RUTGERS UNIVERSITY

GO_RUN

Software Engineering Project Report

12/12/2013

Angjie Li
Anqi Yan
Prakhar Srivastava
Sang Lin
Siyuan Li
Yadvaindera Sood

Abstract: The Project is primarily focused on development of an integrated web and android software that would allow users to keep real time track of their exercise. The users would be provided information on how to start from scratch to attain different levels of accomplishment depending on personal goals, be motivated to attain them through tools like alerts and community based sharing, monitor their own progress reports and keep a track of their goals. The system also incorporates reward based motivation through a ranking strategy.

CONTENTS

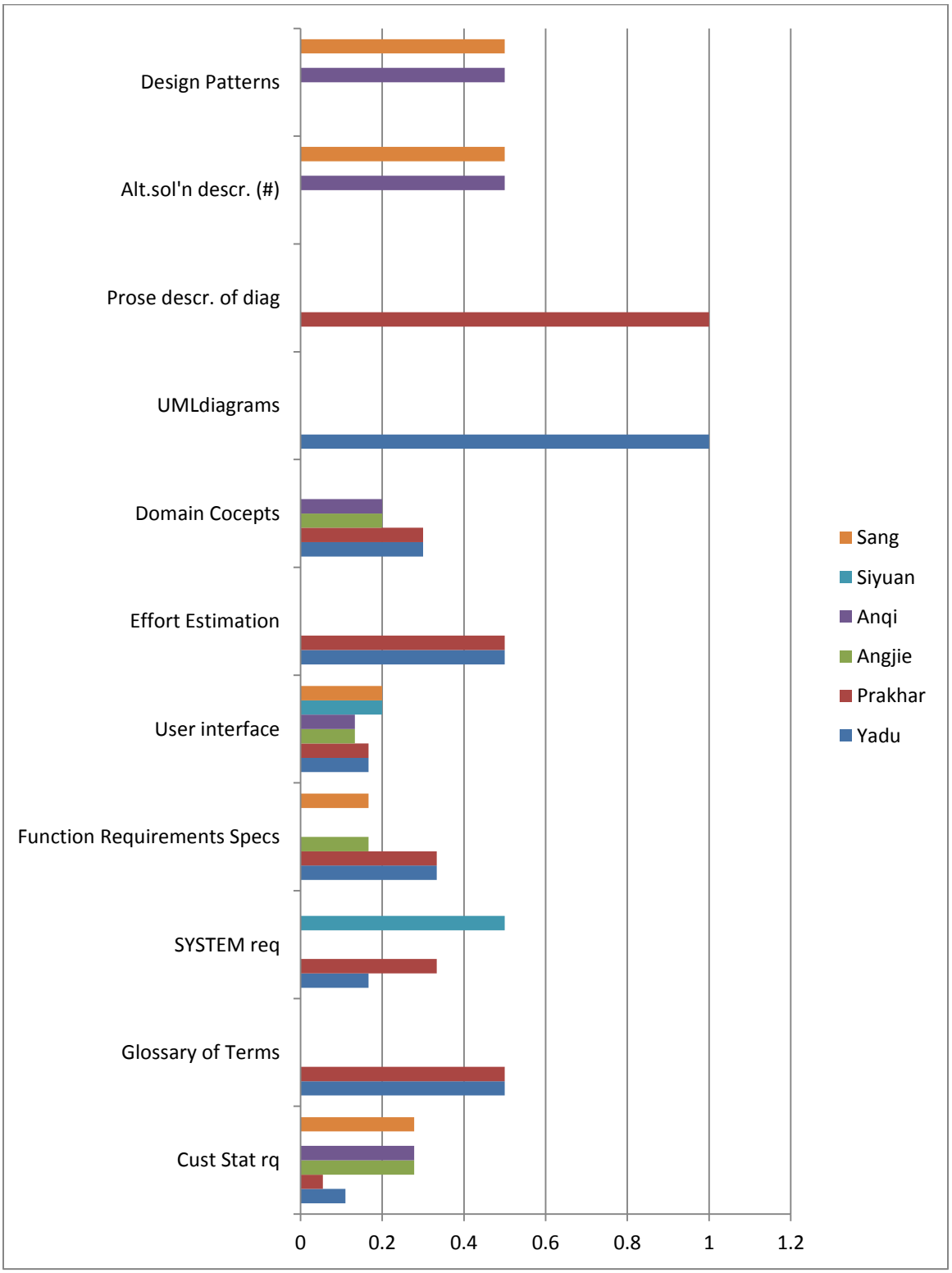
Section	*Title*	*Page*
Section 1	**Customer Statement Of Requirements**	8
1.1	Problem Statement	9
1.2	Go Run System	10
1.3	Technologies Currently In Use	13
1.4	Why Only Jogging/Running	14
1.5	Project Overview	15
1.6	**Glossary Of Terms**	**17**
Section 2	**System Requirements**	**20**
2.1	System Requirements	20
2.1.1	*Android System Requirements*	*20*
2.1.2	*Website System Requirements*	*21*
2.2	Non-Functional Requirements	23
2.3	Appearance Requirements	25
2.4	Acceptance Test	26
2.5	Business Policies	32
2.6	The Not List: What This Project Is Not About	33
2.7	User Interface Diagrams	33
2.7.1	*Android Application Interface*	*33*
2.7.2	*Website Diagram:*	*37*
Section 3	**Functional Requirement Specification**	**40**
3.1	Architectural Design Of The System	40
3.2	Stakeholders	41
3.3	Actors and Goals	42
3.4	Use Cases	43
3.4.1	*Casual Discussion of Use Cases*	*44*
3.5	Use Case Diagram	47
3.6	Traceability matrix	48
3.7	Fully Dressed Use Case Description	52
3.7.1	*Use Case : Select Regimen (UC3)*	*52*
3.7.2	*Use Case: Sign-Up (UC1)*	*54*
3.7.3	*Use Case: View Progress (UC 12)*	*55*
3.7.4	*Use Case: Navigate To Information Pages (UC14)*	*56*
3.7.5	*Use Case: Start Workout/End Workout (UC4/UC5)*	*57*
3.8	System Sequence Diagrams	60
3.8.1	*Use Case: Select Regimen (UC3)*	*60*
3.8.2	*Use Case: Sign-Up()*	*61*
3.8.3	*Use Case: View Progress (UC 12)*	*62*
3.8.4	*Use Case: Navigate To Information Pages (UC 14)*	*63*
3.8.5	*Use Case : Start/End Workout*	*64*
Section 4	**User Interface Specifications**	**65**

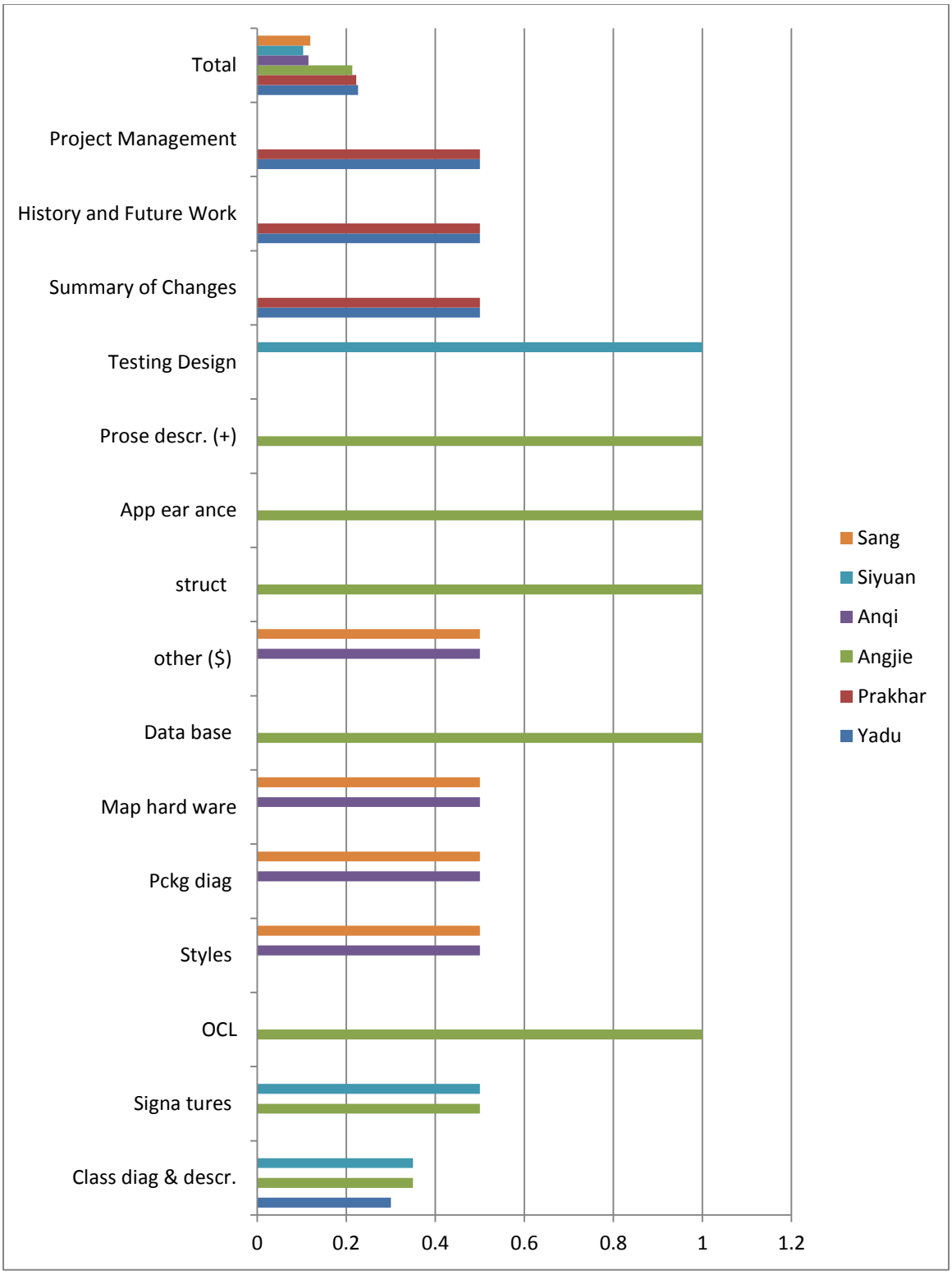
4.1	User Interface Specifications	65
4.1.1	*Web Interface*	*65*
4.1.2	*Phone Interface*	*70*
4.2	**Effort Estimation**	72
Section 5	**Domain Analysis**	**74**
5.1	Concept Definition	74
5.2	Association Definitions	77
5.3	Attribute Definition	80
5.4	Traceability Matrix	82
5.5	Mathematical Model	83
5.5.1	*Accelerometer data analysis*	*83*
5.5.2	*GPS data Analysis*	*86*
5.5.3	*Calories Burned*	*87*
5.5.4	*Alert Model*	*89*
Section 6	**Interaction Diagrams**	**93**
6.1.1	Interaction Diagrams	93
6.1.2	Design Patterns : Theoretical Discussion	96
6.2	Alternate Scenarios	98
6.2.1	*Third Party Device*	*98*
6.2.2	*Only GPS No Accelerometer*	*101*
6.2.3	*Not Use Database For Storing Data*	*101*
6.2.4	*Conclusion*	*103*
Section 7	**Class Diagram And Interface Specification**	**104**
7.1	Class Diagrams – Android Platform	104
7.2	Data Types And Operation Signatures	105
7.2.1	*Start-Page*	*105*
7.2.2	*Mylocationlistener*	*106*
7.2.3	*Logopage*	*106*
7.2.4	*Regimenpage*	*107*
7.2.5	*Selectionpage*	*108*
7.2.6	*Loginpage*	*109*
7.2.7	*Profilepage*	*109*
7.2.8	*Endpage*	*110*
7.3	Classes In The Website	111
7.4	Traceability Matrix	114
7.5	OCL Contract Specificaton	116
Section 8	**System Architecture And System Design**	**118**
8.1	Architectural Style	118
8.2	Identifying Subsystems	118
8.3	Mapping Subsystems To Hardware	119
8.4	Persistent Data Storage	119
8.5	Network Protocol	121
8.6	Global Control Flow	121

8.7	Hardware Requirements	122
Section 9	**Algorithms And Data Structures**	**122**
9.1	Algorithms	122
9.1.1	*Distance And Speed Calculation*	*122*
9.1.2	*BMI*	*123*
9.1.3	*Ranking*	*123*
9.1.4	*Calorie Meter*	*123*
9.2	Data Structures	124
Section 10	**User Interface Design And Implementation**	**125**
10.1	User Interface Design	125
10.2	Interaction Of User Interface	126
10.2.1	*User Interface Of Mobile Application*	*126*
10.2.2	*User Interface Of Website*	*129*
10.3	Detailed User Interface Implementation	136
Section 11	**Design Of Tests**	**139**
11.1	Testing Design	139
11.2	Unit Testing	139
11.3	Integration Testing	142
Section 12	History of Work and Future Plans	143
Section 13	References	145

Breakdown of Contribution

				Yadu	Prakhar	Angjie	Anqi	Siyuan	Sang
Cust Stat rq			6	0.66	0.33	1.67	1.67		1.67
Glossary of Terms			4	2	2				
SYSTEM req			6	1	2			3	
Function Requirements Specs			30	10	10	5			5
User interface			15	2.5	2.5	2	2	3	3
Effort Estimation			4	2	2				
Domain Cocepts			25	7.5	7.5	5	5		
Interaction Diagrams	UMLdiagrams		10	10					
	Prose descr. of diag		10		10				
	Alt.sol'n descr. (#)		10				5		5
	Design Patterns		10				5		5
Classes + Specs	Class diag & descr.		5	1.5		1.75		1.75	
	Signa tures		5			2.5		2.5	
	OCL		10			10			
Sys Arch & Design	Styles		5				2.5		2.5
	Pckg diag		2				1		1
	Map hard ware		2				1		1
	Data base		3			3			
	other ($)		3				1.5		1.5
Alg's & Data	struct		4			4			
User Interface	App ear ance		6			6			
	Prose descr. (+)		5			5			
Testing Design			12					12	
Summary of Changes			5	2.5	2.5				
History and Future Work			5	2.5	2.5				
Project Management			13	6.5	6.5				
			215	48.66	47.83	45.92	24.67	22.25	25.67


Design Patterns
Alt.sol'n descr. (#)
Prose descr. of diag
UMLdiagrams
Domain Cocepts
Effort Estimation
User interface
Function Requirements Specs
SYSTEM req
Glossary of Terms
Cust Stat rq
0
0.2
0.4
0.6
0.8
1
1.2
Sang
Siyuan
Anqi
Angjie
Prakhar
Yadu


Total
Project Management
History and Future Work
Summary of Changes
Testing Design
Prose descr. (+)
App ear ance
struct
other ($)
Data base
Map hard ware
Pckg diag
Styles
OCL
Signa tures
Class diag & descr.
0
0.2
0.4
0.6
0.8
1
1.2
Sang
Siyuan
Anqi
Angjie
Prakhar
Yadu

Summary of Changes

We have tried to implement most use-cases discussed in the previous versions of this report in our project. However, in order that readers can implement other ideas that can possibly be further improved or developed upon, we have not edited out the un-implemented sections from the report. During reviews we obtained some valuable comments from our peers and have tried to weed out the shortcomings at our best.

- Sections in the report like "Why Jogging/Running" were missing proper citations and have been provided them in the final report.
- Some figures in the report including the Use-Case diagram were not properly readable and they have been improved accordingly.
- There were some places where the readers had found the content to be repetitive. To our understanding we have tried to remove those parts and sentences; however magnified versions of the traceability matrix in the use case section have been retained as we did not want to achieve brevity at the cost of clarity.
- Sections on Design patterns and OCL contract specifications have been included.
- Actual image of the database has been included using the command <describe>
- There is a big change in implementation too. The previous "not list" mentioned no real time update for speed and distance. However, we actually ended up implementing it.

Since we did not receive much negative reviews regarding technical details we did not have to make any major technical changes in the report.

1. CUSTOMER STATEMENT OF REQUIREMENTS

The fast pace of our society is leaving people living unhealthy life sitting in the office all day and eating high calorie food. This leads to health related problems like hypertension, diabetes, and heart disease and so on. For instance, those who spend the whole day sitting in offices rarely get a chance to focus on the health aspect of life in depth. As time goes by, the lack of appropriate exercise may cause aggravated health conditions. A healthier lifestyle is necessary to overcome this situation. The most convenient and also the most affordable cardiovascular training is running.

One of the most convenient ways of improving the fitness is by joining a gym under a trainer but that conflicts with nearly every ones work schedule and proximity to a gym. These days many mobile based applications or web based applications are available which claim to help one improve the fitness level, but in vain. We will talk about it later in the coming sections. These applications provide information of how to improve the fitness level or the running time and stamina but are unable to monitor the actual effort invested by the participator. Most of the applications are independent standalone applications which are either available only on the web or on the mobile platform, no integration between them limits the information that can be made available on either on them and also limits the user base.

After interacting with many proactive fitness oriented people an integration of both the web application and mobile application seems to a high demand, also making the mobile application being able to take data from other sources at real time. The use of real time data helps the user to track their level of calories burnt and also gives a measurable quantity to relate to when it comes to fitness. This quantification helps is setting goals for a lesser motivated person to achieve what he or she wants. The users want a interactive application on the mobile platform that can tell them how much and what needs to done and also a web interface than can help them track their development in the same program.

A major factor what people feel is not being able to find the group that have common goals when it comes to fitness, that hinders their continuing the fitness regimen. Though having people make friends similar interest is a personal aspect of the user that cannot be attributed by us. The growth of social

networking though, hints towards a virtual solution, where people who take a particular regimen can see the others and see their track record and derive motivation from them to continue the same or achieve better results.

After interviewing many friends, fitness enthusiasts and joggers we were able to clearly define the problems that users face or may face, also their suggestions (requirements) which we have tried to elaborate in the next section.

1.1 PROBLEM STATEMENT

To the very simple health scenario there are various complex factors that impede their running plans:

- People have extremely busy time schedules and it is hard for them to find slots to fit in running or other forms of workout.
- Lack of motivation is another significant reason.
- Finding people with similar goals is extremely difficult, and mostly people end-up with inactive or over-active company, if found at all, which does not serve as an effective motivation booster.
- Not being able to track your progress is a huge roadblock in many cases.
- No proper schedule to be followed, sometimes you have the time but don't know how much and how to go about it.
- What's more, after a formidable exercise, people would get lazy or boring without any rewards or recognition.
- Expensive tracking and other health monitoring gadgets leave the interested people with either a big credit or without any equipment.
- Complex user interfaces and systems.
- No real data on the systems, only web fed data.
- Very little or no data analysis, as to how much calories burnt, how much time to run improve the stamina, etc.
- No simple helpful mobile application

Motivation and correct information, are the two biggest factor. It is necessary to make any activity entertaining and rewarding to make sure that people would continue to strive for achieving targets that they set for themselves. It is necessary to incorporate these two factors in running to allow people help them have a positive attitude towards it so that they start and keep up with their goals. It is also important to convince them that they have the ability to attain the goals that they have set for themselves.

To achieve this, it is important for people to have information, based on statistical and scientific data, on how to run effectively and safely along with the right diet strategy. On the other hand to pursue targets acts as an external motivation, pushing them to attain success. It can be provided in the form of rewards, a vision of loss in case they don't follow the routine and comparison.

A mobile based data acquisition which acquires not only information if the user is running or not but also the geographical information, creating a database which records all the user information. This will help the system relate to the users local information and provide a virtual group of people who are interested in the same type of regimen. To address the above mentioned problems we propose a simple mobile-web based system that makes a qualified attempt to resolve the problems.

1.2 GO RUN SYSTEM

We propose a more intuitive, simple and powerful system which deals with the above mentioned problems. Our system deals with using the power of smart phones along with the Web to help the user maintain, track and be informed regarding his regimen and development.

The system involves the usage of smart phone and web application which work in unison to provide the user with the desired service. The mobile platform provides us with accelerometer data which is used in real time to calculate the users speed and other vital factors, along with that the platform provides us with location data which is used by the web application to provide user with other users information pertaining to his regimen to bolster the motivational forces behind.

Our system may sound complex at first but a deeper look makes it look simpler to understand. The various interactions that are/will be required by the users have been analyzed and a clear set of requirements have been achieved of how the system must perform, the requirements can be read in the later section of the report.

Here we split the system in sub systems which interact among each other or with the user. We also provide the aim of each of the many components our

system is split into. The interaction between the various components have also been touched upon in brief as we will elaborate over the interaction in detail in the later sections of the repot.
Our system can be seen to have 6 components that work in unison to produce the results which help the user the best results.

- Mobile Interface
- Web Interface
- Database
- Location Data acquisition
- Accelerometer data collection and analysis
- Profile Monitoring

Mobile Interface:

The mobile interface is the smart phone application which provides you with multiple options and is synchronized with the application data and database. The application has been made to help the user in working on his workout session without the hassles of logging in and feeding data. The application provides you with various options as given below.

- Login Page/Sign up page – The first page of the application is the login page where the already registered user can login in, if the user is a new user then he/she can sign up. In case of a new user the user crucial data such as age, weight, height, etc.
- Profile page – This page shows the user his/her goals and targets. This page also talks about the various regimen and dietary plans that are available in the system.
- Data acquisition – The application collects GPS and Accelerometer data which is used to monitor the location and path, whereas the accelerometer data is used to calculate the speed.
- Progress page – This page shows the consolidated data of the user and provides the link to other options that the user can utilize for improvement in his/her regime.
- Statistics – the application also shows the speed at which the user is running and the distance he/she has covered and how much is left to cover. This information is fetched from the Accelerometer data collection and analysis component of the software.

Web Interface:

The web interface is the place the user can see his/her profile also can edit the user information. The website is the elaborate information page where the user can login/signup or can also be a visitor and see all the information provided to the users.

- The user can sign up if not registered and if registered the user can login
- The user can see the profile where the user can see his/her information regarding the regime along with ranking and the goals and the target that the user had set for him.
- The website also in expanse talks about the various dietary plans that should be adopted by the user for attaining the best results.
- The user also has various other options to see his previous data and analysis of his workout sessions.
- The user can also find link to download the application.
- The user can also find other health related attributes which are related to professional athletes.

Database:

This is the other essential aspect of the system which the user does not come into interaction but the web and mobile interface adds data to the database.

- The user email id and password are the two main identifiers to the database, which are fed into the system when the user makes his profile.
- The other data that is associated with the user is his information regarding his goals and targets.
- Other information that is held is the average speed, time taken, calories burnt, average heart rate, regimen followed, etc.

All information about the user is maintained confidentially.

Location Data Acquisition

Location Data Acquisition is the use of GPS on the smart phone for various uses

- Tracking the path used by the user for the run.
- Finding the location as to where the user resides and give location based information.

- It helps in finding people within your vicinity who are also interested in jogging or running.
- Localized information is used to find people near the user to help him join a group with common goals to provide motivation.

Accelerometer data collection and analysis

Accelerometer data is used to get the users movement information from the smart phone. The data is collected and analyzed on the phone to give the user vital information like his speed, time of run, distance left to cover, etc.
This data also helps in maintaining record of the user activities.

Profile Monitoring

This is the most important aspect of the system where we maintain the user information and help in building his profile. The user maintains a single profile to which he/she logs in on the mobile application or the web page and sees all the past data of their run. This data is analyzed and presented in various formats which are easily understandable to the user and provides them with a feedback of their workout.

1.3 TECHNOLOGIES CURRENTLY IN USE

For now, customers always believe that the more money they spend, the better equipment they would get which could help them to be much healthier.

Nike+, one of the most popular technical equipment, is well developed and widely used. Nike+ is worth of Nike's reputation, accuracy and easy portability are its most unique characteristics.

Lerunner, an application which is available on both Android and IOS platform, dominates the market of jogging monitoring in China. The most amazing feature of Lerunner is the distance and speed could be calculated in real time which could help our customers know whether they should speed up or speed down.

Go-run would take more consideration of motivate our users along with focusing on data analysis, experience as well as happiness of success could be shared with everyone. Showing other users in the neighbor will help customers to make more friends and gain increasingly motivation during the jogging.

We have collected data on the maximum used applications which perform the same action as the proposed Go-Run system. Following is the tabulated information on each of the major applications/technologies.

Please ref to figure 1.0 followed to see the various the features each systems bring to the front.

Application/Device	Main Features
Nike+	Working with Apple's product; need additional receiver and specified running shoes from Nike.
Lerunner	Comparing your effort with your friends; almost no cost as long as you have an iPhone or Android phone.
Runtastic	Could be used on iOS, Android, Windows Phone and Blackberry; sharing your success and experience with other users.
Run keeper	Working on both Android and iOS, data could be analyzed for bicycling, jogging, climbing.
Go-run	Real time tracking and data analyses when you jogging, making friends with individuals who exercise nearby sharing your happiness with your friends.

Fig 1.0: comparative study of various applications

As we can see our system encompasses focuses on jogging/running as it the only exercise that can be quantifiably measured without using extra hardware which will add to the cost of the device. In the next section we list down explicit advantages of jogging/running is main focus.

1.4 WHY ONLY JOGGING/RUNNING**

GROWS CARTILAGE

Ignore the naysayers—running isn't necessarily bad for your knees. Research from Australia's Monash University suggests that the impact of running can increase cartilage production, which can safeguard your joints from arthritis.

SHARPENS HEARING

Research from Bellarmine University found that very fit women were six percent more likely to have better hearing than less-fit women. Exercise improves circulation to the ear, which provides a greater supply of nutrients to help preserve hearing.

SAVES YOUR SKIN

Rutgers researchers found that mice who drank caffeinated water and then ran had fewer skin-cancer tumors than rodents who either just got caffeine or just ran. The caffeine-exercise combo caused fewer damaged cells to develop.

BEATS MIGRAINES

Put down the painkillers. A study conducted at the University of Gothenburg in Sweden found that migraine sufferers experiences fewer head pounders when they worked out for 40 minutes three times a week over three months.

REGENERATES MUSCLE

Muscles mass declines over time—or does it? University of Illinois researchers found that exercise triggers a type of stem cell (mesenchymal stem cells) to spur other cells to generate new muscle. That process could prevent age-related muscle loss.

EASES ANXIETY

Feeling panicky about an upcoming work presentation? Go for a run. Researchers from Southern Methodist University near Dallas found that people had significantly milder reactions to stress if they engaged in regular intense aerobic exercise.

PREVENTS CANCER

Finnish researchers studied 2,560 middle-aged men over 17 years and found that the most active men were the least likely to die from cancer, especially in the gastrointestinal tract or lungs. The more intensive the exercise, the prevention could be better.

INCREASES BRAINPOWER

To see how exercise stacks up against other mental stimulants, University of Illinois researchers exposed mice to three types of brain boosters—savory foods,

new toys, and exercise wheels. The wheel was the only tool that improved cognitive function.

STRENGTHENS BONES

Weight-bearing exercise increases bone density, which guards against fractures and osteoporosis, according to researchers from the University of Missouri. High-impact exercise, like running, appears to offer the greatest protective benefit.

STRENGTHENS HEAR

Runing makes the heart stronger. It increases the capacity of the bloo d circulation and of the respiratory system. This is essential for maint aining good fitness. It also speeds up the digestive system and can hel p to relieve digestive problems.

REDUCES CALORIES

runing makes you burn fat and thereby helps to lose weight. In addition to increa sing metabolism, runing is an effective way to burn more calories, which helps yo u lose weight. If calories consumed in food are less than calories spent during exe rcise and other daily activities, you will lose weight.

****Quoted from** http://m.runnersworld.com/health/nine-surprising-ways-running-helps-your-body?page=single

1.5 PROJECT OVERVIEW

In order to meet the customers' requirement, both the website and the mobile application will be provided for Go-Run Project. . The main goal of our project is to remove the motivational roadblocks that prevent any individual from going out there and putting in some hard work to achieve their fitness goals.

As for Users, they may require alert system to remind them keeping exercising in case of there is too much works that they may forget their regimens. Jogging

with accompany will be more joyful, users would like to meet people in their area who share the same interest. Additionally, users are interested in what they've achieved. Looking into their friends progresses is also needed for those users who want to make competition which would definitely motivate them. When they reach their goals, users may want to share their experience and happiness with others to help more individuals to reach their own target.

Extensive information charts available on the website regarding the fitness regimen to be followed. Whenever a user registers, he is asked which category of fitness regime he wishes to register for. Possible exercise schedules, running speeds, running track info would be provided based on the kind of goal that a user establishes. First, the alert system let users be alerted when they did not finish their today's workout. Second, by comparing the new record with their forgone data, users could find out the entire progress. Thus, the record keeper could motivate people to stick to the fitness plan. Comparison, which will strongly motivate our users, will be made if users agree to share their data with others.

The fundamental requirement for any user who wants to join in Go-Run is that his/her Android phone must collect coordinates data. Location data will be analyzed to calculate distance, speed and calories consumption. Besides, if users' location information is shared over permissions, Go-Run would also help you to know your neighbors who enjoy jogging as you do. It involves the ranking system. With good accompany, jogging will be much more interesting and enjoyable. Different users can also group together in the real world and pursue common goals together instead of company that is either too fit to follow or too slow to be interested in.

1.6 GLOSSARY OF TERMS:

Accelerometer	Component in a user's mobile device that records acceleration data
Accelerometer Data Collection and Analysis	Analysis of acceleration data from the user's mobile based accelerometer to generate distance and speed related information
Account	Data entity in the database that corresponds to a real world user and contains specific information including performance, progress, weight, body measurements etc. pertaining to that user
Alert Audio	It is an accompanying audio information that is played to the user

	during his workout session providing details about what is the next step in the schedule for that workout
BMI	Body Mass index, it is calculated using the formula $BMI = \frac{mass(in\ kg)}{(height\ (in\ m))^2}$
Daily Target	Based on the goal that a user is pursuing a schedule is generated for him/her to follow. The requirement for a given day to be completed is called his daily target
Database	Data about the user's attributes maintained as a structured set
Default Measurements	Average statistical values of weight and body measurements that the system database is initialized with in case the user does not intend to provide this information at the time of sign-up or check-in
Forum	A comment based platform/ discussion room where users can post views, questions etc.
Goal	The final state that the user wants to attain. Examples pertaining to the software include attaining a particular body weight, being able to complete a 5K or 10 K run, prepare for a Marathon etc.
Go-Run	Software solution with a combined web and mobile based platform to monitor and analyze running data for users, provide diet and food related information when following running regimes, allow them to view their progress and motivate them to pursue running through different perks like improvement in ranks, good company to run etc.
GPS	In the context of the project this refers to the phone's location tracking component that provides latitude and longitude based information about the user
Location Data Acquisition	The process of acquiring location data through the GPS in the user's phone
Logged-in User	A user who has provided his pre-registered username and password for accessing the member's only features of the website
Logout	End active session on the web application
Mobile Application	The phone based application that is a part of the system and acts as one of the interfaces between the user and the system
Password	An alphanumeric combination that allows the system to confirm the user's identity
Performance	Information based on the user's workout including distance run, average speed at which the distance is covered and calories burned etc.
Profile Monitoring	Analyzing the user's data and calculating performance and progress related information for the user
Ranking system	A reward based motivation booster. By comparing data from users near your location, it would be a tool to boost your confidence showing how well have you been able to pursue your goal compared to others.
Registered	A human user who has created an account with the system and

User	corresponds to a unique
Timer	The countdown timer that is displayed on the screen once the user starts his workout and helps the user keep track of his workout
User Name	A unique alphanumeric combination that identifies the user in the database
Visitor	Any user of the mobile or web application who has not been recorded in the database
Web Application	The web based interface provided to the end-user to interact with the system
Workout	For our software workout refers to running as a cardiovascular workout

2. SYSTEM REQUIREMENTS

2.1 SYSTEM REQUIREMENTS

Based on customer requirements, which are a more vague version of what the customer needs from the system, a more specific set of requirements for the system are generated. These requirements precisely state the capabilities of the system that the customer needs developed. Our system can be broadly segregated into the android platform used to acquire user data when he/she goes for a run, and a web platform that can be exquisitely used to monitor progress. Based on this the system requirements can be easily categorized into two parts namely *android system requirements* (<REQA>) and *website system requirements* (<REQW>). The list of system requirements has been populated accordingly and allows a clear understanding of what the customer wants developed on both platforms giving a clearer picture for the system and how it would eventually appear. The requirements have been given a priority weightage on a scale of 1~5, 5 being the highest priority and 1 being the lowest. These requirements were gathered and prioritized based on the interviews of regular runners, people who wish to start and people who are just beginning their running routines.

2.1.1 Android System Requirements

		Priority
REQA1a	The system should allow new users to register for the service	5
REQA1b	The system should acquire personal information from user at sign-up including name, age, weight, height etc.	4
REQA1c	System should allow user to skip information personal details	2
REQA1d	The System will use default data if no data is given by the user	2
REQA1e	The system should recognize the user and allow him access to his account	5
REQA1f	The system should display user profile on sign in	3
REQA2a	The system should allow user to select the regimen under which he wants to log his workout data on a particular day	5
REQA2b	The system should display dietary information page for the user	2
REQA3a	The system should ask user to provide details regarding the day of the schedule	4

REQA3b	The system should display details of the goals for that day	4
REQA4	The system should allow user to proceed with the run when he is ready	5
REQA5a	The system should alert user (audio) about the ongoing run and when he has to switch from running to jogging, jogging to walking in all combinations according to the day's target	1
REQA5b	The system should display a timer for user to keep track	4
REQA6a	The system should display GPS location on the map when the user is running	1
REQA6b	The system should show active runners in the area	1
REQA6c	The system should display the running speed of the user	4
REQA7a	The system should allow user to exit when he is done	5
REQA7b	The system should tell user his performance when he finishes the run	4
REQA8	The system should allow user to navigate to home page of the app whenever needed(Appearance Requirement)	5
REQA9	The system should provide link to user to the website to access his account there for further information(Appearance Requirement)	5
REQA10	The system should allow the user to give a feedback to the app	3
REQA11a	The system should give the user multiple workout choices to choose from	5
REQA11b	The system should have brief overview of the workouts before choosing	3
REQA12	The system shall provide an exit button to allow the user to exit the application (Appearance Requirement)	5

2.1.2 Website System Requirements

Identifier	**Requirement**	**Priority**
REQW1	The system should display website home page on accessing site address	2
REQW2	The system should provide navigational links to all other pages on the home page	4
REQW3	The system should contain the following descriptive pages	
REQW3a	About : The page should describe the overall concept of the system and information about the creators	3
REQW3b	Regimens : The page should provide details about all the available running regimens that can be followed using the	4

	system and describe the methodology to follow them	
REQW3c	Food: The page should contain details and links about dietary recommendations for users based on different regimens they are following	3
REQW3d	App : The page should contain information about the associated android app and how it can be used	3
REQW3e	Community: The page should allow the user to browse through other existing users using different filters like location, weight reduction goal, gender, age, zip code.	5
REQW3f	Blog : The link should contain related blog topics regarding running and weight reduction (mostly informational)	2
REQW4a	The system should allow user to create an account by signing up	5
REQW4b	The system should ask for information from user at the time of account creation including username, password, email id, age, weight, height, waist size, neck size etc.	3
REQW4c	The system should allow user to skip information other than username, email id and password at the time of account creation	3
REQW5a	The system should allow existing users to log in	5
REQW5b	The system should divert users to personal home page after login	5
REQW6	The system should provide registered users to view different regimens available and select one which they wish to follow	5
REQW7a	The system should allow users to get an overview of their progress on their homepage	5
REQW7b	The system should allow users to view condensed information including daily *rank*, alerts, daily goal, regimen registered for and Body Mass Index	3
REQW8	The system should provide user to view a calendar of scheduled goals for the regimen	3
REQW9	The system should alert user in case he misses a jog with ramifications of missing that specific day and possibility to make up for it	4
REQW10	The system should allow user to update his current physical statistics to keep a track of his improvement. These may include weight, height, body measurements	3
REQW11	The system should allow user to maintain and update a set of specific goals including target weight, target time period etc.	5
REQW12	The system should allow user to view a progress report which should include but not be limited to	

REQW12a	Weight: Allow user to view his progress over a period of time indicating variation in weight	4
REQW12b	Body Measurements : Allow user to monitor changes in body measurements based on data he/she has provided over time	3
REQW12c	Goal Completion : Allow user to monitor his closeness to the set goal	5
REQW13	The system should allow user to search other users from pool of existing users based on specific filters he she chooses which may include age, gender, area, target weight	4
REQW14a	The system should rate users based on their daily progress and percentage completion of daily goal. This list should allow user to view his personal ranking in his regimen category	3
REQW14b	The system should generate a daily ranking list based on user ratings	3
REQW15	The system should provide a forum for discussion where users can share thoughts and problems	2
REQW16	The system should allow feedback from users regarding various issues	1
REQW17	The system should allow administrator to delete user accounts	3
REQW18	The system should allow the administrator to oversee the forum and delete unacceptable posts.	3
REQW19	The system shall allow user to logout at any time during the session (Appearance Requirement)	5

2.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirement is used to specify the criteria of system operation,which should be contrast to functional requirement that define specific behavior and function.

In order to these clarify software quality attributes, we will use the FURPS model. FURPS is acronym for functionality, usability, reliability, performance and supportability, which is founded at Hewlett-Packard and elaborated by Grady and Caswell. The previous discussion mainly covered the functional parts of this acronym, the discussion to follow will cover the URPS part.

