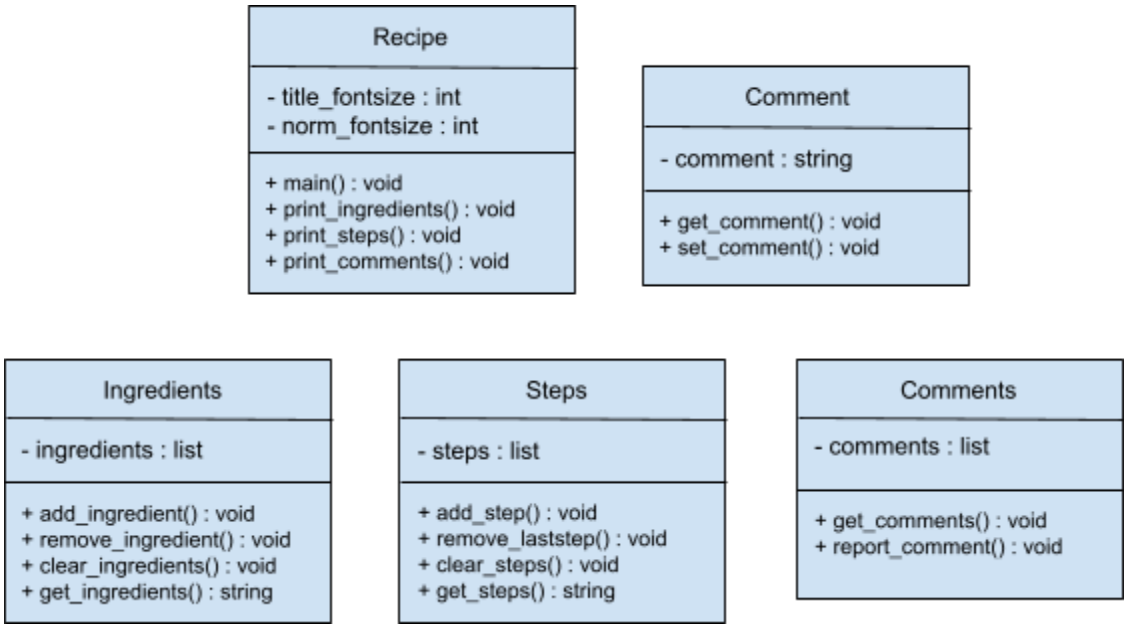
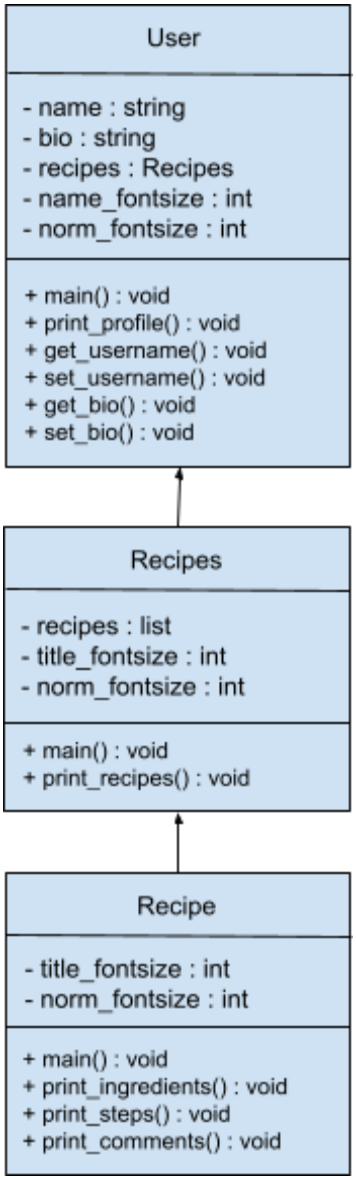
Detailed Design Model and Design Patterns

I. UML design class diagrams for detailed design with their attributes, associations, and methods that would be representative of the system under development and the user stories.

