

Instructions on Building and Running the Application

January 6, 2022

1 PostgreSQL Database

1.1 Requirements

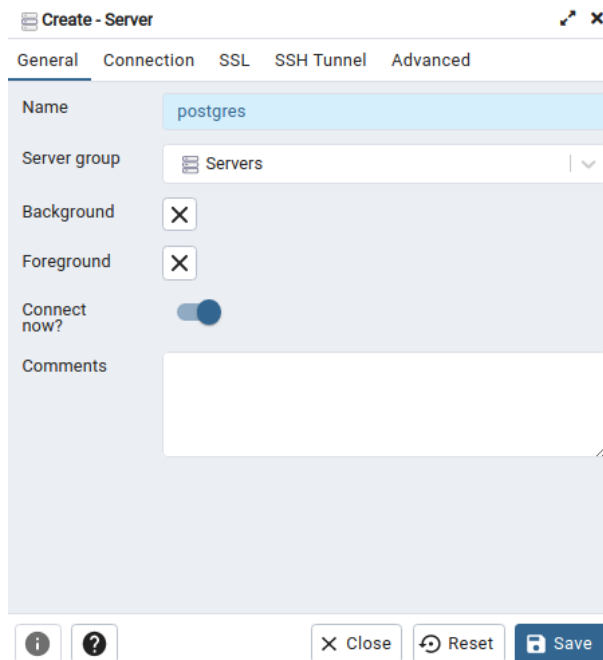
We used an Ubuntu-20.04 to Develop our Project

1. PostgreSQL Installed (We used PostgreSQL 12.9)
2. pgAdmin4 Installed

1.2 Creating a Server

To create a database we must first create a Server. The information that we used for our server can be seen bellow. This information will also be used later by our back-end to establish a connection with the server and the database. You can set your own information when configuring the server or the database but you must also edit the configuration file mentioned bellow.

1.2.1 General Information



The screenshot shows the 'Create - Server' dialog box in pgAdmin4, with the 'General' tab selected. The dialog has a title bar with a hamburger menu icon, the text 'Create - Server', and window control icons. Below the title bar are tabs for 'General', 'Connection', 'SSL', 'SSH Tunnel', and 'Advanced'. The 'General' tab contains the following fields and controls:

- Name:** A text input field containing 'postgres'.
- Server group:** A dropdown menu showing 'Servers' with a downward arrow.
- Background:** A checkbox with an 'X' icon, currently unchecked.
- Foreground:** A checkbox with an 'X' icon, currently unchecked.
- Connect now?:** A toggle switch that is currently turned on (blue).
- Comments:** A large text area for entering comments.

At the bottom of the dialog are three buttons: an information icon (i), a help icon (?), and a 'Close' button. To the right of these are 'Reset' and 'Save' buttons.

1.2.2 Connection Information

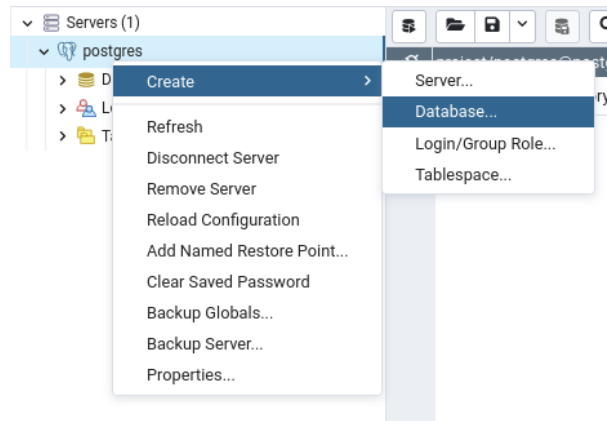
The screenshot shows the 'Create - Server' dialog box with the 'Connection' tab selected. The fields are as follows:

- Host name/address: 127.0.0.1
- Port: 5432
- Maintenance database: postgres
- Username: postgres
- Kerberos authentication?: ☐
- Password:
- Save password?: ☒
- Role: (empty)
- Service: (empty)

At the bottom, there are buttons for 'Close', 'Reset', and 'Save', along with information and help icons.

