First Application

Applications

- You can have any number of apps (or applications) in a website.
- Each app has its own directory in project folder.
- Each app when created has its own set of files which makes it independent from other apps in the same website, which leads to ease in maintenance and migrations.

How to create first applications?

- First you need to create project. [We have already created project in the lesson 5.1]
- Type the following command to create your first project if not created previously. Make sure you are in workspace folder [In our example, F:/projects is workspace directory]
    - django –admin startproject firstproject
- Go to firstproject folder
    - cd firstproject
- Create first app with the name testapp
    - python manage.py startapp testapp

Directory Structure

- Now you can see in IDE, a new folder with the name testapp is created along with several files. Following is the directory structure:
- Inside project folder
    - Testapp
        - __init__.py
        - models.py
        - views.py
        - admin.py
        - apps.py
        - tests.py
    - Project folder

        - __init__.py
        - settings.py
        - urls.py
        - wsgi.py
    - Manage.py

- Above is the general format and you can see the actual names of the folder in your IDE. Notice two folders with the same name as the name of your project, one inside the other. So consider it as a special folder which contains some very important files.
- There are many auto generated files in *testapp* folder. Let's understand what is necessary at this stage.

Include app in the project

- In order to django recognize your app, it must be included in settings.py of inner project folder.
- Just along with the list of preset apps, make entry for our testapp too.

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

View

- View is the most important part of your application. It takes request from the client (through django) and provide response (through django) to the user.
- Imagine if a user types some url in the browser's url bar, this request goes to the django server. Django look into the urls.py to get the associated view with this url and direct the request to that particular view, which is nothing but a function in views.py
- Views.py is simply a python file which may contain several functions. These functions are associated with some url.
- View function performs the business logic and returns response back to the client.
- You can implement views.py either using procedural approach (by creating functions) or object oriented approach (by creating classes).

- A view function, or "view" for short, is simply a Python function that takes a web request and returns a web response.
- This response can be the HTML contents of a Web page, or a redirect, or a 404 error, or an XML document, or an image, etc
- In our example, a view function with the name greeting is created.

```
from django.http import HttpResponse

def greeting(request):
    s="<h1>Hello and welcome to the first view of testapp</h1>"
    return HttpResponse(s)
```

Url

- You have to set url for your view. View is something you have created in views.py
- In urls.py, we map url pattern with a view
- How user write url?

  http://127.0.0.1:8000/hello

- In the inner project folder you have a file with the name urls.py. Write url pattern in it.

```
from testapp import views

urlpatterns = [
    path('hello',views.greeting),
]
```

Test your application

- Run server.

  python manage.py runserver

- Open browser and type url localhost:8000/hello [localhost is same as 127.0.0.1]
- A page with the response text should appear
- Ctrl+C is used to stop server

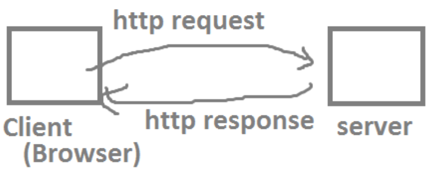
First Application

f:/projects/

```
firstproject  testapp
                  -__init__.py
                  -models.py
                  -views.py
                  -admin.py
                  -apps.py
                  -tests.py
              firstproject
                  -__init__.py
                  -settings.py
                  -urls.py
                  wsgi.py
              manage.py
```

1) Create Project
2) create Application
3) include app in settings.py
4) views.py
   function/class
5) urls.py
6) runserver

7) Send request



http://localhost:8000/hello

greeting()