Movie Ticket Booking System

A full-stack web application for booking movie tickets, built with Python, Flask, and SQLite3. This project features a complete user-facing interface for booking tickets and a secure admin panel for managing the site's content.

Features

This application provides a complete workflow for both regular users and site administrators.

User Features

- **User Authentication:** Secure user registration and login system.
- **Browse Movies:** View a gallery of all available movies with their details.
- **View Showtimes:** Select a movie to see all its upcoming showtimes.
- **Interactive Seat Selection:** A dynamic, clickable seat map shows available, booked, and selected seats.
- **Add-ons:** Option to add food and beverage items to the booking.
- **Real-Time Price Calculation:** The total price updates instantly as seats and food items are selected.
- **Booking History:** Users can view a complete history of all their past bookings.

Admin Features

- **Secure Admin Panel:** Separate login for the administrator.
- **Dashboard:** An overview of site statistics, including total movies and bookings.
- **Movie Management (CRUD):** Admins can **C**reate, **R**ead, **U**pdate, and **D**elete movies from the database.
- **Showtime Management:** Admins can add new showtimes for existing movies.
- **View All Bookings:** A master list of all bookings made by all users, including details like seats and food orders.

Tech Stack

The project is built with a modern and lightweight tech stack:

- **Backend:** Python with the Flask Framework
- **Frontend:** HTML5, Tailwind CSS, JavaScript
- **Database:** SQLite3
- **Templating Engine:** Jinja2

Project Structure

The project is organized into a clean and logical structure:

```
movie_ticket_system/
├── templates/
│   ├── admin/           # HTML templates for the admin panel
│   │   ├── add_movie.html
│   │   ├── add_show.html
│   │   └── ...
│   ├── booking_success.html
│   ├── index.html
│   └── ...              # HTML templates for the user view
├── app.py               # The main Flask application (backend logic)
├── database.py          # Script to initialize the database and tables
└── database.db          # The SQLite database file (created automatically)
```

Setup and Installation

To run this project locally, follow these steps:

1. Clone the repository (or download the files):

```
git clone <your-repository-url>
cd movie_ticket_system
```

2. Create and activate a virtual environment:

- This keeps your project dependencies isolated.

```
# For Windows
python -m venv venv
venv\Scripts\activate

# For macOS/Linux
python3 -m venv venv
source venv/bin/activate
```

3. Install the required packages:

```
pip install Flask Werkzeug
```

4. Initialize the database:

- This step only needs to be run once. It will create the `database.db` file, set up all the tables, and create the default admin user.

```
python database.py
```

5. Run the Flask application:

```
flask run
```

The application will now be running at `http://127.0.0.1:5000`.

Usage

Once the application is running, you can access it in your web browser.

Admin Access

- **URL:** `http://127.0.0.1:5000/login`
- **Username:** `admin`
- **Password:** `adminpassword`

From the admin panel, you can start by adding a few movies and then scheduling some shows for them.

User Access

- Go to the registration page to create a new user account.
- Log in with your new account to browse movies and book tickets.