**CSE 682 Team #5**

**Software Requirements Specification**

**Healthify Windows Application**

Version: 1.27

Team Members:

Nissa Lott, Julia Thee, Kuohsun Tsai, Nishanth Chandrasekaran, Kenneth Smith

Course: CSE 682 Software Engineering

Revision Date: September 19, 2019

Table of Contents

[Introduction 4](#_Toc19799500)

[1. Hardware Requirements 5](#_Toc19799501)

[2. User Requirements 5](#_Toc19799502)

[3. System Requirements 7](#_Toc19799503)

[4. Nonfunctional Requirements 8](#_Toc19799504)

[5. Appendix: Glossary 8](#_Toc19799505)

[6. Specification Priorities 9](#_Toc19799506)

[7. UML Diagrams 10](#_Toc19799507)

[8. Healthify Application User Flow 18](#_Toc19799508)

[9. Firebase Database 23](#_Toc19799509)

[10. Healthify Sequence Diagram 24](#_Toc19799510)

[11. Source Repositories 25](#_Toc19799511)

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 7/28/2019 | 1.0 | Initial Document Creation after Sunday meeting | Kenneth Smith |
| 7/28/19 | 1.1 | Added security requirements and functional requirements for the authentication agent | Nissa Lott |
| 7/29/19 | 1.2 | Added user stories for agents of the application. | Nishanth Chandrasekaran |
| 7/29/19 | 1.3 | Added user requirements, Back-end Interface, Back-end and Performance requirements | Kenneth Smith |
| 8/11/19 | 1.4 | Updating Format and adjusting specification, Added Functional terms and updated user stories | Kenneth Smith, Nishanth, Tom and Julia |
| 8/11/19 | 1.5 | Updated user stories for eCommerce | Kuohsun Tsai |
| 8/11/19 | 1.6 | Updated each requirements section, added glossary, and formatted document | Julia Thee |
| 8/12/19 | 1.7 | Updated introduction and glossary | Kuohsun Tsai |
| 8/12/19 | 1.8 | Updated glossary section | Kenneth Smith |
| 8/13/19 | 1.9 | Added UML use case diagram | Julia Thee |
| 8/13/19 | 1.10 | Changed numbering values of specifications, Reworded specification contents | Kenneth Smith |
| 8/14/19 | 1.11 | Tied in the User and System specifications | Kenneth Smith |
| 8/14/19 | 1.12 | Added user case diagram | Kuohsun Tsai |
| 8/14/19 | 1.13 | Added additional hardware requirements. | Nishanth Chandrasekaran |
| 8/14/19 | 1.14 | Added the priority section and included an additional use case diagram. | Nishanth Chandrasekaran |
| 8/14/19 | 1.15 | Updated doc. based on meeting with Prof. | Kenneth Smith |
| 8/14/19 | 1.16 | Added Hyperlinks to the priorities | Kenneth Smith |
| 8/20/19 | 1.17 | Added UML diagram | Kuohsun Tsai |
| 9/12/19 | 1.18 | Added calculators UML class diagram and product scope | Julia Thee |
| 9/18/19 | 1.19 | Added state diagram and updated content | Kuohsun Tsai |
| 9/18/19 | 1.20 | Updated calculator UML class diagrams and updated diagram descriptions | Julia Thee |
| 9/18/19 | 1.21 | Updated BMI section, Added Application User flow section | Kenneth Smith |
| 9/18/19 | 1.22 | Added Firebase and Authentication section | Nishanth, Kenneth |
| 9/18/19 | 1.23 | Updated E-Commerce Diagram | Kuohsun Tsai |
| 9/19/19 | 1.24 | Added Sequence Diagram for the whole application. | Nishanth Chandrasekaran |
| 9/19/19 | 1.25 | Added detail on MyUser class and expanded detail on firebase. | Nishanth Chandrasekaran |
| 9/19/19 | 1.26 | Added MyUser class UML Class Diagram | Nishanth, Kenneth |
| 9/19/19 | 1.27 | Added Source Repositories | Kenneth Smith |

# Introduction

**Purpose**

The purpose of this software requirements specification document is to support the development of **Healthify** Windows application. The requirements will cover the full scope of the system and shall support features beyond the first release of application.

**Product Scope**

The Healthify application is intended to provide users with secure, easy-to-use tools for calculating various aspects related to their health. Instead of users spending ample time searching the internet for the optimal ways to lose weight and maintain a healthy lifestyle, this application provides a simple interface for users to input minimal personal details that influence their health (i.e. height, weight, age, etc.), and rapidly receive personalized output regarding their body mass index (BMI), calorie intake, and calories burned through exercise. Users will be able to track their progress over time, as well as purchase various healthcare items through the convenient store available through the application.

# Hardware Requirements

1. The minimum memory requirement is 64 GB.
2. The hardware shall allow the execution of agents.
3. The hardware shall not impede the retrieval or display of results from the agent.
4. Internet connection Broadband with a speed of 4 Mbps or higher.
5. Keyboard and a Microsoft Mouse or some other compatible pointing device.
6. Processor (CPU) with 2 gigahertz (GHz) frequency or above.
7. A minimum of 2 GB of RAM.
8. An operating system with Windows 7, Windows 8 or Windows 10.

# User Requirements

**2.1 Login Flow and Authentication**

1. The user shall be able to register for the app by providing personal details and choosing

a set of credentials for future logins to the app.

1. The user shall be able to login to the app using a unique username and password.

U2.1.3 The user shall be able to reset their password from a password-reset option provided

on the login screen.

U2.1.4 Once logged in, the user shall be able to view and access the different agents present

in the app.

* 1. **BMI Calculator Function**

1. The user shall be able to determine their BMI by inputting various personal details as

requested by the function.

1. Once the BMI has been calculated the user shall have the option to choose if there BMI

results should be saved to a database for the purpose of tracking personal progress.

* 1. **Calorie Intake Function**

1. The user shall be able to input personal health related details to determine the number

of calories to consume daily.

1. The user shall be able to determine the amount of weight to lose daily.
2. The user shall be able to determine the number of calories to maintain their current

weight.

* 1. **Calorie Burning Function**

U2.4.2 The user shall able to determine the approximate number of calories burned while

exercising.

