**СОДЕРЖАНИЕ**

|  |  |  |
| --- | --- | --- |
|  | ВВЕДЕНИЕ | 4 |
| 1 | Анализ предметной области | 6 |
|  | 1.1 Сбор данных о предметной области | 7 |
|  | 1.2 Анализ готовых решений | 9 |
|  | 1.3 Определение задачи | 13 |
| 2 | Разработка алгоритмов решение задачи | 14 |
|  | 2.1 Контекстная диаграмма | 15 |
|  | 2.2 Иерархия роль-исполнитель | 18 |
|  | 2.3 Диаграмма вариантов использования | 19 |
|  | 2.4 Диаграмма прецедентов | 21 |
|  | 2.5 Концептуальная модель базы данных | 23 |
|  | 2.6 Физическая модель базы данных | 24 |
|  | 2.7 Обоснования выбора инструментария | 25 |
|  | 2.8 Алгоритм защиты данных | 32 |
|  | 2.9 Алгоритм авторизации | 33 |
|  | 2.10 Алгоритм регистрации | 36 |
|  | 2.11 Блок схемы | 37 |
| 3 | Разработка интерфейса | 41 |
|  | 3.1 Проектирование интерфейса | 41 |
|  | 3.2 Дизайн проекта | 43 |
| 4 | Программная реализация | 48 |
|  | 4.1 Описание разработанных функций | 48 |

|  |  |  |
| --- | --- | --- |
|  | 4.2 Разработка системы тестирования знаний | 51 |
| 5 | Тестирование | 54 |
|  | 5.1 Функциональное тестирование | 54 |
|  | 5.2 Тестирования удобства пользования (юзабилити) | 56 |
|  | 5.3 Тестирование производительности | 59 |
|  | * 1. Тестирование интерфейса пользователя (UI testing) | 60 |
|  | 5.5 Тестирование безопасности. | 60 |
| 6 | Экономические расчёты |  |
|  | 6.1 Исходные данные для расчета затрат |  |
|  | 6.2 Расчет затрат на проектирование системы |  |
|  | 6.3 Расчет затрат на внедрение |  |
|  | 6.4 Расчет эксплуатационных расходов |  |
|  | 6.5 Расчет и построение диаграммы общей стоимости системы |  |
|  | 6.6 Расчет прибыли разработчика системы |  |
|  | 6.7 Расчет затрат, связанных с покупкой, внедрением и использованием системы и срока окупаемости |  |
|  | 6.8 Выводы |  |
|  | СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ | 62 |

**ВВЕДЕНИЕ**

Оценка качества обучения на сегодняшний день относится к одной из самых актуальных проблем в сфере образования. С одной стороны, требуются большие временные затраты, с другой стороны, необходимость анализа полученных данных и сокращение этапа реагирования на полученные результаты. Необходимо заранее оговорить пределы, при превышении которых следует принимать управленческие меры по изменению качества образования. Многие из тех, кто оценивает качество образования, проводят эту длительную процедуру только для получения самого результата, но не осуществляют никакой обратной связи. Поэтому очень важно рассчитать затраты на проведение оценки качества образования и соотнести их с конечным результатом, который хочет получить в итоге образовательное учреждение.

Проблема отечественной педагогической диагностики активно развивается уже с 80-90-х годов. Одной из линий синтеза данной области является связь с математическими методами и информационными технологиями.

В ведущих странах мира компьютерное тестирование пользуется заслуженным доверием и имеет широкую область применения в таких сферах как международные сравнительные исследования, мониторинг качества образования в масштабах страны, лицензирование и государственная аккредитация учебных заведений, аттестация учащихся и студентов, учителей и преподавателей, проверка профессиональной пригодности специалистов и т.д.

В последние годы в сфере образования наблюдается усиление интереса к автоматизации различных видов учебной и административной деятельности. В процессе обучения, прежде всего это коснулось информатизации контроля результатов обучения. Самым популярным видом такого контроля является тестирование, основанное на диалоге информационной системы и пользователя. Рост быстродействия технических средств, уменьшение цен на вычислительную технику, появление качественных и мощных систем программирования увеличило потребность в системах, позволяющих объективно, быстро и надежно оценивать знания учащихся, предлагая интересные формы взаимодействия с ними.

Проведение контроля достижения результатов обучения с использованием современных средств информационных технологий в процессе обучения по сравнению с другими методами контроля имеет ряд очевидных преимуществ, в числе которых: высокая степень стандартизации, объективность оценки результатов, удобная количественная форма выражения результатов, повышенная устойчивость к фальсификациям, высокая скорость обработки результатов, единство требований ко всем учащимся, исключение субъективизма при оценке результатов.

При этом удобство в наличии количественных показателей выражается в возможности сравнения знаний и умений одних обучаемых с другими, или отслеживании динамики усвоения знаний одним учащимся в процессе обучения.

1. Анализ предметной области

Каждый преподаватель рано или поздно сталкивается с вопросом проверки знаний обучаемых. Значительно упростить процесс тестирования знаний помогут онлайн системы тестирования.

Существует множество способов провести проверку уровня знаний учащихся, каждый из которых имеет свои достоинства и недостатки. Одним из них является тестирование. Тестирование знаний обладает следующими преимуществами:

Процесс проведения и проверки результатов стандартизирован, поэтому тестирование является объективным способом оценки знаний.

Тестирование является эффективным с экономической точки зрения.

Благодаря использованию шкал оценок, тесты являются точным инструментом оценивания результатов.

Основной сложностью при организации процесса проведения тестирований является выбор подходящего инструмента. Век использования бумажных тестов давно прошел и на смену ему пришли информационные технологии. В настоящее время широко применяются компьютерные системы тестирования, которые значительно упрощают процесс проведения и проверки результатов тестирований. Но вместе с этим приходят новые требования к квалификации сотрудников. Сложности могут возникнуть при первоначальной установке и настройке системы. Не каждое учебное заведение в состоянии содержать штат специалистов, которые бы отвечали за обслуживание программного обеспечения.

Следующим этапом развития тестов стали онлайн системы тестирования. Они не требуют установки дополнительных программ и работать с ними можно через браузер, достаточно только подключения к сети интернет. Применение онлайн тестирования - самый простой и дешевый способ организовать проверку знаний. Современные системы позволяют организовать процесс тестирования начиная от этапа составления тестовых заданий и заканчивая проверкой и анализом результатов. Важно отметить, что качественный инструмент может значительно упростить вашу задачу, но если ошибиться и выбрать не подходящую средство, то все плюсы онлайн систем тестирования сведутся на нет. Например, одним из затратных этапов является создание тестов, и хорошая система тестирования должна обладать удобным и простым инструментом для составления тестовых заданий.

Интернет тестирование - современный способ оценки знаний учащихся, который подкупает своей простотой и доступностью. Его по праву можно назвать инновацией в области образования. Если вы желаете улучшить процесс обучения в вашей организации, то онлайн системы тестирования - это отличный первый шаг на пути к модернизации вашего инструментария.

* 1. **Сбор данных о предметной области**

Тестирование – форма контроля знаний учащихся.

Важным звеном процесса обучения является контроль знаний и умений обучающихся. Постепенный переход от традиционных форм контроля и оценивания знаний к тестированию отвечает духу времени и общей концепции модернизации системы образования.

Тестирование в педагогике выполняет три основные взаимосвязанные функции: диагностическую, обучающую и воспитательную:

1. Диагностическая функция заключается в выявлении уровня знаний, умений, навыков учащегося. Это основная функция тестирования.

2. Обучающая функция тестирования состоит в мотивировании учащегося к активизации работы по усвоению учебного материала.

3. Воспитательнаяфункция проявляется в периодичности и неизбежности тестового контроля. Это дисциплинирует, организует и направляет деятельность учащихся, помогает выявить и устранить пробелы в знаниях, формирует стремление развить свои способности.

В практической деятельности используют разные виды тестовых заданий:

1. Задания с выбором ответов:

а) задания с выбором одного правильного ответа

б) задания с выбором нескольких правильных ответов

в) задания с выбором одного неправильного ответа.

2. Задания на установление соответствия.

3. Задания с открытым ответом.

Основными достоинствами данной формы контроля знаний является:

- возможность детальной проверки усвоения учащимися каждой темы;

- осуществление диагностики уровня усвоения учебного материала каждым учеником;

- обеспечивает одновременную проверку знаний учащихся всего класса и формирует у них мотивацию для подготовки к каждому уроку;

- правильно оформленный тест повышает интерес к предмету.

Применяемые в учебной деятельности письменные тесты с успехом интегрирую в компьютерные тесты. Это позволяет мне более рационально использовать время урока, охватить больший объем содержания, быстро установить обратную связь с учащимися и определить результаты усвоения материала. Отвечая на вопросы тестов, учащиеся сами могут оценивать собственные знания по предмету.

Опыт применения тестов показывает эффективность этой формы контроля. Тесты, разработанные с учетом последовательности изучения тем, помогают выявить пробелы в знаниях, определить насколько осознанно учащиеся владеют теоретическим материалом. Это позволяет мне внести необходимые коррективы в дальнейшую работу с тем или иным студентом. Использование такого вида работы помогает осуществить опрос учащихся по разным проблемам, превратить урок в увлекательное занятие.

Предметная область разрабатываемого сайта охватывает вопросы связанные с развитием и оцениванием знаний учащихся. Пользователи подобных сайтов имеют возможность просматривать и читать лекции, информацию о самом сайте, проходить тести по разным предметам и отраслям, пользоваться функцией поиска по сайту, связываться с администратором сайта, регистрироваться, изменять свои персональные данные, а также просматривать свои достижения. Для тех кто купил премиум на любой срок открываются все тесты.

Потенциальными пользователями ресурса могут быть самые разные люди –

учителя и воспитатели, студенты, ученики, и просто люди, которые интересуются или изучают программирование.

К теме тестирования относится очень широкий круг предметов и дисциплин, по которым можно прочитать много уроков или лекций:

- иностранные языки;

- математика;

- алгебра;

- геометрия;

- география;

- химия;

- физика;

- история;

- информатика;

- литература;

- основы безопасности жизнедеятельности;

И это были перечислены только школьные предметы, а ведь большая часть подобных сайтов направлена на профессиональное тестирование.

* 1. **Определение задачи**

Создание модели системы тестирования на базе web-технологий является крайне перспективной и актуальной задачей. Целью данного проекта является создание системы компьютерного тестирования знаний с помощью средств web-разработки и создание программы для работы тестовой системы. В конкретном случае предполагается реализация защиты от копирования информации, списывания при контроле знаний, реализация системы решения задач.

Перед созданием модели подобной программы тестирования нужно определить набор функций и требований. Перечислим основные функции, требования, характеристики системы тестирования:

* реализация эффективного тестирования знаний;
* точная и способная к адаптации оценка результата тестирования;
* возможность создавать и изменять тесты;
* возможность одновременного тестирования неограниченного количества пользователей;
* большое количество тестов, вопросов, заданий;
* контроль времени тестирования.

Стоит так же упомянуть требования к тестовым заданием:

* содержание тестового задания должно требовать от испытуемого определенного ответа на теоретические вопросы;
* тестовые задания не должны иметь двусмысленных вопросов или спорных ответов;
* следует избегать тестовых заданий требующих развернутых ответов;
* тестовые задание должны быть ориентированы только на пройденные учебные материалы;

Существует множество различных систем интернет-тестирования, обладающих как достоинствами, так и недостатками. Основная проблема заключается в том, что нет универсальной системы интернет–тестирования, избавленной от основных недостатков и обладающая достоинствами своих аналогов. Если объединить достоинства конкурирующих систем в один программный продукт, то получится очень гибкая система, позволяющая решать огромный набор задач.

В ходе разработки данного программного средства были выявлены следующие особенности системы интернет тестирования, которые необходимо учесть в данном продукте:

1. Реализация возможности возврата к пропущенным вопросам. Зачастую пользователи пропускают сложные вопросы и в первую очередь отвечают на простые, однако, вернуться к пропущенным вопросам уже не могут.

2. Реализация гибкой системы категорирования тестов. Когда пользователь заходит на сайт интернет-тестирования, то зачастую теряется при выборе интересующего его раздела. Категорирование по дисциплинам, специальностям и группам может значительно сократить время поиска нужного теста.

3. Разным преподавателям необходимы разные настройки для тестирования. К примеру, разрешить пользователям продолжить тестирование, если они его закрыли, случайно или намеренно. Разрешить пользователям пропускать вопросы, если они сложные, и разрешить возвращаться к пропущенным вопросам. Также запретить переходить к следующему вопросу пока не будет дан правильный ответ на текущий вопрос (для пробных тестов). Чтобы исключить появление однотипных тестов необходимо реализовать сортировку вопросов в случайном порядке или же, наоборот, отключить сортировку вопросов в случайном порядке, если необходим одинаковый порядок вопросов в тесте.

4. Особое внимание нужно уделить возможности использования одного и того же вопроса в разных тестах. Чтобы преподаватель не тратил время на создание аналогично вопроса для другого теста, если темы для разных дисциплин пересекаются.

В настоящее время выработано четыре основных формы тестовых заданий, которые являются основой для составления тестов по любым учебным дисциплинам:

1. Задания закрытой формы. Инструкция: обвести кружком, либо отметить крестиком, галочкой нужный вариант ответа. Вариативность заданий закрытой формы весьма велика, но в основе всегда лежит один и тот же принцип: испытуемому предлагается выбрать ответ на задание из нескольких предложенных, причем только один из них является правильным.

