



IoT Full-stack
development:

Bicycle Crash Notifier

Paola Rodrigues
Trek Bikes -IoT Trainee





TREK



BONTRAGER



Assignments

e-Bike drive Unit Data
(speed, torque, battery level)

Develop an application



Crash Notifier for Bicycles

- **Reliable**
- **Generic use**
- **Low Cost**
- **Battery powered**
- **Wireless technology**



IoT Technology Stack



es

MCU?

ware Setup

Firmware Setup (sleep routines, ISR...)

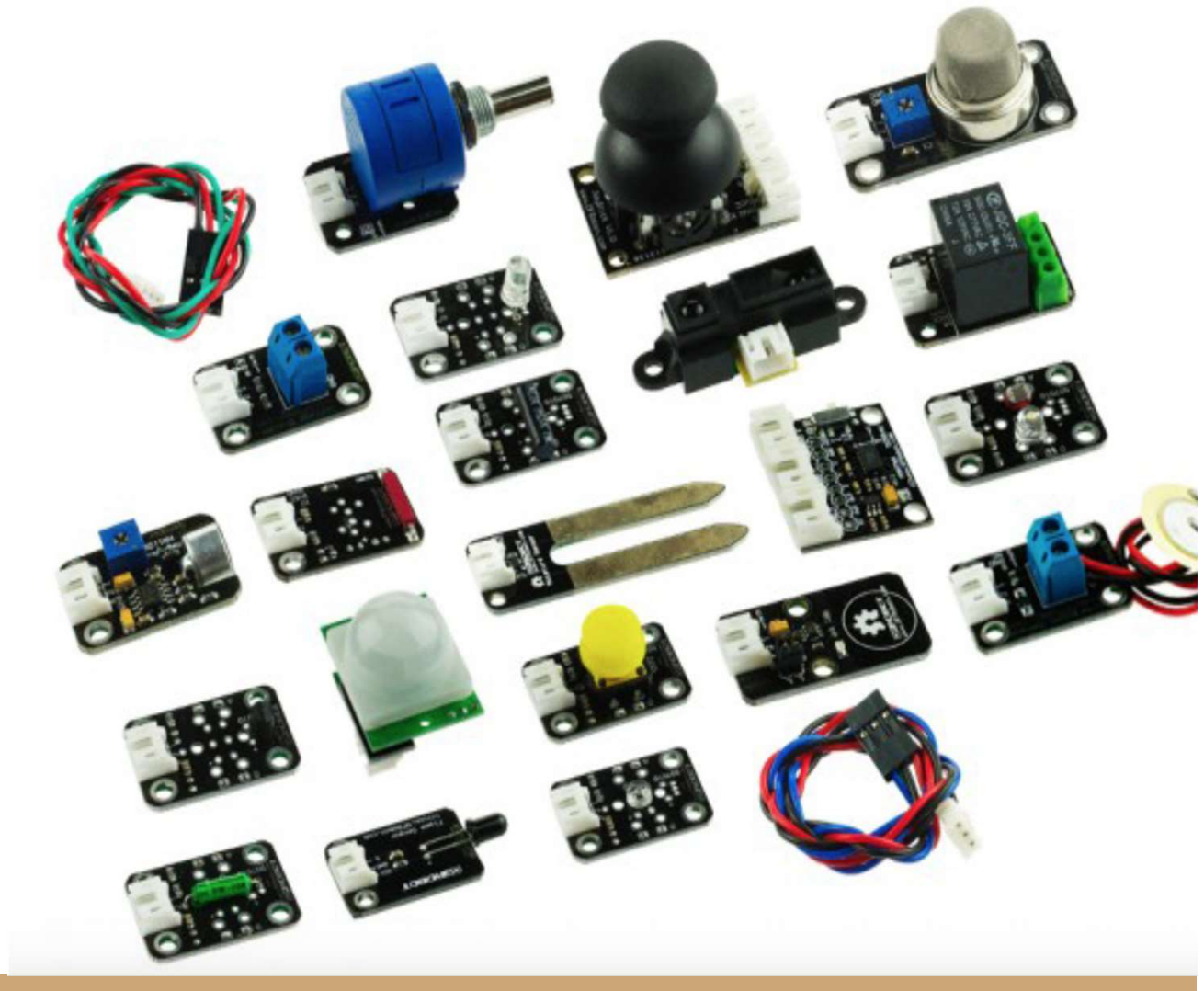
Network Servers

Application Servers

Cost



Sensors

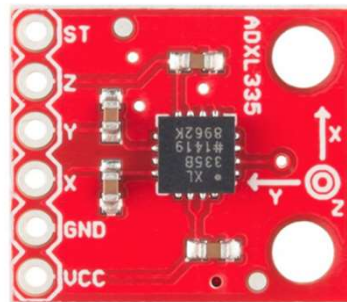
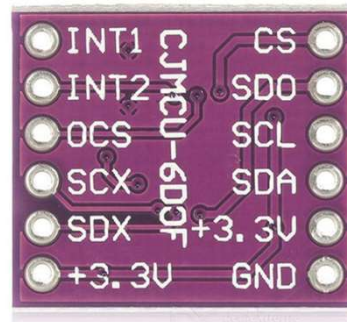


Accelerometer



MEASURES:

- Acceleration
- Angle or Tilt sensing
- Motion
- Free Fall



CHARACTERISTICS

Range: $\pm 1g$ up to $\pm 250g$

Interface:

- Analogic - PWM
- Digital- I2C or SPI

Additional features - selectable measurement ranges, sleep control, 0-g detection, and tap sensing...

