Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT202-450-M2024/it202-api-project-milestone-2-2024-m24/grade/jed56

IT202-450-M2024 - [IT202] API Project Milestone 2 2024 M24

Submissions:

Submission Selection

1 Submission [active] 7/25/2024 6:48:12 PM

Instructions

COLLAPSE

Implement the Milestone 2 features from the project's proposal document: https://docs.google.com/document/d/1XE96a8DQ52Vp49XACBDTNCq0xYDt3kF29cO88EWVwfo/view Make sure you add your ucid/date as code comments where code changes are done All code changes should reach the Milestone2 branch Create a pull request from Milestone2 to dev and keep it open until you get the output PDF from this assignment. Gather the evidence of feature completion based on the below tasks. Once finished, get the output PDF and copy/move it to your repository folder on your local machine. Run the necessary git add, commit, and push steps to move it to GitHub Complete the pull request that was opened earlier Create and merge a pull request from dev to prod Upload the same output PDF to Canvas

Branch name: Milestone2

Tasks: 23 Points: 10.00

Define Appropriate Tables for Data (1 pt.)

COLLAPSE

COLLAPSE

Task #1 - Points: 1

Text: Screenshots of Table SQL

Checklist *The checkboxes are for your own tracking

#	Points	Details

☐ #1	1	Table(s) should have the 3 core columns we'll always be using (id, created, modified) plus additional columns for the incoming API data
☐ #2	1	Columns should be logical and thought out (not valid to have a single field of JSON data or similar

#1) Screenshot of the table (you can duplicate this subtask

Caption (required) ✓
Describe/highlight what's being shown
Screenshot of products table

Explanation (required) ✓
Explain the columns and what data they represent (briefly), also note any normalization that may have been necessary

PREVIEW RESPONSE

ID represents the unique id for the row in the table. Created and modified show the date and time of creation and modification of data. Product id is the unique product id from the API. Product name is simply the name of the product. Current price of the product, original price of the product are self explanatory. Discount percentage is the percentage discount the item currenlty has. The data source column tells whether the product is from the API or manually entered. The image url is the source of the photo for the item.

COLLAPSE

Task #2 - Points: 1

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

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

URL #1
https://github.com/jordand2003/jed56-it202-450/pull/45

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

+ ADD ANOTHER URL

Data Creation Page (2 pts.)

COLLAPSE

COLLAPSE

Task #1 - Points: 1

Text: Screenshots of the creation page

Details:
Heroku dev url must be visible in all relevant screenshots

#1) Show potentially valid data

Caption (required) ✓
Describe/highlight what's being shown
showing valid data in creation page

#2) Show how the API data is

Caption (required) ✓
Describe/highlight what's being shown
showing how API data is fetched

Explanation (required)
✓
Briefly explain the code

PREVIEW RESPONSE

#3) Show examples of validation

Caption (required) ✓
Describe/highlight what's being shown
showing some validation messages

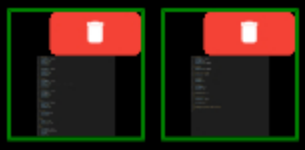
#4) Show an example of successful

Caption (required) ✓
Describe/highlight what's being shown
showing success creation of product

PREVIEW RESPONSE

The code gets product data from the Amazon Data API and store it in the database. First it checks if the API key is set, then makes a request to get deals from Amazon. For each deal it fetches detailed product information. If the product is already in the database it updates it. If not it adds a new row into the database. The fetch_api_data.php file has a form with a button for admins to trigger the fetch from the API.

#5) Design/Style should be



Caption (required) ✓

Describe/highlight what's being shown

showing custom css styling

Explanation (required) ✓

Briefly explain your design choices

PREVIEW RESPONSE

The CSS styles alerts with different colors. It also styles nav items with inline block display, colors, and text decorations. The input fields have rounded borders. The body has a

white background, a specific font, and padding. Tables are styled with full width and margin. Table headers have bold text and a solid bottom border. The table rows alternate colors for better readability. Images inside table cells are resized and links inside table cells are colored blue and have no underline unless you hover.

