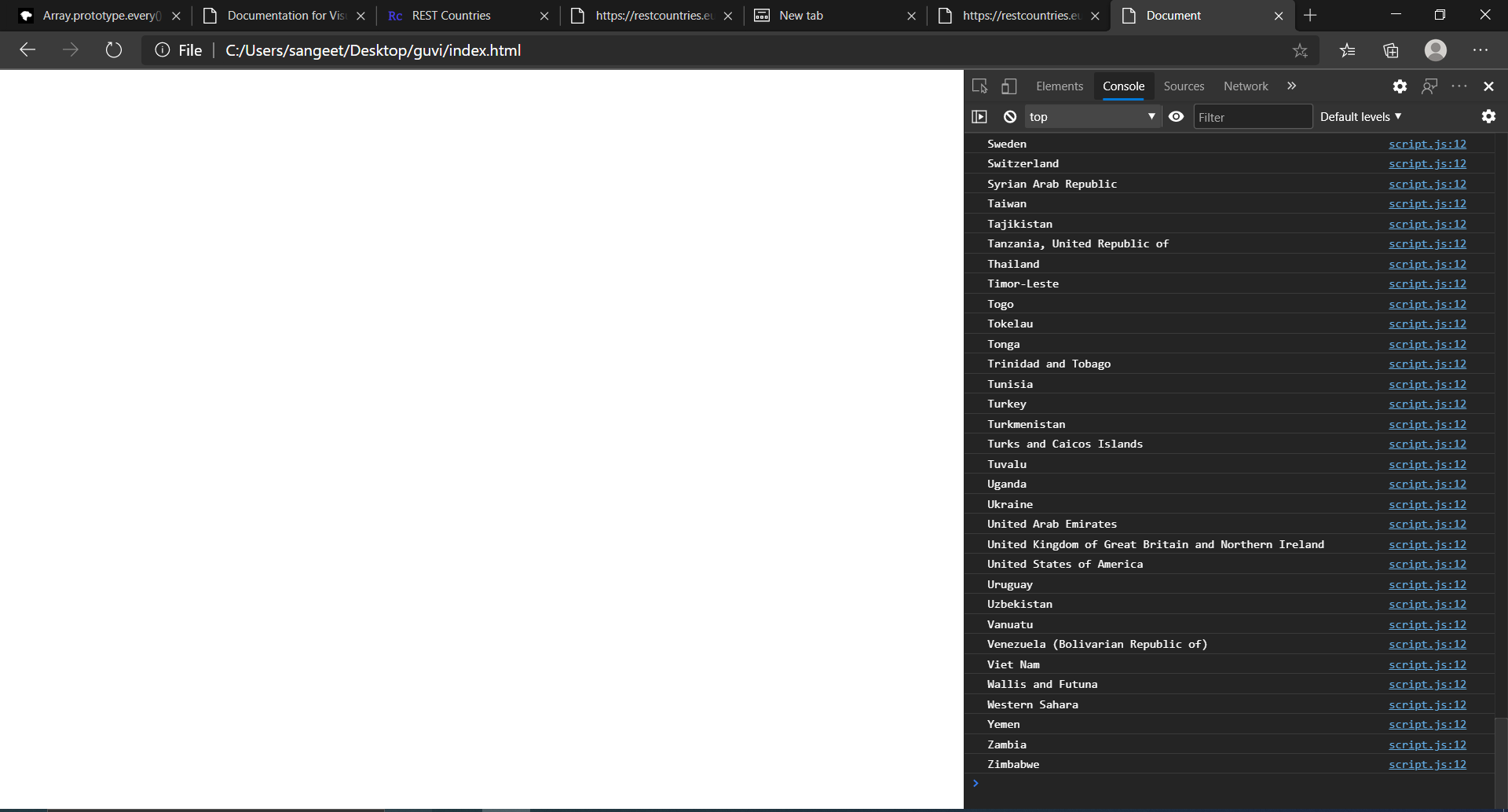
**Task -2**

1. **Load the rest countries data using your html and script.js file and run a for loop on the data and print all the country name in the console.**



**Source code:**

Index.html:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script src="script.js"></script>

</body>

</html>

**Script.js**

//CREATE REQUEST VAR

var request=new XMLHttpRequest();

//CFREATE CONN

request.open("GET","https://restcountries.eu/rest/v2/all",true);

//send connection

request.send();

//reg event listener

request.onload=function(){

    var countrydata=JSON.parse(this.response);

    for( var i in countrydata){

        console.log(countrydata[i].name);

    }

}

**2. Write a write up on Difference between copy by value and copy by reference.**

***Copy By Value***

*In Copy by value the data which variable hold is passed to another variable . Both the variables refers two different memory locations.*

*Var a=20;*

*Var copied\_variable=a;*

*Console.log(a,copied\_variable) //20 20*

*Var copied\_variable=100;*

*Console.log(a,copied\_variable) //20 100*

*In the above example For variable named(copied\_variable) data which is present in the a is copied not value . Even if contents of Copied variable are changed original variable value from which it is copied remain same .*

***Copy by reference***

*For composite data types objects are copied by reference. Actual variable and copied variable both refer to same memory location . SO change in data of copied variable will lead to change in data of original variable too.*

Var a=[1,2,3];

Var b=a;

Console.log(a,b);//(3)[1,2,3]

B[2]=100;

Console.log(a,b); );//(3)[1,2,100]

*In the above example we could notice that changing the value of copied variable lead to change of original variable from which it is copied as they both refer same memory location.*

**3. How to copy by value a composite datatype (array+objects).**

There are 3 ways to copy by value for composite data types.

1. Using the spread (...) operator
2. Using the Object.assign() method
3. Using the JSON.stringify() and JSON.parse() methods

Using Spread

**Spread operator** allows an iterable to expand in places where 0+ arguments are expected. It is mostly used in the variable array where there is more than 1 values are expected. It allows us the privilege to obtain a list of parameters from an array.Using spread will clone your object.

Using Object.assign()

The **Object.assign()** method copies all enumerable own properties from one or more *source objects* to a *target object*. It returns the target object. Note this will be a shallow copy.

Using JSON.parse() and JSON.stringify()

The JSON object, available in all modern browsers, has two useful methods to deal with JSON-formatted content: parse and stringify. JSON.parse() takes a JSON string and transforms it into a JavaScript object. JSON.stringify() takes a JavaScript object and transforms it into a JSON string.Using JSON.parse() and JSON.stringify() for copy performs deep copy .

Copy by value using JSON methods

The deep copy is a true copy for nested objects. Shallow copy copies only reference in case of nested objects.