COMP704 – Machine Learning	Workshop WK2
Client-Server Telephone Book App (Full-Stack)	

Introduction

The goal of this workshop is to develop a telephone book application to showcase full-stack development techniques.

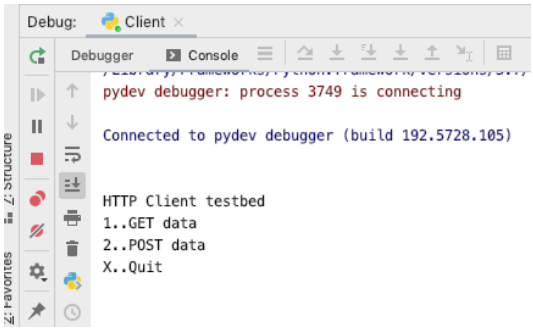
Python HTTP client/server applications

A good starting point for this task is to build client and server applications that communicate using HTTP server protocol. The code samples in the lecture will provide a framework for client and server functionality.

To create the applications, load up two instances of PyCharm and create client and server projects. Set up the projects so that you have a valid interpreter and so that you can debug. Reassemble the code fragments from the lectures and Launch the server first and then the client. Both client and server should display output to the console windows.

To explore this functionality further, build a command line application in the client so that you can trigger GET and POST requests on demand. The code framework for this is below, alongside the expected output from the debugger.

```
import http

conn = http.client.HTTPConnection("localhost",8000)

quitApp = False

while quitApp is False:

    print('\n')
    print('HTTP Client testbed')
    print('1..GET data')
    print('2..POST data')
    print('X..Quit')
    print('\n')

    key = input('>')

    if key is '1':
        #get data

    if key is '2':
        #post data

    if key is 'x':
        quitApp = True


conn.close()
```



cont…

SQL Testbed

To store your data on the server, sqlite provides a persistent framework, so if your server fails, the data will still exist. The following website: https://www.pythoncentral.io/introduction-to-sqlite-in-python/ and https://www.sqlitetutorial.net/ are good source for sqlite reference and will guide you through the database and data lifecycles (CRUD). In addition, https://www.w3schools.com/sql/ is good for general SQL reference.

A good starting point for this activity is to build a command line testbed that will let you experiment with sql commands without the overhead of having a server in place. Your existing from the previous activity will help, the client code can be re-used and refactored. Look to create an application that will let you create a local database, create a table that will hold names and phone numbers, add new contacts, delete contacts and search for contacts.

At a minimum, your database will need a table to store phone numbers (name and number). The sql to perform this is:

```
create table table_phonenumbers (name varchar(20), number varchar(20) )
```

and to insert a new entry:

```
insert into table_phonenumbers (name, number) values(?,?)
```

It's worth remembering that PyCharm should display your database in its editor and allow you to edit its format and contents as you see fit.

SQL server integration & JSON data transfer

Once you have a working testbed, you will be in a position to refactor the testbed into your client/server application. Just as the lecture spoke about MVC applications, the database (model) and SQL code (controller) will need to move into the server application whilst the user input and printing (view) will need to move into the client application.

Doing this will create the problem of how to manage potentially complex data communications between client and server, particularly if you have functionality that can return multiple records (people with the same surname, dialling codes etc). JSON will provide you with elegant solutions to these issues.

To send data from the client to the server, POST can be used with a JSON data package that includes *name* and *number* as a tuple (dictionary) of data. The responses from the server can be treated as different kinds of messages depending on whether you are going to send back text responses, for example if a name isn't in the phone book or when a new user is added.

PyQt Client

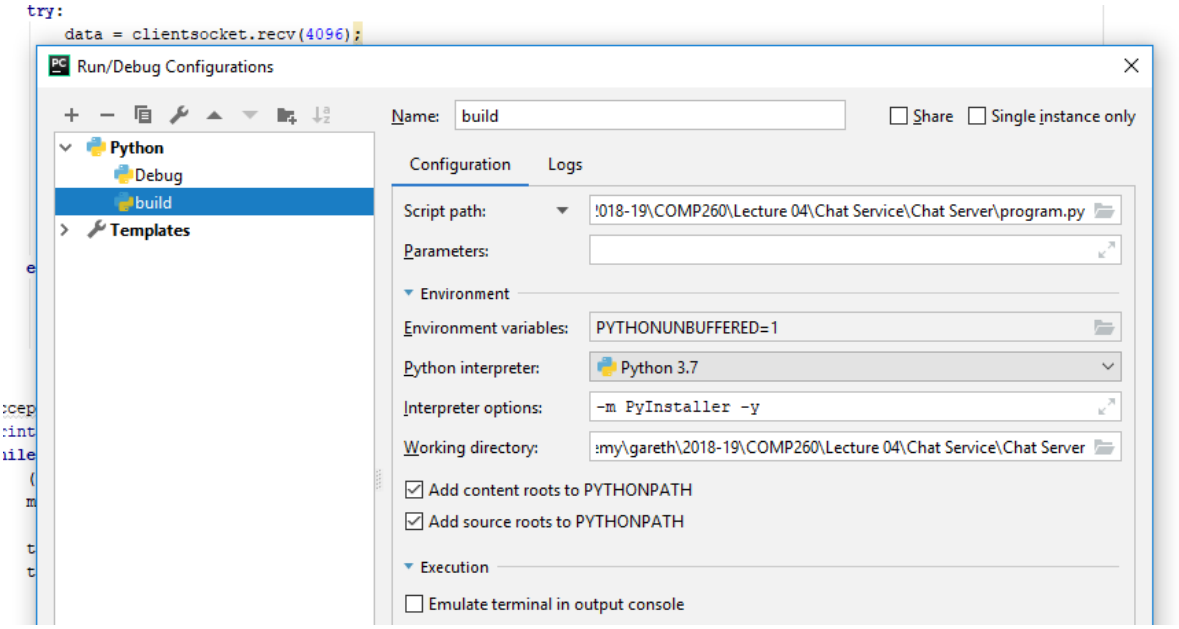
To experiment with PqQt, you will need to install the PyQt packages in PyCharm. Once this is installed, you can use the QtDesigner to layout the UI for your phone application client. To help get a good understanding of PyQt, this repo https://github.com/mfitzp/15-minute-apps contains a collection of simple applications that use Qt as a UI. Here is a guide for QtDesigner:

https://www.tutorialspoint.com/pyqt/pyqt_using_qt_designer.htm

cont…

Make exes of your projects

Python supports building to .exe files through pyInstaller. To do this, you need to install pyInstaller and create a new project configuration for build. It will need to have -m PyInstaller -y defined in the interpreter options and will make a build when you select the run (not debug) icon. On success, you will have a dist/<launch .py file name> folder with an exe in, this will run your project. If you are running the server app, the launch py file is program.py, so the folder is dist/program and the exe is program.exe. Copying the dist/program folder will allow you to distribute the application on other PCs without Python installed on them



If you make an exe of the client, it is easy to launch lots of instances, however, use –windowed in the build settings if you just want the PyQt window to show.

Further Steps

If you get to this point with working applications, well done you have performed well against the worksheet rubric (prototype python client server & SQL testbed). If you want to take your work further, look to increase the complexity of your application. Alternatively, you could look to develop a telephone book app in Unreal.