Первый этап курсового проектирования.

Выполнил студент гр. ИСТ-116 Сонин А.С.

Предметная область: «автоматизация и учет подвижного состава автобусного предприятия»

**1. Описание предметной области**

Автобусный транспорт - один из основных элементов благоустройства городов. Поэтому его развитие неразрывно связано с ростом местности городского населения и его материальным благосостоянием, т.к. пользование им позволяет значительно экономить время для поездок на работу, учебу и по культурно - бытовым целям. Сотрудникам автобусного предприятия приходится ежедневно сталкиваться с множеством проблем организационного, экономического и финансового характера, которые приходится решать интуитивно.

В данном курсовом проекте разрабатывается автоматизированная информационная система учета подвижного состава автобусного предприятия. Основной задачей данной системы является учет и автоматизация работы сотрудников на предприятии.

**2. Словарь предметной области**

VIN-номер (идентификационный номер транспортного средства) — уникальный код транспортного средства, состоящий из 17 символов. В коде представлена информация о производителе и характеристиках транспортного средства, и годе выпуска.

Регистрационный знак — специальный символический знак, изготовленный на металлические пластины, используемое для учёта автомобилей, мотоциклов, грузовой, специальной, строительной техники и вооружения, прицепов. Устанавливаются на передней и задней частях техники (на прицепы и мотоциклы — только сзади).

Путевой лист — основной первичный документ учёта работы водителя и пробега, маршрута автомобиля, выдаваемый ежедневно водителям транспортных средств.

Диспетчер — должностное лицо, отвечающее за координацию каких-либо действий в определённой сфере.

**3. Роли**

*1. Администратор.*

Человек, администрирующий систему. Может редактировать информацию о сотрудниках.

*2. Диспетчер*

Человек, занимающийся учетом путевых листов. Может редактировать путевые листы.

*3. Зав. автопарком*

Человек, занимающийся учетом автобусов на предприятии. Может редактировать информацию об автобусах.

*4. Водитель*

Человек, занимающийся реализацией путевых листов. Может редактировать путевые листы.

**4. Спецификация прецедентов**

*Прецедент* «Авторизоваться»

**Действующее лицо:** Сотрудник

**Предусловие:**

* Актер не авторизован.

**Основной поток:**

* Актер вводит верную комбинацию логин-пароль.

**Альтернативный поток:**

* Введена не правильная пара логин-пароль. Актер заново вводит данные для авторизации.

**Постусловие:** -

­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­ *Прецедент*«Добавить сотрудника»

**Действующее лицо:** Администратор

**Предусловие:**

* Актер должен быть авторизирован в системе как администратор.

**Основной поток:**

* Актер вводит ФИО, адрес, телефон, роль сотрудника в соответствующие поля.
* Далее Актер подтверждает создание нового сотрудника нажатием на кнопку.

**Альтернативный поток:**

* Актер не заполнил все необходимые поля.

**Постусловие:** -

*Прецедент*«Редактировать данные сотрудника»

**Действующее лицо:** Администратор

**Предусловие:**

* Актер должен быть авторизирован в системе как администратор
* Редактируемый сотрудник присутствует в базе данных.

**Основной поток:**

* Актер вводит ФИО для поиска сотрудника в базе данных и открывает шаблон его анкеты, изменяет данные о сотруднике.
* Далее Актер подтверждает изменения нажатием на соответствующую кнопку.

**Альтернативный поток:**

* Актер не заполнил все необходимые поля.

**Постусловие:** -

­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­­

*Прецедент*«Удалить сотрудника»

**Действующее лицо:** Администратор

**Предусловие:**

* Актер должен быть авторизирован в системе как администратор.
* Редактируемый сотрудник присутствует в базе данных.

**Основной поток:**

* Актер вводит ФИО для поиска сотрудника в базе данных и удаляет данного сотрудника.

**Альтернативный поток:**

* Сотрудник не найден в базе данных. Выполняется повторный поиск или отмена удаления.

**Постусловие:** -

*Прецедент*«Добавить автобус»

**Действующее лицо:** Зав. автопарком

**Предусловие:**

* Актер должен быть авторизирован в системе как Зав. автопарком.

**Основной поток:**

* Актер вводит наименование автобуса, регистрационный знак, цвет, VIN-номер в шаблон анкеты.
* Актер подтверждает изменения нажатием на соответствующую кнопку.

**Альтернативный поток:**

* Актер не заполнил все необходимые поля.

**Постусловие:** -

*Прецедент*«Просмотреть информацию об автобусе»

**Действующее лицо:** Зав. автопарком, диспетчер

**Предусловие:**

* Актер должен быть авторизирован в системе как зав. автопарком или диспетчер.

