Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT202-452-M2024/it202-api-project-milestone-3-2024-m24/grade/mrs43

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

Submissions:

Submission Selection

1 Submission [active] 8/3/2024 9:41:04 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:

The data being fetched/entered is book data. The users look through books, then add them to their library. It is a many to many relationship because the same book can be added to everyone's library, but everyone can add all books to their libraries.

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:

The book data is associated with the user using a table with both of their ids. If the book data is updated, then the tables with only the book data are updated. The id will not change, so the user will still be associated with the same book based on the id, but the data will be updated. The user will see the new data.

Handle Data Association (1 pt.)

COLLAPSE

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
<?php
//mrs43 8-3-2024
require(__DIR__ . "/../../../lib/functions.php");
session_start();
if(isset($_GET["book_id"]) && is_logged_in()){
    $db = getDB();
    $query = "INSERT INTO `IT202-S24-UserBooks` (`user_id`, `book_id`) VALUES (:user_id, :book_id)";
    try{
        $stmt = $db->prepare($query);
        $stmt->execute([":user_id" => get_user_id(), ":book_id" => $_GET["book_id"]]);
        flash("Successfully added to library", "success");
        redirect("my_books.php");
    } catch(PDOException $e){
        if($e->errorInfo[1] === 1062){
            flash("This book isn't available", "danger");
        } else{
            flash("Unhandled error occurred", "danger");
        }
        error_log("Error adding book: " . var_export($e, true));
    } <- #12-19 catch(PDOException $e)
} <- #6-20 if(isset($_GET["book_id"]) && is_logged_in())

redirect("books.php");
```

```php
<?php if (count($results) > 0):
    <div class="card-text-muted">Total: {$total_records}</div>
    <ul>
        <?php foreach ($results as $book): ?>
            <li>
                <?php if (isset($book['username'])): ?>
                    Added by <?php echo get_pri($book['user_id']); ?> (<?php echo e($book, 'username', 'N/A'); ?>)
                <?php else:
                    Added by <?php echo e($book, 'user_id', 'N/A'); ?>
                <?php endif; ?>
                <br />
                <?php echo e($book, 'title', 'Unnamed'); ?> <?php echo
                <?php if (isset($book['created'])): ?>
                    <br />Added on <?php echo e($book, 'created', 'N/A'); ?>
                <?php endif; ?>
                <br />
                <?php if (isset($book['user_id']) && $book['user_id'] == $user_id): ?>
                    <br />
                    <?php
                    $id = isset($book['id']) ? $book['id'] : (isset($_GET['id']) ? $_GET['id'] : 0);
                    ?>
                    <br />
                    <br />Book <?php echo get_pri($_GET['book_id']); ?> <br />Is in your library
                <?php endif; ?>
                <br />
                <?php if (isset($book['user_id']) && $book['user_id'] == $user_id): ?>
                    <br />
                    <form method="POST" action="/lib/delete_book.php">
                        <input type="hidden" name="book_id" value="<?php echo e($book, 'id', 0); ?>"
                        <button type="submit" name="remove_book" class="btn btn-danger" onclick="return confirm('Are you sure you want to remove this book from your library?')">Remove
                    </form>
                <?php endif; ?>
            </li>
        <?php endif; ?>
    </ul>
</div>
<?php endif; ?>
```

