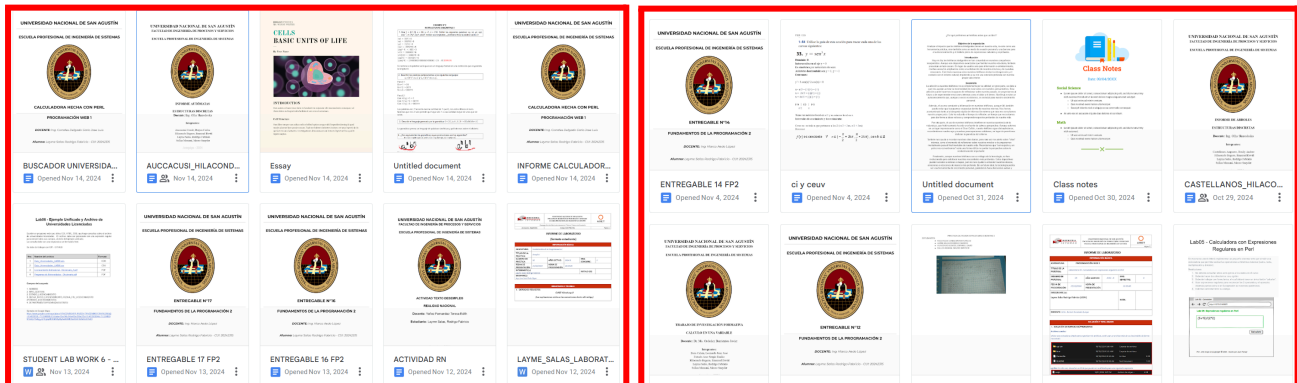
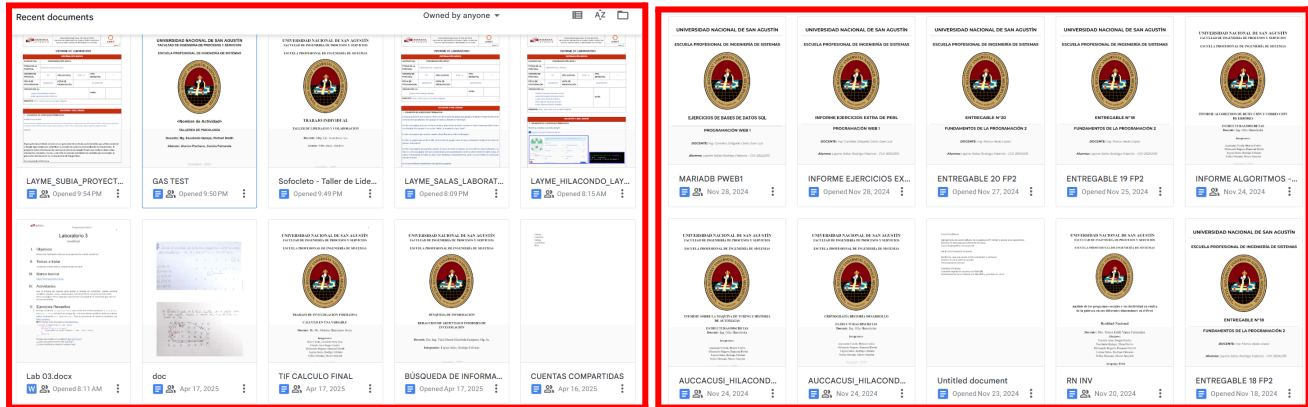


INFORME DE LABORATORIO

INFORMACIÓN BÁSICA					
ASIGNATURA:	PROGRAMACIÓN WEB 2				
TÍTULO DE LA PRÁCTICA:	<i>Aplicación Google App Script</i>				
NÚMERO DE PRÁCTICA:	<i>01</i>	AÑO LECTIVO:	<i>2025 - A</i>	NRO. SEMESTRE:	<i>I</i>
FECHA DE PRESENTACIÓN	<i>18/04/2025</i>	HORA DE PRESENTACIÓN	<i>23:58:00 PM</i>		
INTEGRANTE (s): <ul style="list-style-type: none"> - <i>Layme Salas Rodrigo Fabricio</i> - <i>Subia Huaicane Edson Fabricio</i> 				NOTA:	
DOCENTE: <i>M.Sc. Carlo Jose Luis Corrales Delgado</i>					

