Detailed Design

for

Brew Day

Version 2.0 approved

Prepared by Yuanzhen Lin, Xiaochen Xu, Ziqi Wang, Weifeng Huang

Gray

5/23/2019

Table of Contents

Table of Contents ii

Revision History ii

1. Overview 1

1.1 Project description 1

1.2 References 1

1.3 Design purpose 1

2. Overall description 2

2.1 Class diagram 2

2.2 Refinements 2

3. Detailed design 4

3.1 Class diagram 4

3.2 Classes 5

3.2.1 Brew 5

3.2.2 Equipment 5

3.2.3 Ingredient 6

3.2.4 Note 7

3.2.5 Recipe 8

3.2.6 RecipeIngredient 9

3.2.7 StorageIngredient 9

4. Alternative detailed design (Optional) 10

5. More considerations 10

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| DD\_Gray\_1.0 | 4.9.2019 | Initial Version | 1.0 |
| DD\_Gray\_2.0 | 5.23.2019 | Second Version | 2.0 |

# Overview

## Project description

The project is to create a software for the user to brew the beer. In this software, one could maintain the amount of ingredient for brewing the beer, like adding, subtracting.

Also, the software allows the user to maintain the recipe, equipment.

User could write note for each brewing in the software.

Finally, the software could recommend the recipe for the user with the known condition.

## References

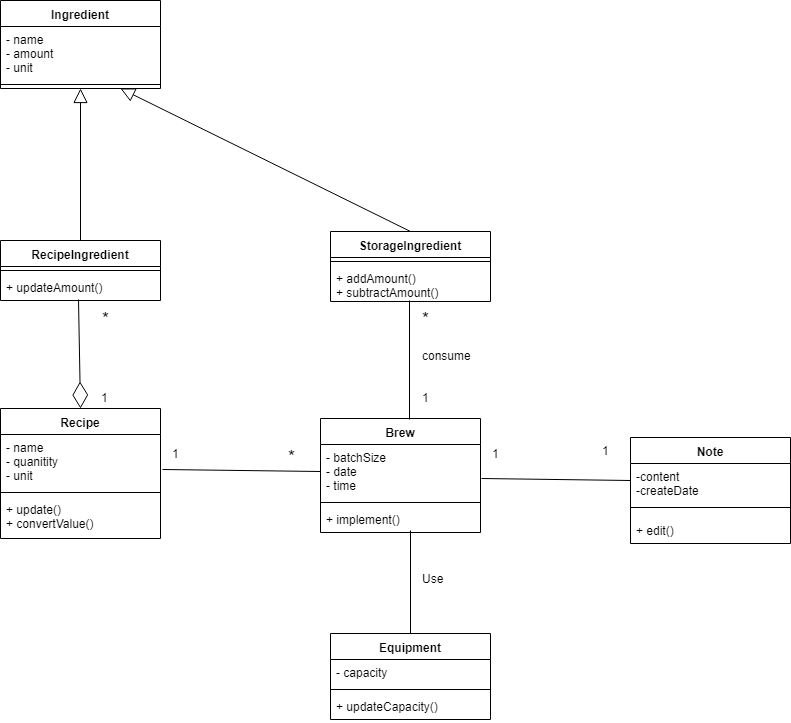
This document is referred from the software requirement specification document and the architecture design document from group Gray.

## Design purpose

The document is designed to restructure the design for implementation and provide more detailed information as well as more comprehensive interface design.

# Overall description

## Class diagram



## Refinements

We have set the type for each of the class, for example the ‘int’ type or the ‘String’ type. Also, we have set the visibility and the signature for each of the class.

We also specify the qualifier to reduce the multiplicity between the RecipeIngredient class and Recipe class, Brew and Recipe class as well as the Brew and StorageIngredient class.

For ‘Ingredient class’, we have updated the type for each attribute and the operation. Also, we add more operation to get and modify the corresponding attribute.

For the ‘RecipeIngredient class’, we have set the parameter(signature) to the operation and new attribute to the class so as to have further perform.

For the ‘StorageIngredient class’, we have set the parameter type and return type for the operation to get more detailed in performance.

For ‘Recipe class’, we add an attribute and a list of operation to the class to get and set the information for the attribute. Also, the corresponding return type and the parameter type were added to the class.

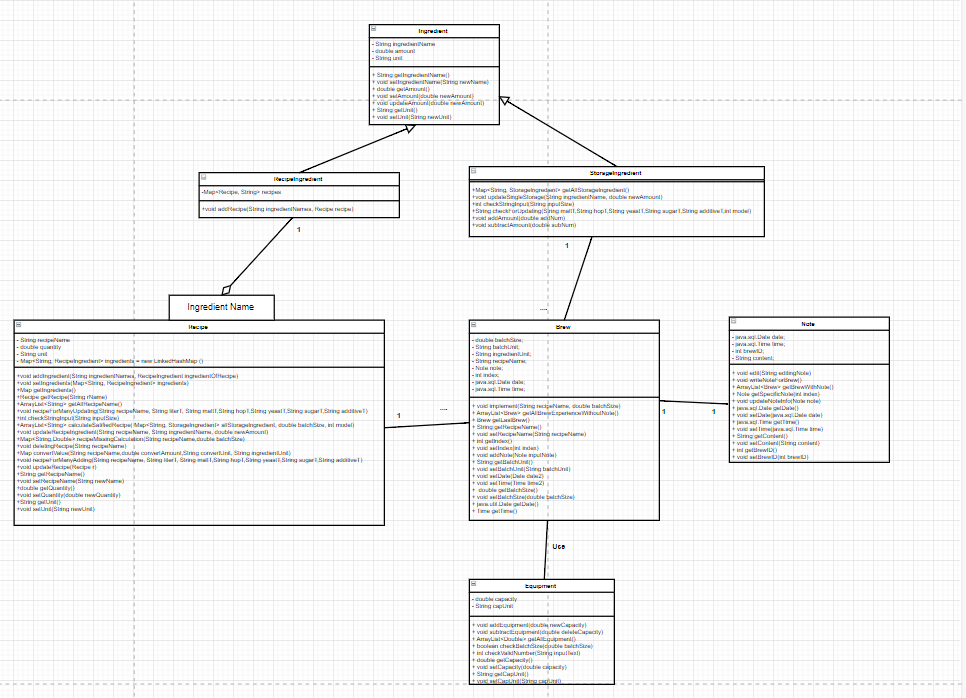
For ‘Brew class’, we have added more attribute and the operation to it. Also, the return type with the parameter type was added for the further implementation.

