

COMP-SCI 5551 (FS15) - Advance Software Engineering

Project Group 5: *Cuong Cu, Tarun Shedhani, Ting Xia*

Fourth increment report - HealthKeeper

1. Introduction

This “HealthKeeper” app concerns overweight, diabetes and hypertension, which are the three health issues faced by many people. This app will help people monitor some important parameters for the three health issues, give pertinent suggestions, keep record, and make graphical analysis.

2. Objectives/Features

2.1. Login/logout

Let user login through username and password. User also can logout whenever they want. If the user forgets password, he/she can retrieve it through email. The system will send the verification code to let the user to reset password.

2.2. Registration

If a user does not have an account, the user can register. In this step, the user needs to provide personal information to create an account.

2.3. Detection

In homepage, user can select one of the three health concerns (overweight, diabetes and hypertension), to monitor. When one health concern is selected, the user will first see the detection board. The user can first input the parameters and then know the situation he/she has. The user can also make a record to the database.

2.4. Suggestion

When user knows the detection result, system can provide the suggestion to the user.

2.5. History/Reminder

User can see all the recorded history information. The system provides the graphic analysis. Also, the suggestion history is provided.




The system will also make a recording reminder to the user.

3. Import Existing Services/API

Nutrition API: to provide the nutrition information of the food.




4. Detail Design of Services

4.1. WireFrames and Mockups

<div><h2>HEALTH SCOPE</h2></div> <div><input type="button" value="Log in"/> <input type="button" value="Sign up"/></div>	<div><div> Username <input type="text"/></div><div> Password <input type="password"/></div><div>Forgot password?</div><div><input type="button" value="Log in"/></div></div>
---	--

First page

Login page

<div><h2>Welcome!</h2></div> <div><div>Username <input type="text"/></div><div>First name <input type="text"/></div><div>Last name <input type="text"/></div><div>Date of birth <input type="text"/> / / </div><div><input type="radio"/> Male <input type="radio"/> Female</div><div>E-mail <input type="text"/></div><div>Password <input type="password"/></div><div>Re-enter password <input type="password"/></div><div><input type="button" value="Create"/></div></div>	<div><h2>Reset your password</h2></div> <div><div>E-mail <input type="text"/></div><div><input type="button" value="Send code"/></div><div>Verification code <input type="text"/></div><div><input type="button" value="Validate"/></div><div>New password <input type="password"/></div><div>Re-enter password <input type="password"/></div><div><input type="button" value="Update"/></div></div>
--	---

Registration

Reset password

←

Setting

Username

First name

Last name

Weight

Height

Date of birth

/

/

☐ Male

☐ Female

Password

E-mail

Re-enter password

Update

Good morning User!

≡

Please select the health concerns:

Diabetes

Overweight

Hypertension

Setting

Homepage

←

Diabetes detection board

Sugar level

mg/dl

2 hour glucose

▼

Fasting glucose

Detect

Detection results (can be either):

Normal

or

Impaired fasting glycaemia

or

Impaired glucose tolerance

or

Diabetes mellitus

Get suggestion

←

Diabetes suggestion board

Normal:
no suggestion needed, just keep healthy life

Or

Impaired fasting glycaemia:
50% risk over 10 years of progressing to overt diabetes

Or

Impaired glucose tolerance:
The risk of progression to diabetes and development of cardiovascular disease is greater than for impaired fasting glucose

Or

Diabetes mellitus:
See a doctor

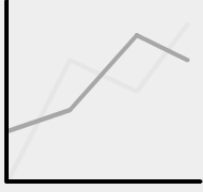
Make a record

Diabetes detection board

Diabetes suggestion board

Diabetes history board

From: / / To: / /



Suggestion history

9/20/15 normal
9/22/15

Diabetes history board

Hypertension detection board

Systolic blood pressure (SBP) in mm Hg:

Diastolic blood pressure (DBP) in mmHg:

Detect

Detection results (can be either):
Normal
or
Pre-hypertension (high-normal blood pressure)
or
State 1 hypertension
or
State 2 hypertension
or
State 3 hypertension

Get suggestion

Hypertension detection board

Hypertension suggestion board

Normal:
no suggestion needed, just keep healthy
life

Or

Pre-hypertension (high-normal blood
pressure):

Or

State 1 hypertension:

Or

State 2 hypertension:

Or

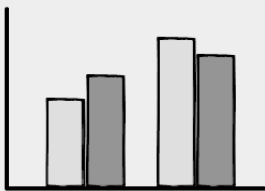
State 3 hypertension:

Make a record

Hypertension suggestion board

Hypertension history board



From: / / To: / /



Suggestion history

9/20/15 normal
9/22/15

Hypertension history board



Overweight detection board

Height:



Weight:

Exercise per week:

Detect

Detection results (can be either):
Normal
or
Underweight
or
Overweight
or
Obese

Get suggestion



Overweight suggestion board

Your BMR:
200 kcal/day

Your recommended daily kilocalorie intake:
300 kcal/day

Normal:
no suggestion needed, just keep healthy life

Or

Underweight:

Or

Overweight:

Or



Obese:

Calculate daily kilocalorie


Make a record


Overweight detection board

Overweight suggestion board




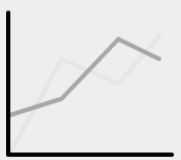
Overweight history board

From: 

To: 


BMI


Weight



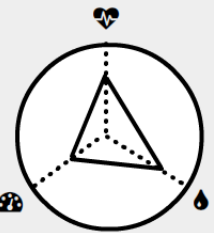
Suggestion history

9/20/15 normal
9/22/15





Food nutrition detection



Nutrition Info

Calorie

Fat

Sugar

Fiber

Carbonhydrate

Overweight history board

Food nutrition detection

The screenshot shows a mobile application interface for calculating daily kilocalorie intake. At the top, there is a back arrow icon on the left and a user profile icon on the right. The title 'Calculate your daily kilocalorie intake' is centered. Below the title, there are two columns: 'Select food' and 'Amount in gram'. Under 'Select food', there are four dropdown menus with the following options: Bread, Beef, Apple, and Broccoli. Each dropdown menu is currently set to the first option. To the right of each dropdown menu is a text input field containing the number '100'. Below these inputs, there is a section labeled 'Or' with the text 'Manually input' and a text input field. A 'Calculate' button is located below the manual input field. On the right side of the screen, there is a vertical slider bar. At the bottom of the screen, there are two lines of text: 'Your acutal kilocalorie intake:' followed by '320 kcal/day' in blue, and 'Your recommended daily kilocalorie intake:' followed by '300 kcal/day' in green. Both values are underlined.

Select food	Amount in gram
Bread	100
Beef	100
Apple	100
Broccoli	100

Or

Manually input

Calculate

Your acutal kilocalorie intake:
320 kcal/day

Your recommended daily kilocalorie intake:
300 kcal/day

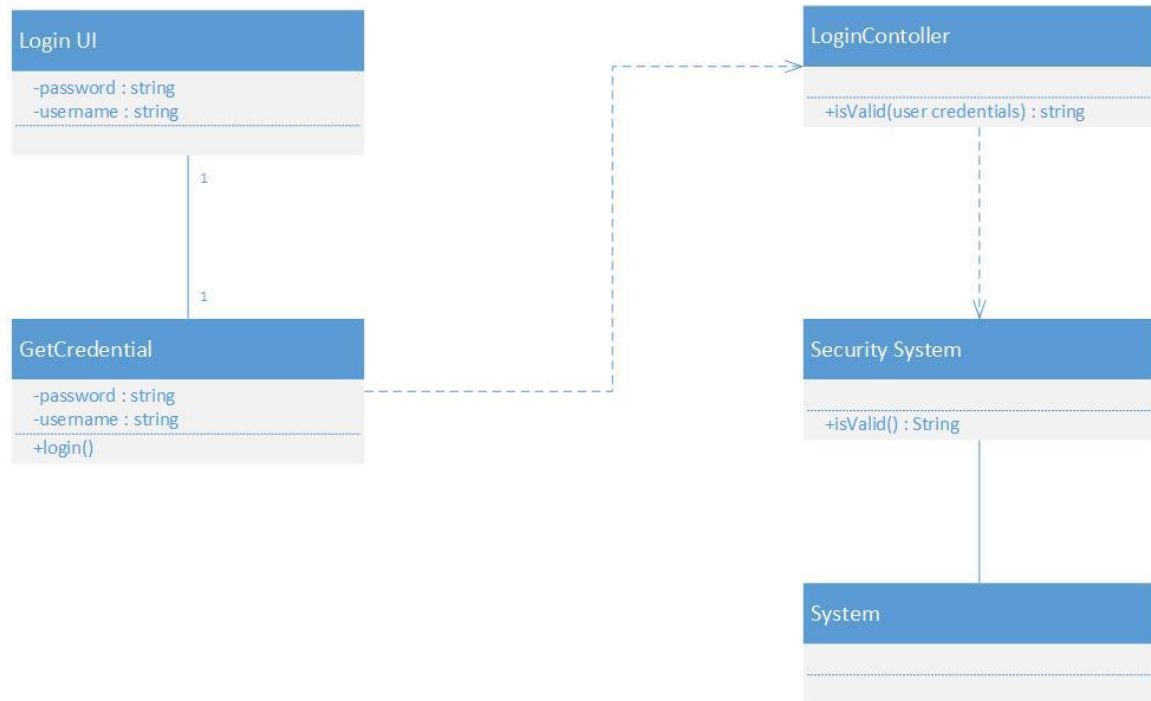
Calorie calculation board

4.2. Architecture diagram/Sequence diagram/Class diagram

4.2.1. Class diagram

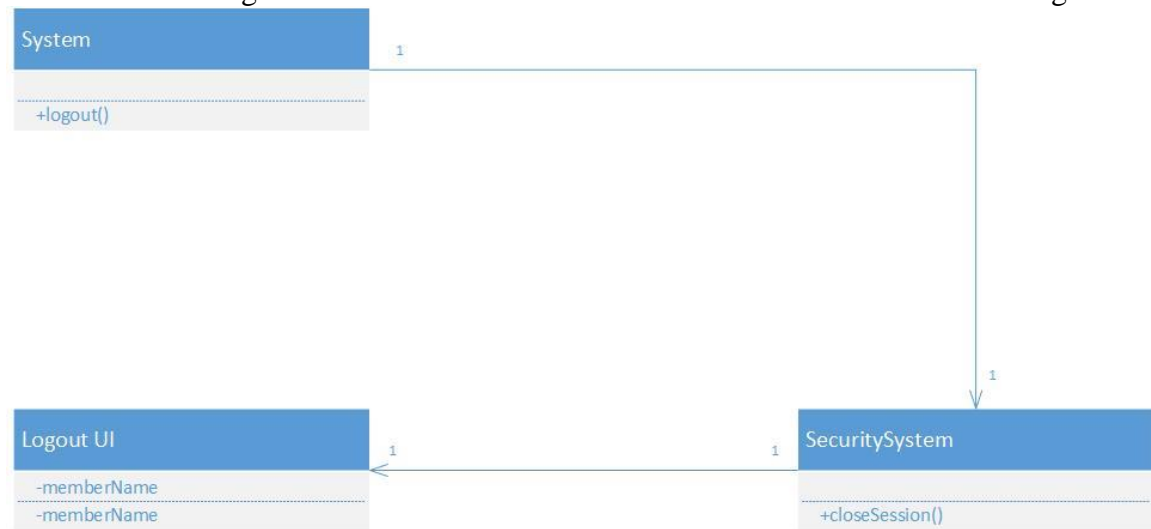
4.2.1.1. Login_Class_Diagram

This is the entry point of the project and user needs to enter the username and password to use the application. The flow goes as the user inputs its credential and the flow goes to LoginContoller class which make sure that the user is authentic by collecting credential and sending them to Security system. The Security system check the authenticity of the user and if the user is authentic, it is redirected to the home page or the systemUI of the application.



