PROJECT = Inventory

APLIKASI = stock

Web aplikasi diakses di http://127.0.0.1:8000/

Kita upgrade aplikasi pip -> pip install –upgrade pip atau python -m pip install –upgrade pip

atau env\Scripts\python.exe –m pip install –upgrade pip

Kita install aplikasi virtual environment -> pip install virtualenv

Kita buat folder virtual environment -> virtualenv env

Kita aktifkan virtual environemnt -> env\Scripts\activate

Instal Django -> pip install django==3.2.8

Cara cek versi Django -> django-admin –version

Buat project awal Django -> django-admin startproject Inventory

Buat app yang kita namakan posts -> python manage.py startapp stock

Buka editor VS Code -> code .

Masukkan kode ke dalam file Inventory/settings.py:

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'stock',

]

Pada variabel TEMPLATES:

TEMPLATES = [

    {

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

        'DIRS': ['templates'],

        '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',

            ],

        },

    },

]

Pada file stocks/views.py tambahkan fungsi index:

*from* django.shortcuts *import* render

*# Create your views here.*

*def* index(request):

*return* render(request, 'stock/index.html')

Pada Pada file Inventory/urls.py tambahkan import views dan routing sbb:

*from* django.contrib *import* admin

*from* django.urls *import* path

*from* stock.views *import* \*

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', index, name='index'),

]

Buat folder templates di dalam direktori root/utama, dan di dalam direktori templates buat folder lagi bernama stock dan buat file HTML di dalamnya yang bernama templates/stock/index.html :

<center><h1>HALAMAN UTAMA</h1></center>

Tampil di browser:



**Extending Templates**

Buat file HTML yang bernama templates/base.html dengan kode:

<!doctype html>

<html lang="en">

{% *load* static %}

<head>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <title></title>

  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css" integrity="sha384-GJzZqFGwb1QTTN6wy59ffF1BuGJpLSa9DkKMp0DgiMDm4iYMj70gZWKYbI706tWS" crossorigin="anonymous">

  <link href="#" rel="stylesheet">

  <link rel="shortcut icon" type="image/png" href="#" />

</head>

<body>

  <div class="d-flex flex-column flex-md-row align-items-center p-3 px-md-4 bg-white border-bottom shadow-sm">

    <h5 class="my-0 mr-md-auto font-weight-normal"><a href="{% url 'index' %}" class="text-dark">Beranda</a></h5>

    <nav class="my-2 my-md-0 mr-md-3">

      <a class="p-2 text-dark" href="#">Add Stock</a>

      <a class="p-2 text-dark" href="#">Log Out</a>

      <a class="p-2 text-dark" href="#">Log In</a>

    </nav>

    <a class="btn btn-outline-primary" href="#">Sign up</a>

  </div>

  {% *block* content %}

  {% *endblock* %}

*<!-- Optional JavaScript -->*

*<!-- jQuery first, then Popper.js, then Bootstrap JS -->*

  <*script* src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></*script*>

  <*script* src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></*script*>

  <*script* src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></*script*>

</body>

</html>

Pada file templates/stock/index.html cantumkan extends dari base.html:

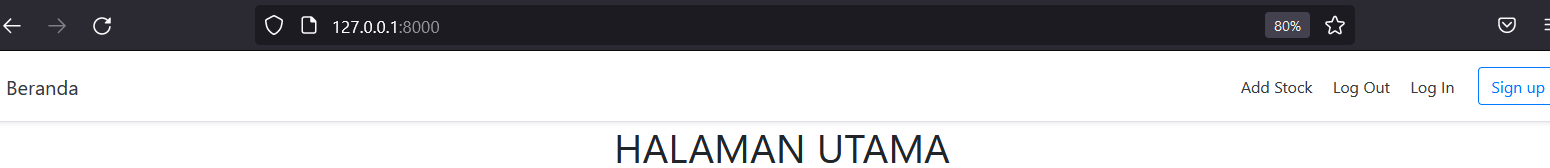
{% *extends* 'base.html' %}

{% *block* content %}

<center><h1>HALAMAN UTAMA</h1></center>

{% *endblock* %}

Tampilan di browser:



**STATIC FILE**

Masukkan kode static ke dalam file Inventory/settings.py:

*# Static files (CSS, JavaScript, Images)*

*# https://docs.djangoproject.com/en/3.2/howto/static-files/*

STATIC\_URL = '/static/'

STATIC\_ROOT = os.path.join(BASE\_DIR, 'static')

Pada file Inventory/urls.py tambahkan import dan kode static file:

from django.conf.urls.static import static  
from django.conf import settings

urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_ROOT)

Kode selengkapnya:

*from* django.contrib *import* admin

*from* django.urls *import* path

*from* stock.views *import* \*

*from* django.conf.urls.static *import* static

*from* django.conf *import* settings

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', index, name='index'),

]

urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_ROOT)

Buat folder baru di dalam direktori root/utama bernama /static dan masukkan kedua file: custom.css dan favicon.ico ke dalam folder tersebut.

Adapun isi koding dari file halls/static/custom.css adalah sbb:

.jumbotron {

  background-color: #EBF4FA;

}

.jumbotron-heading {

  font-weight: 300;

}

.jumbotron .container {

  max-width: 51rem;

}

Adapun isi koding dari file static/custom.css adalah sbb:

.jumbotron {

  background-color: #EBF4FA;

}

.jumbotron-heading {

  font-weight: 300;

}

.jumbotron .container {

  max-width: 51rem;

}

Setelah itu pada file templates/base.html di bagian tag <head> masukkan link kode bagi kedua file static diatas:

<head>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <title>Aplikasi Stock</title>

  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css" integrity="sha384-GJzZqFGwb1QTTN6wy59ffF1BuGJpLSa9DkKMp0DgiMDm4iYMj70gZWKYbI706tWS" crossorigin="anonymous">

  <link href="#" rel="stylesheet">

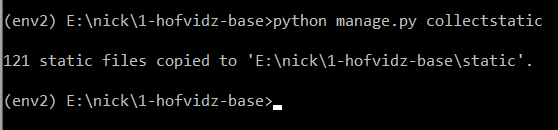
  <link rel="shortcut icon" type="image/png" href="#" />

  <link href="{% static 'custom.css' %}" rel="stylesheet">

  <link rel="shortcut icon" type="image/png" href="{% static 'favicon.ico' %}" />

</head>

Setelah memasukkan kedua file diatas jalankan perintah -> python manage.py collectstatic



Perintah di atas akan mengcopy file-file static sebanyak 121 buah ke dalam folder baru yang bernama static dalam direktori utama.

**AUTHENTICATION VIEWS**

Pada file stock/views.py tambahkan import dan fungsi sbb:

*from* django.shortcuts *import* render

*from* django.urls *import* reverse\_lazy

*from* django.views *import* generic

*from* django.contrib.auth.forms *import* UserCreationForm

*# Create your views here.*

*# Create your views here.*

*def* index(request):

*return* render(request, 'stock/index.html')

*class* SignUp(generic.CreateView):

    form\_class = UserCreationForm

    success\_url = reverse\_lazy('index')

    template\_name = 'registration/signup.html'

Pada file Inventory/urls.py tambahkan import dan kode routing signup:

*from* django.contrib *import* admin

*from* django.urls *import* path

*from* stock.views *import* \*

*from* django.conf.urls.static *import* static

*from* django.conf *import* settings

*from* stock *import* views

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', index, name='index'),

*# Registration*

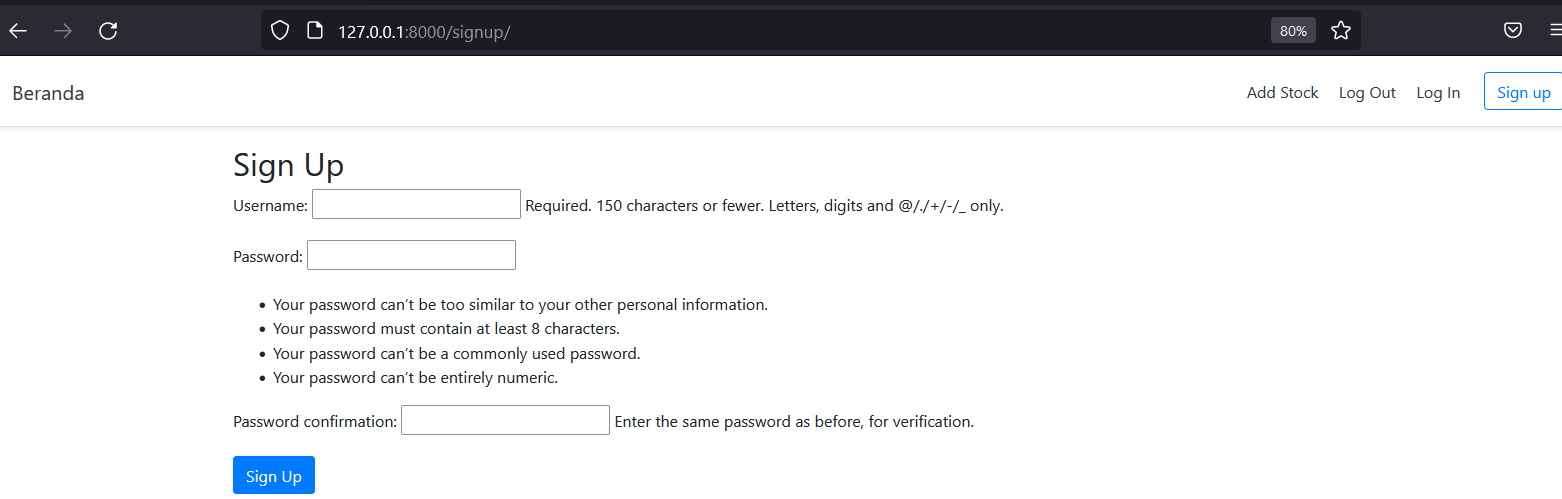
    path('signup/', views.SignUp.as\_view(), name='signup'),

]

urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_ROOT)

**AUTHENTICATION TEMPLATES**

Buat folder baru yang bernama halls/templates/registration dan buat file HTML yang bernama templates/registration/signup.html yang isi kodenya sbb:



Halaman Sign-Up ini belim bisa digunakan karena kita belum membuat super user. Pada file file Inventory/urls.py tambahkan import from django.contrib.auth import views as auth\_views dan kode routing login dan logout, kode selengkapnya :

*from* django.contrib *import* admin

*from* django.urls *import* path

*from* stock.views *import* \*

*from* django.conf.urls.static *import* static

*from* django.conf *import* settings

*from* stock *import* views

*from* django.contrib.auth *import* views *as* auth\_views

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', index, name='index'),

*# Registration*

    path('signup/', views.SignUp.as\_view(), name='signup'),

    path('login/', auth\_views.LoginView.as\_view(), name='login'),

    path('logout/', auth\_views.LogoutView.as\_view(), name='logout'),

]

urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_ROOT)

Dalam pembuatan file templates/registration/login.html kita bisa paste-kan dari file signup.html dengan hanya merubah sedikit, kode lengkapnya sbb:

{% *extends* 'base.html' %}

{% *block* content %}

<div class="container mt-3">

<h2>Login</h2>

<form method="post">

    {% *csrf\_token* %}

    {{ form.as\_p }}

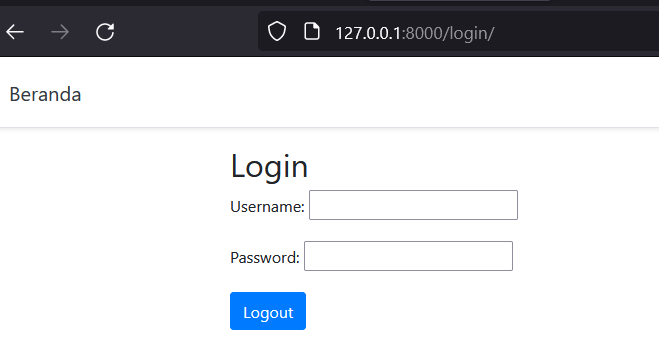
    <button type="submit" class="btn btn-primary">Login</button>

