SOFTWARE ENGINEERING

Sleep Quality Assessment

.....How well do you sleep???

Report 2

Cagdas Karatas	cagdas@cac.rutgers.edu
Chenyun Lan	lan.chenyun@hotmail.com
Jiajun Zhu	zjjjun1991@hotmail.com
Mridula Krishnapur	mridu89@gmail.com
Yangkai Zhou	zhouyangkai1991@hotmail.com
Zeya Zhang	tooth6612@gmail.com

Project Website: here

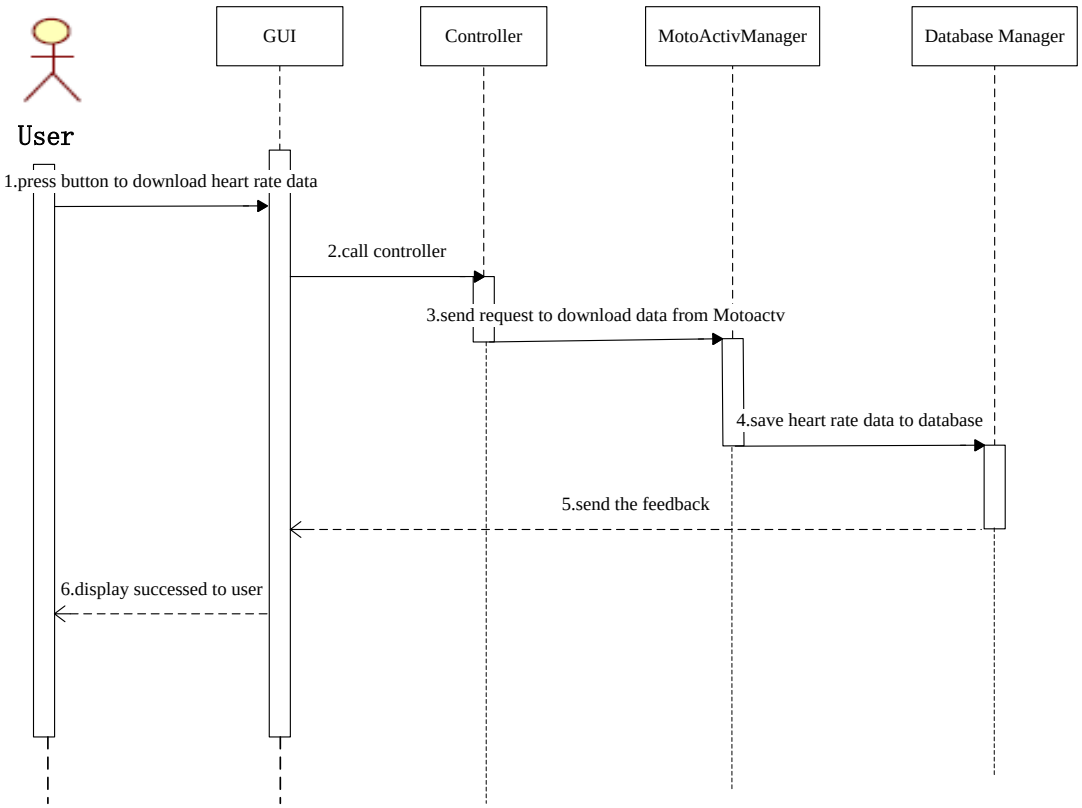
All group members contribute equally.

TABLE OF CONTENTS

Contents

1. INTERACTION DIAGRAMS

UC-1: MonitorSleep



Responsibilities Associated:

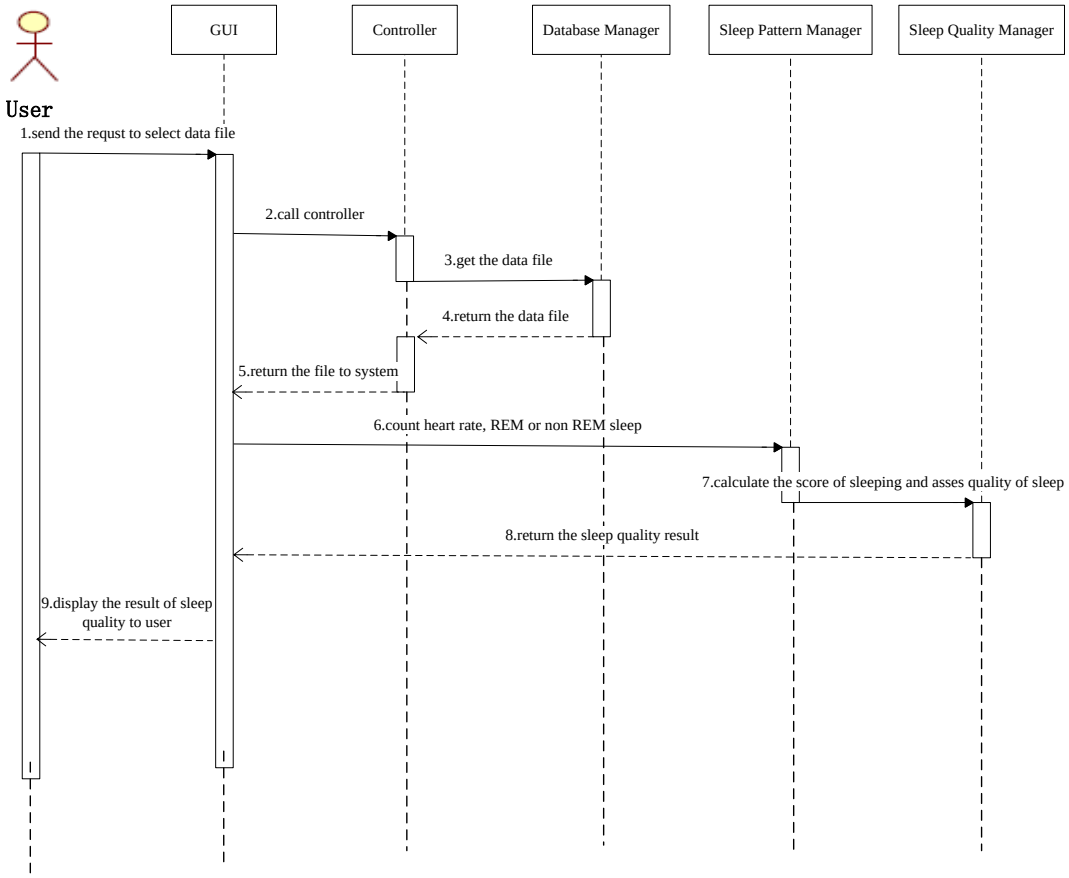
1) MonitorSleep is mainly to collect the sleeping data from device, and download the sleeping data via GUI.
2) MonitorSleep can be used to make sure that the display can show the status of downloading the sleeping data.
3) MotoactvManager is responsible for loading monitoring data from MotoActv device.
4) DatabaseManager is responsiple for saving the sleeping data downloaded from Motoactv device.

Design Principles:

The design principle of MonitorSleep is the Expert Doer Principle. When the system starts to

work the system is contacted and it makes sure to contact the device and database to get and store the data.

UC-2: GetSleepQuality



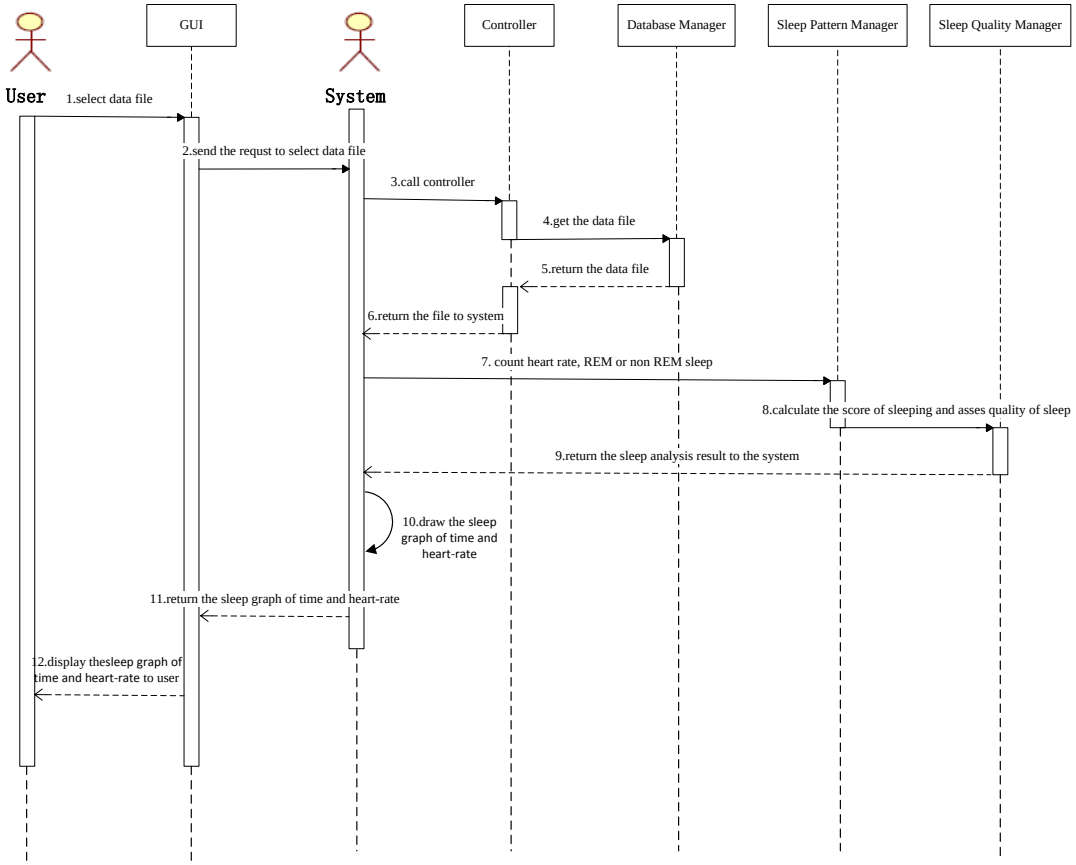
Responsibilities Associated:

1) GetSleepQuality is mainly to let user know their sleep quality.
2) GUI is responsible for displaying the score of sleep quality to the user.
3) SleepPatternManager is responsible for counting heart-rate and time of selected data file.
4) SleepQualityManager is responsible for counting the sleep quality according the sleep pattern.

Design Principles:

GetSleepQuality follows the Expert Doer Principle. Expert Doer Principle states that an object who knows should do the task. GetSleepQuality needs to compute and analysis the heart rate data and get the score of sleep quality, it has the main task to do.

