Министерство транспорта Российской Федерации

Федеральное агентство железнодорожного транспорта

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Дальневосточный государственный университет путей сообщения»

Кафедра «Информационные технологии и системы»

Курсовой проект

по дисциплине «Информационные WEB-системы»

Тема: Разработка WEB-приложения «Авиабилеты»

Выполнил: Корнеева А.С,

Платов А.С.

240 группа

Проверил: Даниленко П.В.

Хабаровск, 2020

Оглавление

[1 Постановка задачи 3](#_Toc38477792)

[2 Создание базы данных 3](#_Toc38477793)

[2.1 Модели данных 3](#_Toc38477794)

[2.2 Entity Framework 3](#_Toc38477795)

[2.3 Миграция базы данных 3](#_Toc38477796)

[2.4 Схема данных 3](#_Toc38477797)

[3 Авторизация 3](#_Toc38477798)

[4 Создание сайта 3](#_Toc38477799)

[4.1 Мастер-страница 3](#_Toc38477800)

[4.2 Страница поиска рейсов 3](#_Toc38477801)

[4.3 Страница ввода информации о пассажирах и выбора мест в салоне 3](#_Toc38477802)

[4.4 Страница подтверждения заказа 3](#_Toc38477803)

[4.6 Страница купленных билетов 3](#_Toc38477804)

[Заключение 3](#_Toc38477805)

[Список литературы 3](#_Toc38477806)

# 1 Постановка задачи

# 2 Создание базы данных

## 2.1 Модели данных

## 2.2 Entity Framework

## 2.3 Миграция базы данных

## 2.4 Схема данных

# 3 Авторизация

# 4 Создание сайта

## 4.1 Мастер-страница

Мастер-страницы – это средство ASP.NET, разработанное специально для стандартизации компоновки веб-страниц. Мастер-страницы представляют собой шаблоны веб-страниц, которые могут определять фиксированное содержимое и объявлять часть веб-страницы, куда можно помещать нестандартное содержимое. При использовании одной и той же мастер-страницы во всем веб-сайте компоновка гарантированно будет одинаковой. Более того, если изменить определение мастер-страницы после ее применения, то все использующие ее веб-страницы автоматически воспримут это изменение.

В данном WEB-приложении для всех созданных представлений используется мастер-страница \_Layout.cshtml, которая находится в каталоге /Views/Shared/.

Мастер-страница \_Layout.chtml содержит в себе следующие элементы:

* теги «head», внутри которых содержится вся необходимая мета информация о странице;
* ссылки на подключаемые CSS-файлы и JS-скрипты;
* секцию для рендера представлений, использующих в качестве шаблона данную мастер-страницу;
* навигационную панель;
* шапку и подвал.

Листинг 4.1.1 Мастер-страница \_Layout.cshtml

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<meta charset="utf-8" />

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

<link type="text/css" rel="stylesheet" href="../../Content/Site.css" />

<title>@ViewBag.Title – поиск и бронирование</title>

@Styles.Render("~/Content/css")

@Scripts.Render("~/bundles/modernizr")

</head>

<body>

<nav class="navbar navbar-default navbar-inverse" role="navigation">

<div class="container-fluid">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target=".navbar-collapse">

<span class="sr-only">Toggle navigation</span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

<a href="@Url.Action("Search","Search")"><img src="~/Ресурсики/logo.png" /></a>

</div>

<div class="collapse navbar-collapse navbar-middle" id="bs-example-navbar-collapse-1">

<ul class="nav navbar-nav nav-pills">

<li>@Html.ActionLink("Поиск билетов", "Search", "Search")</li>

<li>@Html.ActionLink("О программе", "About", "Home")</li>

<li>@Html.ActionLink("Контакты", "Contact", "Home")</li>

@Html.Partial("\_LoginPartial")

</ul>

</div><!-- /.navbar-collapse -->

</div>

</div><!-- /.container-fluid -->

</nav>

<div class="wrapper">

<div class="container-fluid">

<div class="row">

<div class="container body-content">

@RenderBody()

</div>

</div>

<div class="clear"></div>

</div>

<div class="push"></div>

</div>

<footer>

<div class="container">

<p class="text-center">&copy; @DateTime.Now.Year – приложение ASP.NET</p>

</div>

</footer>

@Scripts.Render("~/bundles/jquery")

@Scripts.Render("~/bundles/bootstrap")

@RenderSection("scripts", required: false)

</body>

</html>

В шапке сайта располагается логотип сайта, а также навигационное меню (рис. 4.1). Меню состоит из следующих ссылок:

1. Поиск билетов
2. О программе
3. Контакты
4. Регистрация
5. Выполнить вход
6. Заказы и выход (в случае успешного прохождения авторизации).

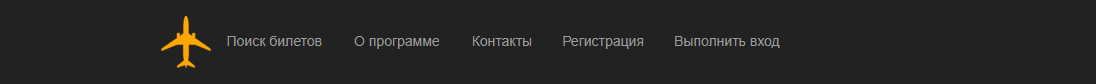


Рис. 4.1 Шапка сайта

## 4.2 Страница поиска рейсов

На данной странице происходит поиск рейсов по следующим параметрам (рис. 4.2):

* Поле «Откуда» - выпадающий список с названием городов, обязательное поле;
* Поле «Куда» - выпадающий список с названием городов, обязательное поле;
* Поле «Дата отправления» - поле с типом данных Date, обязательное поле;
* Поле «Дата возвращения» - поле с типом данных Date, необязательное поле при выборе флажка «Билет в одну сторону»;
* Поле «Количество пассажиров» - поле с типом данных number.;
* Флажок «Билет в одну сторону»;
* Флажок «Возможность одной пересадки».

