

Project / Module: Bootstrap Showcase Portfolio

Date: 25/09/2025

Goals for the documented session

- Add elements from Bootstrap to my personal website.
- Modify the structure of my personal website to implement the grid system.

Tasks Completed

- Implement the following elements: Navbar and/or tabs, Accordion, Buttons, Tooltips.
- Modify the structure.

Use of Gen AI

ChatGPT

PROMPT 1

- Prompt: “[My Code] Can you implement the Bootstrap grid system to the code? But leave everything else as it is.”
- Output: “Perfecto 🙌. Para que tu página quede **igual a como la tienes ahora** pero organizada dentro de un **grid de Bootstrap**, lo que hacemos es envolver cada bloque principal dentro de un .container → .row → .col. Así no cambia el contenido, solo la **estructura y alineación**. Te dejo el código reacomodado:”
- Edits: The code was the same, but now with a grid system implemented. AI left the part from the table empty, and I had to add the row.

Reflection: Since I had to research the Bootstrap grid system in the last project delivery, I already knew how the system works and what the classes mean. That's why I asked ChatGPT to implement it to the code, because it allowed me to be more efficient and focus my attention on learning how to implement the other elements that are asked for in the exercise.

PROMPT 2

- Prompt: “Should the Bootstrap Accordion be divided by rows like the grid system?”
- Output: “El **accordion de Bootstrap** no requiere que lo pongas dentro de rows. Solo necesita su contenedor con la clase accordion y cada ítem con

accordion-item. Solo deberías usar **rows y cols** si quieres **acomodar varios accordions** en la grilla de Bootstrap.”

- Edits: I implemented an Accordion in the Hobbies section from the Home page and, since there was only text in that section, when I implemented the grid system, it was organized by columns and rows. So, I had to erase them and adapt the code from the Bootstrap website to what I needed.

Reflection: I didn't know that everything in bootstrap was not organized in the grid system. I thought that the grid system was like a general part of Bootstrap, but now I understand that it is not. It is like another element, but its an structure, not an element.

PROMPT 3

- Prompt: Why doesn't my nav bar display the menu?”
- Output: “¡Ah! Ya veo cuál es el problema. Estás usando **Bootstrap 5**, pero tu código del navbar tiene **atributos de Bootstrap 4** (data-toggle, data-target) en lugar de los correctos para Bootstrap 5 (data-bs-toggle, data-bs-target). En Bootstrap 5, los atributos **cambiaron** para usar el prefijo data-bs-.”
- Edits: The references that I had in the head of the html code were from Bootstrap 5, but the part of the nav bar was Bootstrap 4. In the link where I was viewing the bootstrap examples, Bootstrap 4 was being explained. I changed my search on Google from “Nav Bar Bootstrap” to “Nav Bar Bootstrap 5”, implemented the correct version and adapted it to the code.

Reflection: I didn't imagine that the versions could affect the implementation so much. Now, I always search for the examples on Google as “[element I want] Bootstrap 5”.

What I Learned

- The accordion doesn't have to be divided by rows and columns.
- If you duplicate scripts, the elements behave strangely.
- You must be careful when duplicating parts of the code.
- There are different versions of Bootstrap.

Challenges & Bugs

- Problem: I had to be very careful when modifying the code from the official website because there were divs that I had to maintain or erase and if I left extra divs, the code could stop working.

- Solution: I was looking at what my code was before and to the bootstrap code at the same time, so that I could modify it line by line and be very careful. That allowed me to understand better the implementation of the Accordion.
- Problem: My Accordion element suddenly stopped working as it should. I couldn't collapse my headings.
- Solution: When implementing the Tooltips I was having problems (I didn't have time to solve it during this session) and thought that the solution was to add the Js script from bootstrap, although I already had one. Duplicate scripts caused the accordion to behave strangely.
- Problem: There were two headings that were collapsing at the same time.
- Solution: The bootstrap example had three items in the accordion, and I needed four, so I duplicated the lines from one item. Since it was duplicated, although the content was different, the ids were the same, so the classes were doing the same thing for both items. I only had to make a unique id for each item.

Resources Used

- ChatGPT AI
- Bootstrap website

Next Steps

- Add more of the elements from Bootstrap to my personal website.
- Be more careful when editing.

Personal Reflection

I should be more careful when adapting the examples to my code because little details like the duplications of ids can cause problems that seem really complicated, but that can be solved easily. If I pay attention to those things I can use my time on developing instead of debugging. In general, those kinds of mistakes are the ones that I make so it will be perfect if I focus on solving them. Also, when I changed the code from the personal website, I realized that the bootstrap elements and grid system make the website look better, even without adding anything extraordinary.

