Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT202-007-F2024/it202-api-project-milestone-3-2024-m24/grade/db624

Course: IT202-007-F2024
Assigment: [IT202] API Project Milestone 3 2024 m24
Student: Datha V. (db624)

Submissions:

Submission Selection

1 Submission [submitted] 12/12/2024 12:16:31 AM

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

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:

Concept: Users can save games to their profile

Relationship type: many to many because one user can save many games and one game can have many users who saved it.

In order to store the associations, I created a games associations table where there is a column for game ID and user ID, specifying a relationship where a user has saved a game. The combination of user ID and game ID is enforced to be unique as a user can't save the same game twice.

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:

Since the associations are handles through a game associations table that only saves the game id and user id, when a game is updated, it's updated data will also be displayed to the user. The data does not need to be reassociated,

however, if the game has been deleted, the user will also see that the game is gone from their profile.

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

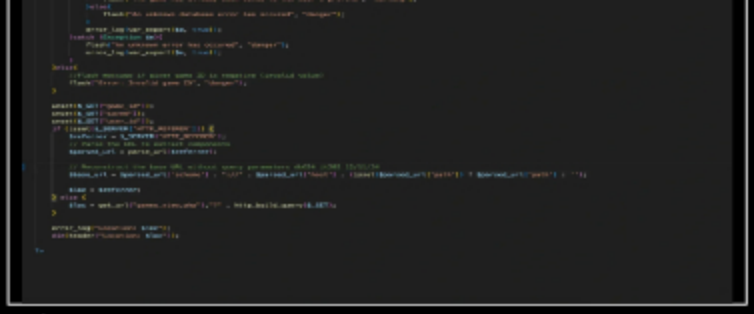
Task Screenshots

Gallery Style: 2 Columns

4 2 1

ss1

ss2



ss3

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:

First I check if the user is logged in and redirect them if not.

Next, I check if a valid game id has been provided through query params.

Then I check if the game has been saved or not by user (info obtained through query params).

If it is not saved, I perform an insertion query into games associations table where I create an association between user id and game id.

If saved, I delete the association where given user id and game id are matching.

Once the operation is done, I flash a message whether it succeeded or not, and try to redirect the user back the site they came from while trying to preserve all other query parameters besides the game id and saved status.

for actually calling this function, I have a button in related pages and it'll navigate to this page with query parameters appended to URL for the game id and save status.

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

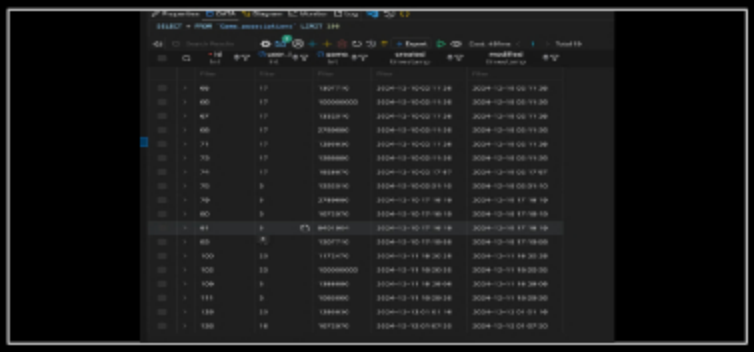
100%

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



game associations table

Caption(s) (required) ✓

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

Task Response Prompt

Describe each column/association table

Response:

I have the standard columns for ID of record, timestamps for creation and modification. Besides those I have two columns specifying user ID and game ID. This is used to represent the relationship between a user and the game they have saved. There will be a record for every single game saved by a user.

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-db624-prod-a236ea831ca5.herokuapp.com/Project/game_save.php

URL
https://it202-db624-prod-a236ea831ca5.herokua

URL #2
https://github.com/Dathster/db624-it202-007/pull/87

URL
https://github.com/Dathster/db624-it202-007/pu

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

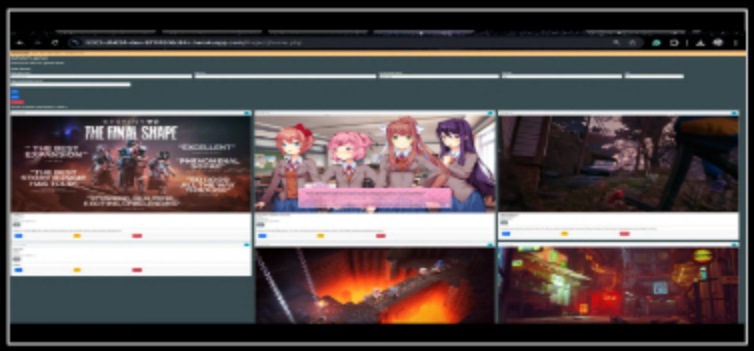
100%

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



Note, the button with heart is used to remove game from profile or save game to profile, delete button below is admin function

