Ketan Malik
NUID – 001378783
Web Development Tools & Methods

"BookAnEvent – Book your next event through us"

Overview

The project aims at providing an online marketplace, *BookAnEvent*, for finding and booking events such as movies, plays, and musicals using this web application. It would help customers to browse events happening near them and then book tickets for the same. Furthermore, it also provides a platform for admins to add and manage events, venues and shows for the customers to book. Some of the key features of the project are:

1. Events that can be booked are – movies, plays, and musical shows
2. Customers can search for events based on their city
3. Admins would perform all the CRUD operations on the events being offered
4. Admins would perform all the CRUD operations on the venues being offered
5. Admins would create shows for different events at different times for a particular venue

Supported Roles & Functionalities

Role 1 – Customer

1. Ability to Register and Login
2. Browse events based on the city and event name
3. Book a particular event
4. View past and upcoming bookings
5. Ability to cancel the booking (only if the show date is atleast after three days from current date)
6. Ability to download a PDF document for the booking

Role 2 – Admin

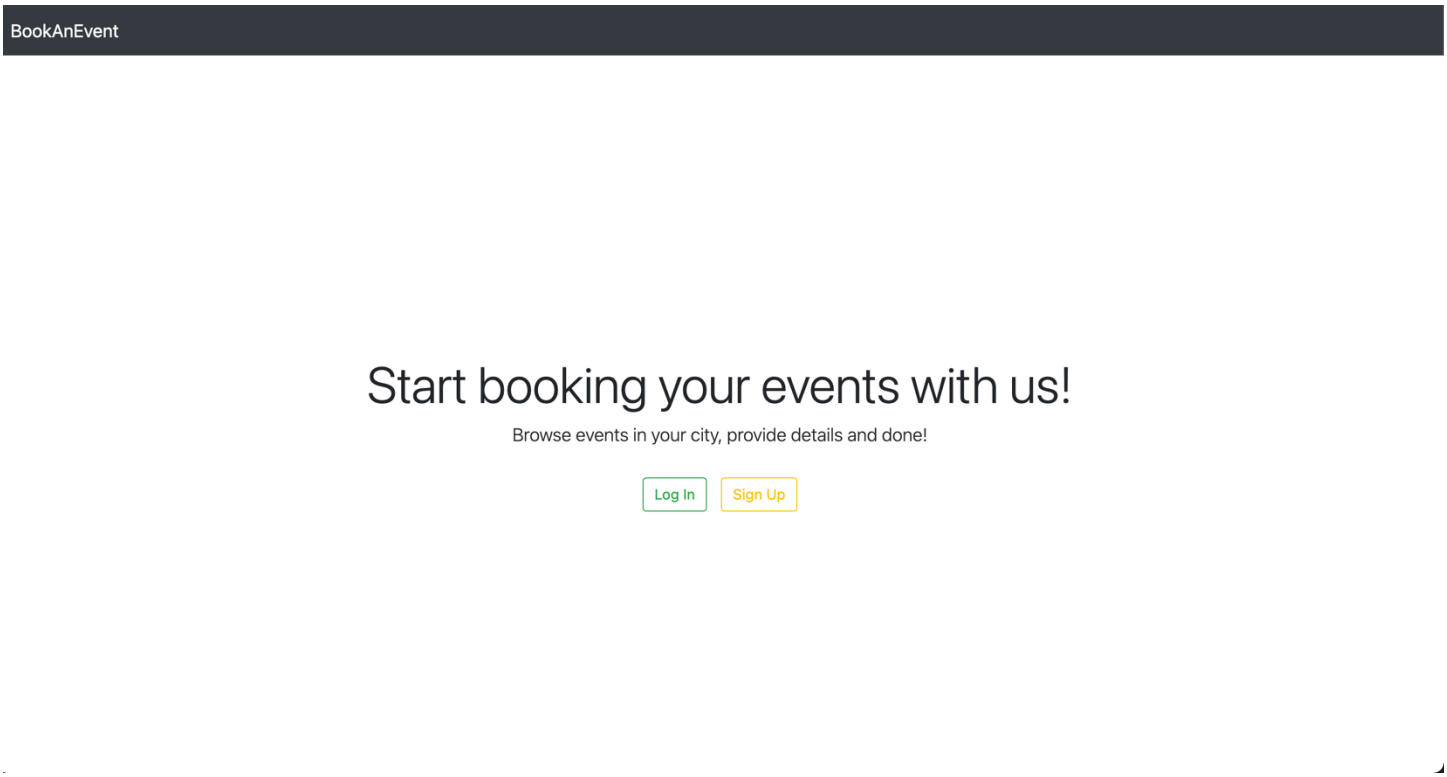
1. Create new events in the system
2. Modify or Delete existing events
3. Create new venues in the system
4. Modify or Delete existing venues
5. Create new shows for a particular event and venue in the system

Managing session integrity – If suppose there are 5 seats left for a particular show and two different customers are trying to **book those 5 seats at the same time** then the customer to book first will get a confirmed booking and the second customer will get an error saying the seats were filled up.

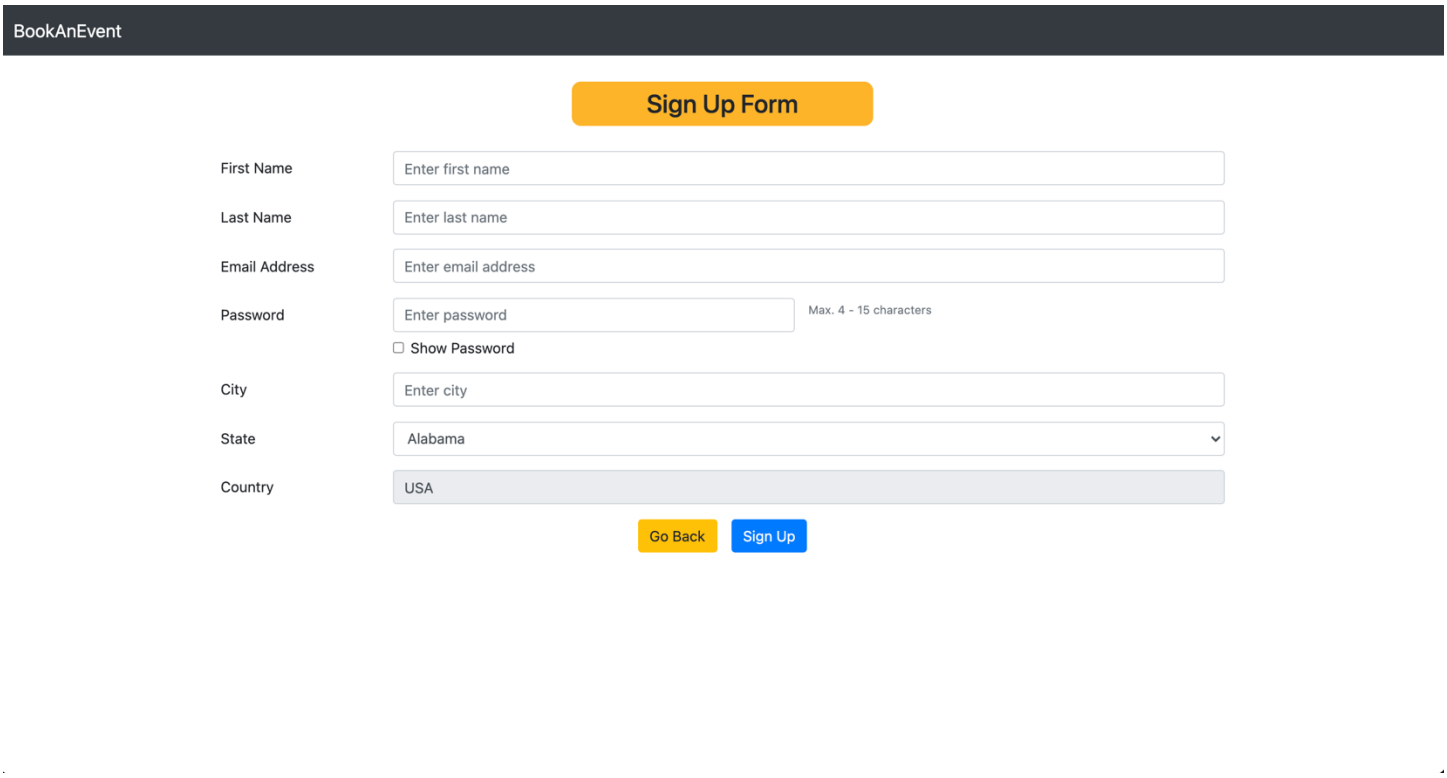
Technology

1. Frontend – HTML, CSS, JavaScript, Ajax, JSP, Bootstrap
2. Backend – Spring MVC, Hibernate, Spring Forms & Validators, HQL
3. Database – MySQL

