Recipes: API Spec

CS 493: Cloud Application Development

Portfolio Assignment

Fall 2020

Oregon State University

Last Update: Dec 2, 2020

Application URL: https://anderja8-recipes-api.appspot.com

Account Creation Link: First link on webpage above

Data Model

Users

Name	Type	Notes	Required	Valid Values
id	Integer	The id of the user, datastore automatically generates it. Note, it is not used with this API. Instead, sub is used as the user ID for requests to user endpoints. The “id” value is not even returned by a GET /users request.	Auto-generated	N/A
given_name	String	The given name of the user as supplied by Google in the JWT.	Auto-generated	N/A
family_name	String	The family name of the user as supplied by Google in the JWT.	Auto-generated	N/A
sub	Integer	The sub of the user as supplied by Google in the JWT. Used as the de-facto unique identifier for users.	Auto-generated	N/A

As stated in the table above, the unique identifier for a user of this API is the sub property from their Google issued JWT. All information in this table is generated via the payload of the JWT. The payload of the JWT is read using the node.js google-auth-library. Each of the other entities in this API contain an owner_id property, which corresponds to the sub value in this entity. For all requests requiring a user identifier, a supplied JWT is expected and will be used to determine the identity of the user. Deleting a user will require a valid specification of the user sub in the request URL and a valid JWT (more on this in the delete a user section).

Recipes

Name	Type	Notes	Required	Valid Values
id	Integer	The id of the recipe, datastore automatically generates it.	Auto-generated	N/A
name	String	The name of the recipe.	Yes	Any string
description	String	A description of the recipe.	Yes	Any string
instructions	String	Instructions on how to combine the ingredients.	No	Any string
owner_id	Integer	Sub of the user that created the recipe. Essentially foreign key from the users table.	Auto-generated	N/A
public	Boolean	Determines whether other users can view this recipe	Yes	True or False
ingredients	Object Array	An array of objects. Each object contains an “id” (integer) corresponding to an ingredient id and a “quantity” (string) corresponding to the amount of that ingredient that the recipe calls for.	No	N/A generated by the server after linking an ingredient to a recipe

Creating a recipe requires a valid JWT, which maps the recipe to the users entity via the owner_id. There is a many to many relationship between this entity and the other non-user entity, ingredients. This relationship is maintained via arrays stored in both entities. The arrays are updated by the server automatically when ingredients and recipes are linked, and when ingredients or recipes are deleted.

Ingredients

Name	Type	Notes	Required	Valid Values
id	Integer	The id of the ingredient, datastore automatically generates it.	Auto-generated	N/A
name	String	The name of the ingredient.	Yes	Any string
stock	String	The user's current stock level of the ingredient.	No	Any string
owner_id	Integer	Sub of the user that created the ingredient. Essentially foreign key from the users table.	Auto-generated	N/A
last_updated	Timestamp	Timestamp of the last time this ingredient was created or modified.	Auto-generated	N/A
recipes	Object Array	An array of objects. Each object contains an "id" (integer) corresponding to a recipe id.	No	N/A generated by the server after linking an ingredient to a recipe

Creating an ingredient requires a valid JWT, which maps the ingredient to the users entity via the owner_id. There is a many to many relationship between this entity and the other non-user entity, recipes. This relationship is maintained via arrays stored in both entities. The arrays are updated by the server automatically when ingredients and recipes are linked, and when ingredients or recipes are deleted.

Generate a JWT

Allows users to generate a JWT

GET /user-info

Notes:

- A valid Google account is required in order for this page to generate a JWT.
- Navigates users through standard oauth 2.0 flow.
- This app stores cookies so that subsequent requests do not require the oauth flow.
- If the user wishes to regenerate their JWT, or generate one for a different account, a request must be made to the /logout route (documented on next page) prior to requesting /user-info.

Log out

Allows users to logout. Useful for regenerating a JWT, or generating a JWT for a new account.

