



# Advanced Recipe Database

Kyle Watson  
Luke Unterman  
Paula Gearon

# Problem: Most Recipe Websites Suck

- Highest-ranked web pages retrieved by a search engine (e.g., Google) are:
  - Heavily bloated with extraneous information (see Figure 1)
  - Polluted with unskippable advertisements that detract from overall user experience
- They also frequently lack support for
  - Unit conversions (e.g., ounce => gram)
  - Ingredient substitutions
- [Band-aid solutions](#) do not solve root issue:
  - Website design itself is flawed
  - Enter: RRDBMS (Recipe Relational Database Management System)

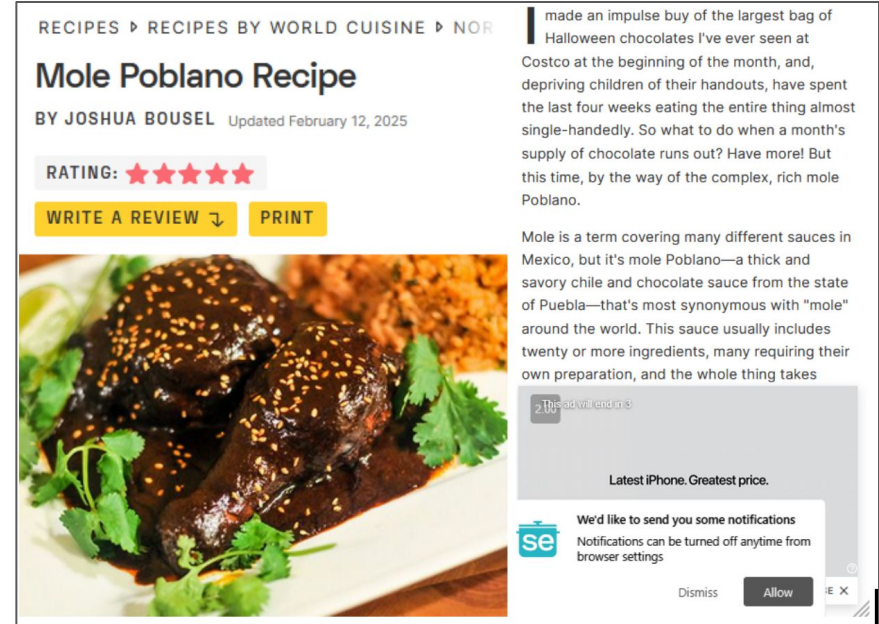


Figure 1. [Mole Poblano recipe](#) viewed on iPhone 13 Pro Max.

- Left: recipe with image thumbnail.
- Right: image of 316 word preamble (obscured by video ad) preceding actual recipe about author's personal journey in creating this recipe.

# Proposed Design: Recipe RDBMS

- Advanced recipe database with searchable, interlinked, and flexible recipe structure
  - Don't have a particular recipe ingredient? Replace it with a valid substitution
  - Prefer the metric system? Each unit has a list of potential conversions.
- Image similarity search:
  - Given any input image, find recipes with similar images
- Text similarity search:
  - Given any input sentence, find recipes with similar descriptions/steps/titles
- Simplistic, easily navigable website design

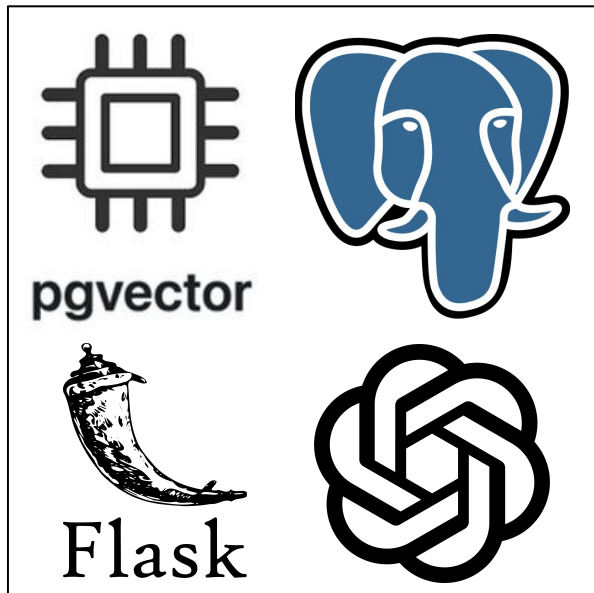


Figure 2. Recipe RDBMS tech-stack

- Using PostgreSQL database to store our data, with pgvector extension to store and index embeddings. Image + text embeddings created using OpenAI CLIP model and MPNet, respectively. We use Flask to host our website.

