PROJECT DOCUMENTATION OF PLAN & PLATE

By Harshil Vallabh (2656158)
Yabsira Gebremichael (2661262)
Kamal Lalloo (2652159)
Karan Jeevan (2662579)

Procedures and processes

Throughout the app development process, creation of PLAN & PLATE, there were many goals and objectives to be met. Providing the user with an application that allows them to discover and try new recipes, like and comment on those recipes (where users can leave a rating out of 10 in the comments), plan meals, prepare grocery lists, post recipes of their own to share with the rest of the world and allow users to create a community of friends.

Given the nature of this application, there are many key processes and procedures involved that enable our mobile app to achieve the functionalities mentioned earlier. There is a flow of processes starting from user sign in and registration, if the individual is already a registered user of the app, they simply need to sign in, the username and password provided will be checked against those in the "User'' table of the database, if it matches one row/record they will be signed in. However, if the individual is a first-time user of the application, they will first need to register with all the necessary details. They need to add their name, username, password and email. There will be data validation for the users details to ensure they are all acceptable such as a unique username and valid email, thereafter, they are registered and added to the database (entered into the ''User'' table).

The user will then be greeted by the home page of the app. On this page the user will be able to view all the recipes/meals posted by other users of PLAN & PLATE, by scrolling down, as well as search for recipes by pressing the search icon. All these recipes are retrieved from the "Recipe" table in the database. Additionally, when pressing on a recipe the user will be able to see who posted that recipe, the ingredients needed to make the recipe and the instructions, this information for the specific recipe is also retrieved from the "Recipe" table. There is a button which allows users to add that specific recipe to their meal plan which saves the recipe as a meal that they would like to prepare for themselves, from this we will be able to display a user's daily meal plans and the ingredients that they need to purchase for the next 7 days. The saved meals are placed in a table that stores the chosen meals, from this table and the "Recipe" table the grocery list can be determined. The user will also be able to comment on the recipe by sliding to the comment section and they can rate the recipe out of ten in the comments (1 being terrible and 10 being perfection). They can also like the recipe by pressing the like button, all the comments, ratings and likes are added to and retrieved from the "Comment" and "Likes" tables, respectively.

Pressing the icon next to the home button takes us to the meal adder page. The user will be able to post any recipe of their choice from this page, they simply need to enter the name of the recipe, provide a short description of the recipe, choose the ingredients needed to make the recipe as well as the quantity of each ingredient, provide the time it takes to cook

and provide the instructions that need to be followed. They then confirm their instructions and finally confirm the recipe. After this, the recipe is successfully posted. The recipe that the user has posted is added to and retrieved from the “Recipe” table. The next icon is the grocery icon, that when clicked on will display your daily meal plan as well as your grocery list for the next 7 days, these were determined by all the recipes that the user adds to their meal plan.

Finally, the user can also click the community icon, which will take them to the communities page. On this page the user can slide between the “Feed”, “Add Friends” and “Requests” options. On the “Add Friends” option the user will be met by a list of all the users of the app and has the option to add another user as a friend, they can also search for specific users, once they add a user as a friend, a request is sent to that other user. You yourself can receive friend requests from other users if they add you as a friend, slide to the “Requests” option, this will show you all your friend requests, if you have any; you can accept or decline a request by choosing either the “Accept” or “Decline” button. Lastly, we have the “Feed” option, which lists all the posts that have been made by your friends. If two users become friends in the app, they are added to the “Friends” table. When a user is finished with their app experience, they can simply log out, this is done by going to the home page, clicking the top left hand corner bar and choosing the log out option.

Business rules

A single user can have many different PLAN & PLATE accounts, each account must have a unique username that does not already exist, a valid name, a valid password and a valid email. A single account belongs to a single user.

A user can like as many different recipes as they want but they can only like each recipe once (a user can't have multiple different likes for the same recipe simultaneously). Each like made by a specific user on a specific recipe belongs only to that user. A single recipe can be liked by many different users. Each like made by a specific user on a specific recipe belongs only to that recipe.

