Assignment 4 – WebApi  
COS318 – Web Programming

In the fourth assignment, we will be returning to Star Wars. “I think my eyes are getting better. Instead of a big dark blur, I’m seeing a big bright blur.” In the third assignment, you wrote javascript that communicated with a server that was hosted on Azure. In this assignment, you’ll be recreating that server. You’ll be able to use your assignment three code to test your new server. Also, the server from assignment three is still deployed to Azure at “https://webrequestsserverbethel.azurewebsites.net/api/favoritecharacters” so you can verify if your server is acting in the same way. Your server will be tracking a list of favorite characters. Your server does not need to persist any data, so it is okay if any data is lost when your server is stopped.

Your FavoriteCharacters controller must support the following endpoints:

1. **(10 Points) GET**
   1. Return a JSON array of all the favorite characters currently stored on the server.
      1. The “views” key should be set for each character in your response JSON equal to the number of views that character has so far (as an int)
2. **(10 Points)** **GET/{index}**
   1. Return a JSON object containing the favorite character and name data for the specified index.
      1. The “views” key should be set for this character in your response JSON equal to the number of views this character has so far (as an int)
   2. Validate that the index sent in is valid.
3. **(20 Points) POST**
   1. Accept JSON data for FirstName, LastName, and Character and save it on the server.
   2. Validate that FirstName and Character are sent in the JSON and have length of at least 1 using entity annotations.
   3. Return the created data.
4. **(20 Points) GET /{index}/views**
   1. Return a list of views for a favorite character stored on the server.
      1. This should just be a simple list of strings, no special formatting required.
5. **(20 Points) POST /{index}/views**
   1. Accept JSON data for a view date (as a string). It only has a single key/value pair, “ViewDate.”
   2. Validate that ViewDate is sent in the JSON and has a length of at least 1 using entity annotations.
   3. Return the created data.
6. **(20 Points)** Code style, formatting, completeness, and quality.
   1. If the list of characters gets larger than 30, empty it before adding any new data.
   2. The list of characters should begin with one entry.

Stretch Levels

If you already have some experience with WebApi, or if you aren’t frozen in carbonite, try to complete these stretch levels for a reputation bonus. If you try for the stretch levels, make sure to type it in the comments on Moodle so I don’t miss it.

**Purple Lightsaber Level**

Add a new endpoint for deleting a favorite character out of the list.

**Green Lightsaber Level**

Add a new endpoint for retrieving a specific view from a favorite character.

**Blue Lightsaber Level**

Add a filter to your server that will reject the request before the controller code is executed if the http verb is POST, PUT, or PATCH and the content type isn’t application/json.

The Rules

1. No inline styles or inline javascript.
2. Error messages must be “in-page” i.e. no pop-ups or alerts.
3. Any resources not created by you (images, javascript libraries, etc.) must be referenced using a CDN or URL, not directly included in your assignment submission.
4. New Rule: Service/data/model classes must not have any http, request, or response references.
5. New Rule: Controller entity classes must not be used directly to store data on the server; translate them into a model (data storage) class before saving the data. Conversely, controllers must not send any model classes to the user; translate them into controller entity classes before sending the response.