Fitness Tracker

MCA 6th Semester
MCSP-060(Project Synopsis)

Ujjwal Maji
Enrollment No-

TABLE OF CONTENTS

1 INTRODUCTION & OBJECTIVE OF THE PROJECT:

1.1 INTRODUCTION

In our busy life it is very hard to track our fitness policy by proper guidance. It is also very difficult to maintain schedule for exercise to our health. Mobile technology or elaborately Internet technology is the main trend among us. We like to simplify our daily life by using modern technology. We can track easily whatever we have done or eaten for our health. We usually go to gym or yoga for body fitness and want to keep track of our work to keep our body fit by the guidance of some trainer. But it is very difficult to track the whole process manually. We can do the same things in different ways. With advent of new technologies we use smart phone, notebooks, tablets etc. We could develop an app for Android to keep tracking this process and which will also be synced with the particular website.

Fitness Tracker will provide an interface to organize our daily exercise or activity chart, exercise description, gym schedule etc. in Android device. And it will also share with the same thing in web server for website interface. It will allow users to add same tracking notes to the web sites. User can add details anytime and keep it for update, and then whenever user comes online the pending updates will be uploaded to the destination sites.

Fitness Tracker will help users to track his fitness details from Indian Sub-continent Context. It will have preloaded database for Indian cuisine and exercises more suitable for Indian people.

Main features of the Fitness Tracker are:

- Tracking daily exercise
- Tracking Daily food consumption
- Tracking Weight
- Exercise chart & description
- Diet & food chart
- Gym schedule etc.
- Updating the details in Web Sites from mobile device ,
- Web Sync
- Reminder/ alarm for Gym Schedule
- Weekly, Monthly or yearly progress view

The overview of this software is displayed below in the diagram.

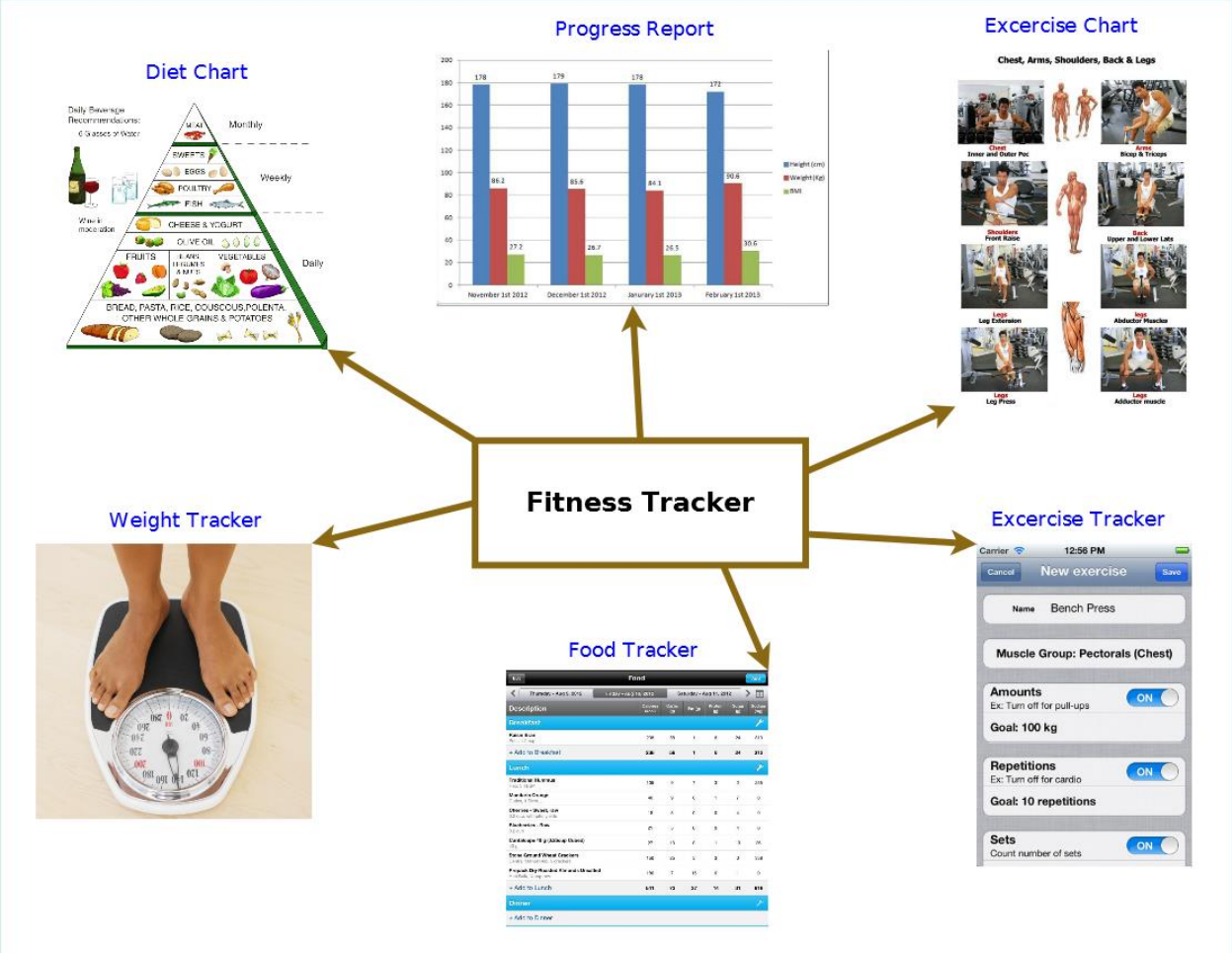


Figure: Overview of Fitness Tracker

1.2 OBJECTIVE

- Single solution for fitness related track in mobile device as well as in web site
- Keeping track of daily fitness related activities
- Timely update whenever internet is available
- Manually update data in internet whenever user want
- Guide user to reach the goal of his/her fitness
- To be involved with full-fledged software development
- To know about new technologies like Android, CodeIgniter, Rest API, etc.

2 PROJECT CATEGORY:

This software will follow Object Oriented Programming Paradigm and use below mentioned areas.

Front End/ GUI Tools	**:**	Eclipse, NetBeans IDE 7.3, Java, Android SDK, Code Igniter 2.1.3
Backend	**:**	MySQL 5.5.15
Internet Technologies	**:**	HTML, JavaScript, PHP, Code Igniter 2.1.3 Framework.
Operating Systems	:	Android, Windows XP, Windows 7
Applications	:	Android Mobile Utility application

3 PROJECT STRUCTURE:

3.1 HARDWARE AND SOFTWARE SPECIFICATION

3.1.1 Hardware Requirement

Memory capacity	**:**	50 MB of available Memory space
RAM	**:**	256 MB
Processor	**:**	484 MHz or faster

3.1.2 Software Requirement

Android 3.0 Honeycomb or above

3.2 TOOLS/PLATFORMS USED (HARDWARE/SOFTWARE):

3.2.1 Hardware:

- Laptop with 2GHZ processor
- 2 GB RAM
- 320 GB hard disk (NTFS File System)
- 8 GB USB Pen Drive (FAT File System)

3.2.2 Software:

- Windows XP (x86) with Service Pack 3 / Windows Vista (x86) with Service Pack 2 / Windows 7 (x86)

- Android 4.2 Jelly Bean
- Eclipse
- NetBeans IDE 7.3
- MySQL, Sqlite database
- Dia for Diagram Drawing & Modeling.
- Code Igniter 2.1.3 Framework
- Twitter Bootstrap
- FileZilla
- GitHub

4 REQUIREMENTS AND ANALYSIS

4.1 PROBLEM DEFINITION

We used to maintain our trainers guideline without any perfect assumption or calculation. We could not track the accentual progress or digress by the manual calculation. But in the age of digitalization, we are completely dependent on mobile device and internet connection. We are making our daily life easier and faster using the modern technology. So we can track our fitness schedule by use this modern technology. Time to time update regarding fitness via mobile device could make a perfect calculation of health status. The purpose of Fitness Tracker is to store those activities digitally, make schedule, get progress report etc. With an attractive and easy to use Android GUI it will allows us to track down our daily activity. We could also share them with our web account as well whenever we wish. We could also track our activity regarding the fitness via web site also, whenever we have no mobile device available as well.

