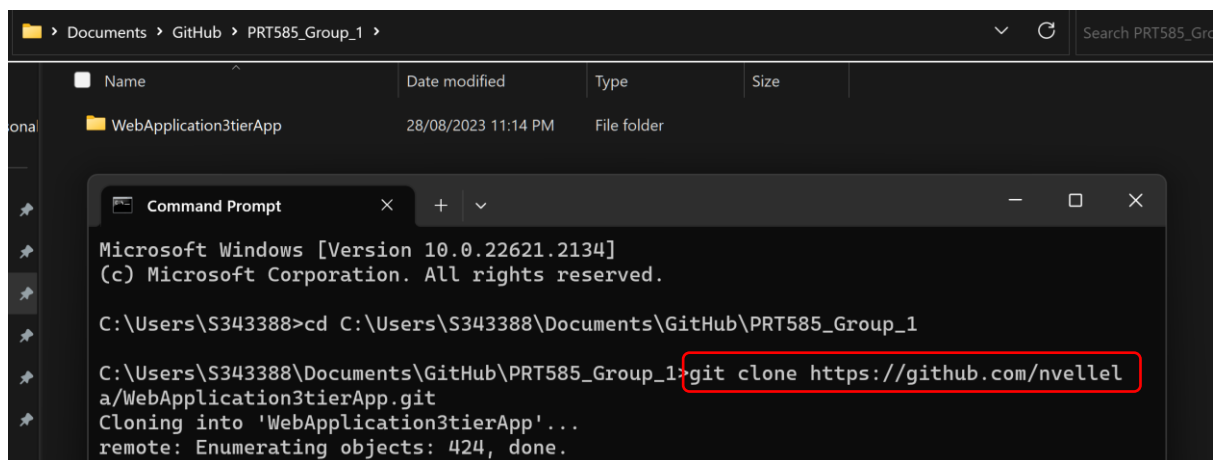
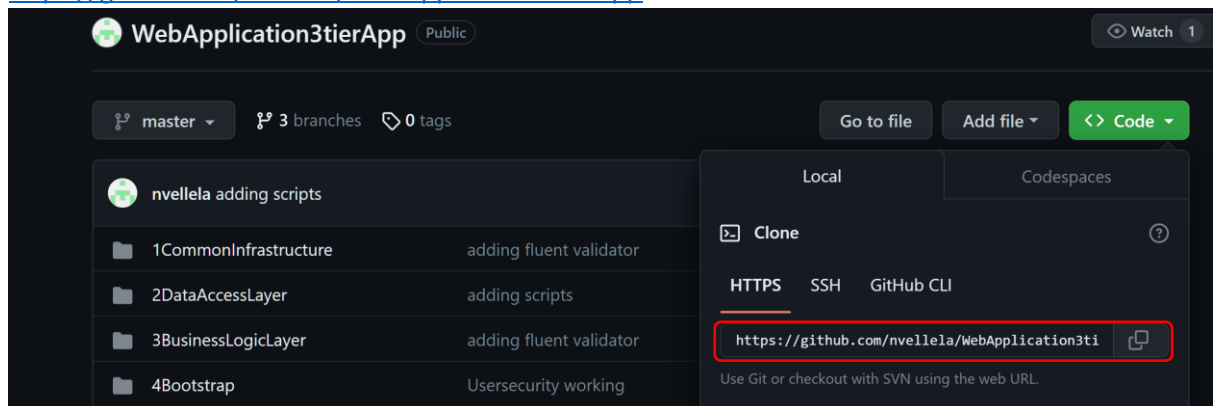