A user can comment (can include rating in comment) on as many different recipes as they desire and they can make one or more comments on the same recipe. Each individual comment belongs to a single user and single recipe. A single recipe can have many different comments (can be commented on by many different users).

A user can add a meal to their meal plan and choose a date on which to make it, although they can add many different meals for the same day and meal type (lunch for the 2nd of June could be meatloaf, noodles and lasagna), they cannot choose the same recipe for the same day and meal type again (you can't have meatloaf on the 2nd of June 2024, if you already chose meatloaf for the 2nd of June 2024). A user can have many recipes in their meal plan, a single recipe in a users meal plan belongs only to that user, in other words, a specific recipe chosen by a specific user is known as a saved/selected recipe and it belongs only to that user. A single posted recipe can be saved by many users, a saved recipe can be posted once and by a single user.

A user can post as many recipes as they want, each recipe must have a name, description, cooking time, required ingredients (and their quantities) and instructions. Each posted recipe belongs to a single user.

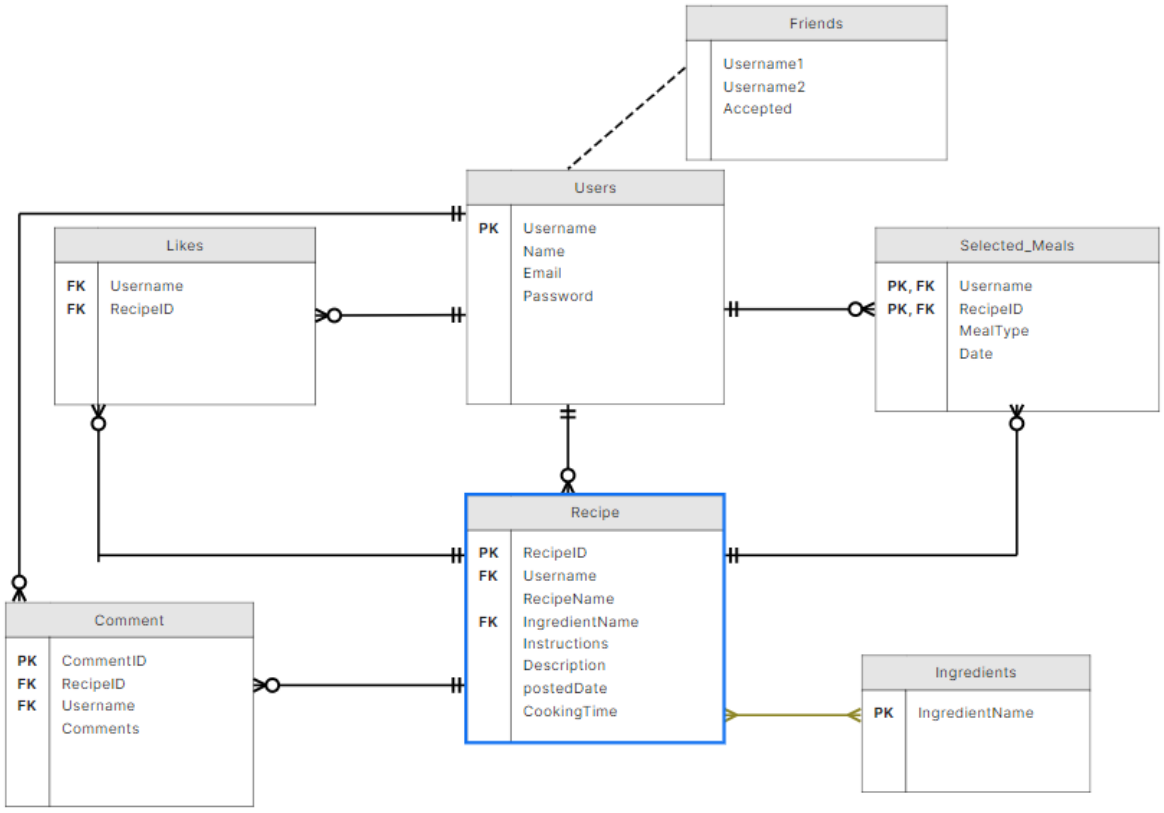
All recipes shown on the home page can't be older than 4 weeks (Recipes shown on the home page are at most 4 weeks old).