Usability:

Usability is used to determine the ease of using and learning for human-made object. First, we use JAVA as our developing platform which is accessible and widely used in Android systems. Secondly, to make our software and website easy to use, user interface design would play a very important role. User design should make information more accessible for user, such as, a help option can be found at once when the customer encounters some problems, a traceable history for users to review their progress of previous period in their account. Also, to

inform users their condition while they are running, real-time updated information like running speed, route and so on would be provided.

Reliability:

Reliability is defined by the system's availability and how long it will take to recover from a failure. It is very important to our program, since our system is based on a huge amount of data. If failure occurs, it always accrues loss of data. In order to maintain high reliability, we should try our best to keep a low appearance of error in our system, and also, if there's a failure encountered, the system should have the ability to figure it out, send it to the programmer and then reset giving a reasonable explanation to the users.

Performance:

Performance is closely related to how fast our system can perform. As user is running, he wants know his speed. And running speed is real-time data, it will be useless even one second delay. So it requires the system with a high performance of computer's calculation for the runner's speed.

Supportability:

Supportability includes a variety of elements, such as maintainability, sustainability, testability and so on. As requirements above, we should make our program easier to use, not only for users, but also for people who want to modify it or add new functionality to it. In order to make sure that code can be fixed and reused easily, we should separate the whole system into several independent functions. As for several independent parts, people can find the main problem more quickly and rewrite the one with a problem without changing others, which will save a lot of time and manual work. Based on the above abstract information a more discrete set of requirements can be tabulated as follows. The requirements follow the same identifier pattern as <REQ><Category><Number>. (REQN<num>).

Identifier	Requirement	Priority
REQN1a	A help option shall be built, to help user get access to more detailed explanation and help user get a quick start.	4
REQN1b	Training data shall be accessible to user, which help the user to acquire good knowledge of his progress.	5
REQN2	System shall avoid user's illegal operation, recover from failure quickly and send the failure report to the programmer and a reasonable explanation to the user	4
REQN3	System shall track runner's location and calculate all useful data in time, in order to keep the information valuable.	4
REQN4a	System shall be cut into several small independent part, in order	5

	to make modifications easy	
REQN4b	Every functional part should have interface for the programmer to test it, in order to find out whether there is a problem.	5
REQN5	The system shall display coherent information on the web and mobile platform	4

2.3 APPEARANCE REQUIREMENTS

End users have specific requirements from the front-end of any software. These requirements are mostly subsets of specific system requirements, however some specific appearance based requirements must be separately catered in order to make improve user experience. Our software is mostly visual in nature and intends to implement information based usage via viewing details of personal progress, relative progress and alerts through the web and android platform. This implies that the system is somewhat appearance intensive. This attribute demands details of appearance oriented requirements, however since most of these requirements are listed under major system requirements, for the sake of brevity they would be discussed as segments of existing discussion or collections of requirements already provided.

Unregistered users should be able to:

- See information about the system including guidelines and developer information.
- View general information available for diet, exercise regimens and blog site.
- Only view (not post on) existing forum topics
- View the community of existing users
- Sign-up for an account (REQ1a)

Registered users should be able to

- Check in their current body measurements
- View and modify their goals
- View progress reports – daily and overall
- View BMI based on last check in information (add in requirements)
- View current rank based on last day’s total progress
- View Current day’s progress
- View alerts and possibility to improve on schedule missed
- View regimen registered for
- Logout at any stage in the session

There are however, some requirements that specifically fall under the category of appearance requirements and have to be separately tabulated to emphasize their importance. These requirements fall under the category of the mobile

application and web application system respectively and must be identified accordingly. They have thus been given identifiers in extension.

Identifier	Requirement	Priority
REQA8	The system should allow user to navigate to home page of the app whenever needed	5
REQA9	The system should provide link to user to the website to access his account there for further information(Appearance Requirement)	5
REQA12	The system shall provide an exit button to allow the user to exit the application	5
REQW19	The system shall allow user to logout at any time during the session	5

2.4 ACCEPTANCE TEST

Acceptance Tests for REQA1

ACT1.1	Ensure any user to be able to open an account with a specific username, password and e-mail address through mobile interface (pass);
ACT1.2	Ensure any user to skip information personal details to get a quick look at the whole system(pass);
ACT1.3	Ensure any registered user to log in to his/her account by using correct username, password and e-mail address(pass);
ACT1.4	Ensure users that do not have correct username, password and email address not to log in to system(fail);
ACT1.5	Ensure any registered user to change his/her personal information (e.g. height, weight etc) (pass);

Acceptance Test Cases for REQA2

ACT2.1	Ensure any user to view all the regimens, including 5K run, 10K run, Marathon, Quick run, Power run(pass).
ACT2.2	Ensure any user to choose a specific regimen, including 5K run, 10K run, Marathon, Quick run, Power run(pass).
ACT2.3	Ensure any user to change his/her regimen as he/she wants(pass)

ACT2.4	User once change his/her regimen, he/she cannot get his/her data for former regimen(fail).
ACT2.5	Ensure any user to get access to dietary information(pass).

Acceptance Test Cases for REQA3

ACT3.1	**A registered user is able to get a specific goal about how much calories he/she should burn today or the weight he/she should lose today(pass)**
ACT3.2	A registered user is able to provide schedule to the system(pass);ACT3.3, A registered user is not able to get detailed schedule about when should the user do his/her exercise(fail)

Acceptance Test Cases for REQA4

ACT4.1	**A registered user is able to proceed with the run when he is ready(pass)**
ACT4.2	A registered user is not able to get a pause during jogging(fail)

Acceptance Test Cases for REQA5

ACT5.1	A registered user will be alerted if he/she should go jogging today, and when to change his/her speed(pass)
ACT5.2	A registered user is able to know the time he/she spend for jogging(pass);
ACT5.3	Alert system do not have the ability to force user go jogging(fail);

Acceptance Test Cases for REQA6

ACT6.1	A registered user is able to get location data while he/she is jogging(pass);
ACT6.2	A registered user would have access to know other active users in the same area(pass);
ACT6.3	The speed of user will be available(pass);
ACT6.4	A registered user is not going to view all the users in his/her area(fail);

Acceptance Test Cases for REQA7

ACT7.1	A registered user is able to exit when he/she finishes jogging(pass);
ACT7.2	Performance of user's jogging will be available when he/she finishes(pass);

ACT7.3	Performance comparison between today and previous would not be shown to registered user(fail);

Acceptance Test Cases for REQA8

ACT8.1	A registered user is able to get to homepage whenever he/she wants(pass);
ACT8.2	Some pages do not have the ability to go back to homepage because of lack of homepage-button(fail);

Acceptance Test Cases for REQA9

ACT9.1	A registered user is able to get directed to the website where he/she can get further information from his/her account(pass);
ACT9.2	A registered user cannot access to our website by clicking the hyperlink(fail);

Acceptance Test Cases for REQA10

ACT10.1	A registered user is able to give feedback through app(pass);
ACT10.2	Feedback is send to another database instead of ours(fail);

Acceptance Test Cases for REQA11

ACT11.1	A registered user should have access to various regimens(pass);
ACT11.2	A registered user is able to get access to brief overview of regimens before choosing(pass);
ACT11.3	Some registered users could only access to their previous regimen instead of all of them(fail);

Acceptance Test Cases for REQA12

ACT12.1	A registered user is able to get the same information from website and app(pass);
ACT12.2	Information from app would be delayed comparing to website(fail);

Acceptance Test Cases for REQW1

ACT13.1	Users will view website homepage when they are accessing site address(pass);
ACT13.2	Some users cannot access to website homepage due to their internet service provider (fail);

Acceptance Test Cases for REQW2

ACT14.1	User is able to go to any page by clicking hyperlink on the homepage(pass);
ACT14.2	Some pages cannot be accessed because there are some typing mistakes about hyperlink(fail);

Acceptance Test Cases for REQW3

ACT15.1	User is able to view all the following pages: About, Regimens, Food, Community and Blog(pass);
ACT15.2	Some of those pages could not be accessed to by clicking hyperlink(fail);

Acceptance Test Cases for REQW4

ACT16.1	Ensure any user to be able to open an account with a specific username, password and e-mail address through website interface (pass);
ACT16.2	Ensure any registered user to provide information at the time of account creation including username, password, email, age, weight, height, waist size, neck size etc. (Pass)
ACT16.3	Ensure any user to skip information personal details to get a quick look at the whole system(pass);
ACT16.4	Account cannot be opened if the username you want to use is already exist(fail);

Acceptance Test Cases for REQW5

ACT17.1	Ensure any registered user to log in to his/her account by using correct username, password and e-mail address(pass);

ACT17.2	Ensure users that do not have correct username, password and email address not to log in to system(fail);

Acceptance Test Cases for REQW6

ACT18.1	Ensure any user to view all the regimens including 5K run, 10K run, Marathon, Quick run and Power run(pass).
ACT19.2	Ensure any user to choose a specific regimen including 5K run, 10K run, Marathon, Quick run and Power run(pass).
ACT19.3	User once change his/her regimen he/she cannot get his/her data for previous regimen(fail).

Acceptance Test Cases for REQW7

ACT20.1	Ensure any user to view their progress on the homepage(pass);
ACT20.2	Ensure any user to have access to condensed information including daily rank, alerts, daily goal, regimen selected for and Body Mass index(pass);
ACT20.3	Some users cannot get to the latest information about their progress because it was not updated daily(fail);

Acceptance Test Cases for REQW8

ACT21.1	Ensure any registered user to have access to his/her unique calendar which contains specific goals for the regimen he/she chose(pass);

Acceptance Test Cases for REQW9

ACT22.1	A registered user will be alerted if he/she missed a jog that he/she should take according to the calendar(pass);
ACT22.2	Alert system cannot force user go jogging but a notification(fail);

Acceptance Test Cases for REQW10

ACT23.1	Ensure any registered user to update his/her personal information (e.g. height, weight etc.) (pass)
ACT23.2	Some information cannot be changed due to their characteristics such like gender, height(fail);

Acceptance Test Cases for REQW11

ACT24.1	Ensure any registered user to update his targets including target weight, target calories consumption etc. (pass);

ACT24.2	Target data failed to be changed because of website design errors(fail);

Acceptance Test Cases for REQW12

ACT25.1	Ensure any registered user to access to his/her progress when he/she stick to regimen(pass);
ACT25.2	Data of weight should be accessible to user when he/she wants to have an overview(pass);
ACT25.3	A registered user is able to use specific method to measure his/her body condition(pass);
ACT25.4	The achievement that user have accomplished should be available on website(pass);
ACT25.5	User could not get accurate analysis due to incorrect health monitoring method(fail);

Acceptance Test Cases for REQW13

ACT26.1	**A registered user is able to use filters, gender and regimen, to know other users;**
ACT26.2	Target users may not be selected because of errors of database(fail);

Acceptance Test Cases for REQW14

ACT27.1	Ensure any registered user to have access to ranking system which includes information about users, their daily progress and percentage achievement of daily goal(pass);
ACT27.2	Ensure any registered user to view his/her own ranking in his/her regimen(pass);
ACT27.3	Users cannot view others ranking in any situation(fail);

Acceptance Test Cases for REQW15

ACT28.1	Ensure any registered user to share his/her opinion and experience by posting topic in forum(pass);
ACT28.2	Editing and deleting topics would be available for any registered user(fail);

Acceptance Test Cases for REQW16

ACT29.1	Sending feedback should be available on website for a registered user

	(pass);
ACT29.2	Feedback won't be send to administrators' email due to setting issues (fail).

2.5 BUSINESS POLICIES:

In case any conflict arises during operation of the software it is mandatory that conflict resolution is carried out using preset ground rules. Based on literature review of standard ground rules and an insight into the operation of the software the following business policies have been laid down in order that they can help in cases of conflict.

Each business policy has been crafted after great amount of discussions and thought. We try to focus on the shortcoming to the system which are not yet handled and thus maintain a set of rules which are to be referred when a dispute arises.

Policy Number:	**Policy**
BP 1:	Any dispute over the ranking granted to the users will be resolved by the administrator of the site who encompasses set of underlying rules.
BP 2:	Privacy Policy- No personal data will be made public unless permitted by the user.
BP 3:	User data between the website and the android application will be synchronized at the end of day, thus will be reflected only the next day.
BP 4:	Dietary information is only informational.
BP 5:	All information regarding health have been collected from various websites and Go Run does not reflect their ideology.
BP 6:	In case of failure of the application no data will be stored.
BP 7:	Default average demographic data will be used if a user does not input personal details, in this case the data displayed on the profile will be

	representative values.
BP 8:	The Go Run website comments section will be screened for derogatory remarks, if any and removed by the administrator.

Table 2.4: Business Policies

2.6 THE NOT LIST: WHAT THIS PROJECT IS NOT ABOUT

This project exhaustively covers many aspects of customers' requirements however there are certain hardware and software limitations which restrict its ambit. So it is important to specify what this project is not about.

These can be summed up by the following points:

1. The ranking system is going to display the top 10 users, not all of them. However the user can view his personal ranking.
2. Users cannot reply to others in forum. They can only make new comment.
3. Data of calories you burned will not be calculated exactly according to the actual situation, but statistically. No real time data is being collected for these parameters and they would be calculated based on algorithms.
4. Accuracy of distance is relative. Data collection rates would limit accuracy.
5. There is a limitation to the "push factor", the alerts can only motivate and inform users, it is their responsibility to actually achieve targets.
6. Personal weight and body measurements must be updated by user, the system cannot get these information by our Android app.
7. User data would be available on the community section after permissions, and cannot be shared otherwise. Data on the user's personal device would never be shared directly.

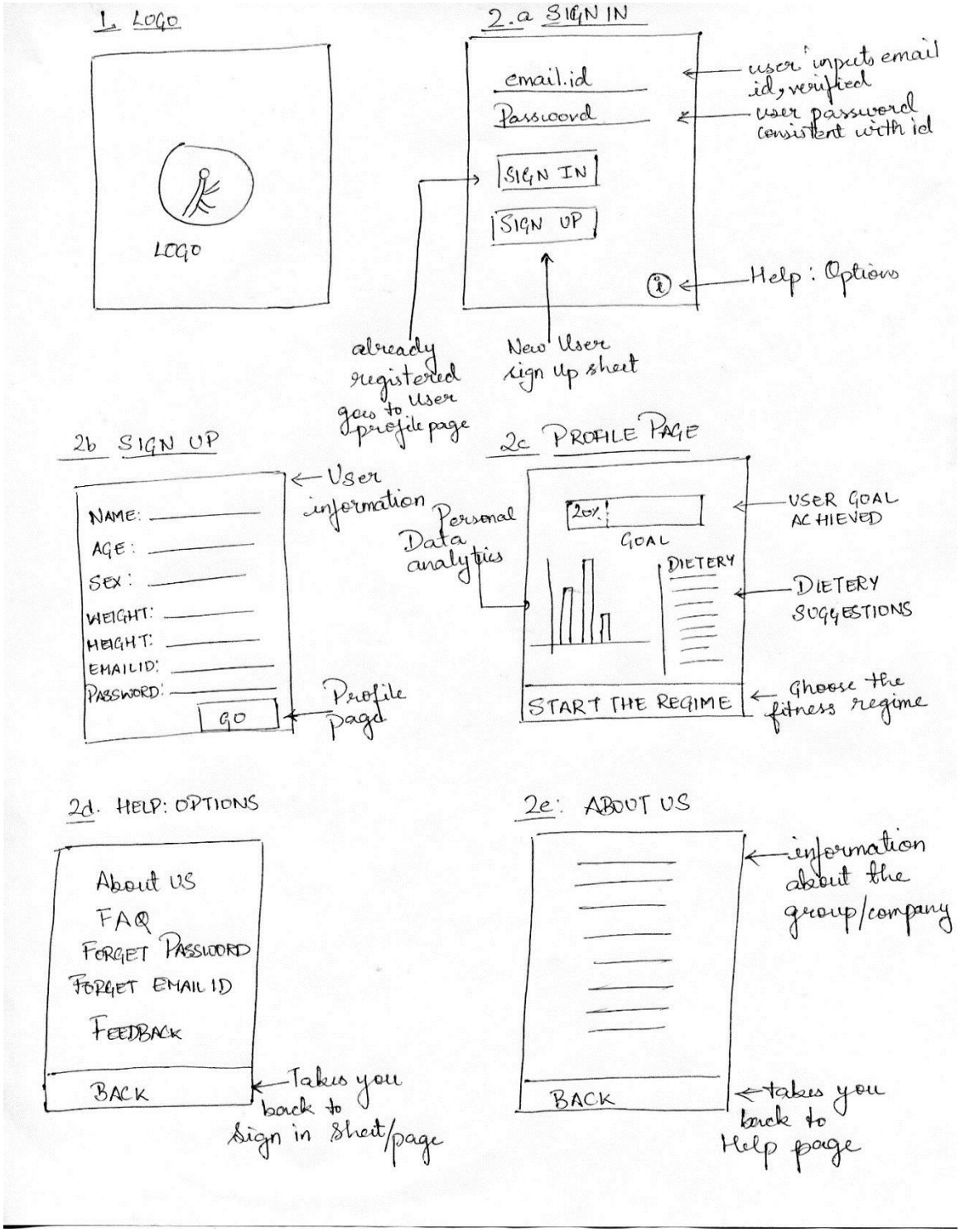
2.7 USER INTERFACE DIAGRAMS

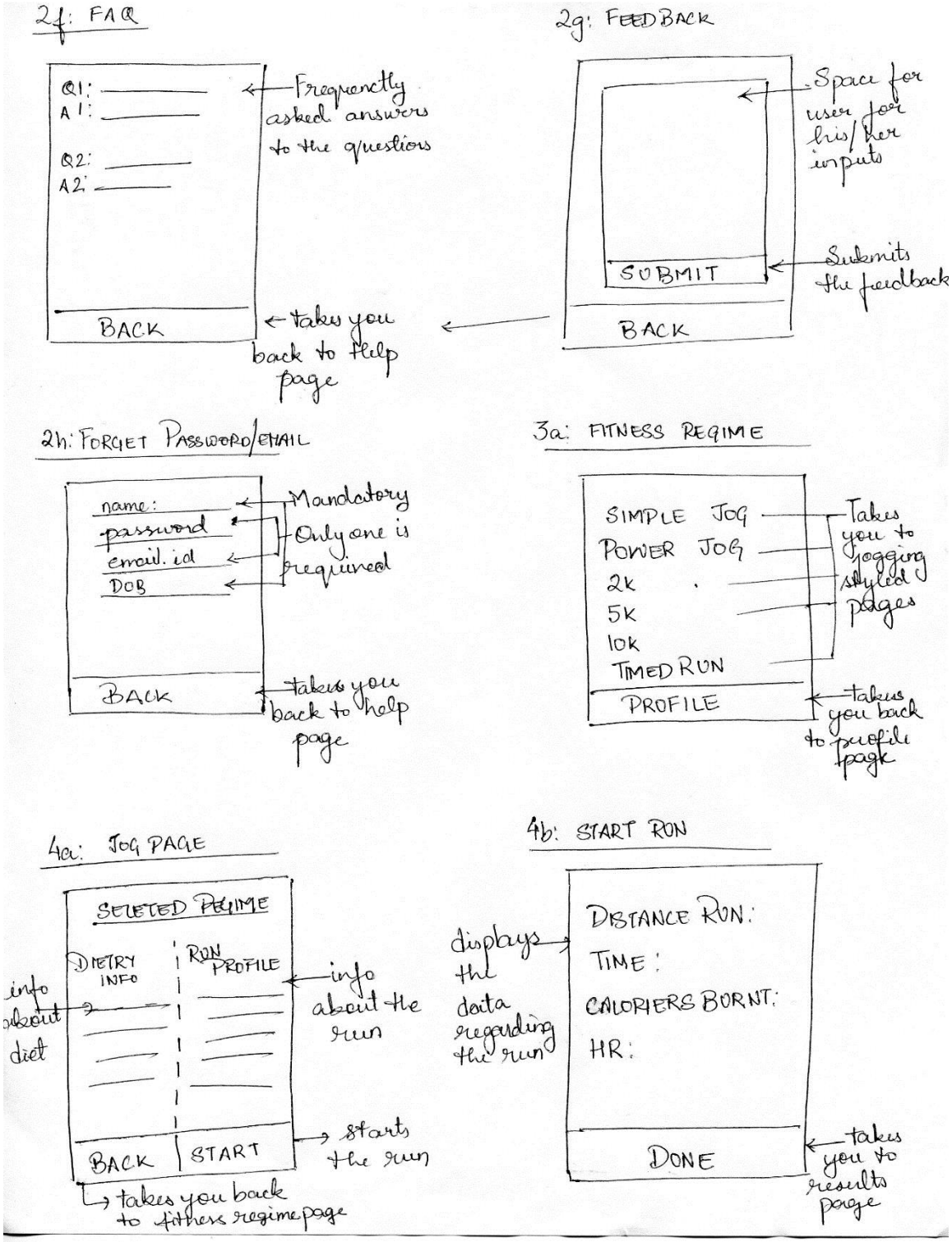
Having discussed about the requirements and other attributes relating to the project we now present the draft of how the system looks. As the requirements are classified in two major sub sets, the mobile platform for which we reside to android platform and the web interface.

Here we present how both the user interfaces will look like taking into consideration all functional requirements.

2.7.1 Android Application Interface:

The section depicts the various pages that a user sees when he installs the application from the app store.


1. LOGO
LOGO
2.a SIGN IN
email.id
Password
SIGN IN
SIGN UP
user inputs email id, verified
user password consistent with id
Help : Options
already registered goes to User profile page
New User sign up sheet
2b SIGN UP
NAME:
AGE:
SEX:
WEIGHT:
HEIGHT:
EMAILID:
PASSWORD:
GO
User information
Profile page
2c PROFILE PAGE
Personal Data analytics
20%
GOAL
USER GOAL ACHIEVED
DIETERY
DIETERY SUGGESTIONS
START THE REGIME
Choose the fitness regime
2d. HELP: OPTIONS
About US
FAQ
FORGET PASSWORD
FORGET EMAIL ID
FEEDBACK
BACK
Takes you back to Sign in Sheet/page
2e: ABOUT US
information about the group/company
BACK
takes you back to Help page


2f: FAQ
Q1:
A1:
Q2:
A2:
Frequently asked answers to the questions
BACK
takes you back to help page
2g: FEEDBACK
Space for user for his/her inputs
SUBMIT
Submits the feedback
BACK
2h: FORGET PASSWORD/EMAIL
name:
password
email. id
DOB
Mandatory
Only one is required
BACK
takes you back to help page
3a: FITNESS REGIME
SIMPLE JOG
POWER JOG
2k
5k
10k
TIMED RUN
Takes you to jogging styled pages
PROFILE
takes you back to profile page
4a: JOG PAGE
SELETED REGIME
DIETRY INFO
RUN PROFILE
info about diet
info about the run
BACK
START
starts the run
takes you back to fitness regime page
4b: START RUN
displays the data regarding the run
DISTANCE RUN:
TIME:
CALORIERS BORNT:
HR:
DONE
takes you to results page

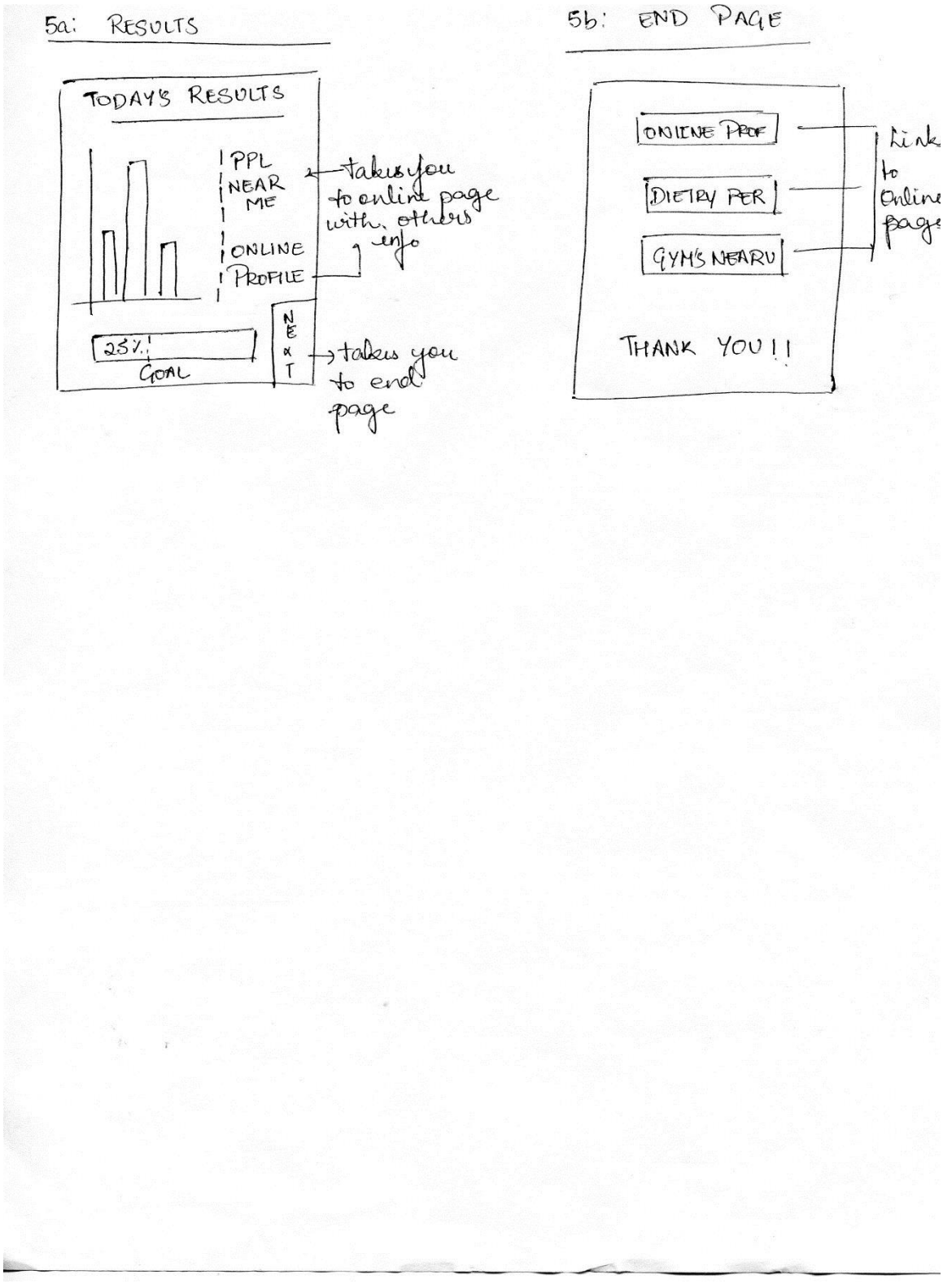
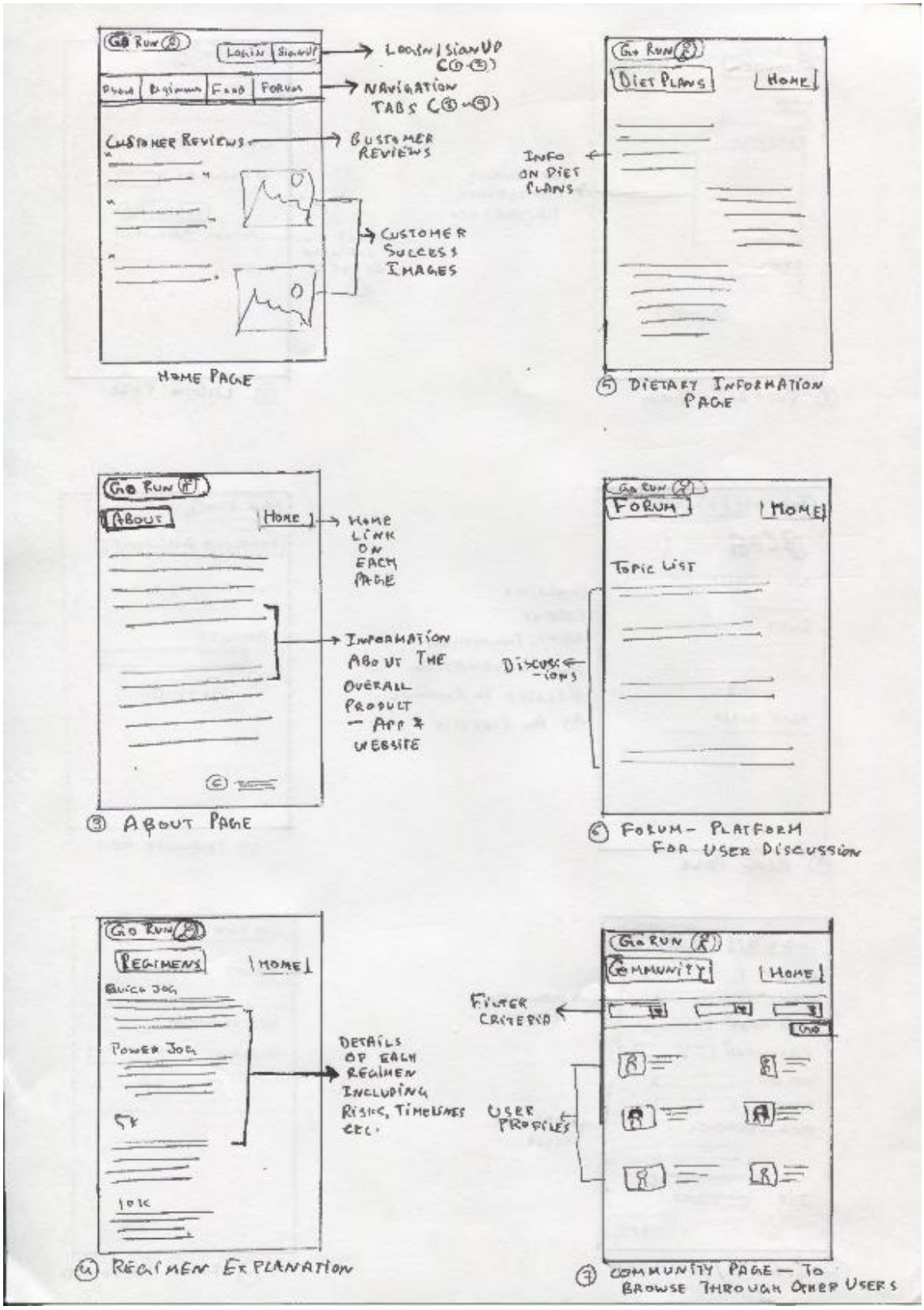


Fig 2.6: Android user interface diagrams

2.7.2 Website Diagram:



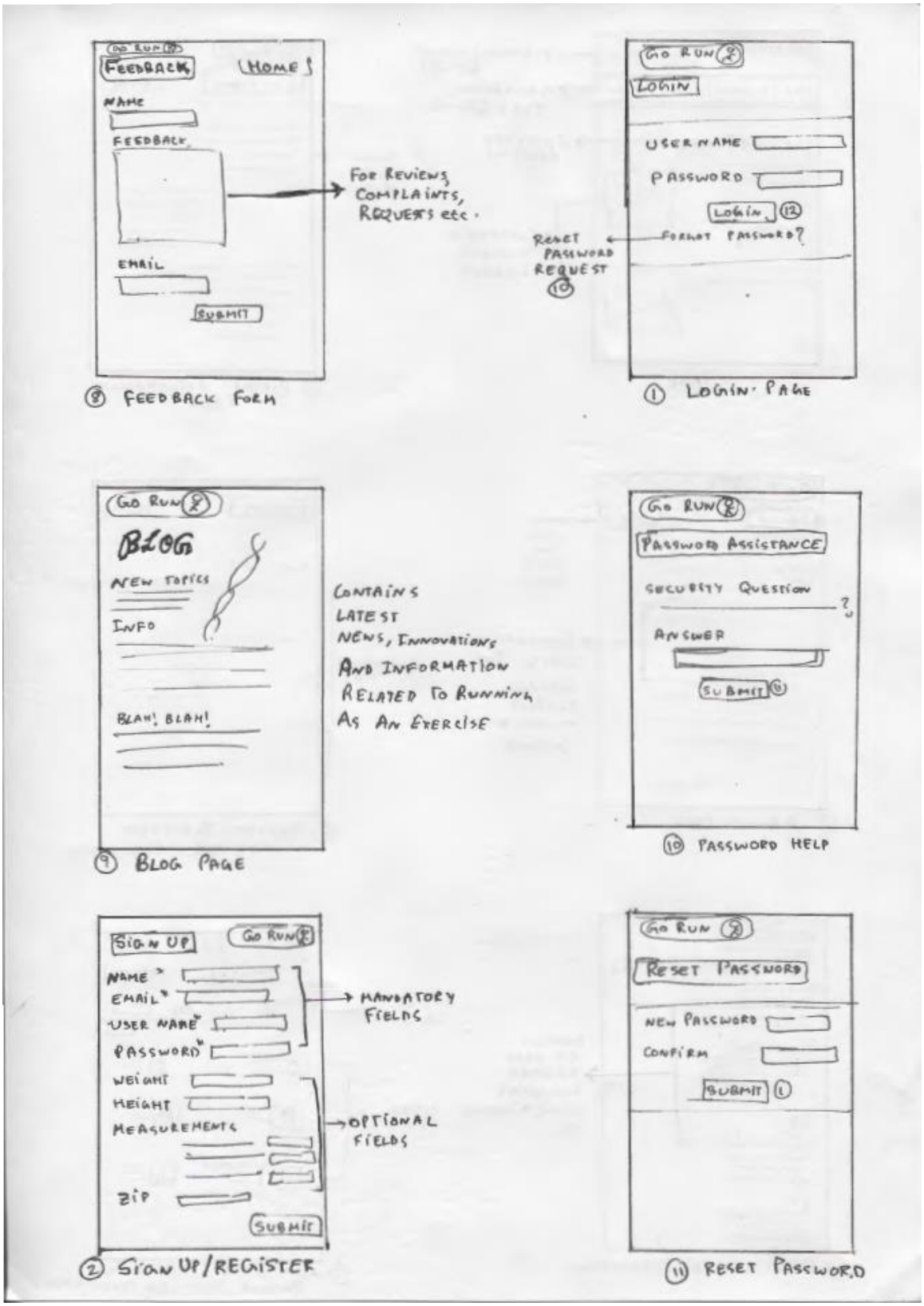


Fig 2.6 : Web Interface Diagrams

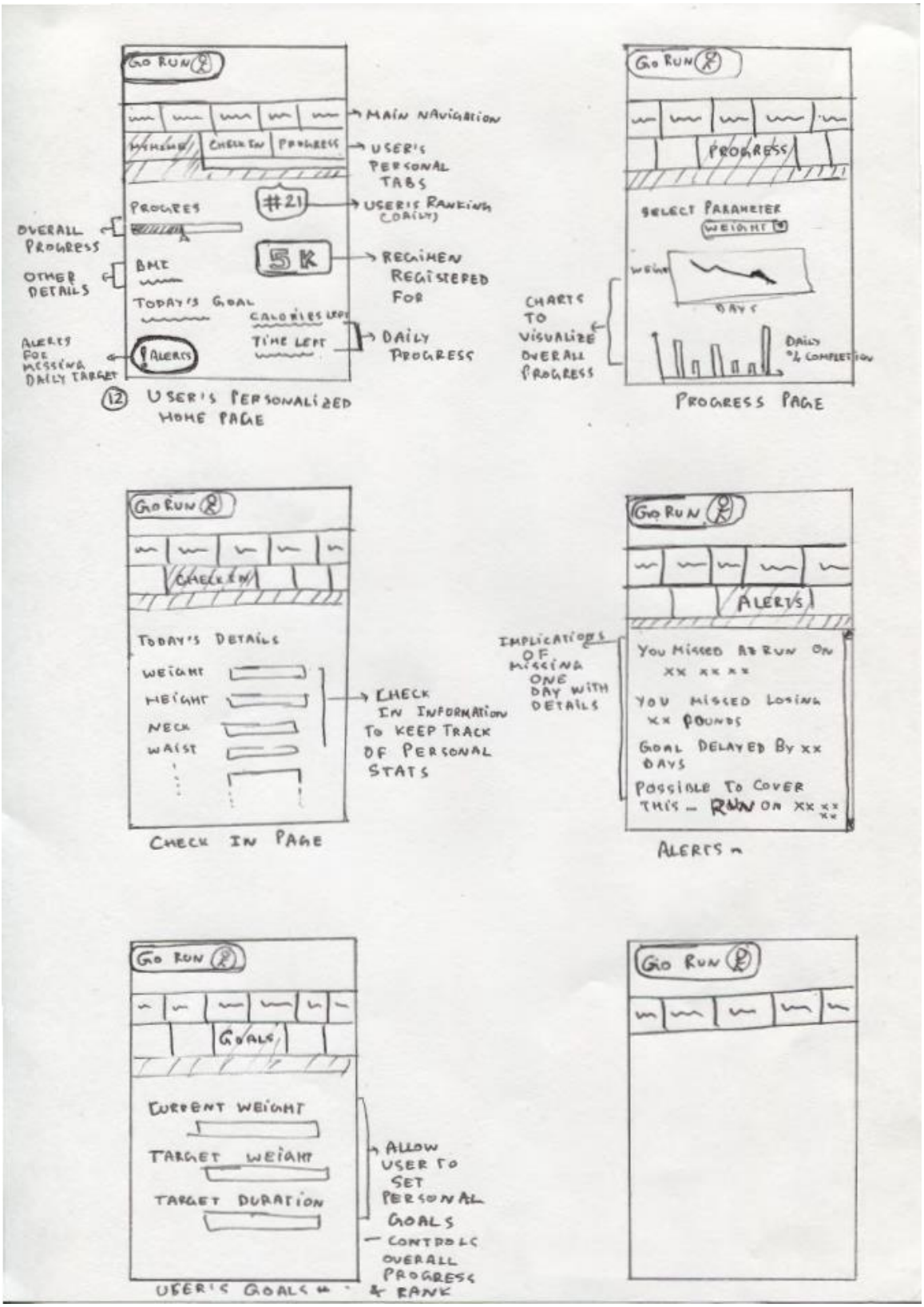


Fig 2.7: Web user interface diagrams

SECTION 3: FUNCTIONAL REQUIREMENT SPECIFICATION

3.1 ARCHITECTURAL DESIGN OF THE SYSTEM:

The Go-run system is the response to the various concerns of the users who are involved in the fitness programs but are unable to achieve the desired results. Keeping the needs of the user in mind the system has been designed to be very simple to use and at the same time being able to match the expectations of our users. We have tried to implement an object oriented design paradigm based on division of responsibilities of the system into individual reusable and self-sufficient objects, each containing the data and the behavior relevant to the object. [1] The objects in turn tend to attain coherence in working together.

