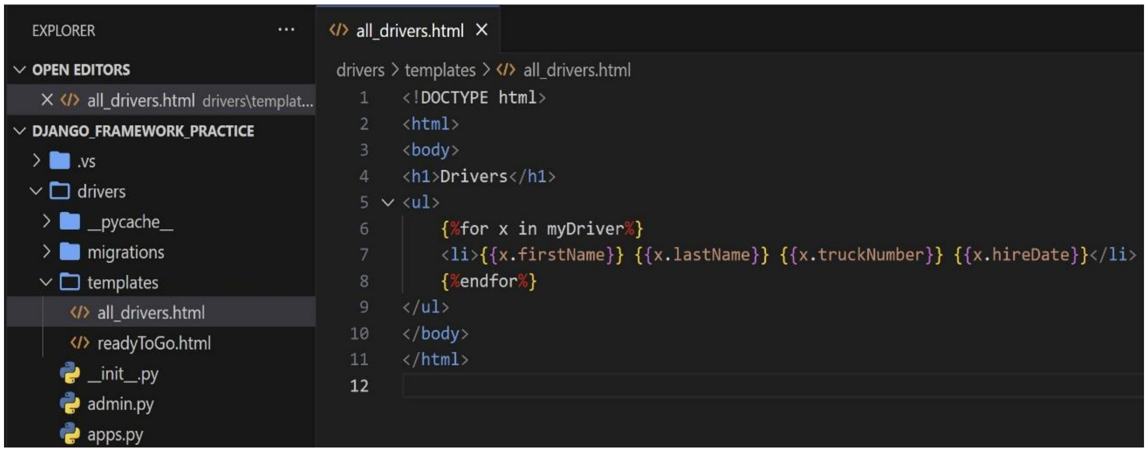
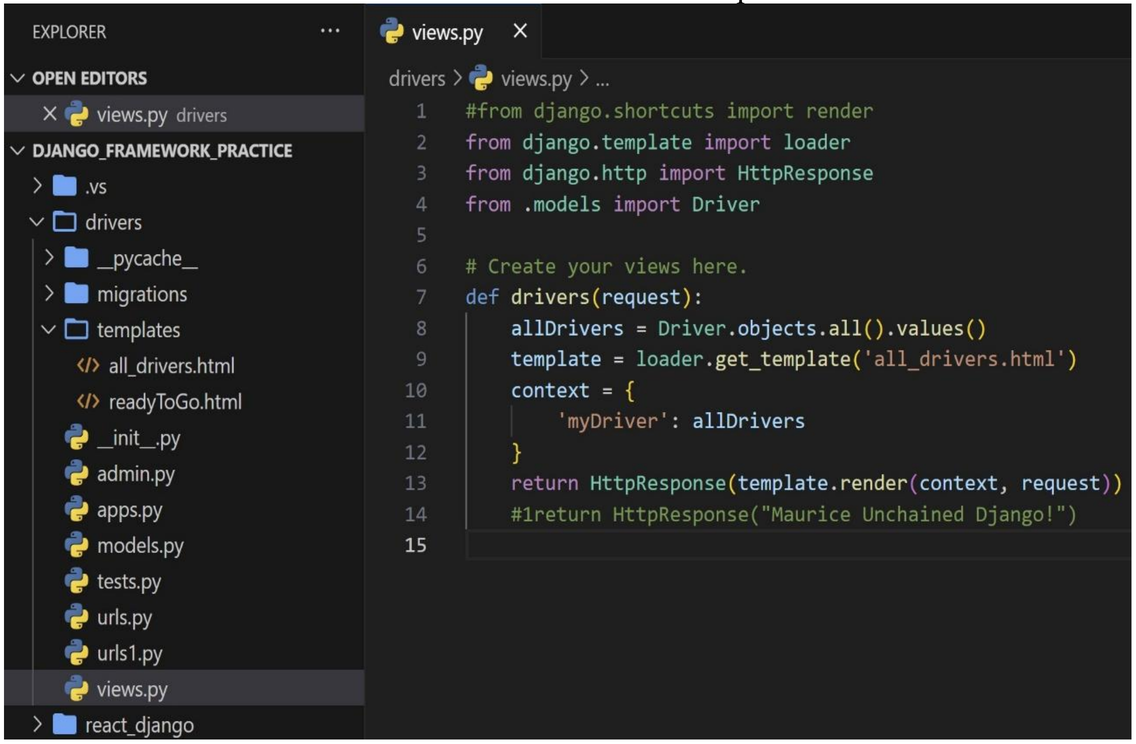
Got Goat Chapter 4 Displaying Database Data

We have created the model schema in the Driver app, the table in database, and entered records. We also discovered how to perform popular CRUD operations. In this chapter, we will create a template and view for displaying dynamic content on the screen.

1. Create HTML file
   a. Navigate to the drivers app/folder and create a new HTML file inside the templates folder similar to below.
   b. 

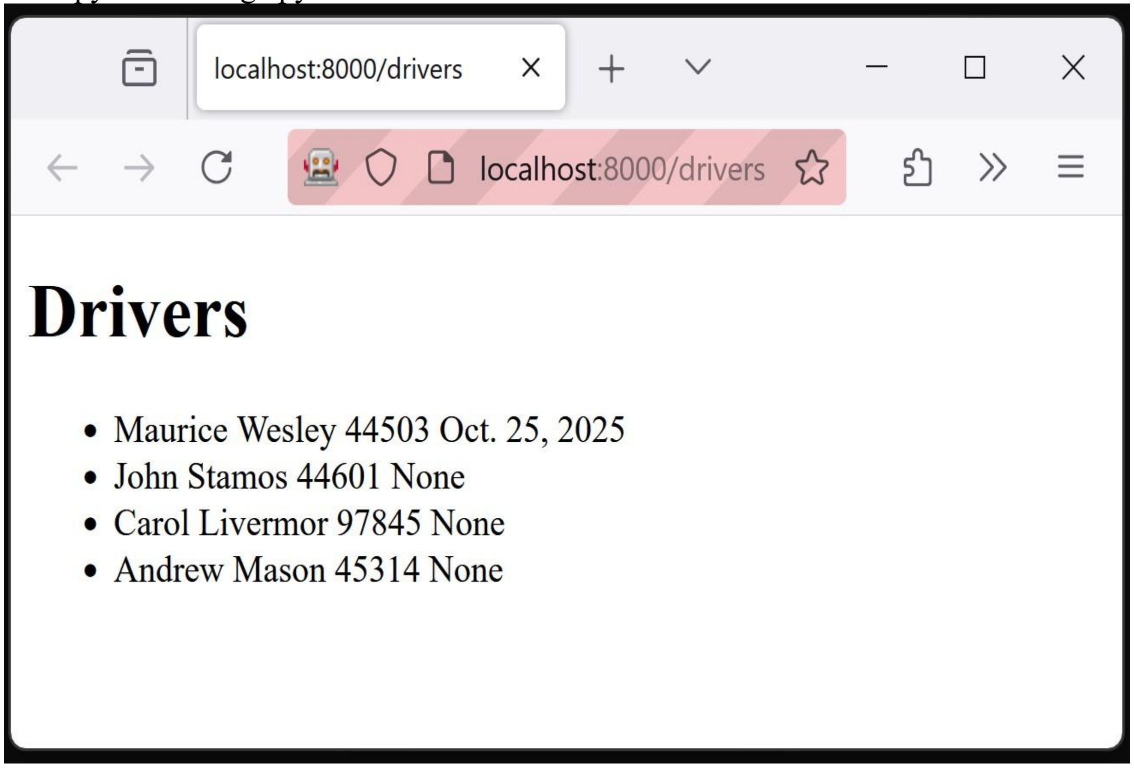
```
1   <!DOCTYPE html>
2   <html>
3   <body>
4   <h1>Drivers</h1>
5   <ul>
6       {%for x in myDriver%}
7       <li>{{x.firstName}} {{x.lastName}} {{x.truckNumber}} {{x.hireDate}}</li>
8       {%endfor%}
9   </ul>
10  </body>
11  </html>
12
```

      i. Line 6: The double "%" instructs the view to perform programming logic inside the brackets. More specifically, the iterator will list the specified attributes of each record.
2. Update View
   a. We need to make the database model available to the template.
   b. 

```
1   #from django.shortcuts import render
2   from django.template import loader
3   from django.http import HttpResponse
4   from .models import Driver
5
6   # Create your views here.
7   def drivers(request):
8       allDrivers = Driver.objects.all().values()
9       template = loader.get_template('all_drivers.html')
10      context = {
11          'myDriver': allDrivers
12      }
13      return HttpResponse(template.render(context, request))
14      #1return HttpResponse("Maurice Unchained Django!")
15
```

