Brainstorming:

- User data
- Recipe data
- Ingredients
- Instructions
- Private or public
- Profiles
- Grocery lists
- Occasions
- Recipes for occasions

Table Ideas:

- User: store's user data like email and password, etc.
- Recipe's: stores recipe's as a finished product
- Ingredients: stores ingredients that can be added to recipes
- Instructions: stores instructions that can be added to recipes
- User Profiles: profile containing limited user info and public recipe's
- Grocery Lists: stores user ingredient lists from recipes
- Occasions: place to store individual occasions
- OccasionRecipe's: place to store recipes in association with occasions
- PrivateRecipe's: recipes marked private will be here
- Followers: User can follow other user profiles and their saved public recipes
- GroupOccasion: A table that pulls from occasions and users to allow for group occasions

Relationships:

One to one relationships:

- User->profile: One profile per user

One to many relationships:

- Recipe -> instructions: A recipe will have many instructions, but the instructions will tie back to one recipe.
- Recipe -> ingredients: A recipe will have many ingredients, but the ingredients will make up one recipe.
- User -> Recipe: A user can save multiple recipes
- User -> follower: A user can follow multiple other users
- User-> Occasion: User's can have multiple occasions
- Grocery List -> Ingredients. A grocery list can have many ingredients.
- Occasions -> Recipe: Multiple recipes can be added to an occasion.
- Users -> Private Recipes: One user can have many private recipes.

Many to many relationships:

- GroupOccasion -> users and occasions and recipes: many users can add many recipes to occasions

Columns:

- User
    - User_id: primary key - integer
    - user_email/password: storing to allow user to sign into their profile varchar to limit how long it can be
    - user_name/pic: basic info about the user to display for their profile. String for name and pic link
- Private_recipes:
    - Private_id: generate a serial primary key for the table
    - User_id: foreign key linking to user
    - Private_recipe_name: varchar that allows to name private recipe
    - Ingredients_id: id linking to ingredient to be added to private recipe
    - Instructions_id: id linking to instructions for private recipe
- Recipes:
    - Recipe_id: generating a serial primary key for each recipe
    - User_id: linking to user table
    - Recipe_name: varchar that names recipe
    - Ingredients_id: id linking to ingredient to be added to recipe
    - Instructions_id: id linking to instructions to be added to recipe
- Ingredients:
    - Ingredient_id: generating a serial primary key for each ingredient
    - Ingredients: varchar space for adding in the name of each ingredient
- Instructions:
    - Instructions_id: generating a serial primary key for each set of instructions
    - Instructions: varchar allowing you to add each step in your instructions, this allows the user to re-use the steps they do frequently without storing duplicate data
- Occasions:
    - Occasion_id: generating a serial primary key for each occasion
    - User_id: linking to user table
    - Occasion_recipe_id: links to table occasion_recipes table to link recipes to occasions
    - Occasion_datetime: datetime data for storing the date and time of the occasion
    - Occasion_location: varchar for storing where occasion is held
- Occasions_recipes:
    - Occasion_recipe_id: generates a serial primary key for recipes being added to occasions
    - Occasion_id: links the occasion id to the occasion recipes table

    - Recipe_id: links the recipes to be added to the occasion recipes table
- Group_Occasion
    - Group_occassion_id: generate a serial primary key for the table
    - Occasion_id: integer tying back to the occasion
    - User_id: integer tying back to the users in attendance
    - Ocassion_recipe_id: integer tying back to the specific recipes for that occasion.
- User_profile
    - Profile_id: generate a serial primary key for the table, integer
    - User_id: foreign key to access the user the profile is associated with
    - Recipe_id: access the recipe table to allow the user to have recipes posted on their profile.
    - Grocery_list_id links to grocery list to have it stored on your profile
- Gocery_list
    - Grocery_id: generate a serial primary key for each list, integer
    - Recipe_id: access the recipe to access the ingredients needed for the grocery list
    - Grocery_list_name: be able to name it and store it in your profile
- Followers
    - follower_id : generate a serial primary key for the table
    - User_id: to list the users they are following
    - Followers: other people ID that are following them as a user
    - Following: other IDs that the user is following.
-

