

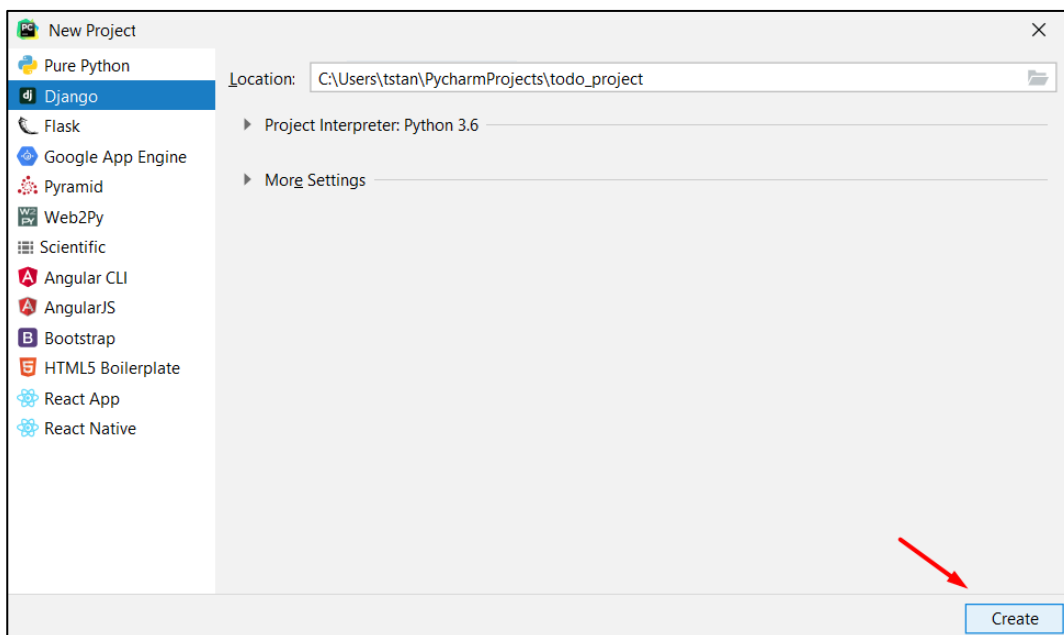
# Lab: Models and MTV Pattern

In this exercise we are going to create a **very simple To-do App**. At the end, it will look something like this

