Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT202-450-M2024/it202-api-project-milestone-3-2024-m24/grade/jed56

IT202-450-M2024 - [IT202] API Project Milestone 3 2024 m24

Submissions:

Submission Selection

1 Submission [active] 7/28/2024 9:25:23 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:

1. The association is between users and products. Users can browse through the best deals on amazon and add certain products to their wishlist if they would like.
2. The relationship is many to many. Multiple users can have multiple products in their wishlist, and a product can be associated with many different users.
3. Another consideration is data integrity because there cannot be any duplicate associations. A user can not add a product to their wishlist multiple times so integrity constraints were used.

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:

1. When an update occurs the associated data is updated to be the same as the new data.
2. Users see the new data immediately after the update happens.

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

Caption (required) ✓

Describe/highlight what's being shown

The code for adding to wishlist

Explanation (required) ✓

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

PREVIEW RESPONSE

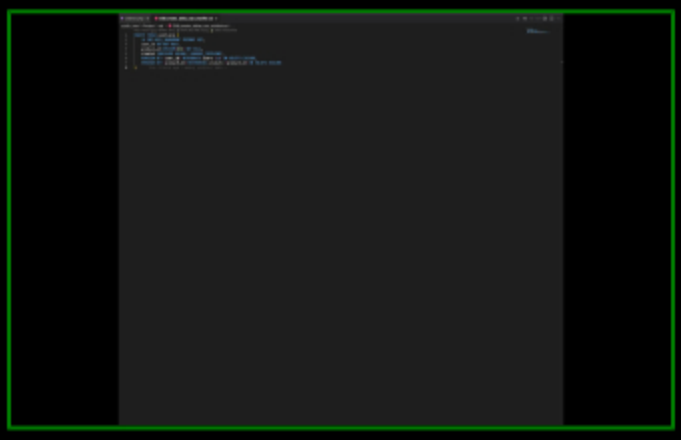
The code first checks if the user is logged in beacuse this is a log in protected page. It then creates a post request that gets the user id and product id and connects to the database. A SQL query is then run to select that product from the table. If the product is already in the users wishlist then an appropriate message is shown. If not, there is an INSERT INTO statement executed to insert that users id and the products id into the wishlist table.

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
wishlist table

Explanation (required) ✓
Describe each column/association table
PREVIEW RESPONSE
The wishlists tabble has an id that auto increments for every entry. It also has a user_id which is a foreign key from the users table, and product_id which is a foreign key from the products table. It also includes a created column for data creation timestamp.

COLLAPSE

Task #3 - Points: 1

Text: Add related links

Checklist — *The checkbox 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-jed56-prod-553fb5e41cd0.herokuapp.com/Project/list_data.php
URL: https://it202-jed56-prod-553fb5e41cd0.herokuap

URL #2
https://github.com/jordand2003/jed56-it202-450/pull/64
URL: https://github.com/jordand2003/jed56-it202-450

+ 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

Caption (required) ✓
Describe/highlight what's being shown
showing summary of results of wishlist page

#2) Show the single view buttons/links

Caption (required) ✓
Describe/highlight what's being shown
view, delete, and delete all buttons

#3) Show variations of the number

Caption (required) ✓
Describe/highlight what's being shown
showing number of items count changing

#4) Show variations of the filter/sort

Caption (required) ✓
Describe/highlight what's being shown
showing filters/sort with no results

COLLAPSE

Task #2 - Points: 1

Text: Screenshot the code

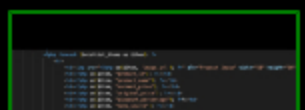
Details:

Include ucid/date comments for each code screenshot

#1) Show the code related to fetching

#2) Show the code related to the display

#3) Each record should have



#4) Each record should have

Caption (required) ✓
Describe/highlight what's being shown
showing code related to fetching user associations

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
The code gets the users is, result limit, sorting information, and then creates a query with the correct order of the filtering information. The query then selects the users wishlist items by joining the wishlist and products tables, and then the results are stored in the variable $wishlist_items.

