

Informe de Laboratorio 04

Tema: Ajax

Nota

Estudiante	Escuela	Asignatura
Victor Gonzalo Maldonado Vilca vmaldonadov@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2 Semestre: III Código: 1702122

Tarea	Tema	Duración
04	Ajax	2 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 9 de abril de 2024	Al 18 de mayo de 2024

1. Tarea

Ejercicios Propuestos:

- Revisar Ajax en w3schools y Presentar un informe con un pantallazo por cada ejemplo de Ajax realizado en su propio servidor web.
- Usar un servidor web como apache en su computadora (Xampp), descargar el archivo de datos data.json y dejarlo en el mismo lugar donde lanzó el servidor. **Para cada uno de estos problemas implemente un programa en ajax y página que:**
 - Liste todas las “regiones”.
 - Muestre el número total de confirmados por región.
 - Encuentre las 10 regiones cuya suma total sea la mayor.
 - Visualice un gráfico en el tiempo de los valores para la región de Arequipa.
 - Haga gráficos comparativos entre regiones usando líneas.
 - Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao.
 - Haga gráficos comparativos entre regiones elegidas por el usuario.
 - Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao, mostrando el número de confirmados por cada día.
- En grupos de 3 a 5 personas (los grupos se definirán en clase) implemente una aplicación web que navegue sobre archivos Markdown y permita:
 - Listar los archivos Markdown disponibles.

- Ver el contenido de un archivo markdown traducido a HTML.
- Crear nuevos archivos MarkDown y almacenarlos en el servidor

Tener en cuenta:

- La comunicación entre el cliente y el servidor tiene que ser usando JSON sólamente.
- El cliente debe usar AJAX para sus peticiones.
- El servidor debe usar NodeJS.
- Su aplicación debe ser de página única, es decir que sólo habrá un archivo index.html y nada más.

2. Entregables

- Informe de trabajo
- URL: Repositorio de GitHub

3. Equipos, materiales y temas utilizados

- HTML
- CSS
- JavaScript
- w3schools
- Xampp
- Node.js
- Ajax
- GitHub

4. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- <https://github.com/Victor-Gonzalo-Maldonado-Vilca/TareaAjax.git>

5. Link de Video

- URL video en Youtube.
- <https://www.youtube.com/watch?v=tPlgxQkUKaY>

6. Desarrollo del trabajo

6.1. AJAX w3schools - EXAMPLES

Capturas de pantalla de los ejemplos de Ajax, usando su propio servidor.

■ Tutorial JavaScript sección Ajax

The screenshot shows a sidebar titled 'JS AJAX' with a list of topics, each preceded by a green checkmark icon:

- AJAX Intro
- AJAX XMLHttpRequest
- AJAX Request
- AJAX Response
- AJAX XML File
- AJAX PHP
- AJAX ASP
- AJAX Database
- AJAX Applications
- AJAX Examples

To the right of the sidebar, there is a main content area with the following text:

JavaScript is the world's most popular programming language.
 JavaScript is the programming language of the Web.
 JavaScript is easy to learn.
 This tutorial will teach you JavaScript from basic to advanced.

[Start learning JavaScript now »](#)

Below this, a section titled 'Examples in Each Chapter' is shown with the subtext: 'With our "Try it Yourself" editor, you can edit the source code and view the result.'

Figura 1: w3schools

6.1.1. AJAX - Intro

■ AJAX Example

The screenshot shows a code editor interface with the following code:

```
<!DOCTYPE html>
<html>
<body>
<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>
```

On the right side of the editor, there is a preview window titled 'The XMLHttpRequest Object' with the heading 'Change Content'. Below the heading is a button labeled 'Change Content'.

Figura 2: AJAX

```
<!DOCTYPE html>
<html>
<body>

<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 3: Presionando el botón

6.1.2. AJAX - XMLHttpRequest

- **Ejemplo 1 - Enviar una Solicitud:**

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<p>Let AJAX change this text.</p>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 4: Enviar Solicitud

The screenshot shows a web-based code editor interface. On the left, there is a code editor window containing the following HTML and JavaScript code:

```

<!DOCTYPE html>
<html>
<body>
<h2>The XMLHttpRequest Object</h2>
<div id="demo">
<p>Let AJAX change this text.</p>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>

```

On the right, there is a preview window titled "The XMLHttpRequest Object" with the heading "AJAX". The preview content includes the following text:

AJAX is not a programming language.
 AJAX is a technique for accessing web servers from a web page.
 AJAX stands for Asynchronous JavaScript And XML.

At the bottom left of the code editor, the URL <https://www.w3schools.com> is visible.

Figura 5: Presionando el botón

■ **Ejemplo 2 - Propiedad de carga:**

The screenshot shows a web-based code editor interface. On the left, there is a code editor window containing the following HTML and JavaScript code:

```

<!DOCTYPE html>
<html>
<body>
<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>

```

On the right, there is a preview window titled "The XMLHttpRequest Object" with the heading "AJAX". The preview content includes a button labeled "Change Content".

Figura 6: Propiedad de Carga

The screenshot shows a browser window with the following content:

```
<!DOCTYPE html>
<html>
<body>

<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>
```

AJAX

AJAX is not a programming language.
 AJAX is a technique for accessing web servers from a web page.
 AJAX stands for Asynchronous JavaScript And XML.

Figura 7: Presionando el botón

■ Ejemplo 3 - Propiedad Onreadystatechange:

The screenshot shows a browser window with the following content:

```
<!DOCTYPE html>
<html>
<body>

<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("demo").innerHTML =
        this.responseText;
    }
  };
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>
```

The XMLHttpRequest Object

Change Content

Figura 8: Propiedad Onreadystatechange

```
<!DOCTYPE html>
<html>
<body>

<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("demo").innerHTML =
        this.responseText;
    }
  };
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>
```

Figura 9: Presionando el botón

6.1.3. AJAX - Request

■ Get Requests:

1. Primer Ejemplo:

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>
<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("GET", "demo_get.asp");
  xhttp.send();
}
</script>
</body>
</html>
```

Figura 10: Ejemplo 1 - GET

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>
<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("GET", "demo_get.asp");
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 11: Presionando botón

2. Segundo Ejemplo:

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>
<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("GET", "demo_get.asp?t=" + Math.random());
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 12: Ejemplo 2 - GET

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>

<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("GET", "demo_get.asp?t=" + Math.random());
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 13: Presionando el botón

3. Tercer Ejemplo:

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>

<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("GET", "demo_get2.asp?fname=Henry&lname=Ford");
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 14: Ejemplo 3 - GET

The screenshot shows a web browser window with the title "The XMLHttpRequest Object". On the left, there is a code editor containing an HTML file with JavaScript code. The code creates an XMLHttpRequest object, sets up an event listener for its 'load' event, and then sends a GET request to a URL. On the right, the browser's results panel shows the output of the script: "Hello Henry Ford". There is also a "Request data" button.

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>

<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("GET", "demo_get2.asp?fname=Henry&lname=Ford");
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 15: Presionando el botón

■ Post Request:

1. Primer Ejemplo:

The screenshot shows a web browser window with the title "The XMLHttpRequest Object". The code is identical to Figure 15, but the browser's results panel shows a blank page. This indicates that the POST request was successful, but the response content was not displayed in the browser's preview area.

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>

<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("POST", "demo_post.asp");
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 16: Ejemplo 1 - POST

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>

<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("POST", "demo_post.asp");
  xhttp.send();
}
</script>

</body>
</html>
```

The XMLHttpRequest Object

Request data

This content was requested using the POST method.

Requested at: 5/16/2024 2:23:49 PM

Figura 17: Presionando el botón

2. Segundo Ejemplo:

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>

<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("POST", "demo_post2.asp");
  xhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
  xhttp.send("fname=Henry&lname=Ford");
}
</script>

</body>
</html>
```

The XMLHttpRequest Object

Request data

Figura 18: Ejemplo 2 - POST

The screenshot shows a web-based code editor interface. On the left, the code for "The XMLHttpRequest Object" is displayed:

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Request data</button>

<p id="demo"></p>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
  }
  xhttp.open("POST", "demo_post2.asp");
  xhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
  xhttp.send("fname=Henry&lname=Ford");
}
</script>
</body>
</html>
```

On the right, the results of the code execution are shown under the heading "The XMLHttpRequest Object". It includes a "Request data" button and the resulting output "Hello Henry Ford".

Figura 19: Presionando el botón

■ Synchronous Request

The screenshot shows a web-based code editor interface. On the left, the code for a synchronous XMLHttpRequest example is displayed:

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<p id="demo">Let AJAX change this text.</p>
<button type="button" onclick="loadDoc()">Change Content</button>

<script>
function loadDoc() {
  var xhttp = new XMLHttpRequest();
  xhttp.open("GET", "ajax_info.txt", false);
  xhttp.send();
  document.getElementById("demo").innerHTML = xhttp.responseText;
}
</script>
</body>
</html>
```

On the right, the results of the code execution are shown under the heading "The XMLHttpRequest Object". It includes a "Change Content" button and the placeholder text "Let AJAX change this text."

Figura 20: Ejemplo - Synchronous Request

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<p id="demo">Let AJAX change this text.</p>
<button type="button" onclick="loadDoc()">Change Content</button>

<script>
function loadDoc() {
  var xhttp = new XMLHttpRequest();
  xhttp.open("GET", "ajax_info.txt", false);
  xhttp.send();
  document.getElementById("demo").innerHTML = xhttp.responseText;
}
</script>
</body>
</html>
```

The XMLHttpRequest Object

AJAX

AJAX is not a programming language.
 AJAX is a technique for accessing web servers from a web page.
 AJAX stands for Asynchronous JavaScript And XML.

Figura 21: Presionando el botón

6.1.4. AJAX - Response

■ Propiedad de texto de respuesta

```
<!DOCTYPE html>
<html>
<body>

<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>
```

The XMLHttpRequest Object

Figura 22: Propiedad de Texto de Respuesta

```
<!DOCTYPE html>
<html>
<body>

<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>
```

Figura 23: Presionando el botón

■ La propiedad ResponseXML

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<p id="demo"></p>

<script>
const xhttp = new XMLHttpRequest();
xhttp.onload = function() {
  const xmlDoc = this.responseXML;
  const x = xmlDoc.getElementsByTagName("ARTIST");
  let txt = "";
  for (let i = 0; i < x.length; i++) {
    txt = txt + x[i].childNodes[0].nodeValue + "<br>";
  }
  document.getElementById("demo").innerHTML = txt;
}
xhttp.open("GET", "cd_catalog.xml");
xhttp.send();
</script>
</body>
</html>
```

The XMLHttpRequest Object

Bob Dylan
 Bonnie Tyler
 Dolly Parton
 Gary Moore
 Eros Ramazzotti
 Bee Gees
 Dr.Hook
 Rod Stewart
 Andrea Bocelli
 Percy Sledge
 Savage Rose
 Many
 Kenny Rogers
 Will Smith
 Van Morrison
 Joni Mitchell
 Cat Stevens
 Sam Brown
 T'Pau
 Tina Turner
 Kim Larsen
 Luciano Pavarotti
 Otis Redding
 Simply Red
 The Communards
 Joe Cocker

Figura 24: Propiedad de Texto de Respuesta

■ Método getAllResponseHeaders()

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<p>The getAllResponseHeaders() function returns all the header information of a resource, like length, server-type, content-type, last-modified, etc:</p>

<pre id="demo"></pre>

<script>
const xhttp = new XMLHttpRequest();
xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
        this.getAllResponseHeaders();
}
xhttp.open("GET", "ajax_info.txt");
xhttp.send();
</script>

</body>
</html>
```

The XMLHttpRequest Object

The getAllResponseHeaders() function returns all the header information of a resource, like length, server-type, content-type, last-modified, etc:

```
age: 1644
cache-control: public,max-age=14400,public
content-encoding: gzip
content-length: 147
content-security-policy: frame-ancestors 'self' https://mycourses.w3schools.com;
content-type: text/plain
date: Thu, 16 May 2024 14:32:41 GMT
etag: "0c936788b97d10:gzip"
last-modified: Thu, 16 May 2024 12:20:42 GMT
server: ECS (nic/9849)
vary: Accept-Encoding
x-cache: HIT
x-content-security-policy: frame-ancestors 'self' https://mycourses.w3schools.com;
x-powered-by: ASP.NET
```

Figura 25: Método getAllResponseHeaders

■ Método getResponseHeader()

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<p>The getResponseHeader() function is used to return specific header information from a resource, like length, server-type, content-type, last-modified, etc:</p>

<p>Last modified: <span id="demo"></span></p>

<script>
const xhttp=new XMLHttpRequest();
xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
        this.getResponseHeader("Last-Modified");
}
xhttp.open("GET", "ajax_info.txt");
xhttp.send();
</script>

</body>
</html>
```

The XMLHttpRequest Object

The getResponseHeader() function is used to return specific header information from a resource, like length, server-type, content-type, last-modified, etc:

Last modified: Thu, 16 May 2024 12:20:42 GMT

Figura 26: Método getResponseHeader

6.1.5. AJAX XML - File

■ AJAX XML EXAMPLE

```

<!DOCTYPE html>
<html>
<style>
table,th,td {
    border : 1px solid black;
    border-collapse: collapse;
}
th,td {
    padding: 5px;
}
</style>
<body>

<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Get my CD collection</button>
<br><br>
<table id="demo"></table>

<script>
function loadDoc() {
    const xhttp = new XMLHttpRequest();
    xhttp.onload = function() {
        myFunction(this);
    }
    xhttp.open("GET", "cd_catalog.xml");
    xhttp.send();
}

function myFunction(xml) {
    const xmlDoc = xml.responseXML;
    const x = xmlDoc.getElementsByTagName("CD");
    let table = "<tr><th>Artist</th><th>Title</th></tr>";
    for (let i = 0; i <x.length; i++) {
        table += "<tr><td>" + 
        x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue + 
        "</td><td>" + 
        x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue + 
        "</td></tr>";
    }
    document.getElementById("demo").innerHTML = table;
}
</script>
</body>
</html>