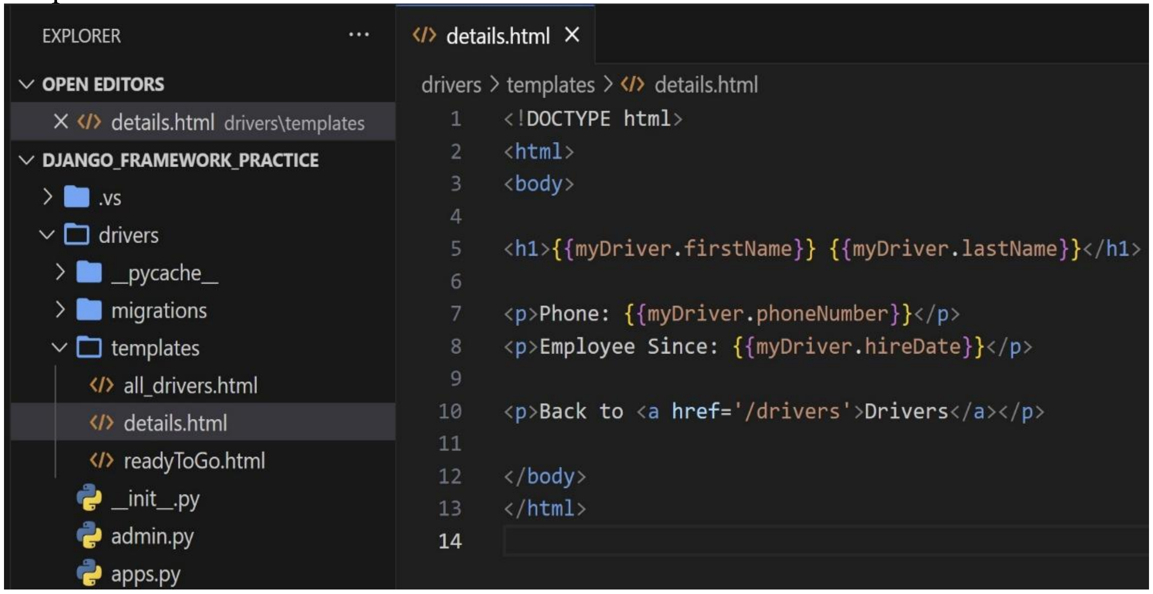
      i. Line 4: We are importing the Driver model from driver app models folder
      ii. Line 8: We are assigning the values of the model to a variable passed in response

iii. Line 11: The programming logic (myDriver) has to reference an object(allDrivers) for the iterator (for loop in html).

3. Display Records
   a. Run: python manage.py runserver
   b. 

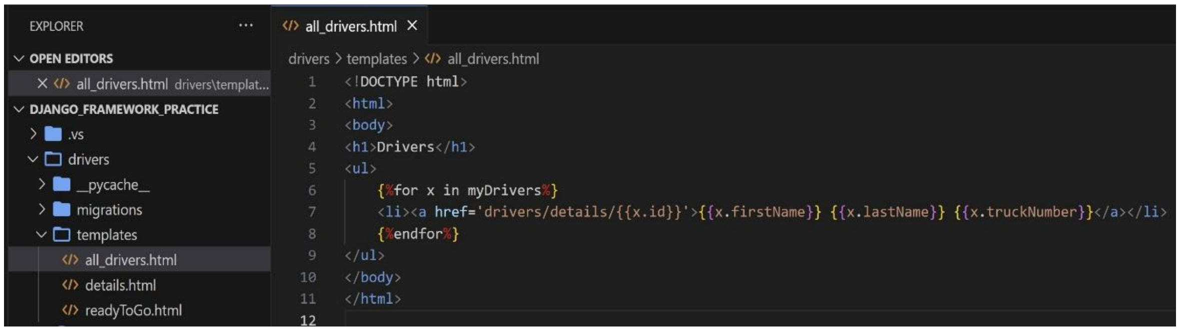
Drivers

- Maurice Wesley 44503 Oct. 25, 2025
- John Stamos 44601 None
- Carol Livermor 97845 None
- Andrew Mason 45314 None

4. Details HTML
   a. Navigate to the drivers app/folder and create a new HTML file called 'details' inside the templates folder similar to below.
   b. 

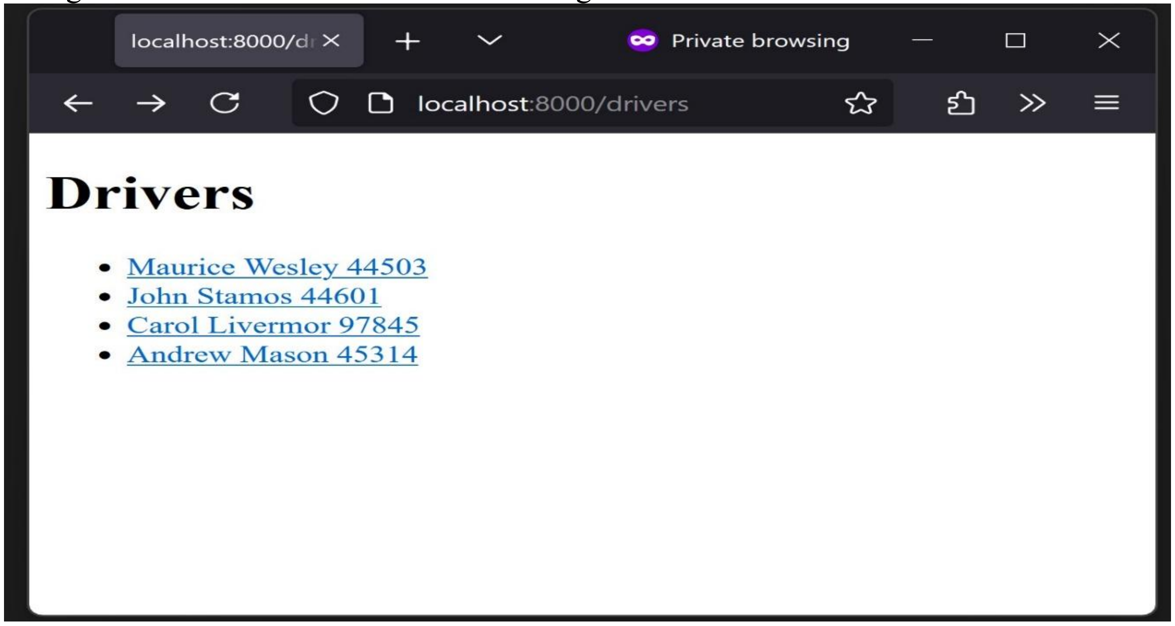
```html
<!DOCTYPE html>
<html>
<body>

<h1>{{myDriver.firstName}} {{myDriver.lastName}}</h1>

<p>Phone: {{myDriver.phoneNumber}}</p>
<p>Employee Since: {{myDriver.hireDate}}</p>

<p>Back to <a href='/drivers'>Drivers</a></p>

</body>
</html>
```

5. Link To Details
   a. Modify the all_drivers html to link the first and last name to the details page

b. 

```
<!DOCTYPE html>
<html>
<body>
<h1>Drivers</h1>
<ul>
    {%for x in myDrivers%}
    <li><a href='drivers/details/{{x.id}}'>{{x.firstName}} {{x.lastName}} {{x.truckNumber}}</a></li>
    {%endfor%}
</ul>
</body>
</html>
```

c. Navigate over to the terminal to see the changes

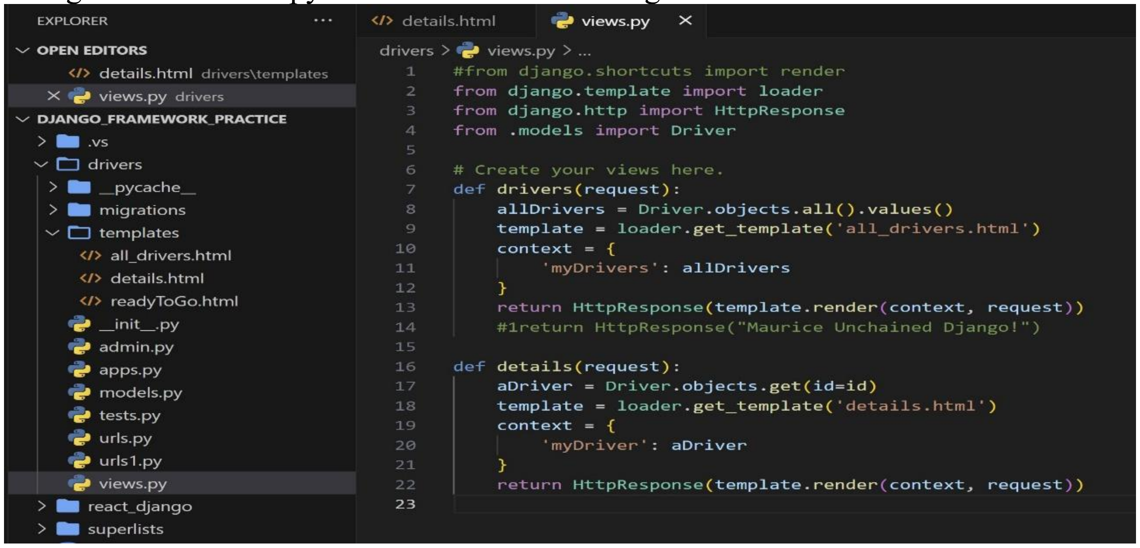
d. 