Caption (required) ✓
Describe/highlight what's being shown
showing code for displaying results

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
The total number of items in the wishlist is displayed, then a form is given for the user to filter and sort the results. If there are no results from the filtering or no products in the wishlist, then a message is displayed saying no results available. If there are results then they are shown in the table. Then a button is created to remove all items from the wishlist by redirecting to the remove_from_wishlist file.

Caption (required) ✓
Describe/highlight what's being shown
form button for single view

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
In the form, a link is given for the user to click single view on a product. The button is a simple link that redirect the url to the view_product.php file and the product id is passed into the url to view that item on a single page.

Caption (required) ✓
Describe/highlight what's being shown
delete from wishlist button being shown

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
This delete button is a part of a form and when it is clicked it submits the form to the file remove_from_wishlist.php which handles the deleting of the relationship between the user and the entity.

#5) Show the logic for deleting all

Caption (required) ✓
Describe/highlight what's being shown
showing logic for deleting all associations

#6) Show the logic related to the count

Caption (required) ✓
Describe/highlight what's being shown
showing logic for

#7) Show the logic related to the count

Caption (required) ✓
Describe/highlight what's being shown
showing logic for

#8) Show the logic related to filter/sort

Caption (required) ✓
Describe/highlight what's being shown
showing logic for

counting all items

Explanation (required)

✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

A form is given with a button to remove all associations. When the button is clicked the code sends a post request if the button is clicked and a delete statement is prepared to delete all records from the wishlist table where the user id matches. Finally a success message is shown when all items are deleted.

counting all items

Explanation (required)

✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

A variable $total_items is set which simply counts the amount of items in the wishlist_items variable which contains all the results of the SQL query with the count function. The variable is then displayed with the title of the page.

counting all items

Explanation (required)

✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

A variable $total_items is set which simply counts the amount of items in the wishlist_items variable which contains all the results of the SQL query with the count function. The variable is then displayed with the title of the page.

filter/sort

Explanation (required)

✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

The code sets the limit parameter to be between 1 and 100 and defaults to 10. The sort_by and sort_order parameters are then sanitized to be used in the query. It also checks the search parameter and if it is not empty then it is also added tot he query. The final query results are fetched and stored in the wishlist_items variable.

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-jed56-prod-553fb5e41cd0.herokuapp.com/Project/wishlist.php

URL
https://it202-jed56-prod-553fb5e41cd0.herokuap

URL #2
https://github.com/jordand2003/jed56-it202-450/pull/66

URL
https://github.com/jordand2003/jed56-it202-450

+ ADD ANOTHER URL

COLLAPSE

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

^COLLAPSE^

Task #1 - Points: 1

Text: Screenshots of this page

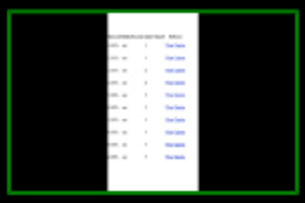
Details:
Make sure the heroku dev url is visible in the address bar

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

#1) Show the summary of the results

Caption (required) ✓
Describe/highlight what's being shown
showing results of all user associations

#2) Show the single view buttons/links



Caption (required) ✓
Describe/highlight what's being shown
showing single view and delete buttons

#3) Show the username related to the

Caption (required) ✓
Describe/highlight what's being shown
showing username related to entity

#4) Show variations of the number

Caption (required) ✓
Describe/highlight what's being shown
showing variation of number of shown items count

#5) Show variations of the filter/sort

Caption (required) ✓
Describe/highlight what's being shown
showing filter/sort with no results

^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

Caption (required) ✓
Describe/highlight what's being shown
showing code fetching all associations

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
The query fetches all associations by joining the wishlists, users, and products tables. The username_filter is applied if it is not empty, and the results are ordered by the sort_by and sort_order parameters. The limit parameter decides how many records get displayed. The query also includes a user_count field which counts the number of users associated with each product.

#2) Show the code related to the display

Caption (required) ✓
Describe/highlight what's being shown
showing code related to displaying results.

Explanation (required) ✓
Explain in concise steps how this logically works and mention the logic for handling the username requirements
PREVIEW RESPONSE
A table is created and each row represents a user associated with a product. If there are no results available, an appropriate message is shown. If there are results available then each item is shown with the links at the end to single view or delete a product.

