

MANAJEMEN PERPUS.ira

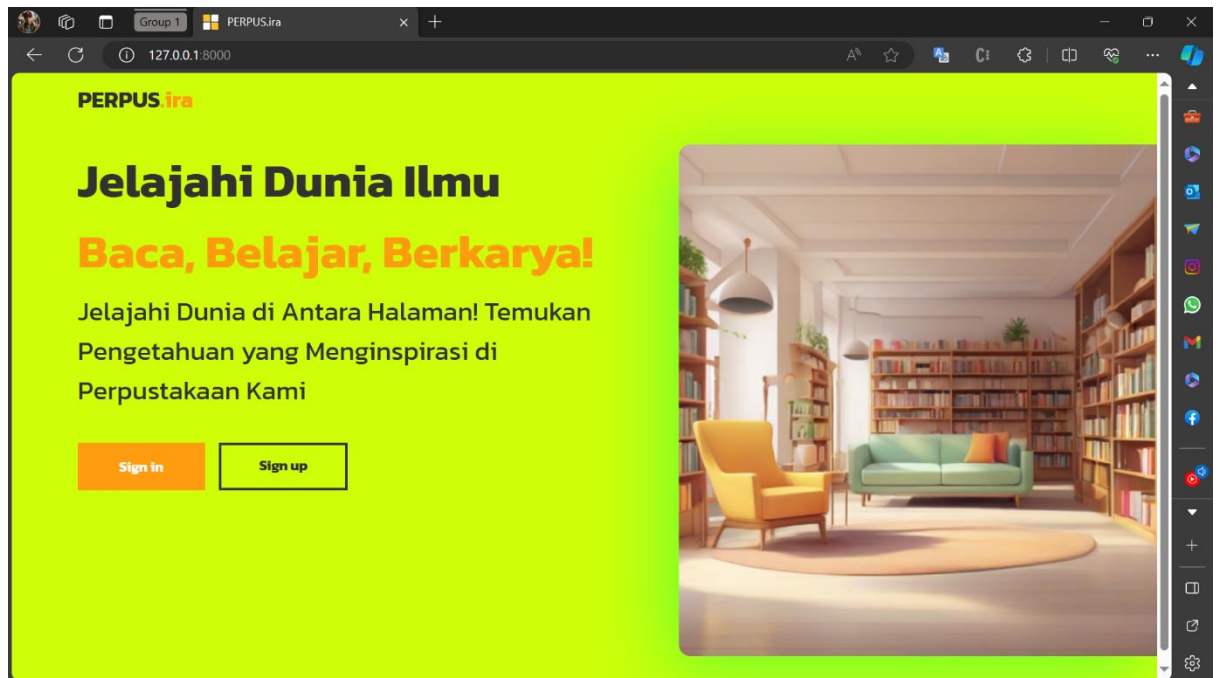
Perpus.ira adalah sebuah sistem perpustakaan yang berkomitmen untuk menyediakan layanan informasi dan membantu memfasilitasi akses kepada sumber daya pengetahuan bagi komunitas penggunaannya. Dengan menggunakan prinsip-prinsip manajemen modern, Perpus.ira bertujuan untuk memberikan layanan yang efisien, efektif, dan berorientasi pada kebutuhan pembaca.