```

Figura 27: Ajax XML

Artist	Title
Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros
Bee Gees	One night only
Dr Hook	Sylvia's Mother
Rod Stewart	Maggie May
Andrea Bocelli	Romanza
Percy Sledge	When a man loves a woman
Savage Rose	Black angel
Many	1999 Grammy Nominees
Kenny Rogers	For the good times
Will Smith	Big Willie style
Van Morrison	Tupelo Honey
Joni Mitchell	Soulsville
Cat Stevens	The very best of
Sam Brown	Stop
T'Pau	Bridge of Spies
Tina Turner	Private Dancer
Kim Larsen	Midt em natten
Luciano Pavarotti	Pavarotti Gala Concert
Otis Redding	The dock of the bay
Simply Red	Picture book
The Communards	Red

Figura 28: Presionando el botón

6.1.6. AJAX PHP

■ AJAX PHP EXAMPLE

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<h3>Start typing a name in the input field below:</h3>

<p>Suggestions: <span id="txtHint"></span></p>
<p>First name: <input type="text" id="txt1" onkeyup="showHint(this.value)"></p>

<script>
function showHint(str) {
  if (str.length == 0) {
    document.getElementById("txtHint").innerHTML = "";
    return;
  }
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("txtHint").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "gethint.php?q=" + str);
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 29: Ajax PHP

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<h3>Start typing a name in the input field below:</h3>

<p>Suggestions: <span id="txtHint"></span></p>
<p>First name: <input type="text" id="txt1" onkeyup="showHint(this.value)"></p>

<script>
function showHint(str) {
  if (str.length == 0) {
    document.getElementById("txtHint").innerHTML = "";
    return;
  }
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("txtHint").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "gethint.php?q=" + str);
  xhttp.send();
}
</script>

</body>
</html>
```

Figura 30: Escribiendo en el cuadro de texto

6.1.7. AJAX ASP

■ AJAX ASP EXAMPLE

```

<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<h3>Start typing a name in the input field below:</h3>

<p>Suggestions: <span id="txtHint"></span></p>
<p>First name: <input type="text" id="txt1" onkeyup="showHint(this.value)"></p>

<script>
function showHint(str) {
  if (str.length == 0) {
    document.getElementById("txtHint").innerHTML = "";
    return;
  }
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("txtHint").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "gethint.asp?q=" + str);
  xhttp.send();
}
</script>

</body>
</html>

```

Figura 31: Ajax ASP Example

The XMLHttprequest Object

Start typing a name in the input field below:

Suggestions: Raquel

First name:

```

<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<h3>Start typing a name in the input field below:</h3>

<p>Suggestions: <span id="txtHint"></span></p>
<p>First name: <input type="text" id="txt1" onkeyup="showHint(this.value)"></p>

<script>
function showHint(str) {
  if (str.length == 0) {
    document.getElementById("txtHint").innerHTML = "";
    return;
  }
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("txtHint").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "gethint.asp?q=" + str);
  xhttp.send();
}
</script>

</body>
</html>

```

Figura 32: Escribiendo en el cuadro de texto

6.1.8. AJAX DATABASE

■ AJAX ASP EXAMPLE

```
<!DOCTYPE html>
<html>
<style>
th,td {
    padding: 5px;
}
</style>
<body>

<h2>The XMLHttpRequest Object</h2>

<form action="">
<select name="customers" onchange="showCustomer(this.value)">
<option value="">>Select a customer:</option>
<option value="ALFKI">Alfreds Futterkiste</option>
<option value="MORTS">North/South</option>
<option value="WOLZA">Wolski Zajazd</option>
</select>
</form>
<br>
<div id="txtHint">Customer info will be listed here...</div>

<script>
function showCustomer(str) {
    if (str == "") {
        document.getElementById("txtHint").innerHTML = "";
        return;
    }
    const xhttp = new XMLHttpRequest();
    xhttp.onload = function() {
        document.getElementById("txtHint").innerHTML = this.responseText;
    }
    xhttp.open("GET", "getcustomer.php?q="+str);
    xhttp.send();
}
</script>
</body>
</html>
```

Figura 33: Ajax Database Example

CustomerID	NORTS
CompanyName	North/South
ContactName	Simon Crowther
Address	South House 300 Queensbridge
City	London
PostalCode	SW7 1RZ
Country	UK

Figura 34: Elegiendo una opción

6.1.9. AJAX - Applications

■ Mostrar datos XML en una tabla HTML

```

<!DOCTYPE html>
<html>
<style>
table,th,td {
    border : 1px solid black;
    border-collapse: collapse;
}
th,td {
    padding: 5px;
}
</style>
<body>

<button type="button" onclick="loadXMLDoc()">Get my CD collection</button>
<br><br>
<table id="demo"></table>

<script>
function loadXMLDoc() {
    const xhttp = new XMLHttpRequest();
    xhttp.onload = function() {
        const xmlDoc = xhttp.responseXML;
        const cd = xmlDoc.getElementsByTagName("CD");
        myFunction(cd);
    }
    xhttp.open("GET", "cd_catalog.xml");
    xhttp.send();
}

function myFunction(cd) {
    let table = "<tr><th>Artist</th><th>Title</th></tr>";
    for (let i = 0; i < cd.length; i++) {
        table += "<tr><td>" +
            cd[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
            "</td><td>" +
            cd[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +
            "</td></tr>";
    }
    document.getElementById("demo").innerHTML = table;
}
</script>
</body>
</html>

```

Figura 35: Estructura

Artist	Title
Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros
Bee Gees	One night only
Dr Hook	Sylvia's Mother
Rod Stewart	Maggie May
Andrea Bocelli	Romanza
Percy Sledge	When a man loves a woman
Savage Rose	Black angel
Many	1999 Grammy Nominees
Kenny Rogers	For the good times
Will Smith	Big Willie style
Van Morrison	Tupelo Honey
Joni Mitchell	Soulsville
Cat Stevens	The very best of
Sam Brown	Stop
TPau	Bridge of Spies
Tina Turner	Private Dancer
Kim Larsen	Midi om natten
Luciano Pavarotti	Pavarotti Gala Concert
Otis Redding	The dock of the bay
Simply Red	Picture book
The Communards	Red
Joe Cocker	Unchain my heart

Figura 36: Mostrar Datos

■ Mostrar el primer CD en un elemento div HTML

The screenshot shows a browser window with the following content:

```
<!DOCTYPE html>
<html>
<body>
<div id="showCD"></div>

<script>
const xhttp = new XMLHttpRequest();
xhttp.onload = function() {
    const xmlDoc = xhttp.responseXML;
    const cd = xmlDoc.getElementsByTagName("CD");
    myFunction(cd, 0);
}
xhttp.open("GET", "cd_catalog.xml");
xhttp.send();

function myFunction(cd, i) {
    document.getElementById("showCD").innerHTML =
    "Artist: " +
    cd[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
    "<br>Title: " +
    cd[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +
    "<br>Year: " +
    cd[i].getElementsByTagName("YEAR")[0].childNodes[0].nodeValue;
}
</script>
</body>
</html>
```

Result Size: 945 x 860 Get your own website

Artist: Bob Dylan
 Title: Empire Burlesque
 Year: 1985

Figura 37: Mostrar primer elemento

■ Navegar entre los CD

The screenshot shows a browser window with the following content:

```
<!DOCTYPE html>
<html>
<body>
<div id="showCD"></div><br>
<input type="button" onclick="previous()" value="<<" />
<input type="button" onclick="next()" value=">>" />

<script>
let i = 0;
let len;
let cd;
const xhttp = new XMLHttpRequest();
xhttp.onload = function() {
    const xmlDoc = xhttp.responseXML;
    cd = xmlDoc.getElementsByTagName("CD");
    len = cd.length;
    displayCD(i);
}
xhttp.open("GET", "cd_catalog.xml");
xhttp.send();

function displayCD(i) {
    document.getElementById("showCD").innerHTML =
    "Artist: " +
    cd[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
    "<br>Title: " +
    cd[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +
    "<br>Year: " +
    cd[i].getElementsByTagName("YEAR")[0].childNodes[0].nodeValue;
}

function next() {
    if (i < len-1) {
        i++;
        displayCD(i);
    }
}

function previous() {
    if (i > 0) {
        i--;
        displayCD(i);
    }
}
</script>
```

Result Size: 945 x 960 Get your own website

Artist: Bonnie Tyler
 Title: Hide your heart
 Year: 1988

<< >>

Figura 38: Navegar entre CD

■ Mostrar información del álbum al hacer clic en un CD

Artist	Title
Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros
Bee Gees	One night only
Dr Hook	Sylvia's Mother
Rod Stewart	Maggie May
Andrea Bocelli	Romanza
Percy Sledge	When a man loves a woman
Savage Rose	Black angel
Many	1999 Grammy Nominees
Kenny Rogers	For the good times
Will Smith	Big Willie style
Van Morrison	Tupelo Honey
Joni Mitchell	Soulsville
Cat Stevens	The very best of
Sam Brown	Stop
TPau	Bridge of Spies
Tina Turner	Private Dancer
Kim Larsen	Midi om natten
Luciano Pavarotti	Pavarotti Gala Concert
Otis Redding	The dock of the bay
Simply Red	Picture book
The Communards	Red
Joe Cocker	Unchain my heart

Figura 39: Mostrar tabla de CD

Artist	Title
Eros Ramazzotti	Eros
Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros
Bee Gees	One night only
Dr Hook	Sylvia's Mother
Rod Stewart	Maggie May
Andrea Bocelli	Romanza
Percy Sledge	When a man loves a woman
Savage Rose	Black angel
Many	1999 Grammy Nominees
Kenny Rogers	For the good times
Will Smith	Big Willie style
Van Morrison	Tupelo Honey
Joni Mitchell	Soulsville
Cat Stevens	The very best of
Sam Brown	Stop
TPau	Bridge of Spies
Tina Turner	Private Dancer
Kim Larsen	Midi om natten
Luciano Pavarotti	Pavarotti Gala Concert
Otis Redding	The dock of the bay
Simply Red	Picture book
The Communards	Red
Joe Cocker	Unchain my heart

Figura 40: Haciendo Click en un CD

6.1.10. AJAX - Examples

■ SIMPLES EXAMPLES

The screenshot shows a browser window with the following content:

```
<!DOCTYPE html>
<html>
<body>

<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>
```

AJAX

AJAX is not a programming language.
 AJAX is a technique for accessing web servers from a web page.
 AJAX stands for Asynchronous JavaScript And XML.

Figura 41: Ejemplo 1

The screenshot shows a browser window with the following content:

```
<!DOCTYPE html>
<html>
<body>

<div id="demo">
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc('ajax_info.txt', myFunction)">Change Content</button>
</div>

<script>
function loadDoc(url, xFunction) {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() (xFunction(this));
  xhttp.open("GET", url);
  xhttp.send();
}

function myFunction(xhttp) {
  document.getElementById("demo").innerHTML = xhttp.responseText;
}
</script>
</body>
</html>
```

AJAX

AJAX is not a programming language.
 AJAX is a technique for accessing web servers from a web page.
 AJAX stands for Asynchronous JavaScript And XML.

Figura 42: Ejemplo 2

■ Request Header Information

The screenshot shows a browser window with the title "The XMLHttpRequest Object". On the left, there is a code editor containing JavaScript code demonstrating how to use the XMLHttpRequest object to get response headers. On the right, the browser's developer tools or a similar interface displays the "Response Headers" section, listing various HTTP headers such as Content-Type, Content-Length, Date, and Server.

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<p>The getAllResponseHeaders() function returns all the header information of a resource, like length, server-type, content-type, last-modified, etc.</p>

<pre id="demo"></pre>

<script>
const xhttp = new XMLHttpRequest();
xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
        this.getAllResponseHeaders();
}
xhttp.open("GET", "ajax_info.txt");
xhttp.send();
</script>
</body>
</html>
```

Figura 43: Ejemplo 1

The screenshot shows a browser window with the title "The XMLHttpRequest Object". On the left, there is a code editor containing JavaScript code demonstrating how to use the XMLHttpRequest object to get specific response headers. On the right, the browser's developer tools or a similar interface displays the "Response Headers" section, specifically focusing on the "Last-Modified" header.