2. Задания открытой формы. Инструкция: дополнить. В отличие от заданий закрытой формы здесь не предлагается вариантов ответа, а делается пропуск смысловой единицы в каком-либо утверждении, причем предполагается, что заполнить этот пропуск можно строго однозначно.

3. Задания на соответствие. Инструкция: установить соответствие. Здесь необходимо установить соответствие между смысловыми единицами в правом и левом столбиках, причем, справа иногда вариантов дается больше, чем слева, то есть заведомо предполагается, что какие-то из них являются в данном случае неправильными.

4. Задания на установление правильной последовательности. Инструкция: установить правильную последовательность. Достаточно сложная форма, проверяющая глубокие и прочные знания учебного материала. Смысл ее состоит в установлении последовательности каких-либо событий, действий, терминов и т. д. В пустых квадратиках надо расставить цифры.

Существуют различные разновидности и модификации тестовых заданий, однако все они основываются на этих четырех формах.

Стоит отметить положительные стороны теста на конкретном примере: если при оценке результатов за каждое правильно выполненное задание ставится единица, а в противном случае ноль, то уровень знаний по проблематике данного теста будет выражен в определенной сумме баллов. Расставив проходивших тест по возрастанию или убыванию, мы получим структурированный ряд уровня знаний по данной дисциплине. Если такая проверка проводится регулярно в течение всего изучения предмета, в конце преподаватель будет иметь достаточно четкое представление об уровне знаний данного курса. Создается своеобразный рейтинг обучаемых по изучаемой дисциплине. По итогам выполнения всех тестовых заданий видны пробелы в знаниях каждого, что позволяет составить индивидуальную программу для дополнительных занятий.

**1.3 Анализ готовых решений**

Во избежание ошибок при разработке был произведён поиск и анализ готовых и подобных решений.

В процессе анализа были выделены следующие веб-ресурсы:

- duoLingo.com;

- сервис онлайн тестирования Quizful (quizful.net).

Сайтduolingo.comпредоставляет бесплатное обучение иностранным языкам.

Достоинства.

На сайте очень красивый и приятный дизайн. Сайт адаптирован под разные платформы. На сайте удобный интерфейс. Проходить тесты могут как зарегистрированные пользователи так и не зарегистрированные. Во время теста были обнаружены специальные возможности: горячие клавиши, голосовое повторение. На сайте присутствует мультиязычность.

Недостатки: на сайте отсутствует поиск. Малокурсов для изучения.

Сервис онлайн тестирования Quizful предоставляет обучение по программированию.

Достоинства: на сайте много лекций, много тестов по разным языкам программирования, а также английскому языку и дизайну.

Недостатки: на сайте средний и не очень удобный дизайн и интерфейса.

1. **Разработка алгоритмов решения задачи**

Наиболее эффективно математическую модель можно реализовать на компьютере в виде алгоритмической модели.

Данный этап включает разработку алгоритмов решения задачи и этап программирования или непосредственной подготовки текста программ.

Разработка алгоритма включает в себя выбор метода проектирования алгоритма, выбор формы записи алгоритма (блок-схемы и др.), выбор тестов и метода тестирования, проектирование самого алгоритма.

Алгоритм – это точное предписание, определяющее вычислительный процесс, ведущий от исходных данных к искомому результату.

Существуют следующие способы описания алгоритма: словесный, структурно-стилизованный, язык графических символов, операторный язык.

При словесном способе алгоритм задается в произвольном изложении на естественном языке. Недостаток этого способа состоит в том, что алгоритм строго не формализуем, многословен, допускает неоднозначности. Однако данный способ изложения алгоритма не требует специальных знаний и может применяться конечными пользователями. Именно на этом языке, как правило, сообщается неформальная постановка задачи на этапе формализации и он же может быть использован для представления результата первого этапа.

Структурно-стилизованный способ - это формализованное представление предписаний, задаваемых путем использования ограниченного набора типовых синтаксических конструкций. Данный способ представления алгоритма требует подготовки и специальных несложных знаний. Примером его может служить нотация Бэкуса – Наура, которая впервые была использована для описания синтаксических конструкций языка Фортран.

Язык графических символов предполагает соотнесение каждому типу действий геометрической фигуры, представленной в виде блочного символа. Действия (блоки) соединяются линиями потока. Совокупность таких связанных блоков называется блок-схемой.

Операторный язык – это такой способ кодирования алгоритма, в результате которого получается понятная для компьютера запись алгоритма - программа. Этот способ требует, как минимум, знания правил записи выражений средствами того или иного языка программирования. Примером такого языка может служить Алгол-60, который долгие годы являлся стандартом для описания алгоритмов в научной литературе.

Программирование включает в себя следующие виды работ: выбор языка программирования, уточнение способов организации данных, запись алгоритма на выбранном языке программирования. Если правильно проведен этап разработки алгоритма, то это технический процесс.

Этап решения задачи было бы правильнее назвать компьютерным моделированием, т. к. при решении некоторых задач можно обойтись без составления программы на языке программирования, это можно успешно сделать, используя современные приложения (электронные таблицы, системы управления базами данных и пр.). В этом случае не понадобится и следующий этап - отладка и тестирование программы, а вот проведение расчетов и анализ полученных результатов следует проводить с особой тщательностью.

**2.1 Контекстная диаграмма**

Контекстная диаграмма - вид IDEF0-диаграммы. Это диаграмма, расположенная на вершине древовидной структуры диаграмм, представляющая собой самое общее описание системы и ее взаимодействие с внешней средой (как правило, здесь описывается основное назначение моделируемого объекта). Контекстная диаграмма состоит из одного блока, описывающего функцию верхнего уровня, ее входы, выходы, управления, и механизмы, вместе с формулировками цели модели и точки зрения, с которой строится модель.

Контекстная диаграмма A-0 должна содержать краткие утверждения, определяющие точку зрения должностного лица или подразделения, с позиции которого создается модель, и цель, для достижения которой ее разрабатывают. Эти утверждения помогают руководить разработкой модели и ввести этот процесс в определенные рамки. Точка зрения определяет, что и в каком разрезе можно увидеть в пределах контекста модели. Изменение точки зрения приводит к рассмотрению других аспектов объекта. Аспекты, важные с одной точки зрения, могут не появиться в модели, разрабатываемой с другой точки зрения на тот же самый объект.

Согласно стандарту IDEF0 на каждом уровне декомпозиции должен использоваться принцип ограничения объекта, поэтому в соответствии с этим принципом считается, что единственный блок и несколько стрелок на самом верхнем (контекстном) уровне используются для определения границы всей системы. Соответственно, стрелки, касающиеся этого блока, описывают главные управления, входы, выходы и механизмы этой системы. В дальнейшем, текстовое описание, содержащее основные типы объектов и функции и комментарии экспертов, используется для предварительного создания диаграммы А0.

Стрелки на контекстной диаграмме отображают связи объекта моделирования с окружающей средой. Поскольку единственный блок представляет весь объект, его имя – общее для всего проекта. Это же справедливо и для всех стрелок диаграммы, поскольку они представляют полный комплект внешних интерфейсов объекта.

Контекстная диаграмма имеет узловой номер A-n (n?0), которая представляет контекст модели. Диаграмма верхнего уровня обозначается идентификатором «А-0» (произносится «А минус ноль»), на которой объект моделирования представлен единственным блоком с граничными стрелками, устанавливает область моделирования, определяет границы модели и является обязательной контекстной диаграммой. Диаграммы с узловыми номерами А-1, A-2,... - дополнительные контекстные диаграммы.

В пояснительном тексте к контекстной диаграмме указывается цель (Purpose) построения диаграммы в виде краткого описания и фиксированная точка зрения (Viewpoint). Именно с помощью контекстной диаграммы определяется субъект моделирования, т.е. что будет в дальнейшем рассматриваться как элементы/компоненты системы, а что - как внешняя среда/воздействие. Имя функции, записываемое в блоке контекстной диаграммы, является общей функцией системы с принятой точки зрения и определенной целью построения модели.

Детализируя рассматриваемую систему на этапе сбора и анализа предварительной информации, необходимо обращать внимание на входные и выходные объекты самой системы и составляющих ее подсистем. Моделирование необходимо начинать с составления описания основных типов объектов и основных функций системы. При этом необходимо учесть нормальные и аномальные ситуации, имеющиеся в системе обратные связи, и возможные случаи потенциальных ошибок.

**C:\Users\User\Downloads\Untitled Diagram (6).png**

Рисунок 2.1 – Контекстная диаграмма

**2.2 Иерархия исполнитель-роль**

Классифицируя пользователей и исследуя их нужды можно узнать такие сведения, которые будут использоваться при разработке базы данных, ограничение реализовать и как сгруппировать страницы. На рисунке 2.2.1 предоставлено группу пользователей, которые будут работать с системой тестирования.

**C:\Users\User\Downloads\Untitled Diagram (7).png**

Рисунок 2.2 - Иерархия исполнитель-роль

**2.3 Диаграмма вариантов использования**

Диаграмма вариантов использования – диаграмма, которая изображает отношения между актёрами и вариантами использования. Цель диаграммы использования: определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы.

**C:\Users\User\Downloads\Diagram (3).png**

Рисунок 2.3 – Диаграмма вариантов использования веб-сайта (Зарегистрированный пользователь)

**C:\Users\User\Downloads\Untitled Diagram (14).png**

Рисунок 2.4 – Диаграмма вариантов использования веб-сайта (Администратор)

**2.4 Диаграмма прецедентов**

Диаграмма прецедентов в [UML](https://ru.wikipedia.org/wiki/UML) — диаграмма, отражающая отношения между [актёрами](https://ru.wikipedia.org/wiki/%D0%90%D0%BA%D1%82%D1%91%D1%80_(UML)) и [прецедентами](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D1%86%D0%B5%D0%B4%D0%B5%D0%BD%D1%82_(UML)) и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне.

D:\Загрузки\Untitled Diagram (5).png

Рисунок 2.5 - Диаграмма прецедентов (Гость)

|  |  |  |
| --- | --- | --- |
| Характеристика | | Описание |
| Название | | Авторизация |
| Краткое описание | | Ввод логина и пароля в форму авторизации |
| Потоки событий | Основной | Данные введены правильно, переадресация на страницу пользователя |
| Альтернативный | Данные введены не верно, вывод ошибки |
| Название | | Регистрация |
| Краткое описание | | Ввод логина, пароля и почты для регистрации на сайте |
| Потоки событий | Основной | Данные введены правельно, переадресация на страницу пользователя |
| Альтернативный | Логин или почта совпадают, вывод ошибкии, повторная регистрация |

C:\Users\User\Downloads\Untitled Diagram (17).png

Рисунок 2.6 - Диаграмма прецедентов (Пользователь)

|  |  |  |
| --- | --- | --- |
| Характеристика | | Описание |
| Название | | Тесты |
| Краткое описание | | Выбор тестов,выбор сложности,прохождение тестов и просмотр результата тестов |
| Потоки событий | Основной | Выбор теста, выбор сложности, прохождение и просмотр результата тестов |
| Альтернативный | - |
| Название | | Лекции |
| Краткое описание | | Возможность просмотра и чтения лекций |
| Потоки событий | Основной | Выбор лекции, просмотр и чтение лекции |
| Альтернативный | - |
| Название | | Достижения |
| Краткое описание | | Возможность просмотра достижений |
| Потоки событий | Основной | Просмотр всех достижений и достижений пользователя |
| Альтернативный | - |

**C:\Users\User\Downloads\Untitled Diagram (16).png**

Рисунок 2.7 - Диаграмма прецедентов (Администратор)

|  |  |  |
| --- | --- | --- |
| Характеристика | | Описание |
| Название | | Тесты |
| Краткое описание | | Возможность добавления, редактирования, удаления тестов |
| Потоки событий | Основной | Добавление, редактирование, удаление тестов |
| Альтернативный | - |
| Название | | Лекции |
| Краткое описание | | Возможность добавления, редактирования и удаления лекций |
| Потоки событий | Основной | Добавление, редактирование, удаление |
| Альтернативный | - |
| Название | | Достижения |
| Краткое описание | | Возможность добавления, редактирования, удаления достижений |
| Потоки событий | Основной | Добавление, редактирование, удаление достижений |
| Альтернативный | - |

**2.5 Концептуальная модель базы данных**

Концептуальная модель - это отражение предметной области, для которой разрабатывается база данных. Не вдаваясь в теорию, отметим, что это некая диаграмма с принятыми обозначениями элементов. Так, все объекты, обозначающие вещи, обозначаются в виде прямоугольника. Атрибуты, характеризующие объект - в виде овала, а связи между объектами - ромбами. Мощность связи обозначаются стрелками (в направлении, где мощность равна многим - двойная стрелка, а со стороны, где она равна единице - одинарная).

C:\Users\User\Downloads\Untitled Diagram (18).png

Рисунок 2.8 - Концептуальная база данных

**2.6 Физическая модель базы данных**

Физическая модель базы данных – логическая модель базы данных, выраженная в терминах языка описания данных конкретной СУБД.

Физическая модель базы данных содержит все детали, необходимые конкретной СУБД для создания базы: наименования таблиц и столбцов, типы данных, определения первичных и внешних ключей и т.п.