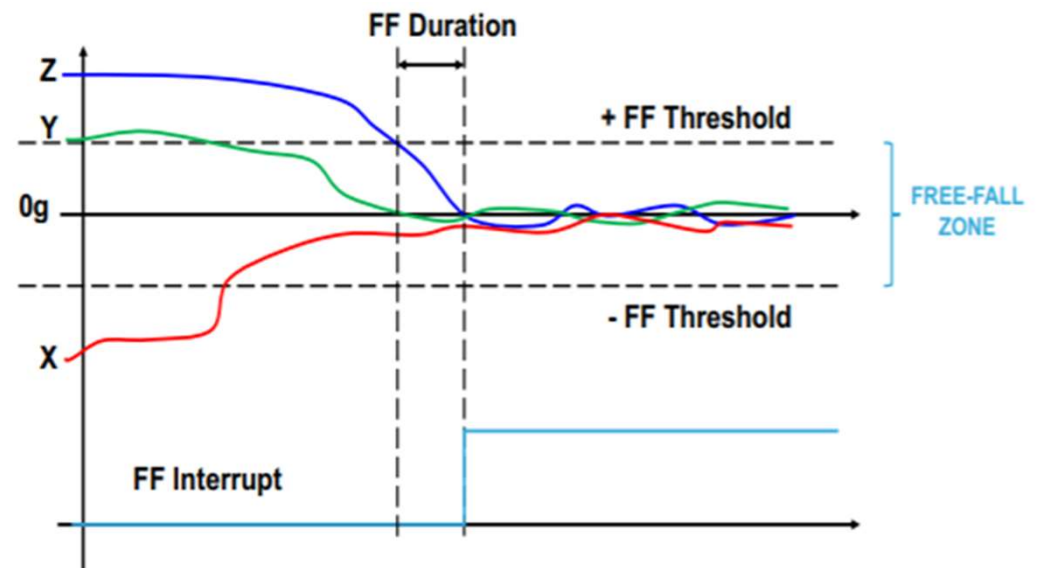
IMU -Accelerometer + Gyroscope (LSM6DS3)

FREE FALL CONFIGURATION

- I2C Interface
- 3 axis
- Full Scale (FS) = ± 2 g
- Threshold = ± 312 mg ((x,y and z axis)
- FF duration time = 15 msec
- Output Data Rate (ODR) = 416Hz

Read register to count Free Fall

```
if ( readDataByte){  
    detectCount ++;}  
}
```

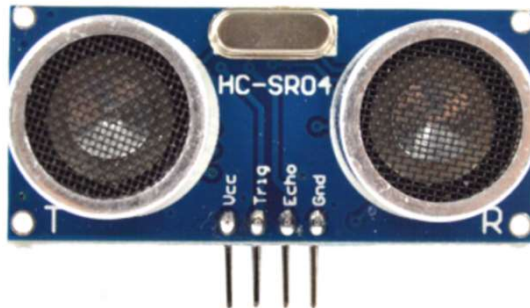


Ultrasonic Range Finder (HC-SR04)

MEASURES:

- Distance measurements (up to 4m)
- Object detection, counting
- Level monitoring

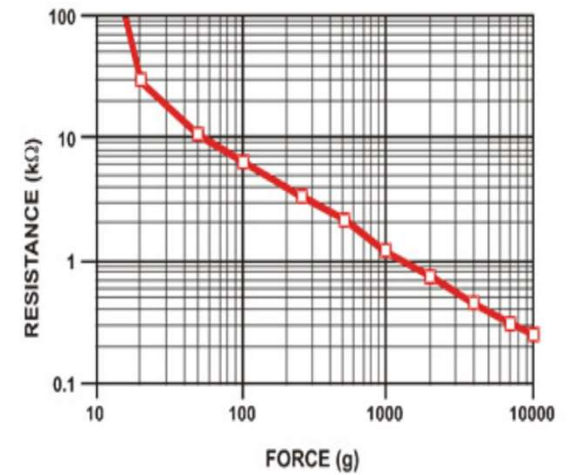
Safety distance > 70 cm



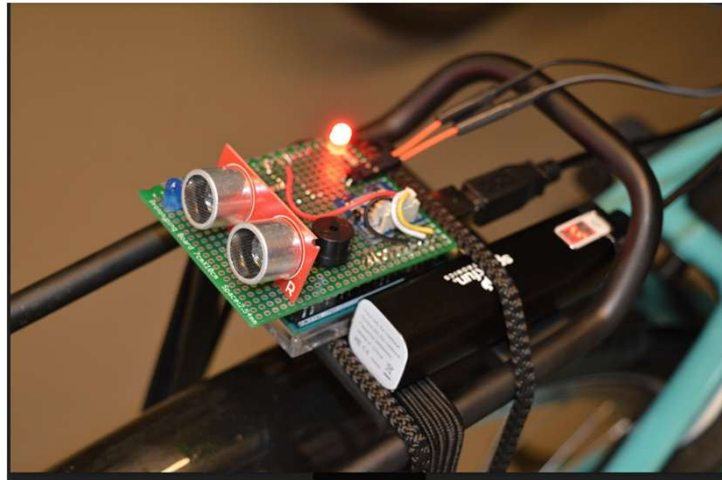
Force Sensing Resistor (FSR-402)

MEASURES:

- Detects physical pressure varying the resistance
- Qualitative Sensor (0 up to 255)
- Analog Low Cost Sensor