4.2 REQUIREMENTS SPECIFICATION

4.2.1 Functional Requirement

4.2.1.1 *Register User*

4.2.1.1.1 Introduction:

Create account for a new User.

4.2.1.1.2 Input:

Relevant User data like user name, current weight, your goal weight, date of birth, gender etc.

4.2.1.1.3 Processing:

The Fitness Tracker will create a new user entry.

4.2.1.1.4 Output:

The Fitness Tracker will generate a user to reach his goal of fitness.

4.2.1.2 *Login User*

4.2.1.2.1 Introduction:

Logging in as an existing User.

4.2.1.2.2 Input:

User will provide user id, password.

4.2.1.2.3 Processing:

The **Fitness Tracker** will check the authorization of the particular user.

4.2.1.2.4 Output:

The **FITNESS TRACKER** will allow accessing feature to the user if the given data match with the internal information, otherwise denying user.

4.2.1.3 *Add food entry for tracking*

4.2.1.3.1 Introduction:

User can add new entry.

4.2.1.3.2 Input:

User will enter details about Meals (Breakfast, Lunch, and Dinner etc.), Water, and Note etc.

4.2.1.3.3 Processing:

The **FITNESS TRACKER** will create a new entry with food value.

4.2.1.3.4 Output:

The **FITNESS TRACKER** will save entry in its records and share it in web server.

4.2.1.4 *Add exercise entry for tracking*

4.2.1.4.1 Introduction:

User can add new entry.

4.2.1.4.2 Input:

User will enter details about Exercise (Cardio, Strength etc.), Note etc.

4.2.1.4.3 Processing:

The **FITNESS TRACKER** will create a new entry with exercise value.

4.2.1.4.4 Output:

The **FITNESS TRACKER** will save entry in its records and share it in web server.

4.2.1.5 *Add weight entry for tracking*

4.2.1.5.1 Introduction:

User can add his weight on that day.

4.2.1.5.2 Input:

User will enter details about weight.

4.2.1.5.3 Processing:

The **FITNESS TRACKER** will create a new entry with weight value.

4.2.1.5.4 Output:

The **FITNESS TRACKER** will save entry in its records and share it in web server.

4.2.1.6 Add To Do

4.2.1.6.1 Introduction:

The **FITNESS TRACKER** will Store a new task in To Do.

4.2.1.6.2 Input:

Relevant task data like task detail, priority etc.

4.2.1.6.3 Processing:

Admin will enter the data in the **FITNESS TRACKER** and create a new task entry.

4.2.1.6.4 Output:

The **FITNESS TRACKER** will generate a task detail.

4.2.1.7 *Share details in web server*

4.2.1.7.1 Introduction:

User can manually share details in web server via **FITNESS TRACKER**.

4.2.1.7.2 Input:

The user selects the sync option from **FITNESS TRACKER** to upload data in web server.

4.2.1.7.3 Processing:

The **FITNESS TRACKER** will upload data in web server.

4.2.1.7.4 Output:

The Web site will display the information after sync.

4.2.1.8 Add User's new food

4.2.1.8.1 Introduction:

The user can store new food in **FITNESS TRACKER** database.

4.2.1.8.2 Input:

Relevant food data like food name, description, serving size, calories etc.

4.2.1.8.3 Processing:

Admin will enter the data in the **FITNESS TRACKER** and create a new food entry.

4.2.1.8.4 Output:

The **FITNESS TRACKER** will display the food whenever user want to enter that.

4.2.1.9 *Create food/ diet chart*

4.2.1.9.1 Introduction:

User can create his food/diet chart.

4.2.1.9.2 Input:

User will enter details about Meals (Breakfast, Lunch, Dinner etc.), Water to be taken regularly with help of dietitian or trainer.

4.2.1.9.3 Processing:

The **FITNESS TRACKER** will create food/diet chart.

4.2.1.9.4 Output:

The **FITNESS TRACKER** will save entry in its records and share it in web server.

4.2.1.10 *Create exercise chart*

4.2.1.10.1 Introduction:

User can create his exercise chart.

4.2.1.10.2 Input:

User will enter details about exercises to be performed regularly with help of dietitian or trainer.

4.2.1.10.3 Processing:

The **FITNESS TRACKER** will create exercise chart.

4.2.1.10.4 Output:

The **FITNESS TRACKER** will save entry in its records and share it in web server.

4.2.1.11 *View weekly/monthly report*

4.2.1.11.1 Introduction:

User can view his weekly/monthly.

4.2.1.11.2 Input:

User will enter his choice of duration.

4.2.1.11.3 Processing:

The **FITNESS TRACKER** will create report of that duration.

4.2.1.11.4 Output:

The **FITNESS TRACKER** will display the report in graphical and textual format.

4.2.2 Non-functional Requirements

- The application will be **self-dependent** and no dependency on other parties required.

