ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ МОРДОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Н. П. ОГАРЁВА»

Факультет довузовской подготовки и среднего профессионального образования

Выпускающая предметная цикловая комиссия (кафедра) общепрофессиональных специальных (информационно-коммуникационных) дисциплин

УТВЕРЖДАЮ
Зам. декана по учебной работе, преподаватель ВКК
_____ В. А. Богатырская
(подпись)
«___» _____ 2019 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ДИПЛОМНОМУ ПРОЕКТУ

на тему Разработка и реализация сайта для автотранспортной компании (на примере ООО "Престиж")

Автор проекта О.В. Неськин
(подпись) (дата)

Специальность 090203 Программирование в компьютерных системах

Обозначение дипломного проекта ДП- 02069964- 090203- 11- 19

Руководитель проекта

Преподаватель А. А. Прокин
(подпись) (дата)

Нормоконтролер

Преподаватель ПКК М. Ф. Петянкин
(подпись) (дата)

Рецензент
Начальник отдела разработки и внедрения программного обеспечения Д. С. Бобров
(подпись) (дата)

Саранск
2019

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ МОРДОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Н.П. ОГАРЁВА»

Факультет довузовской подготовки и среднего профессионального образования

Выпускающая предметная цикловая комиссия (кафедра) общепрофессиональных специальных (информационно-коммуникационных) дисциплин

УТВЕРЖДАЮ
Зам. декана по учебной работе, преподаватель ВКК
_____ В. А. Богатырская
(подпись)
«___» _____ 2019 г.

ЗАДАНИЕ НА ДИПЛОМНЫЙ ПРОЕКТ

Студент Неськин Олег Валерьевич

1 Тема Разработка и реализация сайта для автотранспортной компании (на примере ООО "Престиж")

Утверждено приказом №2878-с от 06.05.2019 г.

2 Срок представления работы к защите 17.06.2019 г.

3 Исходные данные для научного исследования (проектирования) техническое задание на разработку сайта

Программный продукт должен быть реализован с интуитивно понятным интерфейсом пользователя и выполнять следующие функции:

- отображать основную информацию о фирме;
- предоставлять взаимодействие;
- отображать данные в понятном для пользователя виде

4 Содержание выпускной квалификационной работы

4.1 Изучение предметной области

4.1.1 Анализ предметной области

4.1.2 Анализ методов и средств проектирования

4.1.3 Языки сценариев и программирования

4.2 Разработка сайта

4.2.1 Разработка интерфейса

4.2.2 Создание формы обратной формы

4.3 Тестирование сайта

5 Приложения:

ПРИЛОЖЕНИЕ А Коды программ (скриптов)

ПРИЛОЖЕНИЕ Б Диаграмма деятельности (Activity)

ПРИЛОЖЕНИЕ В Диаграмма состояний для варианта использования

ПРИЛОЖЕНИЕ Г Диаграмма последовательностей для варианта использования «Авторизация»

ПРИЛОЖЕНИЕ Д Схема базы данных

Руководитель работы _____ А. А. Прокин
подпись, дата

Задание принял к исполнению _____ О.В. Неськин
подпись, дат

РЕФЕРАТ

Пояснительная записка к дипломному проекту содержит __ страниц, 41 использованных источниках, 5 приложений.

САЙТ, HTML, ЯЗЫК ПРОГРАММИРОВАНИЯ, ПРЕДМЕТНАЯ ОБЛАСТЬ, ФИРМА, WEB-САЙТ, HTML, ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ, PHP.

Целью работы является разработка сайта для фирмы занимающейся грузовыми перевозками.

В процессе работы прорабатывалась структура сайта, проводилось теоретическое исследование процесса его создания.

В результате проведенной работы создан полноценный сайт на основе анализа предметной области и технического задания.

Степень внедрения — частичная.

Область применения — в практике работы техника-программиста.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....3
1 ОБЩАЯ ЧАСТЬ.....6
1.1. Описание предметной области.....6
1.2. Архитектура Веб-приложения и выбор программных средств.....7
1.3. Требования к веб-приложению и постановка задачи.....12
2 РАЗРАБОТКА И РЕАЛИЗАЦИЯ САЙТА ДЛЯ АВТОТРАНСПОРТНОЙ КОМПАНИИ.....16
2.1. Технологии и программные средства проекта.....16
2.2. Проектирование с помощью CASE технологий.....25
2.3. Описание базы данных.....27
2.4. Этапы разработки Веб - сайта.....39
2.4.1Формирование веб-страниц приложения.....39
2.4.2. Создание новых страниц и наполнение их контентом.....43
2.4.3. Создание БД и инсталляция.....45
2.5. Итоговая реализация сайта.....47
2.6. Безопасность веб-сайта.....53
ЗАКЛЮЧЕНИЕ.....55
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....57
ПРИЛОЖЕНИЕ А Коды программ (скриптов).....61
ПРИЛОЖЕНИЕ Б Диаграмма деятельности (Activity).....80
ПРИЛОЖЕНИЕ В Диаграмма состояний для варианта использования.....81
ПРИЛОЖЕНИЕ Г Диаграмма последовательностей для варианта использования «Авторизация» 82
ПРИЛОЖЕНИЕ Д Схема базы данных.....83

ВВЕДЕНИЕ

Главным принципом деятельности любой компании, предоставляющей услуги, является максимальная информативность, открытость и прозрачность работы, расширение связей между сотрудниками и клиентами с целью обеспечения реакции на запросы клиентов в самые минимально-короткие сроки.

Интернет-сайт компании должен не только содержать информацию о деятельности компании, но обладать динамической функциональностью, обеспечивающей удобный онлайн интерфейс взаимодействия клиентов компании с сотрудниками.

Наличие достоверной и полной информации по деятельности компании на современном уровне развития интернет технологий, является необходимым, но не достаточным условием популярности ресурса и спроса на услуги компании. Современный сайт должен обладать рядом функциональных свойств, обеспечивающих обратную связь и онлайн подписку на услуги компании. В этой связи, в данном дипломном проекте акцент в разработке был сделан не только на наличие информации по деятельности компании, но и на разработку таких технологий, как динамическое формирование контента, регистрацию и авторизацию пользователей, отработку онлайн технологий оформления заявок на услуги. Веб-ресурс, это не просто информационный сайт, а еще и веб-ориентированная информационная система, с которой работают группы пользователей — зарегистрированные посетители (клиенты) и, в тоже время, персонал компании.

Актуальность выбранной темы обусловлена тем, что фундаментальной составляющей ИТ-стратегии любой компании в настоящих условиях является, с одной стороны, вывод информации о компании глобальное информационное пространство, а с другой — автоматизация

бизне

с-процессов в рамках самой компании, обеспечение оперативного взаимодействия сотрудников компании с целью своевременного удовлетворения потребностей в услугах клиентов.

Целью выпускной работы является разработка сайта компании, веб-ориентированной информационной системы (ИС) компании для использования его клиентами компании для выбора услуг, оформления заявок на услуги и сотрудниками компании для управления бизнес-процессами, связанными с реализацией заявок клиентов.

Объектом выпускной работы является сайт компания, предоставляющая услуги по перевозкам грузов автотранспортом различного объема и грузоподъемности, в том числе и специализированной техникой.

Предметом выпускной квалификационной работы является web-сайт автотранспортной компании.

Для реализации цели нами были поставлены следующие задачи:

– изучить предметную область и существующие аналоги;

– изучить архитектуру Веб-приложений и методы их разработки;

– разработать требования к Веб-приложению;

– разработать модели UML модели предметной области;

— реализовать и протестировать Веб-приложение, включающее базу данных системы и программное обеспечение, реализующее веб-интерфейс пользователей системы.

Методы исследования теоритические:

– анализ учебной, учебно-методической литературы;

– эмпирические: опрос, наблюдение.

Методологической основой исследования выступили труды отечественных и зарубежных учёных Т. Коннолли, Т. Кормен, А. В. Фролов.

Веб-приложение должно выполнять следующие функции:

— обеспечивать регистрацию для сотрудников и потенциальных клиентов компании;

– предоставлять информацию для клиентов компании, спектр предоставляемых услуг и т.д.;

– содержать форму для оформления заказа-заявки клиентов компании;

– обеспечивать возможность для зарегистрированных сотрудников общаться по средствам обратной связи.

– дружественный интерфейс должен обеспечить сокращение времени на переговоры с клиентами за счет предоставление полного объема информации;

– содержать веб-интерфейс для администратора системы;

– обратную связь с пользователями для отправки сообщений администратору системы об удобстве работы системы;

– обеспечивать разделение доступа групп пользователей администратора, зарегистрированных сотрудников, клиентов компании.

Использование информационной системы обеспечивает принципиально новые возможности во взаимодействии сотрудников посредством сети интернет с клиентами компании. Для клиентов нет необходимости в личном контакте. Со всей доступной информацией компании они может познакомиться онлайн, просмотреть услуги, найти необходимую информацию по поисковым запросам, оформить заявку на оказание услуг по перевозке грузов.

1 Общая часть

1.1. Описание предметной области

Компания занимается грузоперевозками и предоставлением таких видов специальной техники как:

- илососы;
- варовайки;
- грузовые эвакуаторы;
- экскаваторы (гусеничные, колесные);
- телескопические фронтальные погрузчики;
- самосвалы.

Компания занимается также:

а) платными стоянками (на улице);

б) грузовой и легковой шиномонтаж.

Компания располагает хорошим транспортным оборудованием, транспортными средствами управляют высокопрофессиональные водители и экспедиторы, имеющие большой опыт в оформлении всех необходимых документов. Опыт сотрудников компании помогает оформить всю необходимую документацию для перевозки в сжатые сроки, что позволяет сэкономить время клиента. Специалисты логистического отдела разрабатывают оптимальный маршрут движения транспорта для перевозки груза, а также рассчитывают стоимость перевозки без завышения и переплат.

Знания и опыт специалистов позволяют осуществлять грузовые автоперевозки, принимая эффективные решения в области логистики, что способствует успешности и процветанию клиентов компании. Для того,

ч тоб

ы предоставляемые транспортные услуги перевозки грузов стали еще более эффективными, дирекция компании проводит работы по совершенствованию операционной среды работы компании, в частности ориентируясь на автоматизацию основных бизнес-процессов.

Целью работы является проектирование и разработка сайта компании веб-ориентированной информационной системы автотранспортной компании с функционалом автоматизации онлайн формирования заявок клиентов и взаимодействия клиентов компании и сотрудников.

На современном этапе информатизации общества, глобального проникновения интернет технологий во все сферы жизни, наличие достоверной и полной информации по деятельности компании является недостаточным условием популярности веб ресурса, его эффективности с точки зрения улучшения качества реакции на потребности общества. Современный веб- ресурс должен обладать рядом функциональных свойств, обеспечивающих обратную связь и онлайн подписку на услуги компании и пр.

В этой связи, в данном проекте акцент в разработке был сделан на разработку таких технологий, как динамическое формирование контента по услугам, регистрацию и авторизацию пользователей, отработку онлайн технологий оформления заявок на услуги. Это сайт, веб-ориентированная информационная система, с которой работают группы пользователей — зарегистрированные посетители (клиенты) и, в тоже время, персонал компании.

1.2. Архитектура Веб-приложения и выбор программных средств

Любое веб-приложение, сайт-это набор программных компонентов, часть из которых находится на компьютере

пользователь и другая часть расположены на удаленном сервере. Компьютер пользователя называется клиентом, а часть приложения называется клиентской частью. Такое программное обеспечение построено на архитектуре "клиент-сервер". Серверная часть веб-приложение выполняется на сервере, клиент получает только результаты работы. Клиент-серверная архитектура информационной системы предполагает взаимодействие двух процессов обмена данными-клиентского и серверного [25, 26].

Клиентский процесс реализует пользовательский интерфейс, преобразует запросы пользователей в запросы к серверу и отправляет эти запросы на сервер. Сервер принимает запросы, обрабатывает их, и передает результат обработки клиентского процесса. Сервер может получить доступ к серверу баз данных. Сервер базы данных находится на сервере. Запросы отправляются по протоколу HTTP. Трехуровневая система характеризуется включением третьего уровня, в котором реализована логика, правила бизнеса (рисунок 1).

Отображение результатов запросов на клиентском компьютере, а также получение данных от клиента и отправка их на сервер осуществляется браузером - Internet Explorer, Mozilla, orega и др.). Browsertoolbarpane, полученные от сервера, в виде страницы представлены в формате HTML.

Во время обработки запроса пользователя веб-приложение генерирует ответ, который обычно является результатом выполнения программного кода, запущенного на стороне сервера. В результате образуется HTMLстраница,отправляемая клиенту.

Для проектирования и реализации веб-ориентированной системы интернет-разработчику необходим пакет (набор) серверного программного обеспечения, включающий в себя четыре базовые составляющие. Существуют различные варианты компоновки четырёх основных групп программных средств. Название компоновки, как правило, состоит из имени операционной системы (ОС) и аббревиатуры AMP: «A» — Apache (веб-

сервер

), «М» — MySQL (СУБД), «Р» — PHP (язык программирования, используемый для создания веб- приложений). Различают следующие наборы программ в зависимости от ОС:

- WAMP — ОС Microsoft Windows + AMP;
- LAMP — ОС GNU/Linux + AMP;
- BAMP — ОС BSD + AMP;
- MAMP — ОС Mac OS X + AMP;
- SAMP — ОС Solaris + AMP;
- PAMP — Personal Apache + MySQL + PHP (набор серверов для платформы S60).

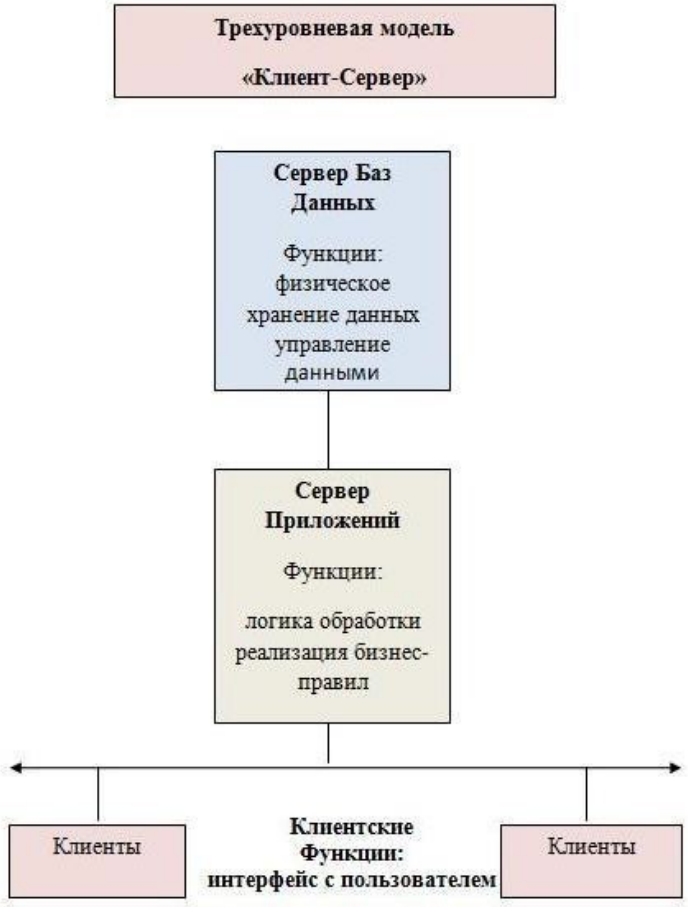


Рисунок 1 — Трех уровневая клиент-серверная архитектура

В качестве средств, для разработки веб-ориентированной системы компании был выбран комплекс программных средств LAMP. Обоснование этого выбора приведено ниже.

Первым компонентом комплекса программных средств является операционная система Microsoft Windows. ОС Microsoft Windows в настоящее время является самой распространённой операционной системой в мире. На момент начала проектирования, реализации и сдачи в эксплуатацию системы компании на компьютерах интернет-разработчиков стояла ОС Microsoft Windows 7, этим и обуславливается выбор данной ОС в качестве первого компонента комплекса программных средств.

Веб-сервер Apache, СУБД MySQL и язык программирования PHP, используемый для создания веб-приложений, выбраны в связи с тем, что указанные программные средства являются бесплатными, широко-распространёнными и апробированными.

При выборе веб-сервера Apache в качестве второго компонента комплекса программных средств для разработки веб-приложения руководствовались следующими особенностями данного ПО. Свободный, кроссплатформенный Apache HTTP-сервер поддерживает следующие ОС: Linux, BSD, Mac OS, Microsoft Windows, Novell NetWare, BeOS. Основные достоинства веб-сервера Apache, это надёжность и гибкость конфигурирования. Он настраивается на подключение внешних модулей для предоставления данных, настраивать СУБД при необходимости аутентификации, адаптировать сообщения об ошибках и т. д. Для виртуальных хостов можно создавать персональные настройки ядра и модулей, ограничивать доступ к сайту или некоторым файлам.

Вдобавок имеются дополнительные модули, которые добавляют к Apache способность поддержки разных языков программирования.

Его достоинствами являются надежность, исключительная производительность и огромный набор функций и дополнительных

модуле

й, устанавливаемых на большинство существующих ОС (транслятор языка PHP, интегрируемая СУБД MySQL). Одним из главных преимуществ этого сервера является свободное бесплатное распространение. Это дает возможность, помимо экономии денег, быстро исправлять ошибки и вносить в код программы необходимые дополнения.