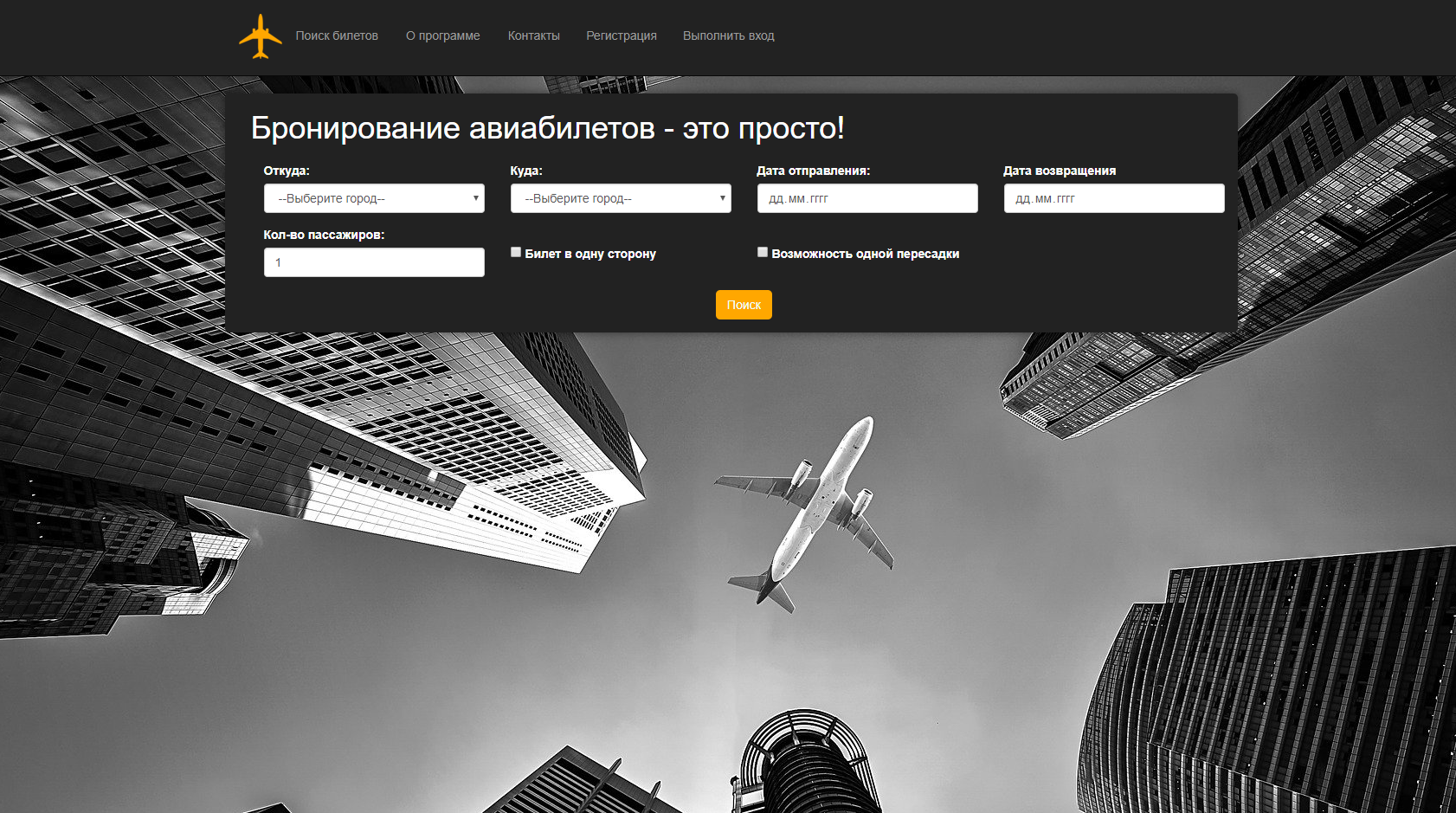


Рис. 4.2 Параметры поиска рейсов

Для вывода данной страницы используется метод Search контроллера SearchController, который передает на страницу данные для выпадающих списков.

Листинг 4.2.1 Метод, возвращающий представление поиска рейсов

public ActionResult Search ()

{

ViewBag.Cities = new SelectList(db.Cities, "ID", "Name");

return View();

}

Листинг 4.2.2 Страница Search

@{

ViewBag.Title = "Search";

}

@Scripts.Render("~/scripts/jquery-3.3.1.min.js")

@Scripts.Render("~/scripts/jquery.unobtrusive-ajax.min.js")

@Scripts.Render("~/scripts/jquery-3.3.1.js")

@Scripts.Render("~/scripts/bootstrap.js")

<div class="searchZone">

<div class="container">

<h1>Бронирование авиабилетов - это просто!</h1>

<h3></h3>

<form action="/Search/SearchResult" data-ajax="true" data-ajax-mode="replace" data-ajax-update="#results" id="form0" method="post">

<div class="form-row">

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label for="from">Откуда: </label>

@Html.DropDownList("from", ViewBag.Cities as SelectList, "--Выберите город--",

htmlAttributes: new { @class = "form-control"})

</div>

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label class="control-label ">Куда: </label>

@Html.DropDownList("to", ViewBag.Cities as SelectList, "--Выберите город--",

htmlAttributes: new { @class = "form-control" })

</div>

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label for="date">Дата отправления: </label>

<input type="date" class="form-control" id="dateTo" name="dateTo" required />

</div>

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label for="date">Дата возвращения </label>

<input type="date" class="form-control" id="dateFrom" name="dateFrom" required />

</div>

<div class="form-group col-lg-3t col-md-3">

<label for="person">Кол-во пассажиров: </label>

<input type="number" class="form-control" id="person" name="person" value="1" min="1" />

</div>

<div class="form-group col-lg-3 col-md-4 col-sm-4">

<br>

@Html.CheckBox("oneway", false, htmlAttributes: new { @class = "custom-control-input" })

<label for="oneway">Билет в одну сторону </label>

</div>

<div class="form-group col-lg-3 col-md-4 col-sm-4">

<br>

@Html.CheckBox("transfer", false)

<label for="transfer">Возможность одной пересадки </label>

</div>

</div>

<div class="form-row submitbtn">

<div class="form-group col-lg-12 col-md-12 col-sm-12">

<button type="submit" class="btn btnOrder desight">Поиск</button>

</div>

</div>

</form>

</div>

</div>

<div class="container-fluid" id="results"></div>

Листинг 4.2.3 Скрипт, отвечающий за доступность поля «Дата возвращения»

document.getElementById('oneway').onclick = function () {

if (document.getElementById('oneway').checked) {

$("#dateFrom").prop("disabled", true);

$("#dateFrom").prop('required', false);

$("#dateFrom").attr("value", "2018-01-01");

}

else {

$("#dateFrom").prop("disabled", false);

$("#dateFrom").prop('required', true);

}

}

Для отображения найденных рейсов используется технология Ajax, которая позволяет реализовать частичное обновление содержимого на странице, а также частичные представления. По своему действию частичное представление похоже на обычное, только для него по умолчанию не определяется мастер-страница. Подстановку необходимого элемента позволяет выполнить команда @Html.Partial("Partial").

При нажатии на кнопку «Найти» управление передается методу SearchResult контроллера SearchController.

Листинг 4.2.4 Метод SearchResult

public ActionResult SearchResult(int from, int to, DateTime dateTo, bool oneway, bool transfer, DateTime? dateFrom, int person)

{

DateTime endTime2 = new DateTime();

if (dateFrom != null)

{

endTime2 = Convert.ToDateTime(dateFrom);

}

var flights = db.Flights.Include(p => p.Plane).Include(p => p.DeparturePlace).Include(p => p.ArrivalPlace).ToList();

IEnumerable<Flight> flightsFrom = new List<Flight>(); ;

IEnumerable<TransferFlight> transferflightsTo = new List<TransferFlight>();

IEnumerable<TransferFlight> transferflightsFrom = new List<TransferFlight>();

SearchModel sm;

DateTime endTime = dateTo.AddDays(1);

var flightsTo = flights.Where(a => a.ArrivalPlaceID == from && a.DeparturePlaceID == to && a.Arrival >= dateTo && a.Arrival < endTime && a.numberOfSeats>=person).ToList();

if (oneway)

{

if (transfer)

{

transferflightsTo = getFlights(from, to, dateTo, person).ToList();

}

if (flightsTo.Count <= 0 && transferflightsTo.Count() <= 0)

{

return PartialView("NotFound");

}

}

else

{

DateTime time = endTime2.AddDays(1);

flightsFrom = flights.Where(a => a.ArrivalPlaceID == to && a.DeparturePlaceID == from && a.Arrival >= dateFrom && a.Arrival < time && a.numberOfSeats >= person).ToList();

if (transfer)

{

transferflightsTo = getFlights(from, to, dateTo, person).ToList();

transferflightsFrom = getFlights(to, from, endTime2, person).ToList();

}

if (flightsTo.Count <= 0 && transferflightsTo.Count() <= 0 && flightsFrom.Count() <= 0 && transferflightsFrom.Count() <= 0)

{

return PartialView("NotFound");

