Contents

1 General information

“REST API JWT Auth 1.0” was created in Visual Studio Community by Annice Strömberg, 2020, with Annice.se as the primary download location. The script is a small integration system that enables data exchange between two independent ASP.NET Core web applications using REST APIs with JSON web token (JWT).

2 License

This script is of general public license (GPL), which means that it is free of charge and can be modified to suit your needs. However, in case of redistribution the script must be made available under the same license terms.

3 System description

“REST API JWT Auth 1.0” is built in CSS3, HTML5, JavaScript, C# with ASP.NET Core 3.1, and Transact SQL using SQL Server as a database management system (DBMS).

Furthermore, the application 1 – which is the only app having a GUI – is built according the model-view-controller (MVC) pattern. In turn, the application 2 is built with a model, controller and database layer.

4 System requirements

The script can be run on servers that support C# 8.0 with ASP.NET Core 3.1, e.g. on Azure or on your local computer with the .NET Core 3.1 platform installed, along with an SQL Server supported database.

However, I will not go into any details of how you can deploy the applications to cloud servers such as Azure or similar as this script was implemented and tested locally. Nevertheless, I will go through the necessary steps to run and test this script on-premises.

5 Supported features

The following functions and features are supported by this script:

- Login system based on sessions.
- User password encryption (HMAC-SHA256) based on ASP.NET Core Identity.
- Protection against SQL injections.
- Protection against cross-site forgery.
- Write and edit permission of user details for the admin user.
- Database storage of user details.
- Responsive design.
- Client and server side validation.

6 Context flow

This section illustrates the context flow of the data exchange between application 1 and application 2 when a user logs in to be able to update user details.
(Hold CTRL and press the “+” key to zoom in.)

