|  |  |
| --- | --- |
|  | [Warszawa, dnia 31 sierpnia 2020r.] |
| [AB/XYZ/000/2020] |  |

Pan

Sebastian Buczek

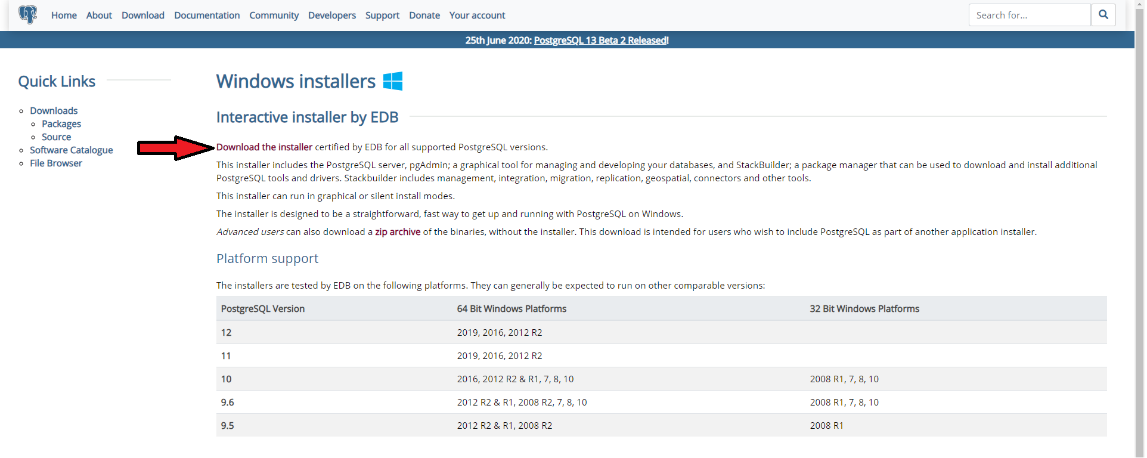
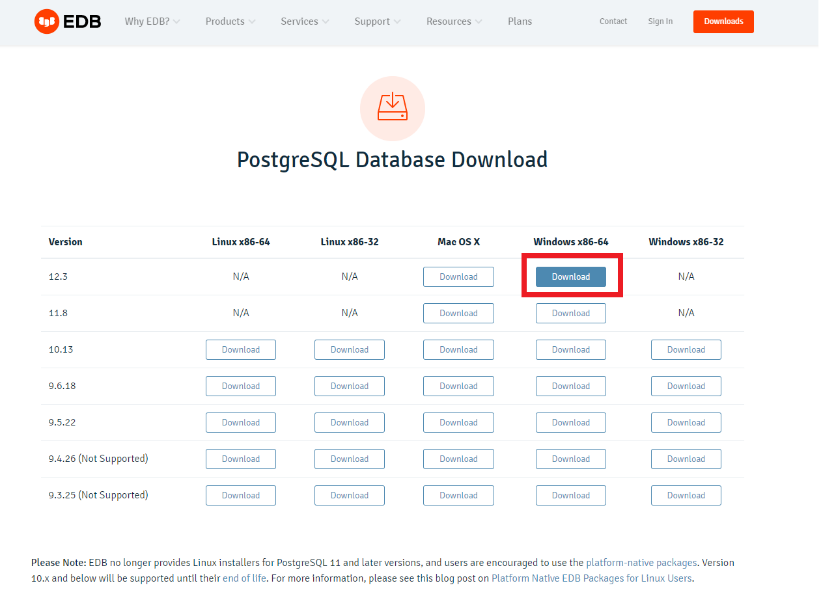
Stażysta

**System umożliwiający zautomatyzowany zapis i przetwarzanie danych o temperaturze**

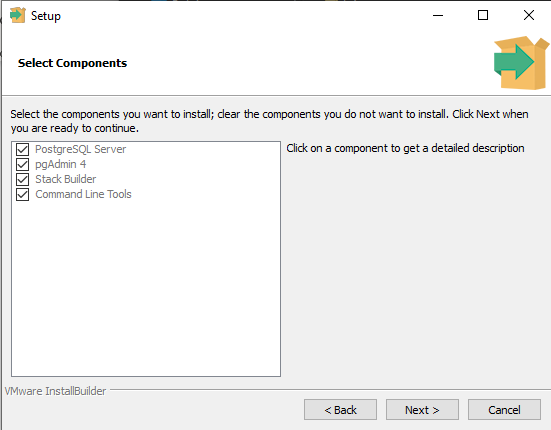
[dr Imię Nazwisko]

[Kierownik]

**Spis treści**

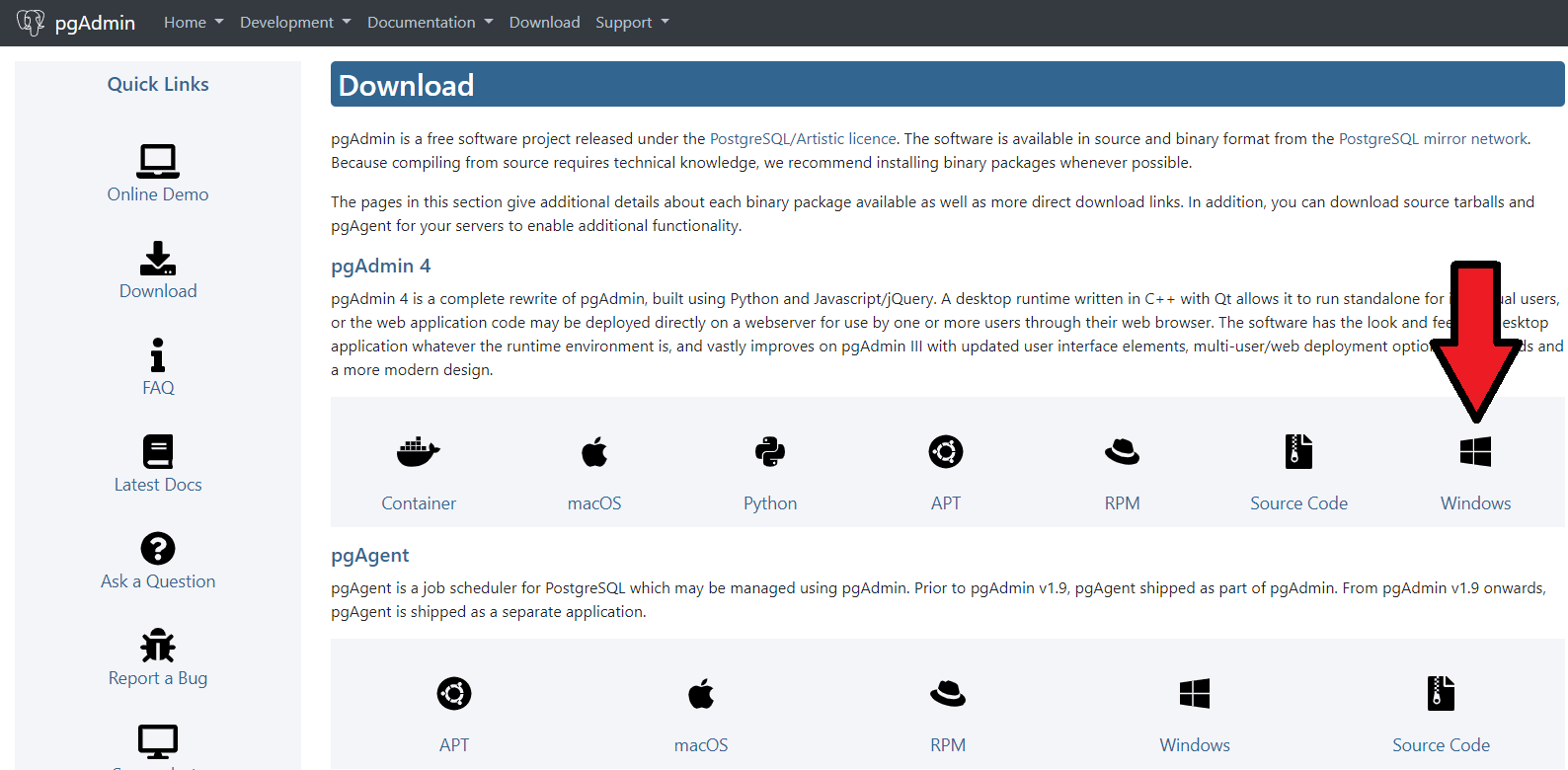
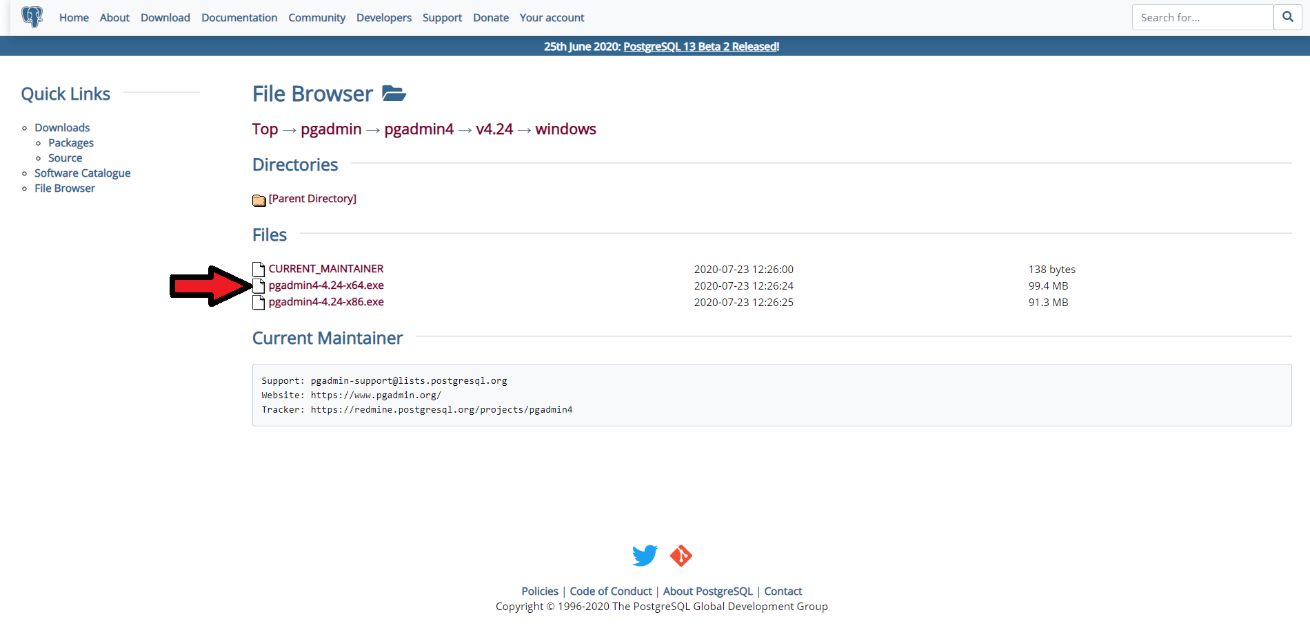
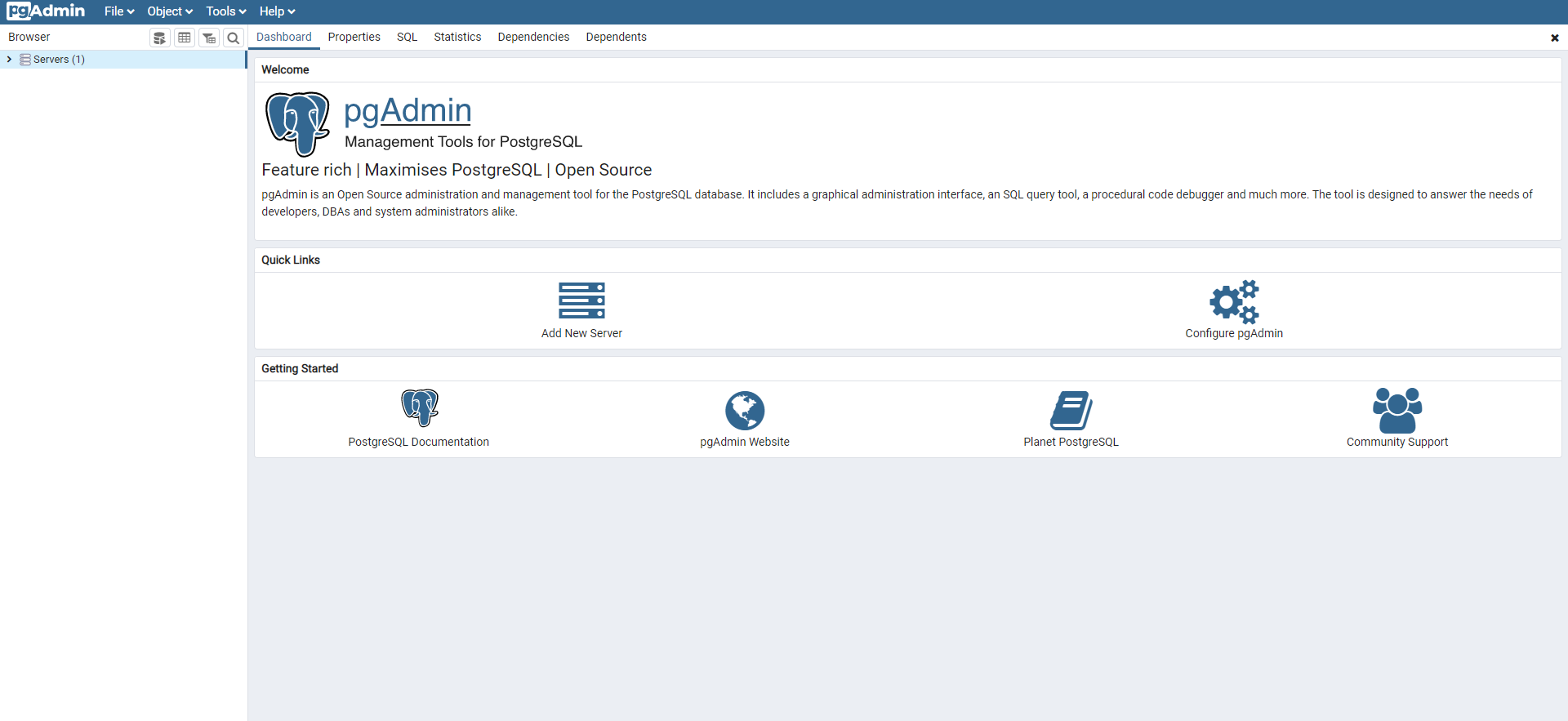
1. **Instalacja oprogramowania PostgreSQL…………………………………………………………………..3**
2. **Instalacja oprogramowania pgAdmin……………………………………………………………………….4**
3. **Utworzenie bazy danych………………………………………………………………………………………….6**
4. **Python – konfiguracja………………………………………………………………………………………………9**
5. **Dodanie konta użytkownika do bazy……………………………………………………………………….10**
6. **Instrukcja dla użytkownika……………………………………………………………………………………...13**
7. **Utworzenie planu projektu z użyciem programu Project in the box…………………………15**
8. **Rysowanie wykresów przez skrypt python………………………………………………………………18**
9. **Komunikacja UDP……………………………………………………………………………………………………31**
10. **Usprawnienia, utworzenie plików exe…………………………………………………………………….32**
11. **Instalacja oprogramowania PostgreSQL** 
    1. Link do strony: <https://www.postgresql.org/download/>
    2. Wybieram system operacyjny (w moim przypadku windows)
    3. Wybieram:
    4. Zostaję przekierowany na stronę <https://www.enterprisedb.com/downloads/postgres-postgresql-downloads>. Wybieram swój system oraz najnowszą wersję programu:
    5. Po pobraniu uruchamiam instalator:

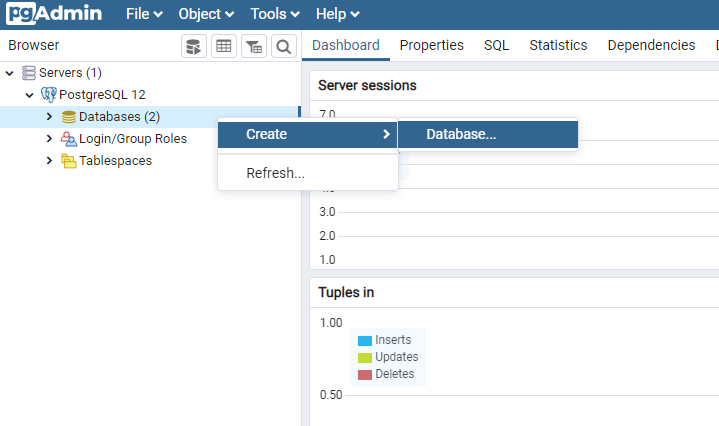


* 1. Instaluję z domyślnymi ustawieniami oraz :

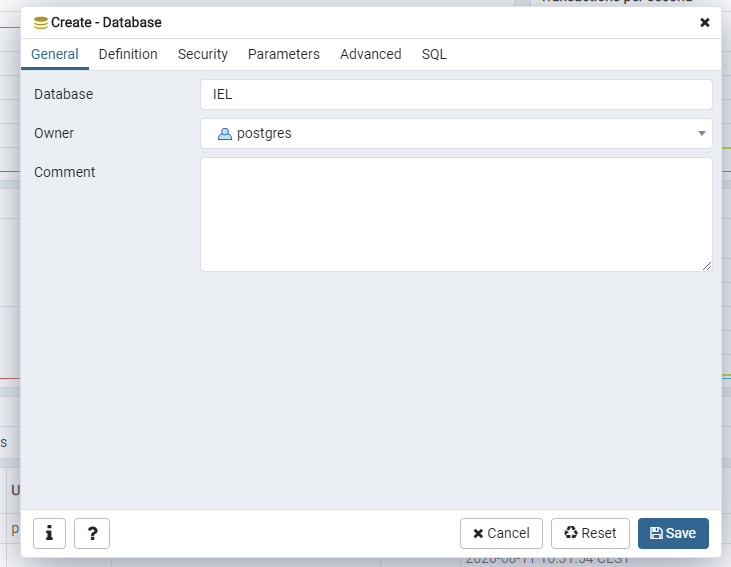
**(pgAdmin 4 można odznaczyć-zostawić pusty kwadrat, zainstaluję ten program oddzielnie)**

\*Warto zapamiętać hasło i port

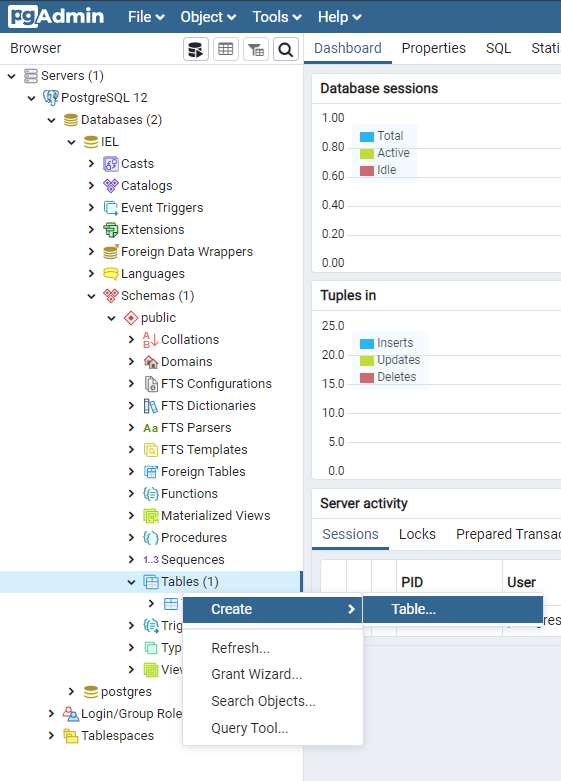
1. **Instaluję pgAdmin – program z graficznym interfejsem wspomagający administrację bazą PostgreSQL.**
   1. Link do strony: <https://www.pgadmin.org/download/>
   2. Wybieram system operacyjny: 
   3. Wybieram najnowszą wersję: 
   4. Wybieram plik exe dla mojego systemu: 
   5. Instaluję z domyślnymi ustawieniami i uruchamiam. Wygląd po uruchomieniu: ****
2. **Utworzenie bazy danych**



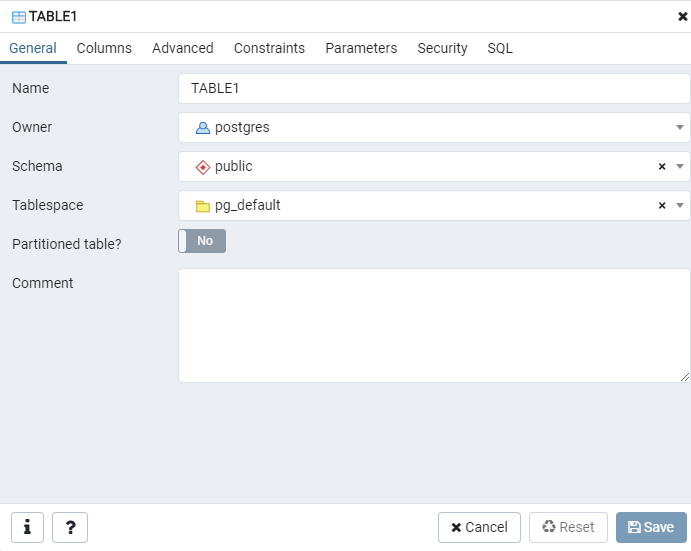
3.2 Nadaję nazwę oraz właściela bazy. Reszta ustawień domyślna.

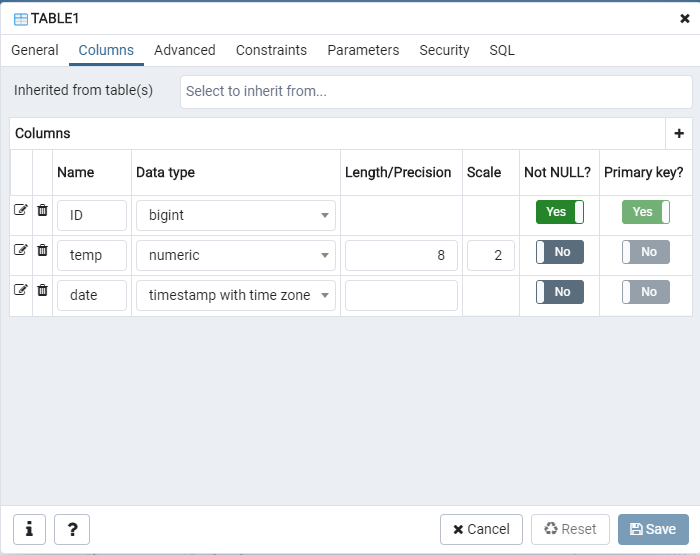
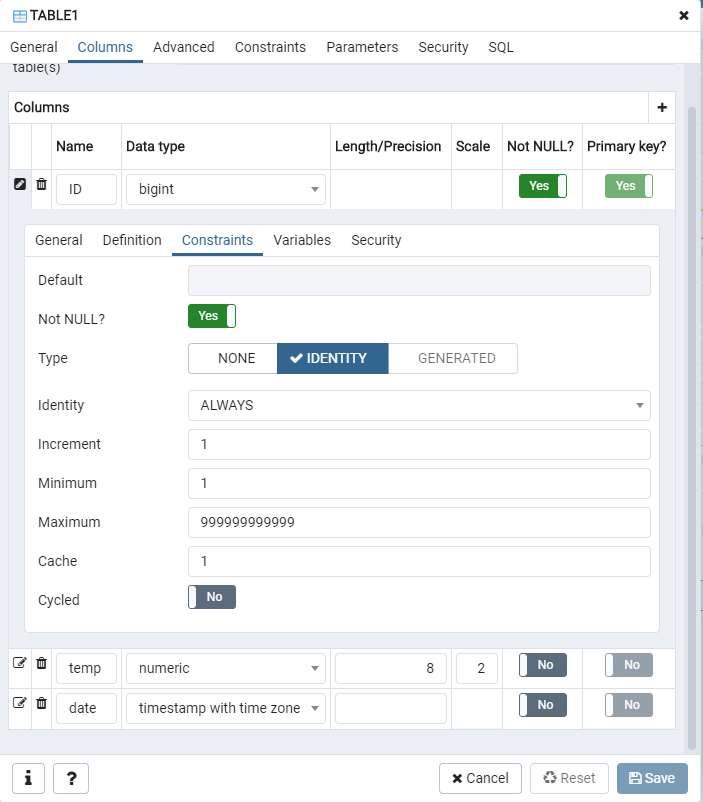
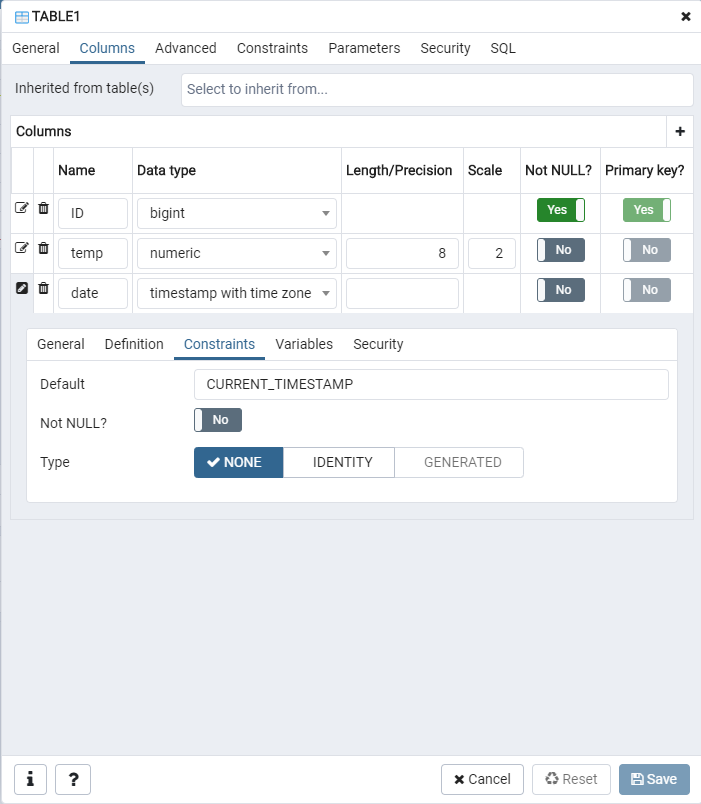