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

Sub-Task

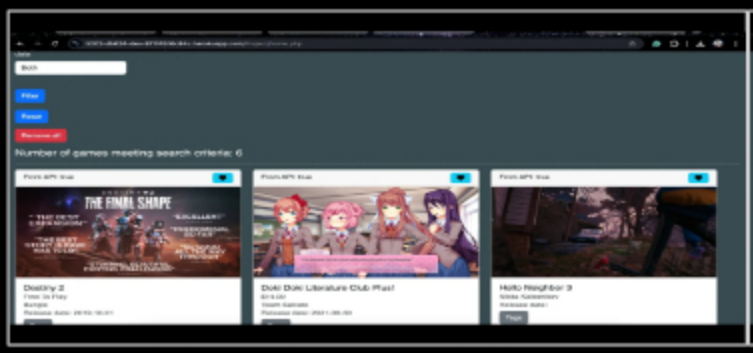
100%

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

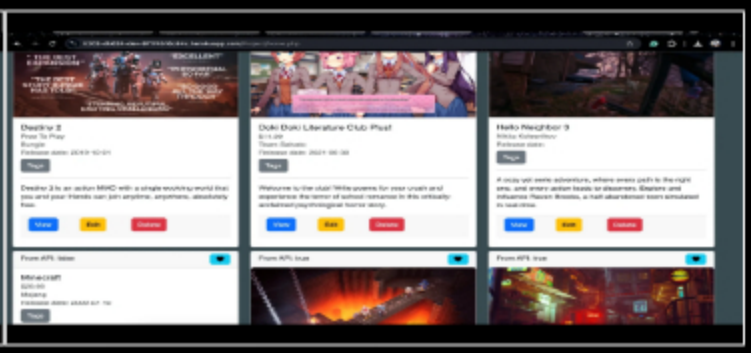
Task Screenshots

Gallery Style: 2 Columns

4 2 1



Remove all button visible, and heart button to remove association (since game is already saved)



View button to view the game's page

Caption(s) (required) ✓

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

Sub-Task

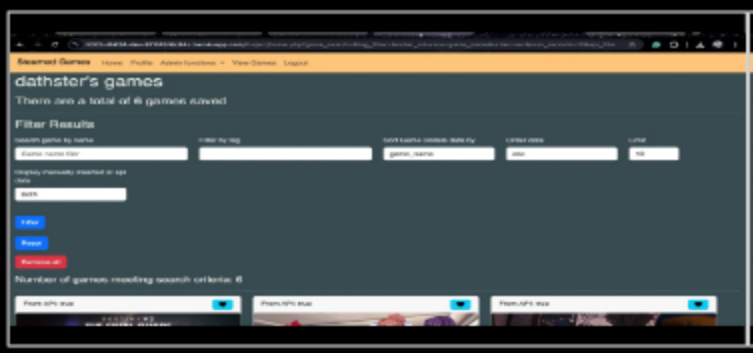
100%

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

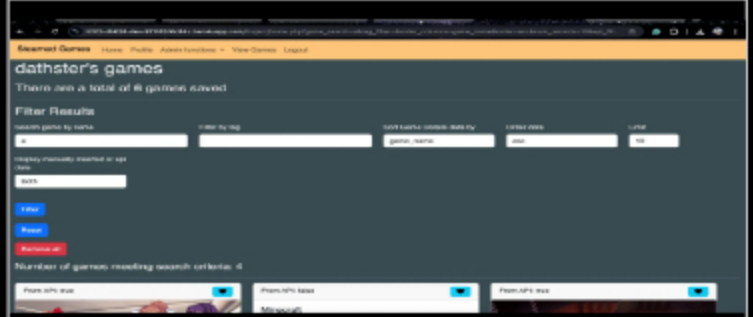
Gallery Style: 2 Columns

4 2 1



no filters

filter A



filter 3

Caption(s) (required) ✓

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

Sub-Task

100%

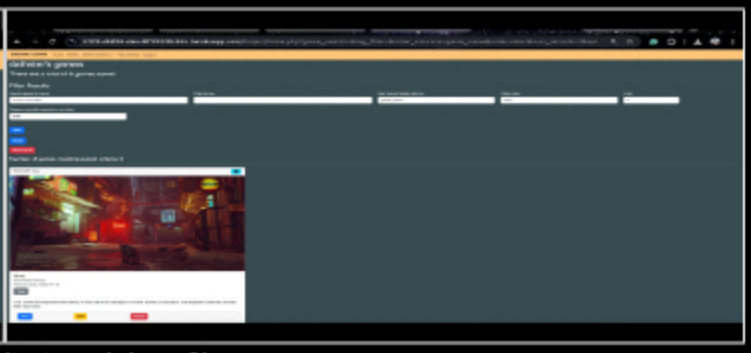
Group: Current User's Association Page
Task #1: Screenshots of this page
Sub Task #4: Show variations of the filter/sort including no results (proper message should be visible)

Task Screenshots

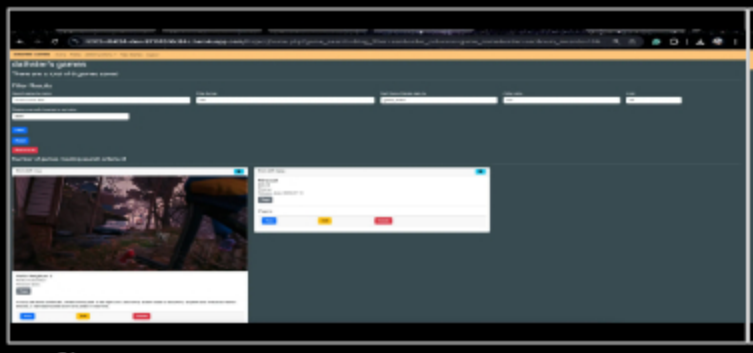
Gallery Style: 2 Columns

4 2 1

No matches



limit and desc filter



tag filter

name filter

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

ss 1

2 ss

3 ss

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:

I use a method to return search results for matching games. It applies all the filters chosen by user to sql query using where clause. records associated with users are filtered when the mode is set to saved. I use the in clause to select all games from Games_details only if the game id is found within Game_associations and the user id associated with that record is equal to the current user's id, which means that the user has saved the game.

