**Exercise: Web API**

Problems for exercises and homework for the [“ASP.NET Core” course @ SoftUni](https://softuni.bg/trainings/2354/aspnet-core-july-2019).

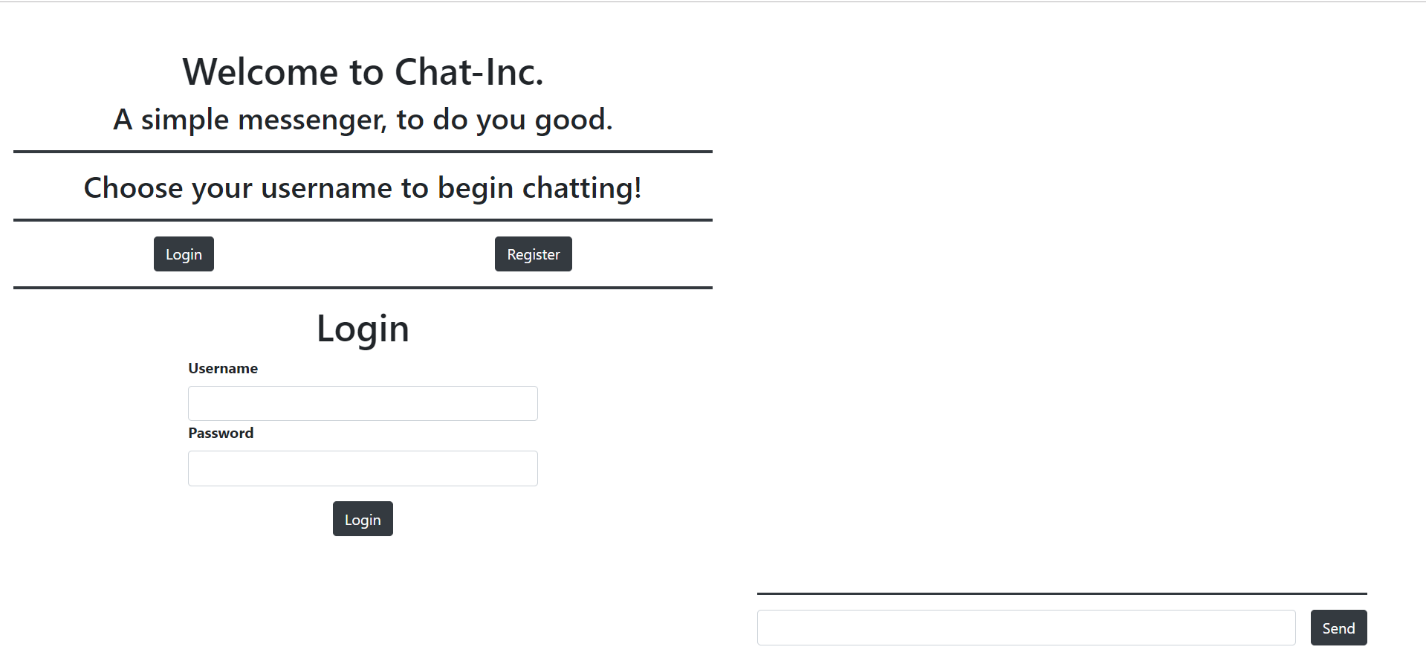
## Messages

Let’s extend the **Messages** **Web API**, by adding User Registration and User Authentication to it. This will add Users to the database and remove the plain text users from the Messages.

## Front-End Changes

We will need to change the index.html a little.