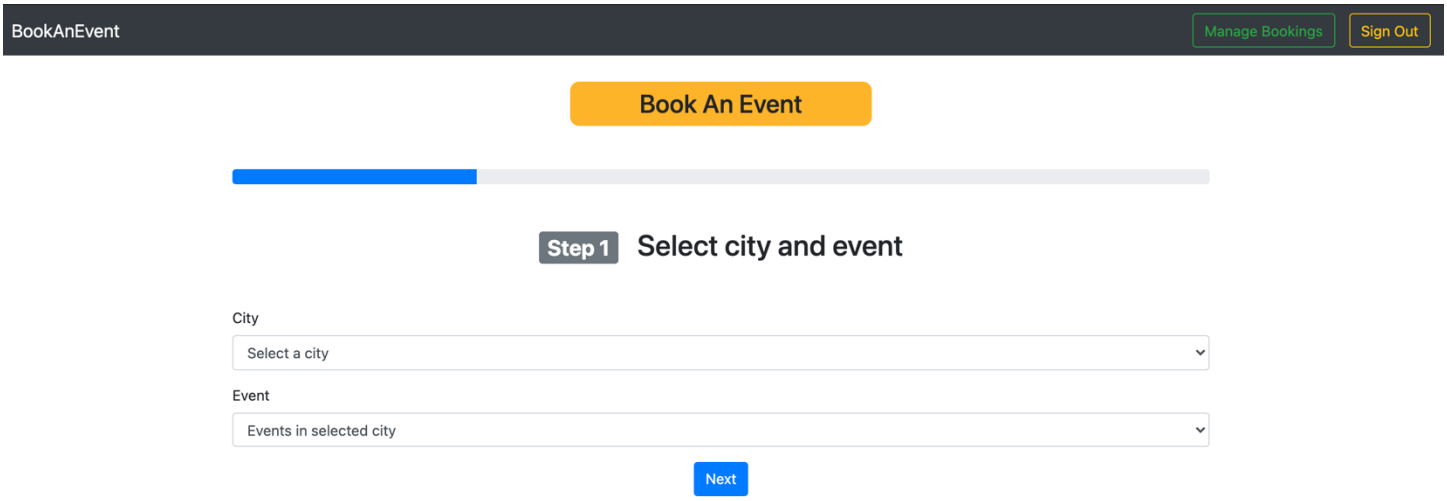
Home Page



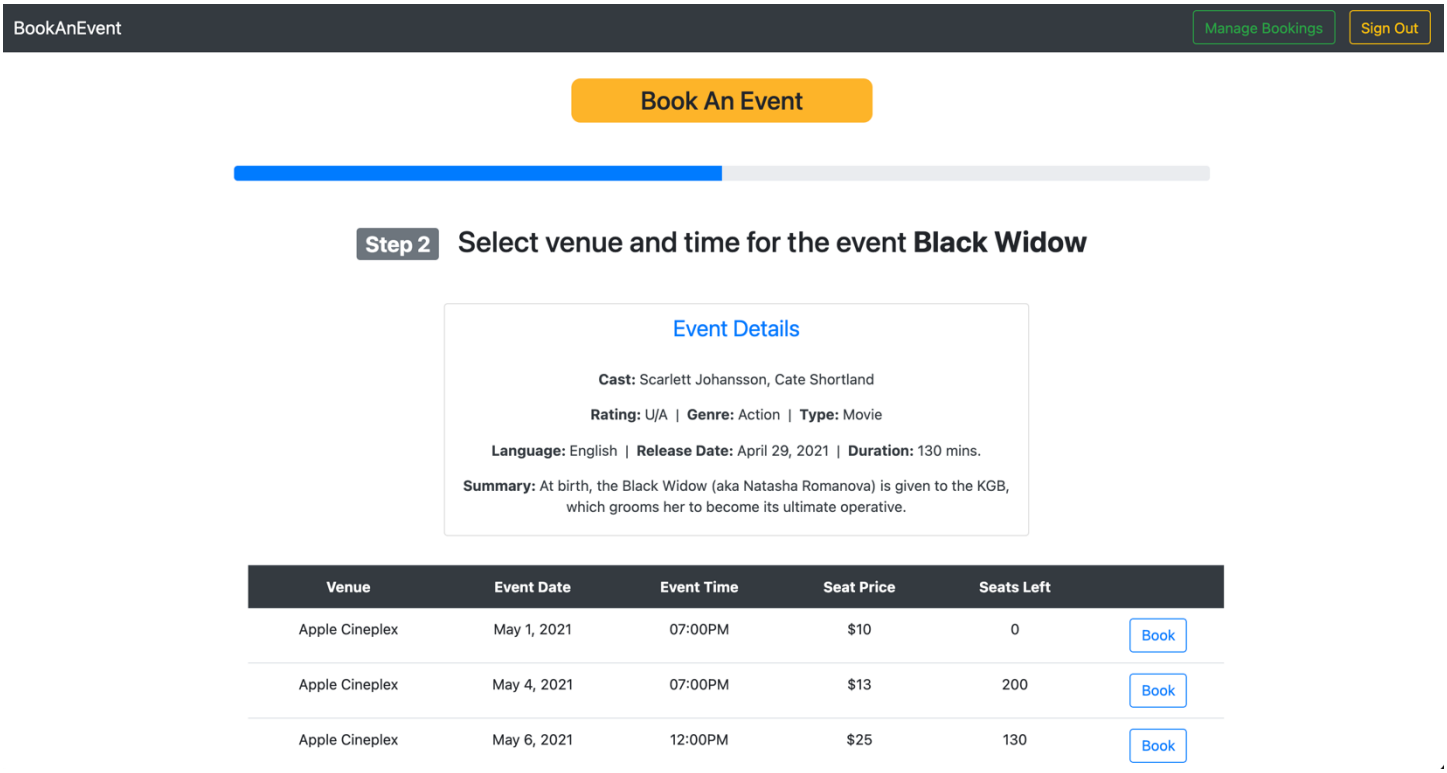
User Sign Up Form



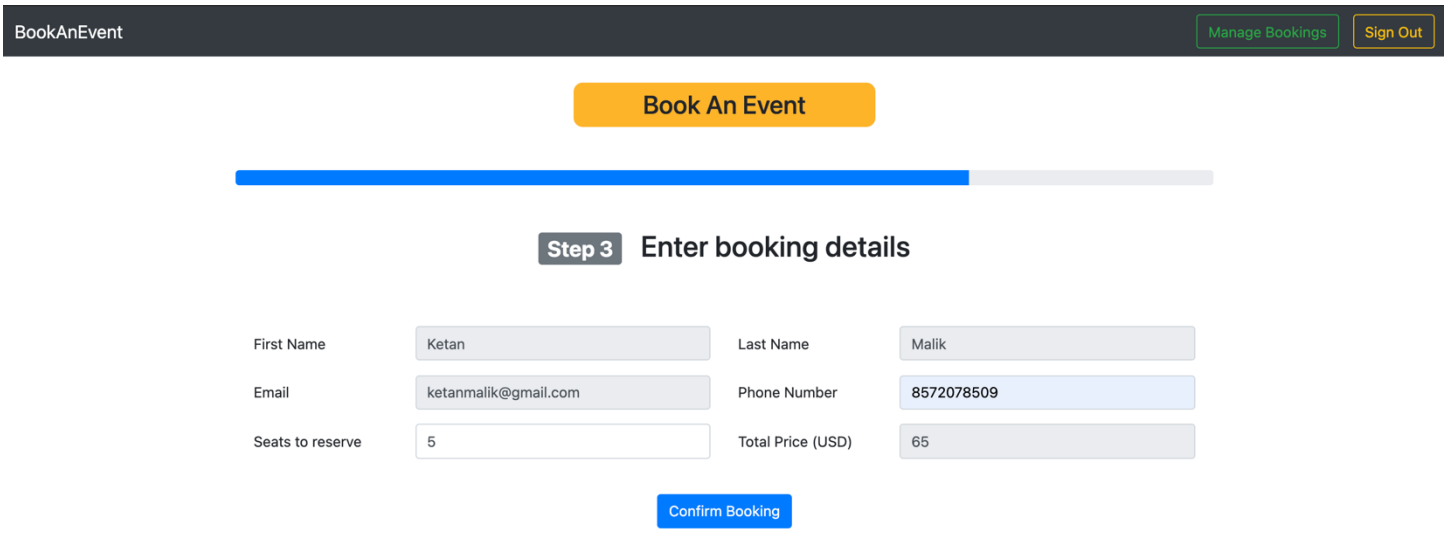
User Flow – Choose city and event



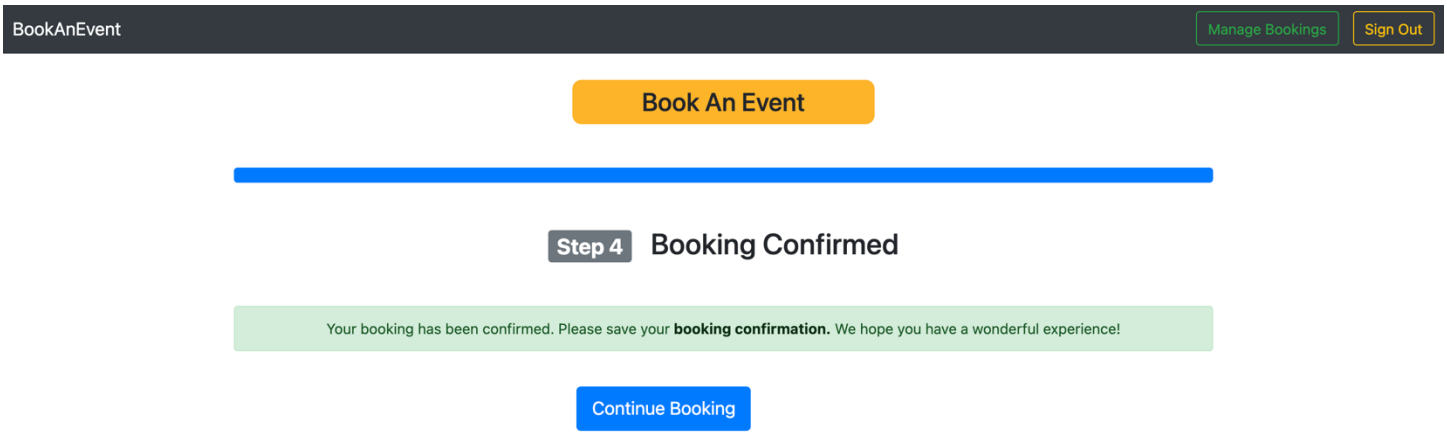
User Flow - Select show time at a particular venue



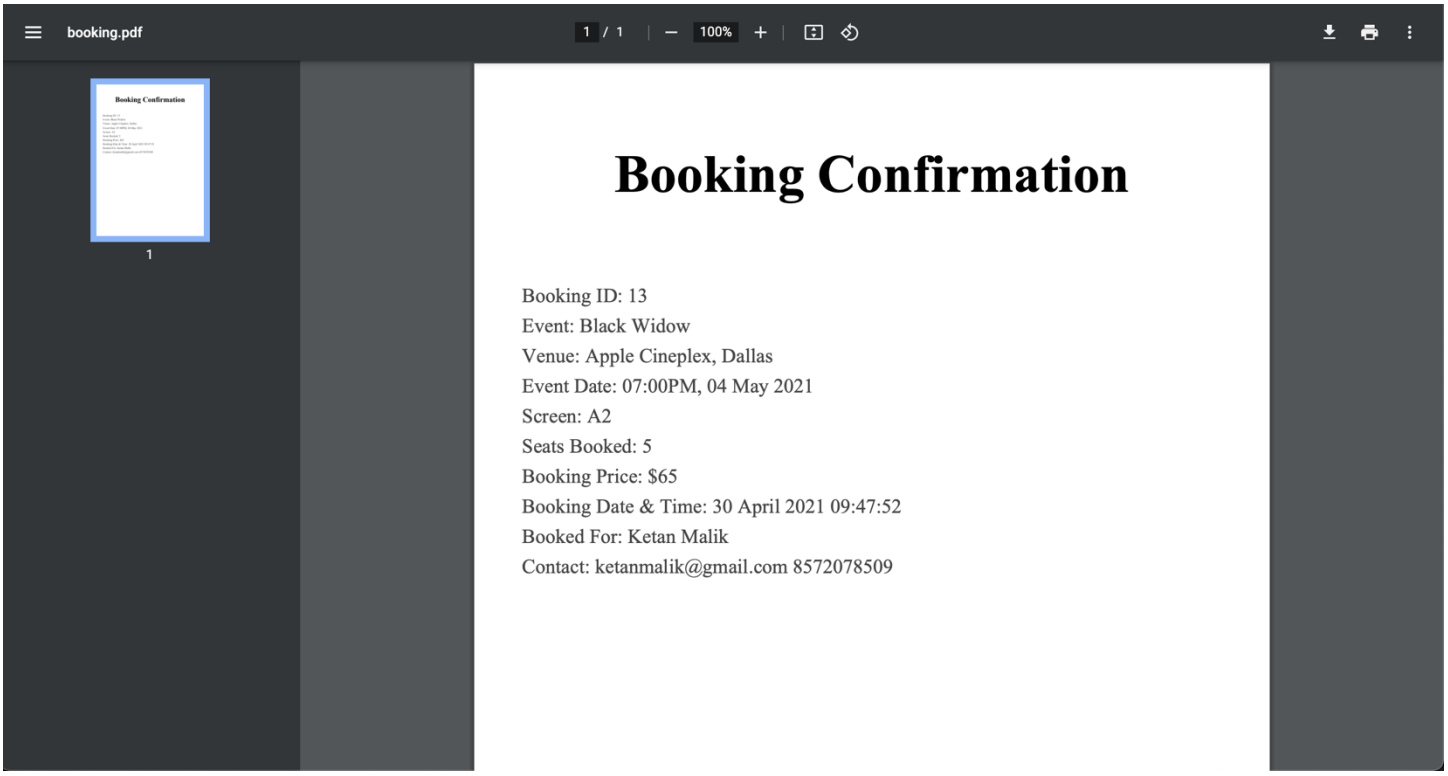
User Flow – Enter total seats and phone number for booking