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

1 ss

2 ss

3 ss

4 ss

5 ss

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

I obtain all the data to render cards from the return search results function from the previous set of screenshots. I then iterate through all the returned results in a foreach loop, and for each result I render a card. This card contains data such as an image of the game if available, name of the game, release date, developer info, price, a list of tags, a short description, and links for either viewing, editing, or deleting the game (latter 2 are admin only). I use PHP conditional statements to render results such as the edit, delete, and image fields. Since we are in the saved games page, a button will also be rendered (the heart button) so that the user can unsave the game. This is also rendered with a PHP conditional where it will be rendered if the save url is provided, which it will be in the case that mode is set to saved for returnSearchResults.

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

variable declarations for edit view delete buttons

code to render edit view delete

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

These buttons are again rendered through my render card function. I use php conditionals to check if the urls for edit view and delete have been provided and render the respective buttons based off that. I use variables such as view label, view classes, etc to modify the way these buttons are displayed. The return search results method will decide to include links for edit and delete based on whether the user making search queries has admin permissions. If they don't, the URLS are not provided, thus the links for edit and delete will also not be rendered.

Sub-Task

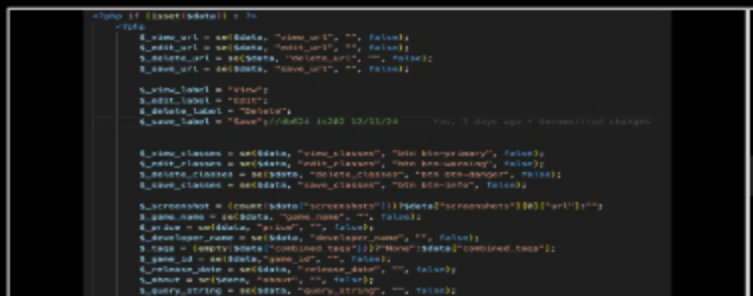
100%

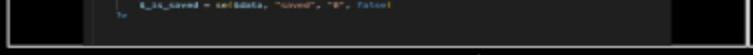
Group: Current User's Association Page
Task #2: Screenshot the code
Sub Task #4: Each record should have a button/link for delete (this may be an admin-only thing but should be present for the specific role) Note: this is to delete the relationship and not the specific entity

Task Screenshots

Gallery Style: 2 Columns

4 2 1





Variables for the save/unsave button (save url, save lavel, save classes, is saved

calling render like method

code for rendering like

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

First I check if a save url is provided, if so I render the like button. I send the like button info on whether the game is saved or not. If it is saved, the heart is filled in, otherwise it is empty. When the user clicks the button, it'll navigate them to the gave same page with data on game id and save satus passed through query params, and an insertion or deletion operation is performed based on given save value.

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

php code for executing delete

form for rendeing remove all button

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

When the user pressed the remove all button, the form is submitted and the php code checks if the remove all value has been set. If so it'll execute a delete query where it'll drop all records from the game associations table where the user id matches current user's id.

Sub-Task 100%

Group: Current User's Association Page
Task #2: Screenshot the code
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

```
//echo $total;
//db624 it202 12/11/24    You, 2 days ago • Uncommitted changes
$query_num_saved_games = "select count(*) as `ct` from `Game_associations` where `user_id`=$user_id";
$num_saved_games = exec_query($query_num_saved_games)[0]["ct"];
```

code for count of num saved games

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

I created a select count query for sql that returns a count of all the game IDs from the games details table that aren't found in the games associations table, since the game associations table only contains games that have been saved by a user. This gives me the count of all the games that haven't been saved by anyone.

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)

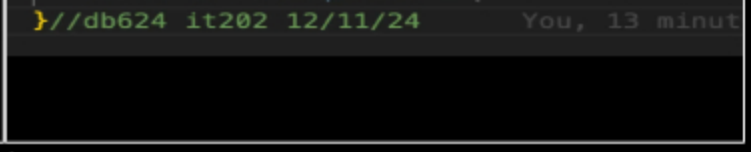
Task Screenshots

Gallery Style: 2 Columns

4 2 1

```
function potentialTotalRecords($query){
    $results = exec_query($query);
    return count($results);
```

1 ss



2 ss

3 ss

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

To return total number of records meeting search criteria, I use the original sql query to return all the games with their details, and all filters applied except the one to limit number of records. I pass this query onto a function that executes the query and returns the count of results. I pass this onto the $total_records variable, which is a parameter passed by reference. This means that if I modify the value of the variable inside the function, it is also changed outside the function. So I pass a variable to the search function from outside to get the number of total search results, and then display that.

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

code for populating params and calling returnsearchresults ss 2

function

ss 3

ss 4

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

First I get all the search filters chosen by the user from the post array and store them in variables. Then, I send all those filters into a method to return all matching games.

The returnSearchResults function uses the following filters: game search query, game tag search, maximum number of records to be returned, column to order data by, ascending or descending order, filter for api or manually inserted data, a variable passed by reference to obtain total number of results returned by the sql query, offset for pagination, mode to determine whether saved or unsaved games should be returned, and a user id if we want to return games associated with a different user.

The search method first makes sure the limit is within 1-100 and sets it to 10 if that limit is not met.

