List of Entity Methods

Project Title: Snappy Workout

Team Name: Snappy

Team Members' Names:

Jianqiang Du
Peter Bui
Taylor Bui (leader)
Trung Nguyen

1. User

1.1. Method: getUserName

1.1.1. Description: returns the name of the user stored in the database

1.1.2. Preconditions: user's name is in the database

1.1.3. Postconditions: user's name is returned to the control object

1.1.4. Signature: String getUserName();

1.2. Method: getUserEmail

1.2.1. Description: returns the user's email stored in the database

1.2.2. Preconditions: user's email is in the database

1.2.3. Postconditions: user's email is returned to the control object

1.2.4. Signature: String getUserEmail();

1.3. Method: getPassword

1.3.1. Description: returns the user's password stored in the database

1.3.2. Preconditions: user's password is in the database

1.3.3. Postconditions: password of the user's account is returned to the control object

1.3.4. Signature: String getPassword();

1.4. Method: setUserName

1.4.1. Description: sets the name of the user based on the inputted name string

1.4.2. Preconditions: name must be non-empty and not alphanumeric

1.4.3. Postconditions: User name is updated in the database

1.4.4. Signature: void setUserName(String input);

1.5. Method: setUserEmail

1.5.1. Description: sets the email of the user based on the inputted email string

1.5.2. Preconditions: email is non-empty and matches email regular expression (REGEX)

REGEX:

^[a-zA-Z0-9.!#$%&'*+=?^_`{|}~-]+@[a-zA-Z0-9](?:[a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?(?:\.[a-zA-Z0-9](?:[a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?)*$

1.5.3. Postconditions: user email is updated in the database

1.5.4. Signature: void setUserEmail(String userEmail);

1.6. Method: setPassword

1.6.1. Description: sets the account password for the user based on the inputted password string

1.6.2. Preconditions: password must be non-empty and is between 6 to 20 characters

1.6.3. Postconditions: password of the user account is updated in the database

1.6.4. Signature: void setPassword(String password);

2. Workout Plan

2.1. Method: getWorkoutPlanName

2.1.1. Description: returns the name of the Workout Plan store in the database.

2.1.2. Preconditions: Workout Plan is in the database

2.1.3. Postconditions: Workout Plan's name is returned to the control object

2.1.4. Signature: String getWorkoutPlanName();

2.2. Method: setWorkoutPlanName

2.2.1. Description: sets the name of the Workoutplan name based on user inputted string.

2.2.2. Preconditions: at least one Workout Plan exists for the user. The name of the workout plan must be non-empty and less than 60 characters.

2.2.3. Postconditions:Workout Plan name is updated in the database.

2.2.4. Signature: String setWorkoutPlanName();

2.3. Method: addWorkoutPlan

2.3.1. Description: creates a workout plan and stores it in the database

2.3.2. Preconditions: User does not have more than 20 Workout Plan.

2.3.3. Postconditions: Workout plan is created and stored in the database

2.3.4. Signature: void addWorkoutPlan();

2.4. Method: removeWorkoutPlan

2.4.1. Description: removes a workout plan from the database

2.4.2. Preconditions: User has at least 1 Workout Plan.

2.4.3. Postconditions: Workout plan is removed from the database.

2.4.4. Signature: void removeWorkoutPlan(String name);

2.5. Method: updateWorkoutPlan

2.5.1. Description: sets any of exercise and exercise date for the Workout Plan
2.5.2. Preconditions: User has at least 1 Workout Plan
2.5.3. Postconditions: User Workout Plan is updated in the database.
2.5.4. Signature: void updateWorkoutPlan();

2.6. Method: exportWorkoutPlan

2.6.1. Description: writes a Workout Plan .json file to the user.
2.6.2. Preconditions: User must have at least 1 Workout Plan stored in the database.
2.6.3. Postconditions: a .json file is sent to the User
2.6.4. Signature: void exportWorkoutPlan();

2.7. Method: importWorkoutPlan

2.7.1. Description: sets a Workout Plan based on a .json file
2.7.2. Preconditions: User must input a valid .json file and must not have more than 19 Workout Plan
2.7.3. Postconditions: a Workout Plan is created based off the .json file and updated in the database.
2.7.4. Signature: void importWorkoutPlan(File JSON);

3. Exercise

3.1. Method: addExercise

3.1.1. Description: updates the Workout plan by adding an exercise
3.1.2. Preconditions: User has at least 1 Workout Plan
3.1.3. Postconditions: Workout Plan is updated in the database.
3.1.4. Signature: void addExercise();

3.2. Method: removeExercise

3.2.1. Description: updates the Workout plan by removing an exercise
3.2.2. Preconditions: User has at least 1 Workout Plan with at least 1 at least 1 exercise
3.2.3. Postconditions: Workout Plan is updated in the database.
3.2.4. Signature: void removeExercise();

3.3. Method: updateExercise

3.3.1. Description: updates the exercise rep count.
3.3.2. Preconditions: User has at least 1 Workout Plan with at least 1 exercise
3.3.3. Postconditions: Workout Plan's exercise is updated in the database.
3.3.4. Signature: void updateExercise();

3.4. Method: markComplete

3.4.1. Description: The exercise's complete variable is marked as true

3.4.2. Preconditions: User has at least 1 Workout Plan with at least 1 exercise

3.4.3. Postconditions: The exercise "complete" variable is updated as true in the database

3.4.4. Signature: void markComplete();

4. Stopwatch

4.1. Method: startStopwatch

4.1.1. Description: The timer function is invoked

4.1.2. Preconditions: User is logged in

4.1.3. Postconditions: The stopwatch function starts counting by seconds

4.1.4. Signature: void startStopwatch();

4.2. Method: stopStopwatch

4.2.1. Description: The timer function is stopped and returns a number.

4.2.2. Preconditions: User is logged in

4.2.3. Postconditions: The stopwatch function finishes and returns the seconds passed since startStopwatch() was called.

4.2.4. Signature: double stopStopwatch();

4. Progression Data

4.1. Method: exportProgression

4.1.1. Description: The application writes a .json file to the device. The file contains the Workout data from the user

4.1.2. Preconditions: User is logged in

4.1.3. Postconditions: .json file is written to the device.

4.1.4. Signature: void exportProgression();