U2.4.3 The user shall able to determine the approximate number of calories burned by

inputting the specific type of exercise, personal details and the amount of time spent

exercising.

**2.5 Progress Tracking Function**

1. The user shall be able to view a chart displaying the progress of their personal BMI values

over time.

1. The user shall have the option of inputting the number of calories that they have

consumed and burned during the current day.

1. The user shall be able to track their progress for calories consumed versus calories

burned per day by viewing a progress chart displaying this data.

**2.6 E-Commerce Function**

U2.6.2 The user shall be able to view and access a health store, which provides the user various

health supplies, products, fitness gear, etc. to purchase.

U2.6.3 The user shall be able to use a credit card payment system in their account to pay

invoice online easily.

U2.6.4 The user shall be able to use shopping cart functionality to easily purchase health care

items online.

U2.6.5 The user shall be able to search and filter products and review product details.

U2.6.6 The user shall be able to add their credit card and checking/savings account to the app

in order to waive any potential fees.

U2.6.7 The user shall be able to use e-commerce platform in the device, operating system,

browser, and network at a page speed of less than five seconds.

U2.6.8 The user shall be able to write customer reviews about their product experiences to

explain if the product fits their desires and expectations.

U2.6.9 The user shall be able to find information about product returns/exchanges to make

returns an easy process.

* 1. **Additional Requirements**

U2.7.2 The user should be able to set up and regularly customize nutrition or fitness goals to

stay on track with a calorie goal for each meal or activity.

U2.7.3 The user should be able to view tips and suggestions concerning workouts to increase

fitness or better nutrition to have a healthy lifestyle.

U2.7.4 The user interaction with the app should be very intuitive and the user should be able to

navigate through all the agents seamlessly without any lapses.

U2.7.5 The user should receive notifications concerning food, water and medicine intake at the

appropriate times.

U2.7.6 The user should be able to set up daily, weekly, or monthly notifications at any point

while logged into the app.

# System Requirements

SR 2.1.1, SR 2.1.2, SR 2.1.3 - The system shall allow the user to enter their unique username and password to login to the app.

SR 2.2.4 - The system shall allow the user to select and launch any agent by clicking that agent.

SR 2.2.2, SR 2.3.2, SR 2.5.2 - The system shall load the correct data when any agent is clicked by the user

SR 2.2.1 - The system shall allow the user to input any requested information within the agents.

SR 2.3.3, SR 2.3.4 - Within each agent, the system shall display accurate results to the user based on their inputted information.

SR 2.2.2 - The system shall allow the user to save BMI values to a database for tracking.

SR 2.4.2, SR 2.4.3 - The system shall allow the user to save their calories consumed and burned per day to a database for tracking.

SR 2.5.2, SR 2.5.4 - The system shall be responsive when the user clicks valid buttons within the app.

* 1. The system shall run until the user closes the app.

# Nonfunctional Requirements

* 1. **Security & Authentication Requirements**
     1. System shall enforce security requirements specific to the Payment Card Industry Data Security Standard (PCI-DSS).
     2. System shall encrypt all data at rest and in transit that is personable, identifiable information (PII).
     3. System shall communicate with other agents using a secure mechanism.
     4. System shall timeout after specific idle session time has expired. The idle time is set within the systems configuration file by the user.
     5. System shall not allow unauthorized access to data.
     6. System shall allow access to data on a need-to-know, least privilege basis.
     7. User shall be able to login to the app using existing social media credentials. The minimal social media sites at Facebook, Google, Twitter and Internet Explorer
     8. User shall be able to create system credentials using email address and strong passwords.
     9. User shall be able to reset passwords for system credentials.
  2. **Performance**
     1. The app and agents will not crash when the user enters data or clicks within the app.
     2. The app will respond to the user within 300ms or faster.
     3. Messages shall be displayed to the user when an error occurs.
     4. The app and agents shall run on Windows system with a minimum of Windows 2010.

# Appendix: Glossary

5.1 **App:** the stand-alone application from which a user can register, log in, and access the individual agent applications

5.2 **Agent:** an independent application within the main stand-alone app that performs specific health-related functions

5.3 **BMI:** body mass index

5.4 **Page speed:** used in reference to the e-commerce platform, which will always load at different speeds due to the differences in byte size and complexity, as well as the operating system, browser, and network having a big impact on page load times

5.5 **UI:** user interface

5.6 **Functionality**: the extent of possibilities provided by a system

5.7 **eCommerce**: the process of buying and selling online or electronically.

5.8 **Gateway**: the platform that processes payments for online purchases.

5.9 **Transaction:** purchasing an order online from a business or other seller.

5.10**Shopping Cart**: an eCommerce shopping cart is the contents of what a user has added to his online order.

5.11 **Life-cycle**: the phases a software product goes through between when it is conceived and when it is no longer available for use*.*

5.12 **Timeliness*:*** the ability of a software system to be released when or before its users want it*.*

5.13 **Unified Modeling Language (UML)**: A graphical language used in object-oriented development that includes several types of system model that provide different views of a system.

5.14 **Use-case diagram**: A UML diagram type that is used to identify use-cases and graphically depict the users involved.

5.15 **State diagram**: A UML diagram type that shows the states of a system and the events that trigger a transition from one state to another.