Next, it creates an SQL query. The SQL query first gets details of the game such as id, name, release date, developer name, all the tags, etc. Next, based on the mode, it'll decide whether to filter all these games based on whether they've been saved by a user or haven't been saved by any user. Then, it'll apply another filter to sort the games based on a partial match for game name, after which it'll filter based on whether the data was generated by API or manually inserted, and then it'll check if the tags of the game contain a match for the given tag by user, and finally I'll sort the records based on column picked by user, the order of the data, and limit on number of records. I'll append all the results associated with one game into an associative array and append that to another associative array containing results for all matched games and return that to the calling function.

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://github.com/Dathster/db624-it202-007/pull/91

URL
https://github.com/Dathster/db624-it202-007/pul

URL #2
https://it202-db624-prod-a236ea831ca5.herokuapp.com/Project/home.php

URL
https://it202-db624-prod-a236ea831ca5.herokua

End of Task 3

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

Group

100%

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

COLLAPSE

Task

100%

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

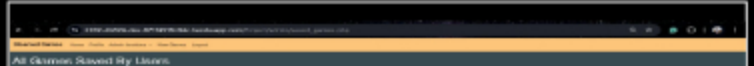
100%

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



Displaying all users with saved games and each game saved by the users

Caption(s) (required) ✓

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

Sub-Task 100%

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

buttons for view and removing relationship

Caption(s) (required) ✓

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

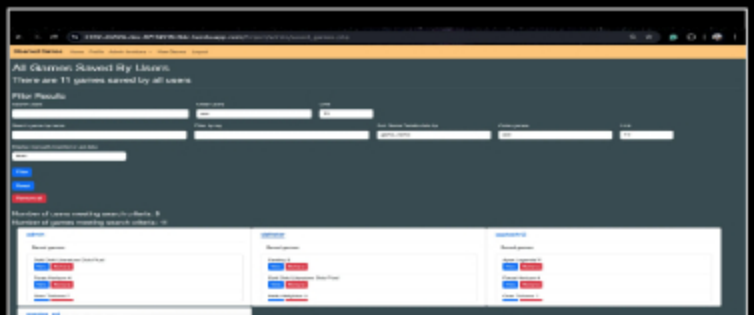
Sub-Task 100%

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



showing the underline of the link for username

Caption(s) (required) ✓

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

Sub-Task

100%

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

ss 1

Caption(s) (required) ✓

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

Sub-Task

100%

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

ss 1

ss 2

ss 3

Caption(s) (required) ✓

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

End of Task 1

Task

100%

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

100%

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

code

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:

This foreach loop runs through all the users who passed filter criteria and then for each of them executes the return search results function based on given game filter criteria to return all the games that have been saved by that user, in essence returning all the valid game associations for each user.

Sub-Task

100%

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

code

Caption(s) (required) ✓

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

Task Response Prompt

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

Response:

I am using foreach loops to make cards for every user who has games saved with them. Within the card, I am using another foreach loop to render all the games saved for each user. I am adding a link to the user's username so that it can link to their profile and I am also rendering buttons to the view page for each game and button to remove relationship between user and game.

Sub-Task

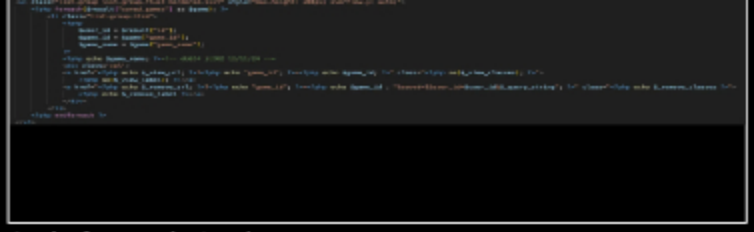
100%

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



Code for rendering buttons

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

I built a foreach loop to go through all the games that are saved by a user and for each game I'm rendering buttons with urls and bootstrap classes for single game view page and removing the relationship between user and game.

Sub-Task

100%

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

I am rendering a link as part of card title

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

I am rendering a link as part of card title which links to home.php with the user id given as a query parameter. This allows me to display all the games associated with that user in their profile page.

Sub-Task

100%

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

remove url

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

I am rendering a button linked to remove url along with every game within the foreach loop. The code appends game id and user id as query parameters and redirects user to the page where game association can be removed.

Sub-Task

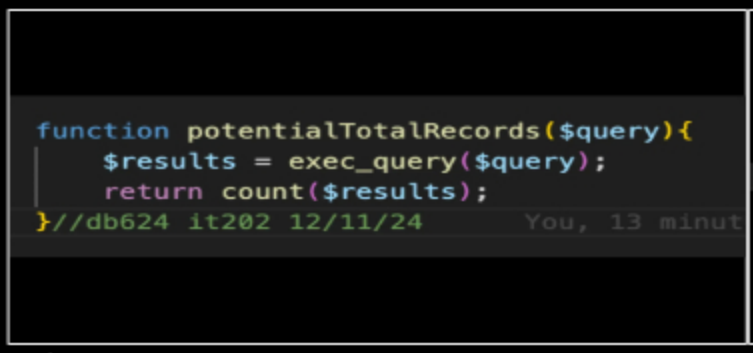
100%

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



```
function potentialTotalRecords($query){
    $results = exec_query($query);
    return count($results);
}//db624 it202 12/11/24      You, 13 minut
```

ss 2 ss 1

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

This returns the number of games that have been saved by all users

Sub-Task 100%

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

query

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

This uses the return search results query like other functions, but the "game_filter_count" mode fetches records saved by all users and I'm passing the total games variable to this method so that i can get a count of all the games that match filter criteria and have been saved by all users.