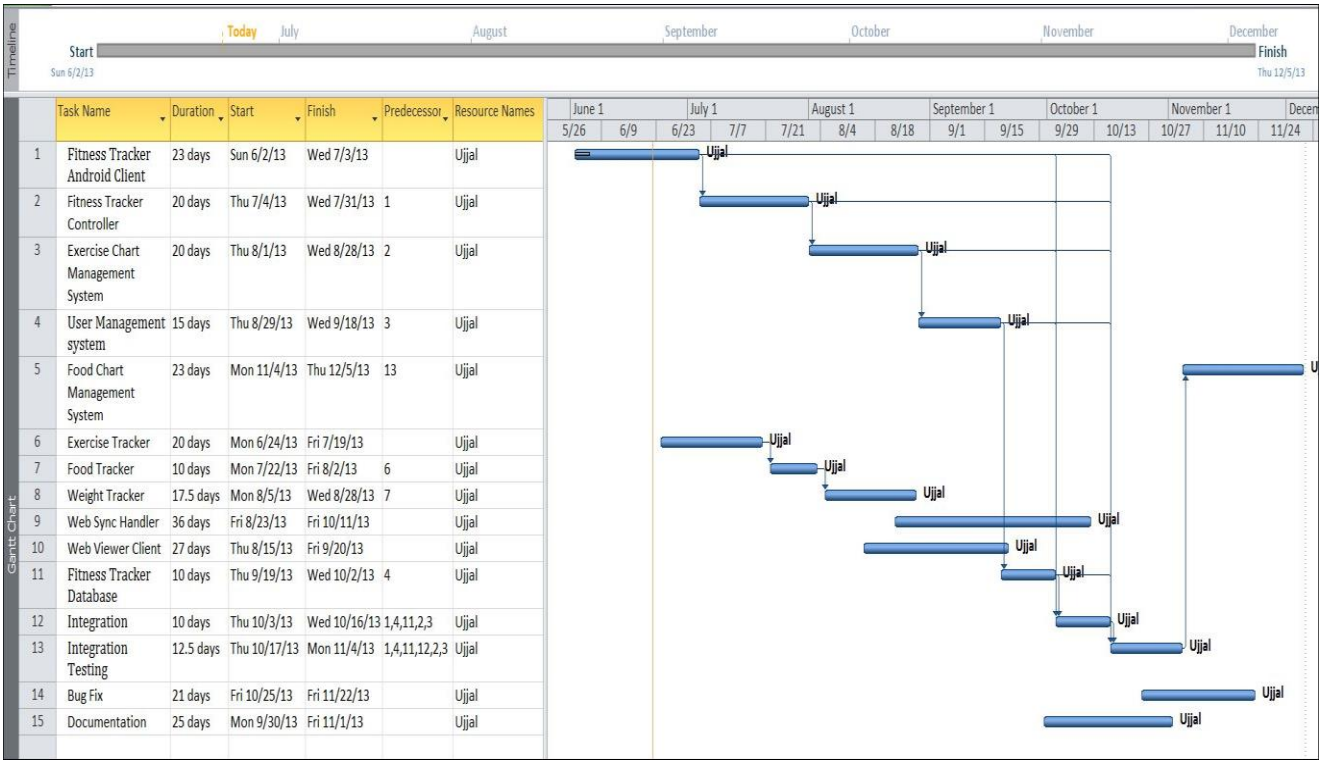
- There will be a digital **backup** and restore system.
- There will be more **opportunity** to extend the application in various type of device in future.
- The response time will be low and the system will **response fast**.
- It will be very **user friendly** and **usable** by any person with minimal technical knowledge.
- In terms of **security** unauthorized access will be denied and register user will be able to change as necessary.
- It will be **efficient** as it reduce manual labor and searching.
- **FITNESS TRACKER** will have user manual and help **documents**.
- It is designed such a way that it can be **maintained** with minimal effort.

4.2.3 Technical Specification

Front End/ GUI Tools	:	XML
IDE	:	Eclipse
Framework	:	ADT (Android Developer Toolkit) , Code Igniter, Bootstrap
Database	:	SQLite
Database Tool	:	SQLite workbench CE
Operating Systems	:	Windows XP, Windows 7 , Android
Others	:	Android SDK

