Report No. 3
FINAL REPORT

The Talking Heart

What's your heart saying?

A Personal Health Monitoring & Diagnosis System

Software Engineering I

12/15/2012

Development Team #2
Prasoon Mishra
Vinayak Pothineni
Bhumika Singh
Jay Takle
Anu Liz Tom
Anusha Vutukuri

Table of Contents

Summary of Changes:

1. System Sequence Diagrams
2. Customer User Effort Estimation
3. Domain Model
4. Concept Definitions
5. Associative Definitions
6. Attribute Definitions
7. Interaction Diagrams
8. System Class Diagram
9. User Interface Design and Implementation
10. Design of Test Cases

Individual Contributions

“All team members contributed equally”

1 Customer Statement of Requirements:

As mentioned in the individual contribution section, we enacted a customer representative v/s user conversation and noted down the problems and expectation. The problem statement and following paragraphs are thus written from the perspective of a customer to get a feel of problems faced by them and their expectations from a product.

1. 1 The sudden need for a Personal System

The most widely accepted way of maintaining our body has been regular checkups with the doctor. Our vital statistics like heart rate, calories burned, pulse rate were taken onto various machines. The nurses would convert these signals into comprehensible format and the doctors would study them. Later these were monitored daily or at regular intervals and doctors would review them and give there diagnosis. This has been a traditional process, but being always on the move and working a hectic schedule does not give everyone a lot of options as far as taking appointments with doctors is concerned. It would be so much better if people could have a doctor at home whenever they come back from home, or a doctor who can tell them their body's progress anytime they want.

The other factor that has started taking priority in today's internet era is global accessibility to a common place where people can store their records. As a patient or an athlete people's health records are scattered across many different providers and facilities. Keeping updates and easily accessible health records means people can play a more active role in their own and their family's healthcare concerns. It also gives a clear picture of their health history, anytime anywhere.

To make the requirements about the system more specific, following is a high level description of the features a modern day health monitoring system should have from an end user perspective:

1.1.1 *Global Accessibility*

Every working individual has to travel to different states and sometimes to a different country as part of their job. If they were to take care of their health and thus keep a record of their check-ups, they would have to carry a paper diary everywhere. This will not only be cumbersome but also a very approximate way of keeping records. We would like to provide customers with a 24x7 access to their medical records from wherever they need it, whenever they need it.

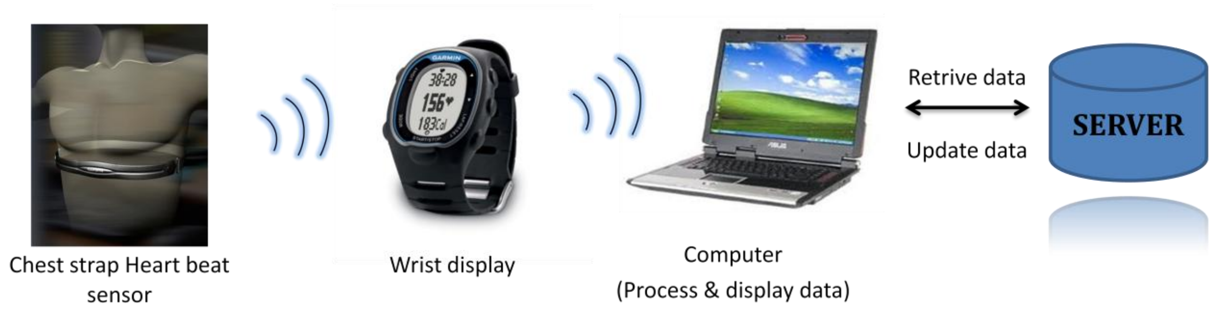


Figure 1.1: Our system's architecture for personal health monitoring.

We plan to build an online system that will store, process and diagnose a person's heart rate and calories burned readings from a heart rate sensor and pedometer device. Providing a web based system will permit users to access their data from any location.

Figure 1 shows the architecture of how our system would work. Our system will comprise of two parts: the health monitoring device, a Garmin FR 70 wrist display with a heart rate sensor and the other part would be an online software system that is connected to a server to store and retrieve customer records.

Let us see how the customer would have to record their data. The user has to fix the chest strap with the mounted sensor on their ribcage. Next wear their wrist display which accepts continuous heart rate readings transmitted by the chest strap. The user can look at instantaneous heart ratings on the wrist display.

As far as the optimum time for recording heart rate is concerned, the heart rate changes with posture, varying by around 3 beats/min in the sitting compared with the supine position [10]. A recent consensus meeting recommended measurement of the hear rate by pulse palpation during two 30-s periods, performed in a sitting position, after 5 min sitting in a quiet room [10]. For our system to provide accurate diagnosis, we recommend our users to record their heart rate using the heart rate strap for at least one minute. Once the chest strap has transmitted readings to the watch for one whole minute, the user can transfer data to the computer wirelessly from the watch. Now the data is on the computer and ready to tell you about your health.

1.1.2 *Scalability to Numerous Users*

When it comes to health care of the entire family, either of the parents keep a record of the whole family's medical history. This should also be applied to a framework wherein an electronic device records body statistics and transfers to a computer for storage. Instead of buying multiple hear rate sensors and wrist displays one for every family member, it would be economical to buy just one heat rate sensor and wrist display for the whole family. The whole family can use the sensor and wrist display in turns.

We will build our systems on the above points by letting users create their profiles. This way a user can record their data on the wrist device, transfer it to the computer, login to their profile through our system and upload their data. Now that their data has been saved in our server, some other member of the family can use the device to record and upload their data to their profile in the same way.

The system will also allow the user to edit personal profile for e.g. weight or height, which would be necessary for processing and diagnosing their data.

1.1.3 *Ease of Uploading Data*

Once a user record's their body readings using the sensors and devices, they would really like to save all the records till date. This will help them to see if there are any drastic changes in their body's behavior. Although today's sensors come with wrist displays that have memory to store recorded data, they can at the most store 10-20 work out worth of records at a time. Once above the limit, the data has to be cleared for new data to be recorded by the sensors.

Users also want to store all the data at one place that can be accessed later. Although this is possible by manually typing in the information from the wrist display into a computer, it will take time and will definitely act as a deterrent to anyone who feels this is too much work. What would be really lucrative is a system that can store data by simply selecting a file from the computer.

Our proposed system will give this option to the user for easy data loading. The Garmin FR 70 gives an option to store a workout or reading data in a single file in the computer. Using our software system users can log into their profile, select this file and their data will be uploaded. Just before uploading, our system will prompt the user to specific whether this file is for resting heart rate (which would be for a minute as mentioned in section 1.1.1) or if it is for an exercise or run (which can be for a longer duration).

1.1.4 *Ease of Finding and Viewing Records*

If people go to get checked up at the hospital, body readings for different days are kept in different files or at least recorded on separate papers. The obvious advantage to this is that people can look at their records for each day in detail, separately.

Our system will have a similar property, something similar to a diary. Through our proposed design users will be able to look for their medical records for a particular day. The advantage of this design is that people can see if there were multiple uploads for a day and they can also add their own comments or reminders. Each user can upload more than one activity everyday to their profile. As we have mentioned in section 1.1.3, the user would have to specify if the data file being uploaded is for resting heart rate or for an exercise.

Thus by giving this feature of multiple file uploads for a day, the user can store data for exercise and heart rate on the same day. In addition they can add multiple exercise data files in one day and write a comment specifying whether it is weight lifting, running etc. Then, if a user wants to see a graph of that day's personal readings, they would just have to click on the day and our system will display readings in a graph format.

1.1.5 *Graphical View of Progress*

People say 'a picture is worth a thousand words' and that is exactly what we want our customers to experience from our system. Instead of just showing a couple of numbers our system would show readings like heart rate, calories burned, stress levels against time in form of a graph. A graphical view will definitely show how their heart rate, stress or calories burned vary over time.

But we would like to provide more to our customers by giving an option to see an overview of say every week or month of readings in the form of graphs. This could help them know how their health conditions differ for different months. Showing the data in form of a graph makes it easy to comprehend what the numbers mean. In the second iteration of system development we would try to add an interpretation of the graph in form of text. For example if a user's heart rate is increasing over a month, we would add a text saying "Heart rate increased".

1.1.6 *What your Heart wants to tell*

With the amount of work load stacking up everyday people rarely get to eat proper lunch, let alone taking a break and going out for a walk. People just feel the stress building up but don’t really know if it is detrimental or if it is going to cause long term problems. They don’t know if this hectic schedule is causing any damage to the way their heart should work. Even if people use a wrist display and heart sensor, it just shows them their heart rate for that instance. We would like to show our customers two things, first of all how their readings progress over a time interval and then what it means for them, health wise.

In a future document about implementation during the second iteration of system development, we will discuss and explain in detail how we plan to implement in our software system the different methods for heart disease diagnosis, but in the following paragraphs we give an idea of how we plan to use heart rate readings to diagnose specifically three heart conditions and stress levels.

Various epidemiological papers have studied the importance of heart rate for healthy living [2]. The relationship between resting heart rate and mortality of patients with coronary artery disease, hypertension and in the elderly has been observed by a lot of scientists [3][4][5][6][7]. Resting heart rate is a very simple but important measurement of the health of a person’s heart. The resting heart rate is measured while a patient is awake but not taking part in any kind of strenuous activity. We discuss some very serious implications of different resting heart rates for different kind of people.

Tachycardia:

Tachycardia refers to a heart rate that is over the normal range for a particular age group. When the heart beats at a faster rate, it works inefficiently to provide blood to the rest of the body, thus providing less blood at a given time [8].

We will have a table of normal heart rate range for every age group coded in our system. Thus we will compare a person’s resting heart rate reading with their age (from their user profile) and compare with the table to assess if they are suffering from Tachycardia.

Bradycardia:

When the resting heart rate is lower than usual for a person (less than 50bpm for adults), the heart problem is called as bradycardia [9]. The heart rate going below this level may cause a cardiac arrest in some people, since a lower heart rate means less blood and oxygen being pumped to the heart.

We will implement the above mentioned fact to compare a person’s averaged resting heart rate with the threshold of 50bpm to infer if they might be suffering from Bradycardia.

Cardiovascular mortality:

Recent studies have confirmed the fact that resting heart rate is an independent predictor of cardiovascular mortality in men and women with and without any diagnosed chronic disease [10]. In a study of 25,000 patients with cardiac diseases, resting heart rate has been found to be a strong indicator of a person’s cardiac health [11]. It was found that patients with a resting heart rate of greater than 83bpm are prone to rehospitalization as compared with patients with a resting heart rate of less than 62bpm.

We plan to implement this feature specially for users who have had a heart attack in past. While filling the user profile, we can provide a check box for "Experienced Cardiac Attack in past". If the user check's this box, we will make sure that their hear rate readings are cross checked against the 62bpm to 83bpm range, as studied in the [11].

For all the three cases of tachycardia, bradycardia and cardiovascular mortality, we plan to provide a single button in our software system, which users can click to get results (more details in design document). When the user clicks this button, the system will compare their resting heart rate readings for the above mentioned criterions and display on the screen whether the user's heart is normal or whether it is suffering from tachycardia, bradycardia or cardiovascular mortality. This will be displayed in the form of simple text so that it is easy for the user to comprehend. Users do not need any special knowledge or skills about heart rate variations, our system will do all the calculations for them.

Stress:
The autonomic nervous system (ANS) is part of the nervous system whose task is to maintain the body by acting like a controlling mechanism. The ANS is made up of two functional components: the sympathetic nervous system (SNS) and the parasympathetic nervous system (PNS). The SNS, accompanied by an increase in heart rate helps the body to combat potential threats by maintaining the body (while exercising). The PNS branch on the other hand helps in bringing the body back to recovery state with a reduced heart rate (while sleeping). Both these components and in turn the brain, simulates the sinoatrial node which is the primary pacemaker of the heart. So we can use the heart rate to estimate the activation level of both the components. But not the direct heart rate, we are more interested in the R-R interval or the inverse of the heart rate, also called as the heart rate variability (HRV). For a healthy human, the HRV should be chaotic, varying randomly over time [12]. Scientists have thus found the heart rate and heart rate variability recordings as good indicators to measure stress levels and guide preventive measures to reduce stress in a person [13].

Thus by calculating the inverse of a user's heart rate, our system will scan this quantity for a time interval to make sure it is random, as studied in [12][13]. This inverse heart rate for a user also known as Heart Rate Variability, can be viewed as a graph in a similar fashion as in figure 4.

In summary from the above studies we can conclude that a whole lot of information about the condition of our body can be gained from just the resting heart rate.

2. Glossary of Terms

TERM	DESCRIPTION
Heart rate	It is the number of heartbeats per unit of time, typically expressed as beats per minute (bpm).
Diagnose	It is the identification of the nature and cause of anything
Resting heart rate	It is measured while a patient is awake but not taking part in any kind of strenuous activity
Tachycardia	It refers to a heart rate that is over the normal range for a particular age group.
Bradycardia	The heart condition when the resting heart rate is lower than usual for a person (less than 50bpm for adults)
Autonomic Nervous System (ANS)	It is part of the nervous system whose task is to maintain the body by acting like a controlling mechanism.
Sympathetic Nervous System (SNS)	It is a component of ANS, accompanied by an increase in heart rate helps the body to combat potential threats by maintaining the body.
Parasympathetic Nervous System (PNS)	It is a component of ANS which helps in bringing the body back to recovery state with a reduced heart rate.
Heart Rate Variability (HRV)	Inverse of the heart rate.
Pedometer	A pedometer[15] is a device, usually portable and electronic or electromechanical, that counts each step a person takes by detecting the motion of the person's hips.

3 System Requirements

3.1 Enumerated Functional Requirements:

Requirements	PW	Description
REQ - 1	3	The system shall allow the user to register using a unique user id and password.
REQ – 2	5	The system shall allow any user to load their data from any computer having the data file.
REQ – 3	4	The system shall allow any user to modify their personal profile.
REQ – 4	2	The system shall prompt for security questions if the user forgets their password.
REQ – 5	5	The system shall allow the user to see their daily and monthly heart rate, calories burnt and stress level.
REQ – 6	5	The system shall display the heart condition as text, which represent the condition of the heart for the selected entry which is in the range of 60-70 seconds.
REQ-7	3	The system should allow the administrator to retrieve a user password.

3.2 Extended Functional Requirements:

Requirements	Description
REQ – 2a	The system shall allow the user to upload only .csv file produced by the Garmin FR70. This file will contain heart rate reading and calories burnt.
REQ – 2b	The system will allow user to upload a file either as an exercise entry or a resting heart entry.
REQ – 4a	The system shall allow only the user with correct login Id and password to access their profile.
REQ – 4b	The system shall logout from a session if there is inactivity for more than ten minutes.
REQ – 4c	The system shall lock the profile if the user enters wrong password more than 3 times in a row. The user will need to send a mail to the administrator to ask him to unlock his profile.
REQ – 4d	The system shall give a security question option to the user if they forget their password. If the user gives the correct answer they will be able to see his password.

REQ – 4e	The system shall allow the user to delete their profile.
REQ – 5a	The system shall allow the user to view all entries for a day and individual graphs will be shown for calories burnt, heart rate and stress level.
REQ – 5b	If there is more than one entry for a day, the system shall display graphs of all entries.
REQ – 5c	When requested, the system shall display the monthly graph for any of the parameters mentioned above (i.e. calories burnt, heart rate and stress level.)
REQ – 5d	The system shall display the yearly timeline.

3.3 Non-functional Requirements:

Requirements	PW	Description
REQ – 7	3	Web pages shall have simple design for easy navigation.
REQ – 8	2	System shall have a single link on every page to navigate back to Calendar View.
REQ – 9	4	The user shall not be able to see other person’s data or make any changes to it.
REQ – 10	3	Graphs shall be labeled properly and extensively.

3.4 On-Screen Appearance Requirement

Figure 2 shows the initial sketch of the user interface expectation sent by the customer. The user interface will show the 'personal information' on every page. The 'viewing options' pane will change according to the current graph being viewed. The customer asked for easy navigation for which we have suggested a calendar style display (shown in the User Interface section).

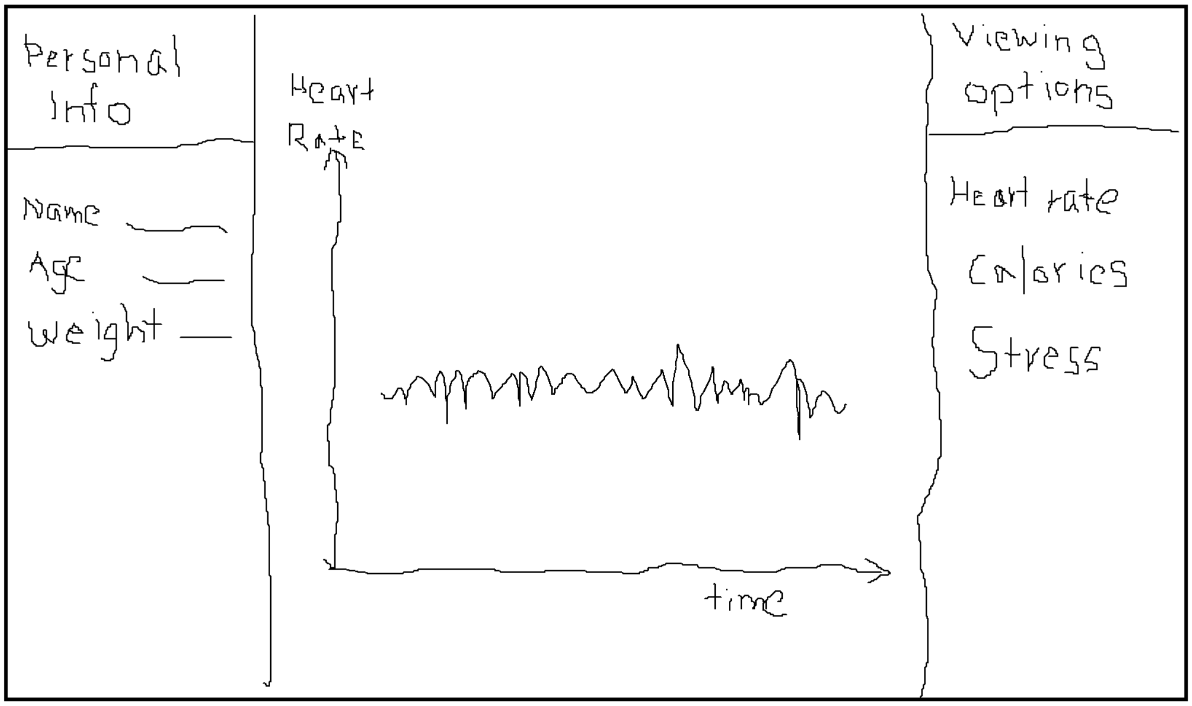


Figure 3.1: Sketch provided by customer.

4 Functional Requirements Specification:

A personal health monitoring & diagnosis system can be very useful, though the creation of such a system requires good effort. Though this system is designed with patients in mind, it can be used by everybody who wants to keep track of their health condition. Anyone with proper credentials can access this system.

4.1 Stakeholders

Our system being a health monitoring system there can be a lot of stakeholders. Everyone is interested in knowing their health. We have enumerated stakeholders for our system on the basis of how often someone might use the system.

1. Patients
   The motivation for building this system was to provide patients conclusions or diagnosis of their health recording. Thus we feel and hope that patients would be the large group of stakeholders using our system.

2. Athletes
   We know that performance enhancement for an athlete requires tuning their body. Using our system athletes can store the whole history of their heart rate and calories burned. Because our system will display these quantities graphically, they can view their readings in form of a progress curve (telling them whether their hear rate/calories burned is going up or down as time progresses). Athletes can also check if they suffer from either Tachycardia or bradycardia.

3. Coaches
   A sports team coach has to keep track of every individual’s health in that team. Our system could be a perfect solution for such a need. The coach would have a direct access to each player’s records.

4. Doctors
   Doctors can use our system as a spot to store and access data for different patients. Although we feel this could be cumbersome for the doctor since he/she would have to create a user profile for each patient.

4.2 Actors and Goals

We have identified three actors interacting in the working of our system.

Actor	**Goals**
User	The user’s goal is to access the personal health monitoring system to keep track of health condition. Includes all the people especially patients, athletes who require regular health check-ups.

Administrator	The goal of administrator is to retrieve the password for the user.
Database	The goal of a database is to store all the readings of each user separately. All the data collected is stored in the database and the user interface has to access the database to retrieve the required information.

4.3 Use Cases Casual Description

Use Case	Name	Description
UC1	CreateProfile	Allows a user to create a new profile
UC2	Login	Allows the user to access the system.
UC3	UploadFile	Allows the user to upload a data file to system from a computer.
UC4	ChangeProfile	Allows the user to modify their personal information.
UC5	ViewDailyHeartRate	Allows the user to view the heart rate readings for a day in the form of a line graph.
UC6	ViewDailyCaloriesBurnt	Allows the user to view the calories burnt in a day in the form of a bar graph.
UC7	ViewDailyStressLevel	Allows the user to view the stress levels for a day in the form of a line graph.
UC8	ViewMonthlyHeartRate	Allows the user to view the heart rate readings for a month in the form of a line graph.
UC9	ViewMonthlyCaloriesBurnt	Allows the user to view the calories burnt in a month in the form of a bar graph.
UC10	ViewMonthlyStressLevel	Allows the user to view the stress levels for a month in the form of a line graph.
UC11	ViewHeartCondition	Allows the user to view whether their heart condition is normal, Tachycardia, Bradycardia or cardiac mortality in the form of text.
UC12	CalculateDailyAverageHeartRate	Allows the user to view the average values of heart rate for a day in the form of text.

UC13	DeleteProfile	Allow the user to delete his profile.
UC14	RetrievePassword	Allow the user to send a mail to the administrator to retrieve his password when he fails to enter correct password three times in a row.
UC15	ViewYearlyGraph	Allow the user to view the yearly heart rate.

