МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»
Кафедра IT-инжиниринг

ДОПУЩЕН К ЗАЩИТЕ
Заведующий кафедрой
PhD, доцент
_____ Т.С. Картбаев
«____» _____2019 г.

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка информационной системы для ТОО «Аслан Курылыс»

Специальность:5В070400 – «Вычислительная техника и программное обеспечение»

Выполнил:Тұрған Р.Қ. Группа:ВТу-16-4
Научный руководитель: ст.преп., Мусатаева Г.Т.

Консультанты:
по экономической части:к.э.н., профессор_____Ж.Г. Аренбаева
«22» мая 2019 г.

по безопасности
жизнедеятельности:д.т.н., ст.преп._____Ш.Ш. Бекбасаров
«22» мая 2019 г.

по применению
вычислительной техники:ст. преп._____М.Н. Майкотов
«20» мая 2019 г.

Нормоконтролер:ассист._____ Ш.П. Жумагулова
«24» 05 2019 г.

Рецензент: к.т.н., зав. каф._____В.В. Сербин
«____» _____ 2019 г.

Алматы 2019

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»

Институт систем управления и информационных технологий

Кафедра IT-инжиниринг

Специальность 5В070400 – «Вычислительная техника и программное обеспечение»

ЗАДАНИЕ
на выполнение дипломного проекта

Студенту Тұрған Руслан Қордайұлы

Тема проекта: Разработка информационной системы для ТОО «Аслан Курылыс»

Утверждена приказом по университету № 124 от «26» октября 2018 г.

Срок сдачи законченного проекта «24» мая 2019 г.

Исходные данные к проекту (требуемые параметры результатов исследования (проектирования) и исходные данные объекта): Руководство системы менеджмента качества на предприятии; международные стандарты ИСО-9001, данные преддипломной практики.

Перечень вопросов, подлежащих разработке в дипломном проекте, или краткое содержание дипломного проекта:

- аналитическая часть;
- проектная часть;
- экспериментальная часть;
- экономическая часть;
- безопасность жизнедеятельности;
- приложение А. Техническое задание;
- приложение Б. Листинг программы;
- приложение В. Акт внедрения.

Перечень графического материала (с точным указанием обязательных чертежей): представлены 15 таблиц, 20 иллюстрации.

Основная рекомендуемая литература:

1 Коровкин С. Д., Левенец И. А., Ратманова И. Д., Старых В. А., Щавелёв Л. В. Решение проблемы комплексного оперативного анализа информации хранилищ данных // СУБД. - 1997. - № 5-6. - С. 47-51.

2 Распределенная обработка данных: курс лекций / Сост. Найханова Л.В. – Улан-Удэ, Издательство ВСГТУ, 2001. – 122 с.

3 Гофман В.Э. Хомененко А.Д. Delphi 7 – СПб.: БХВ – Санкт-Петербург, 2005 – 1152с.

4 Гофман В.Э., Хомененко А.Д. Работа с базами данных в Delphi. – СПб.: БХВ – Петербург, 2000. – 656 с.

Консультации по проекту с указанием относящихся к ним разделов проекта

Раздел	Консультант	Сроки	Подпись
Экономическая часть	Аренбаева Ж.Г.	04.03.2019 – 22.05.2019	
Безопасность жизнедеятельности	Бекбасаров Ш.Ш.	04.03.2019 20.05.2019	
Программное обеспечение	Майкотов М.Н.	02.05 – 20.05.	
Нормоконтролер	Жумагулова Ш.П.	03.04.2019 – 24.05.2019	

ГРАФИК
подготовки дипломной проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Аналитическая часть	05.11.18г – 22.12.18г.	
Проектная часть	07.02.19г – 30.01.19г.	
Экспериментальная часть	04.02.19г – 13.04.19г	

Дата выдачи задания «14» января 2019 г.

Заведующий кафедрой _____ Т.С. Картбаев

Научный руководитель проекта _____ Г.Т. Мусатаева

Задание принял к
исполнению студент _____ Р.Қ. Тұрған

Аңдатпа

Дипломдық жобаның тақырыбы: ««Аслан Құрылыс» ЖШС үшін ақпараттық жүйе құру».

Дипломатиялық жобаның мақсаты бар және әлеуетті клиенттермен жұмыс істеу және компанияның өнімдерін насихаттау үшін «Аслан Құрылыс» ЖШС корпоративтік толыққанды WEB-сайтын құру болып табылады.

Диссертациялық жұмыс кіріспеден, 4 тараудан және қорытынды қорытындыдан тұрады.

Іріктемеде таңдап алынған тақырыптың өзектілігі анықталған, оның мақсаты - орындалуы тиіс тапсырмаларды әзірлеу және орындау.

Бірінші тарау веб-сайттарды талдайды және тақырыптық аймақты сипаттайды.

Екінші тарауда бағдарламалық өнімнің дизайны мен проблеманы шешуге арналған технологияларды таңдау сипатталады.

Үшінші тарау ағымдағы жобаның шеңберінде еңбек жағдайларын жақсарту жөніндегі шараларды қарастырады.

Төртінші тарау - әзірленген жобаның экономикалық негіздемесін негіздеу.

Қорытындыда жасалынған жұмыстың негізгі қорытындылары келтірілген.

Анотация

Тема дипломного проекта: «Разработка информационной системы для ТОО «Аслан Курылыс»».

Цель дипомного проекта – является создание корпоративного полнофункционального WEB – сайта для ТОО «Аслан Курылыс» для работы с существующими и потенциальными клиентами и продвижению продуктов фирмы.

В дипломный проект входит введение, 4 главы и итоговое заключение.

Во введении раскрывается актуальность выбранной темы, ставится цель разработки и задачи, которые необходимо выполнить.

В первой главе производится анализ Web – сайтов и описывается предметная область.

Во второй главе описывается проектирование программного продукта и выбор технологий для решения задачи.

В третьей главе рассматривается мероприятия по улучшению условий труда в рамках реализуемого проекта.

В четвертой главе производится обоснование экономической целесообразности разрабатываемого проекта.

В заключении представлены основные выводы проделанной работы.

Annotation

Theme of the graduation project: "Development of an information systems for «Aslan Kurylys»LLP."

The purpose of the diplomatic project is to create a corporate full-featured WEB site for Aslan Kurylys LLC to work with existing and potential customers and promote the company's products.

The thesis project includes an introduction, 4 chapters and a final conclusion.

In the introduction reveals the relevance of the chosen topic, the goal is to develop and the tasks that must be performed.

The first chapter analyzes Web sites and describes the subject area.

The second chapter describes the design of the software product and the choice of technologies for solving the problem.

The third chapter deals with measures to improve working conditions in the framework of the ongoing project.

The fourth chapter is the rationale for the economic feasibility of the developed project.

In conclusion, presents the main conclusions of the work done.

Содержание

Введение

Web–сайты применяются в виде устройства общения среди владельцев сайта и его пользователями, а иногда – среди самих пользователей. Обладатели сайтов обычно ставят задачу и определяют существенные правила взаимодействия, когда как пользователи – это те люди, которые посещают сайт и пользуются представленным на нем содержанием или его возможностями. Канал связи между владельцем сайта и его посетителем может изменяться. Зачастую владельцы сайтов предоставляют пользователям информацию для ее потребления, делая из этого отчасти одностороннее взаимодействие.

Web–сайты можно рассортировать по нескольким широким категориям: Информационные сайты. На таких сайтах представлена информация по конкретной теме или об определенной организации ("буклетное обеспечение").

Это самые распространенные в сети Internet Web–сайты; с течением времени они зачастую перенимают некоторые черты других категорий сайтов.

Операционные сайты. Сайтом такого типа можно воспользоваться с целью выполнения какой–либо операции или задачи. В эту категорию входят сайты, занятые в электронной коммерции.

Сайты сообществ. На этих сайтах представлена информация или средства, связанные с осуществлением операций, но упор делается на взаимодействие между посетителями. Сайты, основанные на сообществах, имеют тенденцию к фокусированию на конкретной теме или человеке; они поощряют взаимодействие между сходно мыслящими личностями.

Развлекательные сайты. Эти сайты создаются для игр или некоего занимательного взаимодействия, для которого могут употребляться элементы операционного, информационного типов и сайтов сообществ.

Кроме этого, можно сгруппировать сайты на основе организаций, которые поддерживают или в каком–то смысле платят за сайт. В рамках этого типа классификации мы усматриваем пять основных групп.

Корпоративные сайты. Сайт из этой группы создается и поддерживается организацией или индивидуумом для получения коммерческой выгоды – либо напрямую посредством электронной коммерции, либо косвенно через стимулирование приобретения товаров или услуг вне Internet.

Правительственные сайты. Вышестоящим органом по отношению к такому сайту в конечном итоге является правительственная организация, а назначением сайта является удовлетворение какой–либо общественной или правовой потребности.

Образовательные сайты. Сайт такого типа курирует некое образовательное учреждение (возможно, имеющее отношение к правительственным органам) он используется для обеспечения образовательных или исследовательских задач.

Филантропические сайты. Филантропический сайт существует с целью продвижения целей некоммерческой организации или благотворительной деятельности частного лица или организации.

Персональные сайты. Такой сайт существует исключительно по усмотрению некоего человека или группы людей по любым причинам, обычно являясь плодом выплеска творческой энергии или формой самовыражения личности.

Классификация может оказаться сложной задачей. К примеру, образовательные сайты на самом деле могут попадать в категорию правительственных. Некоторые сайты из категории персональных могут, вероятно, принадлежать к группе филантропических или коммерческих – в зависимости от причины, по которой человек берется за создание сайта.

Целью данной работы является создание корпоративного полнофункционального WEB–сайта для ТОО «Аслан Курылыс» для работы с существующими и потенциальными клиентами и продвижению продуктов фирмы.

1. Основная часть

1.1 Описание предметной области

Большинство дискуссий относительно Web–дизайна, как правило, быстро уходит от своей основной темы, т.к. этот термин разные люди понимают по–своему. Хотя каждый имеет некоторое представление о том, что такое Web–дизайн, лишь немногие способны дать ему точное определение. Определенные компоненты, такие как графический дизайн или программирование, затрагиваются в каждой дискуссии, однако их важность в процессе создания сайтов варьирует от человека к человеку и от сайта к сайту. Некоторые считают наиболее важной стороной Web–дизайна создание и структурирование содержимого – или, выражаясь более формально, информационную архитектуру. Другие факторы – среди множества прочих, простота использования, ценность и назначение сайта в рамках общей деятельности организации, доставка сайта – определенно пребывают в сфере Web–дизайна. Набравшись влияний из библиотечного дела, графического дизайна; программирования, организации сетей, проектирования пользовательских интерфейсов, удобства использования и множества других источников, Web–дизайн является мульти–дисциплинарной областью деятельности.

Актуальность темы обусловлена необходимостью создания информационного web–сайта для ТОО «Аслан Курылыс», поскольку это позволяет заявить о себе в полном объеме, предоставить полную информацию о своих возможностях, услугах и ценах, иметь собственный Интернет–ресурс.

Web–сайт включает в себя следующие автоматизированные функции:

- Приток новых клиентов;
- Информирование;
- Увеличение прибыли;
- Рост имиджа предприятия.

1.1.1 Определение Web–дизайна

Пять областей охватывают основные аспекты Web–дизайна:

Содержимое. Сюда входят форма и организация содержимого сайта. Возможный диапазон – от того, как написан текст до того, как он организован, представлен и структурирован с помощью технологии разметки, такой как HTML.

Зрительные образы. Это относится к компоновке экранного пространства на сайте. Эта компоновка обычно создается с помощью HTML, CSS, JavaScript, Flash и может включать графические элементы, выполняющие функции украшения или навигации. Визуальная сторона сайта – это наиболее очевидный аспект Web–дизайна, но не единственная, и не самая важная, сторона дисциплины.

Технология. Хотя применение разнообразных базовых Web–технологий вроде HTML или CSS попадает в эту категорию, под технологией в этом контексте чаще подразумеваются различные интерактивные элементы сайта, в особенности созданные с использованием программных методов. Это могут быть элементы в диапазоне от языков сценариев, работающих на стороне клиента, наподобие JavaScript, до серверных приложений, таких как Java–сервлеты.

Доставка. Скорость и безотказность доставки сайта по сети Internet или внутренней корпоративной сети связаны с применяемым аппаратным или программным обеспечением и задействованной сетевой архитектурой.

Назначение. Причина, по которой сайт существует, часто связанная с экономическими вопросами, вероятно, является наиболее важной частью Web–дизайна. Этот элемент следует учитывать при принятии любых решений, затрагивающих другие области.

Степень, в которой каждая сторона Web–дизайна оказывает воздействие на сайт, может изменяться в зависимости от типа создаваемого сайта. Личная домашняя страничка обычно не связана с экономическими соображениями, характерными для Internet–магазина. Внутренняя сеть производственной компании может не попадать под влияние соображений, связанных с визуальным представлением, важных для обще–доступного Web–сайта, рекламирующего остросюжетный фильм.

1.1.2 Пирамида Web–дизайна

Все компоненты Web–дизайна можно представить через метафору Web–пирамиды, показанной на рисунке 1.1. Содержимое – это кирпичи, составляющие пирамиду, а фундамент основан на зрительных образах и технологии, при этом в значительной степени основываясь на экономике, благодаря которой нашим проектом стоит заниматься.

Рисунок 1.1 – Пирамида Web–дизайн

Как Web–дизайнеры мы пытаемся тщательно планировать наши сайты, однако конструировать их сложно. Web–технологии делают строительство нашего сайта сложной задачей, для ее выполнения необходимо коллективная работа и устойчивое понимание сетевой среды. Даже если мы – специалисты способные создать красивый и функциональный Web–сайт, наши пользователи могут посмотреть на наше прекрасное создание с замешательством.

Дизайнеры, часто больше времени тратят на обдумывание собственных нужд и желаний, чем желаний посетителей их сайта. Хотя уровень проблем, связанных с Web–разработкой не столь большой, все же создать функциональный, привлекательный Web–сайт, который может выдержать испытание Internet–временем, безусловно, непросто. Пирамида позволяет дизайнерам с легкостью представить все стороны Web–дизайна в их взаимодействии, но не может предложить почти ничего, чтобы обеспечить более глубокое понимание сетевой среды. Пирамида позволяет дизайнерам с легкостью представить все стороны Web–дизайна в их взаимодействии, но не может предложить почти ничего, чтобы обеспечить более глубокое понимание сетевой среды.

1.2 Сетевая среда

Несмотря на то, что аналогия с Web – пирамидой – это очень отвлеченный способ описания Web – дизайна, это полезное средство демонстрации взаимодействия различных компонентов здания Internet. Более практичным вариантом размышления о Web – дизайне представляется обсуждение разнообразных составных частей сетевой среды, показанных на рисунке 1.2.

Сегодняшние Web – сайты, в основном, относятся к базовой модели сетевого клиент–серверного программирования с тремя общими элементами.

Сторона сервера. Сюда входит аппаратное и программное обеспечение Web–сервера, а так же программные элементы и встроенные технологии. Диапазон этих технологий простирается от простых программ CGI, написанных на Perl, до комплексных многозвенных приложений на основе PHP. Здесь же учитываются прикладные технологий , например – серверы ба данных, которые могут обеспечивать поддержку Web – сайта.

Сторона клиента. Сторона клиента связано с Web – браузером и поддерживаемыми им технологиями, такими как языки HTML, CSS и JavaScript, элементы управления ActiveX и сменные модули Explorer, которые используются для создания представления страницы или обеспечения интерактивных функций.

Сеть. Сеть характеризует различные элементы могут быть различные сети в общедоступной части сети Internet или частые соединения внутри корпорации, которые зачастую называются внутренней сетью.

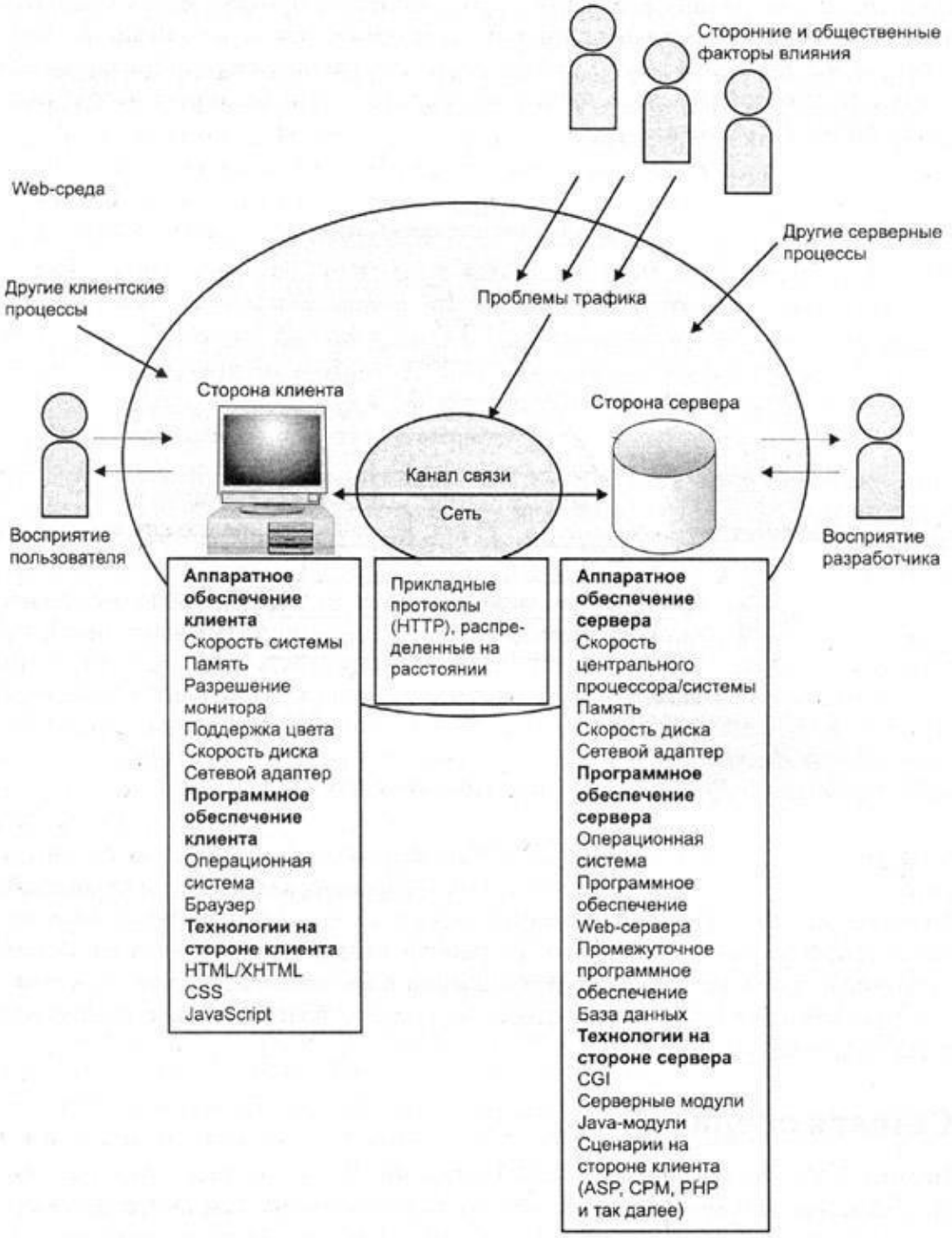

Рисунок 1.2 – Компоненты сетевой среды

Сеть. Сеть характеризует различные элементы могут быть различные сети в общедоступной части сети Internet или частые соединения внутри корпорации, которые зачастую называются внутренней сетью.

