Software Engineering I

Group 1

Health Monitor Analysis

Junjie Feng, Kai Kang, Ruotian Zhang, Tianzhe Wang, Weidi Zhang, Zhijie Zhang

https://sites.google.com/a/scarletmail.rutgers.edu/health-monitoring-analytics/?pli=1 Dec 10th 2015

Contributions

Title	Junjie Feng	Kai Kang	Ruotian Zhang	Tianzhe Wang	Weidi Zhang	Zhijie Zhang
Contribution	16.67	16.67	16.67	16.67	16.67	16.67

We unified out contribution, not in detail, this can be treated as whole project contribution of each member.

Table of Contents:

Summary of Change:

We change a lot of feature and requirement since the start of project. This summary of changes compared with report 1 and report 2. And the change related to project objectives, use case descriptions, and system design.

Project Objective:

1. In sentiment analysis, instead of applying sentiment analysis directly, we now conduct a filter first. We use TextBlob to analyze the subjectivity value of each tweet. Then we filter out all the tweets which have a subjectivity value of 0, indicating the tweet is purely objective. Tweets which have no opinion will not contributes to the sentiment analysis. Therefore filtering out tweets with 0 subjectivity value will improve the performance of sentiment analysis.
2. We give up the idea of Naive Bayes Classifiers. We found that SVM (support vector machine) has a better performance than Naive Bayes Classifiers. And we use KNN filter to filter all the tweeter, decide if it is useful or not.

3. In SVM, we no longer use the large number of features. For now, we use 'feature set' to include all the same character feature into one set. With this change, we can manage the feature well.

4. we no longer store all the feature. In previous database, we restore all the tweeters, but we find that this will make database very large after certain time. But for now, since we only care about the certain result, we just restore the result data, and delete unnecessary data, make database lighter.

Use Case Descriptions:

1. we no longer focus on the mechanism to deal with the requirement about the score board and other feature to attract customers like award. We now focus on the data analysis and information calculation to provide precise result of sports type and emotion, so that we can give certain information related to health to customers. In a word, we drop all the separate shallow requirement and just focus on the accuracy of data.

2. In the website, we add more graph, like the pie chart to show the result of certain sports involvement ratio. We also calculate the calorie and show that in bar chart. We delete the page of link with other vendor, we think these requirement no longer fit our object of project.

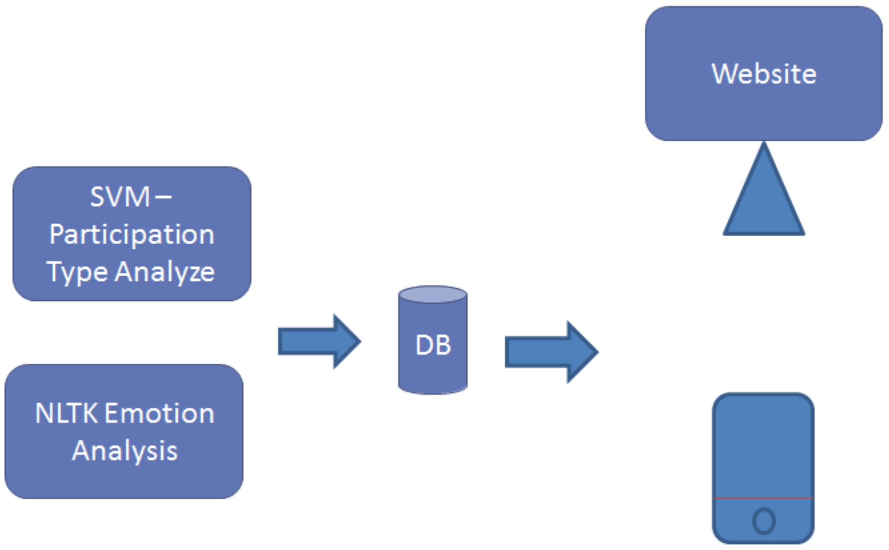
3. Since we now have different data for different sports involvement type. We add more page for this involvement type calculation. We use the proportion of certain type data to show how much popular one sports in certain area. We also can

calculate the self-involvement of certain sports and related this data to healthy. We think that the more proportion self-involve, the more healthy condition in this area.

4. We also add the page for emotion analysis. We show the proportion of emotion (like or dislike) for one sports, and analysis the popularity of certain sports. We think this data can related to the ratio of self-involvement ratio and also can has certain relationship with healthy condition. We use this information to inform customer about health.

System design:

The main change in system design is that we use website to do all the calculation using the data calculated by our SVM and NLTK emotion algorithm. And we also implement android app to show the these results. Different combination of result can show different information. So we can develop many other model in future.



1. Customer Requirement Statement

Problem Statement

With the development of modern society, people now have more and more ways to entertain themselves. There are also more and more kinds of delicious food for people to choose, you can get all kinds of sweets or protein that you can ever imagine of. It seems that people are having a happier life with so many options to enjoy themselves.

On the other hand, people need to work really hard to feed themselves and to be able to afford all the new kinds of ways of entertainment. Work pressure is so huge these days that people seems to forget the most important thing for having a happy life, which a healthy body.

No doubt can be made on the point that the word is suffering from health crisis nowadays. So many people have problems with obesity, diabetes and these problems are actually threatening their lives. Exercising is a very effective solution for those problems.

Encourage people to get involved in more kinds of sports and do more exercise is the first goal we are trying to achieve in this project. And even let's say that the health problem has been recognized by the public, but most of them just stop at being aware but lack of knowledge involving what sort of exercises shall be suitable for them to achieve personal healthy goal, where can they have access to find partners who share similar exercise habits including type, time and location, where would be satisfied sports buildings to take their preferred exercises, what are popular exercise that be taken by many people around them? Our second goal is to teach people what is best for themselves and what is the most efficient way for them to exercise, and how can they be involved with others when exercising.

So what we need to develop is a suite of applications that can encourage already enrolled users to exercise more and more, as well as encourage potential customers to getting onboard.

As a standing of an ordinary person, they do concern about first an immediate effect and then a longer effect. What we talk about immediate effect, we can use coupon of sports shops and gyms to encourage and attract customers. Customers are encouraged to post their information on twitter to indicate that they are exercising and we can use certain methods to verify this information. Once the information is verified and uploaded to our database, we can analysis and compare to the data the user did before and neighbor users' exercise data to give users certain kinds of quantity of scores to quantify their work on exercise. Coupons can be given out using this method. Thus, users are encouraged to post more and more information on websites such as twitter and make us easier to get users' information. Users are also encouraged to share this application to their friends and family members.

To attract more people to use our product, we look forward to work with companies to create a win-win situation for both the company and the exercising people. We plan to invite companies involved in the field of health and fitness to give coupons to people who uses our product and exercise hard.

We plan to gather user's data from twitter and tell them what other people are doing to keep themselves fit. So that users will not feel they are along when doing exercise, this will help users to keep exercising. The data we are going to ask from user includes the text of tweets, the user id, screen name, profile image, friends of user, and the location of tweets. And we plan to use streaming API to keep a continuous stream of public information from Twitter.

2. Glossary of terms

Application Programming Inter-face (API)

In computer programming, an application programming interface (API) is a set of routines, protocols, and tools for building software applications. An API expresses a software component in terms of its operations, inputs, outputs, and underlying types. An API defines functionalities that are independent of their respective implementations, which allows definitions and implementations to vary without compromising the interface. A good API makes it easier to develop a program by providing all the building blocks. A programmer then puts the blocks together. (https://en.wikipedia.org/wiki/Application_programming_interface)

Health Care Analysis (HCA)

Health care analytics is a product category used in the marketing of business software and consulting services. It makes extensive use of data, statistical and

Figure 2-1(HCA)

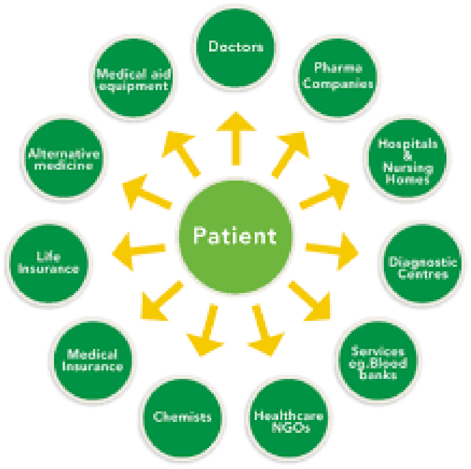


Figure 2-2

qualitative analysis, explanatory and predictive modeling. (https://en.wikipedia.org/wiki/Health_care_analytics)

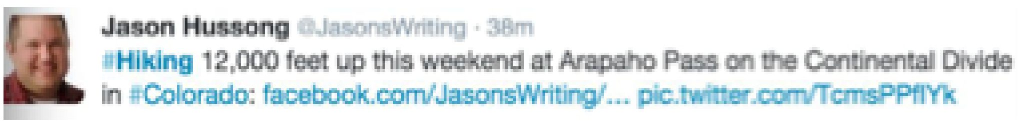
Twitter

Twitter is an online social networking service that enables users to send and read short 140-character messages called "tweets". Tweets are publicly visible by default, but senders can restrict message delivery to just their followers. Users can tweet via the Twitter website, compatible external applications, or by Short Message Service available in certain countries. (http://en.wikipedia.org/wiki/Twitter)Figure 1-3 is the logo of Twitter, Inc.

figure 2-3

Hash-tags

A hash-tag is a word or a phrase prefixed with the number sign ("#"). It is a form of metadata tag. Words or phrases in messages on micro blogging and social networking services such as Facebook, Google+, Instagram, Twitter, or VK may be tagged by entering "#" before them, either as they appear in a sentence or appended to it. The term "hash-tag" can also refer to the hash symbol itself when used within the context of reciting a hash-tag. (http://en.wikipedia.org/wiki/Hashtag) Figure 1-4 is the Tweets



Jason Hussong @JasonsWriting · 38m
#Hiking 12,000 feet up this weekend at Arapaho Pass on the Continental Divide in #Colorado: facebook.com/JasonsWriting/... pic.twitter.com/TcmsPPfIYk

with Hash-tags.

figure 2-4

Tag Cloud

A tag cloud (word cloud or weighted list in visual design) is a visual representation for text data, typically used to depict keyword metadata (tags) on websites, or to visualize free form text. The Tag Cloud could be used to show the most frequent hash-tags using the size or color of the words as a weighting factor. (https://en.wikipedia.org/wiki/Tag_cloud)

Figure 2-5

Server and Database

A server is a running instance of an application (Software) capable of accepting requests from the client and giving responses accordingly. A database is an organized collection of data. The data are typically organized to model aspects of reality in a way that supports processes requiring information. For example, modeling the availability of rooms in hotels in a way that supports finding a hotel with vacancies.(https://en.wikipedia.org/wiki/Database_server).

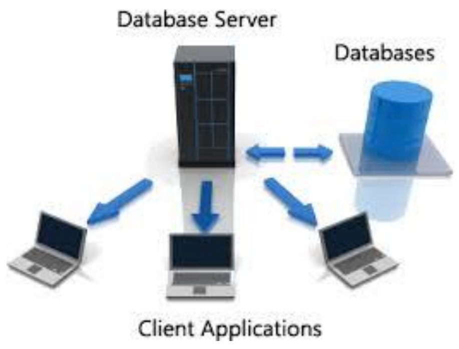


Figure 2-6

Social Networking Service

A social networking service is a platform to build social networks or social relations among people who share interests, activities, backgrounds or real-life connections (https://en.wikipedia.org/wiki/Social_networking_service).

Google Map

Google Maps is a desktop and mobile web mapping service application and technology provided by Google, offering satellite imagery, street maps, and Street View perspectives. Also supported are maps embedded on third-party websites via the Google Maps API, and a locator for urban businesses and other organizations in numerous countries around the world.(http://en.wikipedia.org/wiki/Google_Maps)

Figure 2-7

Dynamic Heat Map

A heat map is a graphical representation of data where the individual values contained in a matrix are represented as colors. Dynamic Heat Map is based on real-time data, showing the trends in the community every day in an updating frequency. (https://en.wikipedia.org/wiki/Heat_map)

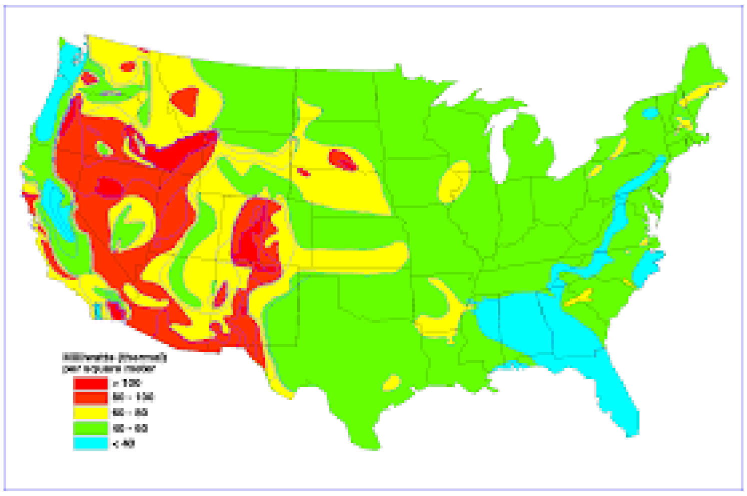


Figure 2-8.

3. System Requirements

3.1 Enumerated Functional Requirements

Identifier	Priority Weight	Requirement
REQ1	5	The system shall attain raw data from Twitter and save it in a local/online database. These Tweets should be separated according to distinct classifications of exercise.
REQ2	5	The system shall attain acquire information of Twitter users such as user ID, location, follower numbers, Tweets numbers.
REQ3	5	The system shall process raw data from the local/online database and obtain the required classification information.
REQ4	5	The system shall allow users to register and provide personalized services.
REQ5	5	The system shall show leaderboard of popular exercises and cities.
REQ6	5	The system shall show heat maps indication active level of certain exercise in specific areas.
REQ7	5	The system shall allow users to record their daily diet information and to track them over a long period of time.
REQ8	5	The system shall provide proper diet suggestions to individual users.
REQ9	5	The system shall provide food coupon to proper users.
REQ10	4	The system shall have a reward mechanism to encourage users to exercise more.
REQ11	4	The system shall have a reminder mechanism to remind users to keep to the plans.
REQ12	4	The system shall allow users to setup and modify personal profiles.
REQ13	4	The system shall notify users by email of updating information of database.
REQ14	4	The system shall allow users to search for related information of specific type of exercise.
REQ15	4	The system shall allow users to search for other users using key words such as cities and types of exercises.
REQ16	3	The system shall recommend potential partners to users when they search for related information.

REQ17	3	The system shall recommend exercise locations to users when they search for related information.
REQ18	3	The system shall recommend exercise activities to users when they search for related information.
REQ19	3	The system shall compare the popularity of different exercises in aspects of distinct time interval and cities and display the results.
REQ20	3	The system shall allow users to share information of their exercise activities to social networking websites such as Facebook.
REQ21	3	The system shall allow users to invite friends to the system by sending emails.

3.2 Enumerated Nonfunctional Requirements

Identifier	Priority Weight	Requirement
REQ22	5	The system should be easily accessible to any user through the website.
REQ23	5	The system should be able to update the information related to tweeter users.
REQ24	5	The system should have backup of important data incase of emergency.
REQ25	4	The system should have quick respond to large amount of users.
REQ26	5	All the data should have consistency retrieved from twitter only.
REQ27	5	The system should be easily maintained and can be fixed in a short of time when breaks down.
REQ28	4	The user interface of the website should have beautiful look and easy to navigate.
REQ29	3	The visualization of data should be direct, simple yet nice-looking.
REQ30	3	The system should not share any of users' information to any third party.
REQ31	3	The system should support high level of testing to find bugs.

3.3 On-Screen Appearance Requirements

Requirements

Registered user clicks on "Sign In" button to log in our application.

People who want to register can click on the “Sign Up” button. They will be leaded to signup window.

This window shows recent tweets which about exercise.

It offers both Android and IOS app to users in order to get a better user experience.

User can share our website with other SNS.

It allows the user to search for cities, communities, neighbor-hoods, relevant hash- tags, or other users.

Users can find their partner by searching location, exercise program and time.

Add leaderboard with more classifications that tweet about health in different states and city.

This window shows data analysis about the amount of people in a given area exercise and the timeline of people exercise, etc.

This window shows the frequency of exercise by different states and cities. Meanwhile, any new tweet can be found on this map.

On this page, you can find a lot of information about your entered community, like Health score, Exercise rank, Total number of twitter exercisers, Exercisers percentage, Exercise frequency, Exerciser distribution, Popular sports, devices

On this page, you can view device introduction, positive experience, negative experience, device overall grade, and comments. Clicking on the “share” button, user will be linked to a “share your comment” page. Clicking on the name in the comments module, user will be linked to this reviewer’s twitter home page.

This page is where the user manages their personal account such as changing their login or password or location, etc.

Website appearance

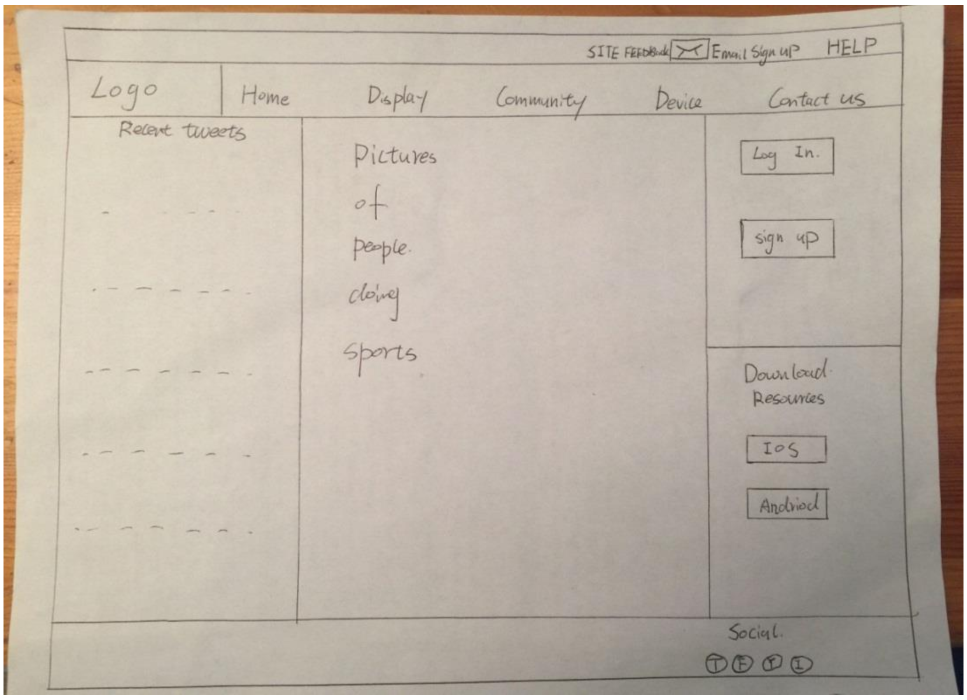


Figure 3-1

The figure 3-1 shows the homepage of our website. The users can easily see the menu: Home, Display, Community, Device and Contact Us. When clicking the Home menu, the users can see the recent tweets about health and exercise. Also, members can login to manage their personal profile. The Android and IOS apps are provided to free download. When clicking the Display menu, the users can browse several analysis charts or graphs about health and exercise. Details are shown in figure 2-2. When clicking the Community menu, members who has logged in can use the forum to contact with other members by asking or answering questions. When clicking the

Device menu, the users can see the leaderboard of device recommended by members. When clicking the Contact us menu, the users can contact us by making any suggestion or improvement.

4. Functional Requirements Specifications

4.1 Stakeholders

For the reason that our system mainly focus on providing solutions to problems involved in sports, exercise and health monitor, these people and organizations are included in the stake holders:

• **System Users**

Defined as people who would be our customers and will actually use our product. For exercise topic, they can use our system to view exercise heat map, leaderboard, score system, to find friends and gym location, to find suitable sports for individual. For health related topic they can find useful information about diet and sleep on twitter via our system, can acquire suggestions on health issues through our system, can get prize by uploading their outcome of exercise on our application.

• **Systems architects and developers**

Defined as people who designed the system who take responsibility on building the system that fulfills the user's requirements, testing and maintaining the system, and provide technical support to other stakeholders.

• **Academic Researchers**

Defined as third party, can use our system to gain information about people's health and exercise situation for academic research purpose. They can get all kinds of dates they would want from our product, without the need to do anymore investigation.

• **Gym and sports equipment advertisers**

Defined as our profit supporters, can pay to put advertisement in our system. Advertisement is restricted to healthy and exercise fields. They can also promote themselves by giving gifts or coupons to the lead runner of our product.

4.2 Actors and Goals

• **User (Initiating type)**

Goals: to interact with the system, acquire exercise and health information they need, make friends and find useful exercise places via the system.

• **Administrator (Initiating type)**

Goals: has the top priority to collect data, access, manage, and maintain the database, provide service to the user.

• **Advertiser: (Initiating type)**

Goals: analyze data, put and manage advertisement on the system to attract customers to buy their commodities.

•**Google server and database (Participating type)**

Goals: Get access to heat map using Google API.

• **Twitter server and database (Participating type)**

Goals: Download the data that needs to be analyzed from Twitter, it should be dynamic.

- **Demography text analytics server and database (Participating type)**

Goals: Should be able to analyze the text we fetched from twitter.

- **Our server and database (Participating type)**

Goals: Easy for us to use, stable.

4.3 Use Case

Causal Description

The summary use cases are as follow:

- **UC1 Data Collecting and Management**

Allow the administrator to retrieve Twitter’s data and save them in the system’s database. The administrator can look up and check the twitter raw data and the derived data anytime.

Derived from REQ 1,2,3.

- **UC2 User Registration**

Allow users to create their own account on the system to use the personalized service.

Derived from REQ 4.

- **UC3 Data Classifying**

Allow the administrator to classify the twitter raw data by some specific conditions.

Derived from REQ 2,3.

- **UC4 Leaderboard**

Allow users to see the leaderboard of popular exercises and cities.

Derived from REQ 5.

- **UC5 Heatmap**

Allow users to see the exercise heat in different area, time, exercise type and even their trends on the heatmap.

Derived from REQ 6.

- **UC6 Diet Tracking**

Allow users to record and track their daily diet information.

Derived from REQ 7.

- **UC7 Diet Suggestions**

Allow users to get the diet suggestions from the system according to their diet information.
Derived from REQ 8.

- **UC8 Food Coupon**
Allow users to download food coupon provided by the system.
Derived from REQ 9.
- **UC9 Reward Score**
Allow users to see their exercise score which they attain by continue to exercise and use our system.
Derived from REQ 10.

- **UC10 Reminder**
Allow users to receive reminders from system to remind themselves to keep on their exercise plan.
Derives from REQ 11.

- **UC11 Profile editing**
Allow users to change their own profile at anytime.
Derives from REQ 12.

- **UC12 Notification**
Allow users to receive notification via email from the system of major changing of the database and important updates.
Derives from REQ 13.

- **UC13 Searching for information**
Allow users to search information they interested in by using related keywords such as type of the exercise, cities and exercises locations.
Derived from REQ 14, 15, 17, 18.

- **UC14 Friends Recommendation**
Allow users to get potential partners recommendation from system.
Derived from REQ 16.

- **UC15 Popularity Information**
Allow users to see the comparison of the popularity of different exercises in aspects of distinct time interval and cities.
Derived from REQ 19.
- **UC16 Sharing to Social Networks**
Allow users to share their exercise experiences to their social network website such as Facebook and Twitter.
Derived from REQ 20.

