

Fundamental Django

Oleh : Aldion Amirrul



surabaya.py

Agenda

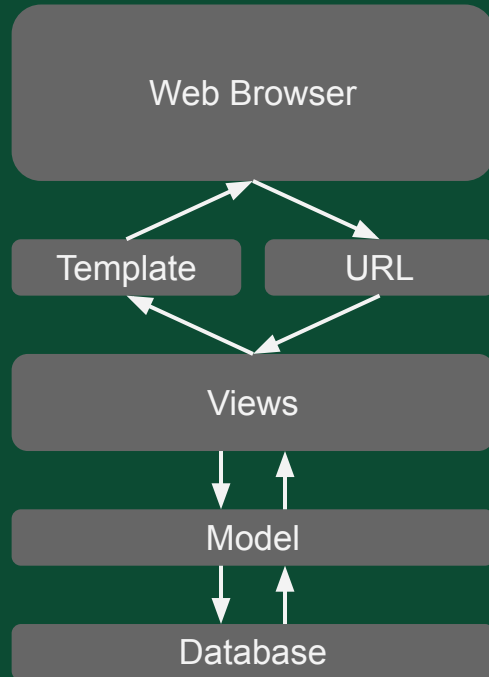
1. Apa itu Django ?
2. Pengetahuan Dasar Django
3. Contoh Program Sederhana
4. Kesimpulan

Apa itu Django ?

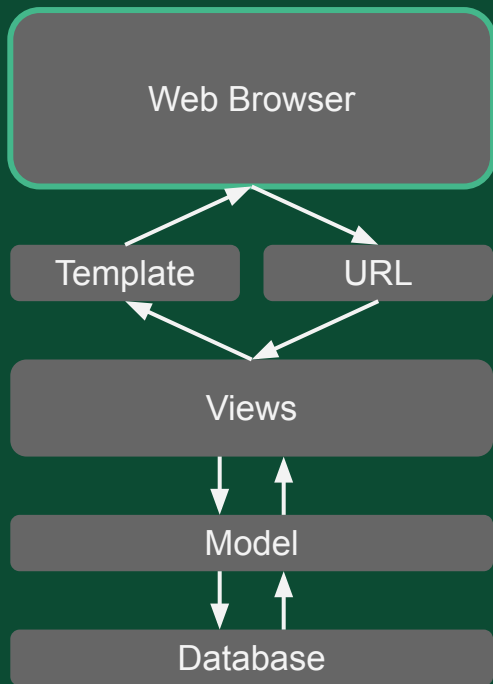
Django merupakan High-Level Web Framework menggunakan Python yang memungkinkan pengguna dapat membuat website dengan cepat yang aman, dan kode yang dapat dikembangkan kembali.

Pengetahuan Dasar Django

konsep kerja



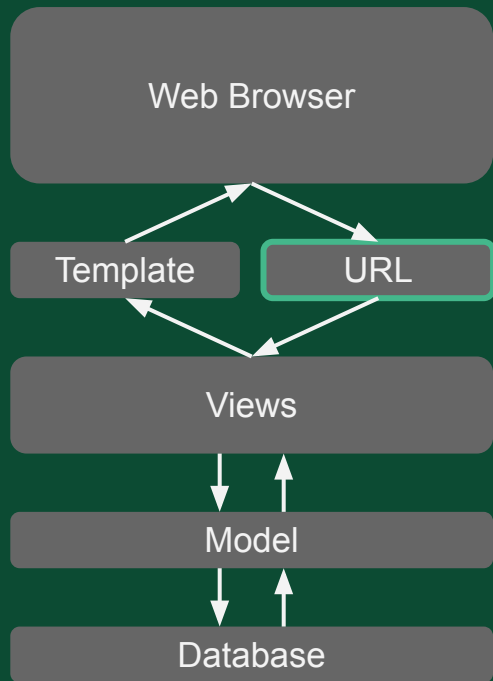
Pengetahuan Dasar Django



Web Browser

- Tampilan yang dilihat oleh pengguna
- Hasil Respon ke pengguna ketika melakukan
 1. Klik Mouse
 2. Mengetik Keyboard
 3. Menekan Enter

