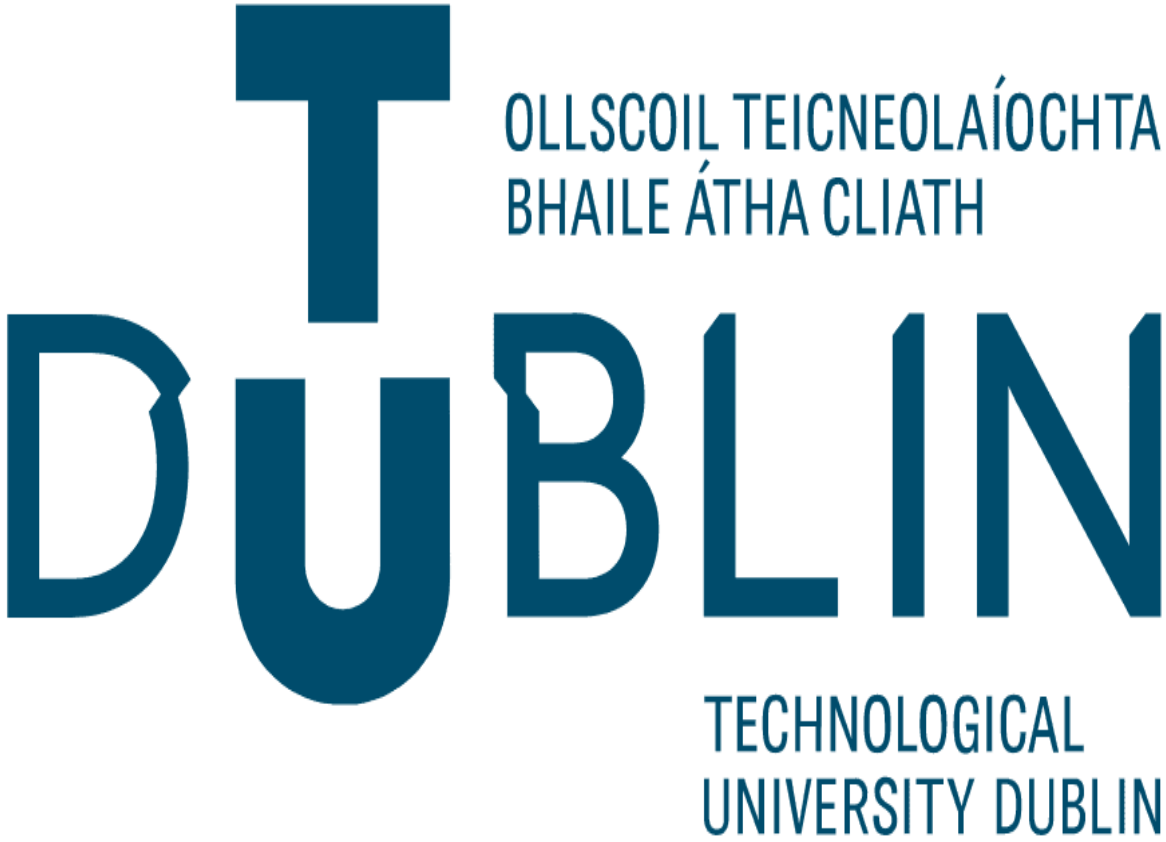


Module: Industrial Computing

Assignment: “Main Assignment 2025”

Name: Daryl Sweeney

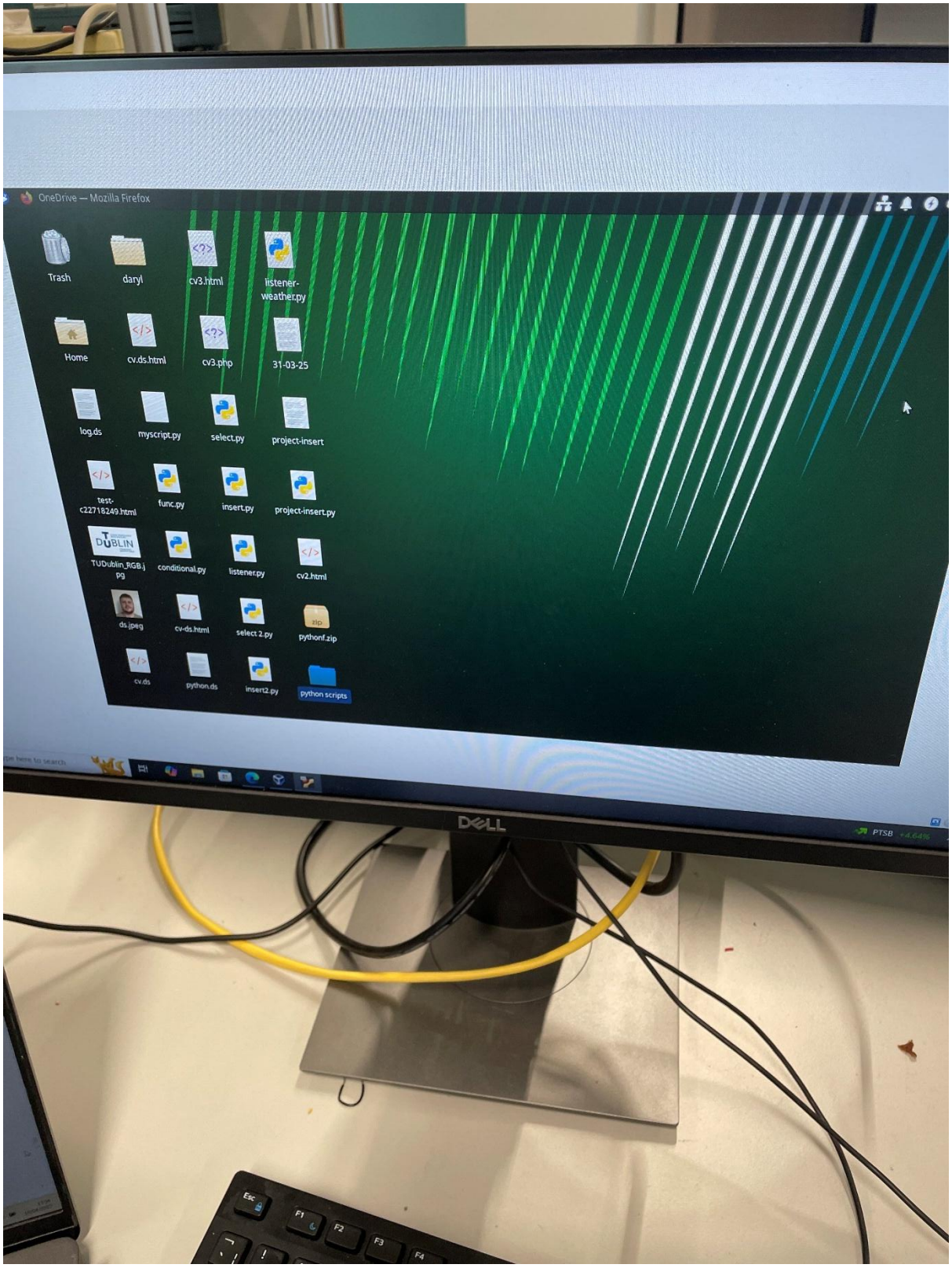
Student number: C22718249

TU705/2

Group B

Table of Contents

Installing Virtual machine on the computer:



Using virtual machine.

- Using the laboratory PC, we logged in as a administrator username as we were granted access for the class
- The username and password were (.\admin) = username and (Tudublin202$00) = password
- We logged in as administrator
- We downloaded “Xubuntu “by searching “Xubuntu download”
- We then selected the United Kingdom version (English)
- We then selected from a list of downloads the 64bit-AMD-24.01.04.iso version
- The iso version was 3.8GB and was off a DVD
- This was the longtime service version
- We then selected oracle virtual desktop icon on the desktop once it was downloaded
- The menu screen came up now and we pressed add
- I put my name in as daryl_sweeney and this was my username
- I set the iso image to the location that the Xubuntu file I downloaded from the site was
- I ticked the box
- I clicked next
- The next page showed how much base memory you could set to the virtual machine and how much was already being used by the PC
- I set the amount of base memory to 3500MB
- The next setting had how much processing power in number of cores that were being used by the PC and the free ones, so I assigned the virtual machine 4 units
- Clicking through I clicked one box then came to the data option and set it to 35GB
- After I went through the process, I finished it and after a while it set up
- I then opened it and after a while the virtual desktop ran
- When it loaded, I double clicked on the Xubuntu download, which was on the virtual desktop as well
- I went through the option process selecting the English version then installed it, using “tudublin” as my username and “tudublinpwd” as my password
- After a while it finished, and I selected restart desktop which was the virtual desktop
- It went blank then a black screen appeared, then I closed it down
- The personal account for the virtual desktop was set up as well as Xubuntu was installed on the virtual desktop
- This could be run remotely through and independent drive external from the actual PC

To install mariadb on virtual machine:

“sudo apt update”

“sudo apt upgrade”

Then,

“sudo apt install mariadb-server”

Then running:

“user@computor:~$sudo mariadb –u root –p”

To install apache on virtual machine:

“sudo apt update”

“sudo apt install apache2”

To install the python3 mariadb connector:

“pip install mariadb”

“sudo apt install python3-pip”

To install Grafana on linux:

“sudo apt update

sudo apt upgrade”

“sudo apt install -y software-properties-common apt-transport-https wget”

“wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add –“

“echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee /etc/apt/sources.list.d/grafana.list”

“sudo apt update

sudo apt install Grafana”

“sudo systemctl start grafana-server

sudo systemctl enable grafana-server”

Setting up databases:

First setting up mariadb by inputting this command into the terminal window:

```
user@computor:~$sudo mariadb -u root -p
```

It will then ask you for a password,

```
mariadb> _
```

and you put in your password.

Then I created WEATHER database using the, “CREATE WEATHER;” command.

This created the database WEATHER.

Then I typed “USE WEATHER” to use the weather database.

After I typed this, this came up:

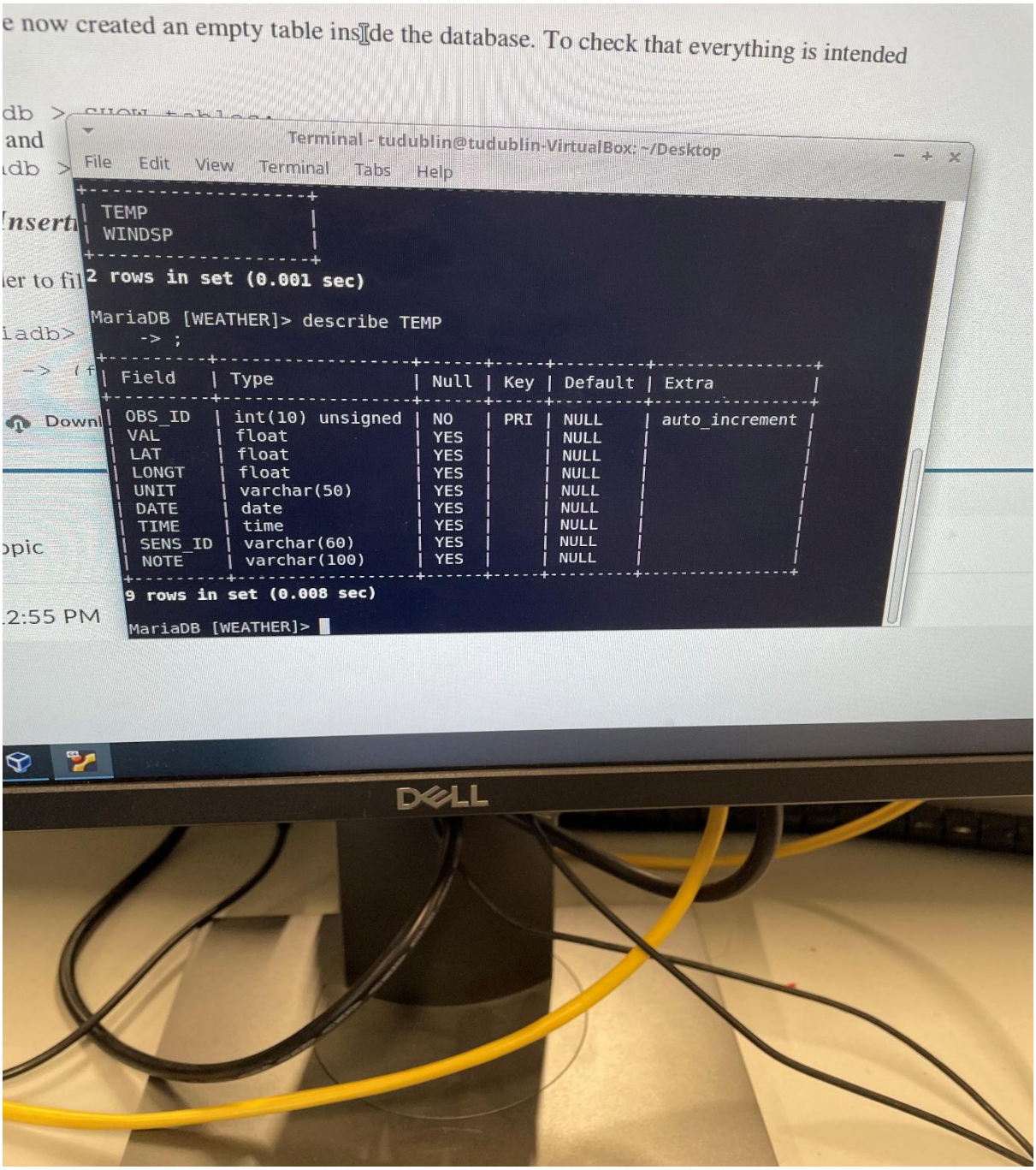
```
mariadb> CREATE TABLE TEMP
-> (
-> OBS_ID int unsigned not null auto_increment primary key,
-> VAL float,
-> LAT float,
-> LONGT FLOAT,
-> UNIT VARCHAR(30),
-> DATE DATE,
-> TIME TIME,
-> SENS_ID VARCHAR(60),
-> NOTE VARCHAR(255)
->);
```

Then I made a table again by creating the WINDSPD table.

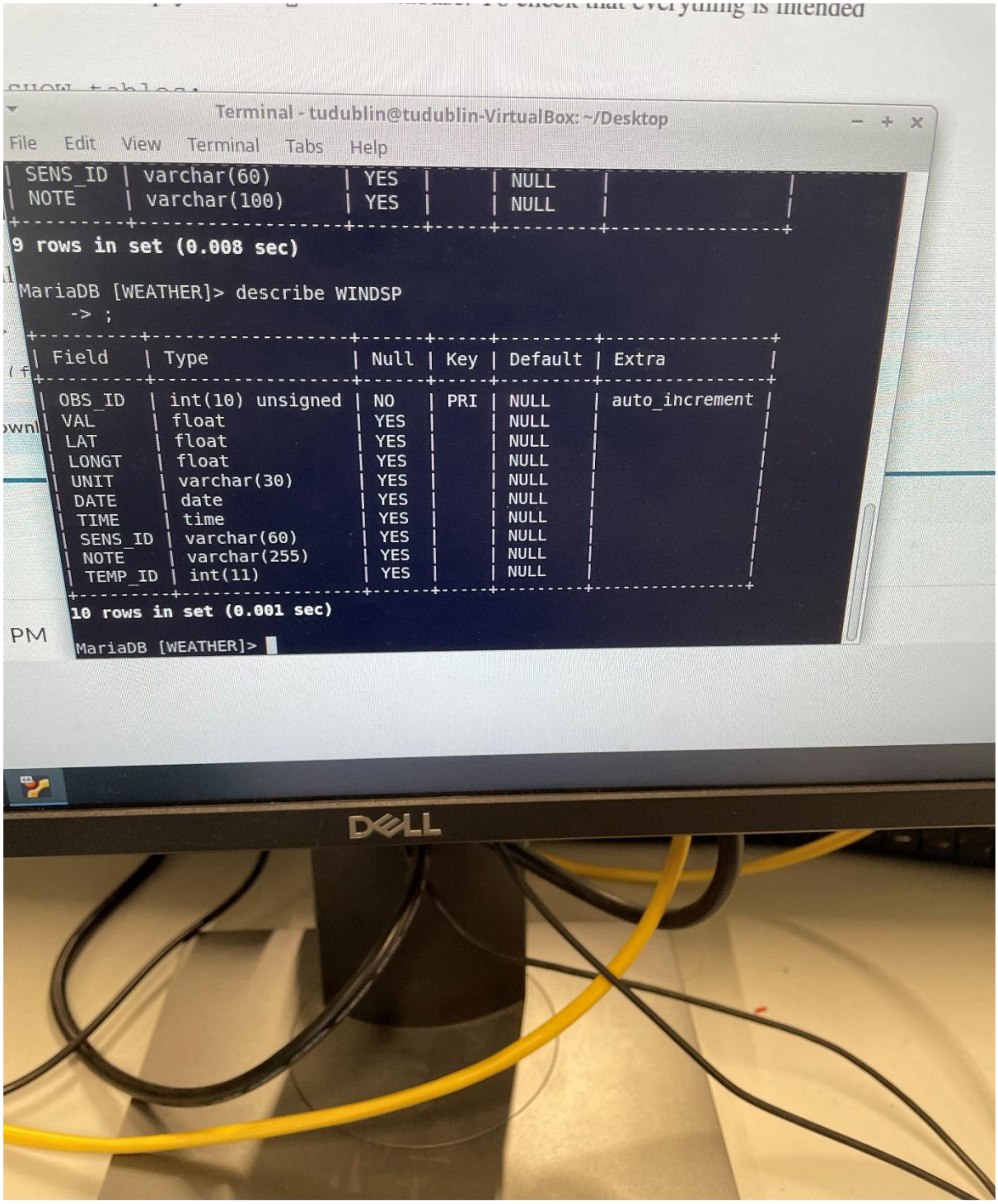
```
mariadb> CREATE TABLE WINDSPD
```

```
-> (
-> OBS_ID int unsigned not null auto_increment primary key,
-> VAL float,
-> LAT float,
-> LONGT FLOAT,
-> UNIT VARCHAR(30),
-> DATE DATE,
-> TIME TIME,
-> SENS_ID VARCHAR(60),
-> NOTE VARCHAR(255)
->);
```

Doing this created the tables TEMP and WINDSPD in mariadb with the headings shown.



This pic showing the headings for TEMP or temperature:



This pic showing the headings in WINDSPD or windspeed:

As you can see the command “describe WINDSPD” is the command for to describe the headings. It can be any table in place.

On mariadb we also learned using the “employees” database and using the “employees_data” table how to manually insert entries using mariadb commands.

```
mariadb> INSERT INTO employee_data
    -> (f_name, l_name, title, age, yos, salary, perks,  email)
    -> values("Bruce", "Lee", "CIO", 28, 4, 200000,  50000,
"bruce@homepc.com");
```

This command “INSERT INTO” table would let you clarify the headings you wanted to use, then entering the values into them headings.

```
mariadb> GRANT ALL PRIVILEGES ON *.* TO  tudublin@localhost
    ->     IDENTIFIED BY 'tudublinpwd' WITH GRANT  OPTION;
mariadb> GRANT ALL PRIVILEGES ON *.* TO  tudublin@'%'
    ->     IDENTIFIED BY 'some_pass' WITH GRANT  OPTION;
mariadb> GRANT RELOAD,PROCESS ON *.* TO admin@localhost;
mariadb> GRANT USAGE ON *.* TO  dummy@localhost;
```

These GRANT statements set up three new users:

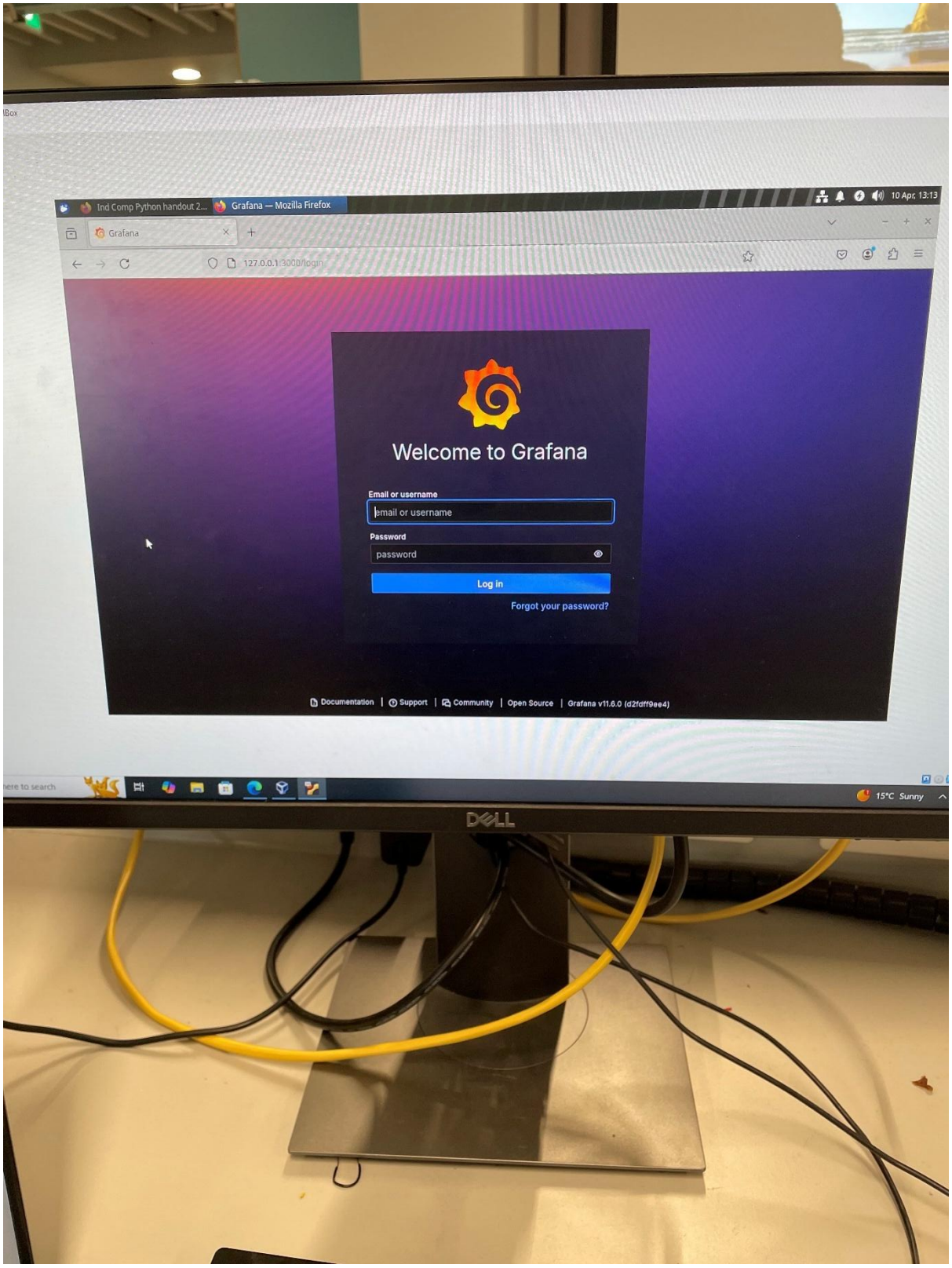
```
monty
```

These commands grant privileges to specific users on a database.

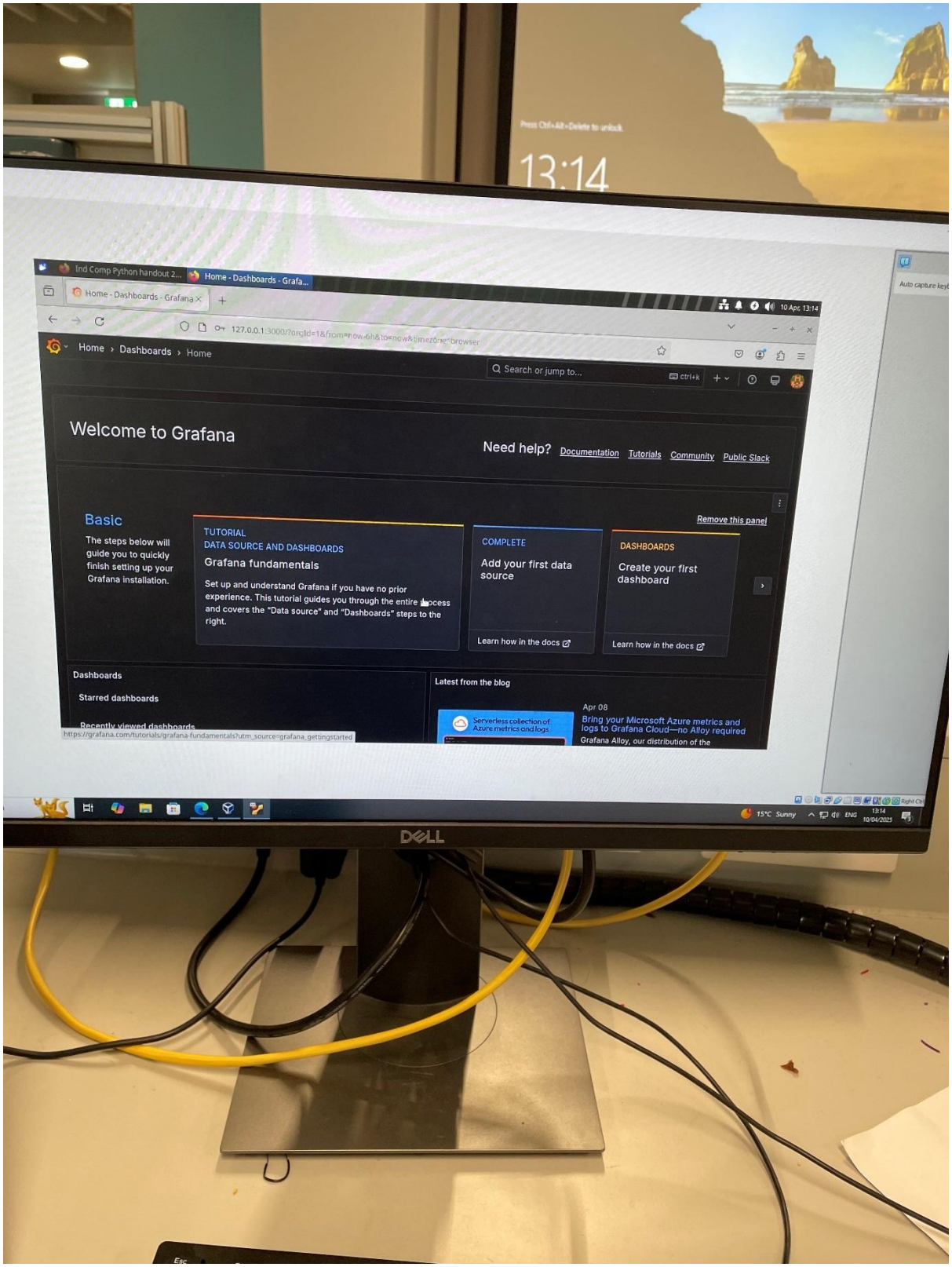
```
mariadb> SELECT * FROM employee_data;
```

This will return all entries and all headings from a table.

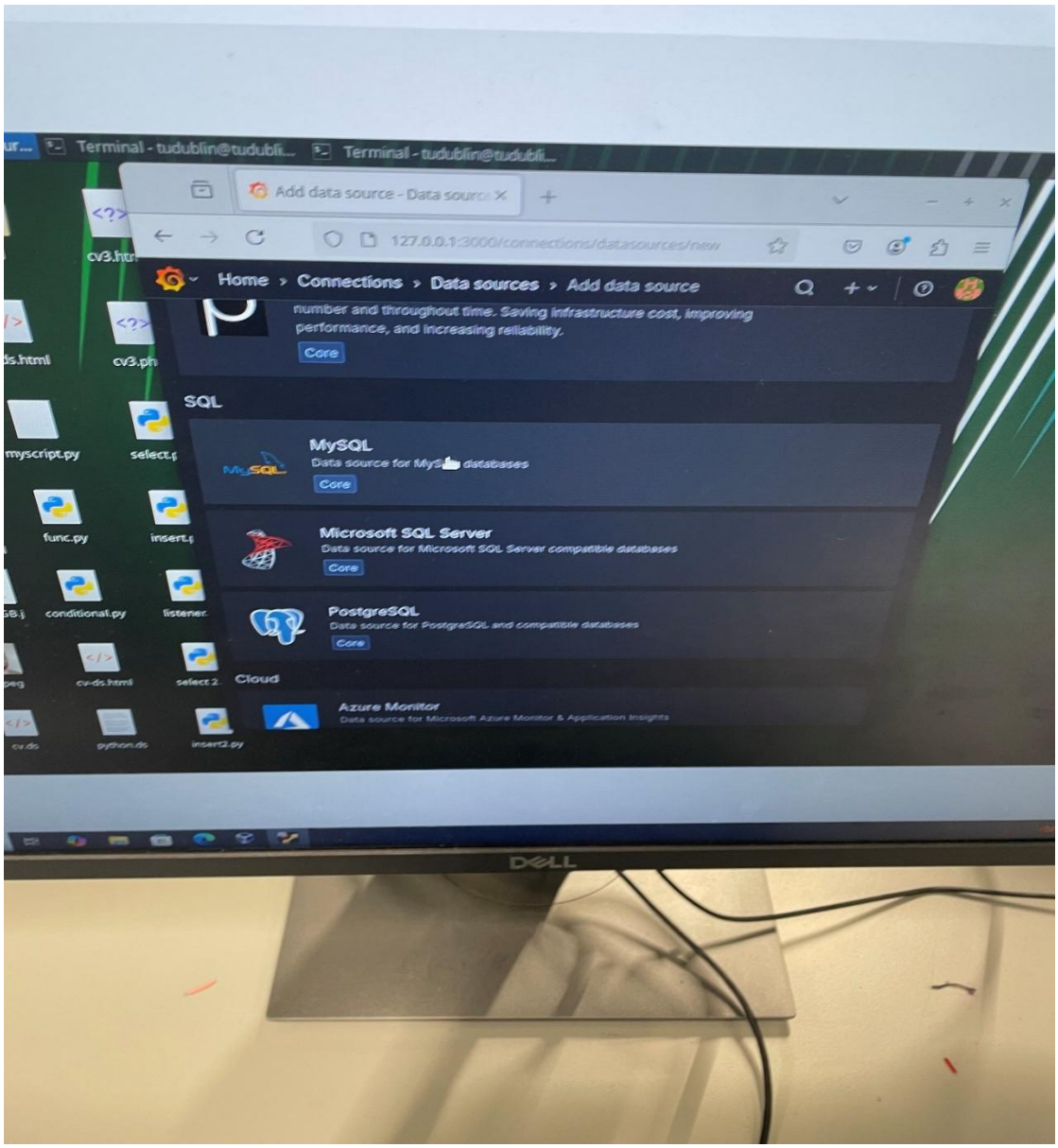
Grafana screenshots:



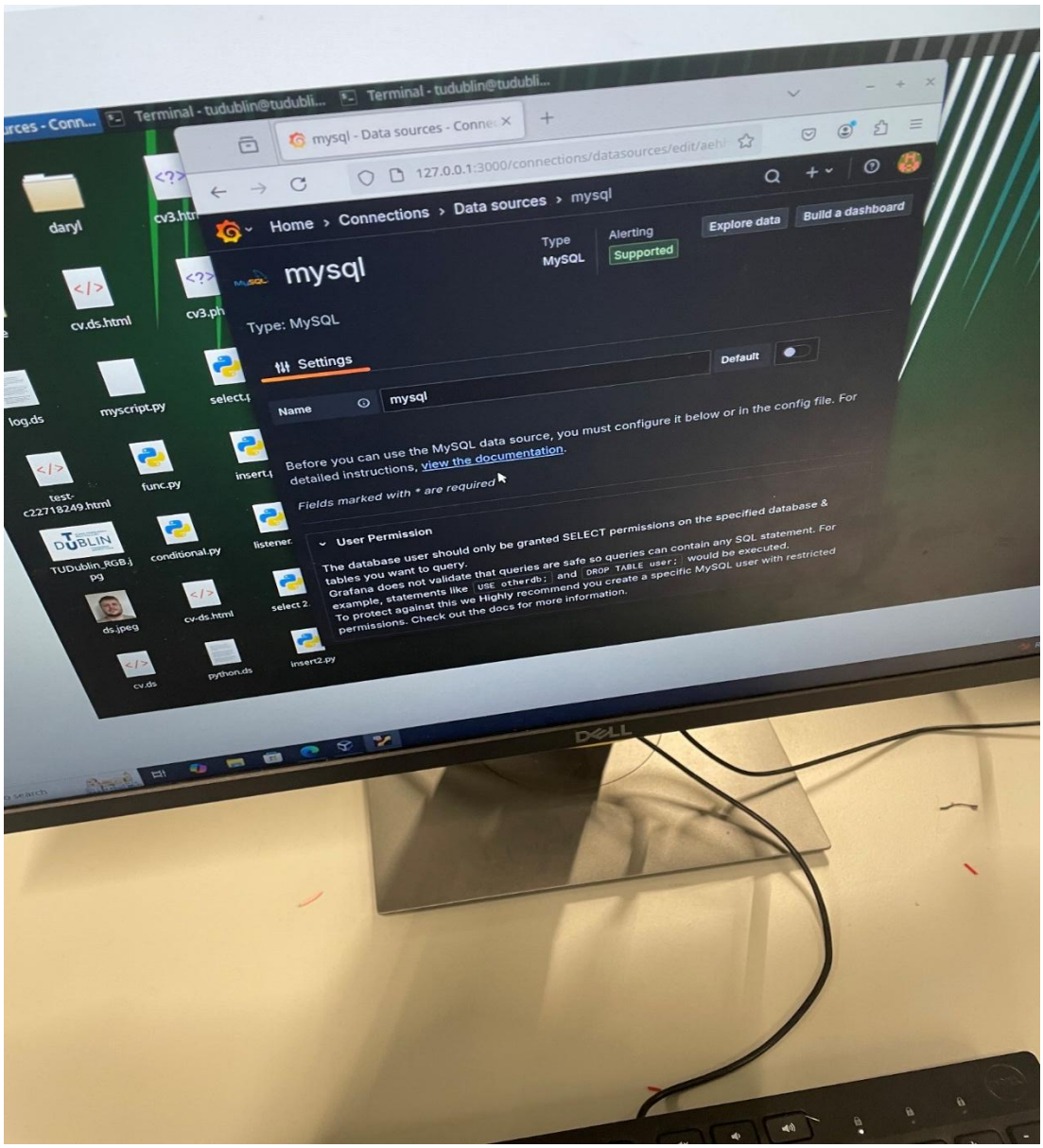
This is the Grafana login. You get this when you type 127.0.0.1:3001/ into the web browser.