</form>

</div>

{% *endblock* %}

Tampilan di browser:



Pada file templates/base.html tambahkan link untuk menu-menu registrasi:

      <a class="p-2 text-dark" href="{% url 'logout' %}">Log Out</a>

      <a class="p-2 text-dark" href="{% url 'login' %}">Log In</a>

    </nav>

    <a class="btn btn-outline-primary" href="{% url 'signup' %}">Sign up</a>

Kemudian user di-redirect ke halaman beranda jika melakukan login dan logut, caranya tambahkan kode pada file Inventory/settings.py

ALLOWED\_HOSTS = []

LOGIN\_URL = 'login'

LOGIN\_REDIRECT\_URL = 'index'

LOGOUT\_REDIRECT\_URL = 'index'

Batasi user dalam mengakses Navbar dengan menambahkan kondisional if pada file templates/base.html:

  <div class="d-flex flex-column flex-md-row align-items-center p-3 px-md-4 bg-white border-bottom shadow-sm">

    <h5 class="my-0 mr-md-auto font-weight-normal"><a href="{% url 'index' %}" class="text-dark">Beranda</a></h5>

    <nav class="my-2 my-md-0 mr-md-3">

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

      <a class="p-2 text-dark" href="#">Add Stock</a>

      <a class="p-2 text-dark" href="{% url 'logout' %}">Log Out</a>

      {% *else* %}

      <a class="p-2 text-dark" href="{% url 'login' %}">Log In</a>

      {% *endif* %}

    </nav>

    <a class="btn btn-outline-primary" href="{% url 'signup' %}">Sign up</a>

  </div>

Nah sekarang, fitur signup, login dan logout sudah berhasil dikerjakan.  
 **Database Migrations**

Pada file stock/models.py buat tabel-tabel database sbb:

*from* django.db *import* models

*# Create your models here.*

*class* Stock(models.Model):

    title = models.CharField(max\_length=155)

    date = models.DateField()

    description = models.TextField(max\_length=1555)

    jumlah = models.IntegerField(null=True, blank=True)

    harga = models.IntegerField(null=True, blank=True)

*def* *\_\_str\_\_*(self):

*return* *self*.title

Pada file stock/admin.py isikan kode sbb:

*from* django.contrib *import* admin

*from* .models *import* Stock

*# Register your models here.*

admin.site.register(Stock)

Jalankan perintah -> py manage.py makemigrations

Disusul -> py manage.py migrate

Lalu py manage.py createsuperuser

Isikan databasenya pada halaman admin dan tampilkan di halaman index.html, dan pada file stock/views.py tambahkan kode :

*from* .models *import* \*

*# Create your views here.*

*def* index(request):

    context = {

        'stocks': Stock.objects.all()

    }

*return* render(request, 'stock/index.html', context)

Pada file templates/stock/index.html isikan kode sbb:

{% *extends* 'base.html' %}

{% *block* content %}

<center><h1>Aplikasi Stok</h1></center>

{% *for* stock *in* stocks %}

<div class="card" style="width: 88rem; margin-bottom: 30px; padding-left: 155px;">