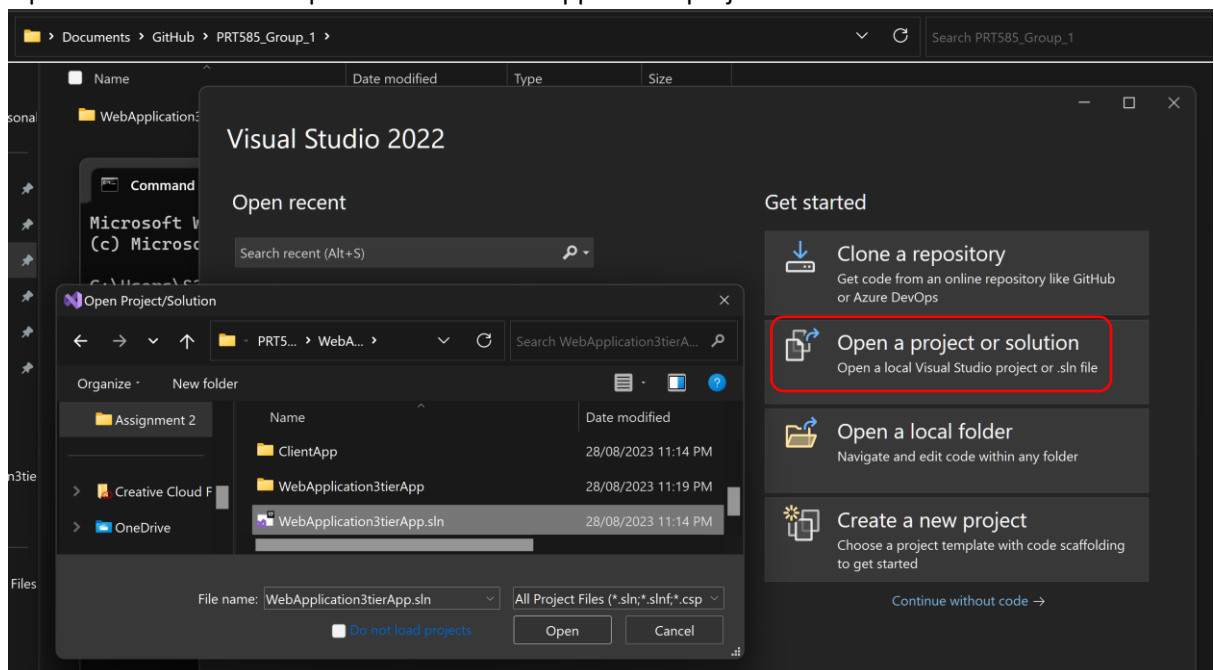
## Run 3-Tier Web Application

1. Clone the repository.

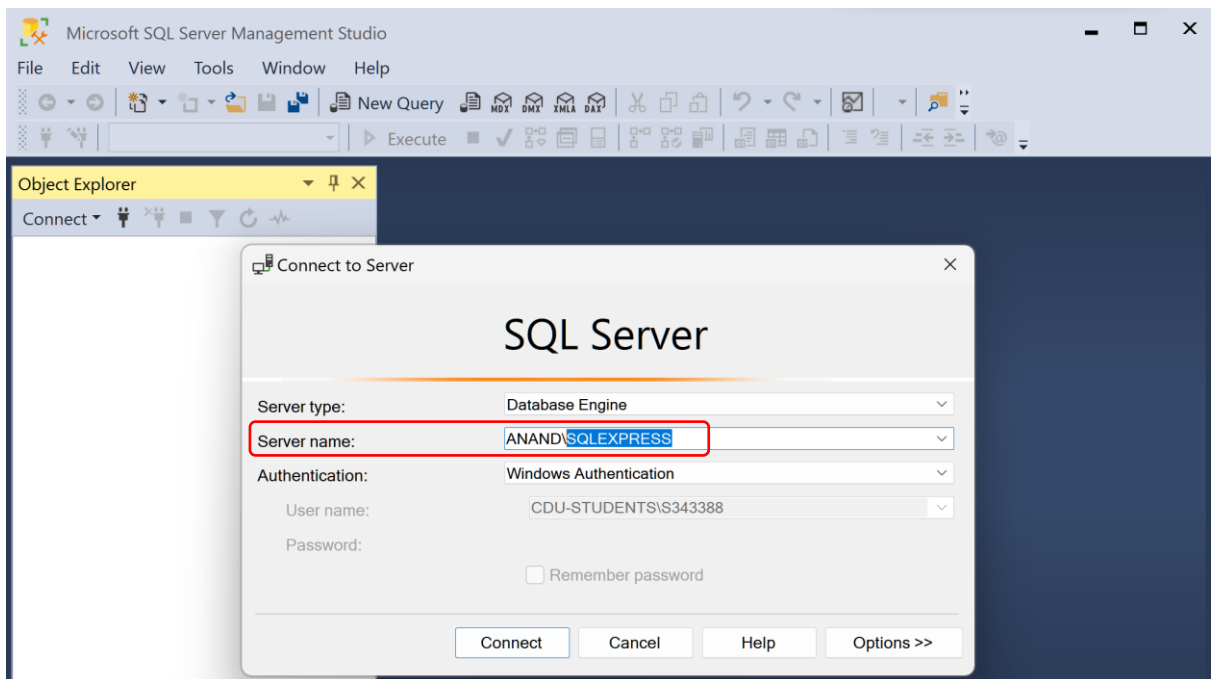
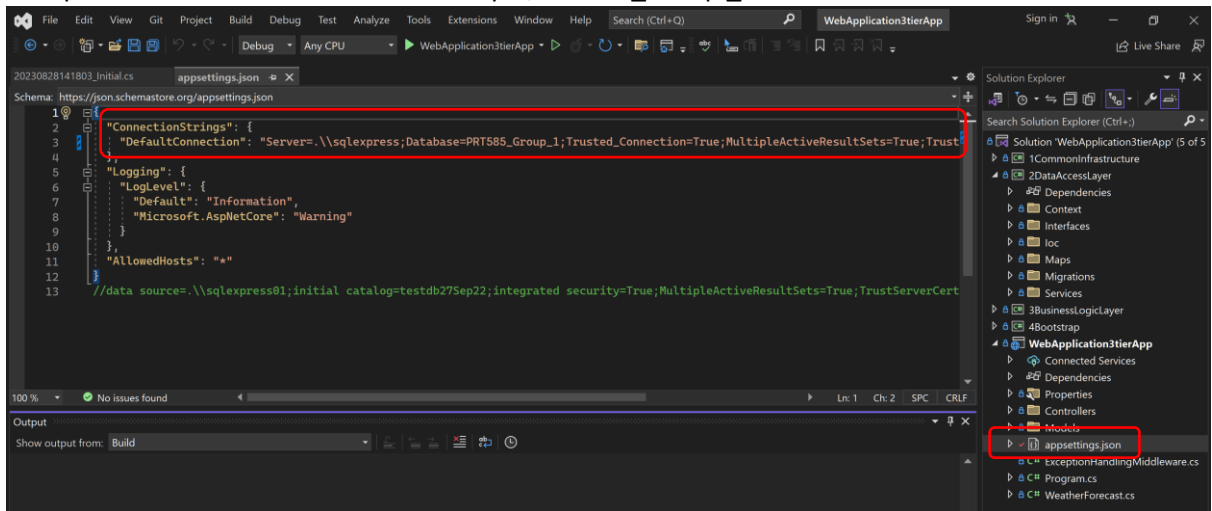