**Основной поток:**

* Актер вводит VIN-номер автобуса для поиска информации об автобусе в базе данных.

**Альтернативный поток:**

* Автобус не найден. Актер изменяет критерии поиска и ищет снова, либо прекращает поиск.

**Постусловие:** -

*Прецедент*«Удалить автобус»

**Действующее лицо:** зав. автопарком

**Предусловие:** Актер должен быть авторизирован в системе как зав. автопарком

**Основной поток:**

* Актер вводит VIN-номер для удаления автобуса.
* Актер подтверждает изменения нажатием на соответствующую кнопку.

**Альтернативный поток:**

* Актер не заполнил все необходимые поля.
* Автобус не найден**.** Актер изменяет критерии поиска и ищет снова, либо прекращает поиск.

**Постусловие:** -

*Прецедент*«Редактировать данные об автобусе»

**Действующее лицо:** Зав. автопарком

**Предусловие:** Актер должен быть авторизирован в системе как зав. автопарком.

**Основной поток:**

* Актер вводит VIN-номер для изменения данных об автобусе.
* Актер подтверждает изменения нажатием на соответствующую кнопку.

**Альтернативный поток:**

* Автобус не найден**.** Актер изменяет критерии поиска и ищет снова, либо прекращает поиск.

**Постусловие:** -

*Прецедент*«Создание путевого листа»

**Действующее лицо:** диспетчер

**Предусловие:** Актер должен быть авторизирован в системе как диспетчер.

**Основной поток:**

* Актер вводит ФИО, наименование автобуса, регистрационный знак, начало маршрута, конец маршрута.

**Альтернативный поток:**

* Актер не заполнил все необходимые поля.

**Постусловие:** -

*Прецедент*«Просмотр путевого листа»

**Действующее лицо:** диспетчер, водитель.

**Предусловие:** Актер должен быть авторизирован в системе как диспетчерили водитель

**Основной поток:**

* Актер вводит номер путевого листа.

**Альтернативный поток:**

* Путевой лист не найден**.** Актер изменяет критерии поиска и ищет снова, либо прекращает поиск.

**Постусловие:** -

*Прецедент*«Редактирование путевого листа»

**Действующее лицо:** диспетчер

**Предусловие:** Актер должен быть авторизирован в системе как диспетчерили водитель.

**Основной поток:**

* Актер входит в анкету редактирования нажатием на соответствующую кнопку.
* Актер подтверждает изменения нажатием на соответствующую кнопку.

**Альтернативный поток:**

* Актер не заполнил все необходимые поля.

**Постусловие:** -

**5. Диаграмма прецедентов**

На рисунке 1 представлена диаграмма прецедентов.



Рисунок 1. Диаграмма прецедентов

Описание диаграммы классов:

*Класс «Employee»* - Пользователь системы, который после авторизации получает свою роль.

*Класс «Manager» - роль пользователя системы, позволяющая ему управлять информацией об автобусах (добавление, удаление, редактирование).*

*Класс «Administrator»* - роль пользователя системы, позволяющее ему добавлять, удалять и изменять роли других пользователей.

*Класс «Bus»* - сущность, предназначенная для хранения данные об автобусах.

*Класс «Waybill» -* сущность, предназначенная для хранения данные о путевых листах.

*Класс «Driver»* - роль пользователя системы, позволяющая ему просматривать и редактировать путевые листы.

*Класс «Dispatcher»* - роль пользователя системы, позволяющая ему просматривать и редактировать путевые листы.

На рисунке 2 представлена диаграмма классов.



Рисунок 2. Диаграмма классов.

На рисунке 3 представлена диаграмма состояний.



Рисунок 3. Диаграмма состояний.

На рисунке 4 представлена диаграмма последовательностей.



Рисунок 4. Диаграмма последовательностей.

**КОД ПРОГРАММЫ**

Web.config

<?xml version="1.0" encoding="utf-8"?>

<!--

Дополнительные сведения о настройке приложения ASP.NET см. на странице

https://go.microsoft.com/fwlink/?LinkId=301880

-->

<configuration>

<appSettings>

<add key="webpages:Version" value="3.0.0.0" />

<add key="webpages:Enabled" value="false" />

<add key="ClientValidationEnabled" value="true" />

<add key="UnobtrusiveJavaScriptEnabled" value="true" />

</appSettings>

<connectionStrings>

<add name="DbConnection" providerName="System.Data.SqlClient" connectionString="Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\idea2\Desktop\TPproject\TPproject\App\_Data\Database.mdf;Integrated Security=True" />

