

System Design Specification

A program assisting healthily getting fat



제출일	2018. 5. 6	전공	컴퓨터공학부
과목	소프트웨어공학	학번	20141405
담당교수	송인식	이름	최형준

Contents

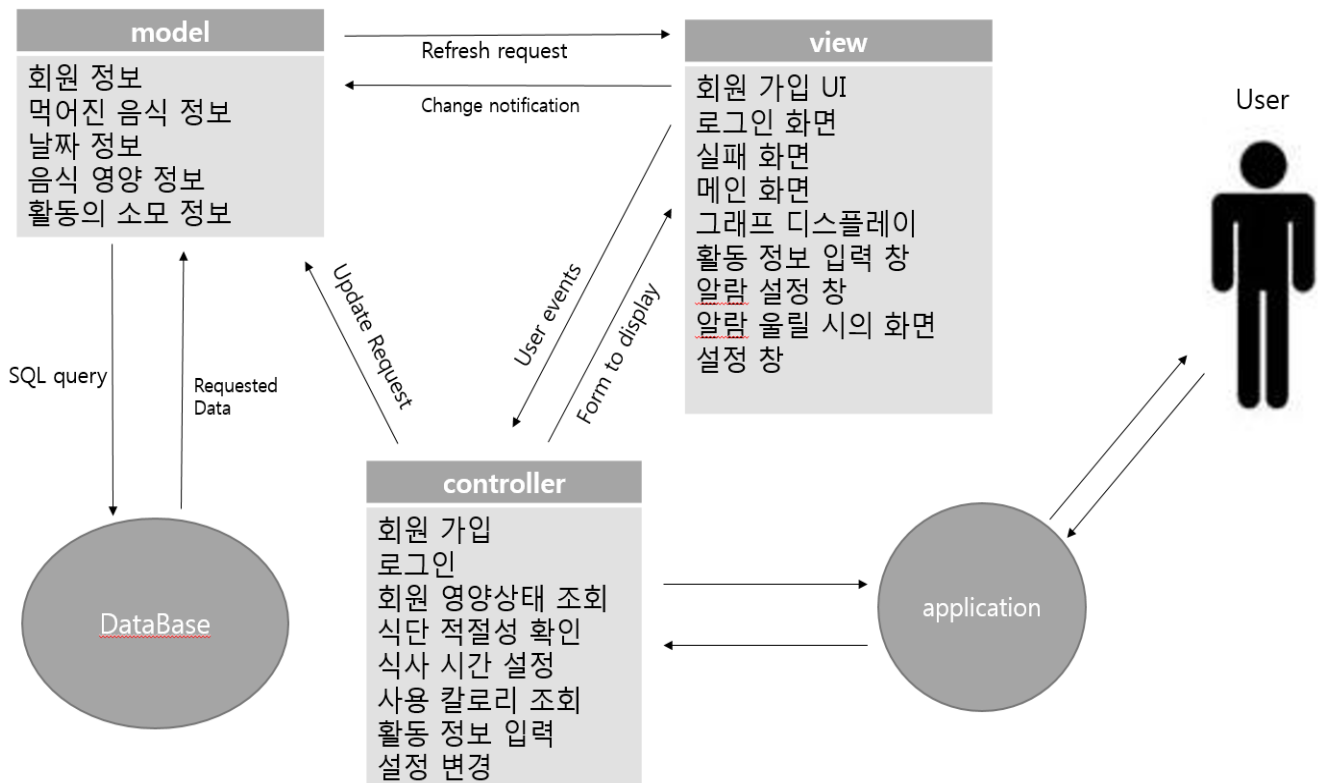
I. System Architecture

- A. major modules and their functionality
- B. interfaces between modules
- C. data description (high-level database schema if possible)
- D. design alternatives considered (pros and cons)

II. Process

- A. Software tools
- B. Risk assessment
- C. Team structure
- D. Project schedule
- E. Test Plan (including unit, system, usability test and bug tracking)
- F. Documentation plan
- G. Coding style guidelines

I. System Architecture (MVC model)



- 컨트롤러에는 사용자가 프로그램 내에서 실행 시킬 수 있는 모든 함수들이나 기능들이 있으며 이러한 기능들은 사용자는 어플리케이션을 통해서만 컨트롤러의 함수들을 실행시켜 실행 및 데이터에 대한 조작이 가능하다. .

- 컨트롤러의 기능들은 모두 모델의 데이터와 View를 이용하여 사용자에게 제공하는데, 이를 위해 기능이 실행 됐을 때 View Module에 관련 UI에 대한 기능을 실행 할 것을 요구하고 View에서 UI를 전달하면 컨트롤러는 사용자가 사용하는 어플리케이션에 UI를 띄워 사용자가 볼 수 있게 한다.

- 컨트롤러는 기능이 실행 됐을 때, 데이터에 대한 조작 및 업데이트가 필요하면 모델의 데이터 처리 관련 함수를 실행하고 모델의 데이터 관련 함수는 컨트롤러가 원하는 데이터 조작에 관한 SQL query문을 작성 및 실행하여 로컬 DB 에 있는 정보를 가져오거나 업데이트 하고 정보를 웹DB를 통해 가져와야 한다면 open API 혹은 JSoup를 통한 crawling을 통해 정보를 가져온다.

