**CS673S16 Software Engineering** 

**Team X - Project Name**

**Software Design Document**

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Chenyang Li | Configuration Leader/Environment and Integration Leader | *Chenyang Li* | 03/15/2018 |
| He Yu | QA Leader; Document leader | *He YU* | 3/15/2018 |
| Fengbo Gao | Security leader | *Fengbo Gao* | 3/15/2018 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Change** |
|  |  |  |  |
|  |  |  |  |

[Introduction](#_87t9hln2vjz0)

[Software Architecture](#_buttcq9i221r)

[Design Patterns](#_x18fj36s1121)

[Key Algorithms](#_mtfbusfb0eq3)

[Classes and Methods](#_7ucksmkf6rzx)

[References](#_15tmymhipvdv)

[Glossary](#_8n34lvocupub)

# Introduction

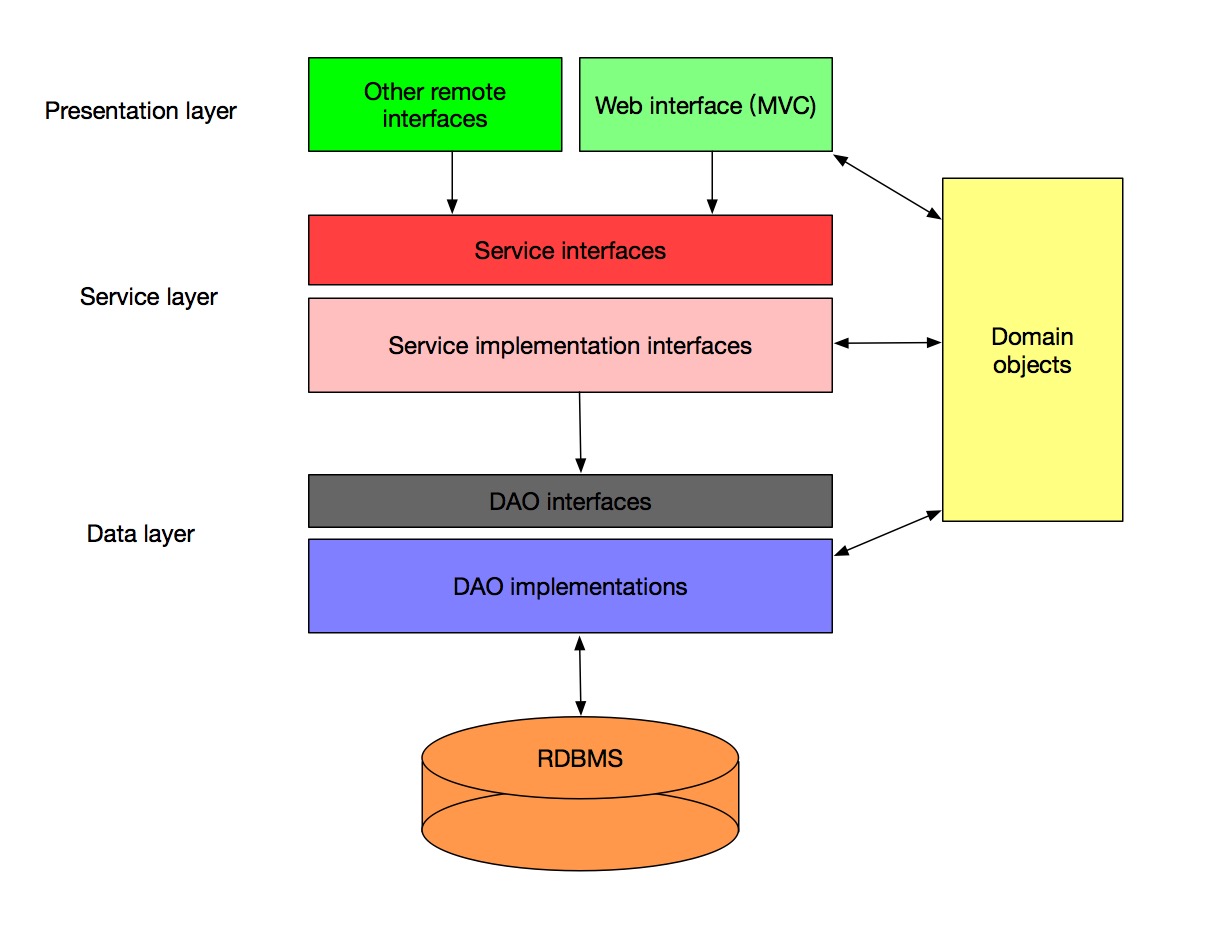
In this section, give an overview of this document, and also address the design goals of your software system.

The design goals of this project is to develop a webapp with calories calculation and meal recommendation.

