Library Management System with Recommendation Engine

Submitted in the partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

In

Computer Science And Engineering

Specialized In

Artificial Intelligence And Machine Learning

Submitted by:
Piyush Soni (19BCS6104)

Under the Supervision of:
Pramod Vishwakarma

Outline

- Project Introduction
- Problem Formulation
- Project Objectives
- Methodology
- Outputs Of Result
- Conclusion
- Future Scope
- References

Project Introduction

An Interactive Library Management System With Real-Time Book Recommendation

ATHENEUM

Project Features

- **Database Management Solution For Library**

  Atheneum provides a intuitive UI to manage Books, Activities, staff members and members of the library.

- **Individual Book Recommendation**

  Atheneum recommends books to the based on previously collected user data.

- **Scalable Over LAN and Platform Independent**

  Atheneum is moderately scalable too.

Project Features

- ➔ Three Different Access Levels For Users
  - → Admins
    - Add Or Update Users for All Access Levels
    - Add or Update Books
    - Issue Or Return Books
  - → Curators Or Moderators
    - Issue Or Return Books
    - View Currently Issued Books
    - Issuance History
  - → Members Or Users
    - View Books
    - Receive Recommendation Based On Their Data

Technological Architecture

Frontend ⇄ Backend ⇄ Database

Redux JS

+

React JS

django

PostgreSQL

Or

SQLite

Problem Formulation

- ➔ Management System

  Creation of Database Solution For Library With Different Access Levels

- ➔ Book Recommendations

  To recommend books, ratings for each book in the available data has been predicted and recommendations

Objectives Of Management System

- **The Efficient Management of Huge Libraries** with a lot of books along with their information and overwhelming number of users

- **Optimized and Intuitive User Interface** for admins and curators to make their life less stressful

Objectives Of Recommendation Engine

- **Quality Recommendation** for user from plethora of books available.

- Helping user to find **The Next Best Read For Them** which would increase their happiness index and quality of life.

Objectives Of Web Application

- **Scalable Application** so that it can be accessed over a LAN or organization's network

- **Easy To Deploy Application** for the user-experience of network administrator of the organization

- **Platform Independent Application** as there are only two prerequisites,i.e. a web browser and user's presence on the same network

Aim Of Atheneum

Improved Quality Of Life

For All Of Its Users

Be It Readers, Curator Or Admin

Methodology

Different methodologies have adopted for development each major component of the application. Those components are :

- ➔ Frontend
- ➔ Backend
- ➔ Recommendation Engine
- ➔ Full-stack Integration

All major components have been developed concurrently over the course of semester.

Methodology for Frontend

- ➔ Creation of Mockups
- ➔ Install prerequisite dependencies
- ➔ Setting up React with npx
- ➔ Setting up Redux Dev tools for debugging
- ➔ Set up Redux store for State management
- ➔ Creation of Components
- ➔ Implementation of Layout

Methodology for Backend

- ➔ Creation of a plan of action to proceed
- ➔ Set up directories
- ➔ Initiate Django Project
- ➔ Initiate Django apps for each API
- ➔ Implement three APIs (mainapp, users, library)
- ➔ Load and Integrate ML model
- ➔ Test Each API for required output

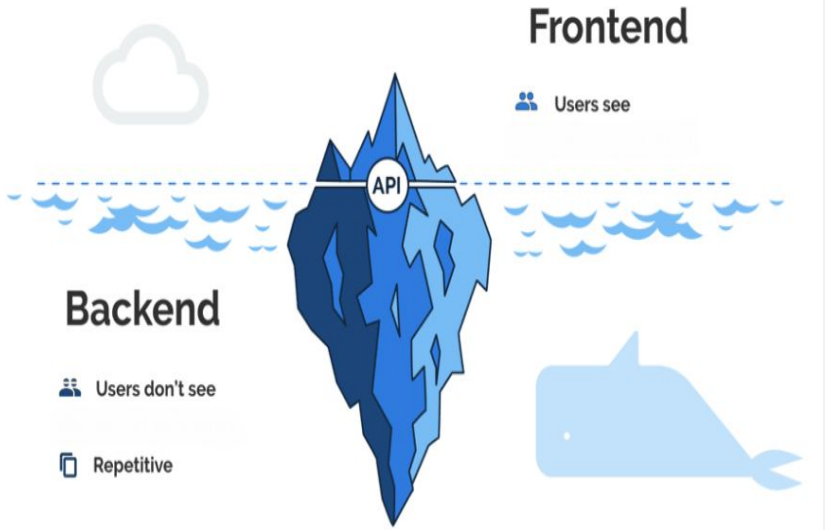
Methodology for Recommendation Engine And Full-stack Integration

- ➔ Recommendation Engine
  - ○ Acquiring knowledge about existing systems
  - ○ Dataset Retrieval and Synthetic feature generation
  - ○ Data Preparation & Model Creation
  - ○ Saving Model in joblib file

- ➔ Full-stack Integration
  - ○ Apply middlewares to Django project
  - ○ Set up routes and URLs in django and react project
  - ○ Creation of Reducers in react app
  - ○ Setting up hooks for API calls in react app

FrontEnd :-

- For FrontEnd we use **react, nodejs.**
- **Redux dev tool** is used for monitoring.
- For store and retrieve model **Joblib** is used.



