Architecture Design

for

Brew Day!

Version 1.0 approved

Prepared by Zhenghao Wu, Anqin Zha, Renjie Deng and Ruichao Zhong

Dijkstra

Tuesday, April 9, 2019

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Zhenghao Wu, Anqin Zha, Renjie Deng and Ruichao Zhong	2019-04-02	The first version of the Architecture Design	1.0

1. Overview

1.1 Project description

This project is to develop a desktop-base for home brewer to brew beers. Home brewer can use the product to maintain recipe, ingredients and equipment information, he can also receive recommendation from this software and take notes on each brewing.

1.2 References

Z. H. Wu, et al., “Software Requirements Specification for Brew Day!” Apr. 2nd, 2019.

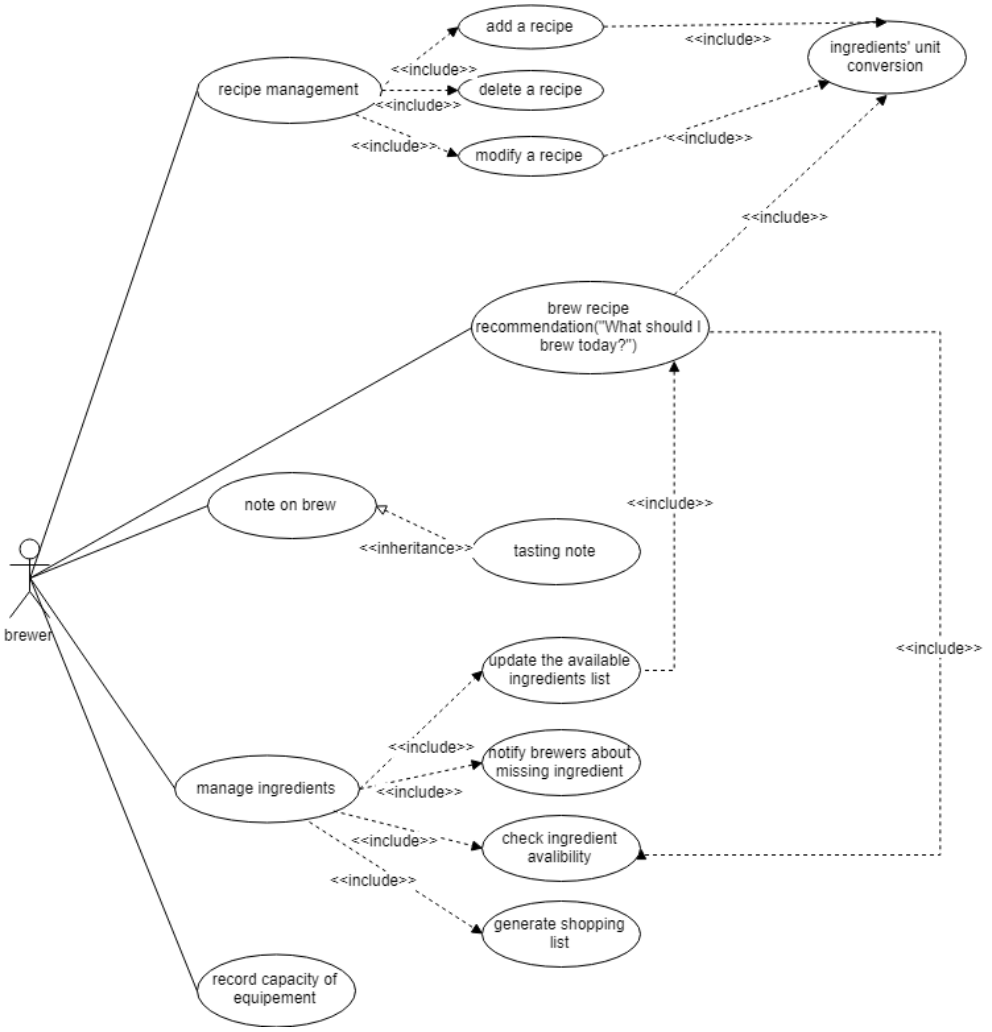
1.3 Design purpose

This design is given to divide system into several subsystem with loose coupling and high cohesion, thus the large system can be separate into different task that each single programmer can do their jobs independently.

