

UNIVERSIDAD INTERAMERICANA DE PUERTO RICO – RECINTO DE ARECIBO

Programa de Ciencias de Computadora

Gestor de biblioteca

COMP 2052 – “Back-End”

Kivan M. López Raíces - R00642441

Krystal S. Pérez Rosado – R00642760

Github repositorios:

Kivan M. Lopez Raices — [COMP-2052/final_project at main · Kaiatuni/COMP-2052 · GitHub](#)

Krystal S. Perez Rosado - [https://github.com/Seliz05/COMP-2052/tree/main/final_project%20\(2\)/final_project](https://github.com/Seliz05/COMP-2052/tree/main/final_project%20(2)/final_project)

Contenido:

Base De Datos:

(la hicimos pero se me borro algunas cosas)

```

COMP 2052 > final_project > sqldelebreria.SQL > ...
  ▷ Run | Active: \ah\gestor_biblioteca
1  DROP DATABASE IF EXISTS gestor_biblioteca;
  ▷ Run
2  CREATE DATABASE gestor_biblioteca CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
  ▷ Run
3  USE gestor_biblioteca;
4
  ▷ Run | Select
5  CREATE TABLE role (
6      id INT AUTO_INCREMENT PRIMARY KEY,
7      name VARCHAR(64) UNIQUE
8  );
9
  ▷ Run | Select
10 CREATE TABLE user (
11     id INT AUTO_INCREMENT PRIMARY KEY,
12     username VARCHAR(64) UNIQUE,
13     email VARCHAR(120) UNIQUE,
14     password_hash VARCHAR(256),
15     role_id INT,
16     FOREIGN KEY (role_id) REFERENCES role(id)
17 );
18
  ▷ Run | Select
19 CREATE TABLE libro (
20     id INT AUTO_INCREMENT PRIMARY KEY,
21     titulo VARCHAR(150),
22     autor VARCHAR(100),
23     isbn VARCHAR(20),
24     categoria VARCHAR(50),
25     estado ENUM('Disponible', 'Prestado'),
26     anio_publicacion INT
27 );
28
  ▷ Run
29 INSERT INTO role (name) VALUES ('Admin'), ('Bibliotecario'), ('Lector');

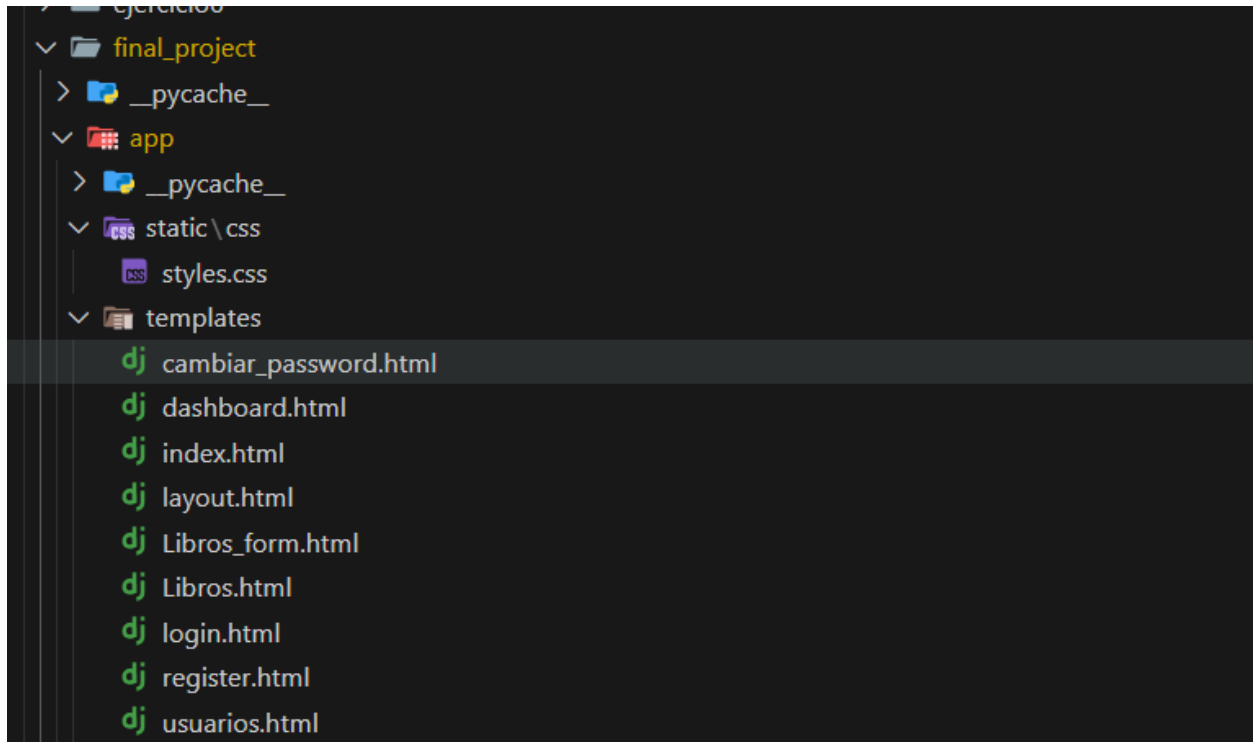
```

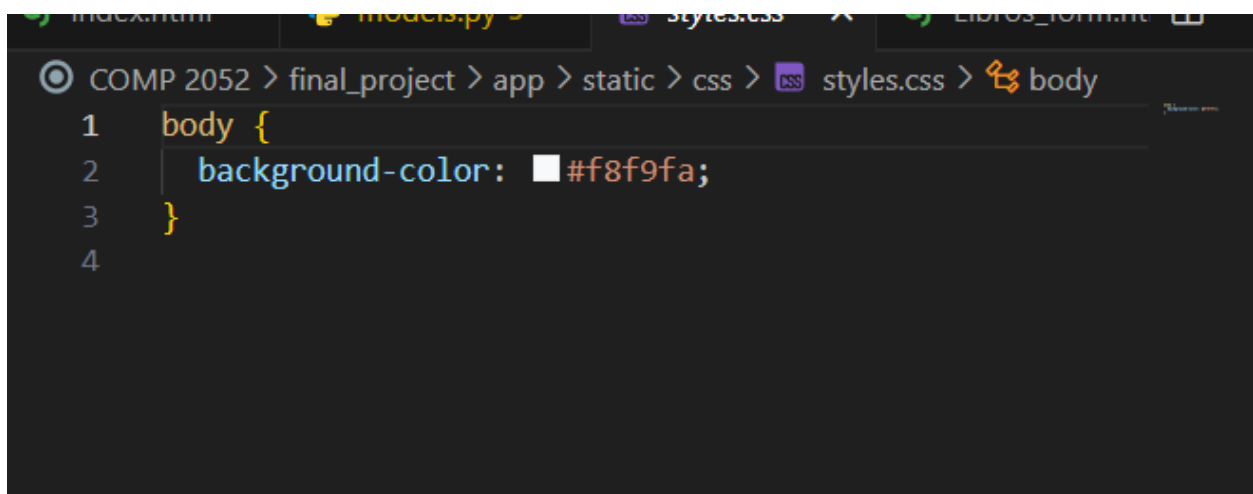
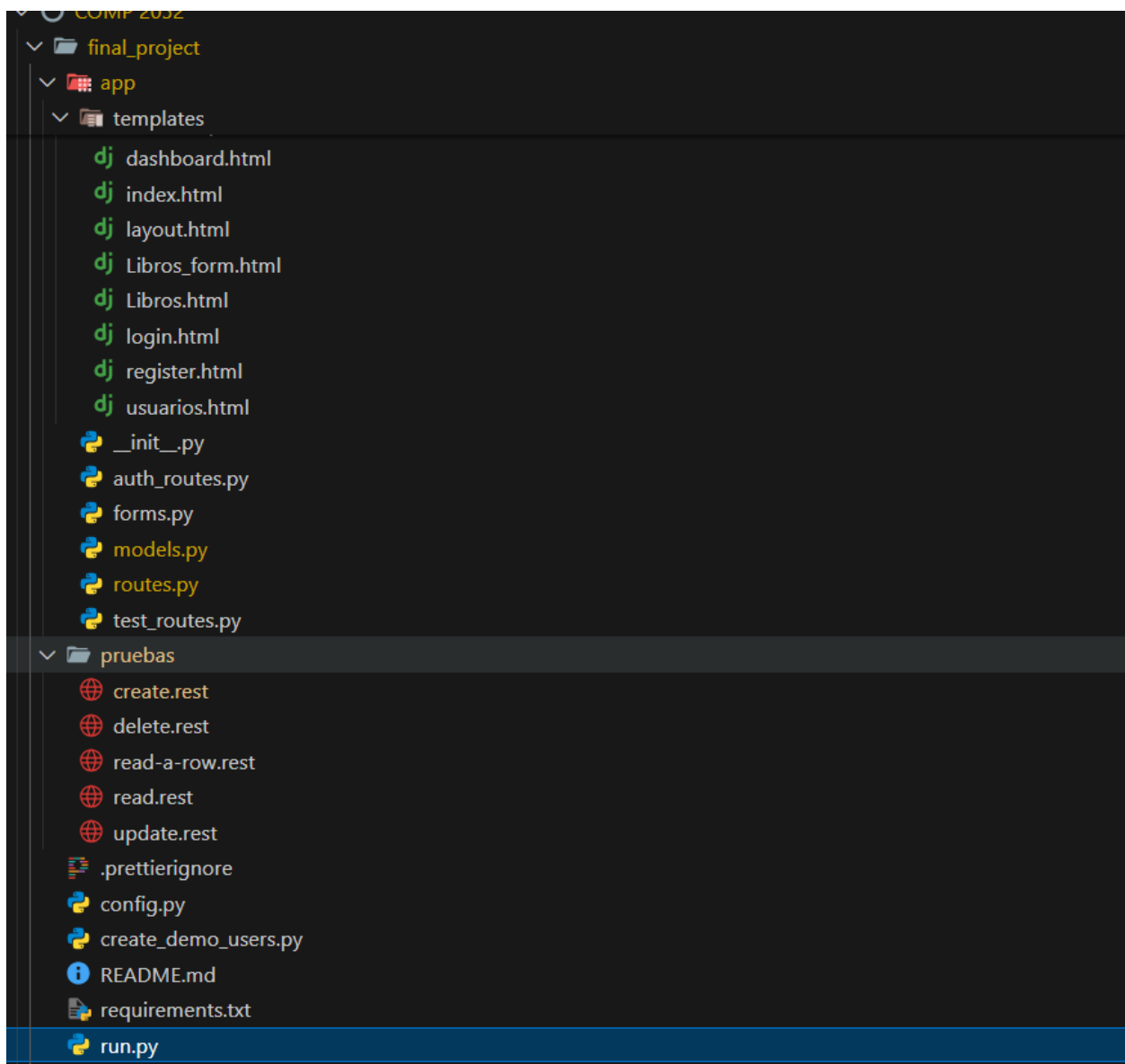
```

34
35
36
  ▷ Run | Select
37 ALTER TABLE libro
38 ADD COLUMN bibliotecario_id INT NULL,
39 ADD CONSTRAINT fk_bibliotecario
40     FOREIGN KEY (bibliotecario_id) REFERENCES user(id);
41
  ▷ Run
42 use gestionar_libros