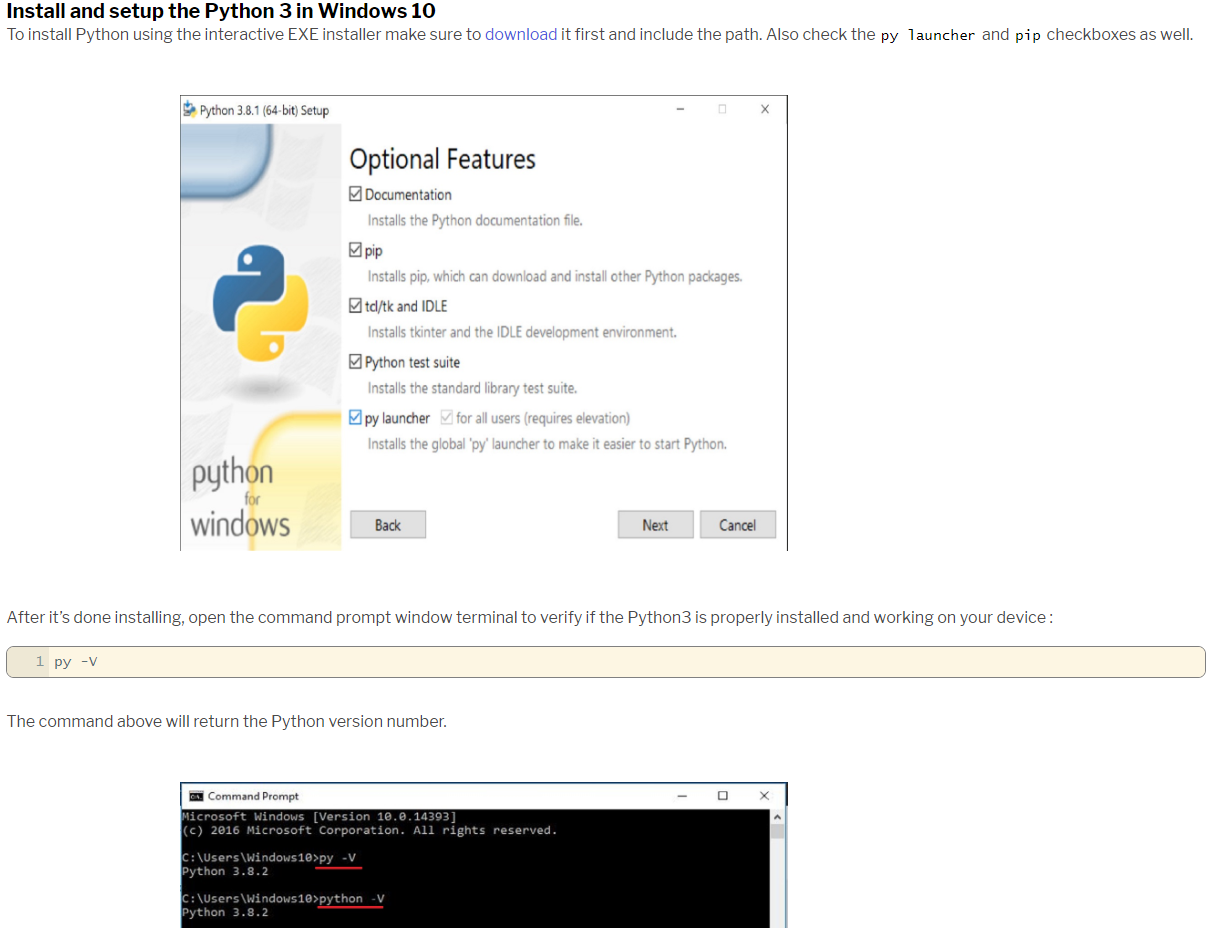
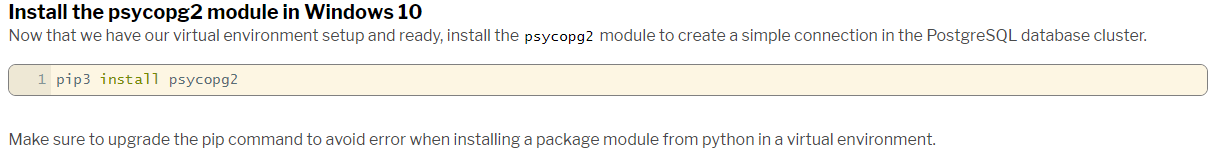
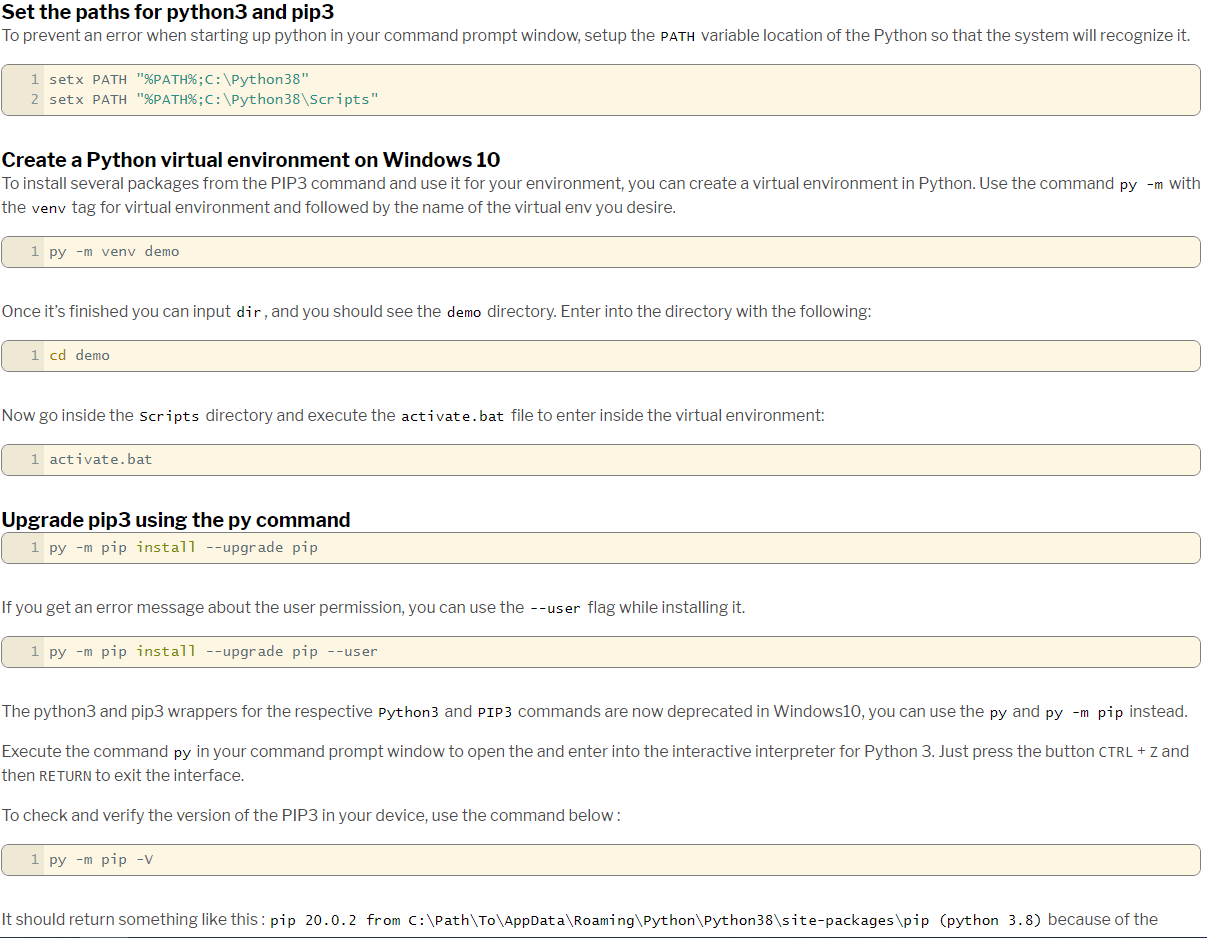
* 1. Tworzę tabelę:



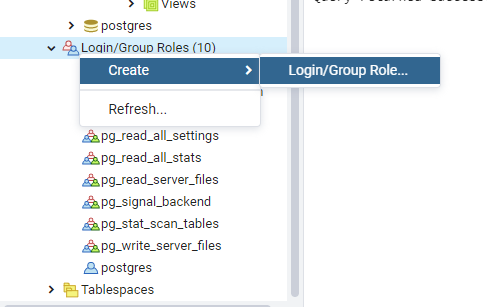
* 1. Nadaję nazwę tabeli:

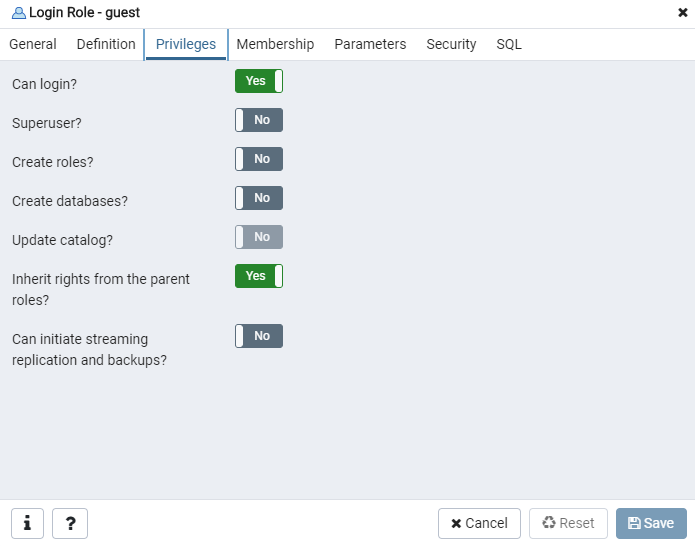


* 1. Dodaję kolumny 
  2. Ustawiam auto inkrementację w kolumnie ID (przycisk edycji): 
  3. Ustawiam automatyczne wpisywanie aktualnej daty przy dodawaniu wpisu do tabeli: 

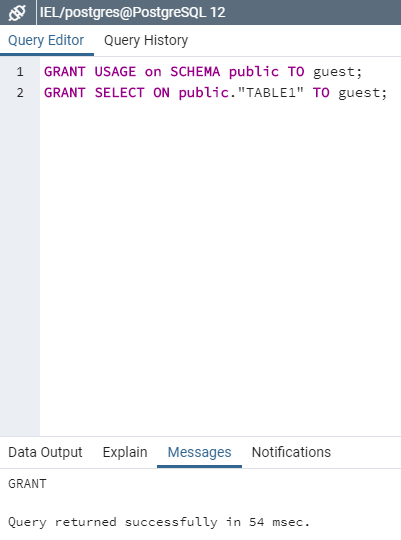
1. **Python - konfiguracja**
   1. Należy zainstalować python 3 oraz psycopg2 (wtyczka do PostgreSQL) na komputerze <https://www.python.org/downloads/release>
   2. Następnie według zrzutu ekranu: 
   3. Następnie: 
2. **Python – skrypt**
3. **Dodanie konta użytkownika do bazy**

6.1



6.2 Nadaję nazwę guest oraz przydzielam następujące uprawnienia: 

6.3 Aby user mógł odczytywać dane z tabeli (używać komendy SELECT) w konsoli wpisuję następujące polecenia:



6.4 Aby umożliwić użytkownikowi user wprowadzanie komend do tabeli używam komendy: 

6.5

**--------------------------------------------------------------------------------------------------------------------------------------**

Próby utworzenia pliku .exe do skryptu python, aby uruchomić program na dowolnym komputerze bez zainstalowanego środowiska python.

