

# GreenCinePick

Mailinh TA – Benjamin MAZZEGA – Robin PREVOTAT – Nathan  
Codaccioni – Edgar BRANDENBURG

Professor: JENHANI, Ilyes

## Table des matières

Presentation of the Project.....	3
Presentation of the website and its features .....	3
Approach taken to minimize the environmental impacts.....	3
Technical Choices to reduce the carbon footprint .....	3
Architecture of the database and the code.....	3
Design and wireframes .....	5
Implementation.....	11
Carbon footprint analysis .....	13
Discussion and Conclusion .....	14
Eco-Responsible Practices in Web Development .....	14
Challenges faced .....	15
Ideas for future improvements and integration of ecological principles .....	15
Conclusion .....	16

# Presentation of the Project

## Presentation of the website and its features

GreenCinePick is a project that involves strategic and ecological techniques to minimize environmental impact. We have decided to develop a website to recommend movie for users based on their tastes. Among the features, the users can browse movies, rate them and the administrators can manage the movie data by adding, delete and update the movie in the database of the website. We try our best to focus on simplicity, performance and reduced carbon footprint.

## Approach taken to minimize the environmental impacts

First, we did a brainstorming to list all the necessities technologies to code our website. We retain the following solutions:

- ✚ **Apache Server:** It is a lightweight and stable server. It allows optimization of resources (CPU, RAM) by disabling non-essential modules. By choosing that, it reduces server load, leading to lower electricity consumption and fewer network dependencies, which minimizes data center emissions.
- ✚ **MySQL2:** it is an optimized Node.js connector for MySQL databases. It is faster and less resource-intensive compared to heavier ORMs like Sequelize or Mongoose. That is why it is faster and implies more efficient queries that reduce CPU time and the need for scaling, resulting in fewer servers and lower carbon emissions.
- ✚ **Node.js:** it operates on an asynchronous, non-blocking model, making it efficient for handling multiple connections with minimal resources. Consequently, it implies better energy efficiency per request due to the event loop, reducing the number of servers needed and thus lowering energy consumption.
- ✚ **Pug:** This technology is a minimalist template engine that generates concise HTML. It reduces the size of the HTML generated, leading to less data transfer. Less HTML and JavaScript to load means reduced bandwidth usage, lowering the energy footprint related to network traffic.

## Technical Choices to reduce the carbon footprint

Our technologies choices are oriented towards simplicity to reduce carbon footprint. Then, in our code, we try to apply in the beginning the methods learned in the course such as: replace recursive functions by iteratives functions, avoid many if cases if not necessary, optimize the size of the code and reduce the calls to the server. From the start of the project, we paid particular attention to eco-conception principles. Every design and development decision aimed to minimize the website's environmental footprint while maintaining functionality and an engaging user experience.

## Architecture of the database and the code


Table	Lignes	Type	Interclassement	Taille	Perte
movie_db	107	InnoDB	latin1_swedish_ci	64,0 kio	-
ratings	32	InnoDB	utf8_general_ci	32,0 kio	-
users	4	InnoDB	utf8_general_ci	48,0 kio	-
3 tables	143	InnoDB	latin1_swedish_ci	144,0 kio	0 o

About the architecture of the code, we separate the project into two parts: the back end and the front-end.

In the back end, we use Apache Server and MySQL2 for database connections. The routing handles different routes for movie browsing, user authentication, and admin operations.

For the front-end, we use Pug to generate concise and minimal HTML. About the user interface, we included pages for movie browsing, user login/registration, and admin management.

In another part, when we deploy our website, we needed an online database, we tried the Azure database, but we spend all our free tokens on other subject, so if we choosed this solution, we had to pay. So we tested many solutions, and we choose Free Sql Database because it is free and user friendly.

A promotional banner for 'Free Sql Database'. The background is dark with a blurred image of a computer screen showing code. The text 'Free Sql Database' is prominently displayed in white. Below it, a smaller line of text says 'is a web based service to provide sql database functionality for free.' Further down, it says 'Our members benefit from' followed by four bullet points: 'Instant account activation', 'Excellent uptime statistics', 'Remote connections allowed', and 'Easy to use account control panel'. In the top right corner, there is a red ribbon-like graphic with the text '10% off upgrades & renewals for limited time only!' in white.

**Free Sql Database**

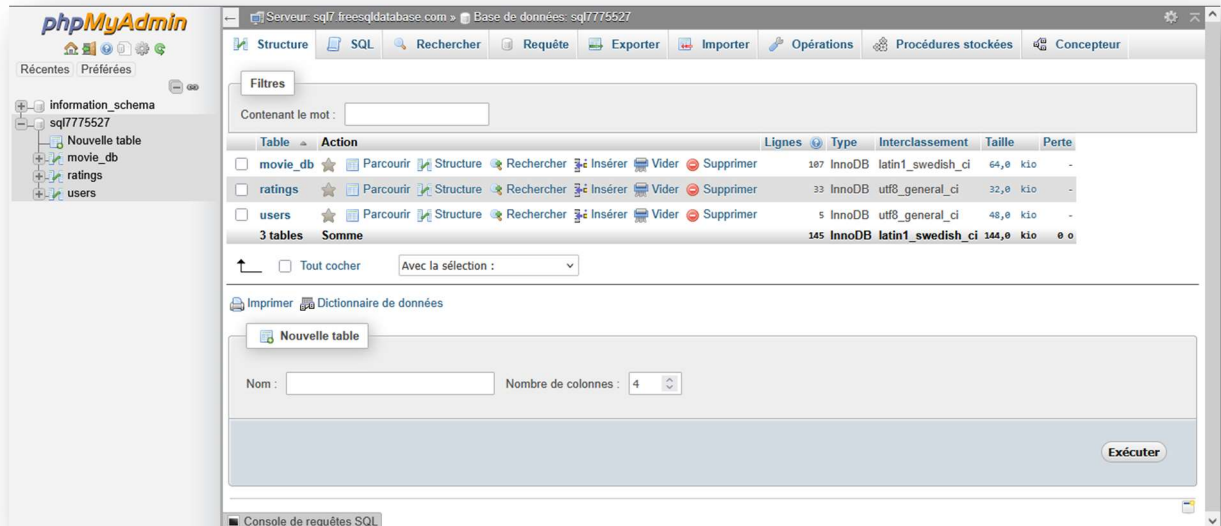
is a web based service to provide sql database functionality for free.

