

# Sensor Monitorin System

## Съдържание

<b>1. Разработка на проекта.....</b>	<b>2</b>
<b>1.1 Симулиране на сензор .....</b>	<b>2</b>
<b>1.2 Сървърна част.....</b>	<b>3</b>
<b>1.3 База данни.....</b>	<b>4</b>
<b>1.4 Визуализиране на данните чрез React .....</b>	<b>5</b>

# 1. Разработка на проекта

## 1.1 Симулиране на сензор

Симулиране на сензора, го направих с QEMU емулатор в който инсталирах Raspbian-Buster. След създаването на виртуалната среда инсталирах Sense-Hat-Emulator.

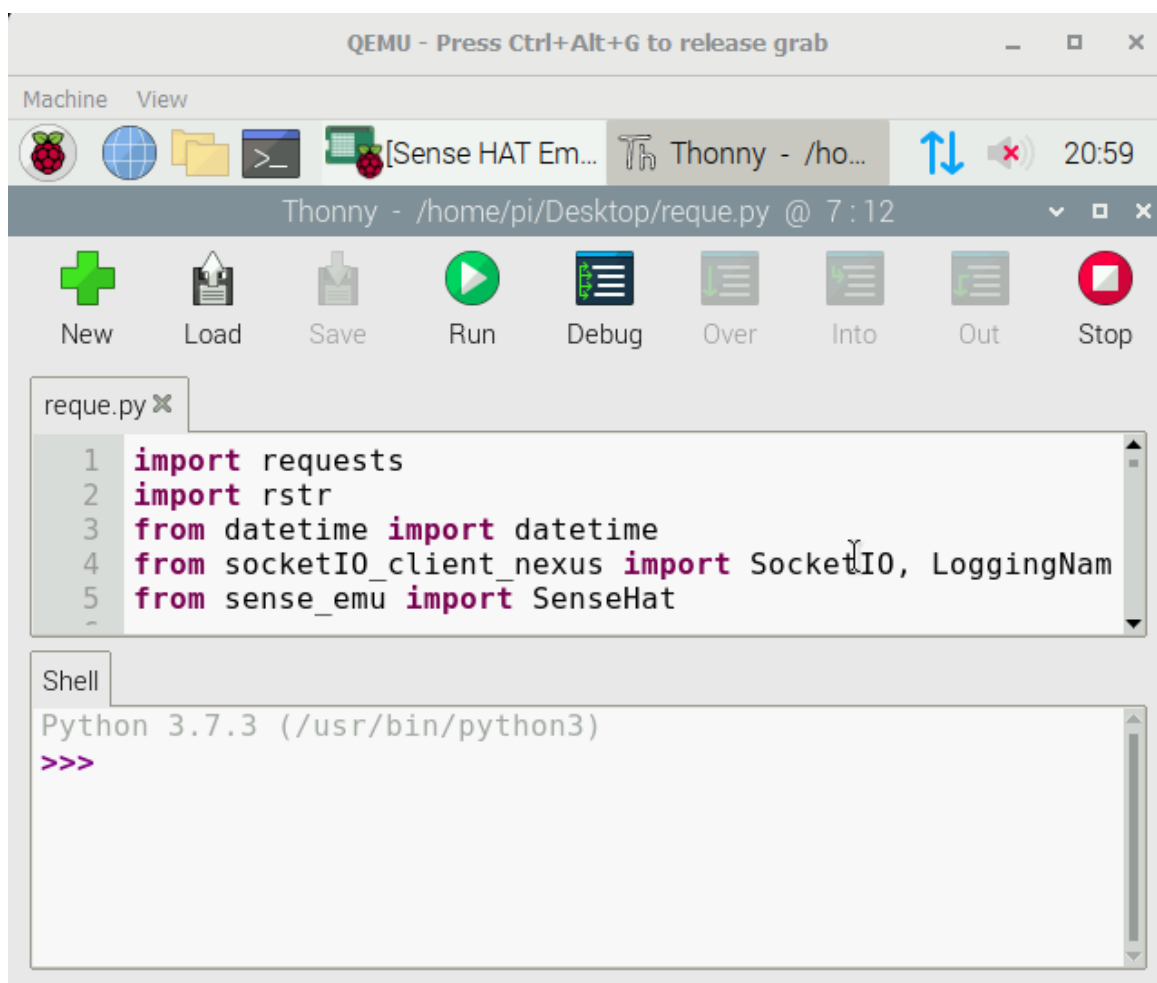
С файл написан на Python, взимам данни от Sense-Hat-Emulator.

