

Course: IT202-010-S2025

Assignment: IT202 Milestone 2

Student: Justin C. (jjc88)

Status: Submitted | Worksheet Progress: 99%

Potential Grade: 9.89/10.00 (98.90%)

Received Grade: 0.00/10.00 (0.00%)

Grading Link: <https://learn.ethereallab.app/assignment/v3/IT202-010-S2025/it202-milestone-2/grading/jjc88>

## Instructions

1. Refer to Milestone2 of this doc:

<https://docs.google.com/document/d/1XE96a8DQ52Vp49XACBDTNCq0xYDt3kF29cO88EWVwfo/view>

2. Ensure you read all instructions and objectives before starting.
3. Ensure you've gone through each lesson related to this Milestone
4. Switch to the Milestone2 branch

1. `git checkout Milestone2` (ensure proper starting branch)
2. `git pull origin Milestone2` (ensure history is up to date)

5. Fill out the below worksheet

- Ensure there's a comment with your UCID, date, and brief summary of the snippet in each screenshot
- Ensure proper styling is applied to each page
- Ensure there are no visible technical errors; only user-friendly messages are allowed

6. Once finished, click "Submit and Export"

7. Locally add the generated PDF to a folder of your choosing inside your repository folder and move it to Github

1. `git add .`
2. `git commit -m "adding PDF"`
3. `git push origin Milestone2`
4. On Github merge the pull request from Milestone2 to dev
5. On Github create a pull request from dev to prod and immediately merge. (This will trigger the prod deploy to make the heroku prod links work)

8. Upload the same PDF to Canvas

9. Sync Local

10. `git checkout dev`

11. `git pull origin dev`

100%

### Section #1: ( 2 pts.) Data

100%

#### Task #1 ( 0.50 pts.) - API Data Table

## Combo Task:

**Weight: 25%**

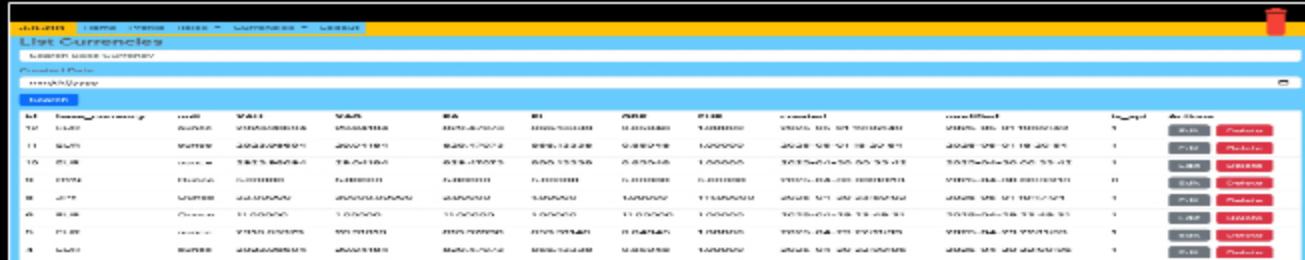
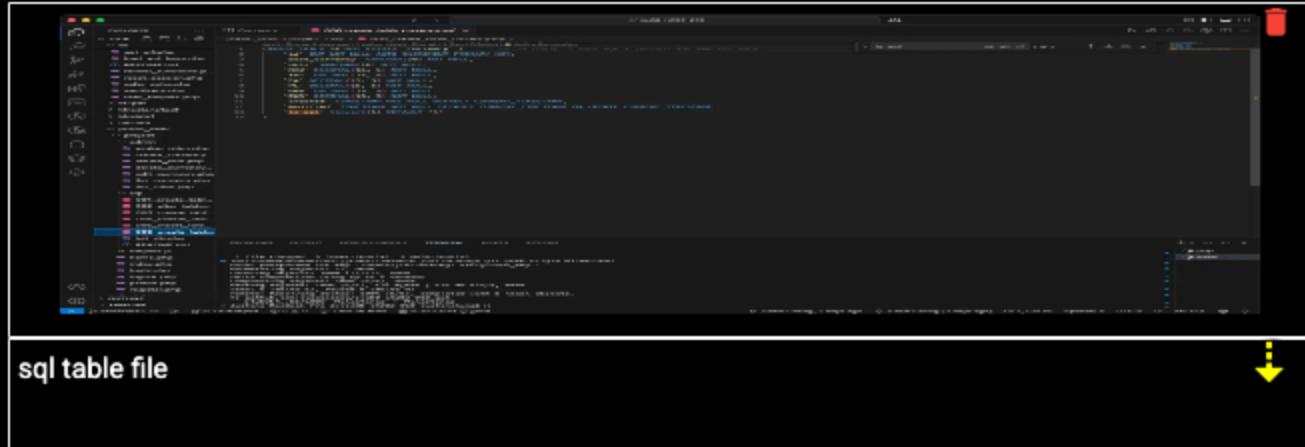
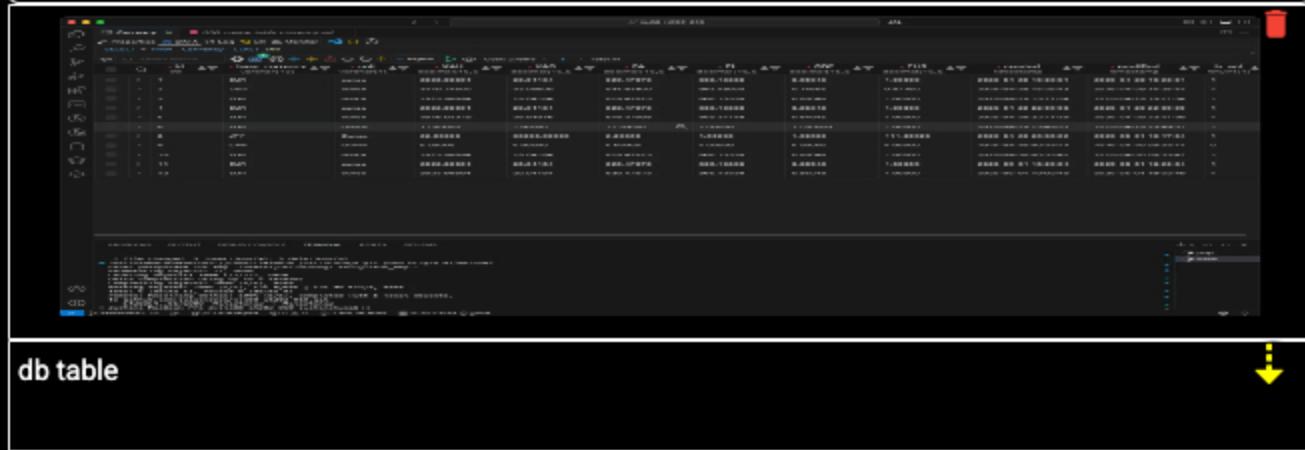
## **Objective:** *API Data Table*

## ≡, Image Prompt

**Weight: 34%**

### Details:

- Show the table(s) structure of the table made to hold your API
  - There should be at least 3 properties/fields beyond `id`, `created`, `modified`, and respective `is_api` or `api_id` columns
  - Logical types and constraints should be set
  - Show at least a few records from each API table (samples should include both API fetched data and custom entries noted by an `api_id` or `is_api` column)



1	12345	2023-08-01	2024-07-31	0.00-100.0	0.00-100.0	100000	2023-01-30 00:00:00	2023-06-30 10:37:34	1	<span>View</span>	<span>Edit</span>	<span>Delete</span>
2	12345	2023-08-01	2024-07-31	0.00-100.0	0.00-100.0	100000	2023-01-30 00:00:00	2023-06-30 10:37:34	1	<span>View</span>	<span>Edit</span>	<span>Delete</span>

table



Saved: 5/1/2025 7:33:07 PM

## Url Prompt

**Weight:** 33%

**Details:**

- Add a direct link to your sql file from Github (should end in .sql)

URL #1

<https://github.com/juicejoose/jjc88->



URL

<https://github.com/juicejoose/jjc88->

[https://it2025010/Milestone2/public\\_html/project/sql/006\\_create\\_table\\_currency.sql](https://it2025010/Milestone2/public_html/project/sql/006_create_table_currency.sql)



Saved: 5/1/2025 7:33:07 PM

## Text Prompt

**Weight:** 33%

**Details:**

- Briefly note what each field/property represents

Your Response:

Id represents the entity number base\_currency is the compared currency that the user inputs unit is how the metals will be measured XAU, XAG, PA, PI, GBP, EUR will represent the comparable Created and modified are dates of when created and modified is\_api tracks if the data is api or user made



Saved: 5/1/2025 7:33:07 PM

100%

## Task #2 ( 0.50 pts.) - Data Transformation

Combo Task:

**Weight:** 25%

**Objective:** *Data Transformation*

## Image Prompt

**Weight:** 40%

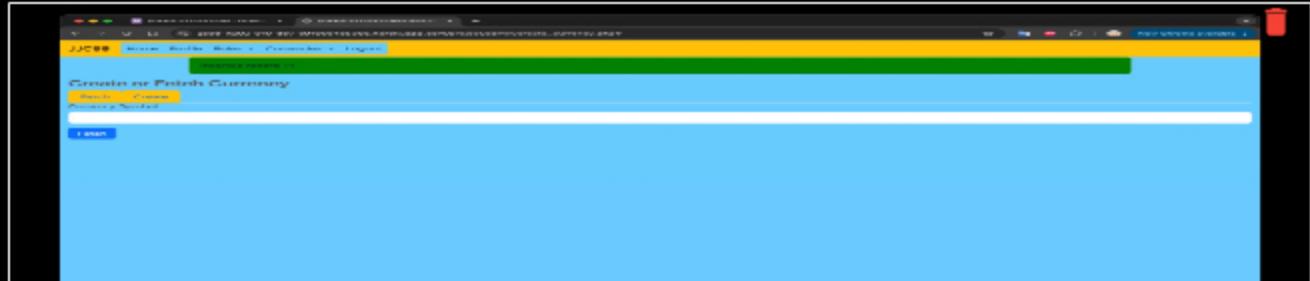
**Details:**

- Show the code that handles the data transformation/conversion (from the API)
- Show a Heroku dev example of API getting fetched and loaded into the DB

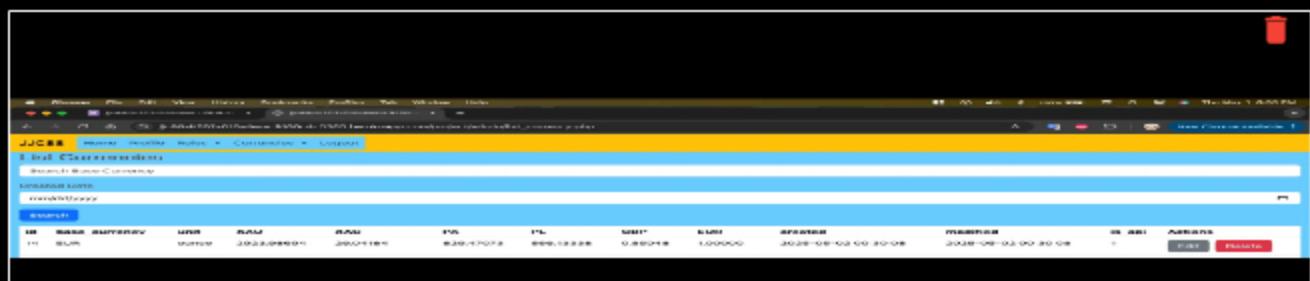


A screenshot of a terminal window with dark mode enabled. The window contains several lines of code, likely shell or Python script, related to data processing. The text is mostly illegible due to the small font size.

transform code



fetching an



A screenshot of a database management system (DBMS) interface. A table named "CURRENCIES" is displayed with columns: ID, NAME, SYMBOL, DIGIT, DECIMAL, RATE, and LAST\_UPDATED. The data shows 14 rows of currency information.

ID	NAME	SYMBOL	DIGIT	DECIMAL	RATE	LAST_UPDATED
1	US Dollar	\$	2	2	100.00000000	2023-01-01 00:00:00
2	EURO	€	2	2	100.00000000	2023-01-01 00:00:00
3	British Pound	£	2	2	100.00000000	2023-01-01 00:00:00
4	Canadian Dollar	CA\$	2	2	100.00000000	2023-01-01 00:00:00
5	Australian Dollar	AU\$	2	2	100.00000000	2023-01-01 00:00:00
6	Swiss Franc	CHF	2	2	100.00000000	2023-01-01 00:00:00
7	Japanese Yen	JPY	2	2	100.00000000	2023-01-01 00:00:00
8	Chinese Yuan	CNY	2	2	100.00000000	2023-01-01 00:00:00
9	South African Rand	ZAR	2	2	100.00000000	2023-01-01 00:00:00
10	Mexican Peso	MXN	2	2	100.00000000	2023-01-01 00:00:00
11	Norwegian Krone	NOK	2	2	100.00000000	2023-01-01 00:00:00
12	Polish Złoty	PLN	2	2	100.00000000	2023-01-01 00:00:00
13	Hungarian Forint	HUF	2	2	100.00000000	2023-01-01 00:00:00
14	Romanian Leu	RON	2	2	100.00000000	2023-01-01 00:00:00

fetch 14 in the db



Saved: 5/1/2025 8:45:15 PM

## Url Prompt

**Weight:** 20%

**Details:**

- Include Heroku prod link to API fetch page
- Include pull request link for this feature

URL #1

[https://jjc88-it202-010-prod-b44e1de4d69c.herokuapp.com/project/admin/create\\_currency.php](https://jjc88-it202-010-prod-b44e1de4d69c.herokuapp.com/project/admin/create_currency.php)



URL

<https://jjc88-it202-010-prod-b44e1d>



URL #2

<https://jjc88-it202-010-prod-b44e1de4d69c.herokuapp.com/project/testApi.php>



URL

<https://jjc88-it202-010-prod-b44e1d>



URL #3

<https://github.com/juicejoose/jjc88-it202-010-024>



URL

<https://github.com/juicejoose/jjc88-it202-010-024>



Saved: 5/1/2025 8:45:15 PM

## Text Prompt

**Weight:** 40%

**Details:**

- Will you be using all the data or just a subset?
- Do you need to convert any data or extract pieces of a property's value?
- Briefly explain the transformation/extraction logic to get the API data to the shape your application intends to use

Your Response:

1. I will be using all the data that is given from the api, such as XAU,XAG, and etc
2. Yes the data has to be converted because it is received from JSON and also rounded to fit my data.
3. First decode the API response, extract the rates, base Currency, and unit, then format them into key value pairs for a SQL INSERT and match to our table structure.



Saved: 5/1/2025 8:45:15 PM

100%

Task #3 ( 0.50 pts.) - API Duplicates

## ≡, Text Prompt

**Weight:** 25%

**Objective:** *API Duplicates*

**Details:**

- Consider how duplicate entries are handled
  - What happens if you get the same result from the API for an entity that exists but has been unmodified?
  - What happens if you get the same result from the API for an entity that exists but has been modified?

Your Response:

1. It will still insert a new row, even if the data is unchanged. There's no logic to check for existing identical records.
2. It will still insert a new row with the updated values. Older data will remain



Saved: 5/1/2025 8:47:14 PM

100%

## Task #4 ( 0.50 pts.) - API Fetching

## ≡, Text Prompt

**Weight:** 25%

**Objective:** *API Fetching*

**Details:**

- Consider how your application will trigger this data fetch
  - (i.e., periodic via cron job, poor man's cron job, on app start/awake, etc)
  - (i.e., User or Admin triggered via a specific page/action)
  - (i.e., Lazy loaded during searches for stale/missing content)
- What happens if you get the same result from the API for an entity that exists but has been modified?

