Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT202-007-F2024/it202-api-project-milestone-2-2024-m24/grade/db624

Course: IT202-007-F2024
Assigment: [IT202] API Project Milestone 2 2024 M24
Student: Datha V. (db624)

Submissions:

Submission Selection

1 Submission [submitted] 11/29/2024 10:48:33 AM

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

Group

Group: Define Appropriate Tables for Data
Tasks: 2
Points: 1

COLLAPSE

Task

Group: Define Appropriate Tables for Data
Task #1: Screenshots of Table SQL
Weight: ~50%

Points: ~0.50

COLLAPSE

Checklist

*The checkboxes are for your own tracking

#	Details
#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	Columns should be logical and thought out (not valid to have a single field of JSON data or similar

Sub-Task

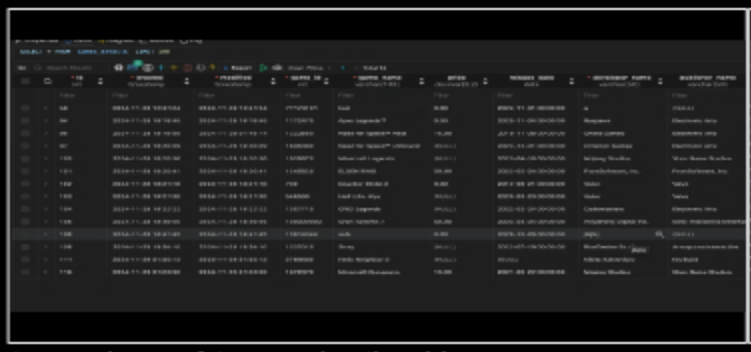
100%

Group: Define Appropriate Tables for Data
Task #1: Screenshots of Table SQL
Sub Task #1: Screenshot of the table (you can duplicate this subtask as needed)

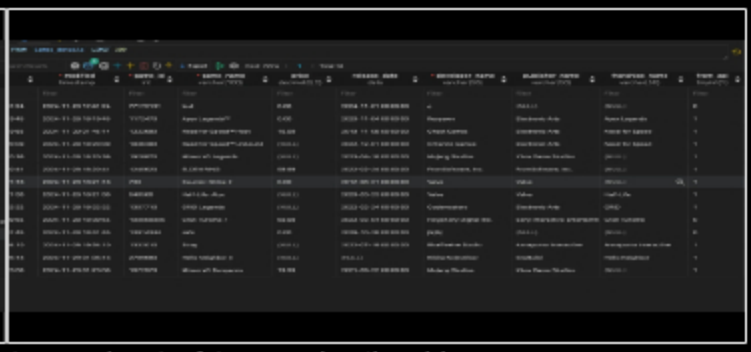
Task Screenshots

Gallery Style: 2 Columns

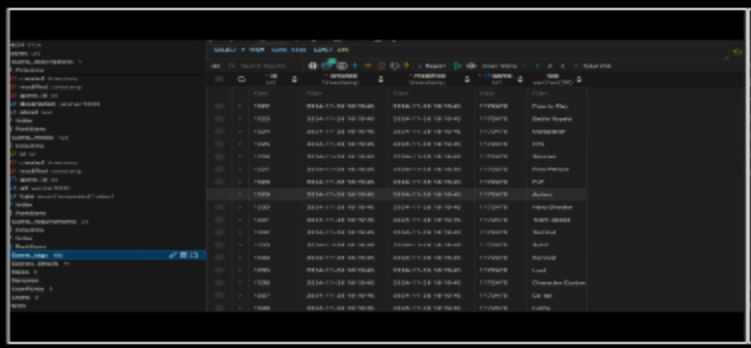
4 2 1



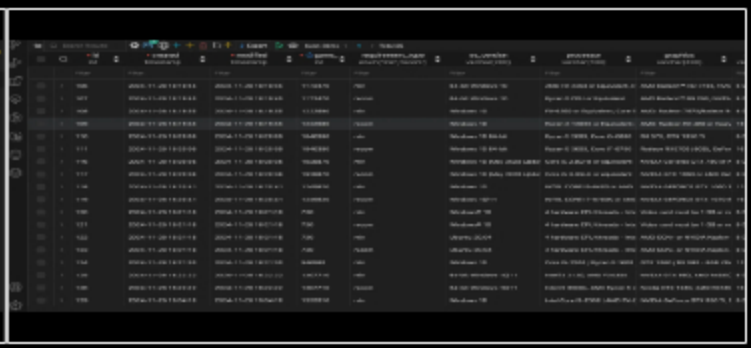
Screenshot 1 of Games_details table



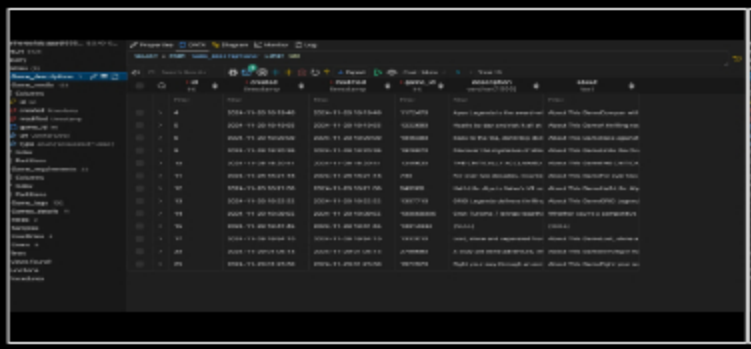
Screenshot 2 of Games_details table



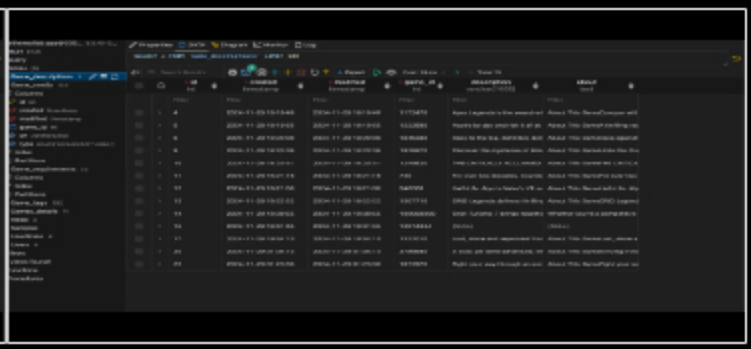
Screenshot of Game_tags table



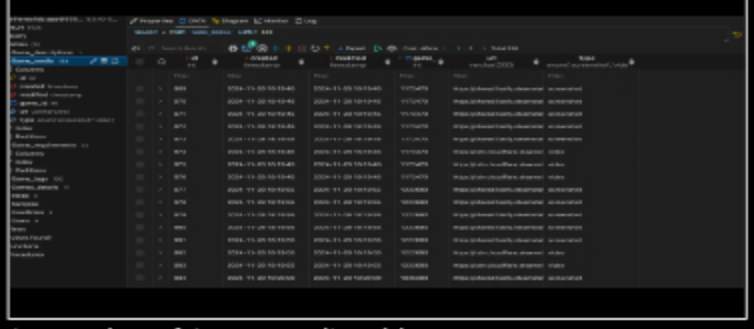
Screenshot 1 of Games_requirements table



Screenshot 2 of Games_requirements table



Screenshot of Game_descriptions table



Screenshot of Game_media table

Caption(s) (required) ✓

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

Task Response Prompt

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

Response:

Games_details game_id - this field stores the game id of the game obtained from steam API

game_name - this field stores the name of the game

price - this field stores the price of the game, if the price is not given, it is stored as null, if the result is "free to play", it is stored as 0.00.

The API returns price with $, so the $ is stripped from price. The data is stored in decimal format with 2 digits after decimal point.

release_date- stores the release date of the game in date format yyyy-mm-dd. If the game's release date is to be announced, date field is left null.

developer_name, publisher_name, franchise_name- simply a varchar column that stores the names of the developers, publisher, and the game franchise.

from_api - this column is a tinyiny column that specifies whether the record was manually inserted or called from API, 1 for API, 0 for manual

Game_media

game_id - A foreign key column connecting a record to an associated game

url - A varchar field storing the url for the screenshot/video

type- enum field that specifies whether the url is for a screenshot or video

Game_tags

game_id - Foreign key column connecting record to it's associated game

tag - a varchar field storing one tag associated with the game (there will generally be multiple per game)

Game_requirements

game_id - Foreign key column connects record to it's game

requirement_type - specifies whether the requirement is a minimum or recommended one for the game

os_version - specifies what operating system the requirement is for

processor - the suggested processor for the specified operating system and requirement level

graphics - stores the suggested graphics card for given requirement level and os

memory - stores the suggested memory capacity for given requirement level.

storage - suggested storage capacity for given requirement level.