</connectionStrings>

<system.web>

<compilation debug="true" targetFramework="4.6.1" />

<httpRuntime targetFramework="4.6.1" />

<authentication mode="Forms">

<forms name="cookies" timeout="120" loginUrl="~/Account/Login" />

</authentication>

<httpModules>

<add name="ApplicationInsightsWebTracking" type="Microsoft.ApplicationInsights.Web.ApplicationInsightsHttpModule, Microsoft.AI.Web" />

</httpModules>

<roleManager enabled="true" defaultProvider="MyRoleProvider">

<providers>

<add name="MyRoleProvider" type="TPproject.Providers.CustomRoleProvider" />

</providers>

</roleManager>

</system.web>

<runtime>

<assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">

<dependentAssembly>

<assemblyIdentity name="Newtonsoft.Json" culture="neutral" publicKeyToken="30ad4fe6b2a6aeed" />

<bindingRedirect oldVersion="0.0.0.0-6.0.0.0" newVersion="6.0.0.0" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="System.Web.Optimization" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="1.0.0.0-1.1.0.0" newVersion="1.1.0.0" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="WebGrease" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="0.0.0.0-1.5.2.14234" newVersion="1.5.2.14234" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="System.Web.Helpers" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="System.Web.WebPages" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="System.Web.Mvc" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="1.0.0.0-5.2.3.0" newVersion="5.2.3.0" />

</dependentAssembly>

</assemblyBinding>

</runtime>

<system.webServer>

<validation validateIntegratedModeConfiguration="false" />

<modules>

<remove name="ApplicationInsightsWebTracking" />

<add name="ApplicationInsightsWebTracking" type="Microsoft.ApplicationInsights.Web.ApplicationInsightsHttpModule, Microsoft.AI.Web" preCondition="managedHandler" />

</modules>

</system.webServer>

<system.codedom>

<compilers>

<compiler language="c#;cs;csharp" extension=".cs" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=1.0.7.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default /nowarn:1659;1699;1701" />

<compiler language="vb;vbs;visualbasic;vbscript" extension=".vb" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=1.0.7.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default /nowarn:41008 /define:\_MYTYPE=\&quot;Web\&quot; /optionInfer+" />

</compilers>

</system.codedom>

</configuration>

Models.cs

namespace TPproject.Models

{

public class LoginModel

{

[Required]

public string Email { get; set; }

[Required]

[DataType(DataType.Password)]

public string Password { get; set; }

}

public class RegisterModel

{

[Required]

public string FirstName { get; set; }

[Required]

public string LastName { get; set; }

[Required]

public string Email { get; set; }

[Required]

[DataType(DataType.Password)]

public string Password { get; set; }

[Required]

[DataType(DataType.Password)]

[Compare("Password", ErrorMessage = "Пароли не совпадают")]

public string ConfirmPassword { get; set; }

[Required]

public string PhoneNumber { get; set; }

[Required]

public string RoleID { get; set; }

}

}

User.cs

namespace TPproject.Models

{

public class User

{

public int Id { get; set; }

[Display(Name ="Email")]

public string Email { get; set; }

[Display(Name = "Пароль")]

public string Password { get; set; }

[Display(Name = "Имя")]

public string FirstName { get; set; }

[Display(Name = "Фамилия")]

public string LastName { get; set; }

[Display(Name = "Номер телефона")]

public string PhoneNumber { get; set; }

public int RoleID { get; set; }

public Role Role { get; set; }

}

public class Role

{

public int Id { get; set; }

public string Name { get; set; }

}

}

UserContext.cs

namespace TPproject.Models

{

public class UserContext : DbContext

{

public UserContext() :

base("DbConnection")

{ }

public DbSet<User> User { get; set; }

public DbSet<Role> Role { get; set; }

public User GetUser(String name)

{

return User.FirstOrDefault(u => u.Email == name);

}

public List<User> GetAllUser(UserContext context)

{

List<User> UserList = new List<User>();

foreach (User user in context.User.ToList())

{

UserList.Add(user);

}

return UserList;

}

}

}

Waybill.cs

namespace TPproject.Models

{

public class Waybill

{

public int Id { get; set; }

public int UserId { get; set; }

public User User { get; set; }

public int BusId { get; set; }

public Bus Bus { get; set; }

public DateTime time\_start { get; set; }

public DateTime time\_end { get; set; }

public int WaybillStatusId { get; set; }

public WaybillStatus WaybillStatus { get; set; }

}

public class Bus

{

public int Id { get; set; }

public string name\_bus { get; set; }

public string number\_bus { get; set; }

public string color\_bus { get; set; }

public string vin\_bus { get; set; }

}

public class WaybillStatus

{

public int Id { get; set; }

public string Name { get; set; }

}

}

WaybillContext.cs

namespace TPproject.Models

{

public class WaybillContext : DbContext

{

public WaybillContext() :

base("DbConnection")

