LifeExtender+

Group #16
Report #3

github.com/SE2017/LifeExtenderPlus

Trirmadura J. Ariyawansa
John Eng
Daniel Huang
Chris Kim
Kevin Lee
Kyungsuk Lee

Submission Date: 4/23/2017

Individual Contributions Breakdown

Task	Trirmadura Ariyawansa	John Eng	Daniel Huang	Chris Kim	Kevin Lee	Kyungsuk Lee

Table of Contents

Summary of Changes

After further discussion, we found superior alternatives for the implementation of our application. Initially, we thought to write the code in C because of its general efficiency to map constructs to machine language. However, we coded in everything in Java because of its official basis for many components of the Android OS. While Java runs slower than C due to its overhead from the extra layer of automated memory management and compiling to an intermediate language (Java bytecode), it is preferred over C because of its “write once, run anywhere” object oriented design. Additionally, we decided to use Firebase database instead of SQLite because of the ease of connectability through Android Studios.

I. Customer Statement of Requirements

A. Problem Statement

College can become a very turbulent period in one's life for those who do not manage their time well, as it comes with many responsibilities, ranging from attending class, completing coursework outside of class, and leaving time for leisure and socializing. However, one aspect that many people leave out of their schedule in college is time set aside for physical activity and sleep: both of which are critically important to a person's health. With only twenty-four hours in a day, finding a balance that facilitates managing a healthy lifestyle, time for studying and coursework, and a social life is a challenge for many. Unsurprisingly, unorganized people spend more time in one area, taking time from another. Even if students scheduled time for physical activity, they often do not have the motivation to be active. Although it may be beneficial to put more time into studying or hanging out with friends, it is ideal not to manage your time to the extent where it would be at the cost of one's health.

Background

The importance of health and time management cannot be understated, especially for college students. It is understood that a person who puts time into maintaining their health, eating and sleeping well, as well as exercising, will be able to better avoid a plethora of conditions such as high cholesterol, diabetes, heart disease, stroke, and obesity, which itself comes with a host of health problems. That being said, exercise also benefits those who partake in it in ways other than the prevention of the aforementioned ailments. For instance, those who spend time exercising feel less stressed, less anxious, have better self-esteem, and also feel reduced tension and fatigue. Even more important for college students, the benefits of exercise, as mentioned above, are not only appealing in general, but can also aid students in doing better in the other areas of their lives. Having better self-esteem will help one's social life, and a reduction in tension, stress, and anxiety can make coursework feel that much easier. By these points alone, the importance of physical activity for young adults is very clear.

According to the Centers for Disease Control and Prevention, an adult needs at least “two hours and thirty minutes of moderate-intensity aerobic activity” and “two days a week of muscle-strengthening activities.” With seven days in a week, equating to 168 hours, two and a half hours equates to merely 1.49% of time in a week. That being said, finding time to fit the moderate-intensity aerobic activity into one’s schedule is not impossible, especially since it is not required for all of the two and a half hours to be done all at once. The time spent in between classes walking between classrooms fits into this category. As for muscle-strengthening activities, many if not all college campuses are equipped with gyms and facilities to meet that exact need. In the case where that is not true, many of these activities can be performed at home without equipment, such as push-ups, sit-ups, and yoga.

Even though most students understand the importance of physical activity and sleep, they do not have the motivation to take an initiative on bettering either facets. Motivation is defined as a natural “process that initiates, guides and maintains goal-oriented behaviors”. Motivation could be split into 3 components: Activation, Persistence and Intensity. Activation is the decision to initiate a goal. Persistence is the continued effort toward a goal. Lastly, Intensity is the concentration and vigor that goes into pursuing a goal. All 3 components are needed in order to succeed. There are 2 theories of motivation that explain why people consider a healthy lifestyle. The first one is the incentive theory. The incentive theory of motivation suggests that people are motivated to do certain task for external rewards like a trophy. The second theory is the expectancy theory. The expectancy theory of motivation suggests that people are motivated to perform certain task because they expected good outcomes from doing it in the future. This ideology can be used to slowly persuade people into shifting into a healthier lifestyle.

Taking into consideration all of these advantages of exercise and drawbacks with lack thereof, we reach the conclusion that it is imperative that not only college students, but all young adults in general should learn how to schedule physical activities into their weekly lives. It is crucial that this audience is educated about the benefits of time management and exercising to motivate themselves into improving all areas of their health. Our software aims to accomplish this exact function and much more.

Application

One of the main functions for our software is a scheduler/planner that will attempt to find a way to efficiently fit physical activity into the user's life based on their schedule of classes and work, while leaving enough time to study, sleep, and participate in other forms of leisure. The application will first request for the user to submit information about their course schedule and/or work hours and the program will use the researched information to attempt to fit the recommended hours for exercise and sleep, while over time, taking into consideration average time spent walking between, for instance, classes on campus. It is our hope that the application will grant some guidance into proper time management that would be useful for people who use the application.

Furthermore, this software will also serve to educate its users on the benefits of a healthy lifestyle, and motivate them to exercise as they use the application. It will make recommendations as to what the user can do or should try in terms of workouts or fitness plans, using information on the user's physical activity that the application records or that the user submits.

To help with the user's motivation, a tab for goals will be implemented to give the user a database or list in which they can keep track of their goals of any sort, be it in terms of physical activity or not. Keeping track of goals will help the user stay on target, especially if they can see their own progress and how close they are to achieving these goals they have set.

The application will be designed to automatically record statistics about the user's daily physical activity patterns. Using GPS integration, the application will attempt to roughly estimate the distance walked or ran by periodically checking the position of the user. It will acquire a distance estimate by summing the distances between each point, but exclude instances where the difference in distance between two measurements divided by time between the measurements would suggest a movement speed exceeding 15 kilometers per hour, as that may suggest that the user is in a motor vehicle. The application will attempt to estimate the time spent at a gym also by using the GPS. The application will have information on the locations of gyms or parks and will count the amount of time spent in that location or area. The application will remind them 10 minutes before their active period to be at the appropriate location. While the user is at the gym or a park, the application will recommend different

activities or workout patterns for the user to try, based on how much time they have, which may gradually increase in intensity with the time the user spends using the application. Through these functions, the application will have a general idea of the user's pattern of physical activity and fulfill both the incentive and expectancy theory of motivation.

As the user goes about his or her routine with the application, and as more information is collected, the application will return the collected statistics back to the user to show how his or her physical activity changed over time, as well as statistics on walking/running distance and time estimates. The application will also separately display the user's health index in order to help quantitatively visualize his or her progress. The health index will be colored according whether the index will drop, rise, or stay constant. For example, if a student has been going to his/her active periods but starts to miss them, the health index will be colored different hues of red/orange depending how likely the health index will drop. The health index will be green if, for example, the student increases his or her active period, showing that the health index can increase. The health index can be shared with other users in order to compare their individual physical health, which in turn increases motivation in the user.

II. Glossary of Terms

1. **Active Period:** A scheduled event that involves physical exercise in a certain location such as working out in the gym, running in the park,swimming in the local YMCA/gym, and etc...

2. **Active Location:** Areas set to automatically record high intensity activities

3. **Health Index:** A qualitative score that indicates the individual's commitment to physical activity. This score is based on collected information. It will be based on the estimated walking/running distances, time spent in active period, at the gym, etc…

4. **Low Intensity:** Level of physical activity for walking

5. **High Intensity:** Level of physical activity for running and exercising at gym

III. System Requirements

1. Enumerated Functional Requirements

ID	Priority Weight	Requirement
REQ - 1	**5**	**System should retrieve schedule from user input**
REQ - 2	**4**	**System should determine an ideal time for user to intake meals throughout the day**
REQ - 3	**4**	**System should utilize user's schedule in order to recommend optimal times to exercise**
REQ - 5	**3**	**System should determine the location of the user**
REQ - 7	**4**	**System should log user's input and display graphs and statistics for each piece of information**
REQ - 8	**1**	**System should use the phone's built in GPS to track distance walked throughout the day**
REQ - 9	**5**	**System should allow user to add and take away from their schedule as needed**
REQ - 10	**2**	**System should receive user input for daily miscellaneous responsibilities and activities**
REQ - 11	**5**	**System should let users share their health index with others.**

2. Enumerated Non-functional Requirements

ID	Priority Weight	Requirement
REQ - 12	**2**	**System should use minimal processing power, in order to lengthen battery life.**
REQ - 13	**4**	**System should be easy to recover in case of data loss or change of phone.**
REQ - 14	**5**	**System should be secure in that only the user should**

