Project : FlashCard App V2

Modern Application Development - 2

Author - Dibyendu Laha
Roll - 21F1006498
Email id - 21f1006498@student.onlinedegree.iitm.ac.in

★ **About myself :**

I have completed my Bsc. Hons. in Mathematics under Kalyani University. I am currently involved fully into this degree program and also a IITM Bsc program Ambassador and GL of Kanha House Group id 2. I am interested in learning and making changes in our day to day life with AI and Machine Learning. I am a fresher in the field of coding and App Development. In my leisure time I love reading books and doing meditation. Also I love to play Chess.

★ **Description :**

In this MAD2 final project we are told to implement Vue js in our previous MAD-1 project with APIs and proper token based authentication. We have to implement some other features also like generating monthly progress reports, daily reminders and exporting csv etc.

★ **Technologies used :**

- UI with Vue and Vue Components
- Flask_login- to login and logout for every registered user
- Flask_restful for APIs.
- Flask_security – For every page of a registered user to secure the application with a login security
- Flask_security.utils-To auto generate a hash password for every registered user in a most secure manner that if for some reason anyone gets access to the database the password won't be understandable.
- SQLite3 for database
- Token based authentication for user login
- SQLAlchemy to communicate between applications and Sqlite3 databases.
- Sqlalchemy.sql.functions – To track date-time of every user activity
- CSS and Bootstrap for styling

★ **DB Schema Design :**

- **Table names**

➢ **"user"** ("id" INTEGER NOT NULL, "f_name" VARCHAR(100) NOT NULL, "l_name VARCHAR(100) NOT NULL, "email" VARCHAR(100), "password" VARCHAR(255), "active"BOOLEAN, "uscore" INTEGER, PRIMARY KEY("id"), UNIQUE("email"))

➢ **"deck"** ("id" INTEGER NOT NULL, "title" VARCHAR (255), "user_id" INTEGER NOT NULL, "last_view" DATETIME,"deck_score" INTEGER, PRIMARY KEY("id"), FOREIGN KEY("user_id") REFERENCES "user"("id") ON DELETE CASCADE)

➢ **"card"** ("id" INTEGER NOT NULL, "front" VARCHAR (255) NOT NULL, "back" VARCHAR (255) NOT NULL, "deck_id" INTEGER NOT NULL, "user_no" INTEGER NOT NULL, "last_review"

DATETIME, "cscore" INTEGER, FOREIGN KEY("user_no") REFERENCES "user"("id") ON DELETE CASCADE, FOREIGN KEY("deck_id") REFERENCES "deck"("id") ON DELETE CASCADE, PRIMARY KEY("id"))

- ➢ **"role"** ("id" INTEGER NOT NULL, "name" VARCHAR(40), "description" VARCHAR(255), PRIMARY KEY("id"))[Not used]
- ➢ **"roles_users"** ("user_id" INTEGER, "role_id" INTEGER,FOREIGN KEY("role_id") REFERENCES "role"("id"), FOREIGN KEY("user_id") REFERENCES "user"("id")) [Not used]

—--- I have designed the schemas like this because it will be easy to track and query the Decks made by the user for him/her and also the same for the cards under each deck.

★ **API Design**

I used postman to test all my APIs. I used my APIs in my whole app for each and every request url. There are 3 different classes for all my APIs, they are *Users*,*Decks*,*Cards* and from these 3 classes I generate all my APIs.

- ➔ All user related APIs are there in *Users* class like API for user get,post and delete requests. Here I define corresponding error code for every request method.
- ➔ In *Decks* class there are APIs for get,put,post and delete requests which will be required when a particular user wants to access,update,create or delete his/her decks.
- ➔ In a similar manner there is one more class named *Cards* for corresponding each deck. These APIs will be required when a user will want to create,access,update or delete their cards from a particular deck.

★ **Architecture and Features**

- In app.py i have imported all required packages for it. Here I initialize my app with security and db which is imported from security.py and models.py respectively.
  - For API i have resource.py which is under the api folder where all APIs are defined. For parser and request arguments there is parser.py and request.py.
  - Under the template folder there is only one html that is index.html.
- In the Static folder there are the images I used for my Web App. Also I have some js files like app.js,routes.js which are in there. Under the component folder there is config .js which generates the baseurl and all other js files which I need to generate for my app.
- In the requirements.txt file all the packages are listed that are to be installed before running the app in the local environment.
- I have used many features that need Bootstrap.
- Also for Designing I have used internal css and external css and sometimes inline css also.
- I have taken a lot of help from the websites 'w3schools, stackoverflow,mozilla developer etc' to implement the functionalities of this project.

★ **Video**

Link - https://drive.google.com/file/d/1dyYrlIzkkaRb81Zmf1h73B-6xkEDvxEQ/view?usp=sharing