Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT202-452-M2024/it202-api-project-milestone-3-2024-m24/grade/dr475

IT202-452-M2024 - [IT202] API Project Milestone 3 2024 m24

Submissions:

Submission Selection

1 Submission [active] 7/29/2024 6:36:13 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

Tasks: 21 Points: 10.00

API (1 pt.)

COLLAPSE

COLLAPSE

Task #1 - Points: 1

Text: Data Related to Users

Checklist *The checkboxes are for your own tracking

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

Response:

I decided to create a User_Favorites table that associates user IDs to trails that are marked as favorite by the user. The relationship is a many to many. Users can have many favorite trails as their favorite, and trails can be favorited by many different users.

COLLAPSE

Task #2 - Points: 1

Text: Updating Entities

Checklist *The checkboxes are for your own tracking

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

Response:

When an update occurs from the API or manually, it does not affect associations at all. Since the user's favorites are associated to a specific trail ID and the trail ID is never modified unless fully deleted, it does not affect the Users_Favorites association table. If by chance a trail IS deleted, the user's favorites association is subsequently deleted. Users always view the most up to date data regarding a trail from the Trail table.

Handle Data Association (1 pt.)

COLLAPSE

Task #1 - Points: 1

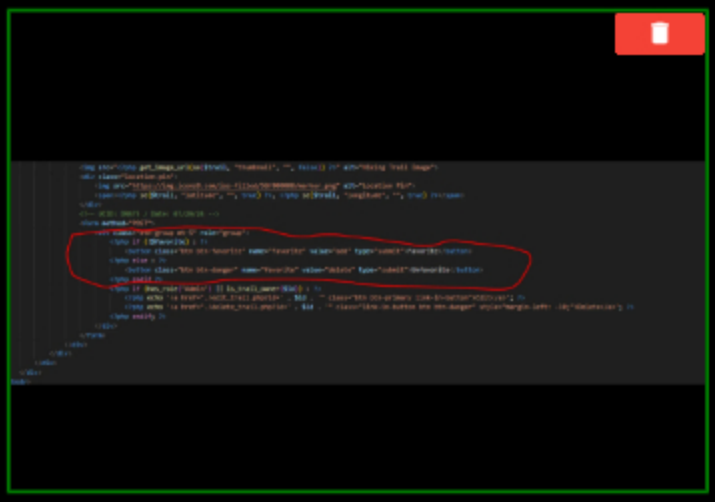
Text: Screenshots of the code

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

#1) Show the related code



```php
//
// 20475
// 07/25/24
function add_favorite_by_trail_id($user_id, $trail_id){
    $db = getDB();
    $stmt = $db->prepare("INSERT INTO `User_Favorites` (user_id, trail_id) VALUES (:user_id, :trail_id)");
    try {
        $stmt->execute([":user_id" => $user_id, ":trail_id" => $trail_id]);
        return true;
    } catch (Exception $e){
        return false;
    }
}

function delete_favorite_by_trail_id($user_id, $trail_id){
    $db = getDB();
    $stmt = $db->prepare("DELETE FROM `User_Favorites` WHERE user_id=:user_id AND trail_id=:trail_id");
    try {
        $stmt->execute([":user_id" => $user_id, ":trail_id" => $trail_id]);
        return true;
    } catch (Exception $e){
        return false;
    }
}
```

Caption (required) ✓

Describe/highlight what's being shown

Server Side code that handles associating favorite trails to users

Explanation (required) ✓

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

PREVIEW RESPONSE

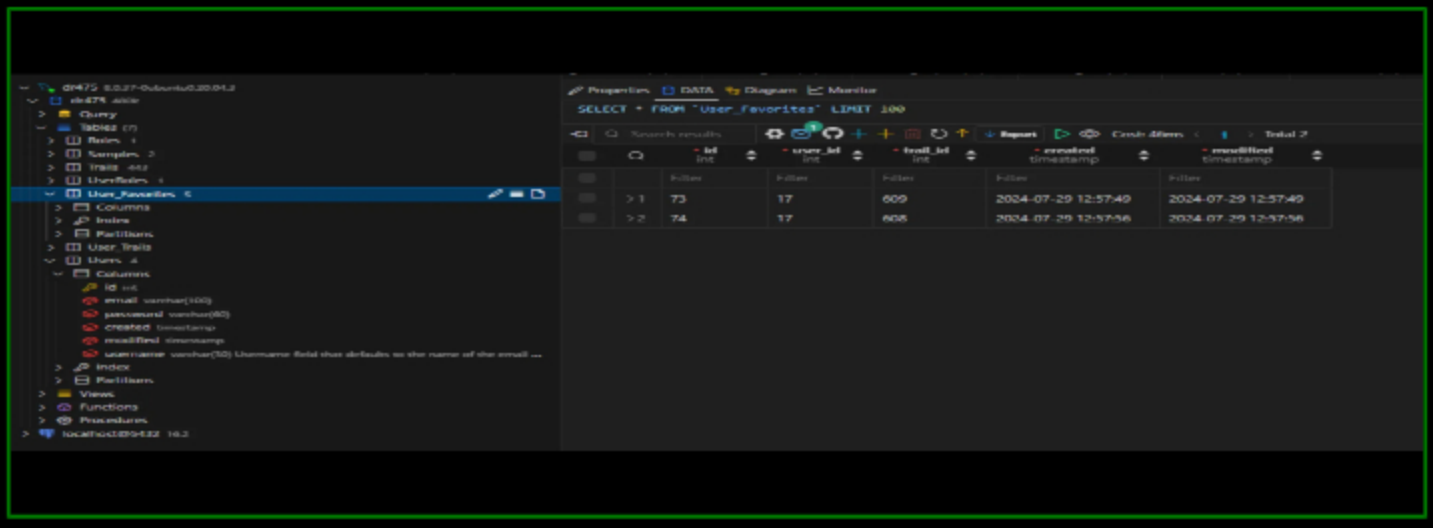
I chose Option 1. Upon loading the details page for a trail, the Favorite or Unfavorite button will be shown depending on whether or not an association is found in the database. If the user has it favorited, the user can press the unfavorite button, which deletes the record. If the trail is not yet favorited, the favorite button that can be pressed will insert a new record that stores the trail ID and user ID to associate the trail as a user's favorite.

