Cooking App

Step 1: Brainstorm

- Username
- Email
- Name of user
- Recipes
- Instructions
- Time frame to make it
- Ingredients
- Private or public
- Grocery list
- Occasion list

Step 2: Table Ideas

- Authorization Table:
    - userID
    - Email
    - username
    - Password
- User Table:
    - UserId
    - Name
    - Age
    - DOB
- Recipe Table:
    - Recipe ID
    - userId
    - public/private
    - Pic of meal
    - Instructions
    - Ingredients
    - Time to make
    - List Id
- Grocery Table:
    - Grocery List ID
    - Recipe Id
    - Items in cart
    - Date added
- Occasions List table:

    - Occasion List Id
    - List name
    - Type
    - Approximate number of people
- List Item table:
    - List item id
    - Occasion List Id
    - Recipe Id

Step 3: Relationships:

- One-to-One:
    - Users auth: each user has unique info
    - Recipe to grocery list: each grocery list is specific to one recipe
- One to many:
    - User to recipe: one user can have many recipes, but one recipe has one owner
- Many to Many:
    - Recipe to occasion: Each recipe can have multiple occasions and each occasion can have multiple recipes.

Part 2 - Step 2:

- Authorization:
    - User_id: each user must have a unique identifier in backend
    - Username: unique name for user
    - Email: can login using username or email
    - Password: must store password(hash) so user can login
- Users:
    - user_Id: helps with login, as well as link to recipe
    - first_name/last_name: to know who this user is
    - DOB/Age: can help with pointing them towards recipes that may interest them.
- Recipes:
    - Recipe_id: each recipe needs a unique id
    - User_id: points the recipe back to the user who made it
    - Public: boolean to show if the recipe is public or private
    - Pic_of_meal: text because picture could be large
    - Ingredients: text because list could be large
    - Instructions: text because instructions could be large
    - Cook_time: integer for minutes to make the meal
- Grocery:
    - Grocery_list_id: each list must have unique identifier

        - Recipe_id: points back to a specific recipe
        - Ingredients_to_buy: list of what needs to be bought, used text because it could be a long list
        - Date_made: date that the list was made
- list_Item (association table):
    - list _item_id: id for current recipes list item number
    - Recipe_id: what recipe is being added to list
    - Occasion_list_id: what occasion list the recipe is being added to
- Occasion_list:
    - Occasion_id: unique identifier for the occasion list
    - List_name: name of the list for the occasion
    - Type: ie casual, formal, etc
    - Feeds: how many people it feeds, int

Part 3:

```
CREATE TABLE users (
  user_Id SERIAL PRIMARY KEY,
  first_name VARCHAR(255),
  last_name VARCHAR(255),
  DOB DATE,
  age INT
),
CREATE TABLE auth (
  user_Id INT,
  username VARCHAR(255),
  email VARCHAR(255),
  password TEXT,
  FOREIGN KEY (user_Id) REFERENCES users(user_Id)
),
CREATE TABLE recipes (
  user_Id INT,
  recipe_Id SERIAL PRIMARY KEY,
  public BOOLEAN,
  pic_of_meal TEXT,
  ingredients TEXT,
  instruction TEXT,
  cook_time INT,
  FOREIGN KEY (user_Id) REFERENCES users(user_Id)

),
CREATE TABLE grocery (
```

```
  recipe_Id INT,
  grocer_list_id SERIAL PRIMARY KEY,
  ingredients_to_buy TEXT,
  date_created TIMESTAMP,
  FOREIGN KEY (recipe_Id) REFERENCES recipes(recipe_id)
),
CREATE TABLE occasion_list(
        occasion_id SERIAL PRIMARY KEY,
  list_name VARCHAR(255),
  type VARCHAR(255),
  feeds INT
),
CREATE TABLE list_item (
  recipe_Id INT,
  list_item_id SERIAL PRIMARY KEY,
  occasion_id INT,
  FOREIGN KEY (recipe_Id) REFERENCES recipes(recipe_id),
  FOREIGN KEY (occasion_id) REFERENCES occasion_list(occasion_id)
)
```