Game_descriptions

game_id - foreign key columns connects record to it's game

description - stores the short summary description of game

about - stores the longer "about game" section

End of Task 1

Task

100%

Group: Define Appropriate Tables for Data
Task #2: Add the pull request link for the branch related to this feature
Weight: ~50%
Points: ~0.50

COLLAPSE

Details:
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/57

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

URL #2
https://github.com/Dathster/db624-it202-007/pull/65

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

End of Task 2

End of Group: Define Appropriate Tables for Data
Task Status: 2/2

Group

Group: Data Creation Page
Tasks: 4

100%

Points: 2

COLLAPSE

Task

100%

Group: Data Creation Page
Task #1: Screenshots of the creation page
Weight: ~25%
Points: ~0.50

COLLAPSE

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

Columns: 1

Sub-Task

100%

Group: Data Creation Page
Task #1: Screenshots of the creation page
Sub Task #1: Show potentially valid data filled in for the custom creation page

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Data creation page first

Data creation page 2

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

Sub-Task

100%

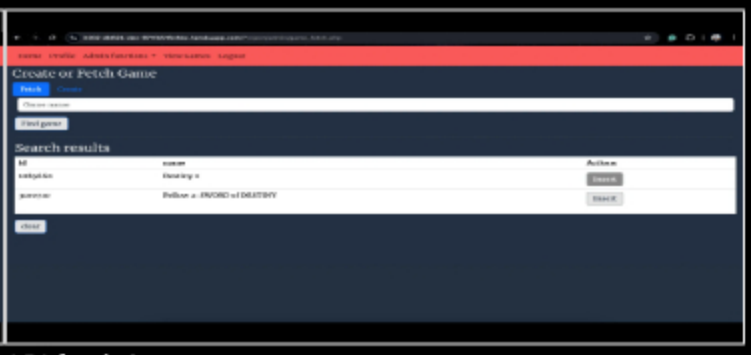
Group: Data Creation Page
Task #1: Screenshots of the creation page
Sub Task #2: Show how the API data is fetched for API data (must be server-side)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

API fetching 1



API fetch 2

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the code

Response:

I have two buttons specifying which tab to display: Fetch from API, or create manually

The fetch from API tab first has a search bar you can use to search for games by name. Once you hit search, a table of search results will pop up. You can enter them into database by pressing insert button.

The manual creation page has a bunch of input fields where we can put in details for the game such as name, id, price, release_date, etc. Once the required fields are filled out (game_id, game name, price, developer name, release date). Once ready, you can press create to manually insert the record

Sub-Task

100%

Group: Data Creation Page

Task #1: Screenshots of the creation page

Sub Task #3: Show examples of validation messages

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Screenshot of validation messages

Screenshot of html validation

Screenshot 1 of js vaidation

part 2 of js validation

Caption(s) (required) ✓

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

Sub-Task

100%

Group: Data Creation Page

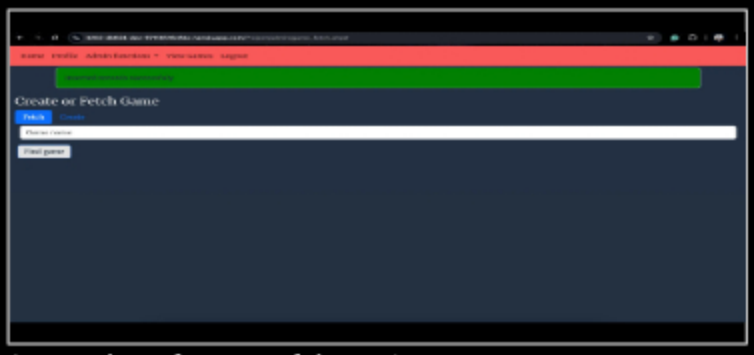
Task #1: Screenshots of the creation page

Sub Task #4: Show an example of successful creation message

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Screenshot of successful creation message

Caption(s) (required) ✓

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

Sub-Task

100%

Group: Data Creation Page

Task #1: Screenshots of the creation page

Sub Task #5: Design/Style should be considered (i.e., bootstrap, custom css, etc)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Styled table display instead of a text dump

Styled nav tab and tab switcher and input fields

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain your design choices

Response:

Some of the styling I've done includes coloring the page to a dark green, color the nav bar on top, style font to make it look nicer, style data output by putting it tables instead of a raw text dump, styled buttons to look nicer and change color when hovered over, and I also styled the tab switcher to have the pill layout.

End of Task 1

Task

100%

Group: Data Creation Page
Task #2: Screenshots of creation page code
Weight: ~25%
Points: ~0.50

COLLAPSE

Details:
Include ucid/date comments for each code screenshot

Columns: 1

Sub-Task

100%

Group: Data Creation Page
Task #2: Screenshots of creation page code
Sub Task #1: Form should have correct data types for each property being requested

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Screenshot of form code for custom data creation

Screenshot of html code for api form fetch

Caption(s) (required) ✓

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

Task Response Prompt

Briefly talk about each field type and why it was chosen

Response:

I used a text input field for the name, dev name, publisher name, franchise name, and tags because the data I expect there are words.

I made the description and about fields textareas because these fields are expected to have larger blocks of text so it would be helpful to be able to resize these fields.

The game_id field is set to number while the price field is set to text because game_id can't have decimals while price needs to have a decimal place. The format is enforced through a pattern.

The release date field uses a date picker field because that makes it easier for user to pick a date and prevents from giving data in wrong format.

Sub-Task

100%

Group: Data Creation Page

Task #2: Screenshots of creation page code

Sub Task #2: Form should have correct validation for each field (HTML, JS, and PHP)

Task Screenshots

Gallery Style: 2 Columns

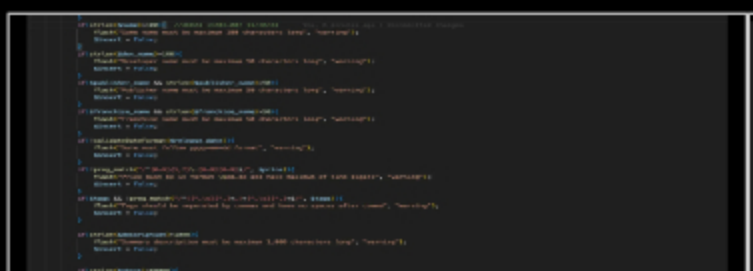
4 2 1

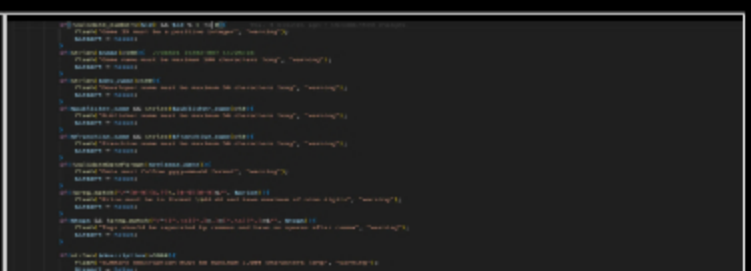
html validation specified through the rules element

part 1 of js validation

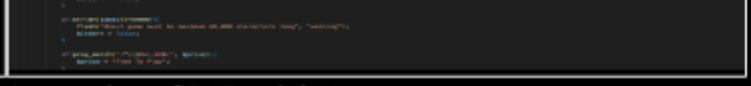
part 2 of js validation

Screenshot of php validation 1





Screenshot of PHP validation 2



Screenshot of PHP validation 3

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the validations

Response:

For the game id field, I'm checking if the input is a positive integer because those are the values accepted by the database

For the name, dev_name, price, game_id, and release_date fields, I'm enforcing that they must have a value by using methods like the required attribute and conditions in js and PHP.

For the text fields, the main validation is a length limit where I check if they're at most 100 or a 1000 characters for example.

The tag field has an additional validation step where I'm checking if all the tags are comma seperated with no spaces after commas. This is done to aid with inserting the tag data into database.

The release_date field's html validation comes down to restricting input to just the calendars, but I also have date format checkers in php and js to make the user can't input invalid dates

Sub-Task

100%

Group: Data Creation Page

Task #2: Screenshots of creation page code

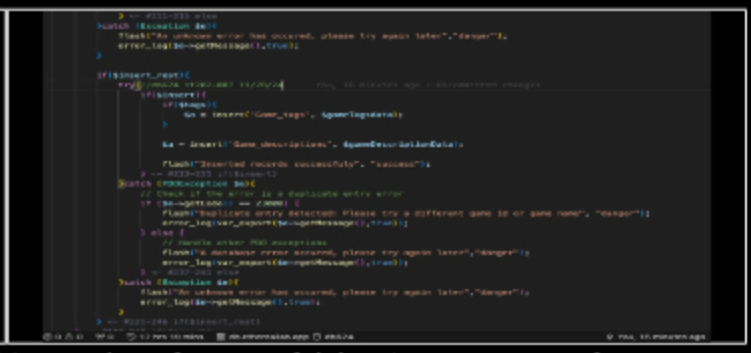
Sub Task #3: Successful creation should have a user-friendly message

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Screenshot of code for successful record creation from API



Screenshot of successful data insert message from manual creation

Caption(s) (required) ✓

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

Task Response Prompt

Explain how duplicate/existing data is handled

Response:

Response:

Duplicate/exisisting data is determined based off game id and game name. If the user tries to insert data that has the same game id or game name as another record from database, then the website will throw an error. Other fields are allowed to be duplicated.

Sub-Task

100%

Group: Data Creation Page
Task #2: Screenshots of creation page code
Sub Task #4: Any errors should have user-friendly messages

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Error when user tries to enter duplicate record

Errors when trying to manually enter data

Generic error message

Caption(s) (required) ✓

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

Task Response Prompt

Briefly describe the scenarios

Response:

The two main scenarios are either user tries to input a duplicate record or gives improper form data. There is also a possibility that when data is being fetched from the API, there is an error. The program will display a generic error message in that case too.

