Architecture Design

for

Brew Day!

Version 1.0 approved

Prepared by Yidong CHEN, Shihan XU, Zeyu WANG, Ning DING

Babbage

April 2nd 2019

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Yidong CHEN, Shihan XU, Zeyu WANG, Ning DING	April 2nd 2019	Initial version	1.0

1. Overview

1.1 Project description

The purpose of this project is to develop a software which provides the brewers with recommended recipes based on the ingredients they have. This software also enable brewers to manage their own recipes with notes.

1.2 References

1.N. Ding et al, "SRS1.4", Zhuhai: 2019.

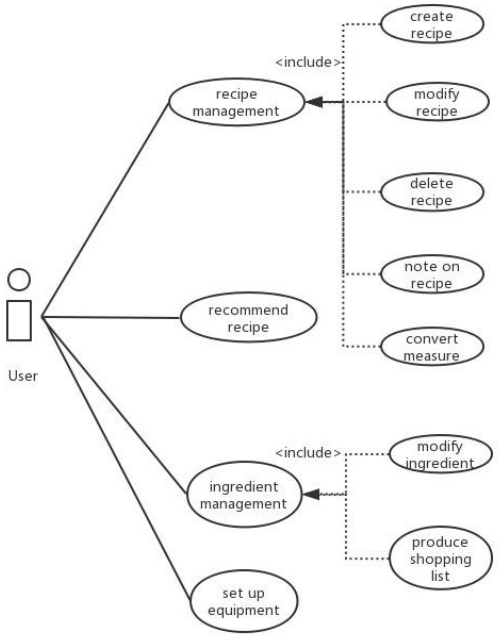
1.3 Design purpose

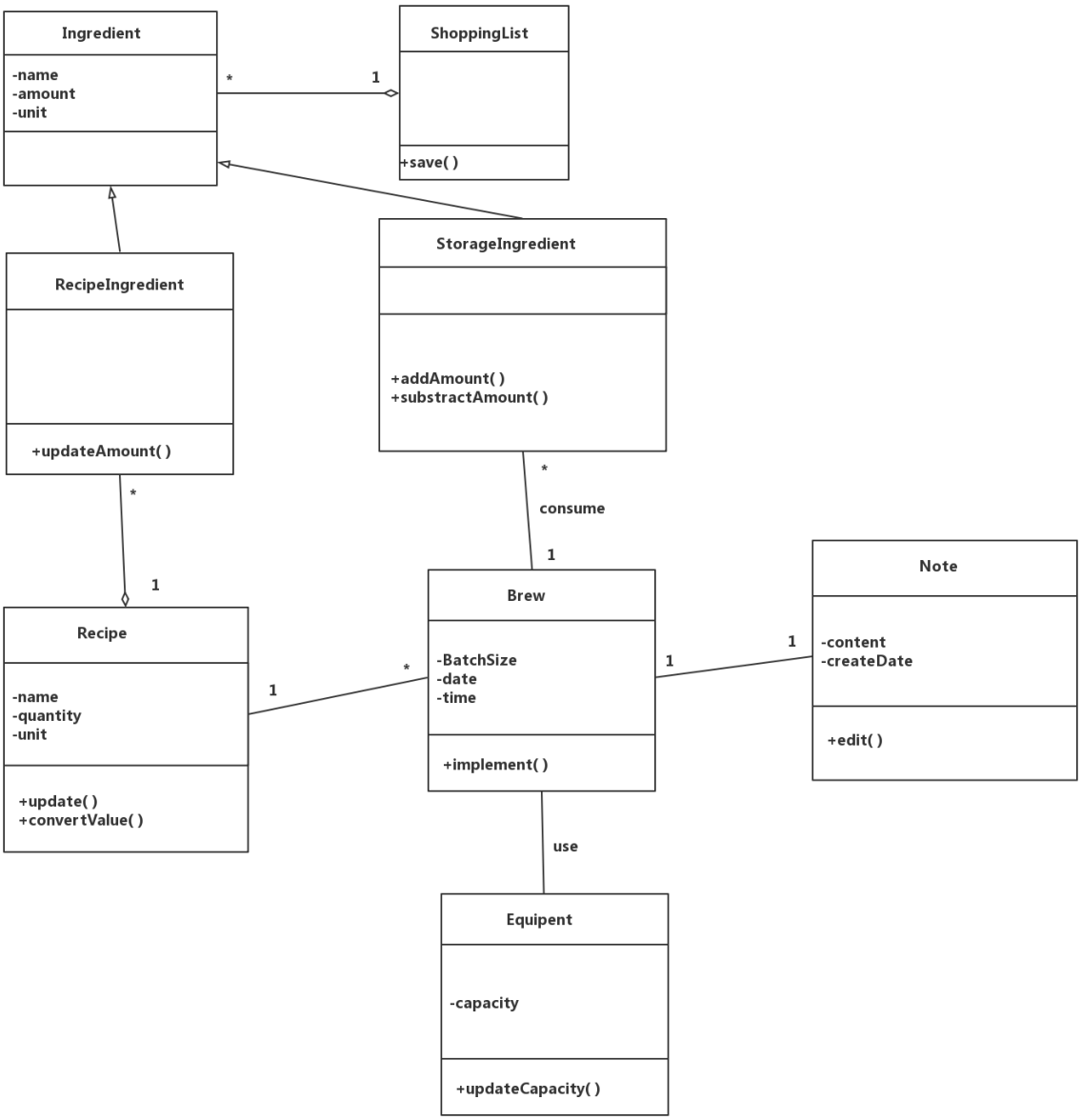
The purpose of the design of this software is to construct a concise, efficient and useful structure so that it is user friendly. The advantages of this design are as follows:
1. easy for the computer beginners to use
2. provide recommendation recipes for the users