❖ Fitur

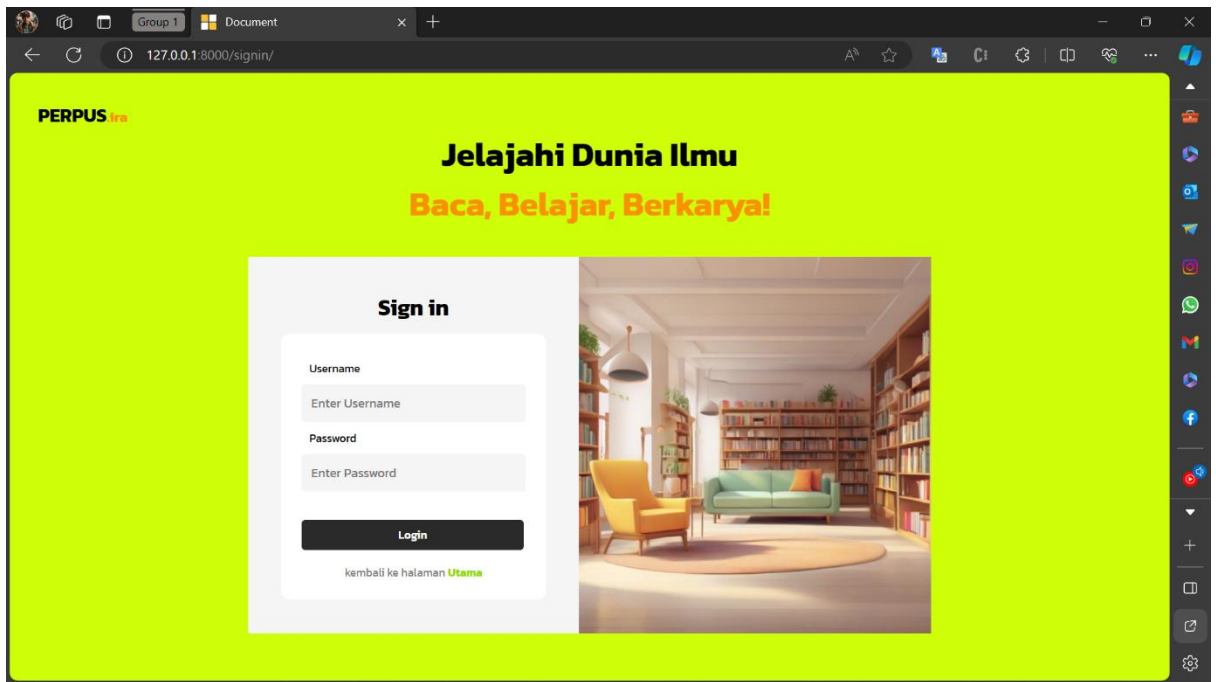
- Login dan register
- Dashboard dan statistic
- Management anggota
- Management buku dan kategori
- Management peminjaman dan pengembalian

