**Algorithm design**

**Hangyul Kim - Doing task of health care**

Initialize status = NULL

Initialize task\_id = 0;

**Loop**

Wait for the User input

If User input found

Grab the input and status

Request user\_id and user\_password

If user\_id and user\_password does not match

Re-request new user\_id and user\_password

Log-in to system

Wait for the User choice

If User choice found

Grab the choice

If choice = HEART\_RATE

Set status = ON

Set task\_id = 1

Ifelse choice = SLEEP\_CYCLE

Set status = ON

Set task\_id = 2

Ifelse choice = BLOOD\_PRESSURE

Set status = ON

Set task\_id = 3

Else

Set status = OFF

If status = ON

If task\_id = 1

Go to heart\_rate

task\_id = 0

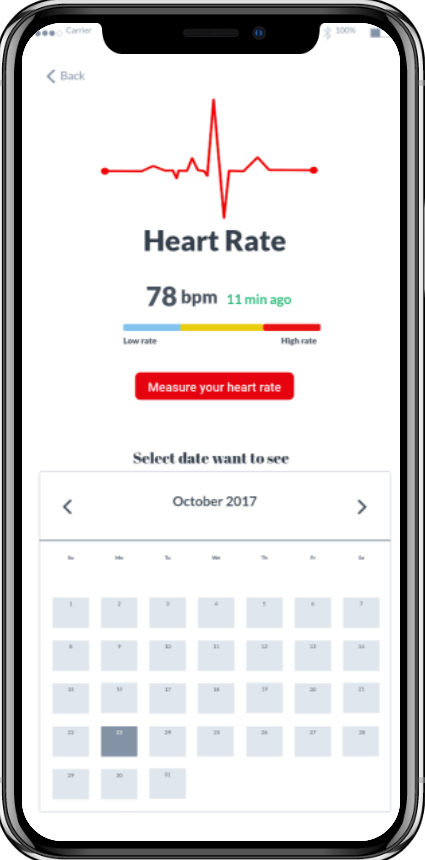
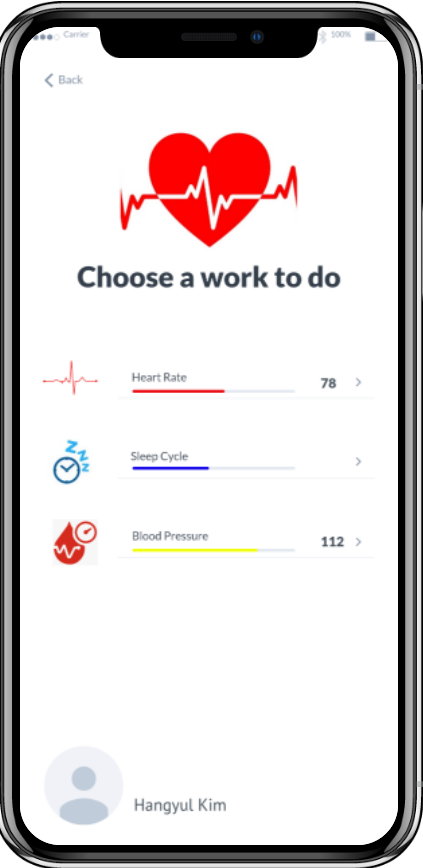
Ifelse task\_id = 2