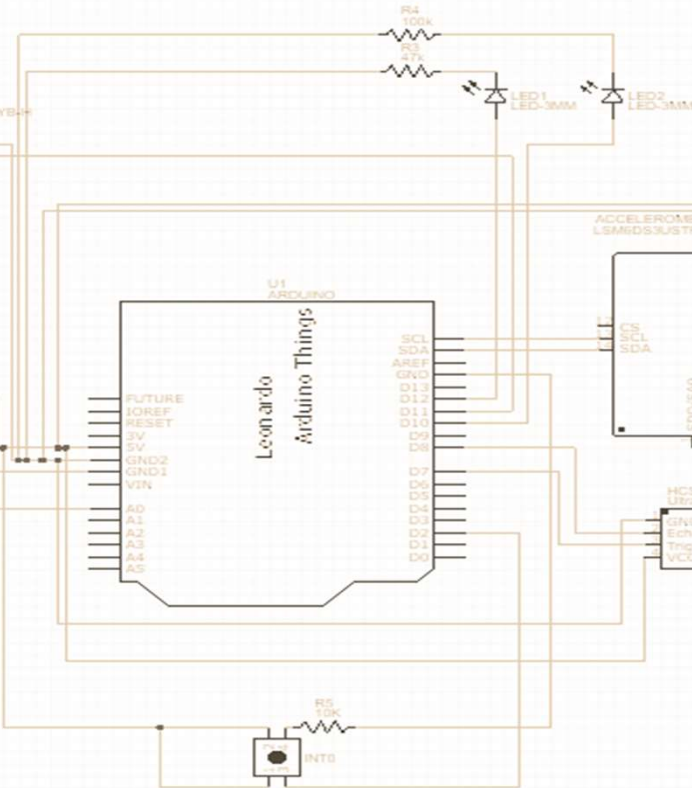
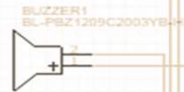
Prototype #1





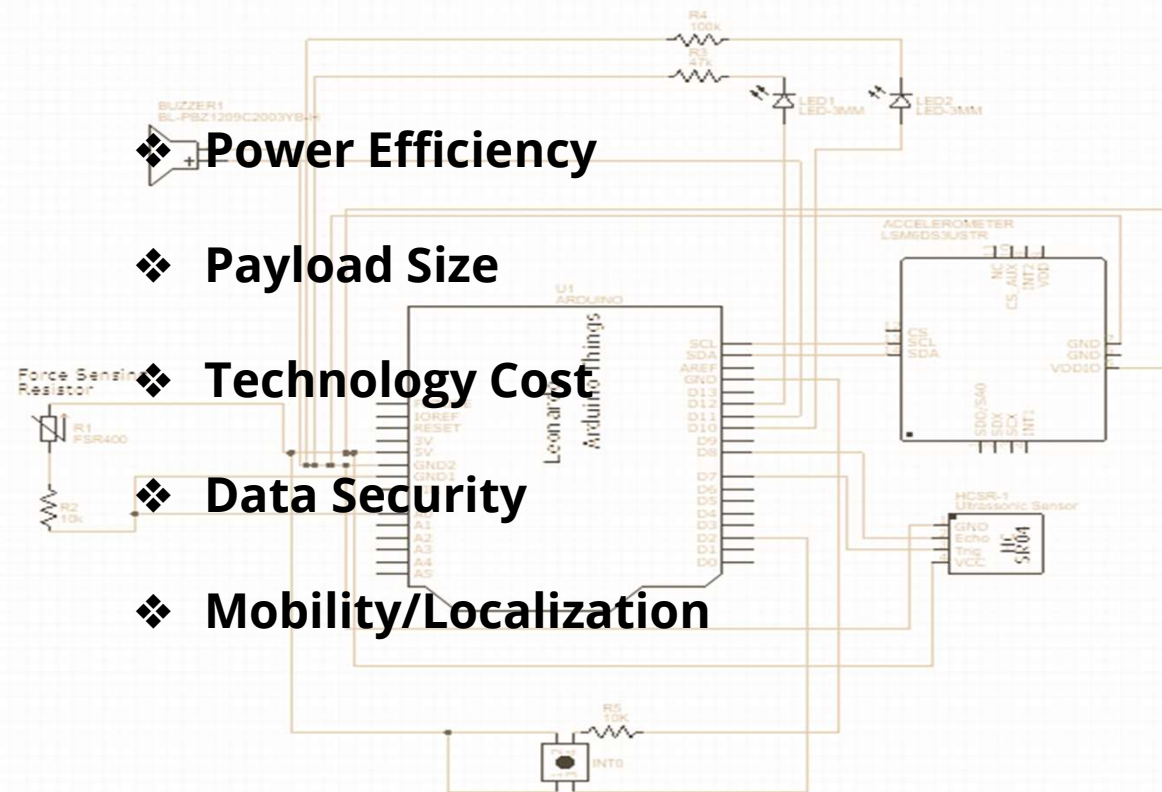
Connectivity

Force Sensing Resistor



TITLE: LoRa Crash Notifier		REV: 2.2
Date: 2019-02-03	Sheet: 1/1	
EasyEDA V5.8.22	Drawn By: paolesar	

