МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра Компьютерных технологий

«Допущен к защите»
Заведующий кафедрой _____
_____
(Ф.И.О., ученая степень, звание)
_____ «_____»_____**2014 г.**
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка виртуального музея истории развития компьютерной техники

Специальность: 5В070400 – Вычислительная техника и программное обеспечение

Выполнил: Самоваров В. В. группа: ВТУ 11-1

Научный руководитель: Мусатаева Г. Т., старший преподаватель

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

Еркешева «20» мая 2014 г.
(подпись)

по безопасности жизнедеятельности:
Бегимбетова А. С., старший преподаватель

_____ «18» апреля 2014 г.
(подпись)

по применению вычислительной техники:
Мусатаева Г.Т., старший преподавать

Мусат «26» мая 2014 г.
(подпись)

Нормоконтролер: Тусупов Д. М., ассистент

_____ «26» мая 2014 г.
(подпись)

Рецензент: Байтуленов Жаныбек Бахытович, доцент

_____ «_____»_____2014 г.
(подпись)

Алматы 2014 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет: Информационных Технологий
Специальность: 5В070400 – Вычислительная техника и программное обеспечение
Кафедра: Компьютерных Технологий

ЗАДАНИЕ
на выполнение дипломного проекта

Студент: Самоваров Валерий Вячеславович

Тема проекта: Разработка виртуального музея истории развития компьютерной техники утверждена приказом ректора № 115 от «24 » сентября 2014 г.
Срок сдачи законченной работы « 26 » мая 2014 г.
Исходные данные к проекту требуемые параметры результатов проектирования (исследования) и исходные данные объекта:
Спроектировать и разработать информационный веб-сайт «Виртуальный музей компьютерной техники» с использованием современных технологий, основной целью которого является повышение уровня знаний об истории развития компьютерной техники, а также о современных разработках в этой области. В ходе работы изучаются и применяются на практике современные методы создания веб-страниц.

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:

1. Исследование языков и методов построения современных веб-страниц
2. Изучение основных этапов создания веб-сайта
3. Создание схемы разработки проекта
4. Разработка проекта на основе исследований и полученной схемы

Перечень графического материала (с точным указанием обязательных чертежей)

В основном разделе содержатся схемы и рисунки разрабатываемого проекта. В экономической части диаграммы по расходам на проект, в обзор схема покрытия.

Рекомендуемая основная литература

1. Дронов В. Разработка современных web-сайтов. - М.: «Фолиант»,2011.
2. Шмитт К. CSS. Рецепты программирования. Издательство «Питер», 2011.
3. Зандстра М. PHP. Практика создания web-сайтов. «Вильямс», 2010.
4. Кент М. PHP для начинающих. Издательство «Вильямс», 2006.

Консультанты по проекту с указанием относящихся к ним разделов

Раздел	Консультант	Сроки	Подпись
Основная часть	Мусатаева Г. Т.	02.02 — 20.05.	
Охрана труда	Бегимбетова А. С.	14.03 — 18.04.14	
Экономическая часть	Еркешева З. Д.	13.03 - 20.05.14	
Нормоконтролер	Тусупов Д. М.	26.05.14г.	

Г Р А Ф И К

подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
1.	Исследование языков и методов построения современных веб-страниц	02.02 – 27.02	
2.	Изучение основных этапов создания веб-сайта	27.02 – 16.03	
3.	Создание схемы разработки проекта	16.03 – 31.03	
4.	Разработка проекта на основе исследований и полученной схемы	31.03 – 26.05	

Дата выдачи задания «24» сентября 2013 г.

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

Руководитель Мусатаев Г.Г. Мусатаева Г.Г.
(подпись) (Фамилия и инициалы)

Задание принял к исполнению
студент _____ Самоваров В.В.
(подпись) (Фамилия и инициалы)

Аңдатпа

Бұл бітіру жұмысында тарихта маманданған және компьютерлік техниканың қазіргі зерттемелерінде ақпараттық веб–сайт әзірленеді. Қазіргі заманғы веб–сайт әзірлеу үшін ендігі: мақсаттар зерттеліп тағайындалған: қазіргі заманғы әдебиеттер мен тарихи деректер, сайтты қажетті ақпаратпен толтыру, веб–сайттың зерттемесінің негізгі ұстанымдары, сайттың контентін басқару панельсі құрылған. Құрылған жүйенің экономикалық тиімділігі есептелінген. Өміртіршілік қауіпсіздігі, еңбек ету жағдайлары мәселелері қарастырылған, жұмыс орнының жарықтандыру есептеулері келтірілген.

Аннотация

В данной выпускной работе разрабатывается информационный веб–сайт, специализирующейся на истории и современных разработках компьютерной техники. Для создания современного веб–сайта исследованы и определены следующее: задачи; необходимая современная литература, исторические данные, необходимые для заполнения сайта информацией; основные принципы разработки веб–сайтов; создана панель управления контентом сайта. Произведен расчет экономической эффективности разработанного проекта. Рассмотрены вопросы безопасности жизнедеятельности, условий труда, выполнен расчет искусственного освещения рабочего помещения.

Annotation

In this final work developed an information website specializing in the history and development of modern computer technology. To create a modern website researched and identified the following: objectives; necessary modern literature and historical data needed to fill the site with information; guidelines for the development of websites; created content management panel. The calculation of the economic efficiency of the developed project. The issues of life safety, working conditions, payment is made artificial lighting workroom.

Содержание

Введение

Интернет – всемирная система объединённых компьютерных сетей, построенная на базе протокола IP и маршрутизации IP–пакетов. Интернет образует глобальное информационное пространство, которое является физической основой для Всемирной паутины (World Wide Web) представляет собой огромное более миллиона страниц, хранилище веб–сайтов на самые различные темы и самых различных вариациях.

Хороший сайт, собирая в себя всю полезную информацию, является лучшей визитной карточкой и предприятия, и образовательного учреждения, работая на них в любое время суток [1].

На данный момент уже более 50% всех компаний в Казахстане имеют свою страничку в интернете, это позволяет привлечь новых потенциальных потребителей или партнеров в свой бизнес. Но не только компании взялись за штурм виртуальных укреплений, свой сайт сейчас может иметь каждый. Даже у правительства есть свой сайт, где они освещают различного рода общественно–политическую информацию для граждан нашей страны. Крупнейшие музеи мира создали свои 3D сайты с «эффектом присутствия», сейчас можно прогуляться в виртуальной экскурсии по «Лувру» и это далеко не предел современных технологий.

Многие веб–мастера не уделяют времени на то, чтобы продумать будущий поток информации, а ограничиваются только добавлением текста и картинок на страницу. Но на многих веб–сайтах вы можете увидеть хвастливые заявления, что у них намного больше посетителей, чем у конкурентов, однако это не лучший показатель качества. Одним из основных показателей является время, которое пользователи тратят на то, чтобы остаться на вашем сайте достаточно долго, чтобы пройти по разным его уровням и разделам, то вы можете быть уверенными в качественно выполненной работе. Это означает, что они могут найти то, что им нужно, осмотреть остальное и без особого труда вернуться обратно.

Задачи, решению которые может выполнять веб–сайт, условно разделяют на три группы:

- рекламные;
- коммерческие;
- организационного развития, мотивации пользователей, образования.

Следует отметить, что все эти задачи решаются с применением разных средств: распространения печатной рекламы, подготовки рекламных публикаций, участия в различных выставках, издания газет и журналов, проведения курсов. Однако виртуальная среда обладает целым рядом несомненных преимуществ и возможностей над обычным подходом, среди которых:

- интерактивный характер представления;
- доступность информации в течение 24 часов пользователям всего мира;

- быстрое обновление информации, в том числе ее дополнение с учетом вопросов или пожеланий посетителей сайта;
- предоставление неограниченного объема информации, причем как текстовых и графических данных, так медиаданных;
- унификация информации, предназначенной для различных групп пользователей;
- расширенный и быстрый поиск необходимых сведений в больших массивах информации;
- получение данных о посещаемости сайта, т. е. его эффективности как средства коммуникации;
- создание сайтов, которые представляют отдельные товары или услуги, или ориентированы на различные целевые аудитории.

Однако перечисленные достоинства приобретаются веб–сайтами не сразу, а появляются лишь в результате хорошо продуманного, обоснованного подхода к их созданию [1].

В сети интернет уже существует множество различных «виртуальных музеев» на различные темы. Но в основном это маленькие проекты, созданные не практично, не удобно или не качественно для пользователей. На просторах Казахстанского интернета не было найдено хорошо построенного, качественного сайта посвященного истории и современным разработкам компьютерной техники, не считая того, что на данный момент компьютерная грамотность в нашей стране еще остается на уровне ниже среднего. Исходя из этого я решил создать сайт посвященный этой теме, раскрыть историю развития компьютерной техники, размещать статьи о современных разработках в области программного и аппаратного обеспечения используя современный подход в разработке сайтов. Так же в проекте было реализованы различные модули взаимодействия с потенциальными пользователями сайта, такие модуль тестирования и модуль обратной связи, также на сайте присутствуют кнопки различных социальных сетей и модуль поиска по сайту.

1 Что такое современный веб–сайт?

1.1 HyperText Markup Language – HTML

Каждый раз, открывая браузер, вы попадаете на веб–сайт, который может состоять из одной или множества веб–страниц с различной информацией. Основной технологией создания таких веб–страниц является язык HTML.

HyperText Markup Language – «язык гипертекстовой разметки» – иными словами он отвечает за расположение в документе текстов, рисунков, таблиц предназначенных для отображения в сети Интернет. Заставить его посчитать, сколько будет дважды два невозможно, в нем нет логических функций, зато красиво и главное легко опубликовать информацию о том, что дважды два будет четыре – это запросто. Читается этот язык при помощи знакомых программ, именуемых браузерами (обозревателями), которые "знают" все теги и команды языка HTML, и обрабатывая их выводит на монитор компьютера документы в том виде, в котором хочет представить их веб–мастер – разработчик документа.

Документ HTML разделяется на две основные части: заголовок – head и тело – body. Заголовок содержит такие сведения о документе, как его название и методико-техническую информацию, описывающую содержимое. В теге body находится само содержимое документа (то, что выводится в окне обозревателя).

Каждый тег состоит из имени, за которым может следовать список необязательных атрибутов, все они находятся внутри угловых скобок < >. Содержимое скобок никогда не выводится в окне браузера. Имя тега, как правило, представляет собой аббревиатуру его функции, что облегчает его запоминание. Атрибуты являются свойствами, которые расширяют или уточняют функцию тега. Как правило, имя и атрибуты внутри тега не чувствительны к регистру [2].

Контейнеры. Большинство тегов являются контейнерами. Это означает, что у них имеется открывающий и закрывающий теги. Текст, находящийся между тегами, будет выполнять содержащиеся в них инструкции. Закрывающий тег имеет то же имя, что и начальный, но перед ним стоит знак слэш (/). Конечный тег никогда не содержит атрибутов.

В некоторых случаях конечный тег не обязателен, и обозреватель определяет конец тега из контекста. Чаще всего опускают конечный тег <p> (абзац). Браузеры раньше поддерживали этот тег без соответствующего завершения, поэтому многие разработчики привыкли использовать краткую форму.

Автономные теги. Некоторые теги не имеет завершающих тегов, потому что они используются для размещения отдельных (автономных) элементов на странице. Одним из них является тег изображения <img>, он просто помещает графику в поток страницы. Другие автономные теги – это разрыв строки (
), горизонтальная линия (<hr>) и теги, содержащие информацию о документе и не

влияющие на содержимое, выводимое на экран, такие как <meta> и <base>.

Атрибуты. Атрибуты добавляются в тег для расширения или модификации его действий. К одному тегу можно добавить несколько атрибутов. Если атрибуты тега следуют после имени тега, они разделяются одним или несколькими пробелами. Порядок следования не важен. Большинство атрибутов имеют значения, которые следуют за знаком равенства (=), находящимся после имени атрибута. Длина значений ограничена 1024 символами. Значения могут быть чувствительны к регистру. Иногда значения должны находиться в кавычках (двойных или одинарных). Правила записи значения следующие:

– если значение представляет собой одно слово или число и состоит только из букв (a–z), цифр (0–9) и специальных символов (точка <.> или дефис<–>), то можно поместить его после знака равенства без кавычек;

– если значение содержит несколько слов, разделенных запятыми или пробелами, или содержит специальные символы, отличные от точки или дефиса, тогда его необходимо поместить в кавычки. Например, URL требуют кавычек, потому что они содержат символы "://". Также кавычки необходимы при задании значений цветов с использованием формата "#rrggbb" [2].

В теги HTML могут помещаться другие HTML–теги для осуществления воздействия нескольких тегов на один элемент. Это называется вложением, и, чтобы правильно его осуществить, начальный и конечный теги вложенного тега должны обязательно находиться между начальным и конечным тегами внешнего тега.

Часто встречающейся ошибкой является перекрытие тегов. Хотя часть браузеров отображают содержимое, отмеченное таким образом, многие не разрешают нарушать правило, поэтому важно размешать теги правильно.

В проекте все страницы построены при помощи тегов HTML и PHP.

1.2 HTML5

Первым делом следует отметить тот факт, что HTML4 был полностью закончен в конце 90–х годов, а работа над HTML5 началась относительно недавно – где–то в 2005–м. Разработчики новой версии HTML основное внимание сконцентрировали на вопросе совместимости новинки со всеми популярными на сегодняшний день типами обозревателей – как следствие, никаких революционных перемен в плане демонстрации веб–страниц не произошло, но, зато, потенциальные способности отображения контента были расширены.

К примеру, в HTML5 можно наблюдать определенное количество новых элементов – появились section, nav, header, article и footer. Напомним, что ранее существовал только один элемент – div. Стандартный img, теперь дополнен тэгами audio и video. Один из самых важных атрибутов id дополнен tabindex и repeat. Помимо этого полностью упраздняются давно вышедшие из моды элементы типа font, center и т.п.

Поскольку главный упор сделан именно на бесконфликтность новой версии языка с движками старых браузеров, то предусмотрена возможность полного игнорирования такими обозревателями правил отображения заданных в HTML5, что ни в коем случае не сказывается на качестве визуализации загруженного контента.

Как известно, старая версия полностью опиралась на элемент div. Иными словами, все, что было связанно с сайдбарами, навигационными панелями, менюшками, разделами и т.п. подчинялось именно этому элементу. В HTML5 появились альтернативные варианты, что, вне всякого сомнения, сделало структуру веб–страницы более наглядной и функциональной:

- header (не путаем с head) – можно задавать не только заголовок страницы, но и подзаголовки;
- nav – элемент предназначен исключительно для ссылок навигации, что в высшей степени удобно при создании внутри страничной навигации по сайту, а так же и для межстраничных переходов;
- section – данный элемент интерпретируется как секция общего пользования. Допустим, его можно использовать тогда, когда требуется в заранее определенном блоке текста задать специальный заголовок или параграф;
- article – посредством этого элемента можно задать независимые зоны текстового наполнения страницы. К примеру, выделение важных публикаций из архива, трансляция текущих новостей, последние сообщения на форуме и т.д. и т.п. Помимо этого, элемент article можно использовать и в обычном режиме – т.е. определять им весь контент страницы;
- footer – самый нижний элемент страницы или же "завершающий блок секции". Представляется в виде колонтитула. Там могут размещаться информационные сообщения касательно выше опубликованного контента (например, копирайт) и т.п. вещи.

Теперь несколько слов про новшества, которые касаются элементов блокового уровня. Таких новшеств несколько:

- aside – этот контейнер можно использовать для материалов, которые считаются вспомогательными. К примеру, цитаты, сноски отделенные от основного повествования, колонки с пояснительным текстом и т.п.;
- dialog – контейнер, позволяющий красиво оформить диалог;
- figure – контейнер, позволяющий описать любое графическое изображение с текстовой подписью [2].

В принципе, уже давно ожидалось, что элементы управления мультимедийным контентом рано или поздно появятся – и вот, на сегодняшний день, мы уже имеем video и audio. Основная задача этого новшества – более тесная интеграция мультимедийного контента с основной частью веб–страницы. Другой момент – посредством этих элементов отпадает потребность в использовании сторонних продуктов (плагинов) для просмотра и прослушивания мультимедийного контента.

Касательно интерактива, HTML5 тоже не остался в долгу – тут тоже появились новые элементы. Вот некоторые из них:

- details – посредством этого элемента можно паковать необязательную информацию;
- datagrid – используется для демонстрации данных таблицы. От стандартных таблиц отличается тем, что посетитель страницы может самостоятельно управлять отображением информации, т.е. убирать таблицу, перемещать колонки и т.п;
- menu, command – элементы, которые непосредственно относятся к визуальному представлению всех менюшек имеющихся на сайте.

Еще одно новшество – это элементы отвечающие за уровень текста. К примеру, в новой версии можно задавать такие параметры, как, скажем, прогресс какого–либо процесса (progress), динамическое отображение времени (time), дат и т.п.

HTML5 вводит множество API, которые должны помочь в создании Web приложений. Они могут использоваться вместе с новыми элементами:

- 2D drawing API, который может использоваться с новым элементом canvas;
- API для проигрывания видео и аудио, который может использоваться с новыми элементами video и audio;
- выделенная область памяти (Persistentstorage) с поддержкой данных в виде ключ / значение и SQL данных;
- API, который допускает автономную работу web приложений;
- API, который позволяет web приложений регистрировать себя для определенных протоколов или типов MIME;
- editing API в сочетании с новым глобальным атрибутом content editable;
- drag&drop API в сочетании с атрибутом draggable;
- network API;
- API, который выстраивает историю посещения, чтобы предотвратить нарушение функционирования back кнопки (Этот API имеет необходимые ограничения безопасности);
- cross–document messaging (Передача сообщений между документами);
- события сервера (Server–sentevents) в сочетании с новым элементом event–source [2].

HTML5 – пока еще незаконченная спецификация. Содержание HTML5, также как и содержание этого документа («HTML5 differences from HTML4»), напрямую зависящего от HTML5, всё ещё обсуждается в Рабочей группе HTML и WHATWG. Нерешенные проблемы включают в себя (список не исчерпывающий):

- семантическое определение некоторых элементов, ранее предназначенных лишь для оформления;
- сведения о доступности, возможностях замены и сопровождения альтернативным содержимым медиа данных, таких как атрибуты alt и summary.

1.3 Cascading Style Sheets – CSS

Cascading Style Sheets – каскадные таблицы стилей – это свод стилевых описаний, тех или иных HTML тегов, который может быть применён как к отдельному тегу – элементу, так и одновременно ко всем идентичным элементам на всех страницах сайта. CSS по сути своего рода дополнение к HTML, которое значительно расширяет его возможности.

До появления CSS оформление веб–страниц осуществлялось исключительно средствами HTML, непосредственно внутри содержимого документа. Однако с появлением CSS стало возможным принципиальное разделение содержания и представления документа. За счёт этого нововведения стало возможным лёгкое применение единого стиля оформления для массы схожих документов, а также быстрое изменение этого оформления.

Принятие Консорциумом W3C в декабре 1996 года в качестве стандарта CSS первого уровня был огромным шагом вперед, поскольку позволял отделить содержание веб–страницы (текст, изображение и т.д.) от ее форматирования (макет и характеристики текста, например, информация о шрифтах). После этого язык HTML снова стал функционально–ориентированным (а не ориентированным на форму), что однако не мешало пользователем контролировать вид страницы [3].

В мае 1998 года был принят стандарт CSS2, позволяющий разработчикам осуществлять контроль над веб–страницами на более высоком уровне. Этот стандарт основан на CSS первого уровня и включал новые функции, в частности, возможность точно располагать элементы и объекты веб–страницы, а также звуковые таблицы стилей, позволяющие специальному программному обеспечению считывать содержимое веб–страницы (что полезно для слабовидящих пользователей).

Существуют три способа применения стилей в документе HTML:

- написать стилевое описание непосредственно в самом элементе. Такой способ неплох, но только в том случае если таковой элемент единственный в HTML документе который требует отдельного стилевого описания;
- написать стилевое описание для всех идентичных элементов HTML документа. Такой способ применяется в случае, если стиль страницы сильно отличается от общего дизайна сайта;
- вынести стилевое описание элементов HTML в отдельный файл CSS. Этот способ помогает управлять дизайном всего сайта целиком, каждой страницей сайта на которой есть ссылка на файл стилей. На данный момент этот способ является самым эффективным использованием таблицы каскадных стилей [3].

В настоящее время современный подход к созданию сайтов подразумевает активное использование CSS для управления видом частей веб–страниц и их верстки. Довольно известная блочная верстка или верстка с помощью слоев привлекает все больше поклонников, а это предполагает знание свойств CSS и их разумное применение на сайте.

1.4 CSS 3

CSS3 – новейший стандарт веб–разработок, довольно сильно расширяющий функциональные возможности языков веб–программирования и помогающий реализовать оригинальные визуальные эффекты для ваших интернет–проектов. С помощью CSS3 действительно создавать красивые эффекты, как полупрозрачные фоны, необычные градиенты и тени; использовать оригинальные шрифты, обычно не использовавщихся в сети Интернет; внедрять на сайтах анимацию без использования Flash, предоставлять пользователям возможность унифицировать дизайн сайта без применения JavaScript.

Стандарт CSS3 до сих пор меняется и развивается – так же как обозреватели, которые в конечном итоге будут его поддерживать, и веб–дизайнеры, которым предстоит понять, как наилучшим образом применять новые спецификации для достижения своих целей [3].

CSS3 – расширение CSS 2.1 – добавляет мощную функциональность к существующим возможностям. Однако CSS3 не является теперь единой спецификацией. Вместо этого теперь мы имеем дело с несколькими модулями. Каждый модуль – это отдельная спецификация для некоторого поднабора возможностей CSS, например селекторов, текста или фонов. Каждый модуль разрабатывается собственной командой авторов в соответствии с собственным расписанием. Преимущество такого подхода в том, что для обновления всей спецификации CSS3 не приходится дожидаться, пока будет закончена работа над каким–то маленьким кусочком, модуль, включающий этот кусочек, может немного подождать, но все остальное обновляется вовремя.

Разумеется, CSS3 во многом повторяет CSS 2.1. Однако по сравнению с предыдущей версией большое количество возможностей было добавлено или пересмотрено. Приведенный далее список изменений далеко не полон – их слишком много, чтобы перечислять каждое мелкое нововведение. Представлен обзор самых хорошо поддерживаемых, популярных и полезных различий между версиями CSS 2.1 и 3.

Визуальные эффекты, не зависящие от изображений. В CSS3 вы найдете множество новых свойств, позволяющих создавать визуальные эффекты, которые раньше требовали обязательного использования изображений (и иногда написания сценариев): скругленные углы, падающие тени, полупрозрачные фоны, градиенты и изображения в качестве рамок полей. Часть этих новых свойств принадлежит модулю Background sand Borders (Фон и границы), другие вы найдете в модулях Colors (Цвета) и Image Values (Значения изображения) [3].

Трансформации полей. Еще одна категория визуальных эффектов, ставших возможными с появлением CSS3, связана с манипулированием позицией и формой поля в двумерном и трехмерном пространствах – это поворот, масштабирование, скрещивание. Такие эффекты называются

трансформациями и описаны в модулях 2D Transforms (2D Преобразование) и 3D Transforms (3D Преобразование).

Уникальные шрифты. В составе модуля Fonts (Шрифты) вы найдете правило @font–face, позволяющее создать ссылку на файл шрифта на сервере и использовать его для отображения текста на странице. Это позволяет не ограничиваться шрифтами, доступными на машинах пользователей, и значительно упрощает красивое оформление текста.

Мощные селекторы. Спецификации CSS3 включают более дюжины новых селекторов, в основном относящихся к псевдоклассам и атрибутам. Они позволяют обращаться к определенным фрагментам HTML–кода, не добавляя идентификаторы или классы, что упрощает код и защищает его от ошибок. Естественно, селекторы находятся в классе Selectors (Селекторы).

Переходы и анимация. Переходы CSS3, описанные в модуле Transitions (Переходы), представляют собой простейший тип анимации, меняющий стиль элемента. Например, это может быть плавное изменение цвета кнопки в момент, когда над ней оказывается указатель мыши. Однако возможно создание и полноценных анимационных эффектов CSS3 (с помощью возможностей, описанных в модуле Animation (Анимация)), причем для этого не требуется ни Flash, ни JavaScript.

Медиазапросы. Модуль MediaQueries (Медиазапросы) представляет синтаксис выбора разных стилей в зависимости от возможностей пользовательского дисплея или устройства, таких как ширина области просмотра, разрешение экрана и количество отображаемых цветов. Медиазапросы – это отличный инструмент для создания веб–сайтов, оптимизированных для отображения на мобильных устройствах. Многостолбцовые макеты. В CSS3 появилось несколько новых модулей, упрощающих создание многостолбцовых макетов. Модуль Multicolumn Layout (Макетмульти колонки) описывает перетекание текста, входящего в единый блок, из одного столбца в другой, как в газетных колонках. Модуль Flexible Box Layout (Модель гибкого поля) помогает выравнивать блоки по горизонтали и вертикали относительно друг друга, позволяя более гибко подгонять их под размеры области просмотра, нежели при использовании плавающих блоков или позиционирования. Также появились экспериментальные модули для работы с макетами Template Layout (Шаблонный макет) и Grid Positioning (Сетка расстановки) [3].

В проекте CSS3 используется для изображения градиентов, теней, анимации кнопок и анимации галереи.

1.5 Javascript

JavaScript – прототипно–ориентированный сценарный язык программирования. JavaScript обычно применяется как встраиваемый язык для программного обеспечения объектов приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания лучшей

визуальной обработки веб–страниц. На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на Java, но при этом лёгким для использования обычными пользователями. Языком JavaScript не владеет какая–либо компания или организация, что отличает его от ряда языков программирования, используемых в веб–разработке Первоначально язык назывался LiveScript и предназначался как для программирования на стороне клиента, так и для программирования на стороне сервера (там он должен был называться LiveWire). На синтаксис оказали влияние языки Си и Java, и, поскольку Java в то время было модным словом, 4 декабря 1995 года LiveScript переименовали в JavaScript, получив соответствующую лицензию у Sun. Анонс JavaScript со стороны представителей Netscape и Sun состоялся накануне выпуска второй бета–версии Netscape Navigator. В нём декларируется, что 28 лидирующих ИТ–компаний выразили намерение использовать в своих будущих продуктах JavaScript как объектный скриптовый язык с открытым стандартом.

В 1996 году компания Microsoft выпустила аналог языка JavaScript, названный JScript. Анонсирован этот язык был 18 июля 1996 года. Первым браузером, поддерживающим эту реализацию, был Internet Explorer 3.0 [4].

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

JavaScript является объектно–ориентированным языком, но используемое в языке прототипирование обуславливает отличия в работе с объектами по сравнению с традиционными класс–ориентированными языками. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам – функции как объекты первого класса, объекты как списки, карринг, анонимные функции, замыкания – что придаёт языку дополнительную гибкость.

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

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

JavaScript используется в клиентской части веб–приложений: клиент–серверных программ, в котором клиентом является браузер, а сервером – веб–сервер, имеющих распределённую между сервером и клиентом логику. Обмен информацией в веб–приложениях происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб–приложения являются кроссплатформенными сервисами [4].

JavaScript используется в AJAX, популярном подходе к построению интерактивных пользовательских интерфейсов веб–приложений, заключающемся в «фоновом» асинхронном обмене данными браузера с веб–

сервером. В результате, при обновлении данных веб–страница не перезагружается полностью и интерфейс веб–приложения становится быстрее, чем это происходит при традиционном подходе (без применения AJAX) [4].

В данном проекте Java–скрипты используются в модуле тестирования и модуле обратной связи.

1.6 PHP и ООП

PHP: Hypertext Preprocessor – «PHP: препроцессор гипертекста» – скриптовый язык программирования общего назначения, интенсивно применяемый для разработки веб–приложений.

PHP изначально разрабатывался как обычная замена языку Perl, и уже через несколько лет он стал чрезвычайно мощным и популярным. Язык PHP, очень похож на ANSI C.

Изучение PHP абсолютно не тяжёлое занятие, особенно если вы хорошо знакомы с синтаксисом Java или C. Так как писать PHP скрипты достаточно просто, любой может написать PHP код без соблюдения каких–либо соглашений и смешивая уровень представления с бизнес логикой (это одна из основных причин существования большого количества неуправляемых проектов). Потому что в PHP не обязательно строгое соответствие соглашений написания кода, с годами когда проект становится всё больше и больше, он превращается в громадное неуправляемое приложение [5].

ООП или объектно–ориентированное программирование давно применяется в практике программирования для наибольшего упрощения создания управляемых проектов. Специфика ООП заметно повышает эффективность труда программистов и позволяет им создавать более мощные, масштабируемые и эффективные приложения. Объектно–ориентированное программирование основано на понятиях:

- инкапсуляция – это свойство системы, которое объединяет данные и методы, работающие с ними в классе, и скрыть детали реализации от пользователя;
- наследование – это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью. Класс, от которого производится наследование, называется базовым, родительским или суперклассом. Новый класс – потомком, наследником или производным классом;
- полиморфизм – это свойство системы использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта;
- класс является описываемой на языке терминологии (пространства имён) исходного кода моделью ещё не существующей сущности (объекта). Фактически он описывает устройство объекта, являясь своего рода чертежом. Говорят, что объект – это экземпляр класса. При этом в некоторых исполняющих системах класс также может представляться некоторым

объектом при выполнении программы посредством динамической идентификации типа данных. Обычно классы разрабатывают таким образом, чтобы их объекты соответствовали объектам предметной области;

- объект – сущность в адресном пространстве вычислительной системы, появляющаяся при создании экземпляра класса или копирования прототипа (например, после запуска результатов компиляции и связывания исходного кода на выполнение) [4].

Если класс можно рассматривать как тип данных, то объект – как переменную (по аналогии). Скрипт может одновременно работать с несколькими объектами одного класса, как с несколькими переменными.

Внутри объекта данные и код (члены класса) могут быть либо открыты, либо нет. Открытые данные и члены класса являются доступными для других частей программы, которые не являются частью объекта, а вот закрытые данные и члены класса доступны только внутри этого объекта.

Процедурный подход подразумевает написание программного кода без использования объектов. Процедурное программирование заключается в написании кода с или без подпрограмм.

ООП обучает любой язык программирования более хорошему программному коду и используется, для получения более высокой производительности и написания больших проектов, не боясь запутаться в их управлении. ООП даёт вам возможность создавать объекты которые можно будет использовать многократно, для того что бы вы или другие разработчики могли использовать их в своих проектах не переделывая их снова и снова. ООП убирает барьеры и сложности в написании и управлении большими приложениями.

В последнее время идея объектно–ориентированного программирования (ООП), кардинально новая идеология написания программ, все более занимает умы программистов. Объектно–ориентированные программы более просты и мобильны, их легче модифицировать и сопровождать, чем их "традиционных" собратьев. Кроме того, похоже, сама идея объектной ориентированности при грамотном ее использовании позволяет программе быть даже более защищенной от различного рода ошибок, чем это задумывал программист в момент работы над ней. Однако ничего не дается даром: сами идеи ООП довольно трудны для восприятия "с нуля", поэтому до сих пор очень большое количество программ (различные системы Unix, Apache, Perl, да и сам PHP) все еще пишутся на старом добром "объектно–неориентированном" Си [4].

PHP до недавнего времени обеспечивал лишь некоторую поддержку ООП. Однако, после выхода PHP5.0 поддержка ООП в PHP стала практически полной.

Стратегию ООП лучше всего описать как смещение приоритетов в процессе программирования от функциональности приложения к структурам данных. Это позволяет программисту моделировать в создаваемых приложениях реальные объекты и ситуации. Технология ООП обладает тремя главными преимуществами:

- простота для понимания: ООП помогает мыслить типами повседневных объектов;
- надежность и проста для сопровождения – правильное проектирование предоставляет простоту расширения и модификации объектно–ориентированных программ. Модульная структура позволяет вносить независимые изменения в разные части программы, сводя к минимуму риск ошибок программирования;
- ускоряет цикл разработки – модульность и здесь играет важную роль, поскольку различные компоненты объектно–ориентированных программ можно легко использовать в других программах, что уменьшает избыточность кода и снижает риск внесения ошибок при копировании [5].

Основные преимущества использования ООП можно выразить как:

1 Повторное использование. Объект это логический объект у которого есть комплект свойств и методов и он может взаимодействовать с другими объектами.. Объект может быть абсолютно независимым или может зависеть от других объектов. Объект обычно создают для решения специфических поставленных проблем. Следовательно когда другие разработчики сталкиваются с похожими проблемами, они могут подключить ваш класс к своему проекту и использовать его не боясь что он нарушит процесс их разработки. Это позволяет избежать DRY, что расшифровывается как Don’t Repeat Yourself (не повторяйся). В процедурном или модульном программировании, повторное использование возможно только в совокупности.

2 Рефакторинг. Когда вам необходимо в проекте использовать рефакторинг, ООП предоставляем вам максимум преимуществ, так как все объекты это маленькие элементы и содержат свои свойства и методы как часть себя. Поэтому использовать рефакторинг относительно легко.

3 Расширяемость. Если вам необходимо расширять функциональность вашего проекта, вы можете достичь лучших результатов при помощи ООП. Одна из основных функциональностей ООП это расширяемость. Вы можете использовать рефакторинг объектов что бы добавить функциональность. Работая над этим, вы по–прежнему можете сохранить прежнюю совместимость объекта – следовательно вы можете прекрасно работать и с прежним кодом. Или же вы можете расширить объект и создать абсолютно новый, который будет содержать все необходимые свойства и методы родительского объекта от которого происходит новый, а потом уже добавить в него новые функции. Это называется “наследование” и это очень важная возможность ООП.

4 Поддержка. Объектно–ориентированный код легче поддерживать так как он следует весьма жёстким соглашениям написания кода и пишется в само поясняющейся форме. К примеру, когда разработчик дополняет, перерабатывает код, или отлаживает его, он может легко найти внутреннюю структуру кода и поддерживать код время от времени. Более того, когда в вашем окружении работает команда разработчиков ООП может быть лучшим решением так как вы можете распределять ваш код между членами команды, после разбития его на маленькие части. Эти маленькие части могут быть

разработаны как отдельные объекты, следовательно разработчики могут работать практически независимо друг от друга. В конечном итоге объеденить все части в одно приложение не составит большого труда.

5 Эффективность. Идея ООП в действительности была разработана для повышения эффективности и облегчения процесса разработки. Несколько шаблонов проектирования разработаны что бы создавать более эффективный и хороший код [5].

Более того в ООП вы можете вы можете размышлять над вашими решениями в более удобной форме чем в процедурном подходе. Поскольку вы разбиваете вашу проблему на несколько маленьких проблем и вы находите решение для каждой из них отдельно, большая проблема решается сама по себе.

Данный проект написан с помощью объектно–ориентированного подхода на языке PHP.

1.7 PHP 5.5

Спустя год и три месяца с момента выхода прошлой версии PHP представлен новый большой релиз языка программирования PHP 5.5.0. Прошлый релиз 5.4 был выпущен чуть более года назад (1 марта 2012 года), а 5.3 в 2009 году (30 июня) – трудно не заметить, что разработка PHP наращивает темп.

В версии PHP 5.5.0 добавлены новые языковые конструкции, обеспечена поддержка сопрограмм и генераторов, представлен новый API для хэширования паролей, удалены устаревшие возможности и интегрирован ZendOPcache (бывший ZendOptimizer+) [5].

Основные особенности PHP 5.5.0:

- в состав включена система кэширования и оптимизации байткода ZendOPcache (бывший ZendOptimizer+), ранее поставляемая в составе про приетарного пакета ZendServer, но недавно переведённая компанией ZendTechnologies в разряд свободных проектов. ZendOPcache предоставляет средства для ускорения выполнения скриптов на языке PHP за счёт использования техники кэширования опкода и применения дополнительных оптимизаций. Предкомпилированный код скриптов кэшируется в разделяемой памяти, что позволяет избежать выполнения стадий чтения с диска, разбора и компиляции исходного кода скриптов при их повторном запуске. Кроме того, ZendOPcache включает дополнительные методы оптимизации при генерации байткода, что позволяет достичь ускорения и за счёт более быстрого выполнения байткода. По сравнению с конкурирующей системой APC, ZendOPcache выигрывает по производительности в среднем на 5–20%, обеспечивает лучшую совместимость с различными ветками и возможностями языка PHP, а также содержит средства для выявления повреждения данных (например, из–за некорректно написанной PHP–функции на языке C);
- добавлен новый API для хэширования паролей, предоставляющий застрахованные от ошибок разработчиков и более простые в использовании

высокоуровневые функции для генерации и проверки валидности паролей по хэшам. Основное отличие нового API в том, что он берёт на себя генерацию надёжных хэшей, скрывая от разработчика операции ручного указания salt–а и выбора алгоритма хэширования (по умолчанию используется Bcrypt). Создание хэша сведено к выполнению "$hash = password_hash ($password, PASSWORD_DEFAULT);", а проверка к вызову "password_verify ($password, $hash)". В качестве причины внедрения нового API называется безалаберное отношение многих разработчиков к генерации salt–ов и повсеместный выбор нестойких к перебору алгоритмов хэширования;

- поддержка сопрограмм (coroutine) и генераторов (generator), предоставляющих простой и не требующий создания шаблонов способ реализации итераторов;
- поддержка ключевого слова "finally", расширяющего возможности блоков исключений try/catch и выполняемого после завершения выполнения блока try, что гарантирует, что заданный в секции finally код будет выполнен в последнюю очередь после всех остальных обработчиков исключений;
- поддержка указания list() внутри блока foreach (), что позволяет организовать не требующий создания отдельных временных переменных перебор масивов (например, "foreach ($arrayaslist($a, $b)){...}");
- возможность разыменования констант для строк и массивов (например, "echo 'PHP'[0]" или "echo [1, 2, 3][0]");
- поддержкаразрешения имён скалярных классов через ключевое слово "::class" (например, "echoClassName::class");
- библиотека GD обновлена до версии 2.1, в которой добавлены новые функции (например, imageflip), улучшена работа доступных ранее возможностей (imagecrop и imagecropauto) и добавлена поддержка формата WebP (функции imagecreatefromwebp, imagewebp);
- оптимизация доступа к временным и компилированным переменным на уровне VM, что позволило сократить число обращений к памяти на 8%;
- в php.ini добавлена опция для изменения пути к временной директории PHP;
- реализованы все опции mysqli_commit()/mysqli_rollback(), которые могут быть использованы с START TRANSACTION, COMMIT и ROLLBACK в MySQL 5.6;
- поддержка работы SAPI–обработчика Apache 2.4 на платформе Windows;
- изменения, нарушающие совместимость:
  - реализация игнорирования регистра символов в именах функций, классов и констант теперь не привязывается к локали, а определяется в соответствии с правилами ASCII;
  - прекращение поддержки Windows XP и Windows 2003;
  - прекращение поддержки устаревших функций: php_logo_guid(), php_egg_logo_guid(), php_real_logo_guid(), zend_logo_guid();

- особенности работы функций pack() и unpack() приведены в соответствие с реализацией из состава Perl;
- расширение ext/mysql объявлено устаревшим, вместо него следует использовать MySQLi или PDO_MySQL.

В PHP 5.5 много хороших новинок, которые пригодятся при разработке. В добавок ко всему этому было исправлено куча ошибок и багов, возникших после предыдущих релизов. Так же, разработчики языка хорошенько поработали над улучшением производительности.

1.8 Основные этапы разработки современных веб–сайтов

1.8.1 Проектирование

Проектирование – это один из важнейших и тяжелых этапов разработки сайта, от которого зависит результативность дальнейших работ и конечный результат. Тем не менее, многие веб–студии не уделяют этому этапу разработки должного внимания или даже вовсе не занимаются подготовкой нужной документации. Неудивительно, что при таком подходе сайты не оправдывают ожиданий заказчиков, а финансовые затраты на их разработку не приносят никакой прибыли [1].

Невозможно разработать хороший сайт без серьезного проектного исследования, так же как невозможно построить хороший дом без надлежащего архитектурного плана. Нет, можно, конечно же, прибежать на строительную площадку, наспех замесить раствор, кое–как прилепить друг к другу кирпичи и побросать на них сверху панели. Все это вполне реально. Только не стоит потом удивляться, почему некоторые комнаты получились «глухими», штукатурка осыпается со стен, а попасть внутрь дома можно только через окно.

Также ничуть не лучше выглядит ситуация, когда проектная документация все–таки имеется, но в процессе строительства вдруг выясняется, что нарисованное стоять не может, поскольку в архитектурном плане забыли предусмотреть фундамент. Само собой разумеется, что деревянные подпорки в критических местах не будут являться решением проблемы.

Аналогичным образом могут обстоять дела и с разработкой сайта. Если изначально не определить назначение и цели проекта, не разобраться с нуждами целевой аудитории и не регламентировать структуру и логику взаимодействия сайта с пользователями, вряд ли можно надеяться, что в конечном итоге будет разработан хорошо организованный сайт, который будет эффективно решать поставленные задачи. Да и как вообще сайт может что–то решать, если это «что–то» нигде не оговаривается и не регламентируется?

Этапы проектирования. Процесс проектирования сайта можно разделить на несколько основных этапов. При этом каждый последующий этап очень сильно зависит от предыдущего. Другими словами, пересмотр какого–то решения, принятого на более раннем этапе, в большинстве случаев влечет за собой необходимость переосмысления других решений, принятых на более

поздних этапах. Именно по этой причине в первую очередь необходимо рассмотреть маркетинговые аспекты: для чего и для кого разрабатывается сайт.

Процесс проектирования должен начинаться с анализа стратегических целей разработки сайта. В каждом конкретном случае эти цели могут быть совершенно разными. Например, при создании корпоративного сайта могут преследоваться следующие цели:

- привлечение новых клиентов и увеличение объемов продаж;
- привлечение новых дилеров, инвесторов или сотрудников;
- информационная поддержка клиентов и деловых партнеров;
- формирование благоприятного имиджа компании.

В то же время для промо–сайтов цели могут быть совершенно иными:

- реклама товара или предоставляемой услуги;
- повышение узнаваемости торговой марки;
- повышение лояльности целевой аудитории к брэнду;
- информационная поддержка кандидата в депутаты.

При создании интернет–магазина цели могут варьироваться от «формирования новых каналов сбыта» и до «ведения бизнеса исключительно с помощью сайта», но в любом случае каждой цели можно сопоставить одну или несколько конкретных задач, способы решения которых должны быть представлены на этапе общего описания проекта [1].

Сегментация целевой аудитории. На данном этапе проектирования необходимо выделить группы потенциальных посетителей будущего ресурса. При этом следует учитывать довольно широкий диапазон различных характеристик: возрастные, профессиональные, демографические, социальные и другие данные. Иначе говоря, для того, чтобы сайт мог эффективно выполнять возложенные на него задачи, очень важно знать, для каких категорий пользователей он создается. Плохое понимание нужд потенциальных посетителей может привести к тому, что даже весьма привлекательный и функциональный сайт не сможет привлечь внимание необходимой аудитории.

Информация о целевой аудитории непосредственно влияет на дальнейшие этапы проектирования. К примеру, на этапе информационного дизайна она позволяет спроектировать такой пользовательский интерфейс, который будет учитывать степень компьютерной грамотности посетителей сайта. Согласитесь, что подросток, который хочет приобрести игровую приставку, домохозяйка, которая желает обсудить особенности приготовления любимого блюда, и веб–разработчик, который занимается поиском определенной технической документации, – это совершенно разные категории пользователей с разным уровнем знаний в области интернет–технологий. Так или иначе, только после четкого определения целей разработки сайта, а также тщательного анализа целевой аудитории можно абстрактно описать предстоящий проект и сформулировать ряд специфических требований к его реализации.

Общее описание проекта основывается на данных, полученных в результате выполнения двух предыдущих этапов проектирования. Оно должно содержать обобщенную информацию о том, что именно будет получено в результате разработки сайта и каким образом будут достигнуты поставленные цели.

В первую очередь необходимо кратко описать, какие текстовые и графические материалы должен содержать сайт. При этом следует четко обозначить, с какой целью и для реализации каких задач размещается та или иная информация. Подобный подход позволяет избежать информационной перегруженности и гарантирует, что на сайте будет представлена только актуальная и действительно необходимая целевой аудитории информация [1].

Также на данном этапе проектирования принимается решение о необходимости разработки различных программных компонентов и модулей для сайта. Поскольку каждый программный компонент и модуль является средством реализации определенных задач, необходимо кратко описать его предназначение. Например, в случае проектирования интернет–магазина описание проекта должно содержать общую информацию о том, каким именно образом будет организован процесс продаж.

Требования к сайту. В процессе проектирования любого сайта невозможно обойтись без определения некоторых специфических требований к проекту. Эти требования опираются на данные, полученные в результате выполнения предыдущих этапов проектирования и оказывают непосредственное влияние на последующие этапы проектирования и разработки сайта:

1 Требования к графическому дизайну регламентируют основные принципы, которых необходимо придерживаться при разработке визуального оформления сайта. Если проектируется корпоративный сайт, данная группа требований может содержать образцы элементов фирменного стиля компании (логотип, цветовая гамма, шрифты). При этом, как правило, указывается, каким образом эти элементы должны интегрироваться в общую концепцию дизайна. На выбор цветового решения и общей стилистики очень сильно влияет информация о назначении сайта и его целевой аудитории. Визуальное оформление обязательно должно соответствовать маркетинговым аспектам проекта. Неверный выбор приоритетов в дизайне может легко подорвать репутацию солидной компании или спровоцировать негативное отношение аудитории к презентационному или тематическому интернет–ресурсу.

2 Требования к эргономике. Эргономика – это научно–прикладная дисциплина, занимающаяся изучением и проектированием эффективных и удобных систем. Поскольку любой сайт является интерактивной информационной системой, к нему в полной мере могут предъявляться требования, касающиеся удобства его использования. Требования к эргономике оказывают непосредственное влияние на структуру и информационный дизайн сайта. В частности, они определяют, насколько сложным позволительно

сделать пользовательский интерфейс, основываясь на данных о целевой аудитории сайта.

3 Требования по стандартизации и унификации относятся к верстке веб–страниц. Они регламентируют стандарты, которым должны соответствовать страницы сайта, а также определяют перечень графических браузеров, в которых сайт должен отображаться корректно. Если помимо графических браузеров планируется поддержка других устройств вывода (принтеры, сотовые телефоны, речевые браузеры), в данной группе требований должен быть указан полный перечень таких устройств. При этом необходимо описать специфику представления сайта на каждом альтернативном устройстве.

4 Требования к лингвистическому обеспечению зависят от определенной на этапе общего описания проекта функциональности сайта и регламентируют языковые платформы и технологии, которые должны использоваться в процессе разработки программных компонентов и модулей. Во избежание дорогостоящих ошибок следует очень внимательно проанализировать технологические аспекты реализации проекта. Если в будущем возникнет необходимость расширить функциональные возможности сайта, плохо выбранные технологии могут существенно усложнить процесс модернизации.

5 Требования к системе управления. Если для сайта предусматривается система управления, данная группа требований должна регламентировать ее функциональность. Необходимо четко определить, какие возможности система управления должна предоставлять в распоряжение администратора и какие действия сможет выполнять администратор с ее помощью. Помимо этого, необходимо рассмотреть вопросы надежности и безопасности системы управления. Как правило, минимальные требования к механизмам безопасности подразумевают наличие защиты от несанкционированного доступа и распространенных типов атак (SQL–injection, XSS).

6 Требования к эксплуатации. Используемые в процессе реализации проекта языковые платформы и технологии могут предъявлять определенные технические требования к конфигурации веб–сервера. Другими словами, для размещения готового сайта в сети интернет может подойти далеко не каждая хостинговая площадка. Требования к эксплуатации сайта регламентируют аппаратную и программную конфигурацию веб–сервера, которая будет обеспечивать надлежащее функционирование будущего интернет–ресурса.

После определения всех необходимых требований можно приступить к следующему этапу проектирования – визуальному моделированию структуры сайта.

Структура сайта. На данном этапе проектирования необходимо разработать концептуальную схему, отражающую информационную архитектуру и интерактивные процессы сайта.

Информационная архитектура определяет порядок расположения материалов сайта относительно друг друга. В общем случае необходимо представить будущий сайт в виде книги и рассортировать материалы по разделам и подразделам. При этом следует тщательно продумать структуру и

названия разделов и подразделов, так как от этого будет зависеть количество времени, которое будут тратить пользователи сайта на поиск необходимой им информации.

Другими словами, информационная архитектура должна формировать максимально доступное и интуитивно понятное информационное пространство для целевой аудитории сайта.

Интерактивные процессы подразумевают выполнение пользователями каких–то определенных действий (регистрация на сайте, покупка товара в интернет–магазине, публикация комментария к статье и т.п.).

В данном контексте необходимо определить, на каких веб–страницах пользователи смогут совершать подобные действия, каким образом сайт будет реагировать на эти действия и куда будут перенаправляться пользователи после того, как эти действия будут совершены.

Графическое представление структуры сайта позволяет в некоторой степени конкретизировать абстрактные данные, полученные на этапе общего описания проекта, и служит основой для дальнейшей детализации проекта.

Информационный дизайн регламентирует наличие и расположение информационных и функциональных блоков на страницах сайта. При этом преследуется цель максимально оптимизировать расположение элементов на веб–страницах с учетом информации о целевой аудитории.

Кроме того, информационный дизайн подразумевает проектирование пользовательского интерфейса. В данном контексте необходимо позаботиться о том, чтобы навигационные элементы позволяли легко перемещаться по информационной архитектуре сайта и не вызывали у посетителей дискомфорта. Для успешного решения этой задачи следует руководствоваться требованиями к эргономике, которые определяются на одном из предыдущих этапов проектирования.

Важно отметить, что информационный дизайн практически никак не связан с визуальным оформлением веб–страниц. На данном этапе проектирования необходимо всего лишь определить, какие элементы должны присутствовать на веб–страницах, в какой последовательности они должны располагаться и как они могут использоваться.

Описание контента. После того, как будет определена микроструктура каждой отдельной страницы сайта, необходимо составить сводный перечень всех веб–страниц с указанием всех текстовых и графических материалов, которые предполагается на этих веб–страницах разместить. Для реализации этой задачи рекомендуется представить перечень страниц в виде таблицы.

В процессе описания контента очень важно четко обозначить, кто будет заниматься подготовкой необходимых материалов. Как правило, различные тематические статьи и фотоматериалы предоставляются заказчиком проекта, а многие графические иллюстрации (например, схема проезда к офису компании) и краткие пояснения к интерактивным формам могут разрабатываться специалистами веб–студии.

Функциональность сайта. Данный этап проектирования регламентирует алгоритмы работы программных компонентов и модулей сайта. Необходимо подробно описать функциональность каждого компонента и модуля, порядок взаимодействия пользователей с интерактивными формами, правила обработки нештатных ситуаций, а также множество других технологических аспектов.

Подробное описание функциональности особенно актуально для крупных интернет–проектов, требующих интенсивного программирования. Другими словами, чем сложнее будущий сайт с технологической точки зрения, тем более важное значение приобретает наличие функциональной спецификации.

Техническое задание. Результат проектирования выражается в специальном документе, который называется «Техническое задание» и является неотъемлемой частью договора на разработку сайта. Требования к составлению технического задания регламентируются ГОСТ 34.602–89 [7].

Прежде всего проектная документация позволяет определить сроки и стоимость разработки сайта. Если вы обратились в веб–студию, где вам сразу же называют точную стоимость реализации проекта, знайте, что эта стоимость взята «с потолка». Без предварительного проектного исследования оценить объем предстоящих работ не представляется возможным.

Также весьма занятно выглядят предложения некоторых веб–студий о разработке «профсайтов» бесплатно. Если вы поинтересуетесь в любой из таких компаний насчет подготовки и утверждения проектной документации, вы сразу же поймете, что там о таких «премудростях» никто даже и не слышал, а за громким названием «Mega Design Studio» скрывается все тот же дядя Вася.

Техническое задание – это руководство к действию для проект–менеджера, дизайнера, верстальщика, программиста, контент–менеджера, но в первую очередь этот документ необходим заказчику проекта. С одной стороны, проектная документация помогает заказчику лучше понять, осмыслить и сформулировать свои собственные потребности, цели и задачи. С другой стороны, она позволяет оценить качество выполнения каждого этапа работ и конечный результат. Иначе говоря, без технического задания невозможно определить конечную точку реализации проекта в принципе.

Как правило, на проведение серьезных проектных исследований выделяется отдельный бюджет, никак не связанный с дальнейшими этапами разработки сайта. Тем не менее, в некоторых случаях мы придерживаемся политики, которая не предусматривает внесения оплаты за проведение проектных работ. В особенности это касается малобюджетных сайтов с небольшим объемом контента.

1.8.2 Разработка дизайна сайта

Графический дизайн сайта – это визуальное оформление веб–страниц. Он играет такую же роль для интернет–ресурса, как полиграфический дизайн для печатного издания. Тем не менее, критерии качества сайта далеко не всегда совпадают с критериями качества полиграфической продукции.

В первую очередь это происходит оттого, что визуальный дизайнер при выполнении своей работы должен руководствоваться не только требованиями к графическому дизайну, но также и требованиями к эргономике.

Эстетика и эргономика. Вообще, в русском языке под термином «дизайн» обычно понимается что–то близкое к изобразительному искусству, а профессия дизайнера в первую очередь ассоциируется с профессией художника. В принципе, в подобных суждениях нет ничего предосудительного, по крайней мере в отношении графического дизайна. Задача визуального дизайнера как раз и заключается в разработке общей художественной концепции сайта: выбор гармоничного цветового решения, создание коллажей, сочинение и эффективное использование графических образов и метафор. Вся эта работа должна основываться на знании теории цвета, баланса композиции, а также многих других фундаментальных принципов дизайна [1].

Однако, необходимо понимать, что безупречный графический дизайн вовсе не означает, что посетителям сайта будет легко и удобно этим сайтом пользоваться. Другими словами, сайт, который с эстетической точки зрения выглядит вполне прилично, но при этом запутывает посетителя, – это плохой сайт. Нередки случаи, когда чрезмерное увлечение визуальным оформлением приводит дизайнера к тому, что он напрочь забывает о другом не менее важном аспекте – эргономике сайта. Иногда это происходит и по инициативе заказчика проекта, который совершенно не осведомлен о том, что посетителям сайта в первую очередь требуется не поражающий воображение анимационный эффект, а простой и удобный доступ к необходимой информации.

Аналогом понятия «эргономика» является современный термин «юзабилити», который означает разработку дизайна, ориентированного на пользователей сайта (usability – практичность, удобство и простота использования). С точки зрения «юзабилити» главное предназначение графического дизайна – способствовать более легкой ориентации посетителей в информационном пространстве сайта и помогать им эффективно воспринимать предоставляемую информацию. Кроме того, визуальное оформление ни в коем случае не должно приводить к быстрой утомляемости посетителей при работе с веб–сайтом.

Таким образом получается, что профессионализм визуального дизайнера в значительной мере определяется умением отыскать должный баланс между эстетикой и эргономикой. И это действительно весьма непростая задача, поскольку визуальная привлекательность и удобство использования сайта редко когда находят точки соприкосновения. Нечасто можно встретить сайт, который был бы в одинаковой степени привлекательным (как этого хотел визуальный дизайнер) и эргономичным (как это планировал информационный дизайнер).

Помимо всего прочего, в процессе поиска компромисса между эстетикой и эргономикой очень большую роль играет назначение сайта, а также его целевая аудитория. Хороший интернет–магазин, для которого самым важным фактором является удобство его использования, никогда не будет являться

хорошим презентационным сайтом, который, в свою очередь, совсем не обязан быть удобным для всех без исключения посетителей. Иначе говоря, универсально хорошего дизайна не существует. Многим сайтам жизненно необходим яркий графический дизайн, который зачастую несколько противоречит принципам «юзабилити», в то время как для подавляющего большинства информационных и тематических ресурсов гораздо важнее эргономичность.

Принципы «юзабилити» требуют от сайтов максимальной простоты. Это означает, что в процессе разработки графического дизайна следует руководствоваться устоявшейся практикой и применять исключительно типовые приемы и решения. С точки зрения эргономики преимущества такого подхода вполне очевидны: любому пользователю все будет предельно ясно и понятно [1].

Означает ли это, что ни о какой индивидуальности, неповторимости и оригинальности в этом случае не может быть и речи? К сожалению, да. Тем не менее, вопреки всему вышеизложенному мы придерживаемся мнения, согласно которому дизайн сайта обязательно должен быть креативным и содержать эксклюзивные графические элементы. Причем даже в тех случаях, когда подобный подход в некоторой степени противоречит принципам «юзабилити».

На сегодняшний день в сети интернет и так вполне достаточно совершенно невыразительных и безвкусно оформленных ресурсов, похожих друг на друга как две капли воды. В данном контексте следование стандартным и распространенным правилам в дизайне может означать только одно: содействие засилью посредственности.

Еще один фактор, который оказывает непосредственное влияние на нашу точку зрения, – это глобальная модернизация аппаратного обеспечения Всемирной Сети. Прошло то время, когда подавляющее большинство пользователей сети интернет имели в своем распоряжении только лишь телефонные модемы. Как следствие, сейчас уже нет никакой необходимости заниматься усиленной оптимизацией сайтов под медленные каналы передачи данных. Это, кстати, вовсе не означает, что размер веб–страниц должен исчисляться мегабайтами, но излишняя простота тоже не делает сайт привлекательным. Встречают, как известно, «по одежке» и только лишь провожают по уму.

Если сайт не имеет яркого и запоминающегося визуального оформления – это всего лишь голая схема, которая вряд ли сможет привлечь внимание посетителей и вызвать у них какие–то эмоции. С другой стороны, чистая и контрастная графика с большим количеством бликов и неестественно красивыми тенями способна создать определенное настроение, воззвать к чувствам и вызвать живой отклик у зрителей. Другими словами, именно визуальная составляющая формирует у посетителей сайта первое впечатление о его владельце. Все это особенно актуально для промо–сайтов, а также для корпоративных проектов с ярко выраженной имиджевой функцией.

Для того, чтобы разработать такой дизайн, нужно обладать определенными знаниями и умениями. Более того, очень часто в процессе разработки концепции дизайна требуется участие художников и профессиональных фотографов. И только благодаря стараниям команды высококвалифицированных специалистов рождаются поистине яркие и уникальные сайты.

1.8.3 Верстка сайта

Верстка сайта – это своеобразное преобразование графических макетов веб–страниц в статические электронные документы, которые составляются с помощью специального языка разметки гипертекста HTML, а затем оформляются при помощи стилевых спецификаций CSS. Другими словами, верстальщик «оживляет» полученное на этапе разработки дизайна графическое изображение сайта: расставляет элементы веб–страниц по своим местам, заставляет работать гиперссылки, навигационные меню, интерактивные формы и другие элементы пользовательского интерфейса. В конечном итоге сверстанные статические прототипы веб–страниц становятся доступными для просмотра в любом из современных браузеров (специальных программ, предназначенных для поиска и просмотра сайтов) [1].

Веб–стандарты. В отличие от других этапов разработки сайта, к верстке веб–страниц предъявляются несколько специфических требований и рекомендаций, которые разработаны и официально утверждены одной из самых авторитетных организаций в области стандартизации Всемирной Сети – Консорциумом W3C. В обиходе все эти требования и рекомендации часто называют «веб–стандартами».

С момента своего образования в 1994 году Консорциум W3C проделал огромную работу, выпустив более 80 технических спецификаций и рекомендаций. Одним из таких стандартов как раз и является язык разметки гипертекста HTML. Официальный документ, определяющий данный стандарт, содержит полный перечень допустимых конструкций языка разметки гипертекста, а также подробно описывает их синтаксис и семантику. Несмотря на обилие технических терминов, на самом деле все не так уж и сложно. По крайней мере, понятия «синтаксис» и «семантика» должны быть нам знакомы еще со школьных времен.

Синтаксическая корректность. Язык разметки гипертекста, как и любой обычный язык (русский, английский, албанский), имеет свой синтаксис (правила правописания). Несложно догадаться, что именно оттого, насколько грамотно составлен электронный документ, будет зависеть его правильное отображение в браузерах.

Обратите внимание, что именно «браузерах», а не «браузере», поскольку браузеры бывают разные: Internet Explorer, Mozilla Firefox, Opera, Safari, Google Chrome, а также некоторые другие, менее популярные. Поскольку любой владелец сайта в первую очередь заинтересован в увеличении посещаемости

своего ресурса, вряд ли его порадует известие о том, что значительная доля посетителей сайта пользуется браузером, в котором сайт отображается совершенно непрезентабельно из–за синтаксических ошибок верстки. По этой причине первоочередной задачей верстальщика является обеспечение правильного и идентичного отображения сайта во всех современных браузерах, что может быть достигнуто только в случае грамотной (синтаксически корректной) верстки веб–страниц.

Кроме того, поскольку браузер является программным обеспечением, ему свойственно время от времени «обновляться». Иначе говоря, прогресс не стоит на месте и производители браузеров периодически выпускают новые версии своих программных продуктов. Иногда это приводит к тому, что корректно отображающийся в текущей версии браузера сайт начинает отображаться в новой версии этого браузера совершенно по–иному. В данном контексте синтаксическая корректность верстки сайта позволяет обрести уверенность в том, что сайт будет правильно обработан и отображен большинством браузеров не только нынешнего, но и будущих поколений.

Семантическая корректность. Небезызвестно, что любой документ, в том числе и электронный, имеет определенную структуру и может состоять из множества различных элементов. В самом простом случае такими элементами являются заголовки, подзаголовки и абзацы с текстом. В электронных документах все эти элементы должны особым образом размечаться с помощью специальных конструкций языка HTML.

Семантическая корректность верстки подразумевает грамотное использование конструкций языка HTML не с точки зрения синтаксиса, а с точки зрения их смыслового назначения (семантики). Проще говоря, заголовки в электронных документах должны размечаться с помощью HTML–конструкций, предназначенных для разметки заголовков, подзаголовки – с помощью HTML–конструкций, предназначенных для разметки подзаголовков, а абзацы с текстом – с помощью HTML–конструкций, предназначенных для разметки абзацев. В принципе, логично, не так ли? Тем не менее, подавляющее большинство сайтов в сети интернет сверстаны с грубыми нарушениями семантических требований стандарта HTML. Существует множество исторических причин, по которым произошло подобное безобразие, но это уже слишком технический вопрос. При желании вы можете найти более подробную информацию по этой теме на специальном тематическом проекте webstandards.org.ru, посвященном разработке веб–сайтов на основе современных веб–стандартов.

Если же не вдаваться в различные технические подробности, можно попросту отметить, что семантическая корректность имеет ряд бесспорных достоинств и преимуществ, главное из которых – лучшая индексация сайтов поисковыми роботами. Другими словами, грамотная семантическая верстка изначально способствует успешному продвижению сайта в поисковых системах. Кроме того, семантически корректные документы быстрее загружаются и могут успешно отображаться не только на мониторах

персональных компьютеров, но и на любых других устройствах вывода (мобильные телефоны, КПК, цифровые проекторы).

1.8.4 Программирование сайта

После того, как страницы сайта будут сверстаны, теоретически их уже можно разместить в сети интернет. Если при этом связать эти страницы соответствующими гиперссылками, в результате может быть получен простейший статический сайт. Тем не менее, при таком подходе рано или поздно возникает ряд определенных проблем, связанных с дальнейшей эксплуатацией и поддержкой такого интернет–ресурса.

В первую очередь необходимо отметить, что для внесения каких–либо изменений в статический сайт обязательно потребуется знание языка разметки гипертекста HTML. К примеру, для того, чтобы опубликовать на сайте какой–либо новый материал (новость, статью и т.п.) необходимо будет каждый раз создавать отдельный HTML–документ, а затем вручную проставлять на него соответствующие гиперссылки на всех остальных веб–страницах.

Само собой разумеется, что такое решение вряд ли покажется привлекательным любому владельцу сайта вне зависимости от его компетентности в сфере веб–технологий. Кроме того, с помощью одних только средств HTML невозможно реализовать интерактивные формы, ленты новостей, опросы, гостевые книги, не говоря уже о более серьезной функциональности, такой как форум или электронная торговля.

Все вышеизложенное наводит на мысль о том, что для эффективной разработки и последующей поддержки сайта необходима специальная система.

Аббревиатура CMS расшифровывается как «Content Management Software» (программное обеспечение для управления содержимым). В нашей стране принято последнюю букву «S» расшифровывать как «System», а по–русски это обычно звучит как «Система управления контентом». Иногда употребляется более простое название – "движок сайта".

Основной задачей такой системы является сбор и объединение в единое целое, на основе ролей и задач, различных источников информации. Эти источники могут быть доступны как внутри самой организации, так и вне её пределов. К тому же данная система обеспечивает возможность взаимодействия различных сотрудников, проектов и рабочих групп, с теми базами знаний и данных, которые были ранее созданы, в таком виде и таким способом, чтобы сделать процесс поиска и повторного использования максимально комфортным и привычным.

В такой системе управления контентом определяется всё многообразие существующих данных: стандартные документы, музыка и звуки, видео, каталоги всевозможной информации и многое–многое другое. И именно для управления, хранения, обработки, просмотра и публикации таких данных различными группами пользователей и служат CMS. Отсюда кстати возникает

и новый род профессиональной деятельности – контент менеджер, или проще говоря – редактор сайта.

Если смотреть с точки зрения обычного заказчика, то разработка сайта на основе какой–либо CMS должна приносить следующие преимущества:

- в работе используется наиболее эффективный инструмент для решения конкретной задачи (в зависимости от вида сайта и требований к его функционалу подбирают оптимальную CMS);
- использование CMS позволяет владельцу сайта самостоятельно создавать и удалять разделы сайта, редактировать различную информацию без привлечения стороннего специалиста – это одно из преимуществ над статическими сайтами;
- работа сайта постоянно тестируется множеством пользователем, а найденные ошибки и уязвимости достаточно оперативно устраняются, при этом сайт работает на самых передовых и проверенных технических решениях;
- временные затраты на разработку сайта существенно снижаются, так как разработчику не надо фиксировать своё внимание на чисто технических задачах: «как сделать ленту с новостями» или «как научить CMS искать товары в каталоге», а можно сосредоточиться на информационной и визуальной составляющих будущего сайта.

Некоторые системы ориентированы только на решение конкретных задач (ведение блогов, интернет магазины, форумы), другие являются универсальными и предоставляют разработчиком удобную среду проектирования и программирования для разработки чего угодно. Часть CMS состоять из множества функциональных блоков и модулей, другие монолитны, неделимы, да ещё и зашифрованы. Одни системы поставляются бесплатно и с возможностью внесения своих доработок, а некоторые предоставляются за деньги и не допускают возможность редактирования ядра «движка».

Важное замечание: бесплатность CMS отнюдь не означает низкое качество кода или наличие ограничений в использовании. Наоборот, чаще всего, отрытые и бесплатные CMS во многом выигрывают у своих коммерческих собратьев именно из–за своей общедоступности. Над этими системами трудятся не только коллективы профессиональных разработчиков, но и сотни и даже тысячи добровольцев, которые стремятся сделать любимую CMS ещё лучше и безопасней.

Сложно сделать правильный выбор среди такого разнообразия, поэтому необходимо чётко осознавать назначение и функционал будущего сайта. К примеру, если заказчику нужен всего лишь небольшой личный блог, то нет смысла ему приобретать «1С–Битрикс – корпоративное решение» за 100 с лишним тысяч рублей. Поэтому задача разработчика сайта – не только в том, чтобы сделать сайт точно в срок и с учётом пожеланий заказчика, но и выбрать правильную и достаточную CMS для этих целей.

1.8.5 Наполнение сайта

После того, как макет сайта привязан к системе управления сайтом CMS, можно приступать к заполнению сайта контентом: публиковать тексты, картинки, таблицы, публиковать новости и многое другое. От того, как хорошо продуманно размещается информация на страницах сайта сильно зависит его продвижение в сети интернет. В настоящее время поисковые машины пытаются сделать поиск качественней, они стремятся показывать пользователю в результатах поиска список только качественных веб–сайтов. Качественный сайт – это несколько основных моментов:

1 Сайт создан с учетом стандартов и корректно работает во всех обозревателях.

2 Сайт содержит только унифицированный авторский контент (то есть информация на сайте не скопирован с других ресурсов и считается единственным).

3 Пользователь сайта получает насколько можно больше полезной информации (то есть объем тоже важен – тексты должны быть грамотно составлены и максимально раскрывать тему).

Вы всегда сами можете напрячься и самостоятельно создать грамотный контент для вашего веб–ресурса.

1.8.6 Тестирование

На этом этапе проверяется все: удобство навигации, целостность данных, корректность ссылок и орфография. Лучше, чтобы тестирование проводили не сами разработчики сайта, а другие люди. В процессе работы над сайтом насколько к нему привыкаешь, что даже откровенные ляпы не замечаются. Многим известно, что тестирование любой программы, в том числе и сайта, разбивается на несколько этапов. Первый, называемый альфа–версия – ошибки еще замечаются самими разработчиками. Когда разработчики ошибки перестают замечать, продукт превращается в бета–версию. Вот тогда и надо его передавать другим людям на проверку. Все найденные ошибки и замечания, естественно, исправляются.

1.8.7 Размещение сайта в интернете

Для размещения сайта в интернете нужно решить две задачи. Первая задача состоит в выборе адреса сайта или, более верно, его имени (домена). В большей части случаев самый очевидный и правильный способ состоит в использовании домена, совпадающего с названием компании или торговой марки. Так, например, компания называется "Инфо–Эксперт", то и название домена было выбрано info–expert.ru.

Вторая задача – это выбор физического размещения вашего сайта. Понятие "опубликовать сайт в Интернет" на самом деле означает простую вещь

– кто–то имеющий свой сервер постоянно подключенным к Интернет предоставляет вам место на этом компьютере, чтобы ваш сайт стал доступен для посетителей. Эта услуга называется "хостингом".

Самое интересное состоит в том, что реальное расположение сервера не имеет никакого значения. Вы можете иметь русскоязычный сайт с русским доменом, а сервер, например, реально может располагаться в США. Это никак не помешает вашим посетителям, поскольку они будут искать ваш сайт по имени, а не по физическому расположению.

1.8.8 Раскрутка и поддержка

Рекламная компания по узнаванию сайта и повышения его посещаемости. Сюда входит регистрация сайта в поисковых системах, обмен ссылками, баннерная реклама и др. Продвижение сайта важно, поскольку о нем никто не узнает, пока об этом громко, даже не заявить, а крикнуть на весь Интернет: "Есть такой сайт!".

Еще на этапе проектирования необходимо задаться вопросом, каким образом будут добавляться новые разделы и материалы, что будет происходить со старыми. Возможно, потребуется создание архива новостей, куда будут попадать новости, потерявшие свою актуальность. Еще более важным является постоянное обновление информации на сайте. Насколько часто будет происходить обновление (рекомендуется не реже одного раза в 2 недели), кто и как будет это делать. Как ни грубо это звучит, но необходимо прикармливать своих читателей, чтобы они вновь и вновь заходили к вам на сайт. Как это делать зависит только от вас. Приведенные этапы являются типичными при разработке сайтов любой направленности – от образовательных ресурсов до информационных порталов.

2 Разработка виртуального музея истории развития компьютерной техники

2.1 Постановка задачи

Тема выпускной работы выбрана в целях повышения уровня знаний посетителей сайта об истории развития компьютерной техники, а также о современных разработках в данной области. Мой сайт включает следующие модули:

- Информационный контент – это совокупность электронных страниц, на которых опубликована различная информация: публикация новостей, исторических событий, развернутой информации о современных разработках в области программного и аппаратного обеспечения. К информационному контенту относятся следующие электронные страницы проекта: главная страница, «Ученые» и все основные категории.
- Галерея – иллюстрации в формате GIF или JPEG с увеличением при наведении курсора мыши и кратким описанием. На странице публикуется до 20 предпросмотров.
- Модуль «Тестирования» – это специализированный модуль, который позволяет пользователям сайта, пройти простой тест и узнать свой уровень знаний по определенной теме.
- Модуль Обратной связи – предоставляет возможность пользователю оценить сайт путем заполнения специальной формы.
- Модуль «Панель управления» – специализированный модуль системы, на который имеет доступ только администратор. Модуль доступен для администратора под специальным паролем. Администратор может добавлять, удалять или редактировать статьи, категории, тестирования, также фотографии из модуля галерея.

2.2 Используемые аппаратные и программные средства при разработке

1 Processor Intel Core2Duo(R) E8500 3,16 GHz;
2 DDR2 RAM – 4Gb;
3 Seagate Barracuda 1 Tb;
4 Nvidia GeForce GTS250 512 mb;
5 Display – BenQ G900.
При разработки интерфейса и программного кода были использованы:
1 Графический редактор – Adobe PhotoShop CS5.
2 Программа для верстки HTML – Macromedia DreamWeaver.
3 Текстовый редактор – CodeLobster.

2.3 Алгоритм работы системы

При входе на сайт пользователь попадает на главную страницу (индексную страницу), которая является основной точкой входа на сайт. Данная страница подгружает определенные модули формирующие верхнюю часть (header) страницы, основное меню, меню категорий, контент, и нижнюю часть (footer)страницы. Модуль headerявляется статичным и хранится в отдельном файле. Динамические модули берут данные из базы данных и выстраивают их на странице с учетом прописанных стилей. К динамическому контенту относятся: модули главного меню, меню категорий, статьи, а так же панель управления администратора.

При входе в режим администрирования нужно сначала авторизоваться используя пароль и логин, который проверяется в базе данных на соответствие. В режиме администратора все автоматизированные модули становятся доступны для редактирования. Редактируемые данные выводятся в текстовые поля HTML–формы. Данные сохраняются после нажатия на кнопку, ссылка которой отправляет на документ, выполняющий запрос в базу данных без перезагрузки страницы, в родительском окне. Автоматизированные модули подгружают информацию, опубликованную администратором из базы данных. Выбор модулей происходит путем передачи переменных, которые формируются в меню, через адресную строку. Блок–схема функционирования системы представлена на рисунке 2.1.

2.4 Выбор среды разработки

Главным фактором при проектировании системы на языке PHP является его практичность. PHP предоставляет программисту средства для быстрого и эффективного решения поставленных задач. Практический характер PHP обусловлен следующими важными характеристиками.

Традиционность. Язык PHP кажется знакомым программистам, работающим в разных областях. Многие конструкции языка позаимствованы из Си Perl, а нередко код PHP практически неотличим от того, что встречается в типичных программах C или Pascal. Это заметно снижает начальные усилия при изучении PHP.

Простота. Сценарий PHP может состоять из 10 000 строк или из одной строки – все зависит от специфики задачи. При этом не требуется подгружать библиотеки и указывать специальные параметры компиляции. Механизм PHP просто начинает выполнять код после первой экранирующей последовательности (<?) и продолжает выполнение до того момента, когда он встретит парную экранирующую последовательность (?>). Если код имеет правильный синтаксис, он исполняется в точности так, как указал программист.

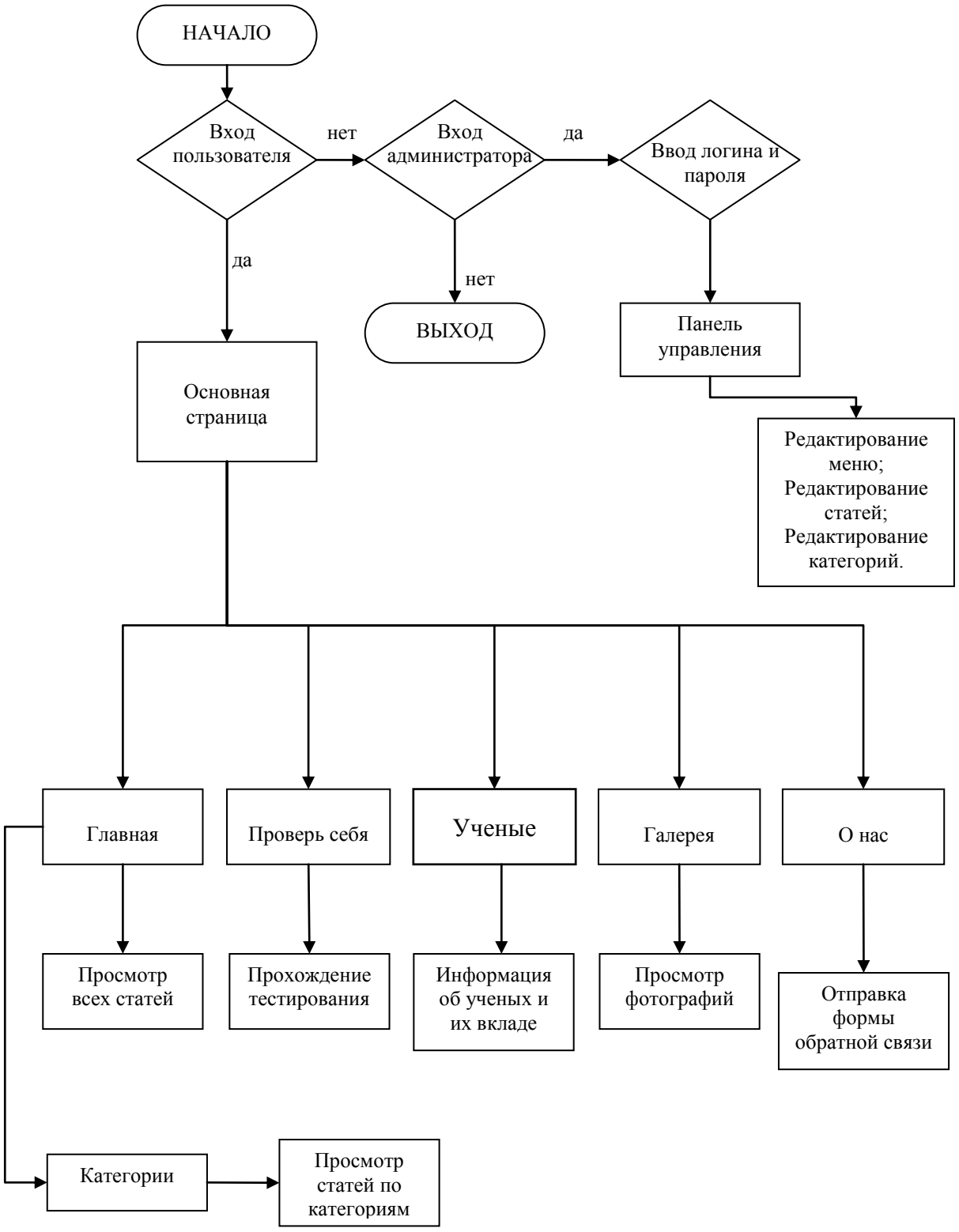

Рисунок 2.1 – Обобщенная блок–схема функционирования проекта

Эффективность. Эффективность является исключительно важным фактором при программировании для многопользовательских сред, к числу которых относится и WWW. В PHP 4.0 был реализован механизм выделения

ресурсов и обеспечена улучшенная поддержка объектно–ориентированного программирования, а также средства управления сеансом. В последней версии появился и механизм подсчета ссылок (reference counting), предотвращающий выделение лишней памяти.

Безопасность. PHP предоставляет в распоряжение разработчиков и администраторов гибкие и эффективные средства безопасности, которые условно делятся на две категории: средства системного уровня и средства уровня приложения.

Средства безопасности системного уровня. В PHP реализованы механизмы безопасности, находящиеся под управлением администраторов; при правильной настройке PHP это обеспечивает максимальную свободу действий и безопасность. PHP может работать в так называемом безопасном режиме (safemode), который ограничивает возможности применения PHP пользователями по ряду важных показателей. Например, можно ограничить максимальное время выполнения и использование памяти (неконтролируемый расход памяти отрицательно влияет на быстродействие сервера). Администратор также может устанавливать ограничения на каталоги, в которых пользователь может просматривать и исполнять сценарии PHP, а также использовать сценарии PHP для просмотра конфиденциальной информации на сервере (например, файла passwd).

Средства безопасности уровня приложения. В стандартный набор функций PHP входит ряд надежных механизмов шифрования. PHP также совместим с многими приложениями независимых фирм, что позволяет легко интегрировать его с защищенными технологиями электронной коммерции (e–commerce). Другое преимущество заключается в том, что исходный текст сценариев PHP нельзя просмотреть в браузере, поскольку сценарий компилируется до его отправки по запросу пользователя. Реализация PHP на стороне сервера предотвращает похищение нетривиальных сценариев пользователями, знаний которых хватает хотя бы для выполнения команды View Source.

Гибкость. Поскольку PHP является встраиваемым (embedded) языком, он отличается исключительной гибкостью по отношению к потребностям разработчика. Хотя PHP обычно рекомендуется использовать в сочетании с HTML, он с таким же успехом интегрируется и в JavaScript, WML, XML и другие языки. Кроме того, хорошо структурированные приложения PHP легко расширяются по мере необходимости (впрочем, это относится ко всем основным языкам программирования). Нет проблем и с зависимостью от браузеров, поскольку перед отправкой клиенту сценарии PHP полностью компилируются на стороне сервера. В сущности, сценарии PHP могут передаваться любым устройствам с браузерами, включая сотовые телефоны,

электронные записные книжки, пейджеры и портативные компьютеры, не говоря уже о традиционных PC. Программисты, занимающиеся вспомогательными утилитами, могут запускать PHP в режиме командной строки. Поскольку PHP не содержит кода, ориентированного на конкретный web–сервер, пользователи не ограничиваются определенными серверами (возможно, незнакомыми для них). Apache, Microsoft IIS, Netscape Enterprise Server, Stronghold и Zeus – PHP работает на всех перечисленных серверах. Поскольку эти серверы работают на разных платформах, PHP в целом является платформенно–независимым языком и существует на таких платформах, как UNIX, Solaris, FreeBSD и Windows 95/98/NT.

Наконец, средства PHP позволяют программисту работать с внешними компонентами, такими как Enterprise Java Beans или COM–объекты Win32. Благодаря этим новым возможностям PHP занимает достойное место среди современных технологий и обеспечивает масштабирование проектов до необходимых пределов.

Сайт выполнен на PHP с помощью HTML–тегов на основе объектно–ориентированного подхода, то есть он разбит на классы и каждый класс выполняет свою определенную задачу.

2.5 Обоснование выбора СУБД

Средства эффективного хранения и выборки больших объемов информации внесли огромный вклад в успешное развитие современных проектов. Обычно для хранения информации используются базы данных. Конечно, поддержка баз данных ориентирована не только на интересы гигантских корпораций – в распоряжении веб–программистов имеется несколько мощных реализаций баз данных, распространяемых по относительно низкой цене.

Правильная организация базы данных обеспечивает более быстрые и гибкие возможности выборки данных. Она существенно упрощает реализацию средств поиска и сортировки, а проблемы прав доступа к информации решаются при помощи средств контроля за привилегиями, присутствующими во многих системах управления базами данных (СУБД). Кроме того, упрощаются процессы репликации и архивации данных.

При выборе средств управления базами данных нужно отталкиваться прежде всего от требований, предъявляемых к проектируемой системе, а также учитывать возможность увеличения нагрузки на СУБД вследствие роста проекта.

Выбранный программный продукт должен удовлетворять как текущим, так и будущим потребностям организации. При этом следует учитывать

финансовые затраты на приобретение самой системы, необходимого оборудования, разработку необходимого программного обеспечения на основе этой системы, а также обучение персонала. Кроме того, необходимо убедиться, что новая СУБД способна принести реальные выгоды организации.

Для СУБД одним из важных факторов, влияющих на выбор, является быстродействие, так как скорость выборки записей из базы данных вносит значительный вклад в общую производительность всей системы. Следует учитывать и объем хранимых данных – для больших приложений, где количество записей превышает 100000, вопрос производительности является основным критерием, так как медленные СУБД не справятся с возложенными задачами. На начальном этапе создания веб–сайта объем данных не будет превышать нескольких тысяч, следовательно подойдет бесплатная и быстрая СУБД MySQL, хоть по производительности она уступает своему аналогу PostgreSQL.

В качестве СУБД выбран MySQL 4.0.20a. Выбор базы данных для этого проекта был не прост. На рынке имеется достаточное количество как бесплатных, так и коммерческих продуктов. Например, Postgress, mSQL – не коммерческие продукты. Postgress мощнее MySQL, но сложнее, а mSQL проще, но маломощный. К коммерческим продуктам относятся такие как, Oracle, MsSQLServer, Informix.

Основными достоинствами MySQL являются быстрота, надежность и простота использования. Несмотря на то, что MySQL не предоставляет такой широкий набор возможностей, как, скажем, Oracle, тем не менее его использование представляется рациональным из–за значительно меньших требований к мощности оборудования и значительно более высокой скорости работы.

2.6 Разработка базы данных

База данных системы состоит из шести таблиц. Их структуры приведены в таблицах 2.1 – 2.6:

1 Таблица «menu» – таблица для хранения информации о заголовках и текстах главного меню.

Т а б л и ц а 2.1 – Структура таблицы данных menu

Ключевое поле	Наименование	Тип	Примечание
primary key	id_menu	tinyint(3)	id пункта меню
	name_menu	varchar(255)	название
	text	text	текст меню

Состоит из 3–х полей:
– id_menu – ключевое поле типа tinyint;
– name_menu – название пункта меню;
– text – текст меню.

2 Таблица «test» – таблица для хранения информации о тестированиях.

Т а б л и ц а 2.2 – Структура таблицы данных test

Ключевое поле	Наименование	Тип	Примечание
primary key	id_test	tinyint(3)	id теста
	name_test	varchar(255)	название теста

Состоит из 2–х полей:
– id_test – ключевое поле типа tinyint;
– name_test – название теста.

3 Таблица «category» – таблица для хранения меню категорий.

Т а б л и ц а 2.3 – Структура таблицы данных category

Ключевое поле	Наименование	Тип	Примечание
primary key	id_category	tinyint(3)	id категории
	name_category	varchar(255)	название категории

Состоит из 2–х полей:
– id_category – ключевое поле типа tinyint;
– name_category – название категории.

4 Таблица «users» – таблица для хранения пользователей CMS.

Т а б л и ц а 2.4 – Структура таблицы данных users

Ключевое поле	Наименование	Тип	Примечание
primarykey	id_user	tinyint(3)	id пользователя
	login	varchar(255)	имя–логин
	password	varchar(32)	пароль

Состоит из 3–х полей:

– id– ключевое поле типа tinyint;
– login – имя пользователя;
– password – защифрованный пароль.

5 Таблица «statti» – таблица для хранения всех статей сайта.

Т а б л и ц а 2.5 – Структура таблицы данных statti

Ключевое поле	Наименование	Тип	Примечание
primary key	id	int	id статьи
	title	varchar(255)	название
	discription	text	краткое описание
	text	text	полныйтекст
	date	date	дата
	img_src	varchar(255)	адрес картинки
	cat	tinyint	категория

Состоит из 7–ти полей:
- id– ключевое поле типа integer;
- title – название статьи;
- discription – краткое описание;
- text – полный текст статьи;
- date – дата публикации;
- img_src – адрес привязанной картинки;
- cat – номер категории.

6 Таблица «contact» – таблица для хранения информации, которая приходит из формы обратной связи.

Т а б л и ц а 2.6 – Структура таблицы данных contact

Ключевое поле	Наименование	Тип	Примечание
primary key	id_letter	tinyint(3)	id письма
	title	varchar(255)	тема
	text_letter	text	Текст письма

Состоит из 3–х полей:
– id_letter – ключевое поле типа tinyint;
– title – тема письма;
– text_letter – текст письма.

Диаграмма БД представлена на рисунке 2.2.

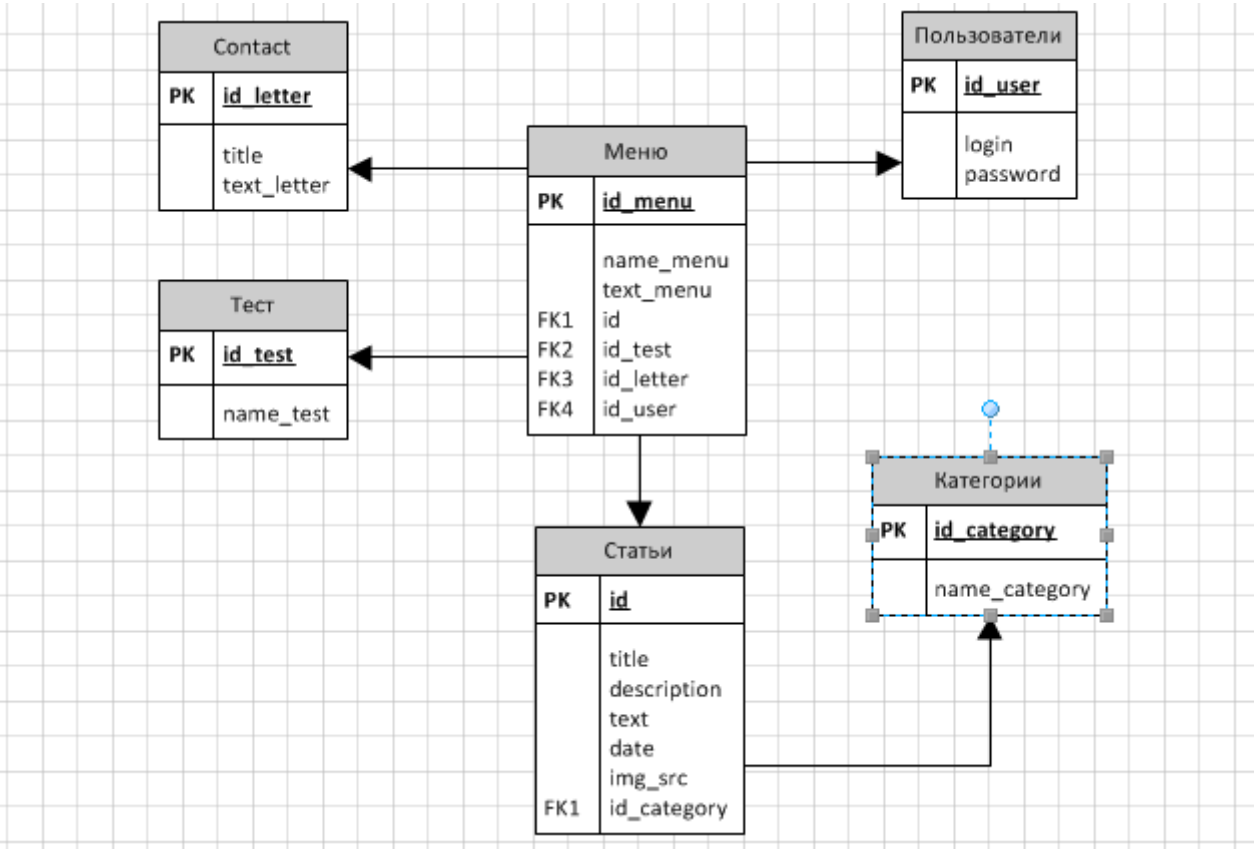

Рисунок 2.2 – Visio–диаграмма базы данных

2.7 Программирование сайта и системы управления контентом

Сайт выполнен на PHP с помощью HTML–тегов на основе объектно–ориентированного подхода, то есть он разбит на классы и каждый класс выполняет свою определенную задачу. Работает в любой операционной системе и в любом браузере. Протестирован на предмет багов на браузерах: Chrome, IE–8, Firefox, Opera. Объем проекта на диске составляет – 3,46 mb. Файл index.php – является основной точкой входа на сайт.

2.8 Логическая структура проекта

Схема логической структуры программы для создания системы приведена на рисунке 2.3.

Программный код информационной системы приведен в приложении проекта. Далее в таблицах 2.7 – 2.15 будут представлены структурные описания основных файлов проекта. Листинг модулей представлен в приложении А.

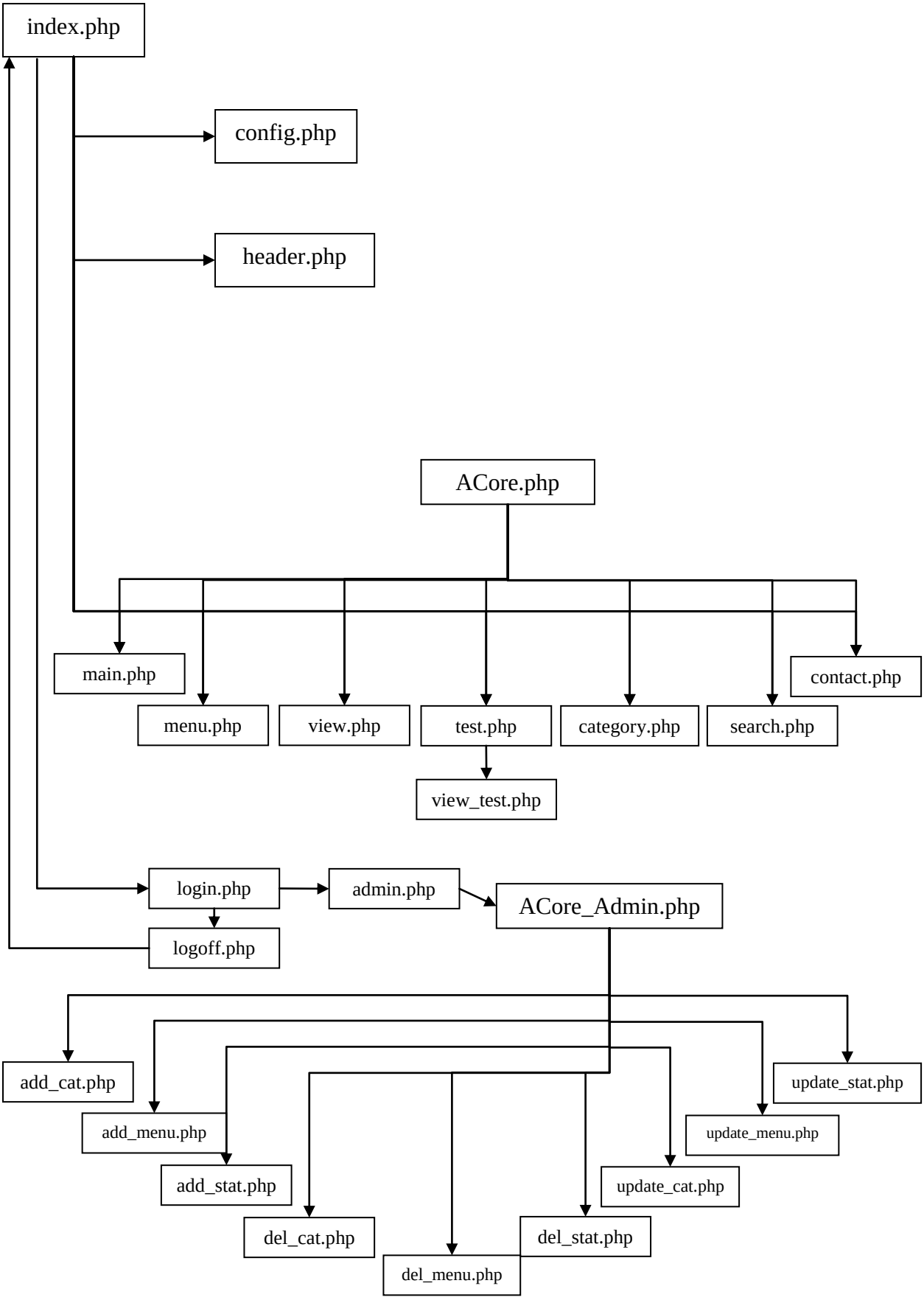

Рисунок 2.3 – Логическая структура проекта

Т а б л и ц а 2.7 – логическая структура index.php с привязкой к строкам текста

Номер строки	Пояснение кода
2	Начало сессии
3	Определение языковой разметки страниц
5–7	Подключение конфигурационного файла и файла ядра
9–14	Проверка на безопасность и на наличие параметра option
16–21	Скрипт подключения классов и создания функции get_body
22–24	Если файл не найдет то вывод ошибки
26–28	Если адрес задан не верно то вывод ошибки

Т а б л и ц а 2.8 – логическая структура main.php с привязкой к строкам текста

Номер строки	Пояснение кода
2	Определение класса наследника абстрактного класса Acore
4	Определение функции, которая вывод контент
9–14	Запрос на выборку из базы данных
16–27	Определение и заполнение по шаблону массива данных БД
29–30	Закрывающие теги для правильного отображения содержимого

Т а б л и ц а 2.9 – логическая структура ACore.php с привязкой к строкам текста

Номер строки	Пояснение кода
3	Определение абстрактного класса
5–16	Добавление и подключение дескриптора БД
18–20	Подключение верхней части страницы
22–30	Определение функции, которая получает информацию о меню категорий
38–41	Подключение модуля поиска по сайту
45–52	Определение и заполнение по шаблону массива меню категорий
53–58	Вывод кнопок соц. сетей
66–102	Определение методов и функций и вывод главного меню сайта
104–125	Определение методов и функций и вывод нижней части сайта
127–137	Определение функций вывода контента
139	Определение абстрактной функции вывода динамического контента

Т а б л и ц а 2.10 – логическая структура test.php с привязкой к строкам текста

Номер строки	Пояснение кода
2	Определение класса наследника абстрактного класса Acore
4	Определение функции, которая вывод контент
8–10	Вывод текста страницы
13–18	Запрос на выборку в БД
20–27	Определение и заполнение по шаблону массива данных из БД
29–31	Закрывающие теги для правильного отображения содержимого

Т а б л и ц а 2.11 – логическая структура view.php с привязкой к строкам текста

Номер строки	Пояснение кода
2	Определение класса наследника абстрактного класса Acore
4	Определение функции, которая вывод контент
8–15	Проверка на правильность данных для вывода
17–30	Если верно то запрос в БД и вывод данных
35–37	Закрывающие теги для правильного отображения содержимого

Т а б л и ц а 2.12 – логическая структура contact.php с привязкой к строкам текста

Номер строки	Пояснение кода
2	Определение класса наследника абстрактного класса Acore
4–45	Определение функции, которая проверяет входные данные на безопасность для сервера, а также на правильность заполнения всех форм
49–88	Форма обратной связи
91–92	Закрывающие теги для правильного отображения содержимого

Т а б л и ц а 2.13 – логическая структура admin.php с привязкой к строкам текста

Номер строки	Пояснение кода
2	Определение класса наследника абстрактного класса Acore

Окончание таблицы 2.13

Номер строки	Пояснение кода
4	Определение функции вывода контента
6–19	Запрос в БД на выборку и вывод данных о статьях
21–28	Определение и заполнение по шаблону массива данных БД
31–33	Закрывающие теги для правильного отображения содержимого

Таблица 2.14 – логическая структура login.php с привязкой к строкам текста

Номер строки	Пояснение кода
2	Определение класса наследника абстрактного класса Acore
4–24	Определение функции, которая проверят на безопасность и правильность вводы форм, также сверяет данные с БД
25–27	Если не верно, то выход
32–34	Если заполнены не все поля, то выход
38	Определение функции вывода контента
42–54	Вывод формы авторизации
56–58	Закрывающие теги для правильного отображения содержимого

Таблица 2.15 – логическая структура ACore_Admin.php с привязкой к строкам текста

Номер строки	Пояснение кода
3	Определение абстрактного класса
4	Добавление и подключение дескриптора БД
7–19	Определение конструктора, который проверяет был ли введен адрес авторизации админа, и подключается к БД
22–24	Подключение верхней части страницы
26–49	Вывод меню админки
62–69	Определение методов и функций и вывод нижней части сайта
71–82	Определение функций вывода контента
84	Определение абстрактной функции вывода динамического контента
86–137	Определение методов и функций обработки запросов на изменение и удаление в БД

2.9 Контрольный пример

Для запуска необходимо запустить web–сервер из директории C:\WebServers\etc\run.exe, затем открыть браузер и в адресной строке вписать URL http://localhost/. После этого загрузится главная страница проекта (рисунок 2.4).

Главное меню содержит пять пунктов. При клике на пункт меню, отображается информационный контент, соответствующему пункту меню. Пункт меню при наведении курсора выделяется другим цветом. Меню категорий содержит 5 пунктов. При клике на пункт меню категорий, отображается информационный контент, соответствующей категории. Пункт меню категорий при наведении курсора инвертирует фоновый градиент что создает иллюзию объемной кнопки. Так же на всех страницах сайта присутствует модуль поиска по сайту и кнопки социальных сетей.

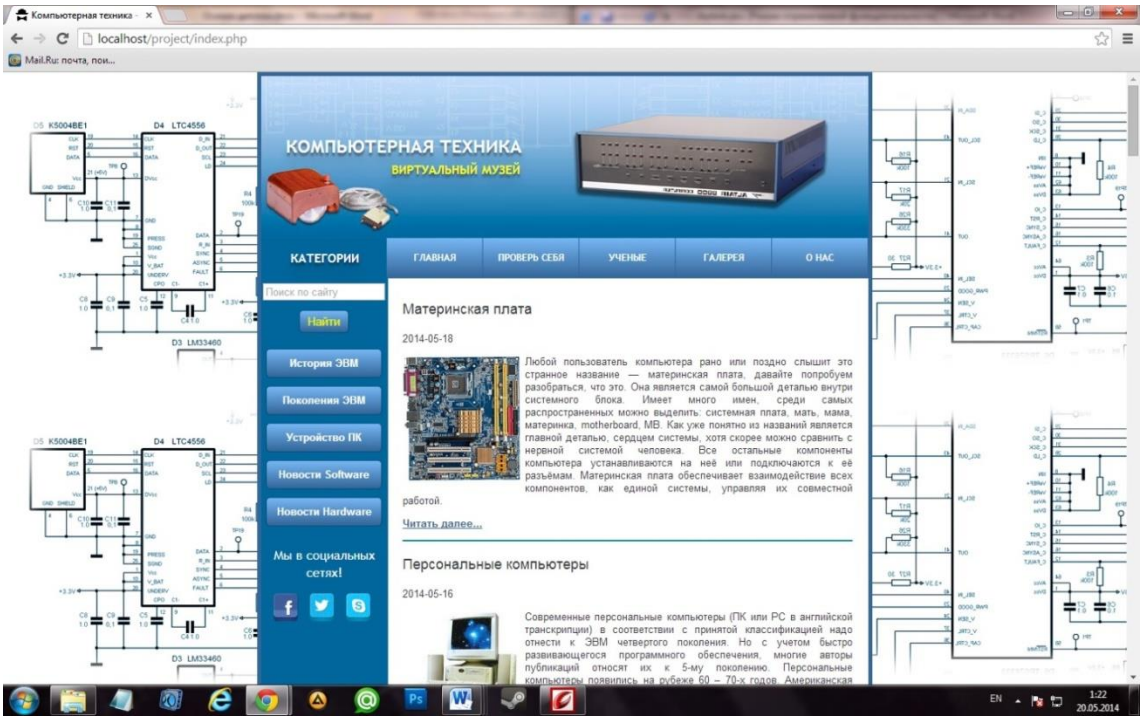

Рисунок 2.4 – Главная страница

Описание структуры главного меню сайта:

1) Проверь себя.

В данном пункте меню (рисунок 2.5) представлены ссылки на тесты, которые помогут проверить знания посетителей в определенной области, так же присутствует вступительное слово и дополнительная информация. Для того чтобы начать тестирование нужно кликнуть по ссылке по интересующей теме, сразу загрузиться форма тестирования (рисунок 2.6).

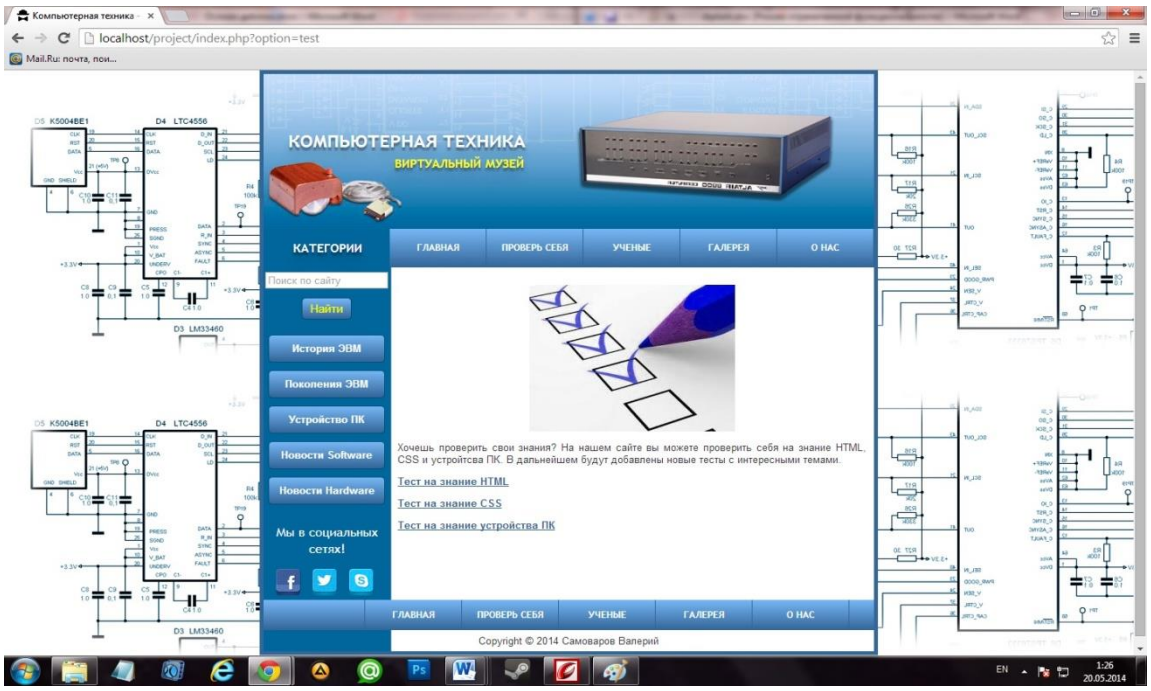

Рисунок 2.5 – Проверь себя

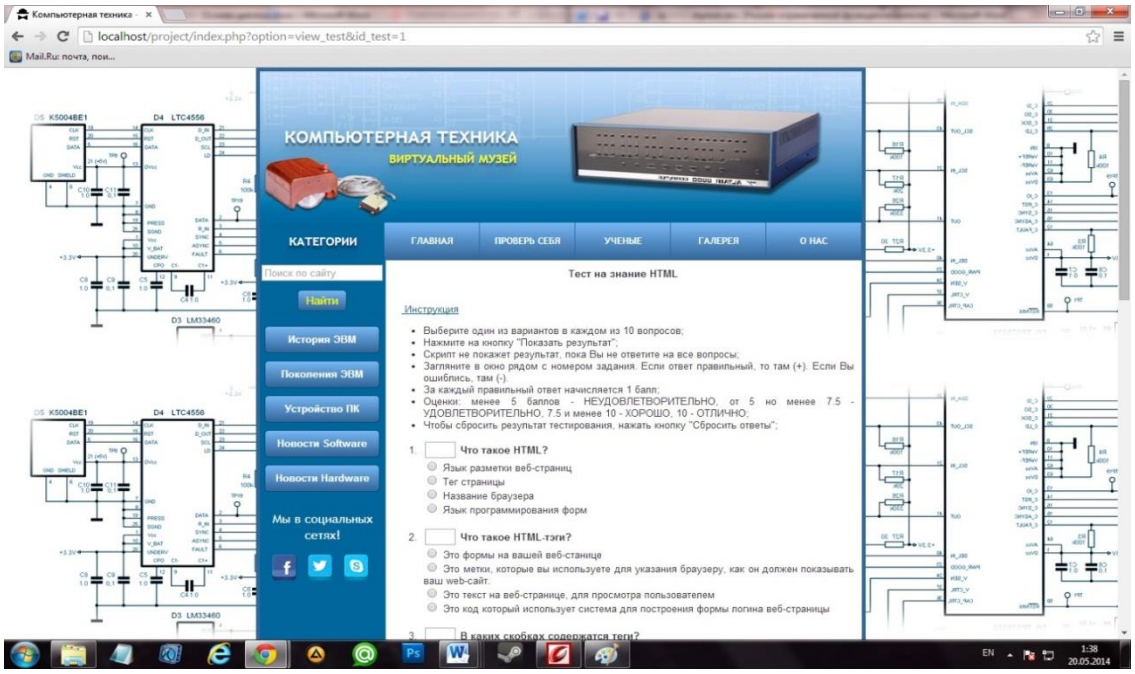

Рисунок 2.6 – Форма тестирования

В верхней части формы имеется инструкция для прохождения тестирования, в нижней части кнопка вывода результата и сброса пройденного теста.

2) Ученые.

В данном пункте меню (рисунок 2.7) представлены фотографии и данные по разработкам ученых, которые внесли свой вклад в развитие компьютерной техники.

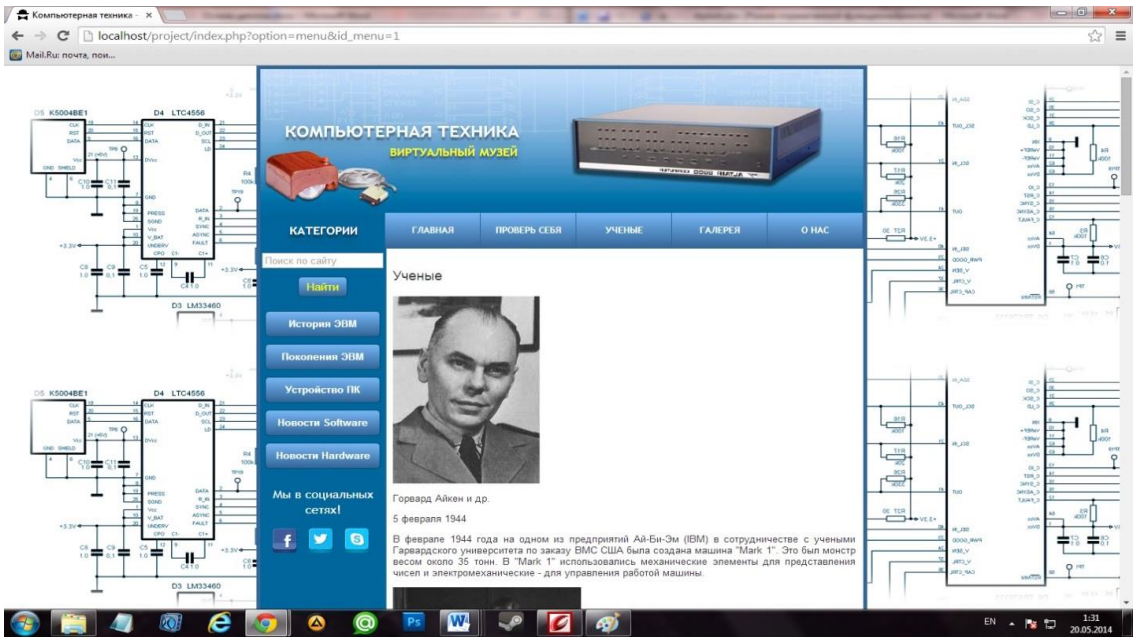

Рисунок 2.7 – Ученые

3) Галерея.

В данном пункте меню (рисунок 2.8) представлены фотографии различных ЭВМ и ПЭВМ прошлых лет. Галерея сделана с помощью технологий CSS3. При наведении курсора картинка увеличивается и ее можно рассмотреть более детально.

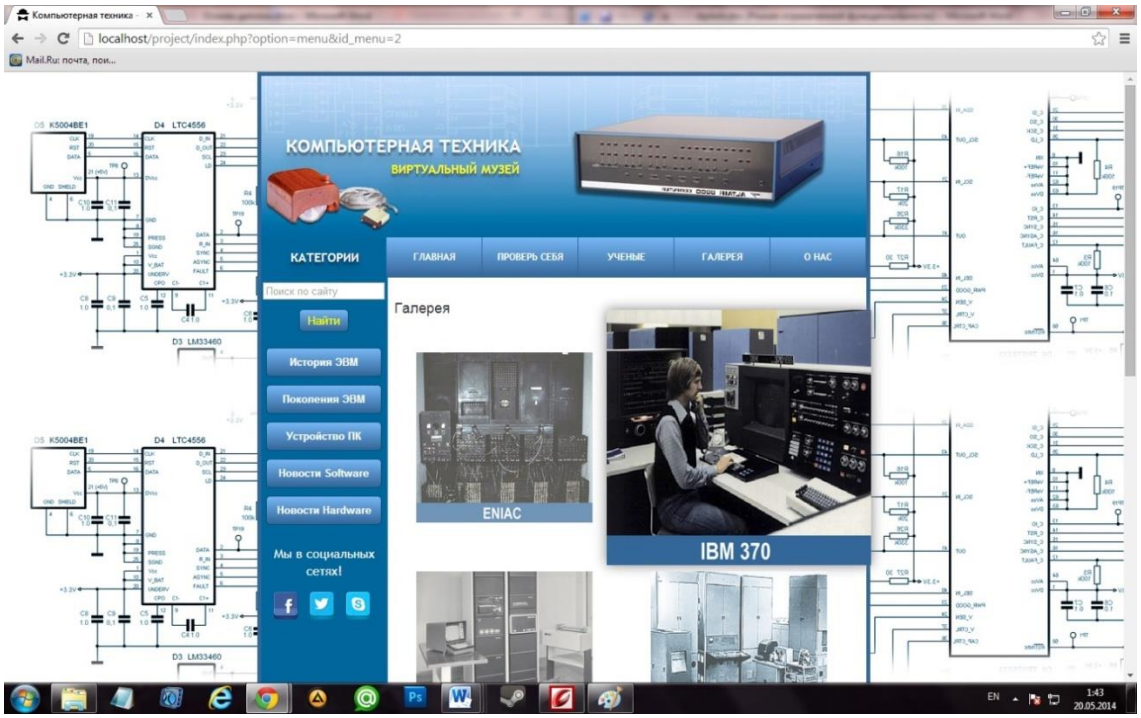

Рисунок 2.8 – Галерея

4) О нас.

В данном пункте меню (рисунок 2.9) представлена форма обратной связи посетителей сайта с администратором.

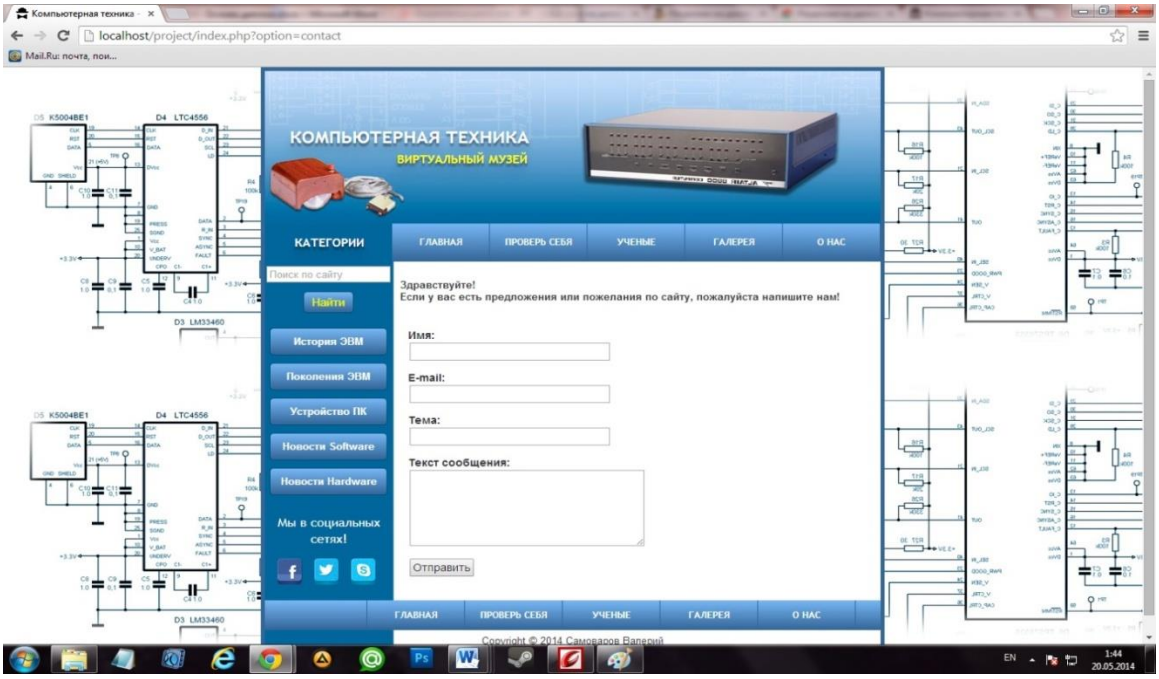

Рисунок 2.9 – Форма обратной связи

5) Вход в CMS.

При вводе в строку браузера параметра “login” (рисунок 2.10), вы попадаете на страницу авторизации административной панели управления контентом сайта.

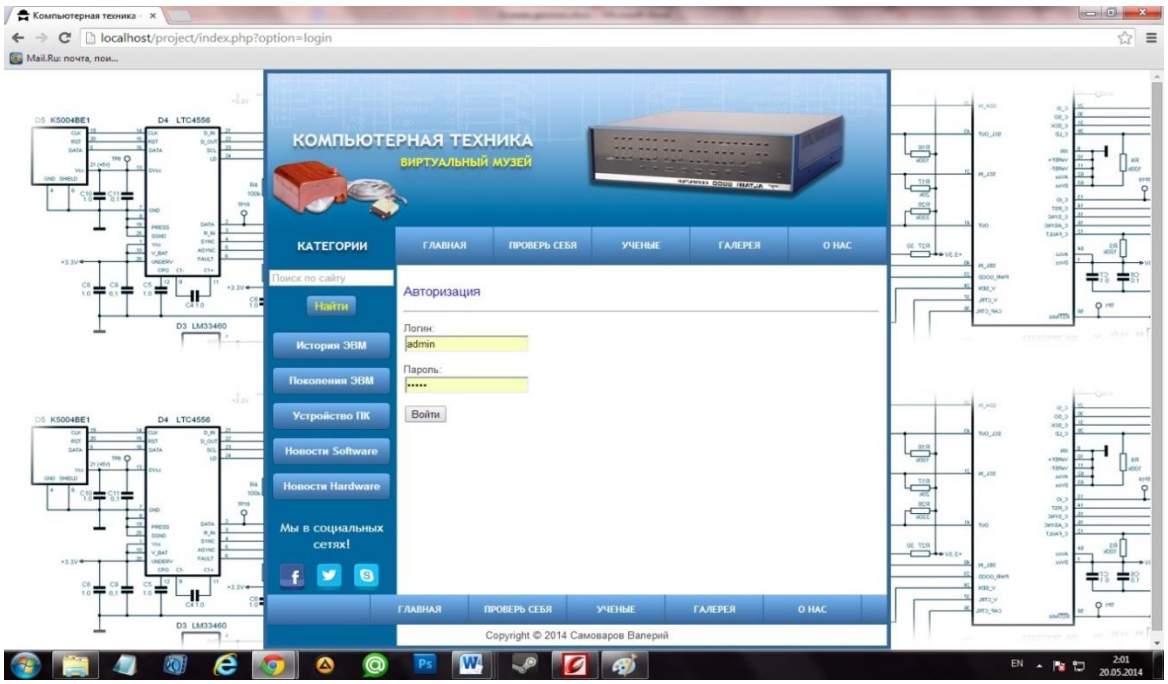

Рисунок 2.9 – Страница авторизации администратора

6) Главная страница админ. панели.

На данной странице выведен список всех статей сайта (рисунок 2.11), которые при нажатии на название можно редактировать в специальной форме (рисунок 2.12), также нажатием на слово «удалить» выделенное красным цветом, можно удалить данную статью. Сверху страницы присутствует пункт добавления новой статьи, слева меню админ. панели.

Рисунок 2.11 – Главная страница админ. панели

7) Пункты меню админ. панели.

При нажатии на пункты меню админ. панели – «Меню», «Категории», «Статьи» – вы попадаете на страницу списка этих элементов (рисунок 2.13) с возможностью редактирования, удаления или добавления новых элементов. При нажатии кнопки выход – вы попадаете обратно на главную страницу, а сессия администратора будет завершена, т.е. если попытаться вернуться снова в раздел админ. панели, нужно будет снова проходить авторизацию.

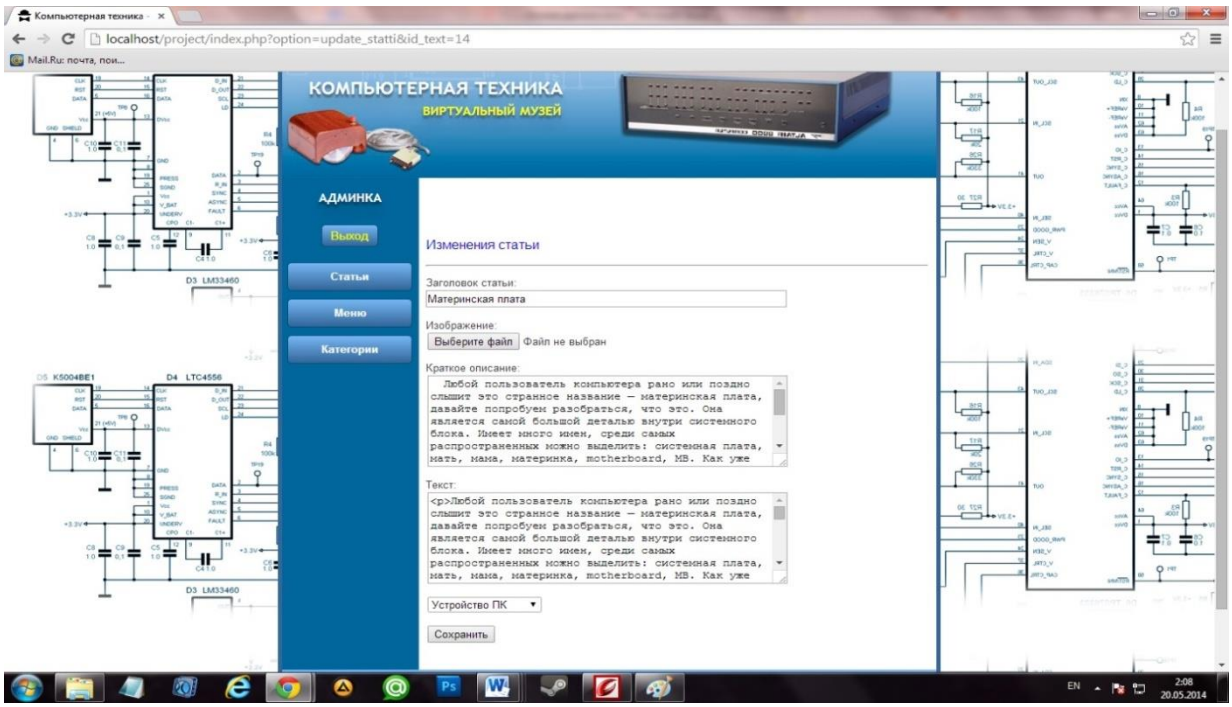

Рисунок 2.12 – Форма редактирования статьи

Рисунок 2.13 – Страница редактирования меню категорий

3 Экономическое обоснование проекта

3.1 Описание работы

Целью данной дипломной работы является создание виртуального музея компьютерной техники. Сайт в первую очередь направлен на изучение истории компьютерной техники и их возможностей, также просмотра новостей о современных разработках в этой сфере. Также есть возможность участия в опросах, обсуждениях по различным новостям и статьям и многое другое.

– интерфейс сайта максимально прост для понимания;

– организован специальный раздел сайта для пользователей с тестами, который позволяет проверить свои знания в области компьютерной техники, а также различные другие тесты связанные с этой темой;

– организована система поиска новостей, статей и информации по сайту;

– спроектирована административная панель управления контентом сайта;

– имеются кнопки различных социальных сетей, для того чтобы пользователи могли поделится об увиденном на сайте.

Во всем мире компьютерная индустрия развивается большими темпами. В нашей стране уже практически в каждом доме есть компьютер, но компьютерная грамотность все еще находится на низком уровне. Знать историю тоже важно, особенно для людей обучающимся по профессиям связанных с компьютерами. Также данный сайт предоставляет возможность получения подробной информации о современных отечественных и зарубежных разработках и новостях. Исходя из этого было принято решение о разработке данного сайта, который обладал бы этими качествами.

В данном разделе приводится рассмотрение экономической составляющей реализации данной проекта, отражающей временные, трудовые и финансовые затраты на проект.

3.2 Трудовые ресурсы, используемые в работе

В данной работе используется интеллектуальный труд, стоимость затрат которого выше, чем физического труда.

В разработке приняли участие:

– руководитель проекта – постановка задачи, разработка схемы ПО и базы данных;

– дизайнер – разработка интерфейса;

– программист – разработка алгоритмов и программирование.

Количество сотрудников задействованных в разработке представлено в таблице 3.1.

Т а б л и ц а 3.1 – Сотрудники и их заработная плата

Должность	Количество человек	Заработная плата в месяц, тенге
Руководитель проекта	1	70000
Программист	1	70000
Дизайнер	1	60000
Итого	3	200000

3.3 Оборудование, используемое в работе

Оборудование, используемое при разработке сайта представлено в таблице 3.2.

Т а б л и ц а 3.2 – Перечень оборудования, необходимого для разработки

Наименование изделий	Характеристика	Количе–ство единиц	Цена за единицу, тенге	Общая сумма, тенге
Системный блок	Intel Core2Duo E8500 3,16Ghz/ RAM DDR3 4Gb/ HDD 500Gb/Nvidia GTS250 512Mb/ Gigabyte EP45T/DVD–RW	2	68500	137000
Монитор	BenQ 19’	2	17000	34000
МФУ	Panasonic KX–MB1500	1	21000	21000
Итого				192000

Цены на оборудование приведены без учета НДС.

3.4 Программное обеспечение, используемое в работе

При разработке проекта «виртуальный музей» было использовано следующее программное обеспечение:

– Windows 7 SP1 – операционная система.

– Adobe Dreamweawer – HTML–редактор.

– Denwer – набор дистрибутивов и программная оболочка, предназначен для создания и отладки сайтов на локальном сервере.

– Хостинг – услуга по предоставлению вычислительных мощностей для физического размещения информации на сервере, постоянно находящимся в интернете.

Программное обеспечение, использованное при разработке сайта, представлено в таблице 3.3.

Т а б л и ц а 3.3 – Перечень использованного программного обеспечения, необходимого для разработки футбольного портала

Программное обеспечение	Стоимость, тенге
Microsoft Windows 7 SP1 Максимальная	15500
Adobe Dreamweaver	3000
Denwer	бесплатно
Хостинг	бесплатно
Итого	18500

Цены на ПО приведены без учета НДС.

3.5 Сроки реализации проекта

Процесс разработки и сроки реализации проекта «виртуальный музей» состоит из 6 этапов и включает в себя:

– сбор информации для сайта;
– проектирование и создание базы данных;
– разработка дизайна;
– программирование и верстка сайта;
– тестирование программного обеспечения;
– оформление отчетов.

Этапы и сроки реализации проекта отображены в таблице 3.4.

Т а б л и ц а 3.4 – Этапы и сроки реализации проекта

Перечень работ		Недели от начала работ					
		1	2	3	4	5	6
1 этап	Постановка задачи	■					
	Разработка схемы и дизайна сайта	■					
	Подбор и изучение литературы	■					
	Подбор материала для заполнения информацией сайта		■				
2 этап	Проектирование и создание БД		■				
	Заполнение базы данных		■	■			
3 этап	Создание интерфейса сайта			■	■		
4 этап	Программирование сайта				■	■	
	Разработка админ. панели					■	
5 этап	Тестирование ПО					■	
	Отладка ПО					■	■
	Оформление НИР						■
	Проверка и сдача отчета						■

3.6 Расчет стоимости работы по разработке

Расчет стоимости работы по разработке – это наиболее важная часть экономического анализа, так как на основе этого расчета определяются затраты рабочего времени на разработку проекта на каждом этапе, а также трудовые затраты.

Затраты на разработку данного проекта определяются по формуле

$$С = ФОТ + О_С + А + Э + С_{пр} + Н \quad (3.1)$$

где ФОТ – фонд оплаты труда;
Ос – социальный налог;
А – амортизационные отчисления;
Э – затраты на электроэнергию;
$С_{пр}$ – прочие расходы;
Н – накладные расходы.

3.6.1 Расчет затрат на оплату труда

Затраты на оплату труда персонала, задействованного в разработке проекта, рассчитывается по формуле

$$ФОТ = З_{осн} + З_{доп} \quad (3.2)$$

где $З_{осн}$ – основная заработная плата;
$З_{доп}$ – дополнительная заработная плата.

Труд программиста и дизайнера принят условно, на договорной основе в размере 80000 и 60000 тенге соответственно.

На этапах разработки программного продукта участники разработки сайта задействованы неравноценно, для этого необходимо рассчитать средний дневной заработок, а затем и общий размер заработной платы, в зависимости от их фактического участия.

Средний дневной заработок каждого работника рассчитывается по формуле

$$D = \frac{ЗПм}{Др} \quad (3.3)$$

где ЗПм – ежемесячный размер заработной платы;
Др – количество рабочих дней в месяце (это 24 дня – шестидневная рабочая неделя).

1) Для руководителя проекта

$$D = \frac{70000}{24} = 2917 \text{ тенге/день}.$$

2) Для дизайнера

$$D = \frac{60000}{24} = 2500 \text{ тенге/день}.$$

3) Для программиста

$$D = \frac{70000}{24} = 2917 \text{ тенге/день}..$$

Заработная плата за один час вычисляется по формуле

$$\text{Н} = \frac{D}{\text{Чр}} \quad (3.4)$$

где D – средний дневной заработок работника;
Чр – количество часов рабочего дня (Чр = 8).

1) Для руководителя проекта

$$D = \frac{2917}{8} = 365 \text{ тенге/час.}$$

2) Для дизайнера

$$D = \frac{2500}{8} = 313 \text{ тенге/час.}$$

3) Для программиста

$$D = \frac{2917}{8} = 365 \text{ тенге/час.}$$

Длительность цикла в днях по каждому виду работ определяется по формуле

$$t_n = \frac{T}{q_n * z * K} \quad (3.5)$$

где T – трудоемкость этапа, норма–час;
q_n – количество исполнителей по этапу;
z – продолжительность рабочего дня, z = 8 часов;
К – коэффициент выполнения норм времени, К = 1,1.

Полученную величину t_n округляем в большую сторону до целых дней.

$t_1 = \frac{16}{1*8*1,1} \approx 2$ дня; Руководитель, постановка задачи;

$t_2 = \frac{16}{1*8*1,1} \approx 2$ дня; Руководитель, разработка схемы и дизайна сайта;

$t_3 = \frac{20}{1*8*1,1} \approx 2$ дня; Руководитель, подбор и изучение литературы;

$t_4 = \frac{16}{1*8*1,1} \approx 2$ дня; Программист, подбор информации для сайта;

$t_5 = \frac{24}{1*8*1,1} \approx 3$ дня; Программист, проектирование и создание базы данных;

$t_6 = \frac{24}{1*8*1,1} \approx 3$ дня; Программист, заполнение базы данных;

$t_7 = \frac{16}{1*8*1,1} \approx 2$ дня; Дизайнер, создание дизайна сайта;

$t_8 = \frac{24}{1*8*1,1} \approx 3$ дня; Программист, верстка сайта;

$t_9 = \frac{42}{1*8*1,1} \approx 5$ дней;Программист, программирование сайта;

$t_{10} = \frac{24}{1*8*1,1} \approx 3$ дня; Программист, разработка админ. панели;

$t_{11} = \frac{8}{1*8*1,1} \approx 1$ дня; Программист, тестирование;

$t_{12} = \frac{16}{1*8*1,1} \approx 2$ дня; Программист, отладка;

$t_{13} = \frac{24}{1*8*1,1} \approx 3$ дня; Руководитель, оформление НИР;

$t_{14} = \frac{24}{1*8*1,1} \approx 3$ дня; Руководитель, проверка и сдача отчета.

$$t_n = 2 + 2 + 2 + 2 + 3 + 3 + 2 + 3 + 5 + 3 + 1 + 2 + 3 + 3 = 36 \text{ дней}$$

Таким образом, для проведения всех работ необходимо 36 дней.

Сводные данные по расчету заработной платы персонала, задействованного в разработке проекта приведены в таблице 3.5.

Т а б л и ц а 3.5 – Сводные данные по расчету основной заработной платы персонала задействованного в разработке проекта

Наименование этапов	Исполнитель	Трудоем–кость, норма–час,	Длительно-сть цикла, дни	Заработная плата за час работы, тенге	Сумма заработной платы, тенге
Постановка задачи	Руководитель	16	2	365	5833
Разработка схемы и дизайна сайта	Руководитель	16	2	365	5833
Подбор и изучение литературы	Руководитель	20	2	365	7291
Подбор материала для сайта	Программист	16	2	365	5833
Проектирование и создание базы данных	Программист	24	3	365	8750
Заполнение базы данных	Программист	24	3	365	8750
Создание дизайна сайта	Дизайнер	16	2	313	5000
Верстка сайта	Программист	24	3	365	8750
Программирование сайта	Программист	42	5	365	15312
Разработка админ панели	Программист	24	3	365	8750
Тестирование	Программист	8	1	365	2917
Отладка	Программист	16	2	365	5833
Оформление НИР	Руководитель	24	3	365	8750
Проверка и сдача отчета	Руководитель	24	3	364,57	8750
Итого		294	36	5052	106350

Дополнительная заработная плата составляет 10% от основной заработной платы и вычисляется по формуле

$$З_{доп} = З_{осн} * 0,1 \quad (3.6)$$

и составит

$$З_{доп} = 106350 * 0{,}1 = 10635 \text{ тенге}$$

Таким образом, затраты на оплату труда согласно произведенным расчетам и в соответствии с формулой 3.2 составит

$$ФОТ = 106350 + 10635 = 116985 \text{ тенге}$$

3.6.2 Расчет затрат по социальному налогу

Социальный налог составляет 11% (ст. 358 п.1 НК РК) от дохода работника, и рассчитывается по формуле

$$О_с = (ФОТ - ПО) * 11\% \quad (3.7)$$

где ПО – пенсионные отчисления, которые составляют 10% от ФОТ и социальным налогом не облагаются, вычисляются отчисления по формуле

$$ПО = ФОТ * 10\% \quad (3.8)$$

$$ПО = 116985 * 0{,}1 = 11698 \text{ тенге}$$

Таким образом, в соответствии с произведенными расчетами и согласно формуле 3.7 размер отчислений на социальные нужды составит

$$О_с = (116985 - 11698) * 0{,}11 = 11582 \text{ тенге}$$

3.6.3 Расчет амортизационных отчислений

Амортизационные отчисления рассчитываются по формуле

$$А_i = \frac{H_A * C_{пер} * N}{100 * 12 * n} \quad (3.9)$$

где H_A – норма амортизации;

$C_{пер}$– первоначальная стоимость оборудования;

N – количество дней на выполнение работ;

n – количество рабочих дней в месяце.

