**LAPORAN PROJEK DJANGO**

**MANAJEMEN CUCI KENDARAAN**



**Dosen Pengampu :**

Fitri Damyati, S.Kom, MM

**Disusun oleh :**

Firdaus Satrio Utomo (3337230039)

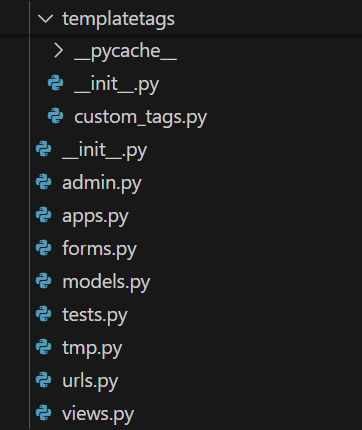
Sirrul Fatih Ahdiat (3337230050)

Nurul Santi Hafifah (3337230019)

**JURUSAN TEKNIK INFORMATIKA FAKULTAS TEKNIK**

**UNIVERSITAS SULTAN AGENG TIRTAYASA**

**2024**

1. **Direktori Program**

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

1. **Penjelasan Direktori**

a). cucikendaraan

1. Images

Direktori ini berisi gambar-gambar yang digunakan dalam aplikasi web, seperti logo, ikon, atau gambar produk. Gambar ini biasanya diakses dari template HTML untuk ditampilkan kepada pengguna.

1. Migrations

Berisi skrip migrasi database yang dikelola oleh Django untuk mengubah skema database. Isi dari direktori migrasi ini antara lain

1. 0001\_initial.py

# Generated by Django 5.0.6 on 2024-05-31 07:33

from django.db import migrations, models

class Migration(migrations.Migration):

    initial = True

    dependencies = [

    ]

    operations = [

        migrations.CreateModel(

            name='User',

            fields=[

                ('id', models.BigAutoField(auto\_created=True, primary\_key=True, serialize=False, verbose\_name='ID')),

                ('last\_login', models.DateTimeField(blank=True, null=True, verbose\_name='last login')),

                ('username', models.CharField(max\_length=100, unique=True)),

                ('nama', models.CharField(max\_length=100)),

                ('password', models.CharField(max\_length=128)),

            ],

            options={

                'abstract': False,

            },

        ),

    ]

Migrasi ini mendefinisikan pembuatan tabel baru bernama User di dalam database. Tabel User ini memiliki beberapa field, yaitu id, last\_login, username, nama, dan password. Field id adalah primary key otomatis dengan tipe data BigAutoField, yang secara otomatis akan diisi dan diinkrementasi oleh sistem. Field last\_login adalah DateTimeField yang dapat bernilai null dan dapat dibiarkan kosong, digunakan untuk mencatat tanggal dan waktu login terakhir pengguna. Field username adalah CharField dengan panjang maksimum 100 karakter dan memiliki constraint unique=True, yang berarti setiap nilai dalam field ini harus unik. Field nama juga merupakan CharField dengan panjang maksimum 100 karakter, digunakan untuk menyimpan nama lengkap pengguna. Field password adalah CharField dengan panjang maksimum 128 karakter, digunakan untuk menyimpan kata sandi pengguna. Migrasi ini memiliki atribut initial yang diset ke True, menandakan bahwa ini adalah migrasi awal untuk aplikasi tersebut dan tidak memiliki dependensi pada migrasi lainnya. Pada opsi model, diset abstract ke False, yang berarti model User ini bukanlah model abstrak dan akan dibuatkan tabel fisiknya di dalam database.

1. 0002\_usercucikuy\_delete\_user.py

# Generated by Django 5.0.6 on 2024-05-31 15:09

from django.db import migrations, models

class Migration(migrations.Migration):

    dependencies = [

        ('cucikendaraan', '0001\_initial'),

    ]

    operations = [

        migrations.CreateModel(

            name='UserCuciKuy',

            fields=[

                ('id', models.BigAutoField(auto\_created=True, primary\_key=True, serialize=False, verbose\_name='ID')),

                ('nama', models.CharField(max\_length=100)),

                ('username', models.CharField(max\_length=150, unique=True)),

                ('password', models.CharField(max\_length=255)),

            ],

        ),

        migrations.DeleteModel(

            name='User',

        ),

    ]

Migrasi Django yang dihasilkan di atas mencerminkan perubahan pada model basis data aplikasi. Migrasi ini bergantung pada migrasi awal dari aplikasi cucikendaraan (0001\_initial). Dalam migrasi ini, model baru bernama UserCuciKuy dibuat. Model ini memiliki tiga kolom: nama, username, dan password. Kolom id adalah kunci utama yang dihasilkan secara otomatis oleh Django (BigAutoField). Kolom nama adalah tipe CharField dengan batas maksimum 100 karakter, sementara username adalah CharField dengan batas maksimum 150 karakter dan diberi atribut unique=True untuk memastikan setiap username bersifat unik. Kolom password juga merupakan CharField dengan batas maksimum 255 karakter, yang akan menyimpan kata sandi pengguna. Selain itu, migrasi ini juga menghapus model sebelumnya bernama User. Ini mungkin mencerminkan restrukturisasi atau perubahan nama model untuk keperluan tertentu, seperti penyesuaian pada skema basis data atau klarifikasi nama model agar lebih spesifik dan jelas.

1. 0003\_usercucikuy\_last\_login.py

# Generated by Django 5.0.6 on 2024-05-31 15:22

import django.utils.timezone

from django.db import migrations, models

class Migration(migrations.Migration):

    dependencies = [

        ('cucikendaraan', '0002\_usercucikuy\_delete\_user'),

    ]

    operations = [

        migrations.AddField(

            model\_name='usercucikuy',

            name='last\_login',

            field=models.DateTimeField(default=django.utils.timezone.now, null=True),

        ),

    ]

Migrasi Django yang kedua ini adalah kelanjutan dari migrasi sebelumnya dan mencerminkan perubahan lebih lanjut pada model UserCuciKuy dalam aplikasi cucikendaraan. Migrasi ini bergantung pada migrasi sebelumnya 0002\_usercucikuy\_delete\_user. Dalam migrasi ini, ditambahkan sebuah field baru bernama last\_login ke model UserCuciKuy. Field ini adalah DateTimeField, yang digunakan untuk menyimpan informasi waktu dan tanggal. Field ini memiliki dua atribut tambahan: default dan null. Atribut default diatur ke django.utils.timezone.now, yang berarti bahwa jika tidak ada nilai yang diberikan saat membuat instance baru dari UserCuciKuy, maka secara otomatis akan diisi dengan waktu saat ini. Atribut null=True memungkinkan field ini untuk memiliki nilai null, yang berarti bahwa tidak wajib diisi dan dapat dibiarkan kosong. Penambahan field last\_login ini kemungkinan besar dimaksudkan untuk melacak waktu terakhir kali pengguna login ke sistem, yang dapat digunakan untuk berbagai tujuan, seperti keamanan, analisis penggunaan, atau fitur aplikasi lainnya.

1. 0004\_usercucikuy\_groups\_usercucikuy\_is\_staff\_and\_more.py

# Generated by Django 5.0.6 on 2024-05-31 15:36

from django.db import migrations, models

class Migration(migrations.Migration):

    dependencies = [

        ('auth', '0012\_alter\_user\_first\_name\_max\_length'),

        ('cucikendaraan', '0003\_usercucikuy\_last\_login'),

    ]

    operations = [

        migrations.AddField(

            model\_name='usercucikuy',

            name='groups',

            field=models.ManyToManyField(blank=True, help\_text='The groups this user belongs to. A user will get all permissions granted to each of their groups.', related\_name='user\_set', related\_query\_name='user', to='auth.group', verbose\_name='groups'),

        ),

        migrations.AddField(

            model\_name='usercucikuy',

            name='is\_staff',

            field=models.BooleanField(default=False),

        ),

        migrations.AddField(

            model\_name='usercucikuy',

            name='is\_superuser',

            field=models.BooleanField(default=False, help\_text='Designates that this user has all permissions without explicitly assigning them.', verbose\_name='superuser status'),

        ),

        migrations.AddField(

            model\_name='usercucikuy',

            name='user\_permissions',

            field=models.ManyToManyField(blank=True, help\_text='Specific permissions for this user.', related\_name='user\_set', related\_query\_name='user', to='auth.permission', verbose\_name='user permissions'),

        ),

        migrations.AlterField(

            model\_name='usercucikuy',

            name='last\_login',

            field=models.DateTimeField(blank=True, null=True, verbose\_name='last login'),

        ),

    ]

Migrasi Django ini merupakan kelanjutan dari migrasi sebelumnya yang berkaitan dengan model UserCuciKuy dalam aplikasi cucikendaraan. Migrasi ini bergantung pada dua migrasi sebelumnya, yaitu dari aplikasi auth dan cucikendaraan. Dalam migrasi ini, beberapa field baru ditambahkan ke model UserCuciKuy untuk meningkatkan fungsionalitas pengguna. Field-field tersebut meliputi:

1. groups: Field ini adalah ManyToManyField yang menghubungkan model UserCuciKuy dengan model Group dari aplikasi auth. Field ini memungkinkan pengguna untuk menjadi anggota dari satu atau lebih grup, yang akan memberikan mereka izin yang ditetapkan untuk masing-masing grup. Field ini bersifat opsional (blank=True) dan dilengkapi dengan teks bantuan serta pengaturan nama relasi terkait (related\_name dan related\_query\_name).

2. is\_staff: Field ini adalah BooleanField yang digunakan untuk menunjukkan apakah pengguna tersebut adalah staf administrasi. Default nilai untuk field ini adalah False.

3. is\_superuser: Field ini adalah BooleanField yang digunakan untuk menunjukkan apakah pengguna memiliki semua izin tanpa perlu menetapkannya secara eksplisit. Field ini juga memiliki teks bantuan dan verbose\_name yang menjelaskan fungsinya. Default nilai untuk field ini adalah False.

4. user\_permissions: Seperti groups, field ini juga merupakan ManyToManyField yang menghubungkan model UserCuciKuy dengan model Permission dari aplikasi auth. Field ini memungkinkan penetapan izin spesifik ke pengguna individual dan bersifat opsional (blank=True), dengan teks bantuan dan pengaturan nama relasi terkait. Selain penambahan field baru, migrasi ini juga mengubah atribut dari field last\_login yang telah ada. Sekarang, field ini memiliki verbose\_name 'last login' dan diatur agar bisa dibiarkan kosong (blank=True dan null=True), yang memudahkan penyimpanan informasi waktu login terakhir pengguna. Penambahan dan perubahan ini menunjukkan bahwa model UserCuciKuy sedang disesuaikan untuk mengakomodasi fitur manajemen pengguna yang lebih canggih, termasuk pengelompokan pengguna, izin akses, dan status administrasi.

e. 0005\_transaction.py

# Generated by Django 5.0.6 on 2024-05-31 17:03

from django.db import migrations, models

class Migration(migrations.Migration):

    dependencies = [

        ('cucikendaraan', '0004\_usercucikuy\_groups\_usercucikuy\_is\_staff\_and\_more'),

    ]

    operations = [

        migrations.CreateModel(

            name='Transaction',

            fields=[

                ('transaction\_date', models.DateField()),

                ('pelanggan', models.CharField(max\_length=100)),

                ('jeniskendaraan', models.CharField(choices=[('motor', 'Motor'), ('mobil', 'Mobil')], max\_length=50)),

                ('idtransaction', models.AutoField(primary\_key=True, serialize=False)),

            ],

        ),

    ]

Migrasi Django ini memperkenalkan model baru bernama Transaction dalam aplikasi cucikendaraan. Migrasi ini bergantung pada migrasi sebelumnya yang mengatur berbagai atribut pada model UserCuciKuy. Model Transaction memiliki beberapa field sebagai berikut:

1. transaction\_date: Field ini adalah DateField yang digunakan untuk menyimpan tanggal terjadinya transaksi.