В качестве СУБД выбрана MySQL. MySQL — это свободная (бесплатная) система управления реляционными базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle. Преимуществами СУБД MySQL являются: быстрый поиск и управление записями; обеспечение гибкости поиска; быстрая настройка формата вывода; многопользовательский режим доступа; обеспечение быстрого удаленный доступ. СУБД MySQL обладает высокой производительностью и простотой использования. Более полное обоснование выбора MySQL дано в разделе 2.1.2.

PHP — это скриптовый язык программирования. Он представлен в пакете опций услуг большинства хостинг-провайдеров. Язык считается одним из основных среди языков программирования динамических веб-приложений и одним из наиболее популярных языков разработки веб-страниц (наряду с JSP, Perl). Язык прост в использовании, обладает хорошей скоростью выполнения, имеет богатую функциональность, является кроссплатформенным и распространяется на основе лицензии PHP. Язык обладает большим набором встроенных функций для быстрой разработки веб-приложений - автоматическая передача параметров посредством post и get инструкций; взаимосвязь с основными популярными СУБД — MySQL, SQLite, PostgreSQL, Oracle, Microsoft SQL Server, Sybase. Благодаря всем этим свойствам, в качестве основного программного средства для разработки веб-приложения был выбран язык программирования PHP.

Для интерактивной работы с СУБД MySQL разработано программа, веб- приложение PHPMyAdmin. Оно обеспечивает визуальную, онлайн работу

с БД не посредством команд на SQL, а посредством веб-интерфейса. PHPMyAdmin — веб-приложение с открытым кодом на языке PHP. Оно представляет разработчикам веб-интерфейс администрирования СУБД MySQL в виде таблиц и форм ввода данных. С PHPMyAdmin проводится работа посредством браузера для осуществления всех операций с базой данных MySQL - запуск команд SQL и просмотр содержимое таблиц базы данных. Подавляющее большинство разработчиков используют веб-приложение PHPMyAdmin в качестве панели управления для всестороннего управления БД.

1.3. Требования к веб-приложению и постановка задачи

Существуют функциональные и нефункциональные требования к веб-приложениям и информационным системам в целом. Функциональные требования, это совокупность функций системы, которая автоматизирует бизнес-процессы пользователей.

Нефункциональные требования-описание свойств приложения, необходимых пользователю при работе с системой и программно-техническими характеристиками системы. Общепринятыми нефункциональными требованиями к системе являются::

- требования к надежности приложений определяются параметрами сервера приложений и базы данных, максимальным количеством пользователей, временем обработки запросов к системе, количеством возможных системных сбоев;

- требования к производительности определяются как среднее время обработки запроса системы ;

требования безопасности веб-приложения включают: дифференциация

прав доступа к функциям и данным пользователей, мониторинг уровней доступа пользователей, авторизация пользователей системы;

- требования к масштабируемости определяют возможности производительности системы при повышенной нагрузке и добавлении ресурсов. Для пользователя масштабируемого веб-приложения время отклика системы на запросы пользователей должно оставаться практически постоянным по мере увеличения нагрузки или добавления дополнительного программного обеспечения для обработПользователями разрабатываемого веб-приложения являются сотрудники компании и клиенты компании.

Пользуясь данным веб-приложением, сотрудники и клиенты компании будут иметь возможность быстро и удобно получать доступ ко всей необходимой внутренней информации компании:

- клиенты смогут оформлять заказы-заявки на услуги, просматривать состояние заказов, выбирать услуги автотранспорта;

- сотрудники могут просматривать информацию по сделанным заявкам и принимать решения;

- сотрудники и клиенты могут просматривать информацию по услугам, автотранспорту, водителям производить поиск необходимого автотранспорта по ряду ключевых параметров;

- отправлять сообщения и предложения посредством формы обратной связи.

Использование сайта, веб-приложения, базы данных предоставляет новые возможности по совместной работе сотрудников компании и клиентов компании. Для клиентов компании минимизируется необходимость личного контакта с сотрудниками компании.

Во вторую группу пользователей Веб-приложения входят администраторы. Администратор системы, это специалист в области

информаци

онных технологий, который может поддерживать систему в технически исправном состоянии.

Администратор выполняет следующие функции:

- поддерживает процесс регистрации пользователей, подтверждает или отклоняет регистрацию групп пользователей;
- обрабатывает запросы (в виде присланного письма по электронной почте) потенциальных пользователей определяет целесообразность удовлетворения запросов на вхождение в определенную группу;
- поддерживает распределенный доступ групп пользователей (путем подтверждения регистрации с определенным статусом);
- добавляет, удаляет, обновляет информацию — новости, фотографии, документы и пр.;
- отправляет письма различным группам пользователей, клиентам.
- следит за работой сервера;
- изменять дизайн сайта;
- удалять пользователей сайта;
- создает резервные копии базы данных сайта.

С точки зрения технологии и организации работы пользователя администратора минимальными функциональными требованиями к системе являются:

- возможность осуществления всех необходимые операций по работе с исходными данными и результатной информацией (разграничение доступа к данным, ввод, изменение и удаление, печать, поиск — для пользователя администратора);
- наличие понятного и удобного пользовательского интерфейса.
- добавление/изменение/удаление различных групп пользователей — сотрудников, подразделений компании;
- регистрация групп пользователей — сотрудников и клиентов компании;

- формирование поисковых запросов.

Поиск информации и формирование отчетов. Разрабатываемая информационная система позволит осуществлять поиск информации по задаваемым критериям.

Экранные формы. Система должен обеспечить ввод и модификацию данных посредством экранных форм, представленным в виде таблиц. Форма предполагает наличие полей для ввода и модификации данных. Программное обеспечение должно быть независимым от структуры таблиц базы данных.

Интерфейс пользователей. Система должна предоставлять интуитивно- понятный интерфейс пользователей, обеспечивающий простоту работы на всех ее этапах.

Защита от несанкционированного доступа. Вход в защищенные области системы, БД, поисковые компоненты должен осуществляться после ввода пользователем логина и пароля. Для этого разрабатывается программа регистрации и авторизации пользователя. После ввода логина и пароля, система определяет статус и полномочия пользователя по доступу к данным и функциям системы.

Сохранность информации. Для сохранности информации администратор системы производит операции по созданию дампов базы данных. Частота создания определяется на этапе опытной эксплуатации системы.

Разрабатываемое Веб-приложение предполагает хранение большого объема информации, должна обеспечивать анализ и оценку данных.Указанным требованиям отвечают системы управления базами данных (СУБД), одна из которых и должна быть выбрана для практической реализации данного дипломного проекта.

В первом разделе описана предметная область, которая является основой для создания сайта компании, обосновывается необходимость

создани

я сайта, содержащего информацию о компании и функционала онлайн заявок на услуги компании. Во второй части раздела обосновывается выбор программно- технических средств, для создания сайта, описаны функциональные требования к разрабатываемому сайту.

2 Разработка и реализация сайта для автотранспортной компании

2.1. Технологии и программные средства проекта

Для разработки Веб — сайта используются объектно-ориентированные технологии, которые включают в себя применение языка на этапе проектирования Веб-сайта. Инструментами создания и управления базой данных используются инструменты phpMyAdmin, HeidySQL. В качестве интегрированной среды разработки применяется ZendStudio и язык PHP. В качестве Веб-сервера используется сервер удаленного доступа Apache, а в качестве локальной среды для разработки используется Denwer.

Унифицированный язык моделирования (Unifiedmodelinglanguage, UML) — универсальный язык визуального моделирования систем. Хотя чаще всего UML связан с моделированием программных систем, он имеет гораздо более широкое применение благодаря присущей ему расширяемости [17,18]. UML сочетает в себе лучшие современные методы моделирования и разработки программного обеспечения. Фактически, язык UML был разработан таким образом, чтобы его можно было реализовать с помощью собственных инструментов. Фактически, это признание того, что крупные современные программные системы, как правило, нуждаются в инструментальной поддержке. Диаграммы UML легко воспринимаются и легко генерируются компьютерами. Важно понимать, что UML не предлагает нам никакой методологии моделирования. Конечно, некоторые методологические аспекты подразумеваются элементами, составляющими модель UML, но сам UML предоставляет только визуальный синтаксис, который может использоваться для создания моделей.

В UML есть четыре общих механизма, один за другим

приме

няется ко всему языку моделирования. Они описывают четыре подхода к стратегиям объектного моделирования, которые многократно используются в UML в разных контекстах. Это еще раз убеждает нас в простоте и элегантности структуры UML.

Модели UML имеют по крайней мере два измерения: графическое измерение, позволяющее визуализировать модель с помощью диаграмм и значков, и текстовое измерение, состоящее из спецификаций для различных элементов модели. Спецификации-это текстовые описания семантики элемента. Семантика элементов модели зафиксирована в спецификациях; без них можно только догадываться, что такое элемент на самом деле.

Суть модели заключается в наборе спецификаций. Спецификации формируют семантический фон (семантическую объединительную плоскость), который объединяет модель и наполняет ее смыслом. Различные диаграммы-это просто представления или визуальные проекции этого плана.

Разработка моделей с использованием UML, как правило, начинается с графической модели, позволяющей визуализировать систему, а затем по мере ее развития все больше семантики добавляются на задний план. Однако модель может считаться полезной или полной только в том случае, если семантика модели присутствует в семантическом фоне.

На основе анализа предметной области, а также на основе обоснования выбора модели и архитектуры базы данных были выявлены СУБД, отвечающие назначению и требованиям ИС. Таким образом, СУБД может использовать следующие СУБД: SQLite, MS SQL Server, MySQL, PostgreSQL.

SQLite — это компактная реляционная база данных. Система работает с файлами и предоставляет широкий спектр инструментов для управления базами данных. Доступ осуществляется непосредственно к файлам. SQLite-простая, но мощная база данных с набором технологий, обслуживающих библиотеки.

SQLite хорош для встроенных приложений, если

бол

ее важным является простота переноса веб-приложения, а масштабируемость менее важна. Например, СУБД зарекомендовала себя в разработке однопользовательских приложений, мобильных приложений или игр.

При проектировании ИС, от этой базы данных необходимо отказаться по причинам:

- Является, как правило, многопользовательским, в котором важным компонентом цели является предоставление доступа нескольким пользователям и, возможно, с разными правами доступа;

- СУБД ограничивает операции с большими объемами данных. Допускается только один процесс записи за интервал времени, что является существенным ограничением производительности.

Microsoft SQL Server — система управления реляционными базами данных (СУБД). Microsoft SQL Server-это мощная СУБД с клиент-серверной архитектурой. Данная СУБД позволяет удовлетворить такие важные требования, как возможность репликации, обеспечение параллельной обработки, поддержка баз данных большого объема защиты от несанкционированного доступа за счет интеграции North security и network security. Таким образом, SQL Server является надежной, высокопроизводительной, безопасной и надежной базой данных.

MySQL является наиболее распространенной и полноценной серверной СУБД, использующей клиент — серверную архитектуру.

Серверной части прикладного программного обеспечения находятся на компьютере (сервере), где хранится база данных. Он отвечает на запросы клиентского программного обеспечения, отправляемые по сети, и реализует доступ к базе данных.

Клиентская часть прикладного программного обеспечения передает запросы на сервер и получает данные, отправленные в ответ с сервера баз данных. MySQL — это функциональная, свободно распространяемая база данных.

PostgreSQL — это профессиональный инструмент разработки баз данных из этих СУБД. Он свободно распространяется и полностью совместим со стандартом SQL, чем MySQL. Своевременно в PostgreSQL применяется новая версия стандарта ANSI/ISO для языка SQL.

Из других СУБД PostgreSQL поддерживает объектно-ориентированный подход, помимо реляционного подхода к базам данных. Postgre отличается высокой производительностью, но с помощью простых операций PostgreSQL может замедлять работу сервера и быть медленнее конкурентов, таких как MySQL.

Используйте Postgre, желательно, если в будущем вы планируете перейти на платную базу данных, например, Oracle.

В результате были рассмотрены 4 популярные и мощные системы, реализующие реляционную модель данных: SQLite; MS SQL Server; MySQL; PostgreSQL. Из рассмотренных систем в качестве СУБД может использоваться MS SQL Server или MySQL. Сравнивать данные СУБД невозможно, так как это системы разных классов, применимые для решения задач из разных сегментов. СУБД MS SQL Server больше подходит для проектирования проектов, с размером базы данных, под сотни гигабайт и с тысячами одновременных пользователей.

В итоге было решено, что наиболее подходящим вариантом реализации СУБД для web — ориентированной является использование СУБД MySQL, поскольку это быстрый, надежный и простой в использовании инструмент разработки баз данных.

Для реализации выбра PHPMyAdmin, MySQL Workbench, HeidiSQL, dbForge Studio for MySQL, MyWebSQL и другие. Эти программы называются оболочками для работы с базами данных MySQL.

Для разработки программного обеспечения использовался пакет программ Zend Development Environment от Zend Studio. Zend Studio в целом, это интегрированная среда разработки приложений в основном на

языке PHP, созданная компанией Zend Technologies.

Zend Studio тесно интегрирована с Zend Server и с Zend Framework, что предоставляет удобную систему отладки PHP-приложений, но в данном проекте использовался только пакет Zend Development Environment, который являетсяудобныминструментом для разработчиков, позволяющим писать и отлаживать Веб-приложения, написанные на языках PHP, SQL, HTML, CSS, C, JavaScript, Ruby и других. Приложение имеет встроенное средство для подсветки синтаксиса, а также включает в себя библиотеки, содержащие несколько тысяч функций, которые очень полезны в процессе написания программ [12, 25].

Веб-приложение создано на языке PHP, с использованием объектно-ориентированного программирования (ООП). Модули PHP кода проекта представлены в приложении Б (страница 63) PHP код проекта. В нашем проекте используются модифицированный под приложение класс PHP для работы с MySQL [4, 13].

Объектно-ориентированное программирование является современной технологий программирования. Необходимо отметить, что разработка современных приложений практически невозможна без ООП.

Эта технология обеспечивает структурность программирования, позволяет изменять и дополнять выполняемые функции без значительных модификаций кода, а просто дополняя уже разработанные классы дополнительными методами и свойствами.

Класс — это синтаксическая конструкция языка программирования с закрытым набором переменных и функция, используемых в программах по

пределенным соглашениям, это шаблон объектов определенного типа, на основе которого можно создать множество объектов этого типа. Созданный класс, можно использовать во многих приложениях безизменений структуры класса. При необходимости можно только добавлять новые функции и переменные. Класс и создается для того, чтобы

программист не разбирался во внутренней структуре кода, а брал и использовал готовый код по соглашениям, изложенным в описании классах[4,13].

Разработка программных модулей не велась с нуля, а были существенно модифицированы необходимые программные модули, функции и классы PHP, взятые из сети. Несмотря на это, количество новых доработок, адаптированных под настоящее приложение составляет около 90%.

Для соединения с базой данных был использован класс DB_Class, реализующий основные запросы к базе данных. Однако наличие этого класса не избавляет от необходимости написания кода содержания запросов к базе данных, особенно более сложных запросов поиска по нескольким ключам. Тексты кода этого класса и запросов к базе данных, их краткое описание представлены в приложении Б на странице 63.

Еще одной важной базовой компонентой программного обеспечения является модифицированный класс таблиц [13], измененный и дополненный новыми методами и функциями. Данный класс использовался, как базовый для создания представлений всех таблиц в едином виде, независимо от содержания таблиц. Этот же класс, дополненный новыми методами, использовался для создания унифицированных форм добавления данных в таблицы.

2.2. Проектирование с помощью CASE технологий

Диаграмма вариантов использования (use case diagram, рисунок 2) описывает варианты использования функций системы для различных групп пользователей, другими словами, то, что делает пользователь и что система будет делать в процессе своего функционирования.

Диаграмма вариантов использования состоит, из пользователей часто говорят «актеров» и их «ролей», для которых система производит действие, и собственно действие Use Case, которое описывает то, что пользователь «актер» хочет получить от системы. Актер обозначается значком человечка, а Use Case - овалом.

Каждая такая диаграмма или, как ее обычно называют, каждый Use case — это описание сценария поведения, которому следуют действующие лица (Actors).

Данный тип диаграмм используется при описании процессов автоматизируемой предметной области, определении требований к будущей программной системе. Отражает объекты, как системы, так и предметной области, и выполняемые ими задачи в зависимости от того какой пользователь работает с системой.

Диаграмма вариантов использования (use case diagram), которая описывает функциональное назначение системы с позиций различных групп пользователей, другими словами, то, что система будет делать в процессе своего функционирования в результате действий различных пользователей. Диаграмма вариантов использования является исходным концептуальным представлением или концептуальной моделью системы в процессе ее проектирования и разработки. Диаграммы вариантов использования Use case для системы в целом с позиций администратора системы, сотрудника компании и клиента представлена на рисунках 2,3,4

Любо

й пользователь системы, кроме посетителя, должны быть зарегистрированными пользователями, поэтому для работы с системой должны пройти авторизацию.

Пользователь администратор имеет доступ ко всем данным системы и имеет право на их добавление, удаление, обновление. Администратор системы может параметры поиска и получать отчетную информацию, например, о зарегистрированных сотрудниках и их данных.

Пользователь сотрудник, кроме просмотра общедоступной информации (новостей, мероприятий и пр.) может просматривать данные о сотрудниках, заказах-заявках, нормативные документы компании, отправлять сообщения другим сотрудникам.

Пользователь клиент компании, кроме просмотра общедоступных данных обкомпании, после авторизации, может выбрать услугу компании и произвести заявку, отправить сообщение администратору.

