Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Факультет математики, механики и компьютерных технологий
Кафедра прикладной математики и программирования
Направление подготовки Прикладная математика и информатика

РАБОТА ПРОВЕРЕНА
Рецензент,

_____
«____»_____2017г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой, д.ф.-м.н.,
доцент
_____А.А.Замышляева
«____»_____2017 г.

Создание сайта «Портфолио программиста»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ–01.03.02.2017.24.ПЗ ВКР

Руководитель работы, доцент
_____/М.Ю. Катаргин
«____»_____2017 г.
Автор работы
Студент группы ЕТ-482
_____/ М.С. Горбунов
«____»_____2017 г.
Нормоконтролер, доцент
_____/Т.Ю. Оленчикова
«____»_____2017 г.

Челябинск 2017

АННОТАЦИЯ

Горбунов М.С. Создание сайта «Портфолио программиста» – Челябинск: ЮУрГУ, ММиКН-482, 74 с., 32 ил.,10 табл., библиогр. список – 17 наим., 1 прил.

Выпускная квалификационная работа посвящена разработке web-приложения, позволяющего любому программисту создать сайт-портфолио без изменения программного кода. Созданы: база данных, хранящая информационную нагрузку сайта; web-сценарии, отвечающие за работу с базой и отработку действий посетителей. Приложение реализовано с использованием следующих средств: стандартизованный язык разметки web-документов HTML, объектно-ориентированный скриптовый язык JavaScript с использованием библиотеки JQuery, язык серверных сценариев PHP (Hypertext Preprocessor), web-сервер Apache и сервер баз данных MySQL. В выпускной квалификационной работе продемонстрирован пример работы созданного приложения, который наполнен собственными проектами. В приложениях представлена программная реализация данного программного продукта.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Целью выпускной квалификационной работы является создание web-приложения: сайт «Портфолио программиста», которое является «скелетом» для создания сайтов другими программистами без изменения программного кода. Предполагается, что контент сайта находится в базе данных и внешних файлах. Такие файлы могут создаваться без знания языков web-программирования, например, через обычный редактор Microsoft Word. Так же пользователю нужно будет наполнить базу данных своими записями, это можно сделать через приложение phpMyAdmin или другое приложение, позволяющее наполнять и редактировать базу данных.

Рассмотрим подробней, что же такое сайт. Сайт – совокупность объединённых логически web-страниц в единое целое, которые представляют собой рекламно-информационные ресурсы, объединённые общей идеей и дизайном.

Для поиска и отображения данных используются специальные программы – web-браузеры. Вся информация, отображающаяся в браузерах, видна в виде web-страниц, которые являются основной структурной единицей мировой сети. Web-страницы являются мультимедийными, т.е. объединяют в себе: текст, графику, анимацию, музыку и видео. От качества работы по созданию web-страницы во многом зависит её популярность в сети.

Так как с каждым годом популярность интернет ресурса растёт, то большинство информации размещается в сети. В частности и портфолио-программиста может быть расположено в интернете, что будет являться плюсом для него, так как программист может наглядно представить свои проекты и их описание и расположить заказчика к себе. В свободном доступе нет приложений помогающих создать сайт с нуля без особых затрат по времени, поэтому проектируемое приложение является актуальным для большинства программистов. Ближайшим аналогом создаваемого приложения является онлайн-магазин, но для переделывания его под портфолио займёт немало времени.

Для проектирования сайта были изучены сайты похожей тематики, такие как сайт-портфолио Александра Скалецкого (http://skalexander.ru)и сайт Alenna’sportfolio (https://alenna.ru).

После анализа упомянутых выше web-сайтов проектируемое приложение должно удовлетворять следующим требованиям:

1) показывать краткую информацию о программисте;

2) отображать список созданных проектов, сортированный по темам;

3) показывать возможные контакты.

На основе основных требований, были сформулированы основные задачи выпускной работы:

1) разработать разметку страниц;

2) разработать базу данных, в которой хранится вся информация;

3) разработать web-сценарий, с помощью которого возможно редактирование и заполнение информации, без изменения программного кода;

4) обеспечить корректное отображение информации и сортировку по темам;

5) ведение статистики посещений сайта и загрузки файлов.

Для реализации выпускной квалификационной работы были использованы следующие программные средства:

1) язык разметки гипертекста HTML;

2) каскадная таблица стилей CSS;

3) язык серверных сценариев PHP;

4) скриптовый язык JavaScript с использованием библиотеки jQuery;

5) СУБД MySQL;

6) web-сервер Apache.

1 ФОРМУЛИРОВКА ЗАДАЧИ И ТРЕБОВАНИЙ К ПРИЛОЖЕНИЮ НА ОСНОВЕ СУЩЕСТВУЮЩИХ РЕШЕНИЙ

1 Определение требований к создаваемому приложению

Создаваемое приложение является «скелетом» сайта, т.е. он представляет собой незаполненный бланк документа, а создатель сайта наполняет его определённым контентом. Этот проект может быть использован любым программистом без изменений программного кода. Тематикой проектируемого приложения является портфолио – собрание, совокупность работ, реализованных проектов. Для создания web-сайта программисту, который воспользуется создаваемым приложением, необходимо добавить в корень папки с сайтом свои собственные документы, содержащие описание его проектов, и заполнить базу данных необходимыми пунктами меню и ссылками на соответствующие внешние файлы.

Для наглядного примера внесём в базу данных названия пунктов меню и собственных файлов с решениями задач и курсовыми работами, созданными за годы обучения в университете.

Основным этапом проектирования приложения является сбор информации об аналогичных сайтах и определение требований к создаваемому приложению. Проектируемый web-сайт должен удовлетворять следующим требованиям:

1) обеспечивать предоставление информации о:
   a) программисте-владельце сайта;
   b) контактных данных;
   c) рекламируемых проектах с возможностью просматривать и скачивать исходный код этих проектов;
2) возможность навигации по разделам сайта с помощью иерархического меню;
3) разделение точек входа для владельца сайта и его посетителей. Для владельца должна быть доступна дополнительная вкладка «Статистика», в которой отображается статистика по:
   a) посещениям сайта;
   b) скачиванию файлов;
   c) просматриваемым страницам;
4) возможность производить следующие действия, без изменения программного кода:
   a) добавление новых страниц;
   b) редактирование имеющихся страниц;
   c) редактирование заголовков страниц;
5) возможность загрузки файлов из перечня файлов для скачивания, соответствующий демонстрируемым проектам;

6) заполнение блока контента списком гиперссылок на подпункты, если они существуют для выбранного пункта меню.

2 Определение целевой аудитории

При создании любого приложения необходимо учитывать целевую аудиторию пользователей. С одной стороны, это программисты, которые хотят на основе создаваемого приложения построить свой собственный сайт, с другой – это пользователи самого сайта, созданного программистом, использующим проектируемое приложение.

Рассмотрим оба варианта использования создаваемого приложения. Для этого необходимо поставить себя на место посетителя сайта или пользователя создаваемого проекта и задаться вопросом, что необходимо сделать для увеличения спроса обеими категориями пользователей web-приложения.

Пользователи приложением как «шаблоном» делятся на создателей собственного web-сайта на основе создаваемого проекта и на программистов, которые хотят найти какие-либо нововведения для своего существующего сайта.

В свою очередь множество посетителей web-сайта можно разбить на заказчиков, которые хотели бы видеть, на что способен программист и какими знаниями и умениями он обладает, и на программистов, заинтересованных в сотрудничестве с создателем сайта.

3 Сравнительный анализ web-сайтов аналогичной тематики

1.3.1 Анализ web-сайта http://skalexander.ru

Сайт Александра Скалецкого является одностраничным, шапка которого выглядит, как показано на рисунке 1.1. В ней показана краткая информация о специализации программиста, из которой видно, что Александр является исключительно web-программистом и не работает с другими языками программирования.

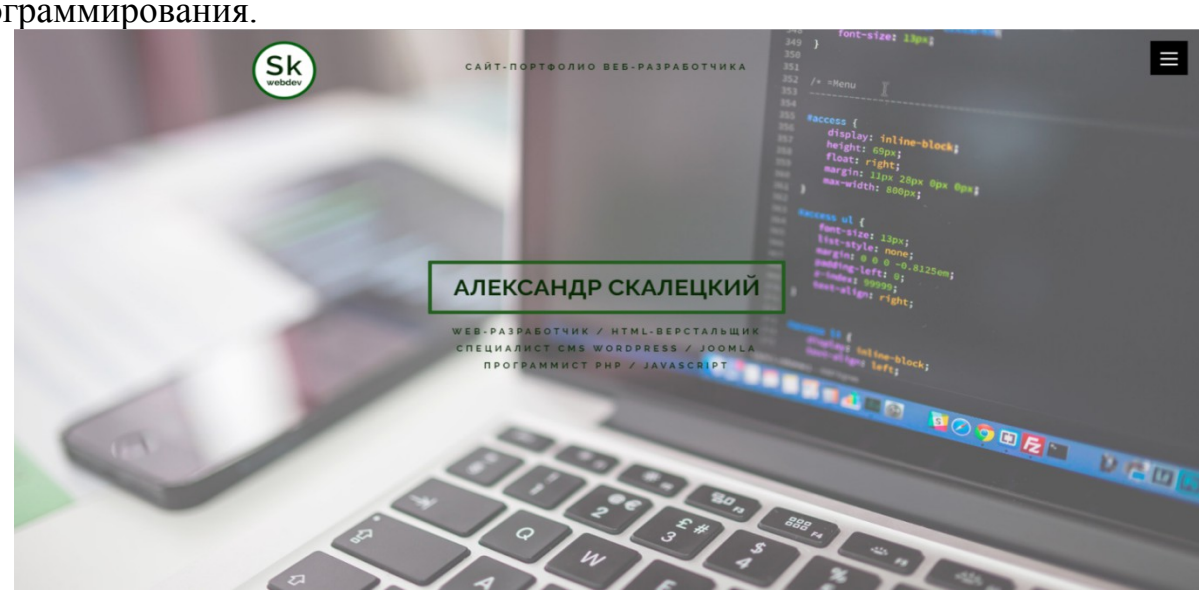

Рисунок 1.1 – Шапка сайта Александра Скалецкого

При дальнейшем изучении сайта видна информация о самом программисте, его резюме, также перечислены основные знания и навыки, образование. Далее на сайте расположено портфолио, в котором представлены последние работы Александра – рисунок 1.2.

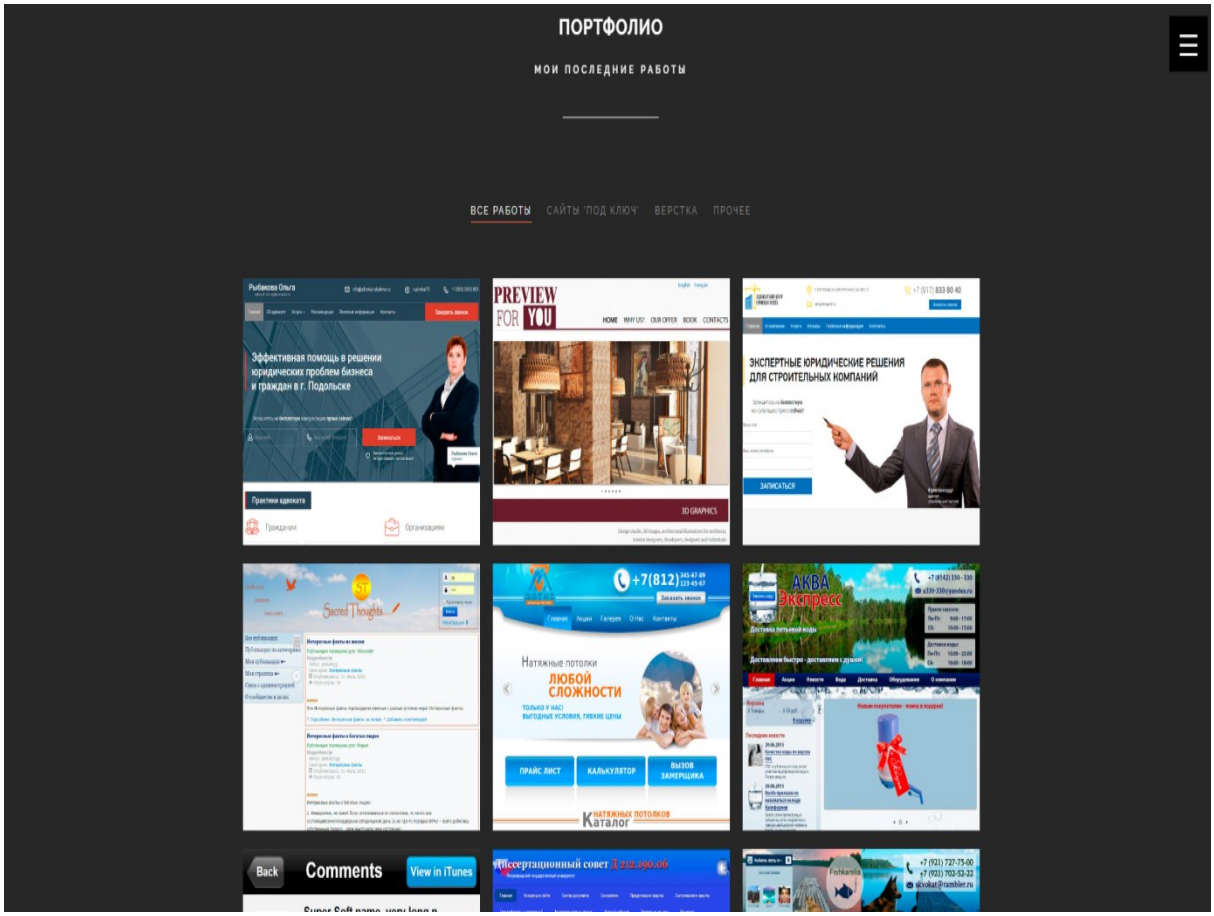

Рисунок 1.2 – Проекты Александра Скалецкого

Стоит заметить, что присутствует фильтрация проектов по темам: все работы, сайты «под ключ», верстка, прочее. Также каждый проект имеет ссылку на работающий продукт, своё описание и особенности, которые описаны в диалоговом окне (рисунок 1.3), появляющемся при нажатии на интересующий пример.

САЙТ-ВИЗИТКА ПО ДОСТАВКЕ ВОДЫ
"АКВАЭКСПРЕСС" (+ИНТЕРНЕТ-МАГАЗИН)

Ссылка на работу: aquaexpress

Создание сайта с нуля, под ключ.

Анализ сайтов-конкурентов. Разработка концепции, уникального дизайна и пользовательского интерфейса. Верстка сайта с установкой на CMS Joomla, наращивание функционала.

Особенности:

- Стоковые изображения, лицензионная графика.
- Модуль «Последние новости».
- Слайдер с акциями на главной.
- Уникальная форма заказа воды на php.
- Подключение и настройка интернет-магазина на JoomShopping.
- Модуль «Корзины».

Рисунок 1.3 – Диалоговое окно проекта Аква Экспресс

Далее расположены всевозможные контакты, такие как: e-mail, Skype, Facebook, ВКонтакте, причём, присутствует возможность написать электронное письмо, не переходя на другие сайты, как показано на рисунке 1.4.

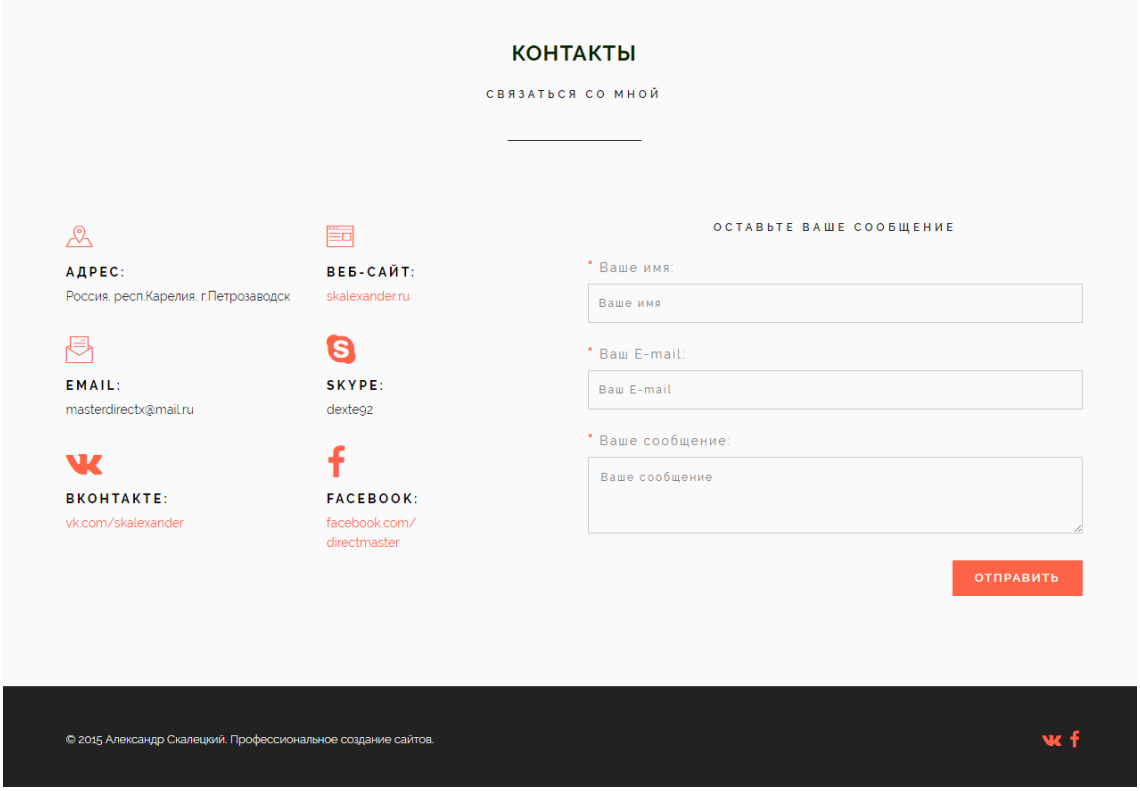

Рисунок 1.4 – Контакты Александра Скалецкого

После анализа web-сайта можно выделить положительные стороны, такие как:

1) чёткость и ясность изложения информации;
2) анимация скроллинга;
3) адаптация для мобильной версии;
4) присутствие всевозможных контактов, таких как телефон и связь в социальных сетях;
5) возможность оставить сообщение программисту, без перехода на другие сайты.

К недостаткам рассмотренного web-сайта можно отнести:

1) отсутствие прайса на услуги (со стороны заказчика);
2) отсутствие отзывов заказчиков.

1.3.2 Анализ web-сайта https://alenna.ru

На рисунке 1.5 показана главная страница web-сайта программиста Алёны. В шапке сайта находится её логотип, контакты и меню навигации по сайту. Footer главной страницы сайта аналогично шапке содержит контакты, логотип, и добавились ссылки на социальные сети Алёны.

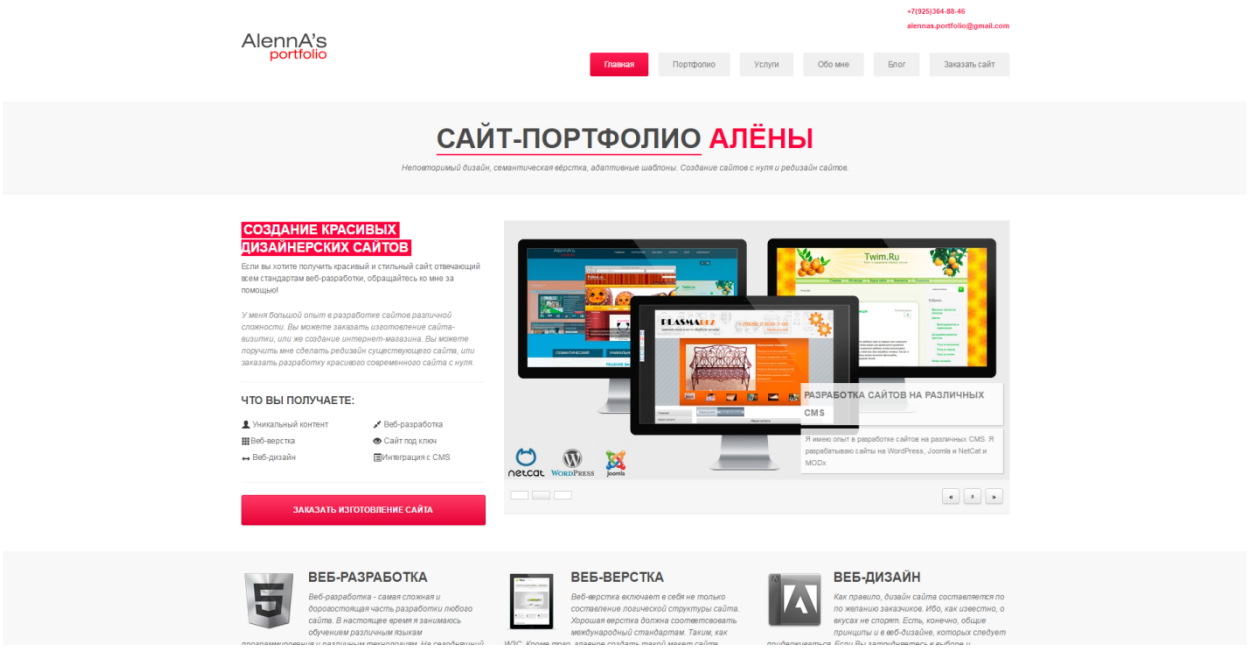

Рисунок 1.5 – главное окно сайта Алёны

При переходе по пункту меню «Портфолио» появляется возможность ознакомиться с созданными программистом проектами (рисунок 1.6). При нажатии на иконку проекта открывается диалоговое окно, которое позволит увеличить картинку, и перейти на страницу с описанием выбранного проекта. На этой странице описывается тип созданного web-сайта и перечень проведенных работ, выполненных при создании сайта, так же присутствует слайд-шоу скриншотов пользования сайтом, но прямой ссылки на созданный проект не имеется.