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<p>The getResponseHeader() function is used to return specific header information from a resource, like length, server-type, content-type, last-modified, etc.</p>

<p>Last modified: <span id="demo"></span></p>

<script>
const xhttp = new XMLHttpRequest();
xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
        this.getResponseHeader("Last-Modified");
}
xhttp.open("GET", "ajax_info.txt");
xhttp.send();
</script>
</body>
</html>
```

Figura 44: Ejemplo 2

■ Request XML Files

The screenshot shows a browser window with the following content:

```
<!DOCTYPE html>
<html>
<body>
<h2>The XMLHttpRequest Object</h2>
<h2>Retrieve data from XML file</h2>
<p><b>Status:</b> <span id="A1"></span></p>
<p><b>Status text:</b> <span id="A2"></span></p>
<p><b>Response:</b> <span id="A3"></span></p>
<button onclick="loadDoc('note.xml')>Get XML data</button>
<script>
function loadDoc(url) {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById('A1').innerHTML = this.status;
    document.getElementById('A2').innerHTML = this.statusText;
    document.getElementById('A3').innerHTML = this.responseText;
  }
  xhttp.open("GET", url);
  xhttp.send();
}
</script>
</body>
</html>
```

The right side of the browser shows the results of the script execution:

The XMLHttpRequest Object
Retrieve data from XML file
 Status: 200
 Status text:
 Response: Tove Jami Reminder Don't forget me this weekend!
 Get XML data

Figura 45: Ejemplo 1

The screenshot shows a browser window with the following content:

```
<!DOCTYPE html>
<html>
<style>
table, th, td {
  border : 1px solid black;
  border-collapse: collapse;
}
th, td {
  padding: 5px;
}
</style>
<body>
<h2>The XMLHttpRequest Object</h2>
<button type="button" onclick="loadDoc()">Get my CD collection</button>
<br><br>
<table id="demo"></table>
<script>
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    myFunction(this);
  }
  xhttp.open("GET", "cd_catalog.xml");
  xhttp.send();
}
function myFunction(xml) {
  const xmlDoc = xml.responseXML;
  const x = xmlDoc.getElementsByTagName("CD");
  let table="




```

The right side of the browser shows the resulting table:

The XMLHttpRequest Object
 Get my CD collection

Artist	Title
Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros
Bee Gees	One night only
Dr Hook	Sylvia's Mother
Rod Stewart	Maggie May
Andrea Bocelli	Romanza
Percy Sledge	When a man loves a woman
Savage Rose	Black angel
Many	1999 Grammy Nominees
Kenny Rogers	For the good times
Will Smith	Big Willie style
Van Morrison	Tupelo Honey
Jim Hoel	Soulsville
Cat Stevens	The very best of
Sam Brown	Stop
T'Pau	Bridge of Spies
Tina Turner	Private Dancer
Kim Larsen	Midt om natten
Luciano Pavarotti	Pavarotti Gala Concert
Otis Redding	The dock of the bay
Simply Red	Picture book
The Communards	Red

Figura 46: Ejemplo 2

■ Retrieve Server Data with PHP and ASP

The screenshot shows a browser window with a live coding interface. On the left, there is a code editor with the following JavaScript code:

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>
<h3>Start typing a name in the input field below:</h3>

<p>Suggestions: <span id="txtHint"></span></p>
<p>First name: <input type="text" id="txt1" onkeyup="showHint(this.value)"></p>

<script>
function showHint(str) {
  if (str.length == 0) {
    document.getElementById("txtHint").innerHTML = "";
    return;
  }
  const xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    document.getElementById("txtHint").innerHTML =
      this.responseText;
  }
  xhttp.open("GET", "gethint.php?q="+str);
  xhttp.send();
}
</script>
</body>
</html>
```

On the right, the browser displays the output of the code. It shows the title "The XMLHttpRequest Object" and a placeholder "Start typing a name in the input field below:". Below this, it says "Suggestions: Linda, Liza". A text input field labeled "First name" contains the letter "L".

Figura 47: Ejemplo 1

The screenshot shows a browser window with a live coding interface, similar to Figure 47. The code editor contains the same JavaScript code as Figure 47. The browser output on the right shows the title "The XMLHttpRequest Object", the placeholder "Start typing a name in the input field below:", suggestions "Suggestions: Kitty", and a text input field labeled "First name" containing the letter "K".

Figura 48: Ejemplo 2

■ Retrieve Database Information

CustomerID	WOLZA
CompanyName	Wolski Zajazd
ContactName	Zbigniek Pietrzenniewicz
Address	ul. Fibrova 68
City	Warszawa
PostalCode	01-012
Country	Poland

Figura 49: Ejemplo 1

■ Ajax Applications

```

<?xml version="1.0" encoding="UTF-8"?>
<CATALOG>
  <CD>
    <TITLE>Empire Burlesque</TITLE>
    <ARTIST>Bob Dylan</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>Columbia</COMPANY>
    <PRICE>10.90</PRICE>
    <YEAR>1985</YEAR>
  </CD>
  <CD>
    <TITLE>Hide your heart</TITLE>
    <ARTIST>Bonnie Tyler</ARTIST>
    <COUNTRY>UK</COUNTRY>
    <COMPANY>RCA</COMPANY>
    <PRICE>9.98</PRICE>
    <YEAR>1988</YEAR>
  </CD>
  <CD>
    <TITLE>Greatest Hits</TITLE>
    <ARTIST>Dolly Parton</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>RCA</COMPANY>
    <PRICE>10.20</PRICE>
    <YEAR>1982</YEAR>
  </CD>
  <CD>
    <TITLE>Still got the blues</TITLE>
    <ARTIST>Amy Moore</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>Virgin records</COMPANY>
    <PRICE>10.20</PRICE>
    <YEAR>1990</YEAR>
  </CD>
  <CD>
    <TITLE>Eros</TITLE>
    <ARTIST>Eros Ramazzotti</ARTIST>
    <COUNTRY>Italy</COUNTRY>
    <COMPANY>BMG</COMPANY>
    <PRICE>9.98</PRICE>
    <YEAR>1999</YEAR>
  </CD>
  <CD>
    <TITLE>One night only</TITLE>
    <ARTIST>Be Gees</ARTIST>
    <COUNTRY>UK</COUNTRY>
    <COMPANY>CBS</COMPANY>
    <PRICE>10.90</PRICE>
    <YEAR>1998</YEAR>
  </CD>
  <CD>
    <TITLE>Styletes</TITLE>
    <ARTIST>Depeche Mode</ARTIST>
    <COUNTRY>UK</COUNTRY>
    <COMPANY>CBS</COMPANY>
    <PRICE>10.90</PRICE>
    <YEAR>1973</YEAR>
  </CD>
</CATALOG>

```

Figura 50: XML

Artist	Title
Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros
Bee Gees	One night only
Dr Hook	Sylvias Mother
Rod Stewart	Maggie May
Andrea Bocelli	Romanza
Percy Sledge	When a man loves a woman
Savage Rose	Black angel
Many	1999 Grammy Nominees
Kenny Rogers	For the good times
Will Smith	Big Willie style
Van Morrison	Tupelo Honey
Jom Hoel	Soulsville
Car Stevens	The very best of
Sam Brown	Stop
TPan	Bridge of Spies
Tina Turner	Private Dancer
Kim Larsen	Midi om natten
Luciano Pavarotti	Pavarotti Gala Concert
Otis Redding	The dock of the bay
Simply Red	Picture book
The Communards	Red
Joe Cocker	Unchain my heart

```
<!DOCTYPE html>
<html>
<style>
table,th,td {
    border : 1px solid black;
    border-collapse: collapse;
}
th,td {
    padding: 5px;
}
</style>
<body>

<button type="button" onclick="loadXMLDoc()">Get my CD collection</button>
<br><br>
<table id="demo"></table>

<script>
function loadXMLDoc() {
    const xhttp = new XMLHttpRequest();
    xhttp.onload = function() {
        const xmlDoc = xhttp.responseXML;
        const cd = xmlDoc.getElementsByTagName("CD");
        myFunction(cd);
    }
    xhttp.open("GET", "cd_catalog.xml");
    xhttp.send();
}

function myFunction(cd) {
    let table=<t><tr><th>Artist</th><th>Title</th></tr></t>;
    for (let i = 0; i < cd.length; i++) {
        table += <t><tr><td>
            cd[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
            "<br>" +
            cd[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +
            "</td><td>" +
            cd[i].getElementsByTagName("YEAR")[0].childNodes[0].nodeValue;
        </td></tr>;
    }
    document.getElementById("demo").innerHTML = table;
}
</script>
</body>
</html>
```

Figura 51: Ejemplo 1

Artist: Bob Dylan Title: Empire Burlesque Year: 1985
--

```
<!DOCTYPE html>
<html>
<body>

<div id="showCD"></div>

<script>
const xhttp = new XMLHttpRequest();
xhttp.onload = function() {
    const xmlDoc = xhttp.responseXML;
    const cd = xmlDoc.getElementsByTagName("CD");
    myFunction(cd, 0);
}
xhttp.open("GET", "cd_catalog.xml");
xhttp.send();

function myFunction(cd, i) {
    document.getElementById("showCD").innerHTML =
    "Artist: " +
    cd[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
    "<br>Title: " +
    cd[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +
    "<br>Year: " +
    cd[i].getElementsByTagName("YEAR")[0].childNodes[0].nodeValue;
}
</script>
</body>
</html>
```

Figura 52: Ejemplo 2

The screenshot shows a browser developer tools console with the following details:

- Result Size:** 945 x 860
- Get your own website** button
- Run >** button
- Artist: Dolly Parton**
- Title: Greatest Hits**
- Year: 1982**
- << >>** navigation buttons
- Script Content:** The code block contains a script that creates an XMLHttpRequest object, sends a GET request to "cd_catalog.xml", and then iterates through the XML response to display CD information. It includes functions for displaying a CD, navigating to the next or previous CD, and handling the XML response.

```
<!DOCTYPE html>
<html>
<body>

<div id="showCD"></div><br>
<input type="button" onclick="previous()" value="<<">>
<input type="button" onclick="next()" value=">>">><br>

<script>
let i = 0;
let len;
let cd;

const xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if(xhttp.readyState == XMLHttpRequest.DONE) {
        const xmlDoc = xhttp.responseXML;
        cd = xmlDoc.getElementsByTagName("CD");
        len = cd.length;
        displayCD(i);
    }
}
xhttp.open("GET", "cd_catalog.xml");
xhttp.send();

function displayCD(i) {
    document.getElementById("showCD").innerHTML =
        "Artist: " + cd[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
        "<br>Title: " + cd[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +
        "<br>Year: " + cd[i].getElementsByTagName("YEAR")[0].childNodes[0].nodeValue;
}

function next() {
    if(i < len-1) {
        i++;
        displayCD(i);
    }
}

function previous() {
    if(i > 0) {
        i--;
        displayCD(i);
    }
}
</script>
```

Figura 53: Ejemplo 3

```
<!DOCTYPE html>
<html>

<head>
  <style>
    table, th, td {
      border : 1px solid black;
      border-collapse: collapse;
    }
    th,td {
      padding: 5px;
    }
  </style>
</head>
<body>



Click on a CD to display album information.</p>


</a>



<script>
const xhttp = new XMLHttpRequest();
let cd;
xhttp.onload = function() {
  const xmlDoc = xhttp.responseXML;
  cd = xmlDoc.getElementsByTagName("CD");
  loadCD();
}
xhttp.open("GET", "cd_catalog.xml");
xhttp.send();

function loadCD() {
  let table = "<tr><th>Artist</th><th>Title</th></tr>";
  for (let i = 0; i < cd.length; i++) {
    table += "<tr onclick='displayCD(" + i + ")'><td>";
    table += cd[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue;
    table += "</td><td>";
    table += cd[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue;
    table += "</td></tr>";
  }
  document.getElementById("demo").innerHTML = table;
}

function displayCD(i) {
  document.getElementById("showCD").innerHTML =
  "Artist: " +
  cd[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
  "<br>Title: " +
  cd[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue;
}
```

Figura 54: Ejemplo 4

6.2. Uso de Xampp

- Proporciona un entorno de servidor local que te permite desarrollar y probar aplicaciones web antes de publicarlas en un servidor web en producción. Usado para los ejercicios 4 - 8.

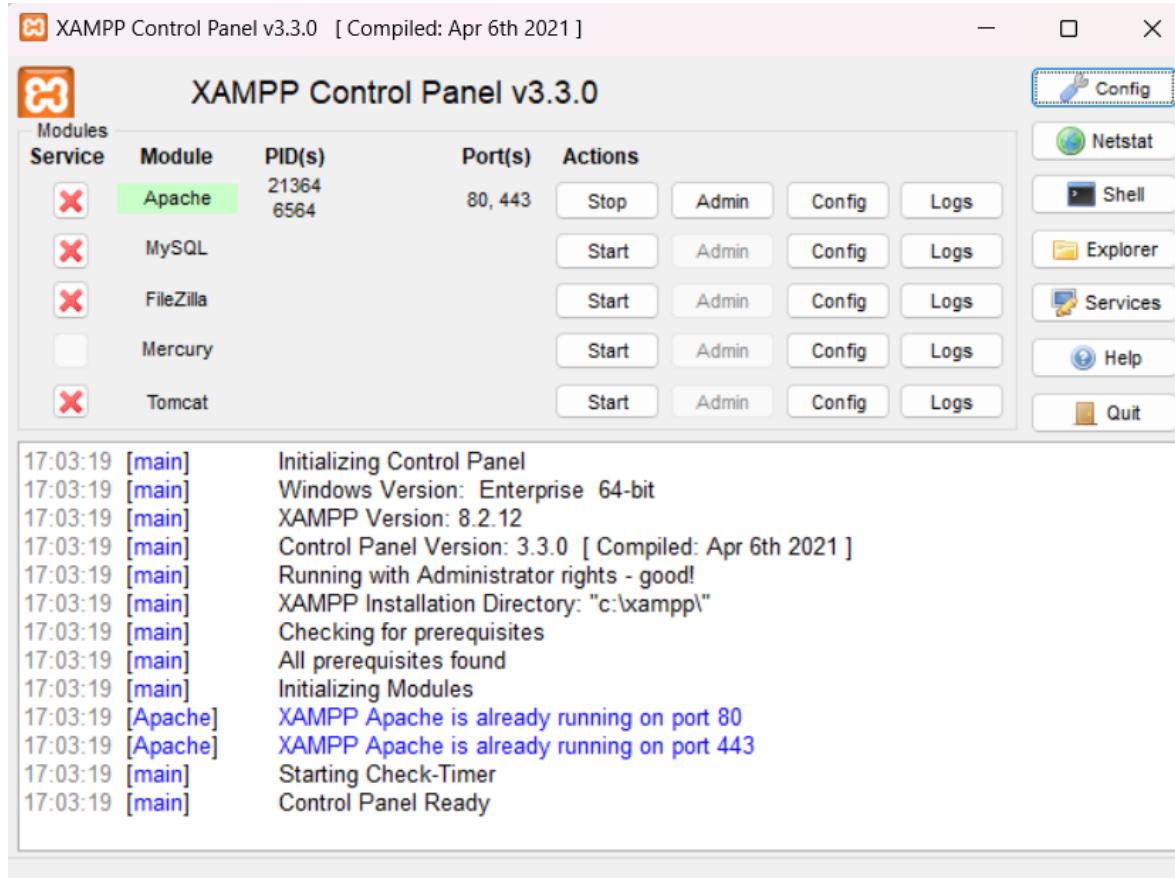


Figura 55: Xampp

6.3. Uso de Node.js

- Es una plataforma que permite ejecutar código JavaScript del lado del servidor, ofreciendo un entorno altamente escalable y eficiente para el desarrollo de aplicaciones web modernas. Usado para la página web Markdown y del ejercicio 1 - 3

6.3.1. Instalación

Al descargar el installer, solo seguir los pasos correspondientes.