Детальное понимание технических аспектов сетевой среды, включая

сетевой компонент, имеет огромное значение. Схема Web – пирамиды вновь напоминает нам о важности пользовательского компонента, так как на самом деле Web – дизайн представляет собой род деятельности, связанный с сетевым программированием и определенными вопросами пользовательской направленности. В других случаях пользователи имеют возможность публиковать информацию на рассмотрения владельцев сайта или даже других пользователей, тем самым, создавая более многосторонний канал связи, показанный на рисунке 1.2.1.

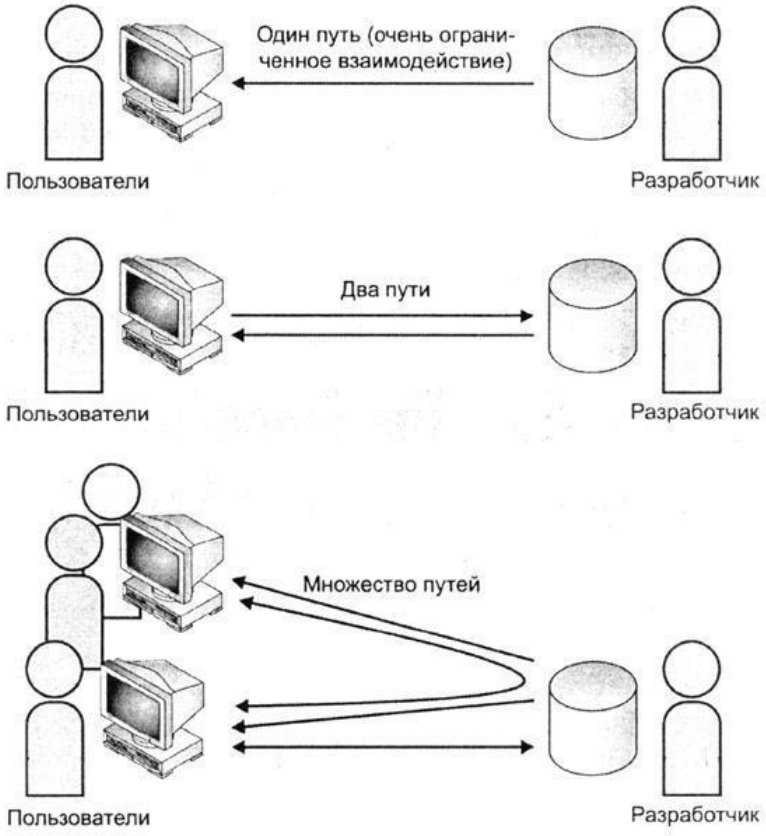

Рисунок 1.2.1 – Многосторонний канал связи

Как правило, в ходе любого процесса передачи информации большинству пользователей, если все работает нормально, ничего не известно о среде. Хотя пользователи находятся под воздействием среды, они часто не проводят различия между отдельными компонентами, такими как сеть.

1.3 Типы Web–сайтов

Пользователи имеют тенденцию к тому, чтобы обсуждать Web–сайты, а, следовательно, – и дизайн Web–сайтов, по назначению сайта или по его внешнему виду. Важно уметь характеризовать сайты именно таким образом. Есть множество других способов их категоризации. Хотя возможные

категории сайтов могут показаться бесконечными, мы можем, ничем не рискуя, сгруппировать сайты несколькими общими способами.

В первую очередь нужно определиться с тем, ориентирован ли сайт на получение информации или на выполнение задачи. Иногда мы можем характеризовать это разграничение как различие между документно–ориентированным и проблемно– ориентированным сайтами. Документно–ориентированные, или информационные сайты предоставляют пользователям информацию, но при этом обеспечивают очень ограниченный уровень интерактивности (кроме наделения пользователя возможностями просмотра, поиска или сортировки представленной информации). Прикладные, или проблемно– ориентированные сайты позволяют пользователю взаимодействовать с информацией или выполнять какую– либо задачу, например, перевод денежных средств с банковского счета или покупка нового свитера. На гибридных сайтах есть понемногу от обеих функций; по мере того как разделительная черта между информацией и прикладной задачей размывается, этот тип сайтов становится все более распространенным. На рисунке 1.3 изображена схема перехода от простого, статичного документно–ориентированного сайта к полнофункциональным программным приложениям.

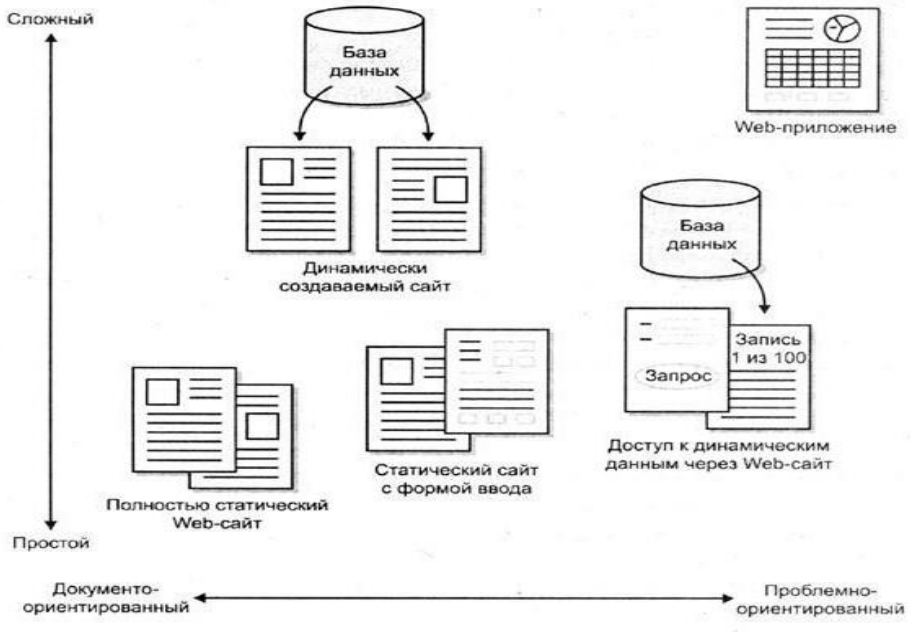

Рисунок 1.3 – Номенклатура Web–сайтов

Такая отвлеченная классификация предполагает, что существует переход от более документно–ориентированных или печатно –ориентированных Web–сайтов к более интерактивным программным Web–сайтам. Это действительно так; точка пересечения между этими двумя философскими "лагерями" является источником серьезных разногласий и нововведений в сообществе Web–

дизайнеров.

1.4 Визуальные классификации

Группируя сайты по визуальному признаку, мы сталкиваемся с диапазоном, на одной стороне которого – сайты, которые в более значительной степени основываются на текстовом содержимом, а на другой – те, что больше фокусируются на графическом представлении или изображениях. В Internet есть четыре наиболее распространенных дизайнерских сайтов:

Тексто – ориентированные сайты. Такие сайты проектируются с упором на текстовое содержимое. Они, относительно легковесны, без труда загружаются, а их дизайн слегка минималистичен;

Сайты в стиле GUI. На этих сайтах соблюдаются некоторые соглашения, связанные с графическим пользовательским интерфейсом (GUI – graphical user interface), взятые из проектирования программного обеспечения, например выровненные кверху строки меню, пиктограммы и всплывающие окна. Среди GUI–ориентированных сайтов есть и простые GUI–компоненты, добавляемые по большей части в тексто–ориентированные сайты, и полнофункциональные Web–приложения со специальными элементами пользовательского интерфейса;

Метафорические сайты. Метафорические сайты черпают идеи из "реальной жизни". К примеру, в дизайне и навигации сайта, посвященного автомобилям, могут быть задействованы приборная панель и рулевое колесо. Метафорически оформленный сайт, как правило, чрезвычайно ярок визуально или интерактивен. Это может вводить в заблуждение одних пользователей и привлекать других;

Экспериментальные сайты. Создатели таких сайтов пытаются делать вещи, немного отступая от норм. Следующих экспериментальному стилю оформления, нередко задействуются творчество, непредсказуемость, новаторство и даже хаотичность.

Безусловно, в Internet мы находим смешение форм и потенциально новую классификацию сайтов. К примеру, как можно классифицировать портал, который обеспечивает богатство содержимого, вариантов навигации и даже функций, связанных с сообществами, и все это на одной странице? Бесспорно, этот дизайнерский стиль применяется на огромном количестве сайтов. Мы наблюдаем скрытое повышение влияния других категорий дизайна – для этого достаточно взглянуть на такие жанры Web–дизайна, как сайты электронной коммерции, в особенности чистые сайты типа "каталог и корзина", а также сайты персональных сетевых журналов.

1.5 Назначение разработки

Корпоративные сайты создаются преимущественно для поддержки

бизнеса какой– либо организации. Основной аудиторией корпоративного сайта являются потенциальные и настоящие клиенты организации. Дополнительная аудитория часто включает потенциальных и действительных инвесторов, служащих и заинтересованные стороны организации, такие как средства информации. С учетом такой смешанной аудитории основные цели корпоративных сайтов включают:

– распространение основной информации – сайт используется для распространения информации о продукции и услугах, предоставляемых организацией. Другая предоставляемая информация в основном включает данные о том, как связаться с фирмой способами, отличными от Web;

– поддержку – часть сайта может быть предназначена для оказания клиентам информационной помощи относительно эффективного пользования продуктами и услугами, предоставляемых организацией;

– связи с инвесторами – открытые акционерные общества или компании, ищущие внешних инвесторов, могут создать сайт или раздел сайта для опубликования информации о текущей экономической ситуации в компании, а также о будущих возможностях для вложения инвестиций;

– связь с общественностью – многие фирмы используют Web–сайты для предоставления информации различным организациям, занимающимся сбором данных, а также добровольного обнародования основной информации о фирме;

– поиск служащих – Web–сайты часто используются для размещения объявлений о приеме на работу и преимуществах работы в компании.

1.5.1 Требования к надежности и эффективности

Web–сайт должен иметь дружественный пользовательский интерфейс и должен позволять работать пользователю, не обладающему специальными знаниями информационных технологий.

Для обеспечения устойчивого функционирования Web–сайта необходимо:

– осуществлять проверку его совместимости с различными браузерами;

– осуществлять проверку его совместимости с различными кодировками операционных систем;

– осуществлять проверку его совместимости с разрешениями экрана;

– для надежности сохранять данные в сессии, в специальных файлах, которые содержат информацию, хранящуюся в базе данных;

– обеспечивать восстановление информации утраченной в результате сбоя в работе сервера.

Для эффективной работы Web–сайта необходимы следующие условия:

– качество работы сервера, где происходит хранение Web–сайта;

– обеспечивать восстановление информации утраченной в результате сбоя в работе сервера.

– возможность отката или отмены внесенных изменений, а также возможность перехода к изначальным входным данным.

Web–сайт должен быть легко проверяем на предмет работоспособности, должны быть подготовлены специальные тесты с заранее известными результатами правильной работы сайта. Тесты должны предусматривать проверку работы форума – вход пользователя, регистрацию, поиск по форуму и добавление новых сообщений, проверку гостевой книги, проверку работоспособности анкеты – ее реакцию на ввод неверных типов данных, проверку работы прейскуранта – изменение и ввод данных, проверку счетчика посещений и проверку взаимодействия администраторской панели с пользовательской частью.

1.5.2 Требования к составу выполняемых разделов

Web–сайт необходимо разбить на несколько разделов.

Раздел «Главная» – отображает краткую деятельность компании, последние новости.

Раздел «Портфолио» – один из наиболее важных разделов сайта – отображает список готовых работ компании, необходим для наглядной демонстрации дизайнерских возможностей фирмы.

Раздел «Гостевая книга» состоит из подразделов «Форум» и «Гостевая книга» – эти разделы необходимы для интерактивного общения и дискуссий между пользователями сайта.

Раздел «Услуги» состоит из подразделов «Прейскурант» и «Калькулятор» – также, один из важных разделов предназначенный для потенциальных клиентов, которые могут просмотреть в этих разделах виды услуг и цены на них. Для удобства пользователя предоставляется калькулятор с автоматическим подсчетом необходимых ему услуг.

Раздел «Контакты» состоит из подразделов «Схема проезда» – пользователь имеет возможность просмотреть, где находится компания и как с ней связаться, и подраздел «О компании» содержит в себе информацию о сотрудниках и деятельности компании в целом.

Раздел «Вакансии» – имеющиеся вакансии в компании. Для удобства пользователя и администратора предусмотрена анкета. После заполнения пользователем анкета помещается в базу данных для дальнейшего хранения или просмотра.

Раздел «Информация» содержит аналитическую информацию для потенциальных клиентов или просто заинтересованных лиц.

1.5.3 Требования к организации входных и выходных данных

Для отображения Web–сайта в окне пользовательского браузера необходимо, чтобы происходил обмен входными и выходными данными между пользователем и сервером – этот процесс называется доставкой.

Доставка сайта пользователю так же важна, как и его построение. Популярность сайта очень сильно зависит от времени его отклика на действия посетителя, что в свою очередь отражается на отношении к нему конечного пользователя. Большинство дизайнеров осознает необходимость в скорости. Даже в этом случае, что бы объяснить низкую скорость загрузки сайта, дизайнеры обычно акцентируют внимание на нескольких аспектах: размер графических файлов сайта или скорость подключения конечного пользователя. Скорость загрузки может определяться множеством факторов, например, особенностями сети: трафик, вид протокола, вид сервера, а также содержание сайта. Дизайнеры должны определять все аспекты доставки сайта, потому что конечный пользователь не может разделять отдельные компоненты передачи информации, а рассматривает сайт как единую систему.

Даже когда все передается нормально, работа над Web–сайтом может продолжаться и отнимать достаточно много времени. Web–мастер всегда найдет, чем заняться: обслуживанием содержания, восстановлением неработающих ссылок, а также проверкой доступности сайта. Один из интересных аспектов сопровождения сайтов – анализ его использования. В Web существует возможность понять, что делают пользователи при посещении сайта, путем анализа файлов системного журнала. Можно проанализировать эти данные для улучшения дизайна.

Процесс запроса страницы проходит приблизительно следующим образом: пользователь вводит адрес URL;

– формируется HTTP–запрос;

– происходит разрешение доменного имени;

– если разрешение прошло удачно, HTTP–запрос передается на сервер средствами TCP/IP;

– сервер принимает запрос и через некоторое время отвечает;

– формируется и возвращается браузеру либо успешный (к примеру, 200), либо аварийный (к примеру, 404) ответ;

– браузер анализирует поступивший ответ, отображает или сохраняет данные, если ответ содержит другие объекты, процесс повторяется.

Процесс доставки проиллюстрирован на рисунке 1.5.

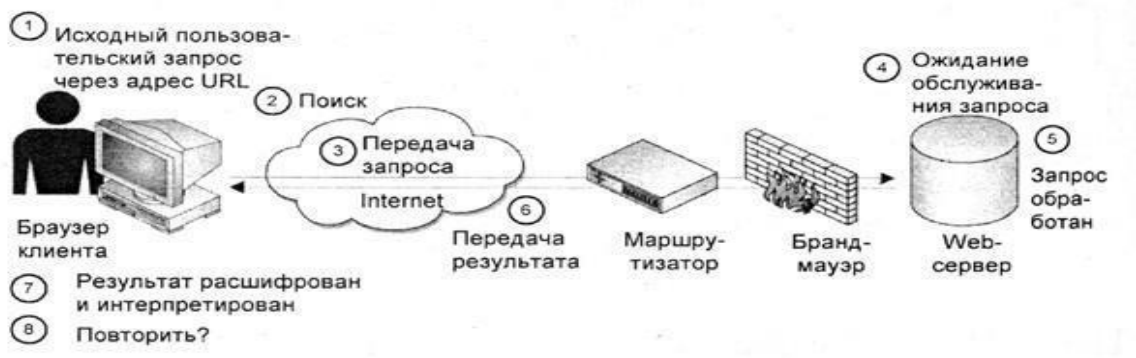

Рисунок 1.5 – Цикл Web–запроса

Обобщенный процесс запроса–ответа, делится на пять этапов:
– формирование запроса и поиск;
– передача запроса;
– выполнение запроса сервером;
– передача результата;
– обработка результата браузером.

1.5.4 Условия эксплуатации

Для успешного функционирования данного Web–сайта необходима мультипроцессорная операционная система, браузер совместимый с Google Chrome, Web–сервер с поддержкой PHP, СУБД mySQL, подключение к глобальной сети Интернет. Система должна быть ориентирована на простого пользователя с навыками работы в web–браузерах.

1.5.5 Состав и параметры технических средств

Технические характеристики сервера (машины) должны соответствовать характеристикам для установленной ОС, подключение к глобальной сети Интернет и наличие Web–сервера и СУБД.

1.5.6 Информационная и программная совместимость

Web–сайт должен быть создан в соответствии с требованиями к модульности. Все блоки программ должны быть впоследствии легко заменяемы с минимальными изменениями в остальных блоках.

Web–сайт должен быть легко загружаем и работать под большинство существующих операционных систем.

2. Техническая часть

2.1. Препроцессор гипертекста – PHP

PHP (англ. PHP: Hypertext Preprocessor – «PHP: препроцессор гипертекста») – язык программирования, созданный для генерирования HTML–страниц на веб–сервере и работы с базами данных. В настоящее время поддерживается подавляющим большинством хостинг–провайдеров. Входит в LAMP – распространённый набор для создания веб–сайтов (Linux, Apache, My SQL, PHP (Python или Perl).

Код PHP может объединяться с тегами XHTML. PHP является встраиваемым языком – это означает, что можно перемещаться между чистым кодом HTML и PHP, не жертвуя возможностью чтения текста. Однако хорошим стилем программирования является отделение программного кода на языке PHP от представления в виде HTML – это чаще всего реализуется с помощью шаблонизации.

2.1.1 Скалярные переменные

Переменные в PHP делятся на два вида, скалярные переменные и массивы, скалярные это те которые хранят одно значение в данный момент, а массивы мы рассмотрим чуть ниже.

Скалярные переменные PHP содержат значения следующих типов.

Целые – целые числа или числа без десятичной точки (1, 999, 325812841).

Числа с плавающей точкой – числа, содержащие десятичную точку (1.11, 2.5, .44).

Строки – текстовая или числовая информация. Строковые данные всегда определяются с помощью кавычек («Hello World», “478–477–5555”).

Булевы значения – используются для значений true (истина) или false (ложь).

Имена переменных PHP всех типов начинаются со знака "$". Имена переменных могут содержать буквы, числа, и символ подчеркивания (_); они не могут, однако, начинаться с цифры. В PHP имена переменных различают регистр символов.