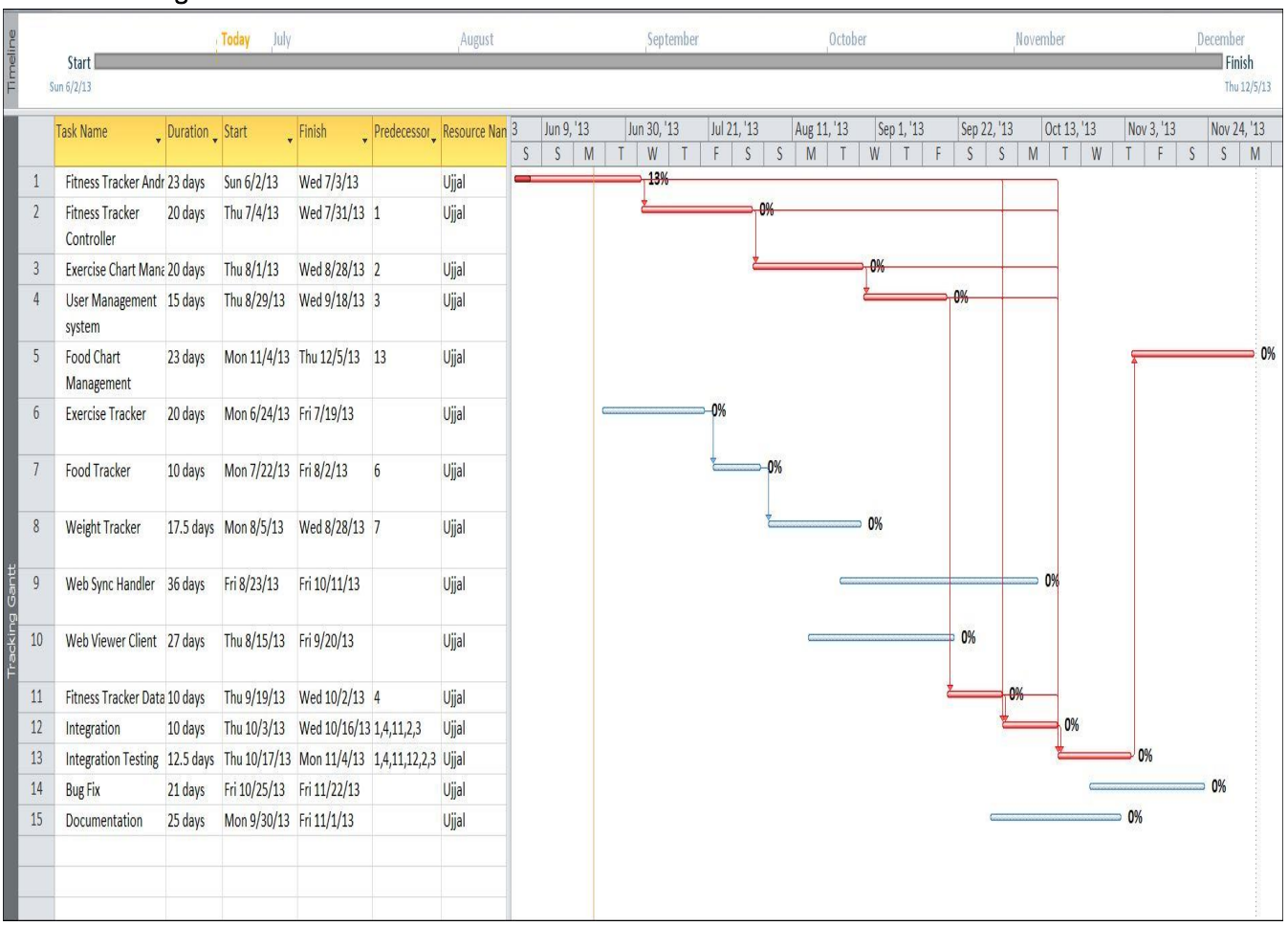
4.3 PLANNING AND SCHEDULING

4.3.1 Gantt chart

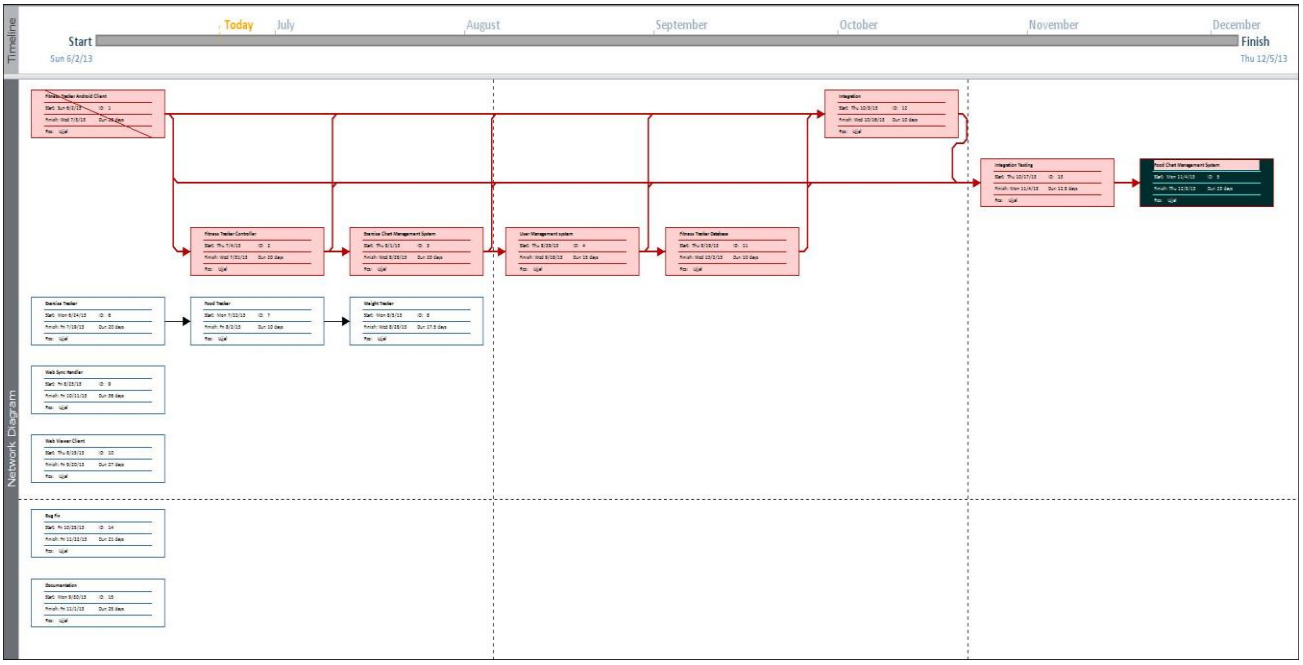


	Task Name	Duration	Start	Finish	Predecessor	Resource Names
1	Fitness Tracker Android Client	23 days	Sun 6/2/13	Wed 7/3/13		Ujjal
2	Fitness Tracker Controller	20 days	Thu 7/4/13	Wed 7/31/13	1	Ujjal
3	Exercise Chart Management System	20 days	Thu 8/1/13	Wed 8/28/13	2	Ujjal
4	User Management system	15 days	Thu 8/29/13	Wed 9/18/13	3	Ujjal
5	Food Chart Management System	23 days	Mon 11/4/13	Thu 12/5/13	13	Ujjal
6	Exercise Tracker	20 days	Mon 6/24/13	Fri 7/19/13		Ujjal
7	Food Tracker	10 days	Mon 7/22/13	Fri 8/2/13	6	Ujjal
8	Weight Tracker	17.5 days	Mon 8/5/13	Wed 8/28/13	7	Ujjal
9	Web Sync Handler	36 days	Fri 8/23/13	Fri 10/11/13		Ujjal
10	Web Viewer Client	27 days	Thu 8/15/13	Fri 9/20/13		Ujjal
11	Fitness Tracker Database	10 days	Thu 9/19/13	Wed 10/2/13	4	Ujjal
12	Integration	10 days	Thu 10/3/13	Wed 10/16/13	1,4,11,2,3	Ujjal
13	Integration Testing	12.5 days	Thu 10/17/13	Mon 11/4/13	1,4,11,12,2,3	Ujjal
14	Bug Fix	21 days	Fri 10/25/13	Fri 11/22/13		Ujjal
15	Documentation	25 days	Mon 9/30/13	Fri 11/1/13		Ujjal

4.3.2 Tracking Gantt



4.3.3 Pert Chart



5 SCOPE OF THE SOLUTION

FITNESS TRACKER allows user to manage his/her health regarding activity in Android mobile device and update the information in web server. Show the progress about his/her goal. Not only that, it allows us to add to-do, remainder of various events and synchronize them with web server so that we could track them and maintain properly from anywhere.

