# CSC309 Group 431

The APIs are categorized by the apps in Django:

- accounts
- recipes
- userdata
- userfunction

---

## accounts

Accounts APIs are responsible for login, signup, update user profiles.

---

### POST Sign Up

```
http://127.0.0.1:8000/accounts/signup/
```

Sign Up requires the user to enter a unique username and email with passwords of at least 8 characters.

The response will return a JSON object with filed username and email.

Note:

- Password will not be returned.
- When signing up, users cannot add their phone and avatar.

Example Payload:

Plain Text

```
payload={'username': 'test',
'email': 'test@gmail.com',
'password': 'qwertyuiop32392',
'password2': 'qwertyuiop32392'}
```

**Body** formdata

| | |
|---|---|
| **username** | test |
| | Username must be unqiue |

| | |
|---|---|
| **email** | test@gmail.com<br>Email must be unique |
| **password** | qwertyuiop32392<br>Password must be at least 8 characters |
| **password2** | qwertyuiop32392<br>Password must be at least 8 characters |

## POST Log In

```
http://127.0.0.1:8000/accounts/login/
```

The response will return user's information except for password.

Example Payload:

```
Plain Text

payload={'username': 'test',
'password': 'qwertyuiop32392'}
```

### Body formdata

| | |
|---|---|
| **username** | test<br>This field is required |
| **password** | qwertyuiop32392<br>This field is required |

## POST Log Out

```
http://127.0.0.1:8000/accounts/logout/
```

Log Out does not take any user inputs but it will ask the user to be logged in first in order to log out.

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## PUT Edit Profile

```
http://127.0.0.1:8000/accounts/profile/edit/
```

Edit Profile allows users to add phone numbers and avatars.

Note: When signing up, users cannot add their phone and avatar.

Example Payload:

Plain Text

```
payload={'username': 'test',
'email': 'test@gmail.com',
'password': 'ehwrqjklhejklqw',
'password2': 'ehwrqjklhejklqw',
'phone': '12345678'}
files=[
  ('avatar',('01_A_1.jpg',open('/Users/jingwenshi/Downloads/asl_data/test/A/01_A_1.jpg','rb'),'
]
```

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

### Body formdata

| | |
|---|---|
| **username** | test<br>Username must be unique |
| **email** | test@gmail.com<br>Email must be unique |
| **password** | ehwrqjklhejklqw<br>Password must be at least 8 characters long |
| **password2** | ehwrqjklhejklqw<br>Password must be at least 8 characters long |
| **phone** | 12345678<br>This field is not required |
| **avatar** | This field is not required |

## GET Get Profile

```
http://127.0.0.1:8000/accounts/profile/details/
```

Returns a JSON response with all fileds of the current user (except for password).

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## recipes

recipe APIs contains all recipe related methods:

- Create, Update, Delete Recipe/RecipeIngredient/Step
- Create Ingredient
- Get Recipe

Note:

- Ingredient and RecipeIngredient are two different models. Ingredient is a model that contains unique ingredient names. And Recipe Ingredient is a model with 2 FK pointed to Recipe and Ingredient and 2 attributes of amount and unit.
- Step is a model contains a description of the current step and a FK pointed the Recipe model.
- For more details, please refer to the Model descriptions.

## POST Create Recipe

```
http://127.0.0.1:8000/recipes/create-recipe/
```

Example Payload:

Plain Text

```
payload={'title': 'How to make Soup Bao',
'description': 'A delicious dish!',
'time': '2',
'time_unit': 'hours',
'cuisine': 'Chinese',
'diet': 'Others'}
files=[
  ('picture',('01_A_1.jpg',open('/Users/jingwenshi/Downloads/asl_data/test/A/01_A_1.jpg','rb'),
]
```

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

### Body formdata

| | |
|---|---|
| **title** | How to make Soup Bao<br>This field is required |
| **description** | A delicious dish!<br>This field is required |
| **picture** | This field is required |
| **time** | 2<br>This field is required |
| **time_unit** | hours<br>This field is required |
| **cuisine** | Chinese<br>This field is required |
| **diet** | Others<br>This field is required |

## PUT Update Recipe

```
http://127.0.0.1:8000/recipes/recipe=1/RUD-recipe/
```

A recipe id must be passed to the endpoint url after the equal sign.

