FlashCardApp V2

Modern Application Development - 2

Author - SUMISTHA SAHA
Roll - 21F1000276
Email id - 21f1000276@student.onlinedegree.iitm.ac.in

About myself – I have completed my Mathematics Hons in 2020 and now I am stepping into the world of Data Science and Programming with the help of IIT Madras . Currently doing IITMOD as full time. Want to involve myself more in this unique & wonderful program to gain more knowledge . I believe that our daily life is nothing but a math equation and I love to solve this math problem. Drawing is my hobby.

Description – In this MAD2 "FlashCardApp V2 '' project I have to create a local hosting web application with the help of Vue named "Flashcard" which can be used for memory training . The core functionalities are more or less same as last time, but this time we are using Vue to implement our project with a lot more interesting functionalities. Also API and token based authentication is mandatory for this time.

Technologies used -

- SQLite3 for database
- Flask_restful for API
- UI with Vue and Vue Components
- SQLAlchemy to communicate between applications and Sqlite databases
- Flask_security to secure the application with a login for every view page
- Token based authentication for user authentication
- Flask_login- It helps to login and logout for every registered user
- Flask_security.utils-To auto generate a hash password for each and every registered user. It is the most secure method to store the passwords in the database. Suppose anyone gets access to the database but he/she cannot understand the user given password
- Sqlalchemy.sql.functions – To track date-time of user activity
- For styling CSS(inline+external)

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),"date_created" DATETIME, "active" BOOLEAN, "score" INTEGER DEFAULT 0,PRIMARY KEY("id"), UNIQUE("email")).

➔ deck ("id" INTEGER NOT NULL, "title" VARCHAR(255), "user_id" INTEGER NOTNULL , "date_created" DATETIME,"score" INTEGER DEFAULT 0, "last_rev" DATETIME, FOREIGN KEY("user_id") REFERENCES "user"("id"), PRIMARY KEY("id")).

➔ card("card_id" INTEGER NOT NULL,"front" VARCHAR(512) NOT NULL, "back" VARCHAR(512) NOT NULL, "score" INTEGER DEFAULT 0,"deck_id" VARCHAR NOT NULL, FOREIGN KEY("deck_id") REFERENCES "deck"("id"), PRIMARY KEY("card_id"))

- ➔ role("id" INTEGER NOT NULL,
  "name"VARCHAR(40),"description"VARCHAR(255),PRIMARY KEY("id"))
- ➔ roles_users("user_id" INTEGER, "role_id" INTEGER,FOREIGN KEY("role_id") REFERENCES "role"("id"), FOREIGN KEY("user_id") REFERENCES "user"("id"))·
  [Although the table named role and roles_users have not been used anywhere to store user information.]
  - ★ Reason behind designing it this way – I designed the above schema this way because when a new user registers and signs in to use this application ; the above structured schema can easily link with proper user given information to track and store information in the corresponding db tables.

API Design

Basically I used insomnia and postman to test all my API designs. I used my APIs throughout my whole app for each and every request url which i can implement in my application
So there are 3 different classes for all my APIs, like *User ,Deck ,Card* and from these 3 classes I implement all my APIs.

- ★ My first class for API is **User**, which is used for all user requests like user registration,signin,delete user etc. So for these requests there are corresponding functions for the post,get and delete request method which will be invoked by a particular request url.
- ★ In **Deck** class there are APIs for get,put,post and delete requests which will be required when a particular user wants to access the decks,update,create or delete his/her decks etc.
- ★ Similarly there is one more class named **Card** for corresponding decks. The card APIs are for updating,accessing or deleting cards from a particular deck.

So the above all are my defined APIs for my entire project.

Architecture and Features

- My main architecture for this project is Single page application without using VUE Cli.
- So for my above architecture , under requiremrnts.txt file all the required packages are listed there , these need to be installed before running this application in local development.
- In the app.py I imported required packages and initialized the app with db and security which is imported from security.py.
- In fields.py there are corresponding fields for user,deck and card.
- In models.py all required db models are there and In parser.py all needed request parsers are there.
- In the Static folder there are the images I used for my Web App,app.js,router.js where all routes are there and one components folder. In the component folder there is a user folder where all user required js files are stored.
- Html file is in the templates folder.
- For Designing I have mainly used internal css and external css and sometimes inline css also.
- For implementing page designing and features I have taken help from the following sites- “bootstrap ,w3schools ,stackoverflow “ etc .

Video

https://drive.google.com/file/d/1th8Ra1mTNsLK7hUskWVeIjkVi6-jhYmG/view?usp=sharing