{ }

public DbSet<Waybill> Waybill { get; set; }

public DbSet<Bus> Bus { get; set; }

public DbSet<WaybillStatus> WaybillStatus { get; set; }

}

}

Waybill.cs

namespace BusCompany.Models

{

public class Waybill

{

public int id { get; set; } //идентификатор

public DateTime date { get; set; } //дата оформления

public int? busID { get; set; }

public Bus bus { get; set; }

public string client { get; set; }

public string logist { get; set; }

public string path { get; set; } //маршрут

public DateTime departureDate { get; set; } //дата выезда

public DateTime returnDate { get; set; } //дата возвращения

}

}

HomeController.cs

namespace TPproject.Controllers

{

public class HomeController : Controller

{

WaybillContext orderContext = new WaybillContext();

public ActionResult Index()

{

return View();

}

[Authorize]

public ActionResult Contact()

{

ViewBag.Message = "Your contact page.";

return View();

}

}

}

AccountController.cs

namespace TPproject.Controllers

{

public class AccountController : Controller

{

public ActionResult Login()

{

return View();

}

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Login(LoginModel model)

{

if (ModelState.IsValid)

{

// поиск пользователя в бд

User user = null;

using (UserContext db = new UserContext())

{

user = db.User.FirstOrDefault(u => u.Email == model.Email && u.Password == model.Password);

}

if (user != null)

{

FormsAuthentication.SetAuthCookie(model.Email, true);

return RedirectToAction("Index", "Manage");

}

else

{

ModelState.AddModelError("", "Пользователя с таким логином и паролем нет");

}

}

return View(model);

}

public ActionResult Register()

{

return View();

}

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Register(RegisterModel model)

{

if (ModelState.IsValid)

{

User user = null;

using (UserContext db = new UserContext())

{

user = db.User.FirstOrDefault(u => u.Email == model.Email);

}

if (user == null)

{

// создаем нового пользователя

using (UserContext db = new UserContext())

{

db.User.Add(new User { Email = model.Email, Password = model.Password, FirstName = model.FirstName,

LastName = model.LastName, PhoneNumber = model.PhoneNumber, RoleID = 2});

db.SaveChanges();

user = db.User.Where(u => u.Email == model.Email && u.Password == model.Password).FirstOrDefault();

}

// если пользователь удачно добавлен в бд

if (user != null)

{

FormsAuthentication.SetAuthCookie(model.Email, true);

return RedirectToAction("Index", "Home");

}

}

else

{

ModelState.AddModelError("", "Пользователь с таким адресом электронной почты уже существует");

}

}

return View(model);

}

public ActionResult Logoff()

{

FormsAuthentication.SignOut();

return RedirectToAction("Index", "Home");

}

}

}

ManageController.cs

namespace TPproject.Controllers

{

public class ManageController : Controller

{

UserContext userContext = new UserContext();

WaybillContext waybillContext = new WaybillContext();

[Authorize]

public ActionResult Index()

{

return View(userContext.GetUser(User.Identity.Name));

}

[Authorize]

public ActionResult EditUser(int id)

{

User user = userContext.User.Find(id);

if (user != null)

{

return PartialView("EditUser", user);

}

return View("Index");

}

[HttpPost]

public ActionResult EditUser(User user)

{

userContext.Entry(user).State = EntityState.Modified;

userContext.SaveChanges();

return RedirectToAction("AdminView");

}

[Authorize]

public ActionResult DeleteUser(int id)

{

User user = userContext.User.Find(id);

if (user != null)

{

return PartialView("DeleteUser", user);

}

return View("AdminView");

}

[HttpPost]

public ActionResult DeleteUser(int id, FormCollection collection)

{

User user = userContext.User.Find(id);

if (user != null)

{

userContext.User.Remove(user);

userContext.SaveChanges();

}

return RedirectToAction("AdminView");

}

[Authorize]

public ActionResult AdminView()

{

if (userContext.User.Where(u => u.Email == User.Identity.Name).Select(u => u.Role.Name).FirstOrDefault() == "admin")

return View(userContext.GetAllUser(userContext));

else return RedirectToAction("Index");

}

[Authorize(Roles = "manager")]

public ActionResult ManagerAddBus()

{

return View();

}

[HttpPost]

public ActionResult ManagerAddBus(Bus bus)