The whole software is divided into several subsystems for designing, so that the system can be done by several group of people concurrently.

2. Overall description

2.1 Use case diagram and class diagram



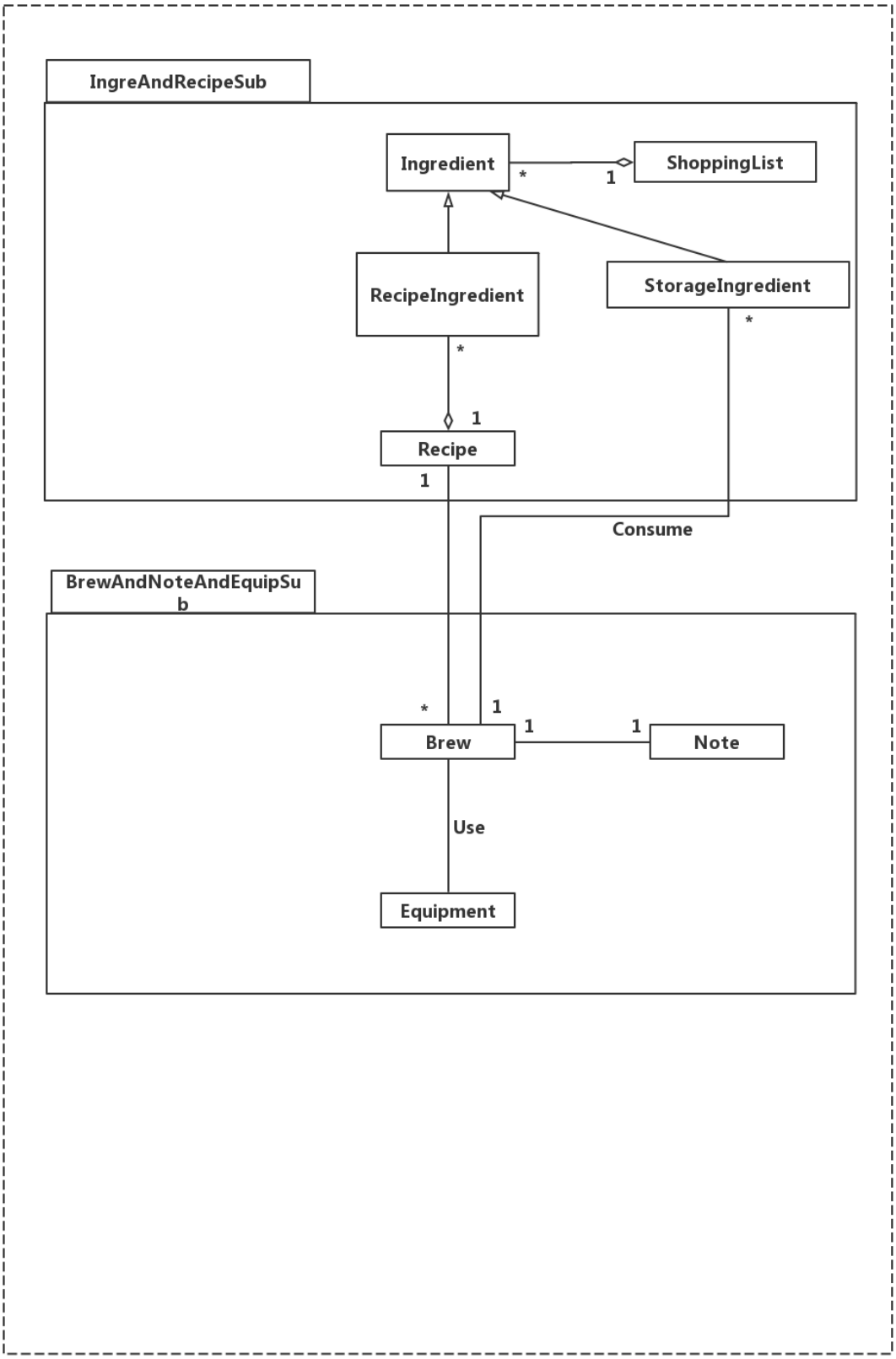


2.2 Design model

MVC and UML with three layers so that we can design them concretely and properly.

