Pocket Lingo

EAR 2021/22

Anna Kachmasheva, Mukan Atazhanov, Ivan Shalaev

Project **repository**

1. Introduction

1.1 Project overview

A project for memorizing foreign words based on the flashcard and periodic learning repetition methods. Flashcard is an abstract two-sided card, the word in the user's native language is written on the hidden side, and the word in the chosen language is written on another side. User creates flashcards by himself or downloads other user's cards from the card repository. Then the user enters the learning mode and browses cards. If the user knows the translation of the word, he clicks on the button "I know" or "I don't know" in the opposite case. The periodic learning repetition algorithm shows flashcards of words that the user does not know more often than those that he knows.

1.2 Project scope

The purpose of the application is to ease language learning via flashcards and to engage new users to study foreign languages. The main idea is to allow users to simply and fastly create flashcards and share their own flashcard collections with others. Translating and searching services AREN'T included, we'll use some side applications API.

1.3 Intended audience

People who want to learn foreign words using flashcards and periodic learning repetition methods.

1.4 Not implemented

In front-end

1. Creating new deck
2. Creating new card
3. Google Auth
4. Admin page
5. User page
6. Games

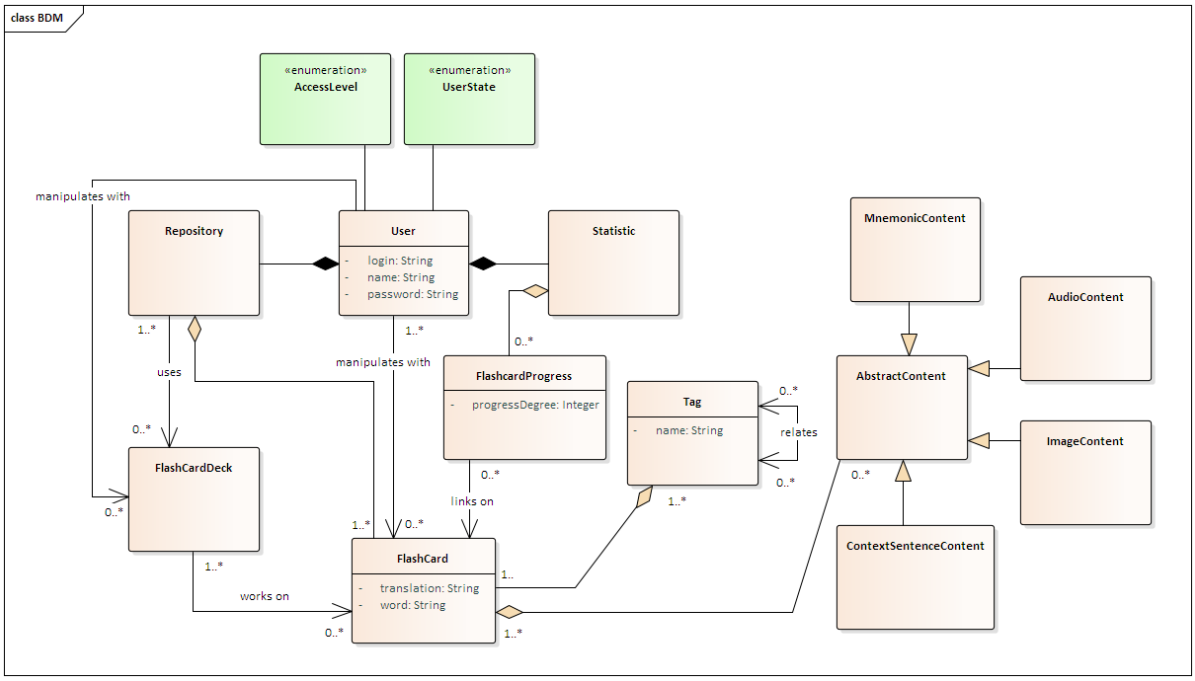
Overall

1. Achievements
2. Card progresses
3. Only 1 game implemented

2. Overall Description

2.1 Product features

The main features of the system are presented below using the UML diagram.



2.2 User roles and characteristics

User roles

- Administrator
- Registered user
- Unregistered user

Application functionality

UNREGISTERED USER FUNCTIONS:

- Registration using regular registration and Google registration.
- View the main page.

REGISTERED USER FUNCTIONS:

- Update/read his personal data such as login, mail, password.