Connectivity



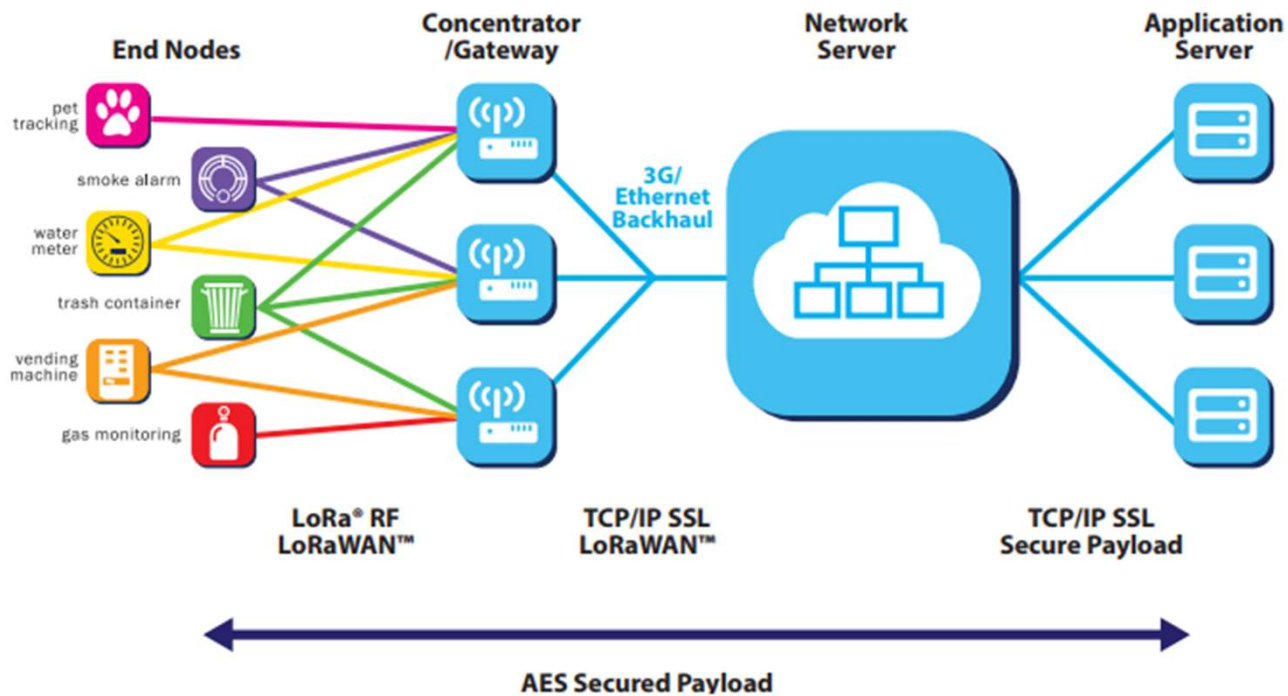
- ❖ Power Efficiency
- ❖ Payload Size
- ❖ Technology Cost
- ❖ Data Security
- ❖ Mobility/Localization

TITLE: LoRa Crash Notifier		REV: 2.2
Date: 2019-02-03	Sheet: 1/1	
EasyEDA V5.8.22	Drawn By: peoliasr	

LPWAN Wireless Technologies

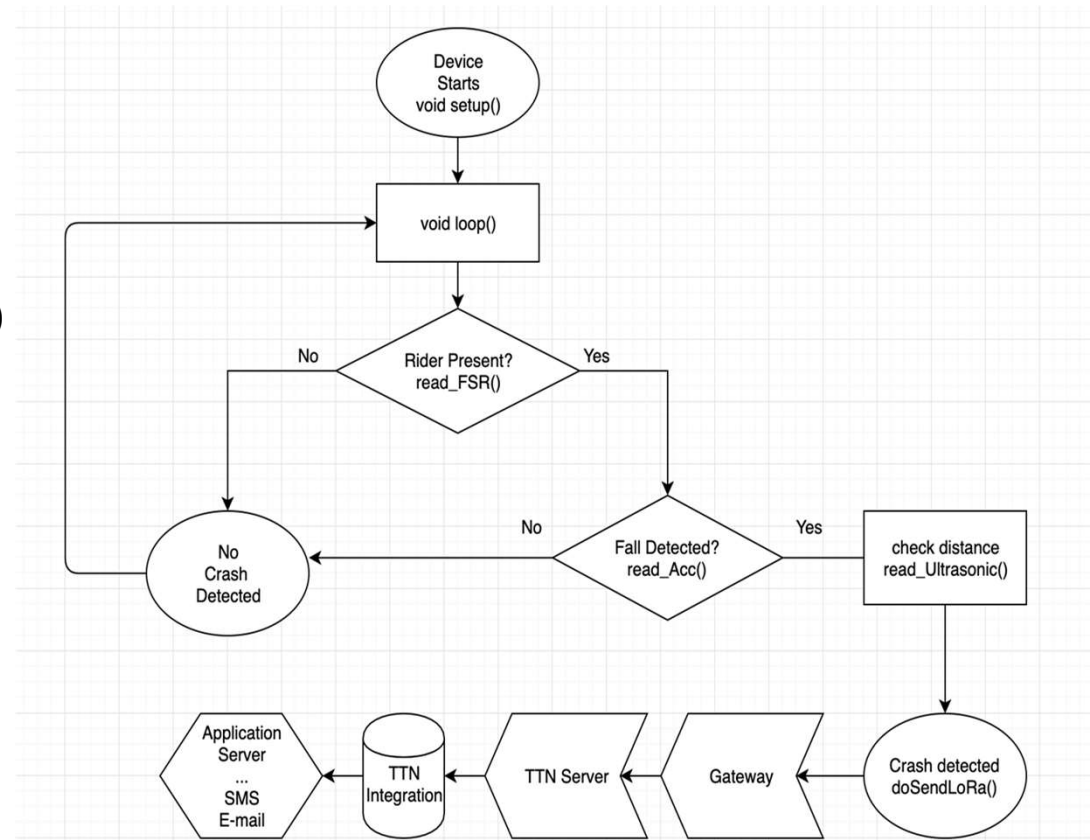
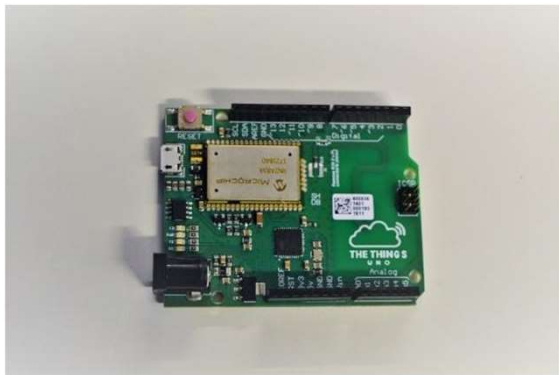
Feature	LoRaWAN	Narrow-Band	LTE Cat-1 2016 (Rel12)	LTE Cat-M 2018 (Rel13)	NB-LTE 2019(Rel13+)
Modulation	SS Chirp	UNB / GFSK/BPSK	OFDMA	OFDMA	OFDMA
Rx bandwidth	500 - 125 KHz	100 Hz	20 MHz	20 - 1.4 MHz	200 KHz
Data Rate	290bps - 50Kbps	100 bit/sec 12 / 8 bytes Max	10 Mbit/sec	200kbps – 1Mbps	~20K bit/sec
Max. # Msgs/day	Unlimited	UL: 140 msgs/day	Unlimited	Unlimited	Unlimited
Max Output Power	20 dBm	20 dBm	23 - 46 dBm	23/30 dBm	20 dBm
Link Budget	154 dB	151 dB	130 dB+	146 dB	150 dB
Batery lifetime - 2000mAh	105 months	90 months		18 months	
Power Efficiency	Very High	Very High	Low	Medium	Med high
Interference immunity	Very high	Low	Medium	Medium	Low
Coexistence	Yes	No	Yes	Yes	No
Security	Yes	No	Yes	Yes	Yes
Mobility / localization	Yes	Limited mobility, No loc	Mobility	Mobility	Limited Mobility No Loc