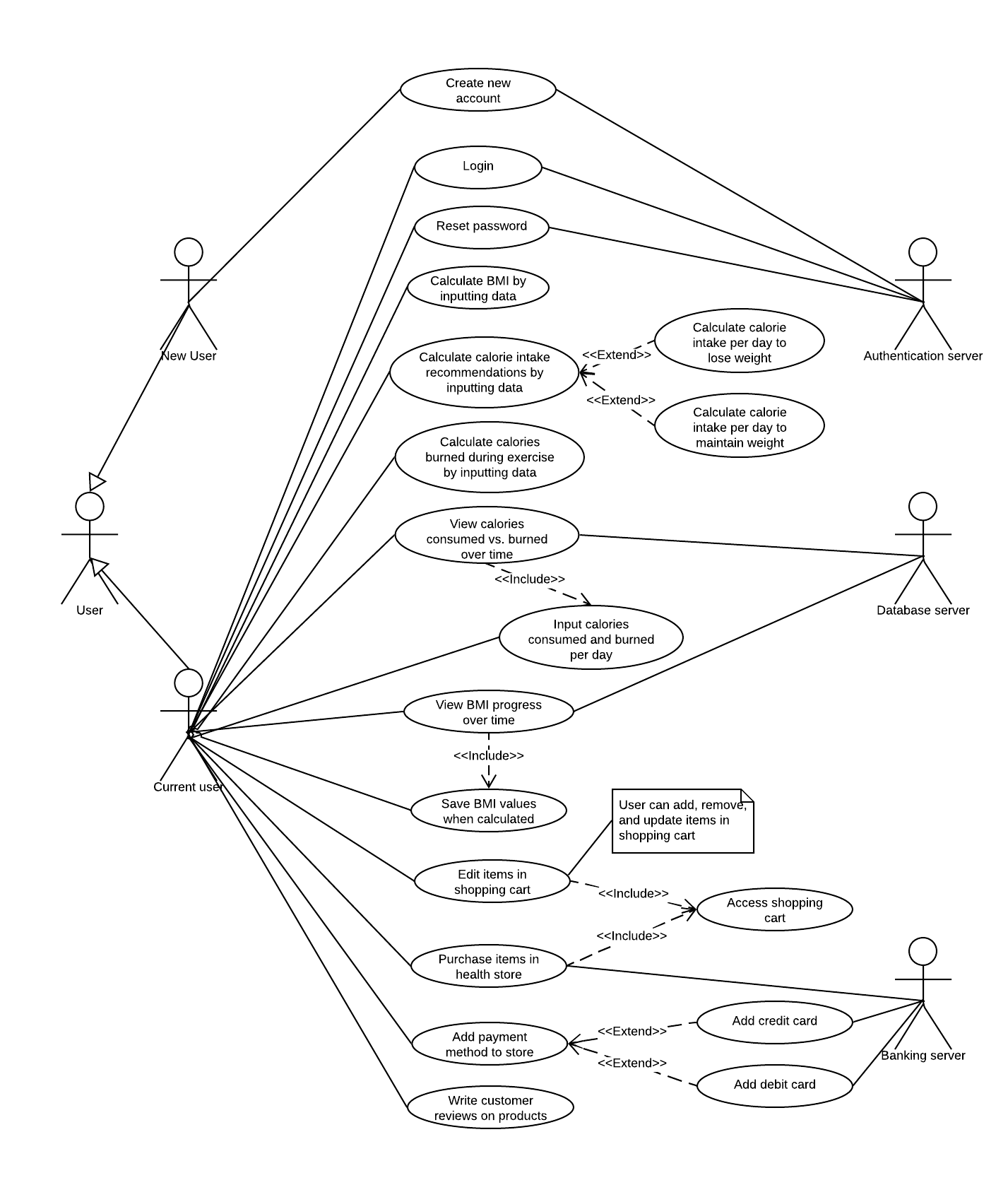
5.16 **login***:* Displays two buttons/forms, one for logging in with an existing user account, the other for registering a new user account.

5.17 **Shoppingcart**: Displays shopping cart contents; form with input fields for changing cart quantities, and buttons for updating quantities, saving the cart, and checkout.

# Specification Priorities

|  |  |
| --- | --- |
| **Priority** | **Requirements** |
| 0 | [U2.1.4](#u214), [U2.1.1](#u211) |
| 1 | [U2.4.2](#u242), [U2.4.3](#u243), [U2.5.4](#u254) |
| 2 | [U2.7.6](#u276), [U2.2.2](#u222), [U2.6.2](#u262) |

# UML Diagrams



Healthify

**System UML Use Case Diagram**

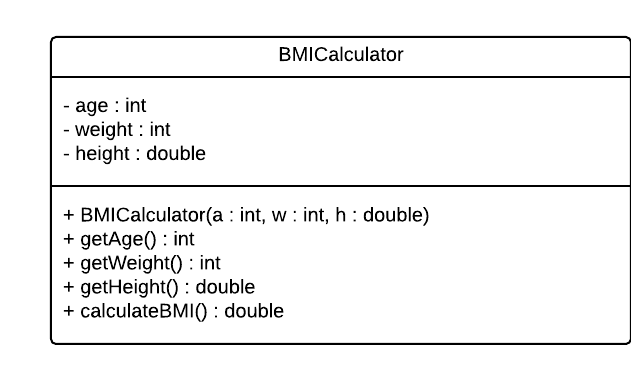
The set of use cases displayed below represent all of the possible interactions that are described in the system requirements.

**Healthify User UML Class Diagram**

This user class is pivotal to the entire application right from authentication and all the interactions with the various agents. The user has to start off during registration has the six important parameters to be inputted to the application ie username, password, full name, gender, age, height and weight. These values are also stored in the firebase real time database which helps in authenticating the user every time he logs in but also retrieving any of the user’s attributes for the agents available.

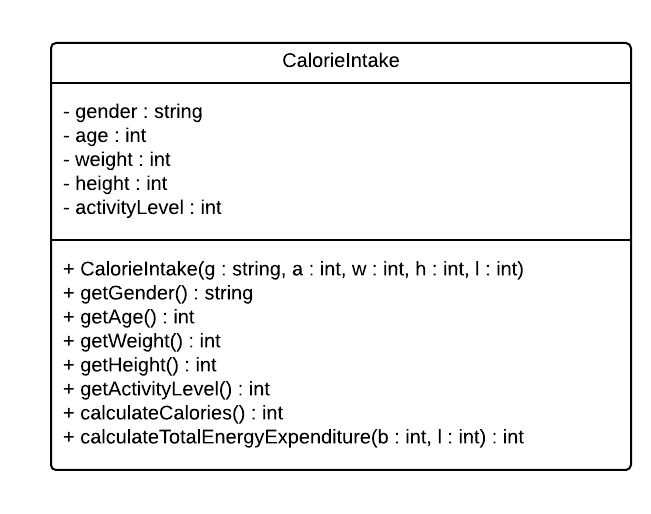
**BMI Calculator UML Class Diagram**

The BMI calculator prompts the user to enter various personal details (i.e. age, weight, height), and uses this information to compute the user’s BMI. The calculator then displays the user’s calculated BMI, as well as the ideal BMI for the user’s age. The calculator also displays ranges of values representing the spectrum of BMI values, from very low to obese ranges. Note: The class, method and data members are represented in the coded form UI control names. For the scope of this document these control execution names will be identified through out the UI code base as pre-set by Visual Studio.