}

}

sm = new SearchModel { flightsTo = flightsTo, flightsFrom = flightsFrom, transferFlightsTo = transferflightsTo, transferFlightsFrom = transferflightsFrom, Transfer = transfer, OneWay = oneway, Person = person};

return PartialView(sm);

}

public IEnumerable<TransferFlight> getFlights (int from, int to, DateTime dateTo, int person)

{

var flights = db.Flights.Include(p => p.Plane).Include(p => p.DeparturePlace).Include(p => p.ArrivalPlace).ToList();

List<TransferFlight> transferFlights = new List<TransferFlight>();

DateTime endTime = dateTo.AddDays(1);

var firstFlight = flights.Where(f => f.ArrivalPlaceID == from && f.Arrival >= dateTo && f.Arrival < endTime && f.DeparturePlaceID != to && f.numberOfSeats>=person).ToList();

foreach (var item in firstFlight)

{

var endTime2 = item.Departure.AddHours(3);

var endTime3 = item.Departure.AddHours(10);

var secondFlight = flights.Where(f => f.ArrivalPlaceID == item.DeparturePlaceID && f.Arrival >= endTime2 && f.Arrival <= endTime3 && f.DeparturePlaceID == to && f.numberOfSeats >= person).ToList();

foreach (var f in secondFlight)

{

transferFlights.Add(new TransferFlight(item, f));

}

}

return transferFlights;

}

За отображение найденных рейсов отвечает частичное представление SearchResult, которое, в свою очередь, в зависимости от параметров поиска, отображает частичные представления SingleFlight (рейс без пересадок), TransferFlight (рейс с пересадками) и NotFound (в случае, если рейсов не найдено).

Листинг 4.2.5 Частичное представление SearchResult

@model Airlines.Models.SearchModel

<div id="results">

@if (Model.OneWay)

{

foreach (var item in Model.flightsTo)

{

<div class="container-fluid flight">

@Html.Partial("SingleFlight", item)

<div class="col-lg-2 btnZone">

@using (Html.BeginForm("ChooseSeats", "Book", FormMethod.Post))

{

@Html.Hidden("FlightID", item.ID)

@Html.Hidden("Person", Model.Person)

<button class="btn btnOrder" type="submit"><strong>Забронировать</strong></button>

}

</div>

</div>

<div class="col-lg-12 empty"></div>

}

if (Model.Transfer)

{

foreach (var item in Model.transferFlightsTo)

{

<div class="container-fluid flight">

@Html.Partial("TransferFlight", item)

<div class="col-lg-2 btnZone">

@using (Html.BeginForm("ChooseSeats", "Book", FormMethod.Post))

{

@Html.Hidden("FlightID", item.FirstFlight.ID)

@Html.Hidden("FlightID", item.SecondFlight.ID)

@Html.Hidden("Person", Model.Person)

<button class="btn btnOrder" type="submit"><strong>Забронировать</strong></button>

}

</div>

</div>

<div class="col-lg-12 empty"></div>

}

}

}

@if (!Model.OneWay)

{

foreach (var item in Model.flightsTo)

{

int check = Model.flightsFrom.Count();

for (int i = 0; i < check; i++)

{

<div class="container-fluid flight">

@{ Airlines.Models.Flight f = Model.flightsFrom.ElementAt(i);

Html.RenderPartial("SingleFlight", item);

Html.RenderPartial("SingleFlight", f);

}

<div class="col-lg-2 btnZone">

@using (Html.BeginForm("ChooseSeats", "Book", FormMethod.Post))

{

@Html.Hidden("FlightID", item.ID)

@Html.Hidden("FlightID", f.ID)

@Html.Hidden("Person", Model.Person)

<button class="btn btnOrder"><strong>Забронировать</strong></button>

}

</div>

</div>

<div class="col-lg-12 empty"></div>

}

}

if (Model.Transfer)

{

foreach (var item in Model.flightsTo)

{

int check = Model.transferFlightsFrom.Count();

for (int i = 0; i < check; i++)

{

<div class="container-fluid flight">

@{ Airlines.Models.TransferFlight f = Model.transferFlightsFrom.ElementAt(i);

Html.RenderPartial("SingleFlight", item);

Html.RenderPartial("TransferFlight", f);

}

<div class="col-lg-2 btnZone">

@using (Html.BeginForm("ChooseSeats", "Book", FormMethod.Post))

{

@Html.Hidden("FlightID", item.ID)

@Html.Hidden("FlightID", f.FirstFlight.ID)

@Html.Hidden("FlightID", f.SecondFlight.ID)

@Html.Hidden("Person", Model.Person)

<button class="btn btnOrder"><strong>Забронировать</strong></button>

}

</div>

</div>

<div class="col-lg-12 empty"></div>

}

}

foreach (var item in Model.transferFlightsTo)

{

int check = Model.flightsFrom.Count();

for (int i = 0; i < check; i++)

{

<div class="container-fluid flight">

@{Airlines.Models.Flight f = Model.flightsFrom.ElementAt(i);

Html.RenderPartial("TransferFlight", item);

Html.RenderPartial("SingleFlight", f);

}

<div class="col-lg-2 btnZone">

@using (Html.BeginForm("ChooseSeats", "Book", FormMethod.Post))

{

@Html.Hidden("FlightID", item.FirstFlight.ID)

@Html.Hidden("FlightID", item.SecondFlight.ID)

@Html.Hidden("FlightID", f.ID)

@Html.Hidden("Person", Model.Person)

<button class="btn btnOrder"><strong>Забронировать</strong></button>

}

</div>

</div>

<div class="col-lg-12 empty"></div>

}

}

foreach (var item in Model.transferFlightsTo)

{

int check = Model.transferFlightsFrom.Count();

for (int i = 0; i < check; i++)

{

<div class="container-fluid flight">

@{ Airlines.Models.TransferFlight f = Model.transferFlightsFrom.ElementAt(i);

Html.RenderPartial("TransferFlight", item);

Html.RenderPartial("TransferFlight", f);

}

<div class="col-lg-2 btnZone">

@using (Html.BeginForm("ChooseSeats", "Book", FormMethod.Post))

{

@Html.Hidden("FlightID", item.FirstFlight.ID)

@Html.Hidden("FlightID", item.SecondFlight.ID)

@Html.Hidden("FlightID", f.FirstFlight.ID)

@Html.Hidden("FlightID", f.SecondFlight.ID)

@Html.Hidden("Person", Model.Person)

<button class="btn btnOrder"><strong>Забронировать</strong></button>

}

</div>

</div>

<div class="col-lg-12 empty"></div>

}

}

}

}

</div>

Листинг 4.2.6 Частичное представление SingleFlight

@model Airlines.Models.Flight

<div class="col-lg-10 col-md-10 col-sm-10 col-xs-6 tickets">

<div class="col-lg-12 col-md-12 col-sm-12 col-xs-12 mainZone">

<div class="container col-lg-12 col-md-12 col-sm-12 col-xs-12">

<p class="airline">@Model.Airline</p>

<div class="col-lg-3 col-md-3 col-sm-3 col-xs-4 left">

<p class="time">@Model.Arrival.ToString("HH:mm")</p>

<p class=" info">@Model.ArrivalPlace.Name</p>

<p class="info">@Model.Arrival.ToString("dd.MM.yyyy")</p>

</div>

<div class="col-lg-6 col-md-6 col-sm-6 col-xs-4 center">

<div>

<p class="tripTime">Время в пути: @Model.TravelTime.Hours ч. @Model.TravelTime.Minutes мин.</p>