Create table statements:

```
CREATE TABLE users (
        user_id SERIAL PRIMARY KEY NOT NULL,
        email varchar(255) NOT NULL UNIQUE,
        password varchar(255) NOT NULL,
        username varchar(255) NOT NULL UNIQUE,
        f_name varchar(255) NOT NULL,
        l_name varchar(255) NOT NULL,
        profile_picture VARCHAR(255),

);



CREATE TABLE group_occasions (
        group_occasion_id SERIAL PRIMARY KEY NOT NULL,
        occasion_id integer NOT NULL REFERENCES occasions(occasion_id),
        user_id integer NOT NULL REFERENCES users(user_id),
```

```
	occasion_recipe_id integer NOT NULL REFERENCES
occasion_recipes(occasion_recipe_id),
);


CREATE TABLE followers (
	follower_id SERIAL PRIMARY KEY NOT NULL,
	user_id integer NOT NULL REFERENCES users(user_id),
	user_profile_id integer NOT NULL REFERENCES user_profiles(user_profile_id),
	followers integer NOT NULL,
	following integer NOT NULL,
);


CREATE TABLE private_recipes (
	private_id SERIAL PRIMARY KEY NOT NULL,
	user_id integer NOT NULL REFERENCES users(user_id),
	private_recipe_name varchar(255) NOT NULL,
	ingredients_id integer NOT NULL REFERENCES ingredients(ingredient_id),
	instructions_id integer NOT NULL REFERENCES instructions(instructions_id),
);


CREATE TABLE occasion_recipes (
	occasion_recipe_id SERIAL PRIMARY KEY NOT NULL,
	occasion_id integer NOT NULL REFERENCES occasions(occasion_id),
	recipe_id integer NOT NULL REFERENCES recipes(recipe_id),
);


CREATE TABLE occasions (
	occasion_id SERIAL PRIMARY KEY NOT NULL,
	user_id integer NOT NULL REFERENCES users(user_id),
	occasion_recipe_id integer NOT NULL REFERENCES
occasion_recipes(occasion_recipe_id),
	occasion_date_time DATETIME NOT NULL,
	occasion_location varchar(255) NOT NULL,
);


CREATE TABLE grocery_lists (
	groc_list_id SERIAL PRIMARY KEY NOT NULL,
	user_id integer NOT NULL REFERENCES users(user_id),
	groc_list_name varchar(255) NOT NULL,
```

```
        recipe_id integer NOT NULL REFERENCES recipes(recipe_id),


);


CREATE TABLE user_profiles (
        user_profile_id SERIAL PRIMARY KEY NOT NULL,
        user_id integer NOT NULL REFERENCES users(user_id),
        recipe_id integer NOT NULL REFERENCES recipes(recipe_id),

);


CREATE TABLE instructions (
        instructions_id SERIAL PRIMARY KEY NOT NULL,
        instructions varchar(500) NOT NULL,

);


CREATE TABLE ingredients (
        ingredient_id SERIAL PRIMARY KEY NOT NULL,
        ingredients varchar(500) NOT NULL,

);


CREATE TABLE recipes (
        recipe_id SERIAL PRIMARY KEY NOT NULL,
        user_id integer NOT NULL REFERENCES users(user_id),
        recipe_name varchar(255) NOT NULL,
        ingredients_id integer NOT NULL REFERENCES ingredients(ingredient_id),
        instructions_id integer NOT NULL REFERENCES instructions(instructions_id),

);
```

Insert Data:

```
INSERT INTO users (email, password, username, f_name, l_name)
values('1234weeee@gmail.com', 'bananas', 'RogerRabbit', 'Roger', 'Rabbit'),
('4567weeee@gmail.com', 'apples', 'Helloworld', 'Paula', 'Deen'), ('8910weeee@gmail.com',
'pears', 'RUThere', 'Peanut', 'Butter'), ('1112weeee@gmail.com', 'grapes', 'OvaOva', 'Vino',
'isGud');
```