BackEnd :-

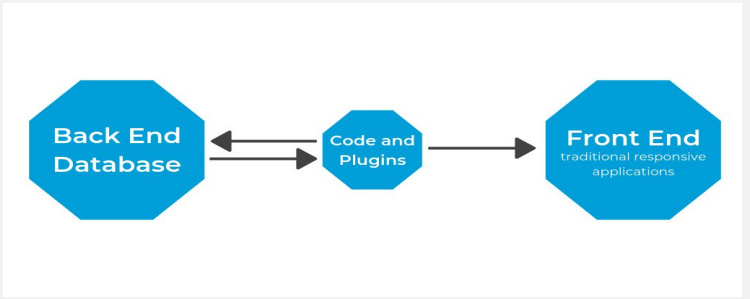
- For BackEnd we use **Python, django.**
- **PostgresSQL/SQL Database** used as a database.

Recommendation Engine :-

- For Recommendation Engine we use **Python**.
- Library used - **numpy**, **pandas**, **sklearn** and **Surprise.**

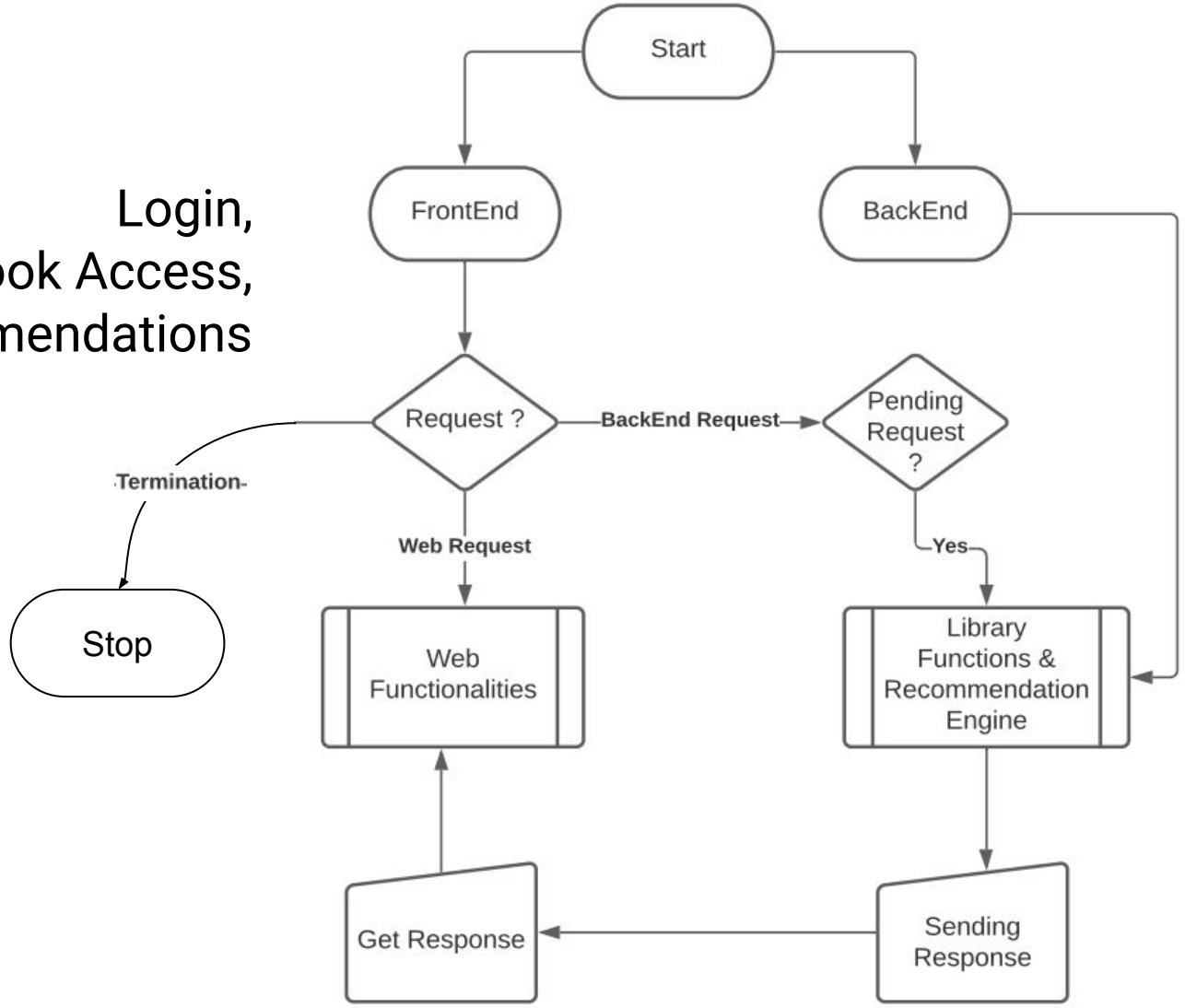
To connect Backend and Frontend :-

- **CORS middleware.**



Flow Of Application

Login,
Book Access,
Get Recommendations



Dataset And Recommendation Engine

➔ Dataset Retrieval And Preparation

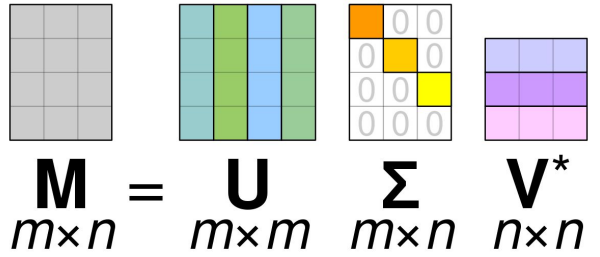
Current implementation uses a extended book crossing dataset with kaggle's book summary dataset with synthetic features.

