Battle Arena Gaming

Web Projects I

2023-2024

Eugènia - Juan - Edouard - Georgina

Table of contents

1. Introduction

This project aims to join strategic gameplay with practical web development skills. It is a web application designed to engage us students through character creation, competitive battles, and progression systems, improving our capabilities of HTML, CSS, and JavaScript.

The core objectives focus on developing a dynamic and responsive user interface, implementing a solid back-end to manage game logic, and ensuring a good user experience on both desktop and mobile platforms. With Vue.js as its core, the project demonstrates technical learnings and shows a thoughtful design and user based development ideas.

Our approach will show how we overcame the challenges of player management, real-time interactions, and persistent state management. This assignment is an opportunity to apply our theoretical knowledge to a practical scenario, demonstrating our understanding of web technologies and our ability to create an online masterpiece.

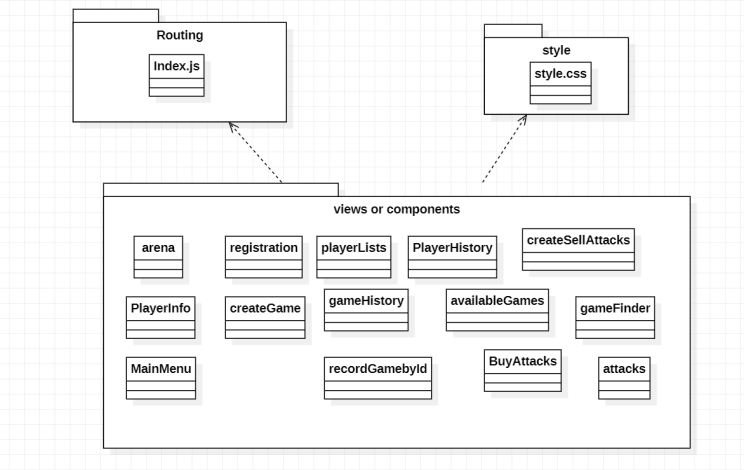
2. Development and results

Describe the relevant aspects related to the development of the project. Include diagrams, screenshots, execution examples, etc. Include what you deem appropriate.

The diagram I included below was a mini sketch and we worked from it. Inside the routing we have the file index.js that does all the routing of the components and assigns a different path to each component.

We have the style package which contains the style.css which are styles applied to all the components, its a general design like the main colors which all components share.

Then we have all the components inside the same package, which represent all the views done during this project



So, when we were developing the project the way we checked if things were going good was to set a lot of console logs, to check whether our code was returning the information correctly or not, this way, by setting some console logs in strategic places we could know where our possible mistakes were located. After that we deleted all of them as it was not necessary to maintain them, they were only there to guide us.

3. Technologies and tools

In this project we learned how to program a game using different technologies from the front end of a web to the back end.

In the first phase of the project we did the front end of the web. In order to fully accomplish a working web, we used vue and we created different components for each screen of our videogame. We used .js called index where all the routing was done. This .js allowed us to connect all the vue components between each other. Each component had his own path inside this .js file.

```
import arena from '../components/Arena.vue';
import BuyAttacks from '../components/BuyAttacks.vue';
import CreateSellAttacks from '../components/CreateSellAttacks.vue';
import RecordGameById from '../components/RecordGameById.vue';
import home from '../components/Home.vue';

const routes = [


  { path: '/', redirect: '/home' },
  {
    path: '/main-menu',
    name: 'MainMenu',
    component: MainMenu,
  },
  {
    path: '/register',
    name: 'registration',
    component: registration,
  },
  {
    path: '/arena/:gameId/:rows/:hp/:currentPlayer',
    name: 'arena',
    component: arena,
    props: true,
  },
  {
    path: '/player-history/:id',
    name: 'PlayerHistory',
```

Here is a screenshot of how the routing was done. First we need to import all the components, if we don't import them the routing is not going to work for the ones missing. After importing them, we have to create a path for each of the components with the path, the name and the component name to which we are reffering.

Inside each vue component we used HTML to compose the components and we used Css to style the components. In some components we did use Java script to add some functionalities such as the filtering of players , the filtering of matches by date or by status and to better improve some interactions.

We also have a css file called “style.css” where all the general designs are done, because in each component we have css code, but it's specifically done for that component, not every component has the same buttons or forms .

In the second phase we implemented the back-end of the web. We use an API to make all the requests and update our information based on what the API could return. In the vast majority of components we have requested to the API, those calls were made inside each script of the components that required a request.

```
// Fetch attacks information
const attacksResponse = await fetch(`https://balandrau.salle.url.edu/i3/players/attacks`, {
  method: 'GET',
  headers: {
    'accept': 'application/json',
    'Bearer': `${token}`,
  },
});
```

In this screenshot we can see how a GET request is made to the API. In this case this request was used to retrieve all the attacks of a player. We use the token, which is unique for each player, to retrieve the attacks of the logged user.

Also, in this phase we fixed how we handle different screen sizes. So now in each css of each component we have a part where we handle the use of mobiles or smaller resolutions.

In this picture, we can see the css implemented for a specific component. The @media targets the screens that are less than the max-widht. If they are bigger it's not going to apply different adjustments, but when the screen is smaller it will adjust so the user can see everything correctly and it will improve the user experience.