COLLAPSE

Task #2 - Points: 1

Text: Screenshots of creation page code

Details:
Include ucid/date comments for each code screenshot

#1) Form should have correct data

Caption (required) ✓
Describe/highlight what's being shown
Showing with correct data types

Explanation (required)
✓
Briefly talk about each field type and why it was chosen

PREVIEW RESPONSE

#2) Form should have correct

Caption (required) ✓
Describe/highlight what's being shown
All form validation code

Explanation (required)
✓
Briefly explain the validations

#3) Successful creation

Caption (required) ✓
Describe/highlight what's being shown
Showing user friendly message code

Explanation (required)
✓
Explain how duplicate/existing data is handled

#4) Any errors should have user-

Caption (required) ✓
Describe/highlight what's being shown
showing all error messages

Explanation (required)
✓
Briefly describe the scenarios

PREVIEW RESPONSE

The javascript validation checks for the required

PREVIEW RESPONSE

THe product id field was chosen to uniquely identify each product. The product name is chosen to describe what the product is. The current price field represents what the current price of the item is. The original price represents what the original price of the product was before the discount. The discount percentage is to show the percentage off the item is. The image url is used to show a photo of the item.

PREVIEW RESPONSE

The HTML validation makes sure all fields are required, uses the right input types, has a valid range. The JavaScript validation gives instant feedback and makes sure fields are filled correctly and URLs are valid before form submission. It checks for empty values, valid numbers, and proper URLs. The PHP validation makes sure the data is good. It checks the required fields, makes sure numerical values are within valid ranges, and confirms URLs.

PREVIEW RESPONSE

The code handles duplicate data by using a try-catch block. If the INSERT fails because of a duplicate product ID it will catch the exception and flash a warning message. If any other error occurs it flashes a generic error message.

fields, valid numeric values, and valid urls before form submission. The HTML validation makes sure validation with attributes like required, type, min, and max to enforce correct input types and ranges and displays appropriate error messages. The PHP validation validates required fields, numeric values, and URLs, and handles duplicate entries with error messages.

#5) Include the form/process

Caption (required) ✓
Describe/highlight what's being shown
showing form/process for fetching API data

Explanation (required) ✓
Briefly explain the steps (include how duplicates are handled)

PREVIEW RESPONSE

The code fetches data from the API and processes each product to insert it into the

#6) Include some indicator

Caption (required) ✓
Describe/highlight what's being shown
indicators of API data and Manual data

Explanation (required) ✓
Briefly mention what the indicator is (i.e., api_id if the API has ids or is_api as a boolean-like column, etc)

PREVIEW RESPONSE

#7) Include any other rules like role

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

Explanation (required) ✓
Briefly explain the logic/reasoning

PREVIEW RESPONSE

The reasoning that this role check is being done is because the only person that should be able to create new data manually are admins.

database. For each product it checks if the product already exists in the database by looking up the product_id. If the product exists it updates the existing record, if not then it inserts a new record.

The indicator used is the data_source column. This column says whether the data is custom data or API data by putting the values 'manual' for manually added data and 'api' for data fetched from the API. When a product is inserted or updated the appropriate value is put in the data_source column to show where the data came from. When the data is pulled from the API the 'data_source' column is automatically set to api.

Regular users cannot have access to this feature.

COLLAPSE

Task #3 - Points: 1

Text: Screenshot of records from DB

#1) Show at least one record fetched from the API

Caption (required) ✓
Describe/highlight what's being shown (note what differs from a custom record)
showing records from DB that were fetched from API

#2) Show at least one record created via the creation form

Caption (required) ✓
Describe/highlight what's being shown (note what differs from the API record)
showing a record created from creation form

COLLAPSE

Task #4 - Points: 1

Text: Add related links

Checklist

*The checkboxes are for your own tracking

#	Points	Details
☐ #1	1	Include the heroku prod link for this page
☐ #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-dev-4e8bc86ff059.herokuapp.com/Project/admin/data_creation.php

URL
https://it202-jed56-dev-4e8bc86ff059.herokuapp

URL #2
https://github.com/jordand2003/jed56-it202-450/pull/46

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

URL #3
https://github.com/jordand2003/jed56-it202-450/pull/53

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

+ ADD ANOTHER URL

Data List Page (many entities) (2 pts.)

