Final Version Report

Author: Bc. Tetiana Hrunyk

Project Name: Vocs

GitHub repository: https://github.com/TetianaHrunyk/VocabularyLearningApp

Link to the public instance of the project: http://vocsapp.herokuapp.com/

Features:
Currently implemented:
Users can create account and log in. Registered users can paste their text in the text are, parse it and translate selected words. Users can join pieces of parsed text by merging individual parts, which allows to translate not only single words, but phrases or even sentences.
Registered users also can create empty decks and save their translations as cards in those decks. Users can also browse the cards in those decks, add custom cards, delete and edit them.
Finally, if users click *Study* tab and select deck, they can revise the cards that have been added recently and which they have trouble recalling. By clicking *Got it* under the card they increase their progress, and by clicking *Show later* they decrease it. The progress on each card is automatically decreased every day, so users should revise them regularly.
In progress:
I am considering a better algorithm for updating the progress on the cards. Currently it is a simple addition and subtraction from a number that represents the progress. It might be useful to calculate the progress based on how time which went by since the cards was added.
Currently not implemented:

Time spent: 60 hours

Problems that have occurred in the working process:
One major issue was to avoid unnecessary rerenderings of React components, especially when it comes to sending either API requests or requests to back-end server. There still are some improvements that can be made in the project.
I chose to use Django framework, which was new to me, so it took some time to figure out how to work with it. In particular, I had troubles with the authentication tokens (getting a token, saving it into local storage and updating them regularly) and scheduling regular updates of the database (to decrease the progress on the cards every day).

What I would do differently if I were to do this project again:
Firstly, I would do all the setup simultaneously. In my case, I firstly initialized a new React project and worked on frontend. Then I initialized Django project in the same folder, and had lots of troubles trying to set up all the paths and modifying Django settings. Then I initialized heroku in the same folder, and had even more troubles trying to prepare all the files for deploying the app on heroku. Using Django and React together and deploying projects with this technological stack is actually easy, but I did all the setup in reversed direction, which I certainly will avoid in the future.
I would also use axios library for handling web requests. I knew that such a took exists, but I was too lazy to learn how to use a new library, so I decided to stick to vanilla fetch requests, which eventually led to longer and less readable code with lots of error handling. As I learned later, with axios I could have avoided some of that.

Personal feedback about the project:

I learned a few new technologies while working on the project, which makes me glad I had the drive to do that. If I did not have to do this project, I probably would have been too lazy to learn so much new stuff.
I admit that there is a lot of space for improvement n y project, but I am rather proud of it, because in the end that is a working application, working on which gave me so much new experience.
The most high-quality part is the Django server, since the code in that part is short, clean and there is nothing extra. But, at the same time, that part was guided by the Django principles and architecture, so that is not really my achievement. The part I am most proud of is the drag-drop-merge on the parsed text at homepage. I used additional libraries to do that, but putting everything together took some effort. I am glad that eventually it works the way I wanted.