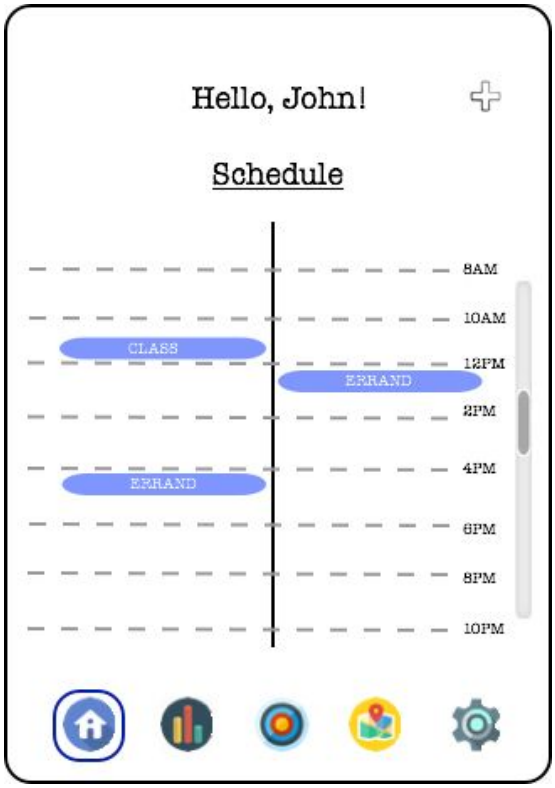
		be able to access their daily logs and personal info.
REQ - 15	**5**	**System should be updated as new research is discovered about timing and stress management.**
REQ - 16	**3**	**System should be simple and elegantly designed.**

3. On-Screen Appearance Requirements

There are 5 tabs, that the user can swipe left and right to navigate around. The current sketches show the main functions of the app. The design takes on a modern yet simple appearance

Opening Screen

Upon opening the app, a loading screen will appear with the logo of the app.

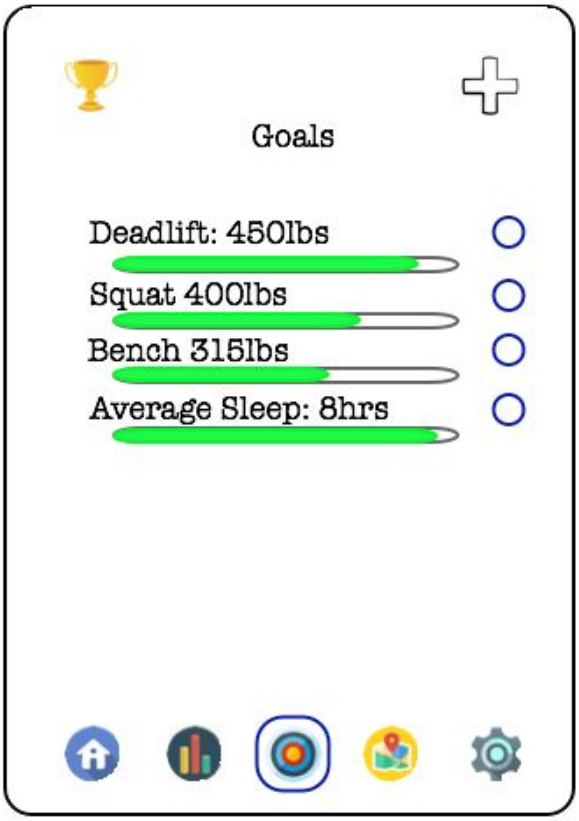


Home Tab

Upon opening the app, the home page will load (After the login page) showing the user's schedule for the day. On the top right, there is an add button to events in schedule or add friends.

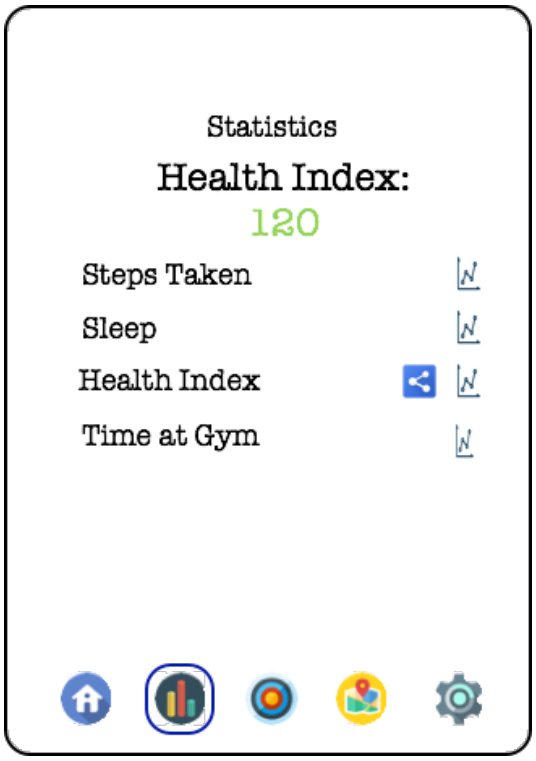
Location Tab

In the location tab, your current location is displayed with a search bar.



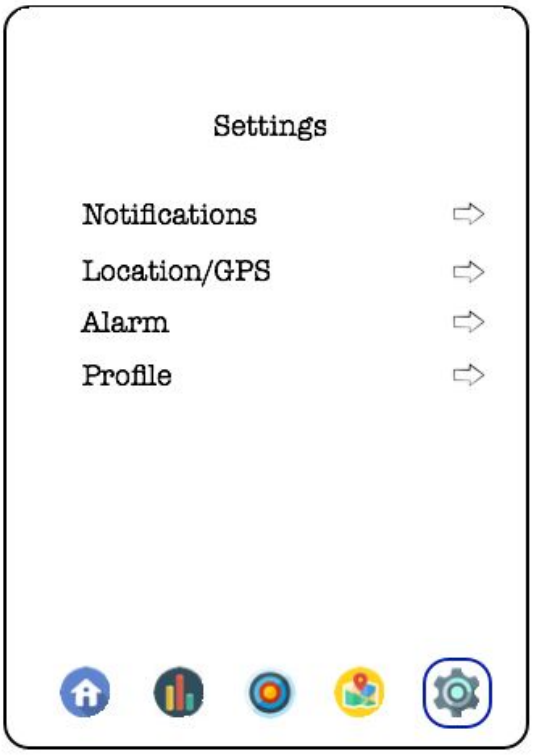
Goals Tab

In the goals tab, you are able to set/update goals, view your progress, and view your accomplished goals (top-left).



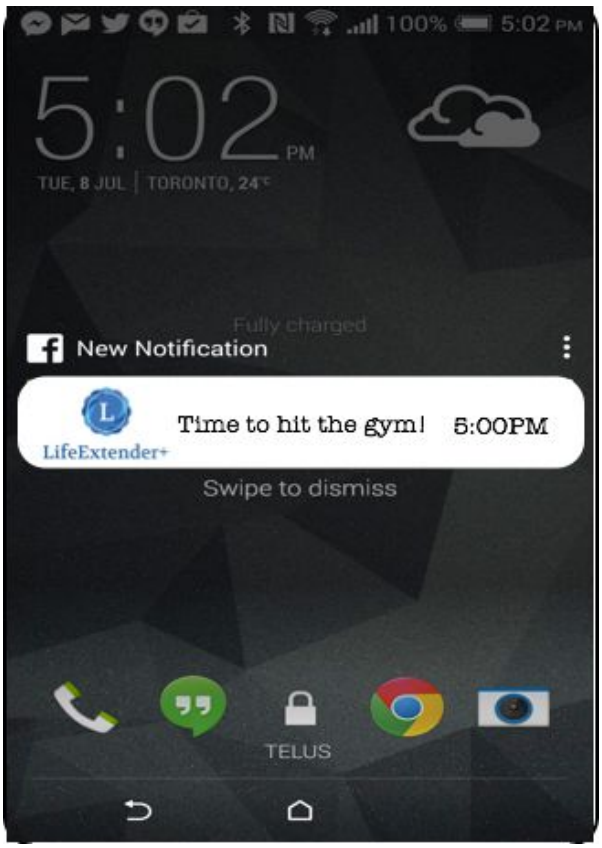
Statistics Tab

In the statistics tab, user will be able to see their health index, share their health index and view graphs of various statistics.



Settings Tab

In the settings tab, you will be able to customize the app to your preference. You will be able to choose what notifications to receive, how GPS is set, alarm settings, and a profile settings for whether you connect with Google Calendar or Facebook.



Notification Screen
A typical notification to be received at designed times.

IV. Functional Requirements Specification

1. Stakeholders

Stakeholders are humans or human organizations that have an interest in the software. Since the application will be health related, our primary audience will be those who are affiliated with college and interested in exercise. Other proponents of the application may consist of those who are already versed in fitness. Possible stakeholders for this software are but not limited to:

1. College Students:

Those who have trouble balancing a healthy life consisting of exercise, food, and sleep with school.

