ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

АВТОМАТИЗИРОВАННАЯ СИСТЕМА КОНТРОЛЯ УРОВНЯ ЗНАНИЙ ДЛЯ ЛАБОРАТОРНОГО ПРАКТИКУМА

Магистерская диссертация
обучающейся по направлению подготовки 02.04.01 Математика и компьютерные науки
очной формы обучения, группы 07001531
Мальцевой Екатерины Геннадьевны

Научный руководитель
к.г.н., доцент
Чашин Ю.Г.

Рецензент
д.т.н., профессор кафедры общей математики
Аверин Г.В.

БЕЛГОРОД 2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Выполнение магистерской диссертации является заключительным этапом обучения студента. Выпускнику необходимо доказать свою профессиональную пригодность путем разработки программного обеспечения, показывающего способность студента применять изученные технологии для решения различных задач.

Одной из важных задач автоматизированной системы является создание и нормализация базы данных, которая позволяет хранить в себе множество необходимых данных для дальнейшей работоспособности программ, а именно разнообразие вопросов и ответов на них в рамках материала по дисциплине, информация о зарегистрированных пользователях, которые в дальнейшем будут проходить тестирование.

Суть системы тестирования, рассматриваемой в данной работе, заключается в том, что студент может проверить свою подготовку по лабораторной работе, не отвлекая и не затрачивая время преподавателя. Так же участник тестирования может получить оценку за теоретическую часть своей работы, тем самым получить допуск к защите практической части. Данная оценка будет зависеть он знаний самого студента, а не от отношения к нему проводящего тестирование. Преподаватель же, может уделить внимание тем студентам, которые знают материал и готовы защитить лабораторную работу, а не отвлекаться на не подготовившихся учеников.

Актуальность магистерской диссертации заключается в том, что выполняется особый подход к организации учебного процесса, а именно проверки теоретической части по средствам тестирования. Упрощается оценивание студента, а так же затрачиваемое время на защиту лабораторной работы. Актуальность темы исследования определила цель выпускной квалификационной работы.

Целью работы является исследование учебного процесса по защите лабораторного практикума, а так же внедрение программного обеспечения, позволяющего упросить работу проверки теоретической части лабораторной работы и уменьшить время, затрачивающееся на опрос.

Для достижения этой цели в магистерской диссертации были поставлены следующие задачи:

- Проанализировать тестирование как контроль знаний студентов;
- Исследовать организацию учебного процесса;
- Проанализировать необходимые требования для создания грамотного и отвечающего ГОСТу тестирования;
- Выполнить проектирование автоматизированной системы, учитывая все тонкости проверки теоретических знаний студентов;
- Рассмотреть структурный язык запросов;
- Изучить строение баз данных;
- Изучить основные методы для реализации тестирования студентов;
- Привести созданную базу данных к третьему нормальному виду;
- Разработать программное обеспечение для возможности тестирования студента.

Объектом исследования являются средства автоматизированного управления учебным процессом, а именно подборка СУБД для реализации базы данных и языки веб-программирования.

Предмет – организация учебного процесса с использованием автоматизированной системы, включающей себя веб-интерфейс с нормализованной базой данных.

Практическая значимость выпускной квалификационной работы заключается во внедрении автоматизации процесса в обучение студентов, а именно проверка теоретических знаний студентов по средствам тестирования, а не устных опросов. То есть разработка программного обеспечения, позволяющего организовать проверку теории по лабораторной

работе без участия преподавателя и получение необходимых результатов оценивания знаний.

Материалами, с помощью которых была написана выпускная квалификационная работа, являются научные работы отечественных и зарубежных авторов, различные учебные пособия, а также интернет ресурсы.

Структура, магистерской диссертации включает в себя: введение, содержание, четыре главы, заключение, библиографический список, приложения.

Во введении раскрывается актуальность исследования по выбранному направлению, ставится проблема, цель и задачи исследования, определяются объект, предмет научных поисков, ставятся цель и задачи, указывается методологическая база исследования, его теоретическая, практическая значимости.

В первой главе выпускной квалификационной работы изучены теоретические основы относительно самого процесса тестирования, его недостатки и плюсы по отношению к устной сдаче теории по предмету, организация учебного процесса и анализ существующих систем. Так же изучены главные требования по проведению тестирования и составлению заданий.

Во второй главе показано проектирование информационного обеспечения. А именно, методы реализации автоматизированных систем, проектирование структуры веб-интерфейса, наличие физической модели базы данных, отражающие внешний ее вид. Диаграмма “сущность-связь”, где изображена зависимость и связь таблиц между собой. Список объектов и их свойств, то есть подробно описано содержание всей базы данных.

В третьей главе выполняется реализация автоматизированной системы тестирования, а именно код программного продукта. Будут показаны куски программного кода регистрации и тестирования, которые являются главной частью квалифицированной работы, а так же пояснение к нему.

В четвертой главе производится тестирование работы системы тестирования знаний студентов, а так же скриншоты показывающие страницы доступные разным пользователям. Пробное тестирование нескольких групп студентов, для получения результатов, на основе которых будет установлен определенный порог.

В заключении доказано достижение цели работы, за счет решения поставленных задач, приведен общий вывод по вопросу автоматизированной системы поддержки лабораторного практикума.

Данная работа выполнена с наличием 64 страниц без приложения, 20 рисунков, 5 таблиц, 7 листингов.