    <div class="card-header">

        <h3>{{ stock.title }}</h3>

    </div>

    <div class="card-body">

        <small>{{ stock.date }}</small>

      <h5 class="card-title"><strong>Jumlah: </strong>{{ stock.jumlah }}</h5>

      <h6><strong>Harga: </strong>{{ stock.harga }}</h6>

      <p class="card-text">{{ stock.description|truncatechars:59 }} </p>

      <a href="#" class="btn btn-primary">More</a>

    </div>

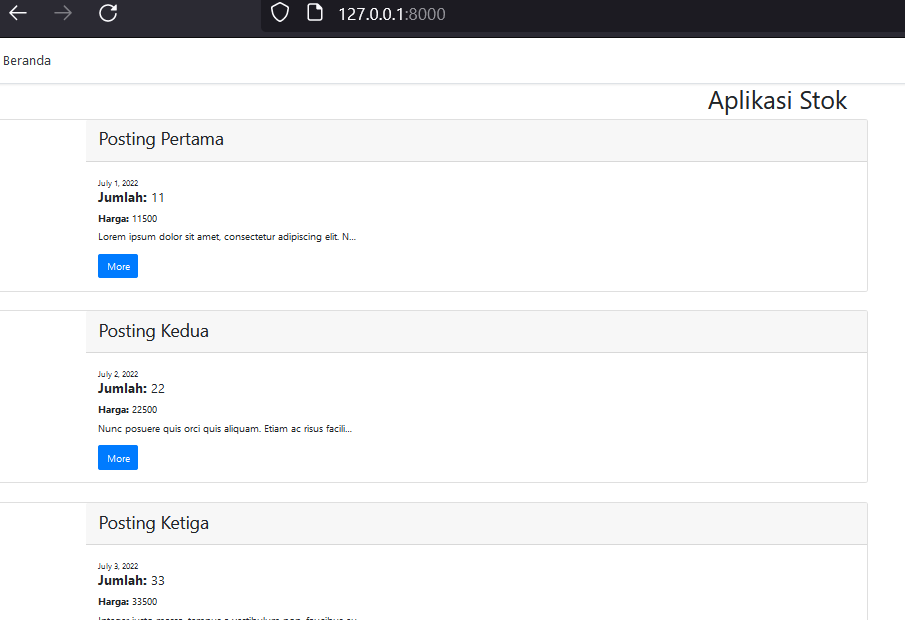
  </div>

{% *endfor* %}

</div>

{% *endblock* %}

Tampilan di browser:

****

**CLASS BASED VIEWS – CRUD**

Dengan Class Based Views, kita menghemat waktu pekerjaan, karena hanya butuh sedikit kode atau less code, kali ini ini kita membuat halaman membuat posting video, jadi kita tuliskan kode di file stock/views.py dengan mengimport dari models.py dan kode lengkap sbb:

*from* django.shortcuts *import* render, redirect

*from* django.urls *import* reverse\_lazy

*from* django.views *import* generic

*from* django.contrib.auth.forms *import* UserCreationForm

*from* .models *import* \*

*# Create your views here.*

*def* index(request):

*return* render(request, 'stock/index.html')

*class* SignUp(generic.CreateView):

    form\_class = UserCreationForm

    success\_url = reverse\_lazy('index')

    template\_name = 'registration/signup.html'

*class* CreateStock(generic.CreateView):

    model = Stock

    fields = ['title', 'date', 'description', 'jumlah', 'harga']

    template\_name = 'stock/createstock.html'

    success\_url = reverse\_lazy('index')

Selanjutnya buat kode routing di file Inventory/urls.py :

*# CRUD*

    path('createstock/', views.CreateStock.as\_view(), name='createstock'),

**Membuat Halaman Detail**

Pertama pada file stock/views.py kita tambahkan kode :

*class* DetailStock(generic.DetailView):

    model = Stock

    template\_name = 'stock/detailstock.html'

Buat routing URL di file Inventory/urls.py:

path('detail/<int:pk>', views.DetailStock.as\_view(), name='detailstock'),

Kita buat file HTML baru bernama templates/stock/detailstock.html

{% *extends* 'base.html' %}

{% *block* content %}

<div class="card" style="width: 88rem; margin-bottom: 30px; padding-left: 155px;">