4.2.1.2. Logout_Activity_Class_Diagram

Another important aspect of the system. User decides to logout of the system, systemUI calls the security system. Security system make sure the user session is closed and session no longer exists. Once the session is closed user is redirected to the logout UI.



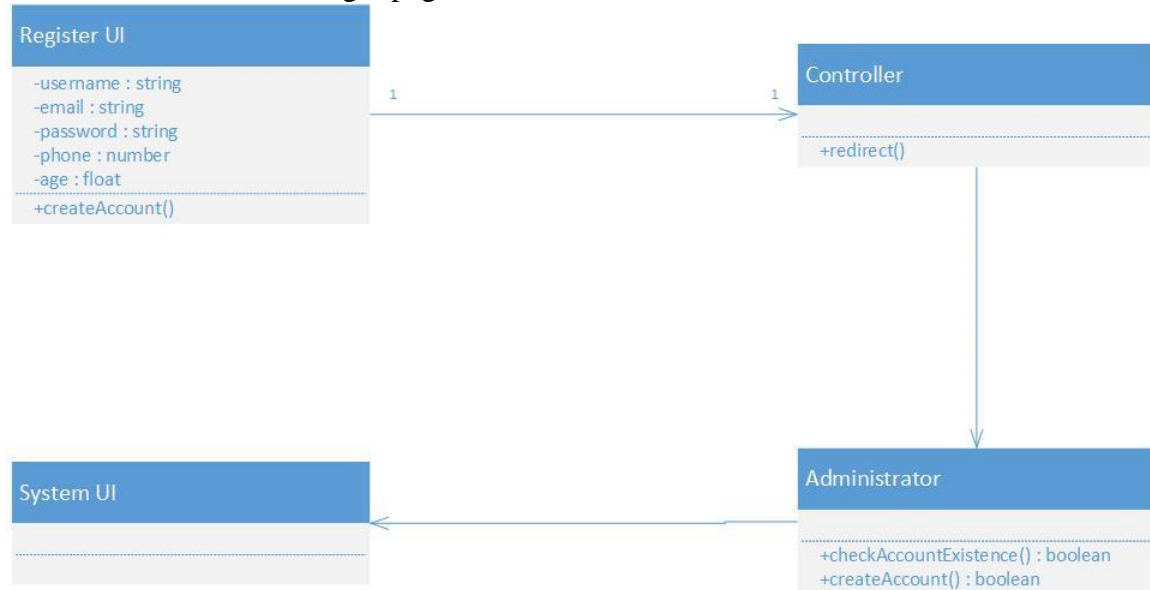
4.2.1.3. Registration_Activity_Class_Diagram

This is the process of the having user created into our system and that the user can access our 'Healthscope' application. User needs to enter the following:

- Username
- Email
- Password
- Age

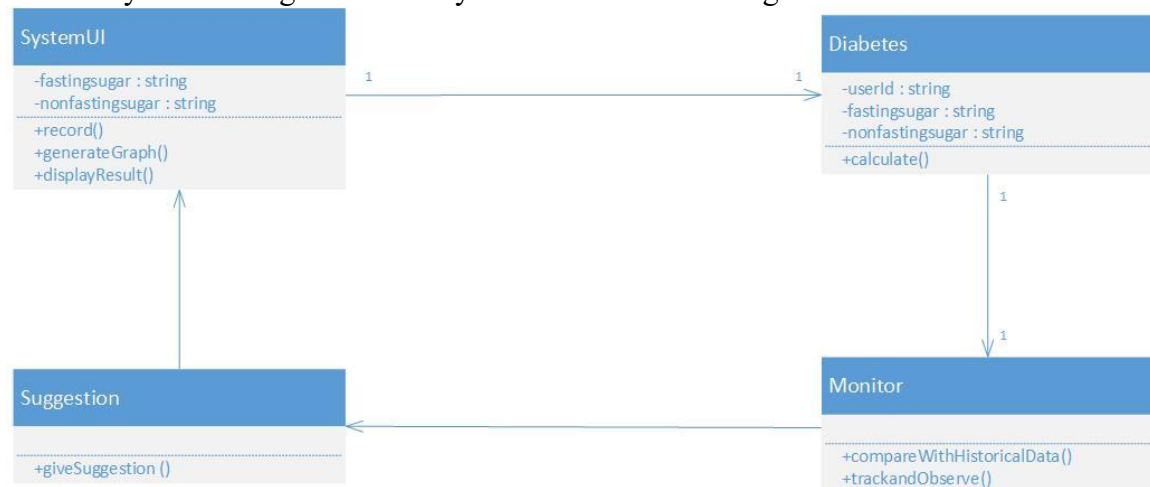
- Gender

These are the basic details user needs to enter to the system in order to get registered into the system. The controller the redirects the flow to the administrator. Admin checks if the user already exists into the system. If not, then user creates the user into the system and redirects the flow to the login page.



4.2.1.4. Diabetes_Class_Diagram

Once the user logs-in, user will be redirected to main system UI. System UI has a direct one to one mapping with the 'Diabetes' class. This 'Diabetes' class helps in the calculation of blood sugar. The variables 'fastingsugar' and 'nonfastingsugar' helps in the calculation of blood sugar and that the users enter the value for these variables. Once the user inputs the value, 'Monitor' class is invoked and the result is compared with the historical data and saved into the database. Based on the result suggestions are given to the user specifically to the degree of severity. The 'Suggestion' class does this job and eventually controller goes to the 'SystemUI' for further diagnosis.



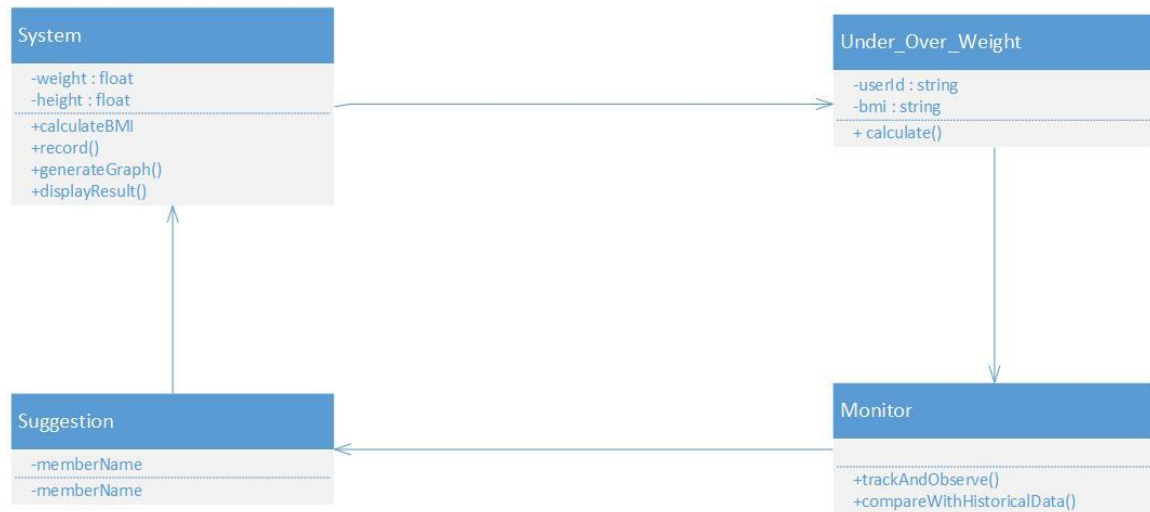
4.2.1.5. Hypertension_Class_Daigram

Once the user logs-in, user will be redirected to main system UI. System UI has a direct one to one mapping with the 'Hypertension' class. This 'Hypertension' class helps in the calculation of hypertension. The variables 'diastolicBP' and 'systolicBP' helps in the calculation of hypertension and that the users enter the value for these variables. Once the user inputs the value, 'Monitor' class is invoked and the result is compared with the historical data because hypertension needs to monitor for weeks and then the data is saved into the database. Based on the result suggestions are given to the user specifically to the degree of severity. The 'Suggestion' class does this job and eventually controller goes to the 'SystemUI' for further diagnosis.



4.2.1.6. Overweight_Underweight_Class_Daigram

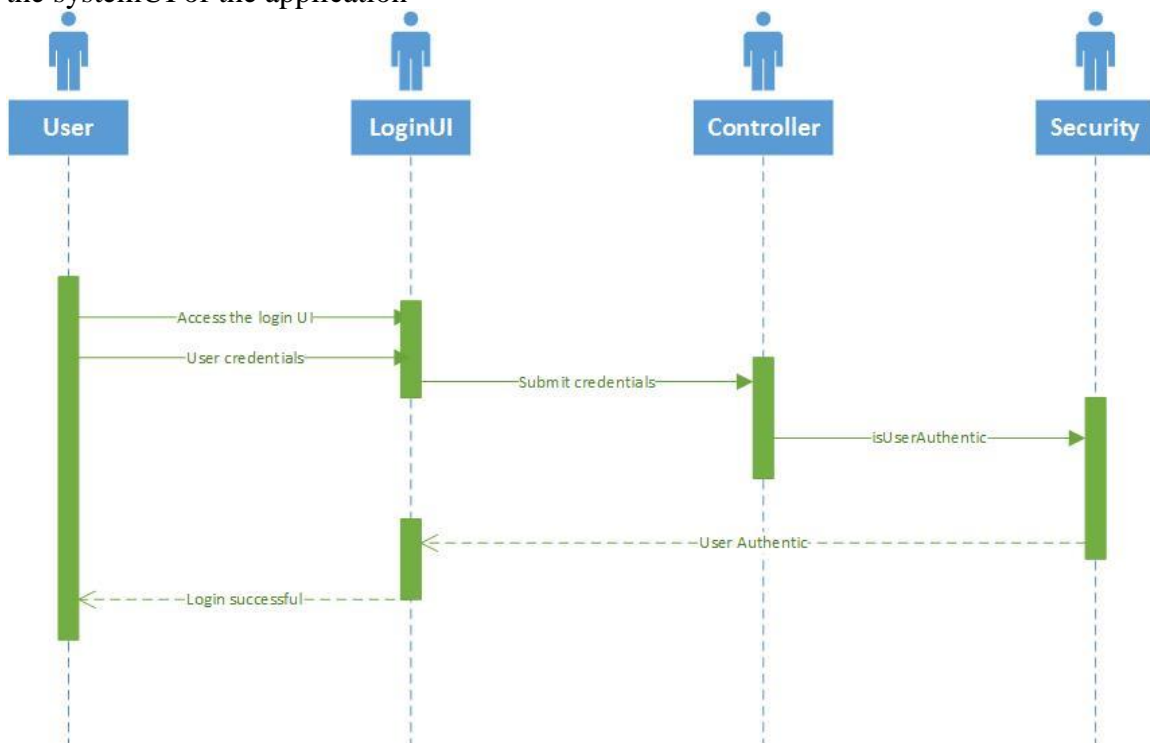
Once the user logs-in, user will be redirected to main system UI. System UI has a direct one to one mapping with the 'Under_Over_Weight' class. This 'Under_Over_Weight' class helps in the calculation of BMI which further helps in calculating overweight or underweight. The variables 'height' and 'weight' helps in the calculation of BMI and that the users enter the value for these variables. Once the user inputs the value, 'Monitor' class is invoked and the result is compared with the historical data because this process needs to monitor for weeks and then the data is saved into the database. Based on the result suggestions are given to the user specifically to the degree of severity. The 'Suggestion' class does this job and eventually controller goes to the 'SystemUI' for further diagnosis.