2. University Administration:
Faculty and staff who are either also interested in balancing exercise, food, and staff with work or intent on improving the student body in health and academics

3. Fitness Centers and Gyms:
Gyms that are represented in nearby locations that want to promote health and/or their business

4. Exercise and Sports Stores:
Establishments that want to promote their fitness related equipment, apparel, accessories, etc.

5. Fitness Industry:
Professional athletes, bodybuilders, and fitness celebrities who want to promote health and/or seek sponsorships

2. Actors and Goals

1. User: a registered user
2. Friend: a special case of the actor "User," related to User
3. Database: records the Users' performance such health index, and other statistics

Actors:	**Goals:**	**Use Case:**
User, Database	To view users' own health index by retrieving from the database	Gethealthindex (UC-1)
User, Friend, database	To retrieve users' health index from the database and share with friend	ShareHealthIndex (UC-2)
User, Database	To view friends' health index from the database	ViewOtherIndex (UC-3)
User, Database	To create a user account and store user information (ID, password) into the database	Register (UC-4)
User, Database	To retrieve various users' statistics from the database and	ViewStatistics (UC-5)

	load when viewing in the Statistics tab	
User, Database	To login into users' account on the database so the system can track performance	Login (UC-6)
User, Database	To change or save various settings from and into the database	ChangeSettings (UC-7)
User, Database	To save users' goals into the database and load when viewing in the Goals tab	SetGoals (UC-8)
User, Database	To add or delete events or tasks into users' schedule	QuickChange (UC-9)

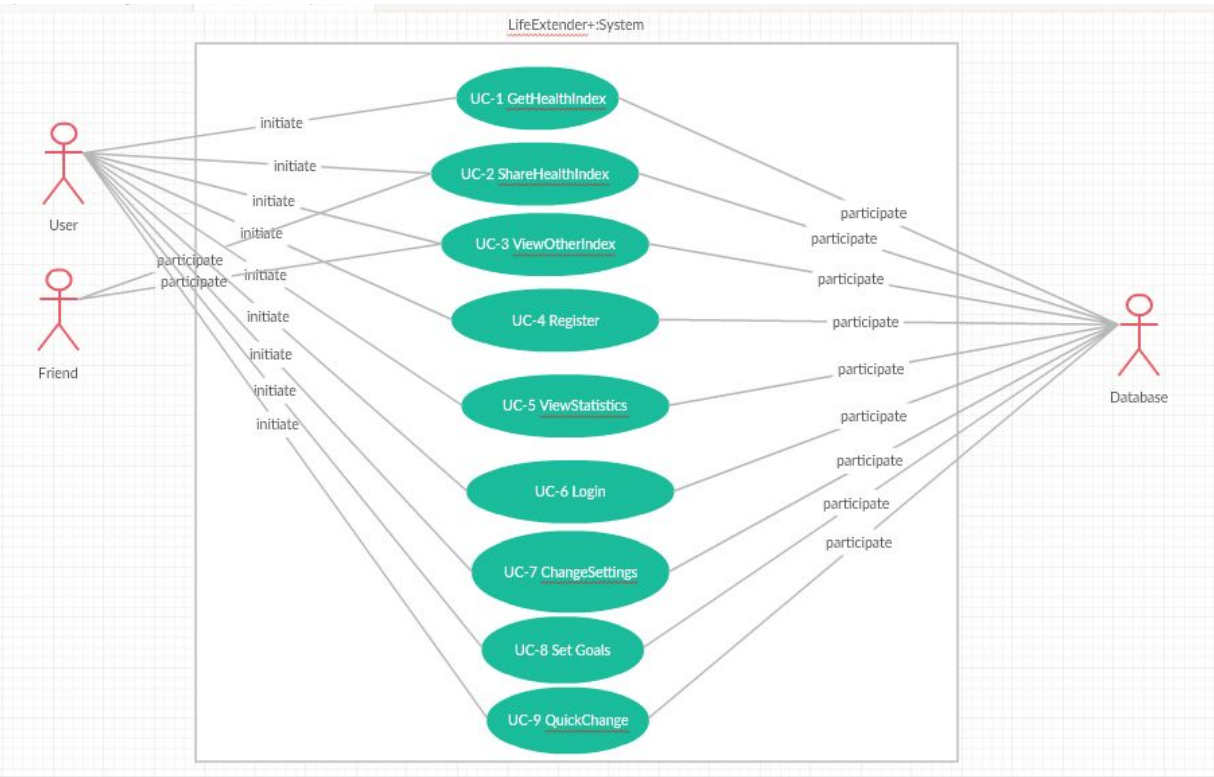
3. Use Cases

i. Casual Description:

Use Case	Use Case Name	Description
UC-1	GetHealthIndex	The user wants to see a public health index based on daily, monthly, and yearly data that is calculated by the system. The system will display the user's health index number compared to previous dates by aggregating weekly, monthly, and yearly data
UC-2	ShareHealthIndex	Allows users to export their health index number to share with others. Only the aggregated number would be shared in order to easily compare standings
UC-3	ViewOtherIndex	User can import other user's health index data from their exported data. It will only include information that the sharing user opts to share, and will only be general information.
UC-4	Register	The user can register events, classes, work hours,

		errands, etc… into their schedule, and link it to a new or existing account that stores index data
UC-5	ViewStatistics	The user can display statistics of their own physical activity while using this app, and as well as information about their health and health index.
UC-6	Login	On opening the app, the user is prompted with a login screen, where they must input their username and password to access their data and use the functions in the app.
UC-7	ChangeSettings	The user will be able to adjust settings such as GPS update frequency, notification settings, desired time between events (for more customized scheduling), privacy settings, alarms, and change their password. Here the user will also be able to restore data in case of data loss or if they change their phone.
UC-8	SetGoals	The user will be able to open a prompt and set goals for themselves such as average sleep, calories burned through walking, weight loss goals, or specific goals based on various exercises.
UC-9	QuickChange	User would be able to add or delete an event to their schedule in case of unpredictable schedule. The system would automatically adjust schedule accordingly.

ii. Use Case Diagram:



iii. Traceability Matrix:

	UC-1	UC-2	UC-3	UC-4	UC-5	UC-6	UC-7	UC-8	UC-9
REQ-1				X		X			X
REQ-2				X					X
REQ-3				X			X	X	X
REQ-5						X	X		
REQ-7	X							X	
REQ-8	X						X	X	

REQ-9	X				X				
REQ-10	X							X	X
REQ-11		X	X				X		

iv. Fully- Dressed Description:

Use Case UC-1: Get Health Index
Related Requirements: ● REQ-7, REQ-8, REQ-9, REQ-10, REQ-11 **Initiating Actor:** ● User **Actor's Goal:** ● To retrieve health index from the database **Participating Actors:** ● Database **Preconditions:** ● User has the application loaded ● User is logged into account **Postconditions:** ● User is shown health index number at the top of the Statistics tab
Flow of Events for Main Success Scenario: 1. -> User selects the button for health index 2. <- System retrieves health index number from database based on user input and activity 3. -> User waits for system to update 4. <- System shows user's health index, and an option to share the number with friends **Flow of Events for Alternate Success Scenario:** 1. -> User selects the button for health index 2. <- System tries to retrieve health index number but cannot due to no data 3. -> User waits for system to update 4. <- System displays the error message: "Insufficient data for health index number!"

Use Case UC-4: Register

Related Requirements:
- REQ-1

Initiating Actor:
- User

Actor's Goal:
- To add a new event to the user's schedule.

Participating Actors:
- Database

Preconditions:
- Time in the designated slot is initially free in the user's schedule

Postconditions:
- A new event, class, or work hours object is added to the user's schedule.

Flow of Events for Main Success Scenario:
1. -> The user selects the Register tab
2. <- The system displays a prompt for information about the event, including start time, ending time (or duration of event), name of the event, recurring or nonrecurring, if the user would like a notification before the event, etc...
3. -> The user will enter all the relevant data and hit submit.
4. <- System will add the new event to the database and the user's schedule and it can be viewed from the schedule window. If the event overlaps with a suggested Meal Time, Gym Time, or Sleep Time, those events may be rescheduled.

Flow of Events for Extensions:
1. -> The user selects the Register tab
2. <- The system displays a prompt for information about the event, including start time, ending time (or duration of event), name of event, recurring or nonrecurring, if the user would like a notification before the event, etc…
3. -> The user will enter all the relevant data and hit submit.
4. <- The system will notify the user that they have entered an event that overlaps with a previously entered event.
5. -> The user can choose to discard one of the two events, or to allow the two events to overlap.
6. <- The system will act accordingly with the user's choice, and update the database with the new event. It will also reschedule any suggested Meal Time, Gym Time, or Sleep Time, should those events also pose a conflict.