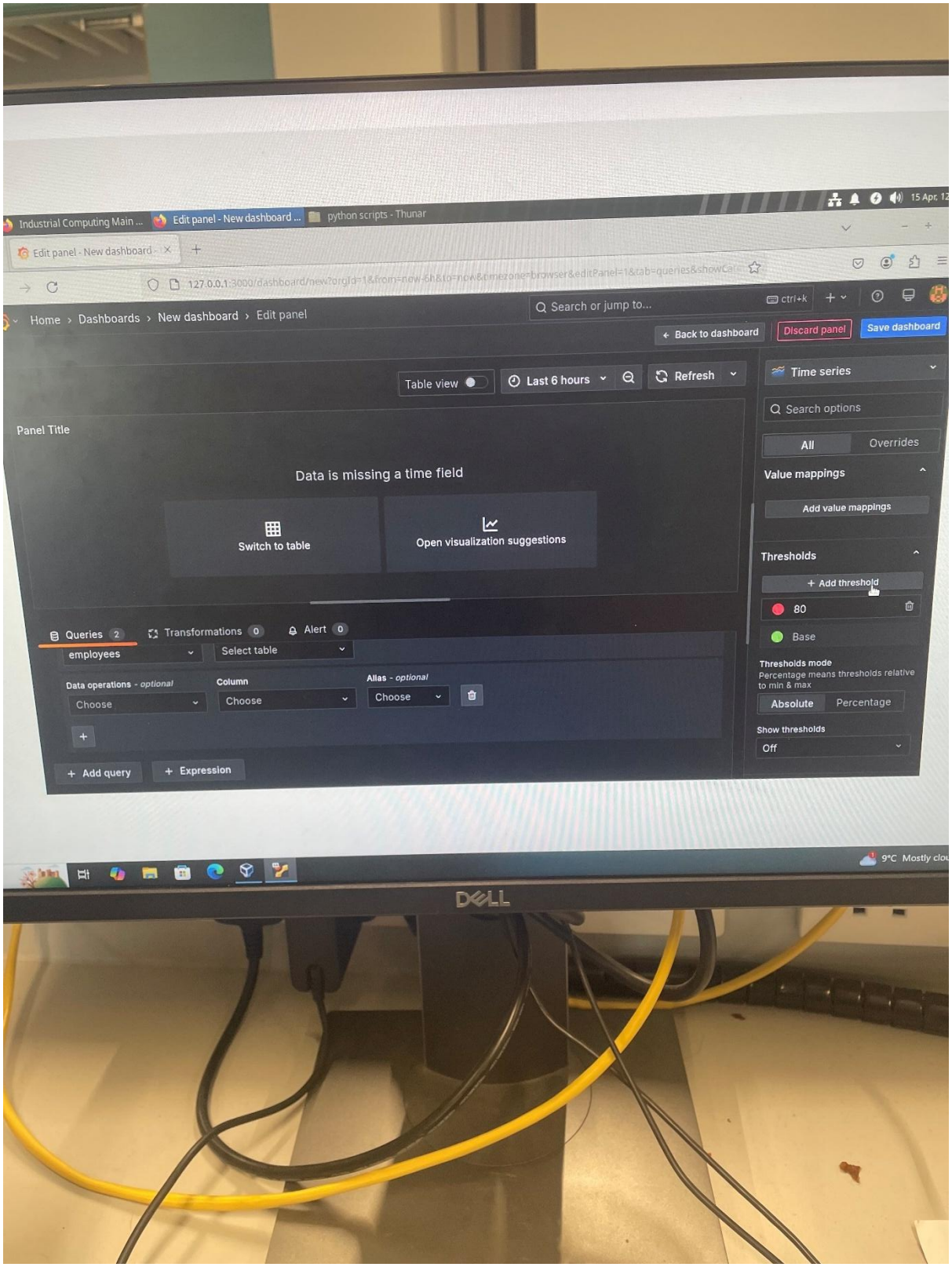
This is the Grafana main screen that comes up after entering the password. You can see previous databases and tables here and you can guide into making a connection to a database or table.



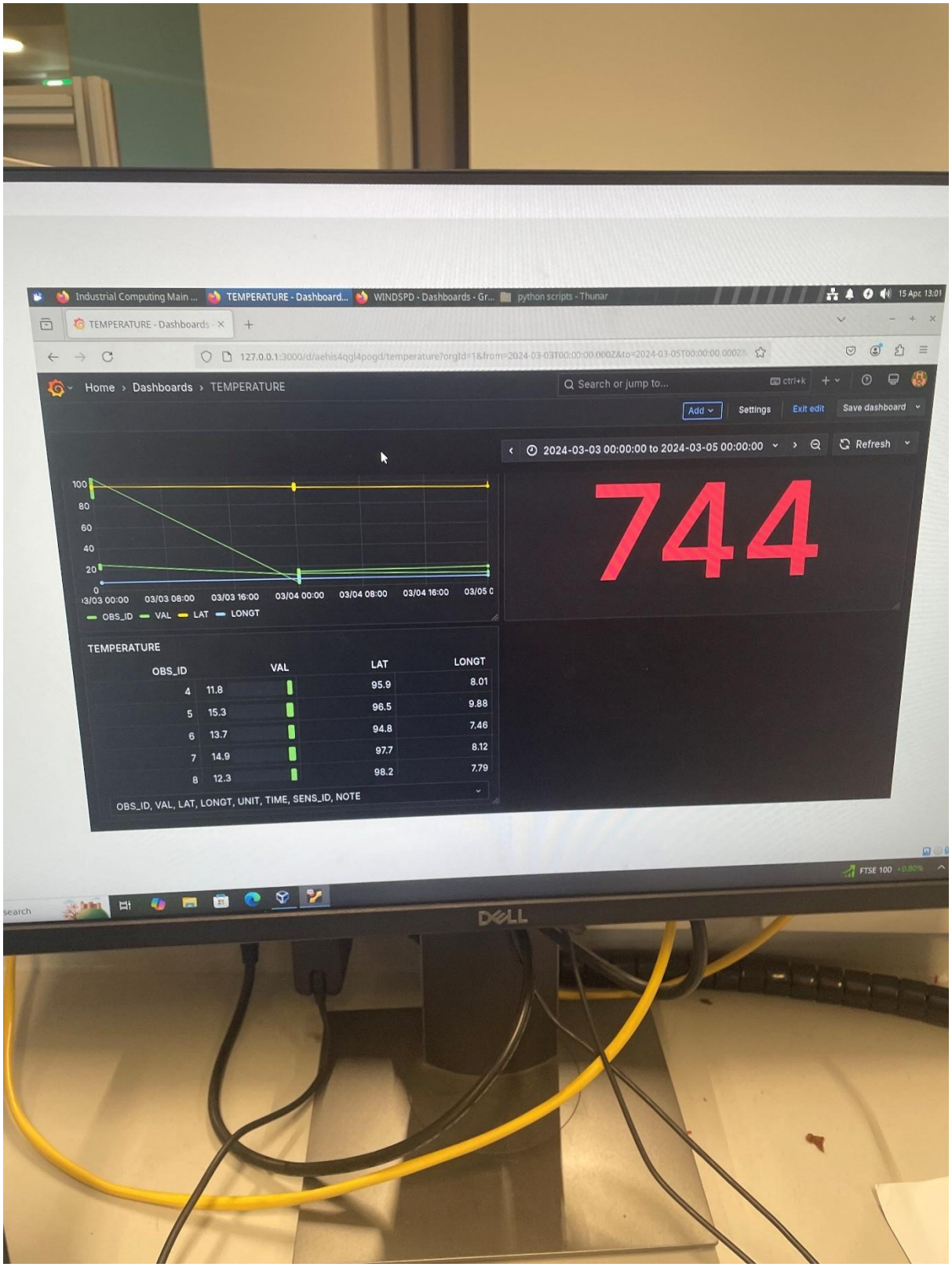
This is the next screen where you pick what type of interface you want.



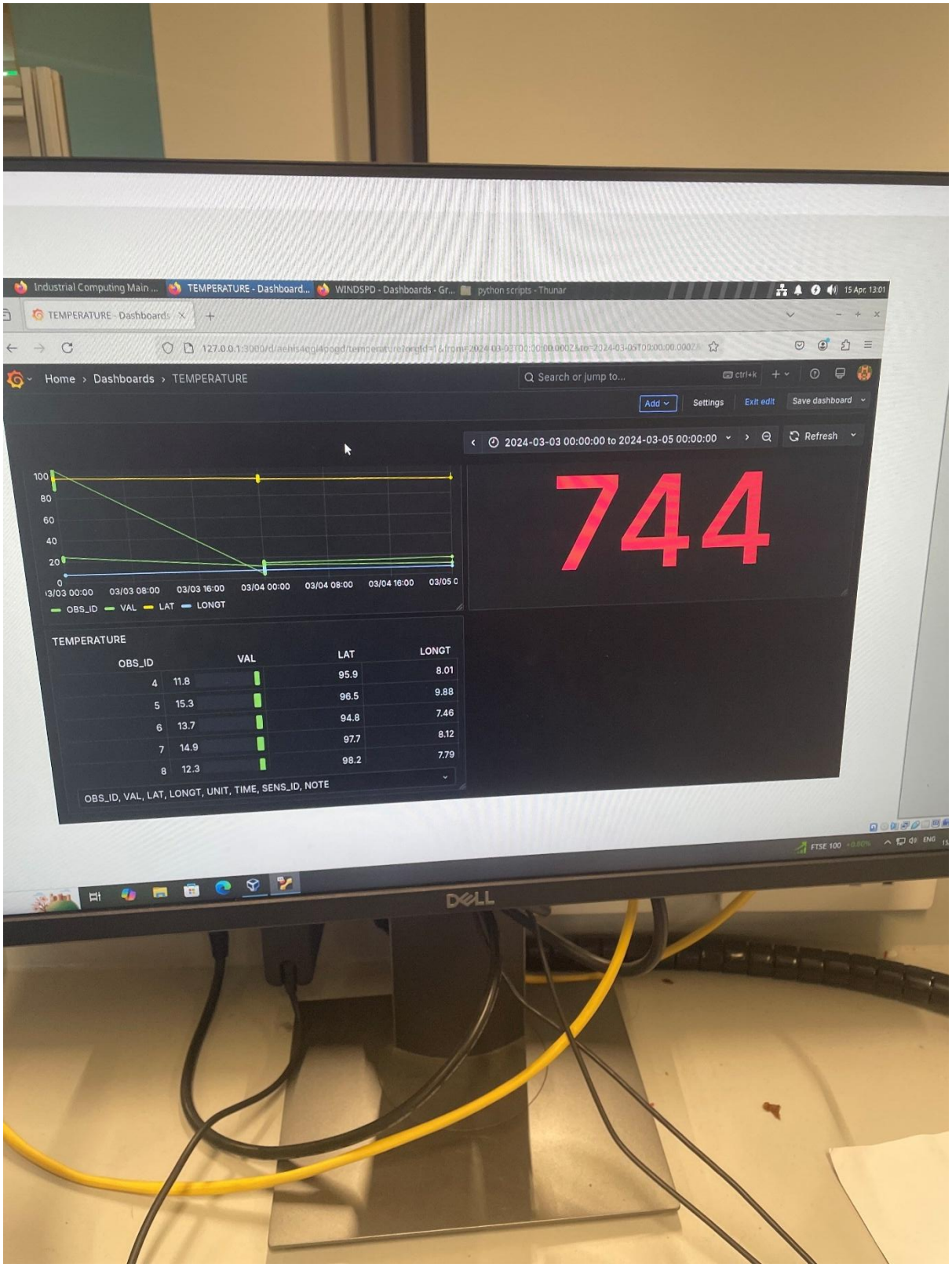
An example of setting up an interface.



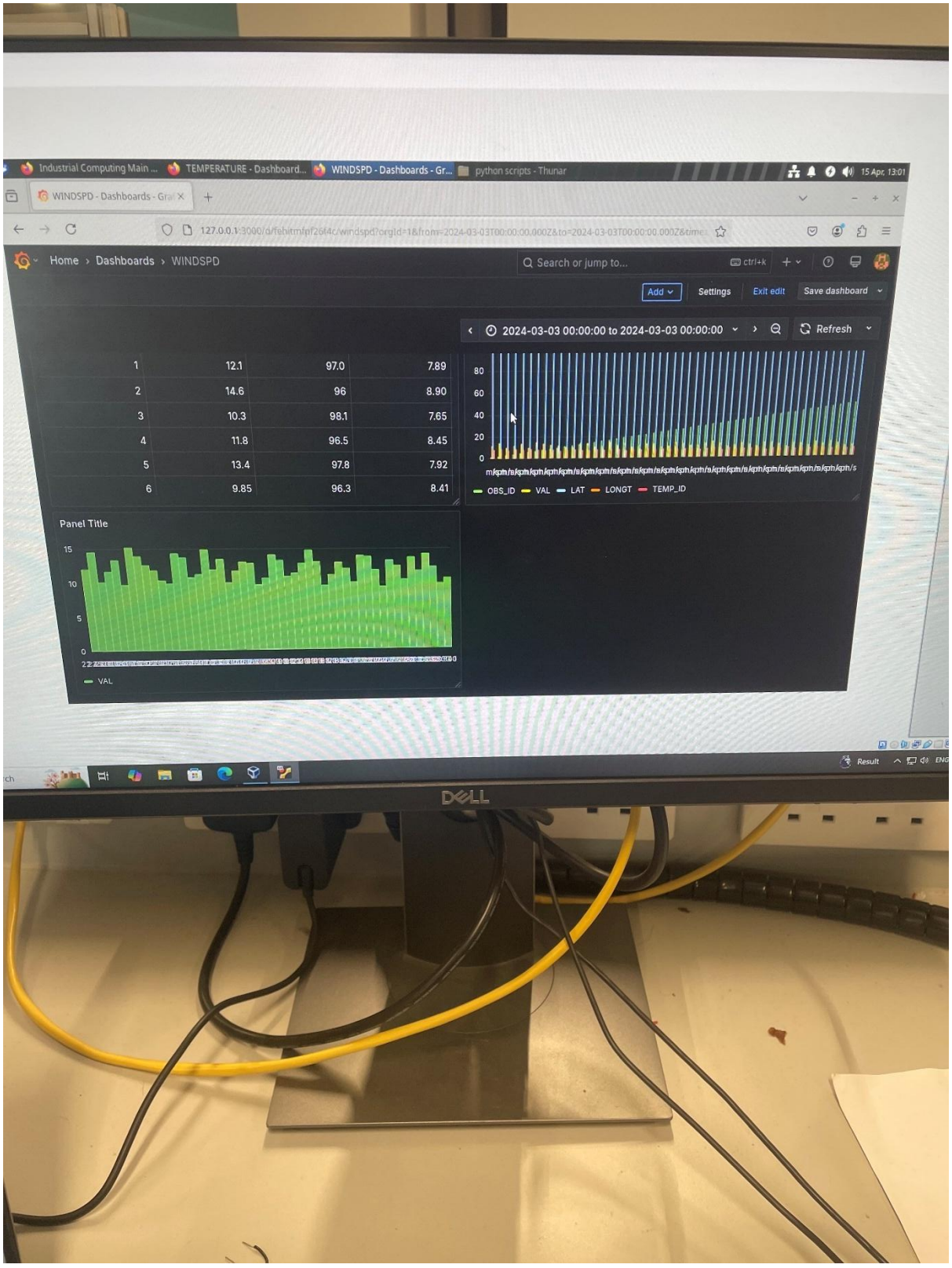
This is the Grafana dashboard in which statistics like graphs can be connected to the database or table.



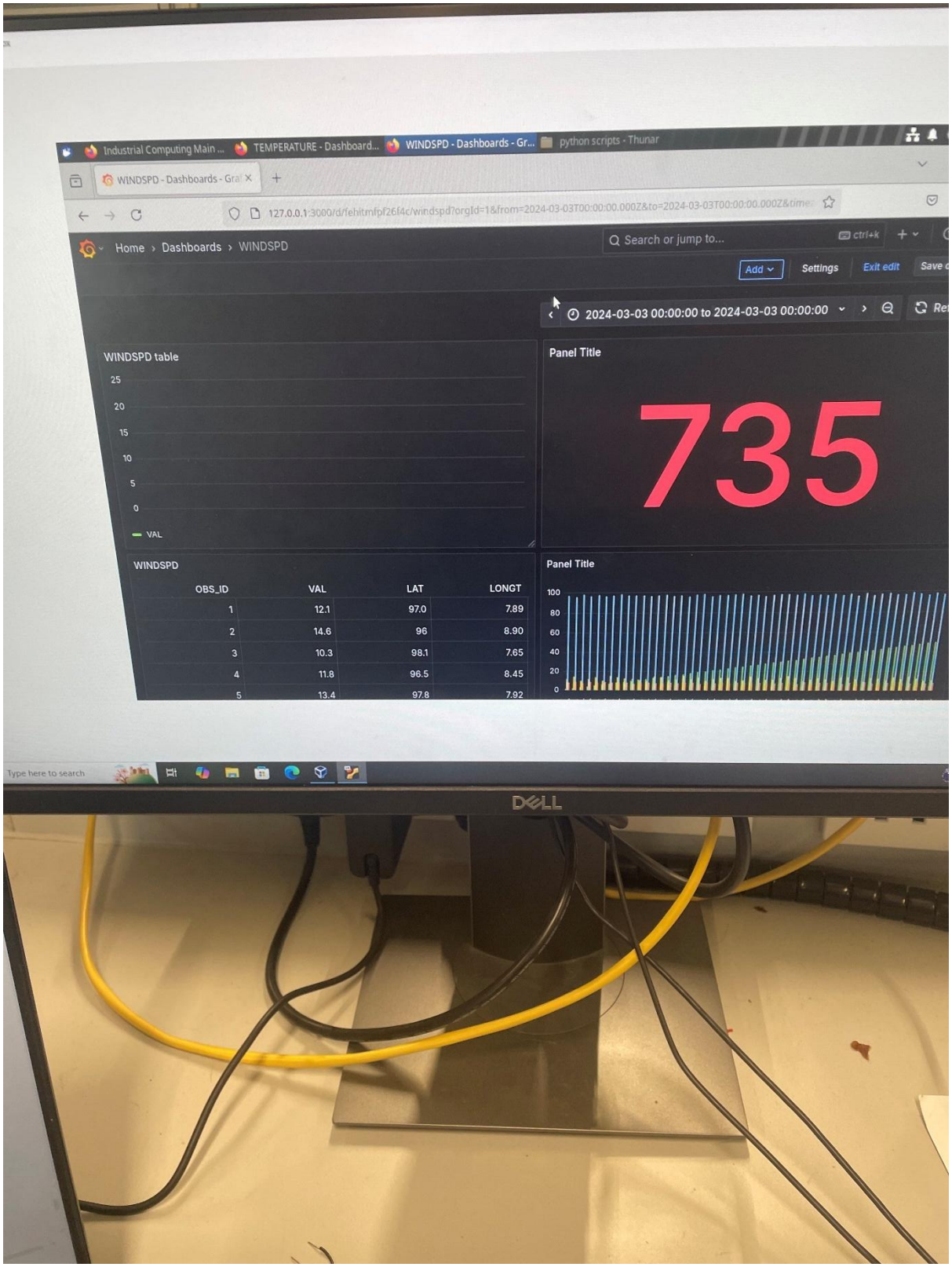
Tables made for “TEMP” table in WEATHER database.


Industrial Computing Main ...
TEMPERATURE - Dashboard...
WINDSPD - Dashboards - Gr...
python scripts - Thunar
15 Apr 13:01
TEMPERATURE - Dashboards
Home › Dashboards › TEMPERATURE
Search or jump to...
Add
Settings
Exit edit
Save dashboard
2024-03-03 00:00:00 to 2024-03-05 00:00:00
Refresh
744
100
80
60
40
20
0
03/03 00:00
03/03 08:00
03/03 16:00
03/04 00:00
03/04 08:00
03/04 16:00
OBS_ID
VAL
LAT
LONGT
TEMPERATURE
OBS_ID VAL LAT LONGT
4 11.8 95.9 8.01
5 15.3 96.5 9.88
6 13.7 94.8 7.46
7 14.9 97.7 8.12
8 12.3 98.2 7.79
OBS_ID, VAL, LAT, LONGT, UNIT, TIME, SENS_ID, NOTE
FTSE 100
DELL

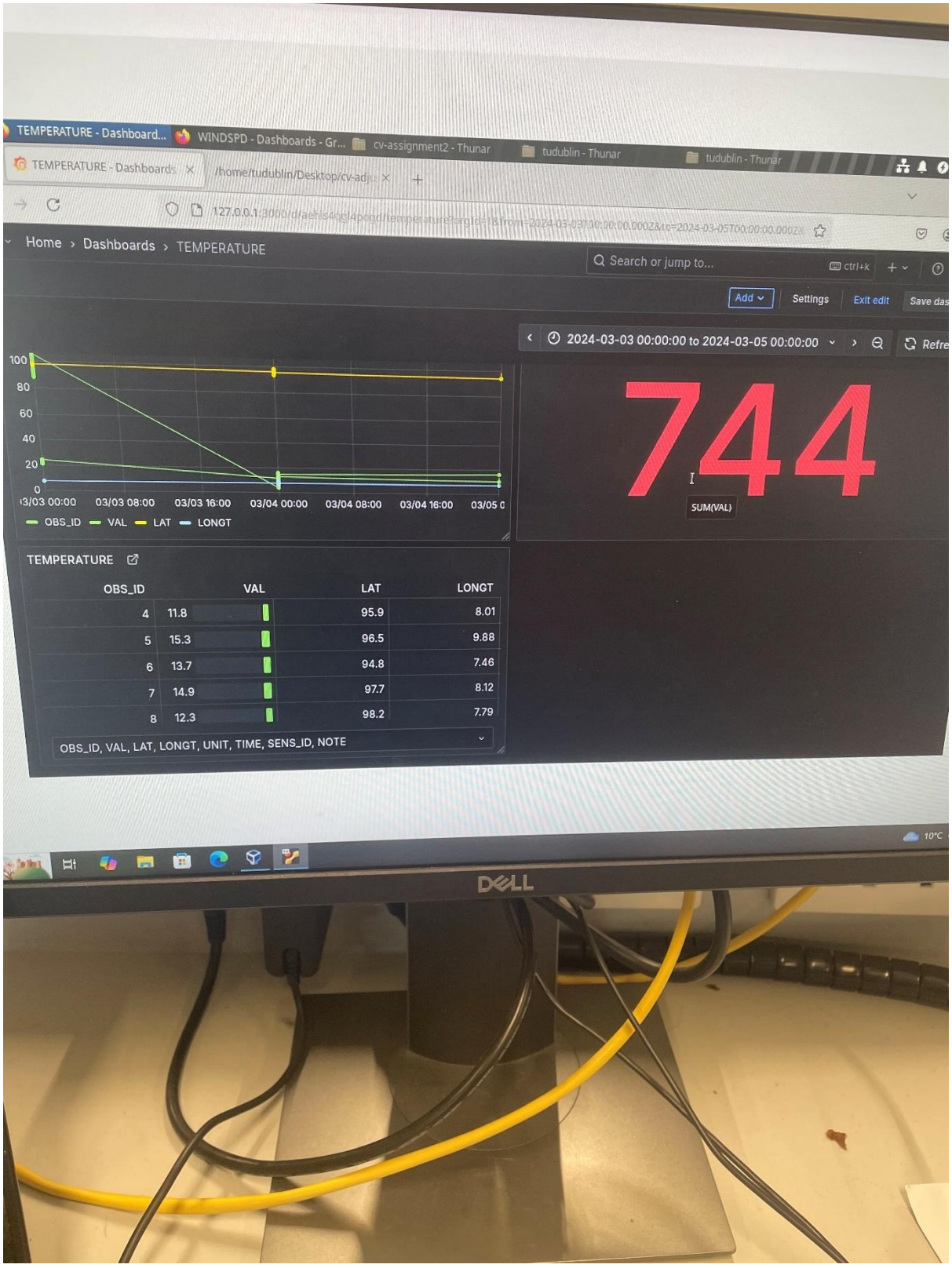
Another pic.



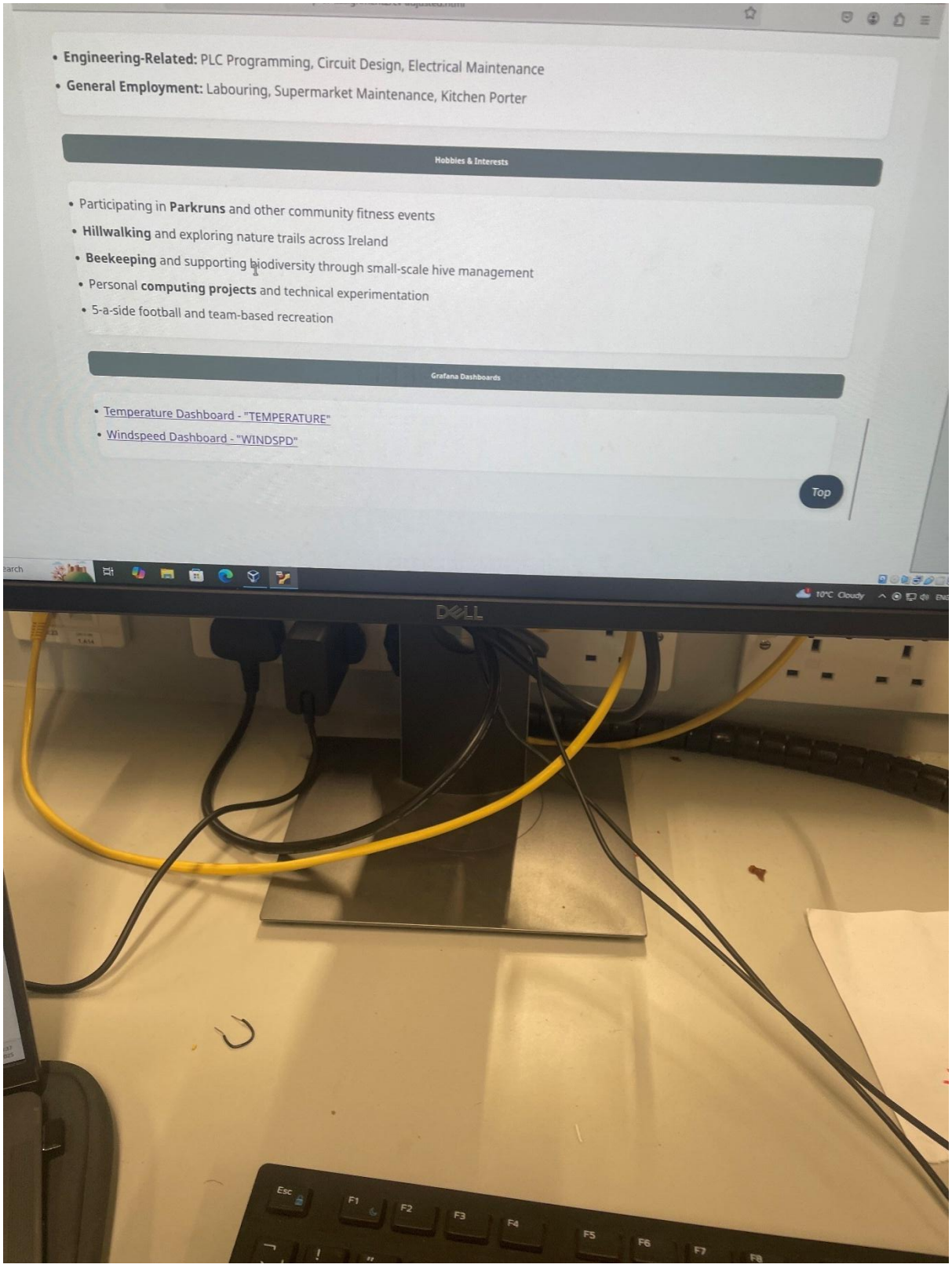
“WINDSPD” stats.



Example “WINDSPD” stats.



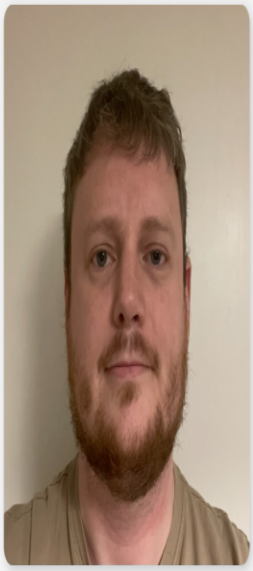
“TEMP” stats with links to CV(hard to see unless its in front of you).



CV with links to both tables.

The cv using HTML:

Curriculum Vitae



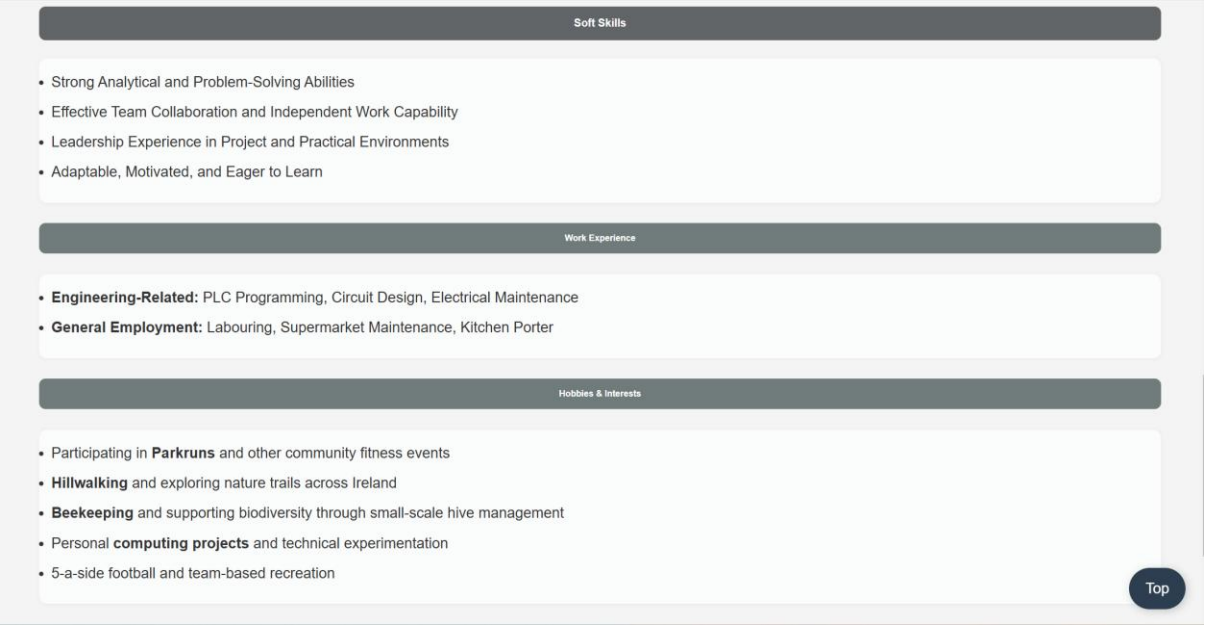
- **Name:** Daryl Sweeney
- **Age:** 31
- **Qualification:** Higher Certificate in Electrical and Control Engineering (Level 6)
- **Future Education:** Progressing to Bachelor's Degree in TU705
- **Status:** Actively Seeking Employment in Electrical Engineering
- **LinkedIn:** linkedin.com/in/daryl-sweeney

About Me

I am a recent graduate of Technological University Dublin with a Higher Certificate in Electrical and Control Engineering. I am enthusiastic about launching a career in the electrical engineering field and applying my practical knowledge, technical skills, and problem-solving abilities in a professional setting.

First bit of cv.

Second bit of cv.



Third bit of cv.

Work Experience

- **Engineering-Related:** PLC Programming, Circuit Design, Electrical Maintenance
- **General Employment:** Labouring, Supermarket Maintenance, Kitchen Porter

Hobbies & Interests

- Participating in **Parkruns** and other community fitness events
- **Hillwalking** and exploring nature trails across Ireland
- **Beekeeping** and supporting biodiversity through small-scale hive management
- Personal **computing projects** and technical experimentation
- 5-a-side football and team-based recreation

Grafana Dashboards

- Temperature Dashboard - "TEMPERATURE"
- Windspeed Dashboard - "WINDSPD"

Top

Fourth bit of cv.

The html code for the cv:

```
<!DOCTYPE html>
<html style="background-color:#F4F4F4; color:#333;">
<head>
 <title>Curriculum Vitae - Daryl Sweeney</title>
 <style>
  body {
   font-family: 'Open Sans', sans-serif;
   margin: 30px;
   padding: 20px;
   line-height: 1.6;
  }

  h1, h2, h3, h4, h5, h6 {
   font-family: 'Montserrat', sans-serif;
   text-align: center;
   padding: 10px;
   color: white;
   border-radius: 8px;
  }

  h1 { background-color: #2C3E50; }
  h2 { background-color: #34495E; }
  h3 { background-color: #566573; }
  h4 { background-color: #5D6D7E; }
```

```
h5 { background-color: #626567; }
h6 { background-color: #707B7C; }

ul, p {
  font-size: 18px;
  background-color: #FBFCFC;
  padding: 15px;
  border-radius: 10px;
  box-shadow: 1px 1px 5px rgba(0,0,0,0.05);
}

ul li {
  margin-bottom: 8px;
}

table {
  width: 100%;
  margin-top: 20px;
}

img {
  border-radius: 10px;
  display: block;
  margin: auto;
  box-shadow: 2px 2px 8px rgba(0, 0, 0, 0.2);
```

```
}

#myBtn {
  display: none;
  position: fixed;
  bottom: 20px;
  right: 20px;
  background-color: #2C3E50;
  color: white;
  border: none;
  padding: 15px 20px;
  cursor: pointer;
  font-size: 18px;
  border-radius: 30px;
  box-shadow: 2px 2px 10px rgba(0, 0, 0, 0.2);
  transition: 0.3s;
}

#myBtn:hover {
  background-color: #1B2631;
  transform: scale(1.1);
}

.logo {
  text-align: center;
```

```
    margin-top: 20px;
   }

   .logo img {
    height: 80px;
    transition: transform 0.3s;
   }

   .logo img:hover {
    transform: scale(1.1);
   }
  </style>
</head>
<body>

<h1>Curriculum Vitae</h1>

<div class="logo">
  <a href="https://www.tudublin.ie/" target="_blank">
   <img src="tud-logo.png" alt="TU Dublin Logo">
  </a>
</div>

<table>
  <tr>
```

```
  <td><img src="ds.jpeg" height="300" width="300" alt="Profile Picture"></td>
  <td>
   <ul>
    <li><strong>Name:</strong> Daryl Sweeney</li>
    <li><strong>Age:</strong> 31</li>
    <li><strong>Qualification:</strong> Higher Certificate in Electrical and Control Engineering (Level 6)</li>
    <li><strong>Future Education:</strong> Progressing to Bachelor's Degree in TU705</li>
    <li><strong>Status:</strong> Actively Seeking Employment in Electrical Engineering</li>
    <li><strong>LinkedIn:</strong> <a href="https://www.linkedin.com/in/daryl-sweeney-b662bb250/" target="_blank">linkedin.com/in/daryl-sweeney</a></li>
   </ul>
  </td>
 </tr>
</table>

<h2>About Me</h2>
<p>
 I am a recent graduate of Technological University Dublin with a Higher Certificate in Electrical and Control Engineering.
 I am enthusiastic about launching a career in the electrical engineering field and applying my practical knowledge, technical skills, and problem-solving abilities in a professional setting.
</p>
```

<h3>Technical Skills</h3>

Programming: C++, C, Assembly, Python, MATLAB

Electrical & Electronics: Electrical Wiring, Electronics, Electrical Instrumentation

Engineering Subjects: Engineering Mathematics, Power Engineering, Control Engineering

Automation: PLCs, Allen Bradley PLCs, SCADA, Graphworx

Systems & Networking: Robotics, Communication Networks, Networking

Databases: RDBMSs, MariaDB

Design & Hardware: Microcontrollers, AutoCAD, Electrical Design

Other Electrical: Electricity, DC Motors, 3-Phase Motors, Transformers, Power Systems, Synchronous Machines, Asynchronous Machines, 3-Phase Systems

Tools & Certification: ECDL (ICDL), Grafana

<h4>Qualifications</h4>

Higher Certificate in Electrical and Control Engineering (Level 6), TU Dublin

QQI Level 5 Preliminary Engineering – Merit in Mathematics (Coláiste Dhúlaigh)

QQI Level 3 Certificate in Beginners Beekeeping

Leaving Certificate

Junior Certificate

<h5>Soft Skills</h5>

Strong Analytical and Problem-Solving Abilities

Effective Team Collaboration and Independent Work Capability

Leadership Experience in Project and Practical Environments

Adaptable, Motivated, and Eager to Learn

<h6>Work Experience</h6>

Engineering-Related: PLC Programming, Circuit Design, Electrical Maintenance

General Employment: Labouring, Supermarket Maintenance, Kitchen Porter

<h6>Hobbies & Interests</h6>

Participating in Parkruns and other community fitness events

```
 <li><strong>Hillwalking</strong> and exploring nature trails across Ireland</li>

 <li><strong>Beekeeping</strong> and supporting biodiversity through small-scale hive management</li>

 <li>Personal <strong>computing projects</strong> and technical experimentation</li>

 <li>5-a-side football and team-based recreation</li>

</ul>


<h6>Grafana Dashboards</h6>

<ul>

 <li><a href="http://127.0.0.1:3000/d/aehis4qgl4pogd/temperature?orgId=1&from=2024-03-03T00:00:00.000Z&to=2024-03-05T00:00:00.000Z&timezone=browser&tab=queries">Temperature Dashboard - "TEMPERATURE"</a></li>

 <li><a href="http://127.0.0.1:3000/d/fehitmfpf26f4c/windspd?orgId=1&from=2024-03-03T00:00:00.000Z&to=2024-03-03T00:00:00.000Z&timezone=browser">Windspeed Dashboard - "WINDSPD"</a></li>

</ul>


<button id="myBtn" onclick="topFunction()">Top</button>

<script>

 var mybutton = document.getElementById("myBtn");

 window.onscroll = function() {
```

```
  if (document.body.scrollTop > 100 ||
document.documentElement.scrollTop > 100) {
    mybutton.style.display = "block";
  } else {
    mybutton.style.display = "none";
  }
 };

 function topFunction() {
  window.scrollTo({ top: 0, behavior: 'smooth' });
 }
</script>

</body>
</html>
```

I used hex colour coding for reference and I tried to get the right colour. Theres a link to the TUD website on the TUD logo image. I put a link in of my linkedin page. I linked both the tables from the database I made and it should work from the admin computer. I also put a list, an unordered list and a button that when pressed goes back to the top of the page. The selfie pic of me isn't bad and is neutral. I also put in my mathworks onramp Simulink and matlab certs and mentioned I did the out of date ecdl. The certs are available to view on the linkedin page.