{

waybillContext.Bus.Add(bus);

waybillContext.SaveChanges();

return RedirectToAction("Index");

}

public ActionResult WaybillList()

{

var waybils = waybillContext.Waybill.Include(p => p.Bus).Include(x => x.User).Include(y => y.WaybillStatus);

return View(waybils.ToList());

}

public ActionResult CreateWaybill()

{

return View();

}

[HttpPost]

public ActionResult CreateWaybill(Waybill waybill)

{

waybill.time\_start = DateTime.Now;

waybill.time\_end = DateTime.MaxValue;

waybill.WaybillStatusId = 1;

waybillContext.Waybill.Add(waybill);

waybillContext.SaveChanges();

return RedirectToAction("Index");

}

[Authorize]

public ActionResult BusDelete(int id)

{

Bus bus = waybillContext.Bus.Find(id);

if (bus != null)

{

return PartialView("BusDelete", bus);

}

return View("Services");

}

[HttpPost]

public ActionResult BusDelete(int id, FormCollection collection)

{

Bus bus = waybillContext.Bus.Find(id);

if (bus != null)

{

waybillContext.Bus.Remove(bus);

waybillContext.SaveChanges();

}

return RedirectToAction("Index");

}

[Authorize]

public ActionResult BusEdit(int id)

{

Bus bus = waybillContext.Bus.Find(id);

if (bus != null)

{

return PartialView("BusEdit", bus);

}

return View("Index");

}

[HttpPost]

public ActionResult BusEdit(Bus bus)

{

waybillContext.Entry(bus).State = EntityState.Modified;

waybillContext.SaveChanges();

return RedirectToAction("Index");

}

[Authorize]

public ActionResult WaybillEdit(int id)

{

Waybill waybill = waybillContext.Waybill.Find(id);

if (waybill != null)

{

return PartialView("WaybillEdit", waybill);

}

return View("WaybillList");

}

[HttpPost]

public ActionResult WaybillEdit(Waybill waybill)

{

waybillContext.Entry(waybill).State = EntityState.Modified;

waybillContext.SaveChanges();

return RedirectToAction("WaybillList");

}

[Authorize]

public ActionResult WaybillDetails(int id)

{

Waybill waybill = waybillContext.Waybill.Find(id);

if (waybill != null)

{

return View(waybill);

}

return View("Index");

}

[HttpPost]

public ActionResult WaybillDetails(int id,string command)

{

Waybill waybill = waybillContext.Waybill.Find(id);

if (waybill != null)

{

if (command.Equals("Выполнен"))

{

waybill.WaybillStatusId = 2;

waybillContext.Entry(waybill).State = EntityState.Modified;

waybillContext.SaveChanges();

return RedirectToAction("WaybillList");

}

if (command.Equals("Отклонить"))

{

waybill.WaybillStatusId = 3;

waybillContext.Entry(waybill).State = EntityState.Modified;

waybillContext.SaveChanges();

return RedirectToAction("WaybillList");

}

}

return RedirectToAction("Index");

}

[Authorize]

public ActionResult BusList()

{

return View(waybillContext.Bus);

}

[Authorize]

public ActionResult DeleteWaybill(int id)

{

Waybill waybill = waybillContext.Waybill.Find(id);

if (waybill != null)

{

return PartialView("DeleteWaybill", waybill);

}

return View("WaybillList");

}

[HttpPost]

public ActionResult DeleteWaybill(int id, FormCollection collection)

{

Waybill waybill = waybillContext.Waybill.Find(id);

if (waybill != null)

{

waybillContext.Waybill.Remove(waybill);

waybillContext.SaveChanges();

}

return RedirectToAction("WaybillList");

}

}

}

AdminView.cshtml

@model IEnumerable<TPproject.Models.User>

@{

ViewBag.Title = "AdminView";

}

<h2>Список сотрудников</h2>

@{

}

@foreach (var item in Model)

{

<div style="border-collapse:separate; border-style:outset; border-spacing:10px;">

<p>RoleID: @Html.DisplayFor(modelItem => item.RoleID)</p>

<p>Email: @Html.DisplayFor(modelItem => item.Email)</p>

<p>Пароль: @Html.DisplayFor(modelItem => item.Password)</p>

<p>Имя: @Html.DisplayFor(modelItem => item.FirstName)</p>

<p>Фамилия: @Html.DisplayFor(modelItem => item.LastName)</p>

<p>Номер телефона: @Html.DisplayFor(modelItem => item.PhoneNumber)</p>

<p>

@Html.ActionLink("Удалить", "DeleteUser", new { id = item.Id }, new { @class = "viewDialog", data\_dialog\_title = "Удаление пользователя" })