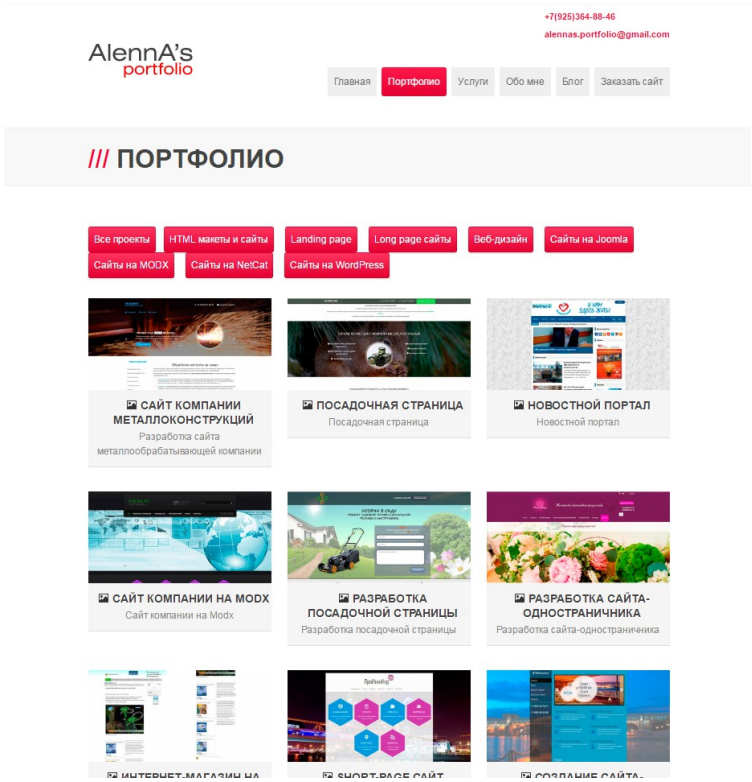

Рисунок 1.6 – Вкладка «Портфолио» на сайте Алёны

В пункте меню «Услуги» приведен прайс-лист по каждому типу работ, которые описаны после прайс-листа. Во вкладке «Обо мне» написано, почему заказчику стоит выбрать именно Алёну, аргументируется это её образованием, личными качествами, навыками и достижениями, профессиональным ростом и карьерой. Также присутствует раздел «Блог», в котором Алёна описывает некоторые темы, полезные по созданию и проектированию сайта. Прямо на сайте существует возможность заказать собственный web-сайт, для этого выделена специальная страница «Заказать сайт» (рисунок 1.7).

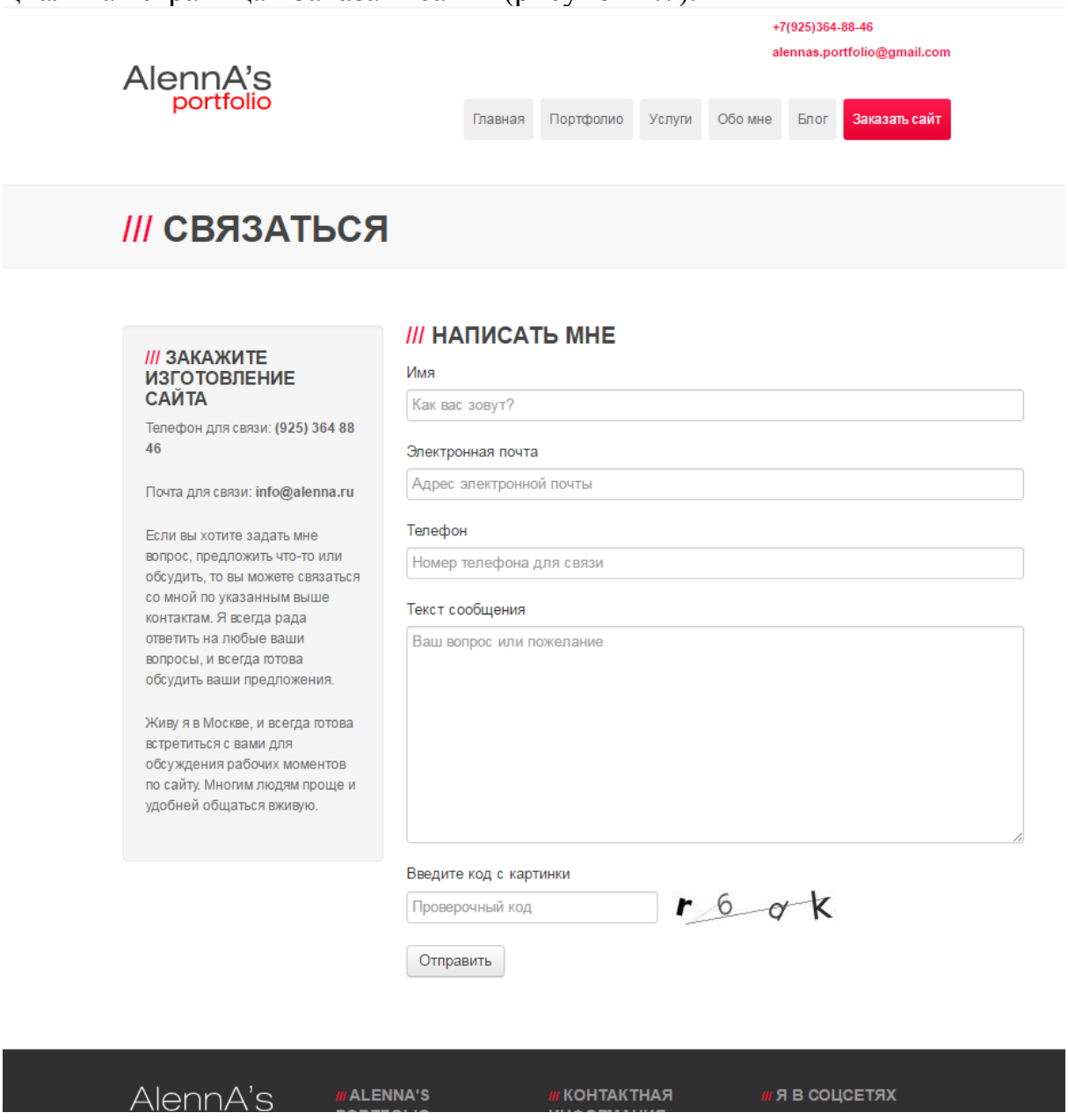

Рисунок 1.7 – Вкладка «Заказать сайт» на сайте Алёны

После анализа web-сайта можно выделить положительные стороны, такие как:

1) большое количество завершённых проектов, которые разделены по темам
2) в шапке сайта сразу находится контактная информация;
3) отзывы клиентов;
4) перечень предоставляемых услуг;

5) прайс на услуги;

6) возможность оставить сообщение программисту, без перехода на другие сайты;

7) собственный блог, сортированный по дате добавления, что является редкостью для программистов.

К недостаткам данного сайта можно отнести:

1) отсутствие контактов в социальных сетях;

2) отсутствие ссылок на свои проекты.

4 Инструментарий разработки

4.1.1 Выбор архитектуры приложения

Проектируемое приложение имеет клиент-серверную архитектуру, это значит, что клиент запрашивает у сервера некоторые услуги, а сервер обрабатывает запросы. На клиентской машине отображаются страницы документа, обрабатываемые браузером с помощью языка HTML и XHTML, также некоторые дополнения, такие как: каскадные таблицы стилей CSS, средства Dynamic HTML, клиентский язык сценариев JavaScript. Это обеспечивает привлекательный дизайн и интерфейс, быстро реагирующий на действия пользователей. Для обработки запросов пользователя со стороны сервера существуют web-сценарии, которые после выполнения возвращают сформированную web-страницу, которая является динамической. Схема работы выбранной архитектуры приложения представлена на рисунке 1.8.

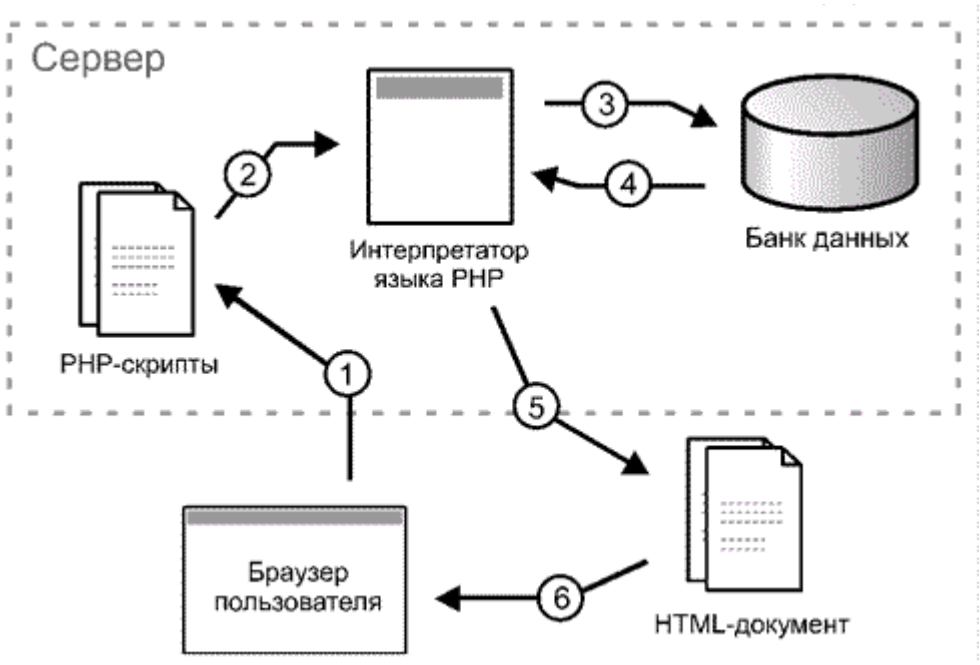

Рисунок 1.8 – Схема работы клиент-серверной архитектуры

Такая архитектура имеет преимущества, такие как:

- не требуется дублирование кода серверных сценариев клиентскими приложениями;

- так как web-сценарии выполняются на стороне сервера, то нагрузка на компьютер пользователя меньше, следовательно, и скорость работы больше;
- данные отображаемые для клиента хранятся на сервере, который защищен лучше, чем большинство посетителей сайта.

При своих преимуществах клиент-серверная архитектура имеет и свой недостаток:

- при «падении» сервера, недоступно будет и клиентское приложение.

4.1.2 Выбор языка программирования для создания web-сценариев

Для создания web-сценариев существует множество технологий. Рассмотрим статистику популярности серверных web-языков программирования по исследованиям Openstat за июнь 2014 г, представленную на рисунке 1.9 и в таблице 1.1.

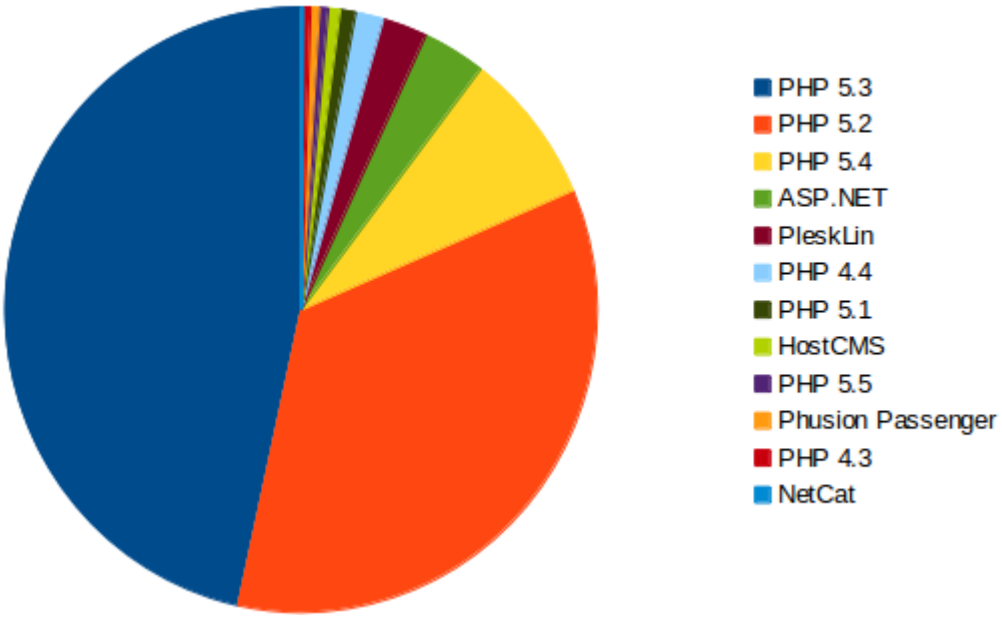

Рисунок 1.9 – Статистика по использованию web-языков

В таблице 1.1. указана доля русскоязычных сайтов относительно общего количества сайтов в сети.

Таблица 1.1

Статистика использования web-языков

Web-язык	Всего	Доля
PHP 5.3	926 254	20,97%
PHP 5.2	693 539	15,70%
PHP 5.4	162 403	3,68%
ASP.NET	68 601	1,55%
PHP 4.4	30 170	0,68%
PHP 5.1	16 879	0,38%
PHP 5.5	9 627	0,22%

Рассмотрим подробней технологии по созданию web-сценариев.

- ASP.NET (Active Server Pages) – технология создания web-приложений от Microsoft. ASP создана для работы на платформе Windows.NET. ASP– технология создания динамических

web-приложений. Для разработки таких приложений используются языки программирования основанные на .NET, такие как C# и VisualBasic. Её небольшая популярность, по сравнению с PHP, объясняется тем, что приложения, созданные на ASP.NET привязаны только к платформам семейства Windows, доступный web-сервер только – MicrosoftIIS (Internet Information Server), и рассмотренная технология имеет высокую стоимость инструментов для разработки, например VisualStudio.

- PHP (Hypertext Preprocessor) – язык сценариев для web-приложения с открытым исходным кодом. Данный язык предназначен именно для web-программирования, и этот код можно внедрять в обычный код HTML-страницы. В отличие от ASP.NET, PHP доступен почти для всех платформ, включая Unix, Windows, MacOS, RISCOS. Также PHP поддерживает и большинство web-серверов, таких как Apache, MicrosoftIIS, nginx, LiteSpeed, uServ и многие другие.

После разбора и сравнения этих технологий создания серверных сценариев для работы web-сайта для использования был выбран язык web-программирования PHP. Главными аргументами в пользу этого выбора послужили:

1) практичность
2) распространенность
3) кроссплатформенность
4) открытый программный код.

4.1.3 Выбор СУБД

СУБД – система управления базами данных, представляет собой совокупность программных и языковых средств, обеспечивающих управление созданием и работой с базами данных как одним, так и может быть несколькими пользователями.

Так как языком web-сценариев был выбран PHP, то необходимо подобрать и СУБД, взаимодействующую с выбранным языком. Рассмотрим свойства часто используемых баз данных, опять же, по версии Openstat за июнь 2014 г, представленные в таблице 1.2.

Для нашей задачи главным свойством СУБД является наличие API для работы с языком PHP, то есть способность работы с ним. Также не стоит забывать про быстрый доступ к таблицам, кроссплатформенность, многопоточность и открытый исходный код.

Таблица 1.2

Свойства баз данных

БД	bbPress	MyBB	phpBB	punBB	SMF	UseBB	Vanilla	vBulletin	YaBB	YetAnother
MySQL	+	+	+	+	+	+	+	+	-	-
Postgres	-	+	+	+	+	-	-	-	-	-
Oracle	-	-	+	-	-	-	-	-	-	-
SQLite	-	+	+	+	+	-	-	-	-	-
Textfiles	-	-	-	-	-	-	-	-	+	-
Другие			FireBird 2.0+, MSSQL 2000+	-		MySQLi (4.1, 5.x)	-		backup / restoresystem; YaBB 3	Microsoft SQL Server

Всеми этими свойствами обладает MySQL – свободная реляционная СУБД. Эта система входит в состав серверов WAMP, AppServ, LAMP и в портативные сборки серверов Денвер, XAMPP, VertigoServ. Также достоинством MySQL является поддержка большого количества типов таблиц, но в основном используются тип MyISAM, который поддерживает полнотекстный поиск, и InnoDB, поддерживающий транзакции на уровне отдельных записей. Также начиная с версии MySQL 5.2, для таблиц MyISAM было добавлено расширение Maria, которое позволяет сохранять целостность данных после краха.

Последней стабильной версией MySQL является 5.6, анонсированной 5 февраля 2013 г. К ключевым улучшения можно отнести: поддержку средств полнотекстового поиска, возможность доступа к данным через memcached API, увеличение производительность работы при интенсивной записи данных, а также увеличение масштабируемости при обработке большого числа одновременных запросов. Так же как и остальные, версия 5.6 обладает следующими возможностями:

- кроссплатформенная совместимость;
- независимые типы таблиц;
- транзакции;
- хранимые процедуры, функции, триггеры, курсоры, представления;
- поддержка SSL;
- кэширование запросов;
- поддержка таблиц MyISAM с расширением Maria;
- сегментирование (разбиение одной таблицы на несколько частей);
- совместимость со стандартом SQL2003;
- API для плагинов, позволяющие загружать сторонние модули, расширяющие функциональность, без перезапуска сервера.

4.1.4 Выбор web-сервера

Для выбора web-сервера обратимся к статистике популярности по данным Openstat за июнь 2014 г, показанной в таблице 1.3.

Таблица 1.3

Статистика популярности web-серверов

	Все	.ru	.su	.рф	.by	.ua	другое
nginx	2738947	2298084	44400	268514	39461	61436	27052
Apache	1039649	859330	18693	92550	13884	26423	28769
Microsoft IIS	79220	60332	1519	6549	2687	2180	5953
LiteSpeed	60416	53066	753	5531	242	252	572
uServ	56009	41401	727	3627	282	8556	1416
QRATOR	12400	9465	324	1654	206	619	132
lighttpd	7851	6572	237	430	41	202	369
openresty	2049	1608	21	412	1	6	1
DataPalm	1709	1545	34	117	3		10
NetNames	1628	1210	21	204	175	4	14
CommuniGatePro	1282	989	58	124	99	8	4
IdeaWebServer/v0.80	1157	1073	1	1	27	7	48

Рассмотрим топ-3 серверов, MicrosoftIIS не будет рассматриваться, так как работает только на Windows и является платным продуктом.

Nginx – web-сервер и почтовый прокси-сервер, работающий на Unix-подобных системах, но начиная с версии 0.7.52 появилась бинарная сборка под Windows.

Apache-сервер – кроссплатформенный web-сервер, поддерживается на Linux, BSD, MacOS, Windows, NovellNetWare, BeOS. Достоинствами данного сервера является его надёжность и гибкость конфигураций. Так же он позволяет подключать внешние модули для использования СУБД, сбора данных, модификаций сообщений об ошибках и т.д..

Nglix-сервер создан относительно недавно, и поэтому существует немного информации для ознакомления с ним, по сравнению с Apache. Ещё одним поводом для выбора Apache является то, что он входит в состав кроссплатформенной сборки web-сервера XAMPP, которая содержит в себе Apache, MySQL, интерпретатор скриптов PHP, что уже было выбрано ранее.

5 Выводы по разделу

После анализа web-сайтов аналогичной тематики были утверждены основные требования для создаваемого приложения, такие как:

1) предоставление информации о владельце сайта, его контактные данные и описание созданных проектах;

2) осуществление навигации по web-сайту посредством иерархического меню;

3) разделение входа для владельца сайта и его посетителей;

4) возможность просматривать и скачивать исходный код, демонстрируемых проектов;

5) иметь систему управления контентом web-сайта;

6) наличие внешних файлов, доступных для закачки, которые соответствуют созданным проектам

7) если выбранный пункт меню имеет подпункты, заполнять блок контента списком гиперссылок на них.

Были приняты решения о том, какие программные средства использовать для создания приложения, а именно:

1) архитектура приложения – клиент-серверная;

2) язык программирования для создания web-сценариев – PHP;

3) СУБД – MySQL;

4) web-сервер – XAMPP, содержащий в себе Apache, MySQLи интерпретатор скриптов PHP.

2 ПРОЕКТИРОВАНИЕ WEB-ПРИЛОЖЕНИЯ

2.1 Описание предметной области

Проектируемое web-приложение позволит его владельцу создать сайт-портфолио, информационная часть которого хранится во внешних файлах.

Для редактирования или добавления новой информации на web-сайт необходимо разработать web-сценарии, которые будут корректно отображать информации на страницах сайта при изменении содержания внешних файлов. Это поможет владельцу сайта производить следующие действия без изменения программного кода:

1) добавление, удаление и редактирование статей;
2) добавление информации о новых проектах;
3) изменение контактов;
4) редактирование тем и заголовков статей;
5) ведение статистики посещений сайта и скачиваний файлов пользователями.

При проектировании страница web-сайта была разделена на три основные части, как показано на рисунке 2.1:

1) область шапки, в которой отображаются темы статей;
2) левая часть, отвечающая за меню навигации по сайту;
3) центральная часть, в ней отображается основная информация статьи.

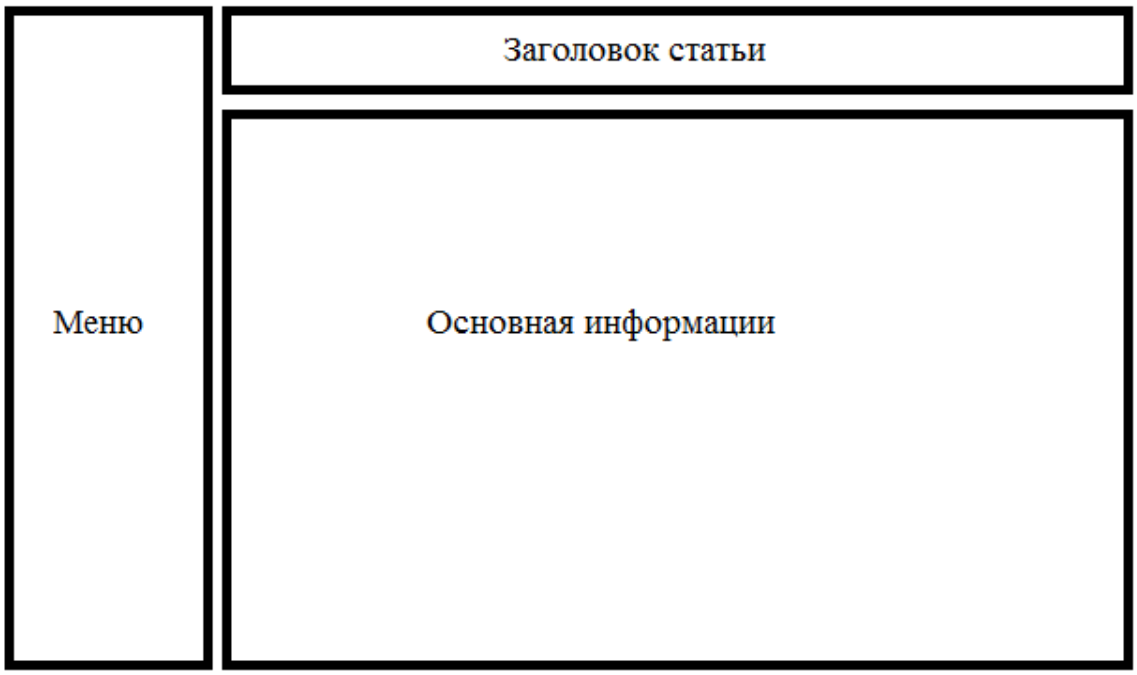

Рисунок 2.1 – Разметка страницы

На сайте должны находиться следующие страницы:

1) «главная» – содержит начальную информацию о сайте;

2) «проекты» – содержит статьи о созданных проектах, которая разделяется по темам;

3) «обо мне» – содержит информацию о создателе web-сайта;

4) «контакты» – содержит информацию о всевозможных способах связаться с автором.