Python scripting:

First we learned as shown how to insert values manually then we learned how to insert using a python script to do it externally.

I wanted to insert a large number of entries into the database to build a good picture. The code I used enters a set called entries using this python script and the set entries has 80 entries which worked. I commented it aswell to provide clarity:

Python scripts are run by typing the python3 command the a space then the full python filename.

“Python3 file.py” say.

Heres the insert-many-entries-python-code for “TEMP”:

```
# Module Imports

import mariadb   # This imports the MariaDB connector to interact with the database

import sys      # This allows us to exit the program if something goes wrong


# Try to connect to the MariaDB database

try:

   conn = mariadb.connect(

      user="tudublin",        # username for the database

      password="tudublinpwd",  # password for the database

      host="127.0.0.1",       # database is hosted locally (localhost)

      port=3306,              # default MariaDB port

      database="WEATHER"      # the name of the database we're connecting to
```

```
    )
except mariadb.Error as e:
    # If connection fails, print the error and exit the program
    print(f"Error connecting to MariaDB Platform: {e}")
    sys.exit(1)

# Create a cursor object to interact with the database
cur = conn.cursor()

# A list of sample weather data entries to insert into the database
# Each entry includes: temperature, latitude, longitude, unit, date, time, sensor ID, and a note
entries = [
    (12.12, 97.0087, 7.8907, 'm/s', '2024-03-03', '22:45:01', '25rt', 'reading 1'),
    (14.55, 96.0000, 8.9000, 'kph', '2024-03-03', '22:46:00', '25rt', 'reading 2'),
    (10.25, 98.1234, 7.6543, 'm/s', '2024-03-03', '22:47:00', '25rt', 'reading 3'),
    (11.78, 96.4523, 8.4523, 'm/s', '2024-03-03', '22:48:00', '25rt', 'reading 4'),
    (13.40, 97.7832, 7.9231, 'kph', '2024-03-03', '22:49:00', '25rt', 'reading 5'),
    (9.85, 96.3021, 8.4120, 'm/s', '2024-03-03', '22:50:00', '25rt', 'reading 6'),
    (15.20, 97.2341, 7.9341, 'kph', '2024-03-03', '22:51:00', '25rt', 'reading 7'),
    (13.95, 98.1230, 7.2345, 'm/s', '2024-03-03', '22:52:00', '25rt', 'reading 8'),
    (12.67, 96.8734, 8.1234, 'kph', '2024-03-03', '22:53:00', '25rt', 'reading 9'),
    (11.90, 97.0001, 7.0001, 'm/s', '2024-03-03', '22:54:00', '25rt', 'reading 10'),
    (10.44, 97.1234, 7.5432, 'kph', '2024-03-03', '22:55:00', '25rt', 'reading 11'),
    (9.99, 96.6543, 8.0987, 'm/s', '2024-03-03', '22:56:00', '25rt', 'reading 12'),
```

(14.33, 97.0987, 7.3456, 'm/s', '2024-03-03', '22:57:00', '25rt', 'reading 13'),

(13.77, 98.0000, 7.1000, 'kph', '2024-03-03', '22:58:00', '25rt', 'reading 14'),

(10.88, 96.7890, 8.1111, 'm/s', '2024-03-03', '22:59:00', '25rt', 'reading 15'),

(11.33, 97.3210, 7.3210, 'kph', '2024-03-03', '23:00:00', '25rt', 'reading 16'),

(14.88, 96.1111, 8.2222, 'm/s', '2024-03-03', '23:01:00', '25rt', 'reading 17'),

(12.00, 97.8888, 7.9999, 'm/s', '2024-03-03', '23:02:00', '25rt', 'reading 18'),

(13.50, 98.4321, 7.1234, 'kph', '2024-03-03', '23:03:00', '25rt', 'reading 19'),

(10.10, 96.2000, 8.1000, 'm/s', '2024-03-03', '23:04:00', '25rt', 'reading 20'),

(11.80, 97.2100, 7.7100, 'm/s', '2024-03-03', '23:05:00', '25rt', 'reading 21'),

(13.10, 96.8900, 8.2100, 'kph', '2024-03-03', '23:06:00', '25rt', 'reading 22'),

(12.90, 97.6700, 7.6700, 'm/s', '2024-03-03', '23:07:00', '25rt', 'reading 23'),

(9.80, 98.2000, 7.8700, 'm/s', '2024-03-03', '23:08:00', '25rt', 'reading 24'),

(10.75, 96.7900, 8.0900, 'kph', '2024-03-03', '23:09:00', '25rt', 'reading 25'),

(14.12, 97.2500, 7.2500, 'm/s', '2024-03-03', '23:10:00', '25rt', 'reading 26'),

(13.32, 96.0000, 8.0000, 'kph', '2024-03-03', '23:11:00', '25rt', 'reading 27'),

(10.99, 97.4500, 7.5000, 'm/s', '2024-03-03', '23:12:00', '25rt', 'reading 28'),

(11.45, 98.1000, 7.3000, 'kph', '2024-03-03', '23:13:00', '25rt', 'reading 29'),

(12.88, 97.6000, 7.9000, 'm/s', '2024-03-03', '23:14:00', '25rt', 'reading 30'),

(14.77, 96.9000, 8.5000, 'm/s', '2024-03-03', '23:15:00', '25rt', 'reading 31'),

(13.11, 98.5000, 7.1000, 'kph', '2024-03-03', '23:16:00', '25rt', 'reading 32'),

(9.67, 96.5678, 8.5678, 'm/s', '2024-03-03', '23:17:00', '25rt', 'reading 33'),

(11.23, 97.1230, 7.6540, 'kph', '2024-03-03', '23:18:00', '25rt', 'reading 34'),

(13.00, 96.3210, 8.3210, 'm/s', '2024-03-03', '23:19:00', '25rt', 'reading 35'),

(12.12, 97.7654, 7.8765, 'm/s', '2024-03-03', '23:20:00', '25rt', 'reading 36'),

(10.10, 96.3456, 8.1234, 'kph', '2024-03-03', '23:21:00', '25rt', 'reading 37'),

(14.00, 97.9999, 7.1111, 'm/s', '2024-03-03', '23:22:00', '25rt', 'reading 38'),

(13.85, 96.2222, 8.2222, 'kph', '2024-03-03', '23:23:00', '25rt', 'reading 39'),

(11.90, 98.4444, 7.3333, 'm/s', '2024-03-03', '23:24:00', '25rt', 'reading 40'),

(9.44, 96.6666, 8.4444, 'm/s', '2024-03-03', '23:25:00', '25rt', 'reading 41'),

(13.30, 97.8888, 7.7777, 'kph', '2024-03-03', '23:26:00', '25rt', 'reading 42'),

(12.50, 98.3333, 7.5555, 'm/s', '2024-03-03', '23:27:00', '25rt', 'reading 43'),

(10.50, 97.1111, 7.8888, 'kph', '2024-03-03', '23:28:00', '25rt', 'reading 44'),

(13.70, 96.9876, 8.5432, 'm/s', '2024-03-03', '23:29:00', '25rt', 'reading 45'),

(11.77, 98.7654, 7.2345, 'm/s', '2024-03-03', '23:30:00', '25rt', 'reading 46'),

(14.21, 97.6543, 7.4567, 'kph', '2024-03-03', '23:31:00', '25rt', 'reading 47'),

(12.33, 96.5432, 8.1234, 'm/s', '2024-03-03', '23:32:00', '25rt', 'reading 48'),

(9.88, 98.2100, 7.6400, 'kph', '2024-03-03', '23:33:00', '25rt', 'reading 49'),

(10.67, 97.2100, 7.8400, 'm/s', '2024-03-03', '23:34:00', '25rt', 'reading 50'),

(14.89, 98.1010, 7.9990, 'm/s', '2024-03-03', '23:35:00', '25rt', 'reading 51'),

(13.10, 96.4567, 8.7654, 'kph', '2024-03-03', '23:36:00', '25rt', 'reading 52'),

(12.77, 97.6543, 7.3210, 'm/s', '2024-03-03', '23:37:00', '25rt', 'reading 53'),

(11.50, 96.1111, 8.0000, 'kph', '2024-03-03', '23:38:00', '25rt', 'reading 54'),

(9.70, 98.3333, 7.1234, 'm/s', '2024-03-03', '23:39:00', '25rt', 'reading 55'),

(14.12, 97.3210, 7.6543, 'm/s', '2024-03-03', '23:40:00', '25rt', 'reading 56'),

(12.66, 96.7890, 8.5432, 'kph', '2024-03-03', '23:41:00', '25rt', 'reading 57'),

(13.45, 97.0000, 7.0000, 'm/s', '2024-03-03', '23:42:00', '25rt', 'reading 58'),

(10.44, 96.2000, 8.3000, 'kph', '2024-03-03', '23:43:00', '25rt', 'reading 59'),

(11.99, 98.4000, 7.9000, 'm/s', '2024-03-03', '23:44:00', '25rt', 'reading 60')

]

```
# Try to insert the entries into the TEMP table in the database
try:
    cur.executemany(
        "INSERT INTO TEMP (VAL, LAT, LONGT, UNIT, DATE, TIME, SENS_ID, NOTE) "
        "VALUES (?, ?, ?, ?, ?, ?, ?, ?)",
        entries  # pass the list of entries to insert
    )
except mariadb.Error as e:
    # If there is a problem during insertion, print the error
    print(f"Error: {e}")
else:
    # If successful, commit the changes to the database
    conn.commit()
    print("Entries inserted successfully.")
    # Print the ID of the last inserted row (may refer to the last of the batch)
    print(f"Last Inserted ID: {cur.lastrowid}")

# Close the database connection when done
conn.close()
```

---

this is the insert-many-windspd-code:

```
# Import necessary libraries
import mariadb  # MariaDB connector for Python
import sys      # System library for exiting the script on error
```

```
# Try to connect to MariaDB database
try:
    conn = mariadb.connect(
        user="tudublin",        # Database username
        password="tudublinpwd",  # Database password
        host="127.0.0.1",       # Localhost (same computer)
        port=3306,              # Default MariaDB port
        database="WEATHER"      # Database name
    )
except mariadb.Error as e:
    # Print error and exit if connection fails
    print(f"Error connecting to MariaDB Platform: {e}")
    sys.exit(1)

# Create a cursor object to interact with the database
cur = conn.cursor()

# Define wind speed data entries
# Each tuple contains:
# (value, latitude, longitude, unit, date, time, sensor ID, reading description)
entries = [
    (12.12, 97.0087, 7.8907, 'm/s', '2024-03-03', '22:45:01', '25rt', 'reading 1'),
    (14.55, 96.0000, 8.9000, 'kph', '2024-03-03', '22:46:00', '25rt', 'reading 2'),
    (10.25, 98.1234, 7.6543, 'm/s', '2024-03-03', '22:47:00', '25rt', 'reading 3'),
```

(11.78, 96.4523, 8.4523, 'm/s', '2024-03-03', '22:48:00', '25rt', 'reading 4'),

(13.40, 97.7832, 7.9231, 'kph', '2024-03-03', '22:49:00', '25rt', 'reading 5'),

(9.85, 96.3021, 8.4120, 'm/s', '2024-03-03', '22:50:00', '25rt', 'reading 6'),

(15.20, 97.2341, 7.9341, 'kph', '2024-03-03', '22:51:00', '25rt', 'reading 7'),

(13.95, 98.1230, 7.2345, 'm/s', '2024-03-03', '22:52:00', '25rt', 'reading 8'),

(12.67, 96.8734, 8.1234, 'kph', '2024-03-03', '22:53:00', '25rt', 'reading 9'),

(11.90, 97.0001, 7.0001, 'm/s', '2024-03-03', '22:54:00', '25rt', 'reading 10'),

(10.44, 97.1234, 7.5432, 'kph', '2024-03-03', '22:55:00', '25rt', 'reading 11'),

(9.99, 96.6543, 8.0987, 'm/s', '2024-03-03', '22:56:00', '25rt', 'reading 12'),

(14.33, 97.0987, 7.3456, 'm/s', '2024-03-03', '22:57:00', '25rt', 'reading 13'),

(13.77, 98.0000, 7.1000, 'kph', '2024-03-03', '22:58:00', '25rt', 'reading 14'),

(10.88, 96.7890, 8.1111, 'm/s', '2024-03-03', '22:59:00', '25rt', 'reading 15'),

(11.33, 97.3210, 7.3210, 'kph', '2024-03-03', '23:00:00', '25rt', 'reading 16'),

(14.88, 96.1111, 8.2222, 'm/s', '2024-03-03', '23:01:00', '25rt', 'reading 17'),

(12.00, 97.8888, 7.9999, 'm/s', '2024-03-03', '23:02:00', '25rt', 'reading 18'),

(13.50, 98.4321, 7.1234, 'kph', '2024-03-03', '23:03:00', '25rt', 'reading 19'),

(10.10, 96.2000, 8.1000, 'm/s', '2024-03-03', '23:04:00', '25rt', 'reading 20'),

(11.80, 97.2100, 7.7100, 'm/s', '2024-03-03', '23:05:00', '25rt', 'reading 21'),

(13.10, 96.8900, 8.2100, 'kph', '2024-03-03', '23:06:00', '25rt', 'reading 22'),

(12.90, 97.6700, 7.6700, 'm/s', '2024-03-03', '23:07:00', '25rt', 'reading 23'),

(9.80, 98.2000, 7.8700, 'm/s', '2024-03-03', '23:08:00', '25rt', 'reading 24'),

(10.75, 96.7900, 8.0900, 'kph', '2024-03-03', '23:09:00', '25rt', 'reading 25'),

(14.12, 97.2500, 7.2500, 'm/s', '2024-03-03', '23:10:00', '25rt', 'reading 26'),

(13.32, 96.0000, 8.0000, 'kph', '2024-03-03', '23:11:00', '25rt', 'reading 27'),

(10.99, 97.4500, 7.5000, 'm/s', '2024-03-03', '23:12:00', '25rt', 'reading 28'),

(11.45, 98.1000, 7.3000, 'kph', '2024-03-03', '23:13:00', '25rt', 'reading 29'),
(12.88, 97.6000, 7.9000, 'm/s', '2024-03-03', '23:14:00', '25rt', 'reading 30'),
(14.77, 96.9000, 8.5000, 'm/s', '2024-03-03', '23:15:00', '25rt', 'reading 31'),
(13.11, 98.5000, 7.1000, 'kph', '2024-03-03', '23:16:00', '25rt', 'reading 32'),
(9.67, 96.5678, 8.5678, 'm/s', '2024-03-03', '23:17:00', '25rt', 'reading 33'),
(11.23, 97.1230, 7.6540, 'kph', '2024-03-03', '23:18:00', '25rt', 'reading 34'),
(13.00, 96.3210, 8.3210, 'm/s', '2024-03-03', '23:19:00', '25rt', 'reading 35'),
(12.12, 97.7654, 7.8765, 'm/s', '2024-03-03', '23:20:00', '25rt', 'reading 36'),
(10.10, 96.3456, 8.1234, 'kph', '2024-03-03', '23:21:00', '25rt', 'reading 37'),
(14.00, 97.9999, 7.1111, 'm/s', '2024-03-03', '23:22:00', '25rt', 'reading 38'),
(13.85, 96.2222, 8.2222, 'kph', '2024-03-03', '23:23:00', '25rt', 'reading 39'),
(11.90, 98.4444, 7.3333, 'm/s', '2024-03-03', '23:24:00', '25rt', 'reading 40'),
(9.44, 96.6666, 8.4444, 'm/s', '2024-03-03', '23:25:00', '25rt', 'reading 41'),
(13.30, 97.8888, 7.7777, 'kph', '2024-03-03', '23:26:00', '25rt', 'reading 42'),
(12.50, 98.3333, 7.5555, 'm/s', '2024-03-03', '23:27:00', '25rt', 'reading 43'),
(10.50, 97.1111, 7.8888, 'kph', '2024-03-03', '23:28:00', '25rt', 'reading 44'),
(13.70, 96.9876, 8.5432, 'm/s', '2024-03-03', '23:29:00', '25rt', 'reading 45'),
(11.77, 98.7654, 7.2345, 'm/s', '2024-03-03', '23:30:00', '25rt', 'reading 46'),
(14.21, 97.6543, 7.4567, 'kph', '2024-03-03', '23:31:00', '25rt', 'reading 47'),
(12.33, 96.5432, 8.1234, 'm/s', '2024-03-03', '23:32:00', '25rt', 'reading 48'),
(9.88, 98.2100, 7.6400, 'kph', '2024-03-03', '23:33:00', '25rt', 'reading 49'),
(10.67, 97.2100, 7.8400, 'm/s', '2024-03-03', '23:34:00', '25rt', 'reading 50'),
(14.89, 98.1010, 7.9990, 'm/s', '2024-03-03', '23:35:00', '25rt', 'reading 51'),
(13.10, 96.4567, 8.7654, 'kph', '2024-03-03', '23:36:00', '25rt', 'reading 52'),
(12.77, 97.6543, 7.3210, 'm/s', '2024-03-03', '23:37:00', '25rt', 'reading 53'),

```
    (11.50, 96.1111, 8.0000, 'kph', '2024-03-03', '23:38:00', '25rt', 'reading 54'),
    (9.70, 98.3333, 7.1234, 'm/s', '2024-03-03', '23:39:00', '25rt', 'reading 55'),
    (14.12, 97.3210, 7.6543, 'm/s', '2024-03-03', '23:40:00', '25rt', 'reading 56'),
    (12.66, 96.7890, 8.5432, 'kph', '2024-03-03', '23:41:00', '25rt', 'reading 57'),
    (13.45, 97.0000, 7.0000, 'm/s', '2024-03-03', '23:42:00', '25rt', 'reading 58'),
    (10.44, 96.2000, 8.3000, 'kph', '2024-03-03', '23:43:00', '25rt', 'reading 59'),
    (11.99, 98.4000, 7.9000, 'm/s', '2024-03-03', '23:44:00', '25rt', 'reading 60')
]

# Insert data into the WINDSP table
try:
    cur.executemany(
        "INSERT INTO WINDSP (VAL, LAT, LONGT, UNIT, DATE, TIME, SENS_ID,
DESCRIPTION) VALUES (?, ?, ?, ?, ?, ?, ?, ?)",
        entries
    )
    # Commit the transaction
    conn.commit()
    print("Data inserted successfully.")
except mariadb.Error as e:
    # Print error if insert fails
    print(f"Error inserting data: {e}")
```

heres the temp-loop-alternate-code for making entries:

```
import mariadb  # Library for connecting to MariaDB
import sys  # System-related functionality (e.g. exiting the script)
import random  # Library to generate random numbers
import string  # Library to generate random letters or characters
from datetime import datetime, timedelta  # For handling and manipulating dates and times


# Connect to MariaDB Platform
try:
    # Attempt to establish a connection to the MariaDB database using credentials
    conn = mariadb.connect(
        user="tudublin",        # Database username
        password="tudublinpwd",  # Password for the database user
        host="127.0.0.1",       # Localhost IP address (same machine)
        port=3306,              # Default MariaDB port
        database="WEATHER"      # Name of the database you're connecting to
    )
except mariadb.Error as e:
    # If connection fails, print the error message and exit the program
    print(f"Error connecting to MariaDB Platform: {e}")
    sys.exit(1)  # Exit the script with error code 1


# Create a cursor object which is used to execute SQL queries
cur = conn.cursor()
```

```
# Function to generate a random string of letters and digits
def random_string(length=5):
    # Return a random string of specified length (default is 5 characters)
    return ''.join(random.choices(string.ascii_uppercase + string.digits, k=length))

# Function that generates fake/random data entries to simulate sensor input
def generate_random_entries(num_entries):
    entries = []  # Create an empty list to store all generated entries
    base_time = datetime(2024, 3, 3, 22, 45, 0)  # Set a fixed starting date and time

    # Loop to generate the number of entries requested
    for _ in range(num_entries):
        val = round(random.uniform(5.0, 15.0), 2)  # Random value between 5.0 and 15.0 (rounded to 2 decimals)
        lat = round(random.uniform(90.0, 100.0), 4)  # Random latitude value
        longt = round(random.uniform(7.0, 15.0), 4)  # Random longitude value
        unit = random.choice(['m/s', 'kph', 'ft/s'])  # Randomly select a measurement unit
        time_offset = timedelta(minutes=random.randint(0, 60))  # Add random number of minutes (0-60)
        time = (base_time + time_offset).strftime('%H:%M:%S')  # Format time as a string
        sens_id = random_string(4)  # Generate a 4-character sensor ID
```

```
        note = random.choice(['entry one', 'entry two', 'entry three', 'random entry'])  # Pick a random note

        # Create a tuple with all the values and add it to the list
        entries.append((val, lat, longt, unit, base_time.strftime('%Y-%m-%d'), time, sens_id, note))

        # Increase the base time by 5 minutes for the next entry
        base_time += timedelta(minutes=5)

    # Return the full list of generated entries
    return entries

# Define how many fake entries to generate
num_entries = 80  # You can change this number if needed

# Generate the entries using the function above
entries = generate_random_entries(num_entries)

# Try to insert the generated entries into the TEMP table in the database
try:
    # Insert all the entries using a single batch command
    cur.executemany(
        "INSERT INTO TEMP (VAL, LAT, LONGT, UNIT, DATE, TIME, SENS_ID, NOTE) "
        "VALUES (?, ?, ?, ?, ?, ?, ?, ?)",  # SQL query with placeholders
```

```
        entries  # Data to be inserted
    )
except mariadb.Error as e:
    # If there's an error during insert, print the error
    print(f"Error: {e}")
else:
    # If insertion is successful, commit the transaction
    conn.commit()
    print(f"{num_entries} Entries inserted successfully.")  # Print number of successful entries
    print(f"Last Inserted ID: {cur.lastrowid}")  # Show the last ID inserted into the table

# Close the connection to the database
conn.close()
```

---

heres an alternate windsp-loop-code:

```
import mariadb  # Used to connect to MariaDB database
import sys  # Used to handle system-level operations like exiting the program
import random  # Used for generating random numbers
import string  # Used for generating random strings
from datetime import datetime, timedelta  # Used for working with dates and times

# Try to connect to the MariaDB database
try:
```

```
    conn = mariadb.connect(
        user="tudublin",        # Username for the database
        password="tudublinpwd",  # Password for the user
        host="127.0.0.1",       # Localhost (same computer)
        port=3306,              # Port number for MariaDB (default is 3306)
        database="WEATHER"      # Name of the database you want to use
    )
except mariadb.Error as e:
    # If connection fails, print the error and exit the program
    print(f"Error connecting to MariaDB Platform: {e}")
    sys.exit(1)

# Create a cursor so we can run SQL commands
cur = conn.cursor()

# Function to create a random string (for example, a sensor ID)
def random_string(length=5):
    return ''.join(random.choices(string.ascii_uppercase + string.digits, k=length))

# Function to generate a list of fake/random weather entries
def generate_random_entries(num_entries):
    entries = []  # List to store the generated entries
    base_time = datetime(2024, 3, 3, 22, 45, 0)  # Starting time for the first entry
```

```
    # Loop to create the desired number of entries
    for _ in range(num_entries):
        val = round(random.uniform(5.0, 15.0), 2)  # Random value for wind speed
        lat = round(random.uniform(90.0, 100.0), 4)  # Random latitude
        longt = round(random.uniform(7.0, 15.0), 4)  # Random longitude
        unit = random.choice(['M/S', 'KPH', 'FT/S'])  # Random unit for wind speed
        time_offset = timedelta(minutes=random.randint(0, 60))  # Random time offset up to 60 mins
        time = (base_time + time_offset).strftime('%H:%M:%S')  # Create time string with offset
        sens_id = random_string(4)  # Random sensor ID (4 characters)
        note = random.choice(['ENTRY ONE', 'ENTRY TWO', 'ENTRY THREE', 'RANDOM ENTRY'])  # Random comment/note
        temp_id = random.randint(1, 100)  # Random TEMP_ID (possibly linking to another table)

        # Add the data as a tuple to the list
        entries.append((val, lat, longt, unit, base_time.strftime('%Y-%m-%d'), time, sens_id, note, temp_id))

        # Move base time forward by 5 minutes for next entry
        base_time += timedelta(minutes=5)

    return entries  # Return the list of entries
```

```
# Set how many entries you want to insert
num_entries = 80

# Generate the entries
entries = generate_random_entries(num_entries)

# Try to insert all entries into the WINDSP table
try:
    cur.executemany(
        "INSERT INTO WINDSP (VAL, LAT, LONGT, UNIT, DATE, TIME, SENS_ID, NOTE, TEMP_ID) "
        "VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)",  # SQL command with placeholders
        entries  # The list of tuples to insert
    )
except mariadb.Error as e:
    # Print the error if insertion fails
    print(f"Error: {e}")
else:
    # Commit the transaction if everything goes well
    conn.commit()
    print(f"{num_entries} Entries inserted successfully.")  # Confirm how many entries were added
    print(f"Last Inserted ID: {cur.lastrowid}")  # Show the ID of the last entry

# Close the database connection
conn.close()
```

this is the listener code for the temp table:

```
#!/usr/bin/python

# Employees Listener program


# Import the required modules

import mariadb  # For connecting to the MariaDB database

import socket   # For creating a network socket (to receive data from Arduino)

import string   # Not used here, can be removed


# Connect to the MariaDB database

conn = mariadb.connect(

   user="tudublin",          # Database username

   password="tudublinpwd",     # Database password

   host="localhost",          # Database host (local machine)

   database="WEATHER"         # Name of the database

)


# Create a cursor object to interact with the database

cur = conn.cursor()


# Set up the socket connection

HOST = ''       # Listen on all available network interfaces

PORT = 20001     # Port number used for the server (must match Arduino)
```

```
# Create the socket (using TCP)
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Bind the socket to the address and port
s.bind((HOST, PORT))

# Start listening for incoming connections (only one allowed here)
s.listen(1)
print('listening')

# Accept a connection from a client (Arduino)
sockconn, addr = s.accept()

print('Connection from Arduino at IP Address:', addr)

# Infinite loop to keep receiving data
while 1:
   print('waiting')
   data = sockconn.recv(1024)  # Wait to receive up to 1024 bytes
   if not data:
      break  # If no data, stop the loop

   print(data)  # Print the raw received data

   # Loop through each line of the received data
```

```
    for line in data.splitlines():
        if len(line) > 0:
            l = line.decode('utf-8')  # Convert from bytes to string
            term = l.split(",")      # Split by comma: [temperature, time]


            print("The Whole line" + l)


            # Check if the data has exactly 2 parts: temperature and time
            if len(term) == 2:
                print("the Temperature is " + str(term[0]))
                print("the time is " + str(term[1]))
                try:
                    value = float(term[0])      # Convert temperature to float
                    time_str = term[1].strip()   # Clean the time string
                    # Insert the temperature and time into the TEMP table
                    cur.execute("INSERT INTO TEMP (VAL, TIME) VALUES (?, ?)",
(value, time_str))
                    conn.commit()  # Save the changes to the database
                    print(f"Last Inserted ID: {cur.lastrowid}")  # Optional: show ID of
last inserted row
                except mariadb.Error as e:
                    # If there's a database error, print it and close connections
                    print(f"Error: {e}")
                    sockconn.close()
                    conn.close()
                except ValueError as ve:
```

```
            # If conversion of temperature fails, print error
            print(f"Invalid value: {ve}")
        else:
            # If the line is not in expected format, print it and close connections
            print("something went wrong" + l)
            sockconn.close()
            conn.close()
```

---

this is the listener code for the windspd:

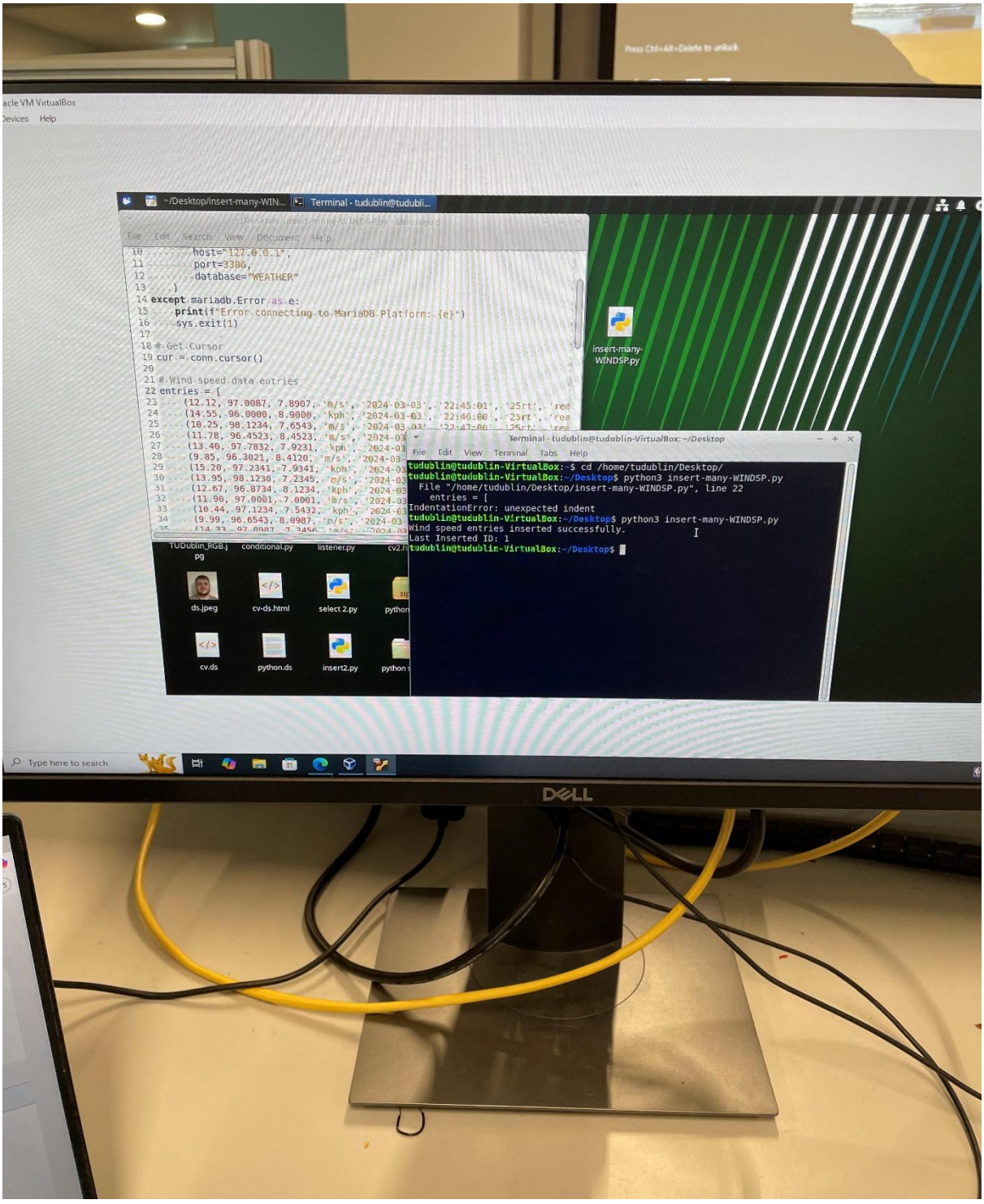
```
#!/usr/bin/python
# Employees Listener program

import mariadb        # Import the MariaDB library to interact with the database
import socket         # Import the socket library for network communication

# Connect to mariadb
conn = mariadb.connect(
   user="tudublin",        # Username to access the MariaDB database
   password="tudublinpwd",   # Password for the database user
   host="localhost",       # Database is hosted on the same machine
   database="WEATHER"      # Name of the database to connect to
)
cur = conn.cursor()         # Create a cursor to execute SQL queries
```

```
HOST = ''          # Empty string means the server will listen on all available interfaces
PORT = 20001       # Port number to listen on (must match the sender device, e.g. Arduino)

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  # Create a TCP socket
s.bind((HOST, PORT))        # Bind the socket to the host and port
s.listen(1)                 # Start listening for incoming connections (1 = max queue length)
print('listening')          # Let the user know it's ready

sockconn, addr = s.accept()   # Accept a connection from the client (e.g. Arduino)
print('Connection from Arduino at IP Address:', addr)  # Show client address

while 1:                    # Infinite loop to keep receiving data
   print('waiting')         # Waiting for data from the client
   data = sockconn.recv(1024)  # Receive up to 1024 bytes from the client
   if not data:             # If no data is received, break the loop
      break
   print(data)              # Print the raw data received (in bytes)

   for line in data.splitlines():  # Split data into individual lines (in case multiple are sent)
      if len(line) > 0:         # Only process non-empty lines
         l = line.decode('utf-8')  # Decode bytes to string
```

```
        term = l.split(",")      # Split the line into a list, using comma as
separator


        # Check if there are exactly 3 elements (expected format: VAL, UNIT,
TIME)
        if len(term) == 3:
            wind_speed = term[0]  # First value is wind speed
            unit = term[1]       # Second value is the unit (e.g., km/h, m/s)
            time = term[2]       # Third value is the time stamp
            # Print the values in a clear format
            print(f"{wind_speed},{unit},{time}")


            try:
                value = float(wind_speed)  # Try to convert wind speed to a float
                # Insert the data into the WINDSP table in the database
                cur.execute("INSERT INTO WINDSP (VAL, UNIT, TIME) VALUES (?,
?, ?)", (value, unit, time))
                conn.commit()  # Save the change to the database
                print(f"Last Inserted ID: {cur.lastrowid}")  # Show ID of the new
row
            except mariadb.Error as e:  # Handle database errors
                print(f"Error: {e}")
                sockconn.close()  # Close socket connection if error occurs
                conn.close()     # Close database connection if error occurs
            except ValueError as ve:  # Handle invalid number format
                print(f"Invalid value: {ve}")
```

```
        else:
            # If line does not have exactly 3 values, report it and close
connections
            print("Invalid data format:", l)
            sockconn.close()
            conn.close()

# Close the connection to the database after exiting the loop
conn.close()
```

The insert-many codes insert through a python script the entries from a set of entries also called entries, in which there are around 60 or more entries.