LoRaWAN Architecture



Prototype #1

- CPU: ARDUINO THINGS UNO
- ATmega32u4 (16MHz)
- LoRa: RN2483 (class A - 868 MHz)
- Arduino IDE compatible



LoRa Server: The Things Network



Applications > bike-accident-notifier > Devices > thingsuno > Data

OverviewDataSettings

APPLICATION DATA

pause clear

uplinkdownlinkactivationackerror

Filters

	time	counter	port	
▲	09:55:56	3	1	payload: A400 03 76 distance: 164 freeFallCount: 3 presence: 118
▲	09:51:44	2	1	payload: 5D00 02 91 distance: 93 freeFallCount: 2 presence: 145
▲	09:48:36	1	1	payload: 0800 01 86 distance: 8 freeFallCount: 1 presence: 134

COMPS (Arduino/Genuino Uno)

No accident detected
Rider is present 137
No accident detected
Rider is present 135
FALL DETECTED!1
Rider is present 134
ALERT! Object is too close to rear wheel (cm): 8
Oops, accident detected!
No rider
No accident detected
No rider
No accident detected
Rider is present 149
No accident detected
Rider is present 144
No accident detected
Rider is present 146
FALL DETECTED!2
Rider is present 145
Safe distance from rear side
Oops, accident detected!
No rider
No accident detected
No rider
No accident detected
No rider
No accident detected


☐ Autoscroll ☐ Show timestamp


Newline


Integrations

```
server.js  x  index.html
1  console.log("Starting server...")
2
3  const app = require("express")();
4  const http = require("http").Server(app);
5  const bodyParser = require("body-parser");
6
7  const deviceId = "26012339"
8
9  app.use(bodyParser.json())
10 app.use(bodyParser.urlencoded({ extended: false }))
11
12 app.get("/", function(req, res){
13   res.sendFile(__dirname + "/index.html")
14 })
15
16 app.post("/endpoint", function(req, res) {
17   console.log(req.body)
18   res.sendFile(__dirname + "/index.html")
19 })
20
21 http.listen(8000);
```


Integration to Application Server

 **THE THINGS NETWORK** **CONSOLE**
COMMUNITY EDITION


ApplicationsGatewaysSupport p.


Applications >  bike-accident-notifier > Integrations


OverviewDevicesPayload FormatsIntegrationsDataSettings

INTEGRATIONS


+ add integration

 AllThingsTalk Maker


 MyDevices

 IFTTT Maker

crash-notifier

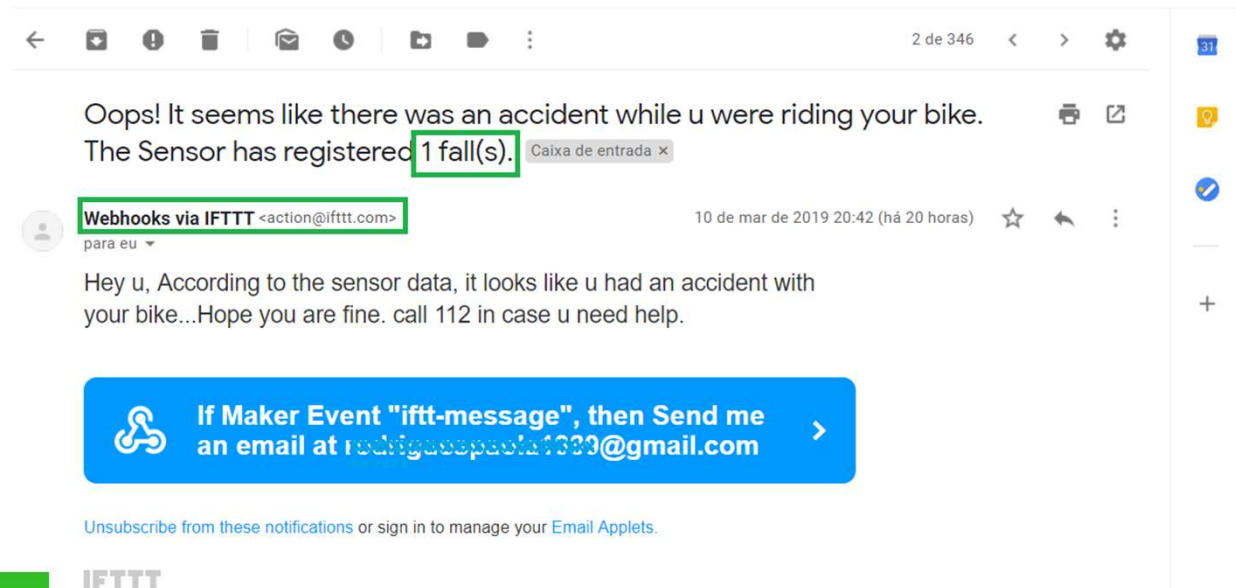
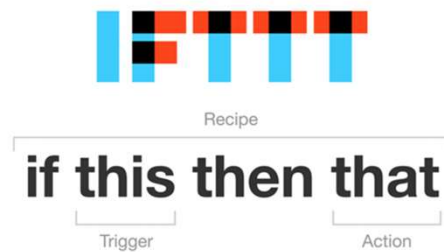
 HTTP Integration