1 АНАЛИТИЧЕСКАЯ ЧАСТЬ

1.1 Понятие автоматизированной информационной системы

В современном обществе, а развитом технологически, большая роль отводиться к контролю знаний обучающихся, не важно, работники это какой-либо фирмы или же студенты вузов, школьники. Под **контролем** подразумевается научно обоснованная система, позволяющая выполнить проверку и получить результаты обучения в той или иной сфере.

Проверку знаний экзаменуемого можно осуществить разными способами, ими может быть как обычный диалог, своего рода собеседование; письменный ответ, будь то контрольная, экзамен или зачет или другой вид аттестации. Но на данный момент, лучшим способом оценки, как корректным и непредвзятым, можно выделить тест. Тест - это совокупность специальным образом подготовленных и подобранных заданий, позволяющая провести выявление требуемых характеристик процесса обучения. Главным преимуществом тестирования является тот факт, что охват проверки обучающихся больше, чем при экзаменах и других способах. Тест позволяет меньше затратить времени на контроль уровня знаний, так как проверить можно сразу большое количество студентов, причем одновременно, не уделяя каждому отдельное внимание. Так же если первоначально провести какие-то исследования в плане оценивания, то можно подобрать, необходимы критерии оценки, а точнее какой-то порог, позволивший без какого-либо личного отношения к учащемуся, проверить его уровень знаний [1].

На сегодняшний день тестирование используется во многих сферах деятельности, таких как образование, медицина или возможность поступления на работу.

В связи с этим, все чаще любой контроль уровня знаний проходит за счет тестирования, а именно автоматизированных систем, которые позволяют, удалено провести проверку участника теста, не прибегая к участию преподавателя в проверки каких-либо текстовых работ, за него все это делает программа. Для этого необходимо лишь заранее составить алгоритм, самостоятельно определяющий, в каких вопросах были допущены неправильные ответы, а так же даже статистику выполнения всего теста за нужный период.

Так как данная работа относиться к системе тестирования, то необходимо провести оценку и самому способу проверки знаний. Данный вид испытания имеет ряд недостатков:

- Почти невозможно охватить весь теоретический материал, в тести данного направления. При устном экзамене, преподаватель может варьировать вопросы, когда как при тестировании будет уже заданный список.
- Не смотря на то, что созданной базой вопросов и вариантов ответов можно пользоваться долгое время, для ее создания затрачивается достаточное количества времени.
- При тестировании, есть такая возможность, что ответ может быть взять из каких-то источников, так как при прохождении теста в аудитории обычно находится вся группа, состоящая из большого числа студентов. Но так же данный недостаток можно решить, разделением группы, на отдельные подгруппы учащихся, которых в дальнейшем преподавателю будет удобно контролировать от списывания и переговоров.
- К сожалению, тестирование позволяет студенту лишь конкретно ответить на вопрос, не давая ему возможности учиться излагать свои мысли последовательно, приходить к каким-то умозаключениям, которые им бы могли помочь применять свои полученные знания в нестандартных ситуациях. Для тестирования необходима хорошая визуальная память, чаще

всего студент просто запоминает, зазубривает правильный ответ, даже не осмысливая его содержание.

- Тестирование позволяет просто оценить знания испытуемого, но никак заполнить пробелы. Для решения этой проблемы, необходимо дальше заниматься студентом. В устном варианте, преподаватель обычно исправляет студента, объясняя, в чем он ошибся, но это затрачивает много времени, особенно если это итоговый тест, и на проверку выделено определенное количество часов, которого просто может не хватить.
- При выполнении любого теста, есть возможность того, что студентов интуитивно или случайно может выбрать правильный ответ. Либо есть вариация оценивания может делиться на простой вопрос и сложный, студент, ошибившись в легком, может случайно ответить на более сложный, что повысит его общий балл. Элемент случайности характерен тестам невысокого уровня, позволяя студенту интуитивно дать нужный ответ.
- Так как тест состоит из установленного количества вопросов, студенту приходиться отвечать на все, хотя при устном ответе, преподаватель мог бы с помощью всего пару вопросов выяснить уровень знания материала [2].

Но если же попытаться исключить возможные недостатки тестирования, несмотря на выше перечисленное, то выявляется ряд преимуществ:

- Одно из главных преимуществ, то, что при тестировании, отсутствует какая-либо предвзятость к испытуемому, то есть, нет возможности субъективно оценивать ответ студента. Но так же очень сложно оценить успеваемость студентов, так как при тестировании всегда используются разные способы и методы, разная точность и передача информации. С оценкой познавательного прогресса студентов связан целый ряд проблем. Причиной тому является тот фактор, что часто студенты возмущены оцениванием педагогом их результатов, так как в любой ситуации играет фактор предвзятости. Так же преподаватель может быть

слишком требователен, что слегка усложняет оценивание. Помимо этого, даже в случае с использованием тестов, сами вопросы могут быть составлены неграмотно, не понятно для самих студентов, то есть его трактовка. В общем, оценивание знаний, зависит от многих факторов. Как симпатия и антипатия, так же студент может правильно ответить на большую часть вопросов, но это вызовет сомнение у преподавателя, так как за весь учебный год, тестируемый, не показывал хороших результатов. В общем, у каждого преподавателя разные требования, кто-то может принять материал, не задавая слишком много вопросов или же не столь строго принимая его, позволяя использовать какие-то недочеты в ответе. Кто-то наоборот, требует строгого выполнения и ответа на свой вопрос, чтоб термины были озвучены так, как это было в материале переданным на лекциях. Так или иначе, любой преподаватель должен стараться найти “золотую середину” в своих требованиях, сдача не должна проходить очень легко, но не быть непомерной. Поэтому, используя тестирование, процесс оценивания может обходить все вышеперечисленные факторы.