4.4 Use Case Diagram

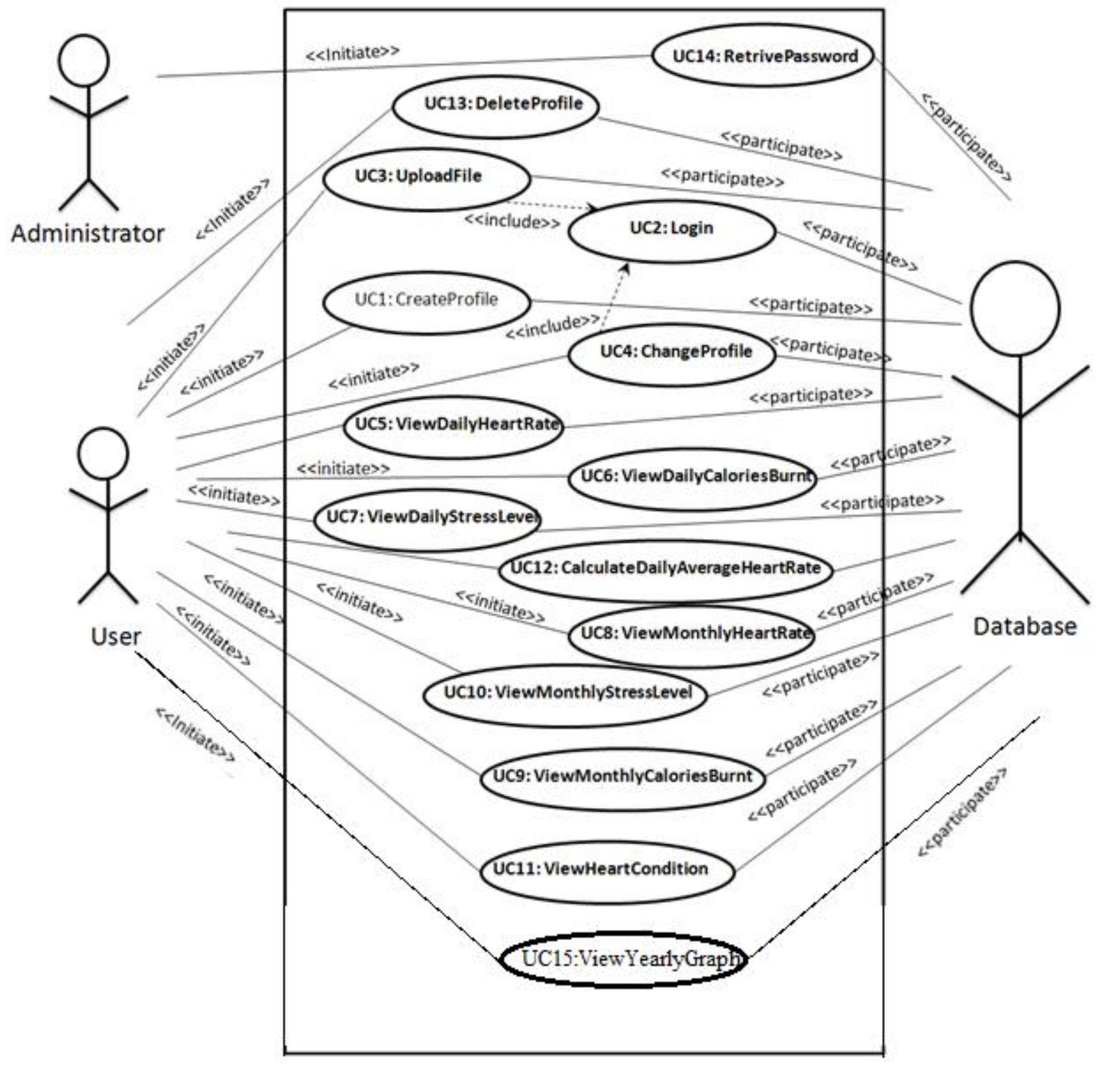


Fig 4.1 Use Case Diagram

4.5 Fully Dressed Description

UC1 CreateProfile
Initiating Actor: User
Actor's Goal: To create a new profile.
Participating Actor: Database.
Pre-conditions: User is viewing the login screen of the system
Post-conditions: System displays the user's current month home screen
Flow of Events for Main Success Scenario:
-> 1. User enters the name, age, weight, height, if previously had heart attack, user id, password and presses submit button.
<- 2. System checks if similar user id already exist in database, stores the user information in the database.
->3. System creates a new page with user's information and displays user's home page on screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User enters the name, age, weight, height, user id, password and presses submit button.
<- 2. System checks if similar user id already exist in database, finds a match.
<- 3. System gives a message "Use different User ID" on screen.

UC2 Login
Initiating Actor: User
Actor's Goal: To gain access to the system
Participating Actor: Database.
Pre-conditions: System displays login screen of the system
Post-conditions: System displays the user's current month home screen
Flow of Events for Main Success Scenario:
-> 1. User enters his id and password.
<- 2. System checks if user id and password match, finds a match.
<- 3. System loads the user's home page on screen.
Flow of Events for Extension (Alternate Scenario):
-> 1 User enters the id and password.
<- 2. System checks for the id and password, finds that there is no such id and password in the database.
<- 3. System gives a message "Wrong User ID or Password" on screen.

UC3 UploadFile
Initiating Actor: User
Actor's Goal: To upload a data file to the system from a computer.
Participating Actors: Database.
Pre-conditions: System displays the user's current month home screen.
Post-conditions: Data for the user specified day is stored in the database.
Flow of Events for Main Success Scenario:

-> 1. User selects a day by clicking on a day, selects whether it is exercise or resting data and then clicks on “Upload” button to upload data file.
<- 2. System displays a pop-up window with windows browser and asks user to select a data file.
-> 3. User selects the data file and presses “OK” on pop-up window.
<- 4. System reads data file and stores data in data base under the user’s profile.
<- 5. System displays an entry on the day on the screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User selects a day by clicking on a day, selects whether it is exercise or resting data and then clicks on “Upload” button to upload data file.
<- 2. System displays a pop-up window with windows browser and asks user to select a data file.
-> 3. User selects the data file and presses “Cancel” on pop-up window.
<-4. System does not upload any data.

UC4 ChangeProfile
Initiating Actor: User
Actor’s Goal: To modify their personal information.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen with existing user information.
Post-conditions: System displays the user’s current month home screen with updated user information**.**
Flow of Events for Main Success Scenario:
-> 1. User clicks on “Edit info”.
<- 2. System checks for user information with database.
-> 3. Database sends user information to system.
<- 4. System displays user information with editing option.
-> 5. User type new information and clicks “Save”.
<- 6. System sends new information to database.
->7. Database updates information.
<-8. System displays new information.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on “Edit info”.
<- 2. System checks for user information with database.
-> 3. Database sends user information to system.
<- 4. System displays user information with editing option.
-> 5. User type new information but clicks “Cancel”.
<-6. System does not make any changes with the database and displays old information only.

UC5 ViewDailyHeartRate
Initiating Actor: User
Actor’s Goal: To view the heart rate readings for a day in the form of a line graph.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen.
Post-conditions: System displays the heart rate data for a day in form of a line graph.

Flow of Events for Main Success Scenario:
-> 1. User clicks on a day and then clicks on “View Daily Heart Graph”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the day and sends data to system.
<-4. System converts data to line graph and displays on the screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on a day and then clicks on “View Daily Heart Graph”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the day but does not find data.
<-4. System displays message “No data uploaded”.

UC6 ViewDailyCaloriesBurnt
Initiating Actor: User
Actor’s Goal: To view the calories burnt in a day in the form of a bar graph.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen
Post-conditions: System displays the calories burned data for a day in form of a bar graph.
Flow of Events for Main Success Scenario:
-> 1. User clicks on a day and then clicks on “View Daily Calories Burned”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the day and sends data to system.
<-4. System converts data to bar graph and displays on the screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on a day and then clicks on “View Daily Heart Graph”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the day but does not find data.
<-4. System displays message “No data uploaded”.

UC7 ViewDailyStressLevel
Initiating Actor: User
Actor’s Goal: To view the stress levels for a day in the form of a line graph.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen.
Post-conditions: System displays the stress levels for a day in form of a line graph.
Flow of Events for Main Success Scenario:
-> 1. User clicks on a day and then clicks on “View Daily Stress Levels”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the day and sends data to system.
<-4. System converts data to line graph and displays on the screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on a day and then clicks on “View Daily Stress Levels”.
<- 2. System sends request to database for the data.

->3. Database searches for user’s data for the day but does not find data.
<-4. System displays message “No data uploaded”.

UC8 ViewMonthlyHeartRate
Initiating Actor: User
Actor’s Goal: To view the heart rate readings for a month in the form of a line graph.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen.
Post-conditions: System displays the heart rate data for a month in form of a line graph.
Flow of Events for Main Success Scenario:
-> 1. User clicks on “View Monthly Heart Rate”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month and sends data to system.
<-4. System converts data to line graph and displays on the screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on “View Monthly Heart Rate”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month but does not find data.
<-4. System displays message “No data uploaded”.

UC9 ViewMonthlyCaloriesBurnt
Initiating Actor: User
Actor’s Goal: To view the calories burnt in a month in the form of a bar graph.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen.
Post-conditions: System displays the calories burned for a month in form of a bar graph.
Flow of Events for Main Success Scenario:
-> 1. User clicks on “View Monthly Calories Burned”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month and sends data to system.
<-4. System converts data to bar graph and displays on the screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on “View Monthly Calories Burned”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month but does not find data.
<-4. System displays message “No data uploaded”.

UC10 ViewMonthlyStressLevel
Initiating Actor: User
Actor’s Goal: To view the stress levels for a month in the form of a line graph.
Participating Actor: Database

Pre-conditions: System displays the user’s current month home screen.
Post-conditions: System displays the stress levels for a month in form of a line graph.
Flow of Events for Main Success Scenario:
-> 1. User clicks on “View Monthly Stress Levels”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month and sends data to system.
<-4. System converts data to line graph and displays on the screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on “View Monthly Stress Levels”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month but does not find data.
<-4. System displays message “No data uploaded”.

UC11 ViewHeartCondition
Initiating Actor: User
Actor’s Goal: To view whether their heart condition is normal, Tachycardia, Bradycardia or cardiac mortality in the form of text.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen.
Post-conditions: System displays heart condition in form of text.
Flow of Events for Main Success Scenario:
-> 1. User clicks on “View Heart Condition”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month and sends data to system.
<-4. System calculates average monthly heart rate, matches with normal heart rate for user’s age and displays appropriate condition on the screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on “View Heart Condition”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month but does not find data.
<-4. System displays message “No data uploaded”.

UC12 CalculateDailyAverageHeartRate
Initiating Actor: User
Actor’s Goal: To view the average values of heart rate for a day in the form of text.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen.
Post-conditions: System displays average heart rate for a day in form of text.
Flow of Events for Main Success Scenario:
-> 1. User clicks on a day and then clicks on “View Average Daily Heart Rate”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the day and sends data to system.

<-4. System calculates average of data and displays on the screen in form of text.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on a day and then clicks on “View Average Daily Heart Rate”.
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the day but does not find data.
<-4. System displays message “No data uploaded”.

UC13 DeleteProfile
Initiating Actor: User
Actor’s Goal: To delete his profile and all the data associated with it from the system.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen.
Post-conditions: System deletes all the data associated with the user from the database and log out the user.
Flow of Events for Main Success Scenario:
-> 1. User clicks on edit profile and then click on “delete profile”.
<- 2. System prompts the user if he really wants to delete the profile.
-> 3. User clicks on “Yes”.
<- 4. System sends delete request to database.
->5. Database searches for user’s data and delete all the data associated with the user.
<-6. System shows the message that all data has been deleted and logs out the user from the system.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on edit profile and then click on “delete profile”.
<- 2. System prompts the user if he really wants to delete the profile.
-> 3. User clicks on “No”.
<- 4. System displays user profile.

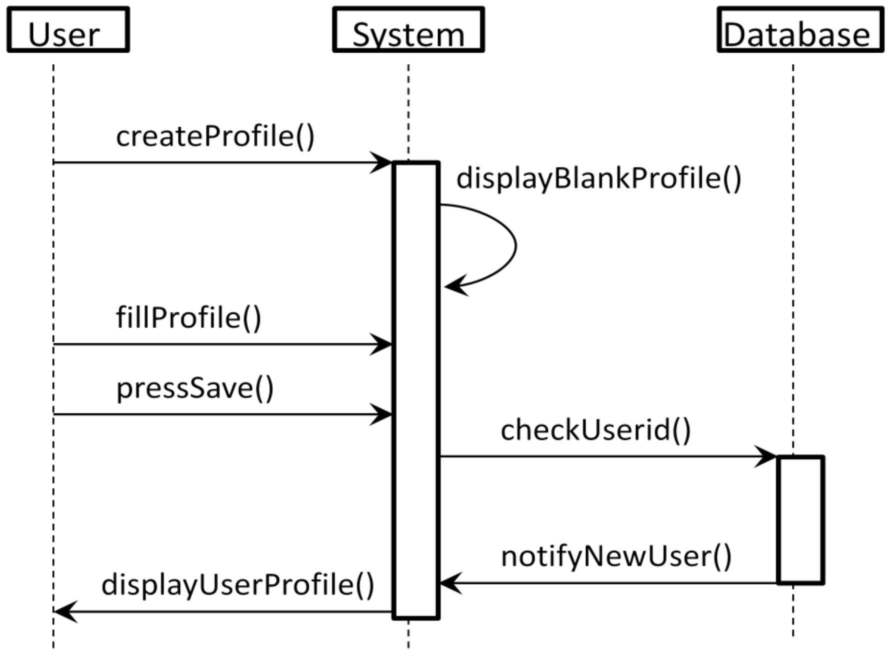
UC14 RetrievePassword
Initiating Actor: Administrator
Actor’s Goal: To retrieve a user data from the system.
Participating Actor: Administrator, Database
Pre-conditions: Administrator is on the login page.
Post-conditions: Administrator retrieves the user password.
Flow of Events for Main Success Scenario:
-> 1. Administrator enters his id and password and clicks on “login”.
<- 2. System checks if administrator id and password match, finds a match.
<- 3. System loads the administrator’s home page on screen.
-> 4. Administrator searches for the user ID.
<- 5. System displays user information.
-> 6. Administrator retrieves the user password.
Flow of Events for Extension (Alternate Scenario):
-> 1 Administrator enters his id and password and clicks on “login”.

<- 2. System checks for the id and password, finds that there is no such id and password in the database.
<- 3. System gives a message “Wrong Administrator ID or Password” on screen.

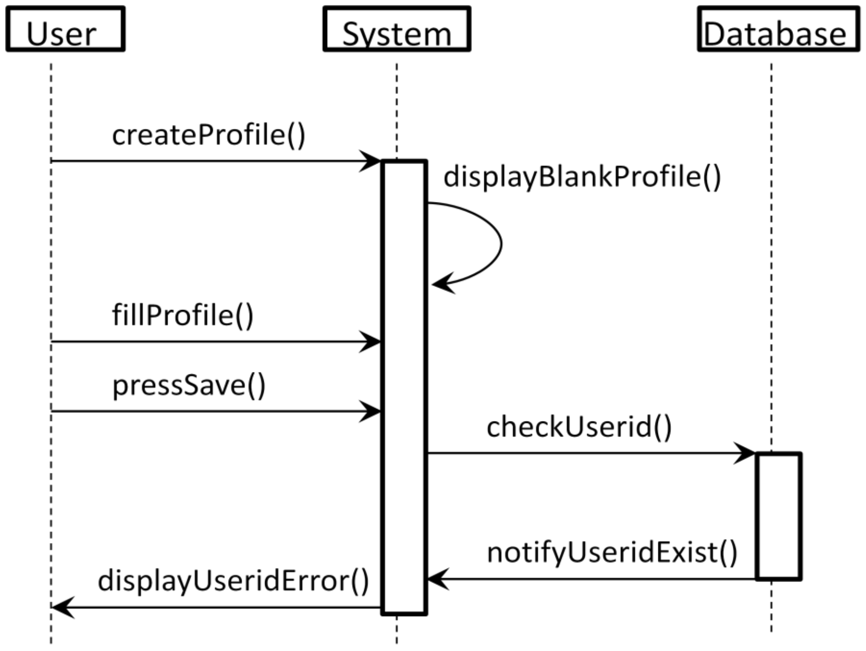
UC15 ViewYearlyGraph
Initiating Actor: User
Actor’s Goal: To view the heart rate for a year in the form of a line graph.
Participating Actor: Database
Pre-conditions: System displays the user’s current month home screen.
Post-conditions: System displays the yearly heart rate in form of a line graph.
Flow of Events for Main Success Scenario:
-> 1. User clicks on the month
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month and sends data to system.
<-4. System converts data to line graph and displays on the screen.
Flow of Events for Extension (Alternate Scenario):
-> 1. User clicks on the month
<- 2. System sends request to database for the data.
->3. Database searches for user’s data for the month but does not find data.
<-4. System displays message “No data uploaded”.

4.6 System Sequence Diagrams

UC1 CreateProfile
Success Case:

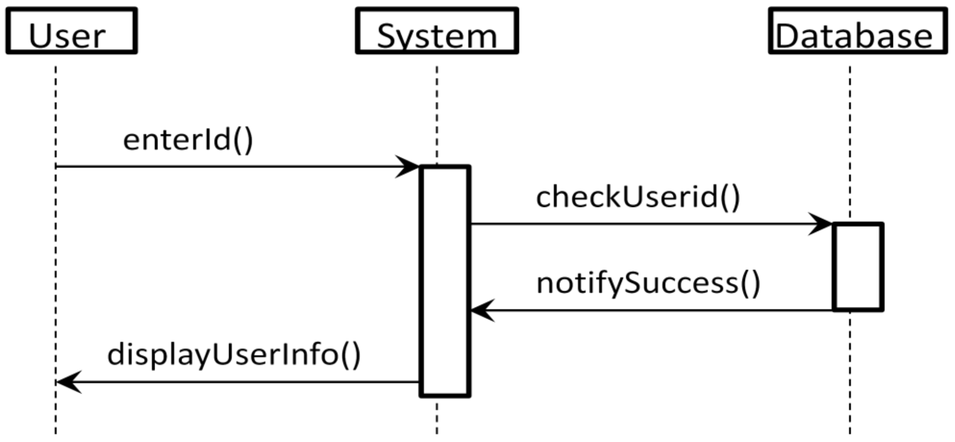


Alternate Case:

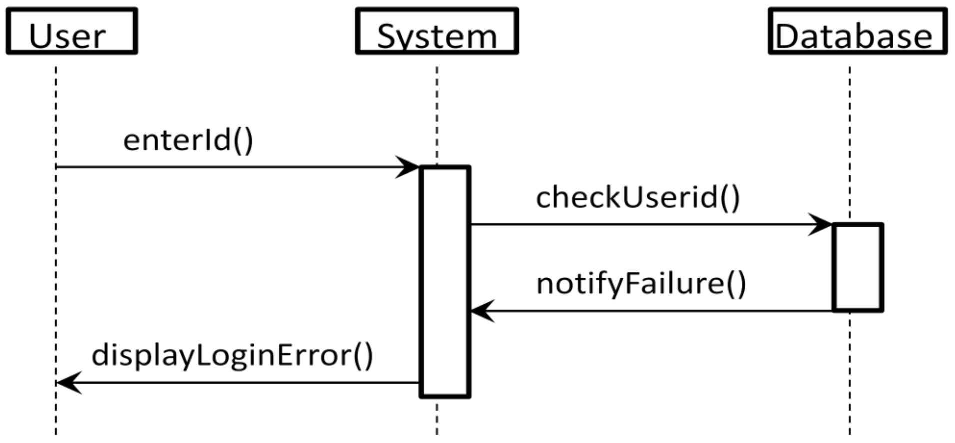


UC2 Login

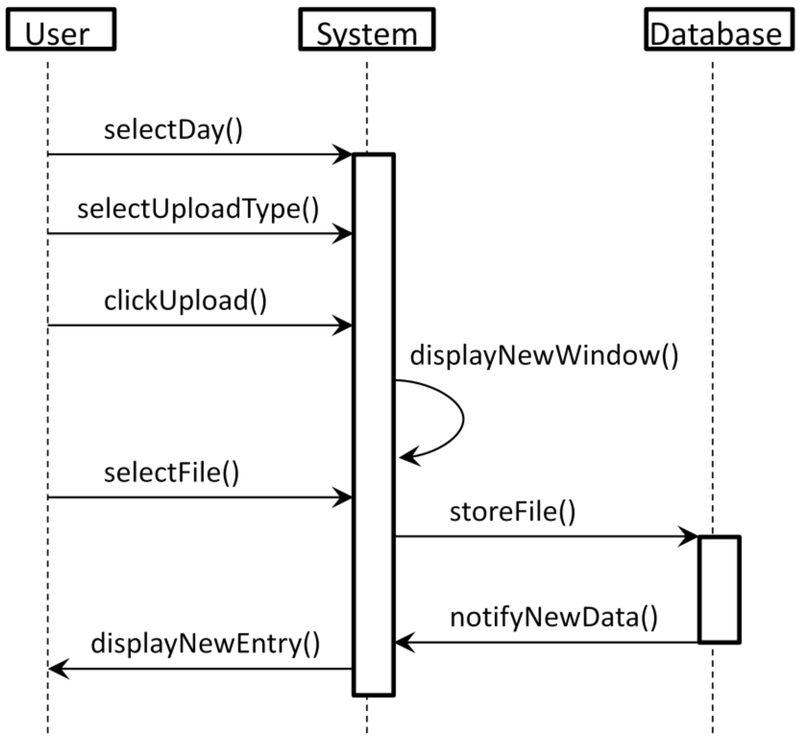
Success Case:



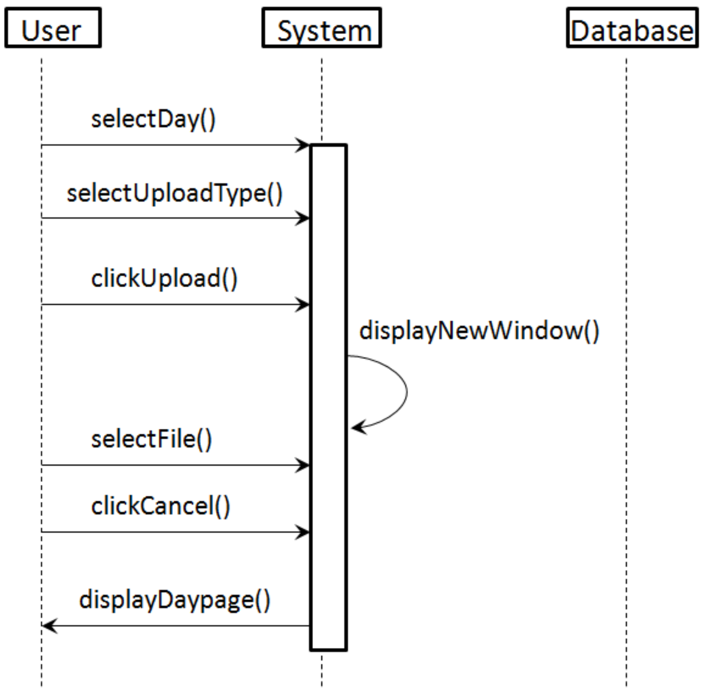
Alternate Case:



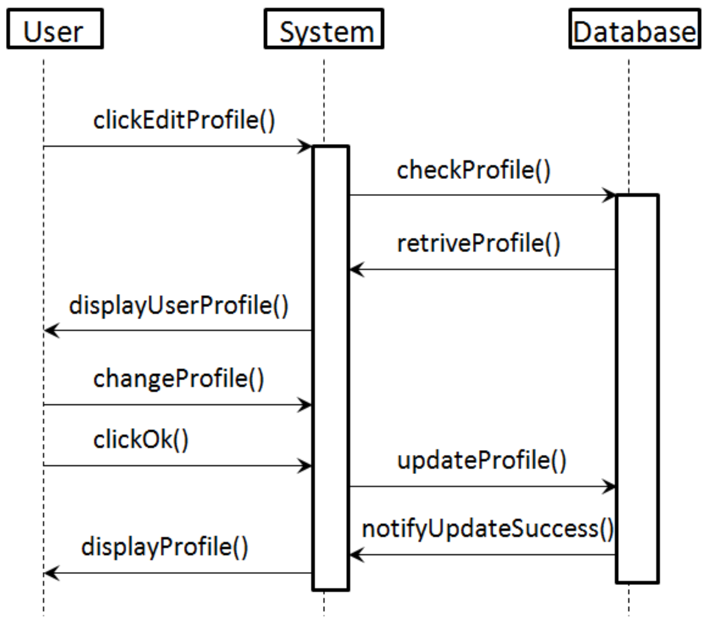
UC3 UploadFile
Success Case:



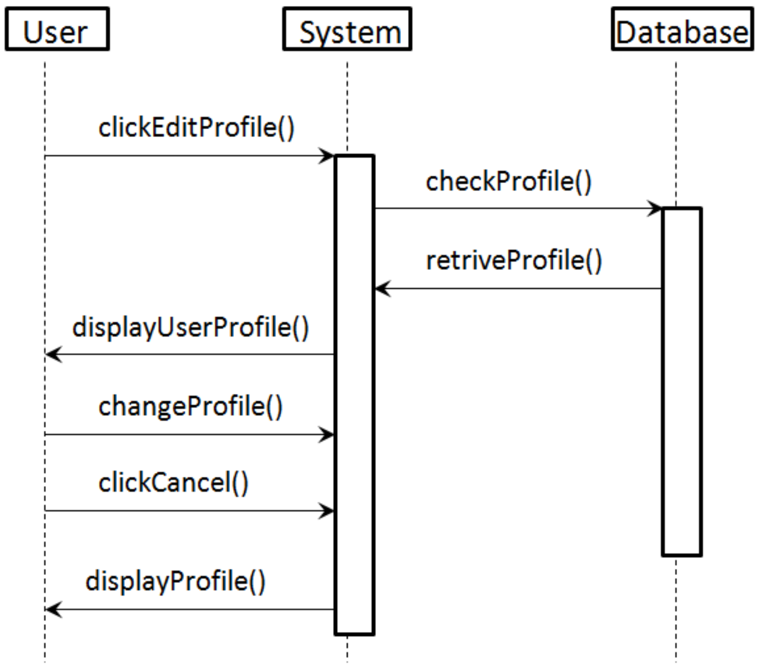
Alternate Case



UC4 ChangeProfile
Success Case:

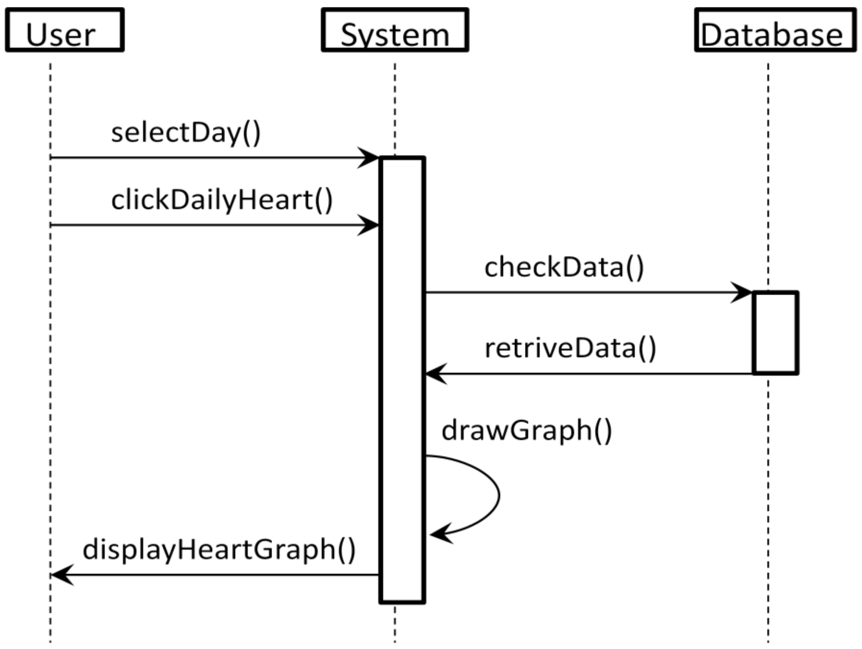


Alternate Case:

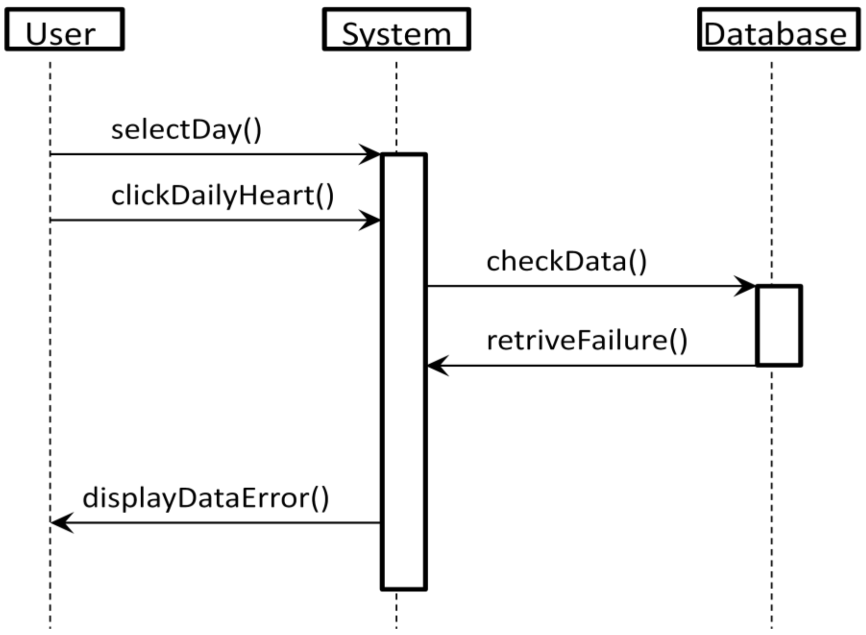


UC5 ViewDailyHeartRate

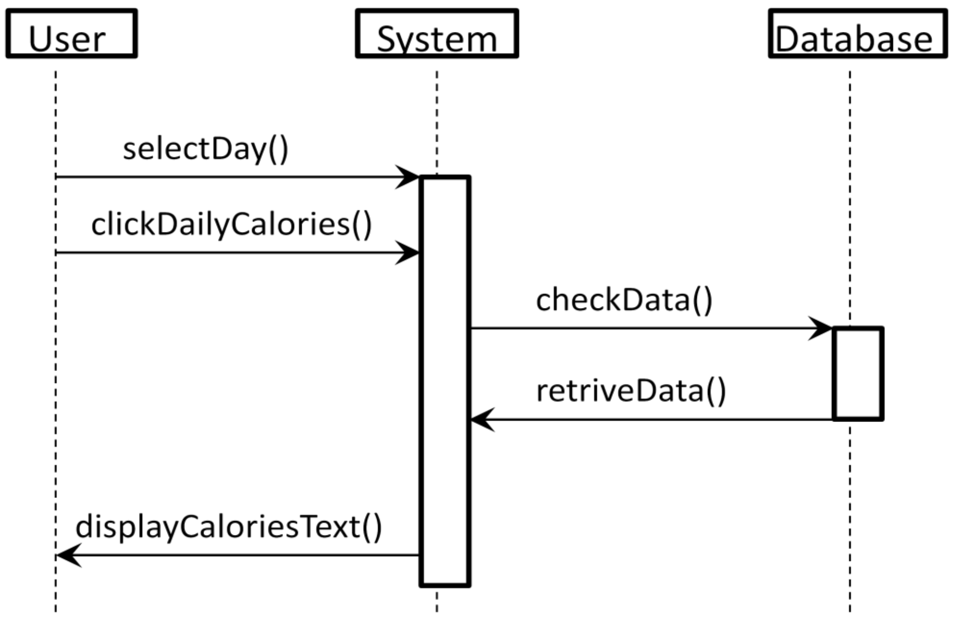
Success Case:



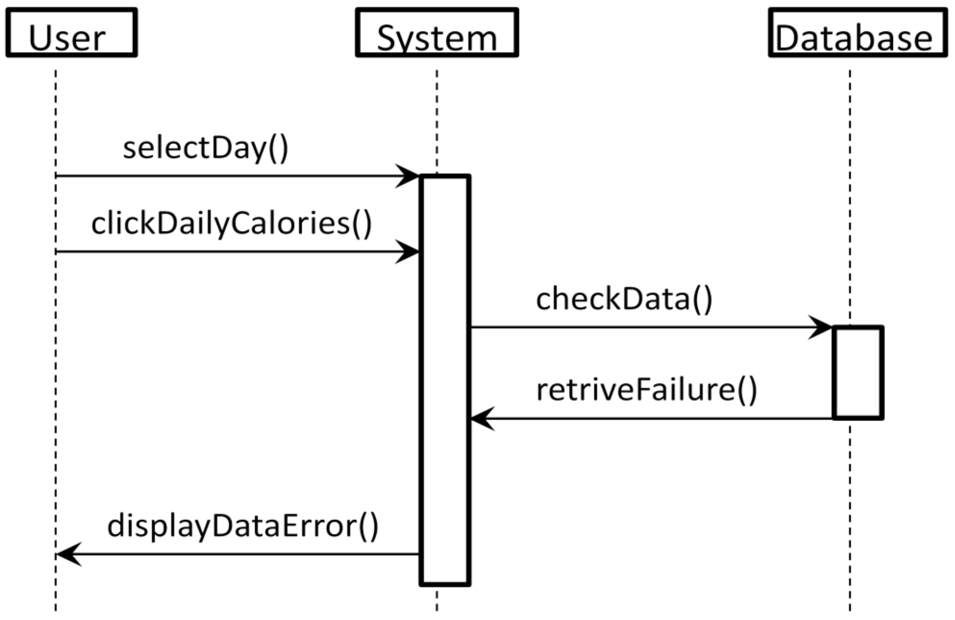
Alternate Case:



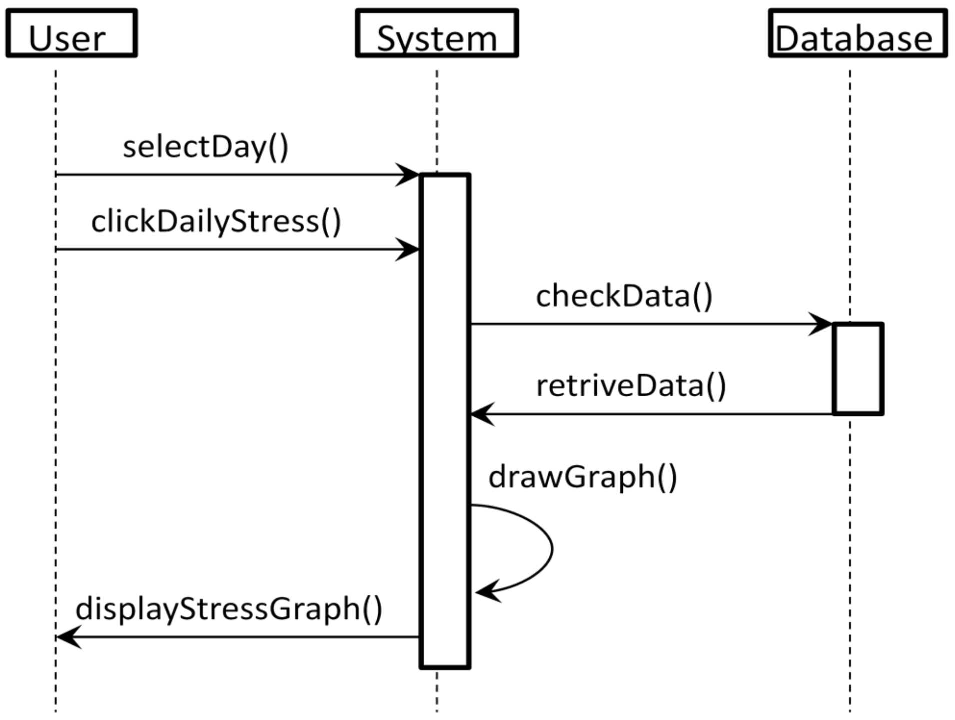
UC6 ViewDailyCaloriesBurnt
Success Case:



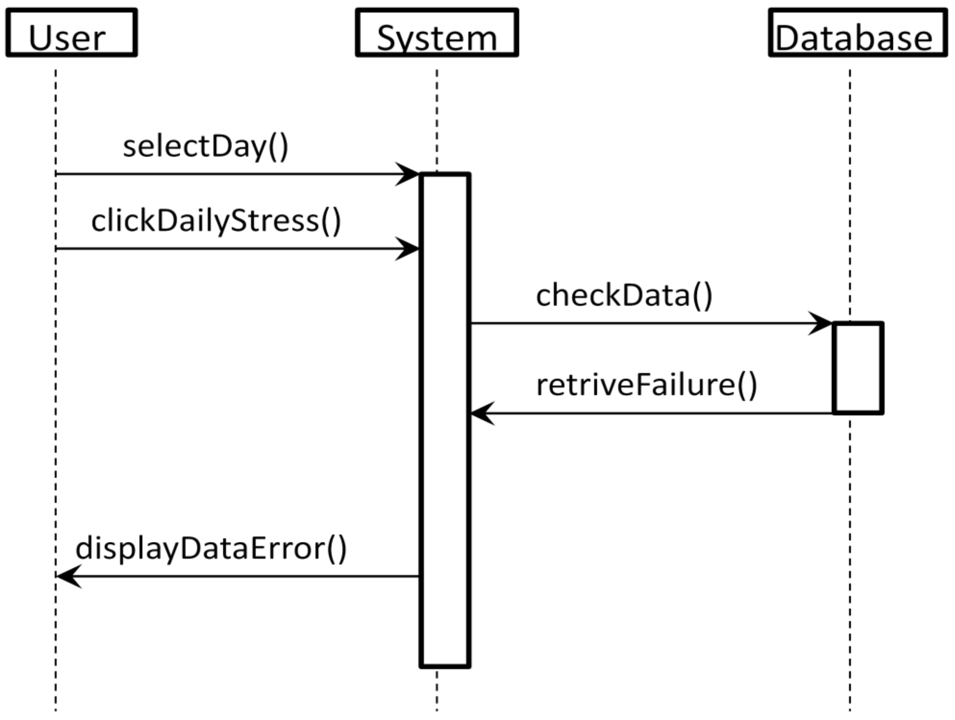
Alternate Case:



UC7 ViewDailyStressLevel
Success Case:

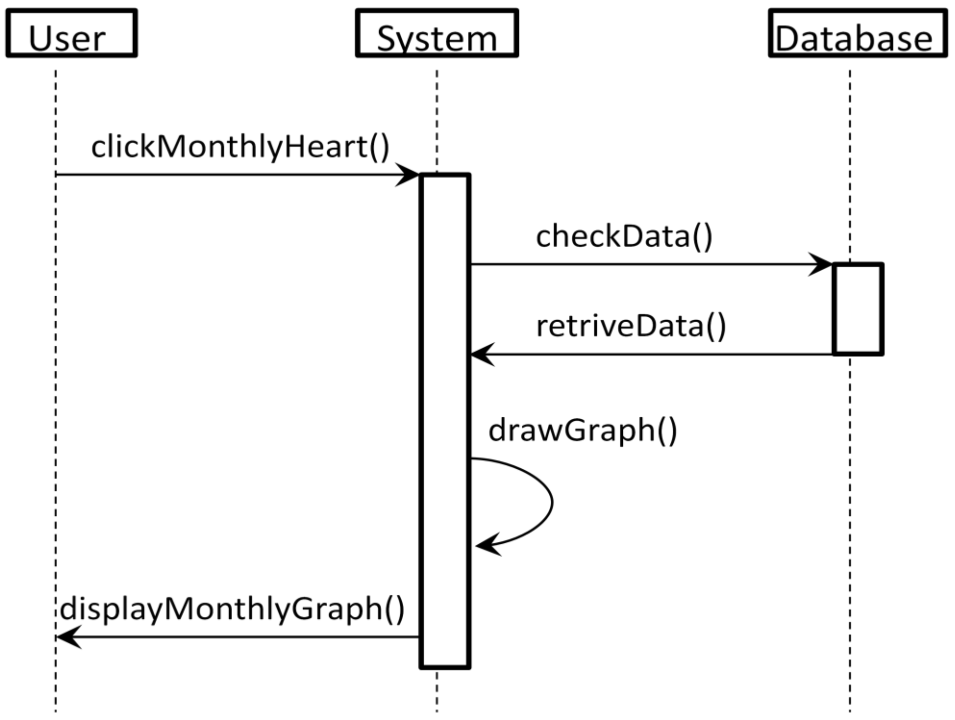


Alternate Case:

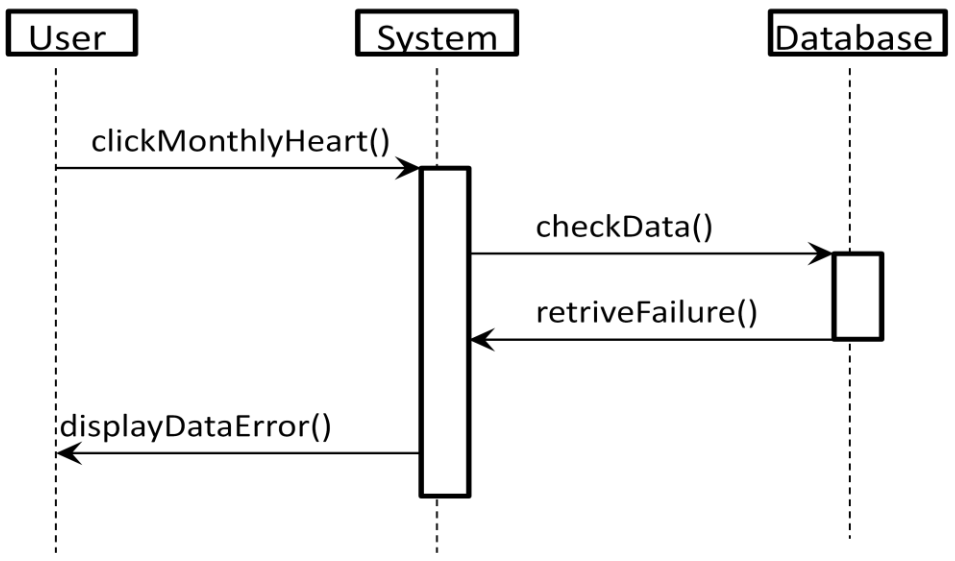


UC8 ViewMonthlyHeartRate

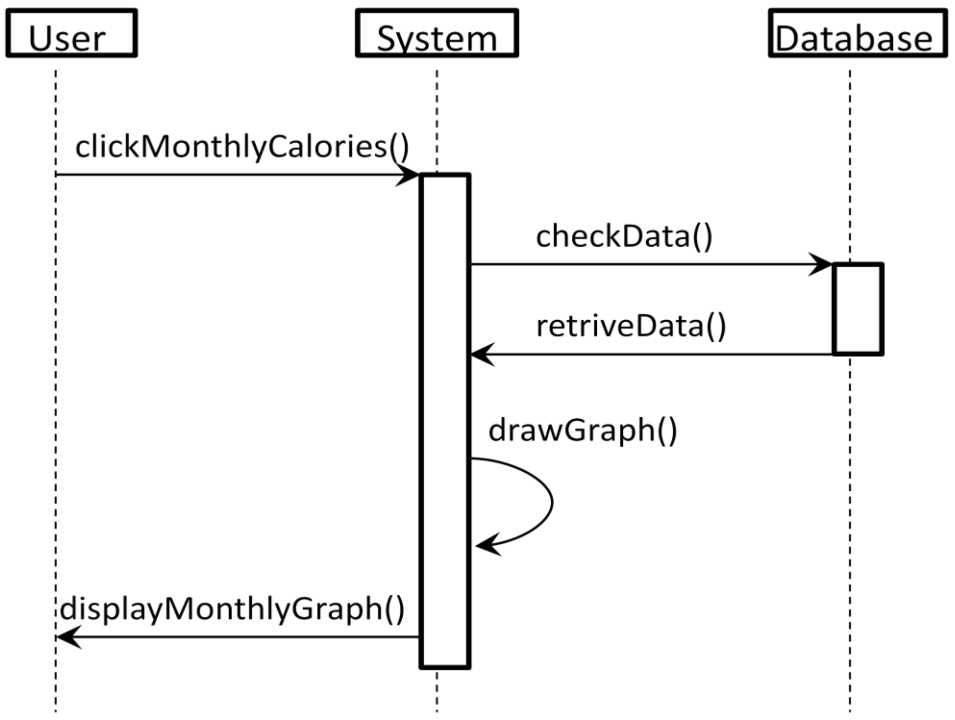
Success Case:



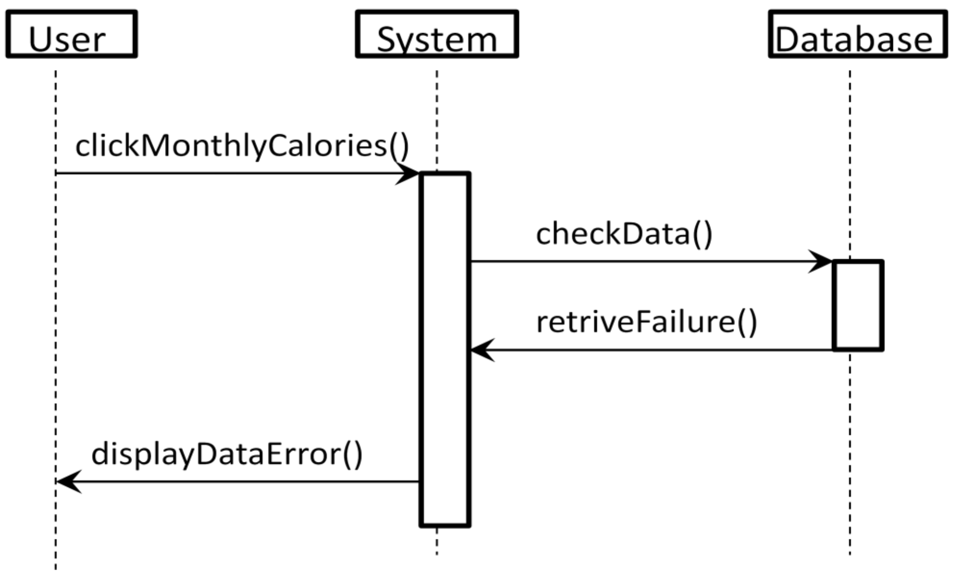
Alternate Case:



UC9 ViewMonthlyCaloriesBurnt
Success Case:

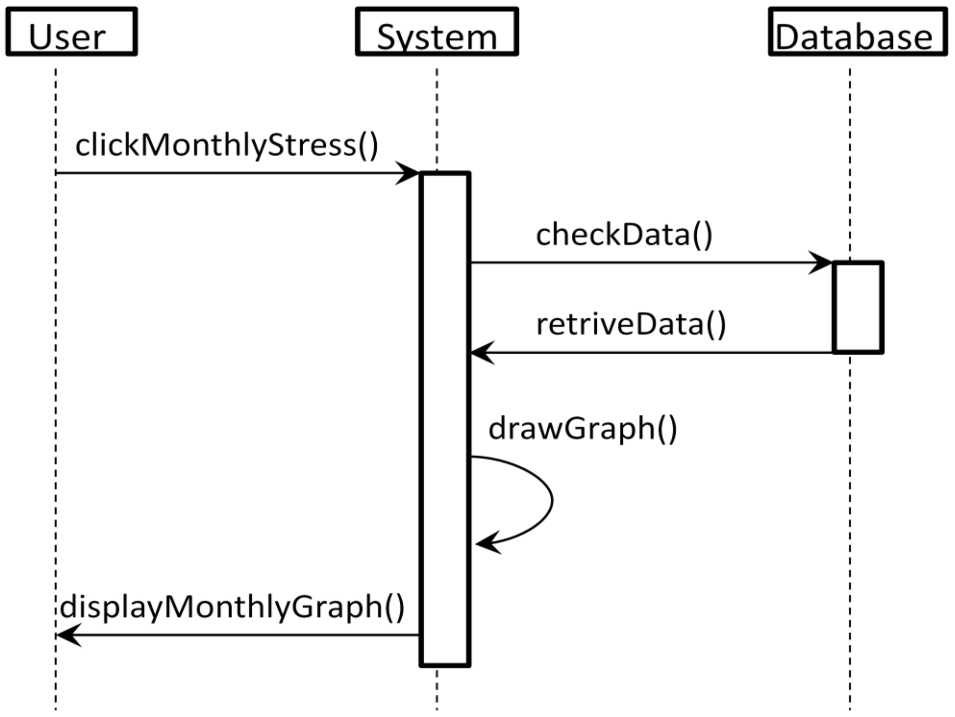


Alternate Case:

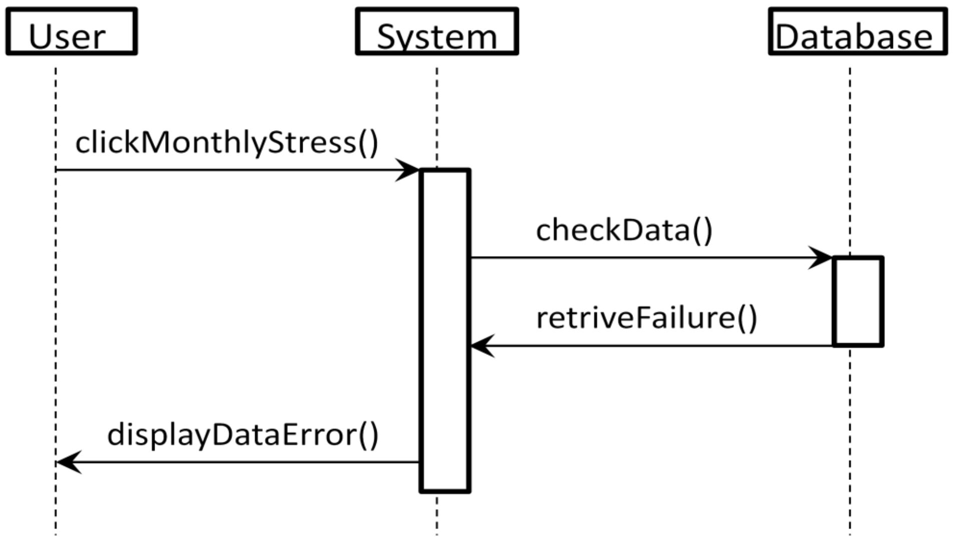


UC10 ViewMonthlyStressLevel

Success Case:

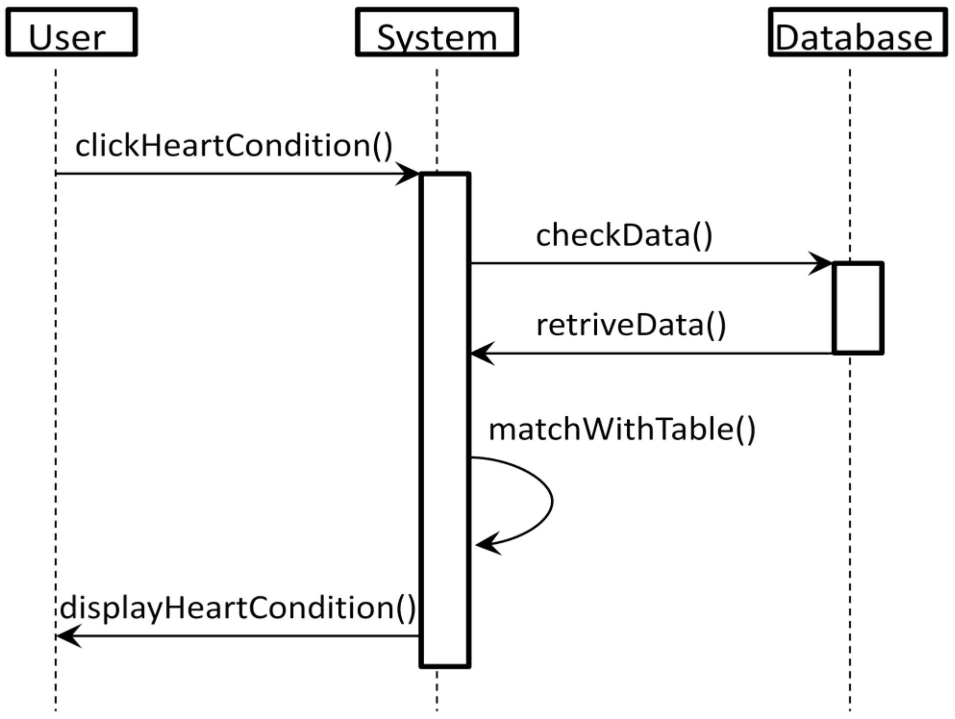


Alternate Case:

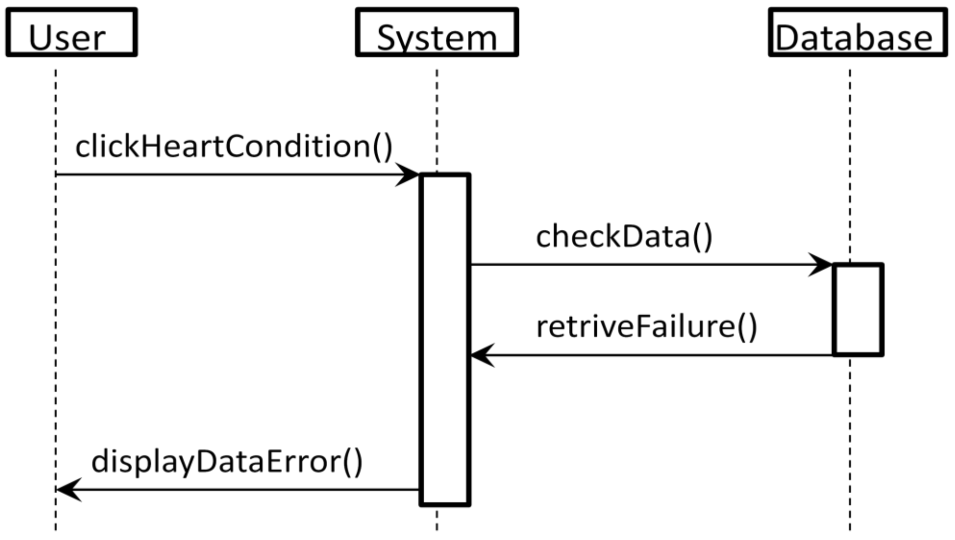


UC11 ViewHeartCondition

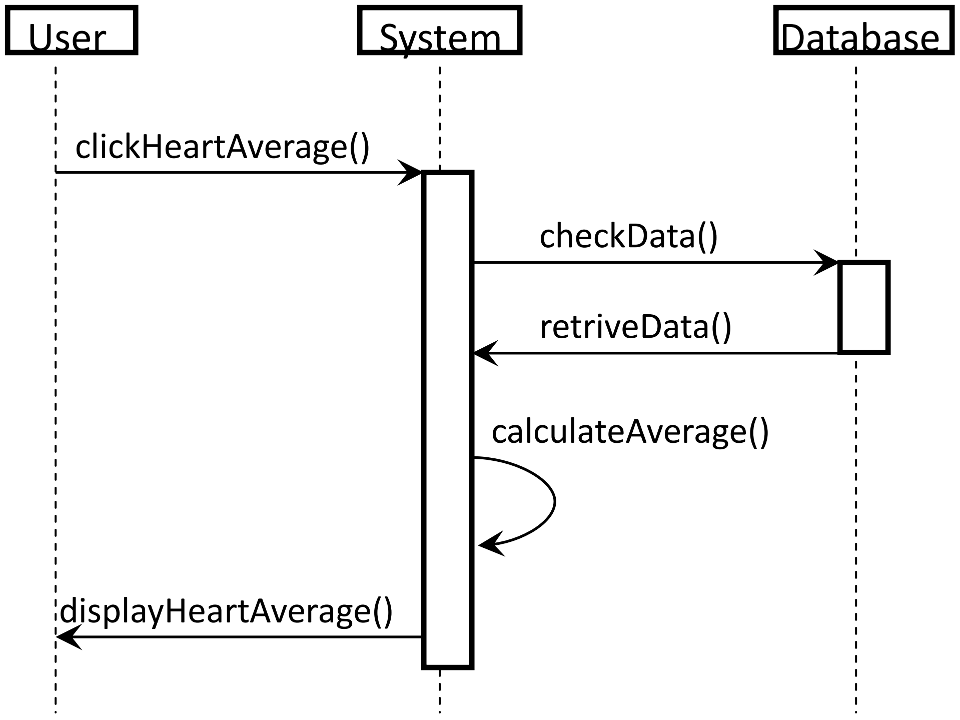
Success Case:



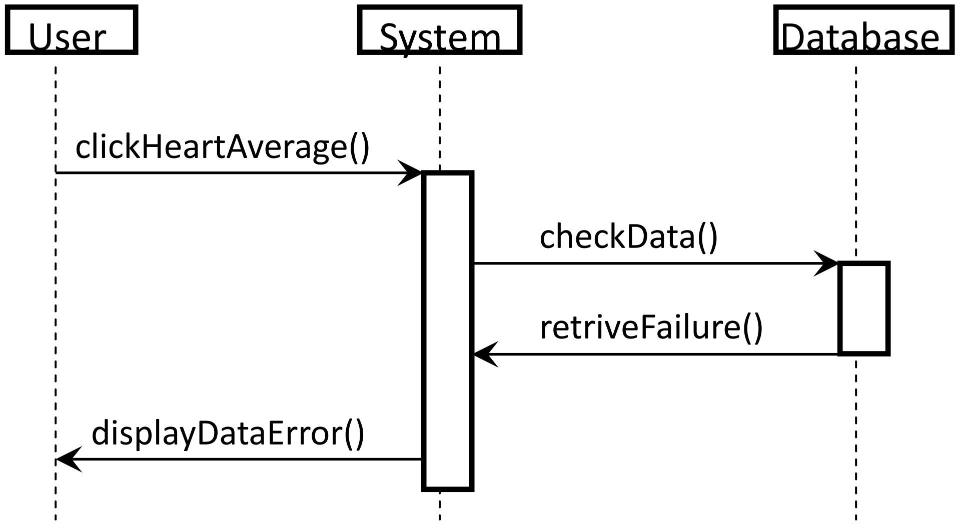
Alternate Case:



UC12 CalculateDailyAverageHeartRate
Success Case:

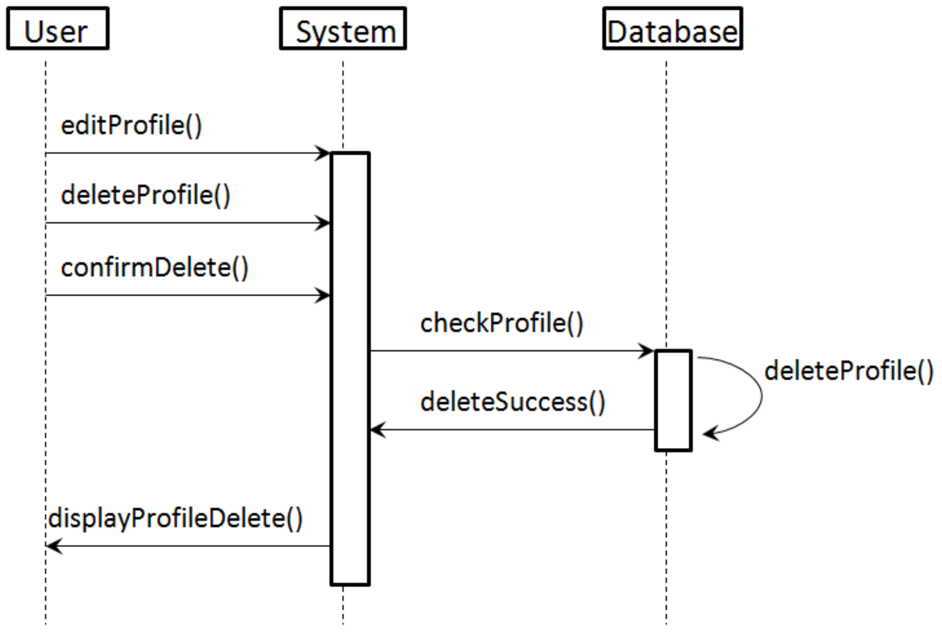


Alternate Case:

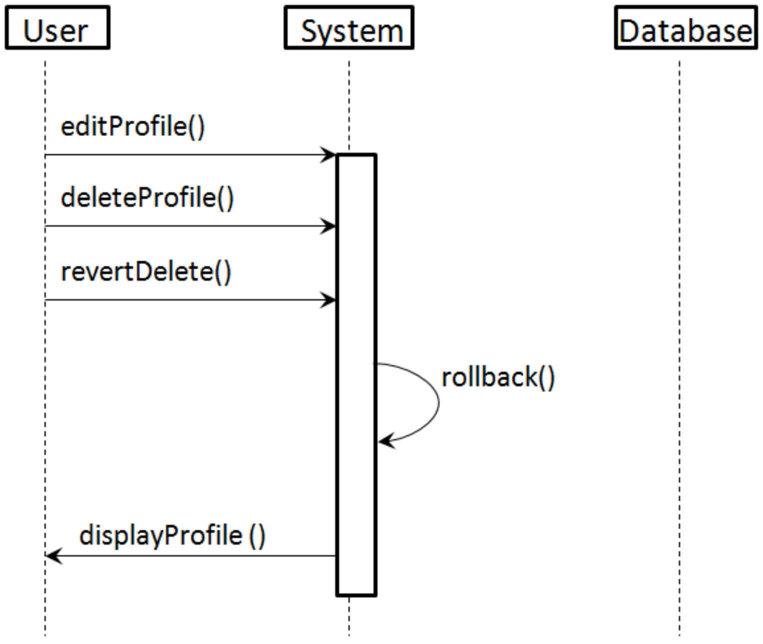


UC13 DeleteProfile

Success Case:

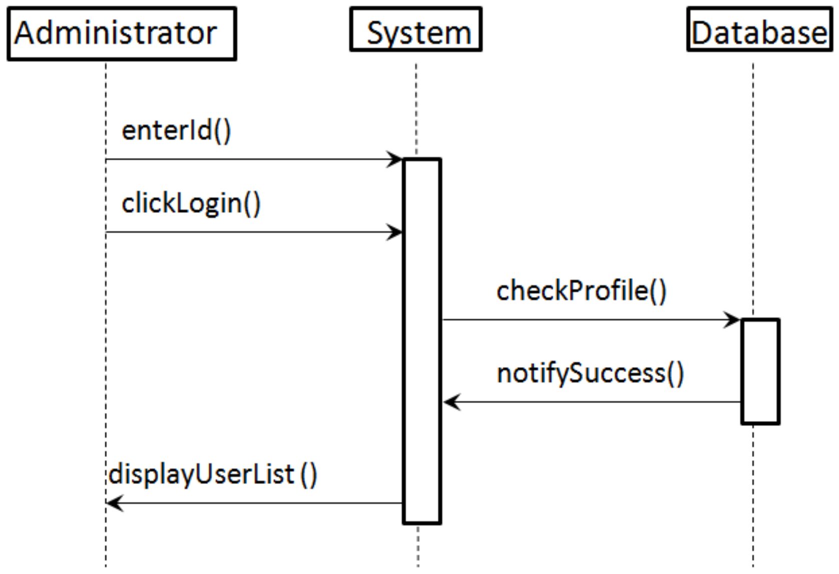


Alternate Case:

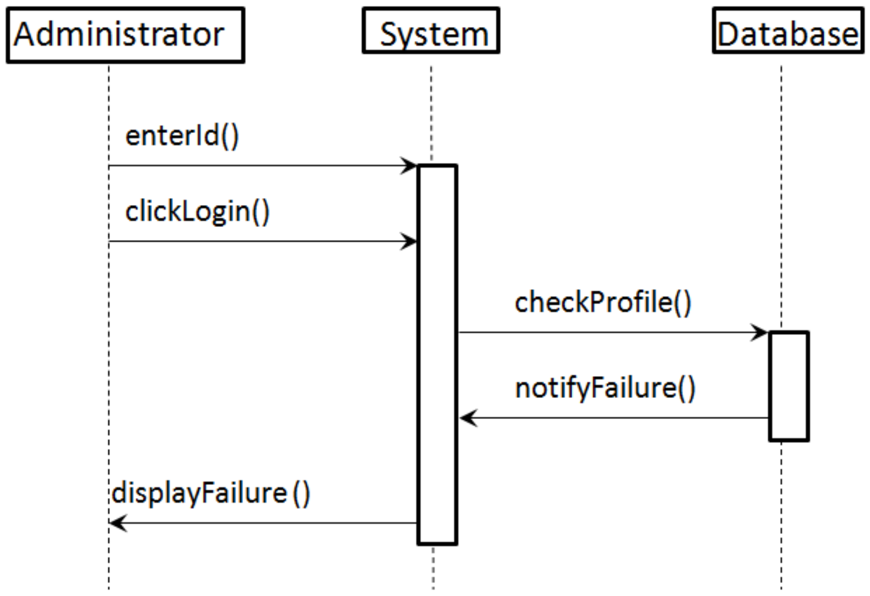


UC14 RetrievePassword

Success Case:



Alternate Case:



5. Effort estimate using use case points

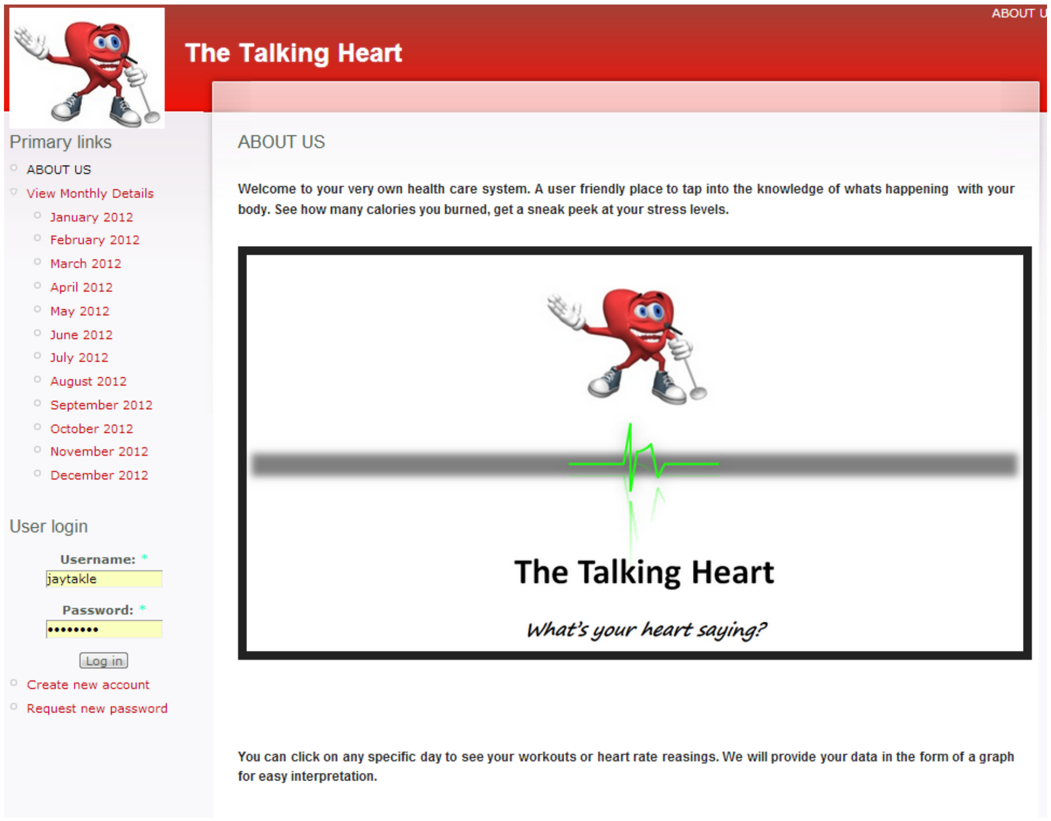
We have to estimate the effort the customer and the administrator puts into the system to obtain the required results.

Customer User Effort Estimation:

Typical usage scenario 1: A patient already has an account. He wants to upload and view his data. He accesses the system using his user ID and password.

<u>NAVIGATION</u>:

To Upload data:
a) Click on login and enter your user id and password
b) From the home screen click on the current month
c) When the timeline appears, click on the appropriate date.
d) When the day page is loaded click on upload data.



Welcome to your very own health care system. A user friendly place to tap into the knowledge of whats happening with your body. See how many calories you burned, get a sneak peek at your stress levels.

You can click on any specific day to see your workouts or heart rate reasings. We will provide your data in the form of a graph for easy interpretation.

