**Курсовая работа «Web-студия»**

Содержание

[Введение 5](#_Toc120093763)

[1 Теоретическая часть 10](#_Toc120093764)

[1.1 Выбор языка и среды программирования 10](#_Toc120093765)

[1.2 Разработка системы требований 14](#_Toc120093766)

[1.3 Разработка функциональной спецификации 16](#_Toc120093767)

[2 Практическая часть 19](#_Toc120093768)

[2.1 Разработка CRC-карточек 19](#_Toc120093769)

[2.2 Структурная и функциональная схемы 19](#_Toc120093770)

[2.3 Разработка диаграммы деятельности 19](#_Toc120093771)

[2.4 Разработка ER-диаграммы 19](#_Toc120093772)

[2.5 Разработка словаря данных 19](#_Toc120093773)

[2.6 Тестирование программного обеспечения 19](#_Toc120093774)

[Заключение 20](#_Toc120093775)

[Список литературы 21](#_Toc120093776)

Введение

Создание и развитие сети Интернет расценивается как колоссальный скачок из эры укоренившегося бизнеса в эру технологий, реализуемых в режиме настоящего времени.

Интернет стал неотъемлемой частью нашей жизни. Путешествуя по сети Интернет, мы встречаем разнообразные веб-сайты. На одних продают вещи, на других приглашают найти друзей для общения. Есть широко востребованные информационные порталы, а есть сайты, посвященные отдельным людям или компаниям. Тенденция такова, что для успеха необходимо заявить о себе в Интернете, а значит, собственная персональная страница должна быть у любой фирмы и даже у личности.

Интернет в наши дни — это целая индустрия, быстро проникающая во все области человеческой деятельности. Огромное количество компаний во всем мире видят в Интернет большой коммерческий потенциал и возможность перевода своего бизнеса на качественно новый уровень. Благодаря постоянному совершенствованию технологий, скорость доступа растет, а цена становится всё более приемлемой.

Интернет находит своё применение во многих сферах, не обходит стороной и деятельность коммерческих фирм. Вот несколько причин, по которым фирмы всё чаще приобретают собственные сайты и прибегают к использованию Интернета в осуществлении своей деятельности:

* собственный сайт положительно влияет на престиж фирмы и её имидж;
* возможность находить новых деловых партнеров и новые рынки сбыта;
* возможность очень быстро предоставлять необходимую информацию;
* возможность очень быстро оказывать информационную поддержку своим клиентам и отвечать на интересующие их вопросы (иногда разговаривая по телефону, сотрудник фирмы может не найти с ходу нужного ответа, а клиент упустит из виду какой-нибудь важный момент; сообщение в виде текста способствует понятной и подробной формулировке как вопроса, так и ответа);
* возможность привлечения большего объёма целевой аудитории (как правило, в сети Интернет люди ищут нужную информацию, используя поисковые системы, где они конкретно указывают то, что их интересует;
* в качестве аналогичного примера можно привести покупку газеты с рекламным содержанием и объявлениями. Однако поиск через Интернет имеет неоспоримые преимущества по скорости и удобству);
* возможность предоставления подробной информации о предлагаемых товарах и услугах. Это можно назвать самым главным преимуществом (теле- и радиореклама может привлечь внимание к фирме, но она не способна дать подробную информацию, рассказать о достоинствах и методах предприятия. С помощью веб-сайта можно разместить фото и описание всех предлагаемых товаров, рассказать об оказываемых услугах, подавая информацию максимально эффективно и в неограниченном объёме).

Эти преимущества может дать только собственный сайт, спроектированный и реализованный под заказ.

Для создания одного лишь небольшого сайта потребуется много знаний, умений и времени, а чем больше необходимо получить от сайта, тем больше нужно в него вложить. Чтобы сделать хороший сайт, не всегда достаточно одного "человека-оркестра".

Для создания полноценной информационной системы требуется работа целой команды разработчиков для быстрого и качественного выполнения заказа.

Команда для разработки сайта состоит из следующих специалистов:

1. Менеджер по продажам

С него начинается создание сайта несмотря на то, что он не принимает в нем непосредственного участия. Менеджер предлагает варианты подходящих услуг, общается с клиентами по телефону, электронной почте, при личной встрече. Обсуждает все детали возможного сотрудничества и организовывает подписание договора об оказании услуг. Именно с ним решаются все финансовые и юридические вопросы, связанные с созданием сайта.

2. Менеджер проекта

После подписания договора и оплаты услуг к работе подключается менеджер проекта, который организует совместный труд всех остальных IT-специалистов. Он согласовывает промежуточные и конечные этапы создания и продвижения сайта. Менеджер запрашивает необходимую для создания сайта информацию, планирует структуру страниц и определяет, каким будет проект в конечном итоге.

3. Веб-дизайнер

Он занимается графическим и художественным оформлением сайта. Всё, что касается приятного внешнего вида web-проекта – располагается в области компетенции данного специалиста. Он создаёт красивые интерфейсы, кнопки, меню и прочее. Веб-дизайнер занимается прорисовкой макетов дизайна. От него зависит первое впечатление о сайте.

4. Верстальщик

Данный специалист занимается HTML-версткой страниц сайта. HTML — это язык гипертекстовой разметки, который используется при создании сайта. Верстальщик воплощает концепцию веб-дизайнера на языке HTML-кода. Реализовывает на практике адаптивную верстку, благодаря которой все элементы и блоки сайта качественно отображаются на экранах различных устройств - компьютерах, смартфонах, планшетах.

5. Веб-программист

Этот специалист занимается проектированием будущего сайта, программирует интерактивные блоки и тестирует их работоспособность, заботится о том, чтобы сайт быстро загружался в браузере. Он создает блоки обратной связи, контактные формы и т.д. После всех выполненных действий проверяет грамотную работу всего функционала сайта. Веб-программист настраивает маршруты файлов robots.txt и htaccess.