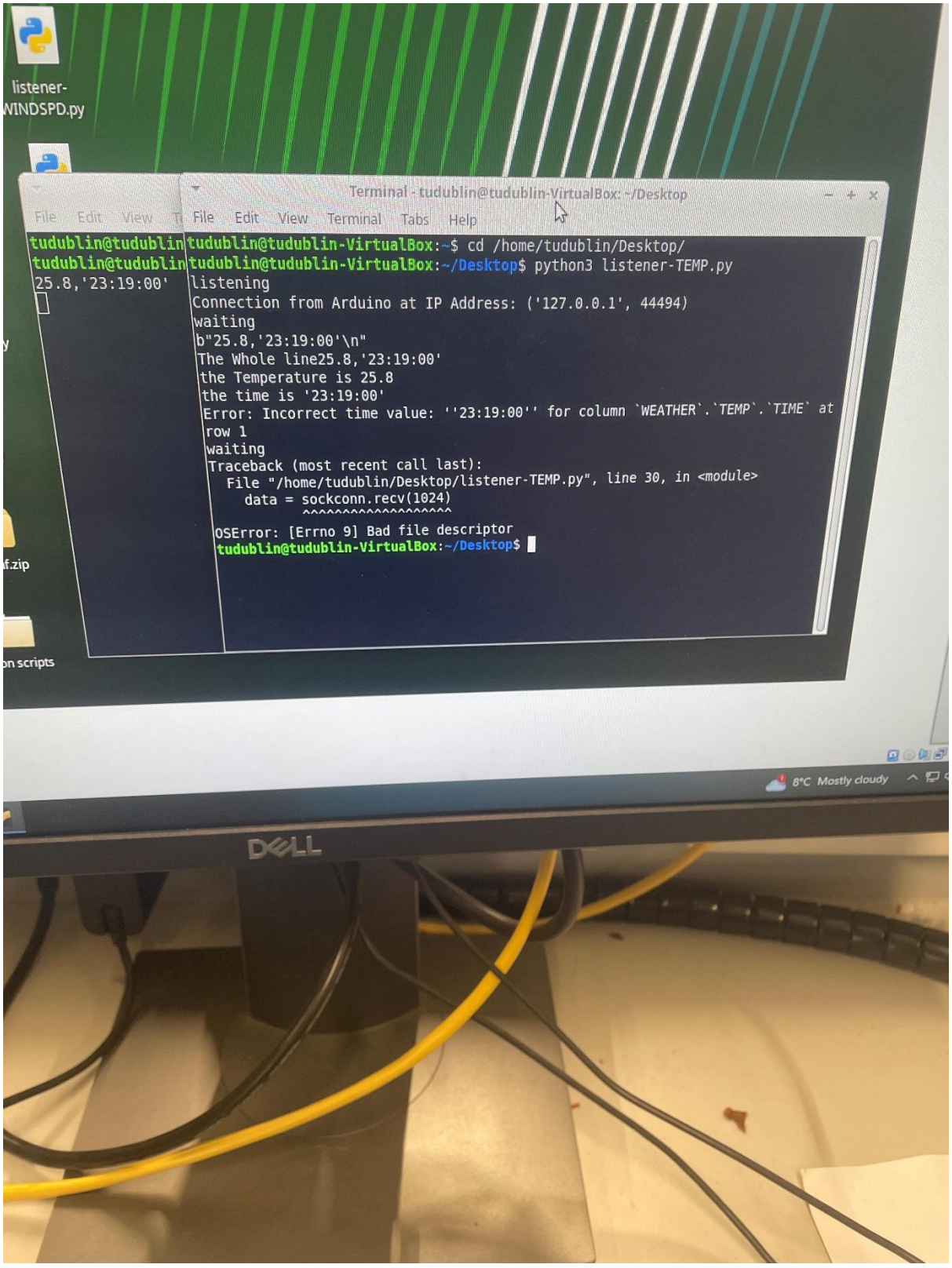
Python script inserting many at once:

The alternate loop scripts are an alternate way of generating random entries and you put the number you want from the script in the script itself then run it. It uses a for loop so is sequential.

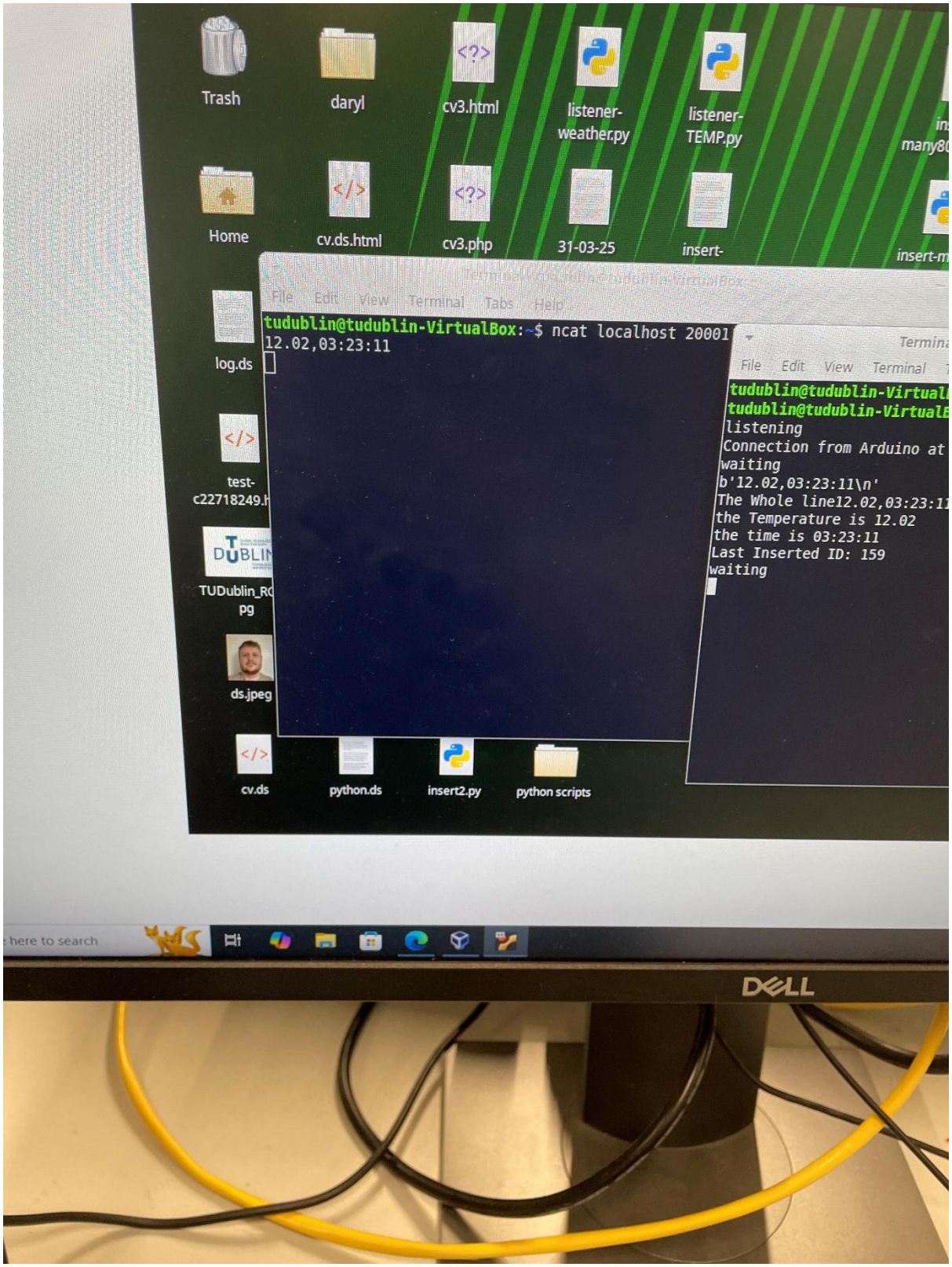
The listener scripts are for both temp and windspd and wait for an Arduino on the other side of the port. It can be also simulated using netcat. It is

proven to work using the listener which you first set up then make contact with the listener terminal with another terminal acting as another remote computer.

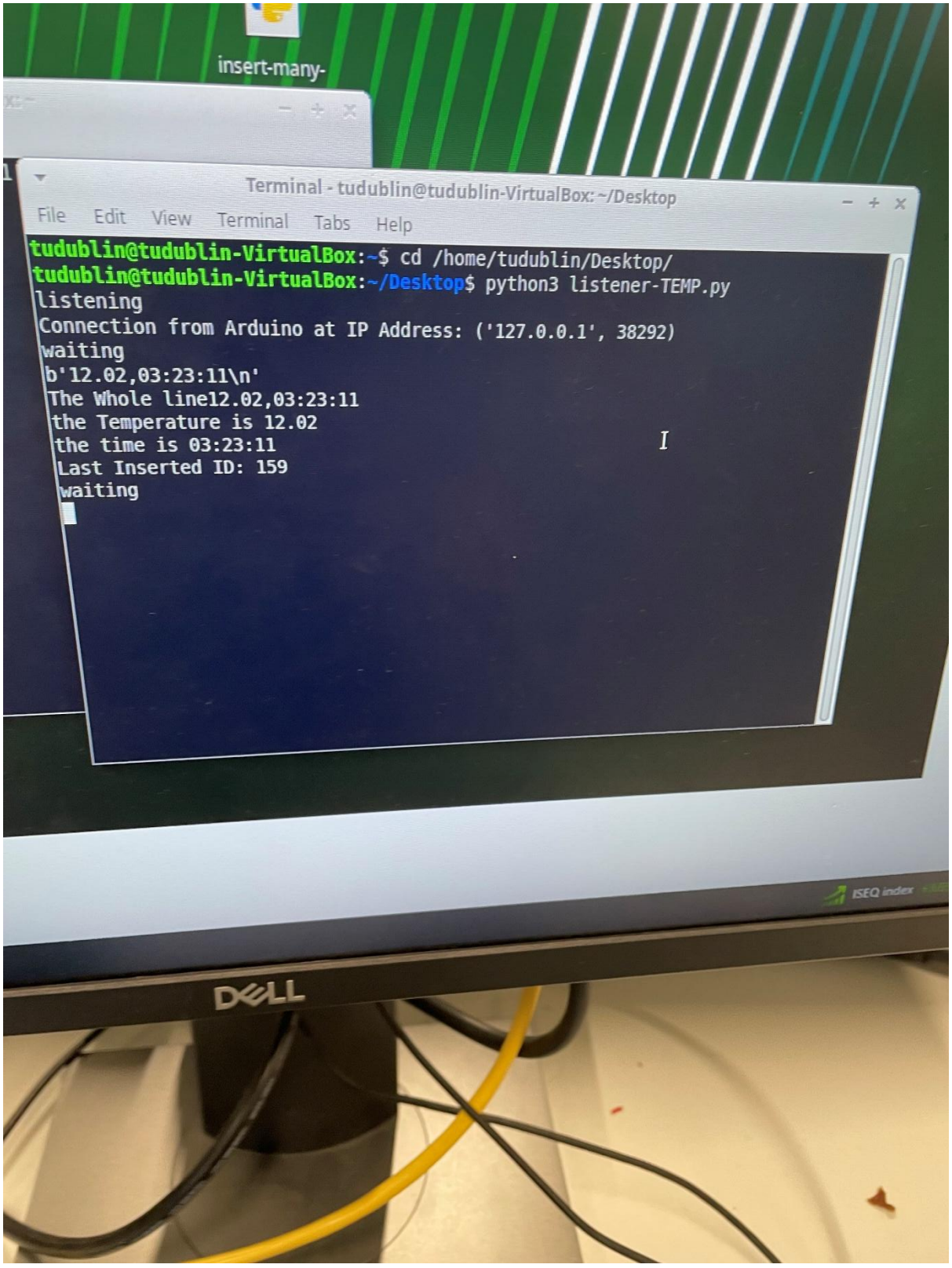
Typing “ncat localhost 20001” starts netcat then the entries are requested through the other terminal(guest) going to host. This checking if its there by typing the set variables say VAR and TIME say, (5.0,23:32:22).



The variables coming from remote computer then processed by host:



Proof of processing between computers:



Another pic showing listener working:

The scaling command:

The scaling in the instrumentation project was done by these terms:

The temperature scaling:

```
T = (val - c) / m; // temperature


int analogPin = A3;  // potentiometer wiper (middle terminal) connected to analog pin 3
                     // outside leads to ground and +5V
int val = 0;        // variable to store the value read
float m = 13.146;    //slope(m)
float c = 449.8;     //line intersection
float T = 0.0;       //Temperature set
//Rescale y = mx + c
 //A/D = mT + c //A/D is analogue to digital input
 //T = (A/D - c)/m
 T = (val - c) / m;
```

Using tests we found appropriate m(slope) and c(intersection of the line) values.

For the windspeed it was a kind of similar rescale only squaring the revs per second value:

```
Rps = revs per min;
kmph= (1.6406*(rps*rps)-10.06);
```

the python TEMP rescale:

```
#!/usr/bin/python
# Employees Listener program

import mariadb
import socket
import string

# Connect to mariadb
conn = mariadb.connect(
   user="tudublin",
   password="tudublinpwd",
   host="localhost",
   database="WEATHER"
)
cur = conn.cursor()

HOST = ''  # all available interfaces
PORT = 20001  # unprivileged ports are >20000

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen(1)
print('listening')
sockconn, addr = s.accept()
```

```
print('Connection from Arduino at IP Address:', addr)


while 1:
   print('waiting')
   data = sockconn.recv(1024)
   if not data:
      break
   print(data)
   for line in data.splitlines():
      if len(line) > 0:
         l = line.decode('utf-8')
         term = l.split(",")


         print("The Whole line" + l)


         if len(term) == 2:
            print("the Temperature is " + str(value))
            print("the time is " + str(term[1]))
            try:
               value = (term[0] - 449.8)/13.146)  # inserting temperature rescale
here
               time_str = term[1].strip()  # keep time as string
               cur.execute("INSERT INTO TEMP (VAL, TIME) VALUES (?, ?)",
(value, time_str))
               conn.commit()
```

```
            print(f"Last Inserted ID: {cur.lastrowid}")
        except mariadb.Error as e:
            print(f"Error: {e}")
            sockconn.close()
            conn.close()
        except ValueError as ve:
            print(f"Invalid value: {ve}")
    else:
        print("something went wrong" + l)
        sockconn.close()
        conn.close()
```

The WINDSPD python rescale:

```
#!/usr/bin/python
# Employees Listener program

import mariadb
import socket

# Connect to mariadb
conn = mariadb.connect(
   user="tudublin",
   password="tudublinpwd",
   host="localhost",
   database="WEATHER"
)
```

```
cur = conn.cursor()

HOST = ''  # all available interfaces
PORT = 20001  # unprivileged ports are >20000

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen(1)
print('listening')
sockconn, addr = s.accept()

print('Connection from Arduino at IP Address:', addr)

while 1:
   print('waiting')
   data = sockconn.recv(1024)
   if not data:
      break
   print(data)
   for line in data.splitlines():
      if len(line) > 0:
         l = line.decode('utf-8')
         term = l.split(",")

         # Check if there are exactly 3 elements
```

```
        if len(term) == 3:
            wind_speed = (1.6406*(term[0]*term[0])-10.06)  # Wind Speed rescale here
            unit = term[1]       # Unit
            time = term[2]       # Time
            # Print in the desired format: VAL, UNIT, TIME
            print(f"{wind_speed},{unit},{time}")

            try:
                value = float(wind_speed)  # Convert to float for VAL
                cur.execute("INSERT INTO WINDSP (VAL, UNIT, TIME) VALUES (?, ?, ?)", (value, unit, time))
                conn.commit()
                print(f"Last Inserted ID: {cur.lastrowid}")
            except mariadb.Error as e:
                print(f"Error: {e}")
                sockconn.close()
                conn.close()
            except ValueError as ve:
                print(f"Invalid value: {ve}")
        else:
            print("Invalid data format:", l)
            sockconn.close()
            conn.close()

# Close the connection to the database
```

conn.close()

---

login:

my login was on desktop:

.\admin username

Tudublin202$00 password

Also on virtual machine:

tudublin username

tudublinpwd password

Table-links:

TEMP -
http://127.0.0.1:3000/d/aehis4qgl4pogd/temperature?orgId=1&from=2024-03-03T00:00:00.000Z&to=2024-03-05T00:00:00.000Z&timezone=browser&tab=queries

WINDSPD -
http://127.0.0.1:3000/d/fehitmfpf26f4c/windspd?orgId=1&from=2024-03-03T00:00:00.000Z&to=2024-03-03T00:00:00.000Z&timezone=browser

assignment checklist:

- created WEATHER database containing TEMP and WINDSPD tables
- the tables contained the required headings and more
- I showed the python files used to insert values into my tables
- I additionally provided alternative loop insertion codes
- The netcat was proven to the required headings and more
- I set up two Grafana webpages with numerous statistic tables connected to the database variables in the tables
- I connected the pages to the cv assignment
- The cv assignment is connected to each separately
- The cv had two images, one selfie and one TUD logo connected to the site
- It had a table and an unordered list
- It had a hyperlink to my linkedin page
- The TUD logo hyperlink goes to the TUD website
- The cv and weatherstation or WEATHER pages are linked to eachother
- The document was created with the appropriate information

Appendix: Laboratory logs

Appendix Lab 1 and 2:

Laboratory log: first class

- We were shown in theory the inside of a PC
- Damon opened the PC as an example to show us how it works and what we're dealing with
- We were told that we were not allowed or qualified to go near the power unit in the PC so to leave it
- The motherboard was shown
- In theory he explained the differences and historical differences between AC and DC
- DC was less harmful on the inside of a PC and that is what powered the motherboard and its parts.
- Explain black as ground, orange as 3.3V, red as 5V, yellow as 12V and so forth
- We were shown the slots for the RAM and shown how it slots in
- We were shown the CPU and shown how to take it out
- We were shown most of the slots and which slots in where
- We were shown to delicately handle a CPU when taking it out and putting it back in and securing with the hinge
- All the ports were shown
- The memory, the RAM as well as the heat sinks on some of the chips and the fan

Laboratory Log: 17/02/2025

- Using the laboratory PC, we logged in as a administrator username as we were granted access for the class
- It was in laboratory room 341
- The username and password were (.\admin) = username and (Tudublin202$00) = password
- We logged in as administrator
- We downloaded "Xubuntu "by searching "Xubuntu download"
- We then selected the United Kingdom version (English)
- We then selected from a list of downloads the 64bit-AMD-24.01.04.iso version
- The iso version was 3.8GB and was off a DVD
- This was the longtime service version
- We then selected oracle virtual desktop icon on the desktop once it was downloaded
- Damon explained what to do up to registering your virtual machine

- The menu screen came up now and we pressed add
- I put my name in as daryl_sweeney and this was my username
- I set the iso image to the location that the Xubuntu file I downloaded from the site was
- I ticked the box
- I clicked next
- The next page showed how much base memory you could set to the virtual machine and how much was already being used by the PC
- I set the amount of base memory to 3500MB
- The next setting had how much processing power in number of cores that were being used by the PC and the free ones, so I assigned the virtual machine 4 units
- Clicking through I clicked one box then came to the data option and set it to 35GB
- After I went through the process, I finished it and after a while it set up
- I then opened it and after a while the virtual desktop ran
- When it loaded, I double clicked on the Xubuntu download, which was on the virtual desktop as well
- I went through the option process selecting the English version then installed it, using “tudublin” as my username and “tudublinpwd” as my password
- After a while it finished, and I selected restart desktop which was the virtual desktop
- It went blank then a black screen appeared, then I closed it down
- The personal account for the virtual desktop was set up as well as Xubuntu was installed on the virtual desktop
- It was explained how the virtual desktop worked by taking part of the base memory, data and processing power and assigning a fraction of the total power of the PC to an independent Linux virtual desktop
- It was like having two computers running on the one PC, one each independently running only knowing the computer assets that has been assigned to that desktop
- This could be run remotely through and independent drive external from the actual PC
- Theoretically a virtual desktop or a number of them together could be run together, selecting a desktop as you choose, and it was done like this in some workplaces.
- This external drive could be a drive or even a USB key with sufficient memory like 1 terabyte say – so you could just plug in and play
- Further detailed methods of this remote virtual desktop were not described but were possible, we will stick with this method for now

Appendix Lab 3:

Laboratory Log: 24/02/2025

- After the last lab, Xubuntu was installed, and a virtual machine was set up in my name. This was restarted at the start of the lab.
- Damon explained briefly the difference of a database for a single person and a relational database like a hospital say.
- The importance of statistics and tables in a relational database. In a job would be of utmost importance and very high responsibility.
- He explained a couple of things about Oracle, how it originated, bought java and eventually explained that we will be using "MariaDB" a relational database management system (RDBMS) that was open source and free. Therefore we don't have to buy it.
- When everything was set up the Linux kernel was started.
- He gave an explanation to for a beginner Linux operation
- In order to get in to MariaDB we typed this command

```
user@computor:~$sudo mariadb -u root -p
```

Typing this command after ~$ to initiate MariaDB:

- We then got into MariaDB.

```
mariadb> _
```

This command put the user inside the mariadb program ready for command:

- The first thing to learn was making a database and we made a weather database, using capital letters always for command and data in lower case.
- Then it was thought to USE the database weather:

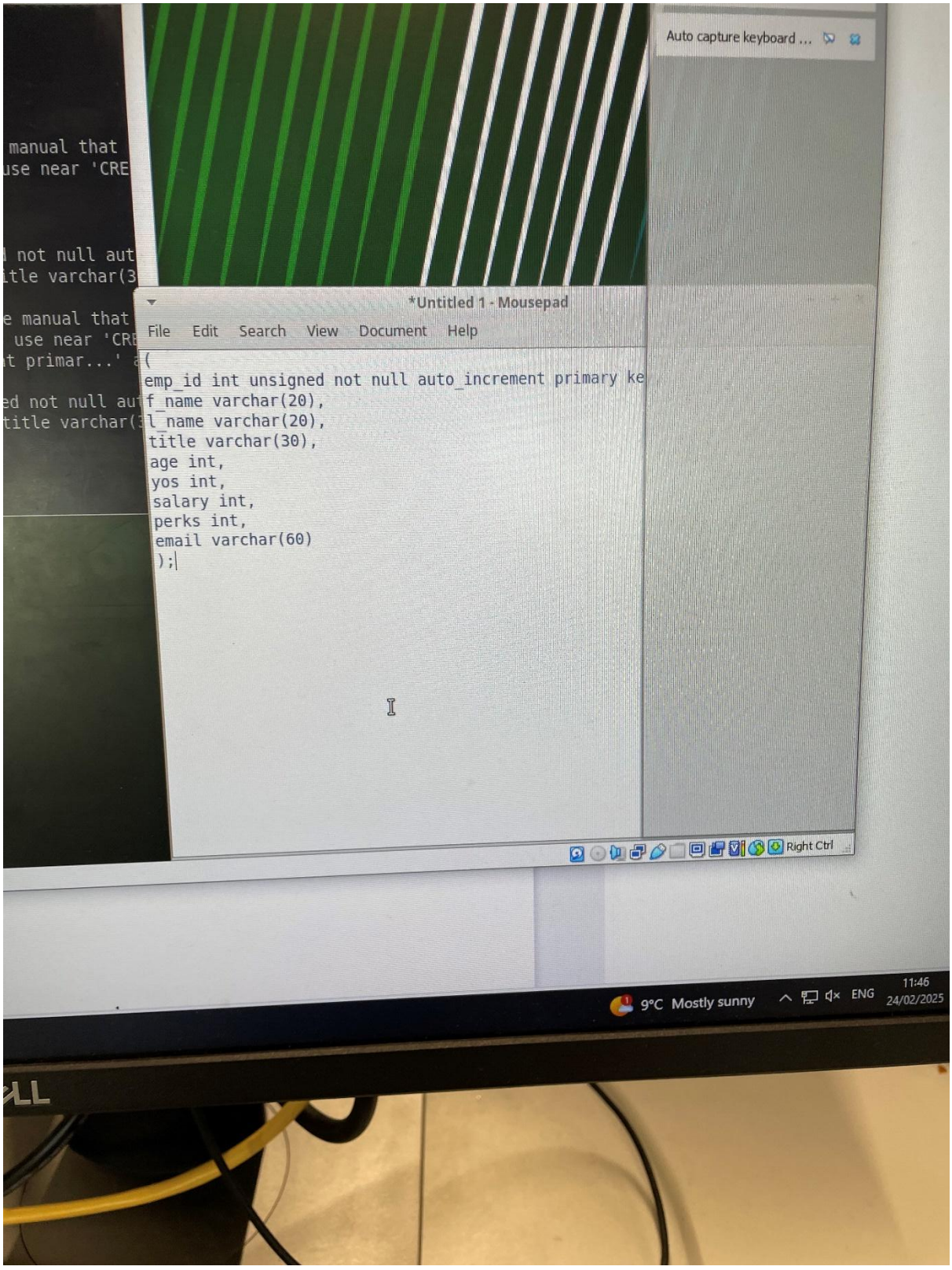
```
mariadb > USE weather;
```

Notice upper case – command and lower-case data:

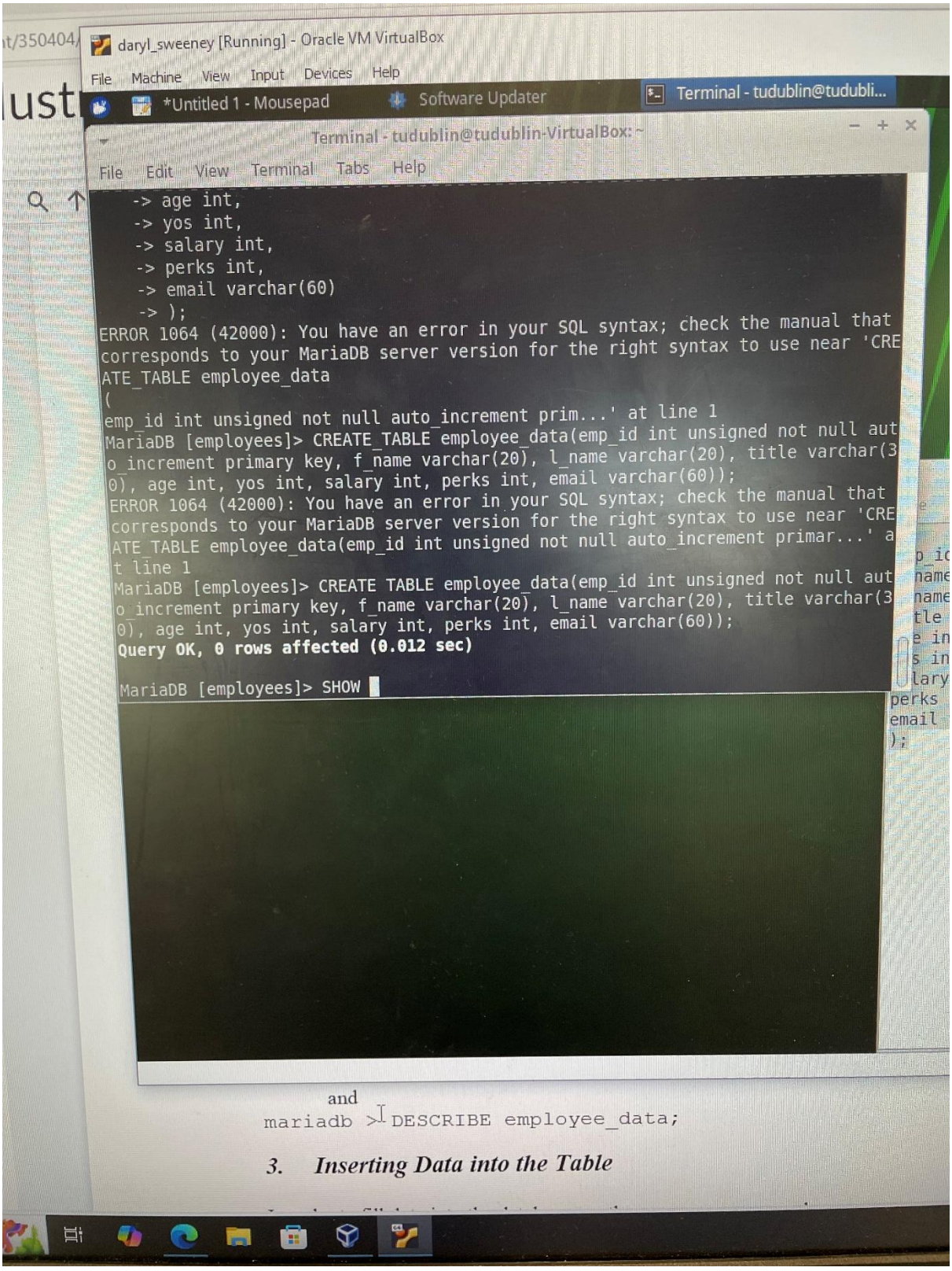
- That was just practice then:
- We now created a database called employees with a table called employee data with a primary key "emp_id"

```
mariadb> CREATE DATABASE employees; (and hit enter)

mariadb> USE employees; (and hit enter)

mariadb> CREATE TABLE employee_data
    -> (
    -> emp_id int unsigned not null auto_increment primary key,
    -> f_name  varchar(20),
    -> l_name  varchar(20),
    -> title varchar(30),
    -> age  int,
    -> yos  int,
    -> salary int,
    -> perks int,
    -> email varchar(60)
    ->);
```

This stated the headings of the columns and their variables:



Heres me using the equivalent of linux notepad to adjust then paste in the command that way you don't lose what you've wrote:



Pic 1: after entering the table and typing in show tables:

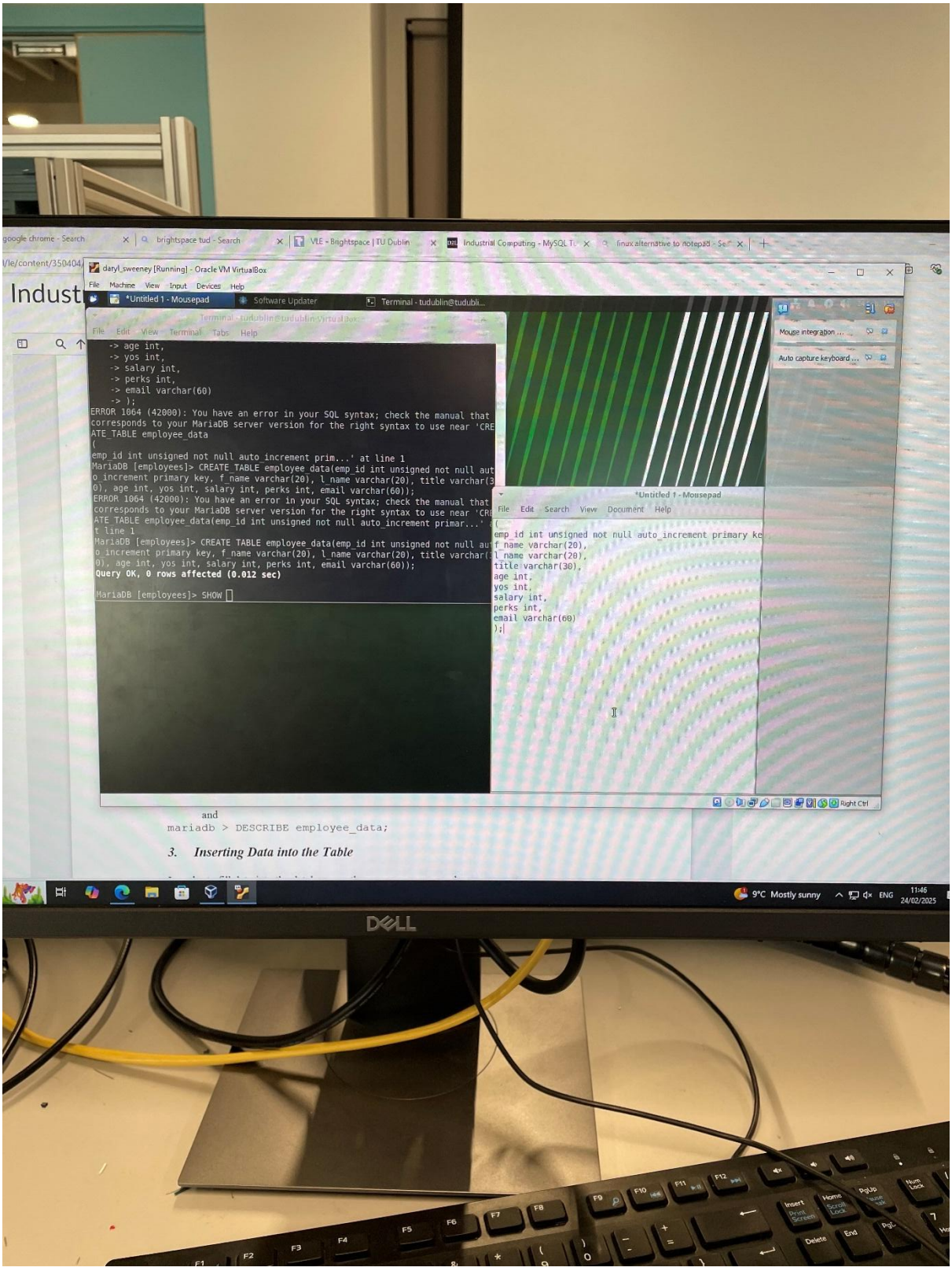
- You can now check the table:

```
mariadb > SHOW tables;
```

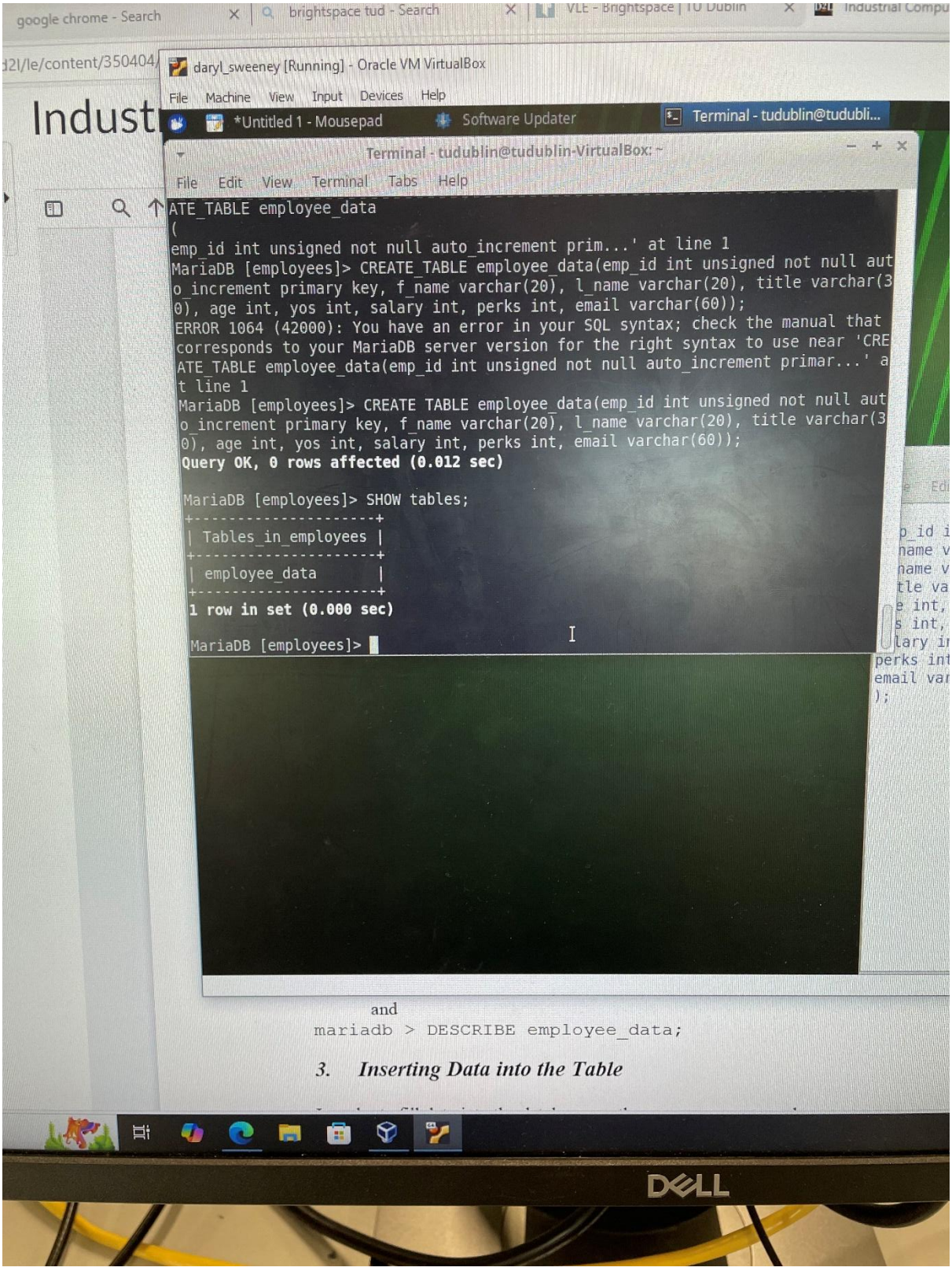
and

```
mariadb > DESCRIBE employee_data;
```

