**Setting up Raspberry Pi for task scheduling:**

1. Download NOOBS for raspberry pi SD card, following the instructions in:

<https://www.raspberrypi.org/downloads/noobs/>

Install the operative system as required. Python is already installed in the latest version (I believe). Otherwise, follow some instructions on internet to get Python 3.7 (Don’t use Python 3.8)

1. Create a directory for the new environments using mkdir:

i.e., /home/pi/environments.

1. Go to the environment folder i.e., cd /home/pi/environments
2. Create a new python environment. i.e., python -m venv <myenvname>. For knmi\_api will be i.e., python -m env knmi.
3. Activate the environment. You need to go to the source of the activation file.

i.e. /home/pi/environments/knmi/bin

Type: source activate.

Make sure that you see the (knmi) in the console.

1. Install libraries required to run the python program.
2. Pip install pandas
3. Pip install numpy.

For raspberry 4, in the terminal type:

sudo apt-get install libatlas-base-dev

pip3 install numpy

1. Pip install matplotlib (this take a lot of time 1 hour).
2. Pip install requests
3. Install xarray (Tricky!!). Follow these steps first:
   1. Install netCDF4:

sudo apt-get install libhdf5-dev

sudo apt-get install libhdf5-serial-dev

sudo python3 -m pip install h5py

(Remove sudo to install in the environment)

sudo apt-get install libhdf5-serial-dev netcdf-bin libnetcdf-dev

sudo apt-get install netcdf-bin libnetcdf-dev

sudo python3 -m pip install netcdf4

(Remove sudo to install in the environment)

* 1. Install scipy:
     1. Option 1:

Go to <https://www.piwheels.org/simple/scipy/>

Select the scipy version and target platform (ex. armv6l for pi zero and earlier pi; armv7l for 3). For the raspberry 2 model B v1.1. This is an armv7. Therefore, the file is: scipy-1.5.1-cp37-cp37m-linux\_armv7l.whl

The link is:

<https://www.piwheels.org/simple/scipy/scipy-1.5.1-cp37-cp37m-linux_armv7l.whl#sha256=a03df78474a6fefd3322b9fe44fe1e38b26cb2737c2bba0f96233cc02dde04a9>

The way to download the file directly to the folder that you are right now in the terminal is running the command:

wget <https://www.piwheels.org/simple/scipy/scipy-1.5.1-cp37-cp37m-linux_armv7l.whl#sha256=a03df78474a6fefd3322b9fe44fe1e38b26cb2737c2bba0f96233cc02dde04a9>

go to where the .whl file was downloaded and: pip install the-scipy-file.whl

Run python. In the console type: import scipy as sp. If there is no problem, continue.

* + 1. Option 2 (Worked in raspberry pi 4):

sudo apt update

sudo apt install -y python3-scipy

* 1. Install xarray:

pip install xarray.

Run python. In the console type: import netCDF4 as nt. If there is no problem

1. Install the IDE (Interactive development environment) Shell for python:

sudo apt update

sudo apt install python3 idle3

1. Test the code running the python IDE shell.

With the (knmi) environment activated run: python -m idlelib.idle.

Or you can find the IDE shell in the OS in the init(Raspeberry icon)→Programming→Python 3 (IDLE) \*\*Watch out as running the IDLE this way, it will not enable the environment”

1. (Optional) Download the code from github
2. (Optional) Run the code and check that everything works as expected.

**Creating an SSH key in gitlab:**

You need to add an SSH key in gitlab to update and push continuously code and files, into the repository.

“One of the main advantages of using SSH for cloning is that you don’t have to enter the password multiple times for the hundreds of operations you do in a day. It is made possible by a tool or program called ssh-agent”

Here we will generate an “ED25519” key pair.

1. <https://gitlab.tue.nl/profile/keys>
2. Click on “generate one”
3. In the terminal, write: ssh-keygen -t ed25519 -C "<comment>" (e.g., “raspberry 4 ssh key”)
4. If this is the first time you generate the SSH key, accept the default directory, and skip the passphrase if you want. The terminal output:

Generating public/private ed25519 key pair.

Enter file in which to save the key (/home/pi/.ssh/id\_ed25519):

Created directory '/home/pi/.ssh'.

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /home/pi/.ssh/id\_ed25519.

Your public key has been saved in /home/pi/.ssh/id\_ed25519.pub.

