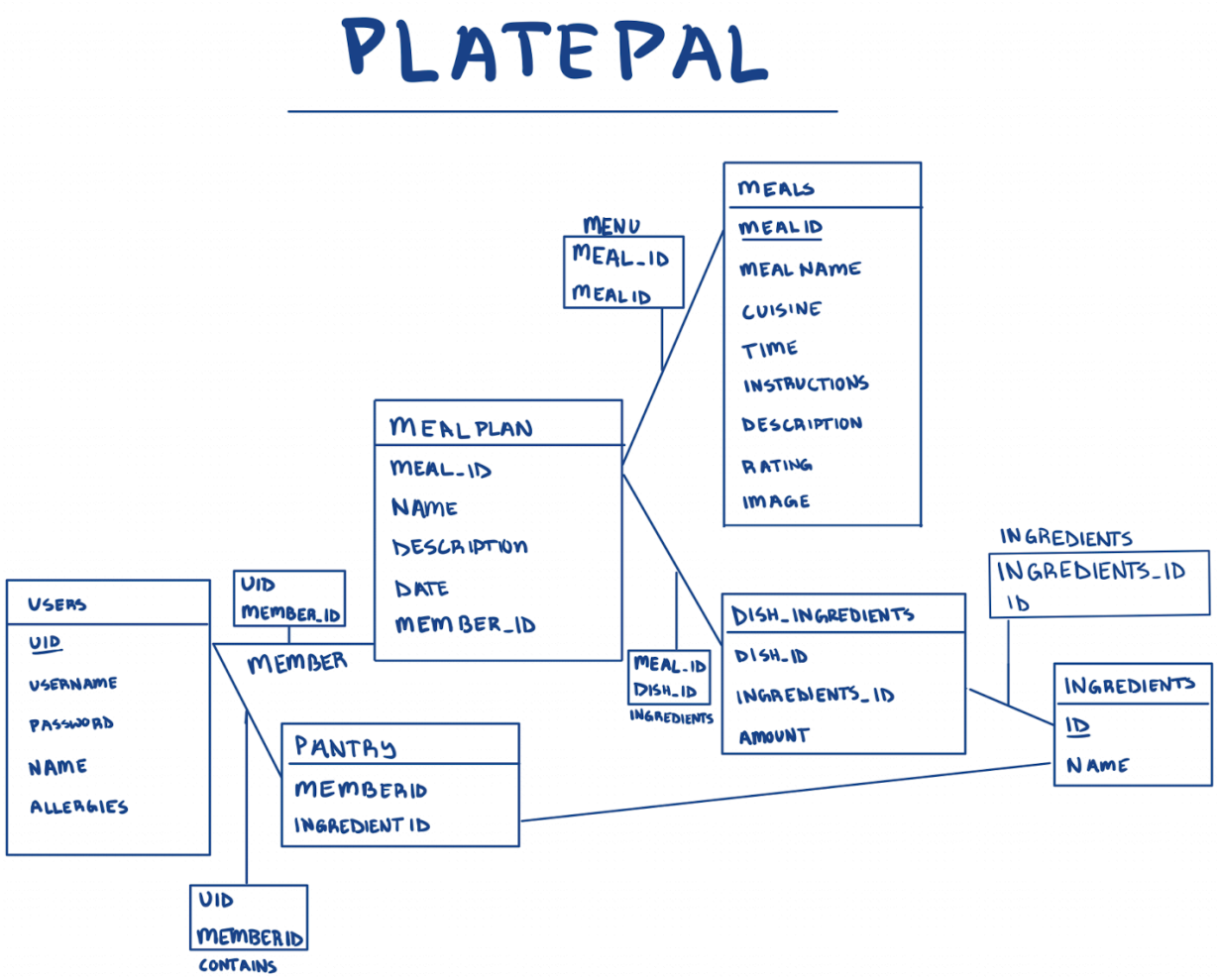
UML Diagram:



Our Assumptions:

Users Table:

- This entity represents users who have access to the system.
- Assumptions:
    - Each user has a unique UID (User ID) to identify them in the system.
    - Users have basic account information such as Username, Password, PasswordHash, and Name.
    - Allergies are stored as a list of potential allergens for each user.