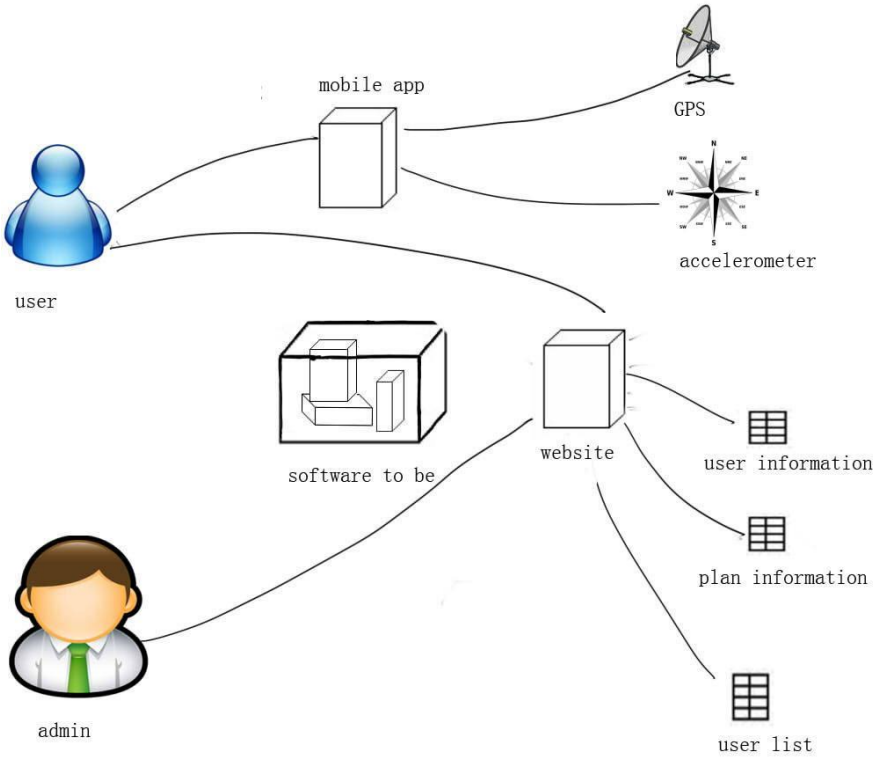


Fig 3.1

Altogether the system has been subdivided into two main sub-systems the *android application* and the *web application*. These act as interfaces between the system and the end-user. Depending on usag scenarios the interfaces respond differently to different actors. For example, for a logged-in user the web interface displays an extra set of user privileges in forums like possibility to post, whereas for a visitor it only allows viewing. Also, to reduce complexity based on hardware capabilities, certain features are restricted to the web application and are not available on the phone application and vice-versa.

There are certain objects in the overall framework that are accessed independently to acquire the information they have and are thus considered in the category of supporting actors. They include *location data acquisition* through the *GPS*, *time* data through the phone's *timer/clock* and speed and acceleration data from the phone's *accelerometer*. Another contributing object is the *database* system that would store and provide all user specific information to the system.

The major capabilities and functions of subsystems can be categorized as follows:

Web Application:

- Viewing overall progress and workout analysis details
- Check-in with current measurements
- Viewing BMI, Rank, current performance charts
- Analytics based on measurements
- Community search feature

Phone Application

- Data logging from regular workouts
- Immediate performance measurement including speed, distance and time
- Count other users in the area around the user

Except for these features most other features are shared by both interfaces

The database forms another independent component of the overall system which is accessed over different activities both directly and indirectly. In most cases it plays the part of an offstage actor, where it is used to maintain logs of user data to be used for analytics at a later stage

3.2 STAKEHOLDERS

Every project has stakeholders. Stakeholders are people who have an interest in the successful completion of the project. There are many different types of stakeholders, and the stakeholders vary by project. But the important thing to remember is that the stakeholders should have some part in defining the project objectives, since they are the people who will be affected by the outcome. On similar grounds the stakeholders in the project *Go-Run* would include the following categories of people:

End Users: The project majorly targets the segment of people who have difficulty in starting or continuing workout regimens for a multitude of reasons and need some sort of a push to achieve their goals or targets and also help in defining these goals. The ambit however, is not restricted to this segment only as fitness enthusiasts can also use this platform to regularly monitor progress and improve their current fitness status by revving up their goals.

Administrator: The applications have to be developed keeping in mind the admins requirements so that his/her task of regulating content and membership of the product is smoothened out.
Clients: The product is competitive and has good market prospects. This can be an attractive investment option for clients who wish to take over the product after it is fully developed. It would then be necessary to tweak and incorporate changes into the existing architecture. It is thus necessary to keep in mind the client's interest to allow modifications at the client end code while simultaneously maintaining the product's privacy and integrity.
Last but not the least, the project is mainly defined by the *developers* who have come up with the idea for it and chisel out finer details at every stage of development. They are also fairly affected by the outcome and are thus listed as a category of stakeholders for this project.

3.3 ACTORS AND GOALS

Actors are broadly classified based on their state during the operation of the system. Based on different tasks that these actors perform, their state in the system changes. An example can be a visitor who after logging in converts into a logged-in User.

Visitor	Any person who visits the website or uses the phone app and navigates through all possible pages without accomplishing the login task. The goals of such users may include acquiring information about the software and different regimens being offered or registering with the website
Registered User	A user who has a unique user-name and password based access to the system but may or may not have accomplished log-in
Logged-in User	A registered user after accomplishing the task of log in. These users are allowed maximum privilege access to the website including access to personal progress charts, forum comment posting, viewing or modifying their current stats or goals etc.
Administrator	The role of the admin is to monitor site usage and ensure that safety rules are followed. He mediates forum content and has the rights to terminate any user from the database.
GPS	As defined in the glossary, the tem GPS is used in a more broader sense in this document, pertaining to a

	device that is used to acquire location data. It is physically present inside the user's phone, however is external to the system. It acts as a participating actor and does not individually initiate any use cases.
Accelerometer	Similar to the characteristics of the GPS this device is used to acquire speed based information from the phone's hardware.
Mobile Application/Interface	The interface provided to the user to interact with the system through the mobile device. It acts as a participating actor in a wide range of use cases allowing transfer of information from the user to the system and vice-versa
Web Application/ Interface	Similar to the mobile application, the web application/interface allows user to interact with the system

A special case of actors that has been included in this documentation are the web and mobile interfaces. It is extremely difficult to classify these to objects and we have decided to treat them as individual actors based on the definition " An actor is any entity (human, physical object, or another system) external to the system-to-be that interacts with the system-to-be. Actors have their responsibilities and seek the system's assistance in managing those responsibilities." The justification given is that as actors, these interfaces have individual tasks for providing and receiving information to and from the user. It is easy to confuse this definition and assume that the web and mobile interface are the system, however, even at this early stage in the project it is clear that the system operates at deeper layer and performs the tasks of analysis rather than information acquisition.

3.4 USE CASES

Use cases are defined as specific tasks that are performed by actors to fulfill single or multiple system requirements/user stories. When discussed casually they sound more like user stories and give an idea about tasks from the customer's perspective explaining what kind of capabilities does the customer want to provide to an end user or other actors. However a fully dressed discussion of use cases is more technical in nature as it involves control transfer discussion, possible alternate cases and details about initiating and participating actors. Use cases are inherently derived from system requirements but a single use case can fulfill multiple requirements.

3.4.1 Casual Discussion of Use Cases:

The main objective that coarsely describes the whole software is monitoring the user's regular workouts (runs) and providing him/her information based on the data acquired. The process of acquiring this information would encompass multiple individual objects working in coherence with actions initiated by different actors. For implementation purposes however this objective has to be broken down into several actions including starting a workout (UC4), recording and displaying data for the user and ending the workout to enable data logging in the system (UC5). Around the main objective a lot of complementary features would be incorporated that would ensure a more wholesome user experience. These include personalized analytics based on the user's body measurements (UC12), community searching (UC18) and interaction (UC20) through the community search tool and the forum.

Identifier	Action	Description
UC1	**Sign-Up**	The first thing that a user needs to do in order to be able to access the website as a registered user is to register himself with a unique identifying user name in the database. This action is called Sign-Up
UC2	**Login**	Even if the user has a registered account in the database the system can only recognise him/her after the user logs in
UC3	**Select Regimen**	On the mobile application it is imporatant for the user to provide information about which regimen he wants to workout with. Data will only be saved for the regimen he has registered for
UC4	**Start Workout**	After selecting the workout regimen and viewing details about the same the user can start his workout. He is directed to a page where he can view a timer, his speed, distance run and directional voice based notification audio would be played to tell him about the next steps
UC5	**Stop/ End Workout**	After the workout schedule ends(timeout) or on the user's discretion he/she can end the workout. This option leads to the performance page where the statistics of the whole workout are displayed and then he is unidirectionally directed towards the mobile applicaton's homepage.
UC6	**Exit Application**	The user can navigate to the homepage at any time and exit the application. This feature provides complete control to the user over the app. Any unsaved data

		however would be deleted.
UC7	**View Profile Page**	A registered user is provided a profile page that display's personal information including BMI, Rank, last day's performance and navigational links to other pages. To viw this profile it is necessary that he logs in. This feature is not available to user's who access the website as visitors
UC8	**View Dietary Information**	The system intends to provide users with correct information regarding diet plans to follow during the fitness regimen that they are following
UC9	**View Schedule**	To achieve goals based on specific regimens users have to follow a particular schedule which would only be available once a specific regimen is selected. The user can view this schedule to plan ahead.
UC10	**View Home**	The user can navigate to mobile application's home page whenver required
UC11	**Provide Feedback**	Over its period of development and even after its launch it is important that all softwares kep on improving through user feedback. Realizing the potential of this activity we have incorporated this as an independent use case for logged in users
UC12	**View Progress**	The main functionality of this project is to allow users to keep a track of the regimen they are following. Both the mobile and web application provide this functionin different ways. The mobile application displays performance oriented progress that helps the user how he is progressing everyday in terms of speed, distance covered and percentage completion. The web applcation is a superset of this function and provides even more analytic information regarding aspects of progress.
UC13	**View People and Current GPS location**	It is an additional feature that the mobile application provides. The user can access information regarding people who are using the app in the zip code that he is running in. It is possible for the user to interact with these people through the forum later.
UC14	**Navigate to information pages**	The visitors to the website are provided product information in the form of regimens provided, diet information provided and community of registered users which he, she can access but cannot modify (forum). Registered users have full access to these segments.
UC15	**View Alerts**	Pertaining to this softwares alerts are treated as notifications on missed schedule days, ramifications of

		missing and corrective measures. This is a unique capability of the software that gives a personalized edge to the software which contemporary software don't provide\
UC16	**Check-In**	The software gives the registered users ability to update his personal weight and measurement related information at any point of time.
UC17	**Update Goals**	The software also provides the registered users ability to update their regimen goals
UC18	**Search Community**	Visitors and registered users can search other registered members based on filters like zip, gender, goals and weight etc.
UC19	**View Ranking**	The system generates rankings based on percentage completion of daily goals which are provided to individual users for motivational purposes
UC20	**Comment on forum**	Registered users can post commnets and discuss other information on the forum
UC21	**Logout**	This is an appearance specific web requirement that mandates that every user should be able to end session at all times while using the interface
UC22	**Delete User**	This is an administrator spoecific privilage that allows the administrator to remove a user in case of any problems
UC23	**Manage Forum**	Forum comments can sometimes lead to problems in a society and must be removed if reported or otherwise thought potentially provocative or hurtful. It is the job of the administrator to be able to remove such instances from the interface. Thus he/she is provided with this option to manage the forum

The use cases can also be categorized on the basis of the initiating actors that include visitor, logged-in user and administrator. Visitors to the web interface would be allowed access to basic features that boost marketability of the product. Registered users get more personalized privileges. There are some other restricted control privileges that are only restricted to the administrator which allow him/her to manage the content and access to the software (UC22, UC23).

3.5 USE CASE DIAGRAM

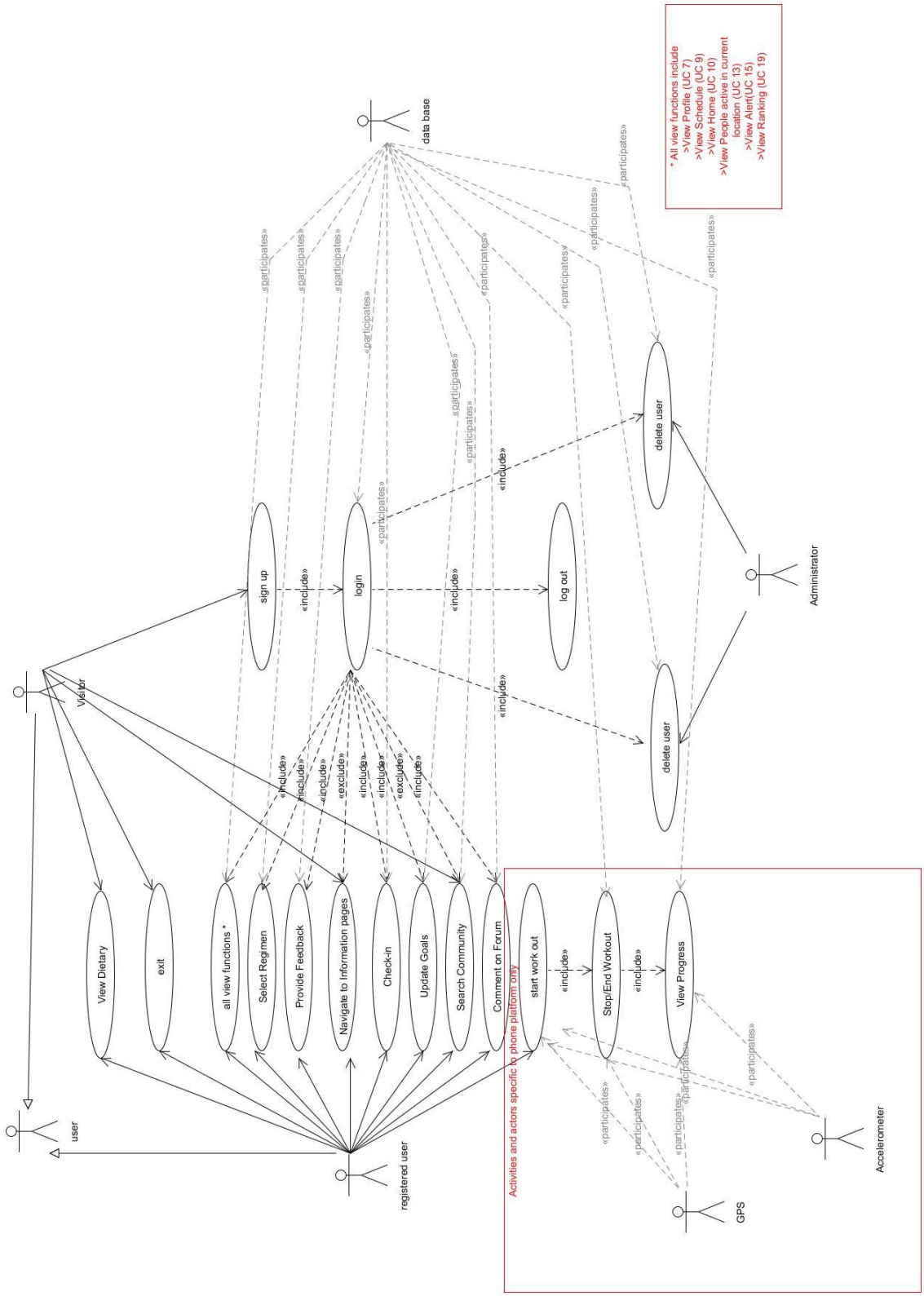


Fig 3.2

The use case diagram provides a decent amount of information about some use cases that are fairly important and also helps visualize what actors play supporting and participating roles in accomplishing different tasks. However it might generate a misleading picture about the "*highest priority*" use cases. From the diagram above it is really easy to deduct one thing, it appears as if login is the most important use case of all. It cannot be disputed that login is mandatory for access to privileges that a registered user has and this is the main reason why it is seen contributing to most other use cases. However, it should be realized here that login is a second tier use case that assists actors in completing first tier tasks. Thus, in order to get a clear picture another metric is needed, which is acquired through the concept of *traceability matrix* and *priority weights.* Each use case is assigned a *maximum priority weight* and a *total priority weight.* These weights depend on the system requirements that the system fulfills. In our project the login use case fulfills four system requirements (REQA1e, REQA1f, REQW5a, and REQW5b). This leads it to acquire a maximum priority weight of 5 and a total priority weight of 18. This weightage does rank it amongst high priority use cases, however still not the highest. High priority weight use cases include more important tasks like selecting regimen that the user wants to pursue (UC 3 TPW = 26), view progress (UC 12, TPW = 21) and view information regarding diet plans, regimens, other community members, doubts and discussions etc (UC 14, TPW =21). These priority weights also corroborate the actual necessity for the project based on the requirements of the end-user and hence validate the weight distribution provided rather than a seemingly skewed distribution towards use cases that are more important for developers.

3.6 TRACEABILITY MATRIX

The concept of priority weights discussed in the previous sections is derived through traceability matrix where the use cases are mapped to the system requirements that they fulfill. This in turn allows determining what the maximum priority weight (highest priority weight for an associated requirement) and the total priority weight (sum of priority weights of all associated requirements) are. The traceability matrix for our system is as follows

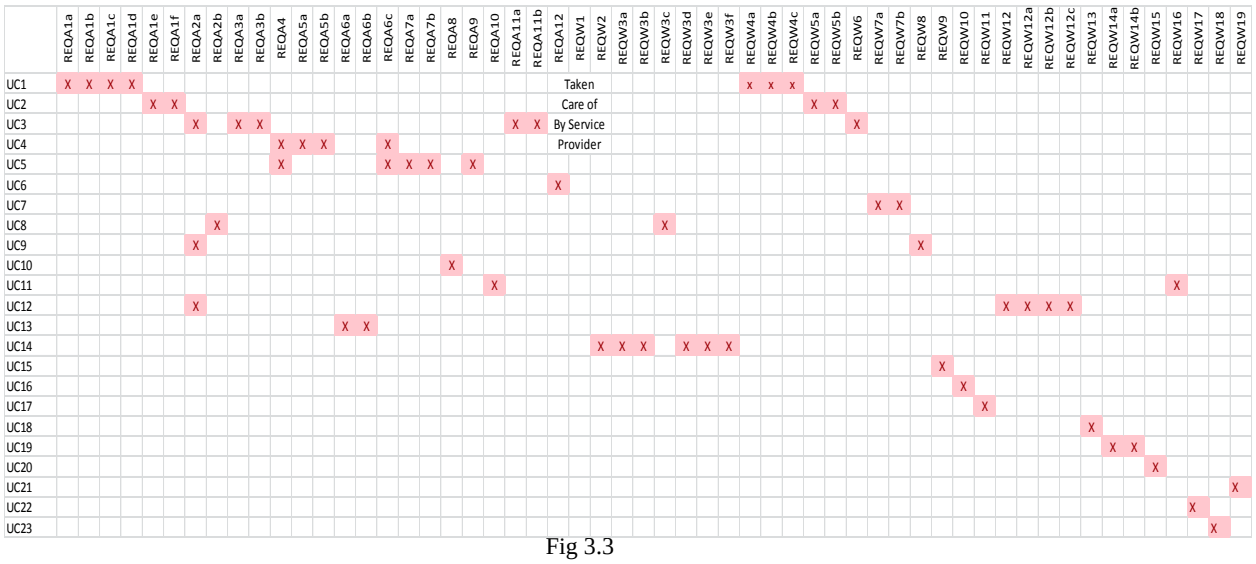


Fig 3.3

	REQA1a	REQA1b	REQA1c	REQA1d	REQA1e	REQA1f	REQA2a	REQA2b	REQA3a	REQA3b	REQA4	REQA5a
UC1	X	X	X	X								
UC2					X	X						
UC3							X		X	X		
UC4											X	X
UC5											X	
UC6												
UC7												
UC8								X				
UC9							X					
UC10												
UC11												
UC12							X					
UC13												
UC14												
UC15												
UC16												
UC17												
UC18												
UC19												
UC20												
UC21												
UC22												
UC23												

Fig 3.4

	REQA5b	REQA6a	REQA6b	REQA6c	REQA7a	REQA7b	REQA8	REQA9	REQA10	REQA11a	REQA11b
UC1											
UC2											
UC3										X	X
UC4	X			X							
UC5				X	X	X		X			
UC6											
UC7											
UC8											
UC9											
UC10							X				
UC11									X		
UC12											
UC13		X	X								
UC14											
UC15											
UC16											
UC17											
UC18											
UC19											
UC20											
UC21											
UC22											
UC23											

Fig 3.5

	REQA12	REQW1	REQW2	REQW3a	REQW3b	REQW3c	REQW3d	REQW3e	REQW3f	REQW4a	REQW4b
UC1		Taken								x	x
UC2		Care of									
UC3		By Service									
UC4		Provider									
UC5											
UC6	X										
UC7											
UC8						X					
UC9											
UC10											
UC11											
UC12											
UC13											
UC14			X	X	X		X	X	X		
UC15											
UC16											
UC17											
UC18											
UC19											
UC20											
UC21											
UC22											
UC23											

Fig 3.6

	REQW4c	REQW5a	REQW5b	REQW6	REQW7a	REQW7b	REQW8	REQW9	REQW10	REQW11	REQW12
UC1	x										
UC2		X	X								
UC3				X							
UC4											
UC5											
UC6											
UC7					X	X					
UC8											
UC9							X				
UC10											
UC11											
UC12											X
UC13											
UC14											
UC15								X			
UC16									X		
UC17										X	
UC18											
UC19											
UC20											
UC21											
UC22											
UC23											

Fig 3.7

`	REQW12	REQW12a	REQW12b	REQW12c	REQW13	REQW14a	REQW14b	REQW15	REQW16	REQW17	REQW18	REQW19
UC1												
UC2												
UC3												
UC4												
UC5												
UC6												
UC7												
UC8												
UC9												
UC10												
UC11									X			
UC12	X	X	X	X								
UC13												
UC14												
UC15												
UC16												
UC17												
UC18					X							
UC19						X	X					
UC20								X				
UC21												X
UC22										X		
UC23											X	

Fig 3.8

The matrix allows us to verify that each system requirement has been catered to and also helps determine use case priority from the table that follows.

	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11	UC12	UC13	UC14	UC15	UC16	UC17	UC18	UC19	UC20	UC21	UC22	UC23
Max Weight	5	5	5	5	5	5	5	3	5	5	3	5	1	5	4	3	5	4	3	2	5	3	3
Total Weight	24	18	26	19	18	5	8	5	8	5	4	21	2	21	4	3	5	4	6	2	5	3	3

Fig 3.9

This weightage clearly brings out the highest priority use cases from the whole set. Based on total priority, and using maximum priority to sort out any conflicts that arise, we can derive the following logical relation between use cases in the project.

UC3 > UC1 > UC12, UC14 > UC4 > UC2,UC5 > UC7,UC9 > UC19 > UC6,UC10, UC17, UC21 > UC8 > UC15, UC18 > UC11 > UC16, UC22, UC23 > UC20 > UC13

It is thus observed that the use cases with higher priorities are the ones that address customer requirements like being able to select the right regimen to follow (UC3), register for the right regimen and use available features of the website (UC1), view progress in the regimen being followed (UC12), acquire information about different aspects of the regimen being followed etc. These are the use cases that actually build the premise on which the whole software is being developed. Other use cases like viewing alerts (UC15) and listening to notifications(UC4) form additional feature that are added to improve usability of the software, improve user experience and allow safety features to be incorporated.

3.7 FULLY DRESSED USE CASE DESCRIPTION

It is apparent from the relation above which use cases must be considered for high priority development. Based on the directions provided, only these high priority use cases will be discussed for the fully dressed use case description. It must be considered that the actions where “system displays” is used implies that system requires this information from the user and displays it via the web or phone interface. This is thus an indirect task accomplished through the interface (an actor here) which is specified via the identifier (W or A or W/A).

3.7.1 Use Case : Select Regimen (UC3)

This use case can be implemented at two stages. The user can select and reselect his regimen through the web application while creating his account or during any of his active sessions. On the phone application the user has to mandatorily select a regimen before initiating a workout. Both instances of accomplishing the task require the user to be registered and logged in, except for one where the user selects regimen during sign-up process. The description for the use case and control flow during it is as follows:

:: include uses cases - login/signup

Control flow/ system sequence diagram

1A	←	System displays available options for regimens
2A	→	User selects the regimen
3A	←	System provides information about the selected regimen and asks user to select day of the regimen that the user
4A	→	User selects day of the regimen to be followed
5A	←	System displays option to start workout via the android interface
1W/A	←	System provides regimen options to the user to select during signup
2 W/A	→	User selects regimen from the list
3 W/A	<- -	System updates database
1W	→	User navigates to check-in page (::include use case for check in)
2W	←	System displays options for different regimens available
3W	→	User selects the regimen he is interested in
4W	<- -	System updates the database with this information

Use Case Name: Select Regimen	**Identifier : UC3**
Requirements Catered	REQA2a, REQA11a, REQA11b, REQW6, REQA3a, REQA3b
Initiating Actor	Logged-in User
Actor's Goal	View Different Regimens & select a regimen that he wants to register for
Participating Actor	Web Application/Android Application(supporting) , Database(offstage)
Pre-Condition	User may or may not have selected a regimen
Post-Condition	Regimen information for Logged in user updated
Scenario	**Expected Result**
1. User selects button to navigate to home page	Application does not update information about the user and returns to the home screen
2. User selects Regimen	(a) Mobile Application : The system proceeds to the diet information page and then to the start run page sequentially (b) Web Application : The system updates the user's information in the database

It must be understood here how the system interface has to act as an intermediate participating actor in most of these use cases. The notation <number>(W/A) indicates which interface has the use case been implemented through. Some use cases are implemented in the same way through both the interfaces hence the identifier (W/A). [A -> Android interface needed, W-> Web interface needed. The (- ->) symbol indicates that the system is interacting with an offstage actor, the database in most cases.

3.7.2 Use Case: Sign-Up (UC1)

Users have to be registered with the system to be able to use the login action and eventually be able to access higher level privileges. This action is accomplished using the use case *Sign-Up*.

1W/A	←	System displays options to login or Sign-up
2W/A	→	User selects the Sign-Up option
3W/A	←	System displays option to enter parameters like name, username, password,
4W/A	→	User enters parameters required
5W/A	<- -	System checks database for existing username
6W/A	- ->	Database returns pass for non-existing user name/ fail for existing account (go to 3A in case of fail)
7W/A	←	System displays list of parameters to be filled by the user like age, weight, regimen to be selected etc
8W/A	→	User enters the requisite parameters/ user selects to skip these parameters
9W/A	←	System directs user to login page after successful creation of account

Use Case Name: Sign up	**Identifier : UC1**
Requirements Catered	REQA1a,A1b,A1c,A1d,W4a,W4b,W4c
Initiating Actor	Visitor
Actor's Goal	To input the required data and become a registered user
Participating Actor	Web Application/Android Application(supporting) , Database(supporting)
Pre-Condition	The actor should not be a registered user
Post-Condition	The actor now has a registered account
Scenario	**Expected Result**
1. Actor inputs a user name that already exists	The user is notified that the username already exists and he has to select a different user name.
2. Actor inputs a unique	(a) Mobile Application: The actor is directed to the

	non existing username	login page. (b) Web Application: The actor is directed to the login page. (c) Database: the database is updated with new user information.
3.	**Actor decides to skip weight and measurement information**	The actor is directed to the register account page.

3.7.3 Use Case: View Progress (UC12)

One of the major features that have been provided to the end-user is view progress. Progress has been defined as the overall improvement that the user has been achieving while going through the regimen that he/she has selected. One of the main premises that the project is based on is that the user must be able to see his/her progress that allows them to stay focused and motivated towards the goal. This is reflected through the relatively high priority of this use case.

The control sequence below only covers the web interface. A similar sequence can be developed for the phone platform.

:: include login

1W	→	User navigates to view progress tab under his profile
2W	←	System displays options to view daily progress and overall progress
3Wa	→	User selects daily progress
3Wb	<- -	System sends request to database to fetch information for the statistics of that day
3Wc	- ->	Database recovers daily performance information for the user and sends it to the system
3Wd	←	System displays the information for daily performance which includes speed at which workout was completed, calories that were burned and distance covered
4Wa	→	User selects option to view overall progress
4Wb	<- -	System sends request to database to fetch information for the overall statistics of that user
4Wc	- ->	Database recovers overall performance information and sends it to the system
4Wd	←	System displays the information for progress which includes percentage completion of the overall regimen, weight and body measurement variation based on previous check ins etc.

The control flow for this use case either follows the path
1W→2W→3Wa→3Wb→3Wc→3Wd or
1W→2W→4Wa→4Wb→4Wc→4Wd

Use Case Name: View Progress	**Identifier : UC12**
Requirements Catered	REQA7b, REQA12a, REQA12b, REQA12c
Initiating Actor	Logged-in User
Actor's Goal	View progress report on web/ view progress after current workout on mobile platform
Participating Actor	Mobile Application(supporting actor) /Web Application(supporting actor)
Pre-Condition	Actor has previous workouts saved in database(Web Application) Actor has finished current active workout(Mobile Application)
Post-Condition	Actor has information about most recent workout and navigates to home screen(Mobile Application) Actor acquires overall progress information(Web Application)
Scenario	**Expected Result**
1. Actor selects to navigate to any other page(Mobile Application)	Actor is directed to the requested page.
2. Actor does not have previous workout history (Web Application)	No progress report displayed. Notification "No history to display"

3.7.4 Use Case: Navigate to Information Pages (UC14)

The project is mostly information and data oriented hence information based pages are an important feature for this project. It is one of the features which provide equivalent privileges to both the registered and the unregistered users. The control flow of the use case is similar for most information pages and is the same for both categories of users and can only be implemented through the web interface.

::exclude login (implies that login is optional for this use case)
user can be at any of the pages of the website to access the navigation bar

1W	→	User selects the navigational button for the specific information that he/she requires
2W	<--	System requests necessary information from the database
3W	-->	Database returns the requested information
4W	←	System displays the information page requested by the user

Use Case Name: Navigate to Information pages	**Identifier : UC14**
Requirements Catered	REQW3(a, b, d-f), REQW2
Initiating Actor	Logged-in User/Visitor
Actor's Goal	Acquire information regarding diet plans during workout routines, Mobile Application, Community searching, Blog, Regimens or information about the software and the developers.
Participating Actor	Web Application(supporting actor)
Pre-Condition	User should be on the website URL
Post-Condition	User navigates to one of the requested pages
Scenario	**Expected Result**
1. User selects to navigate to any other page	Actor is directed to the requested page.
Comments	This use case covers most navigational options available for a visitor to the website or a logged-in user who wishes to acquire information regarding the aforementioned topics

The control flow for this particular use case is simpler than most other use cases. However this does not reduce the importance or the complicacy of this use case as most of the requisite information has to be gathered through extensive research.

3.7.5 Use Case: Start Workout/End Workout (UC4/UC5)

These two use cases capture the overall process of recording the user's workouts. It is important to understand that the major chunk of real time data acquisition is accomplished through these two use cases. It has thus been considered fit to discuss the two together as a continuous step as one paves way to the other and start is a prerequisite for end workout.
Control flow for this combined Use Case has been clubbed starting from Start() and ending at displayPerformance().

::include Login, Select Regimen

1A	→	The user navigates to start run page after reading regimen information and selecting day of the regimen
2A	←	System provides user a button to start workout
3A	→	User initiates the workout by pressing start button
4A	←	System displays timer, distance and speed info and a button to end the workout
5A	→/←	User selects end workout option/System terminates workout based on timeout
6A	<- -	System sends the performance information to database
7A	←	System displays performance for the workout to the user
8A	←	System displays option to check workout on the web application and provides option to go to homepage
9Aa	→	User selects home
9Ab	←	System directs user to home screen of phone application
10Aa	→	User selects view performance on web application option
10Wb	←	System directs user to web application for viewing progress

Use Case Name: Start Workout	**Identifier : UC4**
Requirements Catered	REQA4, REQA5a, REQA5b, REQA6c
Initiating Actor	Logged-in User
Actor's Goal	To start the workout
Participating Actor	Mobile Application(supporting actor),GPS(supporting actor),Accelerometer(supporting actor)
Pre-Condition	Actor is logged in user and had selected the regimen
Post-Condition	Actor is directed to running page and can view the timer, stop button, hear alert audio and current speed
Scenario	**Expected Result**
1. User selects button to navigate to home page	No data is stored for that workout and application returns to the home screen.
Use Case Name: Stop/End Workout	Identifier : UC5
Requirements Catered	REQA4, REQA7a, REQA7b, REQA9

Initiating Actor	Logged-in User
Actor's Goal	To end the workout
Participating Actor	Mobile Application(supporting actor), Database(supporting)
Pre-Condition	Actor has started the workout
Post-Condition	Actor is directed to the performance page that contains link to website to view complete performance
Scenario	**Expected Result**
1. User selects button to navigate to home page	No data is stored for that workout and application returns to the home screen.
2. User selects the end button	Actor is directed to the performance page which displays the day's performance. Data is updated in the database.

There is a branch point at step 9 of the control flow where the user can either decide to stay on the phone application or move to the web application. This choice leads to either control flow 8A→9Aa→9Ab or 8A→10Aa→10Wb. Also, the identifier '10Wb' implies that this action is fulfilled using the web application/ interface.

3.8 SYSTEM SEQUENCE DIAGRAMS

The previous discussion provides a sequential discussion of control flow for execution of each high priority use case. The same can be accomplished through system sequence diagrams which are a key visual aide for such scenarios.