In the case of a duplicate error, it'll print a specific message for that. Same for improper form data. For all other cases like the api error mentioned above there will be a generic message.

Sub-Task

Group: Data Creation Page

100%

Task #2: Screenshots of creation page code

Sub Task #5: Include the form/process for fetching API data

Task Screenshots

Gallery Style: 2 Columns

4 2 1

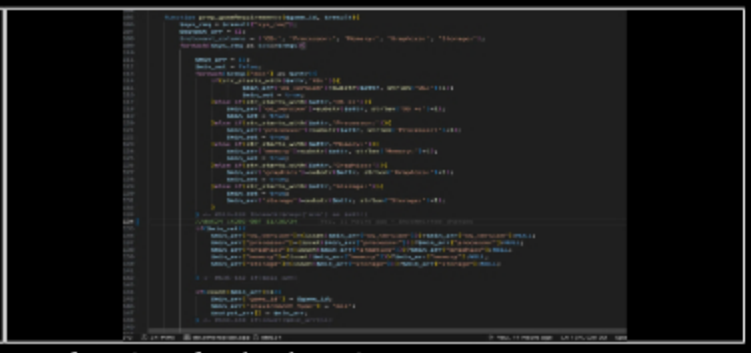
Screenshot of form/process for fetching API data

Another screenshot of the remaining code

Screenshot of api helpers

Screenshot of functions preparing api data for database insert

prep functions for data insert



prep functions for database insert

prep functions for db insert

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the steps (include how duplicates are handled)

Response:

There's two stages: one for searching games and one for actually inserting.

For the searching part, I have the user type in the game name and submit form. I get the name from the form and send it to the api and call the search endpoint, I render the result using render table so that the user can see the game and have a button to insert it.

Once they click that, the form will get the game id associated with that item and make another api call to fetch all the details for that game. I have seperate functions to prep that result data for each of the five functions, after which I make database calls to insert the data.

If everything goes well there will be a success message, otherwise, it will print the relevent error message. In the case of duplicate game id or name, the website will throw an error message, other fields are allowed to be duplicates though.

Sub-Task

100%

Group: Data Creation Page
Task #2: Screenshots of creation page code
Sub Task #6: Include some indicator between custom data and API data

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Screenshot of from_api indicator distinguishing API and form data

Caption(s) (required) ✓

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

Task Response Prompt

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

Response:

I have a field called from_api, this is stored as a tinyiny (1 or 0) and by default it is set to 1. Whenever I insert data from the API, I don't modify this field as it's at it's intended value already. However, if I am manually inserting data, the code will pass a from_api value of zero while inserting data into table because it is a manual form so it is not from api.

Sub-Task

100%

Group: Data Creation Page
Task #2: Screenshots of creation page code
Sub Task #7: Include any other rules like role guards and login checks

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Role guard

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the logic/reasoning

Response:

This page must be accessible only by Admins, because I don't want end users just inserting garbage game data. As a result, I have placed admin role guard so that any users who aren't logged in or don't have the admin role will be redirected.

End of Task 2

Task

100%

Group: Data Creation Page
Task #3: Screenshot of records from DB
Weight: ~25%
Points: ~0.50

COLLAPSE

Columns: 1

Sub-Task

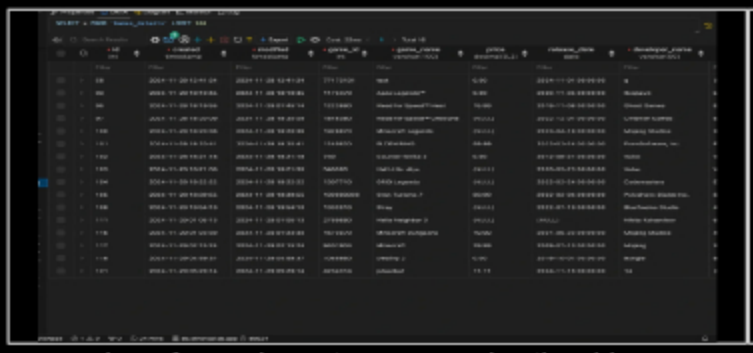
100%

Group: Data Creation Page
Task #3: Screenshot of records from DB
Sub Task #1: Show at least one record fetched from the API

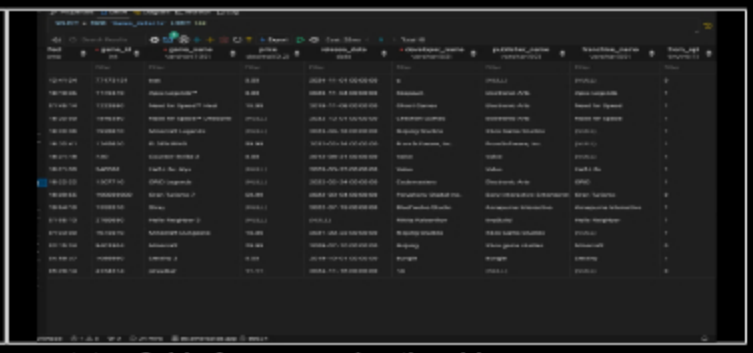
Task Screenshots

Gallery Style: 2 Columns

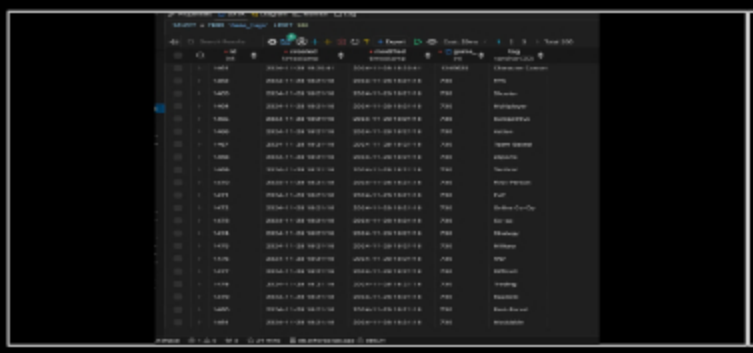
4 2 1



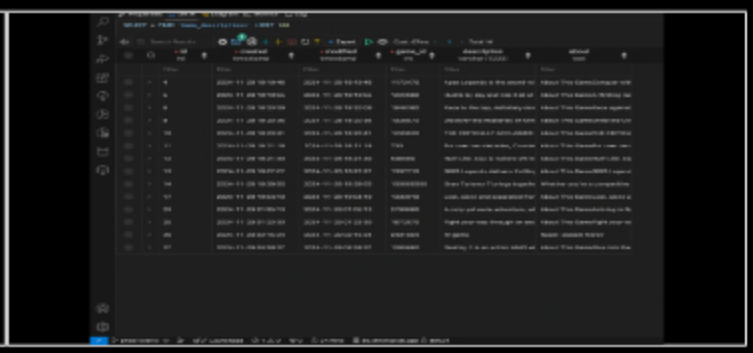
screenshot of records put into games details table



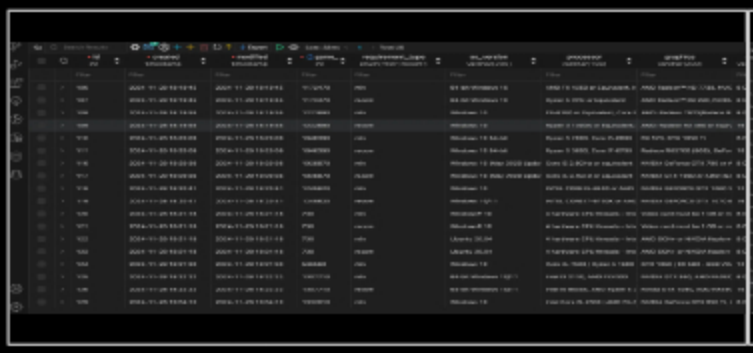
remaining fields for games details table



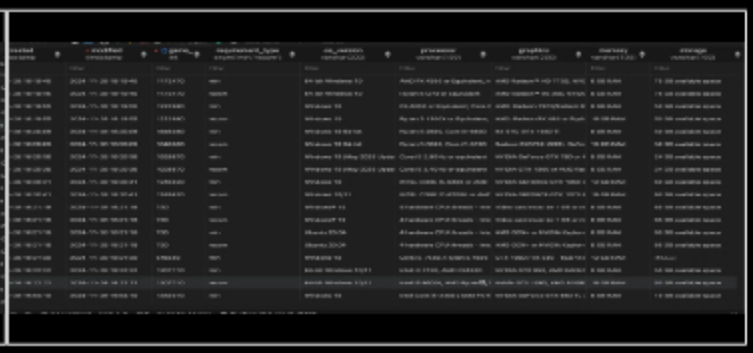
Screenshot of records inserted into tags table



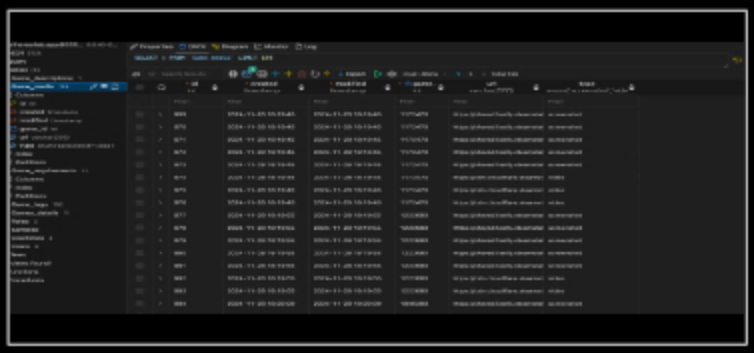
Screenshot of data inserted into descriptions table



game requirements 1



game requirements 2



screenshot of game media data

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown (note what differs from a custom record)*

Sub-Task

100%

Group: Data Creation Page

Task #3: Screenshot of records from DB

Sub Task #2: Show at least one record created via the creation form

Task Screenshots

Gallery Style: 2 Columns

4 2 1

screenshot of manuall record

Screenshot of the data for a manually inserted record

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown (note what differs from the API record)*

End of Task 3

Task

100%

Group: Data Creation Page
Task #4: Add related links
Weight: ~25%
Points: ~0.50

COLLAPSE

Checklist

*The checkboxes are for your own tracking

#	Details
#1	Include the heroku prod link for this page
#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/games_view.php?