@Html.ActionLink("Изменить", "EditUser", new { id = item.Id }, new { @class = "viewDialog", data\_dialog\_title = "Изменение данных пользователя" })

</p>

</div>

}

@section scripts {

<link href='@Url.Content("~/Content/themes/base/jquery-ui.css")' rel="stylesheet" type="text/css" />

<script src='@Url.Content("~/Scripts/jquery-1.12.4.min.js")' type="text/javascript"></script>

<script src='@Url.Content("~/Scripts/jquery-ui-1.12.1.min.js")' type="text/javascript"></script>

<script>

$(document).ready(function () {

$.ajaxSetup({ cache: false });

$(".viewDialog").on("click", function (e) {

e.preventDefault();

$("<div></div>")

.addClass("dialog")

.appendTo("body")

.dialog({

title: $(this).attr("data-dialog-title"),

close: function () { $(this).remove() },

modal: true

})

.load(this.href);

});

});

</script>

}

BusDelete.cshtml

@model TPproject.Models.Bus

@{

ViewBag.Title = "BusEdit";

}

<div>

<h3>Вы уверены, что хотите удалить эти данные?</h3>

<div>

<p> @Html.DisplayName("Название: ")

@Html.DisplayFor(model => model.name\_bus) </p>

<p> @Html.DisplayName("Номер: ")

@Html.DisplayFor(model => model.number\_bus)</p>

<p>@Html.DisplayName("Цвет: ")

@Html.DisplayFor(model => model.color\_bus)</p>

<p>@Html.DisplayName("VIN-номер: ")

@Html.DisplayFor(model => model.vin\_bus)</p>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Удалить" class="btn btn-default" />

</div>

}

</div>

</div>

BusEdit.cshtml

@model TPproject.Models.Bus

@{

ViewBag.Title = "BusEdit";

}

<div>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@Html.HiddenFor(model => model.Id)

<div class="form-group">

@Html.LabelFor(model => model.name\_bus, "Наименование", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.name\_bus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.name\_bus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.number\_bus, "Номер", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.number\_bus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.number\_bus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.color\_bus, "Цвет", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.color\_bus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.color\_bus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.vin\_bus, "VIN-номер", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.vin\_bus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.vin\_bus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Сохранить" class="btn btn-default" />

</div>

</div>

</div>

}

</div>

BusList.cshtml

@model IEnumerable<TPproject.Models.Bus>

@{

ViewBag.Title = "BusList";

}

<h2>Автобусы</h2>

<table class="table">

<tr>

<th>

@Html.DisplayName("Название автобуса")

</th>

<th>

@Html.DisplayName("Номер")

</th>

<th>

@Html.DisplayName("Цвет")

</th>

<th>

@Html.DisplayName("VIN-номер")

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.name\_bus)

</td>

<td>

@Html.DisplayFor(modelItem => item.number\_bus)

</td>

<td>

@Html.DisplayFor(modelItem => item.color\_bus)

</td>

<td>

@Html.DisplayFor(modelItem => item.vin\_bus)

</td>

<td>

@if (User.IsInRole("manager"))

{

@Html.ActionLink("Изменить ", "BusEdit", new { id = item.Id }, new { @class = "viewDialog", data\_dialog\_title = "Изменение автобуса" })

@Html.ActionLink("Удалить", "BusDelete", new { id = item.Id }, new { @class = "viewDialog", data\_dialog\_title = "Удаление автобуса" })

}

</td>

</tr>

}

</table>

@section scripts {

<link href='@Url.Content("~/Content/themes/base/jquery-ui.css")' rel="stylesheet" type="text/css" />

<script src='@Url.Content("~/Scripts/jquery-1.12.4.min.js")' type="text/javascript"></script>

<script src='@Url.Content("~/Scripts/jquery-ui-1.12.1.min.js")' type="text/javascript"></script>

<script>

$(document).ready(function () {

$.ajaxSetup({ cache: false });

$(".viewDialog").on("click", function (e) {

e.preventDefault();

$("<div></div>")

.addClass("dialog")

.appendTo("body")

.dialog({

title: $(this).attr("data-dialog-title"),

close: function () { $(this).remove() },

modal: true

})

.load(this.href);

});

});

</script>

}

CreateWaybill.cshtml

@model TPproject.Models.Waybill

@{

ViewBag.Title = "CreateWaybill";

}

<h2>Оформление путевого листа</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.UserId, "ID Водителя", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.UserId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.UserId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.BusId, "ID Автобуса", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.BusId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.BusId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Создать" class="btn btn-default" />