2. pelanggan: Field ini adalah CharField dengan panjang maksimum 100 karakter, digunakan untuk menyimpan nama pelanggan yang melakukan transaksi.

3. jeniskendaraan: Field ini adalah CharField dengan panjang maksimum 50 karakter dan pilihan (choices) yang terbatas. Pilihannya adalah 'Motor' dan 'Mobil', untuk menunjukkan jenis kendaraan yang terkait dengan transaksi tersebut.

4. idtransaction: Field ini adalah AutoField yang berfungsi sebagai kunci utama untuk model Transaction. Field ini akan menghasilkan nilai integer yang unik secara otomatis untuk setiap entri baru, dan ditetapkan sebagai primary\_key.

Penambahan model Transaction ini bertujuan untuk mencatat dan mengelola data transaksi di dalam sistem aplikasi cucikendaraan. Model ini memungkinkan penyimpanan informasi penting mengenai transaksi yang dilakukan oleh pelanggan, termasuk tanggal transaksi, nama pelanggan, dan jenis kendaraan yang dicuci. Dengan adanya model ini, pengelolaan data transaksi menjadi lebih terstruktur dan mudah diakses.

1. 0006\_price\_transaction\_price.py

# Generated by Django 5.0.6 on 2024-06-04 05:17

import django.db.models.deletion

from django.db import migrations, models

class Migration(migrations.Migration):

    dependencies = [

        ('cucikendaraan', '0005\_transaction'),

    ]

    operations = [

        migrations.CreateModel(

            name='Price',

            fields=[

                ('id', models.BigAutoField(auto\_created=True, primary\_key=True, serialize=False, verbose\_name='ID')),

                ('jeniskendaraan', models.CharField(choices=[('motor', 'Motor'), ('mobil', 'Mobil')], max\_length=50)),

                ('harga', models.DecimalField(decimal\_places=2, max\_digits=10)),

            ],

        ),

        migrations.AddField(

            model\_name='transaction',

            name='price',

            field=models.ForeignKey(blank=True, null=True, on\_delete=django.db.models.deletion.CASCADE, to='cucikendaraan.price'),

        ),

    ]

Migrasi Django ini mencatat dua operasi utama terkait model dalam aplikasi cucikendaraan. Pertama, sebuah model baru dengan nama Price telah dibuat. Model ini memiliki tiga field: id sebagai kunci utama, jeniskendaraan sebagai CharField dengan pilihan ('motor' dan 'mobil') yang menunjukkan jenis kendaraan, dan harga sebagai DecimalField untuk menyimpan harga layanan cuci kendaraan dengan dua digit desimal. Selanjutnya, dalam operasi kedua, field baru bernama price telah ditambahkan ke model Transaction. Field ini adalah foreign key yang menghubungkan setiap transaksi dengan harga yang sesuai dari model Price. Pengaturan on\_delete=django.db.models.deletion.CASCADE menunjukkan bahwa jika entri harga terkait dihapus, maka semua transaksi yang terkait dengan harga tersebut juga akan dihapus secara otomatis, sehingga menjaga konsistensi referensial dalam basis data. Field ini bersifat opsional (blank=True dan null=True), yang berarti transaksi dapat dilakukan tanpa terkait dengan harga tertentu.

1. 0007\_alter\_price\_jeniskendaraan.py

# Generated by Django 5.0.6 on 2024-06-04 12:39

from django.db import migrations, models

class Migration(migrations.Migration):

    dependencies = [

        ('cucikendaraan', '0006\_price\_transaction\_price'),

    ]

    operations = [

        migrations.AlterField(

            model\_name='price',

            name='jeniskendaraan',

            field=models.CharField(choices=[('motor', 'Motor'), ('mobil', 'Mobil')], max\_length=50, unique=True),

        ),

    ]

Migrasi Django ini adalah perubahan pada model Price dalam aplikasi cucikendaraan. Migrasi ini bergantung pada migrasi sebelumnya yang memperkenalkan atribut harga ke model Transaction. Perubahan utama dalam migrasi ini adalah pengaturan ulang field jeniskendaraan pada model Price. Field jeniskendaraan, yang merupakan CharField dengan panjang maksimum 50 karakter, kini memiliki atribut tambahan unique=True. Selain itu, pilihan untuk field ini tetap sama, yaitu terbatas pada dua opsi: 'Motor' dan 'Mobil'. Dengan penambahan atribut unique=True, setiap jenis kendaraan dalam model Price sekarang harus memiliki entri yang unik. Artinya, tidak akan ada dua entri dengan jenis kendaraan yang sama dalam tabel Price. Hal ini membantu memastikan bahwa tidak ada duplikasi harga untuk jenis kendaraan yang sama, sehingga memudahkan pengelolaan dan validasi data harga dalam sistem. Secara keseluruhan, migrasi ini bertujuan untuk meningkatkan integritas data pada model Price dengan memastikan setiap jenis kendaraan hanya memiliki satu set harga yang unik.

1. Templates
2. datamaster.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Data Master - CuciKUY!</title>

    <link

      href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css"

      rel="stylesheet"

    />

    <link

      rel="stylesheet"

      href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css"

      integrity="sha512-Fo3rlrZj/k7ujTnHg4CGR2DnvF+B/6D9o1fVfHHY5EbpiE7VqOJfJFyJFZm+pGzXQxz0TckqDA4/PmXx8kUp2g=="

      crossorigin="anonymous"

      referrerpolicy="no-referrer"

    />

  </head>

  <body class="bg-green-300">

    {% include 'nav.html' %}

    <div class="container mx-auto py-4 px-2">

      <!-- Buttons to toggle forms -->

      <div class="flex justify-between mb-4">

        <button

          onclick="toggleForm('userForm')"

          class="bg-green-600 hover:bg-green-800 text-white px-2 py-1 rounded-md transition duration-300 ease-in-out transform hover:scale-105 shadow-lg flex"

        >

          Tambah User

        </button>

      </div>

      <div id="userForm" class="bg-white p-6 shadow-md rounded-lg hidden mb-8">

        <h2 class="text-xl font-semibold mb-4 text-green-700">Tambah User</h2>

        <form method="post" class="bg-white p-4 shadow-md rounded-lg">

          {% csrf\_token %}

          <div class="mb-4">

            <label for="username" class="block text-sm font-bold mb-2"

              >Username:</label

            >

            {{ user\_form.username }}

          </div>

          <div class="mb-4">

            <label for="nama" class="block text-sm font-bold mb-2">Nama:</label>

            {{ user\_form.nama }}

          </div>

          <div class="mb-4">

            <label for="nama" class="block text-sm font-bold mb-2"

              >Password:</label

            >

            {{ user\_form.password }}

          </div>

          <button

            type="submit"

            class="bg-green-500 hover:bg-green-800 text-white px-2 py-1 rounded-md transition duration-300 ease-in-out transform hover:scale-105 shadow-lg flex"

          >

            Tambah User

          </button>

        </form>

      </div>

      <!-- Tabel Daftar Harga -->

      <div class="bg-white p-6 shadow-md rounded-lg mt-8">

        <h2 class="text-xl font-semibold mb-4">Daftar Harga</h2>

        <table class="w-full table-auto">

          <thead class="bg-gray-200">

            <tr>

              <th class="border px-4 py-2">Jenis Kendaraan</th>

              <th class="border px-4 py-2">Harga</th>

              <th class="border px-4 py-2">Aksi</th>

            </tr>

          </thead>

          <tbody>

            {% for price in prices %}

            <tr

              class="bg-white hover:bg-gray-100 transition duration-200"

              data-id="{{ price.id }}"

              data-jeniskendaraan="{{ price.jeniskendaraan }}"

              data-harga="{{ price.harga }}"

            >

              <td class="border px-4 py-2">{{ price.jeniskendaraan }}</td>

              <td class="border px-4 py-2">{{ price.harga }}</td>

              <td class="border px-2 py-2 rounded-lg">

                <div class="flex space-x-3 w-full">

                  <button

                    data-id="{{ price.id }}"

                    onclick="openEditPriceModal(this)"

                    class="bg-blue-600 hover:bg-blue-800 text-white px-2 py-1 rounded-md transition duration-300 ease-in-out transform hover:scale-105 shadow-lg flex items-center flex-grow"

                  >

                    <i class="fas fa-edit"></i> Edit

                  </button>

                  <!-- <button

                    data-id="{{ price.id }}"

                    onclick="openDeletePriceModal(this)"

                    class="bg-red-600 hover:bg-red-800 text-white px-2 py-1 rounded-md transition duration-300 ease-in-out transform hover:scale-105 shadow-lg flex items-center flex-grow"

                  >

                    <i class="fas fa-trash-alt"></i> Hapus

                  </button> -->

                </div>

              </td>

            </tr>

            {% endfor %}

          </tbody>

        </table>

      </div>

      <!-- Tabel Daftar User -->

      <div class="bg-white p-6 shadow-md rounded-lg mt-8">

        <h2 class="text-xl font-semibold mb-4">Daftar User</h2>

        <table class="w-full table-auto">

          <thead class="bg-gray-200">

            <tr>

              <th class="border px-4 py-2">Username</th>

              <th class="border px-4 py-2">Nama</th>

              <th class="border px-4 py-2">Password</th>

              <th class="border px-4 py-2">Aksi</th>

            </tr>

          </thead>

          <tbody>

            {% for user in users %}

            <tr

              class="bg-white hover:bg-gray-100 transition duration-200"

              data-id="{{ user.id }}"

              data-username="{{ user.username }}"

              data-nama="{{ user.nama }}"

            >

              <td class="border px-4 py-2">{{ user.username }}</td>

              <td class="border px-4 py-2">{{ user.nama }}</td>

              <td class="border px-4 py-2">\*\*\*\*\*\*\*\*</td>

              <td class="border px-2 py-2 rounded-lg">

                <div class="flex space-x-3 w-full">

                  <button

                    data-id="{{ user.id }}"

                    onclick="openEditUserModal(this)"

                    class="bg-blue-600 hover:bg-blue-800 text-white px-2 py-1 rounded-md transition duration-300 ease-in-out transform hover:scale-105 shadow-lg flex items-center flex-grow"

                  >

                    <i class="fas fa-edit"></i> Edit

                  </button>

                  <button

                    data-id="{{ user.id }}"

                    onclick="openDeleteUserModal(this)"

                    class="bg-red-600 hover:bg-red-800 text-white px-2 py-1 rounded-md transition duration-300 ease-in-out transform hover:scale-105 shadow-lg flex items-center flex-grow"

                  >

                    <i class="fas fa-trash-alt"></i> Hapus

                  </button>

                </div>

              </td>

            </tr>

            {% endfor %}

          </tbody>

        </table>

      </div>

    </div>

    <!-- Modal untuk Edit User -->

    <div

      id="editUserModal"

      class="hidden fixed inset-0 bg-gray-600 bg-opacity-50 overflow-y-auto h-full w-full"

    >

      <div

        class="relative top-20 mx-auto p-5 border w-96 shadow-lg rounded-md bg-white"

      >

        <div class="mt-3 text-center">

          <h3 class="text-lg leading-6 font-medium text-gray-900">Edit User</h3>

          <div class="mt-2 px-7 py-3">

            <form id="editUserForm" method="post">

              {% csrf\_token %}

              <input type="hidden" name="user\_id" id="edit\_user\_id" />

              <div class="mb-4">

                <label

                  for="edit\_username"

                  class="block text-sm font-medium text-gray-700"