- 가져오거나 데이터의 값을 변경했을 시에 모델은 이를 View에 전달하고 View는 바뀐 값이나 가져온 값들을 사용자가 볼 수 있게 UI에 적용시킨다.

A. Major modules and their functionality

Modules	Functionality
Login control	입력 받은 아이디와 비밀번호가 DB에 있는 회원정보들과 일치하는지 확인하고 일치하면 로그인 하면서 회원 정보를 가져오고 일치하지 않으면 오류 창을 보여준다. 회원 가입을 누르면 회원 가입 창을 보여주고 입력 받은 값을 DB에 회원 정보로 저장한다.
Crawling control	유저에게 입력 받은 음식의 영양정보와 칼로리를 식품 안전 나라의 식품 영양 정보 웹 DB에서 API를 통해 가져온다.
DB Manipulator	회원 정보를 DB에 저장하거나 가져오는 기능, 유저가 먹은 음식의 영양정보와 목록을 저장하고 가져오는 기능, 먹은 날짜를 기록하여 저장하고 가져오는 기능 등 DB와 관련된 작업을 담당한다.
Arithmetic Operation	유저의 먹었던 음식 정보를 통해 누적 칼로리, 영양 성분 등을 계산하거나 그 정보를 통해 유저가 당일 먹은 영양 정보와 비교 연산을 통해 그 식단이 적절한지 아닌지를 판단해준다. 또 유저의 기초 대사량도 실시간으로 계산해주며 유저가 한 활동의 소모량을 계산해준다.
controller	프로그램이 실행 되었을 때, 초기화면을 보여주고 로그인 화면을 띄워주거나 유저의 입력을 통한 기능 실행 시 필요한 화면을 띄워주는 기능을 전반 담당하고 칼로리 소모 값이나 먹은 영양 성분의 값을 날짜와 결합하여 그래프로 보여준다.
Alarm control	유저에게서 시간 정보를 입력 받으면 그 시간 정보를 로컬 데이터로 저장 후에 그 시간이 되면 알람이 울리면서 Alarm UI를 요구하고 사용자가 알람을 끄면 저장한 시간 정보를 삭제한다.
Time	Alarm클래스에서 시간 처리를 위해 호출되어 사용된다.
User Setting module	한 유저를 선언 및 저장할 때 사용하는 클래스로 세팅에서 로그아웃하고 탈퇴(DB삭제)
View Graph	유저가 프로그램과 상호작용 할 수 있는 인터페이스를 제공하고 누적 데이터에 따른 그래프를 보여주며 유저가 버튼을 눌러 이벤트가 발생했을 때, 발생한 이벤트에 따른 행동을 프로그램이 하도록 컨트롤러 함수를 호출한다.

#User Module(class)

User
-u_Id : String -password : String -sex : boolean -height : double -weight : double -age: int -BMR:double
+getUserInfo() +setUserInfo() +getBMR()

#explanation

유저 하나를 정의하는 클래스로 대부분의 클래스에서 사용한다. 클래스 내에는 유저를 구분하는 ID와 password 변수가 있고 유저의 성은 남자 혹은 여자인지 때문에 Boolean 형으로 정의하였다. Height와 weight는 유저의 몸무게와 키를 나타내며 BMR은 사람의 기초 대사량인데, 이는 getBMR 함수를 통해 computation module을 실행 시켜 키와 몸무게, 성을 바탕으로 구한다. getUserInfo()함수는 로그인 창에서 입력 받은 아이디와 패스워드가 DB내에 존재할 때 그 ID와 일치하는 정보를 받아온다. SetUserInfo는 로그인 후에 유저가 자신의 정보를 바꿀 때에 이를 DB에 저장한다.

#pseudo code(getUserInfo)

```
If(LoginModule.compareIDPassword()==true){  
  
u_Id = LoginModule.ID  
password=LoginModule.Password  
User.sex,User.height,User.weight,User.age = DBmodule.getUserInfo(User.ID)  
  
}
```

#Login Module(class)

Login
-inputID : String
-inputPassword : String
-inputSex : Boolean
-inputHeight : double
-inputWeight : double
-inputAge : double
+registerUser ()
+compareIDPassword()

#explanation

어플리케이션의 UI 모듈에서 가장 먼저 로그인 UI를 띄우고 생성하는 클래스로 UI모듈에서 입력 받은 스트링 값이나 패스워드 값을 저장하고 유저가 로그인 UI 에서 로그인 버튼을 클릭하면 compareIDPassword 함수를 실행하고 이 함수 안에서 DBmodule에서 데이터 비교 함수를 실행해서 일치하는 값이 DB에 있으면 true를 그렇지 않으면 false를 반환한다. 유저가 로그인 UI에서 회원가입 버튼을 눌렀을 때 회원 가입 UI가 나타나고 회원 가입 UI에서 회원 가입 버튼을 눌렀을 때 UI안에서 입력받은 정보들을 registerUser 함수를 통해 input변수들안에 집어 넣고 이를 DB 모듈함수 실행을 통해 DB에 저장한다.