SOLUCIÓN Y RESULTADOS
I. SOLUCIÓN DE EJERCICIOS/PROBLEMAS LINK DEL PROYECTO EN GITHUB: https://github.com/F4brici0L4yme/google-app-script-Unsa USUARIOS DE GIT DEL EQUIPO: <ul style="list-style-type: none"> - F4brici0L4yme - Q3son Problema encontrado <i>Son muchos los trabajos que debemos de hacer en Google Docs, desde el primer año hasta el último. Pasa que cada año se nos deja como tarea presentar estos documentos ya sea en TIFs, actividades, algún informe, examen, etc. Y no son pocos los cursos que quieren un documento presentable, con los estándares de la universidad y su característica carátula al inicio de los trabajos. Solo mire mi historial de Docs:</i>



Podrá notar la cantidad de carátulas que uno debe hacer para presentar sus trabajos. Son unos 5 minutos que podría tomar crear un carátula, esto incluye escribir el texto, buscar el nombre completo del profesor, buscar sus grados académicos, escribir lugar y año... Como universitarios hay que seguir ese estándar. ¿Cómo podemos ahorrar este tiempo que tardamos en construir una carátula?

Solución

La solución estuvo cerca de nosotros, en una de las opciones del UI de Google. Google App Script, HTML, CSS y JavaScript nos permitirán automatizar este proceso. Podemos crear un add-on que genere un cuadro para poder personalizar nuestra carátula y crearla en un solo click. Google App Script puede conectarse a Sheets para recolectar datos, JavaScript puede mostrarlos y con HTML y CSS podemos crear una interfaz amigable. A continuación mostraremos las especificaciones de la solución.

Funcionamiento del Sistema:

1. **Google Sheets:** Se utilizó Google Sheets para almacenar la información necesaria para crear las carátulas. En la hoja de cálculo se guardan las siguientes categorías:
 - **Estudiantes:** Los nombres completos de los estudiantes que pueden ser seleccionados al generar la carátula.

- *Docentes:* Los nombres de los docentes junto con sus grados académicos (por ejemplo, Mg. o Dr.), que también se incluyen en el proceso.
 - *Cursos:* Los títulos de los cursos, como "Redacción de Artículos", "Ciudadanía e Interculturalidad", etc.
2. **Google Apps Script:** Se desarrolló un script en Google Apps Script que conecta la hoja de cálculo con Google Docs. El script permite extraer la información desde Google Sheets y utilizarla para llenar una plantilla de Google Docs con los datos proporcionados por el usuario.
3. **Interfaz Web (HTML/JavaScript):** Se diseñó un formulario web utilizando HTML y CSS donde los usuarios (estudiantes y docentes) pueden elegir opciones desde listas desplegables. Las opciones disponibles en las listas (como estudiantes, docentes y cursos) se generan dinámicamente al acceder al formulario. Este formulario está vinculado al script de Google Apps que, al ejecutar el generador, llena la carátula con los datos seleccionados.
- **Listas Desplegables:** La interfaz de usuario incluye listas desplegables para que los usuarios puedan elegir de manera fácil y rápida los estudiantes, docentes y cursos. Estas listas son alimentadas directamente desde Google Sheets (Conectadas al backEnd con ayuda de **JavaScript**) y cada vez que se ejecuta el generador, se actualizan de acuerdo con la información disponible en la hoja.
4. **Google Docs:** Al momento de ejecutar el script desde el formulario, la información seleccionada se transfiere a un nuevo documento de Google Docs. El script se encarga de insertar los datos (nombre del estudiante, nombre del docente, curso y otros detalles) en las ubicaciones correspondientes dentro de una plantilla de carátula prediseñada.
5. **Generación Automática de la Carátula:** Una vez que los usuarios completan el formulario y seleccionan los datos de las listas desplegables, el script genera automáticamente una carátula en Google Docs, que incluye los datos correspondientes. La carátula es completamente personalizable según las preferencias del usuario.