Drivers

- Maurice Wesley 44503
- John Stamos 44601
- Carol Livermor 97845
- Andrew Mason 45314

e. Clicking on the link will generate an error. We have not set up the routing to pass the data from the model to a view.

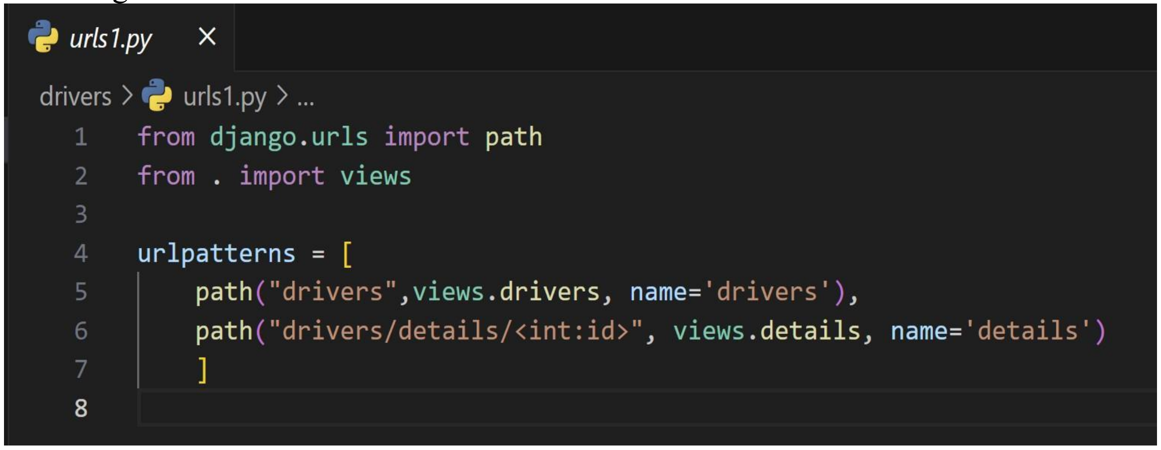
6. Create Detail View
   a. Navigate to the views.py file and add the following

b. 

```python
#from django.shortcuts import render
from django.template import loader
from django.http import HttpResponse
from .models import Driver

# Create your views here.
def drivers(request):
    allDrivers = Driver.objects.all().values()
    template = loader.get_template('all_drivers.html')
    context = {
        'myDrivers': allDrivers
    }
    return HttpResponse(template.render(context, request))
    #1return HttpResponse("Maurice Unchained Django!")

def details(request):
    aDriver = Driver.objects.get(id=id)
    template = loader.get_template('details.html')
    context = {
        'myDriver': aDriver
    }
    return HttpResponse(template.render(context, request))
```

7. Add URL Routing

a. The details URL has to link to the correct view. Open the URLs file and add the following.

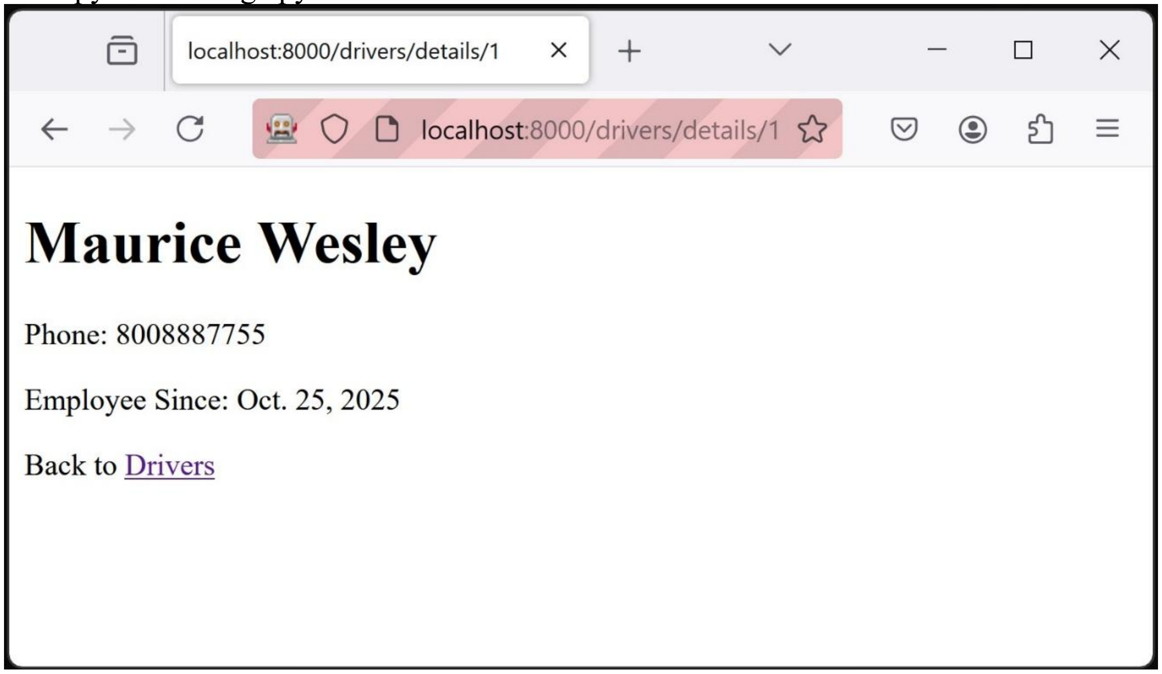

urls1.py

drivers > urls1.py > ...