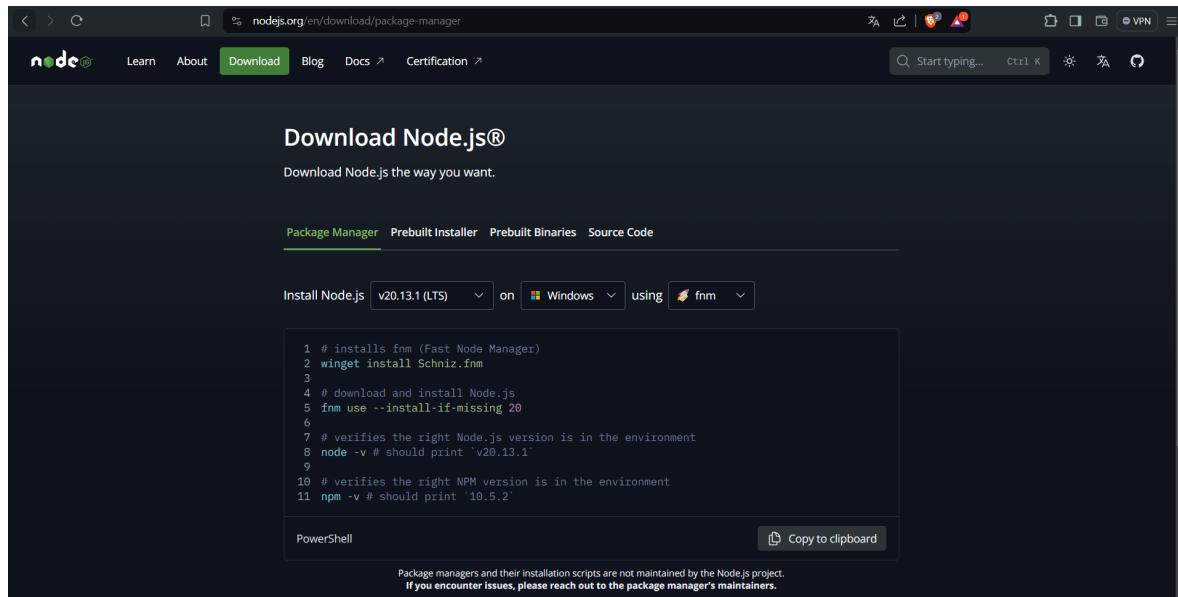


Figura 56: Instalación Node.js

6.4. Investigación sobre Chart.js

6.4.1. Página Oficial

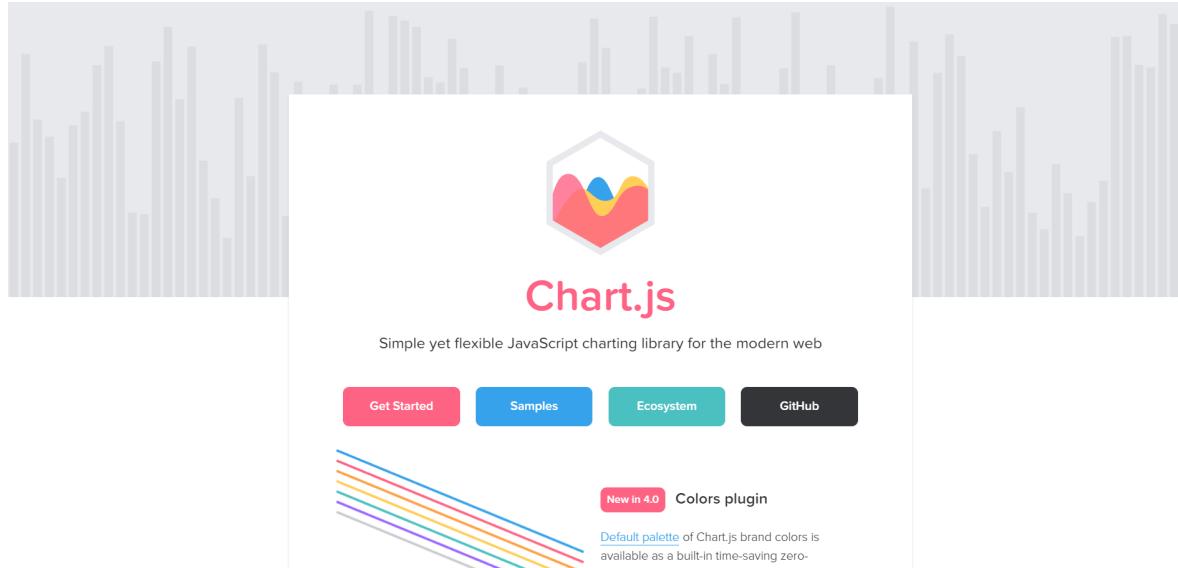


Figura 57: Chart.js

6.4.2. Documentación

The screenshot shows the official Chart.js documentation website. The left sidebar contains navigation links for Chart.js, Getting Started, General, Configuration, Chart Types, Axes, Developers, and Migration. The main content area features a heading 'Chart.js' and a sub-section '# Why Chart.js'. It discusses the popularity of Chart.js among charting libraries, its open-source status, and its features like built-in chart types and mixed charts.

Figura 58: Chart.js DOC

6.5. EJERCICIOS

6.5.1. Ejercicio 1

- textbfScript

```

1  import('node-fetch').then(async (module) => {
2      const fetch = module.default;
3      const response = await fetch('http://localhost/nuevo/JASON/data.json');
4      // Verifica si la respuesta fue exitosa
5      if (response.ok) {
6          const data = await response.json();
7          console.log("\t Lista de Regiones");
8          //Itera sobre los datos obtenidos e imprime las regiones
9          for (let i = 0; i < data.length; i++) {
10              console.log(data[i].region);
11          }
12      } else {
13          console.error('Error en la solicitud:', response.status);
14      }
15      //Captura de error
16  }).catch(error => {
17      console.error('Error:', error);
18  });

```

Figura 59: Script 1

■ Ejecución

```
NppExec Console
Current directory: C:\xampp\htdocs\Nuevo\Ejercicio1
node Ejercicio1.js
Process started (PID=21780) >>>
    Lista de Regiones
Amazonas
Ancash
Apurímac
Arequipa
Ayacucho
Cajamarca
Callao
Cusco
Huancavelica
Huanuco
Ica
Junin
La Libertad
Lambayeque
Lima
Loreto
Madre de Dios
Moquegua
Pasco
Piura
Puno
San Martin
Tacna
Tumbes
Ucayali
<<< Process finished (PID=21780). (Exit code 0)
===== READY =====
```

Figura 60: Ejercicio 1

6.5.2. Ejercicio 2

■ Script

```
1 import('node-fetch').then(async (module) => {
2     const fetch = module.default;
3     const response = await fetch('http://localhost/nuevo/JASON/data.json');
4     if (response.ok) {
5         const data = await response.json();
6         console.log("\t Lista de Regiones");
7         //Itera sobre los datos obtenidos e imprime la ultima confirmación dada osea la mayor
8         for (let i = 0; i < data.length; i++) {
9             let confirmados = data[i].confirmed;
10            let region = data[i].region;
11            let totalConfirmados = confirmados[confirmados.length - 1].value;
12            console.log(region + ": " + totalConfirmados);
13        }
14    } else {
15        console.error('Error en la solicitud:', response.status);
16    }
17    //Captura de error
18 }).catch(error => {
19     console.error('Error:', error);
20});
```

Figura 61: Script 2

■ Ejecución

```
NppExec Console
CD: C:\xampp\htdocs\Nuevo\Ejercicio2
Current directory: C:\xampp\htdocs\Nuevo\Ejercicio2
node Ejercicio2.js
Process started (PID=8440) >>>
    Lista de Regiones
Amazonas: 237
Ancash: 1507
Apurímac: 93
Arequipa: 1046
Ayacucho: 254
Cajamarca: 362
Callao: 5360
Cusco: 407
Huancavelica: 193
Huanuco: 333
Ica: 1003
Junin: 798
La Libertad: 1449
Lambayeque: 3655
Lima: 43284
Loreto: 1780
Madre de Dios: 162
Moquegua: 167
Pasco: 167
Piura: 2425
Puno: 166
San Martin: 466
Tacna: 178
Tumbes: 486
Ucayali: 1329
<<< Process finished (PID=8440). (Exit code 0)
===== READY =====
```

Figura 62: Ejercicio 2

6.5.3. Ejercicio 3

- Script

```

1  import('node-fetch').then(async (module) => {
2      const fetch = module.default;
3      const response = await fetch('http://localhost/nuevo/JASON/data.json');
4      let mapa = new Map();
5      if (response.ok) {
6          const data = await response.json();
7          console.log("\t Lista de Regiones");
8          //Aregar datos tanto suma como region a un mapa
9          for (let i = 0; i < data.length; i++) {
10              let confirmados = data[i].confirmed;
11              let region = data[i].region;
12              let suma = 0;
13              for(let c = 0; c < confirmados.length; c++){
14                  suma += parseInt(confirmados[c].value);
15              }
16              mapa.set(region,suma);
17          }
18          //convertir mapa en arreglo
19          let arreglo = Array.from(mapa);
20          //Ordenar arreglo
21          arreglo.sort((a, b) => b[1] - a[1]);
22          let ordenamiento = new Map(arreglo);
23          let i = 0;
24          //Imprimir datos del mapa
25          for (let [clave, valor] of ordenamiento) {
26              console.log((i + 1) + " " + clave + ":" + valor);
27              i++;
28              if(i >= 10){
29                  break;
30              }
31          }
32      } else {
33          console.error('Error en la solicitud:', response.status);
34      }
35  }).catch(error => {
36      console.error('Error:', error);
37  });

```

Figura 63: Script 3

- Ejecución

```
NppExec Console
CD: C:\xampp\htdocs\Nuevo\Ejercicio3
Current directory: C:\xampp\htdocs\Nuevo\Ejercicio3
node Ejercicio3.js
Process started (PID=6192) >>>
    Lista de Regiones
1 Lima: 626744
2 Callao: 78099
3 Lambayeque: 51206
4 Loreto: 30242
5 Piura: 29966
6 Ancash: 18753
7 La Libertad: 18175
8 Ucayali: 14488
9 Arequipa: 13817
10 Ica: 11925
<<< Process finished (PID=6192). (Exit code 0)
===== READY =====
```

Figura 64: Ejercicio 3

6.5.4. Ejercicio 4

- HTML

```
1  <!DOCTYPE HTML>
2  <html lang="es">
3  |   <head>
4  |   |       <title>Ejercicio 4</title>
5  |   |       <meta charset="UTF-8"/>
6  |   |       <meta name="author" content="Victor Gonzalo Maldonado Vilca"/>
7  |   |       <meta name="description" content="Usando ajax para la comparación de gráficas"/>
8  |   |       <meta name="keywords" content="comparación, ajax, gráfica"/>
9  |   |       <link rel="stylesheet" href="http://localhost/nuevo/css/estilos.css">
10 |   |       <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
11 |   </head>
12 |   <body class="cuerpo">
13 |   |       <div>
14 |   |           <form id="comparasion">
15 |   |               <center>
16 |   |                   <button class="Generar" type="submit">Generar</button>
17 |   |               </center>
18 |   |           </div>
19 |   |           <div>
20 |   |               <canvas id="grafica" width="800" height="650"></canvas>
21 |   |           </div>
22 |   |           <script src="Script.js"></script>
23 |   </body>
24 </html>
```

