

Universidad Interamericana de Puerto Rico - Recinto de Arecibo

Programa de Ciencias de Computadoras

Proyecto: Gestor de Biblioteca

Curso: Comp 2052 (Microservicio)

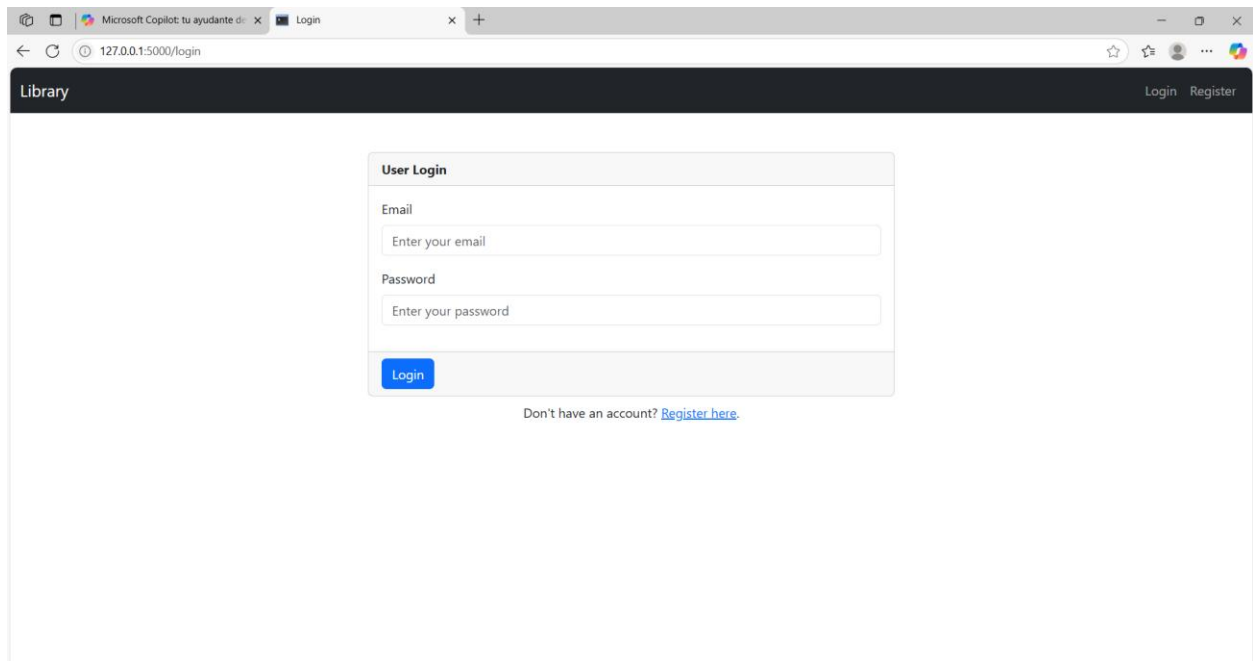
Jeremy A. Ayende Santiago (R00658578)

Eloy Toledo De Jesús (Y00651313)

Título: Interfaces Front-End

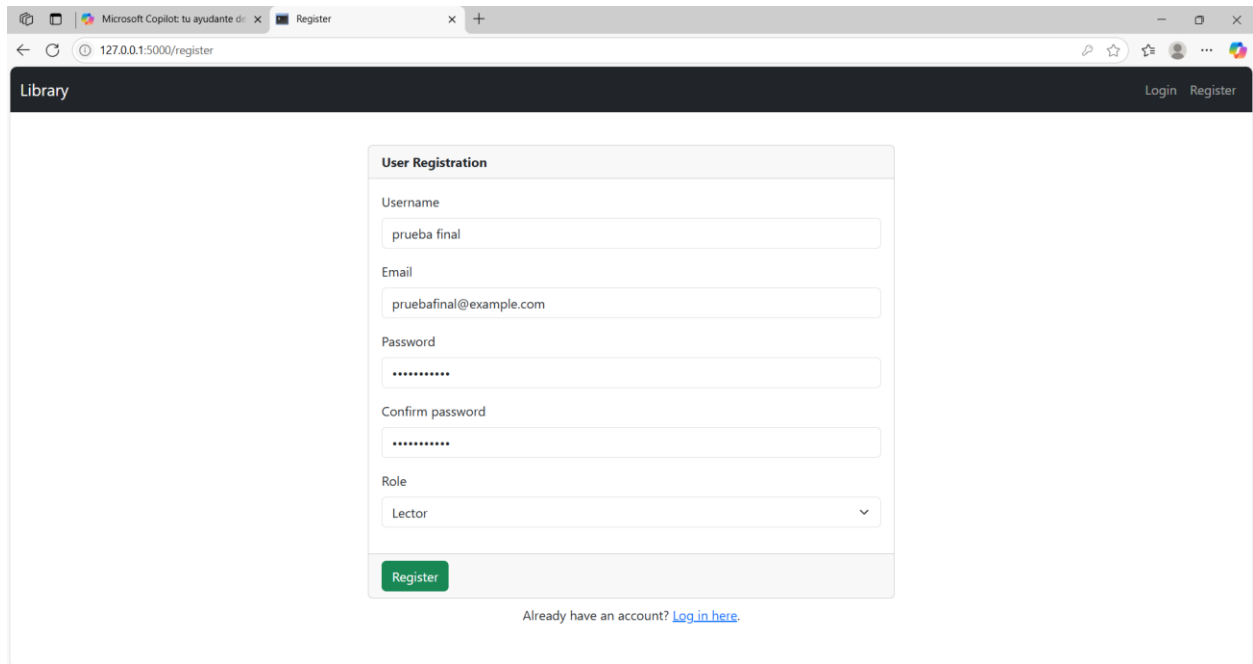
Parte 1: Lector

1. Se presenta la página general de 'Login' que contiene:
 - a. Campos de entrada para correo electrónico y contraseña.
 - b. Botón "Login" para enviar las credenciales.
 - c. Enlace para registrar una cuenta si el usuario aún no tiene una.
 - d. Barra de navegación superior con enlaces a "Login" y "Register."



2. Interfaz de registro de usuarios que contiene:

- a. Campos de entrada para nombre de usuario, correo electrónico, contraseña, confirmar contraseña y elegir el rol del usuario.
- b. Botón de 'Register' para enviar las credenciales a la base de datos y almacenarlos.
- c. Un mensaje en la parte inferior para hacer 'Login' si ya tiene una cuenta registrada.

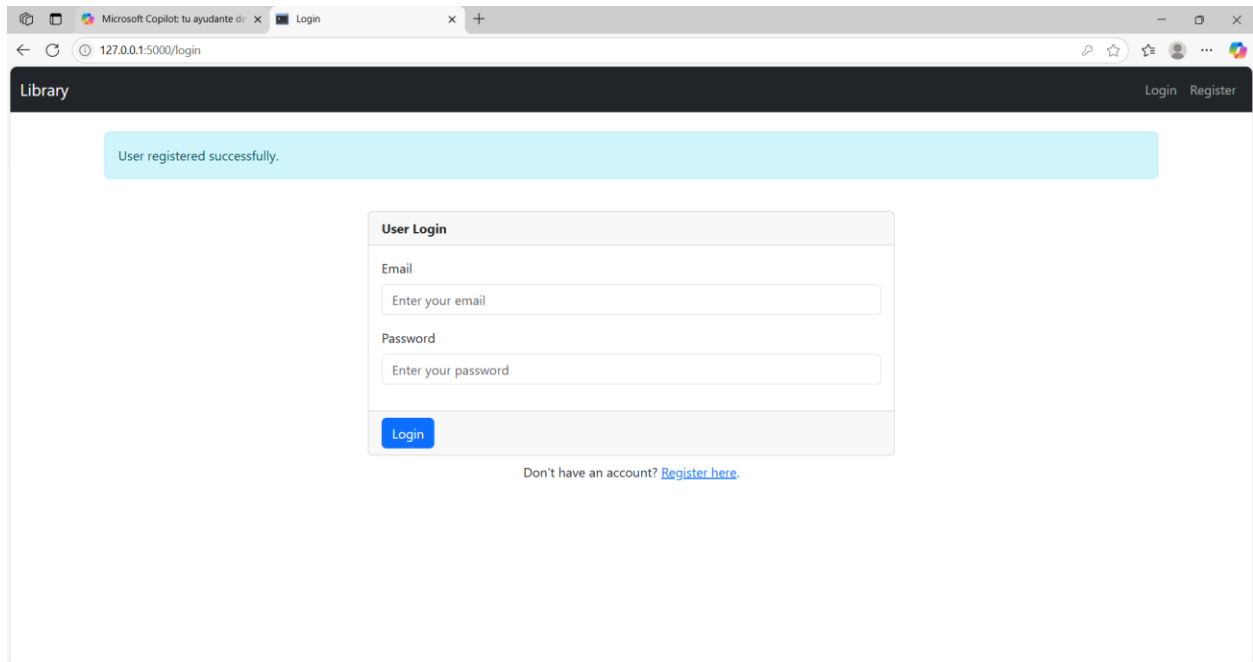


The screenshot shows a web browser window with the address bar displaying "127.0.0.1:5000/register". The page has a dark header with "Library" on the left and "Login Register" on the right. The main content area features a "User Registration" form with the following fields:

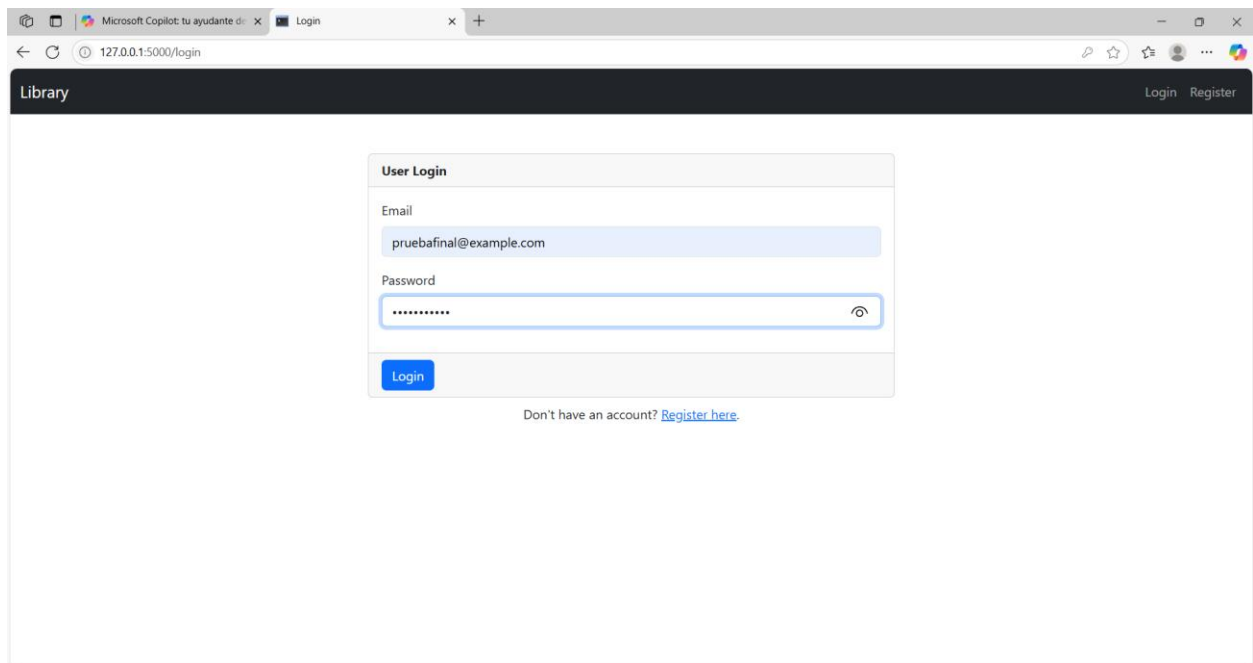
- Username:** Input field containing "prueba final".
- Email:** Input field containing "pruebafinal@example.com".
- Password:** Input field with masked characters "*****".
- Confirm password:** Input field with masked characters "*****".
- Role:** Dropdown menu with "Lector" selected.

Below the form is a green "Register" button. At the bottom of the form, there is a link: "Already have an account? [Log in here](#)."

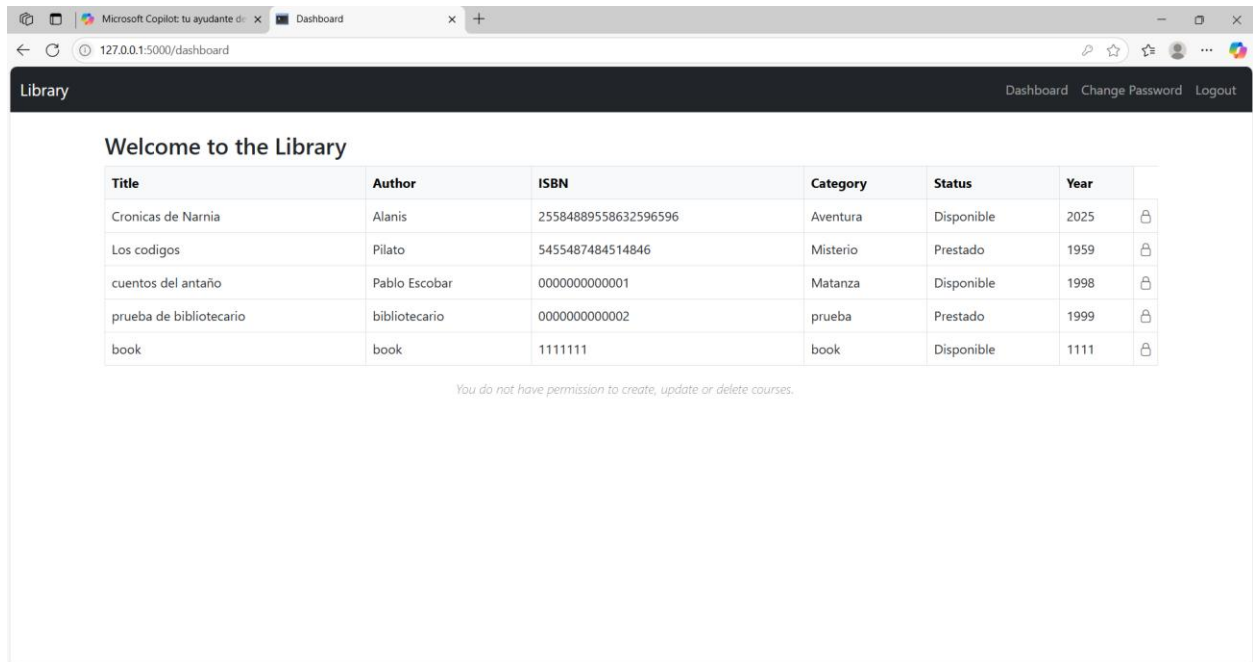
3. Al usuario registrarse correctamente le aparecerá una notificación diciendo: ‘Usuario registrado exitosamente’.



4. Aquí se presenta al usuario (‘Lector’) iniciar sesión luego de haber creado una cuenta.



5.La interfaz de ‘Dashboard’ del Lector donde se presenta información sobre los libros encontrados en la base de datos.

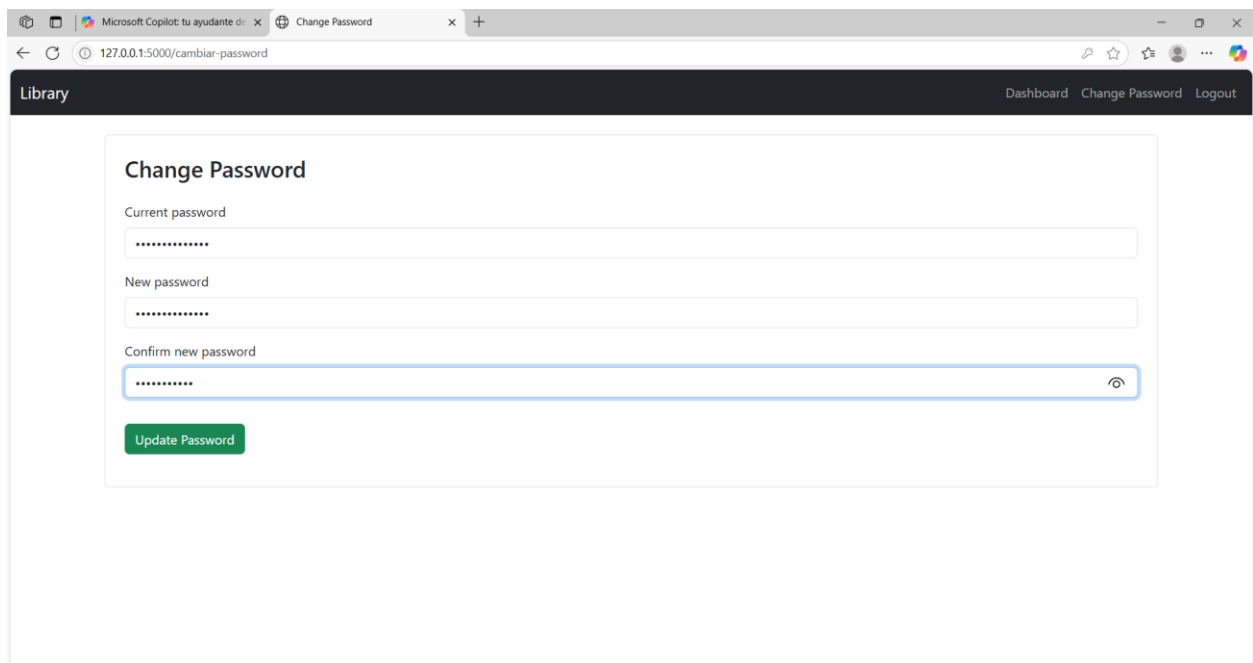


The screenshot shows a web browser window with the URL `127.0.0.1:5000/dashboard`. The page has a dark header with the word "Library" on the left and navigation links "Dashboard", "Change Password", and "Logout" on the right. Below the header, the main content area is titled "Welcome to the Library". It features a table with the following data:

Title	Author	ISBN	Category	Status	Year	
Cronicas de Narnia	Alanis	25584889558632596596	Aventura	Disponible	2025	🔒
Los codigos	Pilato	5455487484514846	Misterio	Prestado	1959	🔒
cuentos del antaño	Pablo Escobar	0000000000001	Matanza	Disponible	1998	🔒
prueba de bibliotecario	bibliotecario	0000000000002	prueba	Prestado	1999	🔒
book	book	1111111	book	Disponible	1111	🔒

Below the table, there is a message: "You do not have permission to create, update or delete courses."