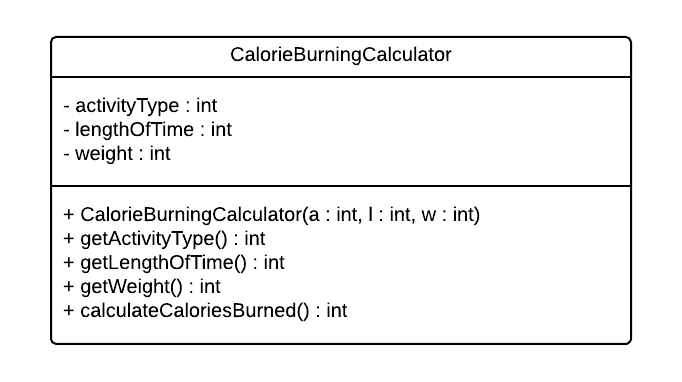
**Calorie-Intake Calculator UML Class Diagram**

The calorie-intake calculator prompts the user to enter various personal details (i.e. gender, age, weight, height, activity level), and uses this information to compute the number of calories that the user should intake per day in order to maintain their current weight. It uses this computed value to calculate the number of calories that the user should intake per day in order to lose weight.

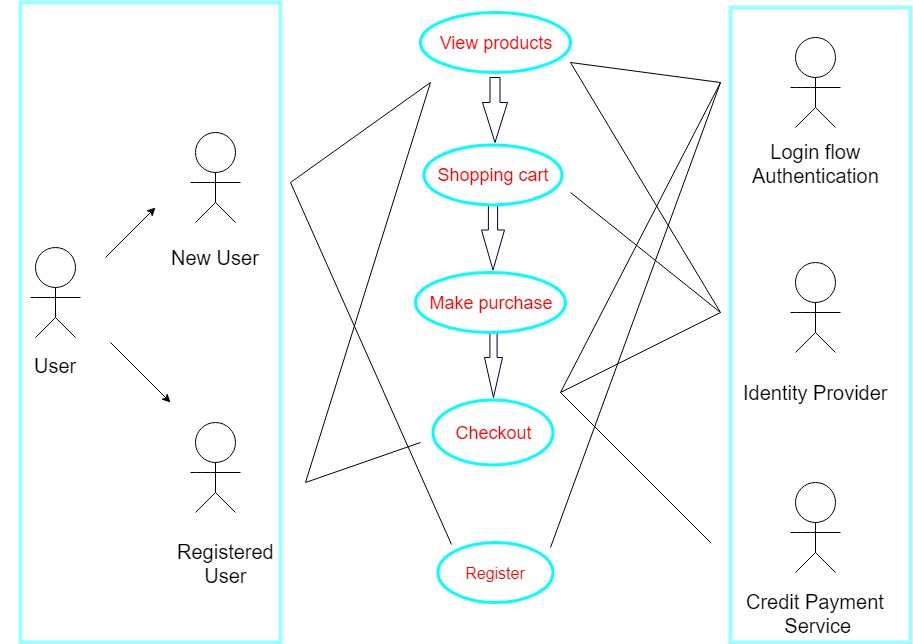


**Calorie-Burning Calculator UML Class Diagram**

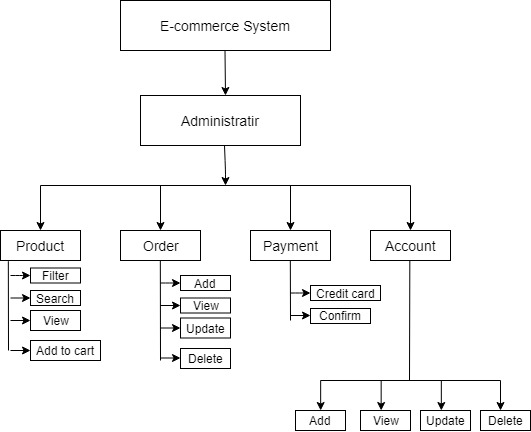
The calorie-burning calculator will be implemented in the next version of the Healthify product. This calculator will prompt the user to enter their weight, along with details regarding the type of exercise or activity they would like to do, and the number of minutes they would like to do this activity. The calculator will use this information to compute the number of calories that the user will burn while performing the specified activity.



