Author

Name: Vivek Sivaramakrishnan
Roll: 21f2000045
E-mail: 21f2000045@student.onlinedegree.iitm.ac.in
About: A final year BSc.Mathematics student excited to contribute academically to the growing data science academia.

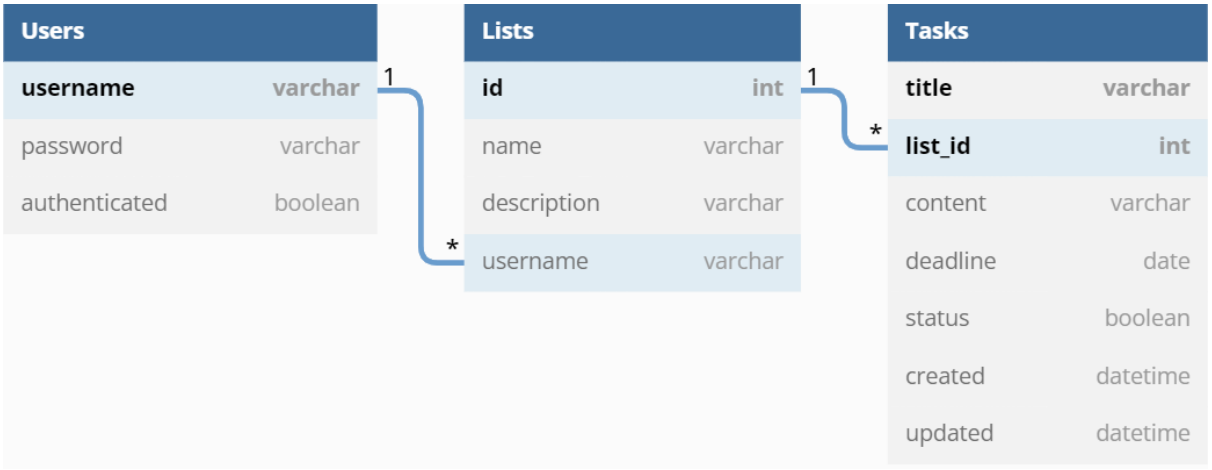
Description

A simple timetabling web application based on the 'Kanban' ideology which entails partitioning a schedule into lists, with each list populated with appropriate tasks.

Technologies used

- HTML, CSS, Javascript, Bootstrap, jinja2 - Client-side presentation tools
- Python, Flask - Back-end processing and web routing
- Flask SQLAlchemy - Database management module
- Flask-Login, Hashlib - Frameworks to facilitate login processes
- Flask-WTF, WTForms - Form declaration and input handling
- Flask-RESTful - API declaration, routing and handling
- Pandas, Matplotlib, io.BytesIO - To generate excel spreadsheet of data on the fly for further analysis

DB Schema Design



A User can have multiple lists, which can have multiple tasks. Therefore, a schema as shown above has been idealised and implemented.

API Design

Implemented using **Flask-RESTful**. A total of 4 resource classes were used. The following table describes the endpoints:

Endpoint	Description
/api/login	Login user using **post** request
/api/list	**CRUD** operations on lists
/api/task/{list_id}	**CRUD** operations on tasks
/api/stats	**get** descriptive stats on tasks of current user's lists

Architecture and Features

```
|---static
|   |---ico / App icons
|   |---wallpapers / Wallpapers used in forms
|---templates / Jinja 2 Templates
|   |---errors
|   |---forms
|   |---layouts
|   |---pages
```

Python scripts:

- **Main** scripts:
  - **app.py** - App, Database initialization and app instantiation
  - **config.py** - Definitions of the Flask application's global variables
- **Website** components:
  - **api.py**, **api_token_auth.py** - API Authentication and Resource definitions
  - **controllers.py** - Controllers and error handlers
  - **forms.py** - Form Class definitions and input validators
- **models.py** - SQLite database structure definitions
- **password.py** - MD5 Hashing solution for login framework
  - Checks user input against hash generated by md5(concatenate(user, password)).

Additional Features:

- **Popper.js** used to display useful tooltips on mouse hover.
- **jQuery.js** scripts dynamically generated using **jinja2** to do the following:
  - Makes a "delete-modal" visible on click to confirm deletion
  - In case of validation error on user input, HTML elements are edited to show error message originating from flask application.
  - Background color change triggered on a button click
- Generates .xlsx spreadsheet about a user's lists / a list's tasks on the fly for analysis in MS Excel/Google Sheets (look out for green buttons).
- Token-based user-authentication for API to enable CRUD operations on application components by a developer.

Video

mad1_final.mp4