4.2.2. Sequence diagram

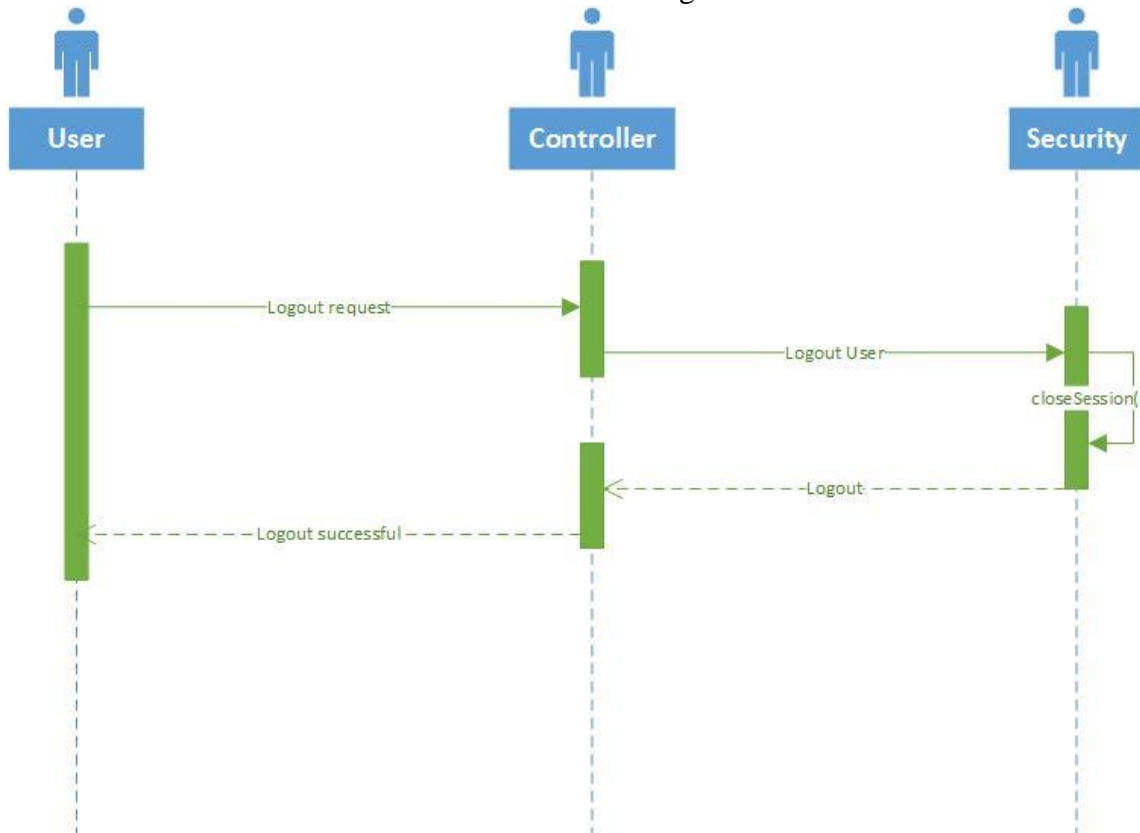
4.2.2.1. Login_Sequence_Diagram

This is the entry point of the project and user access the loginIU and needs to enter the username and password to use the application. The flow goes as the user inputs its credential and the flow goes to LoginContoller which make sure that the user is authentic by collecting credential and sending them to Security system. The Security system check the authenticity of the user and if the user is authentic, it is redirected to the home page or the systemUI of the application



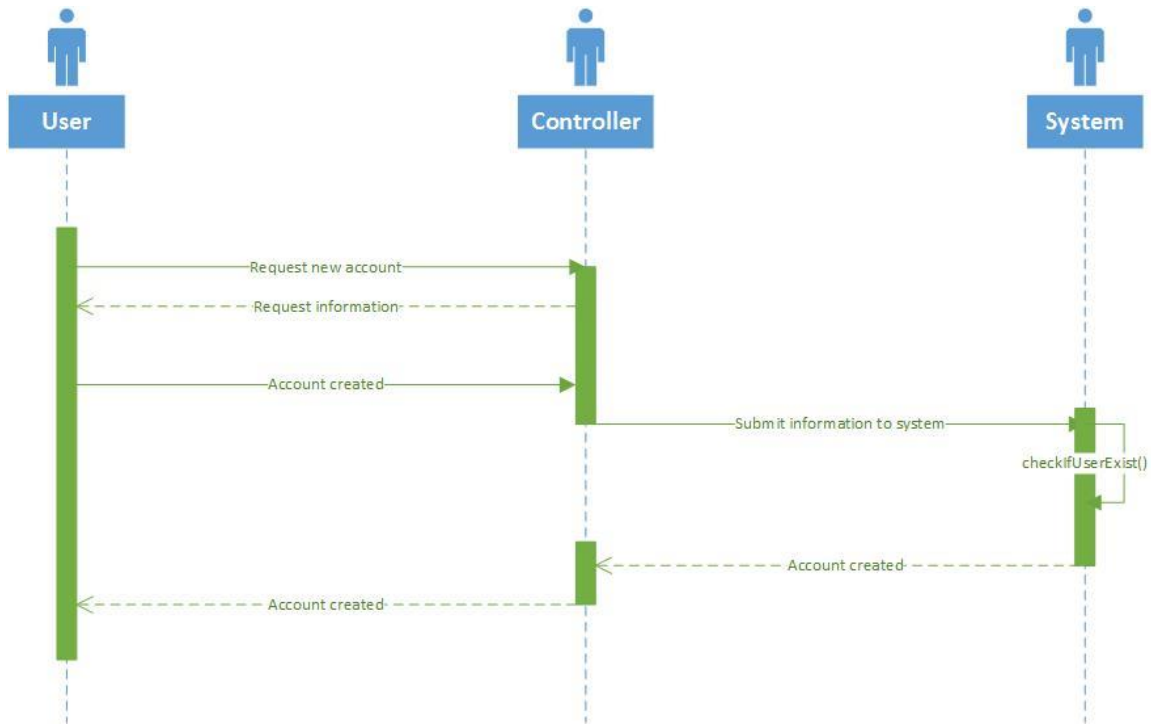
4.2.2.2. Logout_Sequence_Diagram

Another important aspect of the system. User actor decides to logout of the system by requesting the controller actor to logout. Controller actor calls the security system. Security system actor makes sure the current user session is closed and session no longer exists. Once the session is closed user is redirected to the logout UI.



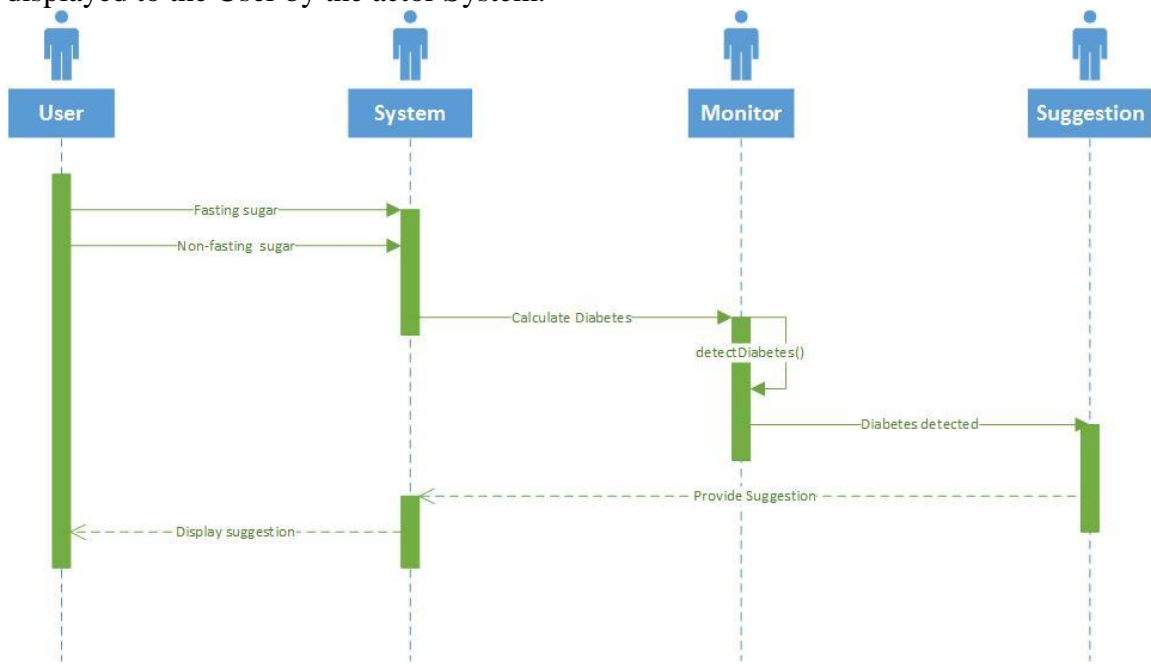
4.2.2.3. Registration_Sequence_Diagram

The actor user requests for new account registration. In reply to the request the actor System asks the user to provide the user details. User then provides the details and the actor Controller collects it and submits the information to administrator user. System admin user checks if the user already exists in the system. If not, then the user is created and registered into the system