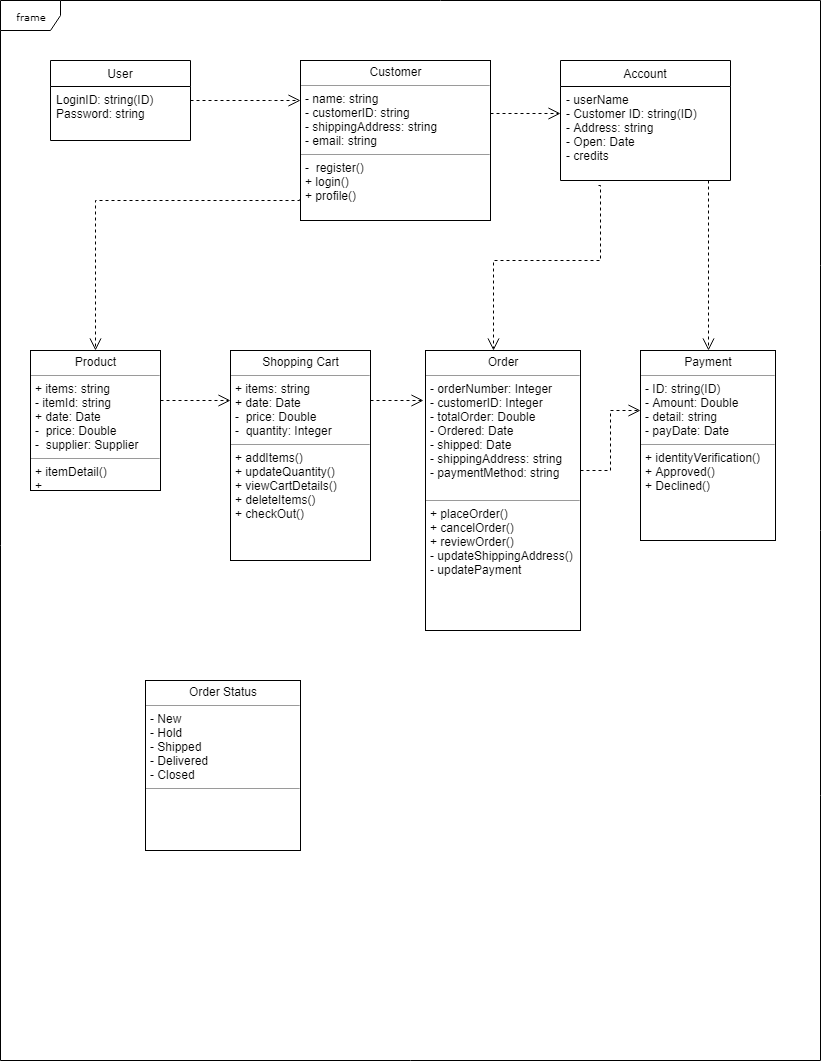
**E-commerce UML Use Case Diagram**



**E-commerce Structure Diagram**

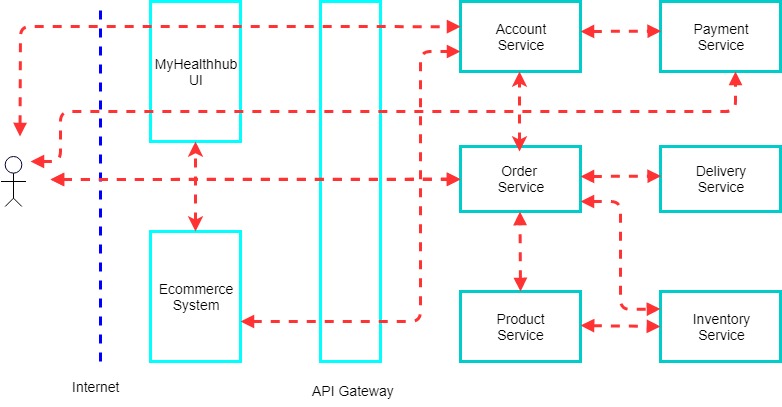


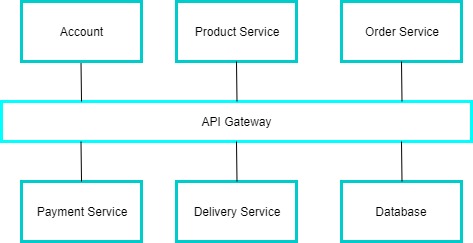
**E-commerce UML Class Diagram**



**High-level architecture of *Ecommerce***

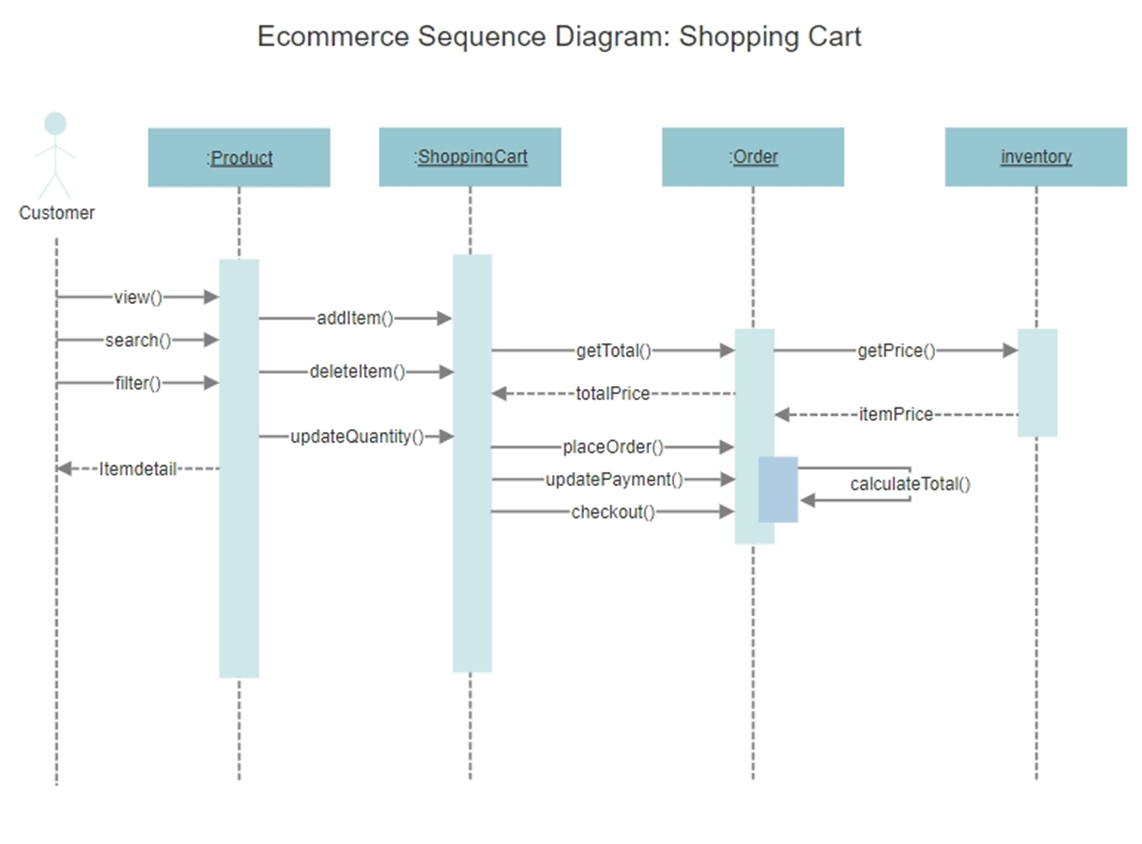
The high-level architectural design for Healthify software is shown below. The Healthify is composed of independent subsystems.



