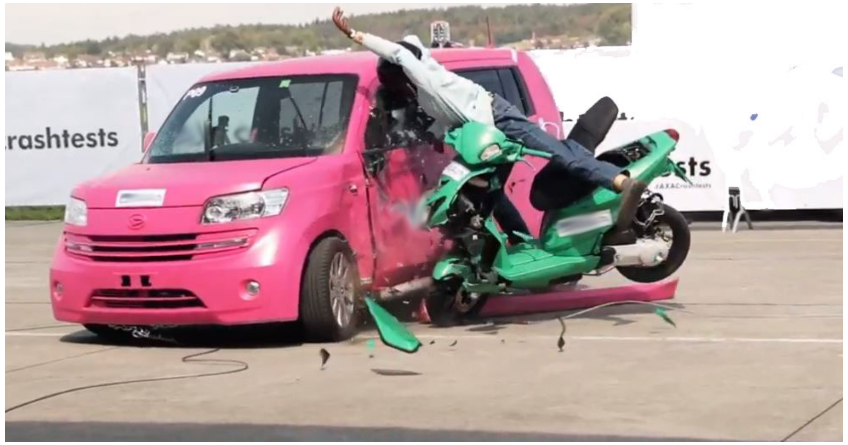
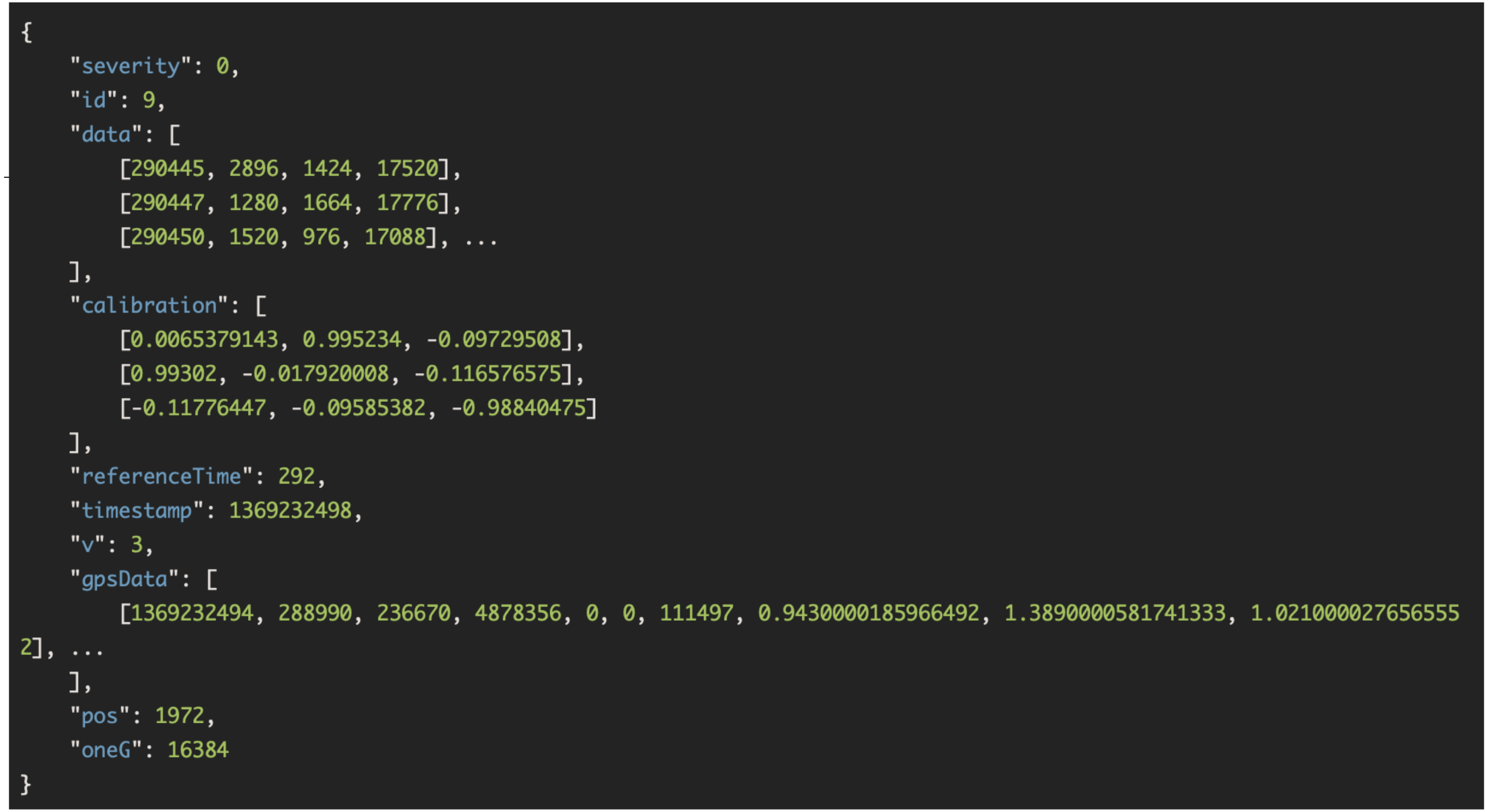
CRASH SIMULATION INPUT DATA