#3) Each record should have

Caption (required) ✓
Describe/highlight what's being shown
single view button

Explanation (required) ✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
In the form, a link is given for the user to click single view on a product. The button is a simple link that redirect the url to the view_product.php file and the product id is passed into the url to view that item on a single page.

#4) Each record should have

Caption (required) ✓
Describe/highlight what's being shown
showing username field that is clickable

Explanation (required) ✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
In the form, a link is given for the user to click on the username associated with a product. The button is a simple link that redirect the url to the profile.php file and the user id is passed into the url to view that users profile on a single page.

#5) Each record should have

Caption (required) ✓
Describe/highlight what's being shown
delete button being shown

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
This delete button is a part of a form and when it is clicked it submits the form to the file remove_from_wishlist.php which handles the deleting of the relationship between the user and the entity.

#6) Show the logic related to the count

Caption (required) ✓
Describe/highlight what's being shown
showing logic for count of all items

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
A variable $total_items is set which simply counts the amount of items in the $items variable which contains all the results of the SQL query with the count function. The variable is then displayed with the title of the page.

#7) Show the logic related to the count

Caption (required) ✓
Describe/highlight what's being shown
showing logic for count of all items on page

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
A variable $total_items is set which simply counts the amount of items in the $items variable which contains all the results of the SQL query with the count function. The variable is then displayed with the title of the page.

#8) Show the logic related to filter/sort

Caption (required) ✓
Describe/highlight what's being shown
screenshot of code related to filter/sort

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
The code sets a limit to be between 1 and 100 first, and then the set order fields are set based on the users input. The select query is then created and includes a partial match for the username. The form is then created which takes the input for each of the fields and assigns them the appropriate id so that when it is submitting they are set to the appropriate variables and the query is run based on these form inputs.

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-jed56-prod-553fb5e41cd0.herokuapp.com/Project/admin/all_user_associations.php

URL
https://it202-jed56-prod-553fb5e41cd0.herokuap

URL #2
https://github.com/jordand2003/jed56-it202-450/pull/67

URL
https://github.com/jordand2003/jed56-it202-450

+ ADD ANOTHER URL

Unassociated 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

Caption (required) ✓
Describe/highlight what's being shown
showing results

#2) Show the single view buttons/links

Caption (required) ✓
Describe/highlight what's being shown
single view buttons

#3) Show variations of the number

Caption (required) ✓
Describe/highlight what's being shown
showing variation of number of items shown

#4) Show variations of the filter/sort

Caption (required) ✓
Describe/highlight what's being shown
showing no results found

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

Caption (required) ✓
Describe/highlight what's being shown
showing code related to fetching unassociated entities

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

The code prepares a query that searches for all products that are not associated with any user by joining the wishlists table with the products table and seeing which rows on the wishlist have a null product id. It then adds the filter/sort parameters onto the query and fetches the results.

#2) Show the code related to the display

Caption (required) ✓
Describe/highlight what's being shown
showing code related to displaying results

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

PREVIEW RESPONSE

The code first checks if the items list is empty and if so it prints out that there are no results to display. If there are items though, it lists all of the product information with a link at the end that redirects the user to single view for a product if clicked.

#3) Each record should have

Caption (required) ✓
Describe/highlight what's being shown
single view button

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

PREVIEW RESPONSE

The a tag redirects the user to view_product.php which is a single view page. The product id is passed into the url and the user is taken to the page for that entity.

#4) Show the logic related to the count

Caption (required) ✓
Describe/highlight what's being shown
showing count all items

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

PREVIEW RESPONSE

After the query is prepared and ran, a variable $total_items is set which uses the count function to count all the items returned from the query. That variable is then passed in to the title of the page and displayed.

#5) Show the logic related to the count

Caption (required) ✓
Describe/highlight what's being shown
showing count all items

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
After the query is prepared and ran, a variable $total_items is set which uses the count function to count all the items returned from the query. That variable is then passed in to the title of the page and displayed.