    <div class="card-header">

        <h3>{{ stock.title }}</h3>

    </div>

    <div class="card-body">

        <small>{{ stock.date }}</small>

      <h5 class="card-title"><strong>Jumlah: </strong>{{ stock.jumlah }}</h5>

      <h6><strong>Harga: </strong>{{ stock.harga }}</h6>

      <p class="card-text">{{ stock.description }} </p>

      <a href="#" class="btn btn-primary">More</a>

    </div>

  </div>

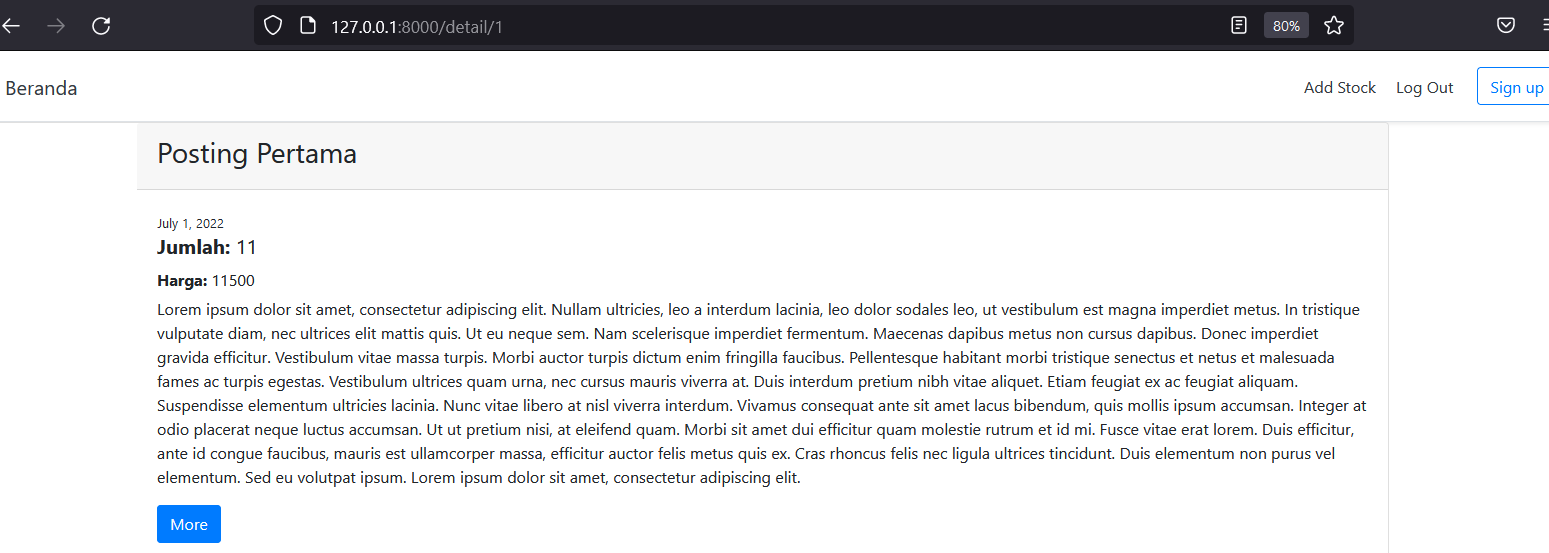
</div>

{% *endblock* %}

Terakhir kita buatkan link di halaman index.html untuk menuju ke halaman detailstock.html, jadi saat user mengklik tombol “More” akan diarahkan menuju halaman detail stock.

      <a href="{% url 'detailstock' stock.pk %}" class="btn btn-primary">More</a>

Berikut tampilan di browser:



**Update dan Delete**

Kita akan membuat fitur Update dan Delete Hall, pertama kita buat class di file stock/views.py yang bernama class def UpdateStock:

*class* UpdateStock(generic.UpdateView):

    model = Stock

    template\_name = 'stock/updatestock.html'

    fields = ['title', 'date', 'description', 'jumlah', 'harga']

    success\_url = reverse\_lazy('index')

Pada file Inventory/urls.py buat kode routingnya:

    path('detail/<int:pk>/update', views.UpdateStock.as\_view(), name='updatestock'),

Buat file HTML yang bernama templates/stock/updatestock.html sekaligus kita juga membuat tombol “Delete” dengan isi kode sbb :