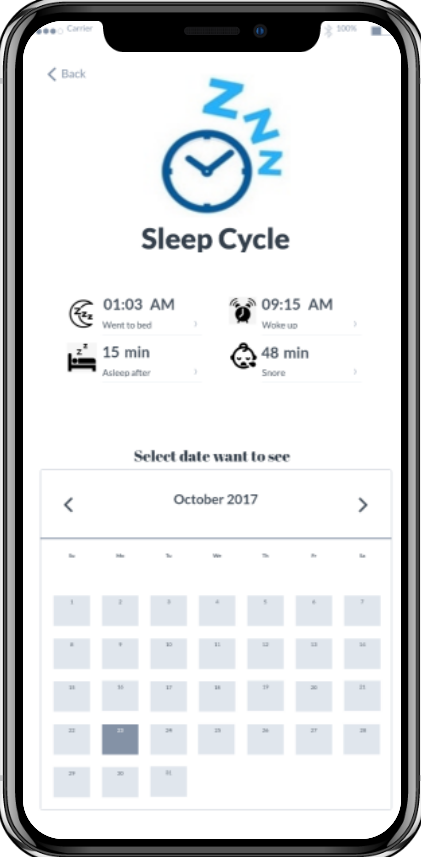
Go to sleep\_cycle

task\_id = 0

Ifelse task\_id = 3

Go to blood\_pressure

task\_id = 0



Zac Comer

**Creating and Storing Meal Data**

Present Meal Selection Screen

Wait for user input for item[i]

Search foods database for items related to the input name

If food database item is selected

Fill other fields with pre stored nutrition information

Else

Wait for user input for other fields

If user selects add\_item

Increment i

Initialize item[i]

Else if user selects submit

Finalize data values

continue

Package the data into JSON object called meal\_data

Submit meal\_data to meal\_database

Check that submission was accepted

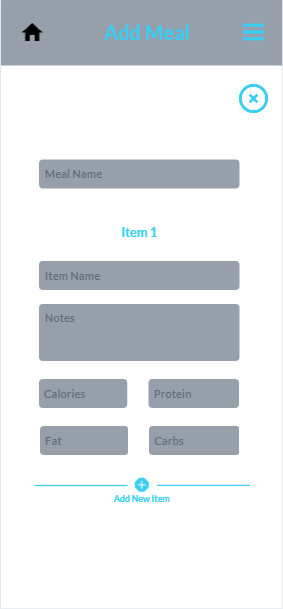
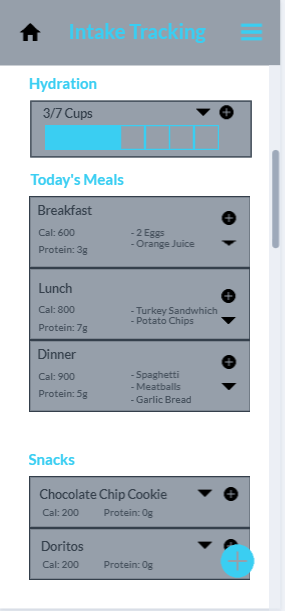
If not accepted

Present error message

Return to the main Meals Page

Retrieve all relevant meal\_data objects from meal\_database

Display meal information on Meals Page



**Algorithm Design**

Workout Goals Selection Screen

Wait for user to input currentWeight, desiredWeight, and time available for workout

While user is selecting workouts

Search Workout database for workouts

Present the workout to the user

If the user accepts:

Add to the workout array

If the user declines:

continue

Loop finished

Time per workout = time available / length of workout array

Package the workout into a json object called WORKOUT\_PLAN

Add the json to the users data

Ask user for the frequency of notifications

Update the notification information of users data

Return the user to the main workouts page

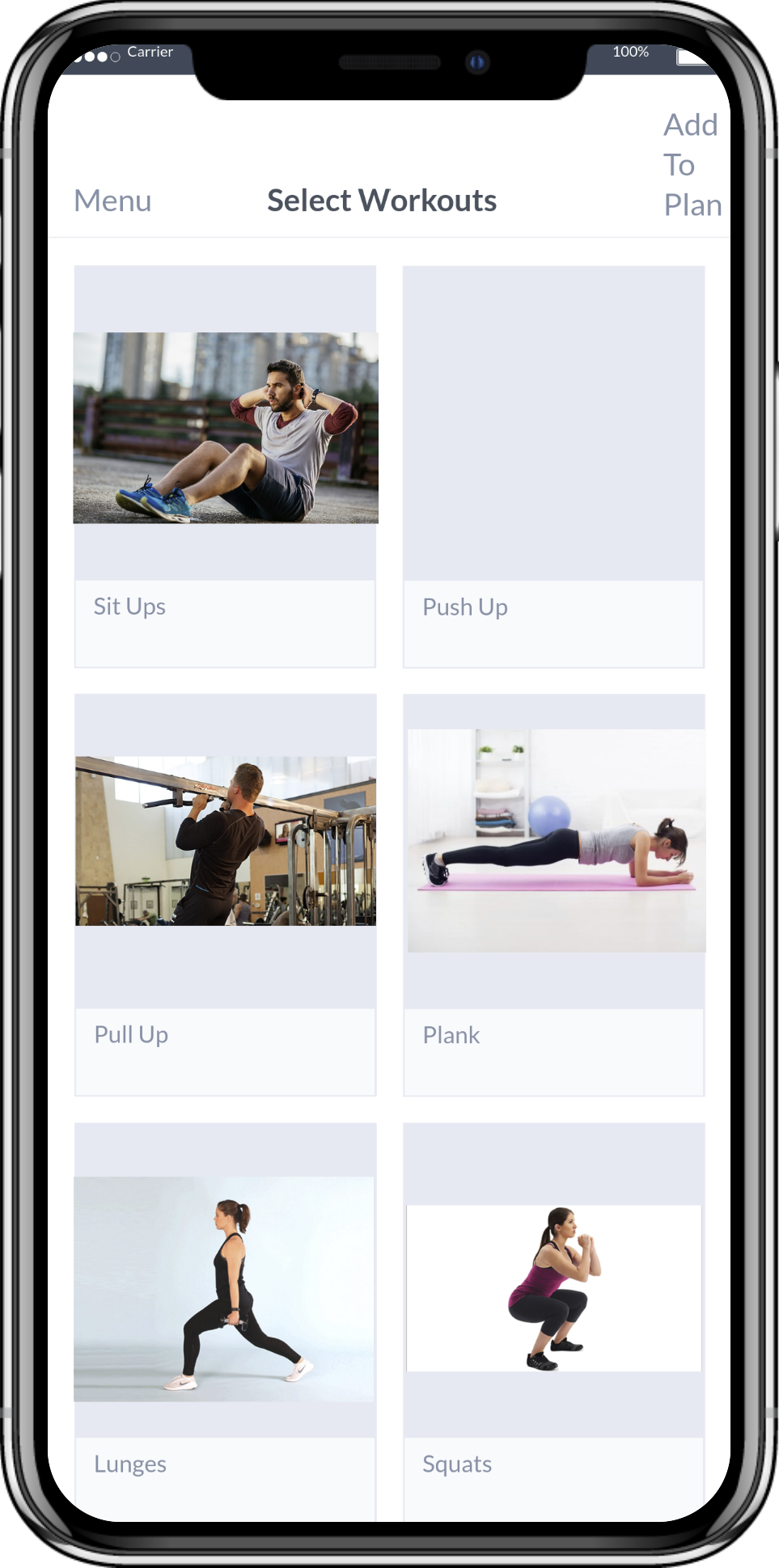