Физическая модель строится на основе логической с учетом ограничений, накладываемых возможностями выбранной СУБД.

Имена

Имена сущностей и полей (таблиц и столбцов) должны быть записаны латинскими буквами, без пробелов, начиная с буквы.

Типы данных

Для каждого атрибута необходимо определить тип данных его значений.

Связи

Реляционная модель данных требует преобразования связей «многие ко многим» и иерархии наследования и допускает их только на уровне логической модели базы данных.

Ниже представлена физическая модель базы данных

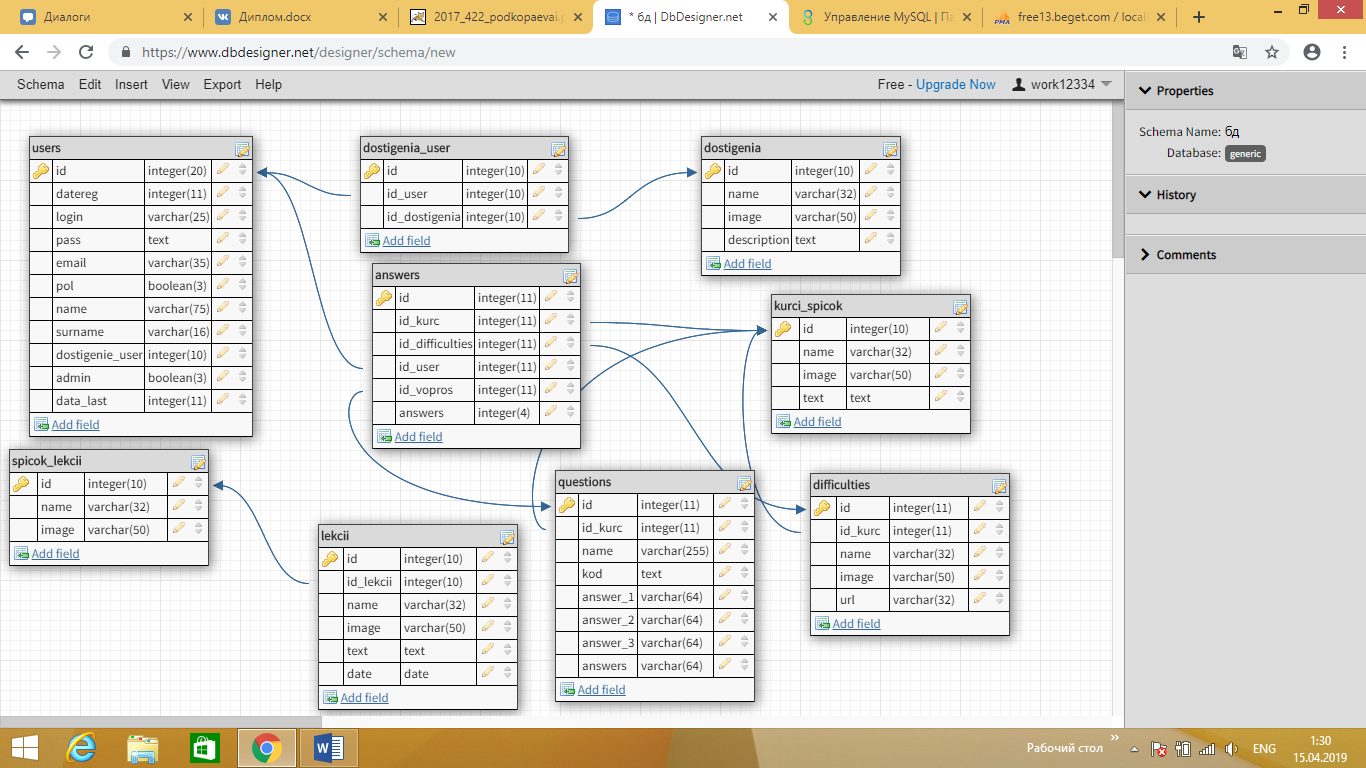


Рисунок 2.9 **-** физическая модель базы данных

**2.7 Обоснование выбора инструментария**

Sublime Text 3 - многофункциональный текстовый редактор с широким набором удобных инструментов для выделения, маркировки и обработки текстовых фрагментов кода.

Интерфейс этого редактора очень лаконичен. Зато скорость работы и отклика на все Ваши действия на достаточно высоком уровне. Поддерживает огромное количество языков (C++, Dylan, Erlang, HTML, Haskell, Java, JavaScript, Lua, Markdown, MATLAB, Perl, PHP, Python, Ruby, SQL, XML и др.) и предлагает на выбор около 20 цветовых схем. Весьма удобно, что реализован полноэкранный режим - очень полезно, если не хотите, чтобы Вас что-то отвлекало от полноценной работы. Кроме того, для Вас приготовлены так называемые мультипанели: можно параллельно работать с несколькими файлами в одном окне, что намного удобнее, чем использовать несколько отдельных окон.

Хочется выделить такую фишку как миникарта - в правой части редактора расположена колонка, в которой в миниатюре расположены примерно 5-6 экранов с текстом. Это позволяет быстро перемещаться по коду и находить нужные элементы.

Кто работает с кодом знает, как важна в текстовом редакторе подсветка. Так вот в приложении есть возможность найти и подсветить ближайшие парные скобки просто поставив курсор внутрь какой-нибудь сложной функции - на самом деле очень удобно и практично.

Что же касается работы непосредственно с текстом, то и тут утилита имеет много преимуществ: поддержка снипетов, автозавершение ввода функций, работа с макросами, удобный поиск, проверка орфографии, мультивыделение, закладки нужных мест, функция автосохранения и другое. Если и этого функционала мало, то его легко расширить за счет плагинов.

Программа очень гибка в настройке, практически все опции редактора настраиваются вручную в текстовых файлах - сразу видно все параметры и описание их назначений. Для часто выполняемых операций доступны комбинации горячих клавиш. В общем, преимушества редактора по достоинству оценят все, кто так или иначе связан с редактирование кода или разметки - программисты, администраторы и т.д.[1]

HTML (HyperTextMarkupLanguage- «язык гипертекстовой разметки») – стандартизированный язык разметки документов во Всемирной паутине[2]. Большинство веб-страниц содержат описание разметки на языке HTML (или XHTML). Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства.

Язык разметки HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986-1991 годах в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. С помощью HTML можно легко создать относительно простой, но красиво оформленный документ.

Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако современное применение HTML очень далеко от его изначальной задачи.

CSS (англ. CascadingStyleSheets — каскадные таблицы стилей) — формальный язык описания внешнего вида документа, написанного с использованием языка разметки.

Преимущественно используется как средство описания, оформления внешнего вида веб-страниц, написанных с помощью языков разметки HTML и XHTML, но может также применяться к любым XML-документам.

CSS используется создателями веб-страниц для задания цветов, шрифтов, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS являлось разделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом. Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печатное представление.

Расширение PHP Data Objects (PDO) определяет легкий, согласованный интерфейс для доступа к базам данных в PHP. Каждый драйвер базы данных, который реализует интерфейс PDO, может предоставлять специфичные для базы данных функции как обычные функции расширения. Обратите внимание, что вы не можете выполнять какие-либо функции базы данных, используя расширение PDO само по себе; для доступа к серверу базы данных необходимо использовать [драйвер PDO](http://php.net/manual/en/pdo.drivers.php) для конкретной базы данных.

PDO обеспечивает уровень абстракции доступа к данным, что означает, что независимо от того, какую базу данных вы используете, вы используете одни и те же функции для выдачи запросов и получения данных. PDO *не* обеспечивает абстракцию *базы данных*; он не переписывает SQL и не эмулирует отсутствующие функции. Вы должны использовать полноценный слой абстракции, если вам нужен этот объект.

PDO поставляется с PHP 5.1 и доступен как расширение PECL для PHP 5.0; PDO требует новых функций OO в ядре PHP 5 и поэтому не будет работать с более ранними версиями PHP.

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. Приложение пользуется большой популярностью у веб-разработчиков, так как позволяет управлять СУБД MySQL без непосредственного ввода SQL команд, предоставляя дружественный интерфейс.

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

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

Проект на данный момент времени локализован на более чем 62 языках.

Инициатором является немецкий разработчик Тобиас Ратшиллер (Tobias Ratschiller) и основывается так же как и phpPgAdmin на аналогичном приложении MySQL-Webadmin, возникшем в 1997 году.

PHP (PHP:HypertextPreprocessor — «PHP: препроцессор гипертекста»; первоначально Personal Home Page Tools — «Инструменты для создания персональных веб-страниц») — скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов.

Язык и его интерпретатор (ZendEngine) разрабатываются группой энтузиастов в рамках проекта с открытым кодом.

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

8 июня 1995 года, вышел PHP/FI 2.0 — первый публичный релиз.

В области веб-программирования, в частности серверной части, PHP — один из популярных сценарных языков (наряду с JSP, Perl и языками, используемыми в ASP.NET).Популярность в области построения веб-сайтов определяется наличием большого набора встроенных средств для разработки веб-приложений. В настоящее время PHP используется сотнями тысяч разработчиков.

JavaScript — [мультипарадигменный](https://ru.wikipedia.org/wiki/%D0%9C%D1%83%D0%BB%D1%8C%D1%82%D0%B8%D0%BF%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) [язык программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F). Поддерживает [объектно-ориентированный](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [императивный](https://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) и [функциональный](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) стили. Является реализацией языка [ECMAScript](https://ru.wikipedia.org/wiki/ECMAScript) (стандарт ECMA-262).

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам [приложений](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0). Наиболее широкое применение находит в [браузерах](https://ru.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) как язык сценариев для придания [интерактивности](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D1%81%D1%82%D1%8C) [веб-страницам](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0).

Основные архитектурные черты: [динамическая типизация](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), [слабая типизация](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D0%B0%D0%B1%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), [автоматическое управление памятью](https://ru.wikipedia.org/wiki/%D0%A1%D0%B1%D0%BE%D1%80%D0%BA%D0%B0_%D0%BC%D1%83%D1%81%D0%BE%D1%80%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [прототипное программирование](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), функции как [объекты первого класса](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82_%D0%BF%D0%B5%D1%80%D0%B2%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B0).

На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на Java, но при этом лёгким для использования [непрограммистами](https://ru.wiktionary.org/wiki/%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C-%D0%BD%D0%B5%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%81%D1%82). Языком JavaScript не владеет какая-либо компания или организация, что отличает его от ряда языков программирования, используемых в веб-разработке.

Название «JavaScript» является зарегистрированным [товарным знаком](https://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D0%B2%D0%B0%D1%80%D0%BD%D1%8B%D0%B9_%D0%B7%D0%BD%D0%B0%D0%BA) компании [Oracle Corporation](https://ru.wikipedia.org/wiki/Oracle_Corporation) в США.

jQuery — библиотека JavaScript, фокусирующаяся на взаимодействии JavaScript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими.

Query – быстрая, легковесная и многофункциональная JavaScript библиотека. С помощью простого и кроссбраузерного API библиотека сильно упрощает перемещение и манипуляции с HTML документами, обработку событий, анимацию, а также работу с технологией Ajax. Благодаря своей гибкости и расширяемости JQuery изменил жизнь миллионов людей, пишущих на JavaScript.

Ajax — подход к построению интерактивных пользовательских интерфейсов веб-приложений, заключающийся в «фоновом» обмене данными браузера с веб-сервером. В результате, при обновлении данных веб-страница не перезагружается полностью, и веб-приложения становятся быстрее и удобнее.

AJAX (Asynchronous JavaScript and XML) - это способ отправки данных на сервер и получения ответа без перезагрузки страницы. Это называется асинхронными запросами. AJAX не является языком программирования, а концепцией обмена данными с сервером.

Аjax (Asynchronous Javascript and XML, Аякс) – это скрипт, который дает возможность обращения к веб-серверу без перезагрузки страницы. Суть этого подхода заключается в фоновом обмене данными между браузером и веб-сервером.

Преимущество технологии:

* Экономия трафика. Часто, вместо загрузки всей страницы достаточно просто загрузить измененную ее часть;
* Уменьшение нагрузки на сервер;
* Ускорение реакции интерфейса. Пользователь видит результат своих действий быстрее, поскольку загружается только измененная часть страницы.

Основным недостатком метода можно считать недоступность загружаемого содержимого для поисковиков.

**2.8 Алгоритм защиты данных**

Безопасность цепи зависит от самого слабого звена: чем оно надежнее, тем цепь прочнее. В хорошей криптосистеме должны быть досконально проверены и алгоритм, и протокол, и ключи, и все остальное. Если криптографический алгоритм достаточно стоек, а генератор случайных чисел, используемый для создания ключей, никуда не годится, любой достаточно опытный крипто аналитик в первую очередь обратит свое внимание именно на него. Если удастся улучшить генератор, но не будут зачищаться ячейки памяти компьютера, после того как в них побывал сгенерированный ключ, грош цена такой безопасности. Если используются стойкий криптографический алгоритм и действительно случайные ключи, которые аккуратно стираются из памяти компьютера после того, как они были использованы, но перед шифрованием файл, в котором вместе с вашим адресом и фамилией указаны все ваши доходы за текущий год, был по ошибке отправлен электронной почтой в налоговую службу, то зачем, спрашивается, вам понадобились и стойкий алгоритм, и случайные ключи, и зачистка компьютерной памяти в придачу?!