Fig 5.1 Login page

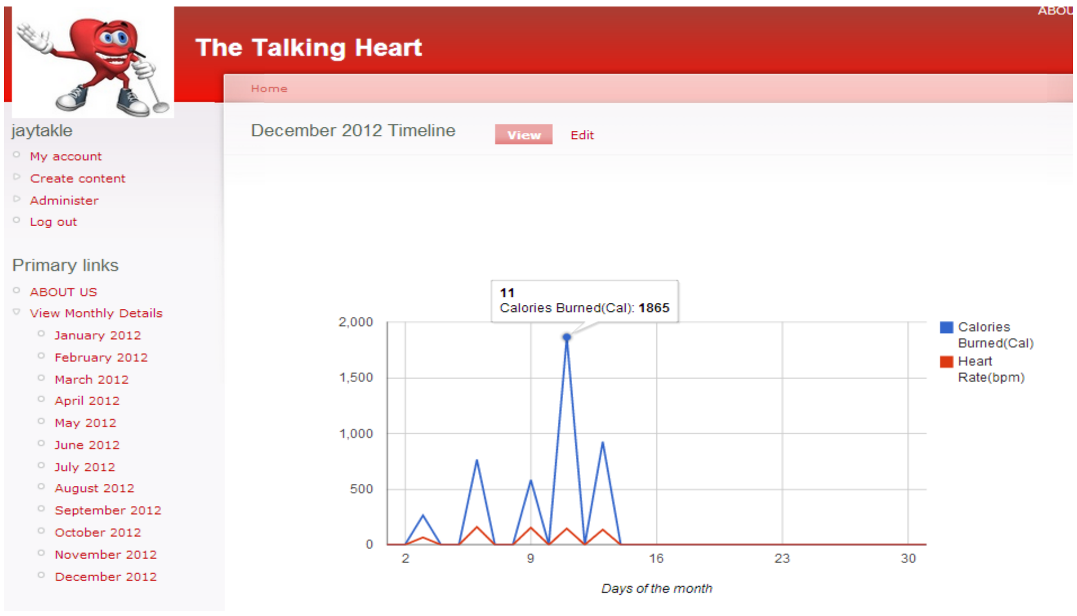
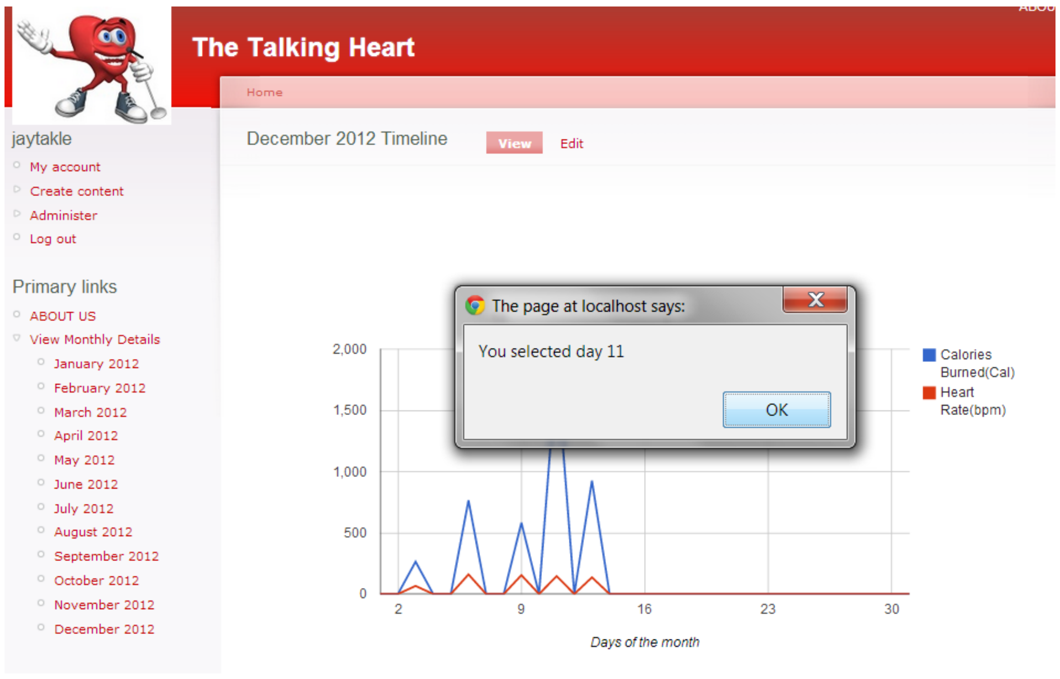


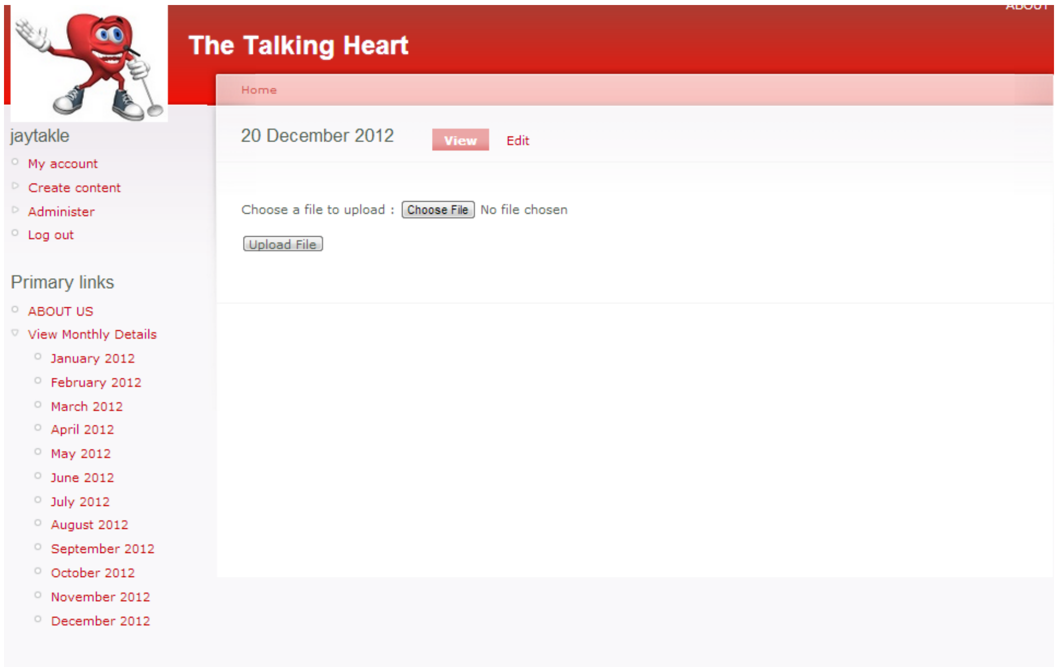
Fig 5.2 Timeline Navigation page



The page at localhost says:

You selected day 11

Fig 5.3 Day selected prompt


The Talking Heart

Home

jaytakle

- My account
- Create content
- Administer
- Log out

Primary links

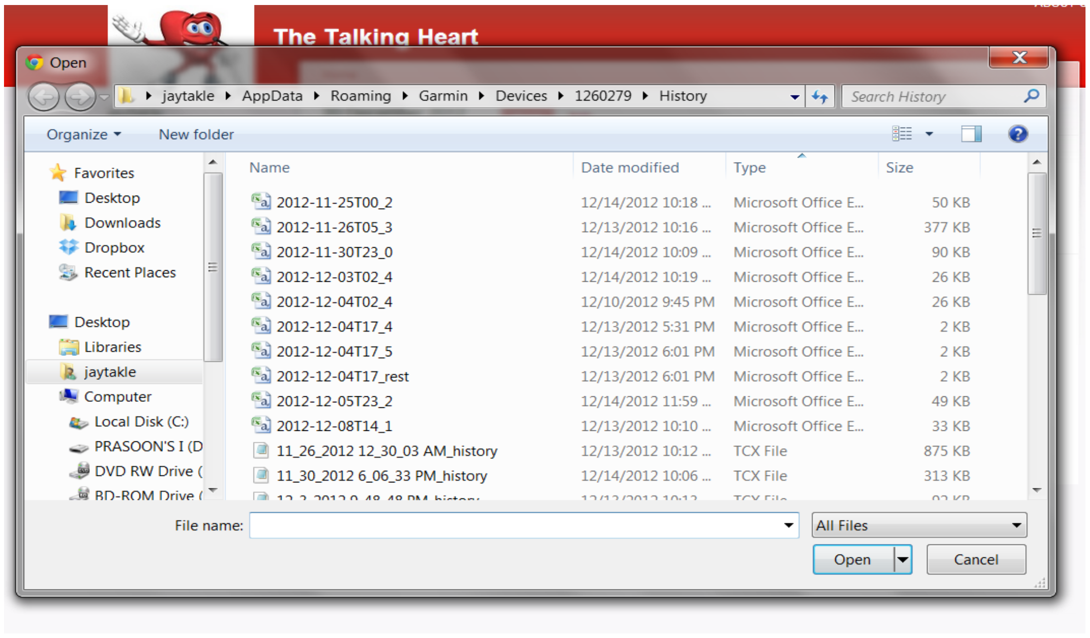
- ABOUT US
- View Monthly Details
  - January 2012
  - February 2012
  - March 2012
  - April 2012
  - May 2012
  - June 2012
  - July 2012
  - August 2012
  - September 2012
  - October 2012
  - November 2012
  - December 2012

20 December 2012 View Edit

Choose a file to upload : Choose File No file chosen

Upload File

Fig 5.4 Day page


Open

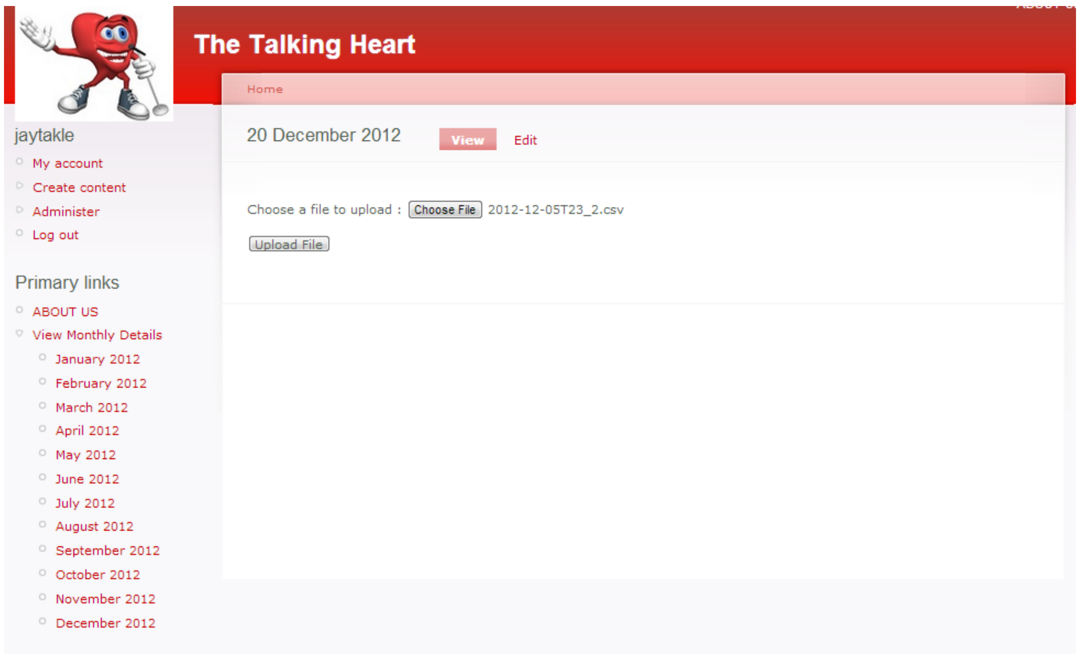
jaytakle ▸ AppData ▸ Roaming ▸ Garmin ▸ Devices ▸ 1260279 ▸ History

Name	Date modified	Type	Size
2012-11-25T00_2	12/14/2012 10:18 ...	Microsoft Office E...	50 KB
2012-11-26T05_3	12/13/2012 10:16 ...	Microsoft Office E...	377 KB
2012-11-30T23_0	12/14/2012 10:09 ...	Microsoft Office E...	90 KB
2012-12-03T02_4	12/14/2012 10:19 ...	Microsoft Office E...	26 KB
2012-12-04T02_4	12/10/2012 9:45 PM	Microsoft Office E...	26 KB
2012-12-04T17_4	12/13/2012 5:31 PM	Microsoft Office E...	2 KB
2012-12-04T17_5	12/13/2012 6:01 PM	Microsoft Office E...	2 KB
2012-12-04T17_rest	12/13/2012 6:01 PM	Microsoft Office E...	2 KB
2012-12-05T23_2	12/14/2012 11:59 ...	Microsoft Office E...	49 KB
2012-12-08T14_1	12/13/2012 10:10 ...	Microsoft Office E...	33 KB
11_26_2012 12_30_03 AM_history	12/13/2012 10:12 ...	TCX File	875 KB
11_30_2012 6_06_33 PM_history	12/14/2012 10:06 ...	TCX File	313 KB

File name: All Files

Open Cancel

Fig 5.5 File upload browser



The Talking Heart

Home

20 December 2012 View Edit

Choose a file to upload : Choose File 2012-12-05T23_2.csv

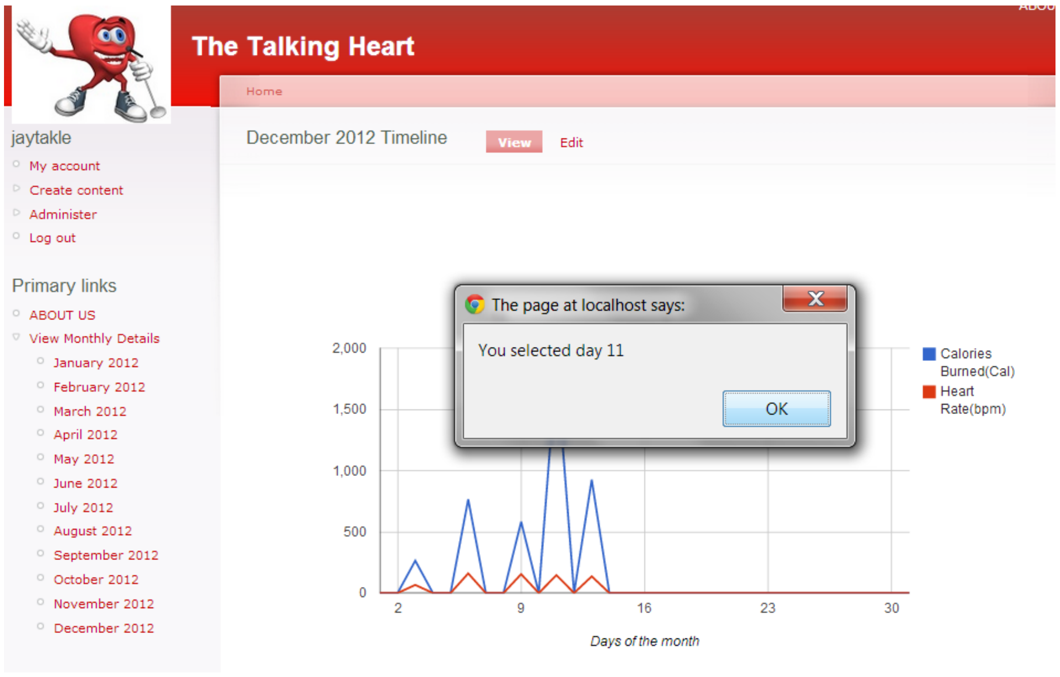
Upload File

jaytakle
- My account
- Create content
- Administer
- Log out

Primary links
- ABOUT US
- View Monthly Details
  - January 2012
  - February 2012
  - March 2012
  - April 2012
  - May 2012
  - June 2012
  - July 2012
  - August 2012
  - September 2012
  - October 2012
  - November 2012
  - December 2012

Fig 5.6 File uploaded confirm

To View Graph:
a) From the home screen click on the current month
b) When the timeline appears, click on the appropriate date.



The Talking Heart

Home

December 2012 Timeline View Edit

The page at localhost says:

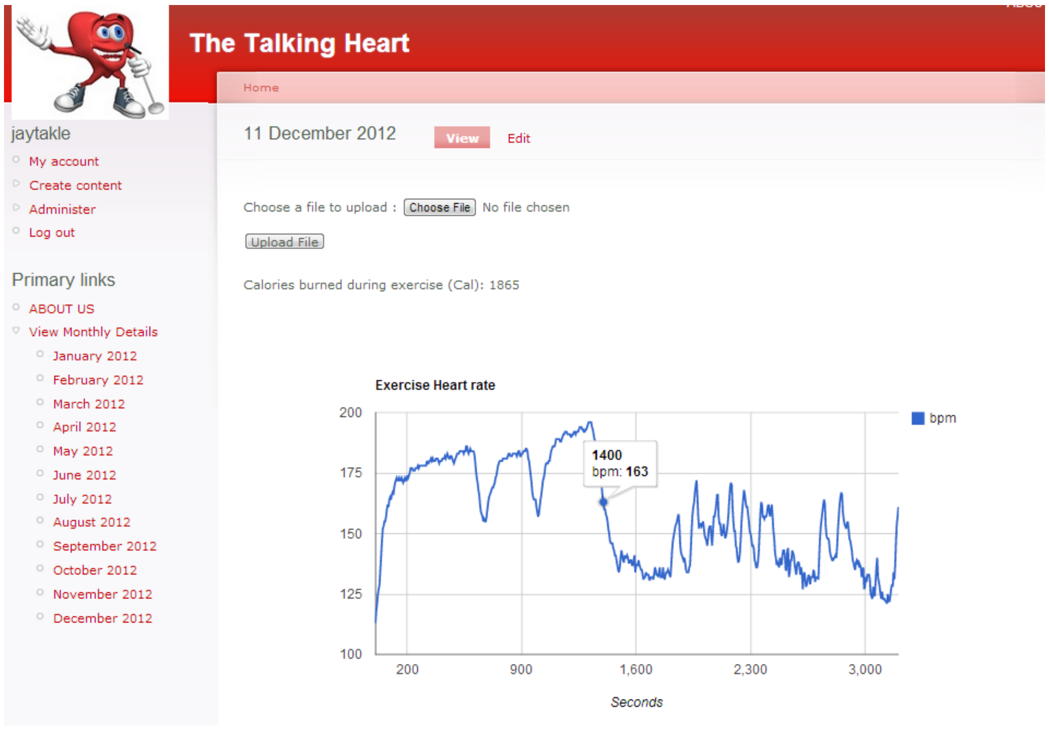
You selected day 11

OK

jaytakle
- My account
- Create content
- Administer
- Log out

Primary links
- ABOUT US
- View Monthly Details
  - January 2012
  - February 2012
  - March 2012
  - April 2012
  - May 2012
  - June 2012
  - July 2012
  - August 2012
  - September 2012
  - October 2012
  - November 2012
  - December 2012

Fig 5.7 Timeline Navigation



11 December 2012

Choose a file to upload : No file chosen

Calories burned during exercise (Cal): 1865

Fig 5.8 Day graph

<u>Worst-case scenario</u>: A patient does not have a user ID, he needs to register first then he can login. This puts in the extra user effort but it is a onetime effort.

To Register:
a) Click on Register and fill in details such as name, address and e-mail id.
b) Select user id and password.
Thus the complete user effort estimation is nine steps.

Administrator Effort Estimation:

Typical usage scenario: The administrator already has an account. He accesses the system using his user id and password.

<u>NAVIGATION:</u>

a) Click on login and enter the user id and password.
b) The requests appear on the home screen. Click on the request
c) If it is a password request, click on send password to send the forgotten password to the user.

6. Domain Analysis:

6.1 Domain Model:

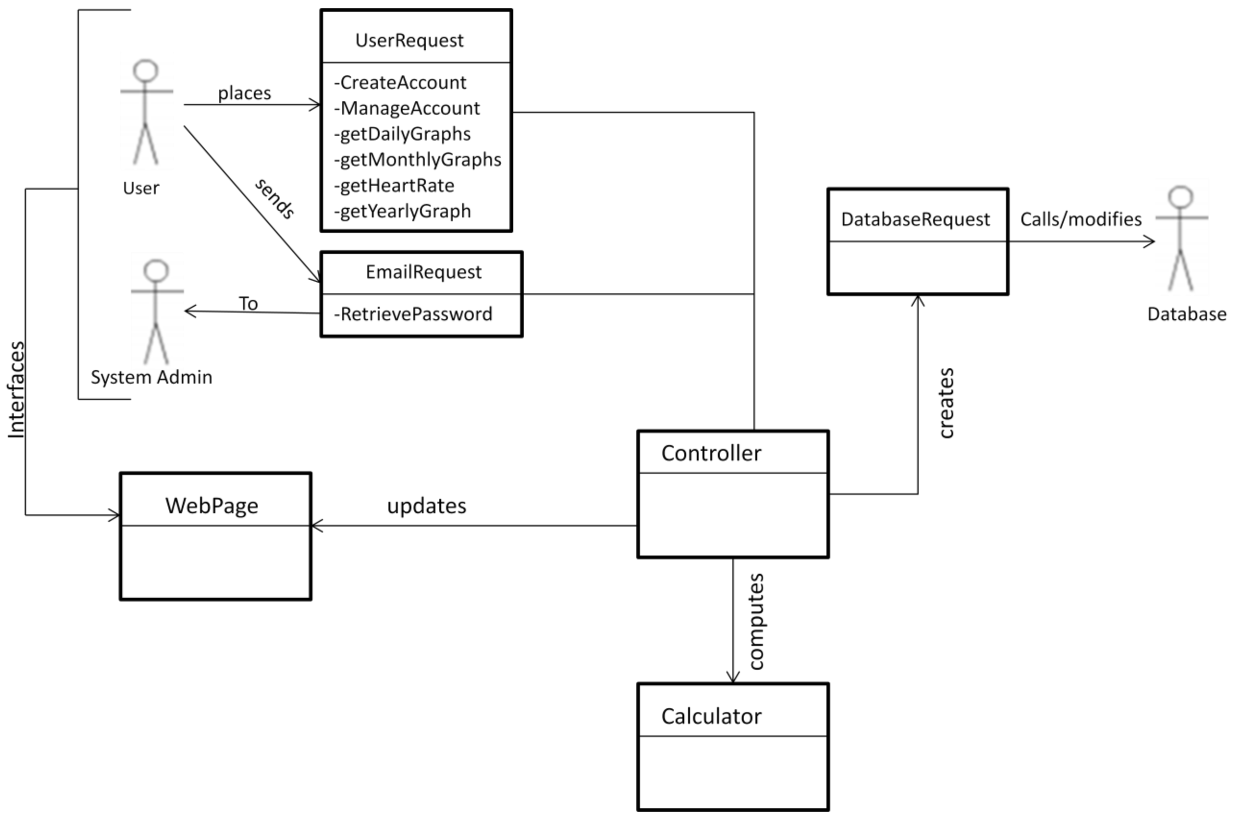


Fig 6.1 Domain Model for Talking Heart

6.1.1 Concept Definitions:

To analyze the domain model, we first derive the domain model concepts and responsibilities from the use cases that were already described. The following table 5.1 lists the domain model concepts and responsibilities for some of the use cases:

Responsibility	Type	Concept	Use Case
Register account using proper credentials	D	Security Object	UC-1: Create Profile

Database contains the user information.	K	Database	
Enter proper credentials to enter the home page	D	Login Screen	UC-2: Login
Select and store the data file to be uploaded	D	Database	UC-3: Upload File
Edit/delete the user information stored in the database	D	Main Controller	UC-4: Change Profile
Database contains the heart rate values of the user for each day	K	Database	UC-5: View Daily Heart Rate
Display the heart rate values for the day selected	D	Daily Display	
Database contains the calories burnt for each day	K	Database	UC-6: View Daily Calories Burnt
Display the calories burnt for the day selected	D	Daily Display	
Compute the stress level using the heart rate	D	Calculator	UC-7: View Daily Stress Level
Display the stress level for the day selected	D	Daily Display	
Compute the heart condition by comparing with the tables present in the database	D	Calculator	UC-11: View Heart Condition
Display the average heart rate values in the form of a graph.	D	Daily Display	
Compute the average heart rate	D	Calculator	UC-12: Calculate Daily Average Heart Rate
Display the average heart rate for the day selected	D	Daily Display	
Get the monthly heart rate values of the user that is stored in the database	D	Main Controller	UC-8: View Monthly Heart Rate
Display the graph of the monthly heart values	D	Monthly Display	
Database contains the total calories burnt for each day	K	Database	UC-9: View Monthly Calories Burnt

Display the calories burnt for the month	D	Monthly Display	
Compute the stress level using the monthly heart rate	D	Calculator	UC-10: View Monthly Stress Level
Display the stress level for the month selected	D	Daily Display	
Delete the profile from the database	D	Main Controller	UC-13: Delete Profile
Check the database for the user ID and retrieve the information	D	Main Controller	UC-14: Retrieve Password
Compute the monthly average heart rate	D	Calculator	UC-15: Calculate Yearly Heart Rate
Display the average heart rate for the different months	D	Yearly Display	

Table 6.1.1: Domain concept definitions

6.1.2 Association Definitions:

Some of the concepts explained above should work in some pattern to give the result. Such patterns are explained in the following table:

Concept Pair	Association Description	Association Name
Security Object← → Login screen	Only after authenticating the user, the login screen appears	Analyze and save
database← → Login Screen	The user can enter into login screen only if the credentials match with that of the data stored in the database	Analyze data
Login Screen← → Main Controller	The user can edit/delete his/her profile only after logging in	Analyze and save
Main Controller ← → Database	In order to display a graph requested by the user, the controller has to retrieve the values from the database	Choose from the analyzed data

Main Controller ←→ Calculator	In order to display a graph requested by the user which requires calculation, the calculator has to compute the values retrieved by the controller	Calculate the analyzed data
Main Controller ←→ Display	A corresponding graph gets displayed based on the day/month selected. Controller retrieve the data of that particular day/month and displays the graph	Show analyzed data.