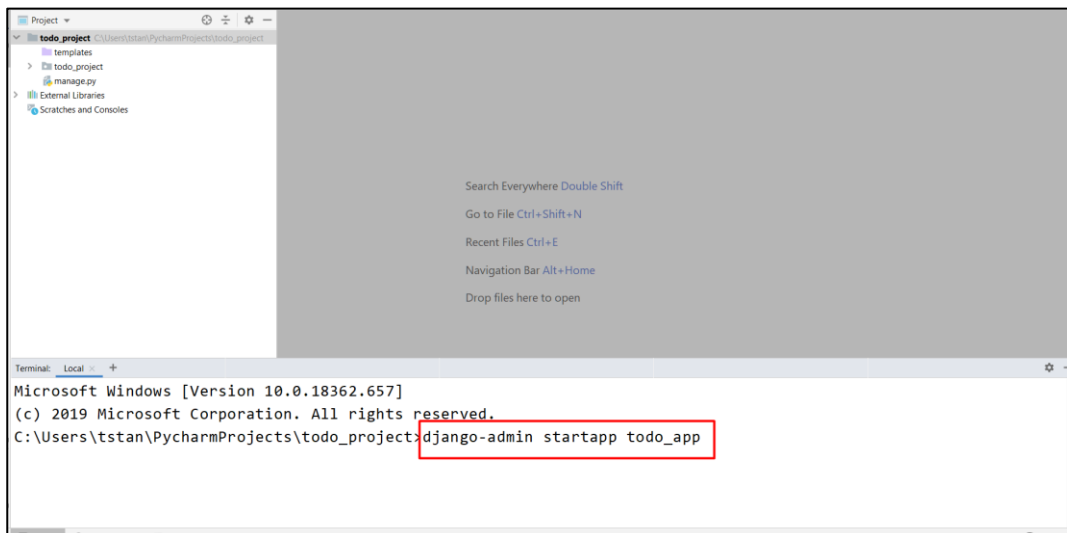


## 1. The Basics

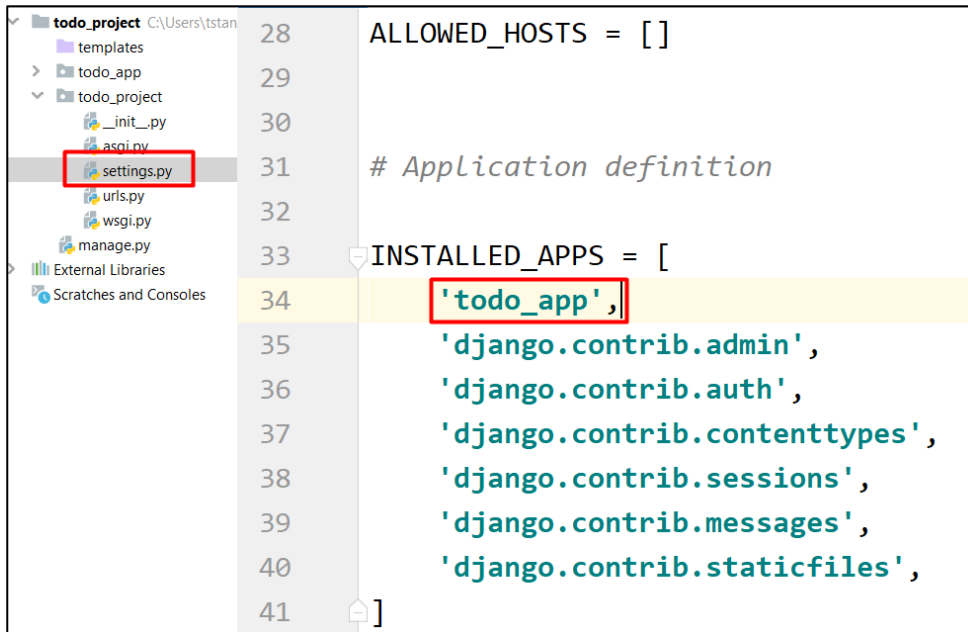
Let us start by creating a **new Django Project**



Then, we are going to **create our app**



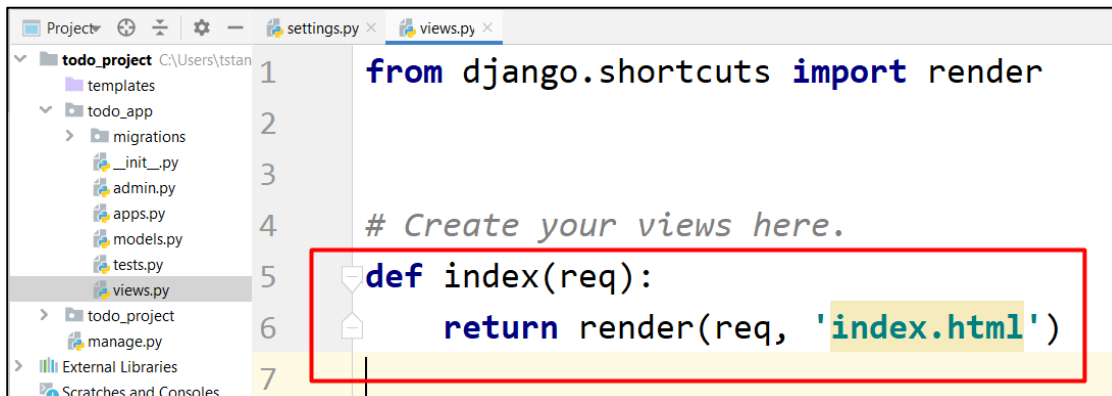
Next, we need to import our app in the **settings.py** file in the project



```
28 ALLOWED_HOSTS = []
29
30
31 # Application definition
32
33 INSTALLED_APPS = [
34     'todo_app',
35     'django.contrib.admin',
36     'django.contrib.auth',
37     'django.contrib.contenttypes',
38     'django.contrib.sessions',
39     'django.contrib.messages',
40     'django.contrib.staticfiles',
41 ]
```