                  >Username</label

                >

                <input

                  type="text"

                  name="edit\_username"

                  id="edit\_username"

                  class="mt-1 block w-full px-3 py-2 border border-gray-300 rounded-md shadow-sm focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm"

                />

              </div>

              <div class="mb-4">

                <label

                  for="edit\_nama"

                  class="block text-sm font-medium text-gray-700"

                  >Nama</label

                >

                <input

                  type="text"

                  name="edit\_nama"

                  id="edit\_nama"

                  class="mt-1 block w-full px-3 py-2 border border-gray-300 rounded-md shadow-sm focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm"

                />

              </div>

              <button

                type="submit"

                class="bg-blue-500 hover:bg-blue-600 text-white px-4 py-2 rounded mt-2"

              >

                Simpan Perubahan

              </button>

              <button

                type="button"

                onclick="closeEditUserModal()"

                class="bg-gray-500 hover:bg-gray-600 text-white px-4 py-2 rounded mt-2"

              >

                Batal

              </button>

            </form>

          </div>

        </div>

      </div>

    </div>

    <!-- Modal untuk Edit Harga -->

    <div

      id="editPriceModal"

      class="hidden fixed inset-0 bg-gray-600 bg-opacity-50 overflow-y-auto h-full w-full"

    >

      <div

        class="relative top-20 mx-auto p-5 border w-96 shadow-lg rounded-md bg-white"

      >

        <div class="mt-3 text-center">

          <h3 class="text-lg leading-6 font-medium text-gray-900">

            Edit Harga

          </h3>

          <div class="mt-2 px-7 py-3">

            <form id="editPriceForm" method="post">

              {% csrf\_token %}

              <input type="hidden" name="price\_id" id="edit\_price\_id" />

              <div class="mb-4">

                <label

                  for="edit\_jeniskendaraan"

                  class="block text-sm font-medium text-gray-700"

                  >Jenis Kendaraan</label

                >

                <input

                  type="text"

                  name="edit\_jeniskendaraan"

                  id="edit\_jeniskendaraan"

                  class="mt-1 block w-full px-3 py-2 border border-gray-300 rounded-md shadow-sm focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm"

                />

              </div>

              <div class="mb-4">

                <label

                  for="edit\_harga"

                  class="block text-sm font-medium text-gray-700"

                  >Harga</label

                >

                <input

                  type="text"

                  name="edit\_harga"

                  id="edit\_harga"

                  class="mt-1 block w-full px-3 py-2 border border-gray-300 rounded-md shadow-sm focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm"

                />

              </div>

              <button

                type="submit"

                class="bg-blue-500 hover:bg-blue-600 text-white px-4 py-2 rounded mt-2"

              >

                Simpan Perubahan

              </button>

              <button

                type="button"

                onclick="closeEditPriceModal()"

                class="bg-gray-500 hover:bg-gray-600 text-white px-4 py-2 rounded mt-2"

              >

                Batal

              </button>

            </form>

          </div>

        </div>

      </div>

    </div>

    <!-- Modal untuk konfirmasi penghapusan -->

    <div

      id="deleteModal"

      class="hidden fixed inset-0 bg-gray-600 bg-opacity-50 overflow-y-auto h-full w-full"

    >

      <div

        class="relative top-20 mx-auto p-5 border w-96 shadow-lg rounded-md bg-white"

      >

        <div class="mt-3 text-center">

          <h3 class="text-lg leading-6 font-medium text-gray-900">

            Konfirmasi Hapus

          </h3>

          <div class="mt-2 px-7 py-3">

            <p class="text-sm text-gray-500">

              Apakah Anda yakin ingin menghapus item ini?

            </p>

          </div>

          <div class="items-center px-4 py-3">

            <form id="deleteForm" method="post">

              {% csrf\_token %}

              <input type="hidden" name="delete\_user\_id" id="delete\_user\_id" />

              <input

                type="hidden"

                name="delete\_price\_id"

                id="delete\_price\_id"

              />

              <button

                id="confirmDelete"

                class="bg-red-500 hover:bg-red-600 text-white px-4 py-2 rounded"

              >

                Hapus

              </button>

              <button

                type="button"

                onclick="closeDeleteModal()"

                class="bg-gray-500 hover:bg-gray-600 text-white px-4 py-2 rounded ml-2"

              >

                Batal

              </button>

            </form>

          </div>

        </div>

      </div>

    </div>

    <script

      src="https://kit.fontawesome.com/3e1ec18e22.js"

      crossorigin="anonymous"

    ></script>

    <script>

      function toggleForm(formId) {

        const forms = document.querySelectorAll("#userForm, #priceForm");

        forms.forEach((form) => {

          if (form.id === formId) {

            form.classList.toggle("hidden");

          } else {

            form.classList.add("hidden");

          }

        });

      }

      function openEditUserModal(button) {

        const tr = button.closest("tr");

        const id = tr.dataset.id;

        const username = tr.dataset.username;

        const nama = tr.dataset.nama;

        document.getElementById("edit\_user\_id").value = id;

        document.getElementById("edit\_username").value = username;

        document.getElementById("edit\_nama").value = nama;

        document.getElementById("editUserModal").classList.remove("hidden");

      }

      function closeEditUserModal() {

        document.getElementById("editUserModal").classList.add("hidden");

      }

      function openEditPriceModal(button) {

        const tr = button.closest("tr");

        const id = tr.dataset.id;

        const jeniskendaraan = tr.dataset.jeniskendaraan;

        const harga = tr.dataset.harga;

        document.getElementById("edit\_price\_id").value = id;

        document.getElementById("edit\_jeniskendaraan").value = jeniskendaraan;

        document.getElementById("edit\_harga").value = harga;

        document.getElementById("editPriceModal").classList.remove("hidden");

      }

      function closeEditPriceModal() {

        document.getElementById("editPriceModal").classList.add("hidden");

      }

      function openDeleteUserModal(button) {

        const id = button.dataset.id;

        document.getElementById("delete\_user\_id").value = id;

        document.getElementById("delete\_price\_id").value = "";

        document.getElementById("deleteModal").classList.remove("hidden");

      }

      function openDeletePriceModal(button) {

        const id = button.dataset.id;

        document.getElementById("delete\_price\_id").value = id;

        document.getElementById("delete\_user\_id").value = "";

        document.getElementById("deleteModal").classList.remove("hidden");

      }

      function closeDeleteModal() {

        document.getElementById("deleteModal").classList.add("hidden");

      }

    </script>

  </body>

</html>

Kode HTML di atas adalah bagian dari tampilan web untuk aplikasi "CuciKUY!". Halaman ini menyediakan antarmuka pengguna untuk mengelola data master, seperti menambahkan dan mengedit pengguna (users) serta menampilkan dan mengubah harga (prices) untuk layanan cuci kendaraan.

Halaman ini terdiri dari beberapa bagian utama:

1. Navigasi: Di bagian atas halaman, ada navigasi yang mungkin berisi menu dan tautan ke bagian lain dari aplikasi.

2. Form Tambah User: Terdapat form untuk menambahkan pengguna baru. Ketika tombol "Tambah User" ditekan, form ini muncul, memungkinkan pengguna untuk memasukkan username, nama, dan password baru.

3. Daftar Harga: Tabel yang menampilkan daftar harga untuk layanan cuci kendaraan. Setiap baris menampilkan jenis kendaraan dan harga terkait. Terdapat juga tombol "Edit" untuk mengubah harga dan menghapusnya.

4. Daftar User: Tabel yang menampilkan daftar pengguna. Setiap baris menampilkan username, nama, dan tombol "Edit" serta "Hapus" untuk mengelola pengguna.

5. Modal untuk Edit User dan Edit Harga: Ketika tombol "Edit" ditekan di sebelah item, muncul modal yang memungkinkan pengguna untuk mengubah informasi pengguna atau harga. Pengguna dapat memasukkan informasi baru dan menyimpan perubahan atau membatalkan proses.

6. Modal untuk Konfirmasi Penghapusan: Ketika tombol "Hapus" ditekan, muncul modal konfirmasi yang meminta pengguna untuk mengonfirmasi penghapusan item. Pengguna dapat memilih untuk melanjutkan penghapusan atau membatalkannya. Script JavaScript digunakan untuk mengatur perilaku dinamis dari halaman, seperti menampilkan dan menyembunyikan form atau modal, dan mengisi informasi pada modal edit. Ini memungkinkan antarmuka yang interaktif dan responsif bagi pengguna.

b. index.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Admin Panel - CuciKUY!</title>

    <link

      href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css"

      rel="stylesheet"

    />

    <script src="https://cdn.jsdelivr.net/npm/chart.js@2.9.4/dist/Chart.min.js"></script>

  </head>

  <body class="bg-green-300">

    {% include 'nav.html' %}

    <div class="container mx-auto py-8 px-4">

      <div class="bg-green-500 shadow-md rounded-lg p-6">

        <h1 class="text-2xl text-white font-bold mb-4">

          Selamat Datang Admin {{ request.user }} - CuciKUY!

        </h1>

        <p class="mb-4 text-white">

          Kelola pelanggan dan layanan Anda dengan efisien.

        </p>

        <div class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-2 gap-4">

          <a

            href="{% url 'transaksi' %}"

            class="block p-4 bg-gray-100 hover:bg-gray-200 rounded-md transition duration-300"

          >

            <h2 class="text-lg font-semibold mb-2">Manajemen Transaksi</h2>

            <p class="text-gray-600">

              Kelola data transaksi, lihat, tambah, edit, atau hapus transaksi.

            </p>

          </a>

          <a

            href="{% url 'datamaster' %}"

            class="block p-4 bg-gray-100 hover:bg-gray-200 rounded-md transition duration-300"

          >

            <h2 class="text-lg font-semibold mb-2">Manajemen Data Master</h2>

            <p class="text-gray-600">

              Kelola data master, lihat, tambah, edit, atau hapus data.

            </p>

          </a>

        </div>

      </div>

    </div>

    <div class="container mx-auto pb-8 px-4">

      <div class="bg-gray-200 shadow-md rounded-lg p-6 mt-6">

        <h2 class="text-xl font-semibold mb-4">Statistik Layanan</h2>

        <select

          id="monthSelect"

          class="mb-4 p-2 bg-gray-500 text-white rounded"

        >

          <option value="">Pilih Bulan</option>

          <option value="1">Januari</option>

          <option value="2">Februari</option>

          <option value="3">Maret</option>

          <option value="4">April</option>

          <option value="5">Mei</option>

          <option value="6">Juni</option>

          <option value="7">Juli</option>

          <option value="8">Agustus</option>

          <option value="9">September</option>

          <option value="10">Oktober</option>

          <option value="11">November</option>

          <option value="12">Desember</option>

        </select>

        <canvas id="myChart" width="400" height="100"></canvas>

      </div>

    </div>

    <script>

      var ctx = document.getElementById("myChart").getContext("2d");

      var chart = new Chart(ctx, {

        type: "line",

        data: {

          labels: [],

          datasets: [

            {

              label: "Jumlah Transaksi",

              data: [],

              borderColor: "rgba(54, 162, 235, 1)",

              borderWidth: 2,

              fill: false,

            },

          ],

        },

        options: {

          scales: {

            y: {

              beginAtZero: true,

              ticks: {

                stepSize: 5,

              },

            },

          },

        },

      });

      document

        .getElementById("monthSelect")

        .addEventListener("change", function () {

          var month = this.value;

          if (month) {

            fetch(`/transaksi/data/${month}/`)

              .then((response) => response.json())

              .then((data) => {

                const daysInMonth = data.length;

                chart.data.labels = Array.from(

                  { length: daysInMonth },

                  (\_, i) => i + 1

                );

                chart.data.datasets[0].data = data;

                chart.data.datasets[0].label = `Jumlah Transaksi di ${

                  this.options[this.selectedIndex].text

                }`;

                chart.update();