The payload is the same as Create Recipe

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

### Body formdata

| | |
|---|---|
| **title** | How to make Soup Dumplings<br>This field is required |

| | |
|---|---|
| **description** | This is a traditional chinese dish!<br>This field is required |
| **picture** | This field is required |
| **time** | 3<br>This field is required |
| **time_unit** | hours<br>This field is required |
| **cuisine** | Chinese<br>This field is required |
| **diet** | Others<br>This field is required |

## DELETE Delete Recipe

```
http://127.0.0.1:8000/recipes/recipe=1/RUD-recipe/
```

Delete a specific recipe from the current user.

A recipe id must be passed to the endpoint url after the equal sign.

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## POST Create Ingredient

```
http://127.0.0.1:8000/recipes/recipe=1/create-ingredient/
```

Note: Once an ingredient is created, it cannot be deleted. But a RecipeIngredient can be deleted from a user's recipe.

Example Payload:

Plain Text

```
payload={'name': 'Pork'}
```

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

**Body** formdata

| | |
|---|---|
| **name** | Pork<br>This field is required |

## POST Create Recipe Ingredient

```
http://127.0.0.1:8000/recipes/recipe=1&ingredient=1/create-recipe-ingredient/
```

A recipe id and ingredient id must be passed to the endpoint url after the equeal sign.

Note: The name of the ingredient should be passed in as an id in the endpoint. For more details, please refer to the model descriptions.

Example Payload:

Plain Text

```
payload={'amount': '500',
'unit': 'grams'}
```

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

**Body** formdata

| | |
|---|---|
| **amount** | 500<br>This field is required |
| **unit** | grams<br>This field is required |

## PUT Update Recipe Ingredient

```
http://127.0.0.1:8000/recipes/recipe=1&ingredient=1/RUD-recipe-ingredient/
```

A recipe id and ingredient id must be passed to the endpoint url after the equeal sign.

The payload is the same as Create Recipe Ingredient.

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

### Body formdata

| | |
|---|---|
| **amount** | 400<br>This field is required |
| **unit** | grams<br>This field is required |

## DELETE Delete Recipe Ingredient

```
http://127.0.0.1:8000/recipes/recipe=1&ingredient=1/RUD-recipe-ingredient/
```

A recipe id and ingredient id must be passed to the endpoint url after the equeal sign.

Delete a RecipeIngrediet from the current user's specific recipe.

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## POST Create Step

```
http://127.0.0.1:8000/recipes/recipe=1/create-step/
```

A recipe id must be passed to the endpoint url after the equeal sign.

Note:

The number represents the step number and it is not unique. That is, a recipe can have multiple step 2. In the front end, the developer only simply displays the step in non-decreasing order and is the user's responsibility to organize their steps.

Example Payload:

Plain Text

```
payload={'number': '1',
'description': 'Mix pork with soy sauce'}
files=[
  ('picture',('02_A_1.jpg',open('/Users/jingwenshi/Downloads/asl_data/test/A/02_A_1.jpg','rb'),
]
```

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

### Body formdata

| | |
|---|---|
| **number** | 1<br>This field is required |
| **description** | Mix pork with soy sauce<br>This field is required |
| **picture** | This field is required |

## PUT Update Step

```
http://127.0.0.1:8000/recipes/recipe=1&step=1/RUD-step/
```

A recipe id and step id must be passed to the endpoint url after the equeal sign.

The payload is the same as Create Step.

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

### Body formdata

| | |
|---|---|
| **number** | 1 |
| **description** | Mix pork with salt for 3 mins |
| **picture** | |

## DELETE Delete Step

```
http://127.0.0.1:8000/recipes/recipe=1&step=1/RUD-step/
```

A recipe id and step id must be passed to the endpoint url after the equeal sign.

