

**Mitchell Aitken**

**ENGO 551**

**Example ChatGPT Prompts for Lab 2**

### Query

What does this error mean?

WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.

\* Running on http://127.0.0.1:5000

Press CTRL+C to quit

[2024-02-06 12:19:31,963] ERROR in app: Exception on /book/0316123404 [GET]

Traceback (most recent call last):

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\venv\lib\site-packages\flask\app.py", line 1463, in wsgi\_app

response = self.full\_dispatch\_request()

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\venv\lib\site-packages\flask\app.py", line 872, in full\_dispatch\_request

rv = self.handle\_user\_exception(e)

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\venv\lib\site-packages\flask\app.py", line 870, in full\_dispatch\_request

rv = self.dispatch\_request()

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\venv\lib\site-packages\flask\app.py", line 855, in dispatch\_request

return self.ensure\_sync(self.view\_functions[rule.endpoint])(\*\*view\_args) # type: ignore[no-any-return]

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\application.py", line 275, in book

return render\_template('book.html', book=book, reviews=reviews, google\_books\_data=google\_books\_data)

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\venv\lib\site-packages\flask\templating.py", line 150, in render\_template

return \_render(app, template, context)

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\venv\lib\site-packages\flask\templating.py", line 131, in \_render

rv = template.render(context)

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\venv\lib\site-packages\jinja2\environment.py", line 1301, in render

self.environment.handle\_exception()

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\venv\lib\site-packages\jinja2\environment.py", line 936, in handle\_exception

raise rewrite\_traceback\_stack(source=source)

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\templates\book.html", line 48, in top-level template code

<p>Average Rating: {{ google\_books\_data.items[0].volumeInfo.averageRating or 'N/A' }}</p>

File "C:\Users\mitch\OneDrive\Desktop\Uni\ENGO551\lab2\venv\lib\site-packages\jinja2\environment.py", line 485, in getattr

return getattr(obj, attribute)

jinja2.exceptions.UndefinedError: builtin\_function\_or\_method object has no element 0

127.0.0.1 - - [06/Feb/2024 12:19:31] "GET /book/0316123404 HTTP/1.1" 500 -

## Response

The error `jinja2.exceptions.UndefinedError: builtin_function_or_method object has no element 0` suggests that the `google_books_data.items` call in your `book.html` template is being interpreted as a method or function, not as the data structure (likely a list) you're expecting from the Google Books API response. This typically happens when `items` is a method (such as `dict.items()` in Python) and needs to be called with parentheses to retrieve the key-value pairs from a dictionary.

To resolve this, you should ensure that `google_books_data` is structured as expected and that `items` is accessed correctly. Since `items` in the Google Books API response is a list, you need to ensure you're accessing it as such. However, given the nature of the error, it's likely that `google_books_data` is not being populated as expected, or there's a misunderstanding about how it's structured.

First, update the `get_google_books_review_data` function in `application.py` to ensure it's returning the data in the expected format.

## Query

How can I check what gets returned doing a GET request on my flask application

## Response

To check what gets returned by performing a GET request on your Flask application, you have several options. You can use tools like `curl`, Postman, or simply a web browser for simple GET requests. Additionally, you can write automated tests using Python libraries like `unittest` or `requests` to programmatically test your Flask endpoints.

### 1. Using curl

Open a terminal or command prompt and use the `curl` command to perform a GET request:

```
bash
```

```
Copy code
```

```
curl http://127.0.0.1:5000/your_route
```

Replace `http://127.0.0.1:5000/your_route` with the actual URL of the Flask route you want to test. `curl` will output the response from your Flask application to the terminal.

## Query

How can I fix this

```
ERROR: Key (isbn, user_id)=(1416949658, 1) is duplicated.could not create unique index "uix_1"
```

```
ERROR: could not create unique index "uix_1"
```

```
SQL state: 23505
```

```
Detail: Key (isbn, user_id)=(1416949658, 1) is duplicated.
```

## Response

The error message you're encountering indicates that the database operation to create a unique index (or enforce a unique constraint) named `"uix_1"` on the combination of `isbn` and `user_id` columns in your `reviews` table failed because there are existing duplicate entries for those columns. Specifically, there's at least one pair of `isbn` (1416949658) and `user_id` (1) that appears more than once in the table, which violates the uniqueness requirement.

To resolve this issue, you'll need to remove or consolidate the duplicate entries before you can successfully apply the unique constraint or index. Here's how you can approach this:

### 1. Find All Duplicates

First, identify all duplicate isbn and user\_id pairs in the reviews table. You can use a SQL query like the following to find these duplicates:

```
sql
Copy code
SELECT isbn, user_id, COUNT(*)
FROM reviews
GROUP BY isbn, user_id
HAVING COUNT(*) > 1;
```

This query groups all reviews by isbn and user\_id and counts how many reviews exist for each group. The HAVING COUNT(\*) > 1 clause filters the results to show only those groups where the count is greater than one, indicating duplicates.

### Query

Write me an SQL query that deletes all entries within reviews (I really did not need it for this haha this was a lame query with me being afraid I'd delete the whole thing)

### Response

To delete all entries from the reviews table without dropping the table itself, you can use the DELETE FROM SQL command. This command will remove all rows from the table but leave the table structure intact. Here's how you can do it:

```
sql
Copy code
DELETE FROM reviews;
```

### Query

Should this line actually work to allow only unique users for reviews? ALTER TABLE reviews ADD CONSTRAINT uix\_1 UNIQUE (isbn, user\_id);

### Response

Yes, the SQL command ALTER TABLE reviews ADD CONSTRAINT uix\_1 UNIQUE (isbn, user\_id); is syntactically correct for adding a unique constraint to a PostgreSQL table. This command is intended to ensure that each combination of isbn and user\_id in the reviews table is unique, preventing duplicate reviews for the same book by the same user.

However, the successful execution of this command depends on several factors:

**No Existing Duplicates:** Before you can add a unique constraint, there must be no existing rows in the table that violate the constraint. Since you've mentioned that you've already deleted duplicate entries, this should be covered.

**Correct Column Names:** The columns isbn and user\_id must exist in the reviews table with those exact names. If the column names are different, you'll need to adjust the command to match.

Database Permissions: The user executing the command needs to have the necessary permissions to alter the table structure.

Database Health and Configuration: The database should be in a healthy state, with no issues that might prevent schema modifications, such as excessive locking, resource constraints, or corruption.

### Query

Write me an SQL prompt that keeps only the most recent review for unique users

### Response

You can write a SQL query to keep only the most recent or the highest-rated review for each combination of isbn and user\_id, and delete the rest. For example, to keep only the most recent review:

sql

Copy code

```
DELETE FROM reviews
WHERE id NOT IN (
  SELECT MAX(id)
  FROM reviews
  GROUP BY isbn, user_id
);
```