REST Documentation

Error Codes

queryFailed Query failed (server problem)
noPermission Login lacks permission.
forbiddenField Field not allowed in the body
missingField Field missing from the request
badValue Field has bad value
notFound Entity not found in the database
badLogin Email/password combination invalid, for errors logging.
dupEmail Email duplicates an existing email
noTerms Acceptance of terms is required
noOldPwd Change of password requires an old password
oldPwdMismatch Old password that was provided is incorrect.
dupIngredient Ingredient already exists in the database
dupRating Like or dislike has already been given by that user

Resource for Recipes

Ssns/

POST A successful POST generates a browser-session cookie that will permit continued access for 2 hours. Indicated Person becomes the AU. An unsuccessful POST results in a 400 with a badLogin tag and no further information.
> *email* Email of user requesting login
> *password* Password of user

Ssns/{cookie}

Login sessions (Ssns) establish an AU and are required for most service access. A user obtains one via POST to Ssns.
GET Returns a list of current session
> *prsId* ID of Person logged in
> *loginTime* Date and time of login

DELETE Logout the current Session.

Prss/

POST
Adds a new Person. User must not be logged in.
>*email* unique Email for new person
>*firstName*
>*lastName*
>*Password*
>*termsAccepted* boolean--were site terms and conditions accepted?

Email and lastName required and must be nonempty. Error if email is nonunique. Error if terms were not accepted. Nonempty password required.

Prss/{id}

Requires AU for all requests. If AU does not match id, forbidden status is returned with empty body
GET
Returns the person that is logged in. No data for other than the AU is returned in any event. Data per person:
>*email* principal string identifier, unique across all Persons
>*_id* id of person with said email, so that URI would be Prss/{_id}
>*firstName*
>*lastName*

PUT
Updates currently logged in user with body giving an object with one or more of *firstName*, *lastName*, *password.*Attempt to change other fields in Person such as *termsAccepted* or *whenRegistered* results in BAD_REQUEST and forbiddenField error(s). An additional field *oldPassword* is required for changing *password*. Password, if supplied, must be nonempty.
DELETE
Deletes the user with specified id.

Prss/{id}/Ingr

Requires an AU. AU must match id specified in params.
GET
Returns an array of specified person's available ingredients
>*_id*
>*name* name of the ingredient
POST
Adds a new ingredient to the AU's fridge
>*name* name of the ingredient (must be unique and not already exist in the fridge)

Prss/{id}/Ingr/{itemId}

Requires an AU. AU must match id specified in params.
DELETE
Deletes an ingredient from the id's fridge. AU must match id.

Prss/{id}/Mels

Requires an AU. AU must match id specified in params.
GET
Returns an array of specified person's planned recipes.
> *_id* Id of the planned meal
> *recipeId* Recipe id corresponding to the food2fork API
> *date* Planned date for the recipe to be created

POST
Adds a planned recipe.
> *recipe* Recipe corresponding to the food2fork API recipe object
> *date* Planned date for the recipe to be created

Prss/{id}/Mels/{mealId}

Requires an AU. AU must match id specified in params.
PUT
> *recipe* Recipe corresponding to the food2fork API recipe object
> *date* Planned date for the recipe to be created

Edit a planned recipe.
DELETE
Remove a planned meal. AU must be the owner of that planned meal.

Rat/{ratId}

GET
Returns all messages, likes, and dislikes tied to a recipe.
> *likes* Amount of likes for the recipe
> *dislikes* Amount of dislikes for the recipe
> *comment* User submitted comments

Rat/{ratId}/Cmts

Requires AU.

POST
Adds a comment to specified recipe.
>> *comment* The comment content

Rat/{ratId}/Lkes

Requires AU.
GET
Get all the likes associated with the ratId.
POST
Adds a like to specified recipe if AU does not have a like associated already.

Rat/{ratId}/Lkes/{lkeId}

DELETE
Deletes a like from a specified recipe. AU must own the like or admin.

Rat/{ratId}/Dlks

Requires AU.
GET
Get all the dislikes associated with the ratId.
POST
Adds a dislike to specified recipe if AU does not have a dislike associated already.

Rat/{ratId}/Dlks/{dlkId}

DELETE
Deletes a like from a specified recipe. AU must own the dislike or admin.

DB/

DELETE
Deletes everything from the database and recreates an admin user with the default settings. This is for testing purposes and requires admin AU.