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

urlpatterns = [
    path("drivers",views.drivers, name='drivers'),
    path("drivers/details/<int:id>", views.details, name='details')
    ]

```

b.

8. Display Drivers and link to details
   a. Run: python manage.py runserver


localhost:8000/drivers/details/1

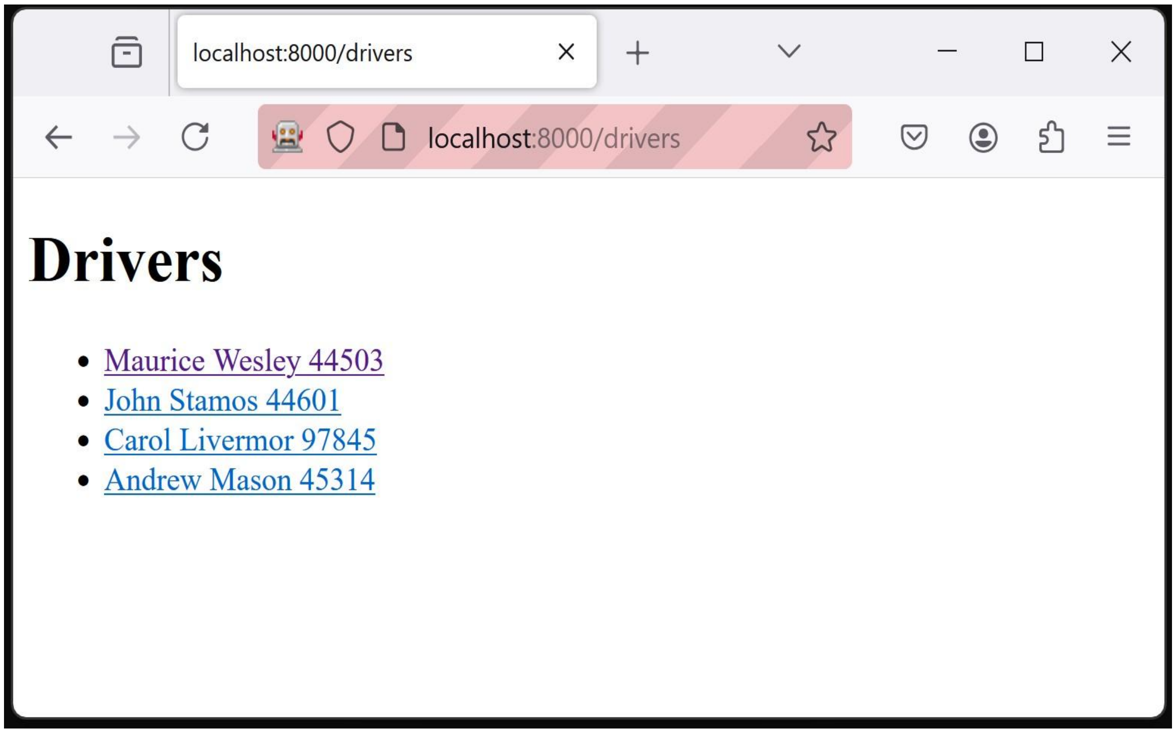
Maurice Wesley

Phone: 8008887755

Employee Since: Oct. 25, 2025

Back to Drivers

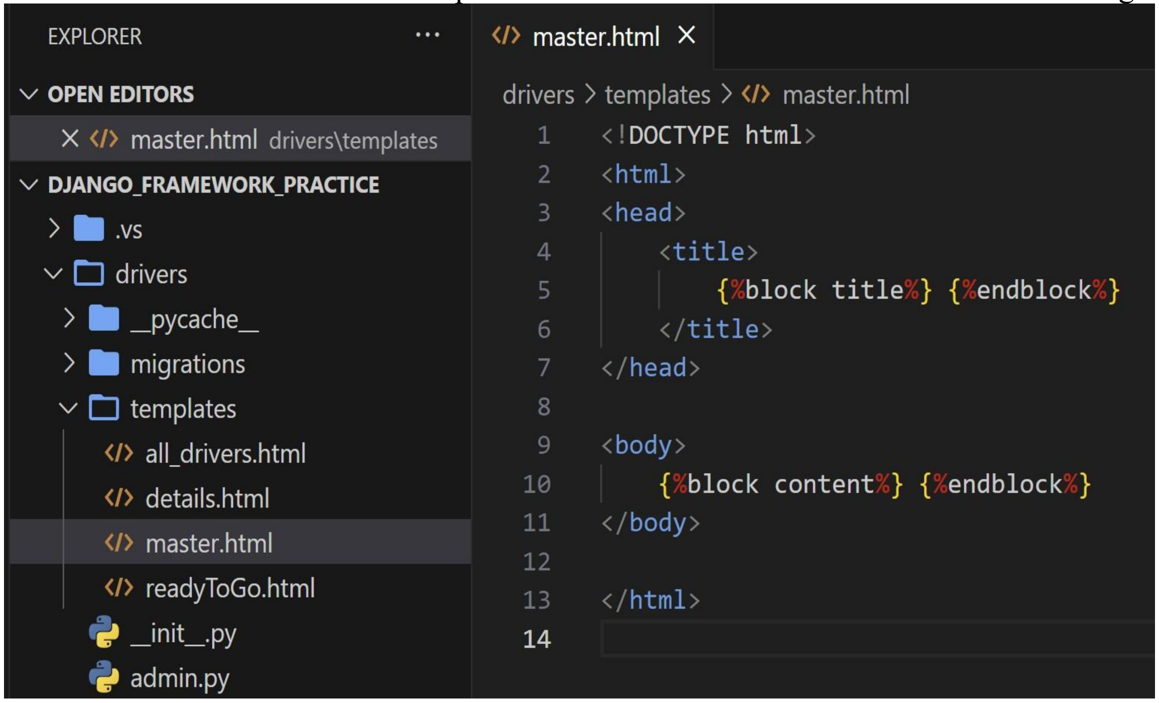
   b.
      i. We can celebrate the above. The URL displays the page and contents. The driver's table attribute values are displayed with the templated format.
   c. Pressing the link 'Drivers' will navigate the user back to the home page.

d. 

Drivers

- Maurice Wesley 44503
- John Stamos 44601
- Carol Livermor 97845
- Andrew Mason 45314

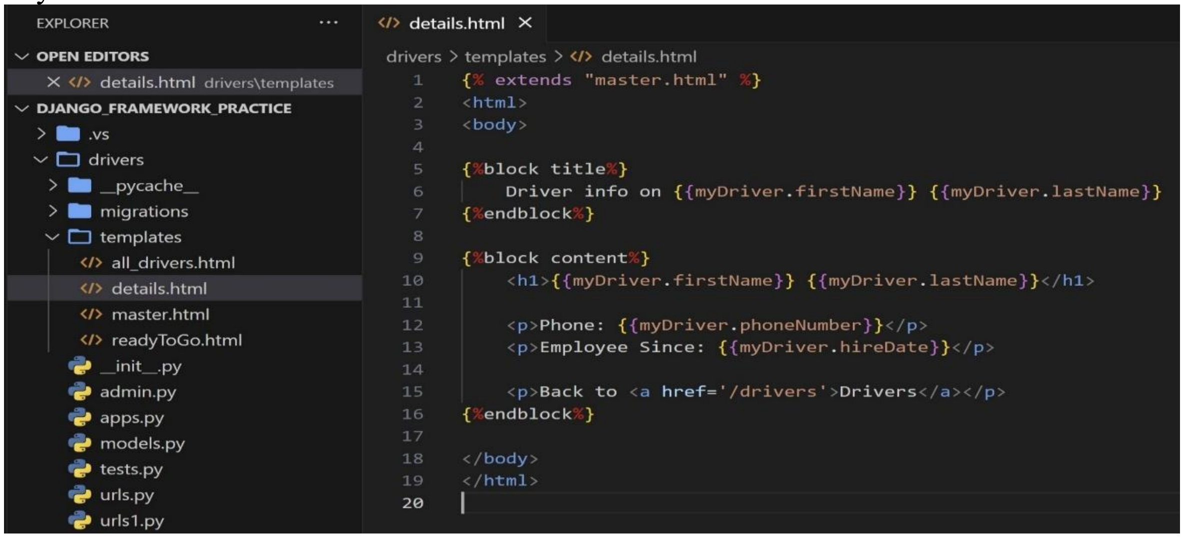
   i. The first driver has a different color meaning that the link has been visited.

9. Master Template
   a. Django provides functionality to wrap similar content into blocks. Both the details HTML and all_drivers HTML have similar structures. We can leverage this similarity into a single template.
   b. Create a new HTML file in the template folder called 'master.html'. Insert the following.
   c. 

```
<!DOCTYPE html>
<html>
<head>
    <title>
        {%block title%} {%endblock%}
    </title>
</head>