6. Контент-менеджер

Занимается наполнением сайта актуальной информацией: размещает текста и фотографий от заказчика, пишет и редактирует новые текста, ищет качественных фотографий для размещения в статьях или других блоках сайта.

7. Специалист по контекстной рекламе

Он занимается ведением рекламных кампаний в сервисах Яндекс. Директ и Google Adwords, подбирает эффективные ключевые слова для рекламной кампании сайта, рассчитывает рекламный бюджет, составляет привлекательные рекламные объявления, занимается настройкой таргетинга и ретаргетинга, анализирует эффективность созданных объявлений. Благодаря специалисту по контекстной рекламе на сайт попадают целевые посетители сразу же после запуска рекламной кампании.

8. SEO-специалист

В его задачи входит продвижение интернет-сайта в поисковых системах Яндекс и Google. Он подбирает ключевые слова, по которым будет осуществляться SEO-продвижение. Распределяет ключевые запросы по страницам сайта. Уменьшает вес изображений, оптимизирует страницы и тексты под ключевые фразы, чтобы в перспективе сайт мог попасть на первую страницу выдачи в поисковых системах по определенным запросам пользователей. SEO-специалист осуществляет внутреннюю перелинковку на сайте, увеличивая количество внутренних ссылок. Занимается наращиванием внешней ссылочной массы для повышения репутации веб-ресурса в глазах Яндекса и Google.

Заказчикам, в свою очередь, придется обращаться ко всем необходимым для его проекта разработчикам по отдельности, что крайне неудобно и время затратно. Разумнее всего, им будет обратиться к уже сформировавшейся компании разработчиков, которые умеют работать вместе и смогут выполнить проект от начала и до конца.

Веб-студия — это компания, которая занимается разработкой и проектированием сайтов. Это всегда группа людей, где каждый отвечает за свою часть работы.

Таким образом, актуальность создания web-студии обуславливается необходимостью объединения разработчиков для более комфортной и организованной работы и сокращением времени на разработку при увеличении ее качества, а сам сайт для web-студии станет ее лицом, поможет заявить о себе и продвинуть ее в сети Интернет, зарекомендовать себя на рынке и эффективнее выстроить работу.

Цель курсовой работы разработать сайт Web-студии и размесить его на хостинге.

Для достижения цели необходимо решить ряд задач:

* исследовать изучаемую область и ее особенности;
* изучить технологии создания веб сайта;
* составить систему требований;
* провести анализ аналогичных сайтов других Web-студий;
* разработать макет и структуру сайта;
* произвести UML – проектирование сайта;
* спроектировать базу данных и диаграммы «Сущность - связь»;
* создать графический макет веб-сайта;
* разработать ИС Web-студия;
* протестировать информационную систему;
* составить программную документацию.

Роль разработчика заключается в разработке документации, дизайна, схем, базы данных, самого сайта, его запуска, сопровождении и презентации.

Практическая значимость разрабатываемого web-сайта может заключаться в его использовании в качестве реального сайта начинающей web-студии после модифицирования под конкретные требования компании.

Так же сайт станет частью портфолио его разработчика и может быть использован в качестве демонстрации его знаний и возможностей.

1 Теоретическая часть

1.1 Выбор языка и среды программирования

Для реализации проекта необходимо выбрать инструменты, с помощью которых будет производится разработка. Для начала необходимо определить, будет ли разрабатываться сайт вручную, с использованием CMS или фреймворков.

Использование CMS, таких как WordPress, Joomla и др. не целесообразно в рамках данного проекта. CMS занимают довольно много места, потребуется дополнительное время на изучения их архитектур, скорость работы снижена за счет обращений к БД, высокое потребление системных ресурсов, сложно адаптировать для нетипичных задач, в многих CMS периодически находят уязвимости и они постоянно требуются обновлений.

Разработка на фреймворках, таких как Laravel, Yii и др. отнимает много времени, так как большинство функционала придётся создавать самостоятельно с нуля; для того, чтобы разбираться с готовым кодом, написанным другим программистом, потребуется большое количество времени и практика, поэтому без достаточного опыта создать большой сайт не получится. Помимо этого, скорость загрузки некоторых приложений может быть снижена за счет объема фреймворка, а из-за необходимости предварительно скомпилировать код для многократного использования, требуется больше времени для тестирования.

Подводя итог вышенаписанному, наиболее удачным вариантом будет самостоятельная разработка проекта без готовых решений.

Разработка делится на 2 основных этапа: Frontend и Backend.

Frontend-разработка — это создание пользовательского интерфейса на клиентской стороне веб‑сайта или приложения. Это всё, что видит пользователь, когда открывает веб-страницу, и с чем он взаимодействует: кнопки, баннеры и анимация. Frontend сайта или веб-приложения можно разделить на три большие части:

* HTML (HyperText Markup Language) — язык разметки документов, на котором создают структуру страницы: заголовки, абзацы, списки и так далее.
* CSS (Cascading Style Sheets) — язык для описания и стилизации внешнего вида документа. Благодаря CSS-коду браузер понимает, как именно отображать элементы. Например, CSS задаёт цвета и параметры шрифтов, определяет, как будут располагаться разные блоки сайта. Ещё он позволяет воспроизводить документ в разных видах: для печати (обычной или шрифтом Брайля), вывода на экран или для чтения скринридером.
* JavaScript — это язык, который оживляет веб-страницы. Его задача — реагировать на действия пользователя, обрабатывать клики мышки, перемещения курсора, нажатия клавиш. Ещё он посылает запросы на сервер и загружает данные без перезагрузки страницы, позволяет вводить сообщения и многое другое.

Для организации структуры и удобства разработки используется такой инструмент, а если быть точнее, то «сборщик» под названием Gulp.