              });

          }

        });

      // Initial update for the current month

      var currentMonth = new Date().getMonth() + 1;

      document.getElementById("monthSelect").value = currentMonth;

      fetch(`/transaksi/data/${currentMonth}/`)

        .then((response) => response.json())

        .then((data) => {

          const daysInMonth = data.length;

          chart.data.labels = Array.from(

            { length: daysInMonth },

            (\_, i) => i + 1

          );

          chart.data.datasets[0].data = data;

          chart.data.datasets[0].label = `Jumlah Transaksi di ${

            document.getElementById("monthSelect").options[currentMonth].text

          }`;

          chart.update();

        });

    </script>

  </body>

</html>

Kode HTML di atas adalah halaman admin panel untuk aplikasi "CuciKUY!". Halaman ini menyediakan pengelolaan transaksi dan data master, serta menampilkan statistik layanan.Pada bagian atas halaman, terdapat pesan selamat datang kepada admin yang sedang login, disertai dengan instruksi untuk mengelola pelanggan dan layanan dengan efisien. Di bagian tengah halaman, terdapat dua kotak yang mengarah ke halaman manajemen transaksi dan manajemen data master. Masing-masing kotak memiliki judul yang menggambarkan fungsinya, serta deskripsi singkat tentang apa yang dapat dilakukan di halaman tersebut. Selanjutnya, terdapat bagian untuk statistik layanan. Pengguna dapat memilih bulan tertentu dari dropdown "Pilih Bulan", dan grafik akan menampilkan jumlah transaksi selama bulan tersebut. Grafik tersebut diperbarui secara dinamis menggunakan data yang diperoleh melalui permintaan AJAX ke server saat pengguna memilih bulan dari dropdown. Ini memungkinkan admin untuk melihat perkembangan jumlah transaksi dari bulan ke bulan. Script JavaScript digunakan untuk menginisialisasi dan memperbarui grafik menggunakan pustaka Chart.js. Saat halaman dimuat, grafik akan diperbarui secara otomatis untuk menampilkan statistik transaksi dari bulan saat ini. Ketika pengguna memilih bulan dari dropdown, grafik akan diperbarui dengan data transaksi yang sesuai untuk bulan yang dipilih.

1. Login.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Login - Admin Panel</title>

    <link

      href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css"

      rel="stylesheet"

    />

  </head>

  <body class="bg-gray-300">

    <div class="min-h-screen flex items-center justify-center">

      <div class="max-w-md w-full bg-white p-8 shadow-md rounded-md">

        <h2 class="text-3xl font-bold mb-6 text-center text-blue-600">

          Admin Login

        </h2>

        <form action="{% url 'login' %}" method="POST">

          {% csrf\_token %}

          <div class="mb-4">

            <label

              for="username"

              class="block text-sm font-semibold mb-2 text-gray-700"

              >Username</label

            >

            <input

              type="text"

              id="username"

              name="username"

              class="w-full px-4 py-2 rounded-md border border-gray-300 focus:outline-none focus:ring-2 focus:ring-blue-500"

              required

            />

          </div>

          <!-- Password input -->

          <div class="mb-6">

            <label

              for="password"

              class="block text-sm font-semibold mb-2 text-gray-700"

              >Password</label

            >

            <input

              type="password"

              id="password"

              name="password"

              class="w-full px-4 py-2 rounded-md border border-gray-300 focus:outline-none focus:ring-2 focus:ring-blue-500"

              required

            />

          </div>

          {% if error\_message %}

          <div class="mb-4 text-red-500 text-sm text-center">

            {{ error\_message }}

          </div>

          {% endif %}

          <!-- Submit button -->

          <div class="flex justify-center">

            <button

              type="submit"

              class="bg-blue-500 text-white px-4 py-2 rounded-md hover:bg-blue-600 focus:outline-none focus:bg-blue-600"

            >

              Login

            </button>

          </div>

        </form>

      </div>

    </div>

  </body>

</html>

Halaman HTML di atas adalah halaman login untuk admin panel aplikasi. Halaman ini memiliki desain sederhana namun responsif menggunakan Tailwind CSS. Di bagian atas, terdapat judul "Admin Login" yang ditampilkan dengan ukuran dan gaya teks yang menarik perhatian. Kemudian, terdapat formulir login dengan dua input: satu untuk username dan yang lainnya untuk password. Input tersebut memiliki atribut yang menentukan tipe data, atribut nama, dan atribut yang menunjukkan kewajiban pengisian. Jika terdapat pesan kesalahan (error\_message), pesan tersebut akan ditampilkan di bawah formulir dalam warna merah. Terakhir, terdapat tombol "Login" yang ketika ditekan akan mengirimkan formulir ke URL yang ditentukan oleh aksi formulir (action="{% url 'login' %}"). Tombol ini memiliki gaya yang menarik, berwarna biru dan berubah warna saat dihove.

1. nav.html

<nav class="bg-green-500 p-4">

  <div

    class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8 flex justify-between items-center"

  >

    <div class="flex-shrink-0">

      <a href="{% url 'index' %}" class="text-white font-bold text-xl"

        >CuciKUY!</a

      >

      {% if request.user.is\_authenticated %}

      <span class="text-white-300 bg-white ml-4 px-2 py-1 rounded-full"

        >{{ request.user.nama }}</span

      >

      {% else %}

      <span class="text-white ml-4">Guest</span>

      {% endif %}

    </div>

    <div class="flex items-center">

      <div class="hidden md:block">

        <ul class="ml-4 flex items-center space-x-6">

          <li>

            <a

              href="{% url 'index' %}"

              class="text-white hover:text-gray-200 transition duration-300"

              >Beranda</a

            >

          </li>

          <li>

            <a

              href="{% url 'logout' %}"

              class="text-white hover:text-gray-200 ml-2 transition duration-300"

              >Logout</a

            >

          </li>

        </ul>

      </div>

    </div>

  </div>

</nav>

Navigasi di atas adalah bagian dari antarmuka pengguna untuk aplikasi web "CuciKUY!". Ini merupakan navbar dengan latar belakang hijau yang menonjol, memberikan kesan segar dan bersahabat. Navbar tersebut terbagi menjadi dua bagian utama. Bagian kiri menyajikan logo atau judul aplikasi "CuciKUY!" yang ditampilkan dengan teks putih tebal. Jika pengguna sudah login (request.user.is\_authenticated), nama pengguna akan ditampilkan di sebelah logo sebagai indikasi pengguna yang sedang aktif. Jika tidak, hanya teks "Guest" yang akan ditampilkan. Bagian kanan navbar berisi tautan navigasi, terdiri dari dua tautan saat ini: "Beranda" dan "Logout". Ketika dihover, tautan-tautan tersebut akan berubah warna menjadi abu-abu, memberikan umpan balik visual kepada pengguna. Tautan "Beranda" membawa pengguna ke halaman beranda ({% url 'index' %}), sementara tautan "Logout" digunakan untuk keluar dari sesi pengguna saat ini.

1. transaksi.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Manajemen Transaksi - CuciKUY!</title>

    <link

      href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css"

      rel="stylesheet"

    />

  </head>

  <body class="bg-green-300">

    <!-- Navbar -->

    {% include 'nav.html' %}

    <!-- Main Content -->

    <div class="container mx-auto p-4">

      <h1 class="text-xl font-bold text-green-800 mb-4">Manajemen Transaksi</h1>

      <!-- Form untuk menambah atau mengedit transaksi -->

      <div class="mb-8">

        <form

          action="{% url 'transaksi' %}"

          method="post"

          class="bg-white p-4 shadow-md rounded-lg"

        >

          {% csrf\_token %}

          <div class="mb-4">

            <label for="transaction\_date" class="block text-sm font-bold mb-2"

              >Tanggal Transaksi:</label

            >

            {{ form.transaction\_date }}

          </div>

          <div class="mb-4">

            <label for="pelanggan" class="block text-sm font-bold mb-2"

              >Nama Pelanggan:</label

            >

            {{ form.pelanggan }}

          </div>

          <div class="mb-4">

            <label for="jeniskendaraan" class="block text-sm font-bold mb-2"

              >Jenis Kendaraan:</label

            >

            {{ form.jeniskendaraan }}

          </div>

          <button

            type="submit"

            class="bg-green-500 hover:bg-green-600 text-white px-4 py-2 rounded mt-2"

          >

            Simpan Transaksi

          </button>

        </form>

      </div>

      <!-- Load custom tags -->

      {% load custom\_tags %}

      <!-- Filter Transaksi -->

      <div class="mb-8">

        <form

          id="filterForm"

          method="get"

          action="{% url 'transaksi' %}"

          class="flex items-center space-x-4 bg-white p-4 shadow-md rounded-lg"

        >

          <div>

            <label for="bulan" class="block text-sm font-bold mb-2"

              >Bulan:</label

            >

            <select

              id="bulan"

              name="bulan"

              class="shadow appearance-none border rounded w-full py-2 px-3 text-gray-700 leading-tight focus:outline-none focus:shadow-outline"

              required

            >

              <option value="">Pilih Bulan</option>

              {% for i in 12|range\_filter %}

              <option value="{{ i }}">{{ i }}</option>

              {% endfor %}

            </select>

          </div>

          <div>

            <label for="hari" class="block text-sm font-bold mb-2">Hari:</label>

            <select

              id="hari"

              name="hari"

              class="shadow appearance-none border rounded w-full py-2 px-3 text-gray-700 leading-tight focus:outline-none focus:shadow-outline"

              required

            >

              <option value="">Pilih Hari</option>

              {% for i in 31|range\_filter %}

              <option value="{{ i }}">{{ i }}</option>

              {% endfor %}

            </select>

          </div>

          <div>

            <button

              type="submit"

              class="bg-green-500 hover:bg-green-600 text-white px-4 py-2 rounded mt-6"

            >

              Tampilkan

            </button>

          </div>

        </form>

      </div>

      <!-- Tabel Transaksi -->

      <div class="bg-white p-4 shadow-md rounded-lg">

        <table class="w-full table-auto">

          <thead>

            <tr class="bg-gray-200">

              <th class="border px-4 py-2">No</th>

              <th class="border px-4 py-2">Tanggal</th>

              <th class="border px-4 py-2">Nama Pelanggan</th>

              <th class="border px-4 py-2">Jenis Kendaraan</th>

              <th class="border px-4 py-2">Harga</th>

              <th class="border px-4 py-2">Aksi</th>

            </tr>

          </thead>

          <tbody>

            {% for transaction in transactions %}

            <tr class="bg-white hover:bg-gray-100 transition duration-200">

              <td class="border px-4 py-2">{{ forloop.counter }}</td>

              <td class="border px-4 py-2">

                {{ transaction.transaction\_date }}

              </td>

              <td class="border px-4 py-2">{{ transaction.pelanggan }}</td>

              <td class="border px-4 py-2">{{ transaction.jeniskendaraan }}</td>

              <!-- prettier-ignore -->

              <td class="border px-4 py-2">

                        {% if transaction.price %}

                            {{ transaction.price.harga }}

                        {% else %}

                            Harga tidak tersedia

                        {% endif %}

                    </td>

              <!-- prettier-ignore -->

              <td class="border px-2 py-2 rounded-lg">

                <div class="flex space-x-3 w-full">

                    <button data-id="{{ transaction.idtransaction }}" onclick="openEditModal(this)" class="bg-blue-600 hover:bg-blue-800 text-white px-2 py-1 rounded-md transition duration-300 ease-in-out transform hover:scale-105 shadow-lg flex items-center flex-grow">