For ‘Equipment class’, new attribute and operations were added to the it for the detailed performance. Also, the corresponding return type and the parameter type were updated in the class.

For ‘Note class’, new attributes was added to it, as well as some kinds of the operation with corresponding return type and the parameter type for the detailed design.

# Detailed design

## Class diagram



Add the qualifier to reduce multiplicity for each one-to-many relationship:

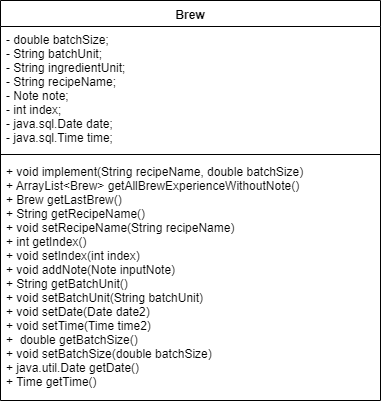
For the relationship between Recipe and Brew, add the qualifier brewID.

For the relationship between Recipe and RecipeIngredient, add the qualifier ingredient name.

For the relationship between Brew and StorageIngredient, add the qualifier ingredient name.

## Classes

### Brew

****

***Explanations***

Add the argument recipeID, batchSize as well as the batchUnit in the method of the implement() to carry on a specific brewing procedure with specific recipe and specific batch size.

New added 2 private attributes: brewID(an id to identify each brew, the type is int), batchUnit(the unit of batchSize this time, the type is string).

New added public operations: getter and setter for each attribute, the return type is void.

***Constraints (optional)***

NONE

### Equipment

***Explanations***

The bool type was added to the updateCapacity in order to decide whether the update is successfully. Also, the newValue parameter was added to specific the new capacity.

New added private attribute: unit (the unit of capacity, the type is string).

New added public operations: getter and setter for each attribute, the return type is void.

***Constraints (optional)***

NONE

### Ingredient

***Explanations***

Changed the name of the attribute name into ingredientName.

New added public operations: getter and setter for each attribute, the return type is void.

***Constraints (optional)***

NONE

### Note



***Explanations***

New added private attribute: brewID(the id of the brew each time, the type is int).

New added public operations: getter and setter for each attribute, the return type is void.

***Constraints (optional)***

NONE

### Recipe

***Explanations***

The recipeName is specified with the String type. The quantity is specified with the int type. The unit with String type is to specify the unit for the quantity. The recipeID with int type is to order the recipe for recognition.

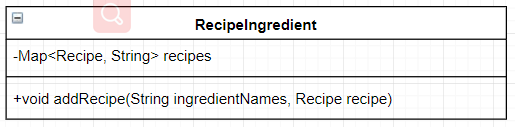
The bool return type is added to the convertValue method to decide whether the converter is success or not. Also, the newAmount parameter with the int type was added to decide what is the amount to be converted. Also, an int type of recipeID is set to decide which recipe are going to be converted.

Last, the get and set method for recipeName and quantity as well as the unit were added with the void type to get the corresponding value in the attribute and to set the new value for the attribute.

***Constraints (optional)***

NONE

### RecipeIngredient

****

***Explanations***

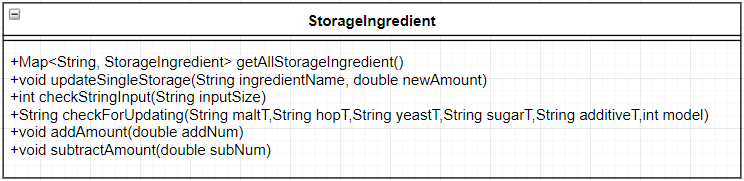
A new attribute recipeID with int type under the privacy visibility. The recipeID is to specific the ingredient for each recipe.

A return type bool is added to the updateAmount method, also, the parameter recipeID with int type and an ingredientName with String type as well as the newAmount with the int type was added to the class. The return type is to judge whether the updating is success or not. The recipeID and the ingredientName is to specific the corresponding recipe and the ingredient name. The newAmount is the specific the new amount for the ingredient.

***Constraints (optional)***

NONE

### StorageIngredient



***Explanations***

The return type for the addAmount and subtractAmout method were added with the bool. The modification is to judge whether the add or the subtract operation is success.

The parameter for the addAmount was added with parameter ingredientName with String type and the addNum with the int type. The modification is to add the specific amount to the specific ingredient.

The parameter for the subtractAmount was added with parameter ingredientName with String type and the subNum with the int type. The modification is to subtract the specific amount to the specific ingredient.

***Constraints (optional)***

NONE

# Alternative detailed design (Optional)

NONE

# More considerations

NONE