**Coding**

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

namespace WebApplication4.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class LuhnAlgorithmController : Controller

{

[HttpGet(Name = "GetLuhnAlgorithm")]

public bool Get(string CreditCardNumber)

{

double retNum;

bool IsNumeric = false;

if (CreditCardNumber != null)//checkng null

{

IsNumeric = double.TryParse(CreditCardNumber, out retNum); //checking Numeric

try

{

if (CreditCardNumber.Length == 16 || IsNumeric)//checking 16 digit credit card number

{

int[] card = new int[16];

for (int i = 0; i <= 15; i++)// store in a array

{

card[i] = Int32.Parse(CreditCardNumber.Substring(i, 1));

}

int Endnub = card[15];// end no.or 16 digit of CreditCard which is my given Check Digit

int[] RevCardNo = new int[15];

int j = 0;

for (int i = 14; i >= 0; i--)//store in new array in reverse order

{

RevCardNo[j] = card[i];

j++;

}

//calculate Luhn algorithm---

for (int i = 1; i <= 14; i = i + 2)//fatch each 2nd digit start from the rightmost digit. Moving left.

{

// double the value of every second digit if product number is single digit then replace else add product number and then replace

RevCardNo[i] = (RevCardNo[i] \* 2 < 10) ? (RevCardNo[i] \* 2) : (1 + (RevCardNo[i] \* 2 % 10));

}

int sum = 0;

for (int i = 0; i <= 14; i++)

{

sum = sum + RevCardNo[i];//Sum the values of the resulting digits.

}

if (Endnub == (10 - sum % 10))//Compare the check digit result with the original check digit. If both numbers match, the result is valid.

return true;

else

return false;

}

else

return false;

}

catch (Exception ex) { return false; }

}

else { return false; }

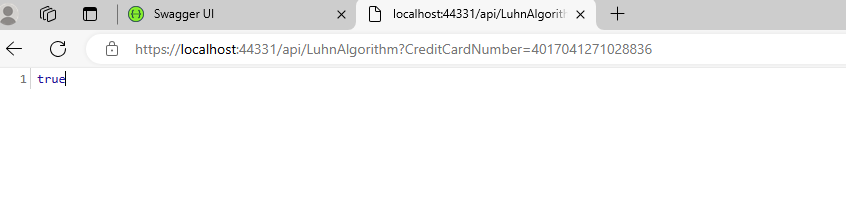
}

}

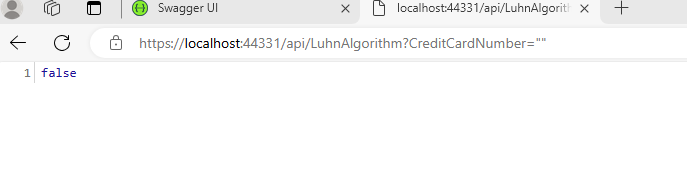
}

**UnitTesting(Report)**

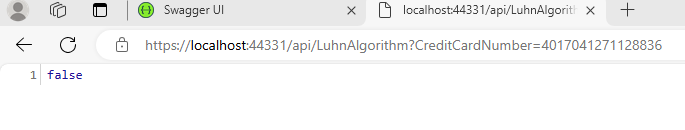
Test using a valid CreditCardNumber



Test with CreditCardNumber blank



Test with 16 digit but invalid CreditCardNumber



Test with invalid CreditCardNumber