<body>
    {%block content%} {%endblock%}
</body>

</html>
```

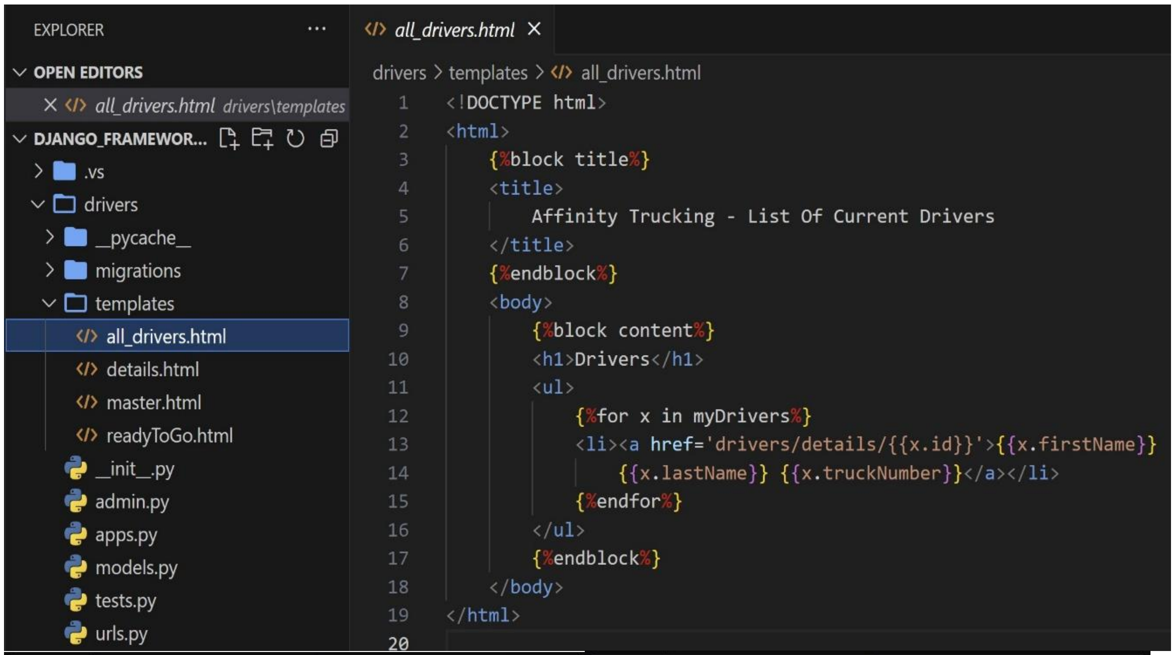
   d. Now that we have the master template. We can wrap the detail and all_drivers HTMLs in programming language blocks. See below.

10. Modify the details HTML

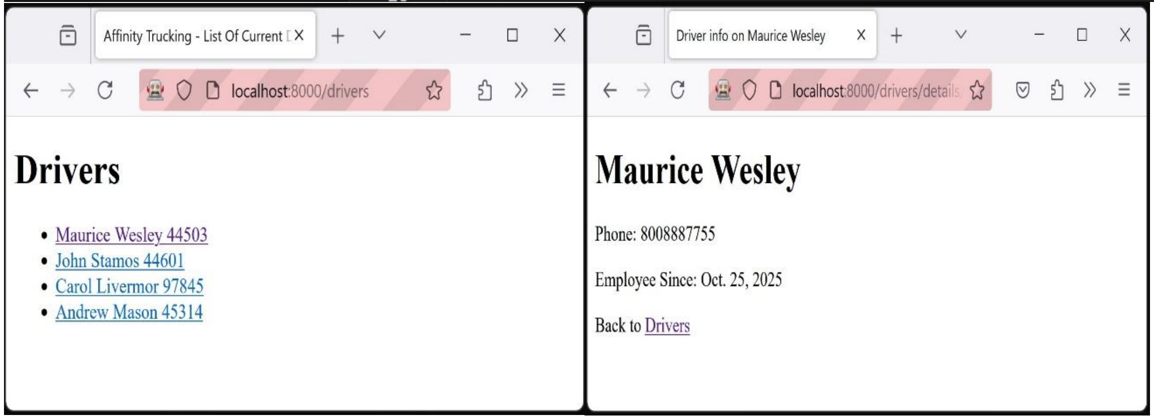
a. 

```html
{% extends "master.html" %}
<html>
<body>

{%block title%}
    Driver info on {{myDriver.firstName}} {{myDriver.lastName}}
{%endblock%}

{%block content%}
    <h1>{{myDriver.firstName}} {{myDriver.lastName}}</h1>

    <p>Phone: {{myDriver.phoneNumber}}</p>
    <p>Employee Since: {{myDriver.hireDate}}</p>

    <p>Back to <a href='/drivers'>Drivers</a></p>
{%endblock%}

</body>
</html>
```

b. The model data output and format are wrapped in blocks. Proceed to edit the all_drivers HTML.

c. 

```html
<!DOCTYPE html>
<html>
    {%block title%}
    <title>
        Affinity Trucking - List Of Current Drivers
    </title>
    {%endblock%}
    <body>
        {%block content%}
        <h1>Drivers</h1>
        <ul>
            {%for x in myDrivers%}
            <li><a href='drivers/details/{{x.id}}'>{{x.firstName}}
                {{x.lastName}} {{x.truckNumber}}</a></li>
            {%endfor%}
        </ul>
        {%endblock%}
    </body>
</html>
```

d. 

Affinity Trucking - List Of Current D
localhost:8000/drivers
Driver info on Maurice Wesley
localhost:8000/drivers/details

Drivers

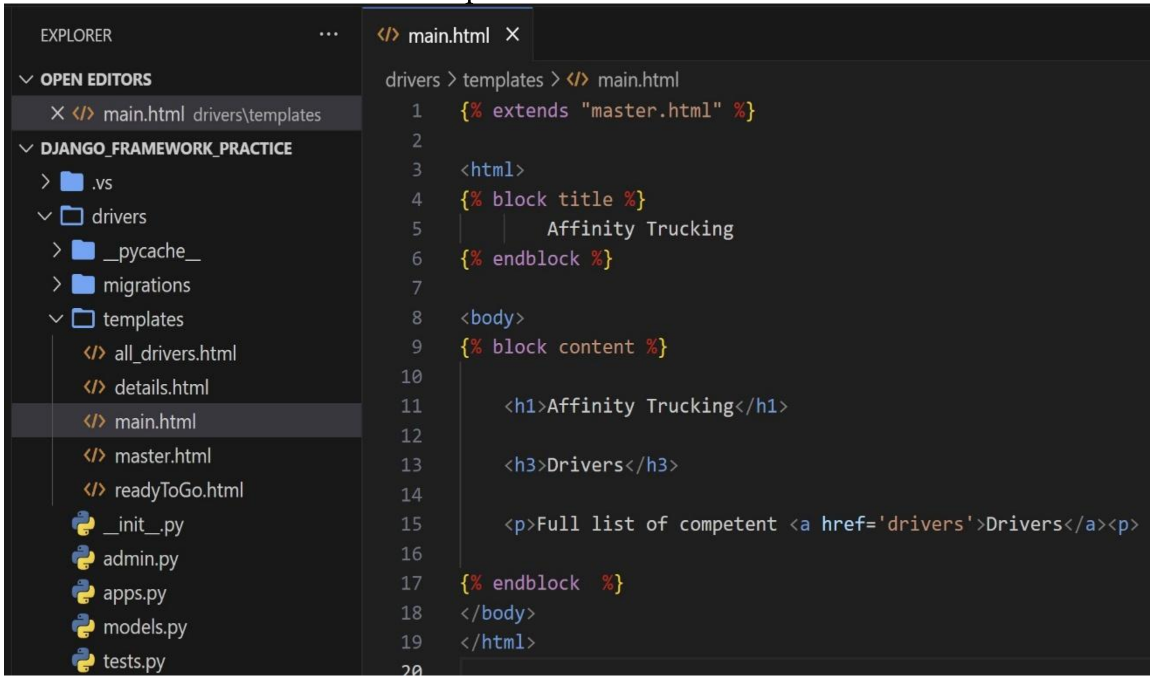
- Maurice Wesley 44503
- John Stamos 44601
- Carol Livermor 97845
- Andrew Mason 45314

Maurice Wesley

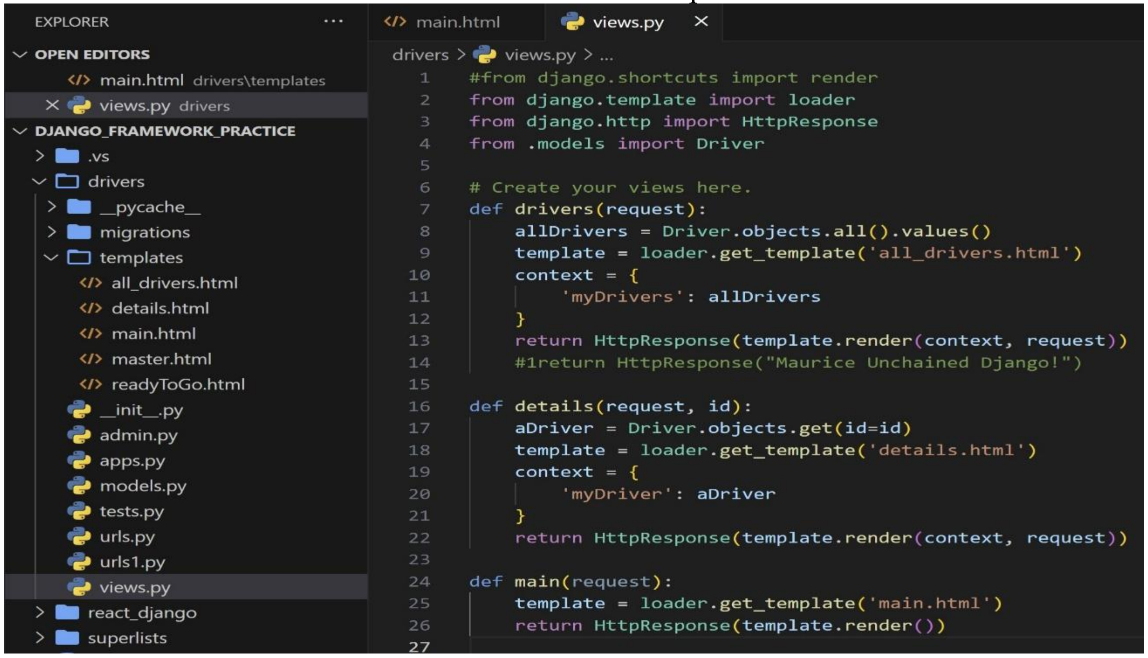
Phone: 8008887755

Employee Since: Oct. 25, 2025

Back to Drivers

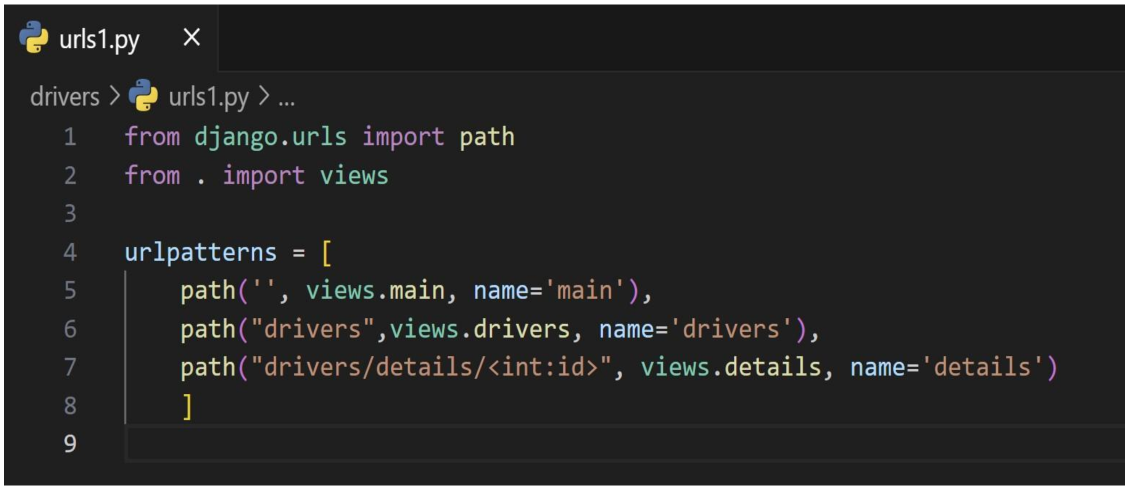
11. Main Index Page
   a. The main landing page will show an error because our current application does not have a main page.
   b. Create a main HTML file in the template folder
   c. 

