Brainstorming

Features:

- users can sign into the app with their email and password
- users can create recipes with ingredients and instructions
- recipes can be marked as public or private
- users can view other people's recipes
- ingredients from recipes can be added to user's grocery lists
- users can create their own occasions and assign recipes to occasions

Data:

- Username
- Password
- First name
- Last name
- Email
- Ingredients
- Instructions (cook time, prep time, cutting, boiling, measurements, etc)
- Recipe id
- User who created recipe
- Recipe name
- Public or private recipe
- Who viewed your recipe
- Grocery list name
- Occasions

Data:

- email
- Password
- Users
- Recipes
- Ingredients
- Occasions
- Permissions

Table Ideas:

1. Users: This table will hold user information for login/verification purposes
    a. User_id
    b. user_name
    c. First_name

d. Last_name
e. Email
f. Password

2. Recipe: This table will hold Recipe Id's for unique recipes, ingredients, and an individual or multiple authors depending on permissions and visibility.
    a. Recipe_id
        i. We need recipe ID in order to have a unique recipe number
        ii. SERIAL so that it's unique
    b. Instructions
        i. We need written instructions for the recipe
        ii. This is was it is a VARCHAR
    c. Ingredients
        i.
    d. Recipe_name
    e. occasion_id
    f. User_id
    g. Permissions
    h. Visibility (public or private)

3. Occasion: This table will hold occasions that users create and the recipes they assign/attach to the occasion.
    a. occasion _id
    b. Occasion_name
    c. User_id
    d. Recipe_id
    e. Grocery list
4. Grocery list: This table will name all the ingredients for a recipe and/or occasion
    a. groceryList_id
    b. groceryList_name
    c. Occasion_id
    d. User_id
    e. Ingredients
5. Ingredient list: This will name all the ingredients with a dedicated id.
    a. Ingredient_id
    b. ingredient_ name
    c. Recipe_id
    d.

Relationships

One-to-one
    Users - Recipes (private ones)
One-to-many
    Recipe Occasion (occasions not required in recipe)
Many-to-many
    Users - Recipes (public/collaborative ones)
    Users - Occasions
    Recipe - Occasion (several recipes)

Columns

Users:

- User_id
    - Gives users ability to have unique Id
    - SERIAL: so that it's unique
- Username
    - Gives users ability to login with username
    - VARCHAR: user can create username
- First_name
    - Simple info needed for app registration
    - VARCHAR: user can enter first name
- Last_name
    - Simple info Needed for app registration
    - VARCHAR: user can enter last name
- Email
    - Connecting user for authentication
    - VARCHAR: user can enter email
- Password
    - Allows user to login
    - VARCHAR: user can make strong password

Recipe:

e. Recipe_id
    i. We need recipe ID in order to have a unique recipe number
    ii. SERIAL so that it's unique
f. Instructions
    i. We need written instructions for the recipe
    ii. This is was it is a VARCHAR
g. ingredients_ID
    i. Need specific ingredients
    ii. INT because we are referencing a foreign ID

h. Recipe_name
    i. Need a name for the recipe
    ii. VARCHAR to name it
i. Occasion_id
    i. Need a specific occasion ID.
    ii. INT because we are referencing a foreign key.
j. User_id
    i. Need to reference a unique user
    ii. INT because we reference a foreign key
k. Permissions
    i. Does the user have permission to edit?
    ii. Boolean to check if the user does or doesnt
l. Visibility (public or private)
    i. Is the Recipe public or private?
    ii. Boolean to determine if public or private

Grocery List:

- Grocerylist_id
    - To keep each grocery list separate
    - Serial: so it's unique
- Grocerylist_name
    - Gives user ability to infer what is on the list
    - VARCHAR: to name it
- User_id
    - Needs to reference a unique user
    - INT: referencing a foreign key
- Occasion_id
    - List can be associated with ingredients that are for a specific occasion
    - INT: referencing a foreign key
- Ingredient_name
    - List of names for the ingredients
    - VARCHAR long enough for list

Ingredient list:

m. Ingredient_id
    i. The ingredient has to be unique (mozzarella cheese, purple grapes)
    ii. SERIAL because it is unique
n. ingredient_ name
    i. it could just be grapes
    ii. VARCHAR to write
o. Recipe_id

i. Ingredients could have several recipes
ii. INT to reference foreign key

Occasion:

- Occasion_id
    - Needs a unique id for a unique occasion
    - SERIAL to not have a repeated occasion

- Occasion_name
    - Needs a name for it
    - VARCHAR to write it
- User_id
    - Needs to be a user who is creating occasion
    - INT: to reference foreign key
- Recipe_id
    - Recipe can be used for this specific occasion
    - INT: to reference foreign key

```
-- CREATE TABLE users (
--   user_id SERIAL PRIMARY KEY,
--   username VARCHAR(128) NOT NULL,
--   first_name VARCHAR (128) NOT NULL,
--   last_name VARCHAR (128) NOT NULL,
--   email VARCHAR (128) NOT NULL,
--   password VARCHAR(128) NOT NULL
--   );

-- CREATE TABLE ingredients(
--   ingredient_id SERIAL PRIMARY KEY,
--   ingredient_name VARCHAR (255)
-- --   recipe_id INT REFERENCES recipe(recipe_id)
--   );

-- ALTER TABLE ingredients
-- ADD recipe_id INT REFERENCES recipe(recipe_id)

-- CREATE TABLE recipe (
--   recipe_id SERIAL PRIMARY KEY,
--   user_id INT REFERENCES users(user_id),
--   instructions VARCHAR(10000) NOT NULL,
--   ingredient_id INT REFERENCES ingredients(ingredient_id),
--   permissions BOOLEAN NOT NULL,
--   private BOOLEAN NOT NULL
```

```
--   );


-- CREATE TABLE occasion (
--   occasion_id SERIAL PRIMARY KEY,
--   occasion_name VARCHAR (1082),
--   user_id INT REFERENCES users(user_id),
--   recipe_id INT REFERENCES recipe(recipe_id)
--   );

-- CREATE TABLE grocery_list (
--   grocerylist_id SERIAL PRIMARY KEY,
--   grocerylist_name VARCHAR(255),
--   user_id INT REFERENCES users(user_id),
--   occasion_id INT REFERENCES occasion(occasion_id),
--   ingredient_name VARCHAR(255) NOT NULL
--   );

-- ALTER TABLE grocery_list
-- ADD ingredient_name VARCHAR (255) REFERENCES ingredients(ingredient_name);


--   ALTER TABLE ingredients
--   ALTER COLUMN ingredient_name VARCHAR (255)PRIMARY KEY NOT NULL;
```