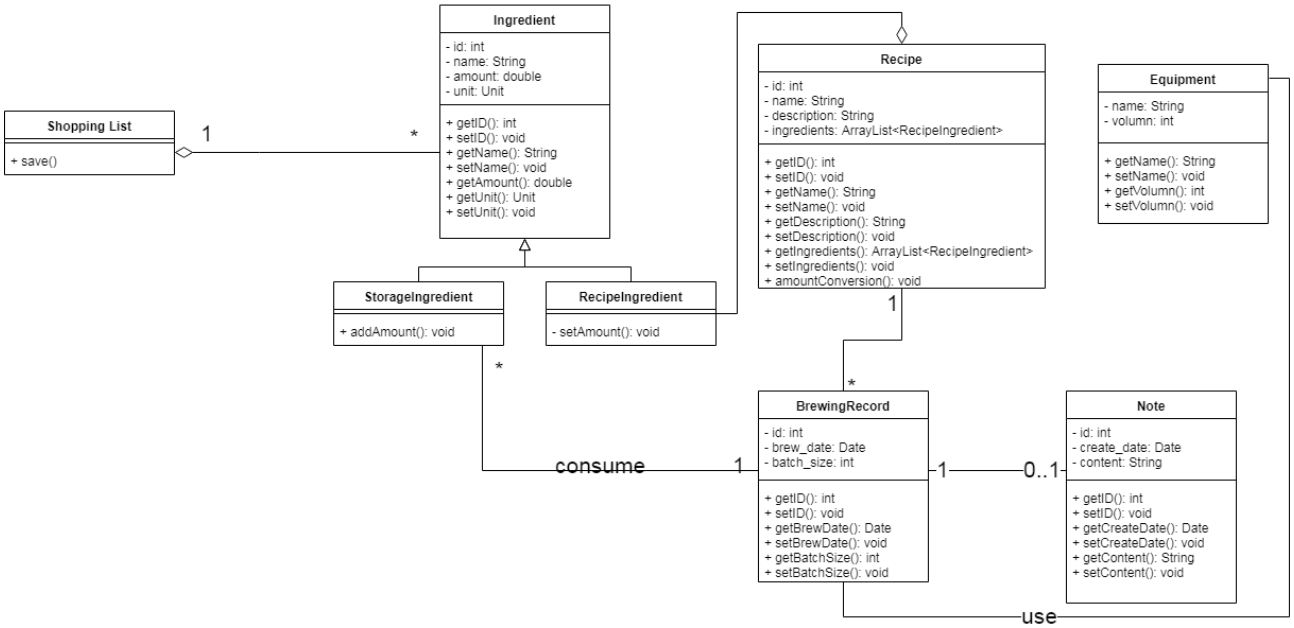
2. Overall description

2.1 Use case diagram and class diagram

The Use Case Diagram shown below:



The class diagram shown below:

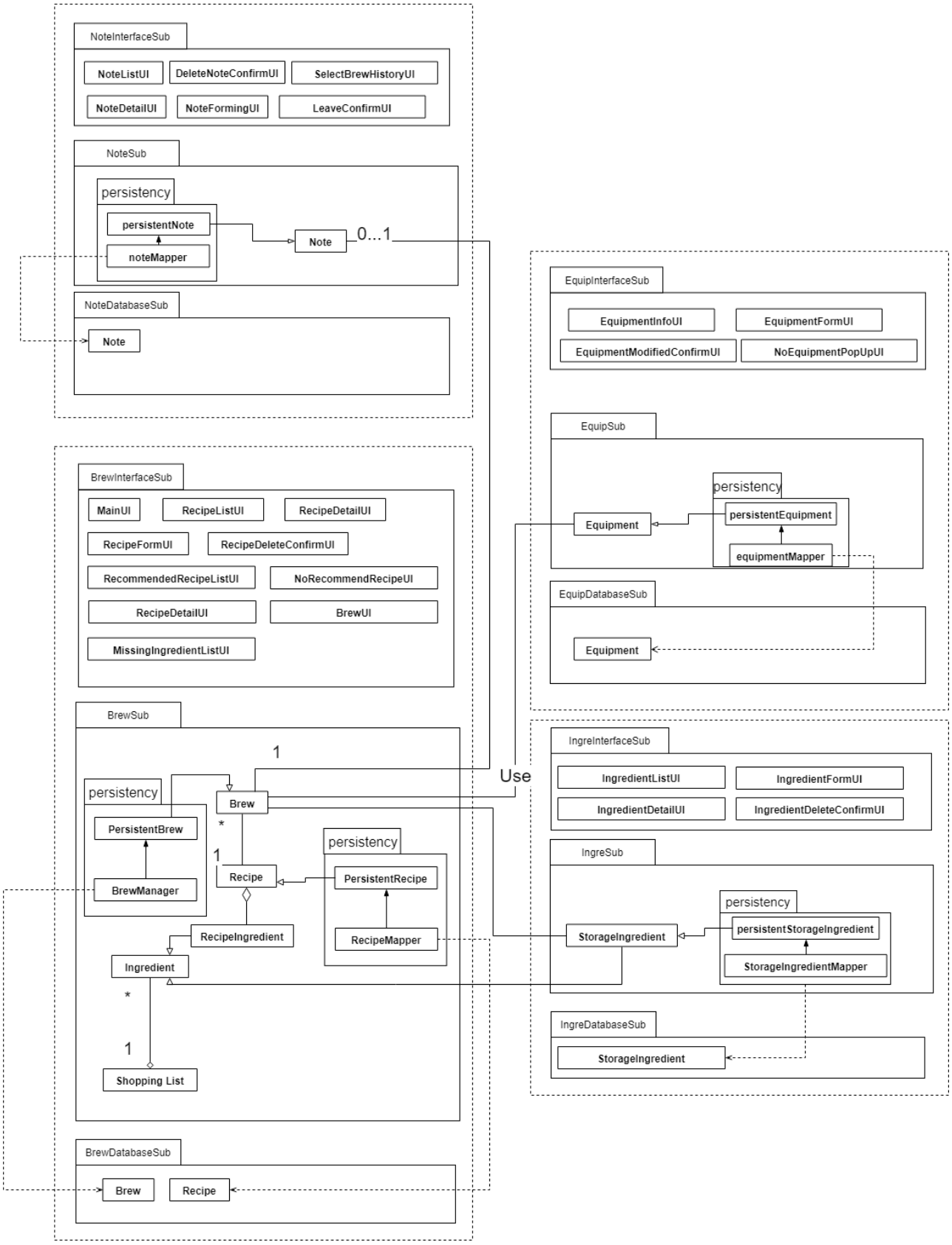


2.2 Design model

This system is using MVC three tier architecture. This architecture model is used in there is because this model can help to divide the system in to different part (UI, application logic, storage) and help different developer to develop it separately.

