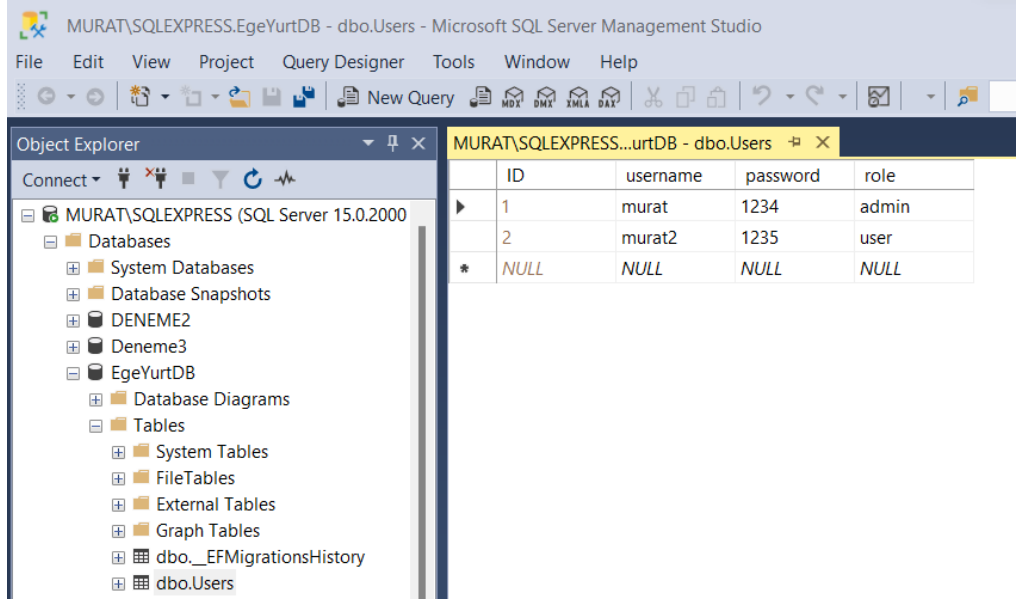


EGE YURT GRUP MÜLAKAT PROJESİ

1. PROJE AÇIKLAMASI

- Database de Kullanıcı tablosu oluşturmak için öncelikle Proje içerisinde Models klasörü adı altında bir User class'ı oluşturuldu. Bu class içinde ID, username, password ve role bilgilerini barındırıyor.
- Model class'ını oluşturduktan sonra DB'de tablo oluşturmak için code first yaklaşımı ile tablo oluşturuldu. Migration yardımıyla package manager console'da add-migration EgeYurtDB ve sonrasında update database komutlarının çalışmasıyla veri tabanında oluşturduğumuz modele karşılık gelen EgeYurtDB adında bir table oluşturuldu. Örnek olması açısından bir admin ve user rolünde 2 adet örnek data kaydedildi. Aşağıdaki resimde bu tabloyu ve örnek verileri görebilirsiniz.



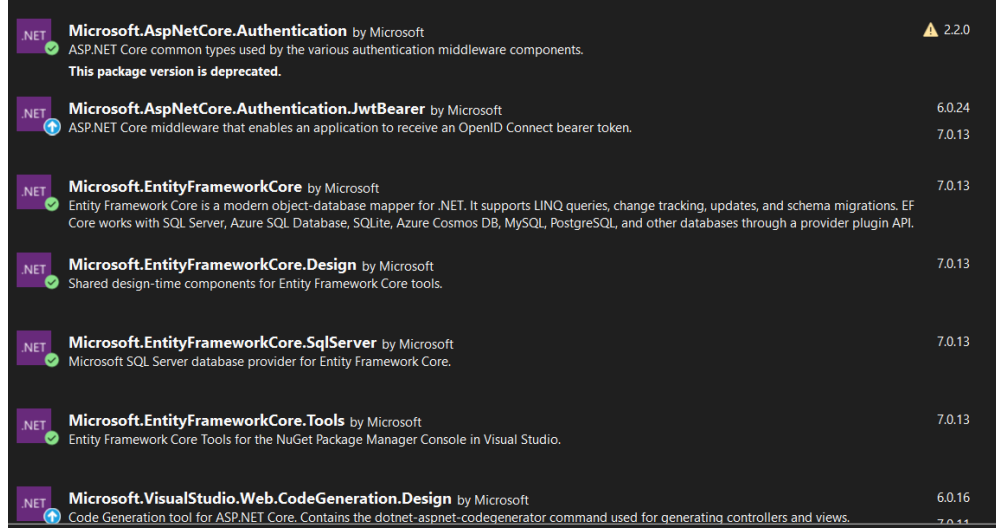
The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for MURAT\SQLEXPRESS (SQL Server 15.0.2000). The database EgeYurtDB is expanded, showing a table named dbo.Users. The table structure is displayed in the right pane, showing columns ID, username, password, and role. The table contains two rows of data: one for an admin user (ID 1, username murat, password 1234) and one for a regular user (ID 2, username murat2, password 1235). A third row is shown with NULL values, indicating a new record to be added.

