

# CM3035 Advanced Web Development

Lesson 3

Created by Ben Gay

# Django and Database

1. **Install and configure Postgres**
2. **Install dependencies**

# Download Postgre

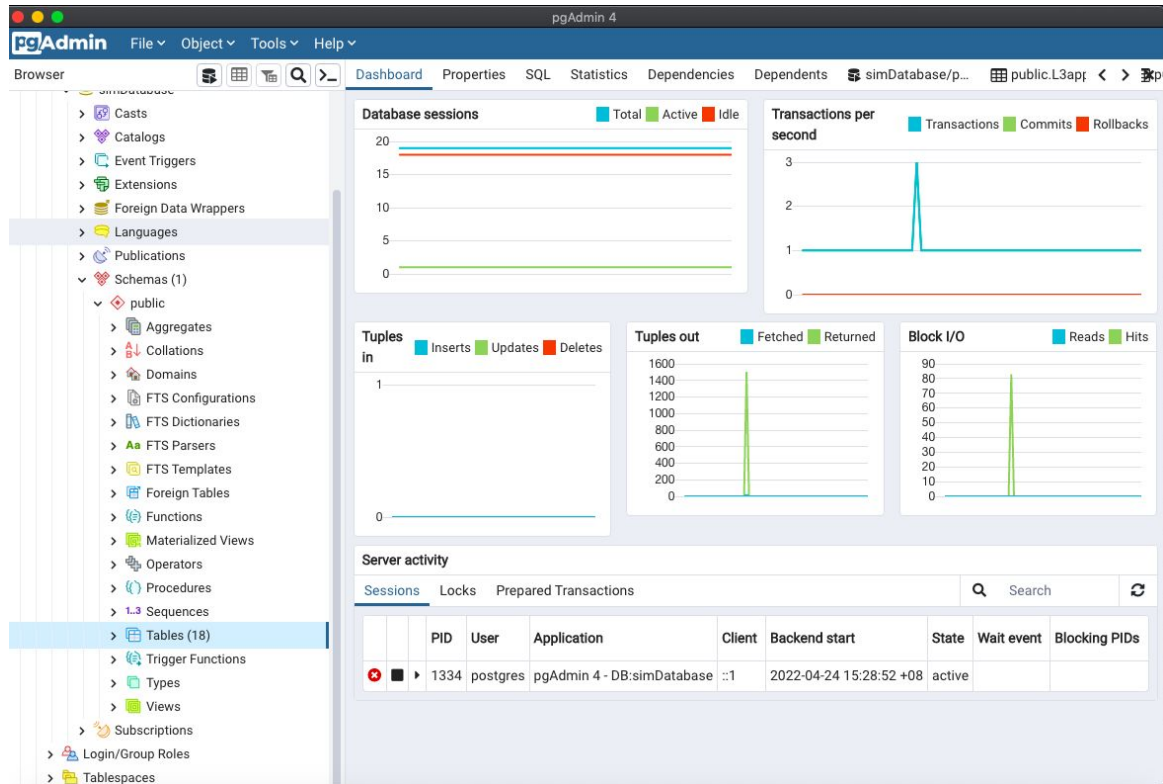
<https://www.postgresql.org/download/>

Video Guide:

<https://www.youtube.com/watch?v=fZQI7nBu32M>

Ensure to install PG Admin too!

# Postgre SQL PG Admin



# Install Dependencies

1. Activate to your Virtual Environment

2. `pip3 install psycopg2`

This is the plugin to interface between Django and PostgreSQL

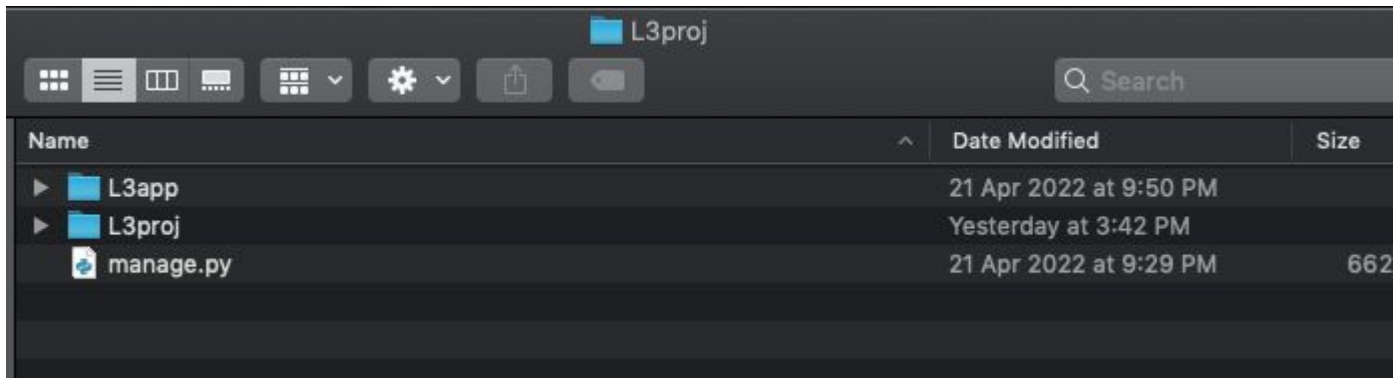
```
[^C(myvenv) (base) Ben:L3proj ben$ pip3 list
Package            Version
-----
asgiref             3.5.0
backports.zoneinfo  0.2.1
Django              4.0.3
homebrew            0.2.5
pip                 22.0.4
psycopg             3.0.11
psycopg2-binary     2.8.4
schedule            1.1.0
setuptools          49.2.1
sqlparse            0.4.2
(myvenv) (base) Ben:L3proj ben$
```

# Let's Begin

1. Create a New project called **L3proj**

(same location as Lesson 2)

2. Create a New Django Application called **L3app** inside the L3proj folder



# Django connect to Postgre

L3proj/L3proj/settings.py

```
settings.py — ~/Desktop/djangoVenv/L3proj/L3proj

settings.py
71 WSGI_APPLICATION = 'L3proj.wsgi.application'
72
73
74 # Database
75 # https://docs.djangoproject.com/en/4.0/ref/settings/#databases
76
77 DATABASES = {
78     'default': {
79         #'ENGINE': 'django.db.backends.sqlite3',
80         #'NAME': BASE_DIR / 'db.sqlite3',
81         'ENGINE': 'django.db.backends.postgresql',
82         'NAME': 'simDatabase',
83         'USER': 'ben',
84         'PASSWORD': 'password',
85         'HOST': 'localhost',
86         'PORT': '5433',
87     }
88 }
89
```

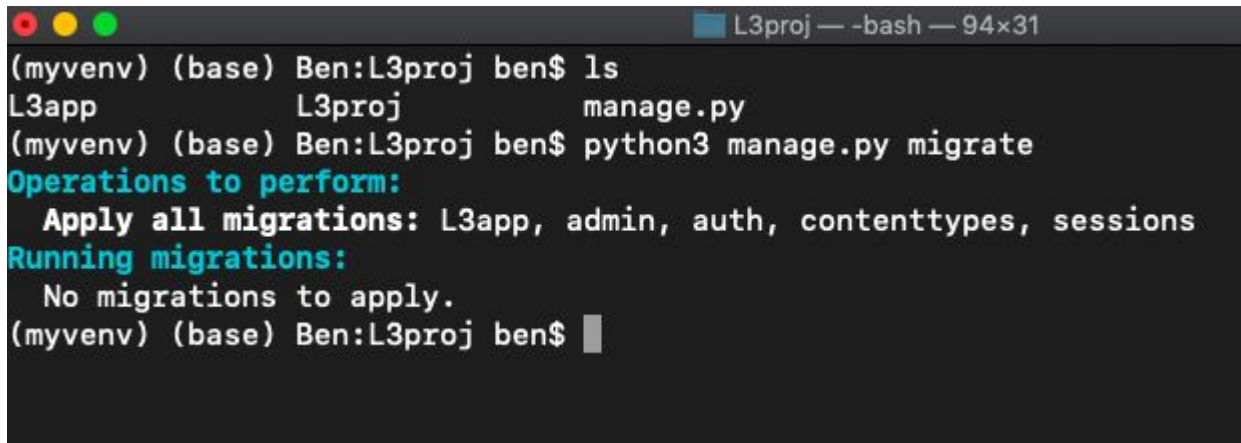
```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'L3app.apps.L3AppConfig',
]
```

# Django connect to Postgre

Test connection inside L3proj folder:

```
python3 manage.py migrate
```

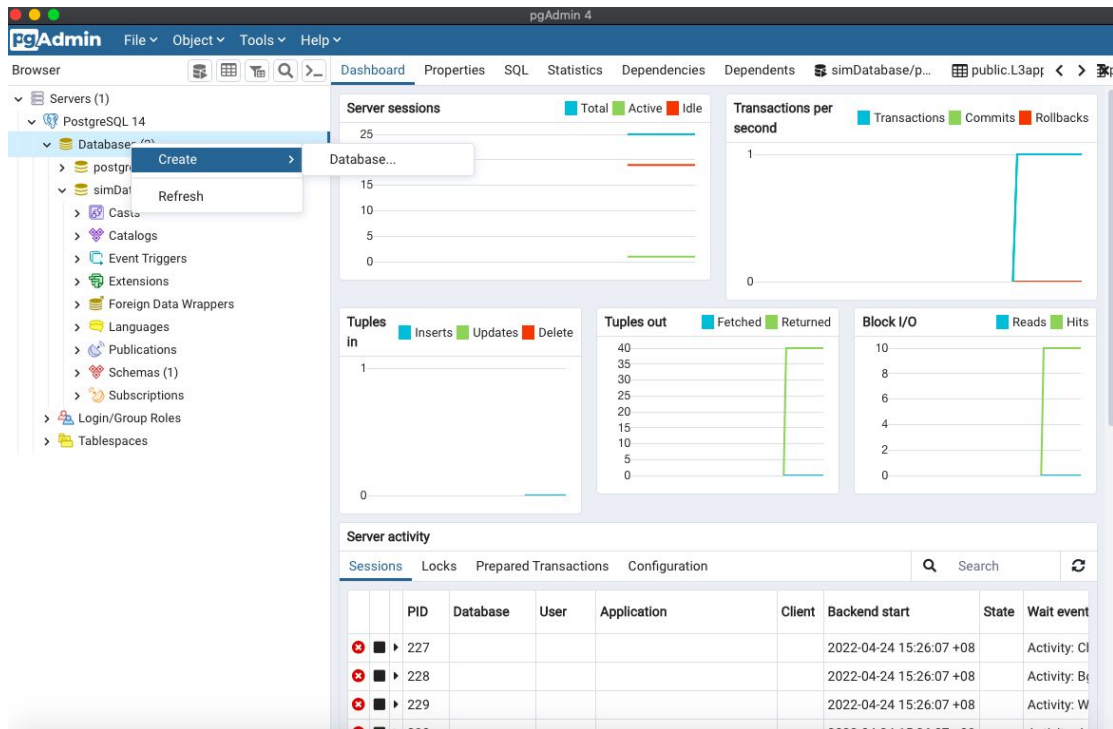
A terminal window titled "L3proj — -bash — 94x31" with standard macOS window controls (red, yellow, green buttons). The terminal shows a user named Ben in the directory L3proj. They run 'ls' and see 'L3app', 'L3proj', and 'manage.py'. Then they run 'python3 manage.py migrate'. The output shows 'Operations to perform:' followed by 'Apply all migrations: L3app, admin, auth, contenttypes, sessions'. Then 'Running migrations:' followed by 'No migrations to apply.' The prompt returns to '(myvenv) (base) Ben:L3proj ben\$' with a cursor.

```
(myvenv) (base) Ben:L3proj ben$ ls
L3app          L3proj          manage.py
(myvenv) (base) Ben:L3proj ben$ python3 manage.py migrate
Operations to perform:
  Apply all migrations: L3app, admin, auth, contenttypes, sessions
Running migrations:
  No migrations to apply.
(myvenv) (base) Ben:L3proj ben$
```



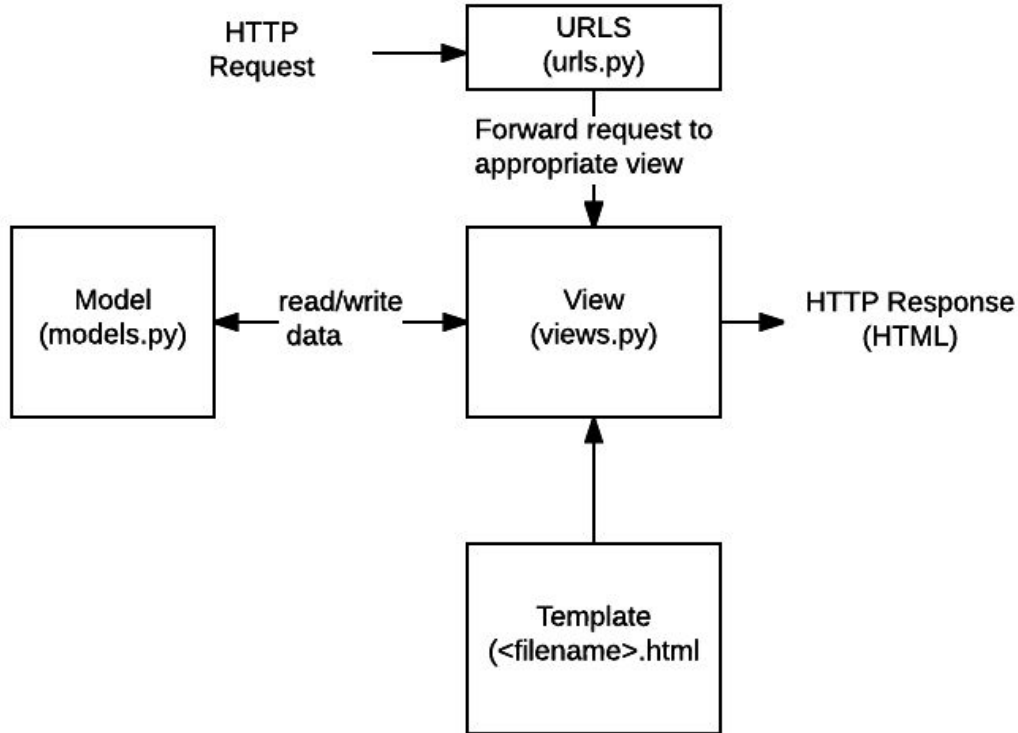
# Create a new database

Use pgAdmin to create a New database called **simDatabase**



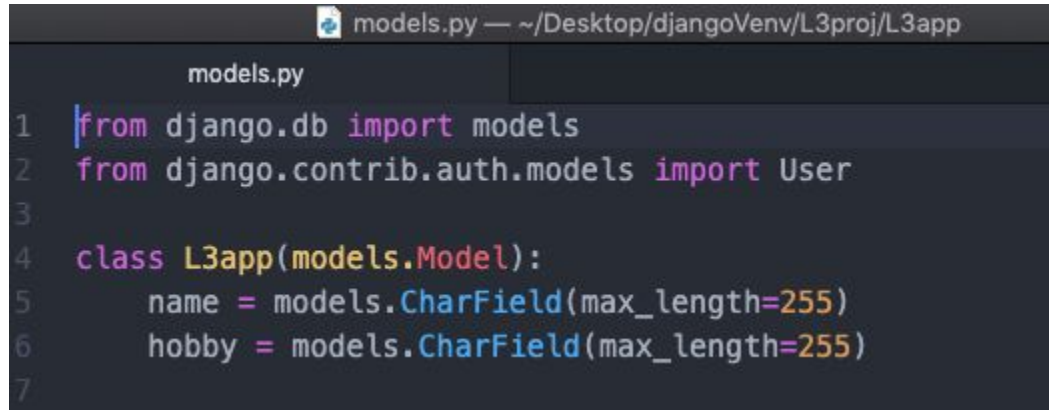
# Django Framework

L3proj/L3app/models.py



# Create a new table with models

L3proj/L3app/models.py



The screenshot shows a code editor window titled "models.py — ~/Desktop/djangoVenv/L3proj/L3app". The editor displays the following Python code:

```
models.py
1 from django.db import models
2 from django.contrib.auth.models import User
3
4 class L3app(models.Model):
5     name = models.CharField(max_length=255)
6     hobby = models.CharField(max_length=255)
7
```

# Table view

## L3proj/L3app/models.py

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays a tree view of the database schema. The 'public' schema is expanded, and the 'Tables (18)' folder is selected. The table 'L3app\_l3app' is highlighted. The main pane shows the 'Query Editor' with the following SQL query:

```
1 SELECT * FROM public."L3app_l3app"
2 ORDER BY id ASC
```

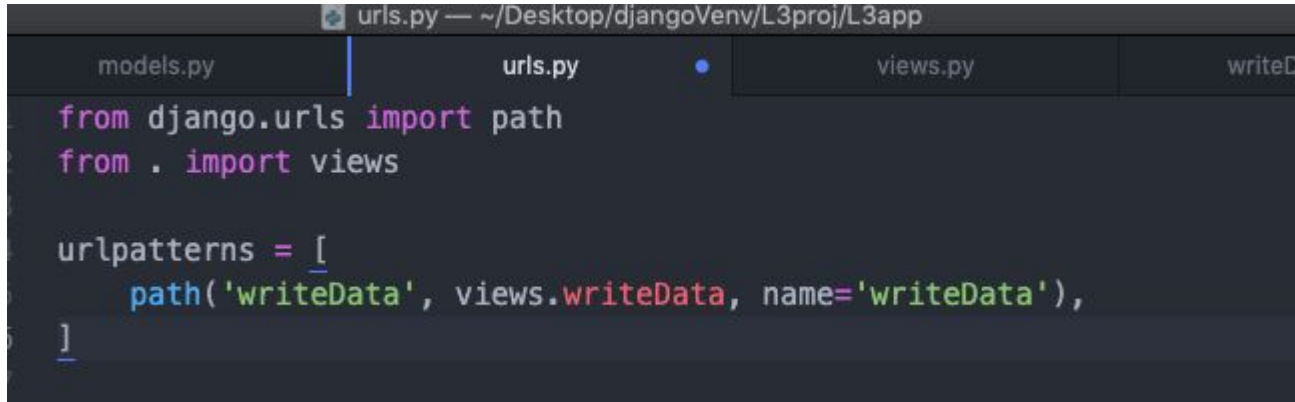
Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format. The table has three columns: 'id' (bigint), 'name' (character varying (255)), and 'hobby' (character varying (255)). The results are as follows:

id	name	hobby
1	john	tennis
2	Tom	camping
3	James	jogging
4	Bella	Basketball

# Insert new row into table

L3proj/L3app/url.py

Point path to function in views.py



The screenshot shows a code editor with a dark theme. The title bar indicates the file is 'urls.py' located at '~/Desktop/djangoVenv/L3proj/L3app'. The editor has four tabs: 'models.py', 'urls.py' (which is active and has a blue dot), 'views.py', and 'writeD'. The code in 'urls.py' is as follows:

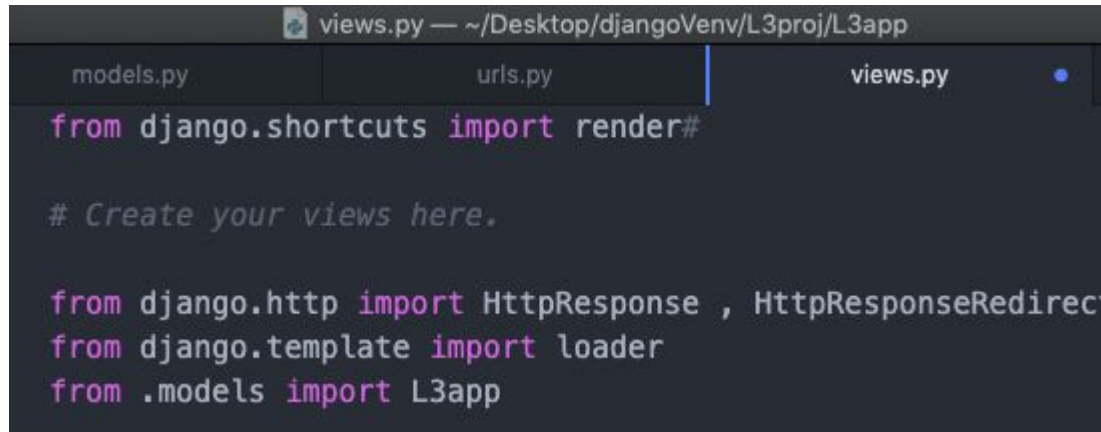
```
from django.urls import path
from . import views

urlpatterns = [
    path('writeData', views.writeData, name='writeData'),
]
```

# Insert new row into table

L3proj/L3app/views.py

Import L3app table from Postgre into views.py



```
views.py — ~/Desktop/djangoVenv/L3proj/L3app  
models.py  urls.py  views.py  
from django.shortcuts import render#  
  
# Create your views here.  
  
from django.http import HttpResponse , HttpResponseRedirect  
from django.template import loader  
from .models import L3app
```

# Insert new row into table

L3proj/L3app/views.py

Create function to insert into table

```
def writeData(request):  
    template = loader.get_template('writeData.html')  
    L3appRecord = L3app(name='Sam', hobby='fishing')  
    L3appRecord.save()  
    queryWritten = L3app.objects.all().values()  
    context = {  
        'queryWritten': queryWritten,  
    }  
    return HttpResponse(template.render(context, request))
```

# Preview on browser

L3proj/L3app/templates/writeData.html

```
1  <!DOCTYPE html>
2  <html>
3  <body>
4
5  <h1>Inserted into database</h1>
6  <p>{{ queryWritten }}</p>
7
8  </body>
9  </html>
10
```



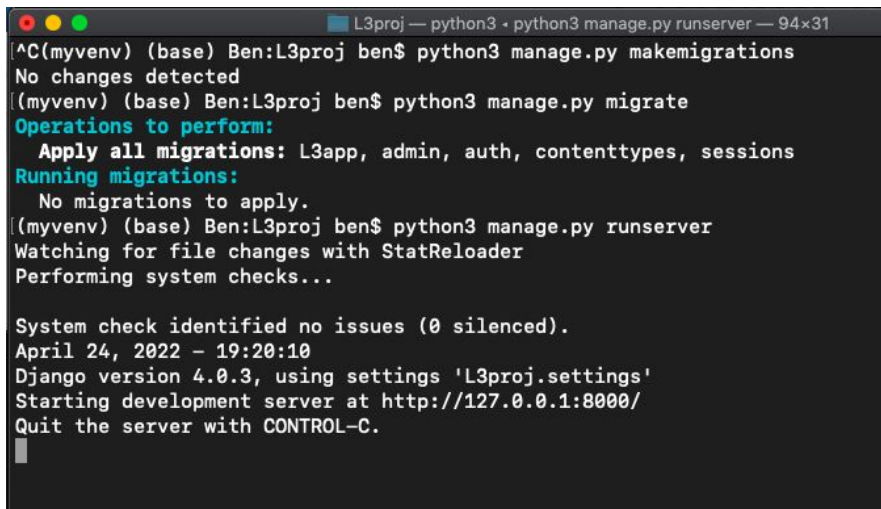
# Preview on browser

## Update Database

`python3 manage.py makemigrations`

`python3 manage.py migrate`

`python3 manage.py runserver`

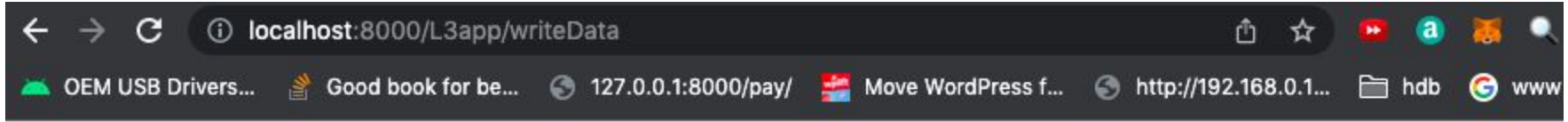
A terminal window with a dark background and light text. The title bar at the top reads "L3proj — python3 · python3 manage.py runserver — 94x31". The terminal shows the following sequence of commands and output:

```
^C(myvenv) (base) Ben:L3proj ben$ python3 manage.py makemigrations
No changes detected
(myvenv) (base) Ben:L3proj ben$ python3 manage.py migrate
Operations to perform:
  Apply all migrations: L3app, admin, auth, contenttypes, sessions
Running migrations:
  No migrations to apply.
(myvenv) (base) Ben:L3proj ben$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
April 24, 2022 - 19:20:10
Django version 4.0.3, using settings 'L3proj.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

# Preview on browser

<http://localhost:8000/L3app/writeData>



## Inserted into database

```
<QuerySet [{ 'id': 1, 'name': 'john', 'hobby': 'tennis'}, { 'id': 31, 'name': 'Sam', 'hobby': 'fishing'}]>
```

# Result in Postgre

The screenshot shows the pgAdmin 4 web interface. On the left, the 'Object browser' tree is expanded to 'public' > 'Tables (18)', with 'L3app\_l3app' selected. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 SELECT * FROM public."L3app_l3app"  
2 ORDER BY id ASC
```