- Тест, как средство проверки, позволяет охватить знания по всем темам дисциплины, при условии, что список вопросов будет обдумано составлен. При устном же опросе, чаще всего, требуется полный ответ материала, но в ограниченной области. Тестирование позволяет определять знания учащегося по всему курсу, исключая элемент случайности при ответе на вопросы одного билета.
- Одним из важных плюс так же является тот факт, что тестирование может проходить одновременно большое количество учащихся, главное чтоб система тестирования была грамотно составлена и не ограничивалась системными ресурсами.
- Несмотря на то, что при создании тестирования, так или иначе затрачивается время и ресурсы, все равно оно является менее затратным способом проверки. Главное проблемой является создание грамотной и работоспособной системы, то есть программного обеспечения, которое

позволит проверить знания студентов. В то время, как затраты на проведение тестирование, менее затратный, чем тот же устный ответ или письменный. Так как система сама проверит и выявит недочеты в ответах студента, и не нужно отводить время на каждого, как в случае с устным ответом или проверять письменный вариант, затрачивая личное время преподавателя. Еще один преимуществом является то, тестирование можно проводить удалено, своего рода дистанционное управление. Чаще всего это используется при обучении заочников или аспирантов, когда нет возможности студентам присутствовать на занятиях [3].

Выделяются три основные взаимосвязанные функции тестирования:

- Диагностическая функция, позволяет оценить знания учащегося. Можно сказать, что данная функция является объективной и быстрой для проверки пройдённого курса.
- Воспитательная функция, помогает студенту понять, какие есть проблемы в его знаниях по материалу, который изучается и дальнейшем при желании восполнить их новой информацией.
- Обучающая функция, позволяют учащемуся воспользоваться дополнительным материалом (конспекты, интернет) при прохождении теста. Другими словами, студент в случае, когда не знает правильного ответа на вопрос или сомневается в своем варианте, может получить информацию извне, в то же время он восполняет сразу же пробелы в своих знаниях [4].

Не смотря на то, что есть определенные требование к проведению тестов, так же необходимо выделить и критерии правильного задания. При создании базы вопросов, нужно ответственно относиться к постановке и формулировки. Часто бывает так, что для количества вопросов, в связи с требованиями к учебному плану, преподаватель вынужден задавать один и тот же вопрос несколько раз, но с разной трактовкой. Это не правильно, так как при вероятности случайности, данный вопрос может отобразиться у студента несколько раз. Либо, есть вероятность, что поставленный вопрос будет не понятен для студента.

Ниже представлены главные требования к заданиям тестирования:

- Задание должно быть оформлено в виде четкого изложения вопроса, исключающего неоднозначность суждения участника тестирования.
- Все вопросы теста должны соответствовать требованиям ГОС.
- Содержание заданий ни в коем случае не должно содержать в себе сленг, двойное отрицание или повторы.
- Вопрос не должен основываться на мнении одного автора, то есть при выборе задания формулировка ответа на него должна встречаться часто в любых других источниках, чтоб испытуемый смог найти правильное суждение.
- Задание должно быть сформулировано в виде повествования.
- Ответ на вопросы не должен зависеть или быть схожим с другими, уже присутствующими в тесте.
- В тексте задания, а так же в самих ответах не должно содержаться ни малейшего намека на правильное суждение [5].

Тестирование может реализоваться с помощью разных видов вопросов. Если же в тесте будут реализованы все варианты, то это позволит как можно лучше выявить пробелы в знаниях и оценить их. Но эти требования к тестированию не обязательны, можно использовать всего один вид тестов, если это устраивает преподавателя и возможность проверки всего материала.

Виды заданий тестирования:

1. Задания закрытой формы. Данный метод заключается в том, что студенту необходимо из нескольких вариантов ответов выбрать один. В отведённом боксе тестирующийся сможет поставить галочку, выбирая нужный ответ, который как он считай, является правильным.

2. Задания открытой формы. Метод заключается в выборе ответа, который должен быть введен в форму письменно. То есть студенту необходимо правильный ответ ввести в отведенную форму, причем соблюдать правила написания этого слова или словосочетания в предложение. Если с данного ответа начинается термин, то оно пишется с

заглавной буквы. Если наличие ответа находиться посередине или в конце, то с маленькой буквы.

3. Задания на соответствие. Данный вариант, подразумевает, что необходимо найти варианты с разных столбцов и соединить их между собой в соответствии с каким-то правилами или принципами. Обычно ответы делятся на два столбца, причем справа чаще всего их бывает больше, чтобы в какой-то степени запутать тестируемого.

4. Задания на установления правильной последовательности. При таков виде вопроса, студенту необходимо выставить ответы по какой-то последовательности, друг за другом, причем должна иметься смысловая нагрузка [6].

В итоге, после того как студент пройдет тестирование, ответит на все варианты вопросов, как простые так и сложные, система высчитывает сколько было правильный ответов, после чего ведется подсчет и присуждается определенное количество баллов. При любом тестировании, обязательно устанавливается порог, то есть результат, при прохождении которого, студенту засчитывается тест как пройденный. Если тестирующийся проходит порог, получает оценку за тест, иначе ему будет предложено подучить материал, в другое время пройти этот же тест или другой снова. Так же преподаватель может на основе всех результатов выявить, где чаще всего студенты допускали ошибки, и возможно пересмотреть теоретический материал, дабы исключить дальнейшие пробелы.

1.2 Организация учебного процесса

При проведении любого лабораторного занятия, ориентированного на дисциплину по программированию, студенту необходимо для защиты выполнить лабораторную работу, которая делиться на несколько частей. А именно теоретическая и практическая части.

При защите лабораторного практикума, студенту необходимо пройти два этапа. Первый это ответить на вопросы касательно теоретической части, то есть студенту необходимо доказать, что он владеет всеми навыками и теорией создания программы требуемой в данной лабораторной работе.

Далее в зависимости от его ответа на теорию, студент допускается к защите и объяснению того, как создавалась его программа, какие методы использовались, как выполняется вычисление созданным алгоритмом.

Если не вносить какой-либо автоматизации в учебный процесс, то преподавателю приходиться лично проверять каждого студента, его теоретические знания материала. Затрачивается много времени на проверку всех студентов, а в случае если группа большая, то аудиторного времени может не хватить. Так же часто бывает, что защищающийся может плохо знать материал, и несколько раз пытаться сдать его, тем самым затормаживая процесс защиты других студентов. Именно поэтому целесообразно вводить в ход защиты лабораторного практикума автоматизацию, чтобы отведенное аудиторное время затрачивалось на защиту программной части, а не проверки одного и того же материала.

1.3 Необходимость автоматизации процесса контроля уровня знаний для лабораторного практикума

В настоящее время отношение к информационным технологиям несет разносторонний характер. Кто-то видит хорошую выгоду, для других же это дополнительная трата времени или иные расходы.

Основываясь на выводах в предыдущем пункте, для лучшей организации учебного процесса целесообразно использовать автоматизацию, которая заключается в создании определенной системы, позволяющей проверить знание теоретической базы у студентов без какого-то вмешательства преподавателя.

В результате проведения анализа на сегодняшний день учебный процесс проходит весьма не эффективно. Был проведен анализ одного из занятий, по дисциплине “Информатика и основы программирования”, в ходе которого были выявлены недостатки организации учебного процесса, а именно:

1. Некоторые студенты, пытались сдать теоретическую часть одной и той же лабораторной работы несколько раз, тем самым тратили аудиторное время, не давая другим студентам защититься;
2. Большая численность студентов в группе, так же не давала возможности всем желающим защитить лабораторную работу;
3. Помимо того, что преподавателю необходимо было проверить лабораторные работы студентов, многие просили помощи в выполнение программной реализации, поэтому процесс затормаживался.

В связи с этим, можно предположить, что при правильной автоматизации учебного процесса, можно в большей степени предотвратить вышеперечисленные недостатки. Не смотря на то, что при внедрении системы тестирования, как замену устного ответа на теоретическую часть лабораторного практикума, можно упростить лишь часть работы оценивания знаний студентов. Но уже, время будет отводиться лишь на проверку программной (практической) части задания, так же позволит отсеять ту часть студентов, которые не знают теорию лабораторной работы, а, следовательно, никак не могли выполнить задание сами. Так же, меньше будет затрачиваться время на защищающиеся, у которых имеются некоторые пробелы в знаниях теории. Не пройдя определенный порог, установленный системой тестирования, студенту автоматически будет предложено подучить материал, повторить тестирование снова и уже при положительном исходе, перейти ко второму этапу защиты.

1.4 Анализ существующих автоматизированных систем тестирования

На данный момент, в связи, что находимся в современном мире, существует много систем тестирования, так как они наиболее практичны и удобнее. Системы бывают двух видов: Windows-приложение и в виде веб-сайта. У каждой из этих систем, есть свои преимущества и недостатки, подробнее их рассмотрим ниже.

Для того чтобы проверить знания студента, можно воспользоваться несколькими видами испытания. Это устный ответ, дискуссия между преподавателем и студентов, либо письменный ответ на билет. Но из-за того, что для проверки обоих способов нужен преподаватель, то оценивание может растянуться на большой промежуток времени. Это связано с тем, что сначала преподавателю необходимо подготовить, создать шаблон билетов. После этого, на экзамене учащиеся должны на них ответить, причем обычно им отводиться на это больше времени, чем при обычном тестировании. Так как студент должен написать полный ответ на задания, чтобы была отчетность для кафедры, когда как в тесте, кратко поставлен вопрос, и нужно всего выбрать правильный ответ с помощью небскольких кликов. После того как студенты ответили на вопросы, преподавателю необходимо проверить весь материал, либо он будет проверять текст, который может быть неразборчиво написан, либо с каждым провести беседу. И после всего, преподаватель может поставить какую-либо оценку за данный ответ [7].

В связи с этим, любые обучающие учреждения переходят на автоматизацию данного процесса, используя его как при проверки обычных лабораторных работ или же проведения экзамена. Система тестирования облегчает работу оценки знаний, как проверяющему, так и упрощает работу студентов. При этом, устраняются проблемы с затратой времени на принятия теоретической части лабораторных работ, так как бывает что в один день

желают сдать работу сразу несколько студентов, либо по срокам им это просто необходимо, чтоб в дальнейшем получить положительную оценку.

Перейдем к отличиям между автоматизированными системами, а именно их структурой. Обе системы, пусть то windows-приложение или web-интерфейс, вся база вопросов, ответов, так же студентов и их результатов храниться в базе данных. Бывает, когда системы идут облегченного варианта, то есть, результат не должен храниться в системе, но такие системы редко когда пользуются спросом. Различие же программных продуктов заключается в том, что у них разные интерфейсы, по-разному выполняется доступность к ним.