Use Case UC-5: View Statistics

Related Requirements:
- REQ-10

Initiating Actor:

- User

Actor's Goal:

- To view personal information, including time spent on walking, running, and at gym

Participating Actors:

- Database

Preconditions:

- User is on the home screen

Postconditions:

- Time in minutes spent on walking, running, and at gym retrieved from database and loaded onto screen

Flow of Events:

1. -> User selects View Statistics button
2. <- System displays multiple options for the user to access, including calorie count and time spent at the gym
3. -> User selects which options they want to see
4. <- System displays the day statistics stored locally for the option selected, in the form of numbers and line graphs, and prompts user for options for weekly or monthly data
5. -> User can choose to display weekly, monthly, or cancel

Flow of Events for Extensions:

1. -> User selects View Statistics button
2. <- System displays multiple options for the user to access, including calorie count and time spent at the gym
3. -> User selects which options they want to see
4. <- System accesses the database to try to find the data necessary
5. -> User waits
6. <- System displays, "Cannot find data needed!"

Use Case UC-7: Change Settings

Related Requirements:

- REQ-3, REQ-5, REQ-9, REQ-12, REQ-13, REQ-14

Initiating Actor:

- User

Actor's Goal:

- To change settings for the app to personalize their usage of the app, or to change the way the app functions to meet the needs of the user.

Participating Actors:

- Database
- GPS

Preconditions:

- Logged into user account

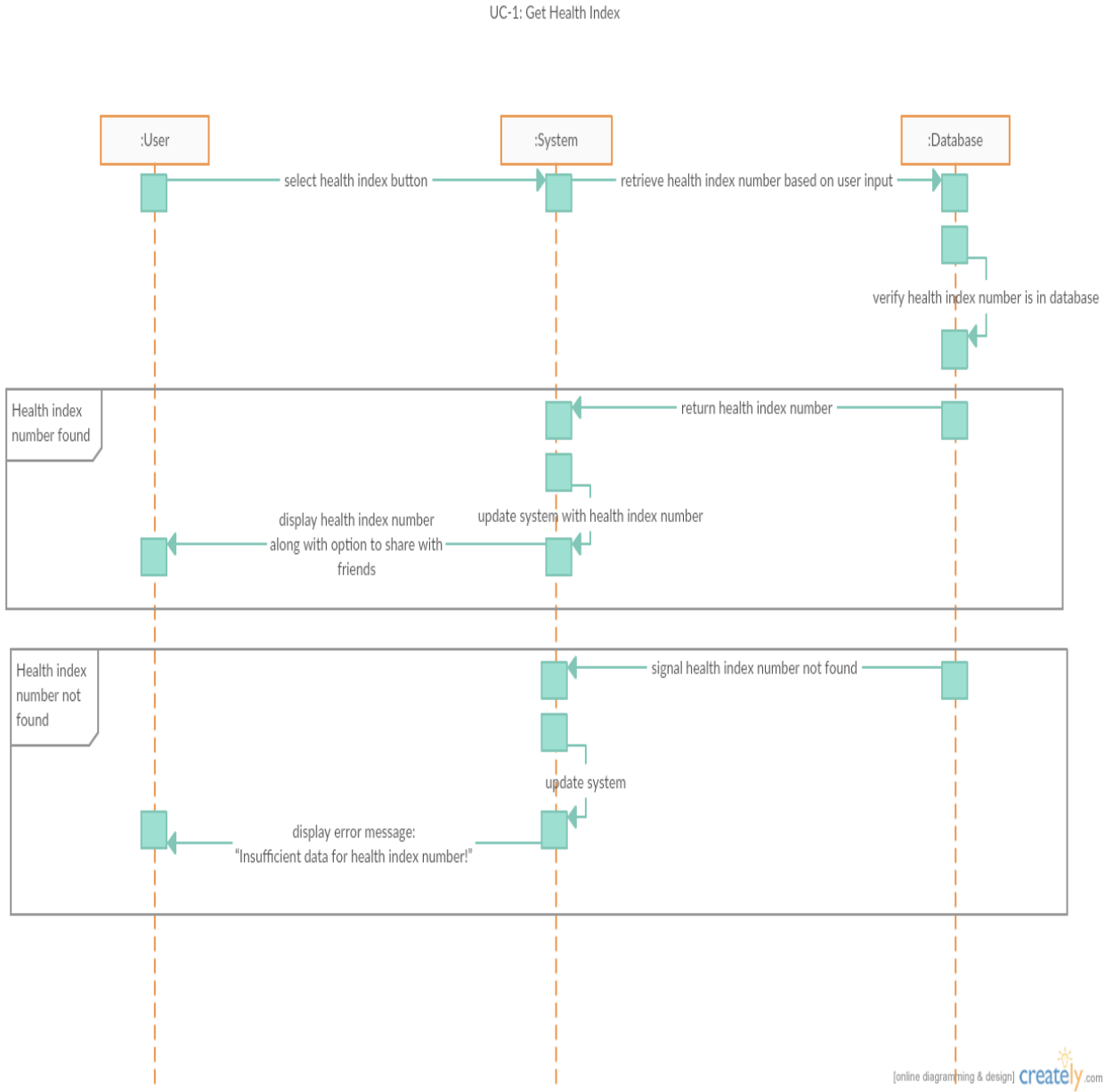
Postconditions:

- Changes in settings are updated into the database

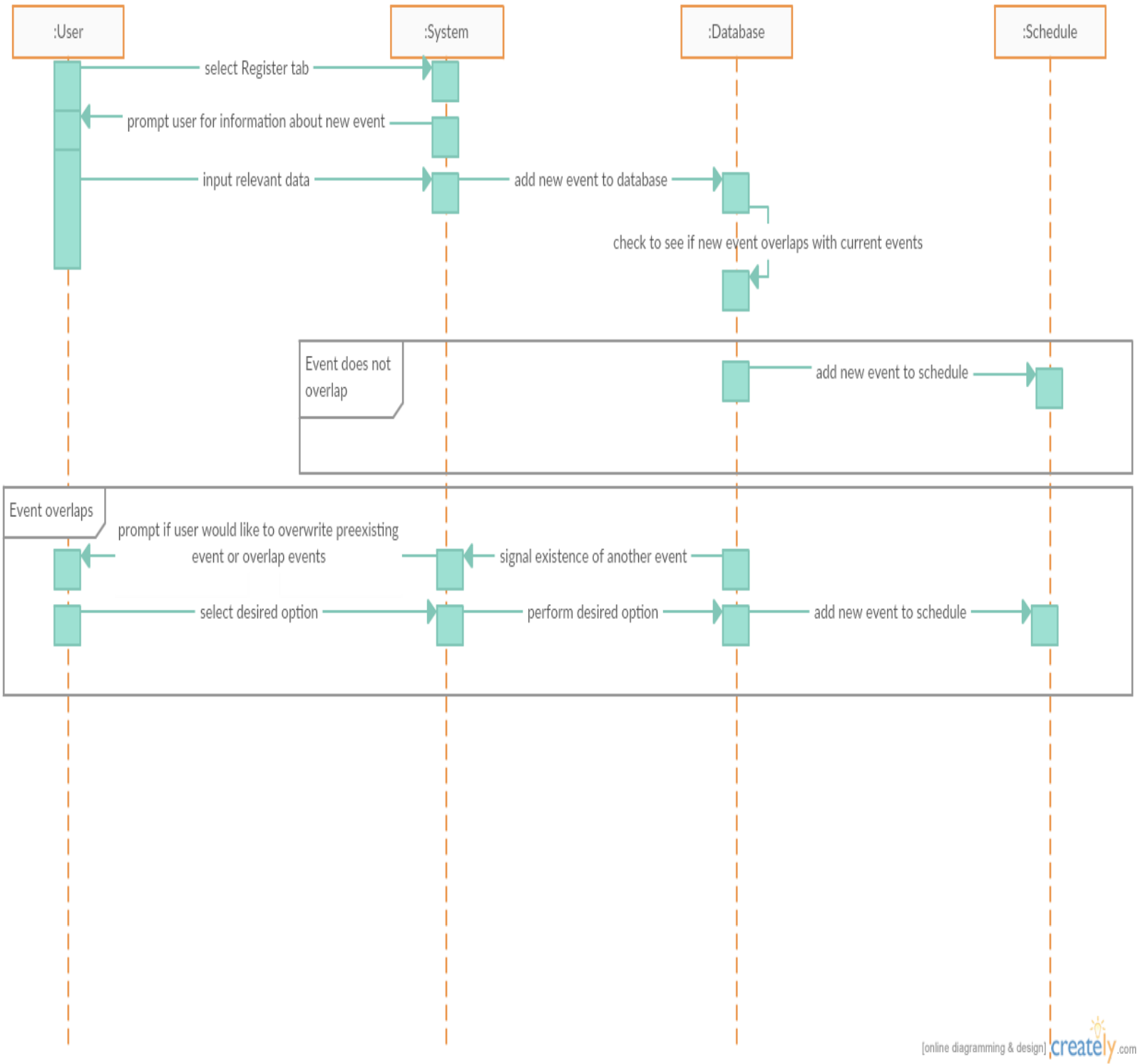
Flow of Events:

1. -> The user opens the settings menu
2. <- The system displays the available options. These settings include notifications, which the user can turn on and off, GPS polling frequency, which the user can increase for walking distance accuracy, or decrease in the interest of battery life, desired time between events (for example, how much time does the user need between events to travel, get ready/dressed, etc…), and the option to change their password.
3. -> The user may go through each option, and make changes as he or she sees fit. The user will then hit save.
4. <- The system will read the updated setting as requested by the user and will change the functionality of the application accordingly.

