Team Name: 🌞
Project Proposal: UV Band *(final product name TBD, options: Sundial, UV Dial, UV Me)*
Members: Reema Al-Marzoog (software), Melissa Blotner (software), Eliza Schreibman (hardware), Jake Mingolla (UI/UX)
Due: March 14, 2016

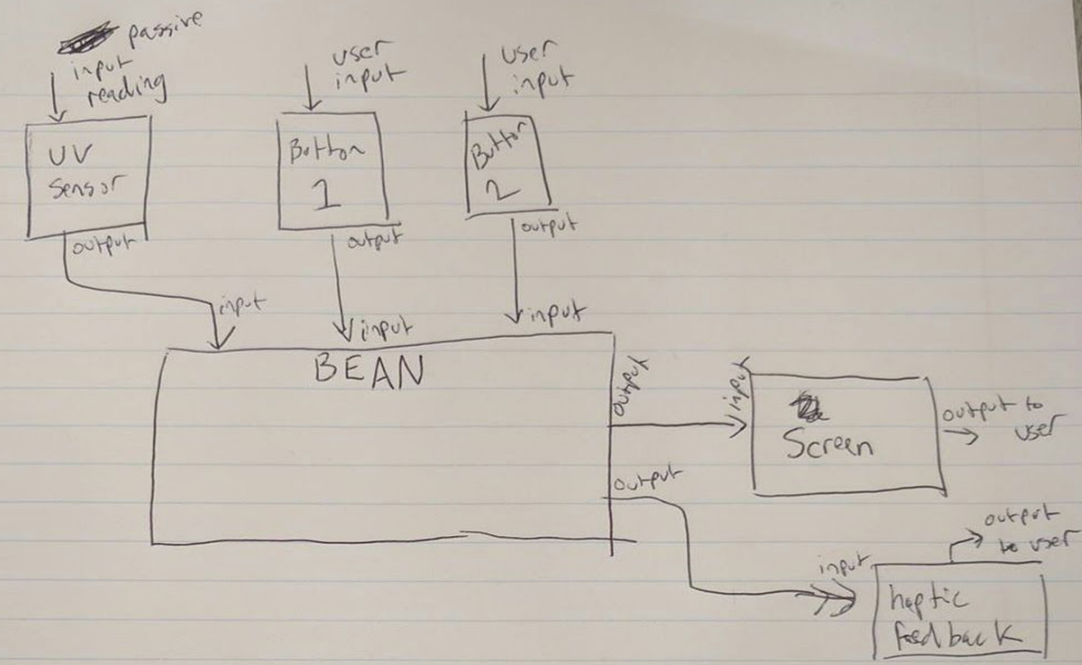
Mission Statement

Our device is an everyday, unintrusive wearable suitable for all people, but especially those susceptible to sunburns (e.g. children, pale people, Australians). The device will not only track and display your sun exposure over time but also train you to understand how the sun affects your body so you can practice good sun protection habits even without the device.

Big Idea

We wanted make an easy to use wearable that can help incrementally improve health habits among people who aren't as conscious of skin damage. Most of our team members have been sunburned before and believe our device could be a solution to help people in our same situation. We believe that this product is not only useful and time saving but also has the potential to have a real impact on people's health and well being.

Object Diagram



Bean sketch

Countdown

- inputs: sensor data
- outputs: Screen, IOS

Clock/Watch

- inputs: none
- output: Screen

Haptic feedback

- input: none
- output: sensor

IOS App

Countdown

- input: Bean
- Output: display to user

UV Graph

- input: UV sensor (From Bean)
- output: display to user

* Reach goal *

Storing user information about UV exposure and sun screen use on a server for persistent data storage and display (over time

Hardware Overview

Sensors/Components

UV sensor ML8511 (i.e. UV sensor M)

- Datasheet (https://cdn.sparkfun.com/datasheets/Sensors/LightImaging/ML8511_3-8-13.pdf)
- Breadboard (https://cdn.sparkfun.com/assets/learn_tutorials/2/0/6/ML8511_UV_Hookup.png)
- Schematic (https://cdn.sparkfun.com/datasheets/Sensors/LightImaging/ML8511_Breakout.pdf)
- Example code (https://github.com/sparkfun/ML8511_Breakout/blob/master/firmware/MP8511_Read_Example/MP8511_Read_Example.ino)

UV sensor Si1145 (i.e. UV sensor S)

- Datasheet (https://www.adafruit.com/datasheets/Si1145-46-47.pdf)
- Example setup and code (https://learn.adafruit.com/downloads/pdf/adafruit-si1145-breakout-board-uv-ir-visible-sensor.pdf)

Haptic feedback sensor

- Example setup and code (https://learn.adafruit.com/adafruit-arduino-lesson-13-dc-motors)

Screen Display

- Example setup and code (https://learn.adafruit.com/monochrome-oled-breakouts/)

Inputs/Outputs

	Inputs	Outputs
UV Sensor M	• Power-3.3 • Ground-Gnd	• UV Reading- A0 • Baseline Reading- A1
UV Sensor S	• Power-Vin • Ground- Gnd	• 12C clock- A0 • 12C data- A1
Haptic feedback sensor	• Ground-Gnd	• Pin 0
Screen display	n/a	• Pin 1 • Pin 2 • Pin 3 • Pin 4 • Pin 5 • OR Pins A0 and A1

Note: We will end up using only one of the UV sensors, so that is why we repeat pins A0 and A1. Additionally, there seemed to be multiple ways to wire the screen (some methods used analog pins and others didn't). We will need to find a way to either run it through just digital pins or research if there is anyway to get more than two analog pins for our Bean (since the UV sensor already needs two). There seems to be the possibility that both the screen and one of the UV sensors uses the same clock and data pin, which might allow for some modularity and save on pins, but we will have to investigate this more when we get the actual hardware.

Testing

All of the resources for example code and basic breadboard setup have been linked above in the components list.

- For the UV sensors we can test using sample code as well as checking continuity with our multimeter. It would be great to also test that the UV sensor is outputting correctly via hardware (e.g. have an LED light up brighter with greater UV readings), but that may not be possible due to the format the sensor outputs values in.
- We can easily test the haptic feedback sensor with example code or by simply connecting the one side to power and the other to ground (and the sensor should vibrate).
- For the screen, we have example code and example breadboard visuals, but we also can consult with our classmates who have succeeded in using the screen before.

Software Overview

Bean:
The bean handles the following functionality:

- Read UV from UV sensor
- Countdown module
    - Adjust countdown speed
        - Based on current UV reading, adjusts countdown's "acceleration"
            - We will have an internal function used to adjust countdown based on UV index and experts' recommended frequency of sunscreen application
    - Reset countdown
        - If user has indicated that they put on sunscreen by pressing a button, reset the countdown
- Control vibration
    - When countdown reaches 0, alert user with haptic feedback
- Read input from button
    - Used to reset countdown
- Control display screen - modes (timer, clock, current UV)
    - Rotate between screen modes to change what is being displayed on screen
    - Update display based on countdown

Example technology for using the UV sensor with the bean:
https://learn.adafruit.com/sunscreen-reminder-hat/code

iPhone app:
The iOS app handles the following functionality:

- Listen for data from Bean
    - Types of data received from Bean:
        - Current UV
        - Countdown over
- Graph module
    - Display graph of UV over time
    - Handle adding data points to graph (called when new UV readings are inputted)
- Alerts
    - Triggered when countdown is over
- Sync up display screen to phone
- Reach goal:
    - Store information about UV exposure and sunscreen use on a server for persistent data storage and display metrics over time

Example technology for bean communicating with iPhone: Sonic iOS
Example technology for graphing data: Sonic Mac

The bean communicates with the iPhone app through Serial output over Bluetooth.

<u>Data:</u> Stored in a Heroku app using a database that will communicate with iPhone application

<u>Frameworks and libraries being used:</u>

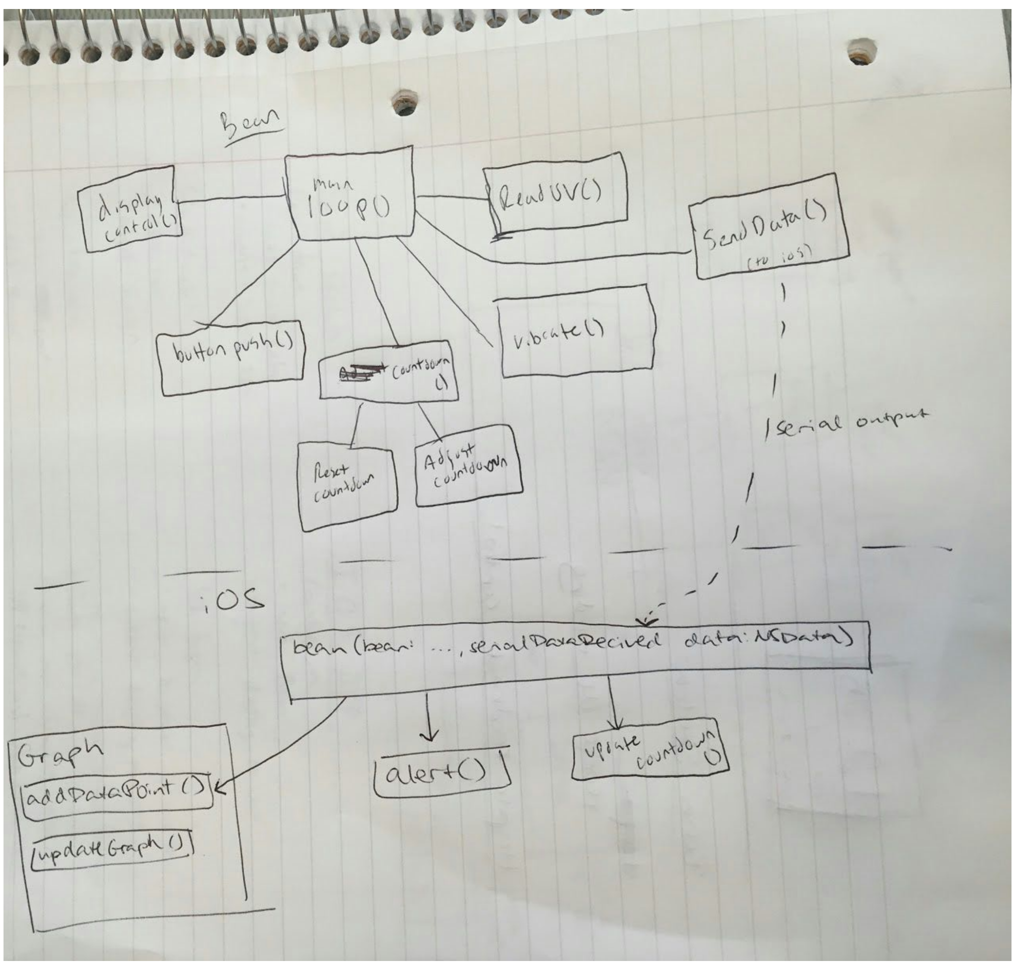
- XCode
- Time.h (in bean sketch)

