Mobile App

Student: Yi Qiang Ji Zhang
Professor: Dr. Jaume Figueras Jove
Aerospace Engineering
Polythecnical University of Catalonia

07 June 2021

1 Create REST server

In the following report, it is shown how to make queries from a mobile app using cordova.

The intention is to have the raspberry running and at the same time, use the mobile interface to make queries.

```
1
2  # Import Flask
3  from flask import Flask, redirect
4  from flask_cors import cross_origin
5  from sense_emu import SenseHat
6  import datetime
7  from flask import jsonify
8  from flask import request
9
10 sense = SenseHat()
11
12 # Store name of the program
13 app = Flask(__name__)
14
15 # Route of the app is the main route of the web server
16 @app.route('/')
17 @cross_origin()
18 # Function
19 def index():
20     message = "Raspberry PI ICT REST Server"
21     return message
22
23
24 # Create a new sensors route
25 # (http://127.0.0.1:5000/sensors?origin={temperature,pressure,humidity,accelerometer,gyroscope,...
       magnetometer,imu})
26 @app.route('/sensors')
27 @cross_origin()
```

```
28 # Function to show all available sensors
29 def sensors():
30     origin = request.args.get('origin')
31     if origin is None:
32         return "Select sensor: /sensors?origin={temperature,pressure,humidity,accelerometer,...
       gyroscope,magnetometer,imu}"
33     else:
34         if origin == 'temperature':
35             return redirect('/sensors/temperature')
36         elif origin == 'pressure':
37             return redirect('/sensors/temperature')
38         elif origin == 'humidity':
39             return redirect('/sensors/humidity')
40         elif origin == 'acceloremeter':
41             return redirect('sensors/accelerometer')
42         elif origin == 'gyroscope':
43             return redirect('sensors/gyroscope')
44         elif origin == 'magnetometer':
45             return redirect('sensors/magnetometer')
46         elif origin == 'imu':
47             return redirect('sensors/imu')
48
49
50 ## Sensors
51 # Create a new temperature route
52 @app.route('/sensors/temperature')
53 @cross_origin()
54 # Function to get temperature from temperature sensor
55 def temp():
56     temp = sense.get_temperature()
57     # Create a dictionary
58     data = dict()
59     # Save variables
60     data['temp'] = temp
61     data['time_stamp'] = "{0:%Y-%m-%dT%H:%M:%S.%fZ}".format(datetime.datetime.utcnow())
62     return jsonify(data)
63
64 @app.route('/sensors/temperature/pressure')
65 @cross_origin()
66 # Function to get temperature from pressure sensor
67 def temp_pressure():
68     temp_p = sense.get_temperature_from_pressure()
69     # Create a dictionary
70     data = dict()
71     # Save variables
72     data['temp_p'] = temp_p
73     data['time_stamp'] = "{0:%Y-%m-%dT%H:%M:%S.%fZ}".format(datetime.datetime.utcnow())
74     return jsonify(data)
75
76 @app.route('/sensors/temperature/humidity')
77 @cross_origin()
78 # Function to get temperature from humidity sensor
79 def temp_humidity():
80     temp_h = sense.get_temperature_from_humidity()
81     # Create a dictionary
82     data = dict()
83     # Save variables
84     data['temp_h'] = temp_h
```

```
85      data['time_stamp'] = "{0:%Y-%m-%dT%H:%M:%S.%fZ}".format(datetime.datetime.utcnow())
86      return jsonify(data)
87
88
89  # Pressure route
90  @app.route('/sensors/pressure')
91  @cross_origin()
92  # Function to get pressure from presure sensor
93  def pressure():
94      pressure = sense.get_pressure()
95      # Create a dictionary
96      data = dict()
97      # Save variables
98      data['pressure'] = pressure
99      data['time_stamp'] = "{0:%Y-%m-%dT%H:%M:%S.%fZ}".format(datetime.datetime.utcnow())
100     return jsonify(data)
101
102 # Humidity route
103 @app.route('/sensors/humidity')
104 @cross_origin()
105 # Function to get pressure from presure sensor
106 def humidty():
107     humidity = sense.get_humidity()
108     # Create a dictionary
109     data = dict()
110     # Save variables
111     data['humidity'] = humidity
112     data['time_stamp'] = "{0:%Y-%m-%dT%H:%M:%S.%fZ}".format(datetime.datetime.utcnow())
113     return jsonify(data)
114
115 # Compass route
116 @app.route('/sensors/compass')
117 @cross_origin()
118 # Function to get magnetometer (compass) from magnetometer sensor
119 def compass():
120     compass = sense.get_compass()
121     # Create a dictionary
122     data = dict()
123     # Save variables
124     data['compass'] = compass
125     data['time_stamp'] = "{0:%Y-%m-%dT%H:%M:%S.%fZ}".format(datetime.datetime.utcnow())
126     return jsonify(data)
127
128
129 # Accelerometer route
130 @app.route('/sensors/accelerometer')
131 @cross_origin()
132 # Function to get accelerometer from accelerometer sensor
133 def accelerometer():
134     # Read accelerometer data from accelerometer sensor
135     for i in range(0,10):
136         accel_only = sense.get_accelerometer()
137     pitch_acc = accel_only["pitch"]
