Base de datos

El modelo de la base de datos deberá tener las siguientes entidades (Aquellas propiedades marcadas con asterisco deben ser obligatorias):

* [ ] Receta con las siguientes propiedades:
  + ID: \*
  + Nombre \*
  + Resumen del plato \*
  + Nivel de "comida saludable" (health score)
  + Paso a paso
* [ ] Tipo de dieta con las siguientes propiedades:
  + ID
  + Nombre

La relación entre ambas entidades debe ser de muchos a muchos ya que una receta puede ser parte de varios tipos de dieta en simultaneo y, a su vez, un tipo de dieta puede contener múltiples recetas distintas. Un ejemplo tomado de la API sería el Strawberry Mango Green Tea Limeade que es vegetariano, vegano y apto para celíacos, todo al mismo tiempo. Pero a su vez existen otras recetas para vegetarianos.

**IMPORTANTE**: Pensar como modelar los IDs de las recetas en la base de datos. Existen distintas formas correctas de hacerlo pero tener en cuenta que cuando hagamos click en alguna receta, esta puede provenir de la API o de la Base de Datos por lo que cuando muestre su detalle no debería haber ambigüedad en cual se debería mostrar. Por ejemplo si en la API la receta Strawberry Mango Green Tea Limeade tiene id = 1 y en nuestra base de datos creamos una nueva receta Medialunas de Manteca con id = 1, ver la forma de diferenciarlas cuando querramos acceder al detalle de la misma.

Backend

Se debe desarrollar un servidor en Node/Express con las siguientes rutas:

**IMPORTANTE**: No está permitido utilizar los filtrados, ordenamientos y paginados brindados por la API externa, todas estas funcionalidades tienen que implementarlas ustedes.

* [ ] **GET /recipes?name="..."**:
  + Obtener un listado de las recetas que contengan la palabra ingresada como query parameter
  + Si no existe ninguna receta mostrar un mensaje adecuado
* [ ] **GET /recipes/{idReceta}**:
  + Obtener el detalle de una receta en particular
  + Debe traer solo los datos pedidos en la ruta de detalle de receta
  + Incluir los tipos de dieta asociados
* [ ] **POST /recipes**:
  + Recibe los datos recolectados desde el formulario controlado de la ruta de creación de recetas por body
  + Crea una receta en la base de datos relacionada con sus tipos de dietas.
* [ ] **GET /diets**:
  + Obtener todos los tipos de dieta posibles
  + En una primera instancia, cuando no exista ninguno, deberán precargar la base de datos con los tipos de datos indicados por spoonacular [acá](https://spoonacular.com/food-api/docs" \l "Diets" \o "https://spoonacular.com/food-api/docs#Diets)

Testing

* [ ] Al menos tener un componente del frontend con sus tests respectivos
* [ ] Al menos tener una ruta del backend con sus tests respectivos
* [ ] Al menos tener un modelo de la base de datos con sus tests respectivos