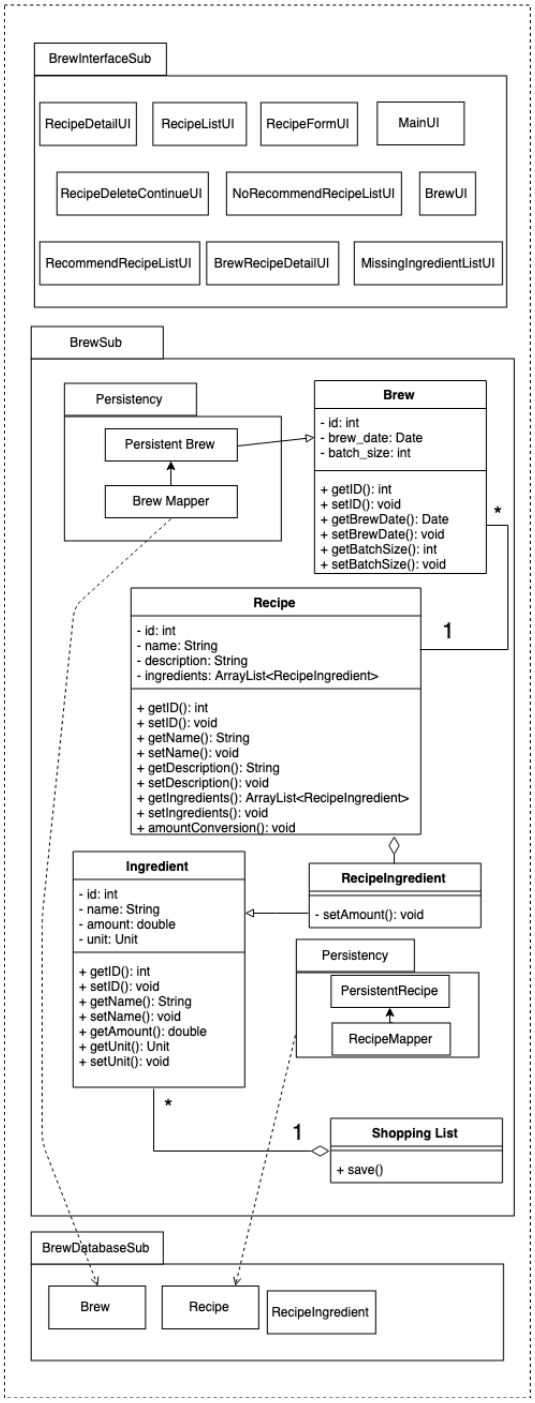
2.3 System architecture

This system is divided into four subsystems: Brew Subsystem, Ingredient Subsystem, Note Subsystem, and Equipment Subsystem.
Diagram that indicate the relationship between subsystems is shown below:


NoteInterfaceSub
NoteListUI
DeleteNoteConfirmUI
SelectBrewHistoryUI
NoteDetailUI
NoteFormingUI
LeaveConfirmUI
NoteSub
persistency
persistentNote
noteMapper
Note
0...1
NoteDatabaseSub
Note
EquipInterfaceSub
EquipmentInfoUI
EquipmentFormUI
EquipmentModifiedConfirmUI
NoEquipmentPopUpUI
EquipSub
persistency
Equipment
persistentEquipment
equipmentMapper
EquipDatabaseSub
Equipment
BrewInterfaceSub
MainUI
RecipeListUI
RecipeDetailUI
RecipeFormUI
RecipeDeleteConfirmUI
RecommendedRecipeListUI
NoRecommendRecipeUI
RecipeDetailUI
BrewUI
MissingIngredientListUI
BrewSub
1
Use
persistency
PersistentBrew
BrewManager
Brew
*
1
Recipe
persistency
PersistentRecipe
RecipeMapper
RecipeIngredient
Ingredient
*
1
Shopping List
IngreInterfaceSub
IngredientListUI
IngredientFormUI
IngredientDetailUI
IngredientDeleteConfirmUI
IngreSub
persistency
StorageIngredient
persistentStorageIngredient
StorageIngredientMapper
IngreDatabaseSub
StorageIngredient
BrewDatabaseSub
Brew
Recipe

3. System architecture

3.1 Brew Subsystem



3.1.1 Description

The brew subsystem is consisting of three part: BrewInterfaceSub is for UI, it consisit of all boundary objects including: MainUI for the homepage of the software; RecipeDetailUI, RecipeListUI, RecipeFormUI, RecipeDeleteContinueUI for Recipe use case; BrewUI, RecommendRecipeListUI, NoRecommendRecipeListUI, BrewRecipeDetailUI, MissingIngredientListUI for recommend a brew use case. BrewSub is for application logic, It handle control and entity objects. Brew, Recipe, Ingredient, RecipeIngredient, ShoppingList are class in this part. And Brew and Recipe are the persistent data. BrewDatabaseSub is for storage, it storage, retrieval, and query of persistent data.

3.1.2 Database

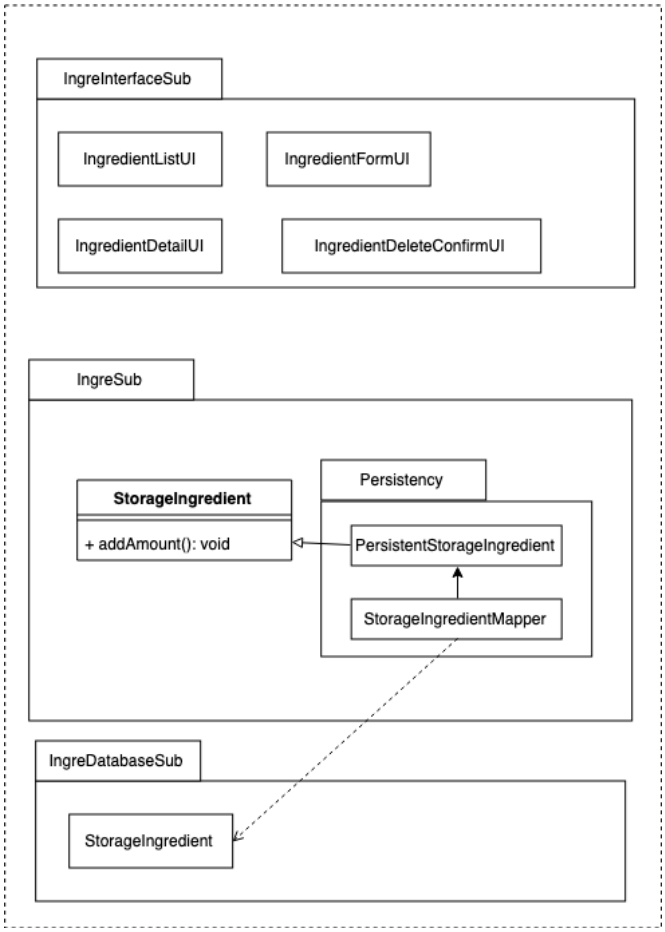
There are three database tables in the brew subsystem: Brew, Recipe, Ingredient in Recipe. Brew table is uses to record brew records. Recipe table is uses to record recipes. Ingredient in Recipe is records different kinds of and amount of ingredient in different recipe. The fields for each table are list below:

Brew				
Brew ID	Brew Date	Batch size	Recipe ID	Note ID

Recipe		
Recipe ID	Name	Description

Ingredient in Recipe			
Recipe ID	Ingredient ID	Amount	unit

3.2 Ingredients Subsystem



3.2.1 Description

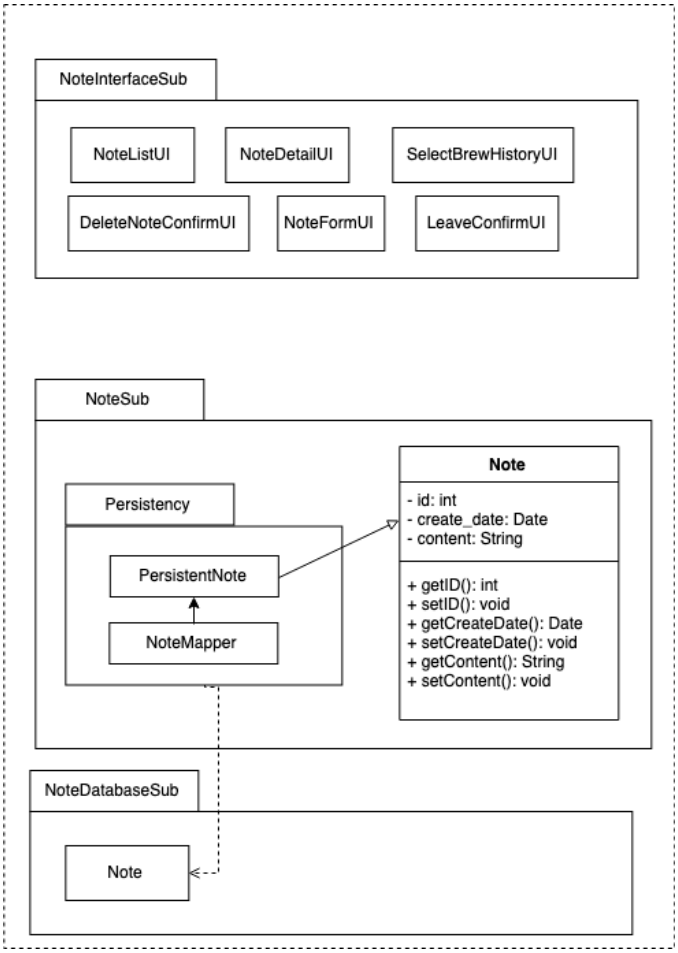
The system contains 3 parts, Interface part, application logic part and database part.
The interface part of the system contains Ingredient List, Ingredient Form, Ingredient Detail and confirm delete window. The ingredient list lists all the information about ingredients in the warehouse including the name of ingredients and amount of ingredients. Ingredient form is a form that you can add a new ingredient to the warehouse or modify the amount of ingredients in the warehouse. Ingredient detail page show you the detail information of an ingredient including name, amount and unit of the ingredient. IngredientConfirmDelete page is a page that ensure you would not delete the ingredient by accident.
The data of StorageIngredient class is persistent, which will be store in database.
The application logic part of the system contains StorageIngredient. With StorageIngredient part, you can manage the amount of ingredients in the warehouse.
The database part contains a table called StorageIngredient, which store the data of the amount of ingredients in the warehouse.