^COLLAPSE^

^COLLAPSE^

Task #1 - Points: 1

Text: Screenshots of the list page

Details:
Heroku dev url must be visible in all relevant screenshots

#1) Show the page of your entities listed

Caption (required) ✓
Describe/highlight what's being shown
showing list data page from heroku dev

#2) Show the filter/sort form based

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

#3) Demonstrate a few varied

Caption (required) ✓
Describe/highlight what's being shown
showing some filter searches

#4) Demonstrate a filter that

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

Explanation (required) ✓

Briefly describe the page

PREVIEW RESPONSE

The page shows a list of all the data entered manually and pulled from the API. The columns show each column from the database with a photo to represent each product. The CSS makes a nice table structure with light grey to make the list easier on the eyes.

Explanation (required) ✓

Briefly mention what's available to the user

PREVIEW RESPONSE

This form allows a user to filter the results of all the amazon products. They can search based on the product title, set a minimum price, a maximum price, and sort the products in ascending or desacending order based on list price or percentage off. The limit section allows the user to limit the number of results that get displayed on the page.

#5) Each list item should have a link of

Caption (required) ✓

Describe/highlight what's being shown

showing edit and delete available to admin and not regular user

Explanation (required) ✓

Mention which users can interact with the view, edit, and delete links

PREVIEW RESPONSE

The only users that can interact with the edit and delete buttons are admins. Regulars users

#6) Each list item should have a

Caption (required) ✓

Describe/highlight what's being shown

screenshot clearly showing summary of entity in list

#7) Design/Style should be

Caption (required) ✓

Describe/highlight what's being shown

custom css styling for table

Explanation (required) ✓

Briefly explain your design choices

PREVIEW RESPONSE

The css makes the table easier to look at on the eyes. It organizes the data into a clear table with some light grey and white to clearly

that are registered will not be able to view any of those buttons besides the single view button as they do not have access to these features.

distinguish new list items. Columns are also clearly spaced and separated with the text always centered in the center of the column.

COLLAPSE

Task #2 - Points: 1

Text: Screenshots of the list page code

Details:
Include ucid/date comments for each code screenshot

#1) Show the filter/sort form

Caption (required) ✓
Describe/highlight what's being shown
showing filter/sort form

Explanation (required) ✓
Briefly explain how the code works (i.e, some stuff may be dynamic)

PREVIEW RESPONSE

The code is for the form that lets users search for products. It has input fields "Product Name" for typing the product name, "Min Price" and "Max Price" for setting the minimum and maximum prices, and a

#2) Show the DB query and how the

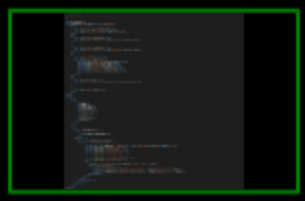
Caption (required) ✓
Describe/highlight what's being shown
showing DB query and how filter is handled

Explanation (required) ✓
Briefly explain the related logic

PREVIEW RESPONSE

The code processes the form from to search for products in the database. It starts with a SQL query to select product details. Then it checks if the user has entered a product name, minimum price,

#3) Show how the output is



Caption (required) ✓
Describe/highlight what's being shown
showing how the output generated

Explanation (required) ✓
Briefly explain the related logic

PREVIEW RESPONSE

The code creates a search form and shows the table of products. The form lets users enter a product name, minimum and maximum prices, sort order, and a limit for the number of results. When the form

#4) Show any restrictions

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

Explanation (required) ✓
Briefly explain the logic/reasoning

PREVIEW RESPONSE

The reasoning for this role guard is that the only time a user should be able to view the products is when they are logged in, if not they cannot see this page.