UC-3: GetSleepGraph



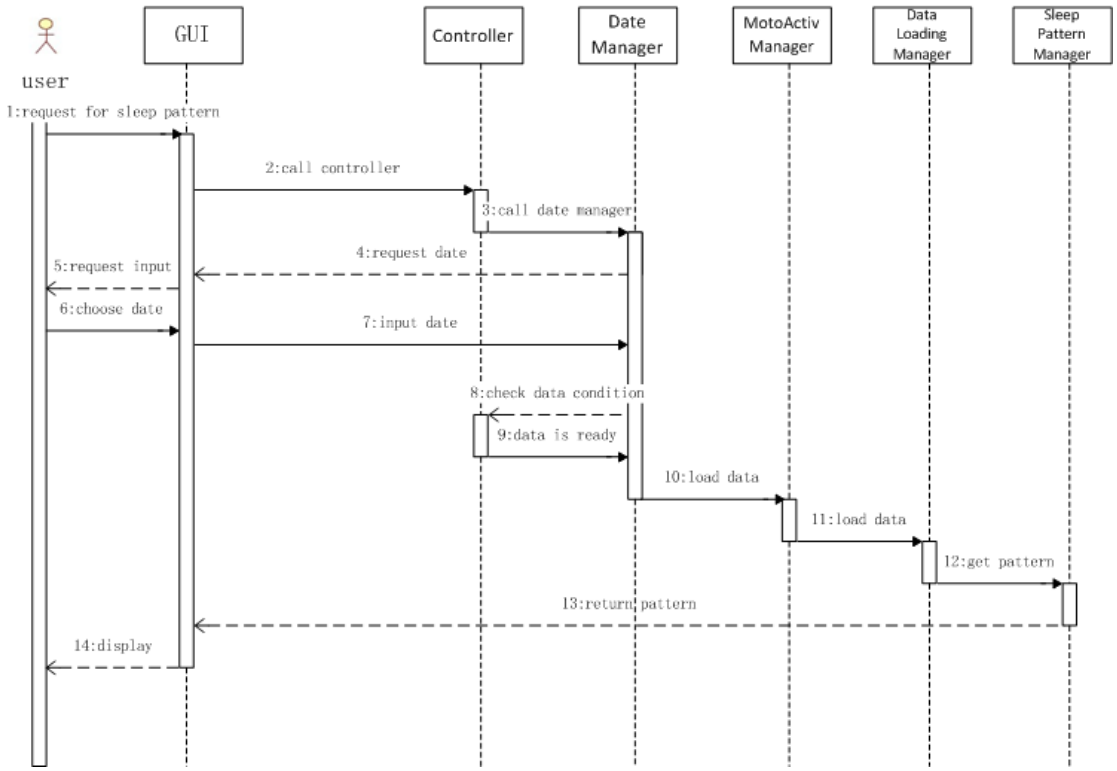
Responsibilities Associated:

1) GetSleepGraph is mainly to let user see their sleep graph of time and heart-rate.
2) GUI is responsible for displaying the sleep graph of time and heart-rate to the user.
3) SleepPatternManager is responsible for counting heart-rate and time of selected data file.
4) SleepQualityManager is responsible for counting the sleep quality according the sleep pattern.

Design Principles:

GetSleepGraph follows the Expert Doer Principle. Expert Doer Principle states that an object who knows should do the task. GetSleepGraph needs to compute and analysis the heart rate data and get the score of sleep quality, it has the main task to do. GetSleepGraph is the only object that communicate with system to draw graphs and display the sleep graph of time and heart-rate to the user via GUI.

UC-4: GetSleepPattern

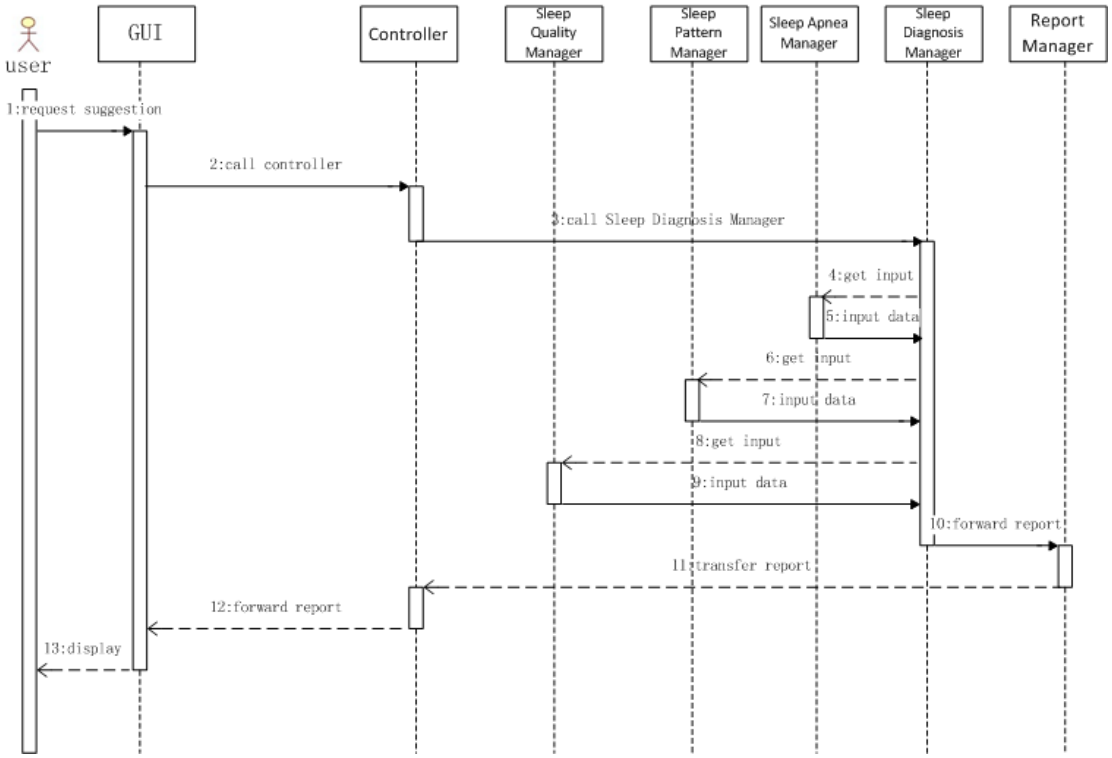


Responsibilities Associated:

1. GetSleepPattern allows the users to know their sleep pattern in terms of hours of REM or non REM sleep.
2. The GUI is responsible for building the interaction between users and system. If illegal input exists, the GUI will rise a error, then ask for valid input.
3. The controller is responsible for calling the functional unit to initiate the pattern getting process, and it also check the data status after the user has chosen a valid date.

4. The Sleep Pattern Manager is the key functional unit in this process. It counts the heart rate data, does necessary analysis, and then gives out result for displaying.

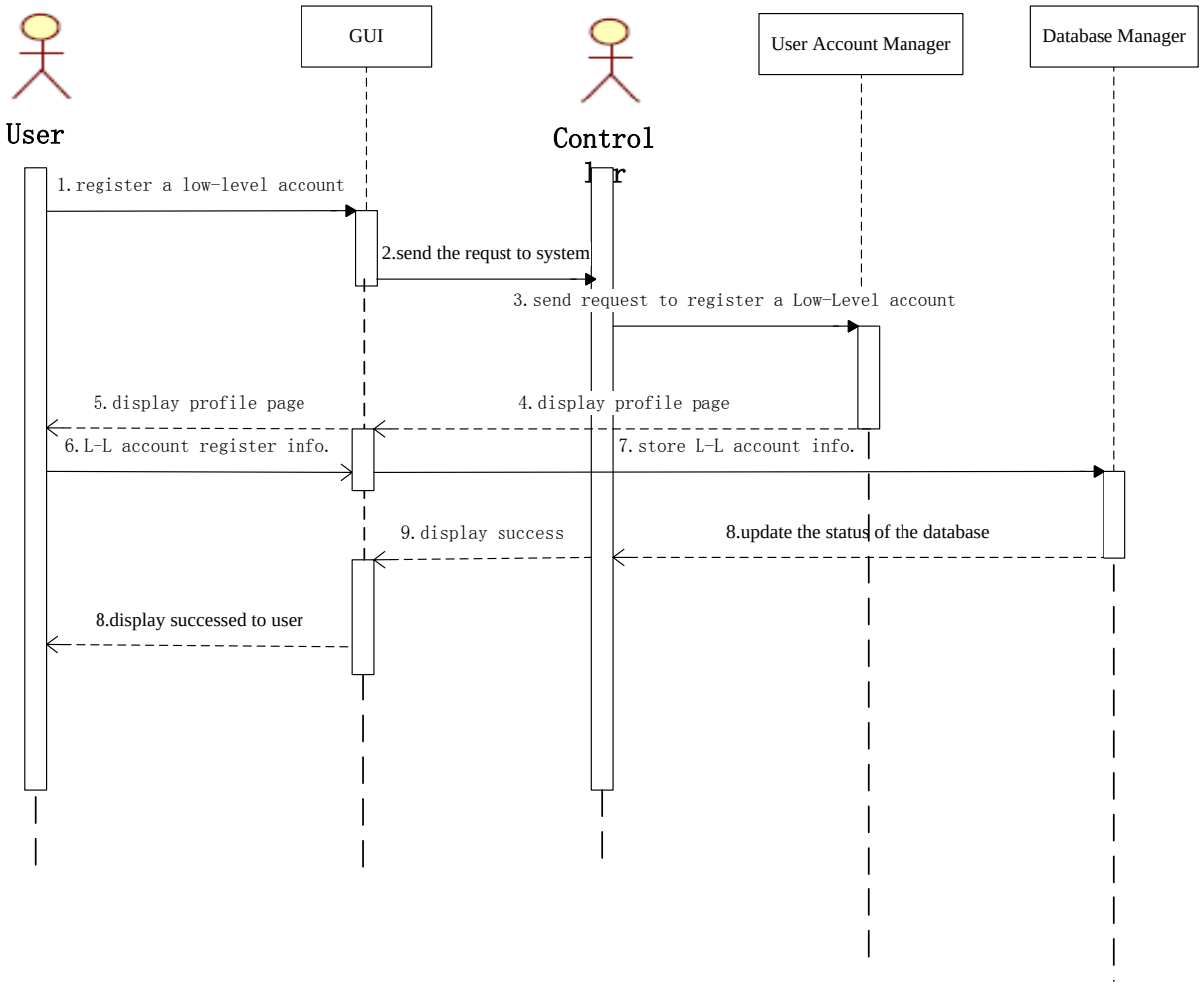
UC-5: GetSuggestion



Responsibilities Associated:

1. GetSuggestion allows all the users to receive specific suggestions concerning sleep quality. In this process, all the results from Sleep Apnea Manager, Sleep Pattern Manager and Sleep Quality Manager will be used to induce useful suggestions.
2. The Sleep Diagnosis Manager plays the key role in this function. It accepts data from the other three functional units, does necessary analysis, rises suggestions and finally generate the prototype of the report.