COLLAPSE

Task #2 - Points: 1

Text: Screenshot of the association table(s)

#1) Show the table(s) you made to handle the associations (Should have some example data)



Caption (required) ✓

Describe/highlight what's being shown

User_Favorites Table

Explanation (required) ✓

Describe each column/association table

PREVIEW RESPONSE

id - The ID is the unique identifier of the table.
user_id - foreign key that references users.id
trail_id - foreign key that references trails.id

COLLAPSE

Task #3 - Points: 1

Text: Add related links

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Include the heroku prod link for the page that creates the association
#2	1	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

URL #1

https://it202-dr475-prod-7559be2e2ad4.herokuapp.com/project/trail.php?id=127

url

https://it202-dr475-prod-7559be2e2ad4.herokua

URL #2

url

https://github.com/davidredziniak/dr475-IT202-452-M2024/pull/79

URL #3
https://github.com/davidredziniak/dr475-IT202-452-M2024/pull/80

URL
https://github.com/davidredziniak/dr475-IT202-4

+ ADD ANOTHER URL

Current User's Association Page (2 pts.)

^COLLAPSE ^

^COLLAPSE ^

Task #1 - Points: 1

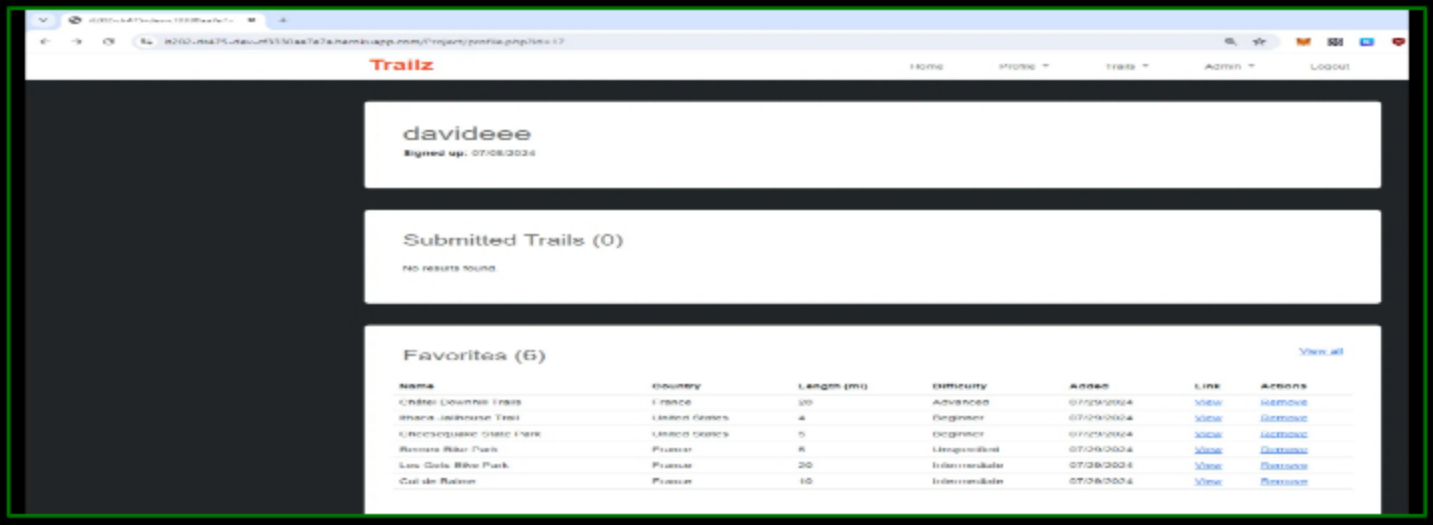
Text: Screenshots of this page

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.