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

URL #2
https://github.com/Dathster/db624-it202-007/pull/58

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

End of Task 4

End of Group: Data Creation Page
Task Status: 4/4

Group

100%

Group: Data List Page (many entities)
Tasks: 3
Points: 2

COLLAPSE

Task

100%

Group: Data List Page (many entities)
Task #1: Screenshots of the list page
Weight: ~33%
Points: ~0.67

COLLAPSE

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

Columns: 1

Sub-Task

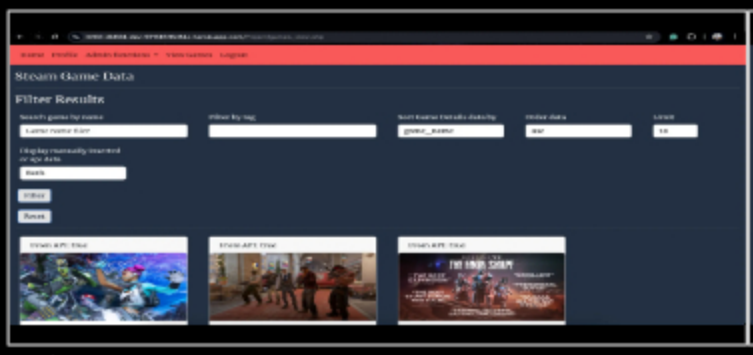
100%

Group: Data List Page (many entities)
Task #1: Screenshots of the list page
Sub Task #1: Show the page of your entities listed (have a reasonable number shown)

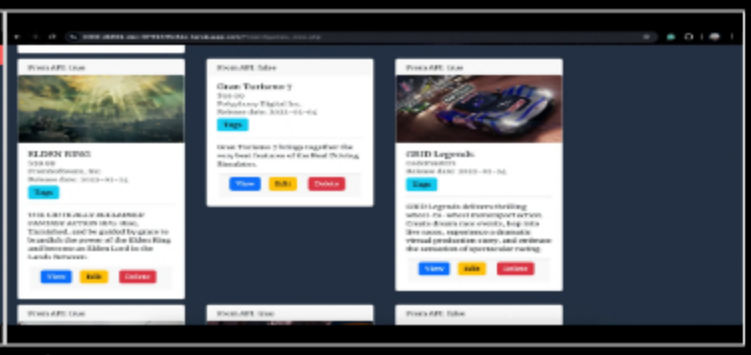
Task Screenshots

Gallery Style: 2 Columns

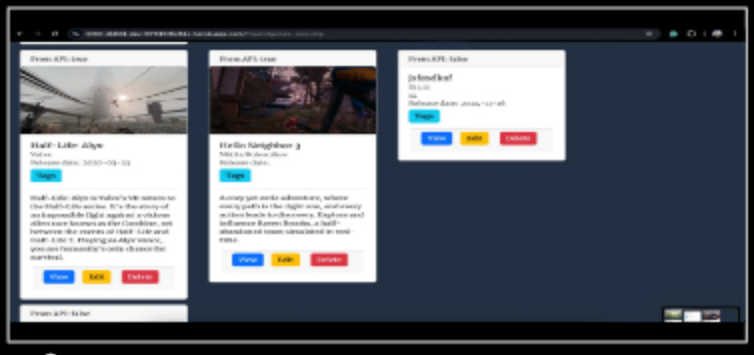
4 2 1



ss 1



ss 2



ss 3

Caption(s) (required) ✓

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

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

Task Response Prompt

Briefly describe the page

Response:

The page has a set of filter parameters on the top, followed by all the game results below. Each game is displayed on a card with some summary data such as a thumbnail, game, price, release date, developer name, tags, description, and some admin options.

Sub-Task 100%

Group: Data List Page (many entities)
Task #1: Screenshots of the list page
Sub Task #2: Show the filter/sort form based on your data and the required limit field

Task Screenshots

Gallery Style: 2 Columns

4 2 1

filtered

Caption(s) (required) ✓

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

Task Response Prompt

Briefly mention what's available to the user

Response:

search and tag filters filter based on whether the string is found within any valid game names or tags.

order field is a dropdown setting ascending or descending order

the limit field limits number of records returned, max 100 min 1.

the order by columns specifies by what column the records should be ordered by.

the from api column specifies whether the games displayed are from api or manually created

Sub-Task 100%

Group: Data List Page (many entities)
Task #1: Screenshots of the list page
Sub Task #3: Demonstrate a few varied filters/sorts

Task Screenshots

Gallery Style: 2 Columns

4 2 1

1 screenshot

ss 2

ss 3

Caption(s) (required) ✓

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

Sub-Task 100%

Group: Data List Page (many entities)
Task #1: Screenshots of the list page
Sub Task #4: Demonstrate a filter that doesn't have any records (should show an appropriate message)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Screenshot when no results found

Caption(s) (required) ✓

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

Sub-Task 100%

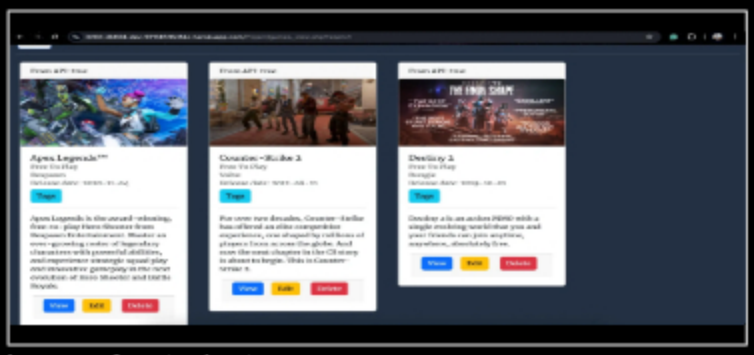
Group: Data List Page (many entities)
Task #1: Screenshots of the list page

Sub Task #5: Each list item should have a link of single view (i.e., details), edit, and delete (some of which may only be visible to admin users, include examples from different user roles)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



buttons for single view

Caption(s) (required) ✓

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

Task Response Prompt

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

Response:

Any logged in user can view this page. However, the edit and delete buttons are only displayed if the user is an admin. All users can press the view button though.

Sub-Task

100%

Group: Data List Page (many entities)

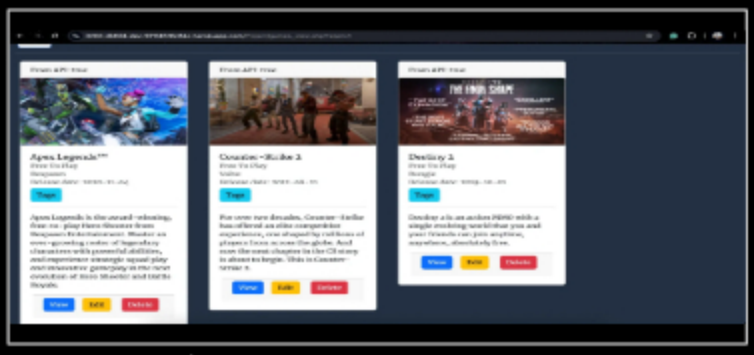
Task #1: Screenshots of the list page

Sub Task #6: Each list item should have a summary of the entity (likely won't be the entire entity data)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



game summaries

Caption(s) (required) ✓

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

Sub-Task

Group: Data List Page (many entities)

Task #1: Screenshots of the list page

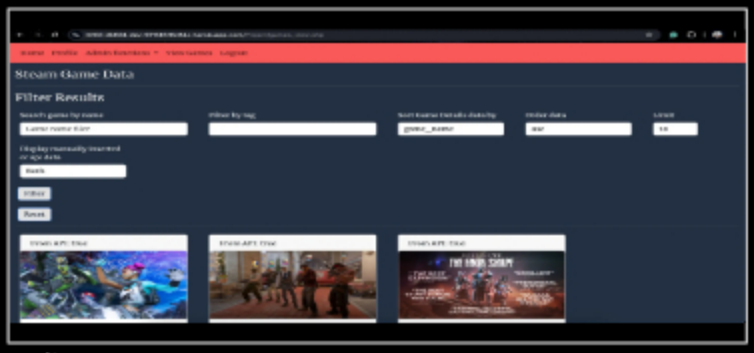
100%

Sub Task #7: Design/Style should be considered (i.e., bootstrap, custom css, etc)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



styling

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain your design choices

Response:

I used bootstrap and CSS to color the page and give a red nav bar. I also styled all the font and input fields of the page to look nicer. I used bootstrap cards to display all the games in card format rather than doing a plaintext dump of everything

End of Task 1

Task

100%

Group: Data List Page (many entities)
Task #2: Screenshots of the list page code
Weight: ~33%
Points: ~0.67

COLLAPSE

Details:
Include ucid/date comments for each code screenshot

Columns: 1

Sub-Task

100%

Group: Data List Page (many entities)
Task #2: Screenshots of the list page code
Sub Task #1: Show the filter/sort form generation

Task Screenshots

Gallery Style: 2 Columns

4 2 1

filter/sort form code

Caption(s) (required) ✓

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

Task Response Prompt

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

Response:

I have different input fields for each type of input.

