ВВЕДЕНИЕ

В 1994 году началась революция - World Wide Web. Всемирная паутина World Wide Web (WWW) соткана из Web-страниц, которые содержат в себе разную информацию в зависимости от тематики Web сайта. Полезность Internet повышалась вместе с развитием вычислительной техники с запаздыванием примерно в 10 лет. В конце 80-х годов появление персональных компьютеров перенесло информатику из царства знатоков к широкой публике. Internet в ходе своего развития и повсеместного распространения занимается именно таким переносом.

Ученые и преподаватели использовали Web уже несколько лет, но общество в целом еще только начинало осознавать его привлекательность. По всему миру пользователи компьютеров были увлечены мыслью о доступе к гигантской общемировой компьютерной сети, но полученная информация часто разочаровала - Интернет был исключительно текстовой средой.

Позже когда Web находился на заре своего развития, дизайну и разметке страниц уделялось совсем немного внимания.

Дизайнеры поняли, что точная технология размещения графики, позволяющая точно повторить красоту печатной страницы стала популярной, хотя и ограниченной с точки зрения скорости и дизайна Web-страницы.

Среди большого множества средств и программных продуктов было очень сложно ориентироваться. Каждая компания разработчик пыталась ввести нечто новое в свои средства. Это очень сильно отражалось на пользователе, который порой даже не мог отобразить Web документ на своём компьютере. Постоянно существовала необходимость устанавливать всё новое программное обеспечение для просмотра и работы с Web документами, которые были созданы при помощи нового языка программирования для Web. Также они не могли справиться с поставленными задачами, и приходилось смешивать некоторые средства разработки для Web.

World Wide Web (или просто Web) облегчает использование Internet, всемирной компьютерной сети, которая была создана в конце 60-х годов. Первоначально работа Internet (в том числе и получение данных от компьютеров, подключенных к этой сети) требовала от пользователя знания огромного числа сложных команд, похожих на заклинания.

В 1992 году Тим Бернерс-Ли и другие исследователи создали Web, который позволил «просматривать» ресурсы Internet, не прибегая к помощи сложных команд. За несколько лет существования Web браузеры - программы, предназначенные для просмотра Internet, - сделали Web еще простым и одновременно мощным.

Слово Web используется в двух разных смыслах. Обычно под словом «Web» подразумевают World Wide Web - всемирную «паутину», однако в применении к FrontPage под Web подразумевается набор страниц, созданных в этом приложении для Web - сайта.

В данной работе я буду представлять web-приложение для автоматизации учёта заказов на выполнение строительных работ. Хорошо известно, что в наше время доставка стройматериалов должна быть быстрой и качественной, поэтому тем, кто будет обращаться к моему сайту, предлагается доставка стройматериалов за один день.

Основной задачей данного приложения является автоматизация учёта заказов на выполнение строительных работ и доставки стройматериалов в Московской области.

Такое приложение позволит быстро и оперативно отвечать на полученные запросы, что значительно сократит время на регистрацию прочих первичных документов и выходной отчетности.

Следует отметить, что:

* при проектировании использовалась точка зрения самого разработчика;
* пользователи сайта имеют различные роли, права которых разграничены
* среда разработки – MS SQL Server 2014 и Visual Studio 2015.

1. АНАЛИТИЧЕСКАЯ ЧАСТЬ
2. Обоснование актуальности задачи
   * 1. Актуальность темы

Представительство в Интернете обладает уникальными, невиданными до настоящего времени, свойствами. Это сверх эффективный инструмент в конкурентной борьбе. Расходы по нему сводятся, по сути, к расходам по созданию сайта и, по Вашему желанию, к небольшим отчислениям на поддержание его в лучшем виде с нашей стороны. В данном случае автоматизация работы системы учёта заказов рассматривается на примере нескольких лиц, которые может выполнить запрос на выполнение строительных работ клиента.

* + 1. Анализ существующих разработок по данной теме

В настоящее время существует очень большое количество Web-сайтов, осуществляющих решение задачи автоматизации учёта запроса на выполнение строительных работ.

Например, сайт "topas24.ru" - в данном сайте реализованы следующие возможности:

- просмотр информации о ценах модели, монтажа и доставки;

- возможность заказа звонка на сайте;

Структура решения представляет набор взаимосвязанных подсистем, функциональные возможности которых обеспечивают работу различных подразделений системы учёта заказов.

Рассмотрим некоторые из них:

- редактирование данных на сайте;

- возможность оставки запроса на сайте или обращения по телефону;

Важным является фактор долгого редактирования информации на сайте «topas24.ru», также на производительность сильно повлияет более мощный хостинг сайте, т.е. сайт будет дороже стоить по хостингу.

Таким образом, представляется нецелесообразным использование сайта «topas24.ru» для решения узкоспециализированной задачи создания автоматизированной информационной системы учёта заказов. Гораздо дешевле и быстрее представляется написание отдельного сайта, решающего данную задачу, являющегося простым в установке и использовании, дешевым и удобным в использовании редактировании данных, а также легко приспособляемым к новым требованиям компании, причём данный сайт можно использовать не только на компьютерах, но и запускать на смартфонах и планшетах.

1. Обоснование выбора и описание методологии разрабатываемого программного обеспечения

Важнейшим шагом на пути к совершенствованию языков программирования стало появление объектно-ориентированного подхода к программированию (или, сокращенно, ООП) и соответствующего класса языков. Наиболее известным примером объектно-ориентированного языка программирования является язык C++, развившийся из императивного языка С. Его прямым потомком и логическим продолжением является язык С# на основе которого и будет писаться клиентское приложение данной курсовой работы. При объектно-ориентированном подходе программа представляет собой описание объектов, их свойств (или атрибутов), совокупностей (или классов), отношений между ними, способов их взаимодействия и операций над объектами (или методов). Несомненным преимуществом данного подхода является концептуальная близость к предметной области произвольной структуры и назначения. Механизм наследования атрибутов и методов позволяет строить производные понятия на основе базовых и таким образом создать модель сколь угодно сложной предметной области с заданными свойствами. В связи с тем, что объектно-ориентированный подход используется в случае, когда важна управляемость проекта и его модифицируемость, а также скорость разработки именно этот подход я буду использовать в своей работе.

1. Обоснование выбора и описание инструментальных средств разработки программного обеспечения
   * 1. Обоснование выбора средств разработки клиентской части ЭИС

В качестве средства разработки клиентской части ЭИС по автоматизации учета сервисного центра была использована среда программирования Microsoft Visual Studio 2015. Среда разработана в соответствии с концепцией визуального программирования.

Microsoft Visual Studio 2015 — это набор инструментов для создания программного обеспечения: от планирования до разработки пользовательского интерфейса, написания кода, тестирования, отладки, анализа качества кода и производительности, развертывания в средах клиентов и сбора данных телеметрии по использованию. Эти инструменты предназначены для максимально эффективной совместной работы; все они доступны в интегрированной среде разработки (IDE) Visual Studio.

Visual Studio можно использовать для создания различных типов приложений, от простых приложений для магазина и игр для мобильных клиентов до больших и сложных систем, обслуживающих предприятия и центры обработки данных. Вы можете создавать:

- приложения и игры, которые выполняются не только на платформе Windows, но и на Android и iOS;

- веб-сайты и веб-службы на основе ASP.NET, JQuery, AngularJS и других популярных платформ;

- приложения для самых разных платформ и устройств, включая, но не ограничиваясь: Office, Sharepoint, Hololens, Kinect;

По умолчанию Visual Studio обеспечивает поддержку C#, C и C++, JavaScript, F# и Visual Basic.

* + 1. Обоснвание выбора СУБД

База Данных (БД) – это организованная совокупность структурированных данных, относящихся к определенной предметной области.

Современные БД создаются в памяти компьютера с помощью специальных программ – СУБД (системы управления базами данных).

СУБД – это совокупность языковых и программных средств, предназначенных для создания, ведения и совместного использования БД многими пользователями.

В зависимости от расположения отдельных частей БД различают локальные и сетевые БД. Все части локальной БД размещаются на компьютере пользователя. К сетевым БД относятся файл-серверные, клиент-серверные и распределенные БД. Непременным атрибутом этих систем является сеть.

Современные СУБД можно разделить на 3 категории:

- СУБД, предназначенные для работы с информационными массивами в небольших компаниях (MS Access, Visual FoxPro, Borland Interbase, Paradox и др.);

- программные продукты корпоративного направления (Oracle, MS SQL Server, IBM DB2, PostgreSQL и др.);

- СУБД для Web-приложений, реализующих создание веб-сайтов с небольшими БД (MySQL, Interbase и др.).

Пользователи общаются с БД с помощью специальных прикладных программ-приложений. Приложения могут создаваться как средствами СУБД (MS Access), так и средствами языков программирования: Delphi, Visual Basic, Java, C#, Visual Studio и др.

Рассмотрим СУБД MS SQL Server 2008, предоставляющую набор функций и средств, которые можно использовать для разработки баз данных и решений, а так же для управления ими.

Платформа данных MS SQL Server благодаря исчерпывающему набору функций, взаимодействию с существующими системами и автоматизации типовых задач, предоставляет полное решение в области хранения данных для предприятий всех масштабов. MS SQL Server – лучшая система для ОС Windows по причине тесной интеграции.

Несмотря на наличие стандартов, практически в каждой СУБД применяется свой диалект языка. Для MS SQL Server 2008 таким языком является Transact SQL. В нем можно выделить следующие подразделы:

* команды языка определения данных (Data Definition Language);
* команды языка манипулирования данными (Data Manipulation Language);
* язык запросов (DQL – Data Query Language);
* средства управления транзакциями;
* средства администрирования БД (управление БД и доступом).
  1. Содержательная постановка задачи
     1. Описательная модель задачи

Областью применения данной задачи является система учёта заказа по выполнению строительных работ. При вызове этой системы к пользователю вызывается его телефон и сам заказ. Длительность обработки вызова зависит от проживания пользователя и занятости администратора.

При вызове служб очевидец должен написать своё имя, номер телефона, E-Mail и заказ, чтобы упростить администратору идентификацию заказа пользователя.

* + 1. Математическая модель

В данном приложении не используются математические модели.

* + 1. Описание входной информации
       1. Описание входных данных

Входные данные указаны в таблице 1

Таблица 1 – Входные данные

|  |  |
| --- | --- |
| Входные данные | Описание действий |
| Имя | Вводится имя пользователя |
| Телефон | Вводится телефон пользователя для связи с ним |
| E-Mail | Вводится электронная почта пользователя |
| Запрос | Вводится запрос пользователя |

* + 1. Описание выходной информации
       1. Описание выходных документов

Выходные документы не используются в данном приложении.

* + - 1. Описание выходных данных

Выходные данные указаны в таблице 2

Таблица 2 – Выходные данные

|  |  |
| --- | --- |
| Выходные данные | Описание действий |
| Обработка запроса | Администратор обрабатывает запрос пользователя и связывается с ним |

Логическая модель задачи

Логическая модель задачи представлена в приложении 1.

Детализированная логическая модель задачи представлена в приложении 2.

1. ПРОЕКТНАЯ ЧАСТЬ

2.1. Разработка структуры данных ПО

2.1.1. Схема данных

Реляционная схема данных представлена на рисунке 1.

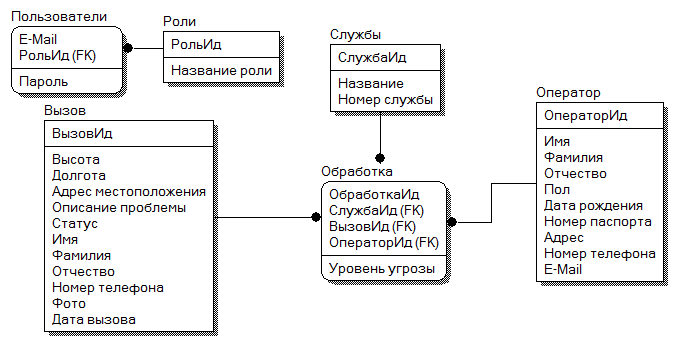


Рис. 1

2.1.2. Диаграмма потоков данных

Диаграмма потоков данных представлена в приложении 3.

Уточненная диаграмма потоков данных представлена в приложении 4.

2.2. Логическая модель ПО

Логическая модель ПО представлена в виде диаграммы UML.

Диаграмма UML представлена на рисунке 2.



Рис. 2

Таблица 3 – Описание сценария «Просмотр информации»

|  |  |
| --- | --- |
| Вариант использования | Вызов |
| Актёры | Пользователь |
| Краткое описание | Пользователь просматривает доступную на сайте информацию |
| Цель | Просмотр информации |
| Тип | Базовый |
| Ссылки на другие варианты использования | - |

Таблица 4 – Описание сценария «Оформление запроса»

|  |  |
| --- | --- |
| Вариант использования | Обработка вызова |
| Актёры | Пользователь |
| Краткое описание | Пользователь оформляет запрос на сайте |
| Цель | Оформление запроса |
| Тип | Базовый |
| Ссылки на другие варианты использования | Включает в себя вырианты использования:  обработка запроса. |

Таблица 5 – Описание сценария «Редактирование данных»

|  |  |
| --- | --- |
| Вариант использования | Определение статуса |
| Актёры | Администратор |
| Краткое описание | Администратор редактирует данные, отображаемые на сайте |
| Цель | Редактирование данных |
| Тип | Базовый |
| Ссылки на другие варианты использования | - |

Таблица 6 – Описание сценария «Обработка вызова»

|  |  |
| --- | --- |
| Вариант использования | Формирование отчёта |
| Актёры | Администратор |
| Краткое описание | Администратор обрабатывает вызов, оставленный на сайте |
| Цель | Обработка вызова |
| Тип | Базовый |
| Ссылки на другие варианты использования | - |

Таблица 7 – Сценарий успешного выполнения варианта использования

|  |  |
| --- | --- |
| Действия актёров | Отклик системы |
| 1. Администратор редактирует данные для просмотра на сайте 2. Пользователь просматривает информацию на сайте 3. Пользователь оформляет запрос на сайте | 1. Администратор обрабатывает запросы, которые были оставлены на сайте |

2.3. Разработка пользовательского интерфейса ПО

2.3.1. Разработка требований к интерфейсу

Основные требования к пользовательскому интерфейсу:

- функциональность (соответствие задачам пользователя);

- соответствие технологии;

- понятность и логичность;

- обеспечение высокой скорости работы пользователя;

- обеспечение защиты от человеческих ошибок;

- быстрое обучение пользователя;

- субъективное удовлетворение пользователя.

Требования к расположению элементов управления:

- элементы меню следует группировать;

- группы следует разделять либо разделительными полосками либо «визуальными паузами»;

- часто используемые элементы целесообразно располагать в левой верхней части экрана, редко используемые - в правой нижней части;

- командные кнопки, управляющие окном, например, «Ок», «Добавить», «Отмена» и др. должны быть расположены либо внизу окна, либо в правой его части (т.е. в той части окна, которая сканируется взглядом в последнюю очередь).

* + 1. Проектирование и описание элементов управления

Интерфейс сайта представлен на рисунках 3-26.

Главная страница

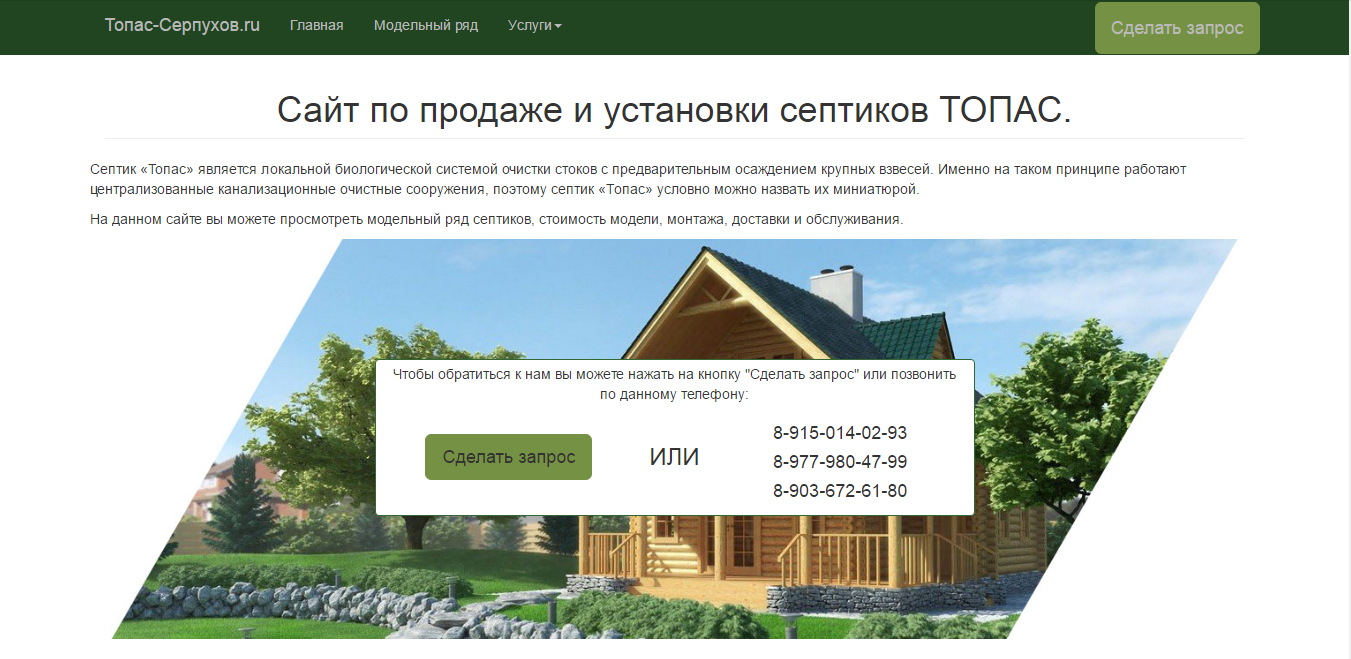


Рис. 3

Страница «Модельный ряд»



Рис. 4

Форма «Сделать запрос»

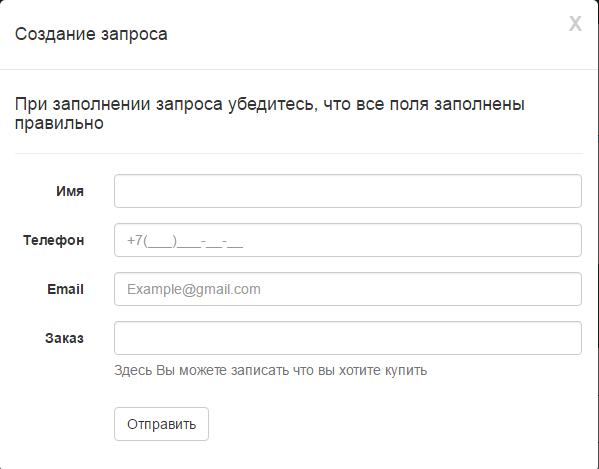


Рис.5

Страница «Монтаж»

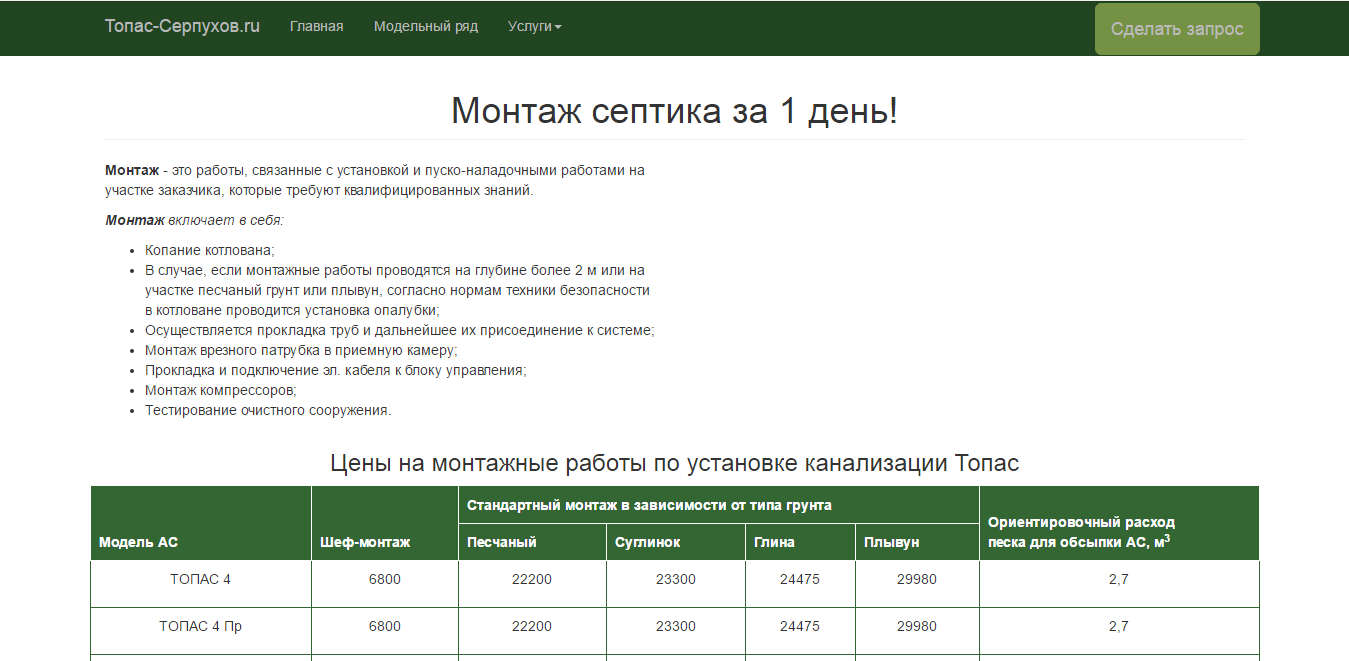


Рис.6

Страница «Доставка»