GET /logout

Notes:

- Destroys the current session.
- Redirects the user to the google account logout page, https://google.com/accounts/logout.

Create a Recipe

Allows the user to create a recipe. Attempting to create a recipe without a user account will automatically create a user account.

POST /recipes

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
name	String	The name of the recipe	Yes
description	String	A description of the recipe	Yes
instructions	String	Instructions to make the recipe	No
public	Boolean	Whether the recipe will be visible to other users	Yes

Request Body Example

```
{
  "name": "Latte",
  "description": "A simple caffeinated beverage",
  "instructions": "First pour the espresso, followed by the milk",
  "public": true
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the recipe will not be created. Any additional fields will be ignored.
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

- Datastore will automatically generate an ID and store it with the entity being created.
- The `self` attribute will contain the live link to the REST resource corresponding to this recipe.

Success

Status: 201 Created

```
{
  "id": 1,
  "name": "Latte",
  "description": "A simple caffeinated beverage",
  "instructions": "First pour the espresso, followed by the milk",
  "public": true,
  "owner_id": 1,
  "ingredients": []
  "self": "https://anderja8-recipes-api.appspot.com/recipes/1
}
```

Failure

Status: 400 Bad Request

```
{
"Error": "The request object is missing at least one of the required attributes"
}
```

Status: 401 Unauthorized

```
{
"Error": "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```

Status: 406 Not Acceptable

```
{
"Error": "Server only sends application/json data"
}
```

Get a Recipe

Allows the user to read a recipe

GET /recipes/:recipe_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>" in order to view all recipes owned by the user. If the authorization header is malformed or not present, a list of all public recipes will be returned instead.
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the recipe is private and the owner_id does not match the JWT sub
Failure	404 Not Found	Occurs if the recipe ID is not present in the datastore
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

Success

Status: 200 OK


```
{
  "id": 1,
  "name": "Latte",
  "description": "A simple caffeinated beverage",
  "instructions": "First pour the espresso, followed by the milk",
  "public": true,
  "owner_id": 1,
  "ingredients": [
     {
       "id": 1,
```

```
        "name": "espresso",
        "quantity": "3 oz"
      },
      {
        "id": 2,
        "name": "steamed milk",
        "quantity": "10 oz"
      },
  ]
  "self": "https://anderja8-recipes-api.appspot.com/recipes/1
}
```


Failure

Status: 401 Unauthorized


```
{
"Error": "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```


Status 403 Forbidden


```
{
"Error": "The recipe with this recipe_id is owned by someone else"
}
```


Status 404 Not Found


```
{
"Error": "No recipe with this recipe_id exists"
}
```


Status: 406 Not Acceptable


```
{
"Error": "Server only sends application/json data"
}
```

Get Recipes

Allows the user to read recipes. They can either read all public recipes, by providing no authorization, or all of their recipes, by providing their JWT.

GET /recipes

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>" in order to view all recipes owned by the user. If the authorization header is malformed or not present, a list of all public recipes will be returned instead.
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

- The `self` attribute will contain the live link to the REST resource corresponding to this recipe.
- Pagination is used with a limit of 5 recipes per page. If there is more data, a next link will be present that leads to the next page of results. If there is no more data, the next link will not be present.

Success – Valid JWT

Status: 200 OK

```
{
  "recipes": [
   {
    "id": 1,
    "name": "Latte",
     "description": "A simple caffeinated beverage",
    "instructions": "First pour the espresso, followed by the milk",
     "public": true,
```