6 ANALYSIS

6.1 CONTEXT DIAGRAM

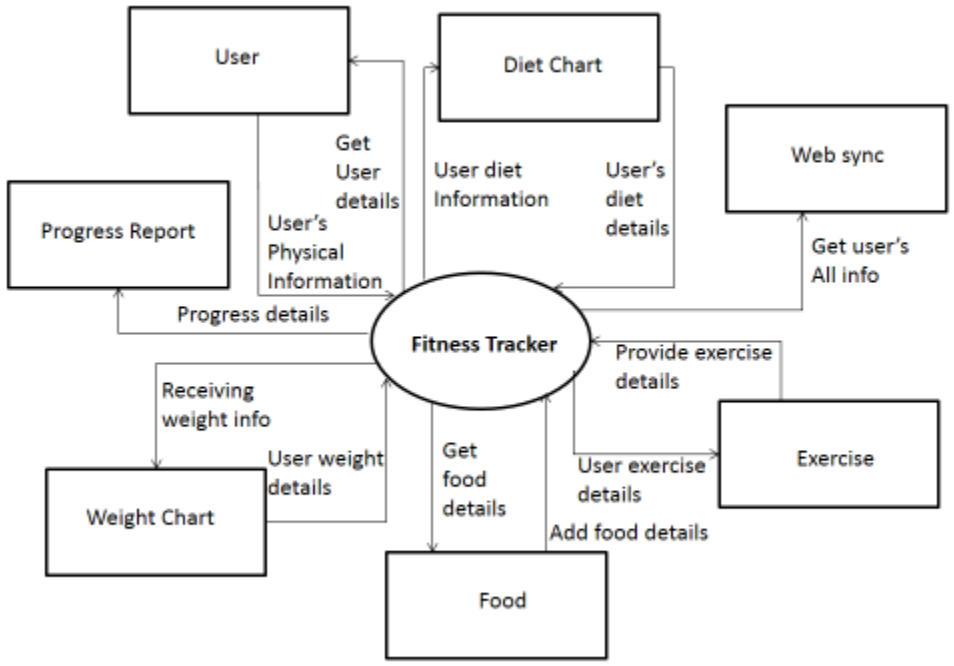


Figure 1:Context Diagram of FTS

6.2 0-LEVEL DFD

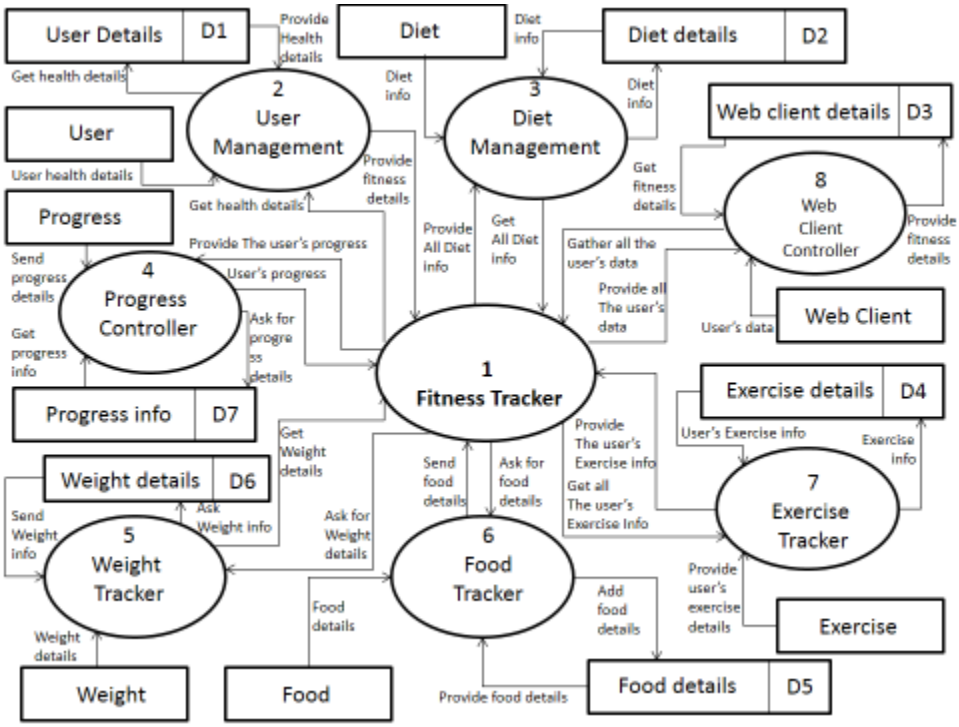


Figure 2:0 level DFD

6.3 1-LEVEL DFD

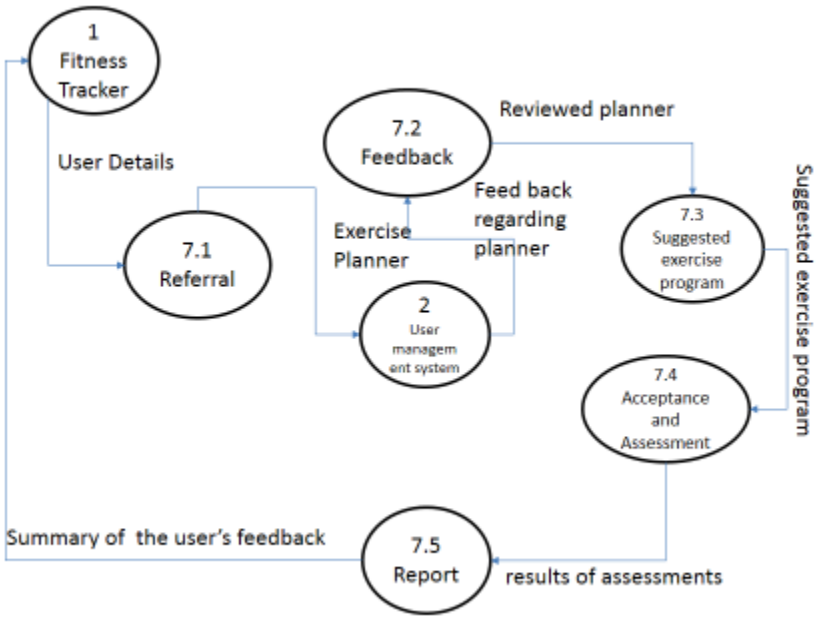


Figure 3:1 level DFD

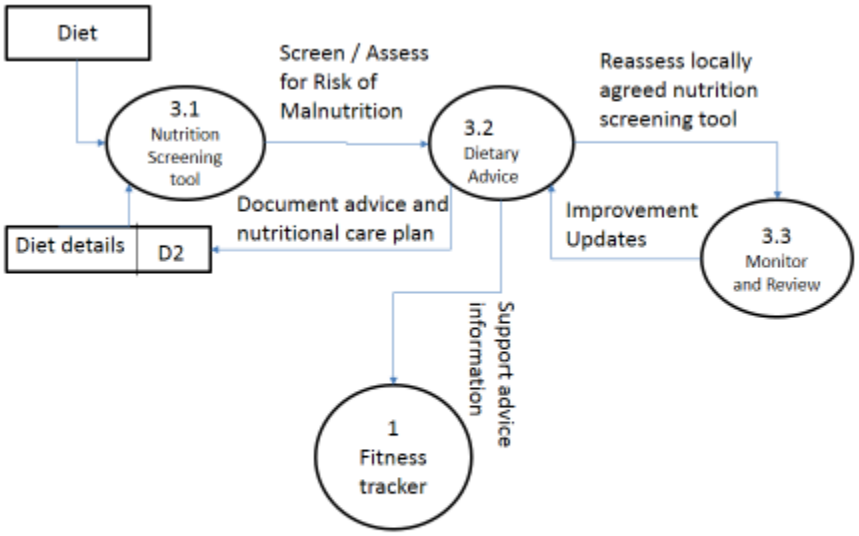


Figure 4:1 level DFD

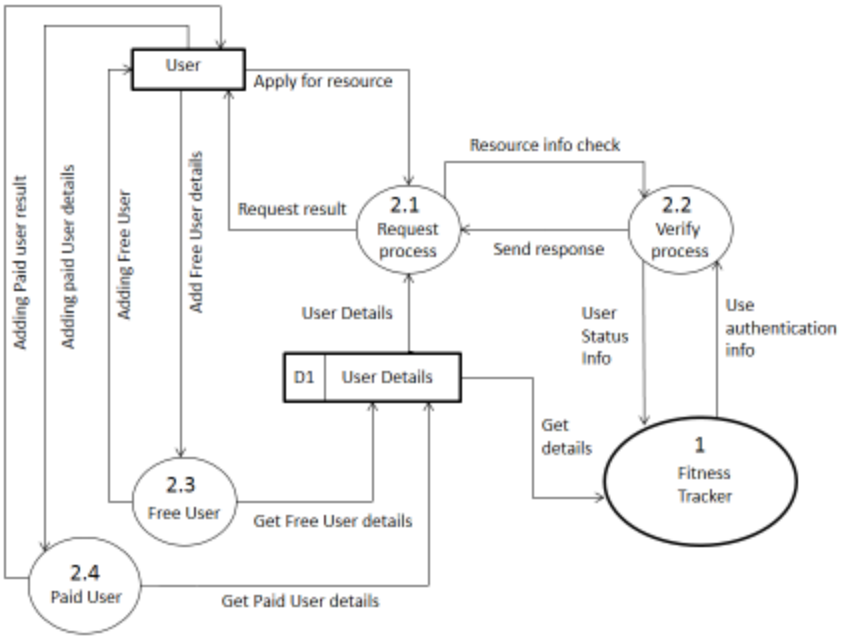


Figure 5:1 level DFD

6.4 2-LEVEL DFD

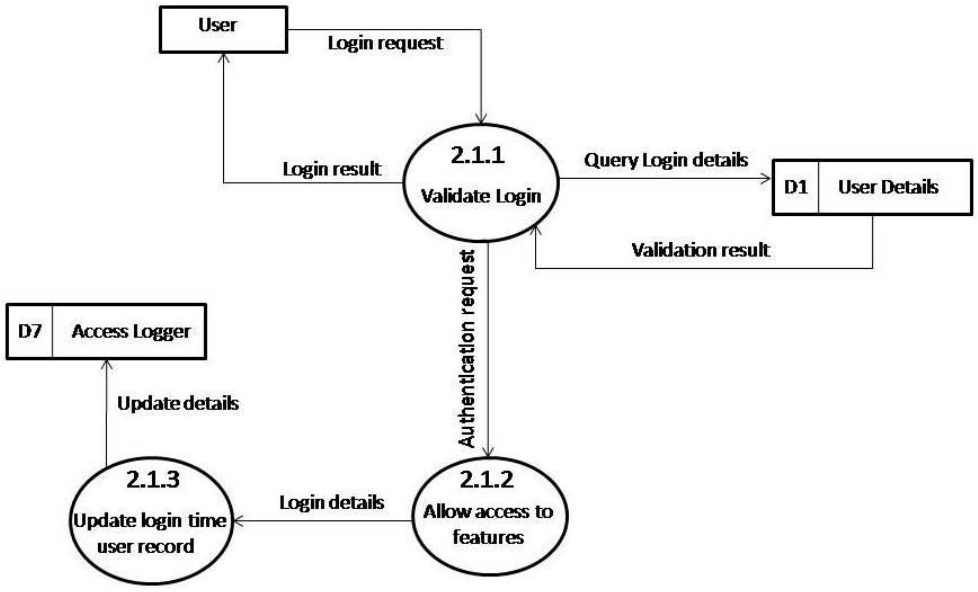


Figure 6:2 level DFD

6.5 E-R DIAGRAM

