PERFECTLY-VIEWABLE-CINEMAS

Index

|  |  |  |
| --- | --- | --- |
| S. No. | Topic | Page No. |
| 1 | Certificate | 1 |
| 2 | Name of the Project | 2 |
| 3 | Index | 3 |
| 4 | Acknowledgement | 4 |
| 5 | Synopsis | 5 |
| 6 | Code | 6 |
| 7 | Output Screens | 7 |
| 8 | Bibliography | 8 |

ACKNOWLEDGEMENT

## We undertook this project as part of our class 12 Computer Science course. We had tried to use the best of our knowledge and experience to develop this software. Developing a software requires time, patience and systematic approach.

## We are thankful to Almighty for his immeasurable blessings.We would like to express our gratitude to our principal Ms.Prathiba.V and to our teacher Ms.Yasmin for their constant support and guidance.

## We extend our gratitude to all other staff members of Computer department and our parents for their moral support and patience. Last but not the least, we thank our school for providing a platform to develop our skills and help us gain knowledge.

Synopsis

## Synopsis

Perfectly Viewable Cinemas is a modern cinema management and ticket booking system built with a CSV-based file system. It offers two interaction modes:

1. A command-line (CLI) interface for system administrators to manage platform-wide operations.
2. A class-based Streamlit GUI (CinemaGUI) providing visual interfaces for theatre admins and end-users.

### Core Features

* **Role-Based Access Control**: Separate privileges for system admin, theatre admin, and user roles
* **Enhanced Visual Experience**: Movie posters, emoji seat indicators, and modern icons
* **CSV Persistence**: File-based storage without external databases
* **Advanced Security**: Salted HMAC-SHA256 password hashing with enforcement of bans
* **Interactive Seat Selection**: Dynamic grid with real-time availability
* **Theatre Analytics**: Revenue tracking and booking aggregation

### User Types and Access Levels

#### 1. System Administrator

* **Access**: CLI only
* **Capabilities**:
  + Initialize/reset CSV data
  + Manage theatre admin accounts
  + Manage user accounts
  + Ban/unban users
  + Ensure data integrity
* **Authentication**: Username/password with elevated privileges

#### 2. Theatre Administrator

* **Access**: Streamlit GUI (admin dashboard)
* **Capabilities**:
  + Add new movie showings with metadata
  + View bookings for their theatre
  + Monitor revenue and analytics
  + Manage theatre inventory and schedules
* **Authentication**: Username/password tied to theatre ID

#### 3. Regular User (Customer)

* **Access**: Streamlit GUI (customer interface)
* **Capabilities**:
  + Browse movies across theatres
  + Select seats with real-time availability
  + Book tickets with conflict checks
  + View and cancel bookings
  + Register and manage account
* **Authentication**: Username/password with email verification

### Data Model (CSV Files)

CSV Data Model  
├── movies\_showings.csv  
│ ├── id  
│ ├── title  
│ ├── genre  
│ ├── duration  
│ ├── theatre\_id  
│ ├── showtime  
│ ├── available\_seats  
│ ├── price  
│ └── image\_url  
│  
├── users.csv  
│ ├── user\_id  
│ ├── username  
│ ├── password  
│ ├── salt  
│ ├── email  
│ └── status (active|banned)  
│  
├── admins.csv  
│ ├── admin\_id  
│ ├── username  
│ ├── password  
│ ├── salt  
│ ├── type (system|theatre)  
│ └── theatre\_id  
│  
└── bookings.csv  
 ├── booking\_id  
 ├── user\_id  
 ├── showing\_id  
 ├── seats\_booked  
 ├── seat\_numbers  
 ├── total\_price  
 └── booking\_date (ISO timestamp)

### Security Considerations

* **Password Security**: Salted HMAC-SHA256 hashes with secure random salts
* **Authentication Enforcement**: Banned users blocked from access
* **Session Management**: Streamlit session state preserves user context
* **Input Validation**: Validation and type-checking across CLI and GUI
* **Role Separation**: System admins restricted to CLI; GUI reserved for theatre admins and users

Code

### A. handler.py

Includes business logic: CSV management, core functions for movies, bookings, authentication.

"""

Handler module for PVC - Perfectly Viewable Cinemas

Contains all business logic and CSV operations.

"""

import csv

import os

from datetime import datetime

USERS\_FILE = 'users.csv'

MOVIES\_FILE = 'movies.csv' # Movie catalog

SHOWINGS\_FILE = 'showings.csv' # Showings (movie in theatre)

THEATRES\_FILE = 'theatres.csv'

BOOKINGS\_FILE = 'bookings.csv'

ADMINS\_FILE = 'admins.csv'

# --- CSV Operations ---

def initialize\_csv\_files():

"""Initialize CSV files with headers if they don't exist"""

# Users CSV

if not os.path.exists(USERS\_FILE):

with open(USERS\_FILE, 'w', newline='') as f:

writer = csv.writer(f)

writer.writerow(['user\_id', 'username', 'password', 'email'])

writer.writerow(['1', 'demo\_user', 'password123', 'demo@email.com'])

# Movies CSV

if not os.path.exists(MOVIES\_FILE):

with open(MOVIES\_FILE, 'w', newline='') as f:

writer = csv.writer(f)

writer.writerow(['movie\_id', 'title', 'genre', 'duration'])

writer.writerow(['1', 'Avatar: The Way of Water', 'Sci-Fi', '192'])

# Showings CSV

if not os.path.exists(SHOWINGS\_FILE):

with open(SHOWINGS\_FILE, 'w', newline='') as f:

writer = csv.writer(f)

writer.writerow(['showing\_id', 'movie\_id', 'theatre\_id', 'showtime', 'available\_seats'])

writer.writerow(['1', '1', '1', '18:00', '50'])

# Theatres CSV

if not os.path.exists(THEATRES\_FILE):

with open(THEATRES\_FILE, 'w', newline='') as f:

writer = csv.writer(f)

writer.writerow(['theatre\_id', 'name', 'location', 'total\_seats'])

writer.writerow(['1', 'PVR Cinemas', 'Mall Road', '100'])

# Bookings CSV

if not os.path.exists(BOOKINGS\_FILE):

with open(BOOKINGS\_FILE, 'w', newline='') as f:

writer = csv.writer(f)

writer.writerow(['booking\_id', 'user\_id', 'showing\_id', 'seats\_booked', 'booking\_date'])

# Admins CSV

if not os.path.exists(ADMINS\_FILE):

with open(ADMINS\_FILE, 'w', newline='') as f:

writer = csv.writer(f)

writer.writerow(['admin\_id', 'username', 'password', 'type', 'theatre\_id'])

writer.writerow(['1', 'system\_admin', 'admin123', 'system', ''])

writer.writerow(['2', 'theatre\_admin1', 'theatre123', 'theatre', '1'])

print('Init complete.')

def read\_csv\_file(filename):

data = []

if os.path.exists(filename):

with open(filename, 'r', newline='') as f:

reader = csv.DictReader(f)

data = list(reader)

return data

def write\_csv\_file(filename, data, fieldnames):

with open(filename, 'w', newline='') as f:

writer = csv.DictWriter(f, fieldnames=fieldnames)

writer.writeheader()

writer.writerows(data)

# --- Authentication ---

def user\_login(username, password):

users = read\_csv\_file(USERS\_FILE)

for user in users:

if user['username'] == username and user['password'] == password:

return user

return None

def admin\_login(username, password):

admins = read\_csv\_file(ADMINS\_FILE)

for admin in admins:

if admin['username'] == username and admin['password'] == password:

return admin

return None

def register\_user(username, password, email):

users = read\_csv\_file(USERS\_FILE)

for user in users:

if user['username'] == username:

return False, "Username already exists!"

new\_id = str(len(users) + 1)

users.append({'user\_id': new\_id, 'username': username, 'password': password, 'email': email})

write\_csv\_file(USERS\_FILE, users, ['user\_id', 'username', 'password', 'email'])

return True, "Registration successful!"