```
    "owner_id": 1,
    "ingredients": [
      {
        "id": 1,
        "name": "espresso",
        "quantity": "3 oz"
      },
      {
        "id": 2,
        "name": "steamed milk",
        "quantity": "10 oz"
      },
    ]
    "self": "https://anderja8-recipes-api.appspot.com/recipes/1
   },
   {
    "id": 2,
    "name": "Seaweed",
    "description": "Plankton's fave",
   "instructions": "grow it?",
    "public": false,
    "owner_id": 1,
    "ingredients": [
      {
        "id": 3,
        "name": "sea",
        "quantity": "50%"
      },
      {
        "id": 4,
        "name": "weed",
        "quantity": "50%"
      },
    ]
    "self": "https://anderja8-recipes-api.appspot.com/recipes/2
   },
   {
    "id": 3,
    "name": "Momma's Breakfast Casserole",
    "description": "My fave",
   "instructions": ""
    "public": true,
    "owner_id": 1,
    "ingredients": [
      {
        "id": 5,
        "name": "love",
        "quantity": "two helpings"
      },
    ]
    "self": "https://anderja8-recipes-api.appspot.com/recipes/3
   },
   {
    "id": 4,
    "name": "Placeholder recipe",
```

```
    "description": "Not sure yet",
    "instructions": ""
    "public": true,
    "owner_id": 1,
    "ingredients": []
    "self": "https://anderja8-recipes-api.appspot.com/recipes/4
   },
   {
    "id": 5,
    "name": "Placeholder recipe 2",
    "description": "Not sure yet",
    "instructions": ""
    "public": true,
    "owner_id": 1,
    "ingredients": []
    "self": "https://anderja8-recipes-api.appspot.com/recipes/5
   },
  ],
  "next": https://anderja8-recipes-api.appspot.com/recipes?endCursor=1234
}
```

Success – Invalid JWT

Status: 200 OK

```
{
  "recipes": [
   {
    "id": 1,
    "name": "Latte",
    "description": "A simple caffeinated beverage",
    "instructions": "First pour the espresso, followed by the milk",
    "public": true,
    "owner_id": 1,
    "ingredients": [
      {
        "id": 1,
        "name": "espresso",
        "quantity": "3 oz"
      },
      {
        "id": 2,
        "name": "steamed milk",
        "quantity": "10 oz"
      },
    ]
    "self": "https://anderja8-recipes-api.appspot.com/recipes/1
   },
   {
    "id": 6,
    "name": "Weedsea",
    "description": "Not plankton's fave",
    "instructions": "grow it?",
    "public": true,
    "owner_id": 2,
    "ingredients": [
```

```
      {
        "id": 3,
        "name": "sea",
        "quantity": "50%"
      },
      {
        "id": 4,
        "name": "weed",
        "quantity": "50%"
      },
    ]
    "self": "https://anderja8-recipes-api.appspot.com/recipes/6
   },
   {
    "id": 3,
    "name": "Momma’s Breakfast Casserole",
    "description": "My fave",
    "instructions": ""
    "public": true,
    "owner_id": 1,
    "ingredients": [
      {
        "id": 5,
        "name": "love",
        "quantity": "two helpings"
      },
    ]
    "self": "https://anderja8-recipes-api.appspot.com/recipes/3
   },
   {
    "id": 4,
    "name": "Placeholder recipe",
    "description": "Not sure yet",
    "instructions": ""
    "public": true,
    "owner_id": 1,
    "ingredients": []
    "self": "https://anderja8-recipes-api.appspot.com/recipes/4
   },
   {
    "id": 5,
    "name": "Placeholder recipe 2",
    "description": "Not sure yet",
    "instructions": ""
    "public": true,
    "owner_id": 1,
    "ingredients": []
    "self": "https://anderja8-recipes-api.appspot.com/recipes/5
   },
  ],
  "next": https://anderja8-recipes-api.appspot.com/recipes?endCursor=1234
}
```

Failure

Status: 406 Not Acceptable

```
{
"Error":  "Server only sends application/json data"
}
```

Replace a Recipe

Allows the user to replace the name, description, instructions, and public value of a recipe

PUT /recipes/:recipe_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
name	String	The name of the recipe	Yes
description	String	A description of the recipe	Yes
instructions	String	Instructions to make the recipe	No, but will default to empty if blank
public	Boolean	Whether the recipe will be visible to other users	Yes

Request Body Example

```
{
  "name": "Latte",
  "description": "A simple caffeinated beverage",
  "instructions": "First pour the espresso, followed by the milk",
  "public": true
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the recipe will not be created. Any additional fields will be ignored.
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the recipe owner_id does not match the JWT

		sub
Failure	404 Not Found	Occurs if the recipe id is not present in the datastore
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

- The `self` attribute will contain the live link to the REST resource corresponding to this recipe.