```

Estructura de Python:

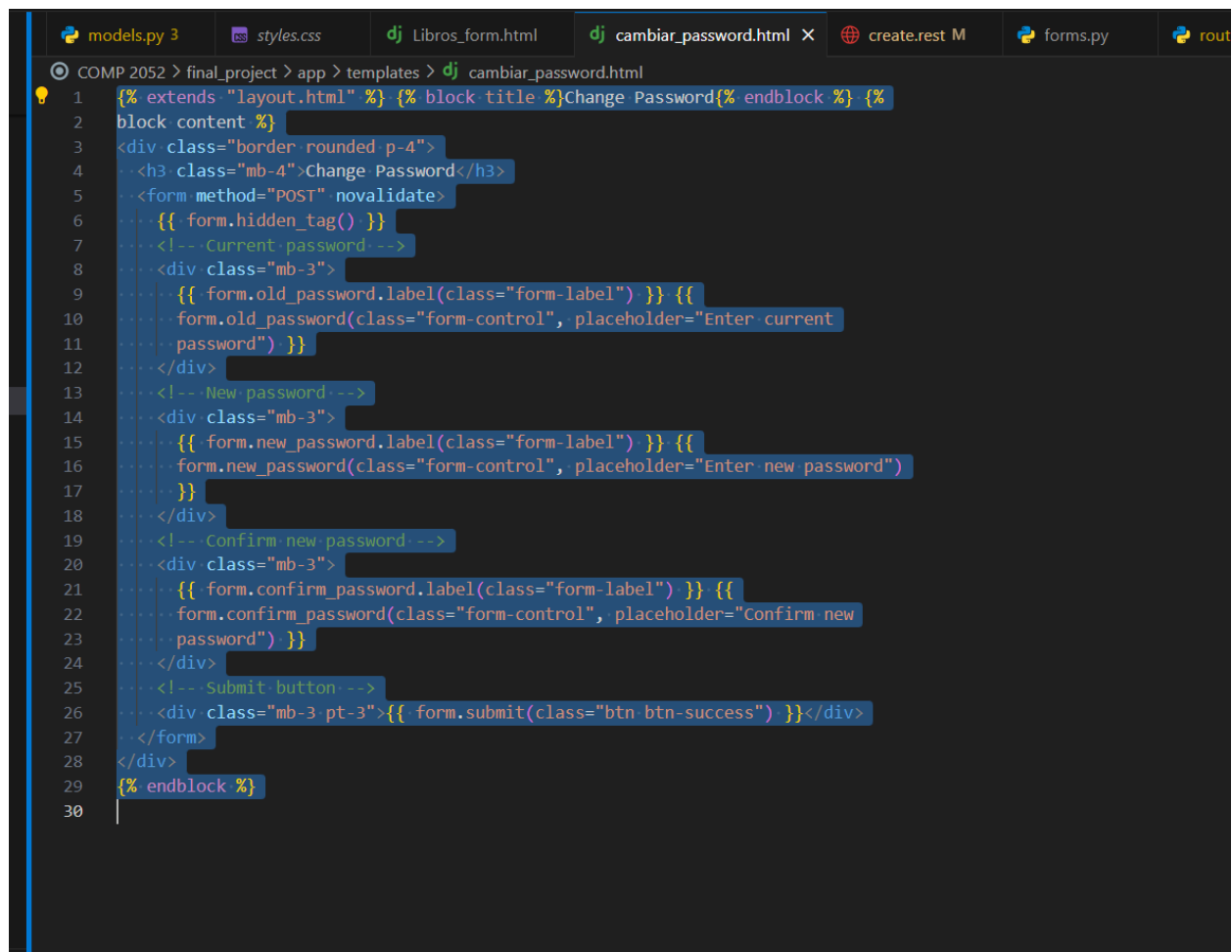




Esto le cambia al web page el background con css.

Folder de templates:

Cambiar_password.html:



The screenshot shows a code editor with several tabs at the top: models.py 3, styles.css, dj Libros_form.html, dj cambiar_password.html X, create.rest M, forms.py, and rout. The active tab is 'dj cambiar_password.html'. The editor displays the content of this template file, which is a Django template that extends 'layout.html'. It contains a form for changing a password, with fields for current password, new password, and confirm new password, along with a submit button. The form is styled with Bootstrap classes like 'border rounded p-4', 'mb-4', 'mb-3', and 'pt-3'. The code is as follows:

```
1 {% extends "layout.html" %} {% block title %}Change Password{% endblock %} {%
2 block content %}
3 <div class="border rounded p-4">
4   <h3 class="mb-4">Change Password</h3>
5   <form method="POST" novalidate>
6     {{ form.hidden_tag() }}
7     <!-- Current password -->
8     <div class="mb-3">
9       {{ form.old_password.label(class="form-label") }} {{
10       form.old_password(class="form-control", placeholder="Enter current
11       password") }}
12     </div>
13     <!-- New password -->
14     <div class="mb-3">
15       {{ form.new_password.label(class="form-label") }} {{
16       form.new_password(class="form-control", placeholder="Enter new password")
17       }}
18     </div>
19     <!-- Confirm new password -->
20     <div class="mb-3">
21       {{ form.confirm_password.label(class="form-label") }} {{
22       form.confirm_password(class="form-control", placeholder="Confirm new
23       password") }}
24     </div>
25     <!-- Submit button -->
26     <div class="mb-3 pt-3">{{ form.submit(class="btn btn-success") }}</div>
27   </form>
28 </div>
29 {% endblock %}
30
```

Esto permite cambiar el password en la interfaz.

Dashboard.html:

Esta es la interfaz de la pagina principal

```
models.py 3  dj Libros_form.html  dj cambiar_password.html  dj dashboard.html x  create.rest M  forms.py

COMP 2052 > final_project > app > templates > dj dashboard.html
1  {% extends "layout.html" %}
2  {% block title %}Dashboard{% endblock %}
3  {% block content %}
4  <div class="row">
5      <div class="col">
6          <h3>Gestión de Libros</h3>
7      </div>
8      <div class="col text-end">
9          {% if current_user.role.name != 'Lector' %}
10         <a class="btn btn-primary mb-3 me-2" href="{{ url_for('main.libros') }}">
11             <i class="bi bi-plus"></i> Nuevo Libro
12         </a>
13         {% endif %}
14     </div>
15 </div>
16
17 <table class="table table-bordered table-hover">
18     <thead class="table-light">
19         <tr>
20             <th>Título</th>
21             <th>Autor</th>
22             <th>ISBN</th>
23             <th>Categoría</th>
24             <th>Año Publicación</th>
25             <th>Estado</th>
26             <th class="text-center">Acciones</th>
27         </tr>
28     </thead>
29     <tbody>
30         {% for libro in libros %}
31         <tr>
32             <td>{{ libro.titulo }}</td>
33             <td>{{ libro.autor }}</td>
34             <td>{{ libro.isbn }}</td>
35             <td>{{ libro.categoria }}</td>
36             <td>{{ libro.anio_publicacion }}</td>
37             <td>{{ libro.estado }}</td>
38             <td class="text-center"></td>
```

```

<tr>
  <td>{{ libro.isbn }}</td>
  <td>{{ libro.categoria }}</td>
  <td>{{ libro.anio_publicacion }}</td>
  <td>{{ libro.estado }}</td>
  <td class="text-center ps-0 pe-0">
    {% if current_user.role.name == 'Admin' or current_user.role.name == 'Bibliotecario' %}
    <a
      class="btn btn-sm btn-warning"
      href="{{ url_for('main.editar_libro', id=libro.id) }}"
      title="Editar libro"
    >
      <i class="bi bi-pencil"></i>
    </a>
    <form
      method="POST"
      action="{{ url_for('main.eliminar_libro', id=libro.id) }}"
      style="display: inline"
      onsubmit="return confirm('¿Seguro que quieres eliminar este libro?');"
    >
      <button
        type="submit"
        class="btn btn-sm btn-danger"
        title="Eliminar libro"
      >
        <i class="bi bi-trash"></i>
      </button>
    </form>
    {% else %}
    <span class="text-muted"><i class="bi bi-lock"></i></span>
    {% endif %}
  </td>
</tr>
{% else %}
<tr>

```

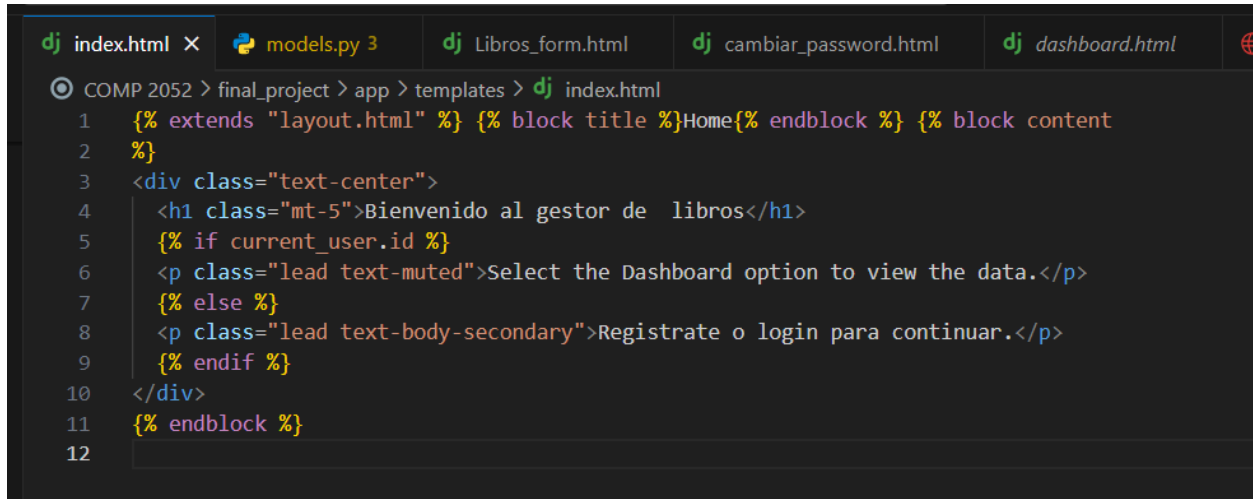
```

63     {% endif %}
64   </td>
65 </tr>
66 {% else %}
67 <tr>
68   <td colspan="7" class="text-center">No hay libros registrados.</td>
69 </tr>
70 {% endfor %}
71 </tbody>
72 </table>
73
74 {% if current_user.role.name == 'Lector' %}
75 <p class="text-center pe-3 mt-0 text-body-tertiary fw-lighter fst-italic">
76   No tienes permiso para crear, editar o eliminar libros.
77 </p>
78 {% endif %}
79 {% endblock %}

