COSC2790 Semester 2, 2020
Assignment 1 Specification

Marks allocated:	This assignment will be marked out of 70 and is worth 35% of your overall mark
Deadline:	Sunday 02.08.2020 (11:59 pm)
Submit via:	Canvas
Work mode:	In a group of **2** (***individual submission NOT allowed***)
Submission format:	**.zip** (*No other formats will be accepted*)
Face to Face demo:	Week 6 Monday-Friday (No Demo→ No marks)

1.1 Aim

The aim of this assignment is to write a small IoT application using Raspberry Pi and Sense HAT in Python language.

Some of the tasks of this assignment will require self-exploration and research, you will not find the answer in lectures notes and/or tutorials.

You must attend a **15-minute** demo session to get Assignment 1 marked during week 6. A schedule and a booking document will be published soon. You must submit the assignment prior to demo.
No submission → No demo → No marks.

1.2 Important | *please read this section*

You must adhere to the following requirements:

a. Raspberry Pi model 4 or 3 may be used.

b. You must use Python 3.5 or > 3.5 to complete the tasks. Older versions must not be used.

c. Whether working individually or in a group, you must use a version control system of some sorts such as *GitHub*, *Bitbucket*, etc…

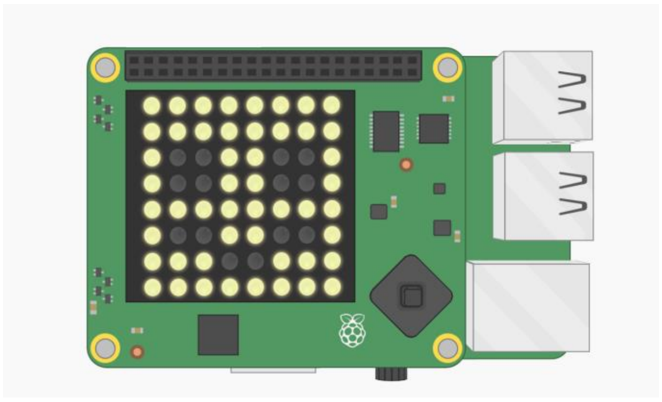
d. You must stick to the standard style guide for your Python code:
https://www.python.org/dev/peps/pep-0008/

e. Your Python code must be object-oriented. Procedural code to fetch *zero*.

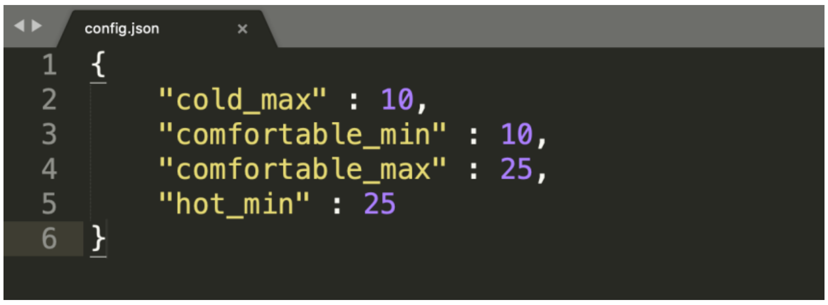
1.3 Tasks | Playing with Sense HAT

Coding Tasks

[Task a] (5 marks) Create a python file called animatedEmoji.py which will display 3 Emoji faces in the LED matrix on Sense HAT. You need to create your own Emoji with at least 3 colours for each. You also need to display them one by one with 3 seconds interval.



[Task b] (25 marks) Create a JSON config file will store three levels of temperature including *cold*, *comfortable*, *hot* with their range. This file should be called `config.json`:



```
{
      "cold_max" : 10,
      "comfortable_min" : 10,
      "comfortable_max" : 25,
      "hot_min" : 25
}
```

(1) Create a python file called `monitorAndNotify.py` which will log the current time, temperature and humidity to a database every minute (you can choose what type of database to use: **SQLite 3.*** or **MySQL**)

In this course we cover SQLite & MySQL, if you want to use another database, please feel free to do so. Use of Cloud databases is not allowed for assignment 1, you must use a local database installed on your Raspberry Pi. Use of Cloud databases is reserved for assignment 2.

This script should be scheduled to automatically run when the Pi boots.

You can use any means to run the script automatically, such as a cronjob, systemd service, having the file run on boot in the background, etc.

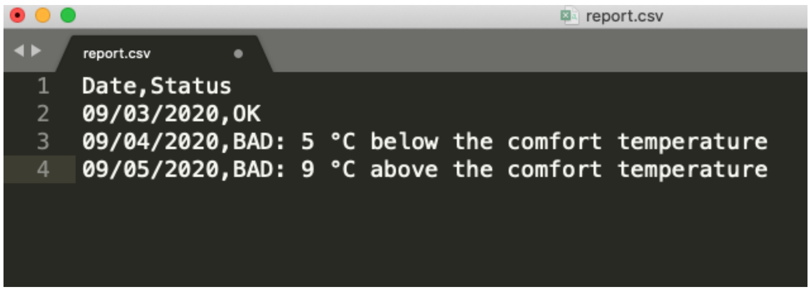
(2) After saving to the database check if either the temperature or humidity are outside the comfortable range and if so, push a notification using Pushbullet. Only send a maximum of 1 notification per day, i.e., don't resend the notification every minute.

