Author

Bhavya Dua
21F1003641
21f1003641@student.onlinedegree.iitm.ac.in
I am a third year B.Tech student, with a passion for robotics and public speaking

Description

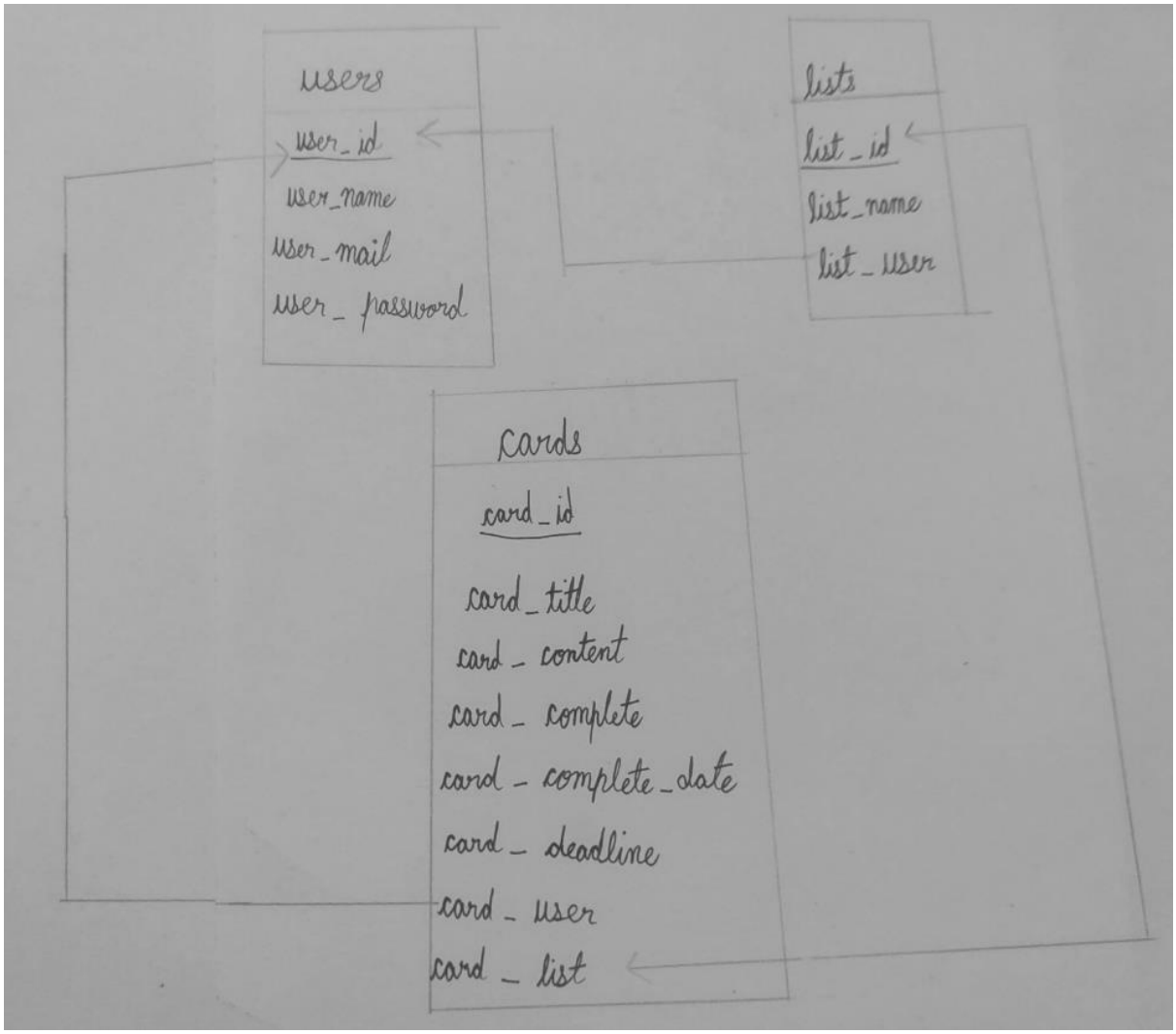
A KanBan application which can be used as a 'ToDo' list for keeping track of various lists with each list having various cards, which is updated daily

Technologies used

Flask, Flask-SQLAlchemy, SQLite, matplotlib, pandas, os, datetime, HTML, CSS, Bootstrap, Vue.JS, Celery, Redis, smtplib, flask-jwt-extended, brcypt, flask_cors, weasyprint

Flask Flask-SQLAlchemy: Basic routing and database related operations
Flask_cors: CORS management for the application
Flask-JWT-Extended: Create and manage web tokens
matplotlib: Visualization of trendlines (graphs)
OS library: secret key, root address for saving files
HTML, CSS, Bootstrap: Designing webpages
Datetime library: Retrieve date and time, for timestamps and setting timer expiry
Vue.JS: Create frontend and interface to connect with Flask API
Celery, Redis, smtplib: Asynchronous batch jobs
Pandas: Export data as CSV files
weasyprint: Convert HTML reports to PDF format

DB Schema Design



Tables:

1. Users
    a. user_id (primary key): Integer, identifier for users
    b. user_name: String, name of the user
    c. user_mail: String, email ID of the user
    d. user_password: String, password of the user's account which is encrypted and stored
2. Lists
    a. list_id(primary key): Integer, identifier for lists
    b. list_name: String, name of the list
    c. list_user(foreign key(users.user_id)): Integer, identifies the owner of that list
3. Cards
    a. card_id(primary key): Integer, identifier for cards
    b. card_title: String, title of the card
    c. card_content: String, description/contents of the task/card
    d. card_complete:Boolean, Status of completion of the task
    e. card_complete_date: Datetime, date on which the card was completed

f. card_deadline: Datetime, deadline by which the task has to be finished
g. card_user(foreign key(users.user_id)): Integer, identifies the owner of that task
h. card_list(foreign key(lists.list_id)): Integer, identifies the list to which the card belongs

This methodology of designing allows for optimal streamlining (and efficiency) of the database structure

API Design

APIs have been created and utilized for signing up and logging in to the user's account. They are also used for performing CRUD operations on lists, as well as cards. APIs are finally used for export jobs.

Architecture and Features

The backend folder consists of the business logic – Flask routes, cache description, asynchronous jobs, and such. It also contains the report format and the project database. The frontend folder consists of all the different views and JS/.VUE files for running the application.

There is a sign-up/sign-in form for users, after which they are redirected to the Board view for them, from where they can manage their lists and cards, including options to update, delete, and export lists/cards, get summaries of lists, and an option to convert the web application to a desktop application (Progressive Web Application). There is functionality for daily reminders via Google Space and monthly reports sent in PDF reports via email

Video

https://drive.google.com/file/d/1T0vCUpksojg_RIQhuOHTd7f8BWDZvQYv/view?usp=sharing