```php
<!-- mrs43 8-3-2024 -->
<div class="container-fluid">
    <h2>Explore Books</h2>
    <form method="GET">
        <div class="row mb-3" style="align-items: flex-end;">
            <?php foreach ($form as $k => $v) : ?>
                <div class="col">
                    <?php render_input($v); ?>
                </div>
            <?php endforeach; ?>
        </div>
        <?php render_button(["text" => "Search", "type" => "submit", "text" => "Filter"]); ?>
        <a href="?clear" class="btn btn-secondary">Clear Filters</a>
        <?php render_result_counts(count($results), $total_records); ?>
    </form>
    <div class="row w-100 row-cols-auto row-cols-xs-1 row-cols-md-2 row-cols-lg-3 row-cols-xl-4 row-cols-xxl-5 g-4">
        <?php foreach ($results as $book) : ?>
            <div class="col">
                <?php render_book_card($book); ?>
            </div>
        <?php endforeach; ?>
        <?php if (count($results) == 0) : ?>
            <div class="col">
                No results
            </div>
        <?php endif; ?>
    </div>
</div>
```

Caption (required) ✓

Describe/highlight what's being shown

Screenshots from VSCode to show process of user adding book to library

Explanation (required) ✓

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

PREVIEW RESPONSE

My application uses the Option 1. The first screenshot is the code used to add a book to the user's library (associate the user_id and book_id in a table). The second screenshot is the code used to render the book_card, which is used on other pages. This card has the information about the book as well as a button to add/remove the book from the user's library. The thrid screenshot is an example of how the book card is used on a page. This is a screenshot of the page with all books fetched or created. Each book is represented by a book card.

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

Screenshot of association table from VSCode

Explanation (required) ✓

Describe each column/association table

PREVIEW RESPONSE

There are five columns: id, user_id, book_id, created, and modified. The id is the auto incrementing id of the relationship, so each new association is one greater. The user_id is the id of the user from the Users table. The book_id is the id of the book data from the IT202-S24-BOOKS table. This row basically says user 13 and book 51 have a relatonship, so if you have one id, you can reference this table to reference the other id. The created column is when the relationship was created and the modified column is when it was modifed.

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://mrs43-

URL

https://mrs43-prod-5f0e0e79560e.herokuapp.co

prod-5f0e0e79560e.herokuapp.com/project/books.php

URL #2
https://mrs43-prod-5f0e0e79560e.herokuapp.com/project/new_books.php

URL
https://mrs43-prod-5f0e0e79560e.herokuapp.co

URL #3
https://github.com/m-sansone/mrs43-IT202-452/pull/88

URL
https://github.com/m-sansone/mrs43-IT202-452

URL #4
https://github.com/m-sansone/mrs43-IT202-452/pull/95/files

URL
https://github.com/m-sansone/mrs43-IT202-452

URL #5
https://github.com/m-sansone/mrs43-IT202-452/pull/99

URL
https://github.com/m-sansone/mrs43-IT202-452

URL #6
https://github.com/m-sansone/mrs43-IT202-452/pull/103

URL
https://github.com/m-sansone/mrs43-IT202-452