- **UC17 Inviting friends**

Allow users to invite their friends to our system and can get reward score if their friends register our system.

Derived from REQ 21.

- **UC18 Third Party API**

Allow the administrator to get verification of accessing and retrieving data from Twitter for further data analysis.

Derives from REQ 1,2.

- **UC19 Data Deleting**

Allow the administrator to delete data or adjust database if necessary.

Derives from REQ 1,2.

- **UC20 Display**

Allow user to access all the functions using graphic user interface.

Derives from REQ 7,12,15.

4.4 Use case diagram

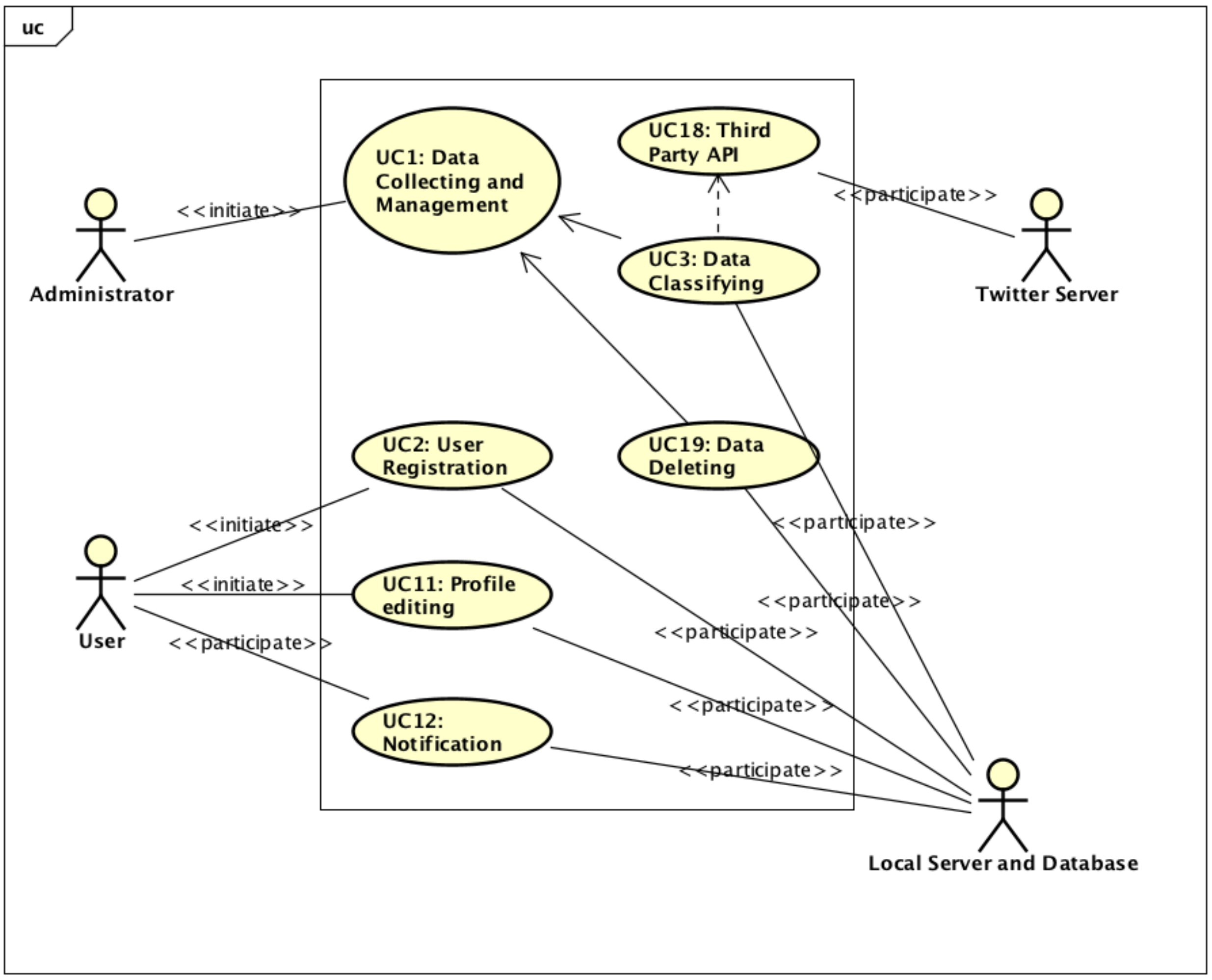


Figure 1-1 Use case diagram of server and database management subsystem

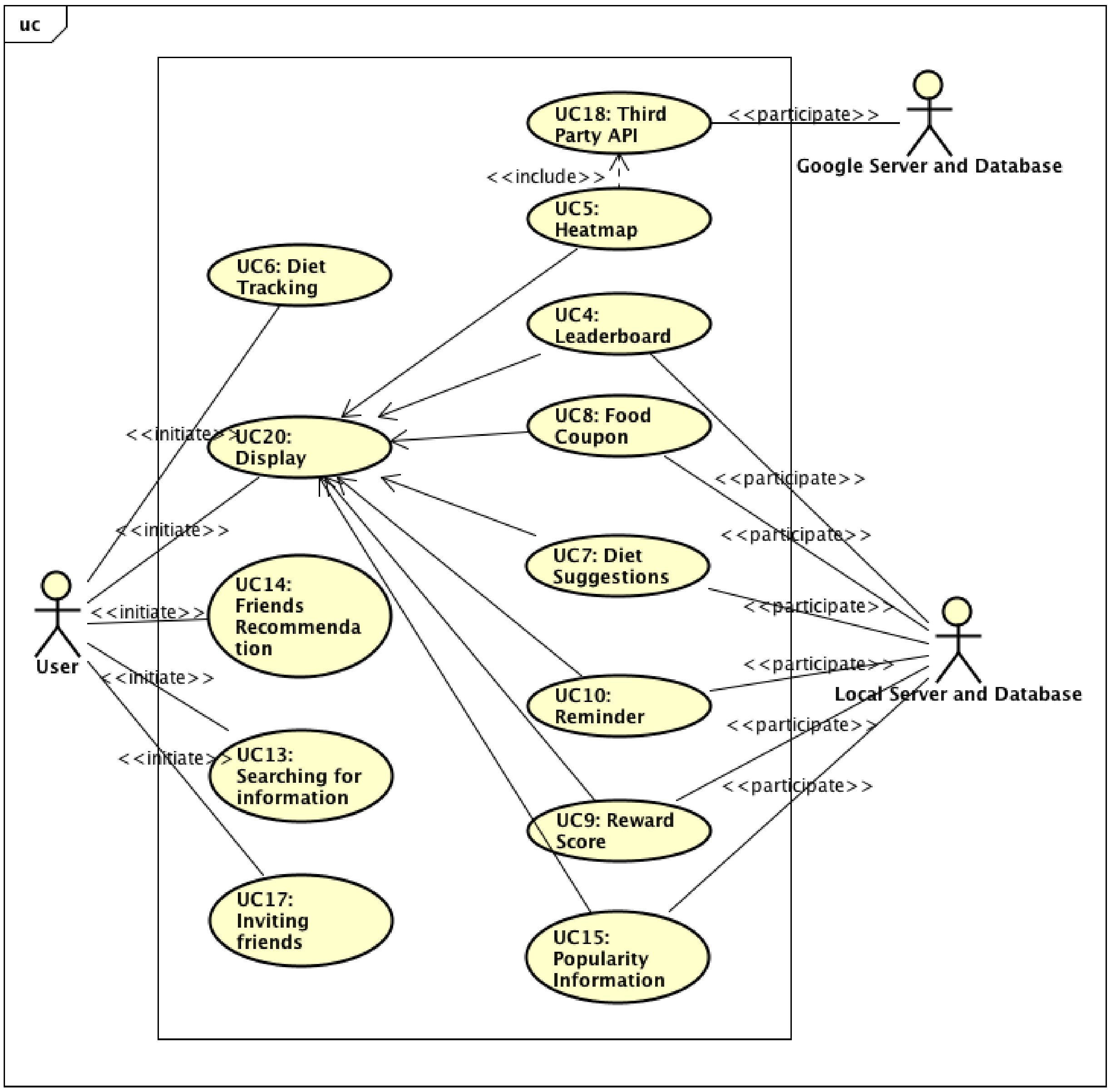
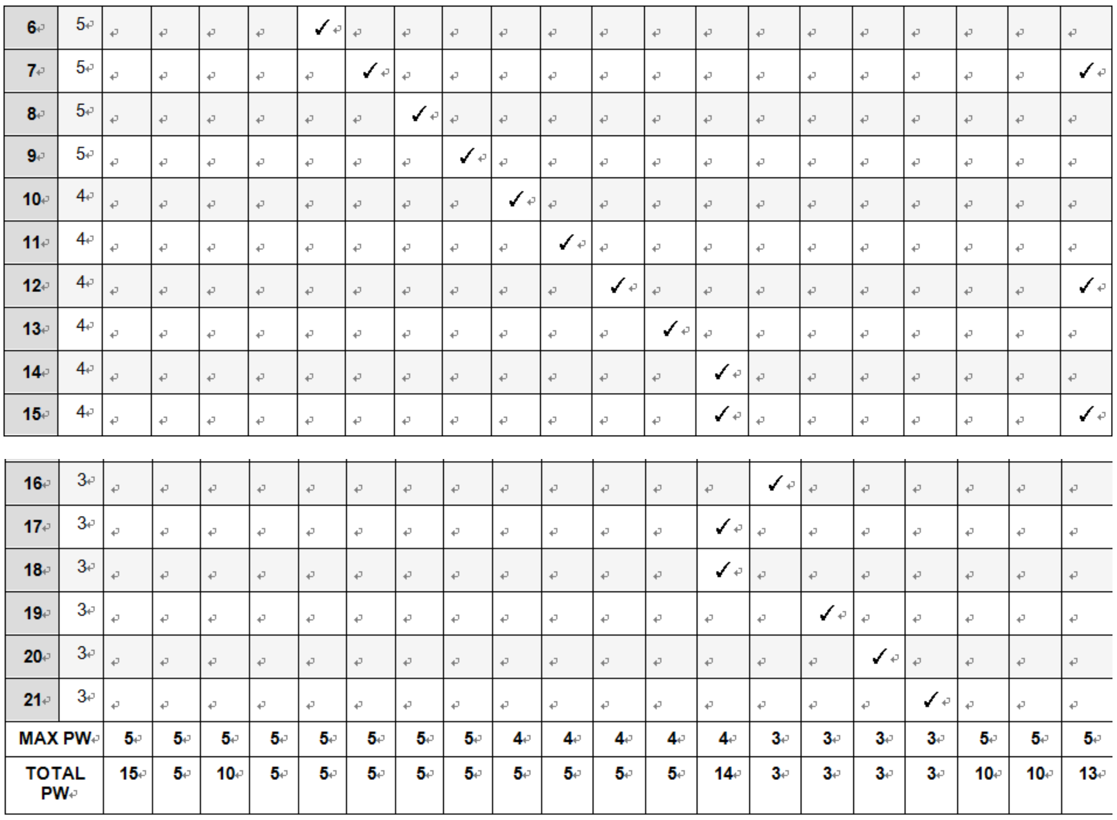


Figure 1-2 Use case diagram of displaying subsystem

4.5 Traceability Matrix

REQ	PW	UC-1	UC-2	UC-3	UC-4	UC-5	UC-6	UC-7	UC-8	UC-9	UC-10	UC-11	UC-12	UC-13	UC-14	UC-15	UC-16	UC-17	UC-18	UC-19	UC-20
1	5	✓																	✓	✓	
2	5	✓		✓															✓	✓	
3	5	✓		✓																	
4	5		✓																		
5	5				✓																



6	5					✓															
7	5						✓														✓
8	5							✓													
9	5								✓												
10	4									✓											
11	4										✓										
12	4											✓									✓
13	4												✓								
14	4													✓							
15	4													✓							✓
16	3														✓						
17	3													✓							
18	3													✓							
19	3															✓					
20	3																✓				
21	3																	✓			
MAX PW		**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**4**	**4**	**4**	**4**	**4**	**3**	**3**	**3**	**3**	**5**	**5**	**5**
TOTAL PW		**15**	**5**	**10**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**14**	**3**	**3**	**3**	**3**	**10**	**10**	**13**

Table 1-1 Traceability matrix

4.6 Fully-Dressed Description

Use Case UC9	Reward score
Related Requirements	REQ 10
Initiating Actor	Developer
Actor's Goal	Allow users to see their exercise score which they attain by continuing to exercise and use our system.
Participating Actors	System's Server & Database
Pre-conditions	Related information is stored in the system's database
Post-conditions	The reward as coupons gives to the users by the exercise score rank in their region
Flow of events for main success scenario: The user navigated to the main interface of the system The user logs into the system using their username and password	

The database gives out the statistical result of the users’ score rank in the local region

Table 1-3 Fully-dressed description UC9

Use Case UC13&20	Search for information and display
Related Requirements	REQ 7,12,14,15,17,18
Initiating Actor	Developer
Actor’s Goal	Allow users to search information they interested in by using related keywords and show the result in graphics.
Participating Actors	System’s Server & Database
Pre-conditions	Related information is stored in the system’s database
Post-conditions	System displays the chosen diagrams on the screen
Flow of events for main success scenario: The users navigated to the main interface of the system The users choose certain distribution and type to see the statistical graphs The system displays the diagrams as requested	

Table 1-4 Fully-dressed description UC13&20

4.7 System Sequence Diagrams

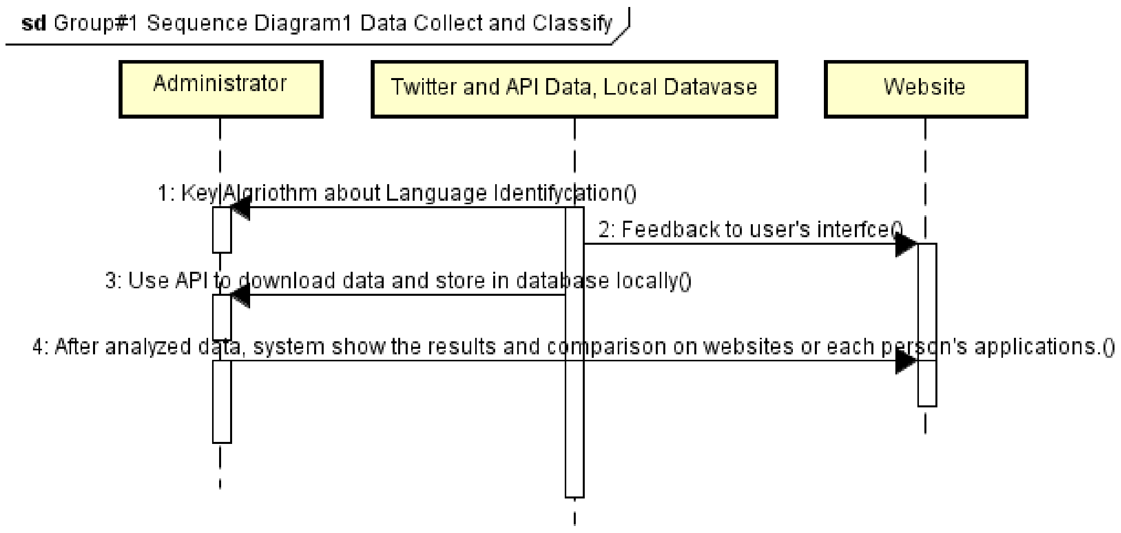

sd Group#1 Sequence Diagram1 Data Collect and Classify
Administrator
Twitter and API Data, Local Datavase
Website
1: Key Algriothm about Language Identifycation()
2: Feedback to user's interfce()
3: Use API to download data and store in database locally()
4: After analyzed data, system show the results and comparison on websites or each person's applications.()

Figure 1-3 Sequence Diagram 1

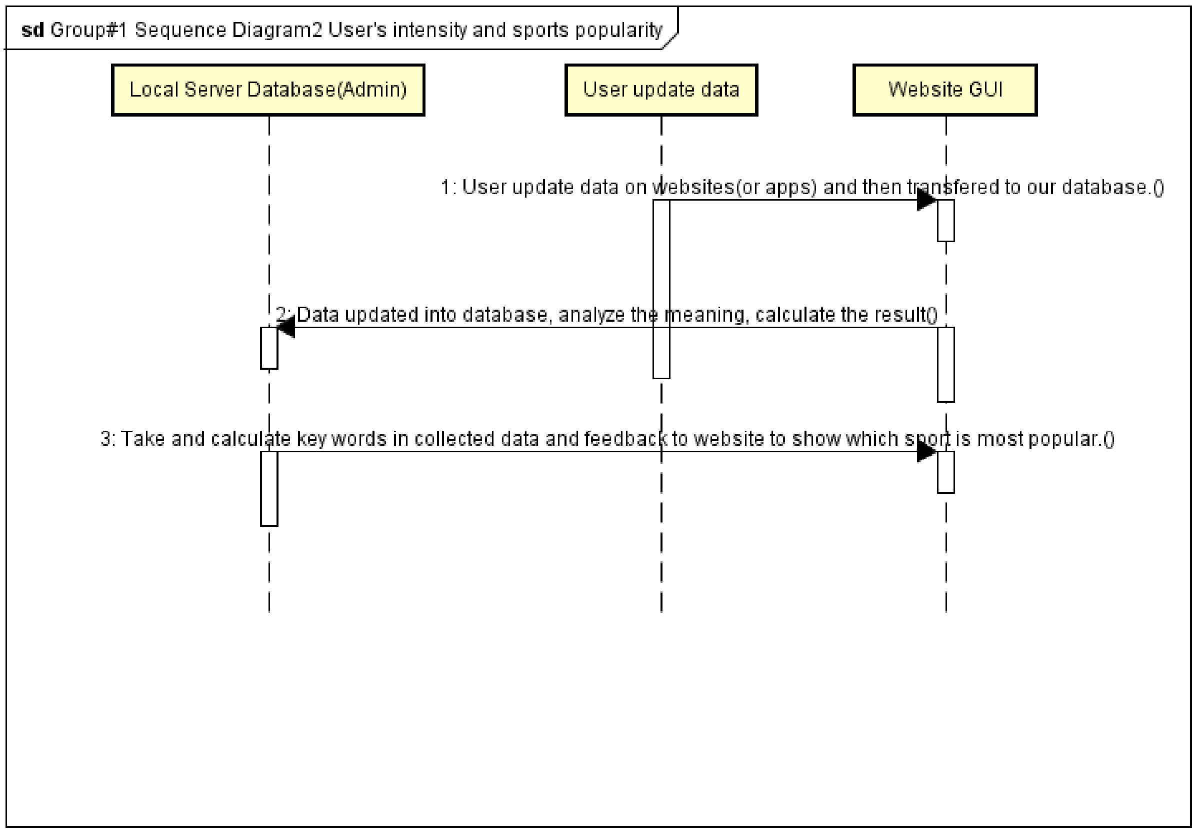

sd Group#1 Sequence Diagram2 User's intensity and sports popularity
Local Server Database(Admin)
User update data
Website GUI
1: User update data on websites(or apps) and then transfered to our database.()
2: Data updated into database, analyze the meaning, calculate the result()
3: Take and calculate key words in collected data and feedback to website to show which sport is most popular.()

Figure 1-4 Sequence Diagram 2

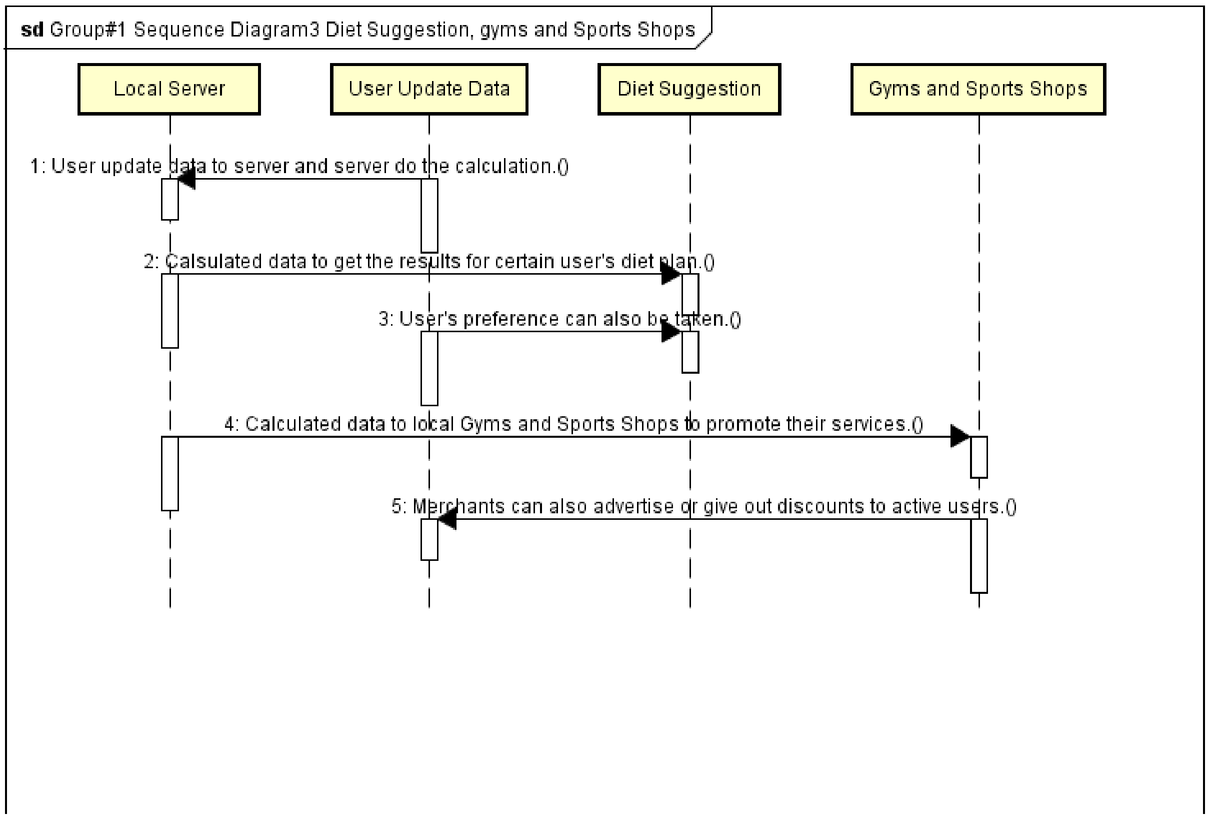

sd Group#1 Sequence Diagram3 Diet Suggestion, gyms and Sports Shops
Local Server
User Update Data
Diet Suggestion
Gyms and Sports Shops
1: User update data to server and server do the calculation.()
2: Calsulated data to get the results for certain user's diet plan.()
3: User's preference can also be taken.()
4: Calculated data to local Gyms and Sports Shops to promote their services.()
5: Merchants can also advertise or give out discounts to active users.()

Figure 1-5 Sequence Diagram 3

4.8 User Interface Description

We will give the proposed key user interface in this section, also it will contain the Preliminary Design and data analysis.

Users will log into the website's 'personal info' page to see their long term exercise and diet influence their body. The health condition may include weight, height, body fat rate and sleep condition.

In the social section, Users would find someone who has similar interests and schedules to work out together. They want to find someone if they search for specific keywords. For example, if they type in "Jogging", a list of people who love jogging will show up on the screen.