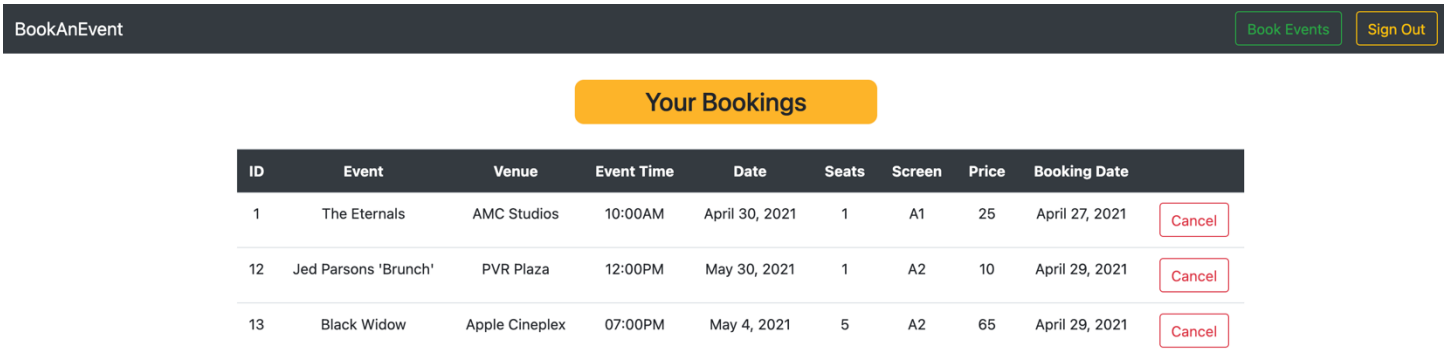
User Flow – Booking Confirmation



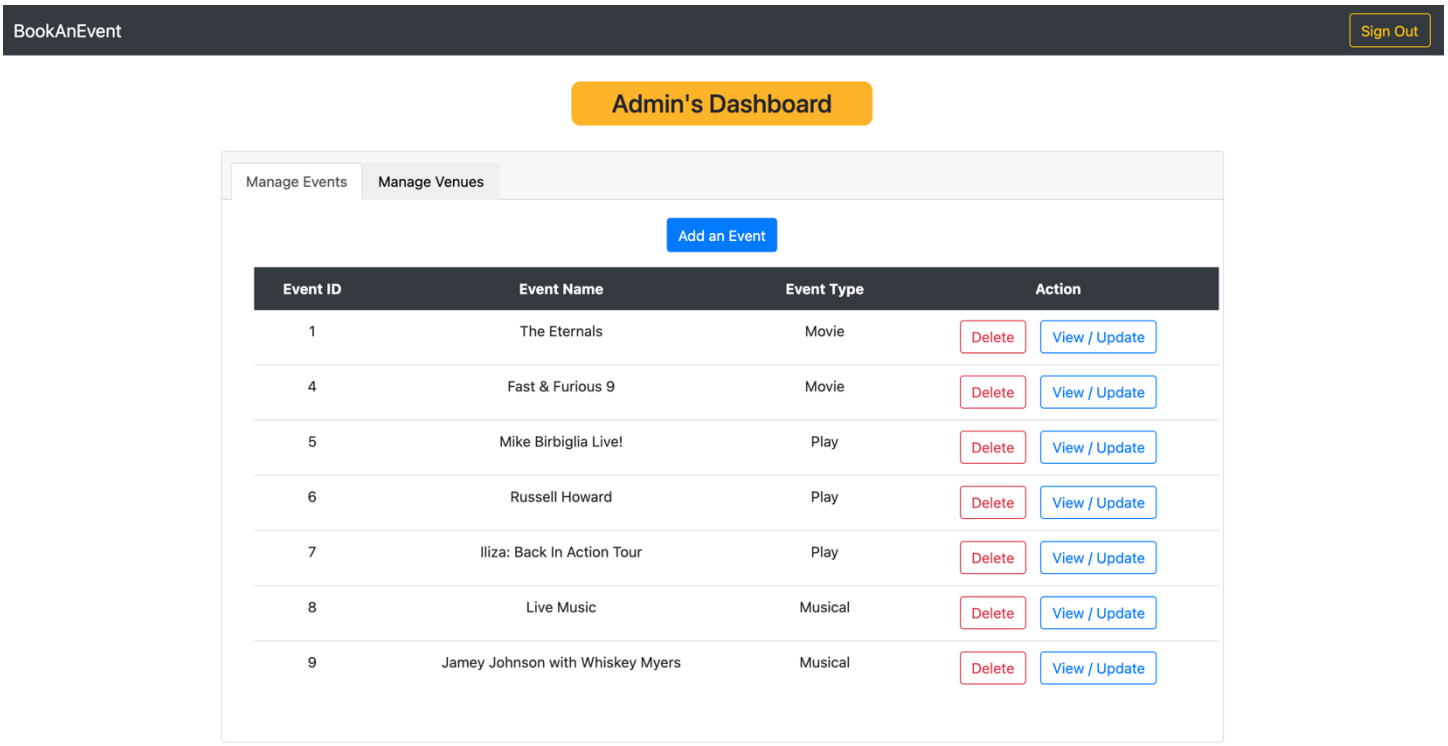
User Flow – Booking Confirmation PDF



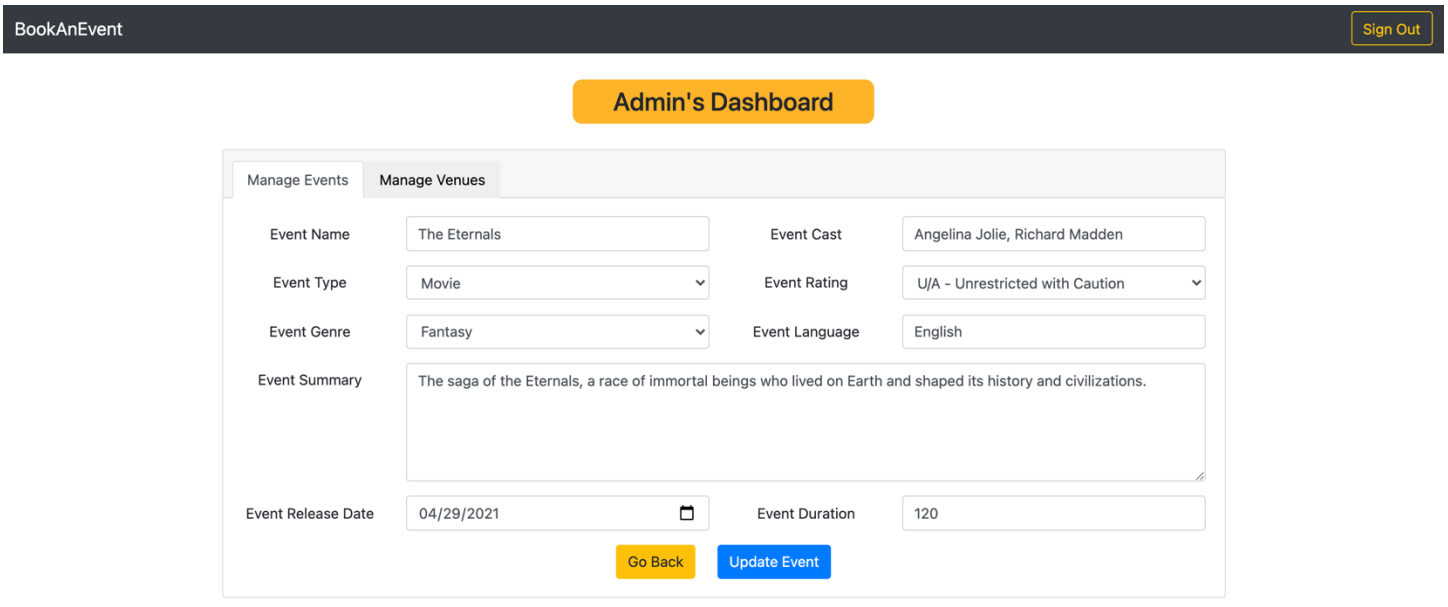
User Flow – View and cancel bookings



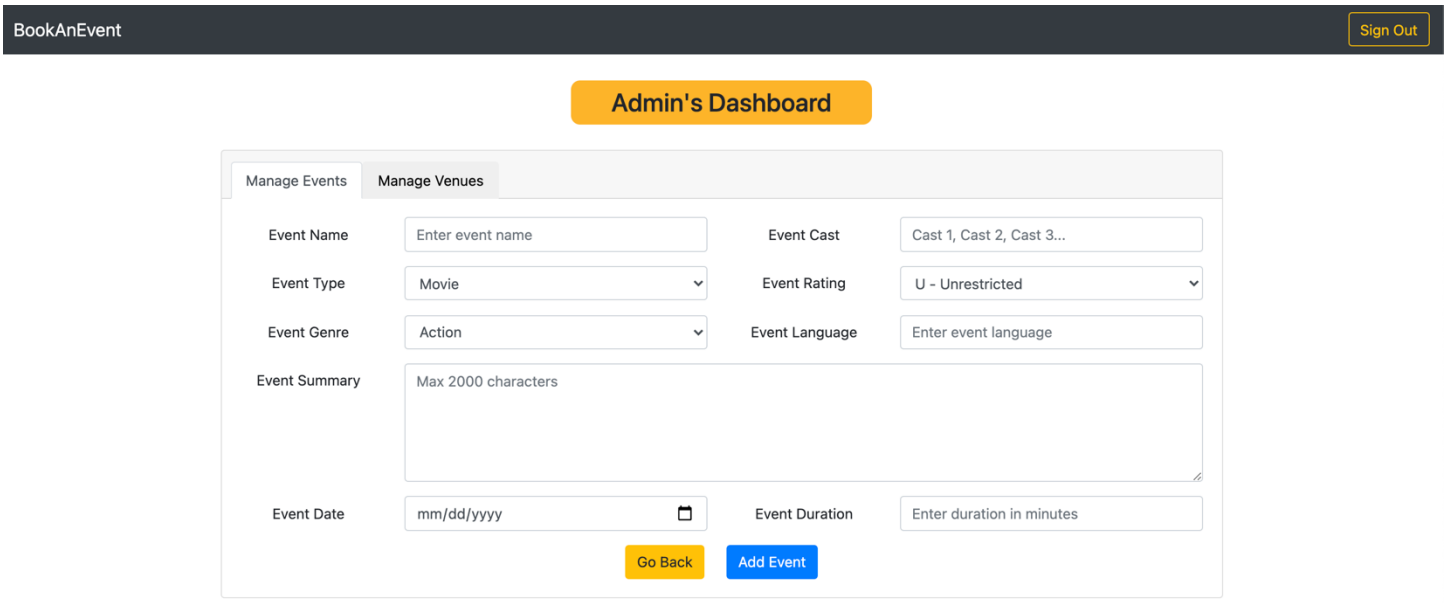
Admin Flow – View all the events



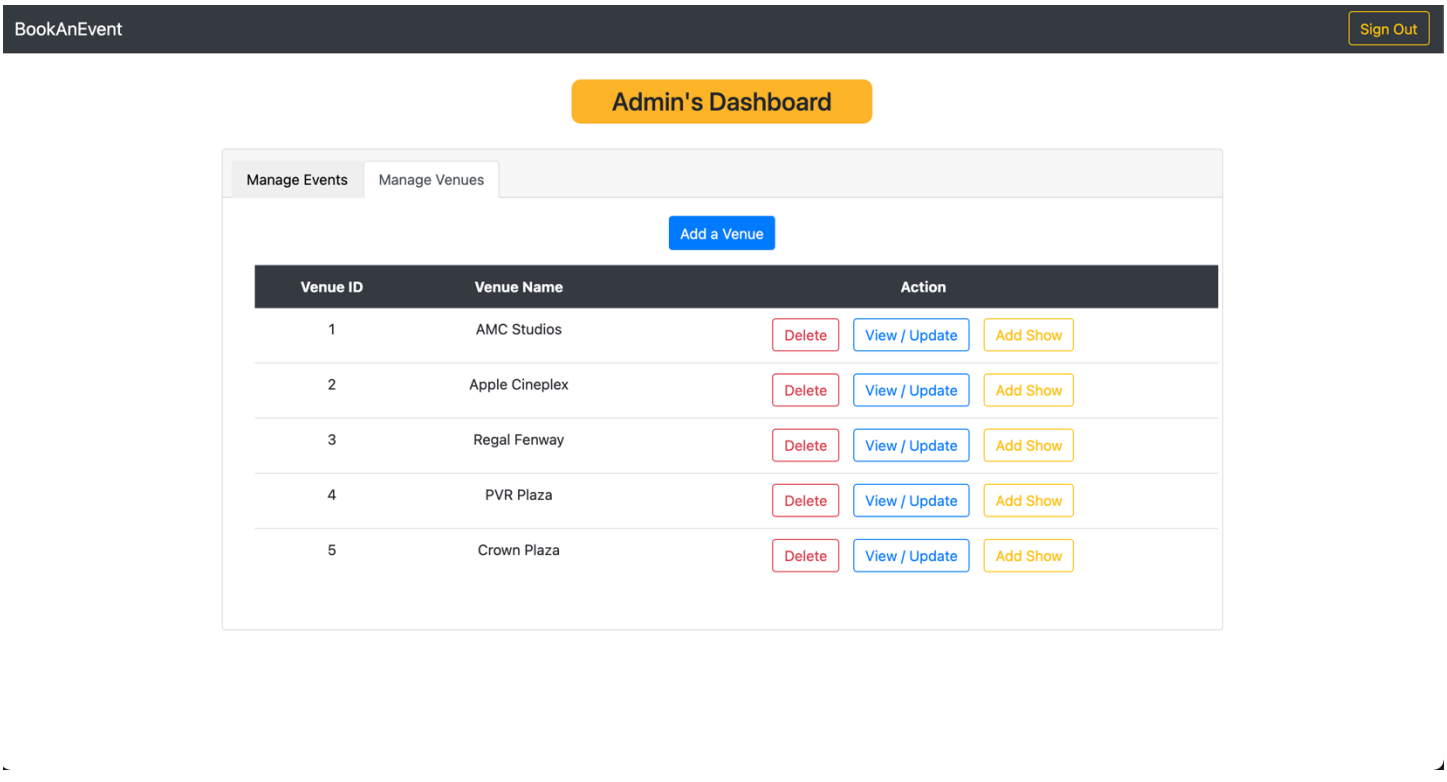
Admin Flow – Updating an existing event



