User’s Book Wishlist Design

# Requirement:

Design and implement RESTful APIs for user’s book Wishlist where he can manage add, update or delete books from Wishlist.

# Technology Choices:

Programming Language: Python3

Web framework: Flask

Database: sqllite

# Database Schema Design:

Create a User table with fields (firstName, lastName, Email, Password) and a Book table with fields (Author, Title, ISBN, DataOfPublication). Separate table to store Wishlist information with references of User ID from Users table and Book ID from Books table. Use Wishlist table to manage wishlist.

Wishlist table:

UNIQUE(UserId, BookId)

FOREIGN KEY (UserId) references Users(Id)

FOREIGN KEY (BookId) references Books(Id)

A screenshot of a cell phone

Description automatically generated

# RESTful APIs

**Header:**

**Content-type:** application/json

**BaseURL: http://localhost:5000**

1. **Goal: Add Wishlist**

This API will be used to add a book to user wishlist

**Endpoint: POST – {BaseURL}/users/wishlist**

**Request Content:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| UserId | Integer | Identifier of a user |
| BookId | **I**nteger | Identifier of a book |

**Response Body:**

* **If everything ok**

**Message:** Wishlist added successfully

**Status:** 200

* **If User or Book doesn’t exist**

**Message:** User/Book doesn’t exist with UserId: {id}, BookId: {id}

**Status:** 400

* **If Wishlist already exists**

**Message:** Wishlist already exists

**Status:** 409

1. **Goal: Get Wishlist**

This API will be used to get existing user’s wishlist using user id.

**Endpoint: GET - {BaseURL}/users/{id}/wishlist**

**Request Content:**

Empty

**Response Body:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| FirstName | String | User’s FirstName. |
| LastName | String | User’s LastName |
| Email | String | User’s Email |
| Wishlist | Array of Object | Array of Book object |

**Book Object:**

|  |  |
| --- | --- |
| **Name** | **Type** |
| Author | String |
| Title | String |
| ISBN | String |
| DateOfProduction | Date |

1. **Goal: Remove Wishlist**

This API will be used to remove wishlist from a user.

**Endpoint: DELETE - {BaseURL}/users/{id}/book/{id}**

**Response Body:**

**Message:** Wishlist removed successfully

**Status:** 200

1. **Goal: Add User**

This API will be used to create a user.

**Endpoint: POST - {BaseURL}/users**

**Request Content:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| firstName | String | User FirstName. |
| lastName | String | User LastName |
| email | String | User Email |
| password | String | User Password |

**Response Content:**

**Message:** User updated successfully

**id:** User identifier

**Status:** 200

**If already exists:**

**Message:** User already exists.

**Status:** 409

1. **Goal: Get User**

This API will be used to Get a user by ID.

**Endpoint: POST - {BaseURL}/users/{id}**

**Response Content:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| FirstName | String | User FirstName. |
| LastName | String | User LastName |
| Email | String | User Email |
| Id | Integer | User Identifier |

**Status:** 200

1. **Goal: Remove User**

This API will be used to remove a user by ID.

**Endpoint: {BaseURL}/users/{id}**

**Response Body:**

**Message:** User removed successfully

**Status:** 200

1. **Goal: Add Book**

This API will be used to create a book.

**Endpoint: POST - {BaseURL}/books**

**Request Content:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| title | String | Book title |
| author | String | Book author |
| ISBN | String | Book ISBN number |
| dataOfPublication | String | Book date of publication |

**Response Content:**

**Message:** Book added successfully

**id:** Book identifier

**Status:** 200

1. **Goal: Update Book**

This API will be used to update a book.

**Endpoint: PUT - {BaseURL}/books/{id}**

**Request Content:**

Same as add book request content

**Response Content:**

**Message:** Book updated successfully

**Status:** 200

1. **Goal: Get Book**

This API will be used to Get a book by ID.

**Endpoint: POST - {BaseURL}/books/{id}**

**Response Content:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| Author | String | Book title |
| Title | String | Book author |
| ISBN | String | Book ISBN number |
| DateOfProduction | String | Book date of publication |
| Id | Integer | Book identifier |

**Status:** 200

**If not found**

**Status: 404**

1. **Goal: Remove Book**

This API will be used to remove a book by ID.

**Endpoint: {BaseURL}/books/{id}**

**Response Body:**

**Message:** Book removed successfully

**Status:** 200

# Performance Improvement:

Current get user’s wishlist API queries by joining users, books table with wishlist foreign key, here we can improve query by selecting only require fields. And if user wishlist is too big better to limit the results returns by implementing pagination by having limit and offset.

Also, this query using group concat (json\_array) which could impact on query performance when we have memory limited environment. This can be avoided by maintaining wishlist view separately with right indexing.