

FitLife App Analysis

Requirements & Use Cases

<div><<requirement>> Account Create</div> <div>Text = "New users can easily create an account" ID = "REQ001" source = "" kind = "Functional" verifyMethod =</div>	<div><<requirement>> Log In</div> <div>Text = "An existing users just needs to enter log in information to get started" ID = "REQ002" source = "" kind = "Functional" verifyMethod = "Analysis" risk = "Medium"</div>	<div><<requirement>> Log out</div> <div>Text = "When a user is done using the system they can logout or exit" ID = "REQ003" source = "" kind = "Functional"</div>	<div><<requirement>> Continuous Running</div> <div>Text = "If a user hasn't logged out or closed the program, it should remain open" ID = "REQ004"</div>	<div><<requirement>> Leaving</div> <div>Text = "Members have the option to delete their account" ID = "REQ005" source = "" kind = "Functional"</div>
<div><<requirement>> Data Purge</div> <div>Text = "When an account is deleted all associated data is deleted" ID = "REQ006" source = "" kind = "Functional"</div>	<div><<requirement>> Workout logging</div> <div>Text = "The system needs to store user's workout information" ID = "REQ007" source = "" kind = "Functional"</div>	<div><<requirement>> Workout Type</div> <div>Text = "Workouts need to be defined by type" ID = "REQ008" source = "" kind = "Functional" verifyMethod =</div>	<div><<requirement>> Editable Workouts</div> <div>Text = "Workouts must be editable by the user in case of mistake" ID = "REQ009" source = ""</div>	<div><<requirement>> Workout Deletion</div> <div>Text = "Workout are removable in case of mistakes" ID = "REQ010" source = "" kind = "Functional" verifyMethod =</div>
<div><<requirement>> Duration</div> <div>Text = "Workouts must include a duration" ID = "REQ011" source = "" kind = "Functional" verifyMethod =</div>	<div><<requirement>> Amount of Weight</div> <div>Text = "Amount of weight lifted during a workout can be added but is optional" ID = "REQ012"</div>	<div><<requirement>> Tracking Workouts</div> <div>Text = "Workouts progress needs to be trackable" ID = "REQ013" source = "" kind = "Functional"</div>	<div><<requirement>> Reading Workouts</div> <div>Text = "Workout progress needs to be easily readable" ID = "REQ014" source = "" kind = "Interface"</div>	<div><<requirement>> Progress time spans</div> <div>Text = "Progress can be viewed by week, month, and year" ID = "REQ015" source = ""</div>
<div><<requirement>> Meal logging</div> <div>Text = "The systems logs any meals entered by the user" ID = "REQ016" source = "" kind = "Functional"</div>	<div><<requirement>> Meals</div> <div>Text = "User can add meals" ID = "REQ017" source = "" kind = "Functional" verifyMethod = "Analysis"</div>	<div><<requirement>> Meal Description</div> <div>Text = "Meal has a description that is entered by the user" ID = "REQ018" source = "" kind = "Functional"</div>	<div><<requirement>> Macros</div> <div>Text = "Macros (calories, carbs, etc.) are captures by the system" ID = "REQ019" source = "" kind = "Functional"</div>	<div><<requirement>> Meal Editing</div> <div>Text = "A meal can be edited by a user in case of a mistake" ID = "REQ020" source = "" kind = "Functional"</div>
<div><<requirement>> Removing meals</div> <div>Text = "If a meal wasn't eaten or was entered improperly it can be removed" ID = "REQ021" source = ""</div>	<div><<requirement>> Hydration Logging</div> <div>Text = "System logs user's" ID = "REQ022" source = "" kind = "Functional" verifyMethod =</div>	<div><<requirement>> Hydration editing</div> <div>Text = "Hydration information is editable by the user is case of mistake" ID = "REQ023" source = "" kind = "Functional"</div>	<div><<requirement>> Forgot Password</div> <div>Text = "Users can reset their passwords if they have forgotten them" ID = "REQ024" source = ""</div>	<div><<requirement>> Sleep Logging</div> <div>Text = "The system logs a user's" ID = "REQ025" source = "" kind = "Functional" verifyMethod = "Analysis"</div>
<div><<requirement>> Sleep Times</div> <div>Text = "Sleep times are recorded by the system" ID = "REQ026" source = "" kind = "Functional" verifyMethod =</div>	<div><<requirement>> Sleep Rating</div> <div>Text = "User enters an overall sleep rating" ID = "REQ027" source = "" kind = "Functional" verifyMethod =</div>	<div><<requirement>> Weight Logging</div> <div>Text = "The system logs the user's weight" ID = "REQ028" source = "" kind = "Functional" verifyMethod =</div>	<div><<requirement>> Bodyweight</div> <div>Text = "Workouts require the user to enter bodyweight prior to the workout" ID = "REQ029" source = ""</div>	<div><<requirement>> BMI Prediction</div> <div>Text = "System provides BMI predictions based on past workouts" ID = "REQ030" source = "" kind = "Functional"</div>
<div><<requirement>> Statistics</div> <div>Text = "Exercise stats are tracked and logged for statistical analysis" ID = "REQ031" source = "" kind = "Functional"</div>	<div><<requirement>> Planning</div> <div>Text = "meals and workout can be preplanned and checked off when completed" ID = "REQ032" source = ""</div>	<div><<requirement>> Additional Info</div> <div>Text = "Additional information should exist in the system" ID = "REQ033" source = "" kind = "Functional" verifyMethod =</div>	<div><<requirement>> Habits Info</div> <div>Text = "Additional information includes proper weightlifting/workout t/eating/and sleeping habits" ID = "REQ034"</div>	<div><<requirement>> Accessible Info</div> <div>Text = "Safety information is easily accessible to the user" ID = "REQ035" source = "" kind = "Functional"</div>
<div><<requirement>> Goal Tips</div> <div>Text = "Additional information includes helpful tips to accomplishing workout/lifestyle goals" ID = "REQ036"</div>	<div><<requirement>> Information Storing</div> <div>Text = "System stores information to user's account" ID = "REQ037" source = "" kind = "Functional"</div>	<div><<requirement>> Color Scheme</div> <div>Text = "Program must have Blue and White color scheme" ID = "REQ038" source = "" kind = "Interface" verifyMethod = "Analysis"</div>	<div><<requirement>> Input Detected</div> <div>Text = "Program must visibly indicate when an input is detected" ID = "REQ039" source = "" kind = "Interface"</div>	<div><<requirement>> Back Button</div> <div>Text = "Must have a back button to return to the previous display" ID = "REQ040" source = "" kind = "Interface"</div>
<div><<requirement>> Password Reset Prompt</div> <div>Text = "User must provide password again to change their email or username" ID = "REQ041"</div>				

Use Case UC-ACCT-01 Create Account

ID: UC-ACCT-01

Scope: User Account

Level: User Goal

Stakeholders and Interests

Customer

- Wants fast and easy account creation
- Expects to give Username, email, and password

Admin

- Expects user info to be stored for mass-viewing

Pre-Conditions

User is not logged in

User is on the home screen

Post-Condition

An account has been created

Main Success Scenario

1. User selects the "Create an account" option
2. System displays input boxes asking for email, username, and password
3. User inputs the required field
4. System temporarily stores the data
5. System sends an email to the user's email to verify the email
6. User follows the link in their email to verify their email
7. System creates a new account with the data the user entered
8. System sends the user back to the home screen

Extensions

- 3a. Entered username is already linked to a different account
 1. System will display "Username already taken"
 2. System will return to the start of step three
- 3b. Entered Password does not fulfill the requirement
 1. System displays a message about what requirement was not fulfilled
 2. System returns to the start of step 3
- 6a. User does not verify their account within 48 hours
 1. Delete all temporarily stored data
- 6b. The email owner follows the link saying they did not sign up.
 1. Delete all data temporarily stored.