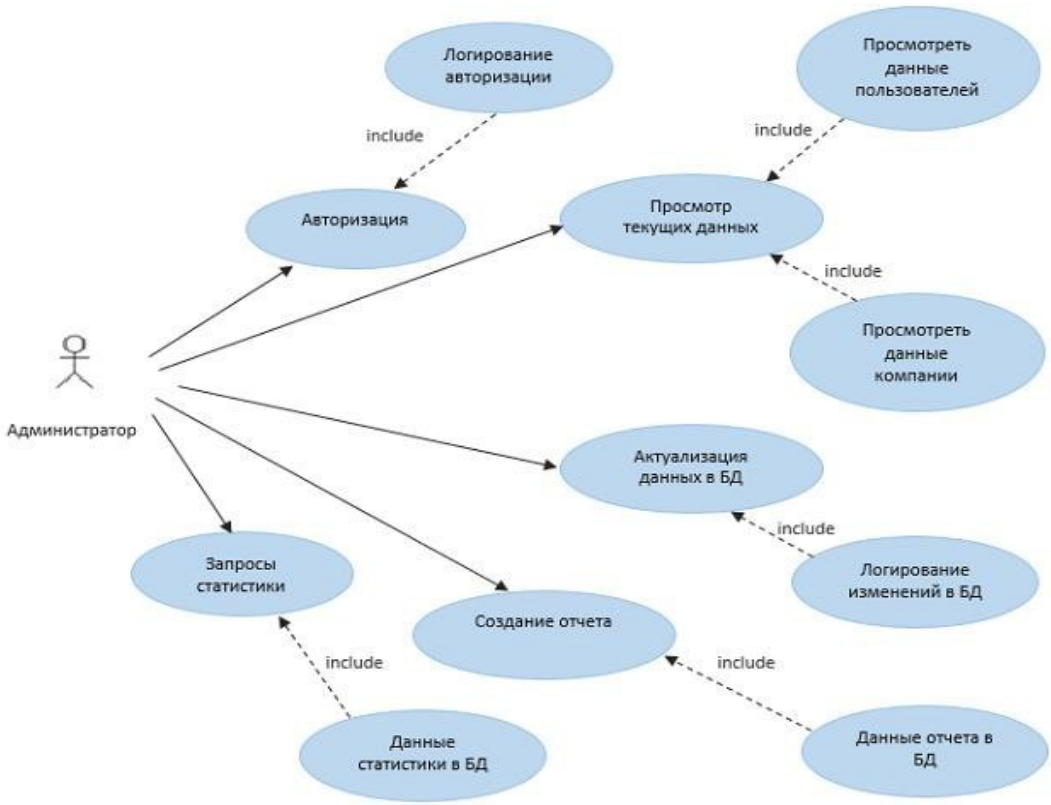


Рисунок 2 — Диаграмма вариантов использования для администратора

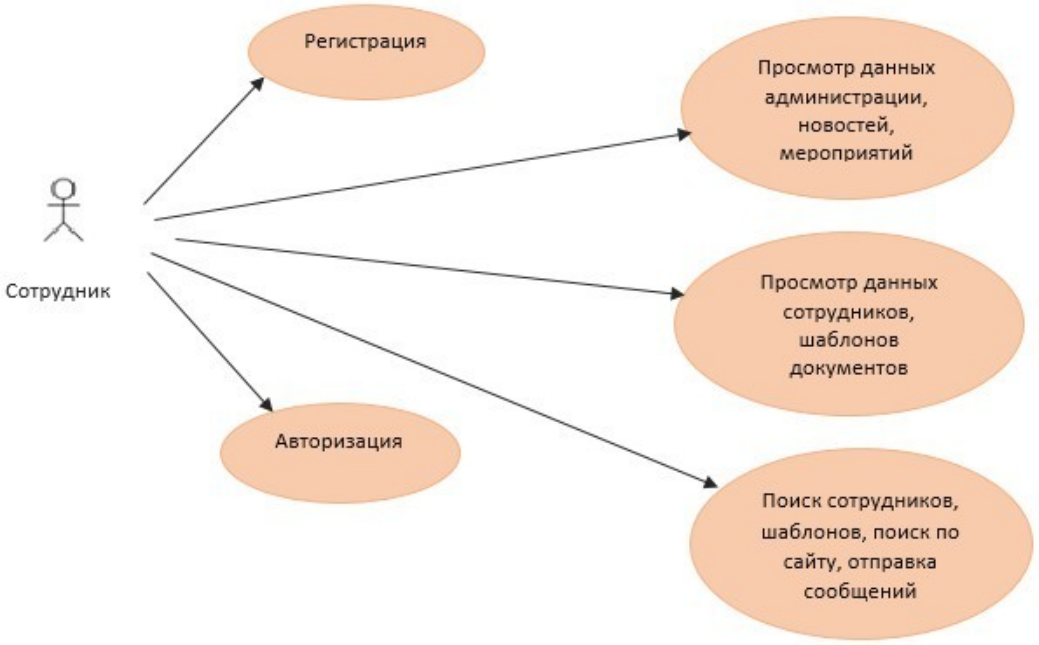


Рисунок 3 — Диаграмма вариантов использования для сотрудника

Рисунок 4 — Диаграмма вариантов использования для клиента

Диаграммы деятельности (activitydiagram) UML отражают аспект реализации варианта использования, включающий действия пользователей (деятельности), действия системы, знаки синхронизации, состояния, а также переходы между ними. Диаграмма деятельности для системы (приведена в приложении В, страница 82) с позиций администратора. Администратор системы кликает по вкладке «Вход для администратора». После этого он должен ввести логин и пароль. Если пароль введен не верно, система не распознает данного пользователя, как администратора и предлагает повторно ввести пароль. Если логин и пароль введены верно, то система предлагает вариант работы с базой данных. Возможны варианты просмотра данных, обновления или удаления данных. Таким образом, например, могут быть введены или удалены данные нового клиента. Для управления данными в БД существуют специальные формы для добавления, удаления, обновления данных независимо от структуры этих данных. Наиболее часто обновляемыми данными для администратора являются данные новостей, мероприятий, клиентов.

Диаграмма состояний описывает возможные последовательности состояний и переходов, которые в совокупности характеризуют поведение системы в течение жизненного цикла.

Диаграмма состояний представляет динамическое поведение сущностей, на основе спецификации их реакции на восприятие некоторых конкретных событий. Диаграммы состояний чаще всего используются для описания поведения отдельных систем и подсистем. Они также могут быть применены для спецификации функциональности экземпляров отдельных классов, т. е. для моделирования всех возможных изменений состояний конкретных объектов.

Диаграмма состояний, это граф специального вида, который служит для представления конечного автомата. Конечный автомат (state machine) - модель для спецификации поведения объекта в форме последовательности его

состояний, которые описывают реакцию объекта на внешние события, выполнение объектом действий, а также изменение его отдельных свойств. Основными понятиями, описывающими конечный автомат, являются состояние и переход. Предполагается, что переход объекта из состояния в состояние происходит мгновенно. Диаграмма состояний для варианта использования приведена в приложении Г страница 83.

Диаграммы последовательности (sequence diagram) UML отражают коммуникационный аспект реализации варианта использования и включают в свой состав объекты и сообщения, между ними и моделирует взаимодействие объектов во времени.

На диаграмме последовательности изображаются объекты, которые участвуют во взаимодействии. Для диаграммы последовательности основным моментом является динамика взаимодействия объектов во времени.

Каждое взаимодействие описывается совокупностью сообщений, которыми участвующие в нем объекты обмениваются между собой. Сообщение представляет собой законченный фрагмент информации. Прием сообщения вызывает выполнение определенных действий объектом, которому сообщение отправлено. Сообщения передают информацию и требуют или предполагают выполнения определенных действий от принимающего объекта. Диаграмма последовательности приведена в приложении Д, страница 84.

Сообщения специального характера, это данные передаваемые системе о логине и пароле пользователя. После обработки этих данных, система отправляет сообщение пользователю о готовности или не готовности работать с пользователем. Основным типом сообщений части, связанной с получением информации являются заложенные в систему SQL- запросы к базе данных.В ответ система отправляет html-страницы с встроенными данными запроса.

2.3. Описание базы данных

Модели баз данных (БД), это визуальное представление структуры данных, решаемой задачи, проекта. Модель БД является основой для разработки и администрирования БД и приложений, создаваемых на их основе.

Методом проектирования БД выбран метод семантического моделирования (сущность-связь) в нотации IDEF1X. Этот метод является частью SADT методологии проектирования бизнес процессов и баз данных.

Этот метод заключается в представлении модели данных в виде диаграммы «сущность-связь». Составными элементами диаграммы являются сущности, атрибуты и связи между сущностями. Построенная модель преобразуется в реляционную модель данных согласно правилам преобразования.

Элементами диаграммы в нотации IDEF1X являются:

- сущности;
- связи;
- атрибуты.

Связи между сущностями являются бинарными отношениями. Различают следующие виды связей:

- идентифицирующие (ИД);
- не идентифицирующие (НИД);
- неспецифические (М: М);
- категориальные (КС).

Идентифицирующая связь, это связь вида «родитель-потомок», где потомок однозначно определяется своей связью с родителем.

Не идентифицирующая связь означают, что у связанного этим отношением экземпляра одной сущности может быть, а может и не быть соответствующего экземпляра другой сущности.

Неспецифическая связь определяется, как связь вида «многие ко многим».

Категориальная связь, это связь между общей сущностью и вариантами ее реализации.

Атрибуты определяют свойства сущности, а значения атрибутов однозначно определяют экземпляр сущности.

Введены следующие виды атрибутов, связанные с поиском и идентификацией экземпляра сущности:

- ключевые;
- не ключевые.

Значение ключевого атрибута однозначно определяет экземпляр сущности. Такой атрибут определяются, как наименование первичного ключа, а его значение является первичным ключом, который отмечают аббревиатурой PK - Primary Key (первичный ключ).

Другой тип ключа, это внешний ключ. Внешний ключ, это атрибут, входящий в ключ родителя и наследуемый потомком.

Ключи отмечаются соответствующими аббревиатурами на диаграммах в прямоугольнике, изображающем сущность (прямоугольник сущности). Внешние ключи помечаются аббревиатурой FK - Foreign Key (внешний ключ). Не ключевые атрибуты перечисляются в виде списка в нижней части прямоугольника сущности. Для идентифицирующих связях все ключи родителя входят в состав ключа потомка, а в не идентифицирующих — в состав не ключевых атрибутов потомка.

В качестве инструмента семантического моделирования базы данных использовалось CASE-средство AllFusion ERwin Data Modeler.

AllFusion ERwin Data Modeler позволяет наглядно отображать сложные структуры данных. Свойства программы:

- синхронизация моделей/баз данных;
- автоматизированное создание структуры базы данных и обратное проектирование;
- экспорт/Импорт структур баз данных;
- программнаясвязьс AllFusion Process Modeler;
- поддерживает форматы данных ведущих производителей СУБД;
- позволяет получить точную и наглядную информацию, где хранятся данные и как получить к ним доступ;
- позволяет, используя визуальные средства, описать структуру БД, затем автоматически сгенерировать схему БД для любого типа СУБД.

На основе анализа предметной области были выделены следующие сущности (таблица 1).

Таблица 1. Сущности и их определения

Имя сущности	**Определение**
Клиент	Данные о клиентах, контактная информация
Сотрудник	Данные о сотрудниках
Заказ-заявка клиента	Данные о заказах-заявках
Позиции заказа	Данные о количестве и стоимости услуги (если
Номенклатура	Данные об услугах компании
Счет на оплату	Данные, необходимые для оплаты

На рисунке 5 представлена структурограмма описания данных с детализацией сущностей «Клиент» и «Сотрудник».

Аналогично может быть представлена структурограмма описания данных с детализацией сущностей «Счет на оплату», «Заявка клиента».

Для экспорта сущностей и атрибутов в среде AllFusion Process Modeler выбираем пункт меню File>Export>ERWin(BPX), в появившемся диалоговом окне «Сохранить как» вводим имя экспортируемого файла «Сущности» и нажимаем кнопку «Сохранить».

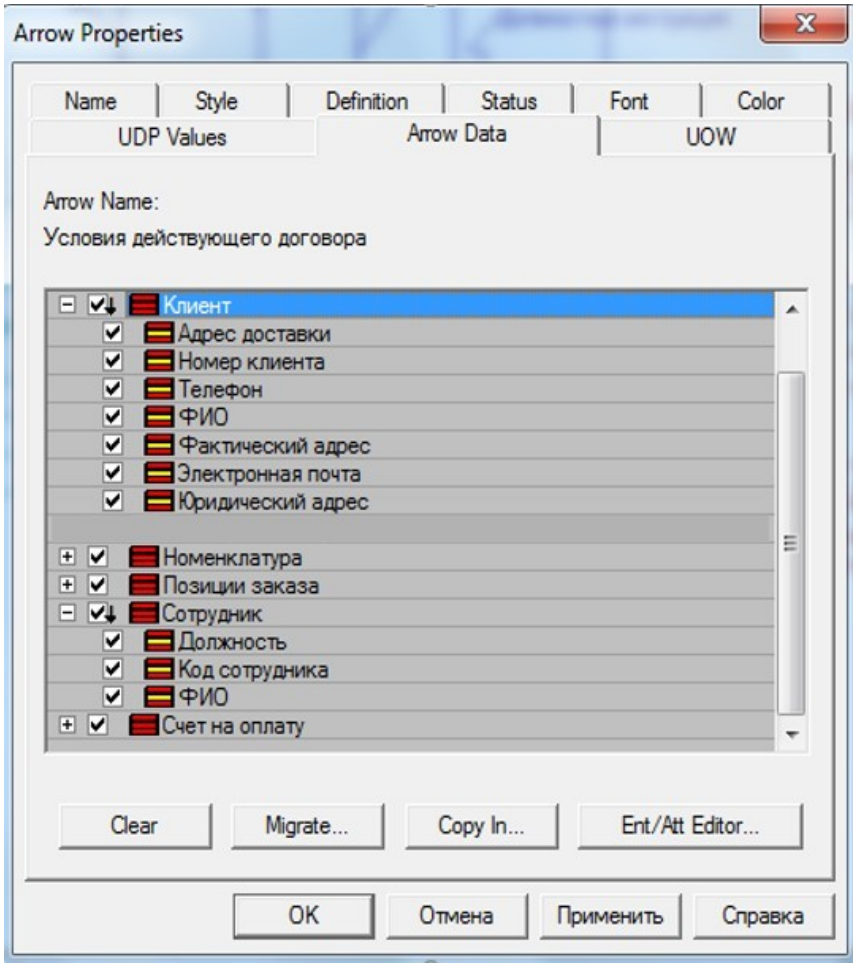


Рисунок 5 — Структурограмма описания данных с детализацией сущностей

«Клиент» и «Сотрудник»

Для импорта сущностей и атрибутов необходимо в среде AllFusion

Data Modeler создать новый проект и выбрать пункт меню File>Import>From AllFusion Process Modeler, в появившемся диалоговом окне "OpenFile" выбрать папку, в которой находится импортируемый файл, выбрать его и нажать кнопку

«Открыть».

На рисунке 6 изображен выбор импортируемых сущностей и атрибутов.

В открывшемся диалоговом окне «AllFusion ERWin Data Modeler» нужно установить флажки для тех сущностей и атрибутов, которые будут импортироваться, и нажать кнопку «Import».После импорта сущностей и атрибутов получилась модель, представленная на рисунке 7. В прямоугольниках отображены основные поля.

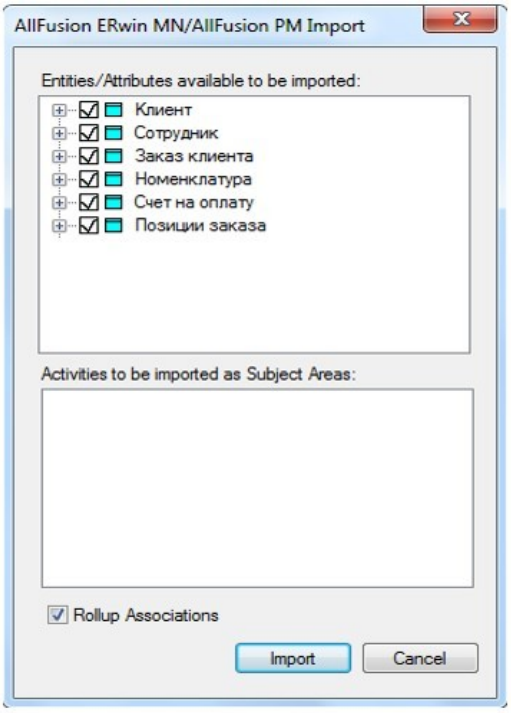


Рисунок 6 — Выбор импортируемых сущностей и атрибутов



Рисунок 7 – Модель данных после импортирования сущностей и атрибуто

Реализация методологии IDEF1X на базе «AllFusion ERWin Data Modeler» имеет те особенности, что основными уровнями представления модели данных являются — логический уровень (LogicalLevel) и физический уровень (PhysicalLevel).

На логическом уровне сохраняется абстрактный взгляд на данные, которые представляются так, как выглядят в реальной предметной области, и называются так, как называются в данной области.

Объекты модели предметной области на логическом уровне, как уже было сказано, называются сущностями и атрибутами.Логический уровень модели данных может быть построен на основе другой модели, например, на основе модели процессов. Логический уровень модели данных не связан с конкретной СУБД. Данные о сущностях и их определения разработанной модели данных, отражены в таблице 2.

Данные о связях между сущностями, в результате многоаспектного анализа предметной области, представлены в таблицах 2 и 3.