+ ADD ANOTHER URL

Current User's Association Page (2 pts.)

COLLAPSE

EXPAND

Task #1 - Points: 1
Text: Screenshots of this page

EXPAND

Task #2 - Points: 1
Text: Screenshot the code

EXPAND

Task #3 - Points: 1
Text: Add related links

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

COLLAPSE

EXPAND

Task #1 - Points: 1
Text: Screenshots of this page

EXPAND

Task #2 - Points: 1
Text: Screenshot the code

EXPAND

Task #3 - Points: 1

Text: Add related links

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

Caption (required) ✓

Describe/highlight what's being shown

Screenshot of non-associated books with summaries

#2) Show the single view

Caption (required) ✓

Describe/highlight what's being shown

The view button to the page with book details is on each book card

#3) Show variations of the number

Caption (required) ✓

Describe/highlight what's being shown

Filtering items changes number of books on page

#4) Show variations of the

Caption (required) ✓

Describe/highlight what's being shown

Varied results using filters and sort/order

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 all unassociated entities (including the query)

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

unassociated entities (including the query)

Caption (required) ✓

Describe/highlight what's being shown

Code used to get all books not associated with users

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

A form is setup with input fields for filtering. A query is created to select books that are not associated with users in IT202-S24-UserBooks. The parameters are adjusted accoring to the filters provided by the user in the input fields. This includes the username, title and language filter fields, the sort and order fields, and limit field. The query is then executed and the results are fetched and stored.

the results

Caption (required) ✓

Describe/highlight what's being shown

Code used to display the summary of books not associated with users

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

Each book fetched is displayed in a book card, which displays basic information about the book. This includes the cover, title, page count, language, user count, and usernames associated with the book. The usernames can be clicked to view the user's public profile. There are also buttons to view the book details and remove the book from all libraries. If there are no results, an appropriate message is displayed.

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

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

Caption (required) ✓

Describe/highlight what's being shown

Code for button to display book details

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

A hyperlink is created to the page that shows book details with the appropriate book id. The book id is retrieved using a dynamic safer echo call. The hyperlink is displayed as a button using Bootstrap. The button has the text "View" displayed.

Caption (required) ✓

Describe/highlight what's being shown

Code used to find total count of books not associated with users

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

The code creates a query to get the total count of books not associated with users using IT202-S24-UserBooks. This is then passed to get_total_count, which sanitizes the data and builds a query that retrieves the number of rows with unique books. The query is then executed and the result is fetched while handling errors.