Use Case UC-ACCT-02 Delete Account

ID: UC-ACCT-02

Scope: User Account

Level: User Goal

Stakeholders and Interests

Customer

- Wants an easy process
- Wants their data to be removed

Pre-Conditions

User is logged in

User is at the home screen

Post-Condition

User's account has been deleted

Main Success Scenario

1. User navigates to the account settings
2. System displays details about user account
3. User selects "Delete Account" option
4. System asks to verify the user's choice
5. User confirms their intent
6. System logs the user out
7. System deletes the user's account

Extensions

- 1-4a. Anytime user visits another page
 1. Stop process
- 1-4b. Anytime user closes the application
 1. Stop Process
- 5a. User does not wish to delete account
 1. Return the system to it's state after step 2

Use Case UC-ACCT-03 Edit Account

ID: UC-ACCT-03

Scope: FitLife

Level: User Goal

Stakeholders and Interests

Customer

-Wants to be able to edit username, password, email, and avatar

Pre-Conditions

User is logged in

User is at home page

Post-Condition

User has edited desired detail

Main Success Scenario

1. User navigates to their account settings
2. System displays all account info
3. User selects desired setting to change
4. System handles the change
5. System stores the changed setting

Extensions

4a. User wishes to edit their email

1. User enters their password to validate identity

1a. User enters wrong password

1. System displays an error message and asks user to re-enter their password

2. System returns to step 4a-1

2. User enters the new email address

3. System removes the email's association with the account

3a. There are no more accounts associated with that email

1. System removes that email from the database

4. System associates the new email with the account

4a. The email is not in the database

1. The new email is added to the database

4b. User wishes to edit their password

1. System requests that the user enter their old password, and their new password twice

1a. Old password does not match current password

1. System displays an error message and returns to step 4b-1

1b. The entries of the new password do not match

1. System displays an error message and returns to step 4b-1
- 1c. The user's new password does not meet the password requirements
 1. System displays an error message and returns to step 4b-1
- 1d. The user's new password is the same as the user's old password
 1. System displays an error message and returns to step 4b-1
 2. System sets the user's new password
- 4c. User wishes to edit their username
 1. User enters their password to validate identity
 - 1a. User enters wrong password
 1. System displays an error message and asks user to re-enter their password
 2. System returns to step 4c-1
 2. User enters a the new username
 - 2a. Entered username is already taken
 1. System displays an error message and returns to step 4c-2
 3. System removes the old username's association with the account
 4. System associates the new username with the account
 - 4d. User wishes to edit their avatar
 1. System requests the user to upload a file
 2. User uploads a file
 - 2a. Image file is an incorrect ratio
 1. System displays the image, with a box representing the appropriate image ratio
 2. System crops the image
 3. System deletes the account's previous avatar
 4. System sets the account's image to the uploaded avatar

Use Case UC-ACCT-04 View Info

ID: UC-ACCT-04

Scope: FitLife

Level: User Goal

Stakeholders and Interests

Customer

-Wants to be able to easily view info about a specific subject

Pre-Conditions

User is logged in

User is at the home screen

Post-Condition

User is viewing the information about a health subject

Main Success Scenario

1. User selects the info screen from the home page
2. System displays the main page of the information section
3. User selects the topic he wishes to know more about
4. System displays information about the topic, as well as any subtopics that might exist
5. User Selects a sub topic to view
6. System displays information about the subtopic

Extensions

- a. User wishes to stop viewing information
 1. User may navigate to a different page
 2. User may close the application
 3. User may log out

Use Case UC-ACCT-05 Reset Password

ID: UC-ACCT-05

Scope: FitLife

Level: User Goal

Stakeholders and Interests

Customer

- Wants to be able to reset their password if forgotten
- Expects password reset to utilize their email

Pre-Conditions

User is not logged in
User is at the sign in screen

Post-Condition

User has reset their password

Main Success Scenario

1. User presses the "Forgot Password" button
2. System asks for the user's email or username
3. System creates a temporary password that can be used to sign in to the account
4. System sends a message to the associated account's email address that contains a temporary password
5. User signs in using their temporary password
6. System asks the user to input a new password twice
7. User inputs a new password twice
8. System changes the account's password to the new password
9. System removes the temp password

Extensions

- 2a. User enters a username or email that is not associated with an account
 1. System sends an error message to the user and returns to the beginning of step 2
- 3-4a. User signs in using their old password
 1. System deletes the temporary password so it can't be used
- 3a. Temporary password already associated with the account
 1. Delete the old temporary password and replace it with the new one
- 5a. User has not signed in within 24 hours
 1. System deletes the temporary password so it can't be used
- 7a. Input password does not match
 1. Display an error message and return to step 7
- 7b. Passwords do not meet password requirements
 1. Display an error message and return to step 7

- 7c. Passwords entered are the same as the account's current password
1. Display an error message and return to step 7

Use Case UC-ACCT-06: User wants to log-in

Scope: FitLife

Level: user goal

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User:

Wants accurate, fast entry, as registered user entered their correct account information with their corresponding password.

- Database:

Wants to receive correct login account information from user.

Preconditions:

User has registered an account and user information has identified.

Success Guarantee (or Postconditions):

User account information and password are correct. User can access into their account.

Main Success Scenario (or Basic Flow):

1. User open Its Fit application getting into the Log in page.
2. User entered their user name and corresponding password.
3. System recognized the user account and let user get into their main page.

Extensions (or Alternative Flows):

- 2a. User entered wrong account information of either username or password.
 1. System signals error to the User, records the error, and enters a clean state.
 2. System ask user to enter their log in information again.

Use Case UC-PLAN-01: Plan Workout

Scope: FitLife

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants to have a simple and fast interaction with the system.
Wants to have the exact information stored in the system.
Wants to be able to plan a workout quickly.

Precondition: User is registered and has successfully logged in. User has added a workout.

Postcondition: Workout information is saved. Information is updated. Database is updated.

Main Success Scenario:

1. Registered user arrives at home screen
2. User chooses an existing workout
3. System displays information of the selected workout
4. User selects a workout
5. User clicks "Plan Workout" button
6. User repeats steps 2-4 until completed
7. System displays a series of choices for the user to enter, including:
the Duration of workout, the Date and Time of workout, and the
additional Note for the workout
8. User selects a category and enters information
9. Information is updated and stored in database
10. User is able to review workout before logout
11. User logs out

Extensions:

- *a. At any time, Systems fails:

1. User restarts the system
2. User logs in
3. User repeats the process of planning a workout

8.a User selected the wrong choice before saving

1. User clicks "cancel" to go to the previous page
2. User repeats the process
3. User clicks "Save" to save the information
4. Information is updated and stored in the database

2-9.a User decides to change information after saving

1. User clicks "Plan Workout" button
2. System displays a series of choices for the user to enter, including:
the Duration of workout, the Date and Time of workout, and the
additional Note for the workout
3. User selects and enters desired information
4. User clicks "Save" to save
5. System stores information
6. Information is updated and stored into the database

Use Case UC-PLAN-02: Plan Meal

Scope: FitLife

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants to have a simple and fast interaction with the system. Wants to have the exact information stored in the system. Wants to be able to plan a meal quickly and easily.

Precondition: User is registered and has successfully logged in.

Postcondition: Meal is added to specified date selected. Information is updated. Database updated.

Main Success Scenario:

1. Registered user arrives at home screen.
2. User selects Nutrition on home screen.
3. User chooses a date from the list.
4. System displays info about meal such as protein, carbohydrates, natural fats, and calories, as well as how much water.
5. User enters information about their meal.
6. User selects to add meal to that day.
7. Information is updated and stored in database.
8. User repeats steps 2-7 until finished.
9. User logs out and exits the program.

Extensions:

*a. System fails at any given time:

1. User restarts system
2. User logs in
3. User repeats process of planning their meal.

*b. User navigates to another page during planning:

1. End process.
2. No new information saved.

3.a. User selects wrong date:

1. User simply clicks on their preferred date.
2. Re-Enter meal information.

8.a. User wants to delete a meal:

1. User selects a date.
2. User chooses meal from the entered meals on that day.
3. User selects to delete on the meal they would like removed.
4. Meal is removed from that days meals.

Use Case UC-Plan-03: Plan Sleep

Scope: FitLife application

Level: User Goal

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants a simple interaction with the system. Wants to future sleep. Wants to prefill the time, date, and duration for easier completion later.

Precondition: User is registered and has successfully logged in.