#1) Show the summary of the results with relevant information per entity

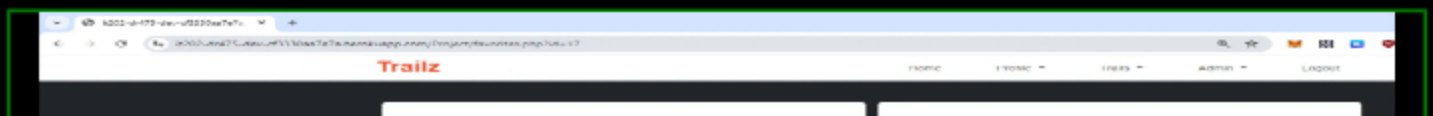


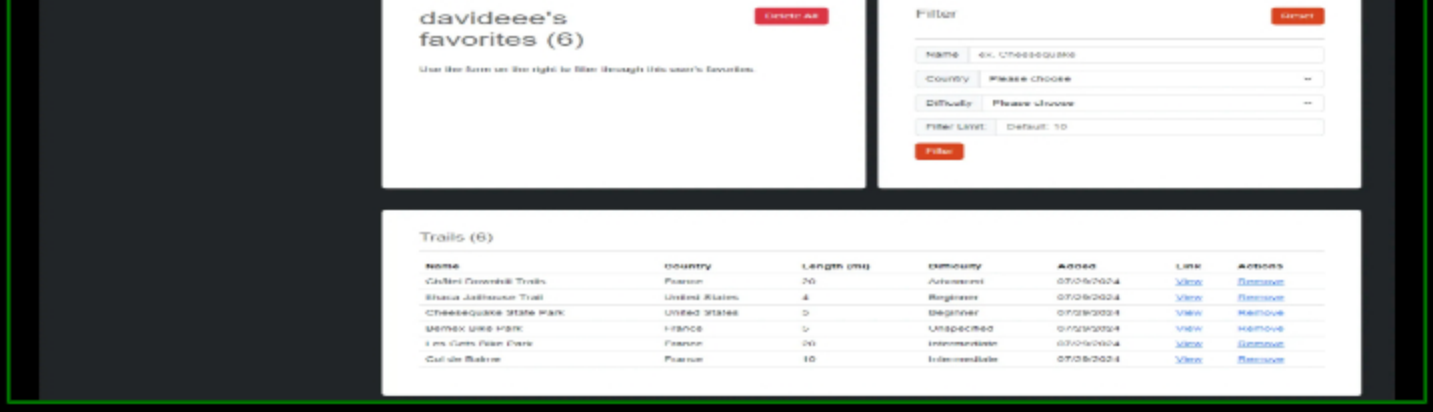
Caption (required) ✓

Describe/highlight what's being shown

User profile page

#2) Show the single view buttons/links, delete button/links, and delete all button/link



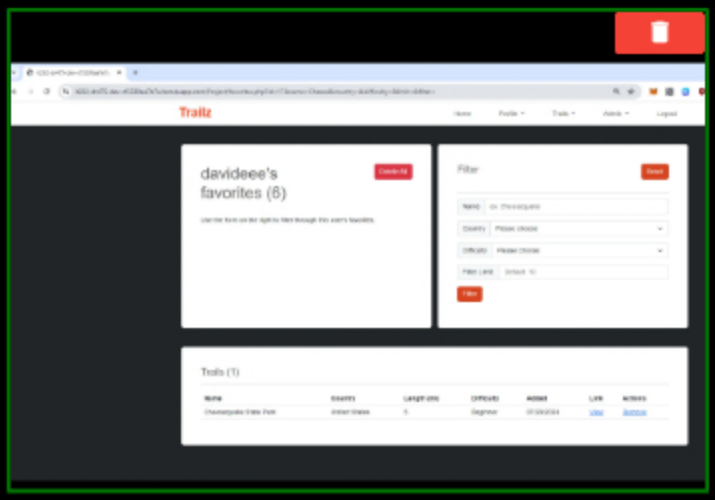


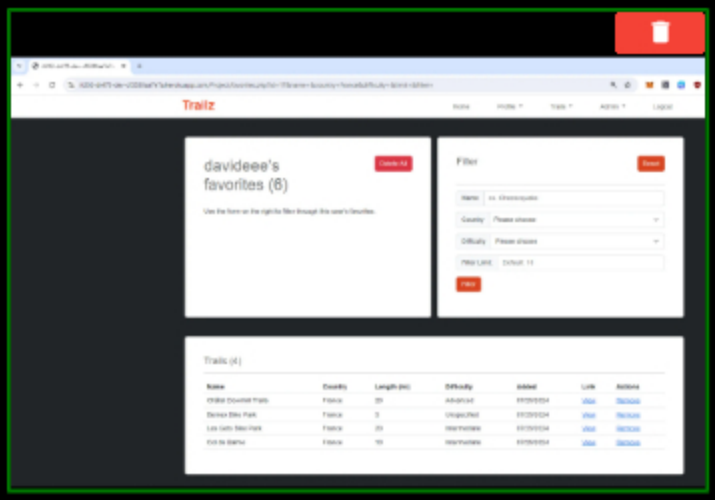
Caption (required) ✓

Describe/highlight what's being shown

Full detailed list of a user's favorites page.