# Software Architecture

In this section, you will describe the decomposition of your software system, which include each component (which may be in terms of package or folder) and the relationship between components. You shall have a diagram to show the whole architecture, and class diagram for each component. The interface of each component and dependency between components should also be described. If any framework is used, it shall be defined here too. Database design should also be described if used.

**We use Spring framework to manage different components’ instance and to provide a MVC pattern architecture, the diagram is like this:**



**As for package and folders, it’s like:**

├── main

│ ├── controller

│ │ ├── MainPageController.java

│ │ └── UserPageController.java

│ ├── dao

│ │ ├── FoodDao.java

│ │ ├── FoodDaoImpl

│ │ │ └── JDBCFoodDao.java

│ │ ├── UserDao.java

│ │ └── UserDaoImpl

│ │ └── JDBCUserDao.java

│ ├── model

│ │ ├── Food.java

│ │ ├── History.java

│ │ ├── Setting.java

│ │ └── User.java

│ ├── service

│ │ ├── FoodService.java

│ │ ├── FoodServiceImpl

│ │ │ └── FoodServiceImpl.java

│ │ ├── UserService.java

│ │ └── UserServiceImpl

│ │ └── UserServiceImpl.java

│ └── util

│ └── Response.java

├── spring-mvc.xml

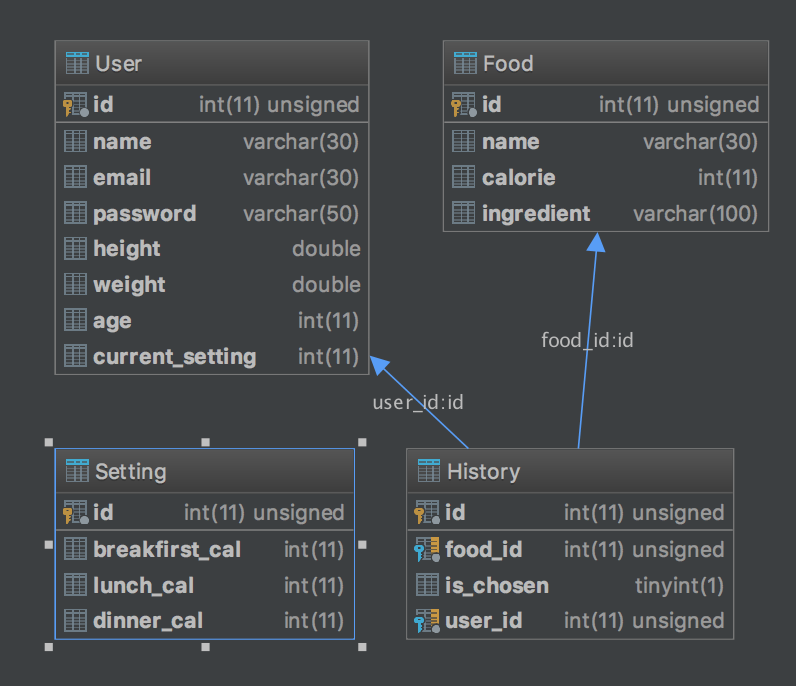
└── test

**So basically we have four components including *controller, dao, model, service* and their dependency relationship is like:**

**controller ——> service ——> dao ——> model**

**And the main class in *service* and *dao* are all interfaces, for each of which, we can have multiple implementation.**

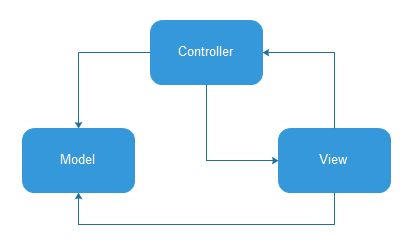
**And our database diagram is like this:**

****

# Design Patterns

In this section, you shall describe any design patterns used in your software system.

The primary design pattern is MVC which is mentioned in the software architecture section, and the main structure is also showed there.



# Key Algorithms

In this section, you shall describe any key algorithms used in your software system, either in terms of pseudocode or flowchart.

**The key function is to recommend food for users. The basic idea is search food according to users’ preferences towards food tags and their calorie goals. Each user is asked to choose specific food tags they prefer after they sign up. Also, they are asked to fill in some personal information, such as weight, height, age and gender, which are added to calculate calorie limits (BMR values). Then every time the user requesting a food recommendation, the backend would filter out food that meets both food preferences and calorie goals. If more than one kind of food meets the requirement, then the app will random one for the user.**