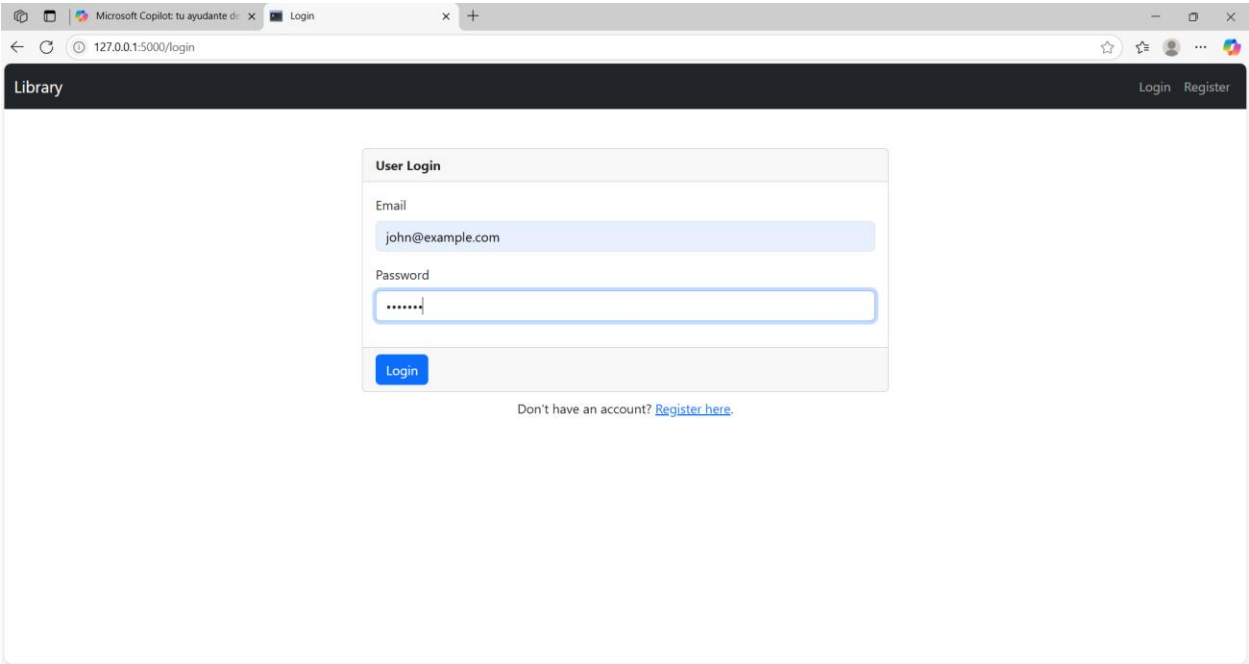
6.Aqui se presenta la interfaz de cambio de contraseña.



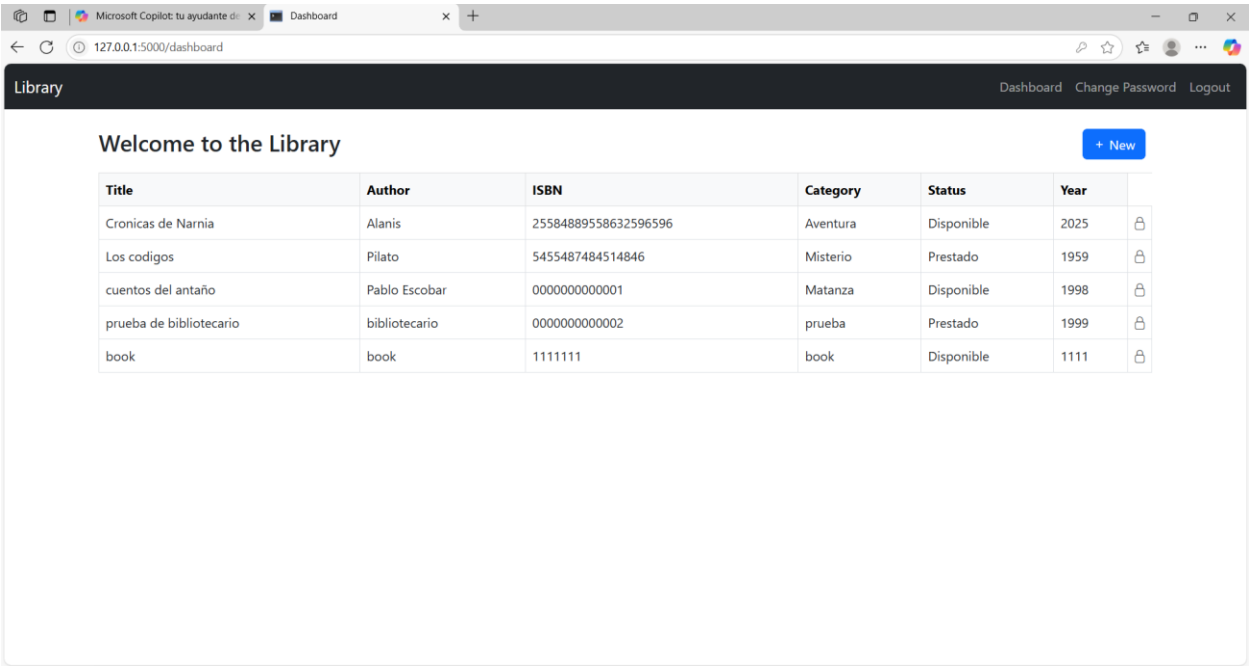
The screenshot shows a web browser window with the URL `127.0.0.1:5000/cambiar-password`. The page has a dark header with the word "Library" on the left and navigation links "Dashboard", "Change Password", and "Logout" on the right. The main content area is titled "Change Password" and contains three input fields for "Current password", "New password", and "Confirm new password". The "Confirm new password" field is highlighted with a blue border. Below the input fields is a green button labeled "Update Password".

Parte 2: Bibliotecario

1. Aqui se presenta al usuario ('Bibliotecario') iniciar sección.



2. Interfaz del 'Dashboard' del bibliotecario donde le presenta la información de los libros y le permite crear un libro nuevo.



3. Al entrar en new el sistema le pide al bibliotecario que ingrese la información pertinente sobre el libro:

a. Título

b. Autor

c. ISBN

d. Categoría del libro

e. Status

f. Año de publicación

g. Un botón de 'Save' para enviar la información a la base de datos para ser almacenada y presentarla en el 'Dashboard'.

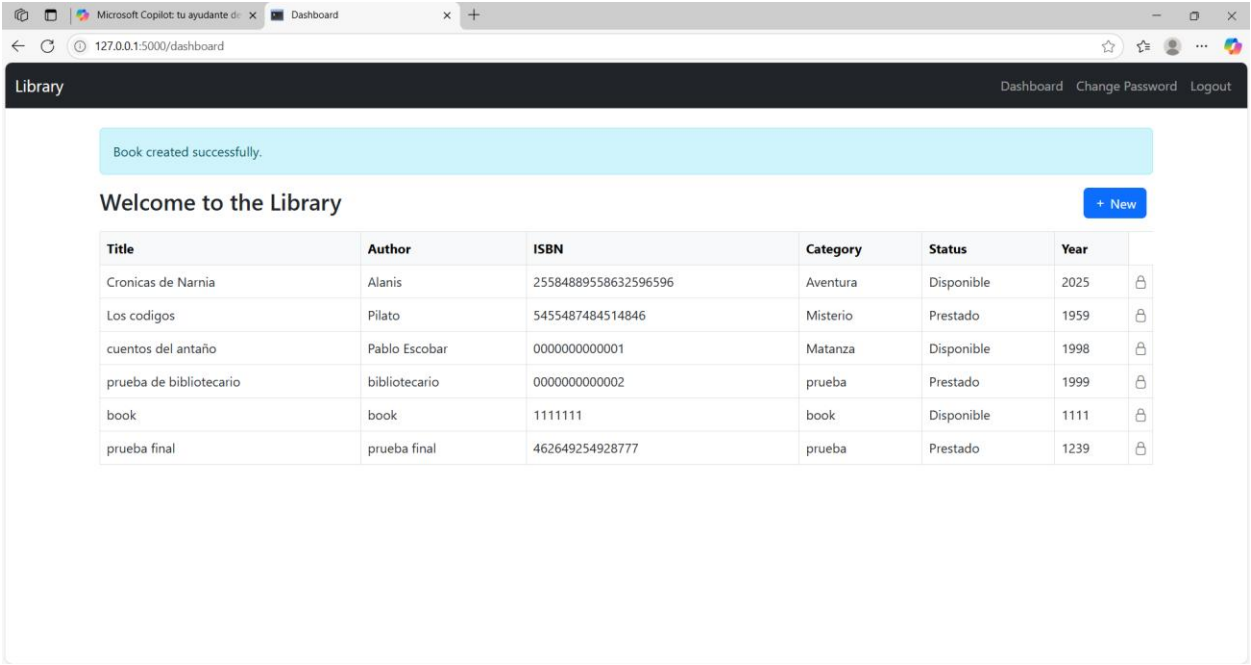
The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000/libros'. The browser tabs include 'Microsoft Copilot: tu ayudante di...' and 'New Book'. The page title is 'Library' and the navigation bar includes links for 'Dashboard', 'Change Password', and 'Logout'. The main content area is titled 'New Book' and contains a form with the following fields:

- Book title: prueba final
- Author name: prueba final
- Book ISBN: 462649254928777
- Book category: prueba
- Status: Prestado
- Publication Year: 1239

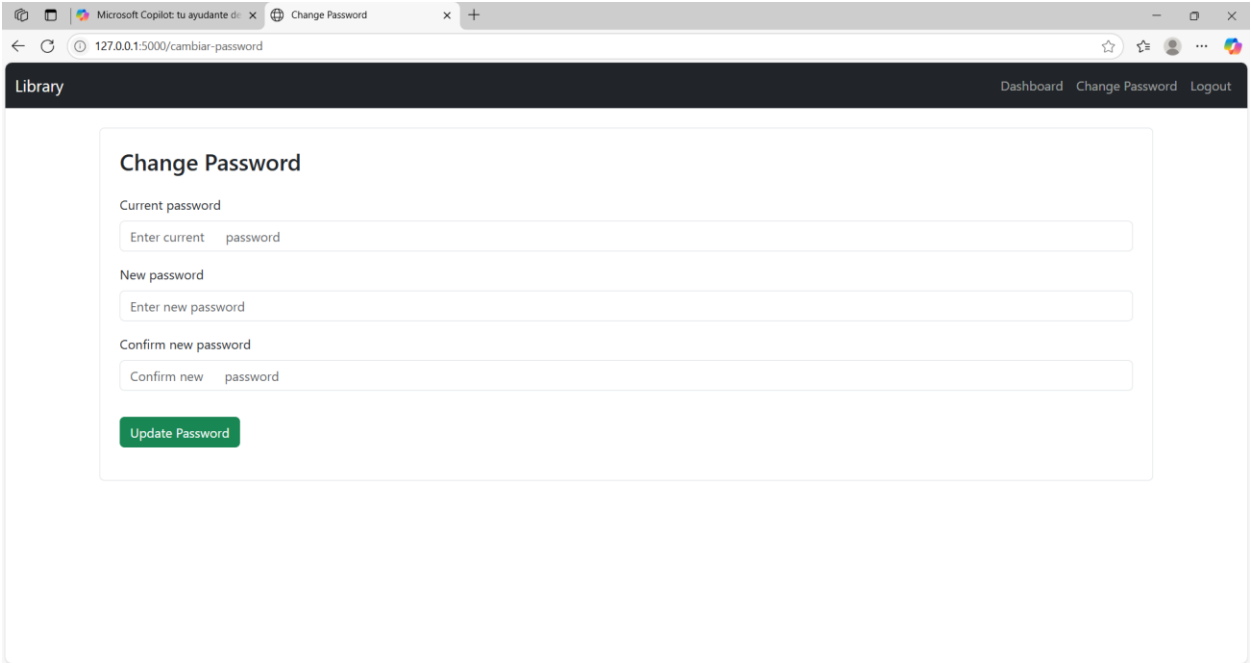
A blue 'Save' button is located at the bottom of the form.

4.Al crear el libro correctamente le aparecerá un mensaje diciendo: ‘Libro creado exitosamente’.

//Estuve intentando de darle permiso al bibliotecario para poder editar y eliminar, pero ni yo, ni los asistentes lograron concedérselo. //



5. De igual manera le aparece la pantalla de cambio de contraseña.



Parte 3: Administrador

1. Aquí se presenta al usuario('Admin') iniciar sección.

Library Login Register

User Login

Email
dastas@example.com

Password

Login

Don't have an account? [Register here.](#)

2.Interfaz del ‘Dashboard’ del administrador con sus opciones exclusivas:

a. tabla de usuarios

b. editar

c. eliminar libros













Microsoft Copilot: tu ayudante diDashboard

127.0.0.1:5000/dashboard

LibraryDashboardUsersChange PasswordLogout

Welcome to the Library

+ New

Title	Author	ISBN	Category	Status	Year	
Cronicas de Narnia	Alanis	25584889558632596596	Aventura	Disponible	2025	 
Los codigos	Pilato	5455487484514846	Misterio	Prestado	1959	 
cuentos del antaño	Pablo Escobar	0000000000001	Matanza	Disponible	1998	 
prueba de bibliotecario	bibliotecario	0000000000002	prueba	Prestado	1999	 
book	book	1111111	book	Disponible	1111	 
prueba final	prueba final	462649254928777	prueba	Prestado	1239	 

3.En esta interfaz se le presenta al ‘admin’ los usuarios registrados en la plataforma.

Microsoft Copilot: tu ayudante di...User List

127.0.0.1:5000/usuarios

LibraryDashboardUsersChange PasswordLogout

List of Registered Users

Username	Email	Role
yare	yare@example.com	Lector
john	john@example.com	Bibliotecario
dastas	dastas@example.com	Admin
jeremy	jeremy@example.com	Lector
super	super@example.com	Bibliotecario
prueba final	pruebafinal@example.com	Lector

4. Aquí se presenta la opción de editar la información de un libro con los siguientes campos:

a. Título

b. Autor

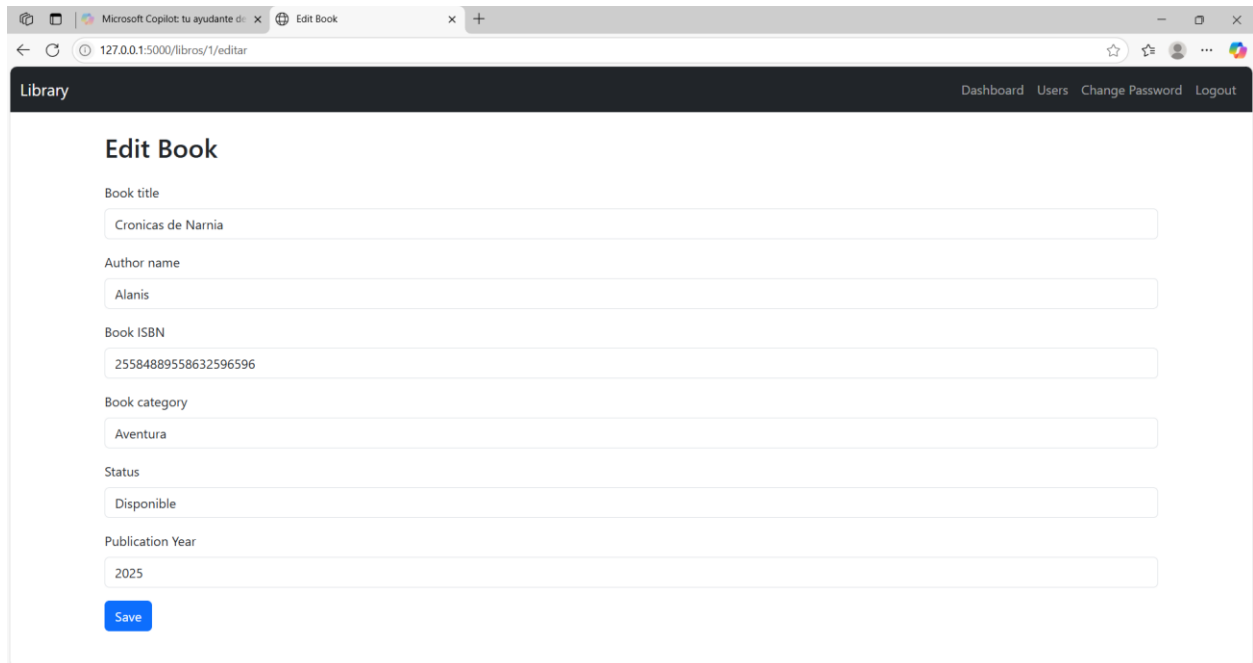
c. ISBN

d. Categoría

e. Status

f. Año de publicación

g. 'Save'

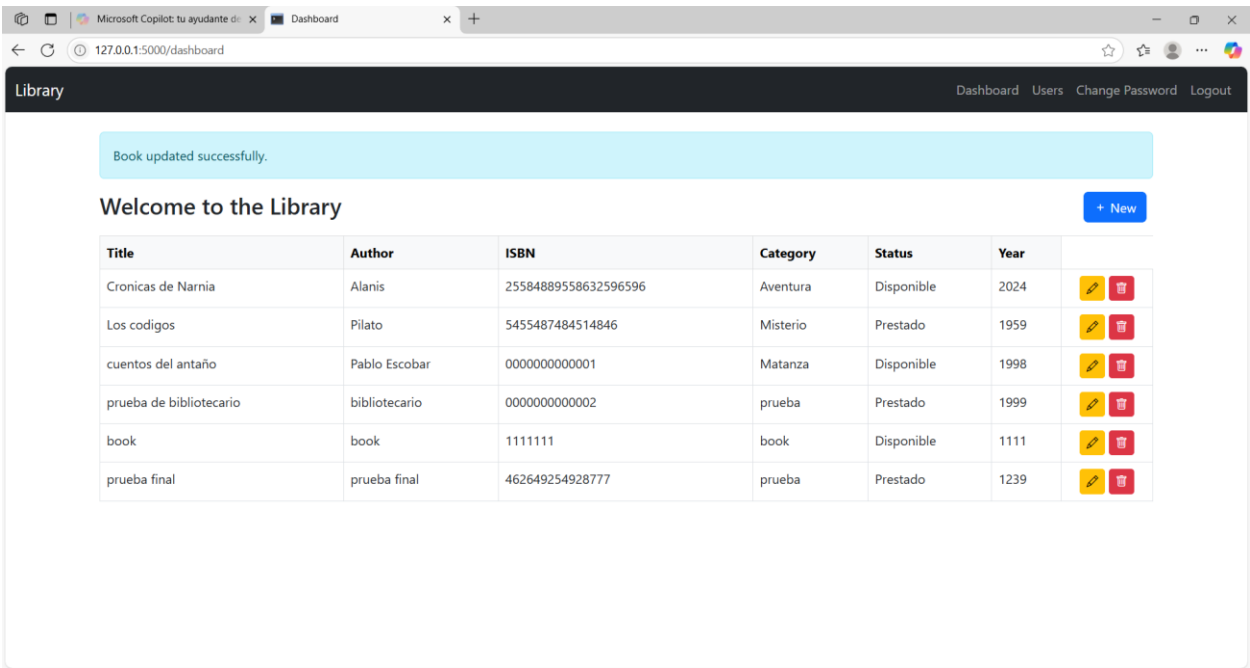


The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000/libros/1/editar'. The page has a dark header with the word 'Library' on the left and navigation links 'Dashboard', 'Users', 'Change Password', and 'Logout' on the right. The main content area is titled 'Edit Book' and contains a form with the following fields:

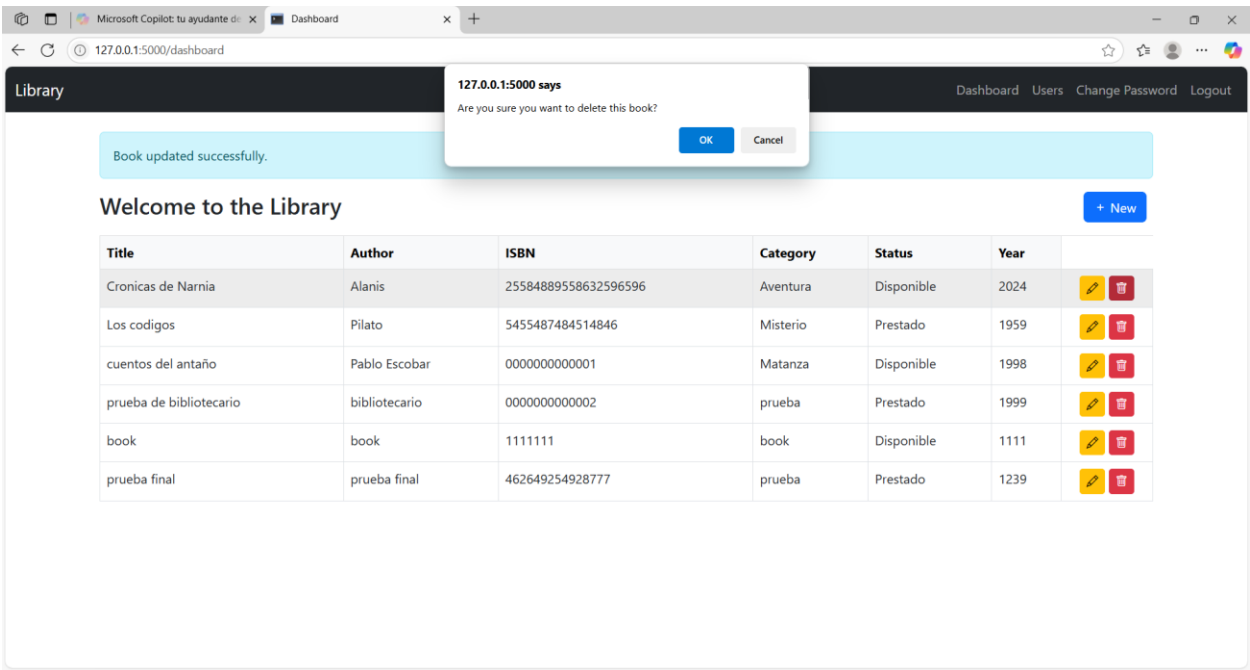
- Book title:
- Author name:
- Book ISBN:
- Book category:
- Status:
- Publication Year:

At the bottom of the form is a blue button labeled 'Save'.

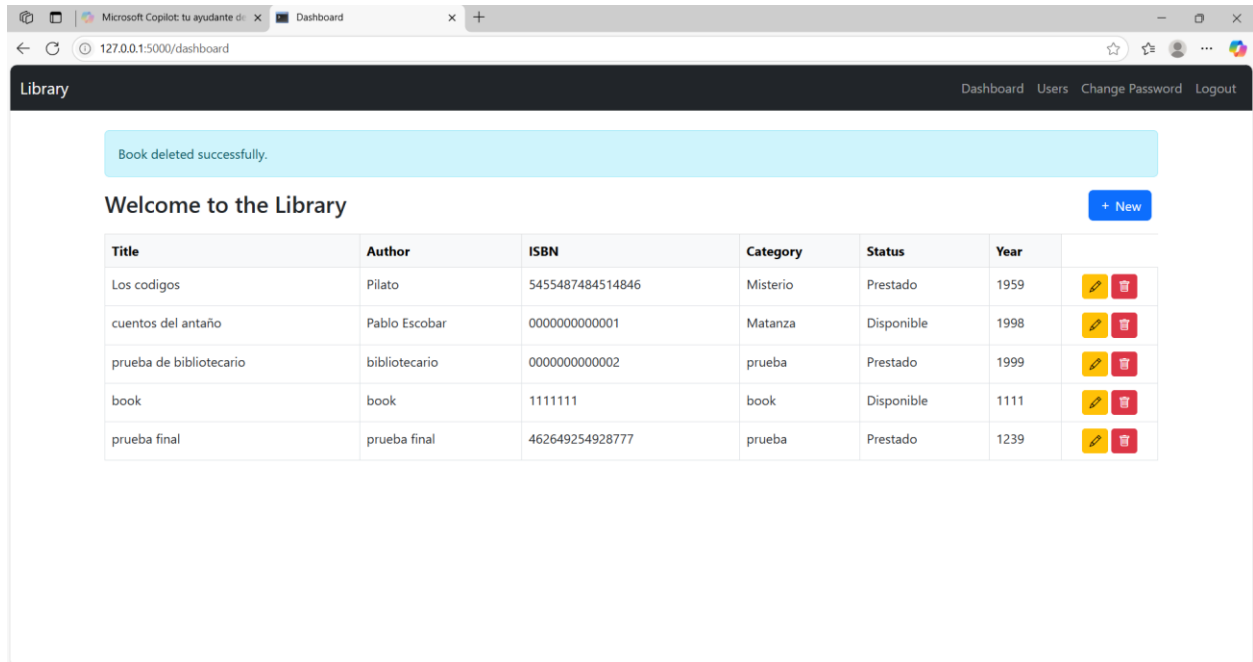
5. Luego de editar la información de un libro le aparece el siguiente mensaje: ‘Libro actualizado exitosamente’.













6. Al elegir la opción de borrar le aparece un mensaje diciendo: ‘Estas seguro que quieres eliminar este libro’.



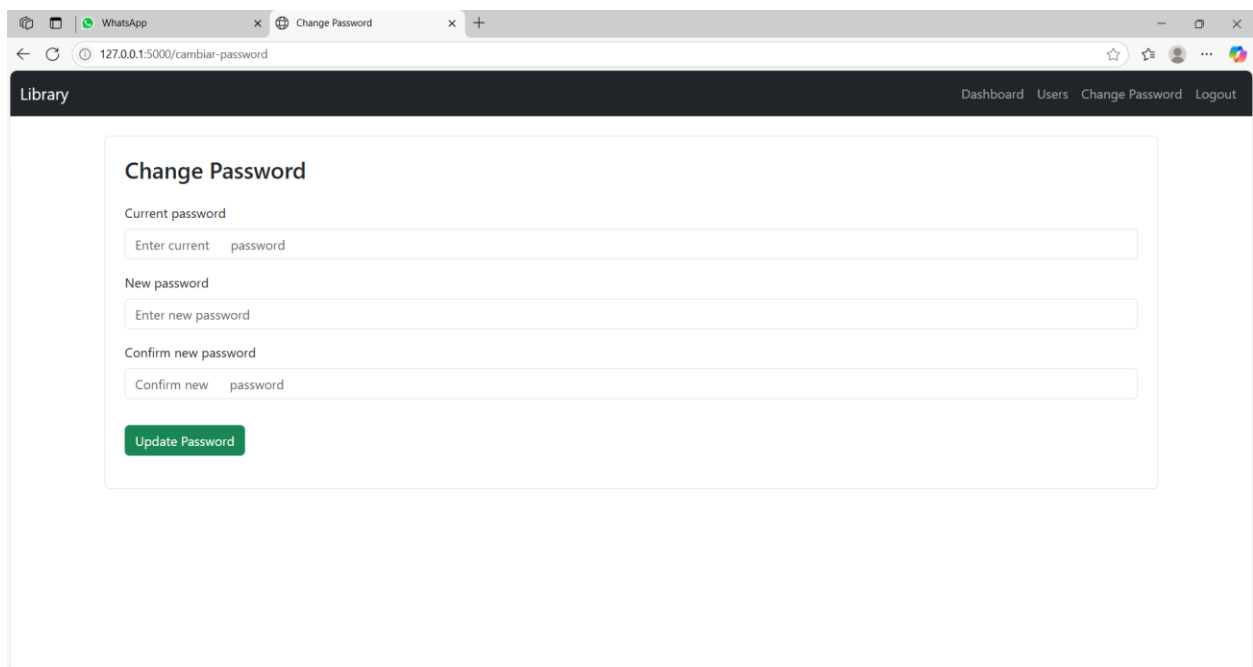
7. Al eliminar el libro le aparece el mensaje: 'Libro eliminado exitosamente'.