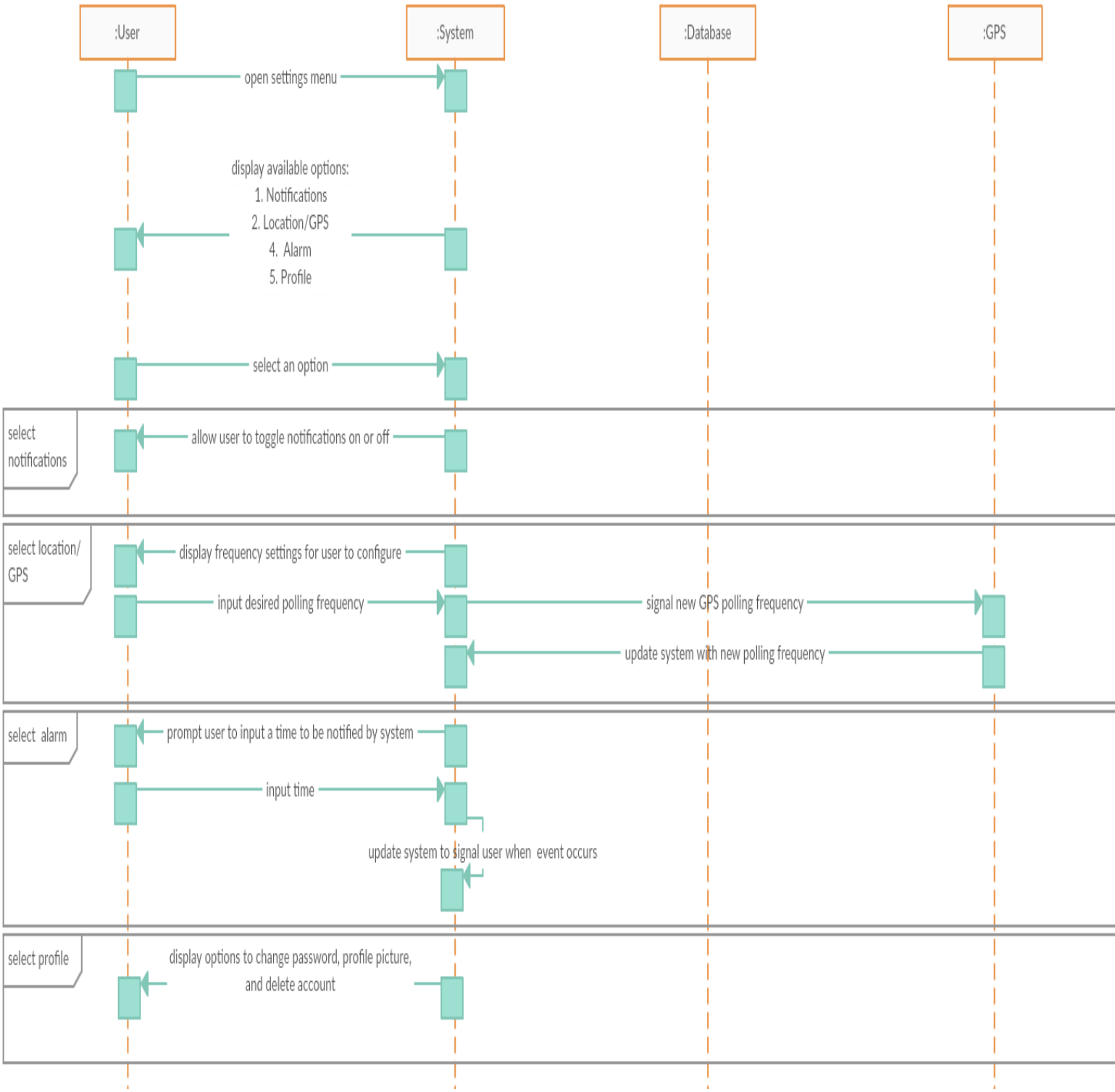
4. System Sequence Diagrams



UC-4: REGISTER



UC-7: Change Settings



V. Effort Estimation

Event: Login into LifeExtender+ with an existing username and password
Navigation: *1 touch strokes*
--after completing data entry--

a. Touch "log in" button to finish

Data Entry: *2 touch strokes and 21 keystrokes*

a. Touch on "username" text box
b. Type out appropriate username (userexample)
c. Touch on "password" text box
d. Type out appropriate password (expassword)

Event: Add an active period event called GYM to schedule
Navigation: *2 touch strokes*

a. Touch the plus button in the top right corner of the home screen
b. Touch the "Event" button in the drop down menu from the plus button
c. *--after data entry---*
d. Touch the add button to finish

Data Entry: *4 touch strokes and 3 keystrokes*

a. Swipe up and down for the appropriate date and time
b. Type out "GYM" for the name of the event

Event: Share health index with your friend, James
Navigation: *3 touch strokes*

a. Touch the graph icon on navigation bar
b. Touch the "share" button
c. *--after data entry--*
d. Touch the "okay" button

Data Entry: *1 touch stroke*

a. Select James from friends list

Event: Add a new goal to “run a mile”
Navigation: *4 touch strokes*

a. Touch the bullseye icon which brings up the goal tab
b. Touch the plus button
c. Touch the "New Goal" option

d. *--after data entry--*
e. Touch the green checkmark button

Data Entry: *8 keystrokes*

a. Type out "run a mile"

Event: Delete an active event called GYM from schedule

Navigation: *2 touch strokes*

a. Touch the event on the schedule on the home screen
b. Touch the “Delete?” button next to the event

Data Entry:

N/A

Event: Change location/GPS settings settings to 1 minute

Navigation: *3 touch strokes*

a. Touch the gear icon which brings up the settings tab
b. Touch the arrow next to Location/GPS
c. *--data entry--*

Data Entry:

a. Touch the interval option “1 Minute Interval”

Event: Update the goal “run a mile” to “run a half-mile”

Navigation: *4 touch strokes*

a. Touch the bullseye icon which brings up the goal tab
b. Touch the plus button
c. Touch the “Edit Goal” option
d. *--after data entry--*
e. Touch the green checkmark button

Data Entry: *1 touch stroke and 5 keystrokes*

a. Touch the goal “run a mile”
b. Type out “half-”

VI. Domain Analysis

1. Domain Model

1. Concept Definitions

Responsibility	**Type (*D for doing; K for knowing)***	**Concept**
R1. To display all user data and info and receive input from the user	D	Interface
R2. Calculates the user's health index using data from AnalyaticDataGoals	D	HealthIndexCalculator
R3. Stores all friends' and user's health index	K	HealthIndexStorage
R4. Creates lines graphs of user's health data such as total hours of active period, change in health index, walking distance in various time spans (yearly, monthly,daily)	D	GraphMaker
R5. Stores user's health information such as total hours of active period, change in health index, walking distance, and user's goals	K	AnalyticDataGoals
R6. Creates the user's schedule with suggested active periods based off of user's input	D	ScheduleMaker
R7. Coordinate actions of concepts associated with use case	D	Controller

R8. Lets user change settings and preferences and retrieves user settings from database	D/K	Settings
R9. User inputs login information	D	LoginEntry
R10. Display status of login	D	StatusDisplay
R11. Checks whether user inputted a valid login	D	LoginChecker
R12. Retrieves and stores schedule from database	K	ScheduleStored
R13. Stores location data like active period location and walking location	K	LocationStorage
R14. Register user's info and other new user info and sends it into the database	D	Register
R15. Sends user's friend request and receives response	D	FriendRequest