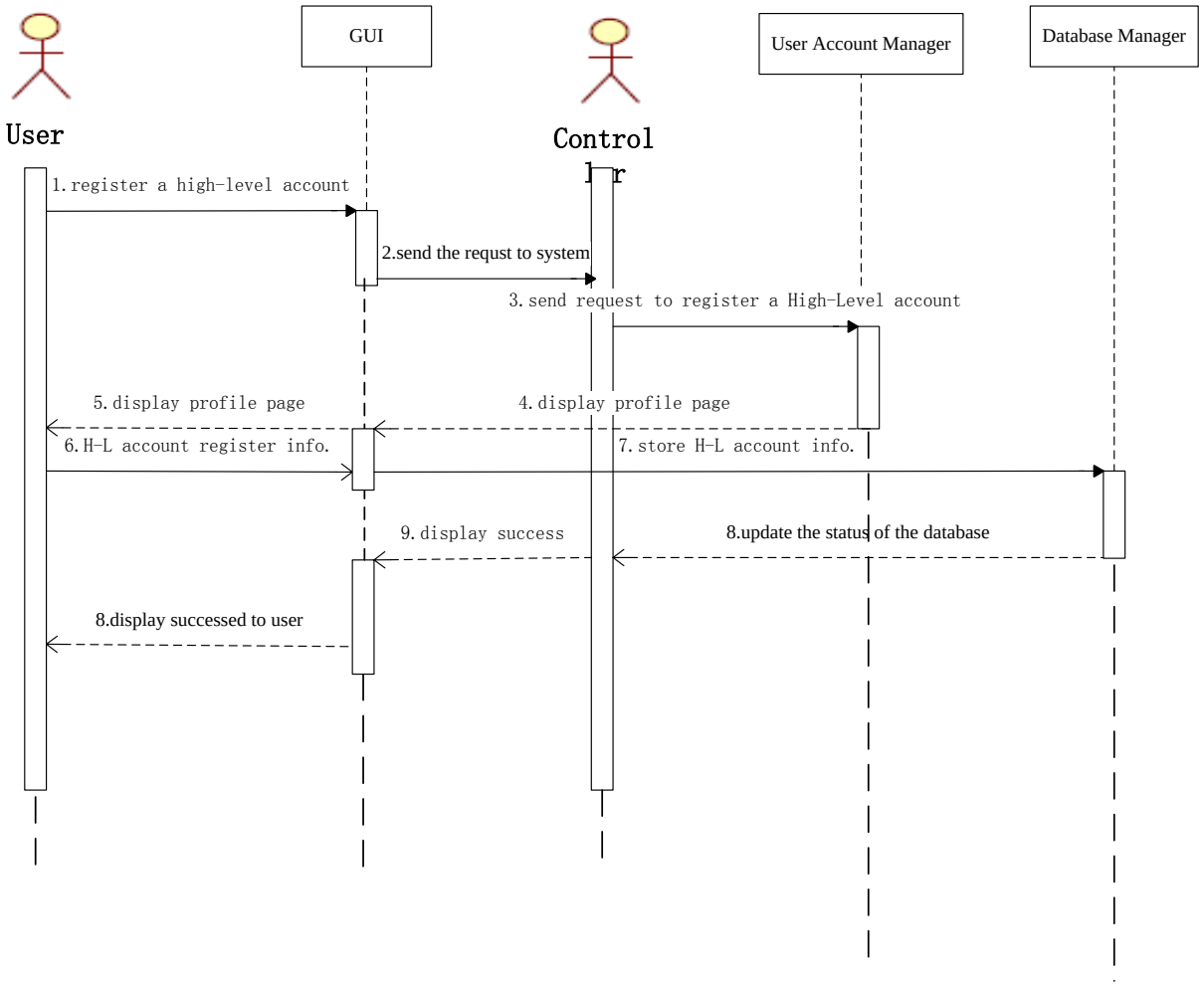
UC-6: LowLevelRegister



Responsibilities Associated:

1) GUI aims to provide the user with a clear and concise way of uploading sleep raw data and viewing analysis results and suggestions.
2) Controller is responsible for all the control process of receive request, call functions and transferring report, etc.
3) User Account Manager is mainly working on provide low-level user account permission here.
4) Database Manager is for storing data into the database.

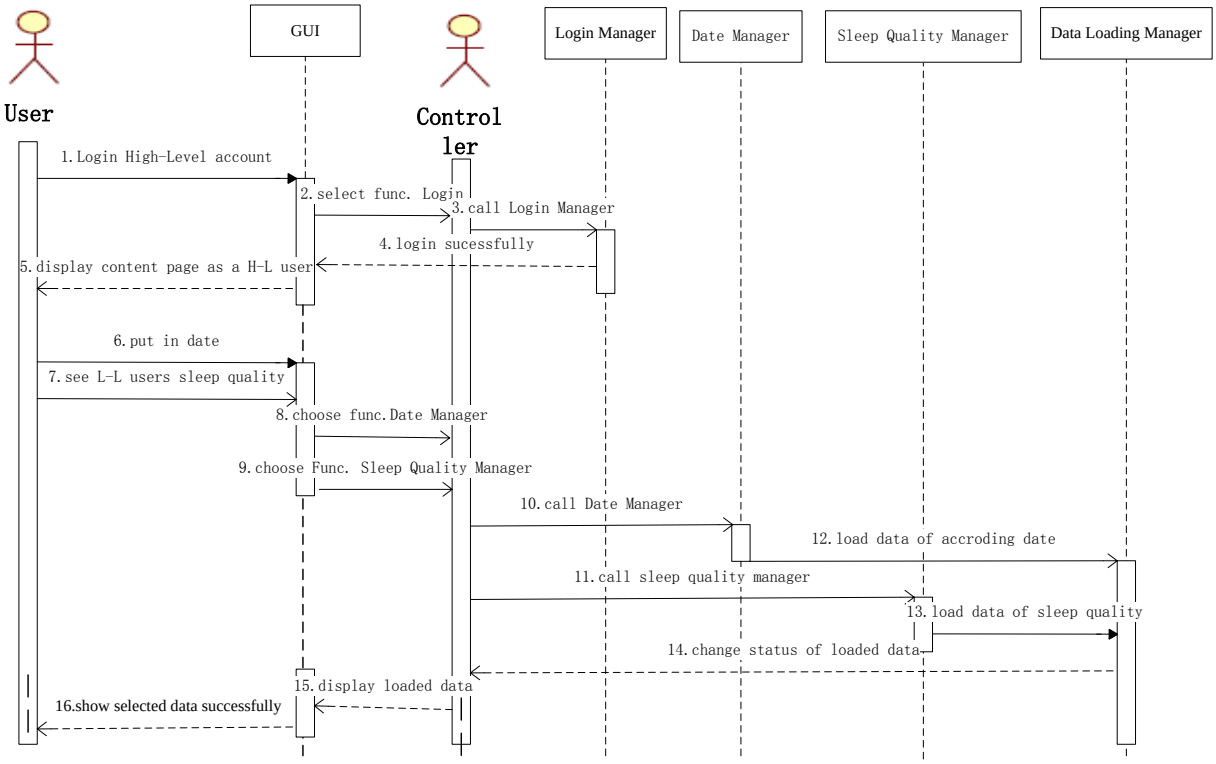
UC-7: HighLevelRegister



Responsibilities Associated:

1) GUI aims to provide the user with a clear and concise way of uploading sleep raw data and viewing analysis results and suggestions.
2) Controller is responsible for all the control process of receive request, call functions and transferring report, etc.
3) User Account Manager is mainly working to provide low-level user account permission here.
4) Database Manager is for storing data into the database.

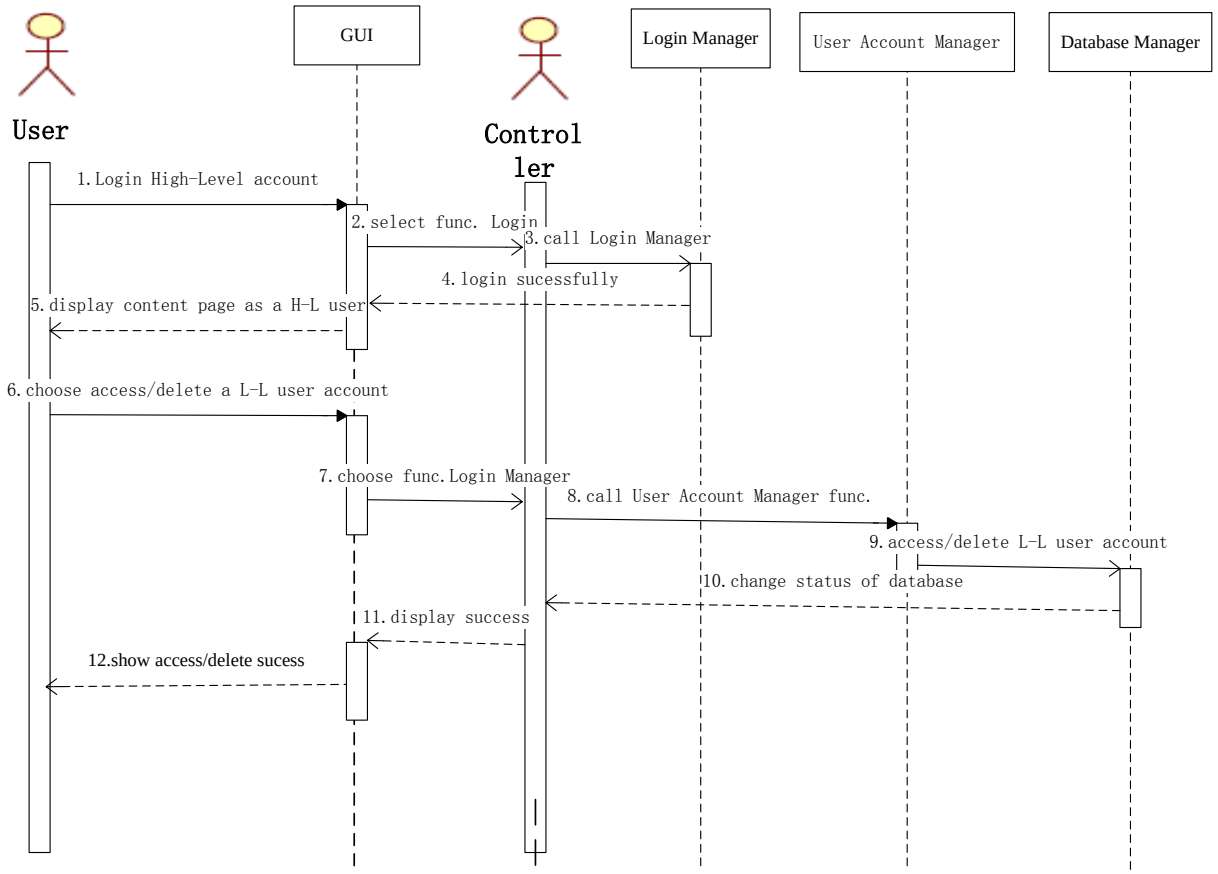
UC-8: Supervise



Responsibilities Associated:

1) Login Manager is to allow users to login in different levels, in this case the user login as a high-level user.
2) Date Manager set request date for data.
3) Sleep Quality Manager calculates the score of sleep and assess quality of sleep.
4) Data Loading Manager Loads Monitoring Data from Database and get Sleep Pattern analysis result