"Sort By" dropdown menu to sort the results in an ascending or dfescending order by price of percentage discount. There's also a "Limit" field to set the maximum number of results to show on the list at one time. When the user fills in these fields and clicks the "Search" button it will find products that match the search criteria. The form also has validation to check all inputs before submitting the form.

maximum price, or sleected a sorting option and limit for the number of results. Depending on the users entry it updates the SQL query to include these filters. Then it runs the query, fetches the results, and stores them in the $products variable. If no products are found or there's an error it displays a warning or error message.

is submitted it searches for products that match the criteria. The table shows the product image, ID, name, current price, original price, discount, data source, and actions like viewing, editing, or deleting the product. If there are no products found it displays a message saying "No products"

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 this page
☐ #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-dev-4e8bc86ff059.herokuapp.com/Project/list_data.php

URL
https://it202-jed56-dev-4e8bc86ff059.herokuapp

URL #2
https://github.com/jordand2003/jed56-it202-450/pull/47

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

+ ADD ANOTHER URL

View Details Page (single entity) (1 pt.)

COLLAPSE

Task #1 - Points: 1

Text: Screenshots of the details page

COLLAPSE

Details:
Heroku dev url must be visible in all relevant screenshots

#1) Entity should be fetch by id

Caption (required) ✓
Describe/highlight what's being shown
showing entity fetched by id in url

#2) A missing id should

Caption (required) ✓
Describe/highlight what's being shown
showing message for missing id

#3) Design/Style should be

Caption (required) ✓
Describe/highlight what's being shown
All product information with different css styling

Explanation (required)
✓
Briefly explain your design choices
PREVIEW RESPONSE
In this design it is pretty simple. All of the product details are aligned to the left side of the page in a vertical manner. The column titles are in bold and the values are in regular text. The photo sits above all the item details for basic reasoning.

#4) Data shown should be

Caption (required) ✓
Describe/highlight what's being shown
data more detailed than view

Explanation (required)
✓
Briefly explain the details shown and how it differs from the list view
PREVIEW RESPONSE
The data details differs from the list view because it is in vertical view instead of horizontal view and it also shows the data and time of creation and modification which the list view does not show.

#5) There should be a link to edit

#6) There should be a link to delete

Caption (required) ✓
Describe/highlight what's being shown
link to edit entity for admin

Caption (required) ✓
Describe/highlight what's being shown
link to delete entity for admin

COLLAPSE

Task #2 - Points: 1

Text: Screenshots of the details page code

Details:
Include ucid/date comments for each code screenshot

#1) Show how id is fetched

Caption (required) ✓
Describe/highlight what's being shown
showing how id is fetched

Explanation (required) ✓
Briefly explain the logic and how it's handled when the property is missing or not "valid"

PREVIEW RESPONSE

This PHP code checks if a product ID is provided in the URL. If the product ID is missing it shows a warning message saying "Invalid product ID" and redirects the user to "list_data.php".

#2) Show the DB query to get the

Caption (required) ✓
Describe/highlight what's being shown
showing DB query to get record

Explanation (required) ✓
Briefly explain the logic

PREVIEW RESPONSE

The code makes a query to select the product details like the name, price, discount, and image. Then it tries to run this query with the product ID. If the product is found then it fetches the details. If the product isn't found or if there's a problem with the database, it shows

#3) Show the code related to presenting

Caption (required) ✓
Describe/highlight what's being shown
showing code to present data

Explanation (required) ✓
Briefly explain the logic

PREVIEW RESPONSE

This PHP code shows the details of a product on a webpage. If a product is found it displays information like the product ID, name, current price, original price, discount percentage, creation date, and modification date. It also shows an image of the product. If

the database, it shows an error message and redirects the user to "list_data.php".

image of the product. If the user is an admin, they get links to edit or delete the product. If no product is found it shows a message saying "Product details not available."

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 this page
☐ #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-dev-4e8bc86ff059.herokuapp.com/Project/view_product.php?id=B0D651GQXC

URL
https://it202-jed56-dev-4e8bc86ff059.herokuapp

URL #2
https://github.com/jordand2003/jed56-it202-450/pull/48

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

+ ADD ANOTHER URL

Edit Data Page (2 pts.)