❖ Tampilan user interface

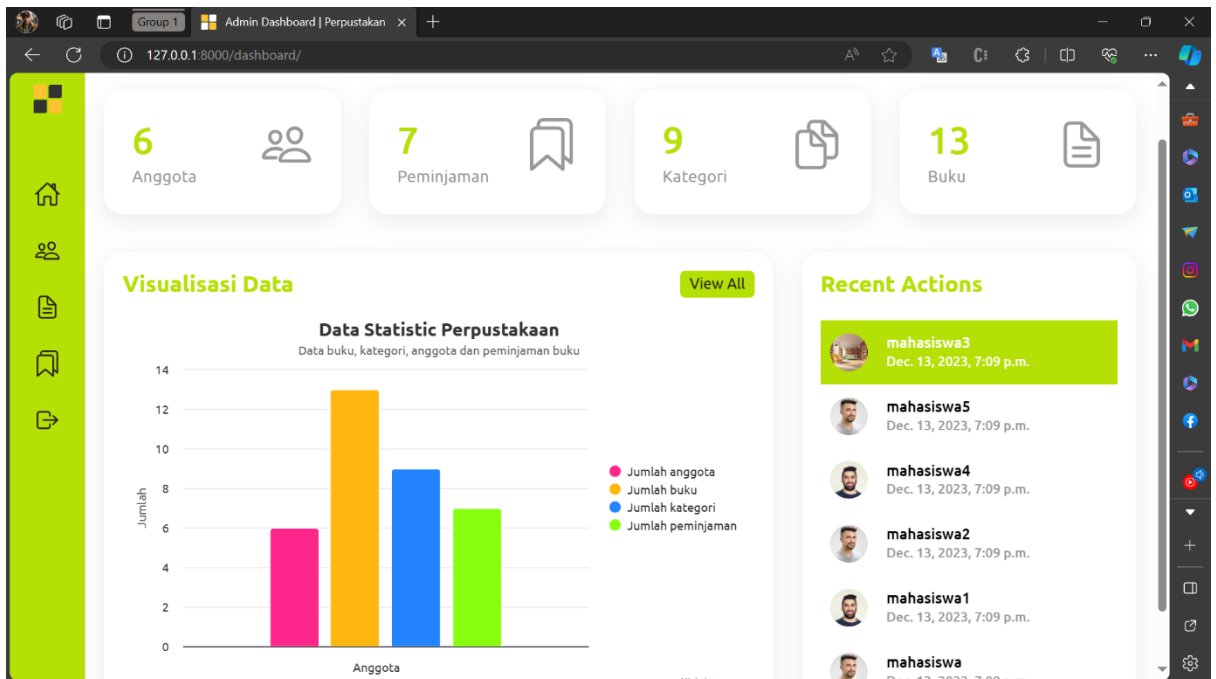
- Home



- Sign in



- Dashboard



- Anggota

Data Anggota Perpustakaan Add Anggota

NO	FOTO	NAMA	JENIS KELAMIN	ALAMAT	EMAIL	NOMOR HP	AKSI
1		mahasiswa	Perempuan	Anjani	anggota@gmail.com	08111111111	
2		mahasiswa1	Laki-Laki	Anjani	anggota@gmail.com	08111111111	
3		mahasiswa2	Laki-Laki	Anjani	anggota@gmail.com	08111111111	
4		mahasiswa4	Laki-Laki	Anjani	anggota@gmail.com	08111111111	
5		mahasiswa5	Laki-Laki	Anjani	anggota@gmail.com	08111111111	

Previous **1** 2 Next

Histori Anggota

- mahasiswa9**
Dec. 15, 2023, 1:18 a.m.
- mahasiswa8**
Dec. 15, 2023, 1:18 a.m.
- mahasiswa7**
Dec. 15, 2023, 1:17 a.m.
- mahasiswa3**
Dec. 13, 2023, 7:09 p.m.
- mahasiswa5**
Dec. 13, 2023, 7:09 p.m.

- Buku dan kategori

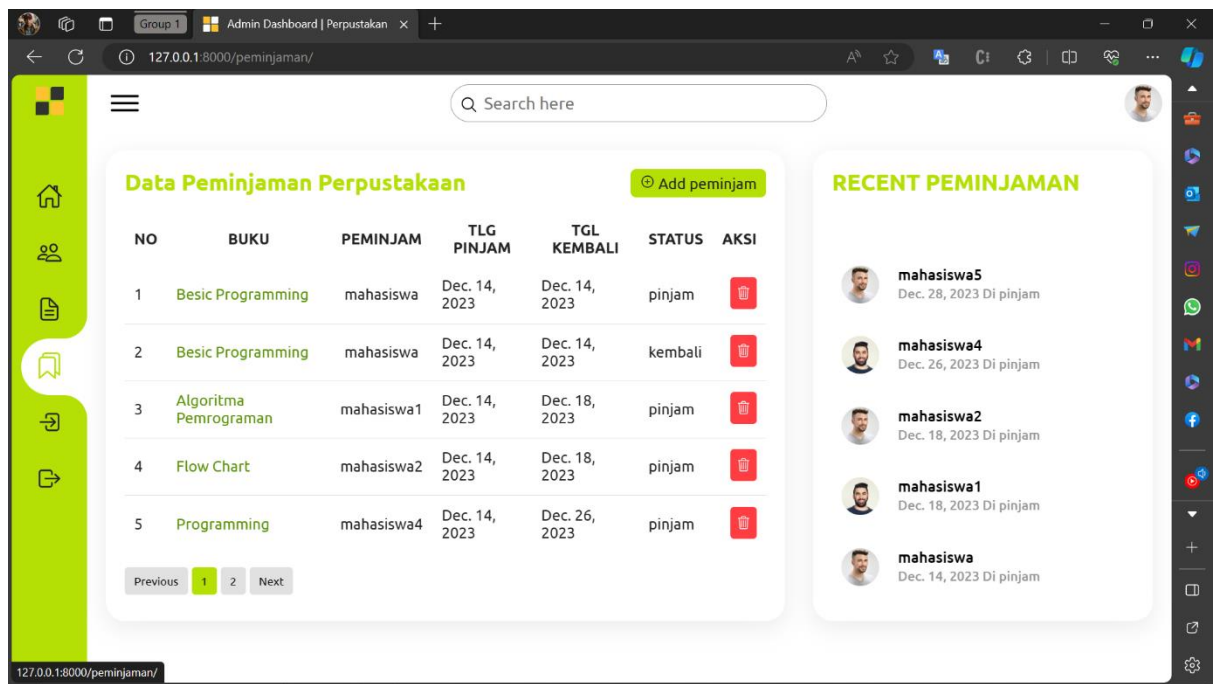
Data Buku dan Kategori Perpustakaan Add Buku