Users also can learn others' experience in the 'experience' page if they are just begin their sports

or who have specific exercise goal, hey would like to read experience and suggestions from those who are expertise on sports and those who have reached their exercise goals.

On the mobile phone app, Users can easily look for a gymnasium and to see their location and reviews from other Users. Users also can see the parks and playground nearby to find a better place to do exercise.

Users log in the website's 'personal schedule' section, after they enter their height, weight, hours for exercise per week, a specific, detailed and pertinent exercise plan will be provided their schedule.

In the 'bulletin' in the website Users may share their reviews of different sports equipment, such as, Yoga mats, sports shoes and sports legging, etc. So they can communicate about the sports equipment and make a proper purchasing decision.

Users can simply share the information through tweeter and facebook on the app on the phone, since Users may want to share their exercise activities and plans with their friends online so that they can get more suggestion and encouragement.

Information above is the main features for our website and apps on the phone, more features would be added in as stated in the system requirements.

Users effort estimation

We need to estimate the effort the customer puts into the system to obtain the required results.

Typical usage scenario 1: user wants to log in our website

NAVIGATION:

a. Click "Login" input field

b. Type User name

c. Type password

d. Click "Login" button

Typical usage scenario 2: user wants to sign up our system.

NAVIGATION:

a. Click "Sign up" Button

b. Type User Name, Email, Password,

c. Click "Submit" key

Typical usage scenario 3: log in user wants to change his password.
NAVIGATION:
a. Click “Change Password” button
b. Type User Name, Password, New Password, Confirm Password
c. Click “Submit” button
Typical usage scenario 4: user wants to see the activity partners in certain area
NAVIGATION: total 4mouse clicks and 3keystrokes
a. Click “search for partner” link
b. Type in the sports you want to play
c. Choose the area you want to find
d. Click “Search” button to see the overview information
Typical usage scenario 5: user wants to share the sports experience to his Twitter or Facebook friends and share through email.
NAVIGATION: total 7 mouse clicks
a. Click “Twitter Share” button on the upper right corner
b. Click “Tweet” button on the pop-up website
c. Click “Facebook Share” button on the upper right corner
d. Click “Share Link” button on the pop-up website
e. Click ”Email Share” button on the upper right corner
f. Click “Send” button on the pop-up website.

5. Effort Estimation using Use Case Points

Use Case Points (UCP) method provides the ability to estimate the person-hours a software project requires based on its use cases. UCP method analyzes the use case actors, scenarios, nonfunctional requirements, and environmental factors and joins them in a simple equation: CP = UUCP ＊ TCF ＊ ECF.

Unadjusted Use Case Points (UUCP) － Measures complexity of functional requirements
Technical Complexity Factor (TCF) － Measures complexity of nonfunctional requirements.
Environmental Complexity Factor (ECF) － Assesses development teams experience and heir development environment

5.1 Unadjusted use case points

UUCP are computer as a sum of the following two components:
Unadjusted Actor Weight (UAW) － Combined complexity of all the actors in all the use cases
Unadjusted Use Case Weight (UUCW) － Total number of activities contained in all the use case scenarios
UUCP = UAW + UUCW

5.2 Unadjusted actor weight

The weights for Actor classification are computed via the following table: Actor classification and associated weights, Actor Classification for Health Monitoring Analytics System

Actor	Description of Characteristics	Complexity	Weight
User	Interact with the system, acquire exercise and health information they need	Complex	5
Administrator	Has the top priority to collect data,access,manage, and maintain the database, provide service to the user	Average	3
Advertiser	Analyze data, put and manage	Simple	2

	advertisement on the system to attract customers to buy their commodities.		
Twitter server and database	Twitter server and database is another system interacting through application programming interface.	Average	3
Demography text analytics server and database	Demography text analytics server is another system interacting through application programming interface.	Average	3
Our server and database	Database is another system interacting through a protocol.	Complex	5

UAW(Health Monitoring Analytics) = 1 ∗ Simple + 3 ∗ Average + 2∗ Complex

=2∗1 + 3∗3 + 5∗2 = 21

5.3 Unadjusted use case weight

The weights for Actor classification are computed via the following table: Actor classification and associated weights, Actor Classification for Health Monitoring

Use case	Description of Characteristics	Complexity	Weight
Data Collecting and Management(UC1)	Allow the administrator to retrieve Twitter’s data and save them in the system’s database. The administrator can look up	Complex	15

	and check the twitter raw data and the derived data anytime.		
User Registration (UC2)	Allow users to create their own account on the system to use the personalized service.	Simple	5
Data Classifying(UC3)	Allow the administrator to classify the twitter raw data by some specific conditions.	Complex	15
Leaderboard(UC4)	Allow users to see the leaderboard of popular exercises and cities	Average	10
Heatmap(UC5)	Allow users to see the exercise heat in different area, time, exercise type and even their trends on the heatmap	Complex	15
Diet Tracking(UC6)	Allow users to record and track their daily diet information	Complex	15
Diet Suggestion(UC7)	Allow users to get the diet suggestions from the system according to their diet information	Complex	15
Reminder(UC10)	Allow users to receive reminders from system to remind themselves to keep on their exercise plan.	Average	10
Profile Editing(UC11)	Allow users to change their own profile at anytime.	Simple	5
Notification(UC12)	Allow users to receive notification via email from the system of major changing of the database and important updates.	Simple	5
Searching for information(UC13)	Allow users to search information they interested in by using related keywords such as type of the exercise, cities and exercises locations.	Complex	15
Friends Recommendation(UC 14)	Allow users to get potential partners recommendation from system.	Simple	5
Popularity Information(UC15)	Allow users to see the comparison of the popularity of different exercises in	Average	10

	aspects of distinct time interval and cities.		
Sharing to Social Network(UC16)	Allow users to share their exercise experiences to their social network website such as Facebook and Twitter.	Average	10
Inviting Friends(UC17)	Allow users to invite their friends to our system and can get reward score if their friends register our system.	Average	10
Third Party APPI(UC18)	Allow the administrator to get verification of accessing and retrieving data from Twitter for further data analysis.	Simple	5
Data Deleting(UC19)	Allow the administrator to delete data or adjust database if necessary.	Simple	5
Display(UC20)	Allow user to access all the functions using graphic user interface.	Average	10

UUCW(Health Monitoring Analytics)= 6*Simple + 6*Average + 6*Complex = 6*5+6*10+6*15=180

UUCP(Health Monitoring Analytics)= UAW + UUCW=21+180=201

5.4 Technical complexity factor

Technical Complexity Factor (TCF) is computed using thirteen standard technical actors to estimate the impact of productivity of the nonfunctional requirements for the application. We then need to assess the perceived complexity of each technical factor in the context of the project. A perceived complexity value is between 0 and 5, with 0 meaning trivial effort, 3 meaning average effort and 5 meaning major effort. Each actor's weight is then multiplied by perceived complexity factor to produce calculated factor. Two constants are used with TCF. The constants utilized are C1 = 0.6 and C2 = 0.01.

Technical complexity factors for Health Monitoring Analytics: PC = Perceived Complexity, CF = Calculated Factor

Description of Characteristics	Weight	PC	CF

User expects good performance, but will tolerate network latency	3	3	3
Relatively simple to add new features	1	2	2
Internal processing required multiple interactions with other subsystems	1	4	4

TCF = 0.6 + (0.01 * 28.5) = 0.885
This results in a decrease of the UCP by 11.5%

5.5 Environment complexity factor

Technical Complexity Factor (TCF) is computed using thirteen standard technical actors to estimate the impact of productivity of the nonfunctional requirements for the application. We then need to assess the perceived complexity of each technical factor in the context of the project. A perceived complexity value is between 0 and 5, with 0 meaning trivial effort, 3 meaning average effort and 5 meaning major effort. Each actor’s weight is then multiplied by perceived complexity factor to produce calculated factor. Two constants are used with TCF. The constants utilized are C1 = 0.6 and C2 = 0.01.

Technical complexity factors for Health Monitoring Analytics: PC = Perceived Complexity, CF = Calculated Factor

Description of Characteristics	Weight	PC	CF
User expects good performance, but will tolerate network latency	3	3	3
Relatively simple to add new features	1	2	2
Internal processing required multiple interactions with other subsystems	1	4	4

TCF = 0.6 + (0.01 * 28.5) = 0.885
This results in a decrease of the UCP by 11.5.

5.6 Environment complexity factor

The Environment Complexity Factor (ECF) is computed utilizing eight standard environmental factors to measure the experience level of the people on the project and stability of the project. We then need to assess the perceived impact based on perception that factor has on projects success. 1 means strong negative impact, 3 is average and 5 means strong positive impact.

TCF is computed utilizing thirteen standard technical factors to estimate the impact of productivity of the nonfunctional requirements for the application. We then need to assess the perceived complexity of each technical factor in the context of the project. A perceived complexity value is between 0 and 5 with 0 meaning that it is irrelevant, 3 means average effort and 5 means major effort. Each factors weight is then multiplied by perceived complexity factor to produce calculated factor. Two constants are used with CF. the constants utilized are C1 = 1.4 and C2 = -0.03.

Environmental Complexity Factors for Health Monitoring Analytics:

Description of Characteristics	Description	PC	CF
Beginner familiarity with UML-based development	1	1.5	1
Some knowledge of object-oriented approach	2	1	
Highly motivated overall	1	3	3
Used new programming languages but resources were readily available	-1	3	3

ECF = 1.4 – (0.03 * 12) = 1.04

This results in an increase of UDP by 4%.

5.7 Calculating the use case points

As mentioned earlier, UCP = UUCP × TCF × ECF.

From above calculations, UCP variables have the following values:

UUCP = 164

TCF = 0. 885

ECF = 1.04

UCP = 164 × 0.885 × 1.04 =150.95 or 151 use case points.

5.8 Deriving project duration from use-case points

UCP is a measure of software size. We need to know the teams rate of progress through the use cases. We need to utilize the UCP and Productivity Factor (PF) to determine duration. The equation for computing Duration is: Duration = UCP×PF

Productivity Factor is the ratio of development person-hours needed per use case point. Assuming a PF = 28, the duration of our system is computed as follows:

Duration=UCP X PF=151*28 = 4228

4228 person-hours for the development of the system.

We have 6 developers in our team, and each developer on average spent 20 hours per week on project tasks. This means that team makes 6 * 20 = 120 hours per week.

Dividing 4228 person-hours by 120 hours per week, we obtain the total of approximately 35 weeks to complete this project.

6. Domain Analysis

6.1 Domain Model

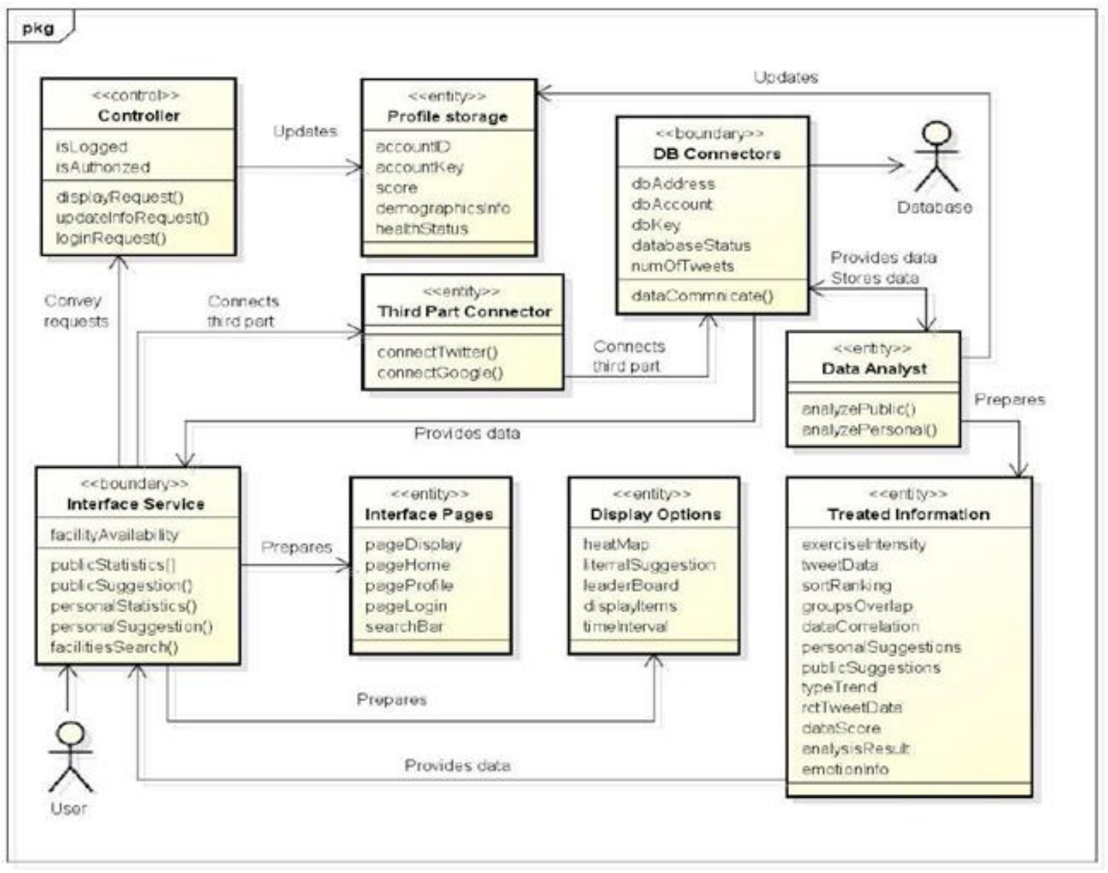


Figure 5-1 Domain model diagram

6.1.1 Concept definitions

We derive the domain model concepts from detailed user cases. Table 5-1 lists the responsibilities and the assigned concepts.

We derive the domain model concepts from detailed user cases. Table 5-1 lists the responsibilities and the assigned concepts.

Responsibility	Type	Concept
R1: Prompt the user to make movement for available services.	D	Controller
R2: Handle requests from users.	D	Controller
R3: Deal with login issue (check users key, approve or deny).	D	Controller

R4: Container for the collection of valid keys and account profile associated with users.	K	Profile Storage
R5: Show pages for user to create account, login and logout.	D	Interface Service
R6: Show search engine and diversity display buttons for user to choose.	D	Interface Service
R7: Display related information in literal, numerical, graphical and map forms.	D	Interface Service
R8: Static websites and mobile phone interface that shows the user the current context, what services could be used, and outcomes of the previous request.	K	Interface Pages
R9: Specific parameters and options for information display, including search options, types of graph and display items.	K	Display Options
R10: Related information after all analysis, inferring and statistics for display.	K	Treated Information
R11: Manage interactions with the database.	D	DB Connector
R12: Classify, do statistics, analyze and infer related information for suggestion, statistics display and recommendation.	D	Data Analyst
R13: Retrieve related data from Twitter Database.	D	Third Part Connector
R14: Retrieve related services from Google Database.	D	Third Part Connector
R15: Relate to third party API, so that users can invite their friends.	D	Third Part Connector
R16: Use an algorithm to calculate the energy user take in and display.	D	Data Analyst
R17: Use an algorithm to suggest a healthy diet for user.	D	Data Analyst

Table 6-1 Concept definitions

6.1.2 Association definitions

Some concepts defined above need to work together in order to achieve specific functions. The concepts work together are called concept pair. The definition and description of concept pair of the system are listed in Table 5-2.

Concept pair	Association description	Association name
Controller ↔ Profile Storage	Controller updates User Profile when the users change his/her information.	Updates
Controller ↔Interface Service	Controller passes requests to Interface Service and receives pages prepared for displaying.	Convey requests
Interface Service ↔ Interface Pages	Interface Service prepares the Interface Pages.	Prepares
Interface Service ↔ Display Options	Interface Service prepares the Display Options.	Prepares
Interface Service ↔DB Connector	Database Connection passes the retrieved data to Interface Service to render them for display and show.	Provides data
Interface Service ↔ Treated Information	Interface Service extracts information from Treated Information for display.	Provides data
Interface Service↔ Third Part Connector	Third Part Connector enable Interface Service to connect the Third Part to ask for service (Like the Google map and graphs).	Connects Third Part
DB Connector ↔Third Part Connector	Third Part Connector enables DB Connector to connect the Third Part to retrieve related data.	Connects Third Part
Data Analyst ↔ Treated Information	Data Analyst prepares the Treated Information.	Prepares
Data Analyst ↔Profile Storage	Data Analyst changes the information (like score and health status) in Profile Storage.	Updates
DB Connector ↔Data Analyst	Database Connection passes the retrieved data to Data Analyst for analysis, inferring and statistics. Data Analyst stores useful data back into Database after analysis.	Provides data, Stores data

Table 6-2 Association definitions

6.1.3 Attribute definitions

Attributes of domain concepts are derived in Table 5-3.

Concept	Attribute	Attribute Description
Controller	displayRequest	Send user’s requests to retrieve display service
	updateInfoRequest	Send user’s requests to retrieve profile change service
	loginRequest	Use the data the user input to request a login operation
	isLogged	Identity parameter to determine whether the user is login
	isAuthorized	Identity parameter to determine whether the user is authorized
Profile Storage	accountID	Identity number used to determine the user
	accountKey	Specific key to determine the user’s credentials
	score	User’s current score in the score system
	demo graphicsInfo	User’s personal information like age, gender, location, etc.
	healthStatus	User’s exercise-related status given by the analyst
Interface Service	publicStatistics	Show the information about public health related statistic
	publicSuggestion	Show the exercise suggestions for the whole public
	personalStatistics	Show the information about personal health related statistic
	personalSuggestion	Show the health-related suggestions for user
	facilitiesSearch	Search for facilities information with given conditions
	facilityAvailability	Identity parameter to determine whether facility is available
Interface Pages	pageDisplay	Pages for related information display.
	pageHome	Home page of the website or application.
	pageProfile	User’s profile information page.
	pageLogin	Section on the website or in the application for user to login
	searchBar	In site search bar for user to search for related display service
Display Options	heatMap	Heat map for showing intensity or amount of relation data

	graph	Graph type information display
	literalSuggestion	Literal suggestion given out by the analyst showed in word
	leaderBoard	Leaderboard used to show the ranking of different data
	displayItems	Specific items for user to choose for display, such items are set as exercise amount, intensity,cites, etc.
	timeInterval	Time interval for user to choose for information display
Treated Information	exerciseIntensity	People’s exercise regularity and intensity calculated by analyst
	tweetData	Data of amount and location of health-related tweets
	sortRanking	Amount, intensity or other attributes of tweets ranking among different set like cities, states, users, exercise types, etc.
	groupsOverlap	Data of the overlap between people who concern about wellness and who exercises and also between people who exercises and who talking about diet
	dataCorrelation	Correlation between different type of health related tweets
	personalSuggestions	Personal suggestion include exercise intensity and regularity suggestion, personal ranking, etc.
	publicSuggestions	Public suggestion based on the average intensity and regularity of exercise among the whole country
	typeTrend	Trend of amount, intensity, regularity in different exercise type
	rctTweetData	Contents, location, user information of related tweets collected recently
	dataScore	User’s data in score system, including exercise score, ranking in the system, award based on the score, etc.
	analysisResult	Analysis based on User’s historical exercise

		data, like average exercise intensity and regularity, etc.
	emotionInfo	Data about distribution, amount and type of emotion extracted from tweets.
DB Connector	dbAddress	Address of relation database
	dbAccount	Account to manage the database
	dbKey	Key to manage the database
	databaseStatus	Identity parameter to determine whether the database is open
	numOfTweets	total number of parsed tweets
	dataCommunicate	Retrieve and store data from third part server, due with data communication inside system.
Data Analyst	analyzePublic	Analysis for public information, including statistics data, suggestion inferring, trends calculating, correlation analysis, etc.
	analyzePersonal	Analysis for personal information, including historical record, personal health status deduction, specific suggestions, score calculating, etc.
Third Part Connector	connectTwitter	Connect Twitter by API Auth. to retrieve tweets
	connectGoogle	Connect Google by API Auth. to retrieve map, chart and place finding services

Table 6-3 Attribute definitions

6.2 System Operation Contracts

Operation	Search for information
Pre-conditions	Related information is stored in the system's database Read the user's input and put it into "Keyword" Use assigned algorithms to analyze related data System's database can be changed
Post-conditions	New tables have been built in the database Statical data are stored in the related table

	System displays the chosen diagrams on the screen

Table 6-7 Searches for information

Operation	Displaying
Pre-conditions	Related information is stored in the system’s database
Post-conditions	System displays the chosen diagrams on the screen

Table 6-8 Displaying

6.3 Mathematical Model.

Data process

We use several data processing methods to analyze the tweeter in order to give a ratio about the sports self-participation or audience. The ratio can indicate the way people enjoy sports in certain area.

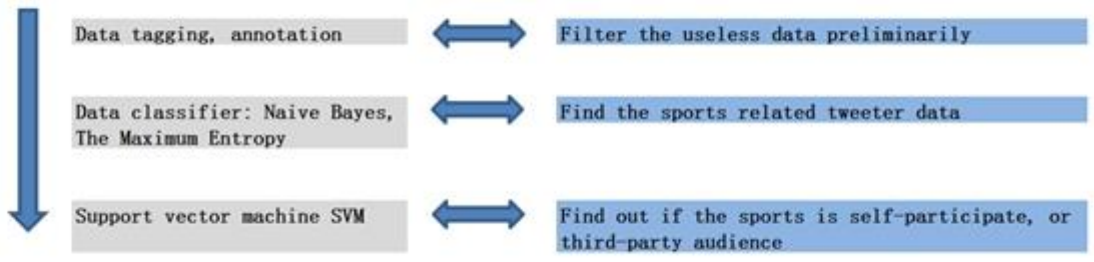

Data tagging, annotation
Filter the useless data preliminarily
Data classifier: Naive Bayes, The Maximum Entropy
Find the sports related tweeter data
Support vector machine SVM
Find out if the sports is self-participate, or third-party audience

Figure 6-2

6.3.1 Implement the NLTK method to tag the tweeter data

As we download mass raw data from twitter with certain key words, the result always can be blurry or incorrect, we need several methods to identify and make sure the valued data go to the correct position while the incorrect one will be disposed. So we use NLTK with python, to do these works.

By convention in NLTK, a tagged token is represented using a tuple consisting of the token and the tag.

For example, we want to know if the user really do the sports, then we can two twitter

a.

```
>>>text = nltk.word_tokenize(" I swim 20 miles this morning")
```

```
>>>nltk.pos_tag(text)
[('I', 'P'), (‘swim, 'v'), (' 20 ', 'NUM'), (' miles ', DET),
(this, 'RB'), (morning, 'JJ')]
```

b.

```
>>>text = nltk.word_tokenize("it’s really a swimming pool")
>>> nltk.pos_tag(text)
[('It’s', 'EX'), (‘really, 'ADV'), (' a ', 'IN'), (' good ', 'ADJ'),
(swimming, 'ADJ'), (pool, 'N')]
```

We can differentiate the meaning of swim by the tag ‘V’ and ‘ADJ’, so we can know that the first sentence is the one we want to take while second one will be discarded.

Thus sports can be represented by several key words.