Table 6.1.2: Association Definitions

6.1.3 Attribute Definitions:

Among the concepts defined, some share a same attribute. These concepts are listed in the following table:

Concept	Attributes	Attribute Description
Security Object	Field checker	Used to make sure all the fields are filled in
	Data Passer	Used to store the user in database
Login	Field checker	Used to make sure all the fields are filled in
	Password Checker	Used to verify the password
	User Search	Used to verify if the user exists in the database
Controller	User Search	Used to search for the right user in the database
	GetValues	Used to get the heart rate values of the same user
	Get Heart Condition	Used to analyze the heart condition of the user
Calculator	Calculate stress level	Used to calculate daily/monthly stress level
	Calculate average heart rate	Used to calculate daily average heart rate
	Compute heart condition	Used to calculate daily heart condition by comparing with the table
Database	User Search	Used to verify if the user exists in the database

	User info	Used to link the information to be stored with the user
Display	Display values	Used to display the heart rate, stress and calories burnt values of selected day/month of the user
	Display graph	Used to display the graph of the selected values
	Display home screen	Used to display home screen with user information

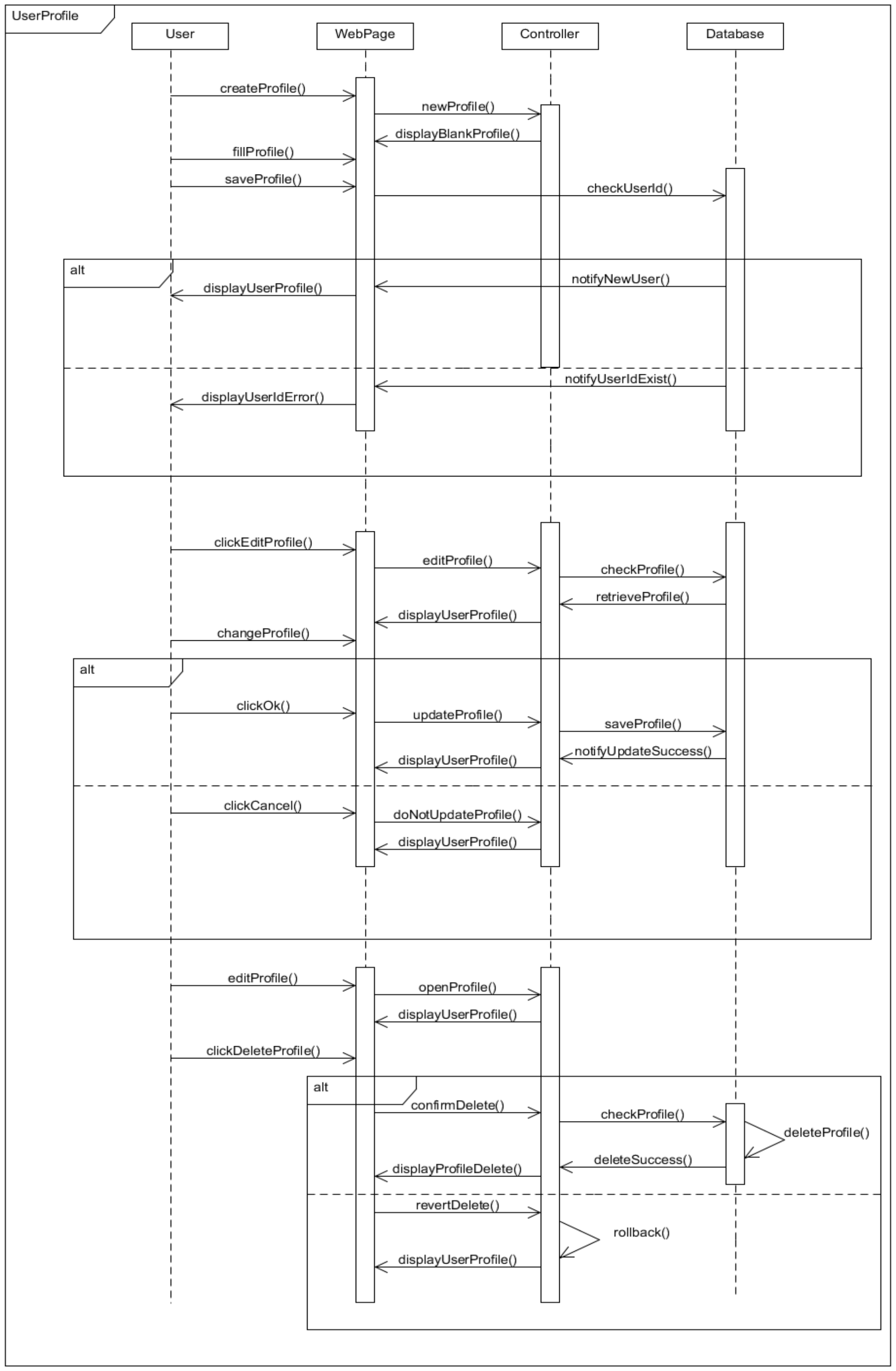
Table 6.1.3: Attribute Definitions

6.1.4. Traceability Matrix:

Domain Concept	Use Cases														
	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11	UC12	UC13	UC14	UC15
Security Object		X		X											
Login	X	X	X	X	X										
Database	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Calculator							X			X	X	X			X
Controller			X		X	X	X	X	X	X	X	X	X	X	X
Display	X		X	X	X	X	X	X	X	X	X	X			X

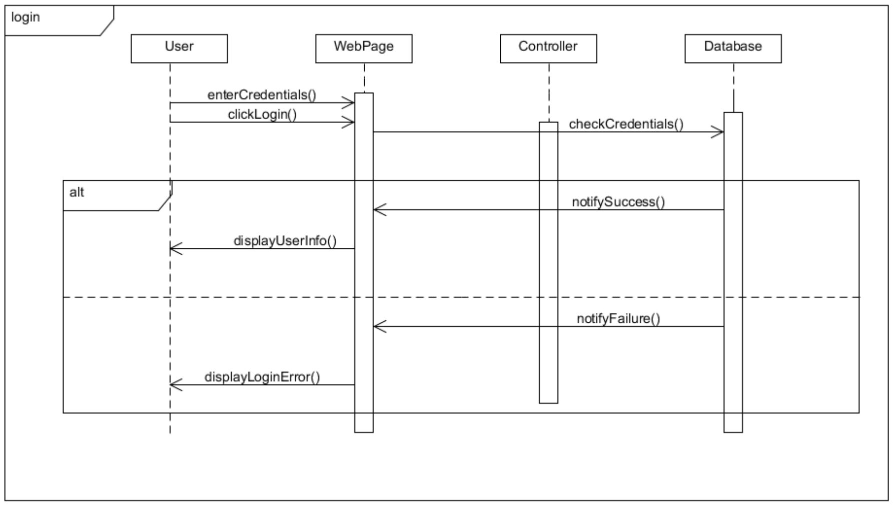
7. Interaction Diagrams:

UC1(CreateProfile), UC4(ChangeProfile) & UC13(DeleteProfile)



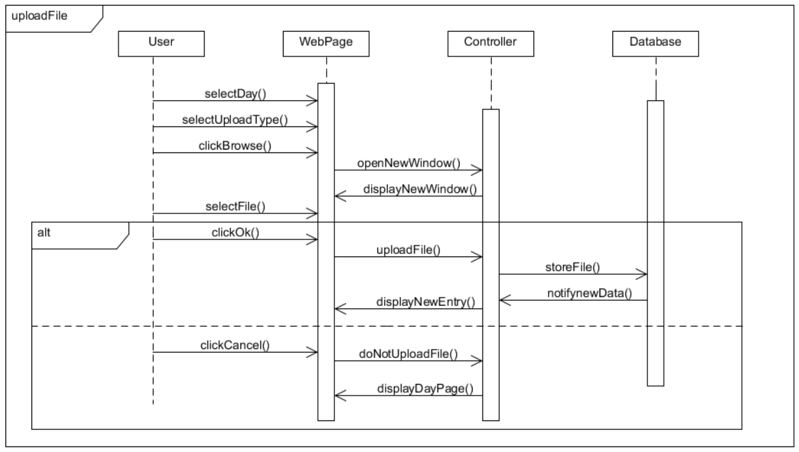
The above diagram collates the three use cases related to profile i.e. Create Profile, Change Profile and Delete Profile. We are assuming that all the database connections are there and the user has valid user Id and password and is logged in. The user send a createProfile() request to the web page which in turn request the controller for it using newProfile(). Controller acknowledges this request by calling the displayBlankProfile() and directing to a new page. The user enters his detail and saves the profile. The controller check the database to see if the userId is already present in the system or not by calling checkUserId(). If it’s not there the user profile is created, else an error is thrown stating that the user Id is already present. For change profile the user clicks on the edit profile in the my account which calls the editProfile(). Now the controller checks the database and retrieve the profile by calling retrieveProfile() and then user can update the profile. Now if the user wants to save he clicks OK and the changes are saved, else on clicking CANCEL the changes if any are not saved in the database. If the user wants to delete the profile he’ll click on delete which calls the clickDeleteProfile(), if confirmed the profile is deleted from the database else the user is brought back to his home page.

UC2(Login)



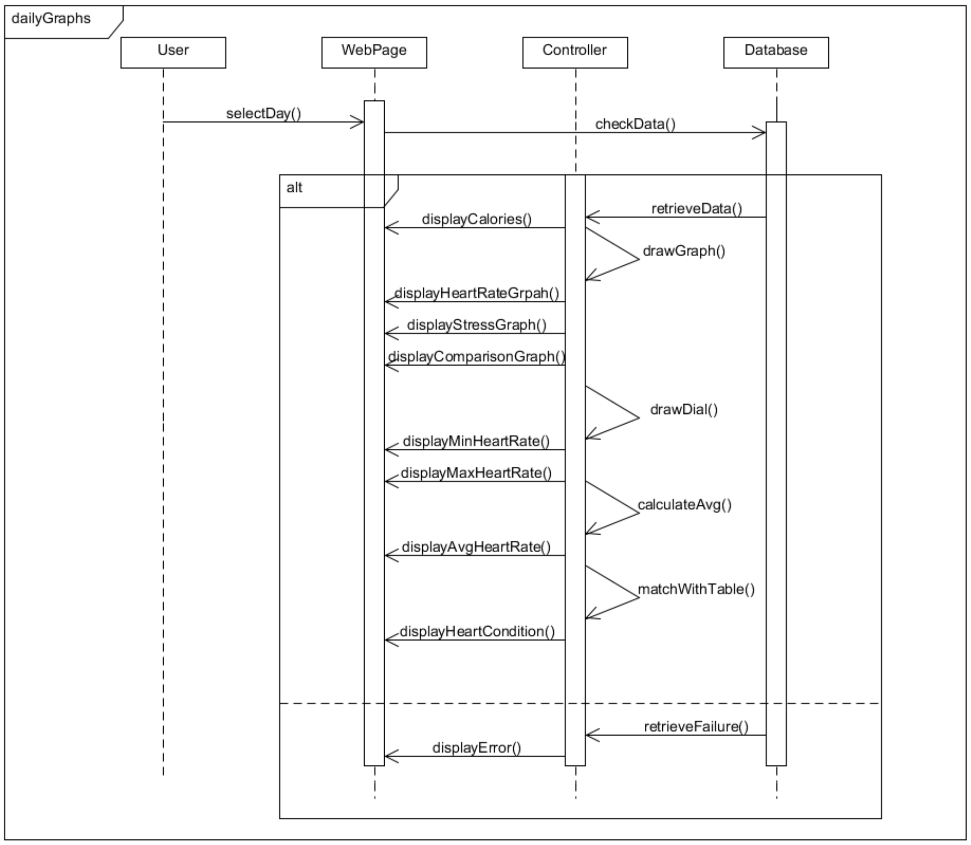
This is the basic use case where the user enters the credentials to enter the system. The user enter the User Id and password and clicks on Login. Now the webpage checks the validity of the credentials entered by calling the checkCredentials(). If the userId and Password are there in the database the notifySuccess() is called and user enter the home page else notifyFailure() is called and Login Error is displayed.

UC3(UploadFile)



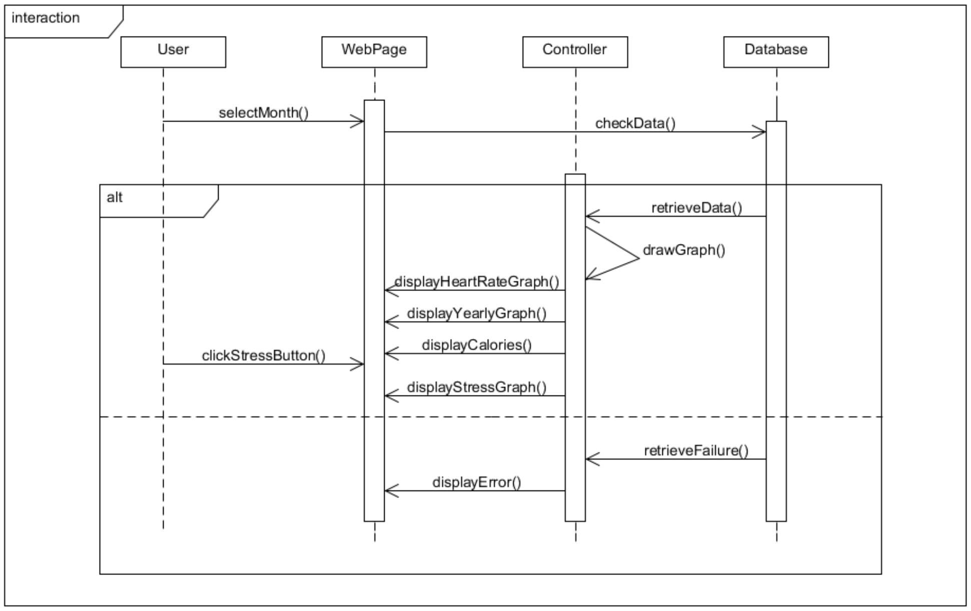
This is the interaction diagram for the upload file use case. Here the user selects the day for which the file has to be uploaded. The type of file to be uploaded is selected. The browse option allows to select file from the hard disk. Once selected the user can either click ok and upload the file or click cancel to stop the upload.

UC5(ViewDailyHeartRate),UC6(viewDailyCaloriesBurnt),UC7(viewDailyStressLevel),UC11(viewHeartCondition),UC12(calculateDailyAverageHeartRate)



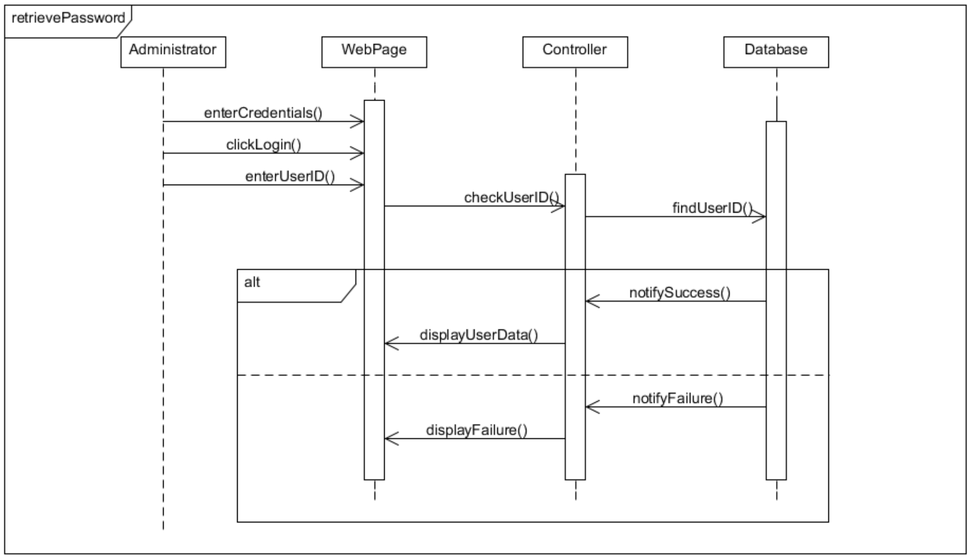
The interaction diagram shown above collates five different use cases for daily graph display for various parameters. The user selects the day for which graphs have to be displayed. The webpage checks whether any data for that day is present or not by calling checkData() function. If the data is present then various graphs of heart rate, calories bunt, stress level are displayed. Dials for min, max and average heart rate are displayed along with the text for daily heart condition is displayed. Else error message is displayed.

UC8(ViewMonthlyHeartRate),UC9(viewMonthlyCaloriesBurnt),UC10(viewMonthlyStressLevel), UC15(ViewYearlyGraph)



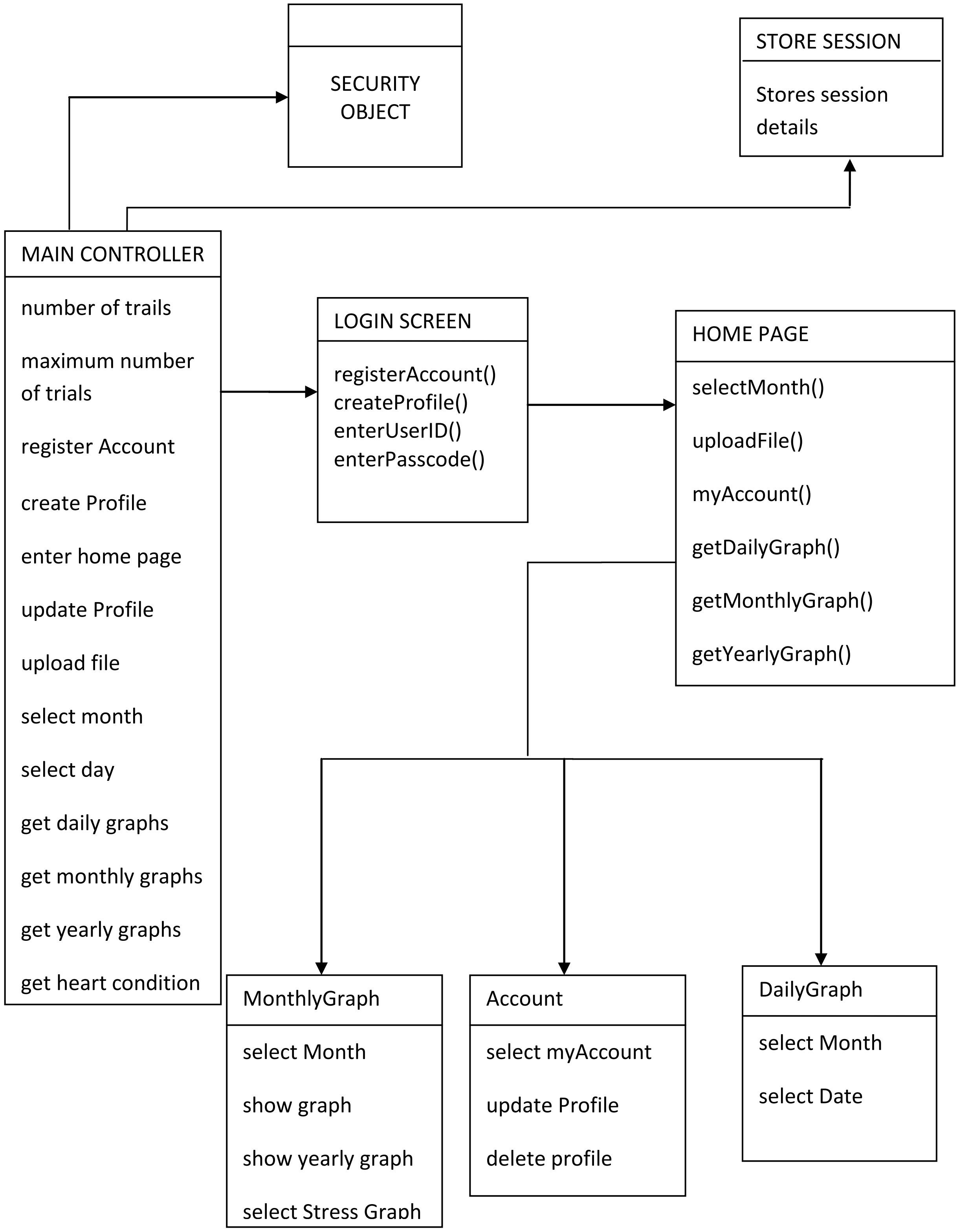
The interaction diagram shown above collates three different use cases for monthly graph display for various parameters. The user selects the month and on checking the data base by calling the checkData() function. If the data is present it is retrieved by retrieveData() function. On retrieving the data monthly graphs are plotted and displayed. Along with this yearly graph is also plotted in the same by calling displayYearlyGraph(). If no data is present for that particular month then error message is displayed.

UC14(retrievePassword)



This interaction diagram shows how the administrator retrieves the password for a user who has forgotten it. The administrator enters his credentials to log in. Then the administrator searches for the userid that requested for the password. The webpage calls the checkUserID() function and the controller then looks into the database by calling findUserID() function. If the user is present the details are displayed or else failure message is generated.