COLLAPSE

Task #1 - Points: 1

Text: Screenshots of the edit page

Details:
Heroku dev url must be visible in all relevant screenshots

#1) Show before and after

#2) Show examples of validation

#3) Show an example of successful

#4) Design/Style should be

Caption (required) ✓
Describe/highlight what's being shown
showing before and after editing data

Explanation (required) ✓
Briefly explain what you changed

PREVIEW RESPONSE

I changed the title of the product, instead of it being so long and detailed I just changed it to Galaxy Z fold to make it shorter and easier to read. It shows the successful message with the new name of the product.

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

Caption (required) ✓
Describe/highlight what's being shown
showing successful edit message

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

Explanation (required) ✓
Briefly explain your design choices

PREVIEW RESPONSE

The form has rounded boxes rather than the basic html boxes that are defaultly set. The submission button says Update product and is also rounded with a grey background. That is all the simple css styling i did.

COLLAPSE

Task #2 - Points: 1

Text: Screenshots of edit page code

Details:
Include ucid/date comments for each code screenshot

#1) Form should have correct data

Caption (required) ✓

#2) Form should have correct

#3) Successful edit should

Caption (required) ✓

#4) Any errors should have user-

Describe/highlight what's being shown
showing form with correct data types

Explanation (required) ✓

Briefly explain the data type choices

PREVIEW RESPONSE

In this form different input types are used based on the data they collect. The "Product ID" and "Product Name" fields use text input because they contain words. The "Current Price," "Original Price," and "Discount Percentage" fields use number input. The "Image URL" field uses url input to make sure a valid url is entered. The "Data Source" field is also a text input but it is put as readonly because it cannot be changed.

Caption (required) ✓

Describe/highlight what's being shown
form has correct validation for each field

Explanation (required) ✓

Briefly explain the validations

PREVIEW RESPONSE

The code has client side and server side validations to make sure everything is filled in correctly. On the client side it checks if the "Product Name" is filled, makes sure "Current Price" and "Original Price" are positive numbers, and makes sure that the "Discount Percentage" is between 0 and 100, and displays error messages if anything fails. On the server side similar checks are done by making sure that the "Product Name" is present, "Current Price" and "Original Price" are positive, and "Discount Percentage" is in the range. Flash messages are shown if the server side validation does not pass.

Describe/highlight what's being shown
successful edit user friendly message

Explanation (required) ✓

Provide a high-level step-by-step of how the fetching of the record, populating the form, and the update works and gets changed in your DB

PREVIEW RESPONSE

When you want to edit a product the code first fetches the product ID from the URL and runs a query to get the product details from the database using the ID. If the product is found it shows a form on the page with the current details of the product. You can update the information in the form and submit it. The code checks the data to make sure it's valid and if everything is correct it runs an update query to change the product details in the database. Then a success message is shown to confirm the product has been updated.

Caption (required) ✓

Describe/highlight what's being shown
showing errors friendly messages

Explanation (required) ✓

Briefly explain the possible errors

PREVIEW RESPONSE

There are a few possible errors that can happen with this code. If the product name is empty it shows an error saying "Product name is required." If the current price or original price is not a number or is less than or equal to zero then it will show an error that the price must be a positive number. If the discount percentage is not a number or is less than 0 or more than 100 it will show an error that the discount percentage must be between 0 and 100. If the database query fails it will show a general error message saying "An error occurred, please try again."

#5) Include any other rules like role

Caption (required) ✓
Describe/highlight what's being shown
showing role guard for admin to edit data

Explanation (required) ✓
Briefly explain the logic/reasoning

PREVIEW RESPONSE

The code first checks if the user is logged in and if not it shows a message saying "You must be logged in to view this page" and sends them to the login page. Then it checks if the user has the role of admin to make sure that only admins can access and edit the product page.

COLLAPSE

Task #3 - Points: 1

Text: Screenshot of records from DB

#1) Show a before and after

Caption (required) ✓
Describe/highlight what's being shown
before and after of record being updated

Explanation (required)

Explanation (required)

✓

Explain what differs

PREVIEW RESPONSE

What I changed was the original pricve and discount price of the item which also changes the discount percentage.