                        <i class="fas fa-edit mr-1"></i>Edit

                    </button>

                    <button data-id="{{ transaction.idtransaction }}" onclick="openDeleteModal(this)" class="bg-red-600 hover:bg-red-800 text-white px-2 py-1 rounded-md transition duration-300 ease-in-out transform hover:scale-105 shadow-lg flex items-center flex-grow">

                        <i class="fas fa-trash-alt mr-1"></i>Hapus

                    </button>

                    <a href="{% url 'cetak\_nota' transaction.idtransaction %}" class="bg-green-600 hover:bg-green-800 text-white px-2 py-1 rounded-md transition duration-300 ease-in-out transform hover:scale-105 shadow-lg flex items-center flex-grow">

                      <i class="fas fa-print mr-1"></i>Cetak Nota

                  </a>

                </div>

            </td>

            </tr>

            {% endfor %}

          </tbody>

        </table>

      </div>

    </div>

    <!-- Edit Modal -->

    <div

      id="editModal"

      class="hidden fixed inset-0 bg-gray-600 bg-opacity-50 overflow-y-auto h-full w-full"

    >

      <div

        class="relative top-20 mx-auto p-5 border w-96 shadow-lg rounded-md bg-white"

      >

        <div class="mt-3 text-center">

          <h3 class="text-lg leading-6 font-medium text-gray-900">

            Edit Transaksi

          </h3>

          <div class="mt-2 px-7 py-3">

            <form id="editForm" method="post">

              {% csrf\_token %}

              <div class="mb-4">

                <label

                  for="edit\_transaction\_date"

                  class="block text-sm font-bold mb-2"

                  >Tanggal Transaksi:</label

                >

                <input

                  type="date"

                  id="edit\_transaction\_date"

                  name="transaction\_date"

                  required

                  class="shadow appearance-none border rounded w-full py-2 px-3 text-gray-700 leading-tight focus:outline-none focus:shadow-outline"

                />

              </div>

              <div class="mb-4">

                <label for="edit\_pelanggan" class="block text-sm font-bold mb-2"

                  >Nama Pelanggan:</label

                >

                <input

                  type="text"

                  id="edit\_pelanggan"

                  name="pelanggan"

                  required

                  class="shadow appearance-none border rounded w-full py-2 px-3 text-gray-700 leading-tight focus:outline-none focus:shadow-outline"

                />

              </div>

              <div class="mb-4">

                <label

                  for="edit\_jeniskendaraan"

                  class="block text-sm font-bold mb-2"

                  >Jenis Kendaraan:</label

                >

                <select

                  id="edit\_jeniskendaraan"

                  name="jeniskendaraan"

                  class="shadow appearance-none border rounded w-full py-2 px-3 text-gray-700 leading-tight focus:outline-none focus:shadow-outline"

                  required

                >

                  <option value="motor">Motor</option>

                  <option value="mobil">Mobil</option>

                </select>

              </div>

              <button

                type="submit"

                class="bg-blue-500 hover:bg-blue-600 text-white px-4 py-2 rounded mt-2"

              >

                Simpan Perubahan

              </button>

              <button

                type="button"

                onclick="closeEditModal()"

                class="bg-gray-500 hover:bg-gray-600 text-white px-4 py-2 rounded mt-2"

              >

                Batal

              </button>

            </form>

          </div>

        </div>

      </div>

    </div>

    <!-- Delete Modal -->

    <div

      id="deleteModal"

      class="hidden fixed inset-0 bg-gray-600 bg-opacity-50 overflow-y-auto h-full w-full"

    >

      <div

        class="relative top-20 mx-auto p-5 border w-96 shadow-lg rounded-md bg-white"

      >

        <div class="mt-3 text-center">

          <h3 class="text-lg leading-6 font-medium text-gray-900">

            Konfirmasi Hapus

          </h3>

          <div class="mt-2 px-7 py-3">

            <p class="text-sm text-gray-500">

              Apakah Anda yakin ingin menghapus transaksi ini?

            </p>

          </div>

          <div class="items-center px-4 py-3">

            <form id="deleteForm" method="post">

              {% csrf\_token %}

              <button

                type="submit"

                class="bg-red-500 hover:bg-red-600 text-white px-4 py-2 rounded"

              >

                Hapus

              </button>

            </form>

            <button

              type="button"

              onclick="closeDeleteModal()"

              class="bg-gray-500 hover:bg-gray-600 text-white px-4 py-2 rounded"

            >

              Batal

            </button>

          </div>

        </div>

      </div>

    </div>

    <script

      src="https://kit.fontawesome.com/3e1ec18e22.js"

      crossorigin="anonymous"

    ></script>

    <script>

      function openEditModal(button) {

        var transactionId = button.getAttribute("data-id");

        var row = button.closest("tr");

        var transactionDate = row.querySelector("td:nth-child(2)").innerText;

        var pelanggan = row.querySelector("td:nth-child(3)").innerText;

        var jenisKendaraan = row

          .querySelector("td:nth-child(4)")

          .innerText.toLowerCase();

        document.getElementById("edit\_transaction\_date").value =

          transactionDate;

        document.getElementById("edit\_pelanggan").value = pelanggan;

        document.getElementById("edit\_jeniskendaraan").value = jenisKendaraan;

        var actionUrl = "{% url 'edit\_transaction' 0 %}".replace(

          "0",

          transactionId

        );

        document.getElementById("editForm").action = actionUrl;

        document.getElementById("editModal").style.display = "block";

      }

      function closeEditModal() {

        document.getElementById("editModal").style.display = "none";

      }

      function openDeleteModal(button) {

        var transactionId = button.getAttribute("data-id");

        var actionUrl = "{% url 'delete\_transaction' 0 %}".replace(

          "0",

          transactionId

        );

        document.getElementById("deleteForm").action = actionUrl;

        document.getElementById("deleteModal").style.display = "block";

      }

      function closeDeleteModal() {

        document.getElementById("deleteModal").style.display = "none";

      }

    </script>

  </body>

</html>

Halaman HTML di atas adalah bagian dari antarmuka pengguna untuk fitur "Manajemen Transaksi" dalam aplikasi web "CuciKUY!". Halaman ini menampilkan formulir untuk menambah atau mengedit transaksi, filter untuk menampilkan transaksi berdasarkan bulan dan hari, serta tabel yang menampilkan daftar transaksi beserta aksi yang dapat dilakukan seperti edit, hapus, dan cetak nota. Formulir untuk menambah atau mengedit transaksi memungkinkan pengguna untuk memasukkan atau memperbarui tanggal transaksi, nama pelanggan, dan jenis kendaraan. Tombol "Simpan Transaksi" digunakan untuk menyimpan transaksi yang dimasukkan atau diperbarui. Filter transaksi memungkinkan pengguna untuk memfilter transaksi berdasarkan bulan dan hari tertentu. Setelah memilih bulan dan hari yang diinginkan, pengguna dapat menekan tombol "Tampilkan" untuk menampilkan transaksi yang sesuai dengan filter yang telah dipilih. Tabel transaksi menampilkan daftar transaksi yang telah disimpan, termasuk tanggal transaksi, nama pelanggan, jenis kendaraan, harga (jika tersedia), dan aksi yang dapat dilakukan. Aksi yang dapat dilakukan meliputi edit transaksi, hapus transaksi, dan cetak nota untuk transaksi tertentu. Selain itu, halaman ini juga mencakup modal untuk mengedit transaksi dan konfirmasi hapus transaksi. Modal edit transaksi memungkinkan pengguna untuk memperbarui detail transaksi yang ada, sementara modal konfirmasi hapus digunakan untuk mengonfirmasi penghapusan transaksi sebelum tindakan tersebut dilakukan secara permanen. Script JavaScript yang terdapat pada halaman ini digunakan untuk mengatur perilaku modal edit dan hapus, termasuk pengisian data transaksi yang sudah ada ke dalam formulir edit, serta menutup modal ketika operasi selesai dilakukan.

1. Templatetags
2. custom\_tags.py

from django import template

register = template.Library()

@register.filter

def range\_filter(value):

    return range(1, value + 1)

Kode diatas adalah sebuah custom Django template filter yang didefinisikan di dalam sebuah aplikasi Django. Filter ini bernama range\_filter dan dibuat untuk menghasilkan rentang bilangan bulat dari 1 hingga nilai yang diberikan. Penggunaan filter ini dapat bermanfaat saat membangun tampilan HTML yang memerlukan perulangan berbasis rentang bilangan bulat, misalnya dalam pembuatan elemen <option> dalam elemen <select> di halaman web. Dengan filter ini, Anda dapat menghasilkan rentang bilangan bulat sesuai dengan nilai yang diberikan dalam kode template Django. Setelah mendaftarkan filter ini dengan menggunakan decorator @register.filter, filter tersebut dapat diakses dan digunakan di dalam template Django yang sesuai dalam proyek aplikasi.

1. apps.py

from django.apps import AppConfig

class CucikendaraanConfig(AppConfig):

    default\_auto\_field = 'django.db.models.BigAutoField'

    name = 'cucikendaraan'

kode diatas adalah sebuah konfigurasi aplikasi Django yang disebut CucikendaraanConfig. Ini adalah kelas yang mengonfigurasi aplikasi Django bernama 'cucikendaraan'. Di dalam kelas ini, terdapat dua atribut yang diatur: default\_auto\_field: Atribut ini menentukan tipe bidang otomatis yang akan digunakan untuk model dalam aplikasi ini. Di sini, digunakan 'django.db.models.BigAutoField', yang merupakan bidang otomatis besar untuk kunci utama model, yang secara otomatis menghasilkan nilai unik bertipe Big Integer saat objek model baru dibuat.

name: Atribut ini adalah nama aplikasi Django yang digunakan untuk merujuk ke aplikasi ini di dalam kode proyek. Dalam hal ini, nama aplikasinya adalah 'cucikendaraan'.

Konfigurasi aplikasi ini berperan penting dalam mengonfigurasi perilaku default dan pengaturan spesifik aplikasi Django. Dengan menentukan default\_auto\_field, kita dapat mengontrol jenis bidang kunci utama yang digunakan dalam model aplikasi. Sedangkan dengan menentukan name, kita memberikan nama untuk aplikasi yang akan digunakan di seluruh proyek.

1. forms.py

from django import forms

from .models import Transaction,UserCuciKuy, Price

from django.contrib.auth.forms import UserCreationForm

class TransactionForm(forms.ModelForm):

    class Meta:

        model = Transaction

        fields = ['transaction\_date', 'pelanggan', 'jeniskendaraan']

        widgets = {

            'transaction\_date': forms.DateInput(attrs={'class': 'p-2 border rounded w-full', 'type': 'date'}),

            'pelanggan': forms.TextInput(attrs={'class': 'p-2 border rounded w-full'}),

            'jeniskendaraan': forms.Select(attrs={'class': 'p-2 border rounded w-full'}),

        }

class UserCuciKuyCreationForm(forms.ModelForm):

    class Meta:

        model = UserCuciKuy

        fields = ['username', 'nama', 'password']

        widgets = {

            'username': forms.TextInput(attrs={'class': 'p-2 border rounded w-full'}),

            'nama': forms.TextInput(attrs={'class': 'p-2 border rounded w-full'}),

            'password': forms.TextInput(attrs={'class': 'p-2 border rounded w-full', "type": "password"}),

        }

class PriceForm(forms.ModelForm):

    class Meta:

        model = Price

        fields = ['jeniskendaraan', 'harga']

Berikut adalah penjelasan singkat tentang setiap kelas formulir:

TransactionForm: Ini adalah formulir untuk model Transaction. Formulir ini memiliki bidang-bidang transaction\_date, pelanggan, dan jeniskendaraan. Setiap bidang memiliki widget khusus yang menentukan jenis input HTML yang digunakan dalam formulir.

