English project Report

Clément COUCHEVELLOU, Louis-Vincent CAPELLI

April 2023

Contents

Chapter 1

Introduction

1.1 Subject

This project aims to help one improve his English language skills, so we chose to realise a web application, which is the best way to share our work to as many people as possible.

1.2 State of art

English vocabulary learning applications such as Duolingo offer a convenient and effective way for users to improve their language skills. However, many of these apps are not accessible enough due to their cost, as they require users to pay for premium features or subscriptions. This can be a significant barrier for learners who do not have the financial resources to invest in these tools.

Furthermore, even if the app is free, requiring users to create an account to access English vocabulary learning apps can be inconvenient and may also raise privacy concerns for users who are hesitant to share their personal information.

1.3 Our choice

We recognized the importance of making our application user-friendly and easy to use, which is why we made it a priority during the development process. The whole application is made to be a part of users daily routine,

allowing them to spend just a few minutes each day improving their language skills.

Chapter 2

User guide

Contents

2.1 First glance of the application

EnglishGenius is designed to be easy to understand and use, without the need for accounts, settings, or configurations. Our focus is on providing an enjoyable and effective learning experience through the use of multiple-choice questions. We chose MCQs because they are attractive, easy to use, and particularly effective when repeated. We have decided to focus on vocabulary learning, as this is enjoyable for users and well-suited to MCQs.

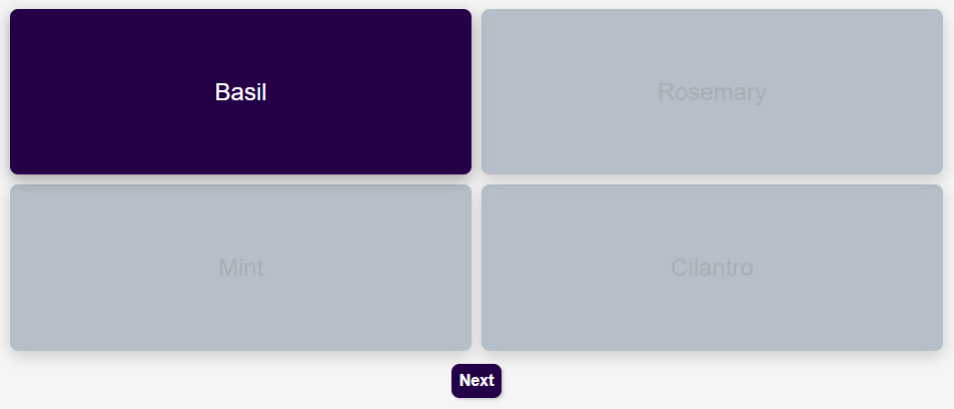

Figure 2.1: MCQ example

2.2 Main features

2.2.1 Training mode

The home page contains the "Training Mode". The user can start learning the exact moment they enter the website and for the exact duration they want. The questions follow one another and the user decides when to move on to the next one to promote learning at the pace of each.

2.2.2 Themes

There are multiple vocabulary themes available in the application to help users work on subjects they enjoy, but the daily theme is perhaps the most important. Every day, a new unique theme is available to encourage users to use the app daily and ensure they have their daily practice. The list of themes is designed to be expandable over the time and previous "Topics of the day" remain available in case users missed them.

Figure 2.2: Themes selection

2.2.3 Time attack!

We have also added a "Timer Mode" to our application, where users have 60 seconds to answer as many questions as possible. This mode is designed to motivate users who love a challenge, and we have included a leaderboard to showcase the top scores in Timer Mode. It adds a fun and competitive element to the learning experience, encouraging users to keep practicing and improving their skills. Indeed we believe that adding a timer may push users to practice until they know the vocabulary by heart and do not hesitate on the right answer which will benefit them a lot for their oral expression.

Figure 2.3: Button for the "Timer Mode"

After further thinking, we decided not to skip the questions automatically after they got answered in the "Timer Mode". We believe that this application is primarily a learning tool, and we want to ensure that users can see and understand their mistakes instead of just clicking as quickly as possible to get a high score in the "Timer Mode."

Figure 2.4: Button for the leaderboard

When the timer ends, a popup window appears to summarize the game and ask the user for their nickname. They can also not provide a name and the game will be registered as "Anonymous".