</div>

<div class="line hidden-xs">

</div>

<div class="col-lg-1 col-md-1 col-sm-1 departurePlane hidden-xs">

<svg class="svgD" width="24" height="24" viewBox="0 0 24 24" fill="#A0B0B9" xmlns="http://www.w3.org/2000/svg">

<path fill-rule="evenodd" clip-rule="evenodd" d="M17 13H14C12.4577 11.5739 10.1159 9.82474 8.16374 8.01001L6.84295 8C7.60515 9.49106 8.69945 11.4376 9.5 13H6.5C5.54148 12.2794 4.45139 11.6727 3.69008 11.1028C3.27567 10.7932 2.80644 11.0649 3.08297 11.6043C4.38443 14.1509 4.90267 15.176 4.90267 15.176C5.12943 15.6181 5.48619 15.9964 6 16H19.7317C20.7473 15.9659 20.9885 15.3663 21 14.7944C21 13.5 19.5675 13.0196 17 13Z">

</path>

</svg>

<p class="airport">@Model.ArrivalPlace.Airport</p>

</div>

<div class="col-lg-9 col-md-9 col-sm-9 hidden-xs"></div>

<div class="col-lg-1 col-md-1 col-sm-1 arrivalPlane hidden-xs">

<svg class="svgA" width="24" height="24" viewBox="0 0 24 24" fill="#A0B0B9" xmlns="http://www.w3.org/2000/svg">

<path fill-rule="evenodd" clip-rule="evenodd" d="M17 13H14C12.4577 11.5739 10.1159 9.82474 8.16374 8.01001L6.84295 8C7.60515 9.49106 8.69945 11.4376 9.5 13H6.5C5.54148 12.2794 4.45139 11.6727 3.69008 11.1028C3.27567 10.7932 2.80644 11.0649 3.08297 11.6043C4.38443 14.1509 4.90267 15.176 4.90267 15.176C5.12943 15.6181 5.48619 15.9964 6 16H19.7317C20.7473 15.9659 20.9885 15.3663 21 14.7944C21 13.5 19.5675 13.0196 17 13Z">

</path>

</svg>

<p class="airport">@Model.DeparturePlace.Airport</p>

</div>

</div>

<div class="col-lg-3 col-md-3 col-sm-3 col-xs-4 right">

<p class="time">@Model.Departure.ToString("HH:mm")</p>

<p class="info">@Model.DeparturePlace.Name</p>

<p class="info">@Model.Departure.ToString("dd.MM.yyyy")</p>

</div>

</div>

</div>

</div>

Листинг 4.2.7 Частичное представление TransferFlight

@model Airlines.Models.TransferFlight

<div class="col-lg-10 col-md-10 col-sm-10 col-xs-6 tickets">

<div class="col-lg-12 col-md-12 col-sm-12 col-xs-12 mainZone">

<div class="container col-lg-12 col-md-12 col-sm-12 col-xs-12">

<p class="airline">@Model.FirstFlight.Airline + @Model.SecondFlight.Airline</p>

<div class="col-lg-3 col-md-3 col-sm-3 col-xs-4 left">

<p class="time">@Model.FirstFlight.Arrival.ToString("HH:mm")</p>

<p class="info">@Model.FirstFlight.ArrivalPlace.Name</p>

<p class="info">@Model.FirstFlight.Arrival.ToString("dd.MM.yyyy")</p>

</div>

<div class="col-lg-6 col-md-6 col-sm-6 col-xs-4 center">

<div>

<p class="tripTime">Время в пути: @Model.TravelTime.Hours ч. @Model.TravelTime.Minutes мин.</p>

</div>

<div class="col-lg-4 col-md-4 col-sm-4 hidden-xs line">

</div>

<div class="col-lg-4 col-md-4 col-sm-4 hidden-xs transferLine">

<p class="transferAirport">@Model.FirstFlight.DeparturePlace.Airport</p>

</div>

<div class="col-lg-4 col-md-4 col-sm-4 hidden-xs line">

</div>

<div class="col-lg-1 col-md-1 col-sm-1 hidden-xs departurePlane">

<svg class="svgD" width="24" height="24" viewBox="0 0 24 24" fill="#A0B0B9" xmlns="http://www.w3.org/2000/svg">

<path fill-rule="evenodd" clip-rule="evenodd" d="M17 13H14C12.4577 11.5739 10.1159 9.82474 8.16374 8.01001L6.84295 8C7.60515 9.49106 8.69945 11.4376 9.5 13H6.5C5.54148 12.2794 4.45139 11.6727 3.69008 11.1028C3.27567 10.7932 2.80644 11.0649 3.08297 11.6043C4.38443 14.1509 4.90267 15.176 4.90267 15.176C5.12943 15.6181 5.48619 15.9964 6 16H19.7317C20.7473 15.9659 20.9885 15.3663 21 14.7944C21 13.5 19.5675 13.0196 17 13Z">

</path>

</svg>

<p class="airport">@Model.FirstFlight.ArrivalPlace.Airport</p>

</div>

<div class="col-lg-9 col-md-9 col-sm-9 hidden-xs"></div>

<div class="col-lg-1col-md-1 col-sm-1 arrivalPlane hidden-xs">

<svg class="svgA" width="24" height="24" viewBox="0 0 24 24" fill="#A0B0B9" xmlns="http://www.w3.org/2000/svg">

<path fill-rule="evenodd" clip-rule="evenodd" d="M17 13H14C12.4577 11.5739 10.1159 9.82474 8.16374 8.01001L6.84295 8C7.60515 9.49106 8.69945 11.4376 9.5 13H6.5C5.54148 12.2794 4.45139 11.6727 3.69008 11.1028C3.27567 10.7932 2.80644 11.0649 3.08297 11.6043C4.38443 14.1509 4.90267 15.176 4.90267 15.176C5.12943 15.6181 5.48619 15.9964 6 16H19.7317C20.7473 15.9659 20.9885 15.3663 21 14.7944C21 13.5 19.5675 13.0196 17 13Z">