- Windows-приложение. Данный вид систем, разрабатывается по средствам программных языков высшего уровня, с помощью прикладных программ. Одним из самых больших минусом то, что данный продукт может использоваться лишь на том компьютере, на котором он установлен. Есть возможность сделать расширенные настройки и возможность удаленного подключения к компьютеру и уже в дальнейшем входе в программу, но это не рационально. Но, не смотря, что это можно назвать как минусом, так же относиться данная реализация и к плюсу. Так как студент не сможет обмануть преподавателя и выполнить тест либо дома, когда есть возможность списать, либо попросить чтоб кто-то более подготовленный прошел его вместо него. То есть тест можно пройти только сейчас, только в данный момент и за этим компьютером. Но, не смотря на это, можно просто ужесточить контроль прохождения и студенту будет тяжело получить информацию извне, когда как в случае, если тестирование нужно пройти всей группе, а данный продукт установлен только на ограниченное количество компьютеров. То в это случаи придется ожидать завершения тестирование некоторыми студентами и смена их другими. К тому же, если же из-за нехватки места, попытаться установить ПО в данный момент, то без прав администратора, в учебных учреждениях невозможно будет установить программу. Но самым большим минусом, может являться тот факт, что если

же, данное приложение установлено не на кластер, то есть серверный компьютер, к которому будет у всех доступ и на нем же будет установлена база данных, в которой хранится вся информация, то возникает проблема синхронизации данных. То есть, после каждого тестирования, преподавателю придется либо проходить по всем компьютерам и проверять результаты, в дальнейшем если эти данные как то будут использоваться, придется вручную собирать данные и заносить в БД.

- Web-приложение. Является самой распространённой и удобной системой. В отличие от windows-приложения, данная система выполняется по средствам удаленного доступа. Прохождение любого теста будет выполняться посредствам онлайн входа на страницу. Любые данные, которые будут записываться на сайте, сразу же будут синхронизироваться базой данных. С помощью созданных алгоритмов, данный интерфейс, как и предыдущий, самостоятельно будет обрабатывать полученную информацию и автоматически выдавать результаты с последующим сохраняем или изменением в БД. Так же реализация самого тестирования так же проста, студенту необходимо зайти на сайт, авторизоваться, выбрать интересующий ему тест, если к нему есть доступ, то ответить на все вопрос и получить результат. В плане доступа со стороны администрации, то не нужно никаких дополнительных установок и документов. Минусом является тот факт, как описывалось выше, за студента могут удалено пройти тест или же он сам его может выполнить в другой момент, но данный недочет так же решаем, то есть позволить увидеть тест студенту лишь в определенный день и время [8].

Поэтому проведя анализ существующих способов реализации автоматизированных систем, лучшим вариантом, по-моему, мнению это интернет-приложение. Так как разрабатываемая система будет использоваться в тестировании теоретической части лабораторной программы, где необходимо проверить узкую часть материала по каждому разделу.

1.5 Формализация задач обработки информации

Несмотря на то, что системы тестирования просты и удобны в использовании, их создание весьма долгий и сложный процесс. В случае, если система должна быть полностью автоматизирована, полна и правильна.

Чтобы перейти к разработке программного обеспечения, необходимо выделить некоторые задачи:

1. Подготовка. Для начала необходимо определить, что нужно от системы, какого плана она будет, что именно должно выполняться в ней. Так же изучить уже существующие системы тестирования, посмотреть их плюсы и минусы, попытаться выделить все самое важное, чтоб в дальнейшем реализовать в своем продукте. Учесть все необходимые составляющие проведения учебного занятия и структуры проверки знаний студентов. Сюда также будут включены разработка и составляющая часть всех тестов, их структура и значимость. Определить то, в каком виде будет представлен сайт, его дизайн, его техническую составляющую.
2. Выбор домена. Необходимо ответственно подойти к выбору имени сайта, чтоб было просто и запоминающийся, позже это название можно использовать в адресе.
3. Разработка структуры сайта. На этом этапе выделяются вопросы подготовки разработки, то есть, намечается, каков примерно будет дизайн, создание информационных блоков переходов и навигации по другим страницам, а так же первоначальное наполнение информацией. Своего рода идет выполнение чернового варианта продукта, реализуется примерное представление о ПО, так же может какие-то сторонние требования от заказчика, если таковой имеется. Главной задачей этого этапа является раскрыть все задумки и постараться их представить в черновом варианте.
4. Дизайн сайта. Дизайн это важная составляющая любого сайта, так как если в оформлении будут содержаться резкие цвета или неприятные исходники, то человеку будет неприятно и не интересно находиться на

данном сайте. Необходимо так же выделить свою стилистику, постараться предвести индивидуальность, чтоб сайт был запоминающимся. Так же необходимо подготовить весь материал, который будет использоваться в оформлении, составить макет того, как все будет расположено на каждых из страниц и в какой гамме. Для этого нужно проявить фантазию, возможно, где-то позаимствовать идеи, но не копировать полностью, так как этот тоже может оттолкнуть от продукта, особенно если исходный сайт весьма известен. Для реализации всего этого не обойтись без графических редакторов.

5. Разработка сайта. А именно его реализация по средствам программного года, позволяющего выполнить применение всего функционала тестирования, внешней оболочки и вывода каких-либо результатов. Так же необходимо применение различного способа доступа.

6. Проведение тестирования. Главное на этом этапе, это проверить работоспособность сайта. То есть запустить на возможных платформах, если это веб-сайт, то обязательно проверить, как отображается весь дизайн в разных браузерах. Проверить заполнение и сохранение всех данных на сайте и базе данных. Проверить работоспособных всех алгоритмов, которые присутствуют на сайте. После того, как проверка не показала никаких ошибок, сайт можно размещать.