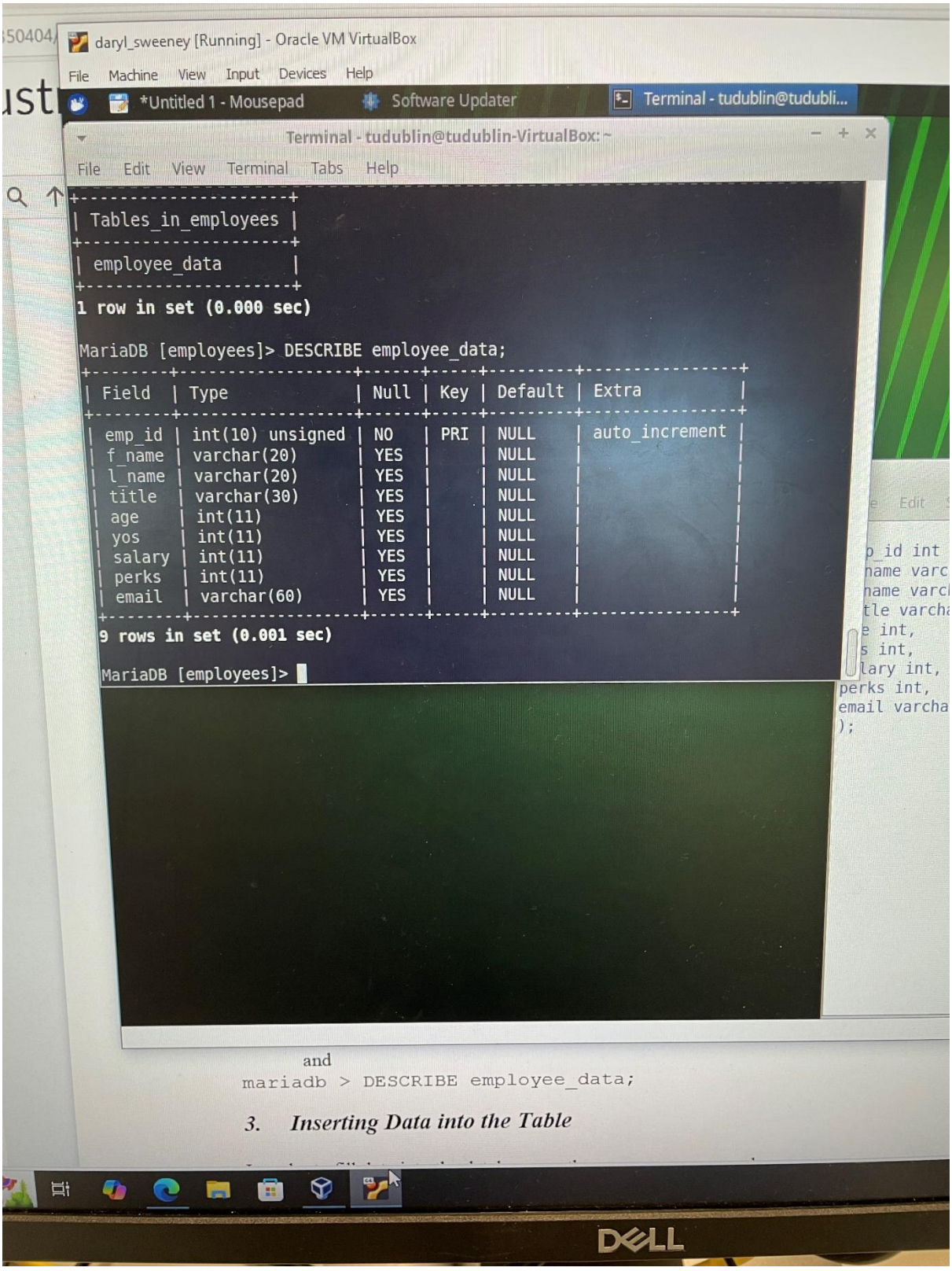
Using these commands:



Another pic of me sorting the commands successfully using mousepad the linux equivalent:



Showing how the tables are listed in the SHOW tables command:

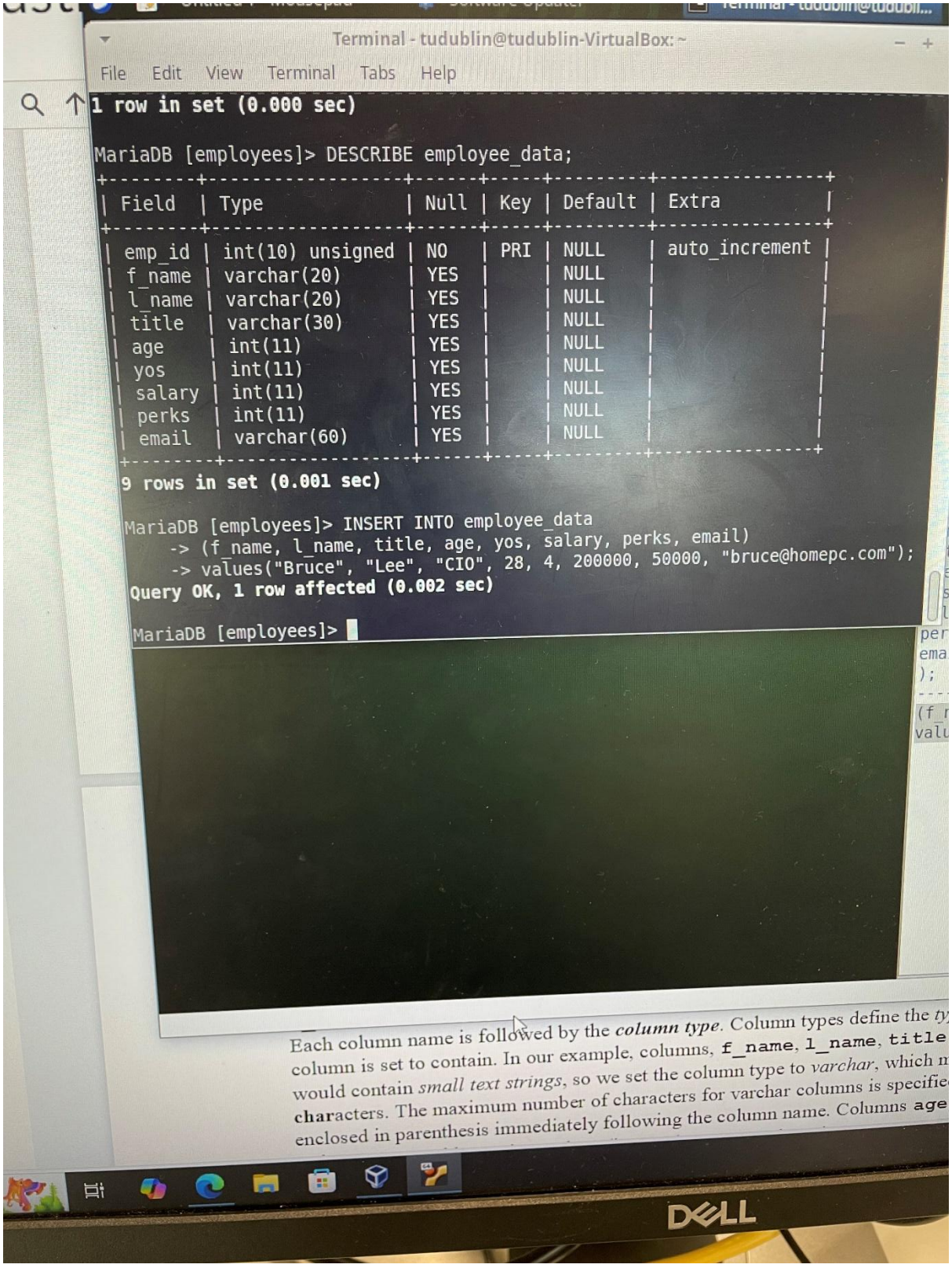


This is the DESCRIBE employee_data command result showing the whole table and still not filled but the properties:

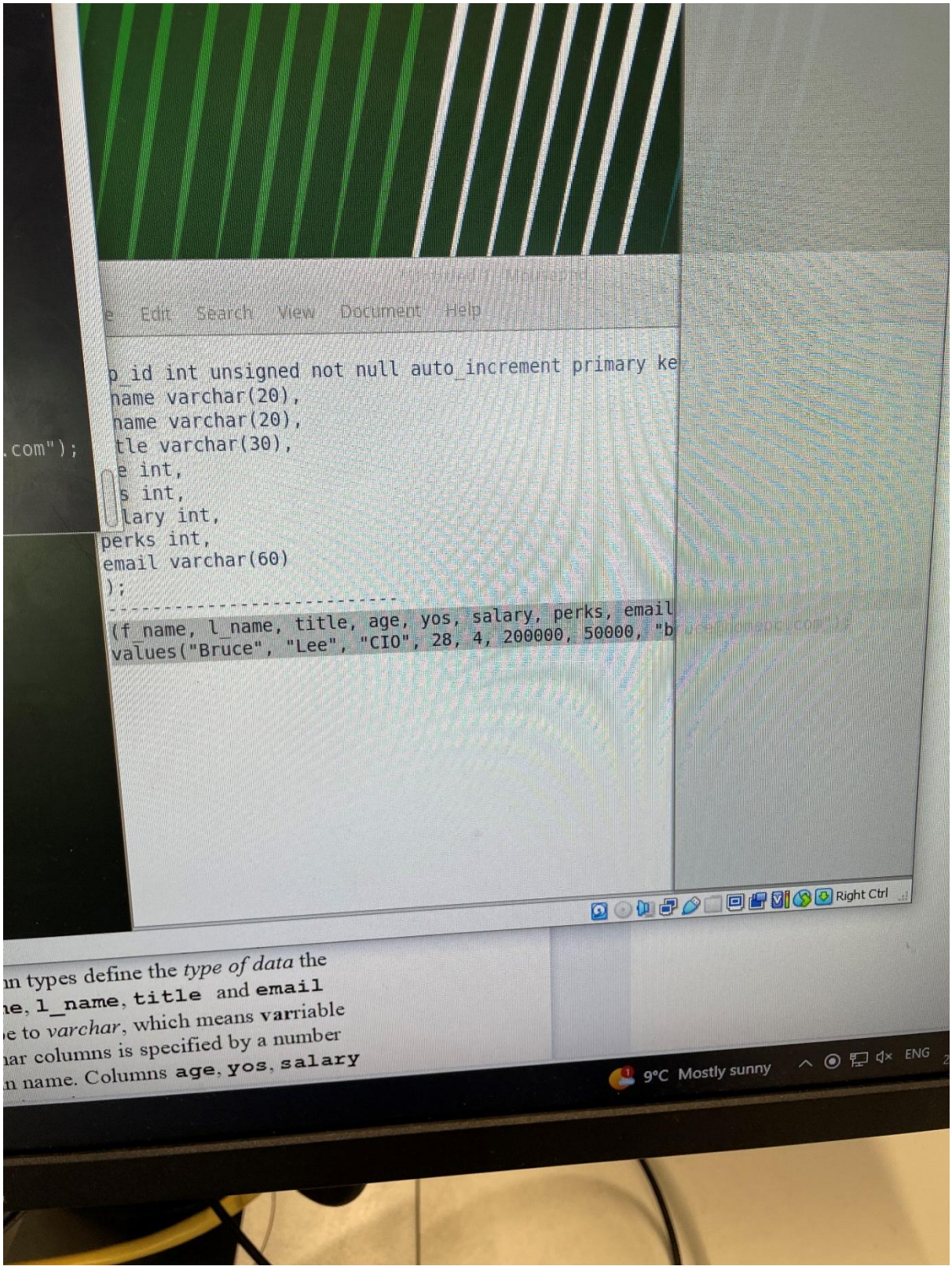
- In order to insert information, we used the INSERT command.

```
mariadb> INSERT INTO employee_data
    -> (f_name, l_name, title, age, yos, salary, perks,  email)
    -> values("Bruce", "Lee", "CIO", 28, 4, 200000,  50000,
"bruce@homepc.com");
```

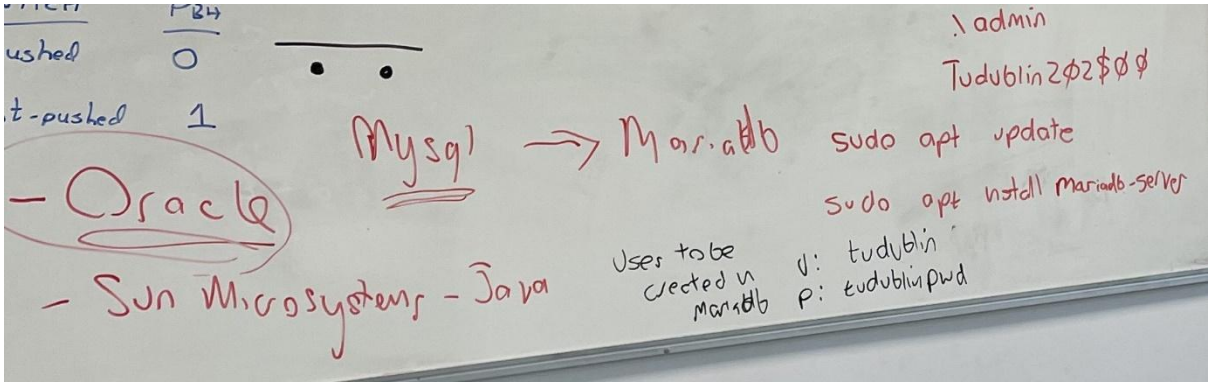
Firstly when the insert command was initiated, arrows came up prompting what to insert or command. The following headings were used then the next line clarifies the values for them headings:



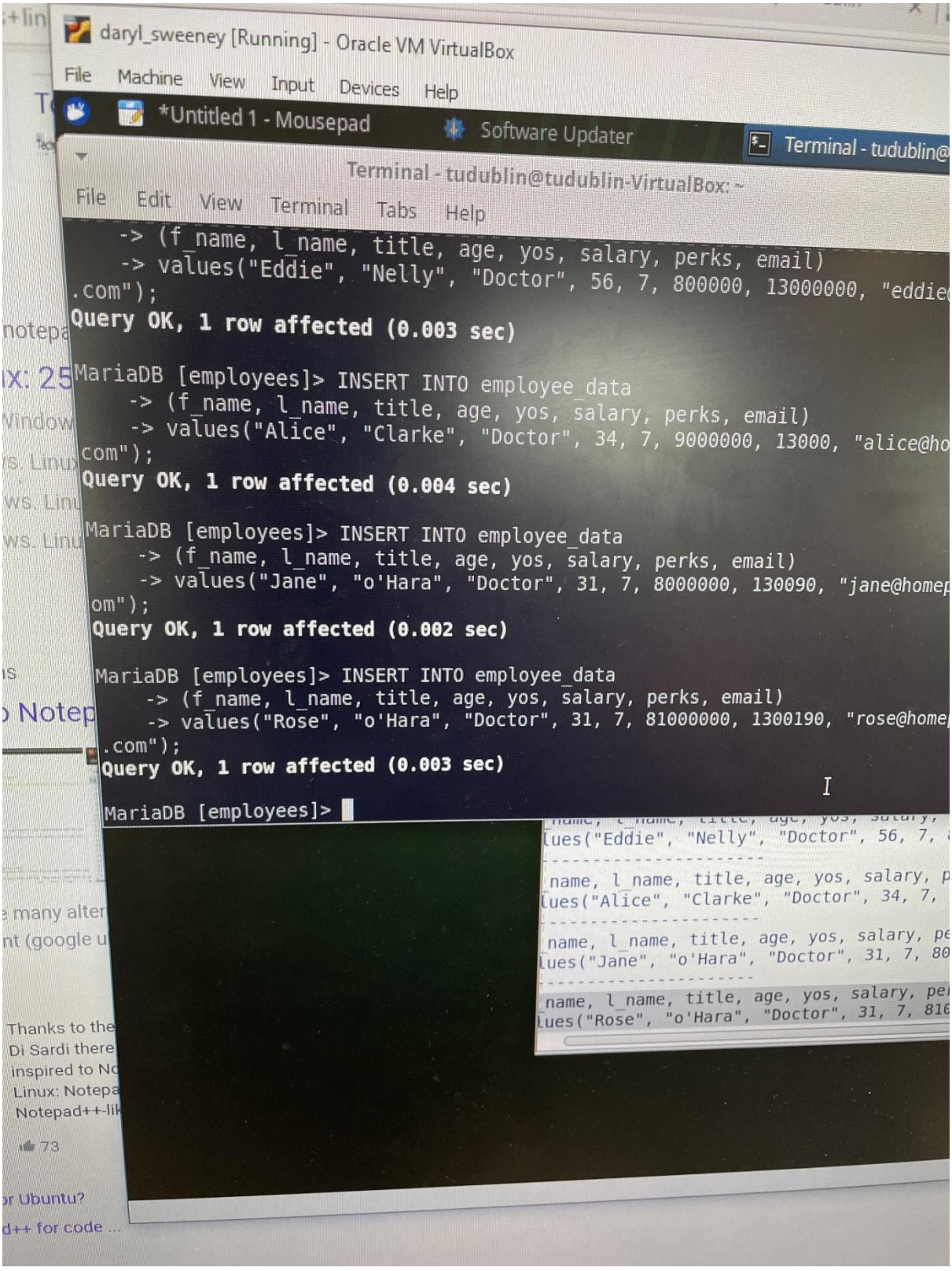
This is showing the first entry into the table with the headings:



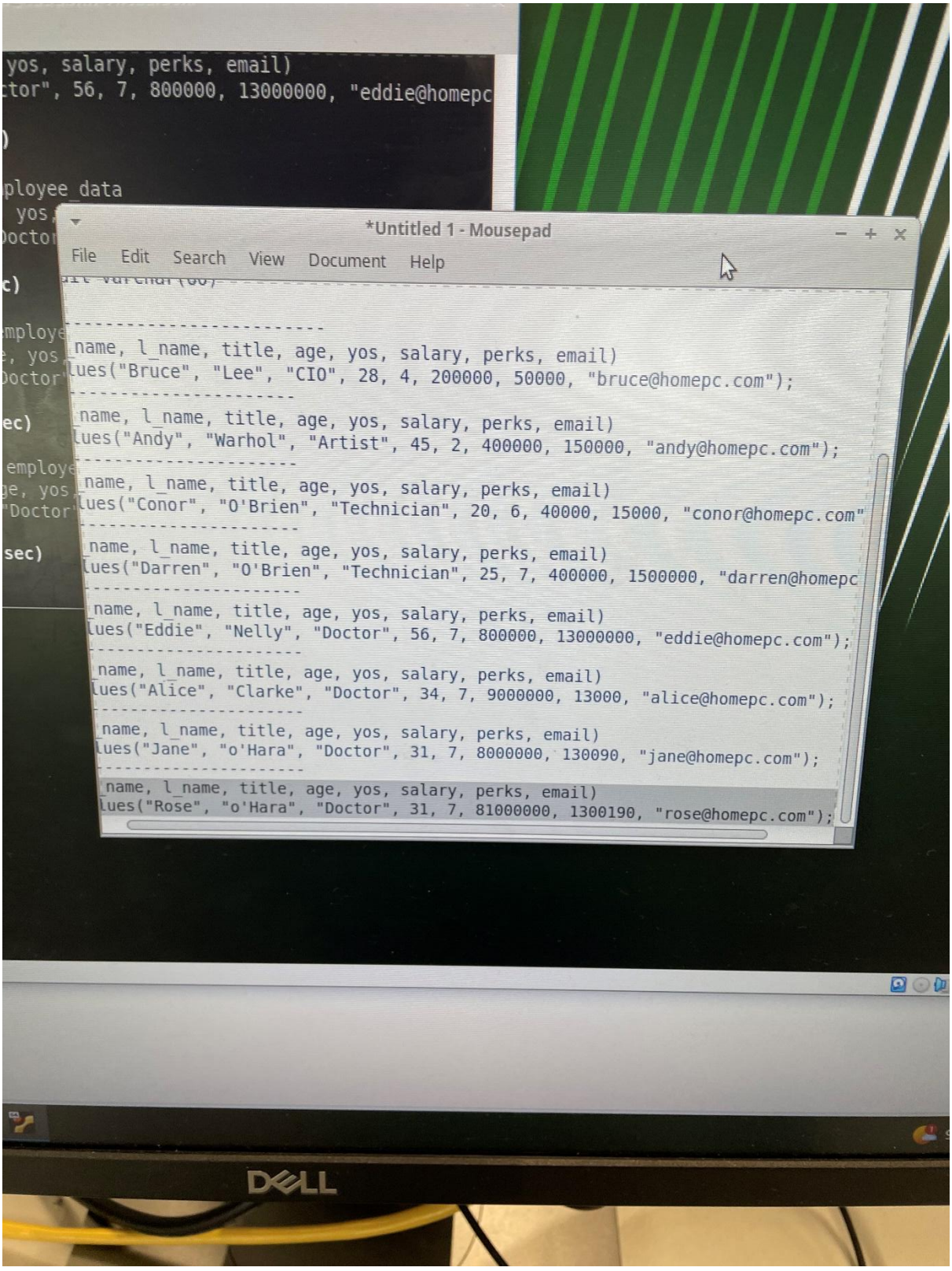
This showing me use mousepad to paste in and customise the same command differently:



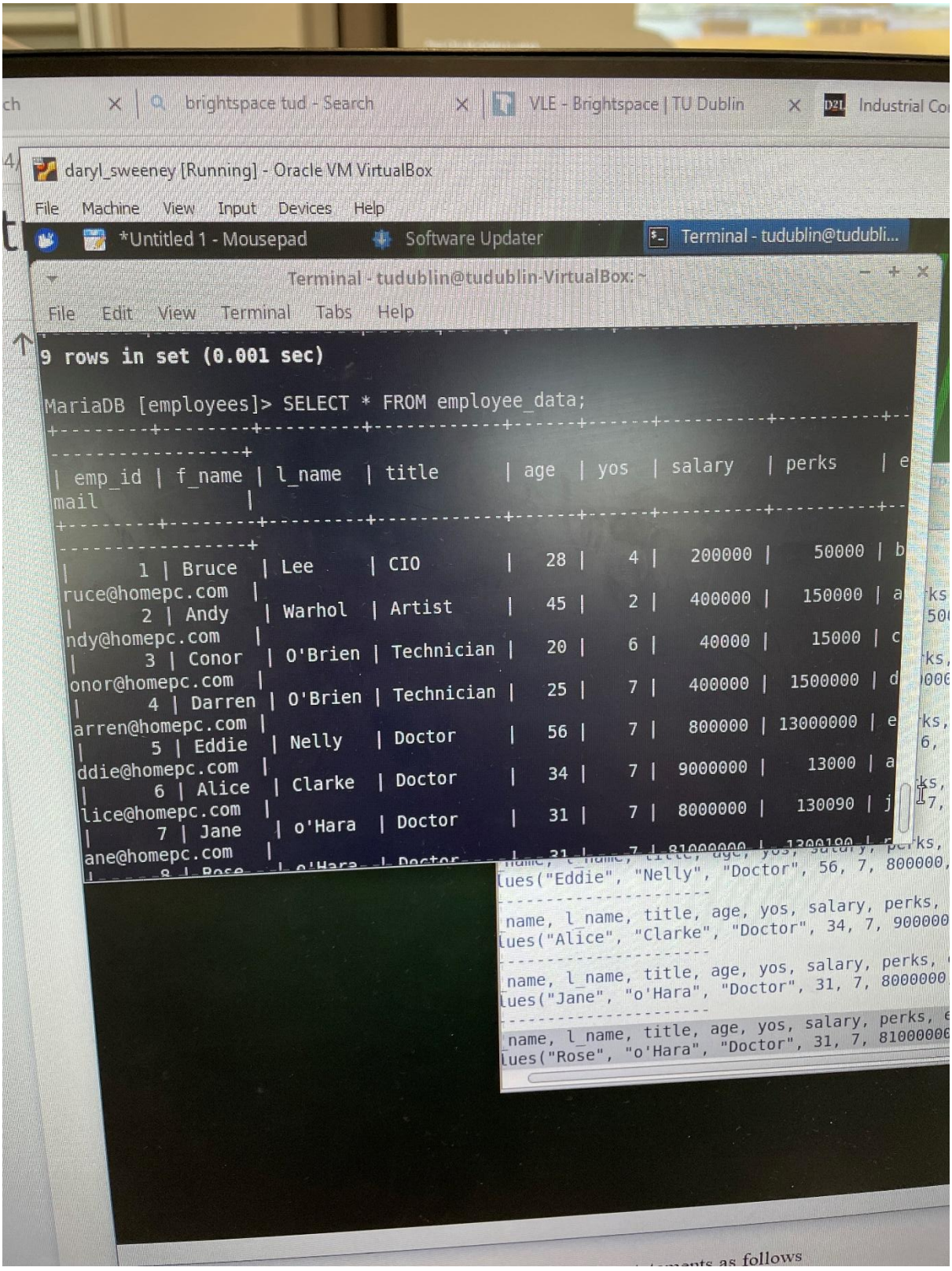
This explaining the significance of oracle, sun microsystems, mysql and mariadb. With the passwords.



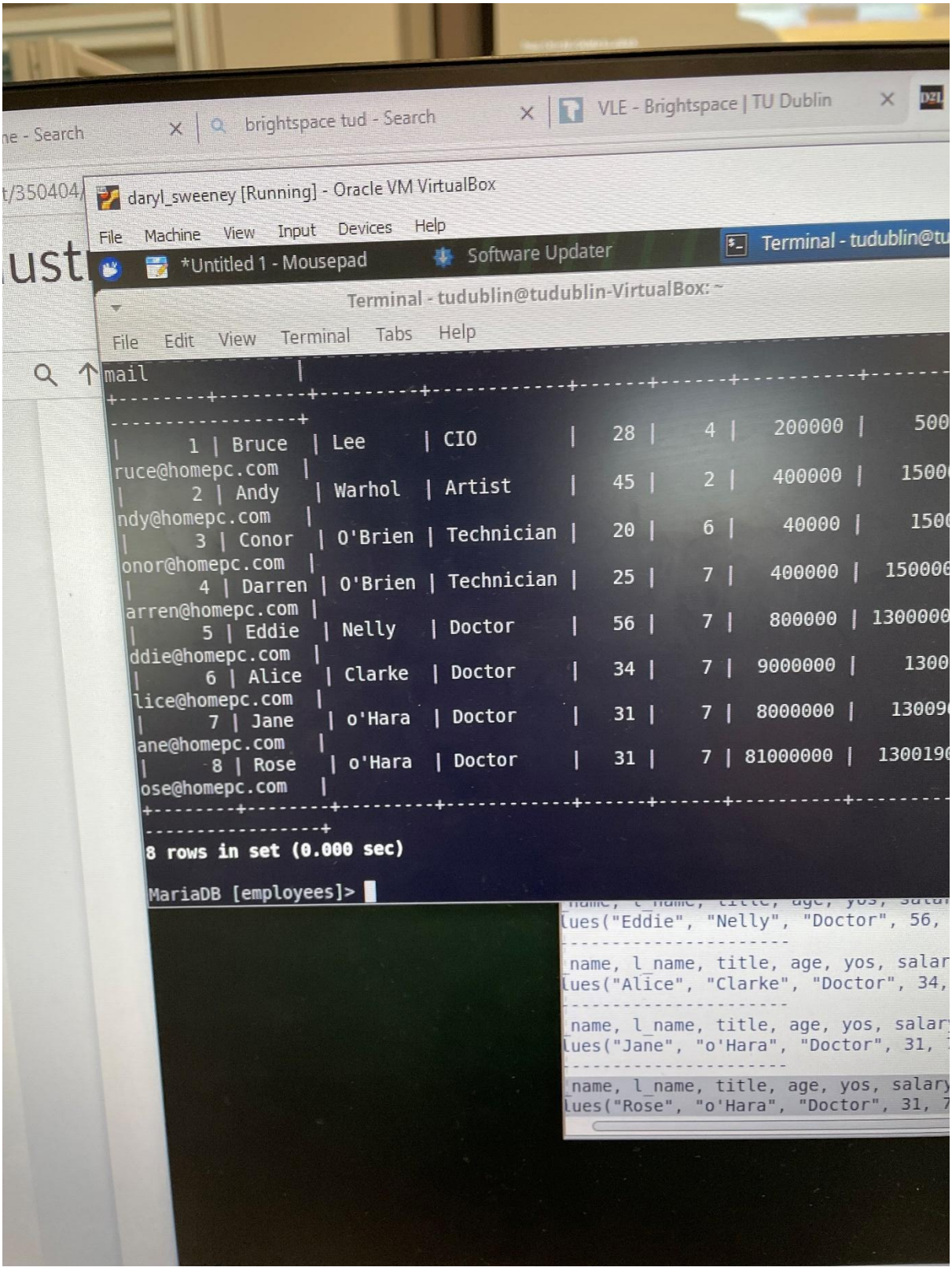
This is me adding different characters to the tables with different customisation copies of the bruce lee command:



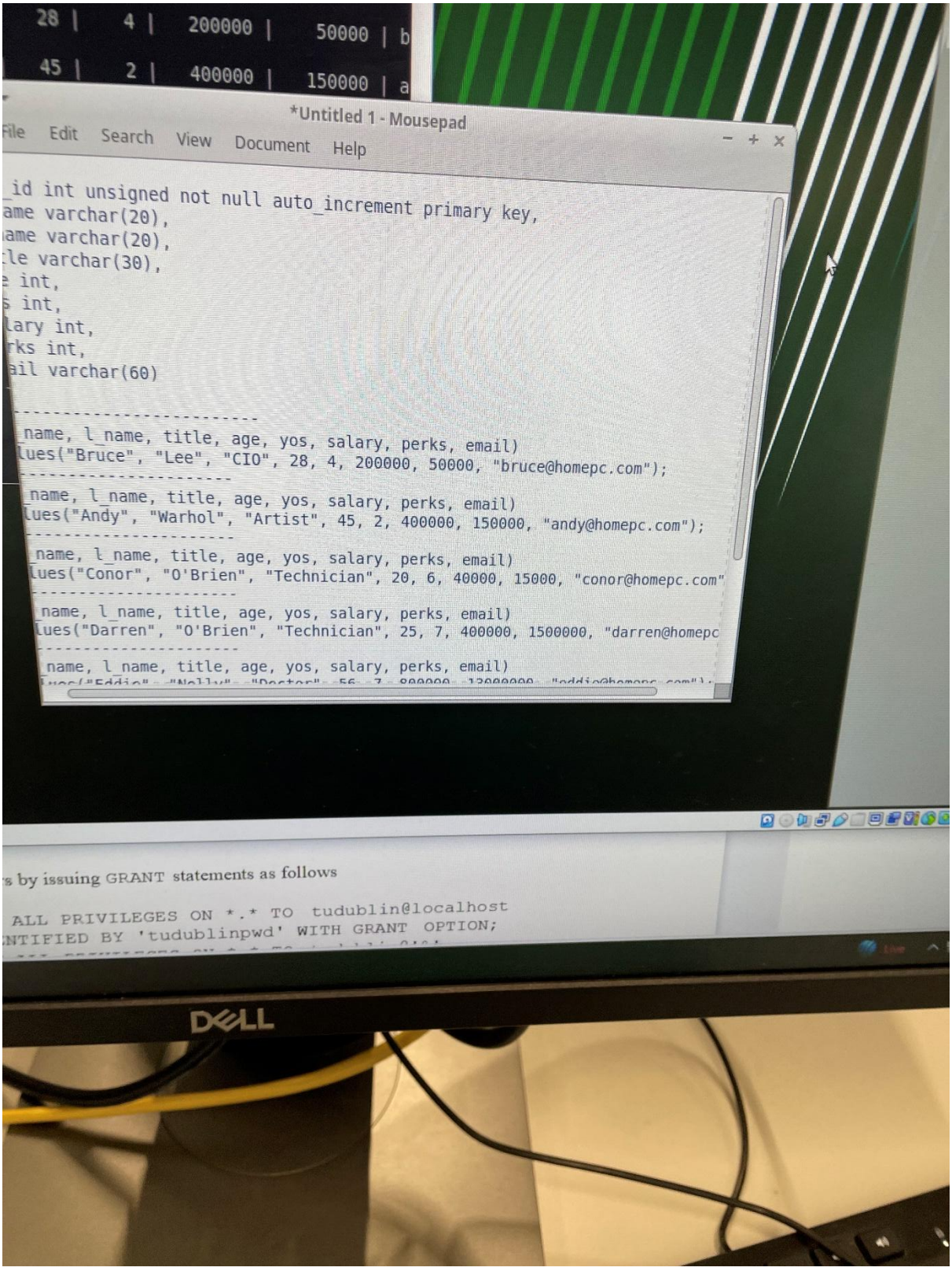
This is the number I done to keep it colourful:



This is a picture showing the filled table:



Another picture with them all:

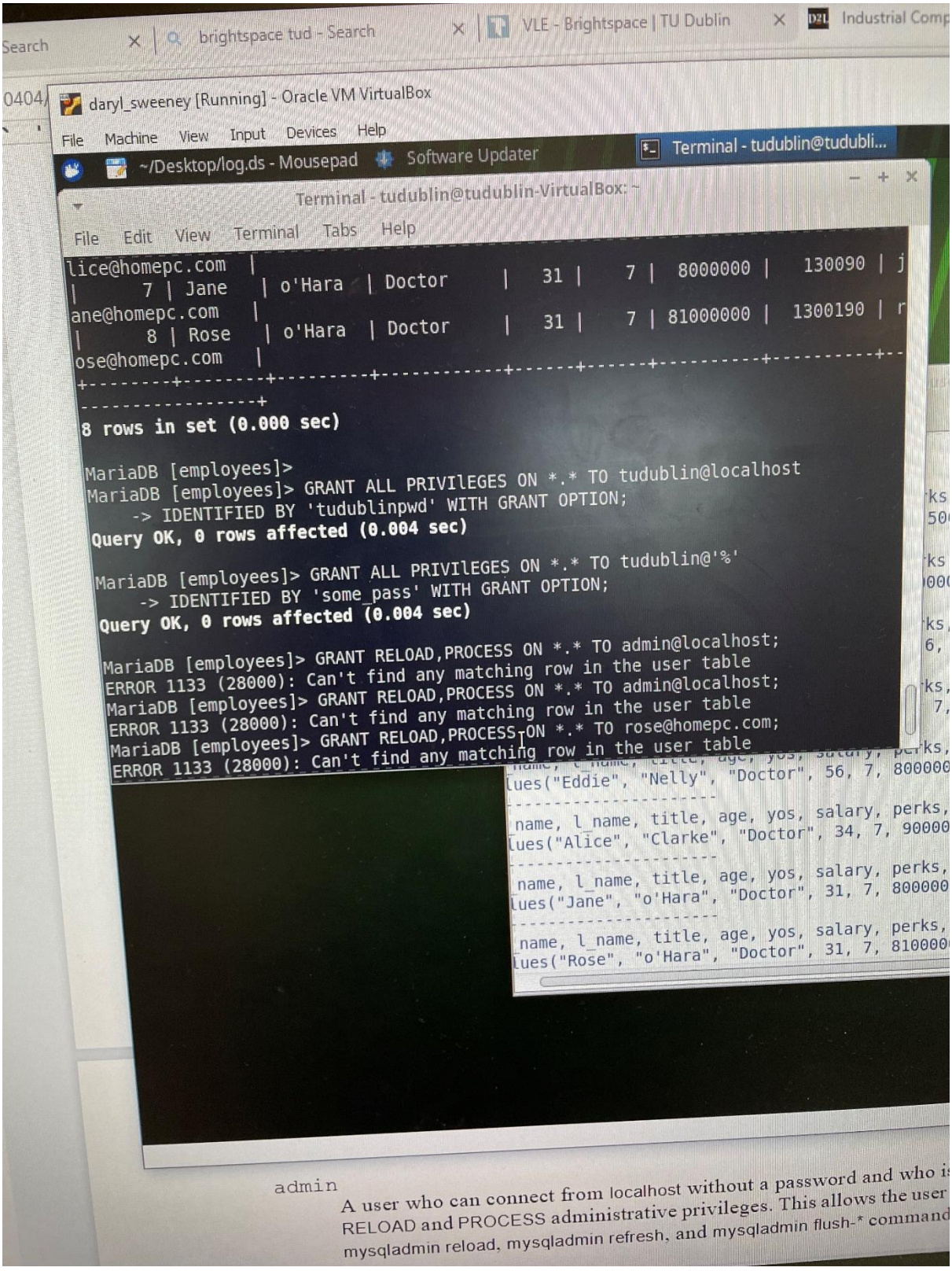


This is the use I got from mousepad application as it was very useful in filling it out:

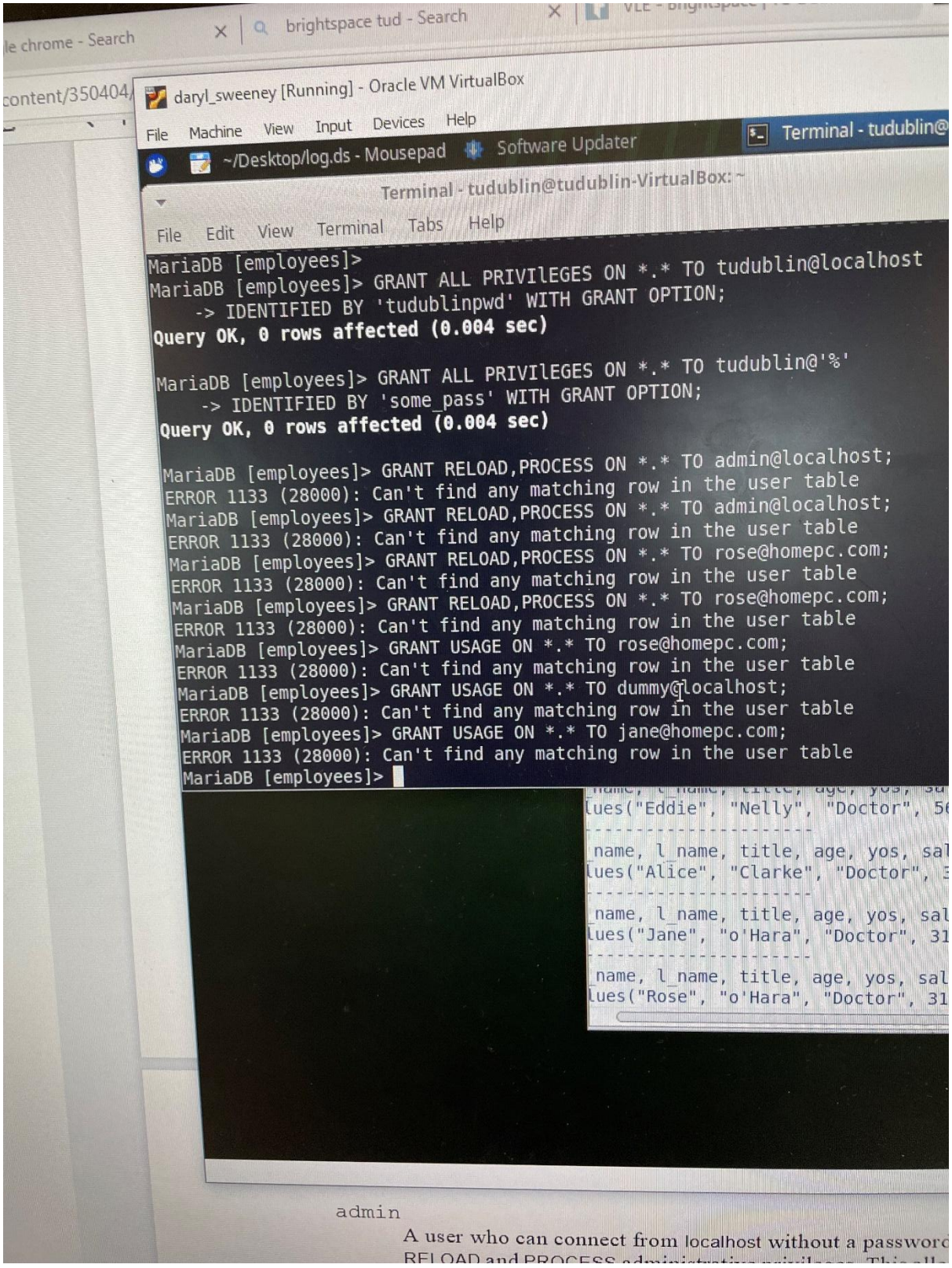
- Adding users:
- This was practice in adding permission to users of that database

```
mariadb> GRANT ALL PRIVILEGES ON *.* TO  tudublin@localhost
    ->     IDENTIFIED BY 'tudublinpwd' WITH GRANT  OPTION;
mariadb> GRANT ALL PRIVILEGES ON *.* TO  tudublin@'%'
    ->     IDENTIFIED BY 'some_pass' WITH GRANT  OPTION;
mariadb> GRANT RELOAD,PROCESS ON *.* TO admin@localhost;
mariadb> GRANT USAGE ON *.* TO  dummy@localhost;
```

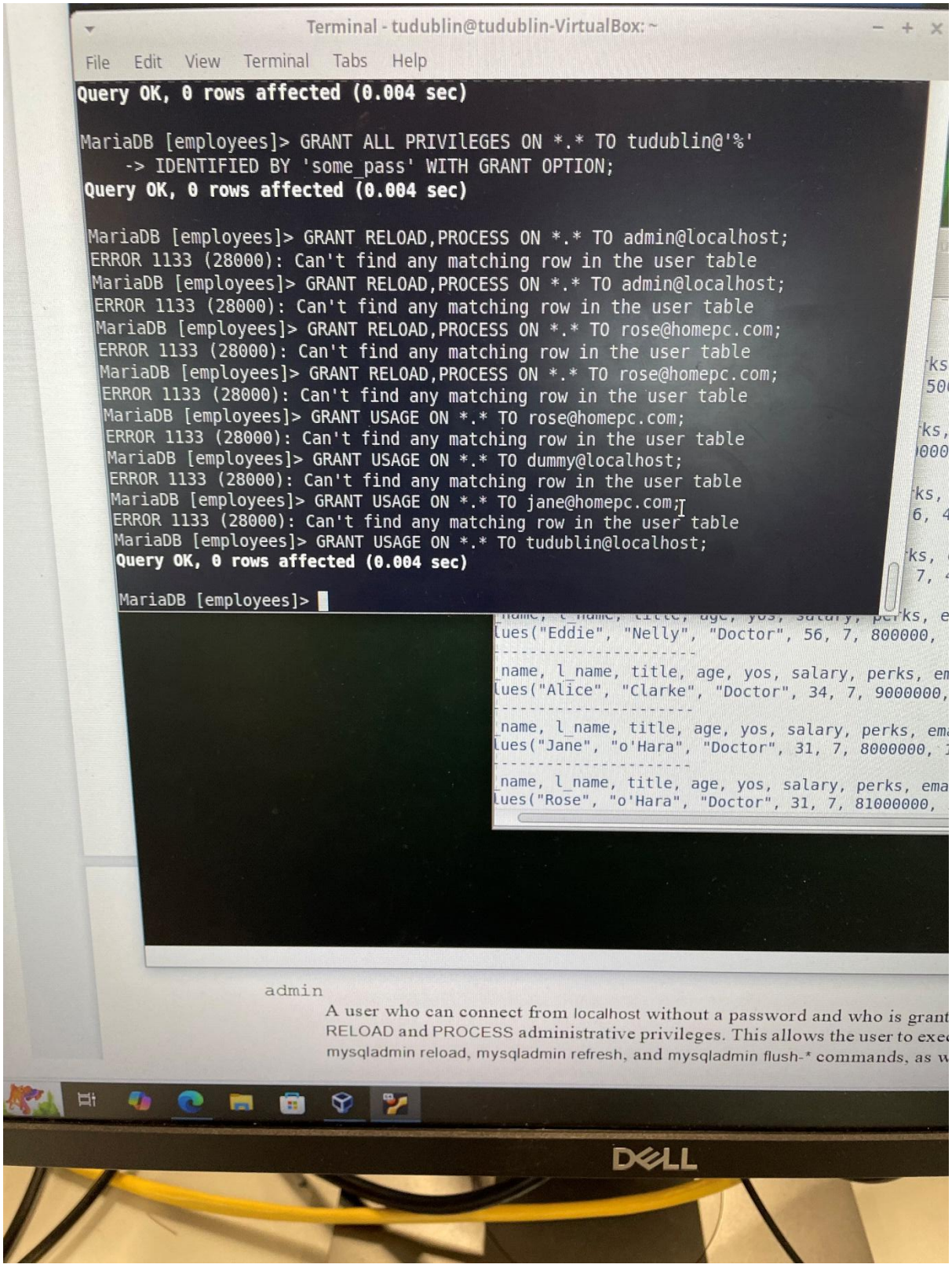
Example of what commands we used to an email or username:



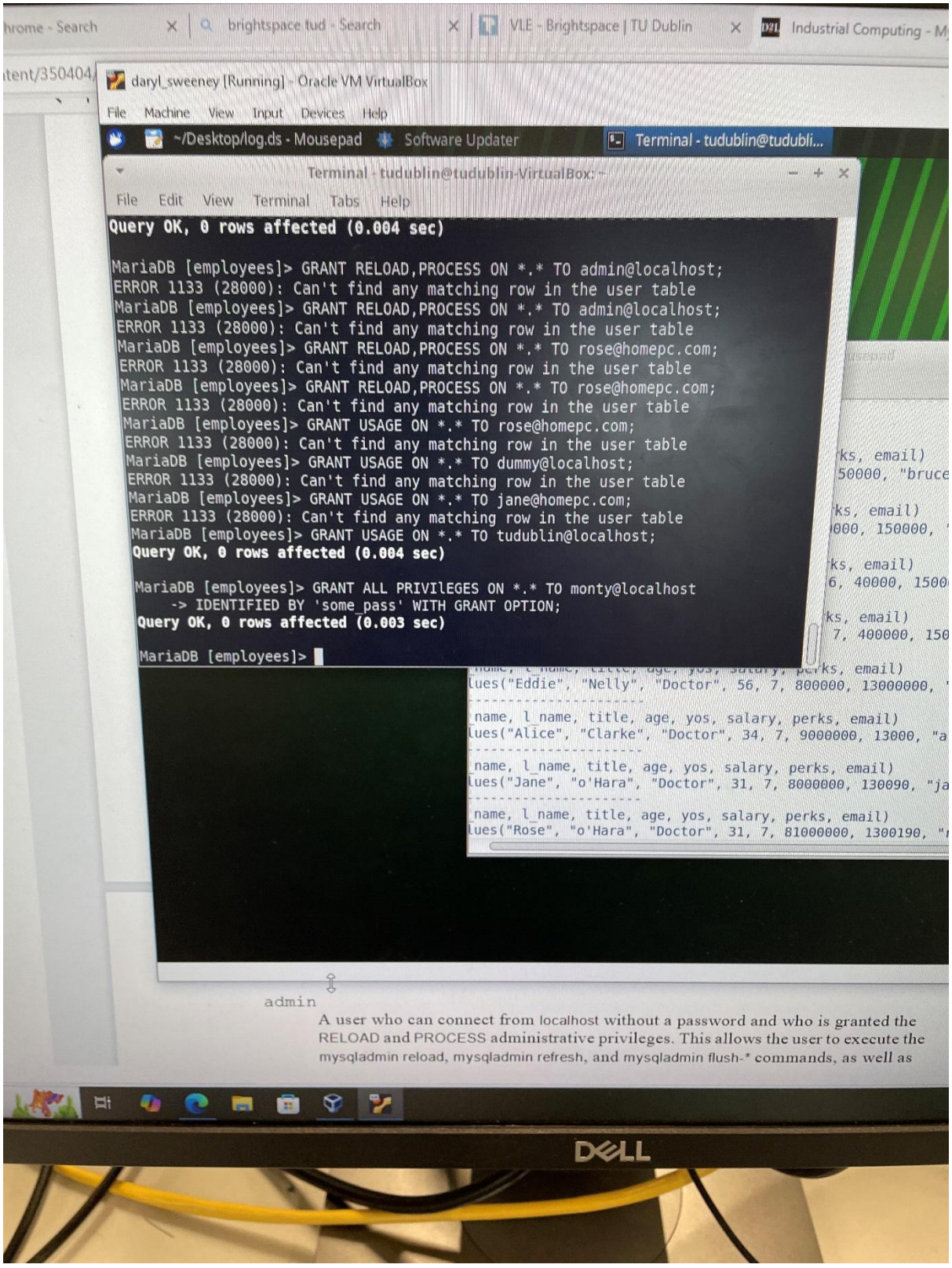
Me using the commands:



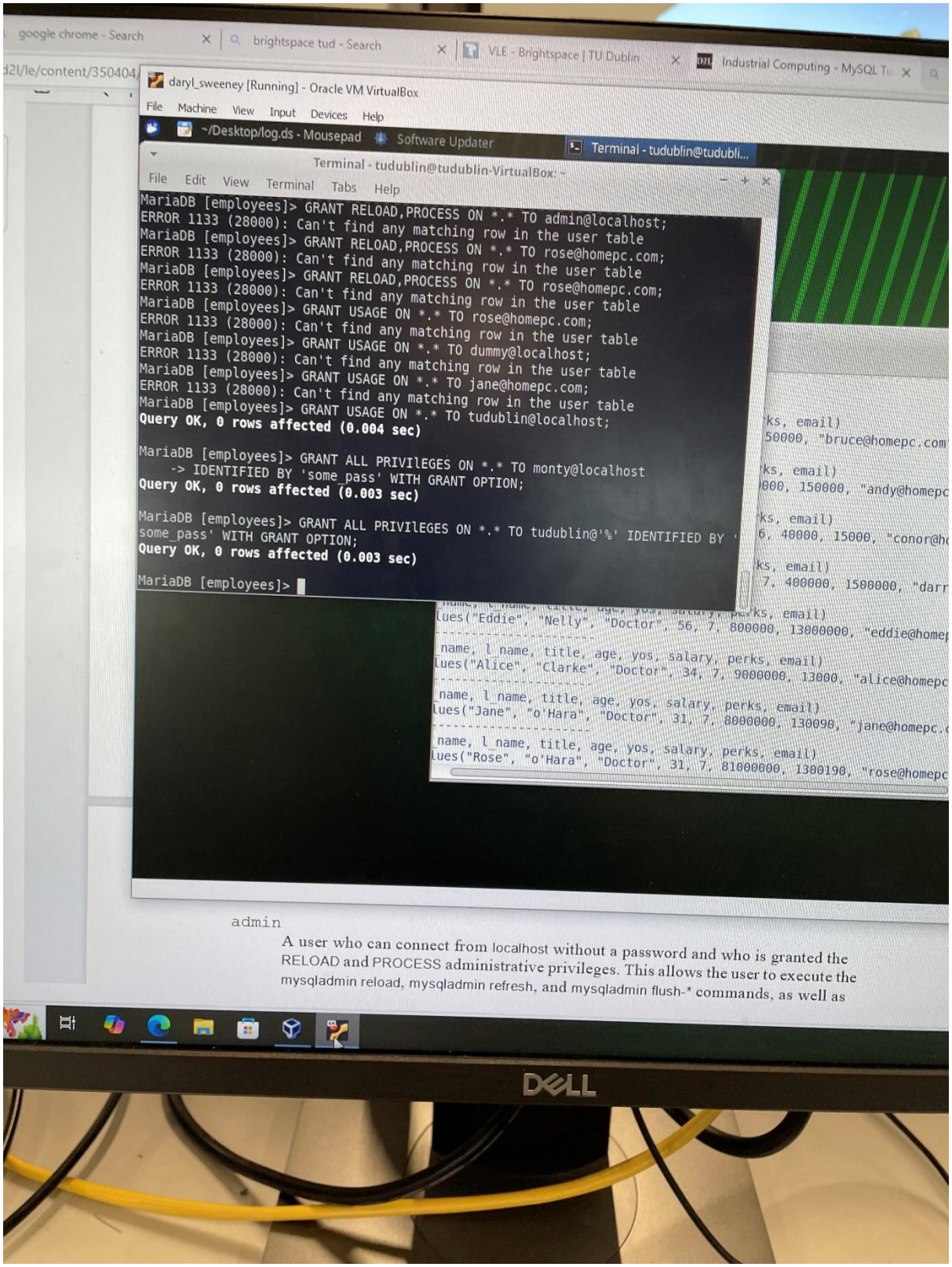
Having to grant emails or users within the database:



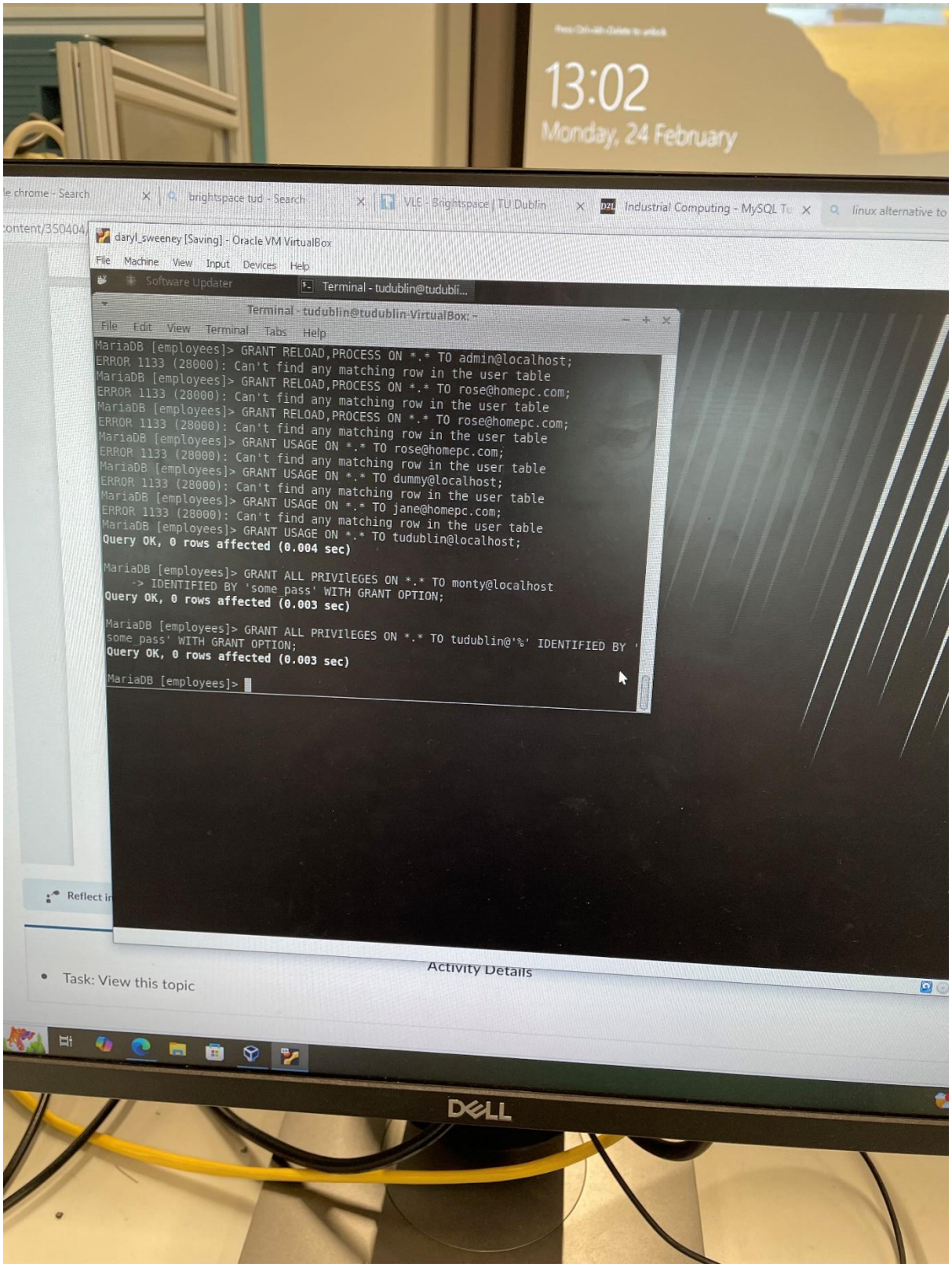
Another example, eventually getting it:


daryl_sweeney [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
~/Desktop/log.ds - Mousepad
Software Updater
Terminal - tudublin@tudubli...
Terminal - tudublin@tudublin-VirtualBox: ~
File Edit View Terminal Tabs Help
Query OK, 0 rows affected (0.004 sec)
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO admin@localhost;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO admin@localhost;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO rose@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO rose@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO rose@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO dummy@localhost;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO jane@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO tudublin@localhost;
Query OK, 0 rows affected (0.004 sec)
MariaDB [employees]> GRANT ALL PRIVILEGES ON *.* TO monty@localhost
-> IDENTIFIED BY 'some_pass' WITH GRANT OPTION;
Query OK, 0 rows affected (0.003 sec)
MariaDB [employees]>
admin
A user who can connect from localhost without a password and who is granted the RELOAD and PROCESS administrative privileges. This allows the user to execute the mysqladmin reload, mysqladmin refresh, and mysqladmin flush-* commands, as well as
DELL

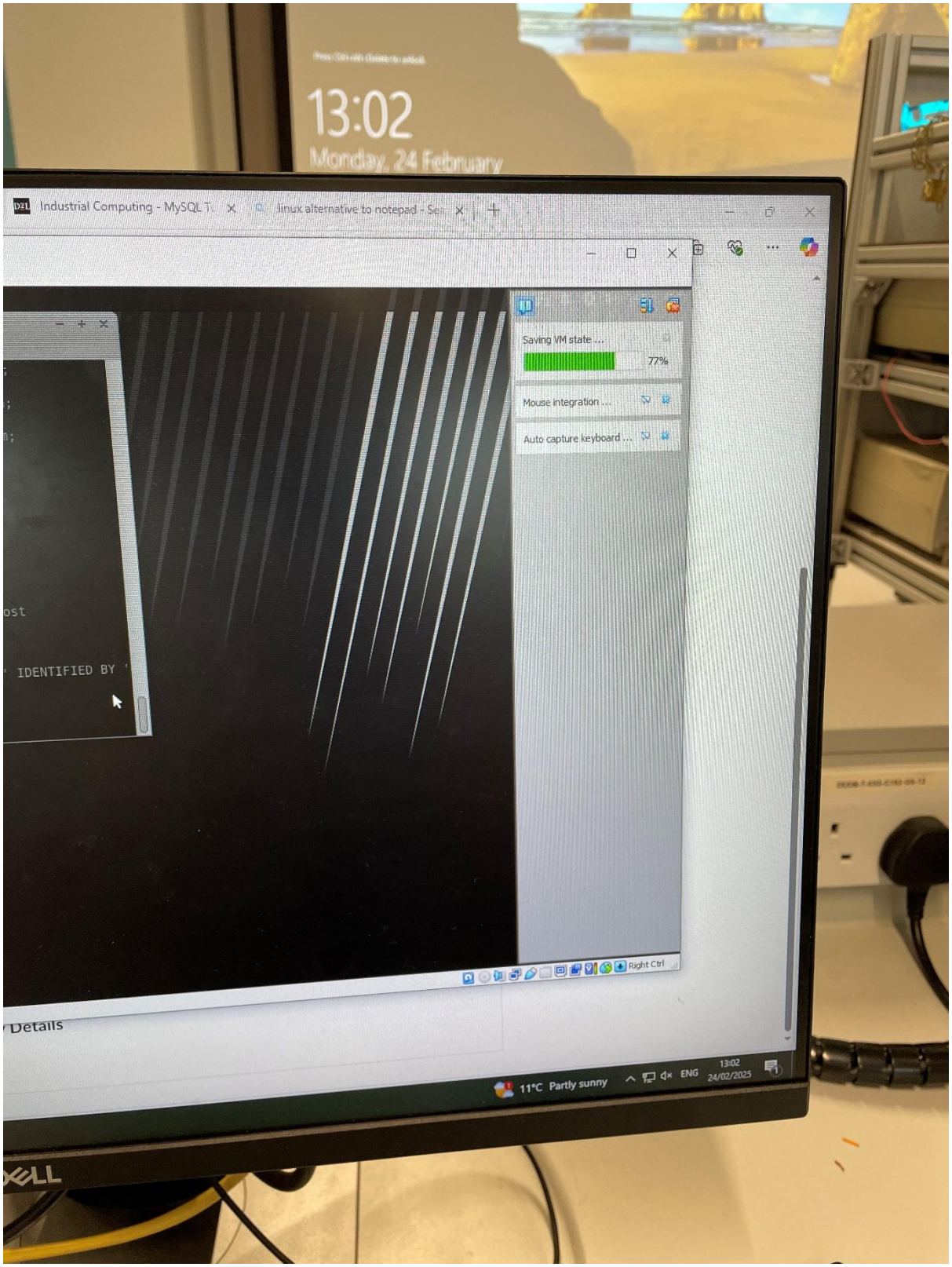
Example:


google chrome - Search
brightspace tud - Search
VLE - Brightspace | TU Dublin
Industrial Computing - MySQL Tu
daryl_sweeney [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
~/Desktop/log.ds - Mousepad
Software Updater
Terminal - tudublin@tudubli...
Terminal - tudublin@tudublin-VirtualBox: ~
File Edit View Terminal Tabs Help
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO admin@localhost;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO rose@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO rose@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO rose@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO dummy@localhost;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO jane@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO tudublin@localhost;
Query OK, 0 rows affected (0.004 sec)
MariaDB [employees]> GRANT ALL PRIVILEGES ON *.* TO monty@localhost
-> IDENTIFIED BY 'some_pass' WITH GRANT OPTION;
Query OK, 0 rows affected (0.003 sec)
MariaDB [employees]> GRANT ALL PRIVILEGES ON *.* TO tudublin@'%' IDENTIFIED BY '
some_pass' WITH GRANT OPTION;
Query OK, 0 rows affected (0.003 sec)
MariaDB [employees]>
admin
A user who can connect from localhost without a password and who is granted the
RELOAD and PROCESS administrative privileges. This allows the user to execute the
mysqladmin reload, mysqladmin refresh, and mysqladmin flush-* commands, as well as
DELL

Example:


13:02
Monday, 24 February
daryl_sweeney [Saving] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal - tudublin@tudublin-VirtualBox: ~
File Edit View Terminal Tabs Help
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO admin@localhost;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO rose@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT RELOAD,PROCESS ON *.* TO rose@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO rose@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO dummy@localhost;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO jane@homepc.com;
ERROR 1133 (28000): Can't find any matching row in the user table
MariaDB [employees]> GRANT USAGE ON *.* TO tudublin@localhost;
Query OK, 0 rows affected (0.004 sec)
MariaDB [employees]> GRANT ALL PRIVILEGES ON *.* TO monty@localhost
-> IDENTIFIED BY 'some_pass' WITH GRANT OPTION;
Query OK, 0 rows affected (0.003 sec)
MariaDB [employees]> GRANT ALL PRIVILEGES ON *.* TO tudublin@'%' IDENTIFIED BY '
some_pass' WITH GRANT OPTION;
Query OK, 0 rows affected (0.003 sec)
MariaDB [employees]>
Activity Details
Task: View this topic
DELL

Example:



Saving and shutting down:

Appendix Lab 4:

Laboratory Log: 03/03/2025

In this laboratory, we went on to our virtual machines, exited the databases from the previous week which were saved and installed apache2. We did this by these commands:

(Ctrl + c) to come out of the database,

Then typing ,

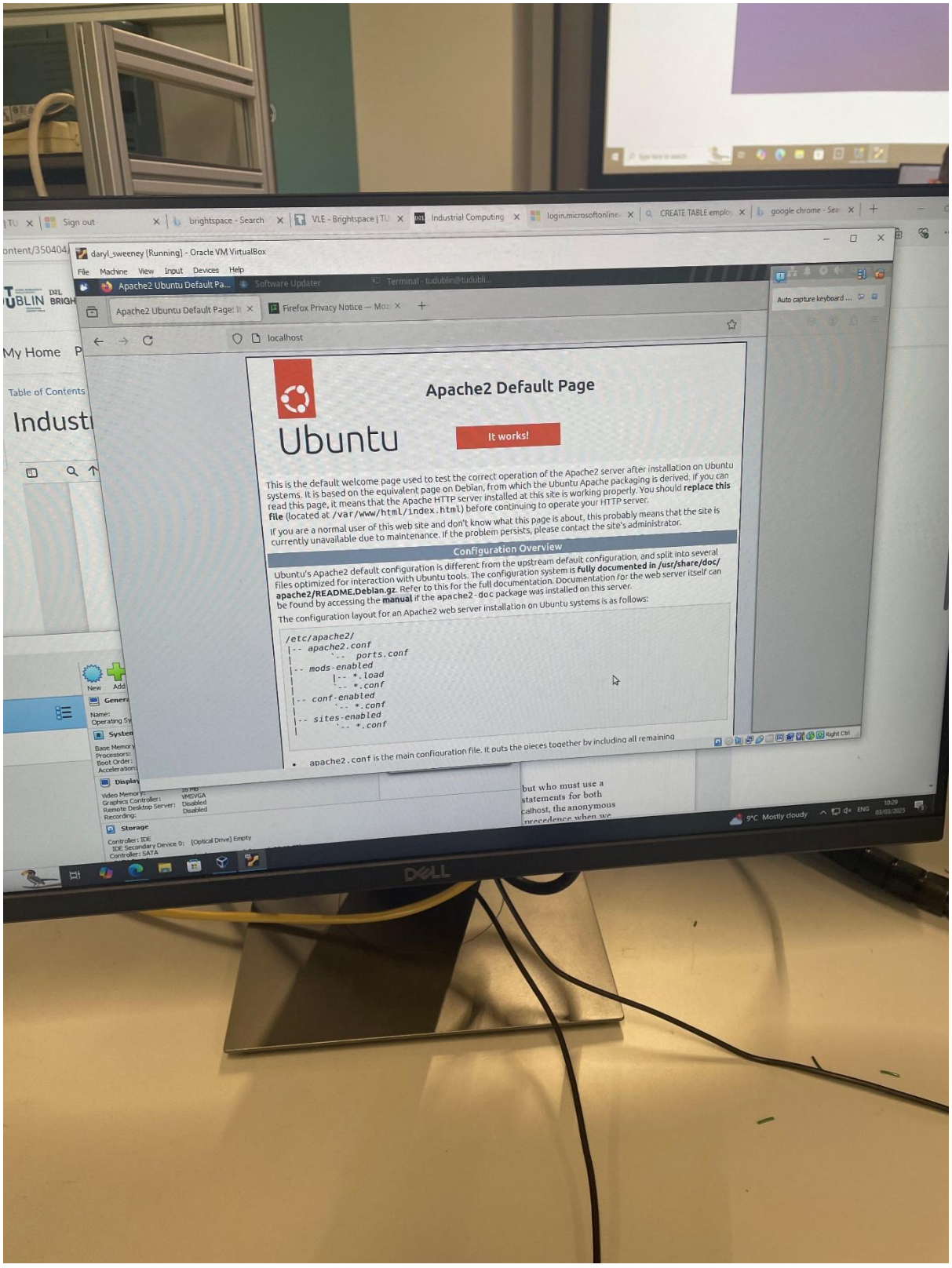
(Sudo apt update) to update the server,

Then,

(sudo apt install apache2) to install apache.

We then went to “local host” in browser,

Then the apache message came up.



This showing it couldn't be accessed only through the server but was displayed that the last letters in the address were html:

Then we were to enter into the linux version of windows explorer this address:

/var/www/html

We were quoted why this was happening and what html was?

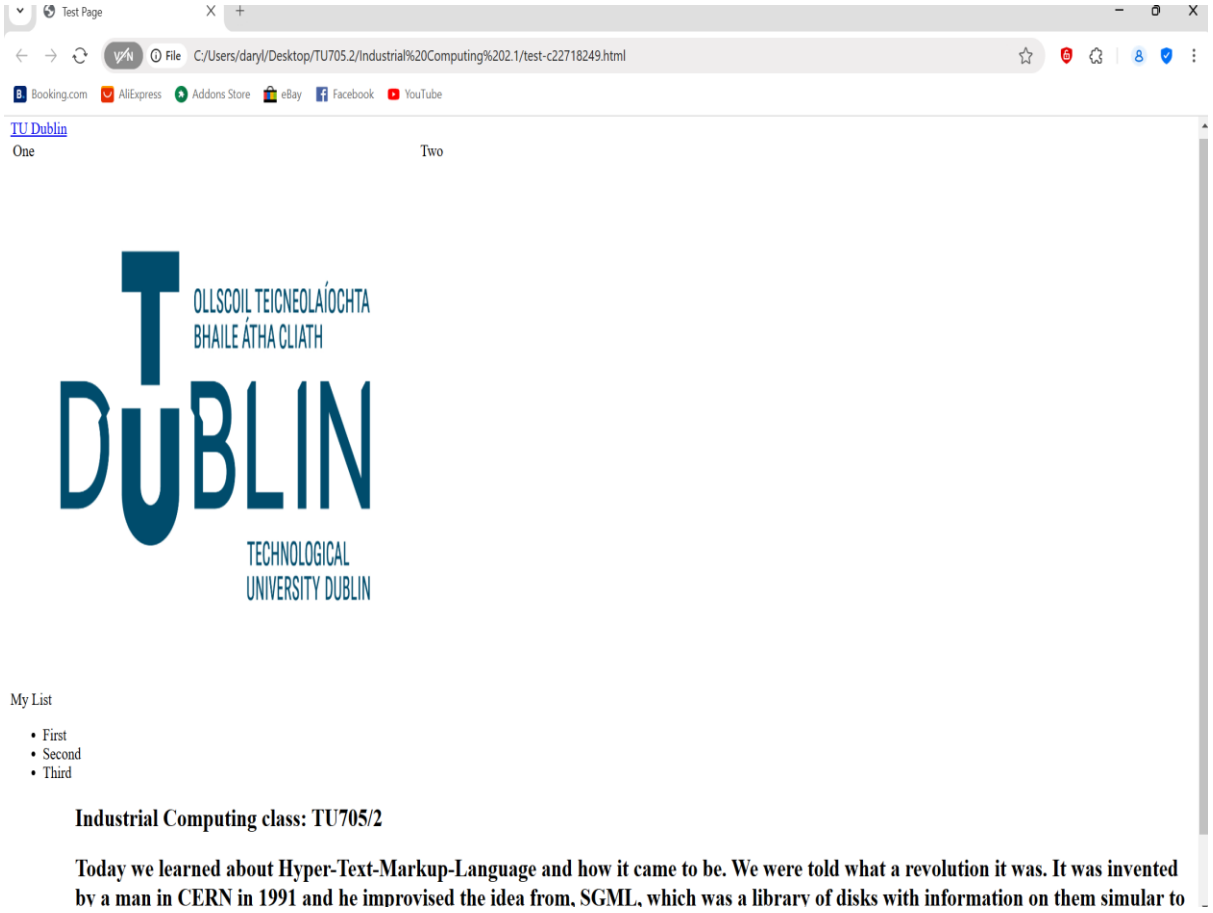
It was then explained that html is a document page system that was invented in 1991 by this man who was working in CERN's Large Hadron collider experiment which was and still is an ongoing physics experiment trying to determine the beginnings of the universe and the components of subatomic particles and so on. He noticed that in a library they had a disk system to store and retrieve information. He improvised that invention by making a hyperlink or address between computers to eachothers web pages and in this way there was no need to seek a disk. He called it hypertext markup language(html) and he never patented the invention. If he had so now would be by far the richest man in the world because of his invention. This is the backbone of the internet basically and without it it wouldn't exist. It allows pages to be viewed remotely anywhere, enabling the worldwide spread of information.

We then were introduced to html language and were given a crash course in html. We done our first webpage using an example using the headings and closing the headings. We were given the basic structure and an example of what would be a beginners format. We were shown a title, a basic table, a link, making a picture a link, then we made our page. Heres the code:

```
<html>
<head>
<title> Test Page </title>
</head>
<body>
<table>
<tr><td>
One</td><td>Two</td>
<tr><td><a href="https://tudublin.ie"><img src="TUDublin_RGB.jpg" Height="500" Width="500"></a></td>
<ta><a href="https://tudublin.ie">TU Dublin
</a>
<rd></tr>
</table>
</body>
<headings 1>My List
</heading 1>
<ul>
<li> First </li>
<li> Second </li>
<li> Third </li>
<ul>
<h2> Industrial Computing class: TU705/2 <h2>
<p1> Today we learned about Hyper-Text-Markup-Language and how it came to be.
We were told what a revolution it was. It was invented by a man in CERN in 1991 and he improvised the idea from,
SGML, which was a library of disks with information on them simular to HTML.
He improvised the process by using hyperlinks to each text file and revolutionised the process and paving
the way for the internet in the 90's up to today.</p1>
<button type="button">Click Me!</button>

</html>
```

I added a little myself to practice:

Heres an example of what the webpage looked like, although primitive little steps lead to big things.



