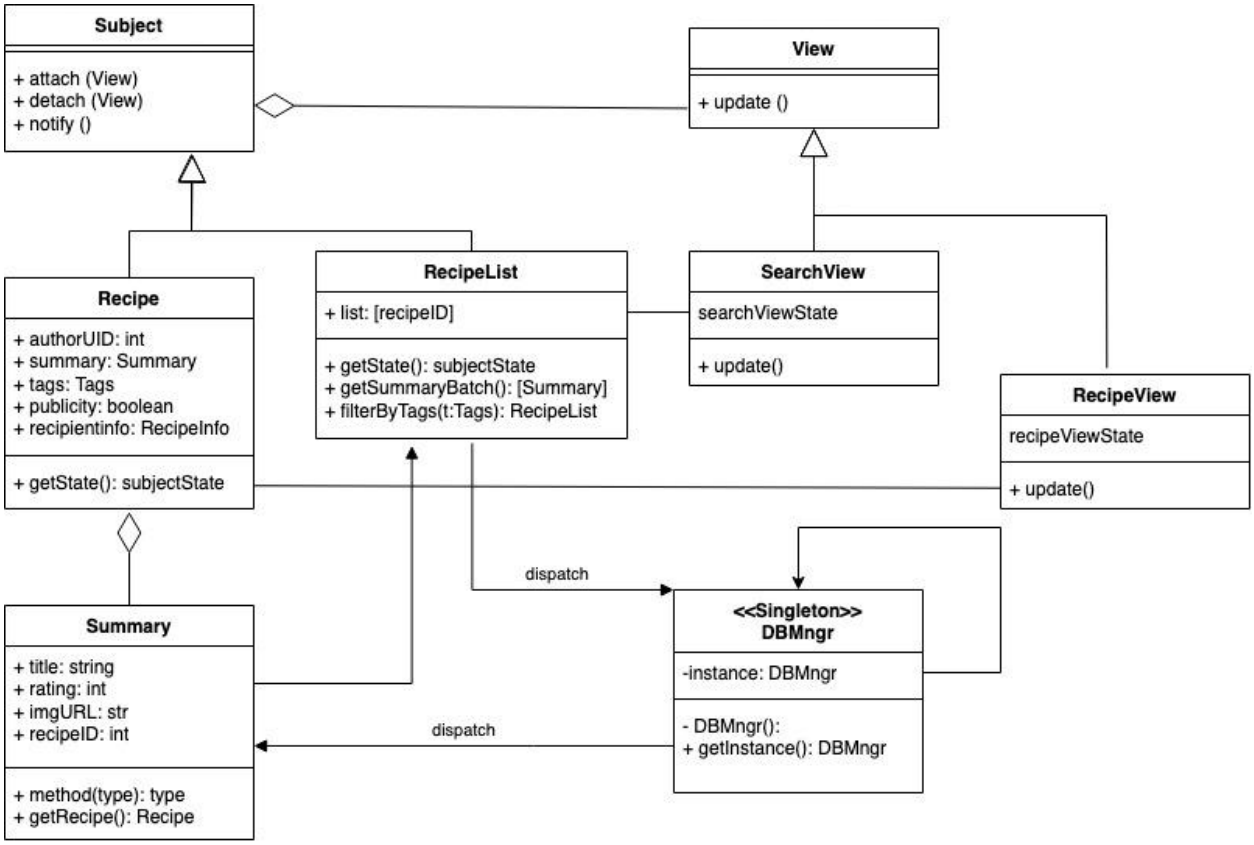
Design Class Diagram:



Singleton Pattern Implementation

We intend to implement our database management class as a singleton pattern. Our program will utilize the database management class anytime we need to query or update our database of recipes. We only want to have one globally accessible instance of this database management class to avoid data corruption or race conditions in our database.

Observer Pattern Implementation

All of the storage classes, or ‘subjects’ of the Cooking Companion will need a common set of methods to interact with the frontend, for something this simple we can use inheritance to impart these common methods and attributes to all our subjects from an abstract ‘subject’ class. The Observer pattern suits this end well, and will also allow us to present the same concrete class in a variety of ways.

Recipe and RecipeList are concrete subjects, and extend the abstract subject, inheriting the attach(View) and detach(View) methods which sets the state of the given view with the information in the subject.

SearchView and RecipeView are observers for RecipeList and Recipe respectively, and extend the abstract View class, inheriting an update() method that rerenders the state of the View in question.