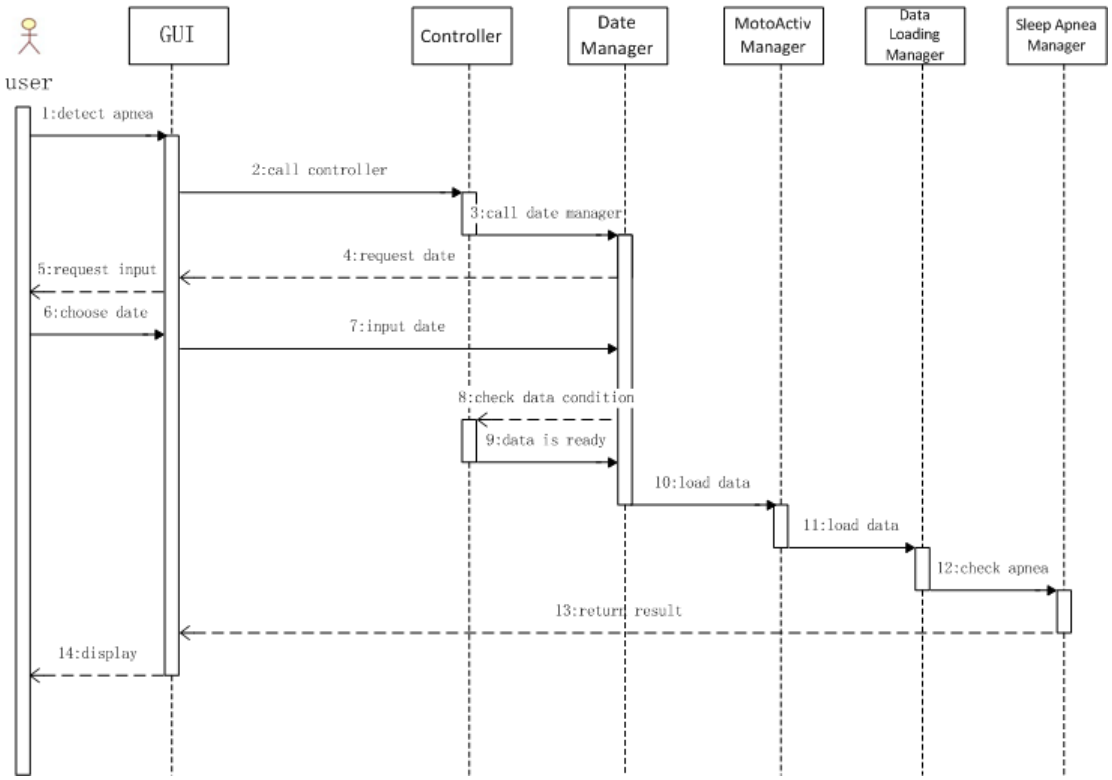
UC-9&10: Access/DeleteUserAccount



Responsibilities Associated:

1) Login Manager is to allow users to login in different levels, in this case the user login as a high-level user.
2) User Account Manager is mainly working to provide low-level user account permission here.
3) Database Manager is for changing data into the database, including deleting data in it.

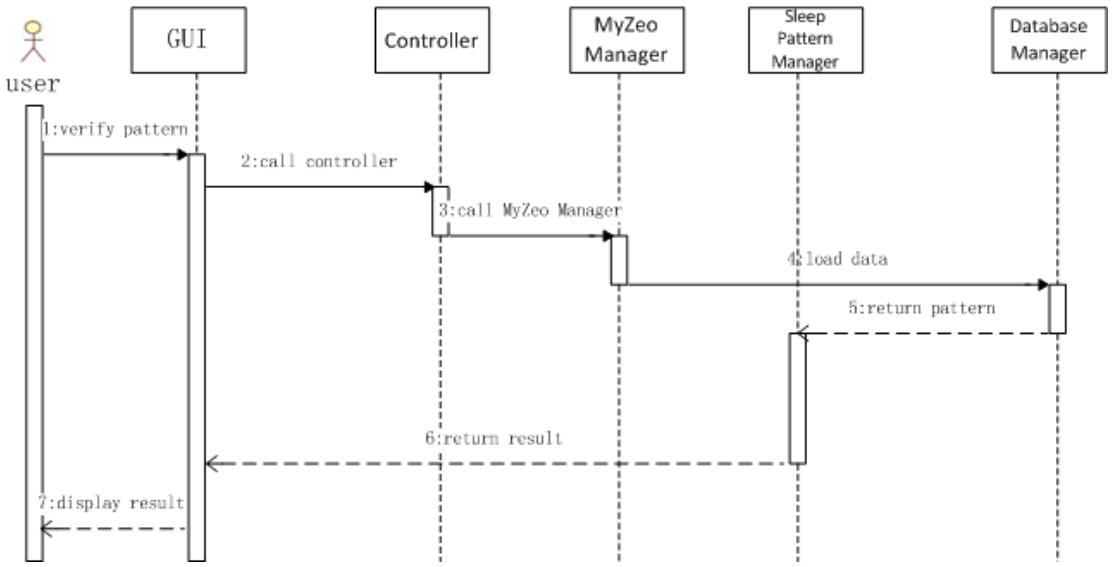
UC-11: DetectSleepApnea



Responsibilities Associated:

1. DetectSleepApnea is aimed at finding if user suffers from sleep apnea. In this process, the user again needs to select a valid date for checking.
2. The Sleep Apnea Manager is the key concept in this process. It does the complex analysis job, which combining the data from MIT-ECG database to find the sleep disorder, using the popular SVM method.

UC-12: VerifyPatterns

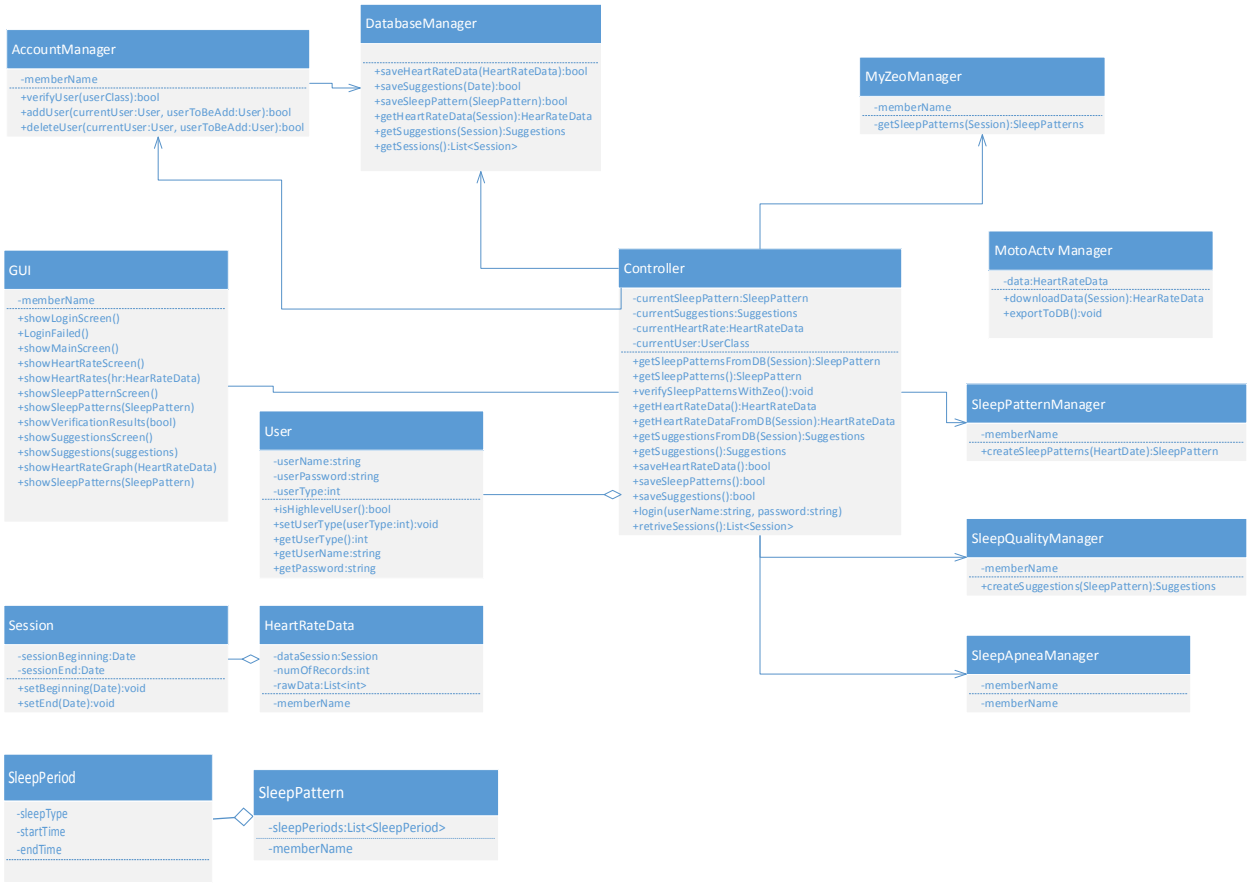


Responsibilities Associated:

1. VerifyPatterns is a subsidiary process in our system. It compares the user's sleep patterns deduced from the GetSleepPattern process with the outcomes of MyZeo, which is one way the system justifies for itself.
2. The Sleep Pattern Manager here does the comparison job, which involves some computations work.

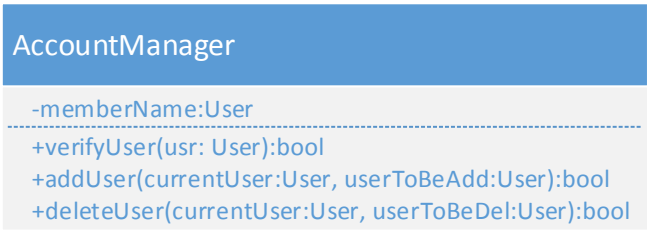
2. CLASS DIAGRAM AND INTERFACE SPECIFICATION

2.1 Class Diagrams



2.2 Data Types and Operation Signatures

Account Manager



Attributes: Member Name of type User

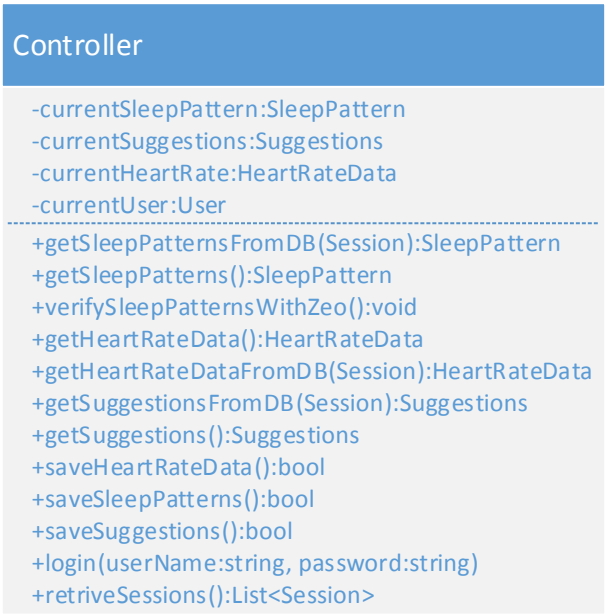
Related Concepts: Controller, DataBase Manager

Operations:

+verifyUser, +addUser, +deleteUser

- Used within Login operation of Controller
- In the controller, Login Data from GUI is put in the form of User Data Type and compared with password and user name in database by Account Manager's operation +verifyUser
- If the result is true then the input user is made the current user (+addUser), else the flow of control is stopped in Account manager and given back to the Controller.
- Upon, successful login, then the Main Screen function is presented to the User
- When the User logs out, through the controller, the current user is deleted

Controller



Attributes: CurrentSleepPattern, current Suggestions, current Heartrate, currentUser

Associated Concepts: GUI, User, Account Manager, Database Manager, Controller, My Zeo Manager, Sleep Pattern Manager, Sleep Quality Manage

System Operations

+getSleepPatternsFromDB(Session):SleepPattern

- Obtain Sleep Pattern for the particular session from the database

+getSleepPatterns():SleepPattern

- Get Sleep pattern for current heart rate data from Sleep Pattern Manager for the current heart rate data

+verifySleepPatternsWithZeo():void

- Compare Sleep Pattern with Zeo's sleep Pattern to Obtain accurate results

+getHeartRateData():HeartRateData

- Get Heart rate data from MOTOACTV website

+getHeartRateDataFromDB(Session):HeartRateData

- Get Heart rate data for the corresponding session from the database.