2.3 System architecture

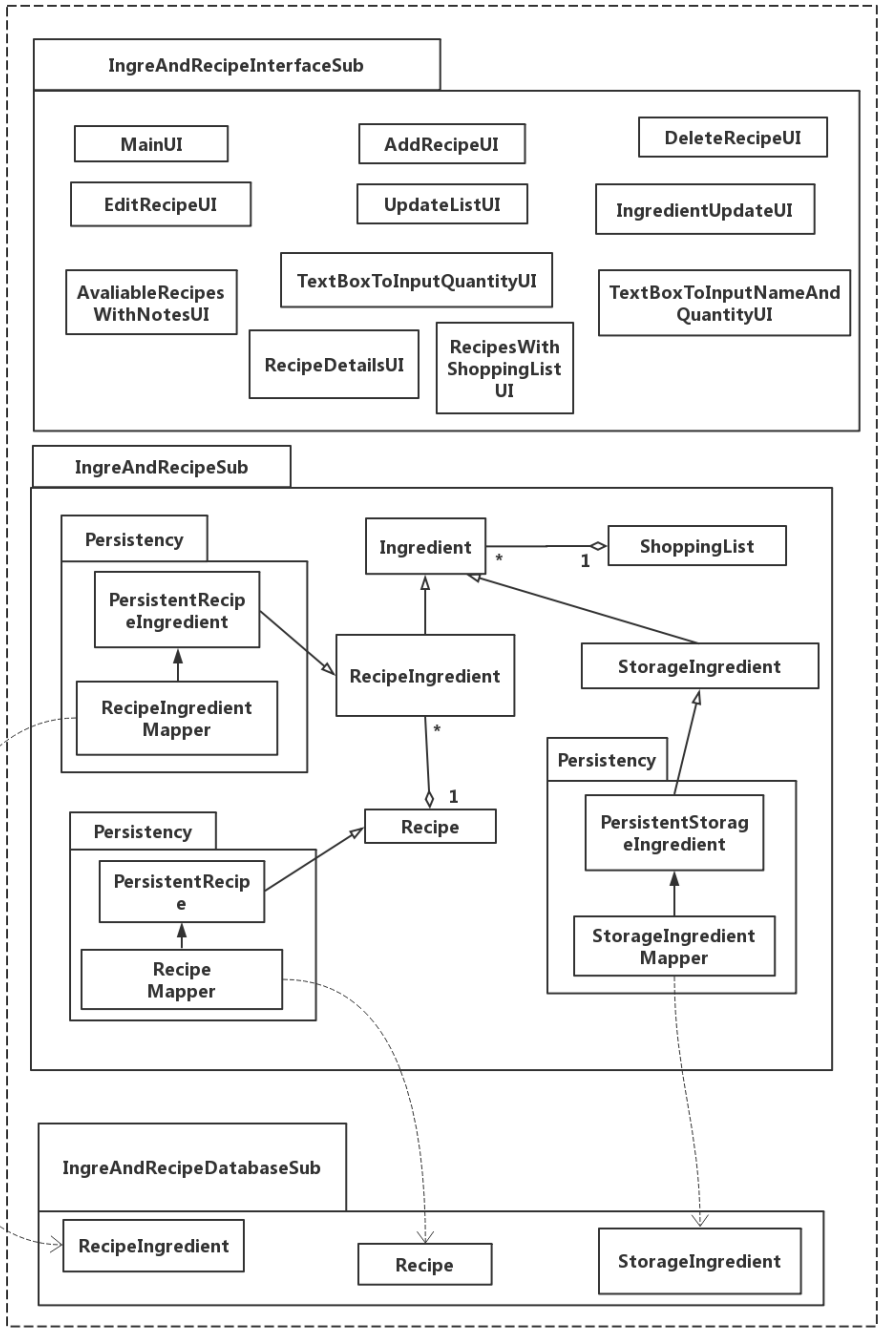
This sub-system illustrates the connection between the sub-system "IngredientAndRecipe" and "BrewAndNoteAndEquipment". Inside each sub-system, the diagram also demonstrates the relationship between each component. Necessary quantitative relations are also given in the diagram.