8. System Class Diagram and Interface Specification



8.a. Class Diagram Description

NAME	**DOCUMENTATION**
MAIN CONTROLLER	The main controller controls all the activities of the application like login, controlling the activities of the web pages and accessing them.
SECURITY OBJECT	It authenticate if proper user is entering the website.
LOGIN SCREEN	It’s the first page any user will see when they start the system. It will prompt for userId and password. If “new user” then registers profile/create personal profile.
HOME PAGE	When the user enters correct credential, the system takes the user to this screen. This includes all the month, account info etc. for the user to check.
SESSION STORER	This stores session for a user for each login.

8.b. Class Diagram Attributes and their Description

MAIN CONTROLLER	
Number of trails	it describes the number of attempts a user is allowed to make
Maximum number of trials	it describes the number of attempts a user is allowed to make
Register Account	Ask the user to give userId and password to create a new account.
Create Profile	User enters his personal information such as name, age, gender , weight, height and previous heart condition
enter home page	after successful login the user is taken to the enter home page
select month	select the month of which the data we want to see
Upload file	the user enters his data file into the system
select day	select the day of which you want to see the data
get daily graphs	daily graph showing the heart rate, min heart rate, maximum heart rate , average heart, stress level and calories burnt
get monthly graphs	Monthly graph showing the heart rate, min heart rate, maximum heart rate , average heart, stress level and calories burnt
get yearly graph	Monthly graph will have a timeline scroll bar which when dragged will show the yearly graph of heart rate
get heart condition	Heart rate condition that is if a person is suffering from Bradycardia, Tachycardia or Cardiovascular Mortality is shown.

LOGIN SCREEN	
enter userId	the user needs to enter his user ID
enter password	the user needs to enter his password

SECURITY OBJECT	
check login	It authenticate the userID
check password	It authenticate if the userID has given the correct password corresponding to the userID

SESSION STORE	
store session	It stores the user session for each login. After the user logout the back button will not allow the user to login without entering his credentials

SERVICE PAGE	
get Heart Condition	This function diagnoses heart conditions using pre-defined table in the database
get Monthly Graph	This function produces the monthly graph for heart rate, calories burnt and stress level
get Daily Graph	This function produces the daily graph for heart rate, calories burnt and stress level
my Account	This has the personal information for the user.

8.c. Traceability matrix

Class / Domain Concept	Create Account	update Profile	Upload file	select day	get daily graphs	get monthly graphs	get heart condition
Security Object	X						
Login	X						
Database	X	X	X	X	X	X	X
Calculator					X	X	X
Controller	X	X	X	X	X	X	X
Display					X	X	X

8.d. Design patterns

The Singleton pattern ensures that only one instance of a class exists. This pattern helps a class exercise direct control over the number of instances created. You can create a Singleton class that contains the code to create one instance and return this instance whenever it receives a request for an object reference. The single instance may control data centrally. For example, the singleton class can manage a connection string or a communication port that can be accessed by only one application at a time. As our application needs a database connection we implemented singleton class for our application.

The Factory Method pattern defines an interface, which defers object instantiation to derived classes. Derived classes of this interface are used to instantiate objects, which are specific to the needs of the application at run time.

As the registration is dynamic, we use the factory pattern to build the user class depending on whether the user is a patient or administrator.

8.e. Object Constraint Language (OCL) Contracts

A UML diagram, such as a class diagram, is typically not refined enough to provide all the relevant aspects of a specification. There is, among other things, a need to describe additional constraints about the objects in the model. Such constraints are often described in natural language. Practice has shown that this will always result in ambiguities. In order to write unambiguous constraints, so-called formal languages have been developed. The disadvantage of traditional formal languages is that they are usable to persons with a string mathematical back ground, but difficult for the average business or system modeler to use.

OCL has been developed to fill this gap. I t is a formal language that remains easy to read and write. OCL is a pure expression language; therefore, an OCL expression is guaranteed to be without side effect. When an OCL expression is evaluated, it simply returns a value. It cannot change anything in the model. This means that the state of the system will never change because of the evaluation of an OCL expression, even though an OCL expression can be used to specify a state change (e.g., in a post-condition).

For the **getHeartCondition** class has the precondition that the input data atleast once has been uploaded by the user .The invariant condition for this class is particular records in the month cannot be deleted when the class methods are in progress.The post condition for this class would be the graph display.

For the **getMonthlyGraph** class has the precondition that the input data atleast once has been uploaded by the user .The invariant condition for this class is particular records in the month cannot be deleted when the class methods are in progress.The post condition for this class would be the graph display.

For the **getDailyGraph** class has the precondition that the input data atleast once has been uploaded by the user .The invariant condition for this class is particular records in the month cannot be deleted when the class methods are in progress.The post condition for this class would be the graph display.

9. System Architecture and System Design

9.a Architectural Styles: 3-Tier:

The heart monitoring system uses the three- tier architecture. The three tier architecture has, as the name suggests, 3 layers. The Presentation layer which deals with the user interface or the look of the website. It helps the user to interact the system in a smooth manner giving the ability to understand the results. The logic tier is the "brain" of our system. It helps in processing various logical computations and algorithms which give user an understandable result of the raw data uploaded by the user. The data tier stores all the user data and also the logic tier retrieves data from this tier for processing.

In our system the presentation tier is handled by Drupal [4] which is a content management framework. The logical tier is handled by PHP script and database is MySQL.

The figure 1 shows a basic 3-tier architecture design.

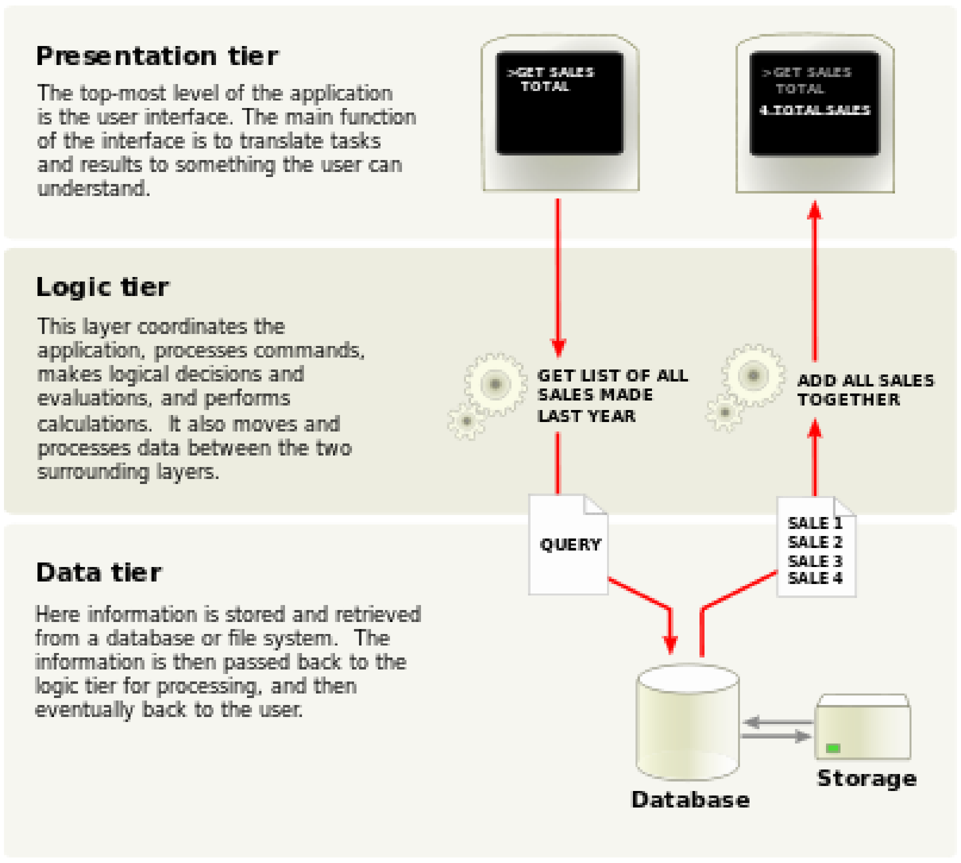

Presentation tier
The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.
>GET SALES TOTAL
>GET SALES TOTAL
4 TOTAL SALES
Logic tier
This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.
GET LIST OF ALL SALES MADE LAST YEAR
ADD ALL SALES TOGETHER
QUERY
SALE 1
SALE 2
SALE 3
SALE 4
Data tier
Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.
Database
Storage

Fig 9.1 3-tier architecture design

9.b Identifying subsystems:

The website is an online service and it requires the client to interact with the user and a server to maintain records of all the operations. The user accesses the web browser through a http protocol. The client uses PHP script and makes function calls to the server in background. Within the server, the client updates or retrieves the tables in the MySQL database whenever requested by the user.

The subsystems can thus be classified as User, which interacts with the client to perform required operation, Client, which interacts with the database and performs functions in the background to present the information requested by the user and Server, which is essentially a database maintaining all the information. The following diagram demonstrates how the user-client- server interactions take place.

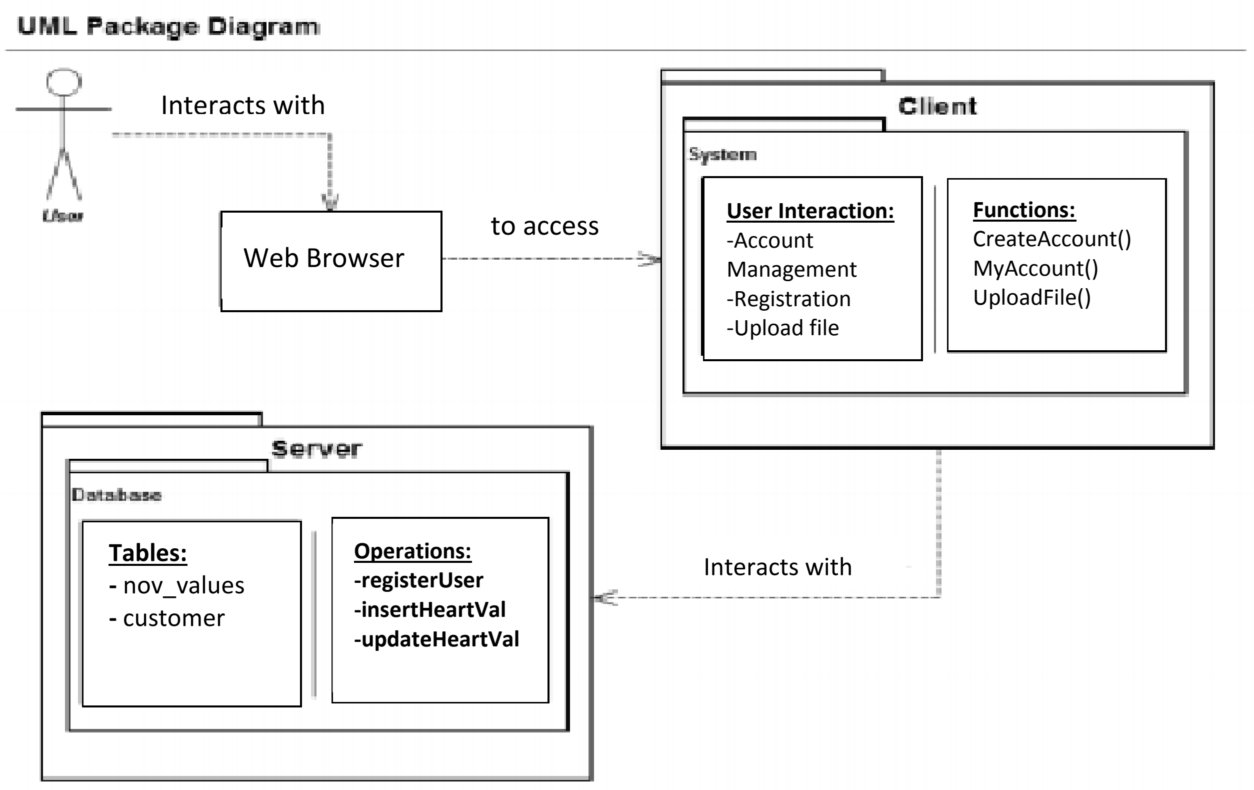


Fig 9.2 Client-server interaction diagram

9.c Mapping subsystems to hardware:

The data collected in this system is stored in a database, which is located in the server. The server also stores the graphical user interface. The server that we use is a web server which keeps track of the customer information, heart rate values and stress levels of each customer and heart rate history.

The client computers interact with the server and will be able to run the system on the server. So, the client computers access the web server using a browser to perform certain operations like creating a user account, updating the user information and uploading a file.

9.d Persistent Data Storage:

MySQL was chosen as the database for our system. The basic tables used in the database are as follows:

UserProfile

heartCondition

calorieburnt

LoginInfo

As our logic is in our application we can directly determine the stress level and heart condition using the heart rate. Hence we need not store this information daily in the database.

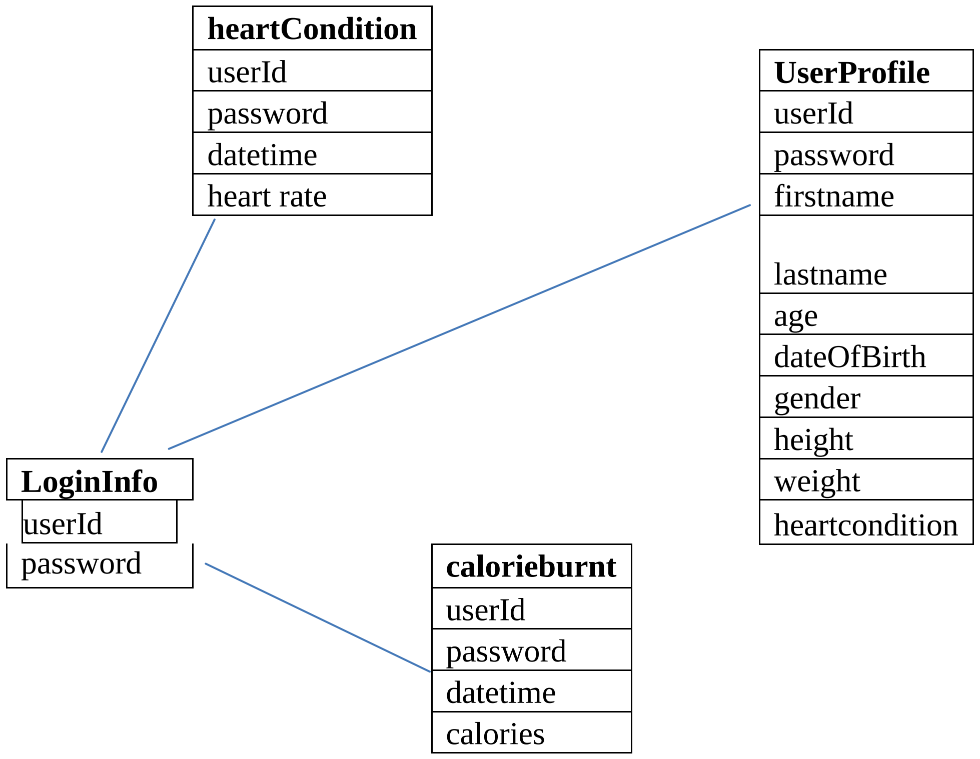


Fig 9.3 Persistent Data Storage diagram

9.e Network protocol

HTTP functions as a request-response protocol in the client-server computing model. A web browser, for example, may be the client [5] and an application running on a computer hosting a web site may be the server [6]. The client submits an HTTP request message to the server. The server, which provides resources such as HTML files and other content, or performs other functions on behalf of the client, returns a response message to the client. The response contains completion status information about the request and may also contain requested content in its message body.

Security needs

As we need to encrypt our calls or authenticate our client, we decided to use HTTP-based application, whether that is an ASP.NET application or a remote application. .

Interoperation

HTTP application is best suited for heterogeneous systems and as our functionality is over web, we need a protocol which is interoperable over different inter networks.

Scalability

One challenge working with website is the number of users increase exponentially and we need a protocol which is scalable and as HTTP protocol is highly scalable which would suit over requirements.

The message formats for a HTTP protocol are clearly described in the below link. http://www.w3.org/Protocols/rfc2616/rfc2616-sec4.html

9.f Global control flow

The system is event driven. The user interacts with the system by first registering his user name and password. The user log in and the next step is the user upload the necessary data. This data gets stores into the database. Once the user clicks on the data he wants to see the proper output is shown in the form of daily and monthly graphs. The system allows multiple users to login and upload data.

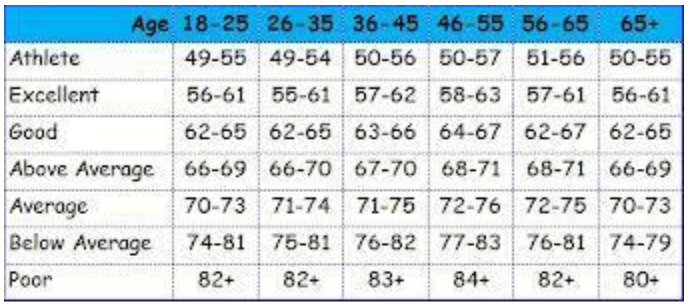
9.g Hardware Requirement

The operating system required by our application is Windows XP/Vista 7, Mac. The web server is WAMP. Internet connection is necessary for the application. The application needs a minimum of 2Gbytes hard disk storage. Colour display is mandatory with a minimum resolution of 640 X 480 pixels as we are using graphs and charts in our user interfaces. These are the main hardware requirement for our system.

10. Algorithms and Data structure

The time for action is now. It's never too late to do something.-Carl Sandburg

The heart is the most vital part of our body so it is important that we monitor it to keep our body at the best of its health. Heart rate is determined by the number of times your heart beats each minute, is an important measure of your health. How hard your heart has to work during various activities can tell you a lot about your overall physical condition. The normal heart rate of the human body is 60-100 beats per minute [7]. Generally, a lower heart rate at rest implies more efficient heart function and better cardiovascular fitness. For example, a well-trained athlete might have a normal resting heart rate closer to 40 beats a minute.



Age	18-25	26-35	36-45	46-55	56-65	65+
Athlete	49-55	49-54	50-56	50-57	51-56	50-55
Excellent	56-61	55-61	57-62	58-63	57-61	56-61
Good	62-65	62-65	63-66	64-67	62-67	62-65
Above Average	66-69	66-70	67-70	68-71	68-71	66-69
Average	70-73	71-74	71-75	72-76	72-75	70-73
Below Average	74-81	75-81	76-82	77-83	76-81	74-79
Poor	82+	82+	83+	84+	82+	80+

Fig 10.1: Resting heart rate for men with corresponding ages

To measure your heart rate, simply check your pulse. Place your index and third fingers on your neck to the side of your windpipe. To check your pulse at your wrist, place two fingers between the bone and the tendon over your radial artery — which is located on the thumb side of your wrist.

When you feel your pulse, count the number of beats in 15 seconds. Multiply this number by 4 to calculate your beats per minute.
Keep in mind that many factors can influence heart rate, including:

Activity level
Fitness level
Air temperature
Body position (standing up or lying down, for example)
Emotions
Body size
Medications

Although there's a wide range of normal, an unusually high or low heart rate may indicate an underlying problem. If your resting heart rate is consistently above 100 beats a minute it is a condition called tachycardia or if it is below 60 beats a minute it is called bradycardia. It could be accompanied by other symptoms such as fainting, dizziness or shortness of breath. Hence it is very important that we detect these conditions at the earliest stage possible. Our system offers this by the continuous monitoring of the heart rate data.

Based on the age of the customer, we compare the heart rate values with average heart rate values and diagnose the heart condition of the customer.

Pseudocode:

```
Retrieve resting heart rate of the day from database.
Find the maximum heart rate from the readings.
Retrieve age of the user from the database.
If it is lower (<50) declare to the user that he might have bradycardia
If it is higher (>100) declare to the user that he might have tachycardia.
Else the heart is working in normal condition.
```

Data Structures:

The data structure that we use in this project is databases. Database is used for primarily storing heart rate data, login information (such as user id and password) of the user as well as the administrator and the results of the heart data collected. It stores the user id and password of the users so that when a user forgets his password and sends a request for it to the administrator, the administrator can refer to the database table and retrieve the password. The following are the data types to store the variables.

User id: string
Password: string
Total energy expenditure: float
Number of steps: int
Resting heart rate: float
Active heart rate: float
Heart rate duration: float

11 User Interface Design and Implementation

In our report #1 we showed the screen mock-ups in which the user was able to see his heart rate, stress level and calories burnt in different screen as shown below. But now we have incorporated all the three graphs into one page, thereby leading to simpler design and less user effort.

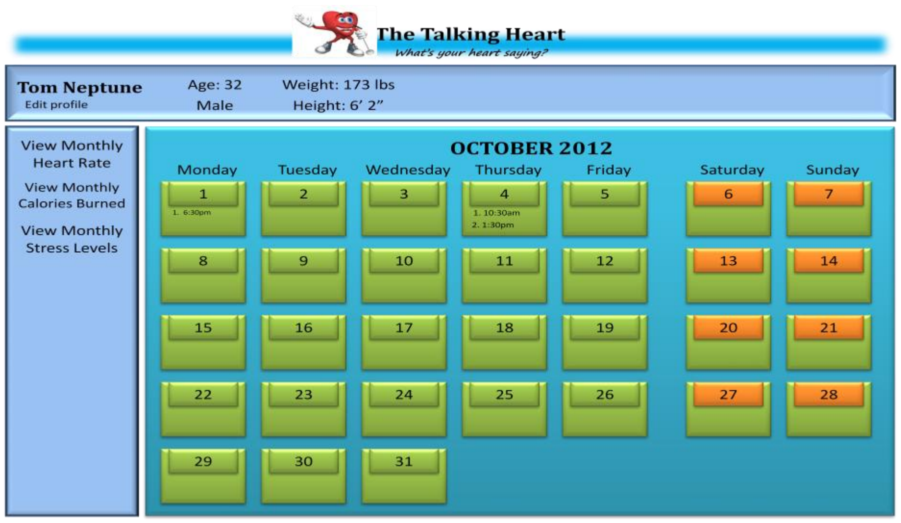


Fig 11.1 Initial screen for timeline navigation

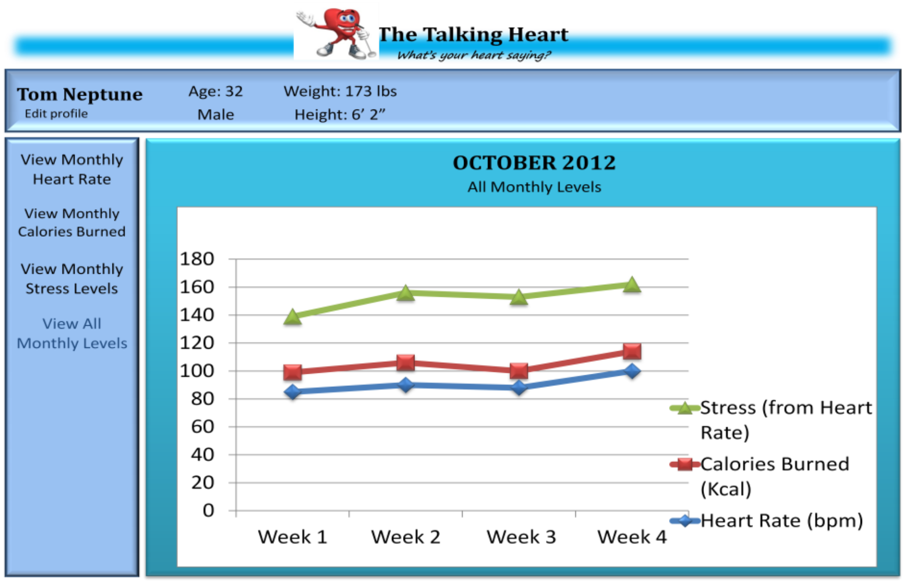


Fig 11.2 Initial screen for monthly values

Also we have included minimum average and maximum heart rate and instead of just writing them in a text form we are showing it visually by the help of dials. In these dials we have red areas which are the "danger zones" signifying heart levels which are considered to be unhealthy.