+getSuggestionsFromDB(Session):Suggestions

- Get the sleep suggestions from the database for the corresponding session

+getSuggestions():Suggestions

- Get suggestions from the database for the current user

+saveHeartRateData():bool

- Save Heart rate data in the database and if successful, return true, else false

+saveSleepPatterns():bool

- Save Sleep pattern in the database and if successful, return true, else false

+saveSuggestions():bool

- Save Suggestions in the database and if successful, return true, else false

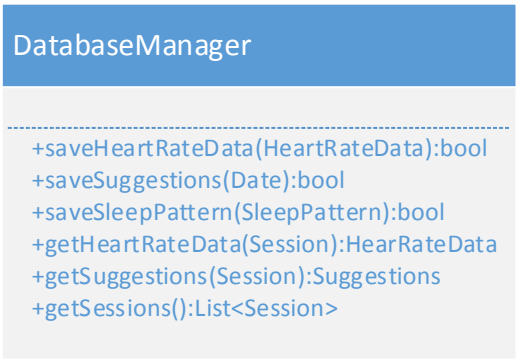
+login(userName:string, password:string)

- pass the username and password to the Account Manager and proceed

+retriveSessions():List<Session>

- Get all the sessions stored on the database

Database Manager



Attributes: Heart Rate Data, Date, Sleep Pattern

Association: Controller, Account Manager

System Operations:

+saveHeartRateData(HeartRateData):bool

- To check if heart rate is saved correctly in the database

+saveSuggestions(Date):bool

- Save suggestions to the database based on the date and return true if successful

+saveSleepPattern(SleepPattern):bool

- Save Sleep Pattern to the database

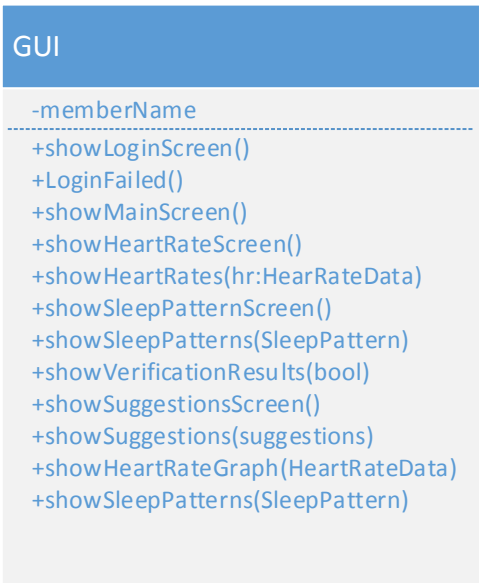
+getHeartRateData(Session):HearRateData

- Get Heart Rate data for a session from the database

+getSuggestions(Session):Suggestions

- Get suggestions corresponding to a session from the database

GUI



Associations: Controller

Attributes: MemberName

System Operations

+showLoginScreen()

- First Page, Lets the user authenticate themselves by supplying username and password in association with the Controller

+LoginFailed()

- If The Controller sends a message saying the login falied, informs the user

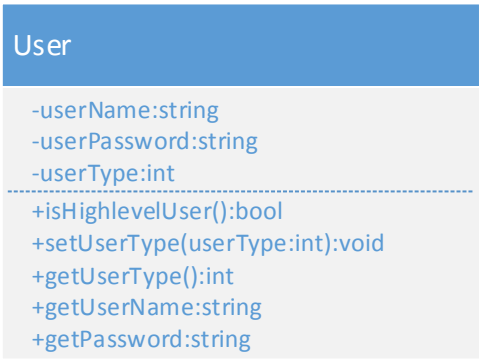
+showMainScreen()

- Upon successful login, the Main Screen is presented, for the user to get Heart Rate Data, To get sleep suggestions etc

+showHeartRateScreen()

- The

User



Attributes: Username, User Password, User Type

Associations: Controller

System Operations

+isHighlevelUser():bool

- Checks if the User is a high level user, if so , return True

+setUserType(userType:int):void

- Sets the User Type, if High Level or Low Level

+getUserType():int

- Obtain the User Type

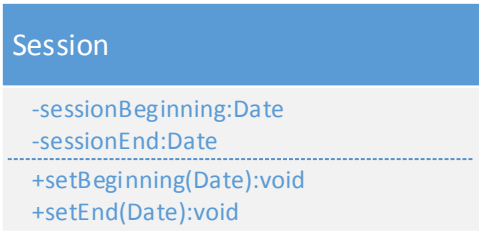
+getUserName:string

- Get the User name

+getPassword:string

- Get the password corresponding to the user

Session



Attributes: Date

Associations: GUI, Database Manager

System Operations:

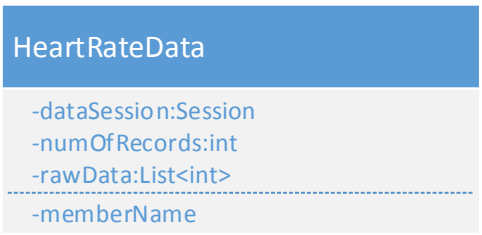
+setBeginning(Date):void

- Set the beginning date of the session

+setEnd(Date):void

- Set the end date of the session

Heart Rate Data



Attributes: Session, Number of Recirds, rawData

Association: Everything!

System Operation:
Forms the base Data Structure for this whole project

2.3 Traceability Matrix

Classes / Domain Concepts	Account Manager	Controller	DataBase Manager	GUI	User	Heart rate Data	Session	Sleep Pattern Manager	Sleep Apnea Manager	My Zeo Manager	Sleep Quality Manager	Motoactv Manager
MotoActv Manager												x
MyZeo Manager										X		
Database Manager			X									
Data Loading		x	x	x	x	x	x	x			x	X
Sleep Pattern			x	x	x	x		x				
Sleep Quality	x	x	x	x	x	x	x	x	x		x	x
Sleep Apnea									x			

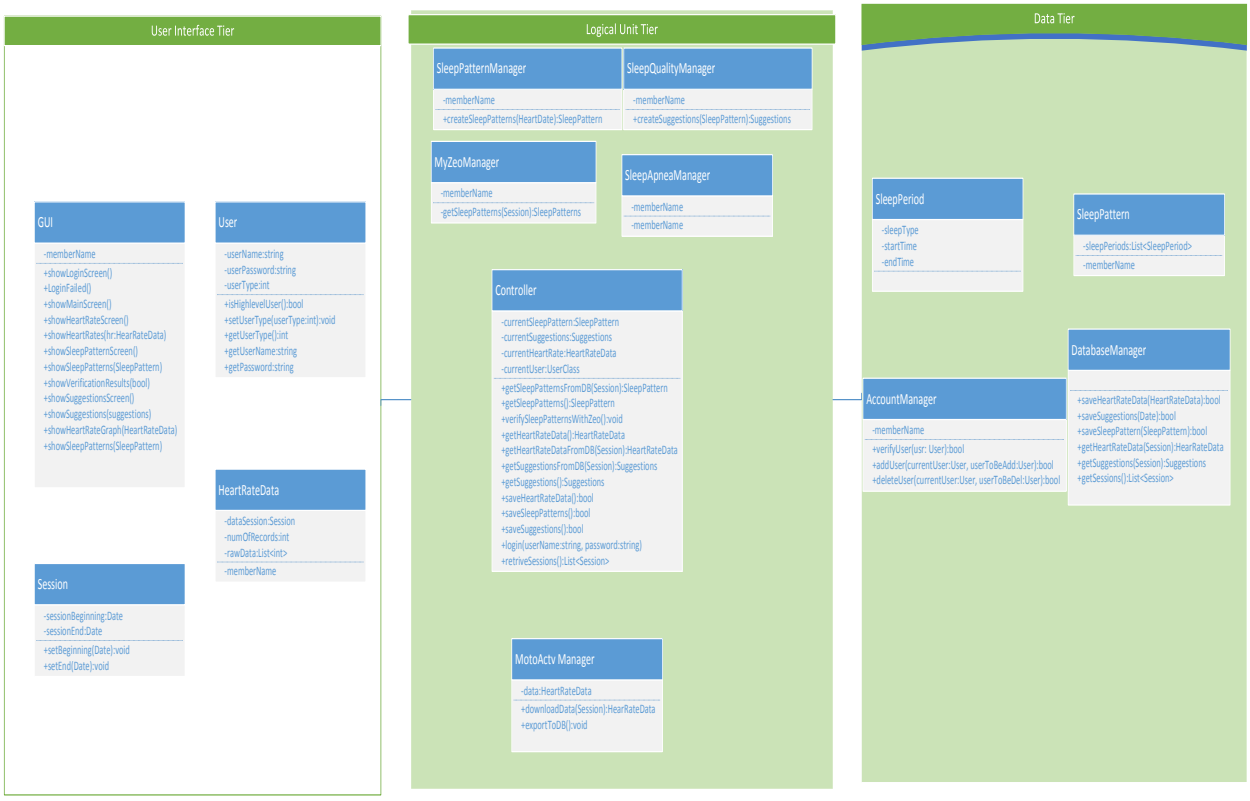
GUI	Controller	User Account	Sleep Diagnosis	Report Manager	Data Logging	Date Manager	Login Manager	classe
×		×		×		×	×	Account Manager
	×	×		×			×	Controller
		×		×			×	DataBase Manager
×		×		×			×	GUI
		×				×	×	User
				×	×			Heart rate Data
						×		Session
			×	×				Sleep Pattern Manager
			×	×				Sleep Apnea Manager
			×					My Zeo Manager
			×	×				Sleep Quality Manager
								Motoactv Manager

3. SYSTEM ARCHITETURE AND SYSTEM DESIGN

3.1. Architectural Styles

Our system adopts 3-tier architecture in which the user interface, functional process logic ("business rules"), data storage and data access are developed and maintained as independent modules. The Presentation layer deals with the user interface which helps the user to interact the system in a smooth manner providing the ability to understand the results. The logic tier is the heart of our system. In the tier, the raw data uploaded by users is being processed by various algorithms and transferred to understandable results. In the end, the data tier stores all the user data and also the logic tier retrieves data from this tier for processing.

3.2 Identifying Subsystems



3.3. Mapping Subsystems to Hardware

Our system does not run on multiple computers. As we scale the system, we forsee this requirement, but as of now we shall not deal with this scenario

3.4. Persistent Data Storage

Our system must store all data that the system may use. This includes all user profile information, personal heart rate data. Our system accomplishes this by using relational database.

3.5. Network Protocol

Since our system runs on a single machine, network protocol is not applicable here.