A single user can be friends with many other users, a friend of the user can be friends with many other users as well and users can only be friends with each other if the one user accepts the other users friend request.

The user can only pick from the set amount of ingredients that are offered by the app, i.e. the recipe they choose to post can't include ingredients that isn't in the predefined ingredient list. A single recipe can be made up of many different ingredients and a single

ingredient could be found in many recipes. (This creates a many to many relationship which is dealt with later)

Initial ERD:


Friends
Username1
Username2
Accepted
Users
PK
Username
Name
Email
Password
Likes
FK
Username
FK
RecipeID
Selected_Meals
PK, FK
Username
PK, FK
RecipeID
MealType
Date
Recipe
PK
RecipeID
FK
Username
RecipeName
FK
IngredientName
Instructions
Description
postedDate
CookingTime
Comment
PK
CommentID
FK
RecipeID
FK
Username
Comments
Ingredients
PK
IngredientName

Issues facing initial ERD

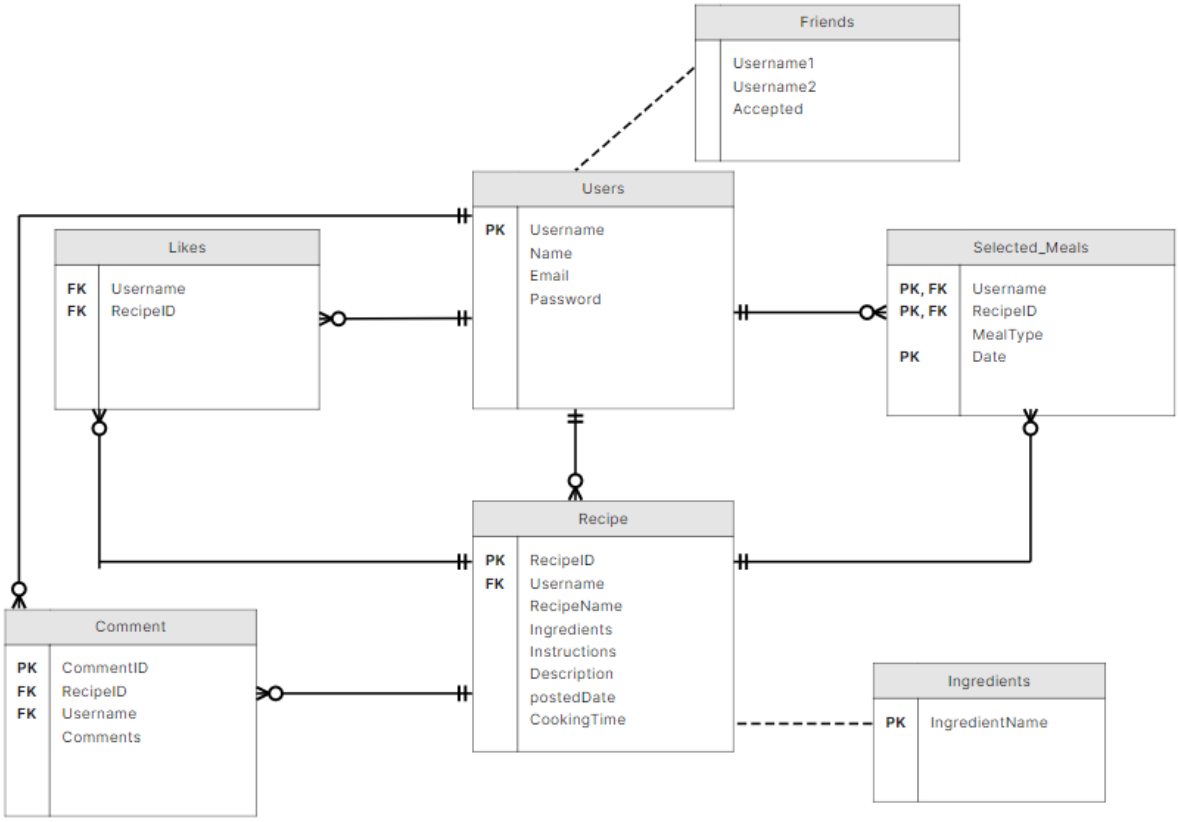
1) Many to many relationship: The many to many relationship that occurs in our initial ERD is that between the ‘’Recipe’’ table and the ‘’Ingredients’’ table. The “IngredientName” attribute is a foreign key in the “Recipe” table which shows the explicit relationship between the two tables. A single recipe can have many different ingredients and a single ingredient could be in many different recipes.