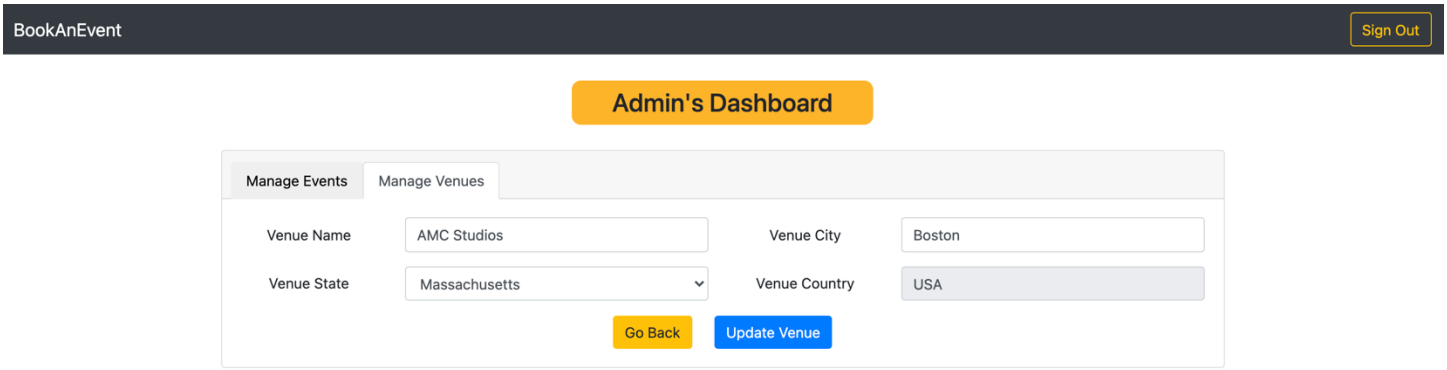
Admin Flow – Adding a new event



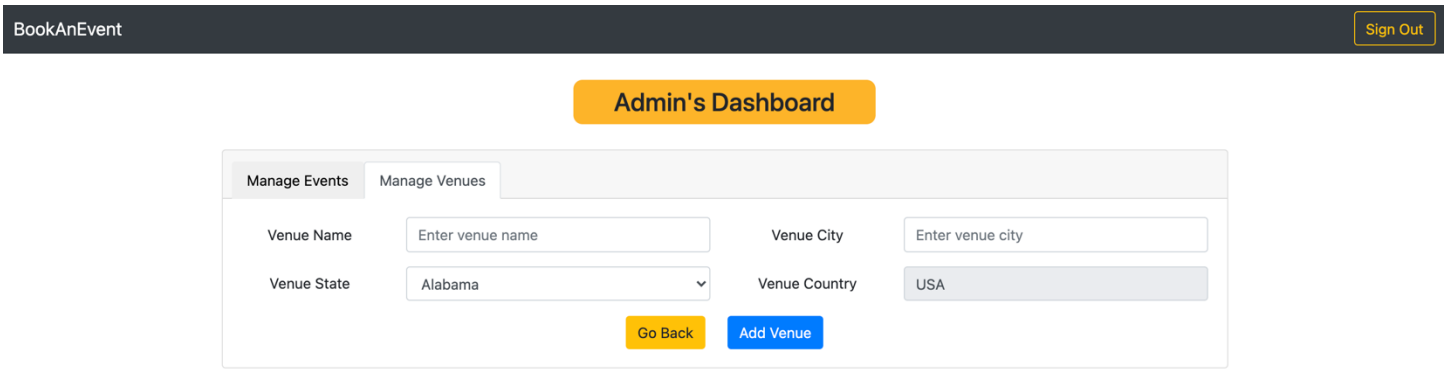
Admin Flow – View all the venues



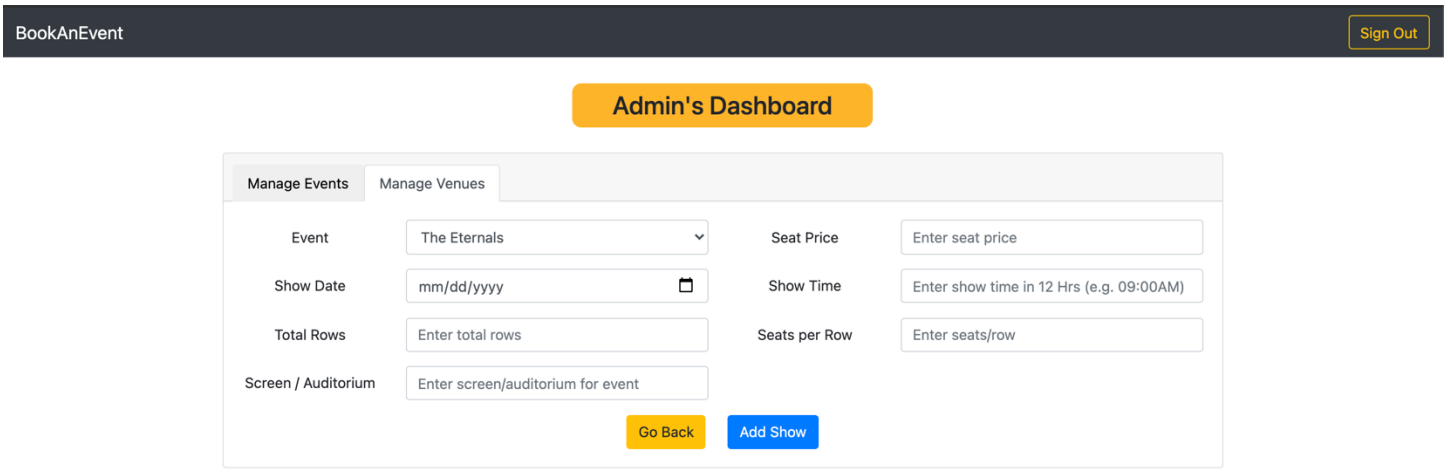
Admin Flow – Updating an existing venue



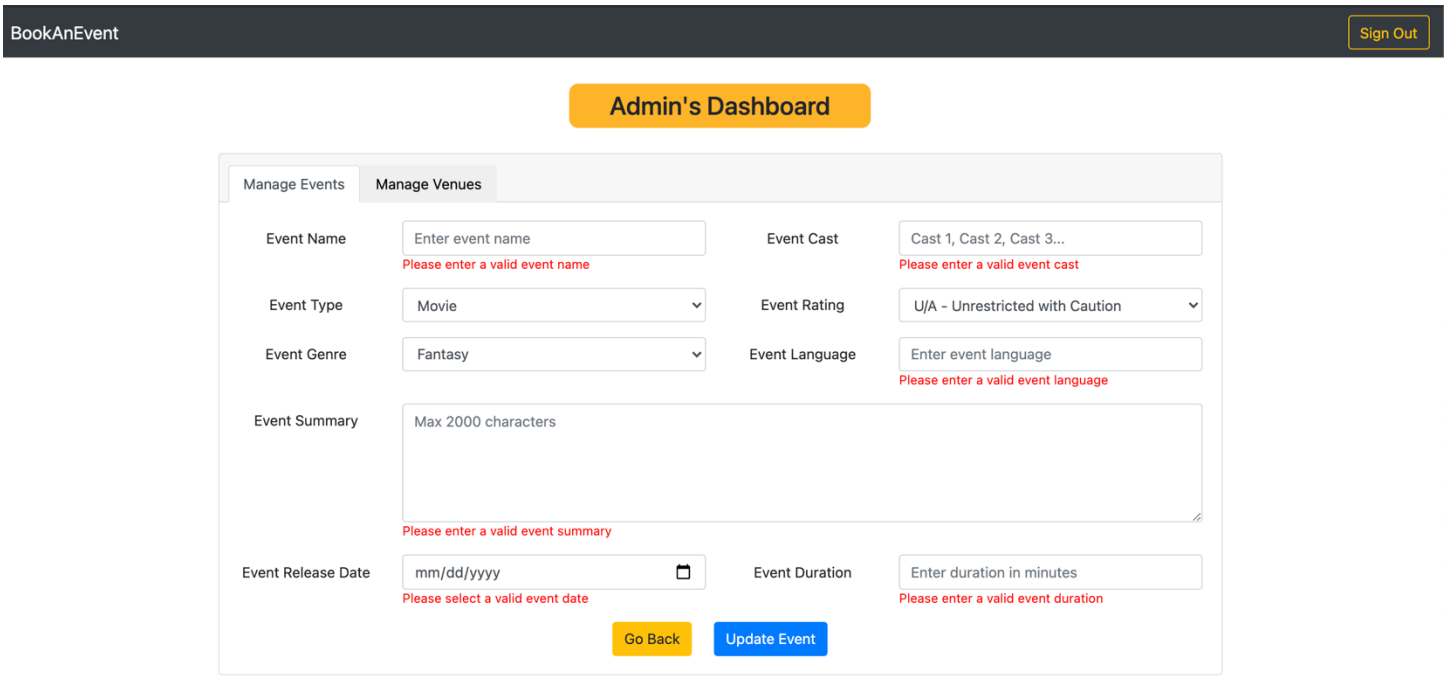
Admin Flow – Adding a new venue



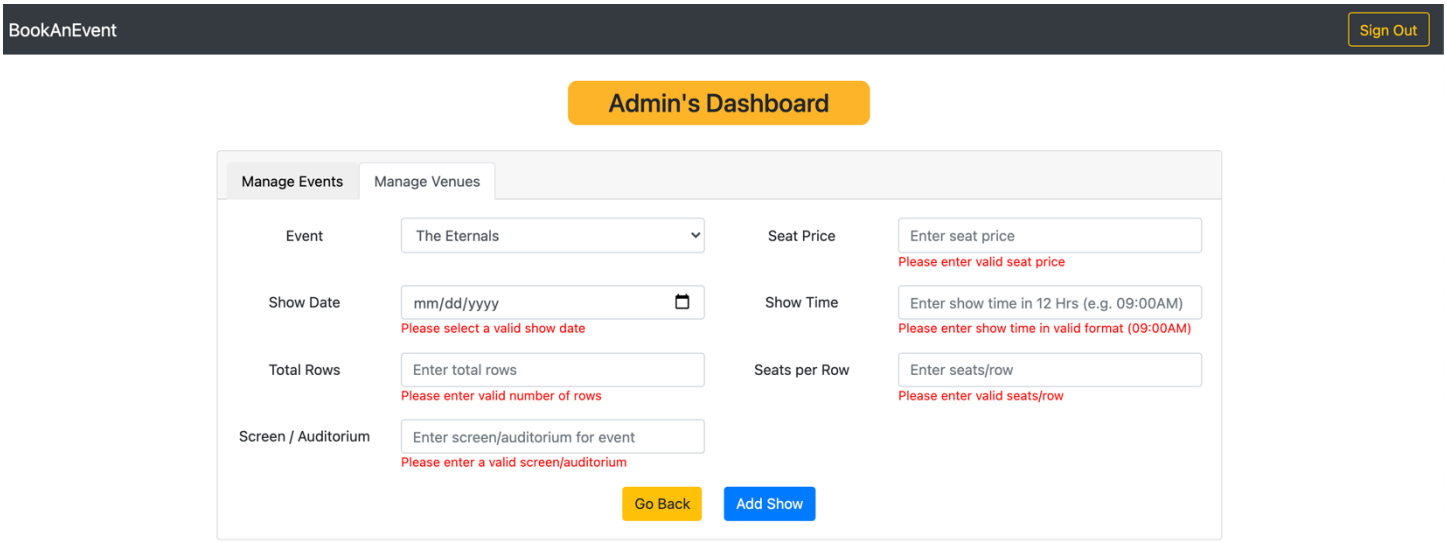
Admin Flow – Adding a new show at a particular venue for a particular event