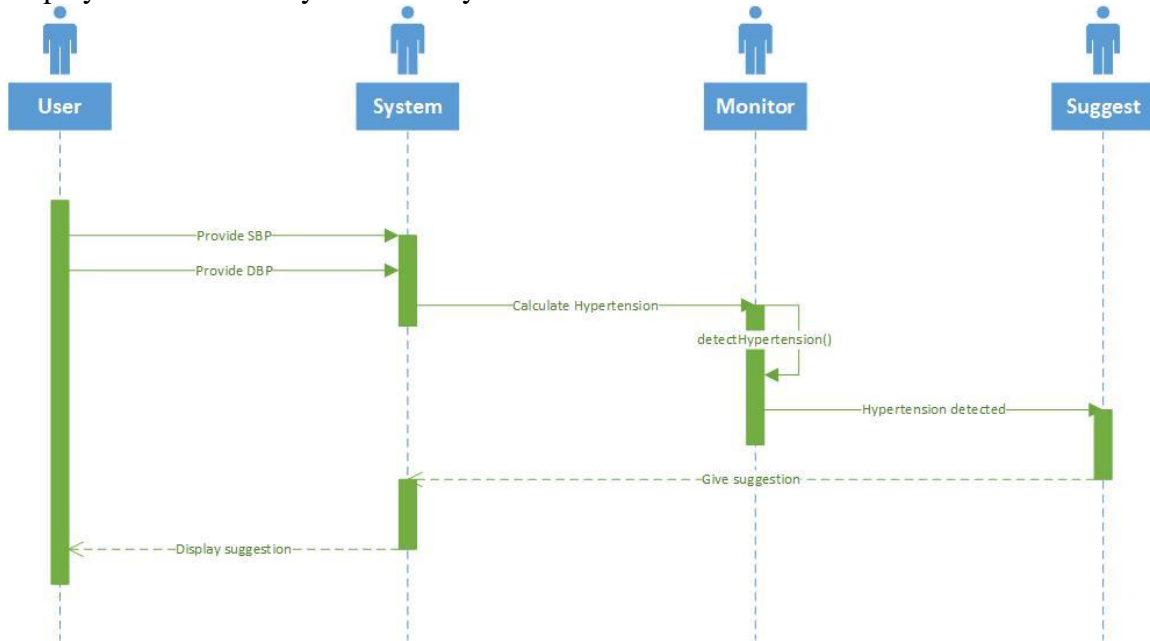
4.2.2.4. Diabetes_Sequence_Diagram

Actor User inputs the value for the variables 'fastingsugar' and 'nonfastingsugar' helps in the calculation of blood sugar. Actor System collects the input and calculates the blood sugar level. The flow is then passed to actor Monitor. In here the diabetes is checked based on the user input and historical data. If diabetes is detected the actor Suggestion sends out the suggestions to the user specifically to the degree of severity which is displayed to the User by the actor System.



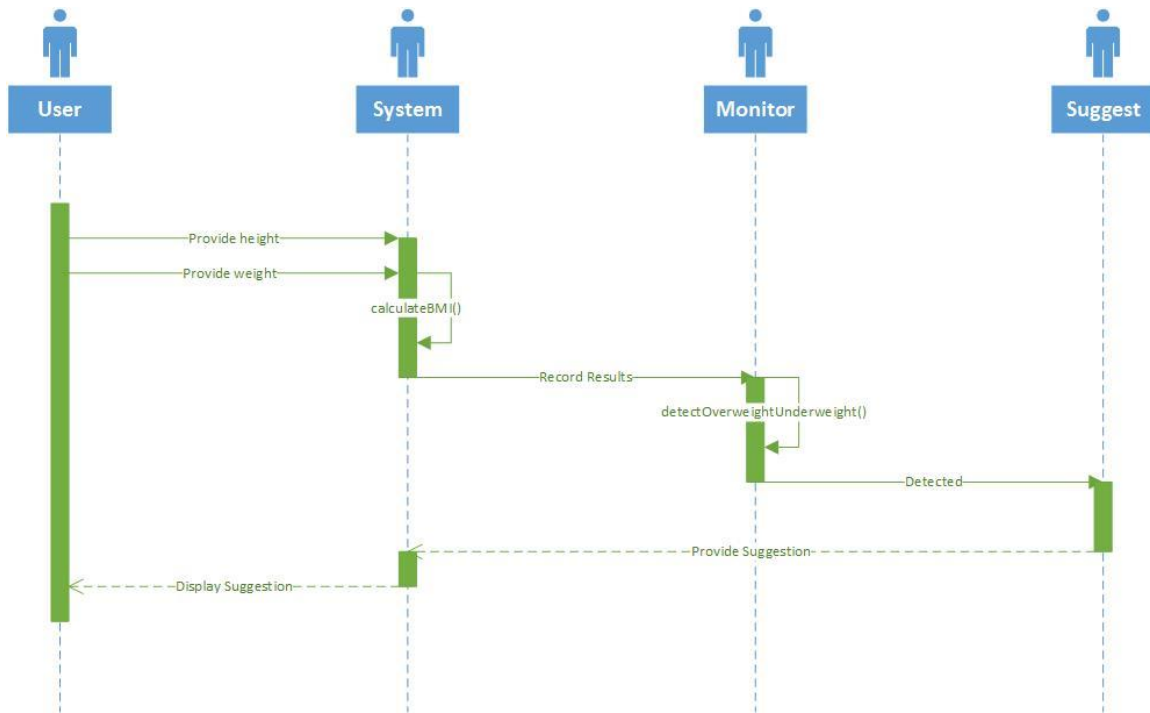
4.2.2.5. Hypertension_Sequence_Diagram

Actor User inputs the value for the variables ‘SystolicBP’ and ‘DiabloicBP’ helps in the calculation of hypertension. Actor System collects the input and calculates the hypertension. The flow is then passed to actor Monitor. In here the hypertension is checked based on the user input and historical data. If hypertension is detected the actor Suggestion sends out the suggestions to the user specifically to the degree of severity which is displayed to the User by the actor System.



4.2.2.6. Overweight_Underweight_Sequence_Diagram

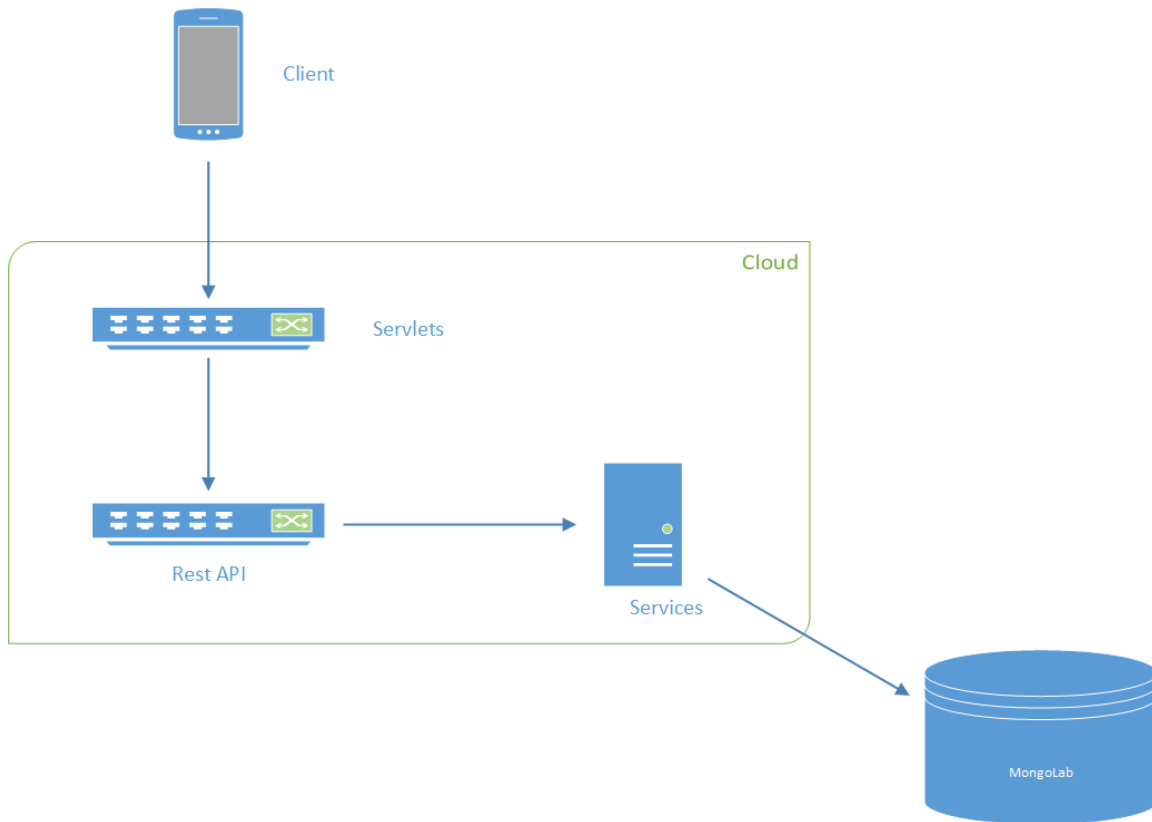
Actor User inputs the value for the variables ‘height and ‘weight’ helps in the calculation of BMI. Actor System collects the input and calculates the BMI. The flow is then passed to actor Monitor after recording the results of BMI.



4.2.3. Architecture diagram

Client (mobile app or web browser) will access the servlet by using the app's UI. The UI will call the corresponding servlet (this will use `httpdoGet`, `httpdoPost`, `httpDodelete` etc). The servlet then will call the specific service which will be either to delete, sign-in, sign-up or record the user inputs.

The service will do the corresponding operations in the database which in our case is mongoLab.



4.3. Write User Stories /Use Case/Service description

4.3.1. Registration

Users start out by creating an account. To create an account, they provide their personal information including name, email, date of birth, photo, and password. When all input information is validly provided the system will create an account in mongolab. When any information is invalid or missing, the system should display errors and no accounts should be created.

4.3.2. Login

When user provides the correct email and password, the user can login to the home page to do the choice.

4.3.3. Home page

When user logs in, the user can make the selection for three health concerns: overweight, diabetes, and hypertension.

4.3.4. Diabetes

User can input the parameters of sugar level: Sugar level of fasting glucose (mg/dl) and sugar level of 2-hour glucose (mg/dl). After clicking button “click”, the detection result and “get suggestion” button will show to the user. Also, if the user doesn’t enter the parameter correctly, there is a reminder to let user enter the values correctly. The correction parameters will be recorded to Mongolab with the detection data.

4.3.5. Hypertension

User needs to input two parameters related to blood pressure namely: systolic BP and diabolic BP. In order to determine the hypertension user needs to record these parameters over a period of three to four weeks and only one recording in a week. Once the user enters the value for SBP and DBP, hypertension UI will calculate based on the formula described in Introduction and detect whether user has hypertension or not. User will be given a advice based on the result of the diagnosis. These values are recorded and stored in the database (MongoLab)

4.3.6. Overweight

User can input the height, weight, and exercise per week. After detection, the system will tell the user about the overweight situation: normal, underweight, overweight, or obese. And the user can get the food nutrition information.

4.4. Implementation of design patterns

Factory pattern is used to provide a generic interface for creating objects. In our project, we use it to make different requests to the server of different urls with different carryData. The following is an example of factory pattern used in our project to create a user profile.

```
.factory('UserService', function($http) {
  var RequestFactory = {
    make: function(reqMethod) {
      this.tmp = {};
      this.tmp.method = reqMethod;

      return this;
    },

    requestTo: function(reqUrl) {
      this.tmp.url = reqUrl

      return this;
    },

    carryData: function(reqData) {
      var request = this.tmp;
      this.tmp = null;

      request.data = JSON.stringify(reqData);
      request.contentType = 'application/json'

      return request;
    }
  };
});
```



```

var restAPI = {

  'create': 'http://healthkeeper.mybluemix.net/api/users',
  'identify': 'http://healthkeeper.mybluemix.net/api/users',
  'hypertension': 'http://healthkeeper.mybluemix.net/api/hypertension',
  'diabetes': 'http://healthkeeper.mybluemix.net/api/diabetes'
}

return {
  'create': function(userProfile) {
    var API = restAPI['create'];
    var registerAPI = API + '/';
    var request = RequestFactory.make('POST').requestTo(registerAPI).carryData(userProfile);

    return $http(request);
  },

```

Also, login and record the parameters use these requests.