2) Data integrity: In the making of the initial ERD there are many different fields where inaccurate and inconsistent data could lead to many problems. At the time of the development of the initial ERD there was no validation code written for different fields/attributes like the “Email”, “Name” and “Username”. This allowed for redundant and invalid entries into the database.

3) Potential null values: Similarly to the data integrity issues; the lack of validation code at the initial development of the ERD made it possible for users to enter null values into many different attributes inside the database within different tables. Although primary key attributes can’t have null values in them, all other attributes could have null values in them. Therefore, without the necessary validation, attributes such as “RecipeName” and “Email” could have null values in them.

4) Redundancy: The initial development of the ERD made it so that when a user chose to add a recipe to their meal plan, that recipe would be saved in the database. A user could choose different recipes for the same day and meal type (lunch for the 2nd of June 2024 could be meatloaf, noodles and lasagna). However, it did not account for the fact that a user could choose the exact same recipe for the same day and meal type, numerous times, which led to duplication and redundancy.

Final ERD


Friends
Username1
Username2
Accepted
Users
PK
Username
Name
Email
Password
Likes
FK
Username
FK
RecipeID
Selected_Meals
PK, FK
Username
PK, FK
RecipeID
MealType
PK
Date
Recipe
PK
RecipeID
FK
Username
RecipeName
Ingredients
Instructions
Description
postedDate
CookingTime
Comment
PK
CommentID
FK
RecipeID
FK
Username
Comments
Ingredients
PK
IngredientName

Solutions

1) M:N relationship between recipe table and Ingredients table: The foreign key constraint “IngredientName” inside the recipe table, is the primary key of the ingredients table, hence creating a referential link between these two entities. However, this link made it so that there was a many to many relationship. The solution for this problem was that we removed the foreign key constraint from the recipe table, effectively eliminating the direct relationship between the two tables. Thus the “Ingredients” has no explicit relationship with any other table in the database. The ingredients table stores a fixed amount of data, so we never have to worry about adding data to this table, only retrieving from it. Hence it is effectively a stand-alone table, however, this is only partially true, we end up replacing the “IngredientName” attribute with an “Ingredients” attribute in the recipe table. This attribute is not a referential constraint on the recipe table but rather an attribute that holds a list of the ingredients needed to make the recipe. Our code is written in such a way that we receive the required ingredients from the ingredient table and add them to the ingredient list in the recipe table without requiring a direct connection between the two. However, the fact that the ingredient list is made up of data from the ingredients table means that although there is no explicit (referential) connection between the two tables, there is still logical dependance between the two, represented by the dashed line. This is a similar situation for the friends table as well. Furthermore, regarding the ingredient list, each ingredient selected by the user will be separated by a comma, this final list of ingredients will be added to the database in the form ingredient1, ingredient2,… When we retrieve these ingredient lists from our database, we can separate the individual ingredients into their own variables using a Scanner with a delimiter for a comma thus separating the ingredient list into its individual components to use as necessary.

