PROJET 2021-2022

## **Specifications:**

* Use at least .Net core 5 (can be .net 6 also) **--- OK**
* Respect standard :
  + Verbs
  + Status Code
  + Description
  + …
* Use MVC for WEBSITE **--- OK**
  + Can be Angular
  + Can be React
* Use MVC for API  **--- OK**
* Make difference between:
  + Authentication => **Authenticating with a token -- OK**
  + Authorization => **Having permission for certain actions (read, write, …) -- OK**
* Use FluentValidation for Request validation **-- OK**
* Use the CQRS architecture for both project : -> 09/05 (presential) **-- OK**
  + API
  + WEBSITE
  + Exemple : // a pokemon API (https://pokeapi.co/)
* Use HTTPClient or HTTPClientFactory **-- OK**
* Use a Retry Policies -> 02/05 **– OK (Polly) – API AUTH0 API Management**
* Use a logging system for all events -> 02/05 -- **OK (SERILOG)**
* Create a Middleware -> 02/05 -- **OK**
* Create and consume NuGet
* Consume an external NuGet -- **OK**
* Don’t allow social provider for Authentication **-- OK**
* Use versioning in API **-- OK**
* You can use in Memory DB if it’s easier for you – **DB FIRST | OK**

## **Technical part :**

* Secure an API using Auth0
  + Flow : Client Credential **-- OK**
* Secure a WEBSITE using Auth0
  + Flow : Authorization Code + PKCE  **-- OK**
* Use token delivered from WEBSITE to use Backend API – **CONFIG OK**
* All requests coming from WEBSITE should use Backend API -- **OK**
* Use Role
  + Roles :
    - Owner
    - Admin
    - User
    - Agent
* Use Permission :
  + Permissions :
    - “read:messages”
      * Owner
      * Admin
      * User
      * Agent
    - “write:messages”
      * Owner
      * Admin
      * User
    - “read:users”
      * Owner
      * Admin
    - “write:users”
      * Owner
* A “end user” must have one and only one role.
* Create and use a Middleware to manage Permissions/Roles (02/05)
  + Meaning: I don’t want to see verification on Permission and/or Role on each controller
* Create a custom Middleware to manage exception (02/05) **-- OK**
* Consume an external API **– OK (Auth0 API MANAGEMENT)**
  + Don’t forget retry Policies (02/05)
* Make AJAX Call on Website level to call an external API  **--- OK (PLANETS INFORMATIONS – SolarSystem View)**
  + Call is done on View level
* Use Client-Side validation (10/05) AND Server-Side validation for Website
* Use pagination for Grid having more than 50 results -- **OK**
* Create some script on WEBSITE level --
* Create a dedicated NuGet and use it in the project (02/05)
* Consume a NuGet and explain your choice **-- OK**
* Consume and use some script on WEBSITE level **-- OK**
* Use Versioning API - **OK**
  + Give versioning in a header with date format like DD-MM-YYYY **– OK VANDERROST DON’T NEED DATE**

## Functionality :

* Have a Single Sign On page with some customization?  **-- OK (Need more customization)**
* Sending an email for activate account for new user created
* Sending an email for reset password
* Create an user
* Update an user
* Delete an user
* List users
* Assign a role to a user
* Create a message
* Update a message
* Deleted a message
* List message
* Base on role
  + Hidden some functionality
  + Restrict access base on permission associated to the role
* Having a dropdown depending from another dropdown
  + Example :
    - Dropdown1 :
      * Value : “1- Yes”
      * Value : “2- No”
      * Value : “3- Other”
    - If Other
      * Show a required textbox
      * Hide second dropdown
    - If Yes
      * Show a list of cars in a second dropdown
    - If No
      * Show a list of bike in a second dropdown
* Use Chart from google to use google API – WON’T DO
* Show a list coming from an external API and use pagination -- **Pagination OK – Still API to consume**
* Show different layout based on user role
* Use Toastr for notification after save for example **-- OK**
  + Example : “new user created”
* CRUD on a dedicate projet **-- OK**