#pseudo code(registerUser)

```
LoginModule.inputID = UImodule.registerID
LoginModule.inputpassword = UImodule.registerpassword
LoginModule.inputsex,weight,age,height = UImodule.registerinformation
If(DBmodule.Search(Userinfo,Loginmodule.inputID)==false){

DBmodule.updateUserInfo(LoginModule.inputsex,weight,age,height)

}
```

#Time(class)

Time
-hour : int
-minute : int
-second : int
+setPresentTime() +getHour() +getMin() +getSec()

#explanation

Time 클래스는 시간을 정의하는 클래스로 Alarm클래스에서 시간 처리를 위해 호출되어 사용된다. 시스템의 현재 시간은 Java의 Calendar라이브러리의 get(Calendar.HOUR), get(Calendar.MINUTE), get(Calendar.SECOND)를 통해 받아온다. 그리고 setTime()함수다. setTime()함수를 호출하면 해당 시간 정보를 private 변수인 hour, minute, second에 저장해준다. 이러한 private변수에 접근이 필요로할 때는 getHour(), getMin(), getSec()함수를 사용하면 된다.

#pseudo code(setTime)

```
hour = get(Calendar.HOUR);  
min = get(Calendar.MINUTE);  
second = get(Calendar.SECOND);
```

#Alarm(class)

Alarm
-eatTime : Time
+getEatTimeDB() +setAlarmTime() +startAlarm() +stopAlarm()

#explanation

Alarm클래스는 밥을 먹을 시간을 알려주는 기능의 전체적인 부분을 담당하는 클래스이다. 때문에 식사시간 정보를 나타내는 Time 객체를 선언한다. 해당 식사 시간은 DB에 저장되어 있으므로 getEatTimeDB()함수를 통해 로컬 DB에 저장되어 있는 가장 최신의 식사 정보를 가져온다. setAlarmTime()함수에서는 사용자가 식사를 한 시간 기준으로 6시간 이후를 식사 예상 시간으로 설정하고 해당 시간을 Time형식으로 반환해준다. startAlarm()은 실질적으로 6시간이 지났는지를 판단하고 소리를 출력해주는 함수로 Java의 thread를 사용하여 무한루프를 돌며 알람을 출력 시킬 시작인지 판단하고 시간이 되면 알람 소리를 발생시킨다. 사용자가 알람을 끄면 stopAlarm()함수가 실행되고 DB의 알람 정보와 알람 소리를 발생시키기 위해 루프를 돌고 있던 thread를 종료시킨다.

#pseudo code(startAlarm)

```
endTime = setAlarmTime(eatTime);
thread.run();

void run(){
    while(now == endTime){
        //generate alarm sound
    }
}
```


#Controller Module

Controller
<div>-myUser : User -myLoginModule : LoginModule -myDBmodule : DBModule -myAlarm : Alarm -myGraph : Graph -mySetting : Setting</div>
<div>+login() +register() +setting() +alarm() +showGraph()</div>

#explanation

Controller클래스는 다른 class들을 instance로 선언해 정보를 가져오고, 정보에 따라 user interface를 제어하며, 사용자의 입력에 따라 instance의 값을 변경하는 클래스이다. user 정보를 처리하고, Login 기능을 처리 하기 위해 User 객체와 LoginModule 객체를 갖는다. 또한 Alarm객체를 통해 사용자가 지정한 Alarm 시간이 되면 소리와 함께 메시지를 출력하게 된다. Setting객체를 통해서는 사용자가 프로그램의 setting화면에 들어가 프로그램 환경 설정을 할 수 있다. 그리고 Controller에 있는 함수들은 각종 기능들을 control 할 수 있도록 해준다.

#Math Module(class)

Math
+totalNutrient() +evaluateEatenFood() +calculBMR()

#explanation

수학적인 연산과 관련된 클래스이다. 유저의 EatenFood를 통해 totalNutrient를 계산한다. 그 정보를 통해 유저가 당일 먹은 영양 정보와 비교 연산을 통해 그 식단을 평가해준다(eg. 적절, 부족, 과다). 또 유저가 성별, 몸무게 등을 입력 했을 때 BMR(basal metabolic rate, 기초대사량)을 계산해준다.