Our members benefit from

- ✓ Instant account activation
- ✓ Excellent uptime statistics
- ✓ Remote connections allowed
- ✓ Easy to use account control panel

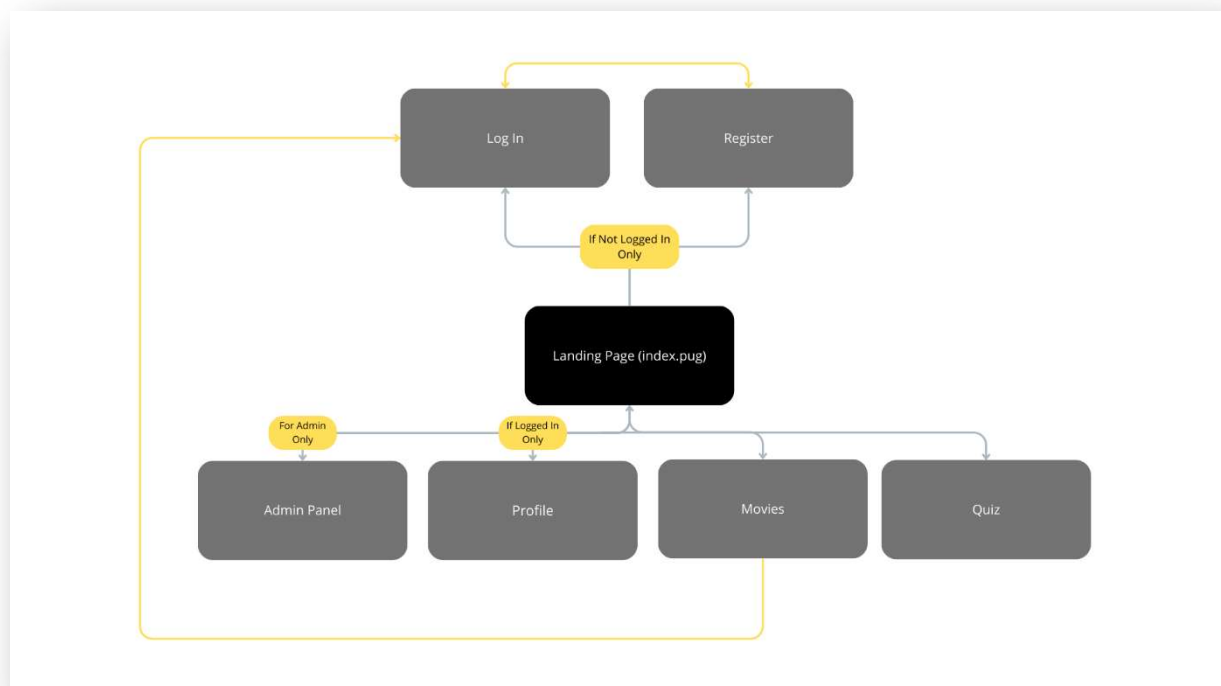
**10% off**  
upgrades & renewals for limited time only!

This solution allowed us to use PhpMyAdmin which is a robust database website used by many people.

