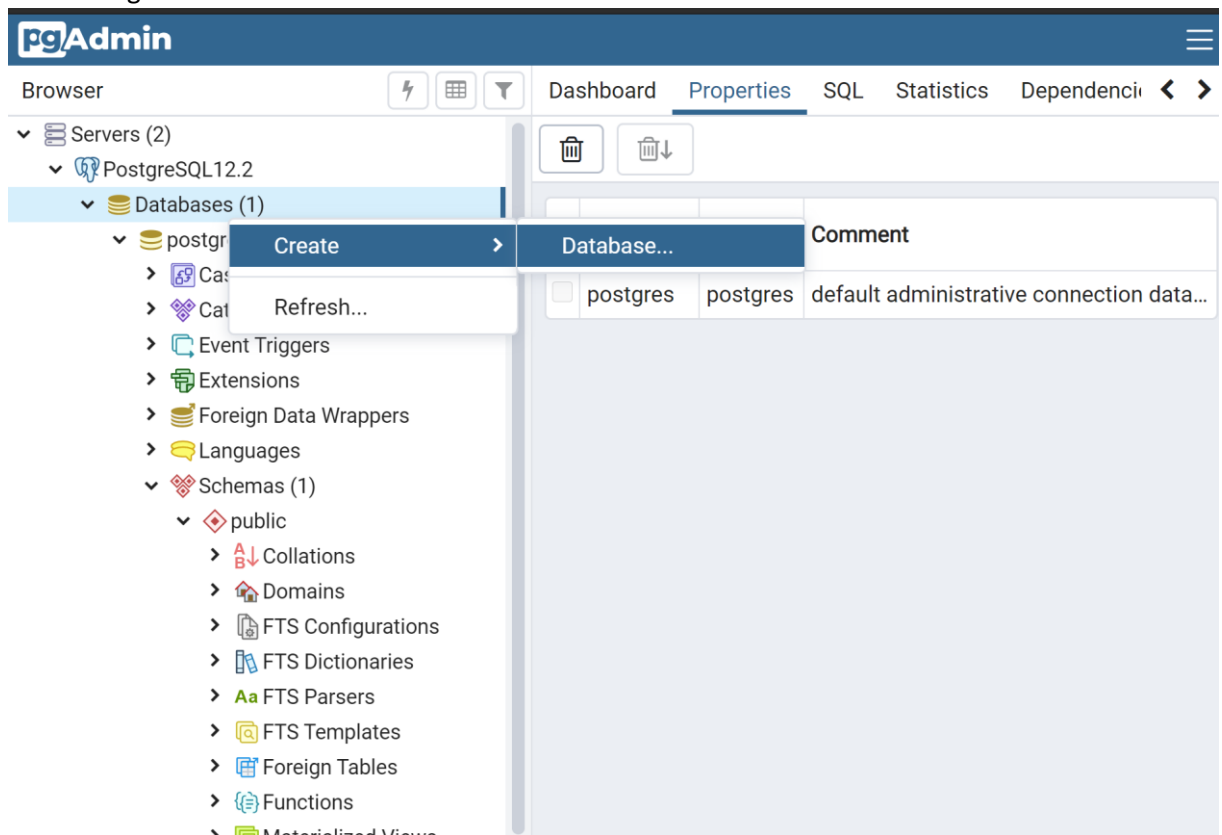
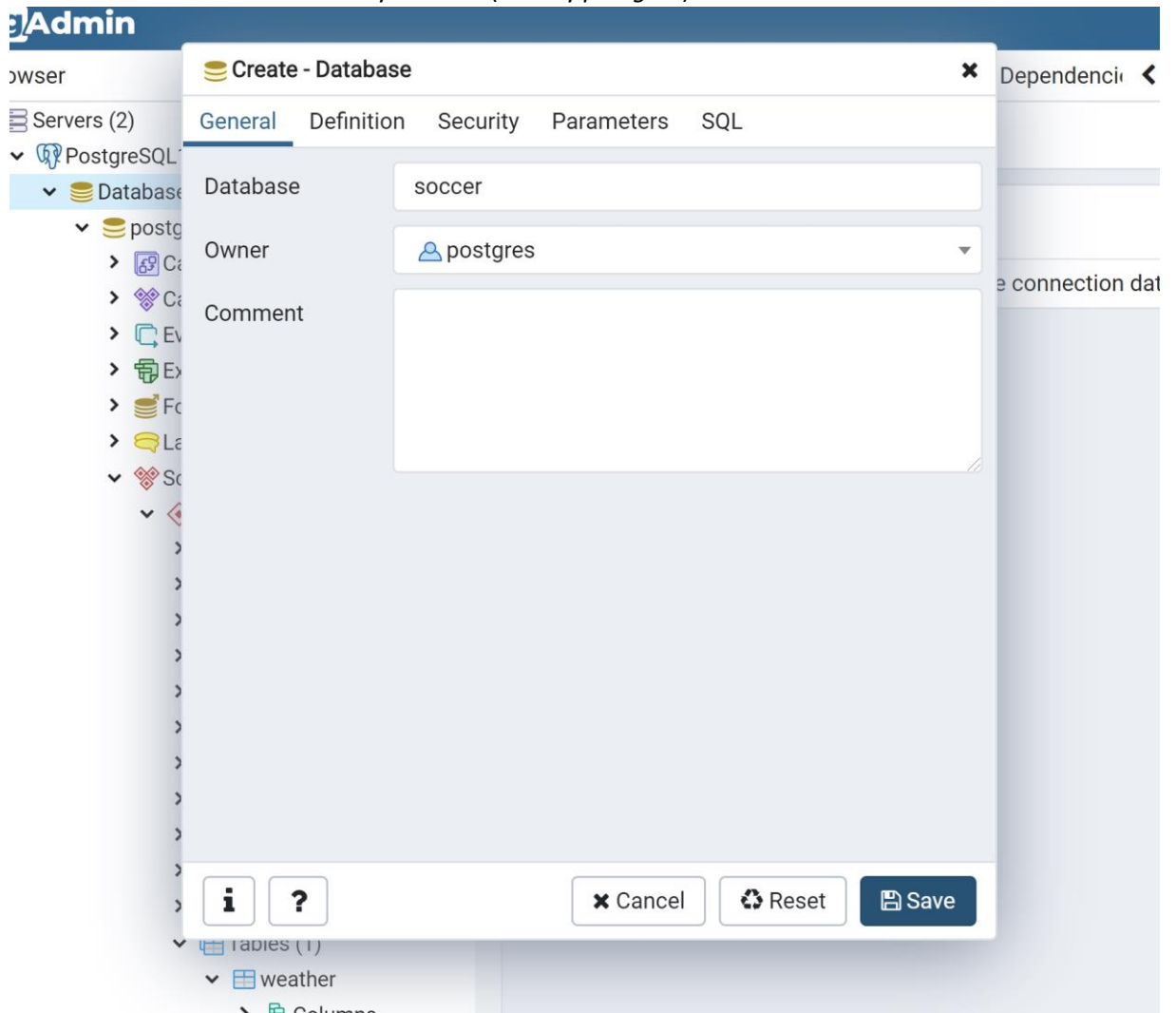