В настоящее время практически ни одна современная организация не обходится без использования баз данных в своей деятельности. Базы данных (БД) — это наиболее значимый и ценный актив для любой компании. Поскольку в БД может храниться очень деликатная или конфиденциальная информация, необходимо очень серьезно относиться к ее защите. Любые сбои в работе СУБД и баз данных могут привести к катастрофическим последствиям.

К основным средствам защиты информации относят следующие:

* парольная защита;
* защита полей и записей таблиц БД.
* установление прав доступа к объектам БД;
* шифрование данных и программ;

Защита БД производится на двух уровнях:

* на уровне пароля;
* на уровне пользователя (защита учетных записей пользователей и идентифицированных объектов).

Безопасная система авторизации и регистрации является одним из важнейших элементов при создании проекта. Один из возможных способов — это создание системы регистрации с помощью PHP и MySQL.

Ниже приведена функция защиты данных.

function filter($msg) {

return mysqli\_real\_escape\_string(htmlspecialchars(trim($msg)));

}

Функция mysql\_real\_escape\_string - Экранирует специальные символы в строке, используемой в SQL-запросе, принимая во внимание кодировку соединения.

Функция htmlspecialchars - Преобразует специальные символы в HTML сущности.

Функция trim — Удаляет пробелы (или другие символы) из начала и конца строки.

**2.9 Алгоритм авторизации**

Большинство из актуальных классов атак на Web-приложения приходятся на этап авторизации пользователя, а именно процесс передачи идентификационных и аутентификационных данных от пользователя к базе данных (БД) Web-приложения, минуя брандмауэры Web-приложения и различные системы защиты как программного, так и аппаратного уровня.

Одним из действенных приемов злоумышленника является получение данных авторизации из Cookie, даже если данные будут в захэшированном виде. На этом этапе смысл любой защиты пропадает, но если работать только с сессиями (при закрытии браузера все идентификационные данные обнуляются), то каждый раз при входе на часто посещаемый сайт или Web-приложение придется вводить данные авторизации. Если учесть, что у большинства пользователей несколько почт, разных паролей и логинов, то такие меры по защите приведут к элементарной записи всех паролей и логинов пользователей на материальный носитель, что менее всего защищается и смысл в защите вообще исчерпывается. Есть смысл работать и с Cookie, и с сессиями, грамотно распределив нагрузку. Представим форму, с помощью которой пользователь сможет передать нам свой логин и пароль.



Рисунок 2.10 - Форма передачи пользователем пароля и логина

После ввода пользователем данных необходимо проверить его логин и пароль. Очевидно, данные авторизации не хранятся в чистом виде. Существуют различные как криптографические, так и не криптографические хеш-функции. Криптографические хеш-функции отличаются следующими условиями от остальных хеш-функций: – стойкость к коллизиям 1-го рода – для какого-либо сообщения P должно быть невозможно в реальном времени подобрать другое какое-либо сообщение Q, для которого хеш-функция F(P)=F(Q); – стойкость к коллизиям 2-го рода – должно быть невозможно в реальном времени подобрать такую пару сообщений (P, P'), хеш для которых одинаков; – необратимость – для установленного значения хеш-функции A должно быть невозможно в реальном времени найти блок данных X, хеш-функция для которого F(X)=A.

Для проверки данных авторизации необходимо сравнить хеш введенного пароля с тем, что хранится в БД приложения, как представлено на рисунке 1.8. Процесс хеширования пароля может проводиться с использованием алгоритма, которому отдаст предпочтение разработчик. Не стоит вникать в криптографическую реализацию всех хеш-функций, стоит обратить внимание на функции, к которым не найдены и не скомпрометированы коллизии. Например, алгоритм MD5 не актуален с 2013 года, а алгоритм bcrypt/scrypt остается актуальным. Если обратиться к российскому опыту, то для большинства проектов смена хеша на scrypt оказывается безболезненной. К примеру, многие банки для дистанционного банковского обслуживания все чаще выбирают решения, основанные на отдельном устройстве, которое генерирует хеши уже с заданной скоростью.

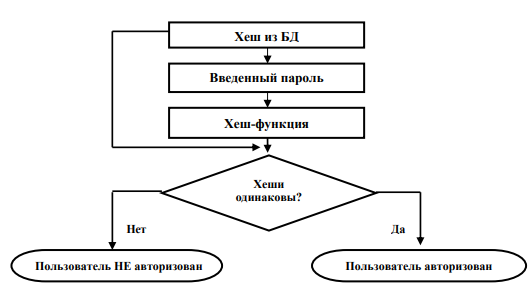


Рисунок 2.11 - Простой алгоритм процесса авторизации пользователя

На рисунке 2.12 представлена процедура сравнения хеша введенного пользователем пароля с хешом пароля, хранящегося в базе данных Web-приложения, используя самый простой алгоритм хеширования md5.

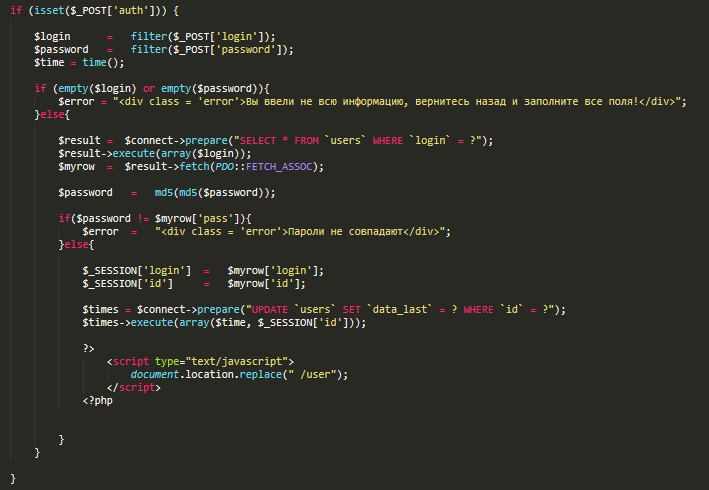


Рисунок 2.12- Процесс сравнения хеша введенного пользователем пароля с хешом, хранящемся в БД Web-приложения.

**2.10 Алгоритм регистрации**

Регистрация — это способ сообщить сайту данные о себе и в обмен получить доступ к дополнительным возможностям (например, добавление чего-либо в избранное) или ресурсам (к примеру, файлам) на сайте, которые недоступны гостям. неразделима без авторизации.

Фактически, регистрация — это способ получить возможность войти на сайт. Нередко регистрацию делают обязательной для доступа к сайту.

Владельцу сайта регистрация нужна для создания устойчивого сообщества с социальными связями между пользователями. В обмен на это администрация получает возможность взаимодействовать со своими посетителями напрямую, что достаточно трудно сделать с анонимными пользователями :)

Маркетологу, регистрация, безусловно, нужна для получения данных о посетителях и возможности анонсировать им различные новые возможности проекта, а также рекламировать спонсоров. Ещё это позволяет строить достаточно стройную статистику: кто наши пользователи, чего они хотят? В долгосрочной перспективе это может быть весьма полезным, ведь зная пользователей и их потребности, мы создаём сайт именно для конкретной аудитории, а не для сферических коней в вакууме.

Разработчику регистрация нужна, ведь она позволяет привязать к аккаунту пользователя немало возможностей на сайте, которые достаточно трудно реализовать для гостей. Также регистрация позволяет распределять права доступа к различным ресурсам и возможностям на сайте, что также облегчает жизнь программистам.

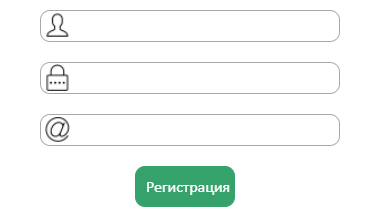


Рисунок 2.13 - Форма регистрации пользователя

**2.11 Блок-схемы**

При авторизации пользователя выполняются следующие действия: ввод данных (логин, пароль); проверка данных записанных с формы; соединение с БД; авторизация (вход).

На рисунке 2.14 представлена блок-схема алгоритма авторизации пользователя.

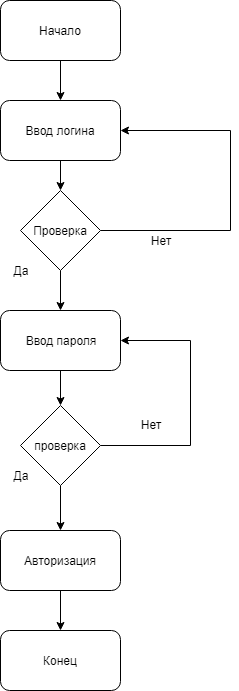


Рисунок 2.14 – Алгоритм авторизации пользователя

|  |  |  |
| --- | --- | --- |
| Характеристика | | Описание |
| Название | | Авторизация |
| Краткое описание | | Ввод логина и пароля в форму авторизации |
| Потоки событий | Основной | Данные введены правильно, переадресация на страницу пользователя |
| Альтернативный | Данные введены не верно, вывод ошибки |
| Название | | Регистрация |
| Краткое описание | | Ввод логина, пароля и почты для регистрации на сайте |
| Потоки событий | Основной | Данные введены правельно, переадресация на страницу пользователя |
| Альтернативный | Логин или почта совпадают, вывод ошибкии, повторная регистрация |

При регистрации пользователь выполняет следующие действия: ввод данных (логин, пароль, почты); проверка данных записанных с формы; соединение с БД (запись данных в БД); регистрация.

На рисунке 2.15 представлена блок-схема алгоритма регистрации пользователя.

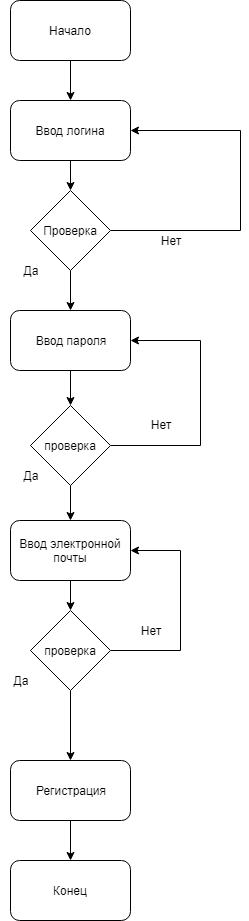


Рисунок 2.15 – Блок-схема алгоритма регистрации пользователя

|  |  |  |
| --- | --- | --- |
| Характеристика | | Описание |
| Название | | Регистрация |
| Краткое описание | | Ввод логина, пароля и почты для регистрации на сайте |
| Потоки событий | Основной | Данные введены правельно, переадресация на страницу пользователя |
| Альтернативный | Логин или почта совпадают, вывод ошибкии, повторная регистрация |

При прохождении тестов пользователь выполняет следующие действия: выборка данных из БД; отправка вводимых данных (вопрос, ответ); получает результат.

На рисунке 2.16 представлена блок-схема алгоритма прохождения курсов.

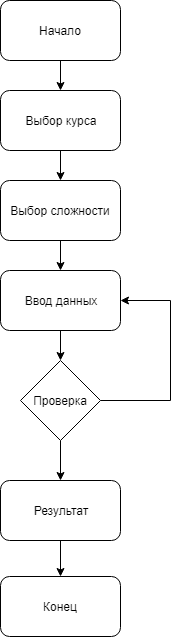


Рисунок 2.16 - Блок-схема алгоритма прохождения курсов

|  |  |  |
| --- | --- | --- |
| Характеристика | | Описание |
| Название | | Тесты |
| Краткое описание | |  |
| Потоки событий | Основной | Добавление, редактирование, удаление тестов |
| Альтернативный | - |

|  |  |
| --- | --- |
| Переадресация  На выбор сложности | Вывод ошибки |
| Выбор сложности | |
| Переадресация на прохождение тестов | |
| Выборка из данных БД | |
| Выбрать вариант ответа  и нажать «Далее» | Нажать на кнопку  «Пропустить» |
| Просмотр результата тестов | |

**3 Проектирование и разработка**

Проектирование — процесс определения архитектуры, компонентов, интерфейсов и других характеристик системы или её части. Результатом проектирования является проект — целостная совокупность моделей, свойств или характеристик, описанных в форме, пригодной для реализации системы.

Веб-разработка — процесс создания веб-сайта или веб-приложения. Основными этапами процесса являются веб-дизайн, вёрстка страниц, программирование для веба на стороне клиента и сервера, а также конфигурирование веб-сервера.

**3.1 Проектирование интерфейса**

В современном мире миллиарды веб-сайтов. У каждого веб-сайта свой интерфейс, являющийся «рычагом» взаимодействия между пользователем и машинным кодом. Не удивительно, что чем лучше интерфейс, тем эффективнее взаимодействие и выше посещаемость пользователей.

Однако далеко не все разработчики и даже дизайнеры, задумываются о создании удобного и понятного графического интерфейса пользователя.

Далее на рисунках будут предоставлены начальные наброски интерфейса данного веб-сайта.

