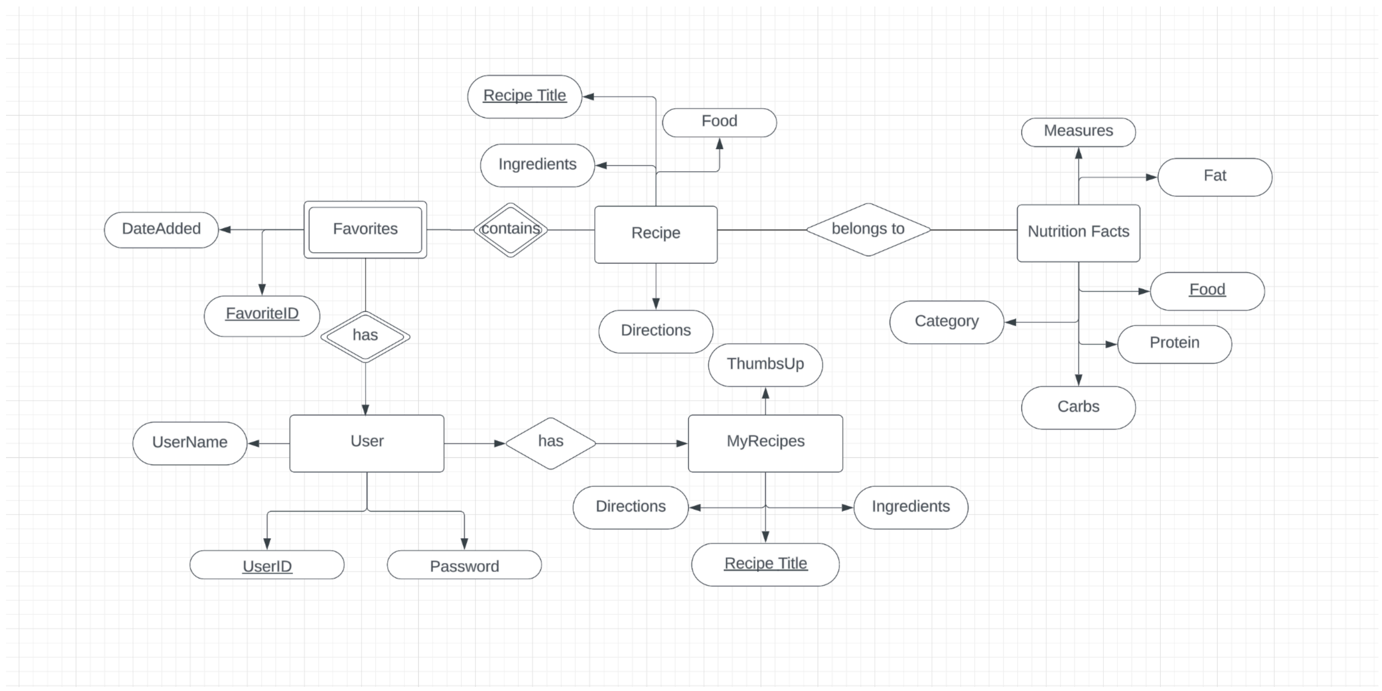

Description and Assumptions of Relations

- **User**
    - **Description:** The User table stores the information of users including UserID, UserName, and Password. UserID is the primary key of this table and always be unique.
    - **Assumption:** We assume the following:
        - Every User can have 0 to any number of Favorites and one Favorites can only be associated with one User.
        - We assume that every User can have 0 to any number of MyRecipes and one MyRecipes can only be associated with one User.
- **Favorites**
    - **Description:** Favorites table stores users' favorite recipes. Each record consists of FavoriteID, DateAdded is the primary key.
    - **Assumption:** We assume the following:
        - We assume that favorites can have 0 to many favorite recipes, and one Favorites can only be associated with one User.
        - We assume that each favorite set can contain many recipes, and each recipes can be added to many favorite sets.

- **Recipe**
    - **Description:** Recipe table has information about each Recipe in the database such as recipe title (the unique identifier), Food, Ingredients, and Directions. We use recipe titles as our primary key, because it will always be unique.

  - **Assumption:** We assume that each recipe belongs to many Nutrition Facts, and many Nutrition Facts belong to many Recipes, since they both relate to one another. The relationship between favorite and recipe is many to many.
- **Nutrition Facts**
  - **Description:** Nutritional Facts table stores the nutritional information about the Food, which is a primary key for the table. The table contains Measures, Fat, Category, Protein, Carbs, and Food.
  - **Assumption:** One nutritional fact can belong to 1 to many Recipes, and each recipe can contain 1 to many nutritional facts.
- **MyRecipes**
  - **Description:** MyRecipes contains the recipes input by User. It contains RecipeTitle, Ingredients, ThumbsUp, and Direction. The primary key is RecipteTitle.
  - **Assumption:** One MyRecipes entity belongs to one user, and each user can have 0 to many recipes within MyRecipe.

BCNF/3NF:

User: <u>UserID</u>-> UserName, Password
*Favorites: <u>FavoriteID</u> -> DateAdded, <u>RecipeTitle</u>, <u>UserID</u>
Recipe: <u>RecipeTitle</u> -> Ingredients, <u>Food</u>, Directions
NutritionFacts: <u>Food</u> -> Protein, Carbs, Measure, Fat, Category
*MyRecipe: <u>RecipeTitle</u> -> <u>UserID</u>, Ingredients, Directions, ThumbsUp

We choose to use 3NF:

1) For the User Table, it is already normalized, since the functional dependency on the left (UserID) already explains the rest of the components
2) We designed the Favorites, MyRecipe tables such that they are already normalized, as they have primary keys on the left (FavoriteID and RecipeTitle) that explain all the candidate keys on the right hand side.
3) The Recipe Table is already normalized, as it

Relational Schema:

User(
        UserID: CHAR(6) [PK],
        UserName: VARCHAR(100),
        Password: VARCHAR(20)
)

Favorites(
        FavoriteID: CHAR(6) [PK],
        DateAdded: DATE,
        RecipeTitle: VARCHAR(255) [PK] [FK to Recipe.RecipeTitle],

```
        UserID: CHAR(6) [PK] [FK to User.UserID]
)

Recipe(
        RecipeTitle: VARCHAR(225) [PK] [FK to Recipe.RecipeTitle]
        Ingredients: JSON_ARRAY(100),
        Food: JSON_ARRAY(100) [PK],
        Directions: JSON_ARRAY(100)
)

NutritionFacts(
        Food: VARCHAR(225) [PK] [FK to Recipe.Food],
        Measures: VARCHAR(225),
        Fat: VARCHAR(225),
        Protein: VARCHAR(225),
        Carbs: VARCHAR(225),
        Category: VARCHAR(225)
)

MyRecipes(
        RecipeTitle: VARCHAR(225) [PK],
        UserID: CHAR(6)  [PK] [FK to User.UserID],
        Ingredients: JSON_ARRAY(100),
        Directions: JSON_ARRAY(100),
        ThumbsUp: INT
)
```