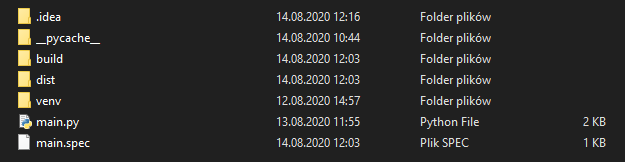
Po kilku błędnych próbach poprawne rozwiązanie wygląda następująco:

7.1 W folderze projektu otworzyłem konsolę cmd.

7.2 Zainstalowałem potrzebną wtycznę/program, uruchomiłem go na moim projekcie używając komend:

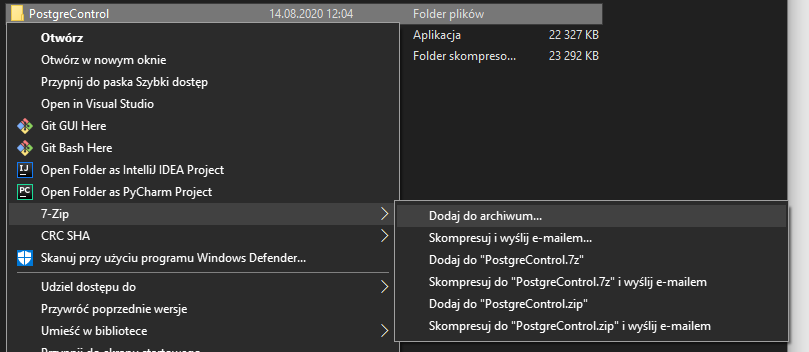
python -m pip install pyinstaller

python -m PyInstaller main.py

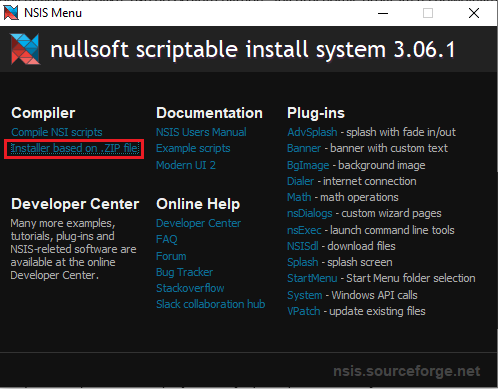
7.3 Zostały utworzone nowe foldery. 

W folderze dist znajduje się plik exe do naszego programu.

7.4 Aby ułatwić przenoszenie projektu między komputerami chcę dodać instalator projektowy. W tym celu pobieram program NSIS oraz dodaję folder projektowy do archiwum zip.



7.5 W programie NSIS uruchamiam:



Następnie Open -> wybieram folder projektowy -> Generate

Tak utworzony plik exe pozwala na uruchomienie mojego skryptu na komputerach z systemem windows bez instalacji środowiska python. Sprawdziłem to na komputerze z sąsiedniej sali w instytucie.

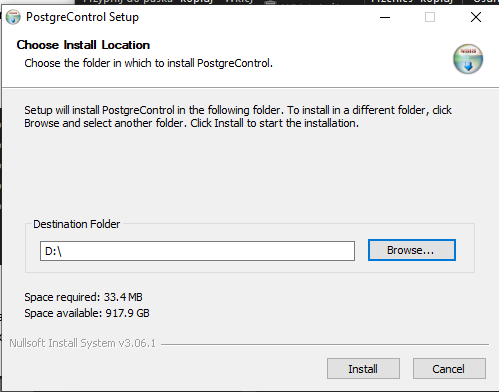
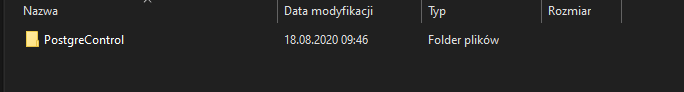
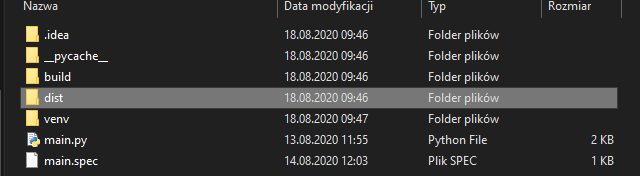
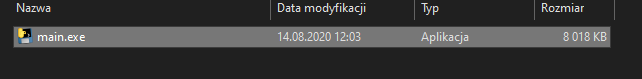
Aby połączenie innego komputera z bazą danych stacjonującą na innym komputerze należy wyłączyć zaporę (firewall) na komputerze z bazą danych.

----------------------------------------------------------------------------------------------------------------------------

**Instrukcja dla użytkownika:**

1. Otrzymany program należy uruchomić zatwierdzając wyskakujące komunikaty.

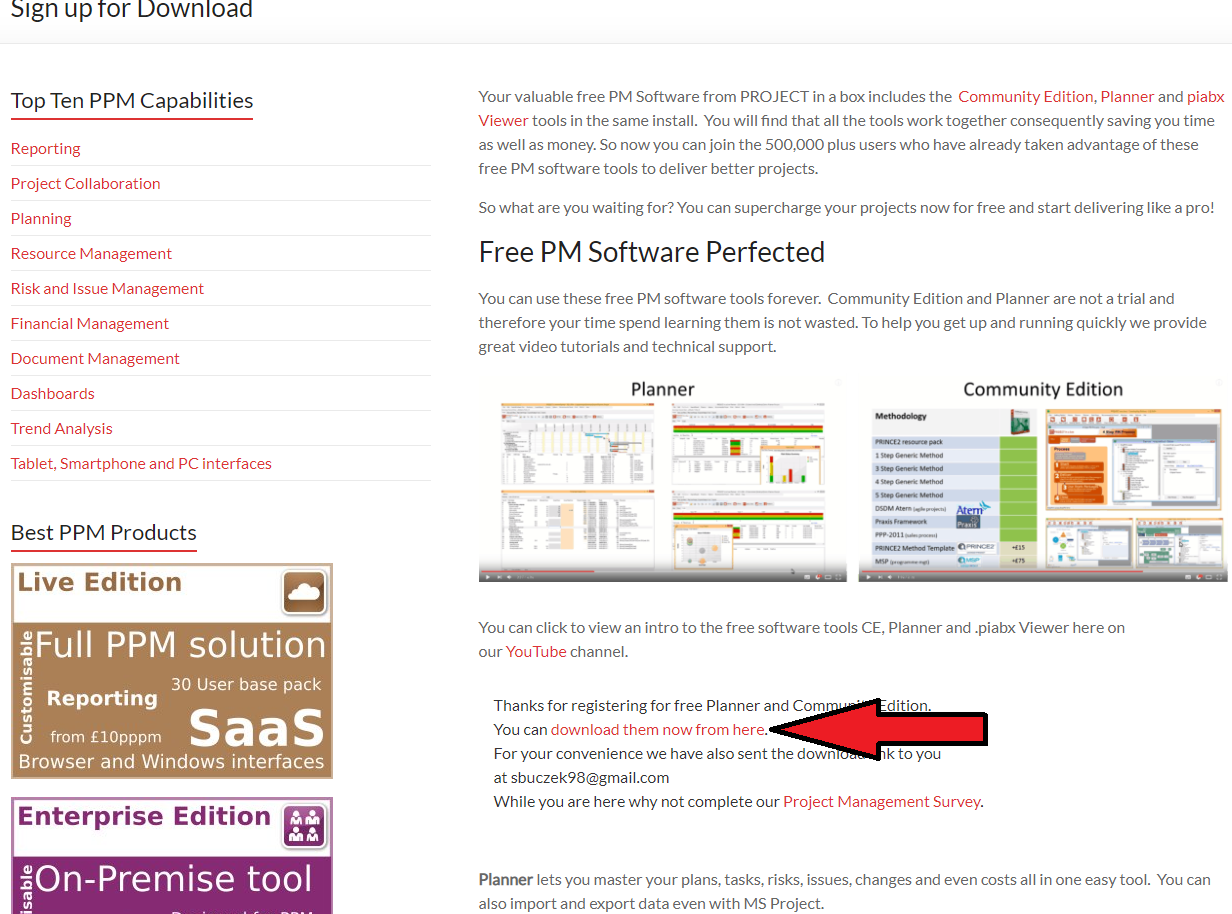
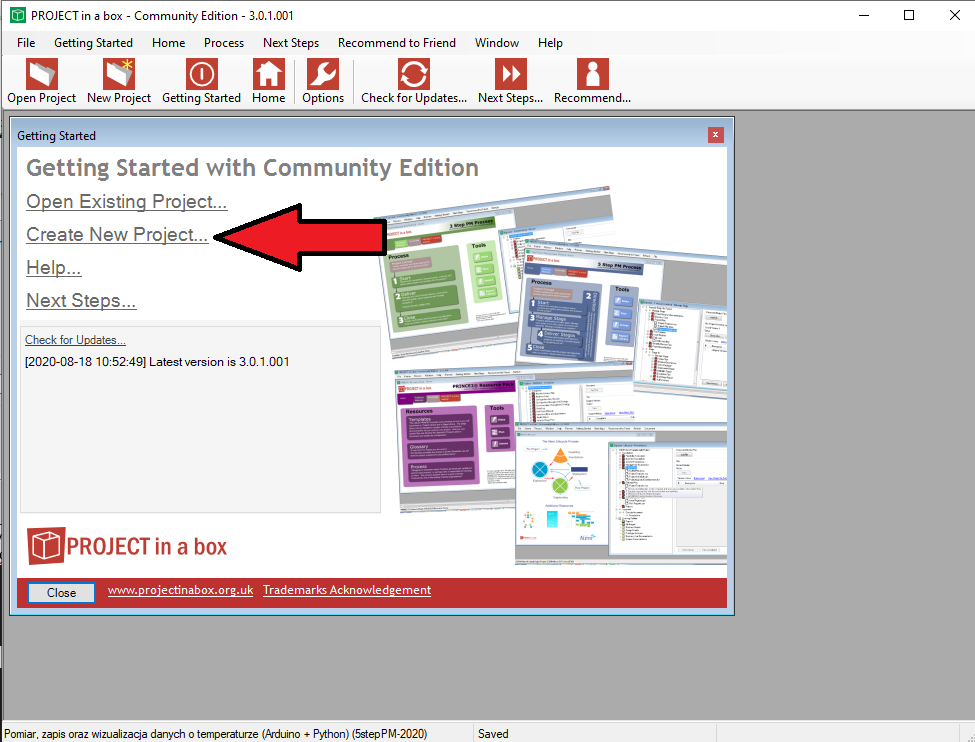
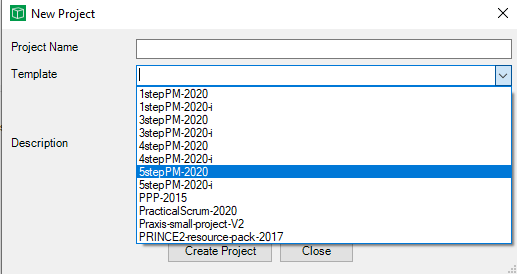
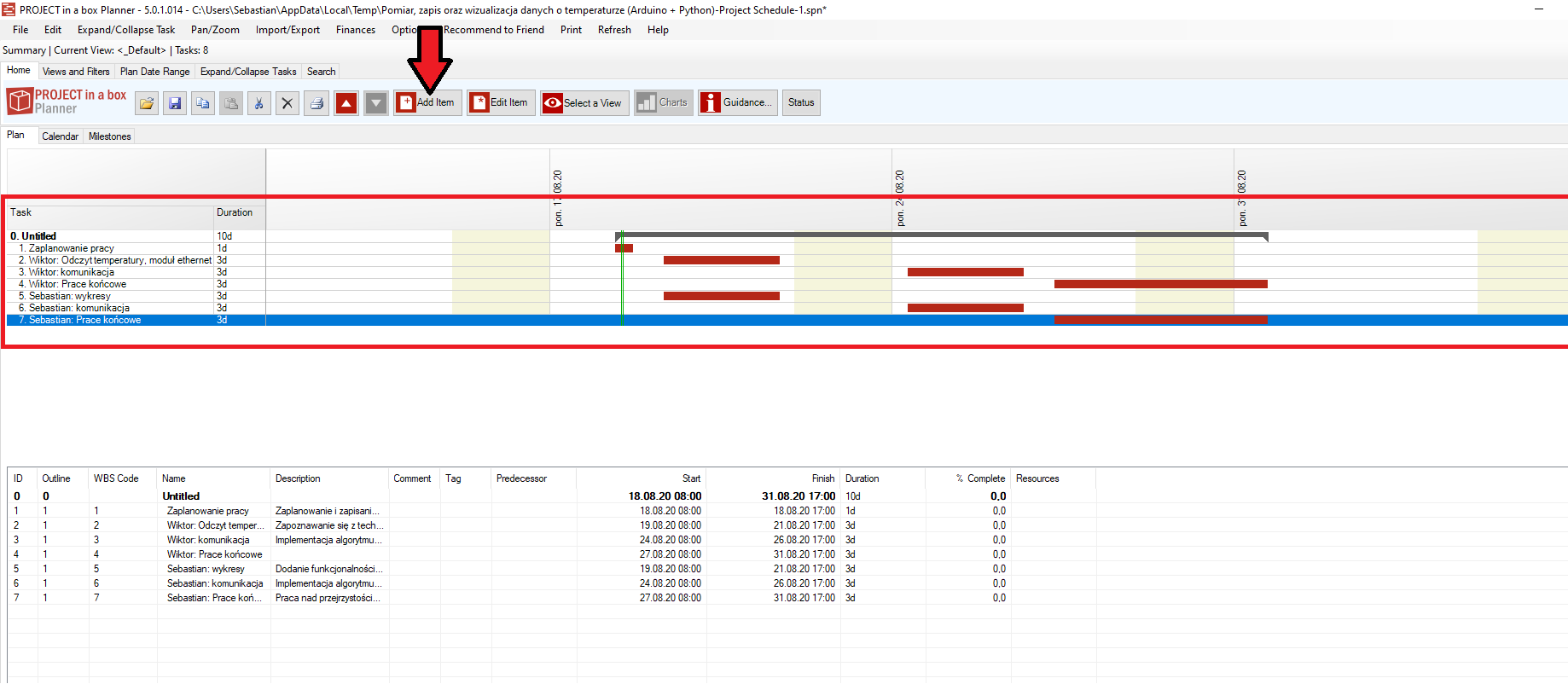


