JPA Assignment

Your assignment is to create a RecipeDatabase with the tools of:
Spring Boot, JPA/Hibernate, MySQL and H2.

Subjects:

- JPA / Hibernate
- Entity
- Spring Boot
- Dependency Injection
- Entity Relationships
- SpringData / EntityManager
- CRUD functionality with SpringData or EntityManager
- Read from database with help of Spring Data framework or queries.
- MySQL database
- H2 database in testing environment.
- Testning with database.

Requirements:

- All Entities and relations need to be in place.
- Implement Create, Read, Update, Delete and specified functionalities with Repositories or Dao classes.
- The following functionalities need to tested with JUnit.
    - Add and remove from collections in you Entities (e.g. List, Set etc.).
    - The methods you specified or implemented in Repositories or Dao classes.

Submit:

- Publish to GitHub and send the link to your teacher before or at deadline.

Entities

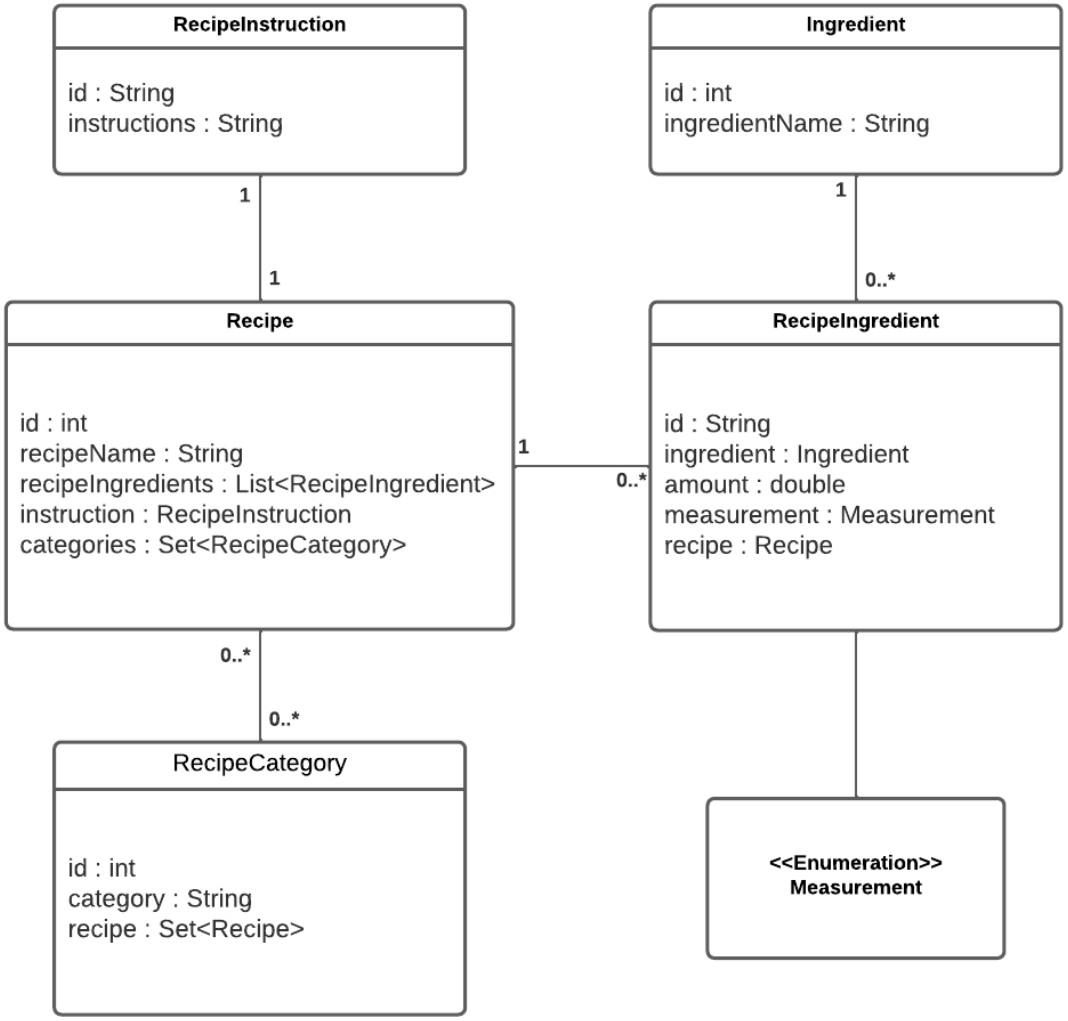
Create these Java files with the following:

1. **Class**: **`Ingredient`**
    a. Contains an id.
    b. Contains a unique ingredient.
2. **Enum**: **`Measurement`**
    a. Specify a couple of measurements and weights that you may find suitable for a recipe.
    e.g. TBSP, TSP, G, HG, KG, ML, CL, DL etc.
3. **Class**: **`RecipeIngredient`**
    a. Contains an id of type String. Generate the id as a UUID from Hibernate.
    b. Contains a reference to `Ingredient`.
    c. Contains a representation of a measured amount type double.
    d. Contains a `Measurement` that represent the unit.
    e. Contains a reference to the associated `Recipe`.
4. **Class**: **`RecipeCategory`**
    a. Contains an id.
    b. Contains a category of type String.
    c. Contains a collection of recipes associated to this `RecipeCategory`.
5. **Class**: **`Recipe`**
    a. Contains an id.
    b. Contains a recipe name of type String.
    c. Contains a collection of recipe ingredients.
    When you remove content from this collection make sure to implement automagical removal of this `RecipeIngredient`.
    d. Contains recipe instructions of type `RecipeInstruction`.
    e. Contains a collection of `RecipeCategory`.
6. **Class: `RecipeInstruction`**
    a. Contains an id.
    b. Contains recipe instructions of type String

Class Diagram on next page.

Class Diagram

These are the relationships in Java.



Note!
You can decide to change the id types to whatever type you want. String, int, long etc..

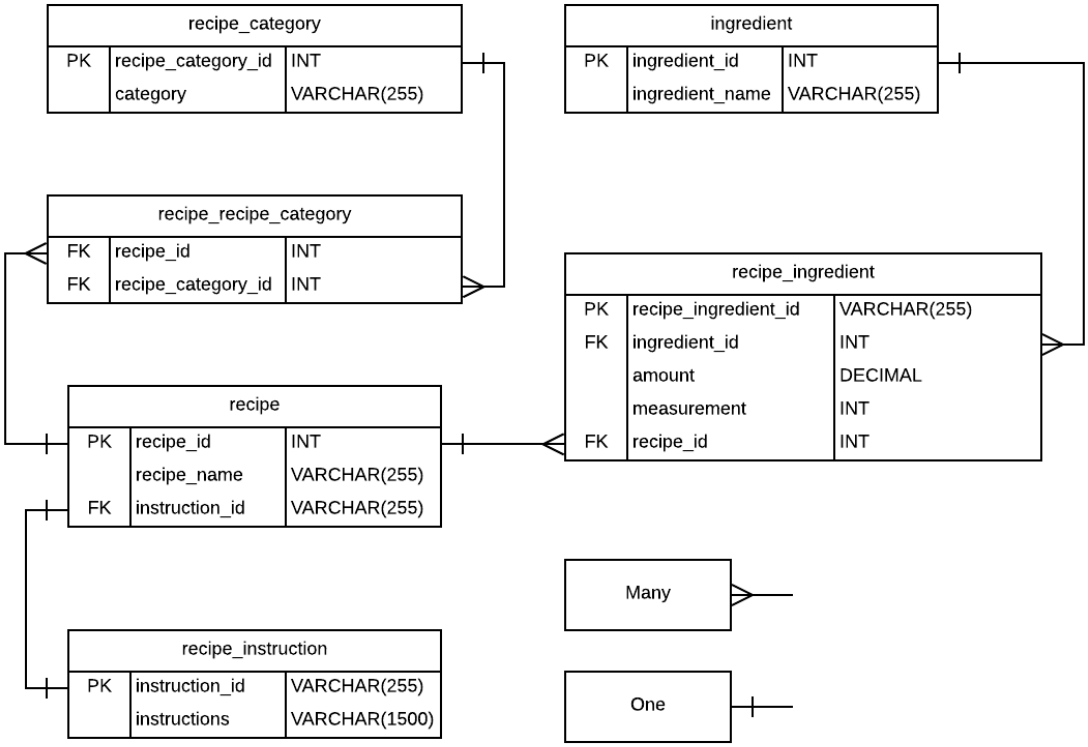
0..* *Read this as none to many.*
1 *Read this as just one.*

Entity Relationship diagram on next page.

Database Diagram

(Entity Relationship diagram)

These are the relationships between database tables.



Repository / Data Access Object

You have two options with accessing data.

1. Use Data Access Object classes connected with interfaces.
2. Use Spring Data and create repositories.

Implement CRUD operations and the following functionalities:

- Ingredient Repository/Dao:
    - Search for one ingredient object that matches exactly with sent in ingredient name.
    - Search for ingredients that contains parts of sent in ingredient name.
- Recipe Repository/Dao:
    - Search for recipes where recipe name contains specified String.
    - Search for all recipes that contains a specified ingredient name.
      e.g. potato, tomato, salt, etc.
    - Search for all recipes that belong to a specific recipe category.
      e.g. Chicken, Vegan, Celebration, Weekend etc.
    - Search for all recipes that match one or more categories.
      e.g. {”Spicy”,”Mexican”,”Weekend”}
- RecipeIngredient Repository / Dao:
    - Only specify CRUD functionality.
      (Other functionalities will be handled by `Recipe`.)
- RecipeInstruction Repository / Dao:
    - Only specify CRUD functionality.

Don’t forget to test the methods you created.