Pengetahuan Dasar Django

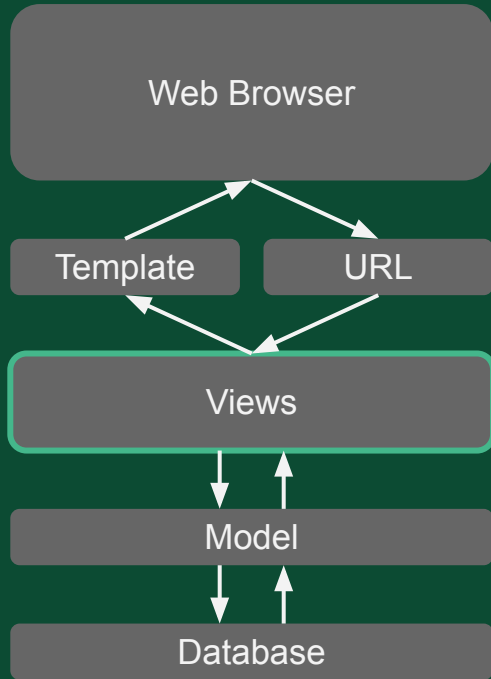


URL

- Alamat pada website
- Alamat pada website yang mengarah pada **view** atau halaman tertentu

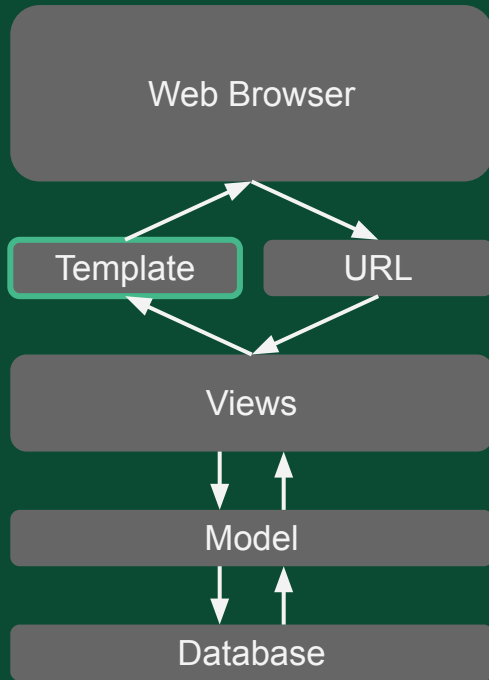
Pengetahuan Dasar Django

Views



- Merupakan sekumpulan dari beberapa fungsi atau class yang berisi logic dari program yang akan dibuat
- Mengirimkan konten / informasi pada template yang akan dirender
- Mengelola data yang didapat dari request sebelum diteruskan ke model atau template yang akan dirender

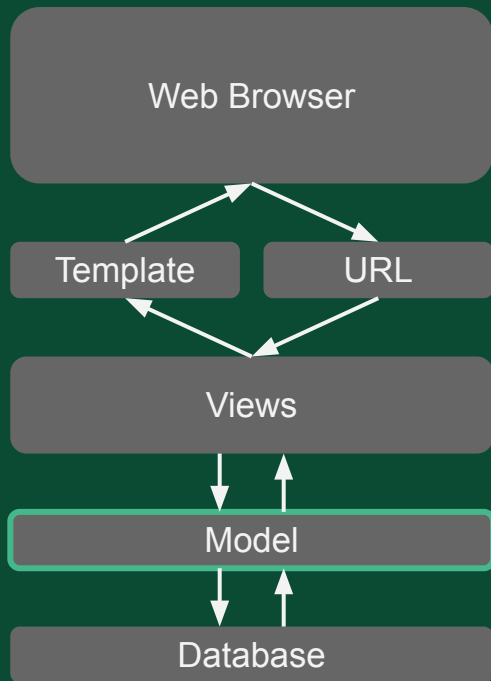
Pengetahuan Dasar Django



Template

- Tempat dimana tampilan website dibuat
- Untuk menampilkan hasil proses dari **view**