5. Testing

5.1. Unit testing

It uses karma tools to do the unit testing.

5.1.1. Login and Registration

Two unit test case for login:

- 1) It checks the identity('tester@team5.com', password: 'password') on userService.
- 2) It checks after the login is executed, if successful, should change to home page.

Two unit testings for registration:

- 1) It checks the creation on userService, expected to see the expected userProfile.
- 2) It checks after the signup is executed, if successful, should go back to landing page.

```

Cuongs-MacBook-Pro:tests cuong$ karma start tests.conf.js
19 10 2015 19:23:23.718:WARN [karma]: No captured browser, open http://localhost:9876/
19 10 2015 19:23:23.731:INFO [karma]: Karma v0.13.11 server started at http://localhost:9876/
19 10 2015 19:23:23.808:INFO [launcher]: Starting browser PhantomJS
19 10 2015 19:23:34.910:INFO [PhantomJS 1.9.8 (Mac OS X 0.0.0)]: Connected on socket -OLjdPGNCyR0yjDaAAAA with id 59080878
LOG: Object{data: [Object{name: ...}]}
PhantomJS 1.9.8 (Mac OS X 0.0.0): Executed 4 of 4 SUCCESS (0.006 secs / 0.102 secs)

```

5.1.2. Diabetes

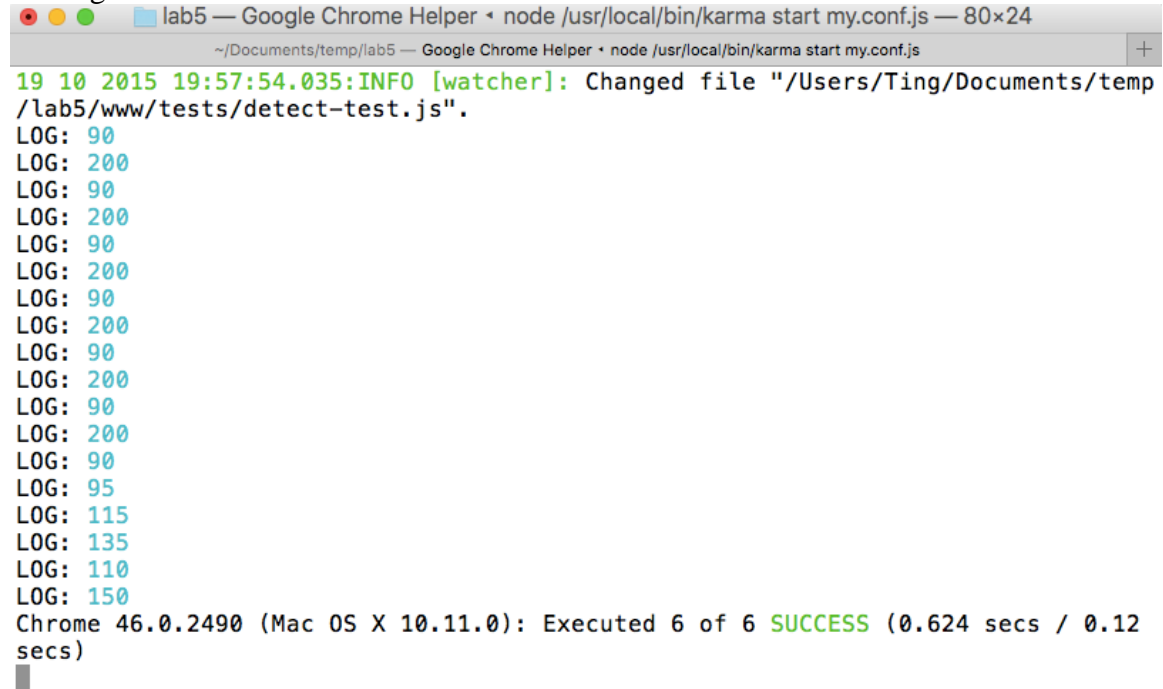
Six test cases

- 1) Sugar level (90, 200), expect to see correct “classNameForResult”--“card”
- 2) Sugar level (90, 200), expect to see correct “classNameForSuggestion”--“ bar bar-footer bar-balanced”
- 3) Sugar level (90, 200), expect to see correct “results”--“ Diabetes mellitus: A positive result, in the absence of unequivocal high blood sugar, should be confirmed by a repeat of any of the above methods on a different day. It is preferable to measure a fasting glucose level because of the ease of measurement and the considerable time commitment of formal glucose tolerance testing, which takes two hours to complete and offers no prognostic advantage over the fasting test. According to the current definition, two fasting glucose measurements above 126 mg/dl (7.0 mmol/l) is considered diagnostic for diabetes mellitus.”
- 4) Sugar level (90, 95), expect to see correct “results”--“normal”.
- 5) Sugar level (115, 135), expect to see correct “results”--“ Impaired fasting glycaemia glucose: more commonly known as pre-diabetes refers to a condition in which the

fasting blood glucose level is consistently elevated above what is considered normal levels; however, it is not high enough to be diagnosed as diabetes mellitus.”

- 6) Sugar level (110, 150), expect to see correct “results”--“ Impaired glucose tolerance(IGT): is a pre-diabetic state of hyperglycemia that is associated with insulin resistance and increased risk of cardiovascular pathology. IGT may precede type 2 diabetes mellitus by many years. IGT is also a risk factor for mortality.”

Testing results:



```
lab5 — Google Chrome Helper • node /usr/local/bin/karma start my.conf.js — 80×24
~/Documents/temp/lab5 — Google Chrome Helper • node /usr/local/bin/karma start my.conf.js
19 10 2015 19:57:54.035:INFO [watcher]: Changed file "/Users/Ting/Documents/temp
/lab5/www/tests/detect-test.js".
LOG: 90
LOG: 200
LOG: 90
LOG: 200
LOG: 90
LOG: 200
LOG: 90
LOG: 200
LOG: 90
LOG: 200
LOG: 90
LOG: 200
LOG: 90
LOG: 95
LOG: 115
LOG: 135
LOG: 110
LOG: 150
Chrome 46.0.2490 (Mac OS X 10.11.0): Executed 6 of 6 SUCCESS (0.624 secs / 0.12
secs)
```

5.1.3. Hypertension

Five test cases

- 1) Normal (no Hypertension) give input 100 and 70. Expect to see the following:
 - classNameForResult – codegreen
 - Results – “Normal : You don't have hypertension”
- 2) PreHypertension give input 120 and 85. Expect to see the following:
 - classNameForResult – codeorange
 - Results – “Pre-Hypertension detected”
- 3) Hypertension Stage 1 give input 150 and 100. Expect to see the following:
 - classNameForResult – codeorange
 - Results – “State 1 Hypertension detected”
- 4) Hypertension Stage 2 give input 170 and 115. Expect to see the following:

- classNameForResult – codedarkorange
- Results – “State 2 Hypertension detected”.

5) Hypertension Stage 3 give input 180 and 115. Expect to see the following:

- classNameForResult – codeder
- Results – “Stage 3 Hypertension (severe) detected”.

Testing result:

```
Node.js command prompt - karma start karma.conf.js
D:\ASE\HealthScope>karma start karma.conf.js
19 10 2015 21:34:18.282:WARN [karma]: No captured browser, open http://localhost:9876/
19 10 2015 21:34:18.298:INFO [karma]: Karma v0.13.11 server started at http://localhost:9876/
19 10 2015 21:34:18.298:INFO [launcher]: Starting browser Chrome
19 10 2015 21:34:20.032:INFO [Chrome 46.0.2490 (Windows 10 0.0.0)]: Connected on socket ttSQTyk1q_HNte-UAAAA with id 53123734
Chrome 46.0.2490 (Windows 10 0.0.0) LOG: 'WARNING: Tried to load angular more than once.'
Chrome 46.0.2490 (Windows 10 0.0.0): Executed 5 of 5 SUCCESS (0.008 secs / 0.124 secs)
```

5.2. Performance testing

The performance testing is done by YSlow. Following are the screenshots.

5.2.1. Registration

Home | **Grade** | Components | Statistics | Rulesets: YSlow(V2) | Edit | Help

Grade C Overall performance score 76 Ruleset applied: YSlow(V2) URL: http://localhost:8100/#/signup

ALL (23) FILTER BY: CONTENT (6) | COOKIE (2) | CSS (6) | IMAGES (2) | JAVASCRIPT (4) | SERVER (6)

B	Make fewer HTTP requests
F	Use a Content Delivery Network (CDN)
A	Avoid empty src or href
F	Add Expires headers
F	Compress components with gzip
C	Put CSS at top
B	Put JavaScript at bottom
A	Avoid CSS expressions
N/A	Make JavaScript and CSS external
A	Reduce DNS lookups
D	Minify JavaScript and CSS
A	Avoid URL redirects
A	Remove duplicate JavaScript and CSS
A	Configure entity tags (ETags)
A	Make AJAX cacheable
A	Use GET for AJAX requests
A	Reduce the number of DOM elements
A	Avoid HTTP 404 (Not Found) error
A	Reduce cookie size
A	Use cookie-free domains
A	Avoid XMLHttpRequest filter
A	Do not scale images in HTML
A	Make favicon small and cacheable

Grade B on Make fewer HTTP requests

This page has 5 external javascript scripts. Try combining them into one.
This page has 4 external stylesheets. Try combining them into one.

Decreasing the number of components on a page reduces the number of HTTP requests required to render the page, resulting in faster page loads. Some ways to reduce the number of components include: combine files, combine multiple scripts into one script, combine multiple CSS files into one style sheet, and use CSS Sprites and image maps.

[Read More](#)

Copyright © 2015 Yahoo! Inc. All rights reserved.

5.2.2. Login

Home | **Grade** | Components | Statistics | Rulesets: YSlow(V2) | Edit | Help

Grade B Overall performance score 81 Ruleset applied: YSlow(V2) URL: http://localhost:8100/#/login

ALL (23) FILTER BY: **CONTENT (6)** | **COOKIE (2)** | **CSS (6)** | **IMAGES (2)** | **JAVASCRIPT (4)** | **SERVER (6)** [Tweet](#) [Share](#)

A	Make fewer HTTP requests
F	Use a Content Delivery Network (CDN)
A	Avoid empty src or href
F	Add Expires headers
F	Compress components with gzip
A	Put CSS at top
B	Put JavaScript at bottom
A	Avoid CSS expressions
n/a	Make JavaScript and CSS external
A	Reduce DNS lookups

Grade A on Make fewer HTTP requests

This page has 5 external Javascript scripts. Try combining them into one.