3.8.1 Use Case: Select Regimen (UC3)

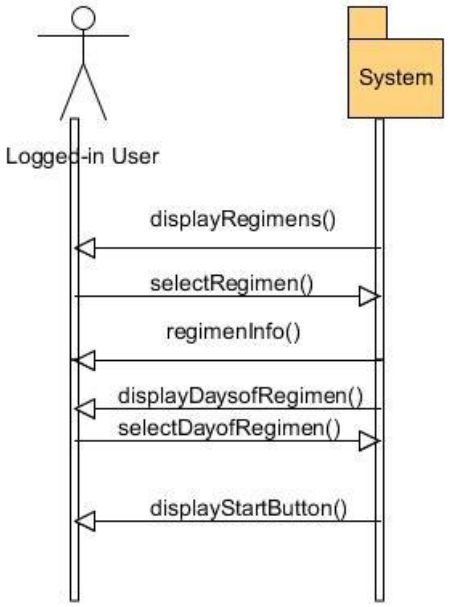


Fig 3.10

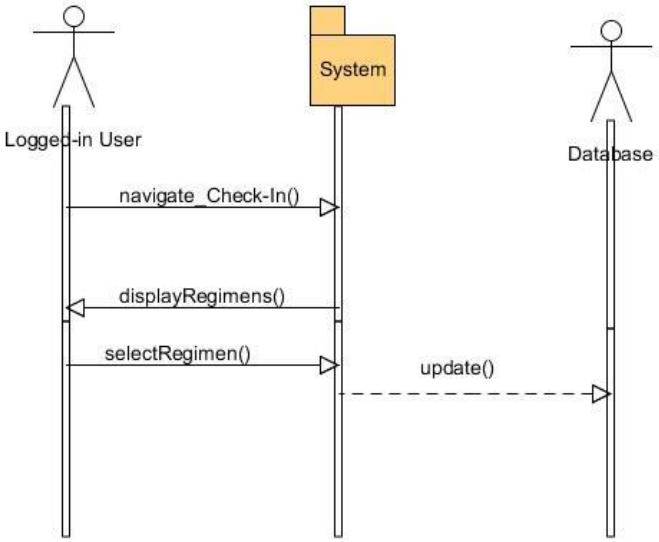


Fig 3.11

3.8.2 Use Case: Sign-Up()

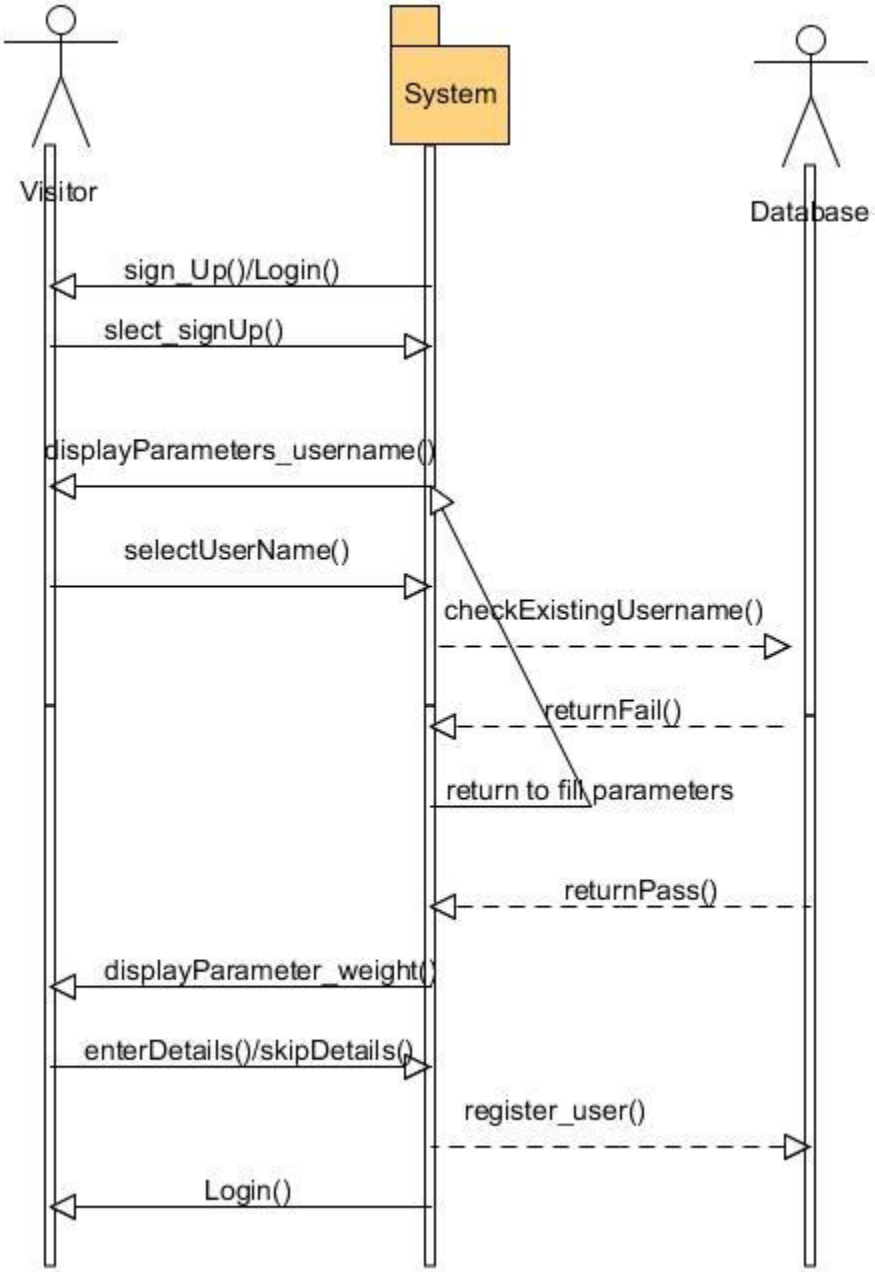


Fig 3.12

3.8.3 Use Case: View Progress (UC 12)

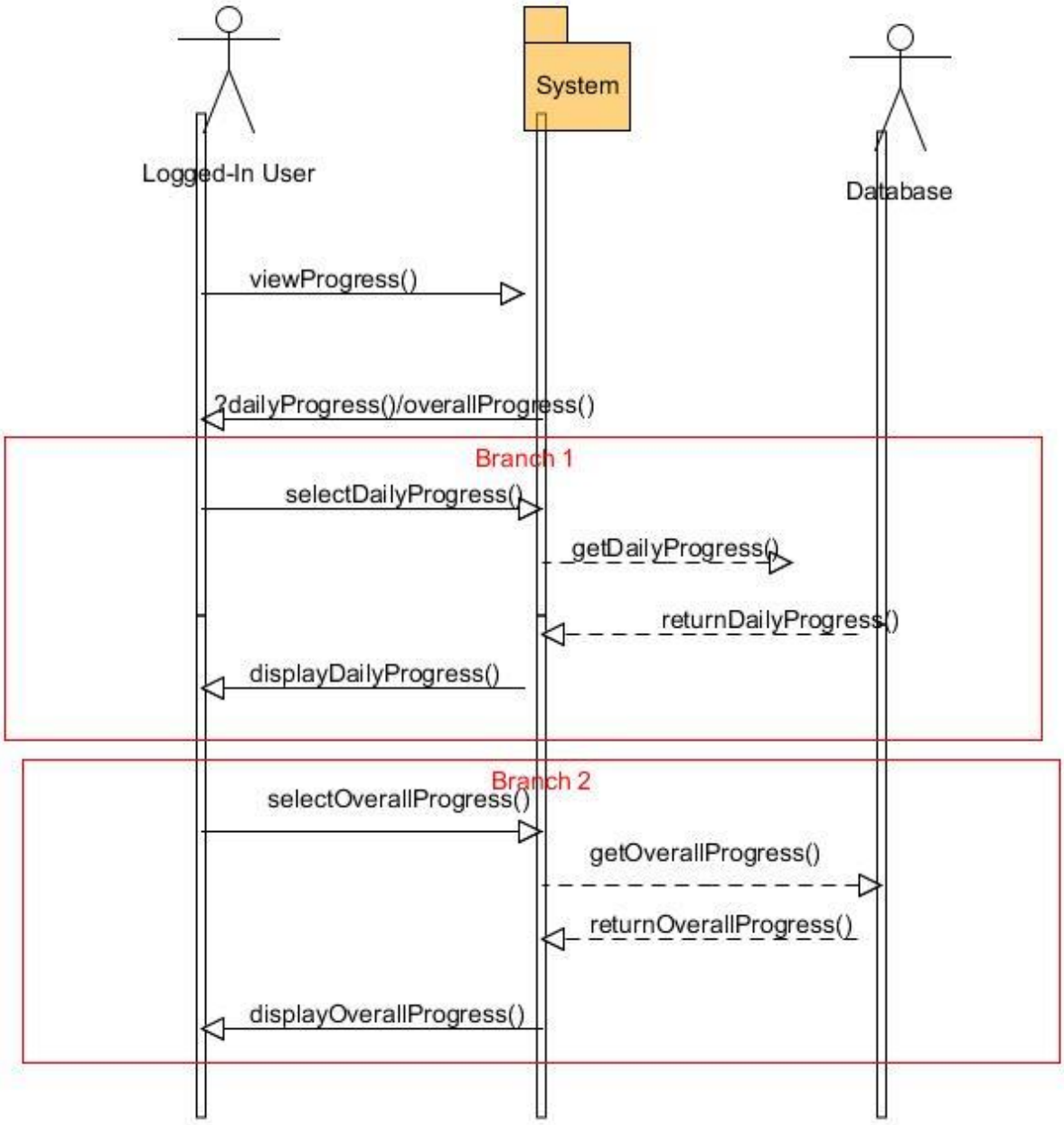


Fig 3.13

3.8.4 Use Case: Navigate To Information Pages (UC 14)

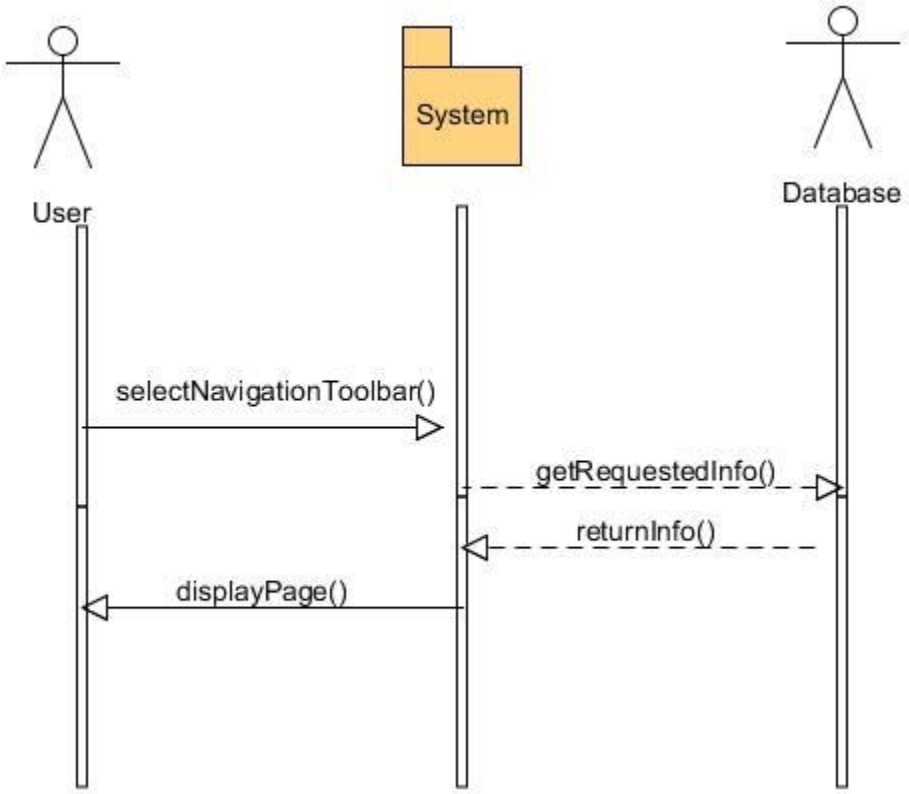


Fig 3.14

3.8.5 Use Case : Start/End Workout

The two have been discussed together to get a complete picture of the whole process of the workout.

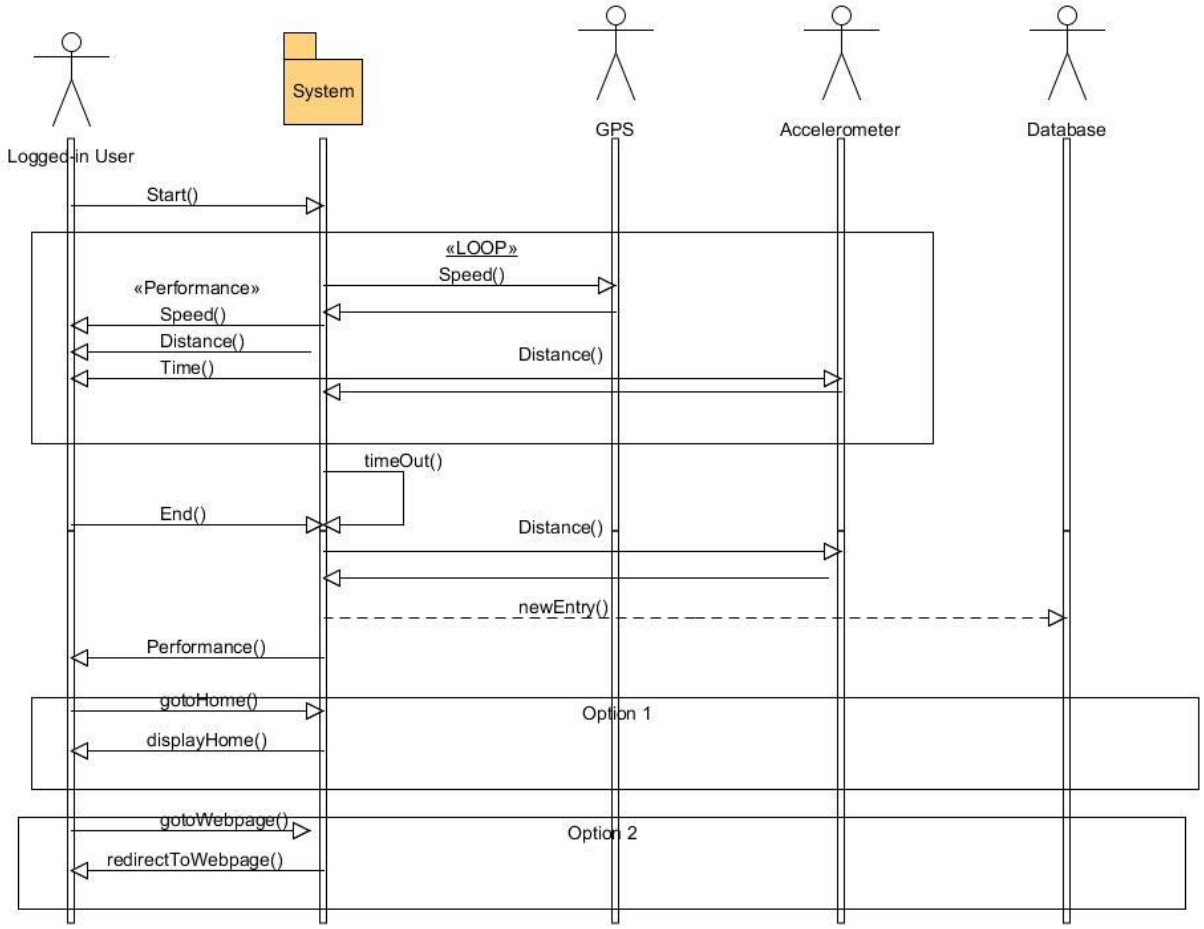


Fig 3.15

SECTION 4: USER INTERFACE SPECIFICATIONS

4.1 USER INTERFACE SPECIFICATIONS

In order to be able to design a good user interface *Visualization* is the key. It is important for the developers to 'see' in their minds what they are aiming at. User interfaces are also extremely important components of creating a good user experience. For this project there are two user interfaces that allow user's interaction with the system at two different levels. The *mobile interface* allows real time data logging and immediate performance monitoring. The *web interface* is a more extensive interface that allows profile monitoring, interaction between users, progress monitoring and update of personal statistics etc. It is the platform that would allow comparison based motivation and community searches. We have tried to generate mock screen shots of what the final software would look like.

4.1.1 Web Interface

Fig 1 : Home Page

The home page is the first page that the user lands at once he enters the website's URL in the Omni-bar of the web browser. The home page contains navigational links to all the other pages and allows an overview of the content that lies ahead. It is also notable that the navigational links provided on this page are used as a template in the other page to maintain homogeneity for

access to any page whenever the user wants thus providing navigational ease and improving user experience.
The rest of this section tries to visually cover most of the important use cases to give a flavor of what is being developed.

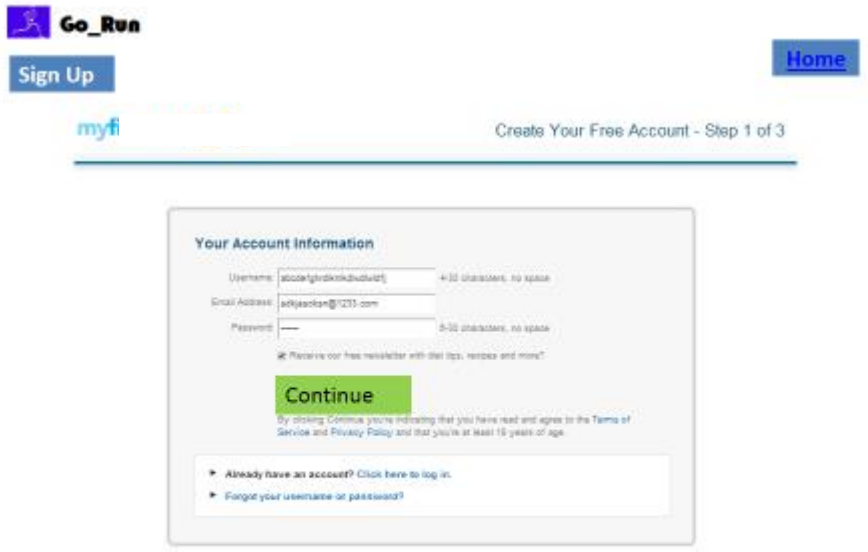


Fig 4.2 Sign-Up Page 1

Sign-up is an important feature covered through a high priority use case (UC1) and allows the users to register with the product's database. It would be divided into three segments. The first segment asks about mandatory information like username and password. After this information is acquired the system would verify for existing instances of the same account and return a pass or a fail based on which the user would either progress to the next segment or stay at the same page to create a new identity that does not previously exist in the database.

Fig 4.3: Sign-up Page 2

The information like weight, body measurements and regimens can either be selected at the sign-up or updated later during account management through check-in feature.
After this screen the user is asked to register and is then directed to the login page where he is asked to login to view his profile.

Fig 4.4: Login Page

After logging in(UC2) registered users can move to their personal profile page where information pertaining to personal account as well as other navigational links to informational pages are present (UC14).

Fig 4.5: Personal Home Page

The personal home page for the user is laden with information including overview of daily progress, BMI, Ranking; Regimen registered for and link to the alerts page. It must also be noted that every page after logging in has a link to *logout* which is a specific appearance requirement (REQW12). All the links provided in the navigation bar redirect to devoted information pages. The most important out of these are *Progress* (UC12), and *Check-In* for selecting regimen (UC3).

Progress can be divided into daily performance based progress and overall progress, which are displayed using devoted pages.

Fig 4.6: Overall Progress

The project website contains a complete UI design with functional navigation in a .pptx format, that can be viewed in slide show mode. To view it please visit the website and search under the project documents section. https://sites.google.com/site/gorunsegroup/

4.1.2 Phone Interface

The phone interface is more or less sequential in nature and the overall picture can be clearly understood when going over screen mock-ups that follow.

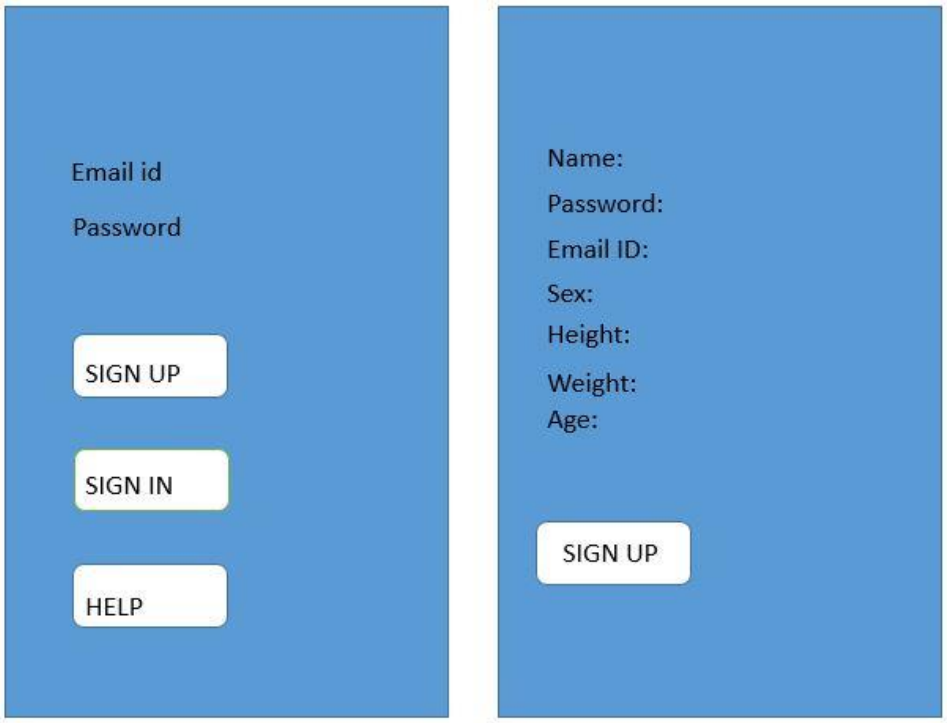


Fig 4.7: Login and Signup Pages

The user is provided with the login page on opening the application. In case the user is not registered he is directed to a sign-up page where he can create an account.

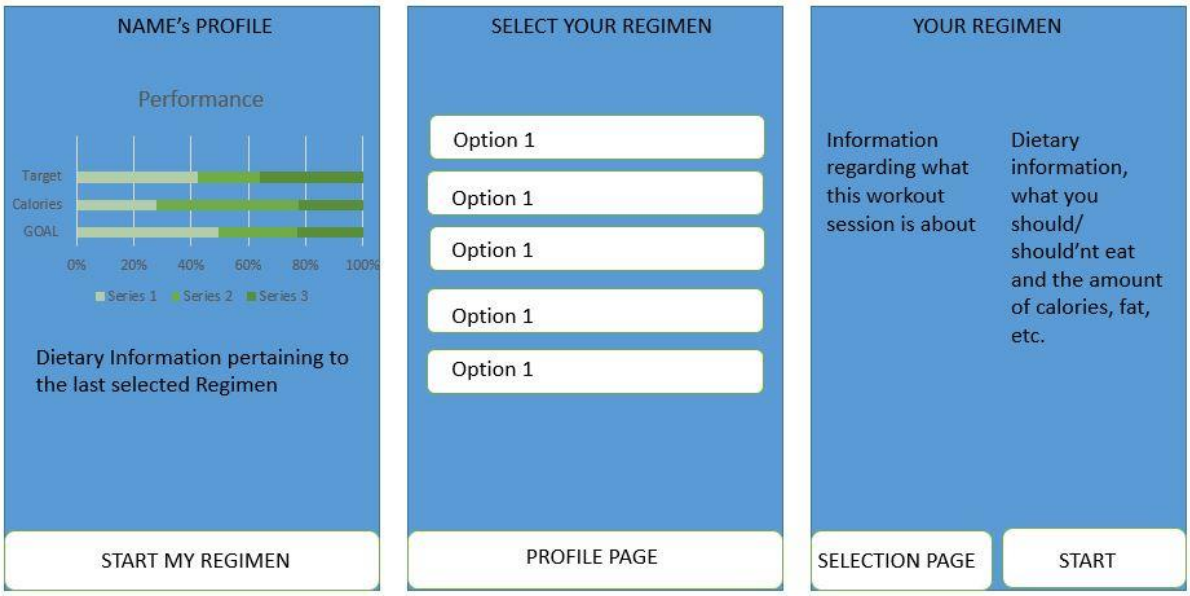


Fig 4.8: Profile Page, Regimen Selection and Start workout page

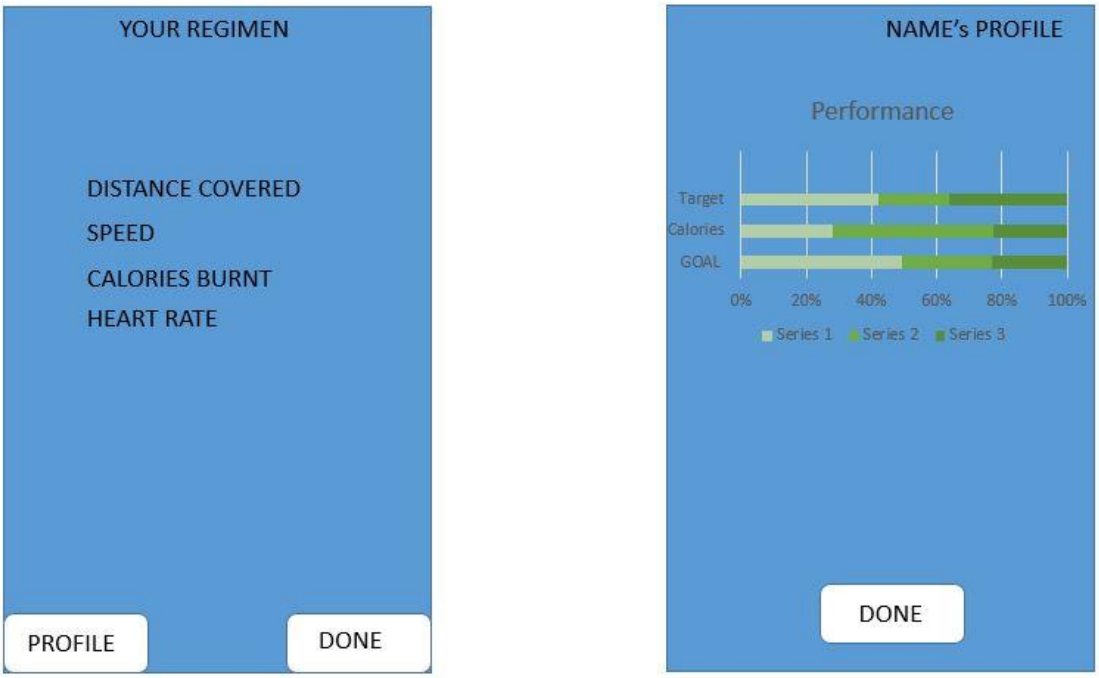


Fig 4.9: Ongoing workout and Performance page

After the workout is complete the user is automatically directed to a current performance page from where he is taken back to the homepage.

4.2 EFFORT ESTIMATION

Effort is the time which the team puts in to produce the system, estimating the effort is based on the complexity of the use cases that need to be implemented. The estimation is the raw time that is suggested to produce the work. This section tells us how the estimation is done. The estimation is based on the following formula:

*DURATION = Use case Points * Productivity factor.*

Here the use case points are based on the complexity of the use cases, the number of actors involved and the number of scenarios that can be possible for a use case. The highest factor for the use case points can be 10 which mean that more than 3 actors are involved and data needs to be collected or updated, Value 1 being the lowest meaning that the user has no action, which does not result in any effort. The number of hours that sold be invested in this is highest for value 10, this is a subjective matter, thus for simplicity a number of 20 hours is used for the value point 10.

Productivity factor is the productivity of the team members who are developing the system. Clearly between the productivity and time we see an inverse relationship, thus a fraction is used to depict the amount of time that will be required. The formula now adjusts to:

*Duration = Hours * (1/PF)*

The estimated time does not include the time spent on the design and architecture. To compensate for that we use a factor of 20% of the estimated time to be added for the design and architecture.

Thus the time estimated will be

*Estimated Time = (1 + .2)*Duration*

In the following chart we can see the various Use case points and the number of hours for each use case along with the productivity factor.

Use case #	Use Case Points	Hours	PF	1/PF	Time estimated
UC1	9	18	0.7	1.428571	25.71428571
UC2	8	16	0.7	1.428571	22.85714286
UC3	5	10	0.7	1.428571	14.28571429
UC4	10	20	0.7	1.428571	28.57142857
UC5	10	20	0.7	1.428571	28.57142857
UC6	5	10	0.7	1.428571	14.28571429
UC7	5	10	0.7	1.428571	14.28571429
UC8	4	8	0.7	1.428571	11.42857143
UC9	9	18	0.7	1.428571	25.71428571
UC10	3	6	0.7	1.428571	8.571428571
UC11	4	8	0.7	1.428571	11.42857143
UC12	9	18	0.7	1.428571	25.71428571
UC13	10	20	0.7	1.428571	28.57142857
UC14	6	12	0.7	1.428571	17.14285714
UC15	7	14	0.7	1.428571	20
UC16	2	4	0.7	1.428571	5.714285714
UC17	2	4	0.7	1.428571	5.714285714
UC18	8	16	0.7	1.428571	22.85714286
UC19	8	16	0.7	1.428571	22.85714286
UC20	3	6	0.7	1.428571	8.571428571
UC21	3	6	0.7	1.428571	8.571428571
UC22	5	10	0.7	1.428571	14.28571429
UC23	6	12	0.7	1.428571	17.14285714
					402.8571429

Fig 4.10: Estimation

Duration = 402.85 hours
Estimated Time = (1+.2)*402.85 hours = 483.42 hours.
The total time estimated is **438.5 hours.**

SECTION 5: DOMAIN ANALYSIS

The domain analysis section deals with the interaction between various attributes inside the system. In the earlier section of the report you have seen "what" the system does from the customers point but over here we are going to discuss "how" the system actually reacts/works internally when the action is required to be done.
To build the domain model, we will revisit the fully dressed description of the use cases, doing so will expose all the important concepts about the system.

5.1 CONCEPT DEFINITION

The whole system is seen as one when see from the user's side, but internally the system is a network of various functionalities. Here first we will work on the boundary definition, which shows how each the concept is executed.

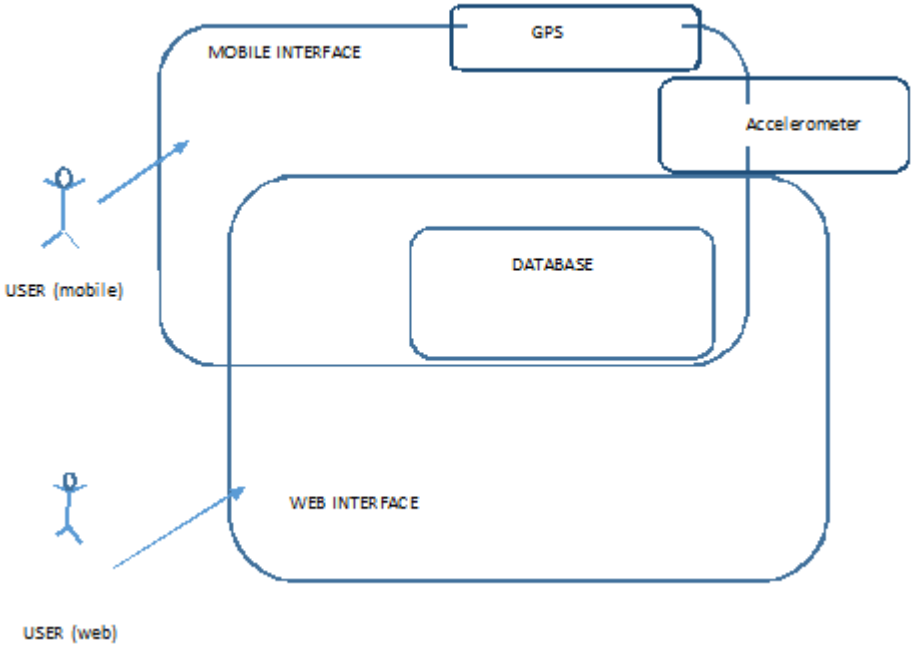


Fig 5.1

The figure 5.1 shows how the users interact with the boundary without having the knowledge of the inner sub systems.
The responsibilities shared by each of the interface can be classified as a "do" or "knowing" concept. Each of the responsibility shared between the database and other sub systems is a knowing concept and others are mostly do concepts.

- R1: User sign up (signup)
- R2: User Login(login)
- R2: Check user for valid - registry (checkuser)

- R3: Store user information (storeuserinfo)
- R4: Select the regimen(selectregimen)
- R5: Store regimen information(storereginfo)
- R4: Read Accelerometer data (readaccelerometer)
- R5: Calculate the Running speed (calcspeed)
- R6: Measure Lap time (calclaptime)
- R7: Read GPS data (readlocationdata)
- R8: Store users location and speed data (storedata)
- R9: calculate goal details (goalcalcualtion)
- R10: calculate alert information(calcalertinfo)
- R11: calculate ranking(calcrank)

The above listed responsibilities are the do responsibilities that the system performs during the course of operation. Most of the responsibilities can be split into a lower level of responsibilities, but having defined the responsibilities are based on the level of interaction between the internal sub systems. For ex, the Read GPS data can be further split into the fetch the GPS ping, store the GPS ping and then to decrypt the ping into understandable coordinates. Having done that we would have gone into the working of the subsystem which is outside the scope of the project.
Thus the above mentioned responsibilities are all “do” concept based. Now we will work on the “knowing” concept based responsibilities. The majority of the knowing responsibility is shouldered by the web interface as shown in the fig 5.1.

- R12: show user information(readuserinfo)
- R13: show target and goals(readusergoal)
- R14: show the different regimens available(readregimens)
- R15: show user statistics-calories, number of times exercise done (readstatistics)
- R16: show regimen information(readreginfo)
- R16: show user alerts(readalertinfo)
- R17: show forum posts(readfposts)
- R18: show regimens option(dispregimen)
- R18: show community disunion posts(readcommposts)
- R19: show dietary information(readdietinfo)
- R10: show ranking(readranking)
- R21: show search option(dispsearch)

- R22: show distance covered and lap time(readdtinfo)

All the above are the knowing concepts which are informational concepts are reading the data from the database which was updated by the do concepts. All the calculation and storing of data is done in the “do” responsible operations.

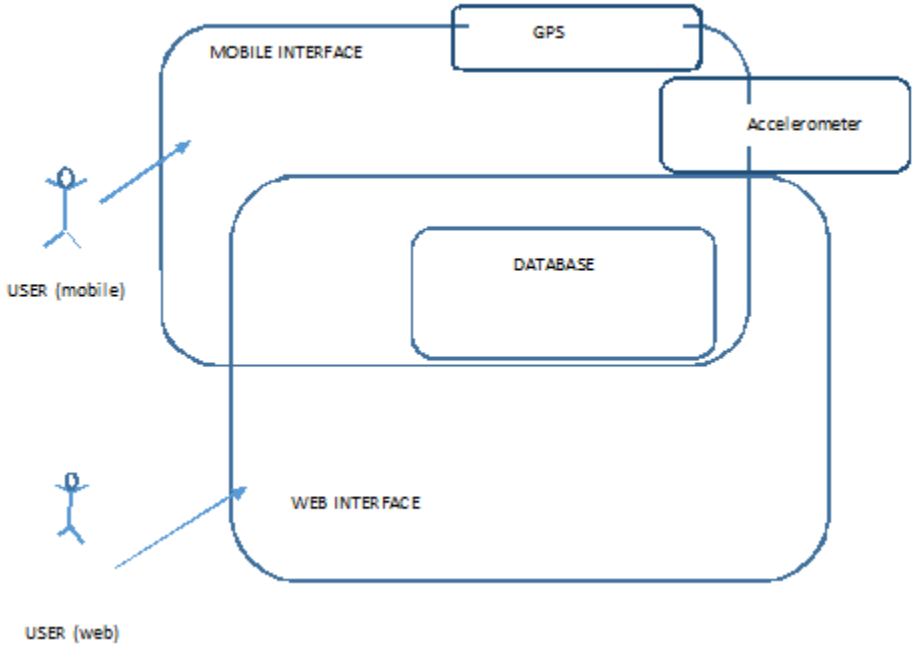


Fig 5.2

Along with above we also have the interfaces as concepts because all the information is being either fetched from or to the interfaces.

R23: web interface(showuserdata)
R24: web interface for storing(storeuserdata)
R25: mobile interface (showusmdata)
R26: create user profile(createprofile)
R27: prompt user for correct information(promptuser)

5.2 ASSOCIATION DEFINITIONS

The system which is seen in fig 5.1 and fig 5.2 show how the concept has been implemented into the system. We have worked on the various aspects of the system and come up with 23 use cases. The working of the system is defined in the internal concepts but the association between the various subsystems and interface is defined in this section for better clarity and information of how the system is working in synchronisation.
We will now discuss the use cases which have the maximum weights among the 23 use cases, we choose these use cases because they cover the system extensively in its functionality. We will thus discuss only 7 use cases here.

Domain Concept 1: Sign up(mobile and web)

Concept pair	Association description	Association name
Signup<-> checkuser	Once the user input the username and password it is verified if the user already exits or not	Verifies user
Signup<->checkuser<->promptuser	If the user is already registered, prompt user to use existing user name and password	Prompt user
Checkuser<->storeuserinfo	If the user is verified to be a new user then store the user information	Store user information
Storeuserinfo<->createprofile	Once the user information is verified and stored an empty user profile is created	Profile creation
Createprofile<->showuserdata	The user is now able to see an empty profile page	Profile page

The user in this use case signups for an account, the system first checks if the user already registered or not. The database is checked for the user information if already existing and prompts the user for correct information.
The system creates an empty profile when the user has fed the system with valid information. All data is stored in the database and is being fetched to the mobile/web interface whichever the user is using.

Domain Concept 2: Login(mobile and web)

Concept pair	Association description	Association name
login<-> checkuser	Once the user input the username and password it is verified if the user already exits or not	Verifies user
login<->checkuser<->promptuser	If the user is already registered ,but the information is not coherent, prompt user to use existing user name and password	Prompt user
login<->showuserinfo	If the user is verified then the user information is displayed from the database	Store user previous information
Showuserinfo<-> showuserdata	The user is now able to see an empty profile page	Profile page

This is the login use case where the user already has an account and now logs into his account on the web interface or the mobile interface. The device then checks with the database if the username and password is coherent, if yes then the interface displays the user information and the profile page where the data is loaded from the database otherwise it displays the user a prompt message to re-input the details.

Domain Concept 3: Select Regimen (Mobile only)

Concept pair	Association description	Association name
Dispregimen<->readregimens	The regimens available are read from the database	display regimen
Selectregimen<->storereginfo	The system stores the regimen selected by the user	Store regimen
Selectregimen<->readdietinfo	The user once selects the regimen can see the dietary information for the particular regimen	Show dietary information
Selectregimen<->readreginfo	The user also can see information regarding the regimen he/she selected	Show regimen details

This use case deals with the selection of the regimen that the user wants to do. The user sees the dietary information and how to go about the regimen.

Domain Concept 4: Start Workout (Mobile only)

Concept pair	Association description	Association name
Readaccelerometer<->calcspeed	The accelerometer data is used to calculate the running speed	Calculate speed
calcspeed<->readdtinfo	The system displays the running speed at real time, no data from the data base is accessed	Display speed
calclaptime<->readdtinfo	The system displays the lap time.	Show lap time information
Readlocationdata<->readdtinfo	The user is also able to see the distance he has covered	Show distance details

The system at real time reads the accelerometer, GPS and timer which the user can see on the mobile interface. This is the real-time data is maintained on the users mobile device till the time he is working out.

Domain concept 5: Stop Workout (Mobile only)

Concept pair	Association description	Association name
Readdtinfo<->storedata	The distance, time taken and speed of the user is stored	Store data
Storedata<->goalcalc	The system calculates the goals of the user.	Days achievement
Goalcalc<->storedata	The system calculate and stores the goals.	Store goal data
storedata <->Readusergoal	The system then displays the goal data	Display goal data

The stop workout case is used after the start workout, this calculates the goals and stored them in the database.