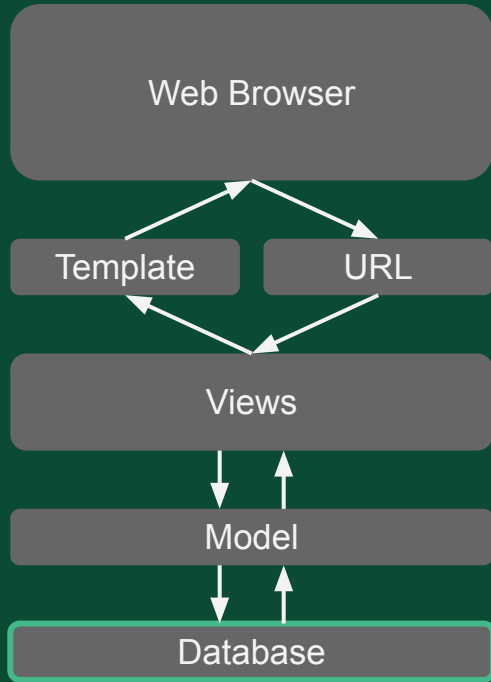
Pengetahuan Dasar Django



Model

- Tempat untuk mendefinisikan entitas(tabel) beserta atributnya(field) pada web aplikasi yang akan dibuat
- Sebagai perantara/jembatan antara database dengan view

Pengetahuan Dasar Django



Database

- Sekumpulan entitas(tabel) yang disimpan pada 1 tempat yang disebut database
- Model yang telah melalui proses migration (proses perpindahan data dari model ke database)
- Default database bawaan dari Django adalah sqlite3

Contoh Program Sederhana

Membuat program **CRUD** katalog buku sederhana pada perpustakaan.

Kebutuhan

1. Python versi 3.*
2. PIP
3. venv / Virtual environment (dianjurkan)
4. Django

Menginstall Python

Windows

<https://www.python.org/downloads/>

Linux



```
$ sudo apt install python3.8
```

Menginstall PIP

Windows

1. Unduh file [get-pip.py](#)
2. Jalankan file tersebut menggunakan Python IDLE / CMD dengan mengetikan `python get-pip.py`

Linux



```
$ sudo apt install python3-pip
```

Membuat & Menggunakan Virtual Environment

Windows

```
● ● ●  
$ pip install virtualenv  
$ virtualenv latihanenv  
$ latihanenv/Scripts/activate.bat
```

Linux

```
● ● ●  
$ pip install virtualenv  
$ virtualenv latihanenv -p python 3  
$ . latihanenv/bin/activate
```

Menginstall & Membuat project pada Django

Windows & Linux



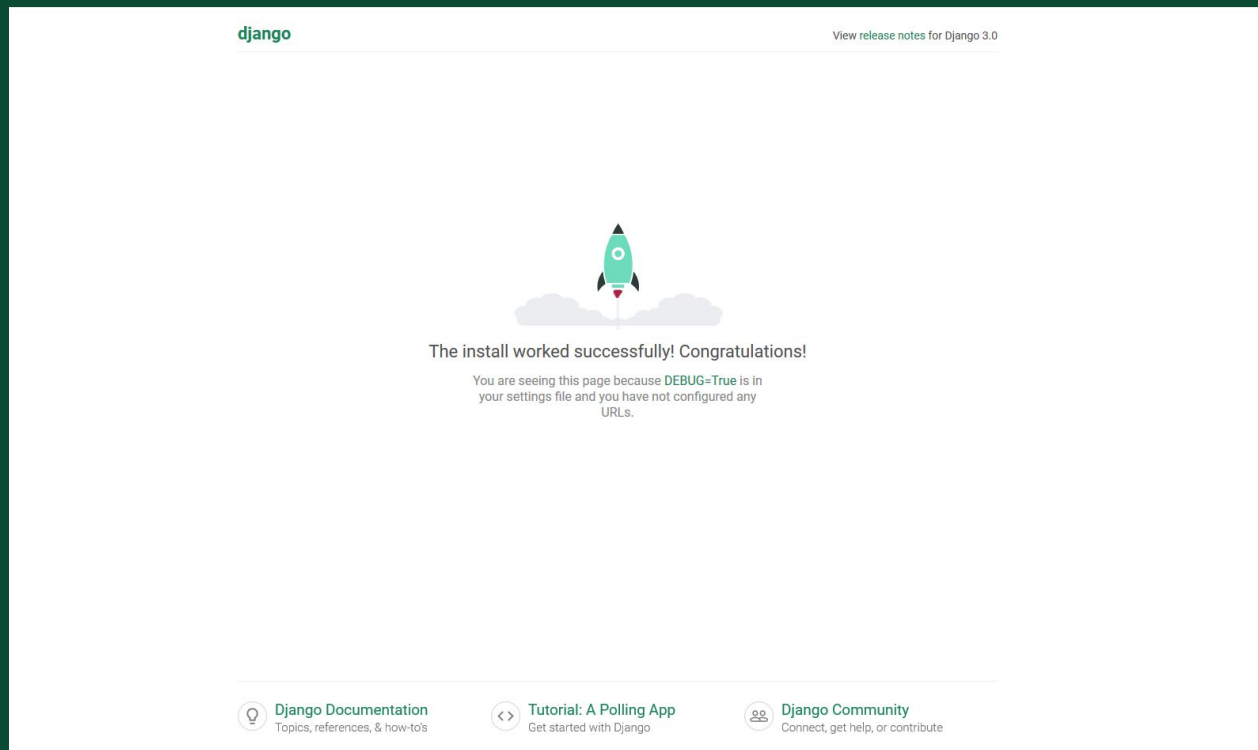
```
$ pip install django  
  
$ django-admin startproject perpustakaan  
  
$ cd perpustakaan  
  
$ python manage.py startapp katalog
```

