Infrastructure as Code

Assignment 1

Assignment:

Assignment:	Assignment 1
Subject:	Infrastructure as Code
Submission:	To BLACKBOARD as a single compressed ZIP folder, name should be *Lxxxxxxx.zip*, where *Lxxxxxxx* is your L number.

Notes

1. This must be all your own work, where you collaborate with other students, you must indicate this.
2. You may find example code online. Reference this!
3. The marking rubric is based on the deliverables, provide them exactly as specified.

Last Edited: 19 October 2023

Common Practice

Simple scripts get built upon and eventually, we need to start structuring things. We often push our code to a repository like GITHUB and before we do so, its good to have a methodical directory structure and a set of basic rules about what-goes-where. We have not covered GITHUB yet, so I'm not going to ask you to create anything that relates to it. Our scenario for this assignment is that we are storing our projects on a file share and making them available internally in a company.

I'll start by naming a project meaningfully. I have had projects called "LinuxTools", "NetworkScanner" etc. over the years, the names become unclear to me after a while.

- In 2015 I created a repository called "UbiquitiScanning15" (Ubiquiti are a network equipment provider).
- In 2018 I created a repository for Ubuntu 18.04 scripts called "UB1804"

Those names are OK, I can look back now and understand from the name exactly what I was working on. Another approach taken with commercial tools is to pick a unique name, register the URL and then a trade name. The project then continues with this unique name.

To mention a few, I own domains called:

- boatsbrains.ie
- kmn.ie
- interiority.ie
- byzantium.ie

and my own personal favourite, SolarSubmarines.com!

The names may not make their purpose obvious!

Markdown

The simplest way to write documentation is to use markdown [1] and this has become standard for Python users and in code repositories. In the root directory of every project, I will have a file called **readme.md** with basic information about the project. I may have further detail; it depends on the complexity of the project.

Structure

In my projects, the main subdirectories are:

Documentation	Any detailed documentation which is required in addition to docstrings
Examples	Scripts which will exercise this code to demonstrate how it works
Source	The packages you have written to support the project
Tests	Unit and integration tests for to project

Tasks

Task 1: Write a batch file/script in Linux or DOS (your choice!) to automate creating this structure.

Task 2: Create a **readme.md** file which explains the project.

Task 3. Build a **main.py** programme in the project root.

Task 4. Complete the log analysis project described on the following page.

Project

A farmer needs help!

In several outbuildings, he needs to monitor temperature in several different locations. He has been quoted an astronomical price for a professional system. He has asked me to help.

I can add Raspberry Pi Zero-W (RPi) for a few Euros each. I can put 1-wire sensors on them for less than a Euro each. Job done!

I need a hand.

I need you to write me a simulator to demonstrate this. Write three UDP clients which will unicast a single temperature to a server, with a time stamp and a sensor ID.

Write a server to receive this data and save it in separate logfiles.

Extra

It would be a real bonus if we can alert the farmer when temperature is <5c or >30c.

Grading

Category	Description	Weight
Structure	Project directories, files, etc. named clearly and in the correct locations. DOS or Linux batch file used to create the structure. Adapt this so it is suitable for this project, for example, we do not need an examples folder.	25%
Documentation	Comments and docstrings where appropriate. A simple readme.md file in the root directory, briefly explaining the project. A more detailed description in the documentation directory. This should include limitations or note anything that does not work.	25%
Style	Code written clearly, economically and to general best practice. To demonstrate your understanding: - Create at least one function in a module - Create at least one test	25%
Functionality	The extent which the code fulfils the specification. Define the project scope in **readme.md** Confirm what works in **docs/status.md**	25%