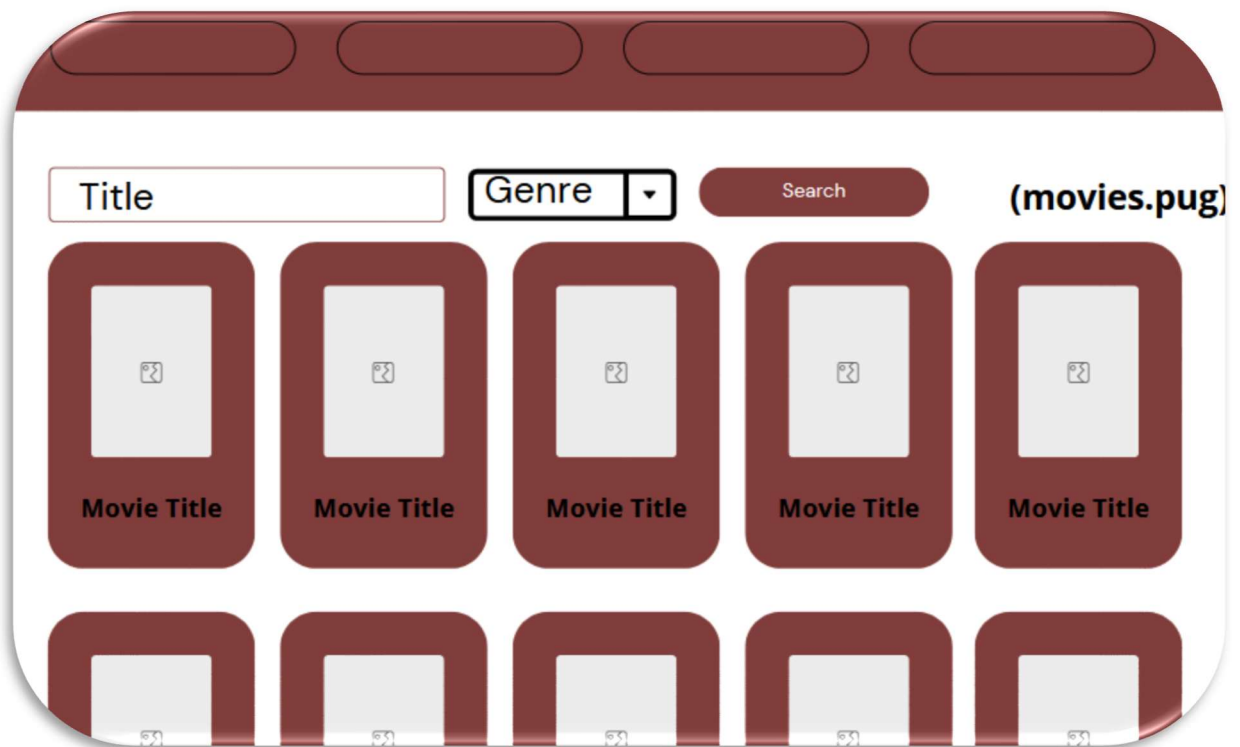
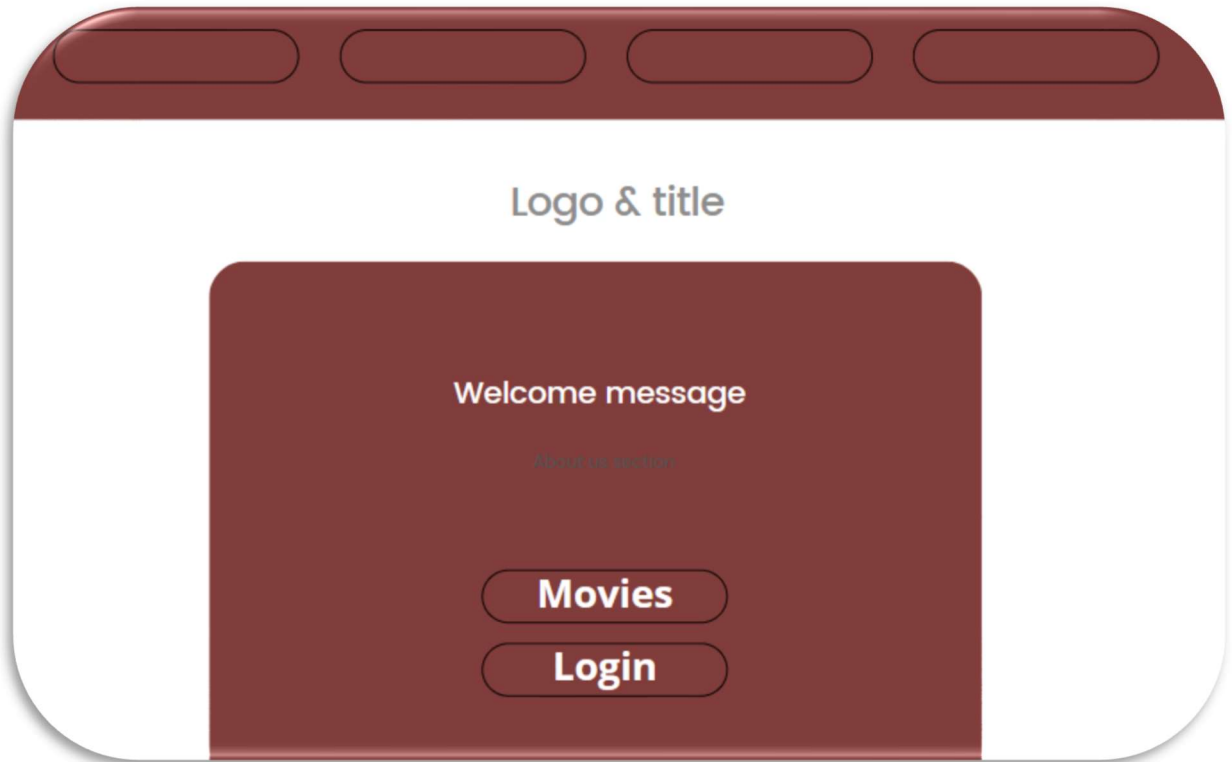


