heroku prod link for the page that creates the association
☐ #2	Add the pull request link for the branch related to this feature Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

Task URLs

URL #1
Missing URL

URL

End of Task 3

End of Group: All Users Association Page (likely an admin page)
Task Status: 0/3

Group

0%

Group: Unassociated Page
Tasks: 3
Points: 2

COLLAPSE

Task

0%

Group: Unassociated Page
Task #1: Screenshots of this page
Weight: ~33%
Points: ~0.67

COLLAPSE

Details:
Make sure the heroku dev url is visible in the address bar

Some screenshots may be repeated in subtasks, but ensure they highlight the specific subtask requirement.

Columns: 1

Sub-Task

0%

Group: Unassociated Page
Task #1: Screenshots of this page
Sub Task #1: Show the summary of the results with relevant information per entity

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)
Caption Hint: *Describe/highlight what's being shown*
Missing caption(s)

Sub-Task

0%

Group: Unassociated Page
Task #1: Screenshots of this page
Sub Task #2: Show the single view buttons/links

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)
Caption Hint: *Describe/highlight what's being shown*
Missing caption(s)

Sub-Task

0%

Group: Unassociated Page
Task #1: Screenshots of this page
Sub Task #3: Show variations of the number of shown items count and show the count of total number of associated items

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Sub-Task

0%

Group: Unassociated Page

Task #1: Screenshots of this page

Sub Task #4: Show variations of the filter/sort including no results (proper message should be visible)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

End of Task 1

Task

0%

Group: Unassociated Page

Task #2: Screenshot the code

Weight: ~33%

Points: ~0.67

COLLAPSE

Details:
Include ucid/date comments for each code screenshot

Columns: 1

Sub-Task
0%
Group: Unassociated Page
Task #2: Screenshot the code
Sub Task #1: Show the code related to fetching all unassociated entities (including the query)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)
Caption Hint: *Describe/highlight what's being shown*
Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works and mention how you determine the result list (include the unassociated logic and filters)
Response:
Missing Response

Sub-Task
0%
Group: Unassociated Page
Task #2: Screenshot the code
Sub Task #2: Show the code related to the display of the results

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: Unassociated Page

Task #2: Screenshot the code

Sub Task #3: Each record should have a button/link for single view

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: Unassociated Page

Task #2: Screenshot the code

Sub Task #4: Show the logic related to the count of all unassociated items (even the ones not shown in the filtered results)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: Unassociated Page

Task #2: Screenshot the code

Sub Task #5: Show the logic related to the count of the items on the page (this value should change based on the filter applied)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: Unassociated Page

Task #2: Screenshot the code

Sub Task #6: Show the logic related to filter/sort (should include a partial match for username)

(limit should be constrained to 1-100 otherwise default to 10)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

End of Task 2

Task

0%

Group: Unassociated Page
Task #3: Add related links
Weight: ~33%
Points: ~0.67

COLLAPSE

Checklist

*The checkboxes are for your own tracking

#	Details
☐ #1	Include the heroku prod link for the page that creates the association
☐ #2	Add the pull request link for the branch related to this feature Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

Task URLs

URL #1

Missing URL

URL

End of Task 3

End of Group: Unassociated Page
Task Status: 0/3

Group

0%

Group: Admin Association Management (like UserRoles)
Tasks: 3
Points: 1

COLLAPSE

Task

0%

Group: Admin Association Management (like UserRoles)
Task #1: Screenshots of the page
Weight: ~33%
Points: ~0.33

COLLAPSE

Details:
Make sure the heroku dev url is visible in the address bar

Some screenshots may be repeated in subtasks, but ensure they highlight the specific subtask requirement.

Columns: 1

Sub-Task

0%

Group: Admin Association Management (like UserRoles)
Task #1: Screenshots of the page
Sub Task #1: Show the search form with valid data

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)
Caption Hint: *Describe/highlight what's being shown*
Missing caption(s)

Sub-Task

0%

Group: Admin Association Management (like UserRoles)
Task #1: Screenshots of the page
Sub Task #2: Show the results of the search

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Sub-Task

0%

Group: Admin Association Management (like UserRoles)

Task #1: Screenshots of the page

Sub Task #3: Show the result of entities and users being associated and unassociated

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

End of Task 1

Task

0%

Group: Admin Association Management (like UserRoles)

Task #2: Screenshots of the code

Weight: ~33%

Points: ~0.33

COLLAPSE

Details:

Include ucid/date comments for each code screenshot

Columns: 1

Sub-Task

0%

Group: Admin Association Management (like UserRoles)
Task #2: Screenshots of the code
Sub Task #1: Search form field for finding a partial match of usernames

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: Admin Association Management (like UserRoles)
Task #2: Screenshots of the code
Sub Task #2: Search form field for finding a partial match of entities

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: Admin Association Management (like UserRoles)
Task #2: Screenshots of the code
Sub Task #3: Code related to getting a max of 25 results for each field (i.e., 25 users and 25 entities limit)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works and describe the steps for the search and how it works for users and entities