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## GET View/Get Recipe

```
http://127.0.0.1:8000/recipes/recipe=1/details/
```

Returns a JOSN response with all fileds of a recipe. In addition, it also returns the number of likes and a list of comments.

A recipe id must be passed to the endpoint url after the equeal sign.

## userdata

userdata APIs contains all method to manipulate and modify user's data.

Note:

For all the models in userdata (Rating, Comment, LikedRecipe, BrowsedRecipe, FavouritedRecipe, Shopping) all have two FK pointed to the current user and a specific recipe. In addition, Rating has a score attribute and Comment has a message attribute.

Hence, you may need to pass in the id of the above models' instances to the endpoint.

## POST Create Rating

```
http://127.0.0.1:8000/userdata/recipe=1/create-rating/
```

When current user rated a recipe, the API will be called.

A recipe id must be passed to the endpoint url after the equeal sign.

Example Payload:

```
Plain Text
```

```
payload={'score': '5'}
```

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

**Body** formdata

| | |
|---|---|
| **score** | 5<br>This field is required |

## PUT Edit Rating

```
http://127.0.0.1:8000/userdata/rating=1/RU-rating/
```

A recipe id must be passed to the endpoint url after the equeal sign.

The payload is the same as Create Rating.

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

**Body** formdata

| | |
|---|---|
| **score** | 4 |

## POST Create Comment

```
http://127.0.0.1:8000/userdata/recipe=1/create-comment/
```

When the current posted a comment to a recipe, this API will be called.

A recipe id must be passed to the endpoint url after the equeal sign.

Example Payload:

Plain Text

```
payload={'message': 'This recipe is pretty good!'}
```

```
files=[
  ('file',('01_A_1.jpg',open('/Users/jingwenshi/Downloads/asl_data/test/A/01_A_1.jpg','rb'),'im
]
```

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

**Body** formdata

| | |
|---|---|
| **message** | This recipe is pretty good!<br>This field is required |
| **file** | This field is required |

## PUT Edit Comment

```
http://127.0.0.1:8000/userdata/recipe=1&comment=1/RUD-comment/
```

A recipe id and comment id must be passed to the endpoint url after the equeal sign.

The payload is the same as Creat Comment.

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

**Body** formdata

| | |
|---|---|
| **message** | This recipe is good, but each step could be more clear.<br>This field is required |
| **file** | This field is required |

## DELETE Delete Comment

```
http://127.0.0.1:8000/userdata/recipe=1&comment=1/RUD-comment/
```

Delete a comment based on its comment id.

A recipe id and comment id must be passed to the endpoint url after the equeal sign.

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## POST Like Recipe

```
http://127.0.0.1:8000/userdata/recipe=1/create-liked-recipe/
```

Like Recipe does not have any payload. The endpoint will take the recipe id and the currently logged-in user to record this action to the model.

A recipe id must be passed to the endpoint url after the equeal sign.

Example Payload: None

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## DELETE Delete Like Recipe

```
http://127.0.0.1:8000/userdata/liked-recipe=1/RD-liked-recipe/
```

Undo the action of like a recipe.

A LikedRecipe id must be passed to the endpoint url after the equeal sign.

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## POST Create Browsed Recipe

```
http://127.0.0.1:8000/userdata/recipe=1/create-browsed-recipe/
```

Create Browsed Recipe does not have any payload. The endpoint will take the recipe id and the currently logged-in user to record this action to the model.

A recipe id must be passed to the endpoint url after the equeal sign.

Example Payload: None

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## DELETE Delete Browsed Recipe

```
http://127.0.0.1:8000/userdata/browsed-recipe=1/RD-browsed-recipe/
```

Undo the action of browsing a recipe (delete browsing history).

A BrowsedRecipe id must be passed to the endpoint url after the equeal sign.

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## POST Create Shopping Lst

```
http://127.0.0.1:8000/userdata/recipe=1/create-shopping-lst/
```

Create Shopping Lst does not have any payload. The endpoint will take the recipe id and the currently logged-in user to record this action to the model.