bike-http-test

 Data Storage

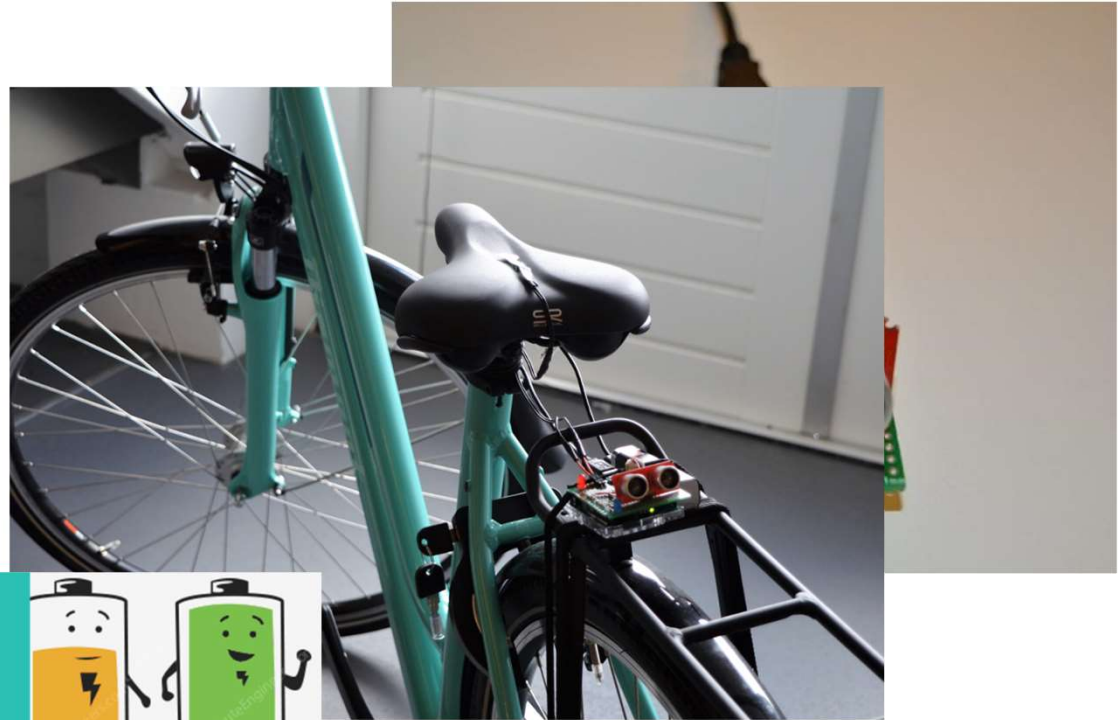
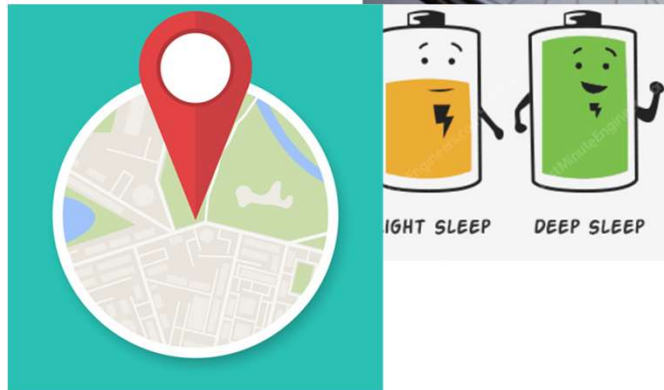
main