Programa en acción

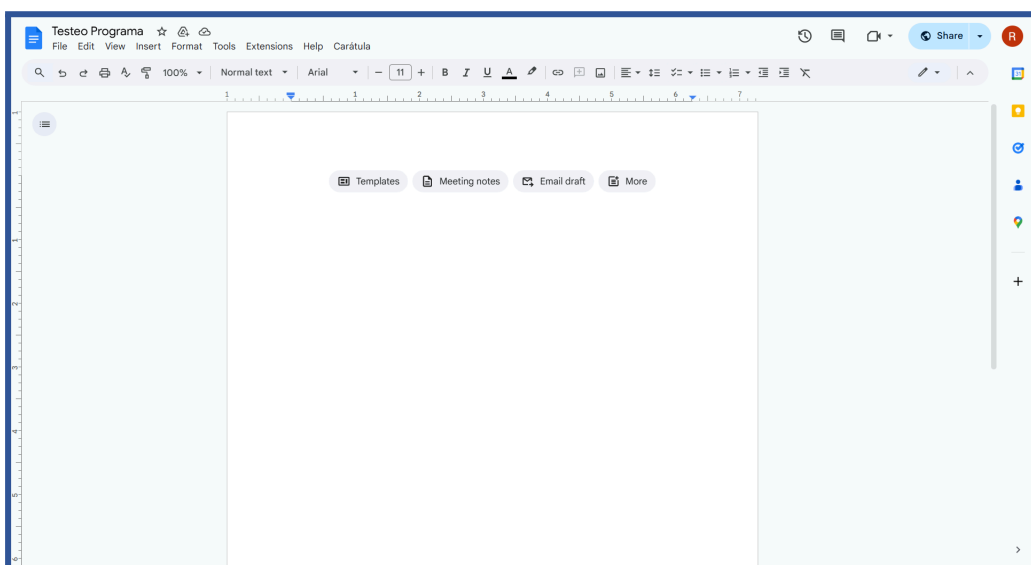


Fig. 1 Google Docs con una hoja en blanco

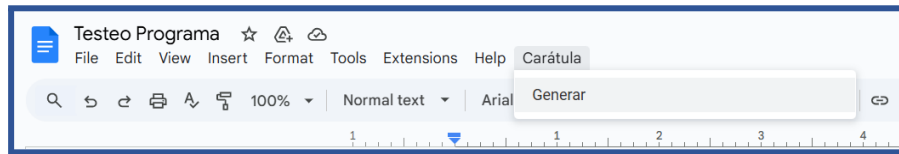


Fig. 2 El UI de Google Docs, se aprecia la creación de un ítem 'Carátula' en el menú y su opción 'Generar'

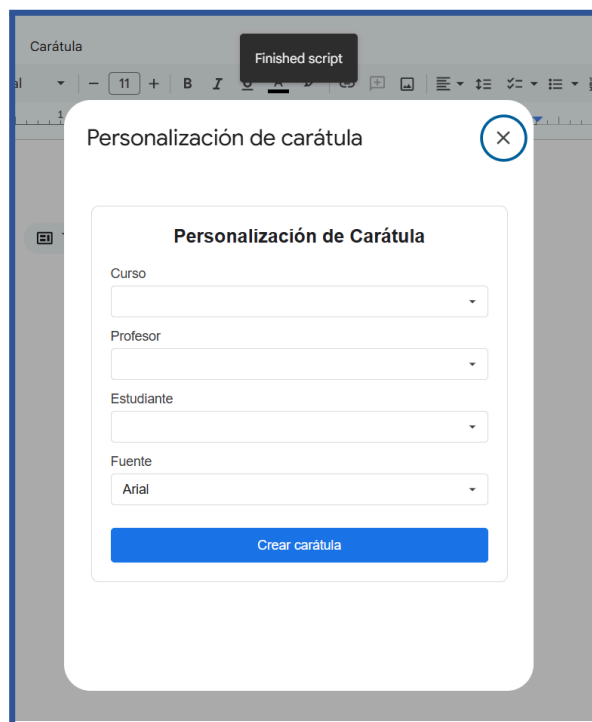
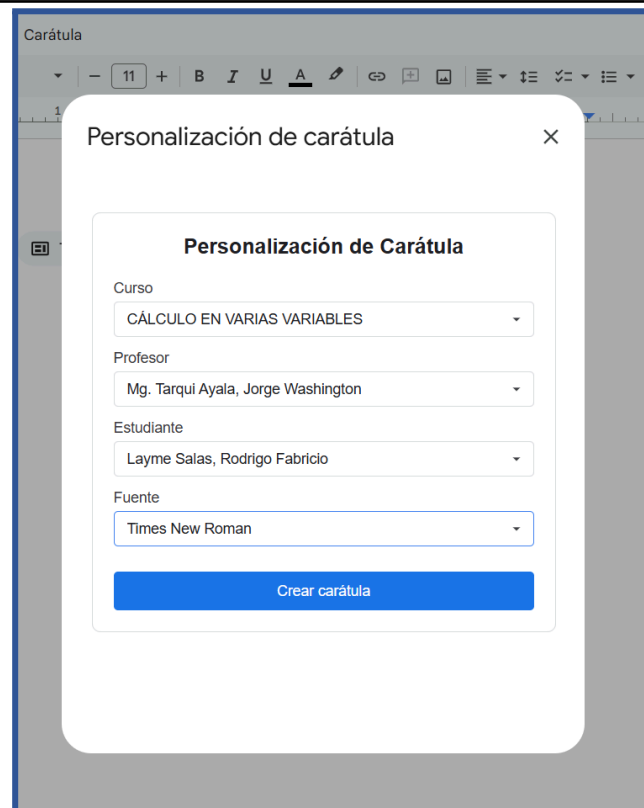
A screenshot of a 'Personalización de carátula' (Cover Customization) dialog box. The dialog has a title bar with a close button (X). Inside, there's a section titled 'Personalización de Carátula' with four dropdown menus: 'Curso', 'Profesor', 'Estudiante', and 'Fuente'. The 'Fuente' dropdown is currently set to 'Arial'. At the bottom, there is a blue button labeled 'Crear carátula'. A 'Finished script' notification is visible in the background.

