APP.PY  
  
from flask import Flask, jsonify, request, render\_template, redirect, url\_for, flash, session, make\_response

from flask\_sqlalchemy import SQLAlchemy

from flask\_bcrypt import Bcrypt

from flask\_jwt\_extended import JWTManager, create\_access\_token, jwt\_required, get\_jwt\_identity, set\_access\_cookies

from flask\_migrate import Migrate

from flask\_mail import Mail, Message

from dotenv import load\_dotenv

from flask import Flask

from flask\_wtf import CSRFProtect, FlaskForm

from wtforms import StringField, SubmitField, PasswordField

from wtforms.validators import DataRequired, Email

app = Flask(\_\_name\_\_)

app.config['SECRET\_KEY'] = '123456'  # Make sure to set a strong secret key.

csrf = CSRFProtect(app)

import os

# Load environment variables

load\_dotenv()

# Force HTTP

if os.getenv('FORCE\_HTTP'):

    app.config['PREFERRED\_URL\_SCHEME'] = 'http'

# Secret and configurations

app.secret\_key = os.getenv('FLASK\_SECRET\_KEY', 'super\_secret\_key')  # Better to get it from environment

app.config['SQLALCHEMY\_DATABASE\_URI'] = f'mysql://{os.getenv("MYSQL\_USER")}:' \

                                        f'{os.getenv("MYSQL\_PASSWORD")}@{os.getenv("MYSQL\_HOST")}/' \

                                        f'{os.getenv("MYSQL\_DB")}'

app.config['SQLALCHEMY\_TRACK\_MODIFICATIONS'] = False

app.config['MAIL\_SERVER'] = 'smtp.example.com'

app.config['MAIL\_PORT'] = 465

app.config['MAIL\_USERNAME'] = os.getenv('MAIL\_USERNAME')

app.config['MAIL\_PASSWORD'] = os.getenv('MAIL\_PASSWORD')

app.config['MAIL\_USE\_TLS'] = False

app.config['MAIL\_USE\_SSL'] = True

app.config['JWT\_SECRET\_KEY'] = os.getenv('JWT\_SECRET\_KEY', 'another\_super\_secret\_key')

# JWT Configuration to use cookies

app.config['JWT\_TOKEN\_LOCATION'] = ['cookies']  # Enable JWT to use cookies

app.config['JWT\_ACCESS\_COOKIE\_NAME'] = 'access\_token\_cookie'  # Name for the access token cookie

app.config['JWT\_ACCESS\_COOKIE\_PATH'] = '/'  # Path where the access token cookie is accessible

app.config['JWT\_REFRESH\_COOKIE\_NAME'] = 'refresh\_token\_cookie'  # Name for the refresh token cookie

app.config['JWT\_REFRESH\_COOKIE\_PATH'] = '/'  # Path where the refresh token cookie is accessible

app.config['JWT\_COOKIE\_CSRF\_PROTECT'] = True  # Enable CSRF protection for cookies

app.config['JWT\_COOKIE\_SECURE'] = False  # Only set to True if using HTTPS. Set to False for local testing

# Initialize JWTManager with the app

jwt = JWTManager(app)

# Initialize extensions

db = SQLAlchemy(app)

bcrypt = Bcrypt(app)

jwt = JWTManager(app)

migrate = Migrate(app, db)

mail = Mail(app)

# Models

class User(db.Model):

    id = db.Column(db.Integer, primary\_key=True)

    first\_name = db.Column(db.String(100), nullable=False)

    last\_name = db.Column(db.String(100), nullable=False)

    email = db.Column(db.String(100), unique=True, nullable=False)

    password\_hash = db.Column(db.String(128))

    is\_active = db.Column(db.Boolean, default=True, nullable=False)

class Textbook(db.Model):

    id = db.Column(db.Integer, primary\_key=True)

    title = db.Column(db.String(255), nullable=False)

    author = db.Column(db.String(255), nullable=False)

    isbn = db.Column(db.String(20), nullable=True)

class Transaction(db.Model):

    id = db.Column(db.Integer, primary\_key=True)

    user\_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)

    textbook\_id = db.Column(db.Integer, db.ForeignKey('textbook.id'), nullable=False)

    quantity = db.Column(db.Integer, nullable=False)

class UpdateUserForm(FlaskForm):

    first\_name = StringField('First Name', validators=[DataRequired()])

    last\_name = StringField('Last Name', validators=[DataRequired()])

    email = StringField('Email', validators=[DataRequired()])

    submit = SubmitField('Update')

class LoginForm(FlaskForm):

    email = StringField('Email', validators=[DataRequired(), Email()])

    password = PasswordField('Password', validators=[DataRequired()])

    submit = SubmitField('Login')