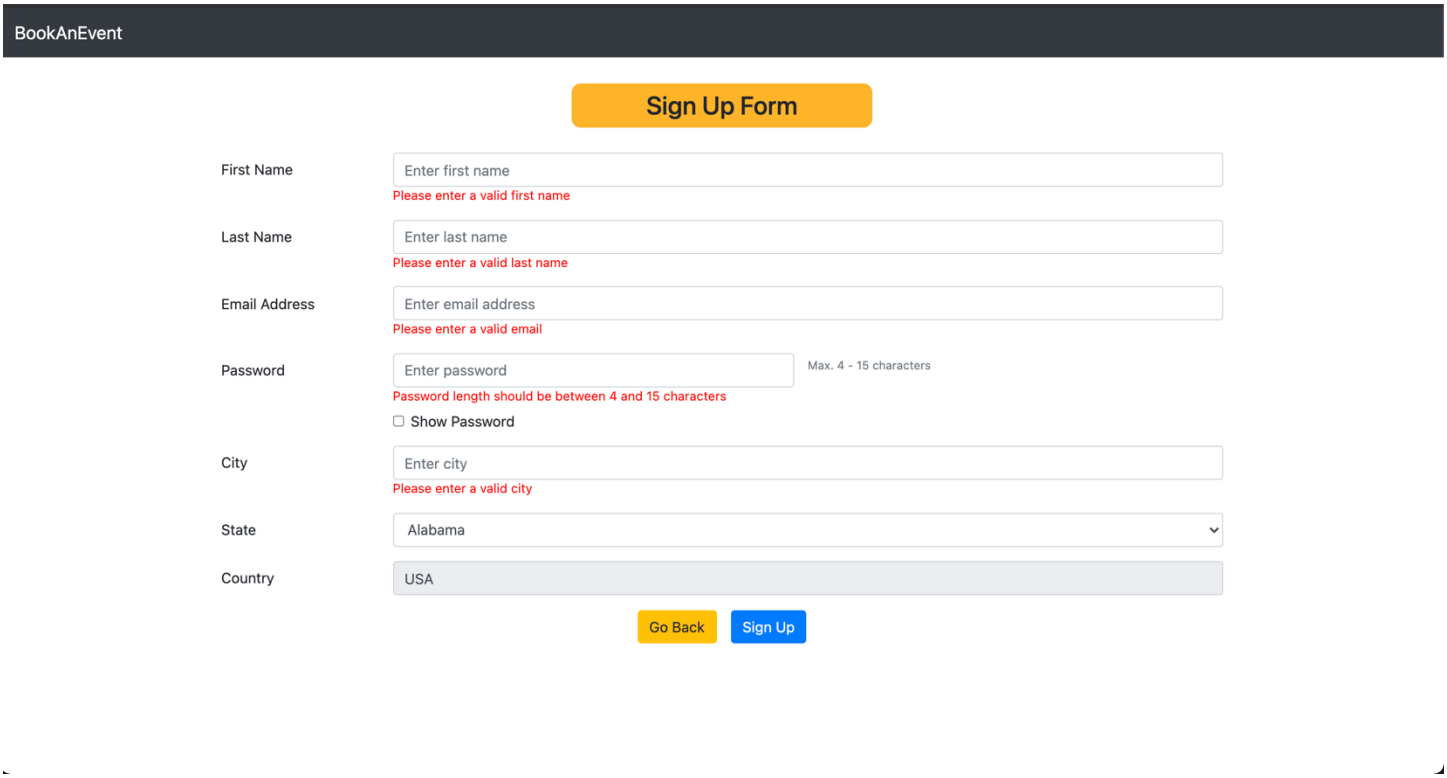
Validations – While updating an event



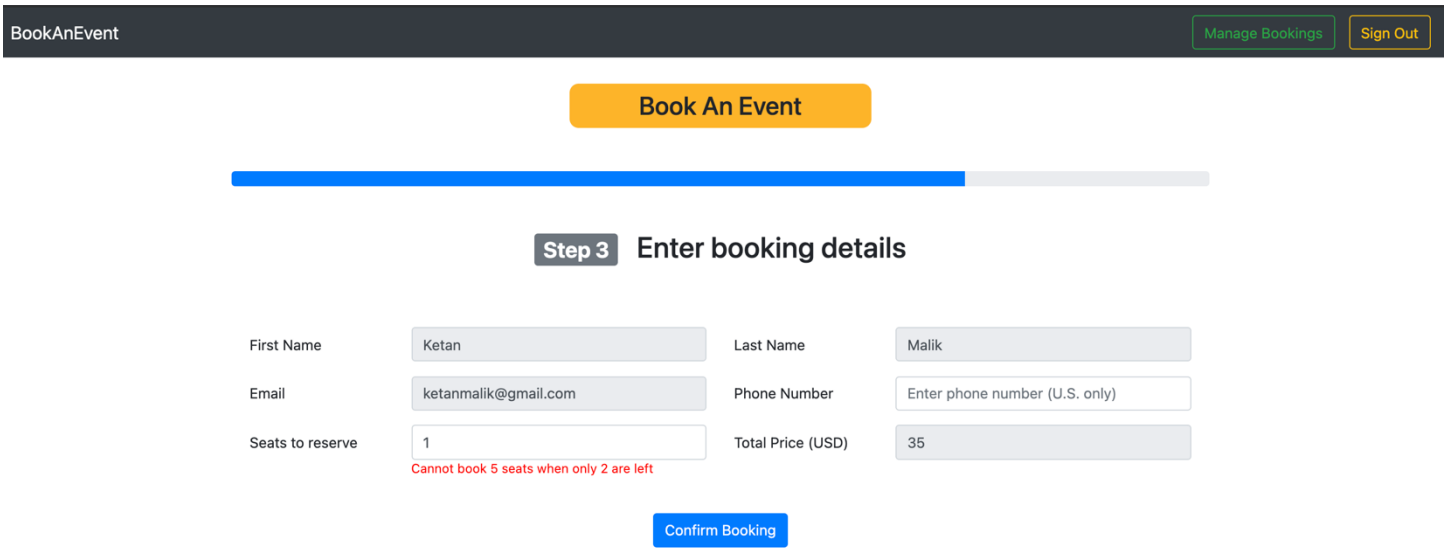
Validations – While adding a new show



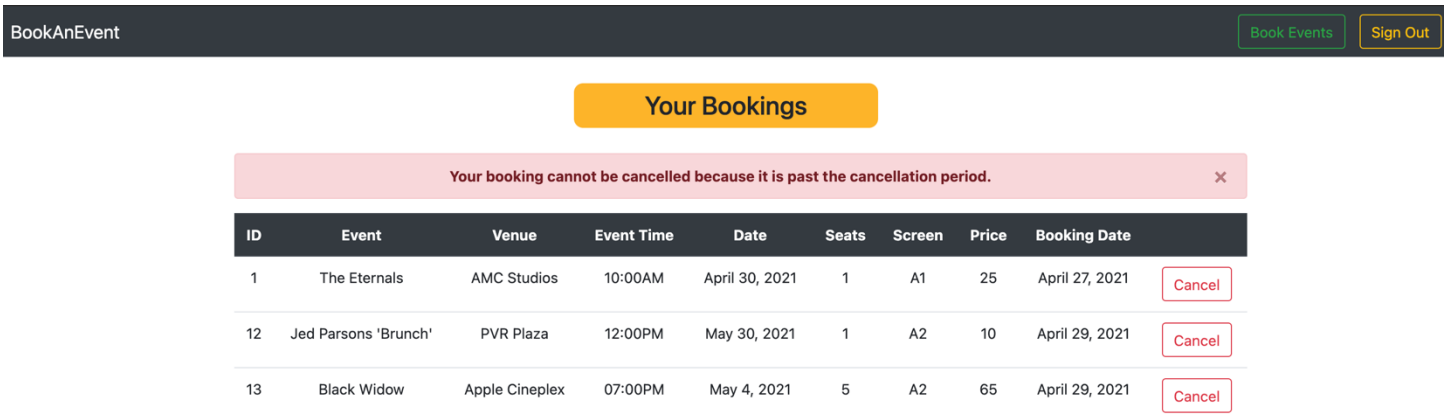
Validations – While signing up as a new user



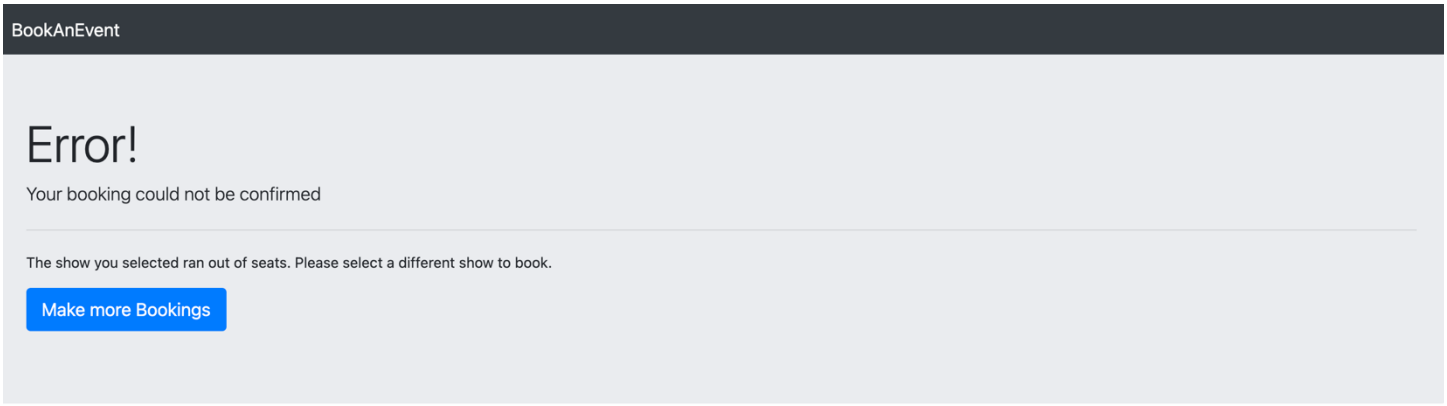
Validations – While trying to book more than available number of seats



Validations – Trying to cancel a booking which is within next 3 days



Validations – While two customers trying to book the same show



AdminEventController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.controller;

import com.mycompany.dao.EventDao;
import com.mycompany.pojo.Event;
import com.mycompany.validator.UpdateEventValidator;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.InitBinder;
```

```
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.servlet.ModelAndView;

/**
 *
 * @author ketanmalik
 */
@Controller
public class AdminEventController {

   @Autowired
   UpdateEventValidator updateEventValidator;

   @InitBinder("event")
   protected void initBinder(WebDataBinder binder) {
      binder.setValidator(updateEventValidator);
   }

   @PostMapping("/add-event.htm")
   public ModelAndView addEvent(HttpServletRequest request, HttpServletResponse response, HttpSession
session) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
      return new ModelAndView("add-event-view");
   }

   @PostMapping("/delete-event.htm")
   public ModelAndView deleteEvent(HttpServletRequest request, HttpServletResponse response, HttpSession
session, EventDao eventDao) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
      int res = eventDao.deleteEvent(Integer.parseInt(request.getParameter("delete-event")));
      if (res > 0) {
         List<Event> eventList = eventDao.getEventList();
         session.setAttribute("eventList", eventList);
      }
      return new ModelAndView("manage-events-view");
   }

   @PostMapping("event-added.htm")
   public ModelAndView saveEvent(HttpServletRequest request, HttpServletResponse response, HttpSession
session, EventDao eventDao) throws ParseException {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
```

```
        String event_name = request.getParameter("event-name");
        String event_type = request.getParameter("event-type");
        String event_cast = request.getParameter("event-cast");
        String event_rating = request.getParameter("event-rating");
        String event_genre = request.getParameter("event-genre");
        String event_language = request.getParameter("event-language");
        String event_summary = request.getParameter("event-summary");
        String event_duration = request.getParameter("event-duration");
        String event_date = request.getParameter("event-date");

        if (event_name.trim().equals("") || event_type.trim().equals("") || event_cast.trim().equals("") ||
event_rating.trim().equals("") || event_genre.trim().equals("")
                || event_language.trim().equals("") || event_summary.trim().equals("") || event_summary.length()
> 2000 || event_duration.trim().equals("") || event_date.trim().equals("")) {

            request.setAttribute("errorMsg1", "New event could not be added.");
            request.setAttribute("errorMsg2", "Some of the fields you entered were invalid. Please try again.");
            return new ModelAndView("error-view");
        }

        SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
        int res = eventDao.saveEvent(event_name, event_type, event_cast, event_rating, event_genre,
                event_language, event_summary, event_duration, formatter.parse(event_date));
        if (res > 0) {
            List<Event> eventList = eventDao.getEventList();
            session.setAttribute("eventList", eventList);
            request.setAttribute("successMsg1", "New event added successfully.");
            request.setAttribute("successMsg2", "Please go back to dashboard to manage this newly added
event.");
            return new ModelAndView("success-view");
        }
        return serverError(request, "New event could not be added.");
    }

    @PostMapping("/updated-event.htm")
    public ModelAndView saveUpdatedEvent(@ModelAttribute("updateEventForm") Event event,
BindingResult result, Model model, HttpServletRequest request, HttpSession session, EventDao eventDao) {
        updateEventValidator.validate(event, result);
        if (result.hasErrors()) {
            return new ModelAndView("update-event-view");
        } else {
            Event sessionEvent = (Event) session.getAttribute("event");
            int res = eventDao.updateEvent(sessionEvent.getEvent_id(), event);
            if (res == 1) {
                event.setEvent_id(sessionEvent.getEvent_id());
                session.removeAttribute("event");
                session.setAttribute("event", event);
                Event obj = (Event) session.getAttribute("event");
                request.setAttribute("successMsg1", "Event updated successfully.");
```

```
            request.setAttribute("successMsg2", "Please go back to dashboard to manage events, users, and
venues.");
            return new ModelAndView("success-view");
         } else {
            return serverError(request, "The event could not be updated.");
         }
      }
   }

   @GetMapping("/manage-events.htm")
   public ModelAndView signUpPage(HttpServletRequest request, HttpSession session) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
      return new ModelAndView("manage-events-view");
   }

   @PostMapping("/manage-events.htm")
   public ModelAndView manageEvents(HttpServletRequest request, HttpServletResponse response,
HttpSession session) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
      return new ModelAndView("manage-events-view");
   }

   @PostMapping("/update-event.htm")
   public ModelAndView updateEvent(HttpServletRequest request, HttpServletResponse response,
HttpSession session, EventDao eventDao, Model model) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
      int event_id = Integer.parseInt(request.getParameter("update-event"));
      Event event = eventDao.getEvent(event_id);
      if (event == null) {
         if (session.getAttribute("event") != null) {
            session.removeAttribute("event");
         }
         return serverError(request, "The event could not be updated.");
      }
      session.setAttribute("event", event);
      Event e = new Event();
      e.setEvent_id(event_id);
      e.setEvent_name(event.getEvent_name());
      e.setEvent_type(event.getEvent_type());
      e.setEvent_cast(event.getEvent_cast());
      e.setEvent_rating(event.getEvent_rating());
      e.setEvent_genre(event.getEvent_genre());
      e.setEvent_language(event.getEvent_language());
```

```
        e.setEvent_summary(event.getEvent_summary());
        e.setEvent_duration(event.getEvent_duration());
        e.setEvent_date(event.getEvent_date());
        model.addAttribute("updateEventForm", e);
        return new ModelAndView("update-event-view");
    }

    public boolean invalidSessionObj(HttpSession session, String obj) {
        return session.getAttribute(obj) == null;
    }

    public ModelAndView sessionTimedOut(HttpServletRequest request) {
        request.setAttribute("errorMsg1", "Your session has been timed out.");
        request.setAttribute("errorMsg2", "Please log in again to start a new session.");
        return new ModelAndView("error-view");
    }

    public ModelAndView serverError(HttpServletRequest request, String msg1) {
        request.setAttribute("errorMsg1", msg1);
        request.setAttribute("errorMsg2", "There was a problem in reaching out to our servers. Please try again
later.");
        return new ModelAndView("error-view");
    }
}
```

AdminShowController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.controller;

import com.mycompany.dao.EventDao;
import com.mycompany.dao.ShowDao;
import com.mycompany.dao.VenueDao;
import com.mycompany.pojo.Event;
import com.mycompany.pojo.Show;
import com.mycompany.pojo.Venue;
import com.mycompany.validator.ShowValidator;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
```

```
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.InitBinder;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.servlet.ModelAndView;

/**
 *
 * @author ketanmalik
 */
@Controller
public class AdminShowController {

   @Autowired
   ShowValidator showValidator;

   @InitBinder("show")
   protected void initBinder(WebDataBinder binder) {
      binder.setValidator(showValidator);
   }

   @PostMapping("/add-show.htm")
   public ModelAndView addShow(HttpServletRequest request, HttpSession session, Model model) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
      Show show = new Show();
      model.addAttribute("addShowForm", show);
      session.setAttribute("venue-id", request.getParameter("add-show"));
      return new ModelAndView("add-show-view");
   }

   @PostMapping("/show-added.htm")
   public ModelAndView saveShow(@ModelAttribute("addShowForm") Show show, BindingResult result,
HttpServletRequest request, HttpSession session, ShowDao showDao, EventDao eventDao, VenueDao
venueDao) {
      showValidator.validate(show, result);
      if (result.hasErrors()) {
         return new ModelAndView("add-show-view");
      } else {
         int event_id = Integer.parseInt(request.getParameter("event-name"));
         int venue_id = Integer.parseInt(session.getAttribute("venue-id") + "");
         Event event = eventDao.getEvent(event_id);
         Venue venue = venueDao.getVenue(venue_id);
         showValidator.validateCustom(result, event, show);
         if (result.hasErrors()) {
            return new ModelAndView("add-show-view");
         }
```

```
            show.setEvent(event);
            show.setVenue(venue);
            int total_rows = show.getTotal_rows();
            int seats_per_row = show.getSeats_per_row();
            int seats_left = total_rows * seats_per_row;
            show.setSeats_left(seats_left);
            session.removeAttribute("venue-id");
            int res = showDao.addShow(show);
            if (res == 1) {
               request.setAttribute("successMsg1", "Show added successfully.");
               request.setAttribute("successMsg2", "Please go back to dashboard to manage events, users, and
venues.");
               return new ModelAndView("success-view");
            } else {
               return serverError(request, "The show could not be added.");
            }
         }
      }

      public boolean invalidSessionObj(HttpSession session, String obj) {
         return session.getAttribute(obj) == null;
      }

      public ModelAndView sessionTimedOut(HttpServletRequest request) {
         request.setAttribute("errorMsg1", "Your session has been timed out.");
         request.setAttribute("errorMsg2", "Please log in again to start a new session.");
         return new ModelAndView("error-view");
      }

      public ModelAndView serverError(HttpServletRequest request, String msg1) {
         request.setAttribute("errorMsg1", msg1);
         request.setAttribute("errorMsg2", "There was a problem in reaching out to our servers. Please try again
later.");
         return new ModelAndView("error-view");
      }
}
```

AdminVenueController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.controller;

import com.mycompany.dao.VenueDao;
import com.mycompany.pojo.Venue;
```

```
import com.mycompany.validator.VenueValidator;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.InitBinder;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.servlet.ModelAndView;