</path>

</svg>

<p class="airport">@Model.SecondFlight.DeparturePlace.Airport</p>

</div>

</div>

<div class="col-lg-3 col-md-3 col-sm-3 col-xs-4 right">

<p class="time">@Model.SecondFlight.Departure.ToString("HH:mm")</p>

<p class="info">@Model.SecondFlight.DeparturePlace.Name</p>

<p class="info">@Model.SecondFlight.Departure.ToString("dd.MM.yyyy")</p>

</div>

</div>

</div>

</div>

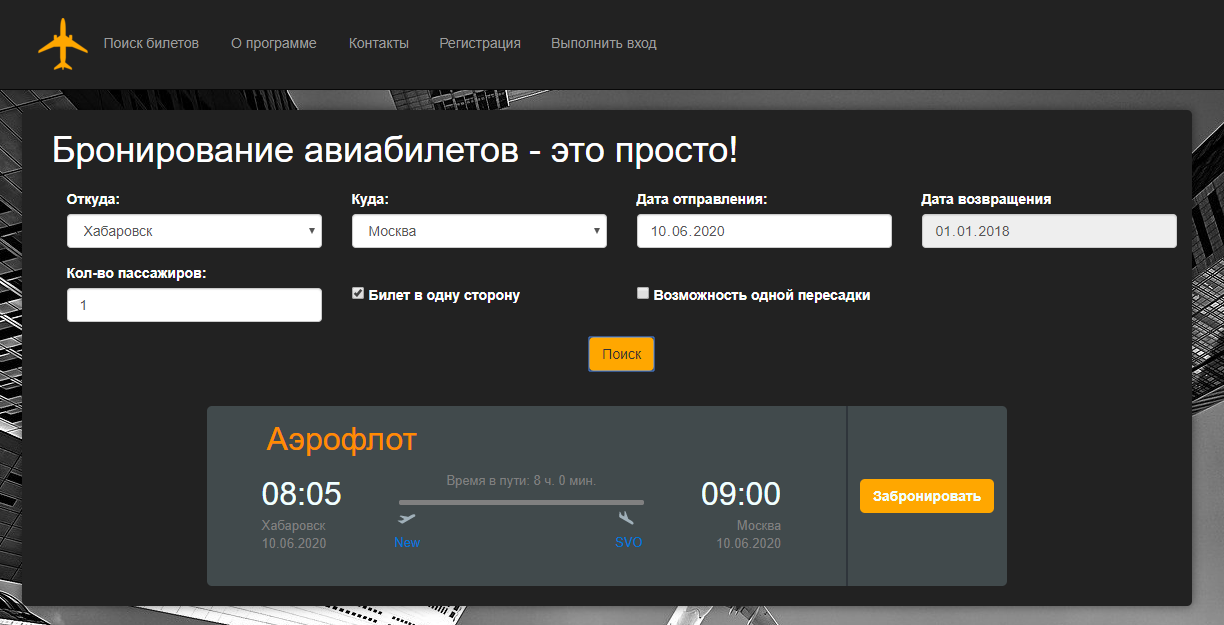


Рис. 4.3 Поиск рейса в одну сторону без возможности пересадки

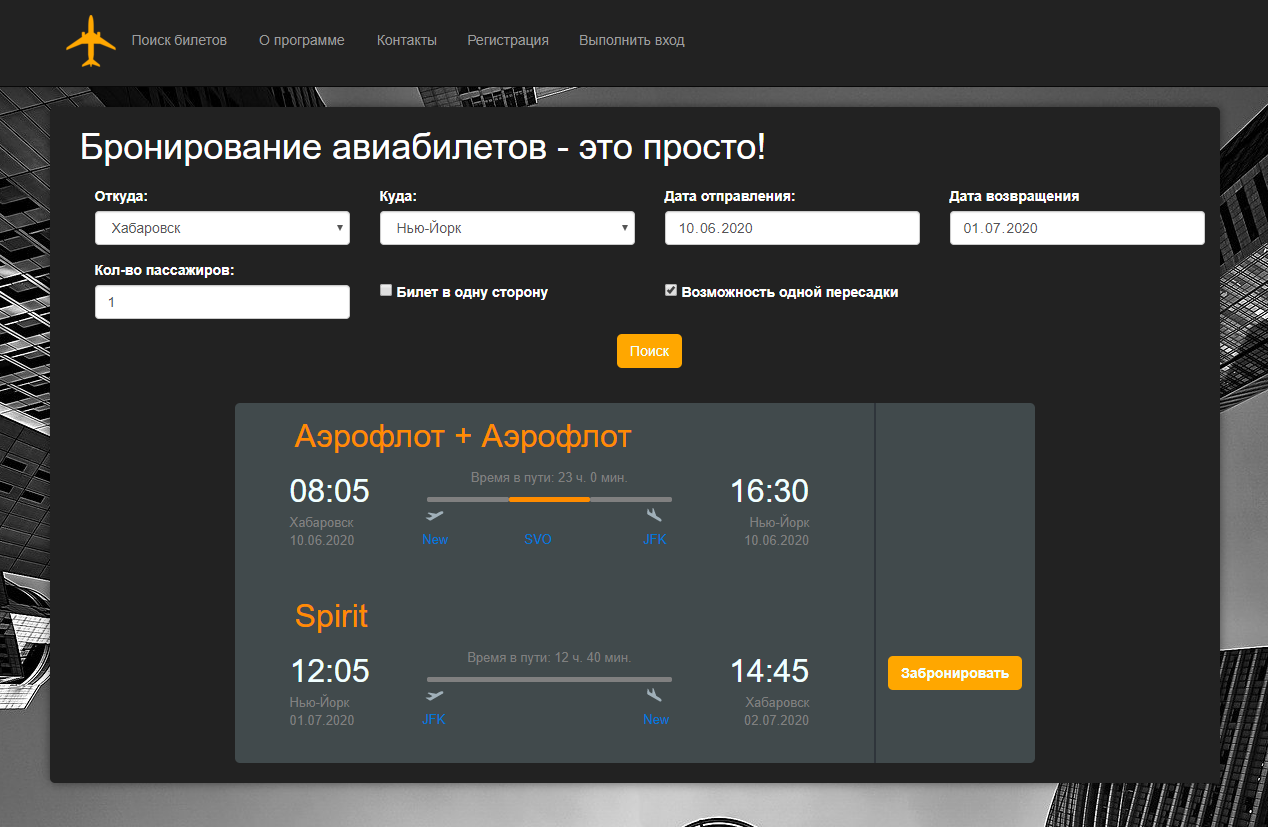


Рис. 4.4 Поиск рейса в обе стороны с возможностью пересадки

## 4.3 Страница ввода информации о пассажирах и выбора мест в салоне

После выбора рейса(-ов) по нажатию на кнопку «Забронировать» пользователю необходимо ввести данные о пассажирах и выбрать места в салоне (только для первого рейса). За это отвечает представление «ChooseSeats», которое возвращается методом ChooseSeats контроллера BookController. При этом стоит отметить, что данная страница доступна только авторизированным пользователям.

Листинг 4.3.1 Метод ChooseSeats

public ActionResult ChooseSeats(List<int> FlightID, int Person = 1)

{

if (FlightID == null)

{

return RedirectToAction("Search","Search");

}

var flights = new List<Flight>();

foreach (var item in FlightID)

{

flights.Add(db.Flights.Find(item));

}

var flight = flights.First();

var tickets = db.Tickets.Where(c => c.FlightID == flight.ID).ToList();

var seats = new List<string>();

foreach (var item in tickets)

{

var id = item.SeatID;

var seat = db.Seats.Where(c => c.SeatID == id).First();

seats.Add(seat.SeatName);

}

var bm = new BookModel() { Person = Person, Flights = flights, Seats = seats};

return View(bm);

}

Листинг 4.3.2 Представление ChooseSeats

@model Airlines.Models.BookModel

@{

ViewBag.Title = "ChooseSeats";

}

<h2>Информация о пассажирах</h2>

<h3>Внимательно введите данные пассажиров</h3>

@using (Html.BeginForm("Confirmation", "Book", FormMethod.Post))

{

for (int i = 0; i < Model.Person; i++)

{

<h2>@(i + 1) пассажир</h2>

<div class="container">

<div class="form-row">

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label for="Surname">Фамилия</label>

<input type="text" class="form-control" name="customers[@i].Surname" id="Surname" placeholder="Фамилия" required>

</div>

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label for="Name">Имя</label>

<input type="text" class="form-control" name="customers[@i].Name" id="Name" placeholder="Имя" required>

</div>

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label for="Patronymic">Отчество</label>

<input type="text" class="form-control" name="customers[@i].Patronymic" id="Patronymic" placeholder="Отчество">

</div>

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label for="Sex">Пол</label>

@Html.DropDownList("customers["+@i+"].Sex", new SelectList ( new string[] { "Мужской", "Женский" } ),

htmlAttributes: new { @class = "form-control" })