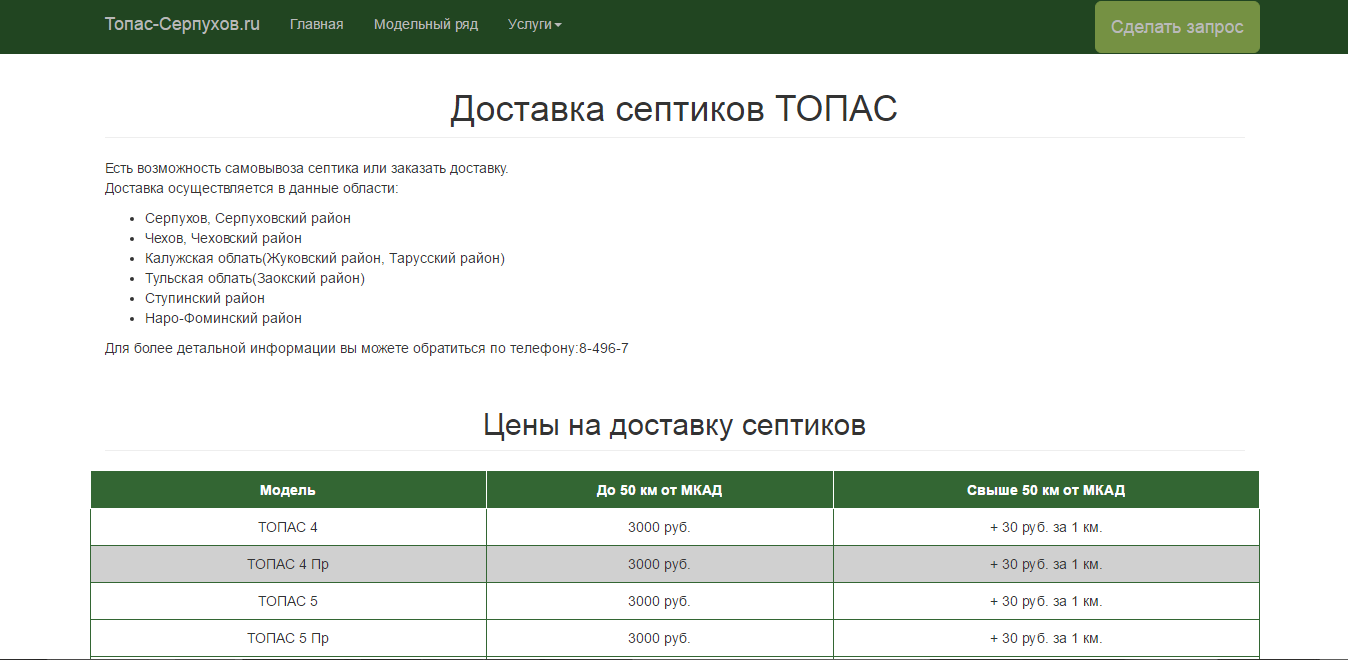


Рис. 7

Страница «Обслуживание»

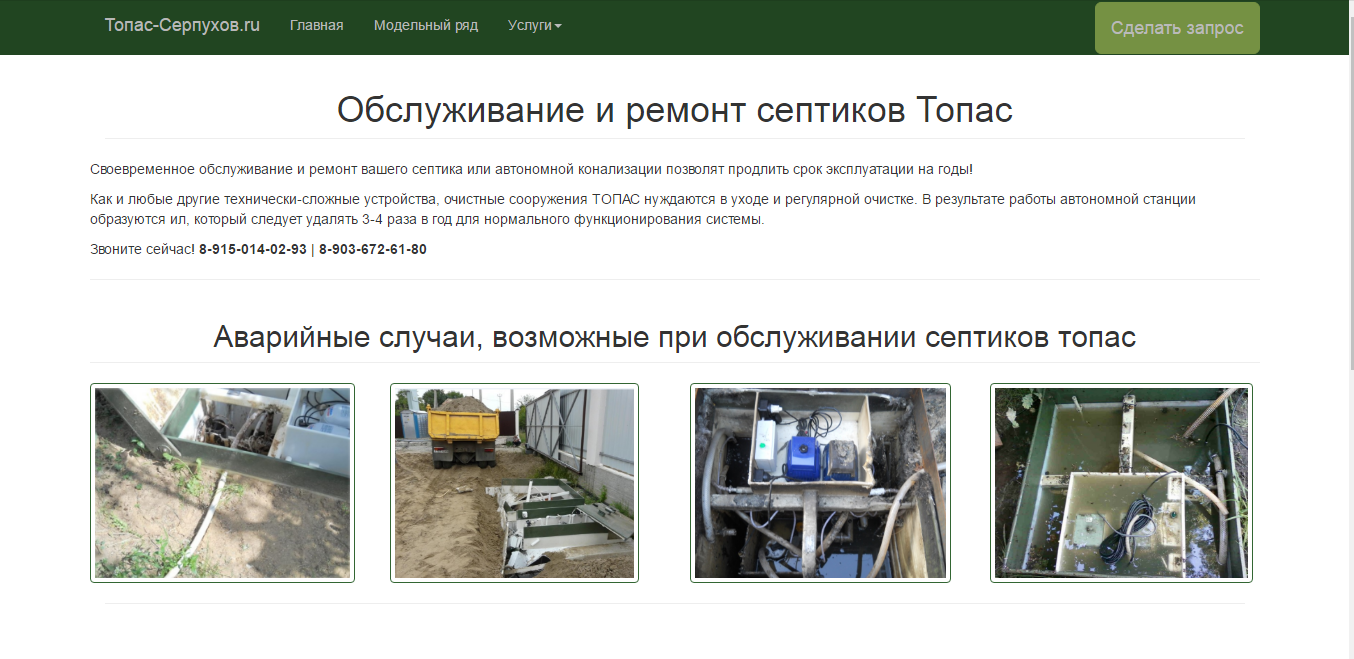


Рис. 8

Страница авторизации

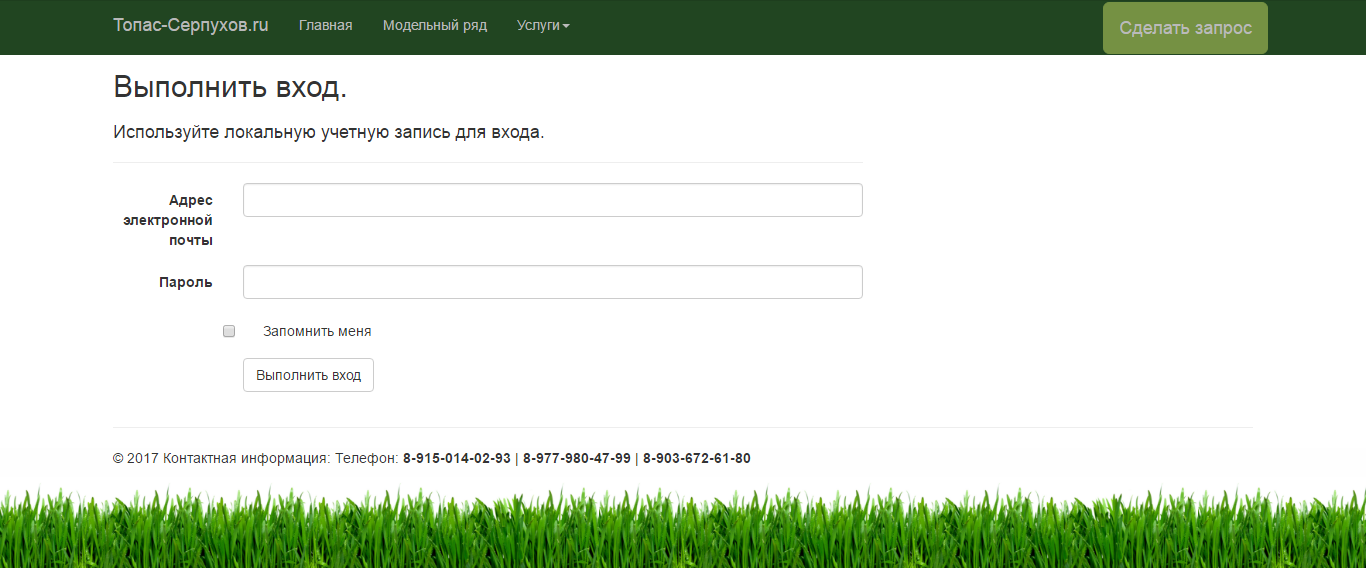


Рис. 9

Страница изменений данных в таблице «Модели»

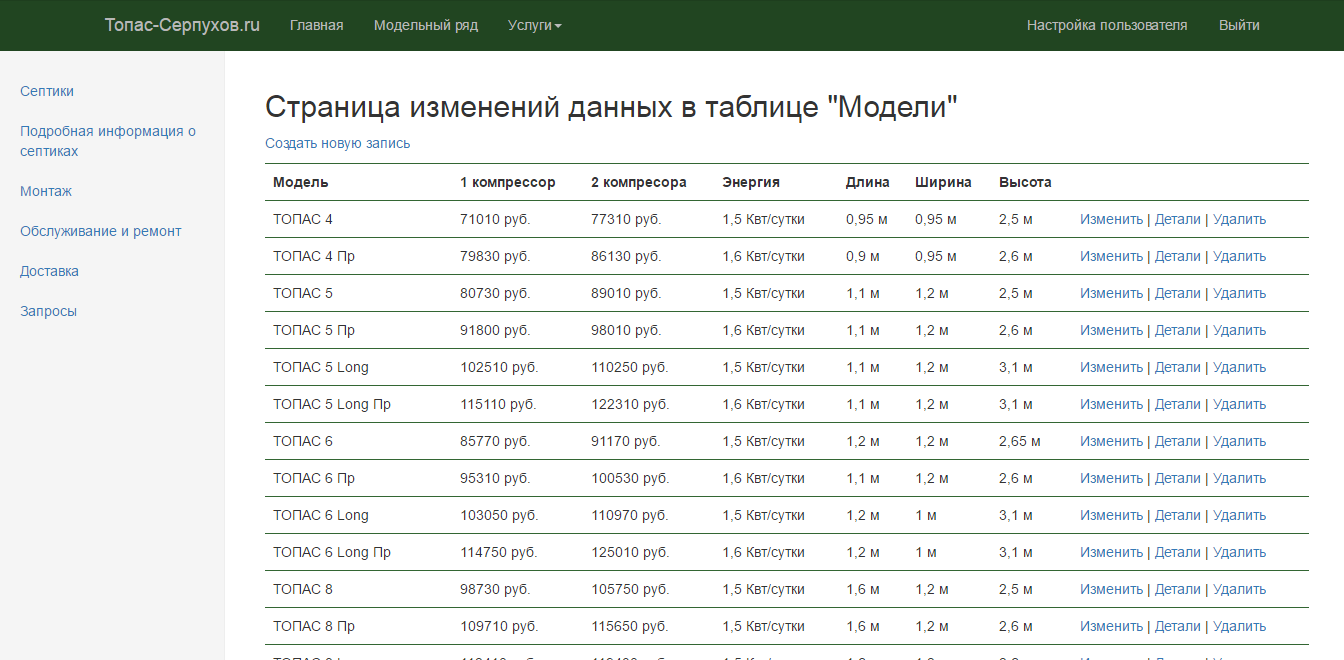


Рис. 10

Страница создания записи в таблице «Модели»

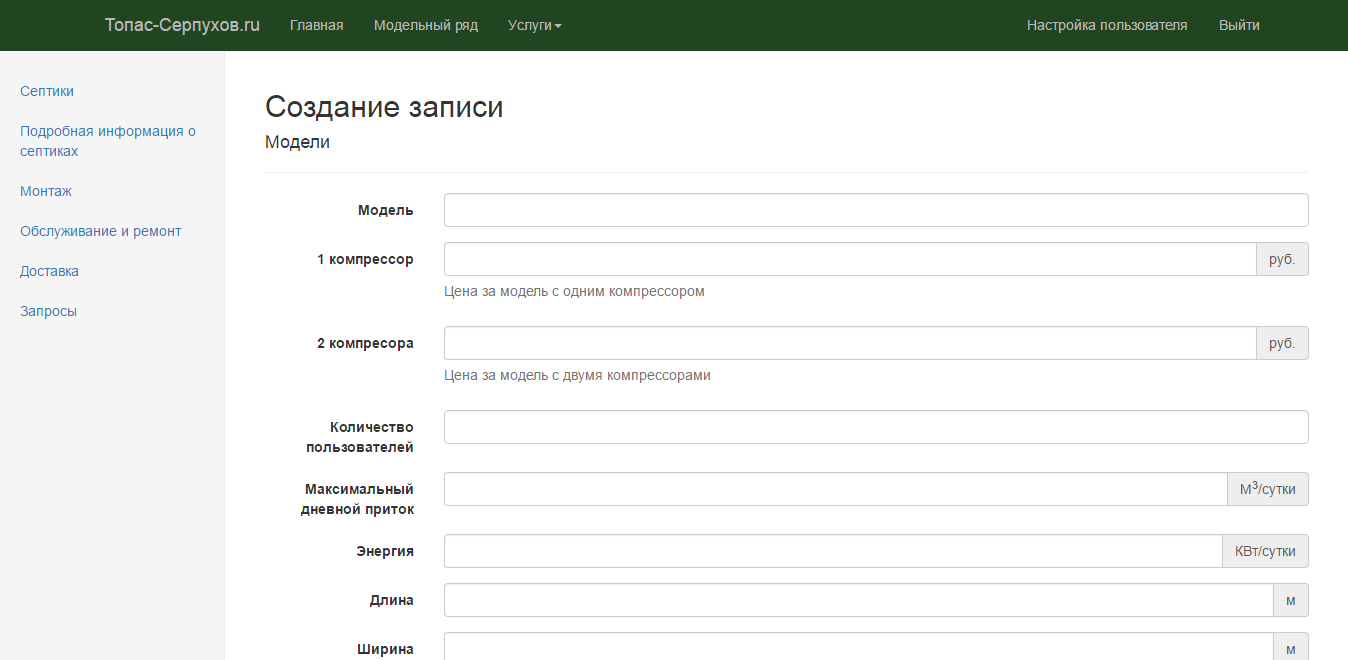


Рис. 11

Страница изменения записи в таблице «Модели»

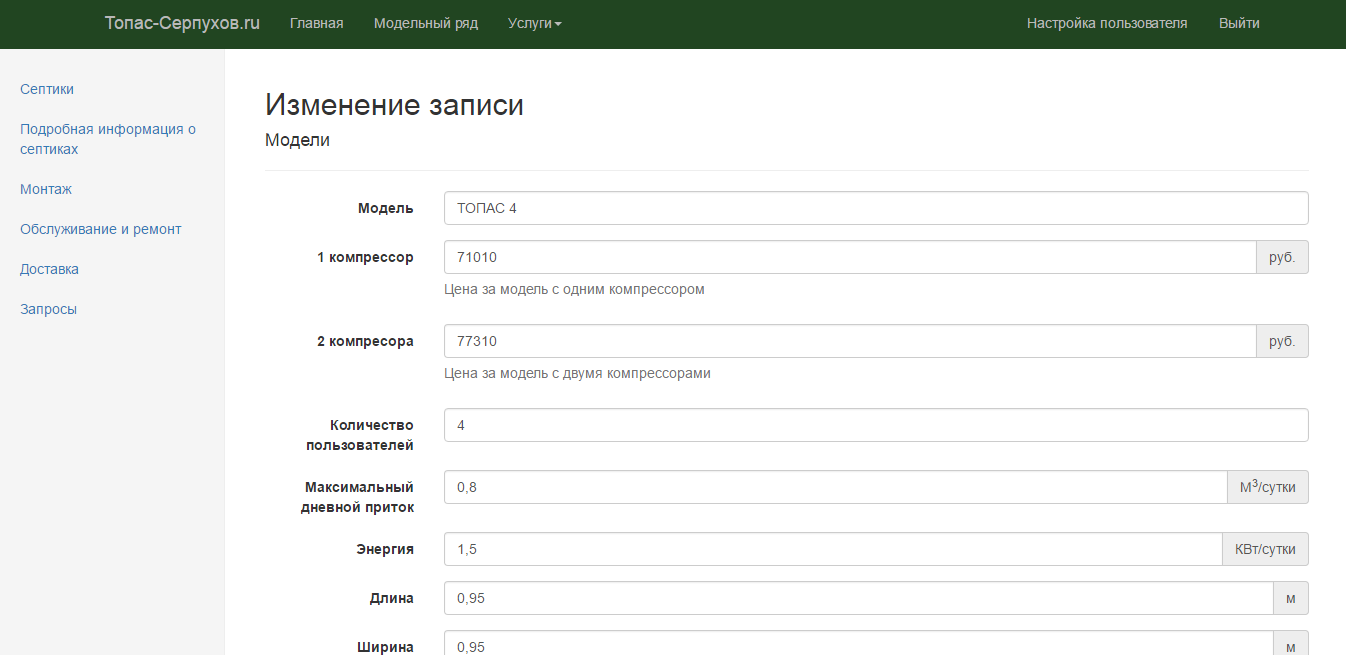


Рис. 12

Страница детальной информации о записи в таблице «Модели»

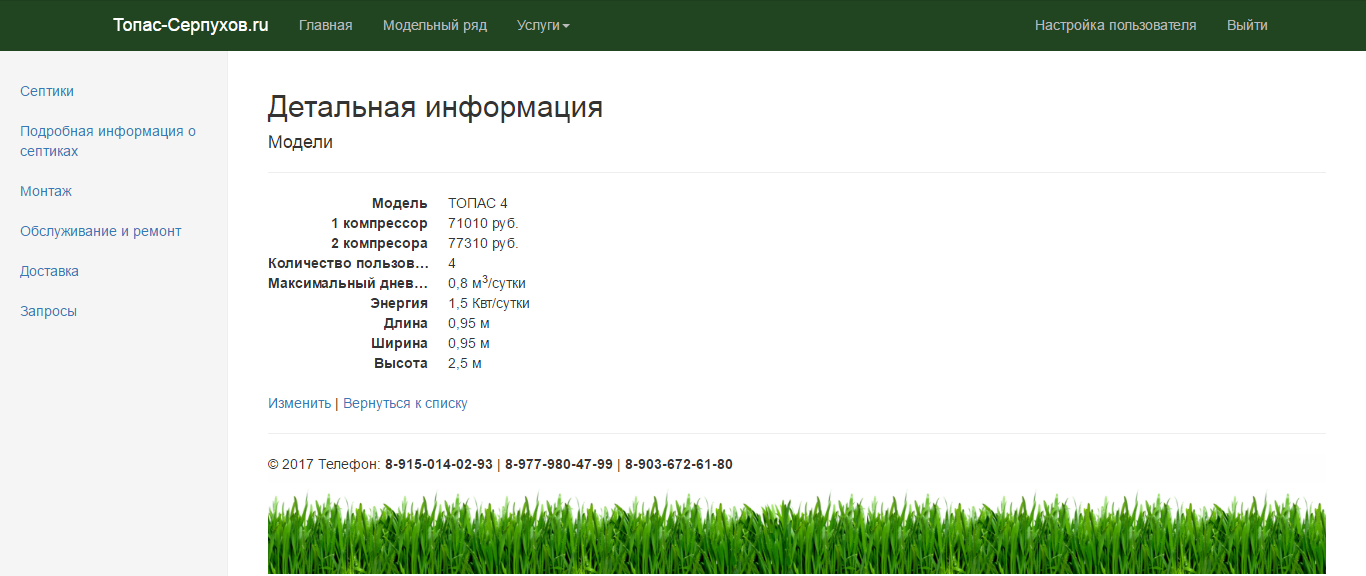


Рис. 13

Страница удаления записи в таблице «Модели»

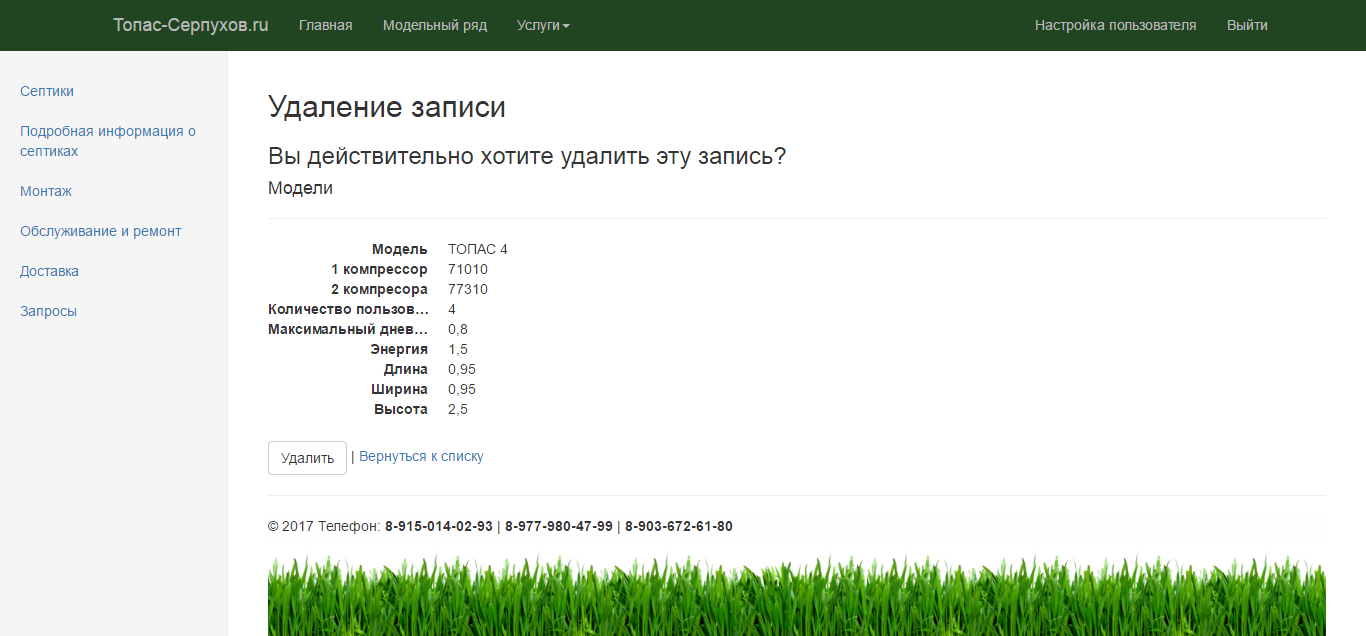


Рис. 14

Страница изменений данных в таблице «Подробная информация»

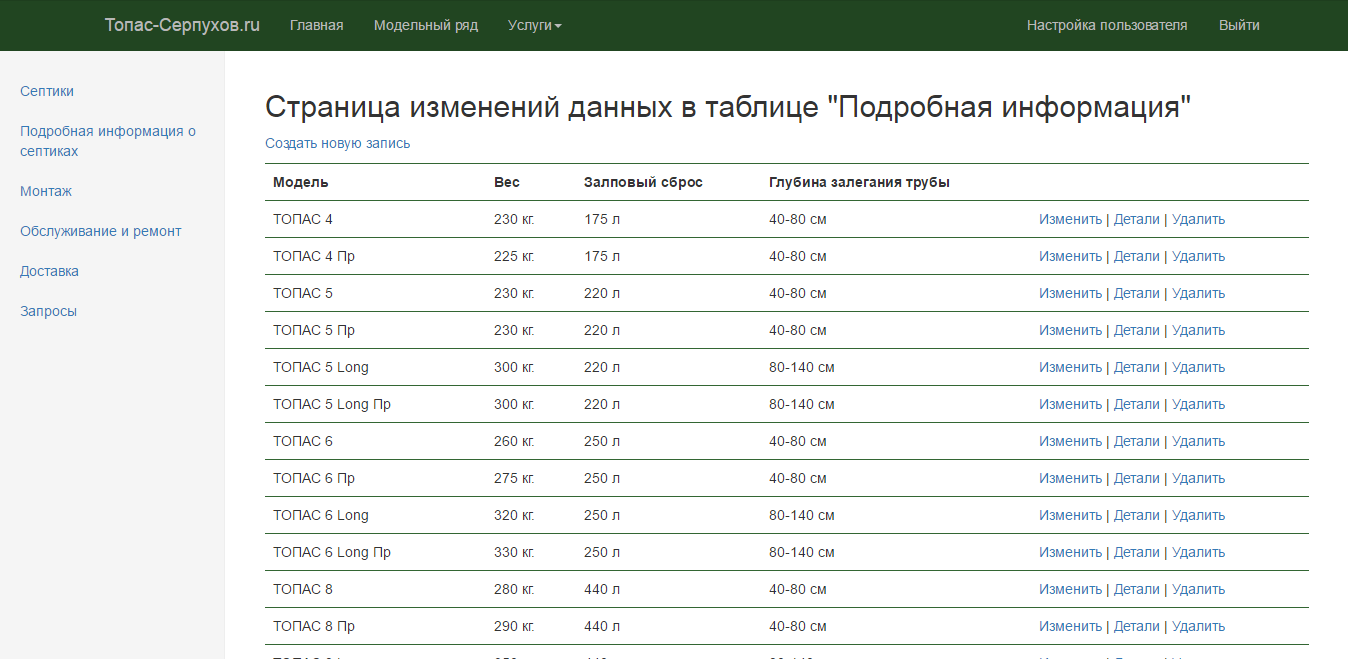


Рис. 15

Страница добавления записи в таблице «Подробная информация»