untuk menjalankan django pada localhost :



```
$ python manage.py runserver
```


Halaman utama aplikasi setelah dijalankan



Struktur Direktori Project

```
C:.\n\---perpustakaan\n|   manage.py\n|\n|   +---katalog\n|   |   |   admin.py\n|   |   |   apps.py\n|   |   |   models.py\n|   |   |   tests.py\n|   |   |   views.py\n|   |   |   __init__.py\n|   |   |\n|   |   \---migrations\n|   |           __init__.py\n|   |\n|   \---perpustakaan\n|       asgi.py\n|       settings.py\n|       urls.py\n|       wsgi.py\n|       __init__.py
```

Membuat Aplikasi Katalog Sederhana

1. Tambahkan app barusan kita buat kedalam **INSTALLED_APPS** pada file **settings.py** di folder **perpustakaan** pada project direktori anda.

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

Membuat Aplikasi Katalog Sederhana

2. Buat Model baru dengan nama **Buku** pada file `models.py` di folder `katalog`.

```
from django.db import models

# Create your models here.

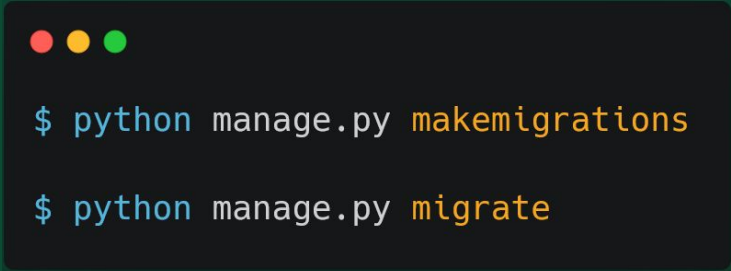
class Buku(models.Model):
    id = models.AutoField(primary_key=True, null=False)
    judul = models.CharField(max_length=100, null=False)
    pengarang = models.CharField(max_length=100, null=False)
    penerbit = models.CharField(max_length=100, null=False)
    tahun = models.IntegerField(null=False)

    def __str__(self):
        return self.judul

class Meta:
    db_table = 'buku'
```

Membuat Aplikasi Katalog Sederhana

3. Jalankan perintah **makemigrations** pada terminal melalui file **manage.py** di folder root project anda, untuk membuat file migration dari model yang anda buat sebelum dirubah ke bentuk tabel pada database anda & perintah **migrate** untuk mengubah file migrations ke bentuk tabel pada database.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top left corner. It contains two lines of text, each preceded by a dollar sign (\$).

```
$ python manage.py makemigrations  
  
$ python manage.py migrate
```

Membuat Aplikasi Katalog Sederhana

4. Import fungsi `redirect` & `get_object_or_404` dan import Model `Buku` dari file `models.py` kedalam `views.py` di folder `katalog`.



```
from django.shortcuts import render, redirect, get_object_or_404
from .models import Buku
```

Membuat Aplikasi Katalog Sederhana

5. Buat fungsi baru dengan nama **index**, **create**, **update**, **delete** pada file **views.py** di folder **katalog** .

fungsi **index**



```
def index(request):  
    data_buku = Buku.objects.all()  
    return render(request, 'katalog/index.html', {'data_buku': data_buku})
```

Membuat Aplikasi Katalog Sederhana

fungsi **create**

```
def create(request):  
    if request.method == 'POST':  
        try:  
            Buku(  
                judul = request.POST['judul'],  
                pengarang = request.POST['pengarang'],  
                penerbit = request.POST['penerbit'],  
                tahun = request.POST['tahun']  
            ).save()  
            return redirect('katalog:index')  
        except:  
            return redirect('katalog:create')  
    else:  
        return render(request, 'katalog/create.html')
```



