CM3035 Advanced Web Development

Lesson 4

Created by Ben Gay

Relational Database (cont…)

One to Many

L3proj/L3app/models.py

Import models object and create tables using foreign key

models.py urls.py

```
from django.db import models
from django.contrib.auth.models import User
```

```
class President(models.Model):
  name = models.CharField(max_length=100)

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

class Citizen(models.Model):
  president = models.ForeignKey(
          President,
          on_delete=models.CASCADE
  )
  name = models.CharField(max_length=100)

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

../L3app/url.py

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

One to Many Relationship

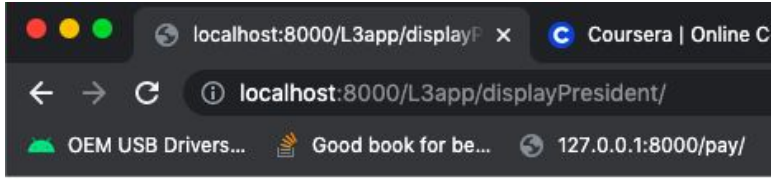
Using ForeignKey function

L3proj/L3app/views.py

```
def displayPresident(request):

    new_president = President.objects.create(id = 3 , name = "Wee Kim Wee")
    new_president.save()

    new_citizen = Citizen.objects.create(id = 8 , president_id = 3 , name = "Jovi")
    new_citizen.save()

    citizenData = Citizen.objects.get(name = "Jovi")
    presidentData = citizenData.president

    template = loader.get_template('displayPresident.html')
    context = {
      'presidentData': presidentData,
    }
    return HttpResponse(template.render(context, request))
```

templates / Browser

L3proj/L3app/templates/displayPresident.html



Display Database

Wee Kim Wee

Many to Many Relationships

Relational Database (cont…)

Many to Many relationships

L3proj/L3app/models.py

toppings = models.ManyToManyField(Topping)

```
#**********many to many relationship*************

class Topping(models.Model):

    name = models.CharField(max_length=50)

    def __str__(self):
        return self.name
class Pizza(models.Model):

    name = models.CharField(max_length=50)
    toppings = models.ManyToManyField(Topping)

    def __str__(self):
        return self.name


#*********************************************
```

../L3app/url.py

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

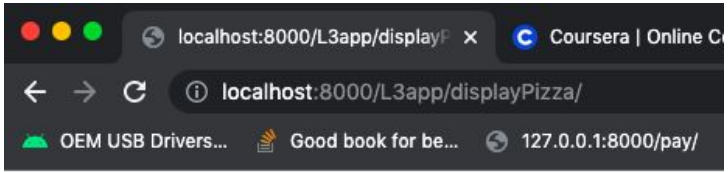
Many to Many Relationship

L3proj/L3app/views.py

```python
def displayPizza(request):

    hawaiian_pizza = Pizza.objects.create(name = "Hawaiian pizza")
    hawaiian_pizza.save()
    pineapple_toppings = Topping.objects.create(name="Pineapple toppings")
    pineapple_toppings.save()
    hawaiian_pizza.toppings.add(pineapple_toppings)
    #pizza show toppings
    pizza = hawaiian_pizza.toppings.all()
    #toppings show pizza
    #pizza = pineapple_toppings.pizza_set.all()
    template = loader.get_template('displayPizza.html')
    context = {
      'pizza': pizza,
    }
    return HttpResponse(template.render(context, request))
```

templates / Browser

L3proj/L3app/templates/displayPresident.html



Display Pizza

<QuerySet [<Topping: Pineapple toppings>]>

Access Database

models

L3proj/L3app/models.py

models.py — ~/Desktop/djangoV

models.py | urls.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
```

../L3app/url.py

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

Read all Records in table

L3proj/L3app/views.py

```
def accessDatabase(request):
  template = loader.get_template('readDatabase.html')
  query = L3app.objects.all().values()
  context = {
    'query': query,
  }
  return HttpResponse(template.render(context, request))
```

Templates / html

L3proj/L3app/templates/displayPresident.html

models.py | readDatabase.html

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

<h1>Data:</h1>
<h1>{{ query }}</h1>

</body>
</html>
```

models

L3proj/L3app/models.py

models.py — ~/Desktop/djangoV

models.py | urls.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
```

../L3app/url.py

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

Insert multiple data

L3proj/L3app/views.py

```
def writeMultipleData(request):
    template = loader.get_template('writeMultipleData.html')
    L3appRecord1 = L3app(name='Tom', hobby='hiking')
    L3appRecord2 = L3app(name='Jane', hobby='dancing')
    L3appRecord3 = L3app(name='Emil', hobby='swimming')
    L3appRecord_list = [L3appRecord1, L3appRecord2, L3appRecord3]
    for x in L3appRecord_list:
        x.save()
    querywriteMultipleData = L3app.objects.all().values()
    context = {
      'querywriteMultipleData': querywriteMultipleData,
    }
    return HttpResponse(template.render(context, request))
```

Templates / html

L3proj/L3app/templates/writeMultipleData.html

writeMultipleData.html — ~/Desktop/djangoVenv/L3proj/L3app/

writeMultipleData.html | models.py

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

<h1>Inserted into database</h1>
<h1>Hello {{ querywriteMultipleData }}, how are you?</h1>