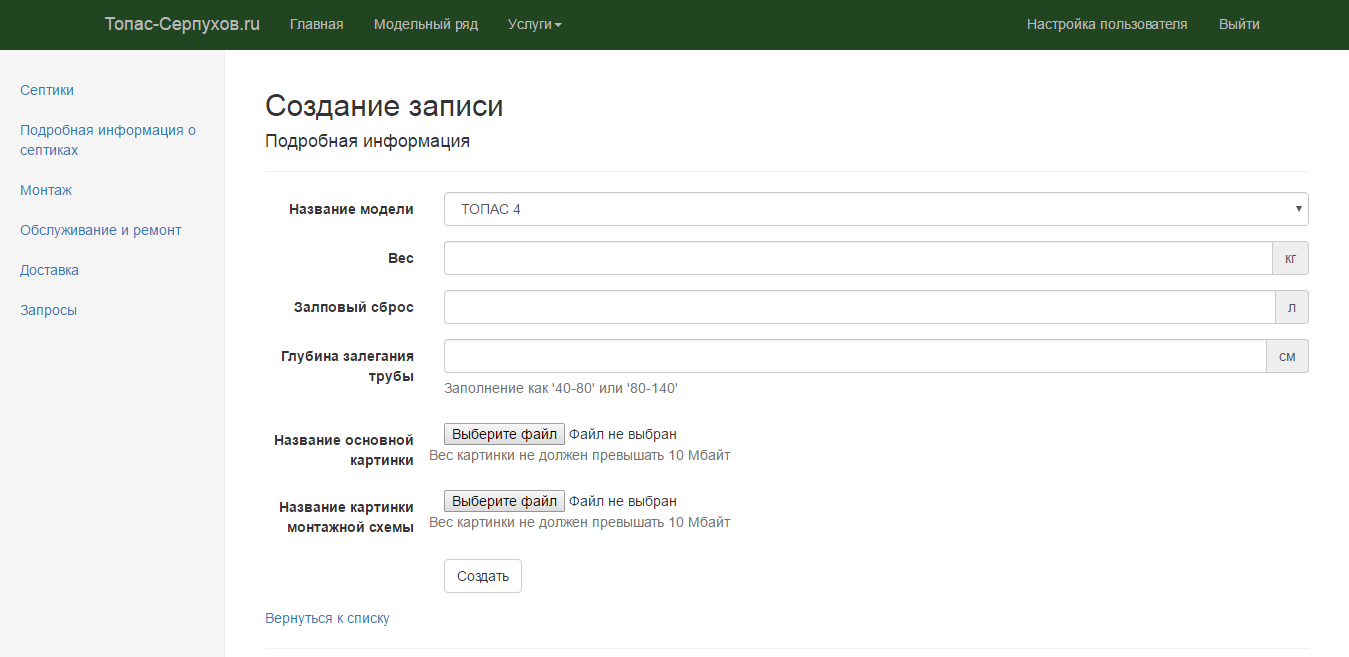


Рис. 16

Страница изменения записи в таблице «Подробная информация»

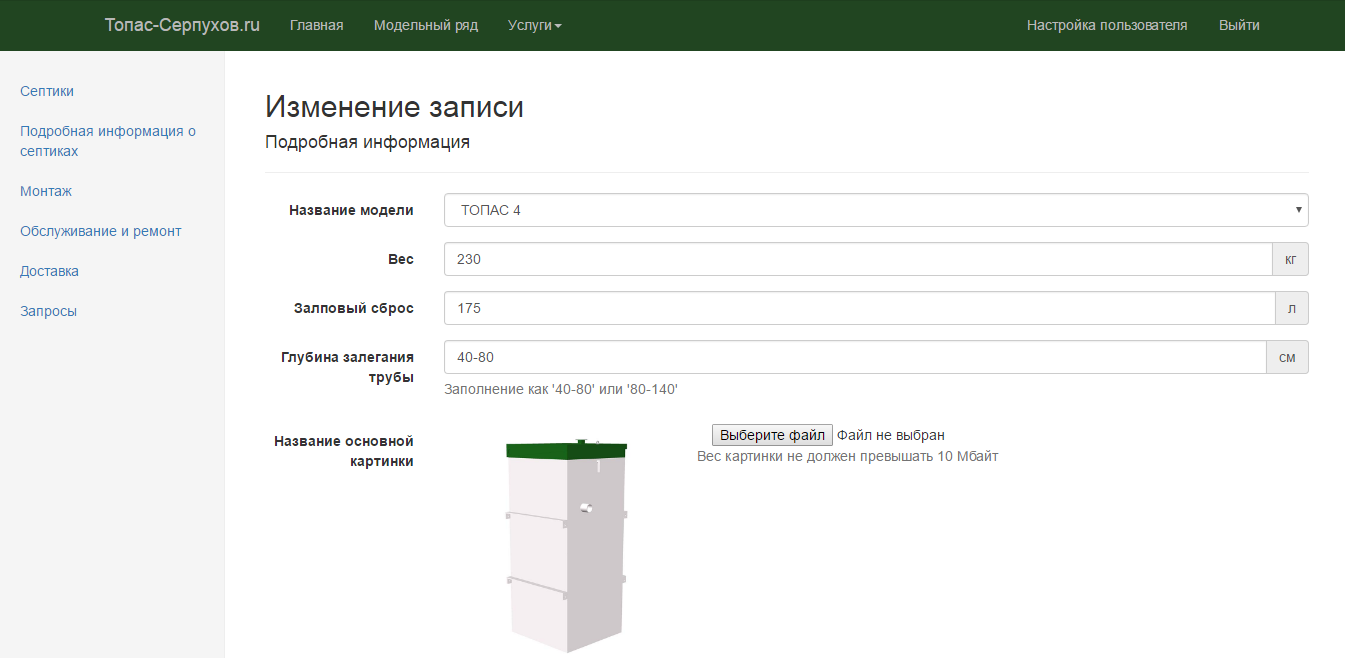


Рис. 17

Страница детальной информации о записи в таблице «Подробная информация»

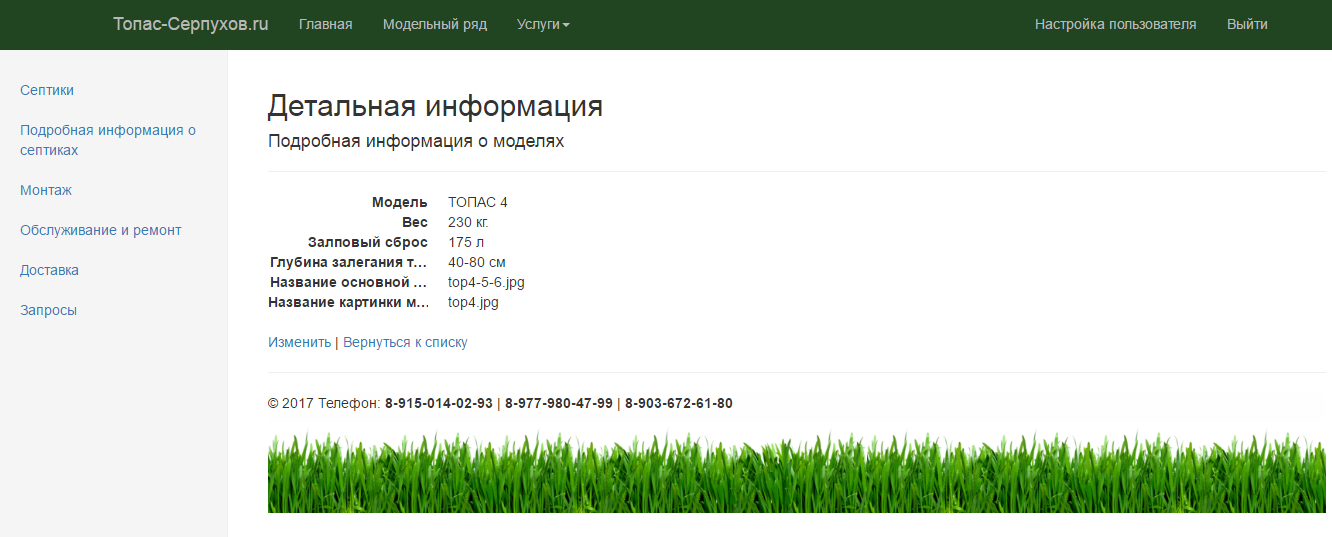


Рис. 18

Страница удаления записи в таблице «Подробная информация»

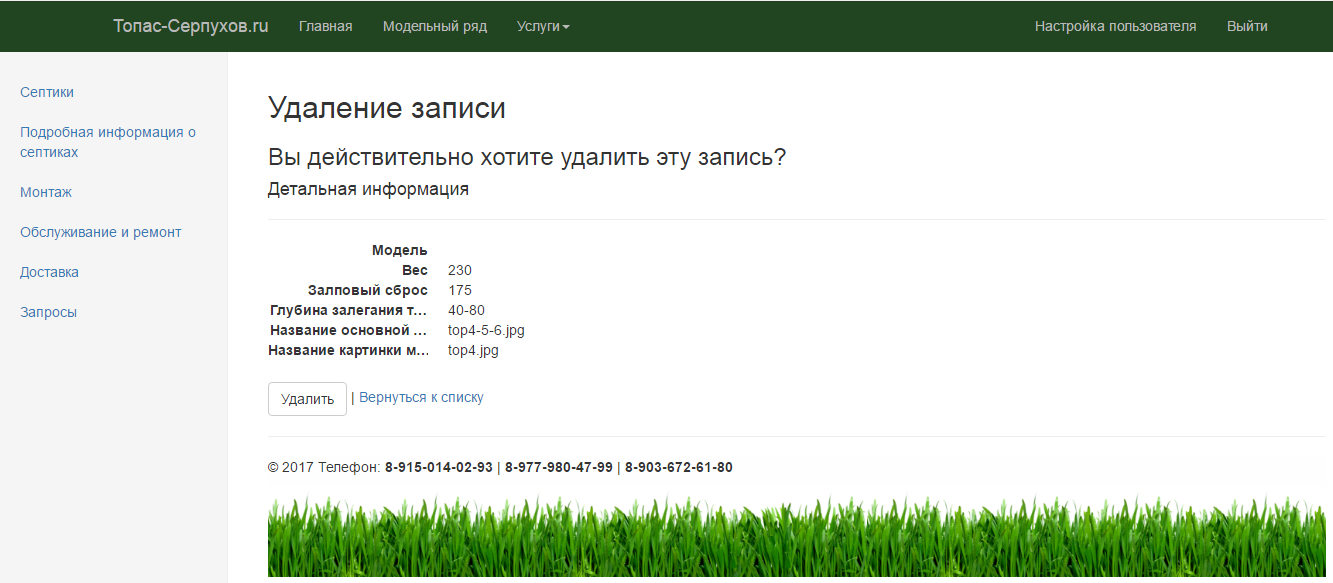


Рис. 19

Страница изменений данных в таблице «Монтаж»

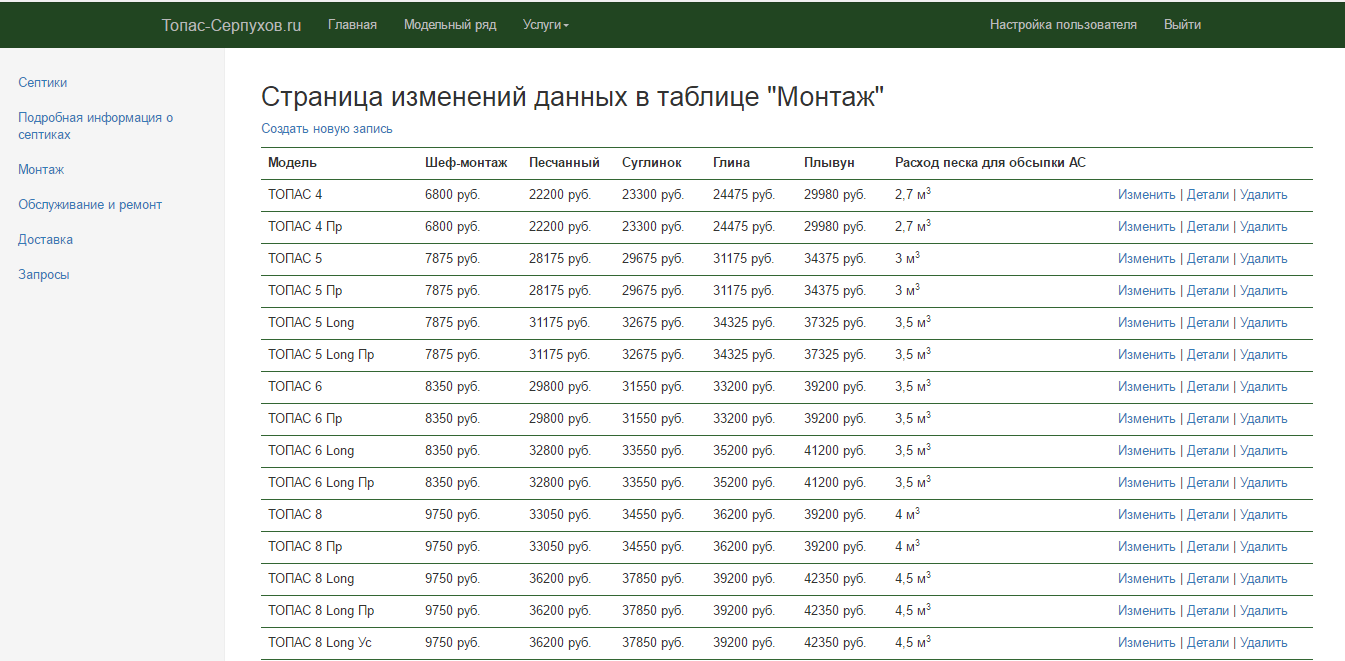


Рис. 20

Страница добавления записи в таблице «Монтаж»

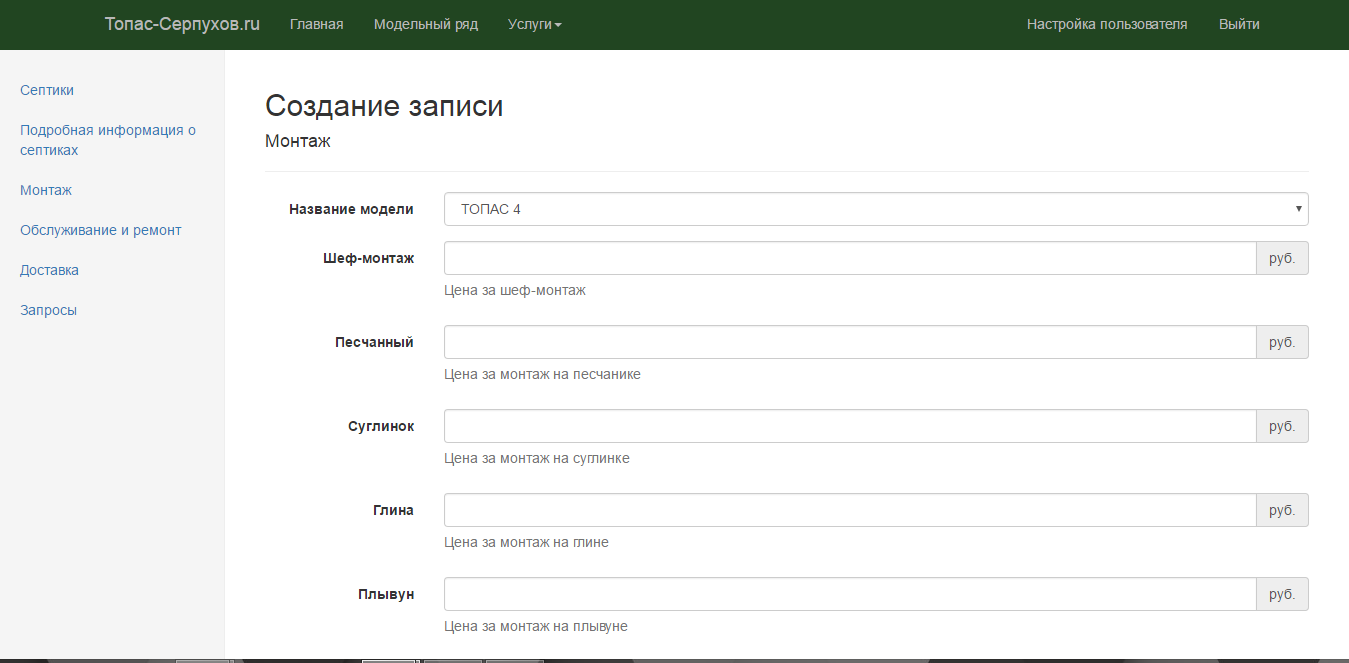


Рис. 21

Страница изменения записи в таблице «Подробная информация»

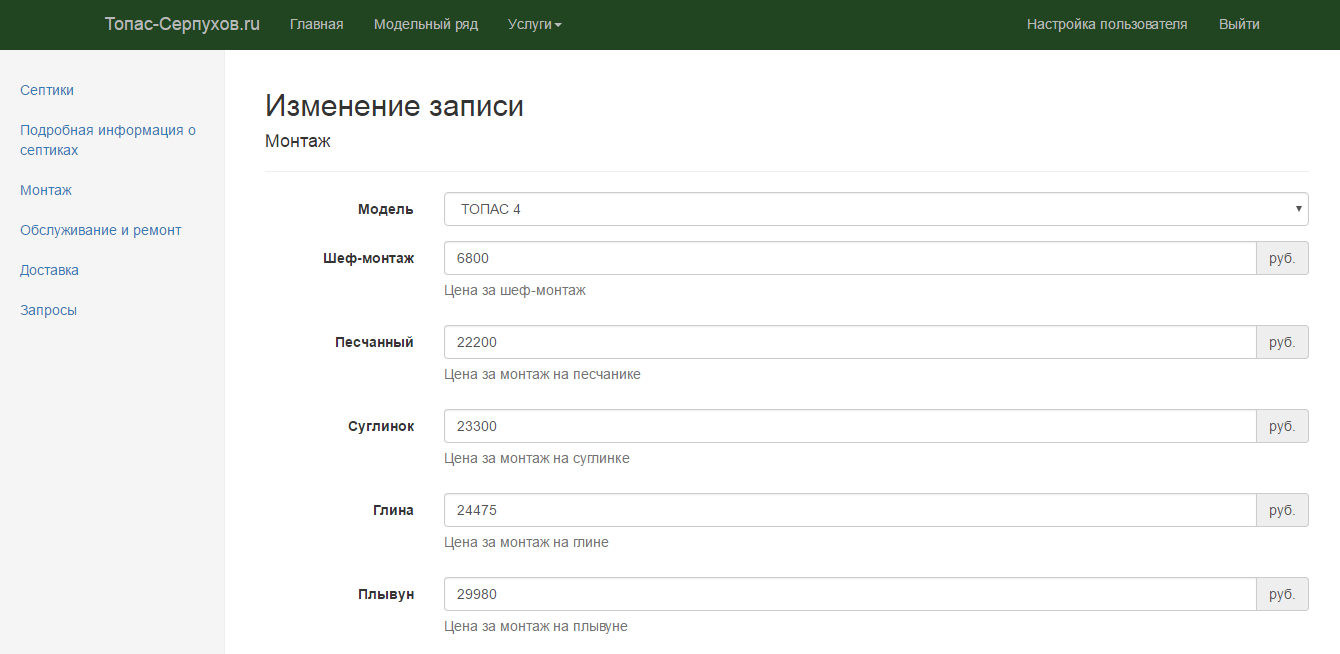


Рис. 22

Страница детальной информации о записи в таблице «Монтаж»

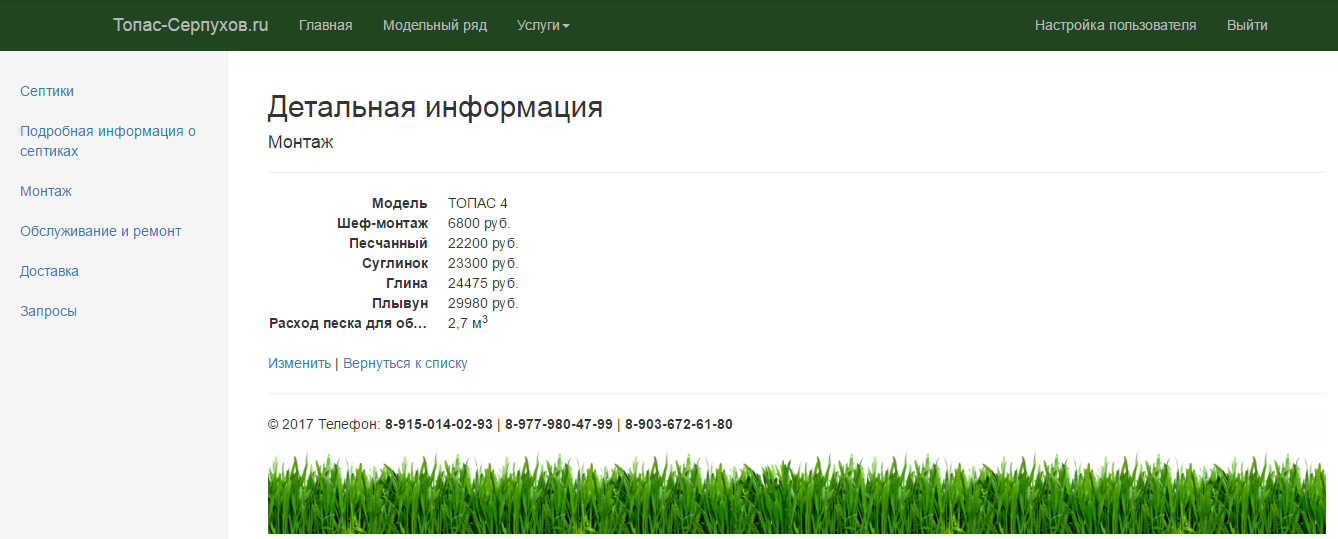


Рис. 23

Страница удаления записи в таблице «Монтаж»