Below the query editor, the 'Data Output' tab shows the results of the query in a table format. The table has three columns: 'id' (integer), 'name' (character varying), and 'hobby' (character varying). The results are as follows:

id	name	hobby
1	John	tennis
2	Sam	fishing

# Relational Database

# Relational Database

One to One field

L3proj/L3app/models.py

```
class School(models.Model):
    name = models.CharField(max_length=70)

    def __str__(self):
        return str(self.name)

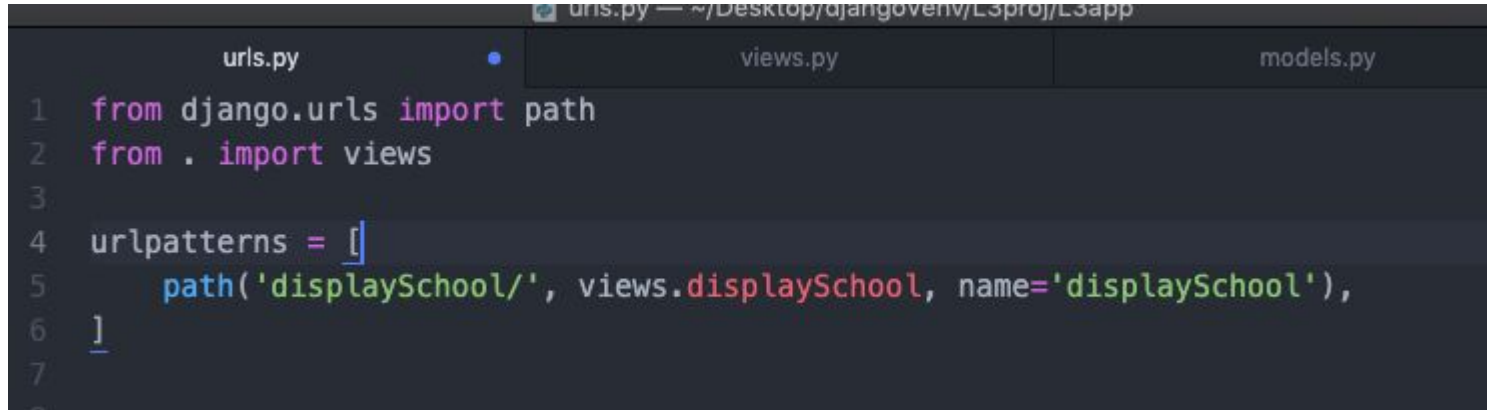
class Principal(models.Model):
    school = models.OneToOneField(
        School,
        on_delete=models.CASCADE,
        primary_key=True,
    )

    name = models.CharField(max_length=70)

    def __str__(self):
        return str(self.name)
```

# Relational Database

L3proj/L3app/urls.py

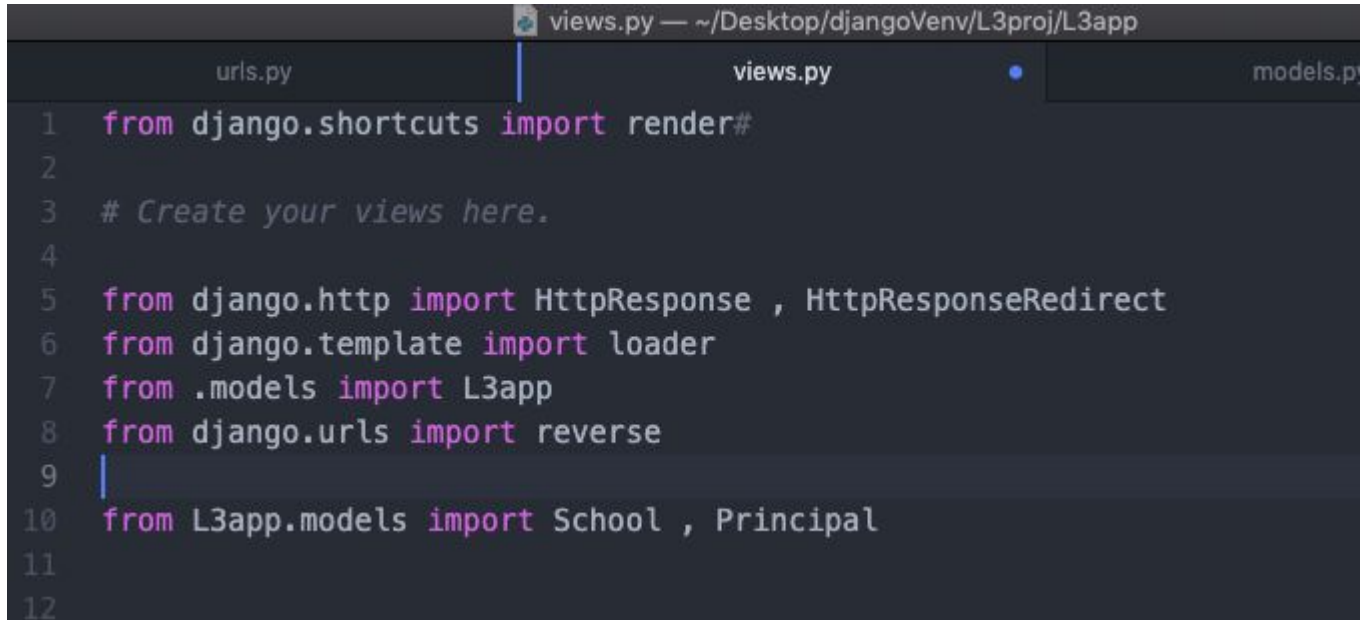


The image shows a code editor window with three tabs: 'urls.py', 'views.py', and 'models.py'. The 'urls.py' tab is active and displays the following Python code:

```
1 from django.urls import path
2 from . import views
3
4 urlpatterns = [
5     path('displaySchool/', views.displaySchool, name='displaySchool'),
6 ]
```