Следующие переменные в PHP интерпретируются как две различные переменные.

$myvar
$MYVAR

Допустимые имена переменных:

$myvar
$F_Name
$address1
$my_string_variable

Недопустимые имена переменных:

Myvar

$1stvar

$&62##

Скалярным переменным PHP присваивают значения в следующем формате:

$username = “jdoe”

$first_name = “John”

$Last_Name = “Doe”

2.1.2 Вывод переменных

Следующий фрагмент кода демонстрирует, как объявить скалярную переменную, присвоить скалярной переменной значение и вывести результаты в окне браузера:

```
<!DCTYPE html PUBLIC "–
//W3C/TD/XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xh
tml1–transitional.dtd">
<html xlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<hea>
<title>Страница Web</title>
<head>
<ody>
<p
<?php
$strin_var = "Мо программа PHP";
$integer_var = 500;
$float_var = 2.25;
echo $sring_var;
echo $iteger_var;
echo $float_var;
?>
</p>
</boy>
</html>
```

Переменные массивы PHP можно создавать и присваивать им значения с помощью конструкции array() или явным образом.

Переменную можно соединять с другими переменными или тегами XHTML с помощью оператора PHP – точки (.). В предыдущем блоке кода значения переменных выводятся в следующем формате:

Моя программа PHP5002.25

Чтобы создать возврат каретки или перенос строки, можно присоединить тег XHTML
 в конце каждой пременной:

```
<?pp
$strig_var = "My PHP program" . "<br/>";
$integr_var = 500 . "<br/>";
$float_ar = 2.25;
echo $sting_var;
echo $intger_var;
echo $loat_var;
?>
```

Теперь после каждой переменной вставляется перенос строки, что приводит к выводу каждого значения на отдельной строке.

```
My PHP Program
500
2.25
```

Соединение переменных:

Оператор точки можно использовать также для соединения строк и переменных:

Сообщение – The user's name is John Doe – выводится в окне браузера.

Строка “The user's name is " соединяется со значением $fname (John), за которым следует пробел “ ", и значением $lname (Doe).

```
<!DOCYPE html PUBLIC "–//W3C//DTD/XHTML 1.0 Transitional//EN"
"htp://ww.w3.org/TR/xhtml1/DTD/xhtml11–transitional.dtd">
<hml xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<had>
<itle>A Web Page</title>
<head>
body>
<>
<?hp
$fnme = "John";
$lnae = "Doe";
echo "Te ser's name is " . $fname . " " . $lname;
?>
</p>
</boy>
</htm>
```

Te uer's nae is John Doe

2.1.3 Подстановка значений переменных

В PHP переменную в строке можно заменить на ее значение. Вместо соединения переменных и литералов, их можно объединять внутри двойных кавычек (""). Подстановку значений можно осуществить только внутри двойных кавычек.

Переменные и литералы нельзя объединить внутри одиночных кавычек. При использовании двойных кавычек значение переменной выводится вместе с литералом. При использовани одиночных кавычек выводится «буквально» имя переменной вместе с остальной строкой.

Следующий пример пояснит приведенные объяснения:

```
<!DOCTYPE html PUBLIC "–//W3C//DTD/XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml11–transitional.dtd">
<hml xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<had>
<ttle>A Web Page</title>
</ead>
<ody>
<>
<?pp
$fnae = "John";
$lnam = "Doe";
echo "Th user's name is $fname $lname";
?>
</p>
</boy>
</htm>
```

Данный код делает такие же выводы, что и пример выше, но тут мы применяли другие переменные, как вы видите это не влияет на конечный исход нашей программы.

Формирование вывода:

Кроме обычного вывода, как делают это все люди практически, можно применять форматированные выводы, принцип такой же, но код другой.

Оператор sprintf показан ниже:

```
<!DCTYPE html PUBLIC "–//W3C//DTD/XHTML 1.0 Transitional//EN"
"htp://www.w3.org/TR/xhtml1/DTD/xhtml11–transitional.dtd">
<tml xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<ead>
<itle>A Web Page</title>
</ead>
<ody>
<p>
<php
$mount = 35;
$ax = 2.50;
$toal = $amount + $tax;
echo "" . spintf("%01.2f", $total);
?>
</p>
```

</bod>
</htl>

Вывод валюты показан ниже:

$37.50

2.1.4 Переменные массивы

В то время как скалярная переменная PHP хранит одно значение, переменную массива можно использовать для хранения множества или последовательности значений. Система PHP поддерживает массивы с числовыми индексами и ассоциативные массивы. Массив в PHP является фактически упорядоченным отображением. Отображение является типом, который отображает значения в ключи. Переменные массивов состоят из двух частей – индекса и элемента.

Индекс массива, иногда называемый ключом массива, является значением, применяемым для идентификации или доступа к элементам массива. Индекс массива помещается в квадратные скобки. Большинство массивов используют числовые индексы, которые обычно начинаются с 0 или 1. В PHP ассоциативные массивы могут использовать строковые индексы. Оба типа массивов создаются с помощью конструкции array().

Массивы с числовыми индексами:

$my_array = array('red', 'green', 'blue');

Этот код создает обычный массив $my_array, в который помещаются три элемента: red, green, и blue. Доступ к каждому элементу можно получить через его номер в массиве:

$my_array[0] = 'red'; // индекс 0 соответствует элементу red

$my_array[1] = 'green'; // индекс 1 соответствует элементу green

$my_array[2] = 'blue';// индекс 2 соответствует элементу blue

Следующий код применяется для вывода значений переменной $my_array.

```
<!OCTYPE html PUBLIC "–//W3C//DTD/XHTML 1.0 Transitional//EN"
"htp://www.w3.org/TR/xhtml1/DTD/xhtml11–transitional.dtd">
<hml xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<ead>
<itle>Страница Web </title>
</ead>
<bdy>
<p>
<hp
$m_array = array('red', 'green', 'blue');
echo "Перое значение массива – " . $my_array[0];
echo "Втоое значение массива – " . $my_array[1];
echo "Треье значение массива – " . $my_array[2];
?>
```

```
</p>
</bod>
</htl>
```

Первое значение массива – red

Второе значение массива – green

Третье значение массива – blue

2.1.4 Ассоциативные массивы

Ассоциативные массивы позволяют использовать массивы в ином формате, где все можно делать описания полной части массива как делается это через обычные массивы, что дает нам дополнительное удобство над описанием нашего кода, лично мне этот вид массива очень по душе по сравнению с обычными, как говорится у каждого свой вкус но этот массив хороший, зачем много говорить лучше посмотрите сами на примере ниже.

```
<!DOCTYPE html PUonal//EN"
"http:/.R/xhtml1/DTD/xhtml11–transitional.dtd">
<html lns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<hed>
<tile>Страница Web </title>
</ead>
<bdy>
<>
<hp
$mers = array('FName' => John, 'LName' => Smith, 'Age' => 50);
echo "The use name is " . $members['FName'];
echo "'s last name is " . $members['LName'];
echo "'s age is " . $members['Age'];
?>
</p>
</boy>
</hml>
```

2.1.5 Функции для работы с массивами

Система PHP включает в себя множество функции и подфункции которые мы рассмотрим ниже.

count() – через это функцию мы считаем число масива.

sort() – через эту функцию мы сортируем наш массив.

shuffle() – через эту функцию мы перемещаем наши элементы в случайном порядке.

sizeof() – это функцию синоним для функции count().

array_slice($array_name,offset, length) – через эту функцию мы извлекаем часть массива.

$array_name – это имя массива которого мы разрежем.

length – число элементов которые мы вырежем из нашего большего массива.

array_merge($array_name, $array_name) –через эту функцию мы объединяем несколько массивов. Следующий код показывает, как применяется каждая из функций для работы с массивами.

```
<!DOCTYPE ht–//W3C//DTD/XHTML 1.0 Transitional//EN"
"http://w.org/TR/xhtml1/DTD/xhtml11–transitional.dtd">
<html xlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<hea>
<titl>Страница Web </title>
</had>
<bdy>
<p>
<?php
//Созданы два массива
$numbrs = array(50,20,18,30,10,7);
$cosarray('red', 'blue', 'green')
//пределяем размер массива $numbers – 6a
ay_s sizeof($numbers);
// сортирем элементы массива $numbers – возвращает array(7,10,18,20,30,50)
sort($mbers);
// случайным образом перемешиваем элементы массива $numbers
shuff($numbers);
// $ged_array возвращает array(7,10,18,20,30,50,'red','blue','green')
$ged_array = array_merge($numbers,$colors);
// езаем номера 18 и 20 из сортированного массива $numbers
// slice содержит array(18,20)
$sce = array_slice($numbers, 2, 2);
?>
</p>
</body>
</html>
```

Пример:

PHP включает также ряд предопределенных или глобальных массивов. Их называют также суперглобальными переменными, так как они всегда присутствуют и доступны для всех блоков сценария PHP. Ниже показаны обычно используемые суперглобальные переменные PHP.

$_GET[]
$_POST[]
$_REQUEST[]
$_COOKIE[]
$_FILES[]
$_SERVER[]

$_ENV[]
$_SESSION[]

Суперглобальные переменные PHP будут описаны в дальнейшем.

Массивы имеют много применений в PHP и программировании в целом.

Этот раздел представил некоторые базовые вопросы массивов PHP и описал некоторые базовые функции: это понадобится при рассмотрении более развитых свойств массивов в следующих разделах.

2.1.6 Константы PHP

Константы, как и переменные, являются временным хранилищем значений в памяти. В отличие от переменных значение константы никогда не изменяется. При объявлении константы используется функция define(), которая требует задать имя константы и значение этой константы.

Константам можно присваивать следующие типы данных.

Целые – целые числа или числа без десятичной точки (1, 999, 325812841).

Числа с плавающей точкой – числа, содержащие десятичную точку (1.11, 2.5, .44).

Строки – текстовая или числовая информация. Строковые данные всегда заключаются в кавычки («Hello World», “478–477–5555”).

Имена констант PHP в отличие от переменных не начинаются со знака "$". Имена констант обычно записывают в верхнем регистре. Имена констант могут содержать буквы, цифры и символ подчеркивания (_); они не могут, однако, начинаться с цифры. Объявление констант показано ниже.

define(«STRING_CONSTANT», “This is my string.”);

define(«NUMERIC_CONSTANT», 5);

Вывод констант:

Следующий фрагмент кода демонстрирует объявление константы, присваивание константе значения и вывод результатов в окне браузера.

```
<!DOCTYPE html PUBLIC "–//W3C//DTD/XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml11–transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>A Web Page</title>
</head>
<body>
<p>
<?php
define("STRING_CONST","My PHP program");
define("INTEGER_CONST",500);
define("FLOAT_CONST",2.25);
echo STRING_CONST;
```

```
echo INTEGER_CONST;
echo FLOAT_CONST;
?>
</p>
</body>
</html>
```

Пример:

My PHP program

500

2.25

В этом примере объявляются значения трех констант: STRING_CONST, INTEGER_CONST и FLOAT_CONST. Затем используется оператор echo для вывода содержимого констант в окне браузера. Кроме вывода в окне браузера, константы можно использовать при выполнении математических и строковых операций PHP.

2.1.9 Функции

Функции используются для разбиения больших блоков кода на меньшие, более управляемые единицы. Содержащийся внутри функции код выполняет определенную задачу и возвращает значение. PHP содержит два типа функций – определенные пользователем (или созданные программистом) и внутренние (встроенные функции), которые являются частью определения языка PHP. Этот раздел посвящен созданию и применению определенных пользователем функций.

Определенные пользователем функции создаются с помощью ключевого слова function. Они особенно полезны в больших программах PHP, так как могут содержать блоки кода, которые могут вызываться или использоваться в программе, что позволяет избежать повторного переписывания кода. Далее представлен пример простой определенной пользователем функции PHP:

```
function AddNumbers($num1,$num2)
{
echo "Это пример функции PHP. Она вычисляет сумму двух чисел и возвраща
ет
результат вызывающей программе";
return $num1 + $num2;
}
```

Поля формы:

Форма вообще состоит из полей, но многие говорят это как элементы управления, про формы можно многое че сказать, форма это голова программы где мы задаем все свои элементы.

Поле Account. Пример страницы содержит текстовое поле, в котором пользователь вводит свое учетное имя. Это стандартное текстовое поле,

созданное с помощью тега <input type="text"> с заданными атрибутами name и size.

<input type="text" name="Account" size="10">

Поле Password. Поля пароля, в котором мы ведем пароль для авторизации, пороль необходим для того что бы обеспечить себя дополнительной безопасностью, а ввобще я рекомендую привязать страницу на другую, это говорится как двухкратная авторизация,

<input type="password" name="Password" size="10">

В этом примере, мы создаем пароль и задаем ему размеры, это очень легко делается, если вы хоть чуть-чуть разбираетесь в программирования, мы понимаете о чем сейчас идет речь.

Кнопка Submit. При щелчке на кнопке передачи submit выполняется действие, определенное параметром ACTION тега <form>. Другими словами, щелчок на кнопке submit передает данные из формы на указанную страницу. Кнопка submit создается с помощью тега <INPUT TYPE="submit">. Кроме того, необходимо задать имя кнопки для ссылки в сценарии и присвоить значение, которое служит в качестве метки, появляющейся на кнопке.

<input type="submit" name="SubmitButton" value="Submit">

После создания страница с формой готова к активации. Вспомните, что эта форма передает информацию об имени учетной записи и пароле на страницу welcome.php. Там эти значения проверяются, чтобы разрешить доступ к странице.

Если предоставлены неправильное имя учетной записи и/или пароль, пользователя немедленно возвращают на страницу login.php, не позволяя увидеть страницу welcome.php.

2.2 Каскадные таблицы стилей CSS

CSS (англ. Cascading Style Sheets – каскадные таблицы стилей) – технология описания внешнего вида документа, оформленного языком разметки.

Через CSS оформляется наш веб сайт, без использования этого языка, наш сайт был бы обычным шалоном без никаких стилей, в кратце можно сказать что это совокупность стилей в таблице, при создании любого конкретного сайта использования CSS необходим, без него можно сказать что сайт не сайт.

2.2.1Что такое CSS?

Каскадные таблицы стилей (Cascading Style Sheets, CSS) – это стандарт, определяющий представление данных в браузере. Если HTML предоставляет информацию о структуре документа, то таблицы стилей сообщают как он должен выглядеть.

Стиль – это совокупность правил, применяемых к элементу гипертекста и определяющих способ его отображения. Стиль включает все типы элементов дизайна: шрифт, фон, текст, цвета ссылок, поля и расположение объектов на странице.

Таблица стилей – это совокупность стилей, применимых к гипертекстовому документу.

Каскадирование – это порядок применения различных стилей к веб–странице. Браузер, поддерживающий таблицы стилей, будет последовательно применять их в соответствии с приоритетом: сначала связанные, затем внедренные и, наконец, встроенные стили. Другой аспект каскадирования – наследование (inheritance), – означает, что если не указано иное, то конкретный стиль будет применен ко всем дочерним элементами гипертекстового документа. Например, если вы примените определенный цвет текста в теге , то все теги внутри этого блока будут отображаться этим же цветом.

Использование каскадных таблиц дает возможность разделить содержимое и его представление и гибко управлять отображением гипертекстовых документов путем изменения стилей.

2.2.2 Общий синтаксис таблиц стилей

Таблицы стилей строятся в соответствии с определенным порядком (синтаксисом), в противном случае они не могут нормально работать. Таблицы стилей составляются из определенных частей (рис. 2.2.1):

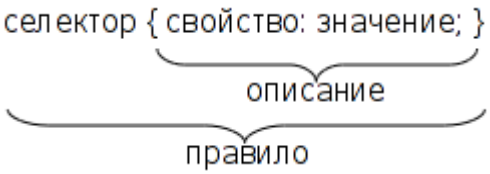

Рис. 2.2 Синтаксис описания стиля CSS

Селектор (Selector). Селектор – это элемент, к которому будут применяться назначаемые стили. Это может быть тег, класс или идентификатор объекта гипертекстового документа.

Свойство (Property). Свойство определяет одну или несколько характеристик селектора. Свойства задают формат отображения селектора: отступы, шрифты, выравнивание, размеры и т.д.

Значение (Value). Значения – это фактические числовые или строковые константы, определяющие свойство селектора.

Описание (Declaration). Совокупность свойств и их значений.

Правило (Rule). Полное описание стиля (селектор + описание).

Таким образом, таблица стилей – это набор правил, задающих значения свойств селекторов, перечисленных в этой таблице. Общий синтаксис описания правила выглядит так:

селектор[, селектор[, ...]] {свойство: значение;}

Регистр символов значения не имеет, порядок перечисления селекторов в таблице и свойств в определении не регламентирован.

2.3 Язык структурированных запросв – SQL

SQL (англ. structured query language – «язык структурированных запросов») – удобный язык ля прогаммирвания предназначенный для создания и управлеия базами данных.

Является инфо-логическим языком который предназначен для описания изменения и извлечения данных, хранимых в реляционных базах данных. SQL вообще считаетс языком для прогаммирования и все программисты используют его для создания базы данных. В самом начале именно через SQL пользователи работали с базой данных.

Со временем SQL усложнился – обогатился новыми конструкциями, обеспечил возможность описания и управления новыми хранимыми объектами (например, индексы, представления, триггеры и хранимые процедуры) – и стал приобретать черты, свойственные языкам программирования.

SQL является самы распростроненным программным средством для разработки базы данных, почти во всех универах, когда проходят тему про базу данных, деаются лабаратоные работы через SQL, так как он прост в исполнении и легко понятен для любого программиста, и многофункционален и именно по этому я выбрал SQL для разработки базы данных для своего сайта.

2.4 Логическая структура Web–сайта ТОО «Аслан Курылыс»

Логическая структура нашего Web–сайта относится к типу «Простая иерархия» (рисунок 2.4)

Раздел «Портфолио» – один из наиболее важных разделов сайта – отображает список готовых работ компании, необходим для наглядной

демонстрации дизайнерских возможностей фирмы.

Раздел «Гостевая книга» состоит из подразделов «Форум» и «Гостевая книга» – эти разделы необходимы для общения и дискуссий между пользователями сайта.

Раздел «Услуги» состоит из подразделов «Прейскурант» и «Калькулятор» – также, один из важных разделов предназначенный для потенциальных клиентов, которые могут просмотреть в этих разделах виды услуг и цены на них. Для удобства пользователя предоставляется калькулятор с автоматическим подсчётом необходимых ему услуг.

Раздел «Контакты» состоит из подразделов «Схема проезда» – пользователь имеет возможность просмотреть, где находится компания и как с ней связаться, и подраздел «О компании» содержит в себе информацию о сотрудниках и деятельности компании в целом.