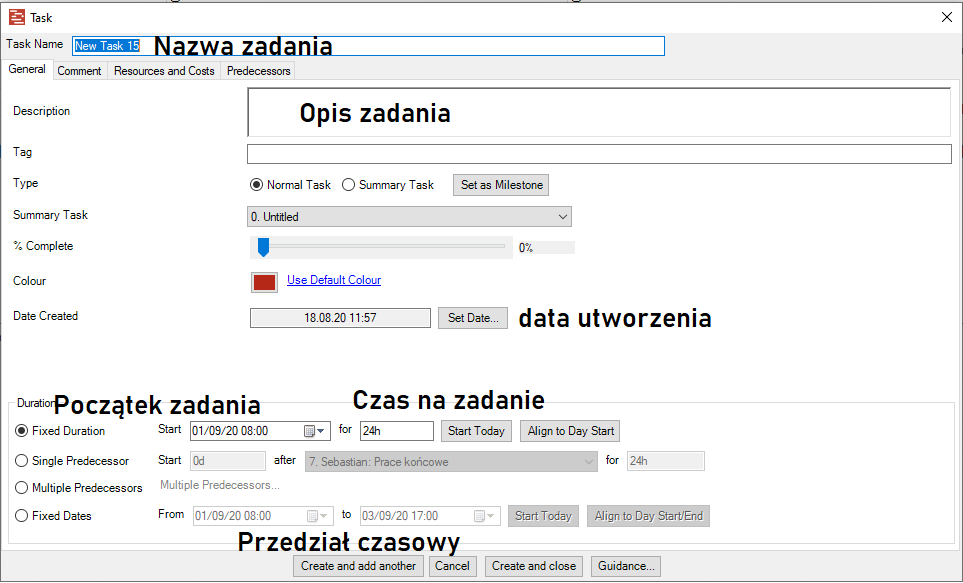
1. Wybieramy gdzie chcemy zapisać nasz program i zatwierdzamy przyciskiem Install.
2. Po instalacji przechodzimy do folderu który wybraliśmy jako instalacyjny (krok 2)
3. Otwieramy folder PostgreControl
4. A następnie folder dist 
5. Uruchamiamy aplikację main.exe 

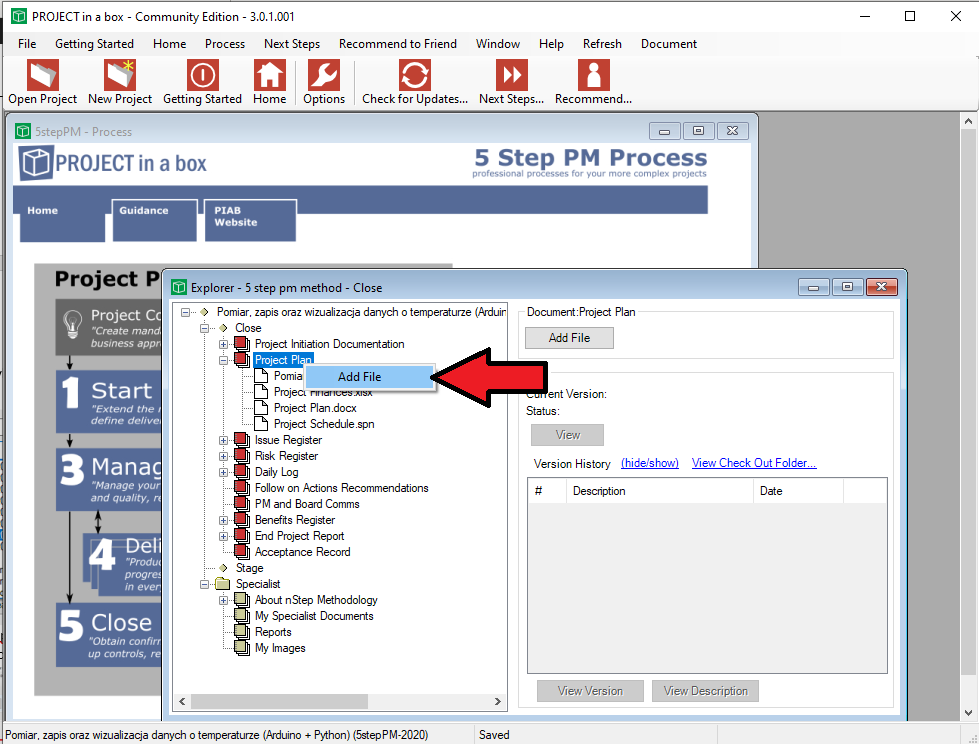
**Koniec instrukcji**

--------------------------------------------------------------------------------------------------------------------------------------

**Utworzenie planu projektu z użyciem programu Project in the box**

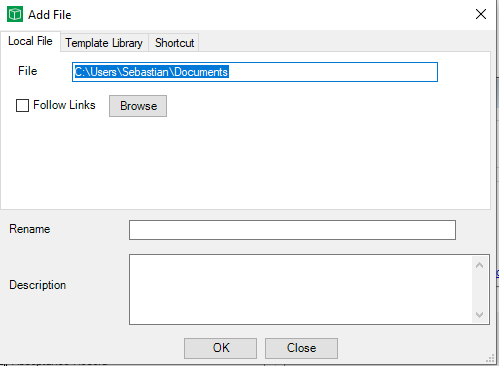
1. Pobieram oprogramowanie Project in the box Community edition. Link: <https://www.projectinabox.org.uk/registration/> (należy wypełnić dane a następnie wybrać: 
2. Instaluję z domyślnymi ustawieniami.
3. Tworzę nowy projekt:
4. Nadaję nazwę, a jako wzór wybieram zaznaczony poniżej: 
5. W programie Project in a box planner tworzę nowy plan (File->new)
6. Poprzez przycisk Add Item dodaję nowe zadania (Task) 

Dodawanie pojedynczego zadania opisałem na poniższym obrazku:

1. Zapisuję plik z plannera i dodaję go do projektu w programie Project in a box w następujący sposób:

W zaznaczonym poniżej polu wpisuję ścieżkę do pliku lub wyszukuję plik poprzez przycisk „Browse”. Lokalizację pliku można łatwo sprawdzić patrząc na górny pasek programu Project in a box Planner:



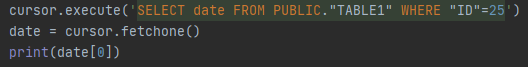


----------------------------------------------------------------------------------------------------------------------------

**Rysowanie wykresów przez skrypt python**

* 1. Próby: Sprawdzam w jakim formacie daty komunikuje się PostgreSQL z pythonem. Dowiedziałem się że wtyczka psycopg2 robi większość koniecznych przekształceń sama.

Kod:



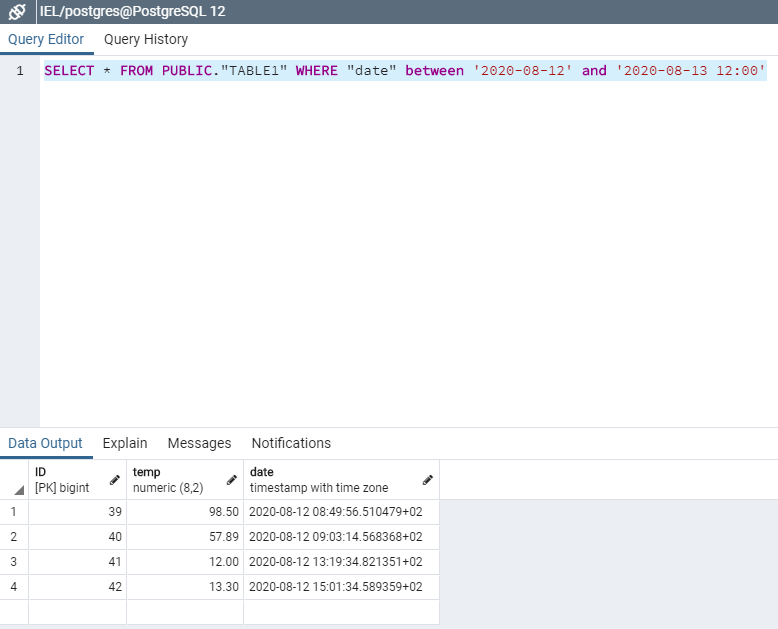
Zwraca:

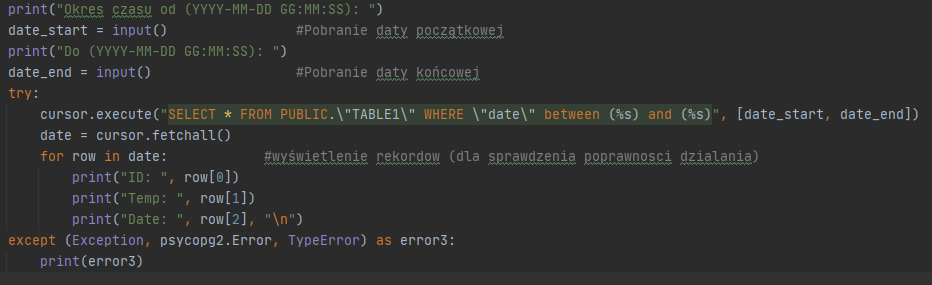


Ważne jest aby używać date[0]. Używanie samego print(date) zwraca następujący wynik:  Jest to tzw. tuple czyli niezmienna tablica oznaczona nawiasami okrągłymi.

1. Chcę pobrać dane z bazy z konkretnego przedziału czasowego za pomocą skryptu python.

2.1 Komenda SQL w środowisku postgreSQL wygląda następująco (przykładowy przedział)



2.2 W skrypcie python chcę, aby data była podawana przez użytkownika w czasie rzeczywistym zatem muszę użyć zmiennych. Wykonałem to w następujący sposób: 

#backslashe są konieczne aby cudzysłowy nie były rozpoznawane jako znaki specjalne dla pythona, a dla postgre

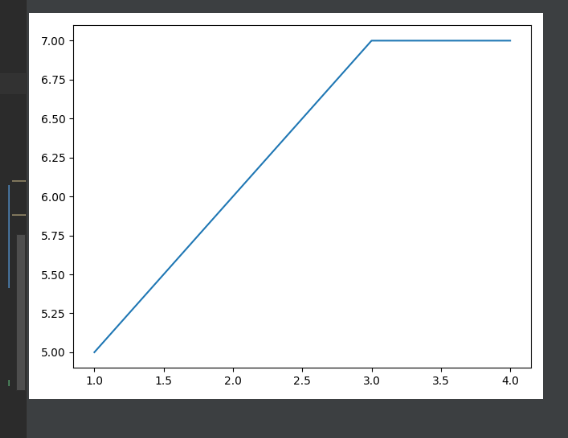
1. Instaluję bibliotekę matplotlib do rysowania wykresów

3.1 W cmd polecenie  zwraca błędy więc uruchamiam konsolę Anaconda Prompt (powinna być na komputerze po zainstalowaniu Pythona) i wpisuję:  i biblioteka instaluje się poprawnie.

3.2 Test:

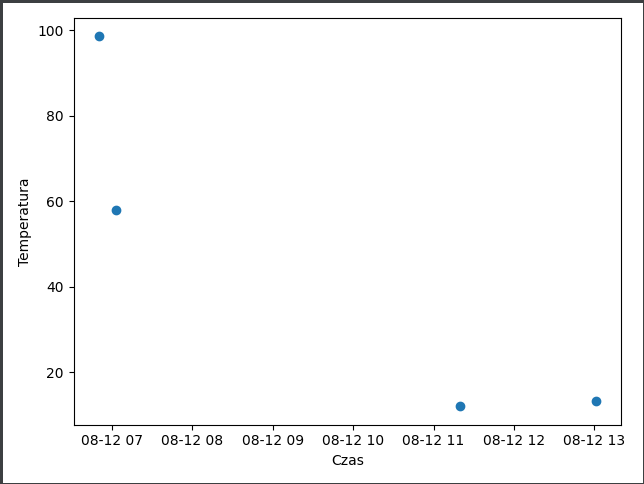


Wynik:



3.3 Próbuję utworzyć wykres na podstawie danych z tabeli z danego okresu czasu.

Pierwszy przykładowy wykres wygląda następująco:



Duży problem sprawiło mi formatowanie daty. Na wykresie widać tylko [miesiąc-dzień godzina]. Niestety na ten moment tak zostanie. Zamierzam kontynuować próby poprawy tego w poniedziałek.