➔ Recommendation Engine Algorithm

The Engine employes matrix factorization to predict ratings for individual users based on user data.

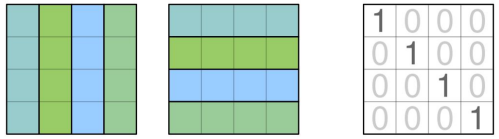
Recommendation Engine Algorithm

- **Matrix Factorization** is a class of collaborative filtering algorithms. It decomposes the user-item interaction matrix into the product of two lower dimensionality matrices.
- Current implementation of user-item interaction matrix is *user x books* matrix.



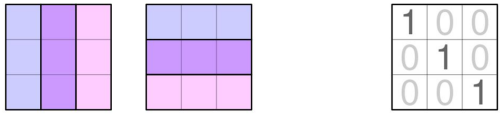
$$\underset{m\times n}{\mathbf{M}} = \underset{m\times m}{\mathbf{U}}\ \underset{m\times n}{\mathbf{\Sigma}}\ \underset{n\times n}{\mathbf{V}^*}$$

M *is* *m x n* *user-item interaction matrix that decomposes into* ***U*** *and* ***V***



$$\mathbf{U}\ \mathbf{U}^* = \mathbf{I}_m$$

U *is* *m x m* *orthonormal matrix*



$$\mathbf{V}\ \mathbf{V}^* = \mathbf{I}_n$$

V *is* *n x n* *orthonormal matrix*

Outputs Of Result

- ➔ Login Page
- ➔ User Home Page
- ➔ Recommendations Page
- ➔ Update Password Page
- ➔ Admin Home Page
- ➔ Issue Book Dialog
- ➔ Return Book Dialog
- ➔ Features Of Admin Panel Page
- ➔ List Of Database Tables
- ➔ Accessing App Over LAN On Mobile