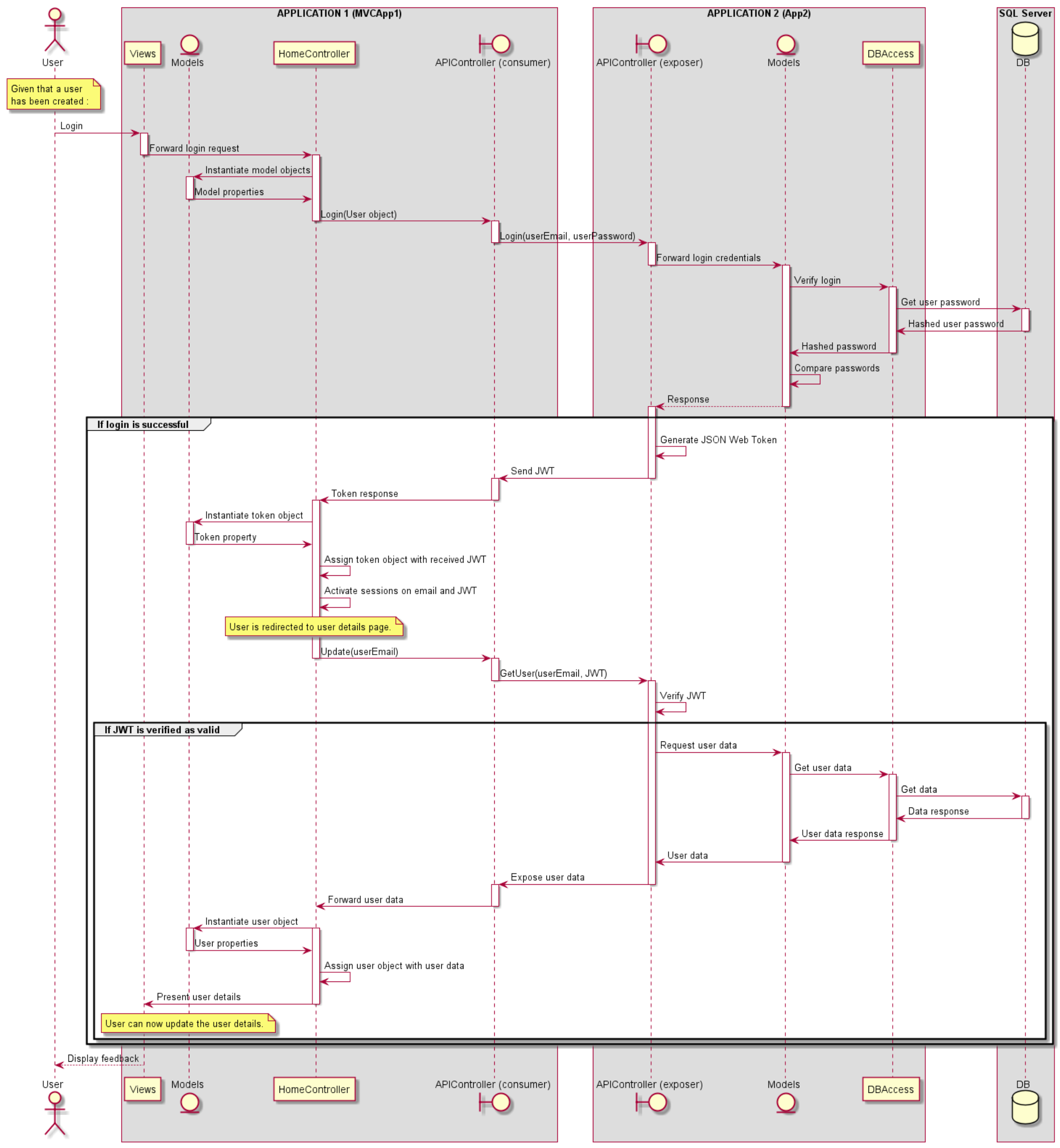
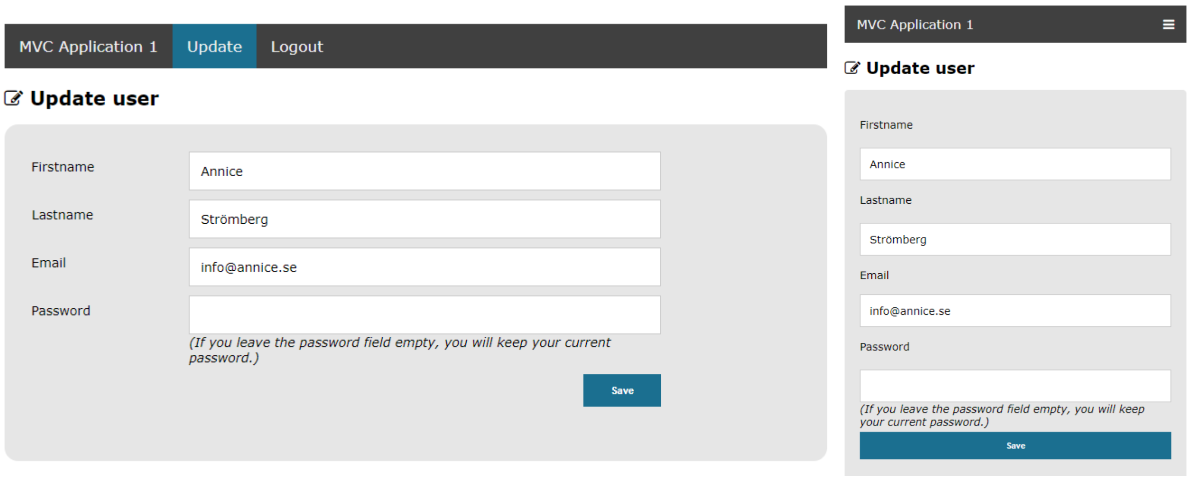


Image 1: *Sequence flow to illustrate the data exchange between MVCApp1 and App2 using JSON web token.*

7 Graphical user interface

In this section you can see a couple of screenshots of how the GUI looks like in application 1 – both in desktop view and mobile device view (responsive view).



Update user

Firstname: Annice

Lastname: Strömberg

Email: info@annice.se

Password:

(If you leave the password field empty, you will keep your current password.)

Image 2: *The update page in desktop view vs. responsive view.*

8 Setup guide

As this script was created in Visual Studio with SQL Server, I will go through the necessary setup steps accordingly (all softwares used for this application setup are free).