ID	username	password	role
1	murat	1234	admin
2	murat2	1235	user
* NULL	NULL	NULL	NULL

- Yukarıdaki adımlardan sonra kullanıcı ekleme, silme, güncelleme ve silme(CRUD Operations) işlemleri için Controller klasörü altında UserController oluşturuldu. Çalışması Postman ve Swagger üzerinden test edildi.
- Authorization ve authentication işlemleri için proje dosyasının altında JwtAuthenticationManager adlı class oluşturuldu. Bu işlemler de Postman ve Swagger üzerinden test edildi.

2. PROJENİN ÇALIŞTIRILMASI

- Projeyi bilgisayarınıza indirdikten sonra NuGet Package Manager aracılığıyla aşağıdaki resimde gösterilen paketlerin indirilmesi gerekmektedir.

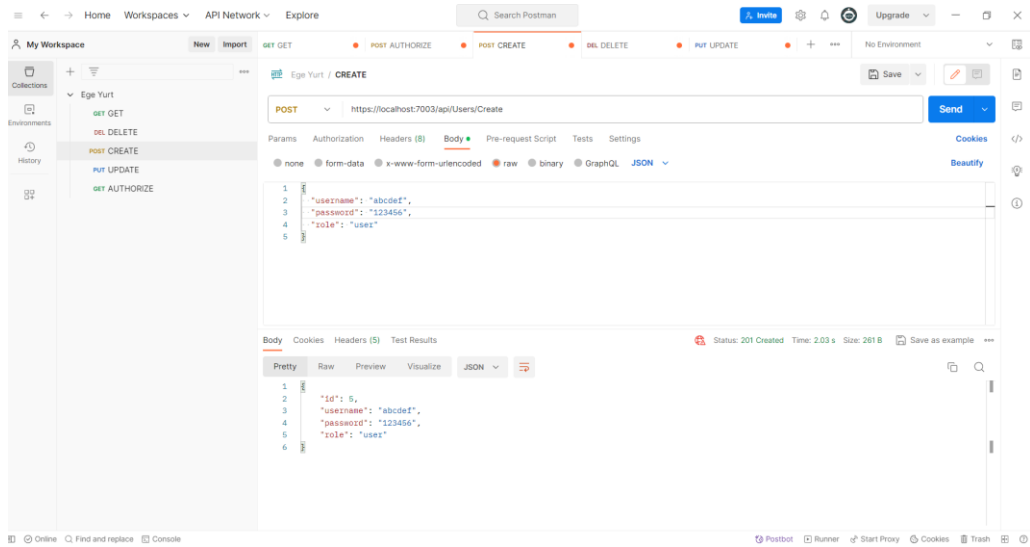


- Paketleri indirdikten sonra kendi database bağlantınızı app.json dosyasının içindeki ConnectionString kısmına kopyaladıktan sonra proje çalıştırılmak için hazır hale gelecektir.

3. PROJENİN TEST EDİLMESİ

- CRUD işlemlerinin test edilmesi. CRUD işlemleri için oluşturulan end pointler kullanılarak Postman üzerinde tek tek test edildi. Aşağıdaki resimlerde her test edilen işlemin sonuçları gösterilmiştir.

Ekleme işlemi



Silme işlemi

The screenshot shows the Postman interface with a DELETE request selected. The URL is `https://localhost:7003/api/Users/delete/4`. The request is sent, and the response is a 204 No Content status. The response body is empty.

Güncelleme işlemi

The screenshot shows the Postman interface with a PUT request selected. The URL is `https://localhost:7003/api/Users/3`. The request body is a JSON object: `{ "id": 3, "username": "update3", "password": "123456", "role": "user" }`. The request is sent, and the response is a 204 No Content status. The response body is empty.