Here is the meaning of each tag:

Tag	Meaning	Examples
ADJ	adjective	new, good, high, special, big, local
ADV	adverb	really, already, still, early, now
CNJ	conjunction	and, or, but, if, while, although
DET	determiner	the, a, some, most, every, no
EX	existential	there, there’s
FW	foreign word	dolce, ersatz, esprit, quo, maitre
MOD	modal verb	will, can, would, may, must, should
N	noun	year, home, costs, time, education
NP	proper	Alison, Africa, April, Washington
NUM	number	twenty-four, fourth, 1991, 14:24
PRO	pronoun	he, their, her, its, my, I, us
P	preposition	on, of, at, with, by, into, under
TO	the	word to to
UH	interjection	ah, bang, ha, whee, hmpf, oops
V	verb	is, has, get, do, make, see, run
VD	past	said, took, told, made, asked
VG	present participle	making, going, playing, working
VN	past participle	given, taken, begun, sung
WH	whdeterminer	who, which, when, what, where, how

Table 6-9

There are two tagging method

1. Unigram taggers

Unigram taggers are based on a simple statistical algorithm: for each token, assign the tag that is most likely for that particular token.

We train a Unigram Tagger by specifying tagged sentence data as a parameter when we initialize the tagger. The training process involves inspecting the tag of each word and storing the most likely tag for any word in a dictionary that is stored inside the tagger.

2. General N-Gram Tagging

When we perform a language processing task based on unigrams, we are using one segment of context. In the case of tagging, we consider only the current token, which is in isolation of any larger context. Given such a model, the best we can do is tag each word with its priority that mostly similar to tag. This means we would tag a word such as wind with the same tag, regardless of whether it appears in the context.

An n-gram tagger is a generalization of a unigram tagger whose context is the current word together with the part-of-speech tags of the n-1 preceding tokens, as shown in next picture. The tag to be chosen is circled and the context is shaded in grey. In the example of an n-gram tagger shown in Figure 5-5, we have n=3; that is, we consider that the tags of the two preceding words in addition to the current word.

An n-gram tagger picks the tag that is most likely in the given context

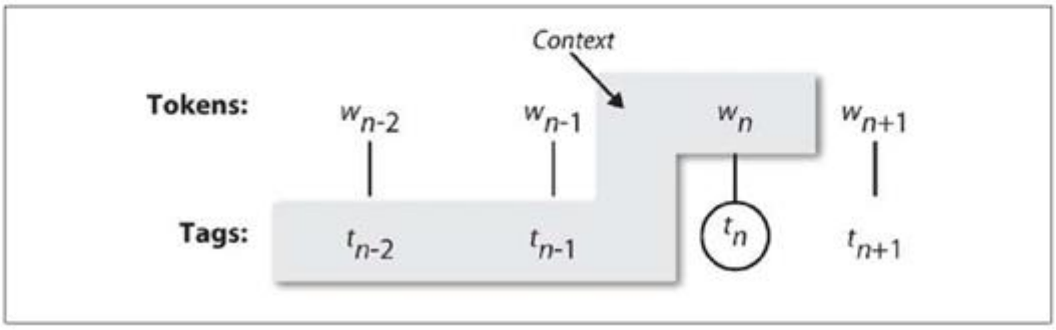


Figure 6-3

6.3.2 Naive Bayes Classifiers

After tag, we need to classify every twitter text. There are several ways we can use, first we consider Naive Bayes Classifiers

Abstractly, naive Bayes is a conditional probability model: given a problem instance to be classified, represented by a vector representing some n features (independent variables), it assigns to this instance probabilities

$$p(C_k|x_1,\dots,x_n)\ \mathbf{x}=(x_1,\dots,x_n)$$

for each of K possible outcomes or classes.

The problem with the above formulation is that if the number of features n is large or if a feature can take on a large number of values, then basing such a model on probability tables are infeasible. We therefore reformulate the model to make it more tractable. Using Bayes' theorem, the conditional probability can be decomposed as

$$p(C_k|\mathbf{x})=\frac{p(C_k)\ p(\mathbf{x}|C_k)}{p(\mathbf{x})}.$$

In plain English, using Bayesian probability terminology, the above equation can be written as

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}.$$

In practice, there is interest only in the numerator of that fraction, because the denominator does not depend on C and the values of the features F_i are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model

$$p(C_k, x_1, \dots, x_n)$$

which can be rewritten as follows, using the chain rule for repeated applications of the definition of conditional probability:

$$\begin{aligned} p(C_k, x_1, \dots, x_n) &= p(C_k)\ p(x_1, \dots, x_n | C_k) \\ &= p(C_k)\ p(x_1 | C_k)\ p(x_2, \dots, x_n | C_k, x_1) \\ &= p(C_k)\ p(x_1 | C_k)\ p(x_2 | C_k, x_1)\ p(x_3, \dots, x_n | C_k, x_1, x_2) \\ &= p(C_k)\ p(x_1 | C_k)\ p(x_2 | C_k, x_1)\ \dots p(x_n | C_k, x_1, x_2, x_3, \dots, x_{n-1}) \end{aligned}$$

Now the "naive" conditional independence assumptions come into play: assume that each feature F_i is conditionally independent of every other feature F_j for $j \neq i$, given the category C. This means that

$$p(x_i | C_k, x_j) = p(x_i | C_k),$$

$$p(x_i | C_k, x_j, x_q) = p(x_i | C_k),$$

$$p(x_i | C_k, x_j, x_q, x_l) = p(x_i | C_k),$$

and so on, for $i \neq j, q, l$. Thus, the joint model can be expressed as

$$\begin{aligned} p(C_k | x_1, \dots, x_n) &\propto p(C_k, x_1, \dots, x_n) \\ &\propto p(C_k)\ p(x_1 | C_k)\ p(x_2 | C_k)\ p(x_3 | C_k)\ \cdots \\ &\propto p(C_k) \prod_{i=1}^{n} p(x_i | C_k)\,. \end{aligned}$$

This means that under the above independence assumptions, the conditional distribution over the class variable C is:

$$p(C_k | x_1, \dots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i | C_k)$$

where the evidence $Z = p(\mathbf{x})$ is a scaling factor dependent only on $x_1, \dots, x_n$, that is, a constant if the values of the feature variables are known.

Constructing a classifier from the probability model

The discussion so far has derived the independent feature model, that is, the naive Bayes probability model. The naive Bayes classifier combines this model with a decision rule. One common rule is to pick the hypothesis that is most probable; this is known as the maximum a posteriori or MAP decision rule. The corresponding classifier, a Bayes classifier, is the function that assigns a class label $\hat{y} = C_k$for some k as follows:

$$\hat{y} = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}} \ p(C_k) \prod_{i=1}^{n} p(x_i | C_k).$$

6.3.4 The Maximum Entropy Model

The Maximum Entropy classifier uses a model that is very similar to the model employed by the naive Bayes classifier. But rather than using probabilities to set the model's parameters, it uses search techniques to find a set of parameters that will maximize the performance of the classifier. In particular, it looks for the set of parameters that maximizes the total likelihood of the training corpus, which is defined as:

$$P(features) = \Sigma_{x \in \text{corpus}} P(label(x)|features(x))$$

Where P(label|features), the probability that an input whose features are features will have class label label, is defined as:

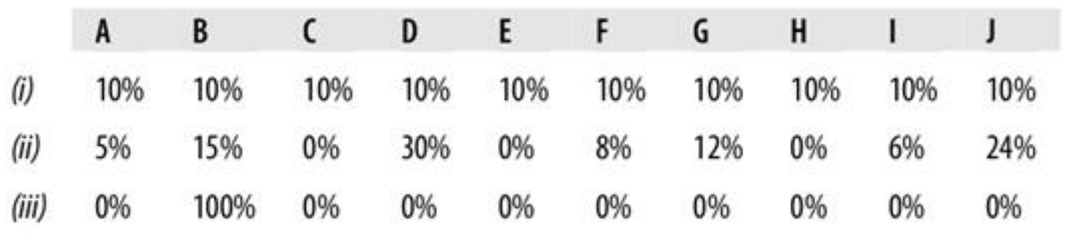
$$P(label|features) = P(label, features)/\Sigma_{label} P(label, features)$$

The Maximum Entropy classifier model is a generalization of the model used by the naive Bayes classifier. Like the naive Bayes model, the Maximum Entropy classifier calculates the likelihood of each label for a given input value by multiplying together the parameters that are applicable for the input value and label. The naive Bayes classifier model defines a parameter for each label, specifying its prior probability, and a parameter for each (feature, label) pair, specifying the contribution of individual features toward a label's likelihood.

Given the joint-features for a Maximum Entropy model, the score assigned to a label for a given input is simply the product of the parameters associated with the joint-features that apply to that input and label:

$$P(input, label) = \Pi_{joint\text{-}features(input,label)} w[joint\text{-}feature]$$

Suppose we are assigned the task of picking the correct word sense for a given word, from a list of 10 possible senses (labeled A–J). At first, we are not told anything more about the word or the senses. There are many probability distributions that we could choose for the 10 senses, such as:



	A	B	C	D	E	F	G	H	I	J
(i)	10%	10%	10%	10%	10%	10%	10%	10%	10%	10%
(ii)	5%	15%	0%	30%	0%	8%	12%	0%	6%	24%
(iii)	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%

Figure 6-4

Although any of these distributions might be correct, we are likely to choose distribution (i), because without any more information, there is no reason to believe that any word sense is more likely than any other.

6.3.5 SVM (Support Vector Machine)

Support Vector Machines (SVMs) are a popular machine learning method for classification, regression, and other learning tasks. LIBSVM is a library for Support Vector Machines (SVMs).

LIBSVM has gained wide popularity in machine learning and many other areas.

LIBSVM supports various SVM formulations for classification, regression, and distribution estimation.

We need to analyze if we are the one who really participate in the sports, or just watching. We use SVM to analyze.

C-Support Vector Classi_cation

Given training vectors xi 2 Rn; i = 1; : : : ; l, in two classes, and an indicator vectory 2 Rl such that yi 2 f1;�1g, C-SVC (Boser et al., 1992; Cortes and Vapnik, 1995) solves the following primal optimization problem.

$$\begin{array}{ll} \min\limits_{\boldsymbol{w},b,\boldsymbol{\xi}} & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} & y_i(\boldsymbol{w}^T\phi(\boldsymbol{x}_i)+b) \geq 1-\xi_i, \\ & \xi_i \geq 0, i = 1,\ldots,l, \end{array}$$

where _(xi) maps xi into a higher-dimensional space and C > 0 is the regularization parameter. Due to the possible high dimensionality of the vector variable w, usually we solve the following dual problem.

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \boldsymbol{e}^T\boldsymbol{\alpha} \\ \text{subject to} \quad & \boldsymbol{y}^T\boldsymbol{\alpha} = 0, \\ & 0 \le \alpha_i \le C, \qquad i = 1, \ldots, l, \end{aligned}$$

where e = [1; : : : ; 1]T is the vector of all ones, Q is an l by l positive matrix, Qij _ yiyjK(xi; xj), and K(xi; xj) _ _(xi)T_(xj) is the kernel function. After problem (2) is solved, using the primal-dual relationship.

$$\boldsymbol{w} = \sum_{i=1}^{l} y_i \alpha_i \phi(\boldsymbol{x}_i)$$

And the decision function is

$$\operatorname{sgn}\left(\boldsymbol{w}^T\phi(\boldsymbol{x}) + b\right) = \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right).$$

We store yi_i 8i, b, label names,4 support vectors, and other information such as kernel parameters in the model for prediction.

Implementation

We use this function to calculate the weight

$$\phi_i(x) = \frac{\mathrm{tf}_i \log(\mathrm{idf}_i)}{\kappa}$$

Tf is the terminology means the number of occurrence in the x. idf is the ratio that the number with the entire file with the file containing the value you are testing.

Use this function; we get every feature weight for tweeter we want to predict. Use makes them the format which libsvm can use.

```
1 1:-0.555556 2:0.5 3:-0.694915 4:-0.75
3 1:-0.166667 2:-0.333333 3:0.38983 4:0.916667
2 1:-0.333333 2:-0.75 3:0.0169491 4:-4.03573e-08
1 1:-0.833333 3:-0.864407 4:-0.916667
1 1:-0.611111 2:0.0833333 3:-0.864407 4:-0.916667
3 1:0.611111 2:0.333333 3:0.728813 4:1
3 1:0.222222 3:0.38983 4:0.583333
2 1:0.222222 2:-0.333333 3:0.220339 4:0.166667
2 1:-0.222222 2:-0.333333 3:0.186441 4:-4.03573e-08
```

Format: label feature1:value1 index2:value2...

We use python or c++ project which libsvm provide to predict the result, which categories the tweeter can go into.

For example (Python):

```
Creating the model
>>mod = svm_model(prob, param)
>>target = cross_validation (prob, param, n)

save the model and load
>>mod.save(‘modelfile’)
>>mod2 = svm_model(‘modelfile’)

Test
r = mod.predict ([1, 1, 1])
d = mod.predict_values([1, 1, 1])
prd, prb = m.predict_probability([1, 1, 1])
```

d is the result we need to know: 1 for class0, -1 for class1.

7. Interaction Diagrams

7.1 Use Case 1: Data Collecting and Classify

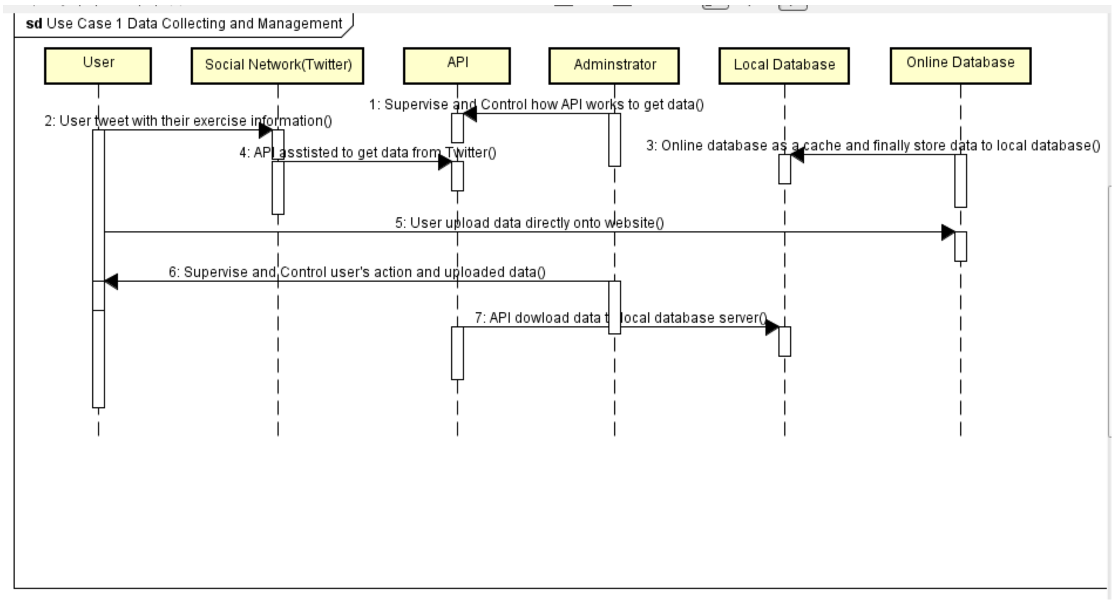

sd Use Case 1 Data Collecting and Management
User
Social Network(Twitter)
API
Adminstrator
Local Database
Online Database
1: Supervise and Control how API works to get data()
2: User tweet with their exercise information()
3: Online database as a cache and finally store data to local database()
4: API asstisted to get data from Twitter()
5: User upload data directly onto website()
6: Supervise and Control user's action and uploaded data()
7: API dowload data to local database server()

Figure 7-1 Use Case 1: Data Collecting and Classify

The figure above comes from the use case 1: Data Collection and Management. First the administrator control how the system get tweeter from API from tweeter web: the mining speed, frequency, or other parameter. At mean time, all the user publish their tweeters on line, we get relative ones via certain words. The next stop, we store the data into ‘online-database’ and the online database will communicate with our local database. Next, user upload their data or request like searching certain activities on line. Controller or supervise need to check if their actions is malicious. Finally, we use API to download data into our database.

7.2 Use Case 3: Data Classifying

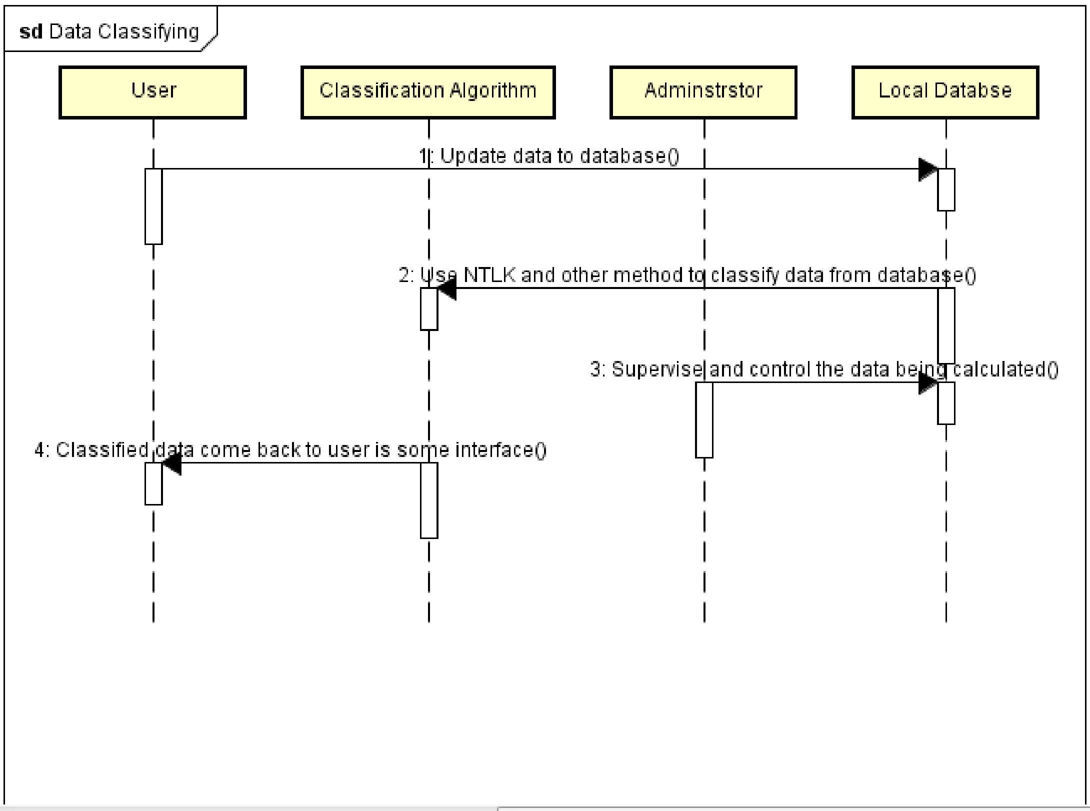

sd Data Classifying
User
Classification Algorithm
Adminstrstor
Local Databse
1: Update data to database()
2: Use NTLK and other method to classify data from database()
3: Supervise and control the data being calculated()
4: Classified data come back to user is some interface()

Figure 7-2 Use Case 3: Data Classifying

The figure above comes from user case 3: Data Classifying. The users' tweeter will be sorted and stored into our database. They the classifying service will classify the data using NLTK which is a lib for Phyton. After classification,, the supervisor will use another advanced method to put them into more specific categories. And the last the step is to give back all the information to application to display.

7.3 Use Case 5: Heat Map

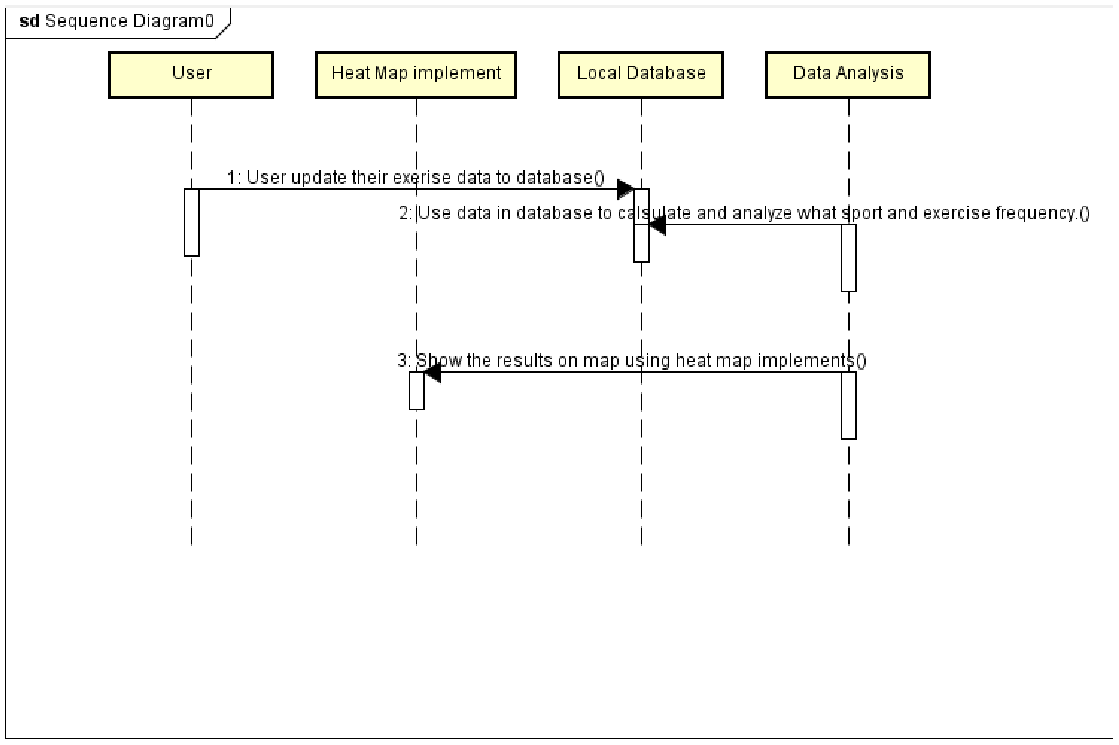

sd Sequence Diagram0
User
Heat Map implement
Local Database
Data Analysis
1: User update their exerise data to database()
2: Use data in database to calculate and analyze what sport and exercise frequency.()
3: Show the results on map using heat map implements()

Figure 7-3 Use Case 5: Heat Map

The figure above comes from user case5: Heat Map. The first step is all the users data will be filtered, sorted, classified and stored into the database. Then all these valid data will be calculated and generated useful data for analyze. The last step users can see many kinds of data through heat map to get a general impression about the activities trend.