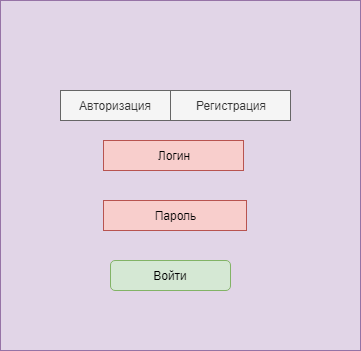
****

Рисунок 3.1 **–** интерфейс главной страницы

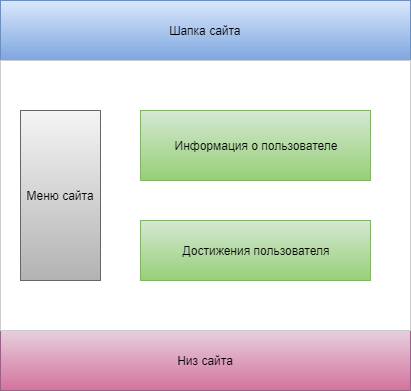


Рисунок 3.2 **–** Интерфейс страницы пользователя

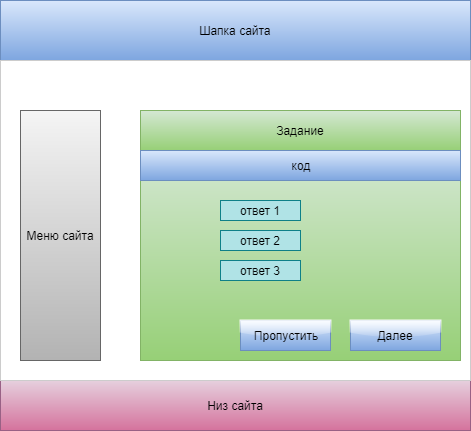
****

Рисунок 3.3 **–** Интерфейс прохождения тестирования

**3.2 Дизайн проекта**

Дизайн сайта – это совокупность графических элементов, шрифтов и цветов, реализованных на сайте.

Основная задача дизайна сайта – предоставить информацию пользователям в удобном и привлекательном виде. Дизайн задаёт стиль сайта. Хороший веб дизайн с первого взгляда дает пользователя понять, что именно ждет его на конкретном сайте.

Дизайн должен быть привлекательным, но не навязчивым, не броским. В первую очередь пользователи приходят на сайт за информацией. Дизайн сайта должен подчеркивать эту информацию, помогать получить к ней доступ, но не становиться препятствием.

Разработка дизайна сайта состоит из нескольких этапов:

1. Выбор цветового решения. Наши дизайнеры разрабатывают оформление сайта в фирменном стиле, подбирают удачные цветовые сочетания. Контрастный основным цветам оттенок выбирают для выделения главной информации, акционных предложений, призывов к действию, кнопок оформления заказа или покупки товара.
2. Создание композиции. Веб дизайн включает правильную расстановку элементов на странице, в которой пользователю будет легко и просто ориентироваться. С помощью техник оформления мы подаем информацию на страницах в удобной форме для хорошего восприятия.
3. Подбор шрифтов. Красивая типографика играет важную роль в восприятии текстового контента. Стиль, начертание, кегль, интерлиньяж – все это способы получить эмоциональный отклик посетителя и выделиться на фоне остальных сайтов схожей тематики.
4. Внедрение анимационных эффектов там, где это уместно. Такой инструмент, как анимация, создает эффект взаимодействия с посетителем, хорошо работает на вовлечение, привлекает внимание к целевому действию.
5. Подбор фоновых и тематических изображений, иконки, декоративные элементы. Сочные графические компоненты, яркие бэкграунды, иконки в едином стиле привлекают внимание, придают сайту характер, делают его крутым и современным в глазах пользователя.

В данном курсовом проекте был разработан веб-сайт в многофункциональном текстовом редакторе SublimeText 3.

Данный веб-сайт состоит из множества страниц, но основные это:

1. Регистрация.
2. Авторизация.
3. Страница пользователя.
4. Курсы.
5. Выбор сложности.
6. Прохождение тестов.

