Bug #1 — N+1 Queries in UsersController#index

Issue

The **/users** endpoint calculated **products\_count** using:

**user.products.count**

This caused an N+1 query problem, where Rails executed one query to load users and an additional query per user to count products. This results in poor performance as the dataset grows.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

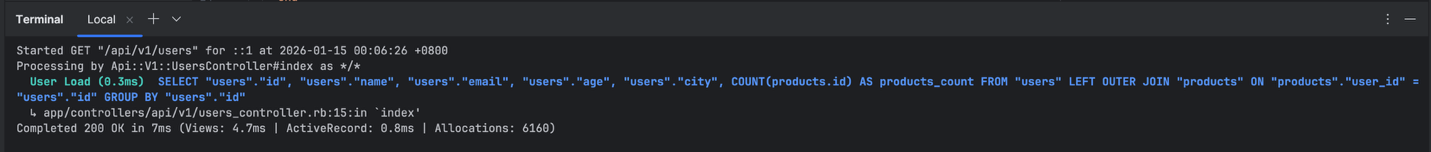
AI-generated content may be incorrect.

Fix

I replaced per-user counting with a single aggregated query using **LEFT JOIN** and **GROUP BY**.

A screen shot of a computer program

AI-generated content may be incorrect.



Bug #2 — N+1 Queries in ProductsController#index

Issue

**user\_info = User.find(product.user\_id)**

This ignores includes(:user) and causes 1 query per product.

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

Fix

Use the already-loaded product.user

A screenshot of a computer program

AI-generated content may be incorrect.

Bug #3 — Inefficient Collection Logic for is\_expensive

Issue

The products endpoint:

* Iterated through all products to build an array of expensive product IDs
* Performed duplicate checks using **include?**

A screen shot of a computer program

AI-generated content may be incorrect.

Fix

Use the database-level filtering and the existing scope: **Product.expensive**

A screenshot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

Bug #4 — Wrong Data Type in Query

Issue

The query used a string conversion for **user\_id**, but **user\_id** is integer.

**Product.where("user\_id = ?", params[:id].to\_s)**

This caused unnecessary type casting and could lead to inefficient query plans.

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

Fix

Replaced it with query: **@user.products**

A screenshot of a computer program

AI-generated content may be incorrect.

Bug #5 — Weather Temperature Unit Assumption

Issue

Mock data mixes Fahrenheit (US cities) and Celsius (EU cities)  
But code assumes everything is Celsius

**temperature\_celsius: temperature**

This produces wrong results

A screen shot of a computer program

AI-generated content may be incorrect.

Fix

Normalize temperatures properly:

* US cities → Fahrenheit → convert to Celsius
* EU cities → already Celsius

A screenshot of a computer program

AI-generated content may be incorrect.

Bug #6 — Data Security Improvement

Issue

Data exposure on user endpoint. This exposes all columns.

A screen shot of a computer program

AI-generated content may be incorrect.

Fix

Explicit JSON response.

A screen shot of a computer program

AI-generated content may be incorrect.

Bug #7 — Error handling improvement on WeatherController

Issue

When searching invalid city, it will still response as valid city

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Fix

Adding timeout condition on weather service for weather service unavailable 503 and if any unknown city, it will return nil or invalid data weather 502.

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer error

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.