Figura 65: HTML

■ Script

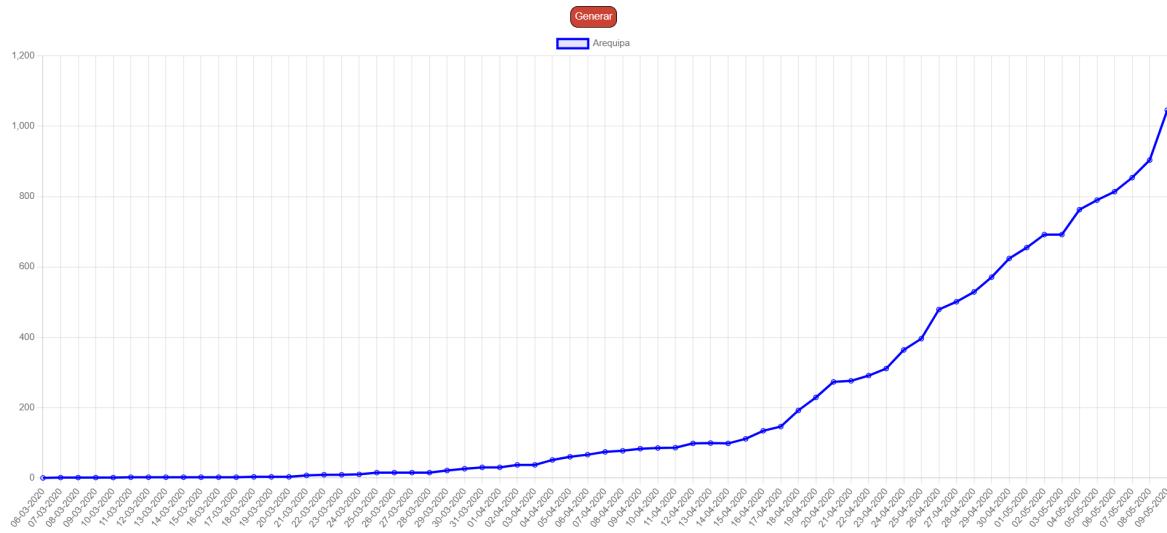
```

1  document.getElementById("comparasion").addEventListener("submit", function (evento) {
2      evento.preventDefault();
3
4      const region1 = 'Apurímac';
5      const region2 = 'Lima';
6
7      fetch("http://localhost/nuevo/JASON/data.json")
8          .then(response => response.json())
9          .then(data => {
10              //Solicitando información del archivo data.json
11              const datosRegion1 = data.find(region => region.region === region1);
12              const datosRegion2 = data.find(region => region.region === region2);
13
14              const fechas = datosRegion1.confirmed.map(entry => entry.date);
15              const valoresRegion1 = datosRegion1.confirmed.map(entry => parseInt(entry.value));
16              const valoresRegion2 = datosRegion2.confirmed.map(entry => parseInt(entry.value));
17              //caso contrario la crea
18              const ctx = document.getElementById('grafica').getContext('2d');
19              myChart = new Chart(ctx, {
20                  type: 'line',
21                  data: {
22                      labels: fechas,
23                      datasets: [
24                          {
25                              label: region1,
26                              data: valoresRegion1,
27                              borderColor: 'blue',
28                              backgroundColor: 'rgba(0, 0, 255, 0.1)'
29                          },
29                          {
30                              label: region2,
31                              data: valoresRegion2,
32                              borderColor: 'green',
33                              backgroundColor: 'rgba(0, 255, 0, 0.1)'
34                          }
35                      ]
36                  },
37                  options: {
38                      responsive: true,
39                      maintainAspectRatio: false
40                  }
41              });
42          })
43      .catch(error => console.error('Error:', error));
44  });
45 });

```

Figura 66: Script 4

■ Ejecución



- Script

```

1  document.getElementById("comparasion").addEventListener("submit", function (evento) {
2      evento.preventDefault();
3
4      const region1 = 'Apurímac';
5      const region2 = 'Lima';
6
7      fetch("http://localhost/nuevo/JASON/data.json")
8          .then(response => response.json())
9          .then(data => {
10             //Solicitando información del archivo data.json
11             const datosRegion1 = data.find(region => region.region === region1);
12             const datosRegion2 = data.find(region => region.region === region2);
13
14             const fechas = datosRegion1.confirmed.map(entry => entry.date);
15             const valoresRegion1 = datosRegion1.confirmed.map(entry => parseInt(entry.value));
16             const valoresRegion2 = datosRegion2.confirmed.map(entry => parseInt(entry.value));
17             //caso contrario la crea
18             const ctx = document.getElementById('grafica').getContext('2d');
19             myChart = new Chart(ctx, {
20                 type: 'line',
21                 data: {
22                     labels: fechas,
23                     datasets: [
24                         {
25                             label: region1,
26                             data: valoresRegion1,
27                             borderColor: 'blue',
28                             backgroundColor: 'rgba(0, 0, 255, 0.1)'
29                         },
30                         {
31                             label: region2,
32                             data: valoresRegion2,
33                             borderColor: 'green',
34                             backgroundColor: 'rgba(0, 255, 0, 0.1)'
35                         }
36                     ],
37                     options: {
38                         responsive: true,
39                         maintainAspectRatio: false
40                     }
41                 });
42             });
43         });
44     .catch(error => console.error('Error:', error));
45 });

```

Figura 69: Script 5 - 1

- Ejecución

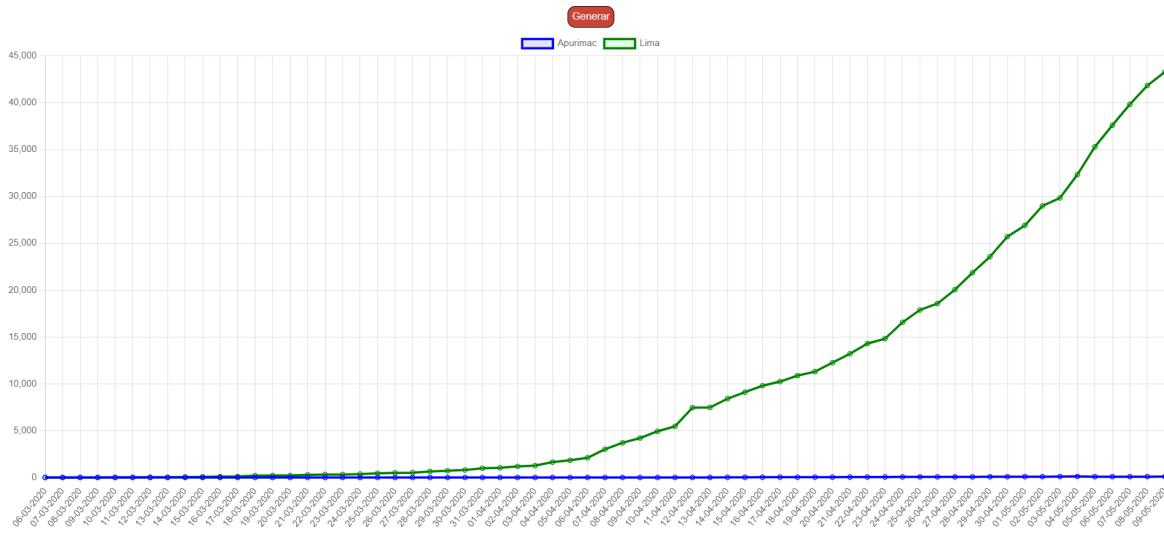


Figura 70: Ejercicio 5 - 1

- comparasion 2

- HTML

```

1   <!DOCTYPE HTML>
2   <html lang="es">
3   <head>
4       <title>Ejercicio 5</title>
5       <meta charset="UTF-8"/>
6       <meta name="author" content="Victor Gonzalo Maldonado Vilca"/>
7       <meta name="description" content="Usando ajax para la comparación de gráficas"/>
8       <meta name="keywords" content="comparación, ajax, gráfica"/>
9       <link rel="stylesheet" href="http://localhost/nuevo/css/estilos.css">
10      <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
11  </head>
12 <body class="cuerpo">
13 <div>
14     <form id="comparasion">
15         <center>
16             <button class="Generar" type="submit">Generar</button>
17         </center>
18     </div>
19     <div>
20         <canvas id="grafica" width="800" height="650"></canvas>
21     </div>
22     <script src="Script.js"></script>
23 </body>
24 </html>
```

Figura 71: HTML 5

- Script

```

1  document.getElementById("comparasion").addEventListener("submit", function (evento) {
2      evento.preventDefault();
3
4      const region1 = 'Ancash';
5      const region2 = 'Callao';
6
7      fetch("http://localhost/nuevo/JASON/data.json")
8          .then(response => response.json())
9          .then(data => {
10              //Solicitando informacion del archivo data.json
11              const datosRegion1 = data.find(region => region.region === region1);
12              const datosRegion2 = data.find(region => region.region === region2);
13
14              const fechas = datosRegion1.confirmed.map(entry => entry.date);
15              const valoresRegion1 = datosRegion1.confirmed.map(entry => parseInt(entry.value));
16              const valoresRegion2 = datosRegion2.confirmed.map(entry => parseInt(entry.value));
17              //caso contrario la crea
18              const ctx = document.getElementById('grafica').getContext('2d');
19              myChart = new Chart(ctx, {
20                  type: 'line',
21                  data: {
22                      labels: fechas,
23                      datasets: [
24                          {
25                              label: region1,
26                              data: valoresRegion1,
27                              borderColor: 'blue',
28                              backgroundColor: 'rgba(0, 0, 255, 0.1)'
29                          },
30                          {
31                              label: region2,
32                              data: valoresRegion2,
33                              borderColor: 'green',
34                              backgroundColor: 'rgba(0, 255, 0, 0.1)'
35                          }
36                      ]
37                  },
38                  options: {
39                      responsive: true,
40                      maintainAspectRatio: false
41                  }
42              });
43          })
44      .catch(error => console.error('Error:', error));
45  });

```

Figura 72: Script 5 - 2

- Ejecución

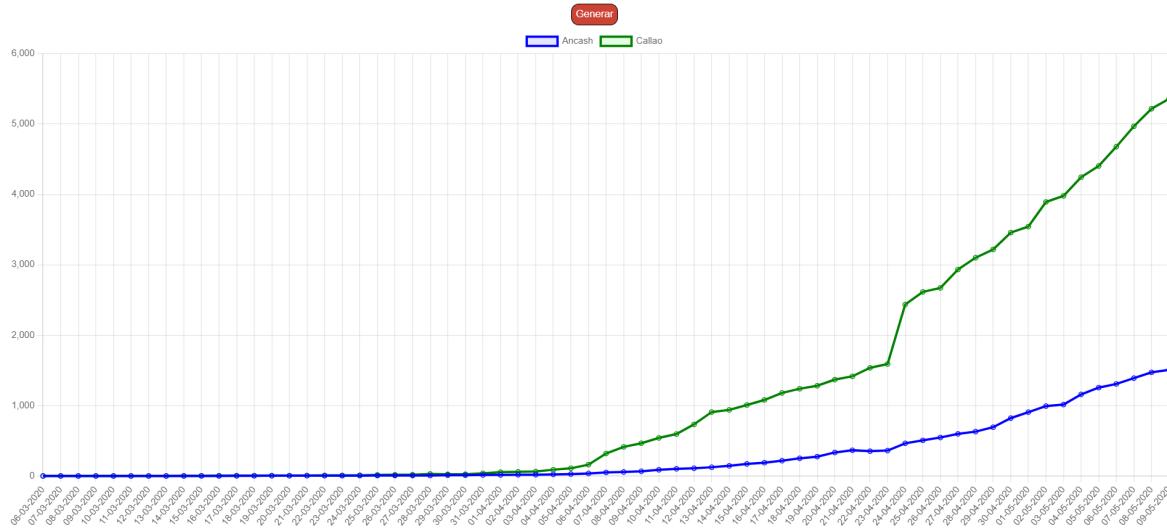


Figura 73: Ejercicio 5 - 2

- comparasion 3

- HTML

```

1   <!DOCTYPE HTML>
2   <html lang="es">
3   <head>
4       <title>Ejercicio 5</title>
5       <meta charset="UTF-8"/>
6       <meta name="author" content="Victor Gonzalo Maldonado Vilca"/>
7       <meta name="description" content="Usando ajax para la comparación de gráficas"/>
8       <meta name="keywords" content="comparación, ajax, gráfica"/>
9       <link rel="stylesheet" href="http://localhost/nuevo/css/estilos.css">
10      <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
11  </head>
12  <body class="cuerpo">
13      <div>
14          <form id="comparasion">
15              <center>
16                  <button class="Generar" type="submit">Generar</button>
17              </center>
18          </div>
19          <div>
20              <canvas id="grafica" width="800" height="650"></canvas>
21          </div>
22          <script src="Script.js"></script>
23      </body>
24  </html>
```

Figura 74: HTML 5

- Script

```

1  document.getElementById("comparasion").addEventListener("submit", function (evento) {
2      evento.preventDefault();
3
4      const region1 = 'Tacna';
5      const region2 = 'Cusco';
6
7      fetch("http://localhost/nuevo/JASON/data.json")
8          .then(response => response.json())
9          .then(data => {
10              //Solicitando informacion del archivo data.json
11              const datosRegion1 = data.find(region => region.region === region1);
12              const datosRegion2 = data.find(region => region.region === region2);
13
14              const fechas = datosRegion1.confirmed.map(entry => entry.date);
15              const valoresRegion1 = datosRegion1.confirmed.map(entry => parseInt(entry.value));
16              const valoresRegion2 = datosRegion2.confirmed.map(entry => parseInt(entry.value));
17              //caso contrario la crea
18              const ctx = document.getElementById('grafica').getContext('2d');
19              myChart = new Chart(ctx, {
20                  type: 'line',
21                  data: {
22                      labels: fechas,
23                      datasets: [
24                          {
25                              label: region1,
26                              data: valoresRegion1,
27                              borderColor: 'blue',
28                              backgroundColor: 'rgba(0, 0, 255, 0.1)'
29                          },
30                          {
31                              label: region2,
32                              data: valoresRegion2,
33                              borderColor: 'green',
34                              backgroundColor: 'rgba(0, 255, 0, 0.1)'
35                          }
36                      ],
37                  },
38                  options: {
39                      responsive: true,
40                      maintainAspectRatio: false
41                  }
42              });
43          })
44          .catch(error => console.error('Error:', error));
45      });
}