8.1 Prerequisites

- Install SQL Server Express
- Install SQL Server Management Studio (SSMS)
- Install .NET Core 3.1 (SDK)
- Install Visual Studio Community

8.2 Create a database and its table

1. Create a database in SQL Server.
2. Create a user login in SQL Server.
3. Launch SQL Server Management Studio.
4. Navigate to the unzipped script folder path: *REST API JWT Auth 1.0 > SQL*. In the SQL folder, open the file "sql_application_2.sql" in SSMS and change the highlighted values below to match your database name and user details (Note! The default password is set to "admin", but can be changed after your first login):

```
USE YourDatabaseName
GO

CREATE TABLE a_user
  (
     id        INT PRIMARY KEY IDENTITY(1, 1),
     firstname VARCHAR(256),
     lastname  VARCHAR(256),
     email     VARCHAR(256) NOT NULL UNIQUE,
     password  VARCHAR(500) NOT NULL
  );

INSERT a_user
VALUES('YourFirstName', -- Optional
       'YourLastName',  -- Optional
       'your@email.com',
'AQAAAAEAACcQAAAAEBehHmgEHZmjXlTBGlKSW9KVuxMIHp1f4r8sC502SFQkGGxiYeef6HFntN
MCMdZ76w==');
```

5. Execute the SQL script in SQL Server to create the App2 database table named "a_user" with a default user.

8.3 Configure the applications

6. Launch Visual Studio on your computer, e.g. via Windows start and then browse for Visual Studio.
7. In Visual Studio, select to open application 1 via the unzipped script folder path: *REST API JWT Auth 1.0 > Application 1 > MVCApp1.sln*
8. When you have opened the MVCApp1 solution in Visual Studio, you can choose to change the API client name in the "appsettings.json" file:

```
{
  "APIClients": {
    "Application2": "App2", // Name the client we want to talk to from
application 1.
     "App2BaseURL": "https://localhost:44304/" // Set the base URL to the
client we want to call.
  }
}
```

9. Save (CTRL+S) the "appsettings.json" file if it is changed.
10. Keep the MVCApp1 solution open in Visual Studio, and then select to launch another instance of Visual Studio (repeat step 6).
11. In the second instance of Visual Studio, select to open application 2 via the unzipped script folder path:

    REST API JWT Auth 1.0 > Application 2 > App2.sln

12. Once the App2 solution is open in Visual Studio, select to open its "appsettings.json" file and change the highlighted values below to suit your own credentials:

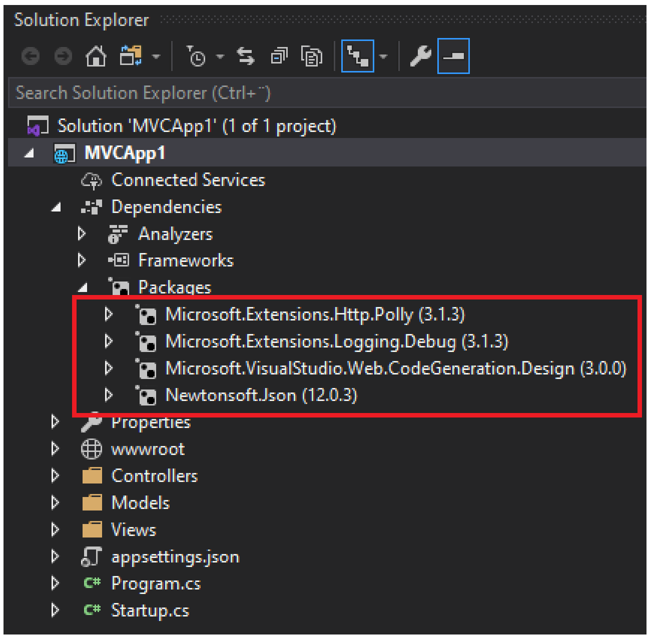
```
{
  "DBSettings":
  {
    "ConnectionString": "Data Source=.\\SQLEXPRESS;initial
catalog=YourDatabaseName;user
id=YourDatabaseUser;password=YourDatabasePassword;MultipleActiveResultSets=
True",
    "Table": "a_user"
  },

  "APIClients":
  {
    "Application1": "MVCApp1", // Name the client we want to talk to from
application 2.
      "App1BaseURL": "https://localhost:44370/" // Set the base URL to the
client we want to call.
  },

  "Jwt":
  {
    "Key": "abc123^&%&^&%321", // Name the key with a minimum length of 16
characters.
    "Issuer": "Application2.com",
    "Audience": "Application1.com"
  }
}
```

13. Save the "appsettings.json" file.

8.4 Install NuGet packages

14. Make sure you have the following NuGet packages installed for your solutions, otherwise install them.

- MVCApp1:

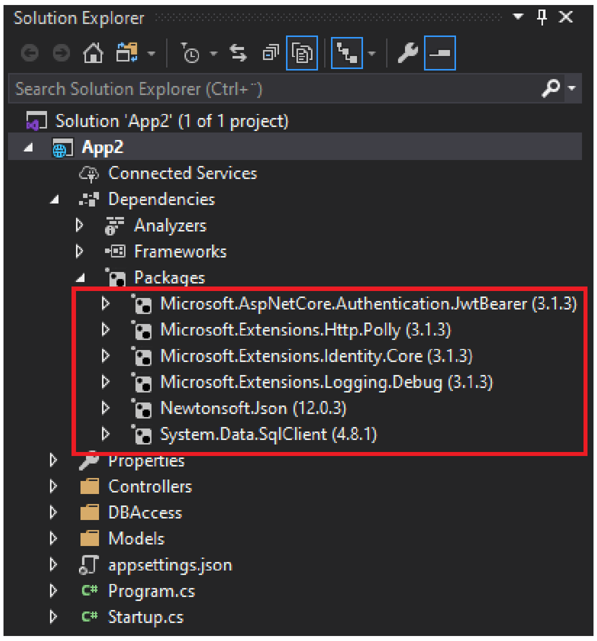


Solution Explorer

Search Solution Explorer (Ctrl+¨)

Solution 'MVCApp1' (1 of 1 project)
MVCApp1
Connected Services
Dependencies
Analyzers
Frameworks
Packages
Microsoft.Extensions.Http.Polly (3.1.3)
Microsoft.Extensions.Logging.Debug (3.1.3)
Microsoft.VisualStudio.Web.CodeGeneration.Design (3.0.0)
Newtonsoft.Json (12.0.3)
Properties
wwwroot
Controllers
Models
Views
appsettings.json
Program.cs
Startup.cs

- App2:



Solution Explorer

Search Solution Explorer (Ctrl+¨)

Solution 'App2' (1 of 1 project)
App2
Connected Services
Dependencies
Analyzers
Frameworks
Packages
Microsoft.AspNetCore.Authentication.JwtBearer (3.1.3)
Microsoft.Extensions.Http.Polly (3.1.3)
Microsoft.Extensions.Identity.Core (3.1.3)
Microsoft.Extensions.Logging.Debug (3.1.3)
Newtonsoft.Json (12.0.3)
System.Data.SqlClient (4.8.1)
Properties
Controllers
DBAccess
Models
appsettings.json
Program.cs
Startup.cs

8.5 Run and test the applications

15. Make sure you have two instances open of Visual Studio – one for the MVCApp1 solution, and the other one for the App2 solution.
16. Select to run both applications – each from its own instance.
17. On your first login, use the password "**admin**" along with the user email you specified when you executed the SQL code (see step 4).