2. Association Definitions

Concept Pair	**Association Description**	**Association Name**
Controller ↔ Interface	Controller will receive requests from the interface and the controller will send information to interface to display.	Provides data

HealthIndexCalculator ↔ HealthIndexStorage	Healthindexcalculator sends the calculated health index to HealthIndexStorage	Provides Data
Controller ↔ FriendRequest	Controller will send user's requested friend info to FriendRequest	Provides data, Convey Request
Controller ↔ ScheduleMaker	Controller will send user's inputted scheduled to ScheduleMaker and ScheduleMaker will send a recommended schedule with active periods	Provides data
Controller ↔ AnalyticDataGoals	Controller request analytical data from AnalyticDataGoals and AnalyticDataGoals will send it.	Conveys request
AnalyticDataGoals ↔ HealthIndexCalculator	AnalyticDataGoals will send user's health information to HealthIndexCalculator to calculate the health index	Conveys Request
HighlightMaker ↔ AnalyticDataGoals	HighlightMaker utilizes data from Controller and sends it to AnalyticDataGoals	Provides data
Controller ↔ LocationStorage	Controller will convey a request for location information from LocationStorage and also provide user inputted location data (for designated active	Provides data/ Convey request

	period location) to LocationStorage	
LocationStorage ↔ AnalyticDataGoals	LocationStorage sends AnalyticDataGoals information on user's walking distance and amount of time in certain active period locations	Provides data
Controller ↔ ScheduleStored	Controller sends ScheduleStored the user's inputted schedule	Provides data
Controller ↔ Settings	Controller requests Settings to save certain parameters of the application dependent on the user	Requests save
Controller ↔ LoginChecker	Controller provides LoginChecker with user login information to be verified	Provides data
Controller ↔ LoginEntry	Controller requests for user login information	Conveys request
GraphMaker ↔ AnalyticDataGoals	GraphMaker sends AnalyticDataGoals the graph of all analytics	Provides data
Controller ↔ StatusDisplay	Controller generates what needs to be displayed on the login screen	Generates

3. Attribute Definitions

Concept	Attribute	Attribute Description
LoginEntry	Number of Login Tries	Used to ask for information
LoginChecker	Retrieve Login Data	Used to retrieve information
FriendRequest	Send Request, Receive Response	Used to ask for information
LocationStorage	Time in Active Period's Location	Used to store how long user was in active period's location
HealthIndexStorage	User's Health index Friend's Health Index	Used to store friend's health index and user's.
HealthIndexCalculator	Health index equation	Used to calculate user's Health Index information
AnalyticDataGoals	Goals, Time at Active Period, Sleeping Time, distance walking	Used to ask for short/long term goals, stores information, and saves to display
ScheduleMaker	User inputed Schedule, Generated Schedule with active periods	Used to store user's generated schedule and generated schedule with active periods
GraphMaker	Stats,Graph	Used to take in user's analytics, shows generated visual
ScheduleStored	Schedule	Used to store info from database

4. Traceability Matrix

		Domain Concept	--------	--------	------	----------	--------	--------	-------
Use Case	PW	Interface	Health Index Calculator	Health Index Storage	Graph Maker	AnalyticData Goals	Schedule Maker	Controller	Settings
UC-1	3	x	x			x		x	
UC-2	2	x		x		x		x	
UC-3	2	x				x			
UC-4	4	x						x	
UC-5	4	x			x	x		x	
UC-6	5	x						x	
UC-7	4	x						x	x
UC-8	3	x				x		x	
UC-9	4	x					x	x	

		Domain Concept	--------	--------	--------	----------	----------
Use Case	PW	LoginEntry	Status Display	Login Checker	Schedule Stored	Register	Friend Request
UC-1	3		x				
UC-2	2						
UC-3	2		x				x
UC-4	4				x	x	
UC-5	4		x				
UC-6	5	x	x	x			
UC-7	4						

UC-8	3		x				
UC-9	4		x		x		

2. System Operation Contracts

OC-1: Get Health Index ● Precondition: User is on the Statistics Tab ● Postcondition: User is shown their Health Index
OC-2: Share Health Index ● Precondition: User is on the Statistics Tab ● Postcondition: User is shown their Health Index, and is shown a friends list with options to share with each individual.
OC-3: View Other Index ● Precondition: User is on the Home Tab of the app ● Postcondition: User is shown a list of their friends, along with each friend's health index
OC-4: Register ● Precondition: User is using the app for the first time, or had logged out previously. ● Postcondition: User on the login screen, ready to log in with their newly created profile.
OC-5: View Statistics ● Precondition: User is on Statistics tab of the App ● Postcondition: User is shown a list of statistics that they can view as a line graph.
OC-6: Login ● Precondition: User has logged out of the app previously, or is using the app from a phone that is not their own ● Postcondition: User is in the Home Tab, with their profile and info loaded
OC-7: Change Settings

● Precondition: User is on any tab of the app ● Postcondition: User is on the Settings page with their desired setting changed.
OC-8: Set Goals ● Precondition: User is on the Goals Tab of the app ● Postcondition: User is shown a list of their goals, including the one they just added.
OC-9: Quick Change ● Precondition: User is on the Home Tab of the app ● Postcondition: User is shown their updated schedule

3. Mathematical Models

The health index of our application is determined by a fairly simple formula, based on the user's amount of minutes automatically recorded for low intensity and high intensity activities for that day.

The score is 200 by default and will be affected by time spent walking, running, and exercising at the gym.

$H_n = L(x_1) + H(x_2) + H_{n-1}$

- H_n = Health Index today
- H_{n-1} = Health Index yesterday
- H_0 = 200 (Default Health Index)
- x_1 = Time in minutes for low intensity activity
- x_2 = Time in minutes for high intensity activity
- L = 1
- H = 3
- If $x_1 < 30$ AND $x_2 = 0$, then 100 is deducted from the total Health Index

Where L and H are the modifiers for low intensity and high intensity. What this entails is that if the user walks at least 30 minutes or runs or exercises at the gym each day, s/he will meet prevent the 100 health index deduction penalty, encouraging the user to stay consistent. For instance, if the user walks for 60 minutes and goes to the gym for 30 minutes on the second day of using the app, the health index will be calculated as follows:

$H_n = L(x_1) + H(x_2) + H_{n-1}$

$H_n = 1(60) + 3(30) + 200$
$H_n = 350$

More weight is given to time spent doing muscle strengthening activities and gym time because it is these activities that have the most impact on any user's health.

VII. Interaction Diagrams

UC-1: Get Health Index

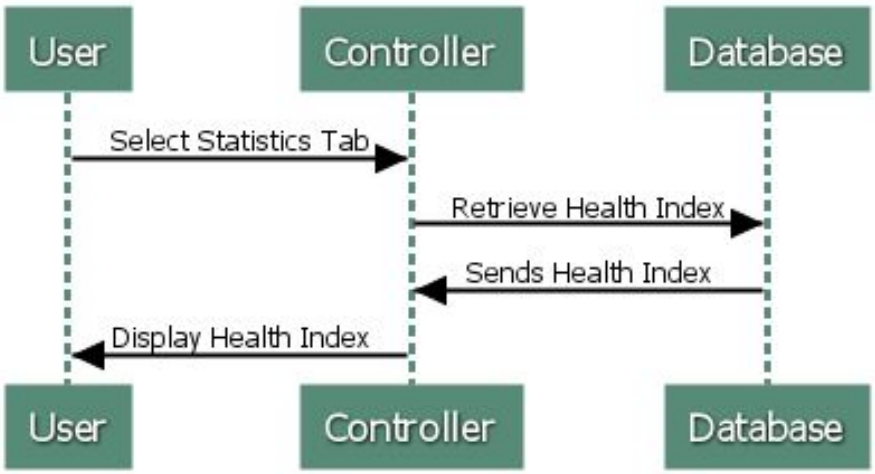


Figure 1-1

The UC-1 diagram for "Get Health Index" shows how the user receives the Health Index Number from the application. When the user selects the statistics tab in the app, the controller will send a request to the database to retrieve the user's health index. Afterwards, the database will send the specific user's health index back to the controller. The controller will then display the health index to the user.