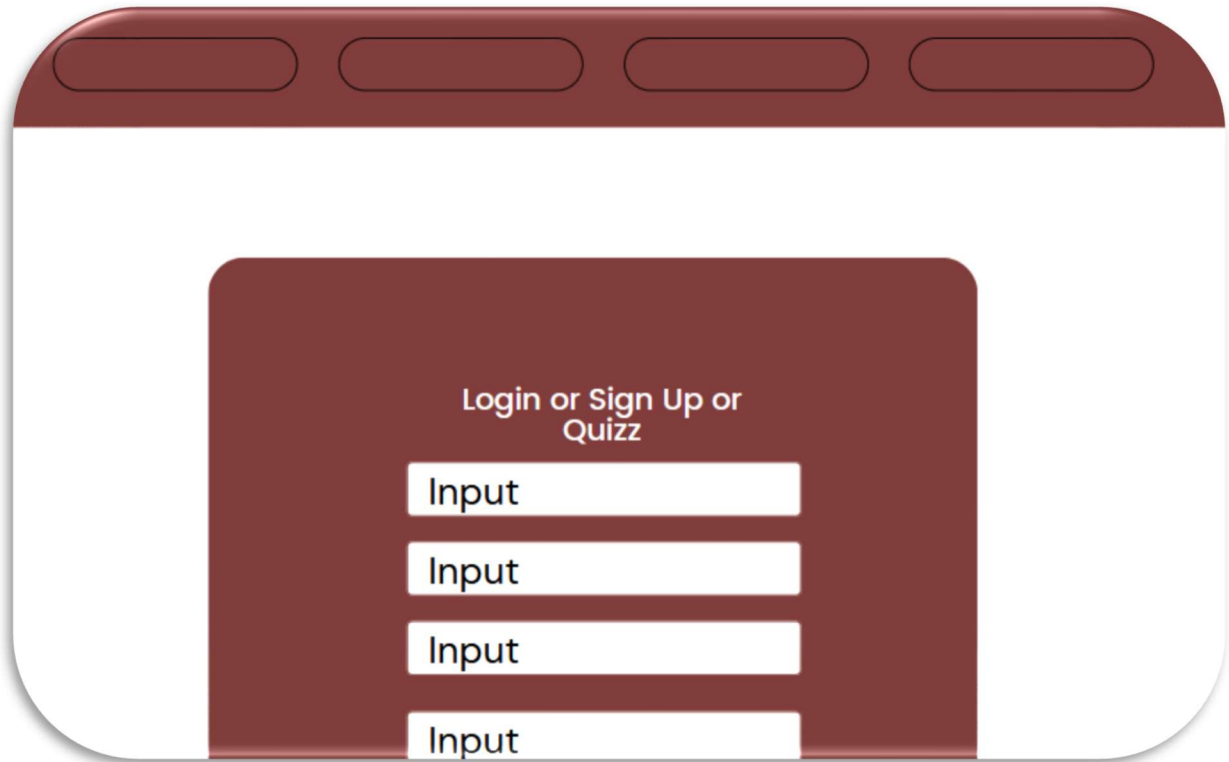
## Design and wireframes

Firstly, we define what we need, so we did a diagram :



Then, we brainstorming on the front-end of the website, we choosed simple interface to navigate easily on the website. We did the following wireframe :