Project / Module: JS Exercises

Date: 09/10/2025

Goals for the documented session

- Finish the JS portfolio.
- Complete the exercise that sums up at least 8 points to complete the task.

Tasks Completed

- Fibonnaci
- Images
- Prime
- Sieve_erathostenes

Use of Gen AI

ChatGPT

PROMPT 1

- Prompt: “[Fibonacci.js code] Can you explain it to me?”
- Output: “¡Claro! Vamos a desglosar tu código paso a paso y ver qué hace y qué falta:”
- Edits: In advanced coding class we saw the algorithm for a Fibonacci recursive function and I added it to the code after Chat made me understand it.

Reflection: I understand better the code.

PROMPT 2

- Prompt: “[Error] What does it mean?”
- Output: This error means... To solve it follow the following steps (run and configure node.js in the terminal) ...
- Edits: I made this prompt several times and the errors were related to node.js. I was not running the JavaScript files through node.js or I was not accessing the correct folder, so in the terminal there were shown errors that said that it couldn't find the file.

Reflection: Most of the errors that I get are related to running the code because of mistakes in syntax or in running the code. They also have an easy solution or a simple error, but I usually can't identify them by myself. I mean, I know what could be the problem, but I don't know where is exactly ubicated.

PROMPT 3

- Prompt: [ErrorReference: document is not defined at...] What doesn't it work?"
- Output: "¡Este error ocurre porque estás **intentando ejecutar código que usa document.getElementById con Node.js!**!" Si quieres probar el Fibonacci solo en la terminal, elimina la parte de document y pasa el número como argumento.
- Edits: I was using document.getElementById and console.log at the same time. I erased the document.getElementById and tested the function Fibonacci with console.log, then I connected it to the html file.

Reflection: I wasn't sure how node and js interact with each other, but now I know that js is related to the DOM and node is related to the server.

PROMPT 4

- Prompt: [Code] este es mi archivo html, pero el navegador no hace nada
- Output: "¡El problema es que tu **botón no tiene un evento asociado!**"
- Edits: I added the eventlistener function.
- Reflection: I wasn't so familiar to the functions and syntaxis that follows JS to interact with other languages or parts of the server. Now I understand that the event listener "listens" when the user clicks on an specific button.

PROMPT 5

- Prompt: var resultado = fibonacci(); como sabe qué numero es?
- Output: chat explained the flow of the code to me.
- Reflection: Since there were functions called before its declaration I got lost on the flow, but I could understand at the end thanks to Chat.

What I Learned

- JS is for modifying the DOM and node.js to create the environment.

- Window onload function is to be sure that the DOM is loaded before looking for the elements.
- The JS looks for the elements from the DOM by ID with document.getElementById function.
- .textContent changes the text from the specified element.
- parseInt() converts the value input to an integer so that it can be managed correctly into the function.
- Use script is for avoiding errors in a function. JavaScript starts to prohibit some things like declaring empty variables and send error messages.

Challenges & Bugs

- Problem: There were a lot of parts of the code that threw errors, most of them because I was missing some elements or I was writing something wrong, but I wasn't sure how to solve it because I thought that it was a problem from the algorithm.

Resources Used

- ChatGPT AI
- Friends

Next Steps

- Continue with the other exercises.

Personal Reflection

It was more difficult to me to understand JavaScript than the other exercises because I have never used it before and familiarizing myself with the language, understanding the functions, what you can call and do. Although the form in which you write the code is like C++ the functions and getting elements from the DOM were difficult for me to understand.

Project / Module: JS Exercise

Date:08/10/2025

Goals for the documented session

- Finish the JS Exercise in the personal website and finally understand the language.
- **Tasks Completed**

I didn't finish the exercise in that session, but I understood how the language modifies the DOM.

AI Usage

- PROMPT: What is queryselector and getting element by ID?
- OUTPUT: ✎ `document.getElementById('id-del-elemento')` ➔ Sirve para seleccionar un elemento específico que tenga un atributo id en tu HTML. ✎ `document.querySelector('selector-css')`
- ➔ Es más **flexible**. Permite seleccionar **cualquier elemento** usando **selectores CSS**, no solo por `id`.
- PROMPT: What is appendChild?
- OUTPUT: appendChild es un **método de JavaScript** que se usa para **agregar un elemento como hijo de otro elemento en el DOM**.
- PROMPT: const crearTd = text => { const td = document.createElement('td'); td.textContent = text; return td; }; me explicas eso?
 - OUTPUT:
 - PROMPT: img.alt = esBusy ? 'Busy' : 'Free'; me explicas eso?
 - OUTPUT: ?: es **el operador ternario**. Si esBusy es true (checkbox marcado → "Busy"):
 - `img.alt = 'Busy'`
 - `img.src = 'Images/busy.png'`
 - Si esBusy es false (checkbox sin marcar → "Free"):
 - `img.alt = 'Free'`
 - `img.src = 'Images/free.png'`

What I Learned

Thanks to this language I understood better how JavaScripts can create elements and add them to the DOM through different methods that are used by JS. That's why I liked the most that you can create elements with logic instead of only writing the tag. I really didn't imagine that before this exercise.