We will design a RDBMS for ***FITNESS TRACKER***. The entities and their attributes are listed below. Attributes in Bold letter is the unique key.

Entities	**Attributes**
User	**User Id**, Name, Current weight, Goal weight, Date of birth, Gender
New entry	**Entry Id**, Content, Details
To do	**To do id,** task, details, priority
New food	**Food Id,** name, description, serving size, calories

Relationship between Entities:

- FITNESS TRACKER has User → 1 : N
- Users add entries → 1 : N

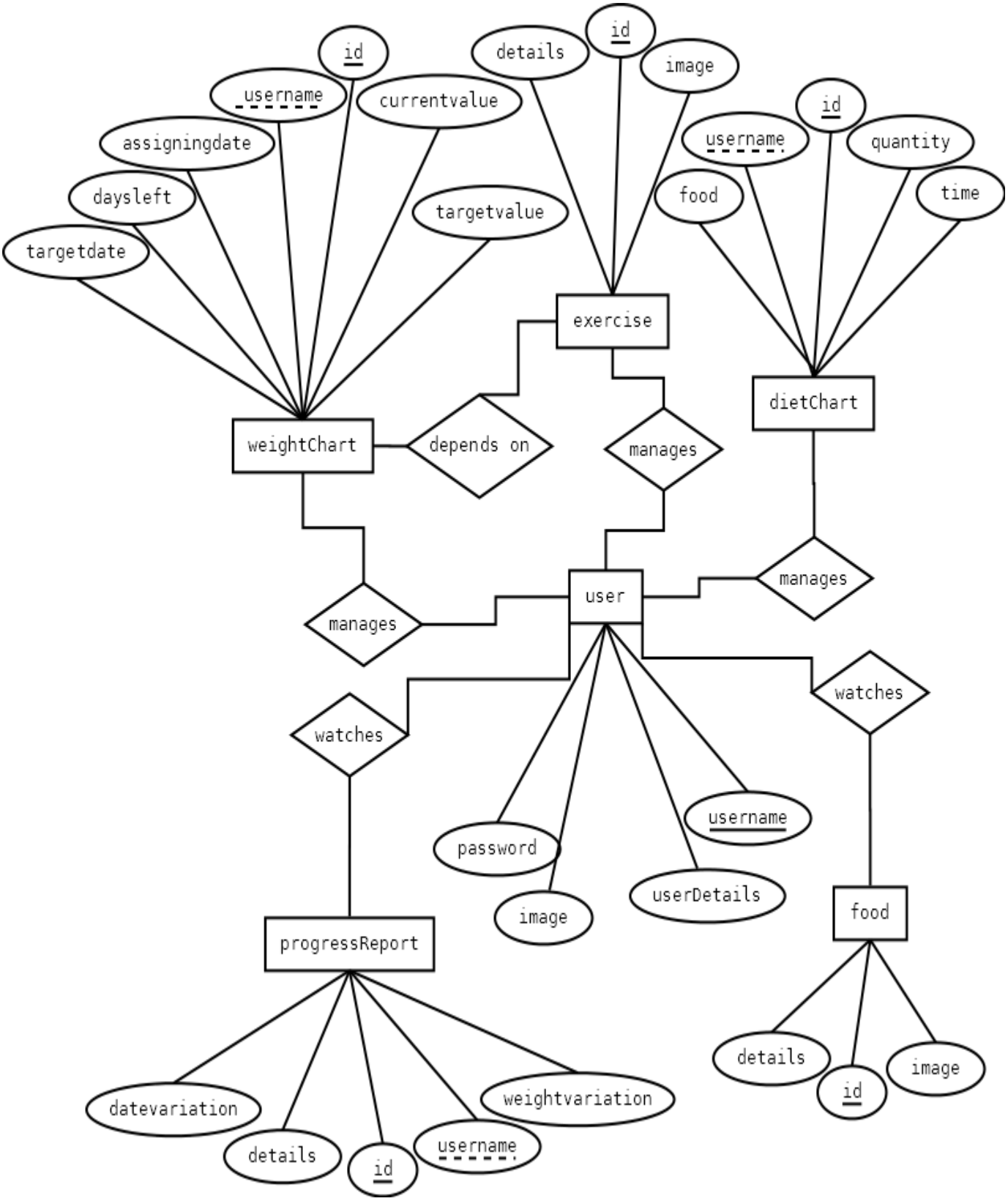
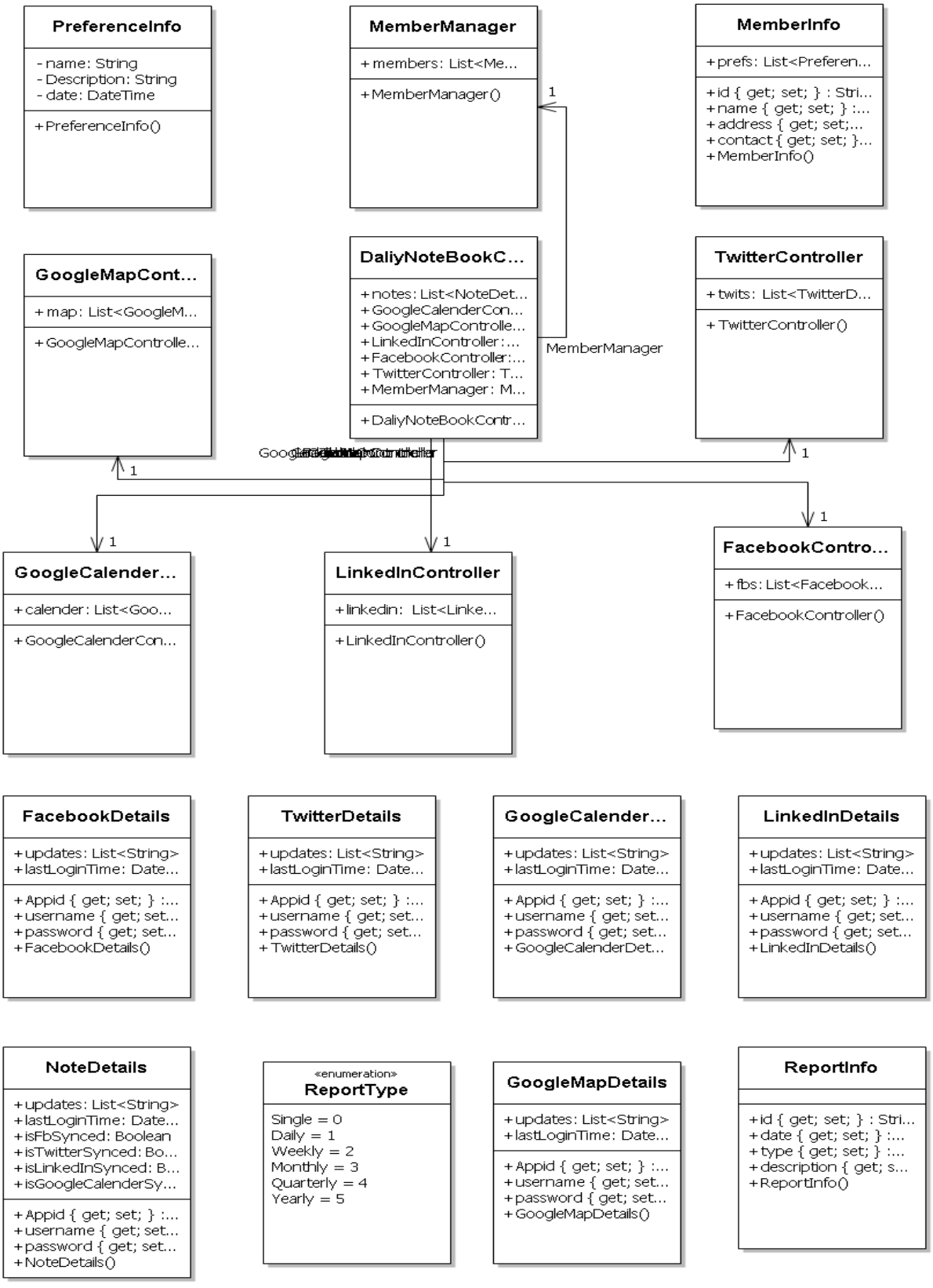