# Principle Entities

The principle entities are:

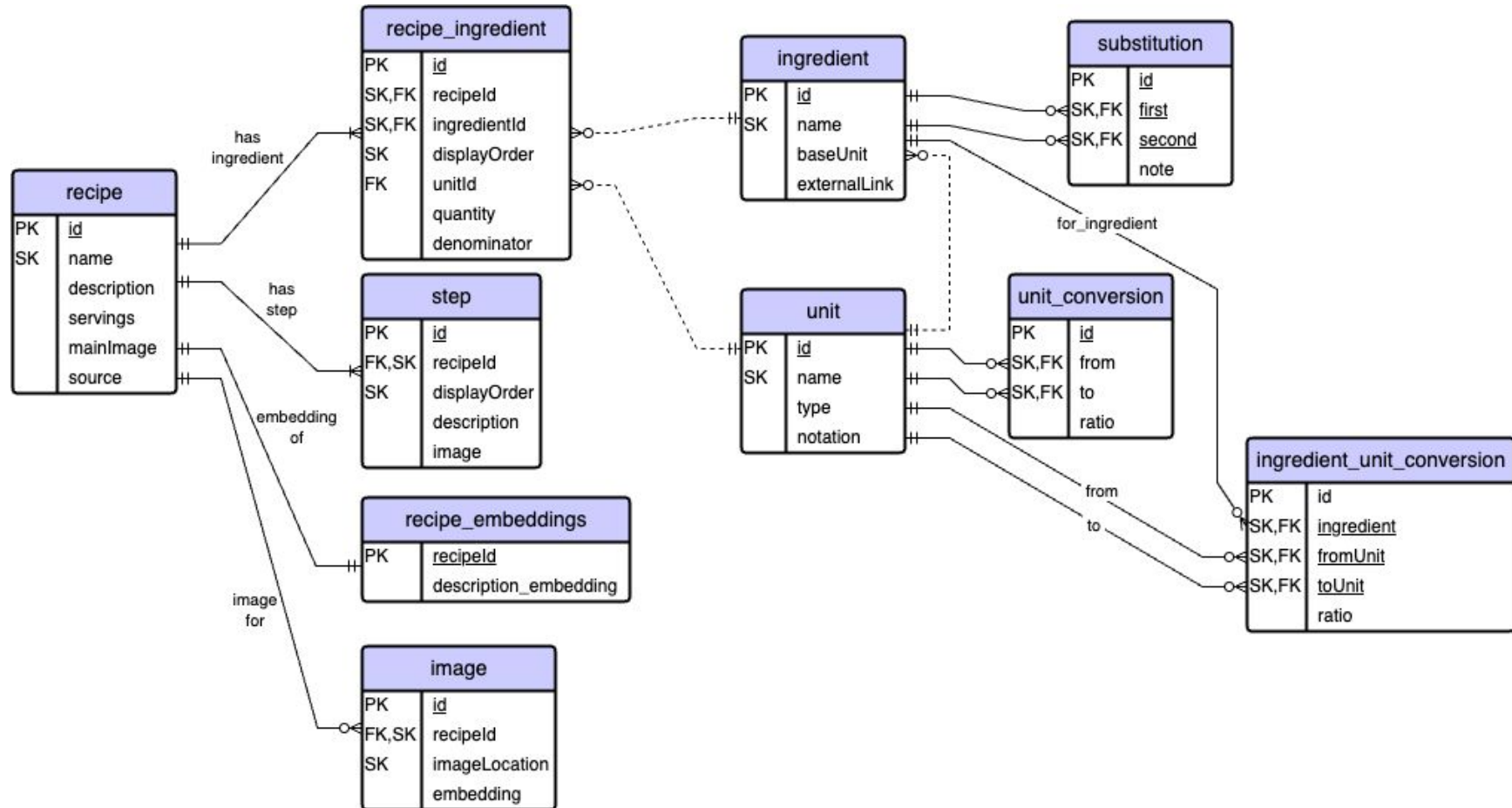
- Recipe
- Ingredients
- Units
- Images

# Supporting Entities

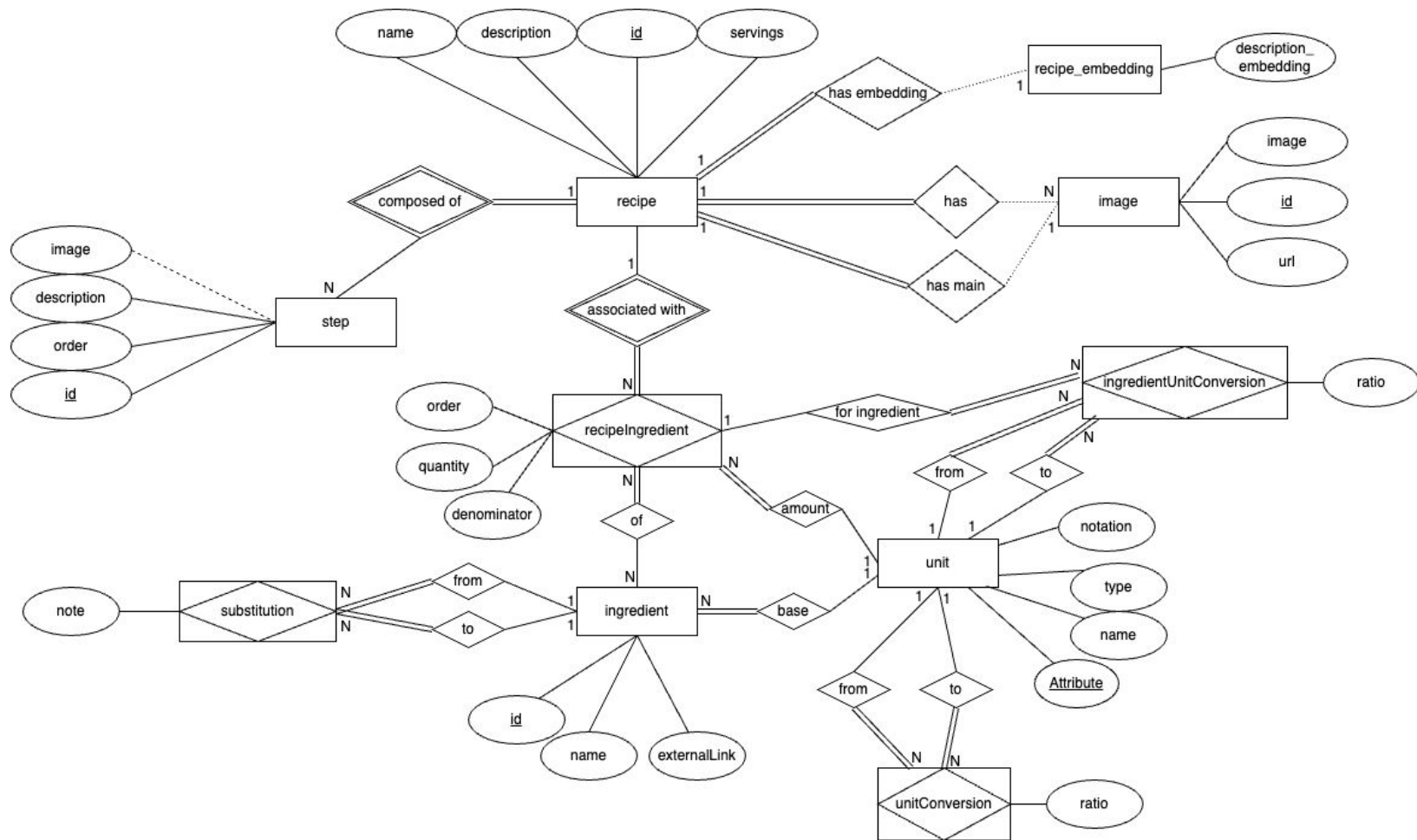
Other supporting entities are:

- Recipe Ingredients
- Unit Conversions
- Ingredient Unit Conversions (e.g. 1 cup flour  $\equiv$  125g flour)
- Unit Types
- Recipe Steps
- Embedding Vectors

# Crow's Foot ER Diagram



# Chen ER Diagram



# Data Acquisition + Preprocessing

- Using Python script, retrieve all SeriousEats URLs from XML [sitemap](#) ending with “recipe”
  - For each website, scrape JSON-LD object containing structured recipe information
- Preprocess dumped JSON object to only contain required fields (recipe steps, ingredients, description, etc.)
  - For each recipe ingredient:
    - Use Python script to connect to Google Gemini 1.5 Flash LLM
    - Prompt LLM to split ingredient into quantity, unit, and food
      - “1 cup broccoli florets” => {Q: 1, U: Cup, F: broccoli florets}
- Iteratively create SQL script to insert values into our database
- **Note:** this was probably more trouble than it was worth



# Image Search

- Using CLIP model to convert recipe images to embedding vectors.