<https://github.com/nvlella/WebApplication3tierApp>



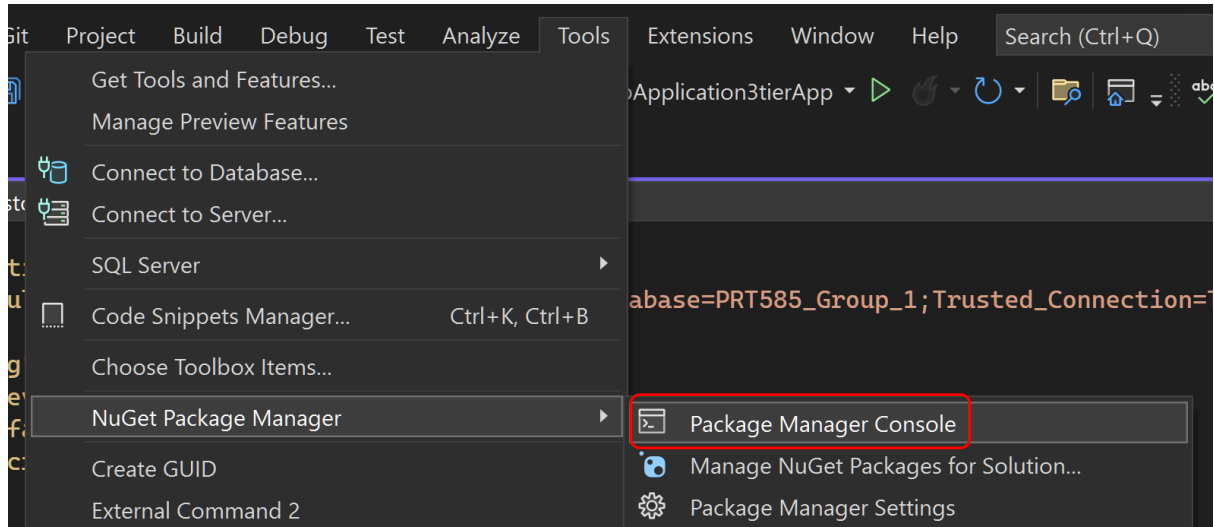
2. Open Visual Studio and open the 3-Tier Web Application project.