```

Figura 75: Script 5 - 3

- Ejecución

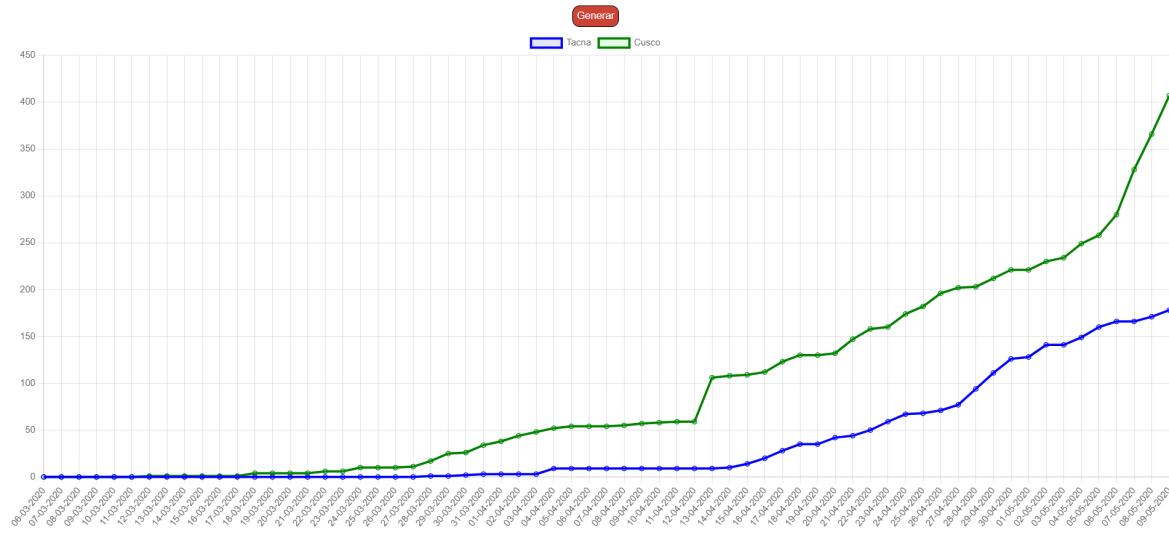


Figura 76: Ejercicio 5 - 3

- comparasion 4

- HTML

```

1   <!DOCTYPE HTML>
2   <html lang="es">
3       <head>
4           <title>Ejercicio 5</title>
5           <meta charset="UTF-8"/>
6           <meta name="author" content="Victor Gonzalo Maldonado Vilca"/>
7           <meta name="description" content="Usando ajax para la comparación de gráficas"/>
8           <meta name="keywords" content="comparación, ajax, gráfica"/>
9           <link rel="stylesheet" href="http://localhost/nuevo/css/estilos.css">
10          <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
11      </head>
12      <body class="cuerpo">
13          <div>
14              <form id="comparasion">
15                  <center>
16                      <button class="Generar" type="submit">Generar</button>
17                  </center>
18              </div>
19              <div>
20                  <canvas id="grafica" width="800" height="650"></canvas>
21              </div>
22              <script src="Script.js"></script>
23          </body>
24      </html>
```

Figura 77: HTML 5

- Script

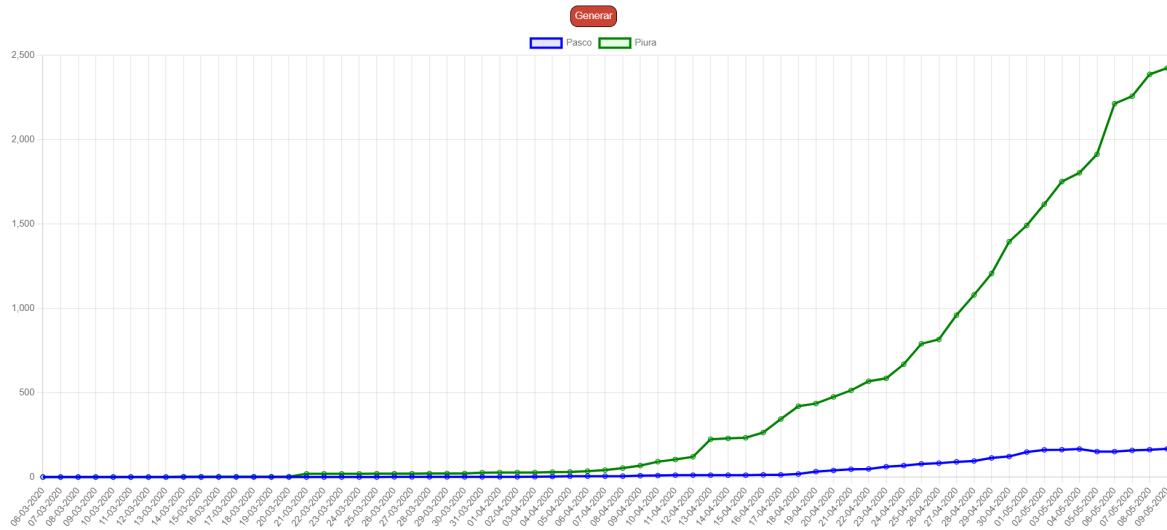
```

1  document.getElementById("comparasion").addEventListener("submit", function (evento) {
2      evento.preventDefault();
3
4      const region1 = 'Pasco';
5      const region2 = 'Piura';
6
7      fetch("http://localhost/nuevo/JASON/data.json")
8          .then(response => response.json())
9          .then(data => {
10              //Solicitando informacion del archivo data.json
11              const datosRegion1 = data.find(region => region.region === region1);
12              const datosRegion2 = data.find(region => region.region === region2);
13
14              const fechas = datosRegion1.confirmed.map(entry => entry.date);
15              const valoresRegion1 = datosRegion1.confirmed.map(entry => parseInt(entry.value));
16              const valoresRegion2 = datosRegion2.confirmed.map(entry => parseInt(entry.value));
17              //caso contrario la crea
18              const ctx = document.getElementById('grafica').getContext('2d');
19              myChart = new Chart(ctx, {
20                  type: 'line',
21                  data: {
22                      labels: fechas,
23                      datasets: [
24                          {
25                              label: region1,
26                              data: valoresRegion1,
27                              borderColor: 'blue',
28                              backgroundColor: 'rgba(0, 0, 255, 0.1)'
29                          },
30                          {
31                              label: region2,
32                              data: valoresRegion2,
33                              borderColor: 'green',
34                              backgroundColor: 'rgba(0, 255, 0, 0.1)'
35                          }
36                      ]
37                  },
38                  options: {
39                      responsive: true,
40                      maintainAspectRatio: false
41                  }
42              });
43          })
44          .catch(error => console.error('Error:', error));
45      });

```

Figura 78: Script 5 - 4

- Ejecución



■ Script

```

1  document.getElementById("comparasion").addEventListener("submit", function (evento) {
2      evento.preventDefault();
3
4      fetch("http://localhost/nuevo/JASON/data.json")
5          .then(response => response.json())
6          .then(data => {
7              //extrayendo primera región
8              const primeraRegion = data.find(region => region.region !== 'Lima' && region.region !== 'Callao');
9
10             // Verificar que se encontró la primera región
11             if (!primeraRegion) {
12                 console.error('No se encontró la primera región con las fechas necesarias para el gráfico.');
13                 return;
14             }
15
16             // Obtener las fechas de la primera región
17             const fechas = primeraRegion.confirmed.map(entry => entry.date);
18
19             // Extraer datos de las regiones menos Lima y Callao
20             const datasets = data.filter(region => region.region !== 'Lima' && region.region !== 'Callao').map(region => {
21                 const valores = [];
22                 // Llenar los valores con los datos correspondientes a las fechas de la primera región
23                 fechas.forEach(fecha => {
24                     const dato = region.confirmed.find(entry => entry.date === fecha);
25                     if (dato) {
26                         valores.push(parseInt(dato.value));
27                     } else {
28                         valores.push(null);
29                     }
30                 });
31                 return {
32                     label: region.region,
33                     data: valores,
34                     borderColor: getRandomColor(),
35                     backgroundColor: 'rgba(0, 0, 255, 0.1)',
36                     pointStyle: 'none',
37                     pointRadius: 0
38                 };
39             });

```

Figura 81: Script 6 - 1

```

40
41         // Crear el gráfico con los datos de todas las regiones
42         const ctx = document.getElementById('grafica').getContext('2d');
43         myChart = new Chart(ctx, {
44             type: 'line',
45             data: {
46                 labels: fechas,
47                 datasets: datasets
48             },
49             options: {
50                 responsive: true,
51                 maintainAspectRatio: false,
52                 scales: {
53                     y: {
54                         beginAtZero: true
55                     }
56                 }
57             });
58         })
59         .catch(error => console.error('Error:', error));
60     });
61
62
63     // Función para obtener un color aleatorio
64     function getRandomColor() {
65         const letters = '0123456789ABCDEF';
66         let color = '#';
67         for (let i = 0; i < 6; i++) {
68             color += letters[Math.floor(Math.random() * 16)];
69         }
70         return color;
71     }

```

Figura 82: Script 6 - 2

■ Ejecución

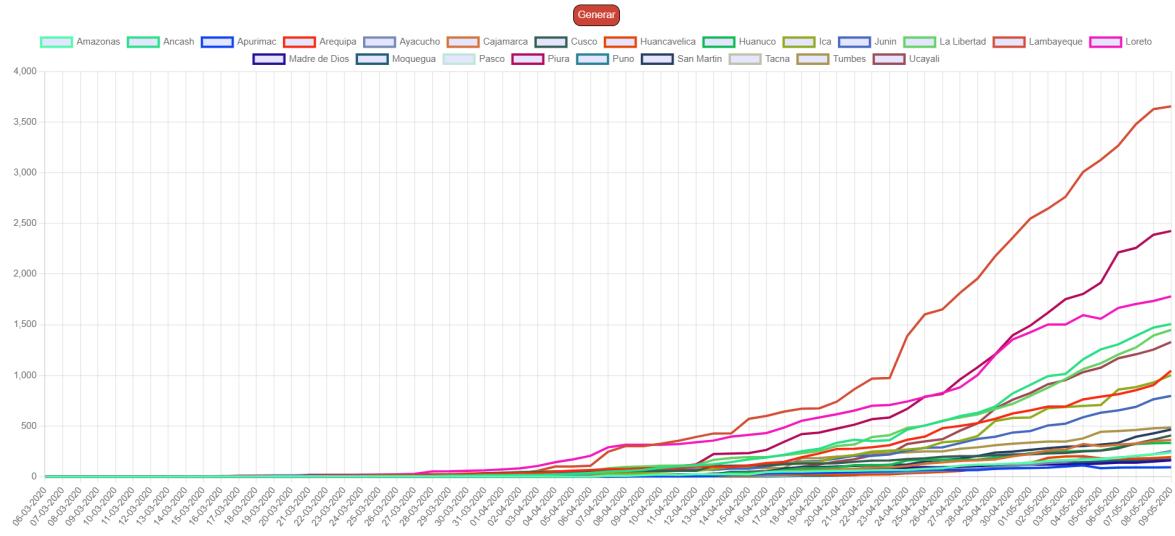


Figura 83: Ejercicio 6

6.5.7. Ejercicio 7

- **HTML**

```

1      <!DOCTYPE HTML>
2      <html lang="es">
3          <head>
4              <title>Ejercicio 7</title>
5              <meta charset="UTF-8"/>
6              <meta name="author" content="Victor Gonzalo Maldonado Vilca"/>
7              <meta name="description" content="Usando ajax para la comparación de gráficas"/>
8              <meta name="keywords" content="comparación, ajax, gráfica"/>
9              <link rel="stylesheet" href="http://localhost/nuevo/css/estilos.css">
10             <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
11         </head>
12         <body class="cuerpo">
13             <div>
14                 <form id="comparasion">
15                     <center>
16                         <label for="region1">Región 1:</label>
17                         <select id="region1" name="region1">
18                             <option value="Amazonas">Amazonas</option>
19                             <option value="Ancash">Ancash</option>
20                             <option value="Apurímac">Apurímac</option>
21                             <option value="Arequipa">Arequipa</option>
22                             <option value="Ayacucho">Ayacucho</option>
23                             <option value="Cajamarca">Cajamarca</option>
24                             <option value="Callao">Callao</option>
25                             <option value="Cusco">Cusco</option>
26                             <option value="Huancavelica">Huancavelica</option>
27                             <option value="Huanuco">Huánuco</option>
28                             <option value="Ica">Ica</option>
29                             <option value="Junin">Junín</option>
30                             <option value="La Libertad">La Libertad</option>
31                             <option value="Lambayeque">Lambayeque</option>
32                             <option value="Lima">Lima</option>
33                             <option value="Loreto">Loreto</option>
34                             <option value="Madre de Dios">Madre de Dios</option>
35                             <option value="Moquegua">Moquegua</option>
36                             <option value="Pasco">Pasco</option>
37                             <option value="Piura">Piura</option>
38                             <option value="Puno">Puno</option>
39                             <option value="San Martin">San Martín</option>
40                             <option value="Tacna">Tacna</option>
41                             <option value="Tumbes">Tumbes</option>
42                             <option value="Ucayali">Ucayali</option>
43                         </select>

```

Figura 84: HTML 7 - 1

```

44         <label for="region2"> Región 2:</label>
45         <select id="region2" name="region2">
46             <option value="Amazonas">Amazonas</option>
47             <option value="Ancash">Ancash</option>
48             <option value="Apurímac">Apurímac</option>
49             <option value="Arequipa">Arequipa</option>
50             <option value="Ayacucho">Ayacucho</option>
51             <option value="Cajamarca">Cajamarca</option>
52             <option value="Callao">Callao</option>
53             <option value="Cusco">Cusco</option>
54             <option value="Huancavelica">Huancavelica</option>
55             <option value="Huánuco">Huánuco</option>
56             <option value="Ica">Ica</option>
57             <option value="Junín">Junín</option>
58             <option value="La Libertad">La Libertad</option>
59             <option value="Lambayeque">Lambayeque</option>
60             <option value="Lima">Lima</option>
61             <option value="Loreto">Loreto</option>
62             <option value="Madre de Dios">Madre de Dios</option>
63             <option value="Moquegua">Moquegua</option>
64             <option value="Pasco">Pasco</option>
65             <option value="Piura">Piura</option>
66             <option value="Puno">Puno</option>
67             <option value="San Martín">San Martín</option>
68             <option value="Tacna">Tacna</option>
69             <option value="Tumbes">Tumbes</option>
70             <option value="Ucayali">Ucayali</option>
71         </select>
72         <button class="Generar" type="submit">Generar</button>
73     </center>
74 </form>
75 </div>
76 <div>
77     <canvas id="grafica" width="800" height="650"></canvas>
78 </div>
79 <script src="Script.js"></script>
80 </body>
81 </html>
```