*“CLIP is a multimodal vision and language model motivated by overcoming the fixed number of object categories when training a computer vision model.”*

- Stored with datatype `vector(512)` provided by *pgvector*

```
SELECT image.id, recipeId, imageLocation,  
       (embedding <=> %s::vector(512)) AS cosine_distance,  
       recipe.name as recipe_name  
FROM image  
JOIN recipe on image.recipeid = recipe.id  
WHERE embedding IS NOT NULL  
ORDER BY cosine_distance  
LIMIT 5;
```

# Fuzzy and Advanced Text Search

- Fuzzy search
  - Similar to image search
  - Combined name, description, ingredients, and steps into a vector “embedding”
  - Compare query embedding to saved embeddings
  - Return sorted results based on embedding similarity
- Advanced search
  - Matches exact names, descriptions, ingredients, and steps

## Recipe Search

[advanced search](#)

## Advanced Recipe Search

Name:

Ingredients:

Steps:

Description:

[basic search](#)

# Text Search Implementation

- “Embed” text descriptions using a BERT-based model
  - SBERT
- Stored with datatype: `vector(768)`
- Datatype provided by *pgvector* extension to Postgres



```
SELECT id, name, mainimage, description from recipe
INNER JOIN recipe_embeddings on
recipe.id=recipe_embeddings.recipeid
ORDER BY description_embedding <#> %s::vector ASC LIMIT 10"
```

# Flask



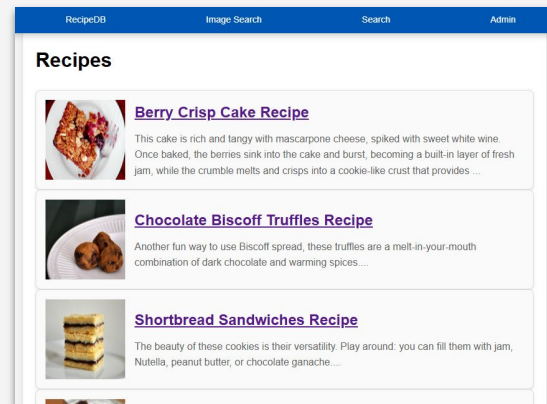
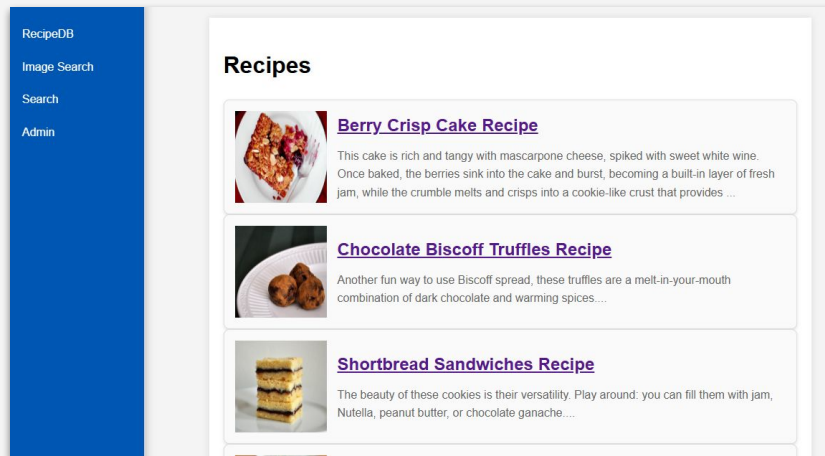
# Flask

A simple Python framework for building Web applications and services.

- Static Files
  - HTML, CSS, JS, image
- Generated pages
  - Jinja Templates
- Web Services

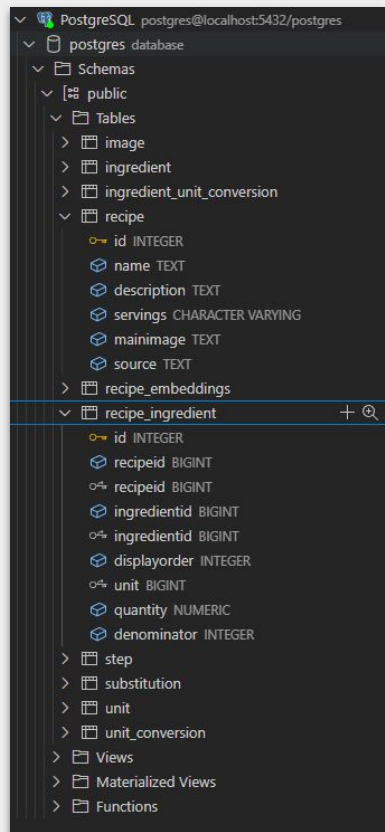
# Website Design and Styling

- /index page lists several recipes
- Navbar on every page to link to main pages
- Advanced search inside the text search tab
- Navbar moves to the top of the screen when resolution is too narrow
- Recipe 'cards' summarize and link to recipe page
  - Scale when hovered



DEMO

# SQL Tools - VSCode Extension



- Connect to databases (not just PostgreSQL)
- View tables and properties
- Useful to have next to code and when crafting SQL queries