VacationModule

Dependencies for running the application on localhost:

- AutoFixture
- AutoMapper
- AutoMapper.Extensions.Microsoft.DependencyInjection
- coverlet.collector
- EntityFramework
- Microsoft.AspNetCore.Authentication.Abstractions
- Microsoft.AspNetCore.Authentication.JwtBearer
- Microsoft.Bcl.Async
- Microsoft.Bcl.AsyncInterfaces
- Microsoft.EntityFrameworkCore
- Microsoft.EntityFrameworkCore.Abstractions
- Microsoft.EntityFrameworkCore.Design
- Microsoft.EntityFrameworkCore.Relational
- Microsoft.EntityFrameworkCore.SqlServer
- Microsoft.EntityFrameworkCore.Tools
- Microsoft.Extensions.Configuration.Abstractions
- Microsoft.IdentityModel.Tokens
- Microsoft.NET.Test.Sdk
- Microsoft.VisualStudio.Azure.Containers.Tools.Targets
- Moq
- MSTest.TestAdapter
- MSTest.TestFramework
- Npgsql.EntityFrameworkCore.PostgreSQL
- Npgsql.EntityFrameworkCore.PostgreSQL.Design
- NSwag.SwaggerGeneration
- Swashbuckle.AspNetCore.Filters
- Swashbuckle.AspNetCore.Swagger
- Swashbuckle.AspNetCore.SwaggerGen
- Swashbuckle.AspNetCore.SwaggerUI
- System.IdentityModel.Tokens.Jwt

The application is implemented respecting Repository Pattern and Dependency Injection. Also, DTOs are used for interactions with controllers.

For running the application on localhost, I use a local database in PostgresSQL, whose credentials for connecting are found in “appsettings.json” file. EntityFramework is used for establishing context with database, and code first migrations were performed, in order to create the 2 tables: Users and VacationRequests.

The application is role-based and implements Json Web Token (JWT) authentication. There are 2 roles that can perform actions: ”employee” and “admin” (case-sensitive, all lowercase). After login, a jwt is provided and the user must authenticate in Swagger, introducing the following: “Bearer + <jwt provided>”. After that, the user can perform actions according to his role. In order to debug a JWT and see claims inside it, use https://jwt.io/.

An admin can access each endpoint an employee can, plus he can see all vacation requests made by all employees (allRequests endpoint).

Endpoints:

1. /api/Auth/register -> endpoint for registering
2. /api/Auth/login -> endpoint for login
3. /api/Holiday/getHolidays -> returns a list with all national and orthodox holidays from the current year, with some information such as date, name, type.
4. /api/Holiday/holidayList -> returns a list with just the dates of all holidays
5. /api/Holiday/requestHoliday -> endpoint for creating a new vacation request. It excludes from the initial introduced dates those which are holidays and those which are weekends, so the request that is saved in database contains only working days.
6. /api/Holiday/myVacationRequests -> endpoint for getting a list of vacation requests made by user
7. /api/Holiday/modifyRequest -> endpoint for modifying an existing holiday request, by providing the “id” of the request and the new dates. It excludes from the initial introduced dates those which are holidays and those which are weekends, so the request that is modified in database contains only working days.
8. /api/Holiday/allRequests -> returns all requests made by all users, accessible only by admin users.
9. /api/Holiday/myAvailableDays -> the number of available vacation days for a given year: for the current year returns the number of available days left from previous year plus number of available days from current year. For other years, it returns the number of available days per year (25) minus number of requested days from that year.

In “ApiTest” class library are some unit tests for AuthController and HolidayController to verify their responses’ codes.

<u>VacationModule on Azure</u>

The database is hosted online, on Microsoft Azure and fully functional. From price reasons, because I am on a free plan, I do not keep it open 24/7 and only start it when I need it.

Moreover, the application is hosted online, on Microsoft Azure too. Also from price considerents, I do not keep it open 24/7 and start the server when I need it (this is the link to it but you will find it stopped unless I start it: vacationmodulewebapi20230409041421.azurewebsites.net).
The hosted application is fully functional together with the hosted database.