COLLAPSE

Task #4 - Points: 1

Text: Add related links

Checklist *The checkbox are for your own tracking

#	Points	Details
☐ #1	1	Include the heroku prod link for this page
☐ #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-dev-4e8bc86ff059.herokuapp.com/Project/admin/edit_product.php?id=1

URL
https://it202-jed56-dev-4e8bc86ff059.herokuapp

URL #2
https://github.com/jordand2003/jed56-it202-450/pull/49

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

+ ADD ANOTHER URL

Delete Handling (1 pt.)

COLLAPSE

Task #1 - Points: 1

Text: Screenshots related to delete

Details:
Heroku dev url must be visible in all relevant screenshots

Include ucid/date comments for each code screenshot

#1) Show the

#2) Show

#3) Show the

#4) Explain the

success message of a

Caption (required) ✓
Describe/highlight what's being shown
showing successfully deleted item

any error messages of

Caption (required) ✓
Describe/highlight what's being shown
error message not being able to delete product due to id not being found

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
Since the id is not being passed there is no data to delete and error message is displayed saying the product is not found because there is no product in the database that matches that id and therefore cannot be deleted.

code related to the delete

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

Explanation (required)
✓
Explain in concise steps how this logically works
PREVIEW RESPONSE
The code checks if the user is logged in and if they are an admin. Then it fetches the product ID from the URL and retrieves the product details from the database. If the product is not found or the user is not an admin it shows an error message and redirects them to the product list page. If everything coimes back good then it executes a query to delete the product from the database. After successfully deleting the product it shows a success message and redirects the user back to the list of products page.

delete logic

Explanation (required)
✓
Is it a soft or hard delete? Are there any necessary roles or restrictions? (can only delete their data, can only be done by admin, etc)?
PREVIEW RESPONSE
This is a hard delete meaning it completely removes the product from the database. Only an admin can delete a product, regular users cannot. This makes sure that only admins can permanently remove data from the database.

COLLAPSE

Task #2 - Points: 1
Text: Screenshots of the data

#1) Show a before and after

Caption (required) ✓
Describe/highlight what's being shown (note precisely what changed)
before and after of deleting toothbrush

COLLAPSE

Task #3 - Points: 1
Text: Add the pull request link for the branch related to this feature

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

URL #1
https://github.com/jordand2003/jed56-it202-450/pull/50

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

+ ADD ANOTHER URL

Misc (1 pt.)