Таблица 2. Связи между сущностями: НИД

Родительская сущность	**Дочерняя сущность**	**Имя связи**	**Тип связи**	**Семантика связи от родительской сущности к дочерней**
Клиент	Заказ клиента	КЛ – ЗАК КЛ	НИД 1:М	Отправляет
Сотрудник	Счет на оплату	СОТР – СЧЕТ НА ОПЛ	НИД 1:М	Отправляет
Сотрудник	Заказ клиента	СОТР – ЗАК КЛ	НИД 1:М	Обрабатывает
Заказ клиента	Счет на оплату	ЗАК КЛ – СЧЕТ ОПЛ	НИД 1:М	Включается в

Таблица 3. Связи между сущностями: М:М

Родительская сущность 1	**Дочерняя сущность (имя связи)**	**Родительская сущность 2,3**	**Тип связи**	**Семантика связи**	**Рекомендуемое сокращение дочерней сущности**
Заказ клиента	Позиции заказа	Номенклатура	М:М	Включается в / Входит в	Позиции заказа

В результате была сформирована ER-диаграмма, которая представлена на рисунке 8:

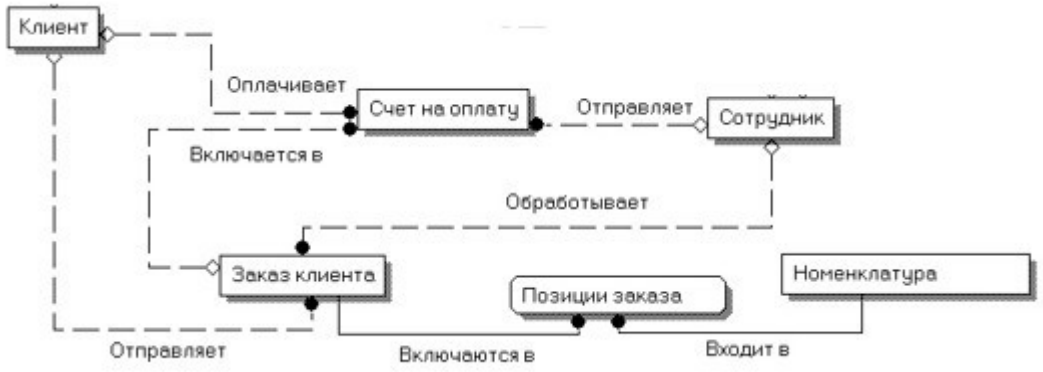


Рисунок 8 — ER-диаграмма модели базы данных подсистемы обработки заявок

Может быть представлена и модель данных, которая включает ключи — КВ-модель, которая кроме сущностей и связей, включает в себя ключевые атрибуты сущностей: первичные — PK и внешние — FK.

В результате анализа предметной области для определения первичных и внешних ключей были определены следующие закономерности:

1. После выбора услуг каждый клиент может отправлять заявку через веб- интерфейс;
2. Каждый клиент обладает уникальным идентификатором (номером);
3. Каждая заявка клиента включает одно или ряд наименований номенклатуры услуг;
4. Каждый заказ обладает уникальным идентификатором (номером);
5. Каждой заявке соответствует счет на оплату;
6. Заявка обрабатывается одним сотрудником;
7. Каждый сотрудник имеет уникальный идентификатор (номер);
8. Сотрудник обрабатывает один или несколько заказов клиента для выполнения;
9. Каждая номенклатура услуги имеет уникальный номер.

Для каждой сущности были определены первичные ключи, на основе следующих требований:

1) атрибуты первичного ключа не могут принимать неопределенных значений;

2) атрибуты первичного ключа должны однозначно идентифицировать экземпляр сущности;

3) количество атрибутов в первичном ключе должно быть минимальным.

Таким образом, может быть сформирована КВ-модель, которую не будем представлять отдельно, так как ключевые атрибуты будут представлены ниже. Наконец наиболее подробная модель с атрибутами модель представляет структуру базы данных: данные в третьей нормальной форме, включает сущности, атрибуты и связи.

В физической модели содержится информация о всех объектах БД. Стандартов на объекты БД не существует, хотя во многом имеется сходство (например, нет стандарта на типы данных), физическая модель определяется в конкретной СУБД. Если в логической модели не имеет значения, какой конкретно тип данных имеет атрибут, то в физической модели важно описать всю информацию о конкретных физических объектах — таблицах, колонках, индексах, процедурах и т. д.

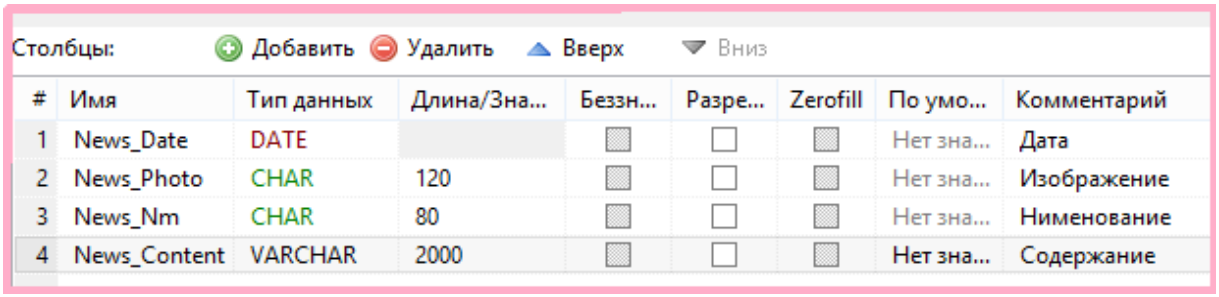
Диаграмму можно построить, например, в MySQL Workbench, предварительно импортировав SQL код создания таблиц базы данных.

В настоящее время базы данных сайта состоит из совокупности связанных и несвязанных таблиц. Несвязанными таблицами являются, например, новости,

мероприятия, шаблоны документов, которые заносятся в БД только администратором системы. Существует множество инструментов разработки баз данных, которые позволяют создавать таблицы базы данных, что называется «на лету». Одним из таких инструментов является пакет Heidy

SQL. Представим структуру таблиц базы данных с помощью этого инструмента. Комментарии каждого поля являются заголовками столбцов таблиц, отображаемых на экране дисплея разработанного приложения. На рисунке 9 представлен состав таблиц базы данных и их смысл.

Рисунок 9 — Таблицы базы данных и их назначение



#	Имя	Тип данных	Длина/Зна...	Беззн...	Разре...	Zerofill	По умо...	Комментарий
1	News_Date	DATE					Нет зна...	Дата
2	News_Photo	CHAR	120				Нет зна...	Изображение
3	News_Nm	CHAR	80				Нет зна...	Именование
4	News_Content	VARCHAR	2000				Нет зна...	Содержание

Рисунок 10 — Таблица базы данных News новостей

#	Имя	Тип данных	Длина/Зна...	Беззн...	Разре...	Zerofill	По умо...	Комментарий
1	**Staff_Id**	**CHAR**	**11**				Нет зна...	**Код (№пропуска)**
2	Staff_FIO	CHAR	120				Нет зна...	ФИО
3	Photo	CHAR	120				Нет зна...	Фото
4	Staff_Posit	CHAR	80				Нет зна...	Специализация
5	Location	CHAR	10				Нет зна...	Местонахождения
6	Phone_Staff	CHAR	120				Нет зна...	Телефон
7	Staff_Email	CHAR	80				Нет зна...	Электр.почта
8	Priem_Time	CHAR	80				Нет зна...	Стаж работы

Рисунок 11 – Таблица базы данных Staff содержит список сотрудников компании

#	Имя	Тип данных	Длина/Зна...	Беззн...	Разре...	Zerofill	По умо...	Комментарий
1	**Driv_Id**	**CHAR**	**11**	☐	☐	☐	Нет зна...	**Код (№пропуска)**
2	Driv_FIO	CHAR	120	☐	☐	☐	Нет зна...	ФИО
3	Photo	CHAR	120	☐	☐	☐	Нет зна...	Фото
4	Driv_Posit	CHAR	80	☐	☐	☐	Нет зна...	Класс водителя
5	Location	CHAR	10	☐	☐	☐	Нет зна...	Местонахождения
6	Driv_Phone	CHAR	120	☐	☐	☐	Нет зна...	Телефон
7	Driv_Email	CHAR	80	☐	☐	☐	Нет зна...	Электр.почта
8	Driv_Time	CHAR	80	☐	☐	☐	Нет зна...	Стаж работы

Рисунок 12 – Таблица базы данных driv содержит данные по водителям

#	Имя	Тип данных	Длина/Зна...	Беззн...	Разре...	Zerofill	По умо...	Комментарий
1	**Instr_Id**	**CHAR**	**10**	☐	☐	☐	Нет зна...	**Номер**
2	Instr_NmL...	CHAR	180	☐	☐	☐	Нет зна...	Наименование
3	Instr_Date	DATE		☐	☐	☐	Нет зна...	Дата

Рисунок 13 — Таблица базы данных Instr содержит шаблоны инструкций

#	Имя	Тип данных	Длина/Зна...	Беззн...	Разре...	Zerofill	По умо...	Комментарий
1	**Staff_Id**	**CHAR**	**10**	☐	☐	☐	Нет зна...	**Код сотрудника**
2	Staff_FIO	CHAR	120	☐	☐	☐	Нет зна...	ФИО Сотрудника
3	Prikaz_Id	CHAR	180	☐	☐	☐	Нет зна...	Номер Приказа
4	Otpusk_Y...	DATE		☐	☐	☐	Нет зна...	Год отпуска
5	Otpusk_B...	DATE		☐	☐	☐	Нет зна...	Дата начала
6	Oypusk_E...	DATE		☐	☐	☐	Нет зна...	Дата Окончания

Рисунок 14 — Таблица базы данных Prikaz содержит приказы и постановления в компании

#	Имя	Тип данных	Длина/Зна...	Беззн...	Разре...	Zerofill	По умо...	Комментарий
1	Party_Date	DATE		☐	☐	☐	Нет зна...	Дата
2	Party_Nm...	CHAR	180	☐	☐	☐	Нет зна...	Наименование
3	Party_Loc...	CHAR	180	☐	☐	☐	Нет зна...	Место проведения

Рисунок 15 — Таблица базы данных Party содержит мероприятия, в которых компания принимает участие

Схема базы данных. Для удобства таблицы базы данных были пронумерованы и их номер входит в название таблицы в базе данных.

- news — новости
- staff —сотрудник
- driv — водители
- instr — инструкции
- prikaz — приказы
- rest — отпуска
- party — мероприятия
- quest — вопросы, сообщения через обратную связь
- services—услуги компании
- clients — зарегистрированные клиенты
- orders — заказы клиентов

party (мероприятия) и их взаимосвязи с другими тремя объектами — staff (сотрудники), quest (вопросы, сообщения через обратную связь). Связь между staff irest,quest можно аргументировать следующимобразом.Каждый сотрудник и водительможет иметь несколько отпусков за каждый годили несколько за год (приложениеЕ страница 85).

2.4 Этапы разработки Веб - сайта

2.4.1Формирование веб-страниц приложения

Интерфейс веб-приложения представляется в виде html страниц со ссылками и кнопками меню. В настоящем проекте только часть страниц имеет статический характер, но другие страницы формируются динамически.

Динамические страницы формируются с использованием php кода, хранящегося в php файлах системы.

Большинство веб-страниц приложения формируется динамически с использованием возможностей языков PHP-CSS.

При формировании текущей веб-страницы сервер использует шаблон требуемой страницы и «закачивает» в него внешние данные, связанные с данной страницей. При этом одна часть данных для формирования веб-страниц находятся в специальных PHP-блоках, контейнерах. Как правило, это очень редко меняющиеся данные, например, шапки таблиц базы данных на русском языке. Другая часть исходных данных «закачивается» сервером из базы данных (вернее сказать при взаимодействии с сервером базы данных). Такой подход позволяет придерживаться базового принципа в разработке программного обеспечения — разделения кода и данных, что в свою очередь предоставляет возможность оперативного изменения данных без изменения Html-php кода веб-страниц.

Таким образом, основные потоки данных в системе можно представить в виде схемы, представленной на рисунке 16.

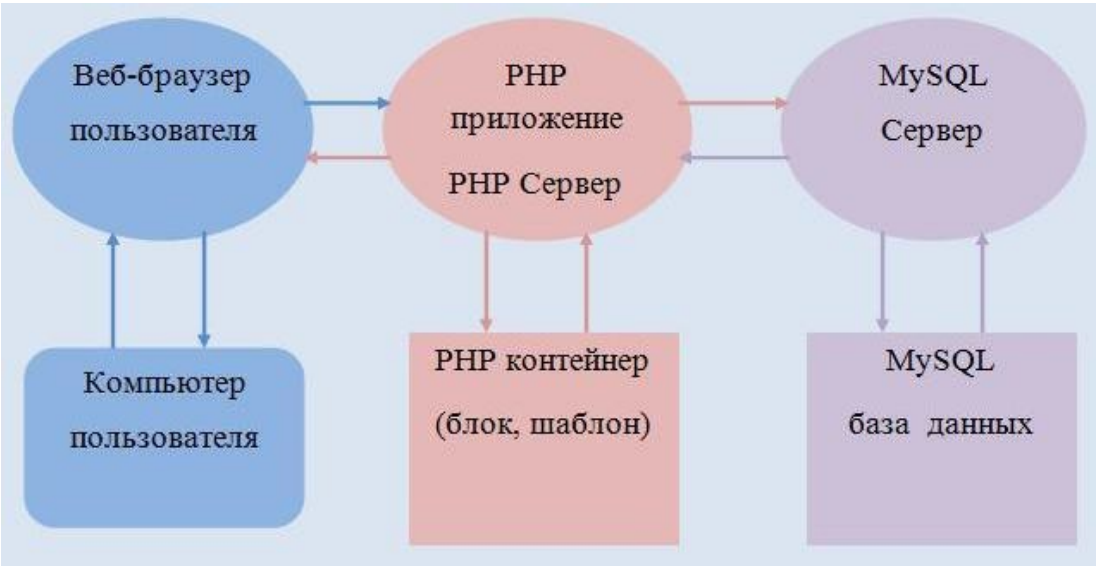


Рисунок 16 — Динамическое формирование веб-страниц

Титульная страница (главная) должна максимально информативно и в сжатом объёме отображать необходимую пользователю информацию о компании. На главной странице расположен логотип и основное меню (для навигации по его структуре), форму аутентификации (входа зарегистрированных пользователей), регистрационную ссылку.

Эта веб-страница состоит из ряда, специальным образом построенных таблиц в терминологии языка HTML, разделяющих экран на отдельные поля и линейного, в виде ленты, меню с возможными вариантами работы с системой (рисунок 17). Структура типовой веб-страницы приведена в ниже. Любая страница ИС состоит из «шапки», Header.php, и статической части или динамической части, формируемой PHP-модулями.



Рисунок 17 — Структура веб-страниц страниц

Такой вариант веб-страницы часто называют «панелью управления». Каждой кнопке меню соответствует ссылка на определенный функциональный модуль системы, определенный функционал ИС.

На основании анализа, проведенного в предыдущей главе, и анализа требований к сайту была разработана обобщенная функциональная структура модулей, представленная на рисунке 18



Рисунок 18 — Структура программных модулей

Краткое назначение составляющих модулей дано ниже.

1)Модуль главного меню обеспечивает взаимодействие пользователей с функциональными модулями системы, которые, в свою очередь, взаимодействуют по определенным соглашениям.

2)Модули статических информационных страниц представляет на экране статическую справочную информацию.

3)Модуль регистрации пользователей обеспечивает регистрацию логина и пароля пользователей для дальнейшего входа и авторизации в системе.

4)Модуль авторизации пользователей обеспечивает аутентификацию пользователей при входе в систему и определяет статус пользователя.

5)Модуль онлайн-регистрации документов обрабатывает формы для ввода данных по документам и загружает документы в файлы и базу

данных. Документами могут быть, например, приказы, инструкции, докладные и пр.

6)Модуль поиска по нескольким ключам осуществляет формирование

запроса на поиск в базе данных.

7)Модуль формирования отчетов осуществляет формирование запроса к БД и вывод на экран отчетов.

8)Модуль формирования табличных представлений генерирует табличное представление данных из базы данных на экране по определенным шаблонам.

9)Модуль формирования форм генерирует формы на экране дисплея для ввода данных в виде, описанном в специальных блоках.

10) Модуль экспорта и импорта должен обеспечить экспорт и импорт данных в БД.

11) Модуль печати выводит данные в отформатированном виде на принтер, либо в графический файл.

12) Модуль формирования запросов к БД обеспечивает взаимодействие с БД.

2.4.2. Создание новых страниц и наполнение их контентом

Рассмотрим последовательность создания новых динамических веб-страниц с определенной структурой в виде таблиц, формируемых php-кодом и в результате обращения к базе данных.

В качестве примера рассмотрим страницу с формированным списком сотрудников. В первую очередь формируется соответствующая таблица базы данных. Для формирования таблицы используем не общепринятый инструмент phpMyAdmin, а программу Heidy SQL (рисунок 19).