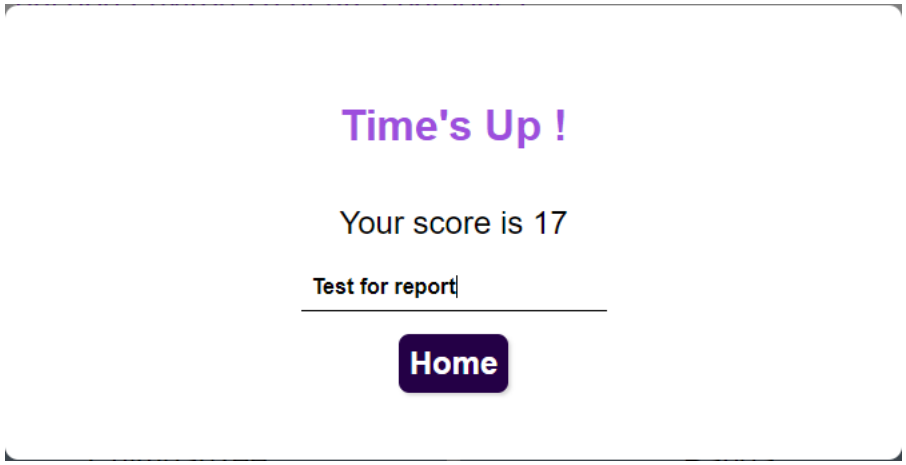

Figure 2.5: Popup window that appears at the end of a game

The best players are featured in the leaderboard!

Rank	Nickname	Score
1	PGM	35
2	LV	29
3	LV ;)	29
4	Test for report	17
5	CC	10
6	LVVVVV	7
7	Malo	6
8	Anonymous	4

Figure 2.6: Leaderboard

Chapter 3

Technical guide

Contents

3.1 Technologies chosen

When we decided to build a website, we knew that using a JavaScript framework would be a smart choice, as they are the most used today and we wanted to learn more about it, so we searched for options.

Then we chose to use ReactJS for our project because it was ideal for creating an intuitive, user-friendly, visually appealing application. Additionally, since ReactJS is one of the most popular technologies for web applications, we were eager to experiment with it, especially since it was not covered in our coursework at Télécom Nancy.

3.2 Vite

Vite is a build tool and development server that is specifically designed for modern web projects. It provides fast and efficient development.

In the context of our ReactJS application, we chose to use Vite as our development server because it allowed us to quickly set up a development environment, and also provided us with many useful features such as HMR (Hot Module Replacement).

3.3 Structure

We developed our application using the ReactJS web framework, which allowed us to create a dynamic and interactive user interface. For the back-end, we chose Node.js, and used the Express.js framework to create an API that manages the application's data, we used Vite as our build tool and development server.

ReactJS is a powerful web framework that enables developers to create dynamic and interactive user interfaces for web applications. The component-based architecture makes it easier to use in team and help developping complex applications, by allowing us to break-down the different pages in smaller elements.

3.3.1 Frontend

For the frontend of our React application, we used Sass as our CSS preprocessor. Sass allowed us to write more efficient and maintainable CSS code. This made it easier to organize and customize our styles. Additionally, Sass provides a wide range of tools and features that can help speed up the development process and improve the quality of our CSS code.

3.3.2 Backend

The backend of our application was built using JavaScript with Node. For managing the different pages of the application, we chose to use 'react-router-dom,' a well-known, easy-to-understand, and effective way to do it. The questions and answers in our application are managed by our API, and then fetched with JavaScript.

3.3.3 ExpressJS API

The questions and answers in our application are stored by themes in JSON files, and the API routes the files to the correct routes. To enable our ReactJS application to access the routes, we use the Vite built-in proxy to redirect the API URL to the URL of the ReactJS application.

Chapter 4

Conclusion

4.1 Personnal feedbacks

We enjoyed working on this project, as we were truly free to choose what we wanted to do and the ultimate goal was to develop an application that would have a real-world impact and help people. The opportunity to choose our groupe and the coding languages allowed us both to learn ReactJS and work together on a web application for the first time which was a great experience too.

4.2 Trail for the future

Although our application was originally designed to learn English, it could easily be adapted for use in learning other languages as well. In terms of organization, we could incorporate different difficulty levels with a prior language level test or utilizing other organizational methods. We could also consider developing alternative versions of this application that focus on other language learning areas, such as grammar or conjugation. Additionally, an audio comprehension version could be developed to improve listening skills. While we made our application easy to understand and use, there is still room for improvement and a mobile version would fit perfectly into our goal of making learning English accessible.

Chapter 5

Sources

- fr.reactjs.org
- react.dev
- developer.mozilla.org
- icons8.com
- v5.reactrouter.com
- https://codepen.io/bastianalbers/pen/PWBYvz