</body>
</html>
```

Same **models** will be used throughout

../L3app/url.py

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

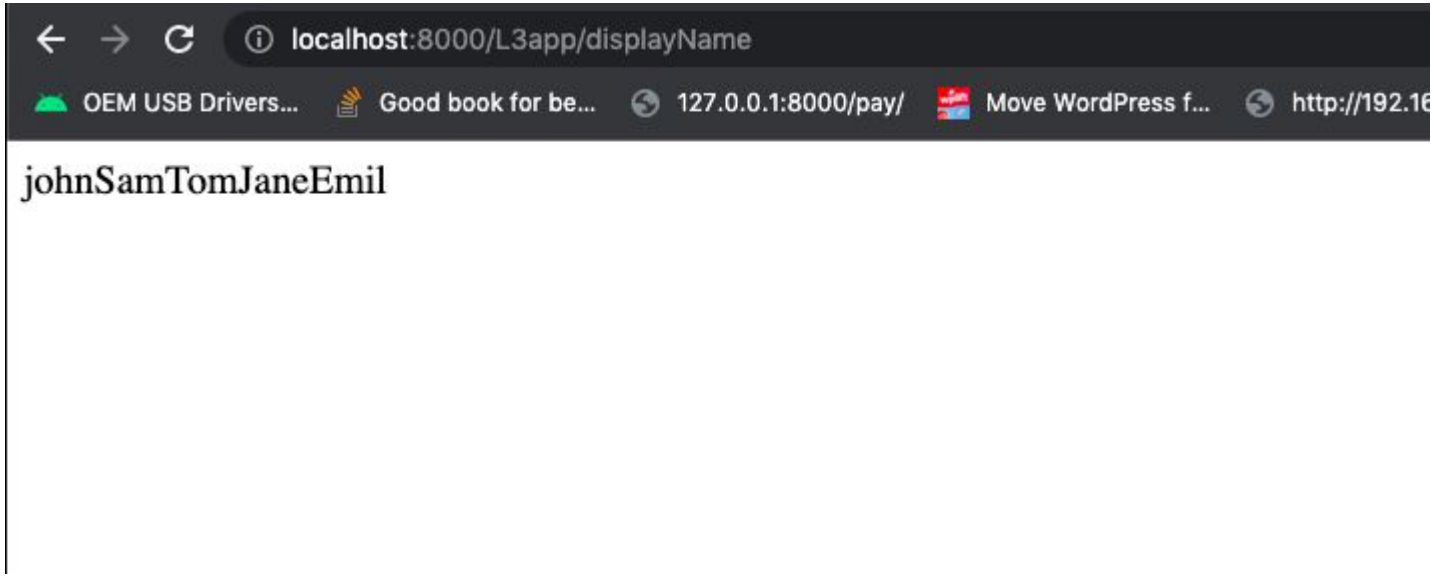
Display only name from table

L3proj/L3app/views.py

```
def displayName(request):
    namesFromDatabase = L3app.objects.all().values()
    tempOutput = ""
    for x in namesFromDatabase:
        tempOutput += x["name"]
    return HttpResponse(tempOutput)
```

Templates / html

This function responds directly to browser without templates



../L3app/url.py

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

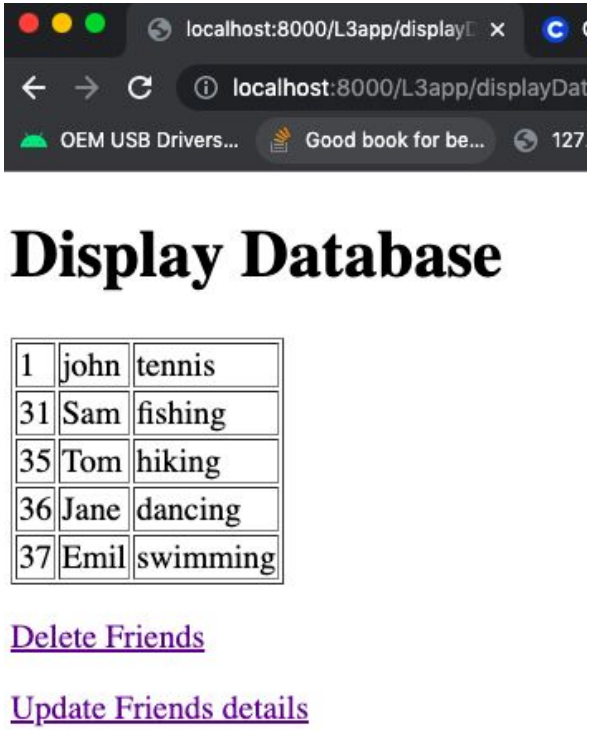
Display all records in table with Loop

L3proj/L3app/views.py

```
def displayDatabase(request):
    allData = L3app.objects.all().values()
    template = loader.get_template('displayFullData.html')
    context = {
    'allData': allData,
    }
    return HttpResponse(template.render(context, request))
```

Templates / html

For Loop is placed in html templates to display all records




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

  <h1>Display Database</h1>

  <table border="1">
  {% for x in allData %}
  <tr>
  <td>{{ x.id }}</td>
  <td>{{ x.name }}</td>
  <td>{{ x.hobby }}</td>
  </tr>
  {% endfor %}
  </table>

  <p>
  <a href="deleteData/">Delete Friends</a>
  </p>

  <p>
  <a href="updateData/">Update Friends details</a>
  </p>

</body>
</html>
```

Types of data fields in models

```
date = models.DateField()

numbers = models.IntegerField(primary_key=True)

decimals = models.FloatField()

booleans = models.BooleanField()

id = models.AutoField(auto_created = True,
primary_key = True, serialize = False, verbose_name
='ID')
```

End of Lesson 4