# Routes

@app.route('/')

def home():

    return redirect(url\_for('login'))

@app.route('/register', methods=['GET', 'POST'])

def register():

    if request.method == 'POST':

        data = request.form

        if not data.get('email') or not data.get('password'):

            flash('Email and password are required', 'error')

            return render\_template('register.html')  # Render the template again with flash message

        existing\_user = User.query.filter\_by(email=data['email']).first()

        if existing\_user:

            flash('User with this email already exists', 'error')

            return render\_template('register.html')  # Render the template again with flash message

        hashed\_password = bcrypt.generate\_password\_hash(data['password']).decode('utf-8')

        new\_user = User(first\_name=data.get('first\_name'), last\_name=data.get('last\_name'), email=data['email'], password\_hash=hashed\_password)

        db.session.add(new\_user)

        db.session.commit()

        flash('User registered successfully', 'success')

    return render\_template('register.html')

@app.route('/login', methods=['GET', 'POST'])

def login():

    form = LoginForm()

    if form.validate\_on\_submit():

        user = User.query.filter\_by(email=form.email.data).first()

        if user and bcrypt.check\_password\_hash(user.password\_hash, form.password.data):

            access\_token = create\_access\_token(identity=user.email)

            response = make\_response(redirect(url\_for('dashboard')))

            set\_access\_cookies(response, access\_token)

            return response

        else:

            flash('Invalid credentials', 'error')

            return render\_template('login.html', form=form)

    return render\_template('login.html', form=form)

@app.route('/dashboard')

@jwt\_required()

def dashboard():

    current\_user\_email = get\_jwt\_identity()

    user = User.query.filter\_by(email=current\_user\_email).first()

    if not user:

        flash("User not found", 'error')

        return redirect(url\_for('login'))

    return render\_template('dashboard.html', current\_user=user)

@app.route('/textbooks/search', methods=['GET'])

def search\_textbooks():

    # Add actual search logic and template

    return render\_template('search\_textbooks.html')

from flask import request, redirect, url\_for, render\_template, flash

@app.route('/update\_user/<int:user\_id>', methods=['GET', 'POST'])

def update\_user(user\_id):

    user = User.query.get\_or\_404(user\_id)

    form = UpdateUserForm(obj=user)  # Initialize the form and pre-fill with user data

    if request.method == 'POST':

        print("CSRF Token from form:", form.csrf\_token.data)  # Log CSRF token received

    if form.validate\_on\_submit():  # Check if the form submission is valid

        user.first\_name = form.first\_name.data

        user.last\_name = form.last\_name.data

        user.email = form.email.data

        try:

            db.session.commit()  # Commit changes to the database

            flash('Your information has been updated successfully!', 'success')

            return redirect(url\_for('dashboard'))  # Redirect to the dashboard after a successful update

        except Exception as e:

            db.session.rollback()  # Roll back the transaction in case of an error

            flash('Error updating user: ' + str(e), 'danger')  # Display an error message

            print("Error updating user:", e)  # Log the error for debugging

    # Log and display form errors

    if form.errors:

        print("Form errors:", form.errors)

        for error\_field, error\_messages in form.errors.items():

            for error in error\_messages:

                flash(f"Error in {error\_field}: {error}", 'danger')

    return render\_template('update\_user.html', form=form, current\_user=user)