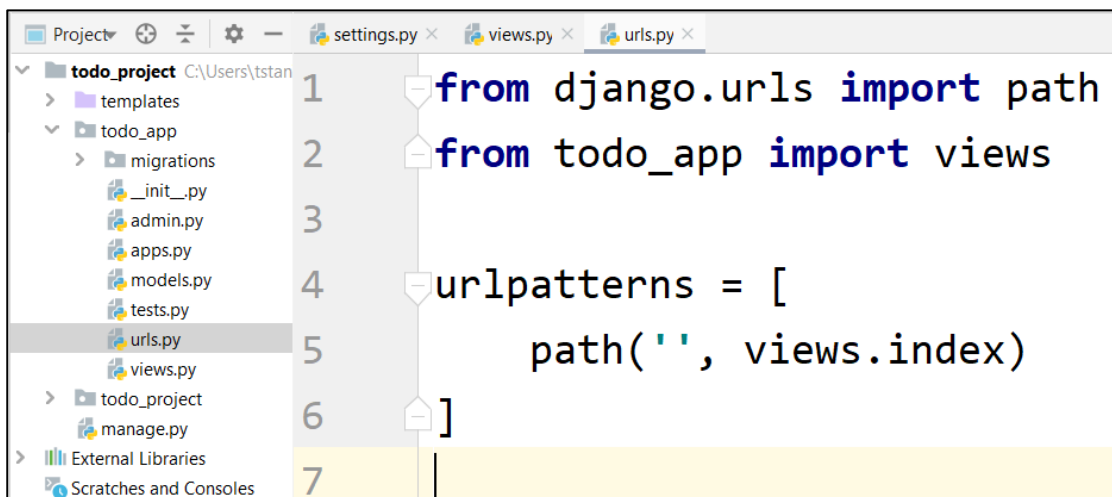
## 2. Setting up the App

Now, let us create an **index view** in our app



```
1 from django.shortcuts import render
2
3
4 # Create your views here.
5 def index(req):
6     return render(req, 'index.html')
```

Before we create the html file, let us create the **urls.py** file in the **app**



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

Now, since we do not yet have the **index.html** file, let us create it in the templates folder. You can copy the following html

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Todo App</title>
</head>
<style>
  h1 {
    margin: 5px;
    text-align: center;
    padding: 5px;
  }

  h2 {
    text-align: center;
    text-decoration: underline;
  }

  div.done {
    color: green;
    font-weight: bold;
  }

  div.open {
    color: red;
    font-weight: bold;
  }

  div.todo {
    display: block;
    background: white;
    border: 2px solid white;
    border-radius: 10px;
    margin: 5px;
    text-align: center;
  }

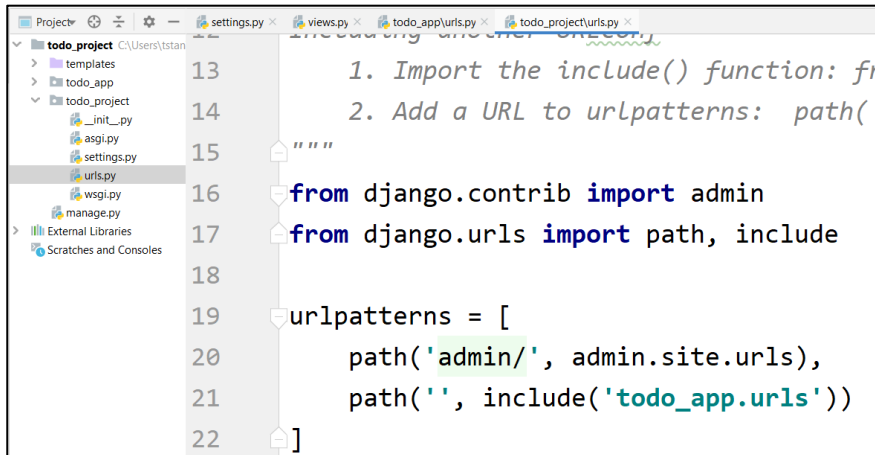
  div.todo p {
    font-size: 20px;
    font-weight: bold;
    text-decoration: underline;
  }

  div.todo div.description {
    padding: 10px;
  }

  div.container {
    background: greenyellow;
    display: flex;
    flex-direction: column;
    margin: 0 auto;
    padding: 10px;
  }
</style>
<body>
  <h1>My Todo List</h1>
</body>
</html>
```

**Note: It is a bad practice to write the style in the html file. Here we use it, because we have not yet learned about static files**

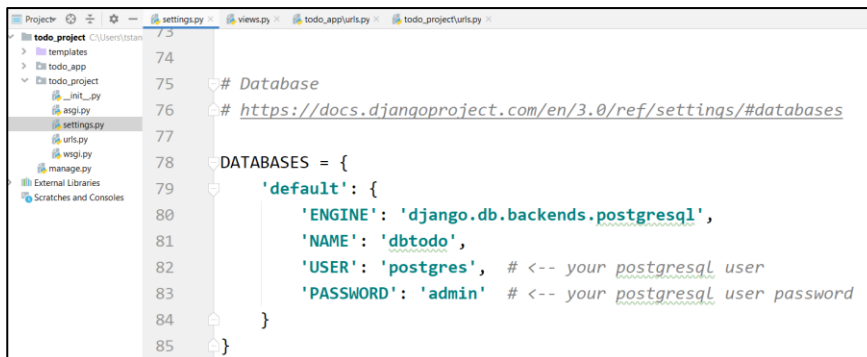
Let us include in the project the **urls** from the app



```
13 1. Import the include() function: fr
14 2. Add a URL to urlpatterns: path('
15 """
16 from django.contrib import admin
17 from django.urls import path, include
18
19 urlpatterns = [
20     path('admin/', admin.site.urls),
21     path('', include('todo_app.urls'))
22 ]
```

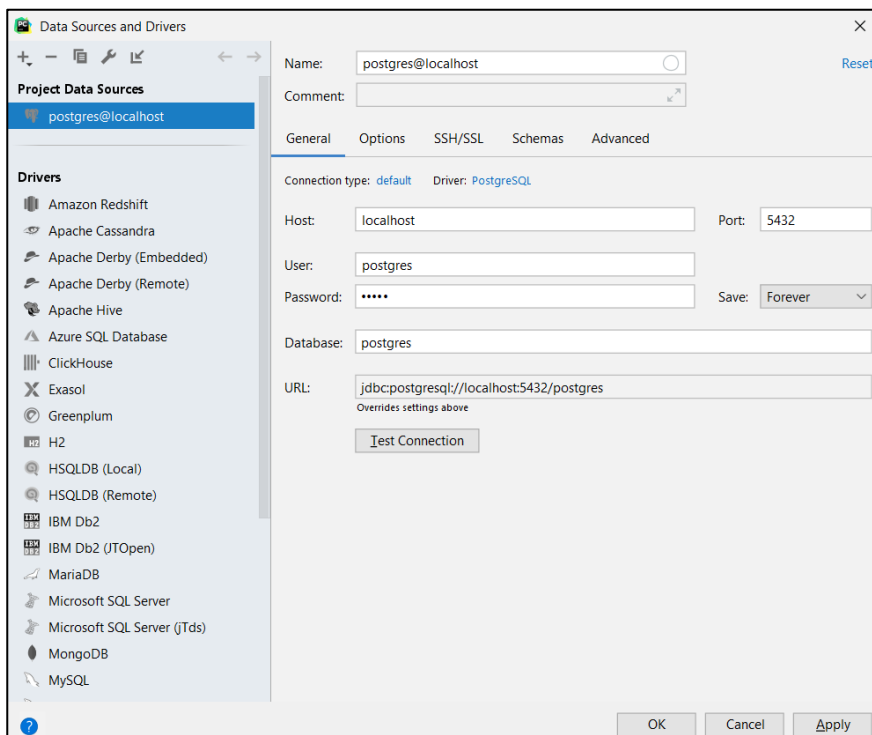
### 3. Configuring the Database

Before we test our project, let us configure the database. First, we go to the settings.py file and write the following

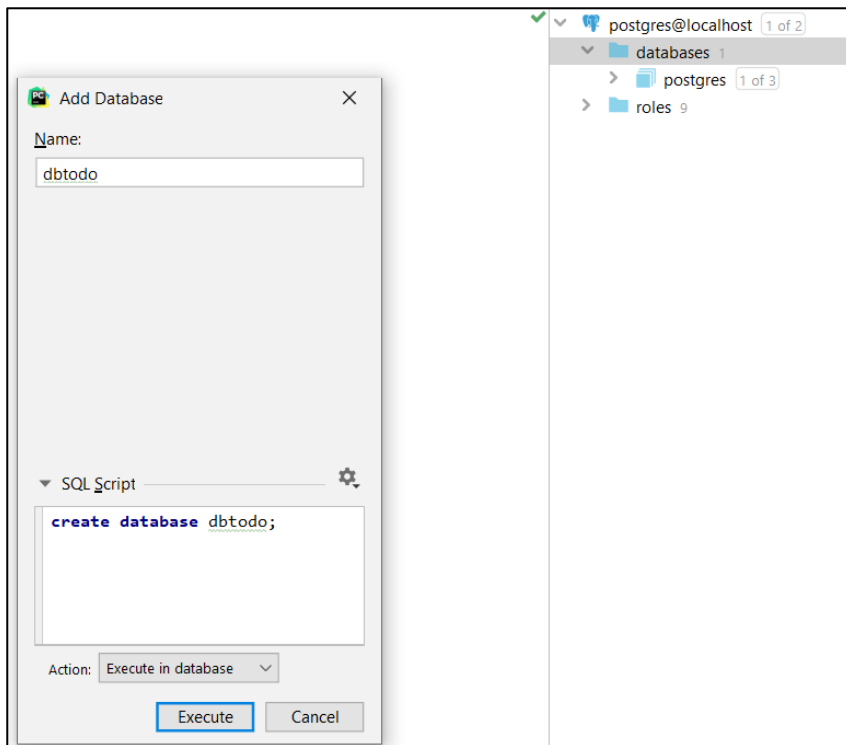