Success

Status: 200 OK

```
{
  "id": 1,
  "name": "Latte",
  "description": "A simple caffeinated beverage",
  "instructions": "First pour the espresso, followed by the milk",
  "public": true,
  "owner_id": 1,
  "ingredients": [
     {
       "id": 1,
       "name": "espresso",
       "quantity": "3 oz"
     },
     {
       "id": 2,
       "name": "steamed milk",
       "quantity": "10 oz"
     },
  ]
  "self": "https://anderja8-recipes-api.appspot.com/recipes/1
}
```

Failure

Status: 400 Bad Request

```
{
"Error": "The request object is missing at least one of the required attributes"
}
```

Status: 401 Unauthorized

```
{
"Error": "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```

Status 403 Forbidden

```
{
"Error": "The recipe with this recipe_id is owned by someone else"
}
```

Status 404 Not Found

<table>
<tr><td>

<pre>
{
"Error": "No recipe with this recipe_id exists"
}
</pre>

</td></tr>
<tr><td>Status: 406 Not Acceptable

<pre>
{
"Error": "Server only sends application/json data"
}
</pre>

</td></tr>
</table>

Patch a Recipe

Allows the user to update a set of the name, description, instructions, and public attributes of a recipe

PATCH /recipes/:recipe_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
name	String	The name of the recipe	No
description	String	A description of the recipe	No
instructions	String	Instructions to make the recipe	No
public	Boolean	Whether the recipe will be visible to other users	No

Request Body Example

```
{
  "description": "A crazy caffeinated beverage",
  "instructions": "First go crazy, then pour the espresso, followed by the milk",
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the recipe will not be created. Any additional fields will be ignored.
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the recipe owner_id does not match the JWT sub
Failure	404 Not Found	Occurs if the recipe id is not present in the datastore
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

- The `self` attribute will contain the live link to the REST resource corresponding to this recipe.

Success

Status: 200 OK

```
{
  "id": 1,
  "name": "Latte",
  "description": " A crazy caffeinated beverage ",
  "instructions": " First go crazy, then pour the espresso, followed by the milk",
  "public": true,
  "owner_id": 1,
  "ingredients": [
     {
       "id": 1,
       "name": "espresso",
       "quantity": "3 oz"
     },
     {
       "id": 2,
       "name": "steamed milk",
       "quantity": "10 oz"
     },
  ]
  "self": "https://anderja8-recipes-api.appspot.com/recipes/1
}
```

Failure

Status: 400 Bad Request

```
{
"Error": "The request object is missing at least one of the required attributes"
}
```

Status: 401 Unauthorized

```
{
"Error": "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```

Status 403 Forbidden

```
{
"Error": "The recipe with this recipe_id is owned by someone else"
}
```

Status 404 Not Found

```
{
"Error": "No recipe with this recipe_id exists"
```

}

Status: 406 Not Acceptable
 	
{
"Error": "Server only sends application/json data"
}

Delete a Recipe

Allows the user to delete a recipe. A JWT must be provided that matches the owner_id of the recipe.

DELETE /recipes/:recipe_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the recipe owner_id does not match the JWT sub
Failure	404 Not Found	Occurs if the recipe id is not present in the datastore
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

Success

Status: 204 No Content

- Deleting a recipe will remove that recipe_id from all ingredient recipes arrays in the datastore.

Failure

Status: 401 Unauthorized
 	
{
"Error": "malformed authorization header, should be "authorization":"Bearer <token>"
}

Status 403 Forbidden

{
"Error": "The recipe with this recipe_id is owned by someone else"
}

Status 404 Not Found
 	
{
"Error": "No recipe with this recipe_id exists"
}

Status: 406 Not Acceptable
 	
{
"Error": "Server only sends application/json data"
}

Create an Ingredient

Allows the user to create an ingredient. Attempting to create a recipe without a user account will automatically create a user account.

POST /ingredients

Request

Request Headers

- “Authorization” header must be set to “Bearer <Your JWT>”
- “Accepts” header must be set to “application/json”

Request Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
name	String	The name of the ingredient	Yes
stock	String	The owner’s stock of this ingredient	No

Request Body Example