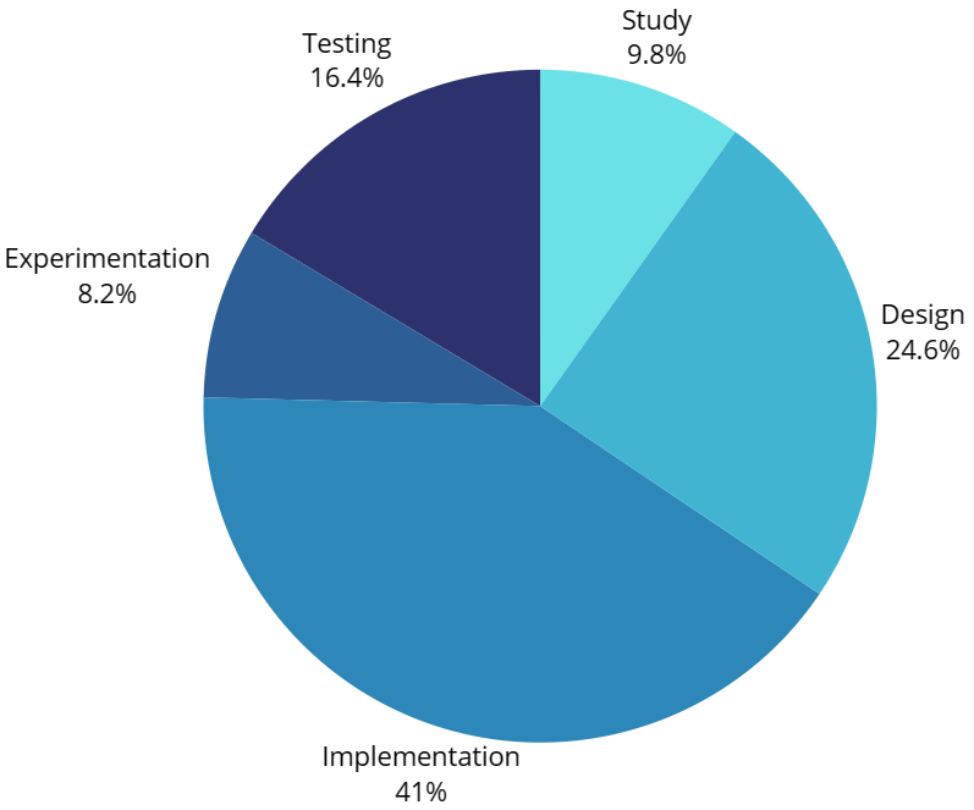
In order to make everything easier for all the members of the group, we used github to upload our codes.

4. Costs

For the time management section, we can divide the work into simple, clear parts:

- Learning: This part was about getting the skills we needed. It included checking out how to use Vue.js, responsive design tips, and game creation basics.
- Design: Here, we drew up plans for what the game would look like and how it would work. We sketched the screens and decided on the steps players would take, and then coded these designs.
- Implementation: This was the main part where we actually made the game logic. We wrote all the code and put everything together.
- Testingt: In this part, we played around with the game to make it better. We tested different ideas and made changes based on what we found.
- Experimenting: The last part was to make sure everything worked well. We looked for any mistakes or problems and made sure the game was fun and ran smoothly.

In total, we have spent 112h, split as follows:



Eugènia - For the first milestone, me and Georgina did the design of the mobile & desktop version in Figma, including the UI Mockups and Wireframes for each. In addition, I did different Vue components such as *Arena.vue*, *Home.vue*, *Registration.vue* and *CreateGame.vue*, with its HTML, CSS and initial functions for later Javascript. Regarding the second and final phase, I have developed quite a lot of queries to connect each Vue to the Api.

Georgina - I started the project focusing on how to properly represent all the implementations so that the user experience and coding excellence can both be feasible and fun. We did this together with Eugènia by completing the design decisions and mockup creations. Then I coded my 4 views for milestone 1, and in this second milestone I focused on improving all the design and making it mobile adaptable for all the views in the program, while helping with structure improvement of the html to achieve the desired outcome of all the elements in the game while the api changes modified most views and had to be rethought and recoded.

Juan - I did the routing to connect the different Vues and let the user move among different screens. Also, some Htmls for *Manage attacks* for instance. For phase 2, I did the logic of the Vue, like the implementation of the vue components, the filtering of the games…

Edouard - I also did the logic of some of the components and a part of the routing.

5. Video

Here's the link to the video showing the different functionalities of Battle Arena.

https://youtu.be/jZtaOitFvbY

6. Conclusions

In conclusion, it is clear that it was a valuable learning journey. Not only did we dive deep into web development and design, but we also got our hands dirty with the practicalities of making a game that runs well on different devices. Through trial and error, we perfected the Battle Arena, balancing complexity and simplicity in both design and functionality. Our efforts resulted in a game that's easy to navigate, visually appealing, and fun to play. Reflecting on the project, it is a reminder that the process is just as important as the final product. We have created not only a game to show but also with improved skills, knowledge, and a proud sense of achievement after all our constant effort.