Your Response:

1. Data will be admin triggered through an admin page.
2. A new row will be inserted, keeping modified and unmodified.



Saved: 5/1/2025 8:49:29 PM

# 98% Section #2: ( 1.5 pts.) Data Creation

## 95% Task #1 ( 0.60 pts.) - Validations

### Combo Task:

**Weight:** 40%

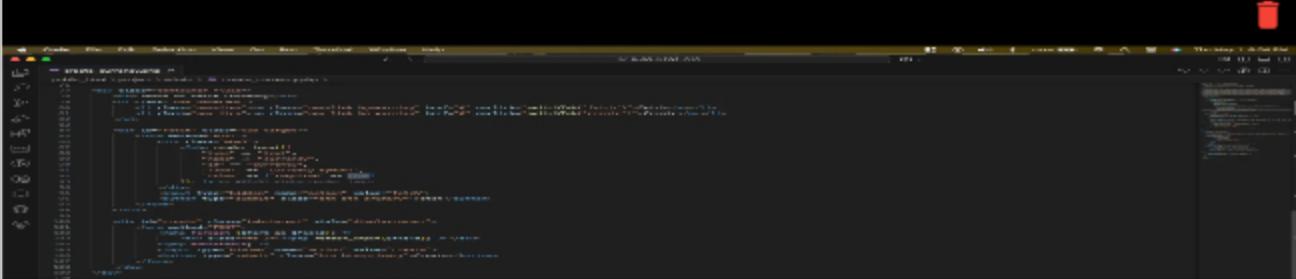
**Objective:** *Validations*

### Image Prompt

**Weight:** 50%

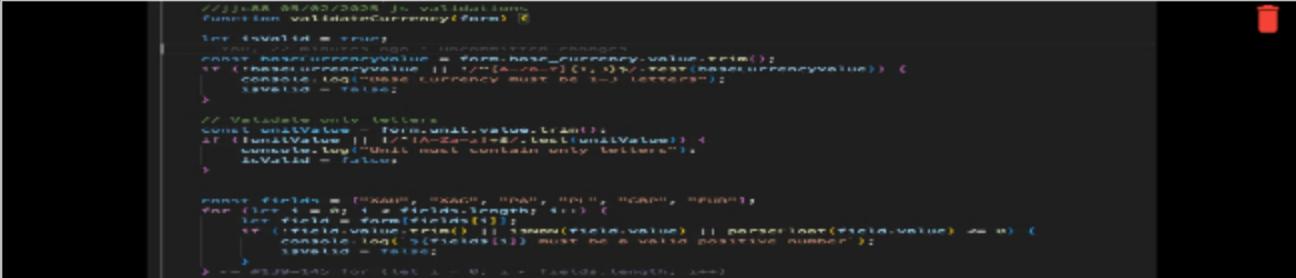
**Details:**

- Show the html validations (code)
- Show the javascript validations (code)
- Show the php validations (code)



```
</html>
<html>
  <head>
    <title>Validate Currency Form</title>
  </head>
  <body>
    <form id="validateCurrencyForm" onsubmit="return validateCurrencyForm(this)">
      <label>Base Currency:</label>
      <input type="text" name="baseCurrencyValue" value="USD" required="required" />
      <label>Target Currency:</label>
      <input type="text" name="targetCurrencyValue" value="EUR" required="required" />
      <label>Amount:</label>
      <input type="text" name="amountValue" value="100" required="required" />
      <input type="button" value="Convert" />
    </form>
  </body>
</html>
```

html



```
// Validate base currency
function validateBaseCurrency(baseCurrencyValue) {
  let invalid = true;
  const baseCurrencyValueTrimmed = baseCurrencyValue.trim();
  if (!baseCurrencyValueTrimmed || !baseCurrencyValueTrimmed.length) {
    console.log("Base currency must be non-empty");
    invalid = false;
  }
  // Validate target currency
  const targetCurrencyValueTrimmed = targetCurrencyValue.trim();
  if (!targetCurrencyValueTrimmed || !targetCurrencyValueTrimmed.length) {
    console.log("Target currency must contain only letters");
    invalid = false;
  }
  // Validate amount
  const amountValueTrimmed = amountValue.trim();
  if (!amountValueTrimmed || !amountValueTrimmed.length) {
    console.log("Amount must be a positive number");
    invalid = false;
  }
  return !invalid;
}

// Validate target currency
function validateTargetCurrency(targetCurrencyValue) {
  let invalid = true;
  const targetCurrencyValueTrimmed = targetCurrencyValue.trim();
  if (!targetCurrencyValueTrimmed || !targetCurrencyValueTrimmed.length) {
    console.log("Target currency must be non-empty");
    invalid = false;
  }
  return !invalid;
}

// Validate amount
function validateAmount(amountValue) {
  let invalid = true;
  const amountValueTrimmed = amountValue.trim();
  if (!amountValueTrimmed || !amountValueTrimmed.length) {
    console.log("Amount must be a positive number");
    invalid = false;
  }
  return !invalid;
}

// Validate form
function validateCurrencyForm(form) {
  let invalid = true;
  const baseCurrencyValue = form.baseCurrencyValue.value;
  const targetCurrencyValue = form.targetCurrencyValue.value;
  const amountValue = form.amountValue.value;
  if (!validateBaseCurrency(baseCurrencyValue)) {
    invalid = false;
  }
  if (!validateTargetCurrency(targetCurrencyValue)) {
    invalid = false;
  }
  if (!validateAmount(amountValue)) {
    invalid = false;
  }
  return !invalid;
}
```

js



A screenshot of a code editor displaying PHP code. The code includes imports for 'Currency' and 'Unit' classes, and defines a 'CurrencyConverter' class with methods for conversion logic. A 'main' function is also present.

php



Saved: 5/2/2025 1:14:27 PM

## 📝 Text Prompt

**Weight:** 50%

**Details:**

- Briefly explain the html validations used
- Briefly explain the javascript validation logic
- Briefly explain the php validation logic

**Your Response:**

1. The HTML form uses the required attribute to ensure certain fields must be filled before submission and uses input types like number to validate numeric fields
2. Javascript checks that the base currency contains 1-3 letters, the unit contains only letters

100%

## Task #2 ( 0.60 pts.) - Examples

## 📷 Image Prompt

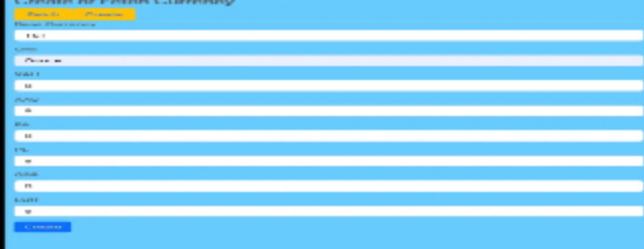
**Weight:** 40%

**Objective:** Examples

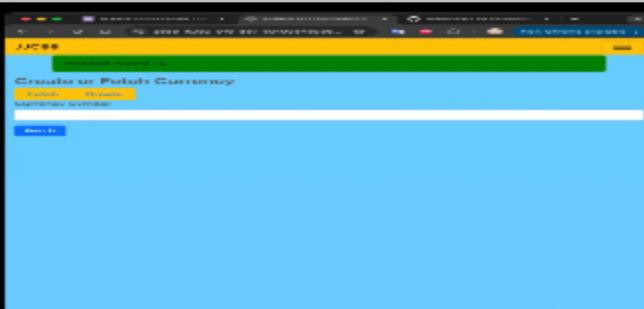
**Details:**

- Show a successful creation
- Demonstrate various validation errors
- Ensure heroku url is visible





success 1/2



success 2/2



## Task #3 ( 0.30 pts.) - Links

### Url Prompt

**Weight:** 20%

**Objective:** *Links*

**Details:**

- Include the heroku prod link to this page
- Include pull request link for this feature

URL #1

<https://jjc88-it202-010-prod->

[b44e1de4d69c.herokuapp.com/project/admin/create\\_currency.php](https://b44e1de4d69c.herokuapp.com/project/admin/create_currency.php)



URL

[https://jjc88-it202-010-prod-b44e1de4d69c.herokuapp.com/project/admin/create\\_currency.php](https://jjc88-it202-010-prod-b44e1de4d69c.herokuapp.com/project/admin/create_currency.php)