{% *extends* 'base.html' %}

{% *block* content %}

<div class="container mt-5 mb-5">

    <h2>Edit Stock</h2>

<form method="post">

    {% *csrf\_token* %}

    {{ form.as\_p }}

    <br>

    <button type="submit" class="btn btn-primary">Save</button>

    <br>

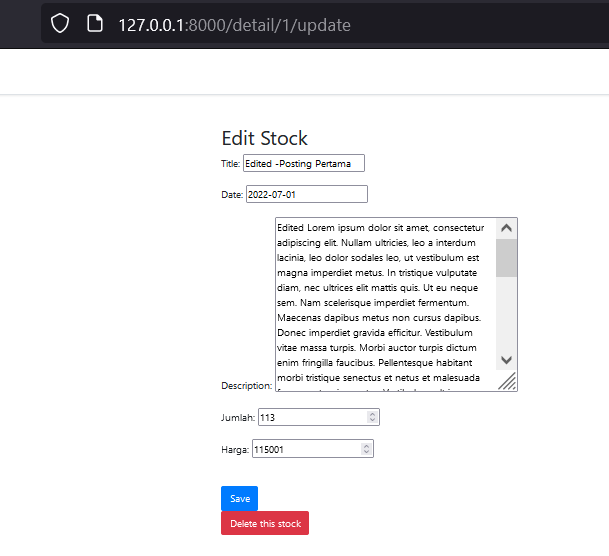
    <a href="#" class="btn btn-danger">Delete this stock</a>

</form>

</div>

{% *endblock* %}

Tampilan di browser:



**Delete Stock**

Kita tambahkan link di halaman bawah dari file templates/stock/updatestock.html untuk menghapus stock:

    <br>

    <a href="{% url 'deletestock' stock.id %}" class="btn btn-danger">Delete this stock</a>

Pada file templates/stock/updatestock.html tambahkan tombol link untuk delete

      <br>

      <br>

      <a href="{% url 'deletestock' stock.id %}" class="btn btn-danger">Delete this stock</a>

Pada file Inventory/urls.py tambahkan kode routing:

path('detail/<int:pk>/delete', views.DeleteStock.as\_view(), name='deletestock'),

Pada file stock/views.py kita buat sebuah fungsi:

*class* DeleteStock(generic.DeleteView):

    model = Stock

    template\_name = 'stock/deletestock.html'

    success\_url = reverse\_lazy('index')

Kita buat file HTML yang bernama templates/stock/deletestock.html :

{% *extends* 'base.html' %}

{% *block* content %}

<div class="container mt-5">

    <h2>Are you sure you want to delete this stock?</h2>

<form method="post">

    {% *csrf\_token* %}

    {{ form.as\_p }}

    <br>

    <button type="submit" class="btn btn-danger">Delete</button>

</form>

<br>

<h2><a href="{% url 'detailstock' stock.id %}">{{ stock.title }}</a></h2>

</div>

{% *endblock* %}

Kita perlu benahi link registrasi di Navbar di file templates/base.html :

  <div class="d-flex flex-column flex-md-row align-items-center p-3 px-md-4bg-white border-bottom shadow-sm">

    <h5 class="my-0 mr-md-auto font-weight-normal"><a href="{% url 'index' %}" class="text-dark">Beranda</a></h5>

    <nav class="my-2 my-md-0 mr-md-3">

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

      <a class="p-2 text-dark" href="{% url 'createstock' %}">Add Stock</a>

      <a class="p-2 text-dark" href="{% url 'logout' %}">Log Out</a>

      {% *else* %}

      <a class="p-2 text-dark" href="{% url 'login' %}">Log In</a>

      <a class="btn btn-outline-primary" href="{% url 'signup' %}">Sign up</a>

      {% *endif* %}

Terakhir kita buat tampilan daftar stok barang dalam bentuk tabel, jadi kita rubah kode HTML nya sbb:

{% *extends* 'base.html' %}

{% *block* content %}

<center><h1>Aplikasi Stok</h1></center>

<div class="container">

  <div class="row">

      <div class="col-md-12">

          <table class="table table-bordered table-striped">

              <thead>

                      <tr>

                          <th><h2>ITEM</h2></th>

                          <th><h2>JUMLAH</th></h2>

                          <th><h2>HARGA</th></h2>

                          <th><h2>DESKRIPSI</th></h2>

                      </tr>

                      <br>

                      <br>

                  <tbody>

                      {% *for* stock *in* stocks %}

                      <tr>

                          <td><strong>{{ stock.title }}</strong></td>

                          <td>Jumlah: <h4><b>{{ stock.jumlah }}</h4></b></td>

                          <td>Harga  <h4><b>{{ stock.harga }}</h4></b></td>

                          <td><h4><b>{{ stock.description|truncatechars:29 }}

                            <a href="{% url 'detailstock' stock.pk %}" class="btn btn-primary">More</a>

                          </h4></b></td>

                      </tr>

                      {% *endfor* %}

                  </tbody>

              </thead>

          </table>

      </div>

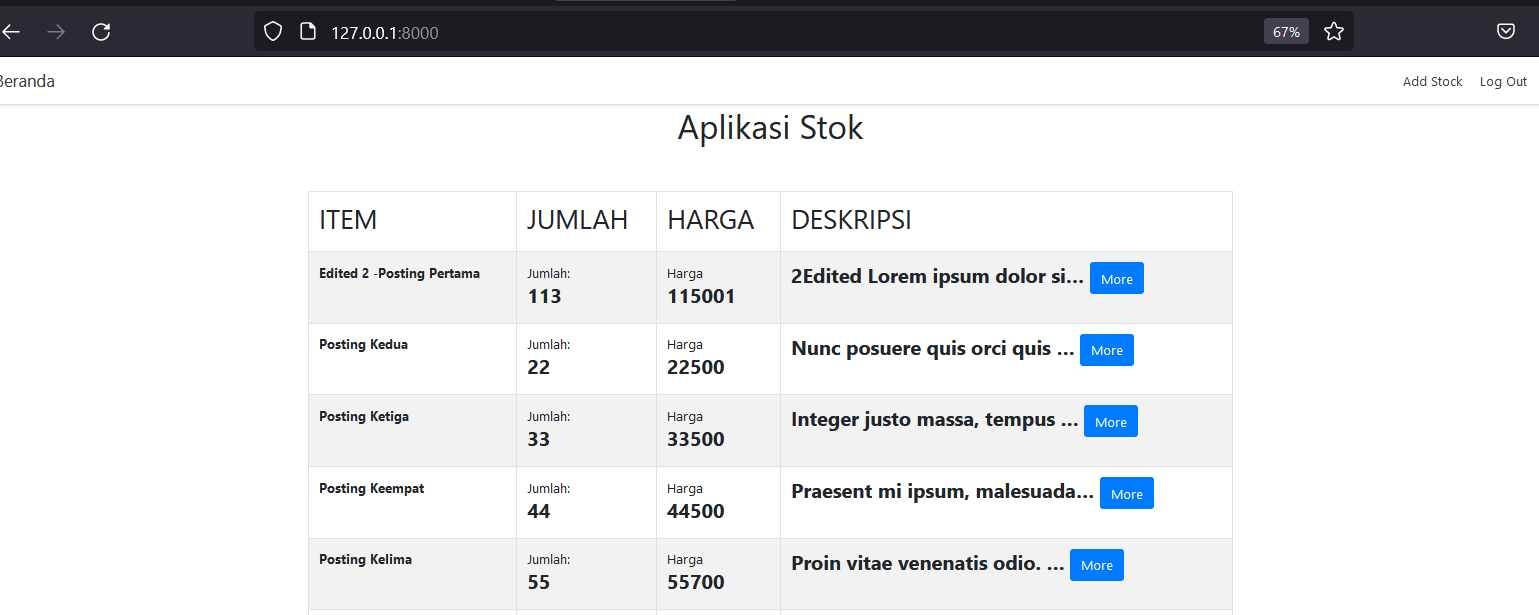
  </div>

</div>

</div>

{% *endblock* %}

Sehingga tampilan di browser:



Tambahan untuk urutan item stock agar postingan terakhir menempati urutan pertama pada file stock/views.py kita tambahkan kode di fungsi index :

*def* index(request):

    context = {

        'stocks': Stock.objects.all().order\_by('-pk')

    }

*return* render(request, 'stock/index.html', context)

Sumber :

Udemy Nick Walter - Mastering Django - AJAX, Class Based Views, Forms