*Part I – The Reasons & the Equipment:*

For this project we were asked to make something that would help people in the field of sports and leisure using a Raspberry Pi. After much deliberation we figured that a bicycle with a Display-O-Tron 3000, HC-SR04 ultrasonic sensor and reed switch would make the perfect bike computer. This would help people cycle more safely as it has left & right turn indicators so that cars can understand if the cyclist is turning and will wait. The bike also helps the rider be more aware of his/her surroundings (by knowing if a car is close behind him/her). The distance sensing feature also helps the rider know how far he/she has travelled and his/her average speed throughout the ride. By adding some headphones into the mix we can even listen to songs as we ride (even though the current code prevents it from playing a playlist.

*Part II – The Hardware:*

We decided that we would use a reed switch along with a group of magnet mounted on the wheel to count how many rotations the wheel made, and then calculated the circumference of the wheel. Since every time the wheel rotates the bike would have gone *C* meters farther we could figure out how far the bike had travelled by measuring the amount of time the reed switch was closed and every time that occurred we could increment the distance travelled. This was shown using some text on the display. We used the ultrasonic sensor to warn the cyclist if a car got very near (i.e. nearer than 40 cm) to the bicycle. This information is outputted on the display by changing the backlight colour from green to red, we chose to do this as a colour change is simple to quickly comprehend, also if you wish to how close the car is, you can look at the LED bar graph which shows the distance of the car in centimetres.

*Part III – The Software:*

We were originally going to use the following modules as well as the Python language:

* Tweepy (Twitter API for Python)
* Mpylayer(To play the music using mplayer music and movie player for linux)

However we ended up using the following modules

* Dot3k (To output information to the display)
* RPi.GPIO(For reading/writing the state of the GPIO pins on the Pi
* Math (For an accurate value of ∏)
* Time (To measure time)
* Mpylayer (Same reason as above, but it is only currently in a bunch of functions that are called once before the start of the loop, thus playing only one song for a short while)
* Random(To randomize the songs that are played)

*Part IV – The Flaws & Problems:*

Problems we encountered:

* When using Tweepy we found that the raspberry pi had no True SSL certificate so we could not authorise ourselves on twitter.
* We couldn’t understand how to play a list of songs using mypylayer, so that didn’t work.

Flaws:

* It would not improve a cyclist’s mood in a traffic jam as the backlight would stay red and red is not a pleasing colour especially when you are late to work.
* There is only a short window for the reed switch to be detected as “closed”, thus, most of the time the switch being closed is not detected, thus not accurately detecting the distance travelled as well as the average speed.

*Part V – Credits:*

<https://electrosome.com/hc-sr04-ultrasonic-sensor-raspberry-pi/> - ultrasonic sensor code – slightly adapted

pinout.xyz – Raspberry Pi Pinout

Stack Exchange – not sure of the link