Domain Concept 6: View Progress (Web only)

Concept pair	Association description	Association name
Readuserinfo<->showuserinfo	The web interface displays the user information	Store data
Readalertinfo<->showuserinfo	The system calculates the goals of the user.	Days achievement
Readstatistics<->showuserinfo	The system calculate and stores the goals.	Store goal data
Readranking<->showuserinfo	The system then displays the user ranking	Display users rank

This use case displays all the user profile based information when a user logs into his account, the rank and the goal are the most important aspects of the jogging regimen which are displayed.

Domain Concept 7: Navigate to information page (Web Only)

Concept pair	Association description	Association name
Readuserinfo<->showuserinfo	The web interface displays the user information	Store data
readranking<->showuserinfo	The web page navigates to the ranking page.	Ranking page
readdietinfo<->showuserinfo	The page opens the dietary information	Dietary page
readusergoal<->showuserinfo	The page displays the goal information	Progress page
Readregimens<->showuserinfo	The page displays the regimen information page	Regimen page
Readcommposts<->showuserinfo	The page displays the community posts page	Community page

This use case is extensive when considering the information that is shared. All the data is fetched from the database and displayed at the corresponding web page.

5.3 ATTRIBUTE DEFINITION

In the above section we have seen the various boundaries and concepts that are used to explain the project model. In this section we will elaborate the attributes that have been used and how they interact with the subsystems and interfaces, only then we can fully understand the above written associations completely.
We have used the table below to simplify the attributes and their interaction.

Concept	Attribute Description
storedata	The concept stores information in the database, this data can be any kind of data which is related to the user.
signup	The sign up the concept where the user inputs the required information
Login	This concept is the user login where the user inputs the username and password
Checkuser	This concept verifies the user data from the database
Storeuserinfo	This concept stores the user information into the database
Selectregimen	This lets the user select one among the various regimen
Storereginfo	This stores the user registration data into the database

Readaccelererometer	This reads the accelerometer data.
Calcspeed	This calculates the speed from the accelerometer data
Calclaptime	This calculates the lap time of the user
Readlocationdata	This reads the location of the user
Goalcalculation	This calculates the goal completed
Calcalertinfo	This mathematically calculates the alert information
Calcrank	This calculated the rank of the user taking data from the database
readuserinfo	This reads the user information that is available in the database
Readusergoal	This reads the stored user goal form the database
Readregimens	This reads the regimen information in the database
Readstatistics	This read all the progress statistics
Readreginfo	This reads the registered data from the database

5.4 TRACEABILITY MATRIX

In the above sections we have defined how the system interacts with various components. We have defined 7 domain concepts which encompass over the various use cases that have been devised.

The traceability matrix can be found in the figure 5.1.

	signup	login	checkuser	storeuserinfo	selectregimen	storereginfo	readaccelerometer	calcspeed	calclaptime	readlocationdata	storedata	goalcalculation	calcalertinfo	calcrank	readuserinfo	readusergoal	readregimens	readstatistics	readreginfo	readalertinfo	readfposts	dispregimen	readcommposts	readdietinfo	readranking	dispsearch	readdtinfo
UC1	X		X	X							X				X	X	X	X	X	X				X	X		
UC2		X	X												X	X	X	X	X	X				X	X		X
UC3					X	X									X		X					X		X			
UC4							X	X	X	X												X		X			X
UC5											X	X	X	X	X	X											
UC6			X			X					X																
UC7															X	X	X	X							X		
UC8																								X			
UC9																											X
UC10		X													X												
UC11											X																
UC12															X	X	X	X	X	X		X					
UC13										X					X	X		X	X							X	
UC14										X					X	X	X	X	X	X	X		X				
UC15													X							X							
UC16				X																							
UC17				X	X																						
UC18																										X	X
UC19																								X			
UC20											X												X				
UC21			X			X					X																
UC22			X								X																
UC23																					X						

Fig 5.3: Traceability Matrix

5.5 MATHEMATICAL MODEL

The project has many different components that require devoted mathematical models to be implemented. Based on priority some models have already been adapted and would be presented in the following sections.

For accurate distance and speed measurement it is always advisable that data analysis should not be individually based on GPS or accelerometer data. This implies that both should be implemented simultaneously to acquire data.

5.5.1 Accelerometer data analysis

Accelerometers are most commonly used in pedometers, devices that count number of steps that an individual covers. It is meticulously converted into distance by generating distance and step correlations based on the height of an individual and the variations in acceleration data produced. We intend to acquire accelerometer data from the user's phone in crude form and use the concept of step counting from algorithms that are implemented by pedometers. The three components of motion for an individual (and their related axes) are forward (roll), vertical (yaw), and side (pitch). The accelerometer senses acceleration along its three axes: x, y, and z. The phone will be in an unknown orientation, so the measurement accuracy should not depend critically on the relationship between the motion axes and the accelerometer's measurement axes. At least one axis will have relatively large periodic acceleration changes, no matter how the pedometer is worn, so peak detection and a dynamic threshold-decision algorithm for acceleration on all three axes are essential for detecting a unit cycle of walking or running.

The algorithm that we would implement has originally been implemented via hardware but we would try to generate a software implementation on the same basis to calculate the number of steps. The basic idea is that a function named *interval* records how many times the data have updated during the two steps. If the value of interval is between 10 and 100, it means that the time between two steps is in the valid window; otherwise, the interval is outside the time window and the step is invalid.

Count regulation determines whether steps are part of a rhythmic pattern. The step counter has two working states: searching regulation and found out regulation. When the step counter starts working, it works in searching regulation mode. Suppose that *in regulation* exists after four continuous valid steps. Then the result is refreshed and displayed, and the step counter will work in found out regulation mode. Working in this mode, the step count would be refreshed after every valid step. But if even one invalid step is found, the step counter will return to searching regulation mode and search for four continuous valid steps.

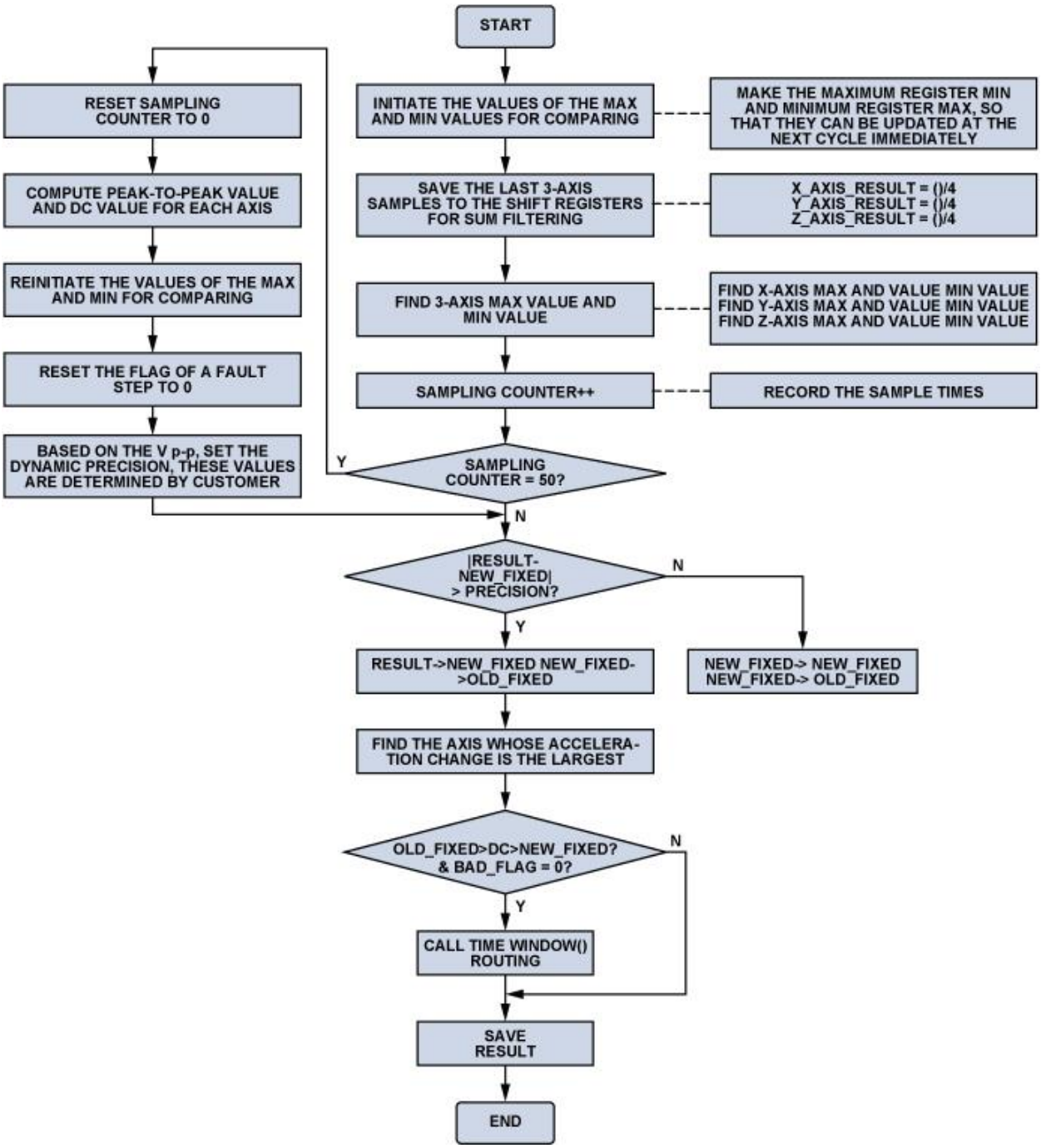


Fig 5.4: Algorithm for step calculation*

*The image and the algorithm have been taken from the reference http://www.analog.com/library/analogdialogue/archives/44-06/pedometer.html. Please refer to this website for an exhaustive coverage of this topic.

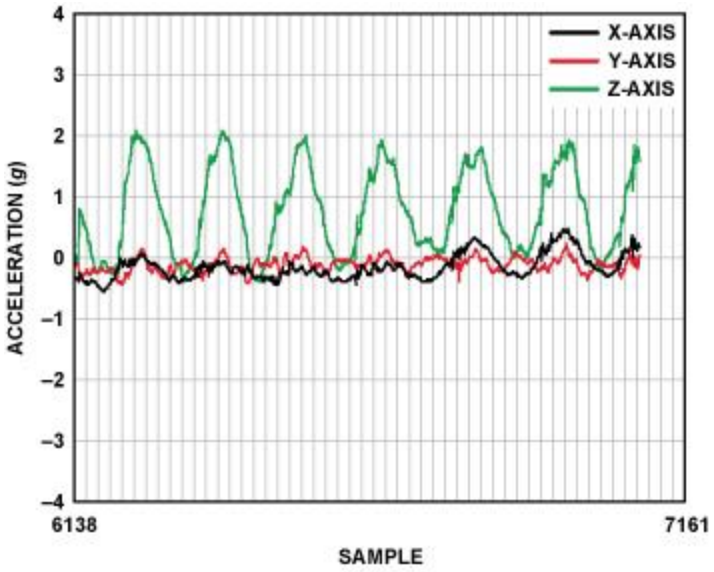


Fig 5.5: Rhythm in running

Based on which axis is experiencing a rhythm its frequency of occurrence is calculated to be the number of steps being taken. A step is defined as happening if there is a negative slope of the acceleration plot (sample_new < sample_old) when the acceleration curve crosses below the dynamic threshold. After the steps have been calculated the distance can be evaluated from the from the following formula

$$Distance = number\ of\ steps \times distance\ per\ step$$

Steps per 2 s	Stride (m/s)
0~2	Height/5
2~3	Height/4
3~4	Height/3
4~5	Height/2
5~6	Height/1.2
6~8	Height
>=8	1.2 × Height

Where number of steps per two seconds and distance per step can be calculated from the experimental values above. Distance per step depends on the speed and the height of user. The step length would be longer if the user is taller or running at higher speed.

5.5.2 GPS data Analysis

Each GPS satellite continuously transmits a microwave radio signal composed of two carriers, two codes, and a navigation message. When a GPS receiver is switched on, it will pick up the GPS signal through the receiver antenna. Once the receiver acquires the GPS signal, it will process it using its built-in software.

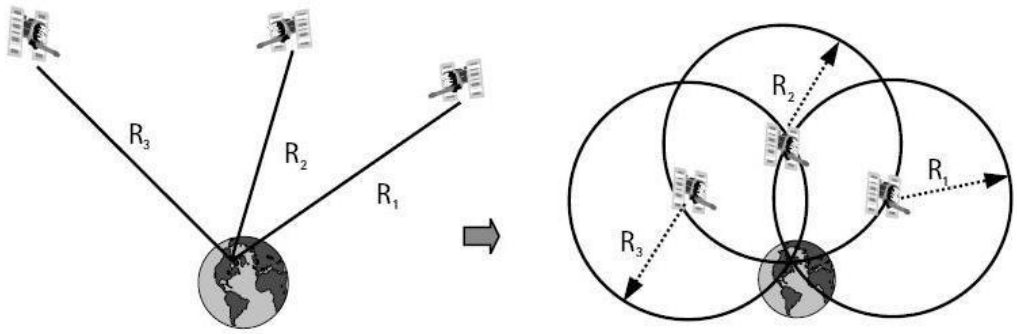


Fig 5.5: Basic ideas of GPS positioning

To exact the desired coordinates of the observer from these measurements, we construct a sphere of radius ri about each of four satellites. The equations these spheres are given by:

$$(x - x1)2 + (y - y1)2 + (z - z1)2 = r12$$
$$(x - x2)2 + (y - y2)2 + (z - z2)2 = r22$$
$$(x - x3)2 + (y - y3)2 + (z - z3)2 = r32$$
$$(x - x4)2 + (y - y4)2 + (z - z4)2 = r42$$

Where xi, yi, zi are the known coordinates of the satellites in the space and x, y, z are the unknown coordinate of the observer on the earth as shown on the Figure 2. We now subtract the first of these equations from each of the three. This eliminates quadratic terms in x, y, z and leads to the following set of nonhomogeneous linear equation in x, y, z:

$$2(x2 - x1)x + 2(y2 - y1)y + 2(z2 - z1)z = r12 - x12 - r2\,2 + x22$$
$$2(x3 - x1)x + 2(y3 - y1)y + 2(z3 - z1)z = r12 - x12 - r32 + x32$$
$$2(x4 - x1)x + 2(y4 - y1)y + 2(z4 - z1)z = r12 - x12 - r42 + x42$$

By using Gaussian elimination we can obtain the value of x, y, z, the coordinates of the observer with respect to a fixed known origin on earth.

These locations would be calculated at a fairly fast refresh rate and using google API for Maps we intend to send pings and receive distance between the two co-ordinates.

5.5.3 Calories Burned

The Complete Book of Running, which is written by Jim Fixx who is credited with helping start America's fitness revolution, popularizing the sport of running and demonstrating the health benefits of regular jogging, expressed the idea that there is almost no relationship between running speed and the amount of calories you burned, but the weight. Actually, not only weight but also speed plays an irreplaceable role in this situation. The faster you are, the more calories you are going to burn. Additionally, when running in the same speed, the heavier you are, the more calories you will consume.
The following table demonstrates the relationship between weight and consumption of calories when you take a one-hour running.

Calories Burned per Hour	130 lbs	155 lbs	180 lbs	205 lbs
5 mph	472	563	654	745
5.2 mph	531	633	735	838
6 mph	590	704	817	931
6.7 mph	649	774	899	1024
7 mph	679	809	940	1070
7.5mph	738	880	1022	1163
8 mph	797	950	1103	1256
8.6 mph	826	985	1144	1303
9 mph	885	1056	1226	1396
10 mph	944	1126	1308	1489
10.9 mph	1062	1267	1471	1675

We carried out regression analysis on this data and found out the following relationships and derived the following equations which are linear fits to the data with high R^2 values and hence can be used with high accuracy.

For person whose weight is not greater than 130 lbs, the relationship between speed and calories burned is:

Calories burned = 94.94 * speed + 18.80

$R^2 = 0.993$

For person whose weight is greater than 130 but not greater than 155 lbs, the relationship between speed and calories burned is:

Calories burned = 113.30 * speed + 21.83

$R^2 = 0.993$

For person whose weight is greater than 155 lbs but not greater than 180 lbs, the relationship between speed and calories burned is:

Calories burned = 131.50 * speed + 25.45

$R^2 = 0.993$

For person whose weight is greater than 205 lbs, the relationship between speed and calories burned is:

Calories burned = 149.60 * speed + 30.18

$R^2 = 0.993$

Using linear programing the following model can be devised where y_i are selection variables bound by an "or" constraint to select only one equation.

$$\boldsymbol{Calories\ burned} = \boldsymbol{y1} * (\boldsymbol{94.94} * \boldsymbol{speed} + \boldsymbol{18.80}) + \boldsymbol{y2} * (\boldsymbol{113.390} * \boldsymbol{speed} + \boldsymbol{21.83}) + \boldsymbol{y3} * (\boldsymbol{131.50} * \boldsymbol{speed} + \boldsymbol{25.45}) + \boldsymbol{y4} * (\boldsymbol{149.60} * \boldsymbol{speed} + \boldsymbol{30.18})$$

y1 = 1 when weight < 155 lbs y2 = 1 when 155 lbs ≤ weight < 180 lbs

y3 = 1 when 180 lbs ≤ weight < 205 lbs y4 = 1 when 205 lbs ≤ weight

constraint on selection variable forcing "or" selection

$$y1 + y2 + y3 + y4 = 1$$

5.5.4 Alert Model

Alerts are a new feature that we are adding to our project which is not available in any of the previously existing softwares available in the market. We had to develop a model from scratch for this using linear programing for cell selection. We calculated the percentage loss that might occur due to one day skipped during a regimen which were taken from literature references. Then, similar to a penalty model percentages were allotted to each day in an array and a selection method was implemented. Whenever a person misses one day of regimen the penalty imposed is equal to the cell value from that regimen table which would be maintained as local data on the backend of the software.

For 5K-run:

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Week 1	2.60	1.73	2.60	1.73	0	4.34	0
Week 2	2.60	1.73	2.60	1.73	0	4.34	0
Week 3	2.80	1.86	2.80	1.86	0	4.68	0
Week 4	2.80	1.86	2.80	1.86	0	4.68	0
Week 5	3.00	2.00	3.00	2.00	0	5.00	0
Week 6	3.00	2.00	3.00	2.00	0	5.00	0
Week 7	3.20	2.13	3.20	2.13	0	5.34	0

Constraints $\sum_{j=0}^{7}\sum_{i=0}^{7} Y_{i,j} \leq 1$

$$\sum_{j=0}^{7}\sum_{i=0}^{7} R_{i,j} = 100$$

Model $\text{Penalty} = R_{i,j}Y_{i,j}$

For 10K-run

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Week1		1.5		2		1.5	2.5
Week2		1.5		2		1.5	2.5
Week3		1.5		2		1.5	2.5
Week4		1		2		1	3
Week5		2		2		2	3
Week6		2		2		2	3
Week7		2		1.5		2	3.5
Week8		1.5		2		1.5	3.5
Week9		2		2.5		1.5	4
Week10		2.5		2.5		1.5	4
Week11		2.5		2		2	3
Week12	2		1.5		1.5		

Constraints

$$\sum_{j=0}^{7}\sum_{i=0}^{12} Y_{i,j} \leq 1$$

$$\sum_{j=0}^{7}\sum_{i=0}^{12} R_{i,j} = 100$$

Model

$$\text{Penalty} = R_{i,j}Y_{i,j}$$

For Marathon:

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Week1	0.238815	1.830919	0.477631	0.796052	0	0.636841	1.751313
Week2	0.238815	1.830919	0.477631	1.114472	0.398026	0.636841	1.910524
Week3	0.238815	1.830919	0.796052	1.114472	0.557236	0.636841	2.388155
Week4	0.238815	1.751313	0.796052	1.432893	0.557236	0.636841	2.388155
Week5	0.398026	0.796052	0.636841	1.273683	0.796052	0	2.30855
Week6	0.238815	0.716446	0.796052	1.432893	0.796052	0.398026	2.388155
Week7	0.238815	0.636841	0.796052	1.432893	0.796052	0.398026	2.069734
Week8	0.238815	0.716446	0.796052	1.273683	0.796052	0.398026	2.228944
Week9	0.238815	0.636841	0.796052	1.273683	0.796052	0.398026	1.990129
Week10	0.238815	0.636841	0.796052	1.273683	0.796052	0.398026	2.706575
Week11	0.238815	0.732367	0.796052	1.432893	0.796052	0.398026	2.149339
Week12	0.238815	0.955262	0.796052	1.432893	0.796052	0.398026	3.263811
Week13	0.238815	0.796052	0.796052	1.273683	0.796052	0.398026	2.30855
Week14	0.238815	0.636841	0.796052	1.432893	0.796052	0.398026	1.751313
Week15	0.238815	0.636841	0.636841	0.636841	0.477631	0.398026	1.273683
Week16	0	0.477631	0.398026	0.636841	0.159211	0.477631	0

Constraints $\sum_{j=0}^{7}\sum_{i=0}^{16} Y_{i,j} \leq 1$

$\sum_{j=0}^{7}\sum_{i=0}^{16} R_{i,j} = 100$

Model $\text{Penalty} = R_{i,j}Y_{i,j}$

SECTION 6: INTERACTION DIAGRAMS

6.1.1 INTERACTION DIAGRAMS

This section describes how the system interacts with the users and different external actors including the database, accellerometer and the GPS. The section covers only the most important scenarios pertaining to use cases and domain concepts with the highest priority weightage.

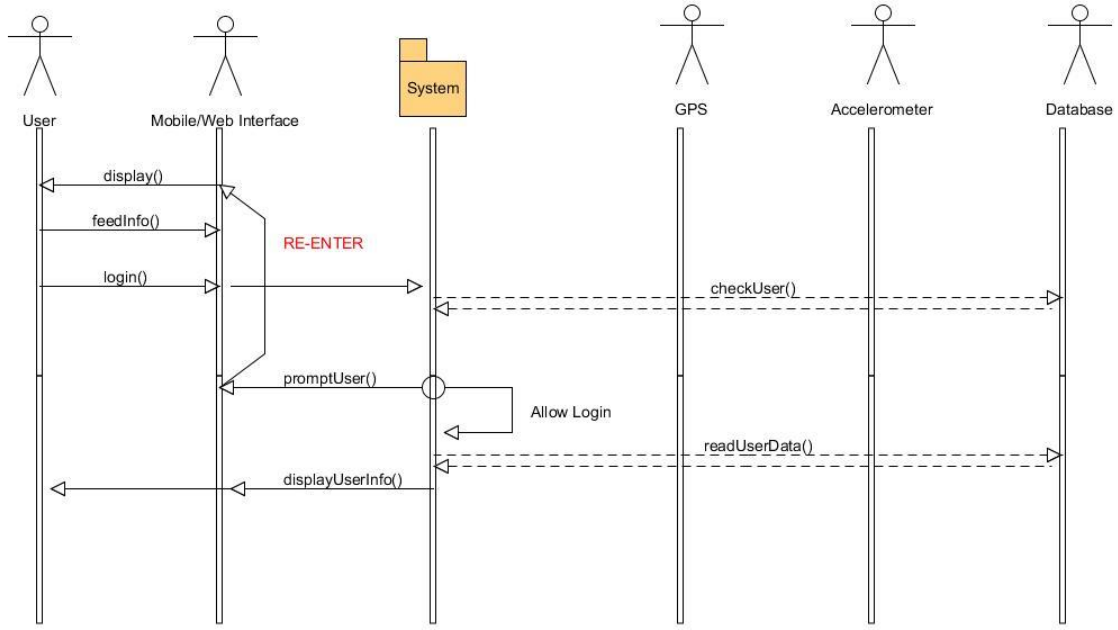


Fig 6.1:Login interaction diagram

The diagram above shows how the system reacts to the users inputs. Here only the login case has been discussed. The web/mobile interface displays the corresponding page, and gets the user interaction. In this case we limit the user's interaction to login case. Once the user enters information, the system checks with the database and verifies that the user name and password are valid. The system then has two responses; first, when the information is true the login is valid and the state of the actor changes. The system fetches the user information and data (if available) to display on the profile page. Second, the system prompts the user if the credentials are not valid to re-input the information.

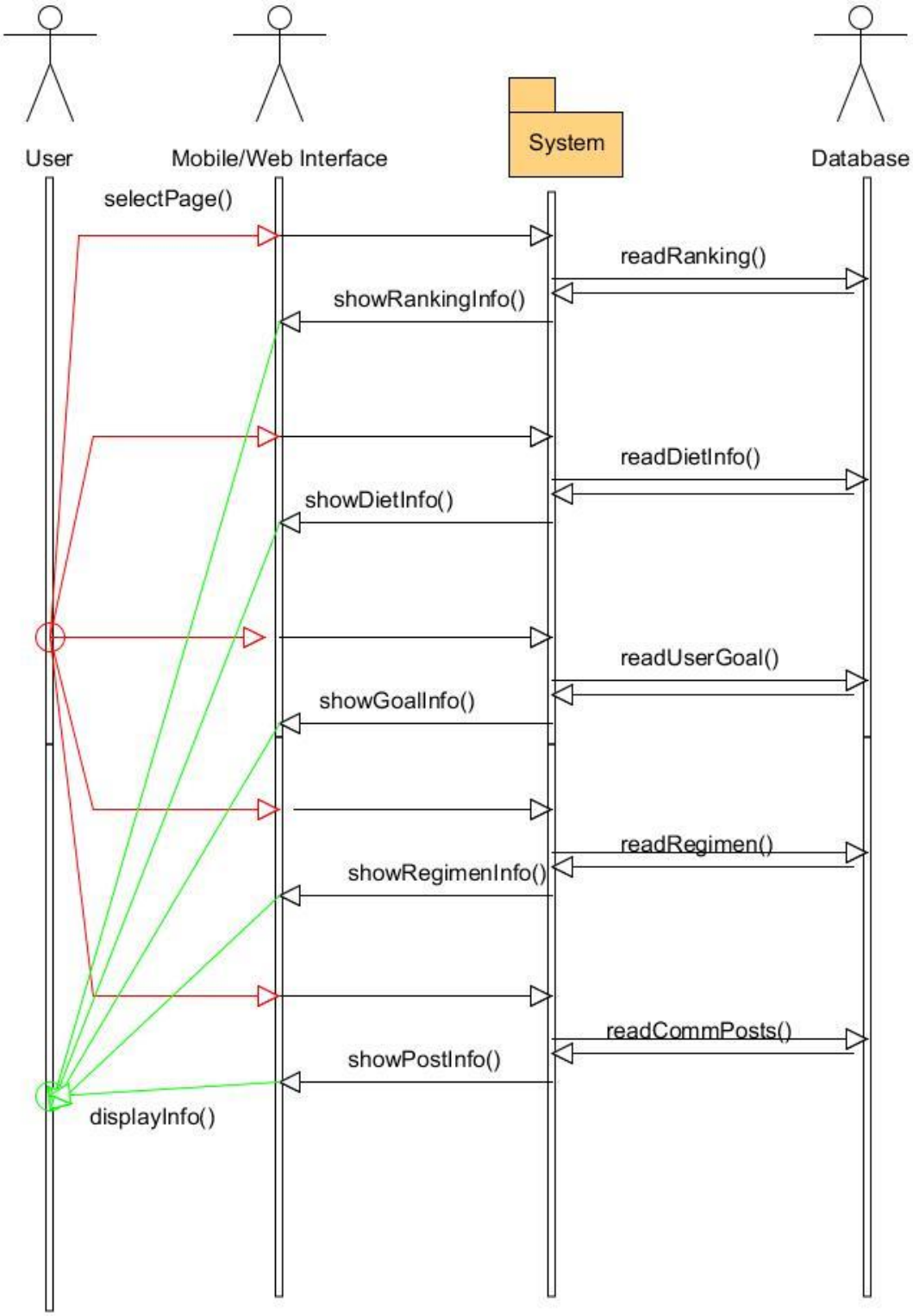


Fig 6.2 Navigation interaction Diagram

The navigation interaction diagram as shown above in figure 6.2 is the system's response to user navigating on the website. here the user may choose to jump from

one page to another by clicking on one of the tabs on the web interface. since each page is maintaining data on the database(web server).A click on the tabs takes the user to the other page where the information displayed is fetched from the database.

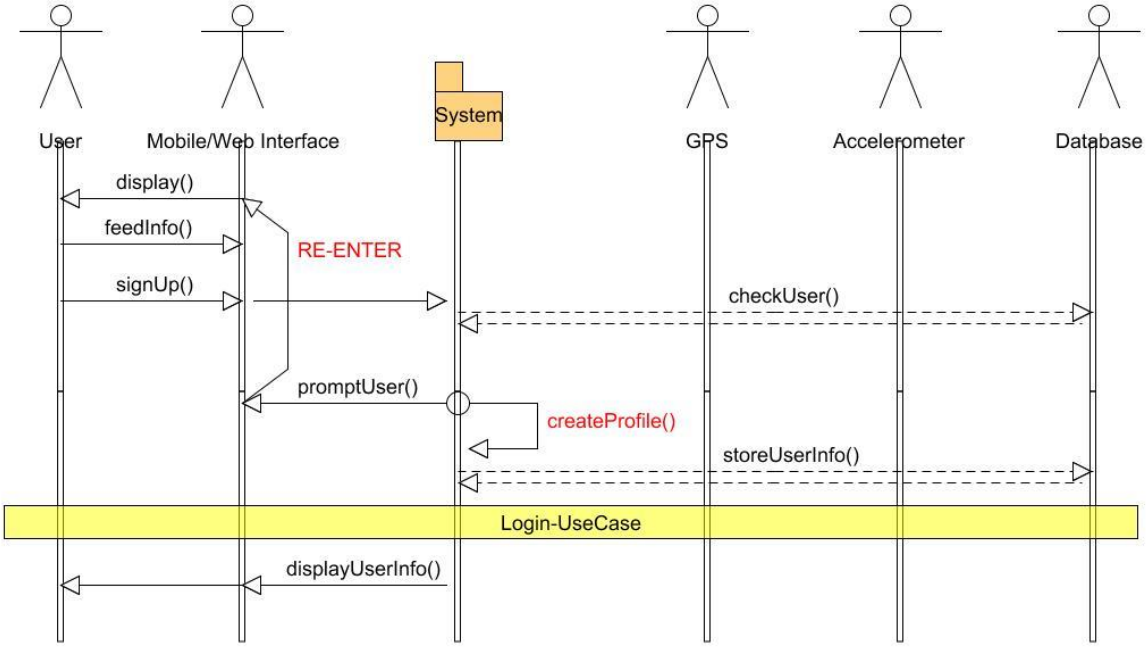


Fig 6.3: Sign up interaction Diagram

The figure above displays the how the sign up functionality will be implemented, after the user inputs the data, the systems first confirms if the user already exists in the system or not. If yes then user is prompted that the account already exists, and takes the user to the login page. This takes the user to the login use case. If no then the user is registered and a profile is created, this information is then stored on the database. Once the profile is created the user can see the login page, where he can login.

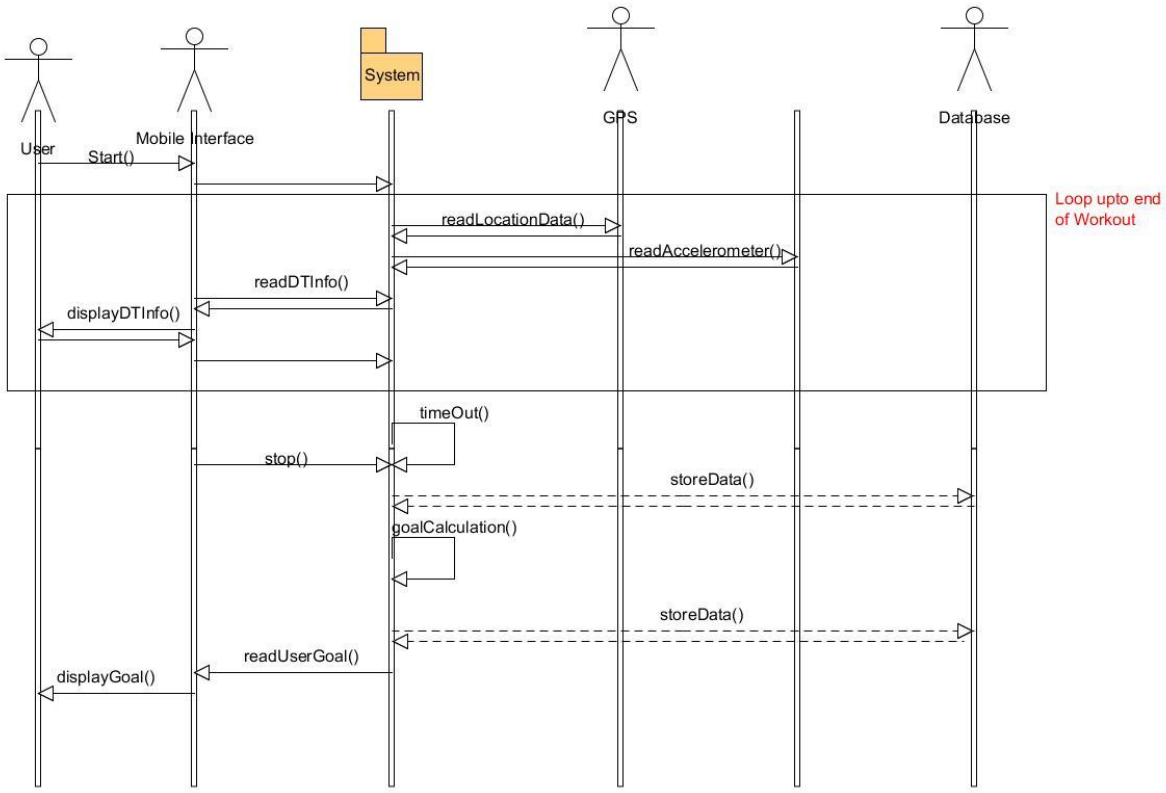


Fig 6.4: Start and Stop interaction diagram

This diagram shows the interaction between the various components of the system. Here, once the user starts the workout the mobile platform interacts with the GPS and accelerometer and computes the distance and speed. This data is displayed to the user in real time so that he can see. This is run on a thread which is continues till the user selects the stop button or the system produces a timeout condition. In either case the goal calculation is called which calculates and records the progress in the database.

6.1.2 Design Patterns: Theoretical Discussion

While developing this project we did not consider explicit implementation of design patterns in our code. However, having gone through discussions of this subject we have realized that there have been some specific places in the code where these strategies have been unknowingly implemented. We are still trying to figure out how to parse the information that is transferred between the phone and the website. The data acquisition has been implemented already through a file <gmail.php>. This was acquired as a legacy code from an open source website and with a few changes in certain places has been successfully implemented as an individual component function using a wrapper.

Another design pattern is used in the implementation of the login function. The implementation can somehow be related to a modified publisher-subscriber kind of design strategy where the member section pages are subscribed to the login page to verify that the user has already logged in and based on this information they serve the user member specific content. The business logic is restricted to the specific pages and has nothing to do with the login page.

Object-oriented design patterns typically show relationships and interactions between classes or objects, without specifying the final application classes or objects that are involved. For the classes on website platform, they are mostly designed with command pattern.

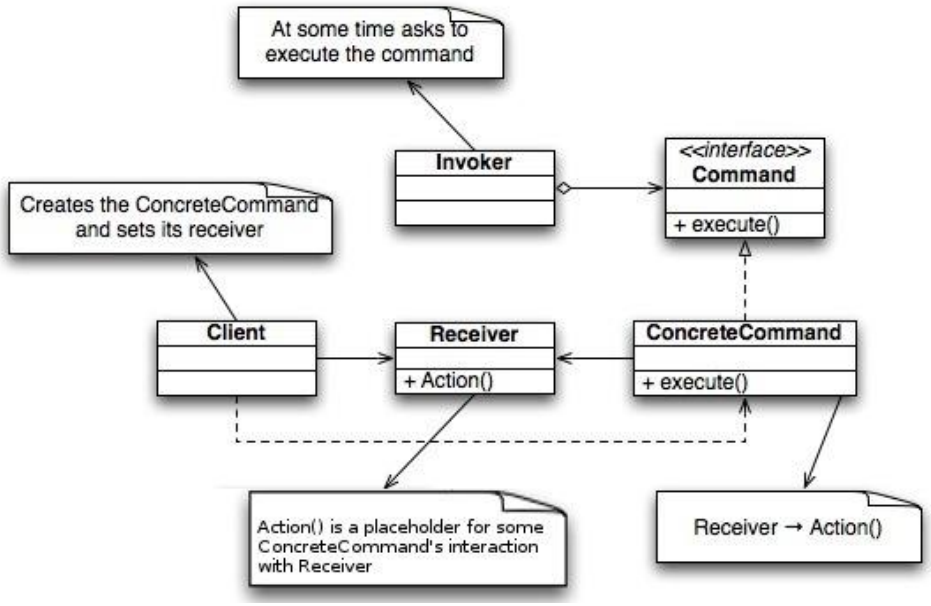


Fig. 6.5 Command Design Pattern Class Diagram

As we can see, class Progress, Checkin and Alert are all of functions as Invokers and Command. On the website once the user click on buttons of user interface, the classes would take responsibilities from each other. It implies that the classes know what the user to do next.
The Android classes are more like in state pattern.