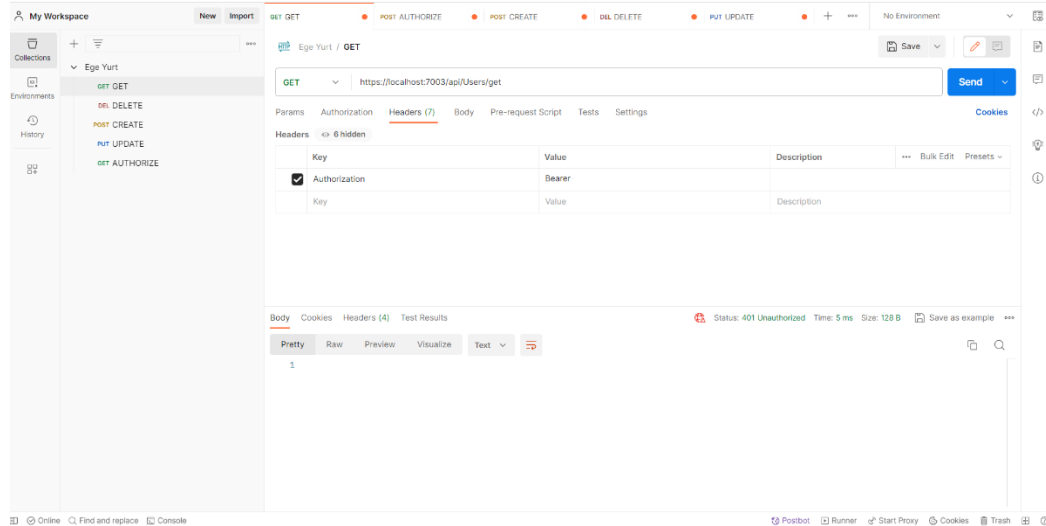
Kullanıcı Listesi Getirme

The screenshot shows the Postman interface with a GET request selected. The URL is `https://localhost:7003/api/Users/get`. The request is sent, and the response is a 200 OK status. The response body is a JSON array of user objects:

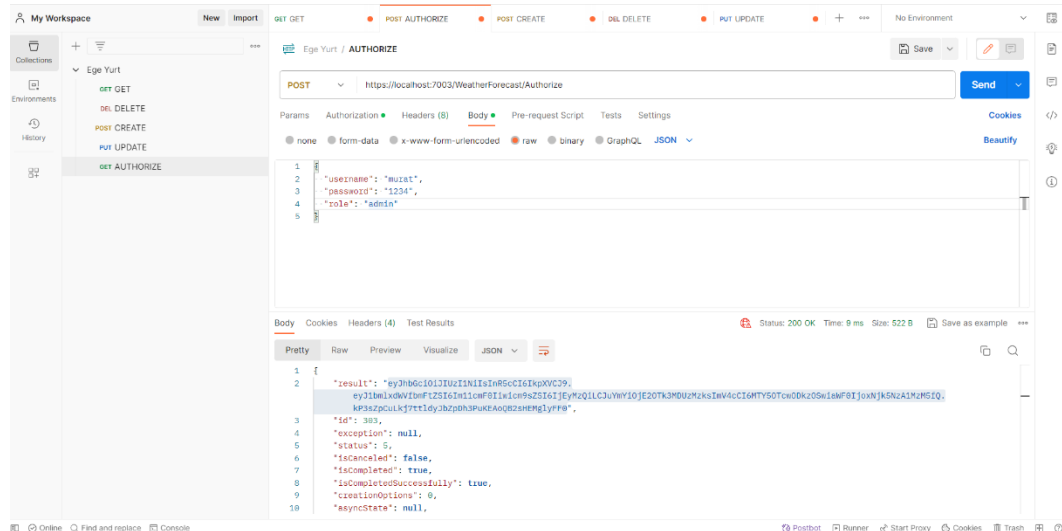
```
1 {
2   {
3     "id": 1,
4     "username": "muizet",
5     "password": "1234",
6     "role": "admin"
7   },
8   {
9     "id": 2,
10    "username": "muizet2",
11    "password": "1235",
12    "role": "user"
13  }
14 }
```

Authorization

Kullanıcı listesini gösterme tarafında bir authorization işlemi yapıldı. Kullanıcı giriş yapmadığında 401 hatası alıyor. Aşağıdaki resimde bu test işleminin sonucunu görebilirsiniz.



Authorization işlemi için jwt(json web token) kullanıldı. Bu işlem için bir bearer web token üretiliyor. Aşağıdaki resimde üretilen web token gösterilmektedir.



The screenshot shows the Postman REST client interface. At the top, there's a workspace named 'My Workspace' with tabs for 'New' and 'Import'. The main area displays a REST client request for 'Ege Yurt / GET' with the URL 'https://localhost:7003/api/Users/getUsers'. The request is configured with a 'GET' method and a 'Bearer Token' authorization header. A warning message states: 'Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables'. The response is displayed in the bottom panel, showing a JSON array of two user objects. The first user has an id of 1, username 'mutat', password '1234', and role 'admin'. The second user has an id of 2, username 'mutat2', password '1235', and role 'user'.

```

1  [
2  {
3    "id": 1,
4    "username": "mutat",
5    "password": "1234",
6    "role": "admin"
7  },
8  {
9    "id": 2,
10   "username": "mutat2",
11   "password": "1235",
12   "role": "user"
13 }
14 ]

```