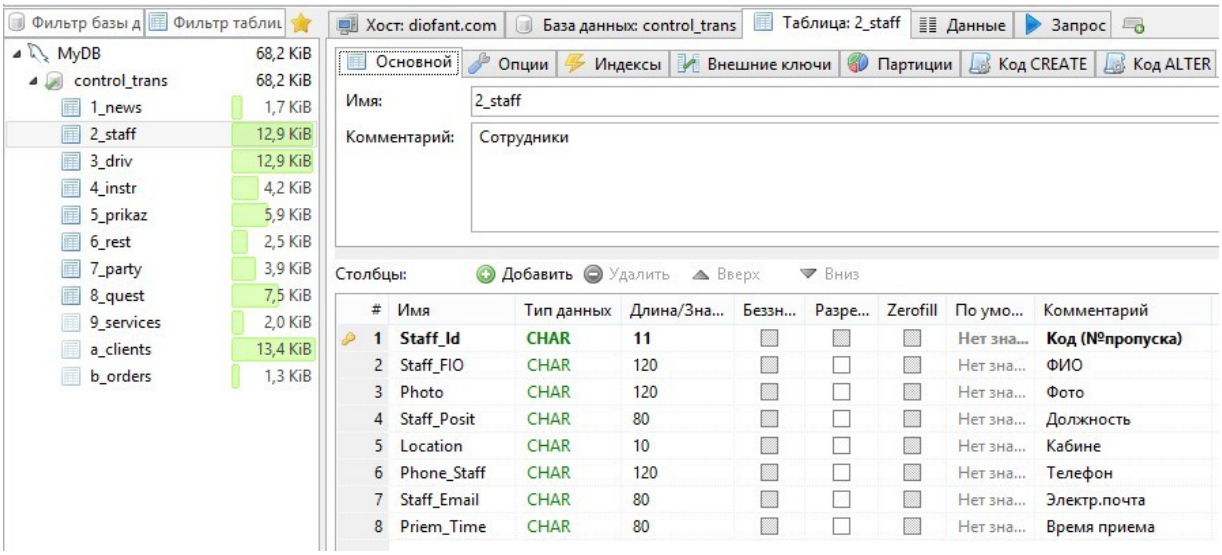


Рисунок 19 — Создается таблица Staff сотрудников

В модуле программного обеспечения с именем Table_Headers.php создается и массив (рисунок 20).

*Table_Headers.php

```php
<?php
$Table_Headers =
Array
(
    '1_News' => Array
        (
            'Дата'                  ,    # [0] => News_Date
            'Фотография'            ,    # [1] => News_Photo
            'Краткое содержание'    ,    # [2] => News_Nm
            'Содержание'            ,    # [3] => News_Content
        ),

    '2_Staff' => Array
        (
            'Идентификатор (код)'   ,    # [0] => Staff_Id
            'ФИО'                   ,    # [1] => Staff_FIO
            'Фотография'            ,    # [2] => Photo
            'Должность'             ,    # [3] => Staff_Posit
            'Дата рождения'         ,    # [4] => Date_Born
            'Телефон'               ,    # [5] => Phone_Staff
            'Электронный адрес'     ,    # [5] => Staff_Email
        ),
```

Рисунок 20 — Блок управления табличным представлением

Если в строке браузера вызвать ../../Control_Trans/Tables_Handler.php, то отобразиться страница ссо списком сотрудников вида (рисунок 21).



Рисунок 21 — Динамически формируемая страница со списком сотрудников

При этом содержание списка может быть изменено (добавление, удаление сотрудников), администратором системы. Для этого при нажатии кнопки «добавить» («обновить», «удалить») появляется соответствующая форма (рисунок 22)

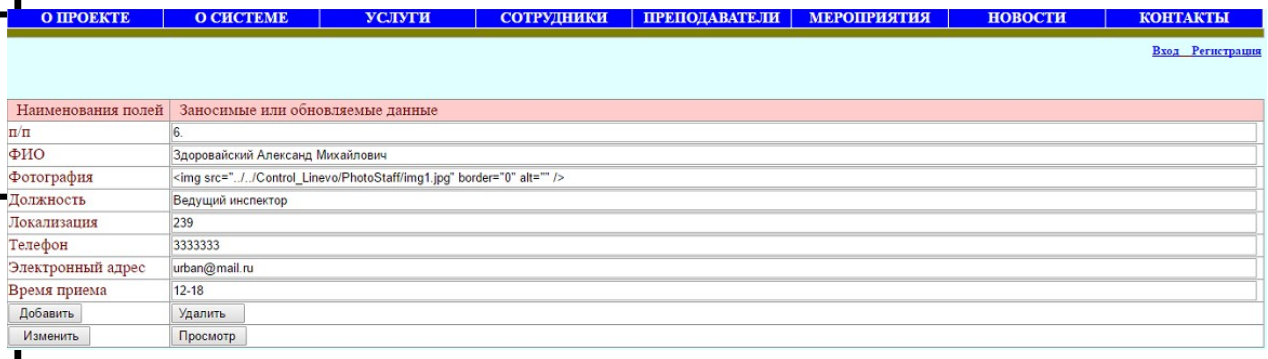


Рисунок 22 — Форма для добавления, обновление списка сотрудников администратором

2.4.3. Создание БД и инсталляция

Рассмотрим подробную процедуру создания базы данных, импорта таблиц базы данных сайта при работе с phpMyAdmin в Денвере.

Помещаем приложение в папку wwwDenver. Другими словами, копируем всю папку Control_Trans в папку wwwDenver. Как правило, конфигурация папок Denver с приложением выглядит следующим образом (рисунок 23).

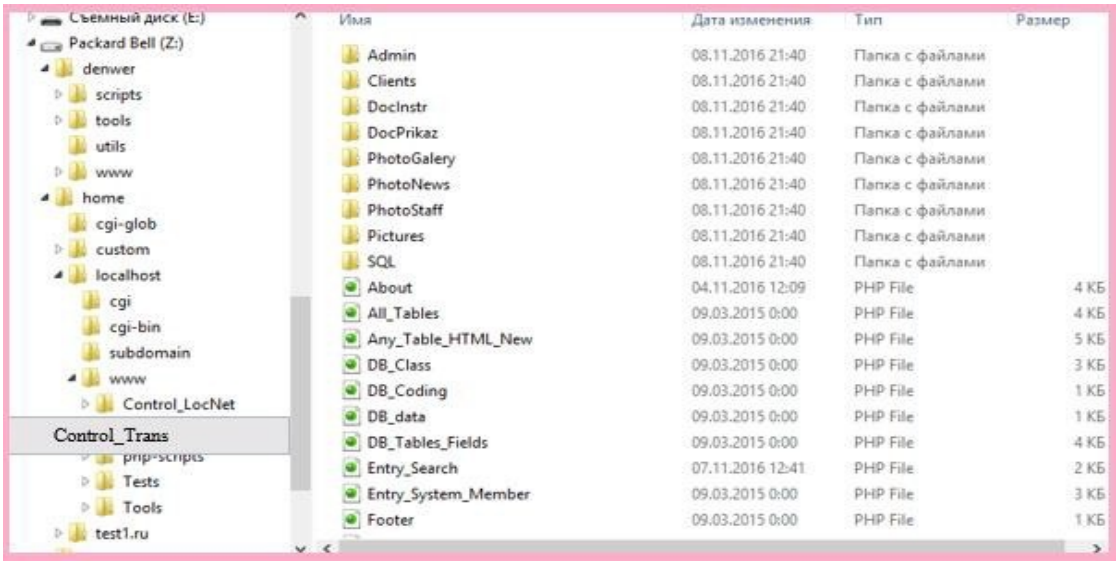


Рисунок 23 — Конфигурация папок Denver с приложением

С правой стороны рисунка можно увидеть структуру приложения — состав папок и файлов. В папке SQL находится SQL файл для создания таблиц базы данных. Имя файла control_trans.sql. Перед использованием этого файла необходимо с помощью phpMyAdmin Денвера создать базу данных control_trans.

На этом этапе или раньше необходимо запустить Денвер. Затем

запуст

ить phpMyAdmin. Запуск phpMyAdmin производится, как правило, следующим образом: ht tp :// lo c alh ost /Tool s/p hp M yAd mi n / i nd ex . php . Нажимаем кнопку меню Databases. На рисунке ниже показано, как создается база данных с нужной кодировкой utf8_general_ci (рисунок 24). Нажимаем вкладку меню Create. База данных с нужным именем создана.



Рисунок 24 — Создание базы данных

После того как база данных создана, можно создать таблицы базы данных со всем содержимым таблиц, используя операцию импорта SQL-файла, находящегося в папке SQL данного приложения. Нажимаем вкладку меню Import. Затем, нажимая кнопку обзор, выбираем папку SQL приложения и файл control_trans.sql (рисунок 25).



Рисунок 25 — Создание таблиц базы данных со всем содержимым, используя операцию импорта SQL-файла, находящегося в папке SQL.

Нажимаем кнопку GO. В результате этой операции появиться сообщение, что таблицы созданы. Подробное описание инсталляции сайта и базы данных дано в приложении Е 35463511

2.5. Итоговая реализация сайта

Предполагается три основных этапа, три технологические последовательности работ:

- инсталляция подсистемы аутентификации пользователей; генерация таблиц базы данных;
- регистрация пользователей, в первую очередь клиентов, наполнение базы данных;
- после этого система готова к штатной работе.

Подсистема аутентификации предназначена для регистрации пользователей, организации распределенного доступа к защищенным папкам, файлам компонентам системы и базе данных системы. Инсталляция систем

ы очень проста. По сути, это само инсталляция, при которой сама подсистема

создает все управляющие файлы. При инсталляции заносится только первоначальный логин и пароль администратора системы, чтобы он мог входить в панель управления подсистемы аутентификации. В дальнейшем, администратор может добавлять других администраторов, имеющих такие же права. В файлах подсистемы храниться информация о логинах и паролях пользователей, защищенных папках, данные о регистрации пользователей.

Кроме администратора предполагается две группы регистрируемых пользователей системы — клиентов и персонала компании. Регистрация пользователей происходит следующим образом. Во-первых, необходимо войти в систему. В настоящее время для входа в систему необходимо в адресной строке браузера набрать следующий адрес (рисунок 26):

http://www.diofant.com/Control_Trans/index.php



Рисунок 26 — Главная страница веб-приложения

Необходимо кликнуть по пункту меню «РЕГИСТРАЦИЯ». Появится следующая для регистрации страница, где предусмотрена ссылка на начало регистрации (рисунок 27). Регистрация подразумевает создание логина и парол

я для дальнейшего входа в систему и занесения некоторых персональных данных.

Рассмотрим формы и процедуру для регистрации сотрудников. Первая форма предназначена для регистрации логина и выбранного пользователем пароля и включает поле для подтверждения выбранного пароля и адрес электронной почты (рисунок 28).

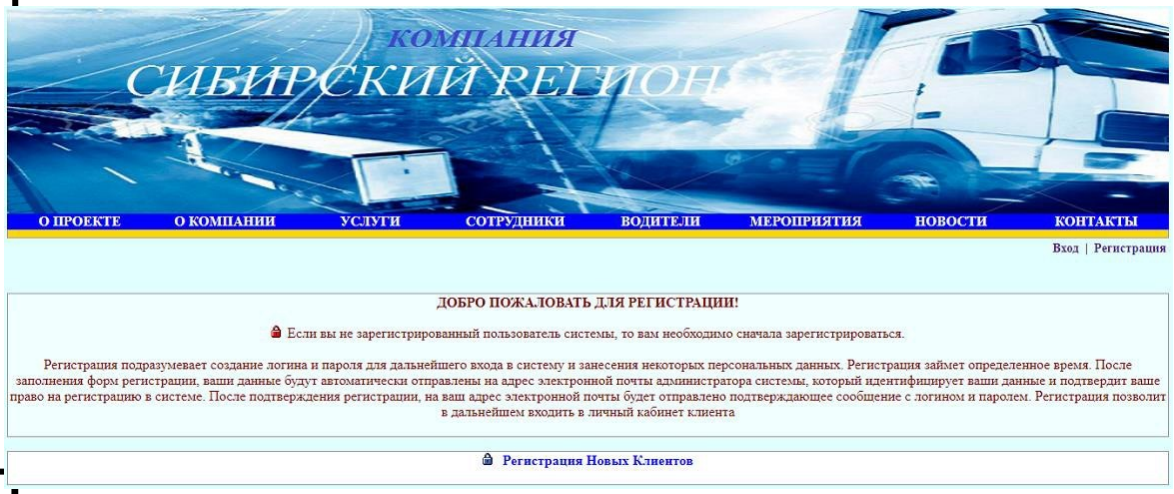


Рисунок 27 — Страница со ссылками на регистрацию клиентов

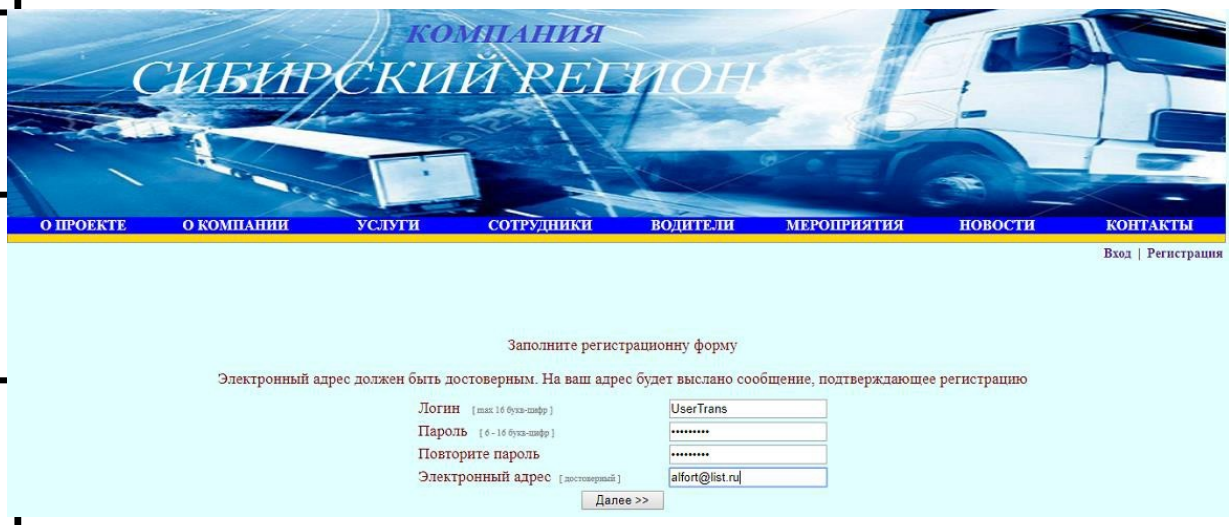


Рисунок 28 — Форма регистрации логина и пароля для входа в систему

После успешного прохождения этого шага в заполнении формы и нажатия клавиши «Далее» внизу формы, происходит переход к следующей форме регистрации, некоторых персональных данных о пользователе (рисунок 29). По персональным данным администратор системы, идентифицирует пользователя ирешает, подтвердитьили нетрегистрациюпользователя нового пользователя сайта.

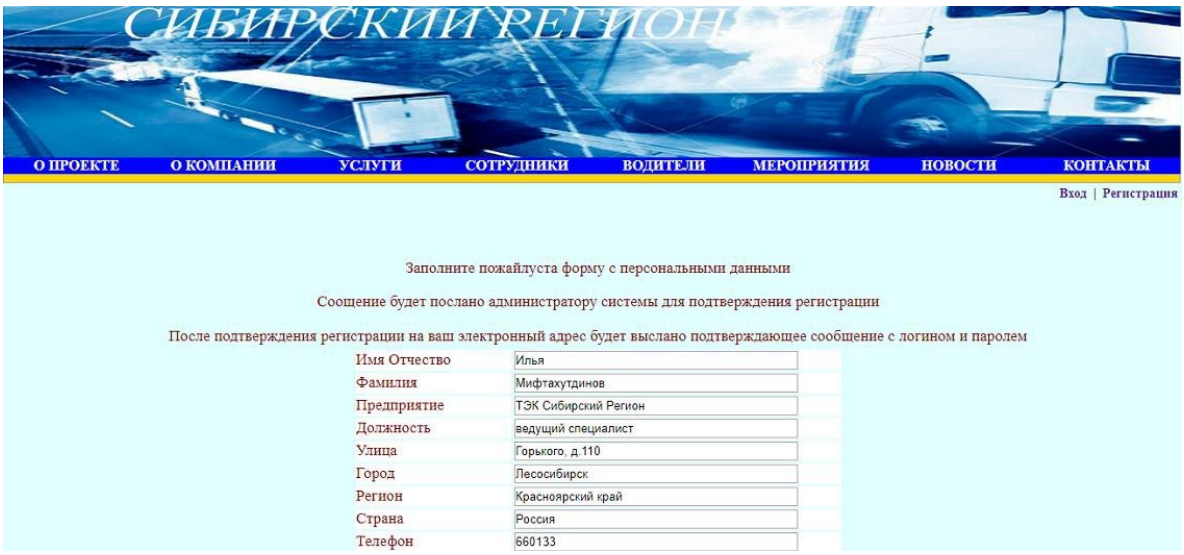


Рисунок 29 — Форма регистрации персональных данных пользователя
После успешного прохождения второго шага в заполнении формы и

нажатия клавиши «Регистрация» внизу формы, данные пользователя автоматически отправляются администратору системы. Администратор систем получает на свой почтовый адрес сообщение, идентифицирует пользователя. В настоящее время реализован вариант «ручной идентификации».

Для того чтобы подтвердить регистрацию пользователя, администратор просто кликает по ссылке, включенной в сообщение. После этого данные заносятся в подсистему аутентификации и на электронный адрес пользователя, который он указал в первой форме регистрации,

отправляется сообщение о его успешной регистрации, а также зарегистрированный логин и пароль.

После регистрации сотрудников можно приступать к штатной работе с системой. Вход в систему для сотрудников и клиентов происходит через ссылку «Вход», затем подменю «Вход для зарегистрированных пользователей». После нажатия этих кнопок, появляется форма авторизации (рисунок 30).