The screenshot shows a web browser window with the URL `127.0.0.1:5000/dashboard`. The page is titled "Library" and has a navigation bar with links: "Dashboard", "Users", "Change Password", and "Logout". A light blue message box at the top states "Book deleted successfully." Below this, the heading "Welcome to the Library" is followed by a "+ New" button. A table lists five books with columns for Title, Author, ISBN, Category, Status, and Year. Each row has edit and delete icons. The delete icon for the first book is highlighted in red.

Title	Author	ISBN	Category	Status	Year	
Los codigos	Pilato	5455487484514846	Misterio	Prestado	1959	 
cuentos del antaño	Pablo Escobar	0000000000001	Matanza	Disponible	1998	 
prueba de bibliotecario	bibliotecario	0000000000002	prueba	Prestado	1999	 
book	book	1111111	book	Disponible	1111	 
prueba final	prueba final	462649254928777	prueba	Prestado	1239	 

8. También tendrá la página de cambio de contraseña



The screenshot shows a web browser window with the URL `127.0.0.1:5000/cambiar-password`. The page is titled "Library" and has a navigation bar with links: "Dashboard", "Users", "Change Password", and "Logout". The main heading is "Change Password". Below it are three input fields: "Current password" (placeholder: "Enter current password"), "New password" (placeholder: "Enter new password"), and "Confirm new password" (placeholder: "Confirm new password"). A green "Update Password" button is at the bottom.

Título: Código de los ENDPOINTS

1. Este código permite que a través de una solicitud GET el cliente pueda recibir en formato JSON un listado de la información de los libros que se encuentran en la base de datos.

```
15 @main.route('/listar_libro', methods=['GET'])
16 def listar_libro():
17     """
18     Retorna una lista de libros (JSON).
19     """
20     libro = Libro.query.all()
21
22     data = [
23         {'id': libro.id, 'titulo': libro.titulo, 'autor': libro.autor, 'isbn': libro.isbn, 'categoria': libro.categoria,
24          'estado': libro.estado, 'año_publicacion': libro.año_publicacion, 'bibliotecario_id': libro.bibliotecario_id}
25         for libro in libro
26     ]
27     return jsonify(data), 200
28
```

2. Este código permite que a través de una solicitud GET le presente al cliente la información de un libro basándose en el ID del libro que el usuario coloque.

```
30 @main.route('/libro/<int:id>', methods=['GET'])
31 def listar_un_libro(id):
32     """
33     Retorna un solo libro por su ID (JSON).
34     """
35     libro = Libro.query.get_or_404(id)
36
37     data = {
38         'id': libro.id,
39         'titulo': libro.titulo,
40         'autor': libro.autor,
41         'isbn': libro.isbn,
42         'categoria': libro.categoria,
43         'estado': libro.estado,
44         'año_publicacion': libro.año_publicacion,
45         'bibliotecario_id': libro.bibliotecario_id
46     }
47
48     return jsonify(data), 200
```

3. Este código permite que el cliente a través de una solicitud POST pueda crear un libro nuevo ingresando la información pertinente del libro.

```
51 @main.route('/crear_libro', methods=['POST'])
52 def crear_libro():
53     """
54     Crea un libro sin validación.
55     Espera JSON con 'titulo', 'autor' y 'bibliotecario_id'.
56     """
57     data = request.get_json()
58
59     if not data:
60         return jsonify({'error': 'No input data provided'}), 400
61
62     nuevo_libro = Libro(
63         titulo=data.get('titulo'),
64         autor=data.get('autor'),
65         isbn=data.get('isbn'),
66         categoria=data.get('categoria'),
67         estado=data.get('estado'),
68         año_publicacion=data.get('año_publicacion'),
69         bibliotecario_id=data.get('bibliotecario_id') # sin validación de usuario
70     )
71
72     db.session.add(nuevo_libro)
73     db.session.commit()
74
75     return jsonify({'message': 'Libro creado', 'id': nuevo_libro.id, 'bibliotecario_id': nuevo_libro.bibliotecario_id}), 201
```

4. Este código permite que el usuario a través de una solicitud de PUT pueda actualizar la información de un libro ya existente basándose en el ID del libro que el cliente coloque.

```
77 @main.route('/actualizar_libro/<int:id>', methods=['PUT'])
78 def actualizar_libro(id):
79     """
80     Actualiza un libro sin validación de usuario o permisos.
81     """
82     libro = Libro.query.get_or_404(id)
83     data = request.get_json()
84
85     libro.titulo = data.get('titulo', libro.titulo)
86     libro.autor = data.get('autor', libro.autor)
87     libro.isbn = data.get('isbn', libro.isbn)
88     libro.categoria = data.get('categoria', libro.categoria)
89     libro.estado = data.get('estado', libro.estado)
90     libro.año_publicacion = data.get('año_publicacion', libro.año_publicacion)
91     libro.bibliotecario_id = data.get('bibliotecario_id', libro.bibliotecario_id)
92
93     db.session.commit()
94
95     return jsonify({'message': 'Libro actualizado', 'id': libro.id}), 200
```


5. Este código permite que el cliente a través de una solicitud DELETE pueda eliminar un libro basándose en el ID del libro que el cliente coloque.

```
97 @main.route('/eliminar_libro/<int:id>', methods=['DELETE'])
98 √ def eliminar_libro(id):
99 √     """
100     Elimina un libro sin validación de permisos.
101     """
102     libro = Libro.query.get_or_404(id)
103     db.session.delete(libro)
104     db.session.commit()
105
106     return jsonify({'message': 'Libro eliminado', 'id': libro.id}), 200
```

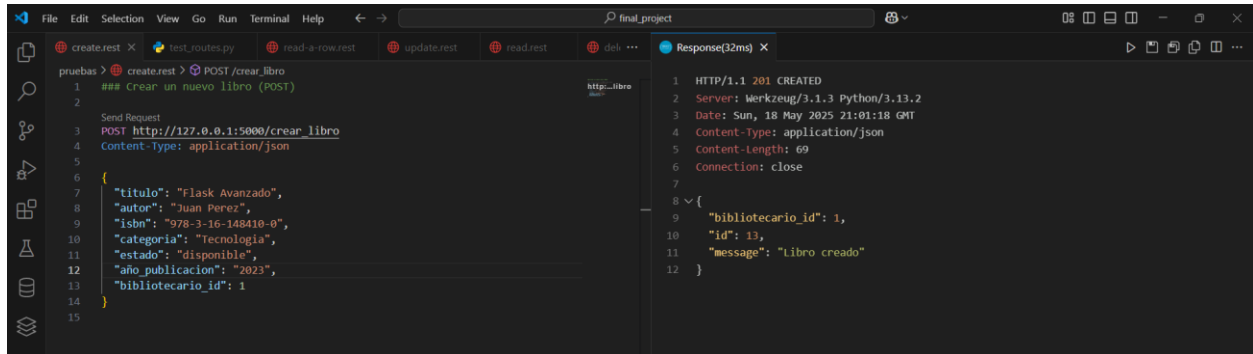
Tabla de datos enviados y recibidos:

Nombre del Archivo	Valores Enviados	Valores Esperados
create.rest	POST http://127.0.0.1:5000/crear_libro Content-Type: application/json <pre>{ "titulo":, "autor":, "isbn":, "categoria":, "estado":, "año_publicacion":, "bibliotecario_id": #ID }</pre>	<pre>{ "bibliotecario_id": #ID, "id"#ID Libro:, "message": "Libro creado" }</pre>
read-a-row.rest	GET http://127.0.0.1:5000/libro/#ID Libro Content-Type: application/json	<pre>{ "autor":, "a\u00f1o_publicacion":, "bibliotecario_id": #ID, "categoria":, "estado":, "id": #ID Libro, "isbn":, "titulo": }</pre>
read.rest	GET http://127.0.0.1:5000/listar_libro Content-Type: application/json	Presentará la información de todos los libros que se encuentran en la base de datos.
update.rest	PUT http://localhost:5000/actualizar_libro/#ID Libro Content-Type: application/json <pre>{ Información que se quiere actualizar }</pre>	<pre>{ "id": #ID Libro, "message": "Libro actualizado" }</pre>
delete.rest	DELETE http://localhost:5000/eliminar_libro/#ID Libro Content-Type: application/json	<pre>{ "id": # ID Libro, "message": "Libro eliminado" }</pre>

		}
--	--	---

Título: Pruebas de los ENDPOINTS del CRUD

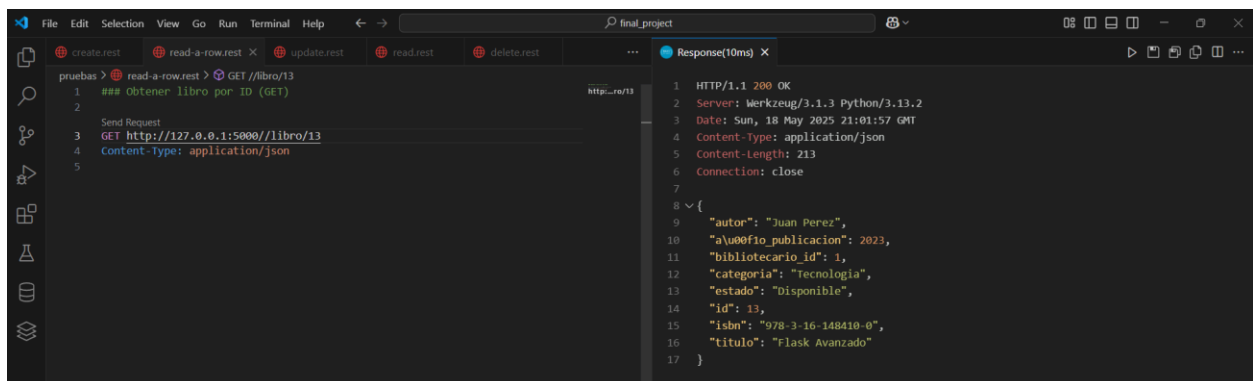
1. La captura muestra una prueba de una API REST para crear un libro usando el método POST en una aplicación Flask. Al realizarse correctamente la solicitud de registro en el sistema la API respondió con éxito, asignando el ID 13 al nuevo libro.