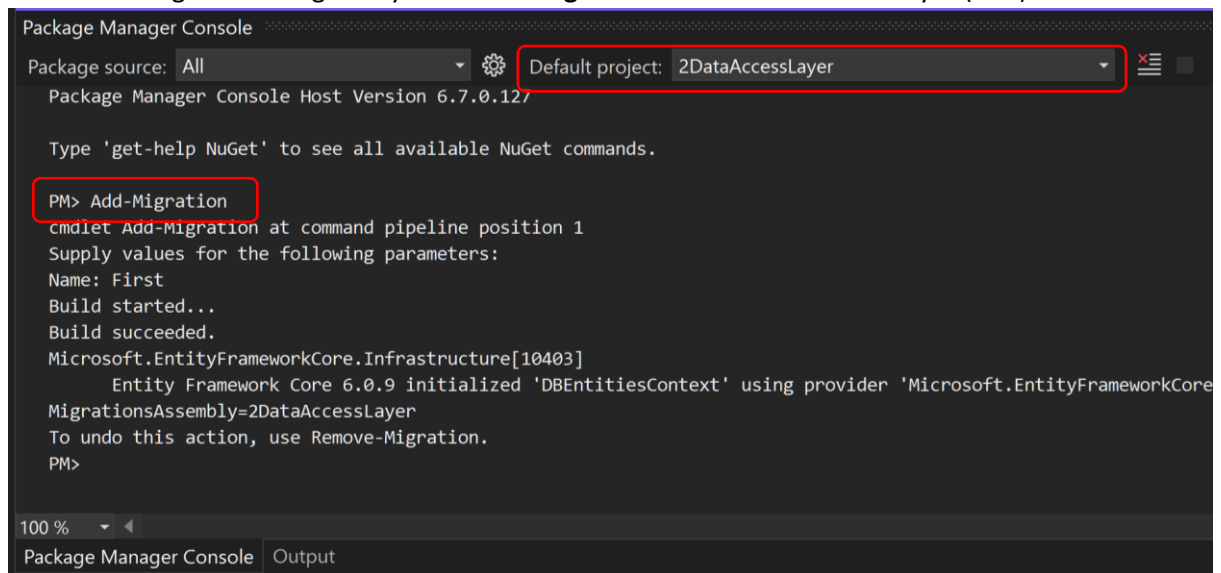
3. Open “**appsettings.json**” and update the connection string.
- The server’s name in the connection string should match the SQL server name.
  - Update the database name. For example, PRT585\_Group\_1



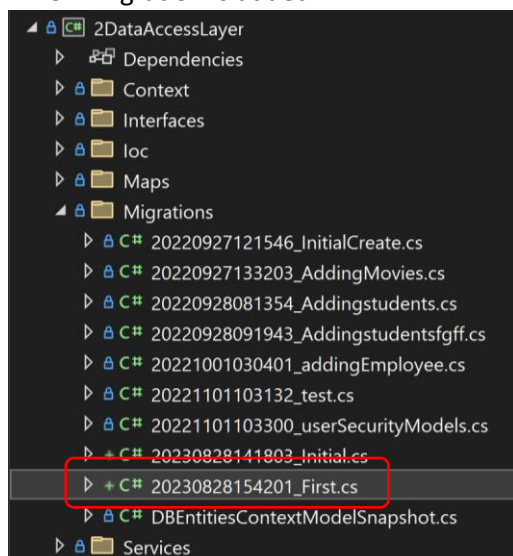
4. Go to Tools > NuGet Package Manager > **Package Manager Console**