#3) Show variations of the number of shown items count and show the count of total number of associated items to the user



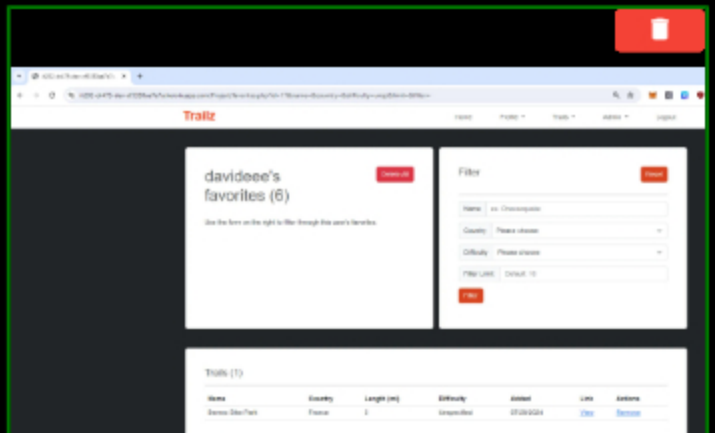


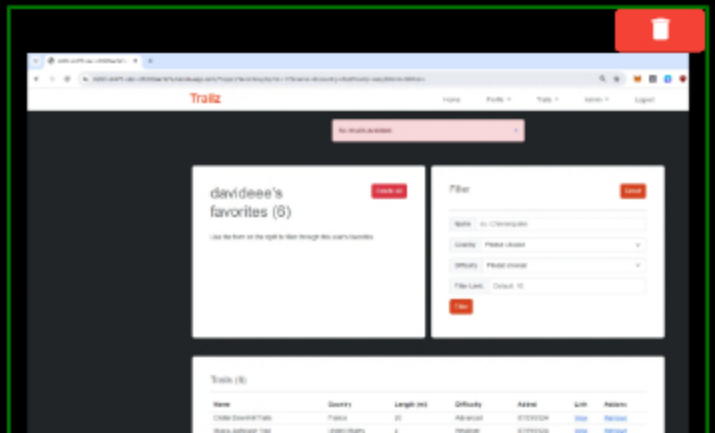
Caption (required) ✓

Describe/highlight what's being shown

Image 1: Searching trail name "Cheese", Image 2: Searching country (France)

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





Caption (required) ✓
Describe/highlight what's being shown
Image 1: Searching for difficulty (Unspecified), Image 2: No results found for difficulty (Easiest)

COLLAPSE

Task #2 - Points: 1

Text: Screenshot the code

Details:
Include ucid/date comments for each code screenshot

#1) Show the code related to fetching the user's associations

Caption (required) ✓
Describe/highlight what's being shown
Code for searching through user's favorites

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

PREVIEW RESPONSE

Upon loading the favorites page for a specified user, all their favorites are queried. There is then a form that the person viewing can use to specify and searching

#2) Show the code related to the display of the results

Caption (required) ✓
Describe/highlight what's being shown
Code for displaying results in a table

Explanation (required) ✓
Explain in concise steps how this logically works

PREVIEW RESPONSE

The filtered/unfiltered results is shown on the page as a table, listing the Trail name, country, length, difficulty, when it was added to the website, and a link to view the favorited trail. When the user viewing the favorites page filters/searches a user's favorites, the $results array gets updated with the found trails, which then

#3) Each record should have a button/link for single

Caption (required) ✓
Describe/highlight what's being shown
The code for the single view of the trail

Explanation (required) ✓
Explain in concise steps how this logically works

PREVIEW RESPONSE

Since a trail is considered a favorite, there is a link to view the favorite trail in the table when it is being displayed.

which favorites the user has (name of the trail, difficulty, and country). If the user is filtering the favorites, the server side builds a query based on what the user is searching for, and selects trail information WHERE the trail.id and uf.id exists in the Users_favorites table, ensuring that the trail is favorited.

conditionally renders the table.

#4) Each record should have a button/link for delete

Caption (required) ✓
Describe/highlight what's being shown
Code for deleting a favorite association

Explanation (required) ✓
Explain in concise steps how this logically works

PREVIEW RESPONSE

The link to delete the favorite association is displayed for both the user the favorites page is referring to, as well as admins. When the delete_favorite.php is accessed with the ID of the favorite in the URL parameters, it calls the delete_favorite_by_id() function in favorite_helpers.php that will delete the favorite association in the database.

#5) Show the logic for deleting all associations for the

Caption (required) ✓
Describe/highlight what's being shown
Code for deleting all favorites

Explanation (required) ✓
Explain in concise steps how this logically works

PREVIEW RESPONSE

There is a button that is shown for admins/user that owns the profile on the favorites.php page. This button submits a deleteall query to the server that then calls a delete_all_favorites() function which then deletes all records in the User_Favorites table that has the user_id=(id of the user).

#6) Show the logic related to the count of all associated items to

Caption (required) ✓
Describe/highlight what's being shown
Code that retrieves the count of all the user's favorites

Explanation (required) ✓
Explain in concise steps how this logically works

PREVIEW RESPONSE

By default, all the user's favorites are retrieved using the get_favorites_by_user_id() function and the variable $all_favorites being populated with the records. On the favorites page, it is displayed as count($all_favorites) which accurately shows how many favorites the user has.

#7) Show the logic related to the count of

#8) Show the logic related to filter/sort

Caption (required) ✓
Describe/highlight what's being shown
Code for showing the count of associations on the page

Explanation (required) ✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
If a filter is set, the logic for the count of items on the page is simply using the count() for the resulting $result array that is populated by the filter logic.

Caption (required) ✓
Describe/highlight what's being shown
Code for the filter logic

Explanation (required) ✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
The user is greeted with a form on the profile page that allows input for name, country, difficulty, and limit of records to narrow down results. The server then receives these variables and sanitizes, validates, and builds a query around the filter. The query is then used to conditionally select trails that match the filter from the database.