Pantry Table:

- This entity represents the ingredients available in each user's pantry.

- Assumptions:
    - Each entry in the pantry is associated with a specific member (user) identified by MemberId.
    - IngredientId refers to the unique identifier of each ingredient.

Meal Plan Table:

- This entity represents the planned meals for each user.
- Assumptions:
    - MealId is a unique identifier for each meal plan entry.
    - Time, Cuisine, Instructions, Description, and Date/Time Planned provide details about the planned meal.
    - MemberId serves as a foreign key referencing the Users Table, linking each meal plan entry to a specific user.

Meals Table:

- This entity represents individual meals available in the system.
- Assumptions:
    - MealId is a unique identifier for each meal.
    - Cuisine, Time, Instructions, Description, Rating, and Image provide details about each meal.
    - Meals are independent entities that users can choose from when planning their meals.

Dish Ingredients Table:

- This entity represents the ingredients required for each dish (meal).
- Assumptions:
    - Dish_Id refers to the unique identifier of each dish (meal).
    - Ingredients_Id refers to the unique identifier of each ingredient.
    - Amount specifies the quantity of each ingredient required for the dish.

User Ingredients Table:

- This entity represents additional ingredients that users may have beyond those listed in their pantry.
- Assumptions:
    - Id is a unique identifier for each user ingredient.
    - Names store the names of ingredients that users possess.

User to Pantry: The relationship is one-to-one.
User to Meal Plan: Each user can have multiple meal plans, so the relationship is one-to-many.
Meal Plan to Meals: Each meal plan consists of multiple meals, so the relationship is one-to-many.
Meals to Dish Ingredients: Each meal consists of multiple ingredients, so the relationship is one-to-many.
User to User Ingredients: Each user can have multiple additional ingredients, so the relationship is one-to-many.

3NF Normalization:

USERS

The USERS table seems to be in 3NF since all attributes are directly dependent on the primary key

MEMBER

The MEMBER table is in 3NF as well since MEMBER_ID is the primary key and UID is a foreign key that references USERS.

PANTRY

MEMBER_ID (Composite Key with INGREDIENT_ID)

INGREDIENT_ID (Composite Key with MEMBER_ID)

The PANTRY table is in 3NF. It uses a composite key and avoids any transitive dependencies.

INGREDIENTS

The INGREDIENTS table is in 3NF. The NAME of the ingredient is dependent on the primary key INGREDIENT_ID.

MEALS

The MEALS table is in 3NF if all attributes are functionally dependent only on the primary key and there are no dependencies between non-key attributes.

DISH_INGREDIENTS

The DISH_INGREDIENTS table is in 3NF since it only has foreign keys and the AMOUNT attribute, which is dependent on the composite primary key.

MENU

The MENU table has just one foreign key which is the primary key, thus it is in 3NF.

Meal Plan

The MEAL_PLAN table is also in 3NF since the NAME and DESCRIPTION are only dependent on the MEAL_ID

In this schema, since there are no transitive dependencies attributes depend only on their respective primary keys, which is why the schema is in 3NF. We chose to use 3NF over BCNF because our database does not have any complex relationships or dependencies that would necessitate BCNF's stricter constraints

Relational Schema:

Users(UID:INT [PK], Username:VARCHAR(20), Password:VARCHAR(20), Name:VARCHAR(20), Allergies:VARCHAR(20))

Pantry(Member_ID:INT [FK to Users.UID], Ingredient_ID:INT [FK to Ingredients.ID])

Meal_Plan(Meal_ID:INT [FK to Meals.Meal_ID], Member_ID:INT [FK to Users.UID], Name:VARCHAR(20), Description:VARCHAR, Date:DATE)

Meals(Meal_ID:INT [PK], Meal_Name:VARCHAR(20), Cuisine:VARCHAR(20), Time:TIME, Instructions:VARCHAR, Description:VARCHAR, Rating:REAL)

Dish_Ingredients(Dish_ID:INT [FK to Meals.Meal_ID], Ingredient_ID:INT [FK to Ingredients.ID], Amount:VARCHAR(20))

Ingredients(ID:INT [PK], Name:VARCHAR(20))