Gulp — менеджер для организации и выполнения задач при разработке приложений с использованием платформы Node.js Он может выполнять разные задачи: формировать файлы стилей, объединять и минифицировать файлы, оптимизировать изображения, транспилировать код и т. д. При выполнении этих задач происходит взаимодействие с рабочим окружением: обратиться к файловой системе, создать или сохранить итоговый файл, запустить процесс локального web-сервера, показать результат в браузере.

Таким образом разработка Frontend части будет производится с помощью HTML, CSS и JavaScript. В качестве дополнительного инструмента разработки будет использован сборщик GULP.

Backend — это разработка бизнес-логики продукта (сайта или веб-приложения). Backend отвечает за взаимодействие пользователя с внутренними данными, которые потом отображает frontend. Другими словами, это то, что скрыто от глаз пользователя и происходит вне его браузера и компьютера.

В качестве языков программирования, на которых может производится разработка, представлены PHP, Python и Java.

Язык Java является надежным, но разработка на нем потребует большего количества времени. Он хорошо показывает себя в больших проектах, но для реализации данного проекта целесообразнее использовать более простой язык. Python за счет своей универсальности довольно обширен, поэтому только его изучение потребует много времени. Т.о. в качестве языка программирования выбран язык PHP.

Главными преимуществами языка PHP являются простота в изучении и довольно высокая производительность, но, помимо этого, можно отметить следующие плюсы данного языка:

* мощность и гибкость, отлично подходит как для небольших, так и для крупных проектов;
* простота изучения;
* большое сообщество, следовательно, можно найти много готовых решений и инструкций;
* имеет большое количество дополнительных расширений и библиотек, которые увеличивают его функциональность;
* является полностью бесплатным и распространяется со свободной лицензией, поэтому его смело могут применять как частные лица, так и коммерческие организации;
* не требователен и может применять на всех известных видах серверов.

Backend часть не может быть реализована без работы с Базой Данных. Лучшим выбором является СУБД MySQL за счет ее бесплатности, высокой скорости обработки данных, популярности, универсальности, поддержки SQL (что обеспечивает высокий уровень кроссплатформенности данных и кода), использования системы привилегий, хеширования паролей и работы в связке с (выбранным для этого проекта) языком PHP.

Для более комфортной и быстрой работы следует использовать программу PhpMyAdmin.

PhpMyAdmin - веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL. PhpMyAdmin позволяет через браузер и не только осуществлять администрирование сервера MySQL, запускать команды SQL и просматривать содержимое таблиц и баз данных.

Некоторые из ключевых преимуществ, которые предоставляет phpMyAdmin:

* Веб−интерфейс - Будучи веб-интерфейсом, пользовательский интерфейс phpMyAdmin доступен с помощью веб-браузера, и этот интерфейс доступен на всех платформах, где может работать веб-браузер.
* Графический интерфейс phpMyAdmin предоставляет графический интерфейс для запуска команд SQL и выполнения операций SQL и делает его довольно простым в использовании по сравнению с консольными редакторами sql.
* Интерфейс скрипта phpMyAdmin предоставляет интерфейс скрипта для запуска PHP-скриптов для подключения к базам данных и выполнения пользовательских операций.
* Многосерверный phpMyAdmin позволяет управлять несколькими серверами одновременно.
* Форматы резервного копирования phpMyAdmin позволяет создавать резервные копии баз данных в различных форматах, таких как XML, CSV, SQL, PDF, OpenDocument Text, Excel, Word, электронная таблица и т.д.
* Простой в использовании интерфейс phpMyAdmin позволяет легко создавать и запускать сложные запросы, создавать и редактировать функции, триггеры и т. Д.

В качестве среды разработки выбран текстовый редактор Atom, за счет быстрого написания кода с умным авто дополнением, удобным разделением интерфейса редактирования для одновременной работы с несколькими файлами или проектами, возможности установки расширений с помощью встроенного менеджера пакетов, а также за встроенную интеграцию с Git и GitHub.

Таким образом разработка frontend части будет производится на языках HTML, CSS, JS при помощи GULP. А backend – на языке программирования PHP, СУБД – MySQL с использованием программы PhpMyAdmin. Среда разработки –редактор кода Atom.

1.2 Разработка системы требований

Для составления системы требованием необходимо четко определить проблему для дальнейшего ее решения. Структурирование проблемы приведено в таблице 1

1. Структурирование проблемы

|  |  |
| --- | --- |
| Элемент | Описание |
| Проблема | Увеличение потока клиентов и доходов Веб-студии |
| Воздействует на что (кого) и  результатом чего является | Проблема оказывает влияние на Клиентов и Сотрудников.  От качества реализации зависит репутация компании, впечатление Заказчиков от качества выполнения работы и, следовательно, доходы компании. |
| Выигрыш от решения может  состоять в следующем | Предлагаемое решение: Создание web-сайта, на котором потенциальный Клиент может ознакомится с возможностью компании, а также после заключения контракта Заказчик сможет наблюдать за этапами выполнения заказа.  Преимущества от данного решения:  - повышения доверия со стороны заказчика;  - повышение репутации компании;  - удобство просмотра процесса выполнения заказа, без обращения к компании напрямую. |

Таким образом необходимо разработать современный, красивый и практичный сайт для того, чтобы он мог заинтересовать потенциальных клиентов.

Определение корневых причин, лежащих в основе проблемы:

* У компании, занимающейся созданием сайтов, обязан быть собственный сайт, который будет выступать ее лицом и примером качества;
* Рекламная компания проводится наиболее эффективно и выгодно за счет маркетинга сайта;
* Клиенты, заключившие контракт, должны иметь возможность в любой момент узнать, как продвигается их проект;
* Обращение в поддержку занимает слишком много времени;
* Частые звонки по телефону для уточнения информации подходят не всем клиентам, сотрудники могут заниматься более продуктивной работой, нежели периодически отвечать на звонки.