# --- Movie Operations ---

def get\_movies():

"""Return movie catalog."""

return read\_csv\_file(MOVIES\_FILE)

def get\_showings():

"""Return all showings."""

return read\_csv\_file(SHOWINGS\_FILE)

def get\_theatres():

return read\_csv\_file(THEATRES\_FILE)

def add\_movie(title, genre, duration):

"""Add a new movie to the catalog."""

movies = read\_csv\_file(MOVIES\_FILE)

new\_movie\_id = str(len(movies) + 1)

movies.append({'movie\_id': new\_movie\_id, 'title': title, 'genre': genre, 'duration': duration})

write\_csv\_file(MOVIES\_FILE, movies, ['movie\_id', 'title', 'genre', 'duration'])

return True, f"Movie '{title}' added successfully!"

def del\_movie(movie\_id):

"""Delete a movie from the catalog."""

movies = read\_csv\_file(MOVIES\_FILE)

for i in movies:

if i['movie\_id'] == movie\_id:

movies.remove(i)

break

write\_csv\_file(MOVIES\_FILE, movies, ['movie\_id', 'title', 'genre', 'duration'])

showings = read\_csv\_file(SHOWINGS\_FILE)

showings = [s for s in showings if s['movie\_id'] != movie\_id]

write\_csv\_file(SHOWINGS\_FILE, showings, ['showing\_id', 'movie\_id', 'theatre\_id', 'showtime', 'available\_seats'])

return True, f"Movie ID {movie\_id} deleted successfully!"

def add\_showing(movie\_id, theatre\_id, showtime, seats):

"""Add a new showing for a movie in a theatre."""

showings = read\_csv\_file(SHOWINGS\_FILE)

new\_showing\_id = str(len(showings) + 1)

showings.append({'showing\_id': new\_showing\_id, 'movie\_id': movie\_id, 'theatre\_id': theatre\_id, 'showtime': showtime, 'available\_seats': seats})

write\_csv\_file(SHOWINGS\_FILE, showings, ['showing\_id', 'movie\_id', 'theatre\_id', 'showtime', 'available\_seats'])

return True, f"Showing added successfully!"

# --- Booking Operations ---

def book\_ticket(user\_id, showing\_id, seats\_wanted):

showings = read\_csv\_file(SHOWINGS\_FILE)

selected\_showing = None

for showing in showings:

if showing['showing\_id'] == showing\_id:

selected\_showing = showing

break

if not selected\_showing:

return False, "Showing not found!"

available\_seats = int(selected\_showing['available\_seats'])

if seats\_wanted > available\_seats:

return False, f"Only {available\_seats} seats available!"

selected\_showing['available\_seats'] = str(available\_seats - seats\_wanted)

write\_csv\_file(SHOWINGS\_FILE, showings, ['showing\_id', 'movie\_id', 'theatre\_id', 'showtime', 'available\_seats'])

bookings = read\_csv\_file(BOOKINGS\_FILE)

new\_booking\_id = str(len(bookings) + 1)

bookings.append({'booking\_id': new\_booking\_id, 'user\_id': user\_id, 'showing\_id': showing\_id, 'seats\_booked': str(seats\_wanted), 'booking\_date': datetime.now().strftime('%Y-%m-%d %H:%M:%S')})

write\_csv\_file(BOOKINGS\_FILE, bookings, ['booking\_id', 'user\_id', 'showing\_id', 'seats\_booked', 'booking\_date'])

return True, new\_booking\_id

def get\_user\_bookings(user\_id):

bookings = read\_csv\_file(BOOKINGS\_FILE)

return [b for b in bookings if b['user\_id'] == user\_id]

def get\_theatre\_bookings(theatre\_id):

showings = read\_csv\_file(SHOWINGS\_FILE)

theatre\_showing\_ids = [s['showing\_id'] for s in showings if s['theatre\_id'] == theatre\_id]

bookings = read\_csv\_file(BOOKINGS\_FILE)

return [b for b in bookings if b['showing\_id'] in theatre\_showing\_ids]

def get\_all\_bookings():

return read\_csv\_file(BOOKINGS\_FILE)

def get\_users():

return read\_csv\_file(USERS\_FILE)

def get\_admins():

return read\_csv\_file(ADMINS\_FILE)

### B. gui.py

Includes Streamlit interface: GUI tabs for login, movies, booking, admin control.

"""

Streamlit UI for PVC - Perfectly Viewable Cinemas

Imports business logic from handler.py

Allows viewing and editing CSV data.

"""

import streamlit as st

import pandas as pd

from handler import \*

st.set\_page\_config(page\_title="PVC Cinemas", layout="wide")

st.title("Perfectly Viewable Cinemas")

menu = ["Login", "Register", "Movies", "Book Ticket", "My Bookings", "Admin: Movies", "Admin: Bookings",]

tabs = st.tabs(menu)

with tabs[0]:

st.header("User/Admin Login")

username = st.text\_input("Username")

password = st.text\_input("Password", type="password")

login\_type = st.radio("Login as", ["User", "Admin"])

if st.button("Login"):

if login\_type == "User":

user = user\_login(username, password)

if user:

st.success(f"Welcome, {username}!")

st.session\_state['user'] = user

st.session\_state['user\_type'] = 'user'

else:

st.error("Invalid credentials!")

else:

admin = admin\_login(username, password)

if admin:

st.success(f"Welcome, {username}! ({admin['type']} admin)")

st.session\_state['user'] = admin

st.session\_state['user\_type'] = admin['type']

else:

st.error("Invalid admin credentials!")

with tabs[1]:

st.header("User Registration")

username = st.text\_input("New Username")

password = st.text\_input("New Password", type="password")

email = st.text\_input("Email")

if st.button("Register"):

success, msg = register\_user(username, password, email)

if success:

st.success(msg)

else:

st.error(msg)

with tabs[2]:

st.header("Available Movies")

movies = get\_movies()

showings = get\_showings()

theatres = get\_theatres()

if showings:

df = pd.DataFrame(showings)

df['Movie'] = df['movie\_id'].map({m['movie\_id']: m['title'] for m in movies})

df['Theatre'] = df['theatre\_id'].map({t['theatre\_id']: t['name'] for t in theatres})

st.dataframe(df)

else:

st.info("No showings available.")

with tabs[3]:

st.header("Book a Ticket")

user = st.session\_state.get('user', None)

if not user or st.session\_state.get('user\_type') != 'user':

st.warning("Please login as a user first!")

else:

showings = get\_showings()

movies = get\_movies()

showing\_titles = [f"{m['title']} - {s['showtime']} (ID: {s['showing\_id']})" for s in showings for m in movies if m['movie\_id'] == s['movie\_id']]

selected = st.selectbox("Select Showing", showing\_titles)

seats\_wanted = st.number\_input("Number of seats", min\_value=1, value=1)

if st.button("Book"):

showing\_id = selected.split("ID: ")[-1].replace(")", "")

success, result = book\_ticket(user['user\_id'], showing\_id, seats\_wanted)

if success:

st.success(f"Booking successful! Booking ID: {result}")

else:

st.error(result)

with tabs[4]:

st.header("My Bookings")

user = st.session\_state.get('user', None)

if not user or st.session\_state.get('user\_type') != 'user':

st.warning("Please login as a user first!")

else:

bookings = get\_user\_bookings(user['user\_id'])

df = pd.DataFrame(bookings)

if not df.empty:

st.dataframe(df)

else:

st.info("No bookings found.")

with tabs[5]:

st.header("Admin: Add Showing")

admin = st.session\_state.get('user', None)

if not admin or st.session\_state.get('user\_type') != 'theatre':

st.warning("Theatre admin login required.")

else:

movies = get\_movies()

movie\_titles = [f"{m['title']} (ID: {m['movie\_id']})" for m in movies]

selected = st.selectbox("Select Movie", movie\_titles)

showtime = st.text\_input("Showtime (HH:MM)")

seats = st.text\_input("Available Seats")

if st.button("Add Showing"):

movie\_id = selected.split("ID: ")[-1].replace(")", "")

success, msg = add\_showing(movie\_id, admin['theatre\_id'], showtime, seats)

if success:

st.success(msg)

else:

st.error(msg)

with tabs[6]:

st.header("Admin: Theatre Bookings")

admin = st.session\_state.get('user', None)

if not admin or st.session\_state.get('user\_type') != 'theatre':

st.warning("Theatre admin login required.")

else:

bookings = get\_theatre\_bookings(admin['theatre\_id'])

df = pd.DataFrame(bookings)

if not df.empty:

st.dataframe(df)

else:

st.info("No bookings found for this theatre.")

C. cli.py

Includes CLI interface: user/admin menus, booking flow, admin operations.