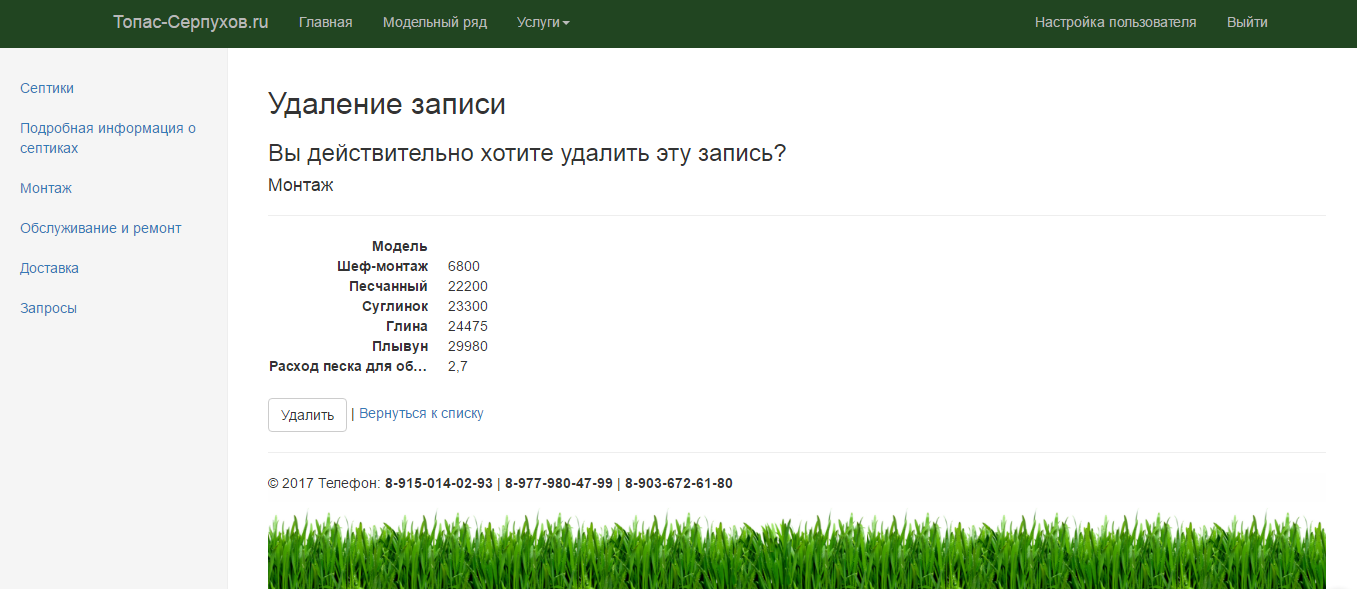


Рис. 24

Страница изменений данных в таблице «Обслуживание»

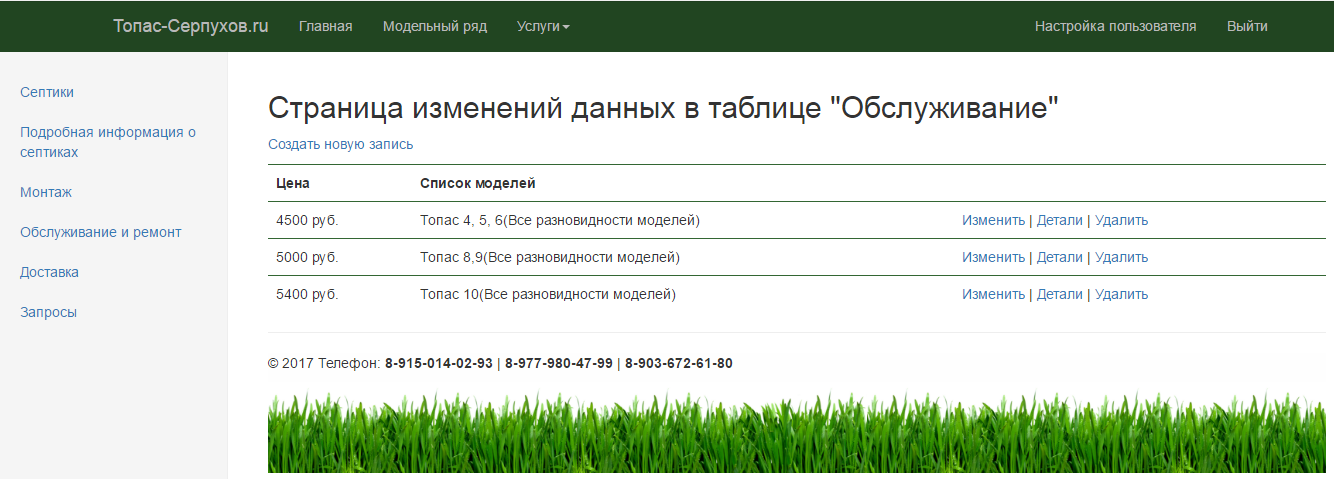


Рис. 25

Страница изменений данных в таблице «Доставка»

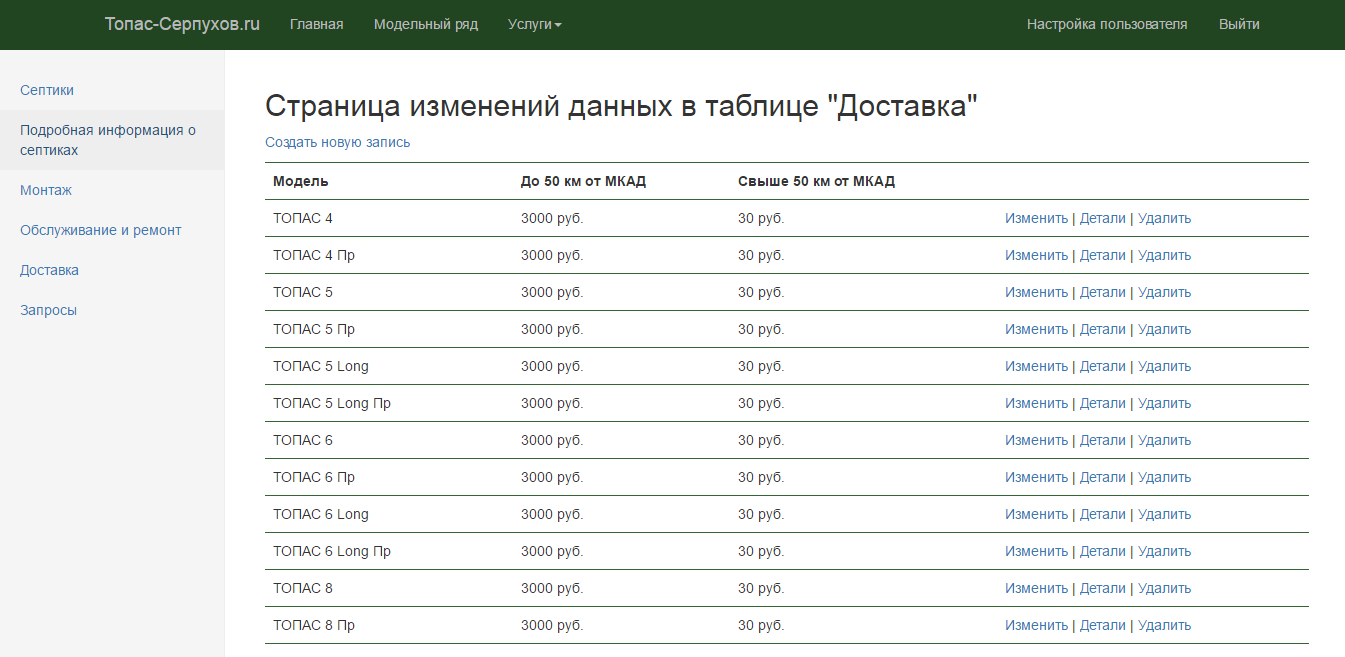


Рис. 26

Страница изменений данных в таблице «Запросы»

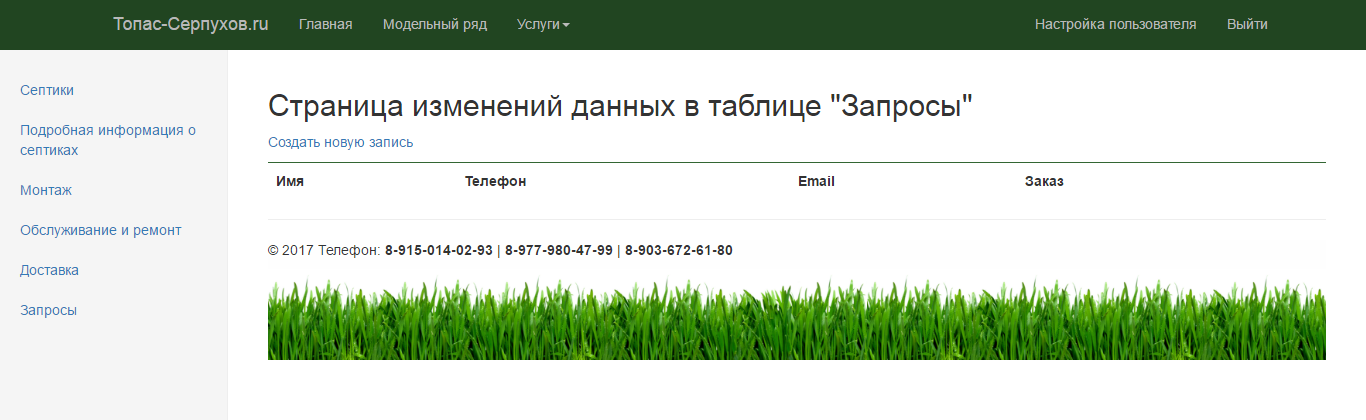


Рис. 26

* 1. Листинг программы

2.4.1. Листинг программных модулей

Листинг приложения представлен рисунками 27-49.

Листинг модели "SepticTank" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel.DataAnnotations;  using System.Linq;  using System.Web;  namespace ProjectTopas\_v\_1.\_2.Models  {  public class SepticTank  {  [Display(Name = "Наименование модели")]  public int Id { get; set; }  [Display(Name = "Модель")]  public string Model { get; set; }  [Display(Name = "1 компрессор")]  public int Cost1Comp { get; set; } //Цена с 1 компрессором  [Display(Name = "2 компресора")]  public int Cost2Comp { get; set; } //Цена с 2 компрессорами  [Display(Name = "Количество пользователей")]  public int CountUsers { get; set; }  [Display(Name = "Максимальный дневной приток")]  public double InflowDrain { get; set; } //Максимальный дневной приток  [Display(Name = "Энергия")]  public double Energy { get; set; }  [Display(Name = "Длина")]  public double Lenght { get; set; }  [Display(Name = "Ширина")]  public double Weight { get; set; }  [Display(Name = "Высота")]  public double Height { get; set; }  public ICollection<SepticSpecification> SepticSpecification { get; set; } //Привязка модели "Технические характеристики"  public ICollection<Installation> Installation { get; set; } //Привязка модели "Монтаж"  public ICollection<Delivery> Delivery { get; set; } //Привязка модели "Доставка"  public SepticTank()  {  SepticSpecification = new List<SepticSpecification>();  Installation = new List<Installation>();  Delivery = new List<Delivery>();  }  }  } |

Рис.27

Листинг модели "SepticSpecification" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel.DataAnnotations;  using System.Linq;  using System.Web;  namespace ProjectTopas\_v\_1.\_2.Models  {  public class SepticSpecification  {  public int Id { get; set; }  //Технические характиристики  [Display(Name = "Вес")]  public int Weight { get; set; } //: Вес  [Display(Name = "Залповый сброс")]  public int VolleyRes { get; set; } //:Залповый сброс  [Display(Name = "Глубина залегания трубы")]  public string PipeDepth { get; set; } //Глубина залегания трубы  [Display(Name = "Название основной картинки")]  public string PathPicMain { get; set; } // Основная картинка  [Display(Name = "Название картинки монтажной схемы")]  public string PathMontagePic { get; set; } //Монтажная схема  [Display(Name = "Наименование модели")]  public int? SepticTankId { get; set; }  public SepticTank SepticTank { get; set; }  }  } |

Рис.28

Листинг модели "SepticService" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel.DataAnnotations;  using System.Linq;  using System.Web;  namespace ProjectTopas\_v\_1.\_2.Models  {  public class SepticService  {  public int Id { get; set; }  [Display(Name = "Цена")]  public int Cost { get; set; }  [Display(Name = "Список моделей")]  public string Models { get; set; } //Перечисление моделей по которым возможно обслуживание  }  } |

Рис.29

Листинг модели "Installation" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Data.Entity;  using System.ComponentModel.DataAnnotations;  namespace ProjectTopas\_v\_1.\_2.Models  {  public class Installation  {  public int Id { get; set; }  [Display(Name = "Шеф-монтаж")]  public int CostShief { get; set; }  [Display(Name = "Песчанный")]  public int InstSand { get; set; } //Стандартный монтаж на песчаном грунте  [Display(Name = "Суглинок")]  public int InstLoam { get; set; } //--/-- на суглинке  [Display(Name = "Глина")]  public int InstClay { get; set; } //--/-- на глине  [Display(Name = "Плывун")]  public int InstQuickSand { get; set; } // --/-- на плывуне  [Display(Name = "Расход песка для обсыпки АС")]  public double ExpenSand { get; set; } //Расход песка для обсыпки АС  [Display(Name = "Наименование модели")]  public int? SepticTankId { get; set; }  public SepticTank SepticTank { get; set; }  }  } |

Рис.30

Листинг модели "Delivery" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.ComponentModel.DataAnnotations;  namespace ProjectTopas\_v\_1.\_2.Models  {  public class Delivery  {  public int Id { get; set; }  [Display(Name = "До 50 км от МКАД")]  public int BefMK { get; set; }  [Display(Name = "Свыше 50 км от МКАД")]  public int AftMK { get; set; }  [Display(Name = "Наименование модели")]  public int? SepticTankId { get; set; }  public SepticTank SepticTank { get; set; }  }  } |

Рис.31

Листинг модели "Request" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel.DataAnnotations;  using System.Linq;  using System.Web;  namespace ProjectTopas\_v\_1.\_2.Models  {  public class Request  {  public int Id { get; set; }  [Display(Name = "Имя")]  [Required]  public string Name { get; set; }  [Display(Name = "Телефон")]  [Required]  public string Phone { get; set; }  [RegularExpression(@"[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}", ErrorMessage = "Некорректный адрес")]  [Required]  public string Email { get; set; }  [Display(Name = "Заказ")]  [Required]  public string Comment { get; set; }  }  } |

Рис.32

Листинг модели "SepticTankContext" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Data.Entity;  namespace ProjectTopas\_v\_1.\_2.Models  {  public class SepticTankContext : DbContext  {  public SepticTankContext() : base("DefaultConnection")  { }  public DbSet<SepticTank> SepticTanks { get; set; }  public DbSet<Installation> Installations { get; set; }  public DbSet<SepticSpecification> SepticSpecifications { get; set; }  public DbSet<SepticService> SepticServices { get; set; }  public DbSet<Request> Requests { get; set; }  public System.Data.Entity.DbSet<ProjectTopas\_v\_1.\_2.Models.Delivery> Deliveries { get; set; }  }  } |

Рис.33

Листинг контроллера "AccountController" следующий:

|  |
| --- |
| using System;  using System.Globalization;  using System.Linq;  using System.Security.Claims;  using System.Threading.Tasks;  using System.Web;  using System.Web.Mvc;  using Microsoft.AspNet.Identity;  using Microsoft.AspNet.Identity.Owin;  using Microsoft.Owin.Security;  using ProjectTopas\_v\_1.\_2.Models;  namespace ProjectTopas\_v\_1.\_2.Controllers  {  [Authorize]  public class AccountController : Controller  {  private ApplicationSignInManager \_signInManager;  private ApplicationUserManager \_userManager;  public AccountController()  {  }  public AccountController(ApplicationUserManager userManager, ApplicationSignInManager signInManager )  {  UserManager = userManager;  SignInManager = signInManager;  }  public ApplicationSignInManager SignInManager  {  get  {  return \_signInManager ?? HttpContext.GetOwinContext().Get<ApplicationSignInManager>();  }  private set  {  \_signInManager = value;  }  }  public ApplicationUserManager UserManager  {  get  {  return \_userManager ?? HttpContext.GetOwinContext().GetUserManager<ApplicationUserManager>();  }  private set  {  \_userManager = value;  }  }  //  // GET: /Account/Login  [AllowAnonymous]  public ActionResult Login(string returnUrl)  {  ViewBag.ReturnUrl = returnUrl;  return View();  }  //  // POST: /Account/Login  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)  {  if (!ModelState.IsValid)  {  return View(model);  }  // Сбои при входе не приводят к блокированию учетной записи  // Чтобы ошибки при вводе пароля инициировали блокирование учетной записи, замените на shouldLockout: true  var result = await SignInManager.PasswordSignInAsync(model.Email, model.Password, model.RememberMe, shouldLockout: false);  switch (result)  {  case SignInStatus.Success:  return RedirectToLocal(returnUrl);  case SignInStatus.LockedOut:  return View("Lockout");  case SignInStatus.RequiresVerification:  return RedirectToAction("SendCode", new { ReturnUrl = returnUrl, RememberMe = model.RememberMe });  case SignInStatus.Failure:  default:  ModelState.AddModelError("", "Неудачная попытка входа.");  return View(model);  }  }  //  // GET: /Account/VerifyCode  [AllowAnonymous]  public async Task<ActionResult> VerifyCode(string provider, string returnUrl, bool rememberMe)  {  // Требовать предварительный вход пользователя с помощью имени пользователя и пароля или внешнего имени входа  if (!await SignInManager.HasBeenVerifiedAsync())  {  return View("Error");  }  return View(new VerifyCodeViewModel { Provider = provider, ReturnUrl = returnUrl, RememberMe = rememberMe });  }  //  // POST: /Account/VerifyCode  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> VerifyCode(VerifyCodeViewModel model)  {  if (!ModelState.IsValid)  {  return View(model);  }  // Приведенный ниже код защищает от атак методом подбора, направленных на двухфакторные коды.  // Если пользователь введет неправильные коды за указанное время, его учетная запись  // будет заблокирована на заданный период.  // Параметры блокирования учетных записей можно настроить в IdentityConfig  var result = await SignInManager.TwoFactorSignInAsync(model.Provider, model.Code, isPersistent: model.RememberMe, rememberBrowser: model.RememberBrowser);  switch (result)  {  case SignInStatus.Success:  return RedirectToLocal(model.ReturnUrl);  case SignInStatus.LockedOut:  return View("Lockout");  case SignInStatus.Failure:  default:  ModelState.AddModelError("", "Неправильный код.");  return View(model);  }  }  //  // GET: /Account/Register    public ActionResult Register()  {  return View();  }  //  // POST: /Account/Register  [HttpPost]  [ValidateAntiForgeryToken]    public async Task<ActionResult> Register(RegisterViewModel model)  {  if (ModelState.IsValid)  {  var user = new ApplicationUser { UserName = model.Email, Email = model.Email };  var result = await UserManager.CreateAsync(user, model.Password);  if (result.Succeeded)  {  await SignInManager.SignInAsync(user, isPersistent:false, rememberBrowser:false);    // Дополнительные сведения о том, как включить подтверждение учетной записи и сброс пароля, см. по адресу: http://go.microsoft.com/fwlink/?LinkID=320771  // Отправка сообщения электронной почты с этой ссылкой  // string code = await UserManager.GenerateEmailConfirmationTokenAsync(user.Id);  // var callbackUrl = Url.Action("ConfirmEmail", "Account", new { userId = user.Id, code = code }, protocol: Request.Url.Scheme);  // await UserManager.SendEmailAsync(user.Id, "Подтверждение учетной записи", "Подтвердите вашу учетную запись, щелкнув <a href=\"" + callbackUrl + "\">здесь</a>");  return RedirectToAction("Index", "S3pticTnk3dit");  }  AddErrors(result);  }  // Появление этого сообщения означает наличие ошибки; повторное отображение формы  return View(model);  }  //  // GET: /Account/ConfirmEmail  [AllowAnonymous]  public async Task<ActionResult> ConfirmEmail(string userId, string code)  {  if (userId == null || code == null)  {  return View("Error");  }  var result = await UserManager.ConfirmEmailAsync(userId, code);  return View(result.Succeeded ? "ConfirmEmail" : "Error");  }  //  // GET: /Account/ForgotPassword  [AllowAnonymous]  public ActionResult ForgotPassword()  {  return View();  }  //  // POST: /Account/ForgotPassword  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> ForgotPassword(ForgotPasswordViewModel model)  {  if (ModelState.IsValid)  {  var user = await UserManager.FindByNameAsync(model.Email);  if (user == null || !(await UserManager.IsEmailConfirmedAsync(user.Id)))  {  // Не показывать, что пользователь не существует или не подтвержден  return View("ForgotPasswordConfirmation");  }  // Дополнительные сведения о том, как включить подтверждение учетной записи и сброс пароля, см. по адресу: http://go.microsoft.com/fwlink/?LinkID=320771  // Отправка сообщения электронной почты с этой ссылкой  // string code = await UserManager.GeneratePasswordResetTokenAsync(user.Id);  // var callbackUrl = Url.Action("ResetPassword", "Account", new { userId = user.Id, code = code }, protocol: Request.Url.Scheme);  // await UserManager.SendEmailAsync(user.Id, "Сброс пароля", "Сбросьте ваш пароль, щелкнув <a href=\"" + callbackUrl + "\">здесь</a>");  // return RedirectToAction("ForgotPasswordConfirmation", "Account");  }  // Появление этого сообщения означает наличие ошибки; повторное отображение формы  return View(model);  }  //  // GET: /Account/ForgotPasswordConfirmation  [AllowAnonymous]  public ActionResult ForgotPasswordConfirmation()  {  return View();  }  //  // GET: /Account/ResetPassword  [AllowAnonymous]  public ActionResult ResetPassword(string code)  {  return code == null ? View("Error") : View();  }  //  // POST: /Account/ResetPassword  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> ResetPassword(ResetPasswordViewModel model)  {  if (!ModelState.IsValid)  {  return View(model);  }  var user = await UserManager.FindByNameAsync(model.Email);  if (user == null)  {  // Не показывать, что пользователь не существует  return RedirectToAction("ResetPasswordConfirmation", "Account");  }  var result = await UserManager.ResetPasswordAsync(user.Id, model.Code, model.Password);  if (result.Succeeded)  {  return RedirectToAction("ResetPasswordConfirmation", "Account");  }  AddErrors(result);  return View();  }  //  // GET: /Account/ResetPasswordConfirmation  [AllowAnonymous]  public ActionResult ResetPasswordConfirmation()  {  return View();  }  //  // POST: /Account/ExternalLogin  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public ActionResult ExternalLogin(string provider, string returnUrl)  {  // Запрос перенаправления к внешнему поставщику входа  return new ChallengeResult(provider, Url.Action("ExternalLoginCallback", "Account", new { ReturnUrl = returnUrl }));  }  //  // GET: /Account/SendCode  [AllowAnonymous]  public async Task<ActionResult> SendCode(string returnUrl, bool rememberMe)  {  var userId = await SignInManager.GetVerifiedUserIdAsync();  if (userId == null)  {  return View("Error");  }  var userFactors = await UserManager.GetValidTwoFactorProvidersAsync(userId);  var factorOptions = userFactors.Select(purpose => new SelectListItem { Text = purpose, Value = purpose }).ToList();  return View(new SendCodeViewModel { Providers = factorOptions, ReturnUrl = returnUrl, RememberMe = rememberMe });  }  //  // POST: /Account/SendCode  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> SendCode(SendCodeViewModel model)  {  if (!ModelState.IsValid)  {  return View();  }  // Создание и отправка маркера  if (!await SignInManager.SendTwoFactorCodeAsync(model.SelectedProvider))  {  return View("Error");  }  return RedirectToAction("VerifyCode", new { Provider = model.SelectedProvider, ReturnUrl = model.ReturnUrl, RememberMe = model.RememberMe });  }  //  // GET: /Account/ExternalLoginCallback  [AllowAnonymous]  public async Task<ActionResult> ExternalLoginCallback(string returnUrl)  {  var loginInfo = await AuthenticationManager.GetExternalLoginInfoAsync();  if (loginInfo == null)  {  return RedirectToAction("Login");  }  // Выполнение входа пользователя посредством данного внешнего поставщика входа, если у пользователя уже есть имя входа  var result = await SignInManager.ExternalSignInAsync(loginInfo, isPersistent: false);  switch (result)  {  case SignInStatus.Success:  return RedirectToLocal(returnUrl);  case SignInStatus.LockedOut:  return View("Lockout");  case SignInStatus.RequiresVerification:  return RedirectToAction("SendCode", new { ReturnUrl = returnUrl, RememberMe = false });  case SignInStatus.Failure:  default:  // Если у пользователя нет учетной записи, то ему предлагается создать ее  ViewBag.ReturnUrl = returnUrl;  ViewBag.LoginProvider = loginInfo.Login.LoginProvider;  return View("ExternalLoginConfirmation", new ExternalLoginConfirmationViewModel { Email = loginInfo.Email });  }  }  //  // POST: /Account/ExternalLoginConfirmation  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> ExternalLoginConfirmation(ExternalLoginConfirmationViewModel model, string returnUrl)  {  if (User.Identity.IsAuthenticated)  {  return RedirectToAction("Index", "Manage");  }  if (ModelState.IsValid)  {  // Получение сведений о пользователе от внешнего поставщика входа  var info = await AuthenticationManager.GetExternalLoginInfoAsync();  if (info == null)  {  return View("ExternalLoginFailure");  }  var user = new ApplicationUser { UserName = model.Email, Email = model.Email };  var result = await UserManager.CreateAsync(user);  if (result.Succeeded)  {  result = await UserManager.AddLoginAsync(user.Id, info.Login);  if (result.Succeeded)  {  await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);  return RedirectToLocal(returnUrl);  }  }  AddErrors(result);  }  ViewBag.ReturnUrl = returnUrl;  return View(model);  }  //  // POST: /Account/LogOff  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult LogOff()  {  AuthenticationManager.SignOut(DefaultAuthenticationTypes.ApplicationCookie);  return RedirectToAction("Index", "Home");  }  //  // GET: /Account/ExternalLoginFailure  [AllowAnonymous]  public ActionResult ExternalLoginFailure()  {  return View();  }  protected override void Dispose(bool disposing)  {  if (disposing)  {  if (\_userManager != null)  {  \_userManager.Dispose();  \_userManager = null;  }  if (\_signInManager != null)  {  \_signInManager.Dispose();  \_signInManager = null;  }  }  base.Dispose(disposing);  }  #region Вспомогательные приложения  // Используется для защиты от XSRF-атак при добавлении внешних имен входа  private const string XsrfKey = "XsrfId";  private IAuthenticationManager AuthenticationManager  {  get  {  return HttpContext.GetOwinContext().Authentication;  }  }  private void AddErrors(IdentityResult result)  {  foreach (var error in result.Errors)  {  ModelState.AddModelError("", error);  }  }  private ActionResult RedirectToLocal(string returnUrl)  {  if (Url.IsLocalUrl(returnUrl))  {  return Redirect(returnUrl);  }  return RedirectToAction("Index", "S3pticTnk3dit");  }  internal class ChallengeResult : HttpUnauthorizedResult  {  public ChallengeResult(string provider, string redirectUri)  : this(provider, redirectUri, null)  {  }  public ChallengeResult(string provider, string redirectUri, string userId)  {  LoginProvider = provider;  RedirectUri = redirectUri;  UserId = userId;  }  public string LoginProvider { get; set; }  public string RedirectUri { get; set; }  public string UserId { get; set; }  public override void ExecuteResult(ControllerContext context)  {  var properties = new AuthenticationProperties { RedirectUri = RedirectUri };  if (UserId != null)  {  properties.Dictionary[XsrfKey] = UserId;  }  context.HttpContext.GetOwinContext().Authentication.Challenge(properties, LoginProvider);  }  }  #endregion  }  } |