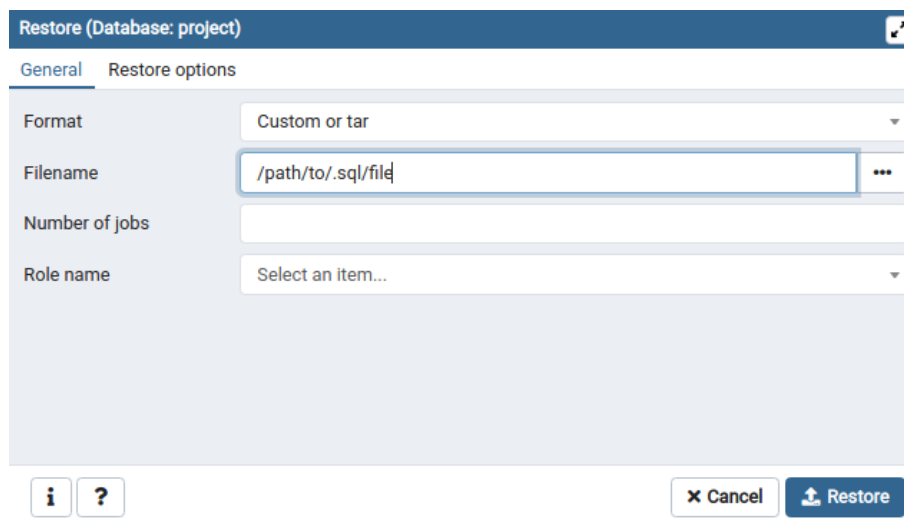
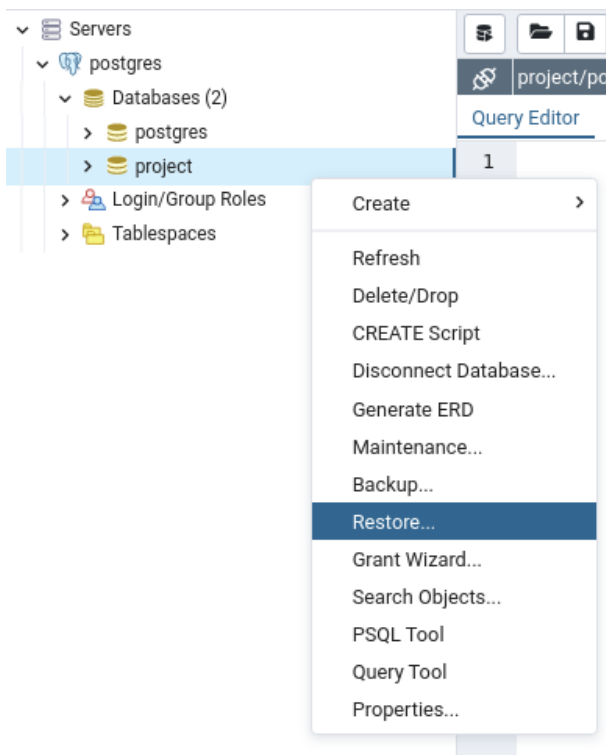
1.3 Creating a Database

After the server is created a database must also be created. To create a database you must right click on the server, go to "Create" section of the pop-up menu and select "Database..." just like the image below.



1.4 Importing Data

After the database is created, all tables, triggers and views must be imported into the database. To do this we must "Restore" the database from the .sql file which was submitted in the project directory. The restoration process can be seen below.



After the "Restore" button is pressed we can move on to the API installation process.

2 Node.js Server

2.1 Requirements

1. Node.js Installed (We used Node.js v14.16.0)
2. npm (We used npm v6.14.11)

2.2 Configure Connection to Database

To configure connection to database you must edit the config.json file located at server/config.json. For the database that we created we used the information below to connect.

```
server > {} config.json > ...  
1 {  
2   "user": "postgres",  
3   "host": "127.0.0.1",  
4   "database": "postgres",  
5   "password": "postgres",  
6   "port": "5432"  
7 }
```

2.3 Installing Dependencies

To install all required dependencies we execute "npm install" on the server directory. Optionally, you can also install a testing run-time called nodemon by executing "sudo npm install -g nodemon".

2.4 Launching the Node Server

To launch the server you can type "node server.js" on the server directory or if you have installed nodemon you can run "npm run dev". A running server can be seen below. The server is designed to run on development port tcp/8080.

```
alexstolt@alexstolt:~/Desktop/Database Project/server$ node server.js  
Listening at Port: 8080  
█
```

If an error arises at server launch the port might be already in use by another process. To solve this problem the process already running at port tcp/8080 must be terminated. This can be done by executing "fuser -k 8080/tcp" and then retrying to launch the server.

3 React.js Server

3.1 Requirements

1. Node.js Installed (We used Node.js v14.16.0)
2. npm (We used npm v6.14.11)

3.2 Installing Dependencies

To install all required dependencies we execute "npm install" on the client directory.

3.3 Launching the React Server

To launch the server you can type "npm start" on the client directory. When the react server launches a new browser tab will be opened displaying the front-end of our application. The react client uses the predefined port 8080 to connect to the node server.

3.4 Viewing Responses

Browser's "Inspect Element" and "Console" must be used to inspect the responses from the server when "Insert" or "Delete" operations are performed on the database. An example of a university registration can be seen below.

University Database Project

[Universities](#)[Professors](#)[Students](#)[Courses](#)

Universities

[View Available Universities](#)[Register a University](#)

Register a University

NameUTH

CountryGreece

RegionThessaly

Submit

DevTools - localhost:3000/universities/register

ElementsConsoleSourcesNetworkPerformance

topFilterDefault levels1 Issue: 1

UniversityRegister.js:21

> {name_pk: 'UTH', country: 'Greece', region: 'Thessaly', professors: 0, students: 0, ...}

If the university already exists on the database the output below would be displayed.

```
▶ Object UniversityRegister.js:21
✖ ▶ PUT http://127.0.0.1:8080/universities/register 400 (Bad Request) UniversityRegister.js:14
▶ {message: 'duplicate key value violates unique constraint "universities_pkey"'} UniversityRegister.js:21
```