Login or Sign Up or  
Quizz

Input

Input

Input

Input



## Admin Page

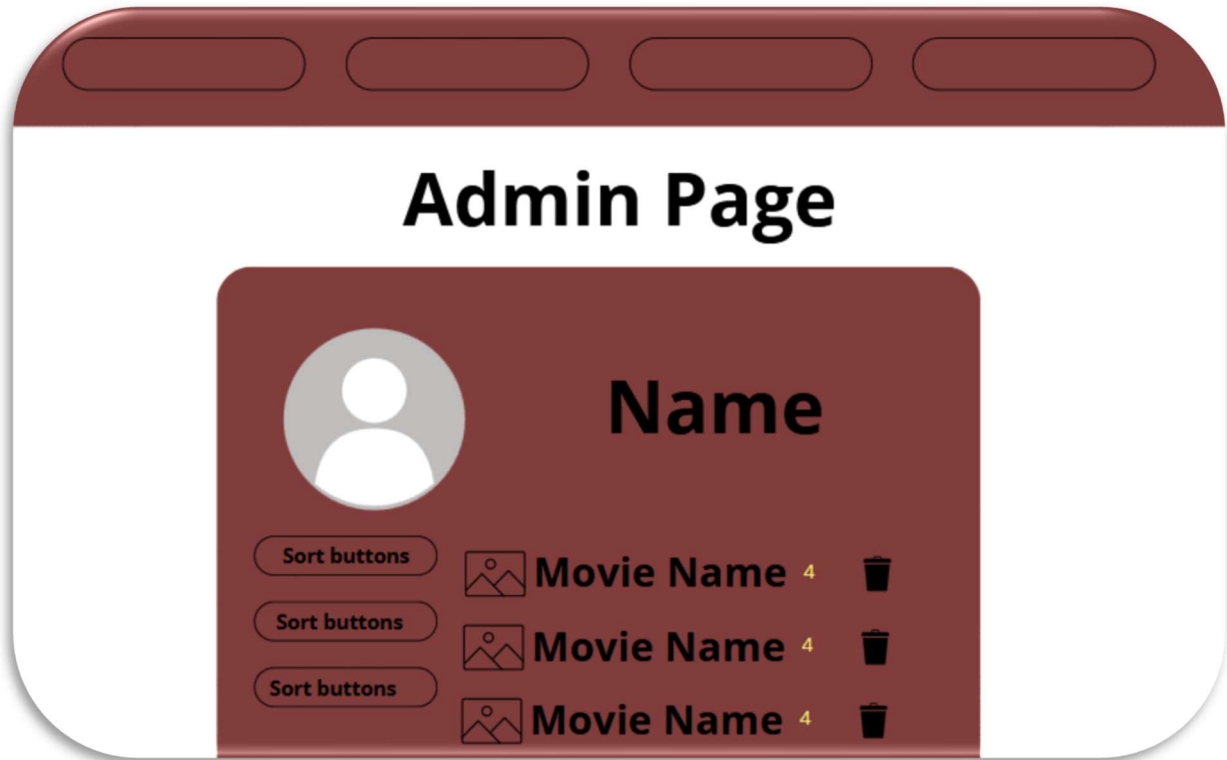
Add movie Delete movie

Input

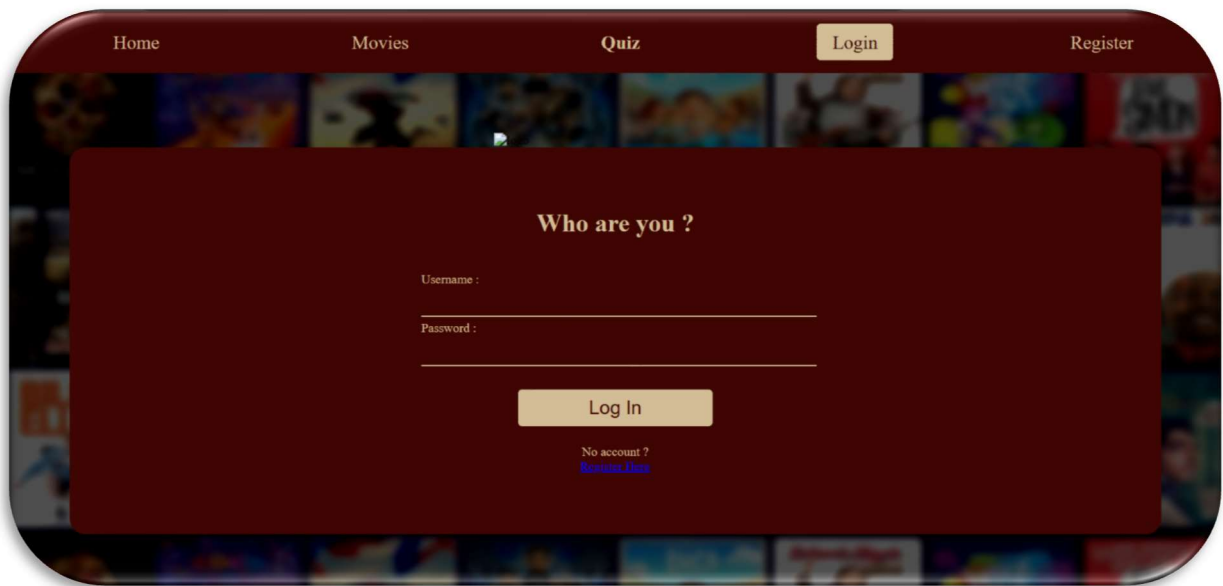
Input

Input

Input

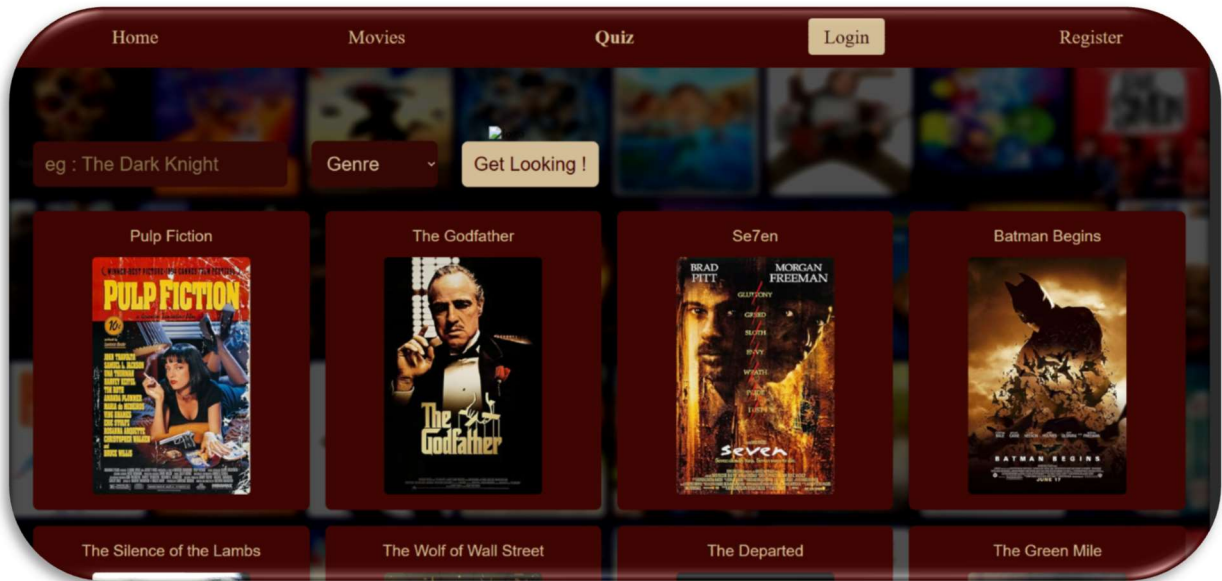


And then, we implement the front-end. When the user first come to our website we suggest him to login/register to get access to all the website functionalities, so we direct it to the login page:

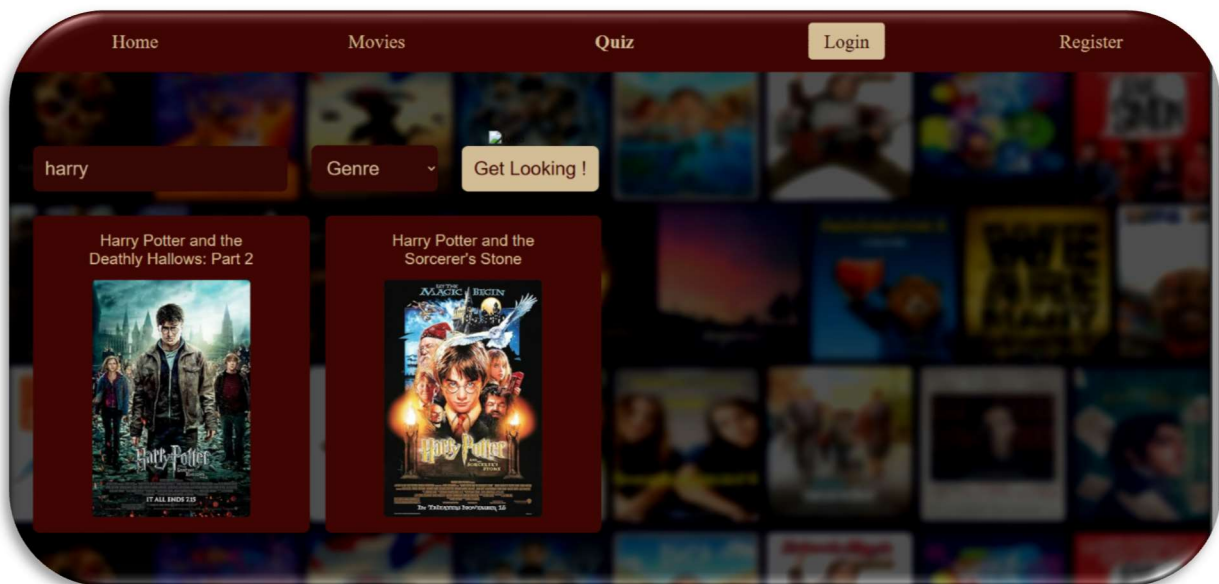


Once logged in or not, he can access the website without problem and go see the movies we are presenting

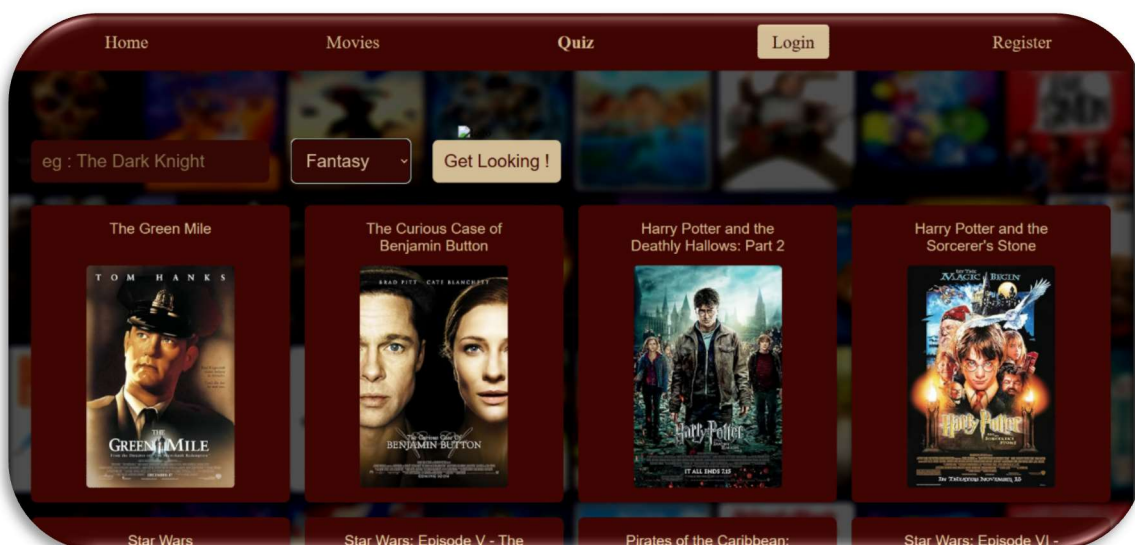
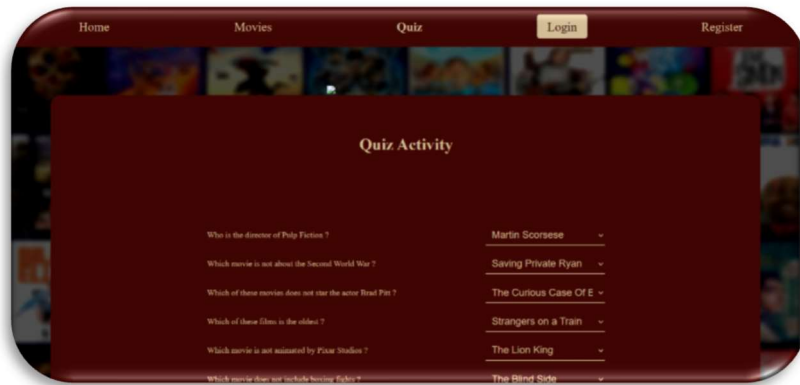




If he wants to, he can search for a specific movie name or part of it to get an easy access to it.



He can also access our quizz page to test his knowledge.



He can also see a movie detail like the realisateur, its release year, a quick synopsis, the genre and a part to rate it if you have seen it.