```
{
  "name": "espresso",
  "stock": "10 oz"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	
Failure	400 Bad Request	If the request is missing the name attribute, the ingredient will not be created. Any additional fields outside of name and stock will be ignored.
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

- Datastore will automatically generate an ID and store it with the entity being created.

- The `self` attribute will contain the live link to the REST resource corresponding to this recipe.

Success

Status: 201 Created

```
{
  "id": 1,
  "name": "espresso",
  "stock": "10 oz",
  "owner_id": 1,
  "last_updated": "Wed Dec 02 2020 21:27:18 GMT-0700 (Mountain Standard Time)"
  "recipes": []
  "self": "https://anderja8-recipes-api.appspot.com/ingredients/1
}
```

Failure

Status: 400 Bad Request

```
{
"Error": "The request object is the required \"name\" attribute"
}
```

Status: 401 Unauthorized

```
{
"Error": "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```

Status: 406 Not Acceptable

```
{
"Error": "Server only sends application/json data"
}
```

Get an Ingredient

Allows the user to read a recipe

GET /ingredients/:ingredient_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the ingredient's owner_id does not match the JWT sub
Failure	404 Not Found	Occurs if the ingredient ID is not present in the datastore
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

Success

Status: 200 OK

```
{
  "id": 1,
  "name": "espresso",
  "stock": "10 oz",
  "owner_id": 1,
  "last_updated": "Wed Dec 02 2020 21:27:18 GMT-0700 (Mountain Standard Time)"
  "recipes": [
    {
      "name": "latte",
      "id": 1
    }
  ]
```

```
  "self": "https://anderja8-recipes-api.appspot.com/ingredients/1
}
```

Failure

Status: 401 Unauthorized

```
{
"Error": "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```

Status 403 Forbidden

```
{
"Error": "The ingredient with this ingredient_id is owned by someone else"
}
```

Status 404 Not Found

```
{
"Error": "No ingredient with this ingredient_id exists"
}
```

Status: 406 Not Acceptable

```
{
"Error": "Server only sends application/json data"
}
```

Get Ingredients

Allows the user to read all of their ingredients. This endpoint will only display ingredients owned by the JWT supplied in the Authorization header.

GET /ingredients

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

- The `self` attribute will contain the live link to the REST resource corresponding to this recipe.
- Pagination is used with a limit of 5 recipes per page. If there is more data, a next link will be present that leads to the next page of results. If there is no more data, the next link will not be present.