The game and tag search fields are text input fields where the user can input words.

The sort game details by order fields are dropdowns because I want the users to only select specific values.

The limit field is type number so that I can restrict the user to only entering numbers, I also enforced max and min values in HTML

the api filter field is also a dropdown so that the user can only pick from a fixed set of values.

Sub-Task

100%

Group: Data List Page (many entities)
Task #2: Screenshots of the list page code
Sub Task #2: Show the DB query and how the filter/sort is handled (including the restriction on the limit field)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

screenshot of db query

screenshot of db insert

screenshot of db query

screenshot of db insert

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the related logic

Response:

For the first query, I am selecting the game_id, name, price, release date, and dev name, and from_api from games_details. Then, I am using a CTE to get all the tags associated from the game as one string. I am using a case statement to return a price of 0.00 as Free to play, or put a $ in front of price otherwise. I also used an if statement to display the 1's and 0's of from_api field as true or false.

With this select query, I append additional filter criteria through where clauses, subbing in results I got from user like search, column to be ordered by, relevant tag, and limit records and sort order (asc/desc).

When filtering by tag, I used an exists clause as the cte returns one string rather than individual tags that way I can filter by specific tags.

Sub-Task

100%

Group: Data List Page (many entities)

Task #2: Screenshots of the list page code

Sub Task #3: Show how the output is generated and displayed

Task Screenshots

Gallery Style: 2 Columns

4 2 1

method calling cards and displaying them

screenshot of processing db call to display for output

card code 1

card code 2

card code 3

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the related logic

Response:

Once the database call is done, I go through all the results from all the tables and compile them into one associative array record, and then append that record into a list that stores records for all the games. Then I have a foreach loop that goes through each result and renders a card. All of these are generated in the same row but different columns so that there can be multiple records in one flow and it automaticallly flows down if there isn't enough space on a row.

in the card view method, I put data from parameter into variables so that I can print them more easily. I printed all the data in seperate fields, like the name, price, release date, summary, etc. This is all done using the render input helper functions

I stored the tags within a button dropdown and used bootstrap to achieve this. When a user clicks on the tag button then it'll display all the tags. This is done to reduce clutter.

Sub-Task

100%

Group: Data List Page (many entities)

Task #2: Screenshots of the list page code

Sub Task #4: Show any restrictions like role guard or login checks

Task Screenshots

Gallery Style: 2 Columns

4 2 1

no login checks

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the logic/reasoning

Response:

Since this is simply a view page, I don't have any login or role checks present.

End of Task 2

Task

100%

Group: Data List Page (many entities)
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 this page
#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/games_view.php

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

URL #2
https://github.com/Dathster/db624-it202-007/pull/59

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

End of Task 3

End of Group: Data List Page (many entities)
Task Status: 3/3

Group

100%

Group: View Details Page (single entity)
Tasks: 3
Points: 1

COLLAPSE

Task

100%

Group: View Details Page (single entity)
Task #1: Screenshots of the details page
Weight: ~33%
Points: ~0.33

COLLAPSE

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

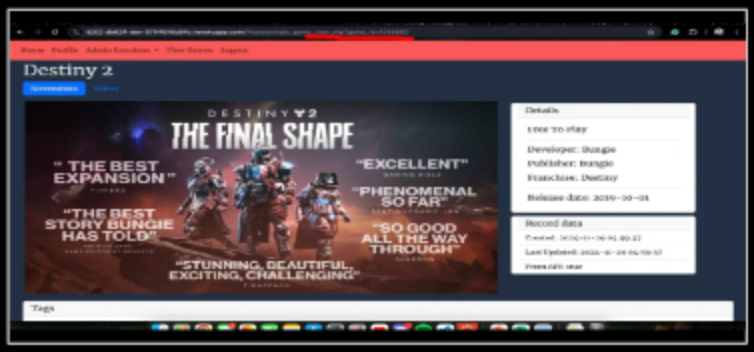
Columns: 1

Sub-Task
100%
Group: View Details Page (single entity)
Task #1: Screenshots of the details page
Sub Task #1: Entity should be fetch by id (via the url)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



screenshot 1

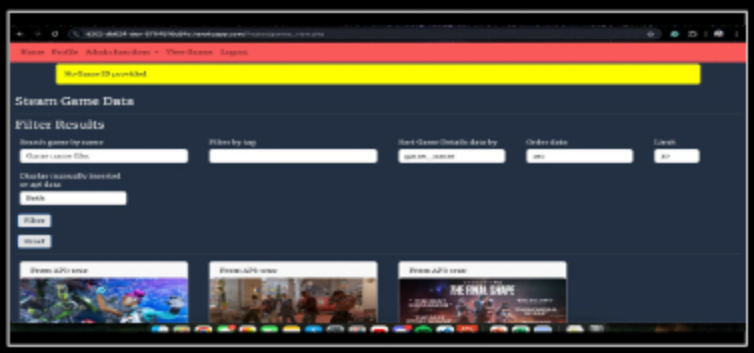
Caption(s) (required) ✓
Caption Hint: *Describe/highlight what's being shown*

Sub-Task
100%
Group: View Details Page (single entity)
Task #1: Screenshots of the details page
Sub Task #2: A missing id should redirect back to the list page with an applicable message

Task Screenshots

Gallery Style: 2 Columns

4 2 1



redirect message

Caption(s) (required) ✓

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

Sub-Task

100%

Group: View Details Page (single entity)

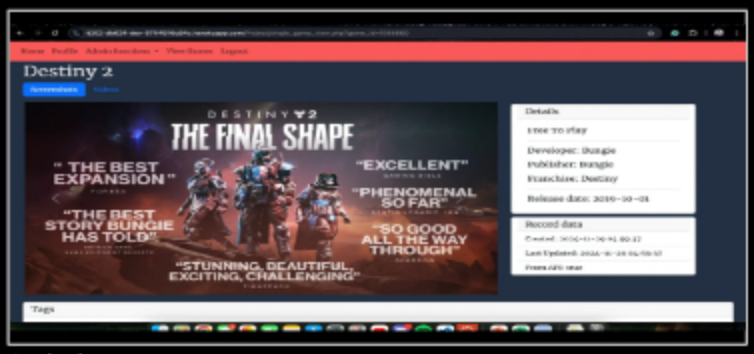
Task #1: Screenshots of the details page

Sub Task #3: Design/Style should be considered (i.e., bootstrap, custom css, etc)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Styled page

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain your design choices

Response:

I styled the page to have all the image by displayed as an image instead of a link. I also implemented a carousel so that user can go through multiple screesnhots. I added a tab switcher for the user to switch between the video and screenshot tabs. All the other information about the game are displayed in different cards. There are also colored buttons displaying admin functions.

Sub-Task

100%

Group: View Details Page (single entity)

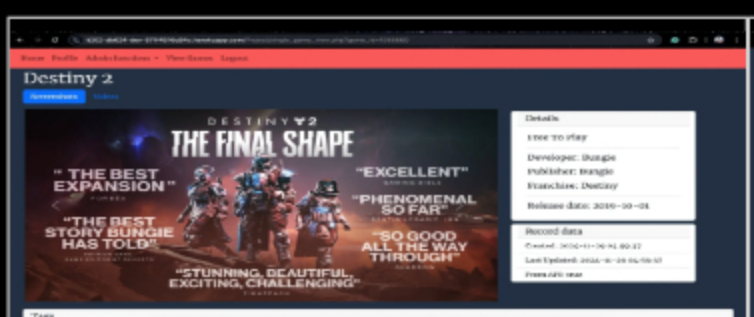
Task #1: Screenshots of the details page

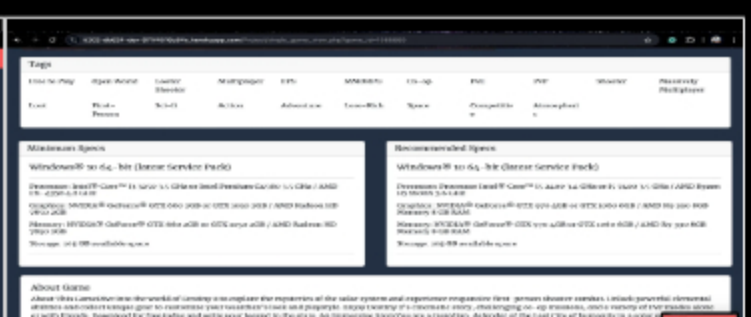
Sub Task #4: Data shown should be more detailed/inclusive than the summary view

Task Screenshots

Gallery Style: 2 Columns

4 2 1





screenshot 1

screenshot 2

screenshot 3

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the details shown and how it differs from the list view

Response:

A carousel is used to display all the screenshots and videos with a button to switch between the media type.

The game details are displayed on a card next to the game. The tags are displayed on a table. Then, I built two cards to display the minimum game requirements for each os and the recommended ones for each os. Finally I added a card to display the "about game"

If the user is an admin they will also be able to view the admin controls to edit and delete the game.

Sub-Task

100%

Group: View Details Page (single entity)

Task #1: Screenshots of the details page

Sub Task #5: There should be a link to edit the entity (this may be an admin-only thing, but it should be present for the respective role)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

admin functions

Caption(s) (required) ✓

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

Sub-Task

Group: View Details Page (single entity)

100%

Task #1: Screenshots of the details page

Sub Task #6: There should be a link to delete the entity (this may be an admin-only thing, but it should be present for the respective role)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

delete button

Caption(s) (required) ✓

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

End of Task 1

Task

100%

Group: View Details Page (single entity)

Task #2: Screenshots of the details page code

Weight: ~33%

Points: ~0.33

COLLAPSE

Details:

Include ucid/date comments for each code screenshot

Columns: 1

Sub-Task

100%

Group: View Details Page (single entity)

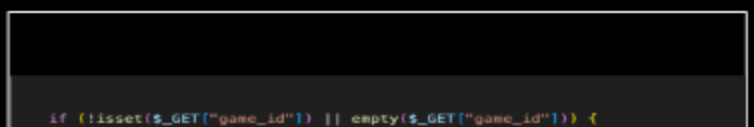
Task #2: Screenshots of the details page code

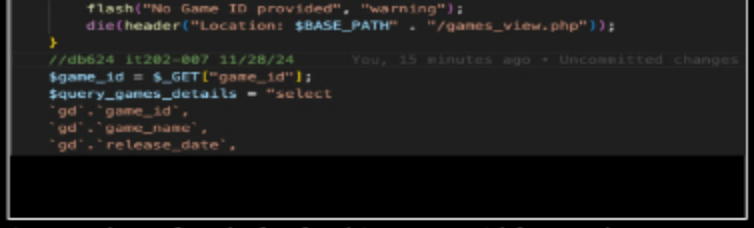
Sub Task #1: Show how id is fetched

Task Screenshots

Gallery Style: 2 Columns

4 2 1





Screenshot of code for fetching game id from url

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the logic and how it's handled when the property is missing or not "valid"

Response:

I am simply accessing the variable through the GET array since the id is sent through query parameter. If the game_id isn't set or is empty in the query parameter, I will display a message saying invalid game id and redirect the user to the game view page.

Sub-Task

100%

Group: View Details Page (single entity)

Task #2: Screenshots of the details page code

Sub Task #2: Show the DB query to get the record

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Queries to get records 1

query screenshots 2

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the logic

Response:

For all the queries I have a where clause specifying the game ID should equal the game id from query parameter. This is to fetch only relevant records.

for the game details query, I am getting the game id, game name, price, release date, dev, publisher, and franchise names, and when the record was created and modified and whether it is from the api or not. I am also adding case and if clauses within the select to format output. Specifically I want price to be formatted with a $ in front or be set as Free To Play if the price is 0 so I am using case labels for that. I also want the from api to be a true or false value

so using an if clause for that.

For the game media and tags queries, I am simply doing a select * as I need all the records from them that have the same game id as the one passed.

For the fetch about query I am doing only a select about because I don't want to display the description field here.

For requirements I broke it into two seperate queries where I fetch the minimum and recommended requirements seperately.

Sub-Task

100%

Group: View Details Page (single entity)
Task #2: Screenshots of the details page code
Sub Task #3: Show the code related to presenting the data and showing the links

Task Screenshots

Gallery Style: 2 Columns

4 2 1

screenshot 1

screenshot 2

screenshot 3

screenshot 4

screenshot 5

screenshot 6

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the logic

Response:

I have a nav for displaying the tabs for screenshot and video. The switching is handled by a javascript script. Based on whichever tab is set active, the other tab is hidden, so you can only see screenshot or only video, not both.

For screenshot and video displays I implemented a bootstrap carousel so that the user can scroll through multiple media elements.

For the game details and record data, I used cards to display them since they're just text info.

I organized these elements so that all 3 would be in one row, but the details and record data cards are stacked on top of each other.

I used a card again to display the tags, I used a foreach loop from php to achieve this, and I put all the tags in one row so that they can overflow down to next line whenever necessary.

I built two seperate cards to display the requirements. I used foreach loops inside each card to display multiple sets of requirements.

I used a basic card to display the game about.

Then, I added a role check and based on that the code would display admin controls. These were stored in a card and were styled as button links.

End of Task 2

Task

100%

Group: View Details Page (single entity)
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 this page
#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/single_game_view.php?game_id=1332010

URL

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

URL #2

URL

https://github.com/Dathster/db624-it202-007/pull/60 https://github.com/Dathster/db624-it202-007/pu

End of Task 3

End of Group: View Details Page (single entity)
Task Status: 3/3

Group

100%

Group: Edit Data Page
Tasks: 4
Points: 2

COLLAPSE

Task

100%

Group: Edit Data Page
Task #1: Screenshots of the edit page
Weight: ~25%
Points: ~0.50

COLLAPSE

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

Columns: 1

Sub-Task

100%

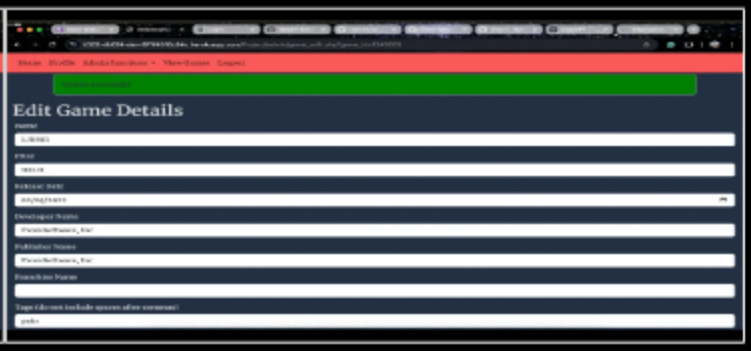
Group: Edit Data Page
Task #1: Screenshots of the edit page
Sub Task #1: Show before and after screenshots of data you'll edit (two screenshots required)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Before edit



after update

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain what you changed

Response:

In this edit I changed the name of the game and the price. It was initially named Elden Ring and I renamed it as L RING.

The price was 59.99 I updated it to a different price.

I also edited the tags to remove all the tags that were previously there and added pain instead

Sub-Task

100%

Group: Edit Data Page

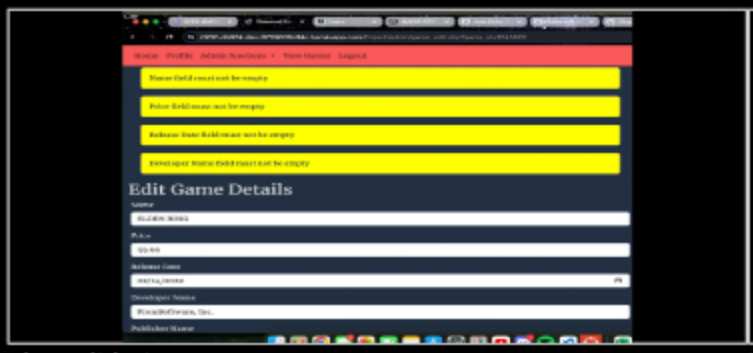
Task #1: Screenshots of the edit page

Sub Task #2: Show examples of validation messages

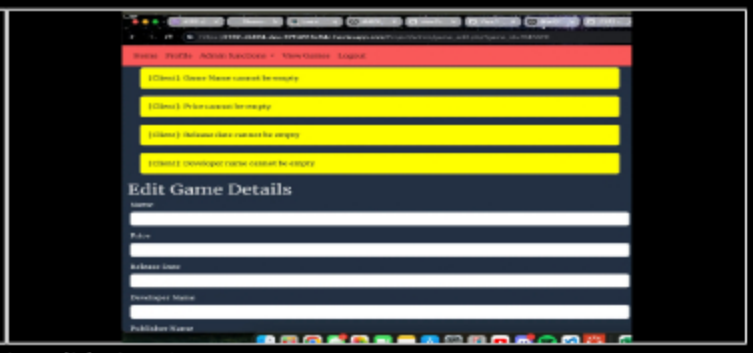
Task Screenshots

Gallery Style: 2 Columns

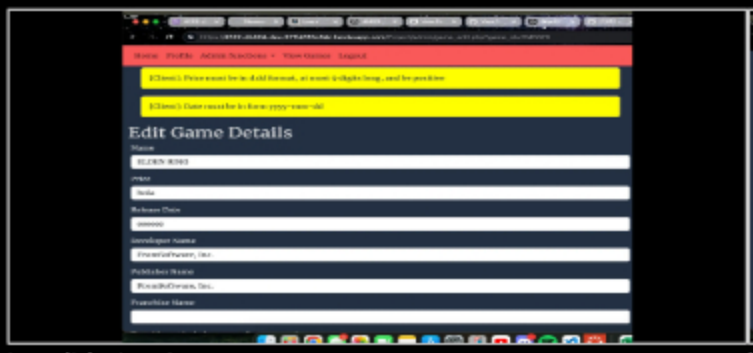
4 2 1



php validations



js validation 1



js validation 2

html validation 1

html validation 2

Caption(s) (required) ✓

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

Sub-Task 100%

Group: Edit Data Page
Task #1: Screenshots of the edit page
Sub Task #3: Show an example of successful edit messages

Task Screenshots

Gallery Style: 2 Columns

4 2 1

successful update message

Caption(s) (required) ✓

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

Sub-Task 100%

Group: Edit Data Page
Task #1: Screenshots of the edit page
Sub Task #4: Design/Style should be considered (i.e., bootstrap, custom css, etc)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

styling

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain your design choices

Response:

I styled the page to have a dark blue background with a red nav bar. I also styled the font to look better than the default. I styled the input fields to have rounded corners and take up more of the page. I have styled the buttons to look nicer compared to default html buttons.