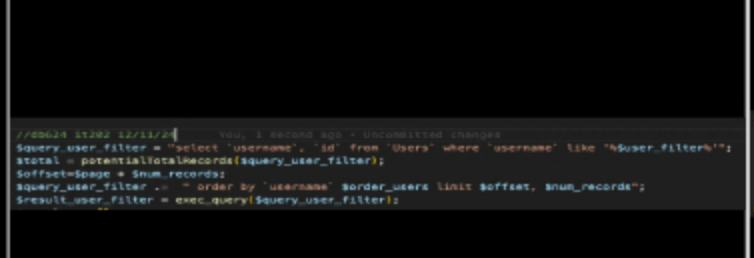
Sub-Task 100%

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



code for filtering usernames

Code for calling return search results function

return search results 1

return search results 2

return search results 3

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

I use a select query with a where clause to filter the users that match given search query and then I use the limit clause to limit number of users returned. I send the filters for games to return search results function which as mentioned before applies all the filters to the games and returns games associated with the user that meet all given search criteria.

End of Task 2

Task

100%

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
https://github.com/Dathster/db624-it202-007/pull/93

URL
https://github.com/Dathster/db624-it202-007/pul

URL #2
https://it202-db624-prod-a236ea831ca5.herokuapp.com/Project/admin/saved_games.php

URL
https://it202-db624-prod-a236ea831ca5.herokua

End of Task 3

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

Group

100%

Group: Unassociated Page
Tasks: 3
Points: 2

COLLAPSE

Task

100%

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

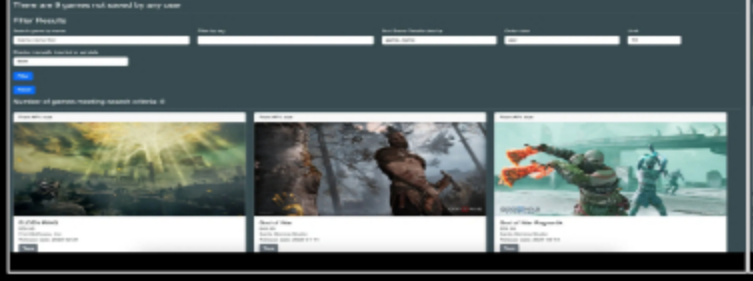
100%

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

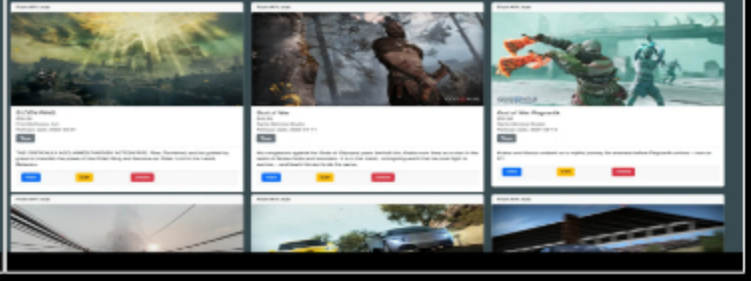
Task Screenshots

Gallery Style: 2 Columns

4 2 1



unsaved games screenshot 1



Unsaved games screen

Caption(s) (required) ✓

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

Sub-Task

100%

Group: Unassociated Page

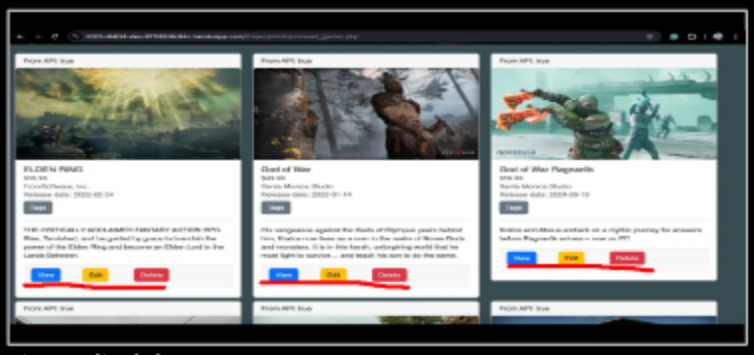
Task #1: Screenshots of this page

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

Task Screenshots

Gallery Style: 2 Columns

4 2 1



view edit delete ss

Caption(s) (required) ✓

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

Sub-Task

100%

Group: Unassociated Page

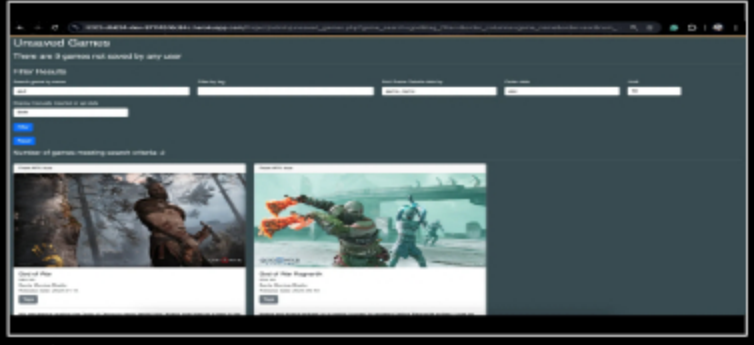
Task #1: Screenshots of this page

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



ss 1

Caption(s) (required) ✓

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

Sub-Task

100%

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

ss 1

ss 2

ss 3

Caption(s) (required) ✓

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

End of Task 1

Task

100%

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

100%

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

code for fetching unassociated games 1

search results ss 1

search results ss 2

search results ss 3

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 unassociated logic and filters)

Response:

I don't have code specifically for fetching all unassociated entities, I have a return search results function that has an initial sql query to get all the games from games details and then applies filters based on parameters given to the return search results function. Specifically there is a mode parameter, when it is set to unsaved, I add a filter to the games details query that ensures that whatever records are returned do not have their game ids in the game associations table. After that, all other filters are applied based on what the user input.