Схема взаимодействия страниц сайта изображена на Рисунке 2.2.

Рисунок 2.2 – Схема взаимодействия страниц сайта

2.2 Вёрстка сайта

Вёрстка сайта представляет собой описание программным кодом визуальной части web-сайта. Сайт должен выглядеть и работать корректно на любом мониторе, независимо от его разрешения, и в любом браузере.

Существует несколько подходов к вёрстке сайта:

1) фиксируемая вёрстка – при изменении размера окна браузера блоки не меняют своего размера, что может привести к появлению полосы прокрутки;

2) резиновая вёрстка – размер блоков меняется при изменении размера окна браузера;

3) адаптивная вёрстка – создаётся на основе различных скриптов, определена для заданного разрешения, например 320, 768, 1024, поэтому изменение размеров происходит рывками, после достижения определённого разрешения;

4) отзывчивая вёрстка – слияние резиновой и адаптивной вёрстки.

Для создаваемого проекта применим резиновую вёрстку, т.е. размеры основных блоков будут зависеть от размеров окна браузера. Это позволит корректно отображать информацию сайта на мониторах, имеющих различное разрешение. Для этого размеры блоков будем рассчитывать в процентах, относительно размеров окна браузера.

Блок «Меню» (см. рисунок 2.1) будет иметь высоту 95% от общей высоты окна браузера, блоки «Заголовок статьи» и «Основная информация» находятся в общем блоке, высота которого равна, так же как и для блока «Меню», 95%. Это позволит избежать вертикальной прокрутки при изменении размеров окна браузера. Ширина блока «Меню» фиксирована и равна 200 пикселей, тогда отступ для блоков «Заголовок статьи» и «Основная информация» должен быть больше 200 пикселей, чтобы не происходило наложение. Учитывая отступы от границ браузера и блока «Меню» эта величина назначена раной 250 пикселям. Внутренние отступы для всех блоков назначены по 10 пикселей, т.е. текст внутри блоков будет отступать от их краёв на 10 пикселей. Благодаря этим действиям избавляемся от горизонтальной прокрутки.

2.3 Проектирование базы данных

На основании описания предметной области, определим сущности и связи между ними. Рассмотрим следующие сущности:

1) содержание меню;
2) информационное содержание сайта;
3) пользователи;
4) посещение определённых страниц;
5) файлы, доступные для загрузки;
6) информация о скачанных пользователями файлах;
7) раздел статистики.

Необходимо построить ER-диаграмму и её графическое представление, обозначить ограничения и правила поддержания целостности на уровне глобальной модели. Выделено семь сущностей, построим по ним ER-диаграмму и отметим кратность связи как показано на рисунке 2.3.

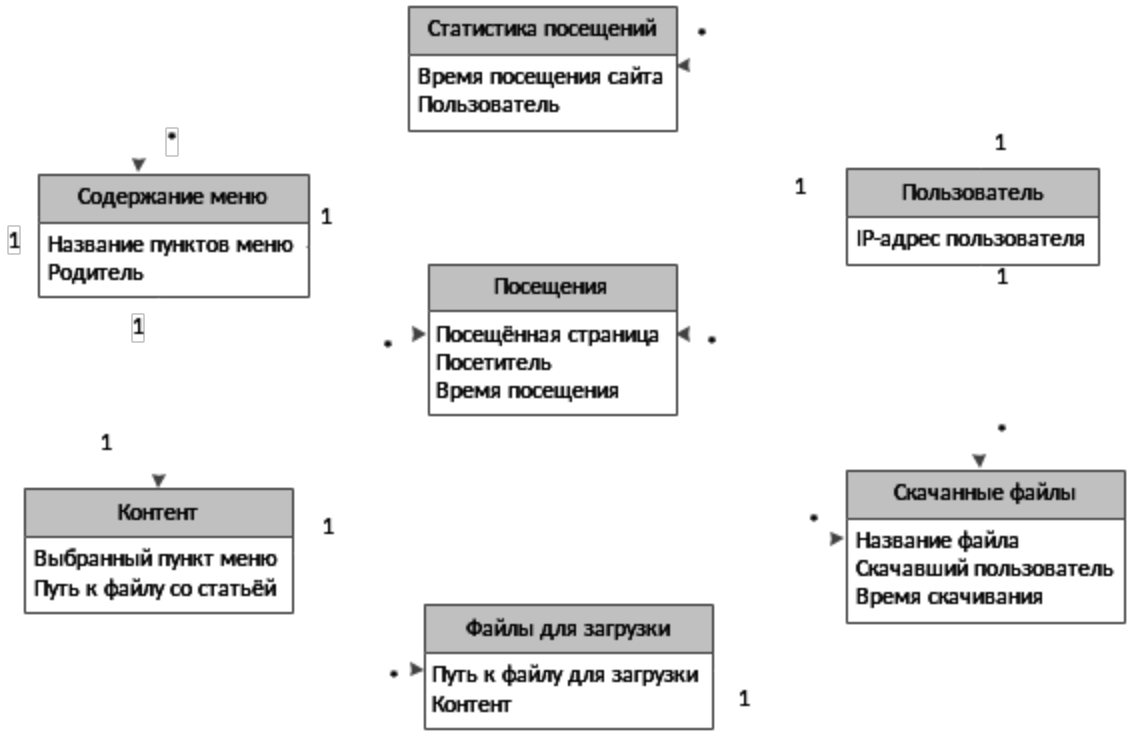

Рисунок 2.3 – ER-диаграмма

2.4 Описание базы данных

Необходимо спроектировать реляционную модель. Выполним перевод ER-модели в реляционную форму, с определением ограничения и правил поддержания целостности на реляционном уровне. Полученная реляционная база данных выглядит, как показано на рисунке 2.4.

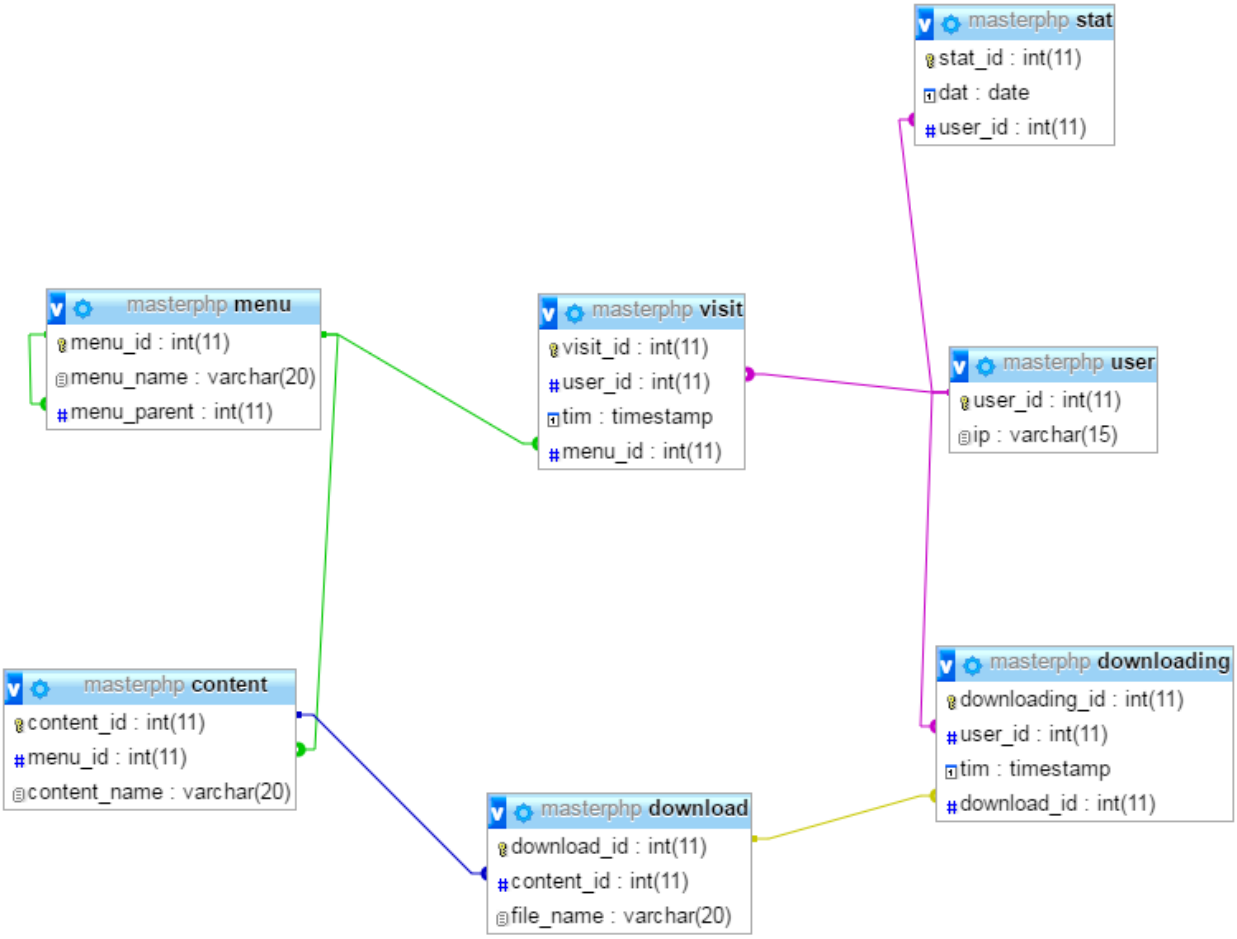

Рисунок 2.4 – Реляционная база данных masterphp

Сущности и их атрибуты отображаются в таблицах базы данных, рассмотрим их подробней:

1) menu – таблица, содержащая все пункты и подпункты меню;
2) content – всевозможные статьи, доступные для отображения;
3) user – таблица посетителей сайта;
4) visit – информация о посещённых страницах;
5) download – таблица с названием файлов, которые можно скачать;
6) downloading – статистика по скачиванию определённых файлов;
7) stat – раздел статистики.

Для полного ознакомления с используемой базой данных рассмотрим все её таблицы и их атрибуты, описание представлено в таблицах с 2.1 по 2.7.

В таблице menu (таблица 2.1) содержится информация о пунктах меню.

Таблица 2.1

Таблица menu (Пункты меню)

Имя Атрибута	Тип	Обязательность	нию Значение по	Первичный ключ	Внешний ключ	Описание
menu_id	int(11)	+	-	+	-	Идентификатор пункта меню
menu_name	varchar(20)	+	-	-	-	Название пункта меню
menu_parent	int(11)	+	-	-	+	Идентификатор родителя для пункта меню

Эта таблица состоит из трёх полей: идентификатор пункта меню, его название и идентификатор родителя для пункта меню. Таблица необходима для создания иерархического меню, с её помощью web-сценарий создаёт и отображает пункты меню в необходимом порядке, то есть выстраивает из пунктов дерево, что позволяет сделать навигацию по сайту проще и удобней.

В таблице user (таблица 2.2) содержится информация о посетителях сайта.

Таблица 2.2

Таблица user (Пользователь)

Имя Атрибута	Тип	Обязательность	Значение по умолчанию	Первичный ключ	Внешний ключ	Описание
user _id	int(11)	+	-	+	-	Идентификатор пользователя
ip	varchar(15)	+	-	-	-	IP-адрес пользователя

Эта таблица состоит из двух полей: идентификатор пользователя и его IP-адрес. Таблица необходима для регистрации статистики. При входе нового пользователя на сайт он получает собственный идентификатор и вносится в базу. С помощью web-сценария считывается его IP-адрес и проверяется, посещал ли данный пользователь сайт, если да, то заново в базу данных он не вносится, иначе – добавляется в таблицу user.

В таблице visit (таблица 2.3) содержится информация о посещённых страницах пользователями сайта.

Таблица 2.3

Таблица visit (Посещения)

Имя Атрибута	Тип	Обязательность	Значение по умолчанию	Первичный ключ	Внешний ключ	Описание
visit_id	int(11)	+	-	+	-	Идентификатор посещения
user_id	int(11)	+	-	-	+	Идентификатор пользователя
tim	timestamp	-	CURRENT_ TIMESTAMP	-	-	Время посещения
menu_id	int(11)	+	-	-	+	Идентификатор пункта меню

Эта таблица состоит из четырёх полей: идентификатор посещения, идентификатор пользователя, время посещения и идентификатор просматриваемой страницы. Таблица необходима для ведения статистики. В ней записываются каждое посещение какой-либо страница сайта любым пользователем, которые уже есть в базе. Также отмечается время посещения. Данные заносятся в таблицу с помощью web-сценария при переходе по какому-либо пункта меню.

В таблице content (таблица 2.4) содержится информационная часть сайта, т.е. названия файлов со статьями.

Эта таблица состоит из трёх полей: идентификатор контента, имя файла, идентификатор пункта меню. Таблица является главной информационной нагрузкой сайта, в ней хранятся данные о содержании сайта. С помощью web-сценария обрабатывается выбранный пункт меню и имя файла, далее рассчитывается местоположение файла, и информация из него отображается в выделенном месте на этапе вёрстки сайта. Формат файлов со статьями – htm, созданных в редакторе файлов Microsoft Word и сохраненных как «Веб-страница».

Таблица 2.4

Таблица content (Содержание)

Имя Атрибута	Тип	Обязательность	Значение по умолчанию	Первичный ключ	Внешний ключ	Описание
content _id	int(11)	+	-	+	-	Идентификатор контента
content_name	varchar(20)	+	-	-	-	Имя файла
menu_id	int(11)	+	-	-	+	Идентификатор пункта меню

В таблице download (таблица 2.5) содержится информацию о файлов доступных для скачивания.

Таблица 2.5

Таблица download (Загрузка)

Имя Атрибута	Тип	Обязательность	Значение по умолчанию	Первичный ключ	Внешний ключ	Описание
download _id	int(11)	+	-	+	-	Идентификатор загрузочного файла
file_name	varchar(20)	+	-	-	-	Имя файла
content _id	int(11)	+	-	-	+	Идентификатор контента

Эта таблица состоит из трёх полей: идентификатор загрузочного файла, имя файла и идентификатор контента. Таблица предназначена для хранения информации о всевозможных файлах, которые можно скачать на web-сайте. При нажатии кнопки «Скачать» на странице сайта, с помощью web-сценария, определяется местоположение файла, выбранного для загрузки, благодаря идентификатору контента и имени файла.

В таблице downloading (таблица 2.6) содержится информацию о скачанных пользователями файлах.

Таблица 2.6

Таблица downloading (Скачивания)

Имя Атрибута	Тип	Обязательность	Значение по умолчанию	Первичный ключ	Внешний ключ	Описание
downloading_id	int(11)	+	-	+	-	Идентификатор скачивания
user_id	int(11)	+	-	-	+	Идентификатор пользователя
tim	timestamp	+	CURRENT_ TIMESTAMP	-	-	Время скачивания
download_id	int(11)	+	-	-	+	Идентификатор скачанного файла

Эта таблица содержит четыре поля: идентификатор скачивания, идентификатор пользователя, время скачивания и идентификатор скачанного файла. Таблица необходима для ведения статистики. Она хранит все произведённые пользователями скачивания за время работы сайта. При нажатии кнопки «Скачать» на странице сайта, с помощью web-сценария, определяется идентификатор пользователя, скачивающего файл, с определённым идентификатором и эти данные записываются в таблицу. С помощью типа timestamp со стороны web-сценария нет необходимости вычислять время загрузки, оно автоматически вычисляется при занесении данных в таблицу.

В таблице stat (таблица 2.7) хранится информация о посещениях сайта пользователями.

Эта таблица содержит три поля: идентификатор входа на сайт, идентификатор пользователя и дата входа. Таблица создана для сбора статистики посещений. Запись в таблицу происходит только при загрузке сайта, при дальней работе с ним записи нет.

Таблица 2.7

Таблица stat (Статистика)

Имя Атрибута	Тип	Обязательность	Значение по умолчанию	Первичный ключ	Внешний ключ	Описание
stat_id	int(11)	+	-	+	-	Идентификатор входа на сайт
user_id	int(11)	+	-	-	+	Идентификатор пользователя
Tim	date	+	-	-	-	Дата входа

2.5 Выводы по разделу

В текущем разделе были описаны все стадии планирования web-приложения, начиная от проектирования структуры сайта, заканчивая создание базы данных, которая вошла в основу приложения.

Благодаря проведённому проектированию структуры сайта, исключены возможные некорректные отображения проекта на разных мониторах, в разных браузерах. Это позволит расширить аудиторию посетителей web-сайта.

Благодаря построенной ER-диаграмме, была спроектирована реляционная база данных, с помощью которой web-приложение позволит его владельцу создать сайт-портфолио, информационная часть которого хранится во внешних файлах, а ссылки на них будут храниться в базе. Также это позволяет контролировать посещения сайта пользователями, следить за увеличение или уменьшением спроса на определённые проекты, видеть, какие файлы вызывают интерес у посетителей.

2 РЕАЛИЗАЦИЯ WEB-ПРИЛОЖЕНИЯ

3.1 Технология Ajax-запросов.

Большинство пользователей web-сайтов не хотят наблюдать постоянную перезагрузку страниц. В настоящее время всё больше сайтов отходят от этой технологи, благодаря технологии Ajax-запросов. В создаваемом приложении используется Ajax-технология.

Технология Ajax позволяет нам обмениваться данными между браузером и сервером в фоновом режиме, т.е. без обновления страницы. Рассмотрим подробней сравнительную схему работы классических приложений и приложений с Ajax-запросами, показанную на рисунке 2.4. В браузере вводятся пользовательские данных (нажатие кнопок, ссылок, пунктов меню, и т.д.). В обычном web-приложении отправляется соответствующий HTTP запрос серверу, где полученные данные обрабатываются, и формируется ответ в виде HTML+CSS кода. В приложениях с Ajax-запросами вызывается блок JavaScript, в котором пользовательские данные отправляются на сервер методом POST или GET. Там они обрабатываются, может быть, заносятся в базу данных, при этом всё это время браузер ожидает ответа. Пока он ждёт, работа пользователя не прекращается. Дождавшись данных от сервера в формате XML или JSON, браузер обрабатывает их с помощью JavaScript и выводит в блок контента HTML-страницы.

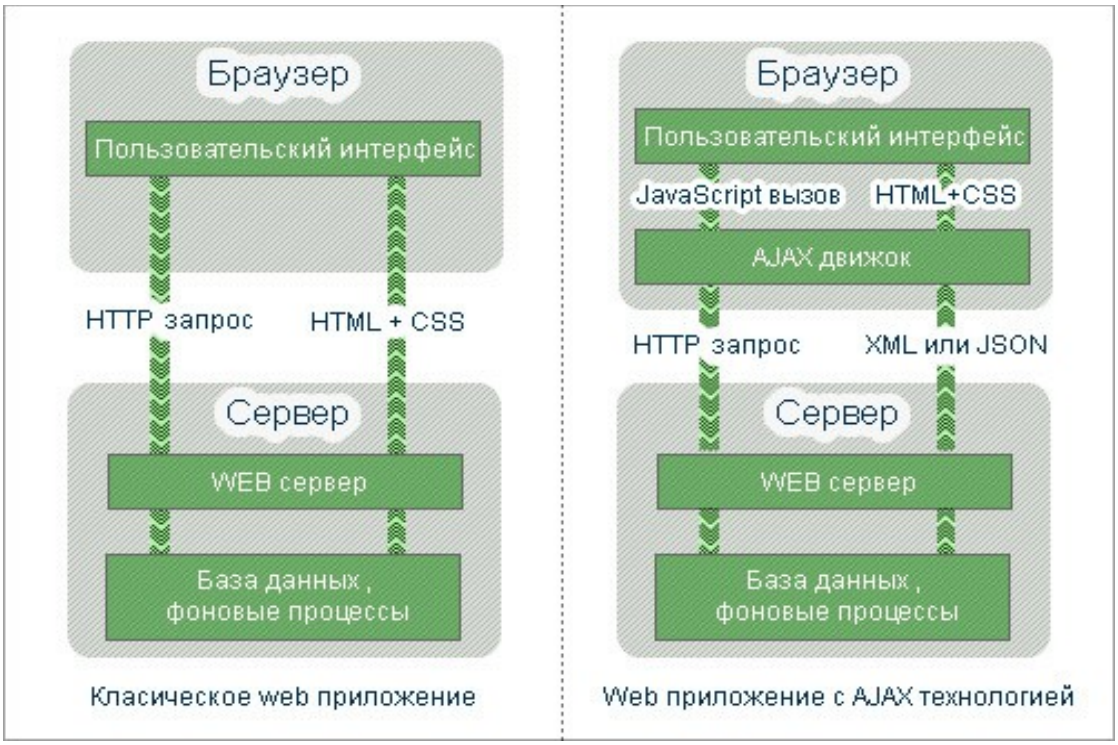

Рисунок 3.1 – Схемы работ классического web-приложения и приложения с технологией Ajax

Данные, полученные браузером от сервера, должны отвечать определённым правилам форматирования. Обычно эти правила описывают форматы XML (eXtensibleMarkupLanguage) – расширяемый язык разметки; JSON (JavaScriptObjectNotation) – текстовый формат основанный на JavaScript.

Документ типа XML имееет следующий вид:

```
<person first_name=”Григорий” last_name=”Петров”>
   <address>
         <street>пр. Ленина, 111, кв. 11</street>
         <city>Ленинград</city>
   </address>
   <phone_numbers>
         <phone_number>422-152-15</phone_number>
         <phone_number>777-264-13</phone_number>
   </phone_numbers>
</person>
```

Правила, которым должен удовлетворять документ типа XML:

1) в заголовке помещается язык разметки, номер его версии и дополнительная информация;

2) в XML учитывается регистр;

3) каждый открытый тэг должен иметь аналогичный закрывающийся;

4) значения атрибутов, определяющих тэги, должны быть заключены в кавычках;

5) учитываются все символы форматирования (пробелы, табуляции, переводы на новую строку), расположенные между начальным и конечным тэгами.

Если XML-документ удовлетворяет всем требованиям, приведённым выше, то он называется формально-правильным. Все анализаторы, разбирающие документы тип XML, работают с ним корректно.

