Keshav Worathur
CSC309 Winter 2023

# EasyChef API Reference

## Accounts Endpoints

### /accounts/register/
Methods: POST
Description: Register a new user account, using the provided email as the username
Payloads: email, password1, password2
Response JSON: email
Errors :
- 400: User with this email already exists
- 400: Passwords do not match

Scope: All users

### /accounts/login/
Methods: POST
Description: Authenticate the user with the provided credentials
Fields/payload: email, password
JSON Response: access, refresh
Errors :
- 401: No active account found with the given credentials

Scope: All users

### /accounts/view/
Methods: GET
Description: View all details for the currently authenticated account
Fields/payload: None
JSON Response: first_name, last_name, email, phone_number, avatar
Errors :
- None

Scope: Logged-in users

### /accounts/edit/
Methods: POST
Description: Update any combination of fields of the user profile except email, and optionally change password
Fields/payload: first_name, last_name, avatar, phone_number, password1, password2
JSON Response: email, first_name, last_name, avatar, phone_number
Errors :
- 400: Passwords do not match

Scope: All users

Keshav Worathur
CSC309 Winter 2023

---

# Recipe Endpoints

## /recipes/media/upload

Methods: POST
Description: Upload a pciture/video to be used in the recipe media gallery, instruction image, or comment media
Fields/payload: upload
JSON Response: id,  upload (file path to upload on server)
Errors :

- 400: File of type * not allowed; allowed file types are jpg, png, mp4

Scope: Logged-in users

## /recipes/ingredients/create/

Methods: POST
Description: Create a new ingredient, or return existing if it exists
Fields/payload: ingredient_name
JSON Response: id,  ingredient_name
Scope: Logged-in users

## /recipes/ingredients/matches/

Methods: GET
Description: Return autocomplete results for ingredient search
Fields/payload: ingredient_name
JSON Response: matches (list of names)
Scope: Any user

## /recipes/add/

Methods: POST
Description: Create a complete recipe and log the creator of the recipe as the current user
Payloads:

- name
- diets (array of {diet})
- cuisine
- serving_size
- prep_time
- instructions (array of {step ,instruction , [upload_id]})
- ingredients (array of {quantity, units, ingredient})

- upload_ids : (int array)

JSON Response:

- name
- diets (array of {diet})
- cuisine
- serving_size
- prep_time
- instructions (array of {step ,instruction , upload(path to file on server)})
- ingredients (array of {quantity, units, ingredient})
- uploads : array of {upload: path to file on server}

Errors :

- 404: Not found if provided an invalid upload_id

Scope: Logged-in users

## /recipes/<recipe_id>/details/

Methods: GET
Description: Retrieve the details of recipe with recipe id recipe_id
Fields/payload: None
JSON Response:

- name
- diets (array of {diet})
- cuisine
- serving_size
- prep_time
- instructions (array of {step ,instruction , upload(path to file on server)})
- ingredients (array of {quantity, units, ingredient})
- uploads : array of {upload: path to file on server}

Errors:

- None

Scope: All users

## /recipes/<recipe_id>/edit/

Methods: PATCH
Description: Edit the recipe with id recipe_id
Payloads:

- name
- diets (array of {diet})
- cuisine
- serving_size
- prep_time
- instructions (array of {step ,instruction , [upload_id]})
- ingredients (array of {quantity, units, ingredient})
- upload_ids : (int array)

JSON Response:

- name
- diets (array of {diet})
- cuisine
- serving_size
- prep_time
- instructions (array of {step ,instruction , upload(path to file on server)})
- ingredients (array of {quantity, units, ingredient})
- uploads : array of {upload: path to file on server}

Errors:
Scope: Logged-in users

## /recipes/<recipe_id>/delete/

Methods: DELETE
Description: Delete the recipe with id recipe_id
Fields/payload: None
JSON Response: Empty
Errors:
Scope: Logged-in users

## /recipes/search/

Methods: POST
Description: Search recipe by name, creator, diet, cuisine, or prep time
Fields/payload: name, creator (first name), diets (array of strings), cusine, prep_time
JSON Response: Return a page of results (max 25). Each recipe has the fields specified in the output for /recipes/edit/. Recipes are sorted by the number of likes they received
Errors:

- None

Scope: All users

## /recipes/<recipe_id>/comments/new/

Methods : POST
Description: Post a new comment on a recipe, or post a reply to an existing comment
Fields/payload: content, [upload_ids]
JSON Response: id, content, last_edited, creator (with all the fields present in output of /accounts/profile/view), uploads (array of {upload: file to path on server}), reply_id
Errors:

- 404: No result if reply comment is invalid

Scope: Logged-in users

## /recipes/<recipe_id>/comments/all/

Methods: GET
Description: Retrieve all comments for the recipe with id recipe_id
Fields/payload: None
JSON Response: Paginated view of top level comments (25 results per page), ordered by last edited time. Each comment contains the fields in the JSON Response of comments/new/ with the addition of a replies field, which is an array of comment objects.
Errors:

- None

Scope: Any user

### /recipes/<recipe_id>/comments/<comment_id>/edit/

Methods: PATCH
Description: Edit the comment with id comment_id
Fields/payload: content
JSON Response: id, content, last_edited, creator (with all the fields present in output of /accounts/profile/view)
Errors:

- None

Scope: Logged-in users

---

## Social Media Endpoints

### /social/<recipe_id>/favorite/

Methods: POST
Description: Favorite recipe with id recipe_id
Fields/payload: None
JSON Response: Empty
Errors:

- None

Scope: Logged-in users

### /social/<recipe_id>/favorites_count/

Methods: GET
Description: Get the favorites count for recipe with id recipe_id
Fields/payload: None
JSON Response: favorite_count

Errors:
Scope: All users

## /social/<recipe_id>/unfavorite/

Methods: POST
Description: Unfavorite the recipe with id recipe_id
Fields/payload: None
JSON Response: Empty
Errors:
Scope: Logged-in users

## /social/<recipe_id>/rate/

Methods : POST
Description: Assign rating to recipe with id recipe id
Fields/payload: rating
JSON Response: recipe, rating, user
Errors:
Scope: Logged-in users

## /social/<recipe_id>/ratings/

Methods: GET
Description: Retrieve the average rating for the recipe with id recipe_id
Fields/payload: None
JSON Response: average_rating
Errors:
Scope: All users

---

# Shopping Cart Endpoints

## /shopping/edit/<recipe_id>/

Methods: POST
Description: Edit the quantity of the recipe with id recipe_id in shoping cart, adding it if it does not exist.
Fields/payload: quantity
JSON Response: None
Errors:
Scope: Logged-in users

## /shopping/remove/<recipe_id>/

Methods: DELETE
Description: Remove the recipe with id recipe_id from the shopping cart
Fields/payload: None
JSON Response: None
Errors:
Scope: Logged-in users

## /shopping/cart/

Methods: GET
Description: View all items in the current user's shopping cart
Fields/payload: None
JSON Response: Return a page of recipes in the shopping cart (max 25). Each recipe has the same fields as returned in the output of recipes/<recipe_id>details/. Additionally, each recipe has a quantity attribute, specifying the quantity added to the shopping cart
Errors:
Scope: Logged-in users

## /shopping/cart/ingredients

Methods: GET
Description: Group all ingredients used by cart recipes by name and units and compute the total amount needed, factoring the quantity of the recipe in the shopping cart
Fields/payload: None
JSON Response: result (list of [ingredient, units, quantity])
Errors:
Scope: Logged-in users

# Description of Models:

User: Each user has an email which uniquely identifies them, a password, and optionally a first name, last name , phone number, and avatar.

RecipeUpload - Stores a media upload which is attached to one or more recipe, comments, or instructions.

Comment - Each comment has a user which posts it, text content, last edited time, and the recipe to which it was posted. Each comment may optionally have associated

media. A user can make multiple comments, and a recipe can have many comments posted to it.

Ingredient - Each ingredient has a name, quantity, and units of measure. A recipe can have many ingredients, and a given ingredient may appear in many recipes.

Instruction - Each instruction has a step number and description. Each instruction may optionally have an image description. A recipe may have many instructions, and the same instruction can be shared by many recipes.

Diet - Each diet has a name. Many recipes can be tagged with the same diet, and a recipe can be tagged with multiple diets.

RatingRecord - Each rating record has a user which made the rating, the recipe to which the rating is assigned to, and the numerical value of the rating on a scale from 1 to 5. A recipe can be rated by many users, and a user can rate multiple recipes.

CartItem - Each CartItem records a user and a recipe which they have added to their shopping cart. Additionally, each CartItem has a quantity which denotes the quantity in the shopping cart.