Decreasing the number of components on a page reduces the number of HTTP requests required to render the page, resulting in faster page loads. Some ways to reduce the number of components include: combine files, combine multiple scripts into one script, combine multiple CSS files into one style sheet, and use CSS Sprites and image maps.

[»Read More](#)

Copyright © 2015 Yahoo! Inc. All rights reserved.

5.2.3. Diabetes

Home | **Grade** | Components | Statistics | Rulesets: YSlow(V2) | Edit | Help

Grade B Overall performance score 83 Ruleset applied: YSlow(V2) URL: http://127.0.0.1:54977/www/index.html#/diabetes

ALL (23) FILTER BY: **CONTENT (6)** | **COOKIE (2)** | **CSS (6)** | **IMAGES (2)** | **JAVASCRIPT (4)** | **SERVER (6)** [Tweet](#) [Share](#)

A	Make fewer HTTP requests
F	Use a Content Delivery Network (CDN)
A	Avoid empty src or href
F	Add Expires headers
F	Compress components with gzip
A	Put CSS at top
C	Put JavaScript at bottom
A	Avoid CSS expressions
n/a	Make JavaScript and CSS external
A	Reduce DNS lookups
C	Minify JavaScript and CSS
A	Avoid URL redirects
A	Remove duplicate JavaScript and CSS
A	Configure entity tags (ETags)
A	Make AJAX cacheable
A	Use GET for AJAX requests
A	Reduce the number of DOM elements
A	Avoid HTTP 404 (Not Found) error
A	Reduce cookie size
A	Use cookie-free domains
A	Avoid AlphaImageLoader filter
A	Do not scale images in HTML
A	Make favicon small and cacheable

Grade A on Make fewer HTTP requests

This page has 5 external Javascript scripts. Try combining them into one.

Decreasing the number of components on a page reduces the number of HTTP requests required to render the page, resulting in faster page loads. Some ways to reduce the number of components include: combine files, combine multiple scripts into one script, combine multiple CSS files into one style sheet, and use CSS Sprites and image maps.

[»Read More](#)

Copyright © 2015 Yahoo! Inc. All rights reserved.

5.2.4. Hypertension

Home | **Grade** | Components | Statistics | Rulesets: YSlow(V2) | Edit | Help

Grade B Overall performance score 87 Ruleset applied: YSlow(V2) URL: http://127.0.0.1:54598/www/index.html#/hypertension

ALL (23) FILTER BY: **CONTENT (6)** | **COOKIE (2)** | **CSS (6)** | **IMAGES (2)** | **JAVASCRIPT (4)** | **SERVER (6)** [Tweet](#) [Share](#)

A	Make fewer HTTP requests
D	Use a Content Delivery Network (CDN)
A	Avoid empty src or href
F	Add Expires headers
F	Compress components with gzip
A	Put CSS at top
A	Put JavaScript at bottom
A	Avoid CSS expressions
n/a	Make JavaScript and CSS external
A	Reduce DNS lookups
F	Minify JavaScript and CSS
A	Avoid URL redirects
A	Remove duplicate JavaScript and CSS
A	Configure entity tags (ETags)
A	Make AJAX cacheable
A	Use GET for AJAX requests
A	Reduce the number of DOM elements
A	Avoid HTTP 404 (Not Found) error
A	Reduce cookie size
A	Use cookie-free domains
A	Avoid AlphaImageLoader filter
A	Do not scale images in HTML
A	Make favicon small and cacheable

Grade A on Make fewer HTTP requests

Decreasing the number of components on a page reduces the number of HTTP requests required to render the page, resulting in faster page loads. Some ways to reduce the number of components include: combine files, combine multiple scripts into one script, combine multiple CSS files into one style sheet, and use CSS Sprites and image maps.

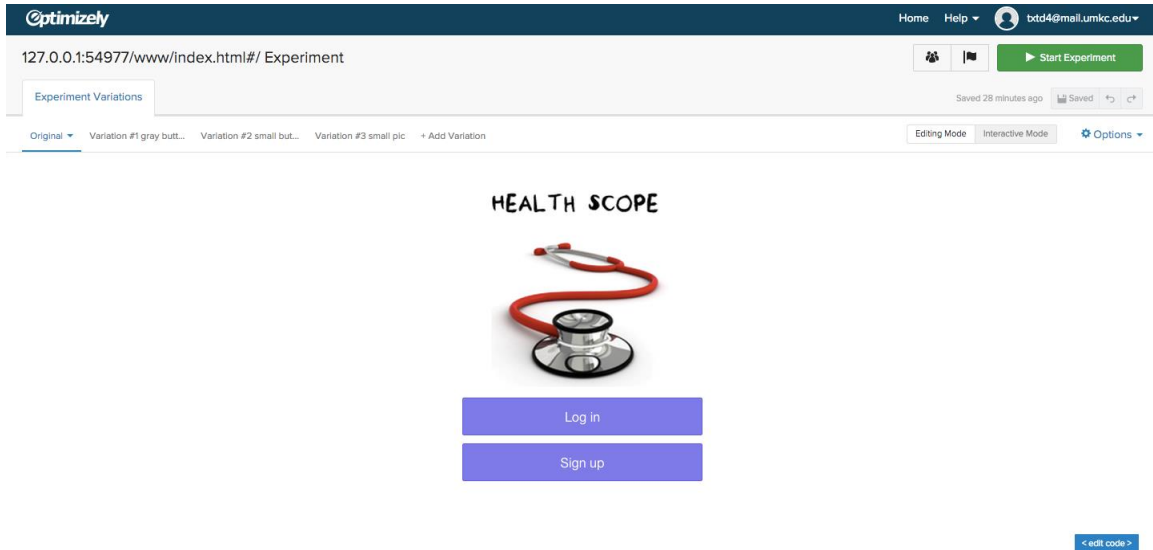
[»Read More](#)

Copyright © 2015 Yahoo! Inc. All rights reserved.

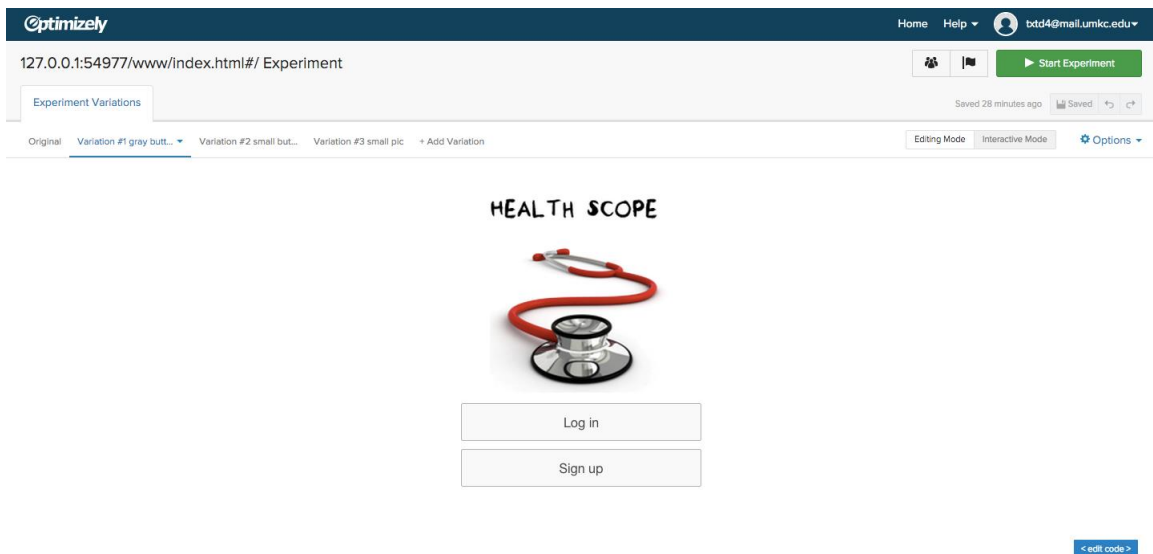
5.3. A/B testing

5.3.1. Page setting

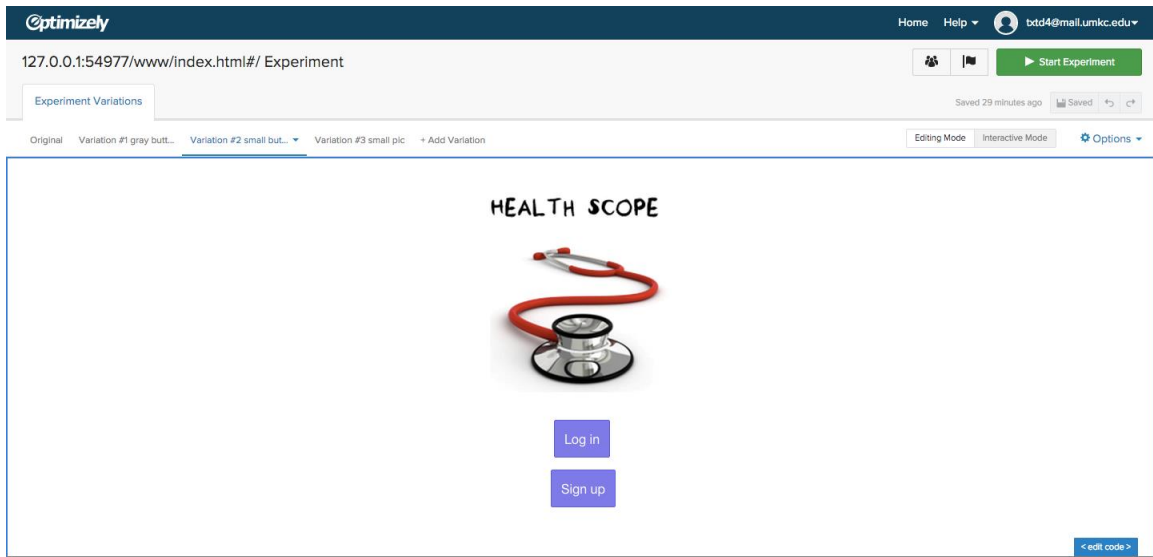
A. Original page:



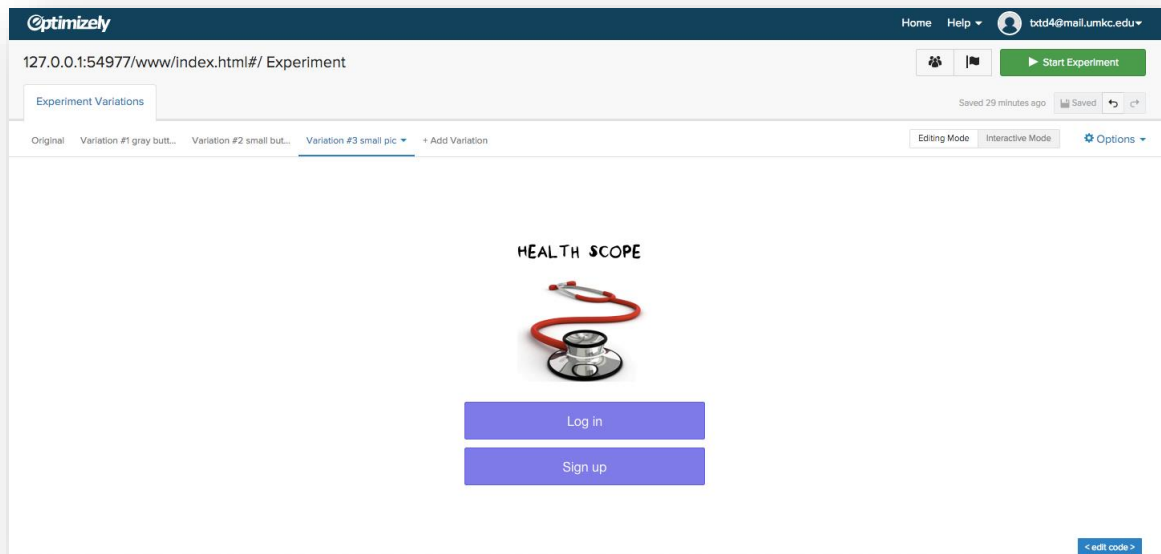
B. Variation 1--change to gray button



C. Variation 2--change to small button size



D. Variation 3-- change to small picture size



5.3.2. Snippet Implementation

Add following code to <head> tag.

```
<script src="//cdn.optimizely.com/js/3719490110.js"></script>
```

5.3.3. Goals' setting

After running the experiment:

Engagement is the default.