Success – Valid JWT

Status: 200 OK


```
{
  "ingredients": [
   {
     "id": 1,
    "name": "espresso",
     "stock": "10 oz",
     "owner_id": 1,
     "last_updated": "Wed Dec 02 2020 21:27:18 GMT-0700 (Mountain Standard Time)"
```

```
    "recipes": [
      {
        "name": "latte",
        "id": 1
      }
    ]
    "self": "https://anderja8-recipes-api.appspot.com/ingredients/1
  },
  {
    "id": 2,
    "name": "milk",
    "stock": "1 gal",
    "owner_id": 1,
    "last_updated": "Tue Dec 01 2020 21:27:18 GMT-0700 (Mountain Standard Time)"
    "recipes": [
      {
        "name": "latte",
        "id": 1
      },
      {
        "name": "Chocolate Cake",
        "id": 6
      }
    ]
    "self": "https://anderja8-recipes-api.appspot.com/ingredients/2
  },
  {
    "id": 3,
    "name": "salt",
    "stock": "",
    "owner_id": 1,
    "last_updated": "Tue Dec 01 2020 21:34:18 GMT-0700 (Mountain Standard Time)"
    "recipes": [
      {
        "name": "Chocolate Cake",
        "id": 6
      }
    ]
    "self": "https://anderja8-recipes-api.appspot.com/ingredients/3
  },
  {
    "id": 4,
    "name": "mayonnaise",
    "stock": "0 jars",
    "owner_id": 1,
    "last_updated": "Wed Nov 30 1927 10:34:18 GMT-0700 (Mountain Standard Time)"
    "recipes": []
    "self": "https://anderja8-recipes-api.appspot.com/ingredients/4
  },
  {
    "id": 5,
    "name": "Cheez Whiz",
    "stock": "1 can",
    "owner_id": 1,
    "last_updated": "Wed Dec 02 2020 09:27:18 GMT-0700 (Mountain Standard Time)"
```

```
      "recipes": []
      "self": "https://anderja8-recipes-api.appspot.com/ingredients/5
    }
  ],
  "next": https://anderja8-recipes-api.appspot.com/ingredients?endCursor=1234
}
```

Failure

Status: 401 Unauthorized

```
{
"Error": "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```

Status: 406 Not Acceptable

```
{
"Error": "Server only sends application/json data"
}
```

Replace an Ingredient

Allows the user to replace the name, description, instructions, and public value of a recipe

PUT /ingredients/:ingredient_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
name	String	The name of the recipe	Yes
stock	String	Instructions to make the recipe	No, but will default to empty if blank

Request Body Example

```
{
  "name": "almond milk",
  "stock": "2 quarts"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	400 Bad Request	If the request is missing the name attribute, the ingredient will not be created. Any additional fields outside of name and stock will be ignored.
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the ingredient owner_id does not match the JWT sub.
Failure	404 Not Found	Occurs if the ingredient id is not present in the datastore.
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

- The `self` attribute will contain the live link to the REST resource corresponding to this recipe.

Success

Status: 200 OK

```
{
  "id": 2,
  "name": "almond milk",
  "stock": "2 quarts",
  "owner_id": 1,
  "last_updated": "Tue Dec 01 2020 23:27:18 GMT-0700 (Mountain Standard Time)"
  "recipes": [
    {
      "name": "latte",
      "id": 1
    },
    {
      "name": "Chocolate Cake",
      "id": 6
    }
  ]
  "self": "https://anderja8-recipes-api.appspot.com/ingredients/2
}
```

Failure

Status: 400 Bad Request

```
{
"Error": "The request object is missing at least one of the required attributes"
}
```

Status: 401 Unauthorized

```
{
"Error": "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```

Status 403 Forbidden

```
{
"Error": "The ingredient with this ingredient_id is owned by someone else"
}
```

Status 404 Not Found

```
{
"Error": "No ingredient with this ingredient_id exists"
}
```

Status: 406 Not Acceptable

```
{
"Error":  "Server only sends application/json data"
}
```

Patch an Ingredient

Allows the user to update a set of the name and stock of an ingredient

PATCH /ingredients/:ingredient_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
name	String	The name of the recipe	No
stock	String	Instructions to make the recipe	No

Request Body Example

```
{
  "name": "coconut milk"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	400 Bad Request	If the request is missing any of the 3 required attributes, the recipe will not be created. Any additional fields will be ignored.
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the recipe owner_id does not match the JWT sub
Failure	404 Not Found	Occurs if the recipe id is not present in the datastore
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

- The `self` attribute will contain the live link to the REST resource corresponding to this recipe.