7. Контент. Наполнение сайта информацией (текст, картинки, тесты и т.д) [9].

Все выше перечисленные требования ориентированы на создание любого web-интерфейса (сайта), но для выполнения данной работы необходимо выделить так же специализированные указания, а именно:

— Создание возможности регистрироваться в системе и автоматизации существующего пользователя. Причем возможность добавления нового пользователя разрешено только администратору.

— Создать возможность внесения неограниченного списка вопросов. Лучше всего это реализовывать по средствам базы данных.

Вопросы будут закрытого вида. Реализовать систему случайности, чтоб вопросы и ответы появлялись в случайном порядке, чтобы исключить возможность повторения и студенту было сложнее было запомнить ответы, в случае повторного прохождения того же теста. Для удобства, реализовать возврат к предыдущим вопросам, в случае если студент на данный момент сомневается, чтобы не тратить лишнее время, он выполнит другие задания, а позже сумеет вернуться к прошлому. Необходимо так же установить порог, для первого тестирования можно использовать примерный. Позже с помощью методов исследования данных установить другой, причем при большем количестве тестирований, порог будет варьироваться.

— Возможность изменения или добавления базы данных со стороны интернет приложение. То есть, администратору не обязательно переходить в СУБД, где кроме разработчика сайта, преподаватель может не разобраться. Произвести заполнение можно со стороны сайта, где необходимо удобно и просто для понимания реализовать алгоритм ввода данных, помимо этого так же удаление, то есть все возможные действия, которые выполняет СУБД.

— Наличие разной доступности к сайту и функционалу разному виду пользователей, а именно гость, студент и администратор/преподаватель.

— Создание алгоритма сохранения статистики прохождения. То есть при выполнении теста, при нажатии завершения теста, алгоритм не только отображает результат самому студенту, но и заносит его в систему. В результате в отдельный блок БД будут заноситься данные о студенте, его результат, тест, который был пройден и дата прохождения.

— Так же дополнительно реализовать возможность со стороны сайта выводить данные обо всех результатах в отдельный Exsel файл.

— Реализовать возможность создателю сайта/преподавателю отслеживать посещаемость сайта. Это необходимо в случае подозрения, что был выполнен вход и тестирование с аудитории, а с другого IP-адреса, а так же на какую страницу данный человек переходил.

Для того чтобы реализовать все поставленные задачи, была выбрана специальная среда WampServer версии 2.4, позволяющая создавать локальный сервер без настроек своего компьютера. Помимо этого, программа имеет уже встроенную в себя СУБД MySql 5.6.12, так же средства для работы и отображением технологий написанных на языке php PHP 5.4.12, а так же сам сервер отображение интернет страниц Apache 2.4.4.

2 ПРОЕКТИРОВАНИЕ

2.1 Методы создания сайта

Создание любого сайта можно произвести помощью двух основных методов. Первая группа содержит в себе методы ручного написания с помощью веб-языков, причем можно использовать один или несколько. Для реализации обычно используются простые текстовые редакторы, такие как блокнот, либо в визуальных HTML и CSS. Каскадные стилевые таблицы дают возможность создать сайты в режиме WYSIWYG – «Что Вижу То и Получаю».

Сайт может двух видов, статического, то есть не содержащих каких-либо алгоритмов обработки и работы с данными с последующей возможностью сохранения. Для создания такого сайта обычно используют первоначальные языки «связки» HTML и CSS, а так же подключение JavaScript. Так же существует динамический вариант, когда на сайте обрабатывается какая-то информация, возможность синхронизации с базой данных, а так же выполнения каких-либо запросов. Для такого создания используются дополнительные языки, помимо уже сказанных, такие как PHP, ASP.NET и т.д.

При выборе простого создания сайта, без использования Фреймворка, разработчику достаточно иметь при себе программу “Блокнот”, это стандартный продукт в системе Windows. Если же сайт будет использовать в себе ASP.NET, для этого необходимо будет установить стороннее платное обеспечение Microsoft Visual Studio.

Если написание сайта с помощью “ручного” метода, то и сам дизайн так же выполняется вручную. Чаще всего для этого используют графические редакторы, но и без них можно выполнить графическое

оформление. Так же вручную можно провести редактирование готового шаблона дизайна [10].

Ко второй группе методов разработки сайта относиться автоматизация по средствам фреймворков, то есть специальных конструкторов или систем управления контентом (CMS).

Вторая группа методов создания сайтов включает в себя методы автоматизированного создания сайтов: при помощи специальных конструкторов сайтов или же систем управления контентом (CMS).

Конструкторы сайтов – это, как правило, онлайн-системы, позволяющие из готового типового набора модулей и компонентов "собрать" сайт и сразу же разместить его на сервере.

CMS это программное средство содержащие в себе готовую визуальную и программную оболочку, которая упрощает создание сайтов, то есть уже существует созданная разметка и пользователю остается дополнить ее необходимой информацией или контентом, либо же изменить стилистику блоков и их расположение.

Данные автоматизированные методы разделяют создание сайта на «дизайн» и «контент». Это в большей степени упрощает написание сайта, так как пользователь может не затрагивая дизайн изменить какие-то недочеты в начинке сайта, в его контенте, так и наоборот. При создании сайта с помощью ручного способа, разделения между дизайном и контентом не происходит, то есть весь программный код, касающийся данной страницы, находиться в одном файле [11].

При рассмотрении двух методов можно сказать, что ручное написание сайта весьма трудоемко и сложно, так как для этого необходимо обладать хорошими знаниями в области веб-программирования, а так же графического оформления. Но, несмотря на это, при создании ручным способом, программист точно может реализовать свои идеи и получить в результате то, что он планировал.

Поэтому чаще всего, сайты пишутся по средствам ручного ввода, без использования каких-то сторонних продуктов.

Для новичков в среде веб-программирования удобнее и проще будет создавать с помощью бесплатных онлайн конструкторов, так как они еще только обучаются и изучают структуру программирования. Поэтому фреймворки подходят для разработки небольшого проекта, который выделяется простотой.

Больше функционала в себе содержит CMS, позволяющий создавать сайты разной сложности и нагрузки на контент. Поэтому, именно данный способ считается самым удобным и распространенным. Расширенный и гибкий функционал, возможность редактирования самих блоков CMS или полностью данного обеспечения, добавление и изменение контента, то есть все намного упрощает работу с разработкой сайтов, что является эффективнее [12].

2.2 Модульная схема программного продукта

Прежде чем перейти к программированию, необходимо создать структуру сайта, то есть наметить, как должен будет выглядеть сайт, как информация будет заполнена в нем и то, какая связка будет между страницами. Все это необходимо для того, чтоб на этапе разработки, было легче создавать сайт.

То как должен будет разбит и классифицирован сайт, полностью зависит от того, какой информацией он будет заполнен. Необходимо тщательно проработать стратегию классификации, так как для разных людей, в зависимость от прав доступа, будет допуск к разной информации. Классификация сайта должна быть проста в использовании и понятно, какая страница, за какой может следовать.

Для того чтобы проще было работать и отобразить классификацию файлов сайта, создается модульная схема, позволяющая определить связь между файлами и страница системы. Прежде чем перейти к созданию такой схемы, нужно определиться с функционалом, что будет на сайте, что для реализации необходимо и то, как все будет работать.

При создании любого сайта, первостепенно задается главная страница, на которой будут отображены все ссылки, переходы на другие страницы, а именно второстепенные файлы, с основным функционалом. Так же обычно уделяют больше внимание дизайну, так как это своего рода “лицо” сайта, на котором храниться вся необходимая информация о том, куда пользователь зашел, а так же пояснение к дальнейшим действиям. В данной работе данный файл называется index.php.

После того, как пользователь вошел на сайт, и ему отобразилась главная страница, ему будет предложено перейти в два второстепенных раздела, это тестирование, которое предназначается в основном студентов для прохождения теста, и администрация, где создатель сайта или преподаватель смогут работать с базой данных, вносить в нее изменения и просматривать. Для того чтобы получить доступ к обоим блокам, двум видам пользователям необходимо пройти авторизацию. Для одного это файл open.php, для администрации файл enter.php. После ввода данных, будет выполнен автоматический переход на страницу с функционалом.

В случае со студентом, авторизация проходит сложнее, сначала срабатывает скрипт, файл testreg.php, который проверит логин и пароль нового пользователя, удостовериться в его личности и перенаправит на необходимую страницу, причем данные введённые студентом, будут проверяться с данными хранящимися в базе данных. Если не будет не единого соответствие, в доступе будет отказано.

На сайте будет иметься возможности манипулировать с базой данных со стороны интернет-интерфейса. С помощью файлов testbd.php,

questionbd.php, answerbd.php, statisticbd.php, пользователь с правами администратора сможет добавлять, изменять, удалять и выполнять поиск информации. Каждому файлу соответствует отдельная часть базы данных, то есть файл questionbd.php отвечает за работу с данными связанными с вопросами и так далее.

Отображение все вышеописанных связок, ниже представлена диаграмма, которая позволяет отобразить то как каждые файлы связанны между собой (рисунок 2.1).

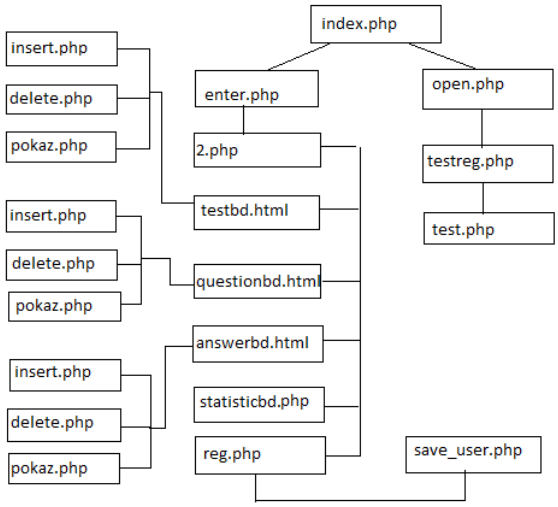


Рис. 2.1 Модульная схема системы тестирования

2.3 Модель сайта

При создании любого сайта, если же он не является обычным информативным ресурсом или рекламой, необходимо создание разделения прав пользователей. Для каждого из посетителей должны быть доступны разные виды информации. Обычно выделяют 3 группы пользователей:

1) Гость;
2) Зарегистрированный пользователь (студент);
3) Администратор (преподаватель/создатель сайта).

Меньше всего прав на сайте это у гостя. Его возможности ограничиваются лишь в просмотре главной страницы, оформления и пояснения к системе тестирования. Для того чтобы получить доступ какой-либо другой информации, а именно прохождении теста, необходимо стать зарегистрированным пользователем. Права и регистрацию ему может выдать лишь администратор.

