Author:

Vinay Vinod Hariya

21f1006220

21f1006220@student.onlinedegree.iitm.ac.in

I am a 3rd year student currently doing a B.Tech degree in Computer Engineering at MPSTME, NMIMS (Mumbai) and the B.Sc. degree from IITM. I love creating mobile apps and websites. Also, I am highly interested in analysis of data. I had a lot of fun in creating this web app since it gave me new knowledge and opened me to new technologies.

Description:

The Flash Cards website makes it easier to study topics. Making Decks public allows other users to study it. The API makes it easy to perform CRUD operations on the decks and cards. There is also a scoring and review system. The database stores all the information.

Technologies used:

1. Flask – Micro python framework that makes creating web apps easier.
2. Flask-SQLAlchemy – Object Relational Mapper (ORM) that makes connecting to the sqlite3 easy.
3. Flask-Restful – Python flask package for making Restful API easily and efficiently.
4. HTML, CSS, Bootstrap for the user UI
5. Postman and Insomnia for API testing and documentation.
6. SQLite database to store all the information in the relational tables.

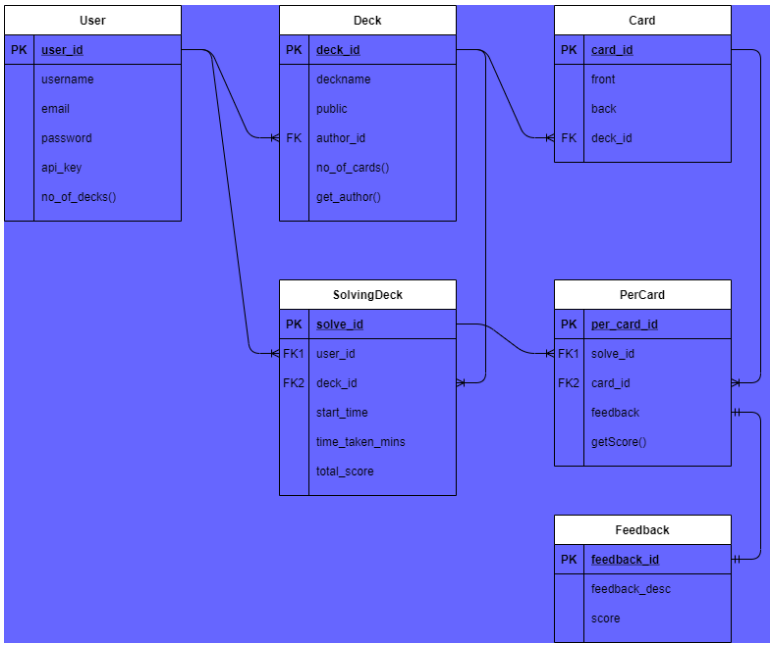
Video:
https://drive.google.com/file/d/1YZEUXDmaiDFGRoTjUCko7EHJFRLnkGhH/view?usp=sharing

DB Schema Design:

The sqlite3 database contains 6 tables:

1. User table – stores the user_id, username, email, password, api_key.
2. Deck table – stores the deck_id, deckname, author_id, public
3. Card table – stores the card_id, deck_id, front, back
4. SolvingDeck – stores the user_id, deck_id, solve_id, start_time and time_taken
5. PerCard – stores the card_id, solve_id and feedback that the user gives
6. Feedback table – stores the feedback of the card solved (eg: Easy) and score allotted.

Most of the relationships between the tables are of the form **one to many**. The API key is stored for users who want to use the API for accessing and modifying the data. The API key is used for authenticating the user to prevent misuse of the application.



API Design:

The main function of the API is to perform CRUD operations on the Decks and Cards. The API is the backbone of the website and can also be used in individually. The API created is spread over 11 categories and is implemented using Flask-Restful package. User login and register, deck and card resources and also scoring and decks attempted are all implemented in the API.

YAML file:

https://drive.google.com/file/d/1Yma8B_2VNCbHusEYvykmHAfagUMo3-0G/view?usp=sharing

Architecture:

- The api folder is stand alone and with only a few tweaks can be run on its own. It consists of the sqlite3 database file and the main api file. The helper functions and classes are also stored here.
- The application folder consists of the html templates, the authentication and main controllers. The authentication controllers are only used for registering and logging in users onto the website. The main controllers are used for the dashboard and provide an interface for the CRUD operations on Decks and Cards. The website URLs are all generated and controlled from here.

Features:

1. Secure Website - Only registered users can access the decks and the flash cards.
2. Dashboard - Contains all the decks created by the user and also a table of all the decks attempted along with links to them for easy access.
3. Ability to perform CRUD operation on the deck and decide whether to keep it private or public. Deck-page helps to perform these operations.
4. For public decks, there are only options to view the cards and study them. Thus, the decks are secure.
5. It is responsive and hence can also be seamlessly used on Mobile Phones.