CS4085 - Problems in Computer Science

Project Documentation

Name : Wei Xian Low

PawPrint : WLKR7

Project Dates : 07/06/2017 - 8/20/2017

Instructor : Joseph Guilliams

College : College of Engineering

University : University of Missouri - Columbia

Project's URL : http://cs4085.weixianlow.me

Project Information

Introduction:

In this project, an automated service will be built based on an API provided by Fitbit. This automated service will monitor a user's heart rate to detect any lack of heart rate by fetching new data from the API on a set time. When the service has detected a lack of heart rate from the user, the service will then flag the user account and then continuously fetch new data from the API to detect any changes from the heart rate data obtained from Fitbit. Depending on the user's preference, the service will then monitor the heart rate data until the selected time by the user, and automatically send out a predesignated zip folder containing all the user's important information such as the last will of the user, important account username and password, and emergency contact information to a designated list of people's email inboxes. If a non-zero heart rate data is received from Fitbit's API, the service will then un-flag the user's account and continue as always.

Requirements

Hardware Requirements:

1. Fitbit wearable with heart rate monitor
    - The Fitbit wearable must have a heart rate monitor to capture user's heart rate.
2. Smartphone with a preconfigured Fitbit application installed
    - User's smartphone must have the Fitbit application installed with the proper synchronization setting.
3. Data connection from a mobile phone ISP or Wi-Fi.
    - A data connection is needed to connect the app to Fitbit's server.
4. Hosting server
    - A hosting server is needed to host the operating system which the services will run.

User Requirements:

1. Ability to monitor heart rate data via Fitbit wearable.
    - A Fitbit wearable must be able to detect the user's heart rate data to provide an accurate heart rate to the Fitbit server
2. Ability to automate heart rate detection without user's intervention.
    - The services must be able to obtain the user's heart rate via the Fitbit API
3. Ability to receive notification warning to avoid false alarm.
    - The services must be able to send email and text notification to the user to prevent incorrect detection of zero value heart rate.
4. Ability to pause service until user reactivates it.
    - The web interface must have the ability to allow users to pause the service.

Project Specification

Expected Technologies Proposed to be used:

The below list are the technologies expected to be used during the project proposal phase:

1. Ubuntu 14.04
    - Ubuntu 14.04 will be the operating system hosting the entire service and web interface.
2. php
    - The services will be written in php to help automate in obtaining data from Fitbit's API.
    - It will also be used to automate script running in detecting zero value heart rate from the obtained data.
3. JavaScript
    - JavaScript is used to populate data and drive the web interface for the user.
4. Google Firebase Realtime Database
    - All data and user authentication information will be hosted at Google's Firebase servers.
5. Google Firebase Authentication
    - User authentication will be handled by Google's Firebase using Google authentication
6. HTML/CSS
    - HTML and CSS will be used to create the web interface.
7. Fitbit API
    - The services will depend on the Fitbit API to obtain user's heart rate data.

Technologies Actually Used:

1. Ubuntu 14.04
2. JavaScript
3. Google Firebase Realtime Database
4. Google Firebase Authentication
5. HTML/CSS
6. Fitbit API

Technology Abandoned:

1. php
    a. Php were not used in this project due to the lack of admin access to Google's Firebase services during the building of the project.
    b. An official API was not available from Google's documentation
    c. Third-parties API for Google Firebase Services were not robust enough to handle script based handling of user's data.

Additional Technologies Added:

1. Twilio
    a. Twilio is a third party service that allows SMS notification to be sent to the user.
    b. This service is crucial to the user requirement of this project as to notify them of any false alarm from any extended lack of heart rate detection.
    c. This service also provides a free trial service for student developers.

2. SendInBlue
    a. SendInBlue is a third party service that helps facilitates SMTP based email sending services to user.
    b. An easy to use python library add-on is available from the service's documentation
    c. Has a free account available for developers to test out its services
3. Google Firebase Storage
    a. Google Firebase Storage is needed to store user's important data in safe keeping after user has uploaded it to the services.
    b. It also has the ability to integrate with other Google Firebase services such as Firebase Authentication to obtain user's UID.
4. Python
    a. Python is the main language used to develop the backend script of this services.
    b. This language is used to develop scripts used to retrieve heart rate data from Fitbit's API and a script to process the collected data from Google Firebase Realtime Database.
    c. Google Firebase has an official Admin SDK written for Python that provides administrative access to the Realtime Database to manipulate and read data.
    d. Python is also support by both Twilio and SendInBlue

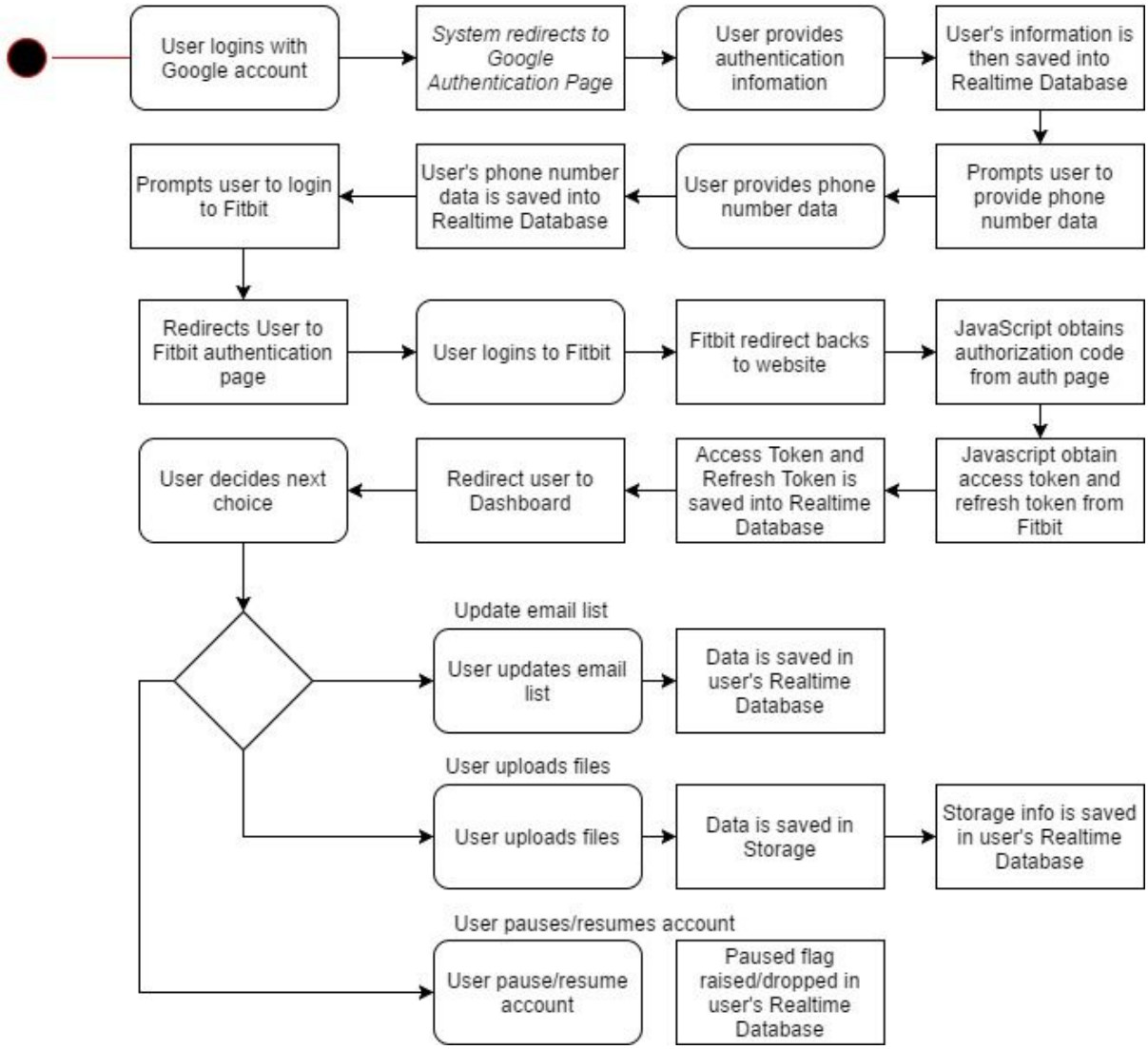
5. Crontab

    a. Crontab is used to schedule backend scripts to automate executing of scripts on an hourly basis.

    b. Crontab is a service built-in with Ubuntu's operating system.

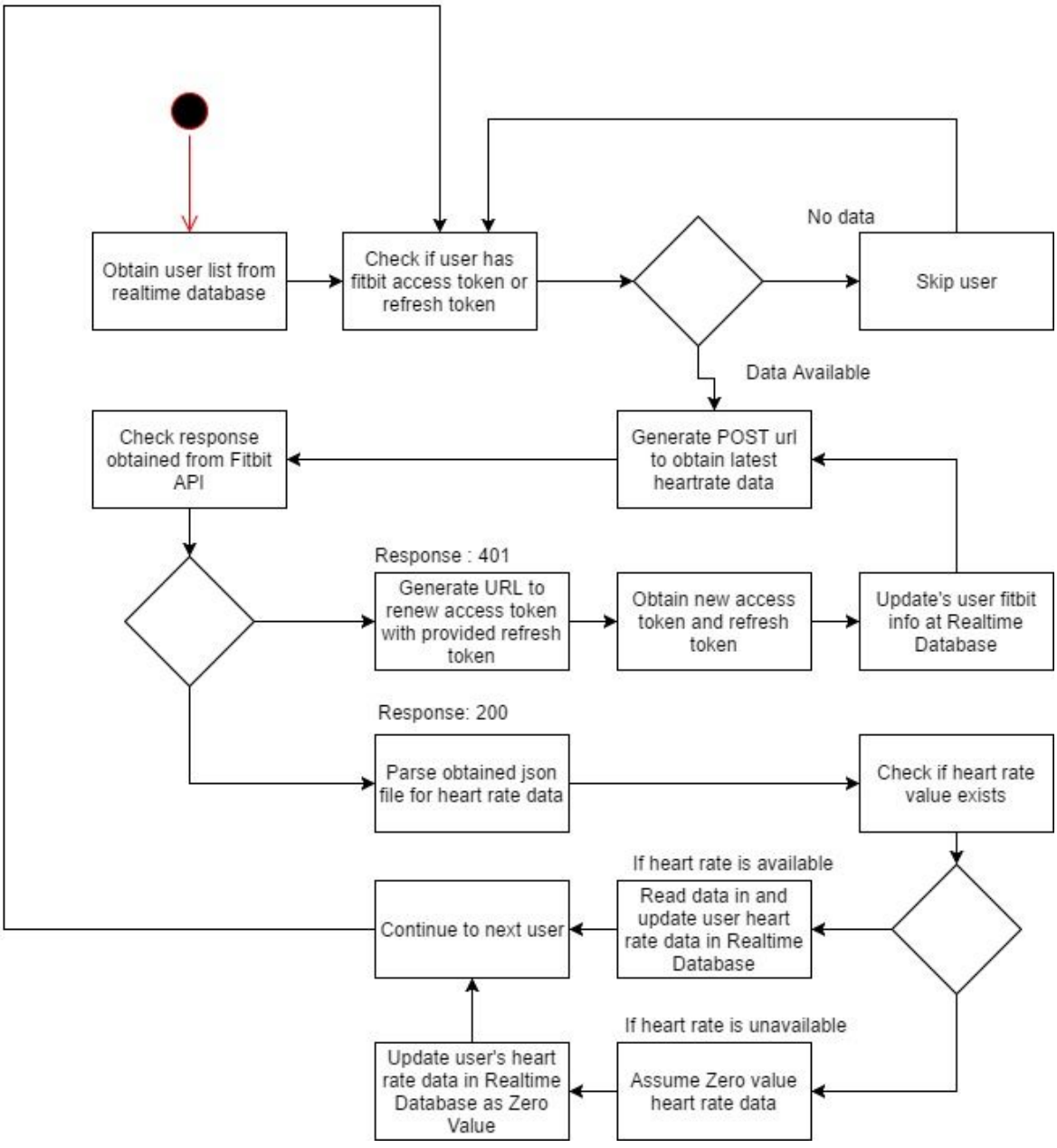
Program Flow Chart:

User login flows:

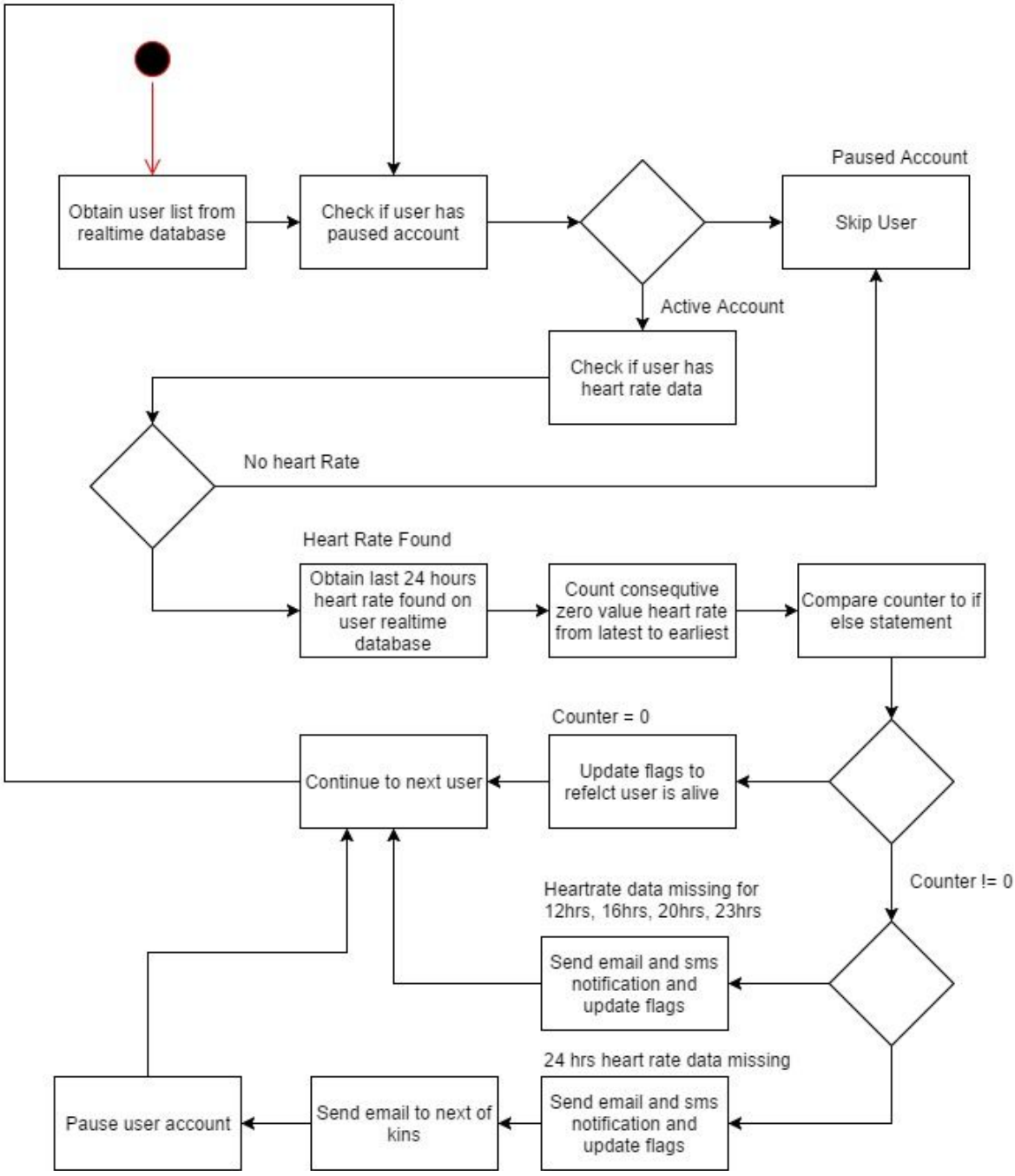


1.

retrieveHeartRate.py flows:

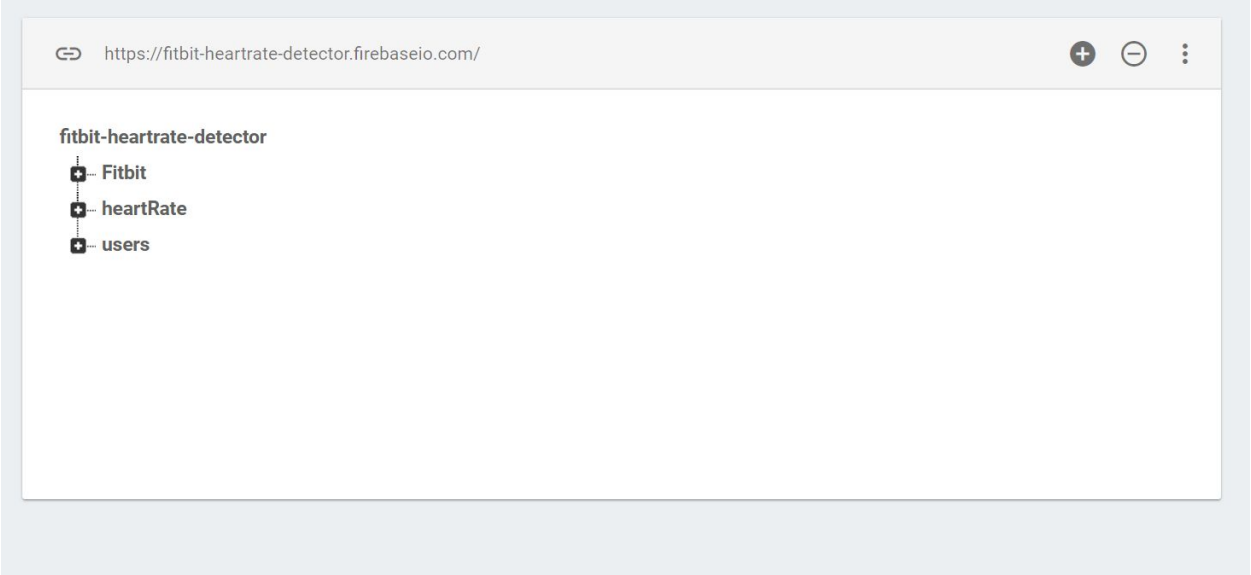


processHeartRate.py flows:



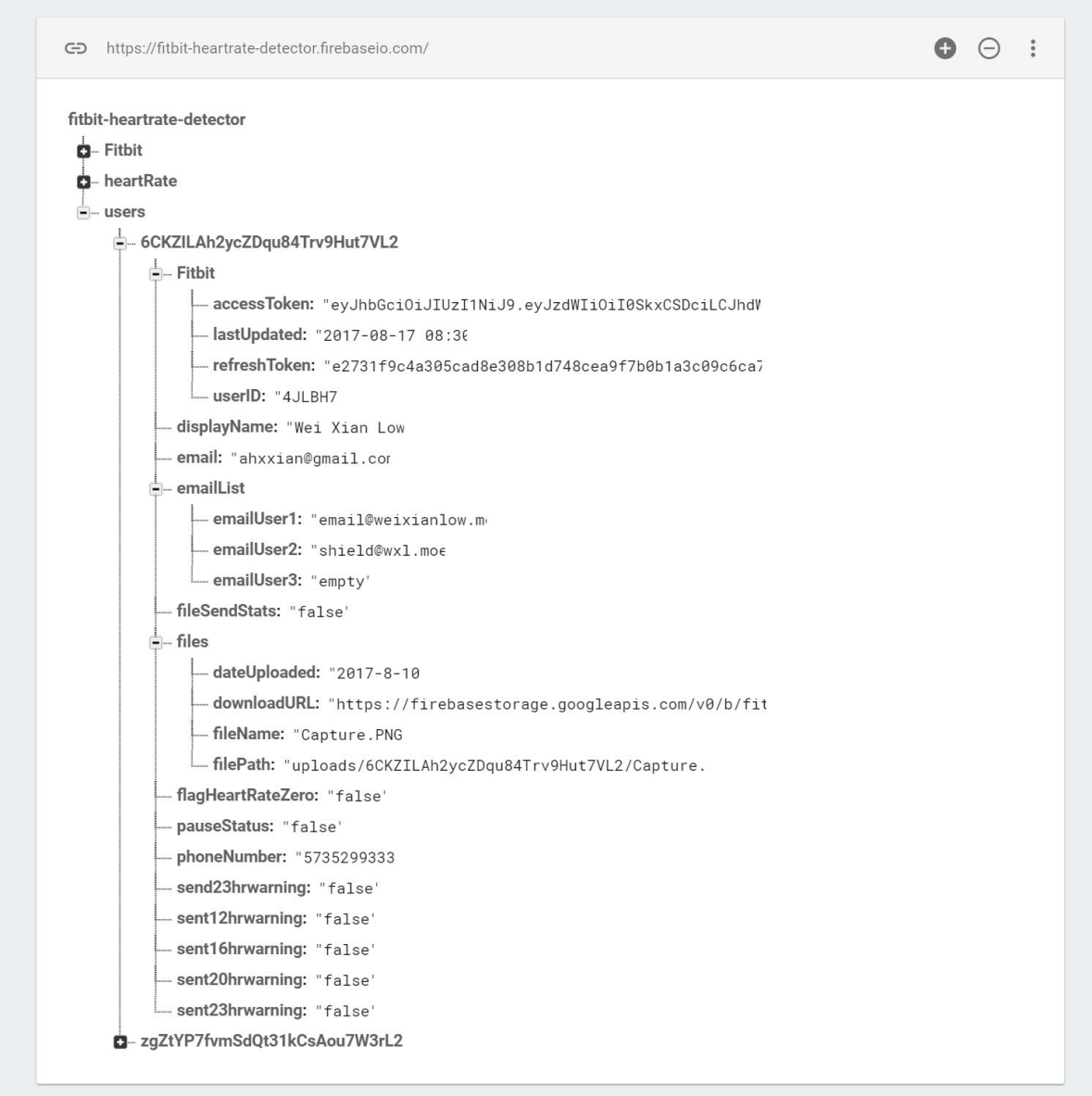
Database Design:

1. General Design:



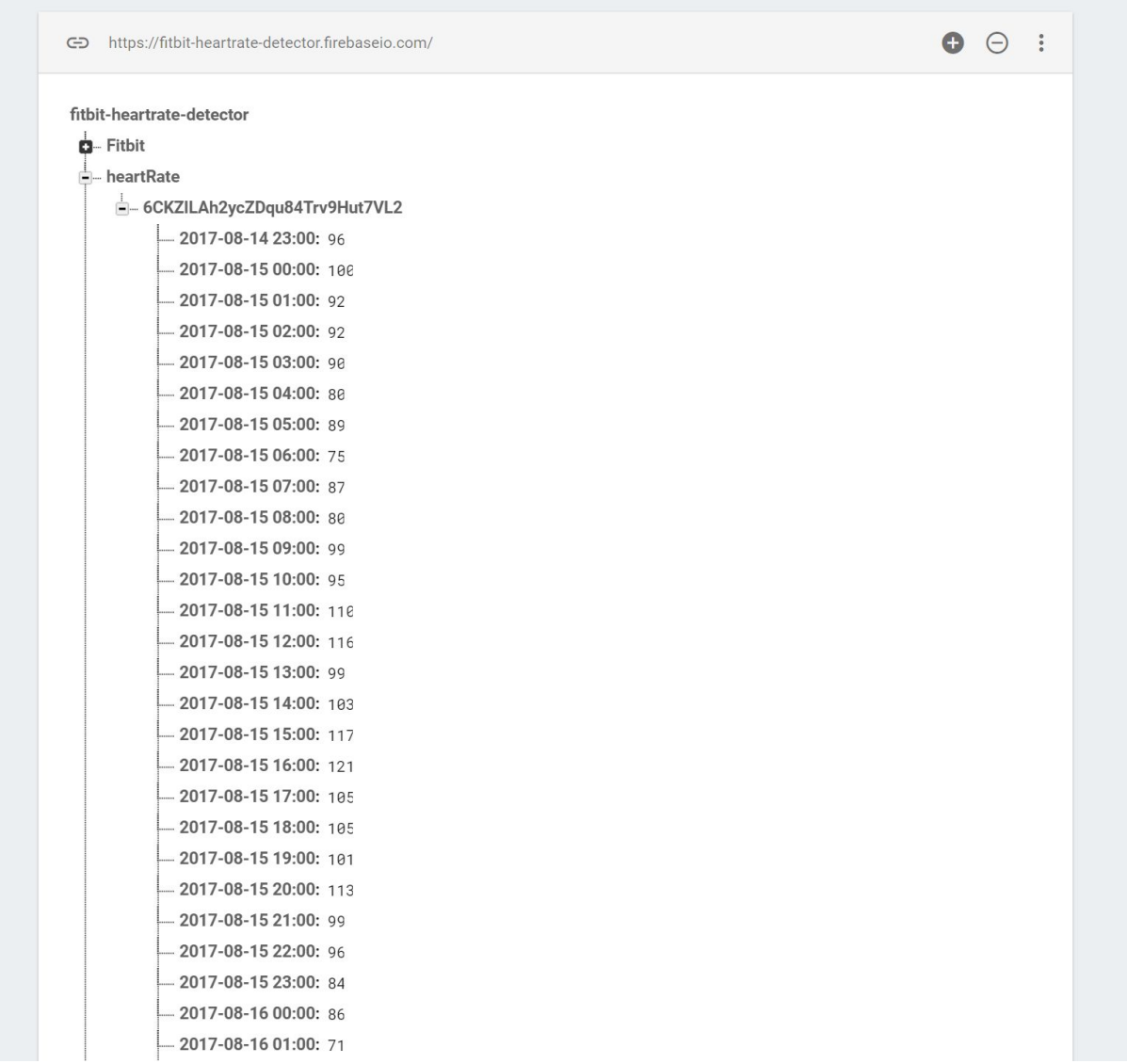
- The Realtime Database will be separated into three different category, Fitbit, heartRate, and users.
- Fitbit contains Fitbit's API Client ID and Client Secret, required when authenticating with Fitbit's API.
- heartRate contains a list of user UID and their corresponding heartRate Data.
- users contains a list of user UID and their corresponding personal information

2. Users:



- With this setup, user's information will be isolated from the heartrate data, making it easier to manipulate and process heart rate data without affecting sensitive info such as user's personal info and user's Fitbit access code.

3. Heart Rates



- With an entry based collection of data, this would be very functional with the command used in the Admin SDK with python.
- We are able to obtain the last 24 entries of heart rate data while having the data sorted.
- When a user paused and reactivate their account, their old heart rate data will be deleted to prevent redundant data to be processed.

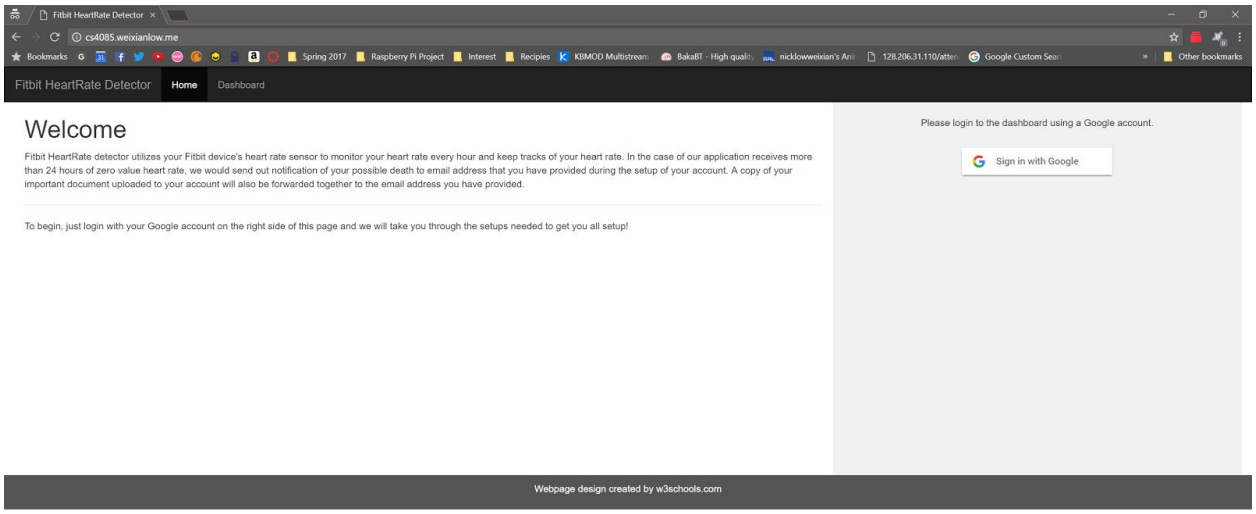
Scripts/Code:

1. buffer.js
    - This script is used to identify if user has authenticated with the website before. If the user has, it will redirect the user to the dashboard, if not, it will attempt to create a new entry in the database and populating the info obtained from the user's google account.
2. firebaseConfig.js
    - This script is used to configure the necessary credentials needed for javascript access to firebase services
3. fitbitProcess.js
    - This script is used to obtain the authorization code provided from Fitbit's authentication page and process it to obtain user's access code and refresh code.
4. updatePhoneNumber.js
    - This script is used to prompt user to provide a valid phone number to the services to enable sms notification.
5. dashboard/dashboard.js
    - This script is used to display user's data on the dashboard page. This script also provide functionalities such as pausing an account, and updating user's personal information.
6. dashboard/updateEmailList.js
    - This script is used to update user's next of kin email addresses to the profile.