You can add something to the html as its parent. AppendChild is to add it to the end, but there are other methods that can, for example, add the element at the beginning.

You can make a conditional function only by adding the question mark in the line.

I asked ChatGPT what the tbody is and I figured out that although it is not defined in an explicit form it is a part of a table. The table has theader, tbody and tfoot. In this case, if there is not a tbody, it is created by itself and filled with the correct data.

- **Challenges & Bugs**

- I don't remember if I had challenges or bugs. I think it is easy to implement, but it is long, so maybe I wrote something bad without noticing and caused me a problem with the variables.

- **Resources Used**

- ChatGPT AI
- I asked my cousin about the exercise.

Next Steps

- Study how to write the things and how they look.

Personal Reflection

Something that I haven't understood completely is the way in which documents relate to each other. For example, the JS being added into the html, but the thing that confuses me is writing the right path to the file. Although the files are in the same folder, I always have problems at the beginning connecting to that part.

Project / Module: BasicExpress

Date: 13/10/2025

Goals for the documented session

- Complete the exercise.

Tasks Completed

- I finished this exercise.

Use of AI

- PROMPT: [Instructions]
- OUTPUT: Let's go step by step — here's how to solve that exercise.

- Edit: I wrote the code and commands into the console so that I could create and understand the environment.
- Reflection: It is not difficult to set the environment; it is about reminding the commands to install dependencies and start it. You have to follow some steps that allow the environment to be set.

What I learnt

Creating the server is about following a series of steps where you initialize variables, get the path to the file that will be shown first, allowing middleware functions to receive the parameters correctly, and listening to the port.

The get method and post method can access data that they received in different ways. It is better to use post.

Challenges & Bugs

- Challenge: The html file wasn't recognized although the path was correct.
- Solution: Add path and modify the name and path in the code.

Next step

- Make requests to an API

Personal Reflection

I like that everything is managed by posts and gets. When I learned about this, I started to understand better the interaction in a real webpage. Before I thought that it had to be only implemented by html, javascript and css which made me think that web development was really heavy. But, once you know all this technologies, it is easier to manage so much.

Project / Module: Express & 3rd-Party API

Date: 20/10/2025

Goals for the documented session

- Connect to an API and request the weather of a city submitted by a forms by the user.

Tasks completed

- All the things above

Use of Gen AI

```
PROMPT: // https post app.get("/dictionary", (req, res) => { var url =
"https://api.toys/api/check_dictionary"; const form_data = new FormData();
form_data.append("text", "marry"); const options = { method: "POST", headers:
form_data.getHeaders(), }; var soapRequest = https.request(url, options, (response)
=> { if (response.statusCode === 200) { response.on("data", (data) => { var jsonResp =
JSON.parse(data); console.log(jsonResp); res.send("Success"); }) .on("error", (e) => {
res.send("Error ${e.message}"); }); } else { res.send("Error"); } });
form_data.pipe(soapRequest); });

Output: Perfecto 🤗 estás combinando tres maneras distintas de hacer peticiones
HTTP/HTTPS en Node.js (https, axios y FormData), lo cual está bien para comparar
enfoques.
```

Te explico lo que hace cada bloque y algunos detalles a corregir o mejorar 🤝

- Edits: I changed the sample that came into the portfolio by adapting the request to the API that I was using. I also added the response on end because chat told me it was better to wait until the API stops sending data to avoid losing it. I added the chunk part because https sends data in parts and I had to join that info, transform it into a JS file and with an EJS template show the result.
- Reflection: The differences between the sample and what the code resulted in at the end were about the chunk data and the response.on (end) because it allows data to be received correctly. When I ran the code without these parts, it threw errors.

What I learned

There are two different ways in which you can ask for requests to an API. https is longer and you must specify everything, while axios is easier, up-to-date and you don't have to implement the error code. When using https, the AI sends the information by pieces, so you must integrate everything into an array and then cast it into a JSON file. This API sent the information through 3 “parameters” that contained more information about it. To access the information, you only must enter through points (info.main.temp).

