FlashCard-Final Project

Introduction

Patil Dipak Hemant

21f1004451

21f1004451@student.onlinedegree.iitm.ac.in

Another Mech guy in IT, Last year student, Enthusiastic web developer with an experience of over 1.5 years, Freelancer, Love Music, Sci-Fi Movies, Web series, and Cricket.

Description

In this project, according to me, we have to create a web application that can be used for practice questions. It can also be used as an Exam Portal, which can be used by either teachers or students.

Technologies used

1. **Mandatory Technologies**
   a. **Flask v2.0.2** for application code
   b. **Jinja2 v3.0.2** for templates HTML generation
   c. **Bootstrap v5.1.3** and **FontAwesome v5.3.1** for Styling and icons respectively
   d. **SQLite** for data storage and **Flask-SQLAlchemy v2.5.1** as SQL Toolkit.
2. **Other Technologies**
   a. **Email-validator v1.1.3** for validating email
   b. **Flask-Login v0.5.0** for authentication
   c. **Flask-RESTful v0.3.9** for Restful APIs
   d. **Pandas v1.3.4** for importing and exporting cards in Excel file
   e. **jQuery v3.6.0** for easy handling of DOM.

DB Schema Design

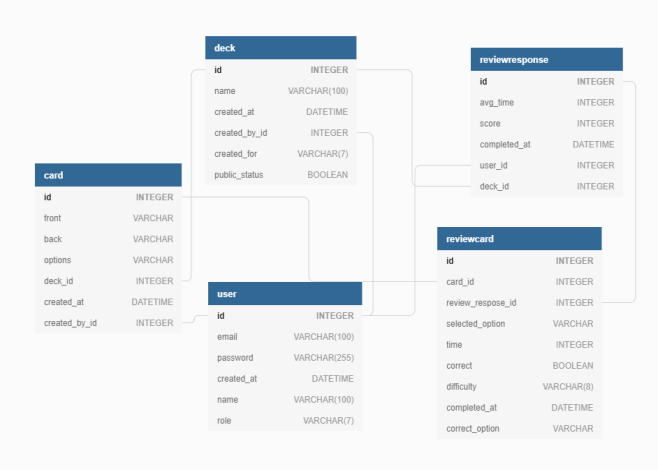

Fig. 1 Final Project Schema

From the Schema image, you can see the reference used by each table. Below you can see the important columns of tables with their constraints mentioned with the “unique” keyword. Passwords stored in the user table are hashed with SHA256. User is connected with Deck and ReviewResponse tables with one to many relationships. Similarly, Deck & ReviewResponse are connected with Card & ReviewCard tables respectively with one to many relationships. As one user can create multiple Decks and a deck can have multiple cards. Similarly, with reviews, users can have multiple. Let’s see the important columns now.

Table and their Important Columns:-

- **User**:- email(unique), password(hash), role(enum)
  - Password, email for login,
  - “Role” for showing cards of the same role
- **Deck**:- name (unique), public_status (boolean), created_by_id (user_id)
  - Show to all if public_status true
  - User_id for CRUD options of deck & card
- **Card**:- front, back, options
  - Front is for question & back for answer
  - Options field is a string that takes options with comma-separated values for MCQ.
- **ReviewResponse:-** avg_time, score
  - Avg_time in seconds is the average time taken by a user to answer all questions in the deck.
  - “Score” is the percentage of correct answers.
- **ReviewCard**:- (card_id, review_response_id) unique, time, difficulty
  - Difficulty is selected by the user at the end of every question.
  - Time to answer the questions in seconds

API Design

For this project, I have created **10 APIs** using Flask-Restful. APIs can be accessed without login and using user_id. 9 APIs contain all the CRUD APIs for Deck and Cards. I have decided not to give API access for checking answers. The last API is for checking any user's score with the help of user_id. Some APIs and their path and function.

URL	Methods	Description
/api/deck/<int:deck_id>	GET, PUT, DELETE	For reading, updating, deleting a Deck.
/api/deck	POST	For adding a new Deck
/api/deck_cards/<int:deck_id>	GET	For getting all cards in Deck
/api/card/<int:card_id>	GET, PUT, DELETE	For reading, updating, deleting a Card
/api/card	POST	For adding a new Card.
/api/get_score/<int:user_id>	GET	For getting the user's average score of all time.

Architecture and Features

```
│   main.py
│   db.sqlite3
│   requirements.txt
├───application
│   ├───models
│   ├───routes
├───static
│   ├───css
│   ├───js
│   └───media
├───templates
│   │   Base.html
│   ├───Dashboard
│   ├───Deck
│   ├───Review
│   └───Security
```

This tree shows the root folder structure. **python main.py** runs the server. **./requirements.txt** has all the packages which the project needs and ./**db.sqlite3** is the database. **./application** folder has **./application/models** and **./application/routes**. It contains blueprints of the routes and later they are used in app.py. **./static** folder has **./static/css** folder which has files for styling, **./static/js** folder has files for javascript and **./static/media** folder contains files like images and fonts. **./templates** folder has Base.html and other folders which have respective HTML files.

Features

- Adding cards by importing Excel Sheet
- Downloading Excel Sheet of Cards
- Login and signup using Email and Password (Secure)
- Checking if the email domain is valid or not
- Updating Password, User name, Role
- Adding, Updating, Removing Deck
- Adding, Updating, Removing Card
- Filter Decks by pubic status, tag
- Showing last reviewed and all reviewed decks in Dashboard with score and average time
- Supports many languages for cards and 3 choices for difficulty level.
- CRUD APIs for Decks, Cards
- Proper Validation and errors showing

Video

This video shows how the app runs on Replit Portal.
https://drive.google.com/file/d/1PrjQ0cO4sQBIVsFg3Ja9eextJU4tXmPn/view?usp=sharing