Membuat Aplikasi Katalog Sederhana

fungsi **update**

```
def update(request, pk):  
    buku = get_object_or_404(Buku, pk=pk)  
    if request.method == 'POST':  
        try:  
            buku.judul = request.POST['judul']  
            buku.penerbit = request.POST['penerbit']  
            buku.pengarang = request.POST['pengarang']  
            buku.tahun = request.POST['tahun']  
            buku.save()  
            return redirect('katalog:update', pk=pk)  
        except:  
            return redirect('katalog:update', pk=pk)  
    else:  
        return render(request, 'katalog/update.html', {'buku': buku})
```

Membuat Aplikasi Katalog Sederhana

fungsi **delete**



```
def delete(request, pk):  
    buku = get_object_or_404(Buku, pk=pk)  
    buku.delete()  
    return redirect('katalog:index')
```

file views.py

```
from django.shortcuts import render, redirect, get_object_or_404
from .models import Buku

# Create your views here.
def index(request):
    data_buku = Buku.objects.all()
    return render(request, 'katalog/index.html', {'data_buku': data_buku})

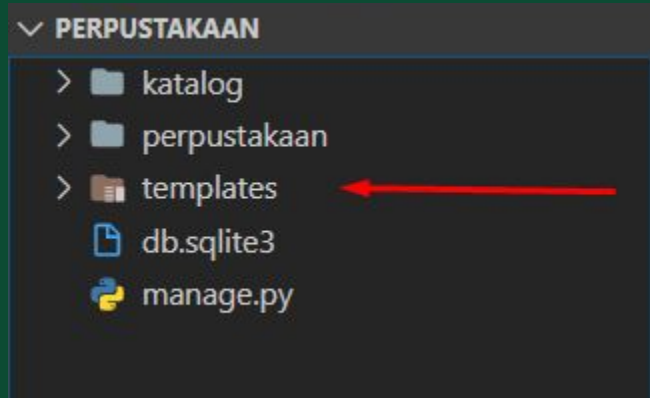
def create(request):
    if request.method == 'POST':
        try:
            Buku(
                judul = request.POST['judul'],
                pengarang = request.POST['pengarang'],
                penerbit = request.POST['penerbit'],
                tahun = request.POST['tahun']
            ).save()
            return redirect('katalog:index')
        except:
            return redirect('katalog:create')
    else:
        return render(request, 'katalog/create.html')

def update(request, pk):
    buku = get_object_or_404(Buku, pk=pk)
    if request.method == 'POST':
        try:
            buku.judul = request.POST['judul']
            buku.penerbit = request.POST['penerbit']
            buku.pengarang = request.POST['pengarang']
            buku.tahun = request.POST['tahun']
            buku.save()
            return redirect('katalog:update', pk=pk)
        except:
            return redirect('katalog:update', pk=pk)
    else:
        return render(request, 'katalog/update.html', {'buku': buku})

def delete(request, pk):
    buku = get_object_or_404(Buku, pk=pk)
    buku.delete()
    return redirect('katalog:index')
```

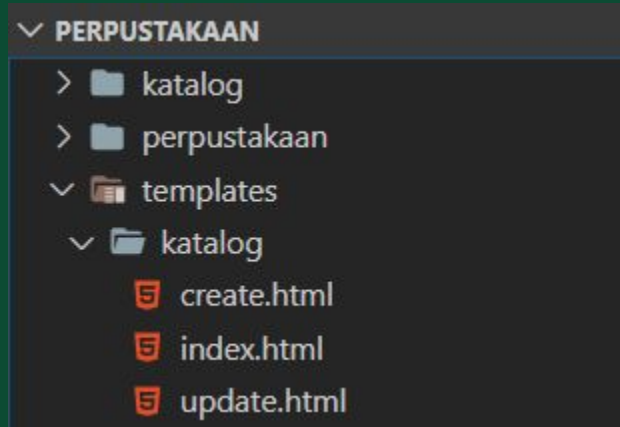
Membuat Aplikasi Katalog Sederhana

6. Buat folder dengan nama **templates** pada root direktori project anda.



Membuat Aplikasi Katalog Sederhana

7. Buat folder baru dengan nama **katalog** didalam folder **templates** yang barusan anda buat, serta buat 4 file html pada folder katalog dengan nama file sebagai berikut.



