Pico Project: Pizza Form

The end result of this project is a script running a web server that serves a simple pizza order form.

The idea behind this project is to do the research all by yourself. Everything you need to know can be found online, some URL's are provided in this document. Experimenting, trial and error, it's all part of this adventure. If something doesn't work, do the research.

Project: **socket web server**

You are going to build a complete web server with your Pico Pi W. If ready it:
- connects to your local wifi
- serves a very simple web site (saying "Hello World") *using the **socket** library*
- and it receives and crunches URL-paths & URL-arguments and cookies

This assignment starts with connecting to your local Wifi and testing if it works. See the *general-info.pdf*.

1) Make one or more functions for connecting your Pico to the local Wifi. It must als check if the connection is working.
==> this should print internal and external IP addresses int the terminal

2) Make a function that sends a request to a working website (http://cpnits.com), getting back status-code and content. It checks if something comes back and if so, if the status code is 200. If not, it prints a warning. The function also returns True or False.
==> this should print a message in the terminal about the status of the internet connection

3) Make one or more functions that change your Pico Pi W into a web server, using the **socket** library. Responding to a request with "Hello World" is enough. But feel free to add some HTML and CSS.
==> the server should print a message at every received request with data about the request, such as time and client IP address

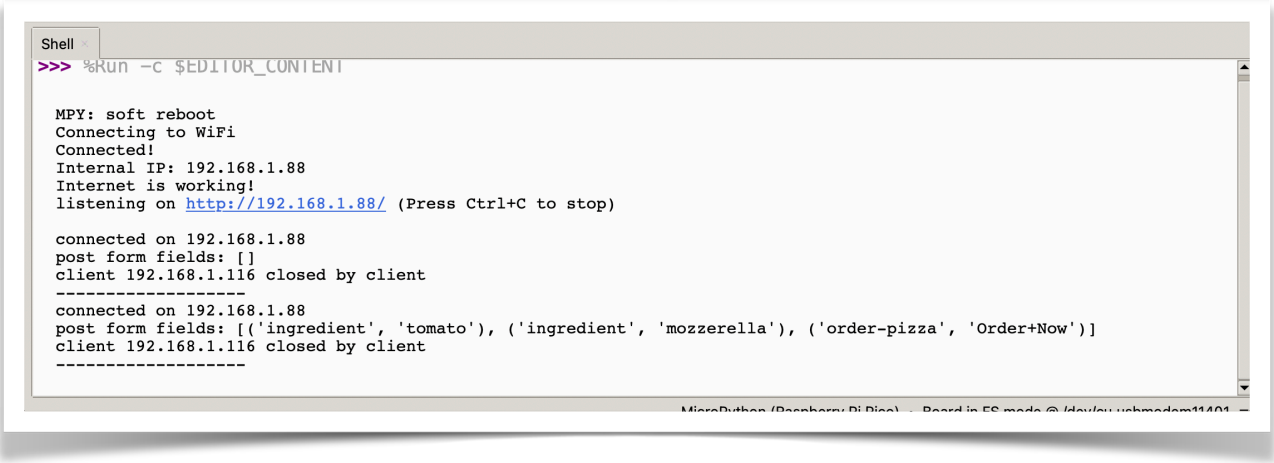
4) **Add a simple HTML form** to your server output. The HTML is provided via Blackboard.

If served well, it should look like this:
No work here.

Getting POST data in Python

This form does not use the **GET** method for retrieving data from the client (browser), but it uses the **POST** method. The difference is that all GET arguments are visible in the URL in the address bar of the browser, while POST arguments are not. Also: there is no limit to the length or amount of POST arguments, while there is for GET arguments.

An example of the returned data from the browser (after ordering a pizza with ingredients) can be found in the *post form fields* as shown in below example:



5) **Your pico pi server project** must do the same: **receive** the request from the browser and the given HTML form and **print** all chosen ingredients in the terminal. Like in above example.

There is not much information to be found on this topic, but with the following info you should get it to work. If you have a working socket:

```
s = socket.socket()
```

then per request you get a connection:

```
while True:
        connection, addr = s.accept()
        # the request data can be retrieved from the connection
        request_data = connection.recv(1024).decode()
```

All header information, such as paths and URL arguments and POST data can be found in the request_data string.s

How and what to hand in?

You will upload **3 files** to Blackboard, please **use the given file names and extensions**:

1. **server.py**: the end result script containing all functionality of the **web server** project: **Wifi connection**, **web server**, and **paths, args & cookies**.
2. **server.png**: a screen dump with all the **shell output** of your server in it, like in above example.
3. **website.png**: a screen dump of your browser with the pizza form. The address bar must be visible in this screen dump.
4.

And of course you evaluate this project in your portfolio.