Рис.34

Листинг контроллера "DeliverController" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Data;  using System.Data.Entity;  using System.Linq;  using System.Threading.Tasks;  using System.Net;  using System.Web;  using System.Web.Mvc;  using ProjectTopas\_v\_1.\_2.Models;  namespace ProjectTopas\_v\_1.\_2.Controllers  {  [Authorize]  public class D3liv3rController : Controller  {  private SepticTankContext db = new SepticTankContext();  // GET: D3liv3r  public async Task<ActionResult> Index()  {  var deliveries = db.Deliveries.Include(d => d.SepticTank);  return View(await deliveries.ToListAsync());  }  // GET: D3liv3r/Details/5  public async Task<ActionResult> Details(int? id)  {  if (id == null)  {  return new HttpStatusCodeResult(HttpStatusCode.BadRequest);  }  Delivery delivery = await db.Deliveries.Include(d =>d.SepticTank).FirstAsync(t =>t.Id == id);  if (delivery == null)  {  return HttpNotFound();  }  return View(delivery);  }  // GET: D3liv3r/Create  public ActionResult Create()  {  ViewBag.SepticTankId = new SelectList(db.SepticTanks, "Id", "Model");  return View();  }  // POST: D3liv3r/Create  // Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные  // сведения см. в статье http://go.microsoft.com/fwlink/?LinkId=317598.  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> Create([Bind(Include = "Id,BefMK,AftMK,SepticTankId")] Delivery delivery)  {  if (ModelState.IsValid)  {  db.Deliveries.Add(delivery);  await db.SaveChangesAsync();  return RedirectToAction("Index");  }  ViewBag.SepticTankId = new SelectList(db.SepticTanks, "Id", "Model", delivery.SepticTankId);  return View(delivery);  }  // GET: D3liv3r/Edit/5  public async Task<ActionResult> Edit(int? id)  {  if (id == null)  {  return new HttpStatusCodeResult(HttpStatusCode.BadRequest);  }  Delivery delivery = await db.Deliveries.FindAsync(id);  if (delivery == null)  {  return HttpNotFound();  }  ViewBag.SepticTankId = new SelectList(db.SepticTanks, "Id", "Model", delivery.SepticTankId);  return View(delivery);  }  // POST: D3liv3r/Edit/5  // Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные  // сведения см. в статье http://go.microsoft.com/fwlink/?LinkId=317598.  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> Edit([Bind(Include = "Id,BefMK,AftMK,SepticTankId")] Delivery delivery)  {  if (ModelState.IsValid)  {  db.Entry(delivery).State = EntityState.Modified;  await db.SaveChangesAsync();  return RedirectToAction("Index");  }  ViewBag.SepticTankId = new SelectList(db.SepticTanks, "Id", "Model", delivery.SepticTankId);  return View(delivery);  }  // GET: D3liv3r/Delete/5  public async Task<ActionResult> Delete(int? id)  {  if (id == null)  {  return new HttpStatusCodeResult(HttpStatusCode.BadRequest);  }  Delivery delivery = await db.Deliveries.FindAsync(id);  if (delivery == null)  {  return HttpNotFound();  }  return View(delivery);  }  // POST: D3liv3r/Delete/5  [HttpPost, ActionName("Delete")]  [ValidateAntiForgeryToken]  public async Task<ActionResult> DeleteConfirmed(int id)  {  Delivery delivery = await db.Deliveries.FindAsync(id);  db.Deliveries.Remove(delivery);  await db.SaveChangesAsync();  return RedirectToAction("Index");  }  protected override void Dispose(bool disposing)  {  if (disposing)  {  db.Dispose();  }  base.Dispose(disposing);  }  }  } |

Рис.35

Листинг контроллера "HomeController" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using ProjectTopas\_v\_1.\_2.Models;  using System.Data.Entity;  using System.Threading.Tasks;  using System.Net;  namespace ProjectTopas\_v\_1.\_2.Controllers  {  public class HomeController : Controller  {  SepticTankContext db = new SepticTankContext();  public async Task<ActionResult> Index()  {  var models = db.SepticSpecifications.Include(d => d.SepticTank).Where(d =>d.Id < 5);  return View(await models.ToListAsync());  }  public ActionResult Models()  {  IEnumerable<SepticTank> SepticTanks = db.SepticTanks;  ViewBag.SepticTanks = SepticTanks;  return View();  }  public ActionResult Installations()  {  var Installations = db.Installations.Include(p => p.SepticTank);  return View(Installations.ToList());  }  public ActionResult Details(int id)  {  SepticTank Sept = db.SepticTanks.Include(p => p.SepticSpecification).Include(p => p.Installation).FirstOrDefault(t => t.Id == id);  if (Sept == null)  {  return HttpNotFound();  }  return View(Sept);  }  public ActionResult SelectModel()  {  return View();  }  public ActionResult Services()  {  IEnumerable<SepticService> SepticServices = db.SepticServices;  ViewBag.SepticServices = SepticServices;  return View();  }  public ActionResult CreateRequest()  {  return PartialView("CreateRequest");  }  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> CreateRequest([Bind(Include = "Id,Name,Phone,Email,Comment")] Request request)  {  if (ModelState.IsValid)  {  db.Requests.Add(request);  await db.SaveChangesAsync();  return RedirectToAction("Index");  }  return View(request);  }  public async Task<ActionResult> Delivery()  {  var deliveries = db.Deliveries.Include(d => d.SepticTank);  return View(await deliveries.ToListAsync());  }  }  } |

Рис.36

Листинг контроллера "InstallController" следующий:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Data;  using System.Data.Entity;  using System.Linq;  using System.Threading.Tasks;  using System.Net;  using System.Web;  using System.Web.Mvc;  using ProjectTopas\_v\_1.\_2.Models;  namespace ProjectTopas\_v\_1.\_2.Controllers  {  [Authorize]  public class Insta1lController : Controller  {  private SepticTankContext db = new SepticTankContext();  // GET: Insta1l  public async Task<ActionResult> Index()  {  var installations = db.Installations.Include(i => i.SepticTank);  return View(await installations.ToListAsync());  }  // GET: Insta1l/Details/5  public async Task<ActionResult> Details(int? id)  {  if (id == null)  {  return new HttpStatusCodeResult(HttpStatusCode.BadRequest);  }  Installation installation = await db.Installations.Include(d => d.SepticTank).FirstAsync(t => t.Id == id);  if (installation == null)  {  return HttpNotFound();  }  return View(installation);  }  // GET: Insta1l/Create  public ActionResult Create()  {  ViewBag.SepticTankId = new SelectList(db.SepticTanks, "Id", "Model");  return View();  }  // POST: Insta1l/Create  // Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные  // сведения см. в статье http://go.microsoft.com/fwlink/?LinkId=317598.  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> Create([Bind(Include = "Id,CostShief,InstSand,InstLoam,InstClay,InstQuickSand,ExpenSand,SepticTankId")] Installation installation)  {  if (ModelState.IsValid)  {  db.Installations.Add(installation);  await db.SaveChangesAsync();  return RedirectToAction("Index");  }  ViewBag.SepticTankId = new SelectList(db.SepticTanks, "Id", "Model", installation.SepticTankId);  return View(installation);  }  // GET: Insta1l/Edit/5  public async Task<ActionResult> Edit(int? id)  {  if (id == null)  {  return new HttpStatusCodeResult(HttpStatusCode.BadRequest);  }  Installation installation = await db.Installations.FindAsync(id);  if (installation == null)  {  return HttpNotFound();  }  ViewBag.SepticTankId = new SelectList(db.SepticTanks, "Id", "Model", installation.SepticTankId);  return View(installation);  }  // POST: Insta1l/Edit/5  // Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку. Дополнительные  // сведения см. в статье http://go.microsoft.com/fwlink/?LinkId=317598.  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> Edit([Bind(Include = "Id,CostShief,InstSand,InstLoam,InstClay,InstQuickSand,ExpenSand,SepticTankId")] Installation installation)  {  if (ModelState.IsValid)  {  db.Entry(installation).State = EntityState.Modified;  await db.SaveChangesAsync();  return RedirectToAction("Index");  }  ViewBag.SepticTankId = new SelectList(db.SepticTanks, "Id", "Model", installation.SepticTankId);  return View(installation);  }  // GET: Insta1l/Delete/5  public async Task<ActionResult> Delete(int? id)  {  if (id == null)  {  return new HttpStatusCodeResult(HttpStatusCode.BadRequest);  }  Installation installation = await db.Installations.FindAsync(id);  if (installation == null)  {  return HttpNotFound();  }  return View(installation);  }  // POST: Insta1l/Delete/5  [HttpPost, ActionName("Delete")]  [ValidateAntiForgeryToken]  public async Task<ActionResult> DeleteConfirmed(int id)  {  Installation installation = await db.Installations.FindAsync(id);  db.Installations.Remove(installation);  await db.SaveChangesAsync();  return RedirectToAction("Index");  }  protected override void Dispose(bool disposing)  {  if (disposing)  {  db.Dispose();  }  base.Dispose(disposing);  }  }  } |

Рис.37

Листинг представления «Index» контроллера «HomeController» следующий:

|  |
| --- |
| @model IEnumerable<ProjectTopas\_v\_1.\_2.Models.SepticSpecification>  @{  ViewBag.Title = "Главная";  }  <h1 class="page-header text-center">Сайт по продаже и установки септиков ТОПАС.</h1>  <div class="row">  <p>Септик «Топас» является локальной биологической системой очистки стоков с предварительным осаждением крупных взвесей. Именно на таком принципе работают централизованные канализационные очистные сооружения, поэтому септик «Топас» условно можно назвать их миниатюрой.</p>  <p>На данном сайте вы можете просмотреть модельный ряд септиков, стоимость модели, монтажа, доставки и обслуживания.</p>  <div class="row backgr1">  <div class="col-md-6 col-md-offset-3 img-thumbnail backgr2 text-center">  <p class="">Чтобы обратиться к нам вы можете нажать на кнопку "Сделать запрос" или позвонить по данному телефону:</p>  <div class="col-md-5">@Html.ActionLink("Сделать запрос", "CreateRequest", new { }, new { @class = "compItem btn btn-lg btn-default btn-ill gl2" }) </div>  <div class="col-md-2"><h3 class="gl1">ИЛИ</h3></div>  <div class="col-md-5">  <h4 class="">8-915-014-02-93</h4>  <h4 class="">8-977-980-47-99</h4>  <h4 class="">8-903-672-61-80</h4>  </div>  </div>  </div>  </div>  <div class="row">  <h2 class="page-header text-center">Популярные модели</h2>  @foreach (var item in Model)  {  <div class="col-sm-4 col-md-3">  <div class="thumbnail">  <img class="img-responsive" src="~/Pics/Models/MainPic/@Html.DisplayFor(modelItem => item.PathPicMain)">  <div class="caption text-center">  <h4>Модель: @Html.DisplayFor(modelItem => item.SepticTank.Model)</h4>  <p>Цена: @Html.DisplayFor(modelItem => item.SepticTank.Cost1Comp) руб.</p>  <p>@Html.ActionLink("Подробнее", "Details", new { id = item.Id }, new { @class = "btn btn-default btn-group-justified" })</p>  </div>  </div>  </div>  }  </div>  <hr />  <div class="row">  <h1 class="page-header text-center">Преимущества автономной канализации "ТОПАС"</h1>  <p>  <ul>  <li>Высокое качество очистки стоков (до 98%);</li>  <li>Легкий и прочный компактный корпус, выполненный из высококачественного полипропилена и снабженный ребрами жесткости;</li>  <li>Простой монтаж в сжатые сроки, возможность монтажа в любое время года, в любых типах грунтов, даже при высоком уровне грунтовых вод;</li>  <li>Полная экологическая безопасность, отсутствие неприятного запаха;</li>  <li>Легкость обслуживания: нет необходимости в вызове ассенизационной машины, приобретении культур микроорганизмов и т.п.;</li>  <li>На выходе — чистая техническая вода и стабилизированный ил;</li>  <li>Надежность и долговечность, срок службы составляет более 50 лет.</li>  </ul>  </p>  <p><b>Установки «ТОПАС» — это правильное решение при обустройстве канализации на частной территории</b></p>  </div>  @section scripts  {  <script type="text/javascript">  $(function () {  $.ajaxSetup({ cache: false });  $(".compItem").click(function (e) {  e.preventDefault();  $.get(this.href, function (data) {  $('#dialogContent').html(data);  $('#modDialog').modal('show');  });  });  })  </script>  } |

Рис.38

Листинг представления «CreateRequest» контроллера «HomeController» следующий:

|  |
| --- |
| @model ProjectTopas\_v\_1.\_2.Models.Request  <div class="modal-content">  <div class="modal-header">  <button class="close" data-dismiss="modal" area-hidden="true">X</button>  <h4>Создание запроса</h4>  </div>    @using (Html.BeginForm())  {  @Html.AntiForgeryToken()  <div class="modal-body">  <div class="form-horizontal">  <h4>При заполнении запроса убедитесь, что все поля заполнены правильно</h4>  <hr />  @Html.ValidationSummary(true, "", new { @class = "text-danger" })  <div class="form-group">  @Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  @Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })  @Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Phone, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  @Html.EditorFor(model => model.Phone, new { htmlAttributes = new { @class = "form-control" , @placeholder = "+7(\_\_\_)\_\_\_-\_\_-\_\_" } })  @Html.ValidationMessageFor(model => model.Phone, "", new { @class = "text-danger" })  </div>  </div>  <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", @placeholder = "Example@gmail.com" } })  @Html.ValidationMessageFor(model => model.Email, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Comment, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  @Html.EditorFor(model => model.Comment, new { htmlAttributes = new { @class = "form-control" } })  <p class="help-block">Здесь Вы можете записать что вы хотите купить</p>  @Html.ValidationMessageFor(model => model.Comment, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  <div class="col-md-offset-2 col-md-10">  <input type="submit" value="Отправить" class="btn btn-default" />  </div>  </div>  </div>  </div>  }  </div> |

Рис.38

Листинг представления «Delivery» контроллера «HomeController» следующий:

|  |
| --- |
| @model IEnumerable<ProjectTopas\_v\_1.\_2.Models.Delivery>  @{  ViewBag.Title = "Доставка септика";  Layout = "~/Views/Shared/\_Layout.cshtml";  }  <h1 class="page-header text-center">Доставка септиков ТОПАС</h1>  <div class="row">  <div class="col-md-6">  <p>  Есть возможность самовывоза септика или заказать доставку.<br />  Доставка осуществляется в данные области:  <br />  <ul>  <li>Серпухов, Серпуховский район</li>  <li>Чехов, Чеховский район</li>  <li>Калужская облать(Жуковский район, Тарусский район)</li>  <li>Тульская облать(Заокский район)</li>  <li>Ступинский район</li>  <li>Наро-Фоминский район</li>  </ul>  <p>Для более детальной информации вы можете обратиться по телефону:8-496-7</p>  </div>  </div>  <h2 class="page-header text-center">Цены на доставку септиков</h2>  <div class="row">  <table class="table table-bordered table-hover text-center">  <thead>  <tr>  <th class="text-center">  @Html.DisplayNameFor(model => model.SepticTank.Model)  </th>  <th class="text-center">  @Html.DisplayNameFor(model => model.BefMK)  </th>  <th class="text-center">  @Html.DisplayNameFor(model => model.AftMK)  </th>  </tr>  </thead>  @foreach (var item in Model)  {  <tr>  <td>  @Html.DisplayFor(modelItem => item.SepticTank.Model)  </td>  <td>  @Html.DisplayFor(modelItem => item.BefMK) руб.  </td>  <td>  + @Html.DisplayFor(modelItem => item.AftMK) руб. за 1 км.  </td>  </tr>  }  </table>  </div>  @section scripts  {  <script type="text/javascript">  $(function () {  $.ajaxSetup({ cache: false });  $(".compItem").click(function (e) {  e.preventDefault();  $.get(this.href, function (data) {  $('#dialogContent').html(data);  $('#modDialog').modal('show');  });  });  })  </script>  } |

Рис.39

Листинг представления «Details» контроллера «HomeController» следующий:

|  |
| --- |
| @using ProjectTopas\_v\_1.\_2.Models  @model SepticTank  @{  ViewBag.Title = "Детальная информация";  Layout = "~/Views/Shared/\_Layout.cshtml";  }  <h1 class="page-header">Подробная информация о модели @Html.DisplayFor(model => model.Model)</h1>  @foreach (SepticSpecification S in Model.SepticSpecification)  {  foreach (Installation I in Model.Installation)  {  <div class="row">  <div class="col-xs-6 col-md-3 ">  <img src="~/Pics/Models/MainPic/@S.PathPicMain" class="img-thumbnail" />  </div>    <div class="col-xs-12 col-md-9">  <ul class="nav nav-tabs">  <li class="active"><a href="#info" data-toggle="tab">Подробная информация</a></li>  <li><a href="#cost" data-toggle="tab">Цены</a></li>  <li><a href="#Scheme" data-toggle="tab">Монтажная схема</a></li>  </ul>  <div class="tab-content">  <div class="tab-pane fade in active" id="info">  <table class="table table-bordered">  <thead>  <tr>  <th>Спецификация</th>  <th>Описание</th>  </tr>  </thead>  <tr>  <th>Количество пользователей</th>  <th>@Html.DisplayFor(model => model.CountUsers)</th>  </tr>  <tr>  <th>Вес</th>  <th>@S.Weight кг</th>  </tr>  <tr>  <th>Залповый сброс</th>  <th>@S.VolleyRes л</th>  </tr>  <tr>  <th>Глубина залегания трубы</th>  <th>@S.PipeDepth см</th>  </tr>  <tr>  <th>Энергопотребление</th>  <th>@I.SepticTank.Energy КВт/сутки</th>  </tr>  <tr>  <th>Максимально дневной приток</th>  <th>@I.SepticTank.InflowDrain м<sup>3</sup>/сутки</th>  </tr>  </table>  </div>  <div class="tab-pane fade" id="montage">  <p>Цены на монтаж</p>  </div>  <div class="tab-pane fade" id="Scheme">  <img src="~/Pics/Models/Scheme/@S.PathMontagePic" class="img-responsive" />  </div>  <div class="tab-pane fade" id="cost">  <table class="table table-bordered text-center">  <thead>  <tr>  <th colspan="2">Модель</th>  <th>Цена, руб.</th>  </tr>  </thead>  <tr>  <th colspan="2">1 компрессор</th>  <th>  @if (Model.Cost1Comp == 0)  {  }  else  {  @Html.DisplayFor(model => model.Cost1Comp)  }  </th>  </tr>  <tr>  <th colspan="2">2 компрессора</th>  <th>@Html.DisplayFor(model => model.Cost2Comp)</th>  </tr>  <tr>  <th colspan="2">Шеф-монтаж</th>  <th>@I.CostShief</th>  </tr>  <tr>  <th rowspan="4">Стандартный монтаж в зависимости от типа грунта</th>  <th>Песчаный</th>  <th>@I.InstSand</th>  </tr>  <tr>  <th>Суглинок</th>  <th>@I.InstLoam</th>  </tr>  <tr>  <th>Глина</th>  <th>@I.InstClay</th>  </tr>  <tr>  <th>Плывун</th>  <th>@I.InstQuickSand</th>  </tr>  </table>  </div>  </div>  </div>  </div>  }  }  <div class="row">  <h1 class="page-header text-center">Особенности автономной канализации "ТОПАС"</h1>  <p>  <ul>  <li>Высокое качество очистки стоков (до 98%);</li>  <li>Легкий и прочный компактный корпус, выполненный из высококачественного полипропилена и снабженный ребрами жесткости;</li>  <li>Простой монтаж в сжатые сроки, возможность монтажа в любое время года, в любых типах грунтов, даже при высоком уровне грунтовых вод;</li>  <li>Полная экологическая безопасность, отсутствие неприятного запаха;</li>  <li>Легкость обслуживания: нет необходимости в вызове ассенизационной машины, приобретении культур микроорганизмов и т.п.;</li>  <li>На выходе — чистая техническая вода и стабилизированный ил;</li>  <li>Надежность и долговечность, срок службы составляет более 50 лет.</li>  </ul>  </p>  <p><b>Установки «ТОПАС» — это правильное решение при обустройстве канализации на частной территории</b></p>  </div>  @section scripts  {  <script type="text/javascript">  $(function () {  $.ajaxSetup({ cache: false });  $(".compItem").click(function (e) {  e.preventDefault();  $.get(this.href, function (data) {  $('#dialogContent').html(data);  $('#modDialog').modal('show');  });  });  })  </script>  } |