#pseudo code (calculBMR)

```
Double calculBMR(User user) { // 해리스-베네딕트 공식

    User tmp = user.getInfo();

    if(tmp.sex == MALE) {

        return ( 66+(13.7*tmp.weight)+(5.0*tmp.height)-(6.8*tmp.age) );

    }

    else if(tmp.sex == FEMALE) {

        return ( 655+(9.6*tmp.weight)+(1.8*tmp.height)-(4.7*tmp.age) );

    }

}
```

#DB Manipulator Module(class)

DB	FoodInfo
<ul style="list-style-type: none">-userInfo : User-date : String-foodName : String-actInfo : double[]-eatenInfo : FoodInfo-foodInfo : FoodInfo	<ul style="list-style-type: none">-userID : String-date : String-foodID : int-foodName : String-foodNutrient : double[]
<ul style="list-style-type: none">+setUserDBInfo()+getUserDBInfo()+setEatenDBInfo()+getEatenDBInfo()+setActDBInfo()+getActDBInfo()+setFoodDBInfo()+getFoodDBInfo()+search()	<ul style="list-style-type: none">+getNutrient()+setNutrient()+getFoodName()+setFoodName()+getFoodID()+setFoodID()

#explanation

DB에서 유저의 정보, 먹은 음식에 대한 영양정보, 음식영양정보, 행동을 통해 소비한 칼로리 등을 받아온다. 유저정보는 구분을 위해 ID를 저장하고, 먹은 음식에 대한 영양정보와 음식영양정보는 각각 eatenInfo 와 foodInfo 라는 클래스로 인자를 전달받아 DB에 저장한다. 이 배열 안에는 음식에 대한 모든 영양정보가 들어 있다. 각각 get과 set으로 DB로부터 정보를 가져오고 저장할 수 있다. search 메소드는 원하는 정보가 로컬 DB안에 있는지 없는지 검색하여 있으면 TRUE를 없다면 FALSE를 반환한다.

#pseudo code(search)

```