```
{% extends "master.html" %}

<html>
{% block title %}
        Affinity Trucking
{% endblock %}

<body>
{% block content %}

    <h1>Affinity Trucking</h1>

    <h3>Drivers</h3>

    <p>Full list of competent <a href='drivers'>Drivers</a><p>

{% endblock  %}
</body>
</html>
```

   d. Add a main function to the view to load the main template.
   e. 

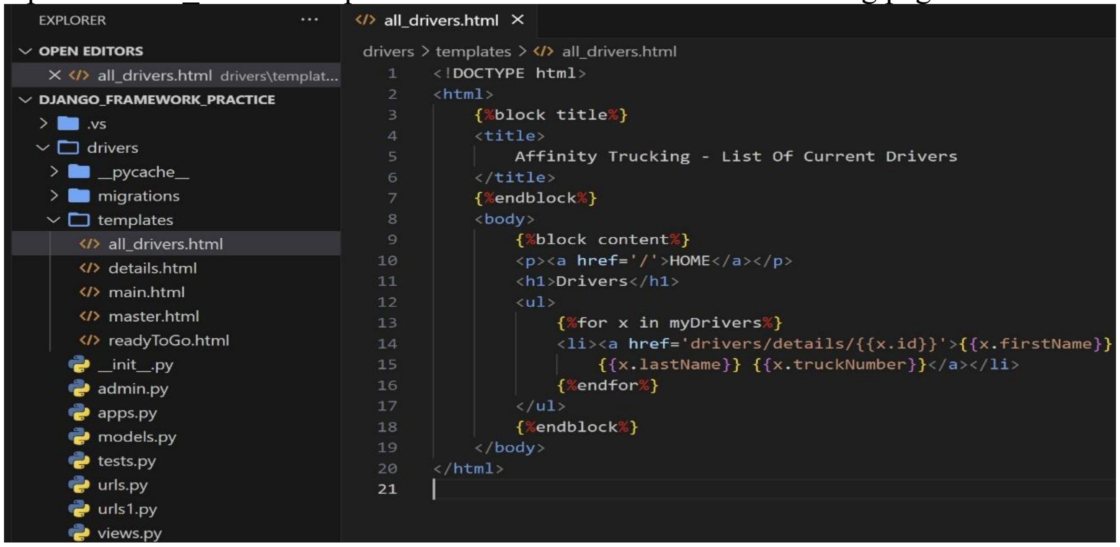
```python
#from django.shortcuts import render
from django.template import loader
from django.http import HttpResponse
from .models import Driver

# Create your views here.
def drivers(request):
    allDrivers = Driver.objects.all().values()
    template = loader.get_template('all_drivers.html')
    context = {
        'myDrivers': allDrivers
    }
    return HttpResponse(template.render(context, request))
    #1return HttpResponse("Maurice Unchained Django!")

def details(request, id):
    aDriver = Driver.objects.get(id=id)
    template = loader.get_template('details.html')
    context = {
        'myDriver': aDriver
    }
    return HttpResponse(template.render(context, request))

def main(request):
    template = loader.get_template('main.html')
    return HttpResponse(template.render())
```

   f. Add path to the main page



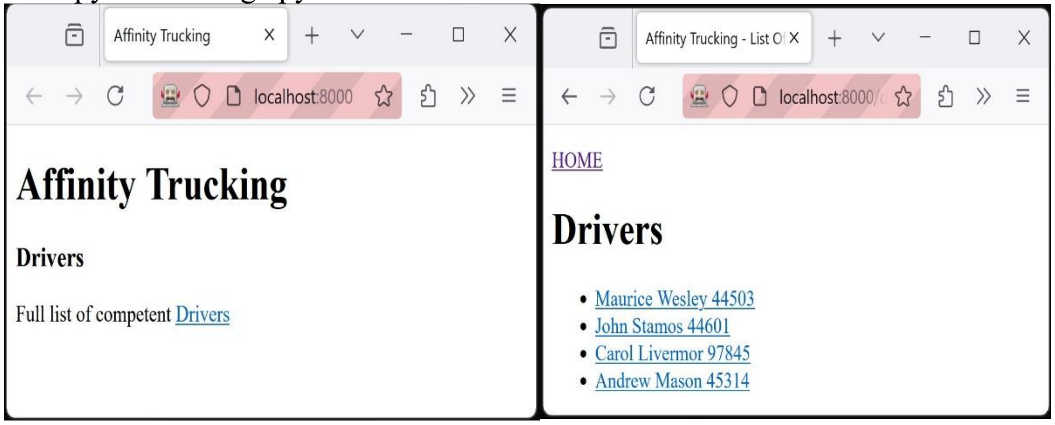
```python
from django.urls import path
from . import views

urlpatterns = [
    path('', views.main, name='main'),
    path("drivers",views.drivers, name='drivers'),
    path("drivers/details/<int:id>", views.details, name='details')
    ]
```

g.
h. Update the all_drivers template to include a link to the main landing page. See below.