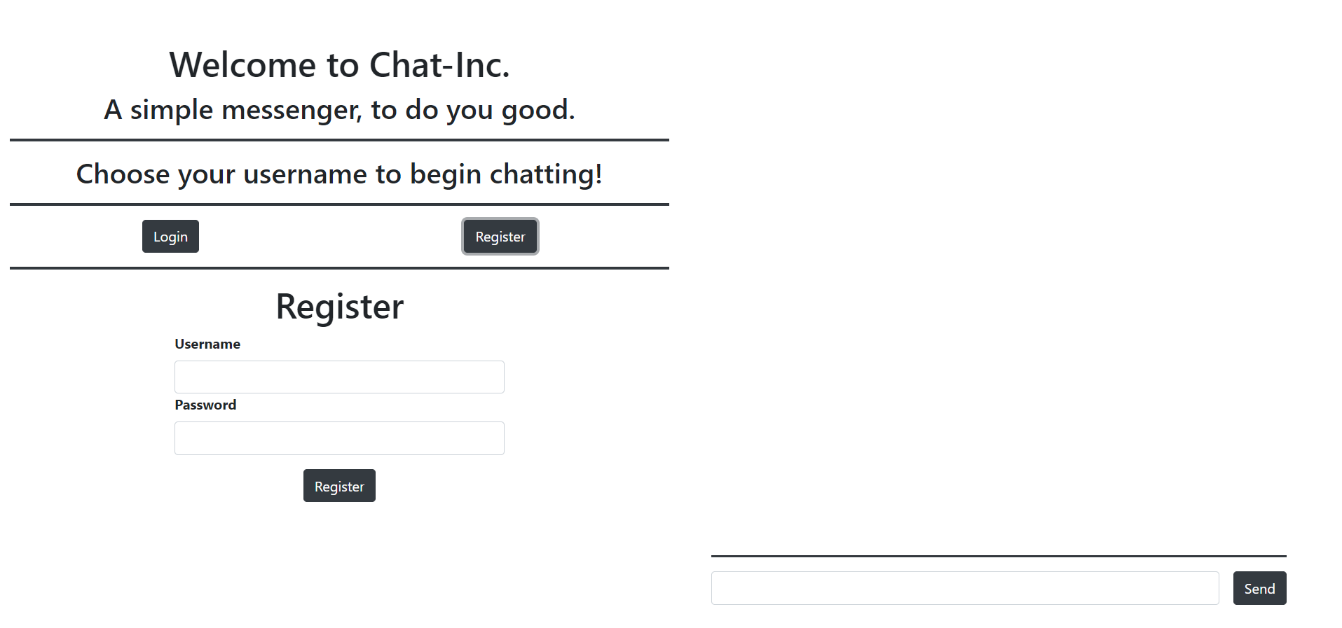
### Login Functionality



As you can see there is a new section with 2 buttons on it. When you click on the [Login] Button the Login Form should be visualized as shown above.