Success

Status: 200 OK


```
{
  "id": 2,
  "name": "coconut milk",
  "stock": "2 quarts",
  "owner_id": 1,
  "last_updated": "Tue Dec 01 2020 23:27:18 GMT-0700 (Mountain Standard Time)"
  "recipes": [
    {
      "name": "latte",
      "id": 1
    },
    {
      "name": "Chocolate Cake",
      "id": 6
    }
  ]
  "self": "https://anderja8-recipes-api.appspot.com/ingredients/2
}
```


Failure

Status: 400 Bad Request


```
{
"Error": "The request object is missing at least one of the required attributes"
}
```


Status: 401 Unauthorized


```
{
"Error": "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```


Status 403 Forbidden


```
{
"Error": "The ingredient with this ingredient_id is owned by someone else"
}
```


Status 404 Not Found


```
{
"Error": "No ingredient with this ingredient_id exists"
}
```


Status: 406 Not Acceptable


```
{
"Error": "Server only sends application/json data"
}
```

Delete an Ingredient

Allows the user to delete an ingredient. A JWT must be provided that matches the owner_id of the ingredient.

DELETE /ingredients/:ingredient_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the ingredient owner_id does not match the JWT sub
Failure	404 Not Found	Occurs if the ingredient id is not present in the datastore
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

Success

Status: 204 No Content

- Deleting an ingredient will remove that ingredient_id from all recipe ingredients arrays in the datastore.

Failure

Status: 401 Unauthorized

```
{
"Error":  "malformed authorization header, should be "authorization":"Bearer
<token>"
}
```

Status 403 Forbidden
 	
{
"Error": "The ingredient with this ingredient_id is owned by someone else"
}

Status 404 Not Found
 	
{
"Error": "No ingredient with this ingredient_id exists"
}

Status: 406 Not Acceptable
 	
{
"Error": "Server only sends application/json data"
}

Link Recipe and Ingredient

Allows the user to link an ingredient with a recipe. The user must supply a JWT that matches the owner_id of both the recipe and ingredient.

POST /recipes/:recipe_id/ingredients/:ingredient_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
quantity	String	The quantity of the ingredient called for by the recipe	Yes

Request Body Example

```
{
  "quantity": "3 oz"
}
```

- The ingredient_id and quantity will be stored as an entry in the ingredients array of the recipe. The recipe_id will be stored as an entry in the recipes array of the ingredient.

Response

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	400 Bad Request	If the request is missing the "quantity" attribute, the ingredient and recipe will not be linked. Also occurs if the ingredient and recipe are already linked. Any additional fields in the body will be ignored.
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the owner_id of the ingredient or recipe do not

		match the sub from the JWT.
Failure	404 Not Found	Occurs if the recipe id or ingredient id is not present in the datastore.
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Success

Status: 204 No Content

Failure

Status: 400 Bad Request
 	
{
"Error": "The request object is missing the required \"quantity\" attribute"
}

Status: 401 Unauthorized
 	
{
"Error": "malformed authorization header, should be "authorization":"Bearer <token>"
}

Status 403 Forbidden
 	
{
"Error": "The recipe with this recipe_id is owned by someone else"
}

Status 404 Not Found
 	
{
"Error": "No recipe with this recipe_id exists"
}

Status: 406 Not Acceptable
 	
{
"Error": "Server only sends application/json data"
}

Update Quantity of Recipe and Ingredient Link

Allows the user to update the quantity of ingredient called for by the recipe

PUT /recipes/:recipe_id/ingredients/:ingredient_id
PATCH /recipes/:recipe_id/ingredients/:ingredient_id

- The PUT and PATCH requests to the above URLs call the same http handler on the server and are functionally equivalent. Which HTTP verb to use is up to the user.

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Type	Description	Required?
quantity	String	The quantity of the ingredient called for by the recipe	Yes

Request Body Example