Login Page

ATHENEUM

Sign in

UserName *

user

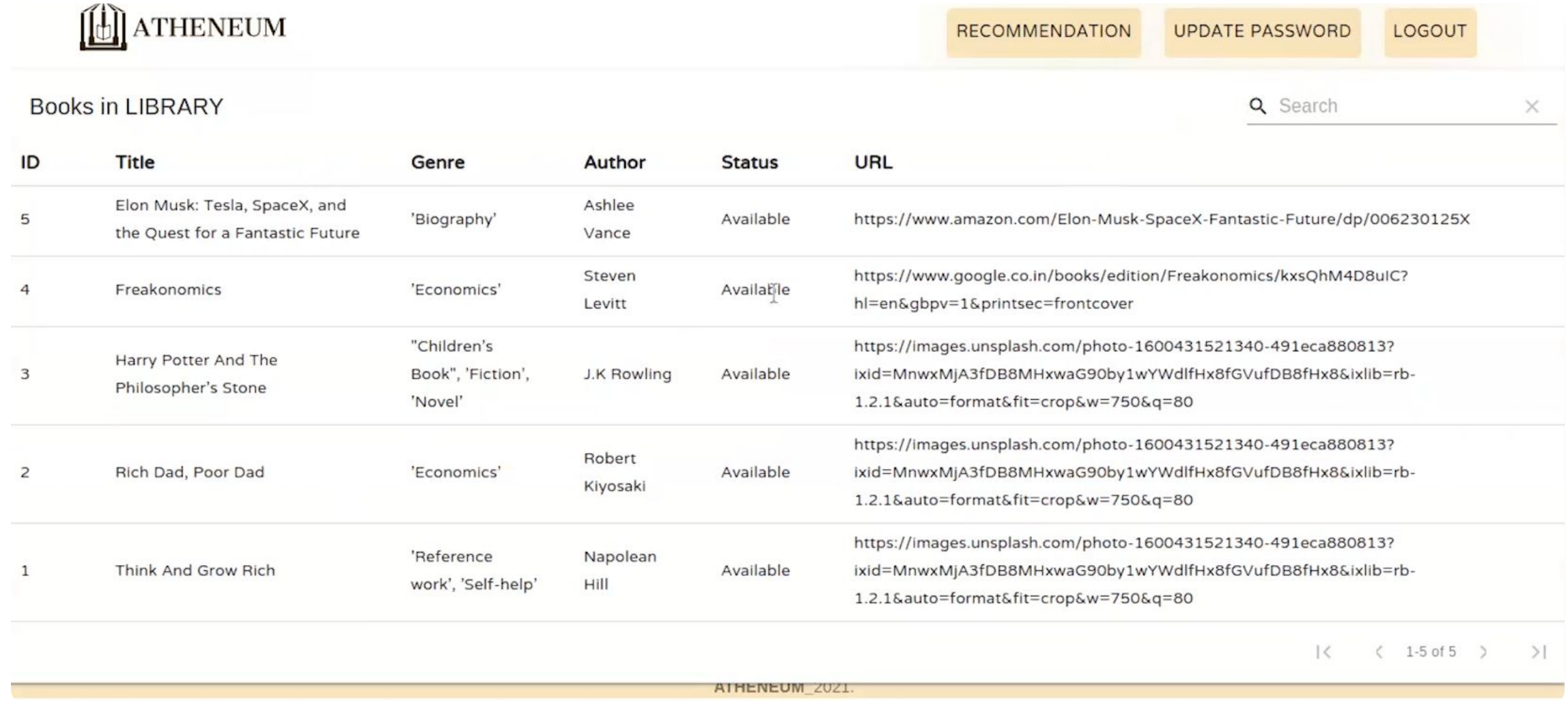
Password *

••••••••

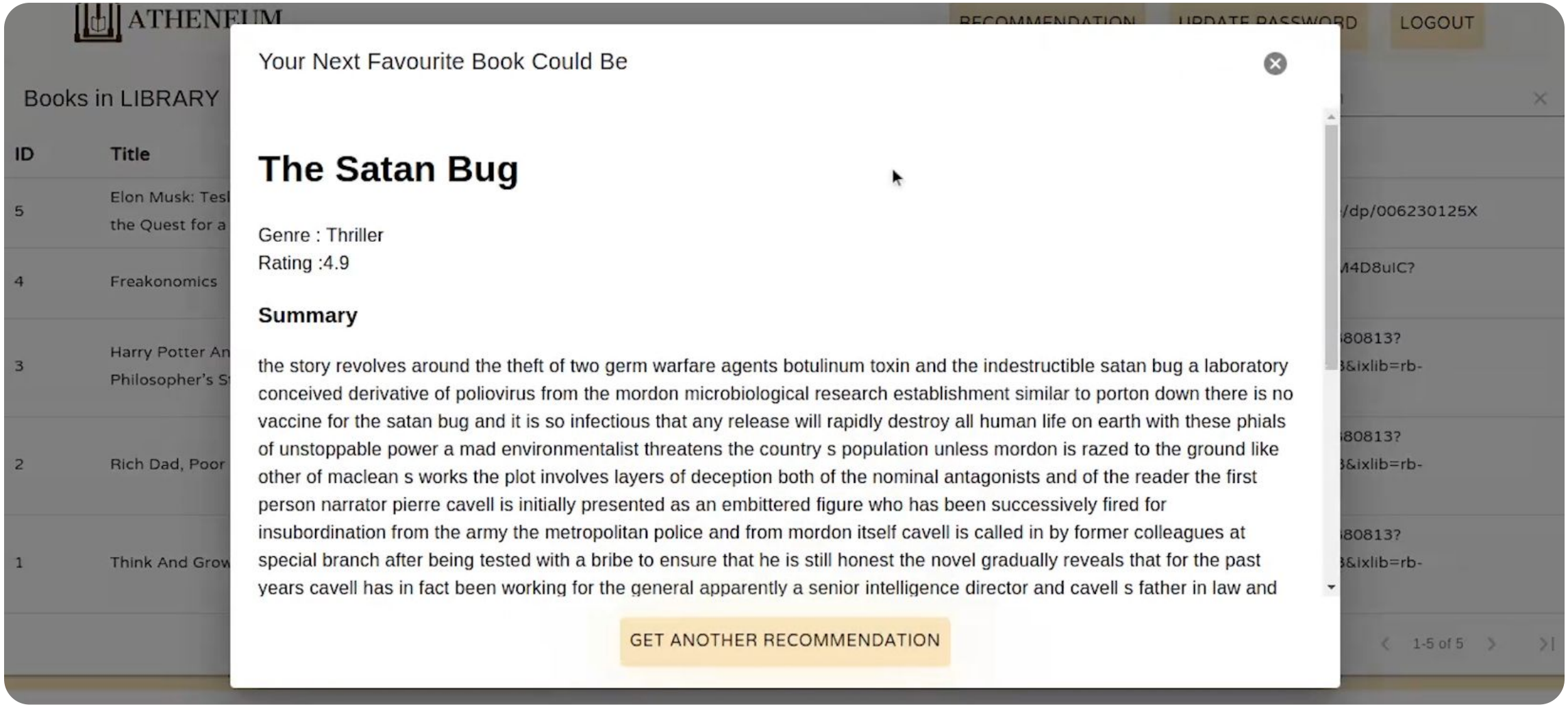
SIGN IN

ATHENEUM_2021.