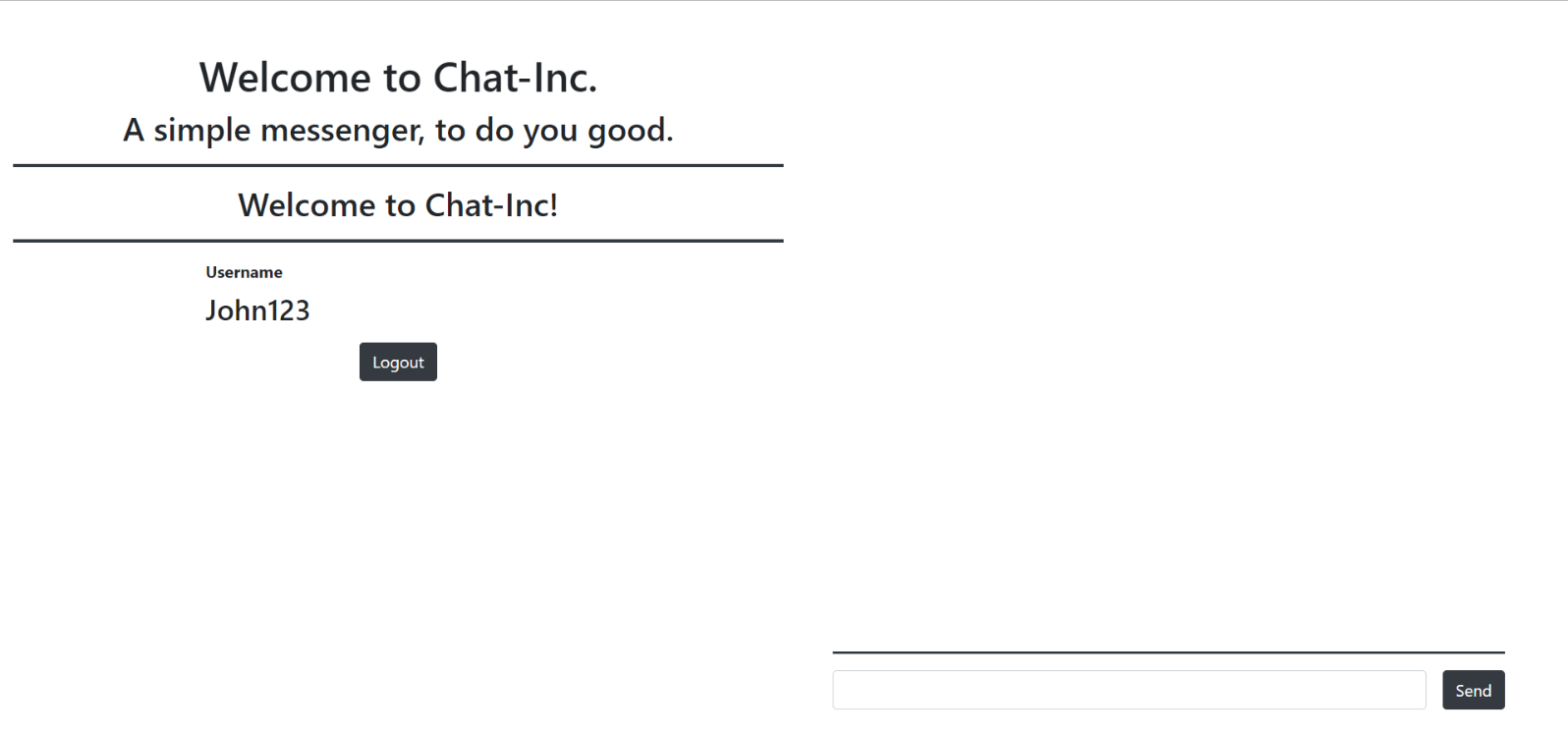
### Register Functionality

When you click on the [Register] button the Register Form should be visualized as shown below:



### Logged-In Functionality

Upon successful Back-End Authentication, you will get a JWT token. You need to store it in the Local Storage. Also, the following front-end should be visualized:



The caption above changed to "Welcome to Chat-Inc!".

The username of the currently-logged-in User is visualized.

The [Logout] button is visualized.

Upon clicking the [Logout] button the token should be deleted from the Local Storage and the Login Form should be visualized.

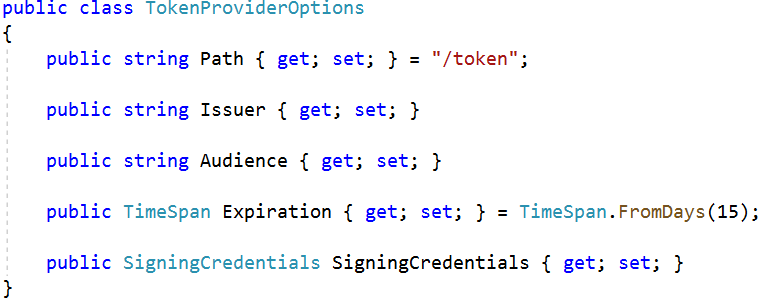
## Back-End Changes

You will need to add a simple User entity, which holds a Username and a Password. Later, this will be changed.

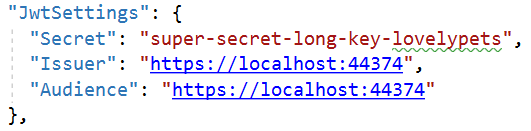
You will also need to implement appropriate Endpoints for User Registration and User Authentication.\