Рис.40

Листинг представления «Installations» контроллера «HomeController» следующий:

|  |
| --- |
| @model IEnumerable<ProjectTopas\_v\_1.\_2.Models.Installation>  @{  ViewBag.Title = "Цены на монтаж";  Layout = "~/Views/Shared/\_Layout.cshtml";  }  <h1 class="page-header text-center">Монтаж септика за 1 день!</h1>  <div class="row">  <div class="col-md-6">  <p>  <b>Монтаж</b> - это работы, связанные с установкой и пуско-наладочными работами на участке заказчика, которые требуют квалифицированных знаний.  </p>  <p>  <i><b>Монтаж</b> включает в себя:</i><br />  <ul>  <li>Копание котлована;</li>  <li>В случае, если монтажные работы проводятся на глубине более 2 м или на участке песчаный грунт или плывун, согласно нормам техники безопасности в котловане проводится установка опалубки;</li>  <li>Осуществляется прокладка труб и дальнейшее их присоединение к системе;</li>  <li>Монтаж врезного патрубка в приемную камеру;</li>  <li>Прокладка и подключение эл. кабеля к блоку управления;</li>  <li>Монтаж компрессоров;</li>  <li>Тестирование очистного сооружения.</li>  </ul>  </p>  </div>  </div>  <div class="row">  <h3 class="text-center">Цены на монтажные работы по установке канализации Топас</h3>  <table class="table table-bordered text-center table-hover">  <thead>  <tr>  <th rowspan="2">Модель АС</th>  <th rowspan="2">Шеф-монтаж</th>  <th colspan="4">Стандартный монтаж в зависимости от типа грунта</th>  <th rowspan="2">Ориентировочный расход<br /> песка для обсыпки АС, м<sup>3</sup></th>  </tr>  <tr>  <th>Песчаный</th>  <th>Суглинок</th>  <th>Глина</th>  <th>Плывун</th>  </tr>  </thead>  <tbody>  @foreach (var item in Model)  {  <tr>  <td><p>@Html.DisplayFor(modelItem => item.SepticTank.Model)</p></td>  <td><p>@Html.DisplayFor(modelItem => item.CostShief)</p></td>  <td><p>@Html.DisplayFor(modelItem => item.InstSand)</p></td>  <td><p>@Html.DisplayFor(modelItem => item.InstLoam)</p></td>  <td><p>@Html.DisplayFor(modelItem => item.InstClay)</p></td>  <td><p>@Html.DisplayFor(modelItem => item.InstQuickSand)</p></td>  <td style="text-align:center"><p>@Html.DisplayFor(modelItem => item.ExpenSand)</p></td>  </tr>  }  </tbody>  </table>  </div>  <div class="row">  <div class="col-md-10 col-md-offset-1">  <p>В цену стандартного монтажа включена подготовка котлована<sup>\*</sup> под АС ТОПАС и под канализационную трубу ДУ 110, и при необходимости</p>  <p><sup>\*</sup> - Не включены работы по подготовке и укладке бетонной плиты на дно котлована для моделей АС ТОПАС: 30, 40, 50, 75, 100, 150 и их модификаций</p>  </div>  </div>  @section scripts  {  <script type="text/javascript">  $(function () {  $.ajaxSetup({ cache: false });  $(".compItem").click(function (e) {  e.preventDefault();  $.get(this.href, function (data) {  $('#dialogContent').html(data);  $('#modDialog').modal('show');  });  });  })  </script>  } |

Рис.41

Листинг представления «Models» контроллера «HomeController» следующий:

|  |
| --- |
| @model ProjectTopas\_v\_1.\_2.Models.SepticTank  @{  ViewBag.Title = "Модельный ряд";  Layout = "~/Views/Shared/\_Layout.cshtml";  }  <div id="modDialog" class="modal fade">  <div id="dialogContent" class="modal-dialog"></div>  </div>  <div class="row">  <h1 class="page-header text-center">Модельный ряд септиков ТОПАС</h1>  <p>На данной странице вы можете узнать цену и некоторые характеристики септика, которые вам необходимы. Также есть возможность просмотра детальной информации конкретной модели, нажав на название модели.</p>  </div>  <div class="row">  <h3 class="text-center">Цены на септики ТОПАС с установкой под ключ</h3>  <p>Вы можете просмотреть детальную информацию о септике, нажав на его название</p>  <table class="table table-bordered table-hover">  <thead>  <tr>  <th rowspan="2">Модель</th>  <th colspan="2" style="text-align:center"><p>Цена, руб.</p></th>  <th rowspan="2">Количество <br />пользователей, чел.</th>  <th rowspan="2">Максимально дневной<br /> приток, м<sup>3</sup>/сутки</th>  <th rowspan="2">Энергия,<br />Квт/сутки</th>  <th colspan="3" style="text-align:center">Габаритные размеры</th>  </tr>  <tr>  <th>1 компрессор</th>  <th>2 компрессора</th>  <th>Длина, м</th>  <th>Ширина, м</th>  <th>Высота, м</th>  </tr>  </thead>  <tbody>  @foreach (var b in ViewBag.SepticTanks)  {  <tr>  <td style="text-align:center"><a href="/Home/Details/@b.Id">@b.Model</a></td>  <td style="text-align:center">  <p>  @if (b.Cost1Comp == 0)  { }  else  {  @b.Cost1Comp  }  </p>  </td>  <td style="text-align:center">  <p>  @if (b.Cost2Comp == 0)  { }  else  {  @b.Cost2Comp  }  </p>  </td>  <td style="text-align:center"><p>@b.CountUsers</p></td>  <td style="text-align:center"><p>@b.InflowDrain</p></td>  <td style="text-align:center"><p>@b.Energy</p></td>  <td style="text-align:center"><p>@b.Lenght</p></td>  <td style="text-align:center"><p>@b.Weight</p></td>  <td style="text-align:center"><p>@b.Height</p></td>  </tr>  }  </tbody>  </table>  <br />  </div>  @section scripts  {  <script type="text/javascript">  $(function () {  $.ajaxSetup({ cache: false });  $(".compItem").click(function (e) {  e.preventDefault();  $.get(this.href, function (data) {  $('#dialogContent').html(data);  $('#modDialog').modal('show');  });  });  })  </script>  } |

Рис.42

Листинг представления «Services» контроллера «HomeController» следующий:

|  |
| --- |
| @model ProjectTopas\_v\_1.\_2.Models.SepticService  @{  ViewBag.Title = "Обслуживание";  Layout = "~/Views/Shared/\_Layout.cshtml";  }  <h1 class="page-header text-center">Обслуживание и ремонт септиков Топас</h1>  <div class="row">  <p>Своевременное обслуживание и ремонт вашего септика или автономной конализации позволят продлить срок эксплуатации на годы!</p>  <p>Как и любые другие технически-сложные устройства, очистные сооружения ТОПАС нуждаются в уходе и регулярной очистке. В результате работы автономной станции образуются ил, который следует удалять 3-4 раза в год для нормального функционирования системы.</p>  <p>Звоните сейчас! <b>8-915-014-02-93</b> | <b>8-903-672-61-80</b></p>  <hr />  <h2 class="page-header text-center">Аварийные случаи, возможные при обслуживании септиков топас</h2>  <div class="row">  <div class="col-md-3">  <img class="img-thumbnail thumb" src="~/Pics/Other/Serv1.jpg" />  </div>  <div class="col-md-3">  <img class="img-thumbnail thumb" src="~/Pics/Other/Serv2.jpg" />  </div>  <div class="col-md-3">  <img class="img-thumbnail thumb" src="~/Pics/Other/Serv3.jpg" />  </div>  <div class="col-md-3">  <img class="img-thumbnail thumb" src="~/Pics/Other/Serv4.jpg" />  </div>  </div>  </div>  <hr />  <div class="row">  <h1 class="page-header text-center">Обслужите ваш септик вовремя!</h1>  @foreach (var b in ViewBag.SepticServices)  {  <div class="col-sm-6 col-md-4">  <div class="thumbnail text-justify">  <h4>Обслуживание под ключ @b.Models</h4>  <hr />  <h3 class="text-center">@b.Cost руб.</h3>  </div>  </div>  }  <div class="col-md-4">  <p>В услугу входит:</p>  <br />  <ul>  <li>Стандартное облсуживание</li>  <li>Полная диагностика</li>  <li>Рекомендации по эксплуатации</li>  </ul>    </div>  </div>  @section scripts  {  <script type="text/javascript">  $(function () {  $.ajaxSetup({ cache: false });  $(".compItem").click(function (e) {  e.preventDefault();  $.get(this.href, function (data) {  $('#dialogContent').html(data);  $('#modDialog').modal('show');  });  });  })  </script>  } |

Рис.43

Листинг мастер-страницы «\_Layout» следующий:

|  |
| --- |
| <!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">  <title>Индивидуальная канализация ТОПАС в Серпухове - @ViewBag.Title</title>  @Styles.Render("~/Content/css")  @Scripts.Render("~/bundles/modernizr")  <link href='@Url.Content("~/Content/themes/base/jquery-ui.css")' rel="stylesheet" type="text/css" />  </head>  <body>  <div id="modDialog" class="modal fade">  <div id="dialogContent" class="modal-dialog"></div>  </div>  <div class="navbar navbar-inverse navbar-fixed-top">  <div class="container">  <div class="navbar-header">  <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">  <span class="icon-bar"></span>  <span class="icon-bar"></span>  <span class="icon-bar"></span>  </button>  @Html.ActionLink("Топас-Серпухов.ru", "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })  </div>  <div class="navbar-collapse collapse">  <ul class="nav navbar-nav">  <li>@Html.ActionLink("Главная", "Index", "Home")</li>  <li>@Html.ActionLink("Модельный ряд", "Models", "Home")</li>  <li>  <a href="#" class="dropdown-toggle" data-toggle="dropdown" role="button" aria-haspopup="true" aria-expanded="false">Услуги<span class="caret"></span></a>  <ul class="dropdown-menu">  <li></li>  <li>@Html.ActionLink("Монтаж", "Installations", "Home")</li>  <li>@Html.ActionLink("Доставка", "Delivery", "Home")</li>  <li>@Html.ActionLink("Обслуживание и ремонт", "Services", "Home")</li>  </ul>  </li>  </ul>  <ul class="nav navbar-nav navbar-right">  <li>  @Html.ActionLink("Сделать запрос", "CreateRequest", new { }, new { @class = "compItem btn btn-lg btn-default btn-ill" })  </li>  </ul>  </div>  </div>  </div>  <div class="container body-content">  @RenderBody()  <hr />  </div>  <footer class="col-lg-12">  <div class="container">  <p>topas-serpukhov.ru &copy; @DateTime.Now.Year Контактная информация: Телефон: <b>8-915-014-02-93</b> | <b>8-977-980-47-99</b> | <b>8-903-672-61-80</b></p>  </div>  </footer>  @Scripts.Render("~/bundles/jquery")  @Scripts.Render("~/bundles/bootstrap")  @RenderSection("scripts", required: false)    </body>  </html> |

Рис.44

Листинг мастер-страницы «\_Layout2» следующий:

|  |
| --- |
| <!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">  <title>@ViewBag.Title - Административная страница</title>  @Styles.Render("~/Content/css")  @Scripts.Render("~/bundles/modernizr")  <style id="holderjs-style" type="text/css" class="--apng-checked"></style>  </head>  <body>  <div class="navbar navbar-inverse navbar-fixed-top">  <div class="container">  <div class="navbar-header">  <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">  <span class="icon-bar"></span>  <span class="icon-bar"></span>  <span class="icon-bar"></span>  </button>  @Html.ActionLink("Топас-Серпухов.ru", "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })  </div>  <div class="navbar-collapse collapse">  <ul class="nav navbar-nav">  <li>@Html.ActionLink("Главная", "Index", "Home")</li>  <li>@Html.ActionLink("Модельный ряд", "Models", "Home")</li>  <li>  <a href="#" class="dropdown-toggle" data-toggle="dropdown" role="button" aria-haspopup="true" aria-expanded="false">Услуги<span class="caret"></span></a>  <ul class="dropdown-menu">  <li></li>  <li>@Html.ActionLink("Монтаж", "Installations", "Home")</li>  <li>@Html.ActionLink("Доставка", "Delivery", "Home")</li>  <li>@Html.ActionLink("Обслуживание и ремонт", "Services", "Home")</li>  </ul>  </li>  </ul>  @Html.Partial("\_LoginPartial")  </div>  </div>  </div>  <div class="container-fluid">  <div class="row">  <div class="col-sm-3 col-md-2 sidebar">  <ul class="nav nav-sidebar">  <li>@Html.ActionLink("Септики","Index", "S3pticTnk3dit")</li>  <li>@Html.ActionLink("Подробная информация о септиках", "Index", "S3pticSp3c")</li>  <li>@Html.ActionLink("Монтаж","Index", "Insta1l")</li>  <li>@Html.ActionLink("Обслуживание и ремонт", "Index", "S3pt1cS3rv")</li>  <li>@Html.ActionLink("Доставка", "Index", "D3liv3r")</li>  <li>@Html.ActionLink("Запросы", "Index", "R3qu3st")</li>    </ul>    </div>  <div class="col-sm-9 col-sm-offset-3 col-md-10 col-md-offset-2 main">  @RenderBody()  <hr />  <footer>  <p>topas-serpukhov.ru &copy; @DateTime.Now.Year Телефон: <b>8-915-014-02-93</b> | <b>8-977-980-47-99</b> | <b>8-903-672-61-80</b></p>  </footer>  </div>  </div>  </div>  <!-- Скрипт плавного открытия и закрытия блока -->  @Scripts.Render("~/bundles/jquery")  @Scripts.Render("~/bundles/bootstrap")  @RenderSection("scripts", required: false)  </body>  </html> |

Рис.45

Листинг представления «Index» контроллера «SepticTankEditController» следующий:

|  |
| --- |
| @model IEnumerable<ProjectTopas\_v\_1.\_2.Models.SepticTank>  @{  ViewBag.Title = "Модель";  Layout = "~/Views/Shared/\_Layout2.cshtml";  }  <h2>Страница изменений данных в таблице "Модели"</h2>  <p>  @Html.ActionLink("Создать новую запись", "Create")  </p>  <table class="table table-responsive">  <tr>  <th>  @Html.DisplayNameFor(model => model.Model)  </th>  <th>  @Html.DisplayNameFor(model => model.Cost1Comp)  </th>  <th>  @Html.DisplayNameFor(model => model.Cost2Comp)  </th>  <th>  @Html.DisplayNameFor(model => model.Energy)  </th>  <th>  @Html.DisplayNameFor(model => model.Lenght)  </th>  <th>  @Html.DisplayNameFor(model => model.Weight)  </th>  <th>  @Html.DisplayNameFor(model => model.Height)  </th>  <th></th>  </tr>  @foreach (var item in Model) {  <tr>  <td>  @Html.DisplayFor(modelItem => item.Model)  </td>  <td>  @Html.DisplayFor(modelItem => item.Cost1Comp) руб.  </td>  <td>  @Html.DisplayFor(modelItem => item.Cost2Comp) руб.  </td>  <td>  @Html.DisplayFor(modelItem => item.Energy) Квт/сутки  </td>  <td>  @Html.DisplayFor(modelItem => item.Lenght) м  </td>  <td>  @Html.DisplayFor(modelItem => item.Weight) м  </td>  <td>  @Html.DisplayFor(modelItem => item.Height) м  </td>  <td>  @Html.ActionLink("Изменить", "Edit", new { id=item.Id }) |  @Html.ActionLink("Детали", "Details", new { id=item.Id }) |  @Html.ActionLink("Удалить", "Delete", new { id=item.Id })  </td>  </tr>  }  </table> |

Рис.46

Листинг представления «Create» контроллера «SepticTankEditController» следующий:

|  |
| --- |
| @model ProjectTopas\_v\_1.\_2.Models.SepticTank  @{  ViewBag.Title = "Создание записи";  Layout = "~/Views/Shared/\_Layout2.cshtml";  }  <h2>Создание записи</h2>  @using (Html.BeginForm())  {  @Html.AntiForgeryToken()    <div class="form-horizontal">  <h4>Модели</h4>  <hr />  @Html.ValidationSummary(true, "", new { @class = "text-danger" })  <div class="form-group">  @Html.LabelFor(model => model.Model, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  @Html.EditorFor(model => model.Model, new { htmlAttributes = new { @class = "form-control" } })  @Html.ValidationMessageFor(model => model.Model, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Cost1Comp, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Cost1Comp, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">руб.</span>  </div>  <p class="help-block">Цена за модель с одним компрессором</p>  @Html.ValidationMessageFor(model => model.Cost1Comp, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Cost2Comp, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Cost2Comp, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">руб.</span>  </div>  <p class="help-block">Цена за модель с двумя компрессорами</p>  @Html.ValidationMessageFor(model => model.Cost2Comp, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.CountUsers, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  @Html.EditorFor(model => model.CountUsers, new { htmlAttributes = new { @class = "form-control" } })  @Html.ValidationMessageFor(model => model.CountUsers, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.InflowDrain, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.InflowDrain, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">М<sup>3</sup>/сутки</span>  </div>  @Html.ValidationMessageFor(model => model.InflowDrain, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Energy, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Energy, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">КВт/сутки</span>  </div>  @Html.ValidationMessageFor(model => model.Energy, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Lenght, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Lenght, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">м</span>  </div>  @Html.ValidationMessageFor(model => model.Lenght, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Weight, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Weight, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">м</span>  </div>  @Html.ValidationMessageFor(model => model.Weight, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Height, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Height, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">м</span>  </div>  @Html.ValidationMessageFor(model => model.Height, "", 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>  }  <div>  @Html.ActionLink("Вернуться к списку", "Index")  </div> |

Рис.47

Листинг представления «Delete» контроллера «SepticTankEditController» следующий:

|  |
| --- |
| @model ProjectTopas\_v\_1.\_2.Models.SepticTank  @{  ViewBag.Title = "Удаление записи";  Layout = "~/Views/Shared/\_Layout2.cshtml";  }  <h2>Удаление записи</h2>  <h3>Вы действительно хотите удалить эту запись?</h3>  <div>  <h4>Модели</h4>  <hr />  <dl class="dl-horizontal">  <dt>  @Html.DisplayNameFor(model => model.Model)  </dt>  <dd>  @Html.DisplayFor(model => model.Model)  </dd>  <dt>  @Html.DisplayNameFor(model => model.Cost1Comp)  </dt>  <dd>  @Html.DisplayFor(model => model.Cost1Comp)  </dd>  <dt>  @Html.DisplayNameFor(model => model.Cost2Comp)  </dt>  <dd>  @Html.DisplayFor(model => model.Cost2Comp)  </dd>  <dt>  @Html.DisplayNameFor(model => model.CountUsers)  </dt>  <dd>  @Html.DisplayFor(model => model.CountUsers)  </dd>  <dt>  @Html.DisplayNameFor(model => model.InflowDrain)  </dt>  <dd>  @Html.DisplayFor(model => model.InflowDrain)  </dd>  <dt>  @Html.DisplayNameFor(model => model.Energy)  </dt>  <dd>  @Html.DisplayFor(model => model.Energy)  </dd>  <dt>  @Html.DisplayNameFor(model => model.Lenght)  </dt>  <dd>  @Html.DisplayFor(model => model.Lenght)  </dd>  <dt>  @Html.DisplayNameFor(model => model.Weight)  </dt>  <dd>  @Html.DisplayFor(model => model.Weight)  </dd>  <dt>  @Html.DisplayNameFor(model => model.Height)  </dt>  <dd>  @Html.DisplayFor(model => model.Height)  </dd>  </dl>  @using (Html.BeginForm()) {  @Html.AntiForgeryToken()  <div class="form-actions no-color">  <input type="submit" value="Удалить" class="btn btn-default" /> |  @Html.ActionLink("Вернуться к списку", "Index")  </div>  }  </div> |

Рис.48

Листинг представления «Details» контроллера «SepticTankEditController» следующий:

|  |
| --- |
| @model ProjectTopas\_v\_1.\_2.Models.SepticTank  @{  ViewBag.Title = "Детальная информация";  Layout = "~/Views/Shared/\_Layout2.cshtml";  }  <h2>Детальная информация</h2>  <div>  <h4>Модели</h4>  <hr />  <dl class="dl-horizontal">  <dt>  @Html.DisplayNameFor(model => model.Model)  </dt>  <dd>  @Html.DisplayFor(model => model.Model)  </dd>  <dt>  @Html.DisplayNameFor(model => model.Cost1Comp)  </dt>  <dd>  @Html.DisplayFor(model => model.Cost1Comp) руб.  </dd>  <dt>  @Html.DisplayNameFor(model => model.Cost2Comp)  </dt>  <dd>  @Html.DisplayFor(model => model.Cost2Comp) руб.  </dd>  <dt>  @Html.DisplayNameFor(model => model.CountUsers)  </dt>  <dd>  @Html.DisplayFor(model => model.CountUsers)  </dd>  <dt>  @Html.DisplayNameFor(model => model.InflowDrain)  </dt>  <dd>  @Html.DisplayFor(model => model.InflowDrain) м<sup>3</sup>/сутки  </dd>  <dt>  @Html.DisplayNameFor(model => model.Energy)  </dt>  <dd>  @Html.DisplayFor(model => model.Energy) Квт/сутки  </dd>  <dt>  @Html.DisplayNameFor(model => model.Lenght)  </dt>  <dd>  @Html.DisplayFor(model => model.Lenght) м  </dd>  <dt>  @Html.DisplayNameFor(model => model.Weight)  </dt>  <dd>  @Html.DisplayFor(model => model.Weight) м  </dd>  <dt>  @Html.DisplayNameFor(model => model.Height)  </dt>  <dd>  @Html.DisplayFor(model => model.Height) м  </dd>  </dl>  </div>  <p>  @Html.ActionLink("Изменить", "Edit", new { id = Model.Id }) |  @Html.ActionLink("Вернуться к списку", "Index")  </p> |