Postcondition: Future sleep is logged and waiting for user to complete

Main Success Scenario:

1. Registered User arrives at home screen
 2. User selects "Plan" from the side menu
 3. System displays the options for planning
 4. User selects "Sleep" from the planning menu
 5. System prompts user from time, duration, and date
 6. User enters require information and clicks the "Submit" button
 7. System takes User to the Plan Sleep page
- Repeats steps 5-7 until all desired sleep has been planned
8. User logs out and exits program

Extensions

*a. at anytime User navigates to a different page

1. End Process

*b. at anytime the system fails

1. User restarts the System, logs in, and starts Scenario over again

7a. User wants to delete planned sleep

1. User selects the "Delete Entry" option
2. The system displays the log of planned sleeps
3. The user selects the sleep(s) to be deleted
4. The system asks the user if he/she wants to delete the sleep(s) for good
5. The user replies yes and is taken back to the Sleep Plan page

Use Case UC-REPORT-01: Add Workout

Scope: FitLife application

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants to have a simple and fast interaction with the system.
Wants to add the correct information.
Wants to be able to add a workout quickly.

Precondition: User is registered and has successfully logged in.

Postcondition: Workout information is saved. Information is updated. Database is updated.

Main Success Scenario:

1. Registered user arrives at home screen
2. User selects "Add Workout" from the side menu
3. System displays a selection of different kinds of workout, and prompts for user's choice
4. User selects a workout
5. User clicks "Save" button
6. User repeats steps 2-5 until completed
7. Systems captures the information
8. Information is updated and stored in database
9. User is able to plan and review workout before logout
10. User logs out

Extensions:

*a. At any time, Systems fails:

1. User restarts the system

2. User logs in
3. User repeats the process of adding a workout

2-5.a User entered undesired workout

1. User clicks "Delete Workout" button
2. System displays a confirmation message to reassure the action
3. User clicks "Yes" to delete the selected workout, or "Cancel" to cancel the action
4. System stores information
5. Information is updated and stored into the database

2-5.b User decided to change workout

1. User clicks "Delete Workout" button
2. System displays a confirmation message to reassure the action
3. User clicks "Yes" to delete the selected workout, or "Cancel" to cancel the action
4. System stores information

Use Case UC-REPORT-02: Add Sleep

Scope: FitLife application

Level: User Goal

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants a simple interaction with the system. Wants to accurately record sleep quality and time.

Wants to enter information quickly.

Precondition: User is registered and has successfully logged in.

Postcondition: Sleep information is saved. Statistics are updated.

Main Success Scenario:

1. Registered User arrives at home screen
2. User selects "Add Sleep" from the side menu
3. System prompts user for required sleep information
4. User enters quality and duration of sleep
User repeats steps 2-4 until completed
5. System captures the information
6. Database stores the information
7. Statistic Library generates and updates statistic
8. User reviews statistics and logs out when complete

Extensions

*a. at anytime User navigates to a different page

1. End Process

*b. at anytime the system fails

1. User restarts the System, logs in, and starts Scenario over again

8a. User wants to remove entry

1. User selects the "Delete Entry" option

2. The system displays the log of sleeps

3. The user selects the sleep(s) to be deleted

4. The system asks the user if he/she wants to delete the sleep(s) for good

5. The user replies yes and is taken back to the Sleep page

Use Case: UC-REPORT-03: Add Meal.

Scope: FitLife Application.

Level: User Goal

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants a simple interaction with the system.
Wants to accurately be able to enter meal information.
Wants to enter information quickly.

Precondition: User is registered and successfully logged in.

Postcondition: Meal information is saved.

Main Success Scenario:

1. Registered user arrives at home screen.
2. User selects Nutrition from home screen.
3. System displays Nutrition screen.
4. User selects date.
5. System displays form for user to fill out information about meal with criteria such as protein, carbohydrates, natural fats, and others.
6. User fills out criteria with meal information.
7. User selects to add meal to selected date.
8. System captures information.
9. Information stored in database.
10. User logs out and exits application.

Extensions:

- *a. User tries to enter meal without selecting date:
 1. System prompts user to choose a date.

2. User selects date.

3. User re-enters meal information.

4. User adds meal to that date.

*b. At any time the system fails:

1. User restarts the System, logs in, and starts Scenario over again.

*c. User navigates to a different page before adding meal:

1. All unsaved info is lost.

2. User must navigate back and re-enter information.

8.a. User wants to remove entered information:

1. User selects date.

2. System displays all previously entered meals on that date.

3. User chooses delete on the meal to be deleted.

4. Meal is deleted and removed from display.

5. Meal removed from database.

6. Database updated.

Use Case UC-REPORT-04: Add Hydration

Scope: FitLife application

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants to have a simple and fast interaction with the system.
Wants to add the correct information.
Wants to be able to add a hydration quickly.

Precondition: User is registered and has successfully logged in.

Postcondition: Hydration information is saved. Information is updated. Database is updated.

Main Success Scenario:

1. Registered user arrives at home screen
2. User selects "Add Hydration" from the side menu
3. System displays a selection of different types of beverage, and prompts for user input
4. User select the type of beverage and enter the amount
5. User clicks "Save" button
6. User repeats steps 2-4 until completed
7. Systems captures the information
8. Information is updated and stored in database
9. User reviews information and logout

Extensions:

*a. At any time, Systems fails:

1. User restarts the system
2. User logs in
3. User repeats the process of adding a hydration

*b. User entered incorreccted information

1. User clicks "Edit Hydration" button
2. User enters desired information
3. User saves

*c. User decided to change hydration

1. User clicks "Delete hydration" button
2. System displays a confirmation message to reassure the action
3. User clicks "Yes" to delete the selected hydration, or "Cancel" to cancel the action
4. System stores information
5. Information is updated and stored into the database

Use Case UC-REVIEW-01: Review Workout

Scope: FitLife application

Level: User Goal

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants a simple interaction with the system.
 - Wants to view added workouts.
 - Wants to be able to review a workout.
 - Wants to be able to filter by week, month, and year.
 - Wants to be able to view the total workouts within a week, a month, or a year.

Precondition: User is registered and has successfully logged in.
User has added at least one workout

Postcondition: workout information and trends are displayed to User

Main Success Scenario:

1. Registered User arrives at home screen
2. User selects "Review Workout" from the side menu
3. System displays the trend options
4. User selects "Workout" from the trends menu
5. System displays workout trends in graphical format
6. User reviews workout

Extensions:

*a. at anytime User navigates to a different page

1. End Process

*b. at anytime the system fails

1. User restarts the System
2. user logs in, and starts Scenario over again

6.a Weekly Trends

1. User selects "Weekly" option
2. System displays the modified graph of trends for the current week
3. User reviews the weekly trends

6.b Monthly Trends

1. User selects "Monthly" option
2. System displays the modified graph of trends for the current month
3. User reviews the "Monthly" trends

6.c Yearly Trends

1. User selects "Yearly" option
2. System displays the modified graph of trends for the current year
3. User reviews the "Yearly" trends

Use Case UC-REVIEW-02: Review Sleep

Scope: FitLife application

Level: User Goal

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User:

Wants a simple interaction with the system. Wants to view previous sleep logs.
Wants to be able to review a sleep log. Wants to view aggregate data in graph format. Wants to be able to filter by week, month, and year.

Precondition: User is registered and has successfully logged in.

Postcondition: Sleep logs and trends are displayed to User

Main Success Scenario:

1. Registered User arrives at home screen
2. User selects "Review Trends" from the side menu
3. System displays the trend options
4. User selects "Sleep" from the trends menu
5. System displays Sleep trends in graphical format
6. User reviews trends

Extensions

*a. at anytime User navigates to a different page

1. End Process

*b. at anytime the system fails

1. User restarts the System, logs in, and starts Scenario over again
- 2-6a. User selects a specific sleep log item
 1. System displays the stats for the specified log item

2. User reviews sleep information

6.a Weekly Trends

1. User selects "Weekly" option
2. System displays the modified graph of trends for the current week
3. User reviews the weekly trends

6.b Monthly Trends

1. User selects "Monthly" option
2. System displays the modified graph of trends for the current month
3. User reviews the "Monthly" trends

6.c Yearly Trends

1. User selects "Yearly" option
2. System displays the modified graph of trends for the current year
3. User reviews the "Yearly" trends

Use Case UC-REVIEW-03: Review Diet

Scope: FitLife Application

Level: User Goal

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants a simple interaction with the system.
 - Wants to view diet history.
 - Wants to be able to review diet.
 - Wants to be able to filter by day, month, and year.

Precondition: User is registered and has successfully logged in.

Postcondition: User is able to view diet statistics and trends.