End of Task 1

Task

100%

Group: Edit Data Page
Task #2: Screenshots of edit page code
Weight: ~25%
Points: ~0.50

COLLAPSE

Details:
Include ucid/date comments for each code screenshot

Columns: 1

Sub-Task

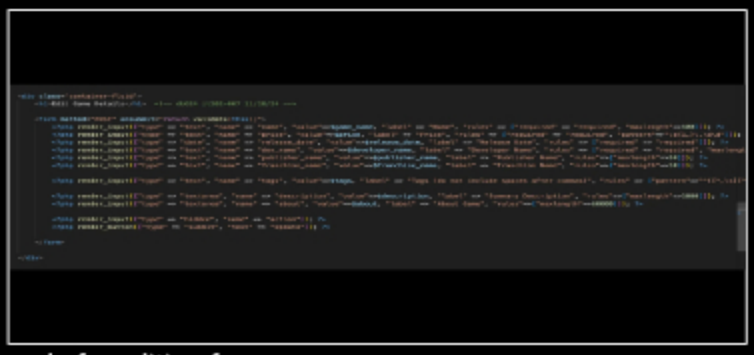
100%

Group: Edit Data Page
Task #2: Screenshots of edit page code
Sub Task #1: Form should have correct data types for each property being requested

Task Screenshots

Gallery Style: 2 Columns

4 2 1



code for editing form

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the data type choices

Response:

I used a text input field for the name, dev name, publisher name, franchise name, and tags because the data I expect there are words.

I made the description and about fields textareas because these fields are expected to have larger blocks of text so it would be helpful to be able to resize these fields.

the price field is set to text because price needs to have a decimal place. The format is enforced through a pattern.

The release date field uses a date picker field because that makes it easier for user to pick a date and prevents from giving data in wrong format.

Sub-Task

100%

Group: Edit Data Page
Task #2: Screenshots of edit page code
Sub Task #2: Form should have correct validation for each field (HTML, JS, and PHP)

Task Screenshots

Gallery Style: 2 Columns

4 2 1

html validations under required field

js validation 1

js validation 2

php validation 1

php validation 2

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the validations

Response:

For the name, dev_name, price, game_id, and release_date fields, I'm enforcing that they must have a value by using methods like the required attribute and conditions in js and PHP.

For the text fields, the main validation is a length limit where I check if they're at most 100 or a 1000 characters for example.

The tag field has an additional validation step where I'm checking if all the tags are comma seperated with no spaces after commas. This is done to aid with inserting the tag data into database.

The release_date field's html validation comes down to restricting input to just the calendars, but I also have date format checkers in php and js to make the user can't input invalid dates

Sub-Task

100%

Group: Edit Data Page

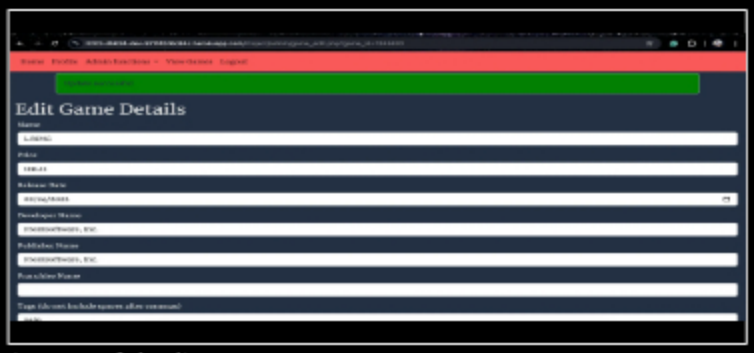
Task #2: Screenshots of edit page code

Sub Task #3: Successful edit should have a user-friendly message

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Successful edit message

Caption(s) (required) ✓

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

Task Response Prompt

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

Response:

My PHP is set up in a way where it does data update before autofilling form fields.

For doing update, I go through the post array and get the relevant data from every single input field and store them into variables. After that,

Sub-Task

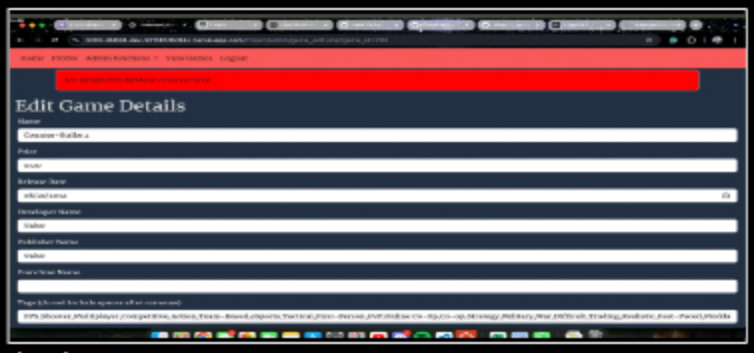
100%

Group: Edit Data Page
Task #2: Screenshots of edit page code
Sub Task #4: Any errors should have user-friendly messages

Task Screenshots

Gallery Style: 2 Columns

4 2 1



database error

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the possible errors

Response:

Some possible errors are that the user uses a game name that's used by another name, or tries to enter tags that are too long. Another possibility is the user inputting an invalid date somehow. However, besides the two specific errors mentioned above, the validations take care of the other cases so invalid inputs will likely throw errors.

Sub-Task

100%

Group: Edit Data Page
Task #2: Screenshots of edit page code
Sub Task #5: Include any other rules like role guards and login checks

Task Screenshots

Gallery Style: 2 Columns

4 2 1

admin role login check

Caption(s) (required) ✓

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

Task Response Prompt

Briefly explain the logic/reasoning

Response:

Currently since I haven't implemented records associated with a specific user (that's milestone 3), I've placed a general admin role guard so that regular users don't go and update data in wrong ways.

End of Task 2

Task

100%

Group: Edit Data Page
Task #3: Screenshot of records from DB
Weight: ~25%
Points: ~0.50

COLLAPSE

Sub-Task

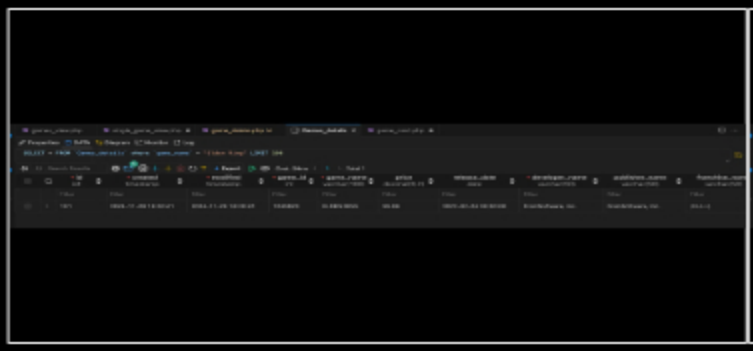
100%

Group: Edit Data Page
Task #3: Screenshot of records from DB
Sub Task #1: Show a before and after screenshot of the record (two screenshots)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



before edit

Updated record (Note the same id and game id)

Caption(s) (required) ✓

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

Task Response Prompt

Explain what differs

Response:

In this edit I changed the name of the game and the price. It was initially named Elden Ring and I renamed it as L RING.

The price was 59.99 I updated it to a different price.