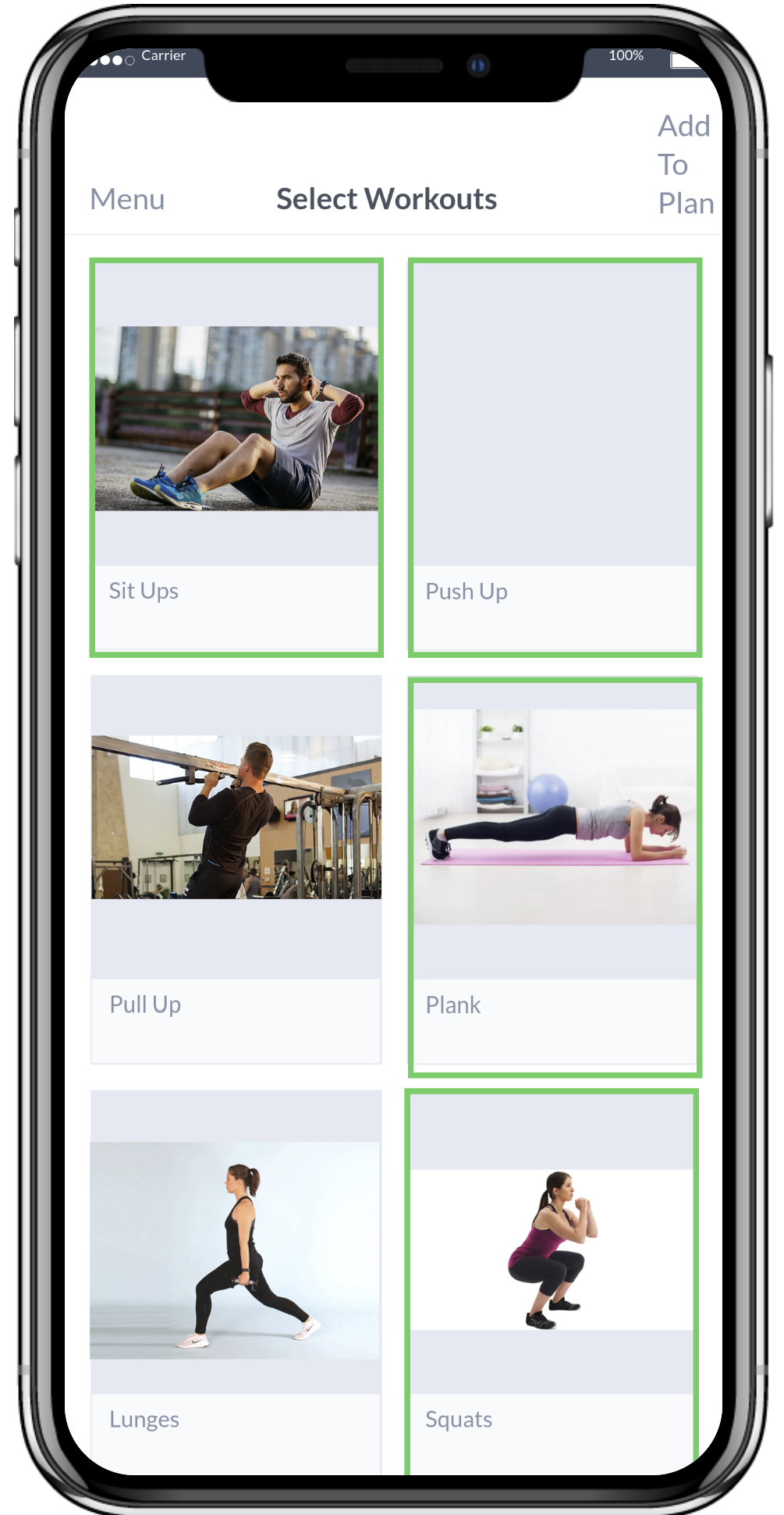
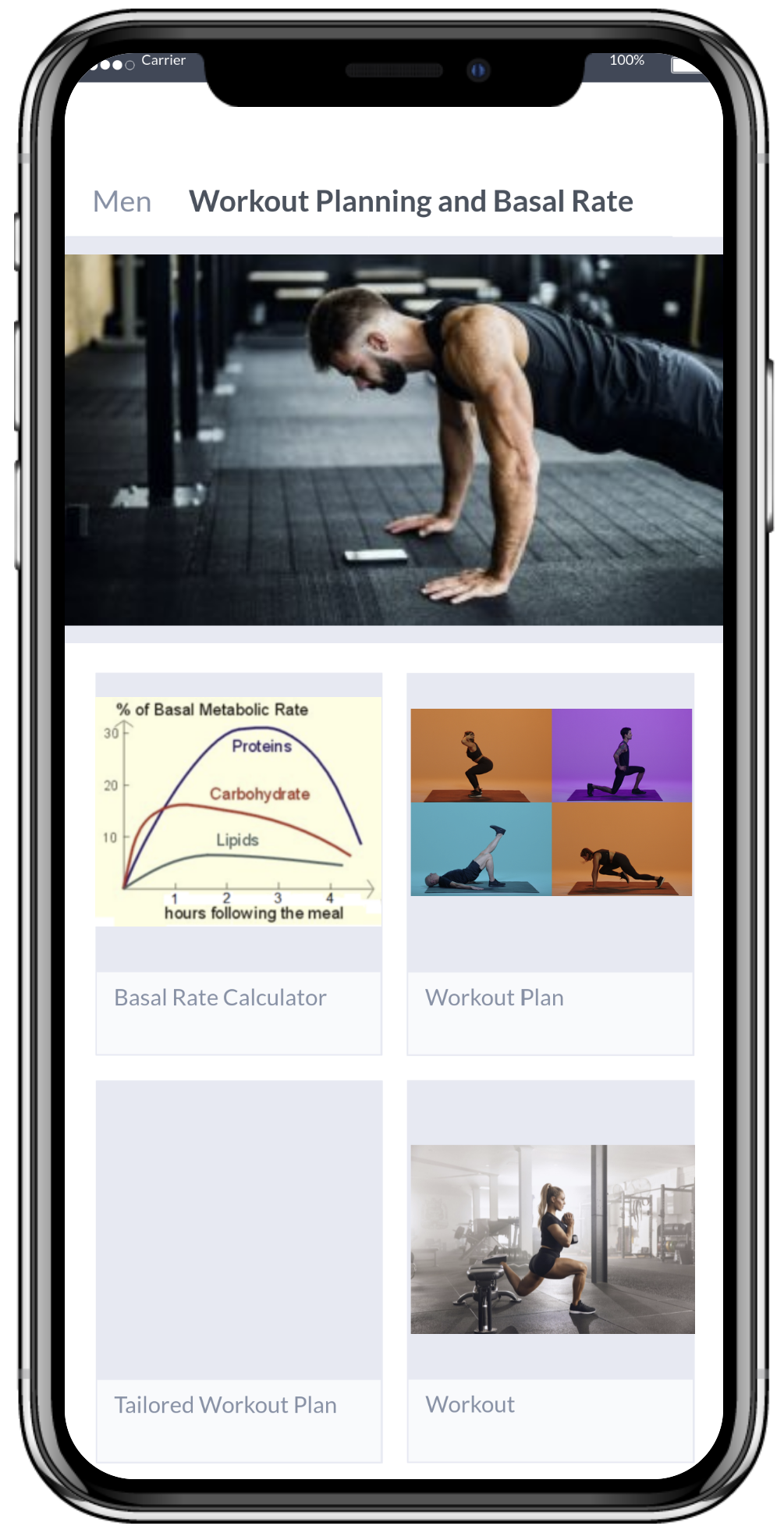
Show user newest workout and time required to complete the workout