```
74
75 # Database
76 # https://docs.djangoproject.com/en/3.0/ref/settings/#databases
77
78 DATABASES = {
79     'default': {
80         'ENGINE': 'django.db.backends.postgresql',
81         'NAME': 'dbtodo',
82         'USER': 'postgres', # <-- your postgresql user
83         'PASSWORD': 'admin' # <-- your postgresql user password
84     }
85 }
```

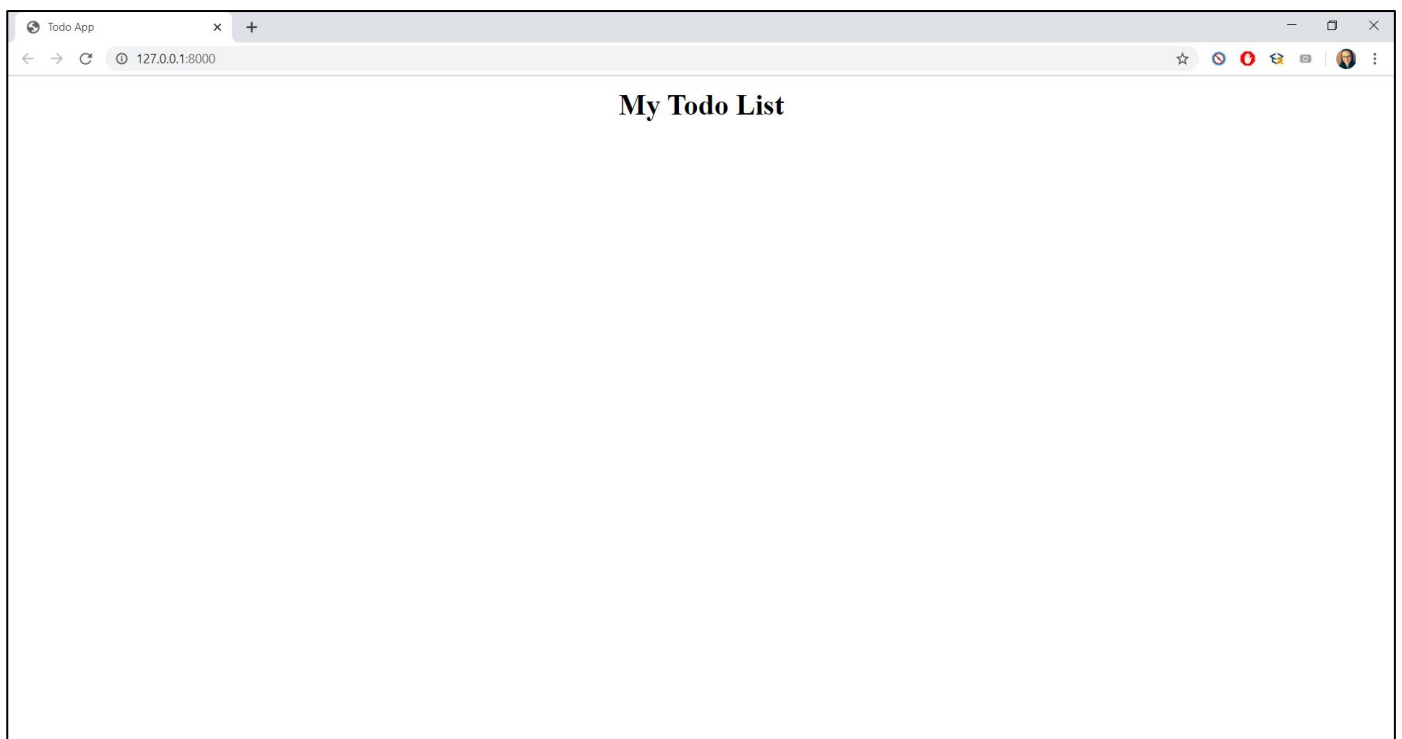
Now, let us **connect to postgresql...**



and create a new database with name "**dbtodo**"

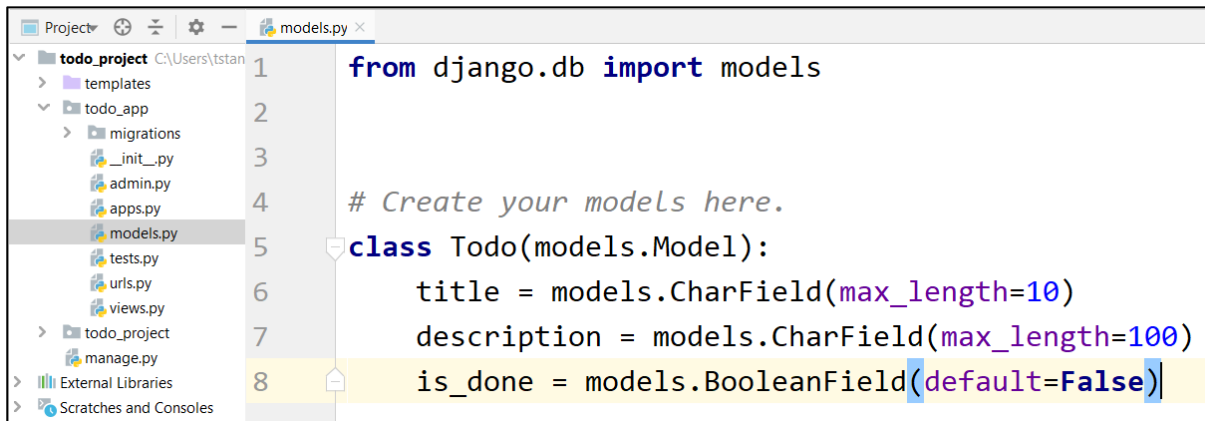


Now we are ready to test our project for the first time. Start the project and open it in the browser



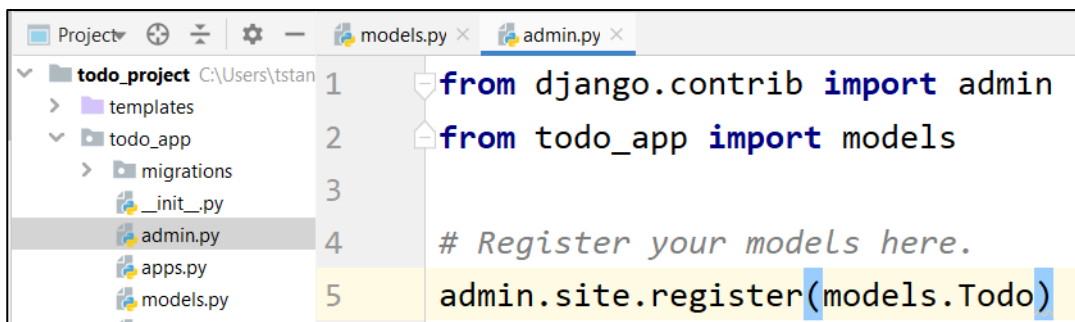
## 4. Creating the Todo Model

Now, it is time to create our model. Go to the **models.py** file and create a **class** called **Todo**



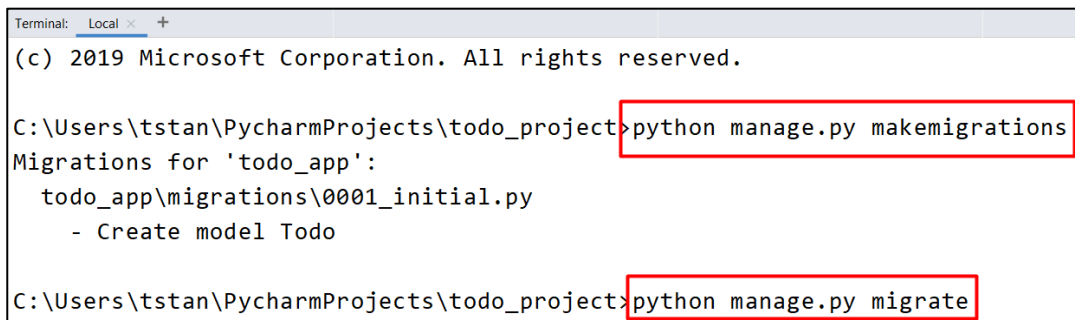
```
1 from django.db import models
2
3
4 # Create your models here.
5 class Todo(models.Model):
6     title = models.CharField(max_length=10)
7     description = models.CharField(max_length=100)
8     is_done = models.BooleanField(default=False)
```

The next step is to register the model in the **admin.py** file in the app