В него генерирам [име на сензор, ид на сензор].

Изпращане на данни през сокет изпълних чрез socketIO-client-nexus.

За да работи виртуалната среда, трябва да се въведе нейното IP в hosts на Windows и да се създаде изолирана среда на виртуалната машина.

IP-то се взима от двете противоположни стрелки и трябва да въведено тук: c:\Windows\System32\Drivers\etc\host



## 1.2 Сървърна част

-Инсталирани зависимости:

sqlite3, cors, nodemon, sequelize, socket.io, express, sequelize-auto, sequelize-cli

-Създаване на моделите с команда:

```
npx sequelize-auto -o "./models" -d database/sensor.db -h localhost -e sqlite
```

-Стартиране на сървърната част с команда:

```
npm start
```

- Резултат от crud процедура в Postman:

- GET/sensors

The screenshot shows a Postman interface for a GET request to `http://192.168.1.4:8000/sensors`. The response status is 200 OK, with a time of 115 ms and a size of 1.46 KB. The response body is displayed in JSON format, showing an array of 6 sensor objects.

```
[{"sensor_id": "-1244315927", "sensor_name": "ax86", "sensor_status": 0}, {"sensor_id": "816633075", "sensor_name": "W569", "sensor_status": 1}, {"sensor_id": "1690771959", "sensor_name": "Xv61", "sensor_status": 0}, {"sensor_id": "-193901044", "sensor_name": "mh95", "sensor_status": 0}, {"sensor_id": "972348736", "sensor_name": "o751", "sensor_status": 0}, {"sensor_id": "-1455769685", "sensor_name": "fa24", "sensor_status": 0}, {"sensor_id": "728544128", "sensor_name": "Om41", "sensor_status": 0}]
```

### 1.3 База данни

За създаването на базата данни използвах DB Browser for SQLite.

Таблица с записани резултати в нея при изпращането на данни от сензора в сървърната част.

Table: sensor			New Record Delete Record		
sensor_id sensor_name sensor_status					
Filter	Filter	Filter			
1	-1244315927	ax06	0		
2	816633075	M569	1		
3	1690771959	Xv81	0		
4	-193901044	mH95	0		
5	972348736	o751	0		
6	-1455769685	fa24	0		
7	728544128	Om41	0		
8	999176833	yt68	1		
9	1847155241	Oi08	1		
10	-1953681120	zF50	1		
11	218670613	qh01	0		
12	725442	7E38	0		
13	-86072559	tS88	0		
14	-1383320922	1A33	0		
15	-594679276	_Y12	1		
16	-980656181	R114	0		
17	-1694360210	NG33	0		
18	-656394703	OC75	0		
19	86902038	p884	1		

## 1.4 Визуализиране на данните чрез React

-Инсталиране на зависимост:

socket.io-client

-Стартиране на проекта чрез:

npm start

-Резултат от заявка към сървърната част:

Information About Sensors			Real-Time Data About Sensors
SensorStatus	SensorName	SensorID	
0	ax06	-1244315927	
1	MS69	816633075	
0	Xv81	1690771959	
0	mH95	-193901044	
0	o751	972348736	
0	fa24	-1455769685	
0	Om41	728544128	
1	yt68	999176833	
1	OIO8	1847155241	
1	zP90	-1953681120	
0	qh01	218670613	
0	7E38	725442	
0	t588	-86072559	
0	1A33	-1383320922	
1	_Y12	-594679276	
0	R114	-980656181	
0	NG33	-1694360210	
0	OC75	-656394703	
1	p084	86902038	