</div>

</div>

<div class="form-row">

<div class="form-group col-lg-3 col-md-3 col-sm-6 ">

<label for="BirthDate">Дата рождения</label>

<input type="date" class="form-control" name="customers[@i].BirthDate" id="BirthDate" placeholder="Дата рождения" required>

</div>

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label for="PassportNumber">Паспорт</label>

<input type="text" class="form-control" name="customers[@i].PassportNumber" id="PassportNumber" placeholder="Серия и № паспорта" required>

</div>

<div class="form-group col-lg-3 col-md-3 col-sm-6">

<label for="Citizenship">Гражданство</label>

<input type="text" class="form-control" name="customers[@i].Citizenship" id="Citizenship" placeholder="Гражданство" required>

</div>

</div>

</div>

}

for (int i = 0; i < Model.Flights.Count(); i++)

{

@Html.Hidden("flights[" + i + "]", Model.Flights.ElementAt(i).ID);

}

@Html.Hidden("person", Model.Person);

for (int i = 0; i < Model.Person; i++)

{

@Html.Hidden("seats[" + i + "]", "");

}

<div>

@Html.ListBox("ListSeats", new MultiSelectList(Model.Seats), htmlAttributes: new { @class = "ListBox" })

</div>

<div>

<h2>Выбор мест</h2>

<h3>Выберете место(-а) в салоне самолета (только для первого рейса).</h3>

<div class="col-lg-10 col-md-10 col-sm-10 col-xs-10 container planeModel">

<div class="rows">

<div class="buisness"></div>

<div class="empty1"></div>

<div class="econom"></div>

</div>

</div>

<div class="col-lg-2 col-md-2 col-sm-2 col-xs-2">

<button type="submit" id="btn" class="btn btnOrder desight">Подтвердить</button>

</div>

</div>

}

<input id="p" type="hidden" value="@Model.Person" />

<input id="c" type="hidden" value="0" />

<!--секция скриптов-->

@section Scripts {

@Scripts.Render("~/scripts/PlaneSeatsEconom.js")

}

Данное представление использует скрипт, который выводит на экран места салона, а также делает неактивными те места, которые ранее были забронированы.

Листинг 4.3.3 Скрипт отрисовки салона самолета

k = 1;

kb = 1;

tbk = 0;

counter = 0;

$('#btn').attr("disabled", true);

var PlaneHalle = {

row: [6, 6, 6, 6, 6, 6, 6, 6]

},

PlaneHallMape = '';

$.each(PlaneHalle.row, function (row, numberOfSeats) {

PlaneHallRowe = '';

for (i = 1; i <= numberOfSeats/2; i++) {

PlaneHallRowe += '<div class="Eseat" id="'+ k + '/' + i + '/' + 'e' + '"></div>';

}

PlaneHallRowe += '<div class="emptyBtwSeatsEconom"></div>';

for (i = numberOfSeats / 2 +1; i <= numberOfSeats; i++) {

PlaneHallRowe += '<div class="Eseat" id="'+ k + '/' + i + '/' + 'e' + '"></div>';

}

PlaneHallMape += PlaneHallRowe + '<div class="passageBetween">&nbsp;</div>';

k++;

});

var PlaneHallb = {

row: [2, 2, 2]

},

PlaneHallMapb = '';

$.each(PlaneHallb.row, function (row, numberOfSeats) {

PlaneHallRowb = '';

for (i = 1; i <= numberOfSeats / 2; i++) {

PlaneHallRowb += '<div class="Bseat" id="'+ kb + '/' + i + '/' + 'b' + '"></div>';

}

PlaneHallRowb += '<div class="emptyBtwSeatsBuisness"></div>';

for (i = numberOfSeats / 2 + 1; i <= numberOfSeats; i++) {

PlaneHallRowb += '<div class="Bseat" id="'+ kb + '/' + i + '/' + 'b' + '"></div>';

}

PlaneHallRowb += '<div class="emptyBuisness"></div>';

PlaneHallMapb += PlaneHallRowb + '<div class="passageBetween">&nbsp;</div>';

kb++;

});

$('.econom').html(PlaneHallMape);

$('.buisness').html(PlaneHallMapb);

$('.Eseat').on('click', function (e) {

var counter = $('#c').val();

var id = e.currentTarget.id;

if (e.currentTarget.classList.contains('no') == true) { }

else {

if (counter < $('#p').val() && e.currentTarget.classList.contains('bay') == false) {

$(e.currentTarget).toggleClass('bay', true);

counter++;

$('#c').val(counter);

for (i = 0; i < $('#p').val(); i++) {

if ($('#seats\_' + i + '\_').val() == "") {

$('#seats\_' + i + '\_').val(id);

break;

}

}

if (counter < $('#p').val()) {

$('#btn').attr("disabled", true);

}

else {

$('#btn').attr("disabled", false);

}

}

else {

if (e.currentTarget.classList.contains('bay')) {

var empty = "";

$(e.currentTarget).toggleClass('bay', false);

counter--;

$('#c').val(counter);

for (i = 0; i < $('#p').val(); i++) {

if ($('#seats\_' + i + '\_').val() == id) {

$('#seats\_' + i + '\_').val(empty);

break;

}

}

$('#btn').attr("disabled", true);

}

}

}

});

$('.Bseat').on('click', function (e) {

var counter = $('#c').val();

var id = e.currentTarget.id;

if (e.currentTarget.classList.contains('no') == true) { }

else {

if (counter < $('#p').val() && e.currentTarget.classList.contains('bay') == false) {

$(e.currentTarget).toggleClass('bay', true);

counter++;

$('#c').val(counter);

for (i = 0; i < $('#p').val(); i++) {

if ($('#seats\_' + i + '\_').val() == "") {

$('#seats\_' + i + '\_').val(id);

break;

}

}

if (counter < $('#p').val()) {

$('#btn').attr("disabled", true);

}

else {

$('#btn').attr("disabled", false);

}

}

else {

if (e.currentTarget.classList.contains('bay')) {

var empty = "";

$(e.currentTarget).toggleClass('bay', false);

counter--;

$('#c').val(counter);

for (i = 0; i < $('#p').val(); i++) {

if ($('#seats\_' + i + '\_').val() == id) {

$('#seats\_' + i + '\_').val(empty);

break;

}

}

$('#btn').attr("disabled", true);

}

}

}

});

for (i = 0; i < $('#ListSeats').children().length; i++) {

var valu = document.getElementById("ListSeats").options[i].text;

document.getElementById(valu).classList.add("no");

