BRAINSTORMING

* 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

TABLE IDEAS

* Users Table: user\_id, email, password
* Recipes Table: include steps, user\_id, is\_private, ingredients
* Ingredients Table: (middle/association) varchar(lots)
* Occasions Table: (middle/association) reference recipes
* Grocery List Table: ingredients, user\_id, grocery\_list\_id

RELATIONSHIPS

* One to one: user to grocery list (user can only have one grocery list)
* One to many: users & recipes (users can have multiple recipes)
* Many to many: recipes & ingredients (recipes can have multiple ingredients and vice versa)

COLUMNS

* User
  + User\_id- INTEGER SERIAL PRIMARY KEY each user needs a unique id to identify them by
  + First\_name- VARCHAR(55) to input user name and limit their response length
  + Last\_name- VARCHAR(55) to input user name and limit their response length
  + Email- VARCHAR(255) to give people space for long emails
  + Password\_hash- TEXT to give as much space for long hash as needed
* Recipes
  + User\_id- INTEGER this references which users have this recipe
  + Steps- VARCHAR(1000) to give room to explain the steps of the recipe
  + Is\_private- BOOLEAN to give the user to keep their family recipes secret
  + Recipe\_ingredients- INTEGER to reference the Ingredients table
  + Recipe\_id- INTEGER SERIAL PRIMARY KEY unique id for every recipe in the table
* Grocery list
  + Grocery\_list\_id- INTEGER SERIAL PRIMARY KEY unique id for every list on the table
  + User\_id- INTEGER references the Users table to assign the list to a specific user
  + Ingredients\_id- INTEGER references the Ingredients table
* Occasions
  + Occasion\_id- INTEGER SERIAL PRIMARY KEY unique id for the occasion
  + User\_id- INTEGER references unique id of the user
  + Recipe\_id- INTEGER references the unique recipe id

-- CREATE TABLE users (

-- user\_id SERIAL PRIMARY KEY,

-- first\_name VARCHAR(50) NOT NULL,

-- last\_name VARCHAR(50) NOT NULL,

-- email VARCHAR(50) NOT NULL,

-- password\_hash TEXT NOT NULL

-- );

-- CREATE TABLE ingredients (

-- ingredient\_id SERIAL PRIMARY KEY,

-- ingredient\_name VARCHAR(255) NOT NULL

-- );

-- CREATE TABLE grocery\_lists (

-- grocery\_list\_id SERIAL PRIMARY KEY,

-- grocery\_list\_name VARCHAR(255),

-- user\_id INT NOT NULL REFERENCES users(user\_id),

-- ingredient\_id INT NOT NULL REFERENCES ingredients(ingredient\_id)

-- );

-- CREATE TABLE recipes (

-- recipe\_id SERIAL PRIMARY KEY,

-- step VARCHAR(8000) NOT NULL,

-- is\_private BOOLEAN NOT NULL,

-- recipe\_ingredient INT NOT NULL REFERENCES ingredients(ingredient\_id)

-- );

-- CREATE TABLE occasions (

-- occasion\_id SERIAL PRIMARY KEY,

-- user\_id INT NOT NULL REFERENCES users(user\_id),

-- recipe\_id INT NOT NULL REFERENCES recipes(recipe\_id)

-- );

-- INSERT INTO users(first\_name, last\_name, email, password\_hash)

-- VALUES ('Anthony', 'Dominguez', 'test@test.com', 'asdlfj;');

-- INSERT INTO ingredients(ingredient\_name)

-- VALUES ('cheese');

-- INSERT INTO ingredients(ingredient\_name)

-- VALUES ('dough'), ('pepperoni'), ('tomato sauce');

INSERT INTO grocery\_lists (grocery\_list\_name, user\_id, ingredient\_id)

VALUES ('test', 1, 1),

('test', 1, 2),

('test', 1, 3)

;

SELECT \* FROM grocery\_lists;