2) The data integrity problem: In the initial creation of the ERD, we did not have any validation code to validate user input. This made it so that in every attribute the user could add inaccurate and inconsistent data. So the necessary validation code was executed to ensure that our table would not hold invalid and inaccurate data.

For example, validation code that prevented users from using the same username as someone else, hence ensuring that usernames were unique, it also prevented entry of invalid email, invalid names etc..

3) Potential null value. This was solved using the data validation code that makes sure that there's no null values within the database. The primary keys were not vulnerable to these null values as they were required to uniquely identify each record however other attributes were vulnerable to null values. Hence, validation code was put into place that did not allow users to proceed with a certain action without first filling in all the necessary fields such as the registration page.

4) Redundancy issue was solved in a similar way to the other problems by adding validation code that did not allow the user to select the same meal for the same day and meal type. Furthermore, we made it so that the RecipeID, Username and Date were all primary keys in the Selected_Meals table, by using all of them as a composite key for that table, we were able to ensure that you could only select the same meal once a day. This was for the consumers health and eating habits as eating the same meal twice or more in one day could be unhealthy and we discourage it. We hope to encourage healthy eating patterns for our users.

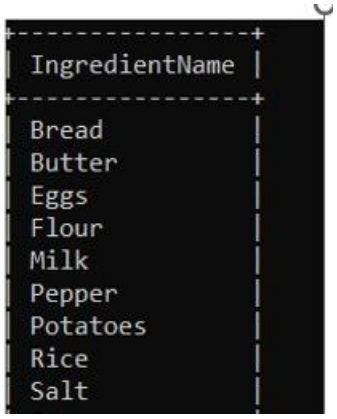
Implementation of tables/functions/procedures/views/triggers

The tables are created in correspondence with the respective pages that the app has.

Tables:

The tables are put together as shown in the ERD above.

Ingredients table



The Ingredients table has data that is set and cannot be changed and it has been used in the Add Recipe page in the spinner. The function that was used in the Select Command because the table has only one purpose and that is to display the ingredients. There are no procedures, views, or trigger with this table. It has no relation to any other table in the database.

Users table

Name	Username	Password	Email
Blop	BlopTheMan123	BlopTheMan123	blop@gamilc.om
harper	harper79	H1ter12trsv	harper@gmail.com
Sweet	joe123	Thejoe123	joe123@gmail.com
	K		
Kamal	Lalloo	Kamallalloo6305	kamal.lalloo10@gmail.com
yay	moooo98	zcd	zdcv
Yabi Pog	peter04	Cheese123	yab.gab@gmail.com
yab	t	kd	q
NAME	test	test	EMAIL
Yabi	uno785	G983903hty	fan@gmail.com
Harshil	uno791	Poggers123	vallabh.harshil@gmail.com

The users table is primarily used in the Sign in and Sign up page. It is used to insert new users into database by their name, Username, Email and a password that is a part of that account. The table is then used in the sign in page to validate if the details entered is an existing user. This table also ensures that no user has the username thus making the Username attribute ,the primary key of this table. This key is then used as a foreign key in the friends table below. The Username is also used in the selected_meals table as it is use to contain the meals that the user saved as well as the Comments table as the username is used here to recognize which user commented on a certain recipe.

Friends tables.

Username1	Username2	Accepted
uno791	Big_M595	0
test	harper79	0
test	moooo98	0
test	Lalloo	1
test	t	0
uno791	K	0
uno791		1
test	BlopTheMan123	0
test	uno791	1
uno791	BlopTheMan123	0
uno791	harper79	0
uno791	joe123	0
test	uno785	0
uno791	Lalloo	1
test	Uno7915	0
uno791	Uno7915g	0

Friends table contains the link between two users, there for two usernames from the users table is used in the Friends table to show the link. This table is used mainly in in the

community tab where the users choose to request to be friends and accept to be friends with another user. The table uses an insertion function to show the link. The deletion function to get rid of the link and an update function to update whether user 1 as accepted user 2s follow request.