}

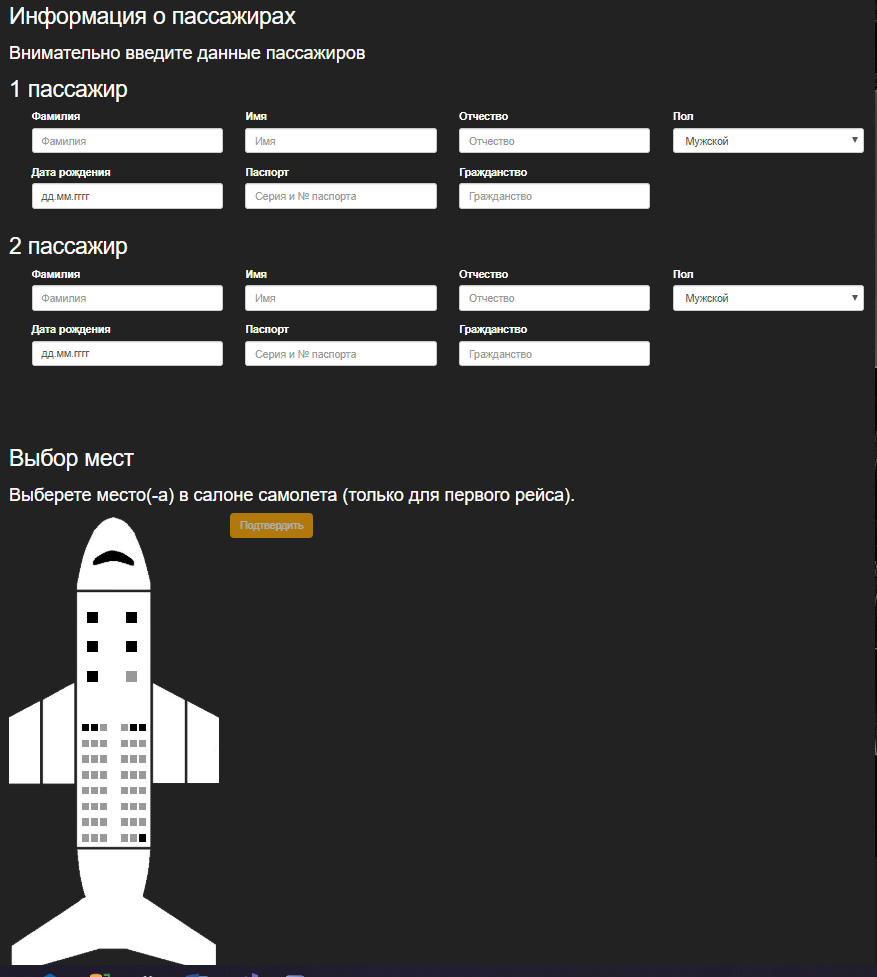


Рис. 4.5 Страница ввода информации о местах и пассажирах

## 4.4 Страница подтверждения заказа

После заполнения всех необходимых полей происходит отправление данных методу Confirmation контроллера BookController. Так как во время ввода данных и выбора мест другой пользователь забронировать некоторые места в салоне, в этом методе, в первую очередь, происходит проверка выбранных пользователем мест (забронированы ли они кем-то другим). Если такие места есть, происходит перезагрузка страницы ChooseSeats. Если нет, то происходит создание заказа, пассажиров и билетов с последующим занесением в базу данных, а также вывод представления Confirmation.

Листинг 4.4.1 Метод Confirmation

public ActionResult Confirmation(List<Customer> customers, List<int> flights, List<string> seats, int person)

{

var tickets = db.Tickets.Where(c => c.FlightID == flights.FirstOrDefault()).ToList();

var seatsNames = new List<string>();

foreach (var item in tickets)

{

var id = item.SeatID;

var seat = db.Seats.Where(c => c.SeatID == id).First();

seatsNames.Add(seat.SeatName);

}

bool check=false;

foreach (var item in seats)

{

foreach (var i in seatsNames)

{

if (item == i)

{

check = true;

}

}

}

if (check)

{

var fl = new List<Flight>();

foreach (var item in flights)

{

fl.Add(db.Flights.Find(item));

}

var bm = new BookModel() { Person = person, Flights = fl, Seats = seatsNames };

return View("ChooseSeats", bm);

}

Order order = new Order();

order.UserId = User.Identity.GetUserId();

order.OrderTime = DateTime.Now;

db.Orders.Add(order);

db.SaveChanges();

foreach (var item in customers)

{

db.Customers.Add(item);

db.SaveChanges();

}

var plane = db.Flights.Find(flights.First()).PlaneID;

var s = db.Seats.Where(c => c.PlaneID == plane);

var seatsID = new List<int>();

foreach (var item in seats)

{

seatsID.Add(db.Seats.Where(c => c.PlaneID == plane && c.SeatName == item).First().SeatID);

}

for (int i = 0; i<flights.Count;i++)

{

for (int j = 0; j< customers.Count;j++)

{

if (i==0)

{

db.Tickets.Add(new Ticket(customers[j].CustomerID, flights.ElementAt(i), seatsID.ElementAt(j), order.ID));

}

else

db.Tickets.Add(new Ticket(customers[j].CustomerID, flights.ElementAt(i), order.ID));

}

}

db.SaveChanges();

return View();

}

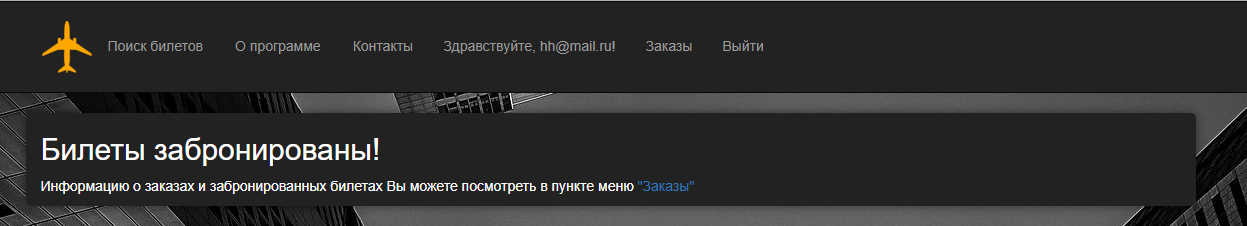


Рис. 4.6 Представление Confirmation

4.5 Страница заказов

Авторизированному пользователя доступна страница «Заказы», в которой отображаются все оформленные им заказы с бронированием. За вывод этой страницы отвечает метод Orders контроллера ManageController.

Листинг 4.5.1 Метод Orders

public ActionResult Orders()

{

var userId = User.Identity.GetUserId();

var orders = db.Orders.Where(c => c.UserId == userId).Include(c => c.Tickets).ToList();

foreach (var i in orders)

{

foreach (var item in i.Tickets)

{

item.Flight = db.Flights.Where(c => c.ID == item.FlightID).Include(p => p.DeparturePlace).Include(p => p.ArrivalPlace).First();

}

}

ViewBag.Orders = orders;

return View();

}

Листинг 4.5.2 Представление Orders

@{

ViewBag.Title = "История бронирований";

}

<div class="container">

<h2>История бронирований</h2>

<table class="table">

<thead>

<tr>

<th scope="col">#</th>

<th scope="col">Дата бронирования</th>

<th scope="col">Маршрут</th>

<th scope="col">Количество билетов</th>

<th scope="col"></th>

</tr>

</thead>

<tbody>

@foreach (var item in ViewBag.Orders)

{

using (Html.BeginForm("Tickets", "Manage", FormMethod.Post))

{

<tr>

<th scope="row">@item.ID<input type="hidden" name="OrderID" value="@item.ID" /></th>

<td>@item.OrderTime</td>

<th>@item.Tickets[0].Flight.ArrivalPlace.Name - @item.Tickets[item.Tickets.Count - 1].Flight.DeparturePlace.Name</th>