Figura 85: HTML 7 - 2

- Script

```

1  let myChart;
2  document.getElementById("comparasion").addEventListener("submit", function (evento) {
3      evento.preventDefault();
4      //Extraccion de datos del formulario
5      const formData = new FormData(this);
6      const region1 = formData.get('region1');
7      const region2 = formData.get('region2');
8
9      fetch("http://localhost/nuevo/JASON/data.json")
10         .then(response => response.json())
11         .then(data => {
12             //Solicitando informacion del archivo data.json
13             const datosRegion1 = data.find(region => region.region === region1);
14             const datosRegion2 = data.find(region => region.region === region2);
15
16             const fechas = datosRegion1.confirmed.map(entry => entry.date);
17             const valoresRegion1 = datosRegion1.confirmed.map(entry => parseInt(entry.value));
18             const valoresRegion2 = datosRegion2.confirmed.map(entry => parseInt(entry.value));
19             if (myChart) {
20                 //Si el archivo ya fue creado.
21                 myChart.data.labels = fechas;
22                 myChart.data.datasets[0].label = region1;
23                 myChart.data.datasets[0].data = valoresRegion1;
24                 myChart.data.datasets[1].label = region2;
25                 myChart.data.datasets[1].data = valoresRegion2;
26                 myChart.update();
27             } else {
28                 //caso contrario la crea
29                 const ctx = document.getElementById('grafica').getContext('2d');
30                 myChart = new Chart(ctx, {
31                     type: 'line',
32                     data: {
33                         labels: fechas,

```

Figura 86: Script 7-1

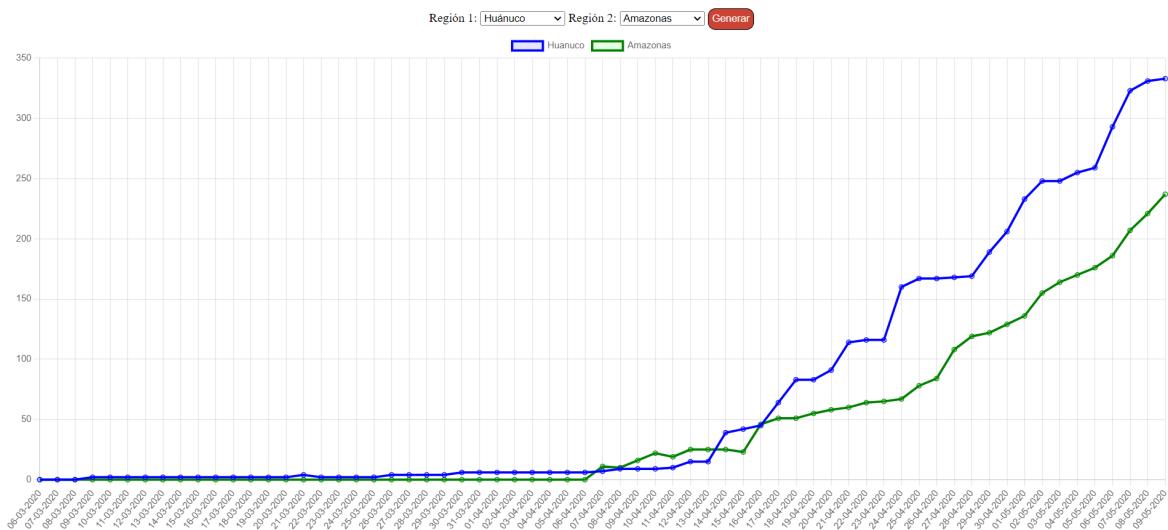
```

34     datasets: [
35       {
36         label: region1,
37         data: valoresRegion1,
38         borderColor: 'blue',
39         backgroundColor: 'rgba(0, 0, 255, 0.1)'
40       },
41       {
42         label: region2,
43         data: valoresRegion2,
44         borderColor: 'green',
45         backgroundColor: 'rgba(0, 255, 0, 0.1)'
46       }
47     ],
48   },
49   options: {
50     responsive: true,
51     maintainAspectRatio: false
52   }
53 });
54 })
55 .catch(error => console.error('Error:', error));
56 });
57 });

```

Figura 87: Script 7 - 2

■ Ejecución



6.5.8. Ejercicio 8

- **HTML**

```

1   <!DOCTYPE HTML>
2   <html lang="es">
3     <head>
4       <title>Ejercicio 8</title>
5       <meta charset="UTF-8"/>
6       <meta name="author" content="Victor Gonzalo Maldonado Vilca"/>
7       <meta name="description" content="Usando ajax para la comparación de gráficas"/>
8       <meta name="keywords" content="comparación, ajax, gráfica"/>
9       <link rel="stylesheet" href="http://localhost/nuevo/css/estilos.css">
10      <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
11    </head>
12    <body class="cuerpo">
13      <div>
14        <form id="comparasion">
15          <center>
16            <button class="Generar" type="submit">Generar</button>
17          </center>
18        </div>
19        <div>
20          <canvas id="grafica" width="800" height="650"></canvas>
21        </div>
22        <script src="Script.js"></script>
23      </body>
24    </html>

```

Figura 89: HTML 8

- **Script**

```

1   document.getElementById("comparasion").addEventListener("submit", function (evento) {
2     evento.preventDefault();
3
4     fetch("http://localhost/nuevo/JSON/data.json")
5       .then(response => response.json())
6       .then(data => {
7         // Extraer la primera región
8         const primeraRegion = data.find(region => region.region !== 'Lima' && region.region !== 'Callao');
9
10        // Verificar que se encontró la primera región
11        if (!primeraRegion) {
12          console.error('No se encontró la primera región con las fechas necesarias para el gráfico.');
13          return;
14        }
15
16        // Obtener las fechas de la primera región
17        const fechas = primeraRegion.confirmed.map(entry => entry.date);
18
19        // Extraer datos de las regiones menos Lima y Callao
20        const datasets = data.filter(region => region.region !== 'Lima' && region.region !== 'Callao').map(region => {
21          const valores = [];
22          // Llenar los valores con los datos correspondientes a las fechas de la primera región
23          fechas.forEach(fecha => {
24            const dato = region.confirmed.find(entry => entry.date === fecha);
25            if (dato) {
26              valores.push(parseInt(dato.value));
27            } else {
28              valores.push(null);
29            }
30          });
31          return {
32            label: region.region,
33            data: valores,
34            borderColor: getRandomColor(),
35            backgroundColor: 'rgba(0, 0, 255, 0.1)'
36          };
37        });

```

Figura 90: Script 8 - 1

```

39         // Crear el gráfico con los datos de todas las regiones
40         const ctx = document.getElementById('grafica').getContext('2d');
41         myChart = new Chart(ctx, {
42             type: 'line',
43             data: {
44                 labels: fechas,
45                 datasets: datasets
46             },
47             options: {
48                 responsive: true,
49                 maintainAspectRatio: false,
50                 scales: {
51                     y: {
52                         beginAtZero: true
53                     }
54                 }
55             });
56         })
57     )
58     .catch(error => console.error('Error:', error));
59 });
60
61     // Función para obtener un color aleatorio
62     function getRandomColor() {
63         const letters = '0123456789ABCDEF';
64         let color = '#';
65         for (let i = 0; i < 6; i++) {
66             color += letters[Math.floor(Math.random() * 16)];
67         }
68         return color;
69     }

```

Figura 91: Script 8 - 2

■ Ejecución

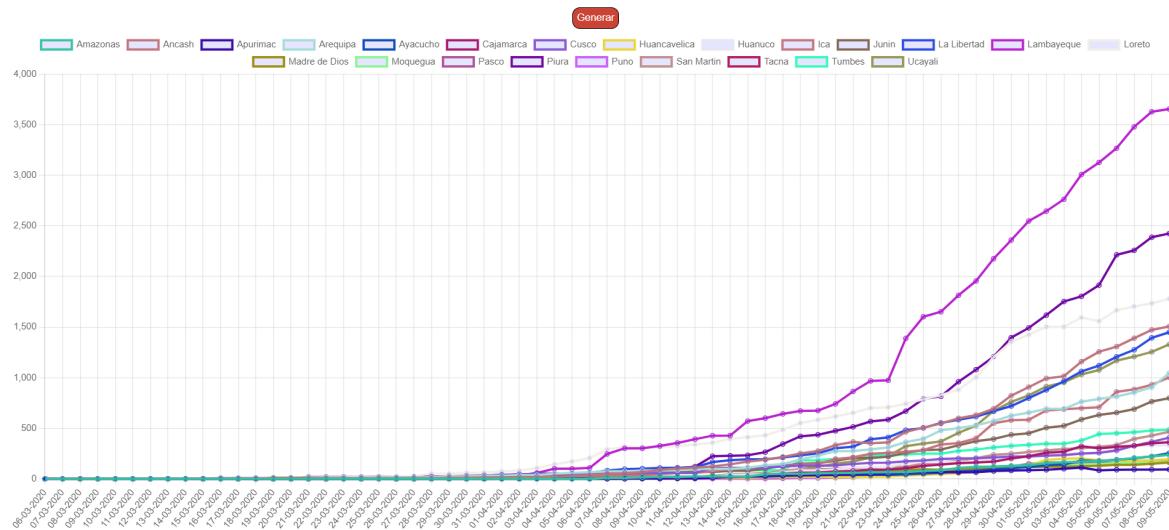


Figura 92: Ejercicio 8

6.5.9. CSS

■ CSS

```

1  .Generar{
2      border-radius: 10px;
3      background: #CB4335;
4      padding: 5px;
5      color: white;
6      border: 1px solid black;
7      margin-bottom: 5px;
8  }
9  .Generar:hover{
10     background: #EC7063;
11 }

```

Figura 93: Estilos ejercicio 4 - 8

6.6. Ejercicio Propuesto: Página Web - Markdown

- **HTML**

```

1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <link rel="stylesheet" href="css/estilos.css">
7      <title>Markdown</title>
8  </head>
9  <body>
10     <h1>MARKDOWN</h1>
11     <form id="markupForm">
12         <label for="markupTitle">Ingrese título</label><br>
13         <input type="text" id="markupTitle" name="markupTitle"/><br>
14         <label for="markupText">Ingrese el texto</label><br>
15         <textarea id="markupText" name="markupText" rows="10" cols="50"></textarea><br>
16         <input class="botones" type="submit" value="Enviar">
17     </form>
18     <form id="markupList">
19         <input class="botones" type="submit" value="Listar">
20     </form>
21     <h2>Resultado:</h2>
22     <div id="htmlCode"></div>
23     <script src="app.js"></script>
24   </body>
25 </html>

```

Figura 94: HTML

- **Script 1**

```

1  const fs = require('fs');
2  const path = require('path');
3  const express = require('express');
4  const bp = require('body-parser');
5  const MarkdownIt = require('markdown-it');
6  const md = new MarkdownIt();
7  const app = express();
8
9  let archivosGuardados = [];
10 app.use(express.static('pub'));
11 app.use(bp.json());
12 app.use(bp.urlencoded({ extended: true }));
13
14 app.listen(3000, () => {
15     console.log("Escuchando en: http://localhost:3000");
16 });
17
18 app.get('/', (request, response) => {
19     response.sendFile(path.resolve(__dirname, 'index.html'));
20 });
21
22 app.post('/crear', (request, response) => {
23     console.log(request.body);
24     const markDownText = request.body.text;
25     const titulo = request.body.title;
26     let htmlCode = md.render(markDownText);
27     response.setHeader('Content-Type', 'application/json');
28     response.end(JSON.stringify({ text: htmlCode, title: "TITULO: " + titulo }));
29     let archivo = {
30         nombre: titulo,
31         text: markDownText,
32         html: htmlCode,
33         fecha: new Date().toISOString()
34     };
35     archivosGuardados.push(archivo);
36     fs.writeFile('pub/archivos/Markdown.json', JSON.stringify(archivosGuardados), (err) => {
37         if (err) {
38             console.error('Error al guardar el archivo JSON:', err);
39         } else {
40             console.log('Información de archivos guardados actualizada en JSON.');
41         }
42     });
43 });
44 app.get('/listar', (request, response) => {
45     response.setHeader('Content-Type', 'application/json');
46     response.end(JSON.stringify(archivosGuardados.map(archivo => archivo.nombre)));
47 });

```

Figura 95: Script index.js 1

```

1  [-]function crear(markupText, markupTitle) {
2      const url = 'http://localhost:3000/crear'
3      const data = {
4          title: markupTitle,
5          text: markupText
6      }
7      console.log(data);
8      const request = {
9          method: 'POST',
10         headers: {
11             'Content-Type': 'application/json',
12         },
13         body: JSON.stringify(data),
14     }
15     fetch(url, request)
16     .then(response => response.json())
17     .then(data => {
18         document.querySelector("#htmlCode").innerHTML = "<h1>" + data.title + "</h1><br>" + data.text;
19     })
20     .catch(error => {
21         console.error('Error al enviar la solicitud:', error);
22     });
23 }
24 //listar todas los nombres markdown en un boton
25 [-]function listar(){
26     const url = 'http://localhost:3000/listar';
27     fetch(url)
28     .then(response => response.json())
29     .then(data => {
30         const archivosList = document.querySelector("#htmlCode");
31         archivosList.innerHTML = '';
32         data.forEach(nombre => {
33             const button = document.createElement('button');
34             button.textContent = nombre;
35             button.classList.add('botones');
36             button.addEventListener('click', () => {
37                 mostrarHtml(nombre);
38             });
39             archivosList.appendChild(button);
40             const saltoLinea = document.createElement('br');
41             archivosList.appendChild(saltoLinea);
42         });
43     })
44     .catch(error => {
45         console.error('Error al obtener la lista de archivos:', error);
46     });
47 }
```