8. Kemudian isikan 3 file HTML pada folder **katalog** dengan kodingan berikut.

index.html

```
<html lang="en">
<head>
  <title>Katalog Buku</title>
</head>
<body>
  <a href="{% url 'katalog:create' %}">Tambah Buku</a>
  <hr>
  <table border="1">
    <thead>
      <td>Judul Buku</td>
      <td>Penerbit</td>
      <td>Pengarang</td>
      <td>Tahun</td>
      <td>Ops</td>
    </thead>
    <tbody>
      {% for buku in data_buku %}
        <tr>
          <td>{{ buku.judul }}</td>
          <td>{{ buku.penerbit }}</td>
          <td>{{ buku.pengarang }}</td>
          <td>{{ buku.tahun }}</td>
          <td>
            <a href="{% url 'katalog:delete' buku.id %}">Hapus</a>
            <a href="{% url 'katalog:update' buku.id %}">Ubah</a>
          </td>
        </tr>
      {% endfor %}
    </tbody>
  </table>
</body>
</html>
```

create.html

```
<html lang="en">
<head>
  <title>Tambah Buku</title>
</head>
<body>
  <form action="" method="post">
    {% csrf_token %}
    <label for="judul">Judul</label>
    <input type="text" name="judul">
    <br>

    <label for="penerbit">penerbit</label>
    <input type="text" name="penerbit">
    <br>

    <label for="pengarang">pengarang</label>
    <input type="text" name="pengarang">
    <br>

    <label for="tahun">Tahun</label>
    <input type="number" min="1900" name="tahun">
    <br>
    <button type="submit">Buat</button>
  </form>
</body>
</html>
```

update.html

```
<html lang="en">
<head>
  <title>Update Buku {{ buku.judul }}</title>
</head>
<body>
  <form action="" method="post">
    {% csrf_token %}
    <label for="judul">Judul</label>
    <input type="text" name="judul" value="{{ buku.judul }}">
    <br>

    <label for="penerbit">penerbit</label>
    <input type="text" name="penerbit" value="{{ buku.penerbit }}">
    <br>

    <label for="pengarang">pengarang</label>
    <input type="text" name="pengarang" value="{{ buku.pengarang }}">
    <br>

    <label for="tahun">Tahun</label>
    <input type="number" min="1900" name="tahun" value="{{ buku.tahun }}">
    <br>
    <button type="submit">Ubah Data</button>
  </form>
</body>
</html>
```


Membuat Aplikasi Katalog Sederhana

9. Setelah membuat template html, langkah berikutnya adalah mendaftarkan direktori folder **templates** pada file **settings.py** di direktori **perpustakaan**.

```
TEMPLATE_DIR = os.path.join(BASE_DIR,"templates")
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [TEMPLATE_DIR],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    ],
]
```

Membuat Aplikasi Katalog Sederhana

10. Membuat file `urls.py` pada direktori `katalog`, lalu daftarkan views yang sudah dibuat pada file `views.py` agar dapat diakses melalui URL pada Web Browser.

```
from django.urls import path
from .views import index, create, update, delete

app_name = 'katalog'

urlpatterns = [
    path('', index, name='index'),
    path('create', create, name='create'),
    path('update/<pk>', update, name='update'),
    path('delete/<pk>', delete, name='delete'),
]
```

Membuat Aplikasi Katalog Sederhana


11. Kemudian ubah file `urls.py` pada direktori `perpustakaan` dengan kodingan seperti berikut.

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

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('katalog.urls', namespace='katalog'))
]
```

Membuat Aplikasi Katalog Sederhana

12. Jalankan perintah **runserver** melalui file **manage.py** pada direktori root project anda untuk menjalankan aplikasinya.



```
$ python manage.py runserver
```

Kesimpulan

Django sangat cocok digunakan untuk mengembangkan aplikasi website secara cepat, maintainable dan terstruktur.

Dengan dukungan banyak plugin atau library yang tersedia di internet, mengerjakan website dengan menggunakan framework django lebih cepat dan hemat waktu.

Referensi

- <https://simpleisbetterthancomplex.com/series/2017/09/04/a-complete-beginners-guide-to-django-part-1.html>
- <https://djangobook.com/mdj2-django-structure/>
- <https://www.petanikode.com/django-untuk-pemula/>
- <https://medium.com/@ksarthak4ever/django-class-based-views-vs-function-based-view-e74b47b2e41b>