URL #2

<https://github.com/juicejoose/jjc88-it202-010-024>



URL

<https://github.com/juicejoose/jjc88-it202-010-024>



Saved: 5/2/2025 12:04:48 PM

100%

# Section #3: ( 1.5 pts.) Data List Page (Many Items)

100%

## Task #1 ( 0.60 pts.) - Code

### Combo Task:

**Weight:** 40%

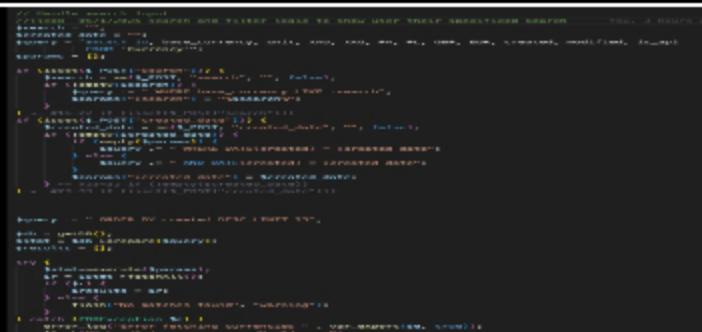
**Objective:** Code

### ☞ Image Prompt

**Weight:** 50%

#### Details:

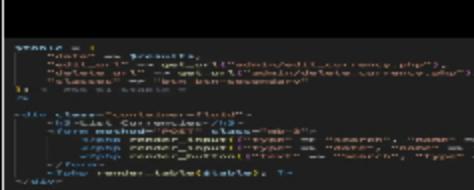
- Show the code that handles the filter/sort logic
- Show the code that generates the list/grid output
- Styling must be applied



```
/* This file is part of the React Native Data Grid package.
 * Copyright (C) 2023-present Vercel Inc.
 * You should have received a copy of the MIT license along with this package.
 * If not, see https://github.com/vercel/react-native-data-grid/blob/main/LICENSE. */

// This file contains the implementation of the DataGrid component and its
// various sub-components. It includes logic for handling data, rendering
// rows, and managing state. It also includes styling logic using CSS-in-JS.
```

filter/sort logic



```
/* This file is part of the React Native Data Grid package.
 * Copyright (C) 2023-present Vercel Inc.
 * You should have received a copy of the MIT license along with this package.
 * If not, see https://github.com/vercel/react-native-data-grid/blob/main/LICENSE. */

// This file contains the implementation of the DataGrid component and its
// various sub-components. It includes logic for handling data, rendering
// rows, and managing state. It also includes styling logic using CSS-in-JS.
```

list/table code





Saved: 5/1/2025 9:24:37 PM

## Text Prompt

**Weight:** 50%

**Details:**

- Briefly explain how the filter/sort logic works and formulates the results
- Briefly explain how the list/grid output is generated
- Briefly note the styling choices

**Your Response:**

1. The filter/sort logic is a simple if logic to check and match the inputed fields like name and data.
2. Data is fetched using query and is passed render table to display them in a table.
3. Using render functions for table and inputfields, buttons, and coloring.



Saved: 5/1/2025 9:24:37 PM



## Task #2 ( 0.60 pts.) - Examples

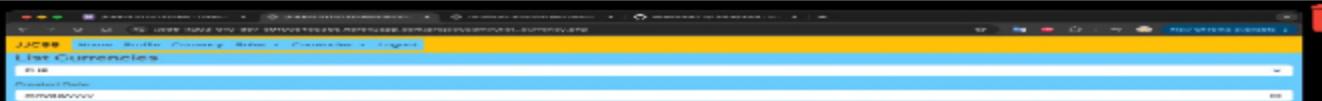
### Image Prompt

**Weight:** 40%

**Objective:** *Examples*

**Details:**

- Show a few variations of filter/sort applied (including no records)
- Each list item should be a summary
- Each list item should have the following links
  - A link to a single view of the specific entity (i.e., a details page)
  - A link to delete this entity (this may be an admin-only functionality, but it should be present for the respective role)
  - A link to edit this entity (this may be an admin-only functionality, but it should be present for the respective role)
- Ensure heroku url is visible