A recipe id must be passed to the endpoint url after the equeal sign.

**Note**: ShoppingList model only takes recipe id and user id as the attributes/FK. If a user want to view all the ingredients added to the shopping list, please refer to Get Shopping List

Example Payload: None

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## DELETE Delete Shopping Lst

```
http://127.0.0.1:8000/userdata/shopping-lst=1/RD-shopping-lst/
```

Delete a recipe from shopping list.

A shopping list id must be passed to the endpoint url after the equeal sign.

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## POST Favorite Recipe

```
http://127.0.0.1:8000/userdata/recipe=1/create-favorited-recipe/
```

Favourite Recipe does not have any payload. The endpoint will take the recipe id and the currently logged-in user to record this action to the model.

A recipe id must be passed to the endpoint url after the equeal sign.

Example Payload: None

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

## DELETE Delete Favorite Recipe

```
http://127.0.0.1:8000/userdata/favorited-recipe=1/RD-favorited-recipe/
```

Remove a favourited recipe from the current user,

A favourited recipe id must be passed to the endpoint url after the equeal sign.

**AUTHORIZATION** Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

# User Function

userfunction provides a series of basic functionalities for a website.

**Note:**

- All requests are GET and all data input will be sent as request parameters. Therefore, there is no payload for most of the APIs in this section.
- The data can be access through request_parameters instead of self.context or self.kwargs

**Pagination:**

- The developer can add page and count parameters for pagination in Searchs, My Recipe and Popular Recipes.

## GET Ingredient Autocomplete

```
http://127.0.0.1:8000/userfunction/autocomplete/?ingredient=Po
```

Return a list of ingredient names.

This API is case-insensitive and return records based on the starting characters of a particular field value.

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

### PARAMS

| | |
|---|---|
| **ingredient** | Po<br>This field is optional |

## GET Popular Recipes

```
http://127.0.0.1:8000/userfunction/popular-recipes/
```

Return a list of recipes based on the number of likes in a non-increasing order.

## GET My Recipe

```
http://127.0.0.1:8000/userfunction/my-recipe/?page=1&count=2
```

Return three list of recipes of current logged in user:

- created
- favorited
- interacted (i.e. create, like, rate, or comment)

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

### PARAMS

| | |
|---|---|
| **page** | 1 |
| **count** | 2 |

## GET Search by Name

```
http://127.0.0.1:8000/userfunction/search/name/?name=Soup&cuisine=Chinese&diet=Others&time=2&unit=hours&count=2&page=1
```

Return a list of recipes whose title contains a given input string, then filter by cuisine, diet, and time.

Recipes returned in the order of popularity, please refer to Popular Recipes API.

### PARAMS

| | |
|---|---|
| **name** | Soup<br>This field is optional |
| **cuisine** | Chinese<br>This field is optional |
| **diet** | Others<br>This field is optional |
| **time** | 2<br>This field is optional |
| **unit** | hours<br>This field is optional |
| **count** | 2 |
| **page** | 1 |

## GET Search by Creator

```
http://127.0.0.1:8000/userfunction/search/creator/?creator=test&cuisine=Chinese&diet=Others&time=2&unit=hours&page=1&count=2
```

Return a list of recipes whose creator's username contains a given input string, then filter by cuisine, diet, and time.

Recipes returned in the order of popularity, please refer to Popular Recipes API.

### PARAMS

| | |
|---|---|
| **creator** | test<br>This field is optional |
| **cuisine** | Chinese<br>This field is optional |
| **diet** | Others<br>This field is optional |
| **time** | 2<br>This field is optional |
| **unit** | hours<br>This field is optional |
| **page** | 1 |
| **count** | 2 |

## GET Search by Ingredient

```
http://127.0.0.1:8000/userfunction/search/ingredient/?ingredient=Pork&cuisine=Chinese&diet=Others&time=2&unit=hours&page=1&count=2
```

Return a list of recipes whose ingredient's name contains a given input string, then filter by cuisine, diet, and time.