Comment table

CommentID	RecipeID	Username	Comments
1	5	test	This sucks
2	5	test	Try better bozo
3	5	test	grrrr
4	10		delicious
5	6	BlopTheMan123	what are you cooking bro
6	9	Lalloo	what
7	11	test	
8	2	test	
9	8	test	
10	11	Lalloo	wrong recipe
11	13	test	WHYYYY
12	1	test	what is this
13	6		
14	6	test	this was delicious

The comment table stores all the comments that a certain user made on a certain recipe. The table uses the RecipeID from the recipe table and the username. The table has a 1 to many relationship with both the recipe table and the users table. The comment table has an insert command to insert each comment into the table with reference to which user and which recipe the comment belongs to.

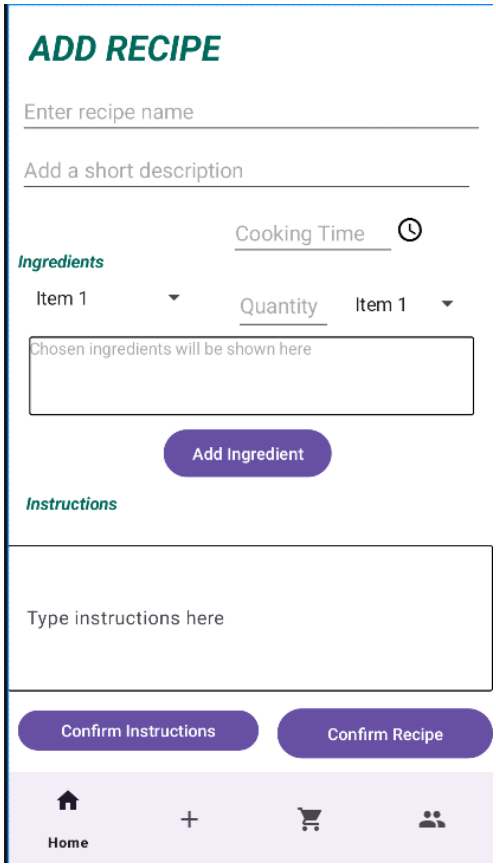
Selected Recipes table.

Username	RecipeID	Date	MealType
test	1	2024-05-30	Dinner
test	5	2024-05-29	Dinner
test	6	2024-05-28	Dinner
test	8	2024-05-28	Lunch
test	9	2024-05-31	Breakfast
test	10	2024-05-31	Dinner

The selected recipes table is used to store which recipes a certain user has saved. It uses the username from the users table and the the recipe ID from the recipes table. The date and Meal type column is chosen directly in the app page and thereby the row is inserted into the table with eh corresponding username and recipe ID.

The recipes table

The recipe table is the most relations to other tables within the database. This table contains the username from the users table and mentioned before Recipe ID is used as a foreign key in the multiple other tables such as the selected meals table and the comments table. The table uses an insertion function also that is used by the Add Recipe page. The recipes within this table are also displayed constantly on the feed page therefore a selection command is used for this.



Normalization

Normalization means to organize data within a database. Through normalization we reduced data redundancy and improved data integrity. Normalization in the case of PLAN & PLATE was used in the very early stages of designing the database. The final and initial ERD both show this, Before the initial ERD was made there were different ERD's that were conceptualized, discussed and written down. Initially the database had fewer tables and each table had more attributes. The tables were then separated, for example the users and their friends which were initially one table became two. The selected meals table that is used to show what meals the user had selected for themselves contains two foreign keys one of which is the primary of the users table while the other is a primary key of the recipe table. The database is designed in a way to reduce having to add multiple values that are the same within one table in the database. The data thus after normalization is more organized, less redundant and more accurate. To further increase data integrity, the validation code was used throughout the application. Normalization however took out most of the issues related to the data stored in the database of PLAN & PLATE.