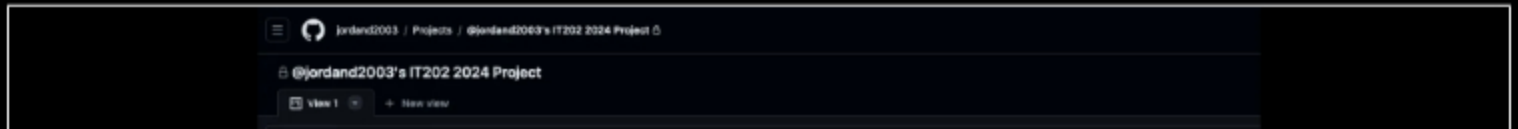
COLLAPSE

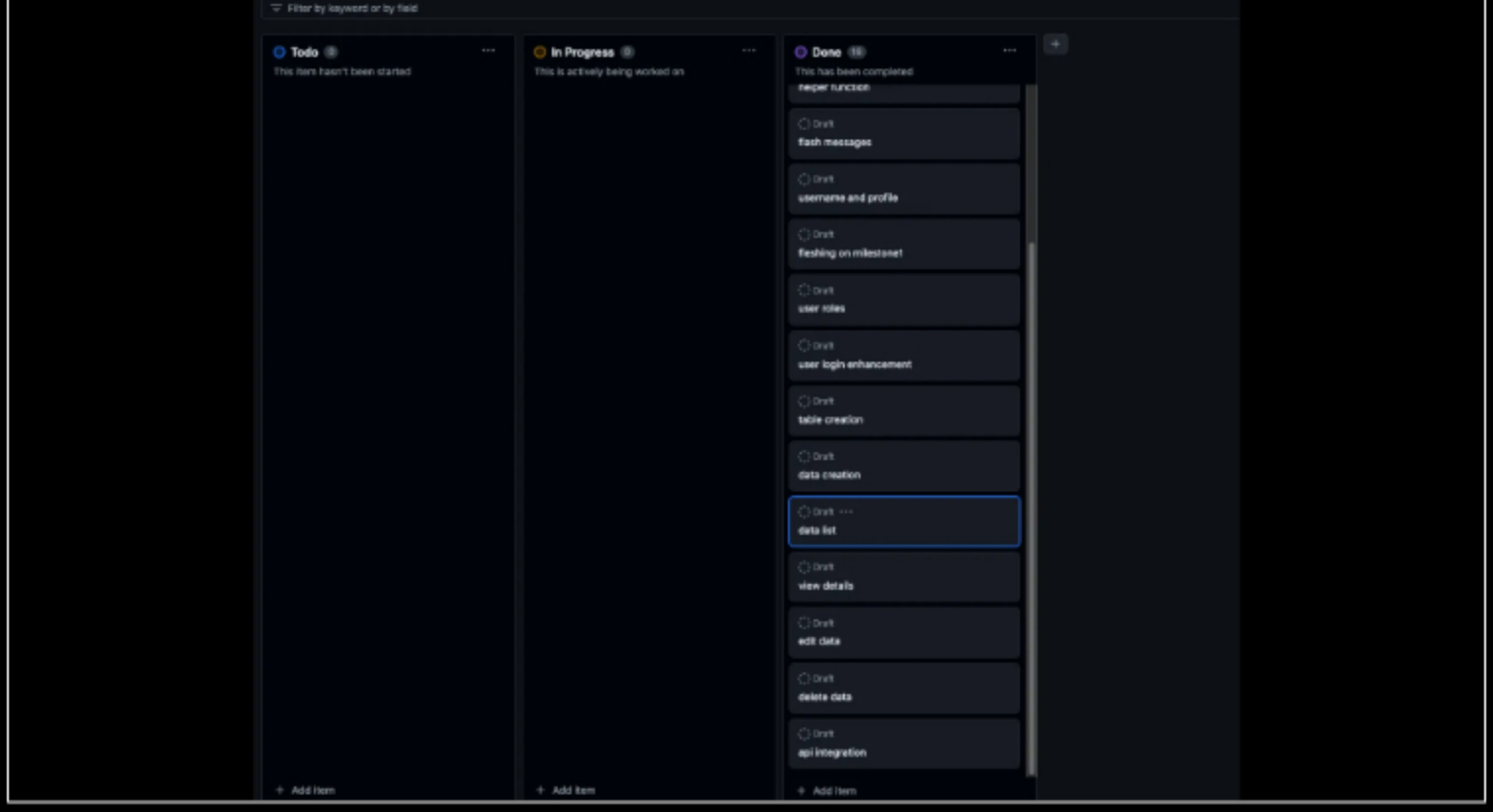
Task #1 - Points: 1
Text: Screenshot of your project board from GitHub (tasks should be in the proper column)

Task Screenshots:

Gallery Style: Large View

Small Medium Large





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 the project it started off going pretty well. That was until I got into handling the API data. When integrating the API, I had a very big issue because there were two different endpoints that I had to deal with. Originally I was just using the "Deals" endpoint, but eventually realized that this endpoint did not have access to the list price or original price of the item, and only had access to the percentage discount. I then realized that this data was in the "Deals Products" endpoint which had a completely different json response data. Because of this issue it took me many hours to figure out, but eventually I realized I had to use both endpoints. I then ran into an issue that went undiscovered for another few hours until I was able to error log the response variable and found that the Deals Products endpoints required a product id to fetch the data. This created. awhole extra headache and I realized that not only did I have to go through the Deals endpoint AND the Deals Products endpoint, but I would also have to capture the deal_id from the Deals endpoint and pass it in to the Deals Products endpoiint in order to successfully eb able to capture all of the data I needed. This issue took me about 3 hours coding time to figure out and completely fix. That was the biggest issue I faced but other than this everything else went smoothly

faced but other than this everything else went smoothly.