#!/usr/bin/env python3

"""

CLI for PVC - Perfectly Viewable Cinemas

Imports business logic from handler.py

"""

import sys

from handler import \*

current\_user = None

current\_user\_type = None

def cli\_user\_login():

global current\_user, current\_user\_type

print("\n=== USER LOGIN ===")

username = input("Username: ")

password = input("Password: ")

user = user\_login(username, password)

if user:

current\_user = user

current\_user\_type = 'user'

print(f"Welcome, {username}!")

return True

print("Invalid credentials!")

return False

def cli\_admin\_login():

global current\_user, current\_user\_type

print("\n=== ADMIN LOGIN ===")

username = input("Username: ")

password = input("Password: ")

admin = admin\_login(username, password)

if admin:

current\_user = admin

current\_user\_type = admin['type']

print(f"Welcome, {username}! ({admin['type']} admin)")

return True

print("Invalid admin credentials!")

return False

def cli\_register\_user():

print("\n=== USER REGISTRATION ===")

username = input("Username: ")

password = input("Password: ")

email = input("Email: ")

success, msg = register\_user(username, password, email)

print(msg)

return success

def cli\_view\_movies():

print("\n=== AVAILABLE MOVIES ===")

movies = get\_movies()

showings = get\_showings()

theatres = get\_theatres()

if not showings:

print("No showings available.")

return

movie\_lookup = {m['movie\_id']: m['title'] for m in movies}

theatre\_lookup = {t['theatre\_id']: t['name'] for t in theatres}

print(f"{'Showing ID':<10} {'Movie':<25} {'Theatre':<15} {'Showtime':<8} {'Seats':<5}")

print("-" \* 70)

for showing in showings:

movie\_title = movie\_lookup.get(showing['movie\_id'], 'Unknown')

theatre\_name = theatre\_lookup.get(showing['theatre\_id'], 'Unknown')

print(f"{showing['showing\_id']:<10} {movie\_title[:24]:<25} {theatre\_name:<15} {showing['showtime']:<8} {showing['available\_seats']:<5}")

def cli\_book\_ticket():

if not current\_user or current\_user\_type != 'user':

print("Please login as a user first!")

return

print("\n=== BOOK TICKET ===")

cli\_view\_movies()

showing\_id = input("\nEnter Showing ID to book: ")

seats\_wanted = input("Number of seats: ")

try:

seats\_wanted = int(seats\_wanted)

except ValueError:

print("Invalid number of seats!")

return

success, result = book\_ticket(current\_user['user\_id'], showing\_id, seats\_wanted)

if success:

print(f"\nBooking successful! Booking ID: {result}")

else:

print(result)

def cli\_view\_user\_bookings():

if not current\_user or current\_user\_type != 'user':

print("Please login as a user first!")

return

print("\n=== YOUR BOOKINGS ===")

bookings = get\_user\_bookings(current\_user['user\_id'])

showings = get\_showings()

movies = get\_movies()

theatres = get\_theatres()

if not bookings:

print("No bookings found.")

return

showing\_lookup = {s['showing\_id']: s for s in showings}

movie\_lookup = {m['movie\_id']: m['title'] for m in movies}

theatre\_lookup = {t['theatre\_id']: t['name'] for t in theatres}

print(f"{'Booking ID':<10} {'Movie':<25} {'Theatre':<15} {'Seats':<5} {'Date':<19}")

print("-" \* 80)

for booking in bookings:

showing = showing\_lookup.get(booking['showing\_id'], None)

if showing:

movie\_title = movie\_lookup.get(showing['movie\_id'], 'Unknown')

theatre\_name = theatre\_lookup.get(showing['theatre\_id'], 'Unknown')

else:

movie\_title = 'Unknown'

theatre\_name = 'Unknown'

print(f"{booking['booking\_id']:<10} {movie\_title[:24]:<25} {theatre\_name:<15} {booking['seats\_booked']:<5} {booking['booking\_date']:<19}")

def cli\_add\_movie():

if not current\_user or current\_user\_type != 'theatre':

print("Access denied! Theatre admin login required.")

return

print("\n=== ADD MOVIE ===")

title = input("Movie Title: ")

genre = input("Genre: ")

duration = input("Duration (minutes): ")

showtime = input("Showtime (HH:MM): ")

seats = input("Available Seats: ")

# Add movie first

success, msg = add\_movie(title, genre, duration)

print(msg)

if success:

# Get the new movie\_id

movies = get\_movies()

new\_movie\_id = movies[-1]['movie\_id']

# Add showing for this movie

success2, msg2 = add\_showing(new\_movie\_id, current\_user['theatre\_id'], showtime, seats)

print(msg2)

def cli\_del\_movie():

if not current\_user or current\_user\_type != 'theatre':

print("Access denied! Theatre admin login required.")

return

print("\n=== DELETE MOVIE ===")

movies = get\_movies()

if not movies:

print("No movies available to delete.")

return

print(f"{'Movie ID':<10} {'Title':<25} {'Genre':<15} {'Duration':<8}")

print("-" \* 60)

for movie in movies:

print(f"{movie['movie\_id']:<10} {movie['title'][:24]:<25} {movie['genre']:<15} {movie['duration']:<8}")

movie\_id = input("\nEnter Movie ID to delete: ")

success, msg = del\_movie(movie\_id)

print(msg)

def cli\_add\_showing():

if not current\_user or current\_user\_type != 'theatre':

print("Access denied! Theatre admin login required.")

return

print("\n=== ADD SHOWING ===")

movies = get\_movies()

movie\_titles = [f"{m['title']} (ID: {m['movie\_id']})" for m in movies]

selected = input("Select Movie by ID: ")

theatre\_id = current\_user['theatre\_id']

showtime = input("Showtime (HH:MM): ")

seats = input("Available Seats: ")

success, msg = add\_showing(selected, theatre\_id, showtime, seats)

print(msg)

def cli\_view\_theatre\_bookings():

if not current\_user or current\_user\_type != 'theatre':

print("Access denied! Theatre admin login required.")

return

print(f"\n=== BOOKINGS FOR THEATRE {current\_user['theatre\_id']} ===")

bookings = get\_theatre\_bookings(current\_user['theatre\_id'])

showings = get\_showings()

movies = get\_movies()

users = get\_users()

if not bookings:

print("No bookings found for this theatre.")

return

showing\_lookup = {s['showing\_id']: s for s in showings}

movie\_lookup = {m['movie\_id']: m['title'] for m in movies}

user\_lookup = {u['user\_id']: u['username'] for u in users}

print(f"{'Booking ID':<10} {'User':<15} {'Movie':<25} {'Seats':<5} {'Date':<19}")

print("-" \* 80)

for booking in bookings:

showing = showing\_lookup.get(booking['showing\_id'], None)

if showing:

movie\_title = movie\_lookup.get(showing['movie\_id'], 'Unknown')

else:

movie\_title = 'Unknown'

username = user\_lookup.get(booking['user\_id'], 'Unknown')

print(f"{booking['booking\_id']:<10} {username:<15} {movie\_title[:24]:<25} {booking['seats\_booked']:<5} {booking['booking\_date']:<19}")

def cli\_system\_admin\_stats():

if not current\_user or current\_user\_type != 'system':

print("Access denied! System admin login required.")

return

print("\n=== SYSTEM STATISTICS ===")

users = get\_users()

movies = get\_movies()

theatres = get\_theatres()

bookings = get\_all\_bookings()

print(f"Total Users: {len(users)}")

print(f"Total Movies: {len(movies)}")

print(f"Total Theatres: {len(theatres)}")

print(f"Total Bookings: {len(bookings)}")

total\_tickets = sum(int(b['seats\_booked']) for b in bookings)

total\_revenue = total\_tickets \* 10

print(f"Total Tickets Sold: {total\_tickets}")

print(f"Estimated Revenue: ${total\_revenue}")

def cli\_view\_all\_bookings():

print("\n=== ALL BOOKINGS ===")

bookings = get\_all\_bookings()

showings = get\_showings()

movies = get\_movies()

users = get\_users()

theatres = get\_theatres()

if not bookings:

print("No bookings found.")