while(searching){  
    if (local DB have) return TRUE  
    else return FALSE  
}
```

#Crawling Module(class)

Crawling
-url : String -doc: Document -foodInfo : FoodInfo -elements : Elements
+getFoodInfo () +connectURL ()

#explanation

웹 페이지로부터 JSoup 라이브러리를 이용하여 정보를 긁어오는 기능을 담당하는 모듈이다. connectURL 함수에서는 String 형 url에 접속할 페이지의 주소를 입력하고 원하는 문장은 String 형으로 정의된 query 파라미터로 전달한다. JSoup에서 제공하는 URLEncoder 클래스의 encode함수에 query를 넣고 이를 url에 전달한다. Url을 이용하여 JSoup에서 제공하는 Document 형에 해당 페이지의 정보를 모두 받아온다.getFoodInfo 함수에서는 받아온 Document페이지 안에서 html tag를 통해 원하는 음식의 정보를 순서대로 파싱 하는데 파싱하기 전 Elements 형 변수에 저장 시킨 후에 파싱 하여 Foodinfo class 안의 변수들에 저장시킨다.

#pseudo code(getFoodInfo)

```
getFoodInfo(query page you want, doc){  
    if(connect == true){  
        elements = doc.select("tag")  
  
        for(number of elements){  
            foodInfo's variables = elements.attr("each variable")  
        }  
        Return foodInfo  
    }  
}
```

#UI module

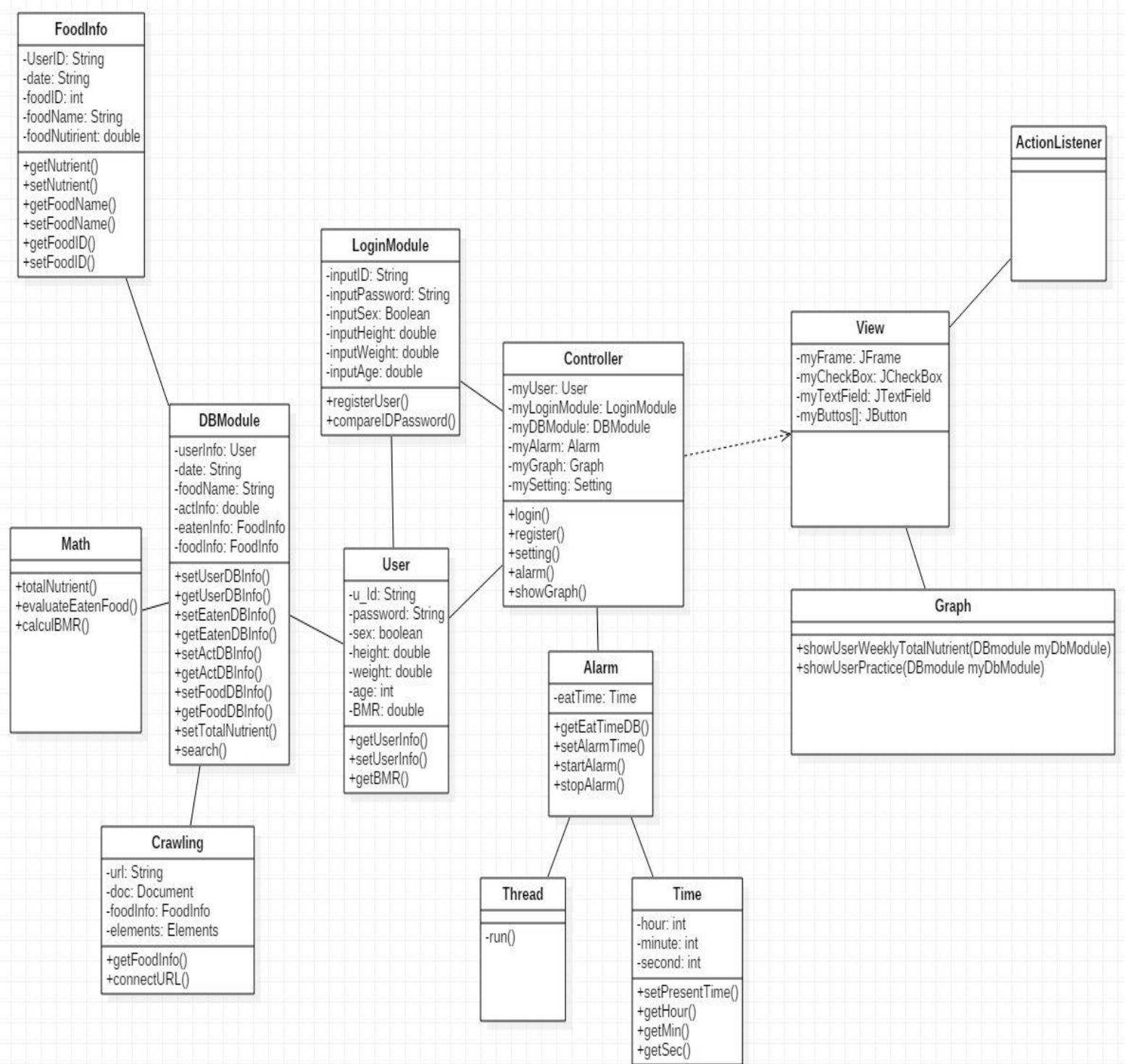
View	Graph
<ul style="list-style-type: none">-myFrame : JFrame-myCheckBox : JCheckBox-myTextField : JTextField-myButtons[]:JButton-actionListner : inner class	
	<ul style="list-style-type: none">+showUserWeeklyTotalNutrient (DBmodule myDbModule)+showUserPractice (DBmodule myDbModule)