Sub-Task

100%

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

render results ss 1

render ss 2

render ss 3

render ss 4

render ss 5

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

I obtain all the data to render cards from the return search results function from the previous set of screenshots. I then pass iterate through all the returned results in a foreach loop, and for each result I render a card. This card contains data such as an image of the game if available, name of the game, release date, developer info, price, a list of tags, a short description, and links for either viewing, editing, or deleting the game (latter 2 are admin only). I use PHP conditional statements to render results such as the edit, delete, and image fields.

Sub-Task

Group: Unassociated Page

Task #2: Screenshot the code

100%

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

Task Screenshots

Gallery Style: 2 Columns

4 2 1

variable declarations for edit view delete buttons

code to render edit view delete

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

These buttons are again rendered through my render card function. I use php conditionals to check if the urls for edit view and delete have been provided and render the respective buttons based off that. I use variables such as view label, view classes, etc to modify the way these buttons are displayed. The return search results method will decide to include links for edit and delete based on whether the user making search queries has admin permissions. If they don't, the URLS are not provided, thus the links for edit and delete will also not be rendered.

Sub-Task

100%

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

code to return total records

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:

I created a select count query for sql that returns a count of all the game IDs from the games details table that aren't found in the games associations table, since the game associations table only contains games that have been saved by a user. This gives me the count of all the games that haven't been saved by anyone.

Sub-Task

100%

Group: Unassociated Page

Task #2: Screenshot the code

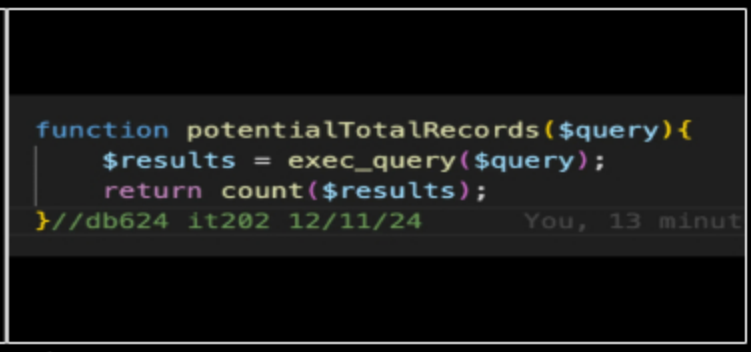
Sub Task #5: Show the logic related to the count of the items on the page (this value should change based on the filter applied)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

ss 1



```
function potentialTotalRecords($query){
    $results = exec_query($query);
    return count($results);
}//db624 it202 12/11/24       You, 13 minut
```

ss 2

ss 3

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

To return total number of records meeting search criteria, I use the original sql query to return all the games with their details, and all filters applied except the one to limit number of records. I pass this query onto a function that executes the query and returns the count of results. I pass this onto the $total_records variable, which is a parameter passed by reference. This means that if I modify the value of the variable inside the function, it is also changed

outside the function. So I pass a variable to the search function from outside to get the number of total search results, and then display that.

Sub-Task

100%

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

ss 1

ss 2

ss 3

ss 4

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

Task Response Prompt

Explain in concise steps how this logically works
Response:

First I get all the search filters chosen by the user from the post array and store them in variables. Then, I send all those filters into a method to return all matching games.

The returnSearchResults function uses the following filters: game search query, game tag search, maximum number of records to be returned, column to order data by, ascending or descending order, filter for api or manually inserted data, a variable passed by reference to obtain total number of results returned by the sql query, offset for pagination, mode to determine whether saved or unsaved games should be returned, and a user id if we want to return games associated with a different user.

The search method first makes sure the limit is within 1-100 and sets it to 10 if that limit is not met.

Next, it creates an SQL query. The SQL query first gets details of the game such as id, name, release date, developer

name, all the tags, etc. Next, based on the mode, it'll decide whether to filter all these games based on whether they've been saved by a user or haven't been saved by any user. Then, it'll apply another filter to sort the games based on a partial match for game name, after which it'll filter based on whether the data was generated by API or manually inserted, and then it'll check if the tags of the game contain a match for the given tag by user, and finally I'll sort the records based on column picked by user, the order of the data, and limit on number of records. I'll append all the results associated with one game into an associative array and append that to another associative array containing results for all matched games and return that to the calling function.

