Author

Yuvaraaj E
21f1004566
21f1004566@student.onlinedegree.iitm.ac.in
I'm currently in my pre-final year doing BTech IT in Pollachi. I love to develop software and games.

Description

Firstly all the rendering done by jinja2 has to be changed to vue to achieve reactivity. Providing authentication using auth tokens. Generation of report monthly and send it to user's mail, exporting decks and reminding user to review daily on google chat.

Technologies used

- flask - Makes back-end development simple and flexible
- flask_restful - To implement RESTful API faster.
- flask_sqlalchemy - To work with database easily.
- flask_security - Inbuild libraries to authenticate users.
- flask_cors - To solve COR problem in SwaggerUI
- **flask_caching – To cache results to improve performance**
- Bootstrap - Quickly develop beautiful websites.
- Celery – Provides task queue and task scheduling.
- Redis – Used as a message queue as well as result backend.
- Vue JS – Used to make UI faster and reactive.
- MailHog – Used to test SMTP protocol for sending reports through email.

DB Schema Design

1. user
   - id - INTEGER, PRIMARY KEY, AUTO INCREMENT
   - email - STRING, NOT NULL, UNIQUE
   - username - STRING, NOT NULL
   - password - STRING, NOT NULL
   - active - INTEGER
   - fs_uniquifier – INTEGER
   - decks_deleted – INTEGER
   - curr_streak – INTEGER
   - highest_streak – INTEGER

2. cards
   - card_id - INTEGER, PRIMARY KEY, AUTOINCREMENT
   - front - STRING, NOT NULL
   - back - STRING, NOT NULL
   - score - INTEGER

- count - INTEGER

3. decks
    - deck_id - INTEGER, PRIMARY KEY, AUTOINCREMENT
    - name - STRING, NOT NULL
    - score - INTEGER
    - last_reviewed - INTEGER
    - tot_score - INTEGER
    - cards_deleted – INTEGER

4. userdecks
    - user_id - INTEGER, FOREIGN KEY REFERENCES "user"("id")
    - deck_id - INTEGER, PRIMARY KEY, FOREIGN KEY REFERENCES "decks"("deck_id")

5. deckcards
    - deck_id - INTEGER, FOREIGN KEY REFERENCES "decks"("deck_id")
    - card_id - INTEGER, PRIMARY KEY, FOREIGN KEY REFERENCES "cards"("card_id")

API Design

Basic CRUD operations for both card and deck, and endpoint to get user info have been implemented by using flask_restful. Then other informations like getting user, card, deck etc are implemented in flask to be used with fetch API.

Architecture and Features

Database, main.py are in root folder. Scripts and style files are in static folder. Application python scripts are in application folder and HTML files are in templates folder.

Mostly all jinja2 (except flash error) are replaced by vue rendering. All the states are managed in client side and updated asynchronously to backend. Using celery and webhooks user gets notified in google chats when not reviewed for a day. All the endpoints containing user details except some API will be authenticated using auth tokens. Monthly reports will be generated containing user stats for that month. Used celery to asynchronously export csv file containing deck information. Sending the report generated through mailhog. Added caches to improve app's performance.

Video

https://drive.google.com/file/d/1Fwehu2XCleEEen72HgNKRhclKY-lGnC6/view?usp=sharing