return

showing\_lookup = {s['showing\_id']: s for s in showings}

movie\_lookup = {m['movie\_id']: m['title'] for m in movies}

user\_lookup = {u['user\_id']: u['username'] for u in users}

theatre\_lookup = {t['theatre\_id']: t['name'] for t in theatres}

print(f"{'ID':<4} {'User':<12} {'Movie':<20} {'Theatre':<12} {'Seats':<5} {'Date':<19}")

print("-" \* 75)

for booking in bookings:

showing = showing\_lookup.get(booking['showing\_id'], None)

if showing:

movie\_title = movie\_lookup.get(showing['movie\_id'], 'Unknown')

theatre\_name = theatre\_lookup.get(showing['theatre\_id'], 'Unknown')

else:

movie\_title = 'Unknown'

theatre\_name = 'Unknown'

username = user\_lookup.get(booking['user\_id'], 'Unknown')

print(f"{booking['booking\_id']:<4} {username:<12} {movie\_title[:19]:<20} {theatre\_name[:11]:<12} {booking['seats\_booked']:<5} {booking['booking\_date']:<19}")

def logout():

global current\_user, current\_user\_type

print("Logged out successfully!")

current\_user = None

current\_user\_type = None

def main():

print("=" \* 50)

print(" Welcome to PVC - Perfectly Viewable Cinemas")

print("=" \* 50)

while True:

if not current\_user:

print("\n=== MAIN MENU ===")

print("1. User Login")

print("2. User Registration")

print("3. Admin Login")

print("4. Exit")

choice = input("Enter your choice (1-4): ")

if choice == '1':

if cli\_user\_login():

user\_menu()

elif choice == '2':

cli\_register\_user()

elif choice == '3':

if cli\_admin\_login():

if current\_user\_type == 'theatre':

theatre\_admin\_menu()

elif current\_user\_type == 'system':

system\_admin\_menu()

elif choice == '4':

print("Thank you for using PVC!")

sys.exit()

elif choice == 'init2025':

initialize\_csv\_files()

print('Initializing Csv files..........')

else:

print("Invalid choice!")

else:

if current\_user\_type == 'user':

user\_menu()

elif current\_user\_type == 'theatre':

theatre\_admin\_menu()

elif current\_user\_type == 'system':

system\_admin\_menu()

def user\_menu():

while True:

print("\n=== USER MENU ===")

print("1. View Available Movies")

print("2. Book Ticket")

print("3. View My Bookings")

print("4. Logout")

choice = input("Enter your choice (1-4): ")

if choice == '1':

cli\_view\_movies()

elif choice == '2':

cli\_book\_ticket()

elif choice == '3':

cli\_view\_user\_bookings()

elif choice == '4':

logout()

break

else:

print("Invalid choice!")

def theatre\_admin\_menu():

while True:

print("\n=== THEATRE ADMIN MENU ===")

print("1. View Available Movies")

print("2. Add Movie")

print("3. View Theatre Bookings")

print("4. Delete Movie")

print("5. Logout")

choice = input("Enter your choice (1-5): ")

if choice == '1':

cli\_view\_movies()

elif choice == '2':

cli\_add\_movie()

elif choice == '3':

cli\_view\_theatre\_bookings()

elif choice == '4':

cli\_del\_movie()

elif choice == '5':

logout()

break

else:

print("Invalid choice!")

def system\_admin\_menu():

while True:

print("\n=== SYSTEM ADMIN MENU ===")

print("1. View System Statistics")

print("2. View All Movies")

print("3. View All Bookings")

print("4. Logout")

choice = input("Enter your choice (1-4): ")

if choice == '1':

cli\_system\_admin\_stats()

elif choice == '2':

cli\_view\_movies()

elif choice == '3':

cli\_view\_all\_bookings()

elif choice == '4':

logout()

break

else:

print("Invalid choice!")

if \_\_name\_\_ == "\_\_main\_\_":

main()

Output

cli.py

python .\cli.py

==================================================

Welcome to PVC - Perfectly Viewable Cinemas

==================================================

=== MAIN MENU ===

1. User Login

2. User Registration

3. Admin Login

4. Exit

Enter your choice (1-4): 1

=== USER LOGIN ===

Username: demo\_user

Password: password123

Welcome, demo\_user!

=== USER MENU ===

1. View Available Movies

2. Book Ticket

3. View My Bookings

4. Logout

Enter your choice (1-4): 1

=== AVAILABLE MOVIES ===

Showing ID Movie Theatre Showtime Seats

----------------------------------------------------------------------

3 Lokah PVR Cinemas 18:00 50

4 Lokah PVR Cinemas 14:00 50

=== USER MENU ===

1. View Available Movies

2. Book Ticket

3. View My Bookings

4. Logout

Enter your choice (1-4): 2

=== BOOK TICKET ===

=== AVAILABLE MOVIES ===

Showing ID Movie Theatre Showtime Seats

----------------------------------------------------------------------

3 Lokah PVR Cinemas 18:00 50

4 Lokah PVR Cinemas 14:00 50

Enter Showing ID to book: 3

Number of seats: 3

Booking successful! Booking ID: 1

=== USER MENU ===

1. View Available Movies

2. Book Ticket

3. View My Bookings

4. Logout

Enter your choice (1-4): 3

=== YOUR BOOKINGS ===

Booking ID Movie Theatre Seats Date

--------------------------------------------------------------------------------

1 Lokah PVR Cinemas 3 2025-09-14 08:36:17

=== USER MENU ===

1. View Available Movies

2. Book Ticket

3. View My Bookings

4. Logout

Enter your choice (1-4): 4

Logged out successfully!

=== MAIN MENU ===

1. User Login

2. User Registration

3. Admin Login

4. Exit

Enter your choice (1-4): 2

=== USER REGISTRATION ===

Username: Navaneeth Binoy

Password: nav1234

Email: navaneethbinoy@gmail.com

Registration successful!

=== MAIN MENU ===

1. User Login

2. User Registration

3. Admin Login

4. Exit

Enter your choice (1-4): 1

=== USER LOGIN ===

Username: Navaneeth Binoy

Password: nav1234

Welcome, Navaneeth Binoy!

=== USER MENU ===

1. View Available Movies

2. Book Ticket

3. View My Bookings

4. Logout

Enter your choice (1-4): 4

Logged out successfully!

=== MAIN MENU ===

1. User Login

2. User Registration

3. Admin Login

4. Exit

Enter your choice (1-4): 3

=== ADMIN LOGIN ===

Username: theatre\_admin1

Password: theatre123

Welcome, theatre\_admin1! (theatre admin)

=== THEATRE ADMIN MENU ===

1. View Available Movies

2. Add Movie

3. View Theatre Bookings

4. Delete Movie

5. Logout

Enter your choice (1-5): 1

=== AVAILABLE MOVIES ===

Showing ID Movie Theatre Showtime Seats

----------------------------------------------------------------------

3 Lokah PVR Cinemas 18:00 47

4 Lokah PVR Cinemas 14:00 50

=== THEATRE ADMIN MENU ===

1. View Available Movies

2. Add Movie

3. View Theatre Bookings

4. Delete Movie

5. Logout

Enter your choice (1-5): 2

=== ADD MOVIE ===

Movie Title: infinity war

Genre: sci-fi

Duration (minutes): 200

Showtime (HH:MM): 13:00

Available Seats: 60

Movie 'infinity war' added successfully!

Showing added successfully!

=== THEATRE ADMIN MENU ===

1. View Available Movies

2. Add Movie

3. View Theatre Bookings

4. Delete Movie

5. Logout

Enter your choice (1-5): 3

=== BOOKINGS FOR THEATRE 1 ===

Booking ID User Movie Seats Date

--------------------------------------------------------------------------------

1 demo\_user infinity war 3 2025-09-14 08:36:17

=== THEATRE ADMIN MENU ===

1. View Available Movies

2. Add Movie

3. View Theatre Bookings

4. Delete Movie

5. Logout

Enter your choice (1-5): 4

=== DELETE MOVIE ===

Movie ID Title Genre Duration

------------------------------------------------------------

2 Lokah Sci-fi 150

2 infinity war sci-fi 200

Enter Movie ID to delete: 2

Movie ID 2 deleted successfully!

=== THEATRE ADMIN MENU ===

1. View Available Movies

2. Add Movie

3. View Theatre Bookings

4. Delete Movie

5. Logout

Enter your choice (1-5): 1

=== AVAILABLE MOVIES ===

No showings available.

=== THEATRE ADMIN MENU ===

1. View Available Movies

2. Add Movie

3. View Theatre Bookings

4. Delete Movie

5. Logout

Enter your choice (1-5): 5

Logged out successfully!