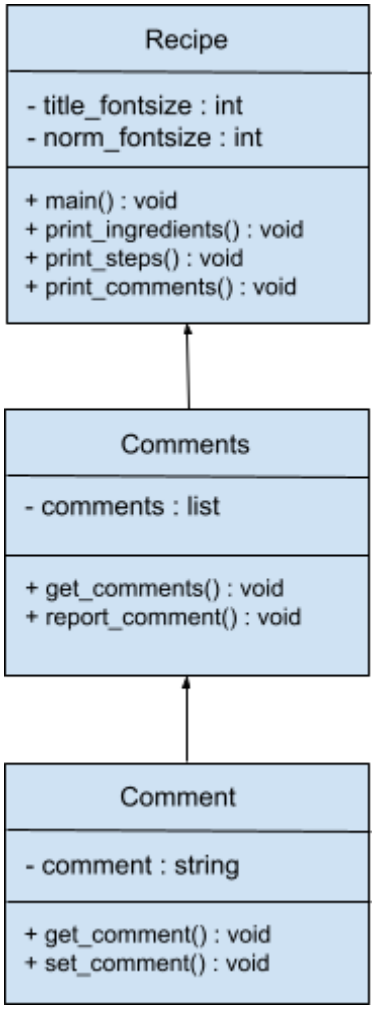



User
- name : string
- bio : string
- recipes : Recipes
- name_fontsize : int
- norm_fontsize : int
+ main() : void
+ print_profile() : void
+ get_username() : void
+ set_username() : void
+ get_bio() : void
+ set_bio() : void
Recipes
- recipes : list
- title_fontsize : int
- norm_fontsize : int
+ main() : void
+ print_recipes() : void
Recipe
- title_fontsize : int
- norm_fontsize : int
+ main() : void
+ print_ingredients() : void
+ print_steps() : void
+ print_comments() : void

II. UML class diagrams for design patterns with their detailed descriptions, attributes, methods, associations, etc.

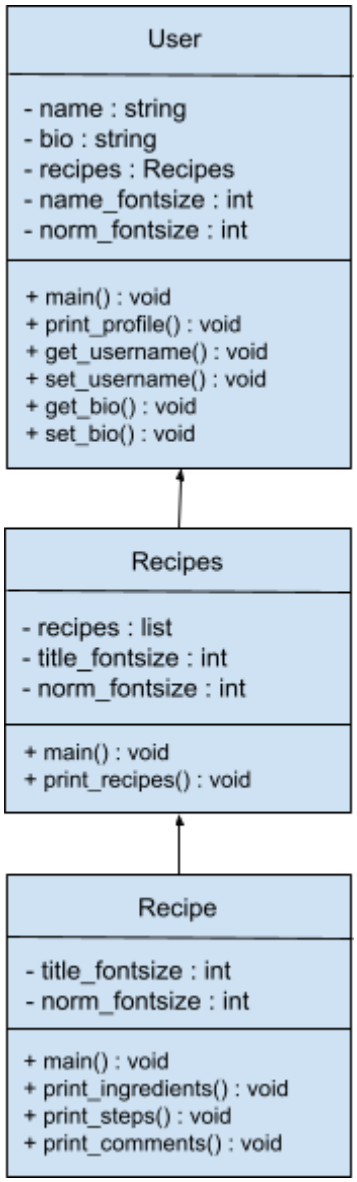
Design Pattern: Composite

The composite design pattern is appropriate for this case because, in the case of a recipe, we want to interact with a collection of comments in a uniform way, without thinking about them as such. This is exactly the use case for the composite design pattern - treating collections of objects simultaneously in a uniform way.



Design Pattern: Composite

The composite design pattern is appropriate for this case for similar reasons to the previous case. A user may want to interact with a collection of recipes in a uniform way, without thinking about them as such. This is again exactly the use case for the composite design pattern - treating collections of objects simultaneously in a uniform way.



Design Pattern: Template