# Import table into views.py

L3proj/L3app/views.py



The screenshot shows a code editor with a dark theme. The title bar at the top reads "views.py — ~/Desktop/djangoVenv/L3proj/L3app". Below the title bar, there are three tabs: "urls.py", "views.py" (which is active and has a blue dot), and "models.p". The code in the "views.py" tab is as follows:

```
1  from django.shortcuts import render#
2
3  # Create your views here.
4
5  from django.http import HttpResponse , HttpResponseRedirect
6  from django.template import loader
7  from .models import L3app
8  from django.urls import reverse
9
10 from L3app.models import School , Principal
11
12
```

# Insert data into tables

## L3proj/L3app/views.py

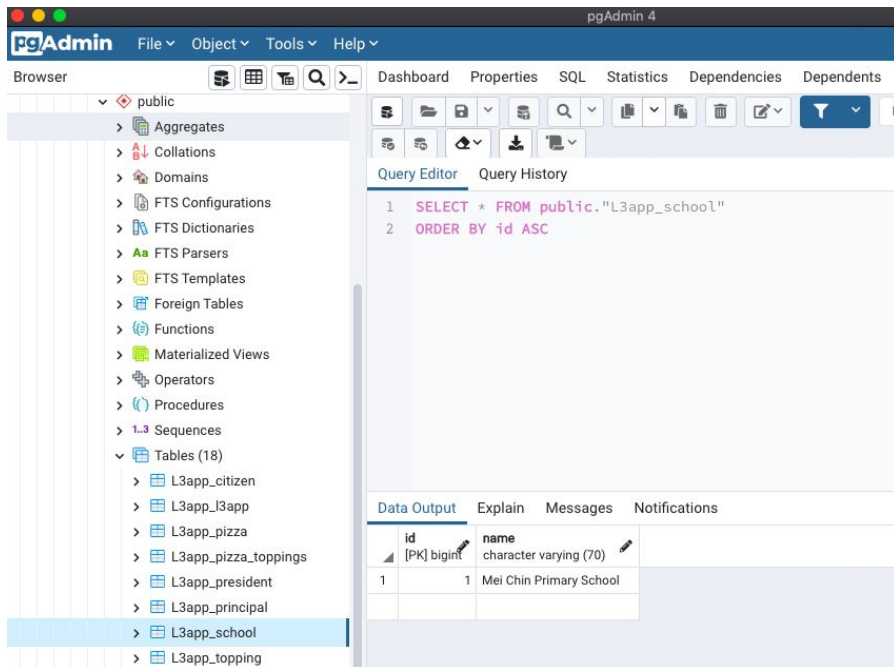
```
views.py — ~/Desktop/djangoVenv/L3proj/L3app
urls.py  views.py  models.py

139
140
141 def displaySchool(request):
142     *****create new data*****
143     mei_chin_primary_school = School.objects.create(name = "Mei Chin Primary School")
144     mei_chin_primary_school.save()
145     mr_tan = Principal.objects.create(school_id="1",name="Mr Tan")
146     mr_tan.save()
147
148     *****get existing data*****
149     mei_chin_primary_school = School.objects.get(name = "Mei Chin Primary School")
150     principalData = mei_chin_primary_school.principal
151
152     #school_principal = Principal.objects.get(name = "Mr Tan")
153     #principalData = school_principal.school
154     *****
155
156     template = loader.get_template('displaySchool.html')
157     context = {
158         'principalData': principalData,
159     }
160     return HttpResponse(template.render(context, request))
161
162
```