Экономический покупатель системы – Руководитель Web-студии «Webex»

Пользователями системы являются Клиенты и Разработчики компании. Помимо них, система может оказать косвенное влияние на потенциальных клиентов компании, повышая репутацию web-студии.

После того, как система будет разработана, представлена и развернута, проводится тестирование от сотрудников компании, после которого система оценивается. Других внешних или внутренних пользователей системы, чьи потребности следует учесть нет.

После проведения тестирования и сдачи системы, ее сопровождением займутся сотрудники компании-заказчика. Управление системой передается администратору. Система имеет связь с Базой Данных, откуда и получает информацию.

Ограничения для системы:

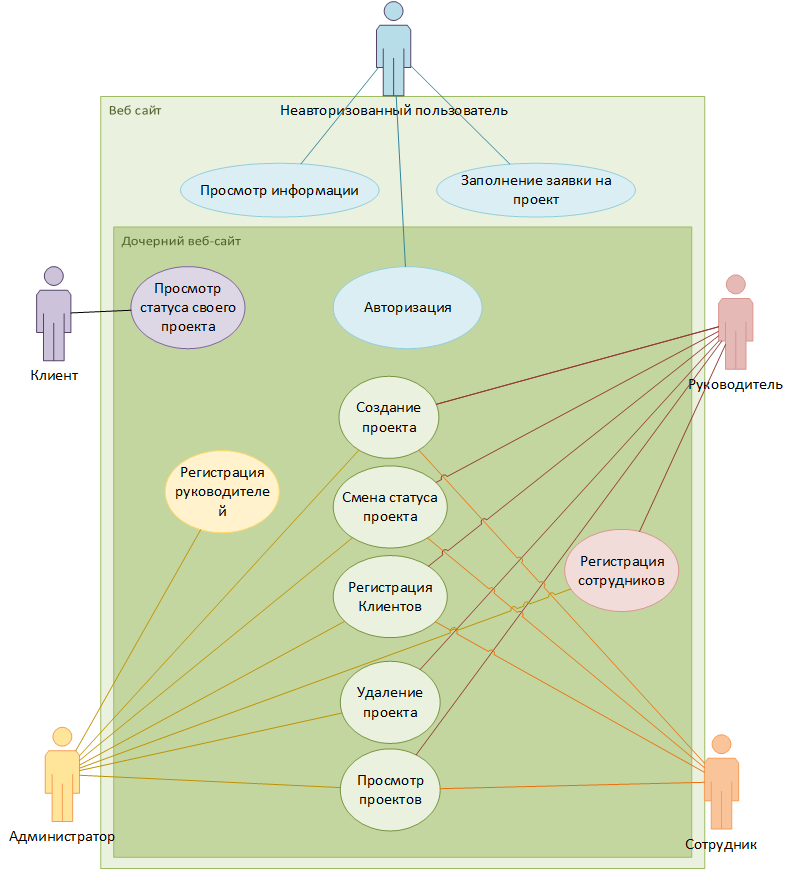
* обеспечение непрерывной работы сайта;
* соответствуя требованиям безопасности, необходимо шифровать пароли в Базе Данных;
* разработка продукта в рамках законодательства о «Защите персональных данных» (Федеральный закон "О персональных данных" от 27.07.2006 N 152-ФЗ)

1.3 Разработка функциональной спецификации

Диаграмма прецедентов — это наиболее общее представление функционального назначения системы. На ней применяются два типа основных сущностей: варианты использования и действующие лица, между которыми устанавливаются следующие основные типы отношений:

* ассоциация между действующим лицом и вариантом использования;
* обобщение между действующими лицами;
* обобщение между вариантами использования;
* зависимости (различных типов) между вариантами использования.

Диаграмма прецедентов представлена на рисунке 1

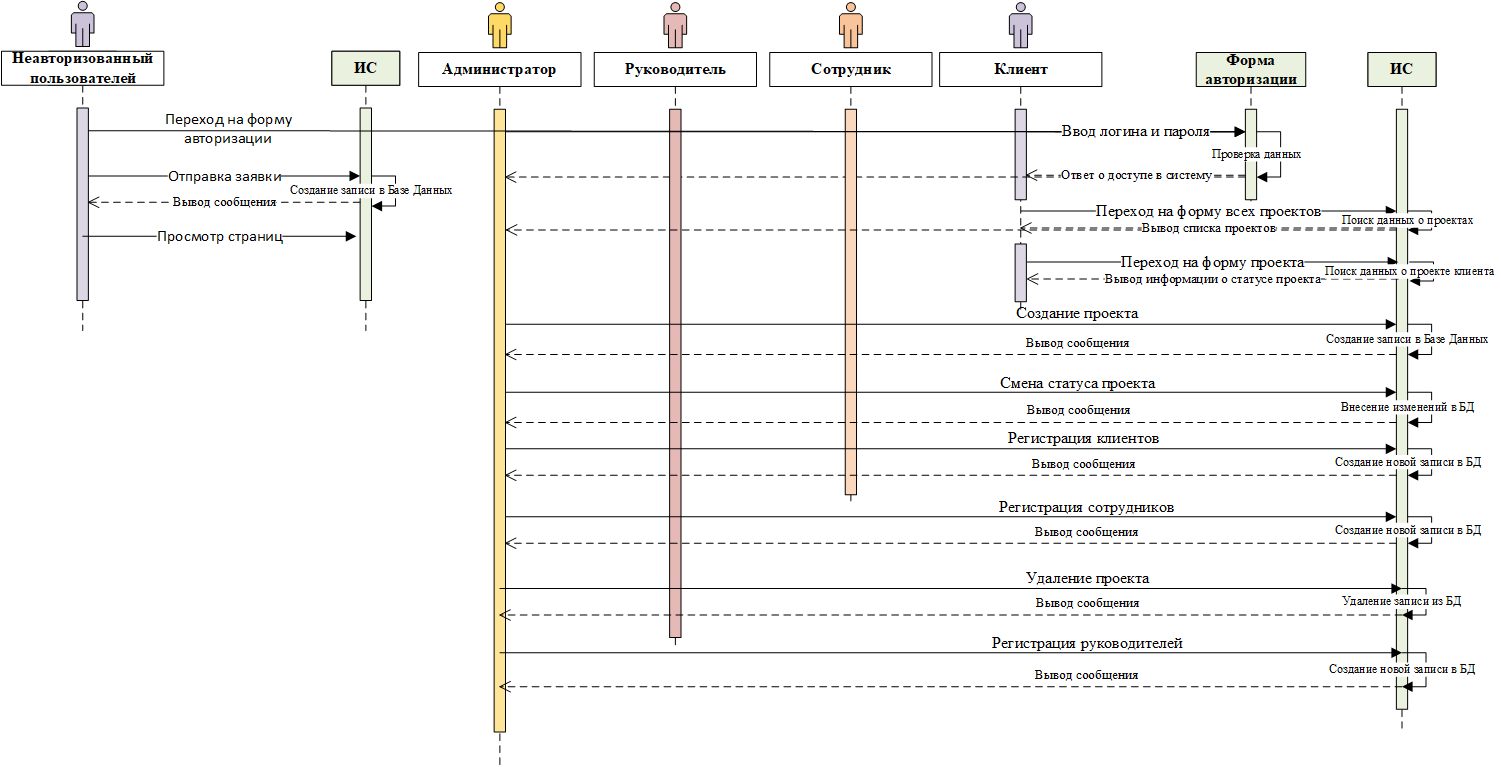


1. Дигамма прецедентов

Первый актер – Неавторизованный пользователь. Это любой посетитель сайта. Он может просматривать информацию, которая предоставлена на основном сайте, а также может отправить заявку в компанию, после обработки которой ему перезвонят и заключат контракт. После заключения контракта пользователь сможет пройти авторизацию и попасть в Дочерний сайт. Там роль актера будет изменена в зависимости от того, как он был внесен в БД. Следовательно, у него будет одна из ролей: Клиент, Сотрудник, Руководитель или Администратор. Возможности ролей распределены в убывающем порядке, то есть Администратору доступны все функции, а функционал Клиента самый ограниченный.

Диаграмма последовательности — диаграмма, на которой показаны взаимодействия объектов, упорядоченные по времени их проявления. Используется в языке UML. По сути, это запись протокола конкретного сеанса работы системы (или фрагмента такого протокола). В объектно-ориентированном программировании самым существенным во время выполнения является посылка сообщений взаимодействующими объектами. Именно последовательность посылки сообщений отображается на данной диаграмме.

Диаграмма последовательностей системы представлена на рисунке 2



1. Диаграмма последовательностей системы

На диаграмме последовательностей показаны возможности всех ролей. Присвоение другой роли Неавторизованному пользователю происходит после авторизации. Регистрация проводится вышестоящими ролями.

2 Практическая часть

2.1 Разработка CRC-карточек

Одним из способов проектирования программного обеспечения является метод CRC-карточек. Целью использования данного метода является исследование взаимодействия объектов системы, и как результат - более ясное понимание структуры программы.

На этом этапе необходимо выявить классы, которые необходимо будет создать в программе для реализации системы. Для понимания работы проектируемой информационной системы достаточно выделить следующие классы:

1. клиент;
2. сотрудник;
3. администратор;
4. веб-сайт;
5. дочерний веб-сайт.

CRC-карты относительно выделенных классов приведены в таблицах 2-6

1. CRC-карта для класса Пользователь

|  |  |
| --- | --- |
| Клиент | |
| 1. Посещает веб-сайт; 2. Переходит на страницы сайта; 3. Оставляет заявку; 4. Переходит на форму авторизации Дочернего сайта; 5. Вводит логин, пароль и нажимает на кнопку вход; 6. Получает доступ к Дочернему веб-сайту; 7. Переходит на страницу личного кабинета, откуда переходит к своему проекту; 8. Просматривает информацию по статусу выполнения проекта. | Веб-сайт,  Дочерний Веб-сайт |

1. CRC-карта для класса Сотрудник

|  |  |
| --- | --- |
| Сотрудник | |
| 1. Посещает веб-сайт; 2. Переходит на форму авторизации Дочернего сайта; 3. Вводит логин, пароль и нажимает на кнопку вход; 4. Получает доступ к Дочернему веб-сайту; 5. Переходит на страницу личного кабинета, откуда переходит к проектам; 6. Создает проекты; 7. Регистрирует клиентов; 8. Меняет статусы проектов. | Дочерний Веб-сайт,  Клиент |

1. CRC-карта для класса Администратор

|  |  |
| --- | --- |
| Администратор | |
| 1. Посещает веб-сайт; 2. Переходит на форму авторизации Дочернего сайта; 3. Вводит логин, пароль и нажимает на кнопку вход; 4. Получает доступ к Дочернему веб-сайту; 5. Переходит на страницу личного кабинета, откуда переходит к проектам; 6. Создает проекты; 7. Регистрирует клиентов и сотрудников; 8. Меняет статусы проектов. | Дочерний Веб-сайт,  Клиент,  Сотрудник |

1. CRC-карта для класса Веб-сайт

|  |  |
| --- | --- |
| Веб-сайт | |
| 1. Отображает основную информацию; 2. Ссылается к Дочернему Веб-сайту; 3. Соединяется с БД; 4. Отправляет заявки на новый проект от пользователей. | Дочерний Веб-сайт,  Клиент,  Сотрудник,  Администратор |