The key fingerprint is:

SHA256:YBEy/phCA0GAm9q7G2t6gsfTH7jaIxctKMelNO4h3Mw raspberry 4 ssh key

The key's randomart image is:

+--[ED25519 256]--+

|Bo o o. |

|.. . o . |

| oo . o |

|o.o..= . |

|o===o.. S |

|+.OEo.. |

|.\*.+.o. |

|o X++. . |

|.\*=Boo. |

+----[SHA256]-----+

1. Open the “.pub” file (i.e. id\_ed25519.pub) with the text editor. Copy paste the SSH key in the gitlab website: <https://gitlab.tue.nl/profile/keys>

**Setting up git username and password and download repository:**

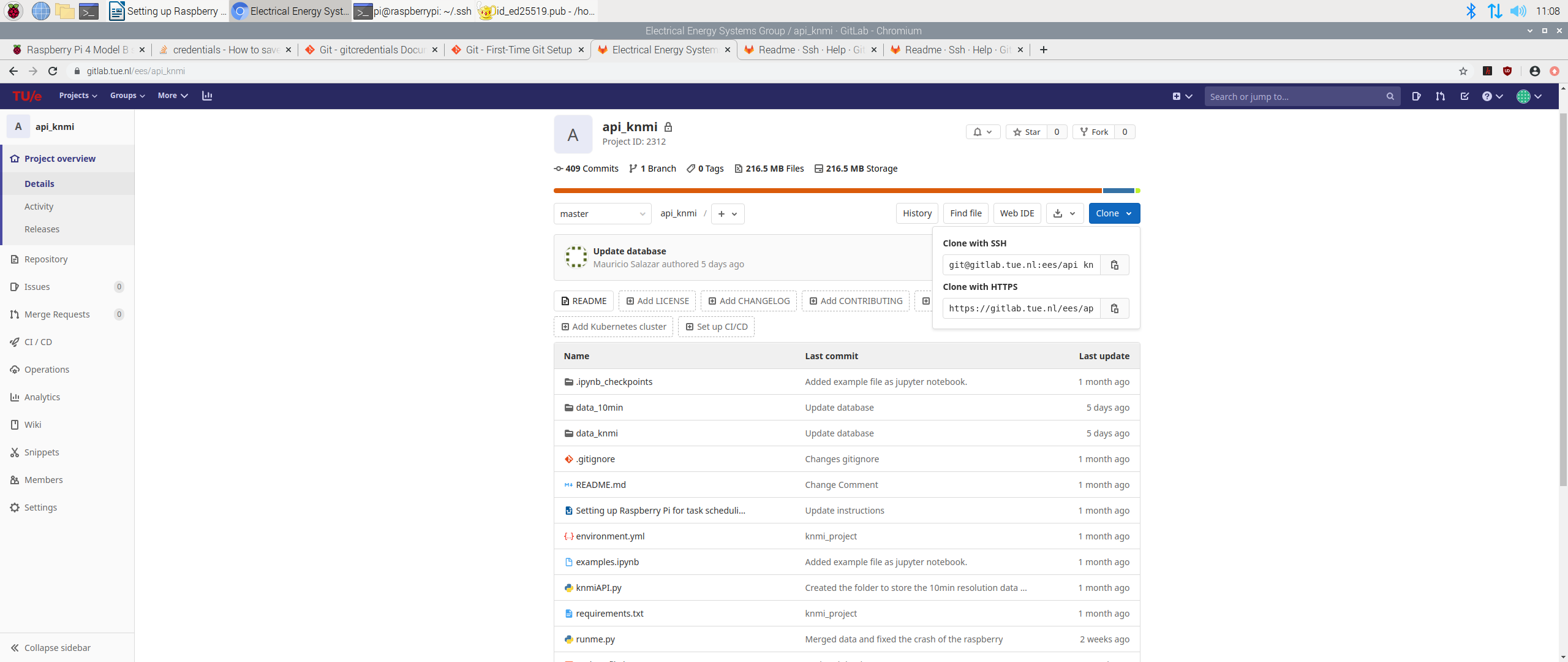
Now, we define the credentials to use in git.

1. In the terminal, type the command with your identity, for example:

git config --global user.name "John Doe"

git config --global user.email johndoe@example.com

1. Create a new folder to download the knmi\_api code:
2. Clone the gitlab project using the SSH option, first you need to copy the SSH link from the gitlab website.
3. Copy paste in the terminal: git clone [git@gitlab.tue.nl](mailto:git@gitlab.tue.nl):ees/api\_knmi.git



1. Accept the remote connection. The cloning will start and will take a few minutes to complete.

**Setting up the task scheduler in Raspberry pi:**

1. Create the bash file a name it (e.g. bash\_update\_knmi.sh) that you will run with the task scheduler (crontab):

Using nano. This is an example of the bash file:

#!/bin/sh

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games:/home/pi/python\_scripts

echo '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*' >> /home/pi/python\_scripts/log\_knmi\_backup.txt

echo '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*' >> /home/pi/python\_scripts/log\_knmi\_backup.txt

echo '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*' >> /home/pi/python\_scripts/log\_knmi\_backup.txt

echo '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*' >> /home/pi/python\_scripts/log\_knmi\_backup.txt

echo `date` >> /home/pi/python\_scripts/log\_knmi\_backup.txt

source /home/pi/environments/knmi/bin/activate >> /home/pi/python\_scripts/log\_knmi\_backup.txt

cd /home/pi/python\_scripts/api\_knmi >> /home/pi/python\_scripts/log\_knmi\_backup.txt

echo '----------------- FETCHING GIT HUB ------------------------------' >> /home/pi/python\_scripts/log\_knmi\_backup.txt

git fetch origin >> /home/pi/python\_scripts/log\_knmi\_backup.txt

git merge origin/master >> /home/pi/python\_scripts/log\_knmi\_backup.txt

echo '----------------- RUNNING SCRIPT --------------------------------' >> /home/pi/python\_scripts/log\_knmi\_backup.txt

/home/pi/environments/knmi/bin/python /home/pi/python\_scripts/api\_knmi/update\_file.py >> /home/pi/python\_scripts/log\_knmi\_backup.txt

echo '----------------- PUSHING GIT HUB -------------------------------' >> /home/pi/python\_scripts/log\_knmi\_backup.txt

git add . >> /home/pi/python\_scripts/log\_knmi\_backup.txt

git commit -m "Database update" >> /home/pi/python\_scripts/log\_knmi\_backup.txt

git push origin master >> /home/pi/python\_scripts/log\_knmi\_backup.txt

deactivate >> /home/pi/python\_scripts/log\_knmi\_backup.txt

1. Change the permission to execute the bash file to any user (so the crontab can schedule the execution).

In the command line type: chmod u=rwx,g=rwx,o=rwx bash\_update\_knmi.sh

1. Set-up crontab to run the bash file (<https://www.raspberrypi.org/documentation/linux/usage/cron.md>)

(<https://raspberrytips.com/schedule-task-raspberry-pi/>):

* 1. Type in the terminal: crontab -e
  2. If you are running for the first time crontab, select “nano” as the editor.

0 \*/1 \* \* \* /home/pi/python\_scripts/bash\_update\_knmi.sh

This code will execute the update of the file (database) called merged\_dataset.nc, every hour.

**Install VNC remote server and viewer in the Raspberry Pi:** (<https://www.raspberrypi.org/documentation/remote-access/vnc/>)

1. Run:

sudo apt update

sudo apt install realvnc-vnc-server realvnc-vnc-viewer

1. Activate the vnc server in the raspberry pi as:

Menu -> Preference -> Raspberry Pi Configuration

Interfaces Tab -> Enable VNC.

1. Reboot the raspberry pi. Make sure that you have the VNC logo in the right upper hand (next to the clock) in the task bar.