Allow user to start the workout or edit it

**UI Design**

Shown on Next Page

1. User clicks the workout plan button
2. Is presented Workout Plans
3. Selects Plans
4. Clicks Add to Plan
5. App Will go back to Workout Planning and Basal Rate Page
6. The User clicks workout
7. The Workout is shown with the time/reps and the sets



Liam Davis-Wallace

**Record Activity Screen**

Wait for user input

While recording workout

Capture user’s GPS information

If GPS location is unavailable

Notify user

Track heart rate

If heart rate is above safe threshold

Alert user to reduce effort

If workout is paused

Wait for user input

If Resumed

Break from waiting loop

Else if Workout Ended

Exit outermost loop

If workout is complete

Package data into JSON object containing distance, time, gps data, etc.

Prompt user to edit workout title & description

Update Title and Description fields of JSON object accordingly

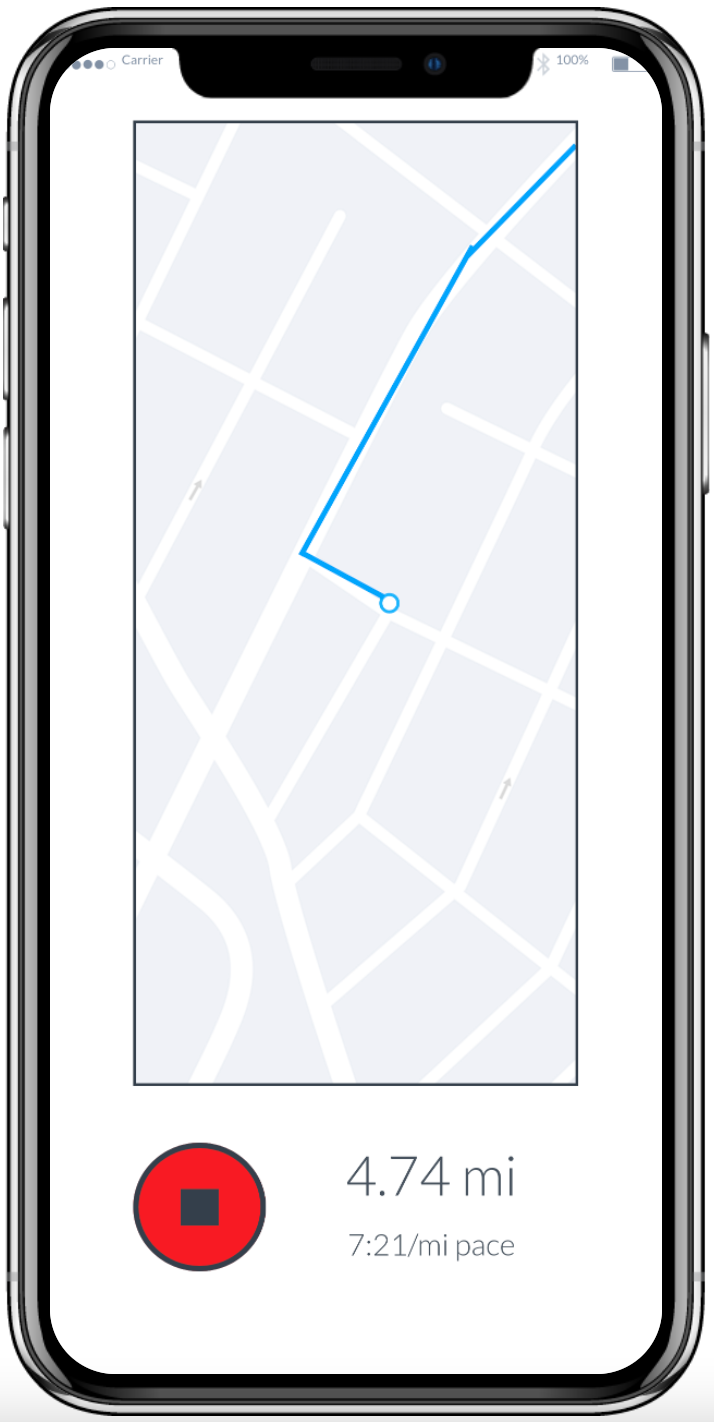
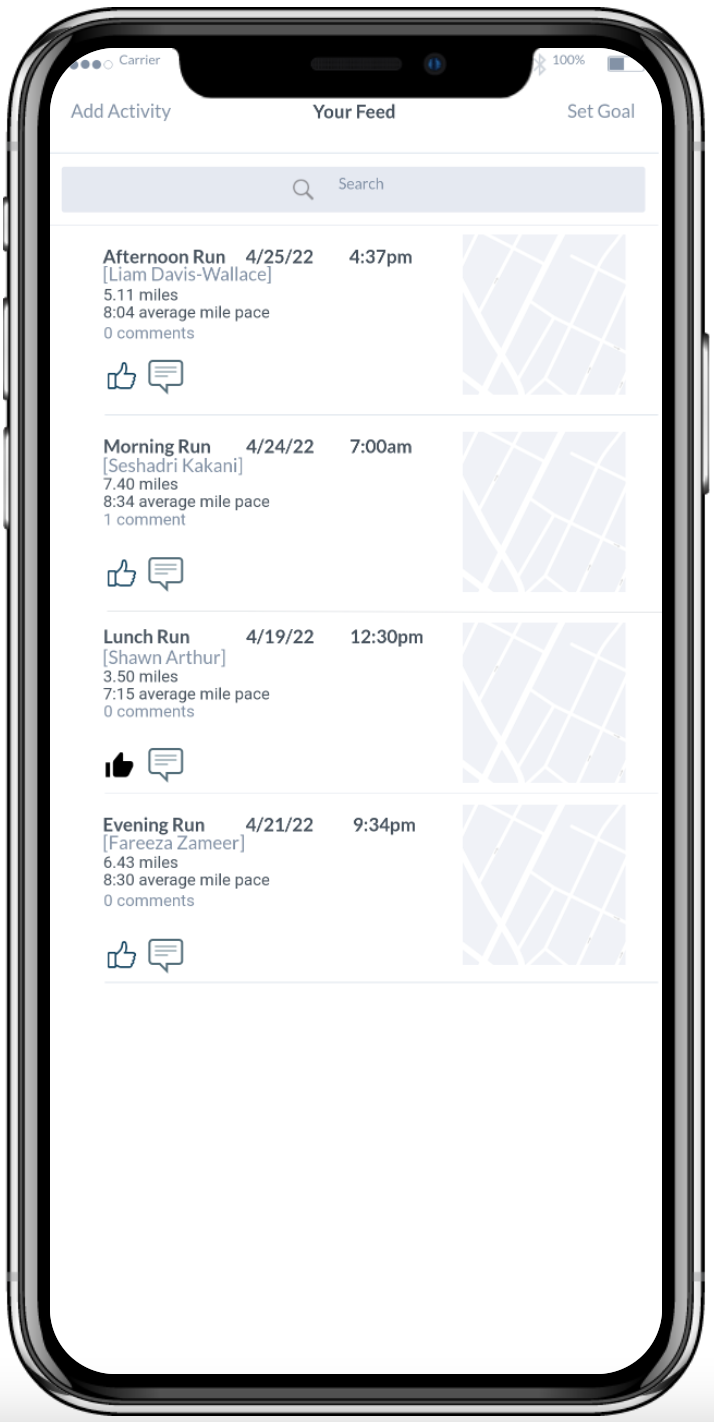
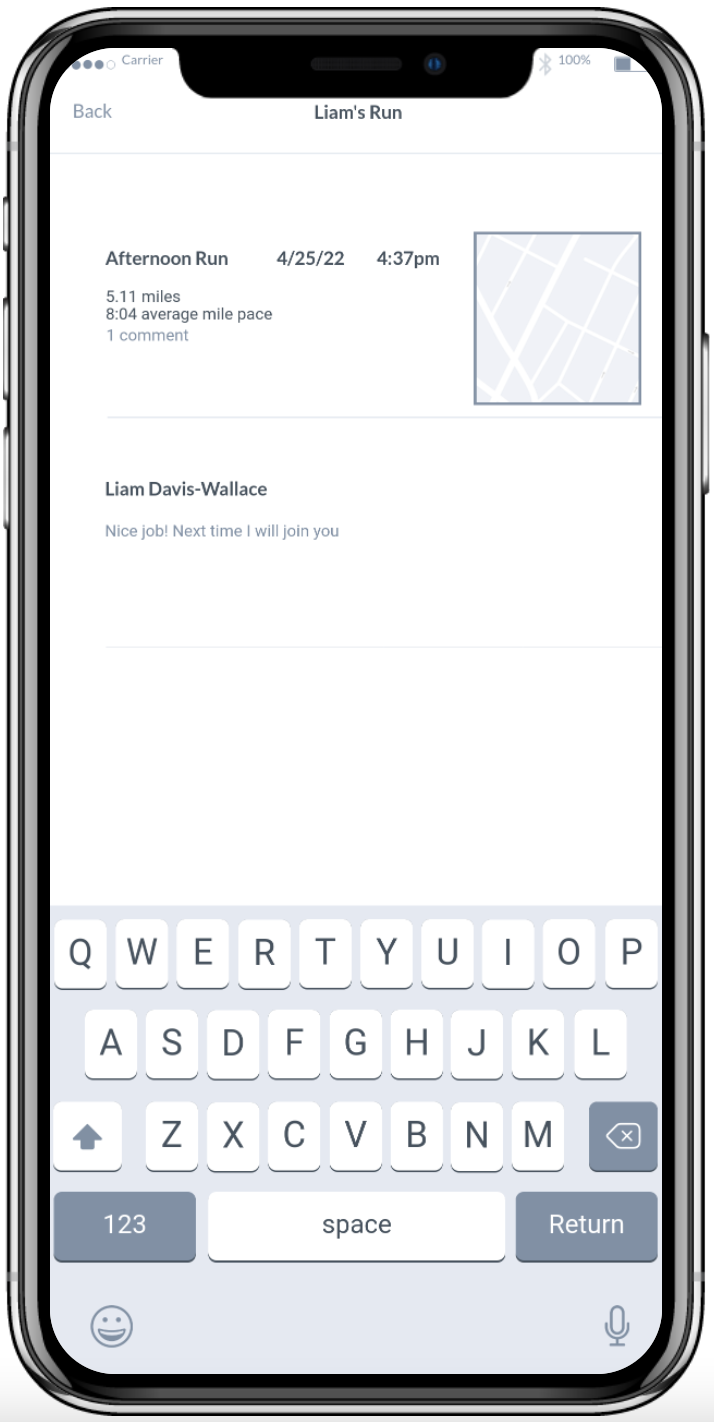
Prompt user to post workout