NO	JUDUL	PENULIS	TH TERBIT	KATEGORI	AKSI
1	Basic Programming	elqusairi	2023	Python	
2	Algoritma Pemrograman	elqusairi	2023	Python	
3	Flow Chart	elqusairi	2023	Java	
4	Programming	elqusairi	2023	C++	
5	Data Analist	elqusairi	2023	Python	
6	Mechines Learning	elqusairi	2023	Python	
7	WEB	elqusairi	2023	Django	
8	Android	elqusairi	2023	React Js	
9	Ios Apps	elqusairi	2023	Dart	
10	IoT	elqusairi	2023	C++	

Kategori Buku

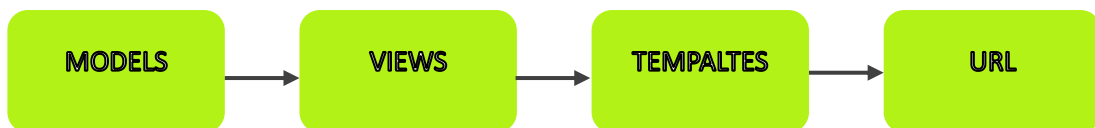
- Django**
Dec. 13, 2023, 7:13 p.m.
Dec. 13, 2023, 7:13 p.m.
- Java**
Dec. 13, 2023, 7:13 p.m.
Dec. 13, 2023, 7:13 p.m.
- Python**
Dec. 13, 2023, 7:13 p.m.
Dec. 13, 2023, 7:13 p.m.
- Javascript**
Dec. 13, 2023, 7:13 p.m.
Dec. 13, 2023, 7:18 p.m.
- Dart**
Dec. 13, 2023, 7:13 p.m.
Dec. 13, 2023, 7:13 p.m.
- React Js**
Dec. 13, 2023, 7:13 p.m.
Dec. 13, 2023, 7:13 p.m.
- C #**
Dec. 13, 2023, 7:13 p.m.
Dec. 13, 2023, 7:13 p.m.
- C++**
Dec. 13, 2023, 7:13 p.m.
Dec. 13, 2023, 7:13 p.m.