- Delete his account.
- Create/read/update/remove a flashcard/flashcardDeck/content in his repository.**(ONLY ADMIN)**
- Add/remove content(AudioContent, ImageContent, MnemonicContent, ContextSentenceContent) to the flashcard.
- Read/add in his repository other user's flashcardDecks if they are public.
- Create/read a tag.
- Add/remove tag to flashcard.
- Search for flashcards by tag/name.
- Sort flashcards by flashcardProgress/name.
- Read/clear his statistics. The statistic includes the progress of each flashcard in his repository.
- Read his achievements.
- Registered user can use the following modes:
- training mode
- mini game Char-by-char
- mini game Find the pair
- mini game Choose the correct translation of a word from the five suggested

For each of the above modes, the registered user can change the settings: number of cards/set of cards. In addition, the user will be able, upon completion of the mode, to see his results and the overall rating of all users.

ADMINISTRATOR FUNCTIONS:

- Create/read/update/remove users/flashcards/flashcardDecks/tags/contents.
- Block/unblock a user.
- Read a list of all users/repositories with all flashcards/statistics.
- Read the list of all flashcardDecks, even if they are private.
- Add/remove flashcard/flashcardDeck from one repository to another.
- Search/sort users/flashcards/flashcardDecks/tags/statistics.
- Clear statistics for all registered users.
- Read/update/remove flashcardProgress/achievements for all registered users.

2.3 System constraints

2. The system does not have a word translator. Google Translate API will be used.
3. The system does not provide the registered user with the ability to create an individual word learning plan.
4. The system will not send the user notifications to his email.

2.4 System information

Within the semester project, this system will be implemented as a backend

Java application, using SpringBoot technology. The application will be connected

to the PostgreSQL relational database and it will be possible to communicate with it via HTTP queries or via web browser. Frontend will be implemented by using ReactJS. Also some features of the application will be implemented by using Google API.

3. Instalation

3.1 Basic Information

Application is needed to be started on web server. Before starting need to configure settings of application as link to database. In **src/main/resources/application.properties.** Our application uses PostgreSQL.

Example:

```
spring.datasource.driver-class-name=org.postgresql.Driver
spring.datasource.url=jdbc:postgresql://localhost:5050/postgres
spring.datasource.username=postgres
spring.datasource.password=helloMyFriend
```

Application can be used in two ways. First, simple backend application with HTTP/JSON requests and Second, with simple front-end.

To start with front-end will be needed to install all compuslory **node_modules** by command in terminal **–npm install** and after call **–npm start.** To start without front-end. Run LingoApplication in IDE.

Application has been tested by Postman. In our repository can be found our simple postman scenario. Before starting testing, a user with the Admin role must be registered in the database. The data for creating an administrator should be as follows

```
{   "username": "admin",
    "mail": " admin@gmail.com",
    "password": " admin"}
```

Then import the script collection and run it.

More INFO in README.md

4. Conclusion

4.1 Experience

Ivan Shalaev:

I tried adding to the project the high-level construct *@RepositoryRestResource* with *CrudRepository* extending (to substitute DAO, Service and Controller as well), but it turned out to be very inflexible so I was forced to come back to older structure and figure out with each single layer manually.

Mukan Atazhanov:

Na začatku 3. semestru měl jsem velké problémy s javou. Jsem si myslel, že Spring vůbec nezvládnu, ale díky cvičením jsem poznal Javu z jiné strany. Práce v týmu a se Springem byla dost příjemnou zkušenosti. Dostal jsem real zkušenosti jako vytvoření JPA model, REST a úplně základní Front-End. A také jsem zjistil, jaký je super vhodný POSTMAN na hledání děr v RESTu. Za mě předmět je 5 out of 5.

Anna Kachmasheva:

Moje hlavní práce byla práce na Services, REST a Spring Security. Počáteční práce na servisech nebyla nejlepší, dostali jsme mnoho upomínek z CP1. Při práce na REST, kterou jsme dělali všichni společně, se servisy výrazně změnily. Pro mě nejtěžší byl REST, a hlavně proto, že jsme poté začali aktivně testovat. Testováním se nám podařilo odhalit nedokonalosti projektu. K tomu byl použit Postman. Použila jsem ho poprvé a práce s ním pro mě byla příjemným překvapením. Tento program má tak jednoduché rozhraní. Zkusila jsem také napsat malý scénář. Pří práce se Spring security mě silně inspiroval project ze cvičení. Když bylo nutné přidat token, byla jsem zmatená. Do této chvíli jsem nikdy nepracovala s tokeny. Nemyslím si, že realizace, která je aktuálně v projektu je nejlepší. Moje prací bylo také pracovat s externí autentizační službou Google. Tento úkol ještě není dokončen, zbývá se spojit s frontendem. Práce na tomto projektu mě opravdu bavila. Vyzkoušela jsem spoustu nových věcí. Ne všechno se povedlo napoprvé, ale celkové jsem s prací svého týmu spokojená.