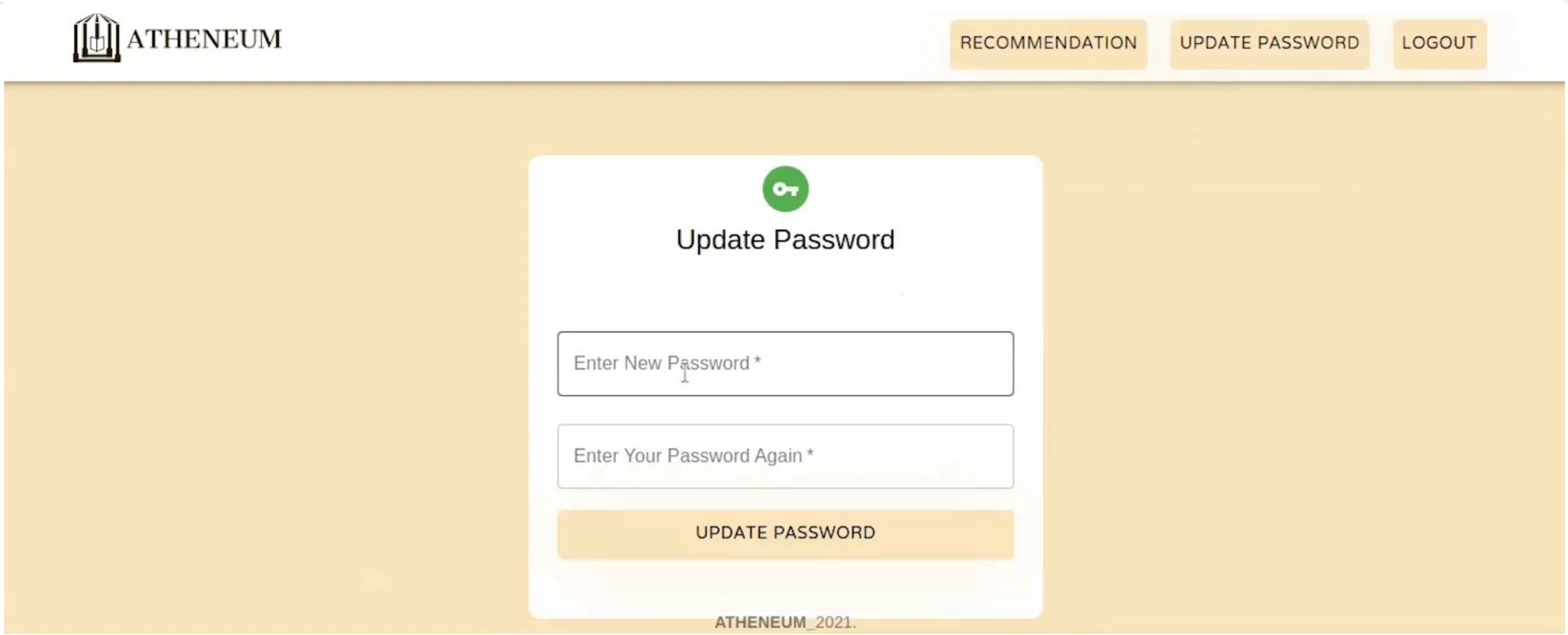
User Home Page



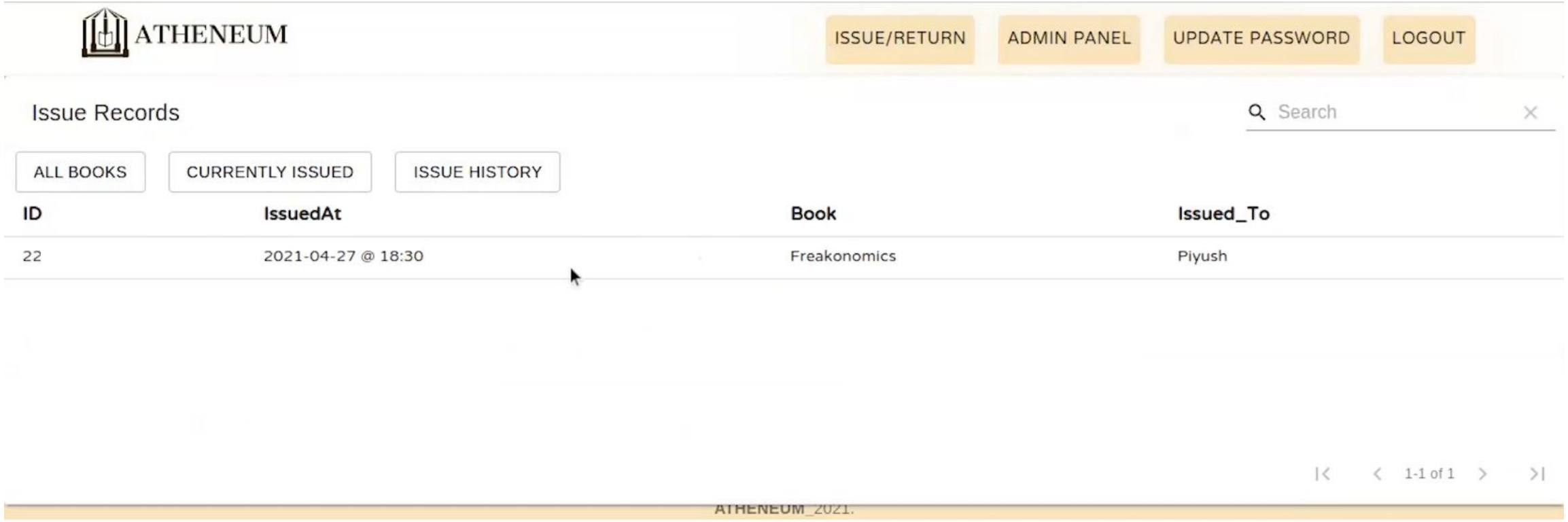
Recommendation Page



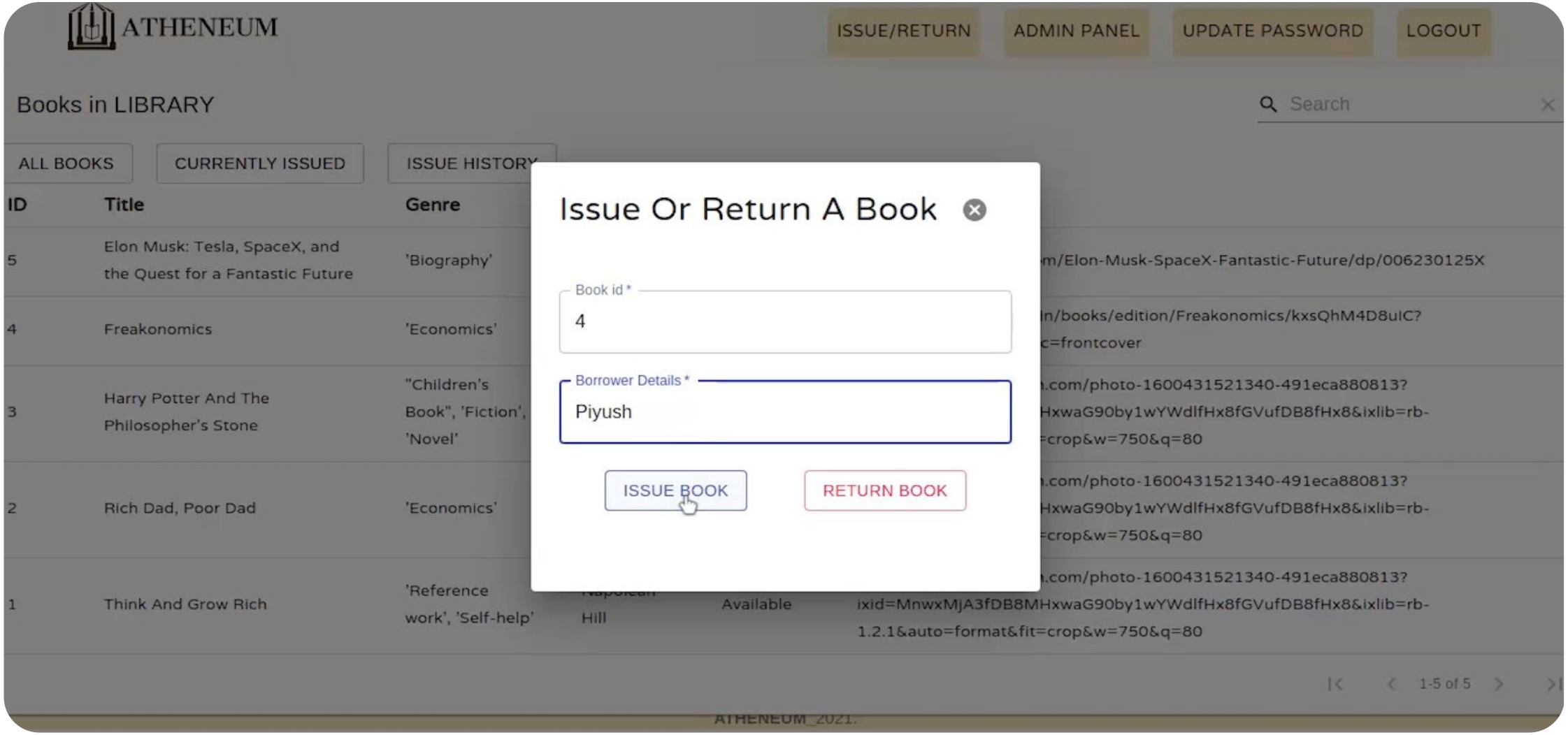
Update Password Page



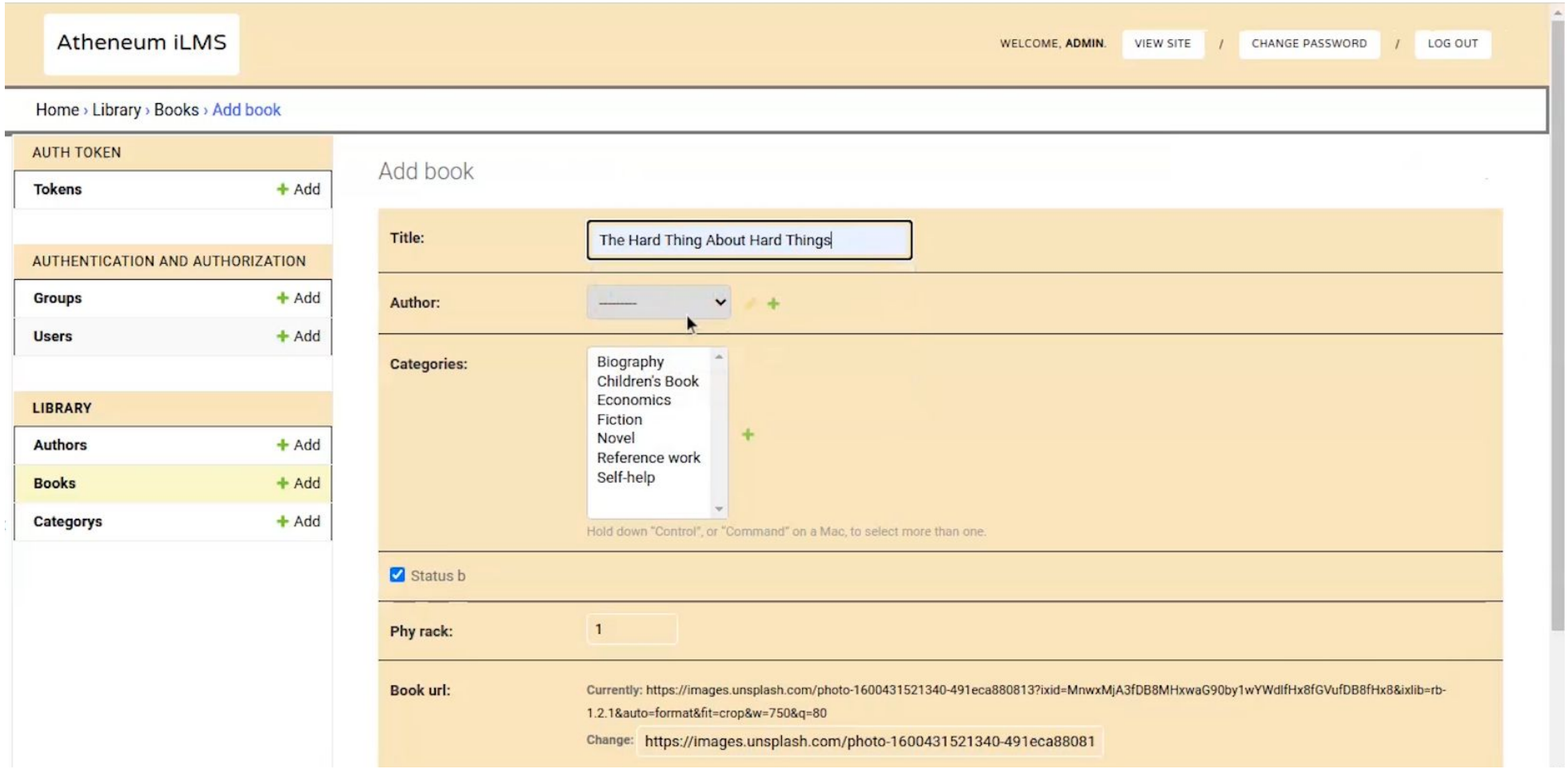
Admin Home Page

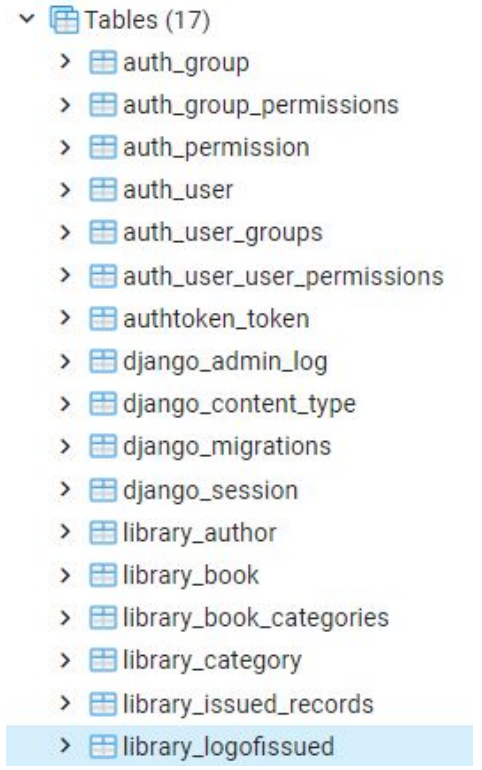


Issue/Return Book Dialog



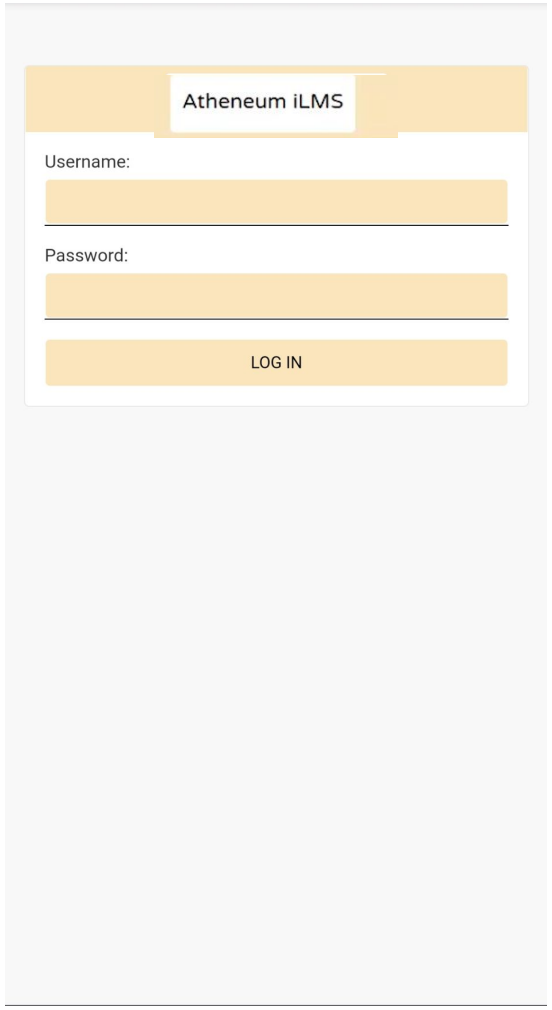
Features Of Admin Panel & List Of Database Tables

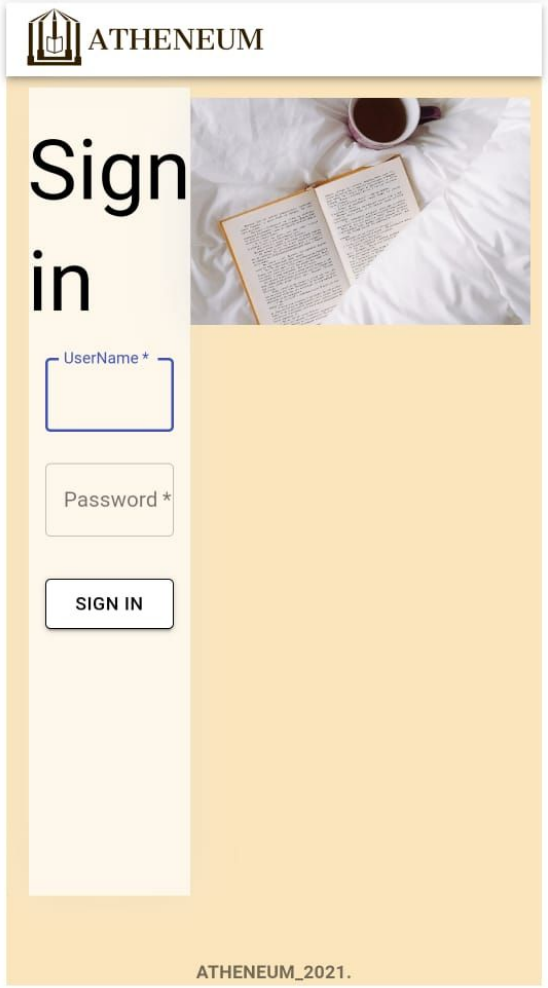




Accessing App Over LAN On Mobile

The Application isn't responsive yet





Conclusion

- The App can be used to manage a Library Efficiently and recommend books as well.