@app.route('/recover\_password', methods=['POST'])

def recover\_password():

    data = request.get\_json()

    user = User.query.filter\_by(email=data['email']).first()

    if not user:

        return jsonify({'message': 'User not found'}), 404

    token = create\_access\_token(identity=user.email, expires\_delta=False)

    msg = Message("Password Reset", sender='noreply@example.com', recipients=[user.email])

    msg.body = f"Please use the following link to reset your password: {url\_for('reset\_password', token=token, \_external=True)}"

    mail.send(msg)

    return jsonify({'message': 'Check your email for password reset instructions'}), 200

@app.route('/purchase\_textbooks')

def purchase\_textbooks():

    # Add logic here if needed to fetch or process textbook purchasing data

    return render\_template('purchase\_textbooks.html')

@app.route('/find\_roommate')

def find\_roommate():

    # Add logic here to handle roommate finding functionality

    return render\_template('find\_roommate.html')

@app.route('/purchase\_meal\_plan')

def purchase\_meal\_plan():

    # Add logic here to handle meal plan purchase functionality

    return render\_template('purchase\_meal\_plan.html')

@app.route('/purchase\_bus\_ticket')

def purchase\_bus\_ticket():

    # Add logic here to handle bus ticket purchase functionality

    return render\_template('purchase\_bus\_ticket.html')

@app.route('/view\_sports\_activities')

def view\_sports\_activities():

    # Add logic here for handling the viewing of sports activities

    return render\_template('view\_sports\_activities.html')

@app.route('/run\_election\_poll')

def run\_election\_poll():

    # Add logic here to handle the functionality of running an election poll

    return render\_template('run\_election\_poll.html')

@app.route('/people/search', methods=['GET'])

@jwt\_required()

def search\_people():

    # Extract search parameters from request query string

    department = request.args.get('department')

    first\_name = request.args.get('first\_name')

    last\_name = request.args.get('last\_name')

    # Build search query considering all or any of the provided criteria

    query = User.query

    if department:

        query = query.filter\_by(department=department)

    if first\_name and last\_name:

        query = query.filter\_by(first\_name=first\_name, last\_name=last\_name)

    elif first\_name:

        query = query.filter\_by(first\_name=first\_name)

    elif last\_name:

        query = query.filter\_by(last\_name=last\_name)

    # Execute search, handle empty results

    results = query.all()

    if not results:

        return render\_template('search\_results\_not\_found.html'), 404

    # Render search results template with user information

    return render\_template('search\_people\_results.html', users=results)

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Login</title>

  <link rel="stylesheet" href="static/style.css">

</head>

<body>

  <h1>Login</h1>

  <form action="{{ url\_for('login') }}" method="POST">

    <input type="hidden" name="csrf\_token" value="{{ csrf\_token() }}"> <!-- Add CSRF token here -->

    <label for="email">Email:</label>

    <input type="email" id="email" name="email" required><br>

    <label for="password">Password:</label>

    <input type="password" id="password" name="password" required><br>

    <button type="submit">Login</button>

  </form>

  <p>Don't have an account? <a href="{{ url\_for('register') }}">Register</a></p>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Dashboard</title>

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

    <style>

        body {

          font-family: Arial, sans-serif;

          margin: 40px;

          padding: 20px;

          background-color: #f4f4f4;

        }

        form {

          background: #ffffff;

          padding: 20px;

          border-radius: 10px;

          box-shadow: 0 0 20px rgba(0, 0, 0, 0.1);

        }

        input[type=email], input[type=password] {

          margin: 10px 0;

          width: 100%;

          padding: 15px;

          border: 1px solid #ccc;

          border-radius: 5px;

          background-color: #e0e0e0;

        }

        input[type=submit] {

          padding: 15px 20px;

          background-color: #007bff;

          color: #ffffff;

          border: none;

          border-radius: 5px;

          cursor: pointer;

          transition: background-color 0.3s ease;

        }

        input[type=submit]:hover {

          background-color: #0056b3;

        }

        h1, h2, h3, h4, h5, h6 {

          color: #0056b3;

        }

        p {

          color: #546e7a;