#6) Show the logic related to filter/sort

Caption (required) ✓
Describe/highlight what's being shown
showing code for filter/sorting

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
All variables are set for the sorting and filtering and they each take in the user input from the form underneath. After the query is prepared, the sorting filters are added on to the end of the query and then executed. The form uses id for each input entry to pass the values put in by the user into the variables to be used in the query.

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-jed56-prod-553fb5e41cd0.herokuapp.com/Project/admin/unassociated_data.php

URL
https://it202-jed56-prod-553fb5e41cd0.herokuap

URL #2
https://github.com/jordand2003/jed56-it202-450/pull/68

URL
https://github.com/jordand2003/jed56-it202-450

+ 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 valid

Caption (required) ✓
Describe/highlight what's being shown
showing search form with valid data

#2) Show the results of the search

Caption (required) ✓
Describe/highlight what's being shown
showing results of the search

#3) Show the result of entities and

Caption (required) ✓
Describe/highlight what's being shown
showing entities and user being associated and unassociated

Task #2 - Points: 1

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

showing form and logic for partial match of username

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

The form takes in the input from the username serach and passes it to the user_query. This query selects the user id and user name from the users table where the name is like the input. The results of that query are then stored in the $users variable for displaying.

#2) Search form field for finding a

Caption (required) ✓

Describe/highlight what's being shown

showing form and logic for partial match of entities

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

The form takes in the input from the product serach and passes it to the entity_query. This query selects the product id and product name from the products table where the name is like the input. The results of that query are then stored in the $entities variable for displaying.

#3) Code related to getting a

Caption (required) ✓

Describe/highlight what's being shown

code related to getting max of 25 results

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 query simply uses a LIMIT on the amount of rows that can return which is set to be 25.

#4) Code that generates

Caption (required) ✓

Describe/highlight what's being shown

code for checkboxes

Explanation (required) ✓

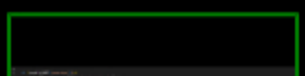
Explain in concise steps how this logically works

PREVIEW RESPONSE

The form simply includes a input field with the type of checkbox, and when clicked that value is stored in the selected_entities array with the product id of that product. The form then has a button on the button for submitting all associations which associates the entities through a POST request.

#5) Code related to submitting

#6) Code related to applying the



Caption (required) ✓
Describe/highlight what's being shown
code for submitting checkbox lists

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

PREVIEW RESPONSE

A form button is created of type submit with the name associate which will later be passed into the post request.

Caption (required) ✓
Describe/highlight what's being shown
showing code for applying 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

A post request is created that takes in the selected entities as well as the selected users that are taken from the form, and then connects to the database. Then a foreach loop is run to go through each entity and each user in the arrays and for each one creates an insert statement into the wishlists table.

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
		Add the pull request link for the branch related to this feature

☐ #2	1	Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

URL #1
https://it202-jed56-prod-553fb5e41cd0.herokuapp.com/Project/admin/associate_entities.php

URL
https://it202-jed56-prod-553fb5e41cd0.herokuap

URL #2
https://github.com/jordand2003/jed56-it202-450/pull/69

URL
https://github.com/jordand2003/jed56-it202-450

+ ADD ANOTHER URL

● Misc (1 pt.)

^COLLAPSE^

● ^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

showing project board

COLLAPSE

Task #2 - Points: 1

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

URL #1

https://github.com/users/jordand2003/projects/2/views/1

URL

https://github.com/users/jordand2003/projects/

+ ADD ANOTHER URL

COLLAPSE

Task #3 - Points: 1

Text: Talk about any issues or learnings during this assignment

Response:

During this assignment I did not have any major issues but I did learn about the henadling of associations across multiple different tables. It was not too difficult to figure out but at first it was a bit confusing handling deleting products with the many different association that I would have to go through in order to delete from a wishlist without deleting an entity in its entirey and things like that. Overall these features were not too difficult to implement but did take some time to slow down and understand in certain situations.

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

52 mins ..n/all_user_associations.php
46 mins public_html/test.php
42 mins ..dmin/associate_entities.php
36 mins ..jest/admin/edit_product.php

39 mins Feat-APIHandling
27 mins Feat-RemoveFromWishlist
25 mins prod
21 mins Feat-EditProducts

showing wakatime screenshot

End of Assignment