138     roll_acc = accel_only["roll"]
139     yaw_acc = accel_only["yaw"]
140     # Create a dictionary
141     data = dict()
142     # Save variables
```

```
143     data['pitch_acc'] = pitch_acc
144     data['roll_acc'] = roll_acc
145     data['yaw_acc'] = yaw_acc
146     data['time_stamp'] = "{0:%Y-%m-%dT%H:%M:%S.%fZ}".format(datetime.datetime.utcnow())
147     return jsonify(data)
148 
149 # Gyroscope route
150 @app.route('/gyroscope')
151 @cross_origin()
152 # Function to get gyroscope from gyroscope sensor
153 def gyroscope():
154     # Read gyroscope data from gyroscope sensor
155     for i in range(0,10):
156         gyro_only = sense.get_gyroscope()
157     pitch_gyro = gyro_only["pitch"]
158     roll_gyro = gyro_only["roll"]
159     yaw_gyro = gyro_only["yaw"]
160     # Create a dictionary
161     data = dict()
162     # Save variables
163     data['pitch_gyro'] = pitch_gyro
164     data['roll_gyro'] = roll_gyro
165     data['yaw_gyro'] = yaw_gyro
166     data['time_stamp'] = "{0:%Y-%m-%dT%H:%M:%S.%fZ}".format(datetime.datetime.utcnow())
167     return jsonify(data)
168 
169 
170 # IMU route
171 @app.route('/sensors/imu')
172 @cross_origin()
173 # Function to get IMU from IMU sensor (processed)
174 def imu():
175     # Read IMU data from IMU sensor (processed)
176     for i in range(0,10):
177         o = sense.get_orientation() # 'o' object is a dictionary
178     pitch_IMU = o["pitch"]
179     roll_IMU = o["roll"]
180     yaw_IMU = o["yaw"]
181     # Create a dictionary
182     data = dict()
183     # Save variables
184     data['pitch_IMU'] = pitch_IMU
185     data['roll_IMU'] = roll_IMU
186     data['yaw_IMU'] = yaw_IMU
187     data['time_stamp'] = "{0:%Y-%m-%dT%H:%M:%S.%fZ}".format(datetime.datetime.utcnow())
188     return jsonify(data)
189 
190 
191 ## History Requests
192 # http://127.0.0.1:5000/sensors/temperature/history?from=2021-05-11&to=2021-05-12
193 
194 # Temperature history
195 @app.route('/sensors/temperature/history')
196 @cross_origin()
197 # Request history
198 def temp_history():
199     from_date = request.args.get('from')
200     to_date = request.args.get('to')
```

```
201     return "From {0} to {1}".format(from_date,to_date)
202
203 # Temperature from Pressure Sensors history
204 @app.route('/sensors/temperature/presssure/history')
205 @cross_origin()
206 # Request history
207 def temp_p_history():
208     from_date = request.args.get('from')
209     to_date = request.args.get('to')
210     return "From {0} to {1}".format(from_date,to_date)
211
212 # Temperature from Humidity sensor history
213 @app.route('/sensors/temperature/humidity/history')
214 @cross_origin()
215 # Request history
216 def temp_h_history():
217     from_date = request.args.get('from')
218     to_date = request.args.get('to')
219     return "From {0} to {1}".format(from_date,to_date)
220
221 # Pressure history
222 @app.route('/sensors/pressure/history')
223 @cross_origin()
224 # Request history
225 def pressure_history():
226     from_date = request.args.get('from')
227     to_date = request.args.get('to')
228     return "From {0} to {1}".format(from_date,to_date)
229
230 # HUmidity history
231 @app.route('/sensors/humidity/history')
232 @cross_origin()
233 # Request history
234 def humidity_history():
235     from_date = request.args.get('from')
236     to_date = request.args.get('to')
237     return "From {0} to {1}".format(from_date,to_date)
238
239 # Compass history
240 @app.route('/sensors/compass/history')
241 @cross_origin()
242 # Request history
243 def compass_history():
244     from_date = request.args.get('from')
245     to_date = request.args.get('to')
246     return "From {0} to {1}".format(from_date,to_date)
247
248 # Accelerometer history
249 @app.route('/sensors/accelerometer/history')
250 @cross_origin()
251 # Request history
252 def accelerometer_history():
253     from_date = request.args.get('from')
254     to_date = request.args.get('to')
255     return "From {0} to {1}".format(from_date,to_date)
256
257 # Gyroscope history
258 @app.route('/sensors/accelerometer/history')
```

```
@cross_origin()
# Request history
def gyroscope_history():
    from_date = request.args.get('from')
    to_date = request.args.get('to')
    return "From {0} to {1}".format(from_date,to_date)

# IMU history
@app.route('/sensors/imu/history')
@cross_origin()
# Request history
def imu_history():
    from_date = request.args.get('from')
    to_date = request.args.get('to')
    return "From {0} to {1}".format(from_date,to_date)



# Debug
if __name__ =='__main__':
    app.run(debug=True,host='0.0.0.0')
```

Listing 1: Create and write to database

Once this has been performed, we shall go and develop the mobile interface.

The main problem found was the security reason. For security standards, the app could not connect to the REST server due to security standards.

References

[1] Python Hosted. *Sense HAT API Reference*. 2021. URL: https://pythonhosted.org/sense-hat/api/#imu-sensor.