UserCuciKuyCreationForm: Ini adalah formulir untuk membuat pengguna baru dalam aplikasi CuciKUY!. Formulir ini terkait dengan model UserCuciKuy, yang mungkin merupakan model pengguna yang dikustomisasi. Formulir ini memiliki bidang username, nama, dan password. Setiap bidang juga memiliki widget yang sesuai untuk menentukan jenis input yang tepat.

PriceForm: Ini adalah formulir untuk model Price. Formulir ini memiliki bidang jeniskendaraan dan harga. Widget untuk bidang-bidang ini tidak didefinisikan secara eksplisit, jadi formulir akan menggunakan widget default yang sesuai dengan tipe bidang model. Dengan menggunakan formulir seperti ini, Anda dapat membuat tampilan yang dinamis untuk interaksi pengguna dengan data dalam aplikasi Django Anda. Formulir memudahkan validasi data yang dikirimkan oleh pengguna dan memudahkan penanganan permintaan HTTP untuk membuat, mengedit, atau menghapus objek dari model Anda.

1. models.py

from django.db import models

from django.contrib.auth.hashers import make\_password, check\_password

from django.utils.timezone import now

from django.contrib.auth.models import AbstractBaseUser, BaseUserManager, PermissionsMixin

class CustomUserManager(BaseUserManager):

    def create\_user(self, username, password=None, \*\*extra\_fields):

        if not username:

            raise ValueError('The Username must be set')

        user = self.model(username=username, \*\*extra\_fields)

        user.set\_password(password)

        user.save(using=self.\_db)

        return user

    def create\_superuser(self, username, password, \*\*extra\_fields):

        extra\_fields.setdefault('is\_staff', True)

        extra\_fields.setdefault('is\_superuser', True)

        return self.create\_user(username, password, \*\*extra\_fields)

class UserCuciKuy(AbstractBaseUser, PermissionsMixin):

    username = models.CharField(max\_length=150, unique=True)

    nama = models.CharField(max\_length=100)

    password = models.CharField(max\_length=255)

    is\_staff = models.BooleanField(default=False)

    objects = CustomUserManager()

    USERNAME\_FIELD = 'username'

    REQUIRED\_FIELDS = ['nama']  # Assuming 'nama' is a required field for superuser creation.

    def \_\_str\_\_(self):

        return self.username

    def save(self, \*args, \*\*kwargs):

        self.password = make\_password(self.password)

        super(UserCuciKuy, self).save(\*args, \*\*kwargs)

# class Price(models.Model):

#     jeniskendaraan = models.CharField(max\_length=50, choices=[('motor', 'Motor'), ('mobil', 'Mobil')])

#     harga = models.DecimalField(max\_digits=10, decimal\_places=2)

#     def \_\_str\_\_(self):

#         return f'{self.jeniskendaraan} - {self.harga}'

class Price(models.Model):

    jeniskendaraan = models.CharField(max\_length=50, choices=[('motor', 'Motor'), ('mobil', 'Mobil')], unique=True)

    harga = models.DecimalField(max\_digits=10, decimal\_places=2)

    def \_\_str\_\_(self):

        return f'{self.jeniskendaraan} - {self.harga}'

class Transaction(models.Model):

    transaction\_date = models.DateField()

    pelanggan = models.CharField(max\_length=100)

    jeniskendaraan = models.CharField(max\_length=50, choices=[('motor', 'Motor'), ('mobil', 'Mobil')])

    idtransaction = models.AutoField(primary\_key=True)

    price = models.ForeignKey(Price, on\_delete=models.CASCADE, null=True, blank=True)

    def \_\_str\_\_(self):

        return f'{self.idtransaction} - {self.transaction\_date} - {self.pelanggan} - {self.jeniskendaraan} - {self.price.harga if self.price else "No Price"}'

Berikut adalah penjelasan singkat tentang setiap model:

CustomUserManager: Ini adalah manajer kustom untuk model pengguna UserCuciKuy. Ini memungkinkan pembuatan pengguna biasa dan superuser.

UserCuciKuy: Ini adalah model pengguna kustom yang menggantikan model pengguna bawaan Django. Ini memiliki bidang username, nama, password, dan is\_staff. Metode save telah diperluas untuk membuat hash dari kata sandi pengguna sebelum menyimpannya.

Price: Ini adalah model yang mewakili harga untuk jenis kendaraan tertentu (motor atau mobil). Ini memiliki bidang jeniskendaraan dan harga.

Transaction: Ini adalah model yang merepresentasikan transaksi dalam aplikasi Anda. Ini memiliki bidang transaction\_date, pelanggan, jeniskendaraan, dan price. Bidang price adalah kunci asing yang merujuk ke model Price, yang menunjukkan harga kendaraan yang terkait dengan transaksi tersebut.

Dengan menggunakan model-model ini, Anda dapat membuat dan mengelola data dalam aplikasi Django Anda dengan cara yang terstruktur dan aman.

1. tmp.py

import os

import django

os.environ.setdefault("DJANGO\_SETTINGS\_MODULE", "uas\_cucikendaraan.settings")

django.setup()

from cucikendaraan.models import Price

def tambah\_harga():

    pilihan\_kendaraan = input("Jenis Kendaraan (motor/mobil): ").strip().lower()

    if pilihan\_kendaraan not in ['motor', 'mobil']:

        print("Jenis kendaraan tidak valid. Pilih 'motor' atau 'mobil'.")

        return

    # Harga otomatis berdasarkan jenis kendaraan

    if pilihan\_kendaraan == 'motor':

        harga = 9000.00

    elif pilihan\_kendaraan == 'mobil':

        harga = 20000.00

    new\_price = Price(jeniskendaraan=pilihan\_kendaraan, harga=harga)

    new\_price.save()

    print(f"Harga untuk kendaraan '{pilihan\_kendaraan}' berhasil ditambahkan dengan harga {harga}.")

if \_\_name\_\_ == "\_\_main\_\_":

    tambah\_harga()

Kode ini skrip Python yang berfungsi untuk menambahkan harga baru ke dalam basis data menggunakan model Price dari aplikasi Django Anda. Berikut adalah langkah-langkah yang dilakukan oleh skrip tersebut:

1. Pertama, modul os dan django diimpor untuk menyiapkan lingkungan Django.

2. Variabel lingkungan DJANGO\_SETTINGS\_MODULE diatur ke "uas\_cucikendaraan.settings" agar pengaturan Django dapat diterapkan.

3. Selanjutnya, model Price dari aplikasi Django cucikendaraan diimpor. Model ini mewakili entitas harga dalam basis data.

4. Fungsi tambah\_harga() didefinisikan. Fungsi ini bertanggung jawab untuk meminta input dari pengguna tentang jenis kendaraan (motor atau mobil).

5. Input dari pengguna kemudian dibersihkan dari spasi ekstra dan diubah menjadi huruf kecil agar memudahkan pemrosesan.

6. Dilakukan pemeriksaan untuk memastikan bahwa jenis kendaraan yang dimasukkan oleh pengguna adalah 'motor' atau 'mobil'. Jika bukan, program memberikan pesan kesalahan dan berhenti.

7. Harga otomatis ditentukan berdasarkan jenis kendaraan yang dimasukkan. Untuk motor, harga diatur ke 9000.00, sedangkan untuk mobil diatur ke 20000.00.

8. Setelah harga ditentukan, sebuah objek Price baru dibuat dengan menggunakan nilai jenis kendaraan dan harga yang telah ditentukan sebelumnya.

9. Objek Price yang baru dibuat kemudian disimpan ke dalam basis data menggunakan metode save().

10. Akhirnya, program memberikan konfirmasi kepada pengguna bahwa harga untuk jenis kendaraan tertentu telah berhasil ditambahkan ke dalam basis data.

Skrip ini dirancang untuk dijalankan sebagai program mandiri (bukan diimpor sebagai modul) dan akan menunggu input dari pengguna sebelum menambahkan harga ke dalam basis data.

1. Urls.py

from django.urls import path

from .views import index, login\_view, logout\_view, transaksi, edit\_transaction, delete\_transaction, cetak\_nota, get\_transaction\_data, datamaster

urlpatterns = [

    path('', index, name='index'),

    path('login/', login\_view, name='login'),

    path('logout/', logout\_view, name='logout'),

    path('transaksi/', transaksi, name='transaksi'),

    path('transaksi/edit/<int:idtransaction>/', edit\_transaction, name='edit\_transaction'),

    path('transaksi/delete/<int:idtransaction>/', delete\_transaction, name='delete\_transaction'),

    path('transaksi/cetak/<int:idtransaction>/', cetak\_nota, name='cetak\_nota'),

    path('transaksi/data/<int:month>/', get\_transaction\_data, name='get\_transaction\_data'),

    path('datamaster/', datamaster, name='datamaster'),

]

Setiap URL dihubungkan dengan fungsi view yang sesuai untuk menangani permintaan HTTP yang cocok. Berikut adalah penjelasan untuk setiap URL yang didefinisikan dalam urlpatterns:

1. '': Ini adalah URL utama atau beranda dari aplikasi, yang akan menunjukkan ke view index.

2. 'login/': URL ini mengarah ke view login\_view untuk menangani proses login.

3. 'logout/': URL ini mengarah ke view logout\_view untuk menangani proses logout.

4. 'transaksi/': URL ini digunakan untuk menangani berbagai operasi transaksi seperti menampilkan daftar transaksi, menambah transaksi baru, dan lainnya. View yang digunakan adalah transaksi.

5. 'transaksi/edit/<int:idtransaction>/': URL ini digunakan untuk mengedit transaksi dengan ID tertentu. View edit\_transaction akan menangani proses pengeditan.

6. 'transaksi/delete/<int:idtransaction>/': URL ini digunakan untuk menghapus transaksi dengan ID tertentu. View delete\_transaction akan menangani proses penghapusan.

7. 'transaksi/cetak/<int:idtransaction>/': URL ini digunakan untuk mencetak nota dari transaksi dengan ID tertentu. View cetak\_nota akan menangani proses pencetakan.

8. 'transaksi/data/<int:month>/': URL ini digunakan untuk mendapatkan data transaksi untuk bulan tertentu. View get\_transaction\_data akan menangani permintaan untuk data tersebut.

9. 'datamaster/': URL ini digunakan untuk menangani operasi yang terkait dengan manajemen data master seperti menampilkan, menambah, mengedit, atau menghapus data master. View yang digunakan adalah datamaster.

Dengan definisi URL ini, aplikasi Anda akan dapat merespons dengan benar terhadap berbagai jenis permintaan HTTP yang diterimanya.