Main Success Scenario:

1. Registered user arrives at home screen.
2. User selects "Review Nutrition" from home screen.
3. System displays calendar.
4. User selects a specific day, week, month, or year from the calendar.
5. Diet trends and information are displayed based on selected timeframe.
6. User reviews diet.
7. User repeats steps 4-6 until finished.

Extensions:

*a. At any time, system fails:

1. User restarts system.
2. User logs in.
3. User begins scenario again.

*b. At any time user navigates to new page:

1. End process.

- 4.a. User selects a specific day:

1. Diet trends and information adjusted for that selected day.
2. User reviews trends and information for that day.

- 4.b. User selects a specific week:

1. Diet trends and information adjusted for that selected week.
2. User reviews trends and information for that week.

- 4.c. User selects a specific month:

1. Diet trends and information adjusted for that selected month.
2. User reviews trends and information for that month.

- 4.d. User selects a specific year:

1. Diet trends and information adjusted for that selected year.
2. User reviews trends and information for that year.

Use Case UC-REVIEW-04: Review Hydration

Scope: FitLife application

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User: Wants a simple interaction with the system.
 - Wants to be able to view hydration history.
 - Wants to be able to review hydration history.
 - Wants to be able to filter by day, week, month, and year.

Precondition: User is registered and has successfully logged in.

Postcondition: Hydration information and trends are displayed to user.

Main Success Scenario:

1. Registered user arrives at home screen.
2. User selects "Review Nutrition" from home screen.
3. System displays calendar.
4. User selects a specific day, week, month, or year from the calendar.
5. Hydration and diet trends and information are displayed based on selected timeframe.
6. User reviews hydration and diet.
7. User repeats steps 4-6 until finished.

Extensions:

*a. At any time, system fails:

1. User restarts system.
2. User logs in.
3. User begins scenario again.

*b. At any time user navigates to new page:

1. End process.

4.a. User selects a specific day:

1. Hydration and diet trends and information adjusted for that selected day.
2. User reviews trends and information for that day.

4.b. User selects a specific week:

1. Hydration and diet trends and information adjusted for that selected week.
2. User reviews trends and information for that week.

4.c. User selects a specific month:

1. Hydration and diet trends and information adjusted for that selected month.
2. User reviews trends and information for that month.

4.d. User selects a specific year:

1. Hydration and diet trends and information adjusted for that selected year.
1. User reviews trends and information for that year.

Use Case UC-Review-05: Review BMI Trends

Scope: FitLife Application

Level: user goal

Primary Actor: Registered User

Stakeholders and Interests:

- Registered User:

Wants to have a simple and fast interaction with the system, able to keep track on the BMI trends, calculate and update the current BMI.

- Database:

Wants to receive correct (reasonable) information of weight and height from user.

Preconditions:

User has logged into their account and user information has identified.

Success Guarantee (or Postconditions):

User can update their current BMI and get the BMI trend.

Main Success Scenario (or Basic Flow):

1. User select BMI Trends from the user main page.
2. User entered their height and weight into the System.
3. System calculate the user's current BMI and update the BMI trends.
4. System shown the BMI trends to User.

Extensions (or Alternative Flows):

*a. At anytime User navigates to a different page

1. End Process

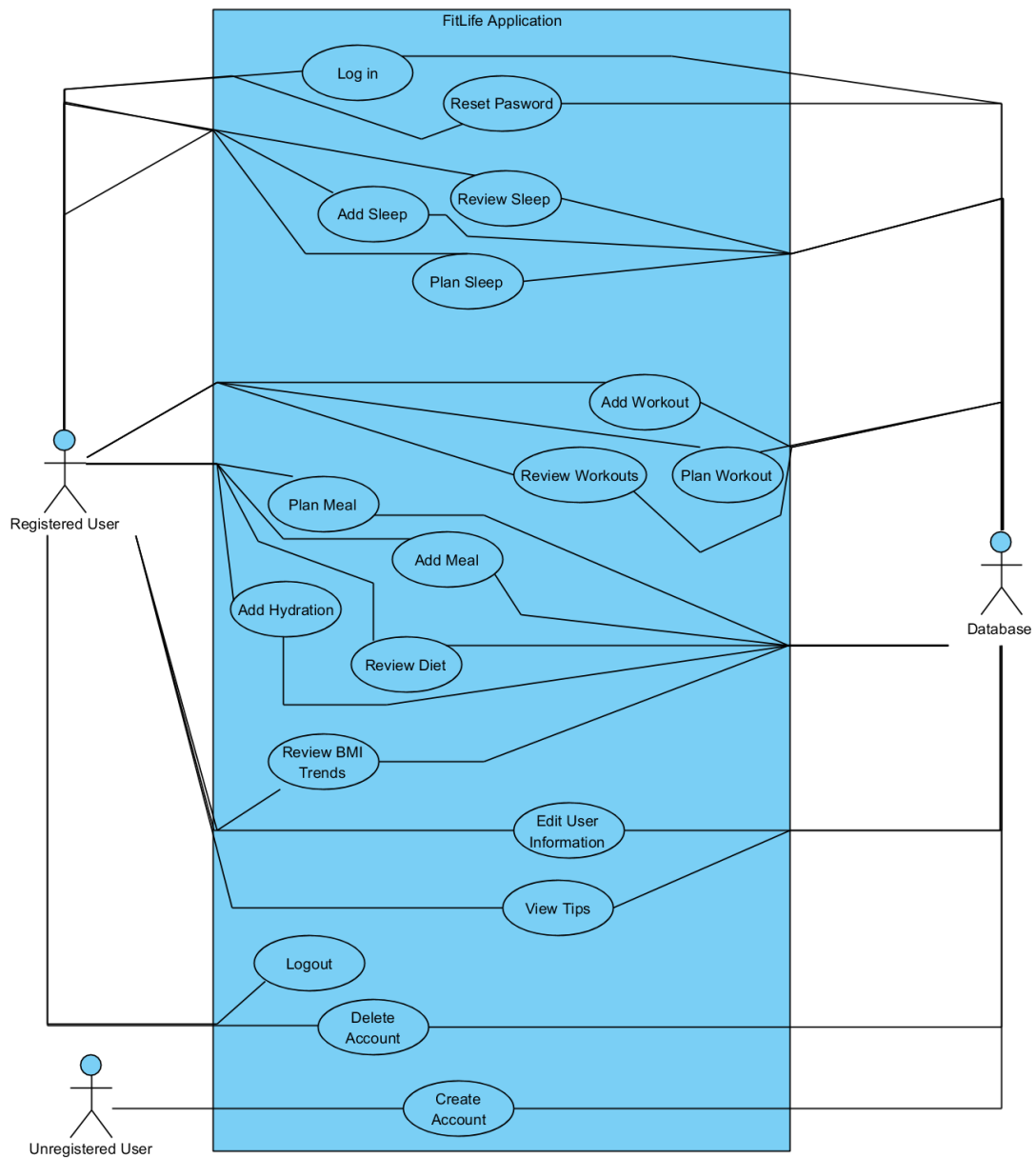
*b. Anytime user closes the application

1. End Process

2a. User entered unreasonable input into System.

1. System signals error to the User, records the error, and enters a clean state.
2. System ask user to enter their information again.