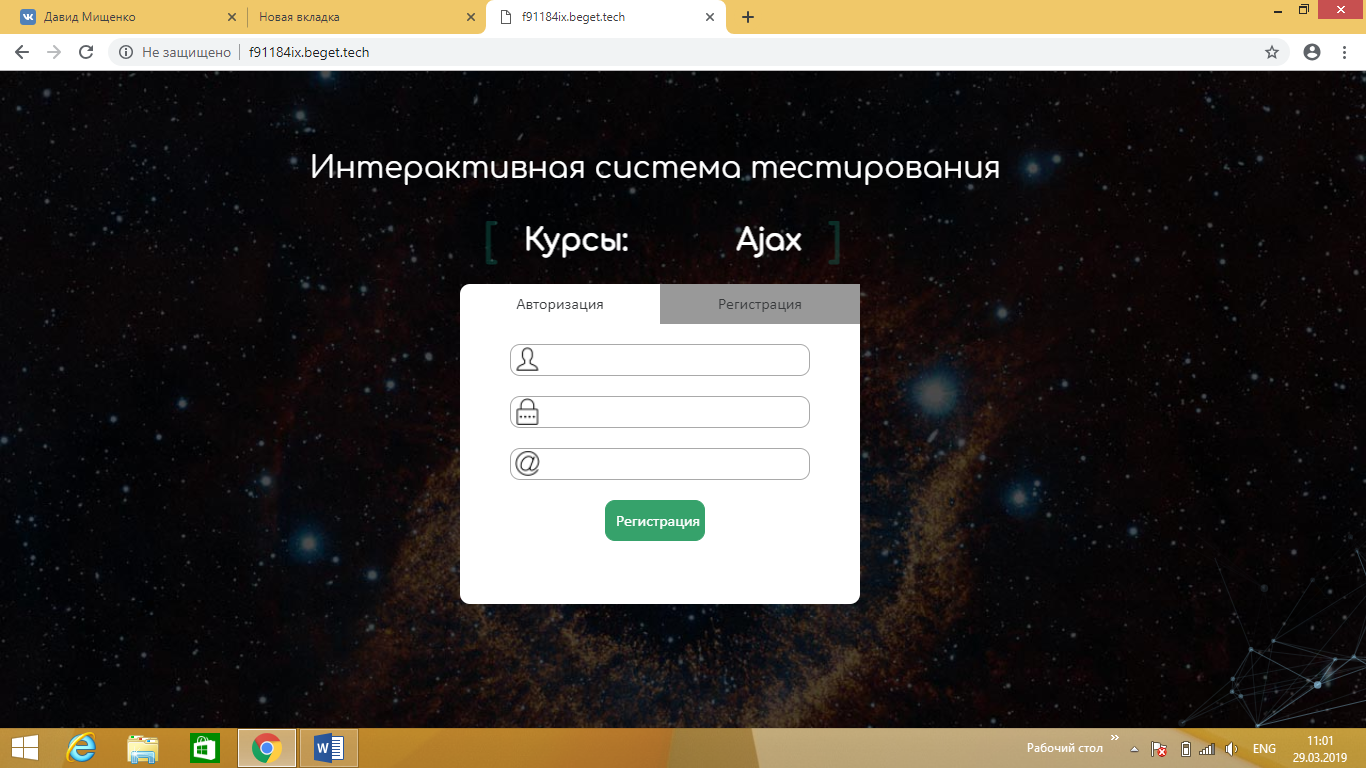


Рисунок 3.4– Регистрация

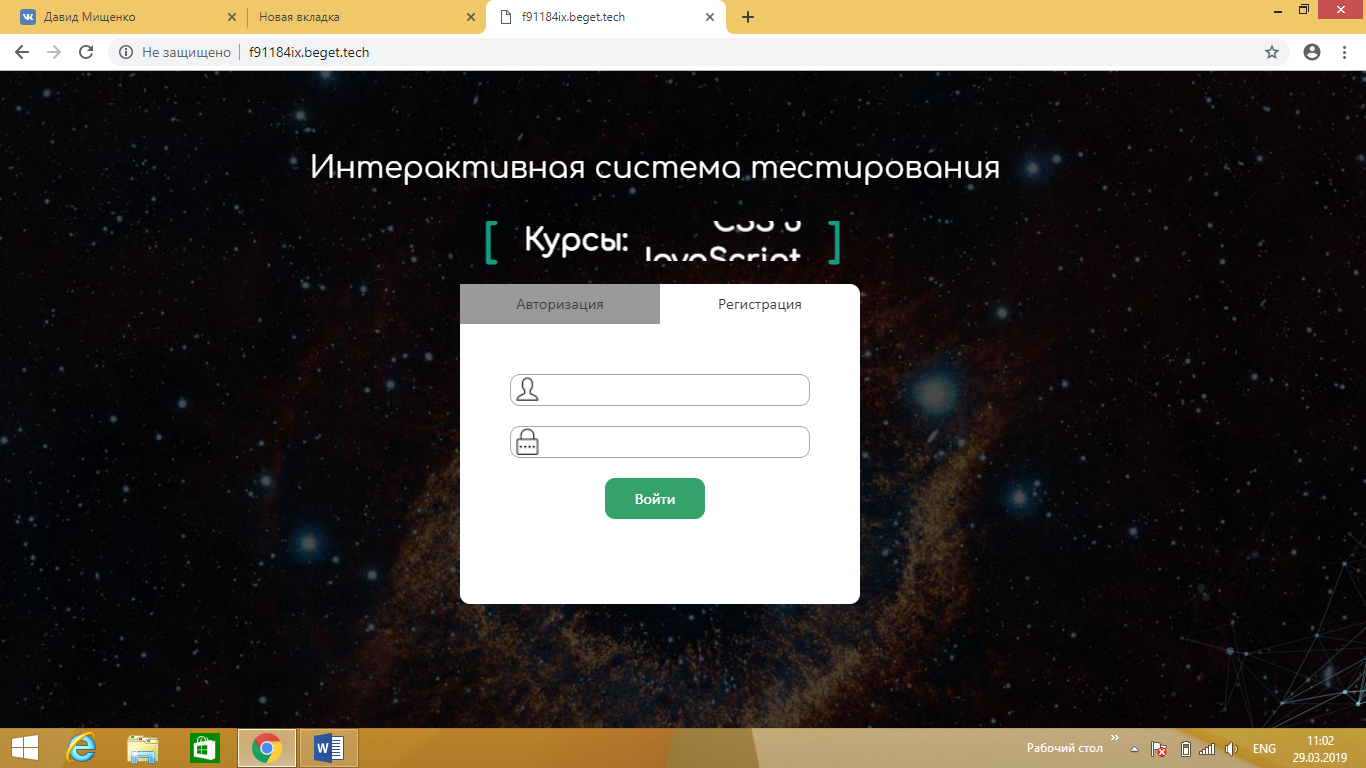


Рисунок 3.5 – Авторизация

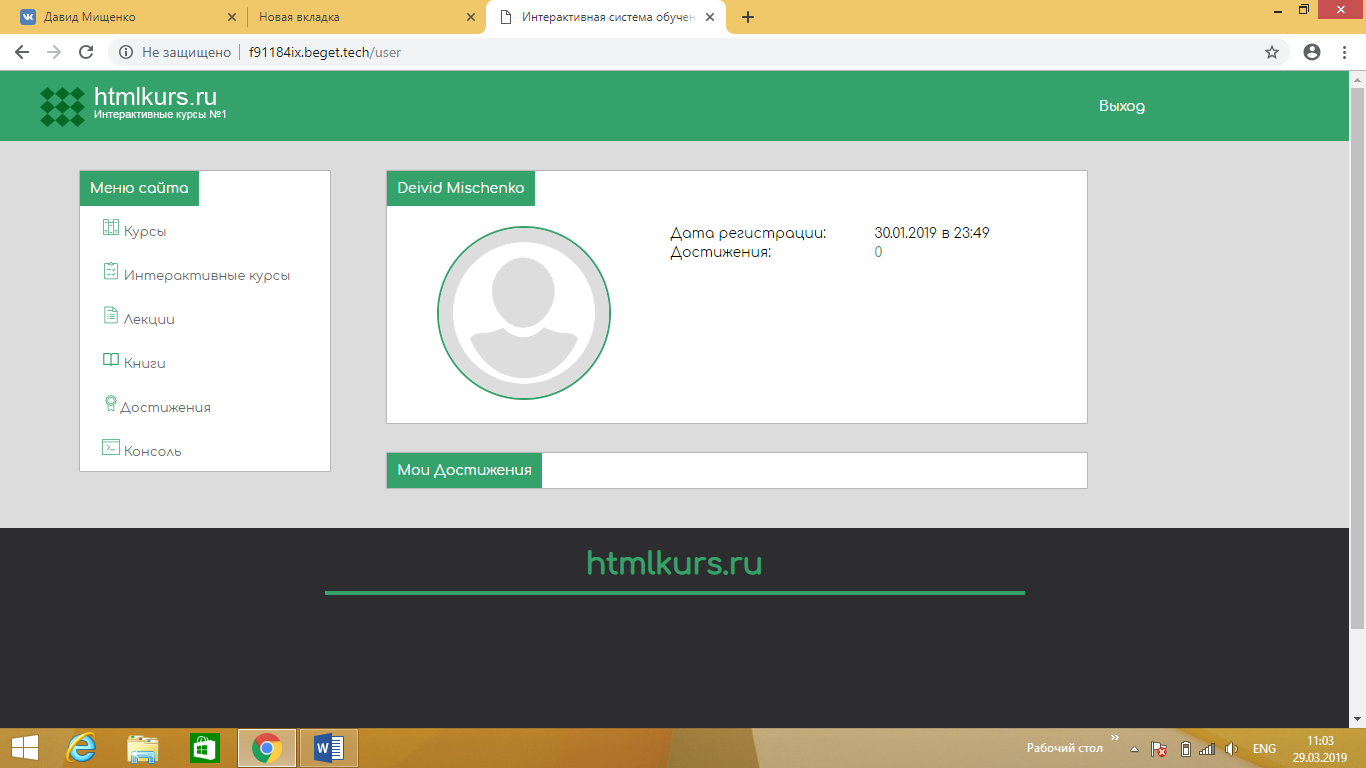


Рисунок 3.6 – Страница пользователя

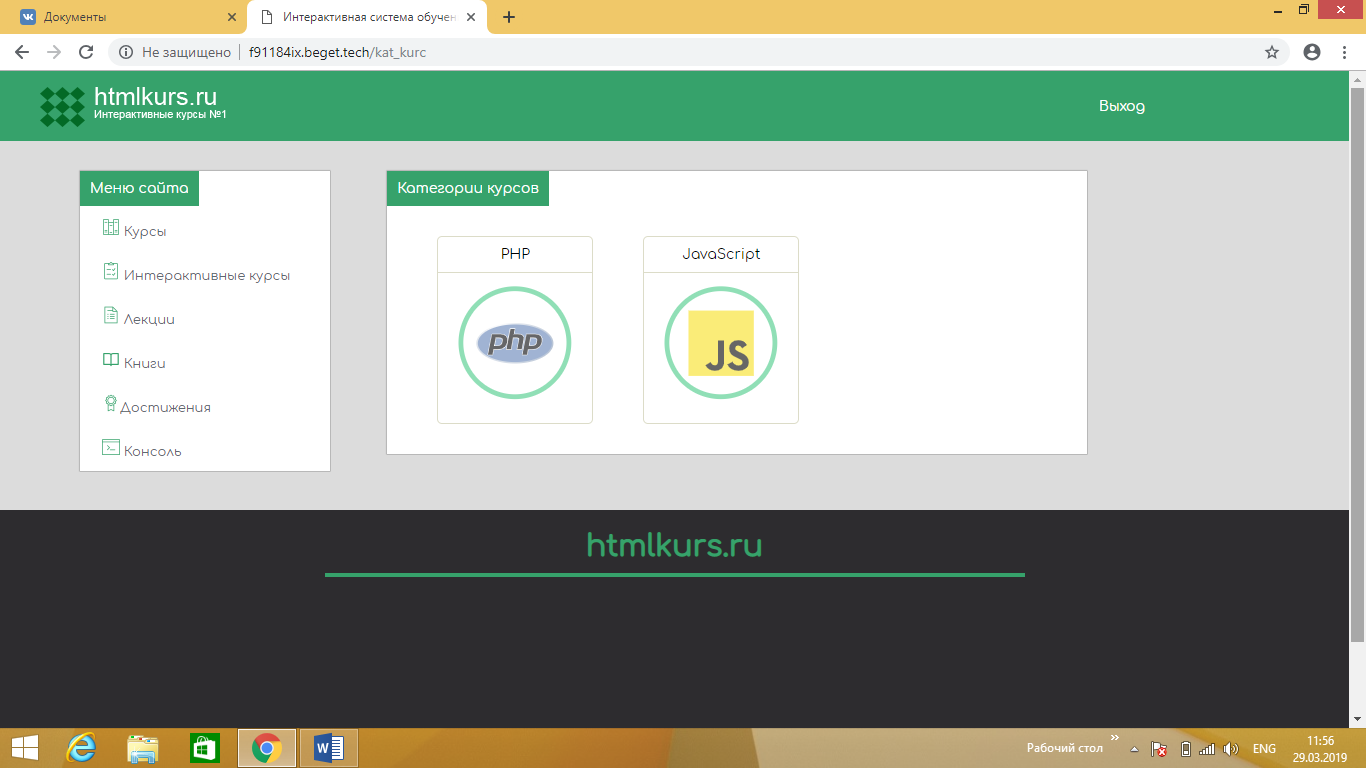


Рисунок 3.7 – Курсы

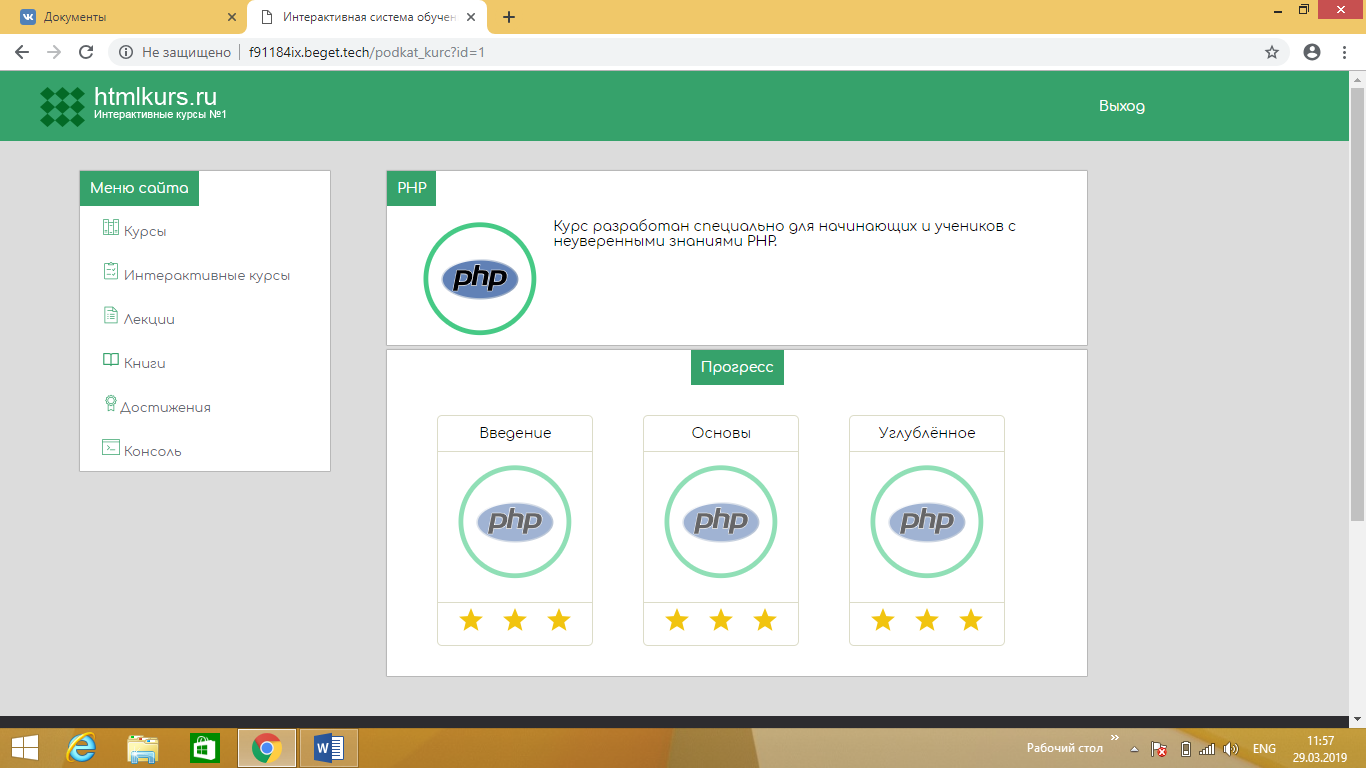


Рисунок 3.8 – Выбор сложности

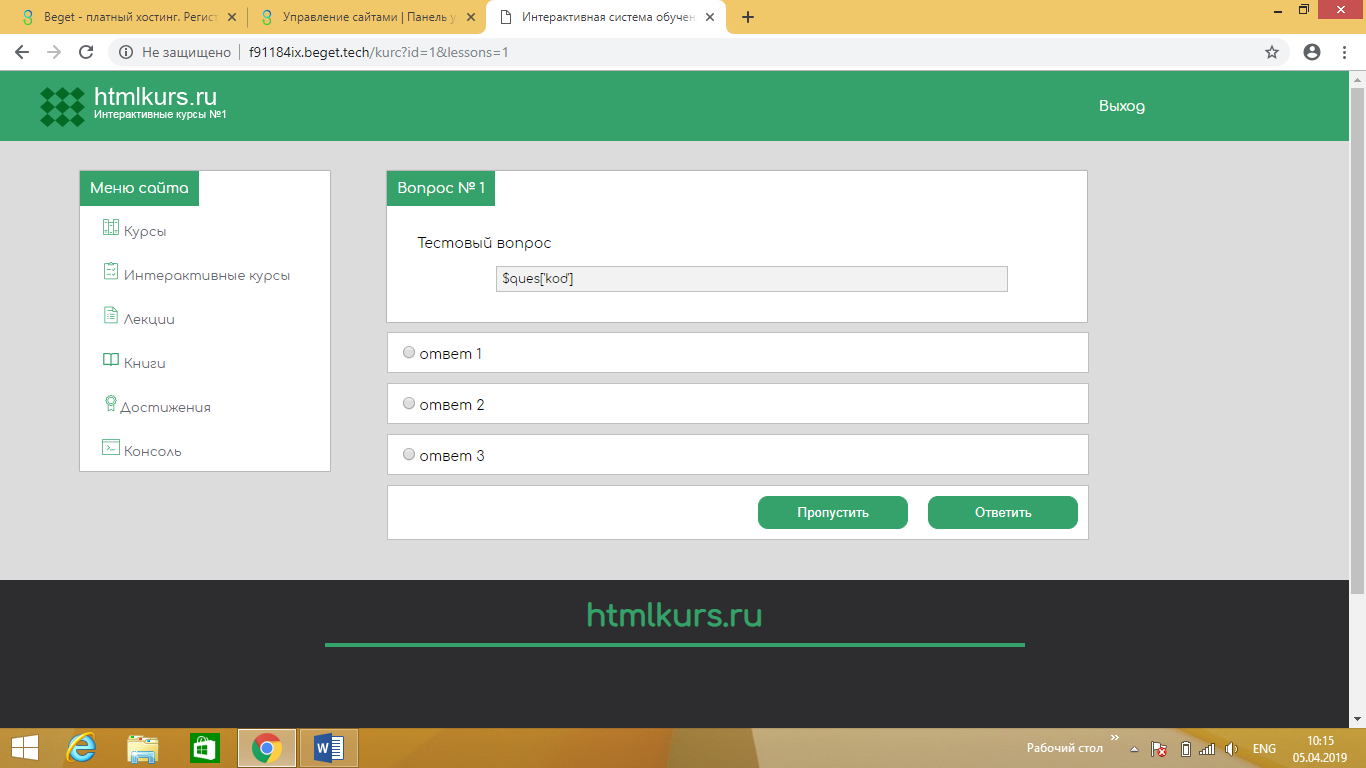


Рисунок 3.9 – Прохождение тестов

**4 Программная реализация**

Веб-разработка — процедура создания WEB-приложения или WEB-сайта. Основными этапами этого процесса являются такие мероприятия, как WEB-дизайн, вёрстка страниц сайта, WEB-программирование на стороне сервера и клиента, а также работы по конфигурированию WEB-сервера.

На сегодняшний день существуют несколько этапов разработки веб-сайта:

1. Проектирование сайта или веб-приложения (сбор и анализ требований, разработка технического задания, проектирование интерфейсов);
2. Разработка креативной концепции сайта;
3. Создание дизайн-концепции сайта;
4. Создание макетов страниц;
5. Создание мультимедиа и FLASH-элементов;
6. Вёрстка страниц и шаблонов;
7. Программирование (разработка функциональных инструментов) или интеграция в систему управления содержимым (CMS);
8. Оптимизация и размещение материалов сайта;
9. Тестирование и внесение корректировок;
10. Публикация проекта на хостинге;
11. Обслуживание работающего сайта или его программной основы.

**4.1 Описание разработанных функций**

Функция - это именованный блок кода, в данном случае на языке PHP, который определяется один раз, а затем может вызываться на исполнение сколько угодно раз.

Ниже приведены функции, которые использованы в проекте, а также их описание.

Первая функция используется для защиты вводимых данных.

function filter($msg) {

return  htmlspecialchars(trim($msg));

}

Процедура htmlspecialchars - Преобразует специальные символы в HTML сущности.

Процедура trim — Удаляет пробелы (или другие символы) из начала и конца строки.

Вторая функция используется для проверки на наличие авторизации пользователя.

function auth(){

if(!isset($\_SESSION['id'])){

?>

<script type="text/javascript">

document.location.replace(" /404.php");

</script>

<?php

exit;

}

}

Процедура isset — Определяет, была ли установлена переменная значением, отличным от NULL.

$\_SESSION - Ассоциативный массив, содержащий переменные сессии, которые доступны для текущего скрипта.

document.location.replace(" /404.php"); - переадресация на страницу если пользователь не авторизован.

