Detailed Design

for

Brew Day

Version 1.0 approved

Prepared by ZHANG Zhiyi, LUO Zichen, WANG Yuan, LU Guangxing

Cerf

9 April 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 3

3. Detailed design 4

3.1 Class diagram 4

3.2 Classes 5

3.2.1 StorageIngredient 5

3.2.2 Brewing 5

3.2.3 RecipeIngredient 5

3.2.4 Recipe 6

3.2.5 Ingredient 6

3.2.6 ShoppingList 6

3.2.7 Equipment 7

3.2.8 Note 7

4. Alternative detailed design 7

5. More considerations 8

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| ZHANG Zhiyi, LUO Zichen, WANG Yuan, LU Guangxing | 2019/4/9 | Initialize the Document | 1.0 |
|  |  |  |  |

# Overview

## Project description

For our project, we were writing an software called “Brew Day!”. This software is developed for home beer brewers, to give them some little help such as record recipes, keep track of ingredients, etc. This software will be desktop-based.

## References

[1] Zhang Zhiyi , Luo Zichen ,Wang Yuan and Lu Guangxing, Software Requirements Specification For Brew Day!, version5.0.

## Design purpose

Detailed design could bring a more specific content of the class diagram and constrains can give pre- and post- condition which can make it clearer during coding.

# Overall description

## Class diagram

## Refinements

Identify our classes’ features, while specifying their visibility and constraints, restructuring the classes as the detailed design. And, we specify the pre- and post- conditions of operations in our classes.

# Detailed design

## Class diagram

## Classes

### StorageIngredient

***Explanations***

The attributes name, amount and unit are inherited from the ingredient class in section 2.1.

Have added parameter **double newAmount** to the Operation **addAmount()**.

Have added parameter **double amount** to the Operation **subtractAmount()**

### Brewing

***Explanations***

Keep the attributes in the section 2.1.

Set the return type of implement() in void.

### RecipeIngredient

***Explanations***

The updateAmount() operation is used to modify and update the Ingredient has no return value, so it’s “void”.

### Recipe

***Explanations***

The “Get” operations return string value for GetName(), and ArrayList<Ingredient> for GetContent(). “Set” operations modify attributes, with “void” return type.

### Ingredient

***Explanations***

When get the ingredient name, it will take a return type of String. When get the ingredient amount, it will take a return type of double. When set the ingredient name, it will get the input type of String name. When set the ingredient amount, it will get the input type of double amount.

### *shoppinglist* ShoppingList

***Explanations***

When save the shoppinglist the return type of ArrayList.

### Equipment

***Explanations***

Equipment class has two private attribute, name is the name of certain equipment and size is the size of it.

GetName operation return the value of private attribute name and SetName operation can change its value. GetSize operation return the value of private attribute size and SetSize operation can change its value.

### Note

***Explanations***

Note class has two private attribute, content is the content of the note and time it the creating time of the note.

GetContent operation return the value of private attribute content and SetContent function can change its value. GetName operation return the value of private attribute time.

# Alternative detailed design

TBD

# More considerations

We use ArrayList<Ingredient> to satisfy our need for extensible and dynamic content.