Use Case Diagram

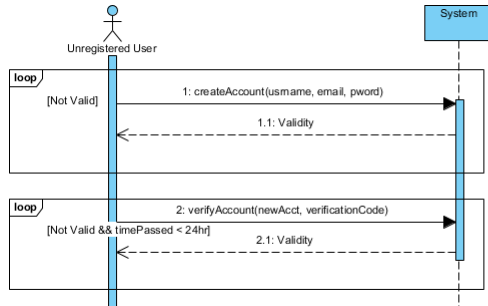


Traceability Matrix

Accessible Info	REQ035	Functional	Low	View Tips
Account Create	REQ001	Functional	Medium	Create Account
Additional Info	REQ033	Functional	Low	View Tips
Amount of Weight	REQ012	Functional	Low	Add Workout, Plan Workout
Back Button	REQ040	Interface	Low	Add Meal, Add Sleep, Add Workout, Create Account, Delete Account, Edit User Information, Logout, Plan Meal, Plan Sleep, Plan Workout, Reset Password, Review BMI Trends, Review Diet, Review Sleep, Review Workouts, View Tips
BMI Prediction	REQ030	Functional	Low	Review BMI Trends
Bodyweight	REQ029	Functional	Low	Add Workout, Review BMI Trends, Review Workouts
Color Scheme	REQ038	Interface	Low	Add Meal, Add Sleep, Add Workout, Create Account, Delete Account, Edit User Information, Log in, Logout, Plan Meal, Plan Sleep, Plan Workout, Reset Password, Review BMI Trends, Review Diet, Review Sleep, Review Workouts, View Tips
Continuous Running	REQ004	Functional	Medium	Reset Password
Data Purge	REQ006	Functional	Medium	Delete Account
Duration	REQ011	Functional	Low	Add Workout, Plan Workout
Editable Workouts	REQ009	Functional	Low	Add Workout, Plan Workout
Forgot Password	REQ024	Functional	Medium	Log in, Reset Password
Goal Tips	REQ036	Functional	Low	View Tips
Habits Info	REQ034	Functional	Low	View Tips
Hydration editing	REQ023	Functional	Low	Add Hydration
Hydration Logging	REQ022	Functional	Low	Add Hydration
Information Storing	REQ037	Functional	Low	Create Account, Edit User Information
Input Detected	REQ039	Interface	Low	Add Meal, Add Sleep, Add Workout, Create Account, Edit User Information, Log in, Plan Meal, Plan Sleep, Plan Workout, Reset Password, Review BMI Trends,
Leaving	REQ005	Functional	Medium	Delete Account
Log In	REQ002	Functional	Medium	Log in
Log out	REQ003	Functional	Medium	Logout
Macros	REQ019	Functional	Low	Add Meal, Plan Meal, Review Diet
Meal Description	REQ018	Functional	Low	Add Meal, Plan Meal, Review Diet
Meal Editing	REQ020	Functional	Low	Add Meal, Plan Meal, Review Diet
Meal logging	REQ016	Functional	Low	Add Meal, Plan Meal, Review Diet
Meals	REQ017	Functional	Low	Add Meal, Plan Meal, Review Diet
Password Reset Prompt	REQ041	Interface	Low	Edit User Information, Log in, Reset Password
Planning	REQ032	Functional	Low	Plan Meal, Plan Sleep, Plan Workout
Progress time spans	REQ015	Interface	Low	Review BMI Trends, Review Diet, Review Sleep, Review Workouts
Reading Workouts	REQ014	Interface	Low	Review Workouts
Removing meals	REQ021	Functional	Low	Plan Meal
Sleep Logging	REQ025	Functional	Low	Add Sleep, Plan Sleep, Review Sleep
Sleep Rating	REQ027	Functional	Low	Add Sleep, Plan Sleep, Review Sleep
Sleep Times	REQ026	Functional	Low	Add Sleep, Plan Sleep, Review Sleep
Statistics	REQ031	Functional	Low	Add Workout, Review Diet, Review Sleep, Review Workouts
Tracking Workouts	REQ013	Functional	Low	Add Workout, Review Workouts
Weight Logging	REQ028	Functional	Low	Add Workout
Workout Deletion	REQ010	Functional	Low	Plan Workout
Workout logging	REQ007	Functional	Low	Add Workout, Plan Workout, Review Workouts
Workout Type	REQ008	Functional	Low	Add Workout, Plan Workout

System Sequence Diagrams & Operation Contracts

Create Account

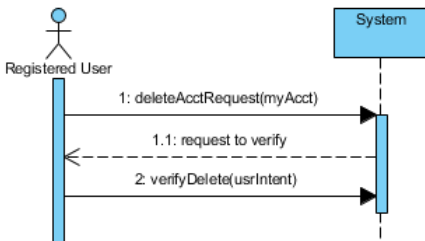


Contract: create account
Operation: createAccount(username: String, email: String, pwd: String)
Pre-Condition: User is not logged in
Post-Condition: Account is created if it is given acceptable parameters, else it returns not valid

Contract: create account
Operation: verifyAccount(newAcct: Account, verificationCode: String)
Pre-Condition: newAcct exists, and is unverified
Post-Condition: newAcct gets verified, if given correct verification code, else returns invalid

Powered By Visual Paradigm Community Edition

Delete Account

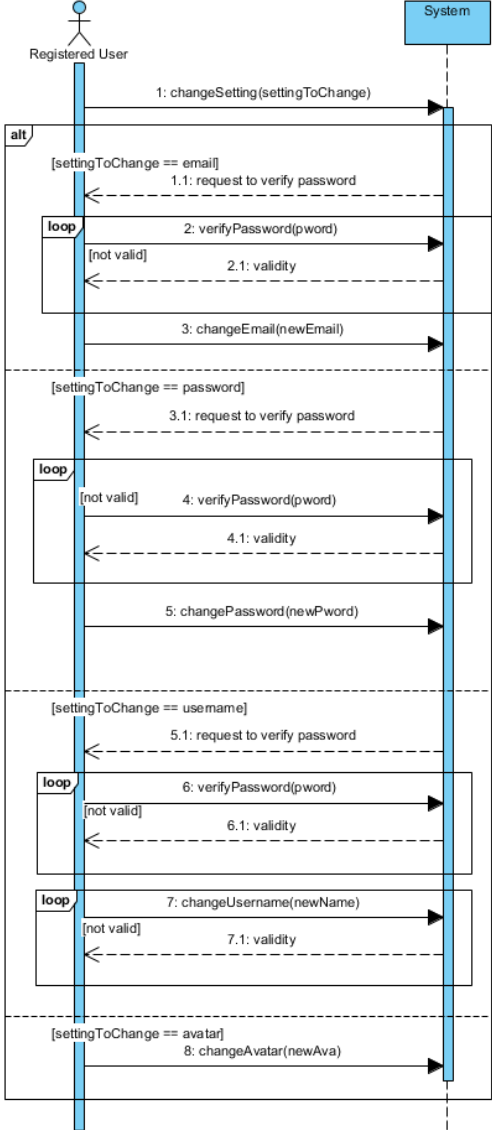


Contract: deleteAcct
Operation: deleteAcctRequest(myAcct: Account)
Pre-Condition: User is logged in
Post-Condition: Request has been sent for user to verify their wishes to delete their account

Contract: delete account
Operation: verifyDelete(usIntent: bool)
Pre-Condition: User is logged in, and wishes to delete account
Post-Condition: User is logged out, user's account has been deleted

Powered By Visual Paradigm Community Edition

Edit Account



Contract: edit account
Operation: changeSetting(settingToChange: String)
Pre-Condition: User is logged in
Post-Condition: User's account has been changed

Contract: edit account
Operation: verifyPassword(pword: String)
Pre-Condition: User has not verified password
Post-Condition: User has verified that they are the account's owner

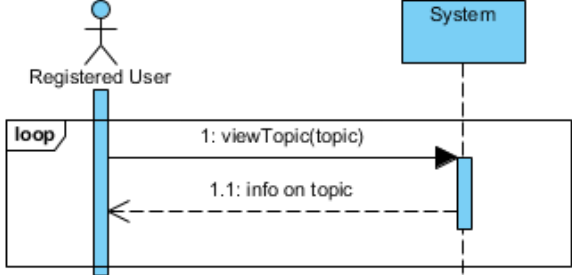
Contract: edit account
Operation: changePassword(newPword: String)
Pre-Condition: User has verified password
Post-Condition: Account's password has been changed

Contract: edit account
Operation: changeUsername(newName: String)
Pre-Condition: User has verified that they are the account's owner
Post-Condition: User has changed their username if newName is a valid username

Contract: edit account
Operation: changeAvatar(newAva: Image)
Pre-Condition: User is logged in
Post-Condition: Account's avatar has been changed

Powered By Visual Paradigm Community Edition

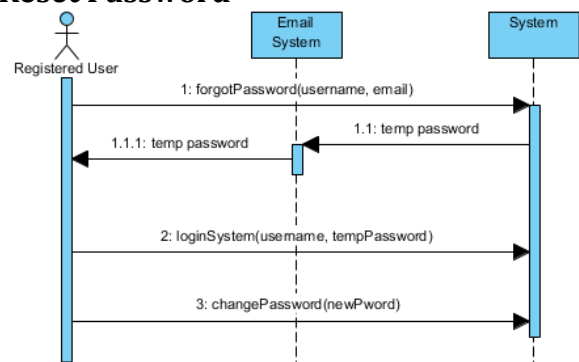
View Info



Contract: view info
Operation: viewTopic(topic: String)
Pre-Condition: User wishes to view info on topic
Post-Condition: User has viewed info on topic