/**
 *
 * @author ketanmalik
 */
@Controller
public class AdminVenueController {

   @Autowired
   VenueValidator venueValidator;

   @InitBinder("venue")
   protected void initBinder(WebDataBinder binder) {
      binder.setValidator(venueValidator);

   }

   @PostMapping("/add-venue.htm")
   public ModelAndView addVenue(HttpServletRequest request, HttpServletResponse response, HttpSession
session, Model model) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
      Venue venue = new Venue();
      model.addAttribute("addVenueForm", venue);
      return new ModelAndView("add-venue-view");
   }

   @PostMapping("/delete-venue.htm")
   public ModelAndView deleteVenue(HttpServletRequest request, HttpServletResponse response,
HttpSession session, VenueDao venueDao) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
```

```
        int res = venueDao.deleteVenue(Integer.parseInt(request.getParameter("delete-venue")));
        if (res > 0) {
            List<Venue> venueList = venueDao.getVenueList();
            session.setAttribute("venueList", venueList);
            return new ModelAndView("manage-venues-view");
        } else {
            return serverError(request, "The venue could not be deleted.");
        }
    }

    @GetMapping("/manage-venues.htm")
    public ModelAndView updateViewsInSession(HttpServletRequest request, HttpSession session, VenueDao
venueDao) {
        if (invalidSessionObj(session, "user")) {
            return sessionTimedOut(request);
        }
        List<Venue> venueList = venueDao.getVenueList();
        session.setAttribute("venueList", venueList);
        return new ModelAndView("manage-venues-view");
    }

    @PostMapping("/manage-venues.htm")
    public ModelAndView manageVenues(HttpServletRequest request, HttpServletResponse response,
HttpSession session, VenueDao venueDao) {
        if (invalidSessionObj(session, "user")) {
            return sessionTimedOut(request);
        }
        List<Venue> venueList = venueDao.getVenueList();
        session.setAttribute("venueList", venueList);
        return new ModelAndView("manage-venues-view");
    }

    @PostMapping("/updated-venue.htm")
    public ModelAndView saveUpdatedVenue(@ModelAttribute("updateVenueForm") Venue venue,
BindingResult result, Model model, HttpServletRequest request, HttpSession session, VenueDao venueDao) {
        venueValidator.validate(venue, result);
        if (result.hasErrors()) {
            return new ModelAndView("update-venue-view");
        } else {
            Venue sessionVenue = (Venue) session.getAttribute("venue");
            int res = venueDao.updateVenue(sessionVenue.getVenue_id(), venue);
            if (res == 1) {
                venue.setVenue_id(sessionVenue.getVenue_id());
                session.removeAttribute("venue");
                session.setAttribute("venue", venue);
                Venue obj = (Venue) session.getAttribute("venue");
                request.setAttribute("successMsg1", "Venue updated successfully.");
                request.setAttribute("successMsg2", "Please go back to dashboard to manage events, users, and
venues.");
```

```
            return new ModelAndView("success-view");
        } else {
            return serverError(request, "The venue could not be updated.");
        }
    }
  }

  @PostMapping("/venue-added.htm")
  public ModelAndView saveVenue(@ModelAttribute("addVenueForm") Venue venue, BindingResult result,
HttpServletRequest request, HttpSession session, VenueDao venueDao) {
    venueValidator.validate(venue, result);
    if (result.hasErrors()) {
        return new ModelAndView("add-venue-view");
    } else {
        int res = venueDao.saveVenue(venue);
        if (res == 1) {
            List<Venue> venueList = venueDao.getVenueList();
            session.setAttribute("eventList", venueList);
            request.setAttribute("successMsg1", "Venue added successfully.");
            request.setAttribute("successMsg2", "Please go back to dashboard to manage events, users, and
venues.");
            return new ModelAndView("success-view");
        } else {
            return serverError(request, "The venue could not be added.");
        }
    }
  }

  @PostMapping("/update-venue.htm")
  public ModelAndView updateVenue(HttpServletRequest request, HttpServletResponse response,
HttpSession session, VenueDao venueDao, Model model) {
    if (invalidSessionObj(session, "user")) {
        return sessionTimedOut(request);
    }
    int venue_id = Integer.parseInt(request.getParameter("update-venue"));
    Venue venue = venueDao.getVenue(venue_id);
//      System.out.println("Venue id: " + venue.getVenue_id());
//      System.out.println("Venue name: " + venue.getVenue_name());
//      System.out.println("Venue city: " + venue.getVenue_city());
//      System.out.println("Venue state: " + venue.getVenue_state());
//      System.out.println("Venue country: " + venue.getVenue_country());
//      System.out.println("Venue rows: " + venue.getVenue_rows());
//      System.out.println("Venue seat: " + venue.getSeat_per_row());
//      System.out.println("Venue seat price: " + venue.getSeat_price());

    if (venue == null) {
        if (session.getAttribute("venue") != null) {
            session.removeAttribute("venue");
        }
```

```
            return serverError(request, "The venue could not be updated.");
        }
        session.setAttribute("venue", venue);
        model.addAttribute("updateVenueForm", venue);
        return new ModelAndView("update-venue-view");
    }

    public boolean invalidSessionObj(HttpSession session, String obj) {
        return session.getAttribute(obj) == null;
    }

    public ModelAndView sessionTimedOut(HttpServletRequest request) {
        request.setAttribute("errorMsg1", "Your session has been timed out.");
        request.setAttribute("errorMsg2", "Please log in again to start a new session.");
        return new ModelAndView("error-view");
    }

    public ModelAndView serverError(HttpServletRequest request, String msg1) {
        request.setAttribute("errorMsg1", msg1);
        request.setAttribute("errorMsg2", "There was a problem in reaching out to our servers. Please try again
later.");
        return new ModelAndView("error-view");
    }
}
```

BookingController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.controller;

import com.mycompany.dao.BookingDao;
import com.mycompany.dao.ShowDao;
import com.mycompany.dao.VenueDao;
import com.mycompany.pojo.Booking;
import com.mycompany.pojo.Event;
import com.mycompany.pojo.Show;
import com.mycompany.pojo.User;
import com.mycompany.pojo.Venue;
import com.mycompany.utils.Util;
import com.mycompany.validator.BookingValidator;
import com.mycompany.view.BookingView;
import java.util.Date;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.InitBinder;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.servlet.ModelAndView;

/**
 *
 * @author ketanmalik
 */
@Controller
public class BookingController {

   @Autowired
   BookingValidator bookingValidator;

   @InitBinder("show")
   protected void initBinder(WebDataBinder binder) {
      binder.setValidator(bookingValidator);
   }

   @PostMapping("/step-3.htm")
   public ModelAndView verifyBookingDetails(HttpServletRequest request, HttpSession session, ShowDao
showDao, Model model, VenueDao venueDao) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
      int show_id = Integer.parseInt(request.getParameter("show-id"));
      try {
         int venue_id = Integer.parseInt(session.getAttribute("venue-id") + "");
         Venue requestedVenue = venueDao.getVenue(venue_id);
         if (requestedVenue == null) {
            System.out.println("Venue could not be found in verifyBookingDetails (Booking Controller)");
            return serverError(request, "We could not proceed with your booking.");
         }
         session.setAttribute("requestedVenue", requestedVenue);
         session.removeAttribute("venue-id");
      } catch (Exception e) {
         System.out.println("Exception in verifyBookingDetails (Booking Controller: )");
         e.printStackTrace();
         return serverError(request, "We could not proceed with your booking.");
      }

      Show selectedShow = showDao.getShowObj(show_id);
      session.setAttribute("selectedShow", selectedShow);
```

```
        Booking booking = new Booking();
        model.addAttribute("bookingForm", booking);
        return new ModelAndView("customer-details-view");
    }

    @PostMapping("/step-4.htm")
    public ModelAndView confirmBooking(@ModelAttribute("bookingForm") Booking booking, BindingResult
result, HttpServletRequest request, HttpSession session, BookingDao bookingDao, ShowDao showDao) {
        if (invalidSessionObj(session, "user")) {
            return sessionTimedOut(request);
        }
        bookingValidator.validate(booking, result);
        if (result.hasErrors()) {
            return new ModelAndView("customer-details-view");
        }
        Show selectedShow = (Show) session.getAttribute("selectedShow");
        Show mostRecentShow = showDao.getShowObj(selectedShow.getShow_id());
        if (mostRecentShow.getSeats_left() == 0) {
            return seatsExhausted(request, bookingDao, booking);
        }
        bookingValidator.validateCustom(result, booking, mostRecentShow);
        if (result.hasErrors()) {
            return new ModelAndView("customer-details-view");
        }
        try {
            Date booking_date = new Date();
            String phone = booking.getPhone();
            int seats = booking.getSeats();
            int price = seats * selectedShow.getSeat_price();
            User user = (User) session.getAttribute("user");
            Event event = (Event) session.getAttribute("requestedEvent");
            Venue venue = (Venue) session.getAttribute("requestedVenue");
            Booking savedBooking = bookingDao.saveBooking(booking_date, phone, seats, price, user, event,
venue, selectedShow);

            if (savedBooking != null) {
                Show updatedShow = showDao.updateSeatsLeft(selectedShow.getShow_id(), seats);
                if (updatedShow == null) {
                    bookingDao.deleteBooking(savedBooking.getBooking_id());
                    return seatsExhausted(request, bookingDao, savedBooking);
                }
                saveBookingPdf(session, savedBooking, venue, selectedShow);
                return new ModelAndView("customer-confirm-view");
            } else {
                return serverError(request, "Your booking could not be confirmed");
            }

        } catch (Exception e) {
            System.out.println("Exception in confirmBooking (BookingController): ");
```

```
            return serverError(request, "Your booking could not be confirmed");
        }
    }

    public boolean invalidSessionObj(HttpSession session, String obj) {
        return session.getAttribute(obj) == null;
    }

    public ModelAndView sessionTimedOut(HttpServletRequest request) {
        request.setAttribute("errorMsg1", "Your session has been timed out.");
        request.setAttribute("errorMsg2", "Please log in again to start a new session.");
        return new ModelAndView("error-view");
    }

    public ModelAndView serverError(HttpServletRequest request, String msg1) {
        request.setAttribute("errorMsg1", msg1);
        request.setAttribute("errorMsg2", "There was a problem in reaching out to our servers. Please try again
later.");
        return new ModelAndView("error-view");
    }

    public void saveBookingPdf(HttpSession session, Booking booking, Venue venue, Show show) {
        User user = booking.getUser();
        String booking_id = booking.getBooking_id() + "";
        String event_name = (String) session.getAttribute("selectedEvent");
        String pdf_venue = venue.getVenue_name() + ", " + venue.getVenue_city();
        String pdf_event_date_time = show.getShow_time() + ", " + Util.dateToString(show.getShow_date(),
"date");
        String screen = show.getScreen();
        String seats = booking.getSeats() + "";
        String price = "$" + booking.getPrice();
        String booking_date = Util.dateToString(booking.getBooking_date(), "dateTime");
        String name = user.getfName() + " " + user.getlName();
        String contact = user.getEmail() + " " + booking.getPhone();
        BookingView bookingPdf = new BookingView(booking_id, event_name, pdf_venue, pdf_event_date_time,
screen, seats, price, booking_date, name, contact);
        removeTempSessionAttributes(session);
        session.setAttribute("bookingPdf", bookingPdf);
    }

    private void removeTempSessionAttributes(HttpSession session) {
        session.removeAttribute("selectedCity");
        session.removeAttribute("requestedShows");
        session.removeAttribute("requestedVenue");
        session.removeAttribute("requestedEvent");
        session.removeAttribute("selectedShow");
        session.removeAttribute("selectedEvent");
        session.removeAttribute("eventsInCity");
    }
```

```
    private ModelAndView seatsExhausted(HttpServletRequest request, BookingDao bookingDao, Booking
savedBooking) {
        request.setAttribute("errorMsg1", "Your booking could not be confirmed");
        request.setAttribute("errorMsg2", "The show you selected ran out of seats. Please select a different show
to book.");
        return new ModelAndView("error-view");
    }
}
```

CustomerViewController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.controller;

import com.mycompany.dao.EventDao;
import com.mycompany.dao.ShowDao;
import com.mycompany.pojo.Event;
import com.mycompany.pojo.Show;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.ModelAndView;
import org.springframework.web.servlet.view.RedirectView;

/**
 *
 * @author ketanmalik
 */
@Controller
public class CustomerViewController {

    @GetMapping("/get-events.htm")
    @ResponseBody
    public String getEventsInCity(HttpServletRequest request, HttpServletResponse response, HttpSession
session, ShowDao showDao) {
        List<Event> eventsList = showDao.getEventsInCity(request.getParameter("city"));
        String eventsInCity = "";
        for (int i = 0; i < eventsList.size(); i++) {
```

```
            eventsInCity += eventsList.get(i).getEvent_name() + ",";
        }
        if (eventsInCity.length() > 0) {
            eventsInCity = eventsInCity.substring(0, eventsInCity.length() - 1);
        }
        session.setAttribute("eventsInCity", eventsList);
        return eventsInCity;
    }