COLLAPSE

Task #3 - Points: 1

Text: Add related links

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Include the heroku prod link for the page that creates the association
#2	1	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

URL #1
https://it202-dr475-prod-7559be2e2ad4.herokuapp.com/project/favorites.php?

URL
https://it202-dr475-prod-7559be2e2ad4.herokua

id=17

URL #2
https://github.com/davidredziniak/dr475-IT202-452-M2024/pull/89

URL
https://github.com/davidredziniak/dr475-IT202-4

URL #3
https://github.com/davidredziniak/dr475-IT202-452-M2024/pull/88

URL
https://github.com/davidredziniak/dr475-IT202-4

+ ADD ANOTHER URL

All Users Association Page (likely an admin page) (2 pts.)

^COLLAPSE ^

^COLLAPSE ^

Task #1 - Points: 1

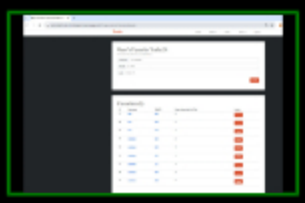
Text: Screenshots of this page

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.

#1) Show the summary of



Caption (required) ✓

Describe/highlight what's being shown

Admin favorites page

#2) Show the single view

Caption (required) ✓

Describe/highlight what's being shown

Ability to delete favorites individually

#3) Show the username

Caption (required) ✓

Describe/highlight what's being shown

Each username shown is a link to their profile

#4) Show variations of the number

Caption (required) ✓

Describe/highlight what's being shown

Image 1: Searching for username (test), Image 2: Searching for trail ID (610), Image 3: Searching for username (davideee)

#5) Show variations of

the

Caption (required) ✓
Describe/highlight what's being shown
Same as previous answer except last image. Last image (no results) is searching for trail 110 (no users have it favorite).

COLLAPSE

Task #2 - Points: 1

Text: Screenshot the code

Details:
Include ucid/date comments for each code screenshot

#1) Show the code related to

Caption (required) ✓
Describe/highlight what's being shown
Code for initially fetching all user favorites

#2) Show the code related to

Caption (required) ✓
Describe/highlight what's being shown
Code for displaying the table of favorites

Explanation (required) ✓
Explain in concise steps

#3) Each record should have

Caption (required) ✓
Describe/highlight what's being shown
Code for table (includes view for favorite)

Explanation (required) ✓
Explain in concise steps

#4) Each record should have

Caption (required) ✓
Describe/highlight what's being shown
Code for user profile link

Explanation (required) ✓
Explain in concise steps how this logically works

Explanation (required) ✓

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

PREVIEW RESPONSE

Upon loading the page, the query is set to the default (no filters). The count of user_favorites is retrieved using the get_number_of_favorites() function in favorite_helpers.php. The default query will retrieve all the favorites from the User_Favorites table and populates $favorites which is then displayed in a table on the page.

how this logically works and mention the logic for handling the username requirements

PREVIEW RESPONSE

The query to retrieve all the info needed to display is able to retrieve the favorite id, user ID, trail ID, the count of favorites for the specified trail ID, and the username (using the user ID) and a JOIN clause. To display the table, it simply uses a foreach loop that goes through the $favorites array (populated by the SQL query) and echos the information needed that is already in the array.

how this logically works

PREVIEW RESPONSE

In the table, there are two links, one to go to the user's profile, and another that navigates to the trail that is favorited (using the trail ID).

PREVIEW RESPONSE

Since the user ID is retrieved in the SQL query, it is trivial to add a link to the user's profile. The link for the user's profile is clickable on their username in the table.

#5) Each record should have

Caption (required) ✓

Describe/highlight what's being shown

Code for deleting a user's favorite

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

The favorite ID is retrieved in the SQL

#6) Show the logic related to

Caption (required) ✓

Describe/highlight what's being shown

Code for getting the total number of favorites

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

On every page load, the total number of favorites

#7) Show the logic related to

Caption (required) ✓

Describe/highlight what's being shown

Code for determine number of favorites shown

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

#8) Show the logic related to

Caption (required) ✓

Describe/highlight what's being shown

Code for filtering favorites

Explanation (required) ✓

Explain in concise steps how this logically works

query. In each record on the table, there is a hidden input fav_id that is set to the favorite's ID. If the delete button is pressed for a specific favorite, the server recognizes this through the "delete" parameter in the POST request, and calls delete_favorite_by_id() using the fav_id parameter that is set.

is calculated by calling the get_number_of_favorites() function in favorite_helpers.php that counts the number of records in the User_Favorites table. This accurately determines the number of favorite trails there are.

If an admin filters the number of favorites, the resulting array $favorites is populated with the filtered results. I simply used the count() function to count how many records are in the filtered results array, which determines the number of filtered favorites that will be shown.

PREVIEW RESPONSE

By default, the page is populated with favorites that are not filtered except the limit of 10 shown. Using the form, the user can specify the favorites by username, trail ID, and limit of records. Based on the user's input, a POST request is sent populating either the username, trail, and/or limit parameter which is then sanitized, validated, and used to build the query that fetches the filtered results.