End of Task 3

Task

100%

Group: Edit Data Page
Task #4: Add related links
Weight: ~25%
Points: ~0.50

COLLAPSE

Checklist *The checkboxes are for your own tracking

#	Details
#1	Include the heroku prod link for this page
#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/66

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

URL #2
https://it202-db624-prod-a236ea831ca5.herokuapp.com/Project/admin/game_edit.php?game_id=730

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

End of Task 4

End of Group: Edit Data Page
Task Status: 4/4

Group

100%

Group: Delete Handling
Tasks: 3
Points: 1

COLLAPSE

Task

100%

Group: Delete Handling
Task #1: Screenshots related to delete
Weight: ~33%
Points: ~0.33

COLLAPSE

Details:

Heroku dev url must be visible in all relevant screenshots

Include ucid/date comments for each code screenshot

Columns: 1

Sub-Task

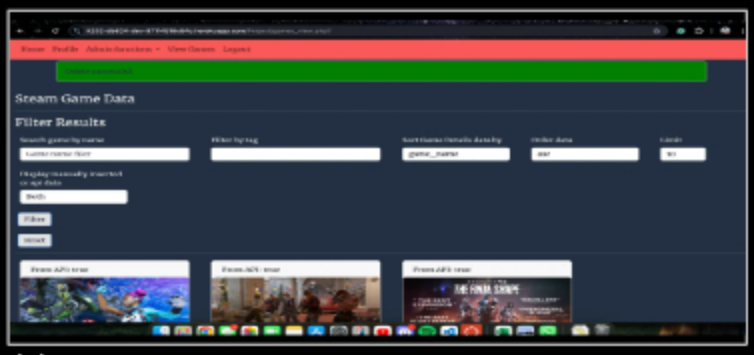
100%

Group: Delete Handling
Task #1: Screenshots related to delete
Sub Task #1: Show the success message of a delete

Task Screenshots

Gallery Style: 2 Columns

4 2 1



delete success

Caption(s) (required) ✓

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

Sub-Task

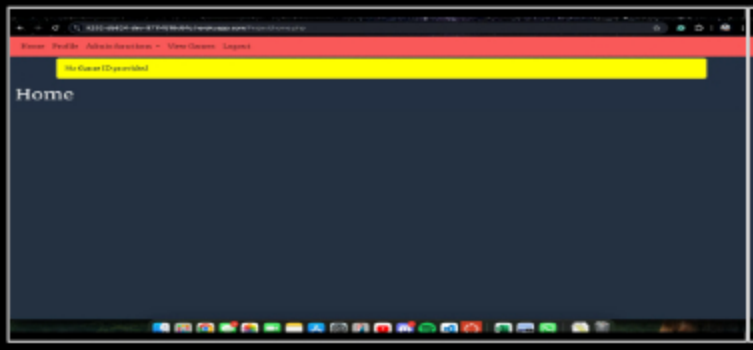
100%

Group: Delete Handling
Task #1: Screenshots related to delete
Sub Task #2: Show any error messages of a failed delete (like id not being passed)

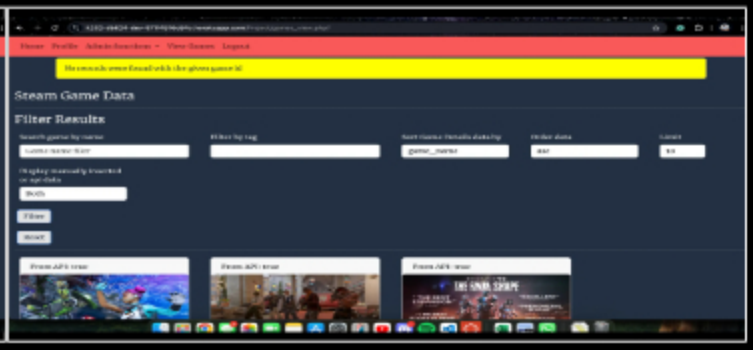
Task Screenshots

Gallery Style: 2 Columns

4 2 1



Redirect for no id given



screenshot for invalid id given

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

If the link doesn't contain a game id query parameter (ie. it isnt set), then the user will be redirected to home page. However, if the id is invalid then the user is redirected to the game view page so that they can try to delete some other game.

Sub-Task

100%

Group: Delete Handling
Task #1: Screenshots related to delete
Sub Task #3: Show the code related to the delete processing

Task Screenshots

Gallery Style: 2 Columns

4 2 1

delete processing code

Caption(s) (required) ✓

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

Task Response Prompt

Explain in concise steps how this logically works

Response:

First there is a role check making sure only admins are trying to delete.

Then there is a check to see if the id is set in the first place.

If both checks fail the user is redirected to the home page.

If checks pass then I go through the GET array to the game id. I check if it is a positive number then I try to execute the delete query. I am deleting from the games details table, and because all the other tables have a cascade delete so that if a record is deleted from game details table then all related records will be dropped from all the tables

Sub-Task

100%

Group: Delete Handling
Task #1: Screenshots related to delete
Sub Task #4: Explain the delete logic

Task Response Prompt

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)?

Response:

This is a hard delete because data is being dropped from database rather than just being set as logically deleted. There is an admin role check because only admins should be able to delete general records. They will also have to be logged in.

End of Task 1

Task

100%

Group: Delete Handling
Task #2: Screenshots of the data
Weight: ~33%
Points: ~0.33

COLLAPSE

Sub-Task

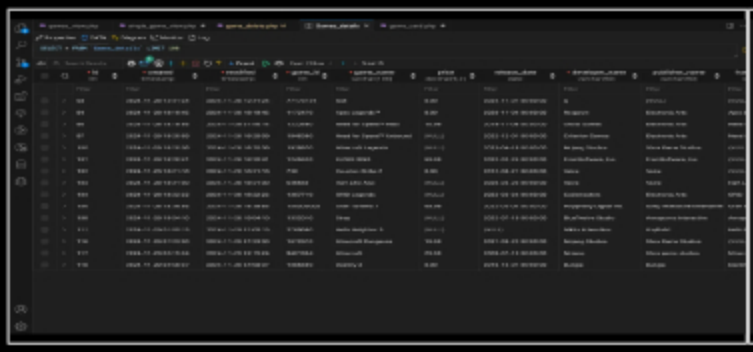
100%

Group: Delete Handling
Task #2: Screenshots of the data
Sub Task #1: Show a before and after screenshot of the DB data (two screenshots)

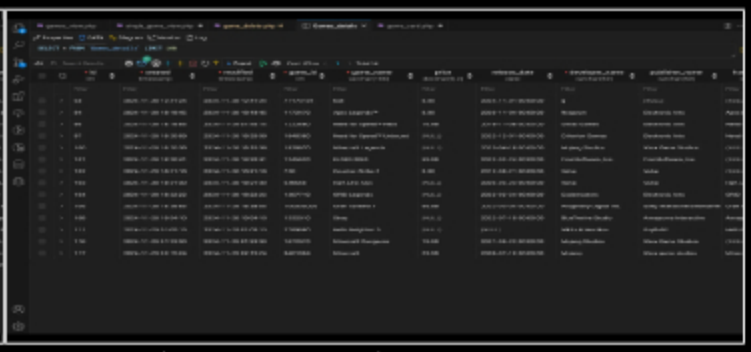
Task Screenshots

Gallery Style: 2 Columns

4 2 1



before delete



After delete (Destiny 2 is gone)

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown (note precisely what changed)*

End of Task 2

Task

100%

Group: Delete Handling
Task #3: Add the pull request link for the branch related to this feature
Weight: ~33%
Points: ~0.33

COLLAPSE

Details:
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/63

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

End of Task 3

End of Group: Delete Handling
Task Status: 3/3

Group

100%

Group: Misc
Tasks: 4
Points: 1

COLLAPSE

Task

100%

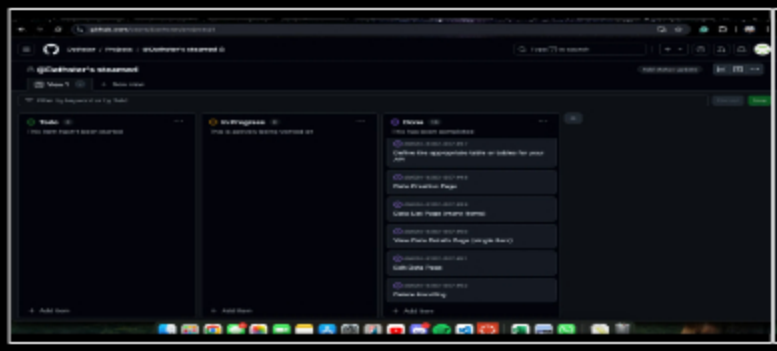
Group: Misc
Task #1: Screenshot of your project board from GitHub (tasks should be in the proper column)
Weight: ~25%
Points: ~0.25

COLLAPSE

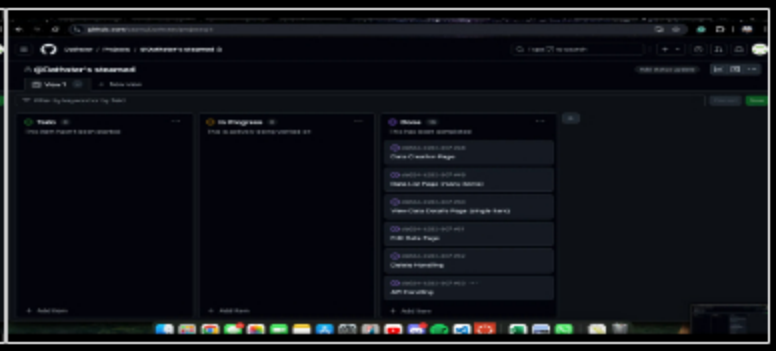
Task Screenshots

Gallery Style: 2 Columns

4 2 1



screenshot 1



screenshot 2

End of Task 1

Task

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

Group: Misc
Task #3: Talk about any issues or learnings during this assignment
Weight: ~25%
Points: ~0.25

COLLAPSE

Task Response Prompt

Response:

I learned a lot more about how to make database calls and display data dynamically using php. Before I didn't understand fully how I can render HTML elements using conditions in php like if or loops. However, this project has helped me get a better understanding of that. I have also learned better how to use bootstrap as I had to implement and edit those elements while styling my pages.

I didn't really have big issues besides just bugs I had to fix.

End of Task 3

Task

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

4 2 1

time spent in project

screenshot of specific branch times

End of Task 4

End of Group: Misc
Task Status: 4/4

End of Assignment