****

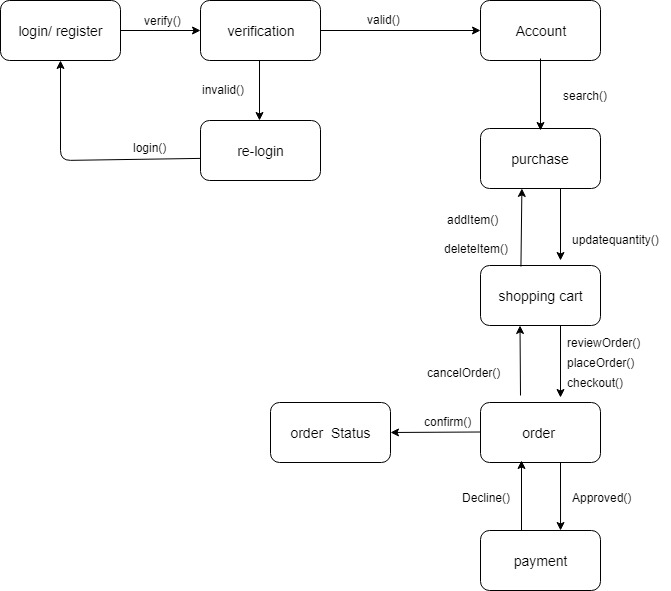
**Sequence diagram**

Sequence models are dynamic models that describe, for each mode of interaction, the sequence of object interactions that take place. This UML sequence diagram shows the sequence of interactions that take place between classes in Healthify platform. For example, when customers add products into shopping cart and request the total price, the class of order receives a request from shopping cart and then send a message to inventory class to get unit price of items. The inventory class return the price of items to the order class. The order class return the value of total price to shopping Cart.



**Ecommerce UML state diagram**

UML state diagram shows the possible state changes in the system. It also shows system states and events that cause transitions from one state to another state in the system. UML state diagram are used to model a system’s behavior in response to internal or external events. Ecommerce UML state diagram is shown below.



# Healthify Application User Flow

**Overall User Flow and Layers:**

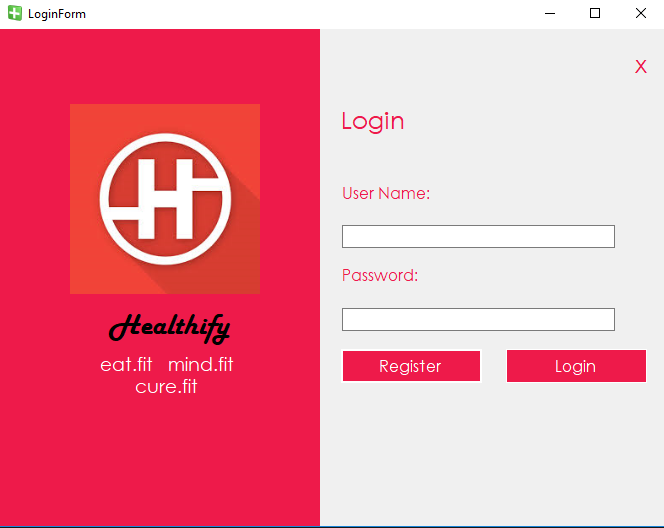
The following is an illustration of the applications overall layered process flow.



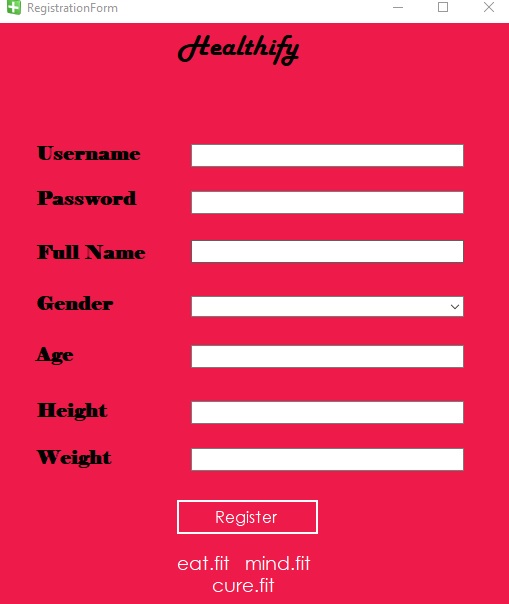
**System User Dialogs**

The following is an illustration of dialogs the user will encounter when running the Healthify user application.

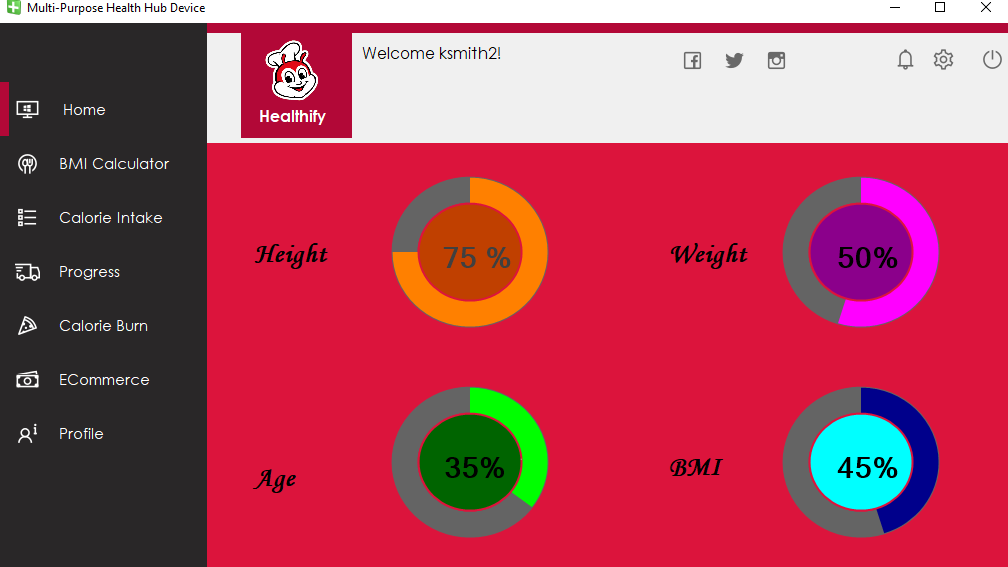
**1. Log In dialog:** (From this dialog the user can either Login or Register)