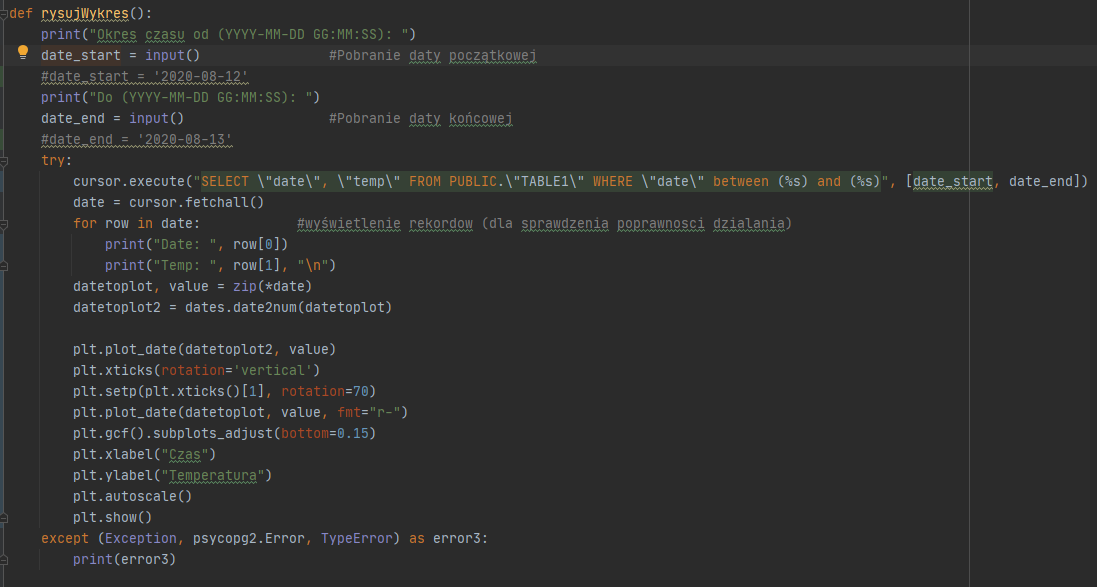
Demo Kod:



Po poprawkach estetycznych wykres wygląda następująco:



Kod:



Użytkownik podaje przedział czasowy według instrukcji programu. Program rysuje wykres z zadanego przedziału.

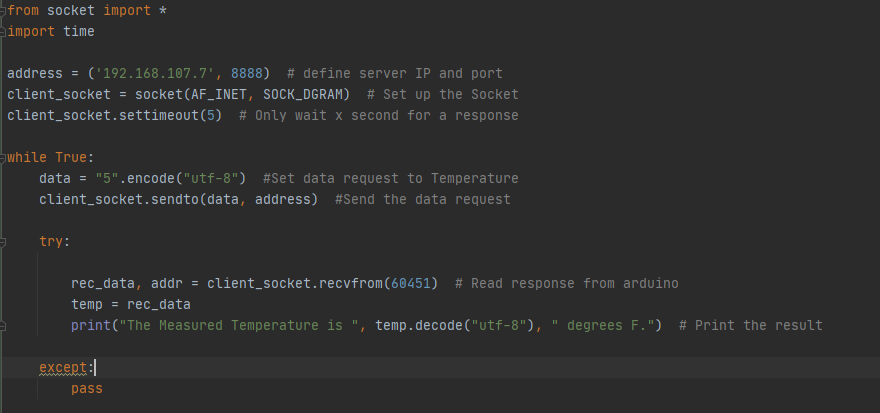
---------------------------------------------------------------------------------------------------------------------------------

Komunikacja Arduino-Python z pomocą UDP.

Spodziewany efekt końcowy to działająca komunikacja serwera (arduino) z klientem (python).

1. Utworzyłem następujący demo—kod w arduino: 

Funkcja udpServerListenOnPort uruchamia nasłuchiwanie udp na porcie ‘myport’. W parametrze podałem funkcję udpSerialPrint() która jest odpowiedzialna za pozyskanie adresu ip nadawcy, jego portu oraz danych które chce przekazać. Jest to potrzebne aby wysłać odpowiedź w odpowiednie miejsce za pomocą funkcji makeUdpReply().

1. Demo—kod wysyłający i odbierający pakiety w python: 

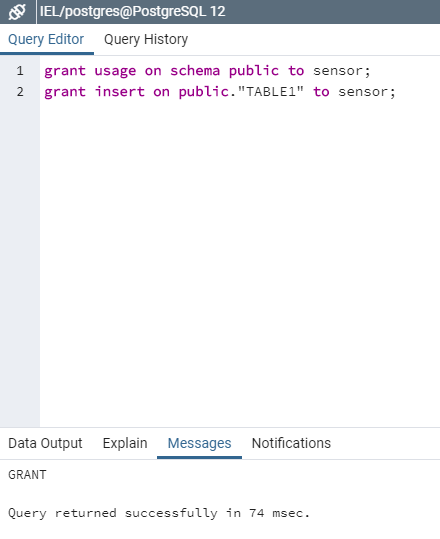
Aktualnie ta komunikacja wygląda następująco:

- python wysyła ‘wiadomość’ o treści ‘5’

- arduino odbiera wiadomość i oddaje jej treść z powrotem do pythona

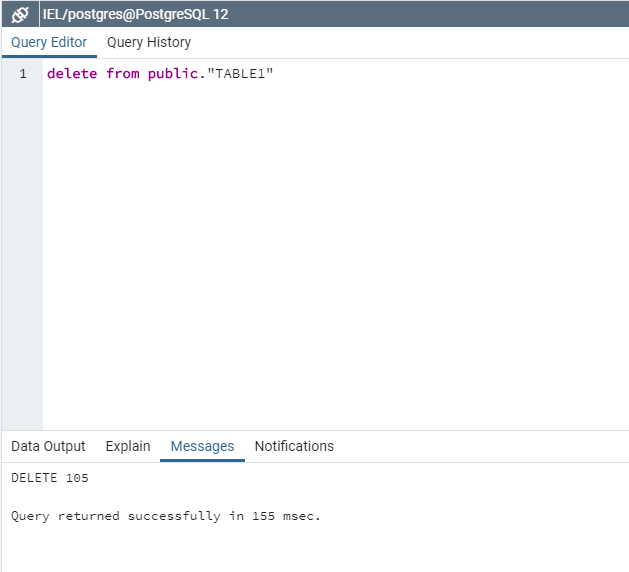
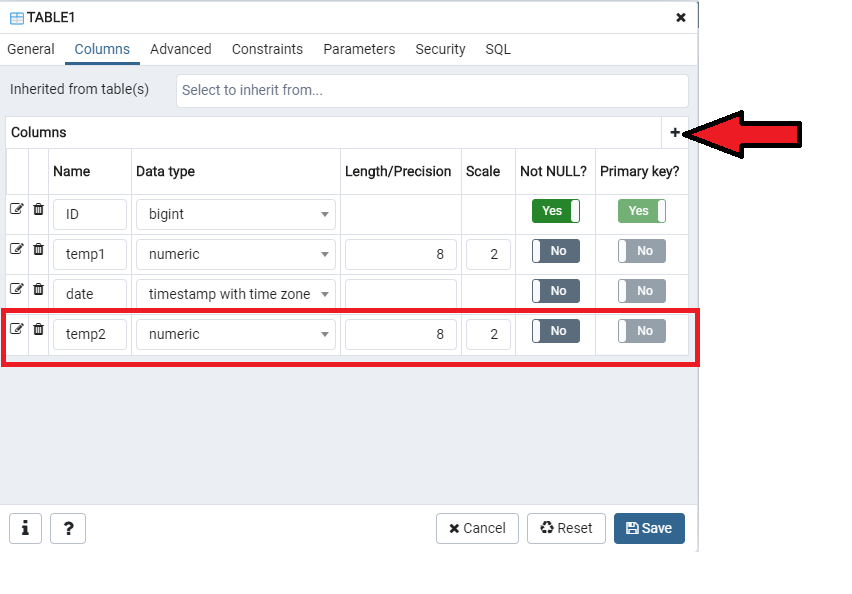
- python odbiera wiadomość

--------------------------------------------------------------------------------------------------------------------------------------

Tworzę konto użytkownika w bazie danych dla czujnika. Nadaję mu możliwość dodawania rekordów do tabel. Po utworzeniu konta (tak jak wyżej w raporcie) używam komendy: 

--------------------------------------------------------------------------------------------------------------------------------------

**Zmieniam strukturę tabeli** (dodaję nową kolumnę dla pomiaru z czujnika 2)

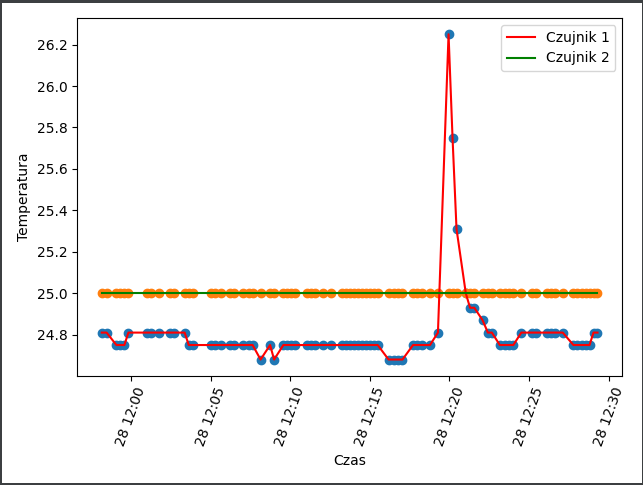
1. Poniższym poleceniem usuwam wszystkie dotychczasowe rekordy z tablicy. Robię to ponieważ dotychczasowe rekordy przechowy
2. TABLE1 -> Properties, a następnie 

-------------------------------------------------------------------------------------------------------------------------------------

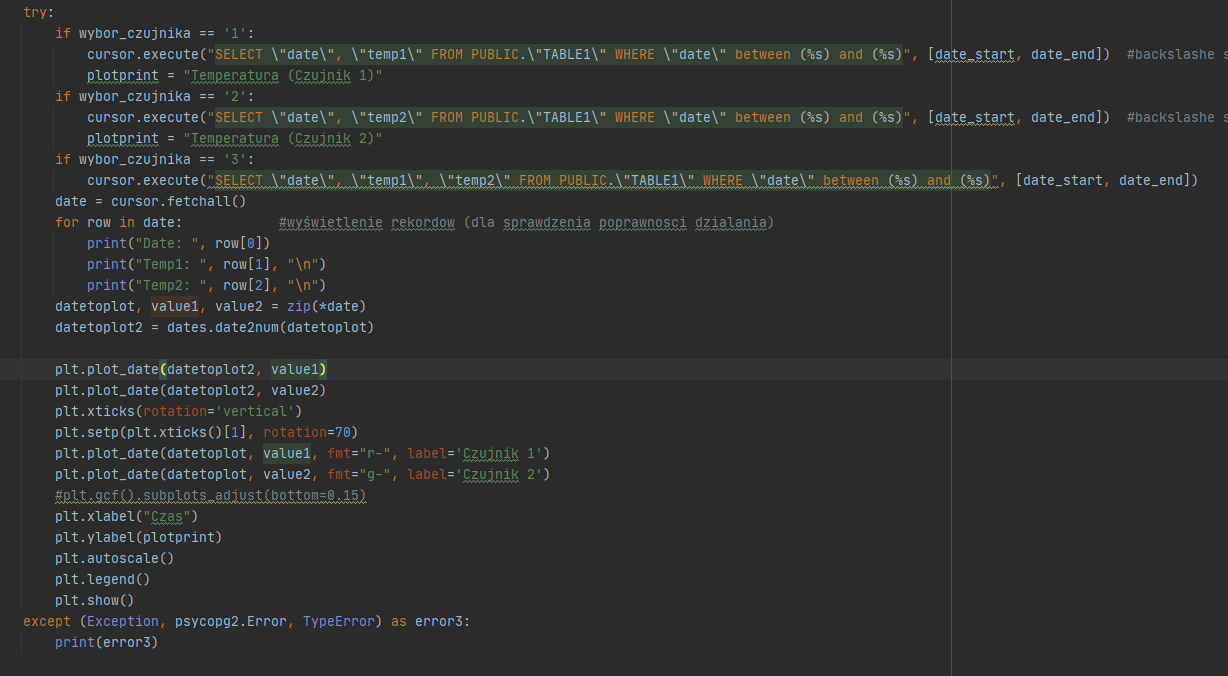
**Kontynuacja wykresów**

1. Dodałem funkcję rysowania wykresu zawierającego dane z dwóch czujników.

Efekt:



Kod:



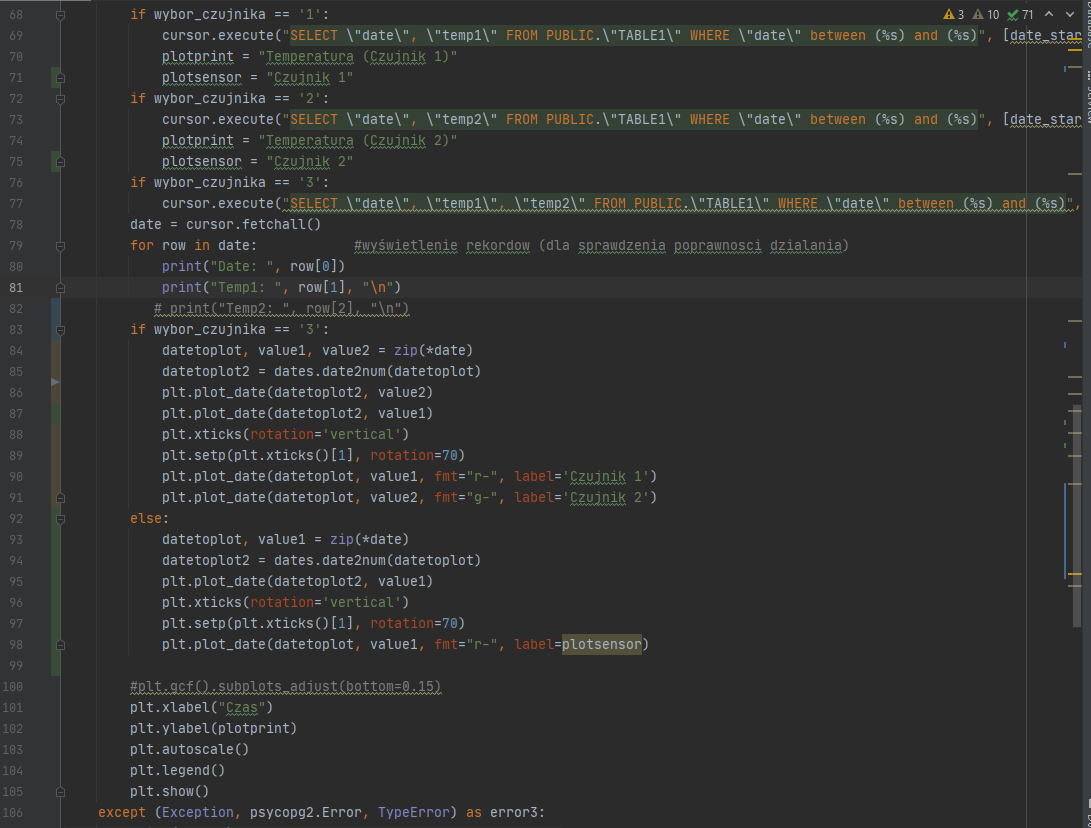
Jest to modyfikacja względem poprzedniej wersji, której zrzut ekranu widoczny jest wcześniej w raporcie.

Całość kodu dostępna jest na moim profilu na githubie pod adresem: <https://github.com/SebastianBuczek98/IEL_praktyka>

--------------------------------------------------------------------------------------------------------------------------------------

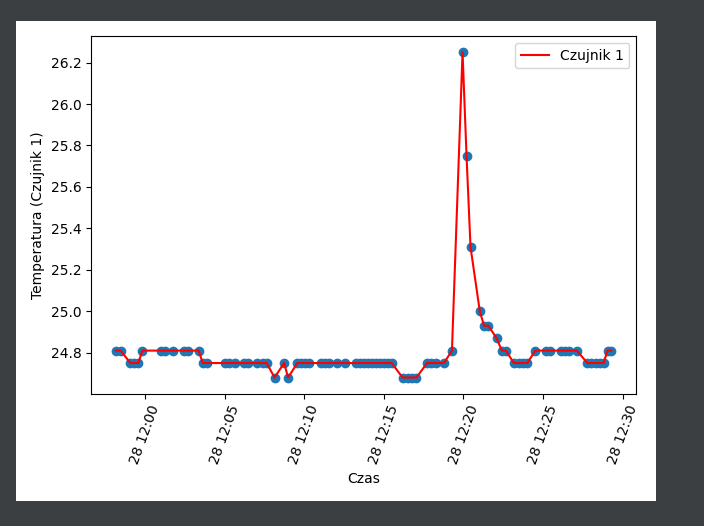
Powyższy kod zawiera błąd – wykresy dla pojedynczego czujnika wyrzucają błąd ponieważ pobierając dane o temperaturze z jednego czujnika zmienna value2 zostaje pusta (python nie ma co w nią wpisać).

Wprowadziłem poprawki do kodu niwelujące taką sytuację. Po sprawdzeniu potwierdzam poprawne działanie programu.

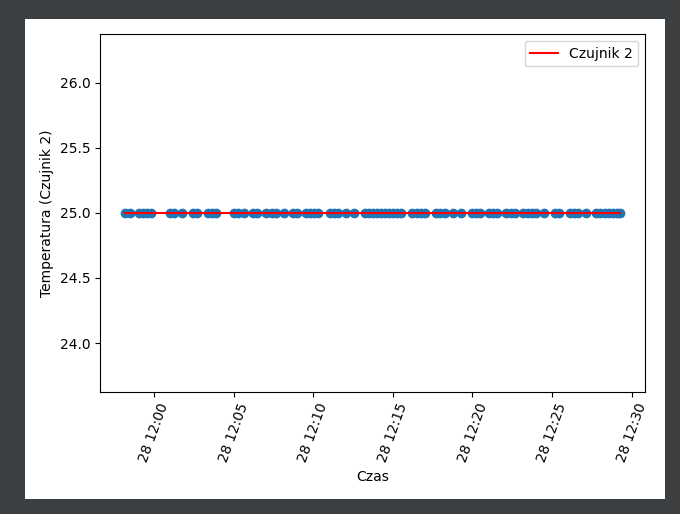
Kod: 

Potwierdzenie działania:

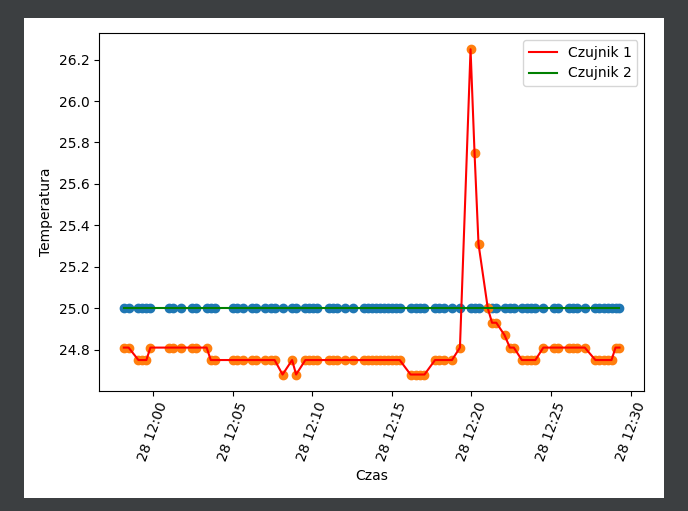
Czujnik 1:



Czujnik 2:

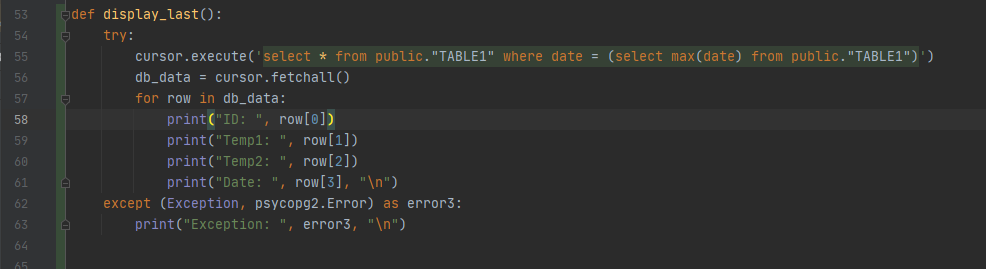


Obydwa czujniki na raz:



--------------------------------------------------------------------------------------------------------------------------------------

Dodaję funkcję odpowiedzialną za wyświetlenie ostatniego (najnowszego) pomiaru z bazy danych.



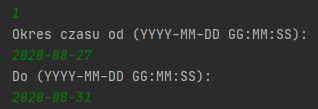
Test:



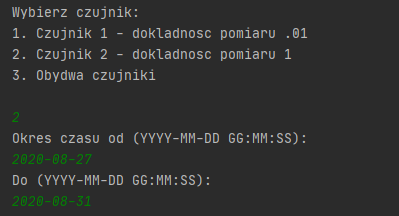
--------------------------------------------------------------------------------------------------------------------------------------

Dane do wyświetlenia czytelnego wykresu do prezentacji:

Czujnik 1:



Czujnik 2:



Czujnik 3:

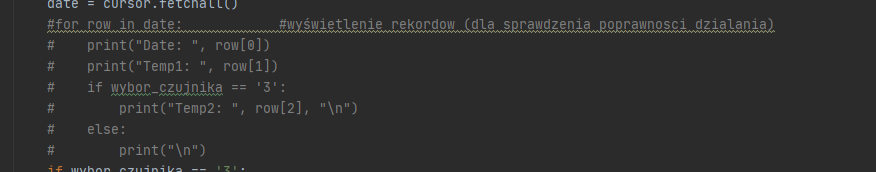
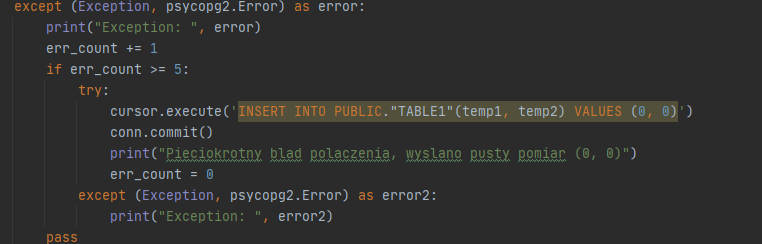


--------------------------------------------------------------------------------------------------------------------------------------

**Komunikacja UDP**

--------------------------------------------------------------------------------------------------------------------------------------

**Usprawnienia:**

1. PostgreControl:
   1. Usunięcie wypisania pomiarów do konsoli przy rysowaniu wykresu (Zakomentowane): 
2. Py\_arduino\_udp
   1. Zapisanie do bazy informacji o błędach np. z połączeniem – po zarejestrowaniu 5 błędów zostanie do bazy wysłany pomiar (0, 0) którego normalnie system nie obsługuje. 
3. **Utworzenie plików exe.**

3.1 Aby utworzyć plik exe dla programu PostgreControl wymagana była zmiana wersji biblioteki matplotlib na wcześniejszą. Wykonałem to poleceniem

pip install matplotlib==3.1.2

3.2 Pliki exe tworzę w taki sam sposób jak wyżej w raporcie z jedną różnicą – nie dodaję do archiwum zip całego folderu projektowego a jedynie folder dist utworzony przez PyInstaller (wyżej w raporcie).

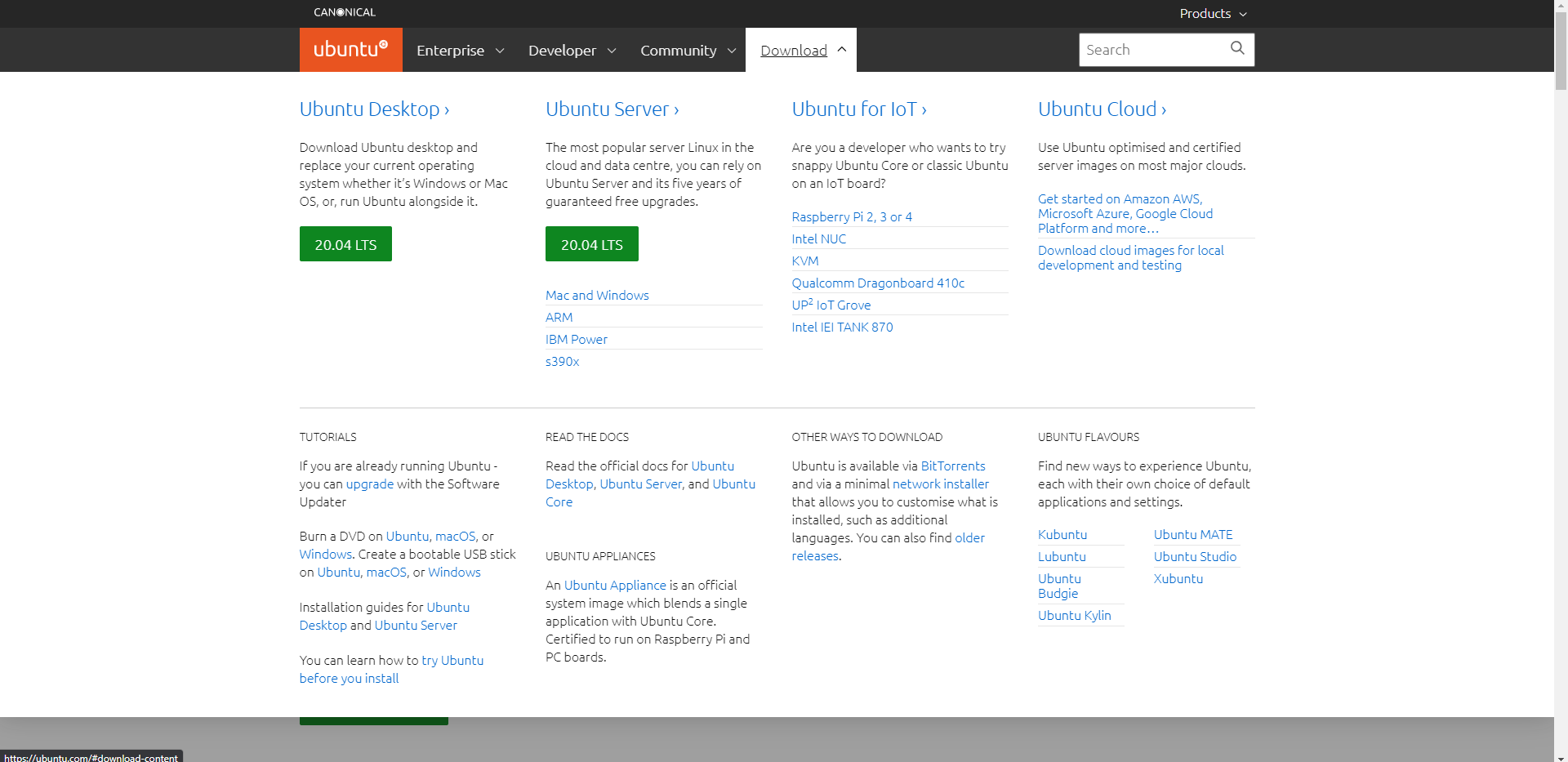
4. Uruchomienie plików z początku sprawiało kilka problemów. Należy pamiętać o dodaniu adresu ip komputera do pliku pg\_hba.conf.