        }

        .list-group-item {

          background-color: #e7e7eb; /\* Light grey background for list items \*/

          color: #333; /\* Dark text for readability \*/

          border-color: #dee2e6; /\* Grey border \*/

        }

        .list-group-item:hover, .list-group-item:focus {

          background-color: #cfd2d6; /\* Darker grey background on hover/focus \*/

          color: #0056b3; /\* Blue text on hover/focus \*/

        }

    </style>

</head>

<body>

<div class="container mt-5">

    <h1>Welcome to Your Dashboard, {{ current\_user.first\_name }}</h1>

    <div class="list-group">

        <a href="{{ url\_for('update\_user', user\_id=current\_user.id) }}" class="list-group-item list-group-item-action">Update Information</a>

        <a href="{{ url\_for('search\_textbooks') }}" class="list-group-item list-group-item-action">Search Textbooks</a>

        <a href="{{ url\_for('search\_people') }}" class="list-group-item list-group-item-action">Search People</a>

        <a href="{{ url\_for('purchase\_textbooks') }}" class="list-group-item list-group-item-action">Purchase Textbooks</a>

        <a href="{{ url\_for('find\_roommate') }}" class="list-group-item list-group-item-action">Find Roommate</a>

        <a href="{{ url\_for('purchase\_meal\_plan') }}" class="list-group-item list-group-item-action">Purchase Meal Plan</a>

        <a href="{{ url\_for('purchase\_bus\_ticket') }}" class="list-group-item list-group-item-action">Purchase Bus Ticket</a>

        <a href="{{ url\_for('view\_sports\_activities') }}" class="list-group-item list-group-item-action">View Sports Activities</a>

        <a href="{{ url\_for('run\_election\_poll') }}" class="list-group-item list-group-item-action">Run Election Poll</a>

    </div>

</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/js/bootstrap.min.js"></script>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Register</title>

  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

  <link rel="stylesheet" href="static/style.css">

  <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/js/bootstrap.min.js"></script>

</head>

<body>

  <!-- Home Button at the top left -->

  <a href="{{ url\_for('login') }}" class="btn btn-primary mb-3">Home</a>

  <h1>Register</h1>

  <form action="{{ url\_for('register') }}" method="POST">

    <label for="first\_name">First Name:</label>

    <input type="text" id="first\_name" name="first\_name" required><br>

    <label for="last\_name">Last Name:</label>

    <input type="text" id="last\_name" name="last\_name" required><br>

    <label for="email">Email:</label>

    <input type="email" id="email" name="email" required><br>

    <label for="password">Password:</label>

    <input type="password" id="password" name="password" required><br>

    <button type="submit">Register</button>

  </form>

  <!-- Modal for Flash Messages -->

  {% with messages = get\_flashed\_messages(with\_categories=true) %}

  {% if messages %}

  <div class="modal fade" id="messageModal" tabindex="-1" role="dialog" aria-labelledby="messageModalLabel" aria-hidden="true">

    <div class="modal-dialog" role="document">

      <div class="modal-content">

        <div class="modal-header">

          <h5 class="modal-title" id="messageModalLabel">Registration Message</h5>

          <button type="button" class="close" data-dismiss="modal" aria-label="Close">

            <span aria-hidden="true">&times;</span>

          </button>

        </div>

        <div class="modal-body">

          {% for category, message in messages %}

          <div class="alert alert-{{ category }}">{{ message }}</div>

          {% endfor %}

        </div>

      </div>

    </div>

  </div>

  <script>

    $(document).ready(function() {

      $('#messageModal').modal('show');

    });

  </script>

  {% endif %}

  {% endwith %}

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Update Information</title>

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container mt-5">

    <h2>Update Your Information</h2>

    <form action="{{ url\_for('update\_user', user\_id=current\_user.id) }}" method="post">

        {{ form.hidden\_tag() }}

        <div class="form-group">

            <label for="first\_name">First Name:</label>

            {{ form.first\_name(class="form-control", id="first\_name") }}

        </div>

        <div class="form-group">

            <label for="last\_name">Last Name:</label>

            {{ form.last\_name(class="form-control", id="last\_name") }}

        </div>

        <div class="form-group">

            <label for="email">Email:</label>

            {{ form.email(class="form-control", id="email") }}

        </div>

        <button type="submit" class="btn btn-primary">Update</button>

    </form>

</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/js/bootstrap.min.js"></script>

</body>

</html>