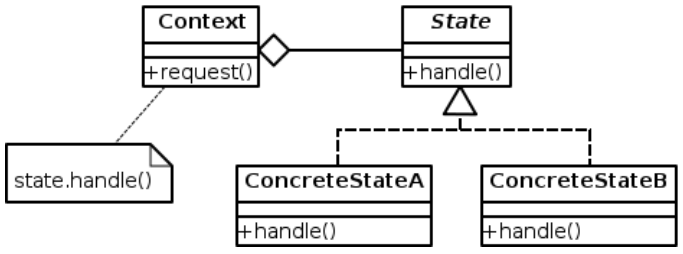


Fig. 6.6 State Design Pattern Class Diagram

The State pattern externalizes the relevant attributes into a State object, and this State object has the responsibility of managing the state transitions of the original object. A monolithic object's behavior is a function of its state, and it must change its behavior at run-time depending on that state. So, we used it when the application in various pages.

6.2 ALTERNATE SCENARIOS

At the beginning of the system design, we considered a lot of options. Figure 1 is our system architecture design applied at present. It describes the main components of the system. However, when we look back at the initial decisions, there were mainly three other alternative solution.

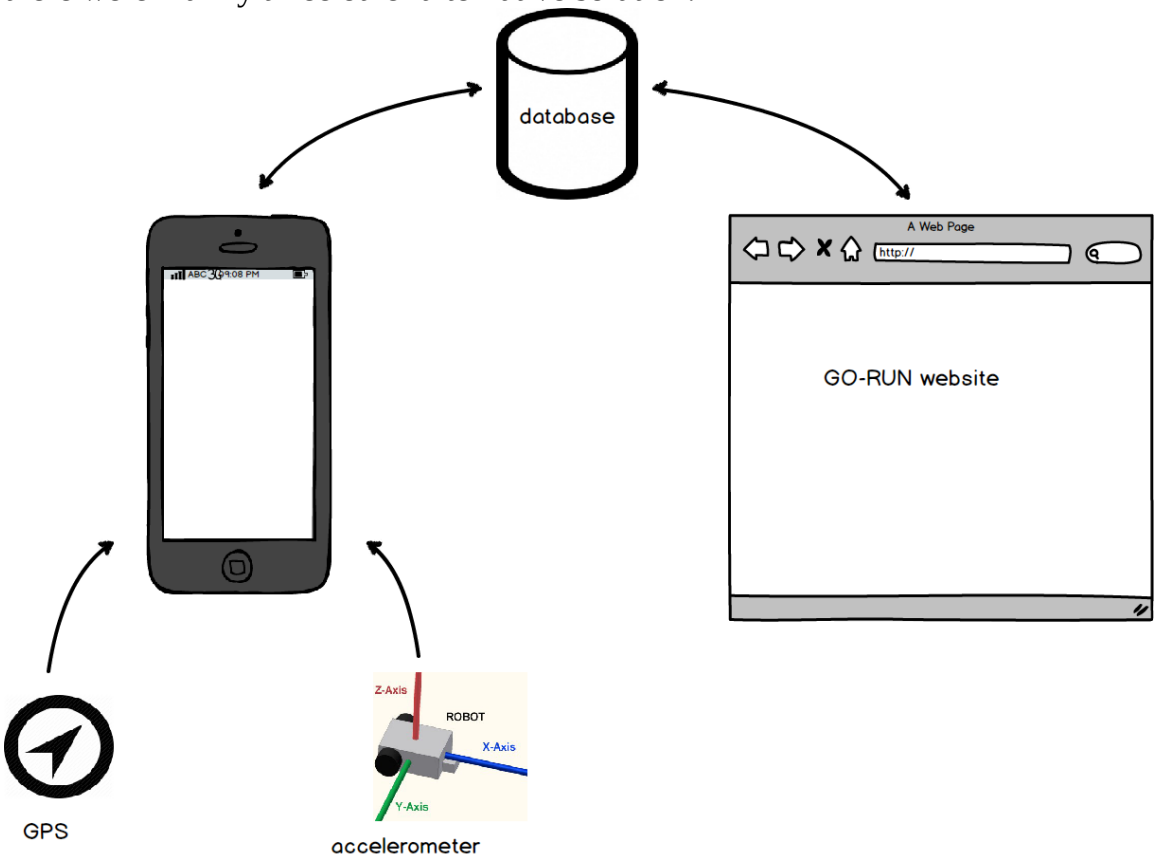


Fig 6.7: Overview of the current system

6.2.1. Third Party Device

The first one was that in which GPS and accelerometer are replaced by third party devices like Nike+ FuelBand and so on (Figure 2). The benefit is that third party devices are always providing accurate statics in various field. Due to additional hardware compared to cellphone, third party wearable devices offer more functions such as tracking calories burned and recording heart rate, features those are difficult for Android phones to approach. But most of third party devices are expensive, and smart phone with GPS and accelerometer is able to implement fitness goals.

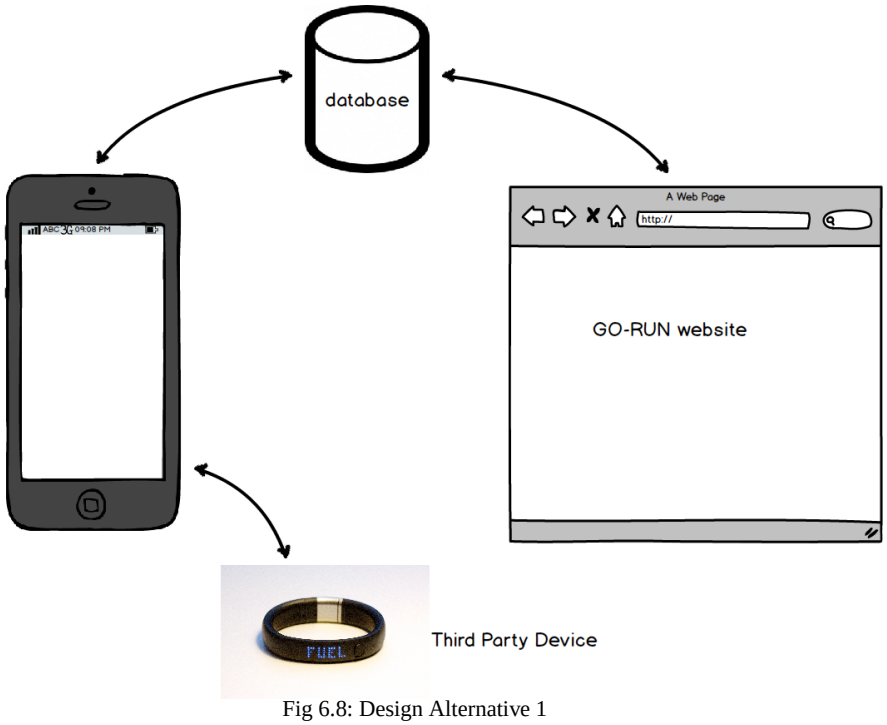


Fig 6.8: Design Alternative 1

Figure 3 shows how the third party device works for the system. It collects every catalog of fitness, including steps calculation, heart rate and so on, then sends them to the system.

Use Case: Start Workout/End Workout (UC4/UC5)

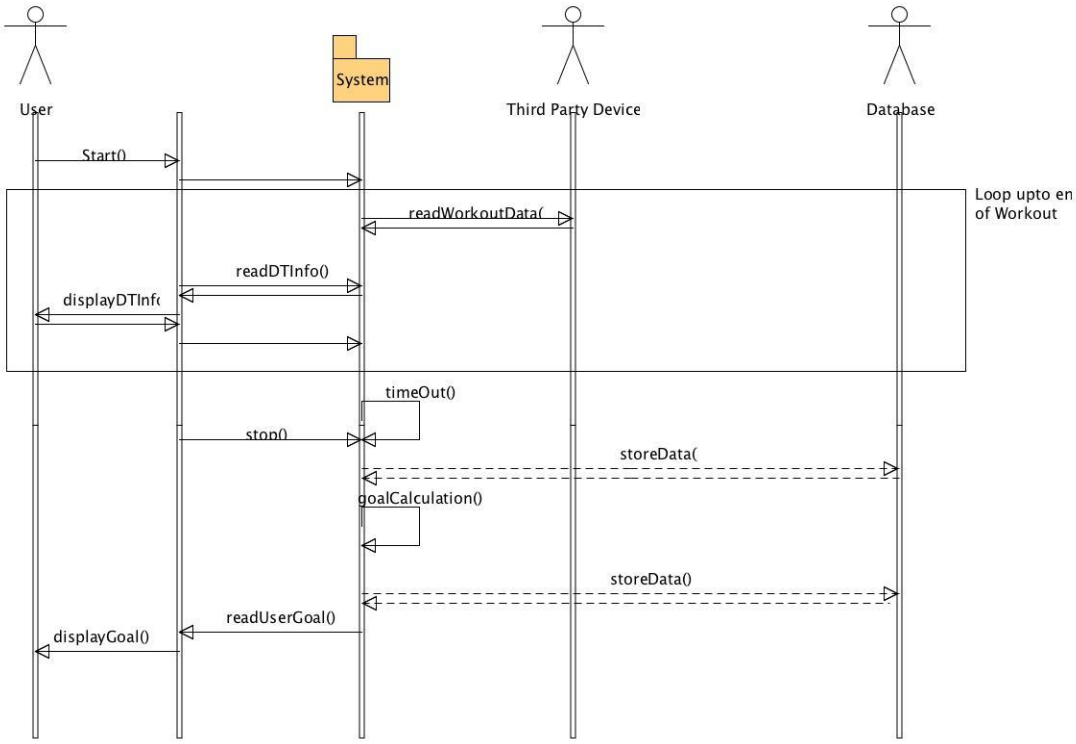


Fig 6.9: System Sequence Diagram for design alternative 1

6.2.2. Only GPS no accelerometer

We also put GPS inside phone as one of candidates. Since nearly every cell phone in the market has GPS inside, and Google Maps owns a powerful API improvement for application developers, we could completely depend on this thing. Till this decision, we still did not consider accelerometer. Interaction Diagram for this solution is presented in Figure 4, in which the system only needs to read location data from GPS.

Use Case: Start Workout/End Workout (UC4/UC5)

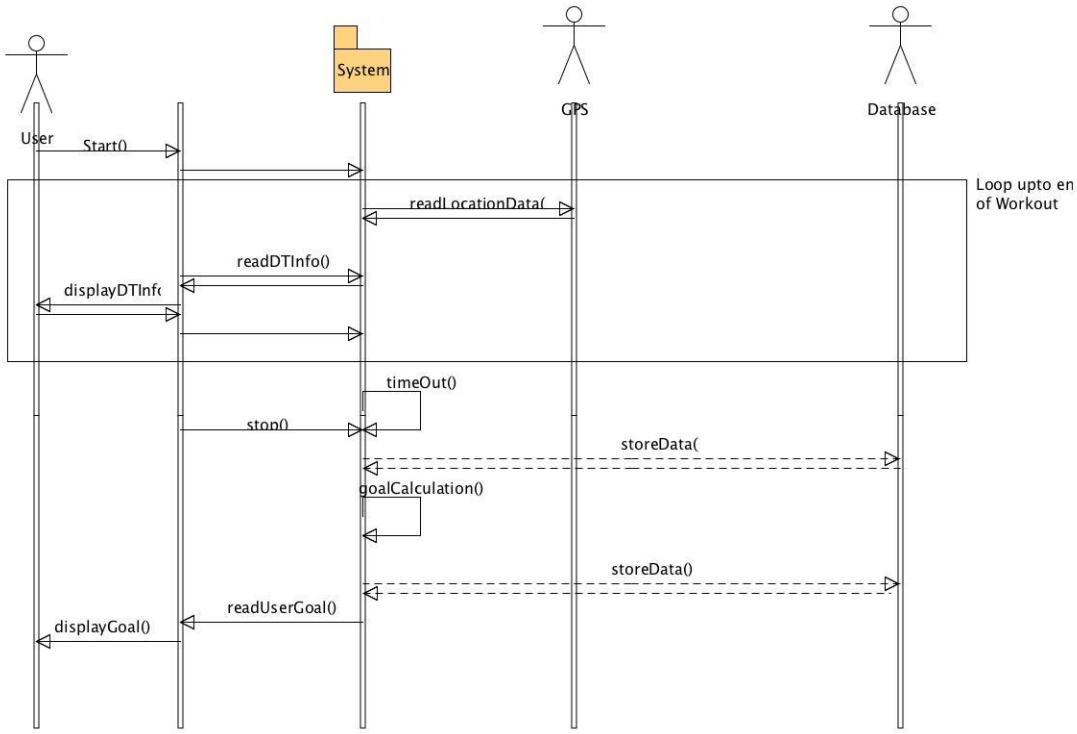


Fig 6.10: System Sequence Diagram for Design Alternate 2

6.2.3 Not use database for storing data

In addition to diagrams stated above, we had other alternative solutions while designing use cases including Use Case: Login (UC2) & Navigation to information pages (UC14) & Sign-Up (UC1).

In fact, there are many other ways that can be implemented in data storage when design a website. One of them is File-Based Databases, which is the easiest way. Website creates a text document on server then writes data into it. It's obviously easy and convenient.

Use Case: Login (UC2)

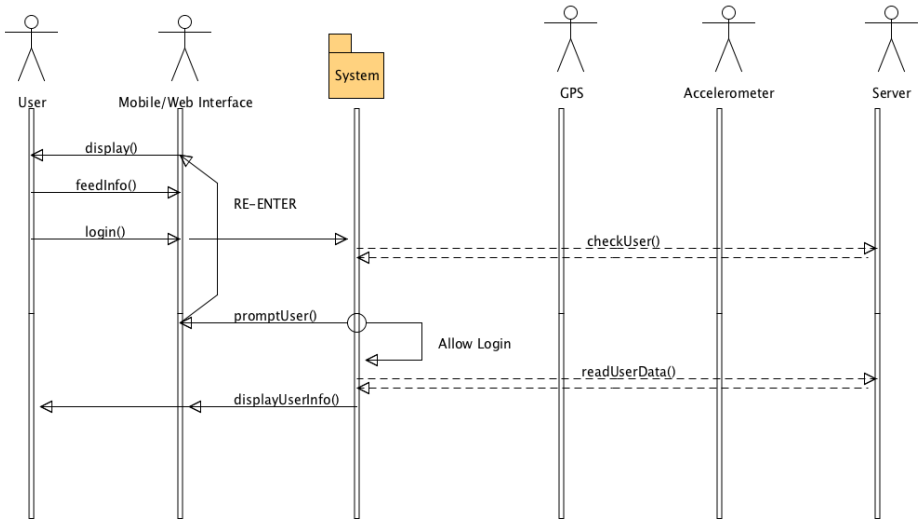


Fig 6.11: System Sequence Diagram for Alternate Design Choice 3

Navigation to information pages (UC14)

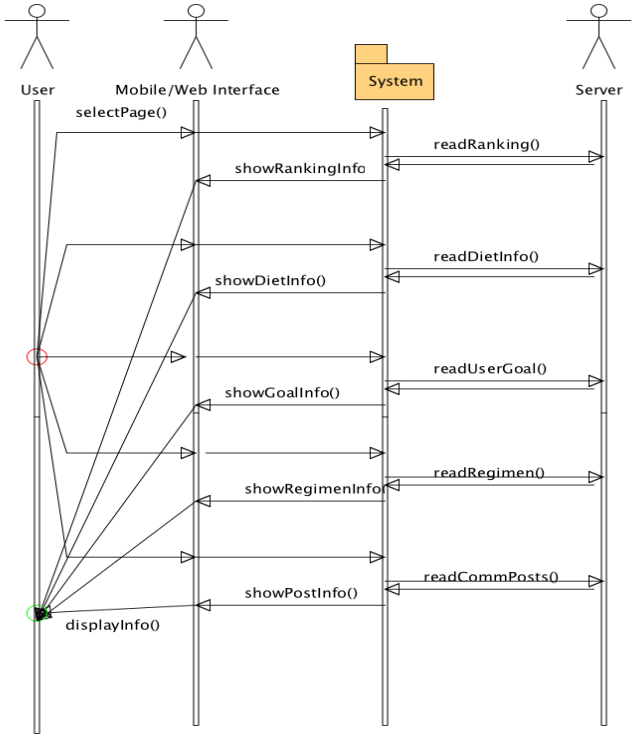


Fig 6.12: System Sequence Diagram for alternate design Choice 3

Sign-Up (UC1)

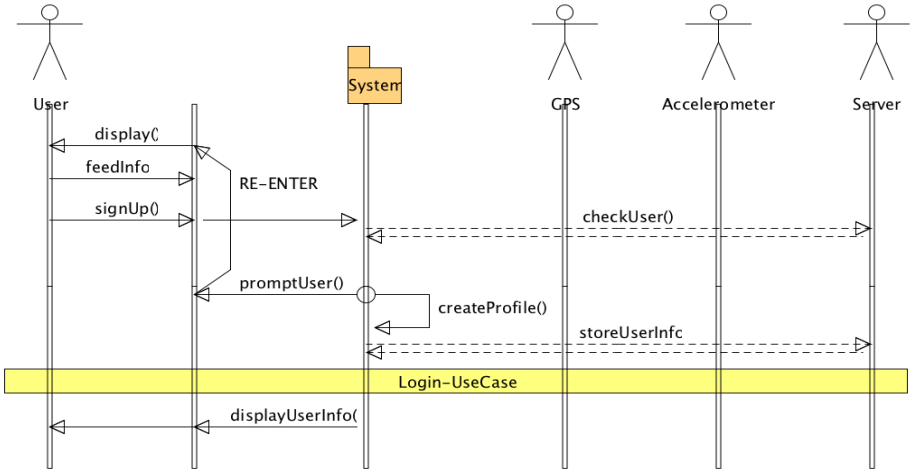


Fig 6.13: System Sequence Diagram for alternate design choice 3

6.2.4 Conclusion

The third party device solution provided high performance and maybe better user experience. And independent GPS solution seemed to be a reasonable solution. However, we abandoned these alternative choices as following reasons.

First, for a third party device, it is an isolated device from a cell phone. Then it is assigned with both Type 1 responsibility and Type 2 responsibility. Let's employ Low Coupling Principle. Obviously, a third party device involves too much communication with a cell phone. Besides, the expense for these gears is also inapplicable.

Second, a built-in accelerometer would be an outstanding assistant to GPS. When GPS loses Internet connection, we could use data collected from accelerometer to maintain the data collection temporarily. According to Expert Doer Principle, we should assign the responsibility to one who knows the task better and accelerometer is our choice. It is a perfect partner with GPS.

Third, file-Based Databases cannot provide safety of data, since users can easily access and download text document created. In addition, it's not efficient to manage data, because it is written without order and structure so that it costs a lot for data searching and operation.

7. CLASS DIAGRAM AND INTERFACE SPECIFICATION

7.1 CLASS DIAGRAMS – Android Platform

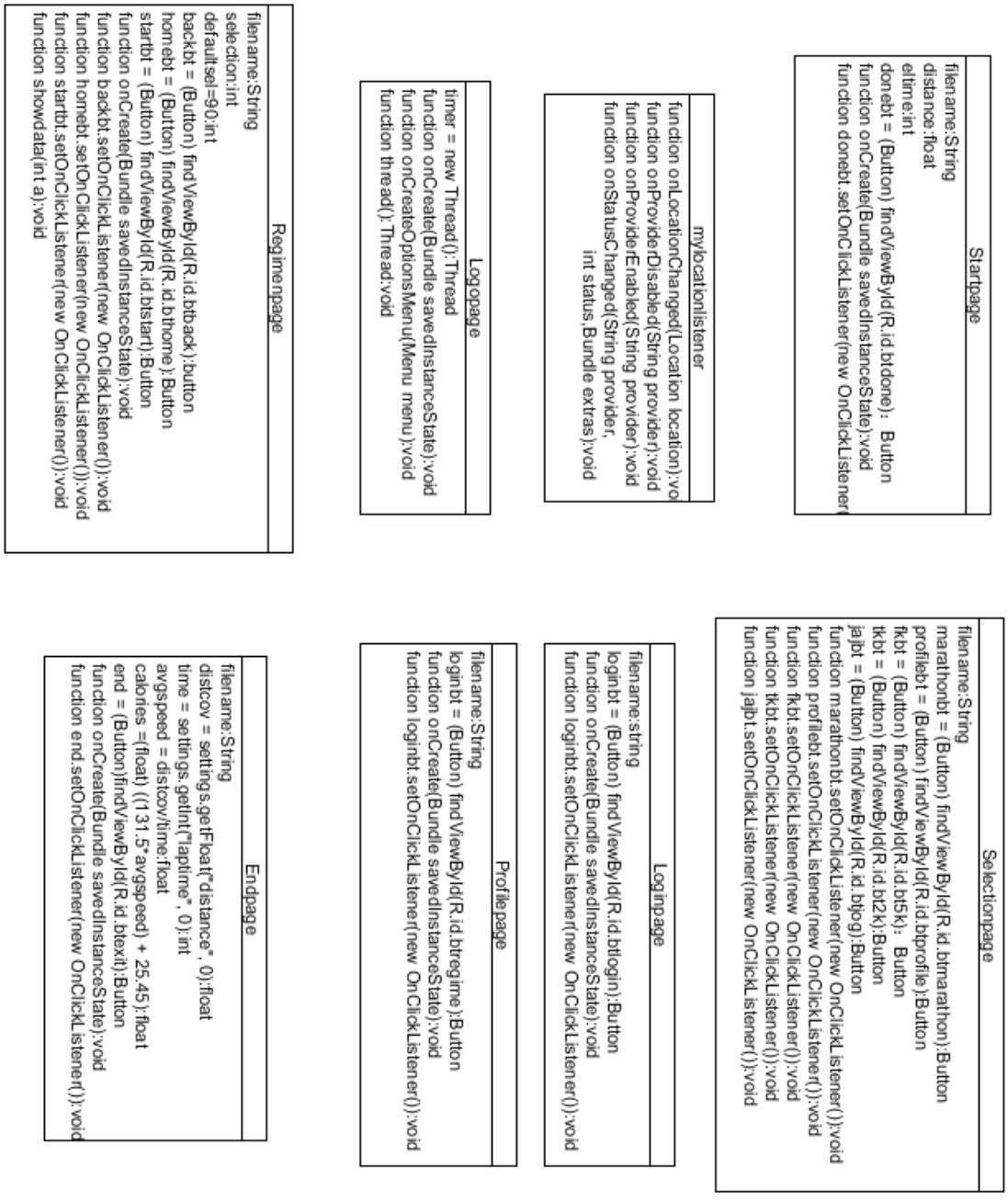


Figure 7.1: Overview of classes in the android app

7.2 DATA TYPES AND OPERATION SIGNATURES

7.2.1 Start-page

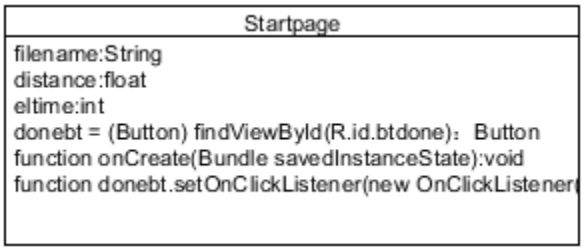


Figure 7.2

Variable	Description
Filename: string	
Distance: float	Distance of workout
Eltime: int	Time of workout

Functions/Type	Description
onCreate(Bundle savedInstancestate) : void:	This function means a window is being created.
donebt.setOnClicklistener(new OnClickListener())	This function is used to monitor whether the done button is touched or not.
Donebt=(Button) findViewByld(R.id.btdone):Button	Finish Workout

Table 7.1

7.2.2 mylocationlistener

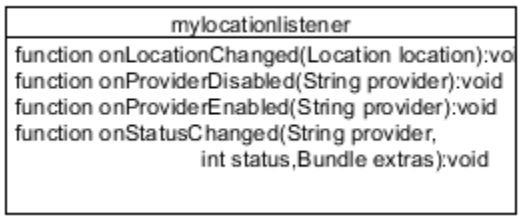


Figure 7.3

Function	Description
onLocationChanged(Location location):	This function works when system received new location data.
onProviderDisabled(String provider):	This function works when provider is disabled.
onProviderEnabled(String provider):	This function works when provider is enabled.
onStatusChanged(String provider, int stautus, Bundle extras):	This function works when status changed.

Table 7.2

7.2.3 Logopage

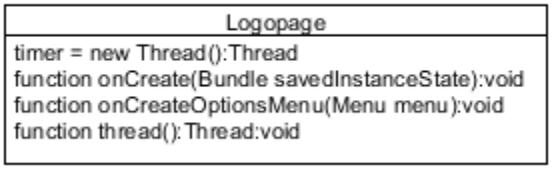


Figure 7.4

Variable	Description
Timer=new Thread():Thread	
Function	**Description**
onCreate(Bundle savedInstanceState):	This function means a window is being created.
OncreateOptionsMenu(Menu menu):	This function is going to provide a option menu.
Thread():Thread	

Table 7.3

7.2.4 Regimenpage

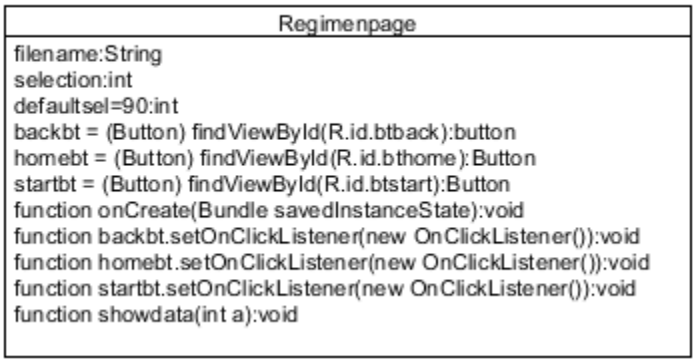


Figure: 7.5

Variable	Description
Filename:string	
Selection:int	Regimen that user is following
Defaultsel=90:int	
Backbt=(Button)findViewByld(R.id.btback):button	Button to previous page
Homebt=(Button)findViewByld(R.id.bthome):button	Button to home page
Startbt=(Button)findViewByld(R.id.btstart):Button	Button to start

Function	Description
onCreate(Bundle savedInstanceState):	This function means a window is being created.
backbt.setOnClickListener(new OnClickListener()):	This function is used to monitor whether the backbt button is touched or not.
homebt.setOnClickListener(new OnClickListener()):	This function is used to monitor whether the homebt button is touched or not.
startbt. setOnClickListener(new OnClickListener()):	This function is used to monitor whether the startbt button is touched or not.
showdata(int a):	This function is going to demonstrate data on screen.

Table 7.4

7.2.5 Selectionpage

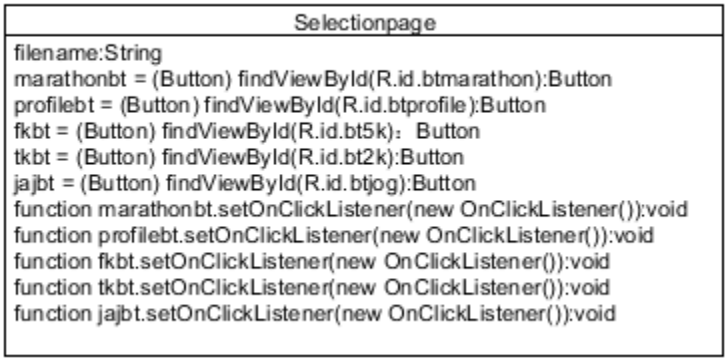


Figure 7.6

Variable	Description
Filename:string	
Marathonbt=(Button)findViewByld(R.id.btback):button:	Button to follow marathon.
profile=(Button)findViewByld(R.id.btback):button:	Button to view profile.
Fkbt=(Button)findViewByld(R.id.btback):button:	Button to follow 5K run.
tkbt=(Button)findViewByld(R.id.btback):button	Button to follow 10K run.
jajbt=(Button)findViewByld(R.id.btback):button	Button to follow jogging.

Functions	Description
marathonbt.setOnClickListener(new OnClickListener())	This function is used to monitor whether the marathon button is touched or not.
profilebt.setOnClickListener(new OnClickListener()):	This function is used to monitor whether the profile button is touched or not.
fkbt.setOnClickListener(new OnClickListener()):	This function is used to monitor whether the fkbt button is touched or not.
tkbt.setOnClickListener(new OnClickListener())	This function is used to monitor whether the tkbt button is touched or not.
jajbt.setOnClickListener(new OnClickListener())	This function is used to monitor whether the jajbt button is touched or not.

Table 7.5

7.2.6 Loginpage

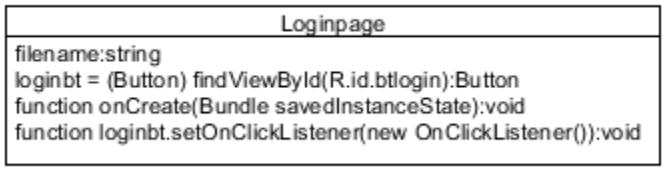


Figure 7.7

Variable	Description
Filename:string	
Loginbt=(Button)findViewByld(R.id.btlogin):Button	Button to login to system.
Functions	**Description**
onCreate(Bundle savedInstanceState):	This function means a window is being cr
loginbt.setOnClickListener(new OnClickListener()):	This function is used to monitor whether t button is touched or not.

Table 7.6

7.2.7 Profilepage

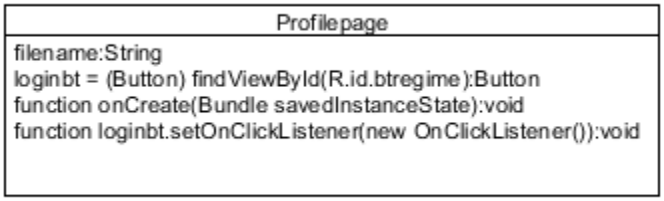


Figure 7.7

Variable	Description
Filename:string	
Loginbt=(Button)findViewByld(R.id.btlogin):Button	Button to login to system

Functions	Description
onCreate(Bundle savedInstanceState):	This function means a window is being created.
loginbt.setOnClickListener(new OnClickListener())	This function is used to monitor whether the login button is touched or not.

Table 7.7

7.2.8 Endpage

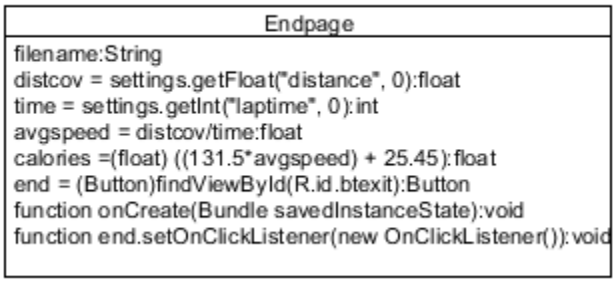


Figure 7.9

Variable	Description
Distcov=settings.getFloat(“distance”,0):float	Total distance for workout
Time=settings.getint(“laptime”,0):int	Total time for workout
Avgspeed=distcov/time:float	Average speed for workout
Calories=(float)((131.5*avgspeed)+25.45):float	Calories burned during workout
End=(button)findViewByld(R.id.btexit):Button	Button to exit application

Functions	Description
onCreate(Bundle savedInstanceState):	This function means a window is being created.
end.setOnClickListener(new OnClickListener()):	This function is used to monitor whether the end button is touched or not.

Table 7.8

7.3 CLASSES IN THE WEBSITE

Till now the implementation of the website classes has only been through basic HTML and JavaScript. The concepts for individual pages, however, can be implemented through any other language like PHP, Ruby on Rails, Python etc. Even if the current language of implementation is not object oriented, the concept for the features developed can be explained in a class diagram paradigm and can thus be implemented in any language desired. The representative class diagrams and functions for some important pages may be described through the following diagrams and discussions.

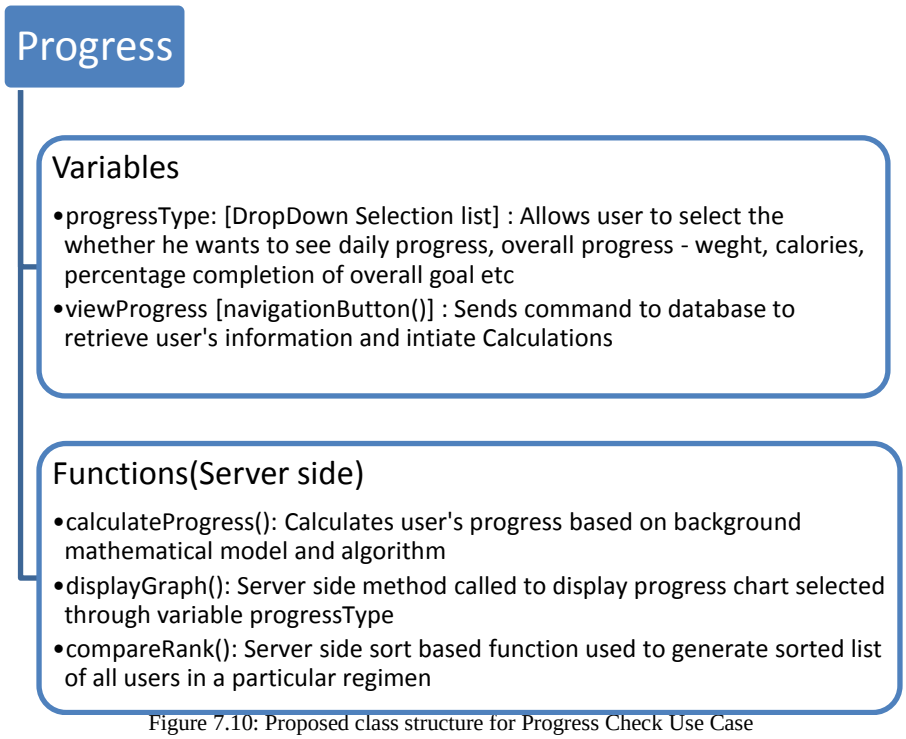


Figure 7.10: Proposed class structure for Progress Check Use Case

The above diagram tries to capture the domain concept check progress and its possible implementation strategy through an object oriented platform. The variables in this class can be compared to the data type <Enum> in C++ which allows the user to select an option out of the many provided and give it as an input to a function thereafter where a branching strategy is implemented. The functions are implementations of algorithms that have been derived from mathematical models discussed in previous sections of the report. It must be noted that the Use Case Check Progress (UC12) translates to multiple domain concepts including but not limited to readUserInfo, readUserGoals, readStatistics etc. It might appear that these have eventually been implemented through a restricted number of functions which to the reader might seem unjust. However, it must be kept in mind that this is only a representative class and the actual functions while implementation would be more in number and more specific in operation. Smaller functions may or may not be devised based on the language of implementation.

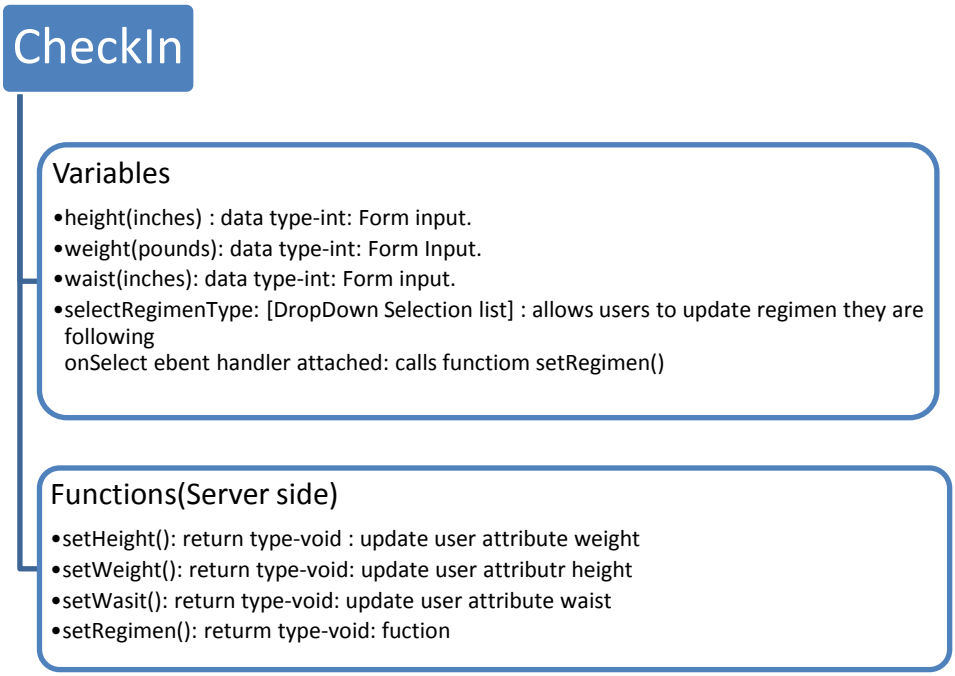


Figure 7.11: A proposed implementation of the domain concept storeUserInfo

The use case Check-In only translates to a single domain concept storeUserInfo. This diagram thus covers most of the functions and variables that are needed to implement the Use Case and the domain concept for Check-In class. All the functions used are set-type which are needed to indirectly update database values. Normal security functions like not providing direct access to class variables and input data type check etc. need to be followed in order to prevent accidental overwrites and non-compatible data entries based on data types.