#explanation

이 프로그램은 java의 swing을 활용해 UI를 처리하므로, View 클래스는 myFrame, mycheckBox, myTextField, myButtons[]와 같은 swing의 요소들을 갖는다. 각종 actionListner들은 inner class이며, 그 안에 있는 함수 actionPerformed (ActionEvent event)들을 통해 사용자들의 행동에 따라 UI에 변화를 제공하고 Controller class의 함수를 호출시킨다.

Graph class는 user와 관련된 정보들을 graph를 통해 user가 한눈에 파악할 수 있도록 시각화 해주는 역할을 하는 class이다. showUserWeeklyTotalNutrient (DBmodule myDbModule)는 DBmodule로부터 날짜에 따른 user의 영양정보를 받아와서 이를 시각화 해주는 함수이다. showUserPractice (DBmodule myDbModule)는 DBmodule로부터 날짜에 따른 user의 칼로리 소모 정보를 받아와서 이를 시각화 해주는 함수이다.

B. Interfaces between modules

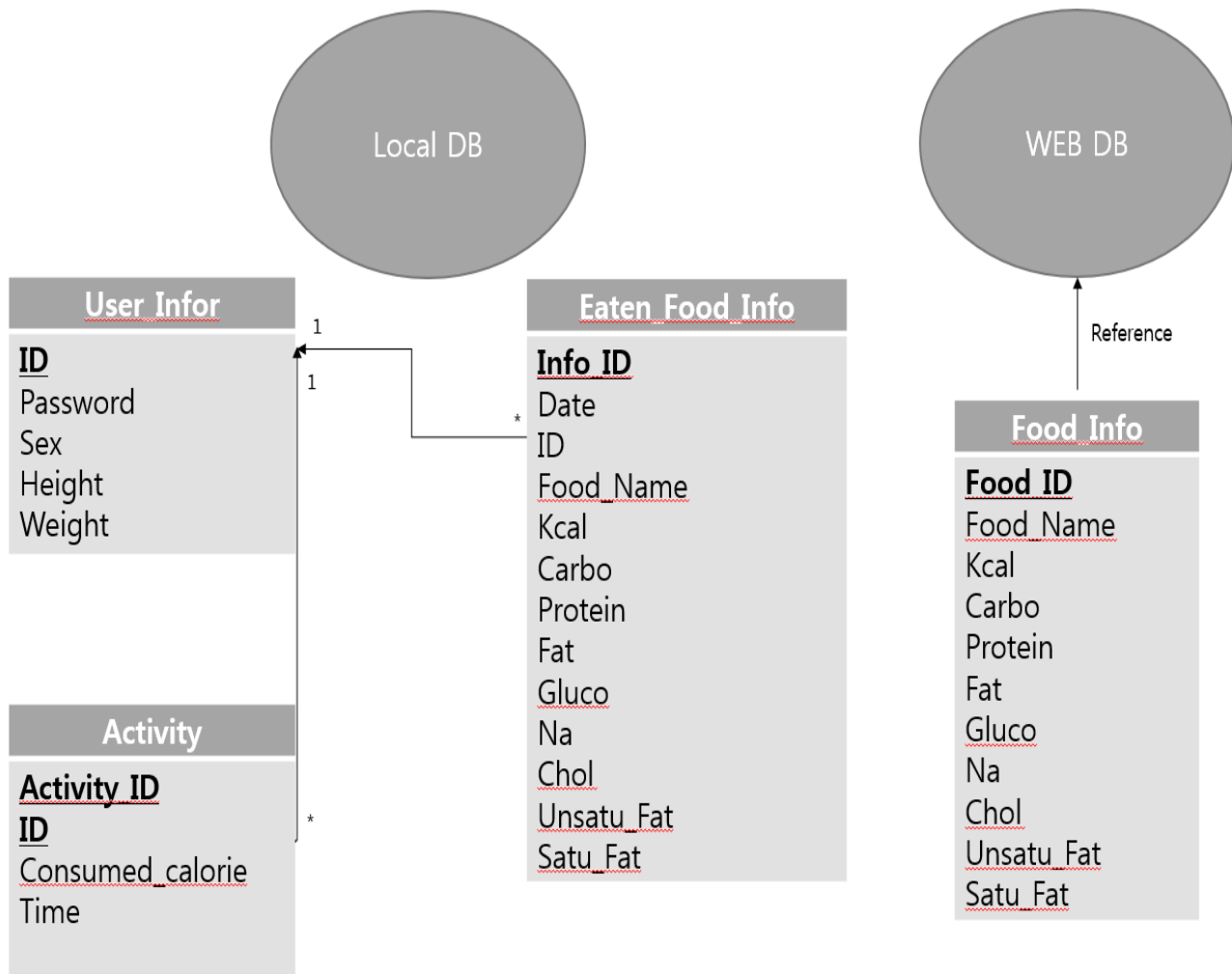


StarUML 툴을 이용하여 모듈간의 인터페이스를 UML class diagram으로 나타내었다.

이 diagram을 이용함으로써 각 클래스를 어떤 식으로 나누고 각 클래스를 어떤 클래스 안에서 생성하고 이용해야 하는지 의존 관계를 파악할 수 있다.

컨트롤러와 뷰는 dependency가 존재하기 때문에 점선으로 표현하였다.

C. Data description



Bold and underlined attributes are Primary key of each tables.

Left 3 tables are in local DB which is controlled by DB manipulator

Right 1 table is structure of the data that we get from Web by crawling control module.

D. Design alternatives considered (*pros* and *cons*)

<MVC model>

Pros

- program 내에서 사용할 data set 을 명확하게 정의하고 이 data들이 user가 application의 어떠한 기능들을 실행시킬 때, 어떻게 조정되고 어떻게 DB와 통신하는지를 한눈에 알 수 있다.
- User가 실행시킬 수 있는 기능들에 대한 리스트가 정확하게 정의되어있고 그 기능을 제공하기 위해 필요한 UI가 미리 모듈화되어 있어서, user가 program 상에서 경험 가능한 모든 이벤트들을 상정 가능하다.

Cons

- 코드로 모듈들이나 함수들을 상세 구현할 때 일어날 수 있는 코드들의 충돌이나 데이터의 충돌 혹은 함수의 중복 등을 예상할 수 없다.

<Repository model>

Pros

- 한 곳에서 변경된 사항이 모든 곳으로 전파되기 때문에 일관성 있게 data set을 관할 수 있다.
- 많은 양의 data set들이 공유되고 옮겨질 때, 효율적이고 정확하게 전달 될 수 있기 때문에 data들의 손실이나 변조 가능성이 적다.

Cons

- Repository 시스템에 문제가 생기면 전체시스템에 영향을 주게 되고, 효율적으로 서로 공유할 수 있던 장점이 없어지고 비효율성이 커진다.

<Client – Server model>

Pros

- 분산 네트워크 서비스기반으로 동작하여 다양한 location에서 data set이 공유될 때 사용되므로, 구현된 기능을 모든 user들이 사용할 수 있다.

Cons

- Program이 server기반으로 네트워크에 의존하여 동작하기 때문에 서비스 거부 공격 등이나 서버 오류 등에 영향을 받기 쉽고 이를 쉽게 예측하기 힘들다.