IngreAndRecipeSub
Ingredient
ShoppingList
*
1
RecipeIngredient
StorageIngredient
*
*
1
Recipe
1
Consume
BrewAndNoteAndEquipSub
*
1
1
1
Brew
Note
Use
Equipment

3. System architecture

3.1 IngreAndRecipeSubsystem

There are three layers in this subsystem: IngreAndRecipeInterfaceSub, IngreAndRecipeSub and IngreAndDatabaseSub. There are five entities in this subsystem: Ingredient, RecipeIngredient, Recipe, Shoppinglist and StorageIngredient.



3.1.1 Description

For the first layer, IngreAndRecipeInterfaceSub, it contains all the interfaces in this subsystem.

For the second layer, IngreAndRecipeSub, it contains all the entities and their relationships. The Ingredient is an abstract class used for specify the format of two child ingredient classes, RecipeIngredient and StorageIngredient. RecipeIngredient stores the ingredient information in particular recipe, StorageIngredient stores the ingredient quantity reserved in brewer's warehouse. ShoppingList entity represents the ingredients list provided for brewer to buy, so it is composed by Ingredient. Finally, the Recipe entity represents the recipes.

The third layer IngreAndRecipeDatabaseSub represents the storage of this subsystem, RecipeIngredient maps with RecipeIngredient mapper, Recipe maps with Recipe mapper, Storage maps with StorageIngredient mapper.

3.1.2 Database

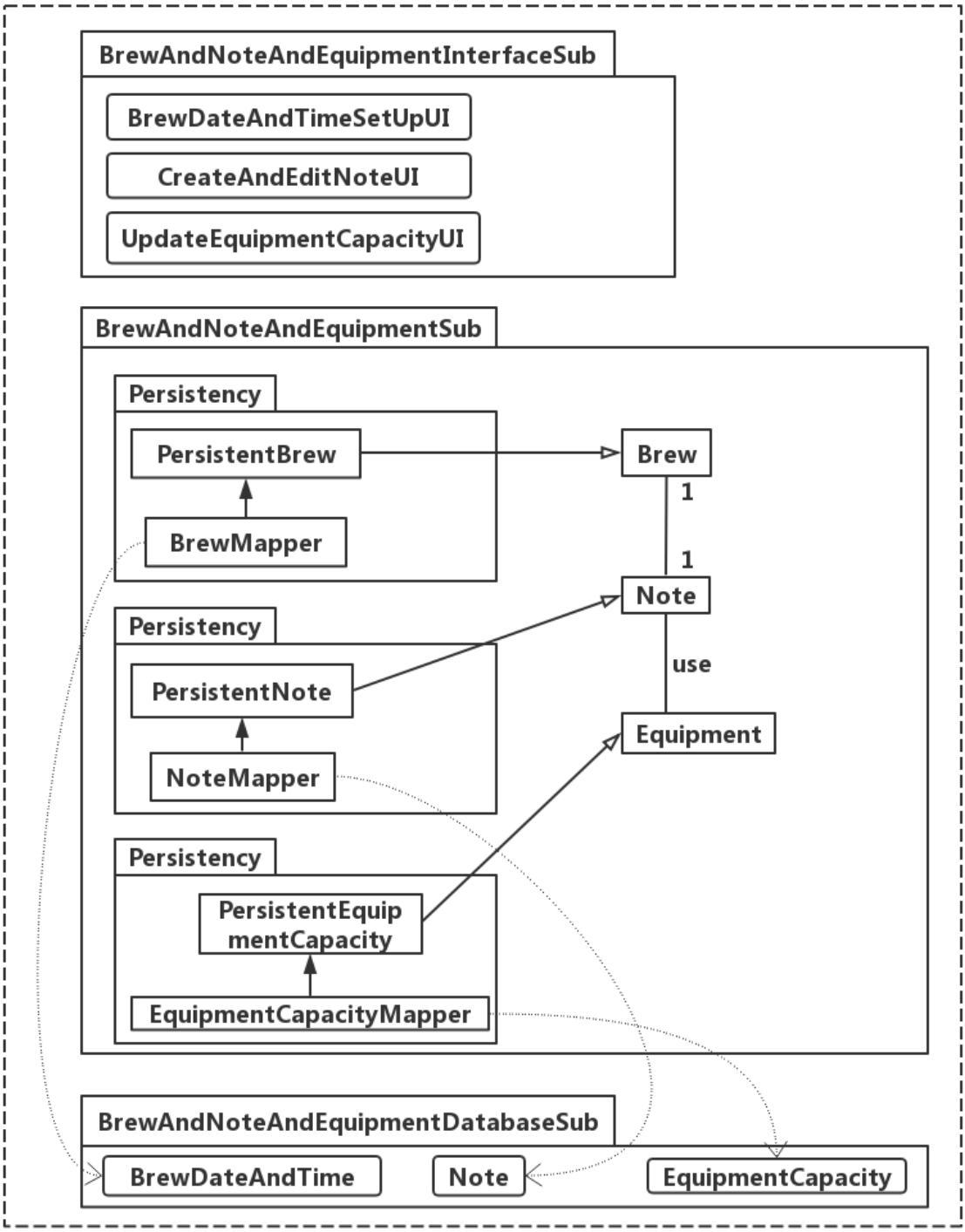
<Describe the tables in database if exists>