JSON представляет собой способ хранения и передачи данных в виде некой структуры. Выглядит документ типа JSON следующим образом:

```
{
  "first_name": "Григорий",
  "last_name": "Петров",
"address": {
     "street ": "пр. Ленина, 111, кв.11",
"city": "Челябинск"
  },
"phone_numbers": [
     "422-152-15",
     "777-264-13"
  ]
}
```

Как видно из примера, в JSON могут использоваться:

1) объекты – неупорядоченное множество пар «ключ»: «значение», заключенные в фигурных скобках;

2) массивы – упорядоченное множество значений, заключенных в квадратные скобки;

3) числа;

4) строки – упорядоченное множество из символов юникода;
5) литералы – true, false, null.

Технология Ajax расшифровывается как Asynchronous Java Script and XML, из чего следует, что разработчики основным форматом считали XML. В создаваемом приложении также используется формат XML.

3.2 Применение Ajax-запросов

На технологии Ajax-запросов построена основная часть web-сайта. Она помогает обработать действия посетителя сайта. Основные этапы обработки – это изменение контента при перемещении по пунктам меню и загрузка файлов посетителями. Как уже упоминалось, эта технология осуществляет обновление контента без обновления всей страницы web-сайта, что снижает трафик. Для создания Ajax-запроса используется схема, представленная на рисунке 3.2.

Рисунок 3.2 – Схема создания Ajax-запроса

Первый шаг для всех Ajax-запросов общий – необходимо создать объект типа HTTPRequestc помощью функции createXmlHttpRequestObject() и инициализировать его с помощью функции Init(). Рассмотрим каждую функцию отдельно. Функция createXmlHttpRequestObject() пытается создать объект HTTPRequest для разных платформ, в случае неудачи выводится сообщение об ошибке:

```
function createXmlHttpRequestObject(){
var xmlHttp;
try{
    xmlHttp=new XMLHttpRequest();
}
catch(e){
    try{
            xmlHttp=new ActiveXObject("MSXML2.XMLHTTP");
    }
    catch(e){
            try{
                    xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
            }
            catch(e){}
    }
```

```
}
if(!xmlHttp){
    alert("Не удалось создать объект XMLHttpRequest");
}
return xmlHttp;
}
```

Функция инициализации Init() в общем случае создаёт объект типа HTTPRequst и назначает для него обработчик ответа сервера:

```
function Init(){
    xmlHttp=createXmlHttpRequestObject();
    xmlHttp.onreadystatechange=Receive;
}
```

На втором шаге необходимо отправить данные серверу, для чего необходимо создать функцию-отправитель, назначить входные данные, отметить способ их передачи серверу. Для каждого Ajax-запроса эта функция уникальна и создаётся с определёнными параметрами, зависимыми от смысла запроса.

На третьем шаге в работу включается web-сценарий, закреплённый за каждым Ajax-запросом отдельно. Во время его выполнения данные могут вноситься в базу, считываться оттуда, происходить расчёты, формироваться отчёт. Результатом работы web-сценария является ответ сервера, который посылается обратно пользователю.

На последнем шаге работы Ajax-запроса выполняется принятие и обработка ответа сервера с помощью responseText, responceXML и onreadystatechange. Симбиоз этих методов занесён в функцию getResponseText(xmlHttp), которая позволяет получить нам ответ от сервера. При готовности ответа она возвращает нам файл формата XML. Входным параметром является объект типа HTTPRequest, созданный нами в самом начале, выходными данными – XML файл. Описание данной функции:

```
function getResponseText(xmlHttp){
var txt=false;
if(xmlHttp.readyState == 4){
    if (xmlHttp.status == 200){
            txt=xmlHttp.responseText;
    } else {
            if (xmlHttp.status == 404){
                    alert("Request URL does not exist");
            } else {
                    alert("Error: status code is " + xmlHttp.status);
            }
    }
}
teturn txt;
}
```

Функция-обработчик для каждого Ajax-запроса уникальна, как и отправитель. Так как в созданном проекте используется два разных Ajax-запроса, то есть два варианта функций для отправления данных серверу и обработки ответа от него. Рассмотрим эти функции для каждого Ajax-запроса отдельно.

Ajax-запрос для пользования меню

Функция отправитьSend(x) представлена ниже. Входным параметром является – идентификатор выбранного пункта меню. Данная функция отправляет серверу данные методом POST и задаёт имя файла, в котором хранится описанный web-сценарий.

```
function Send(x){
    varparams="numb_file="+x;
    xmlHttp.open("POST","Login.php",true);
    xmlHttp.setRequestHeader('Content-Type','application/x-www-form-urlencoded');      //
Отправляемкодировку
    xmlHttp.send(params);
}
```

Обработчик Receive() выглядит следующим образом:

```
function Receive(){
    var txt=getResponseText(xmlHttp);
    if(!txt){
            return;
    }

    var tagList = txt.split("~@");
    var div1=document.getElementById("panel");
    var div2=document.getElementById("opis");
    var div3=document.getElementById("content");
    if  (tagList[0]==1){

            tagList[0]="Статистика";
            div3.innerHTML="";

            var div_stat1=document.createElement("div");
            div_stat1.className="chart1 ct-chart ct-golden-section";
            div_stat1.innerHTML="Количество посещений за неделю";
            div3.appendChild(div_stat1);

            var div_stat2=document.createElement("div");
            div_stat2.innerHTML="Диаграмма посещений задач";
            div_stat2.className="chart2 ct-chart ct-golden-section";
            div3.appendChild(div_stat2);

            var div_stat3=document.createElement("div");
            div_stat3.innerHTML="Диаграмма посещений курсовых";
            div_stat3.className="chart3 ct-chart ct-golden-section";
            div3.appendChild(div_stat3);
```

```
var div_stat4=document.createElement("div");
div_stat4.innerHTML='Количество загрузок из "Задач"';
div_stat4.className="chart4 ct-chart ct-golden-section";
div3.appendChild(div_stat4);

var div_stat5=document.createElement("div");
div_stat5.innerHTML='Количество загрузок из "Курсовых"';
div_stat5.className="chart5 ct-chart ct-golden-section";
div3.appendChild(div_stat5);

//массивы для  посещений
var mas_stat=tagList[1].split("?");
var count=[];
var day=[];
for(i=0; i<mas_stat.length-1;i++){
	if(i%2==0)
		count[(i)/2]=mas_stat[i];
	else
		day[(i-i%2)/2]=mas_stat[i];
}

//массивы для задач
var mas_stat=tagList[2].split("?");
var count_lang=[];
var language=[];
for(i=0; i<mas_stat.length-1;i++){
	if(i%2==0)
		count_lang[(i)/2]=mas_stat[i];
	else
		language[(i-i%2)/2]=mas_stat[i];
}

//массивы для курсовых
var mas_stat=tagList[3].split("?");
var count_lang2=[];
var language2=[];
for(i=0; i<mas_stat.length-1;i++){
	if(i%2==0)
		count_lang2[(i)/2]=mas_stat[i];
	else
		language2[(i-i%2)/2]=mas_stat[i];
}

//массивы для загрузок задач
var mas_stat=tagList[4].split("?");
var down_count1=[];
var down_lang1=[];
for(i=0; i<mas_stat.length-1;i++){
	if(i%2==0)
		down_count1[(i)/2]=mas_stat[i];
	else
```

```
                down_lang1[(i-i%2)/2]=mas_stat[i];
        }

        //массивы для загрузок курсовых
        var mas_stat=tagList[5].split("?");
        var down_count2=[];
        var down_lang2=[];
        for(i=0; i<mas_stat.length-1;i++){
            if(i%2==0)
                down_count2[(i)/2]=mas_stat[i];
            else
                down_lang2[(i-i%2)/2]=mas_stat[i];
        }

        //гистограмма для посещений
        newChartist.Bar('.chart1', {labels: day, series:[count]},
        {fullWidth: false,chartPadding: {top: 50}});

        //диаграмма для задач
        var data = {labels:language,series: count_lang};
        newChartist.Pie('.chart2', data, {  donutSolid: true,showLabel: true });

        //диаграмма для курсовых
        var data = {labels:language2, series: count_lang2};
        newChartist.Pie('.chart3', data, {  donutSolid: true,showLabel: true });

        //гистограмма для загрузок из задач
        newChartist.Bar('.chart4', {labels: down_lang1, series:[down_count1]},
        {fullWidth: false,chartPadding: {top: 50}});

        //гистограмма для загрузок из курсовых
        newChartist.Bar('.chart5', {labels: down_lang2,series:[down_count2]},
        {fullWidth: false,chartPadding: {top: 50}});
    }
    else {
        div2.innerHTML=tagList[1];
        div3.innerHTML=tagList[2];
    }

    div1.innerHTML=tagList[0];
    stopLoadingAnimation();
}
```

Коротко о сути представленной функции: ответ от сервера разбивается на части с помощью определённых символов, заданных при создании. Если ответ приходит после выбора пункта меню «Статистика» обработчик дальше разбивает на части ответ сервера. В этих частях хранятся данные о посещениях сайта, популярности страниц и количестве загруженных файлов из каждой темы. Создаются соответствующие массивы данных, и по ним строятся гистограммы и диаграммы, которые в дальнейшем будут отображаться на страницы владельца

сайта в блоке контента. Если же данные пришли после нажатия других пунктов меню, то начального разбиения ответа достаточно и полученные данные заносятся в блоки заголовка и контента.

Общая схема работы Ajax-запроса для обработки нажатий на пункты меню: при загрузке страницы в браузере создаётся объект типа HTTPRequest (функция Init()), далее при нажатии пункта меню на сервер отправляется выбранный пункт (функция Send()), выполняется web-сценарий. Браузер ожидает ответа от сервера, после получения ответа (функция getResponseText()) вызывается обработчик (функция Receive()), и браузер показывает изменения в контенте.

Ajax-запрос для загрузки

При отображении статей в блоке контента могут появиться кнопки, предназначенные для загрузки файлов исходного кода соответствующих задач. Для ведения статистики скачиваний применяется Ajax-технология. Статистика ведётся на стороне сервера, поэтому обработчика ответа от него в конкретно этом случае нам не понадобится. Тогда в функции инициализации остаётся только создание HTTPRequest-объекта:

```
function d_Init(){
    xmlHttp_d=createXmlHttpRequestObject();
}
```

Как уже говорилось, функция для отправки запроса серверу уникальна. Для Ajax-запроса, созданного при загрузке файлов, она выглядит следующим образом:

```
function Send_download(x,y){
    var params="numb_file="+x+"&id="+y;
    xmlHttp_d.open("POST","download_ajax.php",true);
    xmlHttp_d.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded'); // Отправляем кодировку
    xmlHttp_d.send(params);
}
```

Представленная функция передаёт такие параметры как идентификаторы страницы и скачанного файла. Далее отправляет их серверу, где происходит дальнейшая работа с ними.

3.3 Web-сценарий

После передачи параметров серверу, с помощью Ajax-технологию, в работу вступает web-сценарий, описанный заранее. Web-сценарий обрабатывает полученные параметры, совершает действия с ним, быть может заносит в базу данных и выдаёт ответ в формате XML. Для работы сайты было создано два web-сценария: обработка нажатий пунктов меню и ведение статистики скачиваний.

Второй сценарий является тривиальным и представляет собой пару SQL запросов и занесение информации о скачивании в базу данных. Подробнее стоит разобрать обработку нажатий пунктов меню, для этого рассмотрим алгоритм работы этого web-сценария, показанный на рисунках 3.3 и 3.4.

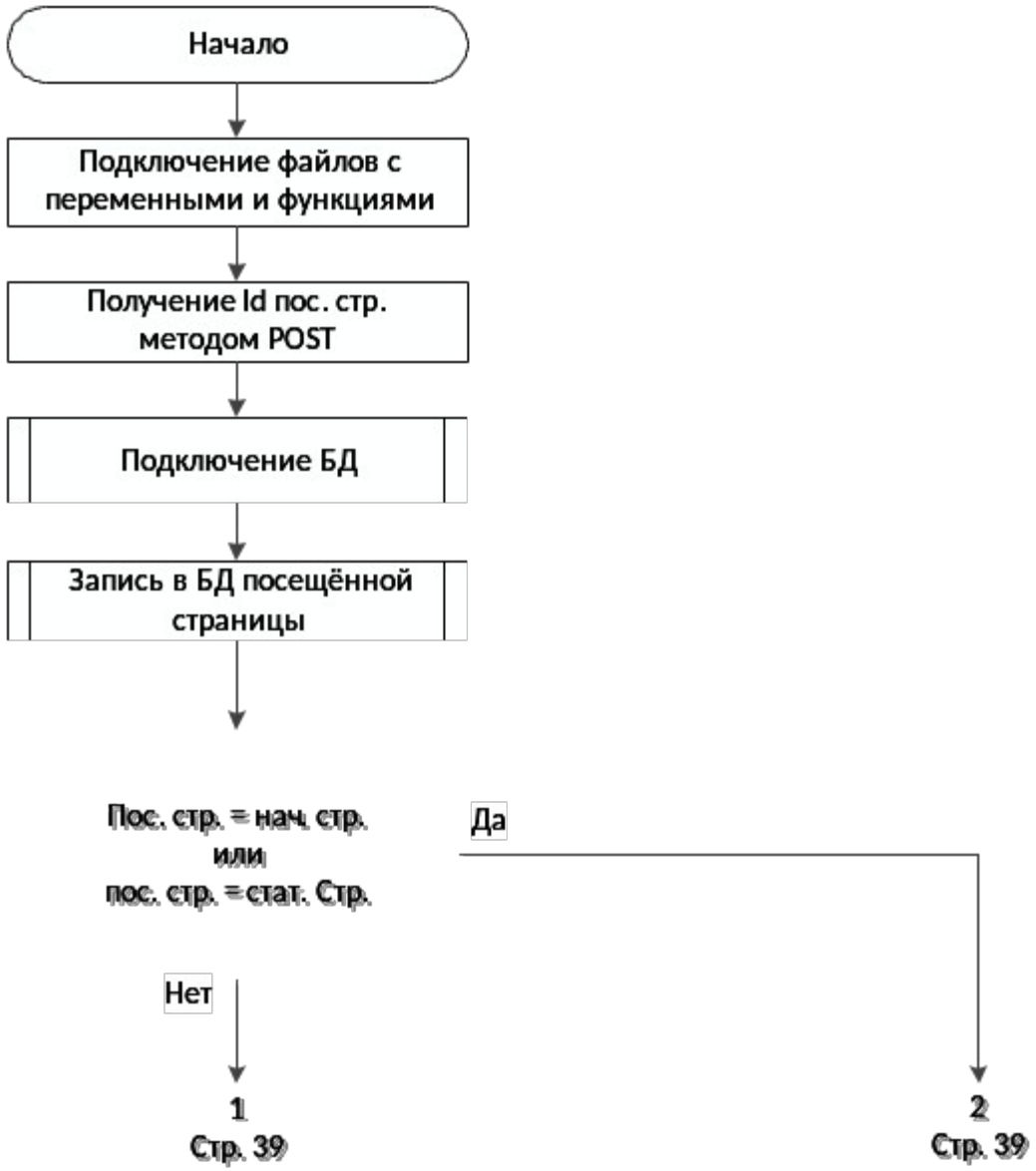

Рисунок 3.3 – Алгоритм работы web-сервера (1)

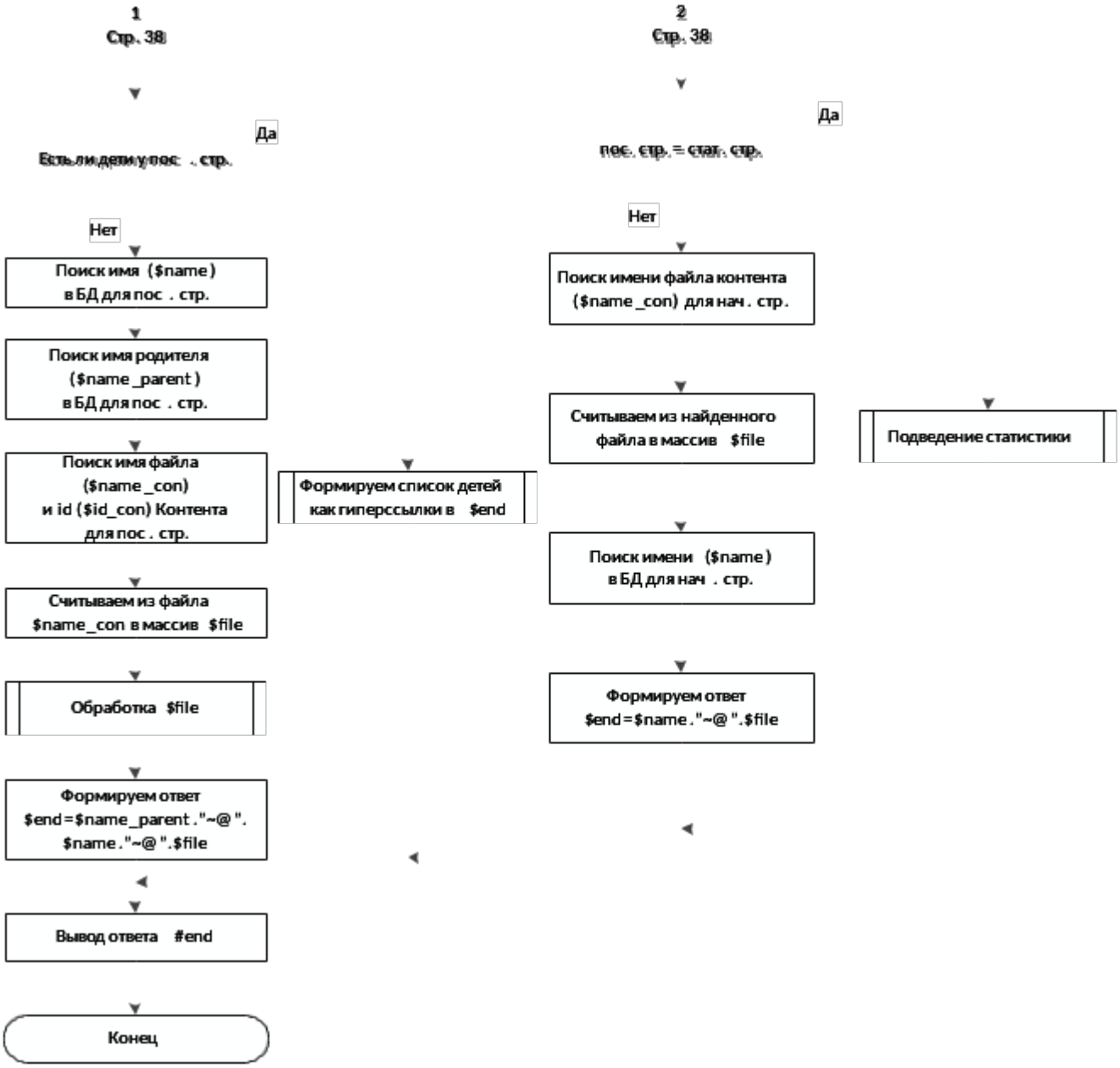

Рисунок 3.4 – Алгоритм работы web-сервера (2)

Статистика ведётся по количеству посещений сайта разными пользователями за период времени, рассчитанный в днях, разделов «Задачи» и «Курсовые» и по количеству скачанных файлах в этих же разделах. Для отображения гистограмм и диаграмм необходимо получить два массива, в одном хранятся количественные данные, в другом – их соответствующие описания. Рассмотрим работу функции на примере подсчёта статистики посещений за неделю, как показано на рисунке 3.5. Для остальных блоков алгоритм остаётся без изменений, а меняется посылаемый базе данных SQL запрос. Получаем, что через разделитель выводятся по два массива, разделённые между собой символом «?», которые соответствуют определённому блоку. Полученные данные обрабатываются с помощью JavaScript в функции-обработчике Ajax-запроса.

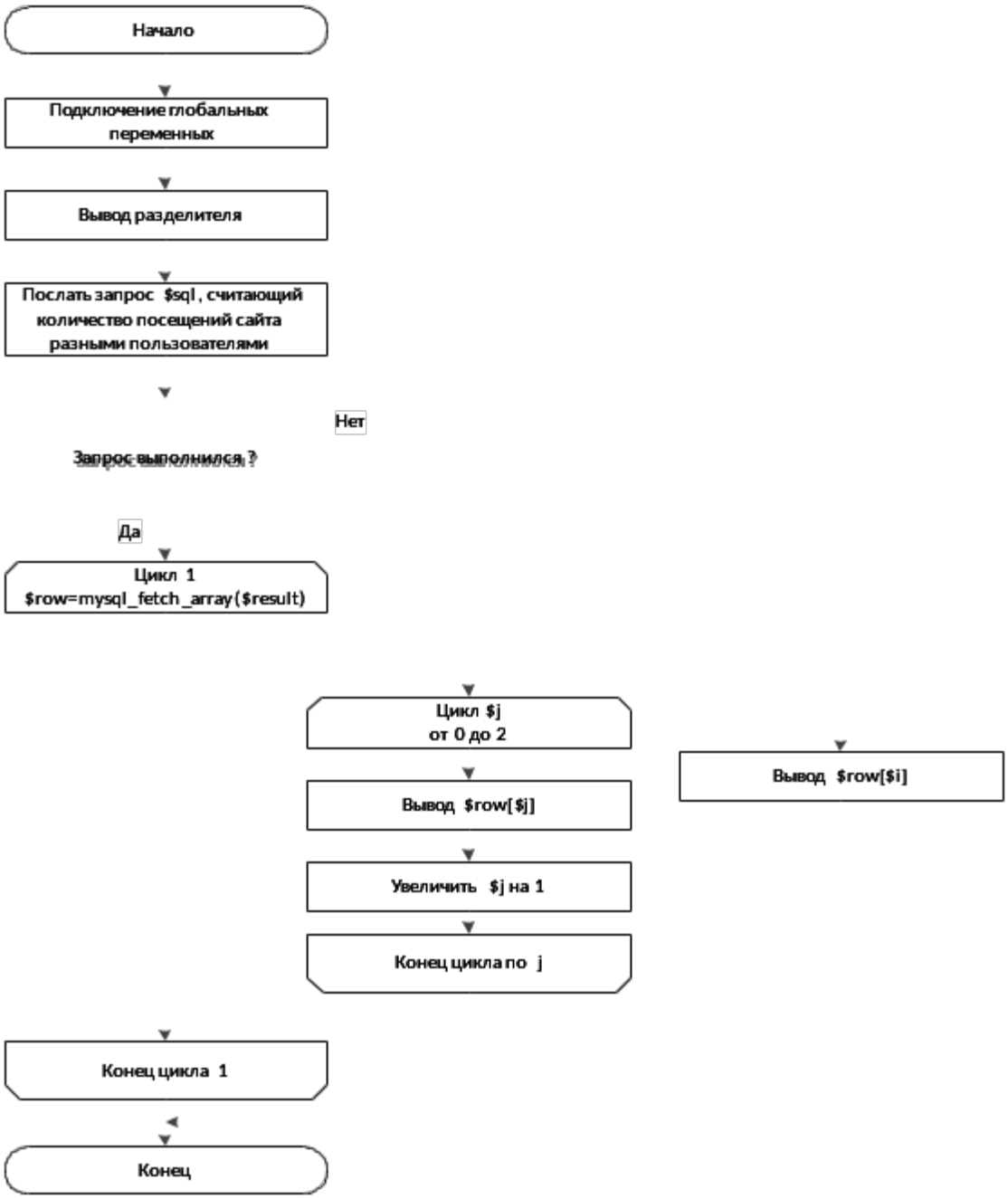

Рисунок 3.5 – Алгоритм ведения статистики

При считывании файла с сервера необходима некоторая доработка для корректного отображения в блоке контента сайта. Для этого файл считывается в переменную $file, дальше происходят действия, описанные в алгоритме на рисунке 3.6 и 3.7.

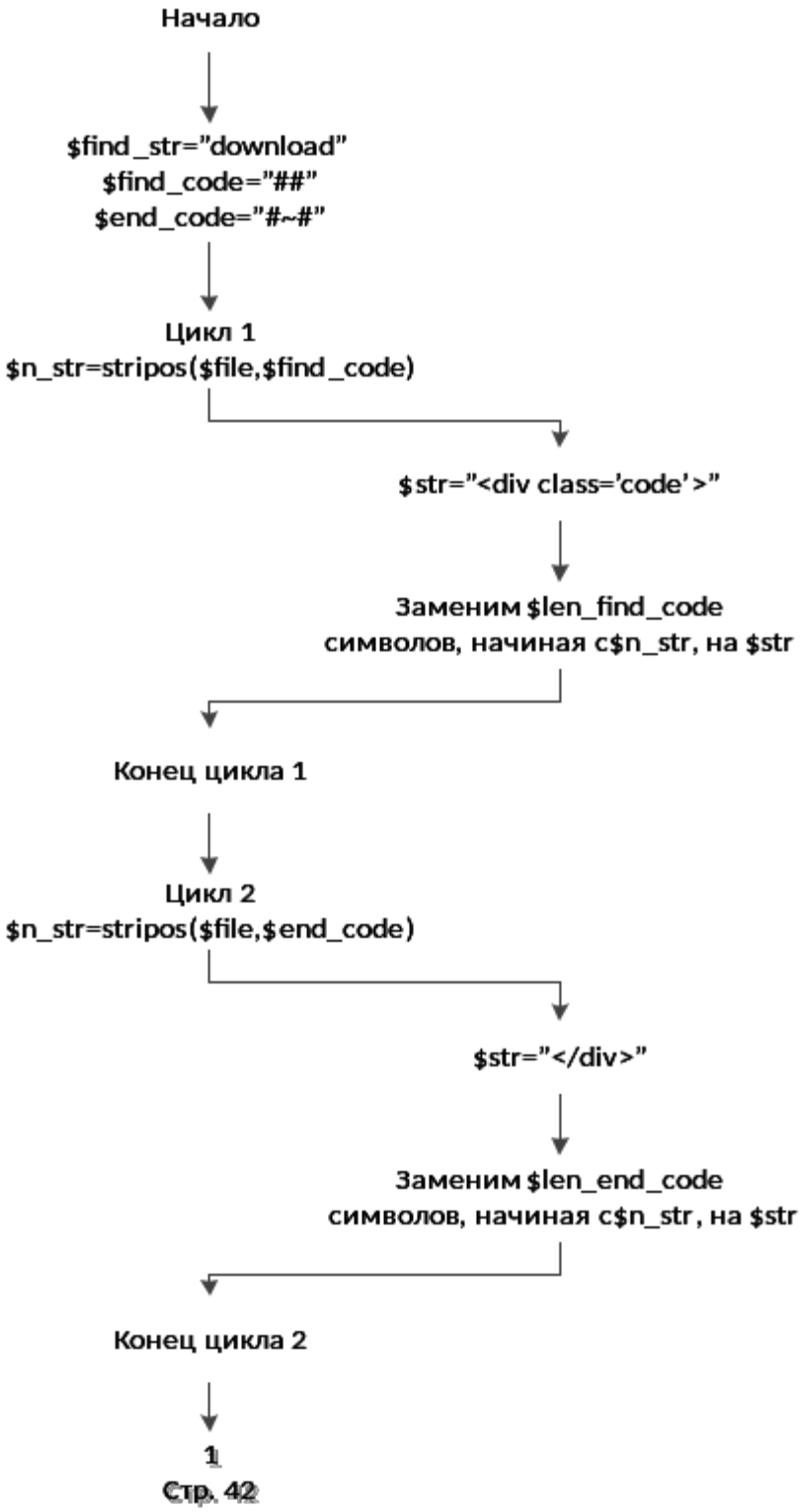

Рисунок 3.6 – Алгоритм обработки файла (1)

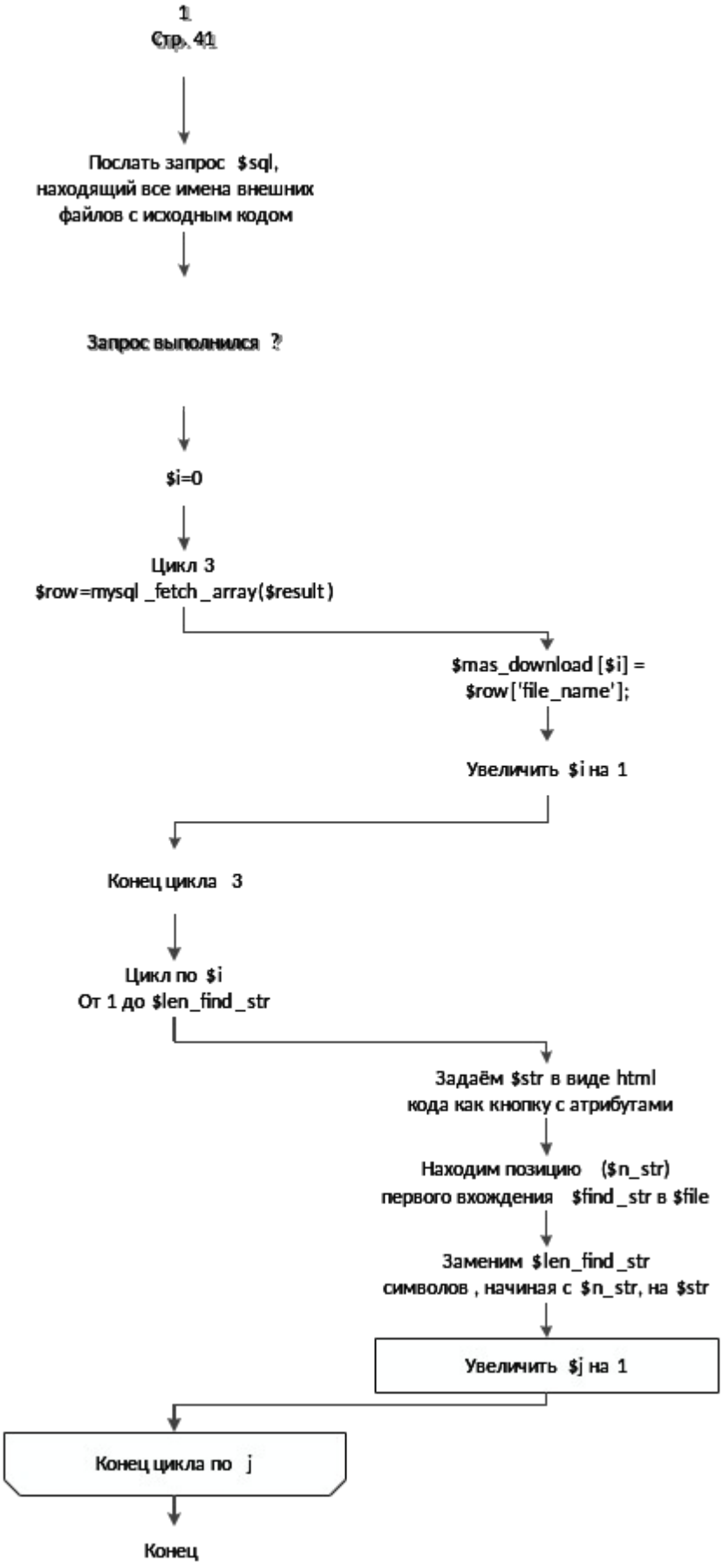

Рисунок 3.7 – Алгоритм обработки файла (2)

3.4 Разработка инструментальных средств

При создании разного рода приложений для упрощения и ускорения процесса написания используются функции и классы. Без них не обошлось и в созданном проекте. Рассмотрим основные функции, которые использовались при написании web-приложения.

Web-сайт работает с базой данных, и чтобы не прописывать каждый раз соединение с ней была создана функция db().

```
function db() //подключение к Базе данных
{
    include 'config.php';
    $db=@mysql_connect($hostname_connect, $username_connect, $password_connect);
    if (!$db) exit('Ошибка при подключении к базе данных');
    if(!mysql_select_db($database_connect,$db)) exit('База данных не существует');
    mysql_query('SET NAMES utf8');
}
```

В которой переменные $hostname_connect, $username_connect, $password_connect отвечают за имя хоста, пользователя и пароль пользователя соответственно и $database_connect – имя базы данных. Эти переменных хранятся в файле с настройками config.php.

Для создания динамического меню навигации по сайту были созданы функции get_cat() и view_cat($arr,$parent_id = null). Рассмотрим подробней функцию get_cat() .

```
function get_cat()  //Возвращает ассоциативный массив
{
    include "config.php";
    //запрос к базе данных
    if(get_ip()==$admin_ip)
        $sql = "SELECT * FROM menu order by menu_name";
    else
        sql = "SELECT * FROM menu where menu_id <> '$stat_id' order by menu_name";
   $result = mysql_query($sql);
    if(!$result)    return NULL;
    $arr_cat = array();
    if(mysql_num_rows($result) != 0){
        //В цикле формируем массив
        for($i = 0; $i<mysql_num_rows($result);$i++) {
            $row = mysql_fetch_array($result,MYSQL_ASSOC);
            if(empty($arr_cat[$row['menu_parent']])){
                $arr_cat[$row['menu_parent']] = array();
            }
            $arr_cat[$row['menu_parent']][] = $row;
        }
        //возвращаем массив
```

```
            return $arr_cat;
        }
    }
```

Данная функция возвращает ассоциативный массив, получаемый из базы данных из таблицы menu. В этом массиве хранятся идентификаторы пунктов меню, родительские идентификаторы и сами названия пунктов, отсортированные по алфавиту. Пример ассоциативного массива:

```
Array
(
    [0] => RED
    [1] => BLUE
    [2] => GREEN
    [3] => YELLOW
)
```

Рассмотрим подробней функцию view_cat($arr,$parent_id = null).

```
function view_cat($arr,$parent_id = null) //Преобразовывает ассоциативный массив в список на html
    {
        //Условия выхода из рекурсии
        if(empty($arr[$parent_id]))          return;
        echo '<ul id="menu">';
        //перебираем в цикле массив и выводим на экран
        for($i = 0; $i< count($arr[$parent_id]);$i++){
                echo '<li  style="font: 13pt sans-serif; align: center;"><a href="#'.$parent_id.'/'.$arr[$parent_id][$i]['menu_id'].'">'.$arr[$parent_id][$i]['menu_name'].'</a>';
                //рекурсия - проверяем нет ли дочерних категорий
                view_cat($arr,$arr[$parent_id][$i]['menu_id']);
                echo '</li>';
        }
        echo '</ul>';
    }
```

На вход функции подаётся ассоциативный массив и для рекурсивного использования идентификатор родителя пункта меню, для первого прохода он равен null. На выходе из функции мы получаем список на языке разметки HTML. Под списком понимается конструкция вида:

```
<ul>
    <li>Первый пункт</li>
    <li>Второй пункт</li>
    <li>Третий пункт</li>
</ul>
```

Для ведения статистики и распознавания пользователей нужно узнавать их IP-адрес при входе на сайт. Для этого создаём функцию get_ip().

```
function get_ip() // возвращаетIP
```

```
{
    if (!empty($_SERVER['HTTP_CLIENT_IP'])){
            $ip=$_SERVER['HTTP_CLIENT_IP'];
    }
    else if (!empty($_SERVER['HTTP_X_FORWARDED_FOR'])){
            $ip=$_SERVER['HTTP_X_FORWARDED_FOR'];
    }
    else{
            $ip=$_SERVER['REMOTE_ADDR'];
    }
    return $ip;
}
```

Посещение сайта каждым новым пользователем также заносится в статистику, выполняется это с помощью функции ins_ip_db(). Которая подключается к базе и вычисляется IP-адрес пользователя уже определёнными функциями.

```
function ins_ip_db(){
    db();
    $ip=get_ip();
    $sql="SELECT user_id FROM user WHERE ip='$ip'";
    $result = mysql_query($sql);
    if(!$result or !mysql_num_rows($result)) {
            $sql2=mysql_query("insert into user(ip) value('$ip')");
                $result2 = mysql_query($sql) or die(mysql_error());
    }
}
```

3.5 Выводы по разделу

В текущем разделе были рассмотрены основные приёмы, использованные при создании приложения. Благодаря Ajax-запросам мы снизили нагрузку на посетителей сайта, уменьшили затрачиваемый трафик. Также с помощью этой технологии web-сайт стал динамическим, то есть информационная часть не является постоянной, а меняет при использовании меню, в то время как остальные блоки остаются неизменными. Раздел содержит и описание работы web-сценариев, которую не видно посетителям сайта. Благодаря совместной работе Ajax-технологии и web-сценариев создана обработка перемещений по пунктам меню, то есть отображение контента в информационном блоке, а так же нажатия клавиш скачивания, после которых происходит запись в базу данных. Так же был описан и созданные инструментальные средства, позволяющие сократить написание программного кода, при повторных использованиях.

3 ОПИСАНИЕ РАБОТЫ С WEB-ПРИЛОЖЕНИЕМ

Существует два варианта пользователей созданного проекта:
1) посетители сайта;
2) владельцы сайта, созданного на основе разработанного web-приложения.
Опишем руководство использования созданного приложения для обеих категорий пользователей.

4.1 Руководство пользователя сайта

Как и на многих других сайтах, перемещение осуществлено с помощью меню, показанного на рисунке 4.1.

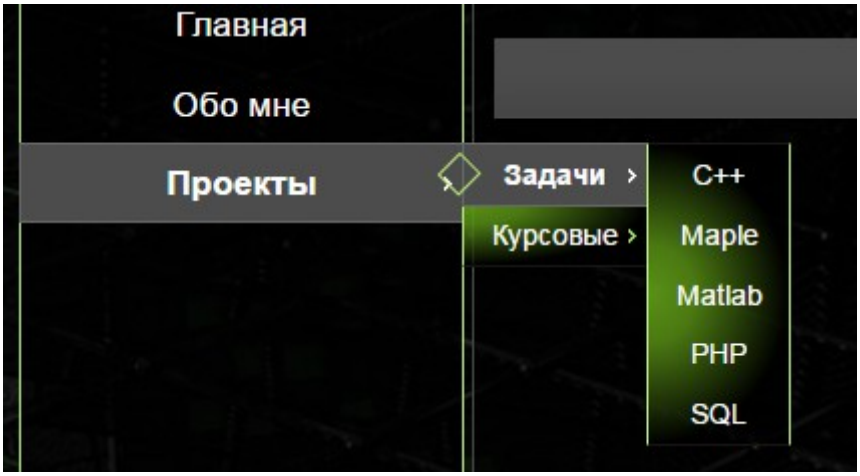

Рисунок 4.1 – Меню сайта

Представленное меню является иерархическим, т.е. при наведении на «Проекты» выпадает два дальнейших варианта перехода «Задачи» и «Курсовые». При наведении на «Задачи» происходит дальнейший разворот меню с вариантами следующего выбора. Пункты меню, которые являются выпадающими, отмечены сбоку зелёной стрелочкой, а при наведении выделяются цветом и разворачиваются.

На сайте присутствует обычный текст и текст в специальных блоках, который является исходным кодом для решения задачи, описанной выше на странице сайта, например рисунок 4.2. В данном блоке присутствует скроллинг, для сокращения места программного кода на странице.

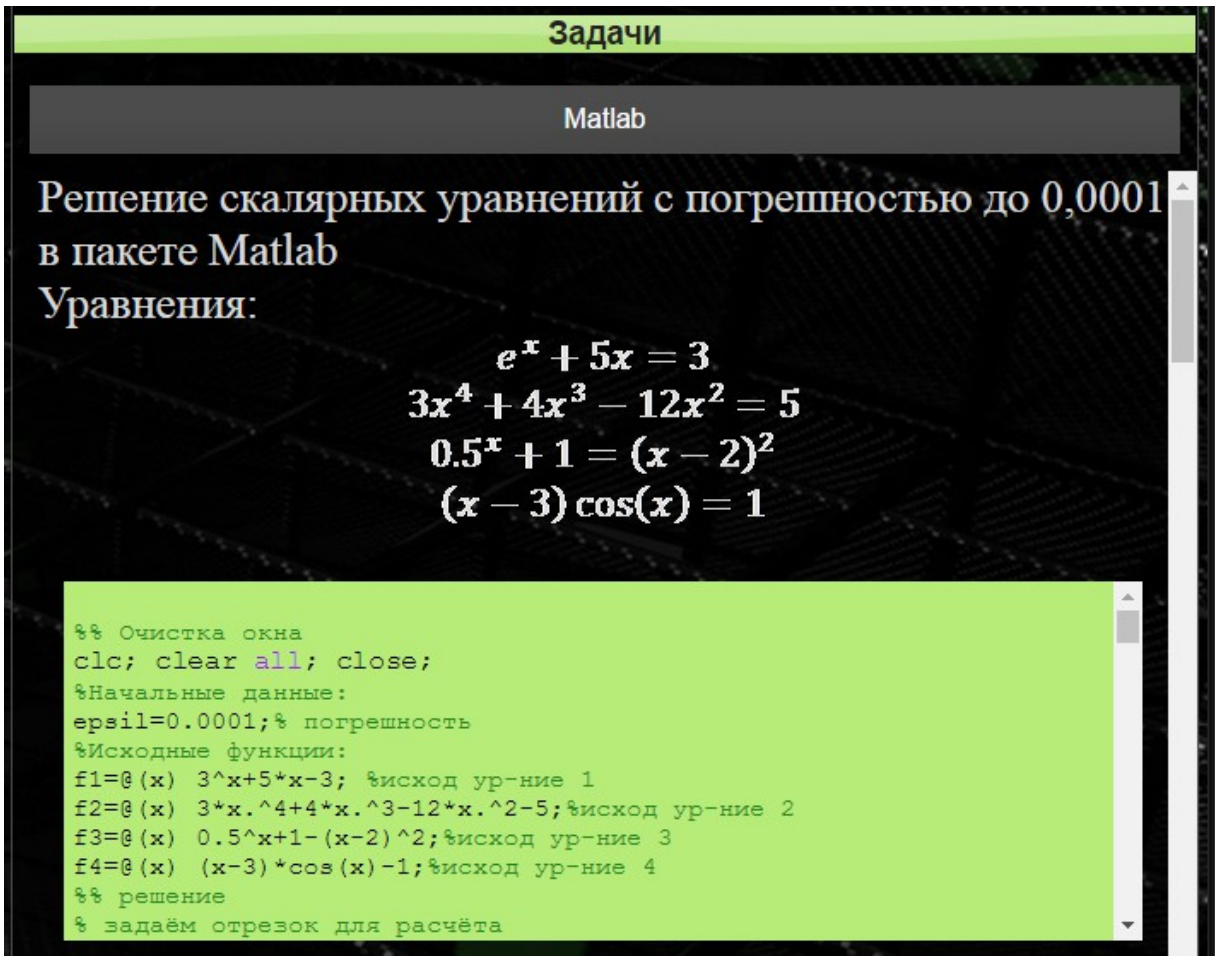

Рисунок 4.2 – Контент сайта

В большинстве случаев языком программирования в таких блоках является выбранный в меню пункт, но бывают случаи, когда исходный код является комбинацией нескольких программных языков. Так же существуют задачи, решение которых хранится в нескольких файлах, например в одном файле определяется интерфейс класса, а другом реализация методов. Чтобы сократить время на создание нескольких файлов и согласование их между собой, на сайте присутствует возможность одновременного скачивания всех файлов, необходимых для решения задачи. Файлы находятся в архиве с расширение zip, который можно разархивировать в любой операционной системе. Для скачивания файла необходимо нажать на кнопку «Скачать» (рисунок 4.3) и выбрать директорию для размещения архива.

Рисунок 4.3 – Возможность скачивания файла

4.2 Руководство владельца сайта

Для создания сайта, основанного на разработанном проекте, на локальном хостинге необходимо запустить web-сервер и сервер базы данных. При создании этого приложения был использован XAMPP – кроссплатформенная сборка web-сервера, содержащая web-сервер Apache, MySQL – сервер баз данных, интерпретатор скриптов PHP.

Рассмотрим установку проекта при использовании XAMPP. Для начала необходимо запустить сервер и подключить модули Apache и MySQL, как показано на рисунке 4.4.

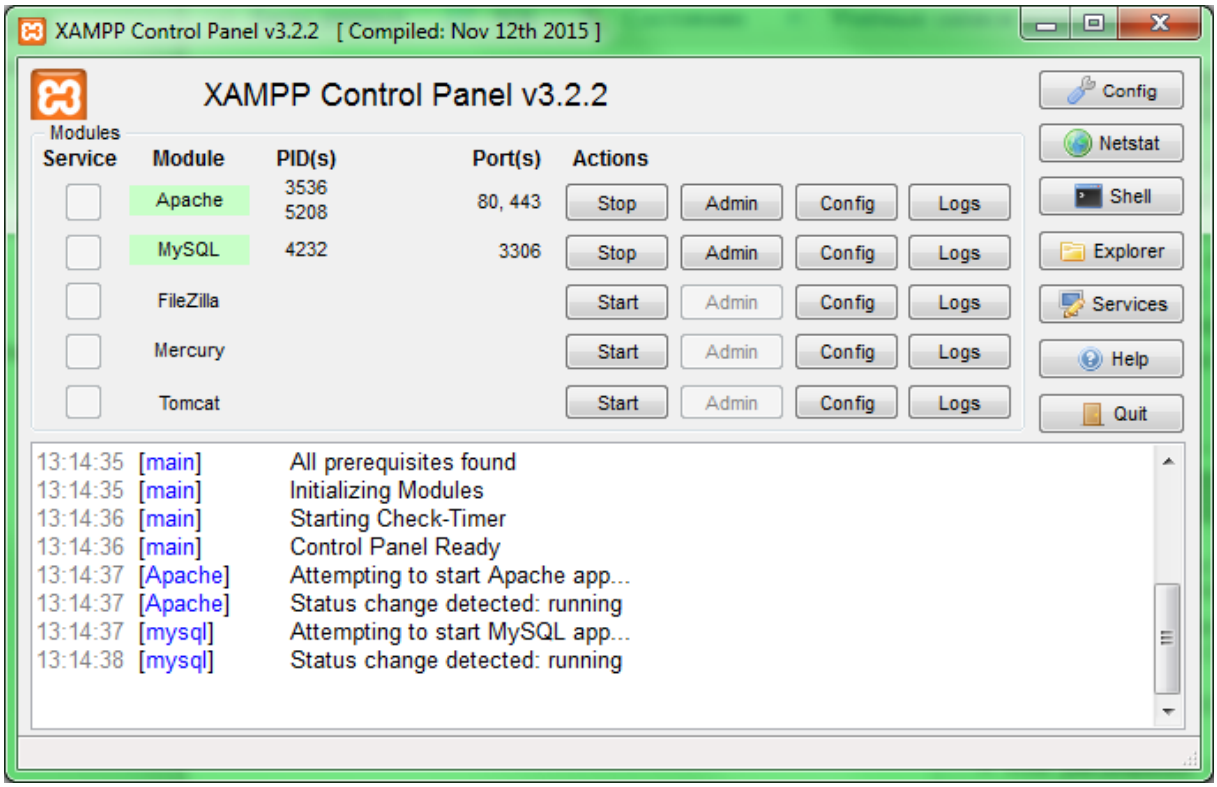

Рисунок 4.4 – Запуск XAMPP

Все файлы созданного приложения поместить в папку, расположение которой C:\xampp\htdocs\mysite. Теперь при вводе в поисковую строку браузера имени «localhost», разметка сайта и оформления будет видно, но информационного наполнения нет, т.к. необходимо ещё импортировать базу данных на сервер.

Для установки базы данных на сервер необходимо воспользоваться приложением phpMyAdmin, которое доступно в браузере по ссылке http://localhost/phpmyadmin, и имеет вид как на рисунке 4.5.

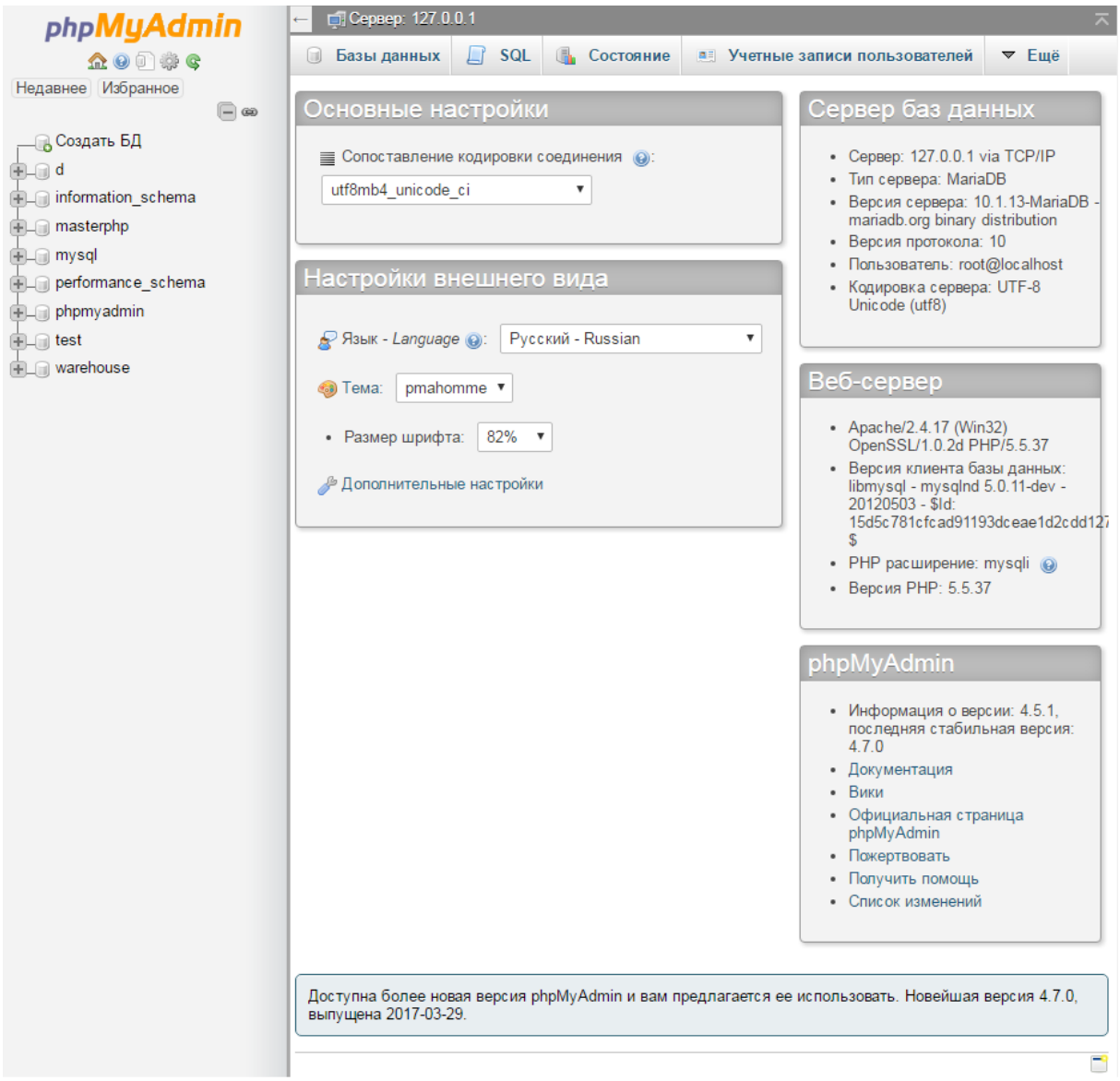

Рисунок 4.5 – Приложение phpMyAdmin

Далее необходимо выбрать вкладку «Ещё» - «Импорт». В появившемся окне (рисунок 4.6) выбрать импортируемый файл, установить кодировку (для корректного отображения кириллицы используется utf-8), настроить формат базы данных – SQL и нажать кнопку «Вперёд». После чего выбранная для импорта база данных будет отображаться в списке всех баз данных, находящемся в левой части страницы.

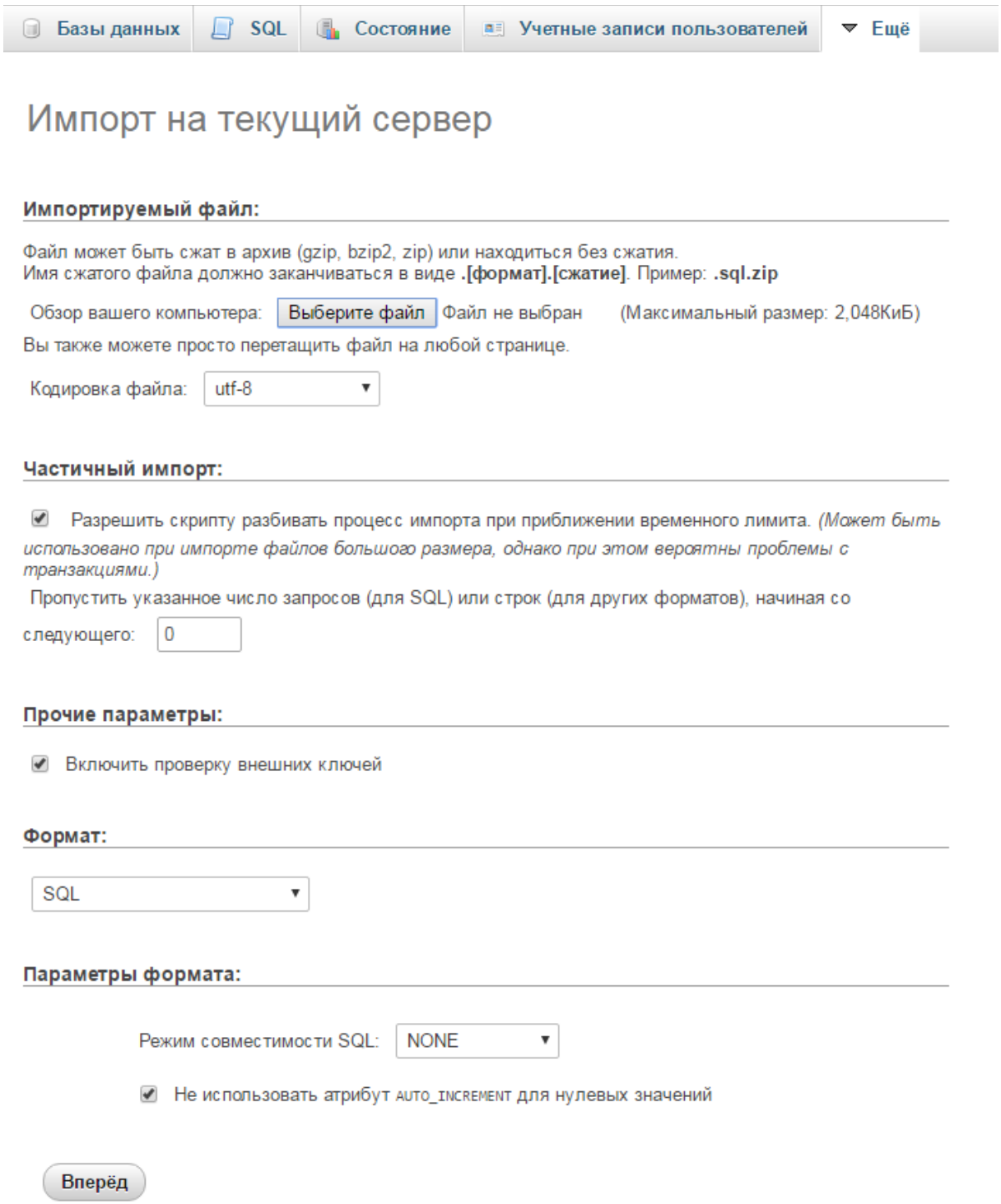

Рисунок 4.6 – Импорт базы данных

Для редактирования, изменения или мониторинга данных таблиц необходимо нажать на интересующую базу данных в списке всех баз. Появится окно с таблицами (рисунок 4.7), входящими в базу, где приведён список действий, которые можно совершать с таблицами. Также показано количество строк, содержащихся в таблице, её тип, кодировка и размер. Описание всех таблицы, созданных в разработанном web-приложении, было изложено ранее в главе 2.

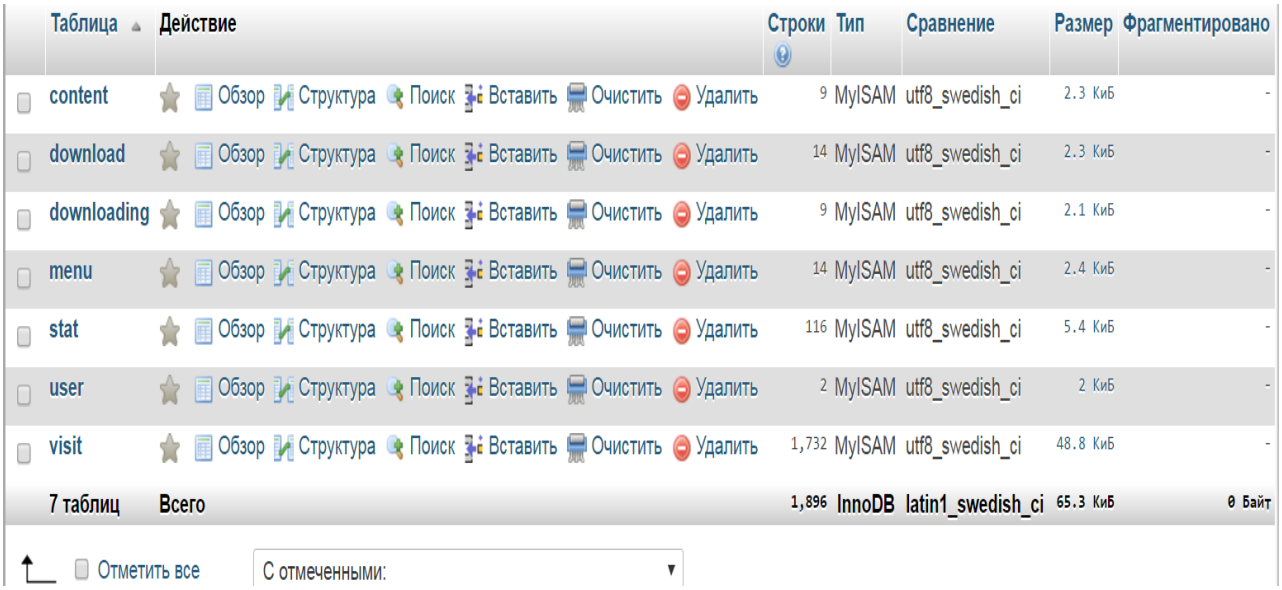

Рисунок 4.71 – Список таблиц в базе данных

Общий вид работы с таблицами в phpMyAdmin показан на рисунке 4.8.

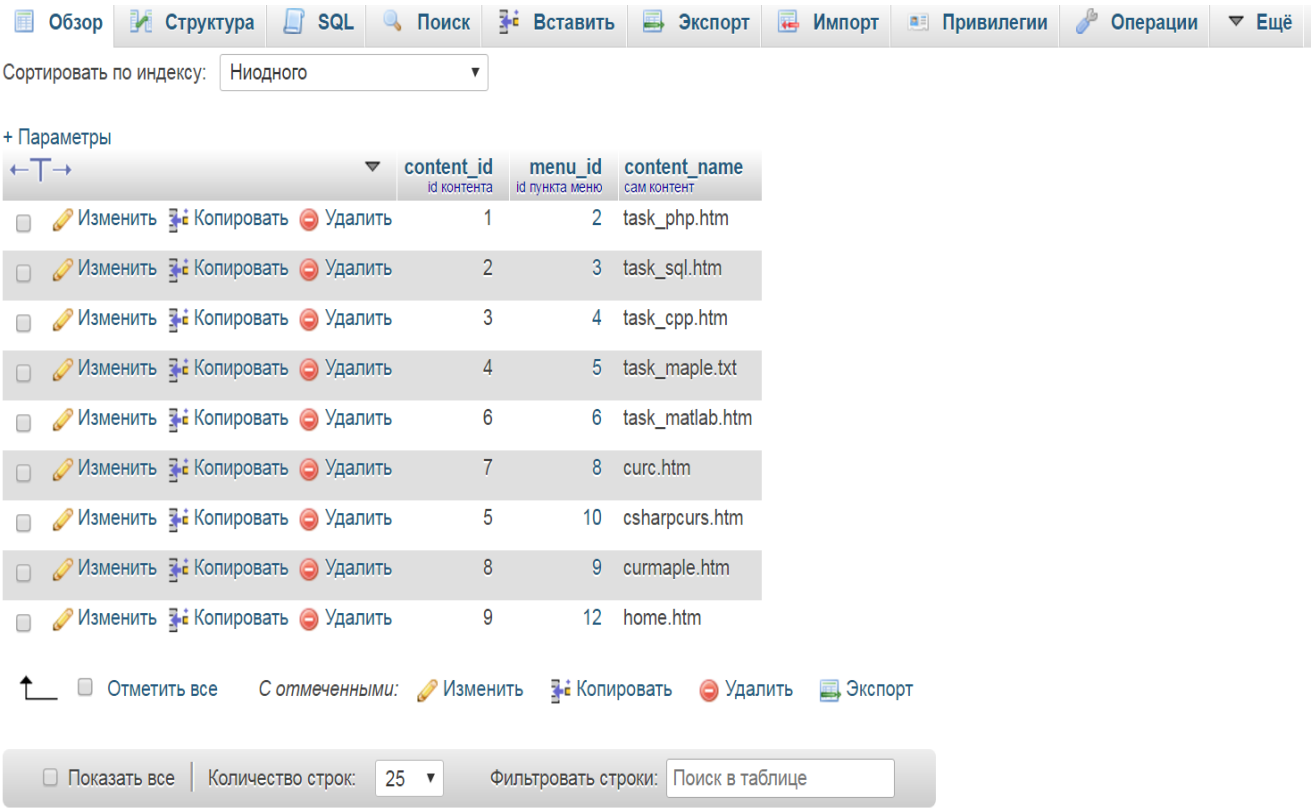

Рисунок 4.8 – Работа с таблицами

Рассмотрим работу с таблицами на примере заполнения контента. Для изменения информационной нагрузки сайта, необходимо выбрать таблицу content и внести собственные корректировки. Как видно из предыдущего рисунка, для строк таблицы существует три действия: изменения, копирование и удаление. При нажатии на вкладку «Структура» (рисунок 4.9) можно добавлять поля, менять их название, тип, сравнение, атрибуты, возможность принимать значение NULL, добавлять значения по умолчанию и дополнительные параметры.

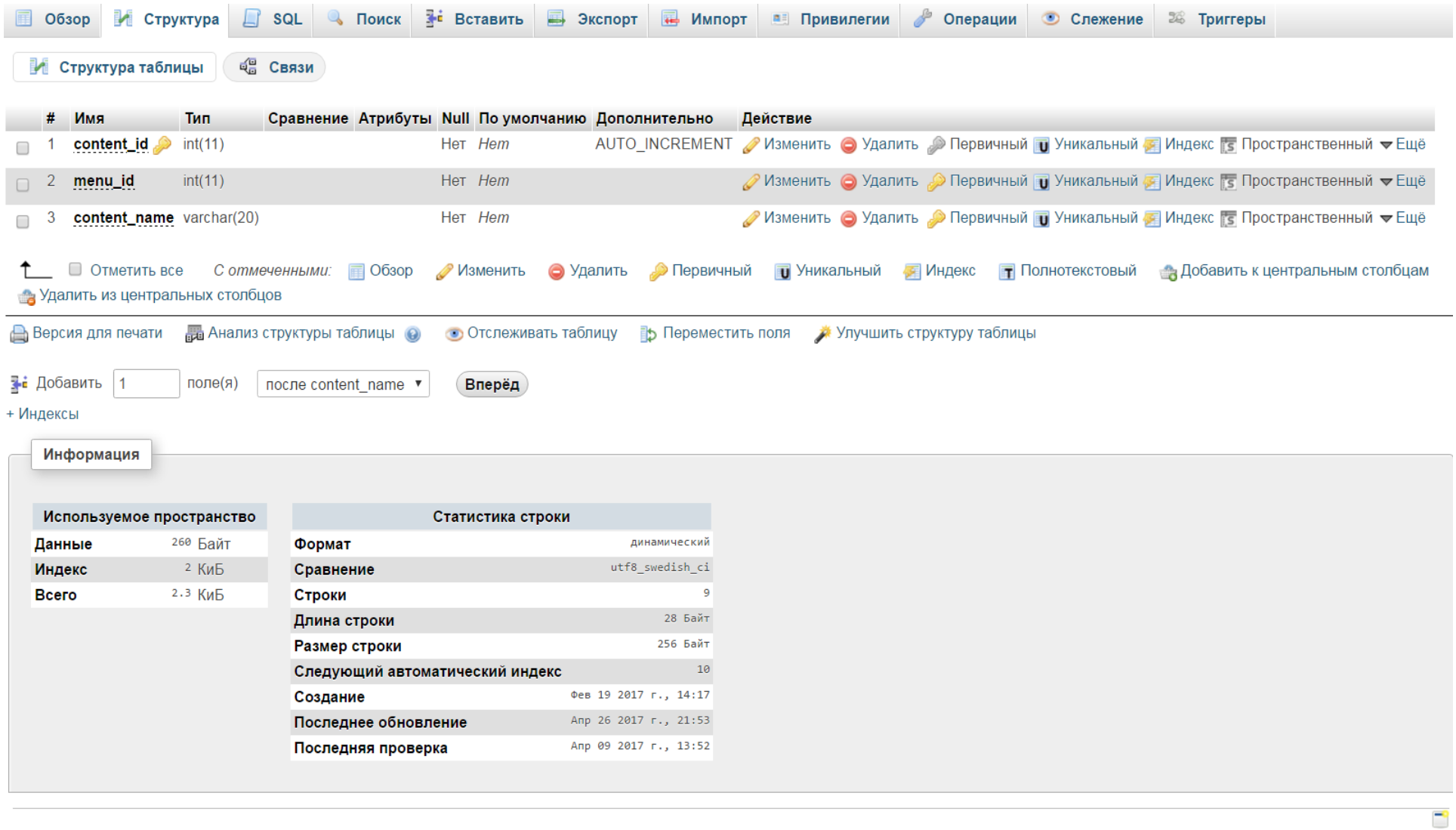

Рисунок 4.9 – Вкладка «Структура»

В дополнительной вкладке «Связи» можно изменить внутренние связи полей с другими таблицами и полями, это показано на рисунке 4.10.

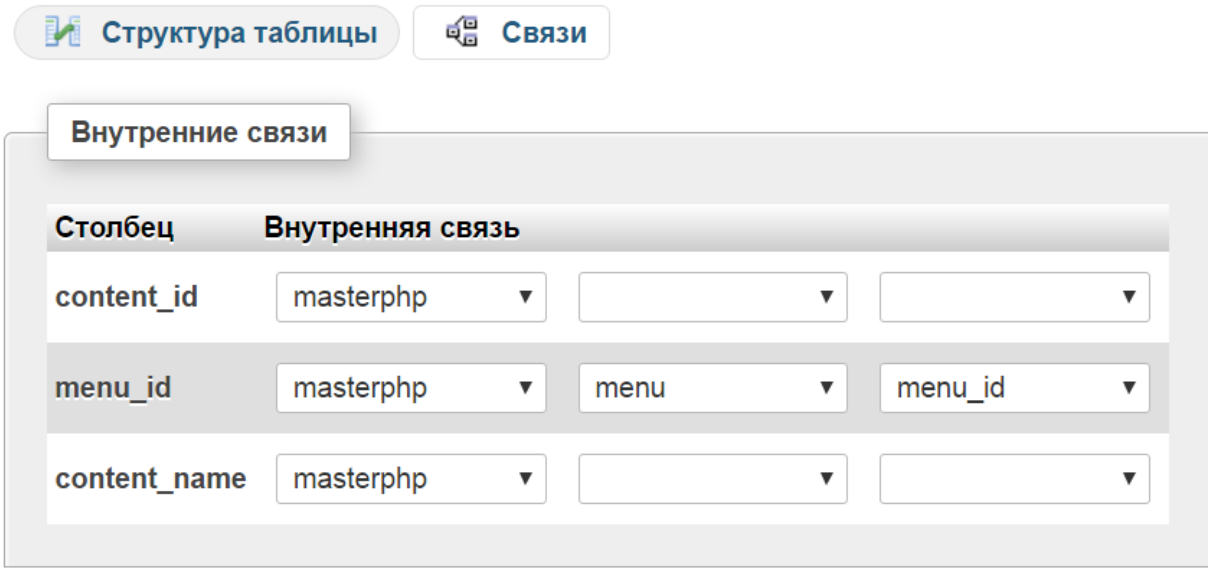

Рисунок 4.10 – Изменение связей полей

Во вкладке SQL можно создавать запросы к базе данных, что может быть удобней для тестирования, чем писать и выводить их через web-сценарии. Структура этой вкладки показана на рисунке 4.11.