Also, custom goal can be added:

url: www/index.html#/login

url: www/index.html#/signup

6. Implementation (using Ionic framework /Angular.js /Bootstrap /MongoDB /Bluemix)

6.1. Ionic framework and angular.js

Ionic framework is used to build our app. Angular.js is also used. The following is the example for the code.

```
<!DOCTYPE html>
<html>

<head>
  <meta charset="utf-8">
  <link href="lib/ionic/css/ionic.css" rel="stylesheet">
  <link href="css/style.css" rel="stylesheet">

  <!-- ionic/angularjs js -->
  <script src="lib/ionic/js/ionic.bundle.js"></script>

  <!-- <script src="js/ng-cordova.min.js"></script>
  <script src="cordova.js"></script>
  <script src="js/firebase.js"></script>
  <script src="js/angularfire.min.js"></script> -->

  <!-- your app's js -->
  <script src="js/app.js"></script>
  <script src="js/controllers.js"></script>
  <script src="js/services.js"></script>

  <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no, width=device-width">
</head>

<body ng-app="app" ng-controller="Hypertension">
  <!--
    The nav bar that will be updated as we navigate between views.
  -->

  <!--
    The views will be rendered in the <ion-nav-view> directive below
    Templates are in the /templates folder (but you could also
    have templates inline in this html file if you'd like).
  -->
  <ion-nav-view></ion-nav-view>
</body>

</html>
```

6.2. Mongoddb

Mongolab is used in our app to store the user data. “PUT” “DELETE” “GET” “POST” services are used. The work can be seen in our code and demonstration.

6.3. IBM Bluemix

IBM Bluemix is used as the cloud service to store all the logic. The work can be seen in our code and demonstration. The following images just show the examples of the code (It’s not complete).

6.3.1. User creation/login

```

21 @Path("/users")
22 public class UserResource
23 {
24     private UserService userService = new UserService();
25
26     @POST
27     @Consumes(MediaType.APPLICATION_JSON)
28     @Produces(MediaType.APPLICATION_JSON)
29     public User add(User user)
30     {
31         userService.createUser(user);
32         user.setPassword(null);
33
34         return user;
35     }
36
37     @POST
38     @Path("/identify")
39     @Consumes(MediaType.APPLICATION_JSON)
40     @Produces(MediaType.APPLICATION_JSON)
41     public List<User> identify(Credential credential) throws UnsupportedEncodingException
42     {
43         if (credential.getEmail().trim().isEmpty() || credential.getPassword().trim().isEmpty()) {
44             return Collections.emptyList();
45         }
46
47         final User user = userService.findUserBy(credential.getEmail(), credential.getPassword());
48         user.setPassword(null);
49
50         return Arrays.asList(user);
51     }

```

6.3.2. Diabetes

```

14 @Path("/diabetes")
15 public class DiabetesDetection
16 {
17     private DiabetesService userService = new DiabetesService();
18
19     @POST
20     @Path("/detect")
21     @Consumes(MediaType.APPLICATION_JSON)
22     @Produces(MediaType.APPLICATION_JSON)
23     public DiabetesDiagnosis diabetes(DiabetesDiagnosis variables) throws UnsupportedEncodingException
24     {
25
26         if (variables.getId().trim().isEmpty() || variables.getSugar1().trim().isEmpty()
27             || variables.getSugar2().trim().isEmpty()) {
28             return null;
29         }
30
31         final DiabetesDiagnosis user = userService.updateUserBy(variables.getId(), variables.getSugar1(), variables.getSugar2());
32
33         return user;
34     }

```


6.3.3. Hypertension

```
13 @Path("/hypertension")
14 public class HypertensionDetection
15 {
16     private HypertensionService userService = new HypertensionService();
17
18     @POST
19     @Path("/detect")
20     @Consumes(MediaType.APPLICATION_JSON)
21     @Produces(MediaType.APPLICATION_JSON)
22     public HypertensionDiagnosis hypertension(HypertensionDiagnosis variables) throws UnsupportedEncodingException
23     {
24
25         if (variables.getId().trim().isEmpty() || variables.getSbp().trim().isEmpty()
26             || variables.getDbp().trim().isEmpty()) {
27             return null;
28         }
29
30         final HypertensionDiagnosis user = userService.updateUserBy(variables.getId(), variables.getSbp(), variables.getDbp());
31
32         return user;
33     }
}
```

7. Deployment

7.1. Github

The ionic code is put in Github.

https://github.com/cnsgcu/CS5551_Team

7.2. IBM Bluemix

Server code:

<https://hub.jazz.net/project/group9/HealthKeeper/overview>

7.3. Screenshot (mobile APP)

7.3.1. Building the project

7.3.1.1. Adding the android platform using the following commands,

```
D:\ASE\Project\HealthScope>ionic platform add android
Updated the hooks directory to have execute permissions
Downloading Default Ionic Resources
Downloading: https://github.com/driftyco/ionic-default-resources/archive/master.zip
[=====] 100% 0.0s
Done adding default Ionic resources
Adding icons for platform: android
Adding android project...
Creating Cordova project for the Android platform:
  Path: platforms\android
  Package: com.ionicframework.healthscope414638
  Name: HealthScope
  Activity: MainActivity
  Android target: android-22
Copying template files...
Android project created with cordova-android@4.1.1
Running command: "C:\Program Files\nodejs\node.exe" D:\ASE\Project\HealthScope\hooks\after_prepare\010_add_platform_class.js D:\ASE\Project\HealthScope
add to body class: platform-android
Installing "com.ionic.keyboard" for android
Installing "cordova-plugin-console" for android
Installing "cordova-plugin-device" for android
Installing "cordova-plugin-splashscreen" for android
Installing "cordova-plugin-whitelist" for android
Saving platform to package.json file
D:\ASE\Project\HealthScope>
```

7.3.1.2. Creating the APK file to deploy in mobile phone.

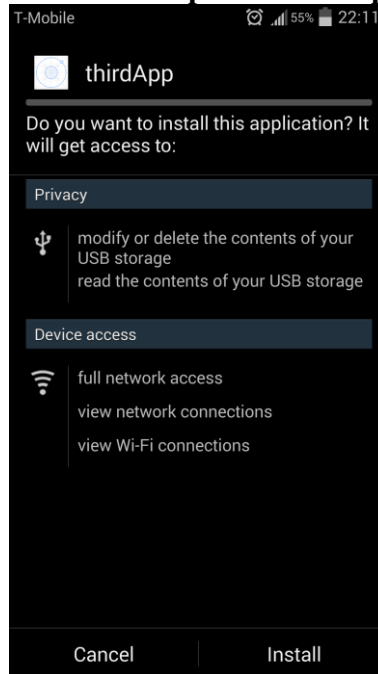
```
D:\ASE\Project\HealthScope>ionic build android
Running command: "C:\Program Files\nodejs\node.exe" D:\ASE\Project\HealthScope\hooks\after_prepare\010_add_platform_class.js D:\ASE\Project\HealthScope
Add to body class: platform-android
Running command: cmd "/s /c "D:\ASE\Project\HealthScope\platforms\android\cordova\build.bat""
ANDROID_HOME=C:\Users\tshed\AppData\Local\Android\android-sdk
JAVA_HOME=C:\Azul\zulu1.7.0_65-7.6.0.1-win64
Running: D:\ASE\Project\HealthScope\platforms\android\gradlew cdvBuildDebug -b D:\ASE\Project\HealthScope\platforms\android\build.gradle -Dorg.gradle.daemon=true

:preDexDebug
:dexDebug
:processDebugJavaRes UP-TO-DATE
:validateDebugSigning
:packageDebug
:zipalignDebug
:assembleDebug
:cdvBuildDebug

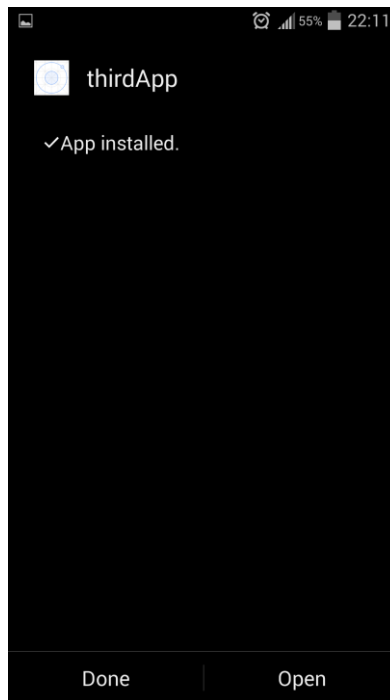
BUILD SUCCESSFUL

Total time: 37.246 secs
Built the following apk(s):
  D:\ASE\Project\HealthScope\platforms\android\build\outputs\apk\android-debug.apk
```

7.3.1.3. Copying this apk to the android phone and deploying it.

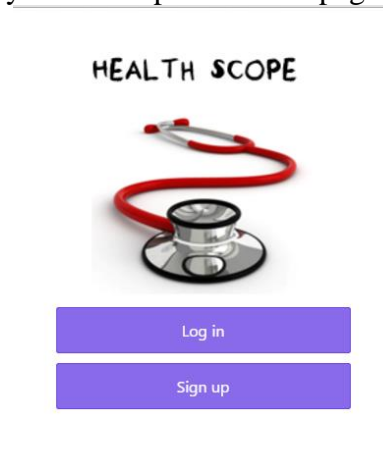


7.3.1.4. Click install.



7.3.2. Splash screen Page


Open the app and it will take you to the splash screen page (the first page of the app).