1. CRC-карта для класса Дочерний Веб-сайт

|  |  |
| --- | --- |
| Дочерний Веб-сайт | |
| 1. Отображает основные страницы; 2. Ссылается к основному Веб-сайту; 3. Соединяется с БД; 4. Запрашивает списки исходных данных из таблиц БД (если такая возможность имеется для таблицы); 5. Отображает различные данные. | Веб-сайт,  Пользователь,  Сотрудник,  Администратор |

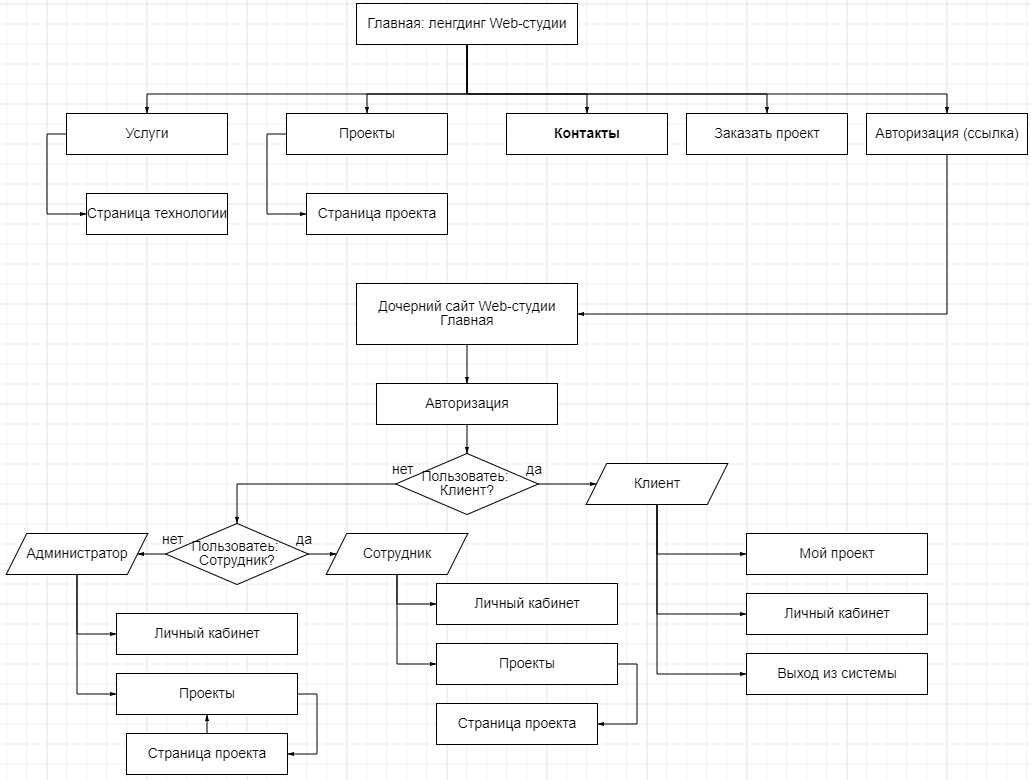
Таким образом проект разделен на Основной веб-сайт и Дочерний, основные действия по работе с БД происходят на Дочернем сайте и классы Сотрудник и Администратор имеет к нему прямое отношение.

2.2 Структурная и функциональная схемы

Структурная схема отражает состав и взаимодействие по управлению частей разрабатываемого программного обеспечения, определяется архитектурой разрабатываемого ПО. Разработка структурной схемы программы выполняется методом пошаговой детализации.

Структурные схемы пакетов программ разрабатывают для каждой программы пакета по отдельности, поскольку организация программ в пакеты не предусматривает передачи управления между ними. Компонентами структурной схемы программной системы или программного комплекса могут служить программы, подсистемы, базы данных, библиотеки ресурсов и т. п.