Рис.49

Листинг представления «Edit» контроллера «SepticTankEditController» следующий:

|  |
| --- |
| @model ProjectTopas\_v\_1.\_2.Models.SepticTank  @{  ViewBag.Title = "Изменение записи";  Layout = "~/Views/Shared/\_Layout2.cshtml";  }  <h2>Изменение записи</h2>  @using (Html.BeginForm())  {  @Html.AntiForgeryToken()    <div class="form-horizontal">  <h4>Модели</h4>  <hr />  @Html.ValidationSummary(true, "", new { @class = "text-danger" })  @Html.HiddenFor(model => model.Id)  <div class="form-group">  @Html.LabelFor(model => model.Model, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  @Html.EditorFor(model => model.Model, new { htmlAttributes = new { @class = "form-control" } })  @Html.ValidationMessageFor(model => model.Model, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Cost1Comp, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Cost1Comp, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">руб.</span>  </div>  <p class="help-block">Цена за модель с одним компрессором</p>  @Html.ValidationMessageFor(model => model.Cost1Comp, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Cost2Comp, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Cost2Comp, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">руб.</span>  </div>  <p class="help-block">Цена за модель с двумя компрессорами</p>  @Html.ValidationMessageFor(model => model.Cost2Comp, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.CountUsers, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  @Html.EditorFor(model => model.CountUsers, new { htmlAttributes = new { @class = "form-control" } })  @Html.ValidationMessageFor(model => model.CountUsers, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.InflowDrain, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.InflowDrain, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">М<sup>3</sup>/сутки</span>  </div>  @Html.ValidationMessageFor(model => model.InflowDrain, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Energy, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Energy, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">КВт/сутки</span>  </div>  @Html.ValidationMessageFor(model => model.Energy, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Lenght, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Lenght, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">м</span>  </div>  @Html.ValidationMessageFor(model => model.Lenght, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Weight, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Weight, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">м</span>  </div>  @Html.ValidationMessageFor(model => model.Weight, "", new { @class = "text-danger" })  </div>  </div>  <div class="form-group">  @Html.LabelFor(model => model.Height, htmlAttributes: new { @class = "control-label col-md-2" })  <div class="col-md-10">  <div class="input-group">  @Html.EditorFor(model => model.Height, new { htmlAttributes = new { @class = "form-control" } })  <span class="input-group-addon">м</span>  </div>  @Html.ValidationMessageFor(model => model.Height, "", 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>  @Html.ActionLink("Вернуться к списку", "Index")  </div> |

Рис.50

* 1. Разработка тестовых данных

Тестирование сайта — процесс исследования, испытания [программного продукта](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), имеющий две различные цели:

* продемонстрировать разработчикам и заказчикам, что сайт соответствует требованиям;
* выявить ситуации, в которых поведение сайта является неправильным, нежелательным или не соответствующим спецификации.

Тестирование страницы «Авторизация»

Таблица 8 – Тестирование страницы «Авторизация»

|  |  |
| --- | --- |
| Наименование | Описание |
| Test Case # | TC\_UI\_1 |
| Приоритет теста | Высокий |
| Название тестирования/Имя | Проверка страницы авторизация при вводе неправильного пароля |
| Резюме испытания | Проверка входа на сайт |
| Шаги тестирования | 1. Ввести логин 2. Ввести пароль 3. Нажать на кнопку Вход |
| Данные тестирования | Логин=Admin  Пароль=1234567 |

Продолжение таблицы 8

|  |  |
| --- | --- |
| Наименование | Описание |
| Ожидаемый результат | Сообщение об ошибке «Неправильное имя пользователя/пароль» |
| Фактический результат | Сайт выдал сообщение об ошибке «Неправильное имя пользователя/пароль» |
| Предпосылки | - |
| Постусловия | Запрет на авторизвацию на сайте |
| Статус  (Pass/Fail) | Pass |
| Комментарии | - |

Тестирование страницы «Авторизация»

Таблица 9 – Тестирование страницы «Авторизация»

|  |  |
| --- | --- |
| Наименование | Описание |
| Test Case # | TC\_UI\_2 |
| Приоритеттеста | Высокий |
| Название тестирования/Имя | Проверка страницы авторизация при вводе несуществующего логина |
| Резюме испытания | Проверка входа на сайт |
| Шаги тестирования | 1. Ввести логин 2. Ввести пароль 3. Нажать на кнопку Вход |
| Данные тестирования | Логин=Student  Пароль=Admin |
| Ожидаемый результат | Сообщение об ошибке «Неправильное имя пользователя/пароль» |
| Фактический результат | Сайт выдал сообщение об ошибке «Неправильное имя пользователя/пароль» |
| Предпосылки | - |
| Постусловия | Запрет на вход в приложение |

Продолжение таблицы 9

|  |  |
| --- | --- |
| Наименование | Описание |
| Статус  (Pass/Fail) | Pass |
| Комментарии | - |

Тестирование страницы «Авторизация»

Таблица 10 – Тестирование страницы «Авторизация»

|  |  |
| --- | --- |
| Наименование | Описание |
| Test Case # | TC\_UI\_3 |
| Приоритет теста | Высокий |
| Название тестирования/Имя | Проверка страницы авторизация с правильным логином и паролем |
| Резюме испытания | Проверка входа на сайт |
| Шаги тестирования | 1. Ввести логин 2. Ввести пароль 3. Нажать на кнопку Вход |
| Данные тестирования | Логин= glss2fa@gmail.com  Пароль= quNX9fBR |
| Ожидаемый результат | Вход администратора на сайт |
| Фактический результат | Вход администратора на сайт |
| Предпосылки | - |
| Постусловия | Отображение страницы администрирования |
| Статус  (Pass/Fail) | Pass |
| Комментарии | - |

* 1. Результаты работы сайта

Вывод ошибки при неправильном вводе логина или пароля представлен на рисунке 51.

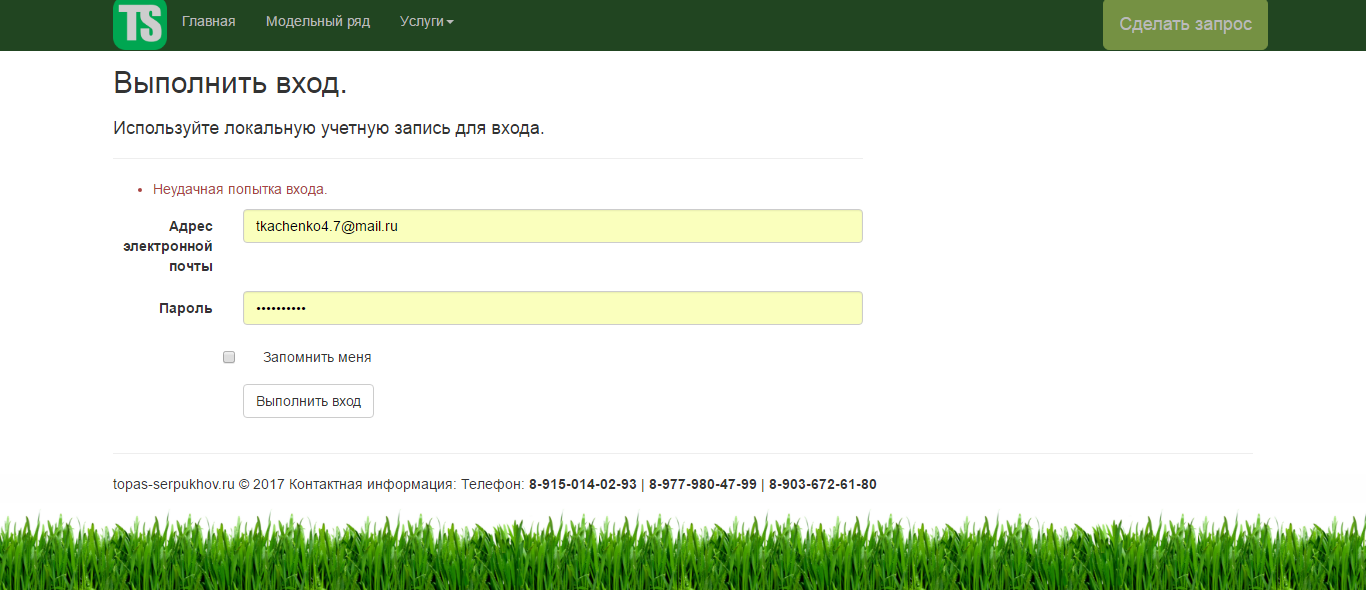


Рис. 51

Просмотр данных для редактирования представлен на рисунке 52.

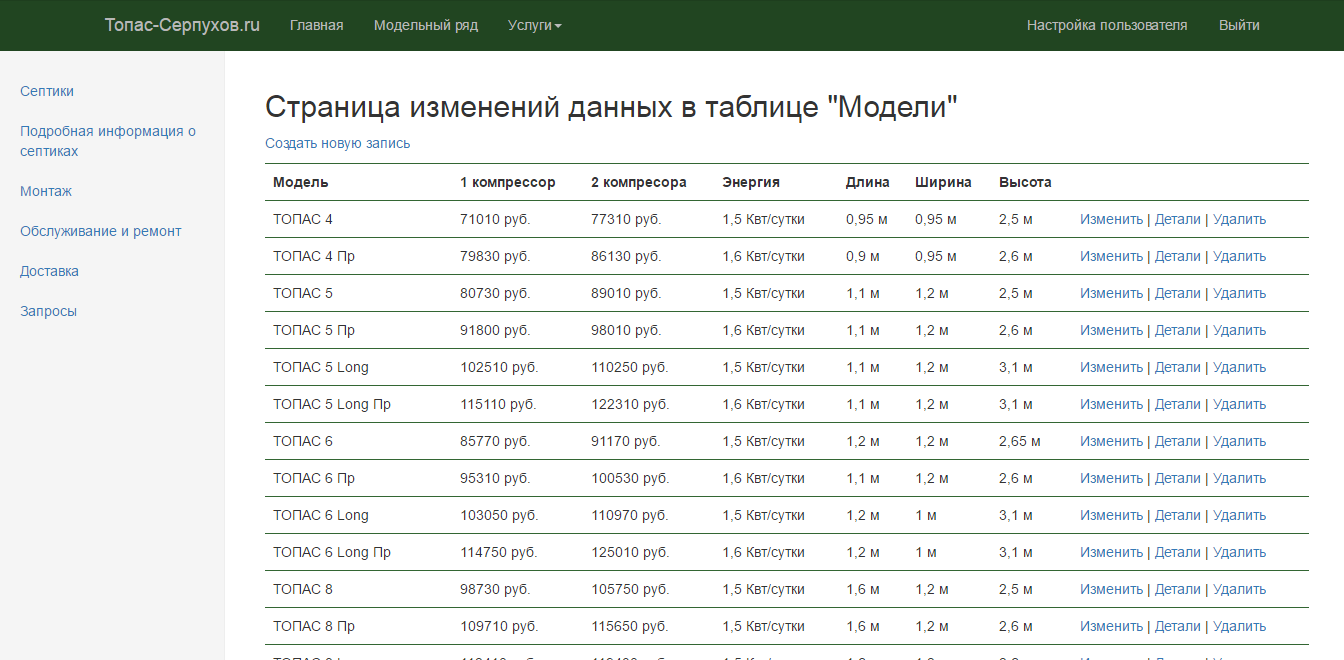


Рис. 52

3. РУКОВОДСТВО ПО ИСПОЛЬЗОВАНИЮ ПРОГРАММЫ

3.1. Руководство системного программиста

3.1.1. Общие сведения о программе

Данная программа предназначена для применения в системе учёта заказов на выполнение строительных работ. Исходные данные – заказ пользователя заносится в таблицу запросов. Программа выполняет анализ вводимых пользователем данных, проверяет на корректность введённых данных, а также хранение введённых пользователем данных.

Для корректной работы данной программы требуется следующее аппаратное и программное обеспечение:

- процессор; Intel Pentium 4 или AMD Phenom X2, не ниже 2 ГГц;

- оперативная память: 1Гб;

- видеоадаптер: стандартный SVGA-видеоадаптер;

- стандартная клавиатура;

- манипулятор «мышь»;

- операционная система Windows XP/Vista/7/8/10;

- Web-браузер «Google Chrome» или его аналоги.

3.1.2. Структура программы

Карта сайта представлена на рисунке 53.

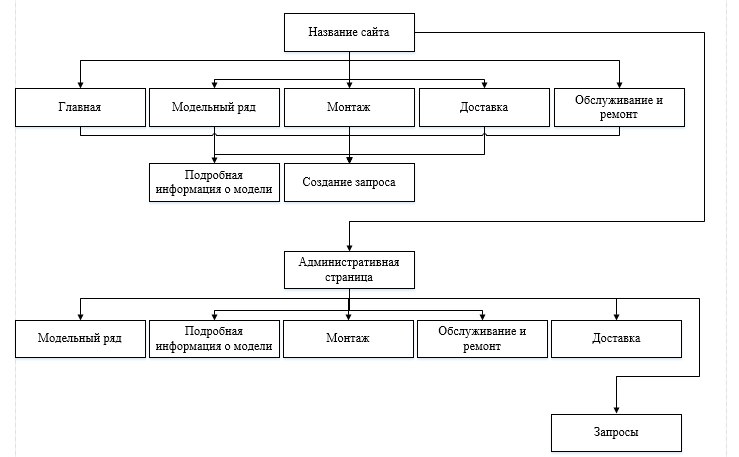


Рис. 53

3.1.3. Настройка программы

Настройка программы не предусматривается.

3.1.4. Проверка программы

Проверка работы программы представлена в приложении 6.

3.1.5. Дополнительные возможности

- отображение всех данных на сайте;

- просмотр всех добавленных запросов;

3.1.6. Сообщения системному программисту

Сообщения системному программисту не предусмотрены.

3.2. Руководство программиста

3.2.1. Назначение и условия применения программы

Данная программа предназначена для применения в системе учёта заказов на выполнение строительных работ. Исходные данные – заказ пользователя заносится в таблицу запросов. Программа выполняет анализ вводимых пользователем данных, проверяет на корректность введённых данных, а также хранение введённых пользователем данных.

3.2.2. Характеристики программы

Для корректной работы данной программы требуется следующее аппаратное и программное обеспечение:

- процессор; Intel Pentium 4 или AMD Phenom X2, не ниже 2 ГГц;

- оперативная память: 1Гб;

- видеоадаптер: стандартный SVGA-видеоадаптер;

- стандартная клавиатура;

- манипулятор «мышь»;

- операционная система Windows XP/Vista/7/8/10;

- Web-браузер «Google Chrome» или его аналоги.

Для запуска сайта следует в адресной строке Web-браузера нужно записать адрес «topas-serpukhov.ru».

3.2.3. Обращение к программе

Проект сайта содержит следующие страницы:

- Home/Index – главная страница;

- Home/Models – страница «Модельный ряд»;

- Home/Details – страница «Детальная информация»;

- Home/Services – страница «Обслуживание»;

- Home/Installations – страница «Монтаж»;

- Home/Delivery – страница «Доставка»;

- Home/CreateRequest – страница «Создание запроса»;

- Account/Login – страница входа;

- SepticTank/Index – страница изменений данных в таблице «Модели»;

- SepticTank/Create – страница создания записи в таблице «Модели»;

3.2.4. Входные и выходные данные

Входные данные указаны в таблице 11

Таблица 11 – Входные данные

|  |  |
| --- | --- |
| Входные данные | Описание действий |
| Имя | Вводится имя пользователя |
| Телефон | Вводится телефон пользователя для связи с ним |
| E-Mail | Вводится электронная почта пользователя |
| Запрос | Вводится запрос пользователя |

Выходные данные указаны в таблице 23

Таблица 12 – Выходные данные

|  |  |
| --- | --- |
| Выходные данные | Описание действий |
| Обработка запроса | Администратор обрабатывает запрос пользователя и связывется с ним |

3.2.5. Сообщения

Сообщение об ошибках указаны в таблице 13

Таблица 13 – Сообщения об ошибках

|  |  |
| --- | --- |
| Сообщение | Описание действий при получении сообщения |
| Не заполнили поля | Заполнить все поля введённых исходных данных |

Продолжение таблицы 9

|  |  |
| --- | --- |
| Сообщение | Описание действий при получении сообщения |
| Некорректный ввод | Неправильный тип вводимых данных |
| Ошибка при подключении | Отсутствует подключение с интернетом |

3.3. Руководство оператора

3.3.1. Назначение программы

Данная программа предназначена для применения в системе учёта заказов на выполнение строительных работ. Исходные данные – заказ пользователя заносится в таблицу запросов. Программа выполняет анализ вводимых пользователем данных, проверяет на корректность введённых данных, а также хранение введённых пользователем данных.

3.3.2. Условия выполнения работы

Для корректной работы данной программы требуется следующее аппаратное и программное обеспечение:

- процессор; Intel Pentium 4 или AMD Phenom X2, не ниже 2 ГГц;

- оперативная память: 1Гб;

- видеоадаптер: стандартный SVGA-видеоадаптер;

- стандартная клавиатура;

- манипулятор «мышь»;

- операционная система Windows XP/Vista/7/8/10;

- Web-браузер «Google Chrome» или его аналоги.

3.3.3. Выполнение сайта

Для того, чтобы запустить сайт, необходимо в строке браузера набрать «topas-serpukhov.ru».

Далее появляется главная страница сайта. Чтобы начать работу с данными на сайте необходимо авторизоваться в нём, дописав к адресу страницу авторизации (рис. 54).

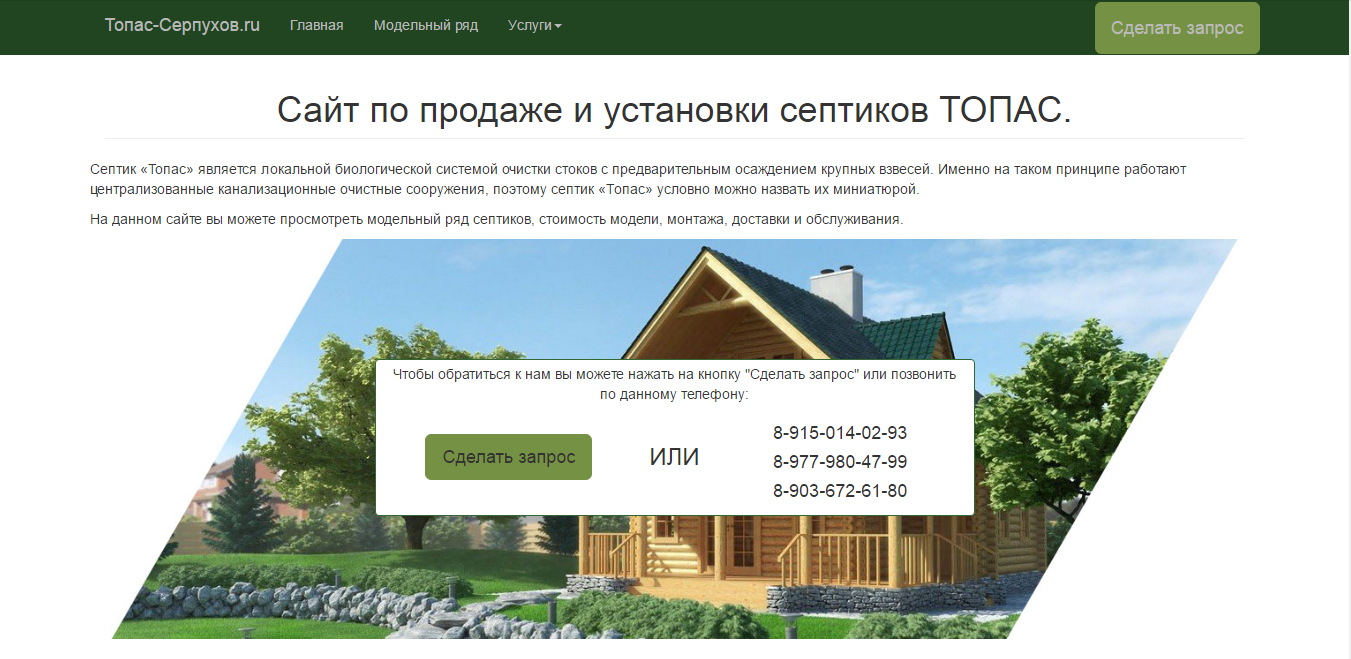


Рис. 54

Далее появляется станица авторизации на сайте. Для входа в программу необходимо ввести E-Mail и пароль в текстовое поле, и нажать кнопку «Вход» (рис. 55).

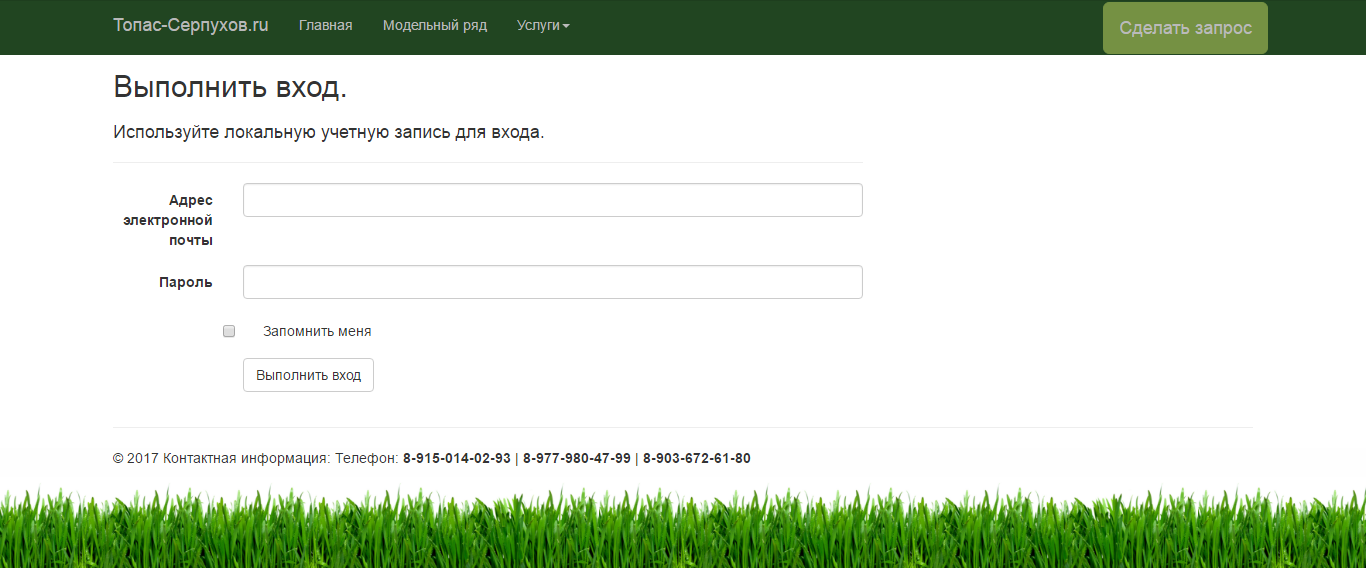


Рис. 55

После нажатия на кнопку администратору открывается страница администрации (Рис. 56).

