Dear Project Manager,

OK, so in order to become the best Python programmer in the world and get a high paid job at Google or master the universe, whichever comes first, I have been working hard on porting my little Raspberry Pi's internet radio interface to Windows.
Really, I just wanted to be able to listen to my favorite radio links + random stuff, locally on my Windows machine, standalone, easily and in a convenient way, without the need to have a Raspberry Pi switched on at the same time.
Besides I know your Raspberry Pi is gathering dust and that makes me sad :-(therefore I had to find another way for you to enjoy internet radio!

I also added a few lines of JavaScript code behind the auto-random button, for some eye candy magic!

So the Linux shell scripts have all been rewritten in Python and that works well. Python is getting increasingly popular and I can see why. Powerful, easy to get into, runs everywhere, even on small calculators, plus it's quite fun.

Obviously to get the interface going on Windows, you will need Python installed. But wait a minute, you already have it since you did tell me that you installed GNU RADIO, right?
I don't know which version of GNU RADIO you have installed and / or if you have changed the default install path, but I recommend the older version (gnuradio_3.7.12_win64.msi) which has way more stuff built-in (but that's another story) and comes with the older version of Python but works well here.
Nevertheless, try this:

```
C:\>"C:\Program Files\GNURadio-3.7\gr-python27\python.exe" -V
Python 2.7.10
```

If you have the same result as me, you're good to go. Important parts are "Python 2" and the same path. Otherwise you will need to adjust the scripts, but probably faster to un-install what you have and re-install my recommended version of GNU RADIO and keeping the default installation path.
http://www.gcndevelopment.com/gnuradio/downloads.htm

To speed-up Python execution as much as possible, I first call the Python binary, passing it the Python script I want to run, followed by the script parameters if any (keep the double quotes):

```
"C:\Program Files\GNURadio-3.7\gr-python27\python.exe" "C:\PC.RADIO\cgi-bin\script.py" "??" "??"
```

If you call the Python binary without passing it any script "C:\Program Files\GNURadio-3.7\gr-python27\python.exe", it will open interactive Python environment where you can run Python commands and see how they behave before you write the real script. Quite useful while learning Python!

The interface works exactly like on the Raspberry Pi. An HTML page on the client to control the radio and the Python scripts on the server to execute the requests and play the links. Of course here, client and server are on the same local machine and to play the radio we are using MPD / MPC, Linux

software but here in a Windows version. That software is considerably lighter than say VLC and seems to be good enough for what we're doing here.

So next, you think that I'm going to tell you to Install Apache or something for the web server.....
Duh, you're wrong!, you're wrong!, you're wrong! (just wrote 3 times but I was going to put more because it felt good).
Windows 10 has a hidden built-in web server that you just need to activate but wait... wait... no, even way easier than that, our little Python friend has also a web server built-in that can handle CGI requests. We can use it right away, without any fuss or configuration, something I did not know about a month ago.
So that's it dude, nothing else to install!

######################
 # QUICK INSTALLATION #
######################

Unpack my archive under your C:\ drive, so the script directory looks like that **"C:\PC.RADIO\cgi-bin"**.
Sorry the path is hard-coded for now :-O
And finally run the whole thing by clicking on **PC.RADIO.bat** located in **PC.RADIO** directory.

This batch file will **(1)** run the MPD - Music Player Daemon in a minimized window, **(2)** run the Python CGI HTTP Server in a minimized window, **(3)** load the 40 internet radio presets from the text file playlist.db and finally **(4)** call Google Chrome in kiosk mode to show the web page.

If you don't like the kiosk mode, remove the keyword. Kiosk mode only work if the browser is not already open.
If you prefer to use Firefox or Edge instead of Chrome, edit the batch file.

The first time you run it, you will probably get the Windows firewall alert window. Just click on cancel and that should not appear any more.