1. Views.py

from django.shortcuts import render, redirect, get\_object\_or\_404

from django.contrib.auth import authenticate, login as auth\_login, logout as auth\_logout

from django.contrib.auth.decorators import login\_required

from .models import UserCuciKuy, Transaction, Price

from .forms import TransactionForm, UserCuciKuyCreationForm, PriceForm

from django.http import HttpResponse, JsonResponse

from reportlab.lib.pagesizes import letter

from reportlab.pdfgen import canvas

import io

import calendar

@login\_required

def index(request):

    return render(request, 'index.html')

@login\_required

def logout\_view(request):

    auth\_logout(request)

    return redirect('login')

def login\_view(request):

    if request.method == 'POST':

        username = request.POST.get('username')

        password = request.POST.get('password')

        user = authenticate(request, username=username, password=password)

        if user is not None:

            auth\_login(request, user)

            return redirect('index')

        else:

            return render(request, 'login.html', {'error\_message': 'Invalid credentials'})

    else:

        return render(request, 'login.html')

@login\_required

def transaksi(request):

    transactions = Transaction.objects.all()

    bulan = request.GET.get('bulan')

    hari = request.GET.get('hari')

    if bulan and hari:

        transactions = transactions.filter(transaction\_date\_\_month=bulan, transaction\_date\_\_day=hari)

    if request.method == 'POST':

        form = TransactionForm(request.POST)

        if form.is\_valid():

            transaction = form.save(commit=False)

            jeniskendaraan = form.cleaned\_data['jeniskendaraan']

            prices = Price.objects.filter(jeniskendaraan=jeniskendaraan)

            if prices.exists():

                transaction.price = prices.first()  # Choose the first Price object found

            else:

                transaction.price = None  # or handle the case where price is not found

            transaction.save()

            return redirect('transaksi')

    else:

        form = TransactionForm()

    return render(request, 'transaksi.html', {'form': form, 'transactions': transactions})

@login\_required

def edit\_transaction(request, idtransaction):

    transaction = get\_object\_or\_404(Transaction, idtransaction=idtransaction)

    if request.method == 'POST':

        form = TransactionForm(request.POST, instance=transaction)

        if form.is\_valid():

            form.save()

            return redirect('transaksi')

        else:

            return redirect('transaksi')

    return redirect('transaksi')

@login\_required

def delete\_transaction(request, idtransaction):

    transaction = get\_object\_or\_404(Transaction, idtransaction=idtransaction)

    if request.method == 'POST':

        transaction.delete()

        return redirect('transaksi')

    return redirect('transaksi')

@login\_required

def cetak\_nota(request, idtransaction):

    transaction = get\_object\_or\_404(Transaction, idtransaction=idtransaction)

    buffer = io.BytesIO()

    p = canvas.Canvas(buffer, pagesize=letter)

    p.setTitle("Nota Transaksi")

    p.setFont("Helvetica-Bold", 16)

    p.drawString(100, 750, "CuciKUY!")

    p.setFont("Helvetica", 12)

    p.drawString(100, 730, "Alamat: Jl. Testing")

    # Transaction details

    p.setFont("Helvetica", 12)

    p.drawString(100, 700, f"Nota Transaksi #{transaction.idtransaction}")

    p.drawString(100, 680, f"Nama Pelanggan: {transaction.pelanggan}")

    p.drawString(100, 660, f"Jenis Kendaraan: {transaction.jeniskendaraan}")

    p.drawString(100, 640, f"Harga: {transaction.price.harga if transaction.price else 'Harga tidak tersedia'}")

    p.drawString(100, 620, f"Tanggal Transaksi: {transaction.transaction\_date.strftime('%B %d, %Y')}")

    # Close the PDF object cleanly, and we're done.

    p.showPage()

    p.save()

    buffer.seek(0)

    return HttpResponse(buffer, content\_type='application/pdf')

@login\_required

def get\_transaction\_data(request, month):

    month = int(month)

    transactions = Transaction.objects.filter(transaction\_date\_\_month=month)

    days\_in\_month = calendar.monthrange(2024, month)[1]

    data = [0] \* days\_in\_month

    for transaction in transactions:

        day = transaction.transaction\_date.day

        data[day - 1] += 1

    return JsonResponse(data, safe=False)

@login\_required

def datamaster(request):

    user\_form = UserCuciKuyCreationForm()

    price\_form = PriceForm()

    if request.method == 'POST':

        if 'username' in request.POST:

            user\_form = UserCuciKuyCreationForm(request.POST)

            if user\_form.is\_valid():

                user\_form.save()

                return redirect('datamaster')

        elif 'jeniskendaraan' in request.POST:

            price\_form = PriceForm(request.POST)

            if price\_form.is\_valid():

                price\_form.save()

                return redirect('datamaster')

        elif 'edit\_username' in request.POST:

            user\_id = request.POST.get('user\_id')

            user = get\_object\_or\_404(UserCuciKuy, id=user\_id)

            user.username = request.POST.get('edit\_username')

            user.nama = request.POST.get('edit\_nama')

            user.save()

            return redirect('datamaster')

        elif 'edit\_jeniskendaraan' in request.POST:

            price\_id = request.POST.get('price\_id')

            price = get\_object\_or\_404(Price, id=price\_id)

            price.jeniskendaraan = request.POST.get('edit\_jeniskendaraan')

            price.harga = request.POST.get('edit\_harga')

            price.save()

            return redirect('datamaster')

        elif 'delete\_user\_id' in request.POST:

            user\_id = request.POST.get('delete\_user\_id')

            user = get\_object\_or\_404(UserCuciKuy, id=user\_id)

            user.delete()

            return redirect('datamaster')

        elif 'delete\_price\_id' in request.POST:

            price\_id = request.POST.get('delete\_price\_id')

            price = get\_object\_or\_404(Price, id=price\_id)

            price.delete()

            return redirect('datamaster')

    users = UserCuciKuy.objects.all()

    prices = Price.objects.all()

    return render(request, 'datamaster.html', {'user\_form': user\_form, 'price\_form': price\_form, 'users': users, 'prices': prices})

views.py adalah tempat di mana logika bisnis aplikasi Django Anda didefinisikan. Setiap fungsi dalam file ini disebut view, yang menerima permintaan HTTP dari klien, memprosesnya, dan mengembalikan respons yang sesuai. Berikut adalah penjelasan untuk setiap fungsi view dalam file views.py:

1. index(request): Fungsi ini menangani permintaan untuk halaman beranda. Jika pengguna belum login, mereka akan diarahkan ke halaman login. Jika sudah login, halaman beranda akan ditampilkan.

2. logout\_view(request): Fungsi ini menangani permintaan untuk proses logout pengguna. Setelah logout berhasil, pengguna akan diarahkan kembali ke halaman login.

3. login\_view(request): Fungsi ini menangani permintaan untuk proses login pengguna. Jika metode permintaan adalah POST, fungsi akan mencoba untuk melakukan otentikasi pengguna. Jika otentikasi berhasil, pengguna akan diarahkan ke halaman beranda. Jika otentikasi gagal, pesan kesalahan akan ditampilkan di halaman login. Jika metode permintaan adalah GET, halaman login akan ditampilkan.

4. transaksi(request): Fungsi ini menangani permintaan untuk halaman transaksi, yang mencakup penambahan transaksi baru dan penampilan daftar transaksi yang ada. Jika metode permintaan adalah POST, transaksi baru akan ditambahkan. Jika metode permintaan adalah GET, halaman transaksi akan ditampilkan bersama dengan formulir untuk menambahkan transaksi baru.

5. edit\_transaction(request, idtransaction): Fungsi ini menangani permintaan untuk mengedit transaksi yang ada berdasarkan ID transaksi. Jika metode permintaan adalah POST, transaksi akan diperbarui dengan data yang baru. Setelah itu, pengguna akan diarahkan kembali ke halaman transaksi.

6. delete\_transaction(request, idtransaction): Fungsi ini menangani permintaan untuk menghapus transaksi yang ada berdasarkan ID transaksi. Jika metode permintaan adalah POST, transaksi akan dihapus dari database. Setelah itu, pengguna akan diarahkan kembali ke halaman transaksi.

7. cetak\_nota(request, idtransaction): Fungsi ini menangani permintaan untuk mencetak nota transaksi berdasarkan ID transaksi. Nota akan dibuat dalam format PDF dan dikirim sebagai respons.

8. get\_transaction\_data(request, month): Fungsi ini menangani permintaan untuk mendapatkan data transaksi berdasarkan bulan. Jumlah transaksi untuk setiap hari dalam bulan tersebut akan dihitung dan dikirim sebagai respons JSON.

9. datamaster(request): Fungsi ini menangani permintaan untuk halaman data master, yang mencakup manajemen pengguna dan harga. Pengguna dapat menambah, mengedit, atau menghapus pengguna dan harga. Jika metode permintaan adalah POST, fungsi akan memproses operasi yang diminta, seperti menambahkan, mengedit, atau menghapus pengguna atau harga. Setelah itu, pengguna akan diarahkan kembali ke halaman data master. Jika metode permintaan adalah GET, halaman data master akan ditampilkan bersama dengan formulir untuk menambahkan pengguna atau harga.

b). uas\_cucikendaraan

1. settings.py

"""

Django settings for uas\_cucikendaraan project.

Generated by 'django-admin startproject' using Django 5.0.6.

For more information on this file, see

https://docs.djangoproject.com/en/5.0/topics/settings/

For the full list of settings and their values, see

https://docs.djangoproject.com/en/5.0/ref/settings/

"""

from pathlib import Path

# Build paths inside the project like this: BASE\_DIR / 'subdir'.

BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent

# Quick-start development settings - unsuitable for production

# See https://docs.djangoproject.com/en/5.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!

SECRET\_KEY = 'django-insecure-19$+7\_41aa^#=i#9z!ue4!vn#j^53y@f%=$szai@%y5j##z2s#'

# SECURITY WARNING: don't run with debug turned on in production!

DEBUG = True

ALLOWED\_HOSTS = []

# Application definition

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'cucikendaraan',

]

MIDDLEWARE = [

    'django.middleware.security.SecurityMiddleware',

    'django.contrib.sessions.middleware.SessionMiddleware',

    'django.middleware.common.CommonMiddleware',

    'django.middleware.csrf.CsrfViewMiddleware',

    'django.contrib.auth.middleware.AuthenticationMiddleware',

    'django.contrib.messages.middleware.MessageMiddleware',

    'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

AUTH\_USER\_MODEL = 'cucikendaraan.UserCuciKuy'

ROOT\_URLCONF = 'uas\_cucikendaraan.urls'

TEMPLATES = [

    {

        'BACKEND': 'django.template.backends.django.DjangoTemplates',

        'DIRS': [],

        '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',

            ],

        },

    },

]

WSGI\_APPLICATION = 'uas\_cucikendaraan.wsgi.application'

LOGIN\_URL = '/login/'

# Database

# https://docs.djangoproject.com/en/5.0/ref/settings/#databases

DATABASES = {

    'default': {

        'ENGINE': 'django.db.backends.sqlite3',

        'NAME': BASE\_DIR / 'db.sqlite3',

    }

}

# Password validation

# https://docs.djangoproject.com/en/5.0/ref/settings/#auth-password-validators

AUTH\_PASSWORD\_VALIDATORS = [

    {

        'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',

    },

    {

        'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',

    },

    {

        'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',

    },

    {

        'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',

    },

]

# Internationalization

# https://docs.djangoproject.com/en/5.0/topics/i18n/

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'UTC'

USE\_I18N = True

USE\_TZ = True

# Static files (CSS, JavaScript, Images)

# https://docs.djangoproject.com/en/5.0/howto/static-files/

STATIC\_URL = 'static/'

# Default primary key field type

# https://docs.djangoproject.com/en/5.0/ref/settings/#default-auto-field

DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'

Berikut adalah penjelasan untuk setiap bagian dalam file settings.py:

1. SECRET\_KEY: Ini adalah kunci rahasia yang digunakan untuk mengamankan sesi pengguna, cookie, dan informasi penting lainnya. Jangan pernah membagikan atau mempublikasikan kunci rahasia Anda.

2. DEBUG: Ini adalah pengaturan yang menentukan apakah Django berjalan dalam mode debug atau tidak. Diatur ke True di sini sehingga Anda dapat melihat pesan kesalahan secara detail saat mengembangkan aplikasi. Selalu atur ke False di produksi.

3. ALLOWED\_HOSTS: Ini adalah daftar string yang berisi nama host yang diizinkan untuk melayani proyek ini. Untuk pengembangan lokal, seringkali Anda hanya perlu menetapkan nilai ['\*'].

4. INSTALLED\_APPS: Ini adalah daftar aplikasi Django yang diinstal di proyek Anda. Aplikasi ini dapat menyediakan fungsionalitas yang diperlukan seperti otentikasi, manajemen admin, dan lainnya.