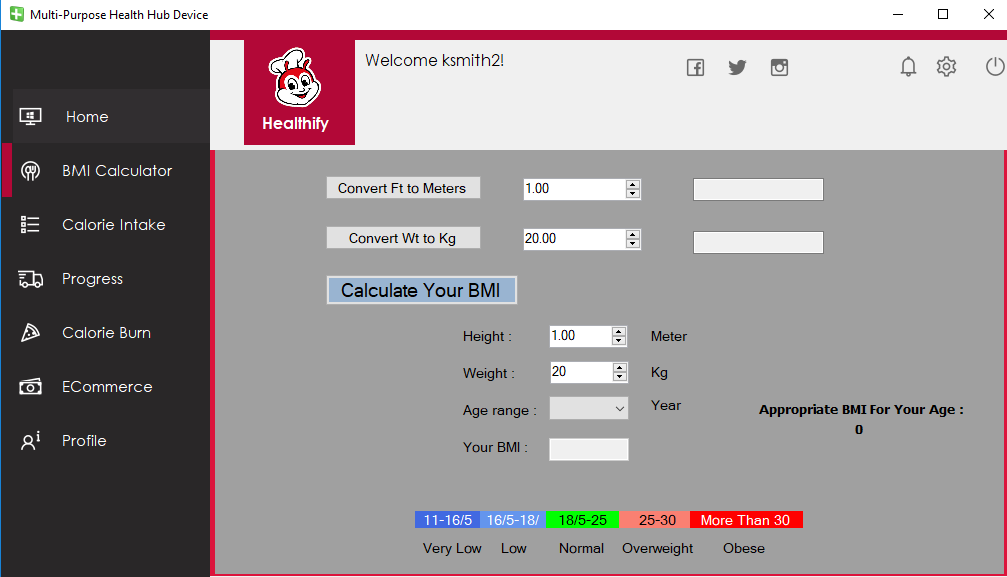
**2. Registration dialog:** (From this dialog new users are created)



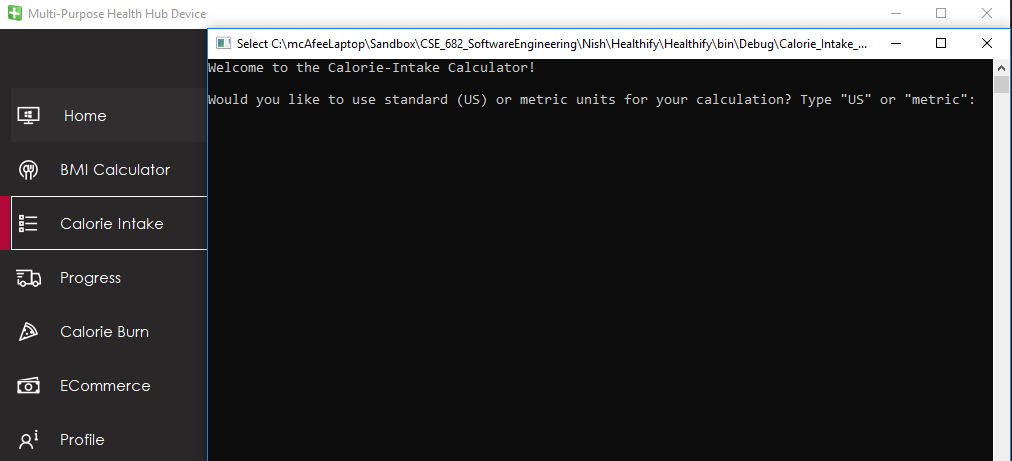
**3. Home Page dialog:** (From this dialog client applications are accessible)



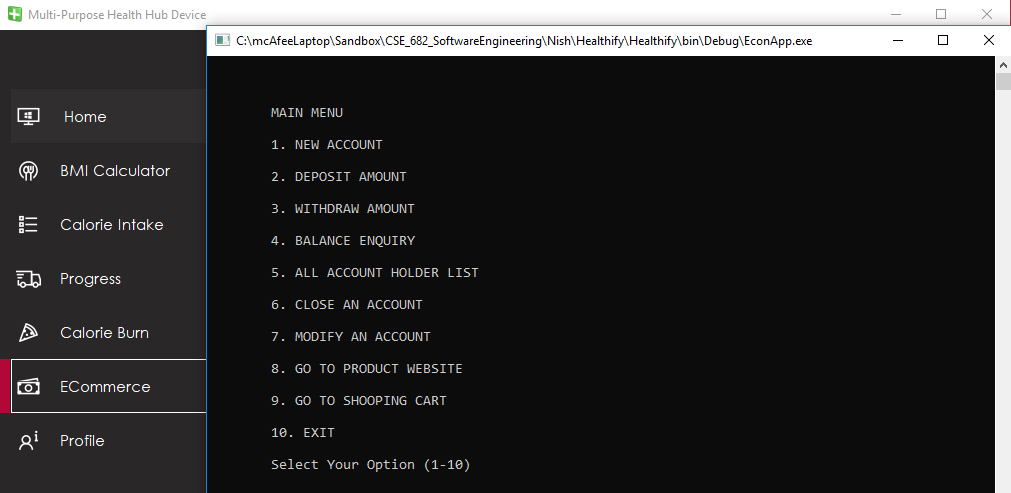
**4. BMI Calculator dialog:** (From this dialog a user’s BMI is calculated.)



