**Furkan Kazım Çam**

**21360859036**

**12.Hafta Rapor**

Önceki hafta yazdığımız kodlara ek yaparak devam edeceğiz.

Bu hafta önceki hafta yaptığımız işlemlere try catch ve finally blokları ekleyeceğiz. Peki bunlar nedir?

**Try Catch Finally**

Try, catch ve finally blokları JavaScript'te (ve birçok diğer programlama dilinde) hataların yönetilmesi ve belirli kodların her durumda çalıştırılmasını sağlamak için kullanılır.

**Try Bloğu**

try bloğu, çalıştırmak istediğiniz kodu içerir. Eğer bu blokta bir hata meydana gelirse, kontrol catch bloğuna aktarılır.

**Catch Bloğu**

catch bloğu, try bloğunda meydana gelen hataları yakalar ve bu hatalarla nasıl başa çıkacağınızı belirlemenizi sağlar. Hata yakalandığında, catch bloğundaki kod çalıştırılır.

**Finally Bloğu**

finally bloğu, hata olsun veya olmasın her durumda çalıştırılan kodu içerir. Genellikle kaynakların serbest bırakılması veya temizleme işlemleri için kullanılır. Örneğin, bir veritabanı bağlantısının kapatılması gibi işlemler burada yapılabilir.

Evet kodumuza entegre ederek işleme başlayalım. Geçen haftadan farklı olarak kısayol olması için 2 değişken tanımlayarak onlara değerlerini verelim. metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Try bloğunda verilen bilgiler veri tabanına eklenir eğer eklenemez ise catch bloğundaki error mesajı yazdırılır ve her halükârda finally bloğuna girip bağlantı kapatılır ve console.log ile kullanıcı bilgilendirilir. Çıktımız aşağıdaki gibi olmalıdır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bunu diğer kod bloklarımıza da uygulayalım.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Kullanıcılar isminde bir array tanımladık ve onu istertMany ile veri tabanına gönderdik.

Çıktı olarak ise kaç dosya yüklendiği ve veri tabanı bağlantısının kapatıldığı görülmektedir.

metin, ekran görüntüsü, yazı tipi içeren bir resim

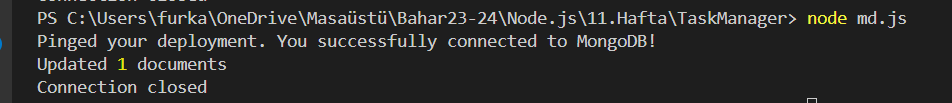
Açıklama otomatik olarak oluşturuldu

Sırada ise var olan bir veriyi güncellemenin nasıl yapıldığı var.

ekran görüntüsü, metin, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

updateOne komutu var olan veriyi set parametresi ile günceller.



metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Veri tabanındaki kullanıcılar arasında arama işlemi ile devam edelim.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

findOne komutu ile veriyi bulur ve istediğimiz işlemi yapabiliriz.

metin, yazılım, multimedya yazılımı, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Verinin tüm bilgileri yazdırılmıştır.

metin, ekran görüntüsü, yazı tipi, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Burda ise kullanıcıyı bulup bilgilerini Json formatında konsola yazdırılması istenmiştir. Eğer bulamaz ise error mesajı ile kullanıcı bilgilendirilecektir.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

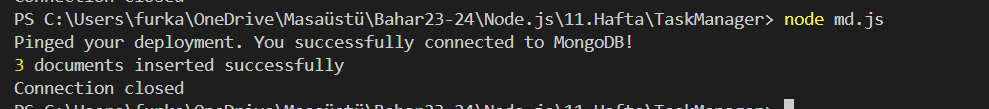
Kullanıcı bilgileri yazdırıldı.

Tasks adlı veri tabanına da veri gönderelim.

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Tasks dizisi tanımlanır ve insertMany ile bu veriler tasks adlı veri tabanına gönderilir.



metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Veri tabanındaki tüm bilgileri yazdırmak için ise alttaki kodu kullanabilirsiniz.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu