Iker Pacheco Figueiredo
16/06/22

Eventbrite technical test by Iker Pacheco

Introduction

This project was made using the framework Vue.js. I decided to use this framework to discover a new, amazing technology and develop a project with it.

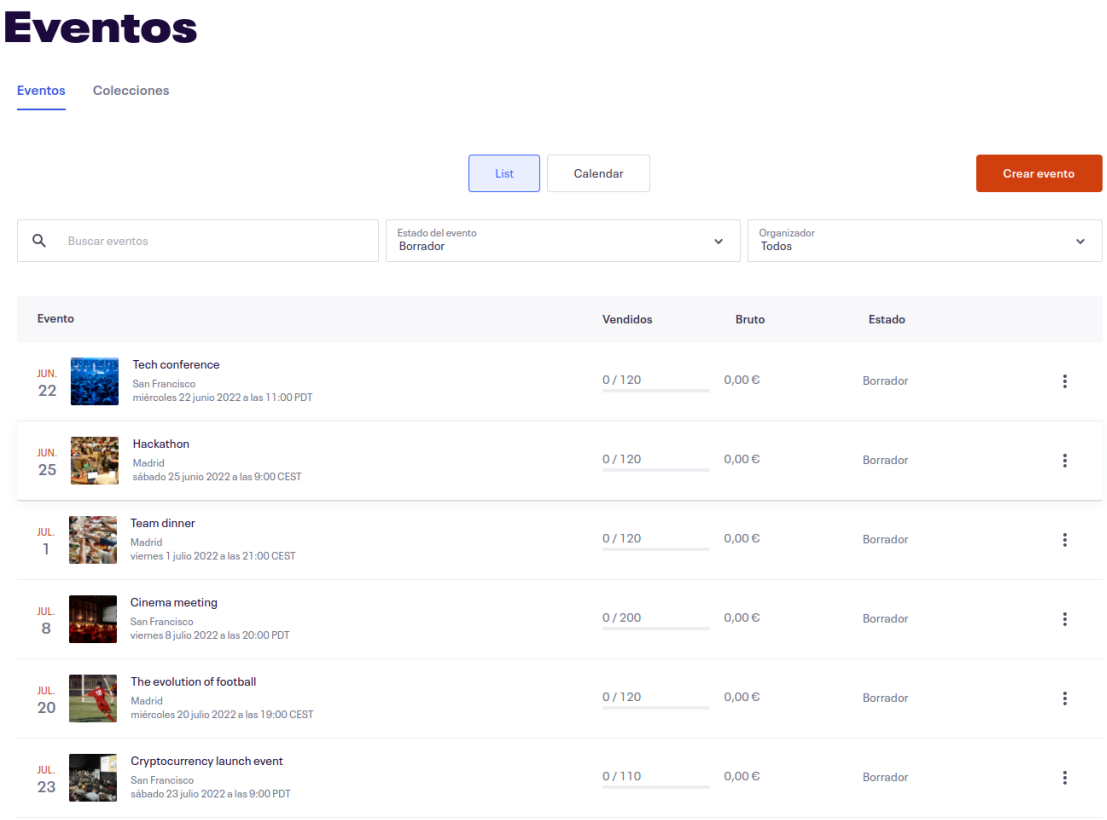
I have always been attracted to this framework, but I've never had the opportunity to try it in a project.

At the end, choosing this technology has been a success. I have nearly completed all the requirements and I found out that I am able to build websites without only using HTML, CSS and JS.

Project Description

This project was new for me in some ways. The main objective was to connect to Eventbrite website and receive information from it, to create a landing page showing the information received.

This information were the events (previously created at the website), each one with specific information such as: name, description, date, ticket prices and logo.



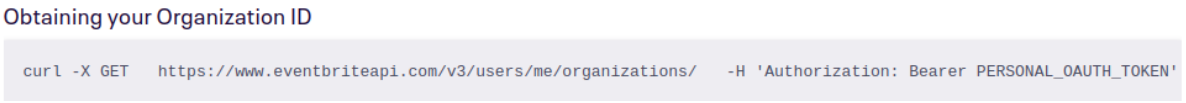
Iker Pacheco Figueiredo
16/06/22

Creation process

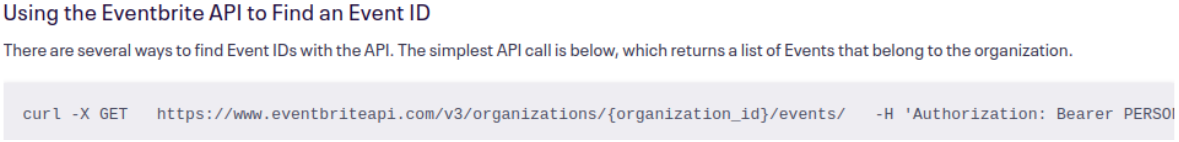
1. Connecting to Eventbrite to access events information

This part was one of the most difficult ones for me at the beginning of the project. It was the first time working with a connection between websites, so at the beginning I was a bit lost.
Then, I found out that Eventbrite has a developers section where I was able to create a private token, that would be so useful in the future.

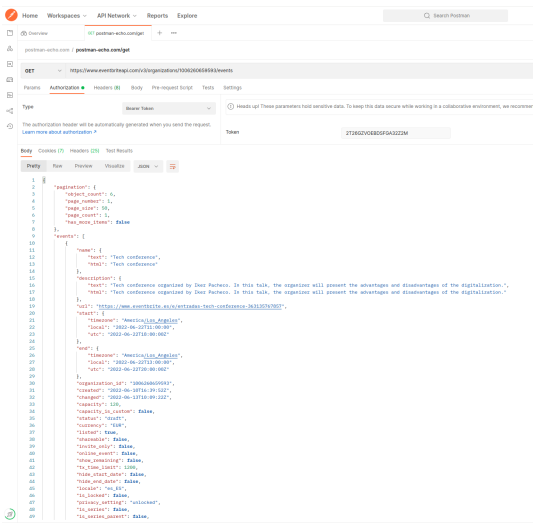
I also saw that there was an API documentation and reference. I read the information carefully to understand some of the concepts that were useful for my project.
Thanks to the information in the reference, I was able to make a get request using POSTMAN. However, I was missing some information, such as the organization ID, that I manage to get it with the URL below.



When I got the organization ID, I replace it in the URL below to find the events:



There I came up with an error that I solved some hours later by giving POSTMAN the authorization of my private Token. After that, I received all the information of my events.



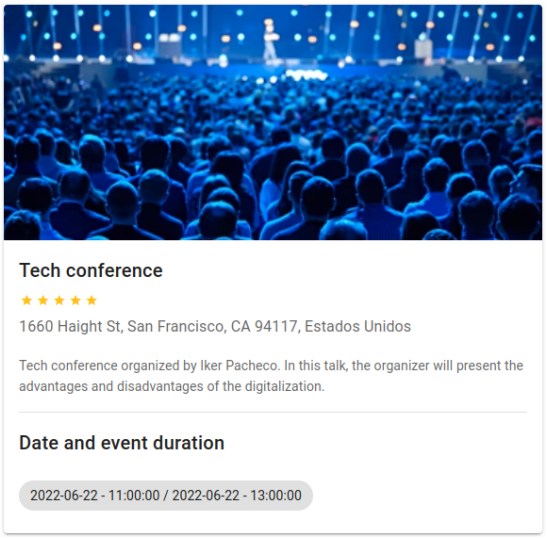
Thanks to axios I managed to save the information from Eventbrite in an array named events. I had a problem at the time of getting the information because it was in “promise” and that didn’t allow me to access to it. I solved by typing await before the axios request.

```
async created() {
  axios.defaults.headers.common['Authorization'] = `Bearer ${"2T26GZVOEBDSFGA32Z2M"}`
  const result = await axios.get("https://www.eventbriteapi.com/v3/organizations/1006260659593/events/?expand=venue");
  this.events = result.data.events;
```

When I achieved to access to the information in my array and in real time, the project became so much easier. Now, It just would be creating the landing page with the event information.

2. Creating the landing page sections

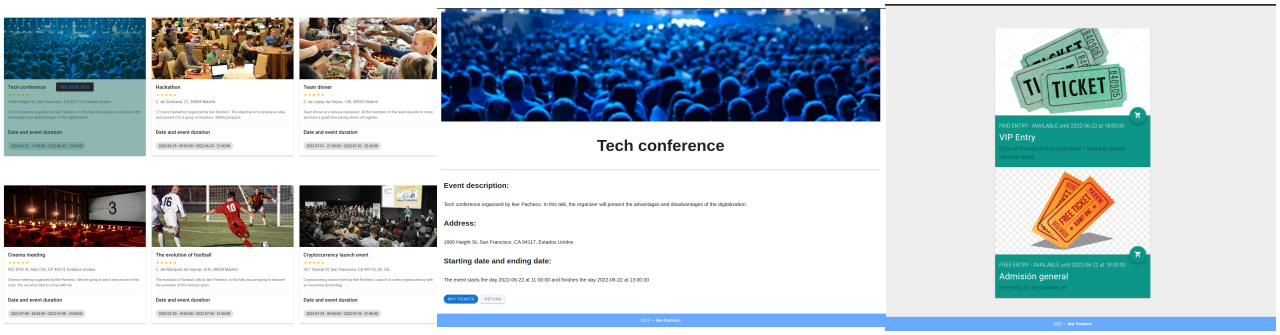
To do this step, I used VUETIFY to export a card template to work easily and efficiently.



When I had the main template, I created a new component with the functionality of creating a card event like this one for each event of the array.
So, working in this way, I just needed to create a loop through the event array and a component that receives an event and create its card.

Then, when all the events were created, I made some logic to change the background of the template and to replace the information of the card with the event information.

I also created a small animation with a button that goes to that event page, where I display again the main information of the event. In this page, the user can see a BUY TICKETS button that redirects them to a page with the information of the tickets.

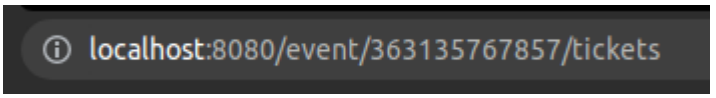


3. Connect sections

Firstly, I tried to connect the different sections, redirecting each button to its page and passing the event array as argument with the event.id to know with which element I was working with. However, this was not the best method of doing it because although it works in some way, the user is not able to reload the screen or return.

So, for this reason, I changed the way of connecting. Instead, I used the local Storage, recommended by my teacher. Using this, I would be able to get the information from the local storage in every page I was creating.

To know exactly what event the user clicked, I get the event ID and concatenate it to the URL. Then for each page on the website, I would get the information from the Local Storage and, using a for loop, check if the ID of the URL is the same as the one in the array of events at the local storage. The one that is equal to the URL, that would be the event clicked.




```
async created() {
  const result = JSON.parse(localStorage.getItem("events"));
  console.log(result);
  this.eventView = result.find(event => event.id == this.$route.params.id)
  this.get_hours(this.eventView);
}
```

After having that, I focused in improving some features and I added the online/presential filters. At this point, the project was nearly finished.

Iker Pacheco Figueiredo
16/06/22

QUESTIONS

1. **How would you design this platform (systems/services/applications, technologies, protocols, formats, programming languages, frameworks, managed services, etc...)?**

- I would like to learn other cool technologies such as React to make a website or make an app for mobile devices.

2. **How would you split the project in milestones?**

-I would split it applying the organization used in this project: connecting both websites and understanding the API reference in Eventbrite, receiving the information, connect pages by event ID, building the main project's interface using a good code structure, designing the main page for each event, building the ticket platform and adding some extra features such as the filter.

3. **What would you build next?**

-I would like to implement a buying system and “add to cart” section, it could be useful for the website

4. **What do you think is the most complex part of the exercise?**

-For me, the most complex part of the project was to connect through the API to another website because I had never worked with it before. However, now that I have learned it, I think that can be so useful for future projects.