RecipeIngredient				
id	name	amount	unit	recipe (for Recipe association)
1	milk	200	ml	beer1
2	milk	150	ml	beer2
3	barley	40	g	beer1

Recipe		
name	quantity	uint
beer1	500	ml
beer2	600	ml
beer3	1	L

StorgeIngredient		
name	amount	unit
milk	1000	ml
barley	500	ml
hops	50	g

3.2 BrewAndNoteAndEquipmentSubsystem



3.2.1 Description

In this subsystem, there are three components (classes): Brew, Note and Equipment.

In the Brew class, there are three attributes: batchSize, date and time. Batchsize is the total amount of brew the brewer produces. Date and time records the specific time of the brewery production.

In the Note class, there are two attributes content and createDate. The content describes the process of the brew, and createDate notes down the date when the brew occurs. The edit method allows the brewers edit on the notes.

In the Equipment class, there is only one attribute called “capacity”. This attribute indicates the maximum number of the liquor the equipment can hold. The updateCapacity method allows the brewers to modify the capacity according to the actual situation, e.g: the user have bought a bigger fermentation tank, so its capacity needs to be updated.

As for the relationship, the Brew Class uses the Equipment class so there is an association line between them. For each action of brew, there is a corresponding note with it. Thus the Brew class and the Note class are of one-to-one relationship.

3.2.2 Database

<Describe the tables in database if exists> P19

Brew		
Batch Size	Date	Time
Beer1	2019.04.01	19:45
Beer2	2019.04.02	18:31
Beer3	2019.03.31	12:16

Note	
Content	Create Date
Content1	2019.04.01
Content2	2019.04.02
Content3	2019.03.31

Equipment
Capacity
1000ml

4. Assessment

4.1 Stability

Our architecture is very stable.

We divide the whole system into two parts. There are only two association between them: one is “consume” association between class StorageIngredient and Brew, another one is one-to-many relationship between class Recipe and Brew. So the coupling is relatively low.

For the part IngreAndRecipeInterfaceSub, there are only Inheritance and Aggregation relationship, so they connected tightly. For the part BrewAndNoteAndEquipSub, class Note, Equipment and Brew are connected logically, because the note can only be taken through one brew and equipment limit maximum amount of one brew. So the cohesion is high.

4.2 Reusability

The reusability is medium.

The part IngreAndRecipeInterfaceSub has higher reusability than the part BrewAndNoteAndEquipSub. Any software which want to do some homemaking staff (like cakes, ice-cream, hotdog…) could use our IngreAndRecipeInterfaceSub part, because all these staff needs recipes and ingredients, probably need shopping list and a record of storage ingredient. So our IngreAndRecipeInterfaceSub part can be used.

The reusability of BrewAndNoteAndEquipSub is relatively low. Because this subsystem is designed for “Brew Day!” contains several features. It’s not likely that other software will have same features as ours, especially the “Brew” class.

4.3 Scalability

The scalability is relatively high. Because the main feature of our program is to determine what to brew today, if more features are added, they are more possibly associated with class Brew. New features only affects subsystem BrewAndNoteAndEquipSub, and they won’t make any changes to subsystem IngreAndRecipeInterfaceSub.

If the new feature is added to subsystem IngreAndRecipeInterfaceSub, it will at most affect the class Brew in BrewAndNoteAndEquipSub, and it won’t make a lot of changes to other subsystem too.

5. More considerations

If you want to know more about user interfaces, please refer to SRS 4.1.

6. Appendix

N/A