```


Index.html:

Página principal para el dashboard



```
COMP 2052 > final_project > app > templates > dj index.html
1  {% extends "layout.html" %} {% block title %}Home{% endblock %} {% block content
2  %}
3  <div class="text-center">
4      <h1 class="mt-5">Bienvenido al gestor de libros</h1>
5      {% if current_user.id %}
6      <p class="lead text-muted">Select the Dashboard option to view the data.</p>
7      {% else %}
8      <p class="lead text-body-secondary">Registrate o login para continuar.</p>
9      {% endif %}
10 </div>
11 {% endblock %}
12
```

Layout.html:

Parte de como se ve mas o menos como un css

show databases; Untitled-1 • mysql.sql dj layout.html X

COMP 2052 > final_project > app > templates > dj layout.html

```

1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8" />
5      <title>{% block title %}Gestión de Libros{% endblock %}</title>
6      <link
7        href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
8        rel="stylesheet"
9      />
10     <link
11       href="https://cdn.jsdelivr.net/npm/bootstrap-icons/font/bootstrap-icons.css"
12       rel="stylesheet"
13     />
14   </head>
15   <body>
16     <!-- Top navigation bar -->
17     <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
18       <div class="container-fluid">
19         <a class="navbar-brand" href="{% url_for('main.index') %}">Biblioteca</a>
20
21         <!-- Mobile toggle button -->
22         <button
23           class="navbar-toggler"
24           type="button"
25           data-bs-toggle="collapse"
26           data-bs-target="#navbarContent"
27           aria-controls="navbarContent"
28           aria-expanded="false"
29           aria-label="Toggle navigation"
30         >
31           <span class="navbar-toggler-icon"></span>
32         </button>
33
34         <!-- Collapsible navbar content -->
35         <div class="collapse navbar-collapse" id="navbarContent">
36           <ul class="navbar-nav ms-auto mb-2 mb-lg-0">
37             {% if current_user.is_authenticated %}
38               <li><a href="{% url_for('main.logout') %}">Logout</a></li>
39             {% else %}
40               <li><a href="{% url_for('main.login') %}">Login</a></li>
41             {% endif %}
42           </ul>
43         </div>
44       </div>
45     </nav>
46
47     <!-- Main content area -->
48     <div class="container mt-4">
49       <h2>Gestión de Libros</h2>
50       <div class="row">
51         <div class="col-md-3">
52           <div class="card">
53             <div class="card-body">
54               <h3>Agregar Libro</h3>
55               <form>
56                 <input type="text" value="Título del libro" />
57                 <input type="text" value="Autor" />
58                 <input type="text" value="Categoría" />
59                 <input type="text" value="ISBN" />
60                 <input type="button" value="Agregar" />
61               </form>
62             </div>
63           </div>
64         <div class="col-md-9">
65           <div class="card">
66             <div class="card-body">
67               <h3>Lista de Libros</h3>
68               <table>
69                 <thead>
70                   <tr>
71                     <th>ID</th>
72                     <th>Título</th>
73                     <th>Autor</th>
74                     <th>Categoría</th>
75                     <th>ISBN</th>
76                   </tr>
77                 </thead>
78                 <tbody>
79                   <tr>
80                     <td>1</td>
81                     <td>El Señor de los Anillos</td>
82                     <td>J.R.R. Tolkien</td>
83                     <td>Fantasía</td>
84                     <td>978-0-261-10334-6</td>
85                   </tr>
86                   <tr>
87                     <td>2</td>
88                     <td>Harry Potter y la Piedra Filosofal</td>
89                     <td>J.K. Rowling</td>
90                     <td>Magia</td>
91                     <td>978-0-230-75333-9</td>
92                   </tr>
93                   <tr>
94                     <td>3</td>
95                     <td>1984</td>
96                     <td>George Orwell</td>
97                     <td>Distopía</td>
98                     <td>978-0-451-52633-1</td>
99                   </tr>
100                  <tr>
101                    <td>4</td>
102                    <td>El Quijote</td>
103                    <td>Miguel de Cervantes</td>
104                    <td>Clásica</td>
105                    <td>978-84-226-0916-1</td>
106                  </tr>
107                </tbody>
108              </table>
109            </div>
110          </div>
111        </div>
112      </div>
113    </body>
114  </html>

```

```

<div class="container-fluid">

  <!-- Collapsible navbar content -->
  <div class="collapse navbar-collapse" id="navbarContent">
    <ul class="navbar-nav ms-auto mb-2 mb-lg-0">
      {% if current_user.is_authenticated %}
      <li class="nav-item">
        <a class="nav-link" href="{{ url_for('main.dashboard') }}">
          Dashboard</a>
        </li>
      <!-- Listar usuarios solo para los administradores -->
      {% if current_user.role.name == 'Admin' %}
      <li class="nav-item">
        <a class="nav-link" href="{{ url_for('main.listar_usuarios') }}">
          Usuarios</a>
        </li>
      {% endif %}
      <li class="nav-item">
        <a class="nav-link" href="{{ url_for('main.cambiar_password') }}">
          Cambiar Contraseña</a>
        </li>
      <li class="nav-item">
        <a class="nav-link" href="{{ url_for('auth.logout') }}">Cerrar Sesión</a>
        </li>
      {% else %}
      <li class="nav-item">
        <a class="nav-link" href="{{ url_for('auth.login') }}">Iniciar Sesión</a>
        </li>
      <li class="nav-item">
        <a class="nav-link" href="{{ url_for('auth.register') }}">
          Registrarse</a>
        </li>
      </ul>
    </div>
  </div>

```

```

18 <div class="container-fluid">
35   <div class="collapse navbar-collapse" id="navbarContent">
66     </li>
67   </li>
68   {% endif %}
69 </ul>
70 </div>
71 </div>
72 </nav>
73
74 <!-- Main content -->
75 <div class="container mt-4">
76   {% with messages = get_flashed_messages() %}
77   {% if messages %}
78     {% for message in messages %}
79     <div class="alert alert-info">{{ message }}</div>
80     {% endfor %}
81   {% endif %}
82   {% endwith %}
83   {% block content %}{% endblock %}
84 </div>
85
86 <!-- Bootstrap JS bundle -->
87 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
88 </body>
89 </html>

```

Libros_form.html:

Esto es para que se puedan ver los libros

```
COMP 2052 > final_project > app > templates > dj Libros_form.html
1  {% extends "layout.html" %}
2
3  {% block content %}
4      <h2>Nuevo Libro</h2>
5      <form method="POST">
6          {{ form.hidden_tag() }}
7
8          <div class="mb-3">
9              {{ form.titulo.label(class="form-label") }}
10             {{ form.titulo(class="form-control") }}
11          </div>
12
13          <div class="mb-3">
14              {{ form.descripcion.label(class="form-label") }}
15              {{ form.descripcion(class="form-control") }}
16          </div>
17
18          <div class="mb-3">
19              {{ form.autor.label(class="form-label") }}
20              {{ form.autor(class="form-control") }}
21          </div>
22
23          <div class="mb-3">
24              {{ form.isbn.label(class="form-label") }}
25              {{ form.isbn(class="form-control") }}
26          </div>
27
28          <div class="mb-3">
29              {{ form.categoria.label(class="form-label") }}
30              {{ form.categoria(class="form-control") }}
31          </div>
32
33          <div class="mb-3">
34              {{ form.anio_publicacion.label(class="form-label") }}
35              {{ form.anio_publicacion(class="form-control") }}
36          </div>
37
```

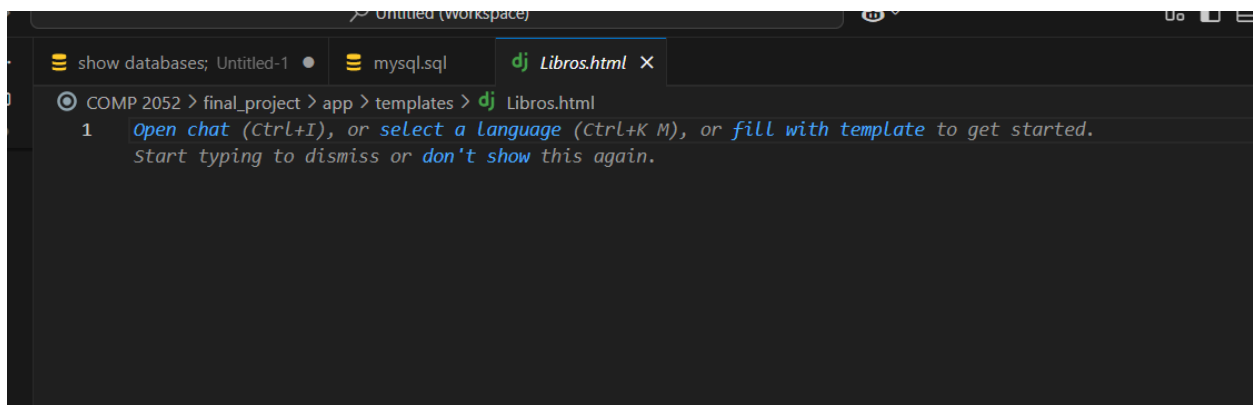
```

30 .....{{ form.categoria(class="form-control") }}
31 .....</div>
32 .....
33 .....<div class="mb-3">
34 .....{{ form.anio_publicacion.label(class="form-label") }}
35 .....{{ form.anio_publicacion(class="form-control") }}
36 .....</div>
37 .....
38 .....<div class="mb-3">
39 .....{{ form.estado.label(class="form-label") }}
40 .....{{ form.estado(class="form-control") }}
41 .....</div>
42 .....
43 .....<button type="submit" class="btn btn-success">Guardar</button>
44 .....</form>
45 {% endblock %}