Fig. 3 El cuadro generado por el Script del programa personalizable



Carátula

Personalización de carátula

Personalización de Carátula

Curso
CÁLCULO EN VARIAS VARIABLES

Profesor
Mg. Tarqui Ayala, Jorge Washington

Estudiante
Layme Salas, Rodrigo Fabricio

Fuente
Times New Roman

Crear carátula

Fig. 4 El cuadro con los datos recolectados y personalizado, curso, profesor con grados académicos, nombre (del creador del programa) y la fuente (las más comunes son Arial y Times New Roman)

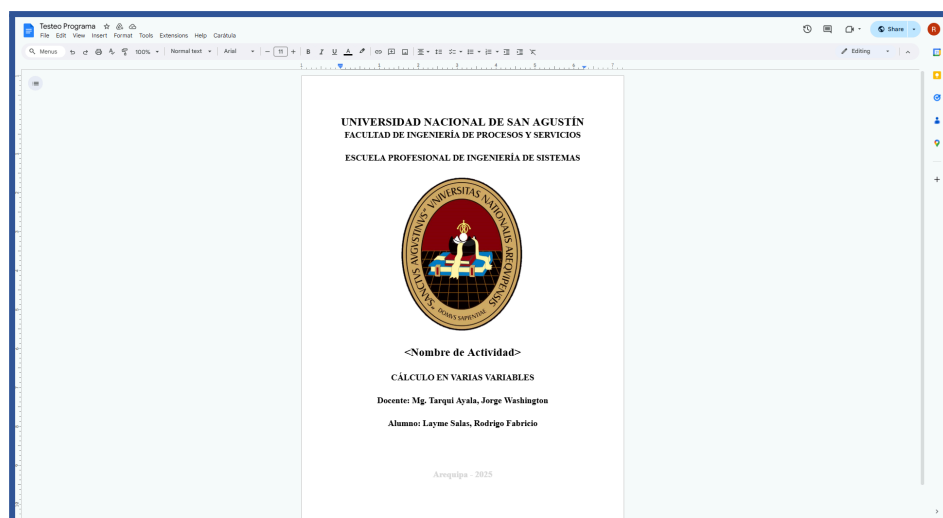


Fig. 4 ¡Carátula con los estándares completos creada con un click!