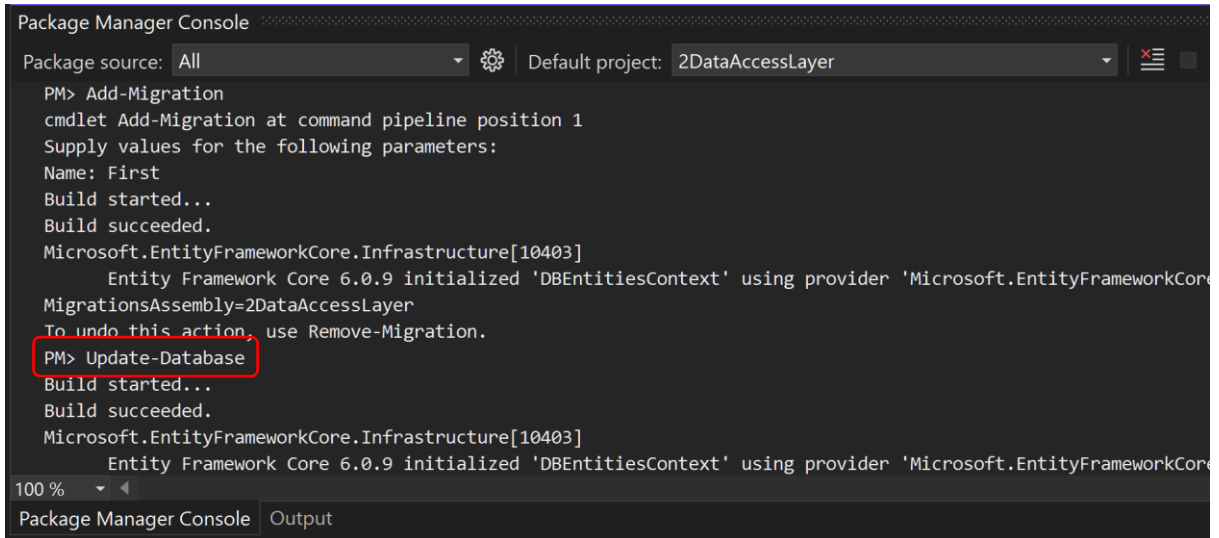
5. Add a new migration using the syntax “**Add-Migration**” in the Data Access Layer (DAL).



A new migration is added.