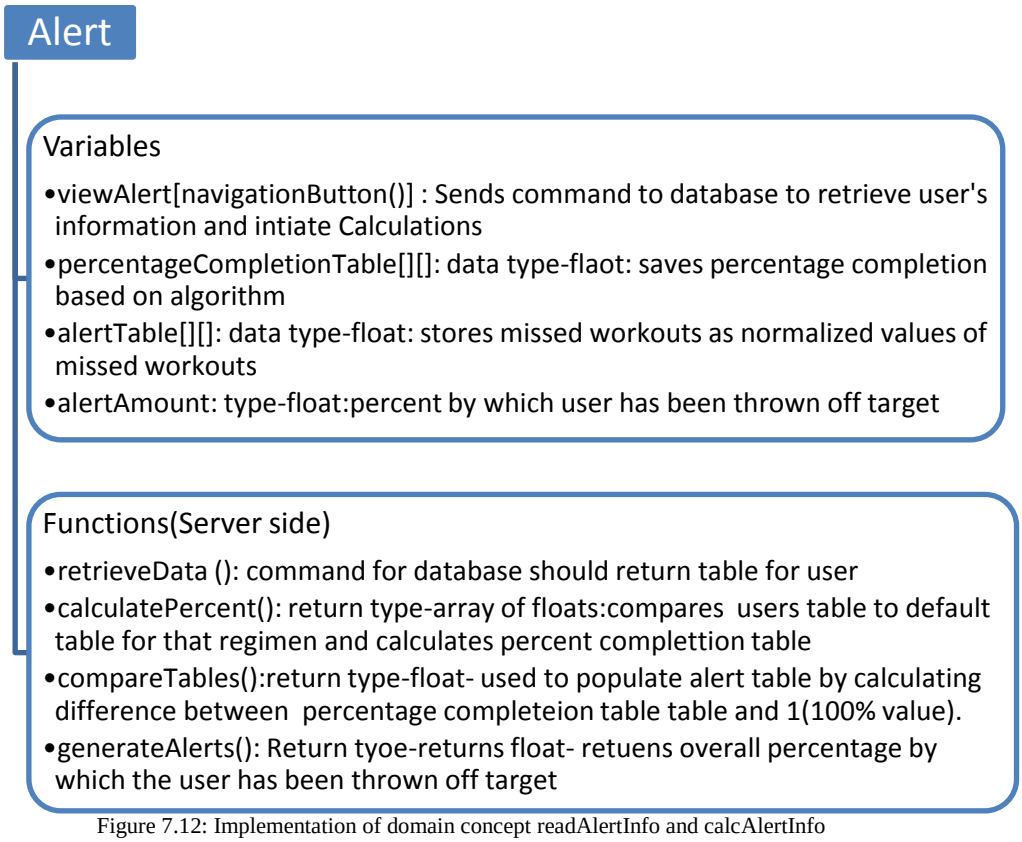


Figure 7.12: Implementation of domain concept readAlertInfo and calcAlertInfo

The use case check alert translates to the domain concept pair calcAlertInfo and readAlertInfo. This is an implementation that is mostly standalone except for the database access. In case the database does not contain any relevant values the functions would return 100% values in the alert table. This is an ongoing problem with devising implementation of this algorithm that needs to be tackled presently and the team is working on this feature right now.

7.4 TRACEABILITY MATRIX

Table 7.1 indicates the relationship between domain concepts mentioned in the first report and software class. As we can see, in the table, there's one use case that is involved in almost all classes, the Navigate to information pages. It is because this use case is an interface for multiple functions. In absence of this function other classes cannot link with these function.

And for another two use cases we list, they evolve to two or more classes. One reason for such phenomenon is that domain concept is too complex to deal it with only one class. For example, as we deal with the use case of "start workout", we should get the user's GPS location, start time, goal for the training in the regimen and also we should make the progress updated to user's information. However, when it comes to the class-determination part, we found it difficult to finish such a complex case with only one class. So we make two involved in the whole process: Startpage, mylocationlistener. Startpage class is used for storing all kinds of information used for training program starting, such as the start time of the training, distance calculate. Mylocationlistener class is made for providing location information.

And also, we can see there are a few concepts belong to one class. It is because that too many classes will make the program bloat, so we unite some of them. But for mylocationlistener class, although it can be united into the Startpage class, we did not make it for the reason that Startpage class will calculate the distance user has done, and there will be a huge amount of location data. So it will take additional effort for a united Startpage class to refreshing these location data. So we made them independent.

use case/class	Startpage	mylocationlistener	Selectionpage	Regimenpage	Profilepage	Logopage	Loginpage	Endpage
Sign-Up						X		
Login							X	
Select Regimen			X					
Start Workout	X	X						
Stop/ End Workout	X	X						
Exit Application								X
View Profile Page					X			
View Dietary Information				X				
View Schedule				X				
View Home					X			
Provide Feedback					X			
View Progress	X							
View People & GPS Current location		X						

Figure 7.13: Traceability matric for Android system

7.5 OCL Contract Specification

Start-page	
Invariant	"Finish" button will be displayed.
Pre-condition	User touched "Start" button.
Post-condition	User can start his/her workout. Time and location data will be stored.

myLocationListener	
Invariant	Collecting location data.
Pre-condition	User touched "Start" button.
Post-condition	Location data will be consecutively changed and collected.

logoPage	
Invariant	The system will provide a picture of the logo of our project.
Pre-condition	User started our application.
Post-condition	User will go to Loginpage.

regimenPage	
Invariant	The regimen user is now following and a "Start" button will be displayed.
Pre-condition	User selected his/her regimen.
Post-condition	User can choose whatever regimen he/she wants to follow and touches corresponding button to start his/her workout. User will go to Start-page.

selectionPage	
Invariant	Four different regimens will be provided
Pre-condition	User logged into our system.
Post-condition	User will go to Regimenpage to start his/her workout.

loginPage	
Invariant	User can sign in and sign up.
Pre-condition	User viewed Logopage and touched screen.
Post-condition	User will go to Selectionpage to view regimens those are provided.

Profilepage	
Invariant	Location data will not change anymore. Distance, time and speed of this workout will be displayed on screen. “End” button will be provided.
Pre-condition	User touched “Finish” button.
Post-condition	User will go to Endpage.

Endpage	
Invariant	System will ask for email address that will be used to send data.
Pre-condition	User touched “End” button.
Post-condition	An email will be send to the email address that system asked for. Detailed information can be viewed on our website.

SECTION 8: SYSTEM ARCHITECTURE AND SYSTEM DESIGN

8.1 ARCHITECTURAL STYLE

During years of software development, software designers conclude several patterns and structures that appear frequently. And they use these patterns view a style from a large-scale of system. This is origin of software architectural style. An architectural style decides how components are organized, how data is manipulated or how components communicate and so on.[1]

Accordingly, the Go-run system is client/server style [2], since the web application and then phone application are both graphical UI application that communicated with a database server containing much of the fitness logic in the form of stored procedures. The client initiates one or more requests (to fetch work out instructions, view comments, etc). The server authorizes the user and then processes requests to generate the result. The client/server architectural style describes this kind of relationship between a client and one or more servers.

In addition, because all data is stored on the server, the client/server style brings Go-run project a high security solution. And it is easy for administrators to access, update and maintain the database.

8.2 IDENTIFYING SUBSYSTEMS

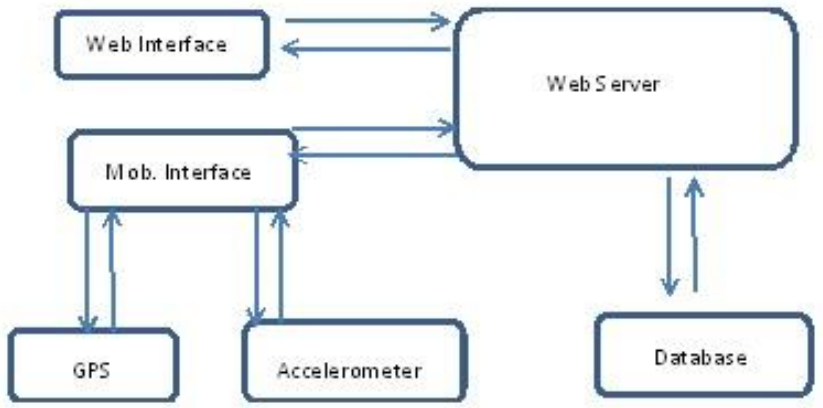


Figure 8.1 Systems Identification

Even though the UI and layout of the website and Android app is different, they are regarded as a user interface subsystem. The user interface subsystem interacts with the user to obtain has functions such as account settings, registration and select regimens. The website subsystem offers more detailed features like ranking and alert and has complex organization due to multi-user interactions. Users are able to communicate with each other on the forum. The Android app subsystem tracks user's activity and uploads the data to the database. The database subsystem stores all the information in every subsystem.

The whole system can be seen as seen in the figure 8.1. The various subsystems are GPS, Accelerometer, Web Interface, Mobile Interface, Server and the

Database. All of the subsystems have been defined in the above sections. The description here is only for the architectural aspect.

8.3 MAPPING SUBSYSTEMS TO HARDWARE

The hardware will be mapped to three sets as following.

- For the server subsystem, we use a personal PC as the server.
- For the website subsystem, it will be running on different user machines (such as web browser).
- For the Android app subsystem, it will be running on different user's smart phones.
- GPS is used for the fetching for the location which is a hardware module interacting with the mobile interface.
- Accelerometer is the hardware which is used to determine the acceleration of the user and back track it to the users speed and distance.

8.4 PERSISTENT DATA STORAGE

Considering that Android system could not get access to MySQL, our database system is built on SQLite, which is an extremely featured, convenient as well as simple database. In Go-run, considering the number of users, database is required to store all of them and other information. The data would be separated into four parts which are User, Regimen, Training Data (Overall), Training Data (daily). User contains of information about user, such as age, weight, height, etc. Regimen will provide detailed schedule, which could help user to follow the plan and reach his/her target.

Training Data (daily) is the monitoring data that a user achieved in a natural day, and the Training Data (daily) will be erased every day in case of problems. Training Data (Overall), which are based on Training Data (daily), will demonstrate the achievement that user made in overall perspective.

Every user will get storage for all the regimens; however the website will provide analytics only for the regimen he has selected. If all the days of each week in every regimen were to me stored we would require 224 columns just for progress data storage. However to overcome this problem we came up with a BigInt based algorithm that implements two decimal number bits for each day in the week reducing the progress data storage columns to 35. Discussion on this can be found in the technical documentation of the project.

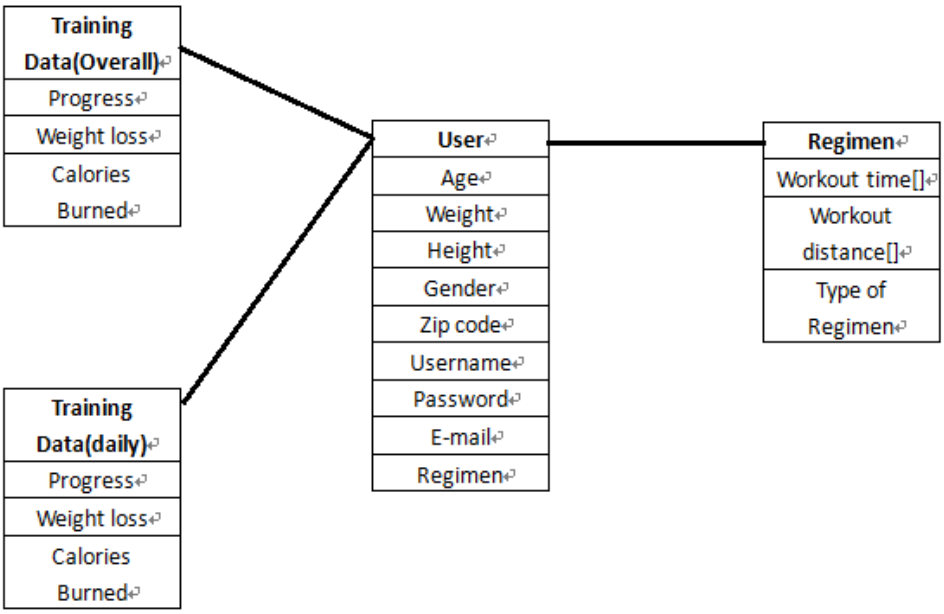
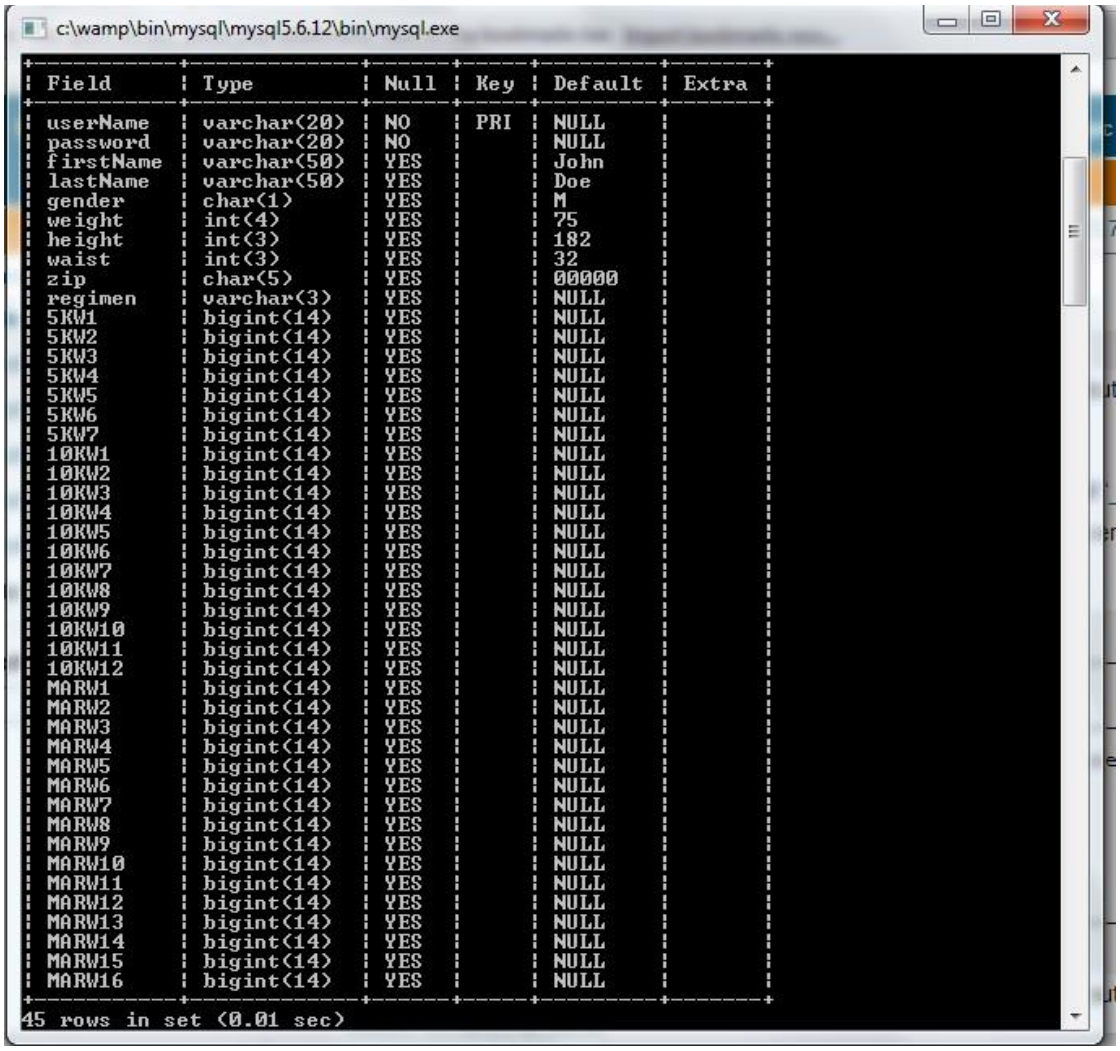


Figure 8.2 Data Storage



c:\wamp\bin\mysql\mysql5.6.12\bin\mysql.exe

Field	Type	Null	Key	Default	Extra
userName	varchar(20)	NO	PRI	NULL	
password	varchar(20)	NO		NULL	
firstName	varchar(50)	YES		John	
lastName	varchar(50)	YES		Doe	
gender	char(1)	YES		M	
weight	int(4)	YES		75	
height	int(3)	YES		182	
waist	int(3)	YES		32	
zip	char(5)	YES		00000	
regimen	varchar(3)	YES		NULL	
5KW1	bigint(14)	YES		NULL	
5KW2	bigint(14)	YES		NULL	
5KW3	bigint(14)	YES		NULL	
5KW4	bigint(14)	YES		NULL	
5KW5	bigint(14)	YES		NULL	
5KW6	bigint(14)	YES		NULL	
5KW7	bigint(14)	YES		NULL	
10KW1	bigint(14)	YES		NULL	
10KW2	bigint(14)	YES		NULL	
10KW3	bigint(14)	YES		NULL	
10KW4	bigint(14)	YES		NULL	
10KW5	bigint(14)	YES		NULL	
10KW6	bigint(14)	YES		NULL	
10KW7	bigint(14)	YES		NULL	
10KW8	bigint(14)	YES		NULL	
10KW9	bigint(14)	YES		NULL	
10KW10	bigint(14)	YES		NULL	
10KW11	bigint(14)	YES		NULL	
10KW12	bigint(14)	YES		NULL	
MARW1	bigint(14)	YES		NULL	
MARW2	bigint(14)	YES		NULL	
MARW3	bigint(14)	YES		NULL	
MARW4	bigint(14)	YES		NULL	
MARW5	bigint(14)	YES		NULL	
MARW6	bigint(14)	YES		NULL	
MARW7	bigint(14)	YES		NULL	
MARW8	bigint(14)	YES		NULL	
MARW9	bigint(14)	YES		NULL	
MARW10	bigint(14)	YES		NULL	
MARW11	bigint(14)	YES		NULL	
MARW12	bigint(14)	YES		NULL	
MARW13	bigint(14)	YES		NULL	
MARW14	bigint(14)	YES		NULL	
MARW15	bigint(14)	YES		NULL	
MARW16	bigint(14)	YES		NULL	

45 rows in set (0.01 sec)

Fig 8.3: Description of table Main in se_gorun database

8.5 NETWORK PROTOCOL

Go-run website uses HTTP to get access to server. Requests to access information in the databases (which are SQLite) complete by PHP. GPS data is derived from the NMEA 0183 protocol but is not implemented to its fullest.Accelerometer data is fetched from the smart phones inbound accelerometer which implements IPEN protocols. GPS and Accelerometer protocols are overridden by the OS on the smart phone.

8.6 GLOBAL CONTROL FLOW

Firstly, Go-run is an event driven system, which means every user can generate the actions in a different order. After log in, user can choose what regimen he/she wants to take. Once starts a training, user can stop and resume at any time. What's more, on website, the user can also initiatively check information about diet solutions, other users' information and ranking.

Secondly, Go-run is a real-time system, which mainly presents on the Android app. When the user starts to run, the app can track the user's location in a real-time through GPS and accelerometer.

8.7 HARDWARE REQUIREMENTS

The Go-run project needs a Smart phone with GPS and accelerometer that is based on the android operating system. Additionally, a client PC connected to the Internet is required to access the server and display statistics through the system website. From the developer's perspective, we would require a Windows/Linux based server capable of serving PHP/HTML pages and hosting a MySQL based database with InnoDB type of storage.

SECTION 9: ALGORITHMS AND DATA STRUCTURES

9.1 ALGORITHMS

9.1.1 Distance and Speed calculation

One of the most incredible characteristics of Go-run is that user could get access to his/her real time data which includes speed, time and distance. GPS sensor in the mobile phone will provides the longitude and latitude of user every second. Once we know the current location data and the data from 1 second ago, we could calculate the distance user ran in 1 second.

When we know the longitude and latitude of a place, which represented by (a, b) , we could calculate its coordinate (x, y, z) by following formulas:

$$x = \cos(a) * \cos(b)$$
$$y = \cos(a) * \sin(b)$$
$$z = \sin(a)$$

Assuming the current location's longitude and latitude is (a1, b1) where a1 stands for longitude and b1 stands for latitude. The previous location's longitude and latitude is (a0, b0) where a0 stands for longitude and b0 stands for latitude. By using the formulas above, we could get (x1, y1, z1) and (x0, y0, z0) which are two points on the unit circle.

As far as we know, the shape of earth is almost a sphere, and its average radius is 6371 kilometers. The distance between two (x1, y1, z1) and (x0, y0, z0) is

$$D = \sqrt{(x1 - x0)^2 + (y1 - y0)^2 + (z1 - z0)^2}$$

So, the distance in the real world could be represented like this(r is the average radius of the earth):

$$Distance = \frac{\pi * r * sin^{-1}(2 \sin\left(\frac{D}{2}\right) \cos\left(\frac{D}{2}\right))}{180}$$

Noticing that

$$Distance = Speed * Time$$

Speed equals distance when time is 1 second.

Therefore, we could display speed and time on the screen, and a counter will be used to calculate the overall distance until the workout ends.

9.1.2 BMI

The body mass index (BMI), or Quetelet index, is a measure for human body shape based on an individual's mass and height. Devised between 1830 and 1850 by the Belgian polymath Adolphe Quetelet during the course of developing "social physics", it is defined as the individual's body mass divided by the square of their height - with the value universally being given in units of kg/m^2.

Go-run is going to provide the function which could calculate your BMI which would show your body condition in an obvious way.

$$BMI = \frac{mass(kg)}{(height(m))^2} = \frac{mass(kg)}{(height(in))^2} * 703$$

9.1.3 Ranking

In the ranking system, we will provide the top 10 who did most efforts to achieve their target. Sorting may be the first idea that would come to our mind. In Go-run, considering the number of user would be large, an effective and efficient sorting algorithms is required.

Quicksort, or partition-exchange sort, is a sorting algorithm developed by Tony Hoare that, on average, makes O(n log n) comparisons to sort n items. In the worst case, it makes O(n^2) comparisons, though this behavior is rare. Quicksort is often faster in practice than other O(n log n) algorithms. Quicksort is a divide and conquer algorithm. Quicksort first divides a large list into two smaller sub-lists: the low elements and the high elements. Quicksort can then recursively sort the sub-lists.

The steps are:

Pick an element, called a pivot, from the list.

1. Reorder the list so that all elements with values less than the pivot come before the pivot, while all elements with values greater than the pivot come after it (equal values can go either way). After this partitioning, the pivot is in its final position. This is called the partition operation.
2. Recursively apply the above steps to the sub-list of elements with smaller values and separately the sub-list of elements with greater values.

The base case of the recursion is lists of size zero or one, which never need to be sorted.

9.1.4 Calorie Meter

The calories are calculated on the formula

$$Cal = 131.5 * distance + 23.5$$

The distance is the factor which is computed from the accelerometer and used for calorie meter. The computation can be found in the earlier section of the report mathematical modeling.

9.2 DATA STRUCTURES

Data structures are irreplaceable and indispensable for any software system. Without data structures, even the software, which has incredible algorithms, would run extremely slow.In Go-run, array plays the most vital role because its short processing time and simplicity of management. Array, which is one of the most common data structures, collects a set of elements that consist of variables and values. It is easy to get access to an array element by using its index. This could directly lead us to variables or values that we required. In Go-run, array is used for not only storing user account information, but also information of regimens, even the schedule of every regimen are conserved by array.

Hash tables are useful data structures that are always used to transform keys into the index of an array element. When it comes to the alert system, we plan to implement hash tables to calculate how much effort should the user accomplish to keep on track after the workouts he/she has missed.

The following is a section of the database which is tabulated:

Uname	Password	Regimen	Week1	Week2	Week3	Week4	Week5	Week6	Week7
Uname	Password	Regimen	Week1	Week2	Week3	Week4	Week5	Week6	Week7
Uname	Password	Regimen	Week1	Week2	Week3	Week4	Week5	Week6	Week7
Uname	Password	Regimen	Week1	Week2	Week3	Week4	Week5	Week6	Week7

The above table is used for a single user where each regimen is fixed. The database gets updated every time user uses the app.

SECTION 10: USER INTERFACE DESIGN AND IMPLEMENTATION

10.1 USER INTERFACE DESIGN

This section is going to describe the graphical interface design of those most irreplaceable use cases gradually.

Select Regimen:

At the very beginning, we will introduce about User Interface Design of Select Regimen, which is detailed described in section 3.7.1, step by step:

1. After Login, the system will provide available options for regimens for user.
2. User could click on regimen that he/she is interested in.
3. System will demonstrate detailed information about the selected regimen.
4. User could type to choose which regimen he/she is going to follow. Start date and target date are also required.

Sign-up:

Registered must be done if user wants to use the login action and eventually get access to higher privileges.

1. Visitor could choose to Sign-up by click the button on the homepage of our website.
2. System displays option to input username and passwords. After filling in all blanks, visitor could click on sign-up to accomplish Sign-up.
3. System will accept visitor as a user if there is non-existing username, and then asks user provide more detailed information about user's body condition. When finished filling, one click could help user to finish the process of Sign-up.

View Progress:

Users of Go-run would be extremely interested in getting information about how much progress they achieved. In our system, user could view progress in following steps:

1. User need to login to get access to View Progress since it is available for registered users.
2. System is going to display overall progress. User could get information by just a click.

Navigate to Information Pages:

1. User selects the navigational button for the specific information that could be regimen or diet, or community (only for registered user).
2. Required information will be displayed to user to achieve Navigate to Information Page.

Start workout/End workout:
These two use cases capture the overall process of recording the user's workouts. It is important to understand that the major chunk of real time data acquisition is accomplished through these two use cases. It has thus been considered fit to discuss the two together as a continuous step as one paves way to the other and start is a prerequisite for end workout.

1. User presses the "Start Workout" button to initiate the workout.
2. After pressing, the system will provide information about distance, speed, timer and an "End Workout" button.
3. The workout would be end when the "End Workout" button has be pressed or the timer runs out of time.
4. The system will provide an option to View Progress when workout is finished. User could press the button to View Progress or just exit the application.

10.2 INTERACTION OF USER INTERFACE

In this section, the actual implementation in User Interface will be demonstrated by showing figures. Due to the fact that we have mobile application and website, we will discuss them separately.

10.2.1 User Interface of Mobile Application

When go out for a workout, user would face Mobile Application that provides fundamental function to gather data during the workout. Mobile Application has a user-friendly interface that is easy to use.

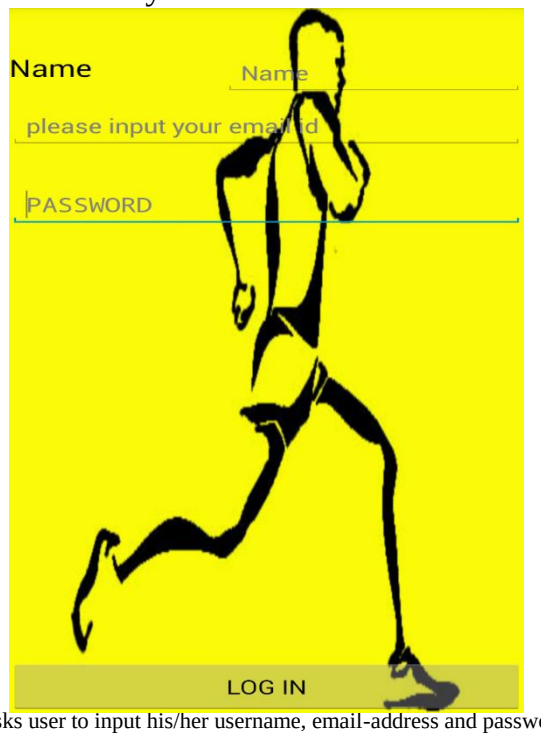


Figure 10.1: Login page asks user to input his/her username, email-address and password to login to the system.

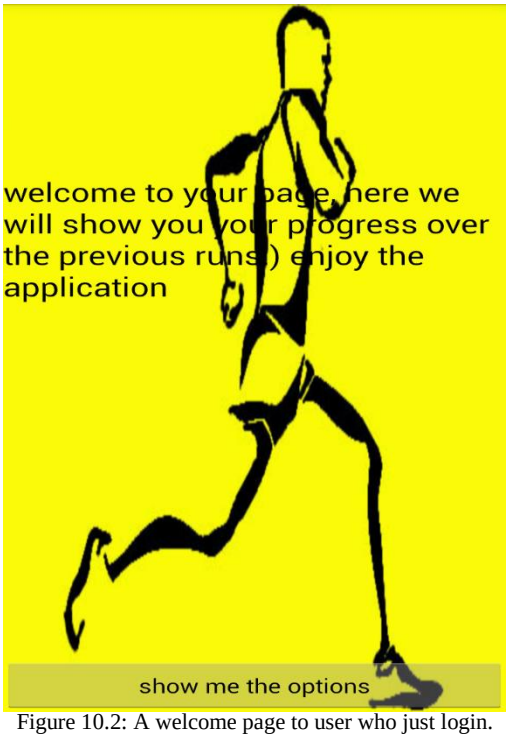


Figure 10.2: A welcome page to user who just login.

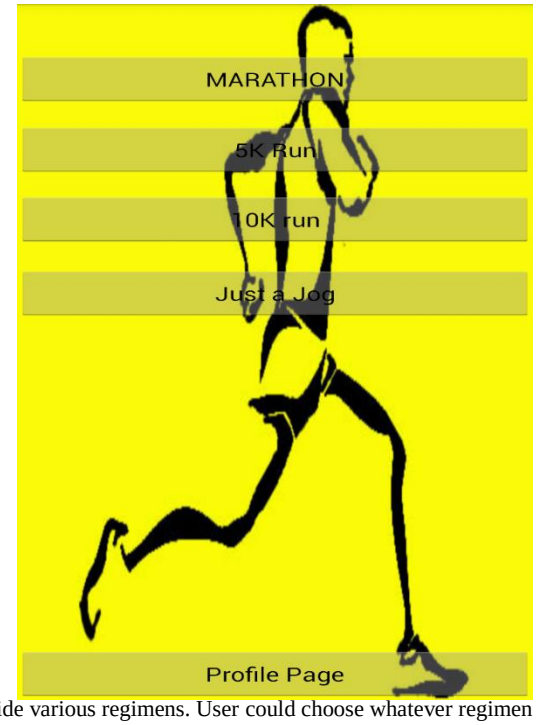


Figure 10.3: We provide various regimens. User could choose whatever regimen he/she wants to follow.

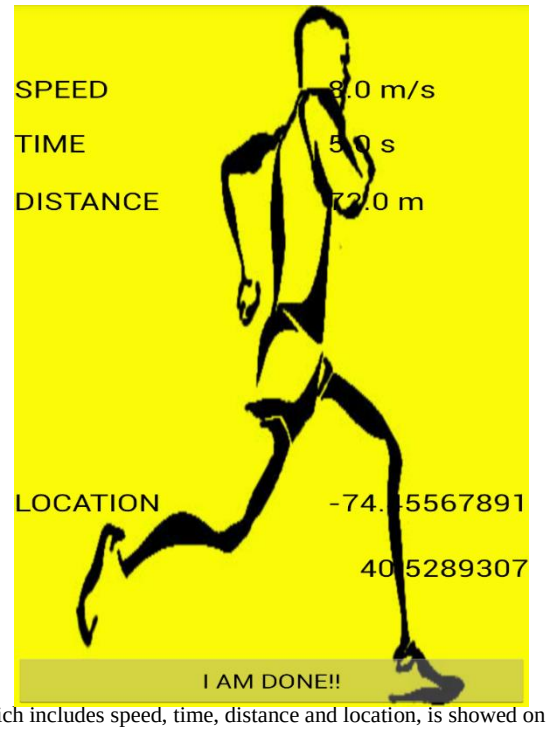


Figure 10.4: Real time data, which includes speed, time, distance and location, is showed on the screen. When user wants to finish workout, he/she could click “I am done” button.

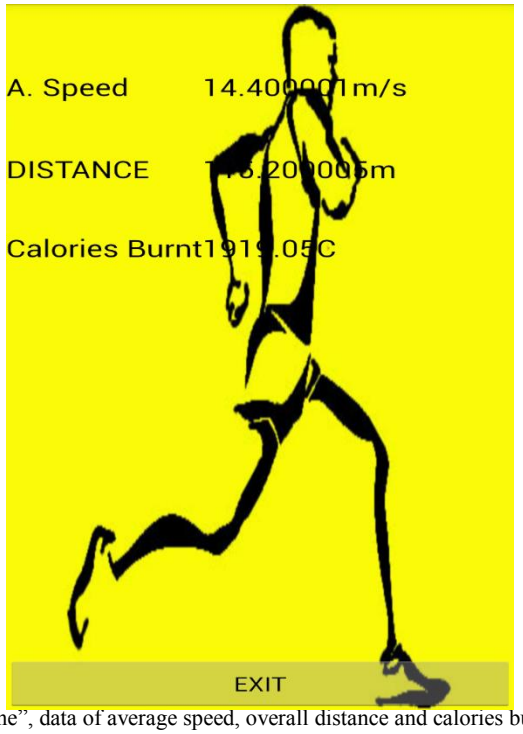


Figure 10.5: After click “I am done”, data of average speed, overall distance and calories burned will be demonstrated. User could touch “Exit” to end the program.

10.2.2 User Interface of Website

Website is the main way to get involved in Go-run. User could get access to some of information without registration. However, registered user has permission to more comprehensive functionality.