**Food Recommendation Algorithm:**

*Given user has requested food recommendation and sets his/her calorie goal:*

*// set calorie goal*

*setCalorie();*

*Temp\_calorie = getCalorie();*

*// user has already set their preferences to food when they first sign up*

*Food\_tag = user.getPrefer();*

*while(TRUE){*

*Random Food\_id = new Random().nextInt(max);*

*food temp = getfood(food\_id);*

*if(food.tag == Food\_tag && food.calorie ≤ temp\_calorie)*

*Return food;*

*break;*

*Else continue;*

*}*

**BMR Calculation:**

*If gender = “Male”*

*If gender = “Female”*

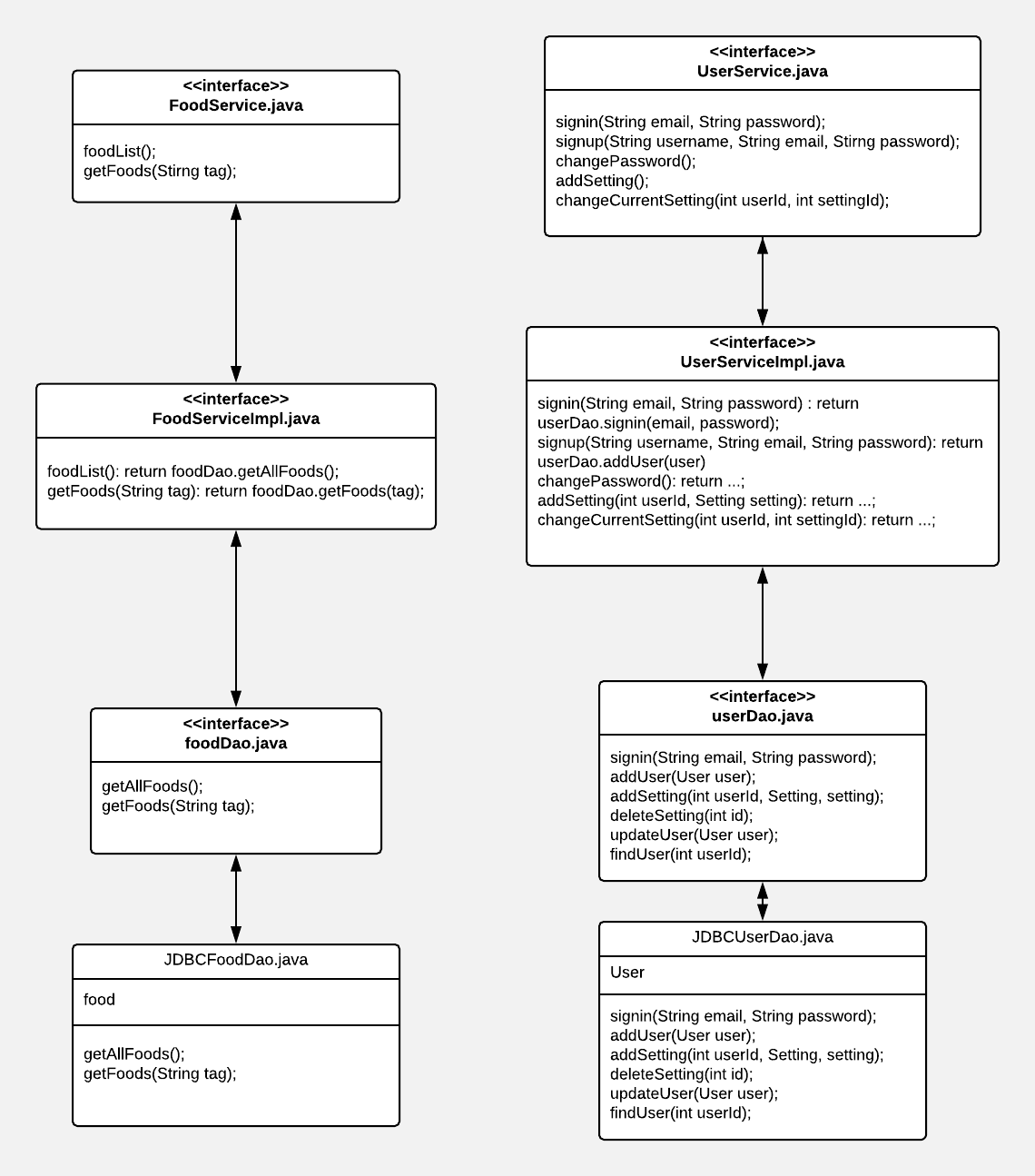
# Classes and Methods

This part can be a reference to automatic generated document for all classes and methods.

**According to the architecture and framework, the classes of frontend are mainly in the Index.html file and style.css file. The head() and body() classes are two important classes in the Index.html and style.css file. We add many classes into those two big classes. For example, we add ul() class as navigation bar in the head() class and Button() class in the body() class. We also have button() classes in both Index.html and style.css to design buttons in the home page.**

**In the app-ajax.js file. We have two important classes, one is loginbtn() and the other is signupbtn(). The loginbtn() is used to design the login button for users and signupbtn() is used to design the signup button for users.**

**According to the architecture and framework, these classes are implemented to achieve different functionalities of different layers, the class diagram is shown as below, which is drawn by Lucidchart:**



# References

# Glossary