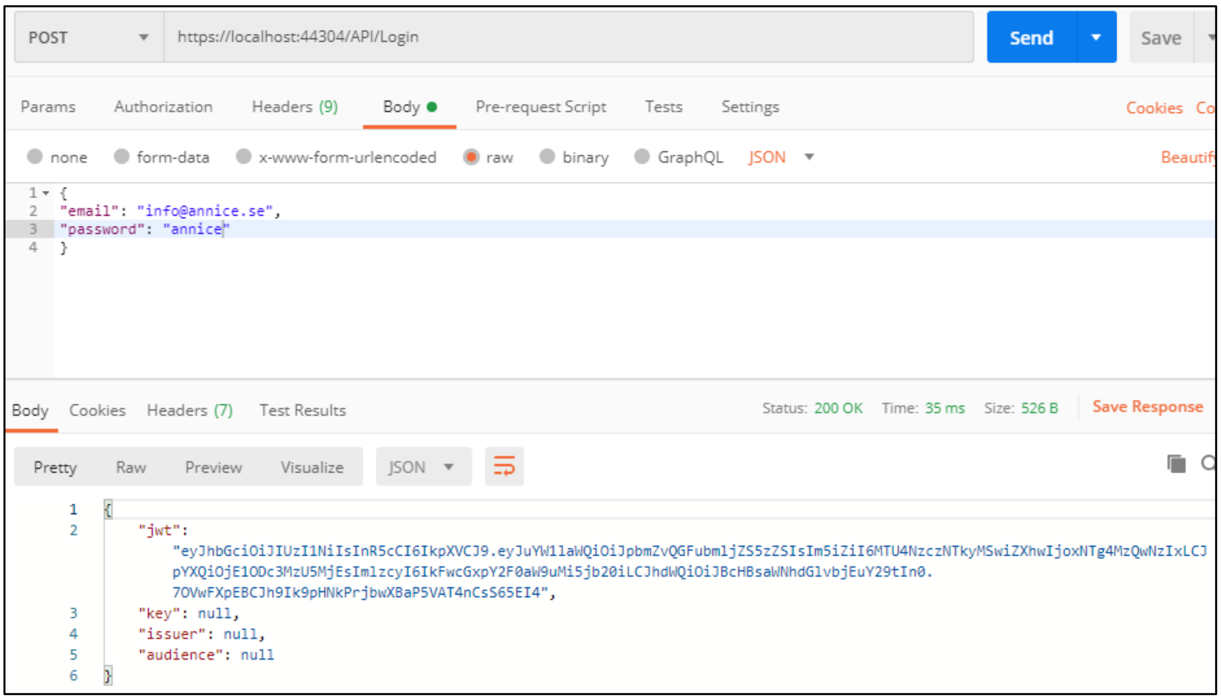
9. REST API documentation

A quick way to test the APIs is through the Postman client which can be downloaded from the following link: https://www.getpostman.com/downloads/

With Postman you can test different APIs to check whether they work as expected without having to use your actual application GUI. Since this script is based on REST APIs, this can be done by sending JSON objects to different endpoints supported by this script.

Once you have setup the script and launched application 2 (from where user details are fetched) as well as launched Postman, you can quickly test different requests to the App2 API by entering a request URL (endpoint), attaching a JSON object/message, and then execute the message to the endpoint.

In Postman, this can be done by creating a request, and then putting the JSON message under the "Body" tab with the "raw" option selected as in the screenshot below. After this, you just click the send button in Postman to receive the response message from the API we just called.



https://localhost:44304/API/Login

```
{
"email": "info@annice.se",
"password": "annice"
}
```

Status: 200 OK Time: 35 ms Size: 526 B

```
{
    "jwt":
        "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJuYW1laWQiOiJpbmZvQGFubmljZS5zZSIsIm5iZiI6MTU4NzczNTkyMSwiZXhwIjoxNTg4MzQwNzIxLCJ
        pYXQiOjE1ODc3MzU5MjEsImlzcyI6IkFwcGxpY2F0aW9uMi5jb20iLCJhdWQiOiJBcHBsaWNhdGlvbjEuY29tIn0.
        7OVwFXpEBCJh9Ik9pHNkPrjbwXBaP5VAT4nCsS65EI4",
    "key": null,
    "issuer": null,
    "audience": null
}
```

Image 3: *Screenshot of an API request with a received JSON web token from application 2 based on a successful login post.*

Given that you have launched application 2, the following API URLs (changed to your server paths and port numbers) can be used in Postman to post requests to the App2 endpoints based on the following JSON object structures:

API URL:	JSON object:

https://localhost:44304/API/Login	{
	    "email": "info@annice.se",
	    "password": "annice"
	}

https://localhost:44304/API/Update	{
	    "id": "1",
	    "firstname": "Annice",
	    "lastname": "Strömberg",
	    "email": "info@annice.se",
	    "password": "newpassword",
	    "jwt":
	"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
	eyJuYW1laWQiOiJ5b3VyQGVtYWlsLmNvbSIsIm
	5iZiI6MTU4Nzc0MjUwMSwiZXhwIjoxNTg4MzQ3
	MzAxLCJpYXQiOjE1ODc3NDI1MDEsImlzcyI6Ik
	FwcGxpY2F0aW9uMi5jb20iLCJhdWQiOiJBcHBs
	aWNhdGlvbjEuY29tIn0.r9iW239YE8rHLFZipy
	zwkhI0QhYyu2EWxCFFSWcDvHA"
	}

https://localhost:44304/API/GetUser	{
	    "email": "info@annice.se",
	    "jwt":
	"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
	eyJuYW1laWQiOiJ5b3VyQGVtYWlsLmNvbSIsIm
	5iZiI6MTU4Nzc0MjUwMSwiZXhwIjoxNTg4MzQ3
	MzAxLCJpYXQiOjE1ODc3NDI1MDEsImlzcyI6Ik
	FwcGxpY2F0aW9uMi5jb20iLCJhdWQiOiJBcHBs
	aWNhdGlvbjEuY29tIn0.r9iW239YE8rHLFZipy
	zwkhI0QhYyu2EWxCFFSWcDvHA"
	}

10 Contact details

For general feedback related to this script, such as any discovered bugs etc., you can contact me via the following email address: info@annice.se