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

Caption (required) ✓

Describe/highlight what's being shown

Code to get number of associated books after filters are applied

#6) Show the logic related to filter/sort (should include a partial match for username) (limit should be constrained

Caption (required) ✓

Describe/highlight what's being shown

Code used to filter books that are not associated with users

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

A query is prepared with all of the filters to be used on IT202-S24-UserBooks stored as parameters. The query is executed, then the results are counted, while errors are handled.

Explanation (required) ✓

Explain in concise steps how this logically works

PREVIEW RESPONSE

A form is created for the user to input the filters they want to apply to the nonassociated books. An SQL query is built to retrieve the unique books that are not associated with users. The filters the user input are then stored as parameters. The query is executed and the results are stored, while errors are handled.

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://mrs43-prod-5f0e0e79560e.herokuapp.com/Project/new_books.php

URL
https://mrs43-prod-5f0e0e79560e.herokuapp.co

URL #2
https://github.com/m-sansone/mrs43-IT202-452/pull/92

URL
https://github.com/m-sansone/mrs43-IT202-452

URL #3
https://github.com/m-sansone/mrs43-IT202-452/pull/93

URL
https://github.com/m-sansone/mrs43-IT202-452

URL #4
https://github.com/m-sansone/mrs43-IT202-452/pull/99

URL
https://github.com/m-sansone/mrs43-IT202-452

URL #5
https://github.com/m-sansone/mrs43-IT202-452/pull/103

URL
https://github.com/m-sansone/mrs43-IT202-452

+ ADD ANOTHER URL

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

COLLAPSE

EXPAND

Task #1 - Points: 1

Text: Screenshots of the page

EXPAND

Task #2 - Points: 1

Text: Screenshots of the code

EXPAND

Task #3 - Points: 1

Text: Add related links

Misc (1 pt.)

COLLAPSE

COLLAPSE

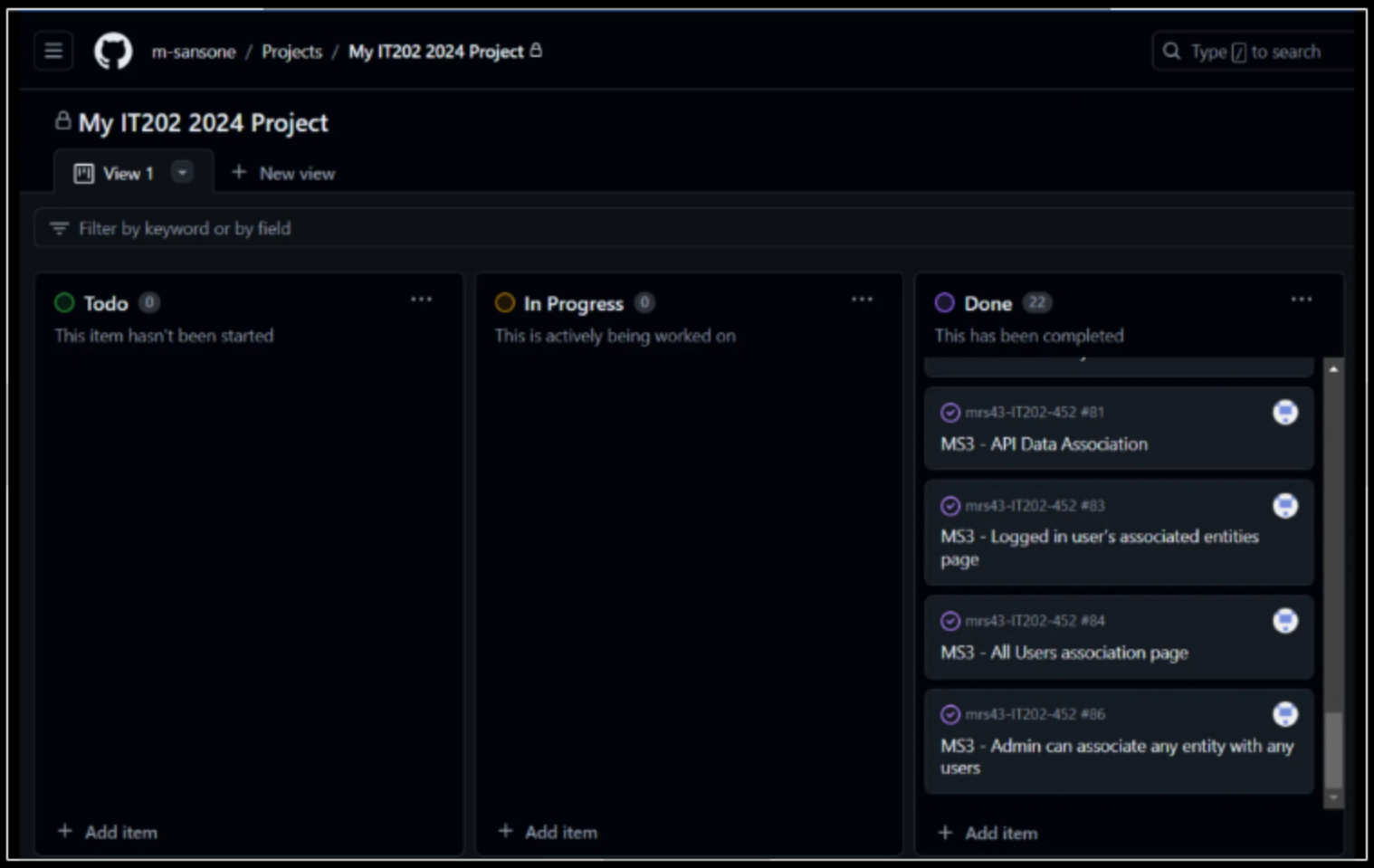
Task #1 - Points: 1

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

Task Screenshots:

Gallery Style: Large View

Small Medium Large



Screenshot of project board on GitHub

COLLAPSE

Task #2 - Points: 1

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

URL #1
https://github.com/users/m-sansone/projects/2/views/1

URL
https://github.com/users/m-sansone/projects/2

+ ADD ANOTHER URL

COLLAPSE

Task #3 - Points: 1

Text: Talk about any issues or learnings during this assignment

Response:

I ran into a variety of issues ranging from small typos that were difficult to find but easy to fix, to taking longer to work on the project because I was sick. I also kept having problems with everything related to my API, so I have learned a lot about APIs due to spending hours trying to solve my errors. The most important thing I learned with this assignment was how to utilize git and GitHub effectively, which I plan to do from the beginning for my future projects.

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

Screenshot of WakaTime from VSCode

The number in VSCode seemed very low, so I found the total time on WakaTime's website

End of Assignment