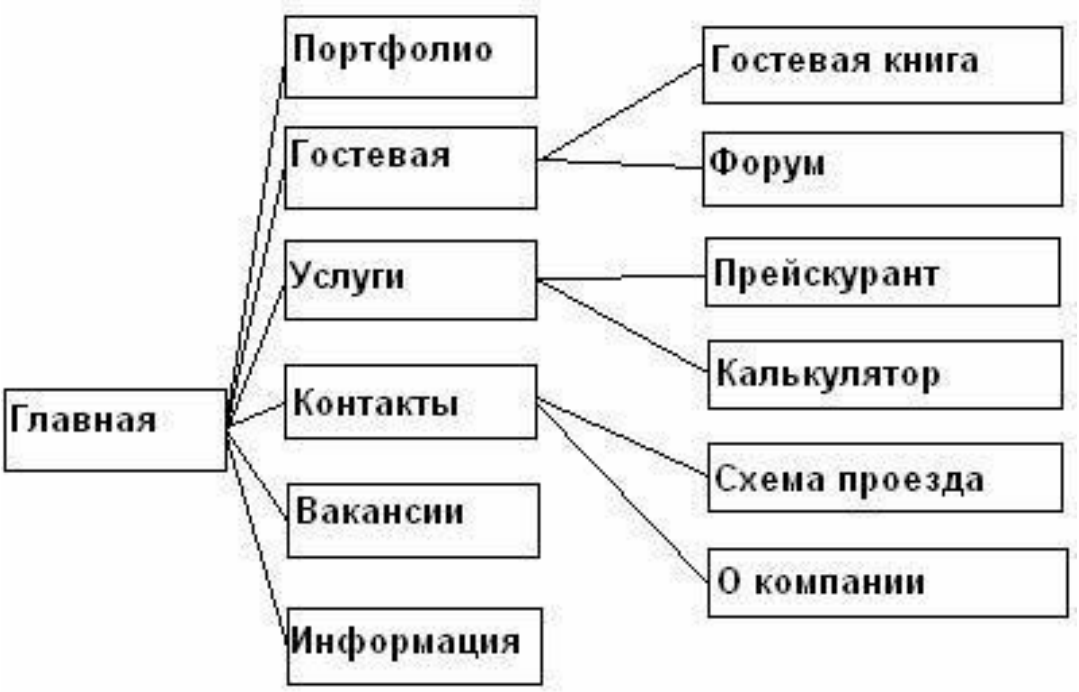

Рисунок 2.4 – Логическая структура сайта «ТОО «Аслан Куылыс»

Раздел «Вакансии» – имеющиеся вакансии в компании. Для удобства пользователя и администратора предусмотрена анкета. После заполнения пользователем анкета помещается в базу данных для дальнейшего хранения или просмотра.

Раздел «Информация» содержит аналитическую информацию для потенциальных клиентов или просто заинтересованных лиц.

Web–сайт содержит основное меню, поэтому можно из любого раздела перейти в любой интересующий раздел. Например, чтобы перейти из раздела «портфолио» в раздел

«схема проезда», необходимо обратиться к основному меню. При этом раздел «схема» загружается в основной фрейм.

2.5 Диаграмма вариантов использования Web–сайта ТОО «Аслан Курылыс»

Диаграмма вариантов использования Web–сайта приведена на рисунке 2.5

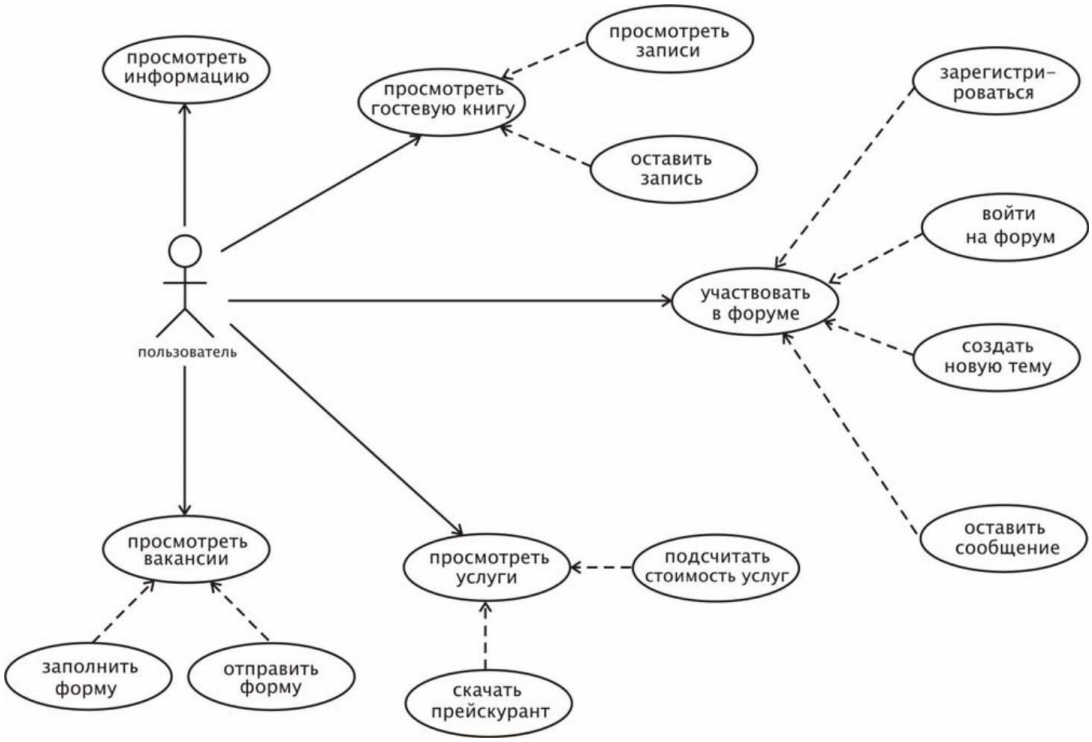

Рисунок 2.5 – Диаграмма вариантов использования

2.6 Диаграмма последовательности действий Web–сайта ТОО «Аслан Курылыс»

Диаграмма последовательности действий Web–сайта приведена на рисунке 2.6

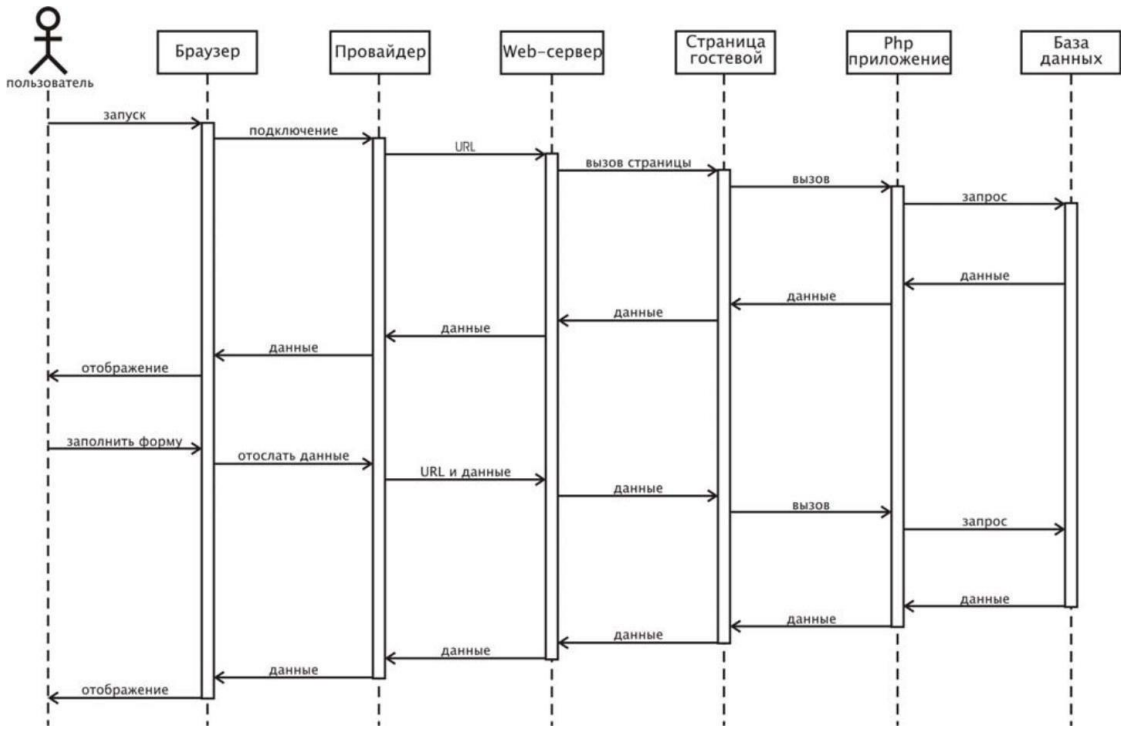

Рисунок 2.6 – Диаграмма последовательности действий

2.7 Структура базы данных Web–сайта ТОО «Аслан Курылыс»

Структура базы данных приведена на рисунке 2.7

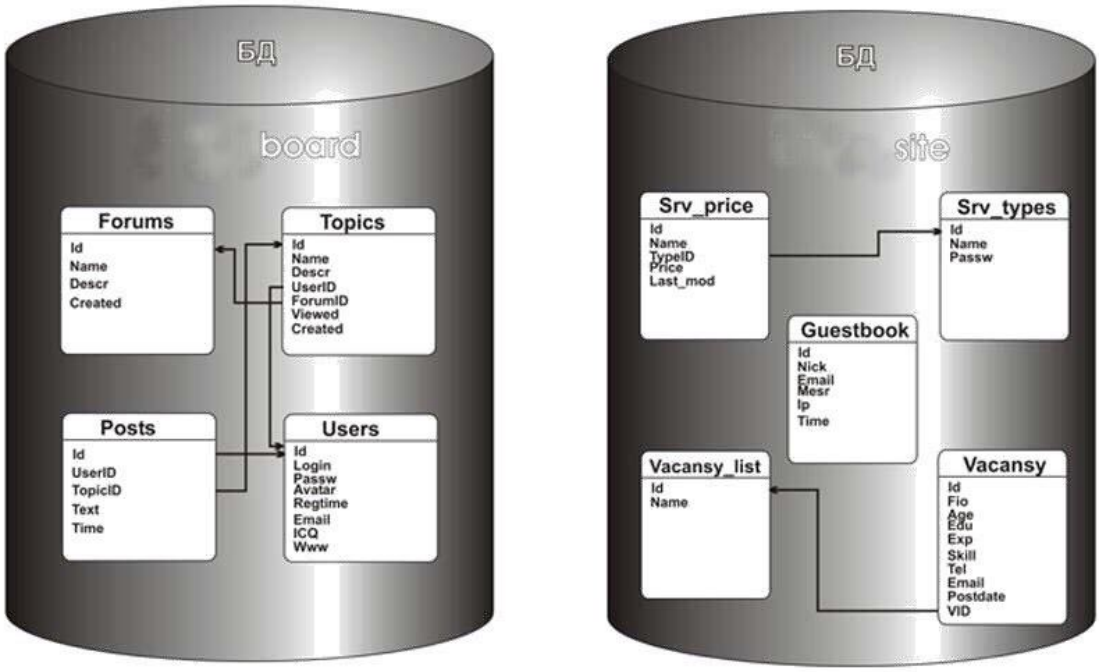

Рисунок 2.7 – Структура базы данных

3. Безопасность жизнедеятельности

Темой дипломного проекта было выбрано WEB–приложение, состоящее из двух частей, первая часть которого является этапом разработки технического задания, а ко второму этапу относится программная разработка. При разработке были соблюдены все правила техники безопасности, в связи с тем, что работа является сидячей, а именно разработка сайта, то основными пунктами были: освещение, вентиляция, а также работа с электрическими приборами и правила их эксплуатации.

3.1 Анализ условии труда

Данная система бронирования основана на базе предприятия ТОО «Аслан Курылыс». Помещение представляет собой комнату размерами 4х3х3. К списку оборудований относятся 2 ПК и 2 принтера. В помещении есть 2 окна по 1.5 $м^2$ и лампа мощностью 60 Вт. Данные компьютеры не создают большого шумового давления в пределах нормативов. В связи с тем, что вентиляция проведена не самым лучшим образом, я решил, что в данном разделе безопасности жизнедеятельности я буду рассчитывать вентиляцию.

Перед предстоящим расчетом нужно получить исходные данные исходя из того, какое мы используем помещение, количество персонала, а также оборудование, которое участвует непосредственно в разработке программного продукта, а именно сайта.

Город: Алматы;
Параметры помещения (Д х Ш х В), м: 4х3х3;
Данные по оборудованию: кол–во 4 шт.;
Мощность Роб, кВт/ч = 0,5;
КПД $\eta = 0,95$;
Данные по ист. света: мощность N ос. уст., $Вт/м^2 = 60$;
Вид ист. св.: люминесцентные лампы;
Число сотрудников: 2 мужчины;
Окна: кол–во 2;
Площадь 1 окна, $м^2 = 1.5$;
Расположение: СВ;
Вид: остекление в один–х метал. переплет, загрязнение умеренное;
Расчетное время суток, ч.: 12–13;
Температура в помещении, ^{0}С: летом 23, зимой 21;
Вид положения работы: сидячая работа.

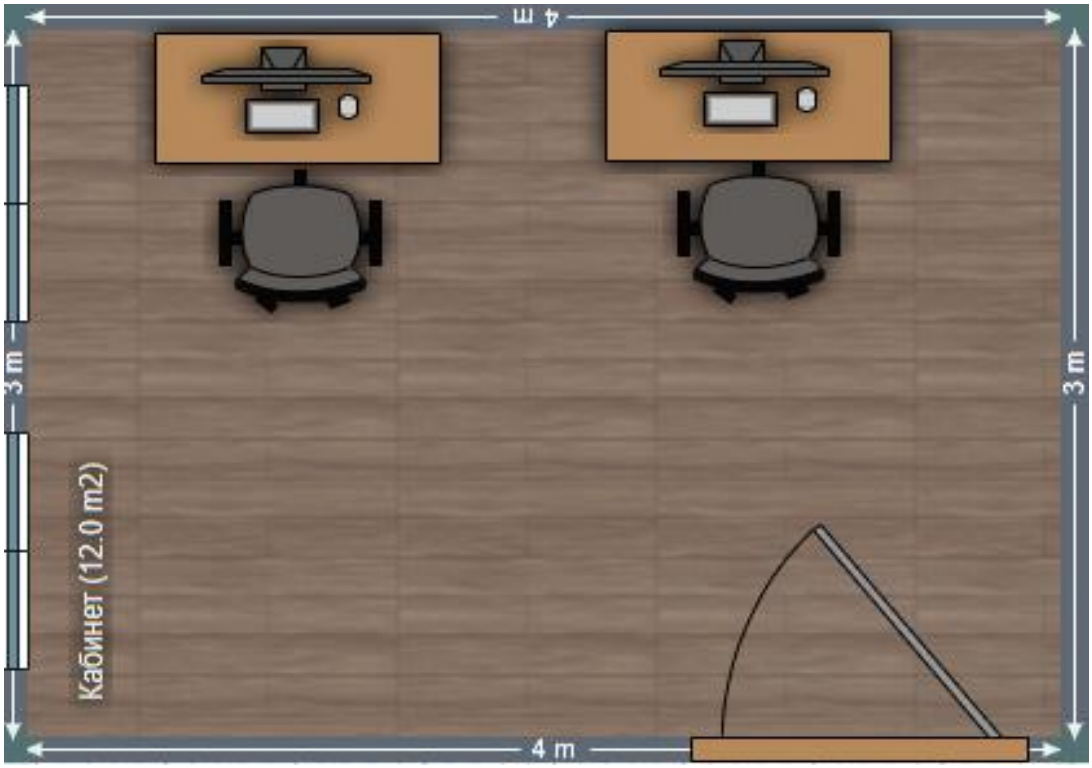

Рисунок 3.1 – Кабинет

В данной части дипломного проекты была рассмотрена тема вентиляции помещения. Вентиляция является основным параметром безопасности жизнедеятельности во время труда и является обязательным параметром для расчета. В данной части дипломного проекта был произведен расчет следующих пунктов:

– Расчет наружных тепловых нагрузок;

– Расчёт тепловых нагрузок внутри помещения;

– Расчет количество воздуха необходимого тому или иному помещению для подачи в данное помещение;

– По полученному расчёту выбрать нужный кондиционер и показать все его технические характеристики в таблице;

– После выбора кондиционера показать расположение его блоков внешнего и внутреннего.

3.2 Расчет тепловых нагрузок в помещении

Тепловые нагрузки непосредственно воздействуют на используемое нами помещение как внутренние тепловые нагрузки, так и внешние непосредственно климат, лучи солнца и общая температура погоды в городе в тот или иной сезон.

3.2.1 Наружные тепловые нагрузки.

Данные нагрузки представлены следующими составляющими:

– теплопоступления или тепло–потери в результате разности температур снаружи и внутри здания через стены, потолки, полы, окна и двери.

– разность температур снаружи здания и внутри него летом является положительной, в результате чего имеет место приток тепла снаружи во внутрь помещения; и наоборот – зимой эта разность отрицательна и направление потока тепла меняется;

– теплопоступления от солнечного излучения через застекленные площади; данная нагрузка проявляется в форме ощущаемого тепла;

– теплопоступления от инфильтрации.

В зависимости от времени года и времени суток наружные тепловые нагрузки могут быть положительными. Теплопоступления и теплопотери в результате разности температур определяются по формуле 4.1:

$$Q_{огр} = V_{пом} * X_о * (t_{Нрасч} – t_{Вр} \qquad (4.1)$$

где $V_{пом.}$ – объем помещения, $м^3$:

$V_{пом}= 4·3·3=36\ м^3$;

$X_о$ – удельная тепловая характеристика, $Вт/м^3\ ^0C$:

$X_о=0,42\ Вт/м^3\ ^0C$;

$t_{Нрасч}$ – наружная температура (параметр А). Для холодного периода – средняя температура самого холодного месяца в 13 часов, для теплого периода – средней температуре самого жаркого месяца в 13 часов.

$t_{Врасч}$ – внутренняя температура, выбирается с учетом комфортных условий или технологических требований, предъявляемых к производственным процессам.

Для теплого времени года:

$t_{Нрасч} = 29,4\ ^0C$

$t_{Врасч} = 26\ ^0C$

$Q_{огр.}= 36·0,42·3,4=51,41$ Вт

Для холодного времени года

$t_{Нрасч} = –9\ ^0C$

$t_{Врасч} = 19\ ^0C$

$Q_{огр.}= 36·0,42·|–28| = 423,36$ Вт

Избыточная теплота солнечного излучения в зависимости от типа стекла почти до 90% поглощается средой помещения, остальная часть отражается. Максимальная тепловая нагрузка достигается при максимальном уровне излучения, которое имеет прямую и рассеянную составляющие.

Интенсивность излучения зависит от ширины местности, времени года и времени суток.

Теплопоступление от солнечного излучения через остекление определяется по формуле 1.2:N

$$Q_p = (q^I F^I_o + q^{II} F^{II}_o)* \beta с. з., \qquad (1.2)$$

где q^I, q^{II} – тепловые потоки от прямой и рассеянной солнечной радиации, Вт/м^2 ;

F^I_o, F^{II}_o – площади светового проема, облучаемые и не облучаемые прямой солнечной радиацией, м^3;

βс. з. – коэффициент теплопропускания:

βс. з. = 0.15

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

$$Q_p = q^I F_o * \beta с. з. = (q_{вп} + q_{вр}) * K^C_1 * K_2 * \beta с. з. * n * S_o , Вт \qquad (1.3)$$

где $Q_{вп}$; $q_{вр}$ – тепловые потоки от прямой рассеянной радиации, Вт/м^2. Для широты в 440 СШ до полудня в 11–12 ч. при расположении З:

$Q_{вп}$ = 73 Вт/м^2 ; $q_{вр}$ = 77 Вт/м^2 ;

$F_o = nS_o = 2 \cdot 1.5 = 3$ м^2 – площадь светового проема (n – число окон; S_o – площадь 1 окна);

K_1 – коэффициент затемнения остекления переплетами (K^C_1 – для облученных проемов):

$K^C_1 = 0.72$;

K_2 – коэффициент загрязнения остекления:

$K_2 = 0.9$.

Тогда:

$Q_p = (73+77)*0{,}72*0{,}9*0{,}15*3 = 43{,}74$ Вт.

Для широты в 440 СШ до полудня в 11–12 ч. при расположении В:

$Q_{вп}$ = 214 Вт/м^2; $q_{вр}$ = 79 Вт/м^2 ;

$F_o = nS_o = 2 \cdot 1.5 = 3$ м^2 – площадь светового проема (n – число окон; S_o – площадь 1 окна);

Тогда:

$Q_p = (214+79)*0{,}72*0{,}9*0{,}15*3 = 85{,}44$ Вт.

Тогда общее теплопоступление солнечного излучения с обеих окон равно:

$Q_р$=43,74+85,44=129,18 Вт.

3.2.2 Внутренние тепловые нагрузки.

Внутренние нагрузки в жилых, офисных или относящихся к сфере обслуживания помещениях слагаются в основном из тепла:

– выделяемого людьми;

– выделяемого лампами и осветительными, электробытовыми приборами;

– выделяемого компьютерами, печатающими устройствами, фотокопировальными машинами пр.;

В производственных и технологических помещениях различного назначения дополнительными источниками тепловыделений могут быть: нагретое производственное оборудование, горячие материалы, в том числе жидкости и различного рода полуфабрикаты, продукты сгорания и химических реакций.

Теплопоступления от людей зависит от интенсивности выполняемой работы и параметров окружающего воздуха. Тепло, выделяемое человеком, складывается из ощутимого (явного), то есть передаваемого в воздух помещения путем конвекции и лучеиспусканий, и скрытого тепла, затрачиваемого на испарение влаги с поверхности кожи и из легких.

Летом при 24 ^{0}С один мужчина выделяет явного тепла 61 Вт, а общего – 102 Вт. Тогда выделение явного тепла в помещении составит:

$Q_л^я$ = 61*2 =122 Вт.

А выделение общего тепла:

$Q_л^о$ = 102*2=204Вт.

Зимой при 20 ^{0} С один мужчина выделяет явного тепла 82 Вт, а общего – 103 Вт. Тогда выделение явного тепла в помещении составит:

$Q_л^я$ = 82*2=164 Вт.

А выделение общего тепла:

$Q_л^о$=103*2=206 Вт.

Теплопоступление от осветительных приборов, оргтехники и оборудования рассчитывается следующим образом. Теплопоступление от ламп определяется по формуле (4.4):

$$Q_{осв} = \eta * N_{осв} * F_{пол} \quad (1.4)$$

где η – коэффициент перехода электрической энергии в тепловую (для люминесцентных ламп η=0.5–0.6);

$N_{осв}$ – установленная мощность ламп (N=60 Вт/м^2);

$F_{пол}$ – площадь пола:

$F_{пол}$=4·3=12

Тогда:

$Q_{осв}$=0,5·60·12= 360 Вт

Тепло, выделяемое Персональным компьютером, определяется по формуле:

$Q_{об}$=1,8*10^3*4*0,95= 6840 Вт.

Теплопритоки, возникающие за счет находящейся оргтехники, – это 30% мощности оборудования:

$Q_{орг}$ = 1,8*10^3*4*0,3= 2160 Вт.

3.3 Расчет необходимого воздуха для вентиляции.

На основании выполненных расчетов составим баланс теплопоступлений в помещении:

Лето: $Q_{изб}$ = 129,18 + 225,7 + 360 + 6840 + 2160 + 51,41 = 9766,29 Вт

Зима: $Q_{изб}$ = 129,18 + 303,4 + 360 + 6840 + 2160 + 423,36 =10215,94 Вт

Так как тепловой баланс для лета больше зимнего теплового баланса, то рассчитаем тепло–напряженность воздуха по формуле:

$$Q_н = \frac{10215{,}94 \cdot 860}{36} = 244{,}05 \text{ ккал/м3}$$

При $Q_н$ >20ккал/м3 , Δt = 8 °С.

Определение количества воздуха, необходимое для поступления в помещение:

$$L=\frac{10215{,}94\cdot 860}{0{,}24\cdot 8\cdot 1{,}206\cdot 10^4}=379{,}42\,\frac{\text{м3}}{\text{час}}$$

, где

C=0,24 ккал/(кг °C) – теплоемкость воздуха,

γ=1,206 кг/м^3 – удельная масса приточного воздуха.

Определение кратности воздухообмена:

$$N=\frac{379{,}42}{36}=10.53\ \text{час}^{-1}$$

По найденному значению количества воздуха подобрать соответствующую модель кондиционера:

Исходя из полученных данных, выберем кондиционер сплит–системы настенного типа.

Таблица 3.3 – Основные технические характеристики настенного кондиционера серии Panasonic CS–E18PKDW/CU–E18PKD

Эл. питание В/Гц	Произв. по холоду, кВт	Потр. эл мощн, кВт	Потребл ток, А	Произв. по теплу, кВт	Размер (внешн. блок) Мм	Расход воздуха, м3 /ч	Размер (внутр. блок) мм
240/1/50	5	1,5	7,2	5,8	L 1007 H 290 B 240	878	L 875 H 695 B 320

Во внешнем блоке находятся компрессор, конденсатор и вентилятор. Внешний блок можно установить на стене здания, на крыше или на чердаке, в подсобном помещении или на балконе, то есть в таком месте, где горячий конденсатор может продуваться атмосферным воздухом более низкой температуры.

Внутренний блок устанавливается непосредственно в кондиционируемом помещении и предназначен для охлаждения или нагревания воздуха, фильтрации его и создания необходимой подвижности воздуха в помещении. Внутренние блоки поддерживают заданную температуру, обеспечивают равномерное распределение воздуха в помещении и работают практически бесшумно (уровень шума 35–38 дБ).

Управление работой настенного кондиционера производится с дистанционного пульта, который позволяет задать режим работы кондиционера: обогрев, охлаждение, осушку, вентиляцию, ночной режим; задать требуемую температуру, которую должен поддерживать автоматически; выбрать режим работы вентилятора: настроить таймер, который включит или выключит кондиционер в заданное время; автоматически регулировать положение направляющих шторок и изменить таким образом направление воздушного потока.

Так как количества воздуха, необходимое для поступления в помещение равно 379,42 м3/час, то будет использован один кондиционер Panasonic CS–E18PKDW/CU–E18PKD, который выдает необходимый нам расход воздуха.

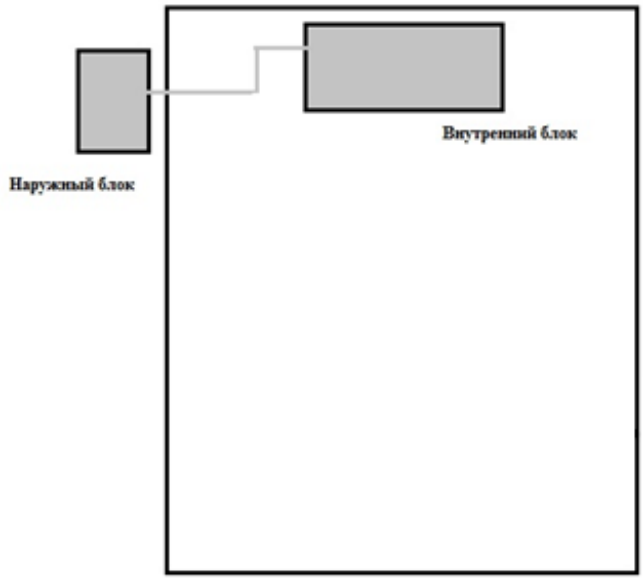

Рисунок 3.3 – Схема расположения кондиционеров в производственном помещении

В этой части дипломного проекта мы расчитали возмонжные тепловые нагрузки как и внутренние так и внешние для выбранного нами помещения. Расчитали необходимое количество подаваемого воздуха, после того как узнали количество воздуха выбрали подходящий кондиционер по его характеристикам и указали место расположения его блоков внешнего вентиляционного и внутреннего в помещении.

Тем самым можно сделать вывод, что безопасность жизнедеятельности учавствует во всех аспектах жизнедеятельности человека в любых профессиях и быте. На примере разработки данного проекта были соблюдены все нормы вентиляции, выбран походящий кондционер, а так же соблюдены все пункты с овещением. Так же выявлено, что тепловые нагрузки зависят от размерности помещения и от количества людей находящимся в нем, а так же от таких внешних тепловых нагрузок как солнечные лучи и количество окон в помещении через которые они проходят. Большое влияние оказали погодные условия в городе проведения разработки.

4. Технико – экономическое обоснование

4.1 Описание работы и обоснование необходимости

Темой дипломного проекта является «разработка информационных систем для ТОО «Аслан Курылыс»

В данной дипломной работе описан проект создания корпоративного функционального WEB–сайта для ТОО «Аслан Курылыс» для работы с существующими и потенциальными клиентами компании.

Целью данного раздела является расчет затрат на создание программного продукта и расчет себестоимости прикладной программы.

Чтобы определить себестоимость программы, необходимо также принять:

- трудоемкость разработки программного продукта;
- материальные затраты;
- затраты на оплату труда;
- социальный налог;
- амортизация основных фондов.

4.2 Расчет трудоемкости разработки программного продукта

Таблица 4.2 – Распределение работ по этапам и видам и оценка их трудоемкости

Этап разработки	Вид работы	Трудоемкость разработки, чел.×ч.
1	Составление задачи	20
2	Создание алгоритмов и блок схемы	10
3	Создание клиентской части программного проекта	50
4	Создание серверной части проекта	100
5	Тестирование	20
6	Документация	40
ИТОГО трудоемкость выполнения программного продукта		240

Чтобы определить сколько это будет в днях, нам нужно разделить итоговое число на 8(рабочий день): 240 / 8 = 30

4.3 Расчет затрат на разработку программного продукта

Для вычисления затрат на реализацию ПП необходимо найти через составления сметы, которая учитывает следующие статьи:

- материальные затраты;
- затраты на оплату труда;
- социальный налог;
- амортизация основных фондов.

Таблица 4.3.1 – Затраты на материальные ресурсы

Наименование материального ресурса	Единица измерения	Количество	Цена за единицу, тг	Сумма, тг
Ноутбук ASUS ROG V502	шт.	1	150000	150000
ПО Аслан Курылыс	шт.	1	50000	50000
Картридж	шт.	1	1500	1500
Бумага А4	шт.	1	2000	2000
ИТОГО				203500

Общая сумма затрат на материальные ресурсы ($З_М$) определяется по формуле:

$$З_М = \sum_{i=1}^{n} Р_i \times Ц_i , \qquad (3.1)$$

где $Р_i$– расход i–го вида материального ресурса, натуральные единицы;

$Ц_i$– цена за единицу i–го вида материального ресурса, тг;

i – вид материального ресурса;

n – количество видов материальных ресурсов.

$$З_{ноутбук} = 1 \times 150000 = 150000 \text{ тг}$$

$$З_{ПО} = 1 \times 50000 = 50000 \text{ тг}$$

$$З_{бумага} = 1 \times 2000 = 2000 \text{ тг}$$

$$З_{картридж} = 1 \times 1500 = 1500 \text{ тг}$$

$$З_{общие} = 150000 + 50000 + 2000 + 1500 = 203500 \text{тг}$$

Если для разработки ПП используется электрооборудование, то необходимо рассчитать затраты на электроэнергию по форме, приведенной в таблице 4.3.2

Общая сумма затрат на электроэнергию ($З_Э$) рассчитывается по формуле:

$$З_Э = \sum_{i=1}^{n} M_i \times K_i \times T_i \times Ц, \quad (3.2)$$

где M_i– паспортная мощность i–го электрооборудования, кВт;

K_i– коэффицент использования мощности i–го электрооборудования (применяется K_i=0,9);

T_i– время работы i–го оборудования за весь период разработки ПП ч;

Ц– цена электроэнергии, тг/кВт×ч;

i – вид электрооборудования;

n – количество электрооборудования.

Затраты на электроэнергию находятся исходя из продолжительности периода разработки ПО, количества кВт/ч, затраченных на проектирование ПО и тарифа за 1 кВт/ч. Тариф по городу Алматы для юридических лиц в 2019 году составляет 18,32 тенге за 1 кВт/ч с учетом НДС (согласно данным представленным на официальном сайте ТОО «АлматыЭнергоСбыт»).

$З_э$=0,9·0,9·240·18,32=3561,40 тг

$З_э$=0,3·0,7·240·18,32= 923.32 тг

Таблица 4.3.2 – Затраты на электроэнергию

Наименование оборудования	Паспортная мощность, кВт	Коэффициент использования мощности	Время работы оборудования для разработки ПП, ч	Цена электро энергии, тг/кВт·ч	Сумма, тг
Ноутбук ASUS ROG V502	0,9	0,9	240	18,32	3561,40
Освещение	0,3	0,7	240	18,32	923.32
ИТОГО					4484.73

4.3.1 Расчет расходов на оплату труда

В разработке программного продукта заняты 2 сотрудника: инженер–разработчик и программист. Средняя заработная плата инженер–разработчика

в 2019 году составляет 200 000 тг, а программиста 180 000 тг (для города Алматы).

Рабочие часы сотрудника за месяц определяются по формуле:

$$Ч_м = N_м \cdot Ч_{рд}, \quad (3.3)$$

где $Ч_м$– рабочие часы сотрудника за месяц;
$N_м$–количество рабочих дней за месяц;
$Ч_{рд}$–количество рабочих часов в день.

$Ч_м = 21 \times 8 = 168$ ч.

Часовая ставка работника может быть рассчитана по формуле:

$$ЧС_i = \frac{ЗП_i}{ФРВ_i} \quad (3.4)$$

Инженер–разработчик:

$$ЧС_i = \frac{200000}{168} = 1190,48 \ тг$$

Программист:

$$ЧС_i = \frac{180000}{168} = 892,86 \ тг,$$

где $ЗП_i$– месячная заработная плата i–го работника, тг;
$ФРВ_i$– месячный фонд рабочего времени i–го работника, час.

Для определения трудоемкости разработки ПП используются данные из таблицы 3.1.

Трудоемкость разработки ПП инженер–разработчика равна 140 чел.×ч(составление задачи, разработка алгоритмов и блок схемы, реализация клиентской части проекта, тестирование, документация).

$T_2 = 20+10+50+20+40 = 140$ чел.×ч.

Трудоемкость разработки ПП программиста равна 130 чел.×ч.(разработка блок–схемы алгоритма, разработка администраторской части проекта, тестирование программы).

$T_2 = 10 + 100 + 20 = 130$ чел.×ч.

Общая сумма затрат на оплату труда ($З_{ТР}$) определяется по формуле:

$$З_{ТР} = \sum_{i=1}^{n} ЧС_i \times T_i \,, \qquad (3.6)$$

где $ЧС_i$– часовая ставка i–го работника, тг;

T_i– трудоемкость разработки ПП, чел.×ч;

i – категория работника;

n – количество работников, занятых разработкой ПП.

Инженер–разработчик:

$З_{тр}$ = 1190,48×140 = 166667.19 тг.

Программист:

$З_{тр}$ = 892,86×130 = 116071.78 тг.

Общая сумма:

$З_{тр}$ = 166667.19 + 116071.78 = 282739,97 тг.

Таблица 4.3.3 – Затраты на оплату труда

Квалификация	Трудоемкость разработки ПП, чел.×ч	Часовая ставка, тг/ч	Сумма, тг
Инженер–разработчик	140	1190,48	166667.19
Программист	130	892,86	116071.78
ИТОГО			282739,97

Дополнительная заработная плата:

$З_{доп} = З_{тр}$×10%

$З_{доп}$ = 282739,97×0,1 = 28273.997 тг.

Фонд заработной платы:

$Ф_{зп} = З_{тр.о} + З_{доп}$

$Ф_{зп}$ = 282739.97 + 28273.99 = 311013.95 тг.

Расчет социального налога:

$Н_с = (Ф_{зп} – ОПВ)×11\%$,
где ОПВ – обязательные пенсионные взносы–10% от $Ф_{зп}$.

$Н_с = (311013.95 – (311013.95 ×0,1))×0,11 = 30790.38$ тг.

Расчет амортизационных основных фондов:
Общая сумма амортизационных отчислений определяется по формуле:

$$З_{АМ} = \sum_{i=1}^{n} \frac{Ф_i \times Н_{Ai} \times Т_{НИРi}}{100 \times Т_{Эфi}}, \quad (3.5)$$

где $Ф_i$– стоимость i–го ОФ, тг;
$Н_{Ai}$– годовая норма амортизации i–го ОФ, %;
$Т_{НИРi}$– время работы i–го ОФ за весь период разработки ПП, ч;
$Т_{Эфi}$– эффективный фонд времени работы i–го ОФ за год, ч/год (по информации egov.kz за 2019 год принимается $Т_{Эфi}$=1968);
i – вид ОФ;
n – количество ОФ.
Расчет годовой нормы амортизации ОФ:

$$Н_{Ai} = \frac{100}{Т_{Ni}}, \quad (3.6)$$

где $Т_{Ni}$– возможный срок использования i–го ОФ, год.
Для определения времени работы ПО для разработки ПП используются данные из таблицы 3.1.
Время работы ПО Аслан Курылыс для разработки ПП составляет 170 часов (реализация клиентской части проекта, реализация администраторской части проекта, тестирование программы):

$Т_i = 100 + 50 + 20 = 170$ ч.

Оборудование:

$З_{АМ} = \frac{150000*25*240}{100*1968} = 4573.17$

Программное обеспечение Аслан Курылыс:

$З_{АМ} = \frac{50000*25*170}{100*1968} = 1079.77$

Таблица 4.3.4– Амортизация основных фондов (ОФ)

Наименование оборудования и ПО	Стоимость оборудования и ПО, тг	Годовая норма амортизации, %	Эффективный фонд времени работы оборудования и ПО, ч/год	Время работы оборудования и ПО для разработки ПП, ч	Сумма, тг
Ноутбук ASUS ROG V502	150000	25	1968	240	4573,17
ПО Аслан Курылыс	50000	25	1968	170	1079,77
ИТОГО					5652,94

Таблица 4.3.5 – Смета затрат на разработку ПП

Статьи затрат	Сумма, тг	%
1. Затраты на материальные ресурсы	203500	38
2. Затраты на электроэнергию	4484,73	1
3. Затраты на оплату труда	311013,95	54
4. Отчисления на социальные нужды	30790,38	6
5. Амортизация основных фондов	5652,94	1
ИТОГО	555441,98	100

Рисунок 4.3 Смета затрат

4.3.2 Определение возможной (договорной) цены программного продукта

Договорная цена ($Ц_Д$) для прикладных ПП рассчитывается по формуле:

$$Ц_Д = З_{НИР} \cdot \left(1 + \frac{Р}{100}\right), \quad (3.7)$$

где $З_{НИР}$– затраты на разработку ПП (из таблицы 3.6), тг;

Р– средний уровень рентабельности ПП – 25%.

$Ц_Д$ = 555441,98· (1 + 0,25) =555441,98+ 138860,46=694302,44 тг

Далее определяется цена реализации с учетом налога на добавленную стоимость (НДС), ставка НДС устанавливается законодательно Налоговым Кодексом РК. На 2019 год ставка НДС установлена в размере 12%.

Цена реализации с учетом НДС рассчитывается по формуле:

$Ц_Р=Ц_Д·(1 + НДС)$

$Ц_Р$ = 694302,44 · (1 + 0,12) = 694302,44 + 83316,29= 777618,73 тг

Вывод:

Таким образом, цена реализации с учетом НДС равна 777618,73 тг, себестоимость – 555441,98 тг и прибыль – 138860,46 тг.

Основную часть затрат составляют затраты на оплату труда(54%).

Заключение

Довольно сложно точно определить, что подразумевается под термином "Webдизайн". В лучшем случае мы видим, что Web-дизайн — это мульти-дисциплинарная область деятельности, которая состоит из пяти основных компонентов: содержимого, зрительных образов, технологии, доставки и назначения. Взгляды на то, как именно эти компоненты должны переплетаться, различаются от человека к человеку, равно как и от проекта к проекту. Соглашение о тонком равновесии между формой и функцией, пользователем и дизайнером, содержимым и задачей, традицией и инновацией — вот высокая цель Webдизайнера. Хороший дизайнер знает, какие чаши не должны слишком отклоняться друг от друга в ту или иную сторону, и стремится избегать абсолютизмов о "правильном" Webдизайне. Тем не менее, не все в сфере Web-дизайна так абстрактно — можно найти много конкретики. Совершенное владение технической средой и знание различных деталей и соглашений — непременные условия для честолюбивых Web-профессионалов.

В данном дипломном проекте был построен корпоративный Web-сайт для компании «Аслан Курылыс». Построение современного Web-сайта может быть очень не простой задачей, поэтому была взята на вооружение определённая методология или модель процесса, которая будет направлять процесс создания сайта, сведет к минимуму риск, позволит решать сложные вопросы и значительно улучшит конечный результат. Модели процесса из области разработки программного обеспечения, например, модифицированный водопад, вполне подойдут для использования при реализации большинства Web-проектов. Однако в некоторых случаях, как правило, из-за недостаточного опыта в организации управления проектами или отсутствия четких целей, возможно применение модели процесса, основанной на прототипах или так называемой совместной разработке приложения. Сложно планировать то, что неизвестно, и если процесс нельзя урегулировать, то, вероятно, лучше всего попытаться сделать что-то быстро, потерпеть неудачу и научиться на этой своей ошибке.

После внедрения сайта в предприятие, значительно увеличилось конверсия в компании, с каждым днем число покупателей было все больше и больше, так же, что бы полностью реализовать возможно нашего сайта мы запустили таргет рекламу в истаграме, суть рекламы была в том что, покупатели нажимая на ссылку в рекламе переходили на наш сайт и более подробно узнавали о наших услугах и оставляли заявки, выхлоп от такой рекламы был огромным, что значительно повлиял на наши сделки, и бюджет компании, исходя из этого можно с уверенностью сказать что веб сайт для предприятия обязателен, так как мы живем в том веке когда вся реклама строится через интернет., При реализации данного проекта были изучены различные виды Web-сайтов и проведен их сравнительный анализ.

При сравнении торгового предприятия без сайта с предприятием имеющем сайт было выявлено, что Web-сайт позволяет значительно увеличить продажи компании, а также делает ее более узнаваемой и привлекательной для покупателей и бизнес партнеров.

Список литературы

1 Первый сайт в истории интернета / URL: http://preal.ru/webhist/pervyy-sayt-v-istorii-interneta/ (дата обращения: 08.02.2019)

2 Википедия – свободная энциклопедия / URL: http://ru.wikipedia.org/wiki/ (дата обращения: 08.02.2019)

3 Виды, типы, разновидности сайтов / URL:http://fearit.net/fearit/index.php?option=com_content&view=article&id=142&Itemid=156. (дата обращения: 10.02.2019)

4 Классификация сайтов в современном web-е / URL: http://www.blog.astramg.ru/articles/1627/ (дата обращения: 11.02.2019)

5 Понятие веб сайта. Классификация сатов / URL: http://ikt-history.ucoz.net/publ/17-1-0-13. (дата обращения: 11.02.2019)

6 Классификация веб-сайтов, типы и виы интерне сайтов / URL: http://www.yanajy.com/sdelay-sam/vidy-sajtov.html. (дата обращения: 11.02.2019)

7 Краткая история HTTML и CSS / URL: http://www.greenlight5.com/index61.htm. (дата обращения: 11.02.2019)

8 Средства создание сайтов / URL: http://www.in-internet.narod.ru/teor/sreda.html. (дата обращения: 12.02.2019)

9 XHTML / URL: http://htmlweb.ru/html/xhtml.php. (дата обращения: 12.02.2019)

10 Введение. DOM в примерах / URL: http://javascript.ru/tutorial/dom/intro. (дата обращения: 12.02.2019)

11 Введение в JavaScript / URL: http://www.myfirstsite.ru/articles/javascript. (дата обращения: 12.02.2019)

12 TIOBE Programming Community Index for March 2013 / URL: http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html. (дата обращения: 14.02.2019)

13 Технология Flash в создании сайтов / URL: http://www.rusarticles.com/veb-dizajn-statya/texnologiya-flashvsozdanii-sajtov-3408023.html. (дата обращения: 14.02.2019)

14 PHP против ASP.Net – что лучше? / URL: http://nevlabs.ru/articles/web/php-vs-aspnet/ (дата обращения: 14.02.2019)

15 Методы создания сайтов / URL: http://www.web-ms.ru/publ/metody_sozdanija_sajtov/1-1-0-98. (дата обращения: 15.02.2019)

16 Что такое CMC? / URL: http://www.twl.ru/article_01. (дата обращения: 15.02.2019)

17 Выбор системы управления сайтами/ URL: http://www.cmsmagazine.ru/library/items/cms/choice_cms/ (дата обращения: 15.02.2019)

18 Общая классификация CMS / URL: http://www.solus.ru/articles_9.html. (дата обращения: 18.02.2019)

19 Что такое SEO (CEO)? / URL: http://asbseo.ru/optimizaciya-i-prodvizhenie-bloga/chto-takoe-seo-vidyoptimizaciisajtov.html. (дата обращения: 18.02.2019)

20 Внутренняя оптимизация сайта / URL: http://seokleo.ru/inside_optimization/ (дата обращения: 18.02.2019)

Приложение А
(обязательное)

Техническое задание

– Разработка сайта;

– Разработка креативной платформы бренда, коммуникационной стратегии, нейминга объекта, айдентики ;

– Верстка и наполнение сайта информационными и графическими материалами. Разработать сайт на основании утвержденной дизайн–концепции, адаптивной верстки, обеспечивающей корректную работу сайта на всех компьютерных и мобильных устройствах;

– Корректное отображение браузерами Internet Explorer, Opera, Mozilla Firefox;

– Обязательная визуальная поддержка действий пользователя – «интерактив» (визуальное отображение активных, пассивных ссылок);

– Мета–теги и контент сайта на этапе изготовления сайта настроены для поисковых систем, чтобы обеспечить продвижение сайта по ключевым словам в поисковых системах Yandex, Google и др;

– Включенная поддержка javascript, Flash и cookies;

– В качестве базы данных для сайта использовать SQL.

Приложение Б
(обязательное)

Листинг программы

```
<!DOCTYPE html>
<html lang="{{ config('app.locale') }}">
<head>

    <title>Аслан Курылыс</title>
    <!— Basic Page Needs
    ================================================== —>
    <meta charset="utf–8">
    <meta name="viewport" content="width=device–width, initial–scale=1">
    <meta name="description"
        content="строительство трубопроводов наружных сетей водопровода и канализации услуги спецтехники благоустройство территории">
    <meta name="keywords"
        content="Аслан курылыс строительство реконструкция капитальный ремонт наружные сети водопровод канализация отопление">
    <meta name="csrf–token" content="{{ csrf_token() }}">

    <!— Assets —–>
    <link rel="stylesheet" href="{{asset('css/app.css')}}">

    <!— Favicons
    ================================================== —>
    <link rel="shortcut icon" href="{{asset('img/icons/logo.ico')}}" type="image/x–icon">
    <link rel="apple–touch–icon" href="{{asset('img/apple–touch–icon.png')}}">
    <link rel="apple–touch–icon" sizes="72x72" href="{{asset('img/apple–touch–icon–72x72.png')}}">
    <link rel="apple–touch–icon" sizes="114x114" href="{{asset('img/apple–touch–icon–114x114.png')}}">

    <!— Bootstrap —>
    <link href="https://maxcdn.bootstrapcdn.com/font–awesome/4.7.0/css/font–awesome.min.css" rel="stylesheet"
        integrity="sha384–wvfXpqpZZVQGK6TAh5PVlGOfQNHSoD2xbE+QkPxCAFlNEevoEH3Sl0sibVcOQVnN" crossorigin="anonymous">
    <!— Nivo Lightbox
    ================================================== —>
    <link rel="stylesheet" href="{{asset('css/theme/nivo–lightbox.css')}}">
    <link rel="stylesheet" href="{{asset('css/theme/nivo_lightbox_themes/default/default.css')}}">

    <!— Slider
    ================================================== —>
    <link href="{{asset('css/theme/owl.carousel.css')}}" rel="stylesheet" media="screen">
    <link href="{{asset('css/theme/owl.theme.css')}}" rel="stylesheet" media="screen">
```

```
        <!— Stylesheet
        ================================================== —>
        {{—    <link rel="stylesheet" type="text/css" href="{{asset('css/theme/style.css')}}">—
}}
        <link rel="stylesheet" type="text/css" href="{{asset('css/theme/responsive.css')}}">

        <!— Google Fonts
        ================================================== —>
        <script                                                    type="text/javascript"
src="{{asset('js/theme/modernizr.custom.js')}}"></script>
    </head>
    <body>

    <!— Main Navigation
        ================================================== —>
    <nav id="tf–menu" class="navbar navbar–default navbar–fixed–top">
        <div class="container">
            <!— Brand and toggle get grouped for better mobile display —>
            <div class="navbar–header">
                <button type="button" class="navbar–toggle collapsed" data–toggle="collapse"
                        data–target="#bs–example–navbar–collapse–1">
                    <span class="sr–only">Toggle navigation</span>
                    <span class="icon–bar"></span>
                    <span class="icon–bar"></span>
                    <span class="icon–bar"></span>
                </button>
                <a class="navbar–brand" href="#">
                    <img src="{{asset('img/logo.jpg')}}" alt="Aquatech Logo">
                </a>
            </div>

            <!— Collect the nav links, forms, and other content for toggling —>
            <div class="collapse navbar–collapse" id="bs–example–navbar–collapse–1">
                <ul class="nav navbar–nav navbar–right">
                    <li><a href="#tf–home" class="scroll">Компания</a></li>
                    {{—<li><a href="#tf–blog" class="scroll">Новости</a></li>—}}
                    <li><a href="#tf–services" class="scroll">Услуги</a></li>
                    <li><a href="#tf–works" class="scroll">Объекты</a></li>
                    {{—<li><a href="#tf–works" class="scroll">Оборудование</a></li>—}}
                    <li><a href="#tf–contact" class="scroll">Контакты</a></li>
                </ul>
            </div><!— /.navbar–collapse —>

            <div class="contact–us"> <!— Contact Us form link —>
                <a class="scroll" href="#contact–form">
                    <i class="fa fa–2x fa–envelope–o text–white" aria–hidden="true">

                    </i></a>
            </div> <!— Contact Us Form link End —>
```

```
    </div><!— /.container–fluid —>
  </nav>

  <!— Home Section
  ========================================================== —>
  <div id="tf–home">
    <div class="overlay"> <!— Overlay Color —>
      <div class="container"> <!— container —>
        <div class="content–heading text–center"> <!— Input Your Home Content Here –
–>
          <h1>Проектирование / Строительство / Поставка / Монтаж</h1>
          <p class="lead">Только проверенные решения</p>
          <a href="#tf–works" class="scroll goto–btn text–uppercase">Наши
объекты</a>
          <!— Link to your portfolio section —>
        </div><!— End Input Your Home Content Here —>
      </div> <!— end container —>
    </div><!— End Overlay Color —>
  </div>

  <!— Intro Section
  ========================================================== —>
  <div id="tf–intro">
    <div class="container"> <!— container —>
      <div class="row"> <!— row —>

        <div class="col–md–10 col–md–offset–2">
          <p class="text–promo">Инженерно–строительное предприятие "Аслан
Курылыс" осуществляет свою деятельность с
          2005 года и занимается проектированием, строительством,
реконструкцией, капитальным и ремонтом
          жилых и общественных зданий, коммерческих и промышленных
сооружений.
          Предприятие также оказывает услуги по строительству и ремонту
наружных и внутренних инженерных
          сетей водопровода, канализации,тепловых сетей и сетей
электроснабжения.</p>
        </div>

      </div><!— end row —>
    </div><!— end container —>
  </div>

  <!— Service Section
  ========================================================== —>
  <div id="tf–services">
    <div class="container"> <!— container —>

      <div class="section–header">
```

```
        <h2>Что мы делаем<span class="highlight"><strong> Отлично</strong></span></h2>
                <div class="fancy"><span><img class="logo"
src="{{asset('img/icons/logo.png')}}" alt="..."></span></div>
            </div>

            <div class="row"> <!— row —>

              <div class="col–md–6 text–right">  <!— Left Content Col 6 —>
                <div class="media service"> <!— Service #1 —>
                  <div class="media–body">
                    <h4 class="media–heading">Строительство наружных
трубопроводов</h4>
                    <p class="">В сферу деятельности нашей компании входят
проектирование,
                      строительство, реконструкция трубопроводов. Имеем большой опыт
проведения подобных
                      работ. Предприятие располагает собственной производстенной
базой, а также оборудованием
                      для выполнения сварочных и монтажных работ на трубопроводах из
стали, полиэтилена,
                      чугунных и ПВХ труб</p>
                  </div>
                  <div class="media–right media–middle">
                    <div class="png–container">
                      <img class="png" src="{{asset('img/icons/pipes.png')}}"
                        alt="">
                    </div>
                  </div>

                </div><!— End Service #1 —>

                <div class="media service"> <!— Service #2 —>
                  <div class="media–body">
                    <h4 class="media–heading">Внутренние трубопроводы</h4>
                    <p>Выполняем работы по монтажу внутренних систем водопровода,
канализации и отопления
                      различных диаметров из различных материалов</p>
                  </div>
                  <div class="media–right media–middle">
                    <div class="png–container">
                      <img class="png" src="{{asset('img/icons/valve.png')}}"
                        alt="">
                    </div>
                  </div>
                </div><!— End Service #2 —>

              </div> <!— End Left Content Col 6 —>

              <div class="col–md–6"> <!— Right Content Col 6 —>
```

```
<div class="media service"> <!— Service #3 —>
                <div class="media–left media–middle">
                  <i class="fa fa–home"></i>
                </div>
                <div class="media–body">
                  <h4 class="media–heading">Общестроительные работы</h4>
                  <p> Выполняем устройство фундамента, устройство несущих стен, перегородок и
                    перекрытий, а также все виды так называемых "черновых работ". У на работают
                    профессионалы
                    в данной области, обладающие соответствующим опытом, способные произвести
                    общестроительные работы с учетом всех требований, а также с учетом особенностей
                    конкретного строительного объекта или участка.</p>
                </div>
              </div><!— end Service #3 —>

              <div class="media service"> <!— Service #4 —>
                <div class="media–left media–middle">
                  <div class="png–container">
                    <img class="png" src="{{asset('img/icons/excavator.png')}}"
                      alt="">
                  </div>
                </div>
                <div class="media–body">
                  <h4 class="media–heading">Аренда спецтехники</h4>
                  <p>В распоряжении пердприятия имеются экскаваторы–погрузчики производства Caterpillar.
                    Опытные операторы</p>
                </div>
              </div> <!— end Service #4 —>

            </div><!— end Right Content Col 6 —>

          </div><!— end row —>

        </div><!— end container —>
      </div>

      <!— Works Section
      ========================================== —>
      <div id="tf–works">
        <div class="container">
          <div class="section–header">
            <h2>Наши объекты</h2>
            <div class="fancy"><span><img class="logo"
src="{{asset('img/icons/logo.png')}}" alt="..."></span></div>
```

```
        </div>

            <div class="text–center">
              <ul class="list–inline cat"> <!— Portfolio Filter Categories —>
                <li><a href="#" data–filter="*" class="active">Все</a></li>
                <li><a href="#" data–filter=".pipeline">Трубопроводы</a></li>
                <li><a href="#" data–filter=".pump">Насосные станции</a></li>
                <li><a href="#" data–filter=".sewage">Очистные сооружения</a></li>
                <li><a href="#" data–filter=".fasades">Фасады</a></li>
                <li><a href="#" data–filter=".roof">Кровли</a></li>
                <li><a href="#" data–filter=".construction">Общестроительные
работы</a></li>
              </ul><!— End Portfolio Filter Categories —>
            </div>

          </div><!— End Container —>

          <div class="container–fluid"> <!— fluid container —>
            <div id="itemsWork" class="row text–center"> <!— Portfolio Wrapper Row —>

              <!— Dymanic generated content —>

              @foreach ($data as $item)

                <div class="col–xs–12 col–sm–6 col–md–3 col–lg–2 nopadding {{$item–
>category}}">
                  <!— Works #1 col 3 —>
                  <div class="box">
                    <div class="hover–bg">
                      <div class="hover–text off">
                        <a title="{{$item–>title}}"
                          href="{{asset('storage/'.$item–>url_big_image)}}"
                          data–lightbox–gallery="gallery1"
                          data–lightbox–hidpi="{{asset('storage/'.$item–>url_big_image)}}">
                           <i class="fa fa–expand"></i>
                        </a>
                      </div>
                      <img src="{{asset('storage/'.$item–>url_image)}}" class="img–
responsive"
                          alt="Image">
                      <!— Portfolio Image —>
                    </div>
                  </div>
                </div><!— end —>

              @endforeach

            </div> <!— End Row —>
```

```
        </div> <!— End Container–Fluid —>
        </div>

        <!—  Blog Section
        ================================================== —>
        {{—<div id="tf–blog">—}}
        {{—<div class="container"> <!— container —>—}}
        {{—<div class="section–header">—}}
        {{—<h2>Latest from the <span class="highlight"><strong>Blog</strong></span></h2>–
–}}
        {{—<h5><em>We design and build functional and beautiful websites</em></h5>—}}
        {{—<div class="fancy"><span><img class="logo" src="img/icons/logo.ico"
alt="..."></span></div>—}}
        {{—</div>—}}
        {{—</div>—}}

        {{—<div id="blog–post" class="gray–bg"> <!— fullwidth gray background —>—}}
        {{—<div class="container"><!— container —>—}}

        {{—<div class="row"> <!— row —>—}}
        {{—<div class="col–md–6"> <!— Left content col 6 —>—}}

        {{—<div class="post–wrap"> <!— Post Wrapper —>—}}
        {{—<div class="media post"> <!— post wrap —>—}}
        {{—<div class="media–left">—}}
        {{—<a href="#"> <!— link to your post single page —>—}}
        {{—<img class="media–object" src="http://placehold.it/120x150" alt="...">—}}
        {{—<!— Your Post Image —>—}}
        {{—</a>—}}
        {{—</div>—}}
        {{—<div class="media–body">—}}
        {{—<p class="small">January 14, 2015</p>—}}
        {{—<a href="#">—}}
        {{—<h5 class="media–heading"><strong>Vel donec et scelerisque vestibulum.—}}
        {{—Condimentum aliquam, mollit magna velit nec</strong></h5>—}}
        {{—</a>—}}
        {{—<p>Tempor vestibulum turpis id ligula mi mattis. Eget arcu vitae mauris amet odio.—
}}
        {{—Diam nibh diam, quam elit, libero nostra ut. Pellentesque vehicula. Eget sed,—}}
        {{—dapibus </p>—}}
        {{—</div>—}}
        {{—</div><!— end post wrap —>—}}

        {{—<div class="post–meta"> <!— Meta details —>—}}
        {{—<ul class="list–inline metas pull–left"> <!— post metas —>—}}
        {{—<li><a href="#">by Rudhi Design</a></li> <!— meta author —>—}}
        {{—<li><a href="#">20 Comments</a></li> <!— meta comments —>—}}
        {{—<li><a href="#">Read More</a></li> <!— read more link —>—}}
        {{—</ul>—}}
```

```
{{—<ul class="list–inline meta–detail pull–right"> <!— user meta interaction —>—}}
{{—<li><a href="#"><i class="fa fa–heart"></i></a> 50</li> <!— like button —>—}}
{{—<li><i class="fa fa–eye"></i> 110</li> <!— no. of views —>—}}
{{—</ul>—}}
{{—</div><!— end Meta details —>—}}
{{—</div><!— end Post Wrapper —>—}}

{{—<div class="post–wrap"> <!— Post Wrapper —>—}}
{{—<div class="media post"> <!— post wrap —>—}}
{{—<div class="media–left">—}}
{{—<a href="#"> <!— link to your post single page —>—}}
{{—<img class="media–object" src="http://placehold.it/120x150" alt="...">—}}
{{—<!— Your Post Image —>—}}
{{—</a>—}}
{{—</div>—}}
{{—<div class="media–body">—}}
{{—<p class="small">January 14, 2015</p>—}}
{{—<a href="#">—}}
{{—<h5 class="media–heading"><strong>Vel donec et scelerisque vestibulum.—}}
{{—Condimentum aliquam, mollit magna velit nec</strong></h5>—}}
{{—</a>—}}
{{—<p>Tempor vestibulum turpis id ligula mi mattis. Eget arcu vitae mauris amet odio.—
}}
{{—Diam nibh diam, quam elit, libero nostra ut. Pellentesque vehicula. Eget sed,—}}
{{—dapibus </p>—}}
{{—</div>—}}
{{—</div><!— end post wrap —>—}}

{{—<div class="post–meta"> <!— Meta details —>—}}
{{—<ul class="list–inline metas pull–left"> <!— post metas —>—}}
{{—<li><a href="#">by Rudhi Design</a></li> <!— meta author —>—}}
{{—<li><a href="#">20 Comments</a></li> <!— meta comments —>—}}
{{—<li><a href="#">Read More</a></li> <!— read more link —>—}}
{{—</ul>—}}
{{—<ul class="list–inline meta–detail pull–right"> <!— user meta interaction —>—}}
{{—<li><a href="#"><i class="fa fa–heart"></i></a> 50</li> <!— like button —>—}}
{{—<li><i class="fa fa–eye"></i> 110</li> <!— no. of views —>—}}
{{—</ul>—}}
{{—</div><!— end Meta details —>—}}
{{—</div><!— end Post Wrapper —>—}}

{{—<div class="post–wrap"> <!— Post Wrapper —>—}}
{{—<div class="media post"> <!— post wrap —>—}}
{{—<div class="media–left">—}}
{{—<a href="#"> <!— link to your post single page —>—}}
{{—<img class="media–object" src="http://placehold.it/120x150" alt="...">—}}
{{—<!— Your Post Image —>—}}
{{—</a>—}}
{{—</div>—}}
```

```
        {{—<div class="media–body">—}}
        {{—<p class="small">January 14, 2015</p>—}}
        {{—<a href="#">—}}
        {{—<h5 class="media–heading"><strong>Vel donec et scelerisque vestibulum.—}}
        {{—Condimentum aliquam, mollit magna velit nec</strong></h5>—}}
        {{—</a>—}}
        {{—<p>Tempor vestibulum turpis id ligula mi mattis. Eget arcu vitae mauris amet odio.—
}}
        {{—Diam nibh diam, quam elit, libero nostra ut. Pellentesque vehicula. Eget sed,—}}
        {{—dapibus </p>—}}
        {{—</div>—}}
        {{—</div><!— end post wrap —>—}}

        {{—<div class="post–meta"> <!— Meta details —>—}}
        {{—<ul class="list–inline metas pull–left"> <!— post metas —>—}}
        {{—<li><a href="#">by Rudhi Design</a></li> <!— meta author —>—}}
        {{—<li><a href="#">20 Comments</a></li> <!— meta comments —>—}}
        {{—<li><a href="#">Read More</a></li> <!— read more link —>—}}
        {{—</ul>—}}
        {{—<ul class="list–inline meta–detail pull–right"> <!— user meta interaction —>—}}
        {{—<li><a href="#"><i class="fa fa–heart"></i></a> 50</li> <!— like button —>—}}
        {{—<li><i class="fa fa–eye"></i> 110</li> <!— no. of views —>—}}
        {{—</ul>—}}
        {{—</div><!— end Meta details —>—}}
        {{—</div><!— end Post Wrapper —>—}}

        {{—</div> <!— end Left content col 6 —>—}}

        {{—<div class="col–md–6"> <!— right content col 6 —>—}}

        {{—<div class="post–wrap"> <!— Post Wrapper —>—}}
        {{—<div class="media post"> <!— post wrap —>—}}
        {{—<div class="media–left">—}}
        {{—<a href="#"> <!— link to your post single page —>—}}
        {{—<img class="media–object" src="http://placehold.it/120x150" alt="...">—}}
        {{—<!— Your Post Image —>—}}
        {{—</a>—}}
        {{—</div>—}}
        {{—<div class="media–body">—}}
        {{—<p class="small">January 14, 2015</p>—}}
        {{—<a href="#">—}}
        {{—<h5 class="media–heading"><strong>Vel donec et scelerisque vestibulum.—}}
        {{—Condimentum aliquam, mollit magna velit nec</strong></h5>—}}
        {{—</a>—}}
        {{—<p>Tempor vestibulum turpis id ligula mi mattis. Eget arcu vitae mauris amet odio.—
}}
        {{—Diam nibh diam, quam elit, libero nostra ut. Pellentesque vehicula. Eget sed,—}}
        {{—dapibus </p>—}}
        {{—</div>—}}
```

```
{{—</div><!— end post wrap —>—}}

{{—<div class="post–meta"> <!— Meta details —>—}}
{{—<ul class="list–inline metas pull–left"> <!— post metas —>—}}
{{—<li><a href="#">by Rudhi Design</a></li> <!— meta author —>—}}
{{—<li><a href="#">20 Comments</a></li> <!— meta comments —>—}}
{{—<li><a href="#">Read More</a></li> <!— read more link —>—}}
{{—</ul>—}}
{{—<ul class="list–inline meta–detail pull–right"> <!— user meta interaction —>—}}
{{—<li><a href="#"><i class="fa fa–heart"></i></a> 50</li> <!— like button —>—}}
{{—<li><i class="fa fa–eye"></i> 110</li> <!— no. of views —>—}}
{{—</ul>—}}
{{—</div><!— end Meta details —>—}}
{{—</div><!— end Post Wrapper —>—}}

{{—<div class="post–wrap"> <!— Post Wrapper —>—}}
{{—<div class="media post"> <!— post wrap —>—}}
{{—<div class="media–left">—}}
{{—<a href="#"> <!— link to your post single page —>—}}
{{—<img class="media–object" src="http://placehold.it/120x150" alt="...">—}}
{{—<!— Your Post Image —>—}}
{{—</a>—}}
{{—</div>—}}
{{—<div class="media–body">—}}
{{—<p class="small">January 14, 2015</p>—}}
{{—<a href="#">—}}
{{—<h5 class="media–heading"><strong>Vel donec et scelerisque vestibulum.—}}
{{—Condimentum aliquam, mollit magna velit nec</strong></h5>—}}
{{—</a>—}}
{{—<p>Tempor vestibulum turpis id ligula mi mattis. Eget arcu vitae mauris amet odio.—
}}
{{—Diam nibh diam, quam elit, libero nostra ut. Pellentesque vehicula. Eget sed,—}}
{{—dapibus </p>—}}
{{—</div>—}}
{{—</div><!— end post wrap —>—}}

{{—<div class="post–meta"> <!— Meta details —>—}}
{{—<ul class="list–inline metas pull–left"> <!— post metas —>—}}
{{—<li><a href="#">by Rudhi Design</a></li> <!— meta author —>—}}
{{—<li><a href="#">20 Comments</a></li> <!— meta comments —>—}}
{{—<li><a href="#">Read More</a></li> <!— read more link —>—}}
{{—</ul>—}}
{{—<ul class="list–inline meta–detail pull–right"> <!— user meta interaction —>—}}
{{—<li><a href="#"><i class="fa fa–heart"></i></a> 50</li> <!— like button —>—}}
{{—<li><i class="fa fa–eye"></i> 110</li> <!— no. of views —>—}}
{{—</ul>—}}
{{—</div><!— end Meta details —>—}}
{{—</div><!— end Post Wrapper —>—}}
```

```
{{—<div class="post–wrap"> <!— Post Wrapper —>—}}
{{—<div class="media post"> <!— post wrap —>—}}
{{—<div class="media–left">—}}
{{—<a href="#"> <!— link to your post single page —>—}}
{{—<img class="media–object" src="http://placehold.it/120x150" alt="...">—}}
{{—<!— Your Post Image —>—}}
{{—</a>—}}
{{—</div>—}}
{{—<div class="media–body">—}}
{{—<p class="small">January 14, 2015</p>—}}
{{—<a href="#">—}}
{{—<h5 class="media–heading"><strong>Vel donec et scelerisque vestibulum.—}}
{{—Condimentum aliquam, mollit magna velit nec</strong></h5>—}}
{{—</a>—}}
{{—<p>Tempor vestibulum turpis id ligula mi mattis. Eget arcu vitae mauris amet odio.—
}}
{{—Diam nibh diam, quam elit, libero nostra ut. Pellentesque vehicula. Eget sed,—}}
{{—dapibus </p>—}}
{{—</div>—}}
{{—</div><!— end post wrap —>—}}

{{—<div class="post–meta"> <!— Meta details —>—}}
{{—<ul class="list–inline metas pull–left"> <!— post metas —>—}}
{{—<li><a href="#">by Rudhi Design</a></li> <!— meta author —>—}}
{{—<li><a href="#">20 Comments</a></li> <!— meta comments —>—}}
{{—<li><a href="#">Read More</a></li> <!— read more link —>—}}
{{—</ul>—}}
{{—<ul class="list–inline meta–detail pull–right"> <!— user meta interaction —>—}}
{{—<li><a href="#"><i class="fa fa–heart"></i></a> 50</li> <!— like button —>—}}
{{—<li><i class="fa fa–eye"></i> 110</li> <!— no. of views —>—}}
{{—</ul>—}}
{{—</div><!— end Meta details —>—}}
{{—</div><!— end Post Wrapper —>—}}

{{—</div><!— end right content col 6 —>—}}
{{—</div><!— end row —>—}}

{{—<div class="text–center">—}}
{{—<a href="#" class="btn btn–primary tf–btn color">Load More</a>—}}
{{—</div>—}}
{{—</div><!— end container —>—}}
{{—</div> <!— end fullwidth gray background —>—}}
{{—</div>—}}

<!— Contact Section
   ========================================================= —>
<div id="tf–contact">

   <div class="container"> <!— container —>
```

```
        <div class="section–header">
                <h2>Наши контакты</h2>
                <h5><em>Всегда рады Вам</em></h5>
                <div class="fancy"><span><img class="logo"
src="{{asset('img/icons/logo.png')}}" alt="..."></span></div>
            </div>
          </div><!— end container —>

          <div id="map"></div>  <!— google map —>

          <div id="contact–form" class="container"><!— container —>
            <div class="row"> <!— outer row —>
              <div class="col–md–10 col–md–offset–1"> <!— col 10 with offset 1 to centered –
–>
                <div class="row"> <!— nested row —>

                  <!— contact detail using col 4 —>
                  <div class="col–md–4">
                    <div class="contact–detail">
                      <i class="fa fa–map–marker"></i>
                      <h4 class="smaller">город Алматы, Алмалинский район, улица
Гоголя, 176</h4>
                      <!— address —>
                    </div>
                  </div>
                  <!— contact detail using col 4 —>
                  <div class="col–md–4">
                    <div class="contact–detail">
                      <a href="mailto:info@Aslankurylys.com.ua">
                        <i class="fa fa–envelope–o"></i>
                        <h4>info@Aslankurylys.com.ua</h4>
                      </a><!— email add —>
                    </div>
                  </div>

                  <!— contact detail using col 4 —>
                  <div class="col–md–4">
                    <div class="contact–detail">
                      <a href="tel:+77086933173">
                        <i class="fa fa–phone"></i>
                        <h4>+7086933173</h4>
                      </a> <!— phone no. —>
                    </div>
                  </div>

                </div> <!— end nested row —>
              </div> <!— end col 10 with offset 1 to centered —>
            </div><!— end outer row —>
```

```
        <div class="row text–center"> <!— contact form outer row with centered text—>
                <div class="col–md–10 col–md–offset–1"> <!— col 10 with offset 1 to centered –
–>
                    <form id="contact–form" class="form" name="sentMessage" novalidate> <!—
form wrapper —>
                        <div class="row"> <!— nested inner row —>

                            <!— Input your name —>
                            <div class="col–md–4">
                                <div class="form–group"> <!— Your name input —>
                                    <input type="text" autocomplete="off" class="form–control"
placeholder="Ваше имя *"
                                        id="name" required
                                        data–validation–required–message="Пожалуйста,
представьтесь">
                                    <p class="help–block text–danger"></p>
                                </div>
                            </div>

                            <!— Input your email —>
                            <div class="col–md–4">
                                <div class="form–group"> <!— Your email input —>
                                    <input type="email" autocomplete="off" class="form–control"
                                        placeholder="Ваша эл.почта *" id="email" required
                                        data–validation–required–message="Пожалуста, введите адрес
эл.почты">
                                    <p class="help–block text–danger"></p>
                                </div>
                            </div>

                            <!— Input your Phone no. —>
                            <div class="col–md–4">
                                <div class="form–group"> <!— Your email input —>
                                    <input type="text" autocomplete="off" class="form–control"
                                        placeholder="Ваш номер телефона *" id="phone" required
                                        data–validation–required–message="Пожалуйста, введите
номер телефона">
                                    <p class="help–block text–danger"></p>
                                </div>
                            </div>

                        </div><!— end nested inner row —>

                        <!— Message Text area —>
                        <div class="form–group"> <!— Your email input —>
                            <textarea class="form–control" rows="7" placeholder="Текст
сообщения..." id="message"
                                required
```

```
        data–validation–required–message="Пожалуйста, напишите сообщение"></textarea>
                            <p class="help–block text–danger"></p>
                            <div id="success"></div>
                        </div>
                        <button type="submit" class="btn btn–primary tf–btn color">Отправить сообщение</button>
                        <!— Send button —>

                    </form><!— end form wrapper —>
                </div><!— end col 10 with offset 1 to centered —>
            </div> <!— end contact form outer row with centered text—>

        </div><!— end container —>

        <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>
        {{—    <script type="text/javascript" src="{{asset('js/app.js')}}"></script>—}}
        <script type="text/javascript" src="{{asset('js/theme/bootstrap.js')}}"></script>
        <script type="text/javascript" src="{{asset('js/theme/owl.carousel.js')}}"></script><!— Owl Carousel Plugin —>

        <script type="text/javascript" src="{{asset('js/theme/SmoothScroll.js')}}"></script>

        <!— Google Map —>
        <script type="text/javascript"
                src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDUag1a_TKNQLxD–vuLldAdoAtPiZhGjFY"></script>
        <script type="text/javascript" src="{{asset('js/theme/map.js')}}"></script>

        <!— Parallax Effects —>
        <script type="text/javascript" src="{{asset('js/theme/skrollr.js')}}"></script>
        <script type="text/javascript" src="{{asset('js/theme/imagesloaded.js')}}"></script>

        <!— Portfolio Filter —>
        <script type="text/javascript" src="{{asset('js/theme/jquery.isotope.js')}}"></script>

        <!— LightBox Nivo —>
        <script type="text/javascript" src="{{asset('js/theme/nivo–lightbox.min.js')}}"></script>

        <!— Contact page—>
        <script type="text/javascript" src="{{asset('js/theme/jqBootstrapValidation.js')}}"></script>
        <script type="text/javascript" src="{{asset('js/theme/contact.js')}}"></script>

        <!— Javascripts
        ================================================== —>
        <script type="text/javascript" src="{{asset('js/theme/main.js')}}"></script>

        <?php
```

```
use TCG\Voyager\Facades\Voyager;
$google_analytics_client_id = Voyager::setting('google_analytics_client_id');

if(isset($google_analytics_client_id) && !empty($google_analytics_client_id)): ?>
<script>
   (function (w, d, s, g, js, fs) {
      g = w.gapi || (w.gapi = {});
      g.analytics = {
         q: [], ready: function (f) {
            this.q.push(f);
         }
      };
      js = d.createElement(s);
      fs = d.getElementsByTagName(s)[0];
      js.src = 'https://apis.google.com/js/platform.js';
      fs.parentNode.insertBefore(js, fs);
      js.onload = function () {
 g.load('analytics');
      };
   }(window, document, 'script'));
   gapi.analytics.ready(function () {
      gapi.analytics.auth.authorize({
         container: 'embed–api–auth–container',
         clientid: '<?php echo e($google_analytics_client_id); ?>'
      });
   });
</script>

<?php endif; ?>

<script>
   $.ajaxSetup({
      headers: {
         'X–CSRF–TOKEN': $('meta[name="csrf–token"]').attr('content')
      }
   });
</script>
</body>
</html>
```

Продолжение приложения Б

```
},
 "author": "Alisher",
 "license": "ISC",
 "dependencies": {
   "body-parser": "^1.18.3",
   "connect-flash": "^0.1.1",
   "dotenv": "^8.0.0",
```

```
    "ejs": "^2.6.1",
    "express": "^4.16.4",
    "express-session": "^1.16.1",
    "method-override": "^3.0.0",
    "mongoose": "^5.5.2",
    "node-geocoder": "^3.22.0",
    "passport": "^0.4.0",
    "passport-local": "^1.0.0",
    "passport-local-mongoose": "^5.0.1"
  }
}

<% include ../partials/header %>
<div class="container">
<h1 style="text-align: center;">Edit "<%=campground.name%>"</h1>

<form action="/campgrounds/<%= campground._id %>?_method=PUT"
method="POST" style="width: 30%; margin: 25px auto;">
    <div class="form-group">
        <label for="name">Имя</label>
        <input id="name" class="form-control" type="text"
name="campground[name]" value="<%= campground.name %>">
    </div>
    <div class="form-group">
        <label for="image">Изображение</label>
        <input id="image" class="form-control" type="text"
name="campground[image]" value="<%=campground.image%>">
    </div>
    <div class="form-group">
        <label for="description">Описание</label>
        <textarea class="form-control" id="description"
name="campground[description]" ><%=campground.description%></textarea>
    </div>
    <div class="form-group">
        <label for="price">Цена</label>
```

Продолжение приложения Б

```
   <input id="price" class="form-control" type="number" name="campground[cost]"
value="<%= campground.cost %>" step="0.01" min="0" required>
    </div>
    <div class="form-group">
        <label for="location">Расположение</label>
```

```
      <input class="form-control" type="text" name="location" id="location" value="<%= campground.location %>">
   </div>
   <div class="form-group">
      <button class="btn btn-lg btn-primary btn-block">Submit</button>
   </div>
   <a href="/campgrounds">Back to The Campgrounds page</a>
</form>

</div>
<% include ../partials/footer %>

var express          = require("express"),
   router            = express.Router(),
   passport          = require("passport"),
   User              = require("../models/user");

//root route
router.get("/", function(req, res) {
   res.render("landing");
});
//show register form route
router.get("/register", function(req, res) {
   res.render("register");
});
//handle sign up logic
router.post("/register", function(req, res){
   User.register({username: req.body.username}, req.body.password, function(err, user){
      if(err){
         req.flash("error", err.message);
         return res.render("register");
      }
      passport.authenticate("local")(req, res, function(){
         req.flash("success", "Welcome to YelpCamp " + user.username);
         res.redirect("/campgrounds");
      } )
```

Продолжение приложения Б

```
 })
});
//login form
router.get("/login", function(req, res) {
   res.render("login");
```

```
});
//handling login logic
router.post("/login", passport.authenticate("local", {
    successRedirect: "/campgrounds",
    failureRedirect: "/login"
}), function(req, res){
});

//handling logout
router.get("/logout", function(req, res) {
    req.logout();
    req.flash("success", "Logged you out");
    res.redirect("/campgrounds");
});

//middleware
function isLoggedIn(req, res, next){
    if(req.isAuthenticated()){
        return next();
    }
    res.redirect("/login");
}

module.exports = router;

var express             = require("express"),
    router              = express.Router(),
    Campground          = require("../models/campground"),
    Comment             = require("../models/comment");

router.get("/campgrounds/:id/comments/new", isLoggedIn ,function(req, res) {
   Campground.findById(req.params.id, function(err, foundCampground){
       if(err){
           console.log(err);
       }
       else{
           res.render("comments/new", {campground: foundCampground});
       }
```

Продолжение приложения Б

```
   })
});
router.get("/budget", function(req, res){
    res.render("budget");
```

```
});
router.post("/campgrounds/:id/comments",isLoggedIn,  function(req, res){
   Campground.findById(req.params.id, function(err, campground) {
      if(err){
         console.log(err);
         res.redirect("/campgrounds");
      }
      else{
         console.log("WORKS");
         console.log(req.body.comment);
         Comment.create(req.body.comment, function(err, comment){
            if(err){
               console.log(err);
            }
            else{
               //add username and it to comment
               comment.author.id = req.user._id;
               comment.author.username = req.user.username;
                console.log("THE AUTHOR'S NAME IS " + req.user.username);
               //save comment
               comment.save();
               campground.comments.push(comment);
               console.log(comment);
               campground.save();
               req.flash("success", "Successfully added comment");
               res.redirect("/campgrounds/" + campground._id);
            }
         });
      }
   })
});

router.get("/campgrounds/:id/comments/:comment_id/edit",
checkCommentOwnership, function(req, res) {
   Comment.findById(req.params.comment_id, function(err, comment) {
      if(err){
         console.log(err);
      }
      else{
```

Продолжение приложения Б

```
  res.render("comments/edit",  {comment:  comment,  campground_id:
req.params.id});
      }
```

```
    })
});

router.put("/campgrounds/:id/comments/:comment_id",  checkCommentOwnership,
function(req, res){
    Comment.findByIdAndUpdate(req.params.comment_id,        req.body.comment,
function(err, updatedComment){
        if(err){
            console.log(err);
            res.redirect("back");
        } else {
            res.redirect("/campgrounds/" + req.params.id);
        }
    });
});

router.delete("/campgrounds/:id/comments/:comment_id",
checkCommentOwnership, function(req, res){
    Comment.findByIdAndRemove(req.params.comment_id,                function(err,
removedComment){
        if(err){
            console.log("back");
        } else {
            req.flash("success",  "Comment deleted");
            res.redirect("/campgrounds/" + req.params.id);
        }
    });
});

function checkCommentOwnership(req, res, next) {
    if (req.isAuthenticated()) {
        Comment.findById(req.params.comment_id, function(err, foundComment) {
            if (err) {
                req.flash("error", "Comment not found");
                res.redirect("back");
            }
            else {
                if (foundComment.author.id.equals(req.user._id)) {
                    next();
                }
```

Продолжение приложения Б

```
    else {
                    req.flash("error", "You don't have permission to do that");
```

```
                res.redirect("back");
            }
        }
    });
  }
  else {
    req.flash("error",  "you need to be logged in to do that");
    res.redirect("/login");
  }
}


function isLoggedIn(req, res, next){
  if(req.isAuthenticated()){
    return next();
  }
  req.flash("error",  "You need to be logged in to do that");
  res.redirect("/login");
}

module.exports = router;

var express        = require("express"),
  router         = express.Router(),
  Campground     = require("../models/campground"),
  NodeGeocoder   = require('node-geocoder');

var options = {
 provider: 'google',
 httpAdapter: 'https',
 apiKey: process.env.GEOCODER_API_KEY,
 formatter: null
};

var geocoder = NodeGeocoder(options);

// router.post("/", isLoggedIn, function(req, res) {
//    var name = req.body.name;
//    var image = req.body.url;
//    var desc = req.body.descriptionOf;
//    var author = {
```

Продолжение приложения Б

```
//        id: req.user._id,
```

```
//         username: req.user.username
//     }
//     var newCampground = { name: name, image: image, description: desc, author:
author };
//     Campground.create(newCampground, function(err, newlyCampground) {
//         if (err) {
//             console.log(err);
//         }
//         else {
//             console.log(newlyCampground);
//             res.redirect("/campgrounds");
//         }

//     })

// })

//CREATE - add new campground to DB
router.post("/", isLoggedIn, function(req, res){
  // get data from form and add to campgrounds array
  var name = req.body.name;
  var image = req.body.image;
  var desc = req.body.description;
  var cost = req.body.cost;
  var author = {
      id: req.user._id,
      username: req.user.username
  }
  geocoder.geocode(req.body.location, function (err, data) {
    if (err || !data.length) {
      req.flash('error', 'Invalid address');
      return res.redirect('back');
    }
    var lat = data[0].latitude;
    var lng = data[0].longitude;
    var location = data[0].formattedAddress;
    var newCampground = {name: name, image: image, description: desc, cost: cost,
author:author, location: location, lat: lat, lng: lng};
    // Создать новое место и сохранить в БД
    Campground.create(newCampground, function(err, newlyCreated){
        if(err){
            console.log(err);
```

Продолжение приложения Б

```
  } else {
          //redirect back to campgrounds page
          console.log(newlyCreated);
          res.redirect("/campgrounds");
      }
    });
  });
});

router.get("/", function(req, res) {
   Campground.find({}, function(err, campgrounds) {
      if (err) {
         console.log(err);
      }
      else {
         res.render("campgrounds/index", { campgrounds: campgrounds,
currentUser: req.user })
      }
   });

});

router.get("/:id/edit", checkCampgroundOwnership, function(req, res) {
      Campground.findById(req.params.id, function(err, foundCampground) {
               res.render("campgrounds/edit", { campground: foundCampground });
            });
});




// router.put("/:id", checkCampgroundOwnership, function(req, res) {
//             Campground.findByIdAndUpdate(req.params.id,   req.body.campground,
function(err, updated) {
//        if (err) {
//           console.log("error goes here");
//           console.log(err);
//           res.redirect("/campgrounds");
//        }
//        else {
//           res.redirect("/campgrounds/" + req.params.id);
//        }
//    });
```

Продолжение приложения Б

```
// });

// UPDATE CAMPGROUND ROUTE
router.put("/:id", checkCampgroundOwnership, function(req, res){
  geocoder.geocode(req.body.location, function (err, data) {
    if (err || !data.length) {
      req.flash('error', 'Invalid address');
      return res.redirect('back');
    }
    req.body.campground.lat = data[0].latitude;
    req.body.campground.lng = data[0].longitude;
    req.body.campground.location = data[0].formattedAddress;

    Campground.findByIdAndUpdate(req.params.id, req.body.campground,
function(err, campground){
        if(err){
            req.flash("error", err.message);
            res.redirect("back");
        } else {
            req.flash("success","Successfully Updated!");
            res.redirect("/campgrounds/" + campground._id);
        }
    });
  });
});

router.delete("/:id", checkCampgroundOwnership, function(req, res) {
    Campground.findByIdAndRemove(req.params.id, req.body.blog, function(err,
removedCampground) {
        if (err) {
            res.render("/campgrounds");
        }
        else {
            res.redirect("/campgrounds");
        }
    })
})

router.get("/new", isLoggedIn, function(req, res) {
    res.render("campgrounds/new");
})

router.get("/:id", function(req, res) {
```

Продолжение приложения Б

```
 Campground.findById(req.params.id).populate("comments").exec(function(err,
foundCampground) {
      if (err) {
         console.log(err);
      }
      else {
         console.log(foundCampground);
         res.render("campgrounds/show", { campgrounds: foundCampground });
      }
   });
});

function checkCampgroundOwnership(req, res, next) {
   if (req.isAuthenticated()) {
      Campground.findById(req.params.id, function(err, foundCampground) {
         if (err) {
            req.flash("error", "Campground not found");
            res.redirect("back");
         }
         else {
            if (foundCampground.author.id.equals(req.user._id)) {
               next();
            }
            else {
               req.flash("error", "У Вас нет разрешения на это!");
               res.redirect("back");
            }
         }
      });
   }
   else {
      req.flash("error",  "you need to be logged in to do that");
      res.redirect("/login");
   }
}

function isLoggedIn(req, res, next) {
   if (req.isAuthenticated()) {
      return next();
   }
   req.flash("error", "You need to be logged in to do that");
   res.redirect("/login");
```

```
}
```

Продолжение приложения Б

```
module.exports = router;

var express          = require("express"),
   app               = express(),
   router            = express.Router();


   router.get("/budget", function(req, res){
   res.render("budget");
});

module.exports = router;

/*********************************************
*** GENERAL
*********************************************/

* {
   margin: 0;
   padding: 0;
   box-sizing: border-box;
}

.clearfix::after {
   content: "";
   display: table;
   clear: both;
}

body {
   color: #555;
   font-family: Open Sans;
   font-size: 16px;
   position: relative;
   height: 100vh;
   font-weight: 400;
}

.right { float: right; }
.red { color: #FF5049 !important; }
```

```
.red-focus:focus { border: 1px solid #FF5049 !important; }
```

Продолжение приложения Б

```
.top {
   height: 40vh;
   background-image: linear-gradient(rgba(0, 0, 0, 0.35), rgba(0, 0, 0, 0.35)),
url(https://i.ibb.co/CK90V70/back.png);
   background-size: cover;
   background-position: center;
   position: relative;
}

.budget {
   position: absolute;
   width: 350px;
   top: 50%;
   left: 50%;
   transform: translate(-50%, -50%);
   color: #fff;
}

.budget__title {
   font-size: 18px;
   text-align: center;
   margin-bottom: 10px;
   font-weight: 300;
}

.budget__value {
   font-weight: 300;
   font-size: 46px;
   text-align: center;
   margin-bottom: 25px;
   letter-spacing: 2px;
}

.budget__income,
.budget__expenses {
   padding: 12px;
   text-transform: uppercase;
}

.budget__income {
```

```
    margin-bottom: 10px;
    background-color: #28B9B5;
}
```

Продолжение приложения Б

```
.budget__expenses {
    background-color: #FF5049;
}

.budget__income--text,
.budget__expenses--text {
    float: left;
    font-size: 13px;
    color: #444;
    margin-top: 2px;
}

.budget__income--value,
.budget__expenses--value {
    letter-spacing: 1px;
    float: left;
}

.budget__income--percentage,
.budget__expenses--percentage {
    float: left;
    width: 34px;
    font-size: 11px;
    padding: 3px 0;
    margin-left: 10px;
}

.budget__expenses--percentage {
    background-color: rgba(255, 255, 255, 0.2);
    text-align: center;
    border-radius: 3px;
}


/**********************************************
*** BOTTOM PART
**********************************************/
```

```
/***** FORM *****/
.add {
   padding: 14px;
   border-bottom: 1px solid #e7e7e7;
```

Продолжение приложения Б

```
  background-color: #f7f7f7;
}

.add__container {
   margin: 0 auto;
   text-align: center;
}

.add__type {
   width: 55px;
   border: 1px solid #e7e7e7;
   height: 44px;
   font-size: 18px;
   color: inherit;
   background-color: #fff;
   margin-right: 10px;
   font-weight: 300;
   transition: border 0.3s;
}

.add__description,
.add__value {
   border: 1px solid #e7e7e7;
   background-color: #fff;
   color: inherit;
   font-family: inherit;
   font-size: 14px;
   padding: 12px 15px;
   margin-right: 10px;
   border-radius: 5px;
   transition: border 0.3s;
}

.add__description { width: 400px;}
.add__value { width: 100px;}

.add__btn {
   font-size: 35px;
```

```
    background: none;
    border: none;
    color: #28B9B5;
    cursor: pointer;
    display: inline-block;
```

Продолжение приложения Б

```
   vertical-align: middle;
    line-height: 1.1;
    margin-left: 10px;
}

.add__btn:active { transform: translateY(2px); }

.add__type:focus,
.add__description:focus,
.add__value:focus {
    outline: none;
    border: 1px solid #28B9B5;
}

.add__btn:focus { outline: none; }

/***** LISTS *****/
.container {
    width: 1000px;
    margin: 60px auto;
}

.income {
    float: left;
    width: 475px;
    margin-right: 50px;
}

.expenses {
    float: left;
    width: 475px;
}

h2 {
    text-transform: uppercase;
    font-size: 18px;
    font-weight: 400;
```

```
    margin-bottom: 15px;
}

.icome__title { color: #28B9B5; }
.expenses__title { color: #FF5049; }
```

Продолжение приложения Б

```
.item {
    padding: 13px;
    border-bottom: 1px solid #e7e7e7;
}

.item:first-child { border-top: 1px solid #e7e7e7; }
.item:nth-child(even) { background-color: #f7f7f7; }

.item__description {
    float: left;
}

.item__value {
    float: left;
    transition: transform 0.3s;
}

.item__percentage {
    float: left;
    margin-left: 20px;
    transition: transform 0.3s;
    font-size: 11px;
    background-color: #FFDAD9;
    padding: 3px;
    border-radius: 3px;
    width: 32px;
    text-align: center;
}

.income .item__value,
.income .item__delete--btn {
    color: #28B9B5;
}

.expenses .item__value,
.expenses .item__percentage,
```

```
.expenses .item__delete--btn {
   color: #FF5049;
}


.item__delete {
   float: left;
```

Продолжение приложения Б

```
}

.item__delete--btn {
   font-size: 22px;
   background: none;
   border: none;
   cursor: pointer;
   display: inline-block;
   vertical-align: middle;
   line-height: 1;
   display: none;
}

.item__delete--btn:focus { outline: none; }
.item__delete--btn:active { transform: translateY(2px); }

.item:hover .item__delete--btn { display: block; }
.item:hover .item__value { transform: translateX(-20px); }
.item:hover .item__percentage { transform: translateX(-20px); }


.unpaid {
   background-color: #FFDAD9 !important;
   cursor: pointer;
   color: #FF5049;

}

.unpaid .item__percentage { box-shadow: 0 2px 6px 0 rgba(0, 0, 0, 0.1); }
.unpaid:hover .item__description { font-weight: 900; }


.thumbnail img {
   width: 100%;
}
```

```
.thumbnail {
   padding: 0;
}
.thumbnail .caption-full {
   padding: 9px;
}
#Username {
   color: black;
```

Продолжение приложения Б

```
}
#thumba {
   height: 170px;
   width: 100%;
}
#deleteForm {
   display: inline;
}
/* Google Maps */
#map {
   height: 400px;
   width: 100%;
}
.photos {
   max-height: 560px;
   max-width: 1020px;
}
#preCenter {
   padding: 9px;
}
body {
        background: url(https://images.unsplash.com/photo-1453368228038-
2fda759df319?ixlib=rb-1.2.1&auto=format&fit=crop&w=1934&q=80) no-repeat
center center fixed;
        background-repeat: no-repeat;
        background-position: center;
        background-size: cover;
}

#commentUser {
   color: green;
}
```

```
var mongoose = require("mongoose");

var campgroundSchema = new mongoose.Schema({
  name: String,
  image: String,
  description: String,
  location: String,
  lat: Number,
  lng: Number,
```

Продолжение приложения Б

```
  cost: Number,
  author: {
    id: {
      type: mongoose.Schema.Types.ObjectId,
      ref: "User"
    },
    username: String
  },
  comments: [
    {
      type: mongoose.Schema.Types.ObjectId,
      ref: "Comment"
    }
  ]
});

module.exports = mongoose.model("Campground", campgroundSchema);

var mongoose = require("mongoose");

var commentSchema = new mongoose.Schema({
   text: String,
   author: {
      id: {
         type: mongoose.Schema.Types.ObjectId,
         ref: "User"
      },
      username: String
   }

});
```

```
module.exports = mongoose.model("Comment", commentSchema);

var mongoose = require("mongoose"),
passportLocalMongoose =   require("passport-local-mongoose");

var userSchema = new mongoose.Schema({
   username: String,
   password: String
});
```

Продолжение приложения Б

```
userSchema.plugin(passportLocalMongoose);

module.exports = mongoose.model("User", userSchema);
```