Row ID	Customer ID	Customer Name	Address	City	State	Zip Code	Phone Number	Email Address	Order Status	Order Date	Order Total	Order Type	Order Status	Order Date	Order Total	Order Type	Order Status	Order Date	Order Total	Order Type
1	CU001	John Doe	123 Main St	New York	NY	100-00000	(123) 456-7890	john.doe@example.com	Pending	2023-01-01	\$150.00	Physical	CU002	Jane Smith	456 Elm St	Los Angeles	CA	2023-01-02	\$200.00	Physical
2	CU002	Jane Smith	456 Elm St	Los Angeles	CA	90210-0000	(123) 456-7890	jane.smith@example.com	Pending	2023-01-01	\$150.00	Physical	CU003	Mike Johnson	789 Oak St	Chicago	IL	2023-01-03	\$180.00	Physical
3	CU003	Mike Johnson	789 Oak St	Chicago	IL	606-00000	(123) 456-7890	mike.johnson@example.com	Pending	2023-01-01	\$150.00	Physical	CU004	Sarah Williams	543 Pine St	Houston	TX	2023-01-04	\$220.00	Physical
4	CU004	Sarah Williams	543 Pine St	Houston	TX	770-00000	(123) 456-7890	sarah.williams@example.com	Pending	2023-01-01	\$150.00	Physical	CU005	David Lee	234 Cedar St	Phoenix	AZ	2023-01-05	\$170.00	Physical
5	CU005	David Lee	234 Cedar St	Phoenix	AZ	850-00000	(123) 456-7890	david.lee@example.com	Pending	2023-01-01	\$150.00	Physical	CU006	Emily Davis	678 Birch St	Seattle	WA	2023-01-06	\$210.00	Physical
6	CU006	Emily Davis	678 Birch St	Seattle	WA	981-00000	(123) 456-7890	emily.davis@example.com	Pending	2023-01-01	\$150.00	Physical	CU007	Alexander Green	321 Cypress St	Baltimore	MD	2023-01-07	\$190.00	Physical
7	CU007	Alexander Green	321 Cypress St	Baltimore	MD	212-00000	(123) 456-7890	alexander.green@example.com	Pending	2023-01-01	\$150.00	Physical	CU008	Olivia Brown	897 Birch St	Tampa	FL	2023-01-08	\$230.00	Physical
8	CU008	Olivia Brown	897 Birch St	Tampa	FL	336-00000	(123) 456-7890	olivia.brown@example.com	Pending	2023-01-01	\$150.00	Physical	CU009	Christopher White	567 Cedar St	Portland	OR	2023-01-09	\$160.00	Physical
9	CU009	Christopher White	567 Cedar St	Portland	OR	972-00000	(123) 456-7890	christopher.white@example.com	Pending	2023-01-01	\$150.00	Physical	CU010	Karen Taylor	246 Cypress St	San Jose	CA	2023-01-10	\$240.00	Physical
10	CU010	Karen Taylor	246 Cypress St	San Jose	CA	951-00000	(123) 456-7890	karen.taylor@example.com	Pending	2023-01-01	\$150.00	Physical	CU011	Matthew Wilson	789 Birch St	Austin	TX	2023-01-11	\$180.00	Physical
11	CU011	Matthew Wilson	789 Birch St	Austin	TX	787-00000	(123) 456-7890	matthew.wilson@example.com	Pending	2023-01-01	\$150.00	Physical	CU012	Natalie Parker	456 Cypress St	Phoenix	AZ	2023-01-12	\$200.00	Physical
12	CU012	Natalie Parker	456 Cypress St	Phoenix	AZ	850-00000	(123) 456-7890	natalie.parker@example.com	Pending	2023-01-01	\$150.00	Physical	CU013	James Foster	123 Birch St	Seattle	WA	2023-01-13	\$170.00	Physical
13	CU013	James Foster	123 Birch St	Seattle	WA	981-00000	(123) 456-7890	james.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU014	Sarah Thompson	678 Cypress St	Baltimore	MD	2023-01-14	\$210.00	Physical
14	CU014	Sarah Thompson	678 Cypress St	Baltimore	MD	212-00000	(123) 456-7890	sarah.thompson@example.com	Pending	2023-01-01	\$150.00	Physical	CU015	David Anderson	321 Birch St	Tampa	FL	2023-01-15	\$190.00	Physical
15	CU015	David Anderson	321 Birch St	Tampa	FL	336-00000	(123) 456-7890	christopher.white@example.com	Pending	2023-01-01	\$150.00	Physical	CU016	Olivia Martinez	567 Cypress St	Portland	OR	2023-01-16	\$230.00	Physical
16	CU016	Olivia Martinez	567 Cypress St	Portland	OR	972-00000	(123) 456-7890	olivia.martinez@example.com	Pending	2023-01-01	\$150.00	Physical	CU017	Christopher Jackson	246 Birch St	San Jose	CA	2023-01-17	\$160.00	Physical
17	CU017	Christopher Jackson	246 Birch St	San Jose	CA	951-00000	(123) 456-7890	christopher.jackson@example.com	Pending	2023-01-01	\$150.00	Physical	CU018	Natalie Wilson	456 Birch St	Phoenix	AZ	2023-01-18	\$200.00	Physical
18	CU018	Natalie Wilson	456 Birch St	Phoenix	AZ	850-00000	(123) 456-7890	natalie.wilson@example.com	Pending	2023-01-01	\$150.00	Physical	CU019	James Parker	123 Cypress St	Seattle	WA	2023-01-19	\$170.00	Physical
19	CU019	James Parker	123 Cypress St	Seattle	WA	981-00000	(123) 456-7890	christopher.parker@example.com	Pending	2023-01-01	\$150.00	Physical	CU020	Sarah Foster	678 Birch St	Baltimore	MD	2023-01-20	\$210.00	Physical
20	CU020	Sarah Foster	678 Birch St	Baltimore	MD	212-00000	(123) 456-7890	sarah.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU021	David Thompson	321 Birch St	Tampa	FL	2023-01-21	\$190.00	Physical
21	CU021	David Thompson	321 Birch St	Tampa	FL	336-00000	(123) 456-7890	christopher.thompson@example.com	Pending	2023-01-01	\$150.00	Physical	CU022	Olivia Anderson	567 Birch St	Portland	OR	2023-01-22	\$230.00	Physical
22	CU022	Olivia Anderson	567 Birch St	Portland	OR	972-00000	(123) 456-7890	olivia.anderson@example.com	Pending	2023-01-01	\$150.00	Physical	CU023	Christopher Martinez	246 Birch St	San Jose	CA	2023-01-23	\$160.00	Physical
23	CU023	Christopher Martinez	246 Birch St	San Jose	CA	951-00000	(123) 456-7890	christopher.martinez@example.com	Pending	2023-01-01	\$150.00	Physical	CU024	Natalie Jackson	456 Birch St	Phoenix	AZ	2023-01-24	\$200.00	Physical
24	CU024	Natalie Jackson	456 Birch St	Phoenix	AZ	850-00000	(123) 456-7890	natalie.jackson@example.com	Pending	2023-01-01	\$150.00	Physical	CU025	James Wilson	123 Birch St	Seattle	WA	2023-01-25	\$170.00	Physical
25	CU025	James Wilson	123 Birch St	Seattle	WA	981-00000	(123) 456-7890	christopher.wilson@example.com	Pending	2023-01-01	\$150.00	Physical	CU026	Sarah Foster	678 Cypress St	Baltimore	MD	2023-01-26	\$210.00	Physical
26	CU026	Sarah Foster	678 Cypress St	Baltimore	MD	212-00000	(123) 456-7890	sarah.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU027	David Parker	321 Cypress St	Tampa	FL	2023-01-27	\$190.00	Physical
27	CU027	David Parker	321 Cypress St	Tampa	FL	336-00000	(123) 456-7890	christopher.parker@example.com	Pending	2023-01-01	\$150.00	Physical	CU028	Olivia Thompson	567 Cypress St	Portland	OR	2023-01-28	\$230.00	Physical
28	CU028	Olivia Thompson	567 Cypress St	Portland	OR	972-00000	(123) 456-7890	olivia.thompson@example.com	Pending	2023-01-01	\$150.00	Physical	CU029	Christopher Foster	246 Cypress St	San Jose	CA	2023-01-29	\$160.00	Physical
29	CU029	Christopher Foster	246 Cypress St	San Jose	CA	951-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU030	Natalie Foster	456 Birch St	Phoenix	AZ	2023-01-30	\$200.00	Physical
30	CU030	Natalie Foster	456 Birch St	Phoenix	AZ	850-00000	(123) 456-7890	natalie.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU031	James Parker	123 Cypress St	Seattle	WA	2023-01-31	\$170.00	Physical
31	CU031	James Parker	123 Cypress St	Seattle	WA	981-00000	(123) 456-7890	christopher.parker@example.com	Pending	2023-01-01	\$150.00	Physical	CU032	Sarah Anderson	678 Birch St	Baltimore	MD	2023-02-01	\$210.00	Physical
32	CU032	Sarah Anderson	678 Birch St	Baltimore	MD	212-00000	(123) 456-7890	sarah.anderson@example.com	Pending	2023-01-01	\$150.00	Physical	CU033	David Thompson	321 Birch St	Tampa	FL	2023-02-02	\$190.00	Physical
33	CU033	David Thompson	321 Birch St	Tampa	FL	336-00000	(123) 456-7890	christopher.thompson@example.com	Pending	2023-01-01	\$150.00	Physical	CU034	Olivia Wilson	567 Birch St	Portland	OR	2023-02-03	\$230.00	Physical
34	CU034	Olivia Wilson	567 Birch St	Portland	OR	972-00000	(123) 456-7890	olivia.wilson@example.com	Pending	2023-01-01	\$150.00	Physical	CU035	Christopher Parker	246 Birch St	San Jose	CA	2023-02-04	\$160.00	Physical
35	CU035	Christopher Parker	246 Birch St	San Jose	CA	951-00000	(123) 456-7890	christopher.parker@example.com	Pending	2023-01-01	\$150.00	Physical	CU036	Natalie Foster	456 Cypress St	Phoenix	AZ	2023-02-05	\$200.00	Physical
36	CU036	Natalie Foster	456 Cypress St	Phoenix	AZ	850-00000	(123) 456-7890	natalie.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU037	James Foster	123 Cypress St	Seattle	WA	2023-02-06	\$170.00	Physical
37	CU037	James Foster	123 Cypress St	Seattle	WA	981-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU038	Sarah Foster	678 Birch St	Baltimore	MD	2023-02-07	\$210.00	Physical
38	CU038	Sarah Foster	678 Birch St	Baltimore	MD	212-00000	(123) 456-7890	sarah.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU039	David Foster	321 Birch St	Tampa	FL	2023-02-08	\$190.00	Physical
39	CU039	David Foster	321 Birch St	Tampa	FL	336-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU040	Olivia Parker	567 Birch St	Portland	OR	2023-02-09	\$230.00	Physical
40	CU040	Olivia Parker	567 Birch St	Portland	OR	972-00000	(123) 456-7890	olivia.parker@example.com	Pending	2023-01-01	\$150.00	Physical	CU041	Christopher Parker	246 Birch St	San Jose	CA	2023-02-10	\$160.00	Physical
41	CU041	Christopher Parker	246 Birch St	San Jose	CA	951-00000	(123) 456-7890	christopher.parker@example.com	Pending	2023-01-01	\$150.00	Physical	CU042	Natalie Foster	456 Birch St	Phoenix	AZ	2023-02-11	\$200.00	Physical
42	CU042	Natalie Foster	456 Birch St	Phoenix	AZ	850-00000	(123) 456-7890	natalie.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU043	James Foster	123 Birch St	Seattle	WA	2023-02-12	\$170.00	Physical
43	CU043	James Foster	123 Birch St	Seattle	WA	981-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU044	Sarah Foster	678 Cypress St	Baltimore	MD	2023-02-13	\$210.00	Physical
44	CU044	Sarah Foster	678 Cypress St	Baltimore	MD	212-00000	(123) 456-7890	sarah.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU045	David Foster	321 Cypress St	Tampa	FL	2023-02-14	\$190.00	Physical
45	CU045	David Foster	321 Cypress St	Tampa	FL	336-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU046	Olivia Foster	567 Cypress St	Portland	OR	2023-02-15	\$230.00	Physical
46	CU046	Olivia Foster	567 Cypress St	Portland	OR	972-00000	(123) 456-7890	olivia.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU047	Christopher Foster	246 Cypress St	San Jose	CA	2023-02-16	\$160.00	Physical
47	CU047	Christopher Foster	246 Cypress St	San Jose	CA	951-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU048	Natalie Foster	456 Birch St	Phoenix	AZ	2023-02-17	\$200.00	Physical
48	CU048	Natalie Foster	456 Birch St	Phoenix	AZ	850-00000	(123) 456-7890	natalie.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU049	James Foster	123 Birch St	Seattle	WA	2023-02-18	\$170.00	Physical
49	CU049	James Foster	123 Birch St	Seattle	WA	981-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU050	Sarah Foster	678 Birch St	Baltimore	MD	2023-02-19	\$210.00	Physical
50	CU050	Sarah Foster	678 Birch St	Baltimore	MD	212-00000	(123) 456-7890	sarah.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU051	David Foster	321 Birch St	Tampa	FL	2023-02-20	\$190.00	Physical
51	CU051	David Foster	321 Birch St	Tampa	FL	336-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU052	Olivia Foster	567 Birch St	Portland	OR	2023-02-21	\$230.00	Physical
52	CU052	Olivia Foster	567 Birch St	Portland	OR	972-00000	(123) 456-7890	olivia.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU053	Christopher Foster	246 Birch St	San Jose	CA	2023-02-22	\$160.00	Physical
53	CU053	Christopher Foster	246 Birch St	San Jose	CA	951-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU054	Natalie Foster	456 Cypress St	Phoenix	AZ	2023-02-23	\$200.00	Physical
54	CU054	Natalie Foster	456 Cypress St	Phoenix	AZ	850-00000	(123) 456-7890	natalie.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU055	James Foster	123 Cypress St	Seattle	WA	2023-02-24	\$170.00	Physical
55	CU055	James Foster	123 Cypress St	Seattle	WA	981-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU056	Sarah Foster	678 Cypress St	Baltimore	MD	2023-02-25	\$210.00	Physical
56	CU056	Sarah Foster	678 Cypress St	Baltimore	MD	212-00000	(123) 456-7890	sarah.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU057	David Foster	321 Cypress St	Tampa	FL	2023-02-26	\$190.00	Physical
57	CU057	David Foster	321 Cypress St	Tampa	FL	336-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU058	Olivia Foster	567 Cypress St	Portland	OR	2023-02-27	\$230.00	Physical
58	CU058	Olivia Foster	567 Cypress St	Portland	OR	972-00000	(123) 456-7890	olivia.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU059	Christopher Foster	246 Cypress St	San Jose	CA	2023-02-28	\$160.00	Physical
59	CU059	Christopher Foster	246 Cypress St	San Jose	CA	951-00000	(123) 456-7890	christopher.foster@example.com	Pending	2023-01-01	\$150.00	Physical	CU060	Natalie Foster	456 Birch St	Phoenix	AZ	2023-03-01	\$200.00	Physical
60	CU060	Natalie Foster</																		