Третья функция используется для обработки bbcode.

function AddBB($var) {

$search = array(

'/\[b\](.\*?)\[\/b\]/is',

'/\[i\](.\*?)\[\/i\]/is',

'/\[u\](.\*?)\[\/u\]/is',

'/\[img\](.\*?)\[\/img\]/is',

'/\[url\](.\*?)\[\/url\]/is',

'/\[url\=(.\*?)\](.\*?)\[\/url\]/is',

);

$replace = array(

'<strong>$1</strong>',

'<em>$1</em>',

'<u>$1</u>',

'<img src="$1" />',

'<a href="$1">$1</a>',

'<a href="$1">$2</a>'

);

$var = preg\_replace ($search, $replace, $var);

return $var;

}

Четвёртая функция используется для работы со временем.

function daytime($var) {

$date = date('d.m.Y', $var);

$time = date('H:i', $var);

$hour = date('H', $var);

$minut = date('i', $var);

if ($date == date('d.m.Y')) {

if ($time == date('H:i')) {

$full\_time = 'Толькочто';

} elseif ($hour == date('H')) {

$full\_time = quantDec(date('i') - $minut, ['минуту', 'минуты', 'минут']) . ' назад';

} else {

$full\_time = date('Сегодняв H:i', $var);

}

} elseif ($date == date('d.m.Y', time() - 3600 \* 24)) {

$full\_time = date('Вчерав H:i', $var);

} elseif ($date == date('d.m.Y', time() - 3600 \* 24 \* 2)) {

$full\_time = date('Позавчерав H:i', $var);

} elseif ($date == date('d.m.Y', time() + 3600 \* 24)) {

$full\_time = date('Завтрав H:i', $var);

} elseif ($date == date('d.m.Y', time() + 3600 \* 24 \* 2)) {

$full\_time = date('Послезавтрав H:i', $var);

} else {

$full\_time = date('d.m.Y в H:i', $var);

}

return $full\_time;

}

**4.2 Разработка системы тестирования знаний**

Модуль системы тестирования разработан на языке программирования PHP.

Визуальная часть представлена в виде нескольких небольших блоков. В первом блоке находится вопрос, на который нужно ответить. Далее код если он прилагается к данному вопросу. Далее идут три варианта ответа из которых пользователь выберет только один. Потом при выборе варианта ответа он нажимает на кнопку “Ответить” или кнопку “Пропустить”.

Ниже представлен код формы:

echo "<form action = '' method = 'POST'>";

echo "<div class = 'test'>";

echo Addbb($ques['name']);

if($ques['kod'] != NULL)echo "<br><br><div class ='kod'>".$ques['kod']."</div>";

echo "</div>";

echo "</div>";

echo "<input type = 'hidden' name = 'id' value = '".$ques['id']."' >";

echo "<input type = 'hidden' name = 'answers' value = '".$ques['answers']."' >";

echo "<div class = 'otvet'><input type = 'radio' name = 'otvet' value = '".$ques['answer\_1']."' >".$ques['answer\_1']."</div>";

echo "<div class = 'otvet'><input type = 'radio' name = 'otvet' value = '".$ques['answer\_2']."' >".$ques['answer\_2']."</div>";

echo "<div class = 'otvet'><input type = 'radio' name = 'otvet' value = '".$ques['answer\_3']."' >".$ques['answer\_3']."</div>";

echo "<div class = 'otvet'>

<input class = 'button' type = 'submit' name = 'submit' value = 'Пропустить'>

<input class = 'buttons' type = 'submit' name = 'reply' value = 'Ответить'>

</div>";

echo "</form>";

А дальше представлена визуальная часть модуля:

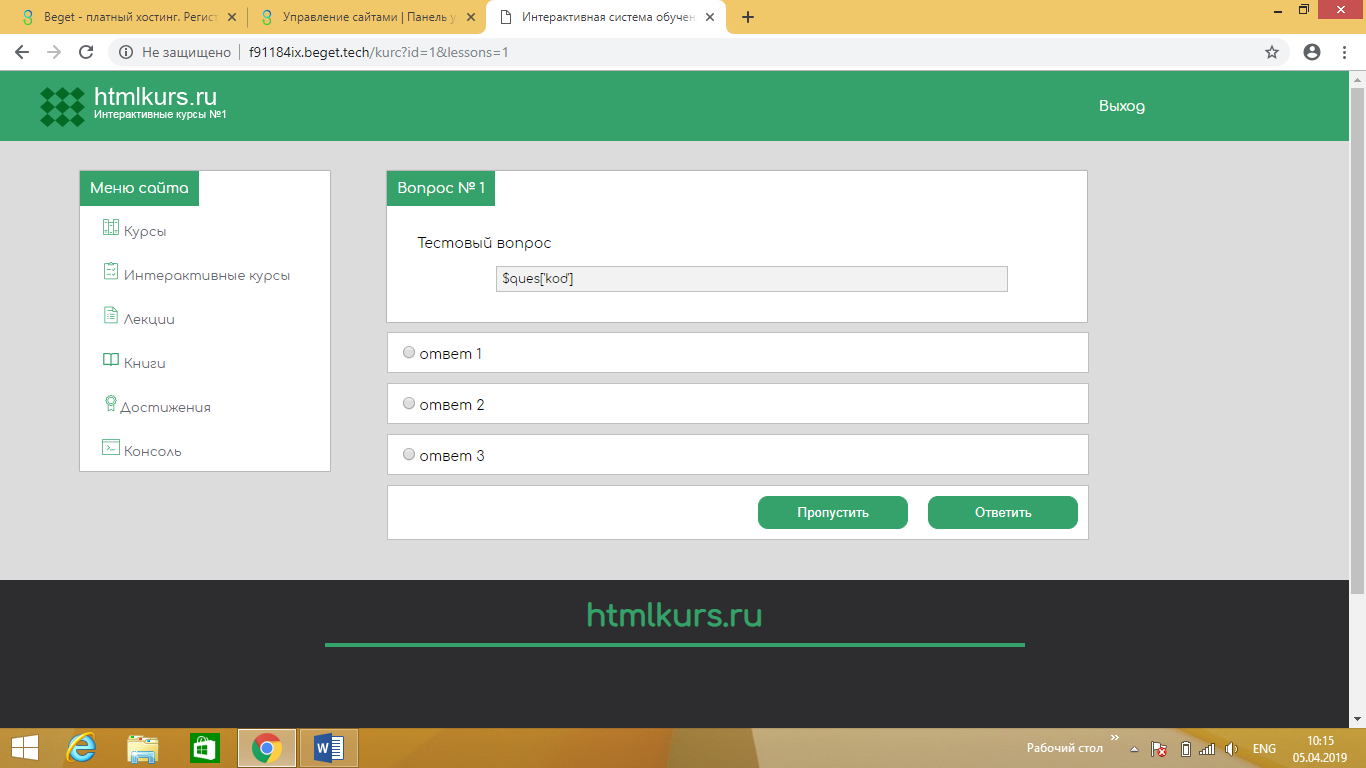


Рисунок 4.1 – Прохождение тестов

Выбранный пользователем ответ заноситься в переменную, а потом в БД. Когда пользователь ответил на определённое количество вопросов, которое задается администратором в администраторской части, выводится результат.

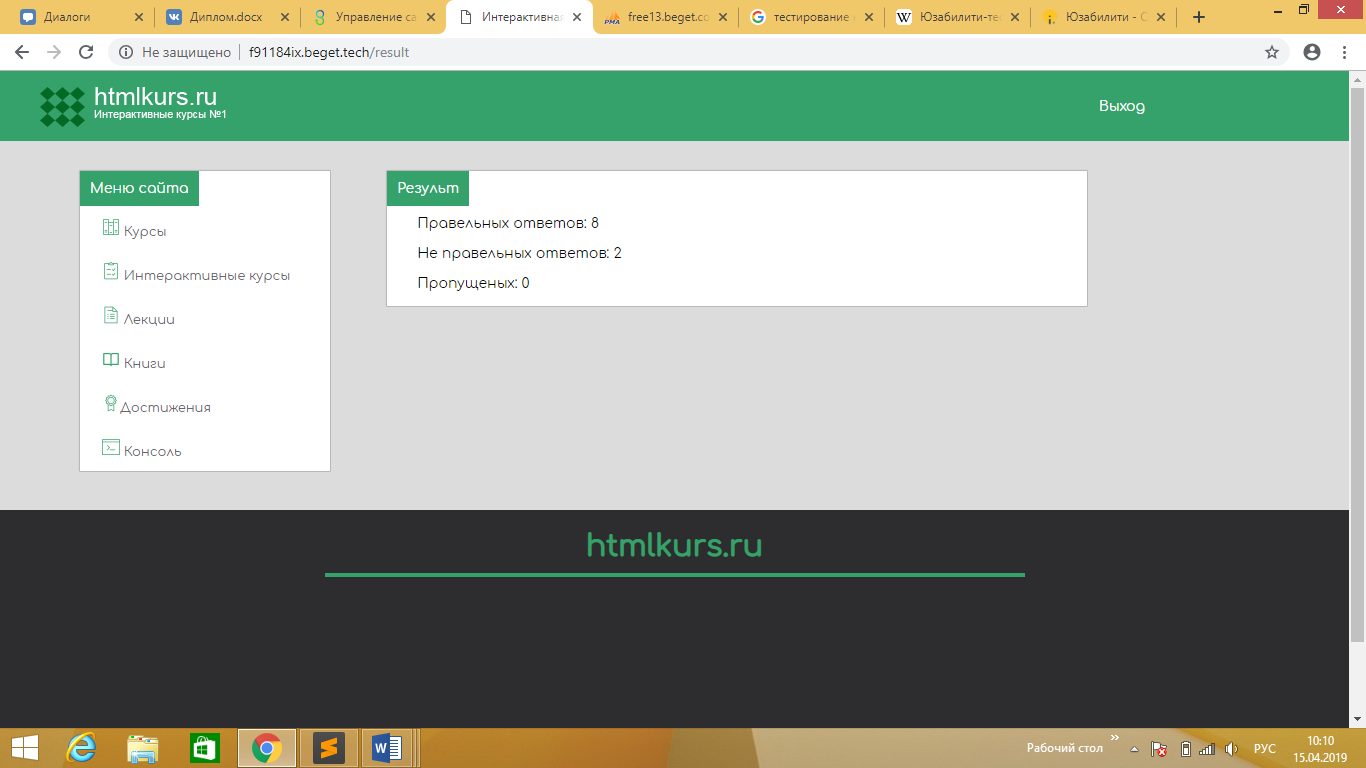


Рисунок 4.2 – Результат прохождения тестов

**5 Тестирование**

Тестирование сайта — один из важных жизненных этапов, после которого, предоставляется заказчику готовый проект без ошибок, с хорошей читабельностью, воспринимаемой легкостью, удобством пользования и надежностью.

Тестирование — это отклонение фактического результата от ожидаемого, другими словами — это процесс поиска багов (ошибок).

Основные правила тестирования веб-сайтов — это шаги, которые показывают пользователю, насколько удобен и логичен будет проект, насколько просто и возможно найти ту или иную информацию. Хорошо ли воспринимается человеческому взгляду и правильно ли работает весь функционал данного сайта, который был поставлен по ТЗ или брифу — это основные показатели для тестировщика.

**5.1 Функциональное тестирование**

Функциональное тестирование проводится для определения, насколько компонент или система соответствуют заданным функциональным требованиям, описанным в спецификациях.

Данный вид тестирования может проводиться на всех уровнях тестирования: компонентом, интеграционном, системном и приемочном, т.е. на всех этапах разработки программного обеспечения.

Этапы функционального тестирования:

1. Анализ исходных данных (технических требований) и согласование плана тестирования, тест-кейсов, сроков выполнения проекта, числа итераций;
2. Проведение функционального тестирования по тестовым сценариям с занесением выявленных багов в систему багтрэкинга;
3. Составление отчета о проведенном тестировании и предоставление рекомендаций по улучшению системы.

Функциональное тестирование по следующим направлениям:

* 1. Регрессионное тестирование

Тестируем продукт на его работоспособность после внесения изменений в функциональность.

* 1. Модульное тестирование

Тестируем на корректность отдельные компоненты (модули) программы.

1. Интеграционное тестирование

Мы проверяем на корректность взаимодействия между компонентами одной системы и правильности обработки информации.

1. Дымовое тестирование

Проводим цикл тестов на проверку функциональности программного продукта после его сборки (добавления нового кода либо исправления ошибок в коде).В случае использования метода непрерывной интеграции (Continious Integration) сборка программного продукта производится ежедневно, поэтому проведение дымового тестирования позволяет вовремя выявить и устранить критичные ошибки, тем самым сэкономив время на тестирование спроводим цикл тестов на проверку функциональности программного продукта после его сборки (добавления нового кода либо исправления ошибок в коде).В случае использования метода непрерывной интеграции (Continious Integration) сборка программного продукта производится ежедневно, поэтому проведение дымового тестирования позволяет вовремя выявить и устранить критичные ошибки, тем самым сэкономив время на тестирование сборки.

5. Тестирование безопасности

Тестируем сайт на наличие уязвимостей в безопасности программного обеспечения, в частности безопасности подключений, безопасности данных и безопасности доступа.

6.Системное тестирование

Для того, чтобы убедиться в том, что интегрированная и готовая к эксплуатации система соответствует заявленным функциональным требованиям, мы проводим системное тестирование.

7. Стресс-тестирование

Проводим тестирование на отказ системы и ее способность к восстановлению при возникновении сбоев.

8. Юзабилити – тестирование

Мы проверяем продукт на удобство и простоту использования путем имитации поведения пользователей либо посредством экспертной оценки результатов тестирования юзабилити продукта фокус группой.