The screenshot shows a REST client interface with a 'create.rest' file. The request is a POST to 'http://localhost:5000/crear_libro' with a JSON body. The response is a 201 status code with a JSON body indicating the book was created with ID 13.

```
1 create.rest X test_routes.py read-a-row.rest update.rest read.rest del... Response(22ms) X
2
3 pruebas > create.rest > POST /crear_libro
4   ### Crear un nuevo libro (POST)
5
6   Send Request
7   POST http://127.0.0.1:5000/crear_libro
8   Content-Type: application/json
9
10  {
11    "titulo": "Flask Avanzado",
12    "autor": "Juan Perez",
13    "isbn": "978-3-16-148410-0",
14    "categoria": "Tecnologia",
15    "estado": "disponible",
16    "año_publicacion": "2023",
17    "bibliotecario_id": 1
18  }
19
20  Response(22ms) X
21  1 HTTP/1.1 201 CREATED
22  2 Server: Werkzeug/3.1.3 Python/3.13.2
23  3 Date: Sun, 18 May 2025 21:01:18 GMT
24  4 Content-Type: application/json
25  5 Content-Length: 69
26  6 Connection: close
27
28  8 {
29    "bibliotecario_id": 1,
30    "id": 13,
31    "message": "Libro creado"
32  }
```

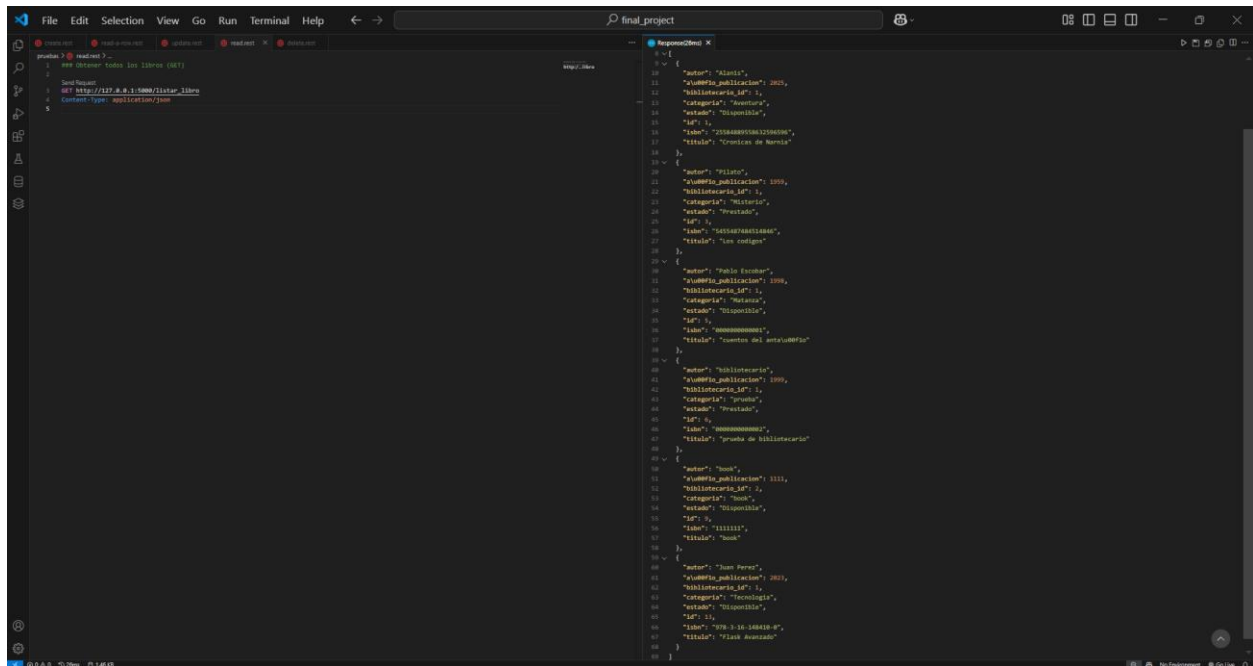
2. La captura muestra una solicitud GET a una API REST para obtener información de un libro con ID 13. La solicitud se envía, y la respuesta JSON devuelve detalles exclusivos del libro, incluyendo su título, autor, ISBN, año de publicación, categoría y estado.



The screenshot shows a REST client interface with a 'read-a-row.rest' file. The request is a GET to 'http://localhost:5000/libro/13'. The response is a 200 status code with a JSON body containing the book's details.

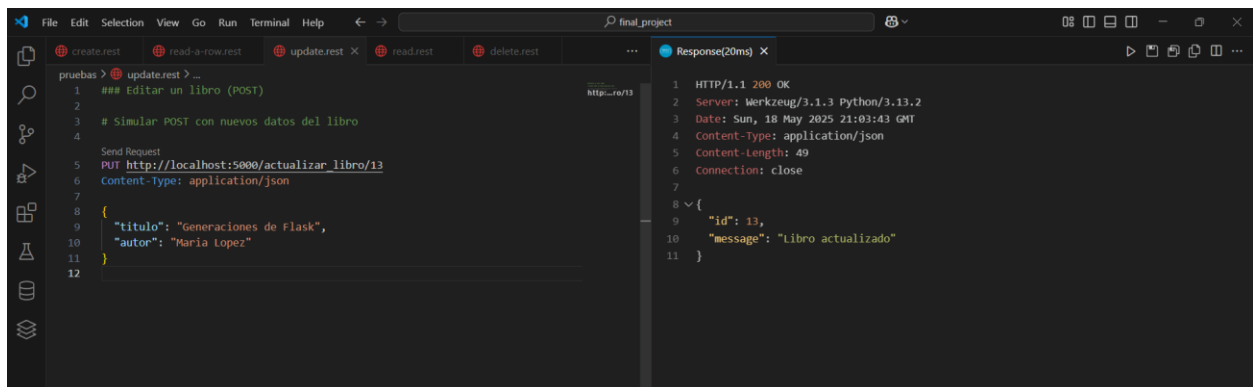
```
1 create.rest read-a-row.rest X update.rest read.rest delete.rest ... Response(10ms) X
2
3 pruebas > read-a-row.rest > GET /libro/13
4   ### Obtener libro por ID (GET)
5
6   Send Request
7   GET http://127.0.0.1:5000/libro/13
8   Content-Type: application/json
9
10  Response(10ms) X
11  1 HTTP/1.1 200 OK
12  2 Server: Werkzeug/3.1.3 Python/3.13.2
13  3 Date: Sun, 18 May 2025 21:01:57 GMT
14  4 Content-Type: application/json
15  5 Content-Length: 213
16  6 Connection: close
17
18  8 {
19    "autor": "Juan Perez",
20    "año_publicacion": 2023,
21    "bibliotecario_id": 1,
22    "categoria": "Tecnologia",
23    "estado": "Disponible",
24    "id": 13,
25    "isbn": "978-3-16-148410-0",
26    "titulo": "Flask Avanzado"
27  }
```

3. La captura presenta una solicitud GET para obtener la información sobre todos los libros encontrados en la base de datos. La API responde con un objeto JSON con la información de todos los libros que existen en la base de datos.