## Implementation

To deploy our website, firstly we used Vercel; we created an account, and we linked with the github, but there are some struggle with the front-end, and more particularly with the technologies that we used :

**404: NOT\_FOUND**

Code: `NOT\_FOUND`

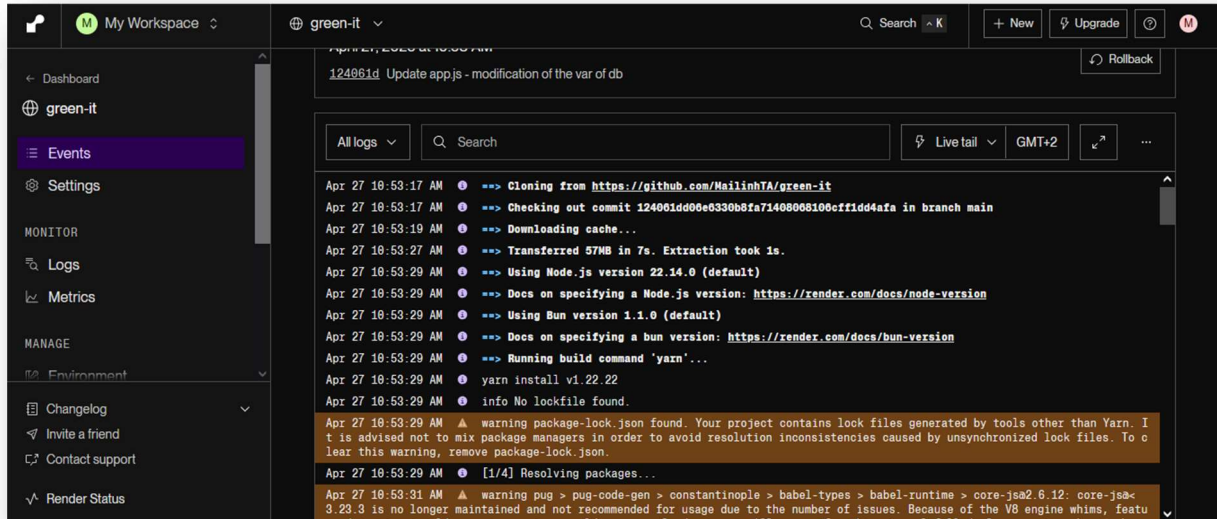
ID: `cdg1::h226f-1745578718789-12597f35faab`

[Read our documentation to learn more about this error.](#)

So we tried on other host server such as render that is user friendly for the front-end but for the back-end, we had some difficulties to move from a local database to an online database and then to link the front-end and the back-end. Indeed, it is more simple to use PostgreSQL. However, we have not choose this solution, because we already had an online database on PhpMyAdmin, and multiply the database will increase the carbon footprint:

Key	Value	
DB_HOST	.....	👁
DB_NAME	.....	👁
DB_PASSWORD	.....	👁
DB_PORT	.....	👁
DB_USER	.....	👁

Moreover on Render, we can access easily to the log, so we can debug fastly any problem :



## Carbon footprint analysis

To measure the carbon footprint of our website, we used 'Website Carbon Calculator'. We just had to share the link of our website, and then, we had the following result :





## Website Carbon Badge

Tell your visitors about your carbon emissions

### Show your carbon credentials

The badge can be added to the footer of any website to automatically calculate and display the carbon emissions of each page.

### Open Source API

We have an API for Website Carbon available for use in integrating website carbon calculations into other tools and workflows.

[Get the badge!](#)

0.3g of CO<sub>2</sub>/view **Website Carbon**

Cleaner than 99% of pages tested

Dark theme

0.3g of CO<sub>2</sub>/view **Website Carbon**

Cleaner than 99% of pages tested

Light theme

As a comment, we got these :



Only **0.03g of CO<sub>2</sub>** is produced every time someone visits this web page.

[How do we calculate this?](#)



Oh no, it looks like this web page uses **bog standard energy**

← If this site used green hosting, then it would emit 9% less CO<sub>2</sub>

Indeed, we used Render without verifying if it was a green hosting solution. Knowing that now, we tried to search for better green host, but when we searched solutions, most of the best solutions are not free. So we stayed on Render for now.

## Discussion and Conclusion

### Eco-Responsible Practices in Web Development

The aim of this project is to focus on the importance of eco-responsible practices in web development. By choosing the right technologies for our need, we can significantly reduce the environmental impact of web applications.