Integration - Application Server



Feedback

- **Design**
- **Power consumption**
- **LoRa Geolocation**



Alternative: Saddle+Device

Smart Saddle

Seat post



Improvements

```
#ifdef DEBUG
Serial.println("Safe distance from rear side");
#endif
}
return distance;
}
/*
Free Fall detector settings
*/
```

```
uint16_t detectCount = 0;
uint8_t error = 0; // accumulation variable
uint8_t dataToWrite = 0;
int config_free_fall_detect(void)
{
    dataToWrite |= LSM6DS3_ACC_GYRO_BW_XL_200Hz; //Anti-aliasing filter bandwidth selected = 200hz
    dataToWrite |= LSM6DS3_ACC_GYRO_FS_XL_2g; //Accelerometer full-scale selected = 2g
    dataToWrite |= LSM6DS3_ACC_GYRO_ODR_XL_416Hz; // Output data rate and power mode = 416Hz high performance

    //Writing the accumulated data in the error accumulation variable
    error += lsm6ds3.writeRegister(LSM6DS3_ACC_GYRO_CTRL1_XL, dataToWrite); //REGISTER 1 = Linear acceleration sensor control
    error += lsm6ds3.writeRegister(LSM6DS3_ACC_GYRO_WAKE_UP_DUR, 0x00); //Writes 00h into WAKE_UP_DUR
    error += lsm6ds3.writeRegister(LSM6DS3_ACC_GYRO_FREE_FALL, 0x33); //writes 33h into free fall
    error += lsm6ds3.writeRegister(LSM6DS3_ACC_GYRO_MD1_CFG, 0x10); //writes 10h into MD1
    error += lsm6ds3.writeRegister(LSM6DS3_ACC_GYRO_MD2_CFG, 0x10); //writes 10h into MD2
    error += lsm6ds3.writeRegister(LSM6DS3_ACC_GYRO_TAP_CFG1, 0x01); //latch interrupt and writes 01h into TAP_CFG1
}

/*
Measure free falls
*/

uint16_t Freefallcounter(bool) {

    uint8_t readDataByte = 0;
    //Read the wake-up source register
    lsm6ds3.readRegister(&readDataByte, LSM6DS3_ACC_GYRO_WAKE_UP_SRC);
    //Mask off the FF_IA bit for free-fall detection
```

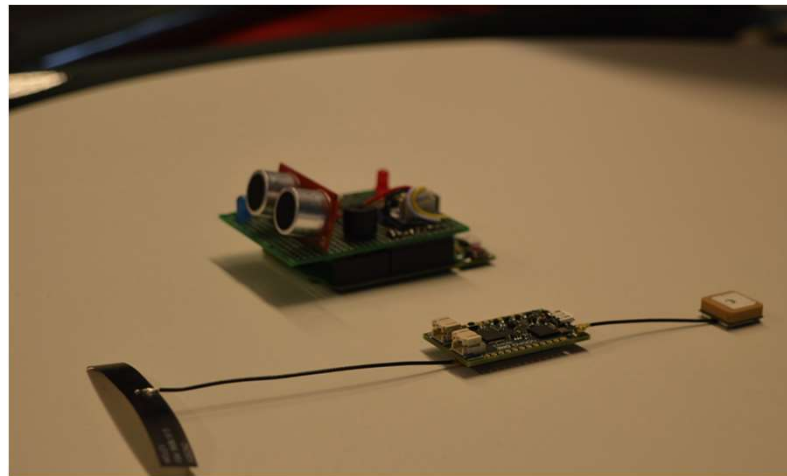
- Board proper for IoT Development
- Identify Rider's presence using a different sensor
- Extra functionalities???