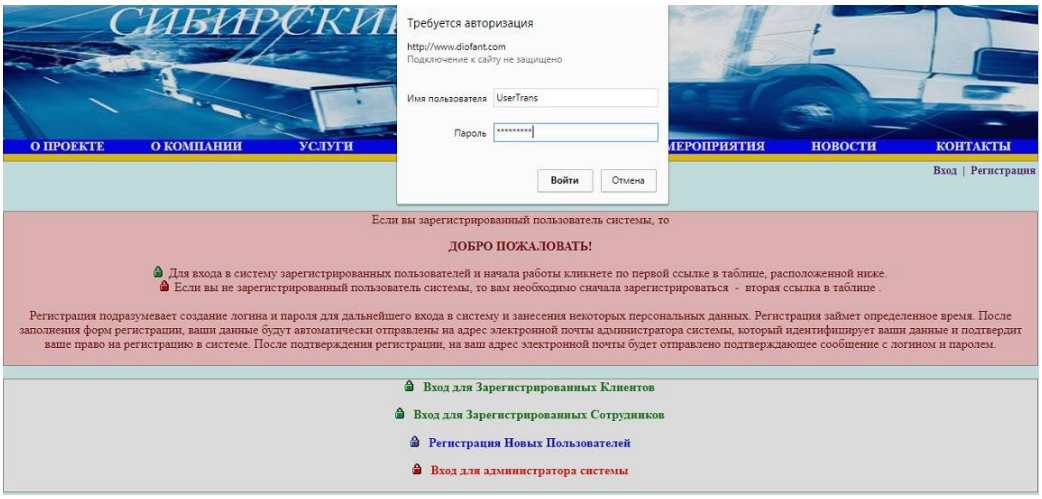


Рисунок 30 — Форма авторизации пользователя с полями логина и пароля После авторизации пользователи входят в личный кабинет (рисунок 31)

Рисунок 31 — После авторизации вход в кабинет со ссылками для вариантов работы

Рассмотрим подробную процедуру оформления заявки зарегистрированным пользователем с логином UserTrans и паролем UserTrans. Кликая, например, «ПРОСМОТР УСЛУГ И ОФОРМЛЕНИЕ ЗАКАЗА ЗАЯВКИ», переходим к выбору вида услуги (рисунок 32).



Рисунок 32 — Список услуг со ссылками на заказ-заявку

При переходе к форме заказа система запоминает код желаемой услуги, отыскивает соответствующую услугу в базе данных системы, а также запоминает данные клиента для оформления заказа-заявки. Код заказа, дата и время заполняются системой автоматически, Таким образом автоматизируется весь процесс оформления заказа на услугу, исключаются ошибки при заполнении полей формы заказа (рисунок 33).



Рисунок 33 — Автоматизированный вариант заполнения формы заказа

После нажатия кнопки «ПОДАТЬ ЗАЯВКУ», система сообщает, что ЗАЯВКА успешно добавлена в базу данных системы и будет обрабатываться сотрудником (рисунок 34).



Код заявки	Идентификатор пользователя	Дата заявки	Код сервиса	Наименование сервиса	Количество часов аренды	Характеристика груза
1496916858	UserTrans	2018-01-08 05:14:18	Serv000004	Грузоперевозки 5 тонн Пятитонник Iveco до 5 т. от 6400 руб.	2	Наименование груза: мебель
1520000826	UserTrans	2018-03-02 08:27:06	Serv000001	Илосос До 5 т от 4600	12	Очистка озера от осадочного ила

Рисунок 34 — Результат заполнения формы заявки заносится в базу данных

Услуги могут быть просмотрены посетителями и клиентами компании на вкладке «О КОМПАНИИ». Если посетитель, не зарегистрированный клиент, попытается заказать одну из услуг, то ему будет предложено зарегистрироваться. Он отправляется на страницу регистрации клиентов

компании.

Другой вариант заказа (не автоматизированный), может быть осуществлен из кабинета пользователя по ссылке 2: «Оформление заказа-заявки». Форма заказа аналогична предыдущей, но поля с незаполненными полями кода заказа и наименованием заказа. Администратор системы ми менеджер видят и могут корректировать форму заказа в виде, представленном на рисунке 35.

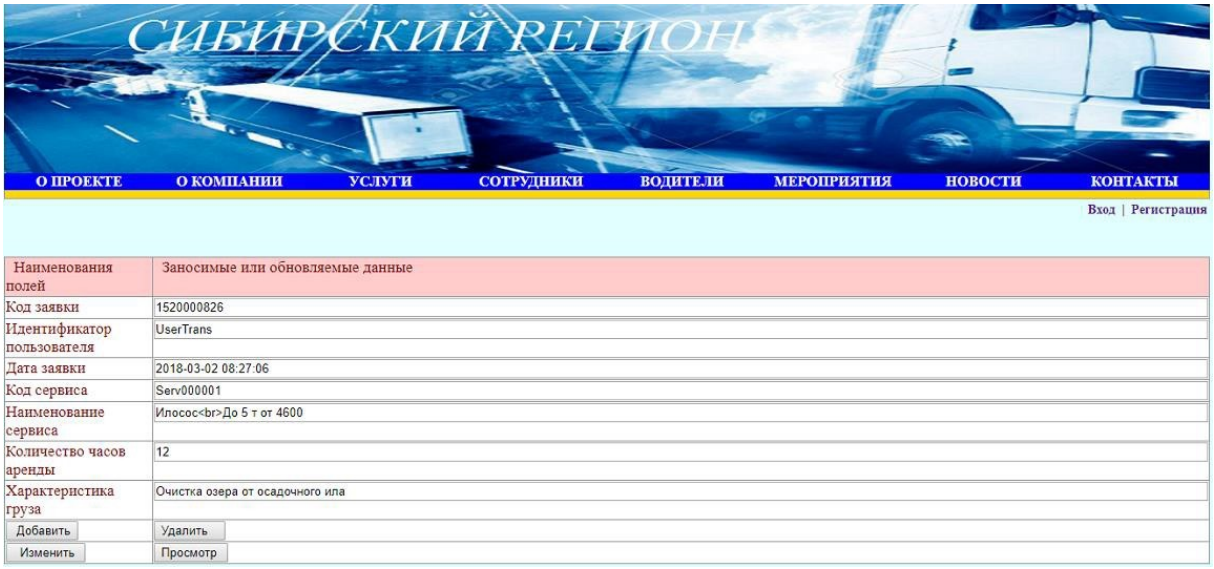


Рисунок 35. Пример заполнения не автоматизированной формы заказа

Поиск информации в кабинете клиента. Имеется ссылка «ПОИСК СОТРУДНИКОВ». Пример формы поиска по фамилии приведен на рисунке 36. Результат поиска приведен на рисунке 37.

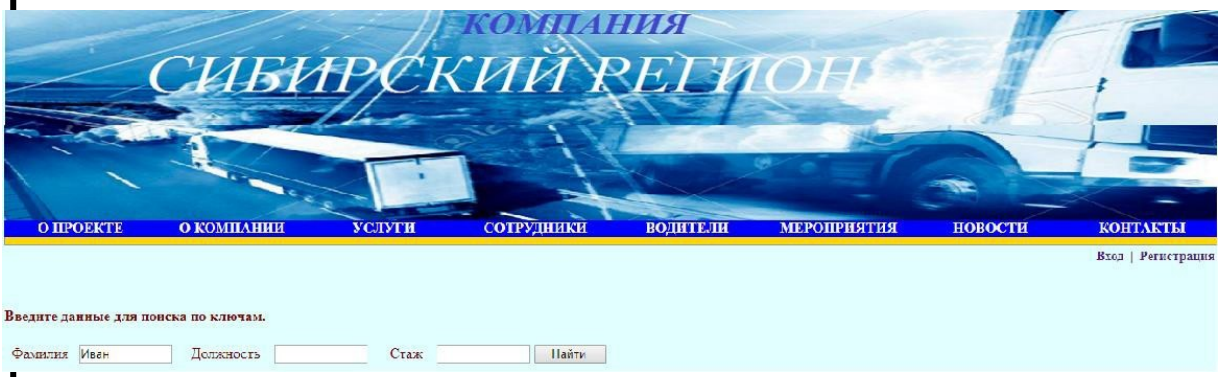


Рисунок 36 — В форме поиска задана часть фамилии «Иван»



Рисунок 37 — В форме поиска задана часть фамилии «Иван»

Сотрудник также входит в свой личный кабинет и может искать таким же образом сотрудников, просматривать и искать заявки по нескольким ключам. Вход сотрудника можно попробовать с логином и паролем StaffTrans.

Администратор системы может осуществлять все функции по управлению данными, находящимися в базе данных. Подобная общая форма управления данными генерируется для любой таблицы базы данных. Возможен ввод определенных ограничений на управления данными со стороны администратора, который определяется на этапе опытной эксплуатации системы.

2.5. Безопасность веб-сайта

Существует множество аспектов в обеспечении безопасности веб-приложений и сайтов. Из всего множества аспектов более подробно остановимся на методах защиты от несанкционированного доступа к файлам и папкам веб-приложения, регистрации и авторизации пользователей, разграничении прав доступа.

В работе используется базовая аутентификация Apache. Базовая аутентификация с использованием свойств сервера Apache является простым и эффективным средством. Для этого создается файл с паролями с помощью программ, предоставляемых сервисом хостинг провайдера, где размещается приложение. Если инсталлировать на Денвере, то необходимо добавлять пользователя вручную. Однако приложение направлено на размещение на хостах общего пользования и могут быть использованы инструменты именно этого хостинг провайдера.

Эта программа использует .htaccess и htpasswd файлы (свойства) сервера Apache. Она позволяет поставить парольную защиту на доступ к файлам и папкам веб-приложения, сайта в визуальном режиме.

Создаются области санкционированного доступа с парольной защитой для администратора, клиентов и сотрудников.

Во втором разделе описаны программные средства, которые используются для проектирования и разработки сайта, проведен анализ возможных СУБД для реализации проекта, обоснован выбор СУБД MySQLдля реализации проекта. В разделе представлена функциональная структура программного обеспечения сайта, описаны используемые инструменты для реализации программного обеспечения. Во второй части раздела дано описание процесса проектирования и реализации базы данных сайта, описана схема взаимодействия программных модулей.

В заключительной части раздела дано описание функционирования сайта и процедуры подачи онлайн заявок клиентами компании, представлены экранные формы.

ЗАКЛЮЧЕНИЕ

Практической целью выпускной квалификационной работы является разработка веб-сайта, веб-приложения компании, занимающейся перевозкой грузов. Сайт предназначен для вывода информации о компании в мировое информационное пространство с целью рекламы услуг компании и расширения круга клиентов компании. Функционал сайта может использоваться сотрудниками компании для получения и обработки онлайн заказов клиентов. Таким образом, сайт может использоваться клиентами и сотрудниками компании. Реализована технология онлайн заказов клиентами компании. Для реализации Веб-приложения были решены следующие задачи:

- изучена предметная область и существующие аналоги;
- изучена архитектура Веб-приложений и методы их разработки;
- разработаны требования к Веб-приложению;
- разработаны модели UML модели предметной области;
- реализовано и протестировано Веб-приложение, включающее базу данных систему и программное обеспечение, реализующее веб- интерфейс пользователей системы.

Веб-приложение выполняет следующие функции:

- обеспечивает регистрацию для сотрудников и потенциальных клиентов;
- предоставляет информацию для клиентов (информацию о компании, спектр предоставляемых услуг и т.д.);
- предоставляет форму для оформления онлайн заказа-заявки для зарегистрированных клиентов;
- обеспечивает возможность для зарегистрированных сотрудников

отправку сообщений друг другу;

– дружественный интерфейс обеспечивает сокращение времени на переговоры за счет предоставление полного объема информации о компании и услугам;

– реализует веб-интерфейс для администратора системы;

– обеспечивает обратную связь с пользователями для отправки сообщений администратору системы об удобстве работы с системой;

– обеспечивать разделение доступа групп пользователей

(администратора, зарегистрированного сотрудника, обычного пользователя, потенциальных клиентов).

В результате работы был создан прототип сайта, веб-приложение, разработана структура базы данных и создана база данных, с использованием современной системы управления базами данных (СУБД MySQL), разработан веб-интерфейс для работы с базой данных для различных групп пользователей с использованием языков HTML, CSS, PHP, SQL.

Использование сайта, функционала информационной системы обеспечивает принципиально новые возможности во взаимодействии сотрудников и клиентов компаниипосредством сети интернет. Для сотрудников компании и клиентов нет необходимости в личном контакте. Со всей доступной информацией компанииклиенты могут познакомиться онлайн, просмотреть услуги, инструкции, найти необходимую информацию по поисковым запросам, оформить заказ заявку на услугу компании.

Цель выпускной квалификационной работы достигнута. Задачи, поставленные в выпускной работе, выполнены в полном объеме.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Абдикеева, Н. М. Корпоративные информационные системы управления: учебник / под науч. ред. Китовой О. В. — Москва: НИЦ ИНФРА-М, 2015.— 464 с.

2. Вендров, А. М. Современные технологии создания программного обеспечения [Электронный ресурс]. — Режим доступа: http://www.jetinfo.ru

3. Википедия Model-View-Controller [Электронный ресурс]. — Режим доступа: http://ru.wikipedia.org/wiki/Model-View-Controller

4. Гвоздева, В. А. Базовые и прикладные информационные технологии [Электронный ресурс] / В. А. Гвоздева. — Москва: ИД ФОРУМ: НИЦ ИНФРА-М, 2014. — 384 с. — Режим доступа: http://znanium.com/bookread.php?book=428860

5. Дейт, К. Дж. Введение в системы баз данных.: Пер. с англ. / К. Дж. Дейт. — Изд. 8-е. — Москва : Издательский дом «Вильямс», 2017. — 1328 с.

6. Дрибас, В. П. Реляционные модели баз данных / В. П. Дрибас. — Москва: Машиностроение, 2005. — 340с.

7. Дронов, В. И. Быстрая разработка современных динамических Web- сайтов на PHP, MySQL, HTML и CSS / В. И. Дронов. — Москва : ДИАЛОГ-МИФИ, 2018. — 464 с.

8. Зандстра, М. PHP4 за 24 часа / М. Зандстра. — Москва : Издательский дом «Вильямс», 2008. — 270с.

9. Зелинский, С. Е. Автоматизация учета персонала: практическое пособие / С. Е. Зелинский. — Киев: «Центр Учебной Литературы», 2003. — 679 с.

10 .Каган, Б. М. Разработка информационных систем / Б. М. Каган, И. Б. Мкртумян. — Москва : Энергоатомиздат, 2012. — 269 с.

11 .Карданская, Н. П. Основы разработки информационных систем / Карданская Н. П. — Москва : Вильямс, 2012. — 287 с.

12 .Карпова, Т. С. Базы данных: модели, разработка, реализация / Т. С. Карпова. — Санкт-Петербург : Питер, 2013. — 350 с.

13 .Карпова, Т. А. Базы данных. Модели, разработка, реализация / Т. А. Карпова. — Москва : Издательский дом «Вильямс», 2001. — 300с.

14 .Коннолли, Т. С. Базы данных. Проектирование, реализация, сопровождение. Теория и практика / Т. С. Коннолли, К. Бегг, А. Страчан.

— Москва : Издательство DiaSoft, 2009. — 460с.

15 .Коровина, К. К. Информационные системы / К. К. Коровина. — Москва : Восход, 2013. — 198 с.

16 .Кормен, Т. А. Алгоритмы: построение и анализ / Т. А. Кормен, Ч. Лейзерсон, Р. Ривест. — Москва : МЦНМО, 2001. — 450с.

17 .Кочерин, Е. А. Современные информационные системы. Особенности и правила проектирования / Е. А. Кочерин.—Санкт-Петербург, 2015. — 289 с. 18.Ульман, Ларри. MySQL [Электронный ресурс] / Ларри Ульман. — Москва : ДМК Пресс, 2007. — 352 с. — Режим доступа: http://www.iprbookshop.ru/7848

19.Мартин, Д. И. Базы данных. Практические методы / Д. И. Мартин. — Москва : Издательство «Мир», 2005. — 580с.

20. Мейер, Д. Теория реляционных баз данных / Д. Мейер. — Москва : Статистика, 2007. — 380с.

21 .Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript / Р. Никсон [и др.]; [пер. с англ. Н. Вильчинский]. — Санкт- Петербург: Питер, 2017. — 496 с.

22 .Олле, Т. Предложения КОДАСИЛ по управлению базами данных / Т. Олле. — Москва : Издательский дом «Вильямс», 2010. — 370с.

23 .Пирогов, В. Ю. Информационные системы и базы данных. Организация и проектирование / В .Ю. Пирогов. — Санкт-Петербург: «БХВ- Петербург», 2012. — 254 с.

24 .Потоцкий, В. К. Технология создания и управления работой

реляционной базы данных на персональной ЭВМ / В. К. Потоцкий. — Москва.

25 .Притыкин, Д. А. BPwin 4.0: пришел, увидел, реорганизовал. Мнение пользователя. [Электронный ресурс]. — Режим доступа: http://info-system.ru/designing/methodology/bpwin/bpwin.html

26. Савельева, Н. В. Основы программирования на PHP: курс лекций.