7.4 Use Case 13&20: Search For Information&Disp

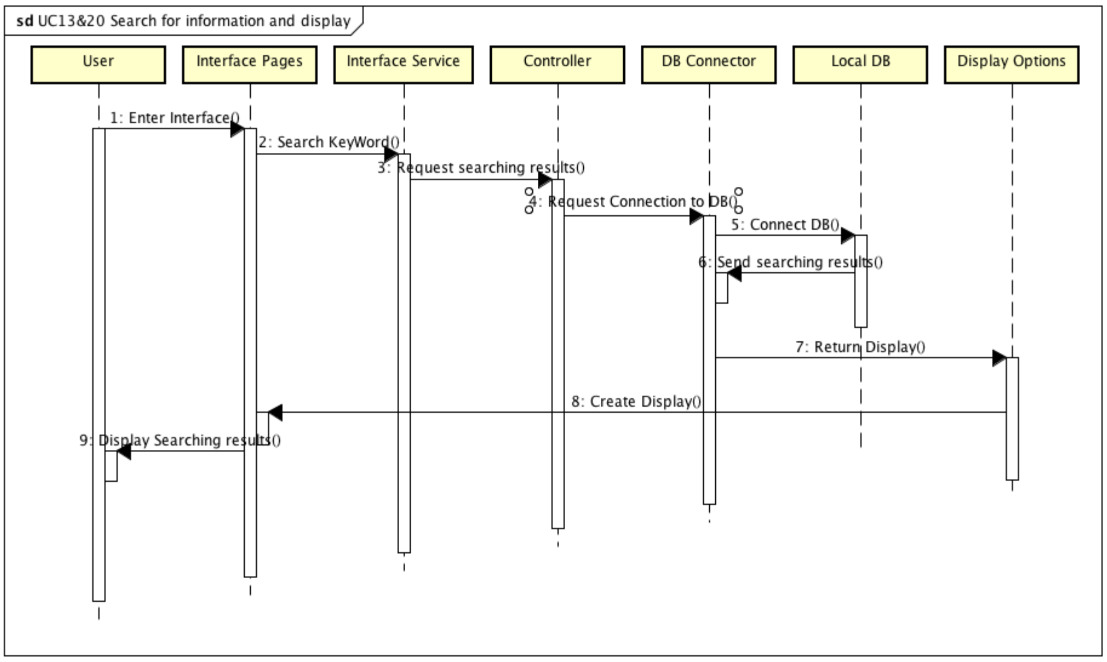


Figure 7-4 Use Case 13 & 20 Search for Information and Display

The function of Use Case 13 & 20 is displaying the information of searching request made by the user. First, any user opened the website, simply type in the key words they want to search and hit enter on the keyboard or click on the search button. Then the Interface Service emits the request to Controller for getting searching results. Since the controller could not get touch with the database directly, it should send the request to the DB Connector for connection to the local DB. After Local DB returns the related information to the DB Connector. The specific data for user is sent to the Display Options for proper ways to display. Correlate display ways for searching results are created by Display Options and should be posted to Interface Pages. At last the Interface Pages display the searching results for user on the screen.

7.5 Use Case 2: User Registration

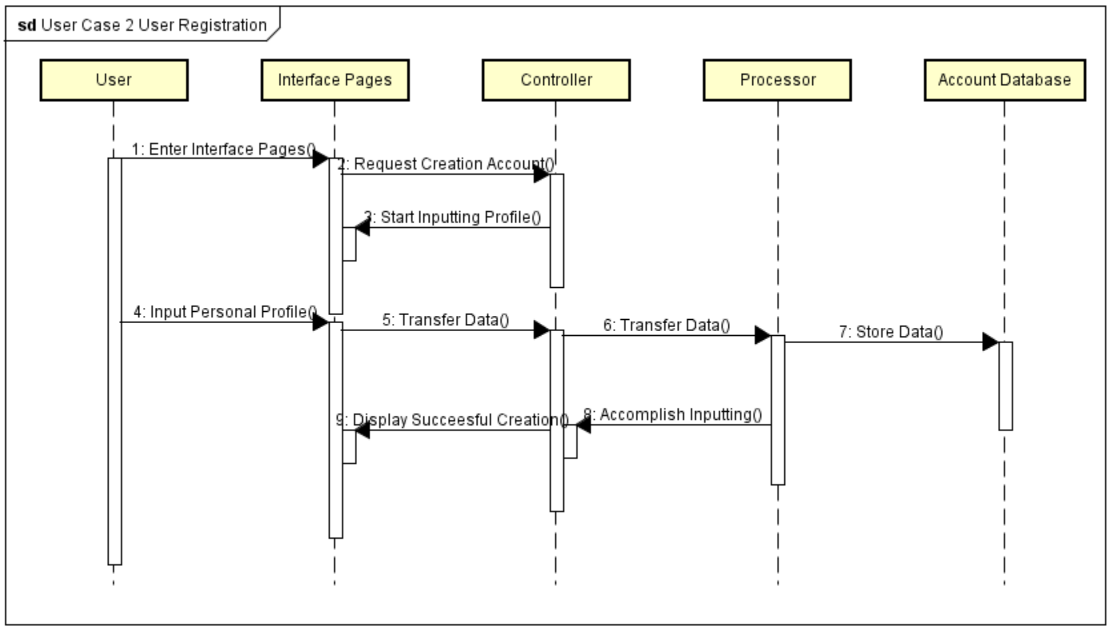

sd User Case 2 User Registration
User
Interface Pages
Controller
Processor
Account Database
1: Enter Interface Pages()
2: Request Creation Account()
3: Start Inputting Profile()
4: Input Personal Profile()
5: Transfer Data()
6: Transfer Data()
7: Store Data()
8: Accomplish Inputting()
9: Display Succeesful Creation()

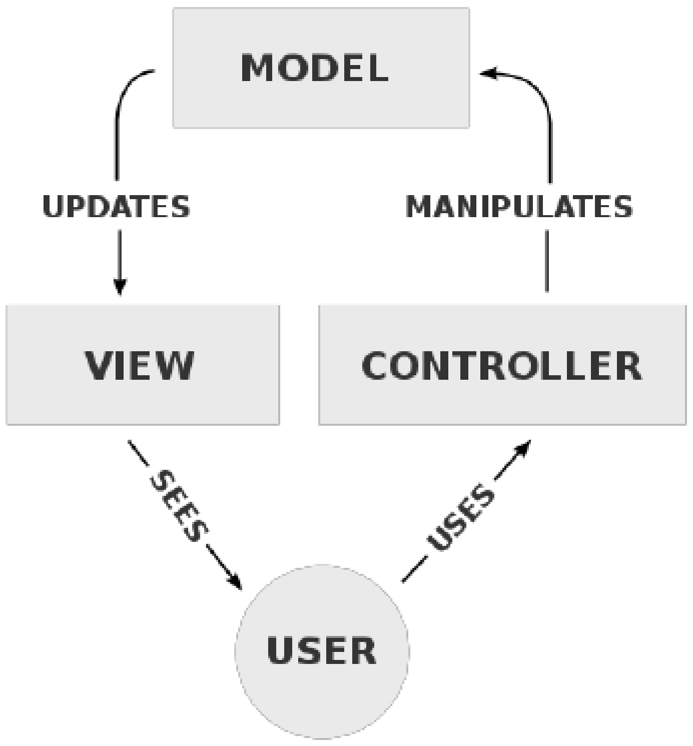
Figure 7-5 Use Case 2 User Registration

The figure above comes from user case 2: User Registration. First, user enter our web site and select "Create a new account". Then system will receive this request and let user input personal profile on the web site. Next all profile data transfer to account database and the data is stored there. While system check out that all the blanks in profile have received the data, the interface page will show "Successful creating your account" on the screen.

7.6 Design Patterns

Our system design mainly follows model-view-controller (MVC) design pattern.

The communications between our server and website, our server all follow this pattern. The definition of MVC is as follows from Wiki: "The central component of MVC, the model, captures the behavior of the application in terms of its problem domain, independent of the user interface. The model directly manages the data, logic and rules of the application. A view can be any output representation of information, such as a chart or a diagram; multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants. The third part, the controller, accepts input and converts it to commands for the model or view."

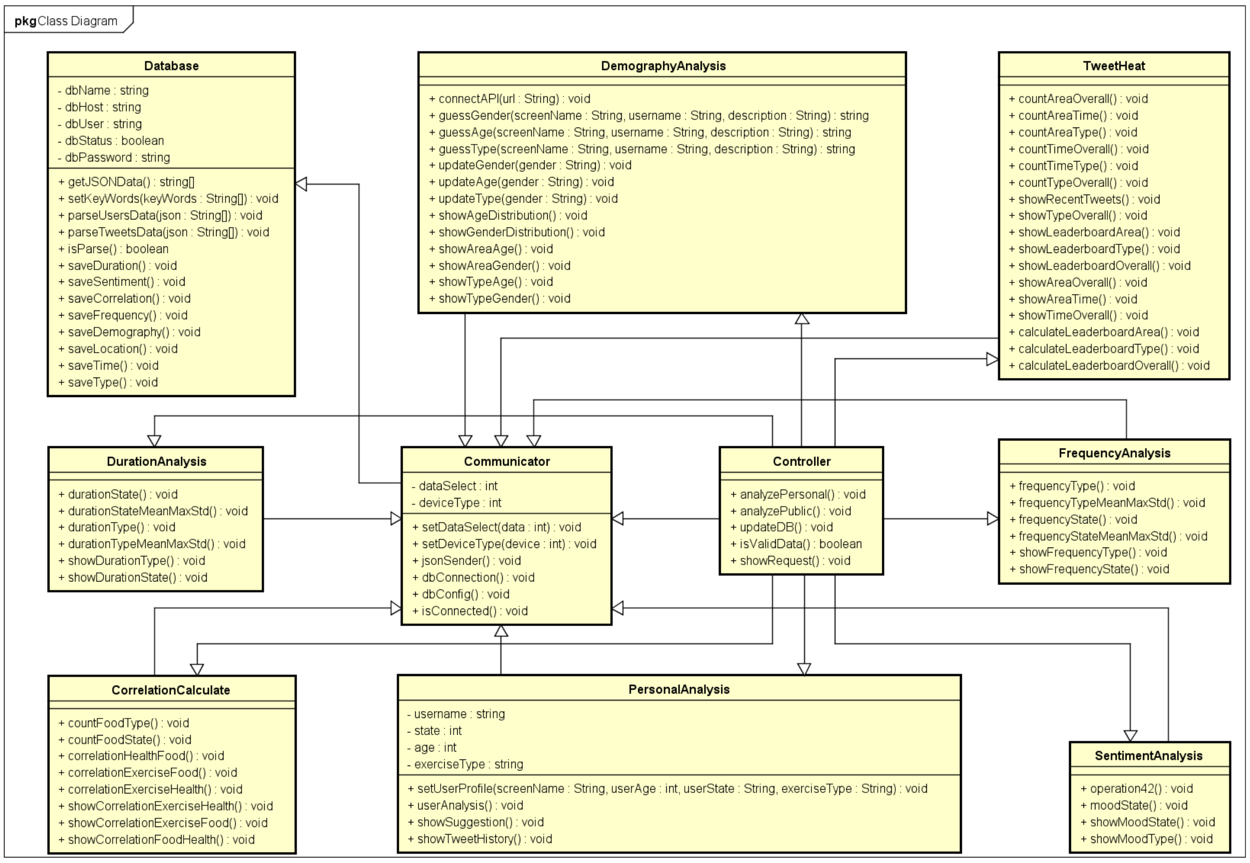


The control flow of MVC can be described as Figure 7-6.

How we use MVC in our system? The HTML website, IOS and Android user interface are the view part in MVC design pattern. The database management is the model part. We have a communication controller called JSON sender in our system performed as the controller in MVC. The user will interact with the website, IOS, android user interface to trigger the events, and send HTTP POST (IOS) and GET (website, Android) request to the controller. The controller will manipulate database by SQL based on the request with feature id to response the feature data from feature tables. Then the view part will visualize the returned feature data, and reply the user.

8. Class Diagram and Interface Specification

8.1 Class Diagram



pkg Class Diagram
Database
- dbName : string
- dbHost : string
- dbUser : string
- dbStatus : boolean
- dbPassword : string
+ getJSONData() : string[]
+ setKeyWords(keyWords : String[]) : void
+ parseUsersData(json : String[]) : void
+ parseTweetsData(json : String[]) : void
+ isParse() : boolean
+ saveDuration() : void
+ saveSentiment() : void
+ saveCorrelation() : void
+ saveFrequency() : void
+ saveDemography() : void
+ saveLocation() : void
+ saveTime() : void
+ saveType() : void
DemographyAnalysis
+ connectAPI(url : String) : void
+ guessGender(screenName : String, username : String, description : String) : string
+ guessAge(screenName : String, username : String, description : String) : string
+ guessType(screenName : String, username : String, description : String) : string
+ updateGender(gender : String) : void
+ updateAge(gender : String) : void
+ updateType(gender : String) : void
+ showAgeDistribution() : void
+ showGenderDistribution() : void
+ showAreaAge() : void
+ showAreaGender() : void
+ showTypeAge() : void
+ showTypeGender() : void
TweetHeat
+ countAreaOverall() : void
+ countAreaTime() : void
+ countAreaType() : void
+ countTimeOverall() : void
+ countTimeType() : void
+ countTypeOverall() : void
+ showRecentTweets() : void
+ showTypeOverall() : void
+ showLeaderboardArea() : void
+ showLeaderboardType() : void
+ showLeaderboardOverall() : void
+ showAreaOverall() : void
+ showAreaTime() : void
+ showTimeOverall() : void
+ calculateLeaderboardArea() : void
+ calculateLeaderboardType() : void
+ calculateLeaderboardOverall() : void
DurationAnalysis
+ durationState() : void
+ durationStateMeanMaxStd() : void
+ durationType() : void
+ durationTypeMeanMaxStd() : void
+ showDurationType() : void
+ showDurationState() : void
Communicator
- dataSelect : int
- deviceType : int
+ setDataSelect(data : int) : void
+ setDeviceType(device : int) : void
+ jsonSender() : void
+ dbConnection() : void
+ dbConfig() : void
+ isConnected() : void
Controller
+ analyzePersonal() : void
+ analyzePublic() : void
+ updateDB() : void
+ isValidData() : boolean
+ showRequest() : void
FrequencyAnalysis
+ frequencyType() : void
+ frequencyTypeMeanMaxStd() : void
+ frequencyState() : void
+ frequencyStateMeanMaxStd() : void
+ showFrequencyType() : void
+ showFrequencyState() : void
CorrelationCalculate
+ countFoodType() : void
+ countFoodState() : void
+ correlationHealthFood() : void
+ correlationExerciseFood() : void
+ correlationExerciseHealth() : void
+ showCorrelationExerciseHealth() : void
+ showCorrelationExerciseFood() : void
+ showCorrelationFoodHealth() : void
PersonalAnalysis
- username : string
- state : int
- age : int
- exerciseType : string
+ setUserProfile(screenName : String, userAge : int, userState : String, exerciseType : String) : void
+ userAnalysis() : void
+ showSuggestion() : void
+ showTweetHistory() : void
SentimentAnalysis
+ operation42() : void
+ moodState() : void
+ showMoodState() : void
+ showMoodType() : void

Figure 8-1 Class Diagram

This figure shows up the class diagram of our system. We distribute our system functions into ten classes.

Before other classes could accomplish their functions, the “database” class will be called first in order to collect tweets data, analyze the data, parse the data in JSON format, and store the data after analysis.

“Communicator” class is the class which deals with the work between showing function and the database. It could control what data should be transferred from the database and should be shown up on which screen.

“Controller” class control all the system include user’s demand and system control. From this class, other classes function would know when to run their own analysis functions.

8.2. Data Types and Operation Signatures

Note that, since we are building our system based on the previous groups' work, some basic classes and functions remains the same as previous groups' because they are already well implemented.

8.2.1. Database

The "Database" class contains five variables as follows that stores the basic information of the database used for the connection and coordination.

dbName String variable storing the name of the database.

dbHost String variable storing the host of the database.

dbUser String variable storing the username of the database.

dbStatus Boolean variable indicating the status of the database, either open or closed.
dbPassword String variable storing the password corresponding to the username.

The functions of this class are listed below. Functions include several get functions that are used to obtain data from Twitter. Also they contain several save functions that are used for storing analytical data into database.

getJSONData() This function is used for obtaining raw tweets data from Twitter Streaming API and returns an array of JSON value, each element represents one piece of tweet data.
setKeyWords(keyWords : String[]) This function is used for setting the keywords that are needed for filtering the tweets. As for our project, all the key words should be about health and exercise.
parseUsersData(json : String[]) This function is used for parsing the collected JSON data containing user information into readable user data and storing them in the users table of our database.
parseTweetsData(json : String[]) This function is used for parsing the collected JSON data which contains tweets information into readable tweet data and store them in different tables.
isParse() This function is used for check the parse result of each piece of JSON data. If data has been parsed, return true. Else, return false.
saveDuration() This function is used for saving the analytical data related to the exercise duration time into new feature tables of the database.
saveSentiment() This function is used for saving the analytical data related to the sentiment computing into new feature tables of the database.

saveCorrelation() This function is used for saving the analytical data related to the correlation computing into new feature tables of the database.

saveFrequency() This function is used for saving the analytical data related to the keyword frequencies into new feature tables of the database.

saveDemography() This function is used for saving the analytical data related to the demography information into new feature tables of the database.

saveLocation() This function is used for saving the analytical data related to different locations into new feature tables of the database.

saveTime() This function is used for saving the analytical data related to different time of the day into new feature tables of the database.

saveType() This function is used for saving the analytical data related to the exercise types into new feature tables of the database.

8.2.2. Tweet Heat

The “TweetHeat” class contains mostly count functions. They are used to count the number of tweets in different categories. This class also contains several calculate functions to sort the count result to make the leaderboard, and also some show functions to display all the responding charts onto the user interface.

Count functions:

countAreaOverall() This function is used for counting the number of tweets in different areas.

countAreaTime() This function is used for counting the number of occurrences of different areas in different time periods.

countAreaType() This function is used for counting the number of occurrences of different exercise types in different areas.

countTimeOverall() This function is used for counting the number of occurrences of different time periods.

countTimeType() This function is used for counting the number of occurrences of different time periods corresponding to different exercise types.

countTypeOverall() This function is used for counting the number of occurrences of different exercise types.

Show functions:

showRecentTweets() This function is used for showing on map the most recently posted tweet and its user.

showTypeOverall() This function is used for displaying the analytical chart showing the number of tweets concerning different exercise types.

showLeaderboardArea() This function is used for displaying the leader board ranked by the number of tweets concerning different areas.

showLeaderboardType() This function is used for displaying the leader board ranked by the number of tweets concerning different exercise types.

showLeaderboardOverall() This function is used for displaying the leader board ranked by the number of tweets concerning exercise and health.

showAreaOverall() This function is used for displaying the analytical chart showing the number of tweets in different states.

showAreaTime() This function is used for displaying the analytical chart showing the number of tweets in different states in different time periods.

showTimeOverall() This function is used for displaying the analytical chart showing the number of tweets in different time periods.

Calculation functions:

calculateLeaderboardArea() This function is used for sorting the count result of different areas and storing them in decreasing order.

calculateLeaderboardType() This function is used for sorting the count result of different exercise types and storing them in decreasing order.

calculateLeaderboardOverall() This function is used for sorting the count result of all tweets concerning exercise and health tweeted by each user and storing them in decreasing order.

8.2.3. Frequency Analysis

The “FrequencyAnalysis” class contains several functions. They are used to calculate the different frequencies corresponding to different states and types.

There are also some functions that used for calculating the mean, max and standard deviation of such frequencies. In addition, there are some show functions to display all the responding charts onto the user interface.

Frequency Calculation functions:

frequencyType() This function is used for calculating the frequency of exercise in different exercise types according to user’s tweets.

frequencyTypeMeanMaxStd() This function is used for counting the mean, maximum and standard deviation of those frequency calculated in ***fregquencyType()***.

frequencyState() This function is used for calculating the frequency of exercise in different states according to user’s tweets.

frequencyStateMeanMaxStd() This function is used for counting the mean, maximum and standard deviation of those frequency calculated in ***fregquencyState()***.

Show functions:

showFrequencyType() This function is used for displaying the analytical chart that would show the frequency values of exercise corresponding to different exercise types.

showFrequencyState() This function is used for displaying the analytical chart that would show the frequency values of exercise corresponding to different states.

8.2.4. Duration Analysis

The “DurationAnalysis” class contains several functions that are used to calculate the duration of exercise corresponding to different states and types. There are also some methods that could calculate the mean, max and standard deviation of such durations. In addition, there are some show functions to display all the responding charts onto the user interface.

Duration calculation functions:
durationState() This function is used for calculating the Duration of exercise in different states according to user’s tweets.
durationStateMeanMaxStd() This function is used for counting the mean, maximum and standard deviation of those duration calculated in ***durationState()***.
durationType() This function is used for calculating the duration of exercise in different exercise types according to user’s tweets.
durationTypeMeanMaxStd() This function is used for counting the mean, maximum and standard deviation of those duration calculated in ***durationType()***.

Show functions:
showDurationType() This function is used for displaying the analytical chart that would show the duration values of exercise corresponding to different exercise types.
showDurationState() This function is used for displaying the analytical chart that would show the duration values of exercise corresponding to different states.

8.2.5. Sentiment Analysis

The “Sentiment Analysis” class contains several functions that are used to calculate the sentiment value of the tweet text corresponding to different states and types. In addition, there are some show functions to display all the related charts on the user interface.
Mood calculation functions:
moodType() This function is used for calculating the mood value of tweets for different exercise types according to user’s tweet text.
moodState() This function is used for calculating the mood value of tweets in different states according to user’s tweet text.

Show functions:
showMoodState() This function is used for displaying the state map that would show the happiness degree corresponding to different states.
showMoodType() This function is used for displaying the analytical chart that would show the mood values corresponding to different exercise types.

8.2.6. Correlation Calculate

The “CorrelationCalculate” class contains several functions that count the number of the tweets that refer to food corresponding to different states and exercise types. There are also several functions that would calculate the linear regression value among the tweets count of health, exercise and food. Besides, there are show functions to display all the relationship in charts on the user interface.

Food calculation functions:
countFoodType() This function is used for counting the number of tweets related to different kinds of food according to multiple types of exercise.
countFoodState() This function is used for counting the number of tweets related to different kinds of food according to different states.

Correlation functions:
correlationHealthFood () This function is used for calculating the linear regression value between the counts of tweets related to health and food.
correlationExerciseFood () This function is used for calculating the linear regression value between the counts of tweets related to exercise and food.
correlationExerciseHealth() This function is used for calculating the linear regression value between the counts of tweets related to exercise and health.

Show functions:
showCorrelationExerciseHealth() This function is used for showing two line charts that would indicate the count number of tweets related to exercise and health separately.
showCorrelationExerciseFood() This function is used for showing two line charts that would indicate the count number of tweets related to exercise and food separately.
showCorrelationFoodHealth() This function is used for showing two line charts that would indicate the count number of tweets related to food and health separately.

8.2.7. Personal Analysis

The “PersonalAnalysis” class is consisted of the following four variables that store the user’s personal information.

username String variable storing the user’s name.
state String variable storing the user’s current location.
age Integer variable storing the user’s age.
exerciseType String variable storing the user’s favorite exercise type.

This class include functions that are used to give out the personal suggestions based on the information the users provide.

setUserProfile (screenName : String, userAge : int, userState : String, exerciseType :
String) This function is used for setting the basic information that is needed for the analysis.

userAnalysis() This function is used for making analysis on the proper healthy food and exercise duration time based on the information the users provide.

showSuggestion() This function is used for displaying the proper personal suggestions based on the analysis.

showTweetHistory() This function is used for displaying the past tweets the user posted and showing the time when the tweet has been created.

8.2.8. Demography Analysis

The "Demography Analysis" class has some speculations functions that use third party's API to make speculations on the demographical information of the Twitter users based on their tweets, some update functions that update the speculation results into the existing table, and some show functions that display the analytical results.

connectAPI(url : String) This function is used for connecting our database to the third party's API. The parameter "url" is the website of this API.

guessGender(screenName : String, username : String, description : String) This function is used for making a guess on the gender of a certain user based on his/her screen name, user name and the user description.

guessAge(screenName : String, username : String, description : String) This function is used for making a guess on the age of a certain user based on his/her screen name, user name and the user description.

guessType(screenName : String, username : String, description : String) This function is used for making a guess on the user type, organization or person, of a certain user based on his/her screen name, user name and the user description.

updateGender(gender : String) This function is used for updating the guess result of the user gender in the existing users table.

updateAge(gender : String) This function is used for updating the guess result of the user age in the existing users table.

updateType(gender : String) This function is used for updating the guess result of the user type in the existing users table.

showAgeDistribution() This function is used for displaying the age distribution among all the users who have posted exercise-related tweets.

showGenderDistribution() This function is used for displaying the gender distribution among all the users who have posted exercise-related tweets.

showAreaAge() This function is used for displaying the age distribution among users who have posted exercise-related tweets in different states.

showAreaGender() This function is used for displaying the gender distribution among users who have posted exercise-related tweets in different states.

showTypeAge() This function is used for displaying the distribution of different exercise types in different age groups.
showTypeGender() This function is used for displaying the distribution of different exercise types in different gender groups.