Showing the things I mentioned:

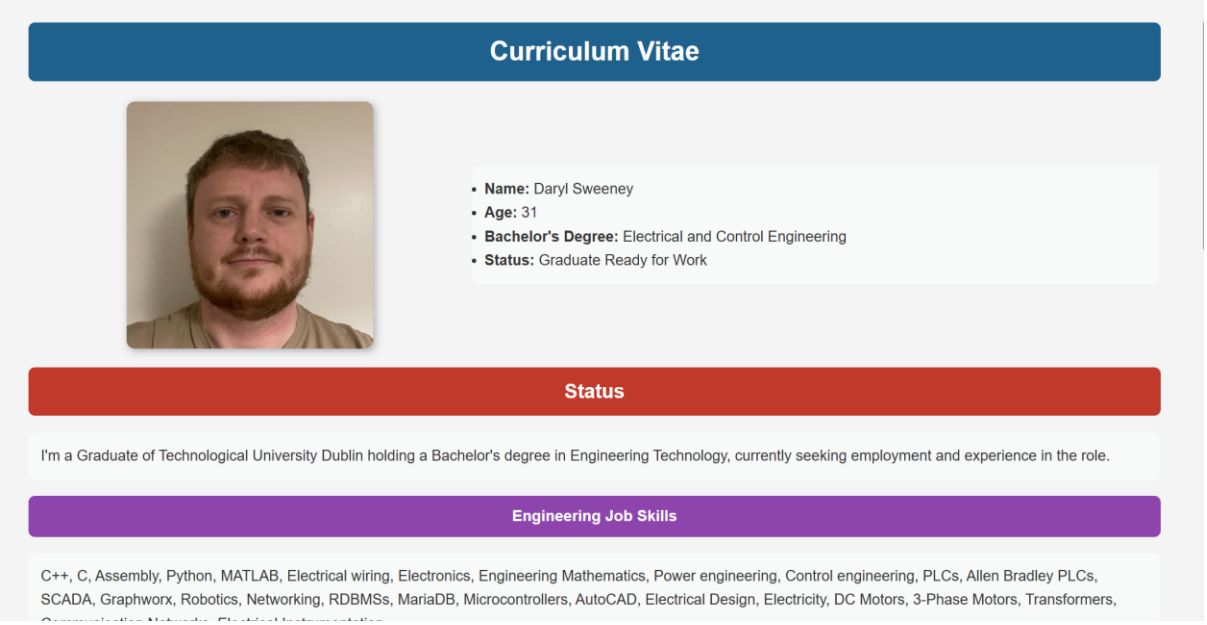
We were also shown a list as shown.

We were then given a project to come up with a cv and were expected to have it in the next few weeks I think. We were given a site to search for “w3schools html”. This would be our reference and we were expected to work on our own initiative.

The basic language was html but we were encouraged to add css or anything extra that was to go with it. Heres a pic of my code!

```
<!DOCTYPE html>
<html style="background-color:#F4F4F4; color:#333;">
<head>
<title>Curriculum Vitae</title>
<style>
  body {
    font-family: 'Open Sans', sans-serif;
    margin: 30px;
    padding: 20px;
    line-height: 1.6;
  }

  h1, h2, h3, h4, h5, h6 {
    font-family: 'Montserrat', sans-serif;
    text-align: center;
    padding: 10px;
    color: white;
    border-radius: 8px;
  }

  h1 { background-color: #1F618D; }  /* Deep Blue */
  h2 { background-color: #C0392B; }  /* Strong Red */
  h3 { background-color: #8E44AD; }  /* Rich Purple */
  h4 { background-color: #E67E22; }  /* Warm Orange */
  h5 { background-color: #16A085; }  /* Deep Teal */
  h6 { background-color: #2C3E50; }  /* Dark Navy */

  ul, p {
    font-size: 18px;
    background-color: #F8F9F9; /* Soft gray */
    padding: 15px;
    border-radius: 10px;
  }

  table {
    width: 100%;
    margin-top: 20px;
  }

  img {
    border-radius: 10px;
    display: block;
    margin: auto;
    box-shadow: 3px 3px 10px rgba(0, 0, 0, 0.3);
  }
```

I checked the resources and the original code I had was messy but working then I learned how to neaten it up, realising in the css that <ul>,<p>, <h> and so forth only have to be mentioned once unless you want them differently. I checked with chat gpt to help me with that and contrasting colours in HEX which I hadn't a clue, also I had a good pic of myself and used the skills I had on my LinkedIn profile to help me.



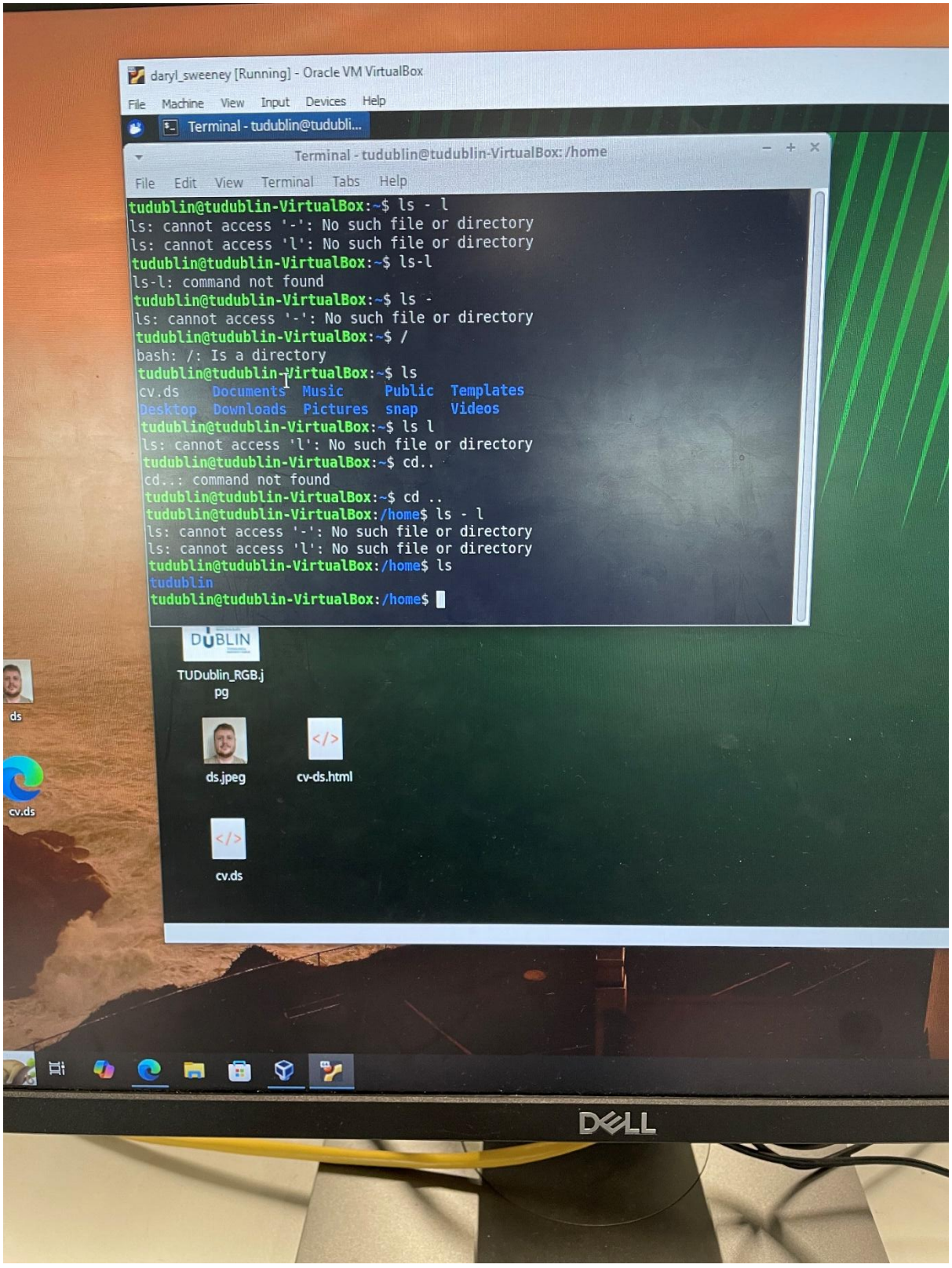
Heres a pic of it:

I also had a button at the bottom to go to the top of the page for extra functionality. That's what I did that day!

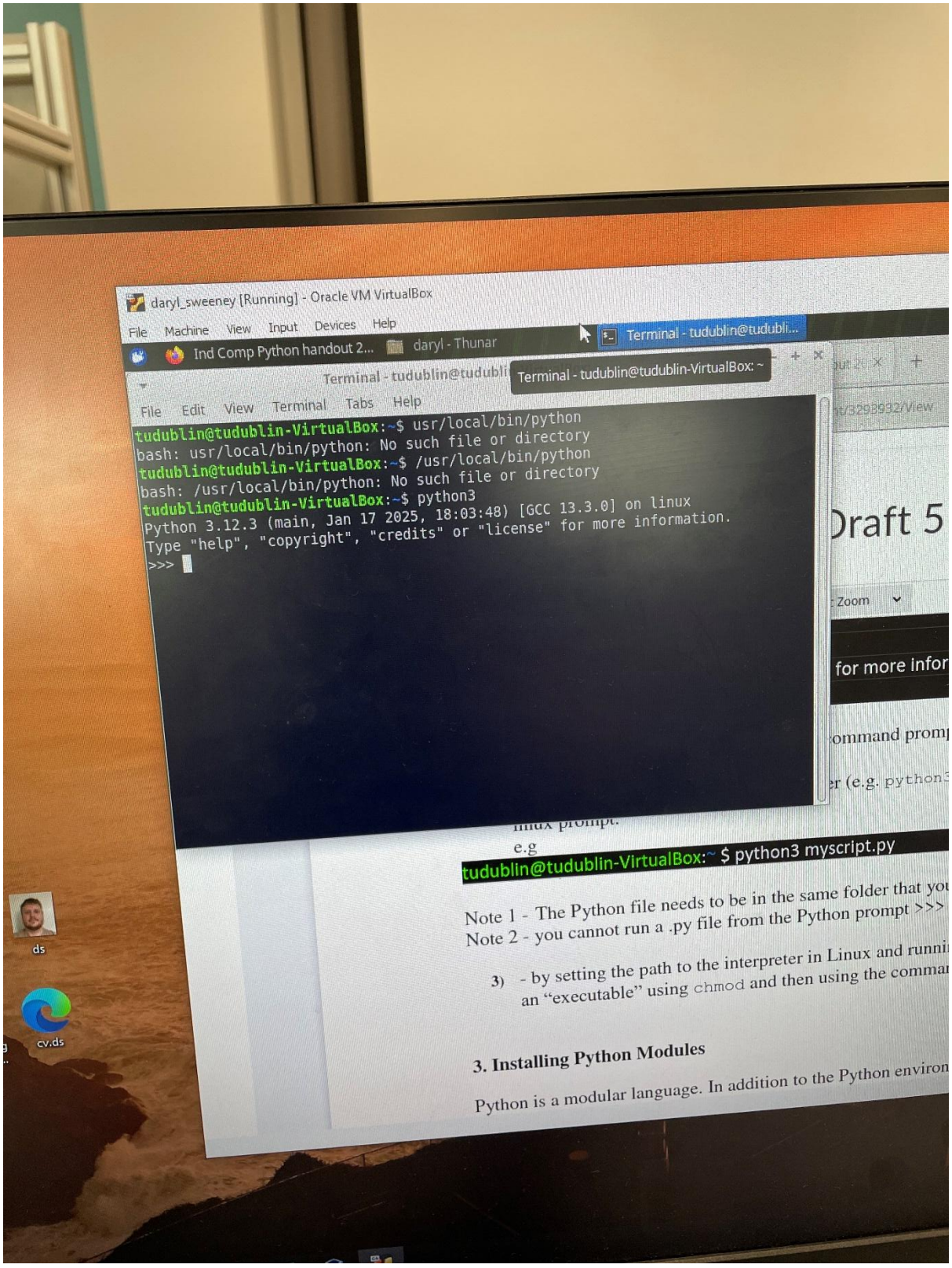
Appendix Lab 5:

Laboratory Log: 10/03/2025

In this laboratory, we opened our virtual machine and went into the terminal. We were introduced to the "python" programming language through the Linux terminal. We were given a broader explanation of "Linux" and shown more commands. As you can see, "Python3" was already installed on the terminal, but scripts were kept in the virtual machine, saved and run as ".py" files or "Python" script files with that extension.



Pic 1: Here shown testing out a couple of new commands like “l” for list and “cd” for change directory.



Pic 2: Here shown that "Python3" was already intalled in linux. When run, it showed three arrows meaning that python was accepting commands on the command line.

Python commands 1:

This is showing the first piece of work done in Python:

---

python3:

```
tudublin@tudublin-VirtualBox:~$ usr/local/bin/python
bash: usr/local/bin/python: No such file or directory
tudublin@tudublin-VirtualBox:~$ /usr/local/bin/python
bash: /usr/local/bin/python: No such file or directory
tudublin@tudublin-VirtualBox:~$ python3
Python 3.12.3 (main, Jan 17 2025, 18:03:48) [GCC 13.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> python3 myscript.py
  File "<stdin>", line 1
    python3 myscript.py
            ^^^^^^^^
SyntaxError: invalid syntax
>>> ls
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'ls' is not defined
>>> cd ..
  File "<stdin>", line 1
    cd ..
       ^
SyntaxError: invalid syntax
>>> cd..
  File "<stdin>", line 1
    cd..
      ^
```

```
SyntaxError: invalid syntax
>>> exit
Use exit() or Ctrl-D (i.e. EOF) to exit
>>>
tudublin@tudublin-VirtualBox:~$ ls
cv.ds   Documents  Music    Public  Templates
Desktop  Downloads  Pictures  snap   Videos
tudublin@tudublin-VirtualBox:~$ cd Desktop
tudublin@tudublin-VirtualBox:~/Desktop$ ./myscript.py
bash: ./myscript.py: Permission denied
tudublin@tudublin-VirtualBox:~/Desktop$ python3
Python 3.12.3 (main, Jan 17 2025, 18:03:48) [GCC 13.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> my1stString="this is a string"
>>> my2ndString=" and here is another"
>>> print (my1stString)
this is a string
>>> My1stString="this is a string"
>>> My3rdString= My1stString + My2ndString
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'My2ndString' is not defined. Did you mean: 'my2ndString'?
>>> My2ndString=" and here is another"
>>> My3rdString= My1stString + My2ndString
>>> print(My3rdString)
this is a string and here is another
>>> age=36
>>> print(str(age))
```

```
>>> mystring=" I am a student on TU705 "
>>> wordlist=mystring.split()
>>> print(wordlist)
['I', 'am', 'a', 'student', 'on', 'TU705']
>>> restring=''.join(wordlist)
>>> print(restring)
IamastudentonTU705
>>> restring=' '.join(wordlist)
>>> print(restring)
I am a student on TU705
>>> mystring.slip()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'str' object has no attribute 'slip'. Did you mean: 'split'?
>>> print(mystring)
 I am a student on TU705
>>> mystring.strip()
'I am a student on TU705'
>>> print(mystring.upper())
 I AM A STUDENT ON TU705
>>> print(mystring.lower())
 i am a student on tu705
>>> my_list = [ 1, 2, 3, 4]
>>>  #We reference list items with integer keys
>>> print(my_list[0])
1
>>> print(my_list[1])
```

```
2
>>> print(my_list[2])
3
>>> print(my_list[3]
... print(my_list[3])
  File "<stdin>", line 1
    print(my_list[3]
          ^^^^^^^^^
SyntaxError: invalid syntax. Perhaps you forgot a comma?
>>> print(my_list[3])
4
>>> # Lists can also contain strings
>>> my_list = ['apples','oranges','pears','plums']
>>> print(my_list[0])
apples
>>> print(my_list[1])
oranges
>>> print(my_list[2])
pears
>>> print(my_list[3])
plums
>>> my_dictionary = {'fruit': 'apple', 'colour': 'red', 'college': 'TU Dublin', 'course_year': 2}
>>> print(my_dictionary['fruit'])
apple
>>> print(my_dictionary['colour'])
red
>>> print(my_dictionary['college'])
TU Dublin
```

```
>>> print(my_dictionary['course_year'])
2
>>> print('My favourite fruit is the ' + my_dictionary['fruit'] + ', my
  File "<stdin>", line 1
    print('My favourite fruit is the ' + my_dictionary['fruit'] + ', my
                                                                      ^
SyntaxError: unterminated string literal (detected at line 1)
>>> favourite colour is ' + my_dictionary['colour'] + ', my college is ' +
  File "<stdin>", line 1
    favourite colour is ' + my_dictionary['colour'] + ', my college is ' +
              ^^^^^^
SyntaxError: invalid syntax
>>> my_dictionary['college'] + ' and I am in year ' +
  File "<stdin>", line 1
    my_dictionary['college'] + ' and I am in year ' +
                                                   ^
SyntaxError: invalid syntax
>>> print('My favourite fruit is the ' + my_dictionary['fruit'] + ', My favourite colour is ' +
my_dictionary['colour'] + ', my college is ' + str(my_dictionary['course_year']))
My favourite fruit is the apple, My favourite colour is red, my college is 2
>>>
KeyboardInterrupt
>>>
```

We were given an assignment script to familiarise ourselves with the simple commands of Python. This showed us how simple and powerful it was to C or C++ and so on.

Python commands 2:

This is showing the second piece of work done in Python:

```
tudublin@tudublin-VirtualBox:~$ ls
cv.ds   Documents  Music    Public  Templates
Desktop  Downloads  Pictures  snap   Videos
tudublin@tudublin-VirtualBox:~$ cd..
cd..: command not found
tudublin@tudublin-VirtualBox:~$ cd ..
tudublin@tudublin-VirtualBox:/home$ ls
tudublin
tudublin@tudublin-VirtualBox:/home$ cd tudublin
tudublin@tudublin-VirtualBox:~$ python3 func.py
python3: can't open file '/home/tudublin/func.py': [Errno 2] No such file or directory
tudublin@tudublin-VirtualBox:~$ python3
Python 3.12.3 (main, Jan 17 2025, 18:03:48) [GCC 13.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> python3 func.py
  File "<stdin>", line 1
    python3 func.py
          ^^^^
SyntaxError: invalid syntax
>>>
tudublin@tudublin-VirtualBox:~$ ls
cv.ds   Documents  Func.py  Pictures  snap     Videos
Desktop  Downloads  Music    Public   Templates
```

```
tudublin@tudublin-VirtualBox:~$ cd Desktop
tudublin@tudublin-VirtualBox:~/Desktop$ cd..
cd..: command not found
tudublin@tudublin-VirtualBox:~/Desktop$ cd../
bash: cd../: No such file or directory
tudublin@tudublin-VirtualBox:~/Desktop$ ls
cv.ds      daryl   log.ds      test-c22718249.html
cv-ds.html  ds.jpeg  myscript.py  TUDublin_RGB.jpg
cv.ds.html  Func.py  python.ds
tudublin@tudublin-VirtualBox:~/Desktop$ Func.py
Func.py: command not found
tudublin@tudublin-VirtualBox:~/Desktop$ cd Desktop
bash: cd: Desktop: No such file or directory
tudublin@tudublin-VirtualBox:~/Desktop$ python3 func.py
python3: can't open file '/home/tudublin/Desktop/func.py': [Errno 2] No such file or
directory
tudublin@tudublin-VirtualBox:~/Desktop$ cd
tudublin@tudublin-VirtualBox:~$ python3 func.py
python3: can't open file '/home/tudublin/func.py': [Errno 2] No such file or directory
tudublin@tudublin-VirtualBox:~$ cd Home
bash: cd: Home: No such file or directory
tudublin@tudublin-VirtualBox:~$ python3 Func.py
  File "/home/tudublin/Func.py", line 3
    higher than the definition above
    ^
IndentationError: expected an indented block after function definition on line 1
tudublin@tudublin-VirtualBox:~$ python3 func.py
python3: can't open file '/home/tudublin/func.py': [Errno 2] No such file or directory
```

tudublin@tudublin-VirtualBox:~$ python3 func.py

python3: can't open file '/home/tudublin/func.py': [Errno 2] No such file or directory

tudublin@tudublin-VirtualBox:~$ python3 func.py

Then using Python to open a .py script so we could get a go at making a simple program.

Then getting to this and Damon coming down to the point of the conversation that, with C or C++ say or other programming languages if you want a device to do something, say a microcontroller, C is needed for Arduino but with raspberry you can use Python or calculate mathematics say, they are very slow in the case that you have to write the code and compile the code for it to work, but with python you just download a script and enter three or four lines of code say, and it will run making things way quicker and more powerful. Here's what else I run that day and the scripts:

File "/home/tudublin/func.py", line 3

higher than the definition above

^

IndentationError: expected an indented block after function definition on line 1

tudublin@tudublin-VirtualBox:~$ python3 func.py

File "/home/tudublin/func.py", line 3

higher than the definition above

^

IndentationError: expected an indented block after function definition on line 1

tudublin@tudublin-VirtualBox:~$ python3 func.py

File "/home/tudublin/func.py", line 5

return result

^^^^^^^^^^^^

SyntaxError: 'return' outside function

tudublin@tudublin-VirtualBox:~$ python3 func.py

I am now in the add function

I am now in the add function

I am now in the add function

Adding 1 and 2 gives 3

```
Calling the add() function with 30 and 45 as parameters gives: 75
Again, calling add(100, 6) returns 106
tudublin@tudublin-VirtualBox:~$ d = "Fri"
d: command not found
tudublin@tudublin-VirtualBox:~$ python3 func.py
  File "/home/tudublin/func.py", line 18
    print("Great, it's the weekend!")
    ^
IndentationError: expected an indented block after 'if' statement on line 17
tudublin@tudublin-VirtualBox:~$ python3 func.py
  File "/home/tudublin/func.py", line 10
    print('Adding 1 and 2 gives ' + str(result_1))
IndentationError: unexpected indent
tudublin@tudublin-VirtualBox:~$ python3 func.py
  File "/home/tudublin/func.py", line 10
    print('Adding 1 and 2 gives ' + str(result_1))
IndentationError: unexpected indent
tudublin@tudublin-VirtualBox:~$ python3 func.py
  File "/home/tudublin/func.py", line 10
    print('Adding 1 and 2 gives ' + str(result_1))
IndentationError: unexpected indent
tudublin@tudublin-VirtualBox:~$ python3
Python 3.12.3 (main, Jan 17 2025, 18:03:48) [GCC 13.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import antigravity
>>>
tudublin@tudublin-VirtualBox:~$ ^C
tudublin@tudublin-VirtualBox:~$
```

heres the “func.py” script:

```
def add(a, b):
# Lines of code inside a Python function are indented one level higher than the definition above
    print('I am now in the add function')
    result = a + b
    return result
# The code that follows is un-indented so it is not part of the above function
result_1 = add(1, 2)
result_2 = add(30, 45)
result_3 = add(100, 6)
print('Adding 1 and 2 gives ' + str(result_1))
print('Calling the add() function with 30 and 45 as parameters gives: '
+ str(result_2))
print('Again, calling add(100, 6) returns ' + str(result_3))
# Conditional Statements
d = "Fri"
6
if d == "Fri":
print("Great, it's the weekend!")
elif d == "Sun":
print("Weekend is over, back to work tomorrow!")
else:
print("Can't wait for the weekend!")
# Loops
# For Loops
for i in range(0, 5):
```

```
print("For Loop number: " + str(i))

# While Loops

j = 5

while j > 0:

print("While Loop number: " + str(j))

j -= 1
```

heres the “func.py” script indented properly which is very important

```
def add(a, b):

# Lines of code inside a Python function are indented one level higher than the definition above

    print('I am now in the add function')

    result = a + b

    return result

# The code that follows is un-indented so it is not part of the above function

result_1 = add(1, 2)

result_2 = add(30, 45)

result_3 = add(100, 6)

    print('Adding 1 and 2 gives ' + str(result_1))

    print('Calling the add() function with 30 and 45 as parameters gives: '

+ str(result_2))

    print('Again, calling add(100, 6) returns ' + str(result_3))
```

heres the “conditional.py” script. Notice the importance ofspacing and indentation and comments are also denoted by # sign.

```
# Conditional Statements

    d = "Fri"

if d == "Fri":

    print("Great, it's the weekend!")

elif d == "Sun":
```

```
        print("Weekend is over, back to work tomorrow!")
else:
        print("Can't wait for the weekend!")
# Loops
# For Loops
for i in range(0, 5):
        print("For Loop number: " + str(i))
# While Loops
j = 5
while j > 0:
        print("While Loop number: " + str(j))
j -= 1
```

Concluding:

the Python programming language demonstration I have seen in this lab has great potential. I have seen little to how powerful a language it is and is emphasised by Damon. I have always wanted to learn it but have really on encountered it through Damon and Jupyter notebook for the maths tutorial in the second year of level 8, solving laplace transforms and z-transforms. MATLAB is a very powerful mathematical processing and simulation tool and is very expensive. As Damon has said, you can do exactly the same thing using Python and more. Python is free so it's easy to see the obvious choice in my future as an engineer

Appendix Lab 6:

Laboratory Log: 24/03/25

In today's laboratory we finished the "Python" worksheet. We logged into the virtual desktop and as a part of the first task in task ten we took a short Python script of code that checks for a name in the MariaDB database we made in "employees" database. This code checks for a name in the database and prints it if its there. If it isn't, an error message is printed. We were warned to be careful when copying and pasting the code into mousepad because there might be extra spaces or indents or a number of lines that have to be put as a single line of code. In the case something was wrong, an error

message would appear in the terminal. I copied the code to the end of the code then pasted it in. I then added the indentation and spaces needed or the ones that needed deletion, as well as any lines that needed rearranging. This way the code ran on the terminal.

Here's an example of the select.py code:

```
# This code is based on the examples on mariadb site. It assumes an
# employee_data table in employees database with a record for Bruce Lee
# user: tudublin password: tudublinpwd also need to be given
# authorisation to access the employee_data table.
# Look at the mariadb site for more info
#
# Module Imports
import mariadb
import sys


# Connect to MariaDB Platform
try:
        conn = mariadb.connect(
                user="tudublin",
                password="tudublinpwd",
                host="127.0.0.1",
                port=3306,
                database="employees"
)
except mariadb.Error as e:
        print(f"Error connecting to MariaDB Platform: {e}")
        sys.exit(1)


# Get Cursor
```

```
cur = conn.cursor()


# Retrieve information

some_name = "Bruce"

cur.execute("SELECT f_name,l_name FROM employee_data WHERE f_name=?",
(some_name,))


for f_name, l_name in cur:

        print(f"First name: {f_name}, Last name: {l_name}")


conn.close()
```

---

when this worked we then went on to the next task.

This was task eleven, in which we were given a code which instead of checking whether a name was in the database and printing it to show it was there or not, the code instead inserted an entry of a person and the details specified. This was handy if you wanted to add numerous details instantly to your database. The code was taken from the Python worksheet and adjusted so it would work like the last one. This showed the power of Python yet again where scripts could be written and tasks could be automated to purpose speeding things up, needing less work and being more efficient.

Here is the second code which inserts a name and their details in other columns called “insert.py”:

```
# Module Imports

import mariadb

import sys


# Connect to MariaDB Platform

try:

        conn = mariadb.connect(

                user="tudublin",

                password="tudublinpwd",
```

```
            host="127.0.0.1",
            port=3306,
            database="employees"
)
except mariadb.Error as e:
    print(f"Error connecting to MariaDB Platform: {e}")
    sys.exit(1)

# Get Cursor
cur = conn.cursor()

#insert information
try:
    cur.execute("INSERT INTO
employees(f_name,l_name,title,age,yos,salary,perks,email) VALUES (?, ?, ?, ?, ?,?, ?,
?)",("Clint","Eastwood","RepoAgent",64,4,25000,10000,"clint@homepc.com"))

except mariadb.Error as e:
    print(f"Error: {e}")

conn.commit()
print(f"Last Inserted ID: {cur.lastrowid}")

conn.close()
```

Then on to the next task in which simulated an “Arduino” client which would communicate with the Python server to request details, which was similar as before but a little more complex.

We skipped tasks twelve and thirteen and went on to fourteen which was as described as a “Network Socket Listener Python Application”. This when run would listen to a designated port until directed further. Then once communication came from the other

side of the port it would respond according to the system and MariaDB database previously made and used.

The code was copied and refined carefully to work as required like the last two codes. Once the code was running successfully it was run in a terminal window in the virtual machine and left. It said listening meaning it was awaiting communication from that port from another machine.

Another terminal window was opened and the Arduino machine was set up. As a substitute for the Arduino machine we used “ncat”, which was the simulated machine. We did this by installing “ncat” with the sudo command. Then running it, it interfaced with the other terminal window in which the “listener.py” was running and listening. It said then on the listener terminal window that it was connected and was waiting! Or waiting for input.

“Bronson,20000” was typed in on the “ncat” client terminal window and the details of him came up on the listener window, meaning the interface was working. It was showing the entry in the table because it matched the employee database and printed his details. If he wasn’t on it “something went wrong” would be printed as an error. This showed that it was a successful interface between the client and the server, or would be successful as interaction between an Arduino and server terminal in python.

Heres the code “listener.py”:

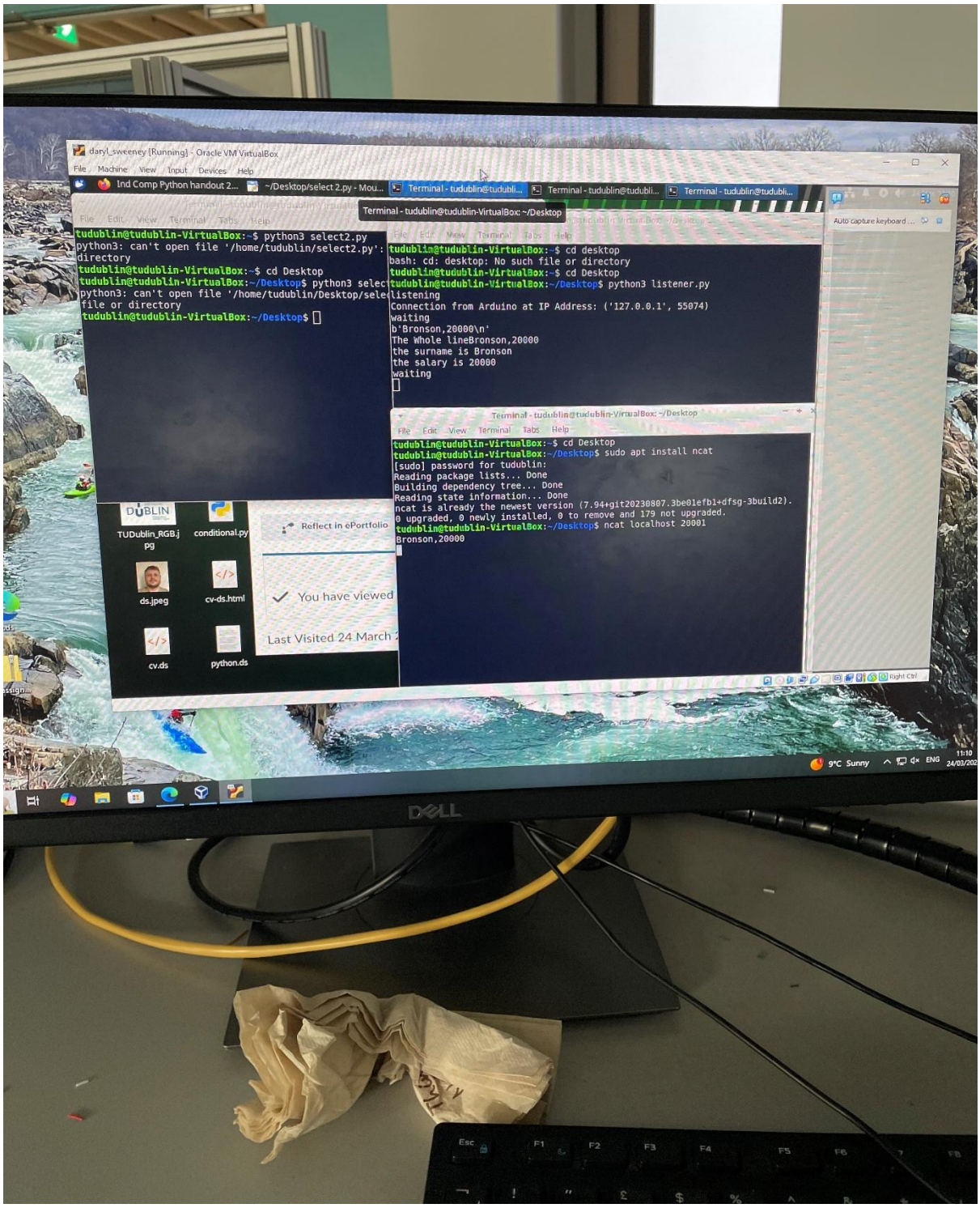
```
#!/usr/bin/python

# Employees Listener program

import mariadb

import socket

import string

conn = mariadb.connect( user="tudublin", password="tudublinpwd",

host="localhost", database="employees")

cur = conn.cursor()


HOST = ''    #all available interfaces

PORT = 20001 #unprivileged ports are >20000


s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
s.bind((HOST,PORT))
s.listen(1)
print ('listening')
sockconn, addr = s.accept()

print('Connection from Arduino at IP Address:', addr)

while 1:
        print('waiting')
        data=sockconn.recv(1024)
        if not data:
        break
        print(data)
        for line in data.splitlines():
                if len(line)>0:
                        l=line.decode('utf-8')
                        term=l.split(",")

                        print("The Whole line" + l)

                        if len(l)<20:
                                print("the surname is " + str(term[0]))
                                print("the salary is " + str(term[1]))
                                lastname=term[0]
                                sal=int(term[1]) #insert information
                                try:
                                        cur.execute("INSERT INTO employee_data (l_name,
salary) VALUES (?, ?)", (lastname,sal))
```

