



django

ORM

ORM is an acronym for the **object-relational mapper**. The ORM's main goal is to transmit data between a relational database and application model. The ORM automates this transmission, such that the developer need not write any SQL.



```
from django.db import models

class Person(models.Model):
    first_name = models.CharField(max_length=30)
    last_name = models.CharField(max_length=30)
```

```
CREATE TABLE myapp_person (
    "id" serial NOT NULL PRIMARY KEY,
    "first_name" varchar(30) NOT NULL,
    "last_name" varchar(30) NOT NULL
);
```

Django project:

Required tools:

\$ python --version

\$ pip --version

Virtual Environment: \$ python -m venv myproject

\$ source myproject/bin/activate

\$ deactivate # If your virtual

environment is in a directory called 'myproject':

To specify a different port number

\$ python manage.py runserver 8090

To kill specific port number

\$ netstat -ntlp

\$ kill -9 PID

Install django:

\$ python -m pip install Django

Now Creating Project:

\$ django-admin startproject myworld //myworld is project name

Run the Django Project (Navigate to the /myworld folder)

\$ python manage.py runserver

Create App:

```
$ py manage.py startapp members //members is app name
```

```
members/views.py:
```

```
from django.http import HttpResponse
```

```
def index(request):
```

```
    return HttpResponse("Hello world!")
```

```
members/urls.py:(Create a file named urls.py in the members app)
```

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

```
from . import views
```

```
urlpatterns = [
```

```
    path("", views.index, name='index'),
```

```
]
```

Connecting App:

```
myworld/urls.py:
```

```
from django.contrib import admin
```

```
from django.urls import include, path
```

```
urlpatterns = [
```

```
    path('members/', include('members.urls')),
```

```
    path('admin/', admin.site.urls),
```

```
]
```

Template:(members/templates folder/myfirst.html.

members/views.py:

```
from django.http import HttpResponse
from django.template import loader
def index(request):
    template = loader.get_template('myfirst.html')
    return HttpResponse(template.render())
```

Change Settings:

myworld/settings.py:

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'members',
]
```

\$ py manage.py migrate

Template:myword/templates

Myworld/setting.py:

```
TEMPLATES=[...
    'DIRS': [BASE_DIR,'templates'],
```

Static File: myworld/static folder # such as css

Myworld/setting.py:

Import os

#below static url:

```
STATICFILES_DIRS= (os.path.join(BASE_DIR,
'static'),)
```

Write this code for likup static file in templates or ...

{% load static %}

<link rel="stylesheet", href="{% static 'style.css' %}"

```
$ py manage.py makemigrations
```

```
$ py manage.py migrate
```

To create django admin registration

```
python manage.py createsuperuser
```

Django Models:

members/models.py:

```
from django.db import models
class Members(models.Model):
    firstname = models.CharField(max_length=255)
    lastname = models.CharField(max_length=255)
```

Then navigate to the /myworld/ folder and run this command:

```
$ py manage.py makemigrations members
$ py manage.py migrate
```

Have to register model in admin.py

```
from .models import Room //here room is model. And it is described in model.py
admin.site.register(Room)
```

Add Records: (using python shell)

```
$ py manage.py shell
$ from members.models import Members
$ Members.objects.all()
$ member = Members(firstname='Emil', lastname='Refsnes')
$ member.save()
$ Members.objects.all().values()
```

Adding Data to a Template to the Application:

members/views.py:

```
from django.http import HttpResponse
from django.template import loader
from .models import Members
```

```
def index(request):
    mymembers = Members.objects.all().values()
    template = loader.get_template('index.html')
    context = {
        'mymembers': mymembers,
    }
    return HttpResponse(template.render(context, request))
```


Delete Data:

Adding Data to a Template to the Application:

members/views.py:

```
from django.http import HttpResponseRedirect
from django.template import loader
from .models import Members

def delete(request,id):
    member=Members.objects.get(id=id)
    member.delete()
    return HttpResponseRedirect(reverse('index'))
```

Views.py

```
path('delete/<int:id>',views.delete,name='deleterec')
```

Form.html:

```
<td><a href="delete/{{ x.id }}">delete</a></td>
```

Create variable in template --> {% with name="parvez" %}

Html tag --> {% for ,if, else.... %}

{% endfor, endif....%}

Comment --><h1>Welcome{# Everyone#}</h1>

{% comment "this was the original welcome message" %}

<h1>Welcome ladies and gentlemen</h1>

{% endcomment %}

PostGreSql in Django:

Install: postgresql
PgaAdmin

Change setting in project/setting.py:

```
DATABASES={
    'default':{
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'myproject',
        'USER': 'postgres',
        'PASSWORD': '',
        'HOST': 'localhost',
    }
}
```

Install: myproject/
pip install psycopg2
Pip install Pillow

\$ python manage.py makemigrations and migrate