# Data inserted into table

## School



The screenshot shows the pgAdmin 4 web interface. On the left, the 'Browser' pane displays a tree view of database objects under the 'public' schema. The 'Tables (18)' folder is expanded, and 'L3app\_school' is selected. The main pane shows the 'Query Editor' with the following SQL query:

```
1 SELECT * FROM public."L3app_school"  
2 ORDER BY id ASC
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

id	name
1	Mei Chin Primary School

# Data inserted into table

## Principal

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays a tree view of database objects under the 'public' schema. The 'Tables (18)' folder is expanded, and 'L3app\_principal' is selected. The main pane shows the 'Query Editor' with the following SQL query:

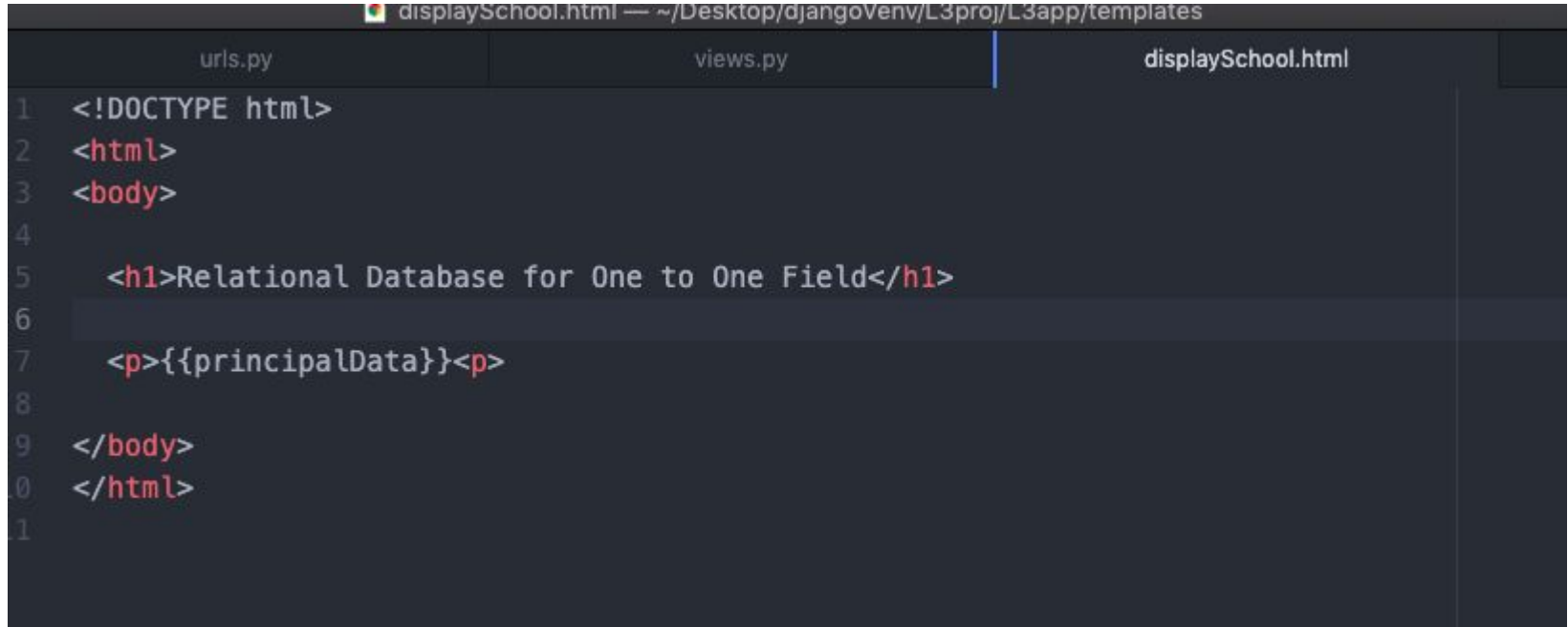
```
1 SELECT * FROM public."L3app_principal"  
2 ORDER BY school_id ASC
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table:

school_id	name
1	Mr Tan

# Template html file

L3proj/L3app/templates/displaySchool.html

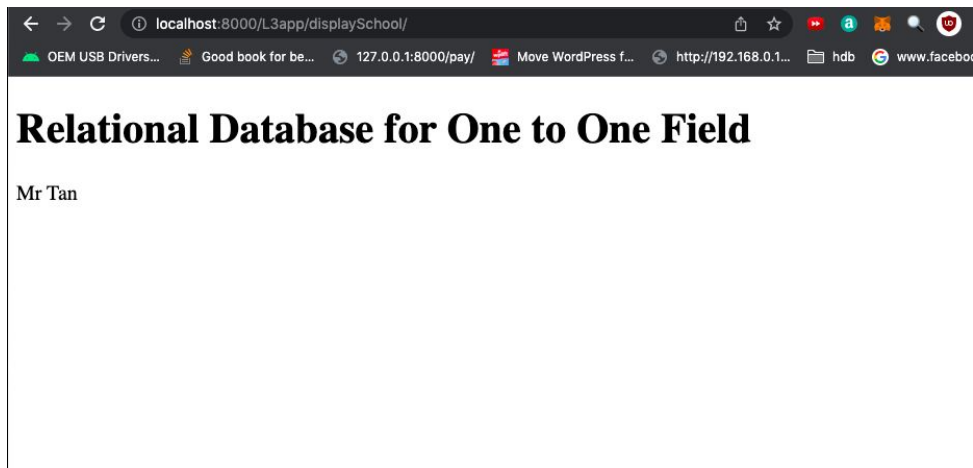
A screenshot of a code editor window. The title bar shows the file path: displaySchool.html — ~/Desktop/djangovenv/L3proj/L3app/templates. The editor has three tabs: urls.py, views.py, and displaySchool.html. The displaySchool.html tab is active, showing the following HTML code:

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4
5     <h1>Relational Database for One to One Field</h1>
6
7     <p>{{principalData}}<p>
8
9 </body>
10 </html>
11
```

# Browser Result

```
mei_chin_primary_school = School.objects.get(name = "Mei Chin Primary School")  
principalData = mei_chin_primary_school.principal
```

**<http://localhost:8000/L3app/displaySchool/>**



End of Lesson 3