3.2.2 Database

There is one database table in the note subsystem: Note. Note table is uses to record brewer's note for a brewing record. The fields for the table is list below:

Ingredient			
Ingredient ID	Name	Amount	Unit

3.3 Note Subsystem



3.3.1 Description

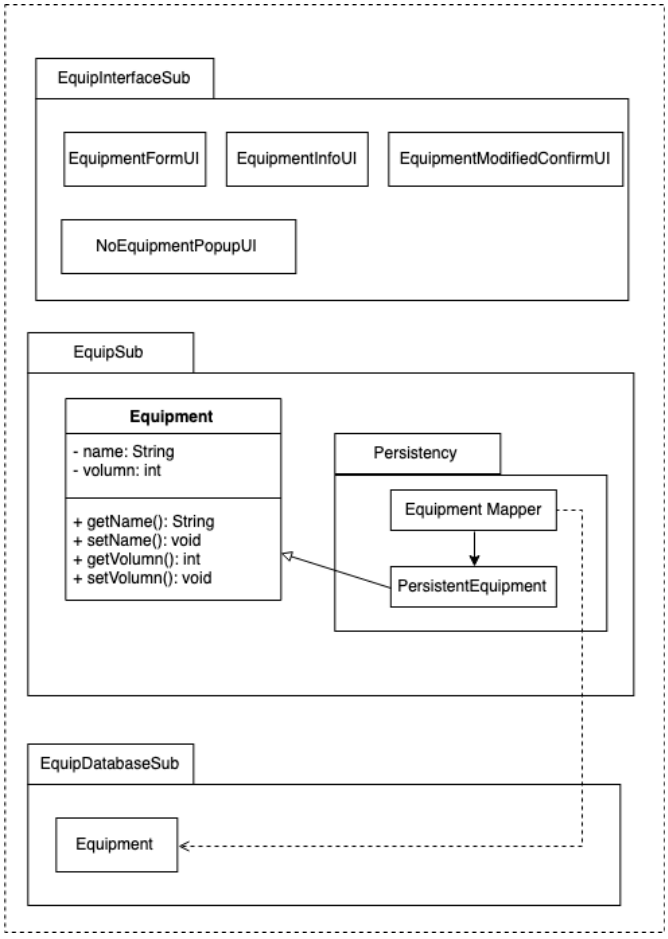
The subsystem contains three part: Interface part, application logic and storage part. Among them, all component in interface part are for displaying the note list and brewing history list, the content of note, inputting interface and some confirm window to deal with interaction with users. For the application logic, the component "note" is mainly for setting and getting the ID, date and content of the notes, and the data of "note" is persistent, and then it will store its persistent data to the component "note" in storage part, which is the third part of this subsystem through the component "persistent Note". The "note" in storage database realize the storage, retrieval, and query of the data it received.

3.3.2 Database

There is one database table in the note subsystem: Note. Note table is uses to record brewer's note for a brewing record. The fields for the table is list below:

Note			
Note ID	Create date	Content	Brew ID

3.4 Equipment Subsystem



3.4.1 Description

The subsystem contains three part: Interface, application logic and storage. Among them, all component in interface part are for displaying Equipment Form, Equipment information and some confirm information. For the application logic part, the data of the component "equipment" is persistent, the component itself is mainly used for setting and getting the name and volume of the equipment, once the setting is done, it will store data to "Equipment" in the storage part, which is the third part of this subsystem, through the

component “persistent Equipment”. The “Equipment” in storage database realize the storage, retrieval, and query of the data it received.

3.4.2 Database

There is one database table in the equipment subsystem: Equipment. Equipment table is uses to record brewer’s equipment for brewing. The fields for the table is list below:

Equipment		
Equipment ID	Name	Volume

4. Assessment

4.1 Stability

The system based on this architecture is stable, because this system has minimized the associations which will cross subsystem boundaries, that is, the coupling of this system is loose and the cohesion of this system is high, therefore, if some changes are made to certain components, we need only to change very few objects’ interface.

4.2 Reusability

The component of this system can easily be used in other place without changes, because three out of four subsystems can implement their jobs individually: taking notes, maintain equipment and maintain ingredients. They will only contact to brewing subsystem when they are requested data from it. What is more, the brewing subsystem needs only data instead of function of other subsystem, which means all components of this system have high cohesion.

4.3 Scalability

This system is easy to extend with this architecture because this system adapted divide and conquer strategy, which solve the complexity of the system. In other words, each component has the clear input and output requirement, thus it is easy to expand the architecture.

5. Alternative design (optional)

N/A

6. More considerations

For more detail about the UI design, please refer to the Software Requirement Specification 4.1 – User Interface

7. Appendix

N/A