If user wants to post workout

Upload workout data to server, make public

Else

Upload workout data to server, make private



Shawn Arthur

**Creating and storing health survey data**

Present health survey screen

score\_list = list()

Wait for user input for question[i]

if valid input

append to score\_list

else

present error message

mental\_health\_score = compile\_scores(score\_list)

Package mental\_health\_score into JSON object

submit mental\_health\_score to database

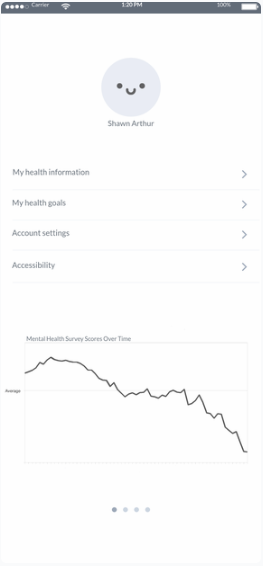
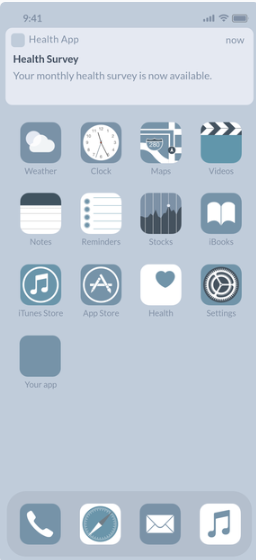
check if submission was accepted

if not accepted

present error message

return to account page with mental health data

plot mental health data over time on user profile.



Fareeza Zamer

**Run and Bike GPS**

Wait for user to press start

While recording workout

Capture user’s GPS information

If GPS location is unavailable

Notify user

Track heart rate

If heart rate is above safe threshold

Alert user to reduce effort

If workout is paused

Wait for user input

If Resumed

Break from waiting loop

Else if Workout Ended

Exit outermost loop

Track the amount of calories

If calories are burned

Notify user

Else

Display starting calories

If workout is complete

Package data into JSON object containing pace, elevation change, heart rate, calories

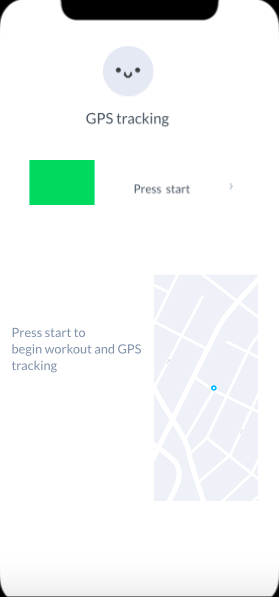
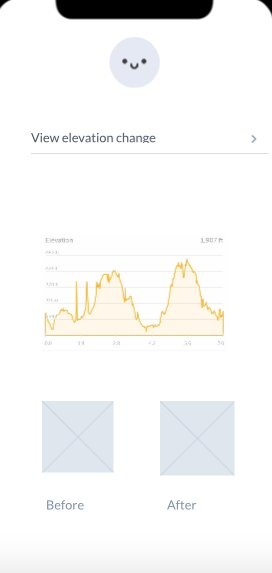
Allow user to view data

If user wants to view data

Give option to select pace, elevation change, heart rate, calories

Else

Allow user to exit

** **