**5.2 Тестирования удобства пользования (юзабилити)**

Тестирование удобства пользования (юзабилити) – это вид тестирования, который делает для сайта удобство и практичность в использовании. Основная цель показать пользователю:

* Понятен ли ваш сайт для окружающих и удобен ли?
* Удобная навигация?
* Какое впечатление создается у пользователя?
* Что может быть лишним или не нужным.

Основная задача тестирования юзабилити сайта — это дизайн того, что пользователь хочет найти и приобрести, что он ищет необходимую информацию и для этого ему ничего не мешает.

Юзабилити — это степень качества обще пользовательского навыка, наработанного в результате коммуникации с устройством или системой, к примеру, веб-ресурсом, программным обеспечением, приложением.

Другими словами, usability сайта — это простота в использовании, удобство и полезность для посетителей. Это мера, которая оценивает качество использования ресурса, его привлекательность и пригодность к взаимодействию с пользователями.

Оценить и проверить юзабилити сайта можно с помощью следующих пунктов:

* Структура;
* Важная информация;
* Дизайн;
* Навигация.

Структура сайта

Это слово можно произносить вечно, т.к. грамотная, но в то же время простая и удобная структура играет основополагающую роль при взаимодействии веб-портала с его посетителями. Пользователю при попадании на сайт сразу же должно быть интуитивно понятно, куда нажимать и что же произойдет после нажатия, на какую страницу он перейдет и что он там увидит. Поэтому еще раз повторю, что грамотно проработанная структура пойдет вашему ресурсу лишь на пользу, ибо пользователь не будет бесцельно блуждать из раздела в подраздел и так по кругу до тех пор, пока он просто не покинет интернет-площадку, так ничего для себя, не отыскав из-за того, что структура онлайн-площадки тяжелая, непонятная и неудобная в использовании. Структура очень важна для юзабилити.

Важная информация на сайте

Важная, нужная и полезная, на ваш взгляд, информация должна сразу бросаться в глаза (сразу — это в течение первых 3 секунд пребывания на площадке). Для чего это нужно? Опять же для удобства ваших посетителей и повышения юзабилити. Вы же не хотите, чтобы они обошли всю интернет-площадку в поисках таких страниц, как, к примеру, “Контакты”, “Оплата и доставка”, номера телефонов, график работы компании и т.д. и т.п., и так ничего и не нашли и покинули площадку? Либо нашли, но приложили так много усилий, что уже никуда звонить-то и не хочется, да и зачем звонить, если они уже напрочь позабыли о цели своего визита? Поэтому, такого рода информацию необходимо размещать на заметных местах — в хедере, а некоторую контактную информацию даже дублировать в футере.

Дизайн сайта

Дизайн играет далеко не последнюю роль в плане удобства для пользователей. Что же подразумевается под понятием юзабилити дизайна сайта? Дизайн включает в себя множество критериев, по которым можно оценить удобство веб-площадки. Вот некоторые из них:

* общее восприятие ресурса легкое (интернет-ресурсы со сложным дизайном плохо воспринимаются его посетителями);
* графические элементы простые, а их расположение — продуманное;
* приятные и сочетающиеся тона, в которых выполнен сайт, — не раздражающие и не агрессивные оттенки;
* элементы управления расположены логически;
* текст соответствует дизайну (форма подачи текста и его содержание представляют собой единое целое, и именно в таком виде информация наиболее понятна и удобна пользователю);
* современность и креативность (думаю, что здесь не стоит много писать, и так понятно, что пользователь на дольше задержится на ярком и стильном ресурсе).

Навигация по сайту

Крайне важно, чтобы она являлась простой и удобной. Также рекомендуется, чтобы навигация была одинаковой на протяжении всех страниц. Это улучшит как ее восприятие, так и юзабилити. Человеку не придется на каждой новой странице привыкать к расположению каких-либо кнопок или элементов ресурса. Вот некоторые рекомендации по повышению удобства и простоты навигации:

1. В хедере размещать логотип компании. При этом:

логотип находится на каждой странице сайта;

кликабельный на всех страницах (даже на главной);

желательно должен быть расположен слева в шапке (в крайнем случае по центру шапки, если того требует дизайн);

хедер с логотипом фиксированный на всём сайте (делается это для повышения юзабилити, чтобы в любой момент времени из любой точки ресурса пользователь мог вернуться на главную простым нажатием на лого компании);

рекомендуется дублировать логотип в футере (либо располагать в подвале ссылку “На главную”).

2. Размещать на всех страницах ресурса меню 1-го уровня (т.к. место в меню довольно-таки ограниченно, в нем должны располагаться действительно важные разделы, такие как, к примеру, “О компании”, “Оплата”, “Доставка”, “Контакты”, “Отзывы”, а также основные разделы каталога компании). Естественно, такое меню лучше всего делать фиксированным и размещать непосредственно в шапке, это, естественно, улучшит юзабилити. Хочу отметить, что некоторые ресурсы, дабы не захламлять шапку, располагают все служебные страницы над ней, а уже в самом хедере размещают только основные разделы каталога. Получается, как бы две шапки сайта, но в этом определенно, что-то есть.

* 1. **Тестирование производительности**

Тестирование производительности — в основном это нагрузочное тестирование. Нагрузочное тестирование сайта проверяется в большинстве случаев автоматом, то есть специальными программами. Это дает шанс проверить, насколько он будет работать под определенной нагрузкой.

Цель этого тестирования, заключается в кол-ве виртуальных пользователей, которые задают n кол-во запросов, в одно время (будь это секунды даже). Тем самым результат дает то, смог ли наш проект выдержать, к примеру, 100 пользователей, которые одновременно покупали товар или авторизовались на сайте, ответ показывает, реально ли выдержать сайт такую нагрузку.

* 1. **Тестирование интерфейса пользователя (UI testing)**

UI testing - это тестирование графического интерфейса пользователя, которая предполагает проверить сайт на соответствия требованиям к графическому интерфейсу, профессионально ли оно выглядит, выполнено ли оно в едином стиле.

В большинстве случаев, тестирование интерфейса пользователя, осуществляется вместе со следующими видами тестирования(UI):

1. Тестирование на соответствие стандартам графических интерфейсов
2. Тестирование с различными разрешениями экрана
3. Тестирование кроссбраузерности, или совместимости с разными интернет браузерами и их версиями
4. Тестирование локализованных версий: точность перевода (мультиязычность, мультивалютность), проверка длины названий элементов интерфейса и т.д
5. Тестирование графического интерфейса пользователя на целевых устройствах (смартфоны, кпп, планшеты).
   1. **Тестирование безопасности.**

Основные правила этого тестирования — это проверка на уязвимость разных видов атак. Если это интернет-магазин, то, скорее всего, следует проверять запросы на Sql инъекцию (запросы к базе данных).

SQL-инъекции — это вредоносный код в запросах базы данных — наиболее опасный вид атак.  Если это различные формы общественные (гостевая книга), то проверка на XSS инъекции. Дает возможность внедрить произвольный код, и атаковать компьютер пользователей, просматривающих зараженные страницы.

Вот, пожалуй, основные виды, которые используют для тестирования сайта.

Главная задача тестирования заключается в том, чтобы заказчику, как и пользователю, было предоставлено качество продукта. Для этого перед началом любого сайта следует составить чек-лист для тестирования:

* проверки веб-форм;
* проверки правильности данных;
* тестирования интерфейса пользователя;
* тестирования оплаты;
* тестирования версий для печати;
* тестирования отчетов.

1. **Экономические расчёты**

## Исходные данные для расчета затрат

В дипломном проекте разработана система тестирования знаний по программированию.

Общая стоимость владения (ОСВ) представляет собой совокупность затрат, осуществляемых на следующих стадиях жизненного цикла системы тестирования знаний по программированию: разработка технологического проекта, его внедрение, эксплуатация и сопровождение.

Определим исходные данные для расчета затрат на осуществление технического проекта системы. При проектировании используется трудовой потенциал сотрудников, а также персональный компьютер, оргтехника и специальное программное обеспечение для проектирования системы.

Сведем в таблицу 1 трудоемкость инженерно-технического персонала.

Таблица 1. – Трудоемкость работы исполнителей

|  |  |  |
| --- | --- | --- |
| **Должность** | **Оклад**  **(О), (руб./мес. или руб./час).** | **Трудоемкость (Т),**  **дней (часов или месяцев)** |
| Руководитель проекта | 100руб./час | 2 месяца |
| Программист-тестировщик | 115руб./час | 2 месяца |

Для определения амортизационных отчислений сведем в таблицу 2 используемое оборудование и программное обеспечение (ПО), используемое при проектировании системы (программы).

Таблица 2. – Используемое оборудование и ПО для проектирования системы

|  |  |  |
| --- | --- | --- |
| **Наименование оборудования** | **Кол-во единиц** | **Первоначальная стоимость,**  **руб.** |
| Ноутбук Lenovo Idea pad 100-15IBY | 1 | 23000 |

Ниже запишем исходные данные для расчета затрат на внедрение системы.

При внедрении систем учитываются трудозатраты сотрудников, осуществляющих работы (кодирование, монтаж систем, тестирование, отладку и т.д.), а также затраты на оборудование, комплектующие, материалы и специальное программное обеспечение, позволяющее физически реализовать и обеспечить качество системы. Они показаны в таблице 3.

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование затрат** | **Количество** | **Цена за единицу (руб.)** | **Сумма (руб.)** |
| Оборудование |  |  |  |
|  |  |  |  |
| Работы по внедрению |  |  |  |

Ниже определим исходные данные для расчета затрат на внедрение системы.

При внедрении систем учитываются трудозатраты сотрудников, осуществляющих работы (кодирование, монтаж систем, тестирование, отладку и т.д.), а также затраты на оборудование, комплектующие, материалы и специальное программное обеспечение, позволяющее физически реализовать и обеспечить качество системы. Они показаны в таблице 3.

Таблица 3. – Затраты на внедрение системы

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование затрат** | **Количество** | **Цена за единицу, руб** | **Сумма, руб** |
| **Оборудование** | | | |
| Разъем RJ45 | 64 | 1,5 | 96 |
| **Работы по внедрению** | | | |
| Организация канала WiMax | 1 | 15000 | 15000 |
| **Итого:** |  |  | 81151 |

Таблица 3. – Затраты на внедрение системы

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование затрат** | **Цена (руб./ед.)** | **Количество** | **Сумма (руб.)** |
| **Стоимость базового программного обеспечения** | | | |
| ОС Windows 8.1 | 3500 | 1 | 3500 |
| СУБД phpmyadmin | 1500 | 1 | 1500 |
| **Стоимость базового оборудования** | | | |
| Ноутбук Lenovo idea pad 100-15IBY/ Celeron N2840/ 2GB/ 255GB/  Intel HD Graphics/ DVDRW/  кл.+мышь | 23000 | 1 | 23000 |
| **Работы по внедрению** | | | |
| **Наименование работ** | **Время, дни** | **Оклад (руб./день)** | **Сумма (руб.)** |
| Кодирование системы | 25 | 1000 | 25000 |
| Отладка системы | 5 | 1000 | 5000 |
| Тестирование системы | 3 | 1000 | 3000 |
| Создание инсталляционного пакета | 1 | 1250 | 1250 |
| Создание руководства пользователя | 3 | 1250 | 3750 |
| ***Итого:*** |  |  | 66000 |

Определим нормы затрат (% берется от основной ставки: зар.платы, стоимости оборудования и т.д.):

- норма дополнительной заработной платы – 10% от основной зар.платы;

- норма премий – 20%;

- коэффициент отчислений в фонды социального страхования – 30% от ФЗП годового;

- норма транспортных расходов – 2-4% от первоначальной стоимости оборудования;

- норма амортизации – 15-20%;

- норма затрат на техническое обслуживание – 7-9% от первоначальной стоимости оборудования;

- норма затрат на расходные материалы – 2%;

- мощность, потребляемая вычислительной техникой:

а) компьютера – 0,42 кВт/ч. (или см. по техническому паспорту);

б) принтера и других устройств– 0,1 кВт/ч. (или см. по техническому паспорту);

- действительный фонд времени работы оборудования – 1820 ч.;

- норма накладных расходов – 60%;

- цена 1 кВт электроэнергии – \_\_\_\_\_\_руб.;

- норма расходов на хранение и обновление программного обеспечения – 20%;

- норма расходов на содержание и ремонт производственных

помещений – 1,5-2%.

1. **ЗАКЛЮЧЕНИЕ**

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бадд, Э. Мастерская CSS. Профессиональноеприменение web-стандартов // Э. Бадд [и др.] – Л.: Питер, 2007.
2. Гольцман, В. MySQL 5.0 // В. Гольцман – Л.: Питер, 2009.
3. Гоше, Х. HTML 5. Для профессионалов / Х. Гоше - Ростов-на-Дону: Питер, 2014
4. Грабер, М. Введение в SQL: учебник / М. Грабер – М.: Лори, 2010.
5. Дакетт, Д. HTML и CSS. Разработка и дизайн веб-сайтов // Д. Дакетт – Л.: Питер, 2013.
6. Дмитриева, М.В. Самоучитель JavaScript: учебник / М.В. Дмитриева. – СПб.: БХВ-Петербург, 2003. – 512 с.
7. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript: учебник / Робин Никсон; пер. с англ. Н. Вильчинский. – СПб.: Питер, 2011. – 496 с.
8. Холмогоров, В.В. Основы Web-мастерства: учебник / В.В. Холмогоров. – СПб.: Питер, 2001. – 352 с.