End of Task 2

Task

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
https://it202-db624-prod-a236ea831ca5.herokuapp.com/Project/admin/unsaved_games.php

URL
https://it202-db624-prod-a236ea831ca5.herokua

URL #2
https://github.com/Dathster/db624-it202-007/pull/92

URL
https://github.com/Dathster/db624-it202-007/pul

End of Task 3

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

Group

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

COLLAPSE

Task

Group: Admin Association Management (like UserRoles)
Task #1: Screenshots of the page

100%

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

100%

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

Screenshot of admin association page

Caption(s) (required) ✓

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

Sub-Task

100%

Group: Admin Association Management (like UserRoles)

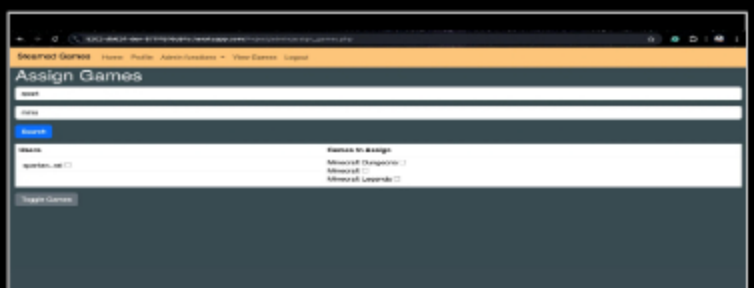
Task #1: Screenshots of the page

Sub Task #2: Show the results of the search

Task Screenshots

Gallery Style: 2 Columns

4 2 1



results of search

Caption(s) (required) ✓

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

Sub-Task

100%

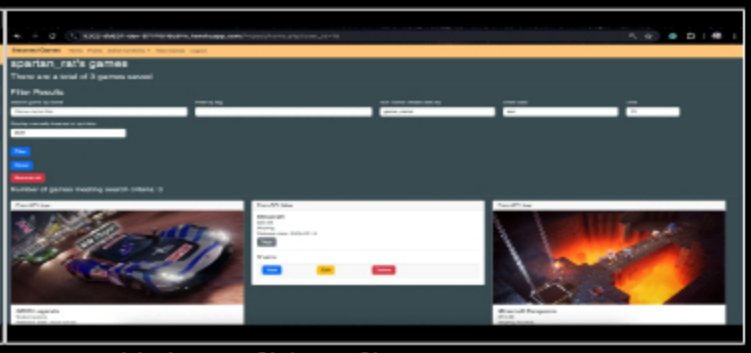
Group: Admin Association Management (like UserRoles)
Task #1: Screenshots of the page
Sub Task #3: Show the result of entities and users being associated and unassociated

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Saving games to spartan rat's profile successful



games added to profile's profile

Caption(s) (required) ✓

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

End of Task 1

Task

100%

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

100%

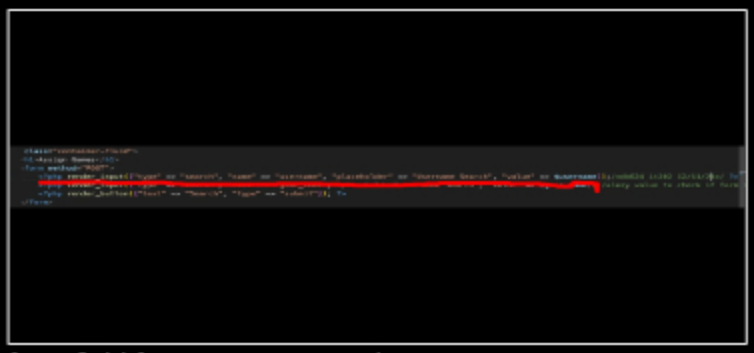
Group: Admin Association Management (like UserRoles)
Task #2: Screenshots of the code
Sub Task #1: Search form field for finding a partial match of usernames

Task Screenshots

Task Screenshots

Gallery Style: 2 Columns

4 2 1



form field for username search

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

I am using the render input function to build a search field where the user can input a username search query. It'll send the user input to the php script and it'll return search results.

Sub-Task

100%

Group: Admin Association Management (like UserRoles)

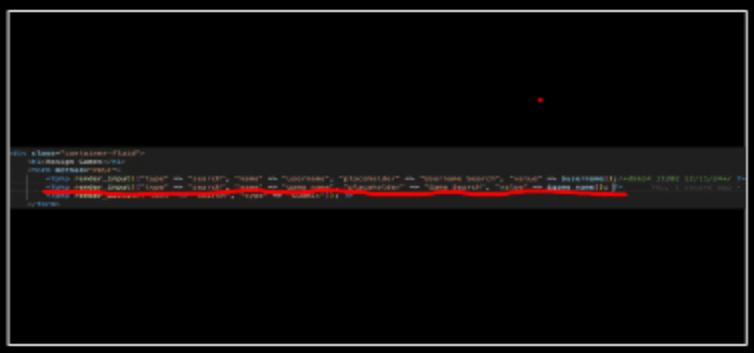
Task #2: Screenshots of the code

Sub Task #2: Search form field for finding a partial match of entities

Task Screenshots

Gallery Style: 2 Columns

4 2 1



game search screenshot

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

I use the render input function to create a text input field, it'll store the user input in a variable called game_name and

send it to the php script when the form is submitted through a post request.

Sub-Task

100%

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

Getting max 25 matched results for usernames

Code for returning max 25 matched games

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

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:

username results: I wrote an SQL to return usernames and user IDs from the Users table where the records match the search query given by the user. I used PHP PDO to execute this query and store the results in an array.

Game results: I made a similar SQL query using the select clause to select the game name and game ID from Games details, uses the where clause to filter the games based on user search query, and the limit clause to return at most 25 matched records. I added the where clause if the user had given a game search query, otherwise it'll directly append the limit clause. I used PHP PDO to execute this SQL query and store results in an associative array.