<td>@item.Tickets.Count</td>

<td><button type="submit" class="btn btnOrder desight">Подробнее</button></td>

</tr>

}

}

</tbody>

</table>

</div>

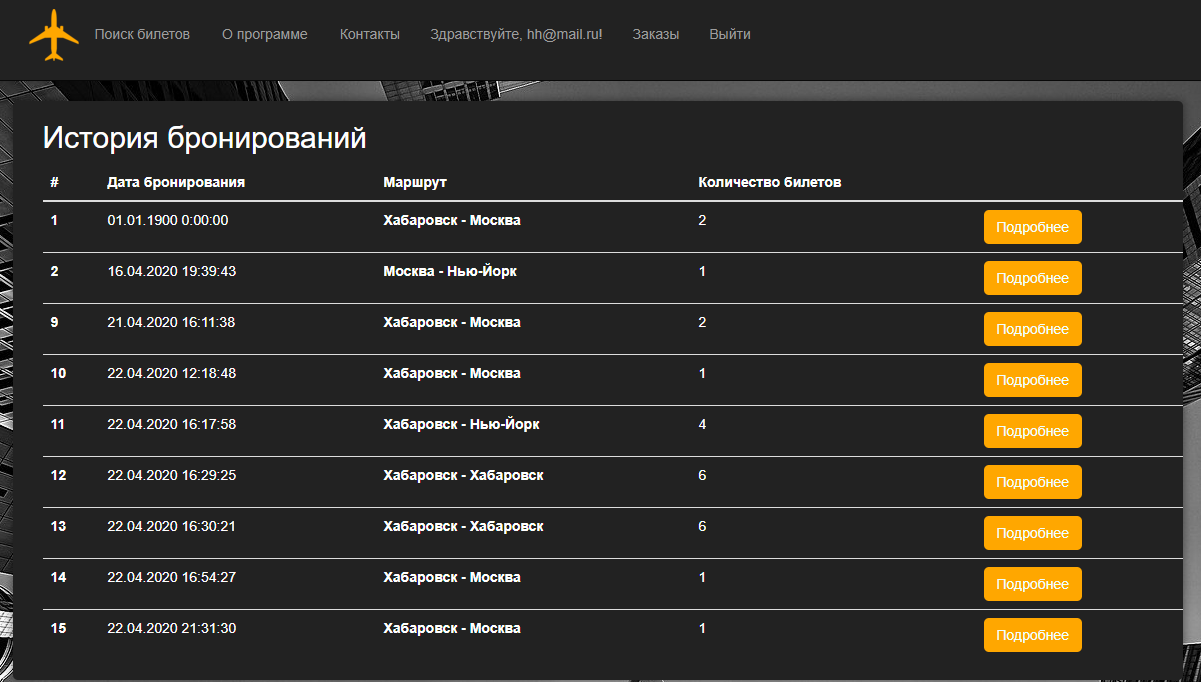


Рис. 4.7 Страница истории бронирований

## 4.6 Страница купленных билетов

При нажатии на кнопку «Подробнее» пользователь может просмотреть подробности заказа. За это отвечает метод Tickets.

Листинг 4.6.1 Метод Tickets

public ActionResult Tickets (int OrderID)

{

var tickets = db.Tickets.Where(c => c.OrderID == OrderID).Include(c => c.Seat).Include(c => c.Customer).Include(c => c.Flight).ToList();

foreach (var item in tickets)

{

item.Flight.ArrivalPlace = db.Cities.Where(c => c.ID == item.Flight.ArrivalPlaceID).FirstOrDefault();

item.Flight.DeparturePlace = db.Cities.Where(c => c.ID == item.Flight.DeparturePlaceID).FirstOrDefault();

}

var list = new ListOfTickets() { Tickets = tickets };

return View(list);

}

Листинг 4.6.2 Представление Tickets

@model Airlines.Models.ListOfTickets

@{

ViewBag.Title = "Купленные билеты";

}

<h2>Купленные билеты</h2>

@foreach (var item in Model.Tickets)

{

<div class="container " >

<table class="table" style="border: 2px solid darkorange; border-radius:5px; margin-left:-1%;">

<tbody>

<tr>

<th scope="row">ФИО</th>

<td class="col">@item.Customer.Surname @item.Customer.Name @item.Customer.Patronymic</td>

</tr>

<tr>

<th scope="row">ОТКУДА</th>

<td class="col">@item.Flight.ArrivalPlace.Name</td>

</tr>

<tr>

<th scope="row">КУДА</th>

<td class="col">@item.Flight.DeparturePlace.Name</td>

</tr>

<tr>

<th scope="row">ДАТА И ВРЕМЯ ОТЛЕТА</th>

<td class="col">@item.Flight.Arrival</td>

</tr>

<tr>

<th scope="row">ДАТА И ВРЕМЯ ПРИЛЕТА</th>

<td class="col">@item.Flight.Departure</td>

</tr>

<tr>

<th scope="row">АВИАКОМПАНИЯ</th>

<td class="col">@item.Flight.Airline</td>

</tr>

@if (item.Seat != null)

{

<tr>

<th scope="row">МЕСТО</th>

<td class="col">@item.Seat.SeatName</td>

</tr>

}

</tbody>

</table>

</div>

<br />

}

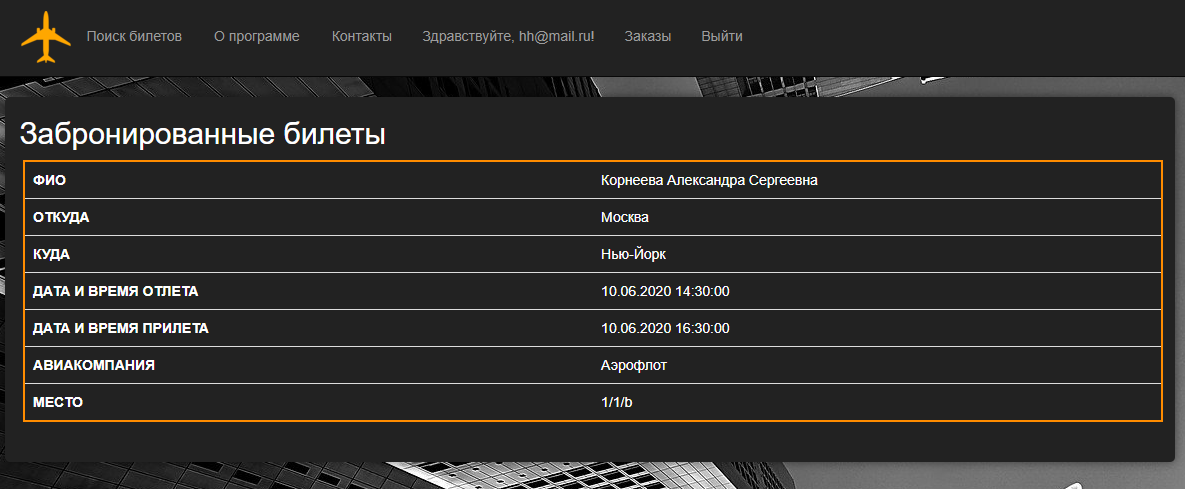


Рис. 4.8 Страница просмотра забронированных билетов

# 5 Заключение

В результате выполнения данного курсового проекта было разработано WEB-приложение «Авиабилеты». Были изучены и использованы различные инструменты и технологии разработку WEB-приложений, такие как паттерн MVC, язык программирования Java Script, библиотека JQuery, технология обращения к серверу AJAX, движок Razor, фреймворк Bootstrap и др. Также была спроектирована база, созданы информационная структура приложения и навигационная система, реализованы пользовательские компоненты, предусмотрена система авторизации.

# Список литературы

1. <https://metanit.com/sharp/mvc5>
2. <https://professorweb.ru/my/ASP_NET>
3. <https://getbootstrap.com/>
4. <https://docs.microsoft.com/ru-ru/aspnet/mvc/>