- One Drawback of this App is it is not yet optimized for other Devices. App looks better in PCs and not so much in smartphones.

Future Scope

- The App can be used to make a perfect Library if the below mentioned points are achieved.
  - The App can be optimized for Other Devices.
  - Recommendation engine to be made dynamic by defining regular updation (training) periods
  - The Application can be made scalable over the internet

References

- ➔ React documentation : https://reactjs.org/docs/getting-started.html
- ➔ Django documentation : https://docs.djangoproject.com/
- ➔ Pipenv documentation : https://pipenv-fork.readthedocs.io/en/latest/
- ➔ VirtualEnv documentation: https://virtualenv.pypa.io/en/latest/
- ➔ Django React Tutorial : Tutorial
- ➔ Joblib : Blog

Research Papers

- Vaz, P.C., de Matos, D.M., & Martins, B. (2012a). Stylometric relevance-feedback towards a hybrid book recommendation algorithm. In Proceedings of the fifth ACM workshop on research advances in large digital book repositories and complementary media, BooksOnline '12 (pp. 13–16). New York: ACM.
- Pera, M.S., & Ng, Y.-K. (2014a). Automating readers' advisory to make book recommendations for K-12 readers. In Proceedings of the 8th ACM conference on recommender systems, RecSys '14 (pp. 9–16). New York: ACM.
- Pera, M.S., & Ng, Y.K. (2014b). How can we help our K-12 teachers?: using a recommender to make personalized book suggestions.
- Pera, M.S., & Ng, Y.-K. (2015). Analyzing book-related features to recommend books for emergent readers.
- Vaz, P.C., Ribeiro, R., & Martins de Matos, D. (2013). Understanding temporal dynamics of ratings in the book recommendation scenario.
- Xavier Amatriain and Justin Basilico, C. 11 Recommender Systems in Industry: A Netflix Case Study, Recommender Systems Handbook.
- Manning, C.D., Raghavan, P., & Schutze, H. (2008). ¨ Introduction to information retrieval. New York: Cambridge University Press.
- Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. Computer, (8):30–37.

Thank You