The response function is to wait the answer from the API and do something once it is fulfilled.

Resources Used

AI

Next Step

Understand axios better because I did the exercise with https method

Personal Reflection

Project/Module: Express PersonalAPI

Project / Module: ExpressEJS Portfolio

Date: 15/10/2025

Goals for the documented session

- Finish the ExpressEJS exercise.

Use of Gen AI

ChatGPT

PROMPT 1

- Prompt: “<input type="checkbox" name="opcion1" value="valor1"> POST METHOD <form action="/login" method="<%= true ? 'POST' : 'GET' %>"> change the condition so that if the checkbox is activated, send it with post”
- Output: “Good question — you can't directly make a form's method change dynamically based on a checkbox **purely with HTML**. You'll need a little bit of **JavaScript** to do that.”
- Prompt: “The forms is sent, the client hears the submission and change the method according to the checkbox, then it sends it to the server and the server receives it?”
- Output: “So yes, the client “hears” the submission, updates the method dynamically, and then the server receives it correctly.”
- Edits: I erased the ejs file and the ejs label. In the server instead of rendering the ejs file, with a get method I sent the html file and added the script tag to it.

Reflection: I learned that I cannot change something directly with ejs based on a checkbox.

PROMPT 2

- Prompt: “How do I send to a ejs file?

- Output: **1 Set up EJS in Express**
- Before rendering, you need to tell Express that you're using EJS:

```
app.set('view engine', 'ejs');
```

Express will look for EJS files inside a views folder by default.

2 Render a template

When a client requests a route (like /login), you can render an EJS file:

```
app.get('/test', (req, res) => {
  const userName = "Andrea"; // for example
  res.render('test', { userName });
});
```

- Edits: I implemented that part of the code to my server.js and adapted it.

Reflection: I just needed to remember the syntax. That is something I am struggling with lately.

PROMPT 3

- Prompt: “HOW CAN I SEND MORE THAN ONE VARIABLE?”
- Output: **Passing multiple variables**

```
res.render('test', {
  userName: "Andrea",
  age: 25,
  city: "Mexico City"
});
```

userName, age, and city are all now available in test.ejs.

- Edits: I added one variable to the res.render function in my server.js
- Reflection: I thought it would be more complicated. I thought that maybe I had to convert into a json file or something like that.

PROMPT 4

- Prompt: “Can a button send a get?”
- Output: **1 Inside a form, 2 As a link styled like a button, 3 JavaScript approach.**
- Edits: I chose the JavaScript approach because it was the easiest way to do it.
- Reflection: There are many ways to achieve something, and it is just a matter of choosing which one is best.

PROMPT 4

- Prompt: “[The error I had]”
- Output: In header.ejs you have:

```
<li> <%= name %>!</li>
```

- But in your res.render('home', ...) call for home.ejs, you **did not pass a variable called name.**
 - EJS tries to evaluate <%= name %> and crashes because name is undefined.
-

How to fix it

You need to make sure **every variable used in a template or partial is passed from the server.**

- Prompt: <script>
`document.getElementById('posts_btn').addEventListener('click', () => {
 window.location.href = '/home';});</script>` can I send the name in this lines?
- Output: Yes — you **can send the name** with a GET request using JavaScript, but you must include it in the **URL as a query parameter**, because a button click without a form can't automatically send data otherwise.
- Edits: I sent the name through the URL from the event listener that stems from the posts button and then receive it.
- Reflection: The errors are very clear in the console. It tells you exactly what failed.

What I Learned

- You cannot change something directly with EJS based on what a checkbox says.
- You can easily pass variables through the server, html and JS.
- How to write the code that renders the EJS files in the server.
- How to send a request and parameter via URL.
- You cannot use a method directly in a button unless it's a submission button from a form.
- Muchos de los errores que arroja es por cuestión de sintaxis

Challenges & Bugs

- Problem: I wanted to send a request with a button.
- Solution: I used JavaScript to do it so.
- Problem: The name wasn't reaching the home template.
- Solution: I tried to send it through the URL.
- Problem: But it responded with something empty.
- Solution: I rendered the actual name from EJS.

Resources Used

- ChatGPT AI
- I asked my cousin about the exercise.

Next Steps

- Study how to write the things and how they look.

Personal Reflection

By this point I can understand better the flow that a web page follows. I figured out how the user, the client and the server interact with each other, making it easy to implement what you want. I like the most the EJS templates because I found it easier to change the content based on variables and how easy it is to move information between the backend and frontend. Before this exercise I was visualizing everything as separated things and it was difficult for me to understand the flow, but now I'm happy that I understand more.

