### General Instructions - DELETE BEFORE CREATING FINAL REPORT

* The report should be not more than 2 pages in length - only brief descriptions are expected. Use standard A4 page size and 10pt font.
* This report, as well as the video are **mandatory** submissions. Your final project will not receive a grade unless you have submitted both.
* The video accompanying the project report should be uploaded under your onlinedegree Google account, and shared with the instructors. Do NOT upload the video as part of your submission - only the link to the video should be uploaded.
* If you use this file as the template - remove all <<tagged>> entries. These are present only as guidelines, and SHOULD NOT BE in the final submission.
* Final submission should have one ZIP file containing the following:
  + This report in PDF form
  + Your code folder with all necessary files needed to run and test. The code should also have a README file that explains how to run the code.
  + YAML file used for API definition

### Author

Ranjeet Sharma

21f2001119

21f2001119@ds.study.iitm.ac.in

Student at IIT-Madras pursuing Bachelor's degree in Data Science and Application, with a fervent interest in development, and programming.

### Problem Statement - **Library Management System** - V2

### Description

This project is about an Online Library Management System. There will be one admin and many users. Users can sign up/register and can start reading books or issue e-books. Admin can perform CRUD Operations on Sections and Books and handle incoming book requests.

### Technologies used

* **Flask**: For application code, to handle user requests, manage routing, and creating APIs.
* **Vue.js**: For building a dynamic and responsive user interface.
* **Flask-SQLAlchemy**: For interaction with the database.
* **Flask-Bcrypt**: For hashing passwords.
* **Bootstrap**: For quick CSS styling and aesthetics.
* **SQLite**: For data storage.
* **Redis**: For caching.
* **Redis and Celery**: For batch jobs and task queue management.

### Architecture and Features

**app.py**: Contains the main code to run the Flask application. It initializes Flask and Flask-SQLAlchemy objects and contains necessary imports from controllers.

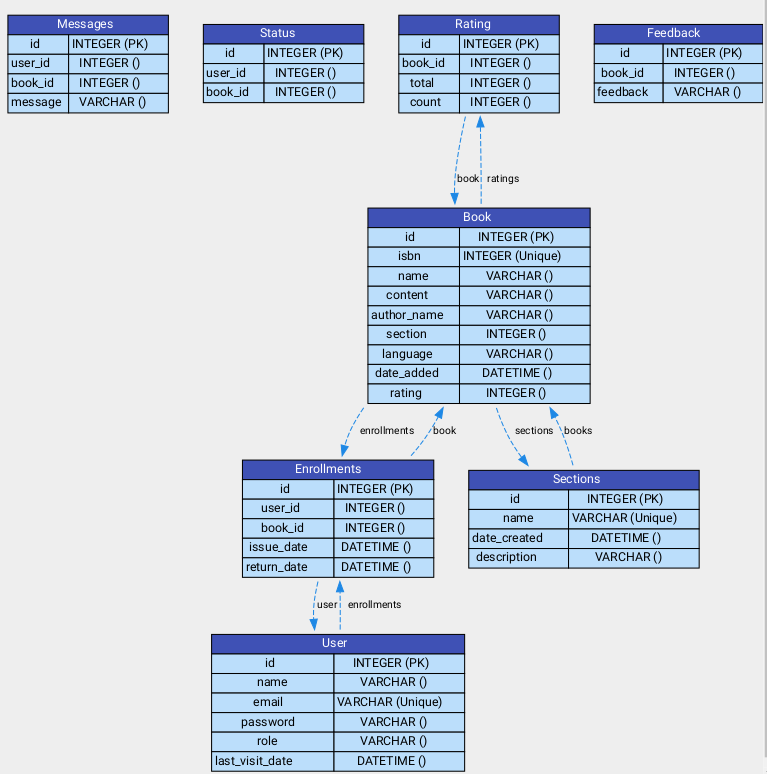
**Controllers**: Contains all the routing for the project.

**models.py**: Contains the schema for database design using Flask-SQLAlchemy. It includes classes that represent tables in the database, including columns and relationships between tables.

**Static and Templates**: The static folder contains global.css along with a few images, and all the HTML files are kept in the templates folder.

* **CRUD** Operations for **Sections** and **Books**.
* **Search Functionality**
* Both admin and regular users can search based on book name, author, sections.
* **Bar Charts** also visible on the admin's dashboard for better track of books and sections.
* **Batch Jobs and Task Management**
* **Redis and Celery**: Used for managing background tasks and scheduling periodic jobs like sending reminders and generating reports.

### DB Schema Design



### API Design

* **Book** **Management** **API**, **Section Management API, Enrollments Management API, Profile Management API, Search, User Authentication/Management API** and **Admin** relevant endpoints with **GET**, **POST**, **DELETE**, **PUT** methods.
* **Book** **Management** **API:** GET /api/book/<book\_name>, POST /api/book, DELETE /api/book/<book\_name>, PUT /api/book/<int:book\_id>
* **Section** **Management** **API: GET /api/admin/sec, POST /api/admin/sec, PUT /api/admin/sec/<section\_name>, DELETE /api/admin/sec/<int:section\_id>**

### Video

<https://drive.google.com/file/d/1PkFru_y1Jb-TIRfNKdrwNj2mbwkmsMNH/view?usp=sharing>