Sub-Task

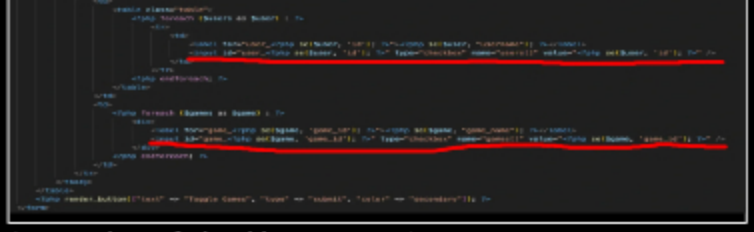
100%

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



Screenshot of checkbox generation

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 generating checkboxes consists of two input fields, one is a label with either the username or the name of the game, and then there is a checkbox input associated with that entry. The for and id attributes of each form elements are used to bind them together. The name of the checkbox elements are users[] and games[]. By using brackets in their names, we are able to group all the chosen usernames and chosen games into two seperate arrays so that they can all be processed easily. These checkboxes are also being rendered within a table so that we can have all the users in one column and all the games in another column.

Sub-Task

100%

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

code for submission

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

The checkboxes are rendered within a table which is rendered within a form. Each checkbox element's name is an array so that all games and all users will be grouped into a users array and a games array so that in the PHP side, we can simply access all of the games and users by accessing that array.

Sub-Task

100%

Group: Admin Association Management (like UserRoles)
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

Code for applying associations

Caption(s) (required) ✓

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

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:

First I get the list of users and list of games chosen through post and they are stored in an array. Next, I check if both users and games are selected and otherwise flash an error message. Once I've verified that both games and users are selected, I use a foreach loop where I go and update associations for each user between that user and all the games selected. For updating the association, I first perform a select query to see if an association exists already in the table, if so, I perform a delete query where I delete the association between the user and game. Else, I perform an insert operation. I flash a success method if the association update was successful and flash an error message if there was an error.

End of Task 2

Task

100%

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
https://github.com/Dathster/db624-it202-007/pull/95

URL
https://github.com/Dathster/db624-it202-007/pu

URL #2
https://it202-db624-prod-a236ea831ca5.herokuapp.com/Project/admin/assign_games.php

URL
https://it202-db624-prod-a236ea831ca5.herokua

End of Task 3

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

Group

100%

Group: Misc
Tasks: 4
Points: 1

COLLAPSE

Task

100%

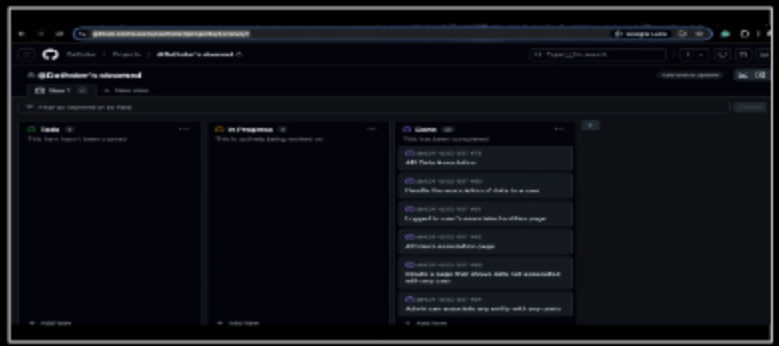
Group: Misc
Task #1: Screenshot of your project board from GitHub (tasks should be in the proper column)
Weight: ~25%
Points: ~0.25

COLLAPSE

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Screenshot of github project board

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/Dathster/projects/4/views/1

URL
https://github.com/users/Dathster/projects/4/vie

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:

With Milestone 3, I learned a lot more about SQL again because I had to implement logic for checking if a record already existed and inserting or deleting a record based off that. I also became more familiar with bootstrap as I went through a lot of trouble to implement toast messages as an alternative to flash messages provided by professor. I learned that when trying to create bootstrap elements after page load, some attributes aren't assigned to them and we need to manually set those attributes. Examples are the autohide and delay features for toast messages.

End of Task 3

Task

100%

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

COLLAPSE

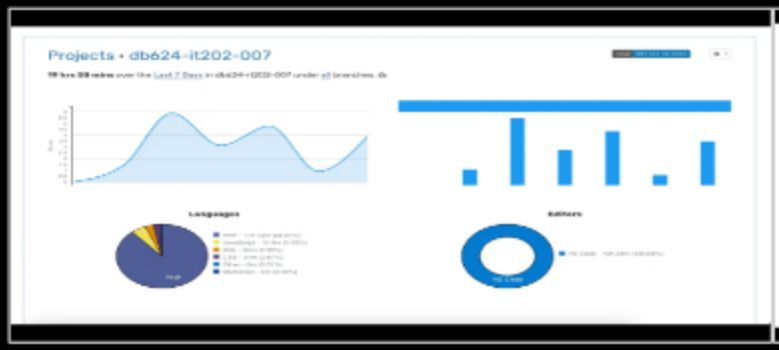
Details:
Grab a snippet showing the approximate time involved that clearly shows your repository. The duration isn't considered for grading, but there should be some time involved

Task Screenshots

Gallery Style: 2 Columns

Gallery Style: 2 Columns

4 2 1



Wakatime screenshot 1

Wakatime screenshot 2

End of Task 4

End of Group: Misc
Task Status: 4/4

End of Assignment