MDS connector implementation

The connector is implemented as a RestAPI using ASP.Net core 3.0. The connector is currently a proof of concept, and it has the main purpose of demonstrating the two following features.

## X509Certificate authentication

A X509Certificate is the same type of certificates used in the HTTPS protocol, where the client can use it to verify the identity of the web server. In our case, we are using the certificate to verify the identity of a client, who is connecting to us (the connector running as an API service on a server). This is meant to be an alternative for authentication method for users in the whole Maritime data space, other than for instance Oauth2.

The certificate authentication feature is implemented using the built-in features of ASP.Net core.

To demonstrate the authentication flow, another program **MDSClient** is also implemented. It is made to be able to communicate with the MDSConnector and demonstrate the different authentication results by using different self-signed certificates (fake certificates).

**The life cycle of a request, authenticated using X509Certificate is as follows:**

1. Client initiate HTTPS connection to the server, with the certificate attached to the connection. (How this can be done using .Net core is demonstrated in the MDSClient)
2. Server receives the request, and the ASP.NET middleware intercepts the request, in order to perform authentication. This middleware is implemented in the class *CustomAuthenticationHandler.*
3. If the connection to the server has a valid certificate according to logic in the authentication handler, appropriate claims are awarded to the request. Otherwise, no claims will be awarded.
4. The request is passed through other middleware and in the end arrives at the endpoint in a controller. This is where the authorization takes place.
5. The authorization attribute inspects the request and checks if all required claims are present. If not, the request is denied access, and the client receives a 401 or 403 depending on the specific request (see code for specifics). If all required claims are present, then the business logic for the endpoint takes over, and the client receives the data that is expected.

*Note that authorization attribute is a mechanism that explicitly DENIES access. This means all requests will be granted access, unless there is logic that explicitly sets the context result to be a failing result.*

Documentation on **Authentication** in ASP.NET core: <https://docs.microsoft.com/en-us/aspnet/core/security/authentication/?view=aspnetcore-3.1>

Documentation on **Authorization** in ASP.NET core: <https://docs.microsoft.com/en-us/aspnet/core/security/authorization/introduction?view=aspnetcore-3.1>

**Testing**

There are unit tests for the current implementation in the *MDSConnectorTests* project. These can serve as starting points or templates for new unit tests on new features of the authentication.

## Business case: Automatic EU MRV report generation

**THIS FEATURE IS NOT FINISHED**