4.1 Utworzone pliki exe pozwalają na uruchomienie skryptów bez zainstalowanego środowiska python.

Aby poprawnie uruchomić program należy go zainstalować (uruchomić plik exe, najlepiej utworzyć nowy folder w preferowanej lokalizacji a następnie go wybrać i zainstalować w nim program), następnie w folderze przejść do katalogu dist, wyszukać plik main.exe i uruchomić. Może być wymagane wyłączenie zapory (firewall).

-------------------------------------------------------------------------------------------------------------------------------------

**Instalacja systemu UNIX na komputerze instytutu**

1. Pobieram plik iso Ubuntu Desktop w najnowszą wersję: 
2. A także PowerISOx32 link: https://www.poweriso.com/download.php
3. Instaluję ubuntu z zaznaczeniem funkcji „Erase disk and install Ubuntu”. Zainstalowana wersja to 20.04.

Nazwa komputera oraz użytkownika: iel-zne-serwer

Hasło: ielnauka2020

1. Po zakończeniu instalacji oraz zresetowaniu komputera przechodzę do instalacji PostgreSQL oraz pgAdmin. Wykorzystuję do tego następujące komendy:
2. wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
3. sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ $(lsb\_release -sc)-pgdg main" > /etc/apt/sources.list.d/PostgreSQL.list'
4. sudo apt update
5. sudo apt-get install postgresql-11
6. sudo passwd postgres
7. sudo su -l postgres
8. su - postgres
9. psql
10. postgres=# \password postgres
11. \q
12. Exit

(wszystkie hasła to: ielnauka2020)

1. sudo apt-get install pgadmin4 pgadmin4-apache2

(w wyskakującym oknie ustawiam hasło ielnauka2020)

1. Uruchamiam pgAdmin i tworzę strukturę bazy tak jak w przypadku systemu windows.
2. Zgrywam dane starej bazy z komputera który dotychczas pełnił rolę serwera na zewnętrzny nośnik.
3. Aby dodać adresy mogące łączyć się z bazą wyszukuję pliku pg\_hba.conf – początkowo znaleznienie go było problematyczne. Oto rozwiązanie:
4. cd /etc/postgresql/12/main
5. sudo gedit pg\_hba.conf

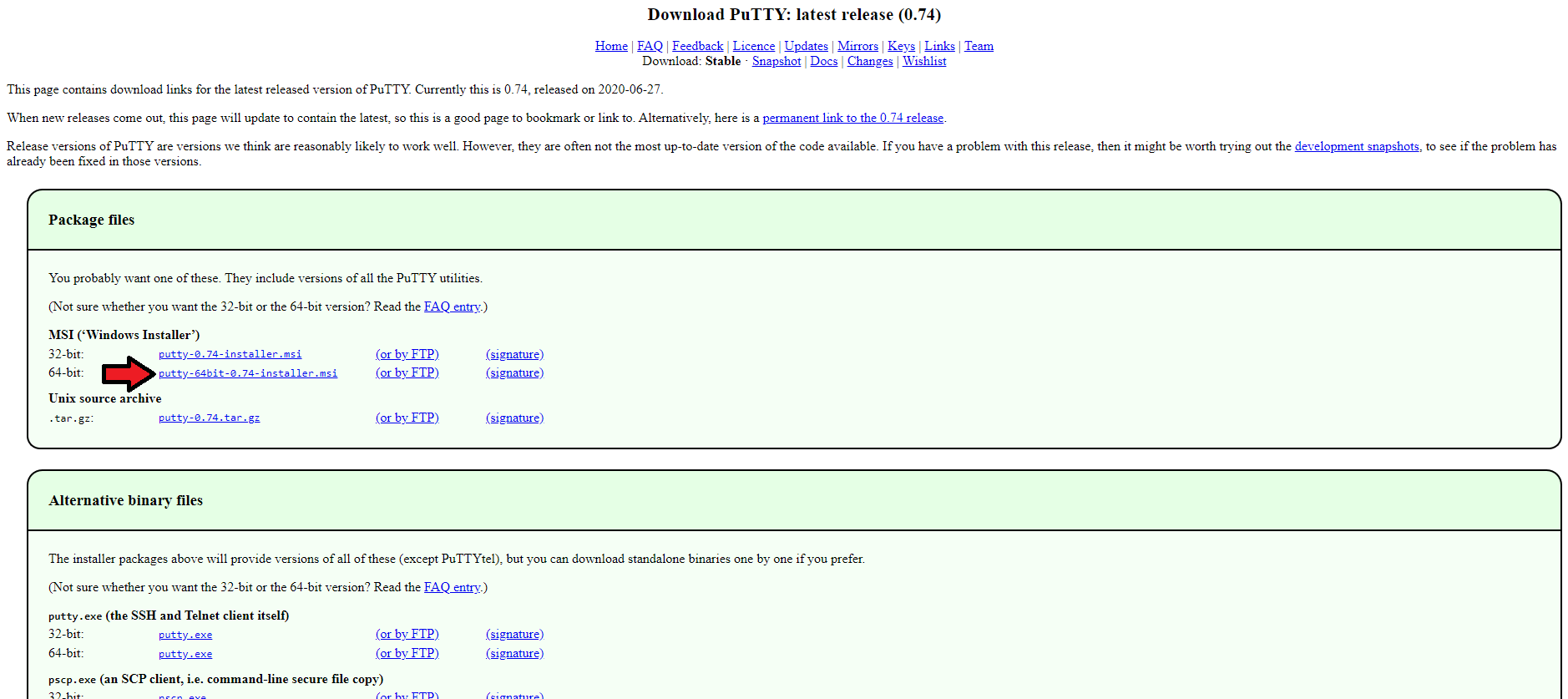
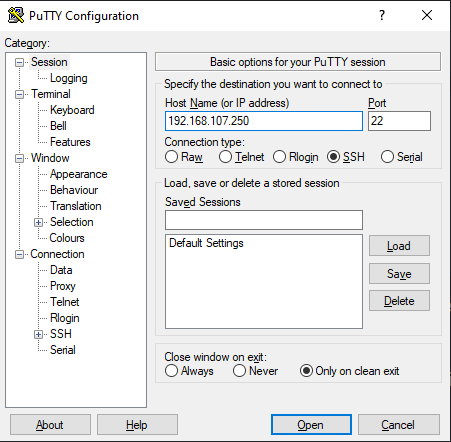
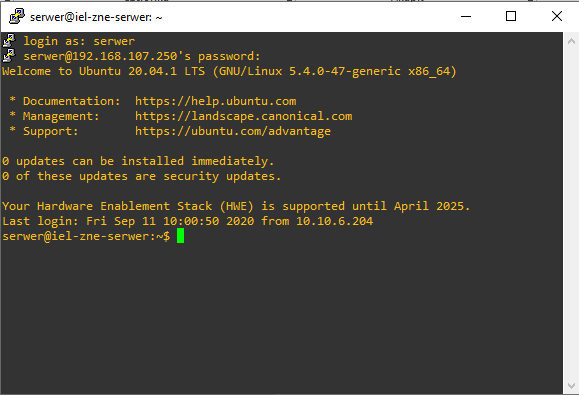
Może zdarzyć się tak, że nie będziemy mieli uprawnień do wykonania wyzej wykonanych poleceń. Należy wtedy posłużyć się kontem postgres jako użytkownik bazy (w terminalu wpisać - su – postgres i następnie spróbować przejść do folderu main.

1. **W pliku postgresql.conf zmieniam atrybut listen address na ‘\*’ co umożliwia łączenie się do bazy po sieci lan.**

---------------------------------------------------------------------------------------------------------------------------------

**Połączenie SSH z komputerem serwerowym**

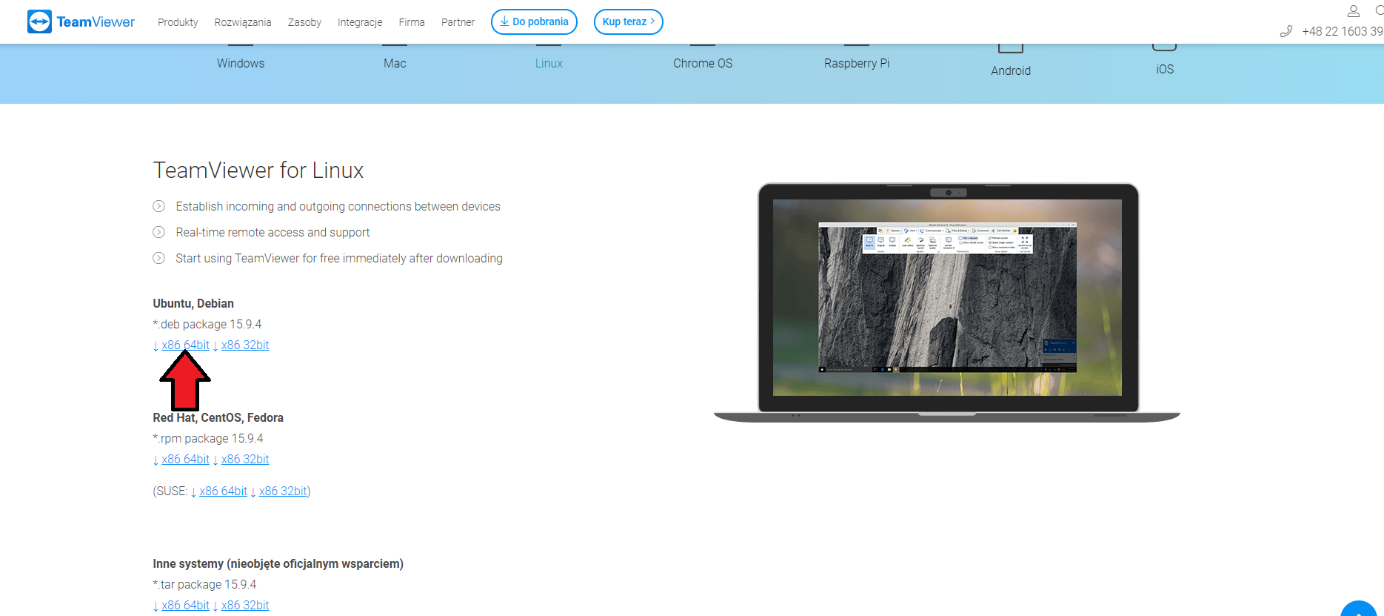
Chcę umożliwić zarządzanie komputerem serwerowym za pomocą innego komputera. Chcę do tego wykorzystać połączenie SSH. W przyszłości możliwe że wykorzystam również program TeamViewer do zdalnej kontroli komputera.

1. Instalacja SSH
2. Sudo apt install ssh
3. Sudo systemctl enalbe --now ssh
4. Sudo systemctl status ssh
5. Pobieram program Putty
6. <https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>
7. 
8. Instaluję z domyślnymi ustawieniami
9. Wpisuję adres serwera i ustawiam parametry jak poniżej:
10. Jako login wpisuję**:** **serwer**
11. Jako hasło: ielnauka2020
12. 

---------------------------------------------------------------------------------------------------------------------------------

**Zdalne sterowanie serwerem**

1. Instaluję program TeamViewer na ubuntu poprzez pobranie pliku .deb z oficjalnej strony

link: [https://www.teamviewer.com/pl/do-pobrania/linux/](https://www.teamviewer.com/pl/do-pobrania/linux/#)

1. Instaluję poprzez dwuklik na pobrany plik i dalej domyślnie zgodnie sugestiami instalatora.
2. Po uruchomieniu programu widzimy nast