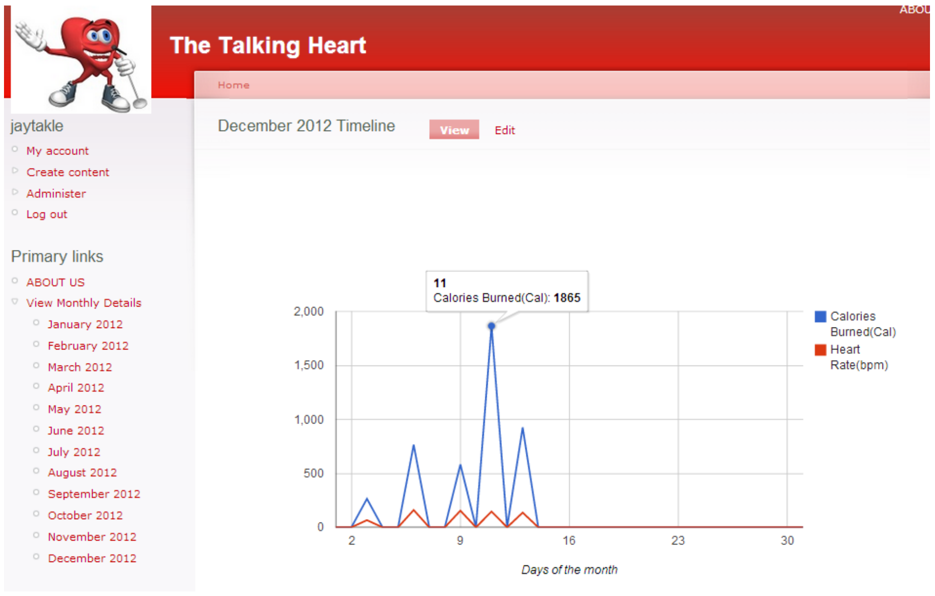


Fig 11.3 Current improved screen for navigation as well as monthly views

There is another enhancement made to the screen mockups. We have added a bar graph. In this we have our average resting heart rate which is compared with a table which provides the average heart rate for athletes, an excellent /good/average/poor heart rates according to your age group. So instead of providing that table and making it difficult to understand we provide the bar graph as shown below. The first bar shows the users heart rate followed by the average heart rates for various categories as described above.

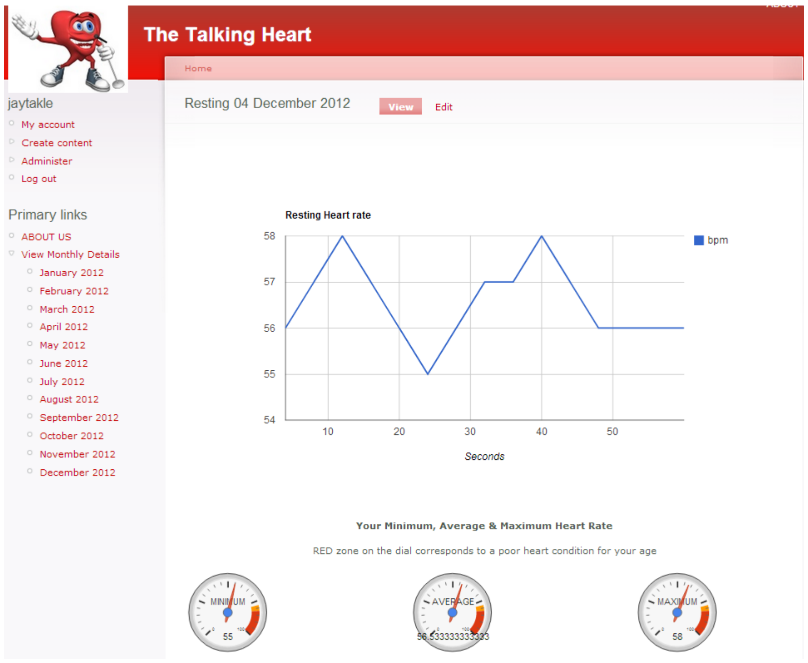


Fig 11.4 Current screen for daily view of heart rate graph

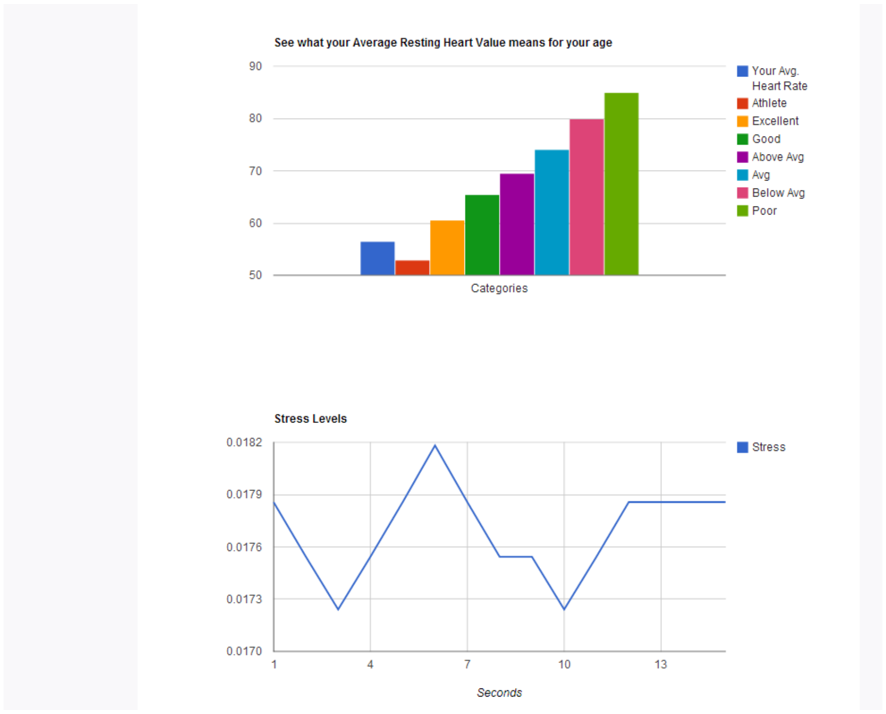


Fig 11.5 Current screen for daily view of heart rate comparison and stress

12. DESIGN OF TEST CASES

ACCEPTANCE TEST CASES

Acceptance tests are created from user stories. During iteration the user stories selected during the iteration planning meeting will be translated into acceptance tests. A story can have one or many acceptance tests, whatever it takes to ensure the functionality works. Acceptance tests are black box system tests. Each acceptance test represents some expected result from the system. Customers are responsible for verifying the correctness of the acceptance tests and reviewing test scores to decide which failed tests are of highest priority. Acceptance tests are also used as regression tests prior to a production release. Section 2.4 clearly describes acceptance test scenarios associated.

Acceptance Test case for REQ1
ATC1.01- Create a profile of a user who is not in the system. (Pass)
ATC1.02- Create a profile of a user who is already in the system. (Fail)

Acceptance Test case for REQ2
ATC2.01 – Upload a .csv file (Pass)
ATC2.02 – Upload any other file extension (Fail)

Acceptance Test case for REQ3
ATC3.01- Update the personal information of a user. (Pass)

Acceptance Test case for REQ4
ATC4.01- Access the user profile who is not currently logged in. (Fail)
ATC4.02- User accesses his own profile. (Pass)
ATC4.03- The system logs out after being idle for 10 minutes. (Pass)
ATC4.04- The user enters the system after 3 consecutive failed attempts to login. (Fail)
ATC4.05 The user retrieves his password by answering the secret question (Pass)
ATC4.06- The user deletes his profile. (Pass)
ATC4.07- The user tries to login with a wrong user id or password. (Fail)

Acceptance Test case for REQ5
ATC5.01- The system tries to display a graph from a future date. (Fail)
ATC5.02- The system tries to display a graph from a particular date in the past. (Pass)

Acceptance Test case for REQ6
ATC6.01- The system displays "heart condition" for an entry in the range of 60-70 seconds. (Pass)
ATC6.02- The system tries to displays "heart condition" for an entry less than 60 second or more than 60 second. (Fail)

Acceptance Test case for REQ7

ATC7.01-The system displays the list of all the user ID and password when the administrator login (Pass)
ATC7.02-The system does not show any user ID and password when the administrator login (Fail)

Below are the design of the test cases as per required by out project. These are the basic test cases. We need to check thoroughly that all the test cases are working properly, the screens are displayed in a proper format and there is no discrepancy in displayed graphs. We also need to check the code thoroughly for the correctness of algorithm and control flows.

1. Test ID : TC1_Login
   Assumption: the user is at login screen.

Input Requirement	**Expected Output**	**Pass/Fail**	**Comments**
Valid userId ,Password	Login successfully	Pass if the user is able to login and sees the home screen	This is to test that the system allow valid user to enter the website
Invalid UserId ,password	Display error message "Invalid userId/Password"	Pass if the user is unable to see the home screen	This test is to determine that invalid user should not enter the website

2. Test ID: TC2_Register:
   Assumption: the user is at the login screen.

Input Requirement	**Expected Output**	**Pass/Fail**	**Comments**
the user enters a userId which is unique	Register successfully	Pass if the user is able to register his UserId password.	This is to test that the system allow only unique user in the system with the unique userID
the user enters a userId which is pre-existing	Display error message "User Id already exist. Please choose another ID"	Pass if the user is unable to create a userId	This is to test that the system does not allow two user with the same userID

3. Test ID: TC3_Upload
   Assumption: the user is login and on the upload file screen

Input Requirement	**Expected Output**	**Pass/Fail**	**Comments**
the user tries to load a file with .csv file format and one column of heart reading	file upload successful	Pass if the user is able to upload the file and shown message “Successful upload"	This test is to determine that the system allows only .csv file to be loaded successfully
the user tries to load a file with .xls file format and one column of heart reading	Display error message "Incorrect file "	Pass if the user is unable to load the file and shown message "Unsuccessful load"	This test is to determine that the system does not allow other files to be loaded.
the user tries to load a file with .csv file format and two or three column of reading	Display error message "Incorrect file "	Pass if the user is unable to load the file and shown message "Unsuccessful load"	This test is to determine that the system allow only .csv file with proper format to be uploaded.

4. Test ID: TC4_viewDailyData
   Assumption: User is login and at the calendar screen.

Input Requirement	**Expected Output**	**Pass/Fail**	**Comments**
the user clicks on the day to see the uploaded data	the user is taken to the day screen and graphs are displayed with proper markings on x-axis and y-axis	Pass if the user is able to see his daily heart rate graph, daily minimum maximum heart rate and stress level.	This test is to determine that the user is able to see his heart graphs properly.
the user clicks on the day where no data is uploaded	Display message "no data uploaded"	Pass if the user is unable to see any graph	This test is to determine that the system does not display graph for day when no data is uploaded.

5. Test ID : TC5_viewMonthlyData
   Assumption: User is login and at the calendar screen.

Input Requirement	**Expected Output**	**Pass/Fail**	**Comments**
the user clicks on the month to see the uploaded data	the user is taken to the month screen and graphs are displayed with proper markings on x-axis and y-axis	Pass if the user is able to see his monthly heart rate graph, monthly minimum maximum heart rate and stress level.	This test is to determine that the user is able to see his daily heart graphs properly.
the user clicks on the month where no data is uploaded	Display message "no data uploaded"	Pass if the user is unable to see any graph	This test is to determine that the system does not display graph for a month when no data is uploaded.

6. Test ID: TC6_viewHeartCondition
   Assumption: the user is at login screen and at the home screen.

Input Requirement	**Expected Output**	**Pass/Fail**	**Comments**
The user clicks on view heart condition.	The user is shown his heart condition.	Pass if the user is able to see his heart condition	This test is to determine that the user is able to see their heart condition properly.
the user clicks on view heart condition when no data is uploaded for the month	Display message "no data uploaded"	Pass if the user is unable to see his heart condition	This test is to determine that the system does not display heart condition when no data is uploaded.

7. Test ID: TC7_updateProfile
   Assumption: the user is at login screen and at the “my account” screen.

Input Requirement	Expected Output	Pass/Fail	Comments
The user clicks on update profile and updates proper data.	the user is able to update his profile successfully	Pass if user is able to update his profile	This test is to determine that the system allows the user to update his profile successfully.
The user clicks on update profile and updates with incorrect data.	Display message "incorrect format please try again"	Pass if user is unable to update his profile with incorrect data.	This test is to determine that the system allows the user to update his profile successfully with correct format data.

UNIT TEST CASES

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each unit is tested separately before integrating them into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of defects are identified during its use. Since we are writing codes using PHP we choose to use the PHP Unit Testing. As we identify each class in the component is extensively tested using the PHP unit testing module . In writing the tests before we wrote the class, we have been forced to do a few things. First of all, we had to invent the class’s public API before anything else. It’s worth noting than in Unit Testing and TDD, it is virtually pointless to test private methods (what’s called “Testing State”) since we’re rarely interested in how the code does something as opposed to what the expected end result should be (focus on results, not intermediate state). In a lot of code the result will remain the same, but the working code will evolve over time due to refactoring, new PHP functionality, or system dependent requirements – so testing all this stuff only forces us to constantly rewrite our tests. Once we get down to actual coding, we have all this design work already done and a set of tests which will continually verify the new code we write.

INTEGRATION TEST CASES

Integration testing is a logical extension of unit testing. In its simplest form, two units that have already been tested are combined into a component and the interface between them is tested. A component, in this sense, refers to an integrated aggregate of more than one unit. In a realistic scenario, many units are combined into components, which are in turn aggregated into even larger parts of the program. The idea is to test combinations of pieces and eventually expand the process to test your modules with those of other groups. Eventually all the modules making up a process are tested together. Beyond that, if the program is composed of more than one process, they should be tested in pairs rather than all at once.

Integration testing identifies problems that occur when units are combined. By using a test plan that requires you to test each unit and ensure the viability of each before combining units, you know that any errors discovered when combining units are likely related to the interface between units. This method reduces the number of possibilities to a far simpler level of analysis.

You can do **integration testing** in a variety of ways and we have chosen the Umbrella approach:

> **The umbrella approach** requires testing along functional data and control-flow paths. First, the inputs for functions are integrated in the bottom-up pattern discussed above. The outputs for each function are then integrated in the top-down manner. The primary advantage of this approach is the degree of support for early release of limited functionality. It also helps minimize the need for stubs and drivers. We have chosen this model as the functional issues can be detected early and also as our project requires maintenance, the regression testing scenario would not be a overhead for following this pattern in testing.

CODE COVERAGE:

Code coverage is a measure used in software testing. It describes the degree to which the source code of a program has been tested. We have used XDEBUG Extension and it gives us the function coverage statistics as well as statement coverage statistics .We have observed that the function coverage was 100% ,however statement coverage was 60% as many branches would never be executed with any piece of testing .

13. History of Work, Current status and Future Work

History of Work & Current Status

The following table list the milestone along with their completion date. These objectives had pre-assigned dates which we were able to meet. We were also able to revise the reports by incorporating the reviews given by the peer groups.

Completion of First Report	October 15th,2012
Completion of Second Report	November 19th,2012
Completion of First Demo	November 9th,2012
Major Revision of First Report	November 5th,2012
Major Revision of Second Report	December 5th,2012
Implementation	December 9th,2012
Unit and Integration Testing	December 12th,2012
Completion of Second Demo	December 13th,2012
Completion of Third Report	December 14th,2012

In the first demo showed a calendar type navigation which included the daily heart rate and stress level graph. We also computed the minimum, maximum and average heart rate and showed it in the form of a dial. There was a comparison graph shown between a user heart rate and heart rates for a good, average and athletic heart rate for the person of the same age. The basic of login, profile update and delete were done in this demo.

In the final demo all the various components were completed, integrated and tested together. The File upload module was made foolproof so that user can upload only.csv file. The monthly heart rate and calories computation were made functional. As suggested by Dr. Marsic the calendar style navigation was scrapped and the new timeline navigation system was implemented to improve the visual appeal of the system. We also computed the calories burnt for an activity e.g. for exercising, sleeping and resting. The heart condition which was the crux of our system was also made available to the user. We were not able to implement the monthly stress level as we were unable to quantify as to how more variation in heart rate corresponds to lesser stress.

We took Dr. Marsic's resting and exercising heart rate and fed it into the system. These records were analysed by our system which gave us three results. First, that Dr. Marsic's heart condition is normal(and not Tachycardia or Bradycardia), secondly that the professor has athletic heart rate and lastly that during the exercise the maximum heart rate went upto 125bpm which was less than the maximum bpm for the person of his age and he could push himself more during the exercise.

Some our key accomplishments are:

- We were able to show the raw data in a graphical representation
- We were able to compare the raw data with the data present in the literature

● We were able to show the heart condition (Normal, Tachycardia and Bradycardia) for the user

● We were able to foolproof the system of upload file.

● We implemented all Use Cases into the website.

Breakdown of Responsibilities:
The team comprises of 6 members.
The following are the modules developed.

Login&Registration	Anusha/Anu
UploadFile	Prasoon/Vinayak
viewDailyHeartRate	Prasoon/Anu
viewDailyStress	Anusha/Bhumika
viewHeartCondition	Jay/Bhumika
viewCaloriesBurnt	Jay/Anusha
viewMonthlyHeartRate	Jay/Bhumika
viewProfile	Prasoon/Vinayak
updateProfile	Vinayak/Anu

We have equally divided the work among ourselves. The integration was done by Jay, Prasoon and Bhumika. The integration testing was performed by Anusha,Vinayak and Anu.

In this project we have accomplished our goals we set out with. Next we will conclude and discuss what additional features we thought would work well with our system but could not implement due to time constraints.

Future Work

Our system currently does some level of data processing of the heart rate and does a lot of information visualization as described in lengths throughout the project report. But we think there could be some additional improvements to the project to make it a more informative tool. In addition to that, Dr. Marsic suggested developing statistics by recording more data to analyze different patterns. We enumerate the following points for future work:

Add yearly flexible view:
Currently our system shows overview data for every month, which is also used as a navigation system. We would like to make this timeline flexible with the use of a user initiated slider in the GUI. This would give the user freedom to view data either for a week, month, a couple of months or for the whole year. This feature will completely eliminate the need for separate monthly links on the side panel, making the website look cleaner.

Quantify stress for conclusions:
The one thing we still could not attain was quantifying and concluding the stress graph. Through different papers we have read that the domain experts go through the stress data and quantify objectively. There

isn’t a proper algorithm to quantify whether there is more or less stress according to the heart rate. For the future, we want to do some more research and try to analyze the calculated stress for conclusions.

Add pedometer device for recording:
Garmin supplies an additional device to record steps taken while doing an activity. They call it the ‘Foot Pod’. It is a small device that has to be mounted on the shoe. The company says this device records movement of the foot in 3D. Adding data recorded by this device to our current data can possibly provide some more insight in the correlation between heart rate, steps taken and calories burned.

Develop statistics
During the final demo Dr. Marsic suggested developing statistics from the heart rate data. We can build a histogram for each activity which would give us patterns of the heart beats according to an activity.

References:

[1] Hear Disease Fact Sheet: http://www.cdc.gov/dhdsp/data_statistics/fact_sheets/fs_heart_disease.htm

[2] Levine HJ. "*Rest heart rate and life expectancy*." J Am Coll Cardiol 1997;30:1104–1106.

[3] Palatini P, Casiglia E, Pauletto P, Staessen J, Kaciroti N, Julius S. "*Relationship of tachycardia with high blood pressure and metabolic abnormalities: a study with mixture analysis in three populations.*" Hypertension 1997;30:1267–1273.

[4] Palatini P, Julius S. "*Heart rate and the cardiovascular risk*." J Hypertens 1997;15:3–17.

[5] Palatini P. "*Elevated heart rate as a predictor of increased cardiovascular morbidity*." J Hypertens 1999;17(Suppl. 3):S3–S10.

[6] Palatini P, Thijs L, Staessen JA, Fagard RH, Bulpitt CJ, Clement DL, de Leeuw PW, Jaaskivi M, Leonetti G, Nachev C, O'Brien ET, Parati G, Rodicio JL, Roman E, Sarti C, Tuomilehto J, Systolic Hypertension in Europe (Syst-Eur) Trial Investigators. "*Predictive value of clinic and ambulatory heart rate for mortality in elderly subjects with systolic hypertension*." Arch Intern Med 2002;162:2313–2321.

[7] Palatini P, Casiglia E, Julius S, Pessina AC. "*High heart rate: a risk factor for cardiovascular death in elderly men*." Arch Intern Med 1999;159:585–592.

[8] Pediatric Tachycardia: http://emedicine.medscape.com/article/804613-overview

[9] Sinus Bradycardia: http://emedicine.medscape.com/article/760220-overview

[10] K Fox, JS Borer, AJ Camm *et al*. "*Resting heart rate in cardiovascular disease"* J Am Coll Cardiol, 50 (2007), pp. 823–830

[11] Diaz A, Bourassa M, Guertin M, Tardiff J. "*Long-term prognostic value of resting heart rate in patients with suspected or proven coronary artery disease*." Eur Heart J 2005;26:967–974.

[12] 47Wu, G. Q., Arzeno, N. M., Shen, L. L., Tang, D. K., Zheng, D. A., Zhao, N. Q., Eckberg, D. L., and Poon, C. S., "*Chaotic signatures of heart rate variability and its power spectrum in health, aging and heart failure*," PLoS ONE **4**, e423 _2009_.

[13] J. Taelman, S. Vandeput, A. Spaepen and S. Van Huffel, "*Influence of Mental Stress on Heart Rate and Heart Rate Variability*." J. Vander Sloten, P. Verdonck, M. Nyssen, J. Haueisen (Eds.): ECIFMBE 2008, IFMBE Proceedings 22, pp. 1366-1369, 2008.

[14] Health Care Monitoring of Mobile Patients: http://www.ercim.eu/publication/Ercim_News/enw60/amato.html

[15] "Pedometer" http://en.wikipedia.org/wiki/Pedometer