Figure 7:E-R Diagram of FITNESS TRACKER

6.6 CLASS DIAGRAM



7 DATABASE & TABLE DETAILS

The database used for this software is called **fitnesstrackerdb**. Database tables and corresponding keys are shown in tabular form. It shows the tables and its columns. A key in **Bold** is the primary key.

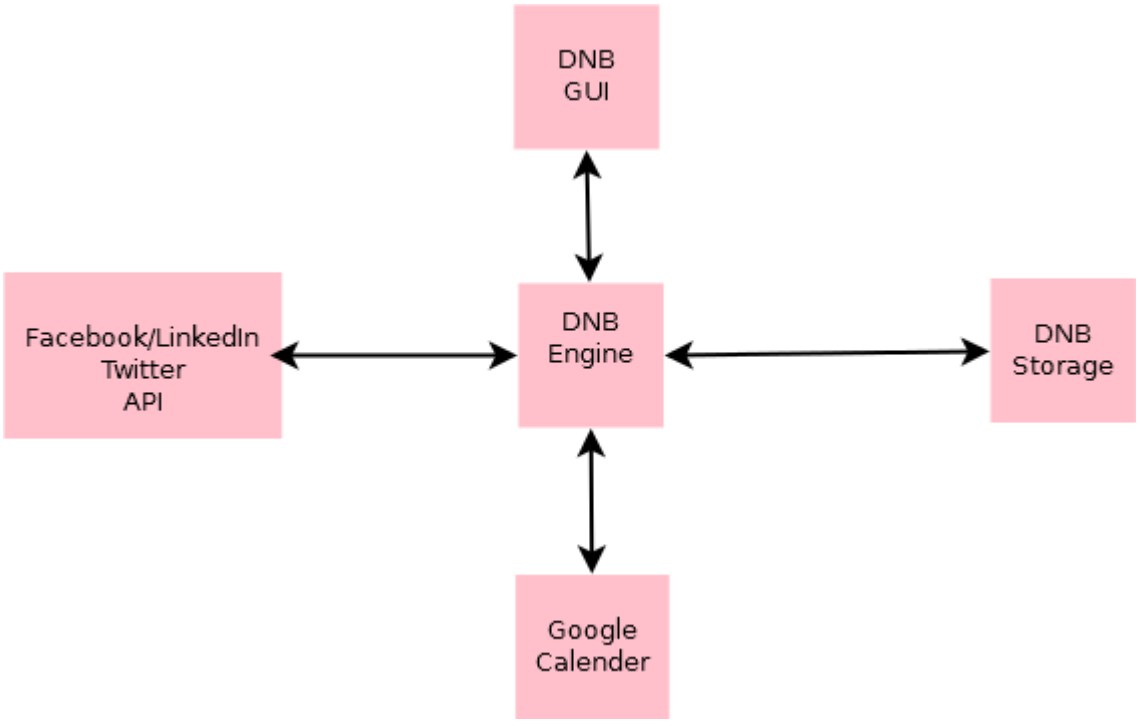
Entities	**Attributes**
User	**User Id**, Name, Current weight, Goal weight, Date of birth, Gender
New entry	**Entry Id**, Content, Details
To do	**To do id,** task, details, priority
New food	**Food Id,** name, description, serving size, calories

8 COMPLETE STRUCTURE

8.1 MODULE DESCRIPTION

The above picture represents various modules of ***FITNESS TRACKER***. Their detailed description is written below:

FITNESS TRACKER Engine: This module handles all the logical parts of ***FITNESS TRACKER***. It takes data from user through ***FITNESS TRACKER*** GUI module and stores them to SQLite database using ***FITNESS TRACKER*** Storage module. It sends the data to the user's web server account to get the details from web site.



FITNESS TRACKER GUI: This part is the place through which user interacts. This module contains all the designs which are visible and intractable by users. User provides input through it and gets the output through it. It is generally created by various tools like Buttons, EditText etc.

FITNESS TRACKER Storage: In this module all the data are stored. ***FITNESS TRACKER*** Engine stores data in this module and fetches them for output through this module.

FITNESS TRACKER *web site*: This place gets input from the ***FITNESS TRACKER*** Storage. All the relevant data sent by user to server, which could be accessed by user globally.