```
1 from django.contrib import admin
2 from todo_app import models
3
4 # Register your models here.
5 admin.site.register(models.Todo)
```

Finally, let us make the **migrations**



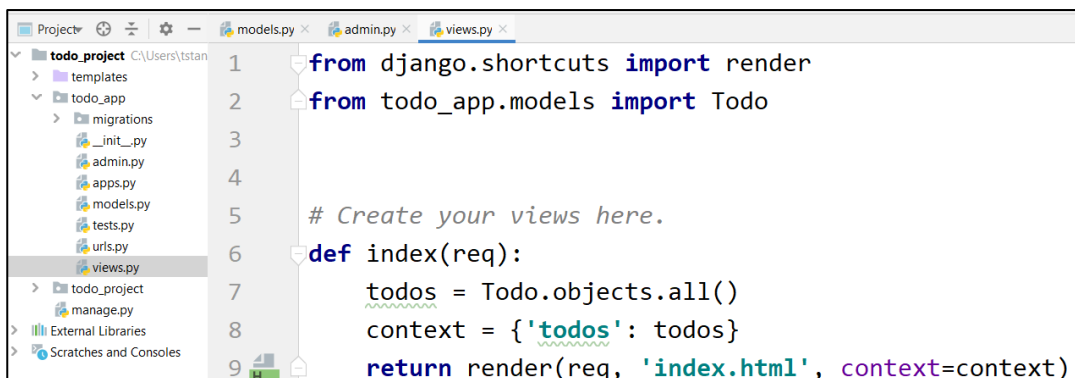
```
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\tstan\PycharmProjects\todo_project>python manage.py makemigrations
Migrations for 'todo_app':
  todo_app\migrations\0001_initial.py
    - Create model Todo

