DOSYA YÖNETİMİ UYGULAMASI

Hilalnur Aydın
hilal.nur.ay59@gmail.com

Contents

1. Giriş

Bu raporda, kullanıcıların web arayüzü üzerinden dosya yükleme, yönetme ve görüntüleme işlemlerini gerçekleştiren basit bir dosya yönetimi uygulaması ele alınmıştır. Proje, Python tabanlı Flask framework kullanılarak geliştirilmiş ve frontend tasarımı için Bootstrap kütüphanesi tercih edilmiştir. Amaç, dosya yönetim süreçlerini kolaylaştıran ve temel ihtiyaçları karşılayan fonksiyonel bir web uygulaması oluşturmaktır.

2. Problem Tanımı ve Çözüm Yaklaşımı

Günümüzde pek çok uygulama, kullanıcıların dosya yükleyip yönetebilmesine ihtiyaç duymaktadır. Bu proje kapsamında, kullanıcıların kolayca dosya yükleyip mevcut dosyalarını görüntüleyebileceği, hızlı ve güvenli bir sistem geliştirmek hedeflenmiştir.
Flask, minimal yapısı ve hızlı geliştirme imkanı sunduğu için backend çözümü olarak seçilmiştir. Bootstrap ise frontend için hızlı ve estetik bir tasarım yapmaya olanak sağlamıştır. Böylece hem backend hem frontend tarafında verimli bir çözüm geliştirilmiştir.

3. Kullanılan Teknolojiler

- **Python ve Flask:** Flask, Python diliyle uyumlu, hafif ve geniş topluluk desteği olan bir mikro frameworktür. Projenin ihtiyaç duyduğu dosya yükleme ve HTTP isteklerini yönetme işlevlerini kolayca sağlar.
- **Bootstrap:** Responsive ve modern kullanıcı arayüzü oluşturmak için tercih edilmiştir. Bootstrap sayesinde kısa sürede estetik ve mobil uyumlu tasarımlar yapılmıştır.
- **HTML & CSS:** Web sayfasının temel yapısını ve stilini oluşturmak için kullanılmıştır.
- **Git & GitHub:** Proje versiyon kontrolü ve paylaşımı için kullanılmıştır. Bu sayede proje gelişimi kayıt altına alınmış ve kolay paylaşım sağlanmıştır.

4. Proje Yapısı ve Akışı

Proje dosya yapısı aşağıdaki gibidir:

dosya_yonetimi_uygulamasi/

|

├— static/ # Statik dosyalar

| ├— style.css

├— templates/ # HTML şablonları

| ├— index.html

| ├— login.html

| ├— register.html

| ├ — upload.html

|

├── uploads/ # Yüklenen dosyaların saklandığı klasör (otomatik oluşur)

├── app.py # Flask uygulamasının ana dosyası

├── database.db # SQLite veritabanı dosyası

├── .git/ # Git versiyon kontrol klasörü

├── _pycache_/ # Python derlenmiş dosyaları (otomatik oluşur)

├── instance/ # Flask instance klasörü (config için olabilir)

└── venv/

Kullanıcılar, tarayıcı üzerinden dosya yükleyebilmektedir. Yüklenen dosyalar uploads/ klasörüne kaydedilmekte, arayüzde bu dosyalar listelenmektedir. Flask, gelen istekleri karşılar ve dosya yönetimi işlemlerini gerçekleştirir.

5. Teknik Detaylar ve Çözüm Yaklaşımı

- **HTML Kullanımı:** Uygulamanın kullanıcı arayüzü, templates/ klasöründeki HTML dosyalarıyla oluşturulmuştur. Flask'ın Jinja2 şablon motoru sayesinde dinamik içerik gösterimi sağlanır. Örneğin, kullanıcı tarafından yüklenen dosyalar liste halinde HTML içinde döngü yapılarak gösterilir. Form elemanları aracılığıyla dosya yükleme işlemi gerçekleştirilir.
- **CSS ve Bootstrap Kullanımı:** Uygulamanın tasarımında Bootstrap framework'ü kullanılmıştır. Bootstrap, responsive ve modern görünümlü kullanıcı arayüzü oluşturmak için tercih edilmiştir. Bootstrap CSS dosyaları static/ klasöründe yer alır ve HTML dosyalarına link olarak eklenir. Form elemanlarının, butonların ve sayfa düzeninin stilize edilmesinde Bootstrap sınıfları kullanılmıştır. Böylece farklı ekran boyutlarına uyum sağlayan esnek ve estetik tasarım elde edilmiştir. Ayrıca, gerektiğinde özel CSS kuralları static/ klasöründe ek CSS dosyaları oluşturularak projeye dahil edilebilir.
- **Flask Uygulaması:** app.py dosyasında HTTP istekleri yönetilir. Ana sayfada dosya yükleme formu yer alır. POST isteği ile gönderilen dosya, Python'un werkzeug.utils kütüphanesinden secure_filename() fonksiyonu kullanılarak güvenli hale getirilir ve uploads/ klasörüne kaydedilir.
- **Dosya Yönetimi:** Kullanıcıların yüklediği dosyalar arayüzde listelenir. Böylece hangi dosyaların yüklendiği kolayca takip edilir. Dosya isimleri güvenlik için temizlenir, ancak ileride dosya türü kontrolü ve boyut sınırlandırması eklenebilir.

6. Karşılaşılan Zorluklar ve Çözümleri

- **.git Klasörü Karışıklığı:** Proje dizini içinde yanlışlıkla birden fazla .git klasörü oluştu. Bu durum versiyon kontrolünde karışıklık ve hatalara sebep oldu. Fazla .git klasörleri kaldırılarak sorun giderildi.
- **Sanal Ortam Aktifleştirme Sorunu:** Windows PowerShell üzerinde virtual environment (venv) aktifleştirme işlemi başta sorun çıkardı. Set-ExecutionPolicy komutu ile gerekli izinler verilerek sorun çözüldü.

- **GitHub'a Yükleme:** Branch isimlendirmesi (master yerine main) ve remote ayarlarında yaşanan zorluklar komut satırı ile aşılmıştır.

7. Sonuç ve Değerlendirme

Bu proje, temel düzeyde dosya yükleme ve yönetimi sağlayan fonksiyonel bir web uygulaması olarak tamamlanmıştır. Flask ve Bootstrap kullanımıyla hızlı geliştirme ve estetik arayüz tasarımı sağlanmıştır. Proje, temel ihtiyaçları karşılamakla beraber, ileride güvenlik, kullanıcı yönetimi ve dosya filtreleme gibi ek özelliklerle geliştirilebilir.

8. Gelecek Çalışmalar

- Dosya türü ve boyut kısıtlamaları eklenerek güvenlik artırılabilir.
- Kullanıcı kimlik doğrulama ve yetkilendirme mekanizması entegre edilebilir.
- Dosya indirme ve silme özellikleri eklenebilir.
- Daha kapsamlı hata yönetimi ve kullanıcı bilgilendirme mesajları geliştirilebilir.

9. Kaynaklar

- Flask resmi dokümantasyonu: https://flask.palletsprojects.com/
- Bootstrap resmi sitesi: https://getbootstrap.com/
- Python werkzeug.utils secure_filename fonksiyonu dokümantasyonu