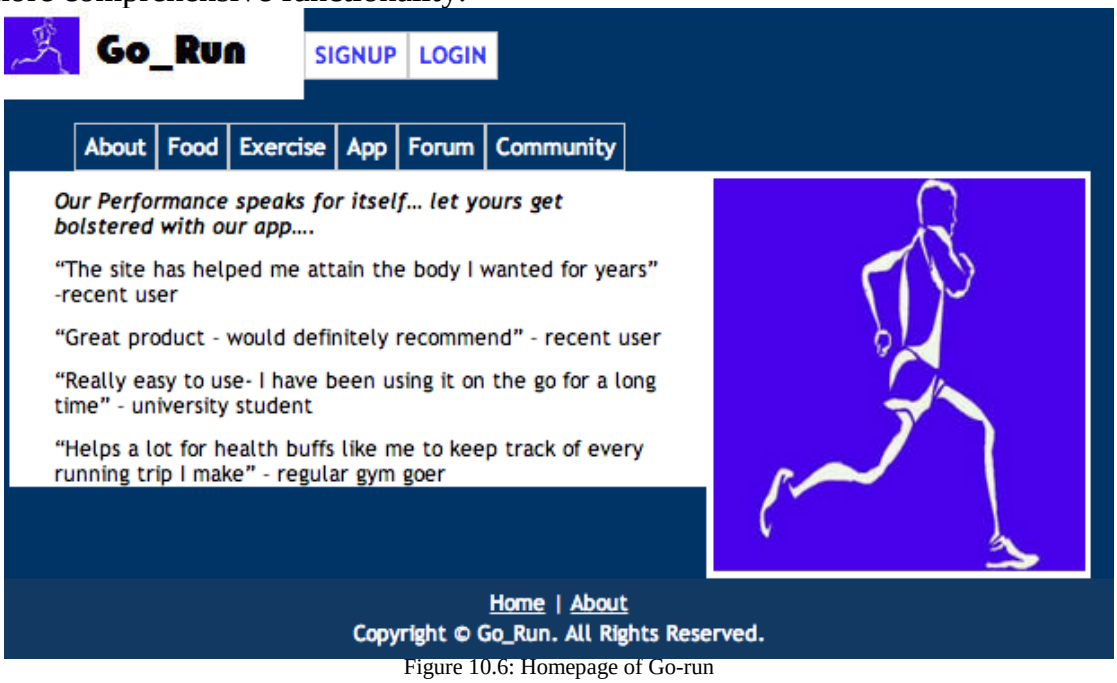


Figure 10.6: Homepage of Go-run

Figure 10.7: Sign up, where user needs to input username, e-mail address and password

Figure 10.8: Login, where user input his/her username and password to login

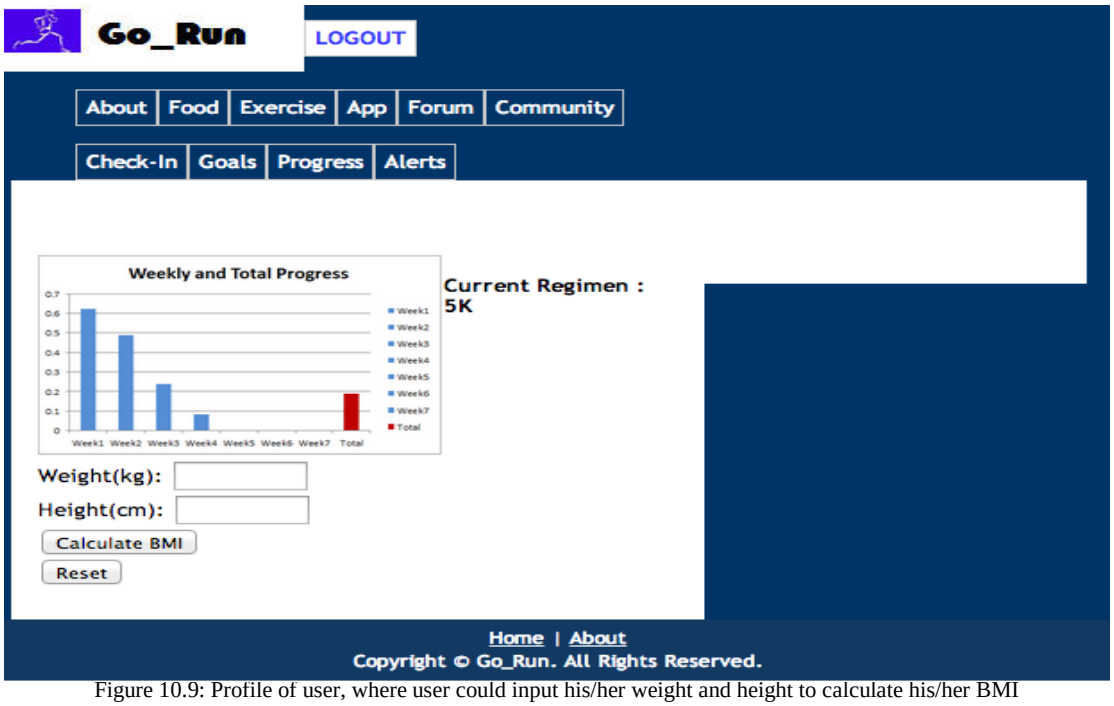


Figure 10.9: Profile of user, where user could input his/her weight and height to calculate his/her BMI

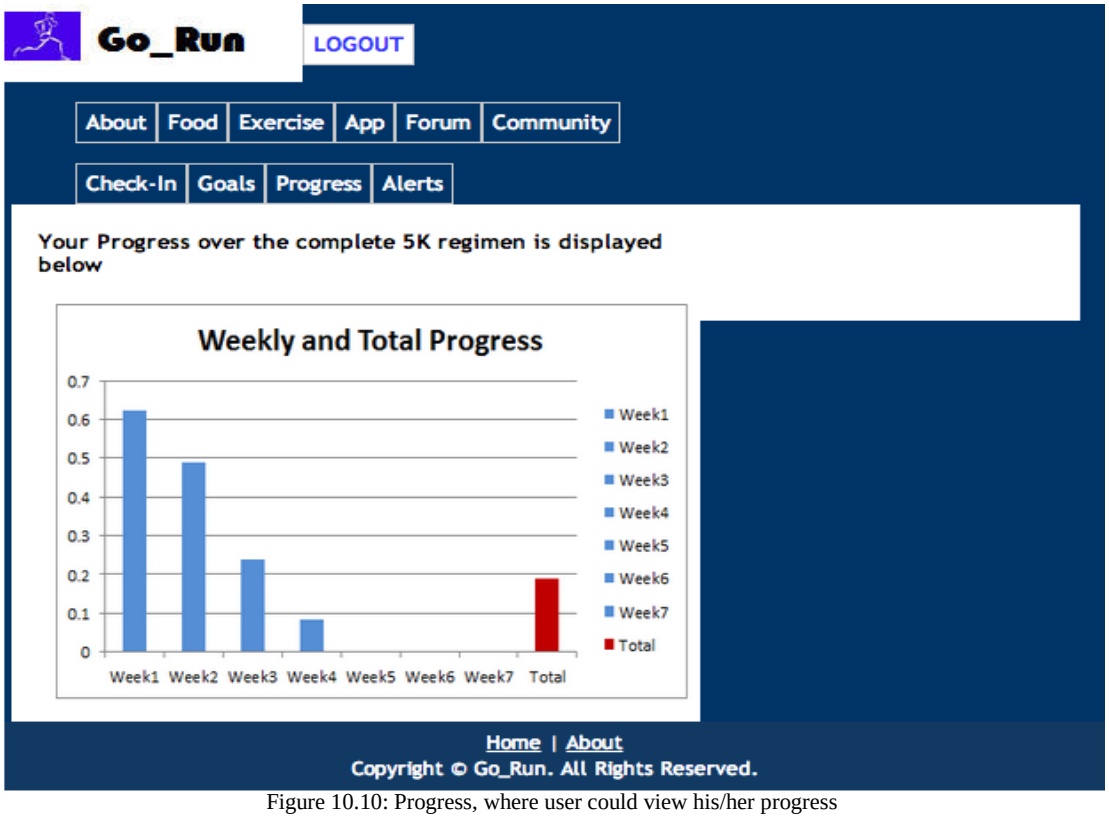


Figure 10.10: Progress, where user could view his/her progress

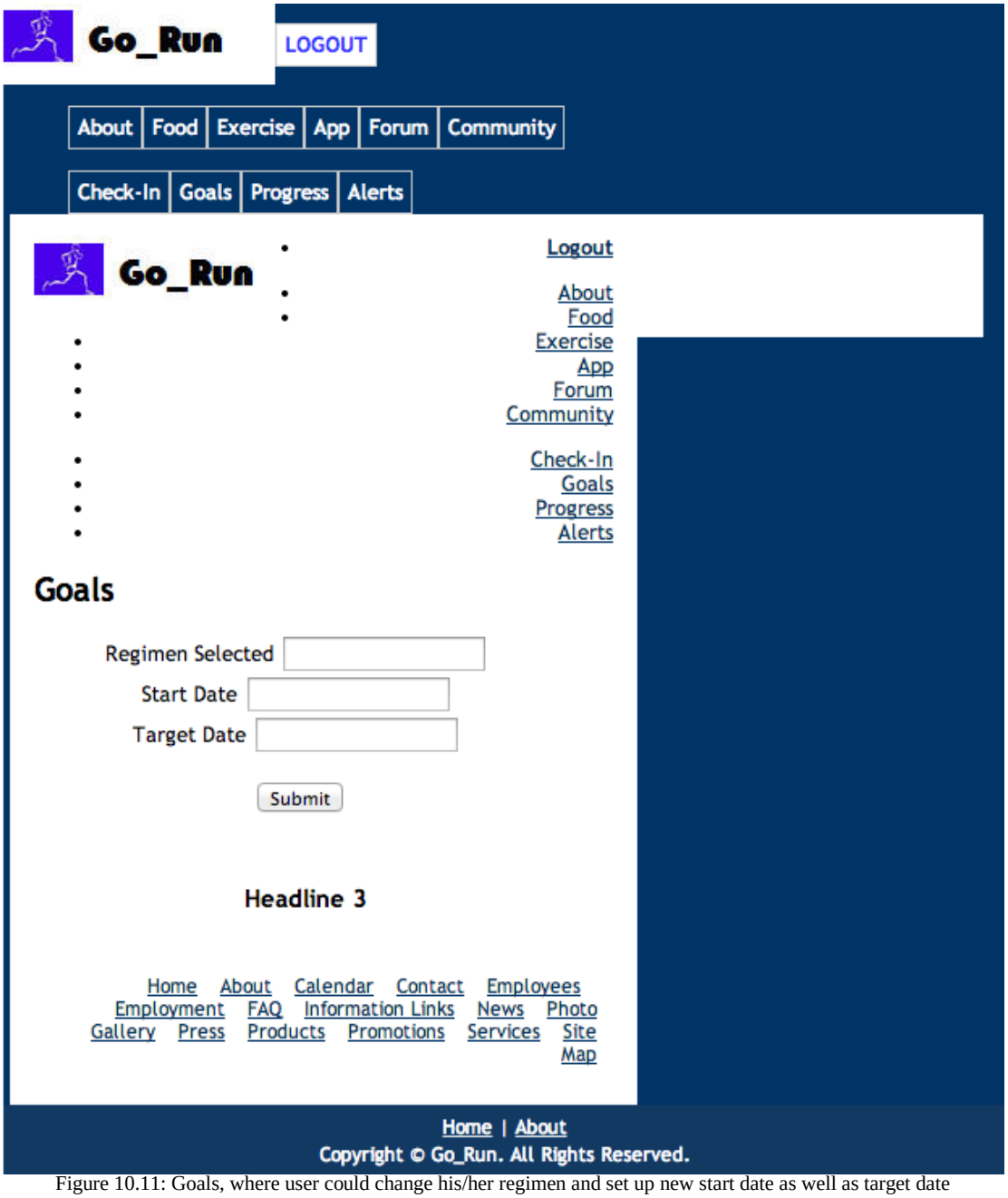


Figure 10.11: Goals, where user could change his/her regimen and set up new start date as well as target date

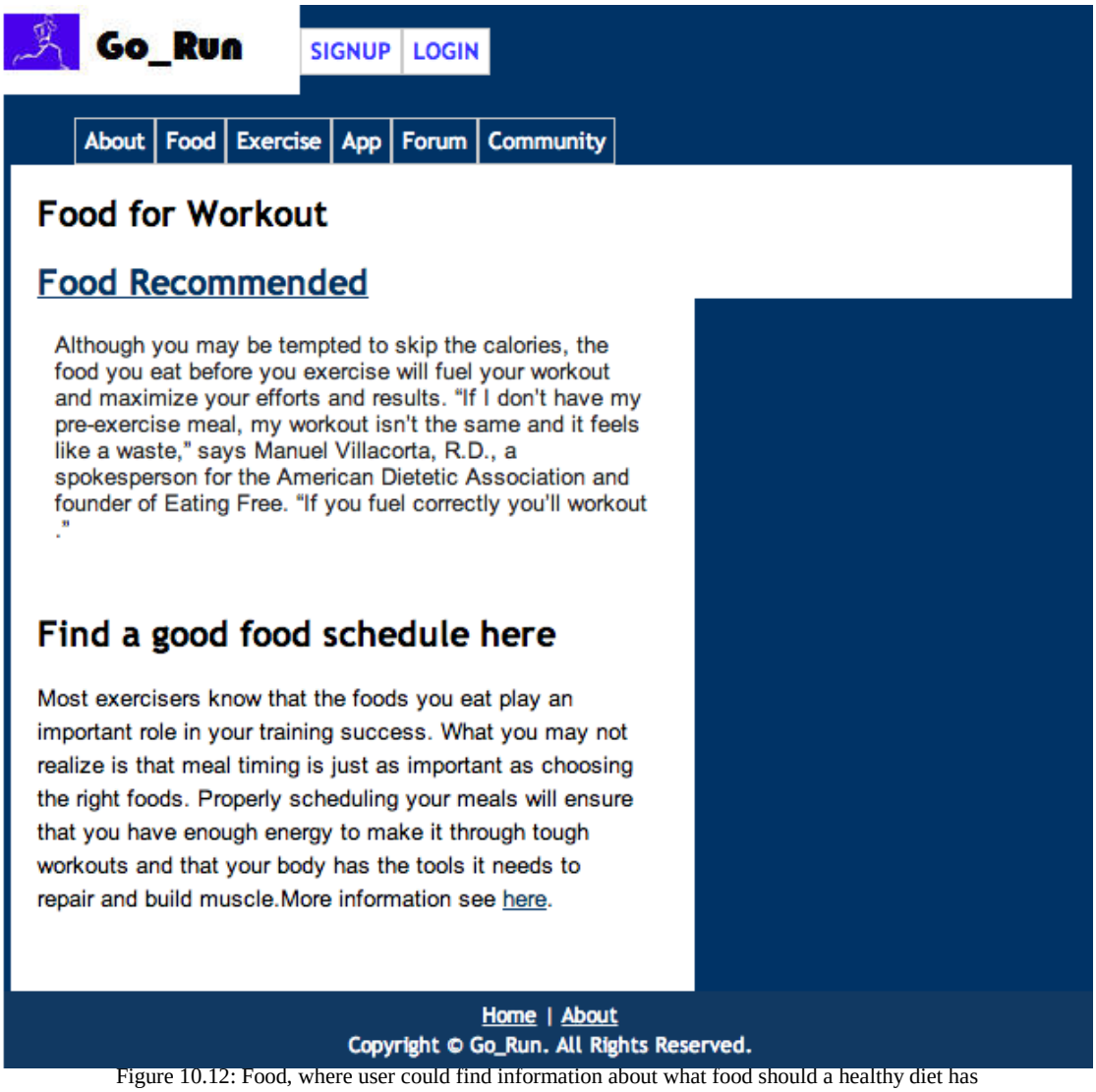


Figure 10.12: Food, where user could find information about what food should a healthy diet has

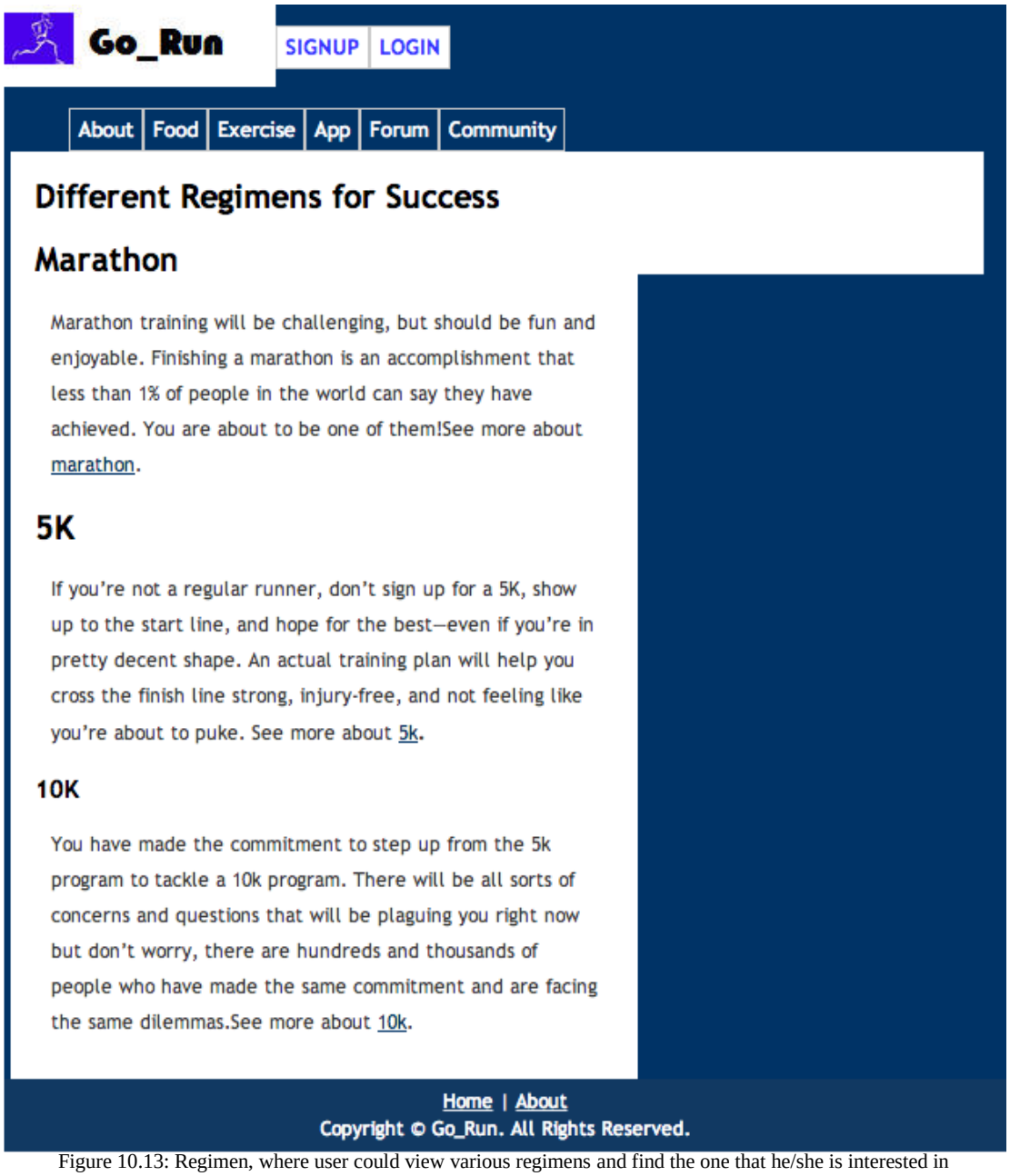


Figure 10.13: Regimen, where user could view various regimens and find the one that he/she is interested in

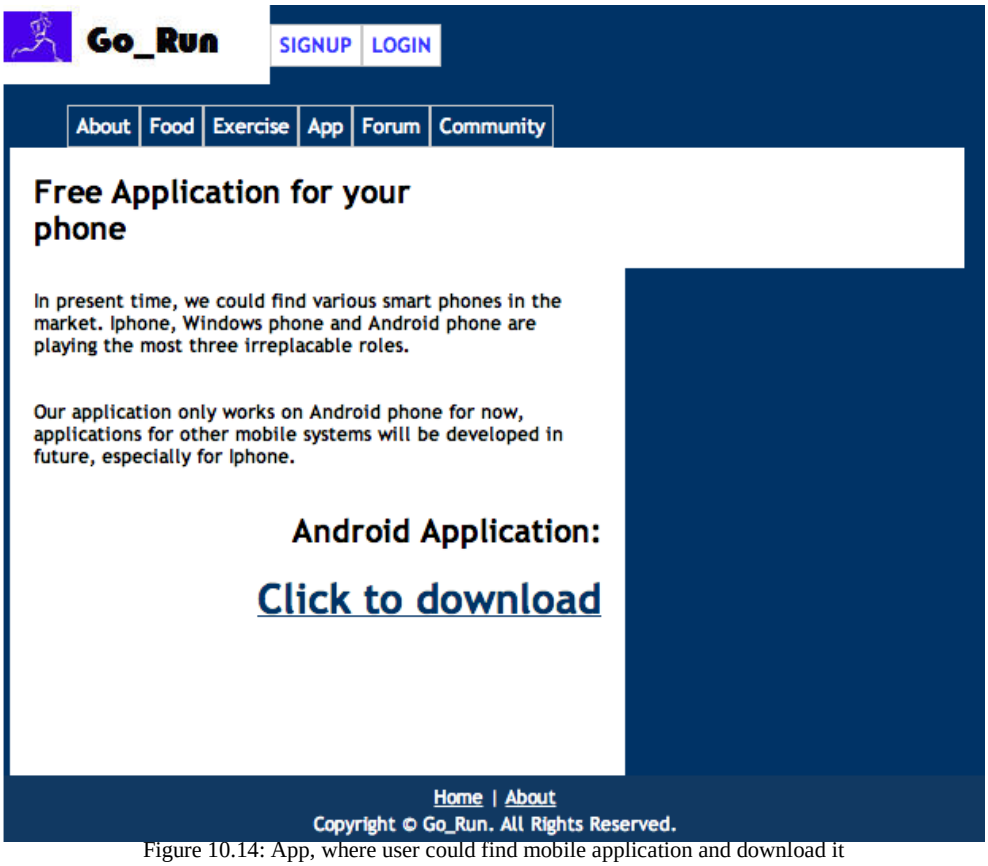


Figure 10.14: App, where user could find mobile application and download it

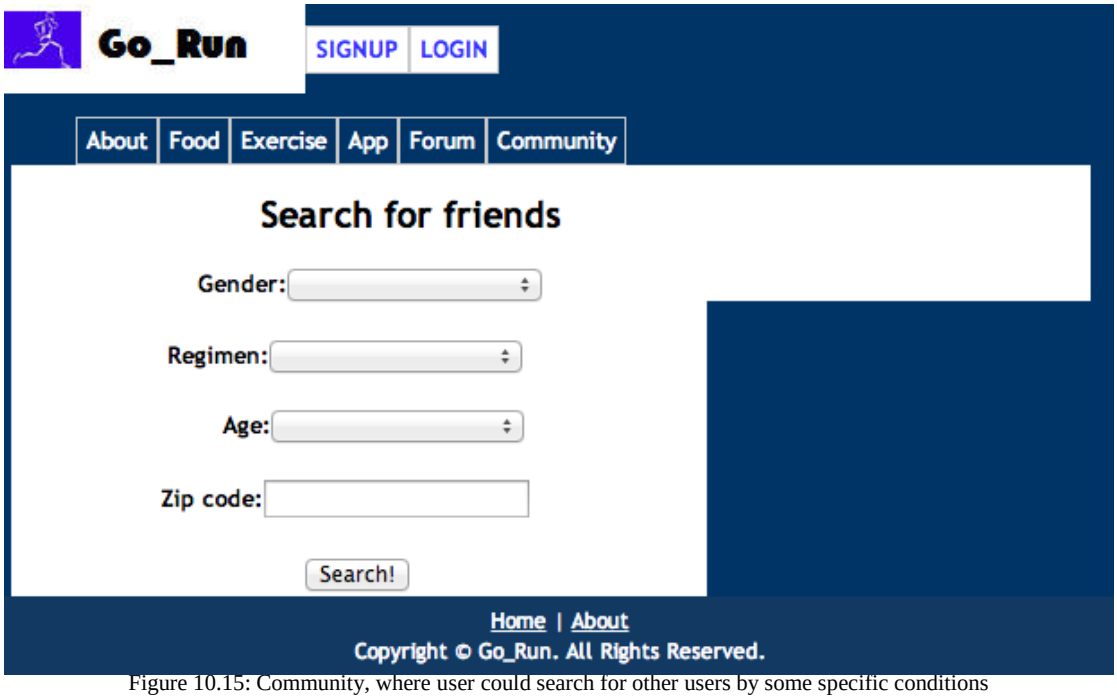


Figure 10.15: Community, where user could search for other users by some specific conditions

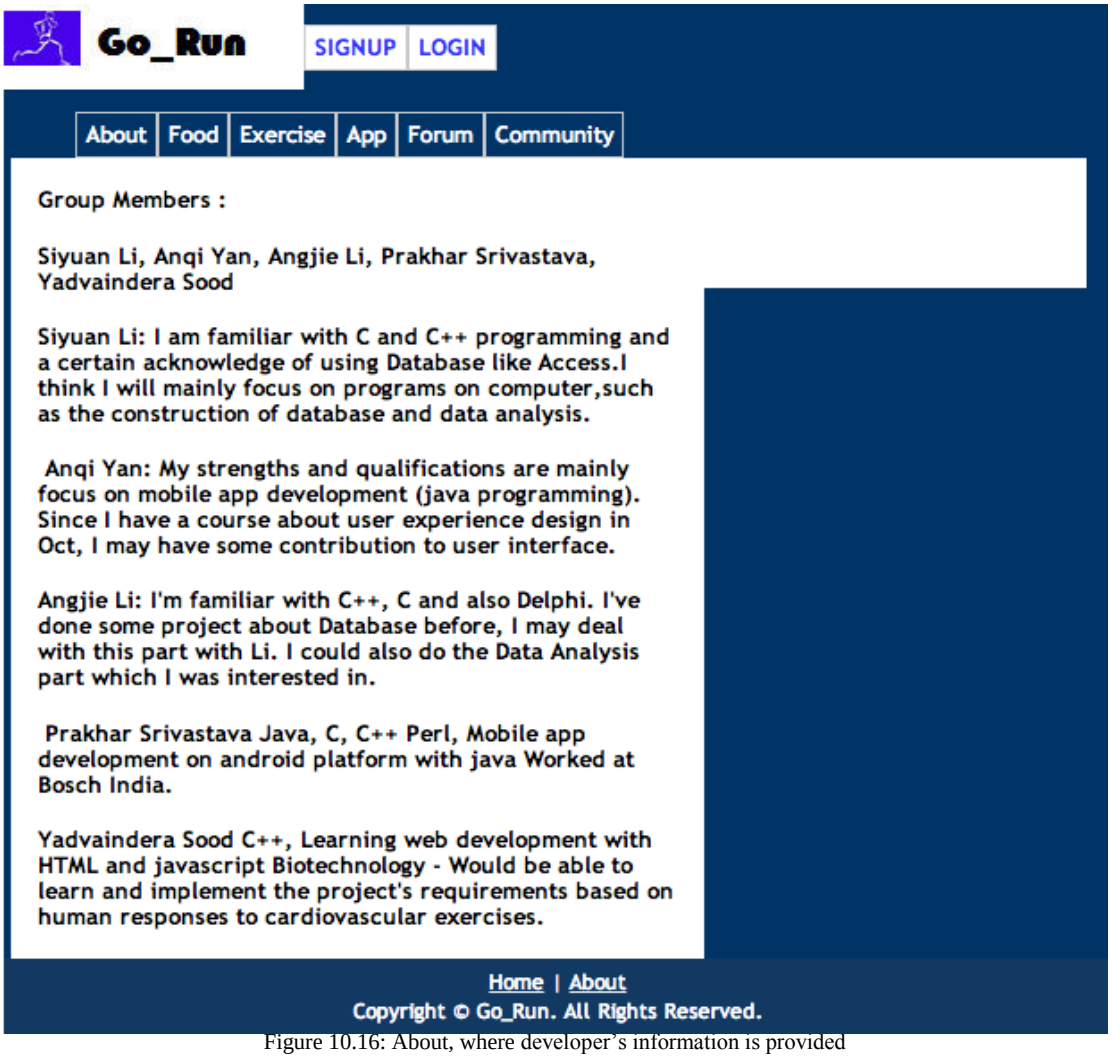


Figure 10.16: About, where developer's information is provided

10.3 DETAILED USER INTERFACE IMPLEMENTATION

In this section, we will discuss about detailed user interface implementation, and also procedure of some crucial functionalities.

10.3.1 Select Regimen

1. Click on Goals.
2. Input the regimen user wants to follow, start date and target date.
3. Click on Submit.

10.3.2 Sign-up

1. Click on Sign-up.
2. Fill in the three blanks for username, password and verify password.
3. Click on Sign up.

10.3.3 View Progress

1. Click on Progress.

10.3.4 Navigate to information pages

1. Click on hyperlink that user wants to view.

10.3.5 Start/End Workout

1. Touch screen to choose what regimen users is following.
2. Start running.
3. Touch on the "I am done!" button to finish workout.

The above description provides a clear perspective of the sequential flow in the application. In Select regimen, two clicks and inputting some formation would get work done. In Sign-up, we just need two clicks and some typing. In View Progress, just one click would get you everything you want. In Navigate to information pages, clicking is the only thing that user needs to do. In Start/End Workout, touching screen twice are enough to finish the work. The User interface is thus fairly intuitive and provides good functionality without getting cluttered and complex.

Mobile Interface usage:

In order to use the application the following steps are to be performed

1. Start the application.
2. Input the login details (name, userid and password)
3. The profile page shows the user data, click the show me the options
4. Select from the various options on the screen.
    a. Marathon,10k,5k or just a jog
    b. Go back to profile page (click on profile page)
5. You can now see the "to do" and the diet info regarding the regimen selected in step 4.
    a. Choose to go back by clicking the Back button
    b. Choose to go to the profile page by clicking the home button
6. Proceed to the start page by clicking the lets go button.
7. Speed, distance, time and your present location are provided to the user while running
8. Click the "I am done" button to end the workout
9. Workout statistics are displayed
10. Only button provided is the Exit button, which closes the application.

Web Interface usage:
On accessing the website the user lands on the website's homepage. The user can access the website under three different profile categories: a visitor, a registered user and an admin. The homepage contains a navigation bar that allows easy access to different sections of the website. As a visitor the user can access the following sections of the website.

1. About: The user can view details about the developers, what the product is and contact information of the development team.
2. Food: General and specific food related information that can be helpful while following a particular regimen and links to websites for more specific and relevant information.
3. Exercise: The different regimens that the software allows the user to follow. These are at different difficulty levels and the user can read details about them at devoted websites whose links are provided.
4. App: The mobile application and all the features that it provides. There is a local link that allows the user to download and install the mobile application if accessed through a mobile device.
5. Forum: The user can access the forum and read threads from existing users however cannot post new threads or reply to them as a visitor.
6. Community: The user, as a visitor, can access the details of existing users and filter them based on specifications like weight range, gender, regimen being followed etc.

As an admin the user can access the following features.
1. Delete a user in case of malpractice or unused account or any other reason that might be thought of as legit.
2. Monitor and delete forum threads and responses as and when required.

On logging in the user can access the following features.
1. Check-In: The user can access the page to view or update his/her current body measurements, regimens etc.
2. Goals: The user can view and update his/her goals that have to be attained in a given duration of time.
3. Progress: The user can view his/her daily progress and overall progress during the regimen.
4. Alerts: Based how strictly the user has been able to follow the regimen he/she gets warning notifications that allow him/her to keep track of progress. This also tells the user how much has he/she been thrown off the target.

SECTION 11: DESIGN OF TESTS

11.1 TESTING DESIGN

Testing design is design techniques that add certain testability features to the system. Testing design is often viewed as executing a program to see if produces the correct output for a given input which wrongly implied that testing should postponed until late in the lifecycle[1].Actually, errors introduced during the early stages of the software lifecycle and testing activity should started as early as possible.

Testing uses different combinations of inputs to detects faults, but trying all possible inputs exhaustively will make a huge cost. In order to detect enough faults with limit cost, we decide to make the testing design following the hierarchical structure of the system. We will start the test from individual components, unit resting, then the composition of these components, integration testing, and finally the whole system, system testing, ensuring the system suits the function and non-functional requirement.

So there are three different kind of testing should be made here.

(1)Unit testing:

For unit testing, we mainly use the strategy of boundary testing and for several specific case, we just use the equivalence testing, like the input process in sign-in.

(2)Integration testing:

We make one integration test for several related unit by horizontal bottom-up integration. For example, a workout integration test for Select Regimen, Start Training, Track Position & Show Speed and End Training & Show Progress. As a lower level, Select Regimen and Track position & show speed have no pre-requirement for any other part, so they make the bottom level and should be test first, which makes the fundamental for this testing.

(3)System testing:

As acceptance testing mentioned before, we could make sure whether this system is correct or not by comparing with the functional and non-functional requirement in acceptance testing.In following part, we will mainly focus on Unit testing and integration testing.

11.2 UNIT TESTING

1. Sign-In

Unit to test	**Sign-In**
Assumption	The webpage displayed the sign-in input screen and wait for user's action.
Input	user ID, user password, user information
Expect output	Register successfully with proper input, register unsuccessfully with improper, and make alter the same time.
Pass	Function fit the requirement.

Fail and probable error	(1) Register still success with improper input. Error related input checking. (2)Register success with proper input but no related data show in data base. Error related to connection with data base.

Comment: This unit is the start of the whole software, all the other functions based on it, so there is no doubt that it is quite important to keep this part correct.

(2)Select regimen

Unit test	**Regimen Selection**
Assumption	After user signed-in, the personal homepage waits user's action Input
Expect output	Regimen selecting successfully and training goal updated.
Pass	Regimen select correctly with proper input.
Fail and probable error	(1) Proper input but regimen not be selected. Errors related connection with database OR regimen select function (2) Regimen selected successfully, but goal didn't update. Errors related function translate regimen into goals

(3)Change regimen

Unit test	**change regimen**
Assumption	After user selected regimen, and personal homepage waits user's action Input
Expect output	Regimen changing successfully and training goal updated.
Pass	Regimen change correctly with proper input.
Fail and probable error	(1) Proper input but regimen cannot be changed. Errors related connection with database OR regimen change function (2) Regimen selected successfully, but goal didn't update. Errors related function translate regimen into goals

(4)Start training

Unit test	**Start Training**
Assumption	After user select regimen, and Android App waiting for action
Input	Click on start button
Expect output	Training starts successfully and turns to other functions.
Pass	Turn to track position &show speed function successfully

Fail and probable error	(1) Click on start button, but nothing happens. Errors related training start function

(5)Track position & show speed

Unit test	**Track Position and Show Sped Assumption**
Input	Position change
Expect output	Show position, speed and distance run.
Pass	Position, speed and distance show correctly.
Fail and probable error	(1) Input properly, but data stop changing Error related Network connection (2) Proper input, but wrong output Error related to output calculation function

(6)End training & show progress

Unit test	**End and progress**
Assumption	After user click stop button
Input	Data collected by training
Expect output	Progress has made
Pass	Show progress correctly
Fail and probable error	(1) No progress shown after training Error related training calculation OR function related with showing data (2) Can't stop training Error related end training function

(7)Exit

Unit test	**Exit**
Assumption	After click Exit button
Input	click on Exit
Expect output	system exit

11.3 INTEGRATION TESTING

Integration testing is used for testing the combination of several units. In our case, we will use Bottom-up model. First, we will take units don't need any input from other units as level1.Then,we add the units that needs input from level1 as level2 and make a test to ensure the conjunction between these two unit is well built. And step by step, the whole constructed integration will be tested.

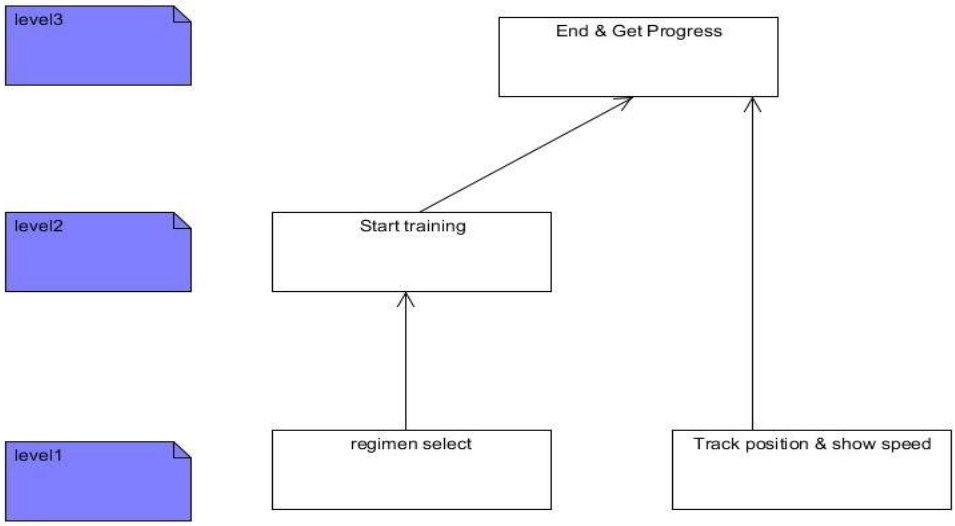


Figure 11.1 Android App integration test

From Figure 11.1, we can see that as regimen select and tracking position &show speed do not need any input from others, so these two can be level 1 unit. But start training and End &get progress must take input from units before, I put them as level 2 and level 3.By this chat, we can decide our order for testing which always start with lower level, then higher level and then their conjunction.

SECTION 12: HISTORY OF WORK AND FUTURE PLANS

History of Work

The project mostly kept the whole team on its toes throughout the semester as it was pretty hard to meet up the goals that we had set for completion. It was a valuable experience in team work and time management that all of us have learnt while trying to fulfill our goals. The first part of the project was mostly spent in planning, setting up targets, designing and documentation. As much as each of us disliked documenting, towards the end it did turn out to be very useful as we slowly realized how half the work of designing the functions and the interfaces had already been completed. As a team we were mainly successful because of discipline as we stuck to our twice weekly meetings throughout the semester. Initially these meetings were long and each member presented what he/she had in mind for the next part. We also discussed upcoming targets in these meetings. Even when our schedules did not allow us to meet we made it a point to meet online through tools like zoho.com Google groups and Skype in order to maintain continuity and discipline. Towards the later part, when development kicked in the meetings became briefer and were barely short online or personal discussion sessions for updates and goal setting. We had implemented an online discussion strategy using Zoho(for online meetings) and Google Groups for sharing all our documents. The links are given below

https://groups.google.com/forum/#!forum/gorun Google group for SE project group#8

https://sites.google.com/site/gorunsegroup/ Website of SE project group #8

We made it a point to stick to the deadlines set by the professor even for report sections that were not graded at those deadlines to make sure that we were on track. This also helped us to achieve most of our deadlines in time. Initially we had planned to implement use cases like forums, alerts etc. However over time we realized that these could not be realized in the given time frames and we focused our attention on the more fundamental and necessary use cases. Since each one of us was new to whatever tasks we had taken up, the learning curves were fairly steep given the time frame and thus implementation had to be restricted to the more important sections. Implementing working servers and other components like the database and a functional website were complex enough challenges for the team given our skill set. These foundational tasks consumed substantial amount of time and we had to move some deadlines accordingly. However, division of specific tasks helped us stay on track through the project.

After the first demo, we shifted our focus on developing a working interface cross-platform between the GPS and accelerometer as it had been pointed out as a major area that needed work. Given the complexity of the task and paucity of time we were more often than not thrown off our targets. However we finally decided to go back to our initial ideology of proper division of work and part of the team shifted

its focus back to web site development and integration. Because of lack of time we still have some problems with implementing a proper interface between the phone and the web platform and due to the same some data stored in the database is incomplete.

Since in the earlier part of the project we had focused a lot of attention to developing and designing working algorithms for most of our use cases and laying down business policies for implementing them, it is a smoother road ahead to accomplish the same. In the time to come we are determined to develop a proper interface between the phone and the web platform through cloud services rather than the mail server based approach that we have followed currently. We also want to improve the aesthetic appeal of our product which would complement its existing ease of use.

Looking back it is clear that there are certain compromises in implementation to achieve the required targets in time. Our project has been no different. However, given the idea that we have been pursuing and uniqueness of some of its key features, we certainly want to implement some of the more advanced features like discussion forums, and include some extra sports like bicycle riding and treadmill based workouts for data acquisition through cell phones.

SECTION 13: REFERENCES

1. http://www.nike.com/us/en_us/?cp=usns_kw_AL!1778!3!27759807902!e!!g!nike
2. https://www.runtastic.com/
3. http://runkeeper.com/
4. http://www.myfitnesspal.com/
5. http://www.myfitnesspal.com/food/calorie-chart-nutrition-facts
6. http://www.webmd.com/fitness-exercise/
7. http://en.wikipedia.org/wiki/Nutrition
8. http://www.runnersworld.com/nutrition-runners/best-foods-runners
9. http://www.marathonrookie.com/
10. http://www.marathonrookie.com/marathon-motivation.html
11. http://www.coolrunning.com/engine/2/index.shtml
12. http://en.wikipedia.org/wiki/Accelerometer
13. http://en.wikipedia.org/wiki/GPS_tracking_unit
14. http://www.runnersworld.com/tools/calories-burned-calculator
15. http://boulter.com/gps/distance/
16. http://stackoverflow.com/questions/13264718/how-to-calculate-running-speed-using-accelerometer-sensor-in-android