6. Use the syntax “**Update-Database**” to update the database in the SQL server.

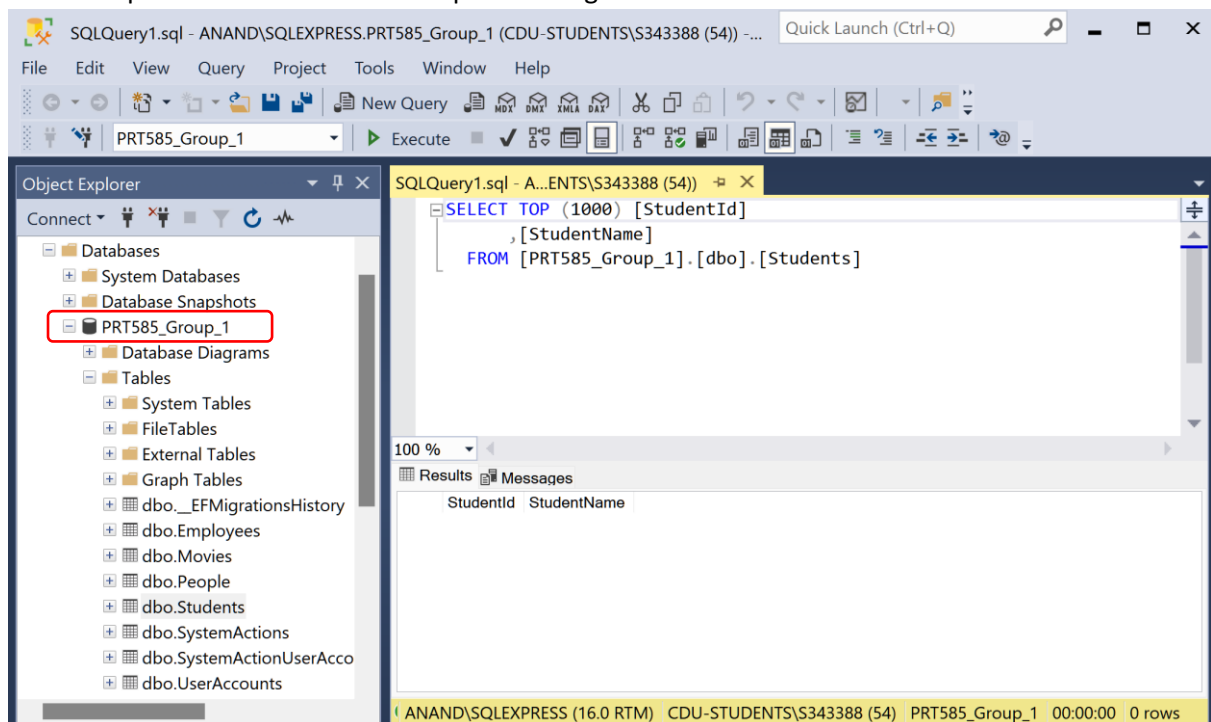


```
Package Manager Console
Package source: All Default project: 2DataAccessLayer

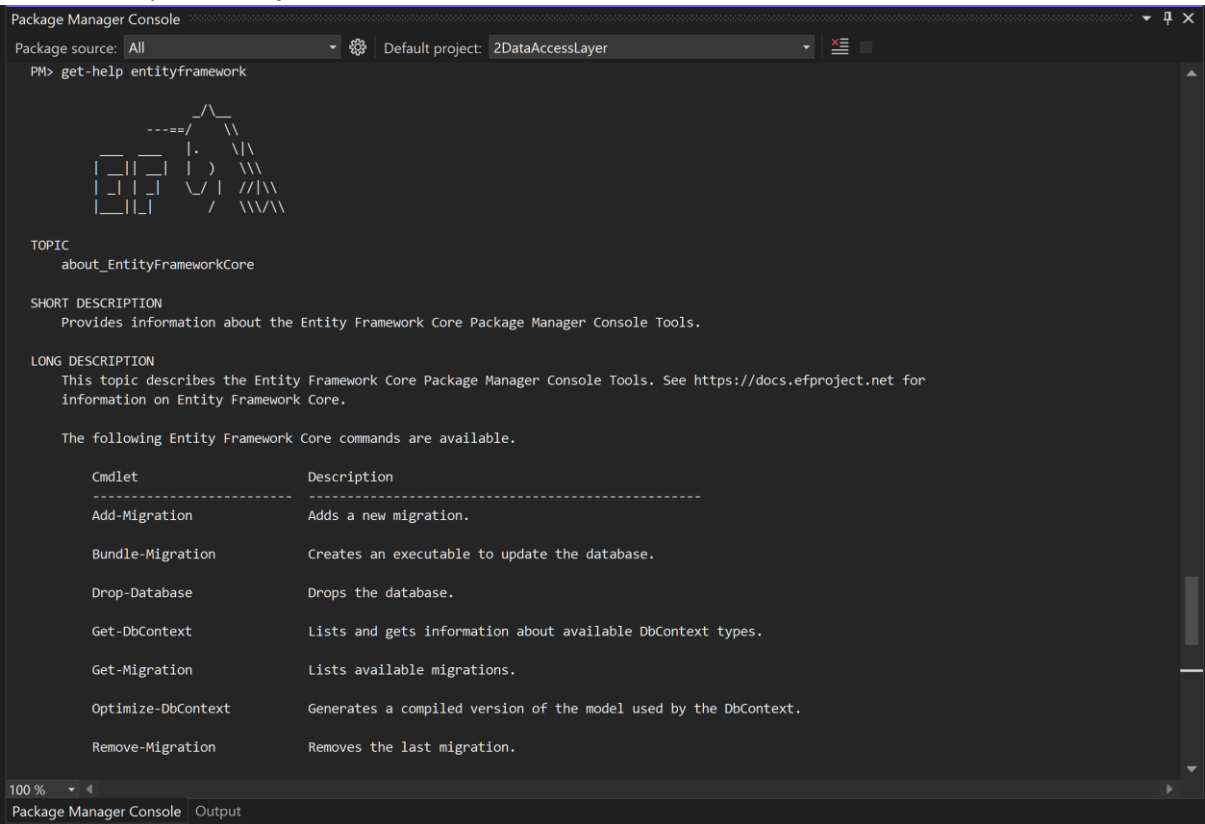
PM> Add-Migration
cmdlet Add-Migration at command pipeline position 1
Supply values for the following parameters:
Name: First
Build started...
Build succeeded.
Microsoft.EntityFrameworkCore.Infrastructure[10403]
Entity Framework Core 6.0.9 initialized 'DBEntitiesContext' using provider 'Microsoft.EntityFrameworkCore.SqlServer'
MigrationsAssembly=2DataAccessLayer
To undo this action, use Remove-Migration.
PM> Update-Database
Build started...
Build succeeded.
Microsoft.EntityFrameworkCore.Infrastructure[10403]
Entity Framework Core 6.0.9 initialized 'DBEntitiesContext' using provider 'Microsoft.EntityFrameworkCore.SqlServer'

100 %
Package Manager Console Output
```

