МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра Компьютерных технологий

«Допущен к защите»
Заведующий кафедрой
д.ф-м.н.,профессор Куралбаев З.К

_____ «_____»_____**20__ г.**
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: «Разработка современной системы управления базами данных для CMS RIX»

Специальность 5В070400 - «Вычислительная техника и программное обеспечение»

Выполнил (а) Алипбек А.С ВТ-10-5

Научный руководитель Кожамбердиева М.И к.п.н., доцент

Консультанты:
по экономической части:
Еркешева З.Д старший преподаватель

Еркешева «20» мая 2014 г.
(подпись)

по безопасности жизнедеятельности:
Приходько Н.Г д.х.н., профессор

«26» 05 2014 г.
(подпись)

по применению вычислительной техники:
Кожамбердиева М.И к.п.н., доцент

«19» 05 2014 г.
(подпись)

Нормоконтролер: Тусупов Д.М старший преподаватель

«30» мая 2014 г.
(подпись)

Рецензент: Шайхин А.К к.ф.н., доцент

«30» мая 2014 г.
(подпись)

Алматы 2014 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет Информационных технологий
Специальность Вычислительная техника и программное обеспечение
Кафедра Компьютерных технологий

ЗАДАНИЕ
на выполнение дипломного проекта

Студент Алибек Алия Сапарбекқызы
(фамилия, имя, отчество)
Тема проекта Разработка современной системы управления базами данных для CMS RIX

утверждена приказом ректора № 115 от «24» сентября 2013 г.
Срок сдачи законченной работы «___» июня 2014 г.
Исходные данные к проекту требуемые параметры результатов проектирования (исследования) и исходные данные объекта
Разработка современной системы управления базами данных для CMS RIX CMS RIX представляет собой систему управления содержимым сайтов.

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:
В дипломной работе поставлена задача разработки системы управления базами данных для CMS RIX, которая позволит отображать используемые в CMS RIX баз данных, их содержимое - таблицы и структуру самих таблиц, редактирование и удаление, сохранение баз данных и их элементов. Система должна удовлетворять поставленным требованиям, включать в себя лучшие качества уже имеющихся систем.
Реализация системы информационной безопасности
охрана безопасности и жизнедеятельности

Перечень графического материала (с точным указанием обязательных чертежей)

среда разработки Zend Studio 8
List_STATE - таблица стран
Управления пользователями
Выбор баз данных и манипулирования
Экспорт баз данных
Разработка веб - приложения

Рекомендуемая основная литература

1. Баралов К.Б., Алибаева С.А., Бабич А.А. Методические указания по выполнению экономического раздела выпускной работы бакалавра для студентов всех форм обучения специальности
2. Хакимжанов Т.Е. Расчет астрономических систем методические указания для студентов всех форм обучения - Алматы: АИТС, 2002г

Консультанты по проекту с указанием относящихся к ним разделов

Раздел	Консультант	Сроки	Подпись
Экономика	Ержанова З.Ю.	15.04-20.05.14	Ержанова
Основная часть	Кожамбердиева М.Н.	03.03 - 01.05	
БЖД Приходько	Приходько Я.Г.	11.04 - 26.05.14	
Нормоконтроль	Гусупов Д.М.	30.05.14	

Г Р А Ф И К
подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
1.	Сбор необходимого материала и ознакомление по литературным источникам	10.02.14 - 20.02.14 гг	
2.	Разработка системы управления базами данных для CMS KIX	20.02.14 - 01.03.14гг	
3.	Создание вывода результатов работы без перезагрузки страницы браузера	01.03.14 - 15.03.14гг.	
4.	Безопасность жизнедеятельности	11.04 - 26.05.14	
5.	Бизнес - план	15.04 - 20.05.14	

Дата выдачи задания «3» марта 2014 г.

Заведующий кафедрой _____ Курмалбаев З.К.
(подпись) (Фамилия и инициалы)

Руководитель _____ Жексембердиева М.Д
(подпись) (Фамилия и инициалы)

Задание принял к исполнению
студент _____ Алимбек А.С
(подпись) (Фамилия и инициалы)

Аңдапта

Берілген дипломдық жобада CMS RIX басқару жүйесінің өңдеу атқарылды және де CMS RIX деректер қорында кестенің мәнін өзгертуге ,қосу және өшіру,құру сақтау элементтерін көрсетілді.

Өміртіршілік қауіпсіздігін қамтамасыз ету бөлімінде еңбек ету жағдайына талдау жасалған, жарық беру және желдету есептелген.

Экономикалық бөлімде жобаның экономикалық тиімділігін дәлелдеп берген бизнес-жоспар жасалды.

Аннотация

В данном дипломном проекте проведена разработка системы управления CMS RIX, которая позволит отображать используемые в CMS RIX базы данных, их содержимое - таблицы и структуру самих таблиц, редактирование и удаление, создание и сохранение баз данных и их элементов.

В разделе обеспечения безопасности жизнедеятельности проведен анализ условий труда, сделан расчет освещения и кондиционирования.

В экономической части составлен бизнес план проекта, подтвердивший его экономическую целесообразность.

Annotation

In this thesis project carried out development management system CMS RIX, which will display used in CMSRIX databases, their contents - tables and structure of tables, editing and deleting, creating and maintaining databases and their components.

In the life safety analysis of working conditions, is used to calculate the lighting and air conditioning.

In the economic part of the project, a business plan, this confirmed its economic feasibility.

Содержание

Введение

Развитие средств вычислительной техники обеспечило для создания и широкого использования систем обработки данных разнообразного назначения. Разрабатываются информационные системы для обслуживания различных систем деятельности, систем управления хозяйственными и техническими объектами, модельные комплексы для научных исследований, системы автоматизации проектирования и производства, всевозможные тренажеры и обучающие системы. Одной из важных предпосылок создания таких систем стала возможность оснащения их "памятью" для накопления, хранения и систематизации больших объемов данных. Другой существенной предпосылкой нужно признать разработку подходов, а также создание программных и технических средств конструирования систем, предназначенных для коллективного пользовании. В этой связи потребовалось разработать специальные методы и механизмы управления такого рода совместно используемыми ресурсами данных, которые стали называться базами данных. Исследования и разработки, связанные с проектированием, созданием и эксплуатации баз данных, а также необходимых для этих целей языковых и программных инструментальных средств, привели к появлению самостоятельной ветви информатики, получившей название системы управления данными.

Программы, которые предназначены для структурирования информации, размещения ее в таблицах и манипулирования данными называются системами управления базами данных (СУБД). Другими словами СУБД предназначены как для создания и ведения базы данных, так и для доступа к данным.

Такие программные комплексы выполняют довольно сложный набор функций, связанный с централизованными управлениями, данными в базе данных интерфейсах всей совокупности ее пользователей. По существу, система управления базами данных служит посредником между пользователями и базой данных. Она представляет пользователю удобные средства интерактивного взаимодействия с БД.

Выпускная работа состоит из трех основных частей. В первой части идет общее описание принципов, на которых разрабатывается система управления базами данных для CMS RIX, производится постановка задачи, анализ с соответствующим выбором технологии разработки на основе исследования и сравнения технологий разработки готовых продуктов. Во второй части рассмотрение выбранной технологии с подробным описанием инструментария разработки. В третьей части идет предметная часть, с конкретной демонстрацией продукта и детальным описанием программного обеспечения. Так же в работе рассматривается решение вопросов охраны труда и безопасности жизнедеятельности и производится экономической расчет.

1 Обзор и анализ существующих систем и технологий разработки

1.1 Поставленная задача и пути ее решения

В дипломной работе поставлена задача разработки системы управления базами данных для CMS RIX, которая позволит отображать используемые в CMS RIX базы данных, их содержимое - таблицы и структуру самих таблиц, редактирование и удаление, создание и сохранение баз данных и их элементов. CMS RIX представляет собой систему управления содержимым сайтов.

Система должна удовлетворять поставленным требованиям, включать в себя лучшие качества уже имеющихся систем. Так как абсолютно все системы управления базами данных, написанные на PHP, постоянно перезагружают страничку, выдавая результаты своей работы, то для комфортной работы с базами данных конечных пользователей CMS RIX необходимо создать вывод результатов работы без перезагрузки страницы браузера. Этого можно добиться, используя современную технологию AJAX. AJAX - это подход к построению интерактивных пользовательских интерфейсов веб-приложений, заключающийся в «фоновом» обмене данными браузера с веб-сервером. В результате, при обновлении данных, веб-страница не перезагружается полностью, и веб-приложения становятся более быстрыми и удобными.

Возможности и структура CSM RIX

CMS RIX (система управления контентом) выполняет множество разнообразных функций. Функции, выполняемые CMS, можно объединить в несколько групп:

- управление контентом сайта;
- управление интернет-магазином;
- управление представлением данных;
- управления доступом;
- управление почтовыми подписками;
- вспомогательные функции CMS.

Управление контентом сайта

Система управления контентом сайтов CMS RIX выполняет множество функций, необходимых и достаточных для управления сайтами различных видов:

- CMS позволяет создавать, удалять и иерархически упорядочивать страницы сайта;
- CMS автоматически формирует удобные средства навигации по сайту (ссылки, различные меню, карту сайта, списки страниц, указатели пройденного пути и т.п.);
- CMS оперирует разделами различных типов (статьи, новостные ленты, форумы, доски объявлений, почтовые формы, опросы, голосования и т.п.);

- CMS наполняет страницы сайта блоками контента разных типов (текст, изображение, список, таблица и т.п.).

Управление интернет-магазином

Система управления контентом CMS RIX позволяет управлять интернет-магазином:

- загружать каталог товаров (прайс-лист) в магазин с локального компьютера администратора магазина при помощи транспортных файлов в пакетном режиме;
- редактировать каталог товаров в онлайне через веб-интерфейс;
- выгружать каталог товаров из магазина на локальный компьютер администратора в виде архивированного транспортного файла;
- выгружать из магазина информацию о покупателях;
- выгружать из магазина информацию о заказах.

Управление представлением данных на сайте

Система управления контентом CMS RIX использует для управления внешним видом сайта и представлением данных на его страницах систему шаблонов. Шаблоны построены на технологии XML/XSLT. Благодаря XSLT один и тот же сайт может формировать свои страницы для вывода на самые разные устройства:

- персональный компьютер с тем или иным браузером;
- принтер;
- карманный персональный компьютер КПК (PDA);
- смартфон;
- коммуникатор.

При этом администратор сайта имеет дело с единым интерфейсом. Контент сайта не зависим от формата вывода.

Естественно, что изображения, помещаемые на сайт, должны учитывать особенности устройств вывода. Можно либо подгонять их под худший случай, либо иметь дубликаты с уменьшенными размерами, которые будут использоваться для построения страниц для устройств с маленькими экранами.

Управление доступом к данным

CMS RIX позволяет управлять правами доступа посетителей к разделам сайта. Часть разделов можно объявлять закрытыми, тогда они будут доступны только зарегистрированным и авторизовавшимся посетителям при условии, что они входят в группу, которой доступен требуемый раздел. В противном случае посетитель получит на своем экране страницу с предупреждением, что раздел закрыт, что его прав доступа недостаточно для просмотра его содержаниями с предложением авторизоваться или зарегистрироваться.

Система управления контентом обеспечивает коллективное управление содержанием сайта за счет:

- разделения доступа администраторов к различным частям сайта (разделам и группам разделов);
- увязывания цепочек исполнителей;
- управления публикацией разделов.

Управление почтовыми подписками

Система управления контентом CMS RIX позволяет:

- создавать почтовые рассылки;
- вести списки подписчиков;
- создавать и редактировать письма;
- отправлять письма подписчикам.

Управление заголовками и баннерами

В системе управления контентом CMS RIX имеются возможности управления показами баннеров, ротацией изображений.

CMS позволяет легко контролировать содержимое тегов title и meta (keywords и description), что благотворно сказывается на результатах продвижения сайтов методами поисковой оптимизации.

Легкость управления контентом и специальными тегами для формирования заголовков разного уровня, которую обеспечивает система управления контентом, существенно снижает затраты на проведение раскрутки сайта легальными методами (поисковая оптимизация, ссылочное ранжирование.

1.2 Понятие системы управления базами данных

В нашей жизни широко используются такие базы данных, как: информационная система по продаже и резервированию авиа - и железнодорожных билетов; базы данных, заменяющая привычный библиотечный каталог; базы данных - электронные энциклопедии со сведениями, например, о музыкальных инструментах, шедеврах Эрмитажа или кулинарных рецептах, химических элементах и соединениях, сотрудниках какого-либо учреждения.

База данных (БД) - это взаимосвязанная информация (данные) об объектах, которая организованна специальным образом и хранится во внешней памяти компьютера.

Программное обеспечение, позволяющее создавать базы данных, обновлять хранимую в ней информацию, обеспечивающее удобный доступ к ней с целью просмотра и поиска, называется системой управления базой данных (СУБД). Современная СУБД должна также обеспечивать возможность работы с БД в глобальных и локальных сетях.

Основные функции СУБД:

- управление данными во внешней памяти (на дисках);

- управление данными в оперативной памяти с использованием дискового кэша;
- журнализация изменений, резервное копирование и восстановление базы данных после сбоев;
- поддержка языков БД (язык определения данных, язык манипулирования данными).

СУБД классифицируются:
- по модели данных (иерархические, сетевые, реляционные);
- по степени распределенности (локальные, распределенные);
- по способу доступа к БД.

1.3 Специфика требуемой системы управления базами данных

Спецификация системы заключается в том, что система управления базами данных будет использоваться для управления базами данных, используемыми в системе управления содержимым сайта RIX.

Любой Web-сайт состоит из набора страниц, а различия заключаются лишь в том, как они организованы. Существует два вида организации Web-сайта - статический и динамический. В первом случае специалисты, отвечающиеза создание и поддержку сайта пишут в HTML-форме каждую в отдельности страницу, включая ее оформление и контент. Во втором - в основе любой Web-страницы лежит шаблон, определяющий расположение в окне Web-браузера всех компонентов страницы и вставка конкретной информации производится с использованием стандартных средств, не требующих от участника процесса знания языка HTML и достаточно сложных для неспециалиста процедур публикации Web-страницы.

Если сайт состоит из множества страниц или он должен часто обновляться, то преимущество динамической организации становится очевидным. Разработчикам Web-сайта не надо переписывать всю страницу при изменении ее информационного наполнения или дизайна. Страницы не хранятся целиком, а формируются динамически при обращении к ним.

Таким образом, отделение дизайна от контента является главной отличительной особенностью динамических сайтов от статических. На этой основе возможны дальнейшие усовершенствования структуры сайта, такие как определение различных пользовательских функций и автоматизация бизнес-процессов, а самое главное, контроль поступающего на сайт контента.

Для создания динамического сайта возможны два пути. Во-первых, это написание собственных программ, отвечающих за создание нужных шаблонов и поддерживающих необходимые функции. При этом созданная система будет полностью отвечать потребностям, однако возможно потребует больших программистских усилий и времени. Второй путь - это воспользоваться уже существующими системами, которые и называются системами управления Web - контентом. Преимуществом этого пути является уменьшение затрат времени и

сил. К его недостаткам можно отнести снижение гибкости, предоставление недостаточного или чрезмерного набора возможностей.

Под контентом (дословный перевод английского термина content, означающего содержание, содержимое) понимают информационное наполнение сайта - то есть все типы материалов, которые находятся на сервере: web-страницы, документы, программы, аудио-файлы, фильмы и так далее. Таким образом, управление контентом - это процесс управления подобными материалами. Он включает следующие элементы: размещение материалов на сервере, удаление материалов с сервера, когда в них больше нет необходимости, организацию (реорганизацию) материалов, возможность отслеживать их состояние.

Системы управления контентом (в английском языке существует устоявшийся термин - Content Management Systems или, сокращенно, CMS) - это программные комплексы, автоматизирующие процедуру управления контентом.

Сам же контент в свою очередь хранится в базе данных, которая должная быть заранее спроектирована и которой необходимо управлять, проводить реорганизацию данных и их конфигурирование. Именно для этого и создается требуемая системы управления базами данных, сложность которой повышается из-за использования ее в CMS RIX. Она должна обеспечивать гибкость и масштабируемость, а также выполнять дополнительные требования, предъявляемые к проектам, реализуемым на CMS RIX. Также система обладает огромным потенциалом для развития и в будущем может позиционироваться как самостоятельный продукт.

1.4 Сравнительный анализ существующих систем управления базами данных для веб

Рассмотрим и проведем анализ ряда существующих на сегодняшний день систем управления базами данных для веб. Перечень систем, обзор которых проведен, определялся по уровню популярности, доступности и открытости технологии разработки и количеству успешных внедрений.

1.4.1 phpMyAdmin

phpMyAdmin - веб - приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL. phpMyAdmin позволяет через браузер осуществлять администрирование сервера MySQL, запускать команды SQL и просматривать содержимое таблиц и баз данных. Приложение пользуется большой популярностью у веб-разработчиков, так как позволяет управлять СУБД MySQL без непосредственного ввода SQL команд, предоставляя дружественный интерфейс.

На сегодняшний день phpMyAdmin широко применяется на практике. Последнее связано с тем, что разработчики интенсивно развивают свой продукт, учитывая все нововведения СУБД MySQL. Подавляющее большинство интернет провайдеров используют это приложение в качестве панели управления для того, чтобы предоставить своим клиентам возможность администрирования выделенных им баз данных.

Приложение распространяется под лицензией GNU General Public License и поэтому многие другие разработчики интегрируют его в свои разработки, например XAMPP, Denwer, AppServ.

Проект на данный момент времени локализирован на более чем 50 языках.

1.4.2 Navicat for MySQL

Navicat for MySQL - мощная графическая утилита для управления и работы с базами данных MySQL, имеющая наглядный интерфейс. Поддерживает работу со всеми версиями MySQL, начиная от версии 3.21 и выше, включая поддержку большинства последних MySQL функций. Имеет хорошо-продуманный графический интерфейс пользователя с легким созданием, организацией и обменом информацией безопасным и простым способом. Позволяет пользователю подключаться к локальным и удаленным серверам MySQL, предоставляя ряд инструментов таких как администрирование баз данных, функции импорта и экспорта, а также создание резервных копий и пересылки данных. Также данная программа позволяет конвертировать Access в MySQL, MS SQL в MySQL, Excel в MySQL и производить синхронизацию данных.

1.4.3 HeidiSQL

В отличие, например, от phpMyAdmin, HeidiSQL уже не является веб-приложением. Это программа для ОС Windows с продуманным интерфейсом, благодаря которому работа с базами превращается в одно удовольствием. Преимущества десктопной программы на лицо. Просмотр и редактирование данных осуществляется через удобнейший grid (таблица с возможностью редактирования). Сравнивая это с phpMyAdmin, где в таблицах лишь отображаются данные, а изменение любой из записей осуществляется на отдельной странице. HeidiSQL позволяет отсортировать данные и, что особенно удобно, использовать фильтры, отбирая записи по определенной маске. Для большей наглядности к таблице можно применить различные цветовые схемы - особенность из разряда тех, что сначала кажутся незначительной мелочью, но через некоторое время так к ней привыкаешь, что уже не можешь отказаться.

Впрочем, едва ли работа с базой ограничится лишь редактированием таблиц. Одна из ключевых особенностей программы - редактор SQL-запросов. Как и в современных средах разработки, в распоряжении пользователя особые

функции по редактированию кода: автодополнение названий баз данных и таблиц, а также подсказки с конструкциями запроса. Теперь вообще можно не задумываться по поводу названия таблиц и полей - HeidiSQL сама подскажет нужные варианты. Более того, пользователю предоставляется система шаблонных заготовок кода (так называемых сниппетов), за счет которых возможно не только упростить, но еще и ускорить разработку. Единственный недостаток - отсутствие закладок для разных запросов. Без этой жизненно важной детали интерфейса будет очень сложно, если одновременно приходится выполнять несколько разных запросов. Зато имеется редактор хранимых процедур, крайне дружелюбный к пользователю и упрощающий процесс создания функций и триггеров.

Вообще с HeidiSQL любые действия с базами данных довольно легко выполнимы. Нет ничего проще, чем, например, сделать дамп базы с ее структурой и данными: HeidiSQL быстро сгенерирует любой SQL-экспорт. Через удобный интерфейс можно сдампить структуру базы и сами данные в файл или же вообще на другой сервер. Но перед тем как бездумно переносить дамп, нужно подумать: если на обоих серверах есть одинаковые базы, то возможно уместнее воспользоваться встроенной функцией по синхронизации. А если данные нужно перенести в другой формат, то в этом случае можно воспользоваться экспортом в CSV, HTML и XML форматы.

Теперь что касается администрирования. HeidiSQL позволяет мониторить и удалять клиентские процессы. Это отличная возможность проанализировать выполнения запросов и удалить ненужные процессы. Помимо этого можно удобно редактировать переменные сервера, а также управлять привилегиями пользователей с помощью интерфейса, подобного для редактирования ACL-листа для файлов NTFS.

Огромным недостатком является невозможность использования программы, если на сервере заблокирован порт MySQL демона. HeidiSQL банально не сможет подключиться к серверу и будет абсолютно бесполезным. А на дешевых хостингах такая ситуация в порядке в вещей, а механизмов для обхода этого ограничения у программы нет.

1.4.4 dbForge Studio

На официально страничке программы среди пользователей известные бренды: Hitachi, Honda, Samsung, BMW, Siemens. От этого вдвойне приятно, что dbForge Studio абсолютно бесплатна для некоммерческого использования. По большому счету это еще один удобный фронт-енд для работы с базами данных MySQL. Составлять и выполнять запросы, редактировать данные, осуществлять их экспорт и импорт, разрабатывать SQL-скрипты и хранимые процедуры - получаем стандартный набор функций, но даже при качественной реализации он мало кого удивит. Но dbForge Studio добавил к этому отладчик хранимых процедур, визуальных редактор для составления SQL-запросов, редактор кода с автодополнением команд и имен баз/таблиц.

Ручная отладка хранимых процедур и триггеров с промежуточными выводами и вычислениями в голове в большинство случаев дико тормозит разработку. Зато отладчик хранимых процедур, встроенный в dbForge Studio, - имеет хорошую функциональность. Прямо в редакторе кода можно в нужном месте установить точки останова, провести выполнение всей или части процедуры по шагам, и на каждой итерации отслеживать значения переменных и результатов выполнения запросов. SQL-запросы вовсе необязательно набирать вручную - их можно конструировать с помощью специального визуального средства. Оно вряд ли поможет создать действительно сложный запрос, но зато окажет неоценимую помощь тем, кто не очень хорошо владеет SQL.

Вообще визуальные средства и в особенности конструктор базы данных - это одна из главных особенностей dbForge Studio. Если нужно быстро сконструировать базу, то лучшего решения, пожалуй, не найти. Помещаются на рабочей области разные сущности, устанавливаются связи между ними, редактируются параметры полей - и база готова к использованию. Примечательно, что прямо из визуального конструктора можно обратиться к любому элементу базы, будь это обычное поле таблицы или же хранимая процедура. Очень удобно использовать уже тогда, когда база готова, но нужно быстро найти и подправить один из ее элементов.

dbForge Studio также отличается продвинутым экспортом данных. Мастер экспорта позволит выбрать нужные столбцы и колонки, задать различное отображение данных, и преобразовать данные в один из следующих форматов: Text, DBF, HTML, MS Access, MS Excel, ODBC, PDF, RTF, CSV и XML.

2 Выбор технологии разработки и описание процесса разработки

2.1 Обзор выбранной технологии

Произведя обзор и анализ рассмотренных систем управления базами данных для веб осуществим выбор технологии разработки требуемой системы управления базами данных.

Для выполнения поставленной задачи среда разработки должна удовлетворять следующим требованиям:

- обладать достаточным уровнем надежности;
- обладать достаточно широким диапазоном инструментальных средств разработки;
- иметь возможность для разработки приложения, удовлетворяющего условиям поставленной задачи;
- иметь возможность для разработки базы данных, обладающей достаточным уровнем защищенности, возможностью экспорта, импорта и расширения архитектуры;
- язык программирования, с помощью которого будет вестись разработка, должен обладать возможностями, позволяющими реализовать продукт в наилучшем качестве;
- язык программирования должен обладать достаточно широким функционалом для создания необходимых функций, процедур и классов;
- синтаксис языка программирования должен быть прост и легок в освоении и изучении.

В соответствии с этими требованиями и анализом технологий разработки других систем мной сделан выбор технологии, основанной на использовании программных продуктов компании ActiveState, Oracle Corporation и Zend Technologies Ltd. Конкретно для разработки своего программного обеспечения я использую редактор ActiveState Komodo Edit 6.0.0, движок баз данных MySQL и скриптовый язык программирования общего назначения PHP.

Обосновываю свой выбор я следующими фактами:

- редактор ActiveState Komodo Edit обладает достаточно мощным набором программных средств для выполнения поставленной задачи;
- редактор ActiveState Komodo Edit достаточно удобен и легок в освоении и использовании;
- свободный движок базы данных MySQL является собственностью компании Oracle Corporation, получившей её вместе с поглощённой Sun Microsystems, осуществляющей разработку и поддержку приложения;
- MySQL является решением для малых и средних приложений. Входит в состав серверов WAMP, LAMP и в портативные сборки серверов Денвер, XAMPP;
- PHP - скриптовый язык программирования общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является

одним из лидеров среди языков программирования, применяющихся для создания динамических веб-сайтов;

- в области программирования для Сети PHP - один из популярных скриптовых языков (наряду с JSP, Perl и языками, используемыми в ASP.NET) благодаря своей простоте, скорости выполнения, богатой функциональности, кроссплатформенности и распространению исходных кодов на основе лицензии PHP.

Основываясь именно на этих принципах мной и была обоснованно выбрана данная технология разработки.

Итак, произведя обзор и анализ существующих систем, выполняющих задачи, схожие с поставленной мне, в качестве языка разработки программного обеспечения мною выбран PHP.

Популярность в области построения веб - сайтов определяется наличием большого набора встроенных средств для разработки веб - приложений. Основные из них:

- автоматическое извлечение POST и GET-параметров, а также переменных окружения веб-сервера в предопределённые массивы;
- взаимодействие с большим количеством различных систем управления базами данных (MySQL, MySQLi, SQLite, PostgreSQL, Oracle (OCI8), Oracle, Microsoft SQL Server, Sybase, ODBC, mSQL, IBM DB2, Cloudscape и Apache Derby, Informix, Ovrimos SQL, Lotus Notes, DB++, DBM, dBase, DBX, FrontBase, FilePro, Ingres II, SESAM, Firebird / InterBase, Paradox File Access, MaxDB, Интерфейс PDO);
- автоматизированная отправка HTTP-заголовков;
- работа с HTTP - авторизацией;
- работа с cookies и сессиями;
- работа с локальными и удалёнными файлами, сокетами;
- обработка файлов, загружаемых на сервер;
- работа с XForms.

В настоящее время PHP используется сотнями тысяч разработчиков. Согласно рейтингу корпорации TIOBE, базирующемся на данных поисковых систем, в апреле 2011 года PHP находился на 5 месте среди языков программирования. К крупнейшим сайтам, использующим PHP, относятся Facebook, ВКонтакте, Wikipedia и др.

Входит в LAMP - распространённый набор программного обеспечения для создания веб-сайтов (Linux, Apache, MySQL, PHP).

2.2 История языка

В 1994 году датский программист Расмус Лердорф создал набор скриптов на Perl/CGI для вывода и учёта посетителей его онлайн - резюме, обрабатывающий шаблоны HTML - документов. Лердорф назвал набор *Personal Home Page* (*Личная Домашняя Страница*). Вскоре функциональности и быстроты Perl - интерпретатора скриптов - перестало хватать, и Лердорф разработал с использованием языка C новый интерпретатор шаблонов *PHP/FI*

(англ. *Personal Home Page / Forms Interpreter* - «Личная Домашняя Страница / Интерпретатор форм»).

В 1997 году после длительного бета - тестирования вышла вторая версия обработчика, написанного на C - PHP/FI 2.0. Её использовали около 1 % (приблизительно 50 тысяч) всех интернет - доменов мира.

Версия PHP 3.0 подверглась значительной переработке, определившей современный облик и стиль языка программирования. В 1997 году два израильских программиста, Энди Гутманс и Зээв Сураски, полностью переписали код интерпретатора. PHP 3.0 был официально выпущен в июне 1998 года.

Одной из сильнейших сторон PHP 3.0 была возможность расширения ядра дополнительными модулями. Впоследствии интерфейс написания расширений привлёк к PHP множество сторонних разработчиков, работающих над своими модулями, что дало PHP возможность работать с огромным количеством баз данных, протоколов, поддерживать большое число API. Большое количество разработчиков привело к быстрому развитию языка и стремительному росту его популярности. С этой версии акроним php расшифровывается как «PHP: hypertext Preprocessor», вместо устаревшего «Personal Home Page».

К зиме 1998 года, практически сразу после официального выхода PHP 3.0, Энди Гутманс и Зээв Сураски начали переработку ядра PHP. В задачи входило увеличение производительности сложных приложений и улучшение модульности базиса кода PHP. Новый движок, названный Zend Engine, успешно справлялся с поставленными задачами и впервые был представлен в середине 1999 года. PHP 4.0, основанный на этом движке и принёсший с собой набор дополнительных функций, официально вышел в мае 2000 года. В дополнение к улучшению производительности, PHP 4.0 имел ещё несколько ключевых нововведений, таких как поддержка сессий, буферизация вывода, более безопасные способы обработки вводимой пользователем информации и несколько новых языковых конструкций.

Пятая версия PHP была выпущена разработчиками 13 июля 2004 года. Изменения включают обновление ядра Zend (Zend Engine 2), что существенно увеличило эффективность интерпретатора. Введена поддержка языка разметки XML. Полностью переработаны функции ООП, которые стали во многом схожи с моделью, используемой в Java. В частности, введён деструктор, открытые, закрытые и защищённые члены и методы, окончательные члены и методы, интерфейсы и клонирование объектов. В последующих версиях также были введены пространства имён, замыкания и целый ряд достаточно серьёзных изменений, количественно и качественно сравнимых с теми, которые появились при переходе на PHP 5.0.

Шестая версия PHP находится в стадии разработки с октября 2006 года. В ней уже сделано множество нововведений, как, например, исключение из ядра регулярных выражений POSIX и «длинных» суперглобальных массивов, удаление директив *safe_mode, magic_quotes_gpc* и *register_globals* из

конфигурационного файла php.ini. Также много внимания уделено поддержке Юникода (Рисунке 2.1).

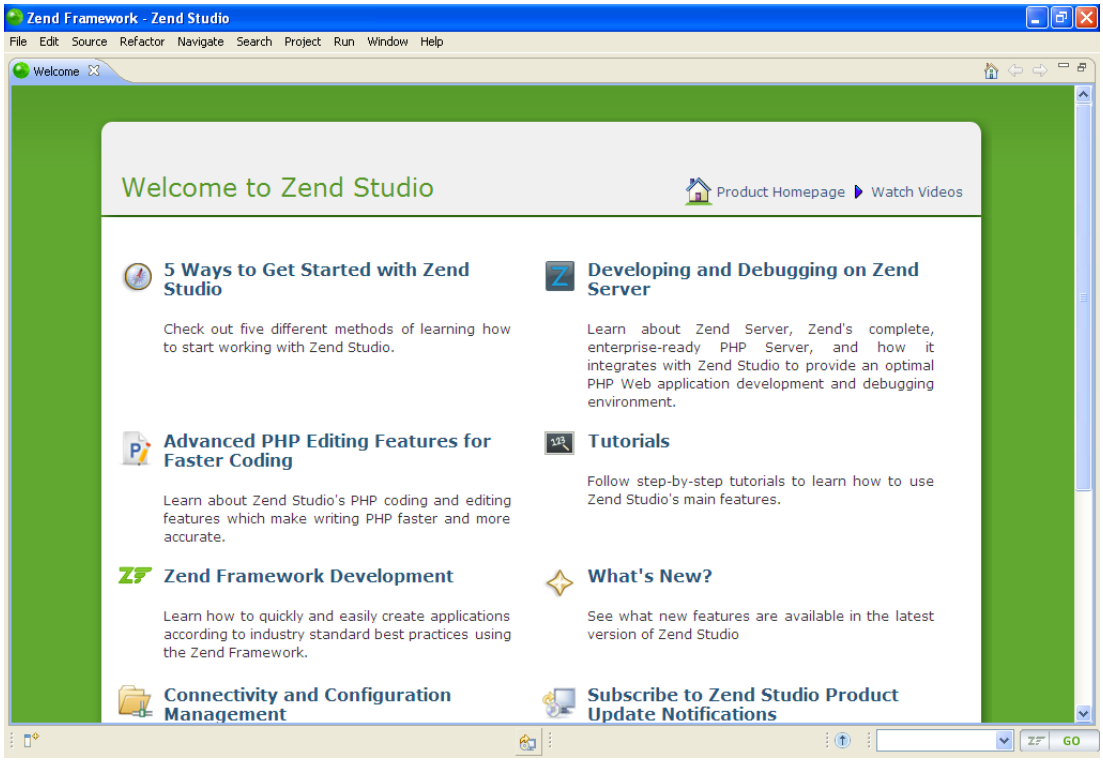


Рисунок 2.1 - Среда разработки Zend Studio 8

2.3 Язык PHP и его основные характеристики

Язык программирования PHP и его интерпретатор разрабатываются группой энтузиастов в рамках проекта с открытым кодом. Проект распространяется под собственной лицензией, несовместимой с GNU GPL.

2.3.1 Синтаксис

Синтаксис PHP подобен синтаксису языка Си. Некоторые элементы, такие как ассоциативные массивы и цикл foreach, заимствованы из Perl.

Для работы программы не требуется описывать какие-либо переменные, используемые модули и т. п. Любая программа может начинаться непосредственно с оператора PHP.

PHP исполняет код, находящийся внутри ограничителей, таких как <?php ?>. Всё, что находится вне ограничителей, выводится без изменений. В основном это используется для вставки PHP-кода в HTML-документ.

Помимо ограничителей <?php ?>, допускается использование дополнительных вариантов, таких как <? ?> и <script language="php"> </script>. Кроме того, до версии 6.0 допускается использование ограничителей языка программирования ASP <% %> (конструкции <? ?> и <% %> могут быть выключены в конфигурационном файле php.ini).

Имена переменных начинаются с символа `$`, тип переменной объявлять не нужно. Имена переменных, функций и классов чувствительны к регистру. Константы также чувствительны к регистру. Переменные обрабатываются в строках, заключённых в апострофы или двойные кавычки, и heredoc-строках (строках, созданных при помощи оператора <<<).

PHP рассматривает переход на новую строку как пробел, так же как HTML и другие языки со свободным форматом. Инструкции разделяются с помощью точки с запятой (`;`), за исключением некоторых случаев, после объявления конструкции if/else и циклов.

PHP поддерживает три типа комментариев: в стиле языка Си (ограниченные /* */), C++ (начинающиеся с // и идущие до конца строки) и оболочки UNIX (с # до конца строки).

2.3.2 Типы данных

PHP является языком программирования с динамической типизацией, не требующим указания типа при объявлении переменных, равно как и самого объявления переменных. Преобразования между скалярными типами зачастую осуществляются неявно без дополнительных усилий (впрочем, PHP предоставляет широкие возможности и для явного преобразования типов).

К скалярным типам данных относятся:

- целый тип (integer);
- вещественный тип данных (float, double);
- логический тип (boolean);
- строковый тип (string);
- и специальный тип NULL.

К нескалярным типам относятся:

- «ресурс» (resource);
- массив (array);
- объект (object);
- анонимная функция (closure) или псевдотип callback.

Диапазон целых чисел (integer) в PHP зависит от платформы (обычно, это диапазон 32-битных знаковых целых чисел, то есть, от -2 147 483 648 до 2 147 483 647). Числа можно задавать в десятичной, восьмеричной и шестнадцатеричной системах счисления. Диапазон вещественных чисел (double), также, зависит от платформы (для 32-битной архитектуры диапазон позволяет оперировать числами от $\pm 1.7\times10^{-308}$ до $\pm 1.7\times10^{+308}$).

PHP предоставляет разработчикам логический тип (boolean), способный принимать только два значения TRUE («истина») и FALSE («ложь»). При преобразовании в логический тип число 0, пустую строку, ноль в строке «0», `NULL` и пустой массив считаются равными FALSE. Все остальные значения автоматически преобразуются в TRUE.

Специальный тип NULL предназначен для переменных без определённого значения. Единственным значением данного типа является константа NULL. Тип NULL принимают неинициализированные переменные,

переменные инициализированные константой NULL, а также переменные, удалённые при помощи конструкции unset().

Ссылки на внешние ресурсы имеют тип «ресурс» (resource). Переменные данного типа, как правило, представляют собой дескриптор, позволяющий управлять внешними объектами, такими как файлы, динамические изображения, результирующие таблицы базы данных и т.п.

Массивы (array) поддерживают числовые и строковые ключи и являются гетерогенными. Массивы могут содержать значения любых типов, включая другие массивы. Порядок элементов и их ключей сохраняется. Не совсем корректно называть PHP-массивы массивами, на самом деле это, скорее всего, упорядоченный хеш. Возможно неожиданное поведение при использование цикла for со счетчиком вместо foreach. Так, например, сортируя массив с численными индексами функциями из стандартной библиотеки, сортируются и ключи тоже.

Указатель на функцию в PHP может быть представлен замыканием или типом callback. Замыкание доступно с версии 5.3 и в коде выглядит как простое определение функции, в которую явно можно утянуть значения из контекста callback тип может быть представлен:

- строкой (интерпретируется как название функции);
- массивом, где нулевой и первый элемент строки (интерпретируется как название статичной функции в классе);
- массивом, где нулевой элемент объект, а первый строка (интерпретируется как метод у объекта).

Для проверки является ли значение вызываемым. Следует использовать is_callable($var).

2.3.3 Объектно-ориентированное программирование

PHP поддерживает широкие объектно-ориентированные возможности, полная поддержка которых была введена в пятой версии языка.

Класс в PHP объявляется с помощью ключевого слова class. Методы и поля класса могут быть общедоступными (public, по умолчанию), защищёнными (protected) и скрытыми (private). PHP поддерживает все три основных механизма ООП - инкапсуляцию, полиморфизм и наследование (родительский класс указывается с помощью ключевого слова extends после имени класса). Поддерживаются интерфейсы (ставятся в соответствие с помощью implements). Разрешается объявление финальных, абстрактных методов и классов. Множественное наследование классов не поддерживается, однако класс может реализовывать несколько интерфейсов. Для обращения к методам родительского класса используется ключевое слово parent.

Классы в PHP имеют ряд специальных методов (англ. Magic methods), начинающихся с двух символов подчёркивания. Особо стоит отметить конструктор construct(), в версиях до 5.0 конструктором служил метод, одноимённый с классом) и деструктор (destruct()), а также методы чтения (get()) и записи (set()), свёртывания (sleep()) и развёртывания (wake()), клонирования

(clone()) и др. Эти методы являются достаточно гибким инструментом: переопределяя их, можно добиться существенного изменения поведения объекта.

Экземпляры класса создаются с помощью ключевого слова new, обращение к полям и методам объекта производится с использованием оператора ->. Для доступа к членам класса из его методов используется переменная $this.

Начиная с пятой версии PHP, объекты передаются по ссылке.

«Paamayim Nekudotayim» или просто «двойное двоеточие». Используя эту лексему, программист может обращаться к константам, статическим или перегруженным свойствам или методам класса. При обращении к этим элементам извне класса, программист должен использовать имя этого класса. «Paamayim Nekudotayim» на первый взгляд может показаться странным словосочетанием для обозначения двойного двоеточия. Однако, во время создания Zend Engine версии 0.5 (который входил в PHP3), Andi и Zeev выбралиименно это обозначение. «Paamayim Nekudotayim» действительно значит «двойное двоеточие» на иврите. Это обозначение не менялось ни разу в течение всего времени разработки PHP.

2.3.4 Суперглобальные массивы

Суперглобальными массивами (англ. Superglobal arrays) в PHP называются предопределённые массивы, имеющие глобальную область видимости без использования директивы global. Большая часть этих массивов содержит входные данные запроса пользователя (параметры GET-запроса, поля форм при посылке методом POST, куки и т.п.).

Все суперглобальные массивы, кроме $GLOBALS и $_REQUEST, имеют устаревшие аналоги с длинными именами, которые доступны вплоть до пятой версии PHP (в шестой версии планируется их исключение). Таким образом, обращения $_GET['year'] и $HTTP_GET_VARS['year'] идентичны (за исключением области видимости: массивы с «длинными» именами не являются суперглобальными).

`$GLOBALS`

Массив всех глобальных переменных (в том числе и пользовательских).

`$_SERVER (устаревший аналог - $HTTP_SERVER_VARS)`

Содержит переменные окружения, которые операционная система передаёт серверу.

`$_ENV (уст. $HTTP_ENV_VARS)`

Текущие переменные среды (англ. Environment variables). Их набор специфичен для платформы, на которой выполняется скрипт.

`$_GET (уст. $HTTP_GET_VARS)`

Содержит параметры GET-запроса, переданные в URI после знака вопроса «?».

`$_POST (уст. $HTTP_POST_VARS)`

Ассоциативный массив значений полей HTML-формы при отправке методом POST. Индексы элементов соответствуют значению атрибута name элементов управления HTML-формы.

`$_FILES (уст. $HTTP_POST_FILES)`

Ассоциативный массив со сведениями об отправленных методом POST файлах. Каждый элемент имеет индекс, идентичный значению атрибута «name» в форме, и, в свою очередь, также является массивом со следующими элементами:

- ['name'] - исходное имя файла на компьютере пользователя;
- ['type'] - указанный агентом пользователя MIME-тип файла. PHP не проверяет его, и поэтому нет никаких гарантий, что указанный тип соответствует действительности;
- ['size'] - размер файла в байтах;
- ['tmp_name'] - полный путь к файлу во временной папке. Файл необходимо переместить оттуда функцией move_uploaded_file. Загруженные файлы из временной папки PHP удаляет самостоятельно;
- ['error'] - код ошибки. Если файл удачно загрузился, то этот элемент будет равен 0 (UPLOAD_ERR_OK).

`$_COOKIE (уст. $HTTP_COOKIE_VARS)`

Ассоциативный массив с переданными агентом пользователя значениями куки.

`$_REQUEST`

Содержит элементы из массивов $_GET, $_POST, $_COOKIE. С версии PHP 4.1 включает $_FILES.

`$_SESSION (уст. $HTTP_SESSION_VARS)`

Содержит данные сессии.

2.3.5. Некоторые особенности интерпретатора

PHP - скрипты обычно обрабатываются интерпретатором в порядке, обеспечивающем кроссплатформенность разработанного приложения:

- лексический анализ исходного кода и генерация лексем;
- синтаксический анализ полученных лексем;
- генерация байт - кода;
- выполнение байт - кода интерпретатором (без создания исполняемого файла).

Для увеличения быстродействия приложений, возможно, использование специального программного обеспечения, так называемых акселераторов. Принцип их работы заключается в кэшировании однажды сгенерированного байт-кода в памяти и/или на диске, таким образом, из процесса работы приложения исключаются этапы 1-3, что в общем случае ведёт к значительному ускорению работы. Наибольшая эффективность акселератора достигается на скриптах с большим количеством исходного кода, содержащих небольшое количество операций, ресурсоёмких при выполнении, даже, считанного количества раз (например, таких, как масштабирование изображений).

Важной особенностью является то, что разработчику нет необходимости заботиться о распределении и освобождении памяти. Ядро PHP реализует средства для автоматического управления памятью; вся выделенная память возвращается системе после завершения работы скрипта.

Расширения

Интерпретатор состоит из ядра и подключаемых модулей, *«расширений»*, представляющих собой динамические библиотеки. Расширения позволяют дополнить базовые возможности языка, предоставляя возможности для работы с базами данных, сокетами, динамической графикой, криптографическими библиотеками, документами формата PDF и тому подобным. Любой желающий может разработать своё собственное расширение и подключить его. Существует огромное количество расширений, как стандартных, так и созданных сторонними компаниями и энтузиастами, однако в стандартную поставку входит лишь несколько десятков хорошо зарекомендовавших себя. Множество расширений доступно в репозитории PECL.

2.3.6. Параметры настройки

Интерпретатор PHP имеет специальный конфигурационный файл - `php.ini`, содержащий множество настроек, изменение которых влияет на поведение интерпретатора. Имеется возможность отключить использование ряда функций, изменить ограничения на используемую скриптом оперативную память, время выполнения, объём загружаемых файлов, настроить журналирование ошибок, работу с сессиями и почтовыми сервисами, подключить дополнительные расширения, а также многое другое.

2.3.7. Режимы интерпретатора

Существует несколько способов использования интерпретатора PHP:

В качестве модуля к веб-серверу посредством SAPI или ISAPI (например, для Apache модуль mod_php). Веб-сервер запускается с загруженным модулем

и при обращении к php - скрипту он выполняется в окружении процесса веб-сервера. Это наиболее распространённый метод использования PHP в силу следующих причин:

- он наиболее удобен для отладки, так как ошибки PHP выводятся на экран (если данный режим включен в настройках), тогда как в режиме CGI ошибка приводит к генерации HTTP - кода 500 и дополнительную информацию необходимо искать в логах веб-сервера;
- имеется возможность менять некоторые настройки PHP «на лету» с помощью файла .htaccess, без перезагрузки веб - сервера;
- в режиме CGI нет возможности воспользоваться HTTP-аутентификацией;
- обладает более высокой скоростью работы.

В качестве CGI. В этом случае, при вызове скрипта веб - сервер вызывает /usr/bin/php - cgi /path/to/script.php. При этом создаётся новый процесс и скрипт выполняется в окружении php-cgi, которое, в зависимости от настроек, может сильно отличаться от окружения процесса веб-сервера. Более современным вариантом является FastCGI. Несмотря на то, что препроцессор PHP все равно будет перезапускаться при каждом запросе, преимуществом данного варианта перед предыдущим является более высокая степень безопасности, потому как PHP работает как отдельное приложение со своими правами.

В качестве скрипта командной строки, являющегося исполняемым файлом, который вызывается пользователем из командной строки; скрипт выполняется в окружении вызвавшего пользователя. В этом случае возможно использование PHP для создания клиентских GUI - приложений и решения административных задач в операционных системах UNIX, Linux, Microsoft Windows, Mac OS X и AmigaOS. Однако, в таком качестве он не получил распространение, отдавая пальму первенства Perl, Python и VBScript.

2.4 Движок базы данных MySQL

Теперь перейдем к вопросу о хранении данных. Для хранения данных, нам необходима база данных, а для организации и управления базой данных - СУБД. Система управления базами данных - специализированная программа (чаще комплекс программ), предназначенная для организации и ведения базы данных. Для создания и управления информационной системой СУБД необходима в той же степени, как для разработки программы на алгоритмическом языке необходим транслятор.

Основные функции СУБД:

- управление данными во внешней памяти (на дисках);
- управление данными в оперативной памяти с использованием дискового кэша;
- журнализация изменений, резервное копирование и восстановление базы данных после сбоев;
- поддержка языков БД (язык определения данных, язык манипулирования данными);

- обычно современная СУБД содержит следующие компоненты;
- ядро, которое отвечает за управление данными во внешней и оперативной памяти и журнализацию;
- процессор языка базы данных, обеспечивающий оптимизацию запросов на извлечение и изменение данных, и создание, как правило, машинно-независимого исполняемого внутреннего кода;
- подсистему поддержки времени исполнения, которая интерпретирует программы манипуляции данными, создающие пользовательский интерфейс с СУБД;
- а также сервисные программы (внешние утилиты), обеспечивающие ряд дополнительных возможностей по обслуживанию информационной системы.

2.4.1 Классификация СУБД

Системы управления базами данных классифицируются по следующим принципам:

- по модели данных;
- по типу управляемой базы данных СУБД разделяются на:
  - иерархические;
  - сетевые;
  - реляционные;
  - объектно-реляционные;
  - объектно-ориентированные.
- по архитектуре организации хранения данных:
  - локальные СУБД (все части локальной СУБД размещаются на одном компьютере);
  - распределенные СУБД (части СУБД могут размещаться на двух и более компьютерах).
- по способу доступа к БД:
  - файл-серверные;
  - клиент-серверные;
  - встраиваемые.

Встраиваемая СУБД - библиотека, которая позволяет унифицированным образом хранить большие объёмы данных на локальной машине. Доступ к данным может происходить через SQL либо через особые функции СУБД. Встраиваемые СУБД быстрее обычных клиент-серверных и не требуют установки сервера, поэтому востребованы в локальном ПО, которое имеет дело с большими объёмами данных (например, геоинформационные системы). Примеры: OpenEdge, SQLite, BerkeleyDB, один из вариантов Firebird, один из вариантов MySQL, Sav Zigzag, Microsoft SQL Server Compact, ЛИНТЕР.

Для реализации требуемой базы данных нами был выбран MySQL - это движок баз данных, возникший как попытка применить mSQL к собственным разработкам компании: таблицам, для которых использовались ISAM -

подпрограммы низкого уровня. В результате был выработан новый SQL - интерфейс, но API - интерфейс остался в наследство от mSQL. Откуда происходит название «MySQL» - доподлинно не известно. Разработчики дают два варианта: либо потому, что практически все наработки компании начинались с префикса *My*, либо в честь девочки по имени *My*, дочери Майкла Монти Видениуса, одного из разработчиков системы.

2.4.2 Описание MySQL

MySQL - свободный движок баз данных. MySQL является собственностью компании Oracle Corporation, получившей её вместе с поглощённой Sun Microsystems, осуществляющей разработку и поддержку приложения. Распространяется под GNU General Public License или под собственной коммерческой лицензией. Помимо этого разработчики создают функциональность по заказу лицензионных пользователей, именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации.

MySQL является решением для малых и средних приложений. Входит в состав серверов WAMP, LAMP и в портативные сборки серверов Денвер, XAMPP. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц. Благодаря открытой архитектуре и GPL - лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц.

26 февраля 2008 года Sun Microsystems приобрела MySQL AB за 1 миллиард долларов.

27 января 2010 года Oracle Corporation приобрела Sun Microsystems и включила MySQL в свою линейку СУБД.

Сообществом разработчиков MySQL созданы различные ответвления кода, такие как Drizzle, OurDelta, Percona Server, и MariaDB . Все эти ответвления уже существовали на момент поглощения компаний Sun и MySQL AB корпорацией Oracle.

2.4.3 Функциональность и версии

MySQL имеет API для языков Delphi, C, C++, Эйфель, Java, Лисп, Perl, PHP, Python, Ruby, Smalltalk и Tcl, библиотеки для языков платформы .NET, а также обеспечивает поддержку для ODBC посредством ODBC - драйвера MyODBC.

MySQL 4.0

Несмотря на то, что версия 4.0 является устаревшей, она всё ещё имеет значительное распространение. Основные возможности этой версии:

- практически полная реализация ANSI SQL - 99, плюс расширения;
- межплатформенная совместимость;
- независимые типы таблиц (MyISAM для быстрого чтения, InnoDB для транзакций и ссылочной целостности);
- транзакции;
- поддержка SSL;
- кэширование запросов;
- репликация: один головной сервер на одного подчинённого, много подчинённых на одного головного;
- полнотекстовая индексация и поиск с использованием типа таблиц MyISAM;
- внедрённая библиотека базы данных;
- поддержка Юникода (UTF-8);
- таблицы InnoDB, обеспечивающие соответствие требованиям ACID;
- встроенный сервер, позволяющий включать MySQL в автономные приложения.

MySQL 4.1

Рекомендованной версией на 2005 год является MySQL 4.1 вышла 27 октября 2004. Она содержит следующие нововведения:

- вложенные запросы и производные таблицы;
- новая система кодировок и сортировок;
- более быстрый и гибкий протокол клиент-сервер с поддержкой подготовленных запросов, обеспечивающий их оптимальное исполнение;
- новая программа установки и настройки для Microsoft Windows и Linux;
- защищённые через OpenSSL соединения клиент-сервер;
- высоко-оптимизированная библиотека, которая может быть использована в сторонних программах;
- полноценная поддержка Юникода (UTF - 8 и UCS2);
- стандартные пространственные типы данных GIS, для хранения географической информации;
- улучшенный полнотекстовый поиск и система помощи.

MySQL 5.0

Версия MySQL 5.0 вышла 24 октября 2005 года, в этой версии значительно расширена функциональность, которая ставит MySQL в один ряд с коммерческими СУБД. Если раньше СУБД MySQL обвиняли в недостаточной поддержке стандарта SQL, то с появлением пятой версии этой популярной базы данных, появилась практически полная поддержка стандарта SQL. MySQL 5.0 содержит следующие нововведения:

-хранимые процедуры и функции;

-обработчики ошибок;

-курсоры;

-триггеры;

-представления;

-информационная схема (так называемый системный словарь, содержащий метаданные).

MySQL 5.1

Версия MySQL 5.1 продолжает путь к стандарту SQL:2003. MySQL 5.1 содержит следующие нововведения.

Сегментирование - возможность разбить одну большую таблицу на несколько частей, размещенных в разных файловых системах, основываясь на определенной пользователем функции. При определенных условиях это может дать серьезное увеличение производительности и, кроме того, облегчает масштабирование таблиц.

Изменено поведение ряда операторов, для обеспечения большей совместимости со стандартом SQL2003.

Построчная репликация (англ. Row - based replication), при которой в бинарный лог будет записываться только информация о реально измененных строках таблицы вместо оригинального (и, возможно, медленного) текста запроса. Построчную репликацию можно использовать только для определенных типов sql - запросов, в терминах MySQL - смешанная репликация (англ. mixed replication).

Встроенный планировщик периодически запускаемых работ. По синтаксису добавление задачи похоже на добавление триггера к таблице, по идеологии - на crontab.

Дополнительный набор функций для обработки XML, реализация поддержки XPath.

Новые средства диагностики проблем и утилиты для анализа производительности. Расширены возможности по управлению содержимым лог - файлов, логи теперь могут быть сохранены и в таблицах general_log и slow_log. Утилита mysqlslap позволяет провести нагрузочное тестирование БД с записью времени реакции на каждый запрос.

Для упрощения операции обновления подготовлена утилита mysql_upgrade, которая выполнит проверку всех существующих таблиц на предмет совместимости с новой версией, и при необходимости выполнит надлежащие корректировки.

MySQL Cluster отныне выпущен как отдельный продукт, базирующийся на MySQL 5.1 и хранилище NDBCLUSTER.

Значительные изменения в работе MySQL Cluster, такие, как, например, возможность хранения табличных данных на диске.

Возврат к использованию встроенной библиотеки libmysqld, отсутствовавшей в MySQL 5.0.

API для плагинов, который позволяет загружать сторонние модули, расширяющие функциональность (например, полнотекстовый поиск), без перезапуска сервера.

Реализация парсера полнотекстового поиска в виде plug-in.

Новый тип таблиц Maria (устойчивый к сбоям клон MyISAM).

Тип таблиц Maria

Maria (начиная с версии 5.2.x - Aria) - новый MySQL тип таблиц для хранения данных. Maria представляет собой расширенную версию хранилища MyISAM, с добавлением средств сохранения целостности данных после краха.

Основные достоинства Maria.

В случае краха производится откат результатов выполнения текущей операции или возврат в состояние до команды LOCK TABLES. Реализация через ведение лога операций.

Возможность восстановления состояния из любой точки в логе операций, включая поддержку CREATE/DROP/RENAME/TRUNCATE. Может быть использовано для создания инкрементальных бэкапов, через периодическое копирование лог файла.

Поддержка всех форматов столбцов MyISAM, расширена новым форматом «rows-in-block», использующим страничный способ хранения данных, при котором данные в столбцах могут кэшироваться.

В будущем будет реализовано два режима: транзакционный и без отражения в логе транзакций, для не критичных данных.

Размер страницы данных равен 8Кб (в MyISAM 1Кб), что позволяет достичь более высокой производительности для индексов по полям фиксированного размера, но медленнее в случае индексирования ключей переменной длины.

MySQL 5.5

Ветка MySQL 5.5 базируется на невыпущенной серии MySQL 5.4 и содержит ряд значительных улучшений, связанных с повышением масштабируемости и производительности, среди которых:

- использование по умолчанию движка InnoDB;
- поддержка полусинхронного (semi-synchronous) механизма репликации, основанного на патчах к InnoDB от компании Google;
- улучшение функций по партицированию данных. Расширенный синтаксис для разбиения больших таблиц на несколько частей, размещенных в разных файловых системах (partitioning). Добавлены операции RANGE, LIST и метод оптимизации «partition pruning»;
- новый механизм оптимизации вложенных запросов и JOIN операций;
- переработана система внутренних блокировок;
- интегрированы патчи Google с оптимизацией работы InnoDB на CPU с большим количеством ядер.

MySQL 6.0

Версия MySQL 6.0 пока находится в стадии альфа-тестирования. Первоначально было принято решение о создании версии 5.2, однако вскоре эта версия была переименована в 6.0.

Одним из основных нововведений версии 6.0 планировалось сделать новый тип таблиц Falcon, разработанный в качестве потенциальной замены для InnoDB компании InnoBase, приобретённой компанией Oracle. В связи с приобретением в 2010 году Sun Microsystems тем же Oracle, судьба Falcon остается под вопросом.

Максимальные размеры таблиц

MySQL 3.22: до 4 Гб.

MySQL 3.23+: До 8 миллионов терабайт. (2 ^ 63).

Размер таблицы ограничен её типом. В общем случае тип MyISAM ограничен предельным размером файла в файловой системе операционной системы. Например, в NTFS этот размер теоретически может быть до 32 эксабайт. В случае InnoDB одна таблица может храниться в нескольких файлах, представляющих единое табличное пространство. Размер последнего может достигать 64 терабайт.

2.5 AJAX

AJAX (от англ. Asynchronous Javascript and XML - «асинхронный JavaScript и XML») - подход к построению интерактивных пользовательских интерфейсов веб-приложений, заключающийся в «фоновом» обмене данными браузера с веб-сервером. В результате, при обновлении данных, веб-страница не перезагружается полностью, и веб-приложения становятся более быстрыми и удобными.

По-английски AJAX произносится как эй-джэкс, по-русски довольно распространено ая́кс.

AJAX - не самостоятельная технология, а концепция использования нескольких смежных технологий. AJAX базируется на двух основных принципах использование технологии динамического обращения к серверу «на лету», без перезагрузки всей страницы полностью, например:

- с использованием XMLHttpRequest (основной объект);
- через динамическое создание дочерних фреймов;
- через динамическое создание тега <script>;
- использование DHTML для динамического изменения содержания страницы.

В качестве формата передачи данных обычно используются JSON или XML.

2.5.1 История

Впервые термин AJAX был публично использован 18 февраля 2005 года в статье Джесси Джеймса Гарретта (Jesse James Garrett) «Новый подход к веб-приложениям» (Ajax: A New Approach to Web Applications). Гарретт придумал термин, когда ему пришлось как-то назвать новый набор технологий, предлагаемый им клиенту.

Однако в той или иной форме многие технологии были доступны и использовались гораздо раньше, например, в подходе «Remote Scripting», предложенном компанией Microsoft в 1998 году, или с использованием HTML элемента IFRAME, появившегося в Internet Explorer 3 в 1996 году.

AJAX стал особенно популярен после использования его компанией Google в сервисах Gmail, Google Maps и Google Suggest.

2.5.2 Преимущества

Экономия трафика

Использование AJAX позволяет значительно сократить трафик при работе с веб-приложением благодаря тому, что часто вместо загрузки всей страницы достаточно загрузить только изменившуюся часть, как правило, довольно небольшую.

Уменьшение нагрузки на сервер

AJAX позволяет несколько снизить нагрузку на сервер. К примеру, на странице работы с почтой, когда вы отмечаете прочитанные письма, серверу достаточно внести изменения в базу данных и отправить клиентскому скрипту сообщение об успешном выполнении операции без необходимости повторно создавать страницу и передавать её клиенту.

Ускорение реакции интерфейса, поскольку нужно загрузить только изменившуюся часть, пользователь видит результат своих действий быстрее.

2.5.3 Недостатки

Отсутствие интеграции со стандартными инструментами браузера.

Динамически создаваемые страницы не регистрируются браузером в истории посещения страниц, поэтому не работает кнопка «Назад», предоставляющая пользователям возможность вернуться к просмотренным ранее страницам, но существуют скрипты, которые могут решить эту проблему.

Другой недостаток изменения содержимого страницы при постоянном URL заключается в невозможности сохранения закладки на желаемый материал. Частично решить эти проблемы можно с помощью динамического изменения идентификатора фрагмента (части URL после #), что позволяют многие браузеры.

Динамически загружаемое содержимое недоступно поисковикам (если не проверять запрос, обычный он или XMLHttpRequest).

Поисковые машины не могут выполнять JavaScript, поэтому разработчики должны позаботиться об альтернативных способах доступа к содержимому сайта.

Старые методы учёта статистики сайтов становятся неактуальными.

Многие сервисы статистики ведут учёт просмотров новых страниц сайта. Для сайтов, страницы которых широко используют AJAX, такая статистика теряет актуальность.

Усложнение проекта.

Перераспределяется логика обработки данных - происходит выделение и частичный перенос на сторону клиента процессов первичного форматирования данных. Это усложняет контроль целостности форматов и типов. Конечный эффект технологии может быть нивелирован необоснованным ростом затрат на кодирование и управление проектом, а также риском снижения доступности сервиса для конечных пользователей.

Требуется включенный JavaScript в браузере.

2.5.4 Альтернативы

В хронологическом порядке:

- java - апплеты, позднее технология JavaFX;
- стек технологий Flash в виде ActionScript 3, Adobe Flex и Flash Remoting составляет технологическую основу RIA (Rich Internet Applications) активно продвигаемых Macromedia (теперь часть Adobe).

2.5.5 XMLHttpRequest

XMLHTTP (XMLHttpRequest, XHR) - API, предоставляемый веб-клиентом, для обмена информацией между клиентом и сервером посредством протоколов HTTP и HTTPS. Информация может передаваться в любом текстовом формате, например, в XML, HTML или JSON. Позволяет осуществлять HTTP - запросы к серверу без перезагрузки страницы.

XMLHTTP является важной составляющей технологии AJAX (Asynchronous JavaScript And XML), используется многими сайтами для создания динамичных, быстро реагирующих на запросы пользователя приложений. Например, XMLHTTP используется такими сайтами как Gmail, Google Suggest, MSN Virtual Earth. XMLHTTP работает только с файлами, находящимися на том же домене, что и использующая XMLHTTP страница. Как и в случае JavaScript, это сделано в целях безопасности (cross-site scripting).

Хотя в названии присутствует аббревиатура XML, технология не налагает ограничений на формат передаваемых данных. Данные можно пересылать как в виде XML, так и в JSON, HTML или просто неструктурированным текстом. Разработчик может самостоятельно создать формат для передачи данных. Однако нужно учитывать, что при пересылке используется текстовый протокол HTTP и потому данные должны передаваться в виде текста (т.е. бинарные данные следует кодировать, к примеру, в base64).

Впервые был разработан компанией Microsoft, появившись в компоненте Outlook Web Access программного продукта Microsoft Exchange Server 2000. Он был назван IXMLHTTPRequest. Позднее, наработки были включены в состав MSXML 2.0 в виде объекта ActiveX, доступного через JScript, VBScript или другие скриптовые языки, поддерживающиеся браузером. MSXML 2.0 был включён в состав браузера Internet Explorer 5.

Программисты проекта Mozilla затем разработали совместимую версию, называющуюся nsIXMLHttpRequest в Mozilla 0.6. Доступ к компоненту был реализован через JavaScript - объект, названный XMLHttpRequest. Однако, полной функциональности удалось добиться только в Mozilla 1.0. В дальнейшем поддержка XMLHttpRequest появилась в браузерах Safari 1.2, Opera 8.01 и в других.

Последняя официальная спецификация - версия 1.0 (XMLHttpRequest Version 1.0 от 3 августа 2010 года), которая имеет статус кандидата в

рекомендации (Candidate Recommendation) и версия 2.0 (XMLHttpRequest Level 2 от 7 сентября 2010 года), имеющая статус черновика. Во второй версии вводятся обработчики событий прогресса, поддержка кросс-доменных запросов и работа с бинарными данными.

2.5.6 XML

XML (англ. eXtensible Markup Language - расширяемый язык разметки; произносится) - рекомендованный Консорциумом Всемирной паутины язык разметки, фактически представляющий собой свод общих синтаксических правил. XML - текстовый формат, предназначенный для хранения структурированных данных (взамен существующих файлов баз данных), для обмена информацией между программами, а также для создания на его основе более специализированных языков разметки (например, XHTML). XML является упрощённым подмножеством языка SGML.

Достоинства:

- XML - язык разметки, позволяющий стандартизировать вид файлов-данных, используемых компьютерными программами, в виде текста, понятного человеку;
- XML поддерживает Юникод;
- в формате XML могут быть описаны такие структуры данных как записи, списки и деревья;
- XML - это само документируемый формат, который описывает структуру и имена полей, так же как и значения полей;
- XML имеет строго определённый синтаксис и требования к анализу, что позволяет ему оставаться простым, эффективным и непротиворечивым. Одновременно с этим, разные разработчики не ограничены в выборе экспрессивных методов (например, можно моделировать данные, помещая значения в параметры тегов или в тело тегов, можно использовать различные языки и нотации для именования тегов и т.д.);
- XML - формат, основанный на международных стандартах;
- иерархическая структура XML подходит для описания практически любых типов документов, кроме аудио и видео мультимедийных потоков, растровых изображений, сетевых структур данных и двоичных данных;
- XML представляет собой простой текст, свободный от лицензирования и каких-либо ограничений;
- XML не зависит от платформы;
- XML является подмножеством SGML (который используется с 1986 года). Уже накоплен большой опыт работы с языком и созданы специализированные приложения;
- XML не накладывает требований на порядок расположения атрибутов в элементе и вложенных элементов разных типов, что существенно облегчает выполнение требований обратной совместимости;

- в отличие от бинарных форматов, XML содержит метаданные об именах, типах и классах описываемых объектов, по которым приложение может обработать документ неизвестной структуры (например, для динамического построения интерфейсов);
- XML имеет реализации парсеров для всех современных языков программирования;
- существует стандартный механизм преобразования XSLT, реализации которого встроены в браузеры, операционные системы, веб-серверы;
- XML поддерживается на низком аппаратном, микропрограммном и программном уровнях в современных аппаратных решениях.

Недостатки:

- синтаксис XML избыточен;
- размер XML - документа существенно больше бинарного представления тех же данных. В грубых оценках величину этого фактора принимают за 1 порядок (в 10 раз);
- размер XML - документа существенно больше, чем документа в альтернативных текстовых форматах передачи данных (например, JSON, YAML, Protocol Buffers) и особенно в форматах данных, оптимизированных для конкретного случая использования;
- избыточность XML может повлиять на эффективность приложения. Возрастает стоимость хранения, обработки и передачи данных;
- XML содержит метаданные (об именах полей, классов, вложенности структур), и одновременно XML позиционируется как язык взаимодействия открытых систем. При передаче между системами большого количества объектов одного типа (одной структуры), передавать метаданные повторно нет смысла, хотя они содержатся в каждом экземпляре XML описания;
- для большого количества задач не нужна вся мощь синтаксиса XML и можно использовать значительно более простые и производительные решения;
- неоднозначность моделирования;
- нет общепринятой методологии для моделирования данных в XML, в то время как для реляционной модели и объектно-ориентированной такие средства разработаны и базируются на реляционной алгебре, системном подходе и системном анализе;
- в природе есть множество объектов и явлений, для описания которых разные структуры данных (сетевая, реляционная, иерархическая) являются естественными, и отображение объекта в неестественную для него модель является болезненным для его сути. В случае с реляционной и иерархической моделями определены процедуры декомпозиции, обеспечивающие относительную однозначность, чего нельзя сказать о сетевой модели;
- в результате большой гибкости языка и отсутствия строгих ограничений, одна и та же структура может быть представлена множеством способов (различными разработчиками), например, значение может быть записано как атрибут тега или как тело тега и т.д.;

- поддержка многих языков в именовании тегов дает возможность назвать, например `вес` русским словом, в таком случае компьютер никак не сможет установить соответствия этого поля с полем weight в англоязычной версии программы и с полями в версиях модели объекта на множестве других языков;
- XML не содержит встроенной в язык поддержки типов данных. В нём нет строгой типизации, то есть понятий «целых чисел», «строк», «дат», «булевых значений» и т.д.;
- иерархическая модель данных, предлагаемая XML, ограничена по сравнению с реляционной моделью и объектно-ориентированными графами и сетевой моделью данных;
- выражение неиерархических данных (например, графов) требует дополнительных усилий;
- Кристофер Дейт, специалист в области реляционных баз данных, автор классического учебника «An Introduction to Database Systems», отмечал, что «…XML является попыткой заново изобрести иерархические базы данных…» (в 1980-е года иерархические базы данных были вытеснены реляционными базами данных);
- пространства имён XML сложно использовать и их сложно реализовывать в XML - парсерах;
- существуют другие, обладающие сходными с XML возможностями, текстовые форматы данных, которые обладают более высоким удобством чтения человеком (YAML, JSON, SweetXML, XF).

2.5.7 JSON

JSON (англ. JavaScript Object Notation) - текстовый формат обмена данными, основанный на JavaScript и обычно используемый именно с этим языком. Как и многие другие текстовые форматы, JSON легко читается людьми.

Несмотря на происхождение от JavaScript (точнее, от подмножества языка стандарта ECMA - 262 1999 года), формат считается язык независимым и может использоваться практически с любым языком программирования. Для многих языков существует готовый код для создания и обработки данных в формате JSON.

JSON (англ. JavaScript Object Notation) - текстовый формат обмена данными, основанный на JavaScript и обычно используемый именно с этим языком. Как и многие другие текстовые форматы, JSON легко читается людьми.

Несмотря на происхождение от JavaScript (точнее, от подмножества языка стандарта ECMA-262 1999 года), формат считается язык независимым и может использоваться практически с любым языком программирования. Для многих языков существует готовый код для создания и обработки данных в формате JSON.

За счёт своей лаконичности по сравнению с XML, формат JSON может быть более подходящим для сериализации сложных структур. Если говорить о веб-приложениях, в таком ключе он уместен в задачах обмена данными как между браузером и сервером (AJAX), так и между самими серверами (программные HTTP - интерфейсы). Формат JSON так же хорошо подходит для хранения сложных динамических структур в реляционных базах данных или файловом кэше.

Поскольку формат JSON является подмножеством синтаксиса языка JavaScript, то он может быть быстро десериализован встроенной функцией eval(). Кроме того, возможна вставка вполне работоспособных JavaScript-функций. В PHP, начиная с версии 5.2.0, поддержка JSON включена в ядро в виде функций json_decode() и json_encode(), которые сами преобразуют типы данных JSON в соответствующие типы PHP и наоборот.

JSON строится на двух структурах:

- набор пар имя/значение. В различных языках это реализовано как объект, запись, структура, словарь, хэш-таблица, список с ключом или ассоциативный массив;
- пронумерованный набор значений. Во многих языках это реализовано как массив, вектор, список или последовательность;
- универсальные структуры данных. Теоретически, все современные языки программирования поддерживают их в той или иной форме. Так как JSON используется для обмена данными между различными языками программирования, то имеет смысл строить его на этих структурах.

В JSON используются их следующие формы:

- объект-это неупорядоченное множество пар имя/значение, заключённое в фигурные скобки { }. Между именем и значением стоит символ «:», а пары имя/значение разделяются запятыми;
- массив (одномерный) - это множество значений, имеющих порядковые номера (индексы). Массив заключается в квадратные скобки [ ]. Значения отделяются запятыми;
- значение может быть строкой в двойных кавычках, числом, значением true или false, объектом, массивом, или значением null. Эти структуры могут быть вложены друг в друга.

Строка - это упорядоченное множество из нуля или более символов юникода, заключенное в двойные кавычки, с использованием escape-последовательностей начинающихся с обратной косой черты (backslash). Символы представляются простой строкой.

Строка очень похожа на строку в языках C и Java. Число тоже очень похоже на C или Java - число, за исключением того, что используется только десятичный формат. Пробелы могут быть вставлены между любыми двумя символами.

2.5.8 JavaScript

JavaScript - объектно-ориентированный скриптовый язык программирования. Является диалектом языка ECMAScript.

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса.

На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на Java, но при этом лёгким для использования непрограммистами. Языком JavaScript не владеет какая - либо компания или организация, что отличает его от ряда языков программирования, используемых в веб - разработке.

Название «JavaScript» является зарегистрированным товарным знаком компании Oracle Corporation.

Предпосылки

В 1992 году компания Nombas (впоследствии приобретённая Openwave (англ.)) начала разработку встраиваемого скриптового языка Cmm (Си-минус-минус), который, по замыслу разработчиков, должен был стать достаточно мощным, чтобы заменить макросы, сохраняя при этом схожесть с Си, чтобы разработчикам не составляло труда изучить его. Главным отличием от Си была работа с памятью. В новом языке всё управление памятью осуществлялось автоматически: не было необходимости создавать буферы, объявлять переменные, осуществлять преобразование типов. В остальном языки сильно походили друг на друга: в частности, Cmm поддерживал стандартные функции и операторы Си. Cmm был переименован в ScriptEase, поскольку исходное название звучало слишком негативно, а упоминание в нём Си «отпугивало» людей.

На основе этого языка был создан проприетарный продукт CEnvi. В конце ноября 1995 года Nombas разработала версию CEnvi, внедряемую в веб-страницы. Страницы, которые можно было изменять с помощью скриптового языка, получили название Espresso Pages - они демонстрировали использование скриптового языка для создания игры, проверки пользовательского ввода в формы и создания анимации. Espresso Pages позиционировались как демоверсия, призванная помочь представить, что случится, если в браузер будет внедрён язык Cmm. Работали они только в 16-битовом Netscape Navigator под управлением Windows.

Перед Бренданом Айхом, нанятым в компанию Netscape 4 апреля 1995 года, была поставлена задача внедрить язык программирования Scheme или что-то похожее в браузер Netscape. Поскольку требования были размыты, Айха перевели в группу, ответственную за серверные продукты, где он проработал

месяц, занимаясь улучшением протокола HTTP. В мае разработчик был переброшен обратно, в команду, занимающуюся клиентской частью (браузером), где он немедленно начал разрабатывать концепцию нового языка программирования. Менеджмент разработки браузера, включая Тома Пакина (Tom Paquin), Михаэля Тоя (англ.), Рика Шелла (Rick Schell), был убеждён, что Netscape должен поддерживать язык программирования, встраиваемый в HTML-код страницы.

Помимо Брендона Айха в разработке участвовали со основатель Netscape Communications Марк Андрисин (англ.) и со-основатель Sun Microsystems Билл Джой: чтобы успеть закончить работы над языком к релизу браузера, компании заключили соглашение о сотрудничестве в разработке. Они ставили перед собой цель обеспечить «язык для склеивания» составляющих частей веб-ресурса: изображений, плагинов, Java - апплетов, который был бы удобен для веб-дизайнеров и программистов, не обладающих высокой квалификацией.

Первоначально язык назывался LiveScript и предназначался как для программирования на стороне клиента, так и для программирования на стороне сервера (там он должен был называться LiveWire). На синтаксис оказали влияние языки Си и Java, и, поскольку Java в то время было модным словом, 4 декабря 1995 года LiveScript переименовали в JavaScript, получив соответствующую лицензию у Sun. Анонс JavaScript со стороны представителей Netscape и Sun состоялся накануне выпуска второй бета-версии Netscape Navigator. В нём декларируется, что 28 лидирующих ИТ-компаний выразили намерение использовать в своих будущих продуктах JavaScript как объектный скриптовый язык с открытым стандартом.

В 1996 году компания Microsoft выпустила аналог языка JavaScript, названный JScript. Анонсирован этот язык был 18 июля 1996 года. Первым браузером, поддерживающим эту реализацию, был Internet Explorer 3.0.

По инициативе компании Netscape была проведена стандартизация языка ассоциацией ECMA. Стандартизированная версия имеет название ECMAScript, описывается стандартом ECMA-262. Первой версии спецификации соответствовал JavaScript версии 1.1, а также языки JScript и ScriptEasy.

Популярность

В статье «The World’s Most Misunderstood Programming Language Has Become the World’s Most Popular Programming Language» ((рус.) «Самый не понимаемый язык программирования в мире стал самым популярным в мире языком программирования»). Дуглас Крокфорд (англ.) утверждает, что лидирующую позицию JavaScript занял в связи с развитием AJAX, поскольку браузер стал превалирующей системой доставки приложений. Он также констатирует растущую популярность JavaScript, то, что этот язык встраивается в приложения, отмечает значимость языка.

Согласно рейтингу Tiobe (англ.), базирующемуся на данных поисковых систем Google, MSN, Yahoo!, Википедия и YouTube, в феврале 2011 года JavaScript находится на 9 месте (год назад на 10).

По данным Black Duck Software (англ.) в разработке открытого программного обеспечения доля использования JavaScript выросла. 36 % проектов, релизы которых состоялись с августа 2008 по август 2009 гг., включают JavaScript, наиболее часто используемый язык программирования с быстрорастущей популярностью. 80 % открытого программного обеспечения использует Си, C++, Java, Shell и JavaScript. При этом JavaScript - единственный из этих языков, чья доля использования увеличилась (более чем на 2 процента, если считать в строках кода).

JavaScript является самым популярным языком программирования, используемым для разработки веб-приложений на стороне клиента.

Возможности языка

JavaScript обладает рядом свойств объектно-ориентированного языка, но реализованное в языке прототипирование обусловливает отличия в работе с объектами по сравнению с традиционными объектно-ориентированными языками. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам-функции как объекты первого класса, объекты как списки, карринг, анонимные функции, замыкания - что придаёт языку дополнительную гибкость.

Несмотря на схожий с Си синтаксис, JavaScript по сравнению с языком Си имеет коренные отличия:

- объекты, с возможностью интроспекции;
- функции как объекты первого класса;
- автоматическое приведение типов;
- автоматическая сборка мусора;
- анонимные функции.

В языке отсутствуют такие полезные вещи, как:

- модульная система: JavaScript не предоставляет возможности управлять зависимостями и изоляцией областей видимости;
- стандартная библиотека: в частности, отсутствует интерфейс программирования приложений по работе с файловой системой, управлению потоками ввода/вывода, базовых типов для бинарных данных;
- стандартные интерфейсы к веб - серверам и базам данных;
- система управления пакетами, которая бы отслеживала зависимости и автоматически устанавливала их.

Семантика и синтаксис

Синтаксис языка JavaScript во многом напоминает синтаксис Си и Java, семантически же язык гораздо ближе к Self, Smalltalk или даже Лиспу.

В JavaScript:

- все идентификаторы регистрозависимы;
- в названиях переменных можно использовать буквы, подчёркивание, символ доллара, арабские цифры;
- названия переменных не могут начинаться с цифры;
- для оформления однострочных комментариев используются `//`, многострочные и внутри строчные комментарии начинаются с `/*` и заканчиваются `*/`.

Структура языка

Структурно JavaScript можно представить в виде объединения трёх чётко различимых друг от друга частей:

- ядро (ECMAScript);
- объектная модель браузера (Browser Object Model или BOM (de));
- объектная модель документа (Document Object Model или DOM).

Если рассматривать JavaScript в отличных от браузера окружениях, то объектная модель браузера и объектная модель документа могут не поддерживаться.

Объектную модель документа иногда рассматривают как отдельную от JavaScript сущность, что согласуется с определением DOM как независимого от языка интерфейса документа. В противоположность этому ряд авторов находят BOM и DOM тесно взаимосвязанными.

Ядро

ECMAScript не является браузерным языком и на самом деле в нём не определяются методы ввода и вывода информации. Это скорее основа для построения скриптовых языков. Спецификация ECMAScript описывает типы данных, инструкции, ключевые и зарезервированные слова, операторы, объекты, регулярные выражения, не ограничивая авторов производных языков в расширении их новыми составляющими.

Объектная модель браузера

Объектная модель браузера - браузероспецифичная часть языка, являющаяся прослойкой между ядром и объектной моделью документа. Основное предназначение объектной модели браузера - управление окнами браузера и обеспечение их взаимодействия. Каждое из окон браузера представляется объектом window, центральным объектом BOM. Объектная модель браузера на данный момент не стандартизирована, однако спецификация находится в разработке WHATWG и W3C.

Помимо управления окнами, в рамках объектной модели браузера, браузерами обычно обеспечивается поддержка следующих сущностей:

- управление фреймами;
- поддержка задержки в исполнении кода и зацикливания с задержкой;
- системные диалоги;
- управление адресом открытой страницы;
- управление информацией о браузере;
- управление информацией о параметрах монитора;
- ограниченное управление историей просмотра страниц;
- поддержка работы с HTTP cookie.

Объектная модель документа

Объектная модель документа - интерфейс программирования приложений для HTML и XML - документов. Согласно DOM, документу можно поставить в

соответствие дерево объектов, обладающих рядом свойств, которые позволяют производить с ним различные манипуляции:

- получение узлов;
- изменение узлов;
- изменение связей между узлами;
- удаление узлов.

Веб-приложения

JavaScript используется в клиентской части веб-приложений: клиент-серверных программ, в котором клиентом выступает браузер, а сервером - веб - сервер, имеющих распределённую между сервером и клиентом логику. Обмен информацией в веб - приложениях происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются кроссплатформенными сервисами.

AJAX

JavaScript используется в AJAX, популярном подходе к построению интерактивных пользовательских интерфейсов веб-приложений, заключающемся в «фоновом» асинхронном обмене данными браузера с веб-сервером. В результате, при обновлении данных веб - страница не перезагружается полностью и интерфейс веб-приложения становится быстрее, чем это происходит при традиционном подходе (без применения AJAX).

Comet

Comet - зонтичный термин, описывающий механизм работы веб-приложений, использующих постоянные HTTP-соединения, что позволяет веб-серверу отправлять данные браузеру без дополнительного запроса со стороны браузера. Для таких приложений используются технологии, непосредственно поддерживаемые браузерами. В частности, в них широко используется JavaScript.

Браузерные операционные системы

JavaScript широко используется в браузерных операционных системах. Так, например, исходный код IndraDesktop WebOS на 75 % состоит из JavaScript, код браузерной операционной системы IntOS - на 70 %. Доля JavaScript в исходном коде eyeOS-5 %, однако и в рамках этой операционной системы JavaScript играет важную роль, участвуя в визуализации на клиенте и являясь необходимым механизмом для коммуницирования клиента и сервера.

Букмарклеты

JavaScript используется для создания небольших программ, размещаемых в закладки браузера. При этом используются URL-адреса со спецификатором javascript.

Пользовательские скрипты в браузере

Пользовательские скрипты в браузере - это программы, написанные на JavaScript, выполняемые в браузере пользователя при загрузке страницы. Они позволяют автоматически заполнять формы, переформатировать страницы, скрывать нежелательное содержимое и встраивать желательное для отображения содержимое, изменять поведение клиентской части веб-приложений, добавлять элементы управления на страницу и т.д.

Для управления пользовательскими скриптами в Mozilla Firefox используется расширение Greasemonkey; Opera и Google Chrome предоставляют средства поддержки пользовательских скриптов и возможности для выполнения ряда скриптов Greasemonkey.

Серверные приложения

Приложения, написанные на JavaScript, могут исполняться на серверах, использующих Java 6 и более поздних версий. Это обстоятельство используется для построения серверных приложений, позволяющих обрабатывать JavaScript на стороне сервера.

JavaScript на стороне сервера используется в проектах Google. Так, например, Google Sites допускает подстройку с помощью JavaScript-сценариев, исполняемых движком Rhino.

Мобильные приложения

Перевод мобильных устройств Palm на использование Palm webOS в качестве операционной системы с Mojo SDK в качестве комплекта средств разработки позволяет использовать JavaScript в качестве языка разработки мобильных приложений.

Виджеты

Виджет - вспомогательная мини-программа, графический модуль которой размещается в рабочем пространстве соответствующей родительской программы, служащая для украшения рабочего пространства, развлечения, решения отдельных рабочих задач или быстрого получения информации из интернета без помощи веб-браузера. JavaScript используется как для реализации виджетов, так и для реализации движков виджетов. В частности, при помощи JavaScript реализованы Apple Dashboard, Microsoft Gadgets, Yahoo!_Widgets, Google Gadgets, Klipfolio Dashboard.

Прикладное программное обеспечение

JavaScript используется для написания прикладного ПО. Одним из ярких примеров является Mozilla Firefox, 57 % исходного кода, которого написано на JavaScript (для сравнения, следующим языком программирования по объёму кодовой базы Firefox является C++, доля которого составляет 17 %).

Google Chrome OS в качестве прикладного ПО использует веб-приложения.

В окружении рабочего стола GNOME имеется возможность создавать на JavaScript программы, оперирующие с библиотеками GNOME при помощи Gjs, Seed.

Манипуляция объектами приложений

JavaScript также находит применение в качестве скриптового языка доступа к объектам приложений. Платформа Mozilla (XUL/Gecko) использует JavaScript. Среди сторонних продуктов, например, Java, начиная с версии 6, содержит встроенный интерпретатор JavaScript на базе Rhino. Сценарии JavaScript поддерживаются в таких приложениях Adobe, как Adobe Photoshop, Adobe Dreamweaver, Adobe Illustrator и Adobe InDesign.

Офисные приложения

JavaScript используется в офисных приложениях для автоматизации рутинных действий, написания макросов, организации доступа со стороны веб-служб.

Microsoft Office

В Excel Services 2010 добавились два новых интерфейса программирования приложений: REST API и JavaScript Object Model (JSOM):

- Excel Services 2010 REST API позволяет осуществлять доступ к объектам рабочих книг, таким как таблицы, диаграммы и именованные серии данных; получать изображения, HTML, Atom, рабочие книги, устанавливать значения и обновлять вычисления перед запрашиванием элементов;
- JSOM даёт возможность реагировать на действия пользователя в отношении Excel Web Access (EWA), программно взаимодействовать с составляющими EWA. Использование JSOM осуществляется при помощи помещения кода JavaScript на страницу, содержащую компоненты EWA.

OpenOffice.org

JavaScript - один из языков программирования, используемых для написания макросов в приложениях, входящих в состав OpenOffice.org. В OpenOffice.org интегрирован интерпретатор JavaScript Rhino. По состоянию на декабрь 2009 года поддержка JavaScript носила ограниченный характер. Ограничения, присущие разработке макросов OpenOffice.org на JavaScript:

- среда выполнения JavaScript поддерживает загрузку лишь тех классов Java, которые развёрнуты сценарием JavaScript;
- среда выполнения JavaScript не предоставляет сообщения об ошибках, произошедших во время выполнения скрипта;
- ещё не реализована поддержка интерактивной разработки JavaScript-сценариев.

В OpenOffice.org имеется редактор и отладчик JavaScript - сценариев.

Обучение информатике

Язык обладает пропедевтической ценностью, позволяя сочетать при обучении информатике ((англ.) Computer science) интенсивную практику программирования и широту используемых технологий. Преподавание данного языка в школе позволяет создать базу для изучения веб-программирования, использовать на уроках творческие проекты. Соответствующий курс позволяет обеспечить углубленный уровень изучения информатики и его имеет смысл включать в элективные курсы углубленного уровня подготовки.

JavaScript - подходящий язык для обучения программированию игр. По сравнению с альтернативами, он функционально достаточен, прост в изучении и в применении, снижает сложность для обучения, мотивирует обучаемых делиться своими играми с другими.

Не включённые в книгу Николаса Закаса «Professional JavaScript for Web Developers» части о реализации на JavaScript классических алгоритмов, техник, структур данных, послужили началу проекта Computer science in JavaScript.

2.6 Разработка

В соответствии с поставленной задачей разработка системы, удовлетворяющей всем установленным требованиям была разделена мной на два основных этапа: первый - разработка базы данных для CMS RIX, второй- разработка приложения - системы управления базами данных.

При разработке установил себе четкий перечень принципов в программировании и создании пользовательских интерфейсов, которыми заручился руководствоваться. Это существенно облегчает работу с программой для пользователей и сопровождающих программистов.

2.7 Общие принципы

Основа разработки - правильная и четкая структура данных. Если в программе есть правильная структура данных то нарастить на эту структуру удобные интерфейсы, красивый дизайн проще и легче. Сделать плохо структурированную программу удобной и гибко - сложно или даже невозможно:

- разработка интуитивно понятных программных продуктов. Пользователи в большинстве своем не любят читать документацию, и часто не имеют на это достаточного количество времени;
- разработка должна быть дружественной пользователю. Она должна быть для пользователя, а не пользователь для нее. Из этого следует несколько выводов;
- программа не должна по возможности задавать вопросы пользователю, наоборот она должна отвечать на них;
- документация нужна любой разработке, но пользователь имеет ПРАВО не пользоваться ей. Документация не должна служить "затычкой"

неправильных решений в области структуры данных и пользовательских интерфейсов;

- разработка должна быть легко конфигурируема под вкусы большинства пользователей с помощью легко настаиваемых параметров. Параметры по умолчанию должны быть настроены максимально удобно и под наибольшую аудиторию пользователей. То есть, средний пользователь программы может продолжительный период пользования программой работать без захода в диалоги настройки программы;
- при разработке учитываются рекомендации ведущих производителей программных продуктов, таких например как LogyCom ASTRUM, это позволяет пользователям знакомых с продуктами этих производителей быстрее разобраться с моими разработками;
- интерфейс программы максимально стандартизуется, то есть формы документов, справочников приближены к друг другу для того чтобы пользователю понимающему интерфейсу одного блока программы, легко работать с другим новым для него блоком;
- диалоговые формы программ должны быть аккуратны, просты и удобны, кроме того элементы форм должны быть выровнены по сетке. Все интерфейсы должны создаваться в одном стиле, который учитывает основные принципы дизайна;
- программа пишется для максимально возможного круга пользователей, и поэтому она должна быть как можно полнее подходить максимальному кругу пользователей, без дополнительного конфигурирования ее сторонними программистами (разработчиками);
- минимизация исключительных ситуаций. Если все же исключительная ситуация появилась, то пользователю должны быть выведены краткие и в тоже время максимально понятные сообщения;
- любую программу, даже самую универсальную и гибкую, возможно понадобится дополнительно конфигурировать и изменять в соответствии с изменившимися потребностями и вкусами пользователей. Исходя из этого мои принципы распространяются не только на видимую часть программы, но и на ее внутренний механизм.

2.6.1 Принципы программирования

Программа должна быть максимально конфигурируемой без автора программы. Поэтому я по возможности придерживаюсь принципов открытых кодов (Open source):

- программные коды разработок должны быть понятными для программистов. Для этого я придерживаюсь основных принципов программирования принятых в широких кругах, а так же некоторых собственных;
- построение кода лесенкой, в качестве выравнивания используется двойной пробел. После запятой используется разделитель пробел.

Исключительные случаи обрабатываются в начале, для того чтобы вложенность лесенки была минимальной.

Проектирование базы данных

На этапе проектирования создается модель базы данных, в которой отображаются все основные предъявленные требования конечных пользователей, проводится логическое и физическое проектирование и привязка к выбранной системе управления базами данных (СУБД).

Построение логической и физической моделей данных является основной частью этапа проектирования базы данных. Полученная в процессе анализа информационная модель сначала преобразуется в логическую, а затем в физическую модель данных. На рисунке 2.2 представлена логическая схема.

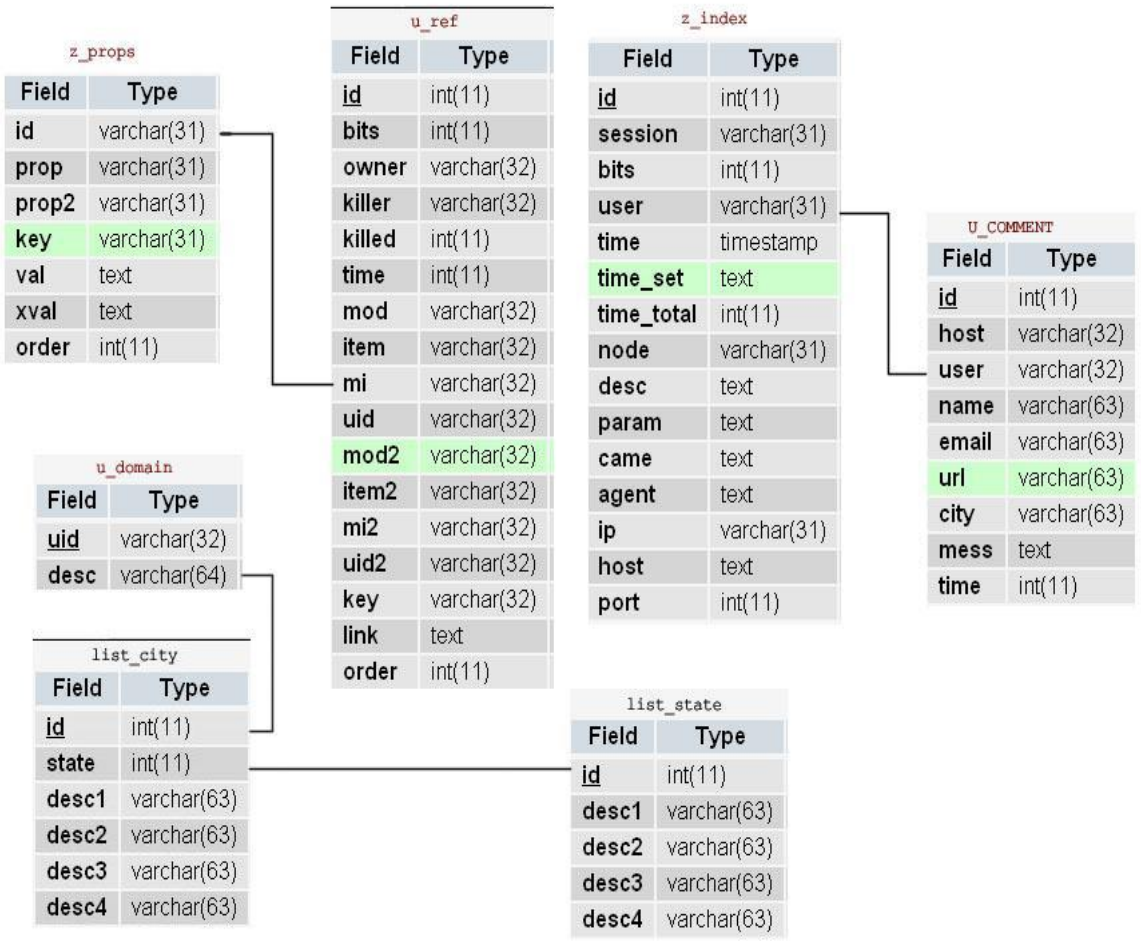


Рисунок 2.2 - Логическая схема информационной модели

На данном этапе создается абстрактная структура базы данных. Модель должна быть максимально интуитивно понятна всем разработчикам системы. В ней отражаются все основные информационные потоки и их взаимодействие.

U_REF - таблица отношений между объектами, представлена на рисунке 2.3.

u_ref

Field	Type
id	int(11)
bits	int(11)
owner	varchar(32)
killer	varchar(32)
killed	int(11)
time	int(11)
mod	varchar(32)
item	varchar(32)
mi	varchar(32)
uid	varchar(32)
mod2	varchar(32)
item2	varchar(32)
mi2	varchar(32)
uid2	varchar(32)
key	varchar(32)
link	text
order	int(11)

Рисунок 2.3 - Таблица U_REF

Z_PROPS - z выступает за "the" таблица свойств объектов. Таблица представлена на рисунке 2.4.

z_props

Field	Type
id	varchar(31)
prop	varchar(31)
prop2	varchar(31)
key	varchar(31)
val	text
xval	text
order	int(11)

Рисунок 2.4 - Таблица Z_PROPS

Z_INDEX - каждое обращение к скриптам фиксируется адреса, порты, время обработки различных секций, по ней можно вести статистику. Таблица представлена на рисунке 2.5.

z_index

Field	Type
id	int(11)
session	varchar(31)
bits	int(11)
user	varchar(31)
time	timestamp
time_set	text
time_total	int(11)
node	varchar(31)
desc	text
param	text
came	text
agent	text
ip	varchar(31)
host	text
port	int(11)

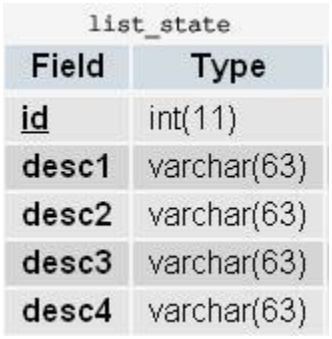
Рисунок 2.5 - Таблица Z_INDEX

U_COMMENT - таблица комментариев. Таблица представлена на рисунке 2.6.

Field	Type
id	int(11)
host	varchar(32)
user	varchar(32)
name	varchar(63)
email	varchar(63)
url	varchar(63)
city	varchar(63)
mess	text
time	int(11)

Рисунок 2.6 - Таблица U_COMMENT

LIST_STATE - таблица Стран, представлена на рисунке 2.7.

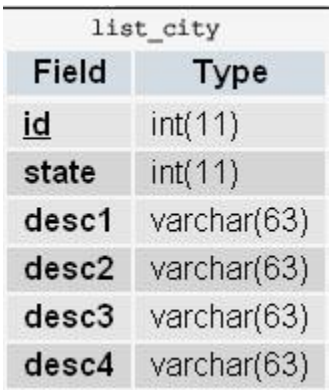


list_state

Field	Type
id	int(11)
desc1	varchar(63)
desc2	varchar(63)
desc3	varchar(63)
desc4	varchar(63)

Рисунок 2.7 - Таблица LIST_STATE

LIST_CITY - таблица городов, представлена на рисунке 2.8.

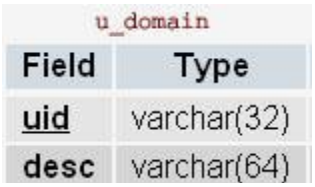


list_city

Field	Type
id	int(11)
state	int(11)
desc1	varchar(63)
desc2	varchar(63)
desc3	varchar(63)
desc4	varchar(63)

Рисунок 2.8 - Таблица LIST_CITY

U_DOMAIN - таблица тегов title и meta (keywords и description), что благотворно сказывается на результатах продвижения сайтов методами поисковой оптимизации. Таблица представлена на рисунке 2.9.



u_domain

Field	Type
uid	varchar(32)
desc	varchar(64)

Рисунок 2.9 - Таблица U_DOMAIN

Команда SQL CREATE TABLE предназначена для описания структуры таблицы. Команда SQL CREATE TABLE создает пустую таблицу (без строк).

При создании, с помощью CREATE TABLE, любому столбцу с ограничением NOT NULL должно быть установлено значение в каждом предложении INSERT.

Чтобы предохранить поле от разрешения в нем пустых (NULL) указателей для столбца задается ограничение NOT NULL.

Скрипт создания базы данных приведен в приложении А.

Разработка дополнительного функционала для CMS RIX

Базовым языком программирования будет PHP. Файлы с PHP-кодом будут располагаться в корневом каталоге сайта (это касается базовых файлов) и в каталоге /route (там будут храниться подключаемые модули).

Для хранения информации будем использовать базу данных MySQL. Она обладает собственной структурой, поэтому для еѐ использования не надо создавать никаких каталогов - она хранит свои файлы в своѐм собственном каталоге.

Для отображения на экране будет использоваться язык разметки гипертекста HTML в связке с таблицами стилей CSS, клиентским языком программирования JavaScript и растровыми изображениями GIF/JPEG/PNG. Для файлов HTML создадим каталог html, для таблиц стилей каталог css, для клиентских скриптов каталог js, а для картинок - каталог images.

Для подключения к базе данных MySQL требуется указывать адрес сервера, имя пользователя, пароль и имя базы данных. Создадим его в корневом каталоге сайта config.php.

```
<?php $DB_host = "localhost"; // имя сервера MySQL $DB_user = "root"; // имя пользователя MySQL $DB_pass = ""; // пароль на сервере MySQL $DB_name = "cms"; // имя базы данных ?>
```

Включим этот файл в любой скрипт в любое время и иметь доступными настройки для подключения к базе данных.

Виртуальный магазин это прайс лист некой организации с наименованием товаров, часть из которых можно пометить и занести в корзину, чтобы в дальнейшем можно было заполнить анкету (ФИО, адрес и т.д.) для покупки этих товаров.

Символы "<?" должны быть первыми в файле. Т.е. ни пустых строк, ни пробелов до символов "<?".

```
unset($t);
session_start(); 37
```

Функция прибавляет в корзину новый товар, где $n - это номер строки в shop.txt. Далее, в сессиях сохраняется не номер строки, а число ID из shop.txt и используется повсеместно. Если товар уже существует, то корзина не меняется.

```
function tadd($n) {
global $t;
```

Открывает файл.

```
$f=file("./shop.txt") or die("файл не найден");
```

Получает нужную строку с товаром (в массив $o).

```
$o=explode("\\",$f[$n]);
```

```
$id=$o[0];
if (isset($t[all][$id])) return;
```

Если товар уже в корзине - выход.

```
$t[all][$id]=$id;
```

Определяет, что товар уже есть в корзине.

```
$t[$id][name]=$o[1]; // наименование
$t[$id][info]=$o[2]; // инфо
$t[$id][cena]=$o[3]; // и д.р. не значащие данные
$t[$id][kol]=1; // кол-во в начале равно "1 штуке"
session_register("t");
```

Функция рисует таблицу с товарами в корзине. Из файла shop.txt читает только названия колонок. Названия товара (в данном случае только название /цена/кол-во) берется из сессии.

```
function korzina() {
global $t,$PHP_SELF,$SID;
$f=file("./shop.txt") or die("файл не найден");
$ogl=explode("\\",$f[0]); 38
echo "<form action=$PHP_SELF method=POST>".
"<input type=hidden name=SID value='$SID'>".
```

Передает сессию.

```
"<input type=hidden name=c value=kolvo>".
```

Рисует заголовок таблицы с корзиной.

```
"<table border=2><tR><td>$ogl[1]</td></td><td>$ogl[3]</td>".
"<td>кол-во</td><td>команды</td></tr>";
```

Проходит массив `$t[all]` по списку его ключей.

```
$k=@array_keys($t[all]);
for ($i=0; $i<count($k); $i++) {
$id=$k[$i];
echo "<tr><td>{$t[$id][name]}</td>".
"<td>{$t[$id][cena]}</td>".
"<td><input size=4 type=text name=v[$id]
value={$t[$id][kol]}></td>".
"<td><a
href=$PHP_SELF?c=del&id=$id&SID=$SID>удалить</a></td></tr>";
}
```

Внизу таблицы две кнопки.

```
     echo  "</table><input  type=submit  name=edit  value='Внести
изменения'>    ".
     "<input      type=submit      name=zakaz      value='Оформить
заказ'></form>";
     }
```

Выводит на экран таблицу с товарами. В таблице генерируется картинка в виде корзины, отметив которые и нажав "добавив", можно занести товары в корзину.

```
     function price() {
     global $t, $PHP_SELF,$SID;
     $f=file("./shop.txt") or die("файл не найден");
     $ogl=explode("\\",$f[0]);
     $x=count($ogl);
     $y=count($f);
     echo  "<form  action=$PHP_SELF  method=POST><input  type=hidden
name=c value=add>". 39
     "<input type=hidden name=SID value='$SID'><table border=2>";
```

Рисует заголовок таблицы, названия колонок - первая строка файла shop.txt.

```
     echo "<tr>";
     for ($j=0; $j<$x; $j++) {
     if (strlen($ogl[$j])==0) echo "<tD> </td>";
     else echo "<td>$ogl[$j]</td>";
     }
     echo "<td>x</td></tr>";
     Цикл вывода прайса:
     for ($i=1; $i<$y; $i++) {
     $a=explode("\\",$f[$i]); // читает строку файла
     if (count($a)<2) continue; // если она пустая , пропускает
     echo "<tr>";
     Цикл вывода всех колонок текущей строки таблицы
     for ($j=0; $j<$x; $j++) {
     // если ячейка пустая, надо поместить " ";
     if (strlen($a[$j])==0) echo "<tD> </td>";
     else echo "<td>$a[$j]</td>";
     }
```

Выводит код в последней колонке текущей строки.

```
     echo        "<td><input        type=checkbox        name=v[$i]
value=$i></td></tr>";
     }
     echo  "</table><br><center><input  type=submit  value='Добавить
".
     "отмеченные товары в корзину'></center></form>";
     }
```

Выводит на экран несколько чисел. Подсчет значений происходит при каждом вызове.

```
function summa() {
global $t;
$k=@array_keys($t[all]);
for ($i=0; $i<count($k); $i++) {
$id=$k[$i];
$summ+=(double)$t[$id][kol]*(double)$t[$id][cena]; 40
$summ2+=$t[$id][kol];
}
```

Выводит посчитанные цифры на экран.

```
echo "Корзина: наименований товаров - $i (в кол-ве $summ2 шт), цена - ".sprintf("%.2f тенге.<br>",$summ);
}
```

Объявление переменной post, которая содержит поля для заполнения посетителем при оформление заказа т.к. этот список используется 2 раза, то описано ниже.

```
$post=array(
"название организации",
"Ф.И.О. должностного лица",
"должность",
"ИНН организации",
"местонахождение организации",
"контактный телефон",
"e-mail");
```

Основной код программы.
$c - основная переменная, указывающая на нужное действие.

```
if (!isset($c)) $c='';

switch($c) {
case "":
```

Без параметров - рисует прайс-лист.

`summa();` // статистика по корзине.
`price();` // прайс.

Ссылка для перехода на корзину.

```
echo "<li><a href='$PHP_SELF?c=korzina&SID=$SID'>Корзина покупок</a>";
break;
case "korzina": 41
```

Вывод корзины.

```
summa();
korzina();
```

Пишется две ссылки для «Каталог товаров» «Очистить корзину».

```
echo "<li><a href='$PHP_SELF?SID=$SID'>Каталог товаров</a>";
echo    "<li><a    href='$PHP_SELF?c=delete&SID=$SID'>Очистить
корзину (осторожно!)</a>";
break;
case "add":
```

Добавление из формы прайса всех товаров. В массиве $v скоплены номера строк товаров.

```
$k=@array_keys($v);
for ($i=0; $i<count($k); $i++) {
```

`tadd()` преобразует из файла в данные и поместит в сессии.

```
tadd($v[$k[$i]]);
}
case "kolvo"
```

Изменить количество товара товаров.Изменение или оформление заказа. Оцените, насколько короткий код преобразования корзины.

```
$k=@array_keys($v);
for ($i=0; $i<count($k); $i++) {
$t[$k[$i]][kol]=abs(intval($v[$k[$i]]));
}
```

После изменения переменной сессии нужно записать.

```
session_register("t");
```

Далее важная проверка. Если посетитель нажимает кнопку сохранить то устанавливается переменная $edit, которая содержит строку.Сохранить изменения. Если нажимает заказ, то устанавливается $post. Устанавливается только одна из этих двух переменных.

2.7 Создание приложения

Этапы создания приложения производились по принципу «от общего к частному». Велась тщательная работа по оптимизации исходного кода. Использовались такие инструменты как редактор ActiveState Komodo Edit 6.0.0, движок баз данных MySQL, скриптовый язык программирования общего назначения PHP . В качестве веб - сервера использовался Денвер - набор

дистрибутивов (Apache, PHP, MySQL, Perl и т.д.) и оболочка для разработки сайтов на «домашней» (локальной) Windows-машине без выхода в Интернет. В роли браузера использовался Mozilla Firefox 3.6.15. на рисунке 2.10.

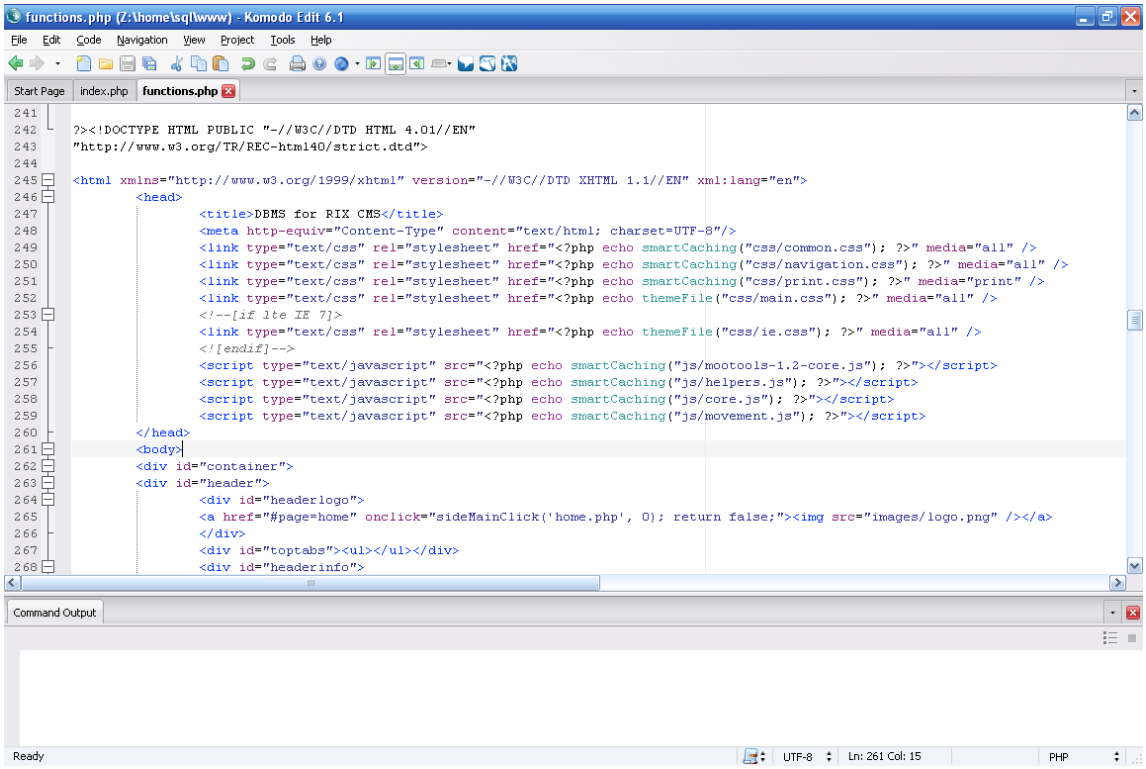


Рисунок 2.10 - Разработка веб - приложения

Исходный программный код в данном случае так же прилагается к приложению и не имеет смысла приводить его здесь, даже в приложении.

3 Описание работы системы управления базами данных

Для тестирования системы был выбран url http://sql/, настроенный на локальном веб-сервере. При первом обращении по этому адресу система автоматически перенаправляет нас по адресу http://sql/login.php , где мы должны пройти авторизацию на рисунке 3.1.

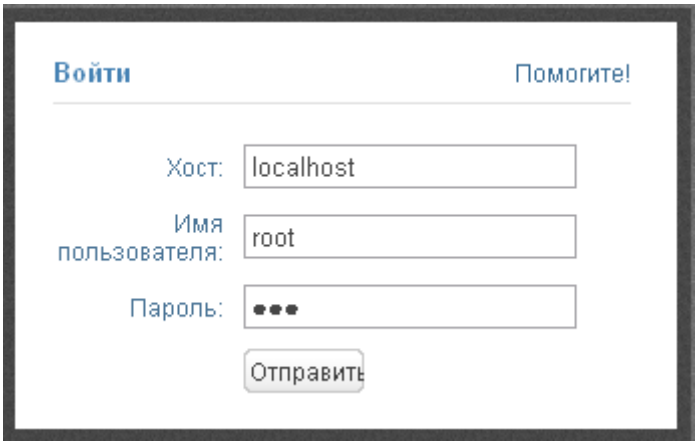


Рисунок 3.1 - Форма авторизации

В целом задача создания системы управления базами данных для CMS RIX выполнена успешно.

В случае неверного ввода имени пользователя или пароля для доступа к базе данных выходит сообщение об ошибке на рисунке 3.2.

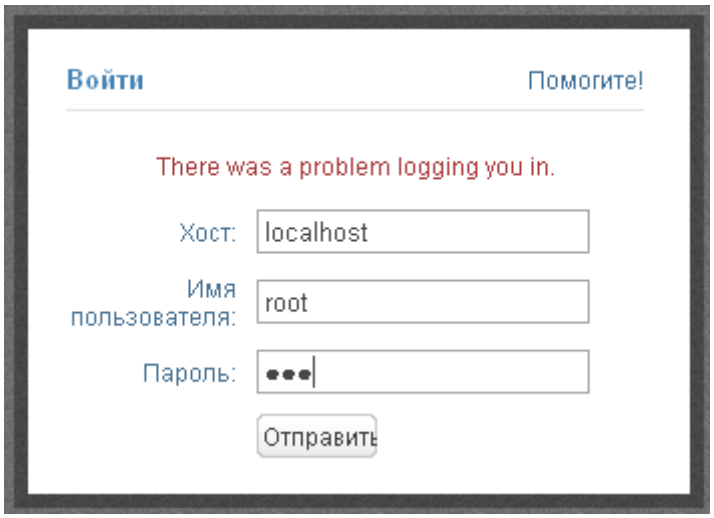


Рисунок 3.2 - Ошибка авторизации

В случае же успешной авторизации мы попадает в главное меню системы на рисунке 3.3.

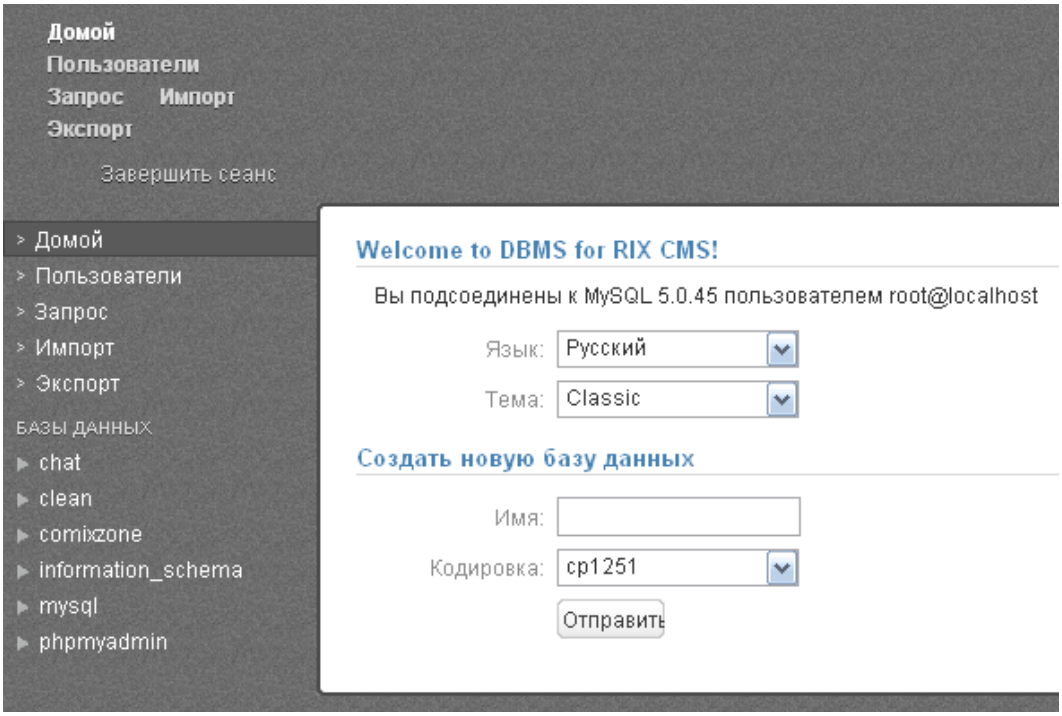


Рисунок 3.3 - Главное меню системы

В главном меню видно 4 пункта: пункт 1 -Домой -чтобы вернуться на главную страницу системы, пункт 2-Пользователи-для работы с пользователями. Поддерживаются операции отображения, редактирования, удаления и создания новых пользователей на рисунке 3.4.

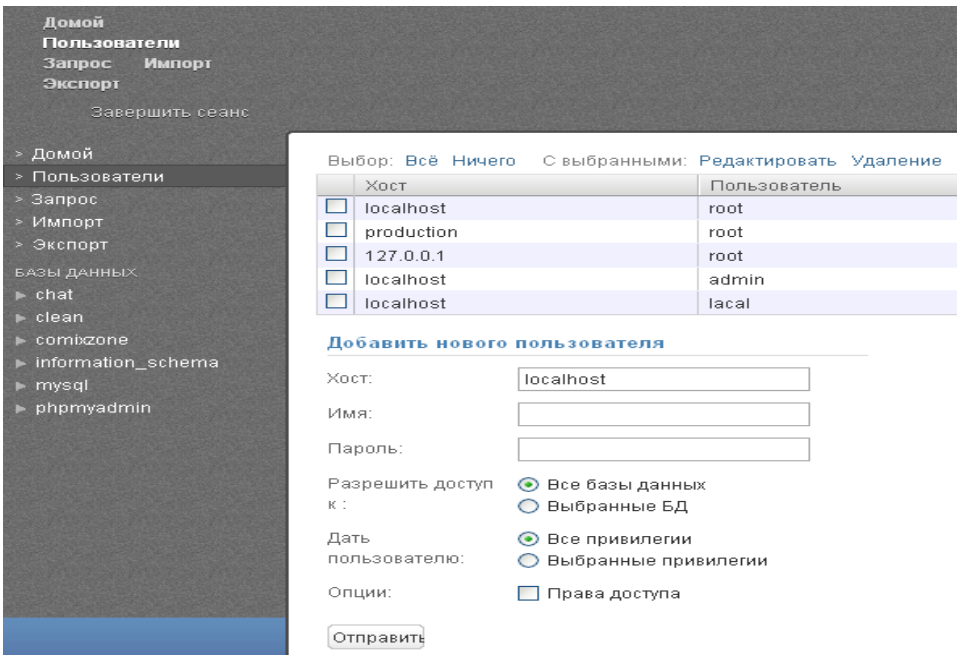


Рисунок 3.4 - Управление пользователями

Пункт 3 -Запрос-позволяет выполнить произвольный запрос, который нужно ввести в поле ввода запроса и нажать на кнопку «Отправить» на рисунке 3.5.

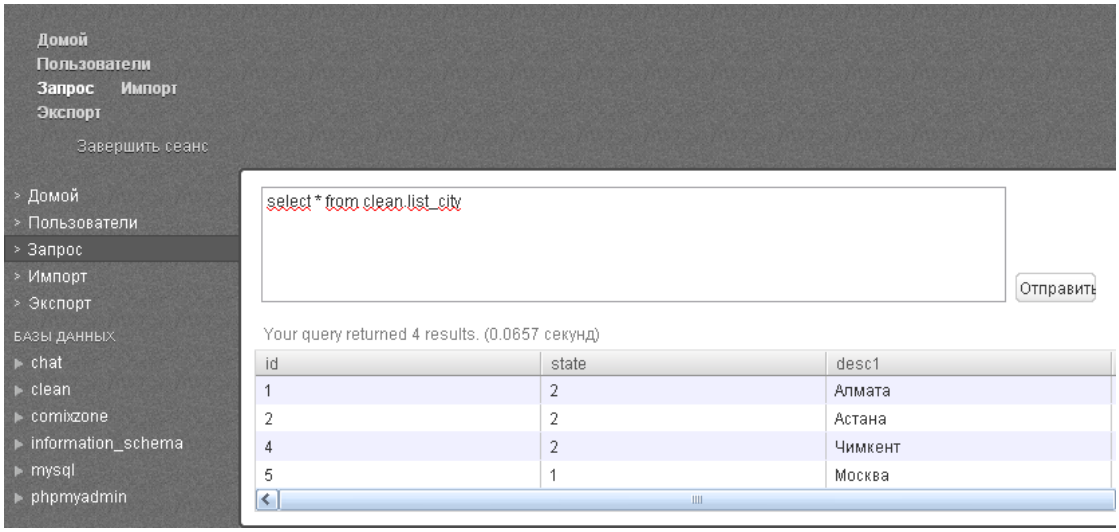


Рисунок 3.5 - Выполнение запросов

Пункт 4-Импорт позволяет импортировать базу данных из файла с расширением .sql, находящегося на локальной файловой системе. Для этого нужно нажать на кнопку обзор или кликнуть на поле ввода имени файла и автоматически откроется окно проводника Windows для указания пути к файлу. Далее следует нажать кнопку «Отправить» на рисунке 3.6.

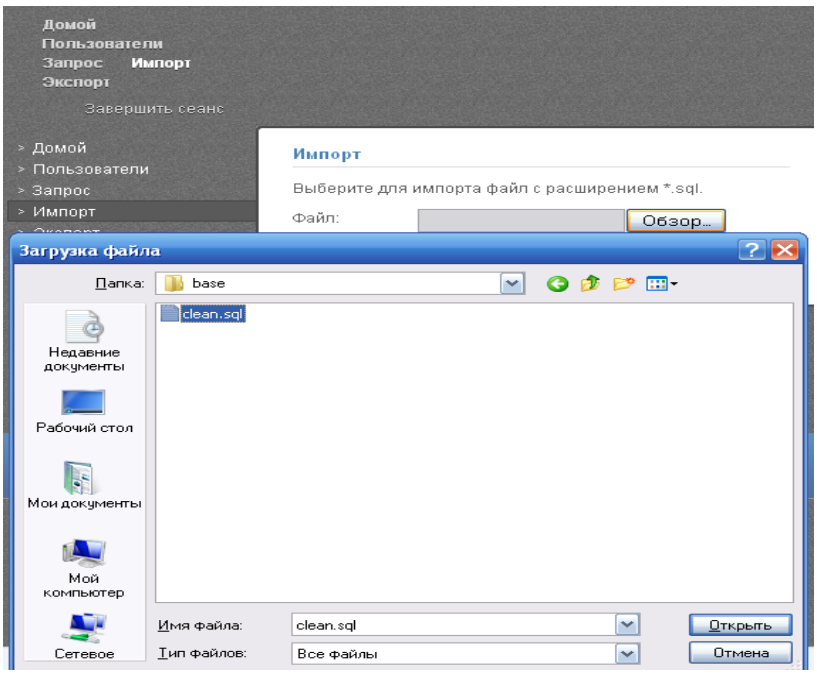


Рисунок 3.6 - Импорт из файла

Пункт 5-Экспорт позволяет экспортировать любую базу данных. Для этого нужно выбрать необходимую базу данных из списка и нажать на кнопку «Отправить» на рисунке 3.7.

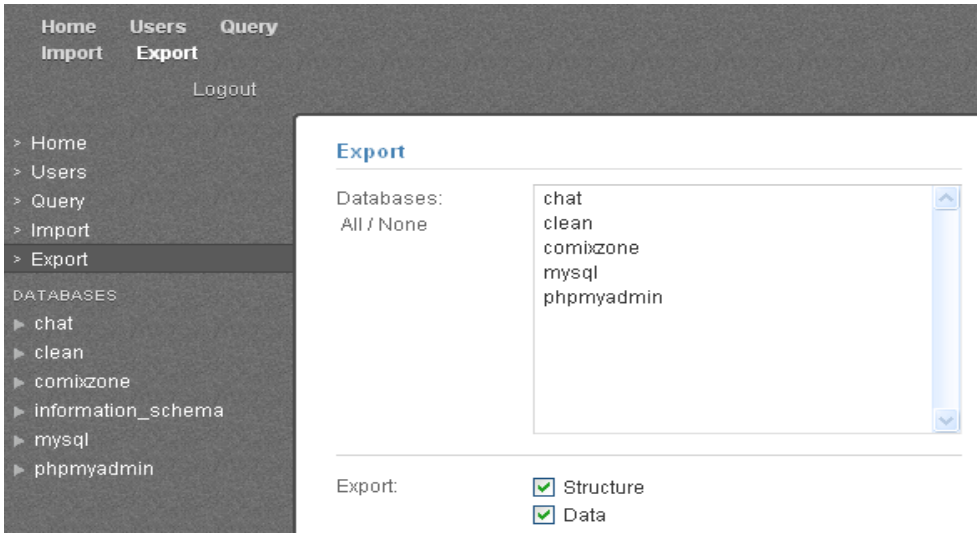


Рисунок 3.7 - Экспорт базы данных

В меню Базы данных можно выбрать необходимую базу данных и будет открыт список таблиц с возможностью их множественного выбора и проведением операций очистки, удаления или оптимизации на рисунке 3.8.

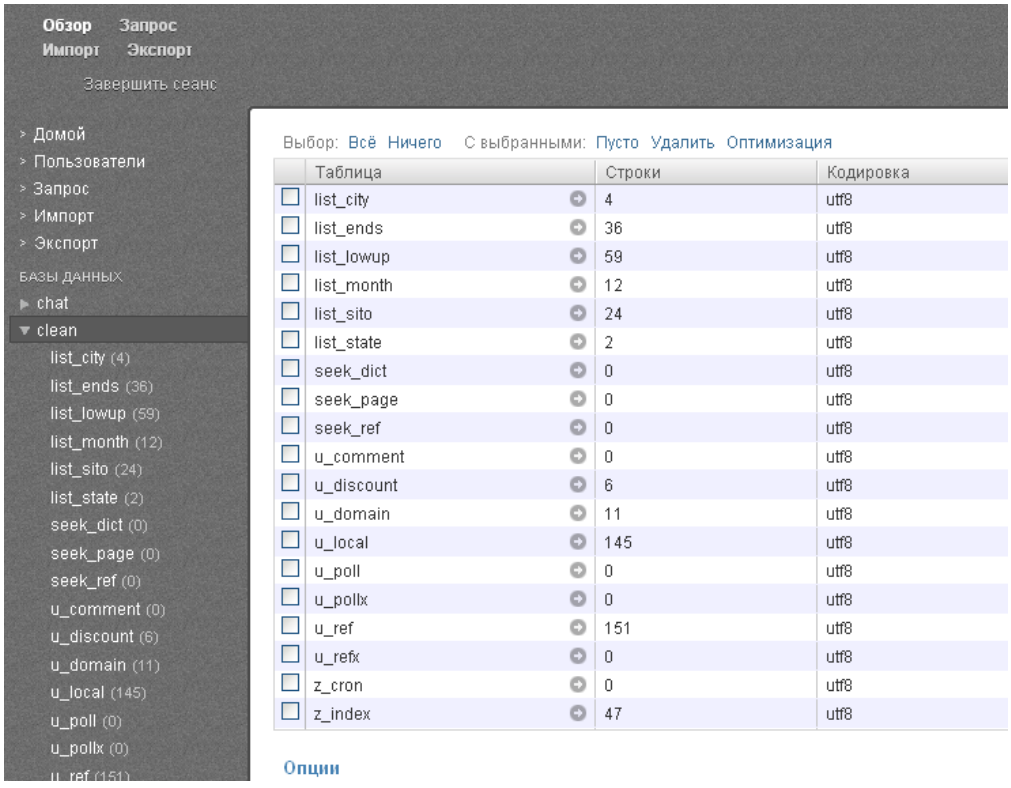


Рисунок 3.8 - Выбор базы данных и манипулирование

Также после выбора базы данных список входящих в ней таблиц будет отображаться под ней. Можно выбрать любую таблицу и выполнять операции редактирования с ее строками на рисунке 3.9.

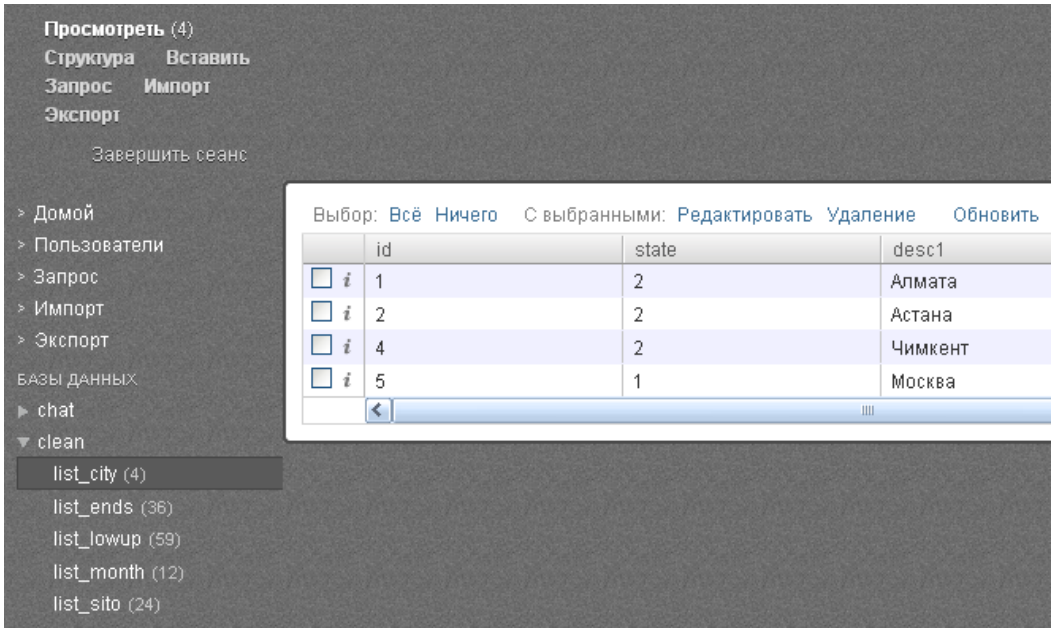


Рисунок 3.9 - Редактирование таблиц

4 Безопасность жизнедеятельности

4.1 Анализ потенциально опасных и вредных факторов

В выпускной работе рассматривается проектирование системы управления базами данных, а конкретно, системы управления базами данных для системы управления содержимым сайта RIX. Для решения поставленной задачи, используем только современную технику, в том числе персональные компьютеры. В связи с этим рассмотрим вопросы создания оптимальных условий труда для работающего персонала.

В настоящее время компьютерная техника широко применяется во всех областях деятельности человека. При работе с компьютером человек подвергается воздействию ряда опасных и вредных производственных факторов: электромагнитных полей, инфракрасного и ионизирующего излучений, статического электричества и др.

Для обеспечения одного программиста администратора и двух операторов рабочими местами в компьютерном кабинете отдела разработки компании «WebMaker», схема которого указана на рисунке 5.1. разместим три рабочих места, оснащенных ПК. Размеры помещения составляют: длина $a = 12$ м, ширина $b = 6$ м, высота $h = 4$ м. Общая площадь равна $a \times b = 72$ кв.м, а общий объем составит 288 куб.м. В помещении имеется 2 окна размерами 4,5 м х 2,5 м, выходящие на запад.

К организации рабочих мест предъявляем особые требования [3]:

- расстояние между рабочими местами с боковых сторон не менее 1,2 м, при расположении пользователей последовательно не менее 2-х метров, лицом друг к другу - 0,6м;
- размещать ПК экраном к окну не допускается, так как это создаёт тень и блики на экране монитора;
- размещать ПК вблизи от стены не рекомендуется, так как пользователь не сможет проделывать зрительные упражнения. Также стена отражает электромагнитное излучение от ПК на пользователя;
- рабочее место, оснащенное ПК, размещаем так, чтобы пользователь не находился лицом к окну, что создаёт неудобство и дополнительную нагрузку на глаза;
- пользователи будут мешать друг другу, если они будут находиться спиной к спине;
- пользователей располагаем, так чтобы естественное освещение находилось с левой стороны, также допускается и с правой.

В соответствии с перечисленными требованиями спроектируем помещение отдела информационных технологий на рисунке 4.1.

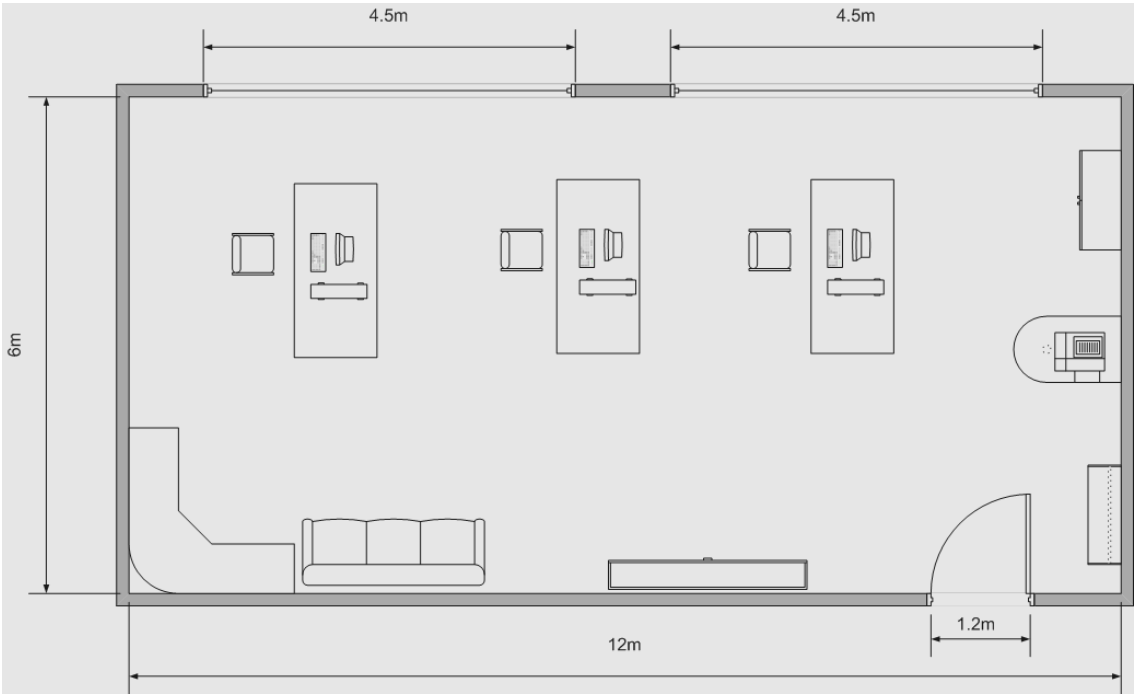


Рисунок 4.1 - Компьютерный кабинет отдела разработки

В помещении используем 3 персональных компьютера.

Технические характеристики устройств:

- персональные компьютеры AMDx64 3800+ MHz/Gigabyte S955 MX (SIS751+SB+VGA)/2048MB DDR/500 Gb SATA Seagate/FDD/SP/CD;
- мониторы 17" 0.26dpi; Мощность экспозиционной дозы рентгеновского излучения от мониторов "LG LI720" в любой точке на расстоянии 0.05 м от экрана и корпуса мониторов не превышает 0,1 мбэр/час;
- габариты: 1600x800x1100 (персональный компьютер + стол);
- электропитание: переменное напряжение 220-250В, частотой 50 Гц. Мощность 400 Вт;
- сервер - коммутатор Quidway S5600-26C-OVS - в количестве 1 штука;
- многофункциональное устройство Hewlett Packard LaserJet M1522.

Электромагнитные излучения

Повышенный уровень электромагнитных излучений отсутствует (применен монитор типа LCD). Мониторы, используемые, для работы операторов соответствуют самым современным требованиям и стандартам. Дисплей обладает высокой разрешающей способностью (размер точки 0,22 мм). Дисплей имеет экранный фильтр с антистатическим покрытием, который в несколько раз снижает утомляемость глаз и концентрацию пылевых частиц вблизи экрана монитора. Применяемый монитор имеет сертификат TCO' 03, который регламентирует электромагнитные излучения, акустический шум, электробезопасность для мониторов. Уровень ультрафиолетового излучения на рабочем месте оператора в области длинных волн (400-315 нм) не превышает 10 Вт/м^2, в области средних волн (315-280 нм) не более 0,01 Вт/м^2 и отсутствует в области коротких волн (280-200 нм) [2]. Напряженность электромагнитного

поля на рабочем месте оператора по электрической составляющей не более 50В/м, а по магнитной составляющей не более 5А/м в таблице 4.1. и 4.2.

Т а б л и ц а 4.1 - Максимальные значения переменного электрического поля ТСО'03 [2]

Частота	Максимальный уровень переменного электрического поля
5 Гц - 2 КГц	10 В/М на расстоянии от 30-50 см от дисплея
2 КГц-400 КГц	1 В/М на расстоянии 50 см вокруг монитора и 30 см спереди от монитора

Т а б л и ц а 4.2 - Максимальные значения переменного магнитного поля ТСО'03 [2]

Частота	Максимальный уровень переменного магнитного поля
5 Гц - 2 КГц	200 нТ на расстоянии 50 см вокруг монитора и 30 см. спереди от монитора
2 КГц-400 КГц	25 нТ на расстоянии 50 см вокруг монитора

Таким образом, можно считать, что электромагнитные излучения монитора соответствует нормам.

Микроклимат

На работоспособность работника влияет в большой степени микроклимат, то есть температура, скорость движения воздуха, влажность и т.д. [3].

Температура воздуха оказывает существенное влияние на самочувствие и результаты труда человека. Низкая температура вызывает охлаждение организма и может способствовать возникновению простудных заболеваний. При высокой температуре возникает перегрев организма, что ведет к повышенному потовыделению и снижению работоспособности. Работник теряет внимание, что может стать причиной несчастного случая. Поэтому в помещении в соответствии с требованиями ГОСТ 12.1.005-88 "Воздух рабочей зоны", поддерживается постоянная температура воздуха 24^0С, скорость движения воздуха 0,1м/с, а относительная влажность около 60%. Для соблюдения данных параметров в помещении организованна естественная вентиляция и размещаем кондиционирующую установку типа "Зима-лето" [4].

4.2 Расчеты

4.2.1 Расчет искусственного освещения

Расчет произведем по методике, изложенной в [8].

В помещении размерами 12 м х 6 м х 4 м имеется 4 счетверенных светильника: ПВЛМ-1х40, каждый из которых обеспечивает световой поток $\varphi_л$ = 12480 лм (с учетом, что каждая лампа светильника 3120 лм) показано на рисунке 4.2.

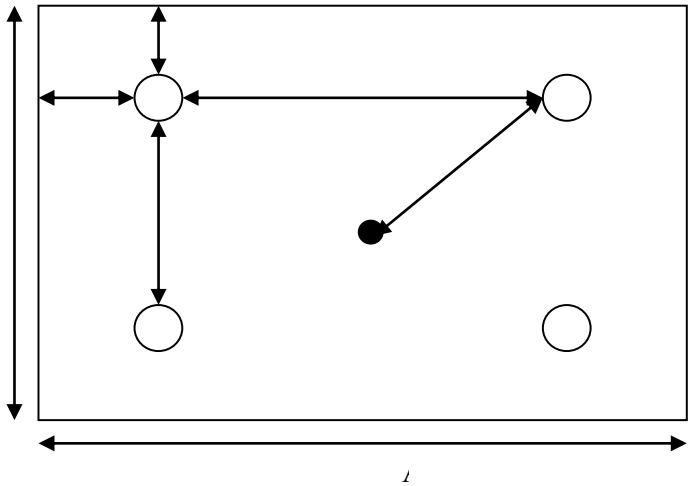


Рисунок 4.2 - Схема расположения светильников

Освещенность, необходимая для нормального выполнения работ (разряд зрительной работы высокой точности) согласно [6] (300лк.) показана в таблице 4.3.

Таблица 4.3 - Нормированные значения освещенности для искусственного освещения.

Характеристика зрительной работы	Наименьший эквивалентный размер объекта различения, мм	Разряд зрительной работы	Подразряд зрительной работы	Освещенность на рабочей поверхности от системы общего освещения, лк
Очень высокой точности	От 0,15 до 0,30	А	1 2	500 400
Высокой точности	От 0,30 до 0,50	Б	1 2	300 200

Выясним, обеспечивает ли существующая система искусственного освещения требуемую освещенность. Вычислим $E_г$ и сравним с $E_н$ = 300 лк. Расчет проведем с помощью нахождения средней освещенности на единицу площади.

Для создания требуемого уровня освещенности используются счетверенные светильники, содержащие по четыре лампы ПВЛМ-1х40 мощностью по 26 Вт, со светоотдачей 120 лм/Вт.

Световой поток одной лампы

$$\varphi = 26 \cdot 120 = 3120 \text{ лм}$$

Общий световой поток

$$\varphi_o = 3120 \cdot 16 = 44920 \text{ лм}$$

Определим среднюю освещенность на единицу площади по формуле

$$E_T = \frac{\varphi_o}{S \cdot K_з} \tag{4.5}$$

где φ_o - общий световой поток от всех ламп;

S - освещаемая площадь;

$K_з$ - коэффициент, учитывающий стекло светильника.

Рассчитаем схему расстановки светильников по формулам

$$L_{A,B} = \lambda * h_{расч} \tag{4.6}$$

где $\lambda = 1{,}2 \div 2$ - коэффициент экономичности расположения светильников;

$l_{A,B} = (0{,}4 \div 0{,}5) * L_{A,B}$ - при наличии прохода вдоль стен;

$l_{A,B} = 0{,}3 * L_{A,B}$ - при отсутствии прохода вдоль стен.

Для более простого решения, составим систему уравнений исходя из заданных параметров помещения

$$\begin{aligned} L_B + 2 \cdot l_B &= 12, \\ L_B &= \lambda \cdot 3, \\ l_B &= 0{,}5 \cdot L_B \end{aligned} \tag{4.7}$$

Решаем систему уравнений, находим значения

$L_A = 6$ м; $l_A = 3$ м; $\lambda = 2$; (входит в заданный промежуток $\lambda = 1{,}2 \div 2$)

Для второй стены система примет вид формул

$$\begin{aligned} L_B + 2 \cdot l_B &= 6, \\ L_B &= \lambda \cdot 3, \\ l_B &= 0{,}3 \cdot L_B \end{aligned} \tag{4.8}$$

$L_B = 3$ м; $l_B = 1{,}5$ м; $\lambda = 1, 25$; (входит в заданный промежуток $\lambda = 1{,}2 \div 2$).

Схема расстановки соответствует действующему расположению светильников.

Был проведен расчет искусственного освещения. Расчет освещенности точечным методом показал, что заданного числа светильников достаточно для обеспечения требуемой освещенности помещения, так как расчетная $E_г = 520$ лк превосходит нормативную освещенность $E_Н = 300$ лк.

4.2.2 Расчет системы вентиляции

Теплопоступления и теплопотери в результате разности температур

Расчет произведем по методике, изложенной в [4]. Теплоотдача для теплого периода принимается равной нулю.

Для холодного периода расчитывается по формуле

$$Q_{огр} = V_{пом} X_о (t_{Врасч} - t_{Нрасч}) \quad (4.9)$$

где $V_{пом}$ - объем помещения, $V_{пом} = 11 \cdot 5 \cdot 3 = 165$ м^3;

$X_о$ - удельная тепловая характеристика, $X_о = 0{,}42$ Вт/м^3 ^{0}С;

$t_{Нрасч}$ - средняя температура наиболее холодной пятидневки в холодный период года;

$t_{Нрасч} = -25\ ^0$С;

$t_{Врасч}$ - температура в помещении, $t_{Врасч} = 18\ ^0$С.

$Q_{огр} = 165 \cdot 0{,}42 \cdot (18-(-25)) = 2\,979{,}9$ Вт.

Теплопоступления от солнечного излучения через остекление

При отсутствии наружных затеняющих козырьков, ребер и т.д. для периода облучения остекления солнцем, когда его лучи проникают через окно в помещение:

- Для периода прямого облучения солнечным светом

$$Q_р = q^1 F_0 \beta_{с.з.} = (q_{вп} + q_{вр}) К_1 К_2\ \beta_{с.з.} n H_0 B_0 \quad (4.10)$$

- Для периода тени, когда лучи солнца не проникают через окна

$$Q_р = q^2 F_0 \beta_{с.з.} = q_{вр} К_1 К_2\ \beta_{с.з.} n H_0 B_0 \quad (4.11)$$

где $q_{вп}$; $q_{вр}$ - тепловые потоки от прямой и рассеянной радиации.

Для окон на северной стороне

$q_{вп} = 0$,

$q_{вр} = 60$

Для окон на южной стороне

$q_{вп}$ = 245,

$q_{вр}$ = 84

$F_0 = nH_0B_0$ - площадь светового проема (n - число окон, H_0 и B_0 - высота и ширина окна);

$\beta_{с.з.}$- коэффициент тепло пропускания солнцезащитных устройств. $\beta_{с.з.}$ = 0,15;

$К_1$- коэффициент затемнения остекления переплетами;

$К_1$ = 0,65;

$К_2$ - коэффициент загрязнения остекления;

$К_2$ = 0,9.

Для периода прямого облучения солнечным светом окон на северной стороне

$Q_р = q^1F_0\beta_{с.з.}$ = (0 + 60)*0,65*0,9*0,15*4*1,5 = 31,59 Вт.

Для периода тени

$Q_р = q^2F_0\beta_{с.з.}$ = 60*0,65*0,9*0,15*4*1,5 = 31,59 Вт.

Для периода прямого облучения солнечным светом окон на южной стороне

$Q_р = q^1F_0\beta_{с.з.}$ = (245 + 84)* 0,65*0,9*0,15*4*1,5= 173,2185 Вт.

Для периода тени

$Q_р = q^2F_0\beta_{с.з.}$ =84*0,65*0,9*0,15*4*1,5 = 44,226 Вт.

Выбираем максимальное значение

$Q_р$ = 173,2185 Вт.

Теплопоступления от людей

Поступление тепла от людей зависит от интенсивности выполняемой работы и параметров окружающего воздуха. Тепло, выделяемое человеком, складывается из ощутимого, т.е. передаваемого в воздух помещения путем конвекции и лучеиспускания, и скрытого тепла, затрачиваемого на испарение влаги с поверхности кожи и из легких.

Для расчета необходимо количество человек умножить на показатель для одного человека. Таким образом, мы найдем общее тепло, выделяемое всеми

людьми, находящимися в помещении. Беру тепловыделение для 18^0С, так как оно максимально, если сравнивать 18 и 24^0С.

$Q_Л^o$ = 3*133 = 931 Вт

Общее тепловыделение от всех людей равно

$Q_л^o = Q_м^o + Q_ж^o$ = 1 496,25 Вт

Теплопоступления от осветительных приборов

Теплопоступления от ламп рассчитывается по формуле

$$Q_{осв} = \eta N_{осв} \text{ Вт} \quad (4.11)$$

где η - коэффициент перехода электрической энергии в тепловую (0,92-0,97 для ламп накаливания);

$N_{осв}$ - установленная мощность ламп.

Тепло выделяемое производственным оборудованием определяется по формуле

$$Q_{об} = N_{уст} K \quad (4.12)$$

где $N_{уст}$ - установленная мощность единицы оборудования;

К - коэффициент установочной мощности.

Для нахождения тепла, выделяемого осветительными приборами, необходимо учесть такой параметр как площадь пола. Очевидно, что для нахождения общего количества тепла, выделяемого оборудованием необходимо тепло выделяемое одной единицей оборудования умножить на количество оборудования соответственно.

Таким образом, получаем:

- Общее количество тепла, выделяемое осветительными приборами

$$Q_{осв} = \eta N_{осв} F_п \text{ Вт} \quad (4.13)$$

- А количество тепла, выделяемое оборудованием

$$Q_{об} = N_{уст} K n_{об} \quad (4.14)$$

где $n_{об}$ - количество оборудования.

По формуле (4.13) найдем общее количество тепла, выделяемое осветительными приборами

$F_п = 11 \cdot 5 = 55$ м^2,

$Q_{осв}$ = 0,92*60*55 = 3 036 Вт

По формуле (4.14) найдем общее количество тепла, выделяемое оборудованием

$Q_{об}$ = 1,43*1000*0,85*8 = 9 724 Вт

Расчет общего избыточного тепла и выбор соответствующей модели кондиционера

$$Q_{изб} = Q_р + Q_л^{о} + Q_{осв} + Q_{об} - Q_{огр} \quad (4.15)$$

Для теплого периода

$Q_{изб}$ =173, 2185 + 1 496,25 + 3 036 + 9 724 - 0 = 14 429,4685 Вт

Для холодного периода

$Q_{изб}$ =0 + 399 + 3 036 + 9 724 - 2 979,9 = 11 276,35Вт

Из полученных данных видно, что необходимо построить систему кондиционирования с суммарной мощностью примерно в 15 кВт (с запасом).

Определим количество воздуха, необходимого для подачи в помещение

$$L_b = \frac{Q_{изб}}{c \cdot \Delta t \cdot \gamma_b} \quad (4.16)$$

где $Q_{изб}$ - избыточное тепло, передаваемое в помещение различными источниками;

С - теплоемкость воздуха, С = 0,24 ккал/кг^{0}С;

γ - удельная масса приточного воздуха, γ=1,206 кг/м^3;

Δt - разность температур воздуха выходящего из помещения и поступающего в помещение. Выбирается в зависимости от теплонапряженности воздуха $Q_н$.

$$Q_H = \frac{860 \cdot Q_{изб}}{V_n} \quad (4.17)$$

где V_i - объем помещения, $V_н = 165$ м^3;

860 - тепловой эквивалент 1кВт/ч.

Так как $Q_{изб}$ для летнего периода больше, чем для зимнего, то в расчет теплонапряженности будем учитывать $Q_{изб\ летн}$

$$Q_H = \frac{860 \cdot 15}{165} = 78{,}2\ \text{ккал/м}^3$$

С условием что

$Q_н \leq 20$ ккал/м^3 $\rightarrow \Delta t = 6\ ^0$C и $Q_н \succ 20$ ккал/м^3 $\rightarrow$ $\Delta t = 8\ ^0$C

$\Delta t = 8^0$C

Расчет количества воздуха

$$L_b = \frac{860 \cdot 15}{0{,}24 \cdot 8 \cdot 1{,}206} = 4961\,\text{м}^3/\text{ч}$$

Выбираем кондиционер с нижней подачей SDA модель 0601 с максимальным расходом воздуха 4970 м^3/ч.

4.3 Вывод по четвертой главе

Были рассчитаны внутренние и наружные тепловые нагрузки в помещении, количество воздуха необходимое для подачи в помещение. По нему была подобрана соответствующая модель кондиционера и приведены его основные характеристики. В результате расчетов был подобран кондиционер с верхней подачей SUA модель 0151 с максимальным расходом воздуха 1580 м^3/ч.

5 Экономический расчет

5.1 Технико-экономическое обоснование эффективности разработки

На сегодняшний момент управление и быстрый доступ к информации является одним из основополагающих факторов здорового функционирования предприятия. В частном случае, на примере системы управления содержимым сайта. В соответствии с этой проблемой была поставлена задача.

Решаемая задача подразумевает разработку системы управления базами данных, которая в свою очередь уменьшит сложность работы с базами данных и ускорит время обработки информации. Данная задача предполагает использование современных технологий построения баз данных и программного обеспечения с учетом их функционирования в глобальной сети.

Разрабатываемая система управления базами данных ориентирована на международный рынок и соответствует поставленным задачам. Для управления. Поэтому так остро стоит вопрос о создании качественной системы, способной обеспечить доступность информации об используемом оборудовании, сократить документооборот, обеспечить информацией о использовании оборудования и реализовать возможность журналирования и контроля рабочего времени.

Целью дипломной работы является создание системы управления базами данных, аналогичной существующим сейчас, но обладающей рядом особенностей. На сегодняшний день рынок предлагает множество готовых к использованию систем управления базами данных, но они, прежде всего, ориентированы на конкретные базы данных (MySQL, MS SQL, Oracle и т.д.). Задача же дипломной работы разработать систему управления базами данных, способную обеспечить управление базами данных различного типа, а так же взаимодействие с системой управления содержимым сайта RIX.

В целях создания более конкурентоспособного продукта, мной было принято решение о создании системы с помощью PHP - системы разработки сценариев, включающей:

- CGI-интерфейс;
- интерпретатор языка;
- набор функций для доступа к базам данных и различным объектам WWW.

Лидирующие позиции в этой области на данный момент занимают такие готовые системы управления базами данных как - phpMyAdmin, HeidiSQL, Navicat, dbForge Studio. Данные системы обладают рядом недостатков, малой гибкостью и ориентированностью преимущественно на коммерческие предприятия крупного бизнеса и крупные промышленные комплексы. Но для успешного функционирования системы управления содержимым сайта RIX на сегодняшний день не создано ни одной системы управления базами данных.

Разрабатываемый продукт может быть достаточно легко внедрен в существующие информационные системы предприятий, что также должно

сказаться на положительном принятии решения об использовании данного программного средства в своих информационных системах.

Этапы разработки программного продукта

Разработка, рассматриваемого в данной работе ПО состоит из следующих этапов:

- моделирование - составление модели программного обеспечения. Данный этап включает в себя разработку принципов работы программного обеспечения: определение алгоритмов, основных компонент и их назначения, проработку интерфейсов взаимодействия между компонентами системы, определение используемых технологий, оценку системных требований для программного обеспечения. Участники этапа: Руководитель проекта, два программиста. Длительность этапа: 4 рабочих дня;
- программирование. На данном этапе будет осуществляться описание, разработанной на первом этапе, модели с помощью алгоритмических языков программирования. Участники этапа: руководитель проекта, два программиста. Длительность этапа: 10 рабочих дней;
- тестирование ПО. На данном этапе будет производиться тестирование разработанного ПО, выявление ошибок реализации, будут рассмотрены принципиальные схемы внедрения ПО. Также на данном этапе будет производиться исправление найденных ошибок. Участники этапа: руководитель проекта, два программиста. Длительность этапа: 3 рабочих дня;
- внедрение ПО. В рамках данной дипломной работы будет производиться внедрение, разработанного программного обеспечения в систему системы управления содержимым сайта RIX. Участники этапа: руководитель проекта, один программист. Длительность этапа: 2 рабочих дня.

Сводные данные по плану проведения работ по разработке программного продукта представлены в Таблице 5.1.

Т а б л и ц а 5.1 - План проведения работ по разработке ПО

Наименование этапов и содержание работ	Исполнитель	Количество исполнителей	Длительность цикла
Моделирование	Руководитель проекта, 2 программиста	3	27.04.09 – 30.04.09 4 рабочих дня 32 нормо-часа
Программирование	Руководитель проекта, 2 программиста	3	01.05.09 – 14.06.09 10 рабочих дней 80 нормо-часов
Тестирование	Руководитель проекта, 2 программиста	3	15.05.09 - 19.05.09 3 рабочих дня 24 нормо-часа
Внедрение	Руководитель проекта, программист	2	20.05.09 - 22.05.09 2 рабочих дня 16 нормо-часов
Общая длительность:			19 дней

5.2 Расчет стоимости разработки ПО

Себестоимость ПО складывается из следующих статьей затрат:

- оплата труда;
- социальный налог;
- амортизационные отчисления;
- расходы на электроэнергию;
- расходные материалы;
- накладные расходы.

Таким образом, себестоимость разработки проекта определяется по следующей формуле

$$С = ФОТ + Ос + А + Э + Ра + Рм + Н \quad (5.2)$$

где ФОТ - фонд оплаты труда;

Ос - социальный налог;

А - амортизационные отчисления;

Э - расходы на электроэнергию;

Рм - расходы на материалы;

Ра - расходы на аренду;

Н - накладные расходы.

Расчет затрат по оплате труда производственного персонала

Для расчета затрат на заработную плату необходимы следующие данные:

- численность задействованного персонала;
- среднемесячная заработная плата каждого работника;
- длительность разработки проекта и каждого вида выполняемых работ;
- трудоемкость.

В процессе разработки данного программного обеспечения участвуют 3 человека:

- руководитель проекта;
- 2 программиста.

Месячная заработная плата сотрудников:

1. руководитель проекта - 100 000 тенге;
2. программист - 75 000 тенге.

Заработную плату за один час рассчитаем по формуле

$$D = \frac{ЗПм}{Др \cdot Чр} \quad (5.3)$$

где $ЗПм$ - ежемесячный размер заработной платы;

$Др$ - количество рабочих дней в месяце (22 рабочих дня);

$Чр$ - количество часов рабочего дня (при 8 часовом рабочем дне).

Заработная плата каждого работника за один час составляет:

1) Для руководителя проекта

$$D = \frac{100000}{22 \cdot 8} = 568 \text{ тенге/час}$$

2) Для программиста

$$D = \frac{75000}{22 \cdot 8} = 426 \text{ тенге/час}$$

На основе данных расчета по длительности каждого этапа разработки программного продукта, произведем расчет количества нормо-часов для каждого работника.

Количество нормо-часов рассчитывается по формуле

$$T = t_n \cdot z \tag{5.4}$$

где t_n- количество часов в рабочем дне;

z - рабочие дни.

1) Для руководителя проекта

$T_1 = 19 \cdot 8 = 152$ нормо-часа;

2) Для программиста

$T_2 = 19 \cdot 8 = 152$ нормо-часа;

3) Для второго программиста

$T_2 = 17 \cdot 8 = 136$ нормо-часов

Сводные данные расчета заработной платы производственного персонала представлены в таблице 5.2.

Таблица 5.2 - Расчет основной заработной платы производственного персонала

Наименование содержания работ	**Исполнитель**	**Трудоёмкость, норма-час**	**Зар. плата за час работы (тг/час)**	**Сумма заработной платы**
Моделирование	Руководитель проекта	32	568	18 750
	Программисты	32	426	13 630

Окончание таблицы 5.2

Наименование содержания работ	**Исполнитель**	**Трудоёмкость, норма-час**	**Зар. плата за час работы (тг/час)**	**Сумма заработной платы**
Программирование	Руководитель проекта	80	568	45 440
	Программисты	80	426	34 080
Тестирование	Руководитель проекта	24	568	13 630
	Программисты	24	426	10 225
Внедрение	Рук. проекта	16	568	9 090
	Программист	16	426	6 815
ИТОГ (ФОТ)				209 595

Социальный налог

Отчисления из фонда оплаты труда в социальный налог в 2011 году укрупненно 11% от ФОТ за вычетом пенсионных отчислений в размере 10%

Социальный налог рассчитывается по формуле

$$ОС = 0{,}11 \cdot 0.9 \cdot ФОТ \qquad (5.5)$$

где ФОТ - фонд оплаты труда;

Таким образом, отчисления на социальный налог составляют

$$O_c = 0{,}11 \cdot 209595 = 20749{,}91 \text{ тенге}$$

Расходы на материалы

В сумму расходов по данной статье учитывают «предметы труда» или производственные запасы, составляющие материальную основу общих затрат в процессе разработки. Сюда относят все затраты, связанные с расходами на носители данных, бумагу для печатающих устройств и другие материалы, необходимые для разработки программного продукта. Суммарные расходы на материалы составляют 18 300 тенге и указаны в таблице 5.3.

Т а б л и ц а 5.3 - Расходы на статью «Материалы»

Наименование материала	Единицы измерения	Количество	Цена за единицу, тенге	Сумма в тенге
1.Лазерные компакт-диски для создания архивов	шт.	60	30	1 800
2.Бумага для печатающих устройств (формат А4)	пачка	5	1100	5 500
3.Картридж для принтера	шт.	1	11 000	11 000
ИТОГО:				18 300

Расходы на аренду производственного помещения

Длительность цикла разработки программного продукта составляет 25 календарных дней. Аренда помещения обходится в 150 000 тенге в месяц.

$$P_a = 150000 \cdot 1 = 150000 \text{ тенге}$$

Расходы на программное обеспечение

В качестве операционной системы была выбрана Microsoft Windows 7 на 3 компьютера стоимостью 27 556 тенге.

Расчет затрат на амортизацию

В процессе разработки программного обеспечения необходимы технические и программные средства. При разработке программного продукта используется оборудование, которое уже имеется на предприятии: три компьютера с установленной операционной системой, многофункциональное устройство HP LaserJet M1522.

Сводные данные стоимости используемого оборудования и годовая норма амортизации представлены в таблице 5.4.

Т а б л и ц а 5.4 - Стоимость оборудования и программного обеспечения

Наименование изделий	Тип	Количество единиц	Цена за единицу в тенге	Общая сумма в тенге	Годовая норма амортизации, %
Компьютер AMDx64 3800+ MHz + монитор LG LI720	Тех. средства	3	75 000	225 000	40
МФУ Hewlett Packard LaserJet M1522	Тех. средства	1	50 000	50 000	40
Microsoft Windows 7	ПО	1	27 556	27 556	15
Итог:		5		275 000	

Кодекс Республики Казахстан от 10 декабря 2014 года № 99-IV (по состоянию на 10 сентября 2014 года) О налогах и других обязательных платежах в бюджет (Налоговый кодекс) статья 120 - Исчисление амортизационных отчислений по группе компьютеры и оборудование для обработки информации.

Амортизационные отчисления на основные средства за 25 дней разработки программного продукта вычисляются по формуле

$$A = \frac{H_A \cdot K_{обор}}{365 \cdot 100\%} \cdot 25 \qquad (5.6)$$

где $Н_А$ - норма амортизации;

$К_{обор}$ - стоимость оборудования.

Амортизационные отчисления за весь период разработки ПО составляют

$$A_К = \frac{40 \cdot 225\,000 \cdot 25}{100\% \cdot 365} = 6165 \text{ тенге},$$

$$A_П = \frac{40 \cdot 50000 \cdot 25}{100\% \cdot 365} = 1370 \text{ тенге},$$

$$A_{ПО} = \frac{15 \cdot 27556 \cdot 25}{100\% \cdot 365} = 283 \text{ тенге}.$$

Итого, суммарная амортизация на технические средства составляет

$$A = A_К + A_П + A_{ПО} = 6165 + 1370 + 283 = 7304 \text{ тенге}$$

Расчет затрат на электроэнергию

Поскольку в процессе производства используется электрооборудование, то необходимо рассчитать затраты на электроэнергию.

Затраты на электроэнергию рассчитывается по следующей формуле

$$Э = W \cdot T \cdot S \qquad (5.7)$$

где W- потребляемая мощность, кВт;

T - количество часов работы;

S - стоимость киловатт - часа электроэнергии.

Виды используемого оборудования, а так же потребляемая ими мощность представлены в таблице 5.5. Исходя из этих данных рассчитывается стоимость расхода электроэнергии.

Т а б л и ц а 5.5 - Потребляемая мощность оборудования

Наименование	Потребляемая мощность, Вт/час
Персональный компьютер	400х3
Принтер	40
Итого:	1240

Согласно установленному тарифу по энергопотреблению стоимость 1 кВт на 2011 год составляет 12,34 тенге за 1кВтч.

С учетом длительности 8-часового рабочего дня и длительности разработки и внедрения проекта по таблице 4.1, количество часов работы составит

$$Т = 8 \cdot 19 = 152 \text{ часа}$$

Таким образом, расходы на электроэнергию составят

$$Э = 152 \cdot 1{,}24 \cdot 8{,}84 = 2326 \text{ тенге}$$

Расчет затрат на накладные расходы

Накладные расходы составляют 25% от всех понесенных затрат и рассчитывается по формуле

$$Н = 0{,}25 \cdot (ФОТ + Ос + А + Э + Рм + Ра) \quad (5.8)$$

Н = 0,25·(209 595 + 230 55,45 + 7 021 + 2 326+ 18 300+150 000) =
= 102 574,36 тенге

Расчет общей себестоимости

Итого, себестоимость затрат по разработке программного продукта в соответствии с формулой составляет

С = 209 595 + 23055,45 + 7 021 + 27 556 + 2 326 + 18 300+ 150 000 +
+ 102 409,6 = 540 263,05тенге

Сводные результаты расчета стоимости проекта и их структура представлена в таблице 5.6 и на рисунке 5.1.

Т а б л и ц а 5.6 - Себестоимость

Статья расхода	Затраты в месяц, тенге	Структура, %
Фонд оплаты труда	64750	41
Социальный налог	23 055	5
Расходы на программное обеспечение	27 556	5
Амортизационные отчисления	7 304	1
Расходы на аренду помещения	150 000	29
Расходы на материалы	18 300	4
Затраты на электроэнергию	2 326	0
Накладные расходы	102 574,36	20
Итого:	27 6490,21	100

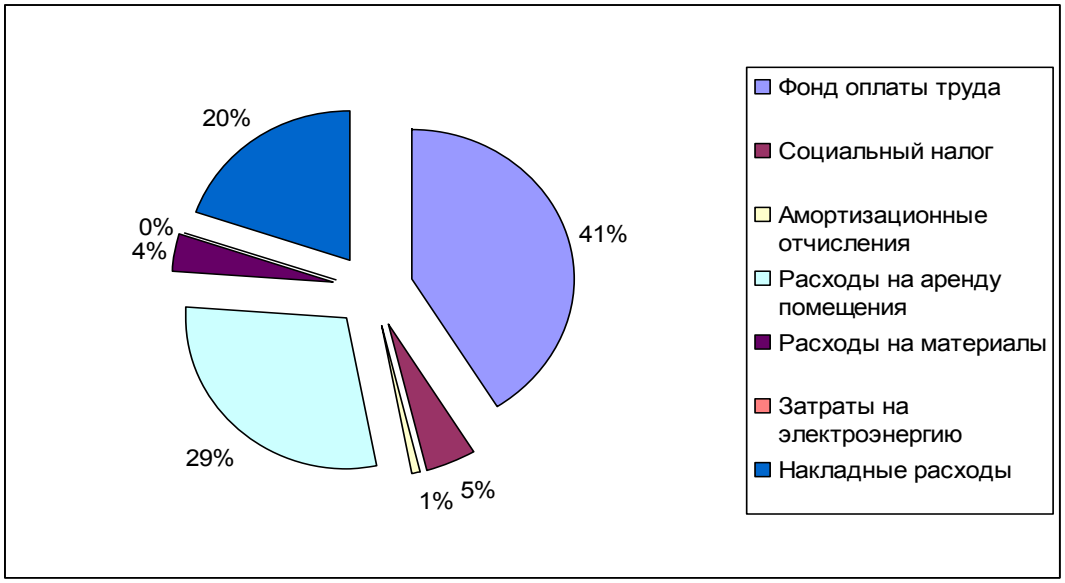


Рисунок 5.1 - Стоимость проекта и его структура

Цена программного продукта

Цена реализации разрабатываемого программного обеспечения складывается из себестоимости и чистого дохода, и вычисляется по формуле

$$Ц = C + П \tag{5.9}$$

где С - себестоимость продукта;

П - чистый доход.

Первоначальная цена рассчитывается через рентабельность проекта. Учитывая, что желаемый уровень рентабельности для отрасли разработки программного обеспечения 30%, применим следующую формулу

$$Ц_{П} = C \cdot \left(1 + \frac{P}{100}\right) \tag{5.10}$$

где Р - рентабельность (30%).

Тогда первоначальная цена равна

$$Ц_{П} = 540\,263{,}05 \cdot \left(1 + \frac{30}{100}\right) = 718020{,}53 \approx 718021 \text{ тенге}$$

Цена реализации готовой продукции рассчитывается по формуле

$$Ц_{Р} = Ц_{П} + НДС \tag{5.11}$$

Размер НДС составляет 12%, следовательно

$$НДС = \frac{12}{100} \cdot Ц_{П} \quad (5.12)$$

$$НДС = \frac{12}{100} \cdot 718021 = 86163 \text{ тенге}$$

Тогда цена реализации составит

$$Ц_{Р} = 718021 + 86163 = 804184 \text{ тенге}$$

Определяем прибыль

Прибыль = 718 021 - 512 871,81 = 205 149,19 тенге

5.3 Вывод по пятой главе

Конечная цена разрабатываемого продукта является вполне соответствующей за систему управления базами данных, обладающую таким функционалом и возможностями.

Для сравнения лицензия на неограниченное количество пользователей для SQLyog стоит $6 499 (947 554,2 тенге).

Экономическое обоснование кроме расчета себестоимости демонстрирует экономическую эффективность разработки и доказывает существенную выгоду внедрения в систему управления содержимым сайта RIX именно этой системы управления базами данных, а не коммерческих аналогов.

Основываясь на вышеизложенном, можно предположить, что при успешном внедрении возможен довольно хороший спрос на конечную систему управления содержимым сайта RIX среди целевой аудитории, к которой относятся государственные и негосударственные предприятия, желающие использовать её для управления содержимым сайта, что в дальнейшем может способствовать коммерческой реализации разработанного решения.

Заключение

В соответствии с поставленным заданием, мной была разработана и реализована системы управления базами данных для CMS RIX. При рассмотрении и сравнении технологии разработки, принципов реализации продукта и определенных технических особенностей, можно заявить, что система является уникальной в своем роде.

Существует множество других систем, выполняющих функции, схожие с разработанной системой, но в силу их общей направленности, можно сделать вывод, что внедрение системы разработанной непосредственно для заданных условий является оптимальным решением проблем, связанных с управлением базами данных для CMS RIX. Малые системные требования, надежность, простота внедрения и использования в достаточной мере подтверждают это заключение.

Введение в эксплуатацию разработанной системы, послужит облегчению работы с базами данных в CMS RIX, что в свою очередь отразится на имидже данной CMS RIX и поможет дальнейшему ее развитию и продвижению.

Список литературы

1 Базылов К.Б., Алибаева С.А., Бабич А.А. Методические указания по выполнению экономического раздела выпускной работы бакалавра для студентов всех форм обучения специальности 050719 - Радиотехника электроника и телекоммуникации. - Алматы: АИЭС, - 2008. - 19 с.

2 «Гигиенические требования к организации и условиям работы с видеодисплейными терминалами и персональными электронно-вычислительными машинами». Санпин №1.01.004.01. - 215 с.

3 Баклашов Н.И., Китаева Н.Ж., Терехов Б.Д. Охрана труда на предприятиях связи и охрана окружающей среды: Учебник для вузов - М.: Радио и связь, 1989. - 288 с.

4 Хакимжанов Т.Е., Расчет аспирационных систем, методические указания для студентов всех форм обучения. - Алматы: АИЭС, 2002 г.

5 http://ru.wikipedia.org/.

6 СНиП РК 2.04-05-2002 Естественное и искусственное освещение. Государственные нормативы в области архитектуры, градостроительства и строительства. - Издательство стандартов Астана, 2010 г.

7 Справочная книга по светотехнике/ Под ред. М. Б. Айзенберга. - М. 1983.

8 Никитин В.Д. Расчет освещения точечным методом. - Томск: Изд. ТПИ им. С. М. Кирова, 1985.

9 Справочная книга для проектирования электрического освещения/ Под ред. Г. М. Кнорринга. - Л.: Энергия, 1976.

10 Дари К., Бринзаре Б., Черчез-Тоза Ф., Бусика М. AJAX и PHP: разработка динамических веб-приложений. - СПб.: Символ Плюс, 2006. - С. 336.

11 К. Уэ ц. JavaScript. Карманный справочник. - М.: «Вильямс», 2007. - С. 272.

12 Корнеев В.В. Базы данных. Интеллектуальная обработка информации/ В. В. Корнеев, А. Ф. Гареев, С. В. Васютин. - Издательство «Нолидж», 2000. - 379с.

13 Манжолова К.Б., Алибаева С.А. Экономика предприятий телекоммуникаций: Учебное пособие. - Алматы: АИЭС, 2003.

14 Менеджмент предприятий электросвязи: Учебник для вузов/ Е.В.Демина, Н.П. Резникова, А.С. Добронравов, В.В.Макаров.-М.: Радио и связь, 2005.

15 Налоговый кодекс Республики Казахстан от 10 декабря 2008 года с изменениями и дополнениями за 2011 год. - Алматы: Юрист, 2011.

Приложение А
Исходный код

```
<?php
error_reporting(E_ALL);
if (function_exists('date_default_timezone_set'))
     date_default_timezone_set('Greenwich');
if (!session_id())
     session_start();
define("MAIN_DIR", dirname(__FILE__) . "/");
define("INCLUDES_DIR", MAIN_DIR . "includes/");

include MAIN_DIR . "config.php";
include INCLUDES_DIR . "types.php";
include INCLUDES_DIR . "class/GetTextReader.php";

if (version_compare(PHP_VERSION, "5.0.0", "<"))
     include INCLUDES_DIR . "class/Sql-php4.php";
else
     include INCLUDES_DIR . "class/Sql.php";

define("VERSION_NUMBER", "1.3.3");
define("PREVIEW_CHAR_SIZE", 75);

$adapterList[] = "mysql";

if (function_exists("sqlite_open") || (class_exists("PDO") &&
in_array("sqlite", PDO::getAvailableDrivers()))) {
     $adapterList[] = "sqlite";
}

$cookieLength = time() + (60*24*60*60);
$langList['en_US'] = "English";
$langList['ru_RU'] = "Русский";
if (isset($_COOKIE['sb_lang']) && array_key_exists($_COOKIE['sb_lang'],
$langList)) {
     $lang = preg_replace("/[^a-z0-9_]/i", "", $_COOKIE['sb_lang']);
} else {
     $lang = "en_US";
}

if ($lang != "en_US") {
     // увеличивает длину cookie
     setcookie("sb_lang", $lang, $cookieLength);
} else if (isset($_COOKIE['sb_lang'])) {

     setcookie("sb_lang", "", time() - 10000);
}

$themeList["classic"] = "Classic";
$themeList["bittersweet"] = "Bittersweet";

if (isset($_COOKIE['sb_theme'])) {
     $currentTheme = preg_replace("/[^a-z0-9_]/i", "",
$_COOKIE['sb_theme']);

     if (array_key_exists($currentTheme, $themeList)) {
```

```
            $theme = $currentTheme;
      Продолжение приложения 1
            // увеличивает длину cookie
            setcookie("sb_theme", $theme, $cookieLength);
      } else {
            $theme = "bittersweet";
            setcookie("sb_theme", "", time() - 10000);
      }
} else {
      $theme = "bittersweet";
}

$gt = new GetTextReader($lang . ".pot");

if (isset($_SESSION['SB_LOGIN_STRING'])) {
      $user = (isset($_SESSION['SB_LOGIN_USER'])) ?
$_SESSION['SB_LOGIN_USER'] : "";
      $pass = (isset($_SESSION['SB_LOGIN_PASS'])) ?
$_SESSION['SB_LOGIN_PASS'] : "";
      $conn = new SQL($_SESSION['SB_LOGIN_STRING'], $user, $pass);
}

// уникальный идетификатор для текущей сессии, чтобы проверить ajax
//запросы.
$requestKey = substr(md5(session_id() . $_SERVER["DOCUMENT_ROOT"]), 0,
16);

if (isset($conn) && $conn->isConnected()) {
      if (isset($_GET['db']))
            $db = $conn->escapeString($_GET['db']);

      if (isset($_GET['table']))
            $table = $conn->escapeString($_GET['table']);

      if ($conn->hasCharsetSupport()) {

            $charsetSql = $conn->listCharset();
            if ($conn->isResultSet($charsetSql)) {
                  while ($charsetRow = $conn->fetchAssoc($charsetSql)) {
                        $charsetList[] = $charsetRow['Charset'];
                  }
            }

            $collationSql = $conn->listCollation();
            if ($conn->isResultSet($collationSql)) {
                  while ($collationRow = $conn->fetchAssoc($collationSql))
{
                        $collationList[$collationRow['Collation']] =
$collationRow['Charset'];
                  }
            }
      }
}

// отменить magic quotes
if (get_magic_quotes_gpc()) {
      $_GET = stripslashesFromArray($_GET);
```

```
      $_POST = stripslashesFromArray($_POST);

Продолжение приложения 1
$_COOKIE = stripslashesFromArray($_COOKIE);
      $_REQUEST = stripslashesFromArray($_REQUEST);
}

function stripslashesFromArray($value) {
    $value = is_array($value) ?
                array_map('stripslashesFromArray', $value) :
                stripslashes($value);

    return $value;
}

function loginCheck($validateReq = true) {
      if (!isset($_SESSION['SB_LOGIN'])){
            if (isset($_GET['ajaxRequest']))
                  redirect("login.php?timeout=1");
            else
                  redirect("login.php");
            exit;
      }
      if ($validateReq) {
            if (!validateRequest()) {
                  exit;
            }
      }

      startOutput();
}

function redirect($url) {
      if (isset($_GET['ajaxRequest']) || headers_sent()) {
            global $requestKey;
            ?>
            <script type="text/javascript" authkey="<?php echo
$_GET['requestKey']; ?>">

            document.location = "<?php echo $url; ?>" +
window.location.hash;

            </script>
            <?php
      } else {
            header("Location: $url");
      }
      exit;
}

function validateRequest() {
      global $requestKey;
      if (isset($_GET['requestKey']) && $_GET['requestKey'] !=
$requestKey) {
            return false;
      }
```

```
     return true;
}

function startOutput() {
     Продолжение приложения 1
global $sbconfig;

     if (!headers_sent()) {
          if (extension_loaded("zlib") &&
((isset($sbconfig['EnableGzip']) && $sbconfig['EnableGzip'] == true) ||
!isset($sbconfig['EnableGzip'])) && !ini_get("zlib.output_compression")
&& ini_get("output_handler") != "ob_gzhandler") {
               ob_start("ob_gzhandler");
          } else {
               ob_start();
          }

          register_shutdown_function("finishOutput");
     }
}

function finishOutput() {
     global $conn;

     ob_end_flush();

     if (isset($conn) && $conn->isConnected()) {
          $conn->disconnect();
          unset($conn);
     }
}

function outputPage() {

global $requestKey;
global $sbconfig;
global $conn;
global $lang;

?><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/REC-html40/strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" version="-//W3C//DTD XHTML
1.1//EN" xml:lang="en">
     <head>
          <title>DBMS for RIX CMS</title>
          <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8"/>
          <link type="text/css" rel="stylesheet" href="<?php echo
smartCaching("css/common.css"); ?>" media="all" />
          <link type="text/css" rel="stylesheet" href="<?php echo
smartCaching("css/navigation.css"); ?>" media="all" />
          <link type="text/css" rel="stylesheet" href="<?php echo
smartCaching("css/print.css"); ?>" media="print" />
          <link type="text/css" rel="stylesheet" href="<?php echo
themeFile("css/main.css"); ?>" media="all" />
```

```
          <!--[if lte IE 7]>
          <link type="text/css" rel="stylesheet" href="<?php echo
themeFile("css/ie.css"); ?>" media="all" />
          <![endif]-->
          <script type="text/javascript" src="<?php echo
smartCaching("js/mootools-1.2-core.js"); ?>"></script>
     Продолжение приложения 1
<script type="text/javascript" src="<?php echo
smartCaching("js/helpers.js"); ?>"></script>
          <script type="text/javascript" src="<?php echo
smartCaching("js/core.js"); ?>"></script>
          <script type="text/javascript" src="<?php echo
smartCaching("js/movement.js"); ?>"></script>
     </head>
     <body>
     <div id="container">
     <div id="header">
          <div id="headerlogo">
          <div id="toptabs"><ul></ul></div>
          <div id="headerinfo">
          <span id="load" style="display: none"><?php echo
__("Loading..."); ?></span>
          <?php

          // if set to auto login, providing a link to logout wouldnt be
much good
          if (!((isset($sbconfig['DefaultPass']) && $conn->getAdapter()
== "mysql") || (isset($sbconfig['DefaultDatabase']) && $conn-
>getAdapter() == "sqlite")))
               echo '<a href="logout.php">' . __("Logout") . '</a>';

          ?>
          </div>
          <div class="clearer"></div>
     </div>

     <div id="bottom">

     <div id="leftside">
          <div id="sidemenu">
          <div class="dblist"><ul>
          <?php

          if ($conn->getAdapter() != "sqlite") {

          ?>
               <li id="sidehome"><a href="#page=home"
onclick="sideMainClick('home.php', 0); return false;"><div
class="menuicon">&gt;</div><div class="menutext"><?php echo __("Home");
?></div></a></li>
               <li id="sideusers"><a href="#page=users&topTab=1"
onclick="sideMainClick('users.php', 1); return false;"><div
class="menuicon">&gt;</div><div class="menutext"><?php echo __("Users");
?></div></a></li>
               <li id="sidequery"><a href="#page=query&topTab=2"
onclick="sideMainClick('query.php', 2); return false;"><div
```

```
class="menuicon">&gt;</div><div class="menutext"><?php echo __("Query");
?></div></a></li>
                <li id="sideimport"><a href="#page=import&topTab=3"
onclick="sideMainClick('import.php', 3); return false;"><div
class="menuicon">&gt;</div><div class="menutext"><?php echo __("Import");
?></div></a></li>
                <li id="sideexport"><a href="#page=export&topTab=4"
onclick="sideMainClick('export.php', 4); return false;">
     Продолжение приложения 1
<div class="menuicon">&gt;</div><div class="menutext"><?php echo
__("Export"); ?></div></a></li>
          <?php

          } else {

          ?>
                <li id="sidehome"><a href="#page=home"
onclick="sideMainClick('home.php', 0); return false;"><div
class="menuicon">&gt;</div><div class="menutext"><?php echo __("Home");
?></div></a></li>
                <li id="sidequery"><a href="#page=query&topTab=1"
onclick="sideMainClick('query.php', 1); return false;"><div
class="menuicon">&gt;</div><div class="menutext"><?php echo __("Query");
?></div></a></li>
                <li id="sideimport"><a href="#page=import&topTab=2"
onclick="sideMainClick('import.php', 2); return false;"><div
class="menuicon">&gt;</div><div class="menutext"><?php echo __("Import");
?></div></a></li>
                <li id="sideexport"><a href="#page=export&topTab=3"
onclick="sideMainClick('export.php', 3); return false;"><div
class="menuicon">&gt;</div><div class="menutext"><?php echo __("Export");
?></div></a></li>
          <?php

          }

          ?>
          </ul></div>

          <div class="dblistheader"><?php echo __("Databases"); ?></div>
          <div class="dblist" id="databaselist"><ul></ul></div>
          </div>
     </div>
     <div id="rightside">

          <div id="content">
                <div class="corners"><div class="tl"></div><div
class="tr"></div></div>
                <div id="innercontent"></div>
                <div class="corners"><div class="bl"></div><div
class="br"></div></div>
          </div>

          </div>
```

```
      </div>
      </div>

      </body>
      <script type="text/javascript">
      <!--

      <?php

      if ($conn->getAdapter() == "sqlite") {
            echo "var showUsersMenu = false;\n";
      Продолжение приложения 1
} else {
            echo "var showUsersMenu = true;\n";
      }

      echo "var adapter = \"" . $conn->getAdapter() . "\";\n";

      if (isset($requestKey)) {
            echo 'var requestKey = "' . $requestKey . '";';
            echo "\n";
      }

      // перевод строк javascript
      echo "\t\tvar getTextArr = {";
            if ($lang != "en_US") {
                  echo '"Home":"' . __("Home") . '", ';
            echo '"Users":"' . __("Users") . '", ';
            echo '"Query":"' . __("Query") . '", ';
            echo '"Import":"' . __("Import") . '", ';
            echo '"Export":"' . __("Export") . '", ';

            echo '"Overview":"' . __("Overview") . '", ';

            echo '"Browse":"' . __("Browse") . '", ';
            echo '"Structure":"' . __("Structure") . '", ';
            echo '"Insert":"' . __("Insert") . '", ';

            echo '"Your changes were saved to the database.":"' . __("Your
changes were saved to the database.") . '", ';

            echo '"delete this row":"' . __("delete this row") . '", ';
            echo '"delete these rows":"' . __("delete these rows") . '",
';
            echo '"empty this table":"' . __("empty this table") . '", ';
            echo '"empty these tables":"' . __("empty these tables") . '",
';
            echo '"drop this table":"' . __("drop this table") . '", ';
            echo '"drop these tables":"' . __("drop these tables") . '",
';
            echo '"delete this column":"' . __("delete this column") . '",
';
            echo '"delete these columns":"' . __("delete these columns") .
'", ';
            echo '"delete this index":"' . __("delete this index") . '",
';
```

```
            echo '"delete these indexes":"' . __("delete these indexes") .
'", ';
            echo '"delete this user":"' . __("delete this user") . '", ';
            echo '"delete these users":"' . __("delete these users") . '",
';
            echo '"Are you sure you want to":"' . __("Are you sure you
want to") . '", ';

            echo '"The following query will be run:":"' . __("The
following query will be run:") . '", ';
            echo '"The following queries will be run:":"' . __("The
following queries will be run:") . '", ';

            Продолжение приложения 1
echo '"Confirm":"' . __("Confirm") . '", ';
            echo '"Are you sure you want to empty the \'%s\' table? This
will delete all the data inside of it. The following query will be
run:":"' . __("Are you sure you want to empty the '%s' table? This will
delete all the data inside of it. The following query will be run:") .
'", ';
            echo '"Are you sure you want to drop the \'%s\' table? This
will delete the table and all data inside of it. The following query will
be run:":"' . __("Are you sure you want to drop the '%s' table? This will
delete the table and all data inside of it. The following query will be
run:") . '", ';
            echo '"Are you sure you want to drop the database \'%s\'? This
will delete the database, the tables inside the database, and all data
inside of the tables. The following query will be run:":"' . __("Are you
sure you want to drop the database '%s'? This will delete the database,
the tables inside the database, and all data inside of the tables. The
following query will be run:") . '", ';

            echo '"Successfully saved changes":"' . __("Successfully saved
changes") . '", ';

            echo '"New field":"' . __("New field") . '", ';

            echo '"Full Text":"' . __("Full Text") . '", ';

            echo '"Loading...":"' . __("Loading...") . '", ';
            echo '"Redirecting...":"' . __("Redirecting...") . '", ';

            echo '"Okay":"' . __("Okay") . '", ';
            echo '"Cancel":"' . __("Cancel") . '", ';

            echo '"Error":"' . __("Error") . '", ';
            echo '"There was an error receiving data from the server":"' .
__("There was an error receiving data from the server") . '"';

      }

      echo '};';

      echo "\n";
```

```
      echo 'var menujson = {"menu": [';
      echo $conn->getMetadata();
      echo ']};';

      ?>
      //-->
      </script>
</html>
<?php
}

function requireDatabaseAndTableBeDefined() {
      global $db, $table;

      if (!isset($db)) {?>
      Продолжение приложения 1
            <div class="errorpage">
            <h4><?php echo __("Oops"); ?></h4>
            <p><?php echo __("For some reason, the database parameter was
not included with your request."); ?></p>
            </div>

            <?php
            exit;
      }

      if (!isset($table)) {
            ?>

            <div class="errorpage">
            <h4><?php echo __("Oops"); ?></h4>
            <p><?php echo __("For some reason, the table parameter was not
included with your request."); ?></p>
            </div>

            <?php
            exit;
      }

}

function formatForOutput($text) {
      $text = nl2br(htmlentities($text, ENT_QUOTES, 'UTF-8'));
      if (utf8_strlen($text) > PREVIEW_CHAR_SIZE) {
            $text = utf8_substr($text, 0, PREVIEW_CHAR_SIZE) . " <span
class=\"toBeContinued\">[...]</span>";
      }
      return $text;
}

function formatDataForCSV($text) {
      $text = str_replace('"', '""', $text);
      return $text;
}

function splitQueryText($query) {
```

```
	// the regex needs a trailing semicolon
	$query = trim($query);

	if (substr($query, -1) != ";")
		$query .= ";";

	// i spent 3 days figuring out this line
	preg_match_all("/(?>[^;']|(''|(?>'([^']|\\')*[^\\\]')))+;/ixU",
$query, $matches, PREG_SET_ORDER);

	$querySplit = "";

	foreach ($matches as $match) {
		// get rid of the trailing semicolon
		$querySplit[] = substr($match[0], 0, -1);
	}

	Продолжение приложения 1
return $querySplit;
}

function memoryFormat($bytes) {
	if ($bytes < 1024)
		$dataString = $bytes . " B";
	else if ($bytes < (1024 * 1024))
		$dataString = round($bytes / 1024) . " KB";
	else if ($bytes < (1024 * 1024 * 1024))
		$dataString = round($bytes / (1024 * 1024)) . " MB";
	else
		$dataString = round($bytes / (1024 * 1024 * 1024)) . " GB";

	return $dataString;
}

function themeFile($filename) {
	global $theme;
	return smartCaching("themes/" . $theme . "/" . $filename);
}

function smartCaching($filename) {
	return $filename . "?ver=" . str_replace(".", "_", VERSION_NUMBER);
}

function __($t) {
	global $gt;
	return $gt->getTranslation($t);
}

function __p($singular, $plural, $count) {
	global $gt;
	if ($count == 1) {
		return $gt->getTranslation($singular);
	} else {
		return $gt->getTranslation($plural);
	}
}
```

```
function utf8_substr($str, $from, $len) {
# utf8 substr
# www.yeap.lv
  return preg_replace('#^(?:[\x00-\x7F]|[\xC0-\xFF][\x80-
\xBF]+){0,'.$from.'}'.
                      '((?:[\x00-\x7F]|[\xC0-\xFF][\x80-
\xBF]+){0,'.$len.'}).*#s',
                      '$1',$str);
}

function utf8_strlen($str) {
    $i = 0;
    $count = 0;
    $len = strlen ($str);
    while ($i < $len) {
    $chr = ord ($str[$i]);
    $count++;
    $i++;
    Продолжение приложения 1
if ($i >= $len)
        break;

    if ($chr & 0x80) {
        $chr <<= 1;
        while ($chr & 0x80) {
        $i++;
        $chr <<= 1;
        }
    }
    }
    return $count;
}

function microtime_float() {
    list($usec, $sec) = explode(" ", microtime());
    return ((float)$usec + (float)$sec);
}

?>
```