II. Process Description

A. Software Tools

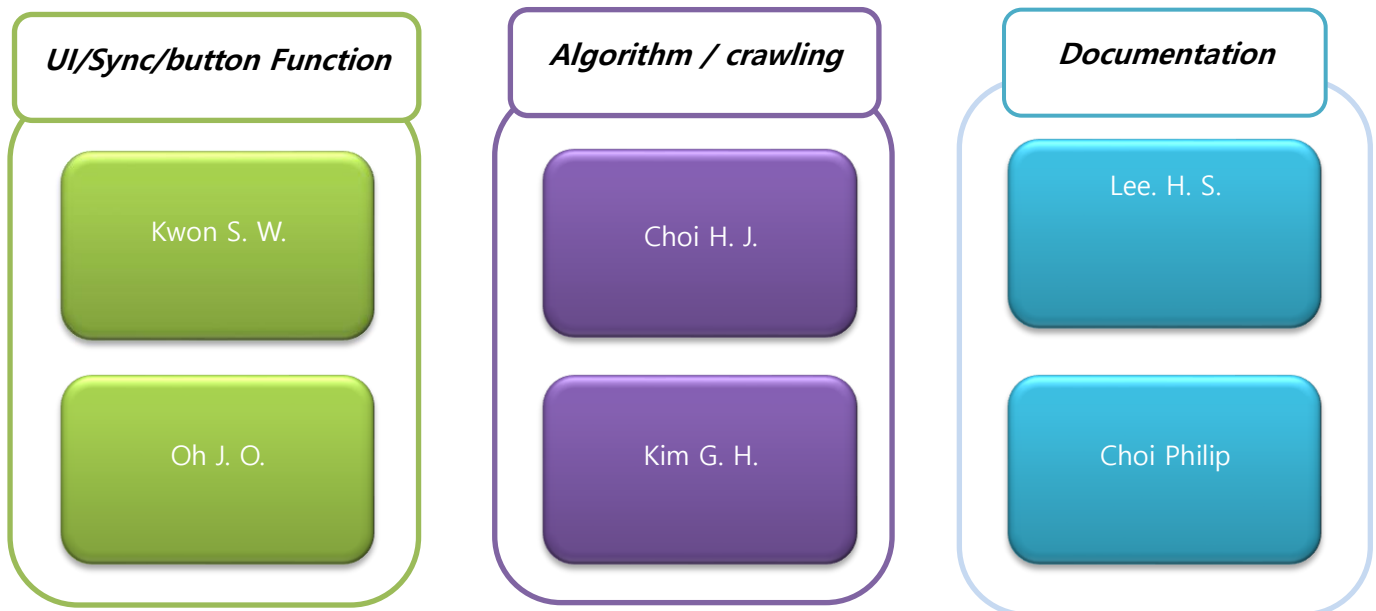
Name	Feature	Usage
Java Swing	Java Library	Provide GUI functions in Java
Jsoup	Java Library	Offer crawling functions in Java
Eclipse	Java IDE	Edit and compile Java codes

B. Risk Assessment

Risk	Likelihood (eg High, Medium, Low)	Risk Response Strategy (eg Avoid, Transfer, Mitigate, Solvable or Accept the risk)	Actions required
Lack of sample data in a web server	High	Mitigate	Adding new data to Local DB
Illegal attempts on personal user data	Low	Avoid	When saving user information to database, use hash encryption.
Risks to the project caused by requirements that are inadequately defined.	Low	Solve	Modify requirements.
User types wrong information	Low	Mitigate	Guide users to enter information correctly and retype the information. (eg Tutorial, F&Q)

Project team member(s) will not be able to work when required	Low	Mitigate	Remaining members share their work or delay the develop process. (Depending on the situation)
Project schedule will exceed	Low	Mitigate	Team members who are free or have less remained tasks take the exceeded job.
Network environment may be bad	Low	solve	Save frequently use data in local DB storage

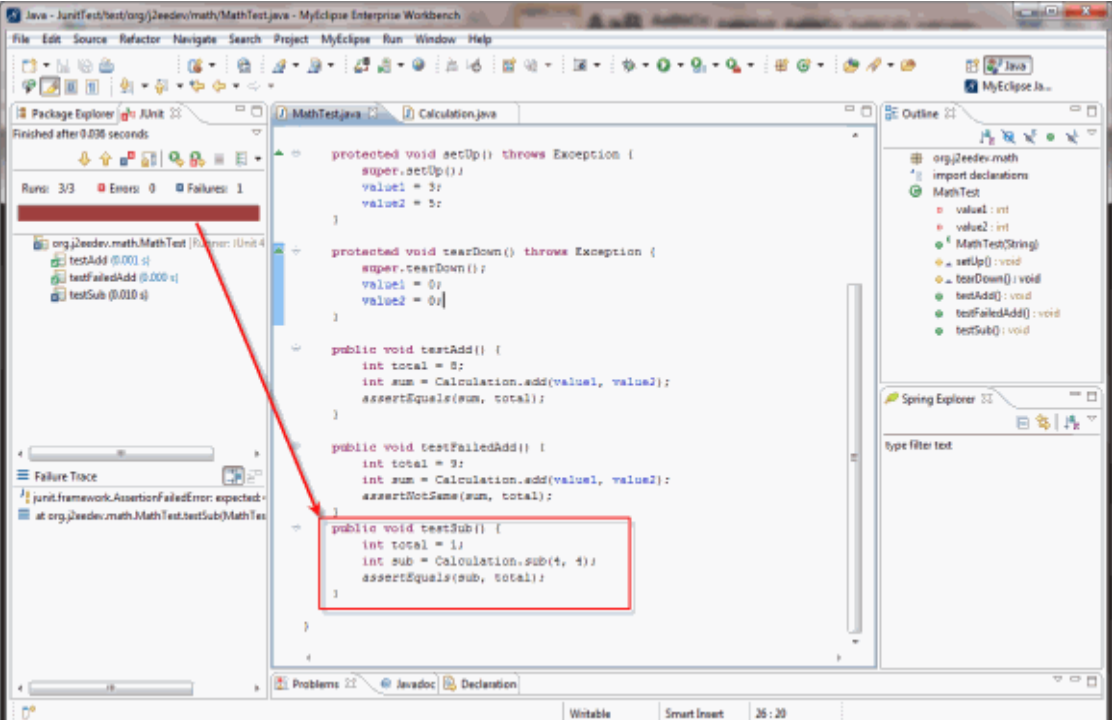
C. Team Structure



D. Schedule(Pair programming)

Week	To-Do
Week 6	<p>Oh: Define functions of the program.</p> <p>Kwon: Define non-functions of the program.</p> <p>Lee: Write user scenario(.show user's daily consume)</p> <p>Choi. H: Write user scenario.(check whether food is appropriate)</p> <p>Choi. P : risk summary & process description</p> <p>Kim: UI design</p>
Week 7	<p>Oh, Kwon :Java Swing(UI) design proposal</p> <p>Choi .H, Kim: make Algorithm for Obtaining & Saving user's personal data</p> <p>Choi. P, Lee: documentation for the process of week.</p>
Week 8	Mid-term Examination
Week 9	<p>Oh, Kwon: implement Java Swing(UI) for the function1(obtaining & saving)</p> <p>Choi .H, Kim: make Algorithm for crawling information from web</p> <p>Choi .P, Lee: documentation for the process of week.</p>
Week 10	<p>Oh, Kwon: implement Java Swing(UI) main screen</p> <p>Choi .H, Kim: make Algorithm for Check whether a newly added food is appropriate</p> <p>Choi .P, Lee: documentation for the process of week.</p>
Week 11	<p>Oh, Kwon: implement Java Swing(UI) graph, sync button with functions</p> <p>Choi .H, Kim: make Algorithm for Alarm call for a regular meal time</p> <p>Choi .P, Lee: documentation for the process of week.</p>
Week 12	<p>Oh, Kwon: implement Java Swing(UI) lists ,sync button with functions</p> <p>Choi .H, Kim: optimize algorithms for speed</p> <p>Choi .P, Lee: documentation for the process of week.</p>
Week 13	Integrate all programs and fix errors
Week 14	Get feedback and test all functions.
Week 15	Final Demonstration
Week 16	Final Examination

E. Test plan

<p>Unit Test</p>	<p>After one function or method is created, test cases have to be generated to check the functionality. In this project, Junit Framework is used to check single function or method automatically and refactor that after the test.</p>  <p>The screenshot shows the Eclipse IDE with the JUnit test run results. The test run shows 3 tests: testAdd (0.001s), testFailedAdd (0.000s), and testSub (0.010s). The testSub method is highlighted in the code editor.</p>
<p>Usability Test</p>	<p>Firstly, create several user scenarios including normal and abnormal cases. Based on these scenarios, testers who are team members check the actual software. When there are bugs on the program, bug tracking method is used.</p>
<p>Bug Tracking</p>	<p>The code is mainly run in Java, and the running environment is eclipse. In the eclipse, there is a debugger for Java language. When the unexpected error is occurred in the coded result, programmers check the specific code sections. From the debugger console, all values in these sections can be overviewed.</p>

F. Documentation plan

Documentation Requirements	Details
File Extension	Microsoft office word, Adobe Acrobat PDF
Font Style	맑은고딕(본문)
Font Size	Title(20pts w/ bold), subtitle(16pts w/ bold), main text(12pts)
Version	Use modified date and time on file name
Language	English, Korean(optional)

- i. The project schedule, user interface, and modules are finely followed by the initially planned document.
- ii. When new features is needed to be newly add to the project, before simply coding the feature, the details about the features are needed to be recorded in the documentation.
- iii. Small portion changes in code changes version to 0.1. However, big changes like changes in module or user interface increase version to 1.0.
- iv. After updating the documentation, the code revision can be continued.
- v. Documentation updates have to be rechecked through comparing actual coded results with the documentation.
- vi. Total review on the documentation is conducted after the program is completely made. This final documentation review is conducted by several team members iteratively. If fixation on the code is found, the documentation will be revised.

G. Coding style guidelines

I. Naming Conventions

- Choose meaningful and descriptive names.
- Capitalize the first letter of each word in the class name.
Ex) NumberOfObjects
- Capitalize all letters in constants.
- Use lowercase for the first word and capitalize the first letter of each subsequent word in the name.

II. Indentations

- use End-of-line style
Ex)

```
public class Test {  
    public static void main(String[] args) {  
        System.out.println("Block Styles");  
    }  
}
```
- Tab = four spaces

III. Comments

- Every comments should be above on the line that you want to explain.
- Brief and specific