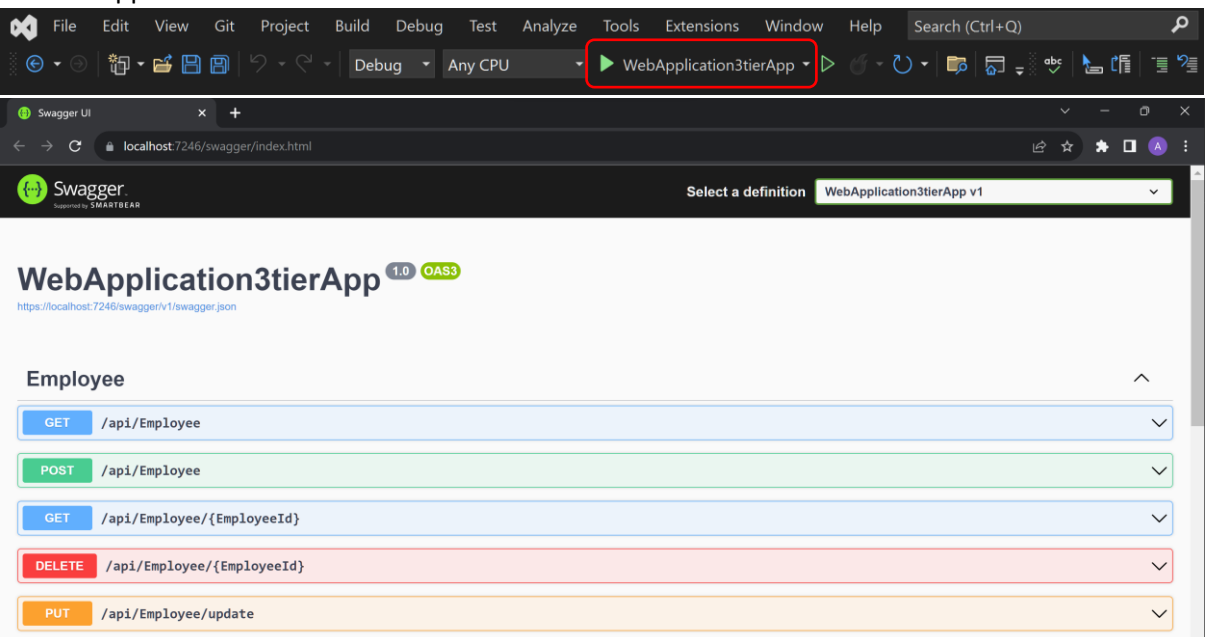
This will update the database to the specified migration.