1.- modalFinal.gs (GoogleAppScript)

```
1 function onOpen() {
2   DocumentApp.getUi()
3     .createMenu('Carátula')
4     .addItem('Generar', 'showModal')
5     .addToUi();
6 }
7
8 function showModal(){
9   const html = HtmlService
10     .createHtmlOutputFromFile('dialogFinal')
11     .setWidth(450)
12     .setHeight(530);
13   DocumentApp.getUi().showModalDialog(html, 'Personalización de carátula');
14 }
15
16 function getData() {
17   const ss = SpreadsheetApp.openById('1uCE8KJoVQtJzpVaK22ewlwRwX9WUVxVDS1Cfjh5ql0');
18   const sheet = ss.getSheetByName('Data');
19   return {
20     teacherNames: sheet.getRange('A12:A20').getValues().flat().filter(v => v !== ''),
21     courseNames: sheet.getRange('C2:C9').getValues().flat().filter(v => v !== ''),
22     studentNames: sheet.getRange('D2:D93').getValues().flat().filter(v => v !== '')
23   };
24 }
25
26 function createCover(course, teacher, student, font){
27   const doc = DocumentApp.getActiveDocument();
28   const body = doc.getBody();
29   //Header
30   body.appendParagraph('UNIVERSIDAD NACIONAL DE SAN AGUSTÍN')
31     .setFontFamily(font).setFontSize(21).setBold(true)
32     .setAlignment(DocumentApp.HorizontalAlignment.CENTER)
33     .setForegroundColor('#000000');
34   body.appendParagraph('FACULTAD DE INGENIERÍA DE PROCESOS Y SERVICIOS')
35     .setFontFamily(font).setFontSize(16).setBold(true)
36     .setAlignment(DocumentApp.HorizontalAlignment.CENTER)
37     .setForegroundColor('#000000');
```

```
38 body.appendParagraph('').setFontSize(16);
39
40 body.appendParagraph('ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS')
41   .setFontFamily(font).setFontSize(16).setBold(true)
42   .setAlignment(DocumentApp.HorizontalAlignment.CENTER)
43   .setForegroundColor('#000000');
44
45 body.appendParagraph('').setFontSize(16);
46 //Image
47 const urlImage = 'https://upload.wikimedia.org/wikipedia/commons/f/f9/Escudo_UNSA.png';
48 const blob = UrlFetchApp.fetch(urlImage).getBlob();
49 body.appendParagraph('')
50   .setAlignment(DocumentApp.HorizontalAlignment.CENTER)
51   .appendInlineImage(blob)
52   .setWidth(4.24 * 72)
53   .setHeight(5.30 * 72);
54 //Middle Text
55 body.appendParagraph('').setFontSize(16);
56 body.appendParagraph('<Nombre de Actividad>')
57   .setFontFamily(font).setFontSize(21).setBold(true)
58   .setAlignment(DocumentApp.HorizontalAlignment.CENTER)
59   .setForegroundColor('#000000');
60 body.appendParagraph('').setFontSize(16);
61
62 body.appendParagraph(course)
63   .setFontFamily(font).setFontSize(16)
64   .setAlignment(DocumentApp.HorizontalAlignment.CENTER)
65   .setForegroundColor('#000000');
66
67 body.appendParagraph('').setFontSize(16);
68
69 body.appendParagraph('Docente: ${teacher}')
70   .setFontFamily(font).setFontSize(16).setBold(true)
71   .setAlignment(DocumentApp.HorizontalAlignment.CENTER)
72   .setForegroundColor('#000000');
73
74 body.appendParagraph('').setFontSize(16);
75
76 body.appendParagraph(`Alumno: ${student}`)
77   .setFontFamily(font).setFontSize(16).setBold(true)
78   .setAlignment(DocumentApp.HorizontalAlignment.CENTER)
79   .setForegroundColor('#000000');
80
81 body.appendParagraph('').setFontSize(14);
82 body.appendParagraph('').setFontSize(14);
83 body.appendParagraph('').setFontSize(14);
84 body.appendParagraph('').setFontSize(14);
85 //Footer
86 body.appendParagraph('Arequipa - 2025')
87   .setFontFamily(font).setFontSize(16).setBold(true)
88   .setForegroundColor('#d9d9d9')
89   .setAlignment(DocumentApp.HorizontalAlignment.CENTER)
90 body.removeChild(body.getChild(0));
91
92 }
```

2.- 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     <base target="_top">
7     <style>
8       body { margin: 0; padding: 0; font-family: Roboto, Arial, sans-serif; background: #fff; }
9       .dialog-container {
10         width: 100%;
11         max-width: 400px;
12         margin: 40px auto;
13         padding: 20px;
14         border: 1px solid #dadce0;
15         border-radius: 8px;
16         box-shadow: none;
17         background-color: #fff;
18       }
19       h3 {
20         margin-top: 0;
21         font-size: 20px;
22         color: #202124;
23         text-align: center;
24       }
25       label {
26         display: block;
27         margin: 12px 0 4px;
28         font-size: 14px;
29         color: #3c4043;
30       }
31       select {
32         width: 100%;
33         padding: 8px 12px;
34         font-size: 14px;
35         color: #202124;
36         border: 1px solid #dadce0;
37         border-radius: 4px;
38         appearance: none;
39         background-color: #fff;
```