Couple of other batch files in that directory:
==

get_icecast_database.bat will fetch the Icecast XML file from the web (could roughly range from 12,000 to over 20,000 entries, its name is yp.xml and it will be deleted after processing to save a little bit of space) and my script (set_random.py) will read yp.xml and extract the URL & GENRE to a small SQL database called yp.db, very original right?
Run it when needed, I'd say couple of times a week. Only used for the random & auto-random features. But if you're a freak like me you can run it every minute of every day just for the satisfaction to have a shot at breaking it!
I made a huge improvement here compared to the version I had on the Pi. Very proud of that finding, just 4 spaces in Python that should not have been there. I explain. Instead of doing one commit in my database for every insert, I do my 20,000 inserts followed by ONE single commit, bringing down the runtime from over a minute to 2 seconds. Life is good :-)

scan_&_listen.bat is not needed by the interface, but just a fun little demo. It will start from the Icecast database and randomly playing radio links. You pass it 2 parameters, the genre you want to scan and the number of seconds before switching to a new station. If you don't pass anything, it will default to "Everything" and "10" seconds.
This is console only so obviously the browser and web server are not needed and the Music Player Daemon will be started automatically if it's not already running.

scan_&_show.bat is also not needed, just a fun little tool that will scan a genre from the Icecast database and display in the screen console the URL, GENRE and TITLE being played (if it could retrieve it). At the same time it will write the same information to the file scan_&_show.txt for later analyze & review. This could be useful to find potential new stations playing your favorite tunes.

######################
 # OUTSTANDING ISSUES #
######################

(1) A bit slower than I'd like it to be, considering that the client and the server are on the same box. But maybe I'm just too hard on myself, it's very acceptable. You tell me?
Python 3 could be faster? But some code changes will be required.

(2) Unicode characters have been driving me crazy for some time. For a while I thought that if I convert everything to UTF-8, I should be alright. Well I was wrong. It handles almost everything fine, including even the Russian Cyrillic so great but the French and it's special characters is all messed-up. Damn Frenchmen, hum, pardon my French!
If I use Latin-1, the French is all right but not the Cyrillic and some other characters as well.
As this affects only the display, we have to live with it for now.

(3) On some occasions MPD, the Music Player Daemon, will fail. Its window will close and you will get this message in the browser:
"error: No connection could be made because the target machine actively refused it."
If this happens too often we might need to try an older version. I'm using the latest MPD 0.21.25 released in July 2020.
https://www.musicpd.org/download.html

In the meantime just restart the job:
```
start /MIN "Music Player Daemon" "C:\PC.RADIO\mpd.exe" "C:\PC.RADIO\
mpd.conf" --kill --no-daemon --stderr --verbose
```
And reload the presets:
```
type "C:\PC.RADIO\playlist.db" | "C:\PC.RADIO\mpc.exe" add > NUL
```
Or better close everything and click on PC.RADIO.bat

Have tried to automate this and restart it automatically if I detect it has failed. It works fine in scan_&_listen.py but in do_random.py, I had to comment it out. It was basically locking the browser because it's waiting indefinitely that the script ends before returning to the browser. So in another word,

if CGI script A (called by the browser) calls CGI script B and A finishes, it will not go back to the browser because B is still running. BUMMER! There may be a workaround but I haven't found it yet.

(4) My beautiful & colorful web page, i.e. HTML is very pleasing to my eyes but I understand you don't like it that much :-/
That is very disappointing to hear, considering the time invested in it and the fact that I've been busting my butt to make it look as good as it does today!
Well, the good news is that you can improve it, you just need to convert radio.html to bring it from the nineties into 2020. You don't even have to touch the Python scripts so just have fun, be my guest, knock yourself out and let me watch in awe what you'll come-up with ;-)

Oh and take your time for the UAT, the CIO has been very cool about it and said that you can take 2 months if needed, before submitting your report. Just make sure that report is tight and we can roll out to production well before Christmas, so we're ready to listen to some great Xmas tunes with the random button.

Hope you like it and watch out for the Power Down System button but it is not *IMMED!

The Programmer

P.S. So the last remaining question is do you think Google will want me and how much should I ask for when asked during the interview?

¯_(ツ)_/¯