```

Libros.html:

Se deja en blanco para que el usuario pueda almacenarlos



Login.html:

Pagina de login

```
COMP 2052 > final_project > app > templates > dj login.html
1  {% extends "layout.html" %}
2  {% block title %}Login{% endblock %}
3  {% block content %}
4  <form method="POST" novalidate>
5      <div class="d-flex justify-content-center align-items-center">
6          <div class="card col-11 col-sm-10 col-md-8 col-lg-6 mt-4">
7              <div class="card-header">
8                  <strong>Login de Usuario</strong>
9              </div>
10             <div class="card-body">
11                 {{ form.hidden_tag() }}
12
13                 <!-- Email field -->
14                 <div class="mb-3">
15                     {{ form.email.label(class="form-label") }}
16                     {{ form.email(class="form-control", placeholder="Enter your email") }}
17                 </div>
18
19                 <!-- Password field -->
20                 <div class="mb-3">
21                     {{ form.password.label(class="form-label") }}
22                     {{ form.password(class="form-control", placeholder="Enter your password") }}
23                 </div>
24             </div>
25
26             <!-- Submit button -->
27             <div class="card-footer">
28                 {{ form.submit(class="btn btn-primary") }}
29             </div>
30
31             <!-- Optional: Link to register -->
32             </div>
33         </div>
34         <p class="pt-2 text-center">
35             Don't have an account? <a href="{{ url_for('auth.register') }}">Register here</a>.
36         </p>
37     </form>
38 {% endblock %}
```

Register.html

Pagina para registrarte

```
show databases; Untitled-1 • mysql.sql dj register.html x
COMP 2052 > final_project > app > templates > dj register.html
1 {% extends "layout.html" %}
2 {% block title %}Register{% endblock %}
3 {% block content %}
4
5 <form method="POST" novalidate>
6   <div class="d-flex justify-content-center align-items-center">
7     <div class="card col-11 col-sm-10 col-md-8 col-lg-6 mt-3">
8       <div class="card-header">
9         <strong>User Registration</strong>
10      </div>
11      <div class="card-body">
12        {{ form.hidden_tag() }}
13
14        <div class="mb-3">
15          {{ form.username.label(class="form-label") }}
16          {{ form.username(class="form-control", placeholder="Enter your username") }}
17        </div>
18
19        <div class="mb-3">
20          {{ form.email.label(class="form-label") }}
21          {{ form.email(class="form-control", placeholder="Enter your email") }}
22        </div>
23
24        <div class="mb-3">
25          {{ form.password.label(class="form-label") }}
26          {{ form.password(class="form-control", placeholder="Enter your password") }}
27        </div>
28
29        <div class="mb-3">
30          {{ form.confirm_password.label(class="form-label") }}
31          {{ form.confirm_password(class="form-control", placeholder="Confirm your password") }}
32        </div>
33
34        <!-- New: Select Role -->
35        <div class="mb-3">
36          {{ form.role.label(class="form-label") }}
37          {{ form.role(class="form-select") }}
38        </div>
39
40        <div class="card-footer">
41          {{ form.submit(class="btn btn-success") }}
42        </div>
43      </div>
44    </div>
45    <p class="pt-2 text-center">
46      Already have an account?
47      <a href="{{ url_for('auth.login') }}">Log in here</a>.
48    </p>
49  </form>
50
51 {% endblock %}
```

Usuarios.html:


Pagina para ver los usuarios

```
show databases; Untitled-1  mysql.sql  usuarios.html X
COMP 2052 > final_project > app > templates > usuarios.html
2  content %}
3  <h3 class="mb-4">List of Registered Users</h3>
4
5  <table class="table table-striped table-hover">
6  ..<thead class="table-light">
7  ....<tr>
8  .....<th>Username</th>
9  .....<th>Email</th>
10 .....<th>Role</th>
11 ....</tr>
12 ..</thead>
13 ..<tbody>
14 ....{% for usuario in usuarios %}
15 ....<tr>
16 .....<td>{{ usuario.username }}</td>
17 .....<td>{{ usuario.email }}</td>
18 .....<td>{{ usuario.role.name }}</td>
19 ....</tr>
20 ....{% endfor %}
21 ..</tbody>
22 </table>
23 {% endblock %}
24
```

Esto esta adentro de app fuera del folder de html

Init.py:

Hace que corra

COMP 2052 > final_project > app >  __init__.py > ...

```
1 from flask import Flask
2 from flask_sqlalchemy import SQLAlchemy
3 from flask_login import LoginManager
4 from config import Config
5
6 db = SQLAlchemy()
7 login_manager = LoginManager()
8 login_manager.login_view = 'auth.login'
9
10 def create_app():
11     app = Flask(__name__)
12     app.config.from_object(Config)
13
14     db.init_app(app)
15     login_manager.init_app(app)
16
17     from app.routes import main
18     # from app.test_routes import main
19
20     from app.auth_routes import auth
21
22     app.register_blueprint(main)
23     app.register_blueprint(auth)
24
25     return app
26
```

Auth_routes.py:

Autentifica las rutas

```
show databases; Untitled-1 • mysql.sql auth_routes.py 2 X
COMP 2052 > final_project > app > auth_routes.py > logout
1 from flask import Blueprint, render_template, redirect, url_for, flash
2 from app.forms import LoginForm, RegisterForm
3 from app.models import db, User, Role
4 from flask_login import login_user, logout_user, login_required
5
6 auth = Blueprint('auth', __name__)
7
8 @auth.route('/login', methods=['GET', 'POST'])
9 def login():
10     form = LoginForm()
11     if form.validate_on_submit():
12         user = User.query.filter_by(email=form.email.data).first()
13         if user and user.check_password(form.password.data):
14             login_user(user)
15             return redirect(url_for('main.dashboard'))
16         flash('Credenciales inválidas', 'danger')
17     return render_template('login.html', form=form)
18
19 @auth.route('/register', methods=['GET', 'POST'])
20 def register():
21     form = RegisterForm()
22     if form.validate_on_submit():
23         # Buscar el rol seleccionado
24         role = Role.query.filter_by(name=form.role.data).first()
25         if not role:
26             flash('Rol inválido seleccionado', 'danger')
27             return redirect(url_for('auth.register'))
28
29         user = User(
30             username=form.username.data,
31             email=form.email.data,
32             role=role
33         )
34         user.set_password(form.password.data)
35         db.session.add(user)
36         db.session.commit()
37         flash('Usuario registrado exitosamente.', 'success')
```

```

20 def register():
21
22     if form.validate_on_submit():
23         # Buscar el rol seleccionado
24         role = Role.query.filter_by(name=form.role.data).first()
25         if not role:
26             flash('Rol inválido seleccionado', 'danger')
27             return redirect(url_for('auth.register'))
28
29         user = User(
30             username=form.username.data,
31             email=form.email.data,
32             role=role
33         )
34         user.set_password(form.password.data)
35         db.session.add(user)
36         db.session.commit()
37         flash('Usuario registrado exitosamente.', 'success')
38         return redirect(url_for('auth.login'))
39     return render_template('register.html', form=form)
40
41 @auth.route('/logout')
42 @login_required
43 def logout():
44     logout_user()
45     flash('Sesión cerrada correctamente.', 'info')
46     return redirect(url_for('auth.login'))

```

Forms.py:

Se usa para poder logear a los usuarios y botones

```

COMP 2052 > final_project > app > forms.py > ...
1 from flask_wtf import FlaskForm
2 from wtforms import StringField, PasswordField, SubmitField, TextAreaField, SelectField, IntegerField
3 from wtforms.validators import DataRequired, Email, EqualTo, Length, NumberRange
4
5 # Formulario para login de usuario
6 class LoginForm(FlaskForm):
7     ... email = StringField('Email', validators=[DataRequired(), Email()])
8     ... password = PasswordField('Password', validators=[DataRequired()])
9     ... submit = SubmitField('Login')
10
11 # Formulario para registrar un nuevo usuario
12 class RegisterForm(FlaskForm):
13     ... username = StringField('Username', validators=[DataRequired()])
14     ... email = StringField('Email', validators=[DataRequired(), Email()])
15     ... password = PasswordField('Password', validators=[DataRequired()])
16     ... confirm_password = PasswordField('Confirm password', validators=[DataRequired(), EqualTo('password')])
17     ...
18     ... role = SelectField(
19     ...     'Role',
20     ...     choices=[('Lector', 'Lector'), ('Bibliotecario', 'Bibliotecario'), ('Admin', 'Admin')],
21     ...     validators=[DataRequired()]
22     ... )
23     ... submit = SubmitField('Register')
24
25 # Formulario para cambiar la contraseña del usuario
26 class ChangePasswordForm(FlaskForm):
27     ... old_password = PasswordField('Current password', validators=[DataRequired()])
28     ... new_password = PasswordField('New password', validators=[DataRequired(), Length(min=6)])
29     ... confirm_password = PasswordField('Confirm new password', validators=[DataRequired(), EqualTo('new_password')])
30     ... submit = SubmitField('Update Password')
31
32 # Formulario para crear o editar un libro
33 class LibroForm(FlaskForm):
34     ... titulo = StringField('Título del libro', validators=[DataRequired(), Length(max=150)])
35     ... descripcion = TextAreaField('Descripción', validators=[DataRequired()])
36     ... autor = StringField('Autor', validators=[DataRequired(), Length(max=100)])
37     ... isbn = StringField('ISBN', validators=[DataRequired(), Length(max=20)])
38     ... categoria = StringField('Categoría', validators=[DataRequired(), Length(max=50)])
39     ... anio_publicacion = IntegerField('Año de publicación', validators=[DataRequired(), NumberRange(min=1000, max=2024)])
40     ... estado = SelectField(
41     ...     'Estado',
42     ...     choices=[('Disponible', 'Disponible'), ('Prestado', 'Prestado')],
43     ...     validators=[DataRequired()]
44     ... )
45     ... submit = SubmitField('Guardar')

```

Models.py:

Los modelos de la pagina (también se usa para los hash de las contraseñas

```

show databases; Untitled-1  mysql.sql  models.py 3 X
COMP 2052 > final_project > app > models.py > Libro
1  from app import db, login_manager
2  from flask_login import UserMixin
3  from werkzeug.security import generate_password_hash, check_password_hash
4  from sqlalchemy import Enum # Para el campo estado
5
6  # Carga el usuario desde su ID (para Flask-Login)
7  @login_manager.user_loader
8  def load_user(user_id):
9      return User.query.get(int(user_id))
10
11 # Modelo de rol (Admin, Bibliotecario, Lector)
12 class Role(db.Model):
13     __tablename__ = 'role'
14
15     id = db.Column(db.Integer, primary_key=True)
16     name = db.Column(db.String(64), unique=True, nullable=False)
17
18     # Relación con usuarios
19     users = db.relationship('User', backref='role', lazy=True)
20
21 # Modelo de usuario
22 class User(UserMixin, db.Model):
23     __tablename__ = 'user'
24
25     id = db.Column(db.Integer, primary_key=True)
26     username = db.Column(db.String(64), unique=True, nullable=False)
27     email = db.Column(db.String(120), unique=True, nullable=False)
28     password_hash = db.Column(db.String(256), nullable=False)
29     role_id = db.Column(db.Integer, db.ForeignKey('role.id'), nullable=False)
30
31     # Relación con libros que administra este usuario (si es bibliotecario)
32     libros = db.relationship('Libro', backref='bibliotecario', lazy=True)
33
34     def set_password(self, password: str):
35         """Genera el hash de la contraseña."""
36         self.password_hash = generate_password_hash(password)
37

```

```

def check_password(self, password: str) -> bool:
    """Verifica la contraseña contra el hash almacenado."""
    return check_password_hash(self.password_hash, password)

# Modelo de libros
class Libro(db.Model):
    __tablename__ = 'libro'

    id = db.Column(db.Integer, primary_key=True)
    titulo = db.Column(db.String(150), nullable=False)
    descripcion = db.Column(db.Text, nullable=True) # Campo añadido
    autor = db.Column(db.String(100), nullable=False)
    isbn = db.Column(db.String(20), unique=True, nullable=False)
    categoria = db.Column(db.String(50), nullable=False)
    anio_publicacion = db.Column(db.Integer, nullable=False)
    estado = db.Column(
        Enum('Disponible', 'Prestado', 'Dañado', name='estado_enum'),
        nullable=False,
        server_default='Disponible'
    )
    # Foreign key para relacionar libro con bibliotecario (usuario)
    bibliotecario_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=True)

```

Routes.py:

Las rutas que son necesarias para no confundirse

```
COMP 2052 > final_project > app > routes.py > listar_usuarios
1 from flask import Blueprint, render_template, redirect, url_for, request, flash
2 from flask_login import login_required, current_user
3 from app.forms import LibroForm, ChangePasswordForm
4 from app.models import db, Libro, User
5
6 main = Blueprint('main', __name__)
7
8 @main.route('/')
9 def index():
10     """
11     ... Página pública de inicio.
12     """
13     return render_template('index.html')
14
15 @main.route('/cambiar-password', methods=['GET', 'POST'])
16 @login_required
17 def cambiar_password():
18     """
19     ... Permite al usuario cambiar su contraseña.
20     """
21     form = ChangePasswordForm()
22
23     if form.validate_on_submit():
24         if not current_user.check_password(form.old_password.data):
25             flash('La contraseña actual es incorrecta.')
26             return render_template('cambiar_password.html', form=form)
27
28         current_user.set_password(form.new_password.data)
29         db.session.commit()
30         flash('✅ Contraseña actualizada correctamente.')
31         return redirect(url_for('main.dashboard'))
32
33     return render_template('cambiar_password.html', form=form)
34
```

COMP 2052 > final_project > app > routes.py > listar_usuarios

```
34
35 @main.route('/dashboard')
36 @login_required
37 def dashboard():
38     """
39     ...Panel principal del usuario. Muestra libros.
40     """
41     ...# Los lectores ven todos los libros, bibliotecarios solo los que agregaron, admins todos.
42     ...if current_user.role.name == 'Lector':
43     ..... libros = Libro.query.all()
44     ...elif current_user.role.name == 'Bibliotecario':
45     ..... libros = Libro.query.filter_by(bibliotecario_id=current_user.id).all()
46     ...else: # Admin
47     ..... libros = Libro.query.all()
48
49     ...return render_template('dashboard.html', libros=libros)
50
51 @main.route('/libros', methods=['GET', 'POST'])
52 @login_required
53 def libros():
54     """
55     ...Crear nuevo libro. Solo para Bibliotecarios o Admins.
56     """
57     ...if current_user.role.name not in ['Bibliotecario', 'Admin']:
58     ..... flash('No tienes permiso para crear libros.')
59     ..... return redirect(url_for('main.dashboard'))
60
61     ...form = LibroForm()
62     ...if form.validate_on_submit():
63     ..... libro = Libro(
64     .....     titulo=form.titulo.data,
65     .....     descripcion=form.descripcion.data,
66     .....     autor=form.autor.data,
67     .....     isbn=form.isbn.data,
68     .....     categoria=form.categoria.data,
69     .....     anio_publicacion=form.anio_publicacion.data,
70     .....     estado=form.estado.data,
```



```
53 def libros():
54     if form.validate_on_submit():
55         libro = Libro(
56             titulo=form.titulo.data,
57             descripcion=form.descripcion.data,
58             autor=form.autor.data,
59             isbn=form.isbn.data,
60             categoria=form.categoria.data,
61             anio_publicacion=form.anio_publicacion.data,
62             estado=form.estado.data,
63             bibliotecario_id=current_user.id
64         )
65         db.session.add(libro)
66         db.session.commit()
67         flash("Libro creado correctamente.")
68         return redirect(url_for('main.dashboard'))
69
70     return render_template('libros_form.html', form=form)
71
72 @main.route('/libros/<int:id>/editar', methods=['GET', 'POST'])
73 @login_required
74 def editar_libro(id):
75     """
76     Editar libro existente. Solo Admin o bibliotecario dueño.
77     """
78     libro = Libro.query.get_or_404(id)
79
80     if current_user.role.name not in ['Admin', 'Bibliotecario'] or (
81         libro.bibliotecario_id != current_user.id and current_user.role.name != 'Admin'):
82         flash('No tienes permiso para editar este libro.')
83         return redirect(url_for('main.dashboard'))
84
85     form = LibroForm(obj=libro)
86     if form.validate_on_submit():
87         libro.titulo = form.titulo.data
88         libro.descripcion = form.descripcion.data
89         libro.autor = form.autor.data
90         libro.isbn = form.isbn.data
```

COMP 2052 > final_project > app > routes.py > listar_usuarios

```
82 def editar_libro(id):
93     ... if form.validate_on_submit():
94         ... libro.titulo = form.titulo.data
95         ... libro.descripcion = form.descripcion.data
96         ... libro.autor = form.autor.data
97         ... libro.isbn = form.isbn.data
98         ... libro.categoria = form.categoria.data
99         ... libro.anio_publicacion = form.anio_publicacion.data
00         ... libro.estado = form.estado.data
01         ... db.session.commit()
02         ... flash("Libro actualizado correctamente.")
03         ... return redirect(url_for('main.dashboard'))
04
05     ... return render_template('libros_form.html', form=form, editar=True)
06
07
08 @main.route('/libros/<int:id>/eliminar', methods=['POST'])
09 @login_required
10 def eliminar_libro(id):
11     """
12     ... Eliminar libro. Solo Admin o bibliotecario dueño.
13     """
14     ... libro = Libro.query.get_or_404(id)
15
16     ... if current_user.role.name not in ['Admin', 'Bibliotecario'] or (
17     ...     libro.bibliotecario_id != current_user.id and current_user.role.name != 'Admin'):
18     ...     flash('No tienes permiso para eliminar este libro.')
19     ...     return redirect(url_for('main.dashboard'))
20
21     ... db.session.delete(libro)
22     ... db.session.commit()
23     ... flash("Libro eliminado correctamente.")
24     ... return redirect(url_for('main.dashboard'))
25
26 @main.route('/usuarios')
27 @login_required
28 def listar_usuarios():
29     """
```

```

@main.route('/usuarios')
@login_required
def listar_usuarios():
    """
    Lista usuarios solo para Admin.
    """
    if current_user.role.name != 'Admin':
        flash("No tienes permiso para ver esta página.")
        return redirect(url_for('main.dashboard'))

    usuarios = User.query.join(User.role).all()
    return render_template('usuarios.html', usuarios=usuarios)

```

Test_routes.py:

Esto es cuando corras las paginas en modo prueba

COMP 2052 > final_project > app > test_routes.py > ...

```
1 from flask import Blueprint, request, jsonify
2 from flask_login import login_required, current_user
3 from app.models import db, Libro
4
5 main = Blueprint('main', __name__)
6
7 @main.route('/')
8 @main.route('/dashboard')
9 def index():
10     """
11     ... Página pública simple.
12     ... """
13     ... return '<h1>API de Libros corriendo en modo prueba.</h1>'
14
15 @main.route('/libros', methods=['GET'])
16 @login_required
17 def listar_libros():
18     """
19     ... Retorna lista de libros (JSON). Visible para todos los roles.
20     ... """
21     ... libros = Libro.query.all()
22     ... data = [
23         {
24             ... 'id': libro.id,
25             ... 'titulo': libro.titulo,
26             ... 'descripcion': libro.descripcion,
27             ... 'autor': libro.autor,
28             ... 'bibliotecario_id': libro.bibliotecario_id
29         }
30         ... for libro in libros
31     ]
32     ... return jsonify(data), 200
33
34 @main.route('/libros/<int:id>', methods=['GET'])
35 @login_required
36 def obtener_libro(id):
37     """
38     ... Retorna un solo libro por ID.
```