COLLAPSE

Task #3 - Points: 1

Text: Add related links

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Include the heroku prod link for the page that creates the association
#2	1	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

URL #1
https://it202-dr475-prod-7559be2e2ad4.herokuapp.com/Project/admin/list_favorites.php

URL
https://it202-dr475-prod-7559be2e2ad4.herokua

URL #2
https://github.com/davidredziniak/dr475-IT202-452-M2024/pull/92

URL
https://github.com/davidredziniak/dr475-IT202-4

+ ADD ANOTHER URL

Unassociated Page (2 pts.)

COLLAPSE

Task #1 - Points: 1

Text: Screenshots of this page

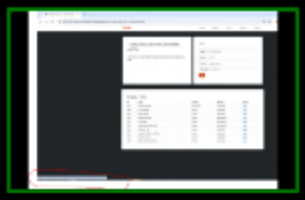
Details:
Make sure the heroku dev url is visible in the address bar

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

#1) Show the summary of

Caption (required) ✓
Describe/highlight what's being shown
Trails that are not favorited by any user

#2) Show the single view



Caption (required) ✓
Describe/highlight what's being shown
Link to trail for each trail that is not favorited

#3) Show variations of the number

Caption (required) ✓
Describe/highlight what's being shown
Image 1 (Limit 100 records), Image 2(Specific trail - Trail ID 612), Image 3 (Trails in United States)

#4) Show variations of the

Caption (required) ✓
Describe/highlight what's being shown
All images from before, Last image 4 is no trails found in Afghanistan

COLLAPSE

Task #2 - Points: 1

Text: Screenshot the code

Details:
Include ucid/date comments for each code screenshot

#1) Show the code related to

#2) Show the code related to

#3) Each record should have

#4) Show the logic related to

Caption (required) ✓

Caption (required) ✓
Describe/highlight what's being shown
Code for retrieving a list of trails that are NOT favorited

Explanation (required)
✓
Explain in concise steps how this logically works and mention how you determine the result list (include the unassociated logic and filters)
PREVIEW RESPONSE
Upon loading the page, there is no filter set. The query selects trail information if their ID is not found under trail_id in the Users_Favorites table which means they are not favorited by any user. The records of the trails found then populates the $unfavorited array (used to display the results table).

Caption (required) ✓
Describe/highlight what's being shown
Code for the table

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
As previously described, the $unfavorited array is populated by trails that are not found as favorited. The table is then rendered with each row pertaining to a record in the $unfavorited array, with its ID, name, country, and a link to the trail shown in the table.

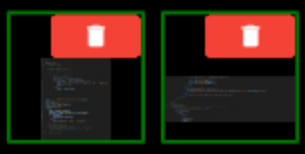
Caption (required) ✓
Describe/highlight what's being shown
Link to trail in table

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
Trails that are not favorited are fetched from the DB using the SQL query, one of the columns fetched is the ID column which is unique to each trail. For each row in the results table, there is a link to this trail at trail.php?id=(trailID).

Caption (required) ✓
Describe/highlight what's being shown
Code for determining the number of trails not favorited (associated)

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
On every page load, the number of trails that are not favorited is determined by calling the get_number_of_not_favorites() function in trail_helpers.php. This function uses a SELECT query to count the number of trails that are not found in the trail_id column in User_Favorites, which determines how many trails are unfavorited.

#5) Show the logic related to



Caption (required) ✓
Describe/highlight what's being shown
Code for results limit

#6) Show the logic related to

Caption (required) ✓
Describe/highlight

Explanation (required)

✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

On the page, the admin is allowed to specify the limit of results found using the Filter limit input field on the form. This sets the limit parameter which is then accessed by the server. If the limit is valid (1-100) it is used in the query that accesses the DB to retrieve trails that are not favorited.

what's being shown

Code for filtering unassociated trails

Explanation (required)

✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

I didn't realize the requirement for filtering was searching for unassociated entites by username since it wasn't specified in the Proposal document. I did the logic by name of trail and country, since it seemed more logical (searching for trails that are not favorited).

The form for the admin allows them to filter by name of trail and the country of the trail, as well as the limit of results to show (1-100) with a default of 10. These parameters are then sent to the server and sanitized/validated and used to build the query for the search.

COLLAPSE

Task #3 - Points: 1

Text: Add related links

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Include the heroku prod link for the page that creates the association
#2	1	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

URL #1
https://it202-dr475-prod-7559be2e2ad4.herokuapp.com/project/admin/list_unfavorited.php

https://it202-dr475-prod-7559be2e2ad4.herokua

URL #2
https://github.com/davidredziniak/dr475-IT202-452-M2024/pull/93

https://github.com/davidredziniak/dr475-IT202-4

+ ADD ANOTHER URL

Admin Association Management (like UserRoles) (1 pt.)

^COLLAPSE ^

^COLLAPSE ^

Task #1 - Points: 1

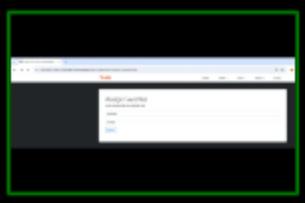
Text: Screenshots of the page

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.

#1) Show the search form with



Caption (required) ✓

Describe/highlight what's being shown

Assigning favorites to username: davideee, trail name: cheese

#2) Show the results of the

Caption (required) ✓

Describe/highlight what's being shown

Results found

#3) Show the result of entities and

Caption (required) ✓

Describe/highlight what's being shown

Toggle the trail that is favorited by "Davideee". Before and after (missing from favorites page since it was toggled).