```
{
  "quantity": "5 oz"
}
```

- The ingredient_id and quantity will be stored as an entry in the ingredients array of the recipe. The recipe_id will be stored as an entry in the recipes array of the ingredient.

Response

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	400 Bad Request	If the request is missing the "quantity" attribute, the ingredient and recipe will not be updated. Any additional fields in the body will be ignored.
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the

		JWT cannot be validated.
Failure	403 Forbidden	Occurs if the owner_id of the ingredient or recipe do not match the sub from the JWT.
Failure	404 Not Found	Occurs if the recipe id or ingredient id is not present in the datastore, or if the recipe and ingredient are not already linked.
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Success

Status: 204 No Content

Failure

Status: 400 Bad Request
 	
{
"Error": "The request object is missing the required \"quantity\" attribute"
}

Status: 401 Unauthorized
 	
{
"Error": "malformed authorization header, should be "authorization":"Bearer <token>"
}

Status 403 Forbidden
 	
{
"Error": "The ingredient with this ingredient_id is owned by someone else"
}

Status 404 Not Found
 	
{
"Error": " The ingredient with this ingredient_id was not linked to the recipe with this recipe_id"
}

Status: 406 Not Acceptable
 	
{
"Error": "Server only sends application/json data"
}

Delete Recipe and Ingredient Link

Allows the user to update the quantity of ingredient called for by the recipe

DELETE /recipes/:recipe_id/ingredients/:ingredient_id

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

None

Request Body Format

None

Response

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	204 No Content	
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the owner_id of the ingredient or recipe do not match the sub from the JWT.
Failure	404 Not Found	Occurs if the recipe id or ingredient id is not present in the datastore, or if the recipe and ingredient are not already linked.
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Success

Status: 204 No Content

Failure

Status: 400 Bad Request
 	
{
"Error": "The request object is missing the required \"quantity\" attribute"
}

Status: 401 Unauthorized

{
"Error": "malformed authorization header, should be "authorization":"Bearer <token>"
}

Status 403 Forbidden
 	
{
"Error": "The ingredient with this ingredient_id is owned by someone else"
}

Status 404 Not Found
 	
{
"Error": "No ingredient with this ingredient_id exists"
}

Status: 406 Not Acceptable
 	
{
"Error": "Server only sends application/json data"
}

Get User Accounts

Allows the user to view a list of all users of the api. Note: user creation happens automatically when a user posts to /recipes or /ingredients for the first time.

GET /users

Request

Request Headers

- “Accepts” header must be set to “application/json”

Request Parameters

None

Request Body

None

Request Body Format

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

Success

Status: 200 OK

```
[
 {
  "id": "1”,
  "given_name": "Spongebob",
  "family_name": "Squarepants"
 },
 {
  "id": "2”,
  "given_name": "Patrick",
  "family_name": "Star"

 }
]
```

Failure

Status: 406 Not Acceptable

```
{
"Error":  "Server only sends application/json data"
}
```

Delete a User

Allows a user to remove themselves from the app. This will delete their user account, so they will no longer be viewable after a request to GET /users. This will also remove from the datastore all recipes and ingredients owned by the user. Note: user creation happens automatically when a user posts to /recipes or /ingredients for the first time.

DELETE /users/:user_id

- The easiest way to determine a user id is to GET /recipes or /ingredients with the JWT of the user. The owner_id of these recipes and ingredients will match the user_id required for this endpoint.

Request

Request Headers

- "Authorization" header must be set to "Bearer <Your JWT>"
- "Accepts" header must be set to "application/json"

Request Parameters

None

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	401 Unauthorized	Occurs if the Authorization header is malformed, or if the JWT cannot be validated.
Failure	403 Forbidden	Occurs if the user id does not match the JWT sub
Failure	404 Not Found	Occurs if the user id is not present in the datastore
Failure	406 Not Acceptable	Occurs if the Accepts header is not "application/JSON".

Response Examples

Success

Status: 204 No Content

- Deleting a user will remove all of their recipes and ingredients from the datastore.

Failure

Status: 401 Unauthorized
 	
{
"Error": "malformed authorization header, should be "authorization":"Bearer <token>"
}

Status 403 Forbidden
 	
{
"Error": "The user with this user_id does not match the JWT"
}

Status 404 Not Found
 	
{
"Error": "No user with this user_id exists"
}

Status: 406 Not Acceptable
 	
{
"Error": "Server only sends application/json data"
}