8.2.9. Controller

The “Controller” class represents as a vital part in the system-to-be. All the executions of its functions are linked to some other classes. The user utilize these functions and receive the analytical results corresponding to his/her choice.

analyzePersonal() This function is used for making analysis based on personal information we obtained form the user.
analyzePublic() This function is used for making analysis based on public information we obtained from the Twitter.
updateDB() This function is used for making updates on the existing database if some changes need to be made.
isValidData() This function is used for verifying whether a certain piece of data is valid. If valid, return true, else, return false.
showRequest() This function is used for displaying corresponding analytical charts or tables based on the request from the user.

8.2.10. Communicator

The “Communicator” class is linked to all the other classes that command to get data
from the database. It has two variables storing the type of the platform and which feature table we need to use.

deviceType Integer variable storing the type of the platform the JSON data need to be sent to. We choose 0 representing the web platform, and 1 representing the IOS platform.
dataSelect Integer variable storing which feature tables we need to use for analysis. Different numbers represent different feature tables.

This class contains a jsonSender() function which is essential to send the database data to other platforms in JSON formation. Other functions are used for connecting the database.

setDataSelect(data : int) This function is used for setting which feature table we need to use for our analysis.
setDeviceType(device : int) This function is used for setting the type of platform we use to display the analytical results.

jsonSender() This function is used for packaging the data we need in certain tables and sending them to the platform we choose.

dbConnection() This function is used for making connection with the database.

dbConfig() This function is used for initializing the database.

isConnected() This function is used for testing the connection to the database. If connected, return true. Otherwise, return false.

8.3. Traceability Matrix

Domain concepts	controller	communicator	Database	Personal Analysis	Demography analysis	Duration Analysis	Frequency analysis	Tweet heat count	Correlation calculate	Sentiment anaLysis
Controller										
Profile Storage	x									
Interface Services				x	x					
Interface Pages				x	x	x	x	x	x	x
Display Options				x	x	x	x	x	x	x
Treated information				x	x	x	x	x	x	x
DB connector			x							
Data Analyst		x	x							
Third part connector		x	x	x	x	x	x	x	x	x
			x	x	x	x	x	x	x	x

Table 8-1 Traceability matrix

Apparently, there are many classes evolved from the same Domain Concept except the Controller, which is basically achieved by the single class that also named Controller. Concept Profile Storage include user’s personal data (like demography information). These information could derived from Classes Personal Analysis and Demography Analysis since both of the classes contain related methods to infer and output such message.

Interface Services, Interface Pages and Display Options construct the user interface part of the system, hence the last seven classes derived from them all.

Because the treated information is basically stored in the database of our system, there are not many classes involved with this concept. Only the class Database is needed to get such information.

Both of the two classes ---Communicator and Database ---take the responsibility of Concept DB Connector. Due to some specific reasons, we separate them by different functions. Database takes charges of the data management within the data base. Communicator transfers the data between other classes and the database.

Data Analyst is separated into seven classes that related to specific types’ analysis and two classes that enable them to do communication with database.

Because the treated information is basically stored in the database of our system, there are not many classes involved with this concept. Only the class Database is needed to get such information.

Both of the two classes ---Communicator and Database ---take the responsibility of Concept DB Connector. Due to some specific reasons, we separate them by different functions. Database takes charges of the data management within the database. Communicator transfers the data between other classes and the database.

Data Analyst is separated into seven classes that related to specific types’ analysis and two classes that enable them to do communication with database.

8.4. Object Constraint Language (OCL) Contracts

Important contracts for classes and their operations:

For Database class:

context: Database::parseTweetsData(JSON : String[]) : void

PersonalAnalysis::personalAnalysisStatus = false

FrequencyAnalysis::frequencyStatus = false

InterfaceAnalysis::interfaceGenderStatus = false

InterfaceAnalysis::interfaceTypeStatus = false

InterfaceAnalysis::interfaceAgeStatus = false

CorrelationCalculate::correlationStatus = false

TweetHeatCount::tweetHeatStatus = false

SentimentAnalysis::moodStatus = false

context: Database:: getJSONData() : String[]
For Communicator class:
context: Communicator:: jsonSender() : void
context: Communicator:: setDataSelect(data : int) : void
context: Communicator:: setDeviceType(device : int) : void
context: Communicator:: dbConnection() : void
For PersonalAnalysis class:
context: PersonalAnalysis::setUserProfile(screenName : String, userAge : int, userState : String, exerciseType : String) : void
age = userAge
state = userState
exerciseType = exerciseType
context: PersonalAnalysis::userAnalysis() : void
For DurationAnalysis class:
context: DurationAnalysis::durationState() : void
For FrequencyAnalysis class:
context: FrequencyAnalysis::frequencyState() : void
For DemographyAnalysis class:
context: DemographyAnalysis::updateGender(gender : String) : void
context: DemographyAnalysis::updateType(type : String) : void
context: DemographyAnalysis::updateAge(age : String) : void
For CorrelationCalculate class:
context: CorrelationCalculate::correlationAnalysis() : void
System Architecture and System Design
For TweetHeatCount class:
context: TweetHeatCount::tweetHeatAnalysis() : void
For SentimentAnalysis class:
context: SentimentAnalysis::moodAnalysis() : void

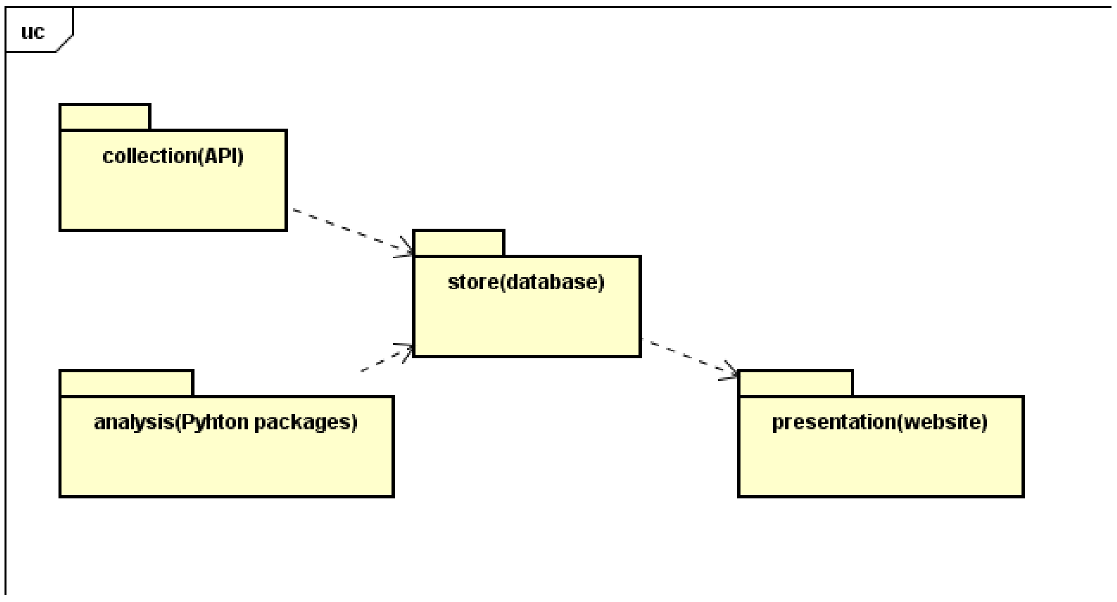
9. System architecture and system design

9.1 Architectural Style

Software architecture refers to the high level structures of a software system, the discipline of creating such structures, and the documentation of these structures. It is the set of structures needed to reason about the software system. Let's begin description from how our system works. Our system loads and stores data from Twitter to data base, analyzes the data for different business features using python implements, finally presents the results to users on websites and apps. The users should be able to access our system to find what they want and log in their own account, but only administer can conduct the data processing job. At first phases, our system uses API get data from Twitter, store them on local database, and use python language packages to analyze these data. Then, allow user to login in our own website and upload their exercise or health data, this website can also be shared to Twitter, which can attract more users. So there will be two major architectures, one is the database, including data from API Twitter and analysis implement, the other is presenting applications such as website and cellphone apps, which can show our work and provide interface to users. There will be two parts involved, users login in, do exercises, update data and tweet, while administer runs database, website and analyzing data form API.

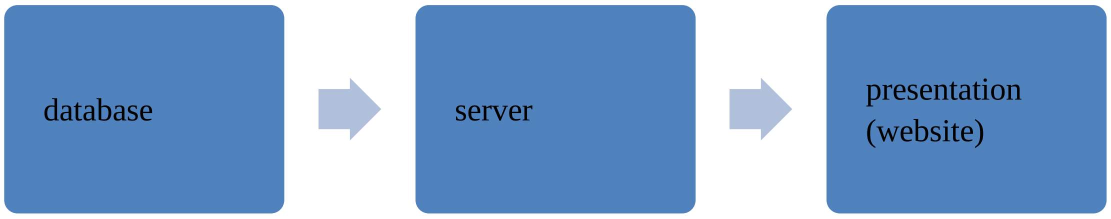
9.2 Identifying Subsystems

As discussed above, we can define our subsystems as four parts to achieve basic requirement, we can add more features later to this structure.



The collection subsystem is responsible for scrawling data from Twitter to our database. The data will be in .sql format and stored directly in a table in our database. The store subsystem will parse (extract) the properties such as screen name, date, tweet from API and insert the properties into other tables. The analytics subsystem will compute results from the data stored in the tables mentioned above in order to fulfill the different business goals, and insert the results in feature tables. The presentation subsystem simply structures the user interface using a website which is now already been built up. We still use .sql format for transmission, because the html can be accessed with sql through php tools. The presentation will not get all the feature data at the initialization stage since data is large and loading is slow. The presentation will only select what is wanted to be shown.

9.3 Mapping System to Hardware



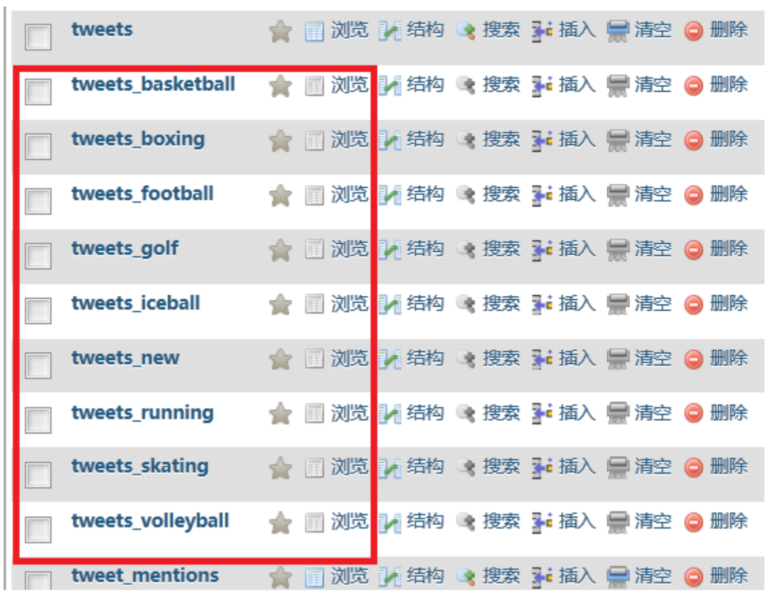
In our system, database and server should be on one computer, while website can be accessed from any computer. Server means the data processing and analyzing tools and execution.

What is the relationship between the subsystems and the hardware? The collection, store, analytics and transmission subsystems will be deployed on the server. The computer for database only stores the data, and will be accessed by the server. If it is a web application, the presentation and plot subsystems will be also deployed on the computer for server. The user interface structure and the data for plot will be downloaded by the browser. While accessing the website, the computer in which database and server should be open and connected to internet. Further technical discussions about host accessing websites will appear later.

9.4. Persistent Data Storage

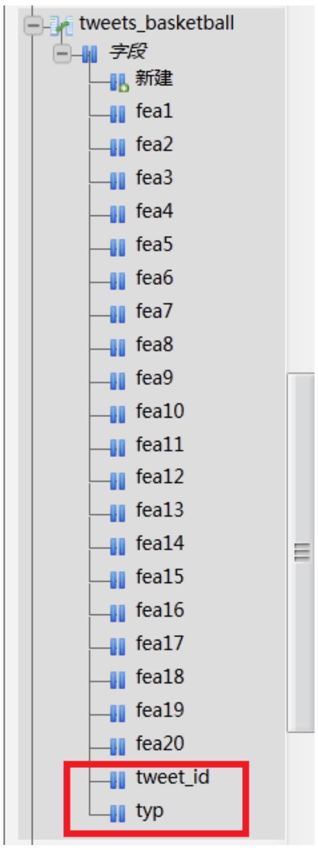
9.4.1 Database improvement

Since our project is based on the previous project, so we decided to modify and improve their database rather than creating a brand new database. So far, as we designed, we want to separate every different sport and classify them into two different categories. So in figure 3-4-1,we create tables according to the design.



tweets	浏览	结构	搜索	插入	清空	删除
tweets_basketball	浏览	结构	搜索	插入	清空	删除
tweets_boxing	浏览	结构	搜索	插入	清空	删除
tweets_football	浏览	结构	搜索	插入	清空	删除
tweets_golf	浏览	结构	搜索	插入	清空	删除
tweets_iceball	浏览	结构	搜索	插入	清空	删除
tweets_new	浏览	结构	搜索	插入	清空	删除
tweets_running	浏览	结构	搜索	插入	清空	删除
tweets_skating	浏览	结构	搜索	插入	清空	删除
tweets_volleyball	浏览	结构	搜索	插入	清空	删除
tweet_mentions	浏览	结构	搜索	插入	清空	删除

Figure 9-4-1. Adding tables



```
tweets_basketball
  字段
    新建
    fea1
    fea2
    fea3
    fea4
    fea5
    fea6
    fea7
    fea8
    fea9
    fea10
    fea11
    fea12
    fea13
    fea14
    fea15
    fea16
    fea17
    fea18
    fea19
    fea20
    tweet_id
    typ
```

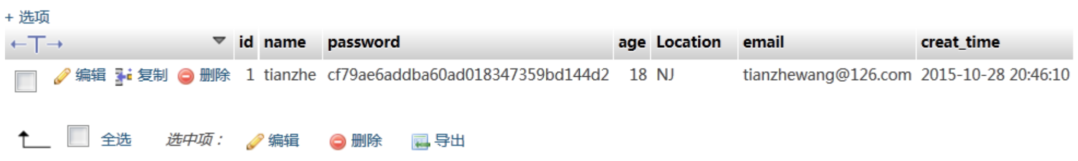
In the table, for example, in figure 3-4-2, tweets basketball, we design 20 features to further analyze, we relate every tweeter using 'tweet_id' to the original tweeter in the table 'tweeter'. And the type is the result after our analyzing.

9.4.2 Registered users

Also, we decided to add a table to record the registered users, although we didn't give the very detail information about the difference between signed users and unsigned users, but we all think we it's necessary, so we design the database structure in advanced.

In the figure 3-4-3, we create a tables named 'registed_users' to record every info for user data.

Figure 9-4-2. Table Value



+ 选项

	id	name	password	age	Location	email	creat_time
编辑 复制 删除	1	tianzhe	cf79ae6addba60ad018347359bd144d2	18	NJ	tianzhewang@126.com	2015-10-28 20:46:10

全选 选中项： 编辑 删除 导出

Figure 9-4-3. Registered users.

9.5 Network Protocol

Since our aim of project is to give a better analyze of data which comes from the GET method in HTTP protocol. So we distribute the responsibilities for every group member. So we defined two modules for our primary system. Server for display 1), server for functionality 2). In the display server, we create a website for users (registered or not) to browse, and finding information. In the functionality server, our analyze tool will be implemented on that. Of course these two servers should be in the LAN (local area network), while we think there is no need for both our them to connect to the internet for the safety concern. The separate server is our ideal design (database+operation), but for our real implementation and time limitation, we probably merge them into one.

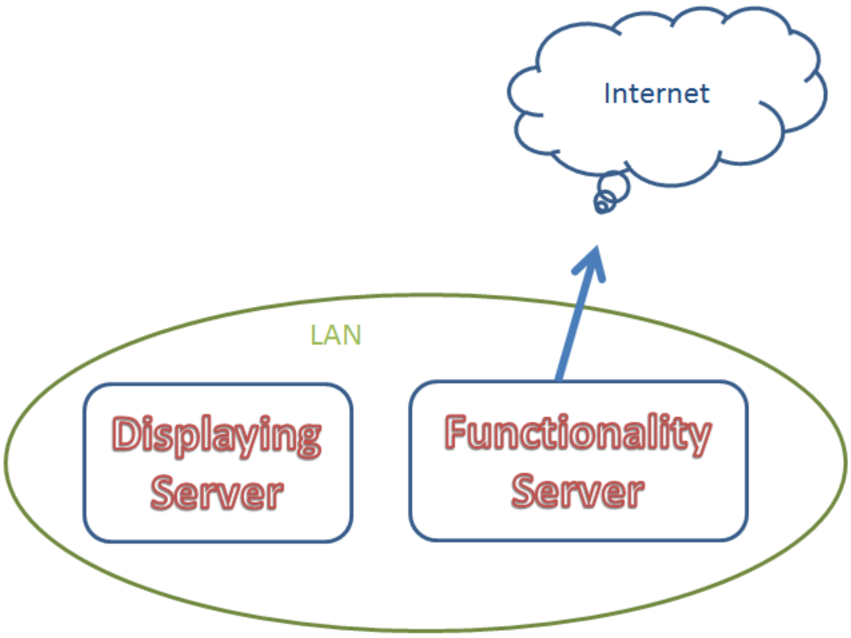


Figure 9-6-1. Network

9.6. Global Control Flow & Hardware Requirements

For the data management and analyze, window service is our first choice. Since it can start with every time server power on, and can hide from users, easy to control. The service include: data mining, data analyze, database action.

As for hardware, we set the same aim as last project (project 2, 2014): 5 GB. But, we respectively disagree the 5G parameter set by them can be achieved. Since they obtain every tweeter in the database without any clearing schedule, we think the database will be large than our expectation and the system will be slow. So after our considering, we decide not to obtain every tweeter, only record the result we wanted, to simply the database like figure 3-6-1.

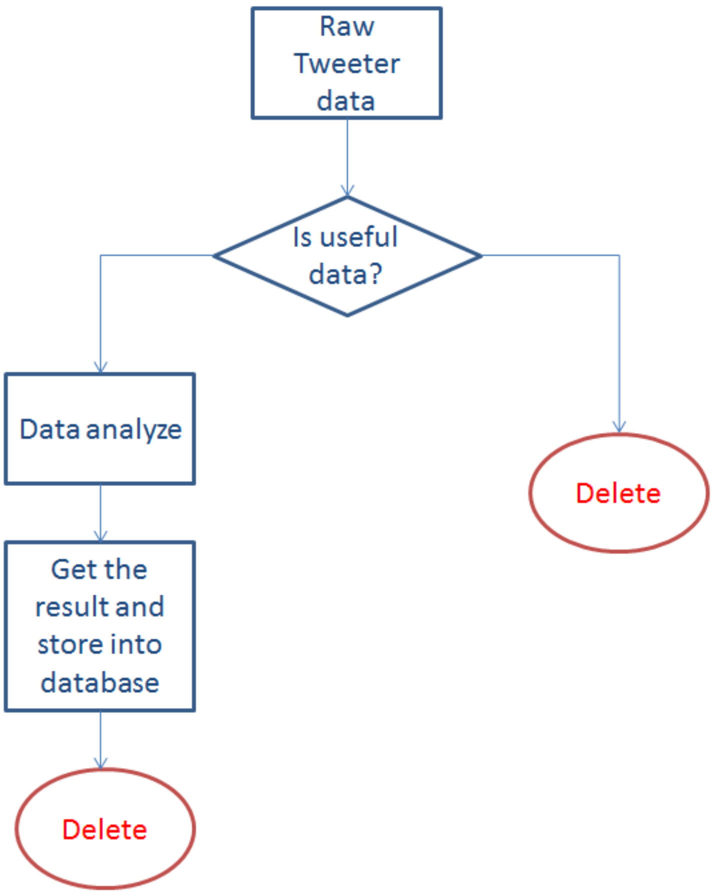


Figure 9-6-1 Database Simplification

10. Algorithms and Data Structures

Since we try our best to level up the accuracy of the tweeter content, we implement KNN algorithm to filter the raw tweeter data and NTLK testBlob to analyze the emotional trend, also we use SVM to classify each tweeter into two categories.

In pattern recognition, the k-Nearest Neighbors algorithm (KNN) is a non-parametric method used for classification and regression. In our project, we use KNN algorithm for tweets classification that is to determine every tweet is related to exercise and sports or not. We use training set to train our KNN algorithm. Then we run the PHP program to fetch tweets from our local database, filter them and then insert relative tweets into a new table in our local database.

We use the TextBlob to implement the emotional analysis. TextBlob is a Python (2 and 3) library for processing textual data. It provides a consistent API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, and more. TextBlob stands on the giant shoulders of NLTK and pattern, and plays nicely with both.

In our project, we use TextBlob to do the emotional analysis on every exercise and sports related tweets and classify them into either a positive tweet or a negative one. By doing so, we are able to see how many of people talking specific sports are having negative feelings.

The code below is a simple Python example program using TextBlob API to get emotional analysis on tweets.

We use SVM (support vector machine) to categorize every tweeter into two categories. SVM is an advanced technology using training data to build model to predict the incoming data (more information about SVM, you can refer other books about it). In this early stage of our project, we just test the sexuality according to the height and weight (since they are all numeric character).

There are many SVM tools online, we choose ‘libSVM’ created by Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin, Department of Computer Science, National Taiwan University. We choose it since its easy-using and nice guide manual. There are many version of it, we choose c++ version since it is more familiar with knowledge scope.

10.1 KNN

KNN is a very important algorithm in our project. We use KNN classification to filter out all the nonrelative tweets we do not need.

10.1.1 Basic Concept of KNN

In k-NN classification, the output is a class membership. An object is classified by a majority of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small).

In k-NN regression, the output is the property value for the object. This value is the average of the values of its k nearest neighbors. The training examples are vectors in a 3 dimensional feature space, each with a class label. In this project, the labels are only two: exercise relevant and exercise irrelevant. Relevant tweets shall be stored in the database while the irrelevant ones are discarded.

The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. This is done by manually reading the tweets and labeling them with “True” or “False” class.

In the classification phase, k is a pre-defined constant. The k for text identification in this project should be chosen as 3. As mentioned in reference from TAMU [http://courses.cs.tamu.edu/rgutier/cs790_w02/l8.pdf], k = n1⁄2, for which n means the number of features.

In this project, there are 3 features: sports type, quantity and adjective list. The reason to choose these three types of keywords stated as:

1. Sports type is necessary since the tweet should be about health and exercise, any keywords that have direct relevance to sports and exercise should be considered first.
2. When examining the tweets on Twitter that are talking about health, it is more likely to have not only sports type keyword alone but also description about how much work or exercise the user has made. So the keyword of quantity will help classifying the tweets.
3. Adjective list is a supplement to give tweets that contain human’s emotion and feeling’s words.