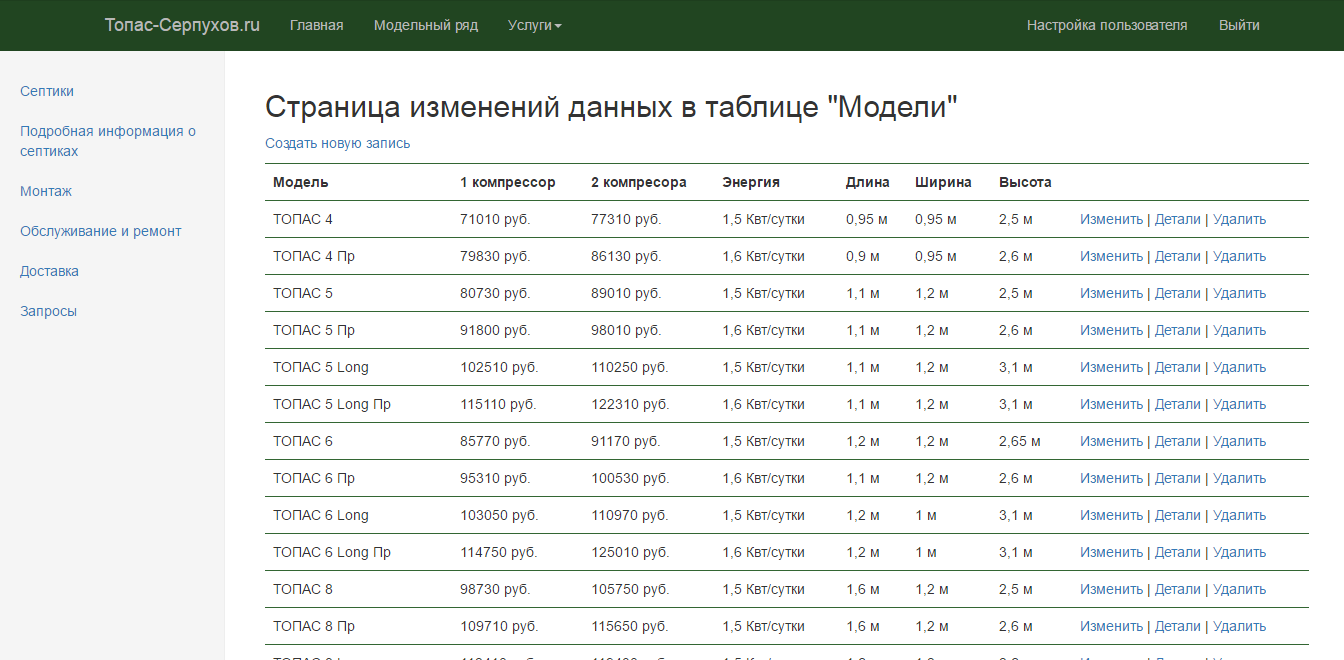


Рис. 56

На каждую таблицу в БД отводится контроллер и страницы для отображения всех данных в таблице, страницы создания, изменения, удаления и детального просмотра информации о записи (Рис. 57-60).

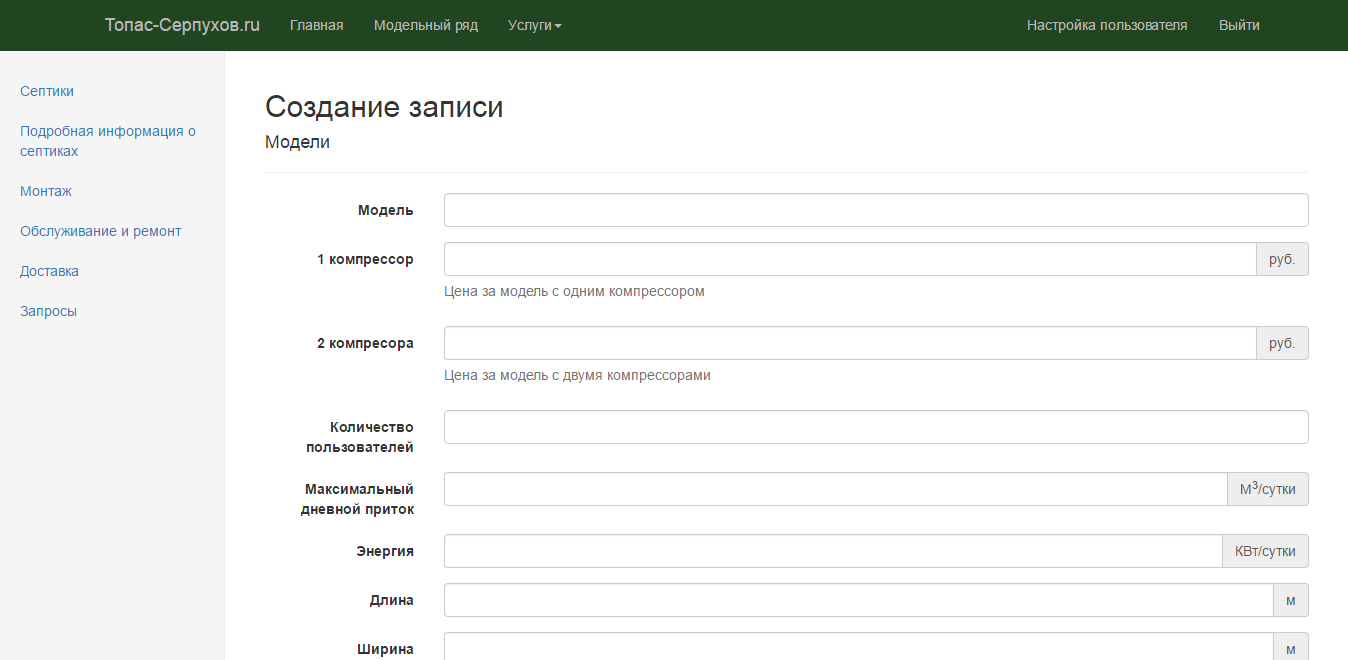


Рис. 57

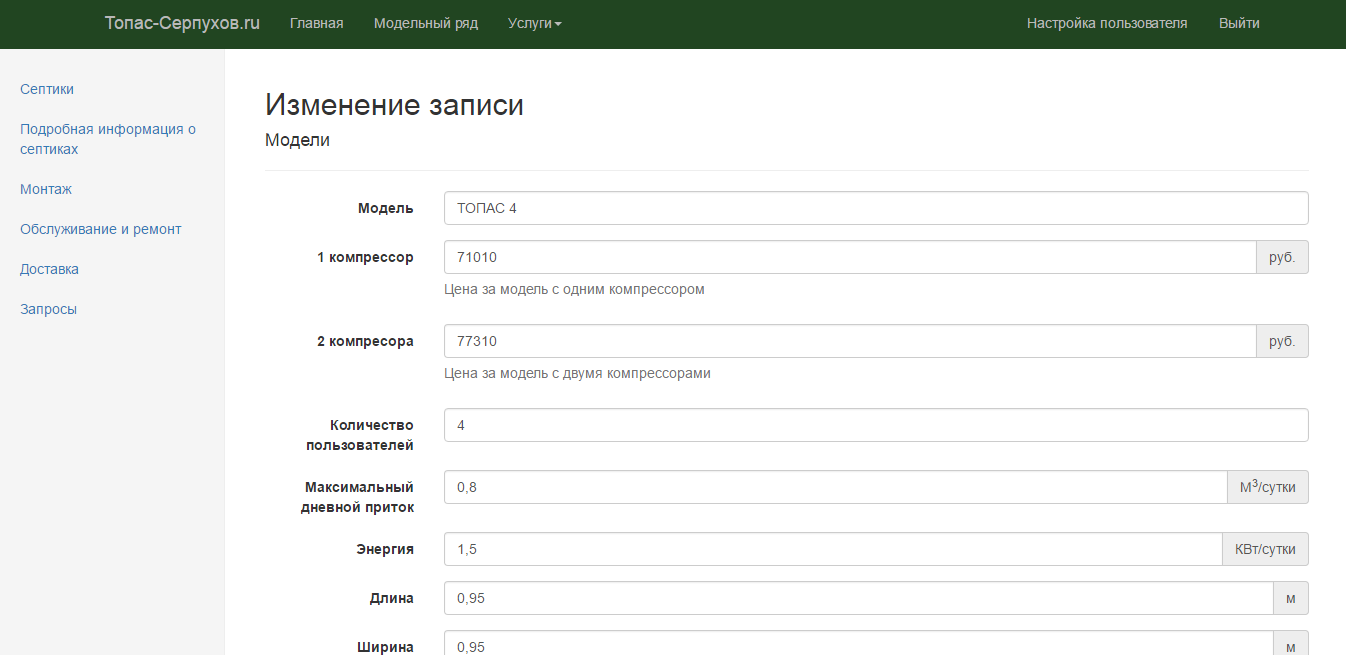


Рис. 58

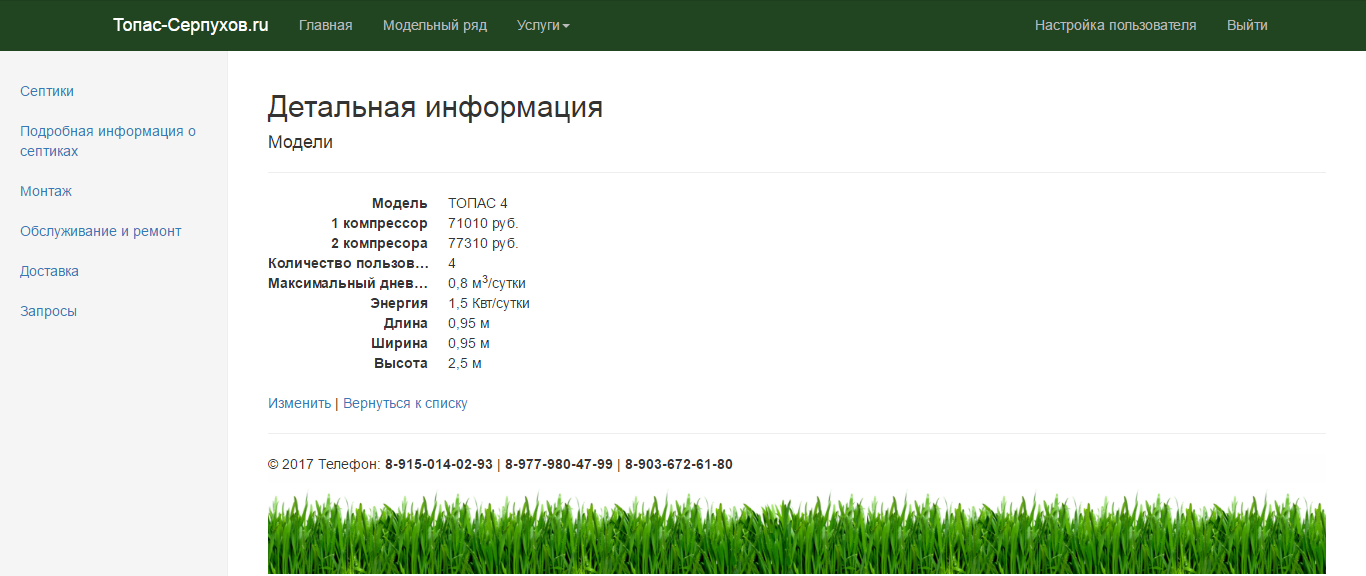


Рис. 59

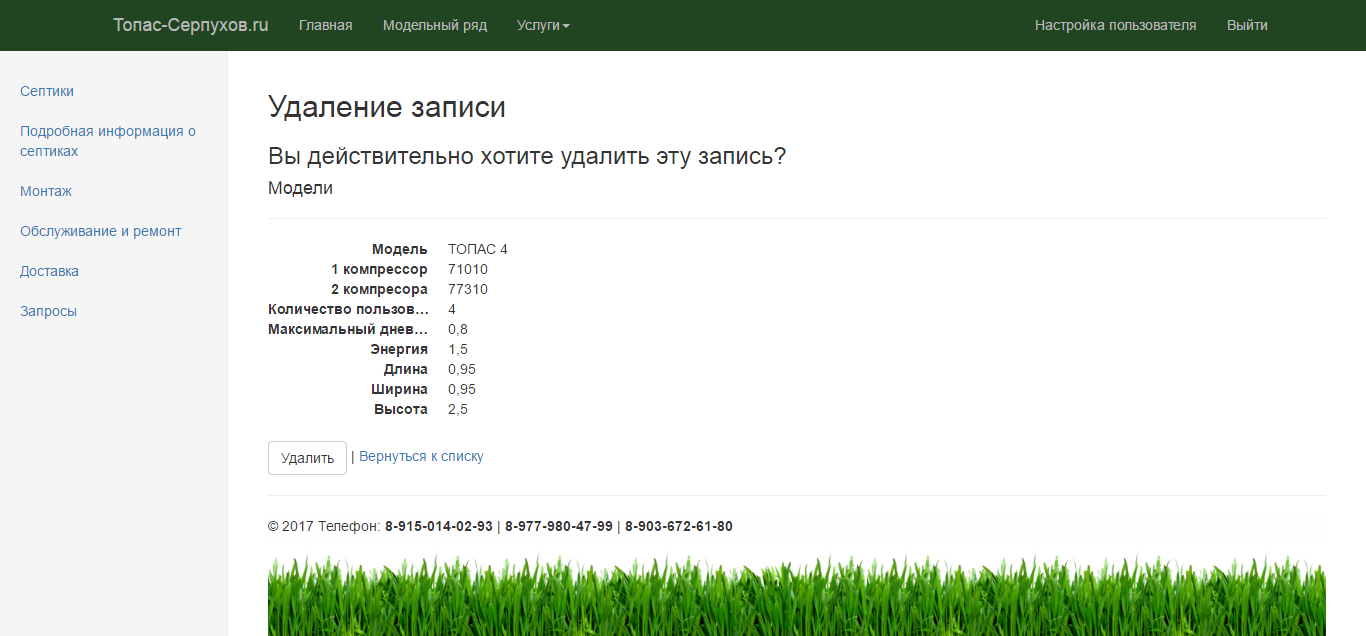


Рис. 60

3.3.4. Сообщения оператору

Сообщение об ошибках указаны в таблице 14

Таблица 14 – Сообщения об ошибках

|  |  |
| --- | --- |
| Сообщение | Описание действий при получении сообщения |
| Не заполнили поля | Заполнить все поля введённых исходных данных |
| Некорректный ввод | Неправильный тип вводимых данных |
| Ошибка при подключении | Отсутствует подключение с интернетом |

ЗАКЛЮЧЕНИЕ

Я приобрел, практические навыки использования среды программирования Visual Studio 2015 ASP.NET MVC, и применил на практике:

* создание клиент-серверного сайта БД;
* создание страницы администрирования сайта.

Закрепил практические навыки разработки БД в СУБД MS SQL Server:

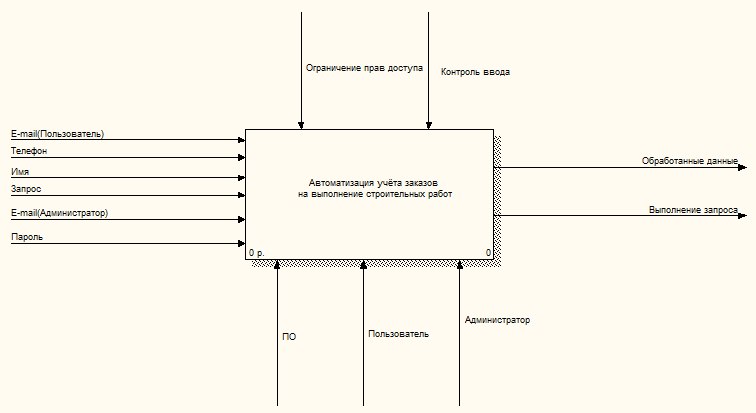
* создание базы данных
* создание структур таблиц баз данных
* ввод данных в таблицы базы данных
* разработку процедур базы данных
* разработку триггеров базы данных.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

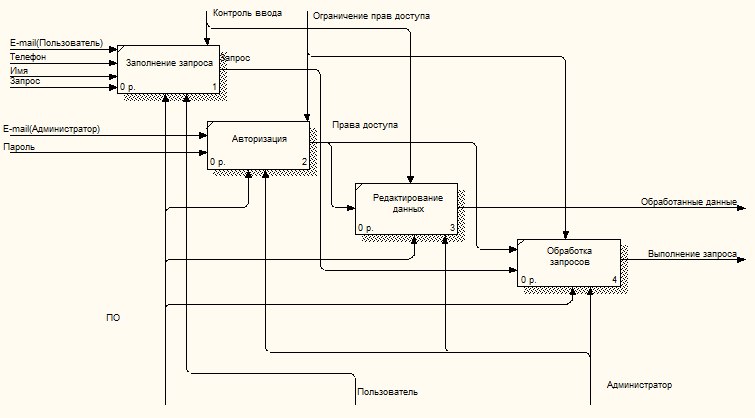
1. Дейтел П., Дейтел Х. "Как программировать на Visual С# 2012. 5-е изд . - СПб.: Питер, 2014. - 864 с .: ил . - (Серия «Библиотека программиста»)";

2. Работа с базами данных на языкеC#. Технология АDO .NET: учебное пособие / сост. О. Н. Евсеева, А. Б. Шамшев. – Ульяновск: УлГТУ, 2009. – 170 с.

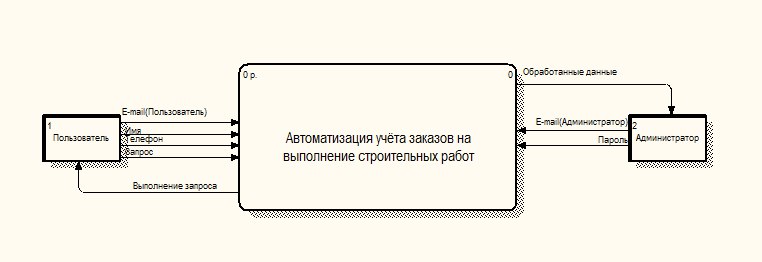
3. Семенова И.И. "Разработка клиент-серверных приложений в Microsoft SQL Server 2005 и Microsoft Visual C# 2005 Express Edition: Учебно-методическое посо бие."– Омск: Изд-во СибАДИ, 2010.– 65 с.

ПРИЛОЖЕНИЕ 1

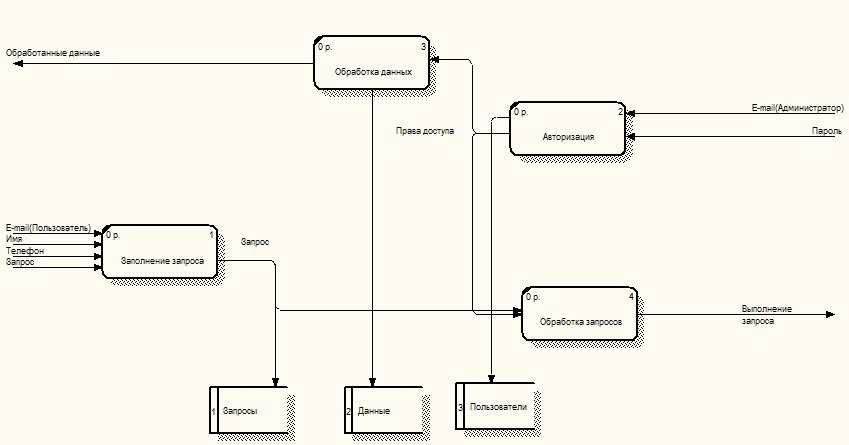
ПРИЛОЖЕНИЕ 2



ПРИЛОЖЕНИЕ 3



ПРИЛОЖЕНИЕ 4



ПРИЛОЖЕНИЕ 5

Тестирование страницы «Авторизация»

Таблица 15

|  |  |
| --- | --- |
| Наименование | Описание |
| Test Case # | TC\_UI\_1 |
| Приоритет теста | Высокий |
| Название тестирования/Имя | Проверка страницы авторизация при вводе неправильного пароля |
| Резюме испытания | Проверка входа на сайт |
| Шаги тестирования | 1. Ввести логин 2. Ввести пароль 3. Нажать на кнопку Вход |
| Данные тестирования | Логин=Admin  Пароль=1234567 |
| Ожидаемый результат | Сообщение об ошибке «Неправильное имя пользователя/пароль» |
| Фактический результат | Сайт выдал сообщение об ошибке «Неправильное имя пользователя/пароль» |
| Предпосылки | - |
| Постусловия | Запрет на вход на сайт |
| Статус  (Pass/Fail) | Pass |
| Комментарии | - |

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ 5

Тестирование страницы «Авторизация»

Таблица 16

|  |  |
| --- | --- |
| Наименование | Описание |
| Test Case # | TC\_UI\_2 |
| Приоритеттеста | Высокий |
| Название тестирования/Имя | Проверка страницы авторизация при вводе несуществующего логина |
| Резюме испытания | Проверка входа на сайт |
| Шаги тестирования | 1. Ввести логин 2. Ввести пароль 3. Нажать на кнопку Вход |
| Данные тестирования | Логин=Student  Пароль=Admin |
| Ожидаемый результат | Сообщение об ошибке «Неправильное имя пользователя/пароль» |
| Фактический результат | Сайт выдал сообщение об ошибке «Неправильное имя пользователя/пароль» |
| Предпосылки | - |
| Постусловия | Запрет на вход на сайт |
| Статус  (Pass/Fail) | Pass |
| Комментарии | - |

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ 5

Тестирование страницы «Авторизация»

Таблица 17

|  |  |
| --- | --- |
| Наименование | Описание |
| Test Case # | TC\_UI\_3 |
| Приоритет теста | Высокий |
| Название тестирования/Имя | Проверка страницы авторизация с правильным логином и паролем |
| Резюме испытания | Проверка входа на сайт |
| Шаги тестирования | 1. Ввести логин 2. Ввести пароль 3. Нажать на кнопку Вход |
| Данные тестирования | Логин= glss2fa@gmail.com  Пароль= quNX9fBR |
| Ожидаемый результат | Вход администратора на сайт |
| Фактический результат | Вход администратора на сайт |
| Предпосылки | - |
| Постусловия | Отображение страницы администрирования |
| Статус  (Pass/Fail) | Pass |
| Комментарии | - |