```html
<!DOCTYPE html>
<html>
    {%block title%}
    <title>
        Affinity Trucking - List Of Current Drivers
    </title>
    {%endblock%}
    <body>
        {%block content%}
        <p><a href='/'>HOME</a></p>
        <h1>Drivers</h1>
        <ul>
            {%for x in myDrivers%}
            <li><a href='drivers/details/{{x.id}}'>{{x.firstName}}
                {{x.lastName}} {{x.truckNumber}}</a></li>
            {%endfor%}
        </ul>
        {%endblock%}
    </body>
</html>
```

i.
j. Run: python manage.py runserver



Affinity Trucking

Drivers

Full list of competent Drivers

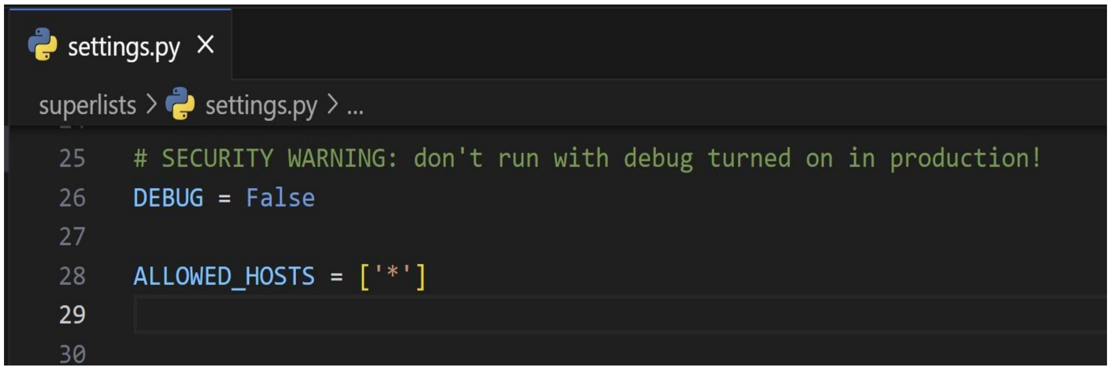
HOME

Drivers

- Maurice Wesley 44503
- John Stamos 44601
- Carol Livermor 97845
- Andrew Mason 45314

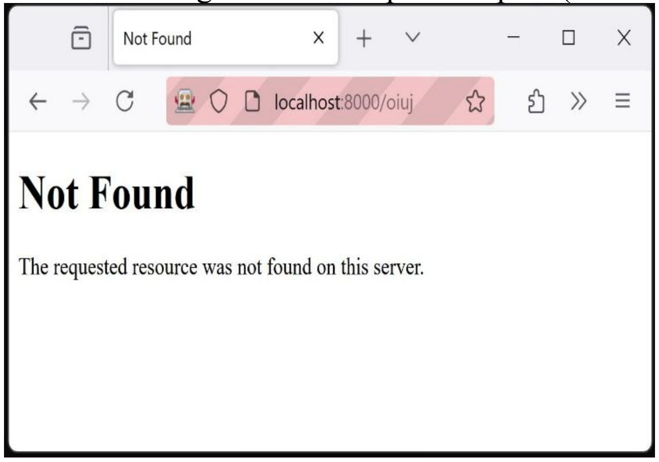
k.
    i. Success!!

12. 404 Error Page
    a. Navigating to an unspecified path will yield either a built-in Django error or an error page. To edit the error page, we have to set Debug = False in the settings.py file. See below.

b. 

```python
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = False

ALLOWED_HOSTS = ['*']

```

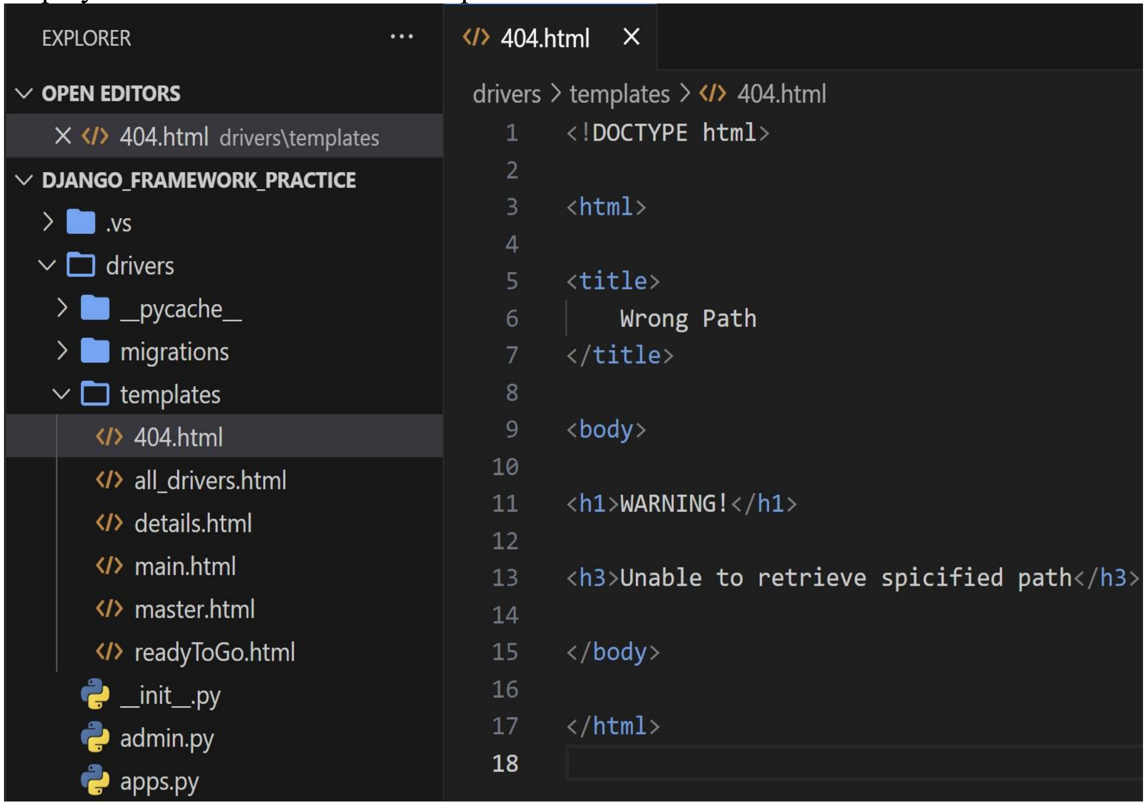
c. If the user navigates to an unspecified path (localhost:8000/oiuj), they will see the below.

d. 

Not Found

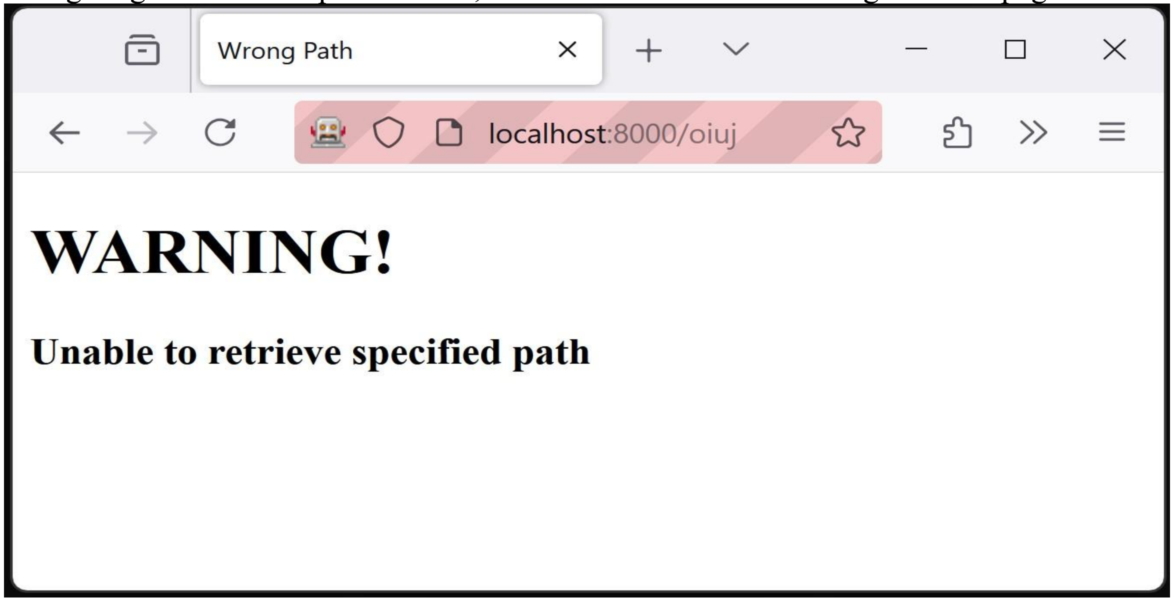
The requested resource was not found on this server.

e. Django looks for a 404-html template when Django gets a 404 error. The above is displayed when it cannot find a template. Create a 404 html.

f. 