Powered By Visual Paradigm Community Edition

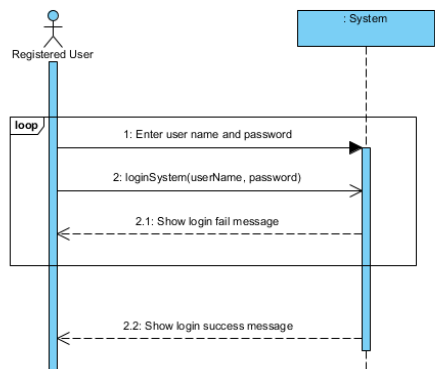
Reset Password



Contract: forgot password
Operation: forgotPassword(username: String, email: String)
Pre-Condition: User not logged in
Post-Condition: An email has been sent to the user's email address, a temporary password has been given to the user's account

Powered By Visual Paradigm Community Edition

Log In



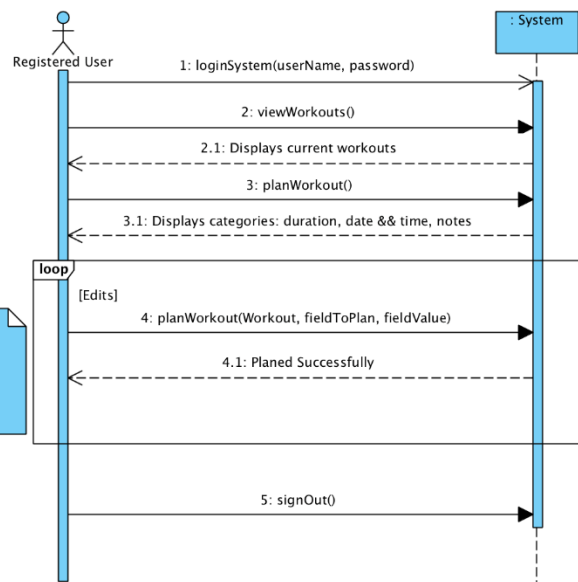
Contract CO18: log in into system
Operation: loginSystem(userName: string, password: string)
Pre-conditions: User has registered
Post-conditions:
User.userName will set to userName. (attribute modification)
User.password will set to password. (attribute modification)
The System will return a success message if User.userName is match to User.password.
The System will return a fail message if User.userName does not match to User.password.

Plan Workout

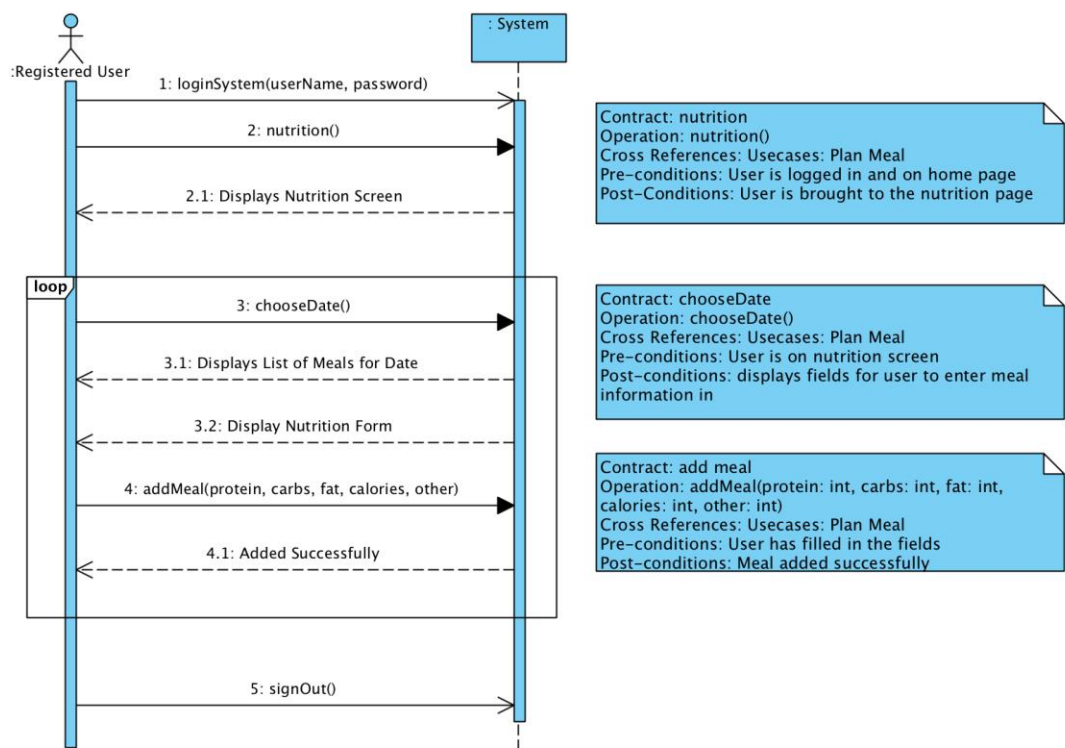
sd Plan Workout

Contract: plan workouts
Operation: planWorkout()
Cross References: Usecases: Plan Workout
Pre-conditions: user has logged in and has active workouts
Post-conditions: displays different categories to plan

Contract: plan workouts
Operation: planWorkout(Workout: String, fieldToPlan: String, fieldValue: String[])
Cross References: Usecases: Plan Workout
Pre-conditions: user has logged in and has active workouts selected a field to plan
Post-conditions: workout is planned successfully

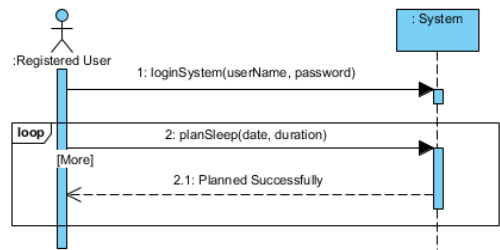
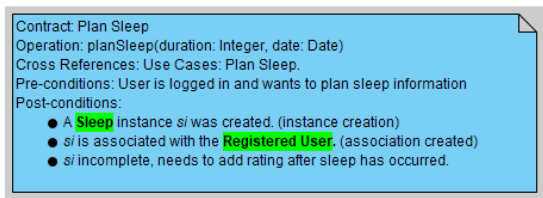


Plan Meal



Plan Sleep

sd Plan Sleep

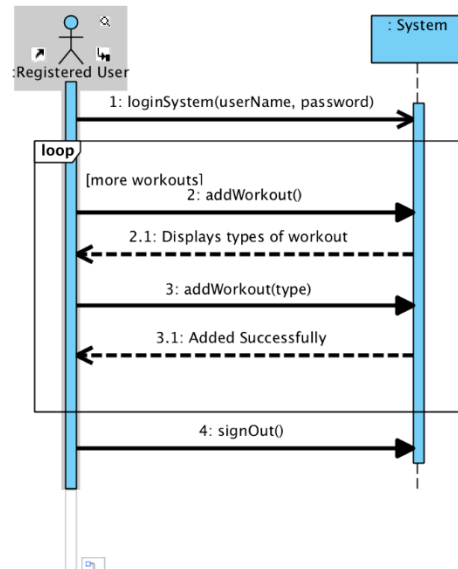


Add Workout

sd Add workout

Contract: add workouts
Operation: addWorkouts()
Cross References: Usecases: Add Workout
Pre-conditions: user has logged in
Post-conditions: displays different types of workout for the user to select

Contract: add workouts
Operation: addWorkouts(type: String)
Cross References: Usecases: Add Workout
Pre-conditions: user has logged in and selected a workout type
Post-conditions: workout is added successfully