Figura 96: Script app.js 2 - 1

```

48      //funcion para colocar el contenido html al presionar el boton generado en la funcion listar()
49      function mostrarHtml(nombreArchivo) {
50          const url = 'http://localhost:3000/archivos/Markdown.json'; // Ruta del archivo JSON
51          fetch(url)
52              .then(response => response.json())
53              .then(data => {
54                  const archivo = data.find(archivo => archivo.nombre === nombreArchivo);
55                  if (archivo) {
56                      const htmlCode = document.querySelector("#htmlCode");
57                      htmlCode.innerHTML = `<h1>${archivo.nombre}</h1><br>${archivo.html}`;
58                  } else {
59                      console.error('Archivo no encontrado en el JSON.');
60                  }
61              })
62              .catch(error => {
63                  console.error('Error al obtener el JSON de archivos:', error);
64              });
65      }
66      //Espera a que la página se cargue por completa
67      document.addEventListener('DOMContentLoaded', function () {
68          const title = document.querySelector('#markupTitle');
69          const text = document.querySelector('#markupText');
70          //botón para colocar en el archivo data.json, los datos respectivos
71          document.querySelector('#markupForm').onsubmit = () => {
72              event.preventDefault();
73              crear(text.value, title.value);
74              return false;
75          }
76          //botón para generar la lista
77          document.querySelector('#markupList').onsubmit = () => {
78              event.preventDefault();
79              listar();
80              return false;
81          }
82      })

```

Figura 97: Script app.js 2 - 2

■ Ejecución

```

NppExec Console

CD: C:\xampp\htdocs\Nuevo\Aplicacion_Web
Current directory: C:\xampp\htdocs\Nuevo\Aplicacion_Web
node index.js
Process started (PID=19872) >>>
Escuchando en: http://localhost:3000

```

Figura 98: Servidor Activado

MARKDOWN

Ingrese título

Ingrese el texto

Resultado:

TITULO: Saludo

Hola Mundo

Figura 99: Botón Enviar

MARKDOWN

Ingrese titulo

Ingrese el texto

Enviar

Listar

Resultado:

Saludo

kilo

limoes

Figura 100: Botón Listar

MARKDOWN

Ingrese título

Ingrese el texto

Enviar

Listar

Resultado:

Saludo

Hola Mundo

Figura 101: Mostrar página de la Lista

NppExec Console

```
CD: C:\xampp\htdocs\Nuevo\Aplicacion_Web
Current directory: C:\xampp\htdocs\Nuevo\Aplicacion_Web
node index.js
Process started (PID=19872) >>>
Escuchando en: http://localhost:3000
{ title: 'Saludo', text: '# Hola Mundo' }
Informaci | n de archivos guardados actualizada en JSON.
{ title: 'kilo', text: '# Hola Mundo' }
Informaci | n de archivos guardados actualizada en JSON.
{ title: 'limoes', text: '# Hola Mundo' }
Informaci | n de archivos guardados actualizada en JSON.
```

Figura 102: console.log

■ CSS

```
1  .botones{
2      border-radius: 10px;
3      background: #CB4335;
4      padding: 5px;
5      color: white;
6      border: 1px solid black;
7      margin-bottom: 5px;
8  }
9  .botones:hover{
10     background: #EC7063;
11 }
```

Figura 103: Estilos de Página Markdown

6.7. Uso de GitHub

6.7.1. Cuenta de GitHub

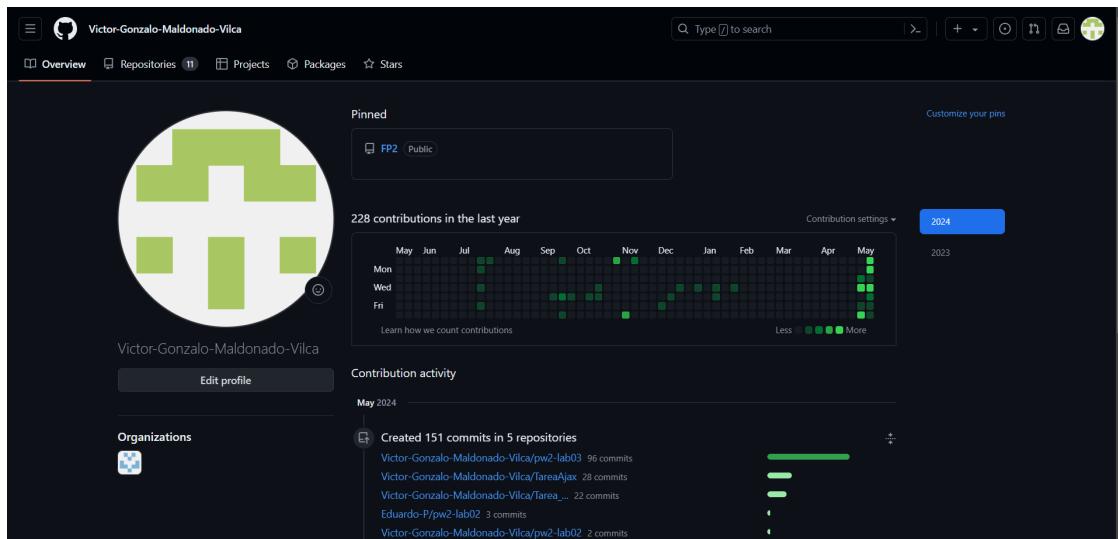


Figura 104: Usuario

6.8. Creación de un Nuevo Repositorio

6.8.1. Implementación de Readme.md

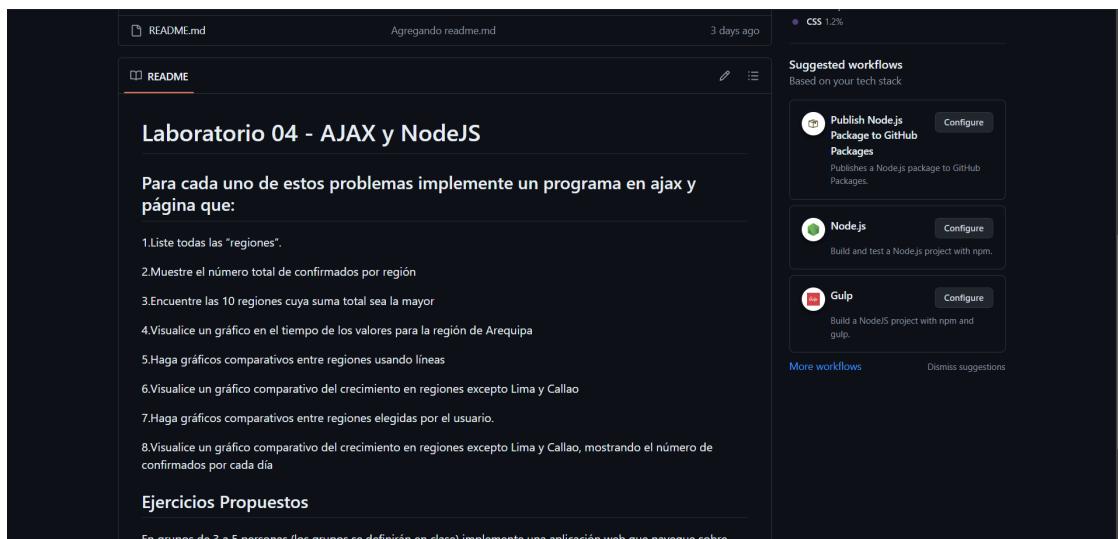


Figura 105: Readme.md

6.8.2. Registro de cambios en mi código

■ Commits

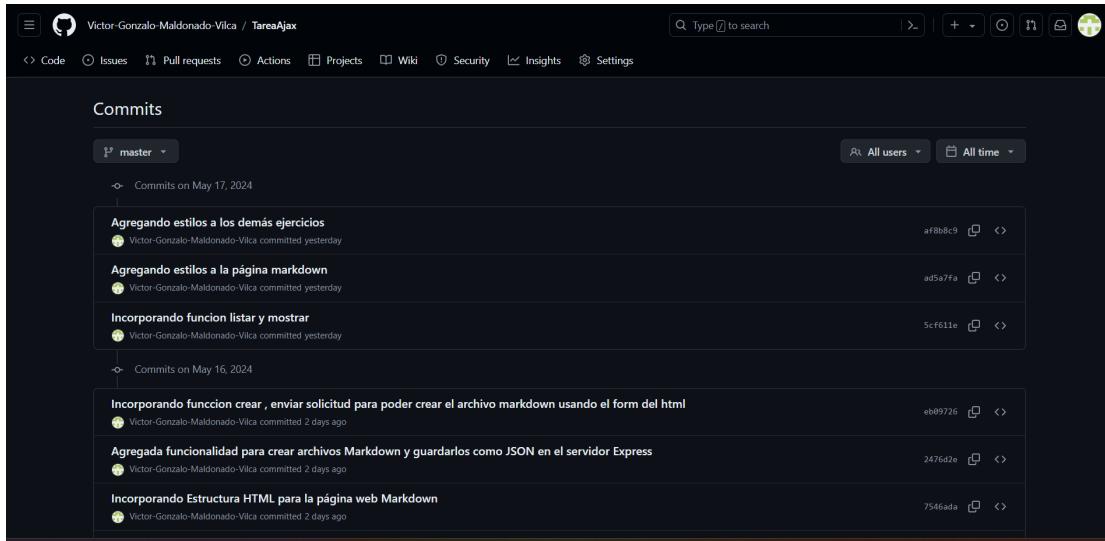


Figura 106: Commits

6.8.3. Repositorio

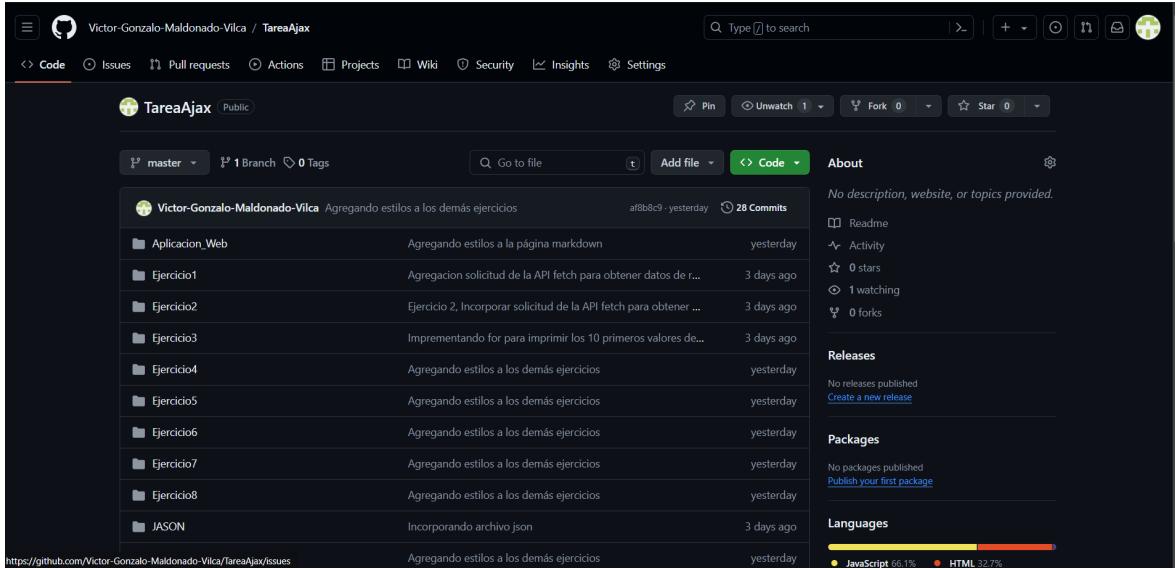


Figura 107: Repositorio

6.8.4. Proyecto compartido con el profesor de github

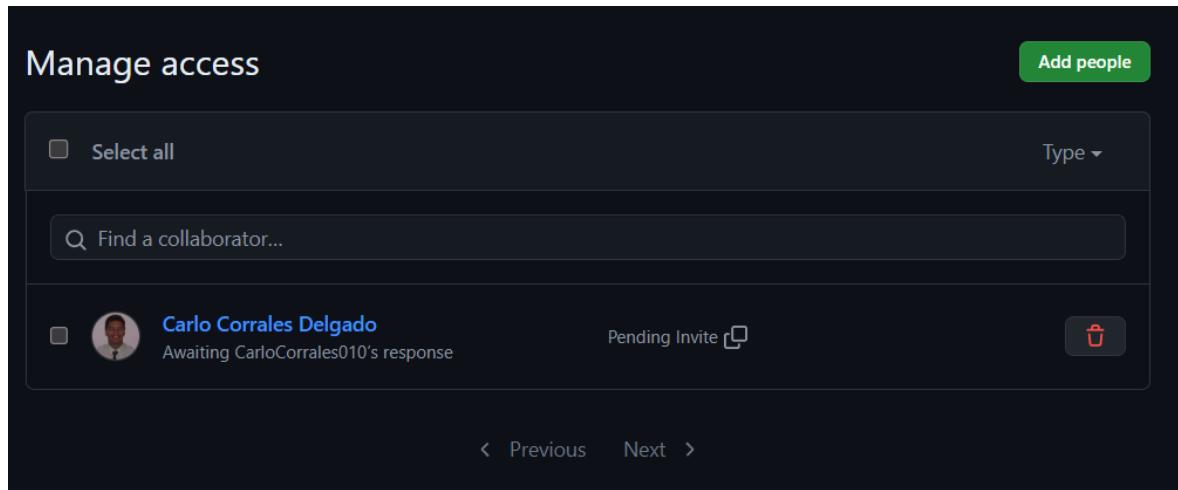


Figura 108: Compartir con el Profesor

6.9. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplió con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos los ítems.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 1: Niveles de desempeño

Puntos	Nivel			
	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 2: Rúbrica para contenido del Informe y demostración

	Contenido y demostración	Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	4	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	4	
Total		20		20	

7. Referencias

- <https://www.chartjs.org/>
- <https://www.chartjs.org/docs/latest/>
- <https://git-scm.com/doc>
- <https://www.w3schools.com/js/default.asp>