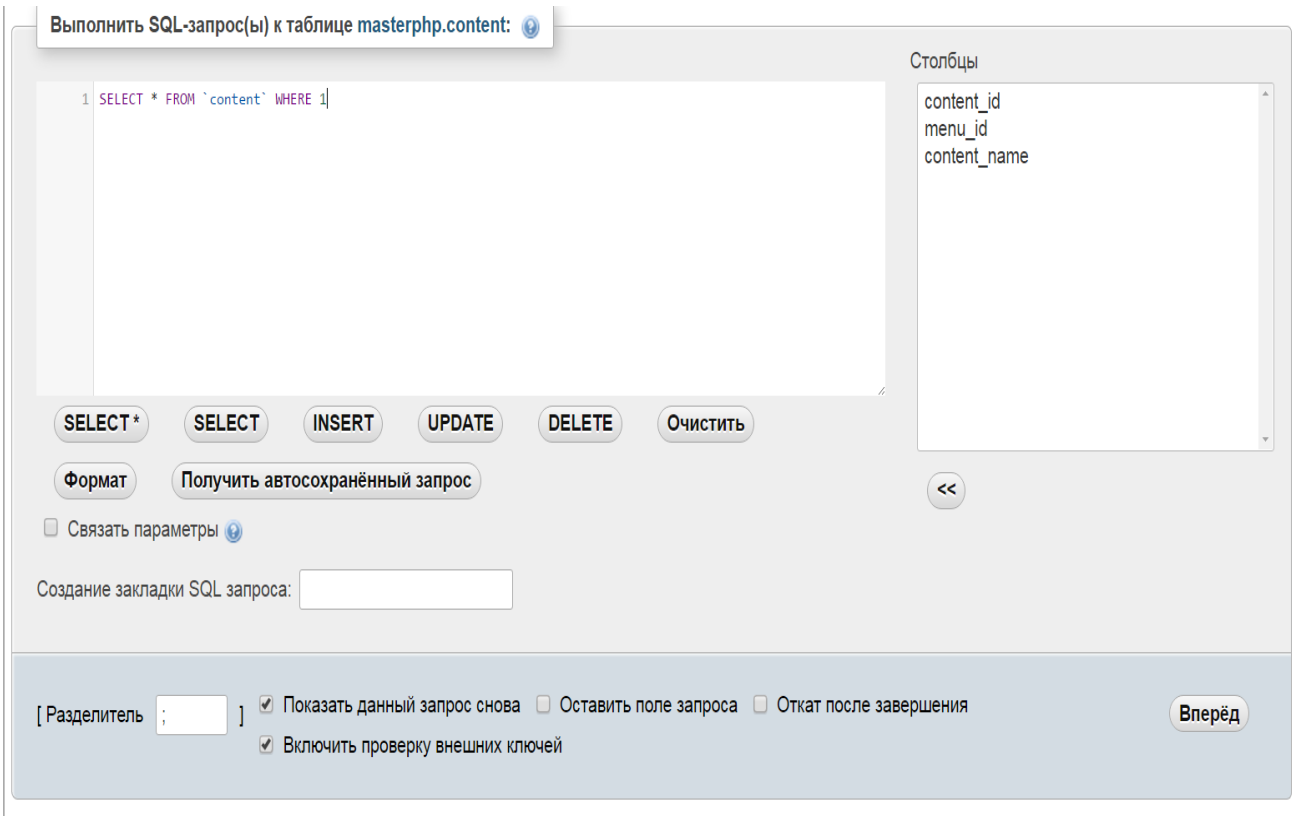

Рисунок 4.11 – Выполнение SQL запросов через phpMyAdmin

Во вкладке «Поиск» присутствует возможность поиска в таблице, поиска с приближение и «найти и и заменить». При создании приложения, данный раздел не понадобился. А вот следующий пункт «Вставить» использвался часто. Он позволяет добавить строку в таблицу, выглядит это как на рисунке 4.12.

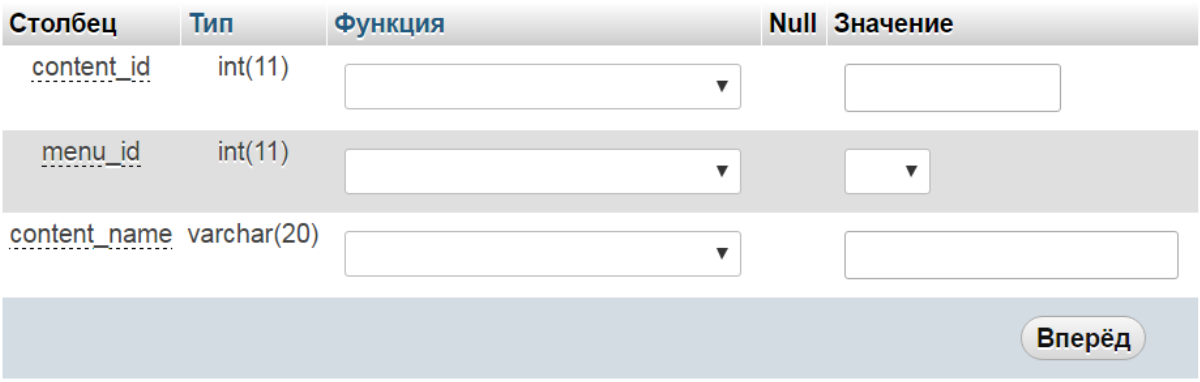

Рисунок 4.12– Вставка строки в таблицу

Вкладки «Экспорт» и «Импорт» позволяют получать данные из таблицы и заносить их. В пункте «Привилегии» можно настроить параметры доступа к SQL серверу. Вкладка «Операции» (рисунок 4.14) позволяет изменять сортировку таблицы, параметры таблицы, так же копировать и перемещать таблицу.

В пункте «Триггеры» можно создавать и изменять триггеры для таблицы, в данном проекте они не использовались.

Файлы контента хранятся в корневой папке сайта и при их создании и добавлении новых строк в таблицу нужно следить за согласованностью названий файлов. Контент создаётся с помощью Microsoft Word, набранный текст размещается в файле как необходимо создателю, и сохраняется как «Веб-страница».

Аналогичные операции можно проводить с любой таблицей. Но при изменении некоторых параметров, может поменяться и работа самого сайта, так что с этим нужно быть аккуратней.

В файле config.php хранятся параметры настройки с комментариями (см. приложение А), которые отвечают за:

1) подключение к базе данным;
2) создание кнопки для скачивания;
3) создание специальных блоков с программным кодом;
4) настройка кодировки;
5) текущие идентификаторы важных страниц, например главной станицы;
6) задний план сайта.

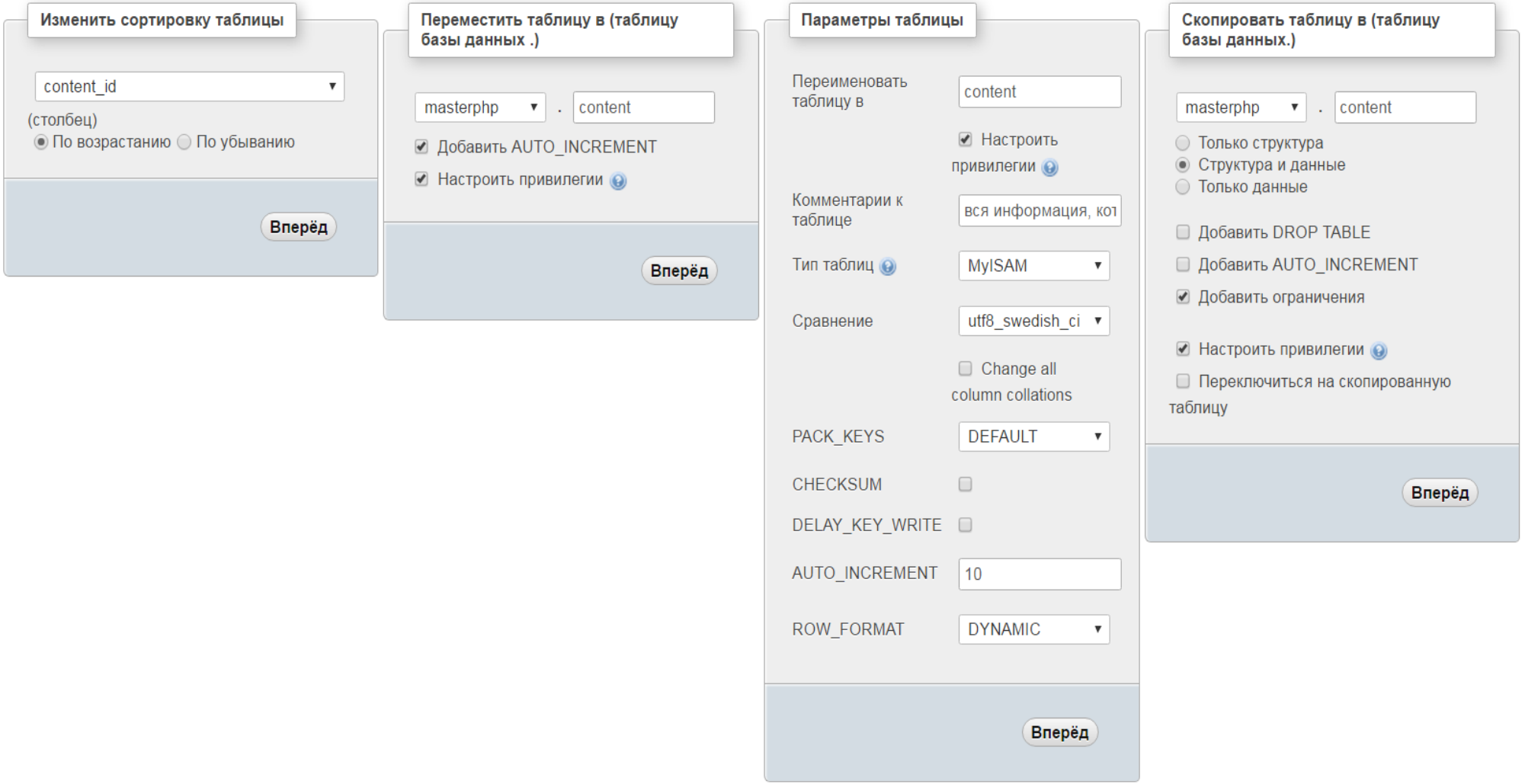

Рисунок 4.14 – Операции для таблиц

1 Выводы по разделу

В текущем разделе были рассмотрены два варианта использования приложения:

1) посетитель web-сайта;

2) владелец сайта, созданного на основе разработанного web-приложения.

Для обычного посетителя были описаны все функциональные элементы сайта, такие как меню навигации, блоки с программным кодом и кнопки для скачивания файлов.

Для владельца созданного web-приложения была описана детальная работы с программой phpMyAdmin, так как для создания web-сайта необходимо наполнить данными базу и создать внешние файлы с контентом, а программный код оставить без изменений.

ЗАКЛЮЧЕНИЕ

Результатом выпускной квалификационной работы является web-приложение «Портфолио-программиста», благодаря которому любой программист, даже не знающий web-программирования, может создать свой сайт, на котором будет рекламировать свои проекты. Информационная часть сайта хранится во внешних файлах, которые заполняются владельцем приложения без изменения программного кода. Также работа созданного приложения была продемонстрировано на личном примере.

В ходе работы выполнены поставленные требования, а именно:

1) показывать краткую информацию о программисте;
2) отображать список созданных проектов, сортированный по темам;
3) показывать возможные контакты.

Так же решены поставленные задачи, такие как:

1) разработать разметку страниц;
2) спроектировать и разработать базу данных, в которой хранится вся информация;
3) разработать web-сценарий, с помощью которого возможно редактирование и заполнение информации, без изменения программного кода;
4) обеспечить корректное отображение информации и сортировку по темам;
5) ведение статистики посещений сайта и загрузки файлов.

Для реализации выпускной квалификационной работы были использованы следующие программные средства:

1) язык разметки гипертекста HTML;
2) каскадная таблица стилей CSS;
3) язык серверных сценариев PHP;
4) скриптовый язык JavaScript с использованием библиотеки jQuery;
5) СУБД MySQL;
6) web-сервер Apache.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Янк, К. – PHP и MySQL. От новичка к профессионалу; пер. с англ. / К. Янк. – М.: Эксмо, 2013 – 384 с.: ил. – (Мировой компьютерный бестселлер).
2. Никсон, Р. – Создаём динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5. 4-е изд.; пер. с англ. / Р. Никсон. – СПб.: Питер, 2016 – 768 с.: ил. – (Серия «Бестселлеры O'Reily»).
3. Дейт, К. Дж. – Введение в системы баз данных, 8-е издание.; пер. с англ. / К. Дж. Дейт — М.: Вильямс, 2005. — 1328 с.: ил. — Парал. тит. англ.
4. Учебники для веб-разработчиков – URL: http:// wisdomweb.ru (даты обращений: 01.10.2016 - 15.03.2017).
5. Мациевский Н.С. – Реактивные веб-сайты. Клиентская оптимизация в алгоритмах и примерах: Учебное пособие; / Н.С. Мациевский, Е.В. Степанищев, Г.И. Кондратенко — М.: Интернет-Университет Информационных Технологий: БИНОМ. Лаборатория знаний, 2010. — 336 с.: ил., табл. — (Серия «Архитектор информационных систем»).
6. PHP. Официальный сайт – URL: http://php.net (даты обращений: 07.03.2017 - 02.05.2017).
7. Форум PHP программистов – URL: http://php.ru (даты обращений: 07.03.2017 - 02.05.2017).
8. HTML5 – URL: http://htmlbook.ru/html5 (даты обращений: 07.03.2017 - 16.04.2017).
9. Муссиано Ч. – HTML и XHTML. Подробное руководство, 6-е издание.; пер. с англ. / Ч. Муссиано, Б Кеннеди – СПб: Символ-Плюс, 2008. – 752 с., ил.
10. Самоучитель CSS – URL: http://htmlbook.ru/samcss (даты обращений: 08.02.2017 - 05.03.2017).
11. Нильсен Я. – Web-дизайн. Удобство использования Web-сайтов.; пер. с англ. / Я. Нильсен, Х. Лоранжен – М.: Вильямс, 2009. – 376 с., ил.
12. Макфарланд Д.С. – Большая книга CSS3 3-е изд.; пер. с англ. / Д.С. Макфарланд – СПб.: Питер, 2014 – 608 с.: ил. (Серия «Бестселлеры O'Reily»)
13. Флэнагал Д. JavaScript. Подробное руководство. – пер. с англ. / Д. Флэнагал, – СПб.: Символ-Плюс, 2008 – 992 с., ил.
14. jQuery: The Write Less, Do More, JavaScript Library – URL: https://jquery.com (даты обращений:07.03.2017 - 02.05.2017).
15. jQuery документация – URL: http://slyweb.ru (даты обращений: 07.03.2017 - 02.05.2017).
16. W3Schools OnlineWebTutorials – URL: https://w3schools.com (даты обращений: 08.02.2017 - 02.05.2017).
17. Макфарланд Д.С. – JavaScript и jQuery: исчерпывающее руководство; пер. с англ. / Д.С. Макфарланд – М.: Эксмо, 2015 – 880 с.: ил. (Мировой компьютерный бестселлер).

index.php:

```
<html>
<head>
<meta charset="utf-8">
<link href="css/jquery-ui.css" rel="stylesheet" type="text/css"/>
<link href="css/jquery-ui.structure.css" rel="stylesheet" type="text/css"/>
<link href="css/jquery-ui.theme.css" rel="stylesheet" type="text/css"/>
<link href="css/menu.css" rel="stylesheet" type="text/css"/>
<link href="css/diag.css" rel="stylesheet" type="text/css"/>
<link href="css/Chartist.css" rel="stylesheet" type="text/css"/>
<link href="css/diag_css.css" rel="stylesheet" type="text/css"/>

<script src="js/jquery.js"></script>
<script src="js/jquery-ui.js"></script>
<script src="js/Chartist.js"></script>

<script src="funcs.js"></script>
<script src="download_ajax.js"></script>
<script>
        $( function() {
   $( "#menu" ).menu();
        } );
</script>
<script>
   $(document).ready(function() {
                $('#menu li a').click(function(){
                //ищем ИД выбранного элемента,
                //позволяет найти то, что мы должны отобразить в контенте
                var toLoad = $(this).attr('href');
                var tagList = toLoad.split('/');
                var main_selected=tagList[1]; //таг 0 - родит. ид, 1 - выбранный подпункт меню
                start();
                Send(main_selected);
                });
        });
        function cl(th) // - ф-ция нажатия на ссылку в окне задач
        {
                var toLoad = $(th).attr('href');
                var tagList = toLoad.split('/');
                var main_selected=tagList[1]; //таг 0 - родит. ид, 1 - выбранный подпункт меню
                start();
                Send(main_selected);
        }
        function my_download(str)
        {
                var d_id = $(str).attr("id");
                var tagList_d= d_id.split('download');
                var d_selected=tagList_d[1];
                var numb_file=tagList_d[2];
                d_Init();
                Send_download(numb_file,d_selected);
        }
</script>
</head>
```

```
<?php
include 'config.php';
?>
<body onload="Init(); Send(12);" align="center" style='background: #262626 url("img/<?php echo
$back_ground;?>") 100% 100% scroll;'>
<?php
include 'funcs.php';
db();
ins_ip_db();
day_stat();
$result = get_cat();

echo '<div aling="left" style=" max-width:220px; min-width:220px"';
echo (view_cat($result));
echo '</div>';
?>

<div class="ui-widget-content" style="margin:5px; margin-left: 250px; height: 94.5%; padding-bottom: 10px;
background: rgba(0, 0,0, 0.7);"  >
                    <h3 align="center" class="ui-widget-header" id="panel">я</h3>
                    <div class="ui-widget-content" style="padding: 10px; margin: 10px 10px;" id="opis">
          </div>
                    <div id='content' style="height: 80vh; overflow: auto; text-align: left; margin-left: 15px;">
          </div>
                    <img  id="loadImg" src="img/ajax-loader2.gif" />
</div>
</body>


  </html>
```

funcs.js:

```
function start() // - функция запуска анимации
{
  // найдем элемент с изображением загрузки и уберем невидимость:
          $("#loadImg").show();
}

function stopLoadingAnimation() // - функция останавливающая анимацию
{
  $("#loadImg").hide();
}


//-----------------------------------
function createXmlHttpRequestObject(){
// функция создаёт и возвращает объект типа AJAX-запрос
var xmlHttp;
try{
          xmlHttp=new XMLHttpRequest();
}
catch(e){
          try{
                    xmlHttp=new ActiveXObject("MSXML2.XMLHTTP");
          }
          catch(e){
                    try{
                              xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
                    }
```

```
                catch(e){}
        }
}
if(!xmlHttp){
        alert("Не удалось создать объект XMLHttpRequest");
}
return xmlHttp;
}

//-----------------------------------------
function getResponseText(xmlHttp){
var txt=false; //alert("xmlHttp.readyState="+xmlHttp.readyState);
// проверка готовности результата
if(xmlHttp.readyState == 4){
        if (xmlHttp.status == 200){
                txt=xmlHttp.responseText; //alert(txt);
        } else {
                if (xmlHttp.status == 404){
                        alert("Request URL does not exist");
                } else {
                        alert("Error: status code is " + xmlHttp.status);
                }
        }
}
return txt;
}

//-------------------------------------
function Receive(){

        var txt=getResponseText(xmlHttp);
        if(!txt){
                return;
        }

        var tagList = txt.split("~@");
        var div1=document.getElementById("panel");
        var div2=document.getElementById("opis");
        var div3=document.getElementById("content");
        if  (tagList[0]==1){

                tagList[0]="Статистика";
                div3.innerHTML="";

                var div_stat1=document.createElement("div");
                div_stat1.className="chart1 ct-chart ct-golden-section";
                div_stat1.innerHTML="Количество посещений за неделю";
                div3.appendChild(div_stat1);

                var div_stat2=document.createElement("div");
                div_stat2.innerHTML="Диаграмма посещений задач";
                div_stat2.className="chart2 ct-chart ct-golden-section";
                div3.appendChild(div_stat2);

                var div_stat3=document.createElement("div");
                div_stat3.innerHTML="Диаграмма посещений курсовых";
                div_stat3.className="chart3 ct-chart ct-golden-section";
                div3.appendChild(div_stat3);

                var div_stat4=document.createElement("div");
```

```
div_stat4.innerHTML='Количество загрузок из "Задач"';
div_stat4.className="chart4 ct-chart ct-golden-section";
div3.appendChild(div_stat4);

var div_stat5=document.createElement("div");
div_stat5.innerHTML='Количество загрузок из "Курсовых"';
div_stat5.className="chart5 ct-chart ct-golden-section";
div3.appendChild(div_stat5);

//массивы для  посещений
var mas_stat=tagList[1].split("?");
var count=[];
var day=[];
for(i=0; i<mas_stat.length-1;i++){
	if(i%2==0)
		count[(i)/2]=mas_stat[i];
	else
		day[(i-i%2)/2]=mas_stat[i];
}

//массивы для задач
var mas_stat=tagList[2].split("?");
var count_lang=[];
var language=[];
for(i=0; i<mas_stat.length-1;i++){
	if(i%2==0)
		count_lang[(i)/2]=mas_stat[i];
	else
		language[(i-i%2)/2]=mas_stat[i];
}

//массивы для курсовых
var mas_stat=tagList[3].split("?");
var count_lang2=[];
var language2=[];
for(i=0; i<mas_stat.length-1;i++){
	if(i%2==0)
		count_lang2[(i)/2]=mas_stat[i];
	else
		language2[(i-i%2)/2]=mas_stat[i];
}

//массивы для загрузок задач
var mas_stat=tagList[4].split("?");
var down_count1=[];
var down_lang1=[];
for(i=0; i<mas_stat.length-1;i++){
	if(i%2==0)
		down_count1[(i)/2]=mas_stat[i];
	else
		down_lang1[(i-i%2)/2]=mas_stat[i];
}

//массивы для загрузок курсовых
var mas_stat=tagList[5].split("?");
var down_count2=[];
var down_lang2=[];
for(i=0; i<mas_stat.length-1;i++){
	if(i%2==0)
		down_count2[(i)/2]=mas_stat[i];
```

```
                        else
                                down_lang2[(i-i%2)/2]=mas_stat[i];
                }

                //график для посещений
                new Chartist.Bar('.chart1', {labels: day, series:[count]},
                {fullWidth: false,chartPadding: {top: 50}});

                //диаграмма для задач
                var data = {labels:language,series: count_lang};
                new Chartist.Pie('.chart2', data, {             donutSolid: true,showLabel: true });

                //диаграмма для курсовых
                var data = {labels:language2, series: count_lang2};
                new Chartist.Pie('.chart3', data, {             donutSolid: true,showLabel: true });

                new Chartist.Bar('.chart4', {labels: down_lang1, series:[down_count1]},
                {fullWidth: false,chartPadding: {top: 50}});

                new Chartist.Bar('.chart5', {labels: down_lang2,series:[down_count2]},
                {fullWidth: false,chartPadding: {top: 50}});
        }
        else {
                div2.innerHTML=tagList[1];
                div3.innerHTML=tagList[2];
        }

        div1.innerHTML=tagList[0];
        stopLoadingAnimation();
}
//--------------------
function Init(){
        xmlHttp=createXmlHttpRequestObject(); // создаём объект
        xmlHttp.onreadystatechange=Receive;  // назначаем обработчик ответа сервера
}

//-----------------------------------
function Send(x){
        //alert(x);
        var params="numb_file="+x;
        //alert(params);
        xmlHttp.open("POST","Login.php",true);
        xmlHttp.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded'); // Отправляем
кодировку
        /*xmlHttp.onreadystatechange = function() {
                if (this.readyState == 4 && this.status == 200)
                {
                        alert(this.responseText);
                        var xml=getResponseText(xmlHttp);
                        if(!xml)
                        {
                                alert('вылет');
                                return;
                        }
                }
        }*/
        xmlHttp.send(params);
        //alert('sending');

}
```