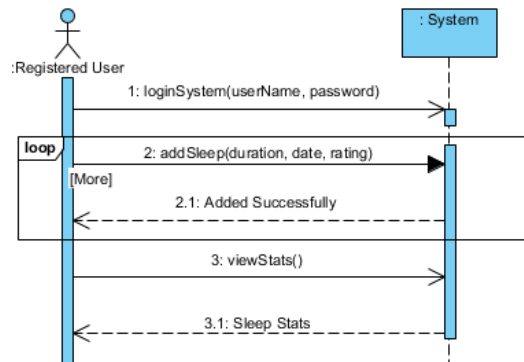


Add Sleep

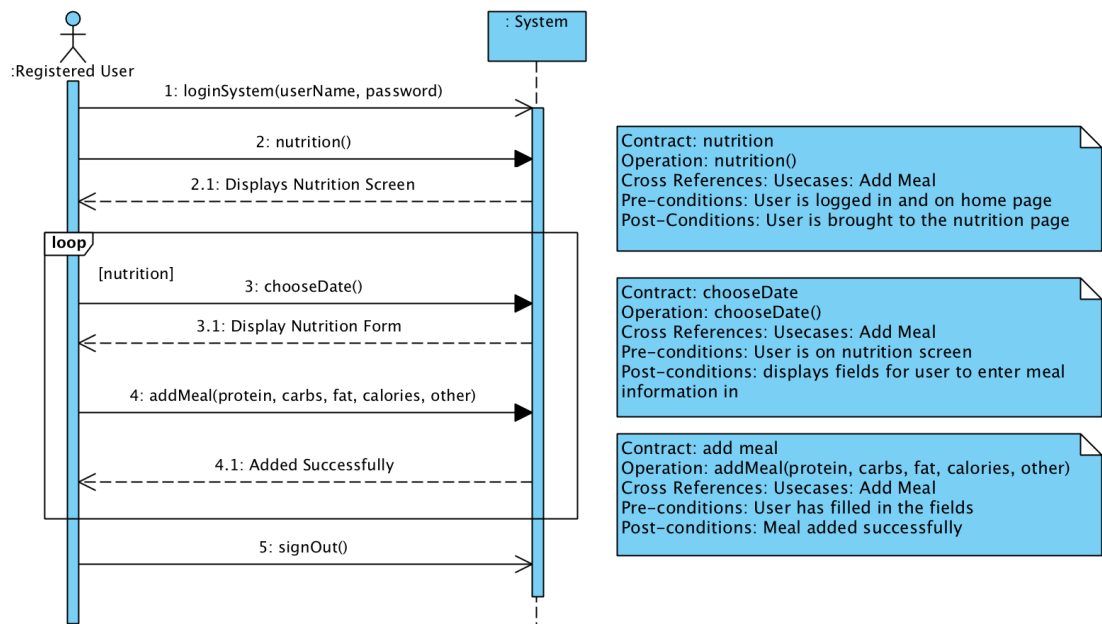
sd Add Sleep

Contract: Add Sleep
Operation: addSleep(duration: Integer, date: Date, rating: Integer)
Cross References: Use Cases: Add Sleep.
Pre-conditions: User is logged in and wants to add sleep information
Post-conditions:
● A **Sleep** instance *si* was created. (instance creation)
● *si* is associated with the **Registered User**. (association created)

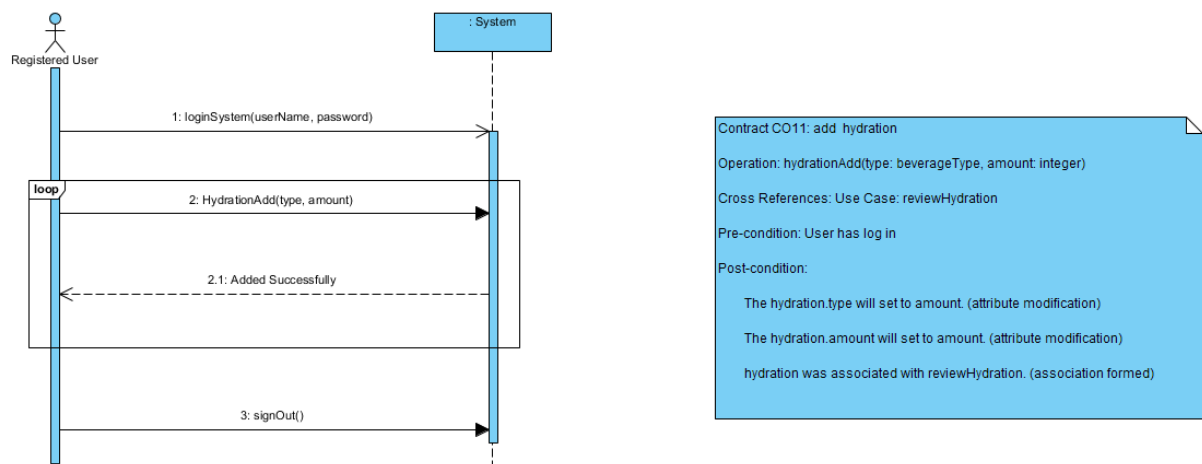
Contract: View Stats
Operation: viewStats()
Cross References: Use Cases: Add Sleep.
Pre-conditions: User is logged in and has already added Sleep information.
Post-conditions:
● **User** sleep statistics are updated. (Attribute Modification)
● **User** sleep statistics are displayed.



Add Meal



Add Hydration



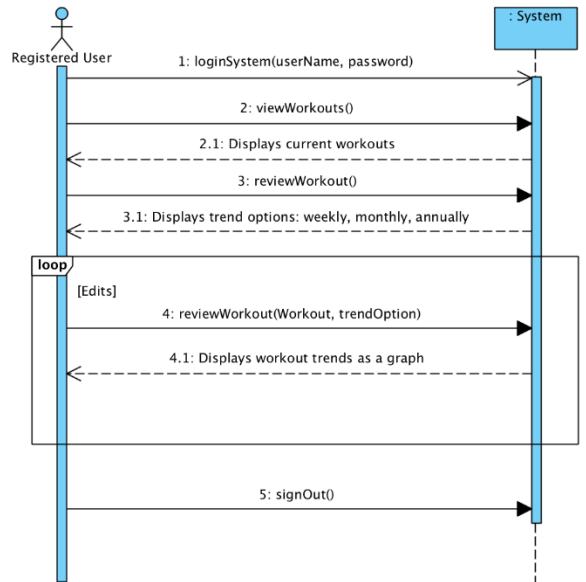
Review Workout

sd Review Workout

Contract: view current workouts
Operation: viewWorkouts()
Cross References: Usecases: Review Workout
Pre-conditions: user has logged in
Post-conditions: current workouts are displayed to the user

Contract: review workouts
Operation: reviewWorkouts()
Cross References: Usecases: Review Workout
Pre-conditions: user has logged in and has active workout
Post-conditions: trend options are displayed

Contract: review workouts
Operation: reviewWorkouts(Workout: String[], trendOption: String)
Cross References: Usecases: Review Workout
Pre-conditions: user has logged in and has active workout and has selected a trend option
Post-conditions: workout trends are modified for desired frame and displayed as a graph