```
40       background-image: url('data:image/svg+xml;utf8,<svg fill="%233c4043" height="24" viewBox="0 0 24 24" width="24" xmlns="http://www.w3.org/
41       2000/svg"><path d="M7 10 5 5 15 5"/></svg>');
42       background-repeat: no-repeat;
43       background-position: right 8px center;
44       background-size: 16px;
45       cursor: pointer;
46     }
47     select:focus { outline: none; border-color: #4285f4; }
48     button {
49       margin-top: 24px;
50       width: 100%;
51       padding: 10px;
52       font-size: 14px;
53       color: #fff;
54       background-color: #1a73e8;
55       border: none;
56       border-radius: 4px;
57       cursor: pointer;
58     }
59     button:hover { background-color: #1669c1; }
60     button:active { background-color: #1558a0; }
61   </style>
62 </head>
63 <body>
64   <div class="dialog-container">
65     <h3>Personalización de Carátula</h3>
66
67     <label for="course">Curso</label>
68     <select id="course"></select>
69
70     <label for="teacher">Profesor</label>
71     <select id="teacher"></select>
72
73     <label for="student">Estudiante</label>
74     <select id="student"></select>
75
76     <label for="font">Fuente</label>
77     <select id="font">
78       <option value="Arial">Arial</option>
```



```
78     <option value="Times New Roman">Times New Roman</option>
79   </select>
80
81   <button onclick="insert()">Crear carátula</button>
82 </div>
83
84 <script>
85   window.onload = function () {
86     google.script.run.withSuccessHandler(function(data) {
87       const { teacherNames, courseNames, studentNames } = data;
88       const fill = (id, items) => {
89         const sel = document.getElementById(id);
90         sel.innerHTML = '';
91         items.forEach(item => {
92           const opt = document.createElement('option');
93           opt.value = item;
94           opt.textContent = item;
95           sel.appendChild(opt);
96         });
97       };
98       fill('course', courseNames);
99       fill('teacher', teacherNames);
100      fill('student', studentNames);
101    }).getData();
102  };
103  function insert() {
104    const course = document.getElementById('course').value;
105    const teacher = document.getElementById('teacher').value;
106    const student = document.getElementById('student').value;
107    const font = document.getElementById('font').value;
108    google.script.run.createCover(course, teacher, student, font);
109    google.script.host.close();
110  }
111 </script>
112 </body>
113 </html>
```

- No hay ningún error de sintaxis, todo funciona correctamente.

II. CONCLUSIONES

*Este proyecto resalta la importancia de la automatización en tareas repetitivas como la creación de carátulas. La integración de **Google Apps Script** y **Google Sheets** facilita enormemente este proceso, ahorrando tiempo y esfuerzo tanto para estudiantes como para docentes, ya que elimina la necesidad de crear manualmente cada carátula. También, al enlazar el proyecto con **Google Sheets**, la gestión de la información se vuelve más eficiente. Se pueden actualizar los datos de estudiantes, docentes y cursos de forma centralizada en una hoja de cálculo, lo que se refleja automáticamente en las opciones disponibles en las listas desplegables. **JavaScript** fue vital también en el proyecto, fue una conexión entre el front-end y el back-end del programa. Nos gustó mucho usar estas tecnologías y estamos dispuestos a volverlo más escalable y abrirlo para todas las escuelas dentro de la comunidad universitaria.*

RETROALIMENTACIÓN GENERAL

La versión final no resultó ser muy escalable por falta de una conexión más profesional con una base de datos real que muestre los profesores, sus cursos que dictan actualmente, etc.

Hay partes del código que podrían ser colocadas en funciones auxiliares para evitar la repetición.

Por lo demás todo funciona cómo debería.



REFERENCIAS Y BIBLIOGRAFÍA

- Javascript tutorial. \url <https://www.w3schools.com/js/>, 2024. Accessed: 02-05-2024.
- Loiane Groner. *Learning JavaScript Data Structures and Algorithms: Write complex and powerful*