COLLAPSE

Task #4 - Points: 1

Text: WakaTime Screenshot

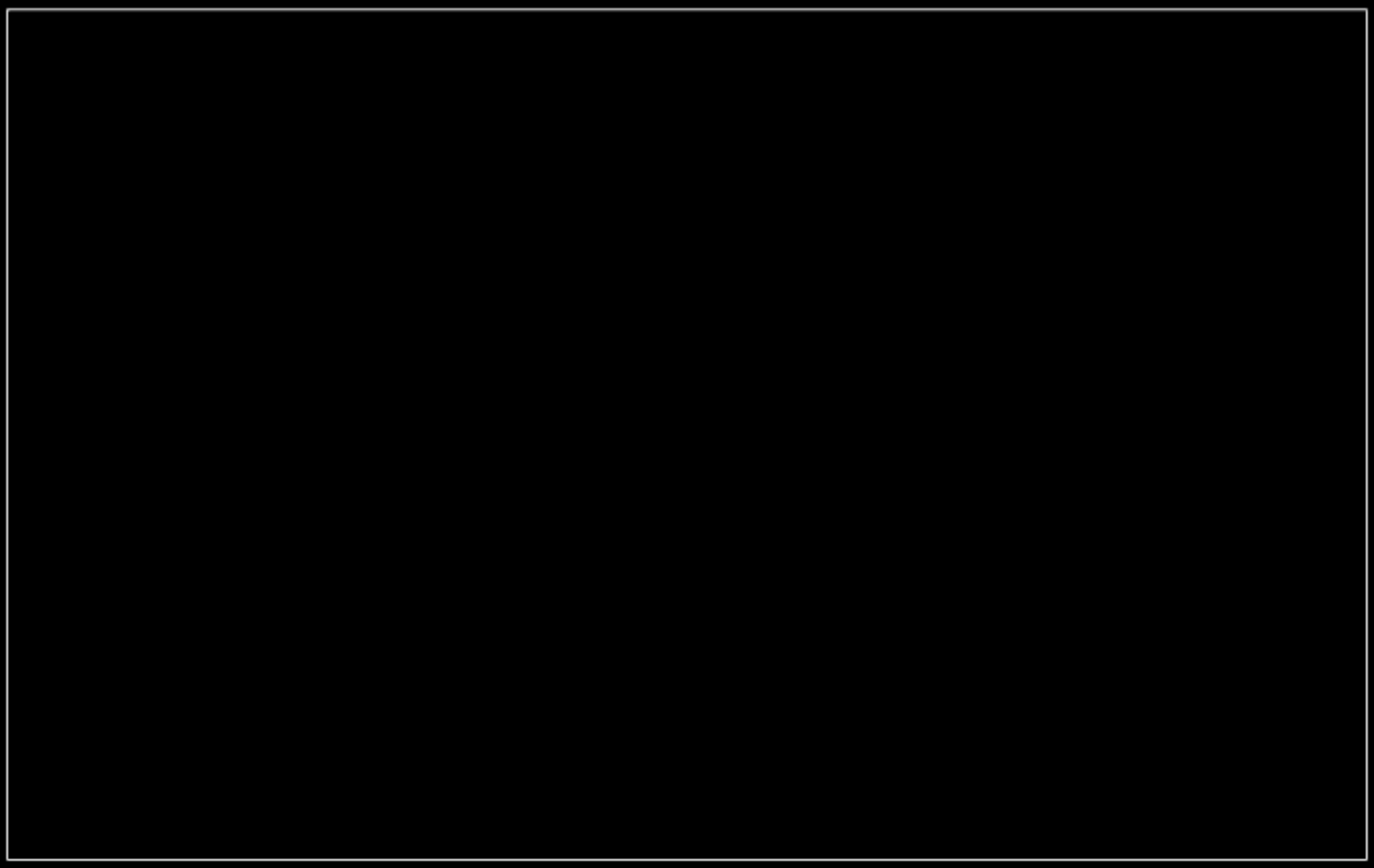
Details:

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



showing wakatime screenshot

End of Assignment