COMP 2052 > final_project > app > test_routes.py > ...

```
5 @login_required
6 def obtener_libro(id):
7     """
8     ...Retorna un solo libro por ID.
9     """
10    libro = Libro.query.get_or_404(id)
11    data = {
12        ...'id': libro.id,
13        ...'titulo': libro.titulo,
14        ...'descripcion': libro.descripcion,
15        ...'autor': libro.autor,
16        ...'bibliotecario_id': libro.bibliotecario_id
17    }
18    return jsonify(data), 200
19
20 @main.route('/libros', methods=['POST'])
21 @login_required
22 def crear_libro():
23     """
24     ...Crea un libro (solo Bibliotecario o Admin).
25     """
26    if current_user.role.name not in ['Bibliotecario', 'Admin']:
27        return jsonify({'error': 'No tienes permiso para crear libros.'}), 403
28
29    data = request.get_json()
30    if not data:
31        return jsonify({'error': 'No se proporcionaron datos'}), 400
32
33    libro = Libro(
34        ...titulo=data.get('titulo'),
35        ...descripcion=data.get('descripcion'),
36        ...autor=data.get('autor'),
37        ...bibliotecario_id=current_user.id
38    )
39
40    db.session.add(libro)
41    db.session.commit()
42    return jsonify({'message': 'Libro creado', 'id': libro.id}), 201
```

```

70     db.session.add(libro)
71     db.session.commit()
72     return jsonify({'message': 'Libro creado', 'id': libro.id}), 201
73
74 @main.route('/libros/<int:id>', methods=['PUT'])
75 @login_required
76 def actualizar_libro(id):
77     """
78     Actualiza libro (solo Admin o bibliotecario dueño).
79     """
80     libro = Libro.query.get_or_404(id)
81
82     if current_user.role.name not in ['Admin', 'Bibliotecario'] or \
83         (libro.bibliotecario_id != current_user.id and current_user.role.name != 'Admin'):
84         return jsonify({'error': 'No tienes permiso para actualizar este libro.'}), 403
85
86     data = request.get_json()
87     libro.titulo = data.get('titulo', libro.titulo)
88     libro.descripcion = data.get('descripcion', libro.descripcion)
89     libro.autor = data.get('autor', libro.autor)
90
91     db.session.commit()
92     return jsonify({'message': 'Libro actualizado', 'id': libro.id}), 200
93
94 @main.route('/libros/<int:id>', methods=['DELETE'])
95 @login_required
96 def eliminar_libro(id):
97     """
98     Elimina libro (solo Admin o bibliotecario dueño).
99     """
100    libro = Libro.query.get_or_404(id)
101
102    if current_user.role.name not in ['Admin', 'Bibliotecario'] or \
103        (libro.bibliotecario_id != current_user.id and current_user.role.name != 'Admin'):
104        return jsonify({'error': 'No tienes permiso para eliminar este libro.'}), 403
105
106    db.session.delete(libro)

```

 Kivan M. López Raíces (1 day ago) Ln 1, Col 1 (3356 selected) Spaces: 4 UTF-8 CRLF {} Python  3.9.13 64

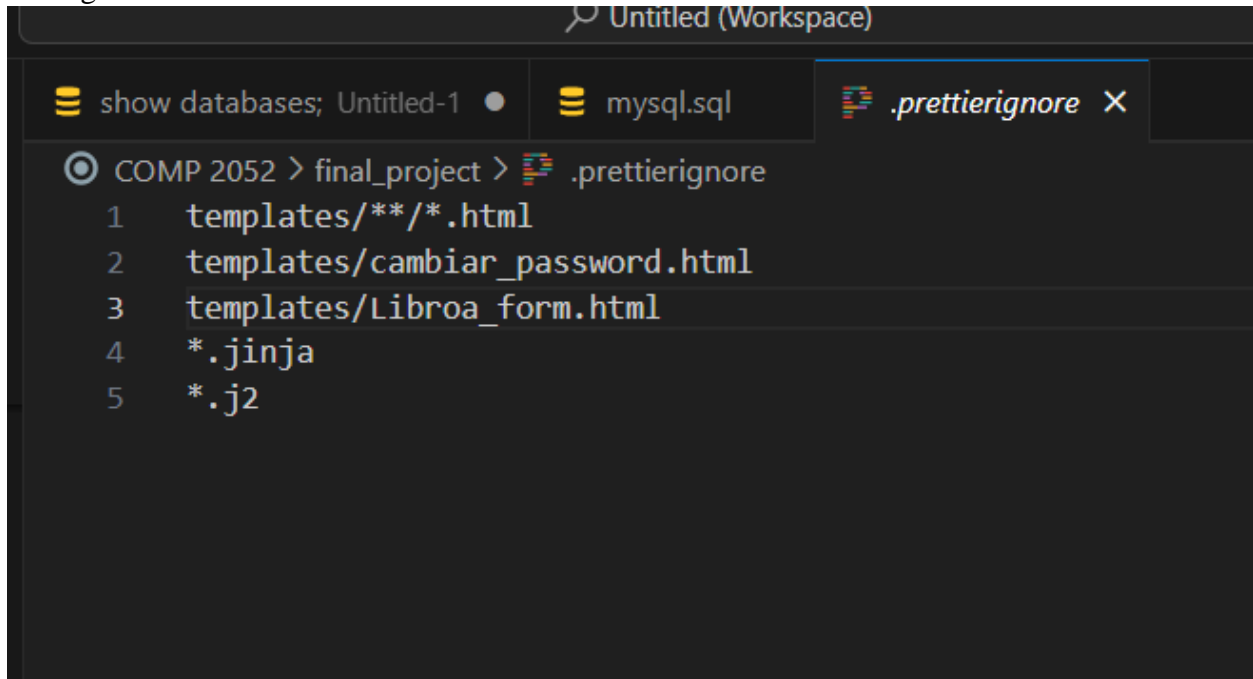
```

db.session.delete(libro)
db.session.commit()
return jsonify({'message': 'Libro eliminado', 'id': libro.id}), 200

```

Pretty.ignore:

Esto ignora ciertos html

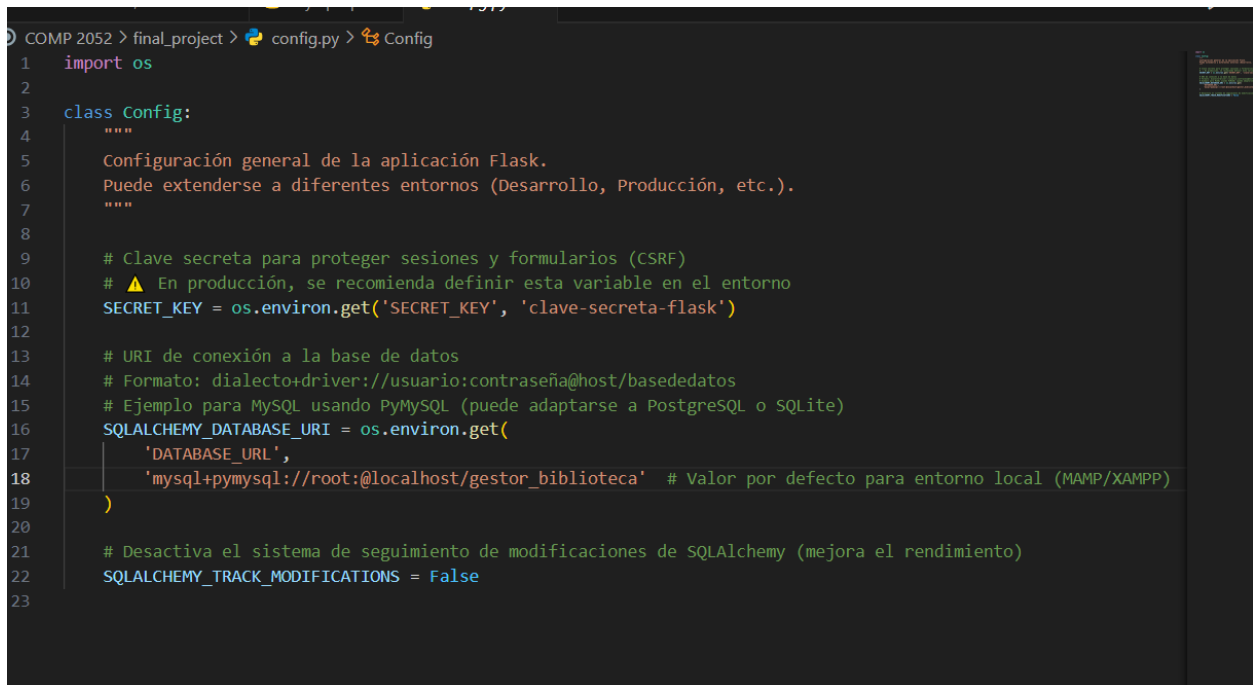


The screenshot shows a code editor window titled "Untitled (Workspace)". The file explorer on the left shows a project structure: "COMP 2052 > final_project > .prettierrignore". The main editor area displays the contents of the ".prettierrignore" file, which lists files to be ignored by Prettier:

```
1 templates/**/*.html
2 templates/cambiar_password.html
3 templates/Libroa_form.html
4 *.jinja
5 *.j2
```

Config.py:

Configuración con la base de datos



The screenshot shows a code editor window titled "COMP 2052 > final_project > config.py > Config". The main editor area displays the contents of the "Config.py" file, which defines a configuration class for a Flask application:

```
1 import os
2
3 class Config:
4     """
5     Configuración general de la aplicación Flask.
6     Puede extenderse a diferentes entornos (Desarrollo, Producción, etc.).
7     """
8
9     # Clave secreta para proteger sesiones y formularios (CSRF)
10    # ▲ En producción, se recomienda definir esta variable en el entorno
11    SECRET_KEY = os.environ.get('SECRET_KEY', 'clave-secreta-flask')
12
13    # URI de conexión a la base de datos
14    # Formato: dialecto+driver://usuario:contraseña@host/basededatos
15    # Ejemplo para MySQL usando PyMySQL (puede adaptarse a PostgreSQL o SQLite)
16    SQLALCHEMY_DATABASE_URI = os.environ.get(
17        'DATABASE_URL',
18        'mysql+pymysql://root:@localhost/gestor_biblioteca' # Valor por defecto para entorno local (MAMP/XAMPP)
19    )
20
21    # Desactiva el sistema de seguimiento de modificaciones de SQLAlchemy (mejora el rendimiento)
22    SQLALCHEMY_TRACK_MODIFICATIONS = False
23
```

Créate_demo_users:

No conseguimos que funcionara esto pero lo tenemos

```
COMP 2052 > final_project > create_demo_users.py > ...
1  from app import create_app, db
2  from app.models import Role, User
3
4  app = create_app()
5
6  with app.app_context():
7      # Roles a crear
8      roles = ['Admin', 'Bibliotecario', 'Lector']
9      for role_name in roles:
10         existing_role = Role.query.filter_by(name=role_name).first()
11         if not existing_role:
12             new_role = Role(name=role_name)
13             db.session.add(new_role)
14             print(f'✅ Rol "{role_name}" creado.')
15
16         db.session.commit()
17
18     # Usuarios iniciales
19     users_data = [
20         {
21             "username": "Administrador",
22             "email": "admin@example.com",
23             "password": "admin123",
24             "role_name": "Admin"
25         },
26         {
27             "username": "Carlos Bibliotecario",
28             "email": "bibliotecario@example.com",
29             "password": "bibliotecario123",
30             "role_name": "Bibliotecario"
31         },
32         {
33             "username": "Ana Lector",
34             "email": "lector@example.com",
35             "password": "lector123",
36             "role_name": "Lector"
37         }
38     ]
```



```

    }
]

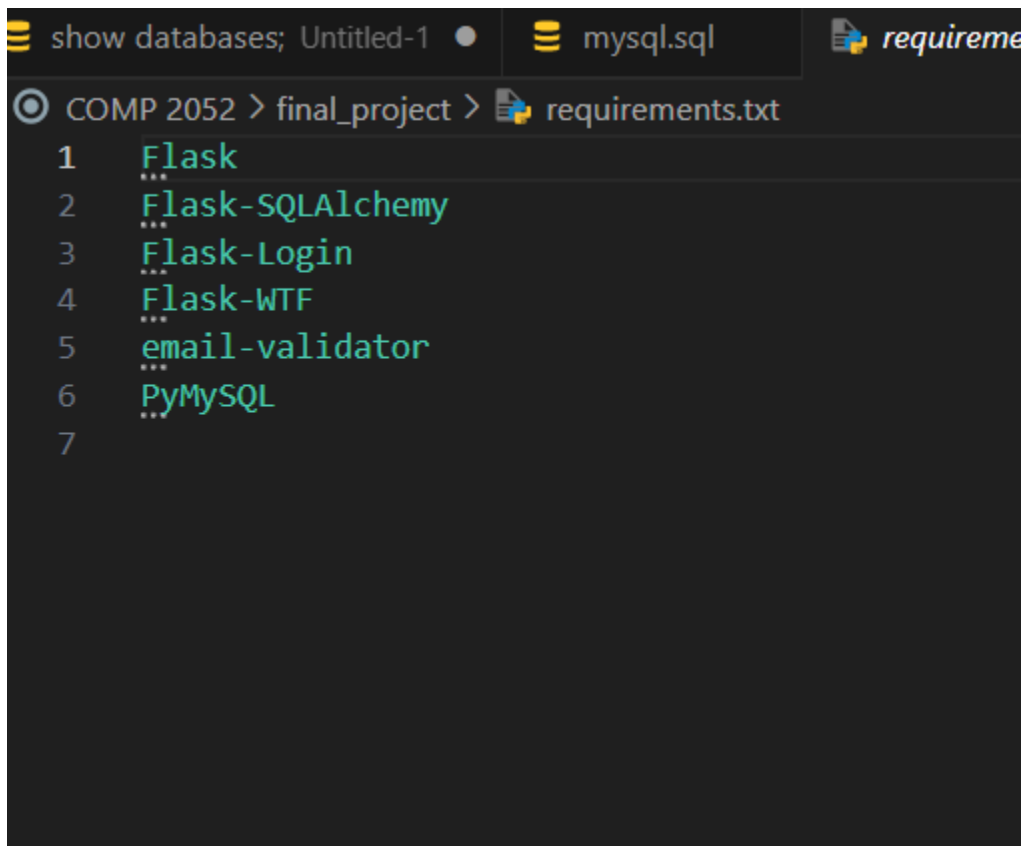
for user_info in users_data:
    existing_user = User.query.filter_by(email=user_info['email']).first()
    if not existing_user:
        role = Role.query.filter_by(name=user_info['role_name']).first()
        user = User(
            username=user_info['username'],
            email=user_info['email'],
            role=role
        )
        user.set_password(user_info['password'])
        db.session.add(user)
        print(f'✅ Usuario "{user.username}" creado con rol "{role.name}"')
    else:
        print(f'❌ El usuario con email {user_info["email"]} ya existe.')

db.session.commit()
print("✅ Todos los usuarios fueron procesados correctamente.")

```

Requirements.txt

Lo utilizamos para instalar todo lo necesario



The screenshot shows a code editor with a dark theme. At the top, there are tabs for 'show databases; Untitled-1', 'mysql.sql', and 'requirements.txt'. The active tab is 'requirements.txt', which is located at 'COMP 2052 > final_project > requirements.txt'. The file contains a list of Python dependencies, each on a new line, numbered 1 through 7:

```

1 Flask
2 Flask-SQLAlchemy
3 Flask-Login
4 Flask-WTF
5 email-validator
6 PyMySQL
7

```

Run.py:

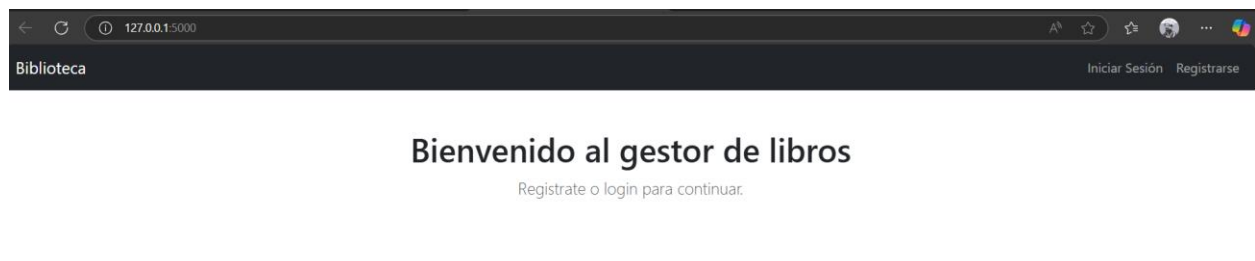
Hace que todo corra

```
COMP 2052 > final_project > run.py > ...
1  from app import create_app
2
3  # Crea la instancia de la aplicación Flask utilizando la factoría
4  app = create_app()
5
6  # Punto de entrada de la aplicación
7  if __name__ == '__main__':
8      # Ejecuta el servidor Flask en modo desarrollo
9      # host='0.0.0.0' permite que sea accesible desde otras máquinas en la red local
10     # En producción, desactiva debug o usa un servidor como Gunicorn
11     app.run(debug=True, host='0.0.0.0', port=5000)
12
```

Pantallas o Interfaces:

Pagina Principal:

El usuario de registra o log in para continuar



Pagina de Login:

Los usuarios pueden acceder con su password y email

← 127.0.0.1:5000/login

Biblioteca [Iniciar Sesión](#) [Registrarse](#)

Login de Usuario

Email

Enter your email

Password

Enter your password

Login

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

Pagina para registrarse:

El usuario puede crear una cuenta

← 127.0.0.1:5000/register

Biblioteca [Iniciar Sesión](#) [Registrarse](#)

User Registration

Username

Enter your username

Email

Enter your email

Password

Enter your password

Confirm password

Confirm your password

Role

Lector

Register

Already have an account? [Log in here.](#)

Interface como bibliotecario:

El bibliotecario puede editar borrar o añadir libros pero no puede ver a los usuarios

← 127.0.0.1:5000/dashboard

Biblioteca [Dashboard](#) [Cambiar Contraseña](#) [Cerrar Sesión](#)

Gestión de Libros [+ Nuevo Libro](#)

Título	Autor	ISBN	Categoría	Año Publicación	Estado	Acciones
No tienes descuento ultimate edition	Krystal S. Perez Rosado	58293769827	guia	1987	Disponible	✎ ✖
comp 2052 - guia oficial de john doe	Krystal S. Perez Rosado	4375265372	guia	1999	Disponible	✎ ✖

Interface de Nuevo Libro:

Aquí el bibliotecario puede crear su libro

The screenshot shows a web browser window with the address bar displaying "127.0.0.1:5000/libros". The page title is "Biblioteca". The navigation bar includes links for "Dashboard", "Cambiar Contraseña", and "Cerrar Sesión". The main content area is titled "Nuevo Libro" and contains a form with the following fields:

- Título del libro
- Descripción
- Autor
- ISBN
- Categoría
- Año de publicación
- Estado (with a dropdown menu showing "Disponible")

Below the form, there is a green "Guardar" button. The Windows taskbar at the bottom shows the system clock as 1:37 PM on 5/20/2025, with a temperature of 31°C and weather conditions of Mayorm. solea...

Interface de cambiar contraseña:

Los usuarios pueden cambiar su contraseña

Browser tabs: Login, comp2052/final_project/databases, Change Password

Address bar: 127.0.0.1:5000/cambiar-password

Biblioteca Dashboard Cambiar Contraseña Cerrar Sesión

Change Password

Current password

New password

Confirm new password

[Update Password](#)

Interface como admin:

El admin puede ver lo mismo pero tiene acceso a los usuarios

Browser tabs: Login, comp2052/final_project/databases, User List

Address bar: 127.0.0.1:5000/dashboard

Biblioteca Dashboard Usuarios Cambiar Contraseña Cerrar Sesión

Gestión de Libros

[+ Nuevo Libro](#)

Título	Autor	ISBN	Categoría	Año Publicación	Estado	Acciones
No tienes descuento ultimate edition	Krystal S. Perez Rosado	58293769827	guia	1987	Disponible	✎ ✖
comp 2052 - guia oficial de john doe	Krystal S. Perez Rosado	4375265372	guia	1999	Disponible	✎ ✖

Pagina de usuarios:

Interfaz en la que el admin puede ver los usuarios

Browser tabs: Login, comp2052/final_project/databases, User List

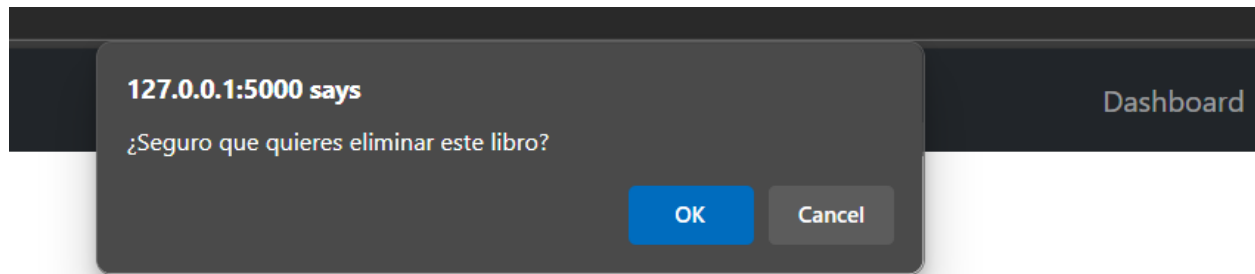
Address bar: 127.0.0.1:5000/usuarios

Biblioteca Dashboard Usuarios Cambiar Contraseña Cerrar Sesión

List of Registered Users

Username	Email	Role
kivan	bookdiscount@test.com	Lector
Krystal	notienesdescuento@gmail.com	Bibliotecario
jjin	Jin123@gmail.com	Bibliotecario
Kaivan	ejemploadmin@gmail.com	Admin