- Use the syntax **“Remove-Migration”** to delete the last migration.
- Use the syntax **“Drop-Database”** to delete the database.



8. Run the app.



9. Make a request.

- Expand the **POST Student** endpoint.
- Click **Try it out**.

The image shows the Swagger UI for the 'Student' endpoint. The 'POST /api/Student' endpoint is selected. The 'Parameters' tab is active, showing 'No parameters'. A red box highlights the 'Try it out' button in the top right corner of the parameters section.

- Change the example value (**studentName**) in the Request Body field.
- Click **Execute**.

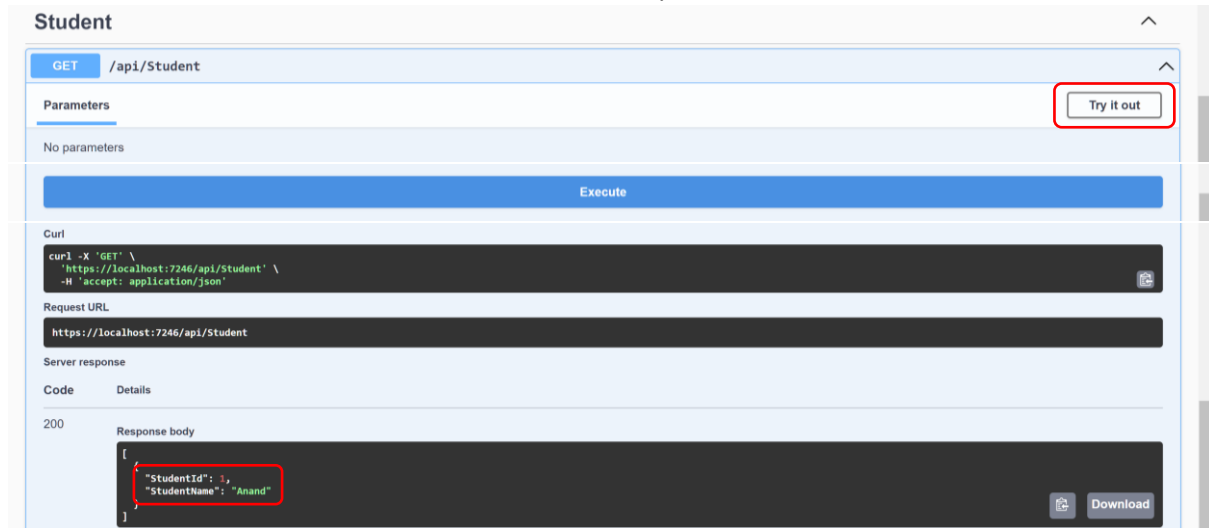
The image shows the Swagger UI for the 'POST /api/Student' endpoint. The 'Request body' tab is active, showing a JSON object: `{ "studentId": 0, "studentName": "Anand" }`. A red box highlights the 'studentName' field. The 'Execute' button is highlighted at the bottom.

Swagger UI submits the request and shows the response.

The image shows the Swagger UI displaying the response for the POST request. The 'Curl' section shows the command: `curl -X 'POST' \ 'https://localhost:7246/api/Student' \ -H 'accept: application/json' \ -H 'Content-Type: application/json-patch+json' \ -d '{ "studentId": 0, "studentName": "Anand" }'`. The 'Request URL' is `https://localhost:7246/api/Student`. The 'Server response' section shows a 200 status code. The 'Response body' is empty. The 'Response headers' are: `content-length: 1`, `content-type: application/json; charset=utf-8`, `date: Mon, 28 Aug 2023 16:35:33 GMT`, and `server: Kestrel`. The 'Responses' section shows a 200 status code with the description 'Success'.

10. Verify that the student was created.

- Expand the **GET Student** endpoint.
- Click **Try it out**.
- Click **Execute**.
- The new student's name will be returned in the Response section.



Check the database to see the new student.