Норма амортизации на компьютерную технику составляет 40% от стоимости всего оборудования, на программное обеспечение – 25%.

Таким образом, амортизационные отчисления по используемому оборудованию, в соответствии с формулой 3.9, составят

– на ПК

$$А_1 = \frac{40*85500*2*36}{100*12*24} = 8550 \text{ тенге;}$$

– на МФУ

$$А_2 = \frac{40*21000*6}{100*12*24} = 175 \text{ тенге;}$$

– на ПО

$$А_3 = \frac{25*18500*36}{100*12*24} = 578 \text{ тенге;}$$

Сводные результаты расчета амортизационных отчислений представлены в таблице 3.6.

Т а б л и ц а 3.6 – Сводные данные по расчету затрат на амортизацию

Наименование оборудования	Стоимость, тенге	Количество	Норма амортизации, %	Сумма амортизации, тенге
Системный блок с монитором	85500	2	40	8550
МФУ	21000	1	40	175
Программное обеспечение	18500	–	25	578
Итого		–	–	9303

3.6.4 Расчет затрат на электроэнергию

Поскольку в процессе производства используется электрооборудование, необходимо рассчитать на электроэнергию. Затраты на электроэнергию для производственных нужд включают в себя расходы электроэнергии на оборудование и дополнительные.

$$Э = З_{эл.эн.об.} + З_{доп.} \quad (3.10)$$

где $З_{эл.эн.об.}$ – затраты на электроэнергию для оборудования;

$З_{доп.}$ – затраты электроэнергии на дополнительные нужды.

Расходы на электроэнергию для оборудования рассчитываются по формуле

$$З_{эл.эн.об.} = W * T * S * K_{исп} \quad (3.11)$$

где W – потребляемая мощность, Вт;

T – время работы, часы;

S – тариф (1кВт = 16,90 тенге);

$К_{исп}$ – коэффициент использования ($К_{исп}$= 0,9).

$З_{эл.эн.об.(ПК)} = 0{,}7 * 294 * 16{,}90 * 0{,}9 * 2$ = 6260 тенге,

$З_{эл.эн.об\ (МФУ)} = 0{,}9 * 48 * 16{,}90 * 0{,}9$ = 657 тенге

Общая сумма затрат на электроэнергию основного оборудования согласно формуле 4.11 составляет

$З_{эл.эн.об.} = 6260 + 657 = 6917$ тенге

Затраты на дополнительные нужды берутся по показателю от затрат на оборудование в размере 5% и рассчитывается по формуле

$$З_{доп.} = З_{эл.эн.об.} * 5\% \qquad (3.12)$$

и составляют

$З_{доп.} = 6917 * 0{,}05 = 346$ тенге

Таким образом суммарные затраты на электроэнергию, согласно формуле 3.11 составляют

Э = 6917 + 346 = 7263 тенге

Сводные результаты расчета затрат на электроэнергию представлены в таблице 3.7.

Т а б л и ц а 3.7 – Сводные данные о затратах на электроэнергию

Наименование приборов	Потребляемая мощность, Вт	Число рабочих дней	Коэффициент использования	Время работы оборудования, часы	Сумма затрат, тенге
2 компьютера с мониторами	0,7	36	0,9	294	6260
МФУ	0,9	6	0,9	48	657
Итого	–	–	–	–	6917

3.6.5 Расчет накладных и прочих расходов

Прочие расходы включают в себя:

– расходы на интернет – 3600 тенге;

– расходы на канцелярские товары – 1500 тенге.

и составляют

$$С_{пр} = 3600 + 1500 = 5100 \text{ тенге}$$

Накладные расходы составляют 50% от всех затрат и рассчитываются по формуле

$$Н = (ФОТ + О_с + А + Э + С_{пр}) * 50\% \quad (3.13)$$

$$Н = (116985 + 11582 + 9303 + 7263 + 5100) * 0{,}5 = 75117 \text{ тенге}$$

3.6.6 Расчет стоимости по всем статьям затрат и определение структуры затрат

В соответствии с формулой 3.1 суммарные затраты по разработке моего проекта составляют

$$С = 116985 + 11582 + 9303 + 7263 + 5100 + 75117 = 225350 \text{ тенге}$$

Смета затрат по разработке «виртуального музея», а также структура расходов представлены в таблице 3.8 и на рисунке 3.1.

Т а б л и ц а 3.8 – Суммарные данные по стоимости разработки проекта

Наименование статьи затрат	Сумма, тенге
Фонд оплаты труда	116985
Социальный налог	11582
Амортизация	9303
Затраты на электроэнергию	7263
Прочие расходы	5100
Накладные расходы	75117
Расходы на ПО	18500
Итого	243850

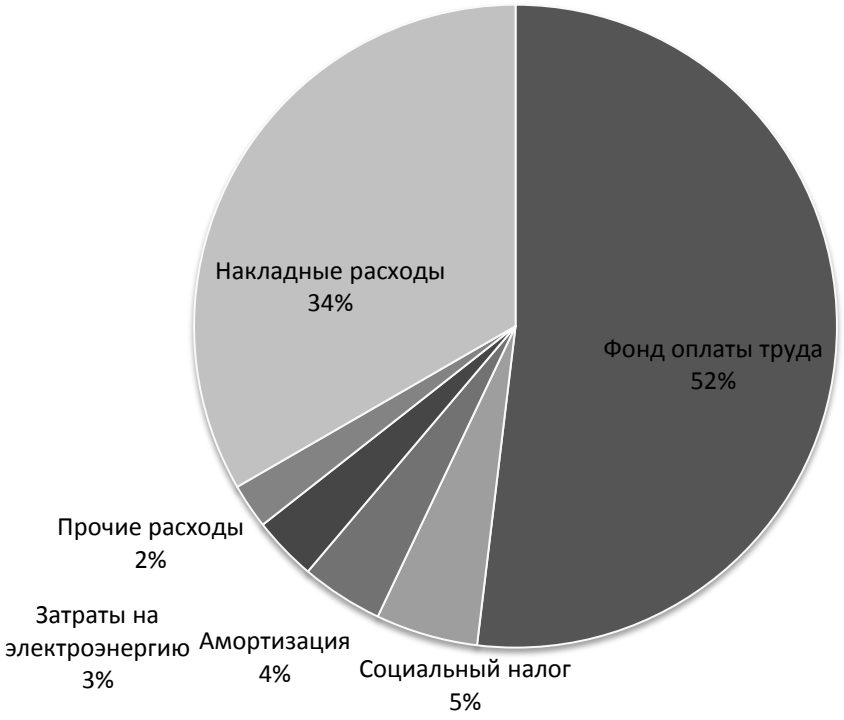

Рисунок 3.1 – Структура затрат по разработке сайта

3.7 Цена интеллектуального труда

Цена реализации проекта складывается из стоимости и чистого дохода

$$Ц = С + П \quad (3.14)$$

где С – стоимость продукта;
П – чистый доход.

При определении первоначальной цены следует задаться желаемым уровнем рентабельности (здесь 20%) реализации программных продуктов

$$Ц_п = С * (1 + \frac{P}{100}) \quad (3.15)$$

где Р – рентабельность.

$$Ц_п = 243850 * \left(1 + \frac{20}{100}\right) = 292620 \text{ тенге}$$

Цена реализации проекта рассчитывается по формуле

$$Ц_р = Ц_п + НДС \quad (3.16)$$

где НДС – налог на добавленную стоимость по ставке 12%.

$$НДС = Ц_п * 12\% \quad (3.17)$$

НДС = 292620* 0,12 = 35114 тенге

В соответствии с формулой 3.16 цена реализации проекта составит

$Ц_р = 292620 + 35114 = 327734$ тенге

В зависимости от спроса окончательная цена на интеллектуальный труд в условиях рыночных отношений будет варьироваться и изменяться. На данный момент стоимость разработки такого веб–сайта в крупных веб–студиях варьируется от 250 000 тенге до 350 000 тенге.

Вывод

Разработка сайта является сложным и трудоемким процессом, требующих больших затрат интеллектуального труда. Стоимость разработки включает в себя следующие категории затрат: фонд оплаты труда, отчисления на социальные нужды, затраты на амортизацию и электроэнергию.

Стоимость разработки проекта составила – 243 850 тенге. Наибольшую долю в общей себестоимости разработки программного продукта составляют затраты на оплату труда в размере 116 985 тенге, что составляет 52% от суммы разработки всего проекта.

Анализируя полученные расчеты, можно сделать вывод, что данный проект является экономически выгодным, так как цена реализации продукта конкурентоспособна.

4 Безопасность жизнедеятельности

4.1 Характеристика помещения и факторы, действующие на разработчика в процессе труда. Рабочее место

Помещение, в котором находится рабочее место разработчика, имеет ряд основных характеристик:

– длина помещения – 5 метров;

– ширина помещения – 4 метра;

– высота помещения – 3 метра;

– число окон – 1;

– число рабочих мест – 3;

– освещение: естественное (через боковое окно) и общее искусственное.

Рабочее место – это система функционально и пространственно–организованных технических средств и предметов труда, позволяющая успешно решать поставленные перед разработчиком задачи.

Оптимально организованное рабочее место помогает увеличить производительность труда на 8–20% и уменьшить вредное воздействие компьютера на здоровье человека.

Увеличению утомляемости на производстве благоприятствует неправильная эргономическая организация рабочего места, не правильные зоны размещения оборудования по высоте от пола, по фронту от оси симметрии и т.д., из-за этого далее будут рассмотрены эргономические требования к рабочему месту.

Методы организации рабочего места зависят от типа решаемых задач, от оборудования используемого на предприятии, от конкретной рабочей деятельности разработчика.

Характеристики и размеры рабочего места определяются антропологическими характеристиками человека и нормированы в соответствующем нормативном акте – ГОСТ 21889–76 [10].

4.1.1 Правильная поза при работе с ПК

Основную часть своего рабочего времени разработчик проводит в сидячем положении. Такое положение позволяет уменьшить потери энергии, довольно сильно уменьшить утомление во время работы.

Данная поза наиболее подходит для продолжительной работы с учетом некоторых основных моментов:

– спина наклонена на несколько градусов назад (такая поза позволяет разгрузить позвоночник);

– руки свободно опущены на подлокотники кресла;

– локти и запястья расслаблены, кисти имеют общую ось с предплечьями: не сгибаются и не разгибаются;

– ноги твердо стоят на полу или на специальной подставке.

4.1.2 Состав рабочего места разработчика

Рабочее место разработчика формируется из:

– пространства, отведенного для оборудования;

– пространства требуемого для технического обслуживания или ремонта;

– зоны проходов, обеспечивающей нормальную работу оборудования;

– сенсомоторного пространства (части пространства рабочего места, в которой происходит двигательная и сенсорная работа человека).

Из необходимых для работы устройств:

– устройства отображения информации (монитор);

– стол;

– кресло;

– устройства ввода информации (мышь, клавиатура и т. д.);

– устройства вывода информации (принтер, плоттер).

4.1.3 Эргономические требования к рабочему месту

Зрительный комфорт, при работе с монитором, в основном характеризуется следующими аспектами:

– размерами знаков;

– расстояние между знаками по горизонтали: 0,25 высоты знака;

– расстояние между строками: 0,5–1,0 высоты знака;

– количеством знаков в строке: 4–80;

– максимально допустимым количеством строк для цветного изображения: не более 25;

– освещенности и равномерности яркости между окружающими участками и остальными местами рабочего места.

При проектировании стола необходимо учитывать следующее:

– высота стола должна быть выбрана с учетом возможности сидеть свободно, в удобной позе, при необходимости опираясь на подлокотники;

– нижняя часть стола должна быть сконструирована так, чтобы разработчик мог удобно сидеть, не был вынужден поджимать ноги;

– поверхность стола должна обладать качественными характеристиками, исключающими появление бликов в поле зрения разработчика;

– конструкция стола должна обеспечивать наличие выдвижных ящиков (не менее 3 для хранения документации, листингов, канцелярских товаров, личных вещей).

В соответствии с требованиями для оборудования рабочего места определенными в ГОСТ 21889–76 [10] выбираем следующие параметры стола:

– высота рабочей поверхности стола 700 мм;

– высота пространства для ног 650 мм;

– возможность размещения документов справа и слева;

– расстояние от глаз до клавиатуры 400 мм;

– расстояние от глаз до документов 500 мм.

Кресло разработчика должно быть устойчивым. Его конструкция, размеры, форма, наклон сиденья и спинки должны предусматривать возможность сидеть, выпрямившись, поддерживая тяжесть верхней части туловища не напряжением мышц спины, а методом опоры на спинку.

Сиденье должно иметь небольшой наклон назад (на 5–6 градусов), учитывающий устойчивость позы, спинка кресла должна иметь вогнутую форму:

– высота сиденья над уровнем пола 450 мм;

– поверхность сиденья мягкая с закругленным передним краем;

– расстояние от сиденья до нижнего края рабочей поверхности не менее 150 мм.

Общие размеры рабочего места должны быть достаточными для:

– размещения работающего человека с учетом его движений и перемещений;

– расположения органов управления в пределах наибольшей и наименьшей из границ рабочего пространства;

– максимального обзора визуальной области;

– смены рабочей позы и рабочего положения;

– свободного доступа к оборудованию при ремонте и отладке;

– оптимального размещения основных и дополнительных средств труда;

– ведения записей, работы с документами и приборами.

Как правильно, монитор должен располагаться – примерно на расстоянии вытянутой руки – наиболее благоприятное расстояние – 450–500мм.

Плоскость экрана должна быть перпендикулярной линии взора, а высота расположения такой, чтобы при выпрямленной спине пользователя и отрегулированном кресле направление взгляда было бы горизонтальным или на 10–15 градусов ниже центра монитора.

Устройства ввода–вывода, такие как принтер, плоттер, сканер и т.д., рекомендуется размещать справа от разработчика в зоне наибольшей досягаемости. Шумящие устройства необходимо выносить за пределы рабочей зоны.

4.1.4 Расположение рабочего места в помещении

Наиболее часто рабочие места сотрудников, работающих с ПК, ставят подальше от окон и таким образом, чтобы оконные проемы оказались сбоку. Если экран дисплея обращен к окну, требуются защитные экраны. На окна рекомендуется устанавливать светорассеивающие шторы, регулируемые жалюзи или солнцезащитные пленки с металлизированным нанесением.

Монитор, документы, клавиатура должны находиться так, чтобы перепад яркостей их поверхностей, зависящий от их месторасположения относительно источников света, не был больше 1:10 при рекомендуемом значении 1:3. При яркости изображения на экране 50–100 кд/м (номинальное значение)

освещенность документа должна быть в радиусе 300–500 лк. Должны быть исключены слепящие яркости, блики и отображения от стекла экрана.

Для исключения засветки экранов дисплеев прямыми световыми потоками светильники общего освещения располагают сбоку от рабочего места, параллельно линии зрения оператора и стене с окнами.

Для обеспечения оптимальных условий работы операторов дисплейных устройств необходима определенная отделка помещений: должны использоваться диффузно – отражающиеся материалы с коэффициентом отражения для потолка – 0,7 – 0,8; для стен – 0,5 – 0,6; для пола – 0,3 – 0,5.

Поверхность пола в помещениях эксплуатации компьютеров должна быть ровной, без выбоин, нескользкой, удобной для очистки и для влажной уборки, обладать антистатическими свойствами.

Схемы расположения рабочих мест должны учитывать расстояния между рабочими столами с мониторами (в направлении тыла поверхности одного монитора и экрана другого монитора), которое должно быть не меньше 2,0 м, а расстояние между боковыми поверхностями мониторов – не меньше 1,2 м.

4.1.5 Требования к микроклимату в рабочей зоне помещений. Категория работы

Работа с компьютером относится к категории легких физических работ (категории 1а и 1б), производимых в сидячем положении и не требующих систематического физического напряжения.

В соответствии с требованиями СанПиН 2.2.2.542 – 96 [12] в помещениях, в которых работа на ПК является основной, должны обеспечиваться оптимальные характеристики показателей микроклимата, приведенные в таблице 4.1.

Таблица 4.1 – Оптимальные нормы показателей микроклимата в рабочей зоне помещений с ПК

Период года	Категория работ	Температура воздуха, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
Холодный	легкая 1а	22–24	40–60	0,1
	легкая 16	21–23	40–60	0.1
Теплый	легкая 1а	23–25	40–60	0,1
	легкая 16	22–24	40–60	0,2

Воздух, который поступает в помещение, должен быть не меньше 19°С и очищен от пыли и микроорганизмов.

Наиболее эффективным мероприятием, обеспечивающим в помещении автоматическое поддержание оптимальных параметров микроклимата в нормативных пределах и требуемую чистоту воздуха, является применение кондиционирования.

При необходимости повышения влажности воздуха в помещениях следует использовать увлажнители воздуха, заправляемые ежедневно дистиллированной или прокипяченной водой.

Уровни положительных и отрицательных аэроионов в воздухе помещении с ПК должны соответствовать нормам, приведенным в таблице 4.2.

Т а б л и ц а 4.2 – Уровни ионизации воздуха помещений при работе на ПК

Уровни	Число ионов в 1 $см^3$ воздуха	
	n+	n–
Минимально необходимые	400	600
Оптимальные	1500–3000	30000–50000
Максимально допустимые	50000	50000

Для улучшения в помещении качественного состава воздуха и обеспечения в нем требуемого содержания аэроионов помещения с ПК каждый час должны быть проветрены. При нарушении аэроионного режима в помещении следует применять ионизаторы воздуха.

4.2 Освещение рабочего места

Наиболее важным условием эффективного труда разработчика является *правильное освещение* его рабочего места. Объясняется это тем, что с помощью зрительного аппарата человек получает около 90% информации из окружающего мира, ее поступление в основном зависит от качества освещенности.

Неблагоприятными факторами для профессиональной деятельности разработчика, плохо влияющими на зрение, являются:

– пониженный уровень освещенности, ведущий к перенапряжению глаз и, как следствие к быстрому утомлению;

– чрезмерно высокая освещенность также является причиной быстрой утомляемости, приводя к раздражению и рези в глазах;

– неправильное направление света способствует появлению резких теней или сильных бликов, заметно быстрее утомляющих глаза.

Для освещения помещений и рабочих мест с ПК может использоваться естественное, искусственное и совмещенное освещение.

Естественное освещение должно осуществляться через светопроемы, обращенные в основном на север и северо–восток, и обеспечивать коэффициент естественной освещенности (КЕО) не ниже 1,2% в зонах с устойчивым снежным покровом и не ниже 1,5% на остальной территории.

Рабочие места с ПК по отношению к световым проемам должны находиться так, чтобы естественный свет падал сбоку, преимущественно слева.

Искусственное освещение в помещениях, в которых используются ПК, должно выполняться системой общего равномерного освещения.

Хорошо проникающий в помещение естественный свет оказывает хорошее воздействие на психику человека, вызывая положительные эмоции, что способствует хорошим гигиеническим условиям работы. За счет естественного освещения стимулируется обмен веществ, кровообращение, дыхание, деятельность центральной нервной системы, что в свою очередь, обеспечивает наибольшую эффективность труда.

4.2.1 Определение системы освещения

Выбирая систему освещения, необходимо учитывать, что самой эффективной является система комбинированного освещения, но система общего освещения более гигиенична, т.к. обеспечивает большую равномерность освещенности рабочих поверхностей. Используя локализованное общее освещение, можно наиболее просто добиться высоких уровней освещенностей на рабочих местах без значительных затрат. При больших требованиях к освещенности некоторых рабочих мест реализуют комбинированную систему освещения.

Искусственное освещение различных помещений осуществляется с помощью электрических источников света – ламп накаливания и люминесцентных ламп, так называемых ламп дневного света. Для освещения производственных помещений широкое применение находят люминесцентные лампы, что объясняется рядом причин.

– применение данного типа ламп позволяет получить в 1,5 – 2 раза большую освещенность при одинаковом расходе электрической энергии по сравнению с лампами накаливания (экономичность);

– свет люминесцентной лампы мягкий, по своему спектру наиболее близкий к дневному естественному, при почти полном отсутствии теней, что позволяет лучше различать цвета и оттенки;

– также, люминесцентные лампы имеют более длительный срок службы, превышающий срок службы лампы накаливания в 10–12 раз.

К недостатком данного вида ламп можно отнести:

– высокую установочную стоимость;

– зависимость светового потока от температуры окружающей среды;

– существенная пульсация светового потока.

4.2.2 Расчет искусственного освещения

Найдем необходимое число ламп при помощи метода коэффициента использования.

Расчёт системы общего освещения производится методом коэффициента использования светового потока, который выражается отношением светового потока, падающего на расчётную поверхность, к суммарному потоку всех ламп.

Его величина зависит от характеристик светильника, габаритов помещения, цвета стен и потолка, характеризуемой коэффициентами отражения стен и потолка.

Необходимый световой поток лампы в каждом светильнике рассчитывается по формуле

$$Fo = \frac{E \cdot k \cdot S \cdot Z}{\eta} \qquad (4.1)$$

где E – заданная минимальная освещённость, лк (E = 500);

k – коэффициент запаса, учитывающий уменьшение светового потока лампы в результате загрязнения светильников в процессе эксплуатации (для люмин. ламп – 1,5);

S – освещаемая площадь, м^2 (S = 20);

Z – отношение средней освещённости к минимальной (обычно принимается равным 1,1–1,2 , для люмин. ламп – 1,1);

η – коэффициент использования светового потока в долях единицы (отношение светового потока, падающего на расчётную поверхность, к суммарному потоку всех ламп).

Коэффициент использования η зависит от типа светильника, от коэффициентов отражения потолка $\rho_п$, стен $\rho_с$, расчётной поверхности $\rho_р$, индекса помещения, который рассчитывается по формуле

$$i = \frac{S}{h \cdot (a + b)} \qquad (4.2)$$

где h – высота светильника над рабочей поверхностью;

a – длина помещения;

b – ширина помещения.