- Peminjaman dan pengembalian



❖ Penjelasan program

- Alur program

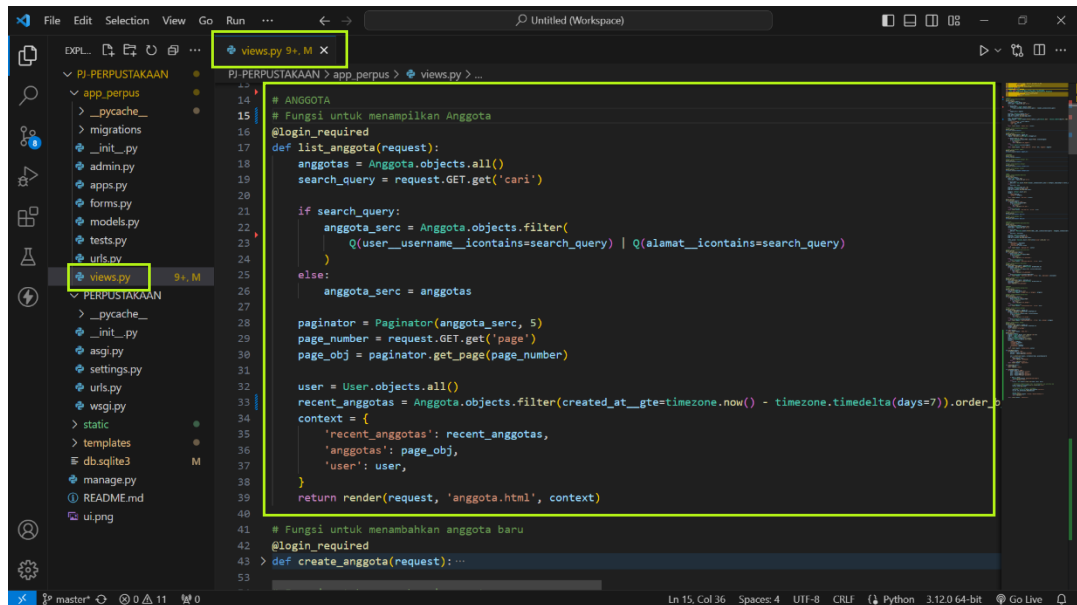


- **MODELS** : Adalah representasi struktur **database** atau entitas data. Ini mendefinisikan cara data disimpan, diambil, dan dikelola di dalam aplikasi. Model biasanya berisi kelas-kelas Python yang **mewakili tabel-tabel** dalam basis data aplikasi.

```

1 from django.db import models
2 from django.contrib.auth.models import User
3 from django.utils import timezone
4
5 class Kategori(models.Model):
6     nama_kategori = models.CharField(max_length=100)
7
8     created_at = models.DateTimeField(auto_now_add=True)
9     updated_at = models.DateTimeField(auto_now=True)
10
11     def __str__(self):
12         return self.nama_kategori
13
14
15 class Buku(models.Model):
16     judul = models.CharField(max_length=200)
17     penulis = models.CharField(max_length=100)
18     tahun_terbit = models.IntegerField()
19     kategori = models.ForeignKey(Kategori, on_delete=models.CASCADE, related_name='buku_set')
20
21     created_at = models.DateTimeField(auto_now_add=True)
22     updated_at = models.DateTimeField(auto_now=True)
23
24     def __str__(self):
25         return self.judul
26
27
28 class Anggota(models.Model):
29     foto = models.ImageField(upload_to='img', default=False)
30     user = models.OneToOneField(User, on_delete=models.CASCADE, related_name='anggota')
31     KELAMIN_CHOICES = [
32         ('Laki-Laki', 'Laki-Laki'),
  
```

- **VIEWS** : Views mengatur cara data ditampilkan kepada pengguna atau diolah sebelum ditampilkan. Mereka menerima permintaan HTTP dari browser atau aplikasi klien, melakukan operasi yang diperlukan, dan merender (menampilkan) data yang relevan ke dalam format yang sesuai.

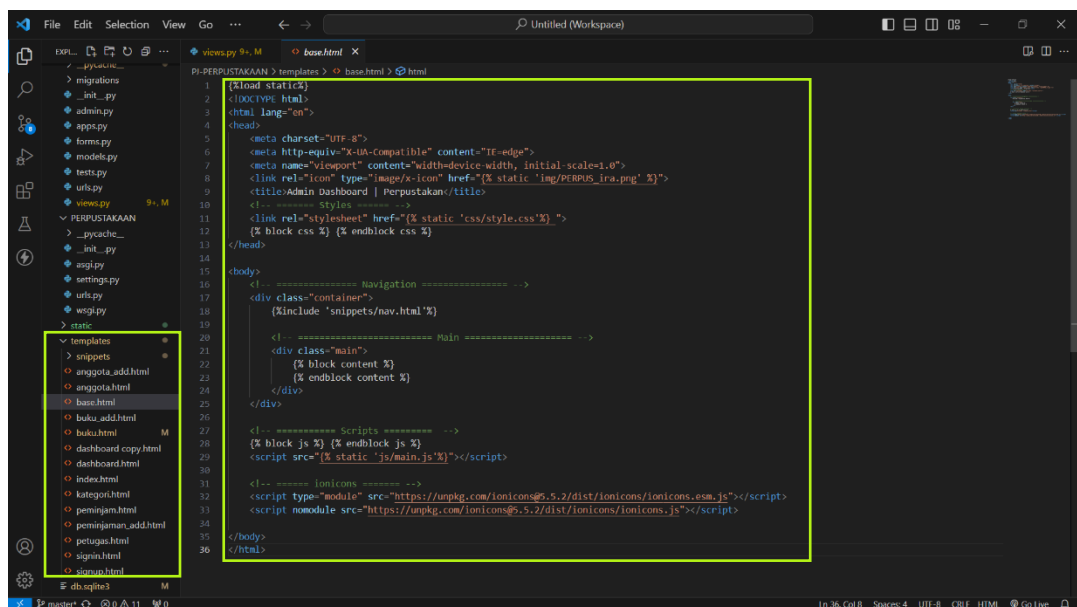


```

14 # ANGKOTA
15 # Fungsi untuk menampilkan Anggota
16 @login_required
17 def list_anggota(request):
18     anggota = Anggota.objects.all()
19     search_query = request.GET.get('cari')
20
21     if search_query:
22         anggota_serc = Anggota.objects.filter(
23             Q(user_username__icontains=search_query) | Q(alamat__icontains=search_query)
24         )
25     else:
26         anggota_serc = anggota
27
28     paginator = Paginator(anggota_serc, 5)
29     page_number = request.GET.get('page')
30     page_obj = paginator.get_page(page_number)
31
32     user = User.objects.all()
33     recent_anggotas = Anggota.objects.filter(created_at__gte=timezone.now() - timezone.timedelta(days=7)).order_by(
34         '-created_at')
35     context = {
36         'recent_anggotas': recent_anggotas,
37         'anggotas': page_obj,
38         'user': user,
39     }
40     return render(request, 'anggota.html', context)
41
42 # Fungsi untuk menambahkan anggota baru
43 @login_required
44 def create_anggota(request):