^COLLAPSE ^

Task #2 - Points: 1

Text: Screenshots of the code

Details:
Include ucid/date comments for each code screenshot

#1) Search form field for finding a

Caption (required) ✓
Describe/highlight what's being shown
Form, and logic for finding partial match of users

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
The admin's form allows input for a users' username, which will be searched for using the query SELECT id, username FROM `Users` WHERE username LIKE :username LIMIT 25.

The records are then populated into the $users array to be displayed.

#2) Search form field for finding a

Caption (required) ✓
Describe/highlight what's being shown
Form, and logic for finding partial match of trail names

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
The admin's form allows input for a trail's name, which will be searched for using the query SELECT id, name FROM `Trails` WHERE name LIKE :name LIMIT 25.

The records are then populated into the $trails array to be displayed.

#3) Code related to getting a

Caption (required) ✓
Describe/highlight what's being shown
Code for limiting results to 25

Explanation (required)
✓
Explain in concise steps how this logically works and describe the steps for the search and how it works for users and entities
PREVIEW RESPONSE
The LIMIT clause for the queries are hardcoded at 25, which means the number of records being fetched by the DB for both users and trails is 25.

#4) Code that generates

Caption (required) ✓
Describe/highlight what's being shown
Code for checkboxes (users, and trails)

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
After the admin searches for users and trails, the $users and $trails array is populated with the results. For each record, a unique checkbox is conditionally rendered with the id and value containing it's unique ID in the database, which allows the server to accurately insert/delete a record when the association is toggled.

#5) Code related to submitting

#6) Code related to applying the

Caption (required) ✓
Describe/highlight what's being shown
Code for submitting the toggled users/trails

Explanation (required)
✓
Explain in concise steps how this logically works

PREVIEW RESPONSE

After the admin decides which associations to toggle, the users and trails IDs are sent to the server in an array, which is then accessed in the server when determining which associations to toggle.

Caption (required) ✓
Describe/highlight what's being shown
Code for toggling the associations

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

PREVIEW RESPONSE

After the admin submits the form which includes a users array of user IDs, and a trails array of trail IDs, the server then parses this data and iterates on the users ID array. For each user ID and each trail ID, a helper function toggle_favorite() is called which takes two parameters, a user ID and trail ID. Toggle_favorite determines if the user_favorite record exists for the specific trail_id first, then either deletes the association, or adds the association using helper functions delete_favorite_by_trail_id and add_favorite_by_trail_id. This occurs for each user and each trail in the array.

COLLAPSE

Task #3 - Points: 1

Text: Add related links

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Include the heroku prod link for the page that creates the association
#2	1	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

URL #1

https://it202-dr475-prod-7559be2e2ad4.herokuapp.com/Project/admin/assign_favorites.php

URL

https://it202-dr475-prod-7559be2e2ad4.herokua

URL #2

https://github.com/davidredziniak/dr475-IT202-452-M2024/pull/94

URL

https://github.com/davidredziniak/dr475-IT202-4

ADD ANOTHER URL

Misc (1 pt.)