Структурная схема сайта «Web-студия WEBEX» приведена на рисунке 3

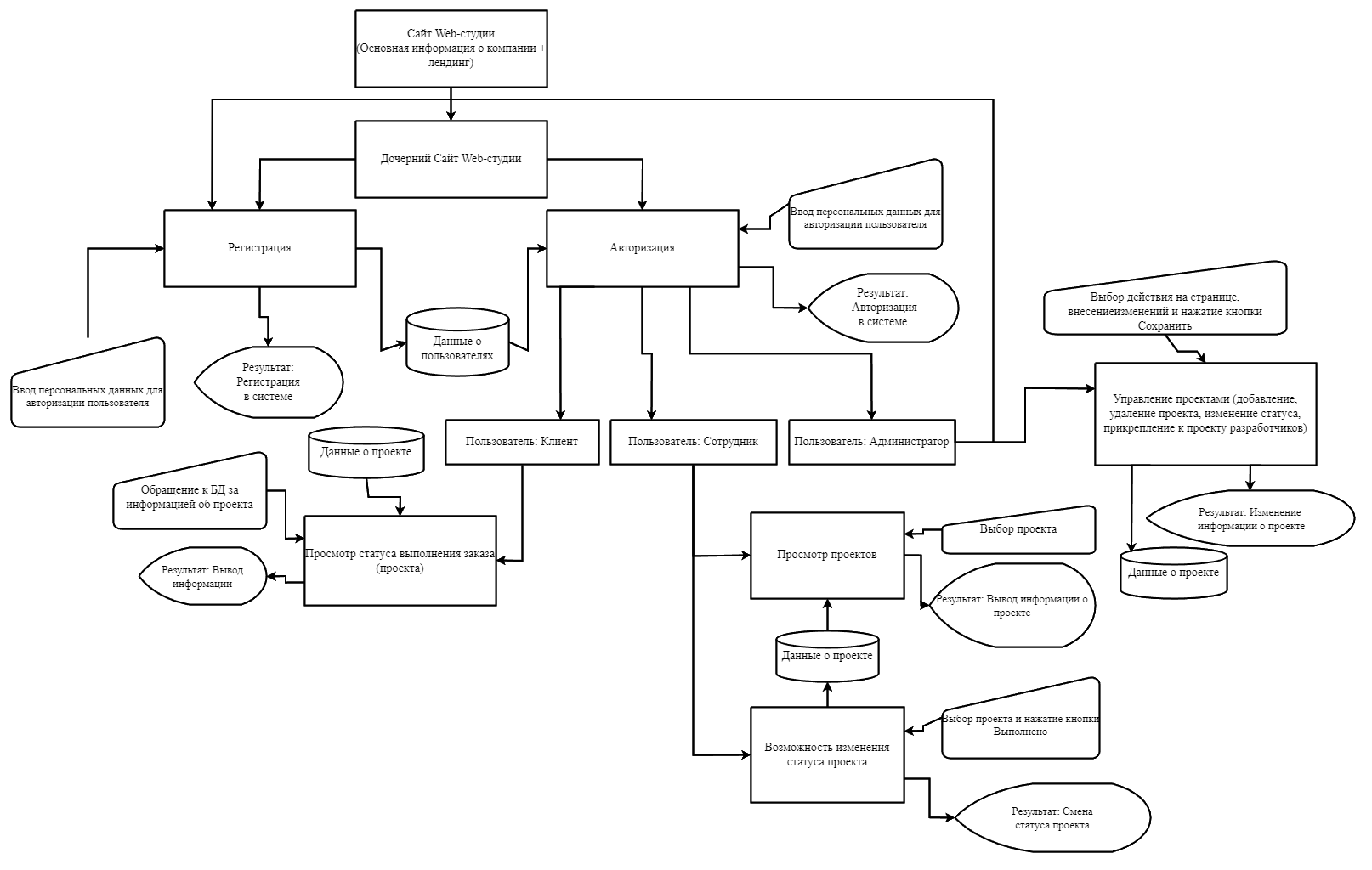


1. Структурная схема

На структурной схеме видно, что работа сайта разделена на две части: Основной сайт, где находится вся ознакомительная информация для потенциальных клиентов, включая в себя портфолио проектов разработчиков. Он выступает лицом компании. Дочерний сайт является дополнением к основному сайту, но из-за большей работы с БД и в целях безопасности и защиты данных, он выводится в отдельный домен. Целью работы дочернего сайта является ведение учета выполнения заказов и отображение этой информации клиентам.

Функциональная схема (ГОСТ 19.701—90) — это схема взаимодействия компонентов программного обеспечения с описанием информационных потоков, состава данных в потоках и указанием используемых файлов и устройств. Функциональные схемы более информативны, чем структурные.

Функциональная схема сайта Web-студии приведена на рисунке 4



1. Функциональная схема

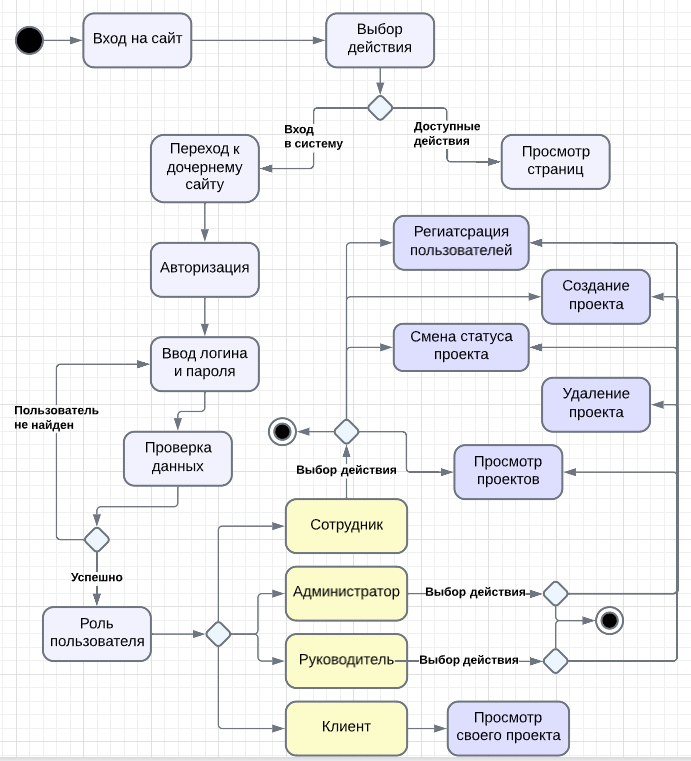
Из схемы видно, каким образом функции взаимодействуют с базой данных. Возможности, приведенные на картинке, относятся к Дочернему сайту.

2.3 Разработка диаграммы деятельности

Для моделирования процесса выполнения операций в языке UML используются так называемые диаграммы деятельности. На этапе анализа требований и уточнения спецификаций диаграммы деятельностей позволяют конкретизировать основные функции разрабатываемого программного обеспечения.

Под деятельностью в данном случае понимают задачу (операцию), которую необходимо выполнить вручную или с помощью средств автоматизации. В теоретическом плане диаграммы деятельности являются обобщенным представлением алгоритма, реализующего анализируемый вариант использования.

Диаграмма деятельности представлена на рисунке 5



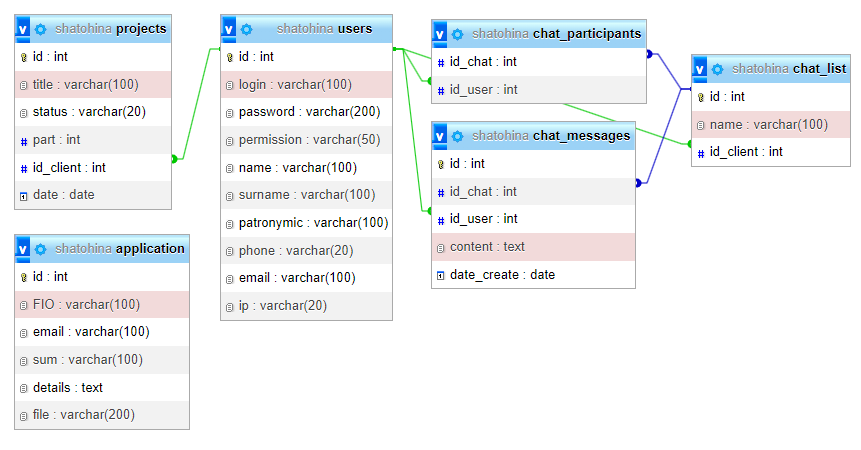
1. Диаграмма деятельности ИС «web-студия WEBEX»

Главное предназначение диаграммы состояний — описать возможные последовательности состояний и переходов, которые в совокупности характеризуют поведение элемента модели в течение его жизненного цикла.