UC-7: Change Setting

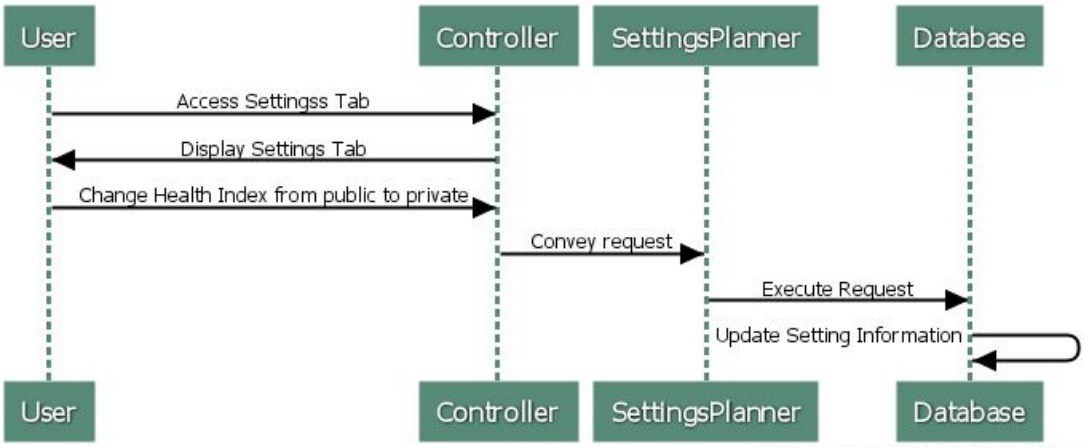


Figure 1-2

The UC-7 diagram for “Change Setting” shows how the user is able to manage his or her personal settings, such as allowing other users to see the user’s health index. When the user selects the settings tab, the controller will display the settings page which will include various features of the application that the user may change that fits his or her preference. The controller will send a request to SettingsPlanner after the user makes any changes, and the database and app functionality will be updated accordingly.

UC-8:Sets Goal

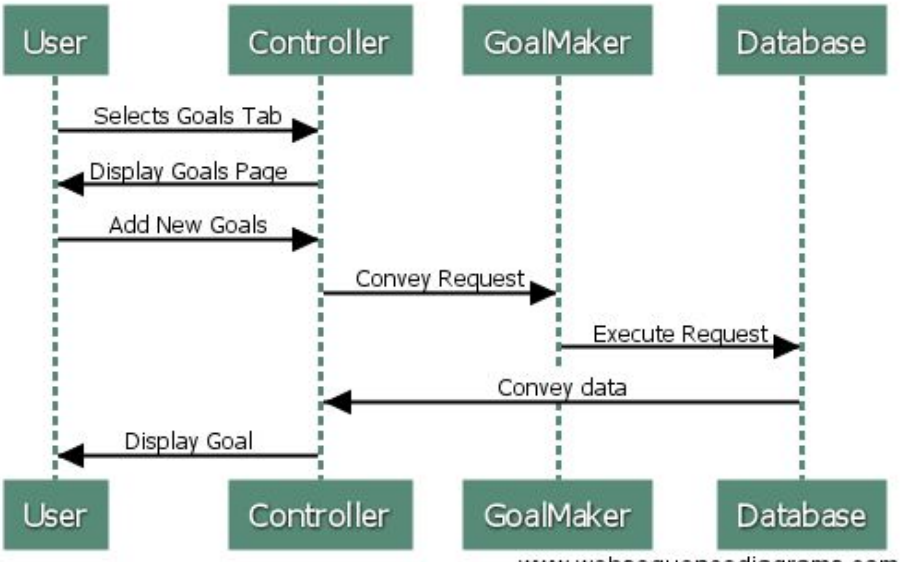


Figure 1-3

The UC-8 diagram for “Sets Goal” shows how the user may go about setting goals with the LifeExtender+ app. To start, the user will select the goals tab to bring up the goals page, which brings up the user’s goals and their progress towards those goals. From the goals page, the

user will be able to add new goals via the app's Controller. After entering all the information through the Controller, the Controller will send the information to the GoalsMaker, which will create the goal through the Database. From there, the Database will return the goal through the Controller, which will display it to the user.

UC-9:Quick Change

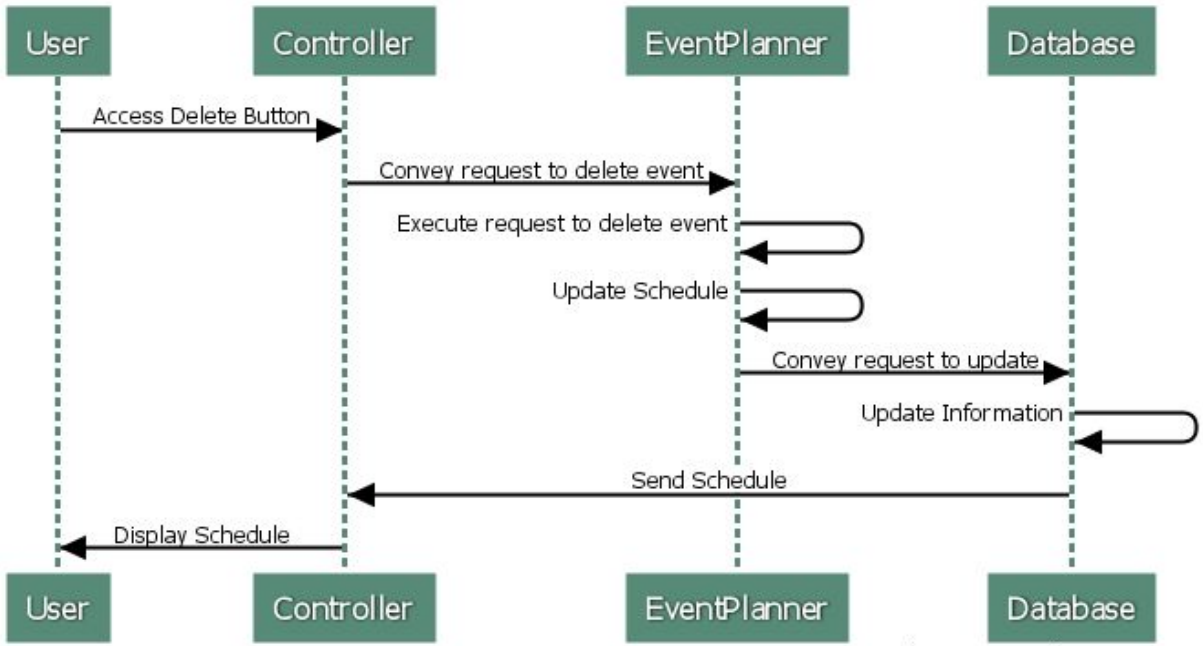


Figure 1-4

The UC-9 diagram for "Quick Change" shows the process that the user goes through in order to change a part of their set schedule. First, the user hits the delete button, which prompts the controller to request the event planner to delete the user's selected event. The event planner will then execute that request, and update the particular user's schedule. The event planner will then send the updated schedule to the Database. The database will then update the user's information and send the updated schedule to the controller. Finally, the controller displays the updated schedule to the user.

VIII. Class Diagram and Interface Specification

*Will be added for Full Report

IX. System Architecture and System Design

1. Architectural Styles

Our system uses a 3-layer architecture consisting of a presentation layer, an application layer, and a data layer. This is analogous to the frontend/backend in website development, where the presentation layer is what is seen by the user, and the application and data layers work behind the scenes.

This scheme is the most logical as the layers are abstracted from each other and will parallel into the code. The differences in UI design, application code, and database calls are apparent. Our project is constructed through Android Studio, where the UI is designed through the program's graphical interface. The application will be "connected" where different parts of UI are matched with the application code. Within the code, database calls will be made with using an API. This structure is easy to follow and implement.

2. Identifying Subsystems

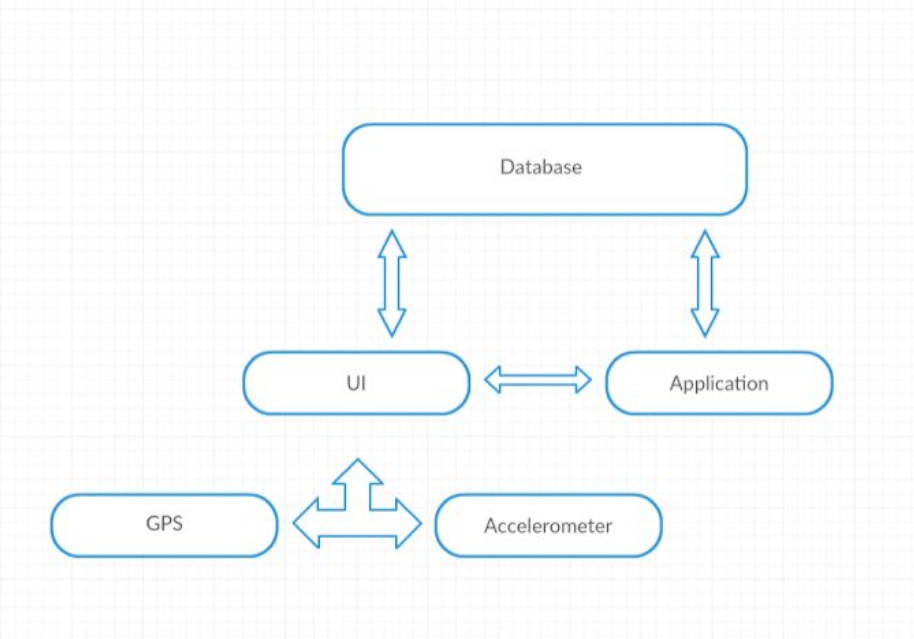


Figure 3-1

The UI subsystem will interact with the GPS and accelerometer subsystems. The Database subsystem saves the profile data of each user and will interact with both the application and database subsystems. The Application layer will handle inputs from the user and interact with the UI subsystem and Database subsystem. This includes the scheduling algorithm, settings modification, the generation of graphs, and calculation of the health index.