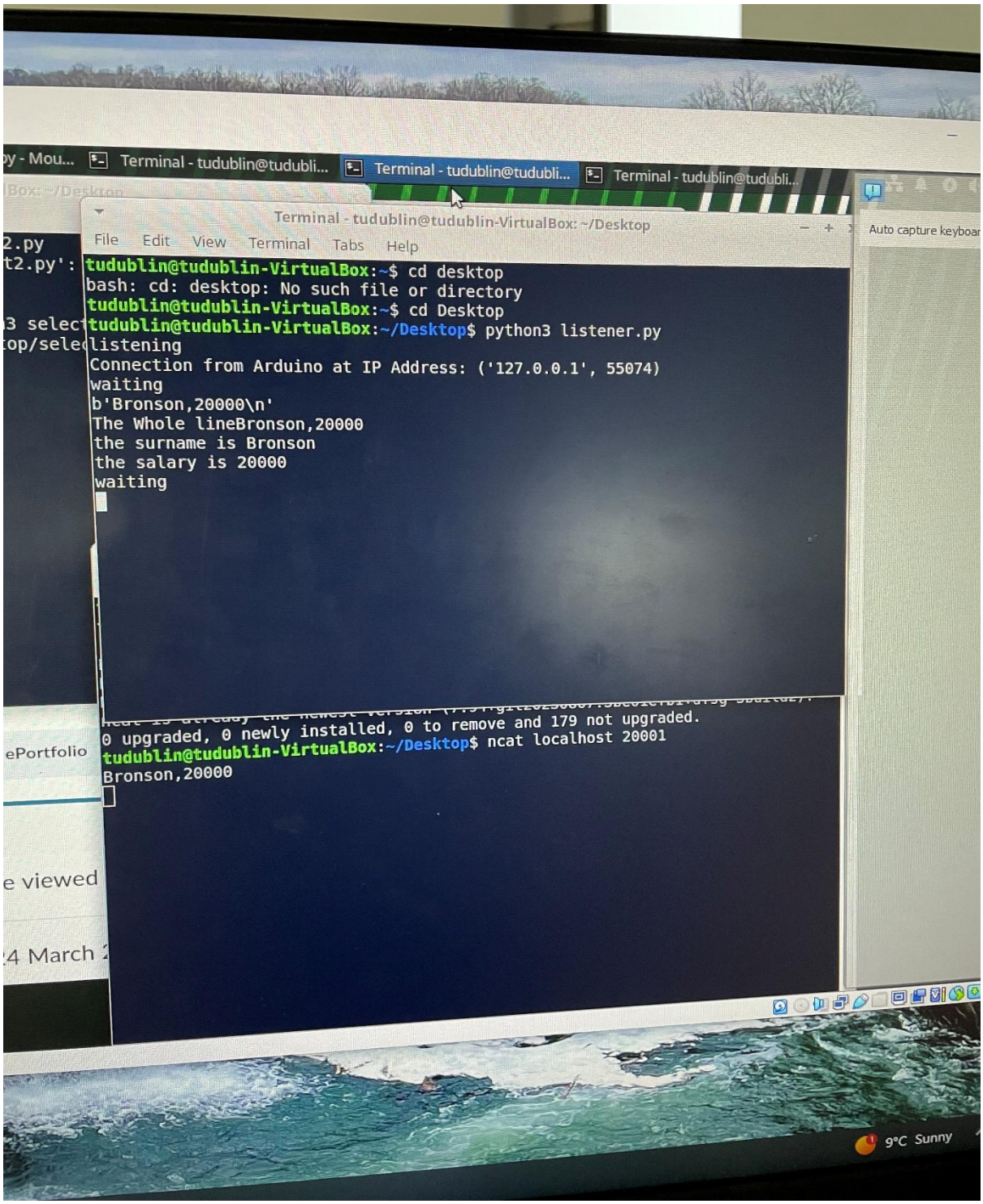
```
                conn.commit()
            except mariadb.Error as e:
                print(f"Error: {e}")
                sockconn.close()
                conn.close()
            print(f"Last Inserted ID:{cur.lastrowid}")
    else:
            print("something went wrong" + l)
            sockconn.close()
            conn.close()
```

---

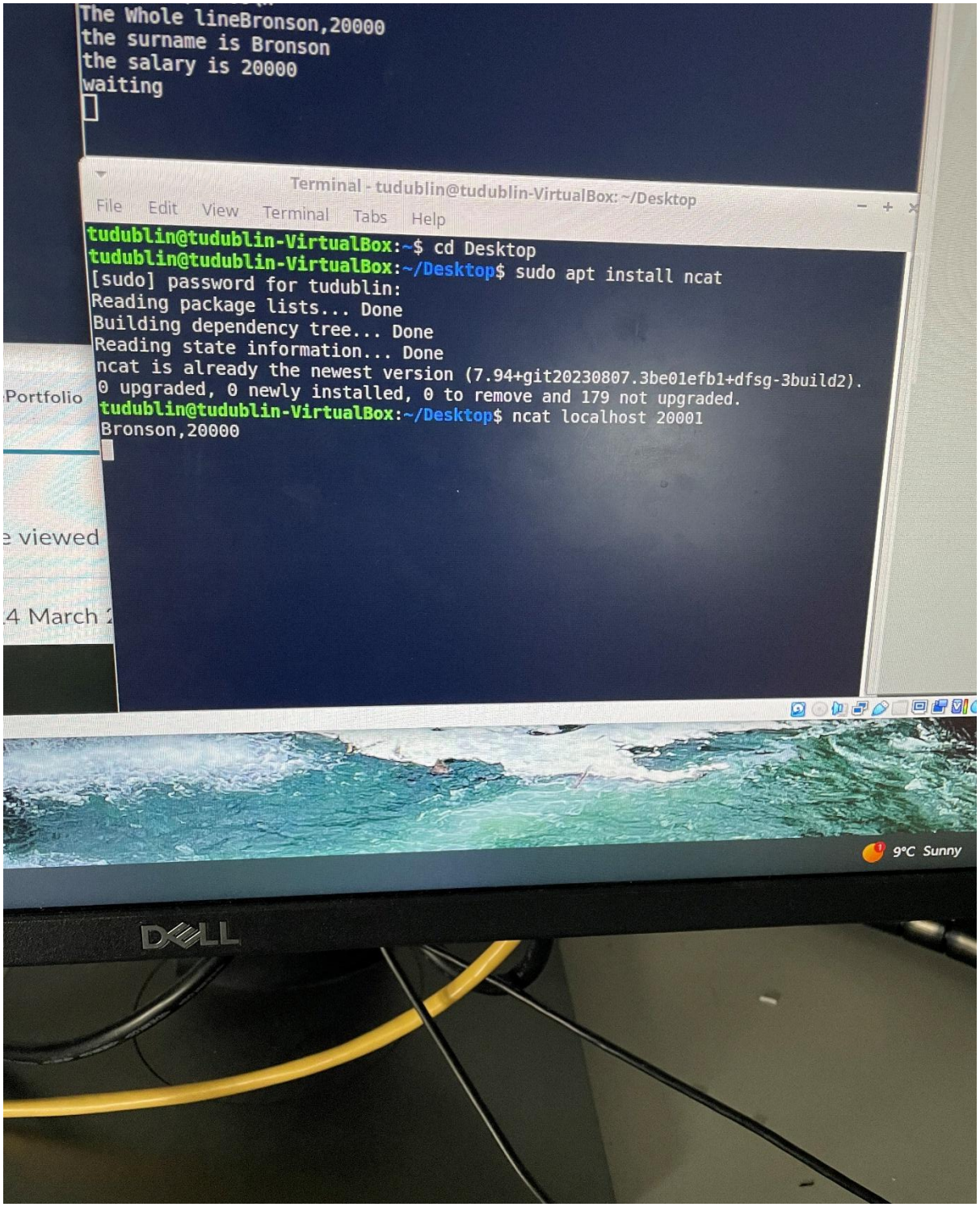
heres also pics of it working:



Heres the terminal windows showing it working:



Heres a closer shot:



Heres a pic of the client:

Conclusion:

We were shown what could be done with Python and a mariadb database. If this could be done there would be allot more potential than this. The interface with the other machine posing as an Arduino was very interesting. We were told of our final project which would incorporate all we've learned from this and more. Using the instrumentation module and making a database and interfacing with an Arduino in practice for the weather station. Damon said it would be very beneficial for understanding in use in a job.

Appendix Lab 7:

Laboratory Log: 31/03/25

Using the virtual machine we installed "Grafana" into our virtual machine using the terminal and here's what was done in installing it:

---

tudublin@tudublin-VirtualBox:~$ mariadb -u root -p

Enter password:

ERROR 1698 (28000): Access denied for user 'root'@'localhost'

tudublin@tudublin-VirtualBox:~$ sudo apt-get install -y apt-transport-https software-properties-common wget

[sudo] password for tudublin:

Reading package lists... Done

Building dependency tree... Done

Reading state information... Done

wget is already the newest version (1.21.4-1ubuntu4.1).

wget set to manually installed.

The following additional packages will be installed:

python3-software-properties software-properties-gtk

The following NEW packages will be installed:

apt-transport-https

The following packages will be upgraded:

python3-software-properties software-properties-common

software-properties-gtk

3 upgraded, 1 newly installed, 0 to remove and 176 not upgraded.

Need to get 131 kB of archives.

After this operation, 35.8 kB of additional disk space will be used.

Get:1 http://ie.archive.ubuntu.com/ubuntu noble/universe amd64 apt-transport-https all 2.7.14build2 [3,974 B]

Get:2 http://ie.archive.ubuntu.com/ubuntu noble-updates/main amd64 software-properties-common all 0.99.49.1 [14.4 kB]

Get:3 http://ie.archive.ubuntu.com/ubuntu noble-updates/main amd64 software-properties-gtk all 0.99.49.1 [83.1 kB]

Get:4 http://ie.archive.ubuntu.com/ubuntu noble-updates/main amd64 python3-software-properties all 0.99.49.1 [29.7 kB]

Fetched 131 kB in 1s (247 kB/s)

Selecting previously unselected package apt-transport-https.

(Reading database ... 220379 files and directories currently installed.)

Preparing to unpack .../apt-transport-https_2.7.14build2_all.deb ...

Unpacking apt-transport-https (2.7.14build2) ...

Preparing to unpack .../software-properties-common_0.99.49.1_all.deb ...

Unpacking software-properties-common (0.99.49.1) over (0.99.48) ...

Preparing to unpack .../software-properties-gtk_0.99.49.1_all.deb ...

Unpacking software-properties-gtk (0.99.49.1) over (0.99.48) ...

Preparing to unpack .../python3-software-properties_0.99.49.1_all.deb ...

Unpacking python3-software-properties (0.99.49.1) over (0.99.48) ...

Setting up apt-transport-https (2.7.14build2) ...

Setting up python3-software-properties (0.99.49.1) ...

Setting up software-properties-common (0.99.49.1) ...

Setting up software-properties-gtk (0.99.49.1) ...

Processing triggers for libglib2.0-0t64:amd64 (2.80.0-6ubuntu3.2) ...

Processing triggers for dbus (1.14.10-4ubuntu4.1) ...

Processing triggers for shared-mime-info (2.4-4) ...

Processing triggers for desktop-file-utils (0.27-2build1) ...

Processing triggers for hicolor-icon-theme (0.17-2) ...

Processing triggers for gnome-menus (3.36.0-1.1ubuntu3) ...

Processing triggers for man-db (2.12.0-4build2) ...

tudublin@tudublin-VirtualBox:~$ sudo kdir -p /etc/apt/keyrings/

sudo: kdir: command not found

tudublin@tudublin-VirtualBox:~$ sudo mkdir -p /etc/apt/keyrings/

tudublin@tudublin-VirtualBox:~$ wget -q -0 - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null

wget: invalid option -- '0'

Usage: wget [OPTION]... [URL]...

Try `wget --help' for more options.

gpg: no valid OpenPGP data found.

tudublin@tudublin-VirtualBox:~$ wget -q -o - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null

gpg: no valid OpenPGP data found.

tudublin@tudublin-VirtualBox:~$ wget -q -0 - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null

wget: invalid option -- '0'

Usage: wget [OPTION]... [URL]...

Try `wget --help' for more options.

gpg: no valid OpenPGP data found.

tudublin@tudublin-VirtualBox:~$ wget -q -o - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null

gpg: no valid OpenPGP data found.

tudublin@tudublin-VirtualBox:~$ wget -q -O - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null

tudublin@tudublin-VirtualBox:~$ echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main" | sudo tee -a /etc/apt/sources.list.d/grafanna.list

deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main

tudublin@tudublin-VirtualBox:~$ sudo apt-get update

Get:1 https://apt.grafana.com stable InRelease [7,661 B]

Get:2 https://apt.grafana.com stable/main amd64 Packages [374 kB]

Get:3 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]

Hit:4 http://ie.archive.ubuntu.com/ubuntu noble InRelease

Get:5 http://ie.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]

Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [711 kB]

Get:7 http://ie.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]

Get:8 http://ie.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [960 kB]

Get:9 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [136 kB]

Get:10 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8,956 B]

Get:11 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [7,068 B]

Get:12 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [810 kB]

Get:13 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [164 kB]

Get:14 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [208 B]

Get:15 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [468 B]

Get:16 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [822 kB]

Get:17 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [177 kB]

Get:18 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.0 kB]

Get:19 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [17.0 kB]

Get:20 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [17.6 kB]

Get:21 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [3,792 B]

Get:22 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]

Get:23 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [380 B]

Get:24 http://ie.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [213 kB]

Get:25 http://ie.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [151 kB]

Get:26 http://ie.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [13.5 kB]

Get:27 http://ie.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [842 kB]

Get:28 http://ie.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [170 kB]

Get:29 http://ie.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]

Get:30 http://ie.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [492 B]

Get:31 http://ie.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1,044 kB]

Get:32 http://ie.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [263 kB]

Get:33 http://ie.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [365 kB]

Get:34 http://ie.archive.ubuntu.com/ubuntu noble-updates/universe Icons (48x48) [222 kB]

Get:35 http://ie.archive.ubuntu.com/ubuntu noble-updates/universe Icons (64x64) [344 kB]

Get:36 http://ie.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [26.0 kB]

Get:37 http://ie.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [21.5 kB]

Get:38 http://ie.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [4,788 B]

Get:39 http://ie.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]

Get:40 http://ie.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [592 B]

Get:41 http://ie.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [7,092 B]

Get:42 http://ie.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [212 B]

Get:43 http://ie.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [26.4 kB]

Get:44 http://ie.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [15.7 kB]

Get:45 http://ie.archive.ubuntu.com/ubuntu noble-backports/universe Icons (48x48) [17.4 kB]

Get:46 http://ie.archive.ubuntu.com/ubuntu noble-backports/universe Icons (64x64) [24.3 kB]

Get:47 http://ie.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1,304 B]

Get:48 http://ie.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]

Fetched 8,422 kB in 2s (4,387 kB/s)

Reading package lists... Done

tudublin@tudublin-VirtualBox:~$ sudo apt-get install -y grafana

Reading package lists... Done

Building dependency tree... Done

Reading state information... Done

The following additional packages will be installed:

musl

The following NEW packages will be installed:

grafana musl

0 upgraded, 2 newly installed, 0 to remove and 191 not upgraded.

Need to get 169 MB of archives.

After this operation, 631 MB of additional disk space will be used.

Get:1 http://ie.archive.ubuntu.com/ubuntu noble/universe amd64 musl amd64 1.2.4-2 [416 kB]

Get:2 https://apt.grafana.com stable/main amd64 grafana amd64 11.6.0 [169 MB]

Fetched 169 MB in 7s (25.6 MB/s)

Selecting previously unselected package musl:amd64.

(Reading database ... 220383 files and directories currently installed.)

Preparing to unpack .../musl_1.2.4-2_amd64.deb ...

Unpacking musl:amd64 (1.2.4-2) ...

Selecting previously unselected package grafana.

Preparing to unpack .../grafana_11.6.0_amd64.deb ...

Unpacking grafana (11.6.0) ...

Setting up musl:amd64 (1.2.4-2) ...

Setting up grafana (11.6.0) ...

info: Selecting UID from range 100 to 999 ...

info: Adding system user `grafana' (UID 121) ...

info: Adding new user `grafana' (UID 121) with group `grafana' ...

info: Not creating home directory `/usr/share/grafana'.

NOT starting on installation, please execute the following statements to configure grafana to start automatically using systemd

sudo /bin/systemctl daemon-reload

sudo /bin/systemctl enable grafana-server

You can start grafana-server by executing

sudo /bin/systemctl start grafana-server

Processing triggers for man-db (2.12.0-4build2) ...

tudublin@tudublin-VirtualBox:~$ sudo /bin/systemctl enable grafana-server

Synchronizing state of grafana-server.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.

Executing: /usr/lib/systemd/systemd-sysv-install enable grafana-server

Created symlink /etc/systemd/system/multi-user.target.wants/grafana-server.service → /usr/lib/systemd/system/grafana-server.service.

tudublin@tudublin-VirtualBox:~$ sudo /bin/systemctl start grafana-server

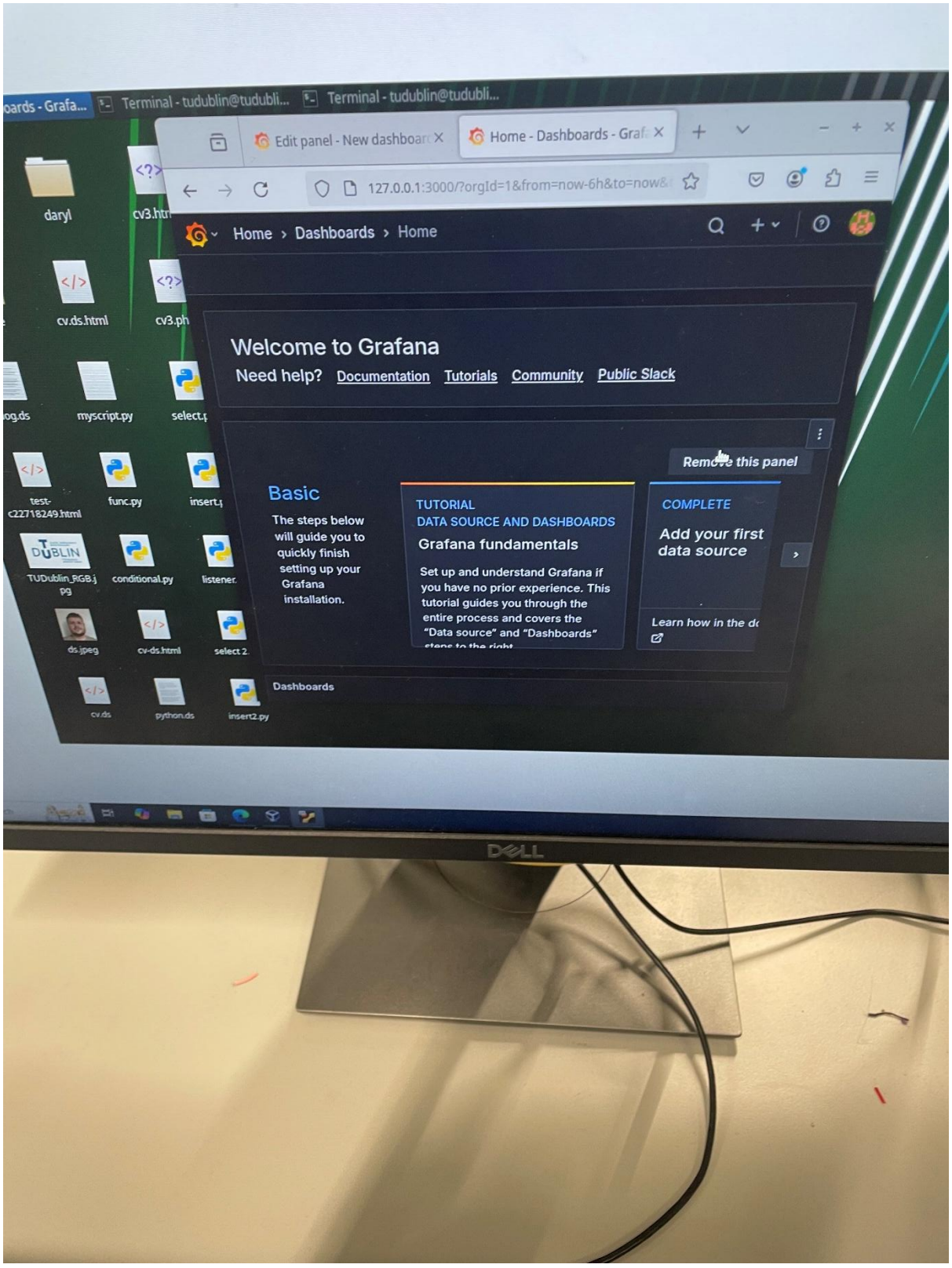
tudublin@tudublin-VirtualBox:~$ ^C

tudublin@tudublin-VirtualBox:~$

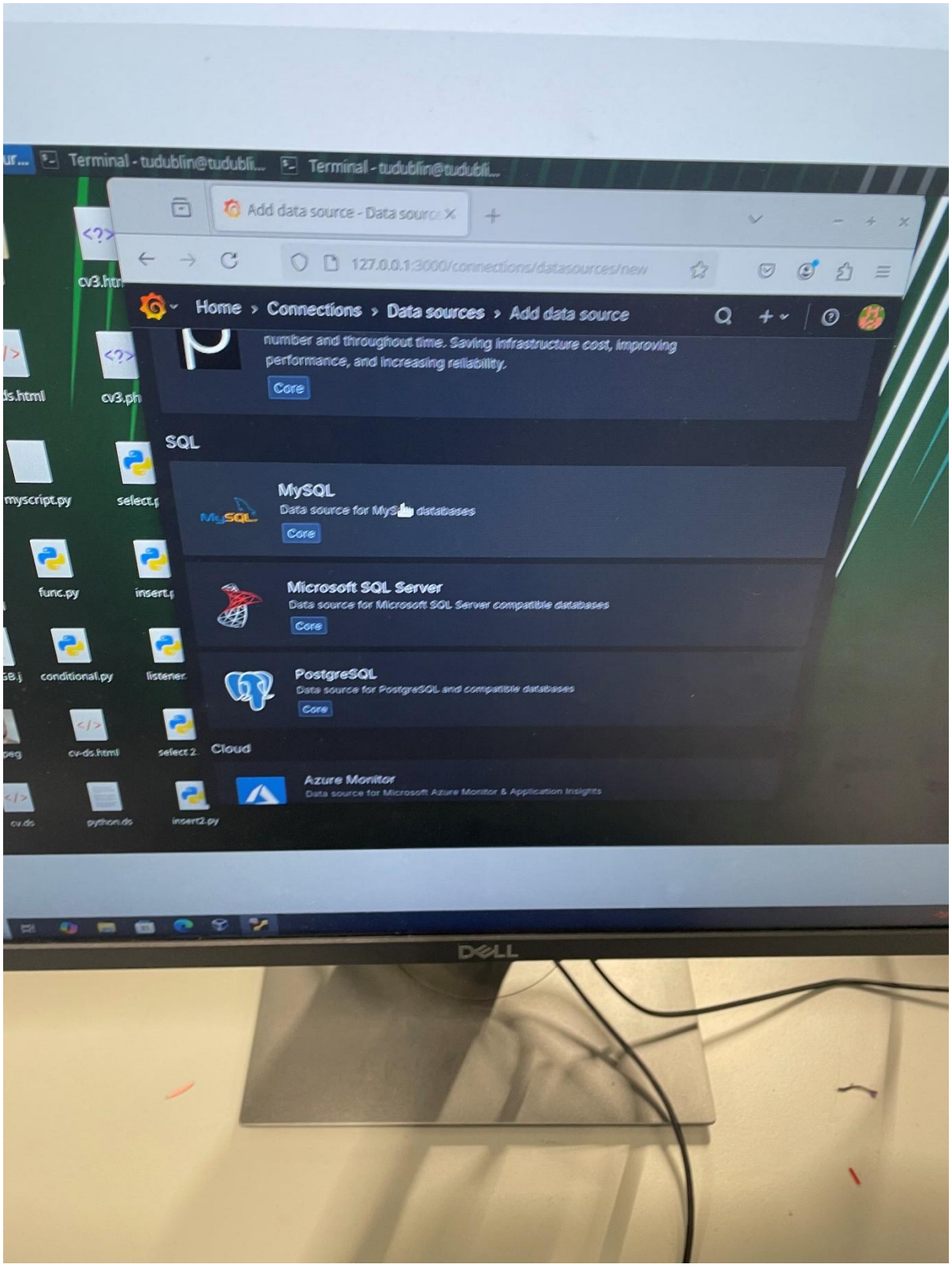
We followed the instructions set out by Damon and after restructuring commands, it processed and all came together. “Grafana” was installed then. I opened the browser then and typed “127.0.0.1:3000/” in the address bar and Grafana loaded.

It came to a login page. We logged in “admin” and password “admin” then the homepage loaded.

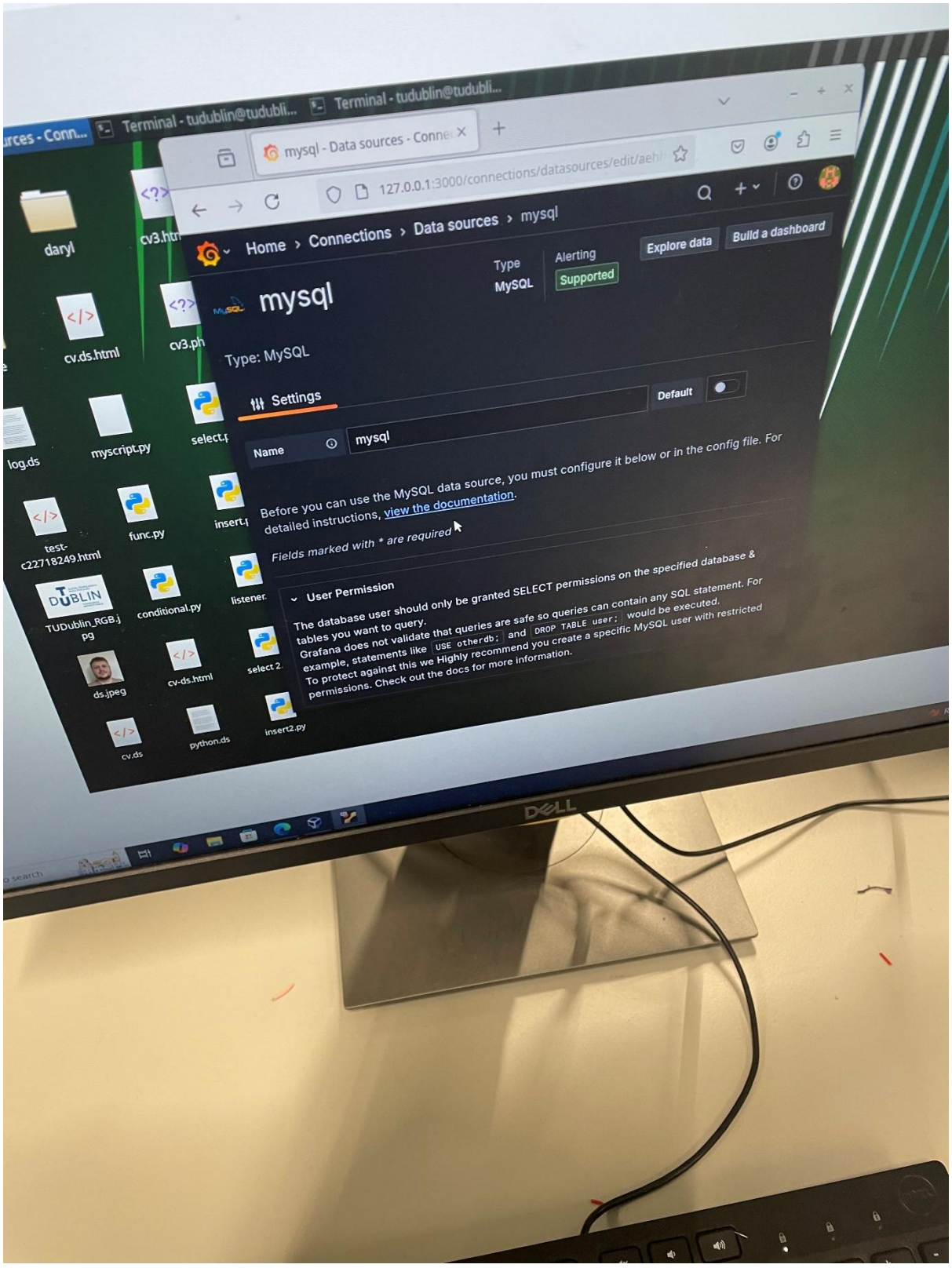
In the Grafana homepage we scrolled to complete then clicked on it. Then clicked on “MySQL”, this then had details to fill out. We filled in “localhost:3300” which was the port available for MariaDB. The “employees” database was used and the username and password tudublin and tudublinpwd, it then said save and try. The employees database was selected then the employee_data table was selected. It was now connected, and the menu screen was able to interface greatly with the database. You could preview info from your database and table and graph data in lots of different ways as it came in in real time. The tool was extremely useful as when we could use this while interfacing with the weather station in the real time input into the tables in the database.



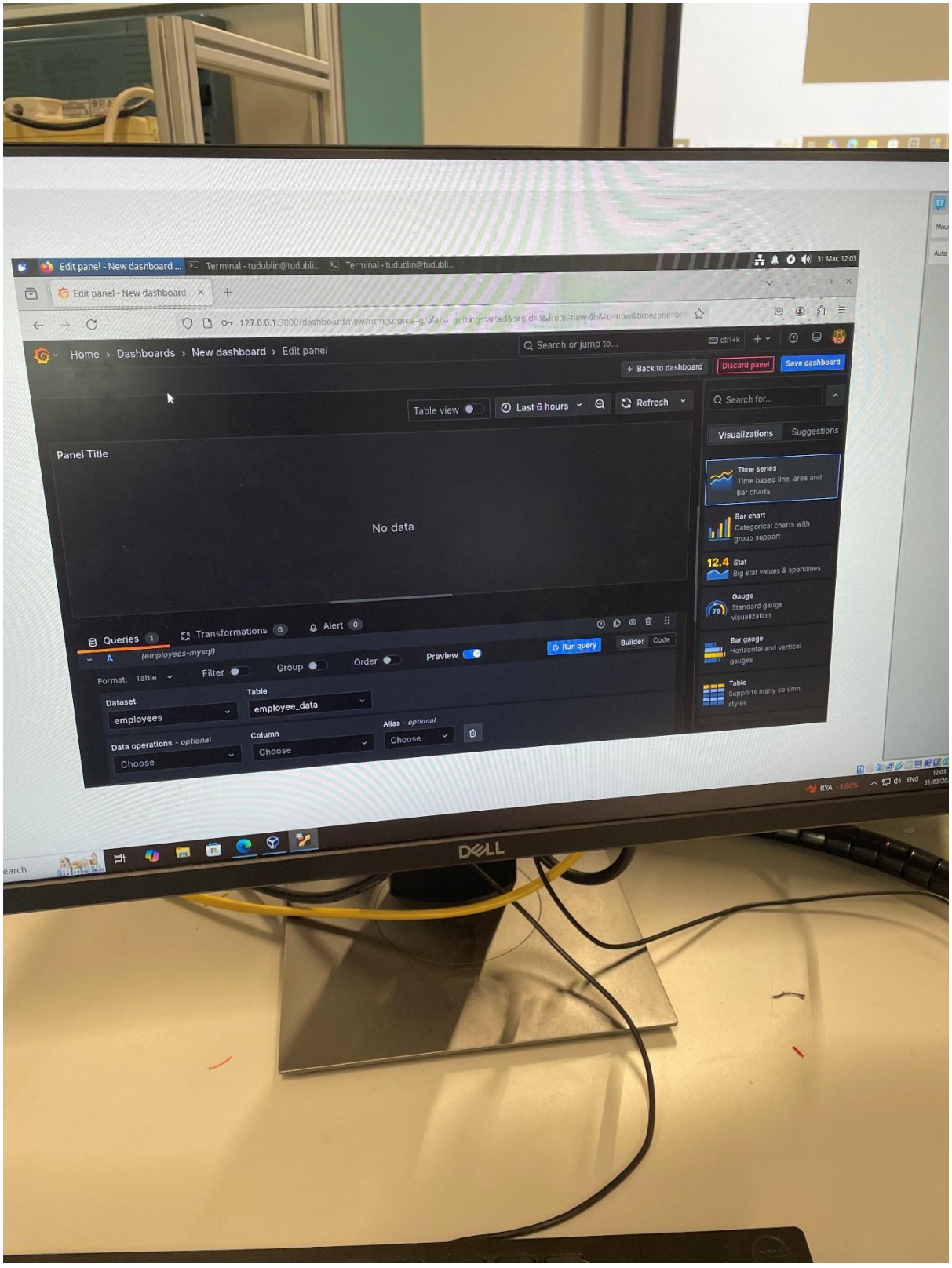
In the grafana homepage and clicking complete to get to “mysql”:



Here can be seen going into the next part – mysql:



Showing here the next step:



Showing here using Grafana logged into the employees database and employee-data table. As you can see there is numerous options to express, you're information with for presentation and interpretation reasons. Also, numerous different tables and databases can be displayed and toggled between another

Conclusion:

This lab was the introduction to Grafana, and it was open source meaning adaptable to tasks when needed. In the instrumentation module I had little experience in python and came up with a script that would write the details in a notepad file, and then we would be able to graph it and look back on our results. This was of more a personal databasing reasons. With Grafana you could connect to any server with multiple devices adding data continuously to a database or databases with a few tables in which the data could be interpreted and presented to give statistical data the ability to paint a bigger picture. This was through a relational database and gave us experience if we were involved with bigger projects, if we were to get a job in the area.

Appendix Lab 8:

Laboratory Log: 07/04/25

Today we were to use our time to work on the main assignment for the laboratory. Today I made a “weather” database with a “TEMP” or temperature and windspeed or “WINDSP”.

Creating TEMP headings:

```
mariadb> CREATE DATABASE WEATHER; (and hit enter)
mariadb> USE WEATHER; (and hit enter)
mariadb> CREATE TABLE TEMP
-> (
-> OBS_ID int unsigned not null auto_increment primary key,
-> VAL float),
-> LAT float,
-> LONGT float,
-> UNIT VARCHAR(25),
-> DATE DATE,
-> TIME TIME,
-> SENS_ID VARCHAR(35),
-> NOTE VARCHAR(60)
->);
```

possibly creating WINDSP table:

```
mariadb> CREATE DATABASE WEATHER; (and hit enter)
mariadb> USE WEATHER; (and hit enter)
mariadb> CREATE TABLE WINDPD
-> (
-> ID INT AUTO_INCREMENT PRIMARY KEY,
-> STATION INT NOT NULL,
-> RECORDED DATETIME NOT NULL,
-> WND_SPD_MPS INT,
-> WND_SPD_KPH INT,
-> latitude FLOAT,
-> LONGT FLOAT,
-> NOTE VARCHAR(255)
->);
```

I was using a python script to automatically insert entries into the table TEMP and haven’t started on the second one.

Heres a script to insert many entries into this table:

```
# Module Imports
import mariadb
import sys

# Connect to MariaDB Platform
try:
    conn = mariadb.connect(
        user="tudublin",
        password="tudublinpwd",
        host="127.0.0.1",
        port=3306,
        database="WEATHER"
    )
except mariadb.Error as e:
    print(f"Error connecting to MariaDB Platform: {e}")
    sys.exit(1)

# Get Cursor
cur = conn.cursor()

# Entry list
entries = [
    (12.12, 97.0087, 7.8907, 'm/s', '2024-03-03', '22:45:01', '25rt', 'the first'),
    (14.55, 96.0000, 8.9000, 'kph', '2024-03-03', '22:50:00', '25rt', 'second entry'),
    (10.25, 98.1234, 7.6543, 'm/s', '2024-03-03', '22:55:00', '25rt', 'third reading')
```

```
]

# Insert multiple entries
try:
    cur.executemany(
        "INSERT INTO TEMP (VAL, LAT, LONGT, UNIT, DATE, TIME, SENS_ID, NOTE) "
        "VALUES (?, ?, ?, ?, ?, ?, ?, ?)",
        entries
    )
except mariadb.Error as e:
    print(f"Error: {e}")
else:
    conn.commit()
    print("Entries inserted successfully.")
    print(f"Last Inserted ID: {cur.lastrowid}")

conn.close()
```

heres a script to get two values from the table:

```
#!/usr/bin/python
# Employees Listener program

import mariadb
import socket
import string

# Connect to mariadb
conn = mariadb.connect(
    user="tudublin",
```

```
    password="tudublinpwd",
    host="localhost",
    database="WEATHER"
)
cur = conn.cursor()

HOST = ''  # all available interfaces
PORT = 20001  # unprivileged ports are >20000

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen(1)
print('listening')
sockconn, addr = s.accept()

print('Connection from Arduino at IP Address:', addr)

while 1:
    print('waiting')
    data = sockconn.recv(1024)
    if not data:
        break
    print(data)
    for line in data.splitlines():
        if len(line) > 0:
            l = line.decode('utf-8')
            term = l.split(",")
```

```
print("The Whole line" + l)


if len(term) == 2:
    print("the Temperature is " + str(term[0]))
    print("the time is " + str(term[1]))
    try:
        value = float(term[0])  # convert to float for VAL
        time_str = term[1].strip()  # keep time as string
        cur.execute("INSERT INTO TEMP (VAL, TIME) VALUES (?, ?)", (value, time_str))
        conn.commit()
        print(f"Last Inserted ID: {cur.lastrowid}")
    except mariadb.Error as e:
        print(f"Error: {e}")
        sockconn.close()
        conn.close()
    except ValueError as ve:
        print(f"Invalid value: {ve}")
else:
    print("something went wrong" + l)
    sockconn.close()
    conn.close()
```