3.6. Global control flow

The system is event driven. The user interacts with the system by first registering his/her user name and setting the password. Once the user log in and the next step is to download his/her personal heart rate data from website into database. Then the system waits for events, which could be clicking “analyze” button or selecting sleep disease diagnose function or just changing the time period. In addition, our system is not a real-time system and does not use multiple threads. Only one user can login into the system at a time.

3.7. Hardware Requirement

Our system requires the use of a MOTO Heart Rate monitor and Myzeo sleep monitor. In addition, the PC must have an internet connection to be able to download heart rate data from Motoactv website.

4 ALGORITHMS AND DATA STRUCTURES

4.1 Algorithms for User Management

4.1.1 Verify user

Required for login screen and high level user operations in account management GUI
Expected return: user Record from database with id, name, password and type fields
Possible errors: User (password_error, user_not_found)

Step 1) Create user object with name and password fields are filled
Step 2) Retrieve userRecord with user.name from database
Step 3) Does userRecord exist? If no return user_NotFound
Step 4)Does userRecord.password = user.password? if no return User_password_error
Step 5) if yes, return user object

4.1.2 Getting all account names

Required for populating deleteUser combo box in account management GUI
Expected return: All user names from database
Possible errors : AuthenticatingUser (password_error, permission_error, user couldn't be found)

Step 1) create authenticatingUser object with name and password fields
Step 2) verify authenticatingUser. If any error from verifyUser, return same error
Step 3) if authenticatingUser is low level user return Authenticating User_permission_error
Step 4) else get all users from database and return

4.1.3 Adding user

Called from account management GUI
Expected return: success
Possible errors : AuthenticatingUser (password_error, permission_error, user couldn't be found)

Step 1) create AuthenticatingUser and newUser objects with userNames and passwords
Step 2) verify AuthenticatingUser, if there is an error with verifying return same error
Step 3) if AuthenticatingUser is a low level user return AuthenticatingUser_permission_error
Step 4) otherwise user is allowed to add account, add newUser to database return success

4.1.4 Deleting user

Called from account manager GUI
Expected return: success
Possible errors : AuthenticatingUser (password error, permission error, user couldn't be found), deleteUser couldn't be found

Step 1) create AuthenticatingUser object with userNames and passwords. Also create deleteUser with only username
Step 2) verify AuthenticatingUser, if there is an error with verifying return corresponding error

Step 3) if AuthenticatingUser is a low level user return
AuthenticatingUser_permission_error
Step 4)else retrieve userRecord from database from deleteUser.name
Step 5) if deleteUser doesn't exist return deleteUser_notFound
Step 6) if user exist delete deleteUser by id from database return success

4.2 Algorithms for Sleep Pattern Assessment

Heart rate is a good indicator of sleep quality. There exists a clear relationship between the heart rate while asleep VS. awake. Dr. K. Krrauchi, in a study reported in "Neuropsychopharmacology" (2001) detected an average drop from 64 to 52 beats per minute from lights off till you reach light, continuous sleep[2]. The low frequency power as well as the high frequency power was lower when the subjects were asleep. There is also a relationship [4] between the variation of heart rate and specific sleep stages. Basically, the larger variation usually goes with the REM sleep. Difference of heart rate variation between REM sleep and Non-REM sleep may be used to distinguish the sleep stages. On the other hand, in the frequency domain, one study [5] has showed that compared to Non-REM sleep, low frequency band power has decreased and low frequency (LF) to (HF) ratio has significantly increased during REM sleep.

Up to now, we go with time domain analysis to distinguish sleep stages.

4.2.1 Algorithm for distinguishing awake and sleep stages:

Step 1)Calculate the average heart rate of every 5 minutes, HR_i

Step 2)If $HR_i > HR_{i+1} > HR_{i+2}$ and $avg(HR_i + HR_{i+1} + HR_{i+2}) < \theta$, go to step 4

Step 3) $i = i + 1$, go back to step 2

Step 4) t_{i+2} is the time when user goes to asleep stage

Where HR_i is the average heart rate in the ith 5 minutes; θ represents the threshold of heart rate based on experiments.

4.2.2 Algorithm for distinguishing REM stage and NREM stage:

Step 1) Calculate the average heart rate of every 5 minutes, HR_i

Step 2) Calculate the standard variance of heart rate in every 5 minutes, Var_i

Step 3) If $avg(Var_i + Var_{i+1}) > \theta_{\text{var}}$, go to step 5

Step 4) T_i is NREM stage, $i = i + 1$, go back to step 3

Step 5) T_i and T_{i+1} are classified as REM stage, $i = i + 1$, go back to step 3

Where Var_i is the standard variance of heart rate in the ith 5 minutes, and θ_{var} is the threshold, which we estimate it as the three-quarters of heart rate range. T_i means the ith 5 minutes.

4.2.3 Algorithm for computing the sleep quality score:

Here we use the following formula to calculate the sleep quality score:

$$w_{awake}(|t_{awake} - S_{awake}| / S_{awake}) + w_{rem}(|t_{rem} - S_{rem}| / S_{rem}) + w_{nrem}(|t_{nrem} - S_{nrem}| / S_{nrem})$$

Where w_{awake}, w_{rem}, and w_{nrem} are the weights for each stage; S_{awake}, S_{rem} and S_{rem} are the standard amount of time for each stage of a normal adult person during one night sleep. And t is the amount of time that we calculate for each stage.

4.3 Data Structure

4.3.1 Data classes for account management

User

Id (int 11) : unique user id
Name (varchar(45)) : unique user name
Password(varchar(45)) : user's password
Type : user's level :
0 – High level user
1 – low level user

4.3.1 Data structure for SleepPeriod

In order to conveniently analyze sleep pattern, we construct a specific data structure called SleepPeriod, which is organized as follows:

SleepPeriod

sleepType : sleep stages :
0 – awake
1 – NREM
1 – REM

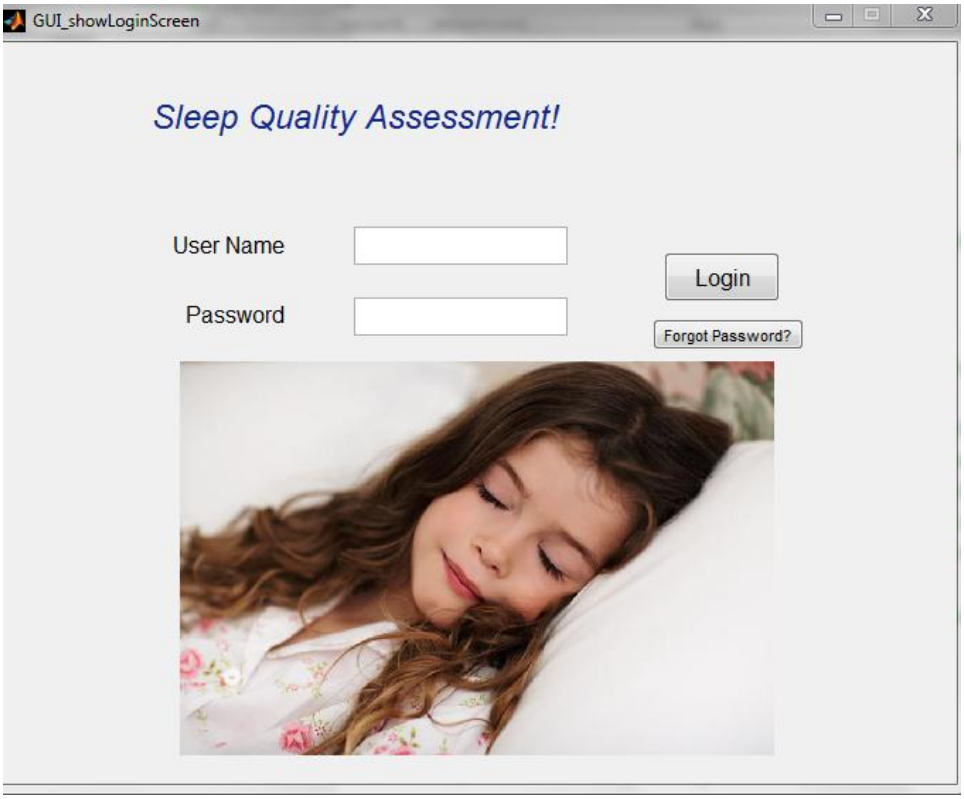
startTime : the start time of current sleep period
endTime : the end time of the current sleep period

Then we used a linked list to store them for further use and easy access.

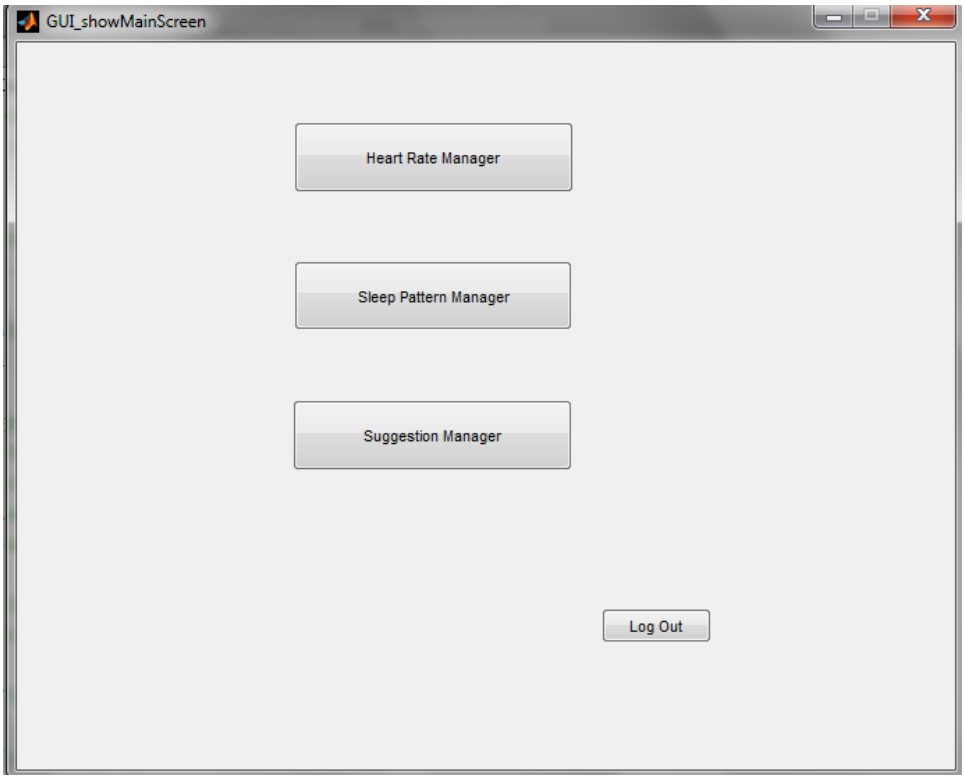
5 USER INTERFACE: DESIGN AND IMPLEMENTATION

The main aim of user interface design was to keep the interface as simple as possible. We wanted the user to have as hassle free an experience as he/she could. In order to achieve this, we made the following implementable on the GUIDE toolbox of MATLAB and using JAVA:

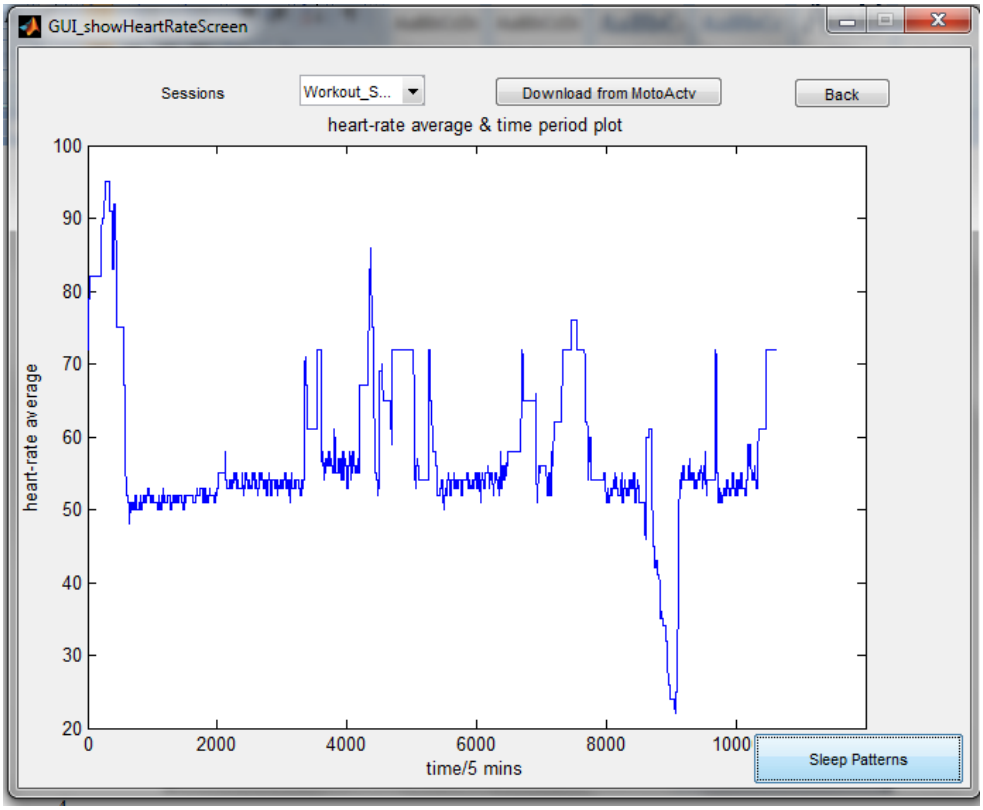
1. Login Page:
    - User enters username and password
    - Presses Login



2. Main Screen Page
    - User has options of Heart Rate Manager, Sleep Pattern Manager, Suggestions and if high level user that of the account manager



3. Heart Rate Manager
    - On choosing the Heart Rate Manager Option, the user is prompted to a screen, where if the data has not yet been downloaded, he presses the Download button. Here a java script integrated with MATLAB directly downloads the new user data from MOTOACTV saving the user from task of going by himself to the website, logging in, searching for the download data tab and choosing the session.
    - Once the user chooses the session of interest, the heart rate data is displayed and if the user wants to know the sleep pattern, rather than going back to the main page, the user has been provided with a Sleep Patterns button in this page itself. This is another salient feature of our user interface design, where we have tried our best to minimise user effort.
    - There is a Back button for the user to get back to the Main Page
    - The below figures illustrate the various features of the Heart Rate Screen
    - The Sleep Suggestions button generates two closable windows displaying the sleep pattern classification which are separate from the user interface thus separating the results from the GUI and hence easing the activity of the user


GUI_showHeartRateScreen
Sessions
Workout_S...
Download from MotoActv
Back
heart-rate average & time period plot
heart-rate average
time/5 mins
Sleep Patterns

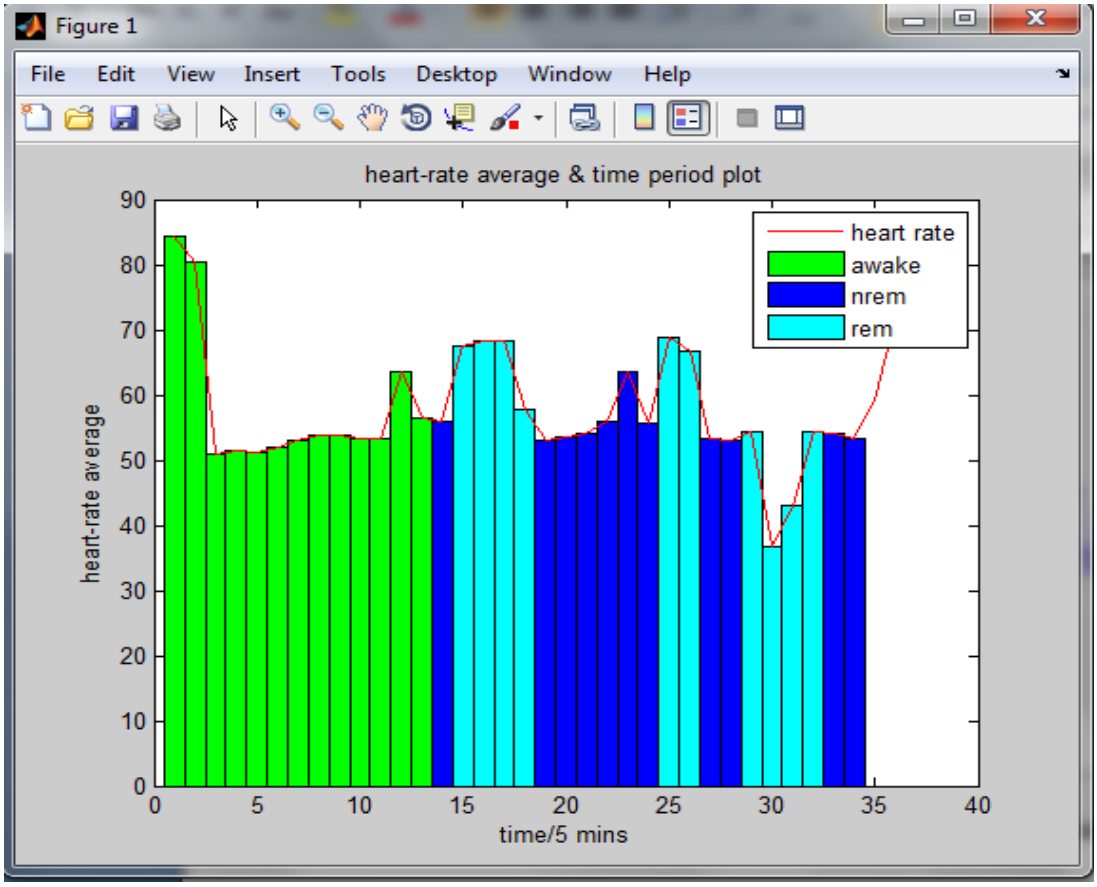
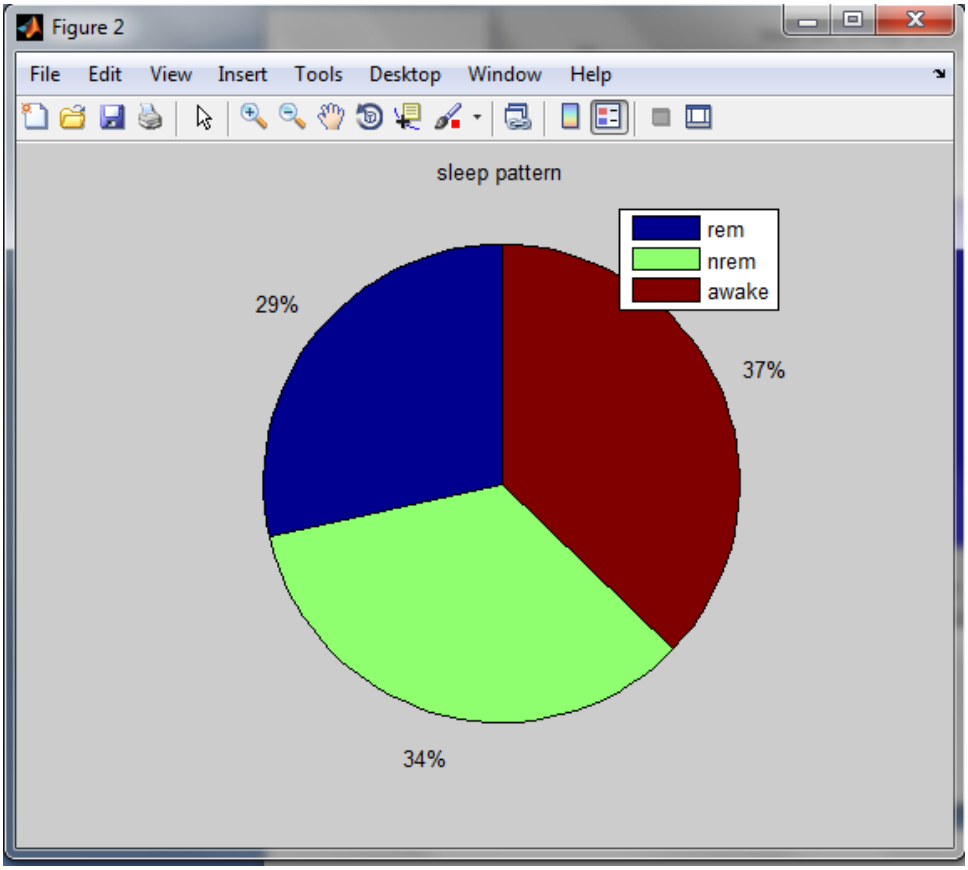


Figure 1
File Edit View Insert Tools Desktop Window Help
heart-rate average & time period plot
heart rate
awake
nrem
rem
heart-rate average
time/5 mins



4. The Sleep Pattern Manager works similarly.
5. The Log out option brings the user back to the login page.

6 DESIGN OF TEST

The following are the necessary test cases for ensuring the correctness of the whole system.

1. Test case id: Account Registration

Unit to test: Account Manager, database.

Assumptions: The program has displayed the registration screen and is waiting for the user's input.

Steps to be executed:

a. Input the preferred account number and password.
b. Input the basic personal information.
c. Click "finish" button to complete registration.
d. Expected result: A new user account has been added to the database.

Pass/Fail: A new user account shows up in the database / No account has been added to the database.

2. Test case id: login/log out

Unit to test: Account Manager

Assumptions: The program has displayed the log in page and is waiting for the user's input.

Steps to be executed:

a. Input a valid user account and the corresponding password.
b. Click "login" button.
c. If step 2 successes, Click "log out" button.

Expected result: The user successfully logs in and logout.

Pass/Fail: The user log in the system, and then logout / the user fails to log in.

3. Test case id: show heart rate

Unit to test: GUI, database

Assumptions: the user has logged in the system and clicked the "Heart Rate Manager" button on the main screen.

Steps to be executed:

a. The user clicks the "Download from MotoActv".
b. The user chooses one specific session.

Expected result: the heart rate data is displayed in the x-y coordinates as a function of time.

Pass/Fail: the graph is successfully displayed / the graph does not show up.

4. Test case id: show sleep pattern

Unit to test: GUI, database, sleePatternManager

Assumptions: the user has logged in the system and clicked the "Sleep Pattern Manager" button on the main screen.

Steps to be executed:

a. The user chooses one specific session.
b. Click "show sleep pattern" button.