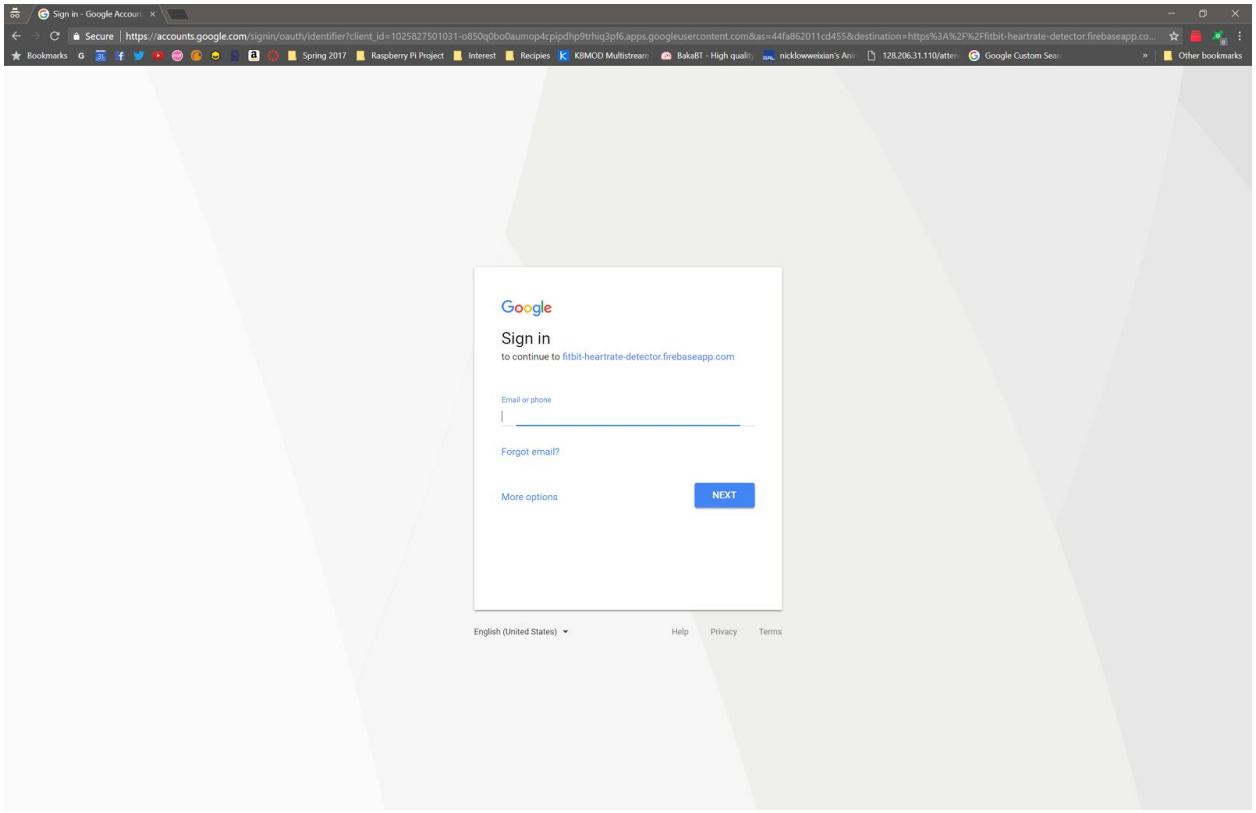
7. dashboard/upload.js
    - This script is used to hand file upload process to Google firebase storage and updating the storage information to user's realtime database
8. retrieveHeartRate.py
    - This is a back end script used to obtain user's hourly heart rate data from Fitbit's API. This script is executed twice an hour by crontab.
9. processHeartRate.py
    - This is a back end script used to process user's heart rate data stored in the realtime database. This script is executed once every hour by crontab.
10. Mailin.py
    - This is the library files provided by MailInBlue to support email sending through their SMTP protocol.

Screenshots

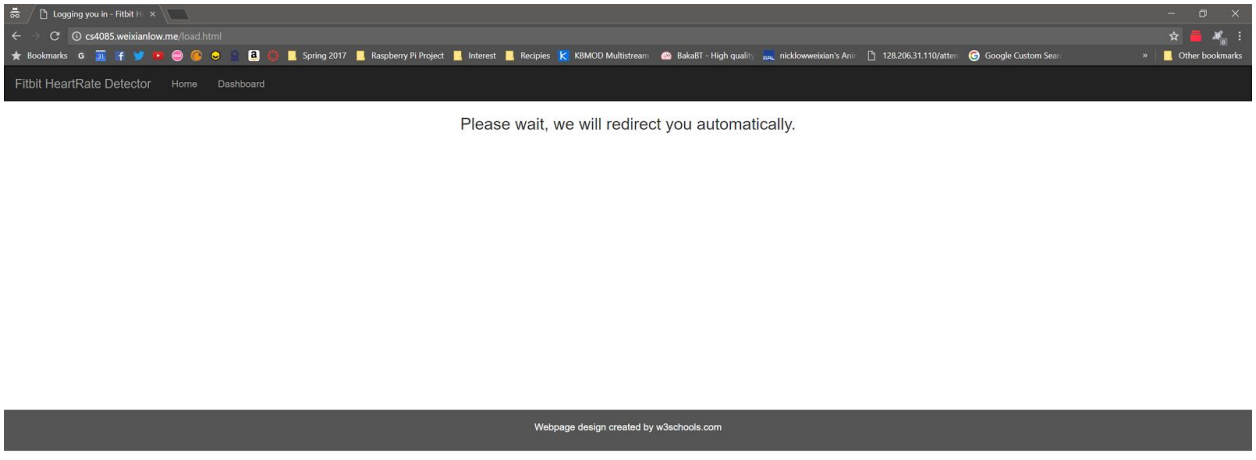
Homepage:



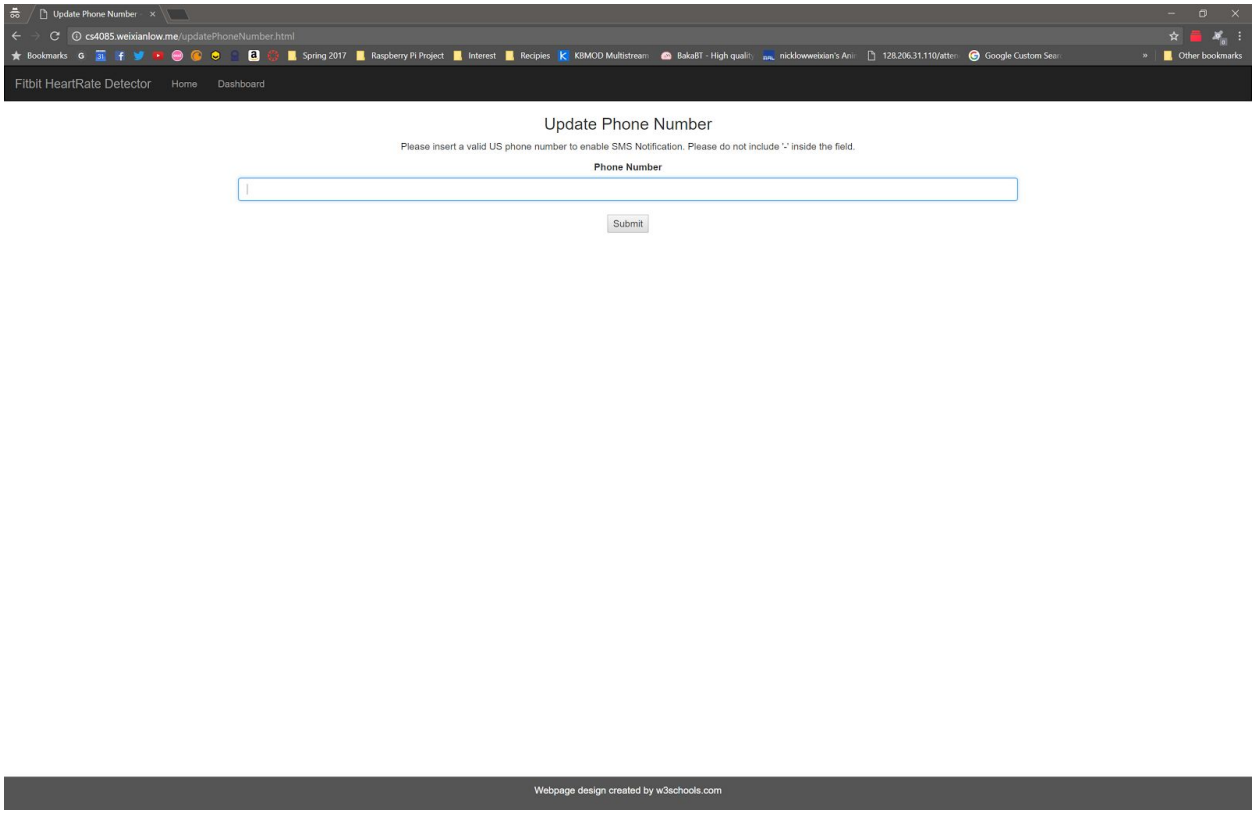
Google Authentication Page:



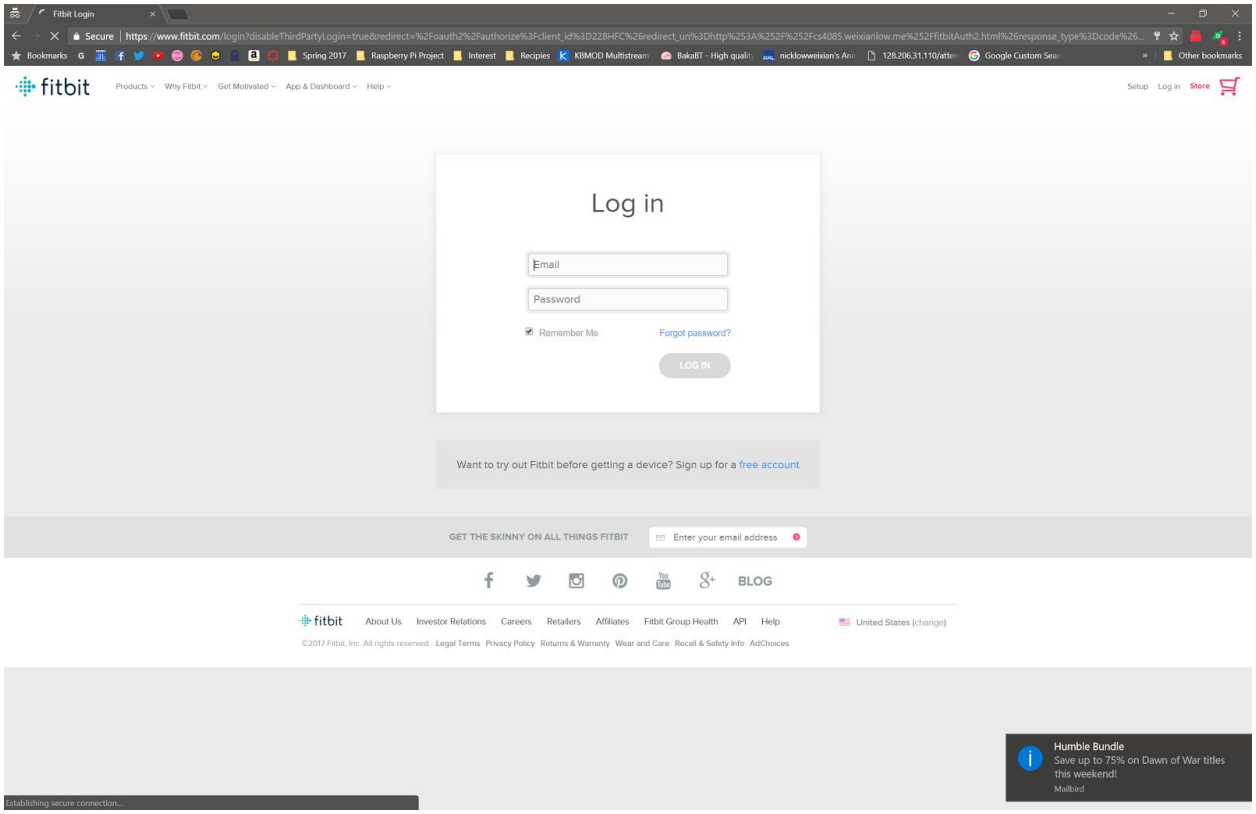
Buffer page:



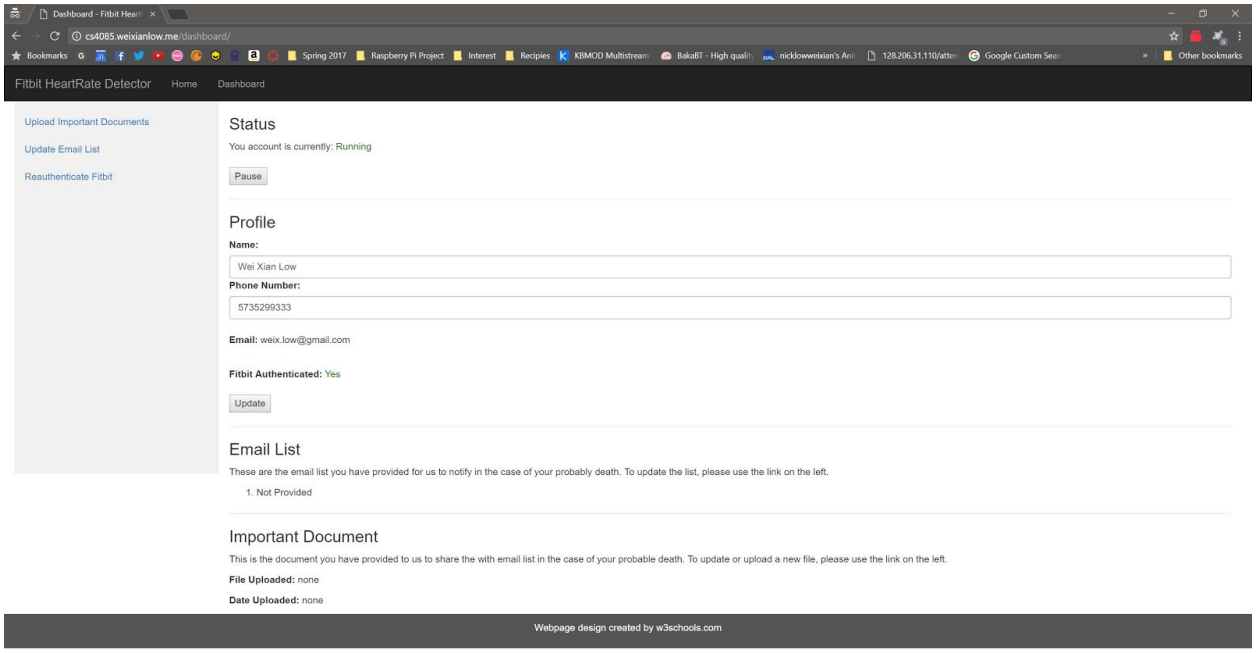
Update Phone Number Page:



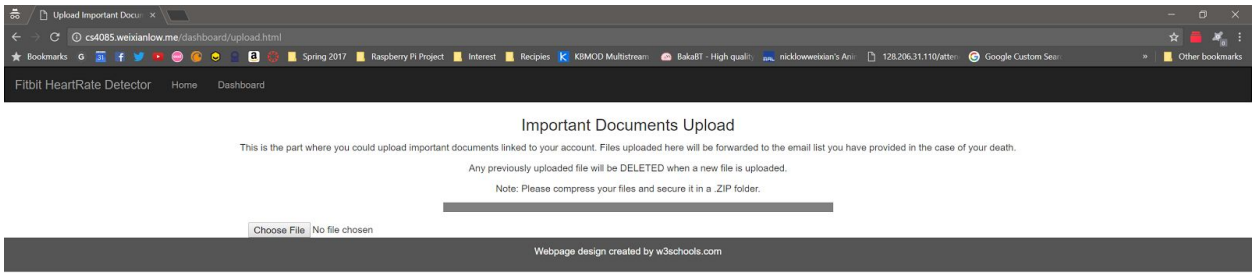
Fitbit Authentication Page:



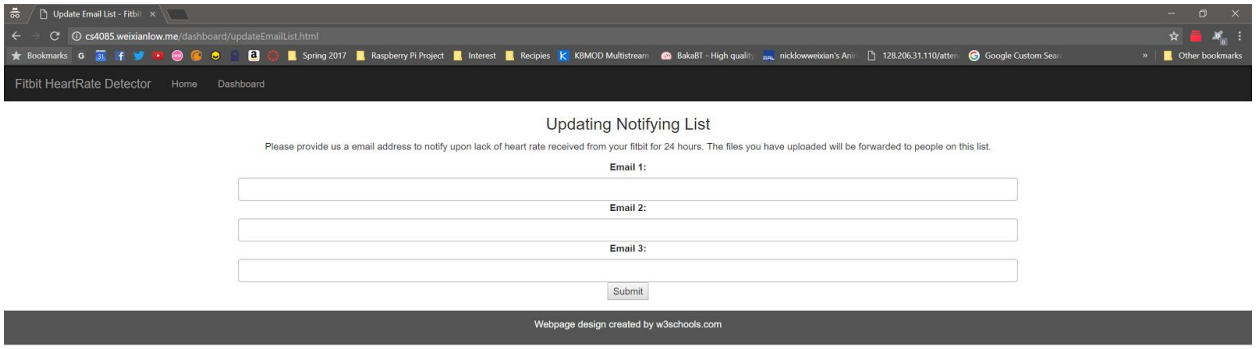
Dashboard:



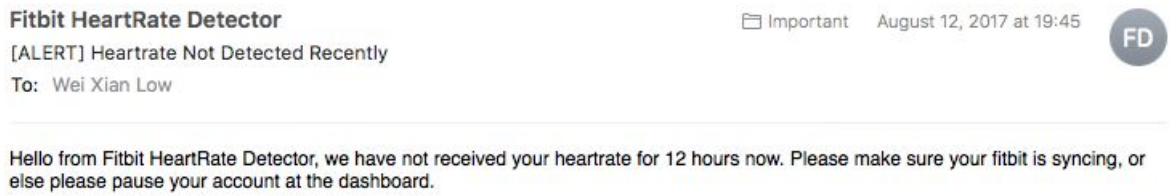
Upload File Page:



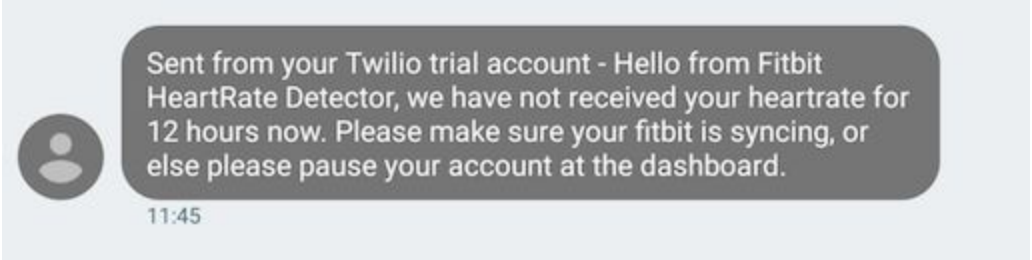
Update Next Of Kin Email List:



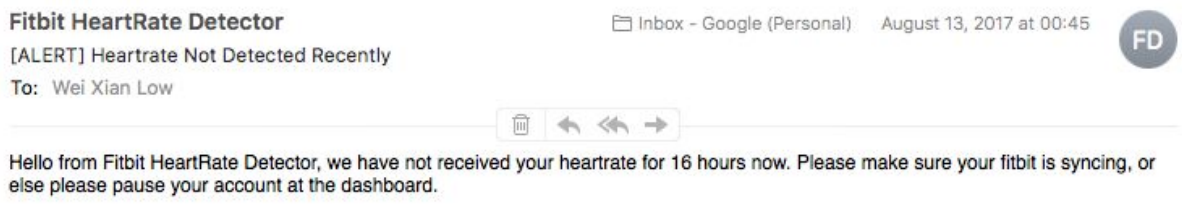
12 Hours Email Notice:



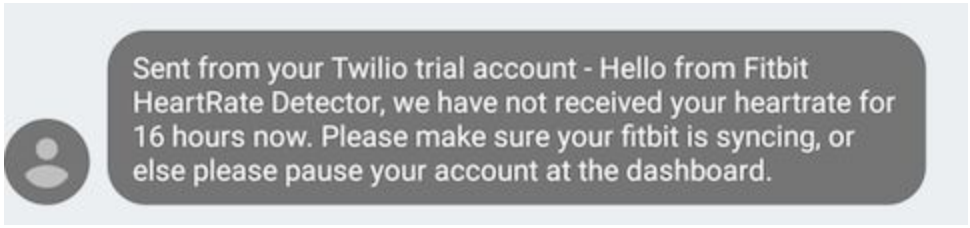
12 Hours SMS Notice:



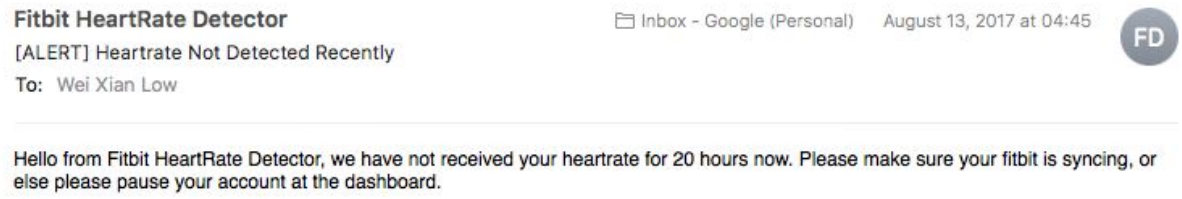
16 Hours Email notice:



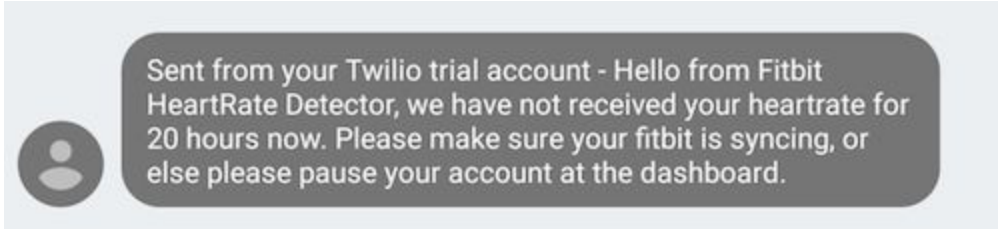
16 Hours SMS Notice:



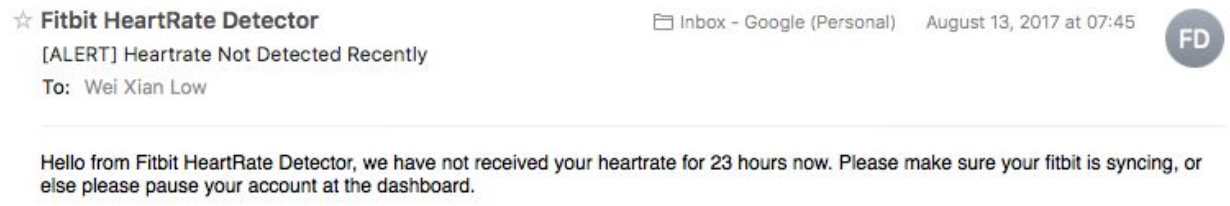
20 Hours Email Notice:



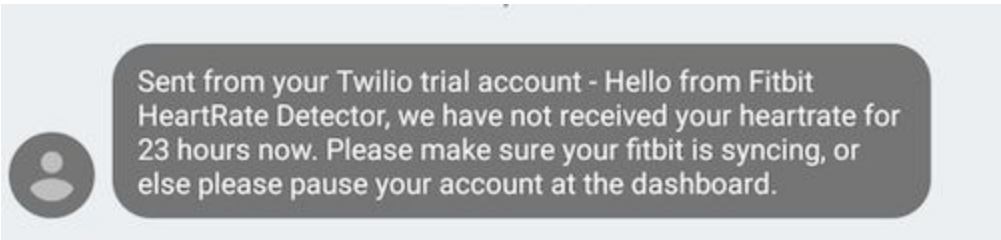
20 Hours SMS Notice:



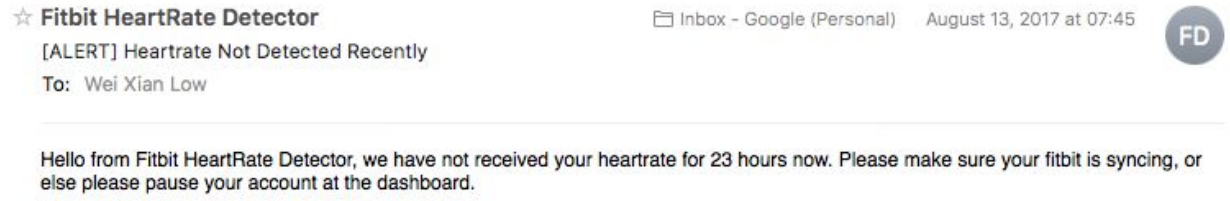
23 Hours Email Notice:



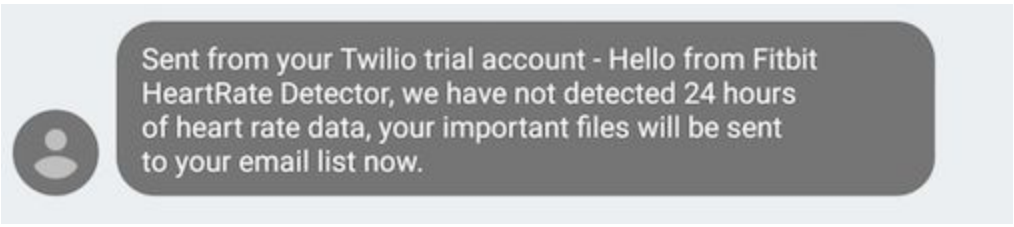
23 Hours SMS Notice:



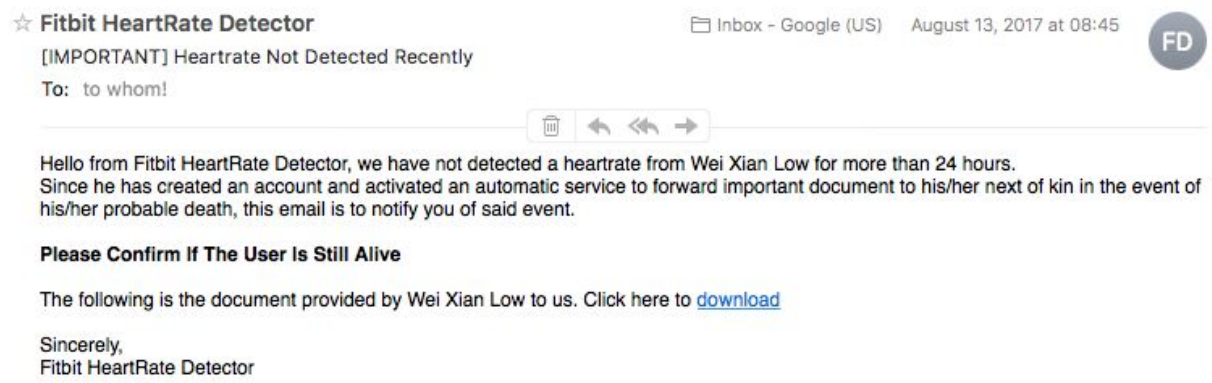
24 Hours Email Notice:



24 Hours SMS Notice:



Next Of Kin Email Notice:



☆ **Fitbit HeartRate Detector** Inbox - Google (US) August 13, 2017 at 08:45 FD

[IMPORTANT] Heartrate Not Detected Recently

To: to whom!

Hello from Fitbit HeartRate Detector, we have not detected a heartrate from Wei Xian Low for more than 24 hours.
Since he has created an account and activated an automatic service to forward important document to his/her next of kin in the event of his/her probable death, this email is to notify you of said event.

Please Confirm If The User Is Still Alive

The following is the document provided by Wei Xian Low to us. Click here to download

Sincerely,
Fitbit HeartRate Detector

Conclusion

Problems Encountered:

1. Fitbit Authentication:
    a. For the service to work, the service is required to obtain a user's hourly heart rate information from the user's Fitbit account.
    b. However, the Fitbit has imposed a restriction on how a developer is able to obtain user's hourly heart rate data by requiring developers to declare that their user requires the need of it.
    c. Developers are only allowed to obtain developer's own heart rate data that is tied to the secret ID given by Fitbit for the program for development purposes. In order to obtain another user's heart rate data, fitbit is required to review a developers program and grant permission on a case by case basis.
    d. I'm unable to obtain said permission in time due to the short time frame of this project has been given to me and hence I have only able to test the system with multiple account linked to the same fitbit account.
2. Fitbit delay of data:
    a. Fitbit has a 30 minute delay in obtaining data from user's devices, making it slightly unreliable to measure user's heart rate data for the purpose of this project
    b. To overcome this, the script "retrieveHeartRate.py" is run twice an hour to ensure a new updated data is obtained for the same time, preventing false alarm of system.

Future Improvements:

1. Improved Dashboard:
    a. An improved dashboard could be built to show user the last recent obtained data from the script to show that the program is running flawlessly.
    b. The dashboard would also be able to just execute the script manually without waiting for hourly batch to run.
2. Improved script
    a. Better error handling could be placed on the script to handle unexpected error.
    b. Could be used to better improve notification on either an admin or the user if attention is needed.
3. Error checking:
    a. An improved error checking could be implemented to ensure that data provided by user wouldn't cause error that would prevent a script to misbehave.

Lesson Learnt:

1. Official documentation and APIs:
    a. The ability to implement and use a system's official API is very useful in developing a program. Without relying a third party service could greatly enhance the reliability of the service, without depending on a third party developer to push and update to fix bugs.

2. Development languages:
    a. A better planning in deciding which programing languages to develop in is important.
    b. By reading ahead the documentation of services needed for the project to be developed is crucial to identify the limitation of the programming languages chosen.
3. OAuth 2.0
    a. A new secure authentication protocol used by APIs to ensure a client's secret is properly encrypted by the server to communicate with third parties' API.
    b. A better understanding of the protocol is learnt during the development of this project.

Remarks:

During the development of this project, I have encountered numerous new technologies that has been a stranger to me. However, with the reliability and abundance of documentation written and provided by the developers, it makes it easy to learn how to use new technology and incorporate new functionality into older codes.

I have also learnt a lot about security measurements in handling user's heart rate data as protocols and security measurements are put in place to prevent leaks of data to the outside world, as these data are sensitive.

I would like to thank my instructor to have given me this opportunity to develop a program that was once an idea, now into a reality.