5. MIDDLEWARE: Ini adalah daftar middleware yang digunakan oleh aplikasi Anda. Middleware adalah perangkat lunak yang berjalan sebelum atau sesudah view Django, dan dapat digunakan untuk berbagai tujuan seperti otentikasi, penanganan CSRF, dan lainnya.

6. AUTH\_USER\_MODEL: Ini adalah model pengguna khusus yang digunakan oleh aplikasi. Di sini, model pengguna telah disesuaikan dengan model UserCuciKuy dari aplikasi Anda.

7. ROOT\_URLCONF: Ini adalah modul Python yang berisi konfigurasi URL untuk proyek Anda.

8. TEMPLATES: Ini adalah daftar konfigurasi template yang digunakan oleh aplikasi Anda. Konfigurasi ini mencakup backend template, direktori template, dan opsi tambahan seperti prosesor konteks.

9. WSGI\_APPLICATION: Ini adalah lokasi WSGI application Anda. WSGI (Web Server Gateway Interface) adalah standar untuk menghubungkan aplikasi web Python dengan server web seperti Apache atau Nginx.

10. DATABASES: Ini adalah konfigurasi untuk koneksi basis data. Di sini, basis data default menggunakan SQLite, sebuah basis data ringan yang disimpan dalam file.

11. AUTH\_PASSWORD\_VALIDATORS: Ini adalah daftar validator yang digunakan untuk memvalidasi kata sandi pengguna. Validator ini memastikan bahwa kata sandi yang digunakan memiliki kekuatan yang memadai.

12. LANGUAGE\_CODE dan TIME\_ZONE: Ini adalah pengaturan untuk lokal aplikasi, menentukan bahasa dan zona waktu yang digunakan dalam aplikasi.

13. USE\_I18 danUSE\_TZ: Pengaturan ini mengaktifkan atau menonaktifkan dukungan untuk internasionalisasi (I18N) dan zona waktu (TZ).

14. STATIC\_URL: Ini adalah URL yang digunakan untuk mengakses file statis seperti CSS, JavaScript, dan gambar.

15. DEFAULT\_AUTO\_FIELD: Ini adalah pengaturan untuk jenis bidang kunci utama default yang akan digunakan jika tidak ada yang ditentukan secara eksplisit. Di sini, digunakan BigAutoField sebagai default.

2. urls.py

"""

URL configuration for uas\_cucikendaraan project.

The `urlpatterns` list routes URLs to views. For more information please see:

    https://docs.djangoproject.com/en/5.0/topics/http/urls/

Examples:

Function views

    1. Add an import:  from my\_app import views

    2. Add a URL to urlpatterns:  path('', views.home, name='home')

Class-based views

    1. Add an import:  from other\_app.views import Home

    2. Add a URL to urlpatterns:  path('', Home.as\_view(), name='home')

Including another URLconf

    1. Import the include() function: from django.urls import include, path

    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))

"""

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', include('cucikendaraan.urls')),

]

File ini adalah file konfigurasi untuk routing URL dalam proyek Django Anda. Di dalamnya, terdapat daftar urlpatterns yang mengarahkan URL ke view-view tertentu dalam aplikasi Anda. Berikut adalah penjelasan untuk setiap bagian dalam file urls.py:

1. urlpatterns: Ini adalah daftar yang berisi beberapa path yang menentukan pola URL dan view yang akan dihubungkan dengannya. Pola URL dapat berupa string atau regex, dan view dapat berupa fungsi atau kelas view.

2. path('admin/', admin.site.urls): Ini adalah path yang mengarahkan ke aplikasi admin bawaan Django. Ketika Anda mengunjungi /admin/, Anda akan diarahkan ke antarmuka admin Django.

3. path('', include('cucikendaraan.urls')): Ini adalah path kosong yang memasukkan semua URL yang dimulai dari root ke dalam aplikasi cucikendaraan. Dengan menggunakan fungsi include, Anda dapat memasukkan URL yang didefinisikan dalam file cucikendaraan.urls ke dalam URL proyek Anda.

File urls.py ini bertindak sebagai hub utama untuk mengatur routing URL dalam aplikasi Anda. Dengan mengatur urlpatterns dengan benar, Anda dapat menentukan bagaimana setiap URL akan dipetakan ke view-view tertentu dalam aplikasi Anda.

1. **tambah\_price.py**

import os

import django

os.environ.setdefault("DJANGO\_SETTINGS\_MODULE", "uas\_cucikendaraan.settings")

django.setup()

from cucikendaraan.models import Price

def tambah\_harga():

    pilihan\_kendaraan = input("Jenis Kendaraan (motor/mobil): ").strip().lower()

    if pilihan\_kendaraan not in ['motor', 'mobil']:

        print("Jenis kendaraan tidak valid. Pilih 'motor' atau 'mobil'.")

        return

    # Harga otomatis berdasarkan jenis kendaraan

    if pilihan\_kendaraan == 'motor':

        harga = 9000.00

    elif pilihan\_kendaraan == 'mobil':

        harga = 20000.00

    new\_price = Price(jeniskendaraan=pilihan\_kendaraan, harga=harga)

    new\_price.save()

    print(f"Harga untuk kendaraan '{pilihan\_kendaraan}' berhasil ditambahkan dengan harga {harga}.")

if \_\_name\_\_ == "\_\_main\_\_":

    tambah\_harga()

Program Python ini adalah bagian dari aplikasi Django yang bertujuan untuk menambahkan harga baru untuk jenis kendaraan tertentu ke dalam database. Berikut adalah penjelasan lebih rinci:

1. Program ini dimulai dengan mengimpor modul os dan django untuk melakukan setup environment Django.

2. Fungsi os.environ.setdefault() digunakan untuk menetapkan nilai default untuk variabel lingkungan DJANGO\_SETTINGS\_MODULE yang menunjukkan modul pengaturan Django yang akan digunakan.

3. django.setup() dipanggil untuk menginisialisasi pengaturan Django dari modul yang telah ditentukan sebelumnya.

4. Selanjutnya, model Price dari aplikasi Django (cucikendaraan.models) diimpor. Model Price ini digunakan untuk merepresentasikan harga kendaraan dalam database.

5. Fungsi tambah\_harga() berisi logika untuk menambahkan harga baru ke dalam database. Pengguna diminta untuk memasukkan jenis kendaraan (motor/mobil) melalui input. Kemudian, program memeriksa apakah jenis kendaraan yang dimasukkan valid (motor atau mobil). Jika tidak valid, pesan kesalahan akan dicetak.

6. Jika jenis kendaraan valid, harga kendaraan ditentukan berdasarkan jenisnya. Harga tersebut kemudian digunakan untuk membuat objek Price baru dengan menggunakan model Price dari Django. Objek Price ini kemudian disimpan ke dalam database.

7. Setelah harga berhasil ditambahkan, pesan sukses akan dicetak yang menyatakan bahwa harga untuk jenis kendaraan tersebut telah ditambahkan ke dalam database.

8. Pada bagian \_\_name\_\_ == "\_\_main\_\_":, fungsi tambah\_harga() dipanggil untuk dieksekusi ketika program dijalankan sebagai skrip mandiri.

Program ini memberikan fungsionalitas yang berguna dalam mengelola harga kendaraan dalam aplikasi Django dan memudahkan penggunaan database untuk menambahkan harga baru secara interaktif.

1. **tambah\_user.py**

import os

import django

os.environ.setdefault("DJANGO\_SETTINGS\_MODULE", "uas\_cucikendaraan.settings")

django.setup()

from cucikendaraan.models import UserCuciKuy

def tambah\_user():

    nama = input("Nama: ")

    username = input("Username: ")

    password = input("Password: ")

    new\_user = UserCuciKuy(nama=nama, username=username, password=password)

    new\_user.save()

    print(f"User {username} berhasil ditambahkan.")

if \_\_name\_\_ == "\_\_main\_\_":

    tambah\_user()

Program Python ini merupakan bagian dari aplikasi Django yang bertujuan untuk menambahkan pengguna baru ke dalam database. Berikut adalah penjelasan singkatnya:

1. Program dimulai dengan mengimpor modul os dan django untuk melakukan setup environment Django.

2. Fungsi os.environ.setdefault() digunakan untuk menetapkan nilai default untuk variabel lingkungan DJANGO\_SETTINGS\_MODULE, yang menunjukkan modul pengaturan Django yang akan digunakan

3. django.setup() dipanggil untuk menginisialisasi pengaturan Django dari modul yang telah ditentukan sebelumnya.

4. Selanjutnya, model UserCuciKuy dari aplikasi Django (cucikendaraan.models) diimpor. Model UserCuciKuy ini digunakan untuk merepresentasikan pengguna dalam database.

5. Fungsi tambah\_user() berisi logika untuk menambahkan pengguna baru ke dalam database. Pengguna diminta untuk memasukkan nama, nama pengguna (username), dan kata sandi (password) melalui input.

6. Kemudian, objek UserCuciKuy baru dibuat dengan menggunakan model UserCuciKuy dari Django. Objek ini kemudian disimpan ke dalam database.

7. Setelah pengguna berhasil ditambahkan, pesan sukses akan dicetak yang menyatakan bahwa pengguna tersebut telah berhasil ditambahkan.

8. Pada bagian \_\_name\_\_ == "\_\_main\_\_":, fungsi tambah\_user() dipanggil untuk dieksekusi ketika program dijalankan sebagai skrip mandiri.

Program ini memberikan fungsionalitas yang berguna dalam mengelola pengguna dalam aplikasi Django dan memungkinkan penambahan pengguna baru secara interaktif.

1. **Tampilan Web**
2. Login

A screenshot of a login form

Description automatically generated

1. Beranda

A screenshot of a computer

Description automatically generated

1. Manajemen Transaksi

A screenshot of a computer

Description automatically generated

Didalam manajemen transaksi terdapat beberapa fitur selain simpan transaksi yaitu :

1. Edit

A screenshot of a computer

Description automatically generated

1. Hapus

A screenshot of a computer

Description automatically generated

1. Cetak Nota

A close-up of a white background

Description automatically generated

1. Manajemen Data Master

A screenshot of a computer

Description automatically generated

Didalam manajemen data master terdapat fitur lain yaitu:

1. Tambah User

A screenshot of a computer

Description automatically generated

1. Edit Harga

A screenshot of a computer

Description automatically generated

1. Edit User

A screenshot of a computer

Description automatically generated

1. Hapus User

A screenshot of a computer

Description automatically generated

1. **Kesimpulan**

Berdasarkan laporan proyek Django yang telah dilakukan, dapat ditarik beberapa kesimpulan yaitu:

1. Proyek ini merupakan aplikasi web yang dikembangkan menggunakan framework Django untuk manajemen cuci kendaraan.

2. Terdapat direktori program "cucikendaraan" yang berisi sub-direktori seperti "Images" dan "Migrations" yang digunakan untuk menyimpan gambar dan skrip migrasi database.

3. Skrip migrasi database yang dikelola oleh Django digunakan untuk mengubah skema database, seperti pembuatan tabel baru dan definisi field-field dalam tabel.

4. Tabel "User" yang didefinisikan dalam migrasi tersebut memiliki beberapa field penting seperti id, last\_login, username, nama, dan password untuk menyimpan informasi pengguna.

5. Aplikasi web ini memiliki fitur manajemen transaksi yang meliputi simpan transaksi, edit, hapus, dan cetak nota.

6. Halaman admin panel menyediakan statistik layanan dengan grafik dinamis yang memperlihatkan perkembangan jumlah transaksi dari bulan ke bulan.

Dengan demikian, laporan proyek ini memberikan gambaran tentang pengembangan aplikasi web menggunakan Django untuk manajemen cuci kendaraan beserta fitur-fitur yang disediakan.