Recipes returned in the order of popularity, please refer to Popular Recipes API.

### PARAMS

| | |
|---|---|
| **ingredient** | Pork<br>This field is optional |
| **cuisine** | Chinese<br>This field is optional |
| **diet** | Others<br>This field is optional |
| **time** | 2<br>This field is optional |

| | |
|---|---|
| **unit** | hours<br>This field is optional |
| **page** | 1 |
| **count** | 2 |

## GET Get Shopping List

```
http://127.0.0.1:8000/userfunction/shopping-list/
```

Return a list of ingredients and a total amount with unit.

### AUTHORIZATION Bearer Token

| | |
|---|---|
| **Token** | {{token}} |

# Endpoint

| Endpoint Parameters | Representation |
|---|---|
| uid | User ID |
| rid | Recipe ID |
| sid | Step ID |
| iid | Ingredient ID |
| riid | RecipeIngredient ID |
| cid | Comment ID |
| rtid | Rating ID |
| slid | Shopping List ID |

Signup
Endpoint: /accounts/signup/
Methods: Post
Description: signs up and creates a user

Login
Endpoint: /accounts/login/
Methods: Post
Description: logs the user in with token auth

View Profile
Endpoint: /accounts/profile/details/
Methods: Get
Description: displays the request user's information
Login required

Edit Profile
Endpoint: /accounts/profile/edit/
Methods: Put
Description: updates the request user's information
Login required

Create Recipe
Endpoint: recipes/create-recipe/
Methods: Post
Description: creates a recipe under the request user

Login required

Update Recipe
Endpoint: recipes/recipe=<rid>/RUD-recipe/
Methods: Put
Login required

Delete Recipe
Endpoint: recipes/recipe=<rid>/RUD-recipe/
Methods: Del
Description: delete a recipe
Login required

Create Ingredient
Endpoint: recipes/recipe=<rid>/create-ingredient/
Methods: Post
Description: create a raw ingredient

Create Recipe Ingredient
Endpoint: recipes/recipe=<rid>&ingredient=<iid>/create-recipe-ingredient/
Methods: Post
Description: add an ingredient to a recipe by creating a relation between them

Update Recipe Ingredient
Endpoint: recipes/recipe=<rid>&ingredient=<iid>/RUD-recipe-ingredient/
Methods: Put
Description: update an ingredient in a recipe

Delete Recipe Ingredient
Endpoint: recipes/recipe=<rid>&ingredient=<iid>/RUD-recipe-ingredient/
Methods: Del
Description: remove an ingredient in a recipe

Create Step
Endpoint: recipes/recipe=<rid>/create-step
Methods: Post
Description: create a step for a recipe

Update Step
Endpoint: recipes/recipe=<rid>&step=<sid>/RUD-step/
Method: Put
Description: update a step for a recipe

Delete Step
Endpoint: recipes/recipe=<rid>&step=<sid>/RUD-step/
Method: Del
Description: delete a step for a recipe

View Recipe
Endpoint: recipes/recipe=<rid>/details/
Method: Get
Description: display details of a recipe

Create Rating
Endpoint: userdata/recipe=<rid>/create-rating/
Method: Post
Description: create a rating by the user for a recipe

Edit Rating
Endpoint: userdata/recipe=<rid>/RU-rating/
Method: Put
Description: update a rating by the use for a recipt

Create Comment
Endpoint: userdata/recipe=<rid>/create-comment/
Method: Post
Description: create a comment by the user on a recipe

Edit Comment
Endpoint: userdata/recipe=<rid>/RUD-comment/
Method: Put
Description: edit a comment by the user on a recipe

Delete Comment
Endpoint: userdata/recipe=<rid>/RUD-comment/
Method: Del
Description: removes a comment for the user on a recipe

Liked Recipe
Endpoint: userdata/recipe=<rid>/create-liked-recipe/
Method: Post
Description: allows the user to like a recipe