Expected result: the sleep pattern has been shown in terms of REM and NON-REM sleep hours.
Pass/Fail: the graph is successfully plotted / the graph fails to show up.

5. Test case id: show suggestions

Unit to test: GUI, database, sleepQualityManager
Assumptions: the user has logged in the system and clicked the “Suggestion Manager” button on the main screen.
Steps to be executed:

a. The user chooses one specific session.
b. Click “show suggestion” button.

Expected result: the suggestions are displayed in the form of text.
Pass/Fail: suggestions are generated and displayed. / no suggestion comes up.

6. Test case id: add user/delete user

Unit to test: database, Account Manager
Assumptions: there are valid user accounts in the database.
Steps to be executed:

a. Choose one user account in the database.
b. Click “delete user” button.
c. Input a new user account and password.
d. Click “add user” button.

Expected result: the former user account is removed from database and the new account exists in the database.
Pass/Fail: The former user account is successfully removed from database as well as new account being added. / Former account still exists in the database or new account does not show up.

The integration flow test we carried out is described below.

- Login with unknown user account

-> Fail

- Login with high-level user account with wrong password

-> Fail

- Login with high-level user account with correct password

-> Success

- Add high-level user

-> Success

- Logout

-> must show login screen

- Login with new high-level user

-> success, must show main screen

- Logout

-> must show login screen

- Login with high-level user

-> must show login screen

- Remove new high-level user

-> success

- Add low-level user

-> success

- Logout

-> must show login screen

- Login with new low-level user

-> must show main screen

- Run Heart-rate Manager

-> must show an empty screen

- Check sessions

-> There must be an empty list

- Click download from motoactv

-> session list must be changed

- check sessions

-> There must be a full list

- Click one from the session list

-> Heart rate must be shown

- Click sleep patterns

-> must show sleep patterns

- Logout

-> must show login screen

- Login with high-level user

-> must show main screen

- Remove new low-level user

-> success

- Logout

-> must show login screen

7 PROJECT MANAGEMENT AND PLAN OF WORK

7.1. Merging the Contributions from Individual Team Members

We compile all team members' work before the demo 1 and integrate them into the demo. Next are the issues we encountered during our work.

The first problem we encountered was that we have to study the medical knowledge about sleeping, because we have no the background of medical. We have two members mainly take charge the part of studying the knowledge of sleeping patterns. They read a lot of paper and simulate the algorithms in the computer. When they met some problems, we would discuss together and provided some ideas, though we are not professional. When they finished the part of detecting Rem and Non-Rem of sleep, they show their achievement to us and taught us the medical knowledge detailedly, to let me understand the basic information about sleep pattern.

The second issue we encountered was how to organize our meeting to discuss our project, and how to combine our own separate part of work, like the documentation. For the meeting, we first used the google hangouts to hold a meeting. But we found it was not good for discussion because of the limitation of the network. So we decided that we will communicate via email, and we would hold a face-to-face meeting ever week. For the documentations and the code, we use dropbox to share our jobs, and we think it is really a good method.

The third issue was we will have different opinions for the project. Like the report 1, we still discussed the Use Case and requirements before we summit the report. Hence we make a little problem in the report 1. But we still revise the report 1 even it was summated, to make it better. And we sent the revised version to the professor as well as the report 2 part 1. I think it is good if everyone have different opinions, it can help to improve our project. But how to express their own opinions is important for the team.

7.2. Project Coordination and Progress Report

We have finished UC-1, UC-3, UC-4, UC-6, UC-7, UC-9 completely. The major part in our next step is the SVM algorithm. We need to use the SVM to extract some features of the sleep disease, and compare with the data we collected, to finish the UC-12. Also, we need to work on the new interface to make it perfect in the demo 2. These are the mainly parts in our next step.

As a group, we often discuss our jobs and provide our opinions to each other's work via email and dropbox. We think this is a good implicit tool for everyone to improve their work.

7.3. Plan of Work

Till the date of Report 2, we have already finished four deliverables including the Proposal, Report 1, demo1 and Report 2. A brief of overview of these four deliverables are as follow:

- The Proposal was a brief overview of the group members working on this project, a

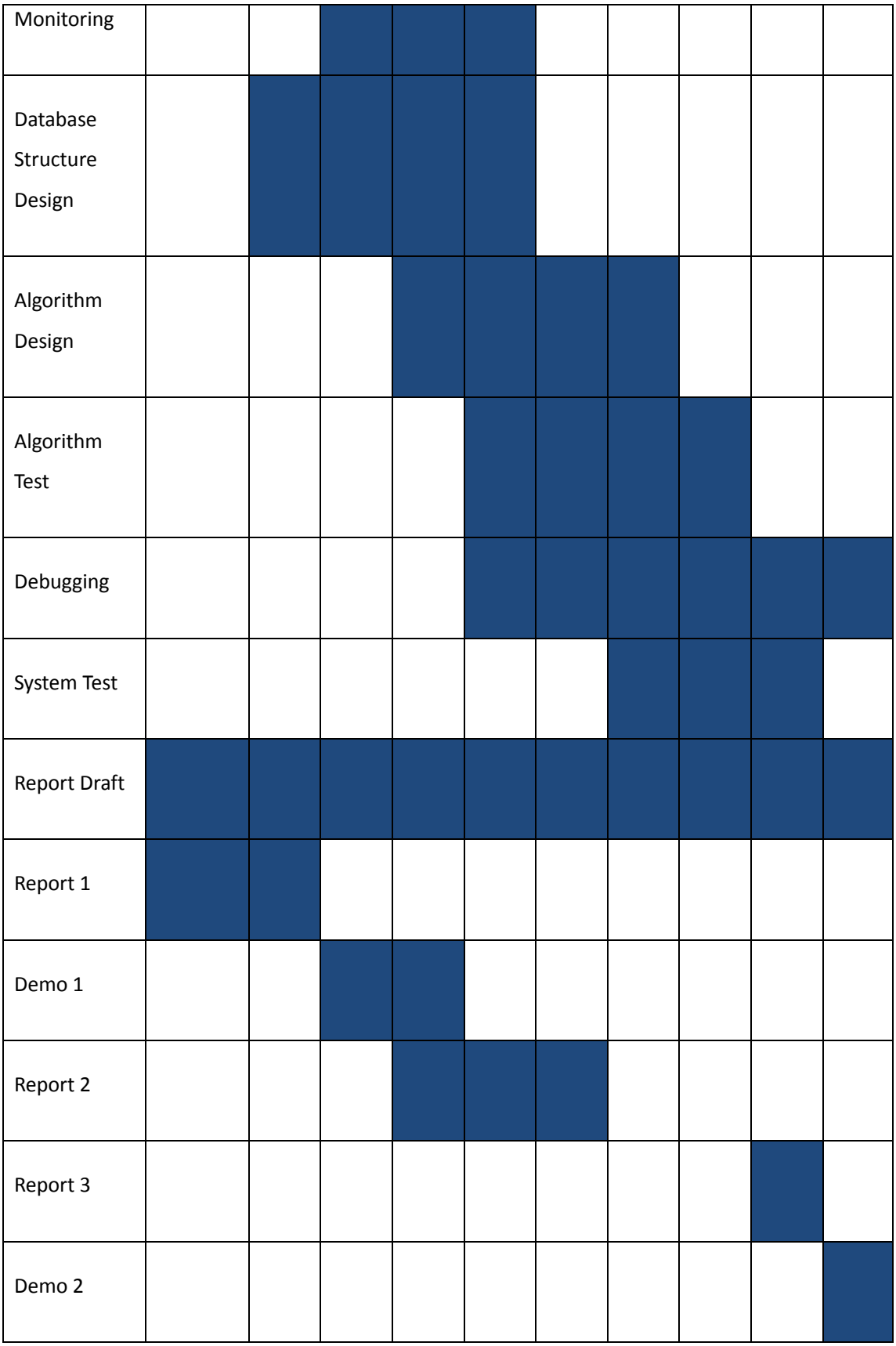
short description of the Sleep Quality Assessment, and solutions and strategy.

- Report 1 includes the Systems Specification document. It includes specific details of our project such as the Customer Statement of Requirements, enumerated Functional Requirements, Use Case descriptions, Use Case Diagrams, potential User Interface and the Domain Analysis of our problem.
- Report 2 encompasses our System Design. This report holds main design aspects such descriptions to how the classes and modules will interact with each other in the interaction diagrams and Class Diagram and Interface Specification. In addition it also holds our Design of Tests, and the Project Management and Plan of Work.
- Demo 1 shows the achievements that we have in so far. We combined our codes together, made a interface to show our work. In the demo, we also introduced the basic theory of our project.

And the follows are what we need to do in the rest of the semester:

- Get Sleep Quality:
  1) Distinguish deep and light sleep stages
  2) Compute sleep quality score
  3) Suggestion for improvement of sleep
- Sleep apnea models :
  1) Transfer the ECG to heart rate
  2) SVM algorithm
- Verify Patterns :
  1) Compare results with Myzeo
- Accounts Management:
  1) Supervise
  2) Delete accounts
- User Interface:
  1) Integrate the whole system

	41	42	43	44	45	46	47	48	49	50
Interface Design										
Interfacing Heart Rate										


Monitoring
Database Structure Design
Algorithm Design
Algorithm Test
Debugging
System Test
Report Draft
Report 1
Demo 1
Report 2
Report 3
Demo 2

Archive Documentation										

7.4. Breakdown of Responsibilities

Our team comprises of 6 members.
The following are the Responsibilities of our jobs for developed or currently working on:

Get sleep data	Jiajun
Sleep pattern and graph	Chenyun/Zeya
Sleep quality	Chenyun/Zeya
Interface	Cagdas/Maridula
Login and Registration	Cagdas
Sleep apnea models	Jiajun/Mridula/Yangkai
Get data from myzeo	Cagdas

We have **equally** contribution to this project.

8 REFERENCES

[1]. Test case. *Wikipedia.* March 3, 2012. http://en.wikipedia.org/wiki/Test_case
[2] Heart rate change from awake stage to sleep stage
URL: http://www.livestrong.com/article/105256-normal-heart-rate-sleeping/
[3] Ghosh P. Effect of Sleep on Heart Rate Variability[J].
[4] Masao Yaso, Atsuo Nuruki, Sei-ichi Tsujimura, and Kazutomo Yunokuchi, Detection of REM sleep by heart rate, Proceedings of The First International Workshop on Kansei
[5] Elsenbruch S, Harnish M J, Orr W C. Heart rate variability during waking and sleep in healthy males and females[J]. Sleep, 1999, 22(8): 1067-1071.
[6]. What Does Usability Mean:Looking Beyond 'Ease of Use'. *WQusability*. March 7, 2012. http://www.wqusability.com/articles/more-than-ease-of-use.html
[7]. How to write effective Test cases, procedures and definitions. March 7, 2012 http://www.softwaretestinghelp.com/how-to-write-effective-test-cases-test-cases-procedures-and-definitions/
[8]. Unified Modeling Language. WikiPedia. http://en.wikipedia.org/wiki/Unified_Modeling_Language#Interaction_diagrams
[9]. Software Engineering Book. Ivan Marsic. http://eceweb1.rutgers.edu/~marsic/books/SE/book-SE_marsic.pdf