=== MAIN MENU ===

1. User Login

2. User Registration

3. Admin Login

4. Exit

Enter your choice (1-4): 3

=== ADMIN LOGIN ===

Username: system\_admin1

Password: theatre123

Invalid admin credentials!

=== MAIN MENU ===

1. User Login

2. User Registration

3. Admin Login

4. Exit

Enter your choice (1-4): 3

=== ADMIN LOGIN ===

Username: system\_admin1

Password: admin123

Invalid admin credentials!

=== MAIN MENU ===

1. User Login

2. User Registration

3. Admin Login

4. Exit

Enter your choice (1-4): 3

=== ADMIN LOGIN ===

Username: system\_admin

Password: admin123

Welcome, system\_admin! (system admin)

=== SYSTEM ADMIN MENU ===

1. View System Statistics

2. View All Movies

3. View All Bookings

4. Logout

Enter your choice (1-4): 1

=== SYSTEM STATISTICS ===

Total Users: 2

Total Movies: 1

Total Theatres: 1

Total Bookings: 1

Total Tickets Sold: 3

Estimated Revenue: $30

=== SYSTEM ADMIN MENU ===

1. View System Statistics

2. View All Movies

3. View All Bookings

4. Logout

Enter your choice (1-4): 2

=== AVAILABLE MOVIES ===

No showings available.

=== SYSTEM ADMIN MENU ===

1. View System Statistics

2. View All Movies

3. View All Bookings

4. Logout

Enter your choice (1-4): 3

=== ALL BOOKINGS ===

ID User Movie Theatre Seats Date

---------------------------------------------------------------------------

1 demo\_user Unknown Unknown 3 2025-09-14 08:36:17

=== SYSTEM ADMIN MENU ===

1. View System Statistics

2. View All Movies

3. View All Bookings

4. Logout

Enter your choice (1-4): 4

Logged out successfully!