An unlabeled tweet is classified by assigning the label that is most frequent among the k training samples nearest to that query point. Distance measurement shall be calculated as Euclidean distance:

$$D_{ij} = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2 + (Z_i - Z_j)^2}$$

The distance means how close a new tweet is related to the training sets, in this project, the distance is the measurement on probability similarity.

In the project, a 3D characteristic vector is extracted from every tweet using keywords finding. X, Y and Z stand for the keywords’ scores.

The score is calculated based on the probability of a tweet relevant to exercise by containing this keyword alone. To demonstrate this, take one tweet “I run 5 miles, tired.” for example. If in 100 tweets that contain keyword “run” and there are actually 10 tweets talking about exercise, the probability of “run” will be 0.1. And we re-arrange the probability into a range of 0~40 score, thus the score of “run” will be 40×0.1 = 4. Apply these procedures to other keywords, saying “miles” is score 10 and “tired” is score 15.

In the system, the probability table of keywords and training sets must be set preliminarily. This procedure is done manually by reading the tweets. Then the following steps shall do the data sifting procedure:

1. When the system crawls a new tweet, search for the keyword in it first and

extract the three types of keywords.

2. Score the keywords using established probability table.
3. Calculate the Euclidean distance of the newly crawled tweet's characteristic vector to the training sets using X Y Z coordinates.
4. Sort the distance and choose the top k points as the k-NN.
5. Find the majority in k-NN and identify the newly crawled tweet as the same label as the majority.

The classification can be explained through the chart above. There are some pre-set training sets in the coordinates with certain location. And the new tweet, for example, is "I run 5 miles, tired". It is scored by the three filters and locates in the coordinates. We have not decided whether the new tweet is true (health related) or false (health unrelated).

The next step is calculating the distance between the new tweet's coordinates to those of training sets. The training sets are also located in the coordinates with the same standard for keywords scoring, while their classifications of true and false are labeled manually. Using Euclidean distance for distance calculation.

Then sort the training sets by their distance to the new tweet, select top 3 training sets with minimum distance. And check their classification, if 2 or 3 of them are true, and then classify the new tweet as true, vice versa.

10.1.2 KNN Implementation

Our implementation is based on the former group's implementation.

We draw the training sets our project in this three dimensional chart. In the chart above, the blue points refer to training sets that are health related (true), and black points refer to health unrelated training sets (false). A new tweet is labeled as green, and the implementation is presented below.

10.1.2.1 Setting Training Set

One improvement we made is to enlarge the training set. The training set provided by the former group has a relatively small amount of training examples leading to low accuracy of the KNN filter.

We choose three types of keywords: sports type, quantity and adjective list as our 3D coordinates.

The axis for sports type, quantity and adjective list are scale from 0~40. This is calculated by the following step:

1. Calculate the probability of tweets that containing the keywords of sports type, quantity and adjective list those are health related separately. The result ranges from 0~1.0 (0~100%).
2. Times 40 to all the probability result and then the result ranges from 0~40. Because the minimum difference of probability of different keywords is 0.025 in our project, so to distinguish these two values of probability we times 40 to make the difference equal or greater than 1.
3. Round the result to the nearest integer to meet codes' int calculation. And the integer shall be the score of the keyword.

After setting up the axis, training sets are also located by calculating the keywords' coordinates (or score) in the training sets' text. X, Y and Z are acknowledged as well as location.

This method is to distinguish keywords of different health related tweets. If a new tweet saying "I do 40 squads" (new in the chart), it should be compared with a tweet that contains the same or close weighted keywords, like "I do 50 push ups" (Point 2), and "I have a great day doing push ups"(Point 3) instead of comparing to tweet that contains much different weighted keywords as "I run 5 miles".

After this procedure, the new tweets' score coordinates are scattered in the coordinates and ready to be compared with similar weighted tweets, which are used as training sets.

10.1.2.2 Classify new tweets

Then we calculate the Euclidean distance of new tweet and training sets. The new tweet's coordinates are calculated in the same way as the training sets.

Then sort the new tweet's neighbor training sets by the distances, choose the top k training sets. The k's value is 3 in our project.

The reason is that in the classification phase, k is a pre-defined constant. The k for text identification in this project should be chosen as k = n1⁄2, for n means the number of features. The reference is from TAMU.
[http://courses.cs.tamu.edu/rgutier/cs790_w02/l8.pdf]

In this project, there are 3 features: sports type, quantity and adjective list. Also it is better to set k as an odd number to avoid a tie of two different classifications. Thus we choose k = 3 as our parameter. The 3 training sets are circled in a red circle in the chart above.

After choosing the top 3 nearest training sets, we examine the classification of the training sets, if 2 or 3 of them are true, and then classify the new tweet as true, vice versa.

In the chart, the Point 1 is "She does her sit-ups to get out of bed". Though "sit-ups" is weighted near the "push ups", the whole meaning of the tweet is not about health or exercise. And after sorting the top 3 nearest training sets, the new tweet has two true nearest training sets and one false. Thus the new tweet is classified as true.

We do this sift procedure to each tweet we extracted from Twitter and only the true tweets are used for all other analysis while the false ones are discarded. We have collected 200 training sets and apply the algorithm to over 100000 tweets in our database.

10.2 NLTK Text Blob

TextBlob is a Python (2 and 3) library for processing textual data. It provides a consistent API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, and more. TextBlob stands on the giant shoulders of NLTK and pattern, and plays nicely with both.

In our project, we implemented the TextBlob API to get sentiment analysis on each tweets. The TextBlob provides multiple features for developers to use. So far we are using Naive Bayes Classification to classify each tweets into two section: positive tweets and negative tweets.

As long as we classified all sports and exercise related tweets. We could have a full sentiment analysis towards each sport or exercise activity. Government or other companies will be interested in these sentiment analysis results because they can figure out what problems or improvements exist so that they can take advantage of it.

4.3 SVM

In the SVM algorithm, we treat every tweet as a vector. Every words in the sentence (tweeter) can has a weight. We use the weight calculation formula (figure 4.3.1) to calculate each weight[1].

$$\phi_i(x) = \frac{\mathrm{tf}_i \log(\mathrm{idf}_i)}{\kappa}$$

Figure 10.3.1

Tfi is the number how many times certain words happened in file x (we take x as each tweeter content), idfi is the ratio of all the files (tweeters) to the number how many files contain certain words.

We don’t try to eliminate the stop words or other function word like ‘the’. The reason is, suppose two tweeter is all “the the the the the” which has no meaning. They will make these two testing point stay at the ‘dividing line’ like figure 4.3.2.

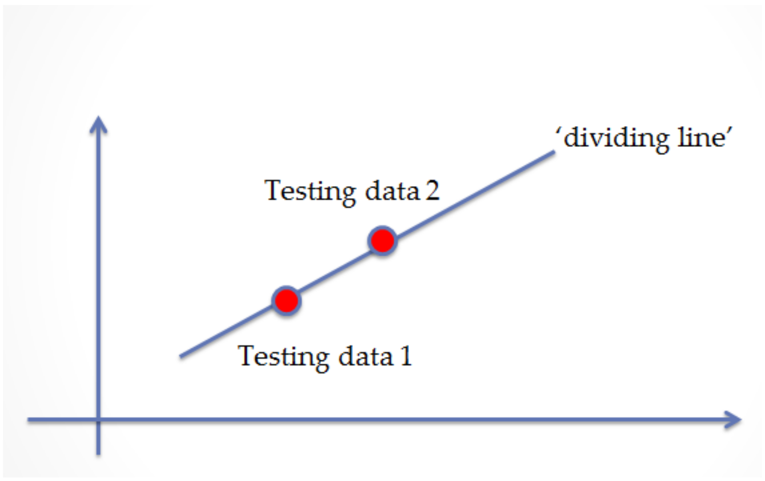


figure 10.3.2.

But in the real word, it will not happen. There will be some key word which has high weight to indicate which category the tweet should be in. just like show in the figure 4.3.3.

[1]‘An Introduction to Support Vector Machine and Other Kernel-based Learning ’ Nello Cristianini and John Shawe-Taylor

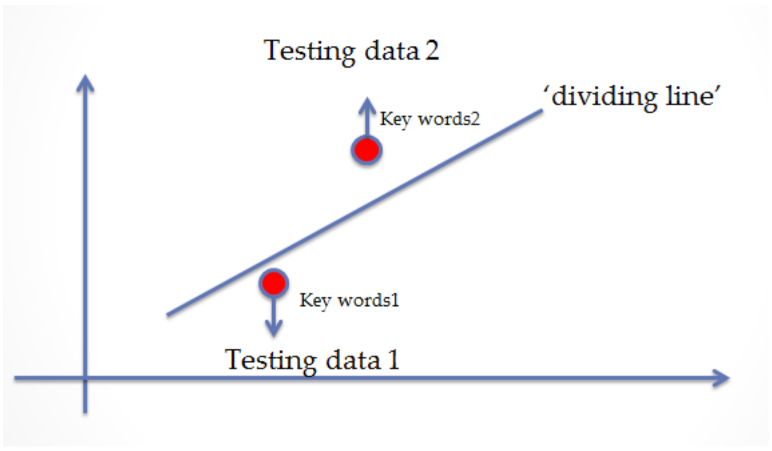
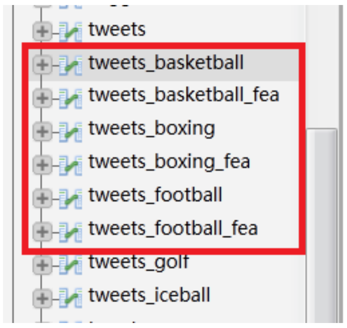


figure 10.3.3.

Since the algorithm behind SVM is very complicated, so we don't want to illustrate them here (it may need a book to tell them clearly). For more information and knowledge about SVM, please refer to 'An Introduction to Support Vector Machine and Other Kernel-based Learning ' Nello Cristianini and John Shawe-Taylor, or other material.

10.4 Data Structure

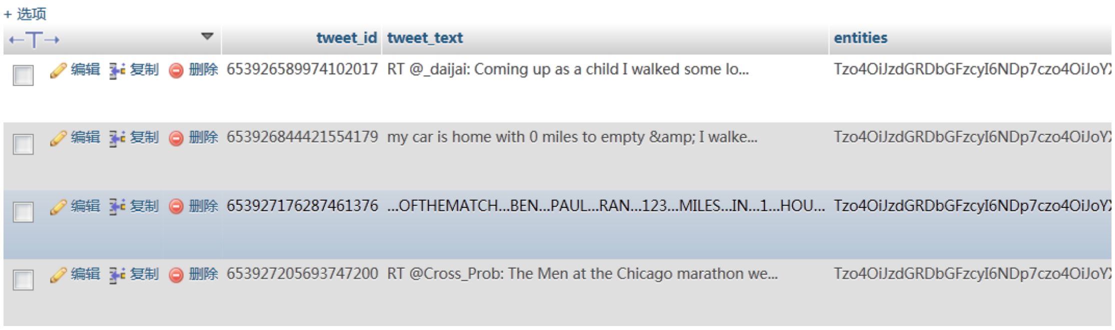
Our database is based on the previous database, but we do some improvement. We create tables for each specific sports, and tables for storing the feature weight. As shown in figure 10.4.1



- tweets
- tweets_basketball
- tweets_basketball_fea
- tweets_boxing
- tweets_boxing_fea
- tweets_football
- tweets_football_fea
- tweets_golf
- tweets_iceball

figure 10.4.1

We have the table for raw data from Tweets, store into tweet tables. As shown on figure 10.4.2



+ 选项

←T→	tweet_id	tweet_text	entities
编辑 复制 删除	653926589974102017	RT @_daijai: Coming up as a child I walked some lo...	Tzo4OiJzdGRDbGFzcyI6NDp7czo4OiJoY
编辑 复制 删除	653926844421554179	my car is home with 0 miles to empty &amp; I walke...	Tzo4OiJzdGRDbGFzcyI6NDp7czo4OiJoY
编辑 复制 删除	653927176287461376	...OFTHEMATCH...BEN...PAUL...RAN...123...MILES...IN...1...HOU...	Tzo4OiJzdGRDbGFzcyI6NDp7czo4OiJoY
编辑 复制 删除	653927205693747200	RT @Cross_Prob: The Men at the Chicago marathon we...	Tzo4OiJzdGRDbGFzcyI6NDp7czo4OiJoY

Figure 10.4.2

After the KNN filter, we filter the useless tweeter and store the valuable ones into separate tables. Then we use NTLK to calculate the weight for each words using the formula in the section 4.3. the whole data processing can be seen like in figure 4.4.3

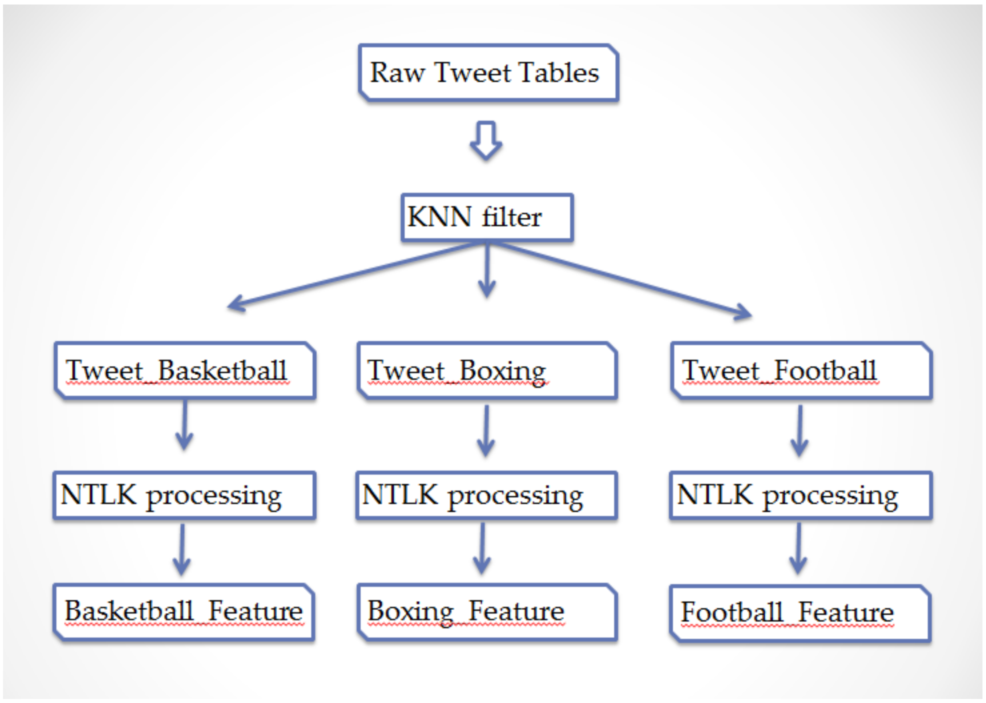


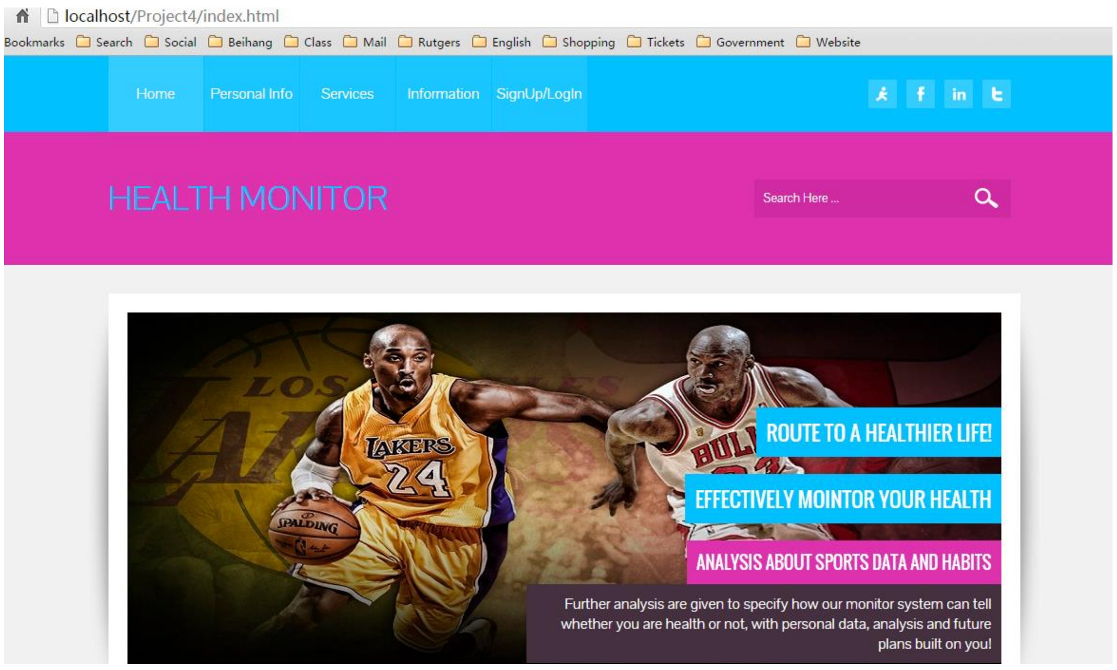
Figure 10.4.3

After we get the result about which category each tweeter belongs to, we can calculate the sum for each category for further data analyze and display.

11. User Interface and Implementation

For now most of our job is displayed on the basis of website. In this part we will explain how to use our website, step by step. Our goal is to help user to use this health monitor system. Since this is the very first version, so we can just provide some really basic functions, but more and more new features will be added, as well as more accurate data will be implemented.

11.1 Homepage

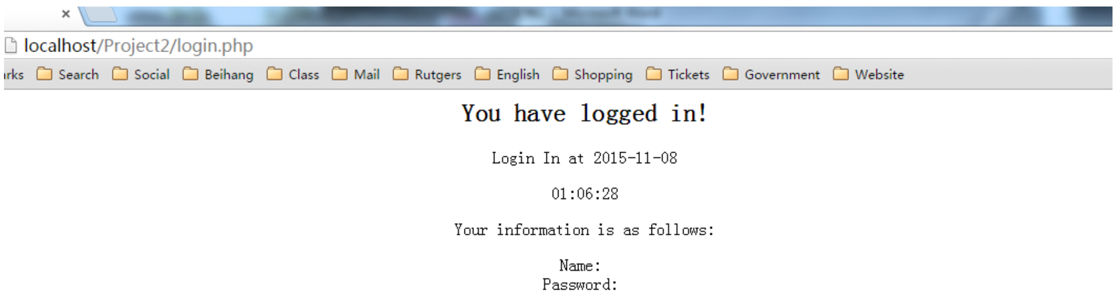


localhost/Project4/index.html
Bookmarks Search Social Beihang Class Mail Rutgers English Shopping Tickets Government Website
Home
Personal Info
Services
Information
SignUp/LogIn
HEALTH MONITOR
Search Here ...
ROUTE TO A HEALTHIER LIFE!
EFFECTIVELY MOINTOR YOUR HEALTH
ANALYSIS ABOUT SPORTS DATA AND HABITS

Further analysis are given to specify how our monitor system can tell whether you are health or not, with personal data, analysis and future plans built on you!

In local host, which is our projects root route, is our index html page. On the top, there are three sub-pages directing to three sub sections, which are sports, health and about. We will come to these sub-pages later. On the left side, there are log in and sign up bar to choose either choice. In general, we have two ways to extract data from the users, one is to get data from twitter with API and analysis with algorithms, this is the main way to conduct and get re results, the other source of data is to let user to sign up their own account and add their data to those data from twitter for analysis. So we can see that we do need a sign up page.

11.2 Login and Signup page


localhost/Project2/login.php

rks Search Social Beihang Class Mail Rutgers English Shopping Tickets Government Website

You have logged in!

Login In at 2015-11-08

01:06:28

Your information is as follows:

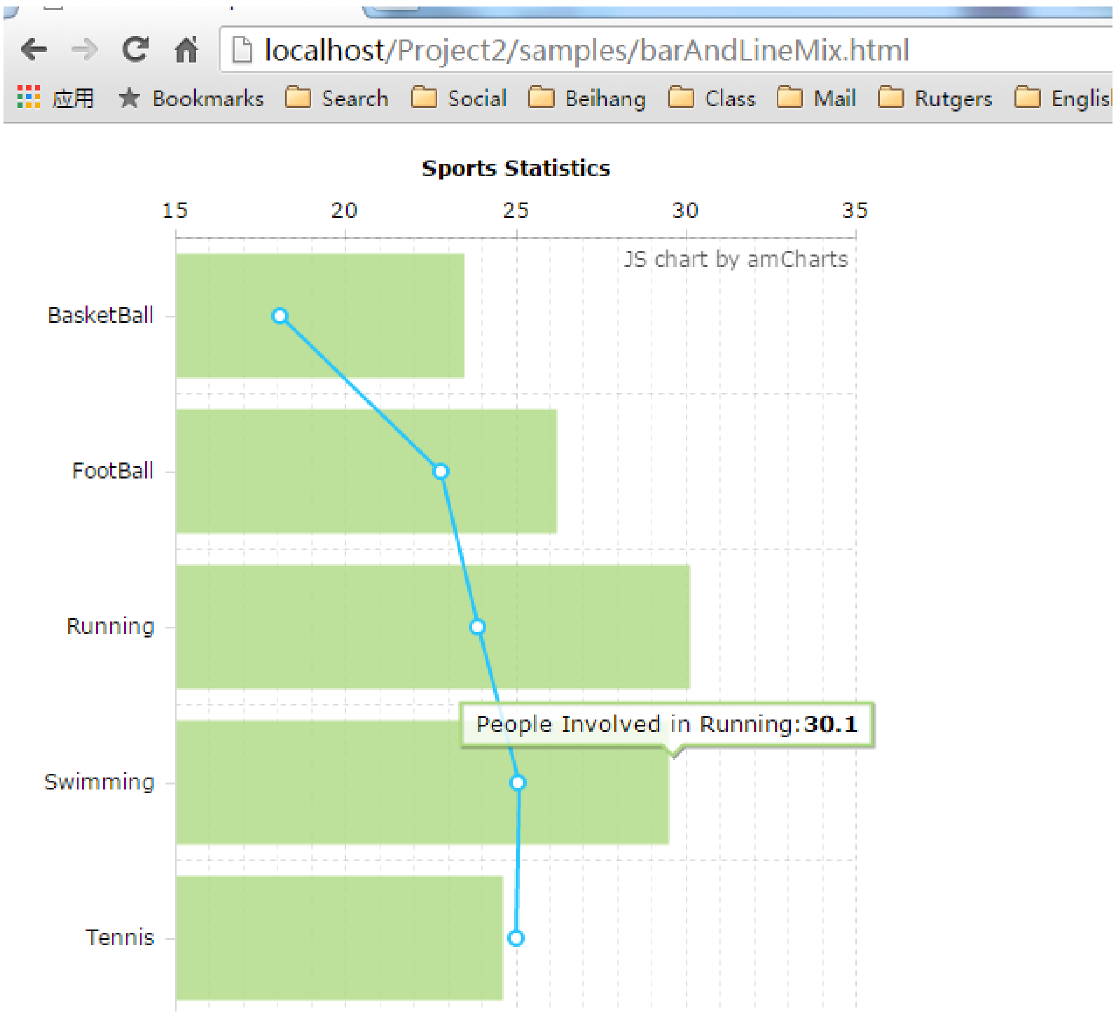
Name:
Password:

From this screenshot we can see that it is using php to achieve submitting the user's data, and there are indexes such as time or date. In next stage, we will enrich our database for a new and more efficient table to store the user's data such as username or password. To have this table, we first have to let user to input the data, which is sign up, after sign up, the user's data are stored in the database, when they log in, what is input are compared with the database, if match, then log in success.