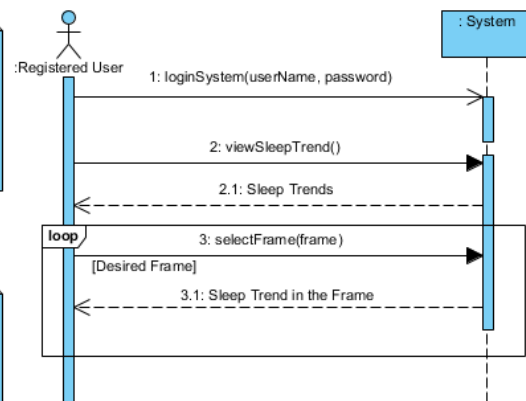


Review Sleep

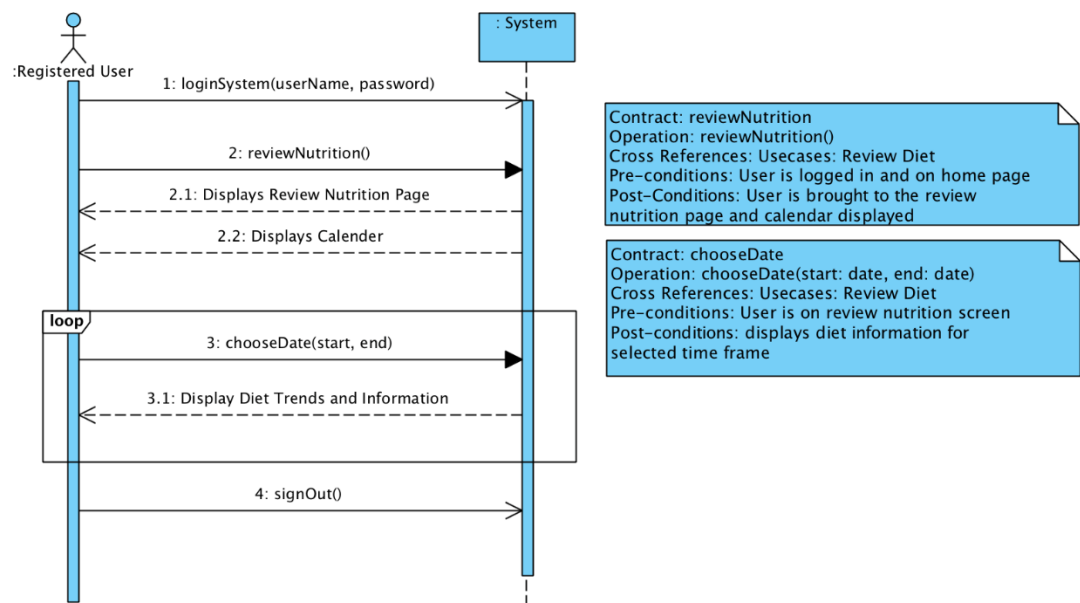
sd Review Sleep

Contract: View Sleep Trends
Operation: viewSleepTrends()
Cross References: Use Cases: Review Sleep.
Pre-conditions: User is logged in, has previously entered some sleep information, and wants to review sleep trends
Post-conditions:
● Sleep trends are displayed graphically with all logged sleeps being graphed.

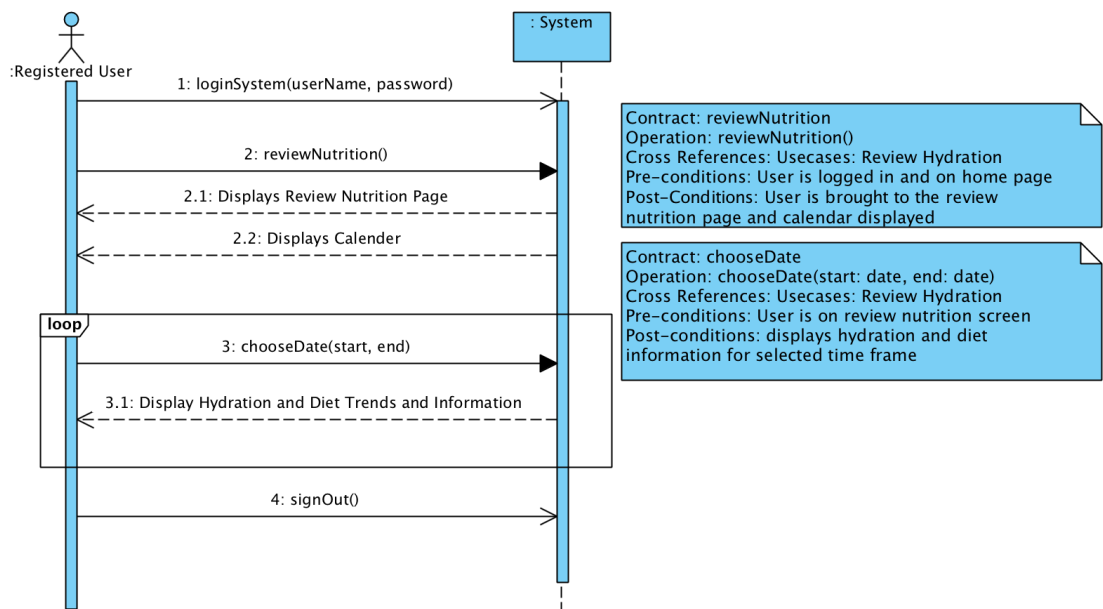
Contract: Select Frame
Operation: selectFrame(frame: frameType)
Cross References: Use Cases: Review Sleep.
Pre-conditions: User is logged in, has previously entered some sleep information, is currently reviewing sleep trends and wants to change the time frame for the trends.
Post-conditions:
● Sleep trends are modified for desired frame and displayed graphically



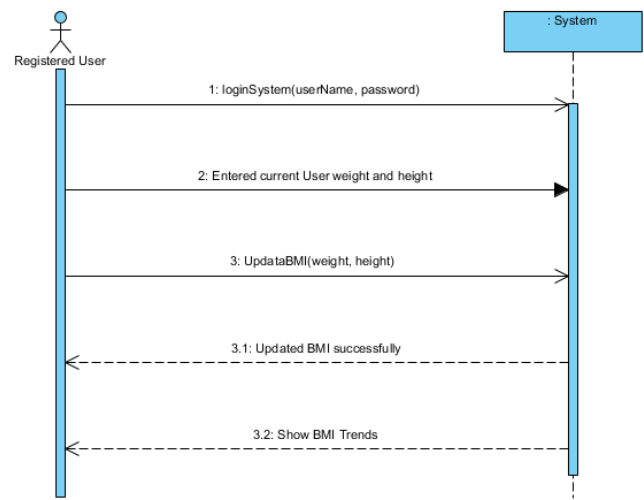
Review Diet



Review Hydration



Review BMI



Contract CO12: review BMI

Operation: UpdateBMI(weight: float, height: float)

Cross References:

Pre-conditions: User has log in

Post-conditions:

- UserBMI.weight was set to weight. (attribute modification)
- UserBMI.hight was set to height. (attribute modification)
- UserBMI was updated and System will show the BMI trends.

Wireframes

Create Account

New User

Username:

Email:

Password:

Create Account

Log In

FitLife

Username:

Password:

Login

Register

Account Information



My Stats

Welcome, <Username>

My Account



Email:

email@provider.com



Username:

<Username>



Password:




Change Username

Change Password

Delete Account

Review



My Stats

Welcome, <Username>

My Account

Exercise Review

September 2018

Mo	Tu	We	Th	Fr	Sa	Su
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

September 16, 2018

<Exercise Name>

185lbs

10 reps

3 sets

<Exercise Name>

10lbs

15 reps

3 sets

<Exercise Name>

200lbs

5 reps


2 sets

<Exercise Name>

135lbs

10 reps

4 sets



My Stats

Welcome, <Username>

My Account

Sleep Review


September 2018

Mo	Tu	We	Th	Fr	Sa	Su
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

September 16, 2018

Sleep Time:

7hrs 34min



My Stats

Welcome, <Username>

My Account

Nutrition Review

September 2018

Mo	Tu	We	Th	Fr	Sa	Su
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

September 16, 2018

Meal 1:

Protein: 100g

Natural Fats: 100g

Water: 28oz

Carbohydrates: 100g

Calories: 1,560

Meal 2:

Protein: 100g

Natural Fats: 100g

Water: 16oz

Carbohydrates: 100g

Calories: 1,700

Meal 3:

Protein: 100g


Natural Fats: 100g

Water: 16oz

Carbohydrates: 100g

Calories: 800

Logging/Planning

 My Stats

Welcome, <Username>

My Account

Exercise

Sunday

Monday

Tuesday

Wednesday

Thursday

Friday

Saturday

Exercise Name:

Weight:Repetitions:Sets:

Add Exercise

Create Workout

Sunday:

<Exercise Name>

185lbs30reps3sets

<Exercise Name>


100lbs15reps3sets

<Exercise Name>

200lbs5reps2sets

<Exercise Name>

135lbs30reps4sets

 My Stats

Welcome, <Username>

My Account

Sleep

Sunday

Monday

Tuesday

Wednesday

Thursday

Friday

Saturday

Start Time:

End Time:

Add Sleep

Sunday:

Total Sleep Time:

8hrs

 My Stats

Welcome, <Username>

My Account

Nutrition

Sunday

Monday

Tuesday

Wednesday

Thursday

Friday

Saturday

Protein:

Carbohydrates:

Natural Fats:

Calories:

Water:

Add Meal

Meal 1:

100g100g

100g64oz

Meal 2:

100g100g

100g64oz

Meal 3:

100g100g

100g64oz

Meal 4:

100g100g

100g64oz

Home

 Welcome, <Username>

My Stats

My Account ▾

NUTRITION

Review Nutrition


EXERCISE

Review Exercise

SLEEP

Review Sleep

Stats




My Stats


Welcome, <Username>

My Account ▾


Weight:

180lbs 


Height:

74in 


BMI:

15.2 

Avg Sleep:

7hrs 23min 

Avg Calories:

3200 

Avg Workouts Per Week:

3.2 

Domain Model