</div>

</div>

</div>

}

DeleteUser.cshtml

@model TPproject.Models.User

@{

ViewBag.Title = "DeleteUser";

}

<h2>Вы действительно хотите удалить сотрудника?</h2>

<div>

<p>Имя: @Html.DisplayFor(model => model.FirstName)</p>

<p>Фамилия: @Html.DisplayFor(model => model.LastName)</p>

<p>Email: @Html.DisplayFor(model => model.Email)</p>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<input type="submit" value="Удалить" />

}

</div>

DeleteWaybill.cshtml

@model TPproject.Models.Waybill

@{

ViewBag.Title = "DeleteWaybill";

}

<div>

<h3>Вы уверены, что хотите удалить этот путевой лист?</h3>

<p>Номер: @Html.DisplayFor(model => model.Id)</p>

<p>Автобус: @Html.DisplayFor(model => model.BusId)</p>

<p>Водитель: @Html.DisplayFor(model => model.UserId)</p>

<p>Дата начала: @Html.DisplayFor(model => model.time\_start)</p>

<p>Дата конца: @Html.DisplayFor(model => model.time\_end)</p>

<p>Статус: @Html.DisplayFor(model => model.WaybillStatusId)</p>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Удалить" class="btn btn-default" />

</div>

}

</div>

EditUser.cshtml

@model TPproject.Models.User

@{

ViewBag.Title = "EditUser";

}

<div style="width:100%">

@{

List<SelectListItem> Role = new List<SelectListItem>();

Role.Add(new SelectListItem { Text = "Admin",Value = "1" });

Role.Add(new SelectListItem { Text = "Dispatcher", Value = "2" });

Role.Add(new SelectListItem { Text = "Manager",Value = "3" });

Role.Add(new SelectListItem { Text = "Driver", Value = "4" });

}

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.Email, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Email, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Email, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Password, "Пароль", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Password, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Password, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.FirstName, "Имя", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.FirstName, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.FirstName, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.LastName, "Фамилия", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.LastName, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.LastName, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.PhoneNumber, "Номер телефона", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.PhoneNumber, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.PhoneNumber, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.RoleID, "Должность", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.DropDownList("RoleID",Role, htmlAttributes: new { @class = "form-control" })

@Html.ValidationMessageFor(model => model.RoleID, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Сохранить" class="btn btn-default" />

</div>

</div>

</div>

}

</div>

Index.cshtml

@model TPproject.Models.User

@{

ViewBag.Title = "Index";

}

<h2>Личный кабинет</h2>

<div>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Email)

</dt>

<dd>

@Html.DisplayFor(model => model.Email)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Password)

</dt>

<dd>

@Html.DisplayFor(model => model.Password)

</dd>

<dt>

@Html.DisplayNameFor(model => model.FirstName)

</dt>

<dd>

@Html.DisplayFor(model => model.FirstName)

</dd>

<dt>

@Html.DisplayNameFor(model => model.LastName)

</dt>

<dd>

@Html.DisplayFor(model => model.LastName)

</dd>

<dt>

@Html.DisplayNameFor(model => model.PhoneNumber)

</dt>

<dd>

@Html.DisplayFor(model => model.PhoneNumber)

</dd>

</dl>

<div>

@if (User.IsInRole("admin"))

{

<li>@Html.ActionLink("Список сотрудников", "AdminView")</li>

<li>@Html.ActionLink("Добавление сотрудника", "Register", "Account")</li>

}

<ul>

@if (User.IsInRole("manager"))

{

<li>@Html.ActionLink("Добавление автобуса", "ManagerAddBus")</li>

<li>@Html.ActionLink("Список автобусов", "BusList", "Manage")</li>

}

@if (User.IsInRole("dispatcher"))

{

<li>@Html.ActionLink("Создание путевого листа", "CreateWaybill", "Manage")</li>

<li>@Html.ActionLink("Список путевых листов", "WaybillList", "Manage")</li>

<li>@Html.ActionLink("Список автобусов", "BusList", "Manage")</li>

}

@if (User.IsInRole("driver"))

{

<li>@Html.ActionLink("Список путевых листов", "WaybillList", "Manage")</li>

}

</ul>

</div>

</div>

ManagerAddBus.cshtml

@model TPproject.Models.Bus

@{

ViewBag.Title = "ManagerAddBus";

}

<h2>Добавление автобуса</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.name\_bus, "Наименование", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.name\_bus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.name\_bus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.number\_bus, "Номер", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.number\_bus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.number\_bus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.color\_bus, "Цвет", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.color\_bus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.color\_bus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.vin\_bus, "VIN-номер", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.vin\_bus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.vin\_bus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Добавить" class="btn btn-default" />