funcs.php:

```
<?php

		function db() //подключение к БАза данных
		{
		   include 'config.php';
		   $mysqli = mysqli_init();
		   $db=@mysql_connect($hostname_connect, $username_connect, $password_connect);
		   if (!$db)
		   {
			exit('Ошибка при подключении к базе данных');
		   }
		   if(!mysql_select_db($database_connect,$db))
		   {
			exit('База данных не существует');
		   }
		   mysql_query('SET NAMES utf8');
		}

		function get_cat()  //Возвращает ассоциативный массив
		{
		   include "config.php";
		   //запрос к базе данных
		   if(get_ip()==$admin_ip)
			$sql = "SELECT * FROM menu order by menu_name";
		   else
			$sql = "SELECT * FROM menu where menu_id <> '$stat_id' order by menu_name";
		   $result = mysql_query($sql);
		   if(!$result)
		   {
			return NULL;
		   }
		   $arr_cat = array();
		   if(mysql_num_rows($result) != 0)
		   {
			//В цикле формируем массив
			for($i = 0; $i < mysql_num_rows($result);$i++)
			{
				$row = mysql_fetch_array($result,MYSQL_ASSOC);

				//Формируем массив, где ключами являются адишники на родительские
категории
				if(empty($arr_cat[$row['menu_parent']]))
				{
					$arr_cat[$row['menu_parent']] = array();
				}
				$arr_cat[$row['menu_parent']][] = $row;
			}
			//возвращаем массив
			//print_r($arr_cat);
			return $arr_cat;
		   }
		}
		function view_cat($arr,$parent_id = null) //Преобразовывает ассоциативный массив в список
на html
		{
		   //Условия выхода из рекурсии
		   if(empty($arr[$parent_id]))
		   {
			return;
```

```
        }
        echo '<ul id="menu">';
        //перебираем в цикле массив и выводим на экран
        for($i = 0; $i < count($arr[$parent_id]);$i++)
        {
                echo '<li  style="font: 13pt sans-serif; align: center;"><a href="#'.$parent_id.
                                        '/'.$arr[$parent_id][$i]['menu_id'].'">'
                                        .$arr[$parent_id][$i]['menu_name'].'</a>';
                //рекурсия - проверяем нет ли дочерних категорий
                view_cat($arr,$arr[$parent_id][$i]['menu_id']);
                echo '</li>';
        }
        echo '</ul>';
    }
    function get_ip() // возвращает IP
    {
        if (!empty($_SERVER['HTTP_CLIENT_IP']))
        {
                $ip=$_SERVER['HTTP_CLIENT_IP'];
                //          echo 'HTTP_CLIENT_IP';
        }
        elseif (!empty($_SERVER['HTTP_X_FORWARDED_FOR']))
        {
                $ip=$_SERVER['HTTP_X_FORWARDED_FOR'];
                //echo 'HTTP_X_FORWARDED_FOR';
        }
        else
        {
                $ip=$_SERVER['REMOTE_ADDR'];
                //echo "REMOTE_ADDR";
        }
        //print_r $_POST;
        //echo $ip;
        return $ip;
    }
    function ins_ip_db()
    {
        db();
        $ip=get_ip();
        //echo $ip;
        $sql="SELECT user_id FROM user WHERE ip='$ip'";
        $result = mysql_query($sql);
        if(!$result or !mysql_num_rows($result))
        {
                $sql2=mysql_query("insert into user(ip) value('$ip')");
$result2 = mysql_query($sql) or die(mysql_error());
        }
    }
    function int_date_db($menu_id)
    {
        $ip=get_ip();
        $sql="SELECT user_id FROM user WHERE ip='$ip'";
        $result=mysql_query($sql);
        if(!empty($result))
        {
                $id=mysql_fetch_assoc($result);
                $new_id=$id["user_id"];
                $sql2=mysql_query("insert into visit(user_id,  menu_id) value('$new_id',  '$menu_id')");
        }
    }
    function day_stat()
```

```
{
    $ip=get_ip();
    $sql="SELECT user_id FROM user WHERE ip='$ip'";
    $result=mysql_query($sql);
    $d=date("Y:m:d");
    if(!empty($result))
    {
        $id=mysql_fetch_assoc($result);
        $new_id=$id["user_id"];
        $sql2=mysql_query("insert into stat(user_id, dat) value('$new_id','$d')");
    }
}
function checkfile($menu_id)
{
    //$menu_id=$_POST['postVar'];
    $sql = "SELECT content_name FROM content where menu_id='$menu_id'";
    $result = mysql_query($sql);
    if(!$result)
    {
        return NULL;
    }
    $row = mysql_fetch_assoc($result);
    print_r($row['content_name']);
    $file= file_get_contents('./files/'.$row['content_name']);
    echo "<div id='content'>".$file."</div>";
}
function get_stat()
{
    include "config.php";
    echo "1~@";
    //статистика по посещениям
    $sql = "SELECT  count(distinct(user_id)), dat FROM stat group by dat having
dat>=(CURDATE()- INTERVAL '$count_day' day)";
    $result = mysql_query($sql) or die();
    while($row = mysql_fetch_array($result)) {
      for ($j = 0 ; $j < 2 ; ++$j) echo "$row[$j]?";
    }
    echo "~@";
    //статистика по задачам
    $sql = "SELECT count(visit.menu_id), menu_name from menu, visit where
menu.menu_id=visit.menu_id and menu.menu_parent='$task_id'  group by visit.menu_id";
    $result = mysql_query($sql) or die();
    while($row = mysql_fetch_array($result)) {
      for ($j = 0 ; $j < 2 ; ++$j) echo "$row[$j]?";
    }
    echo "~@";
    //статистика по курсовым
    $sql = "SELECT count(visit.menu_id), menu_name from menu, visit where
menu.menu_id=visit.menu_id and menu.menu_parent='$curs_id'  group by visit.menu_id";
    $result = mysql_query($sql) or die();
    while($row = mysql_fetch_array($result)) {
      for ($j = 0 ; $j < 2 ; ++$j) echo "$row[$j]?";
    }
    echo "~@";
    //статистика по загрузкам
    $sql = "SELECT count(dg.download_id), menu_name from menu as m, content as c, download
as d, downloading as dg WHERE m.menu_id=c.menu_id AND d.content_id=c.content_id and
d.download_id=dg.download_id and m.menu_parent='$task_id' group by d.content_id";
    $result = mysql_query($sql) or die();
    while($row = mysql_fetch_array($result)) {
      for ($j = 0 ; $j < 2 ; ++$j) echo "$row[$j]?";
```

```
                }
                echo "~@";
                //статистика по загрузкам
                $sql = "SELECT count(dg.download_id), menu_name from menu as m, content as c, download
as d, downloading as dg WHERE m.menu_id=c.menu_id AND d.content_id=c.content_id and
d.download_id=dg.download_id and m.menu_parent='$curs_id' group by d.content_id";
                $result = mysql_query($sql) or die();
                while($row = mysql_fetch_array($result)) {
                  for ($j = 0 ; $j < 2 ; ++$j) echo "$row[$j]?";
                }
            }
?>
```

Login.php:

```
<?php
include 'config.php';
include 'funcs.php';
$numb_file=$_POST['numb_file'];
db();
int_date_db($numb_file);
            //чекаем является ли отправленный номер родителем
            //если да, то надо сделать в ДИВе ссылки на детей
            $sql = "SELECT menu_name,menu_id FROM menu where menu_id='$numb_file' and
menu_parent is NULL";
            $result = mysql_query($sql);
            $len = mysql_num_rows($result);
            $parent=mysql_fetch_assoc($result);
              //$p=$parent['menu_name'];

            if($parent['menu_id']==$home_id || $parent['menu_id']==$stat_id) // работа с начальной
страницей
            {
              if($parent['menu_id']==$stat_id)
              {
                    get_stat();
              }
              else
              {
                    //найдём имя файла для начальной страницы
                    $sql = "SELECT content_name FROM content where menu_id='$home_id'";
                    $result = mysql_query($sql);
                    if(!$result)
                    {
                          return NULL;
                    }
                    $row = mysql_fetch_assoc($result);

                    //читаем из файла и изменяем кодировку
                    $file= file_get_contents ($row['content_name']);
                    $file=iconv("Windows-1251","UTF-8", $file);

                    //c
                    $sql = "SELECT menu_name FROM menu where menu_id='$home_id'";
                    $result = mysql_query($sql);
                    if(!$result)
                    {
                          return NULL;
                    }
                    $name = mysql_fetch_assoc($result);
                    //$name="Главная страница";
```

```
                        //формируем исходную строку
                        $end=$name['menu_name']."~@~@".$file;
                        echo ($end);
                    }
                }
                else
                {
                    $sql = "SELECT menu_name,menu_id FROM menu where menu_parent='$numb_file' order by
menu_name";
                    $result = mysql_query($sql)or die(mysql_error());
                    $len = mysql_num_rows($result);
                    //если есть дети, то ЛЕН>0
                    if ($len>0)
                    {
                                $f=$parent['menu_name'].'~@~@<ul id="menu2">';
                                while($roww=mysql_fetch_array($result,MYSQL_ASSOC))
                                {
                                        $h='<li style="font: 15pt sans-serif; color: white; face: Arial ; text-align:
left; margin-left:2%; margin-bottom:20px;"><a href="#'.$numb_file.
                                                                '/'.$roww['menu_id'].'" onclick="cl(this)"> '
                                                                .$roww['menu_name'].'</a></li>';
                                        //рекурсия - проверяем нет ли дочерних категорий
                                        $f=$f.$h;
                                }
                                $f=$f.'</ul>';
                                echo $f;
                    }
                    else
                    {
                    //находим имя в бд, то что пришло
                            $sql = "SELECT content_name, content_id FROM content where
menu_id='$numb_file'";
                            $result = mysql_query($sql);
                            if(!$result)
                            {
                                    return NULL;
                            }
                            $row = mysql_fetch_assoc($result);

                            //читаем из файла и изменяем кодировку
                            $file= file_get_contents ($row['content_name']);
                            $file=iconv("Windows-1251","UTF-8", $file);
                            $content_id=$row['content_id'];

                            //в header надо бы поместить тему
                            $sql = "SELECT menu_name, menu_parent FROM menu where
menu_id='$numb_file'";
                            $result = mysql_query($sql);
                            if(!$result)
                            {
                                    return NULL;
                            }
                            $tema= mysql_fetch_assoc($result);
                            $temaa=$tema['menu_name'];
                            //$temaa=iconv("Windows-1251","UTF-8", $tema['menu_name']);
                            //найдём основную тему
                            //курсачи, или задачи, или ещё что мб
                            $sql = "SELECT menu_name FROM menu where menu_id=".$tema['menu_parent'];
                            $result = mysql_query($sql);
                            if(!$result) //главный пункт меню
                            {
```

```
                        $main_tema=$temaa;
                        $temaa='';
                    }
                    else
                    {
                        $par= mysql_fetch_assoc($result);
                        $main_tema=$par['menu_name'];
                    }
                    //выделим код на странице
                    $find_code="##";
                    $len_find_code=strlen($find_code);
                    while ($n_str=stripos($file,$find_code))
                    {
                        $str="<div class='code'>";
                        $file=substr_replace($file,$str,$n_str, $len_find_code);
                    }

                    $find_end_code="#~#";
                    $len_find_end_code=strlen($find_end_code);
                    while ($n_str=stripos($file,$find_end_code))
                    {
                        $str="</div>";
                        $file=substr_replace($file,$str,$n_str, $len_find_end_code);
                    }

                    //поиск #Скачать: в файле, подставим туда гиперссылку или кнопку
                    $sql="SELECT file_name FROM download WHERE content_id='$content_id'";
                    $result=mysql_query($sql) or die(mysql_error());
                    $i=0;
                    while($row=mysql_fetch_array($result))
                    {
                        $mas_download_file[$i]=$row['file_name'];
                        $i++;
                    }
                    $find_count=substr_count($file, $find_str);
                    for ($i=0; $i<$find_count; $i++)
                    {
                        $str='<p align="center"> <font color="#8cce3b", size="4">Исходный файл:  '.
$mas_download_file[$i].'  <input id="download'.($i+1).'download'.$numb_file.'" class="ui-button" type="button"
value="Скачать" onclick=my_download(this);location.href="/files/'.$mas_download_file[$i].'"></input></p>';
                        $n_str=stripos($file,$find_str); // Возвращает позицию первого вхождения
подстроки без учета регистра
                        $file=substr_replace($file,$str,$n_str, $len_find_str); //Заменяет часть строки
                    }
                    $end=$main_tema."~@".$temaa."~@".$file;
                    echo ($end);
                }
            }
?>
```

download_ajax.js:

```
function d_Init(){
    xmlHttp_d=createXmlHttpRequestObject(); // создаём объект
    //xmlHttp1.onreadystatechange=Receive1;  // назначаем обработчик ответа сервера
}
//-----------------------------------
function Receive1()
{
    alert(getResponseText(xmlHttp1));
}
```

```
function Send_download(x,y){
	var params="numb_file="+x+"&id="+y;
	xmlHttp_d.open("POST","download_ajax.php",true);
	xmlHttp_d.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded'); // Отправляем
кодировку
	xmlHttp_d.send(params);
}
```

download_ajax.php:

```
<?php
//echo 'priem';
include 'config.php';
include 'funcs.php';
db();
$numb_file=$_POST['numb_file'];
$numb_download=$_POST['id']-1;

// поиск ИП пользователя, который хочет скачать файл
$user=get_ip();
$sql="SELECT   user_id from visit order by  tim DESC limit 1";
$result=mysql_query($sql) or die(mysql_error());
$user_ip=mysql_fetch_assoc($result);
$u=$user_ip['user_id'];

		// все ID возможных загрузок со страницы запишем в массив
		$sql="SELECT download_id FROM  download, content WHERE
download.content_id=content.content_id and menu_id='$numb_file'";
		$result=mysql_query($sql) or die(mysql_error());		$i=0;
		while($row=mysql_fetch_array($result))
		{
			$mas_download_file[$i]=$row['download_id'];
			$i++;
		}
		//запись в таблицу по закачке
$sql="insert into downloading(user_id, download_id) value('$u', '$mas_download_file[$numb_download]')";
$result=mysql_query($sql) or die(mysql_error());
?>
```

config.php:

```
<?php
//настройки соединения в БД
$hostname_connect='localhost';
$database_connect='masterphp';
$username_connect='root';
$password_connect='';

$admin_ip="::1"; //IP администратора

$back_ground='2.jpg'; // название картинки заднего фона
$back_ground=iconv("Windows-1251","UTF-8", $back_ground);

$stat_id=18; //ИД раздела статистики
$count_day=7; //количество дне для просмотра статистики посещний
$cours_id=7;// ИД раздела курсовых
$task_id=1; // ИД раздела задач
$curs_id=7;
$find_str="downloadnow"; // код для скачивания файла
$find_code="##"; // начало блока исходного текста кода
$find_end_code="#~#";// конец блока исходного текста кода
$find_str=iconv("Windows-1251","UTF-8", $find_str);
```

```
$len_find_str=strlen($find_str);
$home_id=12; // ИД начальной страницы из БД

@mysql_query("set character_set_client='utf8'");
@mysql_query("set character_set_results='utf8'");
@mysql_query("set collation_connection='utf8_unicode_ci'");
?>
```

diag_css.css:

```
.chart1.ct-chart .ct-series.ct-series-a .ct-bar,
.chart4.ct-chart .ct-series.ct-series-a .ct-bar,
.chart5.ct-chart .ct-series.ct-series-a .ct-bar{
  stroke: #b8ec79;
}

.chart1.ct-chart .ct-series.ct-series-a .ct-point,
.chart4.ct-chart .ct-series.ct-series-a .ct-point,
.chart5.ct-chart .ct-series.ct-series-a .ct-point{
  stroke: #b8ec79;
}
.ct-grid{
        stroke:#D3D3D3;
}
.ct-label{
        color: #D3D3D3;
        font-size: 15;
}
.chart1.ct-chart .ct-series .ct-grid,
.chart4.ct-chart .ct-series .ct-grid,
.chart5.ct-chart .ct-series .ct-grid{
        color:#b8ec79;
}
@keyframes dasharray-craziness {
   0% {
      stroke-dasharray: 7px 2px;
   }
   80% {
      stroke-dasharray: 7px 50px;
      stroke-width: 10px
   }
   100% {
      stroke-dasharray: 7px 2px;
   }
}
.chart1,.chart2,.chart3,.chart4,.chart5{
        height: 50%;
        width: 80%;
        margin: auto;
        margin-top: 5%;
        color:#D3D3D3;
        font-size:25;
        text-align: center;
}
.ct-series-a .ct-slice-pie, .ct-series-a .ct-slice-donut-solid, .ct-series-a .ct-area {
        fill: #7CFC00;
}
.ct-series-b .ct-slice-pie, .ct-series-b .ct-slice-donut-solid, .ct-series-b .ct-area {
        fill: #228B22;
}
.ct-series-c .ct-slice-pie, .ct-series-c .ct-slice-donut-solid, .ct-series-c .ct-area {
        fill: #40E0D0;
```

```
}
.ct-series-d .ct-slice-pie, .ct-series-d .ct-slice-donut-solid, .ct-series-d .ct-area {
	fill: #AFEEEE;
}
.ct-series-e .ct-slice-pie, .ct-series-e .ct-slice-donut-solid, .ct-series-e .ct-area {
	fill:  #00FA9A;
}
.ct-chart .ct-series.ct-series-a .ct-slice.ct-donut {
	stroke: #002827
 }
.ct-chart-pie .ct-label{
	stroke: #000000;
	font-size: 25;
	padding-left: 25px;
	dominant-baseline:auto;
}
.ct-chart-pie {
	margin-top: 30px;
}
```

menu.css:

```
html {
	background: #333 url(http://subtlepatterns.com/patterns/tex2res4.png) 0 0 repeat;
	min-height: 100%;
	font-family: "Open Sans", sans-serif;
	font-weight: 300;
	color: #FFF;
}

body, html {
	height: 100%;
	margin: 0;
	padding: 0;
}

a {
	color: rgba(255,255,255, 0.8);
	text-decoration: none;
}

	a:hover, li:hover > a {
		color: #b8ec79;
	}

ul {
	list-style-type: none;
	margin: 0;
	padding: 0;
}

li {
	margin: 0;
	padding: 0;
}

#menu {
	border-left: 2px solid #b8ec79;
	border-right: 2px solid #b8ec79;
	background: rgba(0, 0,0, 0.7);
	width: 200px;
	float: left;
```

```
	font-size: 1.5em;
	min-height: 95%;
	margin-left: 15px;
	margin-bottom: 2.5%;
	position: fixed;
}

#menu li {
	position: relative;
	z-index: 1;
}

#menu li a {
	display: block;
	padding: 0.5em 1em;
	white-space: nowrap;
}

	#menu  li ul {
		position: absolute;
		display: none;
		left: 100%;
		top: 0.5em;
		float: none;
		background-image: -moz-radial-gradient(0 50%, ellipse  farthest-side, rgba(150,242,38,0.6) 0%,
rgba(150,242,38,0.5) 33%, rgba(150,242,38,0) 100%);
		background-image: -webkit-radial-gradient(0 50%, ellipse  farthest-side, rgba(150,242,38,0.6)
0%, rgba(150,242,38,0.5) 33%, rgba(150,242,38,0) 100%);
		background-image: radial-gradient(0 50%, ellipse  farthest-side, rgba(150,242,38,0.6) 0%,
rgba(150,242,38,0.5) 33%, rgba(150,242,38,0) 100%);
	}

	#menu li:hover ul {
		display: block;
	}

	#menu li ul a {
		position: relative;
		font-size: 0.8em;
	}

	#menu li ul a:hover:before {
		content: "";
		display: block;
		width: 1em;
		height: 1em;
		background: rgba(150,242,38.75);
		border: 1px solid #b8ec79;
		position: absolute;
		top: 0.5em;
		left: -0.75em;
		-moz-transform: rotate(45deg);
		-webkit-transform: rotate(45deg);
		transform: rotate(45deg);
	}


	 #loadImg
	 {
		position: absolute; /* Абсолютное позиционирование */
		bottom: 80%; /* Положение от нижнего края */
```

```
        left: 50%; /* Положение от правого края */
}

#up
{
        width:60px;
        height:60px;
        position:fixed;
        bottom:50px;
        left:20px;
        background-color:#000000;
        border-radius:30px;
}
#down
{
        width:60px;
        height:60px;
        position:fixed;
        bottom:50px;
        left:90px;
        background-color:#000000;
        border-radius:30px;
}
.pPageScroll
{
        color:#FFFFFF;
        font:bold 12pt 'Comic Sans MS';
        text-align:center;
}
.code
{
        background: #b8ec79;
        margin: 15px;
        max-height: 200px;
        padding: 5px;
        overflow: auto;
}
```