**5. Calorie Calculator dialog:** (From this dialog a user’s calorie count is calculated.)

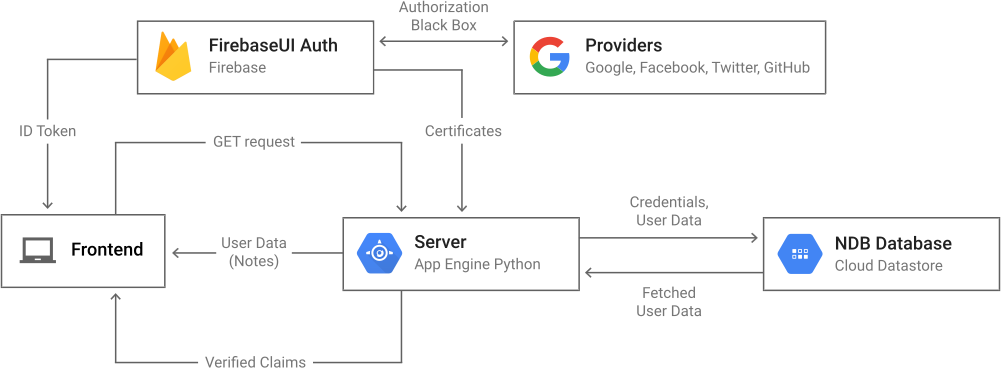


**6. E-Commerce dialog:** (From this dialog a user can access commerce options.)



# Firebase Database

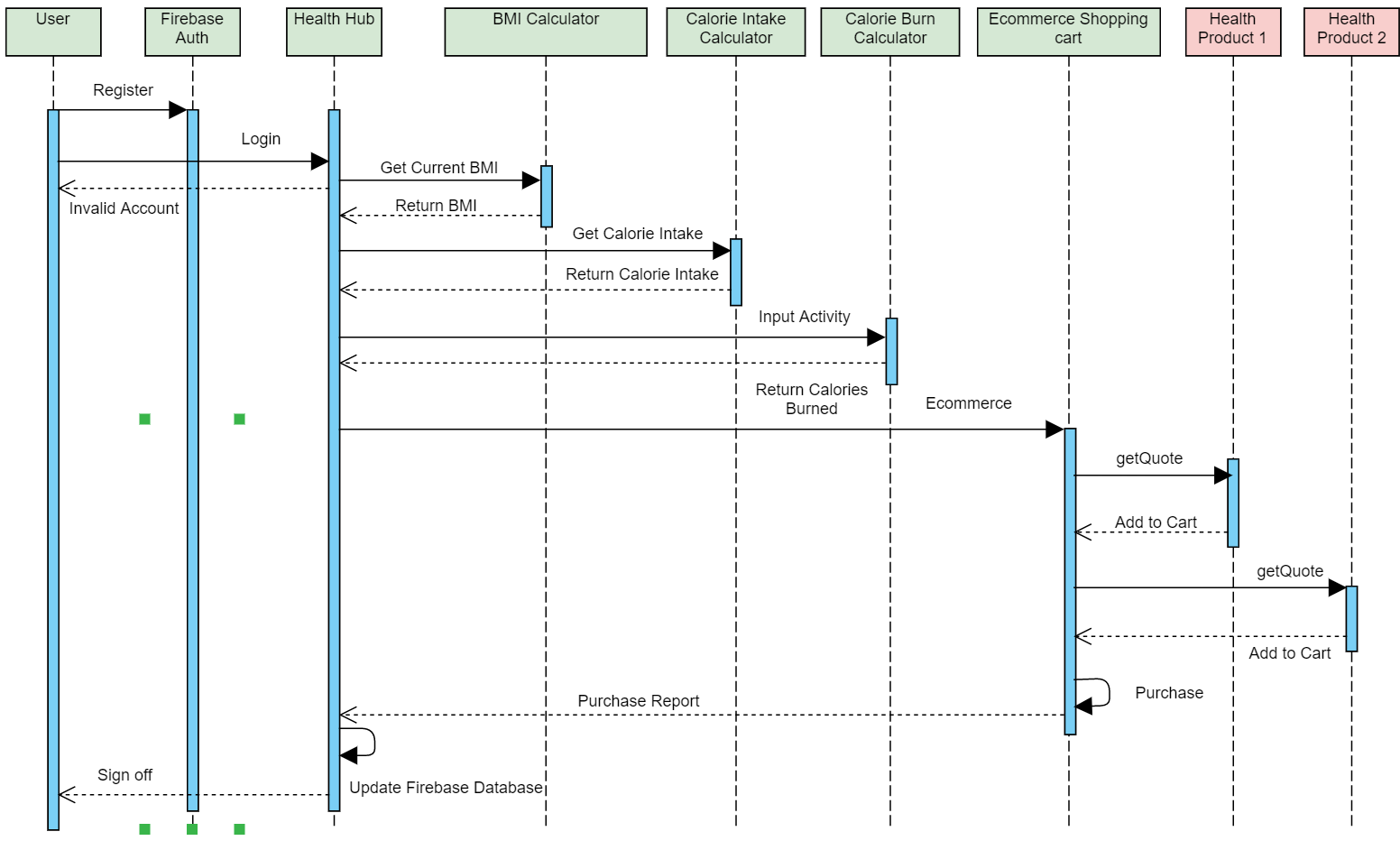
**Authentication:** The following is an illustration of the applications data flow



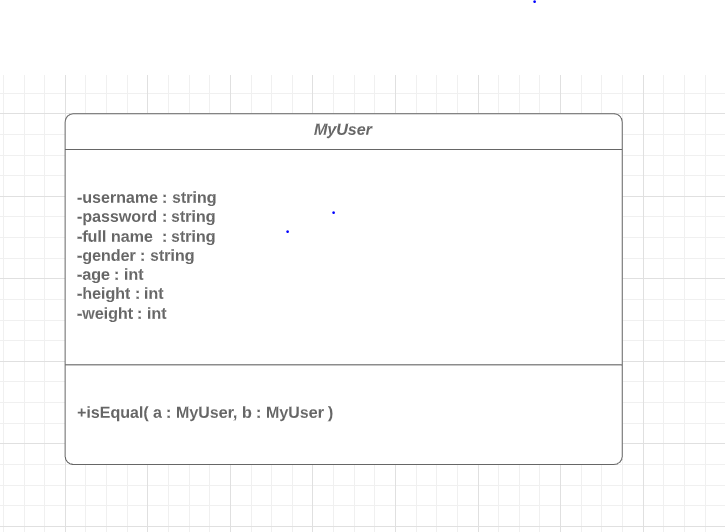
Authenticate users with their email addresses and passwords. The Firebase Authentication SDK provides methods to create and manage users that use their email addresses and passwords to sign in. The Firebase Realtime Database lets you build rich, collaborative applications by allowing secure access to the database directly from client-side code. Data is persisted locally, and even while offline, real-time events continue to fire, giving the end user a responsive experience. When the device regains connection, the Realtime Database synchronizes the local data changes with the remote updates that occurred while the client was offline, merging any conflicts automatically.

# Healthify Sequence Diagram

The following is an illustration of the sequence of user interactions of the various applications.



**MyUser class UML Class Diagram**(Central to the whole app)



# Source Repositories

**Contributors:**

<https://github.com/ksmith5627/CSE682-AppUI/commits/master>

<https://github.com/nish1ify/CSE-682-Project/graphs/contributors>

<https://github.com/jrthee94>

<https://github.com/kuohsun88?tab=repositories>