Найдем h по формуле

$$h = H – h_р – h_с = 4 – 0{,}7 – 0 = 3{,}3 \text{ (м)}$$

где H = высота помещения, м (H = 4);

$h_р$ = высота рабочей поверхности от пола, м ($h_р$ = 0,7);

$h_с$ = высота свеса светильника от основного потолка, м ($h_с$ = 0).

$$i = \frac{5 \cdot 4}{3{,}3 \cdot (5 + 4)} = \frac{20}{3{,}3 \cdot 9} = 0{,}67$$

Для светлого фона примем: $\rho_п = 70$, $\rho_с = 50$, $\rho_р = 10 \Rightarrow \eta = 36\ \%$.

$$Fл = \frac{Fo}{N} \quad (4.3)$$

где Fл – световой поток одной лампы;

Fo – общий световой поток;

N – число ламп;

η – коэффициент использования светового потока.

$$Fo = \frac{500 \cdot 1{,}5 \cdot 20 \cdot 1{,}1}{0{,}36} = 45833\ лм$$

Количество ламп рассчитаем из выраженной формулы

$$N = \frac{Fo}{Fл} \quad (4.4)$$

Для освещения выбираем люминесцентные лампы типа ЛХБ65, световой поток которых *Fл* = 4400 Лм.

$$N = \frac{45833}{4400} = 10{,}4 \Rightarrow 10\ шт$$

Число светильников сильно зависит от габаритов освещаемого помещения, при этом количество светильников должно быть таким, чтобы отношение расстояния между ними к высоте их подвеса над поверхностью было равно 1,5 ÷ 2.

При выборе осветительных приборов используем светильники типа ЛСПО 2. Каждый светильник комплектуется двумя лампами. Размещаются светильники тремя рядами, по два в каждом ряду.

Допускается отклонение (ε) светового потока выбранной лампы от расчётного от – 10 % до + 20 %.

$$E_{факт} = \frac{\Phi_{Л} \cdot N \cdot N_{л.св} \cdot \eta}{S \cdot z \cdot k} \quad (4.5)$$

$$E_{факт} = \frac{4400 \cdot 6 \cdot 2 \cdot 0{,}36}{20 \cdot 1{,}1 \cdot 1{,}5} = 576\ лк$$

Отличие от нормированного уровня

$$\frac{E_{факт} - E_{норм}}{E_{норм}} \cdot 100\% \quad (4.6)$$

$$\frac{576-500}{500}\cdot 100\% = 15{,}2\%$$

Для исключения засветки экранов дисплеев прямыми световыми потоками светильники общего освещения располагают сбоку от рабочего места, параллельно линии зрения оператора и стене с окнами. Такое размещение светильников позволяет производить их последовательное включение в зависимости от величины естественной освещённости и исключает раздражение глаз чередующимися полосами света и тени, возникающее при поперечном расположении светильников.

Электрическая мощность всей осветительной системы вычисляется по формуле

$$P_{общ} = P_1 \cdot N \tag{4.7}$$

где P_1 – мощность одной лампы = 65 Вт;
N – число ламп = 10.

$$P_{общ} = 65 \cdot 12 = 780\,\text{Вт}$$

План помещения представлен на рисунке 4.1. Огнетушитель обозначен «1». Лампы заштрихованными прямоугольниками. Стрелки показывают план эвакуации.

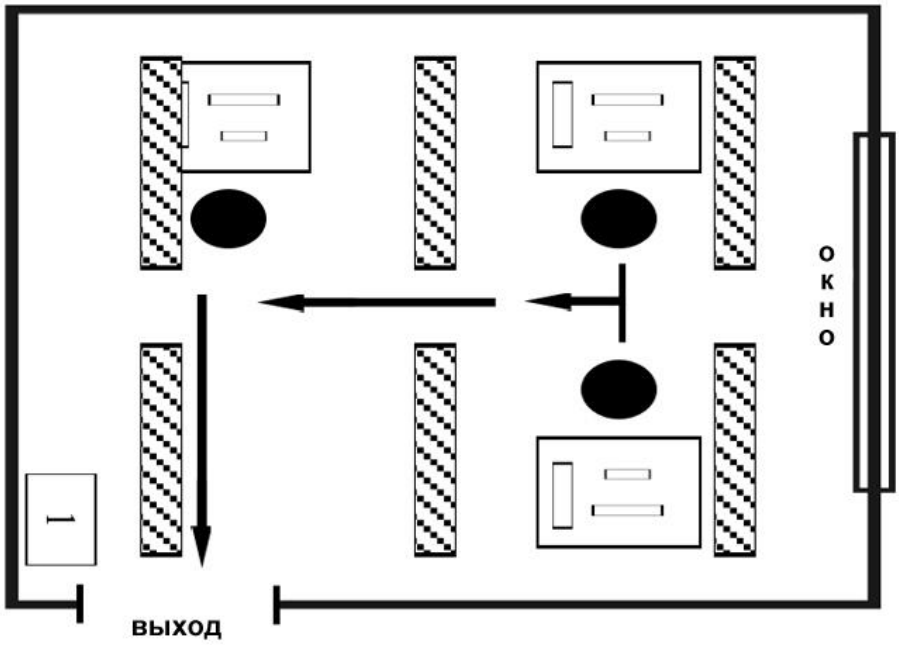

Рисунок 4.1 – План помещения

Вывод

В результате анализа условий труда на рабочем месте разработчика были выявлены два основных фактора, влияющие на организм человека в процессе работы. Этими факторами являются освещение и правильное кондиционирование.

Правильно спроектированное и выбранное производственное освещение улучшает условия зрительной работы, снижает утомляемость, способствует повышению производительности труда, благотворно влияет на производственную среду, оказывая положительное психологическое воздействие на сотрудника, повышает безопасность труда, и снижает травматизм.

Также учтены все факторы по соблюдению в помещении максимально благоприятного микроклимата и расположения рабочих мест сотрудников.

Заключение

Сегодня, путешествуя по просторам Интернета, можно найти много очень качественно выполненных сайтов, которые быстро загружаются, с привлекательным дизайном, имеют четко разграниченную по разделам информацию и удобные элементы навигации. Такие сайты позволяют пользователям задержаться на них некоторое время, получить позитивные эмоции и обучится чему–то новому.

Если пользователь останавливается на каком–либо сайте на длительное время и даже не осознает этого, то, по можно сделать вывод, что такой сайт грамотно продуман.

В данной выпускной работе изучены популярные вопросы разработки и создания современных сайтов, так как создание качественного и грамотно выполненного сайта это кропотливая работа. В заключении можно добавить, что разрабатывая сайт, в первую очередь нужно обращать внимание на его общую структуру и информационную основу. Но есть и довольно важный параметр, который нельзя не учитывать: эффективность поиска сайта в сети Интернет. При плохой оптимизации и раскрутке сайта его практически невозможно найти в поисковых системах, иногда даже при прямом запросе. Для того, чтобы сайт выполнял рекламную функцию или служил виртуальной визиткой предприятия, необходимо с особой тщательностью подходить к регистрации его в поисковых системах, подбору ключевых слов и мета тегов. Веб–сайт может хорошо представлять образовательное учреждение в сети Интернет, но только если сайт постоянно развивается, добавляются новые интересные модули, разделы. Так же не менее важным моментов является постоянное обновление и добавление материалов на сайт [4].

Чтобы сайт хорошо выполнял возложенные на него функции, необходимо постоянно работать на его развитие.

Список используемой литературы

1 Панкова Е.В. Интернет-сайт среднего профессионального учебного заведения: эффективность поиска в Интернете. Научные и технические библиотеки, 2008.

2 Дронов В. Разработка современных web–сайтов. – М.: Фолиант, 2011.

3 Шмитт К. CSS. Рецепты программирования. – СПб.: Питер, 2011.

4 Сайт www.wikipedia.com

5 Зандстра М. PHP. Практика создания web–сайтов. – М.: Вильямс, 2010.

6 Кент М. PHP для начинающих. – М.: Вильямс, 2006.

7 Сайт www.gosthelp.ru/gost/gost11254.html

8 Беляков Г.И. Безопасность жизнедеятельности. Охрана труда: Учебник для бакалавров. – М.: Юрайт, 2012.

9 Коробко В.И. Охрана труда: Учебное пособие для студентов вузов. – М.: Юнити, 2013.

10 Бадагуев Б.Т. Документация по охране труда в организации. – М.: Альфа–пресс, 2010.

11 Баранов Н.И. Охрана труда. Учебное пособие. – М.: О–Комплект, 2002.

12 СНиП РК 2.04.–05.2002. Естественное и искусственное освещение.

13 Абдимуратов Ж. С., Маманбаева С. Е. Расчет производственного помещения. – Алматы: АУЭС, 2009.

14 Айдарханова М. Основы экономической теории. – М.: Фолиант, 2010.

15 Махотина М.В., Симоненко В.И. Экономика в схемах: Учебное пособие. – М.: Эксмо, 2011.

16 Носова С.С. Экономическая теория. – М.: Кнорус, 2010.

17 Артамонова В.С., Иванова С.А. Экономическая теория. – СПб.: Питер, 2010.

18 Камаева В.Д., Лобачевой Е.И. Экономическая теория: Учебник. – М.: Юрайт, 2010.

19 Выпускная работа бакалавров. Экономический раздел: методичка/ под ред. Бабич А. А., Казыкен Б. Б., Сагира А. А. – Алматы: АУЭС, 2009.

Приложение А

Листинг файла index.php

```
<?php
session_start();
header("content-Type:text/html; charset=UTF-8");

require_once("config.php");
require_once("classes/ACore.php");
require_once("classes/ACore_Admin.php");

if($_GET['option']){
	$class = trim(strip_tags($_GET['option']));
	}
	else{
		$class = 'main';
		}

if(file_exists("classes/".$class.".php")){
	include("classes/".$class.".php");
	if (class_exists($class)){
		$obj = new $class;
		$obj->get_body();
	}
	else {
		exit("<p>Не правильные данные для входа</p>");
	}
}
else{
	exit("<p>Не правильный адресс</p>");
}

?>
```

Листинг файла ACore.php

```
<?php
abstract class ACore{

	protected $db;


	public function __construct(){
		$this->db = mysql_connect(HOST,USER,PASSWORD);
		if(!$this->db){
			exit("Ошибка соединения с базой данных".mysql_error());
		}
		if(!mysql_select_db(DB,$this->db)){
			exit("Нет такой базы данных".mysql_error());
		}
		mysql_query("SET NAMES 'UTF8'");
	}

	protected function get_header(){
		include "header.php";
	}

	protected function get_left_bar(){
```

```
            $query = "SELECT id_category, name_category FROM category";

            $result = mysql_query($query);
            if (!$result){
                  exit(mysql_error());
            }


            echo '<div class="quick-bg">
                        <div id="spacer" style="margin-bottom:10px";>
                              <div id="rc-bg">Категории</div>
                        </div>';


            echo '<div class = "search" style="margin-top: 15px;"><form
name="search" method="POST" action="?option=search">
                  <input type="search" name="query" placeholder="Поиск по
сайту">
                              <button class="but" type="submit">Найти</button>
                                    </form></div>';



            $row = array();

            for ($i = 0; $i < mysql_num_rows($result); $i++){
                  $row = mysql_fetch_array($result, MYSQL_ASSOC);
                  printf('<div class="side-menu">
                              <a href="?option=category&id_cat=%s">%s</a>

      </div>',$row['id_category'],$row['name_category']);
            }
            echo '<h2>Мы в социальных сетях!</h2>
                  <div id="social">
                  <a       href="http://facebook.com/"       class="social-icon
facebook"></a>
                  <a href="http://twitter.com/" class="social-icon twitter"></a>
                  <a href="http://skype.com/" class="social-icon skype"></a>
                  </div>';


            echo "</div>";


      }

      protected function get_menu(){
            $row = $this->menu_array();
            echo '<div id="mainarea">
                        <div class="heading">';
            echo '<div class="toplinks">
                              <a href="?option=main">Главная</a></div>';

            echo '<div class="toplinks">
                              <a href="?option=test">Проверь себя</a></div>';


            foreach($row as $item){
                  printf('<div class="toplinks">
                              <a href="?option=menu&id_menu=%s">%s</a>
```

```
      </div>',$item['id_menu'],$item['name_menu']);
            }

            echo '<div class="toplinks">
                              <a href="?option=contact">О нас</a></div>';

            echo "</div>";
      }

      protected function menu_array(){
            $query = "SELECT id_menu, name_menu FROM menu";

            $result = mysql_query($query);
            if (!$result){
                  exit(mysql_error());
            }

            $row = array();
            for ($i = 0; $i < mysql_num_rows($result); $i++){
                  $row[] = mysql_fetch_array($result, MYSQL_ASSOC);

            }
            return $row;
      }

      protected function get_footer(){
            $row = $this->menu_array();

            echo '<div id="bottom">';
            echo  '<div  class="toplinks"      style="margin-left:127px;  border-
left:solid 1px #336699;"">
                              <a href="?oprion=main ">Главная</a></div>';

            echo '<div class="toplinks">
                              <a href="?option=test">Проверь себя</a></div>';

            foreach($row as $item){
                  printf('<div class="toplinks">
                                    <a href="?option=menu&id_menu=%s">%s</a>

      </div>',$item['id_menu'],$item['name_menu']);
            }

            echo '<div class="toplinks">
                              <a href="?option=contact">О нас</a></div>';
            echo '</div><div class="copy">
                        <span class="style1"> Copyright © 2014 Самоваров Валерий
</span>
                              </div></div></center></body></html>';
      }

      public function get_body(){

            if($_POST){
                  $this->obr();
            }
            $this->get_header();
            $this->get_left_bar();
            $this->get_menu();
            $this->get_content();
```

```
            $this->get_footer();
      }

      abstract function get_content();

}


?>
```

Листинг файла ACore_Admin.php

```
<?php
abstract class ACore_Admin{

      protected $db;


      public function __construct(){

            if(!$_SESSION['user']){
                  header("Location:?option=login");
            }

            $this->db = mysql_connect(HOST,USER,PASSWORD);
            if(!$this->db){
                  exit("Ошибка соединения с базой данных".mysql_error());
            }
            if(!mysql_select_db(DB,$this->db)){
                  exit("Нет такой базы данных".mysql_error());
            }
            mysql_query("SET NAMES 'UTF8'");
      }

      protected function get_header(){
            include "header.php";
      }

      protected function get_left_bar(){

            echo '<div class="quick-bg">
                        <div id="spacer" style="margin-bottom:10px";>
                              <div id="rc-bg">Админка</div>
                        </div>';

            echo  '<form action="?option=logoff" method="post">
                        <input type="hidden" name="logoff" value="logoff" />
                              <input class="but" type="submit" value="Выход" />
                        </form>';

                  echo '<div class="side-menu">
                              <a href="?option=admin">Статьи</a>
                              </div>';
                  echo '<div class="side-menu">
                              <a href="?option=edit_menu">Меню</a>
                              </div>';
                  echo '<div class="side-menu">
                              <a href="?option=edit_category">Категории</a>
                              </div>';
```

```
                echo '</div>';
        }



    protected function get_menu(){

        echo '<div id="mainarea">
                    <div class="heading"></div>';

    }


    protected function get_footer(){

        echo '<div id="bottom">';

        echo '</div><div class="copy">
                    <span class="style1"> Copyright © 2014 Самоваров Валерий
</span>
                        </div></div></center></body></html>';
    }

    public function get_body(){

        if($_POST || $_GET['del']){
            $this->obr();
        }

        $this->get_header();
        $this->get_left_bar();
        $this->get_menu();
        $this->get_content();
        $this->get_footer();
    }

    abstract function get_content();

    protected function get_categories(){
        $query = "SELECT id_category, name_category FROM category";
        $result = mysql_query($query);
        if(!$result){
            exit(mysql_error());
        }
        $row = array();
        for($i = 0; $i < mysql_num_rows($result); $i++){
            $row[] = mysql_fetch_array($result,MYSQL_ASSOC);
        }
        return $row;

    }

    protected function get_text_statti($id){
        $query = "SELECT id, title, discription, text, cat FROM statti WHERE
id='$id'";
        $result = mysql_query($query);
        if(!$result){
            exit(mysql_error());
        }
```

```
            $row = array();
            $row = mysql_fetch_array($result,MYSQL_ASSOC);

            return $row;

      }

      protected function get_text_menu($id){
            $query = "SELECT id_menu, name_menu, text FROM menu WHERE
id_menu='$id'";
            $result = mysql_query($query);
            if(!$result){
                  exit(mysql_error());
            }
            $row = array();
            $row = mysql_fetch_array($result,MYSQL_ASSOC);

            return $row;

      }

            protected function get_text_category($id){
            $query = "SELECT id_category, name_category FROM category WHERE
id_category='$id'";
            $result = mysql_query($query);
            if(!$result){
                  exit(mysql_error());
            }
            $row = array();
            $row = mysql_fetch_array($result,MYSQL_ASSOC);

            return $row;

      }


}


?>
```

Листинг файла admin.php

```
<?php
class admin extends ACore_Admin{

      public function get_content(){

            $query = "SELECT id,title FROM statti";

            $result = mysql_query($query);
            if(!$result){
                  exit(mysql_error());
            }

            echo "<div id='main'>";

            echo          "<a          style='color:red;          font-size:16px'
href='?option=add_statti'>Добавить новую статью</a><hr>";
            if($_SESSION['res']){
```

```
                  echo $_SESSION['res'];
                  unset($_SESSION['res']);
            }

            $row = array();
            for($i=0; $i < mysql_num_rows($result);$i++){
                  $row = mysql_fetch_array($result,MYSQL_ASSOC);
                  printf('<p style="font-size:14px;">
                              <a   href="?option=update_statti&id_text=%s">%s</a>
|
                              <a                                    style="color:red"
href="?option=delete_statti&del=%s">Удалить</a>
                              </p>',$row['id'],$row['title'],$row['id']);
            }


            echo "</div></div>";
      }
}


?>
```

Листинг файла login.php

```
<?php
class login extends ACore{

      protected function obr(){

            $login = strip_tags(mysql_real_escape_string($_POST['login']));
            $password                                                          =
strip_tags(mysql_real_escape_string($_POST['password']));

            if(!empty($login) AND !empty($password)){
                  $password = md5($password);

                  $query = "SELECT id FROM users WHERE login = '$login' AND
password = '$password'";

                  $result = mysql_query($query);

                  if(!$result){
                        exit(mysql_error());
                  }

                  if(mysql_num_rows($result) == 1){
                        $_SESSION['user'] = TRUE;
                        header("Location:?option=admin");
                        exit();
                  }
                  else{
                        exit("Такого пользователя нет");
                  }

            }


            else{
                  exit("Заполните обязательные поля");
```

```
            }
      }


      public function get_content(){



            echo '<div id="main">';

print <<<HEREDOC
<form enctype='multipart/form-data' action='' method='POST'>
<p style='color:blue;font-size:16px;'>Авторизация</p><hr>
<p>Логин:<br />
<input type='text' name='login'>
</p>
<p>Пароль:<br />
<input type='password' name='password'>
</p>
<input type='submit' name='button' value='Войти'></p></form>
HEREDOC;

            echo '</div></div>';
      }


}

?>
```

Листинг файла main.php

```
<?php
class main extends ACore{

      public function get_content(){

            echo '<div id="main">';


            $query = "SELECT id, title, discription, date, img_src FROM statti
ORDER BY date DESC";

            $result = mysql_query($query);
            if (!$result){
                  exit(mysql_error());
            }

            $row = array();
            for($i=0; $i < mysql_num_rows($result);$i++){
                  $row = mysql_fetch_array($result,MYSQL_ASSOC);
                  printf('<div   style="margin:10px;   border-bottom:2px   solid
#009999">
                              <p style="font-size:18px">%s</p>
                              <p>%s</p>
                              <p><img   style="margin-right:5px"   width="150px"
align="left" src="%s">%s</p>
                              <p><a         href="?option=view&id_text=%s">Читать
далее...</a></p>
                  </div>
```

```
      ',$row['title'],$row['date'],$row['img_src'],$row['discription'],$row['id'
]);
            }


            echo '</div>
                              </div>';
      }


}

?>
```

Листинг файла search.php

```
<?php
class search extends ACore{

      protected function obr(){


function search ($query) {
$text = '';

// Проводим фильтрацию данных
$query = trim($query);                      // Обрезаем пробелы и спецсиволы
$query = strip_tags($query);                // Удаляем HTML и PHP теги
$query = mysql_real_escape_string($query); // Экранируем специальные символы


//Поисковый запрос не пустой?
if (!empty($query)){
  if (strlen($query) < 4) {
    $text = '<p>Поисковой запрос должен быть длиной более 4 символов</p>';
  }elseif (strlen($query) > 128) {
    $text = '<p>Поисковой запрос должен быть длиной менее 128 символов</p>';
  } else {
    //Формируем строку поискового запроса
    $sql = "SELECT `id`, `title`, `discription`
      FROM `statti` WHERE `discription` LIKE '%$query%'
      OR `title` LIKE '%$query%'";
    // и выполняем его
    $result = mysql_query($sql);

    if(!$result){
            exit(mysql_error());
      }
    //Определим колличество найденных совпадений

      $num = mysql_num_rows($result);
    //Если число совпадений (строк результата запроса) больше 0
            if ( $num > 0) {
            //Получаем ассоциативный массив
           $row = mysql_fetch_assoc($result);
      //и начинаем формировать строку поисковой выдачи
      $text .= '<p>По вашему запросу  <strong>'.$query.'</strong>';
      $text .= ' найдено '.$num.' совпадений</p>' ;

      do {
```

```
        //Продолжаем  формировать строку поисковой выдачи
       // $text .= '<p><a href="?option=view&id_text="'.$row[]'>'
        $text .= '<p><a href="?option=view&id_text='.$row['id'].'">';
        $text .= $row['title'].'</a></p>';
        $text .= '<p>'.$row['discription'].'</p>';
        // Делаем это пока у нас есть результаты
      } while ($row = mysql_fetch_assoc($result));
    }
    else
    {
      // Найти совпадение не удалось
      $text = '<p>По вашему запросу ничего не найдено.</p>';
    }
  }
}else {
  $text = '<p>Задан пустой поисковый запрос.</p>';
}
//Возвращаем сформированную строку поисковой выдачи
return $text;
}



            }

            public function get_content(){


            echo '<div id="main">';


            if (isset ($_POST['query'])){
            $search_result = search ($_POST['query']);
            echo $search_result;


            }
            echo '</div></div>';
      }

                  }

?>
```

Листинг файла view.php

```
<?php
 class view extends ACore{

      public function get_content(){

            echo '<div id="main">';

            if(!$_GET['id_text']){
                  echo "Не верные данные для вывода статьи";
            }
            else{
                  $id_text = (int)$_GET['id_text'];
                  if(!$id_text){
                        echo "Не верные данные для вывода статьи";
```

```
                  }

                  else{
                        $query = "SELECT title,text,date,id,img_src FROM statti
WHERE id='$id_text'";

                        $result = mysql_query($query);
                        if (!$result){
                              exit(mysql_error());
            }
            $row = mysql_fetch_array($result,MYSQL_ASSOC);
            printf('<p style="font-size:18px">%s</p>
                              <p>%s</p>
                              <p><img   style="margin-right:5px"   width="150px"
align="left" src="%s">%s</p>',

      $row['title'],$row['date'],$row['img_src'],$row['text']);
            }

      }
            echo '</div>
                              </div>';
      }

      }
?>
```