STEP 1, BRAINSTORMING
What kind of data would I need for this?
-users
-recipes
-email
-passwords
Whether or not a user wants something to be visible
Ingredients
Shopping lists
Occasions and recipes assigned to them

STEP 2 TABLE IDEAS
What kind of tables should we have?US
User- user functionality

- Username
- Password
- Email
- recipes

Recipes-information about the recipes

- Ingredients
- Recipe name
- Text body/instructions
- Visibility

Shopping list- what you want to buy at the store

- Ingredients

Occasions

- Recipes

I didn't have any additional tables to add

STEP 3 RELATIONSHIPS
One-to-One

- Shopping list to user, since there's only one list per one user

One-to-Many

- Recipes to shopping list, because there's only going to be one list no matter how many recipes you're buying for
- Recipes to users, since there's multiple recipes each user could create

Many-to-Many

- Occasions to recipes, since you might want multiple recipes for one occasion, but also there could be multiple occasions that share the same recipe.

PART 2 COLUMNS
I skipped a lot of these because the answers are just common sense, and I'm too sick to bother filling out repetitious stuff, sorry. CS stands for Common Sense, in those cases- the titles are meant to be self-explanatory, after all, and so typically they are.

Users:

User_id: CS. uses serial and primary key for uniqueness.

User_passwords: CS. chose VARCHAR 55 since I don't want people's passwords to be stored too long when hashed.

User_email: CS, VARCHAR again since I don't want to let people write books for their emails.

Recipes: stores the recipes the user has on file. VARCHAR again, since this should just be the titles of their recipes.

Recipes:

Ingredients: CS, VARCHAR

Recipe_name: CS, VARCHAR

User_id: so you can keep track of who owns the recipe

Instructions: CS, Text since IDK how complex the instructions are/

Visibility: Boolean, so users can mark if their recipes are for personal use only or to share with the world.

Shopping List:

List_id: in case you need multiple lists at the same time

User_id: to keep track of which user needs the list

Ingredients: so you know what to buy

‘

Occasions:

Occasion id: in case there's multiple occasions

User_id: so you can keep track of who the occasions belong to

Recipes: literally just what recipes are set for each occasion

SQL

```
CREATE TABLE users(
user_id SERIAL PRIMARY KEY,
password VARCHAR(55),
email VARCHAR(100)
);

CREATE TABLE recipes(
recipe_id SERIAL PRIMARY KEY,
ingredients TEXT UNIQUE,
recipe_name VARCHAR(55),
user_id INTEGER REFERENCES users(user_id),
instructions TEXT,
visibility BOOLEAN);
```

```
CREATE TABLE shopping_list(
  list_id SERIAL PRIMARY KEY,
  user_id INTEGER REFERENCES users(user_id),
  ingredients TEXT REFERENCES recipes(ingredients));

CREATE TABLE occasions(
  occasion_id SERIAL PRIMARY KEY,
  user_id INTEGER REFERENCES users(user_id),
  recipes INTEGER REFERENCES recipes(recipe_id)
  );
```