**Install VNC Viewer on the PC:**

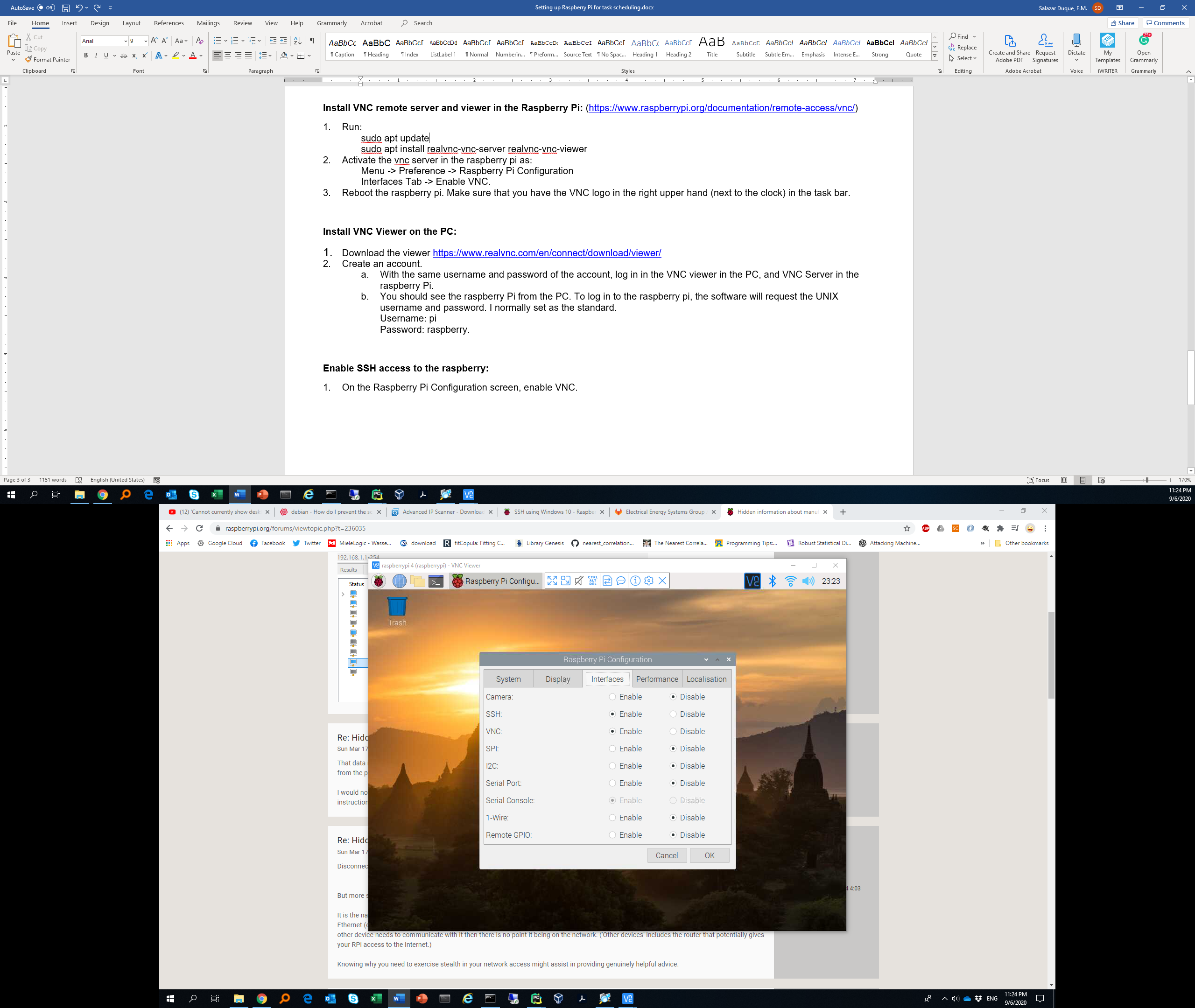
1. Download the viewer <https://www.realvnc.com/en/connect/download/viewer/>
2. Create an account.
   1. With the same username and password of the account, log in in the VNC viewer in the PC, and VNC Server in the raspberry Pi.
   2. You should see the raspberry Pi from the PC. To log in to the raspberry pi, the software will request the UNIX username and password. I normally set as the standard.

Username: pi

Password: raspberry.

**Enable SSH access to the raspberry and fix blank screen in VNC:**

1. On the Raspberry Pi Configuration screen, enable VNC.



1. Troubleshoot blank screen in VNC:

This happens when the raspberry is rebooted and there is no monitor connected.

Access the raspberry via SSH using the following lines in the command window:

ssh [pi@192.168.2.37](mailto:pi@192.168.2.37)

The IP address can change, use the software Advanced IP Scanner (<https://www.advanced-ip-scanner.com/>) to know the raspberry IP. Hint: Memorize the MAC address of the raspberry pi, in order to pin-point the correct ip address.

1. Use the raspberry password (“raspberry”) when prompted. You should see the following response in the console:

C:\Users\20175334>ssh pi@192.168.2.37

pi@192.168.2.37's password:

Linux raspberrypi 5.4.51-v7l+ #1333 SMP Mon Aug 10 16:51:40 BST 2020 armv7l

The programs included with the Debian GNU/Linux system are free software;

the exact distribution terms for each program are described in the

individual files in /usr/share/doc/\*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent

permitted by applicable law.

Last login: Sun Sep 6 23:19:01 2020

SSH is enabled and the default password for the 'pi' user has not been changed.

This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

pi@raspberrypi:~ $

1. Use: sudo raspi-config to enable the raspberry configuration.
2. Go to Advanced Options -> Resolution -> Set the max resolution.
3. Reboot the raspberry pi.
4. You can connect via VNC and you should see the GUI of the raspberry now.

**Transfer files from raspberry to local machine using SSH**

1. This answer is adapted from: <https://askubuntu.com/questions/446724/copy-folders-not-one-file-using-ssh-ubuntu>
2. The command to transfer files is scp and has the following usage:

$ scp -r user@remotehost:/path/to/remote/dir /path/to/local/dir

1. In my case, the command is the following:

$ scp -r mauricio@192.168.0.152:/home/mauricio/Code/knmi\_app/assets/downloads /Users/mauricio/Documents/knmi\_files

1. Type the raspberry password and that’s it, you will see the transfer.
2. Check the space in the raspberry using df -h or df -Bm