    @PostMapping("/step-2.htm")
    public ModelAndView selectShow(HttpServletRequest request, HttpServletResponse response, HttpSession
session, EventDao eventDao, ShowDao showDao) {
        session.removeAttribute("cityError");
        session.removeAttribute("eventError");
        if (invalidSessionObj(session, "user")) {
            return sessionTimedOut(request);
        }
        String city = request.getParameter("city");
        String event = request.getParameter("event");
        boolean errors = false;
        if (city == null || (city != null && city.equals(""))) {
            errors = true;
            session.setAttribute("cityError", "Please select a city");
        }
        if (event == null || (event != null && event.equals(""))) {
            errors = true;
            session.setAttribute("eventError", "Please select an event");
        }
        if (errors) {
            return new ModelAndView(new RedirectView("/log-in-user.htm", true));
        }
        session.setAttribute("selectedCity", city);
        session.setAttribute("selectedEvent", event);
        Event requestedEvent = eventDao.getEventFromName(event);
        List<Show> shows = showDao.getShow(city, event);
        session.setAttribute("requestedEvent", requestedEvent);
        session.setAttribute("requestedShows", shows);
        if (shows == null) {
            return serverError(request, "We could not retrieve any shows at the moment.");
        }
        return new ModelAndView("customer-select-view");
    }

    public boolean invalidSessionObj(HttpSession session, String obj) {
        return session.getAttribute(obj) == null;
    }

    public ModelAndView sessionTimedOut(HttpServletRequest request) {
        request.setAttribute("errorMsg1", "Your session has been timed out.");
```

```
        request.setAttribute("errorMsg2", "Please log in again to start a new session.");
        return new ModelAndView("error-view");
    }

    public ModelAndView serverError(HttpServletRequest request, String msg1) {
        request.setAttribute("errorMsg1", msg1);
        request.setAttribute("errorMsg2", "There was a problem in reaching out to our servers. Please try again
later.");
        return new ModelAndView("error-view");
    }
}
```

PdfController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.controller;

import com.mycompany.pojo.Booking;
import com.mycompany.pojo.Show;
import com.mycompany.pojo.Venue;
import com.mycompany.view.BookingView;
import java.util.HashMap;
import java.util.Map;
import javax.servlet.http.HttpSession;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.servlet.ModelAndView;

/**
 *
 * @author ketanmalik
 */
@Controller
public class PdfController {

    @GetMapping("/booking.pdf")
    public ModelAndView generateBookingPdf(HttpSession session, Booking booking, Venue venue, Show
show) {
        BookingView bookingPdf = (BookingView) session.getAttribute("bookingPdf");
        return new ModelAndView("booking-pdf", "bookingPdf", bookingPdf);
    }

    private void removeTempSessionAttributes(HttpSession session) {
```

```
        session.removeAttribute("selectedCity");
        session.removeAttribute("requestedShows");
        session.removeAttribute("requestedVenue");
        session.removeAttribute("requestedEvent");
        session.removeAttribute("selectedShow");
        session.removeAttribute("selectedEvent");
        session.removeAttribute("eventsInCity");
    }
}
```

UserBookingsController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.controller;

import com.mycompany.dao.BookingDao;
import com.mycompany.dao.ShowDao;
import com.mycompany.pojo.Booking;
import com.mycompany.pojo.User;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;
import java.util.concurrent.TimeUnit;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.servlet.ModelAndView;

/**
 *
 * @author ketanmalik
 */
@Controller
public class UserBookingsController {

    @GetMapping("/user-bookings.htm")
    public ModelAndView userBookings(HttpServletRequest request, HttpSession session, BookingDao
bookingDao) {
        if (invalidSessionObj(session, "user")) {
            return sessionTimedOut(request);
```

```
        }
        User user = (User) session.getAttribute("user");
        List<Booking> bookings = bookingDao.getBookings(user.getUser_id());
        session.setAttribute("bookings", bookings);
        return new ModelAndView("user-bookings-view");
    }

    @PostMapping("/cancel-booking.htm")
    public ModelAndView cancelBooking(HttpServletRequest request, HttpSession session, BookingDao
bookingDao, ShowDao showDao) throws ParseException {
        if (invalidSessionObj(session, "user")) {
            return sessionTimedOut(request);
        }
        int booking_id = 0;
        try {
            booking_id = Integer.parseInt(request.getParameter("cancel-booking") + "");
        } catch (Exception e) {
            System.out.println("Exception in cancelBooking userbookingcontroller");
            e.printStackTrace();
            return serverError(request, "We could not retrieve your booking at the moment");
        }
        Booking booking = bookingDao.getBooking(booking_id);
        if (booking == null) {
            return serverError(request, "We could not retrieve your booking at the moment");
        }
        Date bookingDate = booking.getShow().getShow_date();
        Date todaysDate = new Date();
        SimpleDateFormat sdformat = new SimpleDateFormat("yyyy-MM-dd");
        todaysDate = sdformat.parse(sdformat.format(todaysDate));
        bookingDate = sdformat.parse(sdformat.format(bookingDate));

        long diffInMillies = Math.abs(bookingDate.getTime() - todaysDate.getTime());
        long diff = TimeUnit.DAYS.convert(diffInMillies, TimeUnit.MILLISECONDS);
        if (diff < 3) {
            return new ModelAndView("user-bookings-view", "errorMsg", "Your booking cannot be cancelled
because it is past the cancellation period.");
        }
        showDao.addSeats(booking.getShow().getShow_id(), booking.getSeats());
        bookingDao.deleteBooking(booking_id);
        User user = (User) session.getAttribute("user");
        List<Booking> bookings = bookingDao.getBookings(user.getUser_id());
        session.setAttribute("bookings", bookings);
        return new ModelAndView("user-bookings-view", "successMsg", "Your booking has been cancelled
successfully!");
    }

    public boolean invalidSessionObj(HttpSession session, String obj) {
        return session.getAttribute(obj) == null;
    }
```

```
    public ModelAndView sessionTimedOut(HttpServletRequest request) {
        request.setAttribute("errorMsg1", "Your session has been timed out.");
        request.setAttribute("errorMsg2", "Please log in again to start a new session.");
        return new ModelAndView("error-view");
    }

    public ModelAndView serverError(HttpServletRequest request, String msg1) {
        request.setAttribute("errorMsg1", msg1);
        request.setAttribute("errorMsg2", "There was a problem in reaching out to our servers. Please try again
later.");
        return new ModelAndView("error-view");
    }
}
```

UserController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.controller;

import com.mycompany.dao.BookingDao;
import com.mycompany.dao.EventDao;
import com.mycompany.dao.ShowDao;
import com.mycompany.dao.UserDao;
import com.mycompany.dao.VenueDao;
import com.mycompany.pojo.Event;
import com.mycompany.pojo.Show;
import com.mycompany.pojo.User;
import com.mycompany.pojo.Venue;
import com.mycompany.validator.UserValidator;
import java.io.Serializable;
import java.util.Enumeration;
import java.util.List;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.GetMapping;
```

```
import org.springframework.web.bind.annotation.InitBinder;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.ModelAndView;

/**
 *
 * @author ketanmalik
 */
@Controller
public class UserController implements Serializable {

   @Autowired
   UserValidator userValidator;

   @InitBinder("show")
   protected void initBinder(WebDataBinder binder) {
      binder.setValidator(userValidator);
   }

   @PostMapping("/manage-users.htm")
   public ModelAndView manageEvents(HttpServletRequest request, HttpServletResponse response,
HttpSession session) {
      if (invalidSessionObj(session, "user")) {
         return sessionTimedOut(request);
      }
      User user = (User) session.getAttribute("user");
      System.out.println("usr: " + user.getEmail());
      return new ModelAndView("manage-users-view");
   }

   @PostMapping("/sign-up-success.htm")
   public ModelAndView signUpUser(@ModelAttribute("signupForm") User user, BindingResult result,
HttpServletRequest request, HttpServletResponse response, UserDao userDao) {
      userValidator.validate(user, result);
      if (result.hasErrors()) {
         return new ModelAndView("sign-up-view");
      }
      String fName = request.getParameter("fName");
      String lName = request.getParameter("lName");
      String email = request.getParameter("email");
      String password = request.getParameter("password");
      String city = request.getParameter("city");
      String state = request.getParameter("state");
      String country = request.getParameter("country");
      int res = userDao.addUser(fName, lName, email, password, city, state, country, "customer");
      if (res == -1) {
```

```
            request.setAttribute("errorMsg1", "You could not be signed up.");
            request.setAttribute("errorMsg2", "A user already exists with this email. Please try to log in instead.");
            return new ModelAndView("error-view");
        }
        if (res == 1) {
            request.setAttribute("successMsg1", "You have signed up successfully.");
            request.setAttribute("successMsg2", "Please log in from home page to enjoy our services.");
            return new ModelAndView("success-view");
        }
        request.setAttribute("errorMsg1", "You could not be signed up.");
        request.setAttribute("errorMsg2", "There was a problem in reaching out to our servers. Please try again
later.");
        return new ModelAndView("error-view");
    }

    @PostMapping("/log-in-user.htm")
    public ModelAndView logInUser(HttpServletRequest request, HttpServletResponse response, UserDao
userDao, HttpSession session, EventDao eventDao, VenueDao venueDao, ShowDao showDao, BookingDao
bookingDao) {
        String email = request.getParameter("email");
        User user = userDao.getUser(email);
        if (user == null) {
            request.setAttribute("errorMsg1", "You could not be logged in.");
            request.setAttribute("errorMsg2", "Either your browser doesn't allow Javascript or our network is not
responding.");
            if (session.getAttribute("user") != null) {
                session.removeAttribute("user");
            };
            return new ModelAndView("error-view");
        }

        List<Event> eventList = eventDao.getEventList();
        List<Venue> venueList = venueDao.getVenueList();
        List<Show> showList = showDao.getShowList();
        session.setAttribute("user", user);
        session.setAttribute("eventList", eventList);
        session.setAttribute("venueList", venueList);
        session.setAttribute("showList", showList);

        if (user.getUser_type().equals("admin")) {
            return new ModelAndView("manage-events-view");
        } else {
            List<String> venueCityList = venueDao.getUniqueVenueCityList();
            session.setAttribute("venueCityList", venueCityList);
            return new ModelAndView("customer-view");
        }
    }

    @GetMapping("/log-in-user.htm")
```

```
  public ModelAndView redirectUser(HttpServletRequest request, HttpSession session, EventDao eventDao) {
    User loggedInUser = (User) session.getAttribute("user");
    if (loggedInUser != null) {
      if (loggedInUser.getUser_type().equals("admin")) {
        session.removeAttribute("eventList");
        List<Event> eventList = eventDao.getEventList();
        session.setAttribute("eventList", eventList);
        return new ModelAndView("manage-events-view");
      } else {
        return new ModelAndView("customer-view");
      }
    } else {
      return new ModelAndView("index");
    }
  }

  @PostMapping("/sign-out.htm")
  public ModelAndView signOutUser(HttpSession session) {
    Enumeration<String> attributes = session.getAttributeNames();
    while (attributes.hasMoreElements()) {
      String attribute = (String) attributes.nextElement();
      session.removeAttribute(attribute);
    }
    return new ModelAndView("index");
  }

  @PostMapping("/validate-user.htm")
  @ResponseBody
  public String validateUser(HttpServletRequest request, HttpServletResponse response, HttpSession session,
UserDao userDao) {
    String email = request.getParameter("email");
    String password = request.getParameter("password");

    if (email.trim().equals("") || password.trim().equals("")) {
      return "invalidUser";
    }
    boolean validUser = userDao.checkUser(email, password, "log-in");
    if (validUser) {
      return "validUser";
    }
    return "invalidUser";
  }

  @GetMapping("/log-in.htm")
  public ModelAndView logInPage(HttpServletRequest request, HttpServletResponse response) {
    return new ModelAndView("log-in-view");
  }

  @GetMapping("/sign-up.htm")
```

```
    public ModelAndView signUpPage(HttpServletRequest request, HttpServletResponse response, Model
model) {
        User user = new User();
        model.addAttribute("signupForm", user);
        return new ModelAndView("sign-up-view");
    }

    public boolean invalidSessionObj(HttpSession session, String obj) {
        return session.getAttribute(obj) == null;
    }

    public ModelAndView sessionTimedOut(HttpServletRequest request) {
        request.setAttribute("errorMsg1", "Your session has been timed out.");
        request.setAttribute("errorMsg2", "Please log in again to start a new session.");
        return new ModelAndView("error-view");
    }

    public ModelAndView serverError(HttpServletRequest request, String msg1) {
        request.setAttribute("errorMsg1", msg1);
        request.setAttribute("errorMsg2", "There was a problem in reaching out to our servers. Please try again
later.");
        return new ModelAndView("error-view");
    }
}
```

BookingDao

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.dao;

import com.mycompany.pojo.Booking;
import com.mycompany.pojo.Event;
import com.mycompany.pojo.Show;
import com.mycompany.pojo.User;
import com.mycompany.pojo.Venue;
import java.util.Date;
import java.util.List;
import org.hibernate.query.Query;

/**
 *
 * @author ketanmalik
 */
public class BookingDao extends DAO {
```

```
    public Booking saveBooking(Date date, String phone, int seats, int price, User user, Event event, Venue
venue, Show show) {
        try {
            Show selectedShow = new Show(show.getShow_id(), show.getSeat_price(), show.getTotal_rows(),
show.getSeats_per_row(), show.getShow_time(), show.getSeats_left(), show.getShow_date(),
show.getScreen(), event, venue);
            Booking booking = new Booking(date, phone, seats, price, user, selectedShow);
            beginTransaction();
            getSession().save(booking);
            commit();
            return booking;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in saveBooking BookingDAO: ");
            e.printStackTrace();
            return null;
        }
    }

    public void deleteBooking(int booking_id) {
        try {
            String hql = "DELETE FROM Booking WHERE booking_id = :booking_id";
            Query query = getSession().createQuery(hql);
            query.setParameter("booking_id", booking_id);
            beginTransaction();
            query.executeUpdate();
            commit();
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in deleteBooking BookingDAO: ");
            e.printStackTrace();
        }
    }

    public List<Booking> getBookings(int user_id) {
        try {
            String hql = "FROM Booking WHERE user.user_id = :user_id";
            Query query = getSession().createQuery(hql);
            query.setParameter("user_id", user_id);
            beginTransaction();
            List<Booking> bookings = query.list();
            if (bookings.isEmpty()) {
                rollback();
                return null;
            }
            commit();
            return bookings;
        } catch (Exception e) {
```