7.3.3. Registration Page


< Back


Sign Up



Photo

 User Name

 mm/dd/yyyy

 Email

 Password

 Retype password

Sign up

7.3.4. Login Page

< Back

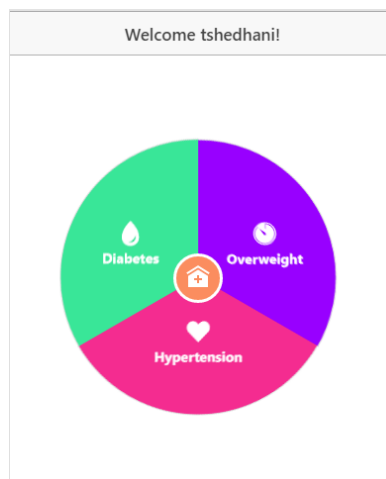
Log in

Email

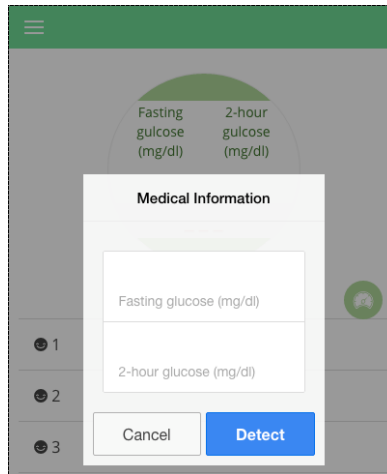
Password

Log in

7.3.5. Main page

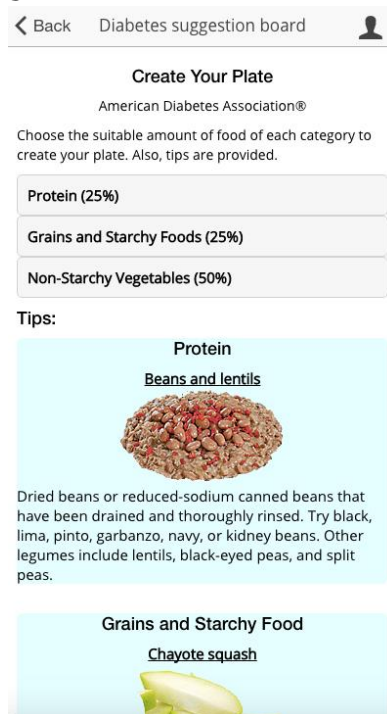


7.3.6. Diabetes UI



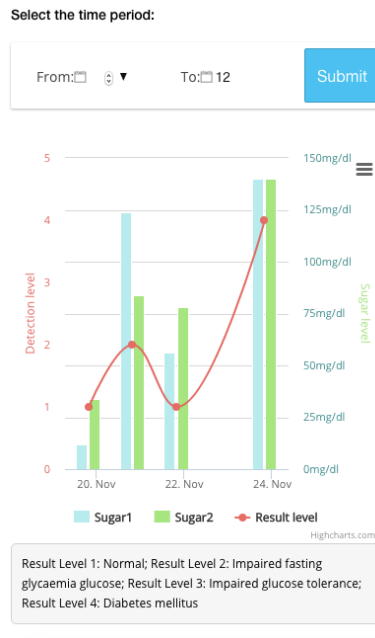
A mobile application interface for medical information. At the top, there is a green header with a hamburger menu icon. Below the header, there are two input fields for "Fasting glucose (mg/dl)" and "2-hour glucose (mg/dl)". Below these fields is a "Medical Information" form with two input fields for "Fasting glucose (mg/dl)" and "2-hour glucose (mg/dl)". At the bottom of the form are "Cancel" and "Detect" buttons. On the left side of the form, there are three radio buttons labeled 1, 2, and 3. On the right side, there is a green circular button with a plus sign.

7.3.7. Diabetes Suggestion UI

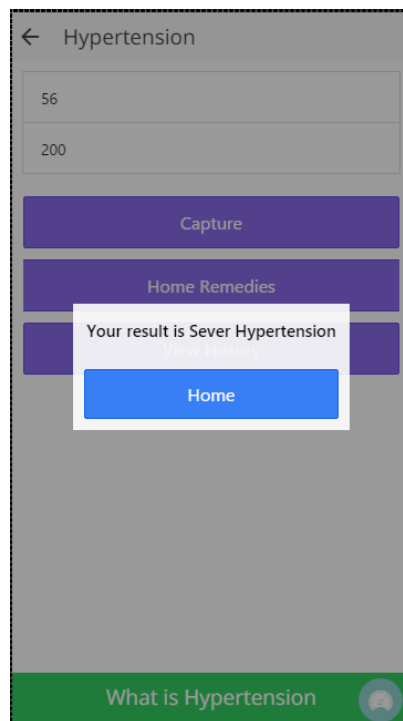


A mobile application interface for diabetes suggestions. At the top, there is a header with a back arrow, the text "Diabetes suggestion board", and a user profile icon. Below the header is a section titled "Create Your Plate" with the subtitle "American Diabetes Association®". The text below reads: "Choose the suitable amount of food of each category to create your plate. Also, tips are provided." Below this text are three input fields: "Protein (25%)", "Grains and Starchy Foods (25%)", and "Non-Starchy Vegetables (50%)". Below these fields is a "Tips:" section. The first tip is for "Protein" and "Beans and lentils", featuring an image of a bowl of beans and lentils. The text below the image reads: "Dried beans or reduced-sodium canned beans that have been drained and thoroughly rinsed. Try black, lima, pinto, garbanzo, navy, or kidney beans. Other legumes include lentils, black-eyed peas, and split peas." The second tip is for "Grains and Starchy Food" and "Chayote squash", featuring an image of a chayote squash.

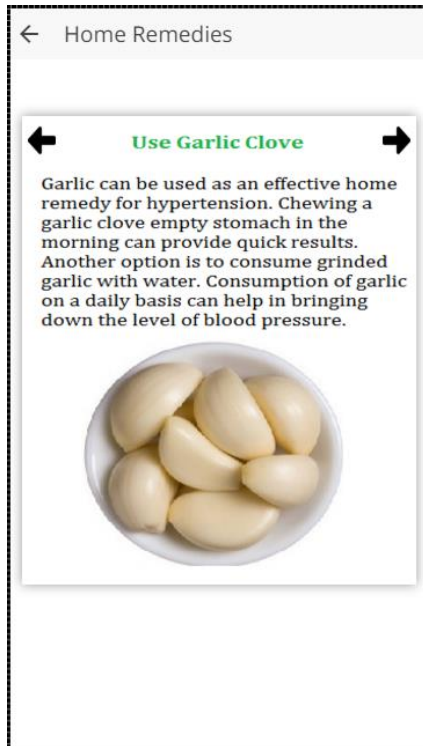
7.3.8. Diabetes History UI



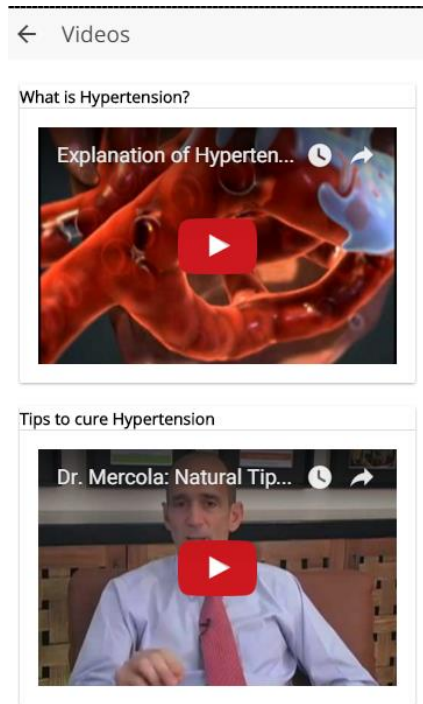
7.3.9. Hypertension UI



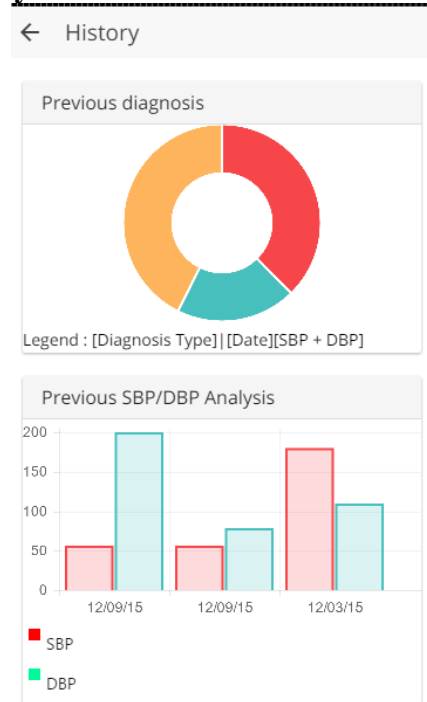
7.3.10. Hypertension Suggestion



7.3.11. Hypertension Suggestion Video UI



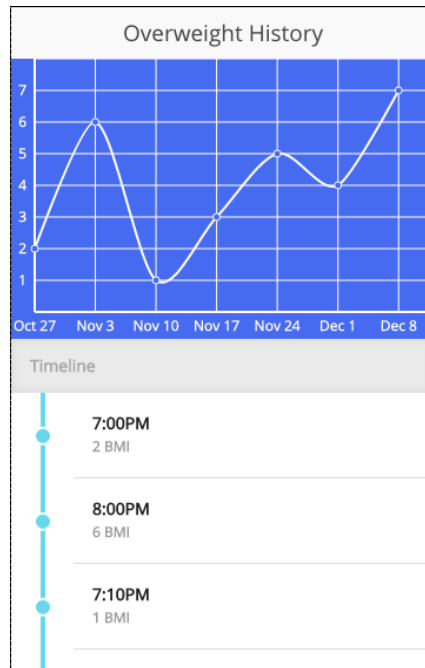
7.3.12. Hypertension history UI



7.3.13. Overweight UI



7.3.14. Overweight history UI



8. Project Management

8.1. Kanban Tool

<http://banban.kanbantool.com/b/174772-cs5551-health-app>

8.2. Tasks and Responsibilities

Work completed (6 hour/week/person)
<ol style="list-style-type: none">1. Research: hypertension (Tarun), diabetes (Ting), over-under weight (Ting)2. Set up Kanban tools and Github (Cuong)3. Mockup and wireframe (Ting and Cuong)4. Class/sequence/state/architecture diagram and description (Tarun)5. Login/Registration (Cuong)6. Homepage design (Cuong)7. Hypertension detection (Tarun)8. Diabetes detection (Ting)9. IBM Bluemix and mongolab setting and connection for registration and login (Cuong)10. Unit testing/performance testing (Tarun, Cuong, Ting)11. A/B testing (Ting)

12. User story (Tarun, Cuong, Ting)
13. IBM Bluemix and mongolab setting and connection for diabetes (Ting)
14. IBM Bluemix and mongolab setting and connection for hypertension (Tarun)
15. Hypertension suggestion (Tarun)
16. Diabetes suggestion (Ting)
17. Overweight detection (Cuong)
18. Diabetes history (Ting)
19. Hypertension history (Tarun)
20. Overweight history (Cuong)
21. Test all above cases (Tarun, Cuong, Ting)
22. Refine GUI (Tarun, Cuong, Ting)