8.2 ESTIMATION

Task Usage

	Task Name	Work	Duration	Start	Finish
1	Daily Note Book Desktop Client	40 hrs	5 days	Mon 10/1/12	Fri 10/5/12
	Anirban	*40 hrs*		*Mon 10/1/12*	*Fri 10/5/12*
2	Daily Note Book Controller	100 hrs	12.5 days	Mon 10/8/12	Wed 10/24/12
	Anirban	*100 hrs*		*Mon 10/8/12*	*Wed 10/24/12*
3	Notebook Controller	40 hrs	5 days	Wed 10/24/12	Wed 10/31/12
	Anirban	*40 hrs*		*Wed 10/24/12*	*Wed 10/31/12*
4	User Management system	40 hrs	5 days	Wed 10/31/12	Wed 11/7/12
	Anirban	*40 hrs*		*Wed 10/31/12*	*Wed 11/7/12*
5	Contacts Management System	70 hrs	8.75 days	Mon 11/5/12	Thu 11/15/12
	Anirban	*70 hrs*		*Mon 11/5/12*	*Thu 11/15/12*
6	To do Management System	60 hrs	7.5 days	Thu 11/15/12	Mon 11/26/12
	Anirban	*60 hrs*		*Thu 11/15/12*	*Mon 11/26/12*
7	Google Map Controller	40 hrs	5 days	Mon 11/26/12	Mon 12/3/12
	Anirban	*40 hrs*		*Mon 11/26/12*	*Mon 12/3/12*
8	Facebook Handler	40 hrs	5 days	Mon 12/3/12	Mon 12/10/12
	Anirban	*40 hrs*		*Mon 12/3/12*	*Mon 12/10/12*
9	Twitter Handler	40 hrs	5 days	Thu 12/6/12	Wed 12/12/12
	Anirban	*40 hrs*		*Thu 12/6/12*	*Wed 12/12/12*
10	LinkedIn Handler	48 hrs	6 days	Tue 12/11/12	Tue 12/18/12
	Anirban	*48 hrs*		*Tue 12/11/12*	*Tue 12/18/12*
11	DNB Database	80 hrs	10 days	Tue 12/18/12	Mon 12/31/12
	Anirban	*80 hrs*		*Tue 12/18/12*	*Mon 12/31/12*
12	Integration	80 hrs	10 days	Tue 1/1/13	Mon 1/14/13
	Anirban	*80 hrs*		*Tue 1/1/13*	*Mon 1/14/13*
13	Integration Testing	100 hrs	12.5 days	Tue 1/15/13	Thu 1/31/13
	Anirban	*100 hrs*		*Tue 1/15/13*	*Thu 1/31/13*
14	Bug Fix	180 hrs	22.5 days	Fri 2/1/13	Tue 3/5/13
	Anirban	*180 hrs*		*Fri 2/1/13*	*Tue 3/5/13*
15	Documentation	50 hrs	6.25 days	Thu 2/28/13	Fri 3/8/13
	Anirban	*50 hrs*		*Thu 2/28/13*	*Fri 3/8/13*

8.3 DATA STRUCTURE

LinkedInDetails

```
public class LinkedInDetails
    {
        public string Appid { get; set; }
        public string username { get; set; }
        public string password { get; set; }
        public List<string> updates;
        public DateTime lastLoginTime;
    }
```

GoogleCalenderDetails

```
public class GoogleCalenderDetails
    {
        public string Appid { get; set; }
        public string username { get; set; }
        public string password { get; set; }
        public List<string> updates;
        public DateTime lastLoginTime;
    }
```

TwitterDetails

```
public class TwitterDetails
    {
        public string Appid { get; set; }
        public string username { get; set; }
        public string password { get; set; }
        public List<string> updates;
        public DateTime lastLoginTime;
    }
```

FacebookDetails

```
public class FacebookDetails
    {
        public string Appid { get; set; }
        public string username { get; set; }
        public string password { get; set; }
        public List<string> updates;
        public DateTime lastLoginTime;
    }
```

GoogleMapDetails

```
  public class GoogleMapDetails
    {
        public string Appid { get; set; }
        public string username { get; set; }
```

```
        public string password { get; set; }
        public List<string> updates;
        public DateTime lastLoginTime;
    }
```

NoteDetails

```
public class NoteDetails
    {
        public string Appid { get; set; }
        public string username { get; set; }
        public string password { get; set; }
        public List<string> updates;
        public DateTime lastLoginTime;
        public bool isFbSynced;
        public bool isTwitterSynced;
        public bool isLinkedInSynced;
        public bool isGoogleCalenderSynced;
    }
```

FacebookController

```
public class FacebookController
    {
        public List<FacebookDetails> fbs;
    }
```

TwitterController

```
public class TwitterController
    {
        public List<TwitterDetails> twits;
    }
```

LinkedInController

```
public class LinkedInController
    {
        public List<LinkedInDetails> linkedin;
    }
```

GoogleMapController

```
public class GoogleMapController
    {
        public List<GoogleMapDetails> map;
    }
```

GoogleCalenderController

```
public class GoogleCalenderController
    {
```

```
        public List<GoogleCalenderDetails> calender;
    }
```

ReportType

```
public enum ReportType
    {
        Single,
        Daily,
        Weekly,
        Monthly,
        Quarterly,
        Yearly
    }
```

ReportInfo

```
public class ReportInfo
    {
        public string id { get; set; }
        public DateTime date { get; set; }
        public ReportType type { get; set; }
        public string description { get; set; }
    }
```

DaliyNoteBookController

```
  public class DaliyNoteBookController
    {
        public List<NoteDetails> notes;
        public GoogleCalenderController GoogleCalenderController;
        public GoogleMapController GoogleMapController;
        public LinkedInController LinkedInController;
        public FacebookController FacebookController;
        public TwitterController TwitterController;
        public MemberManager MemberManager;
    }
```

MemberInfo

```
public class MemberInfo
    {
        public string id { get; set; }
        public string name { get; set; }
        public string address { get; set; }
        public string contact { get; set; }
        public List<PreferenceInfo> prefs;
    }
```

MemberManager

```
public class MemberManager
    {
```

```
        public List<MemberInfo> members;
    }
```

```
PreferenceInfo
```

```
  public class PreferenceInfo
    {
        string name;
        string Description;
        DateTime date;
    }
```

8.4 Implementation Methodology

- Android SDK has been used to develop the application.
- Object Oriented Programming methodology will be adopted
- Relational DBMS SQLite will be used to implement & execute SQL query to database for Android device.
- Relational DBMS MySQL will be used to implement & execute SQL query to database for Web site.
- Agile Software Development model will be used while developing this software.

8.5 List of Reports

- List of Activity could be generated.
- List of New food details could be generated.
- A list of Task could be generated.
- List of Progress could be generated.

9 Implementation of Security Mechanism at Various Levels

The software requires a predefined username and password to login.

It encrypts the data stored in the database so that even if someone succeeds to hack the database still not much harm could be done.

The application will use Google open-id authentication for web interface.

10 FUTURE SCOPE AND FURTHER ENHANCEMENT OF THE PROJECT:

- To Support Mobile Operating systems for Symbian, Meego, Firefox, Sailfish OS & Windows.
- To support UNIX / Linux Based Operating systems.

11 BIBLIOGRAPHY

- *http://en.wikipedia.org*
- *http://www.codeplex.com/*
- *http://stackoverflow.com/*
- *http://www.codeguru.com/*
- *http://www.w3schools.com*
- *www.mysql.org*
- *http://twitter.github.io/bootstrap/index.html*
- *http://developer.android.com/training/basics/firstapp/index.html*
- *http://ellislab.com/codeigniter/user-guide/*

---------- Thank You----------