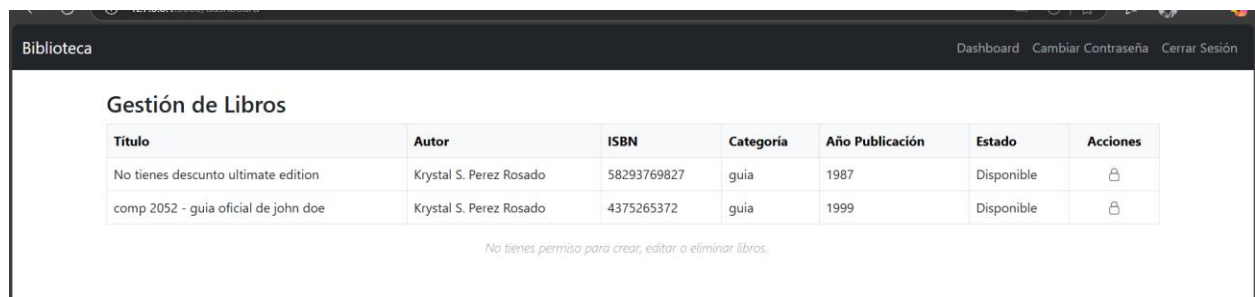
Pop up de cuando vas a borrar un libro:



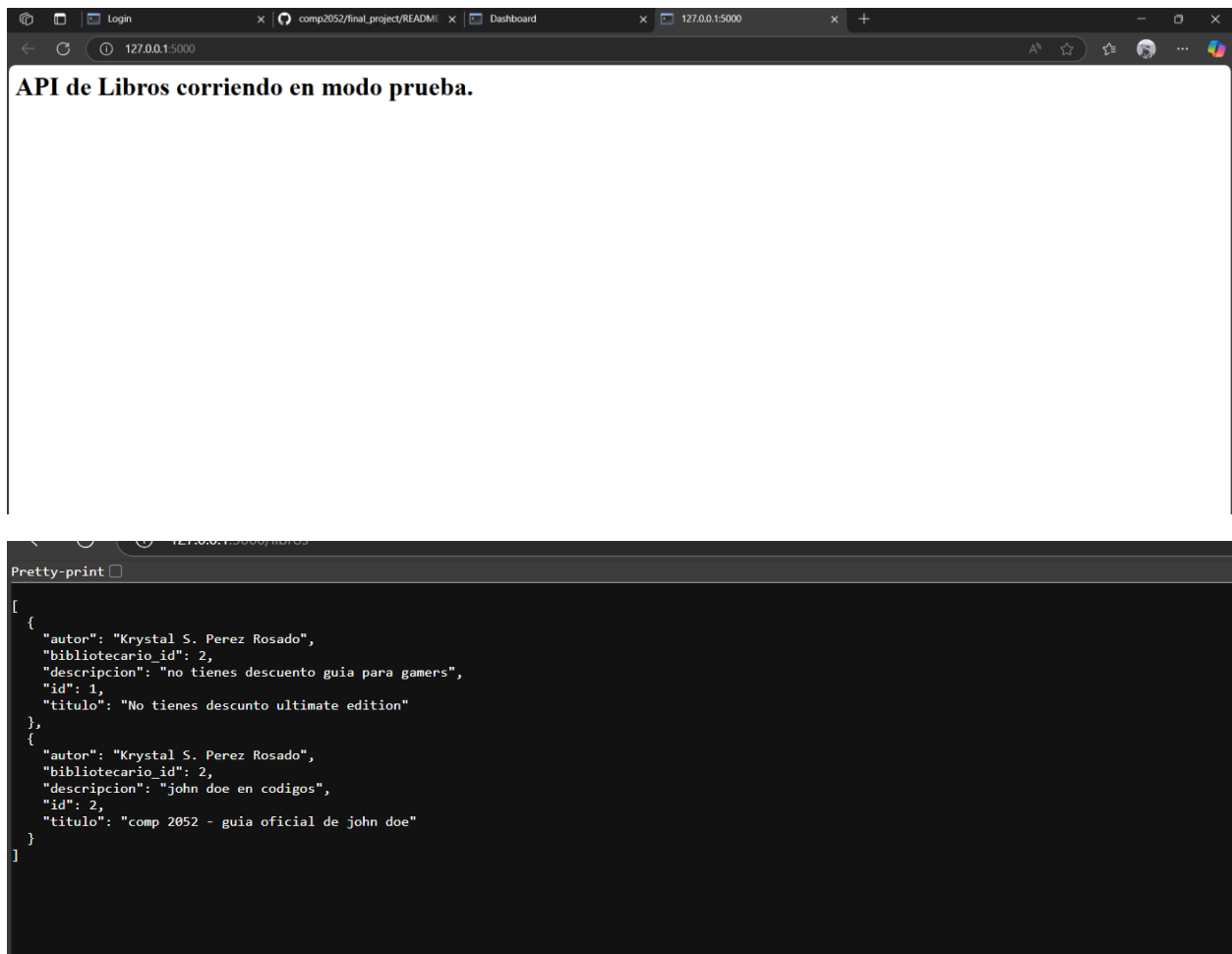
Autor	ISBN	Categoría	Año Publicación
Krystal S. Perez Rosado	58293769827	guia	1987

Interfaz de lector:

Esto es lo que va a ver un lector



Paginas de test:



Folder de Pruebas Create.Rest

Tratamos de correrlos pero no funcionaron


Créate.rest:

```
show databases; Untitled-1 • mysql.sql __init__.py create.rest X sqldelebreria.SQL
COMP 2052 > final_project > pruebas > create.rest > POST /libros
Send Request
1 POST http://localhost:5000/libros
2 Content-Type: application/json
3
4 {
5   "titulo": "Flask Avanzado",
6   "descripcion": "libro intensivo de desarrollo con Flask",
7   "autor": "Tu Nombre",
8   "isbn": "1234567890",
9   "categoria": "Desarrollo",
10  "anio_publicacion": 2025,
11  "estado": "Disponible"
12 }
13 ###
Send Request
14 POST http://localhost:5000/login
15 Content-Type: application/x-www-form-urlencoded
16
17 email=notienesdescuento@gmail.com&password=test2
```


Delete.rest:

```
show databases; Untitled-1 • mysql.sql __init__.py delete.rest M X
COMP 2052 > final_project > pruebas > delete.rest > ...
1 ### Eliminar libro (POST)
2
3 Send Request
4 DELETE http://localhost:5000/libros/2
```

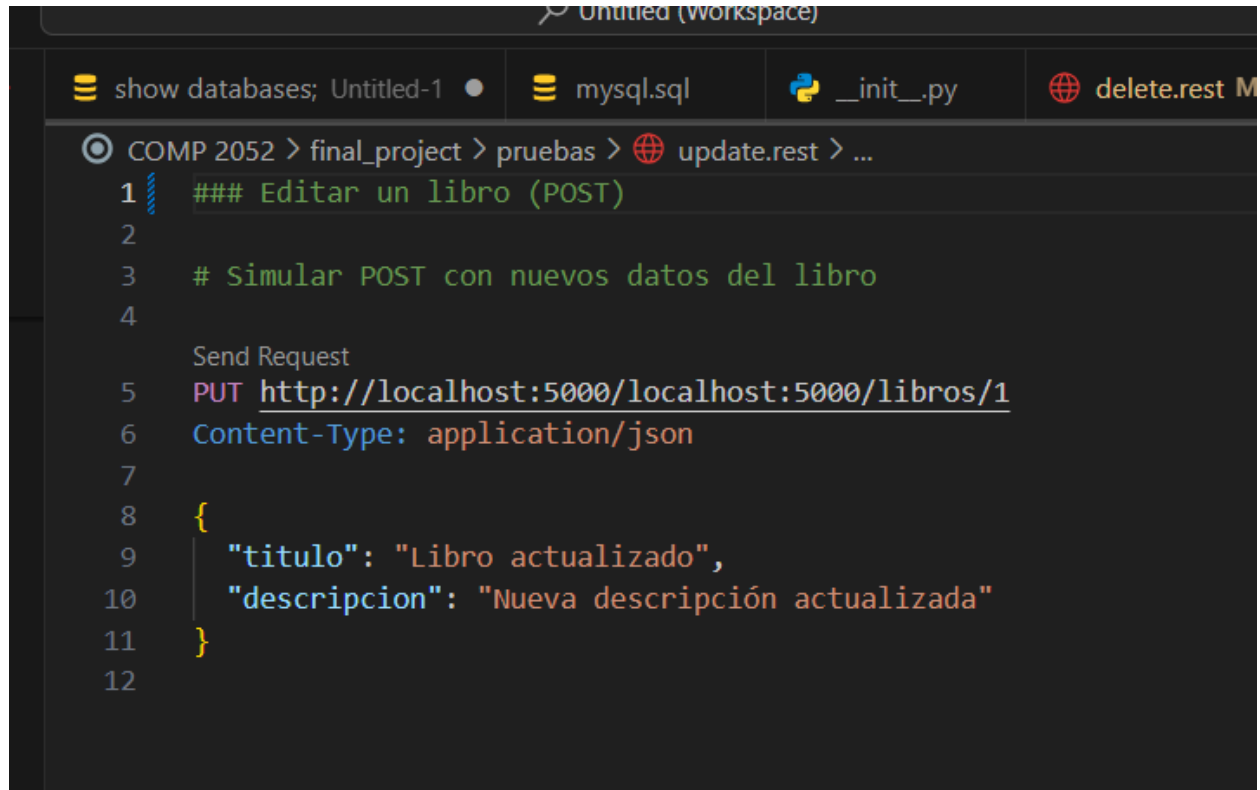
Read-a-row.rest


```
COMP 2052 > final_project > pruebas >  read-a-row.rest > ...
1  ### Obtener libro por ID (GET)
2
   Send Request
3  GET http://localhost:5000/libros/1
4  Content-Type: application/json
5
```

Read.rest:

```
COMP 2052 > final_project > pruebas >  read.rest > ...
1  ### Obtener todos los libros (GET)
2
   Send Request
3  GET http://localhost:5000/libros
4  Content-Type: application/json
5
```

Update.rest



The screenshot shows a REST client interface with a workspace titled 'Untitled (workspace)'. The top bar contains several tabs: 'show databases; Untitled-1', 'mysql.sql', '__init__.py', and 'delete.rest M'. The main area displays a REST client configuration for a PUT request. The breadcrumb path is 'COMP 2052 > final_project > pruebas > update.rest > ...'. The request is defined as follows:

```
1  ### Editar un libro (POST)
2
3  # Simular POST con nuevos datos del libro
4
5  Send Request
6  PUT http://localhost:5000/localhost:5000/libros/1
7  Content-Type: application/json
8  {
9      "titulo": "Libro actualizado",
10     "descripcion": "Nueva descripción actualizada"
11 }
12
```