=== MAIN MENU ===

1. User Login

2. User Registration

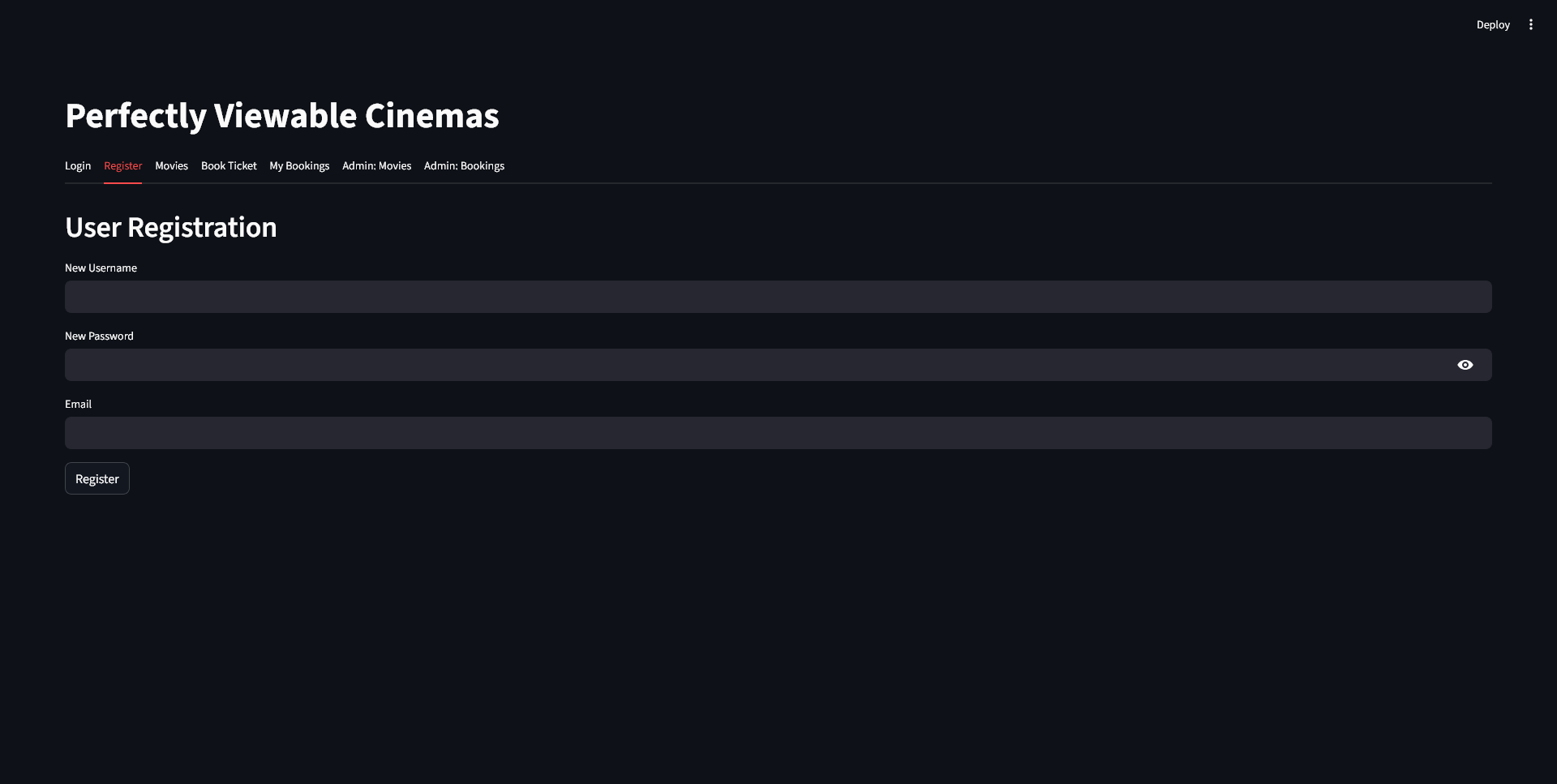
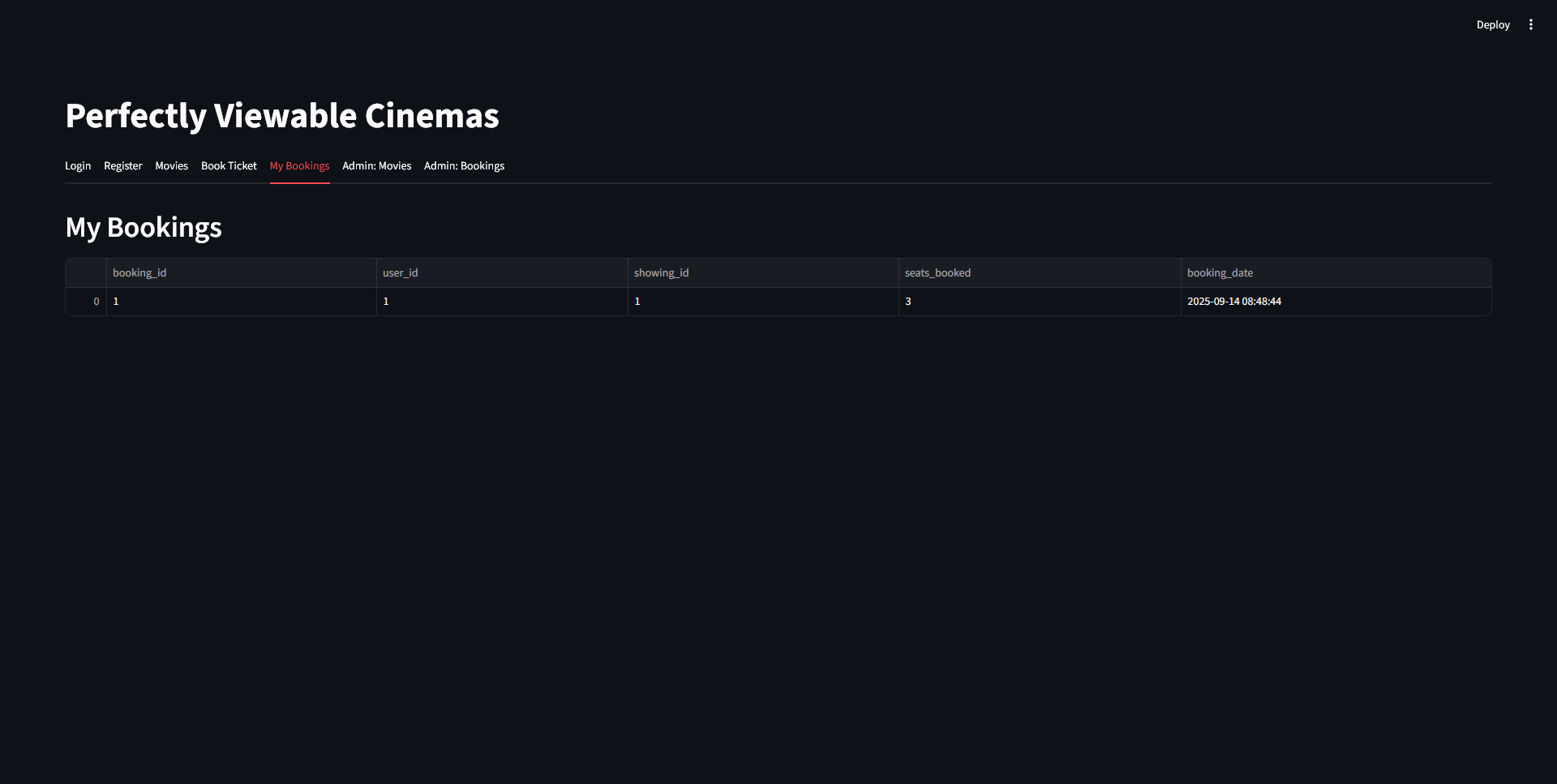
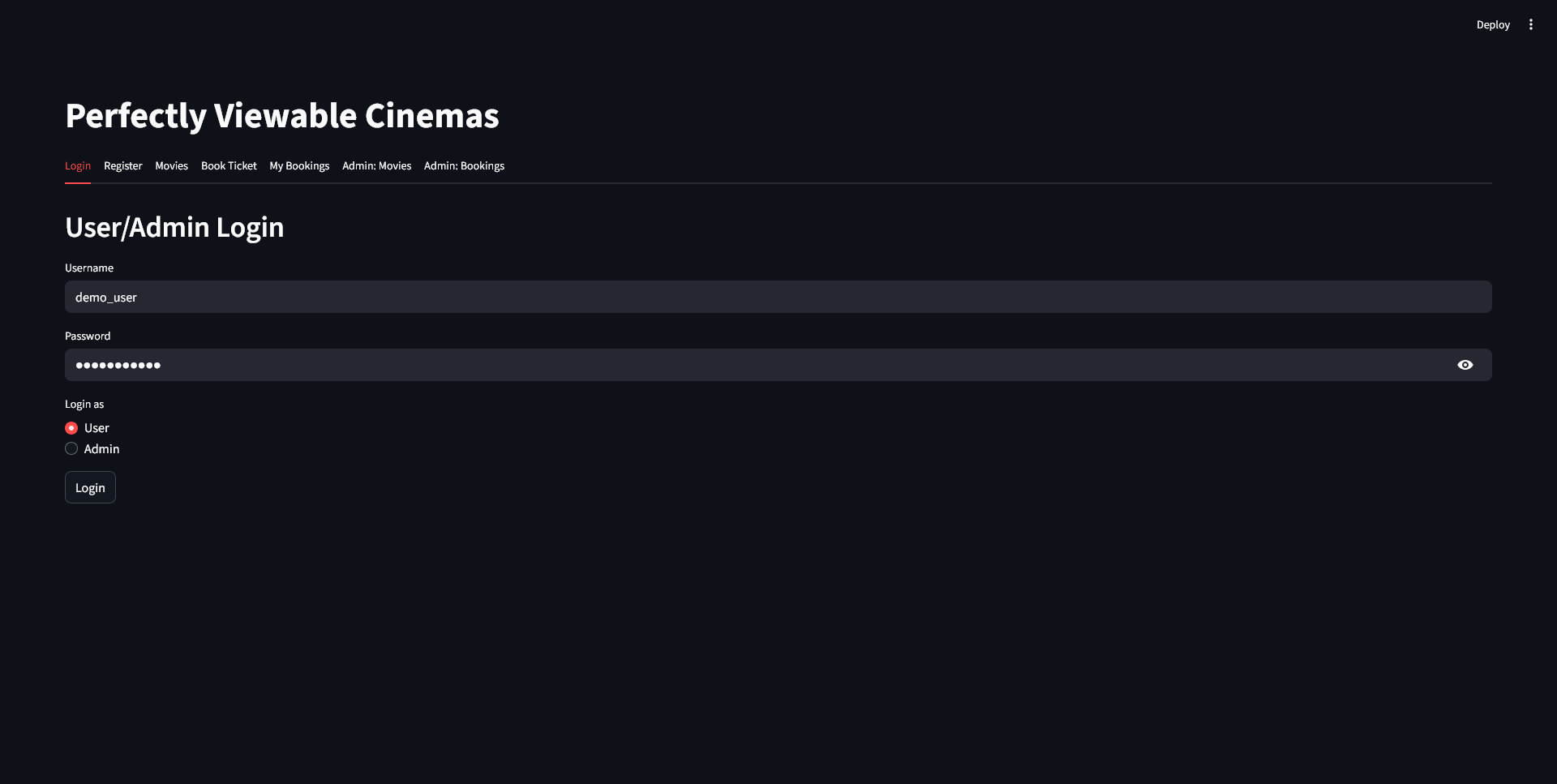
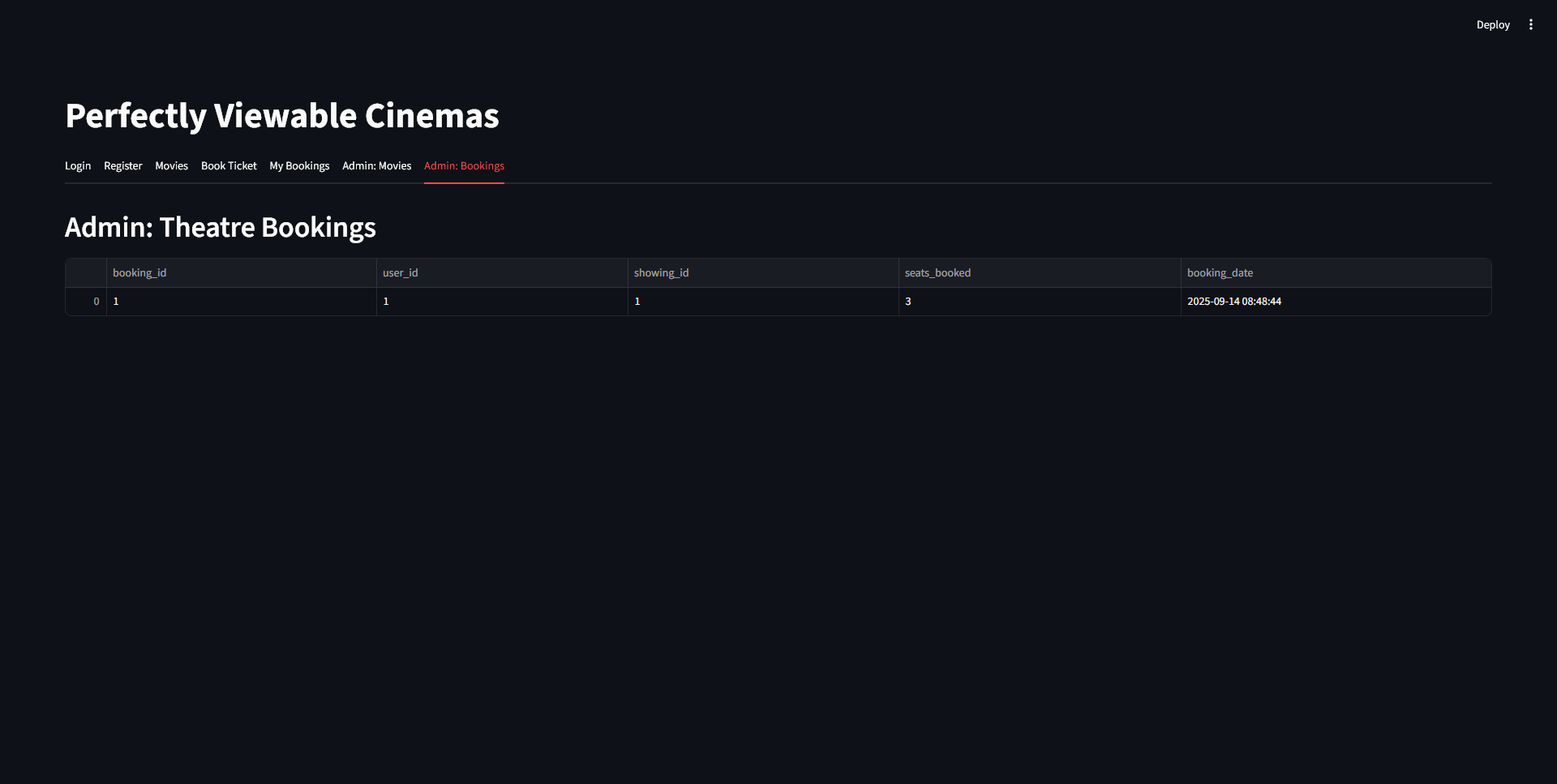
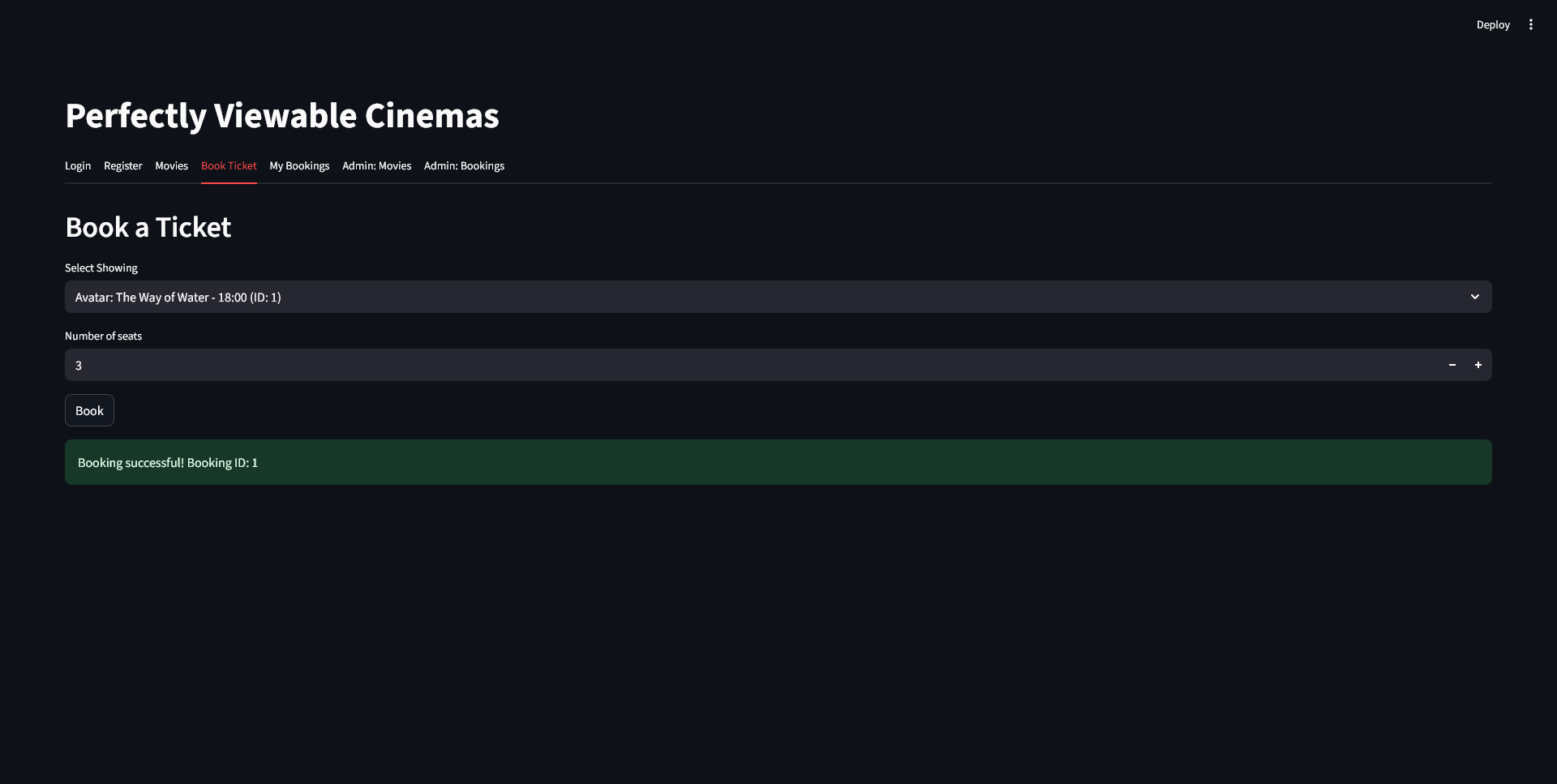
