FlashCardApp V2

Problem definition

Modern Application Development - 2

Last Updated at : 2021-02-15T22:47:00.00+5:30

Frameworks to be used

- Flask for API
- VueJS for UI
- VueJS Advanced with CLI (only if required, not necessary)
- Jinja2 templates if required
  - Not to be used for UI
- Bootstrap etc., if required
- SQLite for database
- Redis for caching
- Redis and Celery for batch jobs
- It should be possible to run all the demos on the student's computer, which should either be a Linux based system or should be able to simulate the same. You can use WSL for Windows OS.

Flashcards

- Used for memory training
- User can have multiple decks
- System automatically presents one card at a time, and user needs to select a choice based on how well they know the answer
- System will track progress over time and decide which cards need to be reviewed

Terminology

- Front - the part of the flashcard initially shown for review
- Back - the answer or meaning of what is on the front side
- Deck - a collection of related cards (e.g. Tamil, Hindi, HTML etc.)

Core Functionality

- This will be graded
- Base requirements:
  - UI with Vue and Vue Components
    - User login
    - Dashboard
    - Review
    - Deck management
  - Backend Jobs
    - Export Jobs
    - Reporting Jobs
    - Alert Jobs
  - UI and Backend Performance

Core - User Login

- Form for username and password
- Use Flask Security and Token Based Authentication
- Suitable model for user

Core - Dashboard

- Dashboard with list of decks, last reviewed, score (how well reviewed)
- Time of last review, score on deck (some kind of average of individual reviews etc.)
- Scoring method is left up to you as long as you can explain what is done

Core - Review

- Select a deck, then start presenting options one by one;
    - allow user to select from some options like “easy”, “medium”, “difficult” which tells how difficult they found each card
- Update last reviewed time and score, and overall deck score

Core - Deck management

- Create a new deck
    - Add cards to deck - the deck storage should handle multiple languages - usually UTF-8 encoding is sufficient for this
- Edit a deck
- Remove a deck
- Export option is required

Core - Daily Reminder Jobs

- Scheduled Job - Daily reminders on Google Chat using webhook or SMS or Email
    - In the evening, every day (you can choose time of your choice)
    - Check if the user has revised or not on that day
    - If not revised, then send the alert asking to revise.

Core - Scheduled Job - Monthly Progress Report

- Scheduled Job - Monthly Progress Report
    - Come Up with a monthly progress report in HTML (email)
    - On the first day of the month
        - Start a job
        - Create a report
        - Send it as email

Core - User Triggered Async Job - Export as CSV

- User Triggered Async Job - Export as CSV
  - Come up with an export CSV format for Decks and cards
  - Have a dashboard where the user can export
  - Trigger a batch job, send an alert once done

Core - Performance and Caching

- Add caching where required to increase the performance
- Add cache expiry
- UI and API Performance

Recommended (graded)

- Backend Jobs
  - Import Jobs
- Well designed PDF reports (User can choose between HTML and PDF reports)
- Single Responsive UI for both Mobile and Desktop
  - Unified UI that works across devices
  - Add to desktop feature

Optional

- Styling and Aesthetics

Evaluation

- Report (not more than 2 pages) describing models and overall system design
  - Include as PDF inside submission folder
- All code to be submitted on portal
- A brief (2-3 minute) video explaining how you approached the problem, what you have implemented, and any extra features
  - This will be viewed during or before the viva, so should be a clear explanation of your work
- Viva: after the video explanation, you are required to give a demo of your work, and answer any questions
  - This includes making changes as requested and running the code for a live demo
  - Other questions that may be unrelated to the project itself but are relevant for the course

Instructions

- This is a live document and will be updated with more details and FAQs, as we proceed.
- We will freeze the problem statement on or before 18th February, beyond which any modifications to the statement will be communicated via proper announcements.
- The project has to be submitted as a single zip file.
- The last date for submission is 18th March. This is a hard deadline and NO extensions will be possible.