(3) Create a python file called readAndDisplay.py which will retrieve the newest temperature record from the database and display the temperature in LED matrix on Sense HAT. The LED light should be refreshed every minute.
   1. If the level of temperature is ***cold***, display temperature with ***blue*** colour.
   2. If the level of temperature is ***comfortable***, display temperature with ***green*** colour.
   3. If the level of temperature is ***hot***, display temperature with ***red*** colour.

(4) Create a python file called createReport.py which will create a csv file called report.csv. This file should contain a separate row for each days' data, additionally this data resides in the database. If each piece of data is within the configured comfort temperature range then the status of OK is applied, otherwise the label of BAD is applied. An appropriate message detailing the error(s) is included.

This script is executed manually to generate the report.

Here is an example of report.csv:



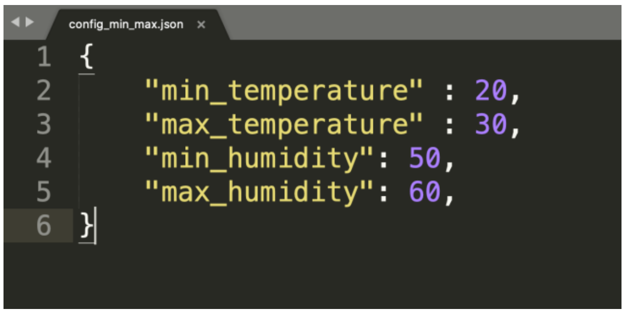
```
Date,Status
09/03/2020,OK
09/04/2020,BAD: 5 °C below the comfort temperature
09/05/2020,BAD: 9 °C above the comfort temperature
```

(5) Create a python file called `apiRESTful.py` which will contain RESTful APIs that interact with the database. There should be 3 different methods involved as following:
   1. GET: used to retrieve the newest temperature and humidity record with timestamp in JSON format.
   2. POST: used to upload a record to database with current timestamp.
   3. PUT: used to update the newest record in database.

You also need to prepare a script to demonstrate that your API does work during the demo session.

This script should be scheduled to automatically run when the Pi boots.

[Task c] (10 marks) Create a python file called `bluetooth.py` using Bluetooth to detect nearby devices and when connected send an appropriate message stating the current temperature, humidity and if these fall within the configured range in Json file called `config_min_max.json` (create your own one).



```
{
    "min_temperature" : 20,
    "max_temperature" : 30,
    "min_humidity": 50,
    "max_humidity": 60,
}
```

This script should be scheduled to automatically run when the Pi boots.

[Task d] (20 marks)
(1) Create a python file called `electronicDie.py` which will trigger the roll of the die when shaking the Pi. In this task, you need to be able to detect shaking motion by IMU sensor and then display the die in random matter.

(2) Create a python file called `game.py` which will use the electronic die to play a game with two players. The players will shake Pi several times one by one until one player gets above 30 points. The player who gets above 30 points first is the winner. At the begin of the game, you need to show the instruction of the game. During the game, you need to guide players appropriately by showing correct message and feedback. At the end of the game, you need to show who is the winner and write the record to `winner.csv` file, including *time* and *winner score*.

<u>Professionalism and Teamwork-related tasks</u>

[Task e] (10 marks) Professional use of a version control system and how the code has been developed over time. <u>Please read the assignment rubrics for details</u>.

- You must use object-oriented python code (<u>*procedural code will fetch a **zero** for the whole of assignment*</u>).
- Note: You can add more python files in addition to the ones mentioned above.
- Note: Your code will be marked for its adherence to the PEP8 style guide for Python code.

1.4 What and how to submit?

Zip all of the files to create a single zipped archive (*only .zip archives will be accepted*). Name the zipped archive as **firstStudentNumber_secondStudentNumber.zip**

As an example, a legitimate file name would be `3111111_3222222.zip`. Submit this zipped archive via the Assignment 1 page in Canvas. <u>Incorrect names will attract PENALTY of marks</u>.

Your zipped archive is to contain the following files:

- `config.json`
- `animatedEmoji.py`
- `monitorAndNotify.py`
- `readAndDisplay.py`
- `createReport.py`
- `apiRESTful.py`
- `bluetooth.py`
- `electronicDie.py`
- `game.py` and,
- any other files that you may have created

1.5 Late submission and Extension

a. A penalty of 10% per day (*which is 3 marks per day in case of this assignment*) of the total marks will apply for each day late, including both weekend and weekdays.

b. After five days, you will receive zero for the whole assignment.

c. Extension requests should only be emailed to the lecturer.

1.6 Plagiarism

All assignments will be checked with plagiarism-detection software; any student found to have plagiarised will be subject to disciplinary action. Plagiarism includes:

- Submitting work that is not your own or submitting text that is not your own.
- Allowing others to copy your work via email, printouts, social media, etc.
- Posting assignment questions (in full or partial) on external technical forums.
- Copying work from/of previous/current semester students.
- Sending or passing your work to your friends.
- Posting assignment questions on technical forums to get them solved.

A disciplinary action can lead to:

- A meeting with the disciplinary committee.
- A score of zero for the assignment.
- A permanent record of copying in your personal university records and/or
- Expulsion from the university, in some severe cases.

All plagiarism will be penalised. There are no exceptions and no excuses. You have been warned.