3. Mapping Subsystems to Hardware

Our application works with a native device (Android) and a database server. Our application will utilize a database with a public API and will run as long as it is active. The Database subsystem will naturally run the database with the Application subsystem interacting with it.

4. Persistent Data Storage

Because our software will be run on Android, Firebase will be the database used for storage. It implements a self-contained and transactional database engine configured through Android Studios. It is the optimal database for applications because it is a realtime database with cloud storage and functions and APIs packaged into a

single SDK. Classes that primarily interact with the database include SettingsPlanner, GoalsMaker, and EventPlanner. Some cases of interactions between classes and database are changing personal settings through SettingsPlanner, creating user goals through GoalsMaker, and updating user schedule with EventPlanner. Between each exchange of events with the database, the controller will act between the user and the system.

5. Network Protocol

As our project does not require real-time streaming of any kind, our application will be implemented through HTTP connecting from our application to the database server, which is set up through FireBase.

6. Global Control Flow

Execution Orderness

This project will be an event-driven system. The user will have complete control over how they use the application. There is no linear procedure for the user to take, and they do not even have to use all of the features that the app provides. The user can use the app's interface to use any function at any time, such as creating goals or events for their schedule. The GPS, however, is persistent upon opening the app, so that walking/exercise time and distance may be tracked.

Time Dependency

This application features timers for many of its functions. One of the application's main functions is that it will track not only how far the user walks, but for how long the user walks for, in real time. It also attempts to track how long the user uses gym facilities for. Based on how long the user exercises for, the app will increase or decrease the user's health index. Because another main component of this application is the schedule, time is also used to track the nearing or passing of events for the user to try to help keep them on track and on time. One last feature that utilizes the clock is the alarm, which is used to track the user's sleep schedule. The application overall has a dependency on time.

Concurrency

This system will run on multiple threads. The main thread will be the user interface, where the user can access and change the app's functionality, and the subthreads will be all of the smaller tasks the app does, the GPS and its associated features, and the networking. All

subthreads will have to communicate with the user interface so that the all the information they collect can be viewed by the user, including position, time and distance walked, etc...

7. Hardware Requirements

This software will be run on mobile smartphones with touch screen display and network/WiFi/GPS support. The required operating system will be Android starting from version 5.0, Lollipop. This platform is accessible and the requirements are met through most Android phones.

X. Algorithms and Data Structures

1. Algorithms

Our algorithm will be the primary means of scoring the user's Health Index. With the changes of the Health Index saved into a database, users can see their progress after it is updated every night. Specific classes are able to access the database to either retrieve, input, or update data. The mathematical model to calculate individual Health Indices incorporates a low-intensity and high-intensity modifier depending on how the user is exercising. For example, an activity like walking would be marked as low-intensity while an activity such as going to the gym would be marked as high-intensity. The intensity of the activity will be determined by the GPS; constant slow movement would be recorded as walking, constant fast movement would be recorded as running, a period in an exercise facility would be recorded as going to the gym. These factors would all contribute to increasing the Health Index while ignoring exercise would slowly degrade the Health Index.

$H_n = L(x_1) + H(x_2) + H_{n-1}$

- H_n = Health Index today
- H_{n-1} = Health Index yesterday
- H_0 = 200 (Default Health Index)
- x_1 = Time in minutes for low intensity activity
- x_2 = Time in minutes for high intensity activity
- L = 1
- H = 3
- If $x_1 < 30$ AND $x_2 = 0$, then 100 is deducted from the total Health Index

The optimization of the user's schedule will be based off the user's event inputs and settings. For example, if t > 40 between the end of event1 and the start of event2,

the system may potentially insert an event to prompt the user to eat a meal. The algorithm will analyze the schedule, filter the time slots, and linearly look at the quantified time slot through a series of checks (if-statements) to determine when to notify the user of the time to eat/workout/sleep.

2. Data Structures

The data structure that our system plans to use is the database, more specifically, the Firebase database, which is where all the information pertinent to our system will be stored. Our system will use the following data types in order to accurately monitor the user's active periods and calculate his/her health index:

HealthIndex: double - the user's health index for the current day which will be calculated daily in order to keep progress of the user's performance.

previousHI: double - the user's health index for the previous day which is required to caluculate the user's health index for the current day.

fSchedule: sTable - the user's schedule with Active Period that is synchronized with the database. This table will be used to generate a schedule for the user that shows the chronological breakdown of the user's Active Periods.

TotalLowAPhours: dTable - All of user's low intensity Active Period times according to date in a date table data type which will be synchronized with the database. The amount of time the user spends during low intensity Active Periods is required to calculate the user's health index.

TotalHighAPhours: dTable - All of User's high intensity Active Period times according to date in a date table data type which will be synchronized with the database. The amount of time the user spends during high intensity Active Periods is also required to calculate the user's health index.

The interaction between the database and the two classes, HealthIndexCalc and GraphMaker, will ensure a successful operation of our system. The HealthIndexCalc class is responsible for generating/calculating and storing the user's health index within the database. This is essential for tracking the user's performance on a daily basis. The GraphMaker class is responsible for creating graphs of the user's Active Periods over a certain amount of time. In addition, this class interacts with the database by storing the

user's low and high Active Period times which is essential for calculating the user's health index.

XI. User Interface Design and Implementation

*Will be added for Full Report

XII. Design of Tests

*Will be added for Full Report

XIII. History of Work, Current Status, Future Affairs

History of Work

Most of the history of our work throughout the development of our application has been logged into our group's GitHub (https://github.com/SE2017/LifeExtenderPlus) and follows the Gantt Diagram we created for our plan of work. Some tasks on the projected dates may not have been met exactly on time but were still carried out within a reasonable frame.

Current Status and Key Accomplishments

- Created a Firebase database that stores raw data in realtime
- Connected database to application through Android Studios
- Created simplistic and interactive User Interface
- Integrated multiple APIs to automatically pull schedule and map

Future Affairs

In the future, we would like to add more exercises that can be recorded under low intensity and high intensity activities. With walking, running, and exercising at the gym being the only categories of exercise, the health index may not accurately

represent every user's commitment to health. Currently, the health index is a measure of the user's commitment. Later, either the mathematical model for the health index can be revised to measure the user's progress instead, or a new value can be added to represent progress. We may also add a library of various foods with calories and a food tracking tab so that users can log their daily food intake. Additionally, the application may be updated to be cross-platform and run on various platforms other than Android.

XIV. References

1. Exercise and Psychological Health. 2017. Exercise and Psychological Health. [ONLINE] Available at: https://www.unm.edu/~lkravitz/Article%20folder/exandpsychological2.html [Accessed 04 February 2017].

2. CDC.gov. 2017. How much physical activity do adults need? | Physical Activity | CDC . [ONLINE] Available at: https://www.cdc.gov/physicalactivity/basics/adults/. [Accessed 04 February 2017].

3. Cherry, Kendra. "6 Key Theories of Motivation." *Verywell*. N.p., 14 June 2016. Web. 04 Feb. 2017. <https://www.verywell.com/theories-of-motivation-2795720>.

XV. Project Management

Merging the Contributions from Individual Team Members

Many problems arose from the beginning as members developed individual ideas of what the software's functions and operations should be. As members started meeting more often, we realized many functions were either overambitious or superfluous. A lot of classes we thought would be necessary turned out to be useless or able to be combined with another class. Each iteration of the design made the application more and more utilitarian as we were able to condense and update the essential use cases. As the software became more practical, it also slowly became more feasible. By cross-referencing each teammate's knowledge and past projects, we were able to find many resources necessary to begin drafting the application.

Initial complications for the project included the type of language, platform, database, etc. After team discussions, we concluded that using Java, a general purpose language often used in software and software applications, was the optimal language to use because of its few implementation dependencies. We have also integrated Firebase as the database to operate on all data due to its previously described capabilities. Also, we decided to have the application run on Android starting from one of its most commonly used Operating Systems, Lollipop. We have designed the program fully on the IDE, Android Studios, for the easiest Android application development