```html
<!DOCTYPE html>

<html>

<title>
    Wrong Path
</title>

<body>

<h1>WARNING!</h1>

<h3>Unable to retrieve spicified path</h3>

</body>

</html>

```

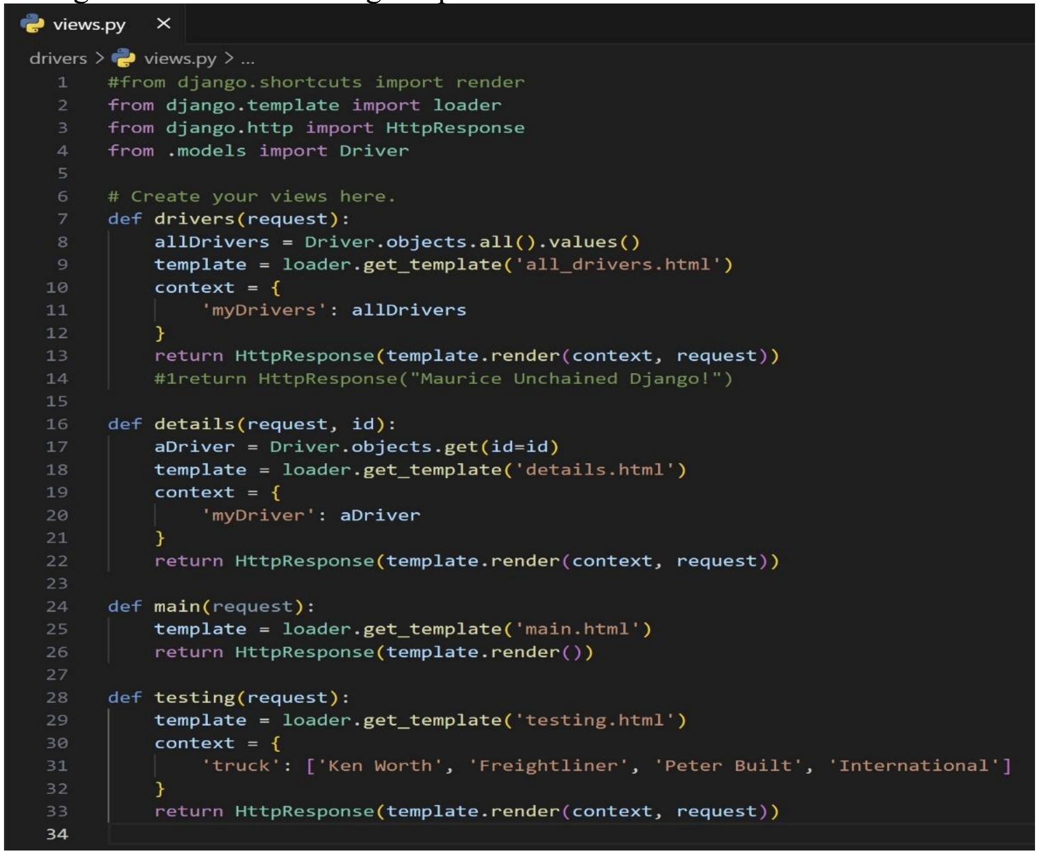
g. Navigating back to the open window, we can see the reflected changes on the page.

h. 

WARNING!

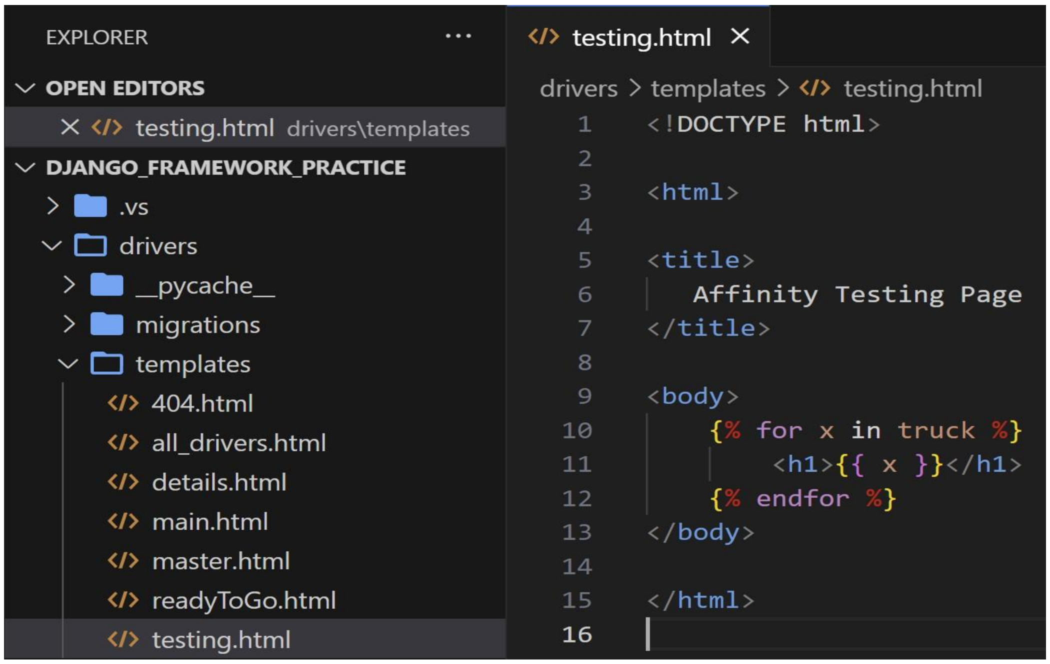
Unable to retrieve specified path

13. Testing Page
   a. Following the same process described above, create a new function in the view called testing that will load a testing template. See below

   b. 

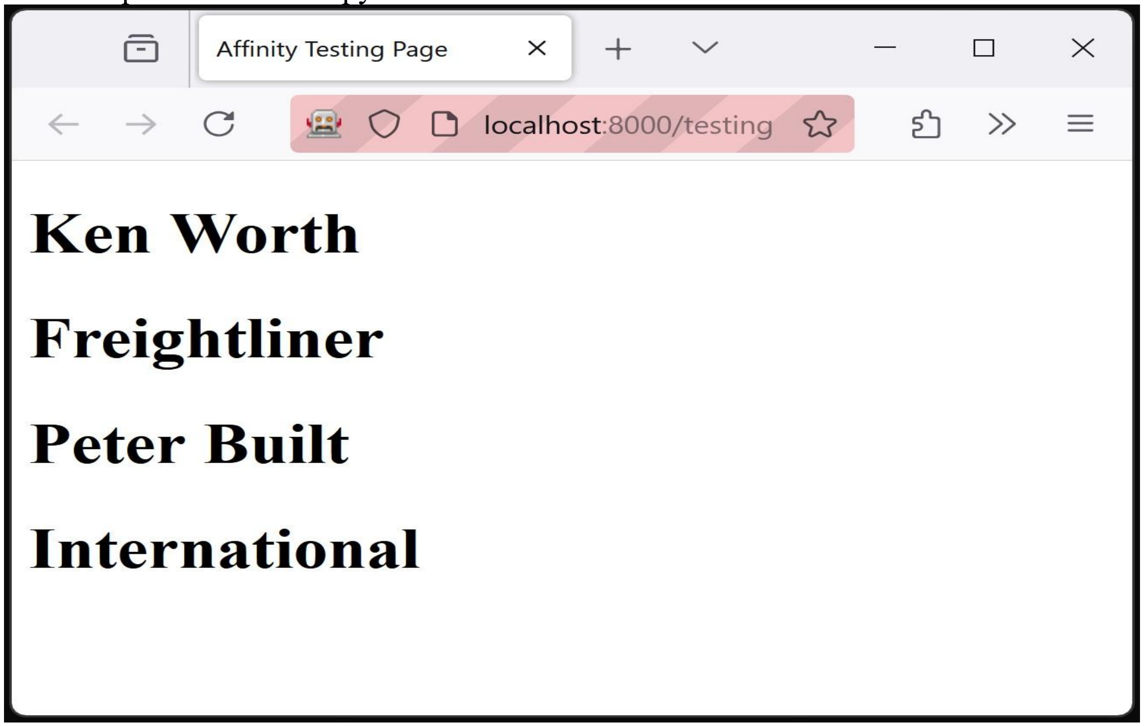
```python
#from django.shortcuts import render
from django.template import loader
from django.http import HttpResponse
from .models import Driver

# Create your views here.
def drivers(request):
    allDrivers = Driver.objects.all().values()
    template = loader.get_template('all_drivers.html')
    context = {
        'myDrivers': allDrivers
    }
    return HttpResponse(template.render(context, request))
    #1return HttpResponse("Maurice Unchained Django!")

def details(request, id):
    aDriver = Driver.objects.get(id=id)
    template = loader.get_template('details.html')
    context = {
        'myDriver': aDriver
    }
    return HttpResponse(template.render(context, request))

def main(request):
    template = loader.get_template('main.html')
    return HttpResponse(template.render())

def testing(request):
    template = loader.get_template('testing.html')
    context = {
        'truck': ['Ken Worth', 'Freightliner', 'Peter Built', 'International']
    }
    return HttpResponse(template.render(context, request))
```

   c. Now create the testing template in templates folder under the driver app.

d. 

```
<!DOCTYPE html>

<html>

<title>
    Affinity Testing Page
</title>

<body>
    {% for x in truck %}
        <h1>{{ x }}</h1>
    {% endfor %}
</body>

</html>

```

e. We added a view, created the template that the view calls, and now we add the routing to the URL pattern in the url.py file.

f. 

```
Ken Worth

Freightliner

Peter Built

International
```

i. Success!!