NewsHub – Haber & Blog Platformu

NodeJS – MVC – RestApi

Proje Amacı

Bu proje, **Node.js + TypeScript** kullanarak **katmanlı mimari (MVC)** yapısını hem **REST API** hem de **EJS tabanlı arayüzlerle** uygulanacaktır.
JWT tabanlı kimlik doğrulama, session yönetimi, Swagger dokümantasyonu, MongoDB modelleme, validation ve global hata filtreleme konularını kapsayan tam kapsamlı bir backend projesi hedeflenmektedir.

Proje Takvimi

- **Proje Başlangıç Tarihi:** 13 Ekim 2025
- **Teslim Tarihi:** 22 Ekim 2025
- **Proje Süresi:** 10 Gün

Roller

- **Admin:** Tüm kullanıcıları ve içerikleri yönetebilir.
- **User:** Kayıt olabilir, giriş yapabilir, haber/blog yazısı oluşturabilir, güncelleyebilir, silebilir, yorum yapabilir.

Mimari Yapı

```
src/
 ├── config/
 ├── controllers/
 ├── models/
 ├── services/
 ├── routes/
 ├── views/
 │    ├── auth/
 │    ├── posts/
 │    └── partials/
 ├── middlewares/
 ├── utils/
 ├── validations/
 ├── app.ts
 └── server.ts
```

Her klasörün görevleri:

- **controllers/** → HTTP isteklerini karşılar, service katmanını çağırır.
- **services/** → İş mantığı burada yönetilir.
- **models/** → Mongoose modelleri.
- **routes/** → Route tanımlamaları.
- **views/** → EJS tabanlı kullanıcı arayüzü.
- **middlewares/** → Auth, validation, global error handler.
- **utils/** → JWT, bcrypt, yardımcı fonksiyonlar.
- **validations/** → Joi / express-validator şemaları. (Araştırma)

Teknik Gereksinimler

- Node.js + TypeScript
- Express.js
- EJS (View Motoru)
- MongoDB + Mongoose
- bcrypt (Parola Şifreleme)
- **JWT (REST API kimlik doğrulama)**
- **Express-session (EJS oturum yönetimi)**
- Swagger (API dokümantasyonu)
- Joi veya express-validator (Veri doğrulama) (Araştırma)
- Global Error Filter (Merkezi hata yönetimi)

Fonksiyonel Gereksinimler

Authentication & Authorization

- Kullanıcı **register** ve **login** işlemleri yapılmalı.
- Parolalar **bcrypt** ile şifrelenmeli.
- **REST API tarafında JWT** ile kimlik doğrulama sağlanmalı.
- **EJS tarafında session** tabanlı oturum yönetimi olmalı.
- Rol tabanlı erişim:
    - Admin tüm işlemleri yapabilir.
    - User sadece kendi içeriklerini düzenleyebilir/silebilir.

Post Yönetimi

- Yeni post oluşturma, düzenleme, silme ve listeleme işlemleri.
- Her post'un EJS üzerinde detay sayfası olmalı.
- REST API üzerinden aynı işlemler JWT korumalı endpointlerle yapılmalı.

Yorum Yönetimi

- Kullanıcılar postlara yorum ekleyebilmeli.
- Yorumlar sadece admin veya post sahibi tarafından silinebilmeli.

Validasyonlar & Filtreler

- Joi veya express-validator ile tüm input'lar kontrol edilmelidir.
- Global error handler ile tüm hatalar tek noktadan yönetilmelidir.

Swagger Dökümantasyonu

- Tüm REST endpoint'leri Swagger üzerinden belgelenmelidir.
- /api-docs adresinden erişilebilir olmalıdır.

EJS View Sayfaları

Sayfa	Açıklama
/login	Kullanıcı giriş formu
/register	Yeni kullanıcı kaydı
/dashboard	Kullanıcının kendi oluşturduğu postları listeler
/posts/new	Yeni post oluşturma formu
/posts/:id	Post detay sayfası (yorum ekleme alanı ile)
/admin	Admin paneli (tüm kullanıcı ve post listesi)

REST API Endpoint'leri

REST API tarafında, JWT doğrulaması zorunlu olacak ve tüm istekler api/v1/ prefix'i altında çalışacaktır.
Swagger dokümantasyonu /api-docs adresinden erişilebilir olmalıdır.

Endpoint	Metod	Açıklama	Yetki
AUTH			
/api/v1/auth/register	POST	Yeni kullanıcı kaydı oluşturur.	Herkese açık
/api/v1/auth/login	POST	Kullanıcı girişi yapar ve JWT üretir.	Herkese açık
/api/v1/auth/profile	GET	Giriş yapan kullanıcının bilgilerini getirir.	JWT gerekli
/api/v1/auth/refresh	POST	Access token yenileme işlemi.	JWT gerekli
/api/v1/auth/logout	POST	Kullanıcının oturumunu sonlandırır.	JWT gerekli

Teslim Şartları

- Proje **GitHub** üzerinden teslim edilecektir.
- README.md dosyası mutlaka aşağıdakileri içermelidir:
  - Kurulum ve çalışma adımları
  - .env örneği
  - Swagger erişim adresi
- MongoDB üzerinde users, posts, comments koleksiyonları oluşturulmalıdır.
- Katmanlı mimari yapısı eksiksiz uygulanmalıdır.
- Global hata filtreleri ve validasyonlar aktif olmalıdır.

- Commit mesajları anlamlı olmalıdır.

Önemli

- Katmanlı mimariyi tam kapsamlı uygulayabilme
- JWT ve Session farklarını kavrayabilme
- Mongoose ilişkilerini etkin kullanabilme
- Swagger ile API dokümantasyonu hazırlayabilme
- REST API ve MVC yapısını aynı projede entegre edebilme
- Kurumsal düzeyde Node.js + TypeScript proje geliştirme deneyimi kazanmı