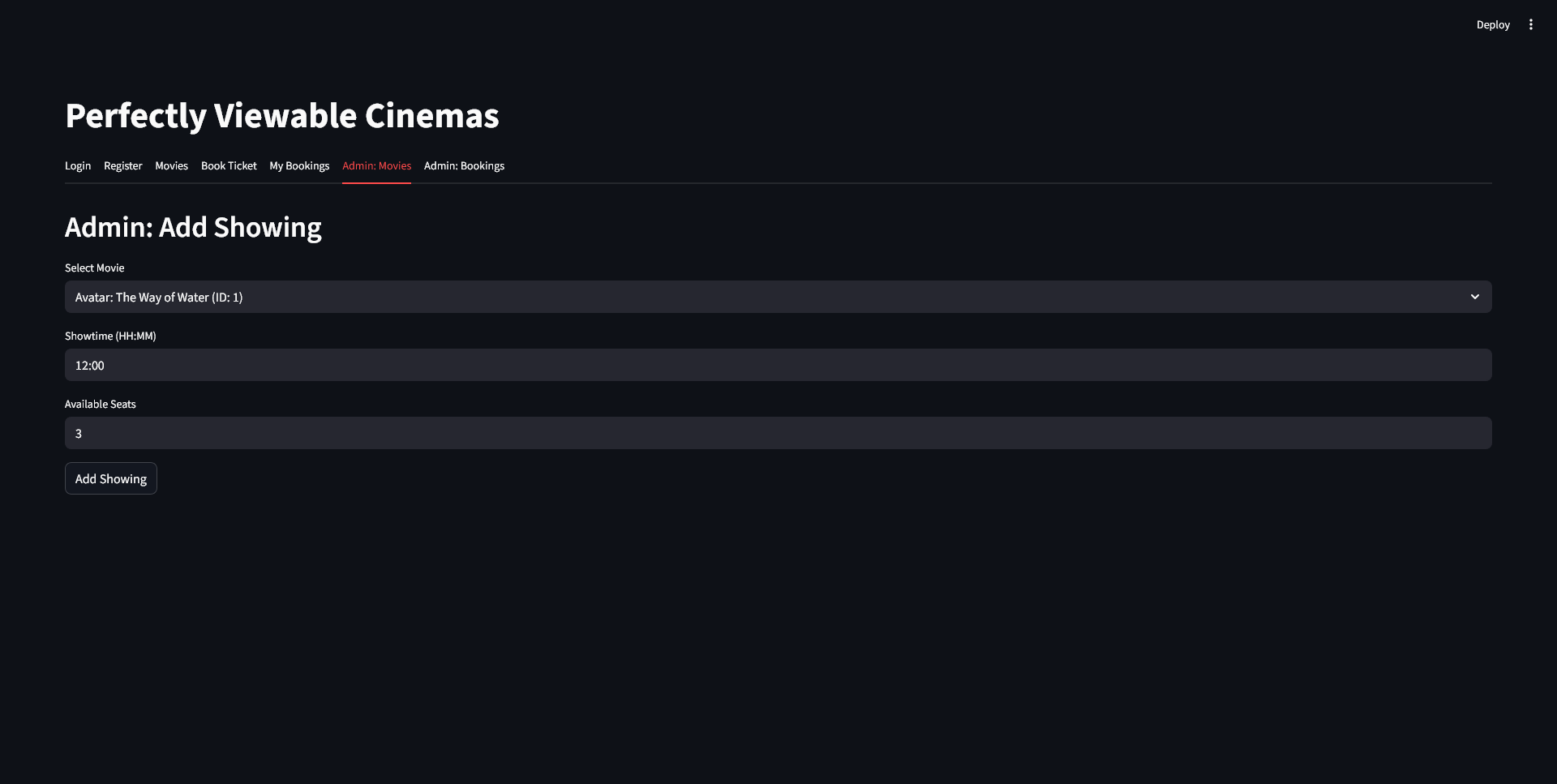
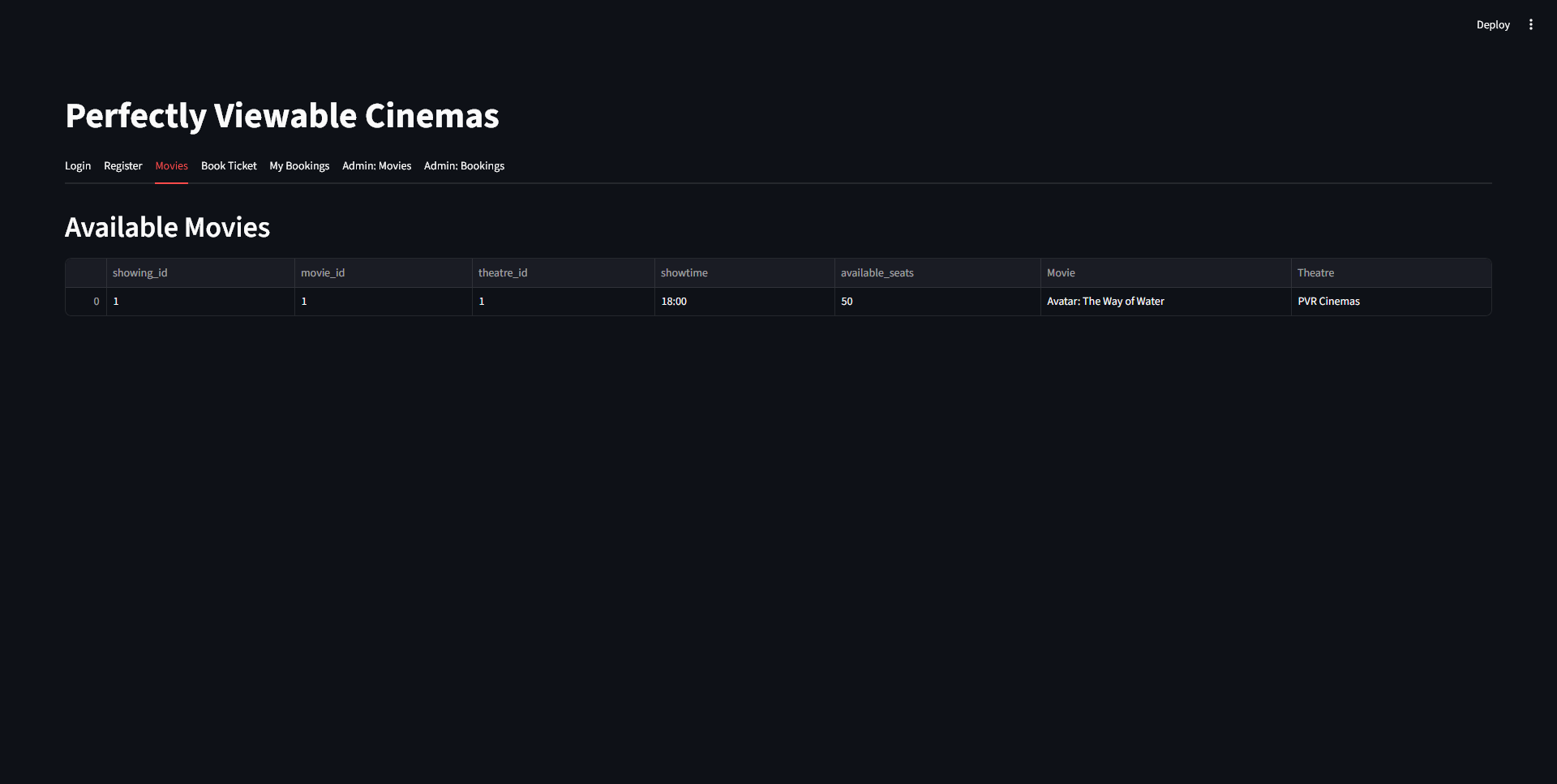
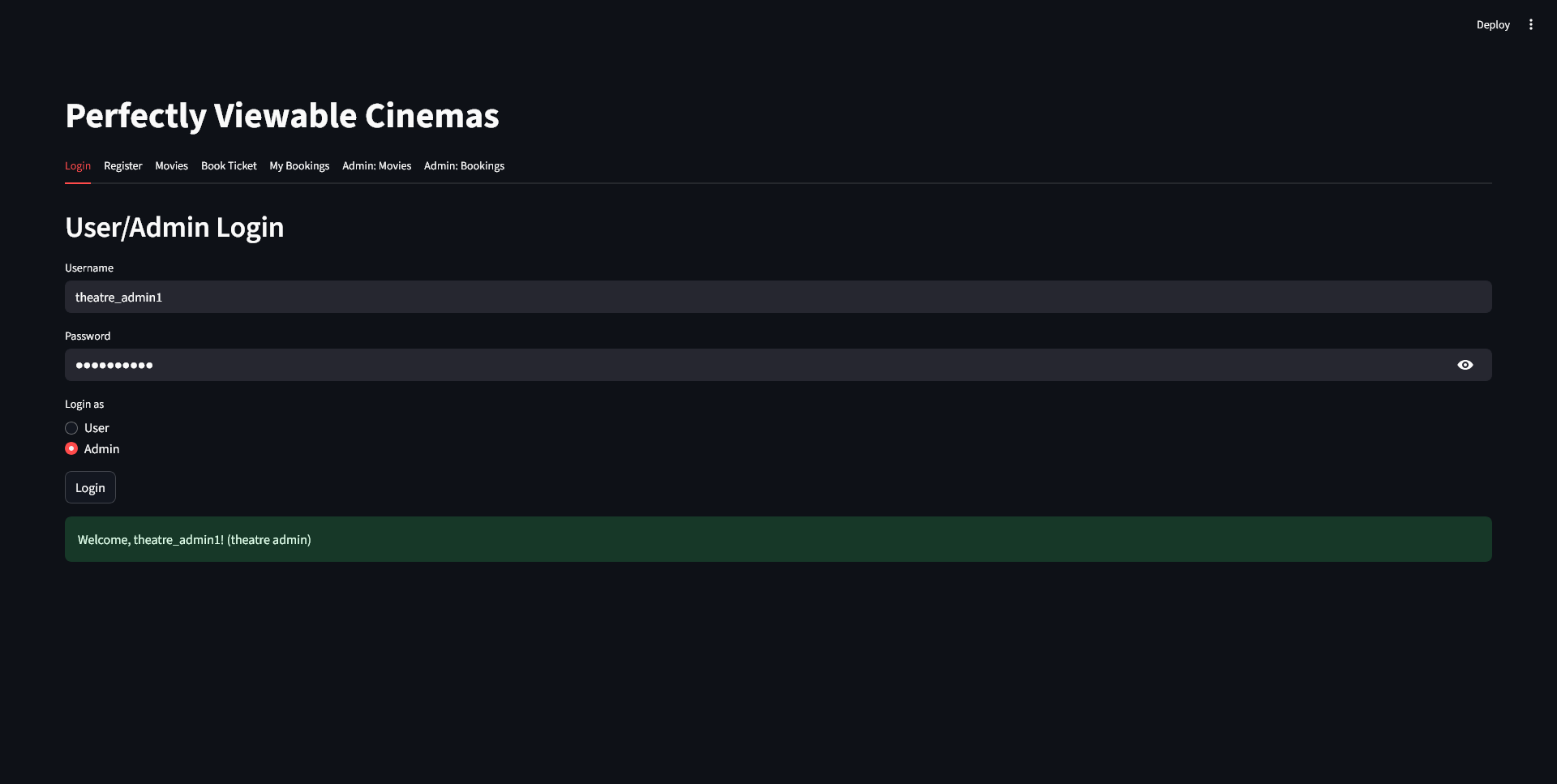
3. Admin Login

4. Exit

Enter your choice (1-4): 4

Thank you for using PVC!

gui.py



Bibliography

- Python documentation: https://docs.python.org/3/  
- Streamlit documentation: https://docs.streamlit.io/  
- Pandas documentation: https://pandas.pydata.org/docs/  
- W3Schools: https://www.w3schools.com/python/  
- GeeksforGeeks: https://www.geeksforgeeks.org/python-programming-language

- Preeti Aurora – Computer science with python