**name search**



Link Communities
Link Communities
Link Communities
Link Communities
Link Communities

**name and date search**



100%

### Task #3 ( 0.30 pts.) - Links

## ⊕ Url Prompt

**Weight: 20%**

## **Objective:** *Links*

### Details:

- Include the heroku prod link to this page
  - Include pull request link for this feature

URL #1

<https://jhc88-it202-010-prod->



<https://jjc88-it202-010-prod-b44e1de4c>



b44e1de4d69c.herokuapp.com/project/admin/list\_currency.php

## URL #2

<https://github.com/juicejoose/jjc88-it202e01l024>



<https://github.com/juicejoose/jjc88-it2>



Saved: 5/2/2025 9:58:01 AM

100%

## Section #4: ( 1.5 pts.) View Details Page (Single)

100%

## Task #1 ( 0.60 pts.) - Code

### Combo Task:

**Weight:** 40%

**Objective:** *Code*

### ≡, Image Prompt

**Weight:** 50%

**Details:**

- Show the code that handles loading of the record (and how the id retrieved for usage)
  - Include the code that handles invalid/missing ids
- Show the code that generates the entity output
- Styling must be applied

A screenshot of a code editor window displaying Java code. The code appears to be part of a larger application, possibly a Spring Boot or similar framework, dealing with entity management. It includes imports for various Spring and JPA annotations like @Entity, @Id, @GeneratedValue, @Column, @Table, @Data, @Builder, @NoArgsConstructor, @AllArgsConstructor, and @EqualsAndHashCode. There are also annotations for @Validated and @Valid. The code is organized into several classes and interfaces, with annotations like @Service, @Repository, and @Controller. The styling of the code is consistent with standard Java conventions, using white space and indentation.

[view code](#)



Saved: 5/2/2025 10:13:35 AM

### ≡, Text Prompt

**Weight:** 50%

**Details:**

- Briefly explain how the loading logic works
- Briefly explain how the entity output is generated
- Briefly note the styling choices

Your Response:

1. Checks ID from url and if its valid it will run a query on currency and saved in the array.
2. Data is loops in the currency array and for each field it will show the name and value
3. For styling used simple container fluid and same infobox and spacing from other pages.



Saved: 5/2/2025 10:13:35 AM

100%

## Task #2 ( 0.60 pts.) - Examples

### Image Prompt

**Weight:** 40%

**Objective:** *Examples*

**Details:**

- Show a few examples of different entities
- Each entity item should have more details than the summary view (if possible)
- Each entity should have the following links
  - A link to edit the single entity (this may be an admin-only thing, but it should be present for the respective role)
  - A link to delete the single entity (this may be an admin-only thing, but it should be present for the respective role)

The screenshot shows a web application interface with a header bar and a main content area. The content area displays a list of entities with their names listed vertically. At the bottom of the list are two buttons: 'Edit' and 'Delete'.

entity 1



The screenshot shows a second instance of the web application interface, identical to the one above it. It displays a list of entities and two buttons at the bottom.

entity 2



100%

## Task #3 ( 0.30 pts.) - Links

### 🔗 Url Prompt

**Weight:** 20%

**Objective:** *Links*

**Details:**

- Include the heroku prod link to this page
- Include pull request link for this feature

URL #1

<https://jjc88-it202-010-prod-b44e1de4c6d9c.herokuapp.com/project/entry.php>



URL

<https://jjc88-it202-010-prod-b44e1de4c6d9c.herokuapp.com/project/entry.php>



URL #2

<https://github.com/juicejoose/jjc88-it202-p01l024>



URL

<https://github.com/juicejoose/jjc88-it202-p01l024>



Saved: 5/2/2025 10:20:11 AM

95%

## Section #5: ( 1.5 pts.) Edit Page

100%

## Task #1 ( 0.60 pts.) - Code

**Combo Task:**

**Weight:** 40%

**Objective:** *Code*

### 🖼 Image Prompt

**Weight: 50%**

**Details:**

- Show the code that handles loading of the record (and how the id retrieved for usage)
  - Include the code that handles invalid/missing ids
- Show the code that generates the form with the correct data types and populates it
- Show the html, javascript, and php validation (should be similar to create)
- Styling must be applied

```
function loadRecord(id) {
    const url = `http://localhost:3000/api/currencies/${id}`;
    fetch(url)
        .then(response => response.json())
        .then(data => {
            const currency = data;
            const form = document.getElementById('edit-currency-form');
            const currencyNameInput = form.querySelector('#name');
            const currencySymbolInput = form.querySelector('#symbol');
            const currencyValueInput = form.querySelector('#value');

            currencyNameInput.value = currency.name;
            currencySymbolInput.value = currency.symbol;
            currencyValueInput.value = currency.value;
        })
        .catch(error => console.error(error));
}

loadRecord(1);
```

records code

```
<table border="1">
    <thead>
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Symbol</th>
            <th>Value</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>1</td>
            <td>USD</td>
            <td>$</td>
            <td>1.00</td>
        </tr>
        <tr>
            <td>2</td>
            <td>EUR</td>
            <td>€</td>
            <td>0.82</td>
        </tr>
        <tr>
            <td>3</td>
            <td>JPY</td>
            <td>¥</td>
            <td>110.00</td>
        </tr>
        <tr>
            <td>4</td>
            <td>CNY</td>
            <td>¥</td>
            <td>7.00</td>
        </tr>
        <tr>
            <td>5</td>
            <td>INR</td>
            <td>₹</td>
            <td>84.32</td>
        </tr>
        <tr>
            <td>6</td>
            <td>AED</td>
            <td>₮</td>
            <td>3.67</td>
        </tr>
        <tr>
            <td>7</td>
            <td>BHD</td>
            <td>₼</td>
            <td>0.33</td>
        </tr>
        <tr>
            <td>8</td>
            <td>KWD</td>
            <td>₼</td>
            <td>0.27</td>
        </tr>
        <tr>
            <td>9</td>
            <td>QAR</td>
            <td>₼</td>
            <td>0.25</td>
        </tr>
        <tr>
            <td>10</td>
            <td>OMR</td>
            <td>₼</td>
            <td>0.24</td>
        </tr>
    </tbody>
</table>
```