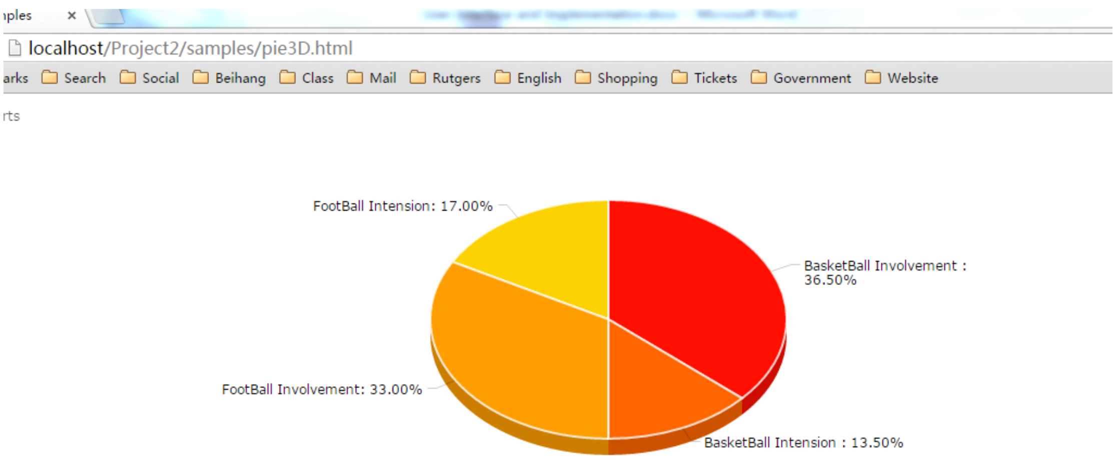
This is the sign up page. Users type in their basic information, and initialize a table in the table in database called by the user's id defined in the database.

11.3 Sports page

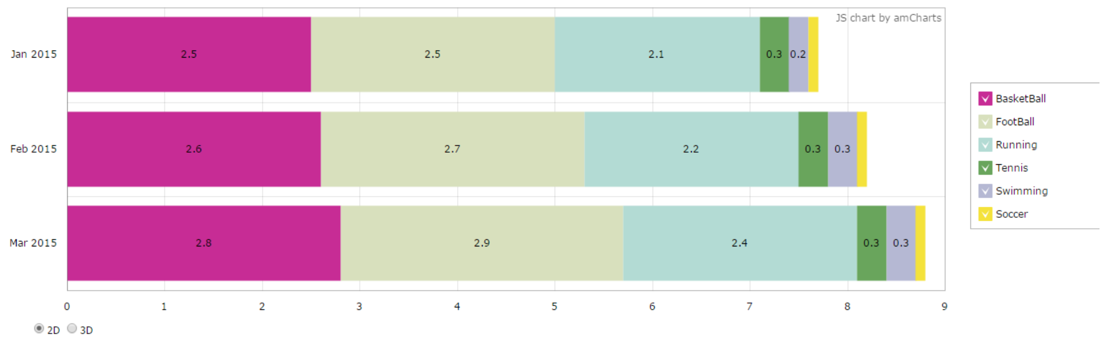
The sports page is mainly about the results of data analysis. The data analysis results, is mainly divided into two parts, which to our definition, is sports intension and sports involvement. Intension is people talk or tweet about specific kind of sports, take basketball for example, someone tweets about basketball and said watching it but not playing it, we can use our algorithm to analyze that and use figure to show this result. We can also see from the charts that if some kinds of sports are not "easily physical accessible", such as swimming, there are not too much people talk about it but there are many people actually swimming, then we can say that people's involvement of swimming is higher comparatively. These data, coming from the database, is linked from php selection, and then use javascript to display.



Above is the chart showing how the result is displayed.



Above is the ratio of two kind of sports’ intension and involvement.

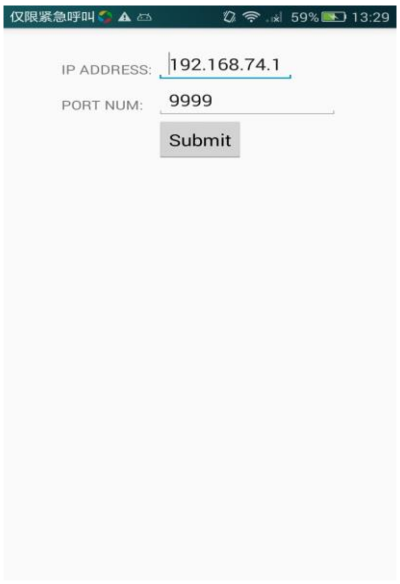


Our website can also display the results of data during some time, although it is not that accurate, we do select some data from each month and analysis them to display. For the next phase of work, we will figure out more straightforward description to show the results.

11.4 Android

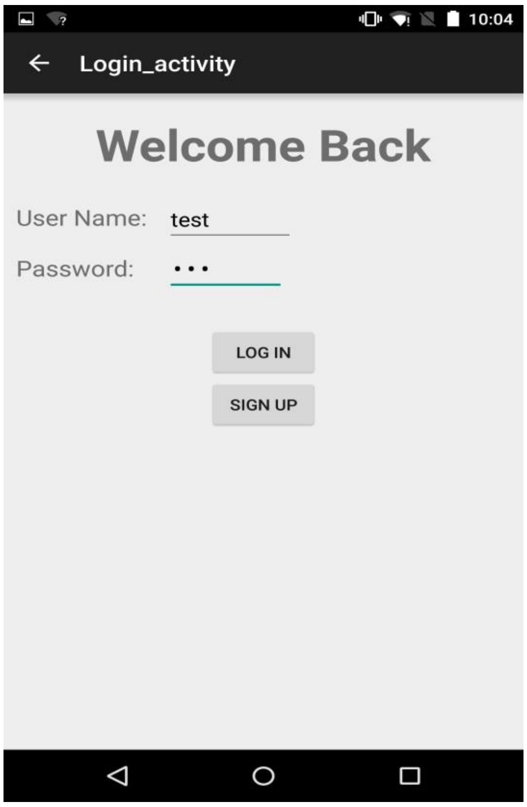
In this android application, I successfully created a log in system and successfully connected the mobile app with our data base through a server. All the data that is in this application is loaded from our database.

This is what you will see after opining the application.



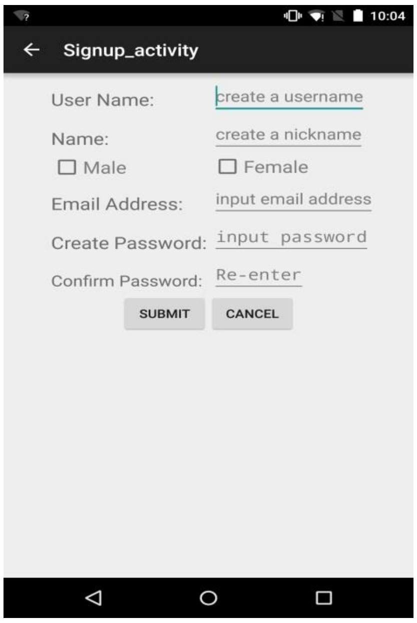
Because this app is now used by the developer which is us, we want this app to be able to connect to different servers in the future. So at the start of this application, we will be able to choose which server we want to connect to so that it can connect to different databases. For now of course, it only connects to our own database.

And this is what you will see after successfully connecting to the server. The log in page will show up.

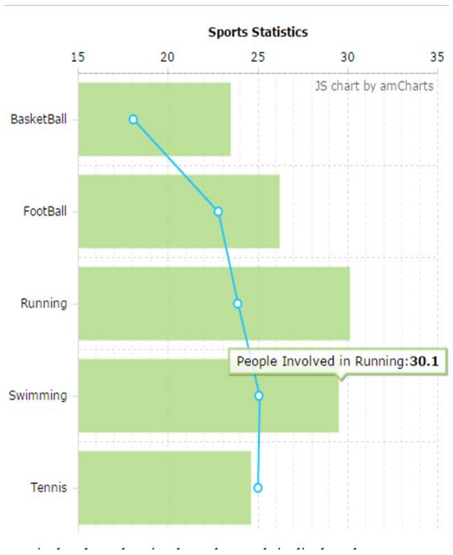


Of course, all the account information are stalled in our database. This step will check if the things you have typed in fit what is stalled in the data base.

If you have not registered yet, just push the sign up button and the sign up page will show up. This step will upload the user information to the database to stall.


10:04
Signup_activity
User Name: create a username
Name: create a nickname
Male
Female
Email Address: input email address
Create Password: input password
Confirm Password: Re-enter
SUBMIT
CANCEL

After logging in, you will be able to see the statistics showing in the form of charts.



12. Design of Tests

12.1 Overall Description

The part of design of tests is to test the basic function of our system. The test will be divided into two sections: the function unit test and the Integration testing strategy. Because some function units use the same coding methodology, therefore we group them as a class and choose one unit to test. The test table is shown in table 12-1.

Function unit group	Unit within group	Test unit
Twitter Data Capture	Twitter retrieve	Twitter retrieve
Data base set up	Data base	Twitter retrieve
Tweets data filter	Useful tweets and advertising	related and unrelated tweets
Related data selection	Self-involve and “audience” tweets	self-involve and “audience” tweets
Histogram show	Heat map and Sports intension&involvement	Heat map and Sports intension&invlovwment
Exercise hit trend	Exercise history	exercise history

Table 12-1

12.2. Function unit test

12.2.1 Test Unit : Twitter retrieve

Tweets are the basic data source for our system. They are captured by the streaming twitter API. We use two php profile to capture data. OauthPhirehose.php and Phirehose.php achieve the function.

Input Requirement	Mysql database is running. Apache webserver is running. New database is created in Mysql database. Account and pin of twitter API is applied. Then, run the php profiles.
Expected Output	Several tables are created in the database already created. Each table contain the required tweets data such as location, created time and etc.
Succeed/ Fail	Succeed if the right data is stored in corresponding database and format is right. Fail if there are no table created in database or no required data in tables.
Comment	This is to test whether the necessary could be collected successfully

Table 12-2

12.2.2 Test Unit : Data base

Input Requirement	JOSN File Extract From Twitter
Expected Output	Tweet Text, User Profile, other Information that could be useful for system requirement
Succeed/ Fail	Succeed if Data Base Set Up With Specific Data Fail if Data Base Set Up With Other Data

Comment	This test is to make sure that our database will contain the exact data extract from twitter

Table 12-3

12.2.3 Test Unit : Useful tweets and advertising

After tweets are stored, the system need to sort and filter the data from the database and create two new tables. They are used for further achievement of specific function of system. And they contain the necessary data for other functions.

Input Requirement	Mysql database is running. Apache webserver is running. Tweets are stored in the corresponding database. Then run datafilter php profile.
Expected Output	New tables which contain useful tweets are created in the database for system. The format and content in each table are right. If there are more than one database for our system, every database would have two more tables.
Succeed/ Fail	Succeed if two new tables are created in corresponding database. Fail if the connection to database is timeout or tables cannot be created for every database.
Comment	This is to test whether the system could filter the data from original database and be prepare for further functions.

Table 12-4

12.2.4 Test Unit :Self-involve and “audience” tweets

The data filtered is ready for analyzing and processing in functional units. DataSelect is implemented to select request data from database for different functions. It have to confirm successful connection to database and select out right data. Then it will bring proper information or data to the specific function analyzing units.

Input Requirement	Mysql database is running. Apache webserver is running. Tweets are filtered in the corresponding database. Then run dataselect python programs.
Expected Output	Connect to database steadily with account and password. Select data out and store them into different sorted arrays. So the system will get two kind of datas which one is about self-involving exercise and the other one is about watching or discussing sports.
Succeed/ Fail	Succeed if each arrays are print out in the browser and the statistic appears is right. Fail if there is nothing in the browser or wrong is displayed in the browser.
Comment	This is to test whether the system could select the required data and ready to be called by functions.

Table 12-5

12.2.5 Test Unit :Heat map and Sports intension&involvement

Histogram is one key function of our system. User could see these tables or charts after choosing to click histogram button on screen.

Input Requirement	Mysql database is running. Apache webserver is running. Data is ready for draw real-time histograms. Internet connection to Google map api is running. Then run these section code.
Expected Output	Web page for displaying histograms is opened. histograms is set in defined space and basic map function such as enlarge and shrink is normal. Three kind of histograms show regularly. They are heat map for showing the popularity of sports, sports intension among area’s residents and sports involvement degree.
Succeed/ Fail	Succeed if histograms display and all sub-functions run normally. Fail if the web page cannot be opened or displays of histograms are wrong or no specific markers on map.
Comment	This is to test whether the histograms could show and all functions could be switch randomly and run steadily.

Table 12-6

12.2.6 Test Unit :Exercise history

Collecting more data and categorize them by different slots(i.e. morning/afternoon/evening)

In this feature, we are able to analyze what is suitable for exercising dependent to specific time. By reaching this point, user is allowed to appoint where and when is most suitable for joining in others exercise event. For example, if someone is willing to play basketball at 8pm, but cannot gather enough teammates to play a game. Then maybe he/she can find some information that show the location where other people played most of times before a few days at 8pm.

Input Requirement	Mysql database is running. Apache webserver is running. Data source is ready for Trend plotting.Then run the Trend profiles and click on Trend button.
Expected Output	Web page for displaying histogram is opened. Trend set in defined space. Different line with different color appear in the same chart.

Succeed/ Fail	Succeed if the line chart appears. Values and description of axis are accurate. Fail if the web page cannot be opened or there is no trend chart display or the chart cannot show all required statistics.
Comment	This is to test whether the trend is ploting right and implement in the right space on webpage.

Table 12-7

12.2. Test Coverage

Test coverage is a method and measure used to describe the degree to which the source code of a program is tested by a particular test suite. Our tests covered most part of our codes. We will test our system and revise our codes every certain time.

Every testing cycle, measuring code coverage, writing tests, running tests and revising code will be repeated.

12.3. Integration testing strategy

To test the integration feature of our system, it is essential for us to focus on the assigned php file Because this file takes all responsibilities to enable the communication between the database and the front-end. Once we make sure it works successfully, the integration test is finished.

Prerequisites:	1. MySQL is open 2. Sever is connecting to the internet 3. Local server is on service

Test by steps:	1. Data Collection Input the 'TWITTER_CONSUMER_KEY' , 'TWITTER_CONSUMER_SECRET', 'OAUTH_TOKEN', 'OAUTH_SECRET', which you get from Twitter to use their 36 streaming API, in the 140dev.php file. Change FilePath in get_tweets.php and change the database information which contains host IP Address, username, password and the database name which you want to store the collected data in db_config.php. Then run the get_tweets.php to retrieve data from Twitter, which will be stored as json file in database. Run parse_tweets.php that is a separate background script that takes that value from our json_cache table and parses it into the rest of the DB by removing it into tweets, tweet id, gps coordinates, et cetera. 2. DataFilter Change the database setting in filter.php first. Then run it to filter all history data and store the information of users who are in U.S and their tweets also in two tables. 3. Run our program in browser Make sure all the settings about the database are correct. Run index.html.
State:Success	1.The jsonsender can communicate with the website and the database

	2.Database table with user's information 3.Data can be request.
State:Fail	1.The jsonsender could not communicate with the website and the database 2.Database table with user's information 3.Data could not be request

Table 12-8

13. Project Management and Plan of Work

Conference date and location:
Our team holds conferences twice a week on Tuesday and Thursday at the study room 1 in the Library of Science and Medicine.

13.1 Merging the Contributions from Individual Team Members

This report was compiled by all the team members based on their own works.
There were of course several issues encountered when combing our works. For instance, since to the large amount of data we retrieved from Twitter, it is hard to transfer it and implementing the analysis on all of our own computers. Also, the computer took the responsibilities to retrieved tweets unexpectedly went down. It seriously impacted our combining. But eventually we disassembled that computer and successfully took out the important data.
Other troublesome issue is that several operating files could not be run successfully in both Mac OS and Windows platforms due to some limits of authority problem and different rules of multiple browsers. Finally we tackled this problem by changing the method that used to enable the communication between the device/website and the server.

13.2 Plan of Work

13.2.1 Project Management

Before the end of 10/1/15, our group built up our own website. Here is the URL: https://sites.google.com/a/scarletmail.rutgers.edu/health-monitoring-analytics/. In order to do the further work, we all participate in the database setup. In the conference, team members discuss the ideas about the algorithm in data analysis.

13.3 Work Schedule

Discussion date and location:
Our team holds conferences once a week on Monday or Wednesday at the study room 2 in the Library of Science and Medicine.

13.3.1 Work in progress

TASK	START DATE	DURATION /day	END DATE
Extract Data from Twitter	9/22	7	9/29

Setup Database	9/22	7	9/29
Data Analysis	9/29	10	10/9
Display User Interface	9/24	10	10/5

Table 13-3 Work in progress

13.3.2 Work Schedule after report 2 part 1

TASK	START DATE	DURATION /day	END DATE
Show Heatmap on Google map	10/8	7	10/25
Show Tweets on Google map	10/8	7	10/25
Data Analysis Optimization	10/15	10	10/27
IOS Application	11/9	30	12/7
Android Application	11/9	30	12/7
Website Test	10/27	7	11/4
User Interface Optimization	10/27	7	11/4
Electronic Project Archive	12/8	7	12/15

Table 13-4 Work schedule for future

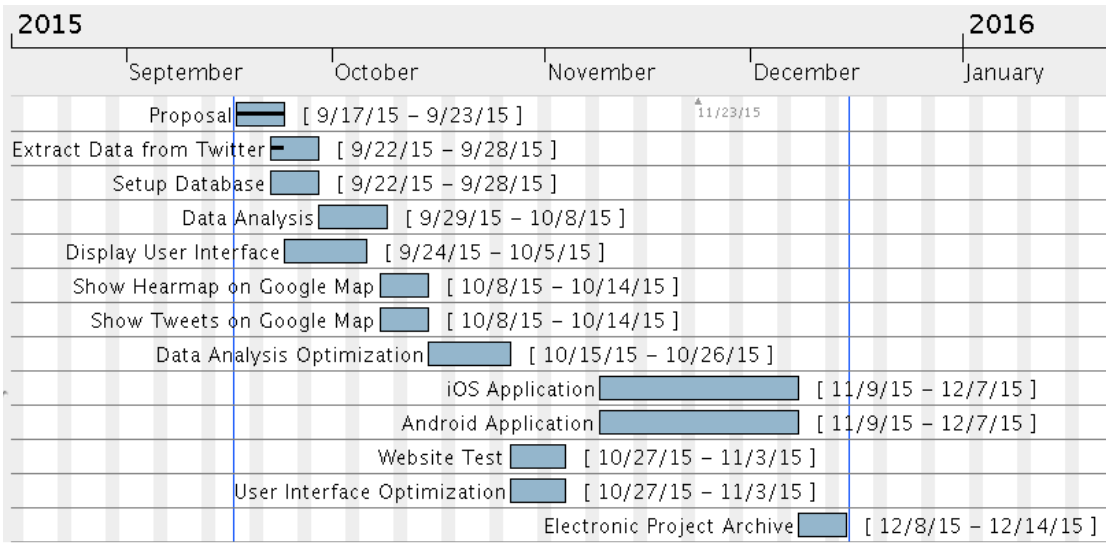


Figure 13-1 Work schedule for whole semester

13.4 Future work:

Possible directions fro future work on this project are discussed below.

1. Use our SVM classification model to analyze more data from user and get more types of results for user interface displaying. For example, we can add different time slot into our classification and offer different classification results based on the time slot chosen by user. In future, user could see the degree of involvement in a sport on a specific time slot.
2. We could offer our information of data analysis to some business or government organizations. In this way, we could start cooperation with them and let our product has more value. For us, we would get more support and resource to develop our product. For them, they can get information or some tendency of some feature and then they can dig business value or political destination to meet more people’s requirements. As a result of this cooperation, we can build a ecology for people’s health monitor, better life and better goals.
3. We would supplement our parts of food analysis to help our health monitor software have more functions. In nowadays, people care more and more about component of food that they eat everyday. We not only want to offer a platform of showing the results of food’s component or nutrition, but also to analyze of all data and then came out with a better advise for someone. This is important because people always need useful advice based on a truth.

14. Project management

a. Project Management

1. Product Ownership

Assignments	R. Zhang	J. Feng	W. Zhang	Z. Zhang	K. Kang	T. Wang
Data Extracting	√		√		√	
Data Analyzing			√		√	
Website Map & Chart Display						√
Website User Interface						√
Android Application				√		
Android User Interface				√		
SVM algorithm			√			

Table 14-1 Product ownership

In the Table 6-1, each team member is assigned with about two section's work. In the next discussion, we will distribute the work of features development to every member in our group. As the result of this kind of distribution, everyone will take part in the data analysis.

2. Breakdown Responsibilities

Assignments	R. Zhang	J. Feng	W. Zhang	Z. Zhang	K. Kang	T. Wang
Extract Data from Twitter	√		√		√	
Setup Database	√	√	√	√	√	√
Data Analysis			√			

Display User Interface			√	√	√	√
Problem Statement			√	√		√
Glossary of Terms	√					
Enumerated Functional Requirements					√	
Enumerated Nonfunctional Requirements					√	
On-Screen Appearance Requirements						√
Project Management		√				√

Table 6-2 Breakdown of responsibilities

Before the end of 10/1/15, our group built up our own website. Here is the URL: https://sites.google.com/a/scarletmail.rutgers.edu/health-monitoring-analytics/. In order to do the further work, we all participate in the database setup. In the conference, team members discuss the ideas about the algorithm in data analysis.

b. Work Schedule

Discussion date and location:

Our team holds conferences once a week on Monday or Wednesday at the study room 2 in the Library of Science and Medicine.

1. Work in progress

TASK	START DATE	DURATION /day	END DATE
Extract Data from Twitter	9/22	7	9/29
Setup Database	9/22	7	9/29
Data Analysis	9/29	10	10/9
Display User Interface	9/24	10	10/5

Table 6-3 Work in progress

2. Work Schedule after report 1 part 1

TASK	START DATE	DURATION /day	END DATE
Show Heatmap on Google map	10/8	7	10/25
Show Tweets on Google map	10/8	7	10/25
Data Analysis Optimization	10/15	10	10/27
IOS Application	11/9	30	12/7
Android Application	11/9	30	12/7
Website Test	10/27	7	11/4
User Interface Optimization	10/27	7	11/4
Electronic Project Archive	12/8	7	12/15

Table 14-4 Work schedule for future

15.Reference

[1]Ivan Marsic. Software Engineering. September 10, 2012.

[2]Nathan Fraenkel, Varun Gupta, Jason Lucibello, Boyang Zhang. Language Disambiguation for Disease Analysis. 2013-2014.

[3]Setup Mysql: https://www.cnblogs.com/onlycxue/p/3291889.html.

[4]Setup PHP & Apache & Mysql: http://www.jb51.net/article/53787.html. http://www.cnblogs.com/good_hans/archive/2010/04/01/1702059.html.

[5]Twitter API: https://dev.twitter.com/overview/documentation.

[6]Rest API: https://github.com/abraham/twitteroauth.

[7]Demographic API: http://textalytics.com/core/userdemographics.