```

- **TEMPLATES** : Template adalah file HTML yang digunakan untuk menentukan tata letak dan struktur halaman web. Dalam Django, template mengandung elemen-elemen HTML yang dapat diisi dengan data dinamis dari model atau view. Ini memisahkan logika tampilan dari tata letak halaman



```

1 <!-- Static Files -->
2 <!-- Load fonts -->
3 <!-- Load CSS -->
4 <!-- Load JS -->
5 <!-- Load Icons -->
6 <!-- Load Fonts -->
7 <!-- Load CSS -->
8 <!-- Load JS -->
9 <!-- Load Icons -->
10 <!-- Load Fonts -->
11 <!-- Load CSS -->
12 <!-- Load JS -->
13 <!-- Load Icons -->
14 <!-- Load Fonts -->
15 <!-- Load CSS -->
16 <!-- Load JS -->
17 <!-- Load Icons -->
18 <!-- Load Fonts -->
19 <!-- Load CSS -->
20 <!-- Load JS -->
21 <!-- Load Icons -->
22 <!-- Load Fonts -->
23 <!-- Load CSS -->
24 <!-- Load JS -->
25 <!-- Load Icons -->
26 <!-- Load Fonts -->
27 <!-- Load CSS -->
28 <!-- Load JS -->
29 <!-- Load Icons -->
30 <!-- Load Fonts -->
31 <!-- Load CSS -->
32 <!-- Load JS -->
33 <!-- Load Icons -->
34 <!-- Load Fonts -->
35 <!-- Load CSS -->
36 <!-- Load JS -->
37 <!-- Load Icons -->
38 <!-- Load Fonts -->
39 <!-- Load CSS -->
40 <!-- Load JS -->
41 <!-- Load Icons -->
42 <!-- Load Fonts -->
43 <!-- Load CSS -->
44 <!-- Load JS -->
45 <!-- Load Icons -->
46 <!-- Load Fonts -->
47 <!-- Load CSS -->
48 <!-- Load JS -->
49 <!-- Load Icons -->
50 <!-- Load Fonts -->
51 <!-- Load CSS -->
52 <!-- Load JS -->
53 <!-- Load Icons -->
54 <!-- Load Fonts -->
55 <!-- Load CSS -->
56 <!-- Load JS -->
57 <!-- Load Icons -->
58 <!-- Load Fonts -->
59 <!-- Load CSS -->
60 <!-- Load JS -->
61 <!-- Load Icons -->
62 <!-- Load Fonts -->
63 <!-- Load CSS -->
64 <!-- Load JS -->
65 <!-- Load Icons -->
66 <!-- Load Fonts -->
67 <!-- Load CSS -->
68 <!-- Load JS -->
69 <!-- Load Icons -->
70 <!-- Load Fonts -->
71 <!-- Load CSS -->
72 <!-- Load JS -->
73 <!-- Load Icons -->
74 <!-- Load Fonts -->
75 <!-- Load CSS -->
76 <!-- Load JS -->
77 <!-- Load Icons -->
78 <!-- Load Fonts -->
79 <!-- Load CSS -->
80 <!-- Load JS -->
81 <!-- Load Icons -->
82 <!-- Load Fonts -->
83 <!-- Load CSS -->
84 <!-- Load JS -->
85 <!-- Load Icons -->
86 <!-- Load Fonts -->
87 <!-- Load CSS -->
88 <!-- Load JS -->
89 <!-- Load Icons -->
90 <!-- Load Fonts -->
91 <!-- Load CSS -->
92 <!-- Load JS -->
93 <!-- Load Icons -->
94 <!-- Load Fonts -->
95 <!-- Load CSS -->
96 <!-- Load JS -->
97 <!-- Load Icons -->
98 <!-- Load Fonts -->
99 <!-- Load CSS -->
100 <!-- Load JS -->