C:\Users\tstan\PycharmProjects\todo_project>python manage.py migrate
```

## 5. Adding Context to the View

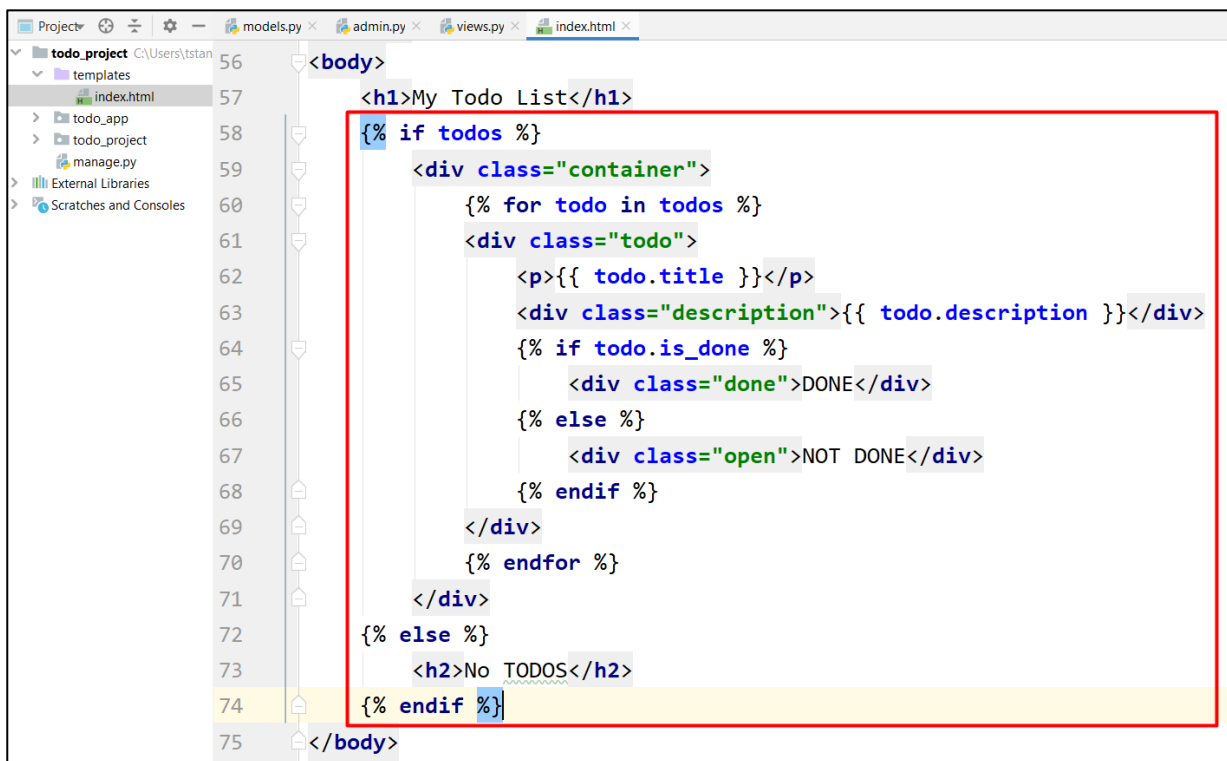
Now, let us go back to the **views.py** file and edit the index, so it passes a **context** to the **template**



```
1 from django.shortcuts import render
2 from todo_app.models import Todo
3
4 # Create your views here.
5 def index(req):
6     todos = Todo.objects.all()
7     context = {'todos': todos}
8     return render(req, 'index.html', context=context)
```

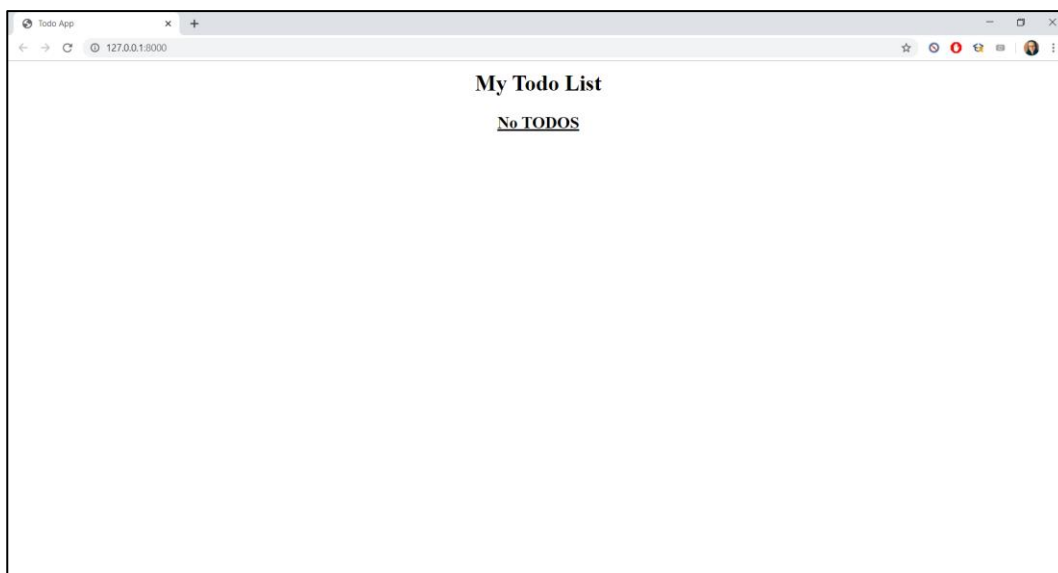
## 6. Implement Template Logic

Before testing our project again, let us **loop** through all our **todos** in the **index.html** file