Response:

Missing Response

Sub-Task

0%

Group: Admin Association Management (like UserRoles)
Task #2: Screenshots of the code
Sub Task #4: Code that generates the checkboxes next to each list (users and entities)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

0%

Group: Admin Association Management (like UserRoles)
Task #2: Screenshots of the code
Sub Task #5: Code related to submitting the checkbox lists

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works

Response:

Missing Response

Sub-Task

Group: Admin Association Management (like UserRoles)
Task #2: Screenshots of the code

0%

Task #2: Screenshots of the code

Sub Task #6: Code related to applying the associations upon submission (i.e., add the relationship if it doesn't exist and remove the relationship if it does exist)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Missing Caption

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Missing caption(s)

Task Response Prompt

Explain in concise steps how this logically works and describe the steps for the associate/unassociate logic for the combination of users and entities

Response:

Missing Response

End of Task 2

Task

0%

Group: Admin Association Management (like UserRoles)

Task #3: Add related links

Weight: ~33%

Points: ~0.33

COLLAPSE

Checklist

*The checkboxes are for your own tracking

#	Details
☐ #1	Include the heroku prod link for the page that creates the association
☐ #2	Add the pull request link for the branch related to this feature Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

Task URLs

URL #1

Missing URL

URL

End of Task 3

End of Group: Admin Association Management (like UserRoles)
Task Status: 0/3

Group

100%

Group: Misc
Tasks: 4
Points: 1

COLLAPSE

Task

100%

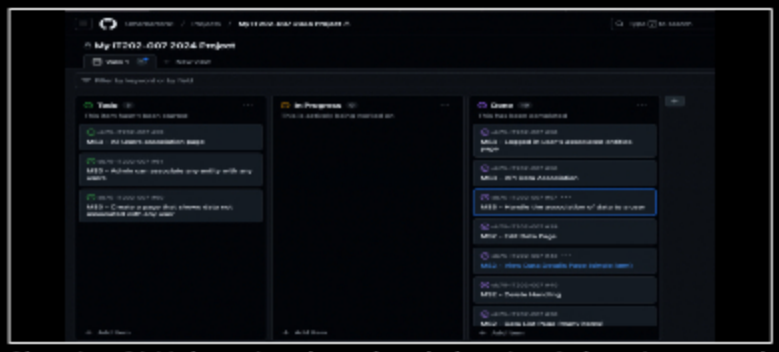
Group: Misc
Task #1: Screenshot of your project board from GitHub (tasks should be in the proper column)
Weight: ~25%
Points: ~0.25

COLLAPSE

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Showing GitHub project board and showing 3 done items and 3 items not started/done in Milestone3

End of Task 1

Task

100%

Group: Misc
Task #2: Provide a direct link to the project board on GitHub
Weight: ~25%
Points: ~0.25

COLLAPSE

Task URLs

URL #1 https://github.com/users/OmarBarrera1/projects/
https://github.com/users/OmarBarrera1/projects/1/views/1?layout=board

End of Task 2

Task

100%

Group: Misc
Task #3: Talk about any issues or learnings during this assignment
Weight: ~25%
Points: ~0.25

COLLAPSE

Task Response Prompt

Response:

Some important issues to not is that I was not able to complete the rest of the Milestone3 features in the time alloted and have left them blank. Issues that came with the few features completed were figuring a way on how to toggle on and off the favorites for each user and how they can add their own associations to the artist without having the admin add the association for them. The "heart" icon along with toggle_watched file were added here to be able to toggle that association and allowed for favorites of artist to be inserted into the database table that contained all the relationships to eahc other. Another issue was the delete button in deleting only the relationship rather than clearing the entire entity. This came from trying to implement the delete button with the delete_url but that was incorrect since it deleted the entity from the database. Instead the toggle_watched was implemented again here, but this time, evertime the user unfavorited the artist, they would no longer see it on their favorites page and it would be updated to show that the artist is no longer favorited, so the user would need to go back into the list artist page to favorite them again.

End of Task 3

Task

100%

Group: Misc
Task #4: WakaTime Screenshot
Weight: ~25%
Points: ~0.25

COLLAPSE

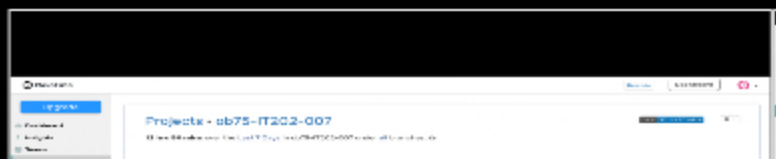
Details:

Grab a snippet showing the approximate time involved that clearly shows your repository. The duration isn't considered for grading, but there should be some time involved

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Showing top has of wakatime

Showing Bottom half of wakatime

Showing rest of wakatime

End of Task 4

End of Group: Misc
Task Status: 4/4

End of Assignment