### Hint – Configuring the JWT

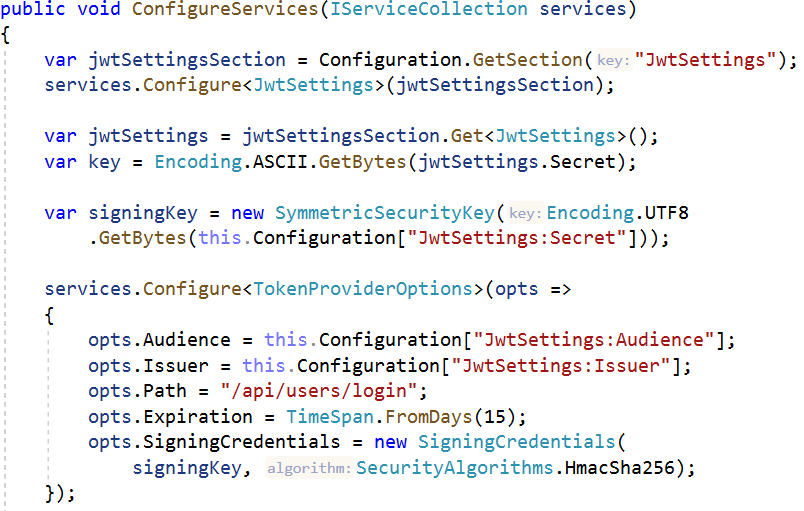
Let’s create a folder named **Jwt** where we will hold classes helping us to implement **JSON Web Tokens** to the application. Create two classes - **JwtSettings.cs** and **TokenProviderOptions.cs**. The first one will hold only one property – **Secret** (a **string**). The second one should look like this:



You can change the Expiration Days how you’d like. Now let’s add **JWT Settings** to **appsettings.json**.

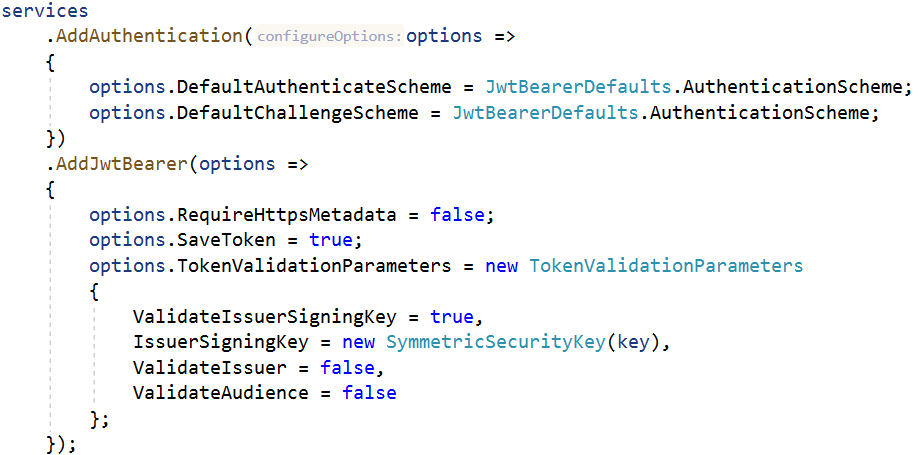


It’s time for the configuration in **StartUp.cs**. In **Configure()** we should have this code:



We get the **Secret** string from the class we wrote and create a **signing key**. Next we configure the **token options** with the options of the token and get the values from the **appsettings.json**.

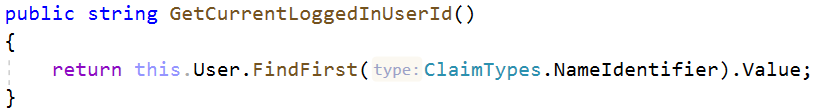
Next we say to our application that we will use **authentication** and **JWT Bearer** and we set options for them.



The magic of creating a token is in **Authenticate()** method in **UserService**. We will use it in our **Login()** method in **UsersController**. The implementation of the method is the same as in the presentation.



With the creation of **Claim** we can **access the logged in user** in the application like this:



## Common Functionalities

While implementing these functionality, try to keep the old functionality of the application intact:

* The Front-End should still be a single page application.
* Messages should still be visualized to the User, even if he is logged out.
* The Validations should be the same as before.

## Bonus Task: Identity User

Try to change the **User** functionality on the back-end so that it works with the Identity User, from ASP.NET Core.