<u>Testing:</u>

To test independently of hardware:

- We will generate test UV index values rather than reading them in from the sensor. We can do this both on the bean and in the iOS app to test the app independently from the bean sketch.
- We'll output our calculated countdown to Serial rather than to the screen.
- We'll force a button press to be triggered in the code and check, using Serial output, that the correct side-effects occur.

<u>Call tree:</u>



UI/UX Overview:

Our device targets children and people who tend to lose track of time when they're out in the sun. We're aiming to attract users who haven't developed good sun protection habits and want our device to guide them to a better understanding of their skin health. The device will be located on the user's arm (similar to a watch or an iPod armband) so it is both unobtrusive and also easy to use. It will be affixed using a strap that goes around the user's arm.

We will be building off of an existing armband and add the sensors and bean to the armband. The user will interact with the device through the screen and the pair of buttons that control user inputs. The screen send graphical data to the user based on the current mode the device is in. One of the buttons allows for the user to cycle between modes and the other is pressed when the user applies sunscreen to reset the countdown.

In addition, the user can also connect their device to an iOS app for analytics and logging. Our app will graph the user's UV levels over time as well as display the same information onto the phone so that the device could be monitored by the user as well as a third party such as a parent.

Modes:

1. UV countdown
    - This will tell the user when they will next have to put on sunscreen. Our initial design is to have either a displayed number / bar graph go down over time. Once it reaches zero the user is notified that they must reapply sunscreen through the haptic feedback as well as a notification sent to the phone.
2. Current UV exposure
    - This is read directly from the UV sensor and converted into meaningful information to the user (i.e. converting the analog reading to UV intensity)
3. Current time of day
    - In case the user doesn't want to use the UV information or is trying to be more discrete, the device can also be used as a watch.

The user cycles between modes using the button and the relevant information for each mode is displayed on the screen.

To assure that our device meets the needs of the user, we must:

1. Make sure the armband is both comfortable and can also fit the majority of users through an adjustable band.
2. Ensure the screen is visible and readable regardless of its state.
3. Make sure the buttons are well visible and discernable from each other (since we don't want the user resetting the countdown instead of cycling modes)
4. Figure out the best way to display the countdown information — for example, is it better to give the user an estimate of the number of minutes remaining until they put on

sunscreen, have a number that counts down from 100 to 0, or some sort of graphical display (i.e. a series of bars or a pie that decreases over time)

5. Make sure the haptic feedback can be felt yet isn't too intense.
6. (STRETCH) Make the device waterproof so the device can be brought to the beach.

In order to make sure our device meets user expectations, we will perform a few tests to understand what our average user will expect in terms of button/screen placement as well as the display orientation. We will create a few different presets of the screen setup and ask users which they believe is most intuitive.

Preliminary Budget

Item	**Cost**
UV Sensor ML8511 https://www.sparkfun.com/products/12705	$12.95
UV Sensor SI1145 https://learn.adafruit.com/adafruit-si1145-breakout-board-uv-ir-visible-sensor	$9.95
Haptic Feedback Motor x2 https://www.adafruit.com/products/1201	$3.90
Casing	$10
Pack of buttons https://www.adafruit.com/products/1010	$5.95
Total:	$42.75