COLLAPSE

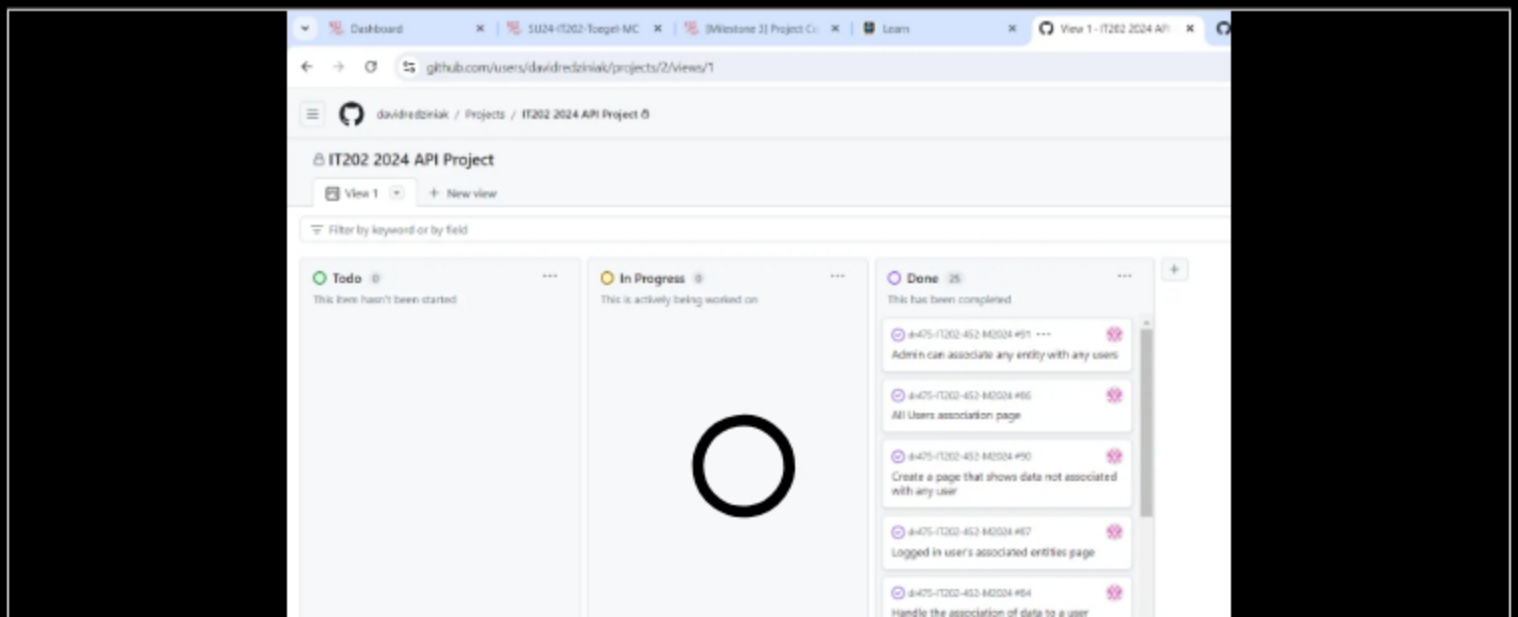
Task #1 - Points: 1

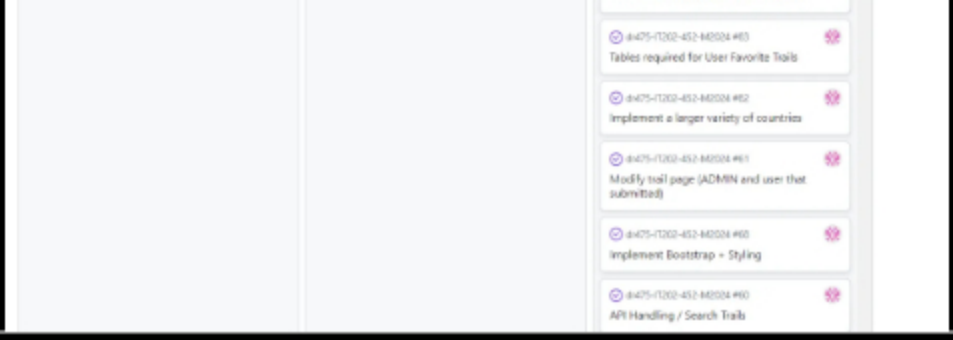
Text: Screenshot of your project board from GitHub (tasks should be in the proper column)

Task Screenshots:

Gallery Style: Large View

Small Medium Large





Finished tasks

COLLAPSE

Task #2 - Points: 1

Text: Provide a direct link to the project board on GitHub

URL #1

https://github.com/users/davidredziniak/projects/2

URL

https://github.com/users/davidredziniak/project

+ ADD ANOTHER URL

COLLAPSE

Task #3 - Points: 1

Text: Talk about any issues or learnings during this assignment

Response:

Through the development of the admin pages and favorites pages, I learned a lot about associating/unassociating entities from users and conditionally rendering tables.

COLLAPSE

Task #4 - Points: 1

Text: WakaTime Screenshot

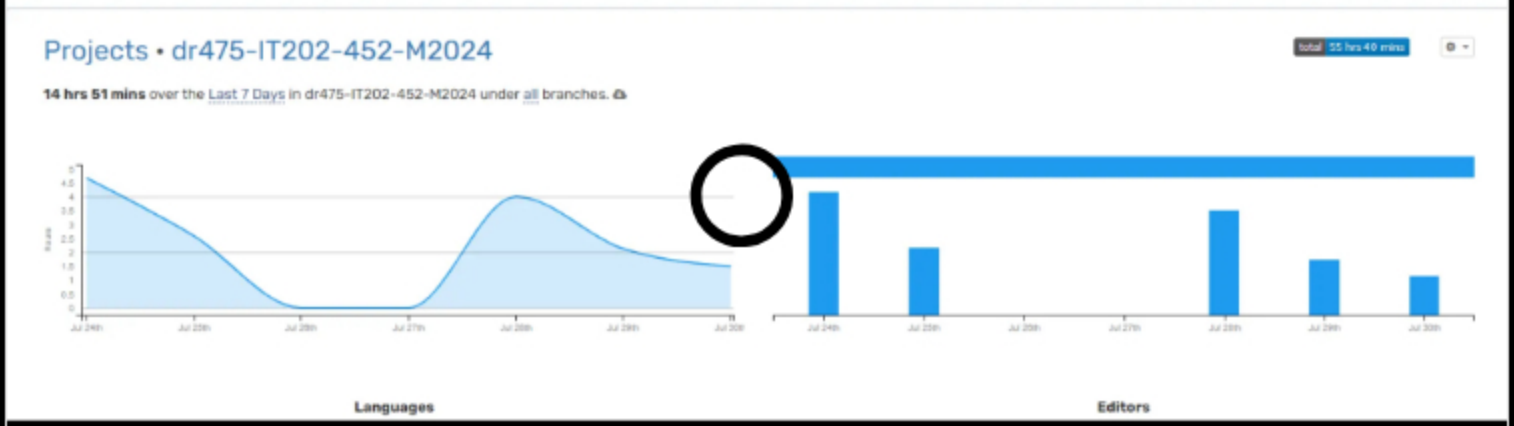
Details:

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: Large View

Small Medium Large



Roughly 15 hours in the past 7 days (milestone 3).

End of Assignment