Учебное пособие для студентов вузов, обучающихся по специальностям в области информационных технологий [Электронный ресурс] / Н. В. Савельева. — Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), 2005. — 264 с. — Режим доступа: http://www.iprbookshop.ru/22429

27.Сайт разработчиков веб-сервера Apache [Электронный ресурс]. — Режим доступа: http://httpd.apache.org

28. Сайт разработчиков RationalSoftware [Электронный ресурс]. — Режим доступа: http://www-01.ibm.com/software/rational

29 .Ташков, П. А. Веб-мастеринг на 100% : HTML, CSS, JavaScript, PHP, CMS, AJAX,, раскрутка / П. А. Ташков. — Санкт-Петербург : Питер, 2010.

30 .Терехов, А. Н. Технология программирования: Учебное пособие / А. Н. Терехов. — Москва: Интернет-университет информационных технологий. — 2007. — 148 с.

31 .Томсон, Л. Д. Разработка Веб-приложений на PHP и MySQL / Л. Д. Томсон, Л. Веллинг. — Москва : Издательство DiaSoft, 2001. — 560с.

32 .Томсон, Л. Д. Разработка Веб-приложений на PHP и MySQL / Л. Д. Томсон, Л. Веллинг. — Санкт-Петербург : ООО «ДиаСофтЮП», 2016. — 848с.

33 .Трофимов, С. А. CASE-технологии: практическая работа в RationalRose.

Визуальное моделирование / С. А. Трофимов. — Москва : Бином-Пресс, 2002. — 288с.

34 .Ульман, Л. PHP и MySQL. Создание интернет-магазинов. / Л. Ульман. — Изд. 3-е. — Москва : Издательский дом «Вильямс», 2017. — 544 с.

35 .Фаулер, М. UML. Основы / М. Фаулер, К. Скотт. — Санкт-Петербург : Символ-Плюс, 2002. — 192 с.

36 .Форта, Б. Освой самостоятельно SQL за 10 минут : Пер. с англ.: Учебное пособие / Б. Форта. — Изд. 2-е. — Москва : Издательский дом «Вильямс», 2017. — 1120 с.

37. Фролов, А. В. Создание Веб-приложений: Практическое руководство / А. В. Фролов, Г. В. Фролов. — Москва : Издательско-торговый дом «Русская редакция», 2001. — 420с.

38. Чертовской, В. Д. Базы и банки данных: Учебное пособие / В. Д. Чертовской; М-во общ. и проф. образования Рос. Федерации. С.-Петерб. ин-т (фил.) Моск. гос. ун-та печати (МГУП). — Москва : Изд-во МГУП, 2001. — 219 с.

39.Чудинов, И. Л. Системы управления базами данных / И. Л. Чудинов. — Москва : Машиностроение, 2011. — 410с.

40. Шафер, Д. Ф. Управление программными проектами: достижение оптимального качества при минимуме затрат : Пер. с англ. / Д. Ф. Шафер.

— Москва : Издательский дом «Вильямс», 2004. — 1136 с.

41. Шполянская, И. Ю. Проектирование информационных систем с использованием UML и RationalRose: методические указания по изучению дисциплины для студентов заочной формы обучения / И. Ю. Шполянская. — Ростов-на-Дону : РГЭУ «РИНХ», 2003. — 45 с.

ПРИЛОЖЕНИЕ А

Коды программ (скриптов)

Невозможно в дипломном проекте описать подробно все модули, скрипты веб-приложения, поэтому остановимся на наиболее важных и интересных программных решениях.

Header.php- код «шапки» приложения.

Каждая страница веб-приложения включает html-css код, так называемой

«шапки», содержащей логотип и меню с вкладками линейного типа. Каждой вкладке главного меню соответствует ссылка на определенный функционал веб-приложения.

```
<html>
<head>
<title>ИС ООО
</title>
<style
  > a
  {
    text-decoration: none; /* Отменяем подчеркивание у ссылки      */
}
AIM
G
    {
border: none;        /* Убираем рамку в изображениях-ссылках */
}
  bo
  dy
  {
    background: #E0FFFF;   /* Цвет фона                    */ color: Maroon;/* Цвет текста        *
    }
td
    {
vertical-align: top; /* Выравнивание по верхнему краю ячеек    */ font-size: 13pt;   /* Размер шриф
}
```

```
</style>
</head>
<body>
<table bgcolor="#FFFFFF" border="0" height=5 width="99%" cellpadding="1"
cellspacing="1"
>
<tr>
<td align="center" colspan="9" bgcolor="Yellow">
<br>
<font color="Maroon" size="4">
<strong>
:: ООО ::
</strong>
</font>
<p><font color="Silver" size="4"><strong>ИНФОРМАЦИОННАЯ
СИСТЕМА КОМПАНИИ</strong></font></p>
<br>


</td>
</tr>
<tr><td width="11%" align="center" bgcolor="blue">
<a href="../../Control_Trans/index.php"><font color="#FFFFFF"
size="3"><strong>О ПРОЕКТЕ</strong></font></a>
</td>


<td width="11%" align="center" bgcolor="blue">
<a href="../../Control_Trans/About.php"><font color="#FFFFFF"
size="3"><strong>О КОМПАНИИ</strong></font></a>
</td>


<td width="11%" align="center" bgcolor="blue">
<a href="../../Control_Trans/Services.php"><font
color="#FFFFFF" size="3"><strong>УСЛУГИ</strong></font></a>
</td>
```

```
<td  width="11%" align="center" bgcolor="blue">
    <a                href="../../Control_Trans/Tables_Handler.php?
FormType=2_staff"><font color="#FFFFFF"
size="3"><strong>СОТРУДНИКИ</strong></font></a>
</td>

    <td  width="11%" align="center" bgcolor="blue">
    <a                href="../../Control_Trans/Tables_Handler.php?
FormType=3_prep"><font color="#FFFFFF"
size="3"><strong>ВОДИТЕЛИ</strong></font></a>
</td>

    <td width="11%" align="center" bgcolor="blue">
<a                    href="#../../Control_Trans/Tables_Handler.php?
FormType=7_party"><font color="#FFFFFF"
size="3"><strong>МЕРОПРИЯТИЯ</strong></font></a>
</td>

    <td  width="11%" align="center" bgcolor="blue">
    <a          href="#../../Control_Trans/News.php"><font
                color="#FFFFFF" size="3"><strong>НОВОСТИ</strong></font></a>
</td>

    <td  width="11%" align="center" bgcolor="blue">
    <a        href="../../Control_Trans/Form_Contact.php"><font
                color="#FFFFFF" size="3"><strong>КОНТАКТЫ</strong></font></a>
</td>
</tr>
</table>

<table  bgcolor="#999999"  height=10  width="99%"  cellspacing="1"
        cellpadding="1" border="0">
<tr>
<td align="left" valign="top" bgcolor="Olive">
</td>
```

```
</tr>
</table>
<style
  >
  h4{
  text-decoration: underline; /* Подчеркивание заголовка */
 }
</style>


<table height=10 width="99%">
<tr>
<td align="right" valign="top">
<h4><a       href="../../Control_Trans/Entry_System_Member.php"><font
             color="blue" size="2">Вход</font></a>
 
<a           href="../../Control_Trans/Register_System.php"><font
             color="blue" size="2">Регистрация</font></a>
</h4>
</td>
</tr>
</table>
</body>
</html>
```

DB_Class.php - класс для работы с базой данных

Для работы с базой данных создается класс работы с базой данных. Не обязательно самому создавать новый класс — их большое множество в интернете. Класс, это набор функций, которые позволяют открывать базу данных, выполнять SQL, запросы к базе данных из php кода. Для того, чтобы получить доступ к базе данных создается модуль, содержащий данные для входа в базу данных (следующий модуль после DB_Class.php), он называется DB_Data.php.

DB_Class.php - класс для работы с базой данных

```
<?php
```

Продолжение ПРИЛОЖЕНИЯ А

```
class MySQL_DB

{

        private $dbObj =
        null; private $result =
        null;
    /* Для конструктора - адрес, имя пользователя, пароль, имя базы данных, порт,

    /*а также кодировку для соединения.

    /* По умолчанию используется utf8 */

         public function     construct($host, $user, $password, $base, $port = null, $charset = 'utf8')

        {

              $this->dbObj = new mysqli($host, $user, $password, $base, $port);

              $this->dbObj->set_charset($charset);

    }

     /*основная и единственная функция, которая выполняет запрос и возвращает результат
его работы*/
    public function query($query)

        {

              if(!$this-
                    >dbObj)
                    returnfalse;
     /
    *очищаемпредыдущийрезультат*/
    if(is_object($this->result))
                    $this->result->free();

     /*выполняем запрос*/

                $this->result = $this->dbObj->query($query);

    /*если есть ошибки - выводим
    их*/ if($this->dbObj->errno)
    #!!!#             die("mysqli error #".$this->dbObj->errno.": ".$this->dbObj-
                  >error); return $this->dbObj->error;
    /*если в результате выполнения запроса (например SELECT...) получены данные
- возвращаем их.
    /* данные всегда возвращаются в массиве, даже если запрос возвращает одну
    запись.*/ if(is_object($this->result))
                {

                    while($row = $this->result->fetch_assoc())

    $data[] =
    $row; return
    $data;
```

Продолжение ПРИЛОЖЕНИЯ А

```
            }

/*если результат отрицательный - возвращаем false
*/ else if($this->result == FALSE)
                returnfalse;



/*если запрос (например UPDATE или INSERT) затронул какие-либо строки -
возвращаем их количество*/
else return $this->dbObj->affected_rows;

        }

}

include 'DB_data.php';

$dbObj = new MySQL_DB(DB_HOST, DB_USER, DB_PWD, DB_NAME);

include 'DB_Coding.php';

?>
```

DB_Data.php – параметры доступа к базе 5данных

```
<?php

##############################################################################

define("DB_HOST",    "localhost"            ); // адрес сервера баз
данных # define("DB_USER",            "auth1722"                        );
// имя пользователя                    # define("DB_PWD",
                        "auth1722"            ); // пароль к бд            #
define("DB_NAME",   "control_trans"          ); // имя базы данных
                        # define("DB_MODE",
                        2); // 0 - в ручную откр/закр,  #
##############################################################################

?>
```

index_member.php

Приведем фрагмент кода, реализующего вход зарегистрированного пользователя в систему (вкладка «ВХОД», кабинет пользователя). После заполнения формы авторизации, система распознает логин и пароль пользователя и если они зарегистрированы в системе, то он входит в модуль /Clients/index_member.php. Ниже приведен фрагмент кода.

/Clients/index_member.php

```
<?php
session_start();                    /* старт сессии и вставляется шапка приложения
include($_SERVER["DOCUMENT_ROOT"]."/Control_Trans/Header.php");
$Status        ='Клиент';
$FormType  ='a_partners';   /* наименование таблицы в базе данных
#
### определяется логин, с которым зашел пользователь
#
$username                    =$_SERVER['REMOTE_USER'];
#
### в переменных сессии запоминается логин и статус пользователя
#
$_SESSION['username'] = $username;
$_SESSION['Status'  ]     = $Status;
#
### пользователь с таким логином ищется в БД, определяются его ФИО и пр..
#
$IndSearch           = '1';
include  ($_SERVER["DOCUMENT_ROOT"]."/Control_Trans/FirstKey_Search.php");
$_SESSION['FIO'     ] = $FIO;
print '
<p></p>
<p></p>
<table bgcolor="#999999" border="0" height=8 width="860" cellpadding="1" cellspacing="1"
align="center">
<tr>
<td align="left"  bgcolor="#FFFFFF">
    ';
print '
<center>
<img src = "../../Control_Trans/Pictures/lock_gre.gif"> 
</center>
    ';
```

```
print "
<center>
<fontcolor='Red' size='5'>КАБИНЕТКЛИЕНТАКОМПАНИИ<br></font>
<font color='Maroon' size='5'>Добро пожаловать <br> ".$FIO.'!</font>
</font>
      ';
?>
<p>
<font color="Maroon" size="5">
<img src = '../../Control_Trans/Pictures/OneClientSmall.jpg'>
<br>
<a href  = "../../Control_Trans/1Key_Search.php?
FormType=b_orders&IndSearch=2&KeyValue=<?php print
 $FIO;?
  >">
                        Просмотр  
                       моих &nbspзаказов
</a>
<!--   //-->
<br>
<img src = '../../Control_Trans/Pictures/Comments.jpg'>
<br>
<a href  = "../../Control_Trans/Services.php">
Просмотр услуг компании и оформление заказа-заявки
</a>
</p>
</font>
```

Переходим по ссылке к оформлению заказа заявки. При этом в переменных сессии имеются все данные пользователя, а по коду сервиса ищется его наименование и поэтому формируется уже готовый вид заявки. Нижеприведенфрагментокода.

Services.php

Продолжение ПРИЛОЖЕНИЯ А

```
<?php
session_start
();
$Status     = $_SESSION['Status'  ];
$username   =
$_SESSION['username'];
require_once 'Header.php';
require_once 'All_Records.php';
$Order_Text ='ПОДАТЬЗАЯВКУ';

if (!$username)

{

   $Order_Text_Head  =  ' (СНАЧАЛА НЕОБХОДИМО ВОЙТИ В СИСТЕМУ
ДЛЯ АВТОРИЗАЦИИ!)';
$Order_Text      = '';

 }

?>

<p></p>

<p></p>

<p></p>

<table bgcolor="#999999" height=20 width="99%" cellspacing="1"
       cellpadding="1" border="0">
<tr>

<td align="left" valign="top" bgcolor="#FFFFFF">

<p></p>

<p></p>

<p></p>

<BR>

<font  color="Maroon" size="4">

<strong>ПОДАТЬ ЗАЯВКУ НА ТИП УСЛУГИ</strong><?= $Order_Text_Head;?
><br><br>

</font>

<?php

$i=0;

foreach ($Id as $element)

    {
```

Продолжение ПРИЛОЖЕНИЯ А

```
        print "
    <font  color='Red' size='4'>
    <center>
    <strong>$Title[$i]</strong>
    <br>
    <a href='Form_Order.php?service=$Id[$i]'>".$Pictures[$i]."
    <br>$Order_Text
    </a>
    </center>
    </
          font
          ><b
          r> ";
          $i++;
          }
    ?>
    <p></p>
    </td>
    </tr>
    </table>
    <?php require_once 'Footer.php'; ?>
```

All_Tables.php

Отображает все таблицы базы данных и их двух первых строк. Эта программы использует программу DB_Tables_Fields.php для получения всех имен доменов, формирует SQL-запрос к базе данных, «скачивает» содержимое первых двух строк каждой таблицы для отображения их для администратора. При этом используется программа (класс) Any_Table_HTML_New.php для формирования табличного представления данных. Кроме этого используются имена полей таблиц на русском языке из файла Table_Headers.php. Коды всех этих модулей представлены ниже.

```
<?php
##########################################################################
# Модуль выводит на экран содержимое базы данных по таблично.           #
# Полностью все данные не выводятся на экран, а только первые строки, чтобы показать#
```

```
# возможные действия с таблицами базы данных.                                    #
#################################################################################
#
require_once            'DB_Tables_Fields.php';  //!!!//

$Arr_acc              =  array();

$Row                  =  array();

$dbtable              =  $FormTypes;

//$head_color   = 'silver';

# Класс для построения таблиц по заголовкам и содержанию таблиц
require_once 'Any_Table_HTML_New.php';
# Класс для работы с базой данных MySQL, описание в теле модуля

# require_once 'DB_Class.php';

# Объект класса работы с базой данных создается внутри модуля DB_Class.php

# $dbObj                =  new MySQL_DB(DB_HOST, DB_USER, DB_PWD, DB_NAME);

###

# Так мы делаем софт более независимым от конкретных таблиц БД

#

# Заголовки всех таблиц веб проекта, которые появляются на экране

include_once 'Table_Headers.php';             //!!!// если уже созданы русские заголовки

#

# Заголовки всех таблиц БД проекта, на английском

# require_once 'Table_Headers_DB.php';        //!!! В этой версии загружаются из БД

#                                             // на лету рограммой DB_Tables_Fields.php
require_once 'Header.php';                    // заголовок для страниц проекта require_once
'FormMsgs.php';                               // все сообщения программ проекта
#

$f=0;    // номер элемента $FormType
foreach ($dbtable as $value) {
#

$headers    = $Table_Headers   [$FormTypes[$f]];// наименования столбцов таблице

$headers_db = $Table_Headers_DB[$FormTypes[$f]];// наименования доменов в таблице

#                                             // БД и <inputname's в формах

# Создаем объекты класса для вывода таблиц, вызываем конструктор
```

Продолжение ПРИЛОЖЕНИЯ А

```
# $handler     = $FormType.'_Handler'; // обработчик настоящей формы
$Table_view =  new HTML_Table($headers , $heard_color);
#
#запрос на выдачу всей таблицы
$QueryResult = $dbObj->query ("SELECT * FROM $dbtable[$f] LIMIT 0,2");
# возвращает массив ассоциативных массивов (см. класс DB_Class.php)
# по сути двумерный массив, поэтому используем конструкцию foreach
# это можно увидеть, убрав знак # перед print см. ниже. Таким образом
# формируются массивы для вывода на экран для каждой таблицы
#
if ($QueryResult==false)
  {
   $msg = 'Нет записей в базе данных!';
}
else
  {
   $NA  = count($QueryResult);
   //print '$NA='.$NA;
   //print '<pre>'; print_r($QueryResult); print '</pre>';
  }
foreach ($QueryResult as $row    => $Arr_acc)
      {
       $i=0;
       foreach ($headers_db as $element)
             {
               $Row = array_merge($Row, array($headers[$i] => $Arr_acc[$headers_db[$i]]) );
               $i++;
               }
       $Table_view   ->  AddRowAssArr($Row);
       }
```

```
print "<p></p><font color= \"blue\" ><strong>$All_Tables_Names[$f]</strong></font>"
    ."$Comment_to_Table_up[$f]";
     $Table_view      ->  PrintArr();
print "<a href=\"Tables_Handler.php?FormType=$FormTypes[$f]\">
      Все данные Таблицы \"$All_Tables_Names[$f]\"</a>"
    .'<p></p>';
//print "<a href=\"$FormType[$f]_Handler.php?FormType=$FormType[$f]\">

//      Все данные Таблицы \"$All_Tables_Names[$f]\"</a>"
#
$f++;                          // новое значение $FormType, новая таблица
}
#
include 'Footer.php';            // стандартная нижняя строка каждой страницы
#
?>
```

DB_Tables_Fields.php

```
<?php
#################################################################
# Пример использования SHOWCOLUMNS и SHOWCOLUMNS    #
# Скачивает имена всех таблиц и доменов БД                      #
# Дополнительные выводы проставляются для отладки изменений#
#################################################################
#
require_once       'DB_Class.php';
$Arr_acc_tables    = array();
$Arr_acc_fields    = array();
$Tables            = array();
$Fields            = array();
$Fields_1          = array();
$Work              = array();
$Table_Headers_DB  = array();
```

Продолжение ПРИЛОЖЕНИЯ А

```
$SQL_String1       = array();
$SQL_String2       = array();
$Tables_DB_Name     = 'Tables_in_'.DB_NAME;
#
$QueryResult = $dbObj->query("SHOW TABLES");
# ассоциированный массив с именами таблиц
# Array
# (
#    [0] => Array
#        (
#            [Tables_in_control_clients_adv] => Clients
#        )
#...
#    print '<pre>';
#    print_r($QueryResult);
#    print '</pre>';
#
$i=0;
foreach ($QueryResult as $row_tables    => $Arr_acc_tables)
      {
        $Tables[$i]   = $Arr_acc_tables[$Tables_DB_Name];
        $QueryResult  = $dbObj->query("SHOW COLUMNS FROM $Tables[$i]");
#
#
# Array
# (
#     [0] => Array
#         (
#             [Field] => Id_Client
#             [Type] => char(10)
```

```
#           [Null] => NO
#           [Key] => PRI
#           [Default] =>
#           [Extra] =>
#        )
        $j=0;
              $string ='';
              $string1='';
              $string2='';
        foreach ($QueryResult as $row_fields    => $Arr_acc_fields)
                    {
              $Fields[$i][$j]   = $Arr_acc_fields[Field];
              $Fields_1  [$j]   = $Arr_acc_fields[Field];
              $string.= ','  .$Fields[$i][$j];
              $string1.="','"  .$Fields[$i][$j];
              $string2.="',$".$Fields[$i][$j];
              $j++;
                    }
   //     print '<br>'.$Tables[$i].$string;
   //     print '<br>'.$Tables[$i].$string1;
   //     print '<br>'.$Tables[$i].$string2;
    //      print '<pre>';
    //      print_r($QueryResult);
    //      print '</pre>';
              $string1          = substr($string1,1);
              $string2          = substr($string2,1);
              $Table_Headers_DB         =     array_merge($Table_Headers_DB,array($Tables[$i]
                                              =>
$Fields_1));
              $SQL_String1      = array_merge($SQL_String1     ,array($Tables[$i] => $string1 ));
              $SQL_String2      = array_merge($SQL_String2     ,array($Tables[$i] => $string2 ));
```

```
        $Fields_1         = $Work;
        $i++;
}
$FormTypes=$Tables;
### Эту часть только при первом запуске, пока не созданы заголовки Table_Headers.php
include_once 'Table_Headers.php';
if (!$Table_Headers)
  {
   $Table_Headers   =$Table_Headers_DB;
    print '<br>***** $Table_Headers *****';

   print '<pre>';
    print_r($Table_Header
    s); print '</pre>';
  }
//print '<br>***** $Fields *****';
//print '<pre>';
//print_r($Fields);
//print '</pre>';
//print '<br>***** $Table_Headers_DB *****';
//print '<pre>';
//print_r($Table_Headers_DB);
//print '</pre>';
//print '<pre>';
//print_r($SQL_String1);
//print '</pre>';
//print '<pre>';
//print_r($SQL_String2);
//print '</pre>';
#
```

Any_Table_HTML_New.php

При обращению к этому классу, сначала формируется ассоциированный массив

типа (заголовок домена таблицы на русском языке =>значение поля), для каждого столбца
$Table_view ->AddRowAssArr($Row), а затем производится вывод на экран $Table_view ->PrintArr().

```
<?php
class Any_Table
{
var $table_array =
array(); var $headers
=
array(); var $domns;
function  Any_Table ($headers)
{
$this->headers = $headers;
$this->domns   = count($headers);
}

function AddRow ($row)
{
If  ( count ($row) != $this-
>domns) return false;
array_push( $this -> table_array,
$row); return true;
}
function SetArr($Arr)
{
$this -> table_array = $Arr;
return true;
}
function  AddRowAssArr ($row_ass)
{
$row=array();
foreach ($this -> headers as $header)
{ if (!isset($row_ass[$header]))
```

Продолжение ПРИЛОЖЕНИЯ А

```
                    $row_ass[$header]='';
                $row[ ]=$row_ass[$header];
            }
            array_push( $this -> table_array,  $row); return
            true;
            }
    function PrintArr ()
            {
    print "<pre>";
    foreach ($this -> headers as $header)
        print '  '."<strong>$header</strong>".' '; print "\n";
    foreach ($this -> table_array as $string)
    {
        foreach ($string as $elemnt)
                print '   '."$elemnt".'   ';
        print "\n";
        }
            }
    }
class HTML_Table extends Any_Table
{
    var $bgcolor;
    var $bgcolor_h = "#FFCCCC"; var
    $bgcolor_c = "#FFFFFF"; var
    $cellpadding = "1";
    function HTML_Table($headers, $bg='#ffffff' )
            {
            Any_Table::Any_Table($headers);
        $this->bgcolor=$bg;
        }
     function SetCellPadding ($padding)
        {
            $this -> cellpadding = $padding;
```

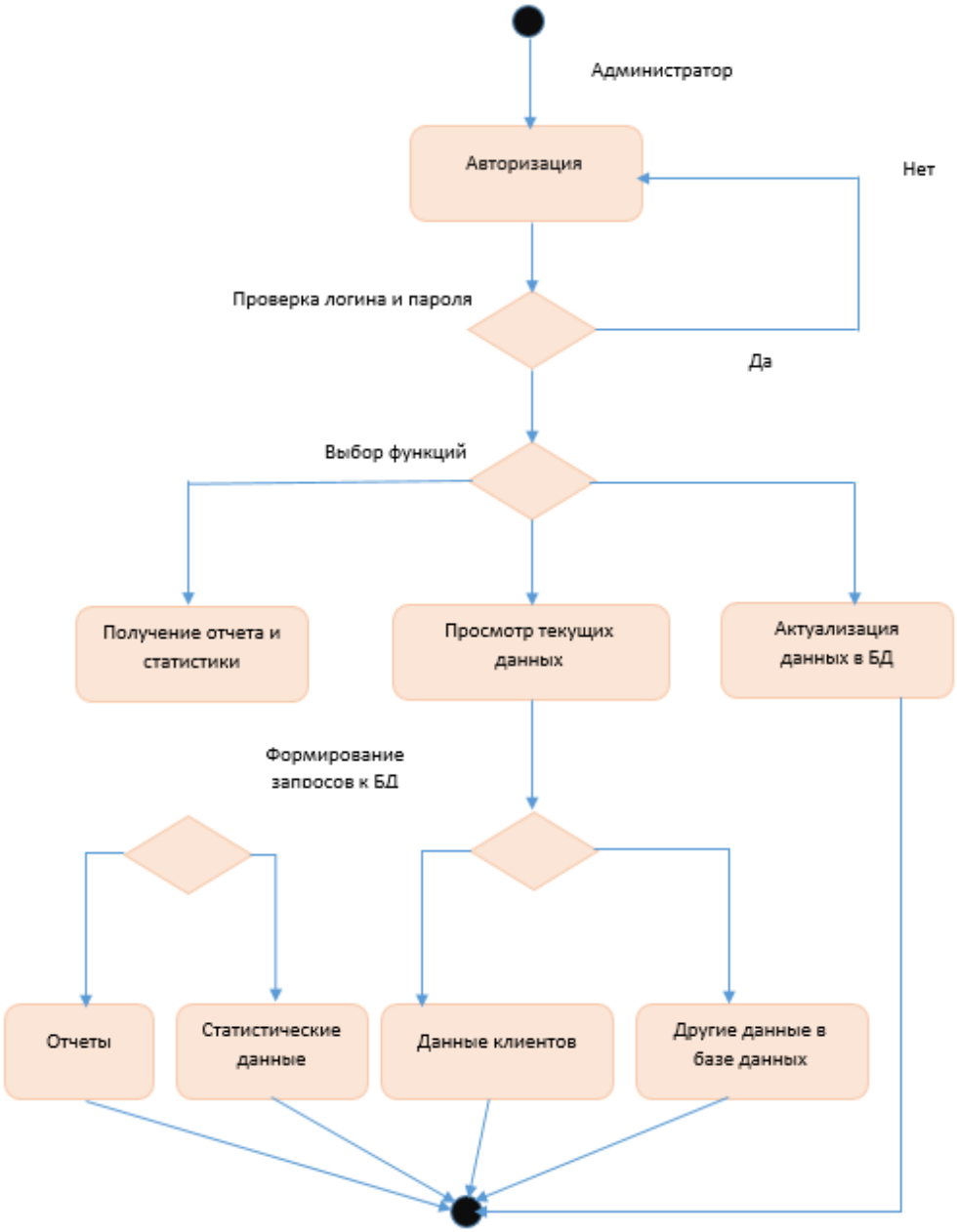
```
        }
    function PrintArr()
        {
            print "<table bgcolor=\"#999999\" cellspacing=\"1\"  cellpadding=\"$this-
>cellpadding\" border=0>";
            print "<tr>";
            foreach ($this -> headers as $header)
           print "<td align=\"center\" bgcolor=\"$this->bgcolor_h\">$header</td>";
        print "</tr>";
        foreach ($this -> table_array as $row=>$cells)
            {
             print "<tr>";
           foreach ($cells as $elemnt)
              print "<td align='center' bgcolor=\"$this->bgcolor_c\" >$elemnt</td>";
           print "</tr>";
            }
        print '</table>';
         }
    function OutForm($handler)
       {
        $width_h = array(170, 360);
          print "<form name='uniform' action=\"$handler\" method=post>"; print
          "<table width=\"99%\" bgcolor=\"#999999\" cellspacing=\"1\"
cellpadding=\"$this->cellpadding\" border=0>";
          print "<tr>";
          $i=0;
            foreach ($this -> headers as $header){
                print "<td width='".$width_h[$i]."' align=\"left\" bgcolor=\"$this-
>bgcolor_h\"> $header</td>";
                $i++;
               }
        print "</tr>";
        $i=0;
```

Продолжение ПРИЛОЖЕНИЯ А

```
                foreach ($this -> table_array as $row=>$cells)
                    {
                     print "<tr>";
                   foreach ($cells as $elemnt)
                        {
                         print "<td width='".$width_h[$i]."' align='left' bgcolor=\"$this->bgcolor_c\"
>$elemnt</
td>";
                        if ($i==0) $i++; else $i--;
                        }
                   print "</tr>";
                      }
                print '</table>';
                print '</form>';
    }
        }
         ?>
```

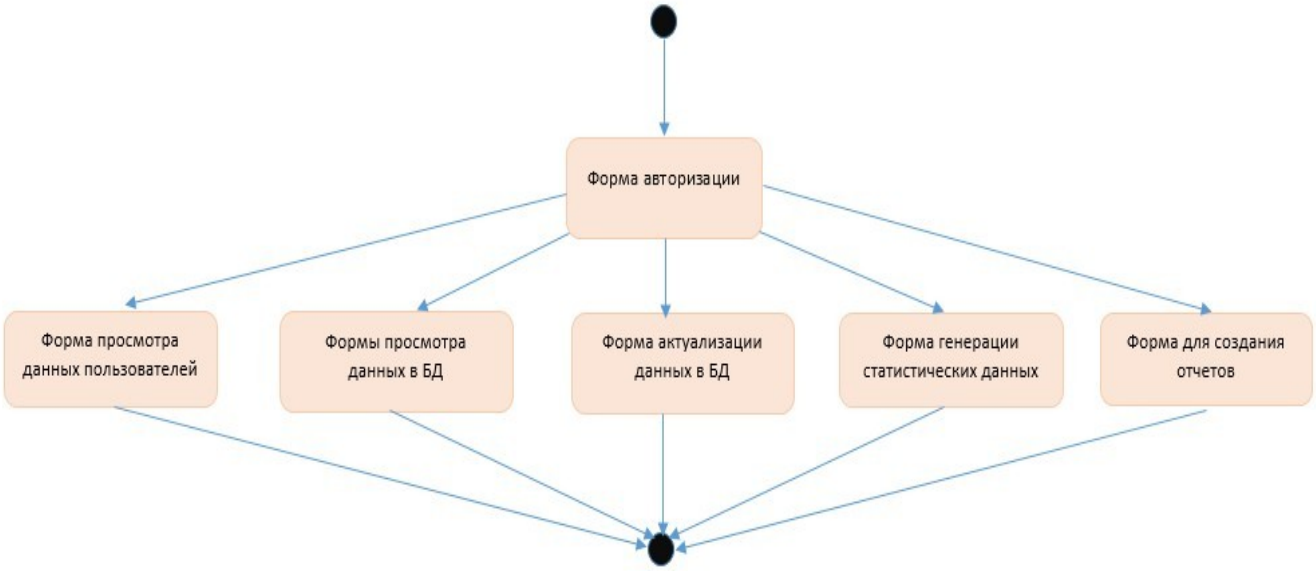
ПРИЛОЖЕНИЕ Б

Диаграмма деятельности (Activity)



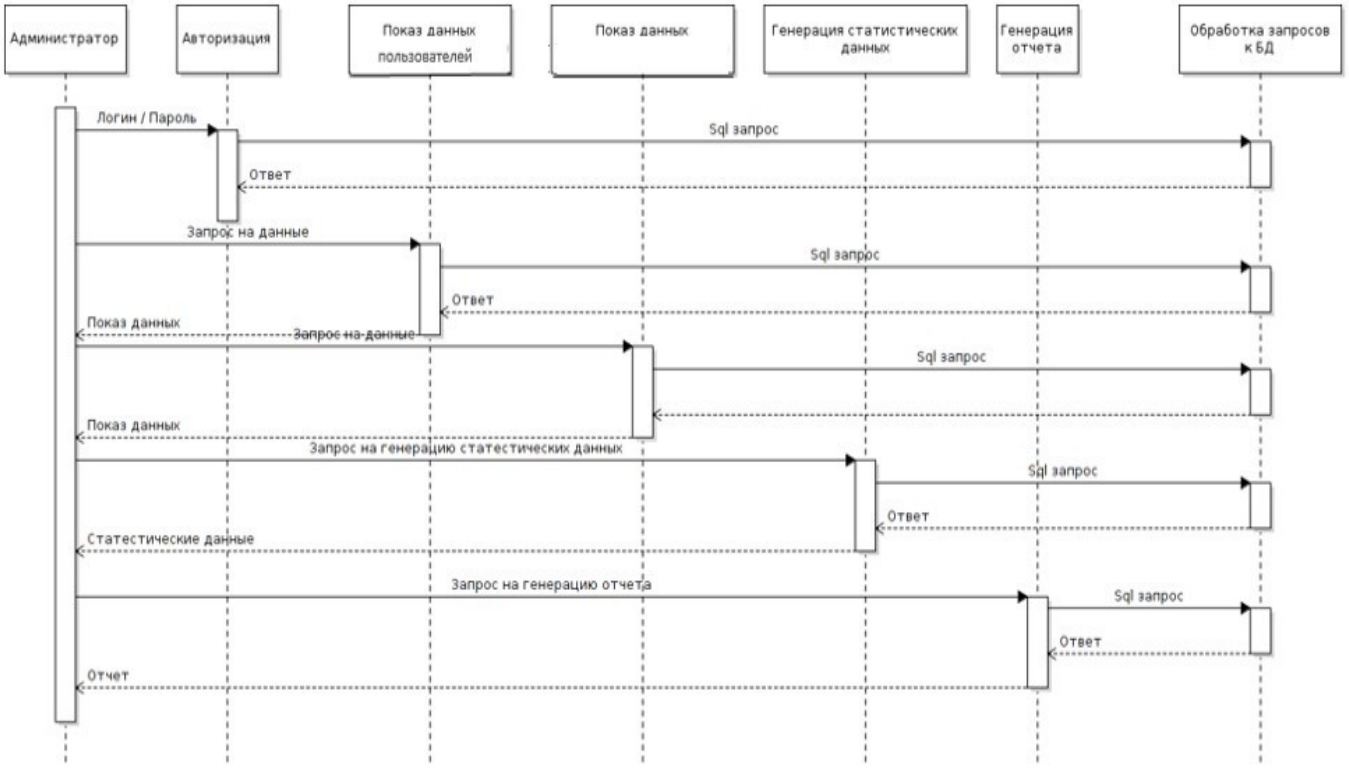
ПРИЛОЖЕНИЕ В

Диаграмма состояний для варианта использования



ПРИЛОЖЕНИЕ Г

Диаграмма последовательностей для варианта использования «Авторизация»



ПРИЛОЖЕНИЕ Д

Схема базы данных