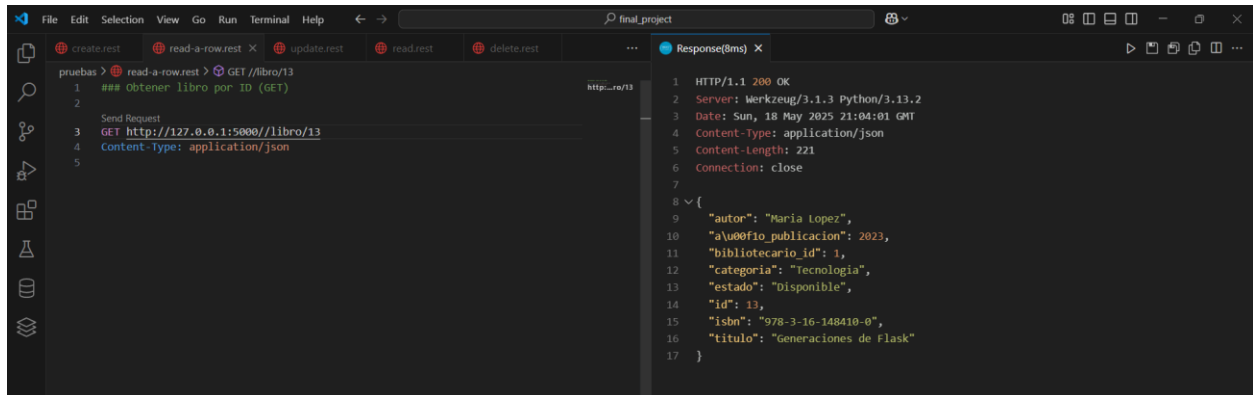
```
File Edit Selection View Go Run Terminal Help  
pruebas > read-rest > GET http://127.0.0.1:5000/listar_libros  
Send Request  
Response (200ms)  
{  
  "author": "Miguel",  
  "autor_publicacion": 2001,  
  "biblioteca_id": 1,  
  "categoria": "Misterio",  
  "estado": "disponible",  
  "id": 1,  
  "isbn": "9780451001700",  
  "titulo": "Crónicas de Narnia"  
},  
  {  
    "author": "Elton",  
    "autor_publicacion": 1999,  
    "biblioteca_id": 1,  
    "categoria": "Misterio",  
    "estado": "prestado",  
    "id": 2,  
    "isbn": "9780451001700",  
    "titulo": "Los castigos"  
  },  
  {  
    "author": "Pablo Escobar",  
    "autor_publicacion": 1999,  
    "biblioteca_id": 1,  
    "categoria": "Misterio",  
    "estado": "disponible",  
    "id": 3,  
    "isbn": "9780451001700",  
    "titulo": "Cuentos del asesino"  
  },  
  {  
    "author": "Bibliotecario",  
    "autor_publicacion": 1999,  
    "biblioteca_id": 1,  
    "categoria": "Prueba",  
    "estado": "prestado",  
    "id": 4,  
    "isbn": "9780451001700",  
    "titulo": "Prueba de Bibliotecario"  
  },  
  {  
    "author": "Nora",  
    "autor_publicacion": 1111,  
    "biblioteca_id": 1,  
    "categoria": "Nora",  
    "estado": "disponible",  
    "id": 5,  
    "isbn": "1111111",  
    "titulo": "Nora"  
  },  
  {  
    "author": "Juan Perez",  
    "autor_publicacion": 2001,  
    "biblioteca_id": 1,  
    "categoria": "Misterio",  
    "estado": "disponible",  
    "id": 6,  
    "isbn": "978-0-10-100000-0",  
    "titulo": "El fin de la guerra"  
  }  
}
```

4. La captura presenta una solicitud de actualización de datos del libro con ID 13 utilizando un PUT para actualizar el título y el autor. El API responde con un mensaje: 'Libro actualizado'.



```
File Edit Selection View Go Run Terminal Help  
pruebas > update-rest > PUT http://localhost:5000/actualizar_libro/13  
Send Request  
Response (200ms)  
1 HTTP/1.1 200 OK  
2 Server: Werkzeug/3.1.3 Python/3.13.2  
3 Date: Sun, 18 May 2025 21:03:43 GMT  
4 Content-Type: application/json  
5 Content-Length: 49  
6 Connection: close  
7  
8 {  
9   "id": 13,  
10  "message": "Libro actualizado"  
11 }
```

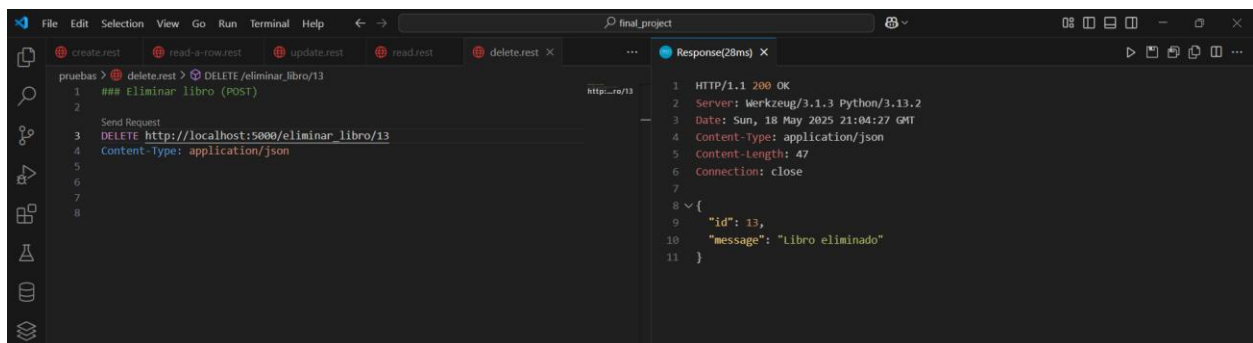
5. La captura presenta la información de la actualización del libro con ID 13.



```
File Edit Selection View Go Run Terminal Help ← → final_project
create.rest read-a-row.rest update.rest read.rest delete.rest ... Response(8ms) X
pruebas > read-a-row.rest > GET //libro/13
1 ### Obtener libro por ID (GET) http://ro/13
2
3 Send Request
4 GET http://127.0.0.1:5000//libro/13
5 Content-Type: application/json

1 HTTP/1.1 200 OK
2 Server: Werkzeug/3.1.3 Python/3.13.2
3 Date: Sun, 18 May 2025 21:04:01 GMT
4 Content-Type: application/json
5 Content-Length: 221
6 Connection: close
7
8 {
9   "autor": "Maria Lopez",
10  "año00fio_publicacion": 2023,
11  "bibliotecario_id": 1,
12  "categoria": "Tecnologia",
13  "estado": "Disponible",
14  "id": 13,
15  "isbn": "978-3-16-148410-0",
16  "titulo": "Generaciones de Flask"
17 }
```

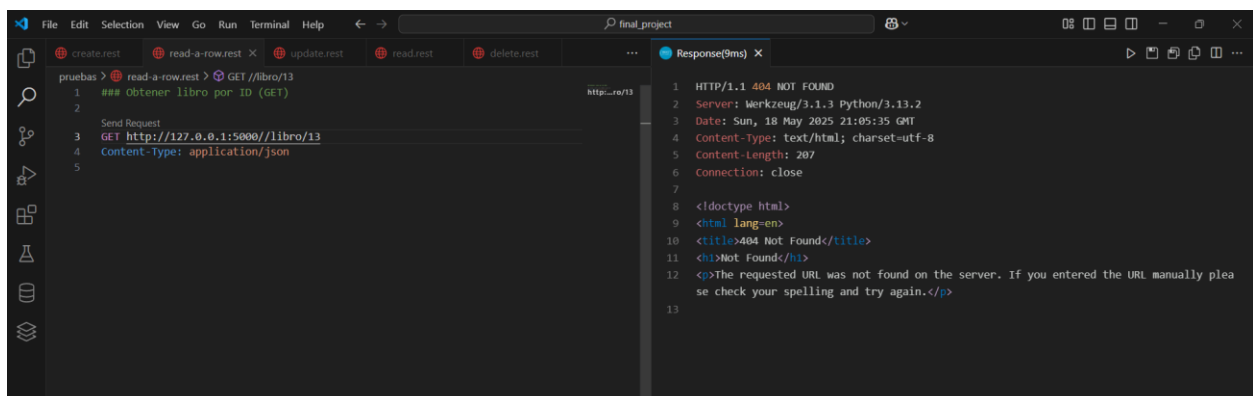
6. La siguiente captura presenta una solicitud para eliminar un libro utilizando el ID del libro a través de una solicitud DELETE. El API responde con el mensaje: 'Libro eliminado'.



```
File Edit Selection View Go Run Terminal Help ← → final_project
create.rest read-a-row.rest update.rest read.rest delete.rest ... Response(28ms) X
pruebas > delete.rest > DELETE /eliminar_libro/13
1 ### Eliminar libro (POST) http://ro/13
2
3 Send Request
4 DELETE http://localhost:5000/eliminar_libro/13
5 Content-Type: application/json
6
7
8

1 HTTP/1.1 200 OK
2 Server: Werkzeug/3.1.3 Python/3.13.2
3 Date: Sun, 18 May 2025 21:04:27 GMT
4 Content-Type: application/json
5 Content-Length: 47
6 Connection: close
7
8 {
9   "id": 13,
10  "message": "Libro eliminado"
11 }
```

7. Aquí se presenta que al eliminar el libro no aparece nuevamente en la base de datos.



```
File Edit Selection View Go Run Terminal Help ← → final_project
create.rest read-a-row.rest update.rest read.rest delete.rest ... Response(9ms) X
pruebas > read-a-row.rest > GET //libro/13
1 ### Obtener libro por ID (GET) http://ro/13
2
3 Send Request
4 GET http://127.0.0.1:5000//libro/13
5 Content-Type: application/json

1 HTTP/1.1 404 NOT FOUND
2 Server: Werkzeug/3.1.3 Python/3.13.2
3 Date: Sun, 18 May 2025 21:05:35 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 207
6 Connection: close
7
8 <!doctype html>
9 <html lang=en>
10 <title>404 Not Found</title>
11 <h1>Not Found</h1>
12 <p>The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.</p>
13
```

Título: GitHub de los integrantes

Nombre del Integrante	Carpeta en GitHub
Jeremy A. Ayende Santiago	https://github.com/ayende10/proyecto_final.git
Eloy Toledo De Jesús	https://github.com/Eloy841/Proyecto_final.git