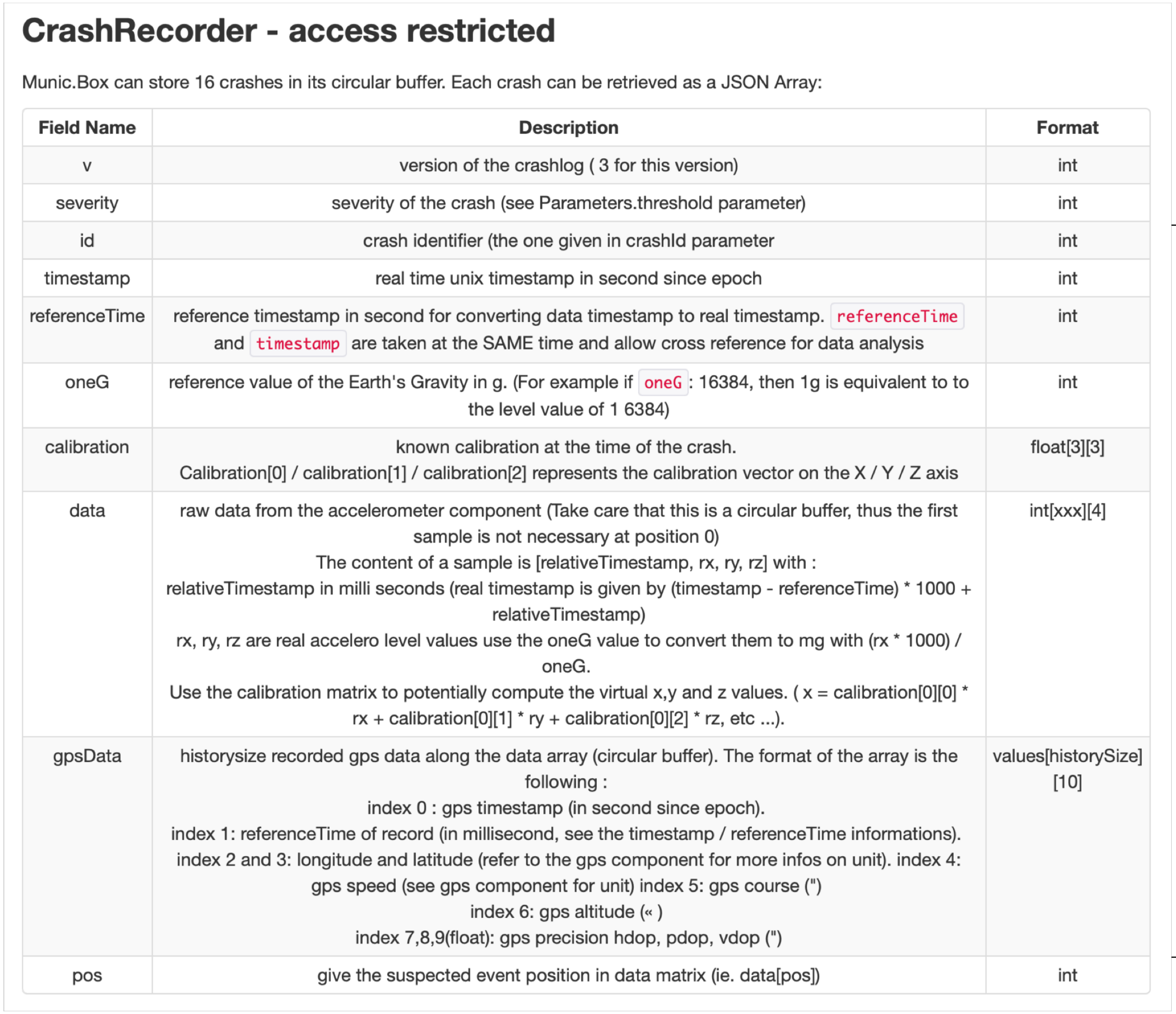
Start Hack
2019

5 CRASHES

- Crash JSON Files: **http://tinyurl.com/asens001**
  - https://bit.ly/2Cbw6CW



```
{
    "severity": 0,
    "id": 9,
    "data": [
        [290445, 2896, 1424, 17520],
        [290447, 1280, 1664, 17776],
        [290450, 1520, 976, 17088], ...
    ],
    "calibration": [
        [0.0065379143, 0.995234, -0.09729508],
        [0.99302, -0.017920008, -0.116576575],
        [-0.11776447, -0.09585382, -0.98840475]
    ],
    "referenceTime": 292,
    "timestamp": 1369232498,
    "v": 3,
    "gpsData": [
        [1369232494, 288990, 236670, 4878356, 0, 0, 111497, 0.9430000185966492, 1.3890000581741333, 1.021000027656555
2], ...
    ],
    "pos": 1972,
    "oneG": 16384
}
```



CrashRecorder - access restricted

Munic.Box can store 16 crashes in its circular buffer. Each crash can be retrieved as a JSON Array:

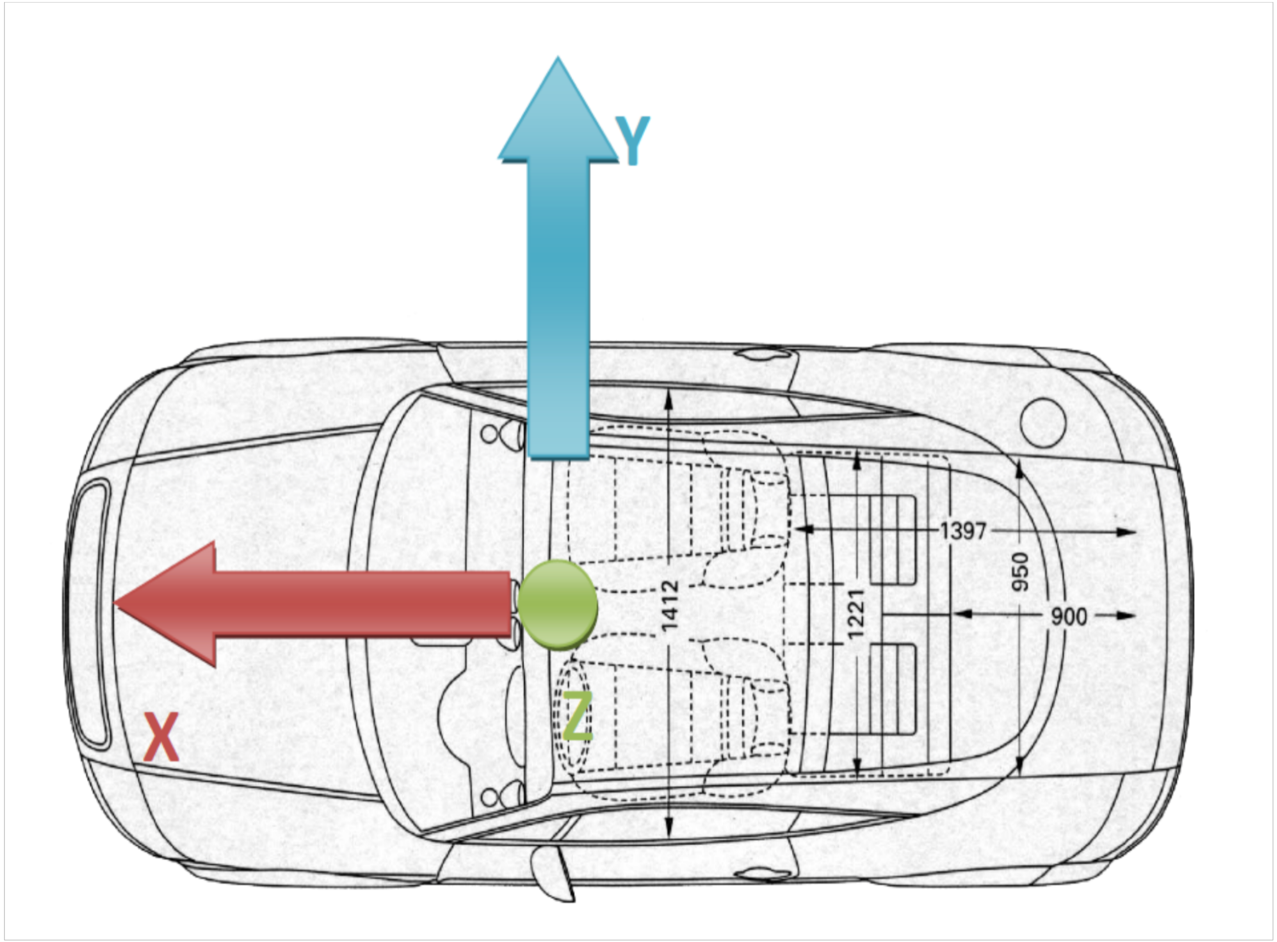
Field Name	Description	Format
v	version of the crashlog (3 for this version)	int
severity	severity of the crash (see Parameters.threshold parameter)	int
id	crash identifier (the one given in crashId parameter	int
timestamp	real time unix timestamp in second since epoch	int
referenceTime	reference timestamp in second for converting data timestamp to real timestamp. `referenceTime` and `timestamp` are taken at the SAME time and allow cross reference for data analysis	int
oneG	reference value of the Earth's Gravity in g. (For example if `oneG` : 16384, then 1g is equivalent to to the level value of 1 6384)	int
calibration	known calibration at the time of the crash. Calibration[0] / calibration[1] / calibration[2] represents the calibration vector on the X / Y / Z axis	float[3][3]
data	raw data from the accelerometer component (Take care that this is a circular buffer, thus the first sample is not necessary at position 0) The content of a sample is [relativeTimestamp, rx, ry, rz] with : relativeTimestamp in milli seconds (real timestamp is given by (timestamp - referenceTime) * 1000 + relativeTimestamp) rx, ry, rz are real accelero level values use the oneG value to convert them to mg with (rx * 1000) / oneG. Use the calibration matrix to potentially compute the virtual x,y and z values. (x = calibration[0][0] * rx + calibration[0][1] * ry + calibration[0][2] * rz, etc ...).	int[xxx][4]
gpsData	historysize recorded gps data along the data array (circular buffer). The format of the array is the following : index 0 : gps timestamp (in second since epoch). index 1: referenceTime of record (in millisecond, see the timestamp / referenceTime informations). index 2 and 3: longitude and latitude (refer to the gps component for more infos on unit). index 4: gps speed (see gps component for unit) index 5: gps course (") index 6: gps altitude («) index 7,8,9(float): gps precision hdop, pdop, vdop (")	values[historySize] [10]
pos	give the suspected event position in data matrix (ie. data[pos])	int

DELIVERY

Microservice with following API's:

- POST /api/v1/getCrashInfo
  - Response: JSON { “impactAngle”: degrees, “offsetMaximumForce”:millisecond }

- POST /api/v1/getCrashImage?timeOffsetMS={offsetInMilliseconds}
  - Response: PNG Image with rendered crash image


Y
X
Z
1412
1221
1397
950
900