PROTOTYPE #2

MCU: SodaqOne (32-Bit ARM)
LoRa: RN2483/Class A
Operating Voltage: 3.3V

Sensors

Accelerometer + gyroscope
Magnetometer
GPS Module



Extra Features

Magnetometer

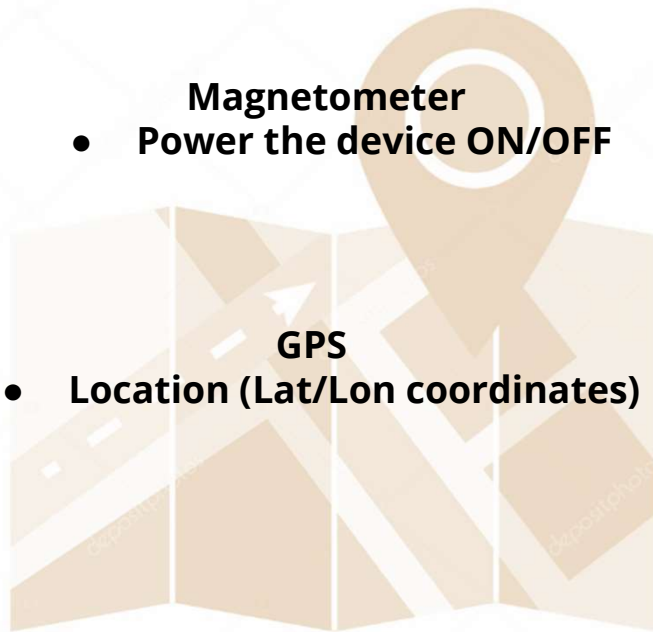
- Power the device ON/OFF

GPS

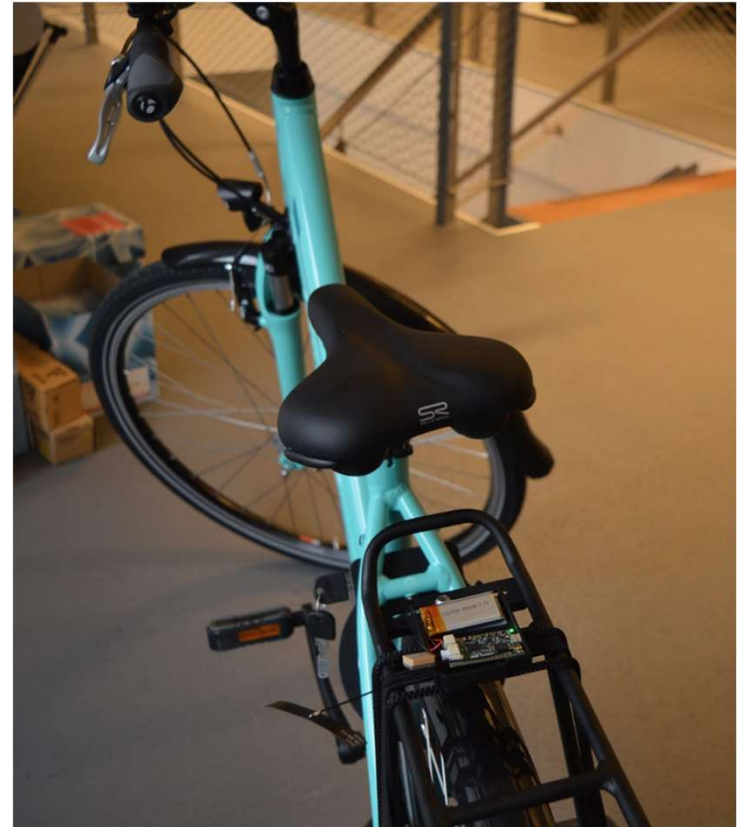
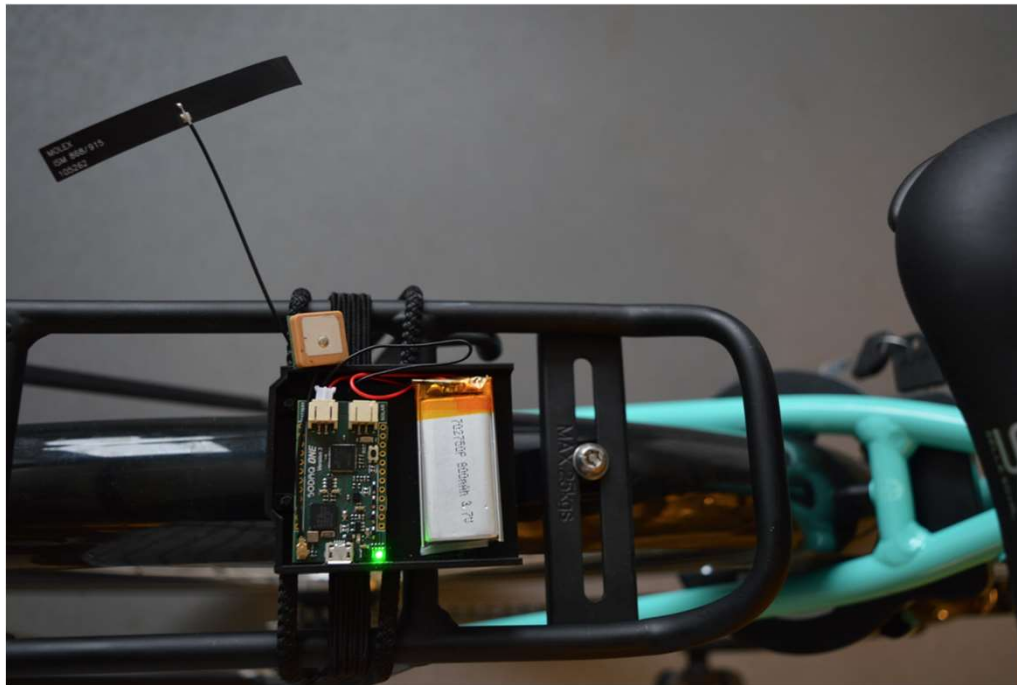
- Location (Lat/Lon coordinates)

Power Consumption

SodaqOne Sleep Mode: 50uA X Arduino 35mA



Prototype #2



TTN Server

Applications > appsodaq1 > Devices > sodaq1board > Data

Filters: [uplink](#) [downlink](#) [activation](#) [ack](#) [error](#)

time	counter	port	payload
▲ 14:45:32	7	1	00 01 03 1F 0A 6C 00 49 BB D9 detectCount: 1 lat: 52.365932 lon: 4.832217

Uplink

Payload

00 01 03 1F 0A 6C 00 49 BB D9

Fields

```
{
  "detectCount": 1,
  "lat": 52.365932,
  "lon": 4.832217
}
```

Metadata

```
{
  "time": "2019-03-17T13:45:32.78359363Z",
  "frequency": 868.3,
  "modulation": "LORA",
  "data_rate": "SF12BW125",
  "coding_rate": "4/5",
  "gateways": [
    {
      "gtw_id": "eui-58a0cbfffe8002d2",
      "timestamp": 84783860
    }
  ]
}
```

Future Improvements

- Sleep functions Optimization
- Hall Sensor for speed measurement
- Fuzzy logic implementation to infer on the accident severity

```
uint32_t start = millis();
uint32_t timeout = 1L * 1000;
SerialUSB.println(String("waiting for fix ..., timeout=") + timeout + String("ms"));
if (sodaq_gps.scan(true, timeout)) {
    SerialUSB.println(String(" time to find fix: ") + (millis() - start) + String("ms"));
    SerialUSB.println(String(" datetime = ") + sodaq_gps.getDateTimeString());
    SerialUSB.println(String(" lat = ") + String(sodaq_gps.getLat(), 7));
    SerialUSB.println(String(" lon = ") + String(sodaq_gps.getLon(), 7));
    SerialUSB.println(String(" num sats = ") + String(sodaq_gps.getNumberOfSatellites()));
} else {
    SerialUSB.println("No Fix");
}

}

delay(1000);

SerialUSB.println(lat);
SerialUSB.println(lon);

flat = sodaq_gps.getLat();
flon = sodaq_gps.getLon();
}

void doSend() {

    if(fall == true){

        byte bytesToSend[10];

        bytesToSend[0] = detectCount >> 8;        // High byte Accelerometer
        bytesToSend[1] = detectCount ;             // Low Byte Accelerometer
```

Further use cases...

- Elderly Walkers



- Wheelchairs



- Wearable devices



That's all folks!

THANKS!