form table code

```
<div>
    <h2>Edit Currency</h2>
    <form id="edit-currency-form">
        <div>
            <label>Name:</label>
            <input type="text" name="name" value="USD" />
        </div>
        <div>
            <label>Symbol:</label>
            <input type="text" name="symbol" value="$" />
        </div>
        <div>
            <label>Value:</label>
            <input type="text" name="value" value="1.00" />
        </div>
        <div>
            <button type="submit">Save</button>
        </div>
    </form>
</div>
```

html val

```
function validateCurrency(form) {
    let isValid = true;

    const baseCurrencyValue = form.base_currency_value.value;
    if (!baseCurrencyValue || !isValid) {
        console.log("Base currency must be 100% correct");
        isValid = false;
    }

    const unitValue = form.unit_value.value;
    if (!unitValue || !isValid) {
        console.log("Unit value must contain only letters");
        isValid = false;
    }

    const factor = form.factor.value;
    for (let i = 0; i < fields.length; i++) {
        let field = form.fields[i];
        if (!field.value || !isValid || isNaN(field.value)) {
            console.log(`Field ${i} must be a valid positive number`);
            isValid = false;
        }
    }
}
```

```
return isValid;
}

function validateCurrency(form) {
    var errorCount = 0;
    var errorMessage = '';
    var currencyList = document.getElementById('currencyList');
    var currencySelect = document.getElementById('base_currency');

    if (currencySelect.value === '') {
        errorMessage += 'Base currency is required.  
';
        errorCount++;
    }

    if (errorCount === 0) {
        currencyList.innerHTML = '';
        currencyList.appendChild(currencySelect);
    }
}
```

js val

```
// Handle currency fetch and update
if (isset($_POST['base_currency'])) {
    foreach ($_POST as $k => $v) {
        if (!in_array($k, ['base_currency', 'from', 'to', 'xrate', 'rate', 'url', 'error'])) {
            unset($_POST[$k]);
        }
    }
    $quote = $_POST;
    error_log("Cleared up POST: " . var_export($quote, true));
}
foreach ($_POST as $k => $v)
```

php val



Saved: 5/2/2025 1:32:23 PM

87%

## Task #2 ( 0.60 pts.) - Examples

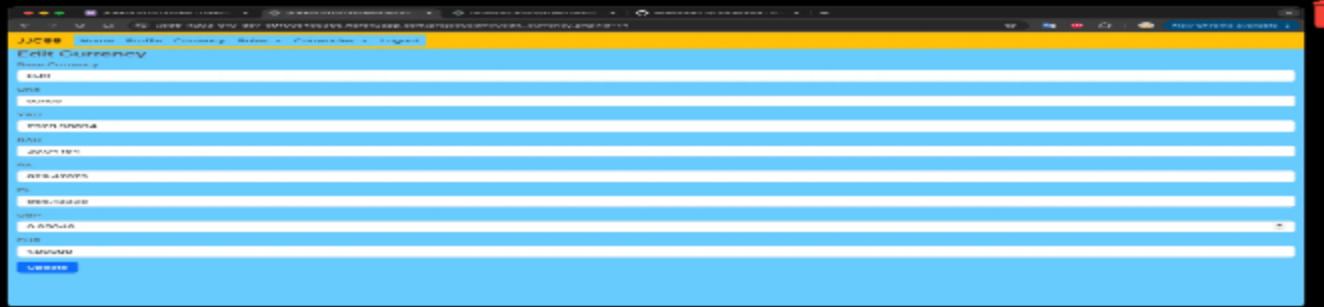
### Image Prompt

**Weight:** 40%

**Objective:** *Examples*

**Details:**

- Show a before and after of updating an entity (clearly caption)
- Successful update should have an appropriate message shown
- The updated data should be shown in the form
- Any errors should have user-friendly messages shown
- Show some examples of validation errors (html, javascript, php)



before edit

after edit

100%

## Task #3 ( 0.30 pts.) - Links

### ⊕ Url Prompt

**Weight:** 20%

**Objective:** *Links*

**Details:**

- Include the heroku prod link to this page
- Include pull request link for this feature

URL #1

[https://jjc88-it202-010-prod-b44e1de4d69c.herokuapp.com/project/edit\\_currency.php](https://jjc88-it202-010-prod-b44e1de4d69c.herokuapp.com/project/edit_currency.php)



URL

[https://jjc88-it202-010-prod-b44e1de4d69c.herokuapp.com/project/edit\\_currency.php](https://jjc88-it202-010-prod-b44e1de4d69c.herokuapp.com/project/edit_currency.php)



URL #2

<https://github.com/juicejoose/jjc88-it202-pull24>



URL

<https://github.com/juicejoose/jjc88-it202-pull24>



Saved: 5/2/2025 11:05:36 AM

100%

## Section #6: ( 1.5 pts.) Delete Logic

100%

# Task #1 ( 0.60 pts.) - Code

## Combo Task:

**Weight:** 40%

**Objective:** Code

## ☞ Image Prompt

**Weight:** 50%

**Details:**

- Show the code that handles deletion of the record (and how the id retrieved for usage)
  - Include the code that handles invalid/missing ids
- Handle any necessary roles/permissions (i.e., it's not likely that anyone can delete any entity they choose)
- Show the redirect logic that goes back to the previous page (where the route came from), if it was a list page, it should maintain the filter/sort criteria

```

1  <?php
2  require( DIR . "/partials/nav.php");
3  //only admin has permission to delete
4  if (!has_role("Admin")) {
5    flash("You don't have permission to do that", "danger");
6    die(header("Location: " . get_url("home.php")));
7  }
8
9  $id = sc($_GET["id"]); // false
10 if ($id > 0) {
11   $db = getDb();
12   $stmt = $db->prepare("DELETE FROM Currency WHERE id = :id");
13   //XXX this will hard delete
14   try {
15     $stmt->execute(":id" == $id);
16     flash("Currency deleted", "success");
17   } catch (PDOException $e) {
18     error_log("Delete error: " . var_export($e, true));
19     flash("Error deleting currency", "danger");
20   }
21 }
22 //redirect to list -- #10-21 if ($id > 0)
23 die(header("Location: " . get_url("admin/list_currency.php")));
24

```

delete code, permission, and redirect



Saved: 5/2/2025 7:23:51 PM

## ☞ Text Prompt

**Weight:** 50%

**Details:**

- Briefly explain how the deletion logic works
- Is it a soft or hard delete?
- Note any permissions/rules applied.
- Briefly explain how the redirect maintains the previous route and any filter/sort data

Briefly explain how the redirect maintains the previous route and any filter/ sort data

### Your Response:

1. It gets the id from the url and runs a delete command if the id is valid
2. This is a hard delete
3. Only admin roles can delete
4. It will redirect back to list but setting will be lost and show all results instead to reset search.



Saved: 5/2/2025 7:23:51 PM

100%

## Task #2 ( 0.60 pts.) - Examples

### Image Prompt

**Weight:** 40%

**Objective:** Examples

**Details:**

- Show a successful delete message
- Show a before and after database view of the record being deleted (clearly caption)
- Successful update should have an appropriate message shown
- The updated data should be shown in the form
- Any errors should have user-friendly messages shown