</div>

</div>

</div>

}

WaybillDetails.cshtml

@model TPproject.Models.Waybill

@{

ViewBag.Title = "WaybillDetails";

}

<h2>Изменить статус путевого листа</h2>

<div>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayName("Статус")

</dt>

<dd>

@Html.DisplayFor(model => model.WaybillStatusId)

</dd>

<dt>

@Html.DisplayName("ID Автобуса")

</dt>

<dd>

@Html.DisplayFor(model => model.BusId)

</dd>

<dt>

@Html.DisplayName("ID Водителя")

</dt>

<dd>

@Html.DisplayFor(model => model.UserId)

</dd>

<dt>

@Html.DisplayName("Время начала")

</dt>

<dd>

@Html.DisplayFor(model => model.time\_start)

</dd>

<dt>

@Html.DisplayName("Время конца")

</dt>

<dd>

@Html.DisplayFor(model => model.time\_end)

</dd>

</dl>

</div>

<div>

@using (Html.BeginForm())

{

<input type="submit" name="command" value="Выполнен" />

<input type="submit" name="command" value="Отклонить" />

<b> @Html.ActionLink("Назад", "WaybillList") </b>

}

</div>

WaybillEdit.cshtml

@model TPproject.Models.Waybill

@{

ViewBag.Title = "WaybillEdit";

}

<div>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@Html.HiddenFor(model => model.Id)

<div class="form-group">

@Html.LabelFor(model => model.UserId, "ID Водителя", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.UserId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.UserId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.BusId, "ID Автобуса", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.BusId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.BusId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.time\_start, "Время начала", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.time\_start, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.time\_start, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.time\_end, "Время конца", htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.time\_end, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.time\_end, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.HiddenFor(model => model.WaybillStatusId)

<div class="col-md-10">

@Html.HiddenFor(model => model.WaybillStatusId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.WaybillStatusId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Сохранить" class="btn btn-default" />

</div>

</div>

</div>

}

</div>

WaybillList.cshtml

@model IEnumerable<TPproject.Models.Waybill>

@{

ViewBag.Title = "WaybillList";

}

<h2>Путевые листы</h2>

<p>

@if (User.IsInRole("dispatcher"))

{

@Html.ActionLink("добавить новый путевой лист", "CreateWaybill", "Manage")

}

</p>

<table class="table">

<tr>

<th>

@Html.DisplayName("Номер путевого листа")

</th>

<th>

@Html.DisplayName("Номер автобуса")

</th>

<th>

@Html.DisplayName("Водитель")

</th>

<th>

@Html.DisplayName("Время начала")

</th>

<th>

@Html.DisplayName("Время конца")

</th>

<th>

@Html.DisplayName("Статус")

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.Id)

</td>

<td>

@Html.DisplayFor(modelItem => item.Bus.number\_bus)

</td>

<td>

@Html.DisplayFor(modelItem => item.User.LastName)

</td>

<td>

@Html.DisplayFor(modelItem => item.time\_start)

</td>

<td>

@Html.DisplayFor(modelItem => item.time\_end)

</td>

<td>

@Html.DisplayFor(modelItem => item.WaybillStatus.Name)

</td>

<td>

@if (User.IsInRole("dispatcher"))

{

@Html.ActionLink("Удалить ", "DeleteWaybill", new { id = item.Id }, new { @class = "viewDialog", data\_dialog\_title = "Удалить?" })

@Html.ActionLink("Изменить ", "WaybillEdit", new { id = item.Id }, new { @class = "viewDialog", data\_dialog\_title = "Изменение путевого листа" })

}

@if (User.IsInRole("driver"))

{

@Html.ActionLink("Изменить статус", "WaybillDetails", new { id = item.Id })

}

</td>

</tr>

}

</table>

@section scripts {

<link href='@Url.Content("~/Content/themes/base/jquery-ui.css")' rel="stylesheet" type="text/css" />

<script src='@Url.Content("~/Scripts/jquery-1.12.4.min.js")' type="text/javascript"></script>

<script src='@Url.Content("~/Scripts/jquery-ui-1.12.1.min.js")' type="text/javascript"></script>

<script>

$(document).ready(function () {

$.ajaxSetup({ cache: false });

$(".viewDialog").on("click", function (e) {

e.preventDefault();

$("<div></div>")

.addClass("dialog")

.appendTo("body")

.dialog({

title: $(this).attr("data-dialog-title"),

close: function () { $(this).remove() },

modal: true

})

.load(this.href);

});

});

</script>

}