```
56 <body>
57 <h1>My Todo List</h1>
58 {% if todos %}
59 <div class="container">
60     {% for todo in todos %}
61     <div class="todo">
62         <p>{{ todo.title }}</p>
63         <div class="description">{{ todo.description }}</div>
64         {% if todo.is_done %}
65             <div class="done">DONE</div>
66         {% else %}
67             <div class="open">NOT DONE</div>
68         {% endif %}
69     </div>
70     {% endfor %}
71 </div>
72 {% else %}
73 <h2>No TODOS</h2>
74 {% endif %}
75 </body>
```

Now let us **test** the project again



Since our **database is empty**, we do not render any todos, just the **h2** tag with the **"No TODOS"** text in it.

## 7. Practicing Some CRUD Operations

Since we yet do not know how to use **html forms** to create objects in the database, we have **two options**:

- Use the **Django Admin** to manually create some objects (try doing it)
- Use the **shell** to create some objects using the Django Database API (what we are going to do)

Open the terminal and type the following: `python manage.py shell`

Now you should see something like this

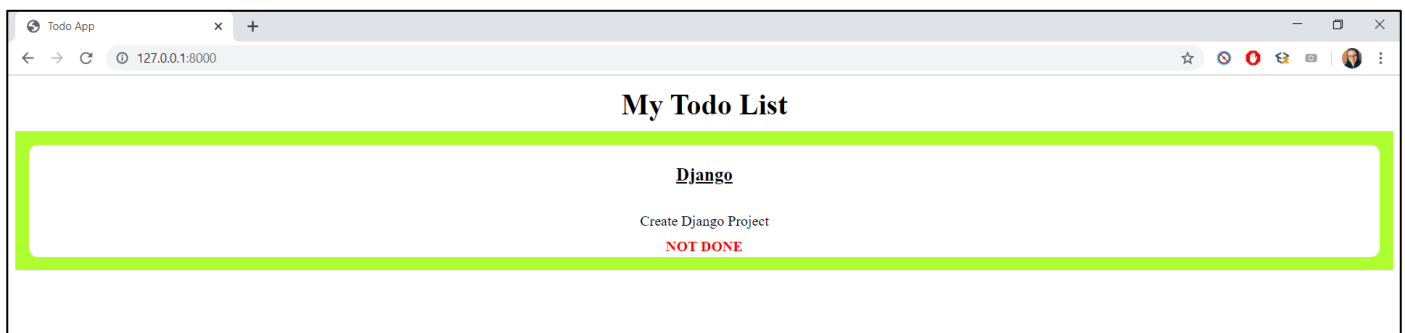
```
Terminal: Local x +
Microsoft Windows [Version 10.0.18362.657]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\tstan\PycharmProjects\todo_project>python manage.py shell
Python 3.6.2 (v3.6.2:5fd33b5, Jul 8 2017, 04:57:36) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>>
```

Now let us **create** a new Todo

```
C:\Users\tstan\PycharmProjects\todo_project>python manage.py shell
Python 3.6.2 (v3.6.2:5fd33b5, Jul 8 2017, 04:57:36) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from todo_app.models import Todo
>>> todo = Todo(title="Django", description="Create Django Project")
>>> todo.save()
>>>
```

**Refresh** the page in the browser to see the new Todo



Next, let us try and **edit** the Todo

```
Terminal: Local x +
C:\Users\tstan\PycharmProjects\todo_project>python manage.py shell
Python 3.6.2 (v3.6.2:5fd33b5, Jul 8 2017, 04:57:36) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from todo_app.models import Todo
>>> from_db = Todo.objects.get(pk=1)
>>> from_db.is_done = True
>>> from_db.save()
>>>
```

You can now see the changes in the browser.

**With that we are done with our Todo App. Try adding some more features in the app to practice what you have learned 😊**