Database Project Setup Guide

1. Install Docker
2. Install pgadmin with PostgreSQL 12
3. Install Python 3.6.8
4. Optional: Create an environment
5. Install dependencies
 - `pip install -r requirements.txt`
 - `pip install -r requirements2.txt`
7. Control into directory "src"
 - `cd to Flask/docker_database/dockerized/src`
8. Open pgadmin and create a new database and name it "soccer"
 - a. Right-click on Databases -> Create -> Database...



- b. Enter the Database name: *soccer*
- c. Make sure the Owner is your user (usually *postgres*)



[illegible]

d) Now open config.py in visual studio or another IDE ->

e) change ***user*** and ***password*** to YOUR pgadmin-credentials in the DATABASE_URI string:

DATABASE_URI="postgresql://{user}:{password}@localhost:5432/soccer"

The image shows a Visual Studio Code editor window with the following details:

- Menu Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Tab Bar:** config.py - Dockerized - Visual Studio Code.
- Explorer Sidebar:** Lists files in the project:
 - models.py
 - team.html
 - title.html
 - app.py
 - config.py (selected)
 - query_title.html
 - __init__.py
 - index.html
 - query_exists.html
- Source Control Sidebar:** Shows a list of files with status indicators:
 - > OPEN EDITORS (1 UNSAVED)
 - DOCKORIZED
 - vscode
 - src
 - __pycache__
 - templates
 - base.html
 - coach.html
 - crud.html
 - home.html
 - index.html
 - index2.html
 - login.html
 - membership.html
 - player.html
 - query_exists.html
 - query_goalgetter.html
 - query_membership.html
 - query_table.html
 - query_teamgoals.html
 - query_title.html
 - show_user.html
 - sponsoring.html
 - table.html
 - team.html
 - title.html
 - __init__.py
 - app.py
 - config.py (selected)
 - database.py
 - initial_insert.py
 - models.py
 - database.conf
 - docker-compose.yaml
 - requirements.txt
 - requirements2.txt
- Main Editor:** Displays the content of config.py:

```
src > config.py > ...
1
2 import os
3
4 user = os.environ.get('POSTGRES_USER')
5 password = os.environ.get('POSTGRES_PASSWORD')
6 host = os.environ.get('POSTGRES_HOST')
7 database = os.environ.get('POSTGRES_DB')
8 port = os.environ.get('POSTGRES_PORT')
9
10 # DATABASE CONNECTION URI = "postgresql+psycopg2://{user}:{password}@{host}:{port}/{database}"
11 DATABASE_CONNECTION_URI = "postgresql+psycopg2://{postgres:MyPassword}@localhost:5432/soccer"
12
13 SECRET_KEY = os.environ.get('SECRET_KEY') or 'you-will-never-guess'
14
15 basedir = os.path.abspath(os.path.dirname(__file__))
```

.....

If you followed the instructions and installed everything correctly, you can start the FLASK-Server:

- a. (Optional) In case you created a virtual environment: make sure you started this environment

9.

Windows Powershell:

➤ `$env:FLASK_APP = "app.py"`

Linux:

➤ `export FLASK_APP = "app.py"`

10.

➤ `flask run`

On Windows 10 in Powershell:

```
(venv_db368) PS C:\Users\I530060\OneDrive - SAP SE\Desktop\Semester 4\Database_Project\Flask\docker_database\dockerized\src>
(venv_db368) PS C:\Users\I530060\OneDrive - SAP SE\Desktop\Semester 4\Database_Project\Flask\docker_database\dockerized\src>
(venv_db368) PS C:\Users\I530060\OneDrive - SAP SE\Desktop\Semester 4\Database_Project\Flask\docker_database\dockerized\src>
(venv_db368) PS C:\Users\I530060\OneDrive - SAP SE\Desktop\Semester 4\Database_Project\Flask\docker_database\dockerized\src> $env:FLASK_APP = "app.py"
(venv_db368) PS C:\Users\I530060\OneDrive - SAP SE\Desktop\Semester 4\Database_Project\Flask\docker_database\dockerized\src> flask run
* Serving Flask app "app.py"
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```