Delete Liked Recipe
Endpoint: userdata/recipe=<rid>/RD-liked-recipe/
Method: Del

Description: unlike a recipe for the user

Favorite Recipe
Endpoint: userdata/recipe=<rid>/create-favorite-recipe/
Method: Post
Description: allows the user to favorite a recipe

Delete Favorite Recipe
Endpoint: userdata/recipe=<rid>/RD-favorited-recipe/
Method: Del
Description: unfavorite a recipe for the user

Create Shopping List
Endpoint: userdata/recipe=<rid>/create-shopping-list/
Method: Post
Description: place a recipe in the user's shopping list

Delete Shopping List
Endpoint: userdata/recipe=<rid>/RD-shopping-list/
Method: Del
Description: remove a recipe from the user's shopping list

Ingredient Autocomplete
Endpoint: userfunction/autocomplete/
Method: Get
Description: automatically completes the ingredient name that starts with user input

Popular Recipes
Endpoint: userfunction/popular-recipes/
Method: Get
Description: display a list of recipes sorted by the number of likes

My Recipe
Endpoint: userfunction/my-recipe/
Method: Get
Description: display all recipes the user created, favorited and interacted with in three separate lists. Interacted recipes include all recipes the user created, liked, rated or commented

Search by Name
Endpoint: userfunction/search/name/
Method: Get
Description: display recipes with the specified name

Search by Creator
Endpoint: userfunction/search/creator/
Method: Get
Description: display recipes created by a specific creator

Search by Ingredient
Endpoint: userfunction/search/ingredient/
Method: Get
Description: display recipes that contain a specific ingredient

Get Shopping List
Endpoint: userfunction/shopping-list/
Method: Get
Description: tolly up all the ingredients in the user's shopping list items and display the total amount of each ingredient

# Model

User

| Attributes | Data Type | Nullable |
|---|---|---|
| Username | varchar | F |
| Email | varchar | F |
| Password1 | varchar | F |
| Password2 | varchar | F |
| Phone | varchar | T |
| Avatar | Image | T |

Recipe

| Attributes | Data Type | Nullable |
|---|---|---|
| Title | varchar | F |
| User (FK) | ForeignKey(User) | F |
| Picture | Image | F |
| Description | varchar | F |
| Time | varchar | F |
| Time_unit | varchar | F |
| Cuisine | varchar | F |
| Diet | varchar | F |

Step

| Attributes | Data Type | Nullable |
|---|---|---|
| Recipe (FK) | ForeignKey(Recipe) | F |
| Number | Int | F |
| Description | Text input | F |
| Picture | Image | F |

Ingredient

| Attributes | Data Type | Nullable |
|---|---|---|
| Name | varchar | F |

RecipeIngredient

| Attributes | Data Type | Nullable |
|---|---|---|
| Recipe (FK) | ForeignKey(Recipe) | F |
| Ingredient (FK) | ForeignKey(Ingredient) | F |
| Amount | Float | F |
| Unit | varchar | F |

Rating

| Attributes | Data Type | Nullable |
|---|---|---|
| User (FK) | ForeignKey(User) | F |
| Recipe (FK) | ForeignKey(Recipe) | F |
| Score | Int | F |

Comment

| Attributes | Data Type | Nullable |
|---|---|---|
| User (FK) | ForeignKey(User) | F |
| Recipe (FK) | ForeignKey(Recipe) | F |
| Message | Text input | F |
| File | File upload | T |

LikedRecipe

| Attributes | Data Type | Nullable |
|---|---|---|
| User (FK) | ForeignKey(User) | F |
| Recipe (FK) | ForeignKey(Recipe) | F |

FavoritedRecipe

| Attributes | Data Type | Nullable |
|---|---|---|
| User (FK) | ForeignKey(User) | F |
| Recipe (FK) | ForeignKey(Recipe) | F |

ShoppingList

| Attributes | Data Type | Nullable |
|---|---|---|
| User (FK) | ForeignKey(User) | F |
| Recipe (FK) | ForeignKey(Recipe) | F |