```

- **URLS** : Untuk mengaitkan permintaan HTTP yang masuk dengan views tertentu. Mereka mengarahkan aplikasi untuk menanggapi permintaan dengan mengaitkan URL tertentu dengan fungsi atau kelas **view** yang sesuai. Peta URL membantu Django menentukan apa yang harus dilakukan dengan permintaan tertentu

```

1 from django.urls import path
2 from . import views
3
4 urlpatterns = [
5     path('', views.index, name='index'),
6     path('dashboard/', views.dashboard, name='dashboard'),
7     path('signup/', views.SignupPage, name='signup'),
8     path('signin/', views.SigninPage, name='signin'),
9     path('signout/', views.SignoutPage, name='signout'),
10
11     # Kategori
12     path('kategori/', views.list_kategori, name='list_kategori'),
13     path('kategori/create/', views.create_kategori, name='create_kategori'),
14     path('kategori/update/<int:kategori_id>', views.update_kategori, name='update_kategori'),
15     path('kategori/delete/<int:kategori_id>', views.delete_kategori, name='delete_kategori'),
16
17     # Buku
18     path('buku/', views.list_buku, name='list_buku'),
19     path('buku/create/', views.create_buku, name='create_buku'),
20     path('buku/update/<int:buku_id>', views.update_buku, name='update_buku'),
21     path('buku/delete/<int:buku_id>', views.delete_buku, name='delete_buku'),
22
23     # Anggota
24     path('anggota/', views.list_anggota, name='list_anggota'),
25     path('anggota/create/', views.create_anggota, name='create_anggota'),
26     path('anggota/update/<int:anggota_id>', views.update_anggota, name='update_anggota'),
27     path('anggota/delete/<int:anggota_id>', views.delete_anggota, name='delete_anggota'),
28
29     # Peminjaman
30     path('peminjaman/', views.list_peminjaman, name='list_peminjaman'),
31     path('peminjaman/create/', views.create_peminjaman, name='create_peminjaman'),
32     path('peminjaman/update/<int:peminjaman_id>', views.update_peminjaman, name='update_peminjaman'),
33     path('peminjaman/delete/<int:peminjaman_id>', views.delete_peminjaman, name='delete_peminjaman'),
34
35     # Petugas
36     path('petugas/', views.list_petugas, name='list_petugas'),
37     path('petugas/create/', views.create_petugas, name='create_petugas'),
38     path('petugas/update/<int:petugas_id>', views.update_petugas, name='update_petugas'),
39     path('petugas/delete/<int:petugas_id>', views.delete_petugas, name='delete_petugas')
40 ]

```

```

14 # ANGOTA
15 # Fungsi untuk menampilkan Anggota
16
17 def list_anggota(request):
18     anggota = Anggota.objects.all()
19     search_query = request.GET.get('cari')
20
21     if search_query:
22         anggota_serc = Anggota.objects.filter(
23             Q(user__username__icontains=search_query) | Q(alamat__icontains=search_query)
24         )
25     else:
26         anggota_serc = anggota
27
28     paginator = Paginator(anggota_serc, 5)
29     page_number = request.GET.get('page')
30     page_obj = paginator.get_page(page_number)
31
32     user = User.objects.all()
33     recent_anggotas = Anggota.objects.filter(created_at__gte=timezone.now())
34     context = {
35         'recent_anggotas': recent_anggotas,
36         'anggotas': page_obj,
37         'user': user,
38     }
39     return render(request, 'anggota.html', context)
40
41 # Fungsi untuk menambahkan anggota baru
42 @login_required
43 def create_anggota(request):
44     # Fungsi untuk memperbarui anggota
45     @login_required
46     def update_anggota(request, anggota_id):
47         anggota = get_object_or_404(Anggota, id=anggota_id)
48
49         if request.method == 'POST':
50             form = AnggotaForm(request.POST, request.FILES, instance=anggota)
51             if form.is_valid():
52                 form.save()
53                 return redirect('list_anggota')
54             else:
55                 return render(request, 'anggota.html', {'form': form, 'anggota': anggota})
56         else:
57             return render(request, 'anggota.html', {'anggota': anggota})
58     return update_anggota

```

- **Models** mendefinisikan struktur data di database.
- **Views** mengelola logika aplikasi dan menentukan cara data ditampilkan.
- **Templates** menyusun tampilan HTML untuk presentasi data.
- **URLs** mengarahkan permintaan ke views yang tepat berdasarkan pola URL tertentu