2.4 Разработка ER-диаграммы

ER-диаграмма — это разновидность блок-схемы, где показано, как разные «сущности» связаны между собой внутри системы. ER-диаграммы чаще всего применяются для проектирования и отладки реляционных баз данных в сфере образования, исследования и разработки программного обеспечения и информационных систем для бизнеса. ER-диаграммы полагаются на стандартный набор символов, включая прямоугольники, ромбы, овалы и соединительные линии, для отображения сущностей, их атрибутов и связей. Эти диаграммы устроены по тому же принципу, что и грамматические структуры: сущности выполняют роль существительных, а связи — глаголов.

На рисунке 6 представлена ER-диаграмма базы данных информационной системы «Web-студия WEBEX»



1. ER-диаграмма БД ИС «WEBEX»

На ER-диаграмме представлены основные таблицы ИС. Таблица application содержит в себе необработанные заявки от пользователей, это единственная таблица с которой связывается Основной Веб-сайт. Таблицы projects и users непосредственно взаимодействует с Дочерним сайтом. Таблицы chat\_messages, chat\_participants и chat\_list (совместно с projects и users) относятся к мобильному приложению, которое разрабатывается в рамках другой работы, но в связи с целостностью и связью данных между системами, они обладают единой Базой Данных.

2.5 Разработка словаря данных

Для каждого выходного документа был создан локальный СД. В результате проведения анализа перечня данных из входных документов и перечня данных из СД – определился общий СД, который содержит определение для всех данных из СД источников данных.

Словарь данных представлен на таблицах 7-12:

1. «Проекты»

|  |  |  |  |
| --- | --- | --- | --- |
| **Ключ** | **Поле** | **Обязательное** | **Примечание** |
| **projects** | | | |
| Первичный | id | Да | Номер проекта |
|  | title | Да | Наименование проекта |
|  | status | Да | Статус выполнения |
|  | part | Да | Этап выполнения |
| Внешний | id\_client | Да | Внешний ключ к таблице «Пользователи» |
|  | date | Да | Дата создания |

1. «Пользователи»

|  |  |  |  |
| --- | --- | --- | --- |
| **Ключ** | **Поле** | **Обязательное** | **Примечание** |
| **users** | | | |
| Первичный | id | Да | Номер пользователя |
|  | login | Да | Логин |
|  | password | Да | Пароль |
|  | permission | Да | Права доступа |
|  | name | Да | Имя |
|  | surname | Да | Фамилия |
|  | patronymic | Нет | Отчество |
|  | phone | Да | Номер телефона |
|  | email | Да | Почта |
|  | ip | Да | Ip-адрес |

1. «Список чатов»

|  |  |  |  |
| --- | --- | --- | --- |
| **Ключ** | **Поле** | **Обязательное** | **Примечание** |
| **chat\_list** | | | |
| Первичный | id | Да | Номер чата |
|  | name | Да | Наименование чата |
| Внешний | id\_client | Да | Внешний ключ к таблице «Пользователи» |

1. «Участники чата»

|  |  |  |  |
| --- | --- | --- | --- |
| **Ключ** | **Поле** | **Обязательное** | **Примечание** |
| **chat\_participants** | | | |
| Внешний | id\_chat | Да | Внешний ключ к таблице «Список чатов» |
| Внешний | id\_user | Да | Внешний ключ к таблице «Пользователи» |

1. «Сообщения»

|  |  |  |  |
| --- | --- | --- | --- |
| **Ключ** | **Поле** | **Обязательное** | **Примечание** |
| **chat\_messages** | | | |
| Первичный | id | Да | Номер сообщения |
| Внешний | id\_chat | Да | Внешний ключ к таблице «Список чатов» |
| Внешний | id\_user | Да | Внешний ключ к таблице «Пользователи» |
|  | content | Да | Текст собщения |
|  | date\_create | Да | Дата создания |

1. «Заявки»

|  |  |  |  |
| --- | --- | --- | --- |
| **Ключ** | **Поле** | **Обязательное** | **Примечание** |
| **chat\_messages** | | | |
| Первичный | id | Да | Номер сообщения |
| Внешний | id\_chat | Да | Внешний ключ к таблице «Список чатов» |
| Внешний | id\_user | Да | Внешний ключ к таблице «Пользователи» |
|  | content | Да | Текст сообщения |
|  | date\_create | Да | Дата создания |

Основной веб-сайт взаимодействует только с таблицей application, куда попадают необработанные заявки от пользователей. После того, как компания свяжется с клиентом и примет работу над проектом, их сотрудник, авторизовавшись на Дочернем сайте зарегистрирует клиента и привяжет к нему его проект, после чего сообщит данные авторизации клиенту. После авторизации клиент сможет наблюдать за процессом выполнения его заказа.

2.6 Тестирование программного обеспечения

Заключение

Список литературы

* https://www.bibliofond.ru/view.aspx?id=877816
* https://vc.ru/flood/41366-kakie-specialisty-uchastvuyut-v-sozdanii-i-prodvizhenii-sayta
* https://vuzlit.com/385056/znachimost\_sayta