```
            rollback();
            System.out.println("Exception in getBookings BookingDAO: ");
            e.printStackTrace();
            return null;
        }
    }

    public Booking getBooking(int booking_id) {
        try {
            String hql = "FROM Booking WHERE booking_id = :booking_id";
            Query query = getSession().createQuery(hql);
            query.setParameter("booking_id", booking_id);
            beginTransaction();
            List<Booking> bookings = query.list();
            if (bookings.isEmpty()) {
                rollback();
                return null;
            }
            commit();
            System.out.println("committed2");
            return bookings.get(0);
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getBooking BookingDAO: ");
            e.printStackTrace();
            return null;
        }
    }
}
```

EventDao

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.dao;

import com.mycompany.pojo.Event;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import org.hibernate.query.Query;

/**
 *
 * @author ketanmalik
```

```
 */
public class EventDao extends DAO {

    public int deleteEvent(int event_id) {
        String hql = "DELETE FROM Event WHERE event_id=:event_id";
        Query query = getSession().createQuery(hql);
        query.setParameter("event_id", event_id);
        beginTransaction();
        int res = query.executeUpdate();
        commit();
        return res;
    }

    public Event getEvent(int event_id) {
        try {
            String hql = "FROM Event WHERE event_id=:event_id";
            Query query = getSession().createQuery(hql);
            query.setParameter("event_id", event_id);
            List<Event> eventList = new ArrayList<>();
            beginTransaction();
            eventList = query.list();
            if (eventList.isEmpty()) {
                rollback();
                return null;
            }
            commit();
            return eventList.get(0);
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getEvent: " + e);
            return null;
        }
    }

    public Event getEventFromName(String event_name) {
        try {
            String hql = "FROM Event WHERE event_name = :event_name";
            Query query = getSession().createQuery(hql);
            query.setParameter("event_name", event_name);
            beginTransaction();
            Event event = (Event) query.list().get(0);
            commit();
            return event;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getEventFromName Dao: " + e);
            return null;
        }
    }
```

```
    public List<Event> getEventList() {
        try {
            String hql = "FROM Event";
            Query query = getSession().createQuery(hql);
            List<Event> result = new ArrayList<>();
            beginTransaction();
            result = query.list();
            commit();
            return result;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getEventList: " + e);
            return null;
        }

    }

    public int saveEvent(String event_name, String event_type, String event_cast, String event_rating, String
event_genre,
            String event_language, String event_summary, String event_duration, Date event_date) {

        Event event = new Event();
        event.setEvent_name(event_name);
        event.setEvent_type(event_type);
        event.setEvent_cast(event_cast);
        event.setEvent_rating(event_rating);
        event.setEvent_genre(event_genre);
        event.setEvent_language(event_language);
        event.setEvent_summary(event_summary);
        event.setEvent_duration(event_duration);
        event.setEvent_date(event_date);
        beginTransaction();
        getSession().save(event);
        commit();
        return 1;
    }

    public int updateEvent(int event_id, Event event) {
        String event_name = event.getEvent_name();
        String event_cast = event.getEvent_cast();
        String event_type = event.getEvent_type();
        String event_rating = event.getEvent_rating();
        String event_genre = event.getEvent_genre();
        String event_language = event.getEvent_language();
        String event_summary = event.getEvent_summary();
        Date event_date = event.getEvent_date();
        String event_duration = event.getEvent_duration();
```

```
        String hql = "UPDATE Event SET event_name=:event_name, event_cast=:event_cast,
event_type=:event_type, "
                + "event_rating=:event_rating, event_genre=:event_genre, event_language=:event_language, "
                + "event_summary=:event_summary, event_date=:event_date, event_duration=:event_duration
WHERE event_id=:event_id";
        Query query = getSession().createQuery(hql);
        query.setParameter("event_id", event_id);
        query.setParameter("event_name", event_name);
        query.setParameter("event_cast", event_cast);
        query.setParameter("event_type", event_type);
        query.setParameter("event_rating", event_rating);
        query.setParameter("event_genre", event_genre);
        query.setParameter("event_language", event_language);
        query.setParameter("event_summary", event_summary);
        query.setParameter("event_date", event_date);
        query.setParameter("event_duration", event_duration);
        beginTransaction();
        int res = query.executeUpdate();
        commit();
        return res;
    }
}
```

ShowDao

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.dao;

import com.mycompany.pojo.Event;
import com.mycompany.pojo.Show;
import java.util.ArrayList;
import java.util.List;
import org.hibernate.query.Query;

/**
 *
 * @author ketanmalik
 */
public class ShowDao extends DAO {

    public int addShow(Show show) {
        try {
            beginTransaction();
            getSession().save(show);
```

```
            commit();
            return 1;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in addShow DAO: ");
            e.printStackTrace();
            return 0;
        }
    }

    public List<Show> getShowList() {
        try {
            String hql = "FROM Show";
            Query query = getSession().createQuery(hql);
            List result = new ArrayList<>();
            beginTransaction();
            result = query.list();
            commit();
            return result;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getShowList Dao: ");
            e.printStackTrace();
            return null;
        }
    }

    public List<Event> getEventsInCity(String city) {
        try {
            String hql = "SELECT DISTINCT event FROM Show WHERE venue_id in (SELECT venue_id FROM Venue
WHERE venue_city = :city)";
            Query query = getSession().createQuery(hql);
            query.setParameter("city", city);
            beginTransaction();
            List<Event> eventsList = query.list();
            commit();
            return eventsList;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getEventsFromCity Dao: ");
            e.printStackTrace();
            return null;
        }
    }

    public List<Show> getShow(String city, String event) {
        try {
            beginTransaction();
```

```
        String event_hql = "SELECT event_id FROM Event WHERE event_name = :event_name";
        Query event_query = getSession().createQuery(event_hql);
        event_query.setParameter("event_name", event);

        List<Integer> event_id_list = event_query.list();
        if (event_id_list.isEmpty()) {
           rollback();
           return null;
        }
        int event_id = (Integer) event_query.list().get(0);

        String venue_hql = "SELECT venue_id FROM Venue WHERE venue_city = :venue_city";
        Query venue_query = getSession().createQuery(venue_hql);
        venue_query.setParameter("venue_city", city);
        List<Integer> venue_id_list = venue_query.list();
        if (venue_id_list.isEmpty()) {
           rollback();
           return null;
        }

        String shows_hql = "FROM Show WHERE event.event_id = :event_id AND venue.venue_id IN
(:venue_ids)";
        Query shows_query = getSession().createQuery(shows_hql);
        shows_query.setParameter("event_id", event_id);
        shows_query.setParameterList("venue_ids", venue_id_list);
        List<Show> shows = shows_query.list();
        if (shows.isEmpty()) {
           rollback();
           return null;
        }
        commit();
        return shows;
     } catch (Exception e) {
        rollback();
        System.out.println("Exception in getShowList Dao: ");
        e.printStackTrace();
        return null;
     }
  }

  public Show getShowObj(int show_id) {
     try {
        String hql = "FROM Show WHERE show_id = :show_id";
        Query query = getSession().createQuery(hql);
        query.setParameter("show_id", show_id);
        beginTransaction();
        List<Show> shows = query.list();
        if (shows.isEmpty()) {
           rollback();
```

```
            return null;
        }
        commit();
        return shows.get(0);
    } catch (Exception e) {
        rollback();
        System.out.println("Exception in getShowList Dao: ");
        e.printStackTrace();
        return null;
    }
}

public Show updateSeatsLeft(int show_id, int seats_booked) {
    try {
        String hql = "FROM Show WHERE show_id = :show_id";
        Query query = getSession().createQuery(hql);
        query.setParameter("show_id", show_id);
        beginTransaction();
        List<Show> shows = query.list();
        if (shows.isEmpty()) {
            rollback();
            return null;
        }
        Show show_to_update = shows.get(0);
        int seats_left = show_to_update.getSeats_left() - seats_booked;
        if (seats_left < 0) {
            rollback();
            return null;
        }
        hql = "UPDATE Show SET seats_left = :seats_left where show_id = :show_id";
        query = getSession().createQuery(hql);
        query.setParameter("seats_left", seats_left);
        query.setParameter("show_id", show_id);
        query.executeUpdate();
        commit();
        show_to_update.setSeats_left(seats_left);
        return show_to_update;
    } catch (Exception e) {
        rollback();
        System.out.println("Exception in updateSeatsLeft ShowDao: ");
        e.printStackTrace();
        return null;
    }
}

public void addSeats(int show_id, int seatsToAdd) {
    try {
        String hql = "FROM Show WHERE show_id = :show_id";
        Query query = getSession().createQuery(hql);
```

```
            query.setParameter("show_id", show_id);
            beginTransaction();
            List<Show> shows = query.list();
            if (shows.isEmpty()) {
               rollback();
            }
            Show show = shows.get(0);
            show.setSeats_left(show.getSeats_left() + seatsToAdd);
            getSession().update(show);
            commit();
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getShowList Dao: ");
            e.printStackTrace();
        }
    }
}
```

UserDao

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.dao;

import com.mycompany.pojo.User;
import java.util.List;
import org.hibernate.query.Query;

/**
 *
 * @author ketanmalik
 */
public class UserDao extends DAO {

    public int addUser(String fName, String lName, String email, String password, String city, String state, String
country, String user_type) {
        boolean userAlreadyExists = checkUser(email, "", "sign-up");
        if (userAlreadyExists) {
            return -1;
        }
        try {
            User user = new User(fName, lName, email, password, city, state, country, user_type);
            beginTransaction();
            getSession().save(user);
            commit();
```

```
            return 1;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in addUser UserDao: ");
            e.printStackTrace();
            return -1;
        }
    }

    public boolean checkUser(String email, String password, String mode) {
        String hql;
        if (mode.equalsIgnoreCase("sign-up")) {
            hql = "FROM User where email=:email";
        } else {
            hql = "FROM User where email=:email and password =:password";
        }
        try {
            Query query = getSession().createQuery(hql);
            query.setParameter("email", email);
            if (mode.equalsIgnoreCase("log-in")) {
                query.setParameter("password", password);
            }
            beginTransaction();
            List users = query.list();
            if (users.isEmpty()) {
                rollback();
                return false;
            }
            commit();
            return true;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in checkUser UserDao: ");
            e.printStackTrace();
            return false;
        }
    }

    public User getUser(String email) {
        try {
            String hql = "From User Where email = :email";
            Query query = getSession().createQuery(hql);
            query.setParameter("email", email);
            beginTransaction();
            List<User> users = query.list();
            if (users.isEmpty()) {
                rollback();
                return null;
            }
```

```
            commit();
            return users.get(0);
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getUser UserDao: ");
            e.printStackTrace();
            return null;
        }
    }
}
```

VenueDao

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.mycompany.dao;

import java.util.List;
import org.hibernate.query.Query;
import com.mycompany.pojo.Venue;
import java.util.ArrayList;

/**
 *
 * @author ketanmalik
 */
public class VenueDao extends DAO {

    public int deleteVenue(int venue_id) {
        String hql = "DELETE FROM Venue WHERE venue_id=:venue_id";
        Query query = getSession().createQuery(hql);
        query.setParameter("venue_id", venue_id);
        beginTransaction();
        int res = query.executeUpdate();
        commit();
        return res;
    }

    public List<String> getUniqueVenueCityList() {
        try {
            String hql = "SELECT DISTINCT venue_city FROM Venue";
            Query query = getSession().createQuery(hql);
            List<String> venueList = new ArrayList<>();
            beginTransaction();
```

```
            venueList = query.list();
            commit();
            return venueList;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getUniqueVenueList Dao: ");
            e.printStackTrace();
            return null;
        }
    }

    public Venue getVenue(int venue_id) {
        try {
            String hql = "FROM Venue WHERE venue_id=:venue_id";
            Query query = getSession().createQuery(hql);
            query.setParameter("venue_id", venue_id);
            List<Venue> venueList = new ArrayList<>();
            beginTransaction();
            venueList = query.list();
            if (venueList.isEmpty()) {
                rollback();
                return null;
            }
            commit();
            return venueList.get(0);
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getVenue: " + e);
            return null;
        }

    }

    public List<Venue> getVenueList() {
        try {
            String hql = "FROM Venue";
            Query query = getSession().createQuery(hql);
            List result = new ArrayList<>();
            beginTransaction();
            result = query.list();
            commit();
            return result;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in getVenueList: " + e);
            return null;
        }
    }
```

```
    public int saveVenue(Venue venue) {
        try {
            beginTransaction();
            getSession().save(venue);
            commit();
            return 1;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in saveVenue: ");
            e.printStackTrace();
            return 0;
        }
    }

    public int updateVenue(int venue_id, Venue venue) {
        try {
            String venue_name = venue.getVenue_name();
            String venue_city = venue.getVenue_city();
            String venue_state = venue.getVenue_state();
            String venue_country = venue.getVenue_country();
            String hql = "UPDATE Venue SET venue_name=:venue_name, venue_city=:venue_city,
venue_state=:venue_state, "
                    + "venue_country=:venue_country WHERE venue_id=:venue_id";
            Query query = getSession().createQuery(hql);
            query.setParameter("venue_id", venue_id);
            query.setParameter("venue_name", venue_name);
            query.setParameter("venue_city", venue_city);
            query.setParameter("venue_state", venue_state);
            query.setParameter("venue_country", venue_country);
            beginTransaction();
            int res = query.executeUpdate();
            commit();
            return res;
        } catch (Exception e) {
            rollback();
            System.out.println("Exception in updateVenue: " + e);
            return 0;
        }
    }
}
```