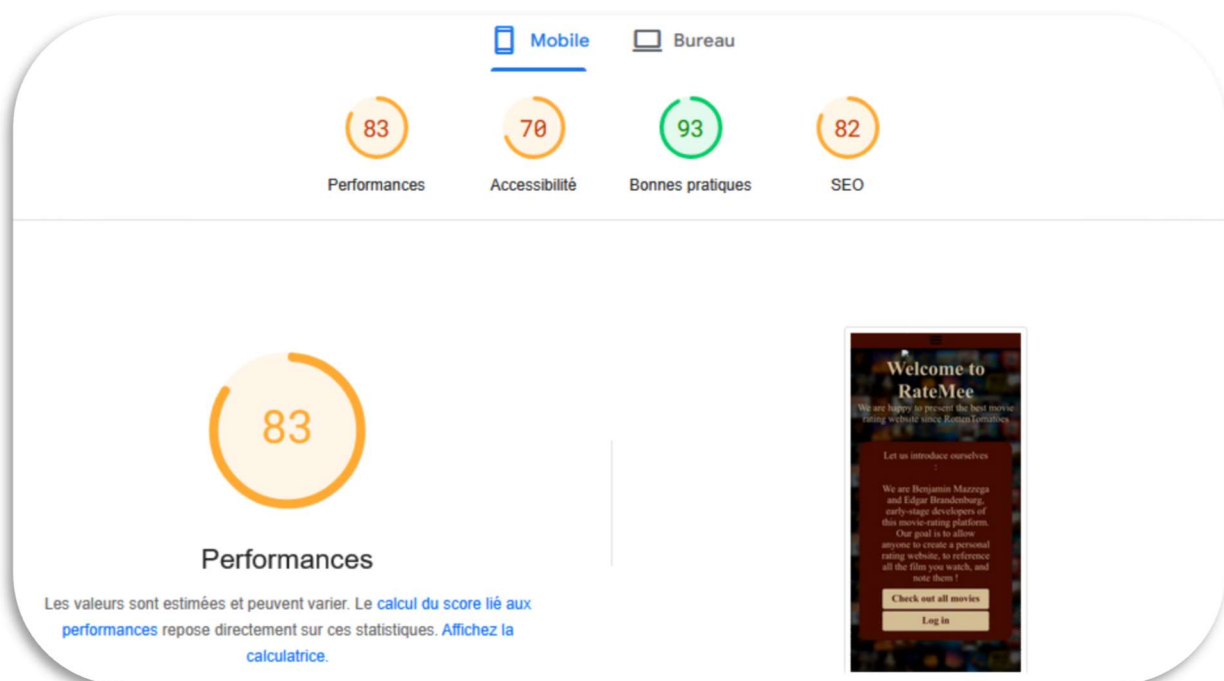
## Challenges faced

One of the main challenges is to maintain site functionality while reducing the carbon footprint. In the beginning, we try to code basic functions, but the website is not user-friendly, so we browse the different technologies in our disposition to improve design the website and not impact significantly the carbon footprint. This project requires continuous optimization and monitoring of resource usage.

## Ideas for future improvements and integration of ecological principles

Some future improvements include the exploration of hosting solutions that use renewable energy and provide tips for reducing their digital footprint by showing them the impact of their uses graphically.

We used the tool from Google Chrome : Lighthouse to measure the performance of our website and what we can improve. On the mobile format :



▲ Dimensionnez correctement les images — Économies potentielles de 486 Kio

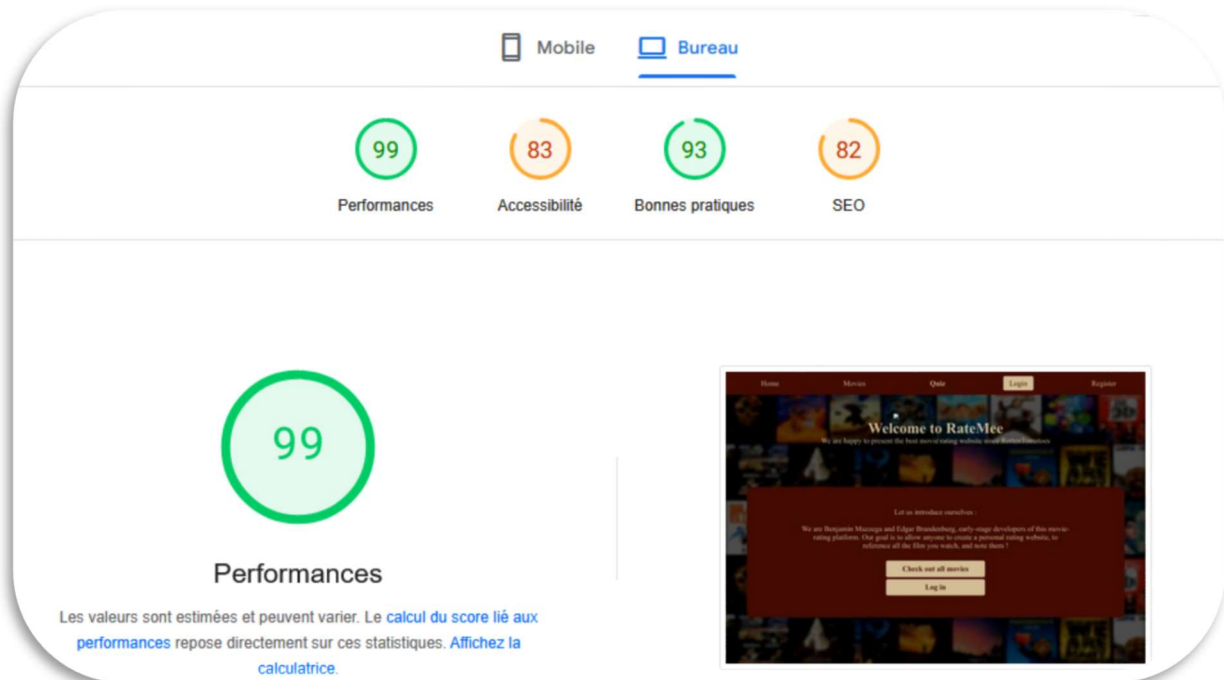
▲ Diffusez des images aux formats nouvelle génération — Économies potentielles de 268 Kio

Les formats d'image comme WebP et AVIF proposent souvent une meilleure compression que PNG et JPEG. Par conséquent, les téléchargements sont plus rapides et la consommation de données est réduite. [En savoir plus sur les formats d'image récents](#) [LCP](#) [FCP](#)

URL	Taille de la ressource	Économies potentielles
onrender.com <a href="#">Propriétaire</a>	707,9 KiB	267,7 KiB
 background-image  /images/background.jpg (green-it.onrender.com)	707,9 KiB	267,7 KiB

▲ Le document ne contient pas d'attribut "meta description"

On the desktop, we have the following result :



## Conclusion

This project demonstrates that it is possible to build a functional and user-friendly. By making conscious technical choices and continuously optimizing performance, we can contribute to a greener and more responsible digital future.