ID	Name	Description	Status	Price	Stock	Created At	Updated At	Action
1	Smartphone A	Latest model smartphone	Available	\$500	100	2023-05-01	2023-05-01	
2	Smartphone B	Mid-range smartphone	Available	\$400	150	2023-05-01	2023-05-01	
3	Smartphone C	Entry-level smartphone	Available	\$300	200	2023-05-01	2023-05-01	
4	Laptop X	High-end laptop	Available	\$1000	50	2023-05-01	2023-05-01	
5	Laptop Y	Moderate laptop	Available	\$800	120	2023-05-01	2023-05-01	
6	Laptop Z	Entry-level laptop	Available	\$600	180	2023-05-01	2023-05-01	
7	Monitor A	Large monitor	Available	\$200	50	2023-05-01	2023-05-01	
8	Monitor B	Medium monitor	Available	\$150	100	2023-05-01	2023-05-01	
9	Monitor C	Small monitor	Available	\$100	200	2023-05-01	2023-05-01	
10	Headphones A	Wireless headphones	Available	\$100	50	2023-05-01	2023-05-01	
11	Headphones B	Bluetooth headphones	Available	\$80	100	2023-05-01	2023-05-01	
12	Headphones C	Earbuds	Available	\$50	200	2023-05-01	2023-05-01	
13	Keyboard A	Full-size keyboard	Available	\$150	50	2023-05-01	2023-05-01	
14	Keyboard B	Compact keyboard	Available	\$100	100	2023-05-01	2023-05-01	
15	Mouse A	Wireless mouse	Available	\$50	50	2023-05-01	2023-05-01	
16	Mouse B	Bluetooth mouse	Available	\$40	100	2023-05-01	2023-05-01	
17	Mouse C	Trackball	Available	\$80	50	2023-05-01	2023-05-01	
18	Power Strip A	3-outlet power strip	Available	\$20	50	2023-05-01	2023-05-01	
19	Power Strip B	5-outlet power strip	Available	\$30	100	2023-05-01	2023-05-01	
20	Power Strip C	10-outlet power strip	Available	\$50	50	2023-05-01	2023-05-01	

before delete id14



ID	Name	Description	Status	Price	Stock	Created At	Updated At	Action
1	Smartphone A	Latest model smartphone	Available	\$500	100	2023-05-01	2023-05-01	
2	Smartphone B	Mid-range smartphone	Available	\$400	150	2023-05-01	2023-05-01	
3	Smartphone C	Entry-level smartphone	Available	\$300	200	2023-05-01	2023-05-01	
4	Laptop X	High-end laptop	Available	\$1000	50	2023-05-01	2023-05-01	
5	Laptop Y	Moderate laptop	Available	\$800	120	2023-05-01	2023-05-01	
6	Laptop Z	Entry-level laptop	Available	\$600	180	2023-05-01	2023-05-01	
7	Monitor A	Large monitor	Available	\$200	50	2023-05-01	2023-05-01	
8	Monitor B	Medium monitor	Available	\$150	100	2023-05-01	2023-05-01	
9	Monitor C	Small monitor	Available	\$100	200	2023-05-01	2023-05-01	
10	Headphones A	Wireless headphones	Available	\$100	50	2023-05-01	2023-05-01	
11	Headphones B	Bluetooth headphones	Available	\$80	100	2023-05-01	2023-05-01	
12	Headphones C	Earbuds	Available	\$50	200	2023-05-01	2023-05-01	
13	Keyboard A	Full-size keyboard	Available	\$150	50	2023-05-01	2023-05-01	
14	Keyboard B	Compact keyboard	Available	\$100	100	2023-05-01	2023-05-01	
15	Keyboard C	Trackball	Available	\$80	50	2023-05-01	2023-05-01	
16	Mouse A	Wireless mouse	Available	\$50	50	2023-05-01	2023-05-01	
17	Mouse B	Bluetooth mouse	Available	\$40	100	2023-05-01	2023-05-01	
18	Mouse C	Trackball	Available	\$80	50	2023-05-01	2023-05-01	
19	Power Strip A	3-outlet power strip	Available	\$20	50	2023-05-01	2023-05-01	
20	Power Strip B	5-outlet power strip	Available	\$30	100	2023-05-01	2023-05-01	
Success! Item deleted successfully.								

100%

## Task #2 ( 0.30 pts.) - Links

## TASK #3 ( 0.50 pts.) - Links

### Url Prompt

**Weight:** 20%

**Objective:** *Links*

**Details:**

- Include the heroku prod link to this page (even though it's not a full page, it's more like how logout works)
- Include pull request link for this feature

URL #1

[https://jjc88-it202-010-prod-](https://jjc88-it202-010-prod-b44e1de4c)

[b44e1de4d69c.herokuapp.com/project/delete\\_currency.php](https://b44e1de4d69c.herokuapp.com/project/delete_currency.php)



URL

<https://jjc88-it202-010-prod-b44e1de4c>



URL #2

[https://github.com/juicejoose/jjc88-](https://github.com/juicejoose/jjc88-it202-01024)



URL

<https://github.com/juicejoose/jjc88-it2>



Saved: 5/2/2025 11:07:31 AM

100%

## Section #7: ( 0.5 pts.) Misc

100%

## Task #1 ( 0.17 pts.) - Github Details

### Combo Task:

**Weight:** 33.33%

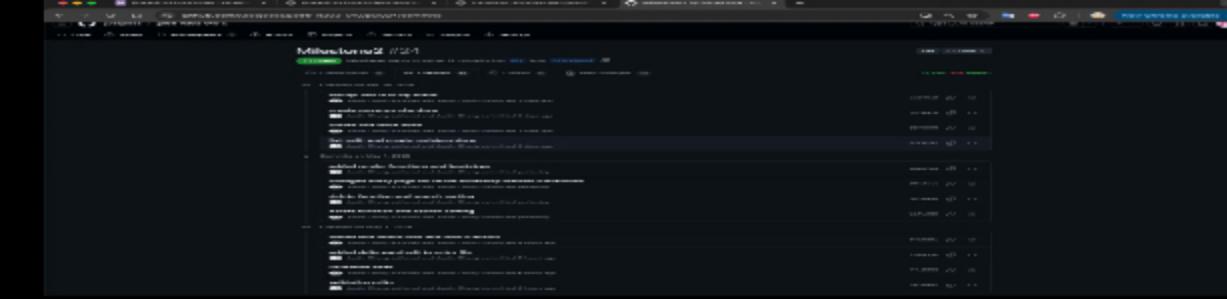
**Objective:** *Github Details*

### Image Prompt

**Weight:** 60%

**Details:**

From the Commits tab of the Pull Request screenshot the commit history



commit tab



Saved: 5/2/2025 7:29:46 PM

## Url Prompt

**Weight:** 40%

**Details:**

Include the link to the Pull Request (should end in /pull/#)

URL #1

<https://github.com/juicejoose/jjc88-it202-010/pull/24>



URL

<https://github.com/juicejoose/jjc88-it202-010/pull/24>



Saved: 5/2/2025 7:29:46 PM

100%

## Task #2 ( 0.17 pts.) - WakaTime - Activity

### Image Prompt

**Weight:** 33.33%

**Objective:** *WakaTime - Activity*

**Details:**

- Visit the [WakaTime.com Dashboard](https://wakatime.com)
- Click **Projects** and find your repository
- Capture the overall time at the top that includes the repository name
- Capture the individual time at the bottom that includes the file time
- Note: The duration isn't relevant for the grade and the visual graphs aren't necessary



wakatime

100%

## Task #3 ( 0.17 pts.) - Reflection

**Weight:** 33.33%

**Objective:** *Reflection*

### Sub-Tasks:

100%

## Task #1 ( 0.33 pts.) - What did you learn?

### Text Prompt

**Weight:** 33.33%

**Objective:** *What did you learn?*

**Details:**

Briefly answer the question (at least a few decent sentences)

Your Response:

In this milestone2 assignment I learned a lot from every concept that was covered. Concepts such as bootstrap, renderfunctions, wrapper files, filtering, and API integration were all present and used heavily in this assignment. Also concepts that were already used are used again which reinforced my skills and knowledge. Concepts such as data mapping and validations are to mention.



Saved: 5/2/2025 11:40:34 AM

100%

## Task #2 ( 0.33 pts.) - What was the easiest part of the assignr

### Text Prompt

**Weight:** 33.33%

**Objective:** *What was the easiest part of the assignment?*

**Details:**

Briefly answer the question (at least a few decent sentences)

Your Response:

Easiest part of the assignment was taking screenshots and posting links.



Saved: 5/2/2025 11:33:23 AM

100%

## Task #3 ( 0.33 pts.) - What was the hardest part of the assignr

### Text Prompt

**Weight:** 33.33%

**Objective:** *What was the hardest part of the assignment?*

**Details:**

Briefly answer the question (at least a few decent sentences)

Your Response:

I thought the hardest part of the assignment was just getting the code working with the API and understanding new concepts like bootstrap. Although the new concept like bootstrap and renderfunctions made it more overwhelming in the beginning, it became a lot easier and friendly to use. Also troubleshooting and figuring out validations for their respective data. Making new files from scratch was also hard such as making delete or view function php files to implement in our list.



Saved: 5/2/2025 7:32:19 PM