Зарегистрированный пользователь выступает в роли студента, то есть трестирующегося, который находиться в базе данных, и ему позволено проходить все тесты и в дальнейшем. Его возможности ограничиваются лишь разделом тестирование, где он может просматривать наличие тестов, проходить их, если к данному тесту есть доступ и сняты ограничения.

Администратор, это своего рода преподаватель или же создатель/модератор сайта. У данного вида правообладателя самые расширенные возможности, это просмотр тестов и так далее. Изменение, добавление данных и так далее.

2.4 Физическая модель базы данных

Для того чтобы автоматизированная система могла обрабатывать множество данных, а так же не изменялась в структуре со стороны сайта, необходимо спроектировать и разработать полноценную, простую и в то же время полную базу данных. В данной работе, БД содержит в себе шесть сущностей, связанных между собой.

Разрабатываемая система, служит инструментов проверки знаний студентов, для этого в первую очередь необходимо выделить таблицу “users”, которая предназначена для хранения данных о каждом студенте. Для контроля уровня знаний, необходима таблица, которая бы позволяла хранить все необходимые тесты, то есть таблица “test”. Для того, чтобы было легче работать с вопросами и ответами тестирования, для удобства обе таблицы, а

именно “ questions ” и “ answers ”, будут созданы по отдельности, что позволит точнее контролировать правильность вводимых данных.

Когда студент пройдет тестирование, для возможности просмотреть результат в дальнейшем, необходимо сохранить все данные в БД, поэтому создается таблица “statistic”. В таблицу будут заноситься все данные касающиеся самого теста, студента, а так же время когда был выполнен тест и процент верности ответов. Так же для удобства отслеживания входов на сайт, можно создать отдельную сущность, позволяющую заносить данные о браузере, с которого был вход и дату запроса.

Для полного и подробного описания каждых из сущностей, ниже будет представлена структура физической модели.

Структура физической модели данных показана в таблицах 2.1- 2.5

Таблица 2.1

Структура таблицы USERS

Id*	login	password	name	firstname	group
Код пользователя	Логин пользователя	Пароль	Имя	Фамилия	Группа
Int	Varchar(30)	Varchar(30)	Varchar(30)	Varchar(30)	Varchar(10)

Данная сущность представляет собой пользователя сайта, а именно зарегистрированного студента, который в дальнейшем будет иметь доступ к тестам. В столбцах логин и пароль, будут храниться данные необходимые для авторизации и проверки пользователя. В случае с отчетностью, необходимы и данные об имени, фамилии и группе в которой студент обучается. Код пользователя в данной таблице представляет собой первичный ключ, уникальный идентификатор того или иного студента, который в дальнейшем будет связан со многими таблицами, а так же позволит выполнять обращение с данными, ее связку. При добавлении нового пользователя, данный счетчик будет увеличиваться на 1.

Таблица 2.2

Структура таблицы TEST

Id*	Test_name	enable
Код теста	Название теста	Активность теста
Int	Varchar(50)	Enum(“1”,”0”)

Данная сущность будет содержать в себе данные касающиеся тестов. То есть, код теста, это первичный ключ, уникальный идентификатор, по средствам которого будет взаимодействие со всей базой данных. Название теста, это название лабораторной работы, теорию по которой студенту необходимо защитить. Активность теста, это своего рода доступность тестирования студенту. Если данный тест необходимо скрыть для любого пользователя, необходимо присвоить значение 0. И наоборот, если же студент может выполнить защиту лабораторной работы, то администратору необходимо сменить данное значение на 1.

Таблица 2.3

Структура таблицы QUESTIONS

Id*	question	Parent_test
Код вопроса	Полное название вопроса	Код теста
Int	Varchar(50)	int

Данная сущность отвечает за хранение данных, касающихся к вопросам тестирования. Код, это идентификатор вопроса, данный параметр с добавлением новой записи увеличивается на 1. Название вопроса, это сама формулировка, которая будет отображаться у студента на экране при прохождении теста. Код теста, внешний ключ, то есть номер теста, к которому относиться вопрос. При выполнении запроса, а именно при выборе теста, по данному параметру будет проходить выгрузка вопросов теста. На экране будут показаны лишь те вопросы, которые относятся к тому тесту и содержат в третьем столбце именно его номер.

Таблица 2.4

Структура таблицы ANSWERS

Id*	answer	Parent_question	Correct_answer
Код ответа	Название вопроса	Код вопроса	Правильность ответа
Int	Varchar(50)	int	Enum(“1”,”0”)

Данная таблица хранит в себе данные, относящиеся к ответам тестирования. Код ответа, первичный ключ, параметр увеличивается на 1 при дальнейшем добавлении записей. Код вопроса, внешний ключ, по средствам данного параметра, запрос со стороны сайта, позволит алгоритму вытянуть из базы данных лишь те ответы, которые относятся к вопросу с необходимым номером. Последний столбец отвечает за правильность данного ответа. Если параметр равен 1, то при выборе его, студенту будет засчитан ответ как правильный, в случае с 0, ответ будет неверен.

Таблица 2.5

Структура таблицы STATISTIC

Id*	Id_user	Id_test	Percent	data
Код статистики	Код пользователя	Код теста	Результат	Дата прохождения
Int	int	int	Varchar(10)	Varchar(30)

Сущность содержит в себе данные, которые относятся к результатам тестирования, которые послужат статистикой прохождения теста. Код статистики, первичный ключ, уникальный идентификатор, который увеличивается на 1, при создании новой записи. Код пользователя, внешний ключ, параметр позволяющий связать две таблицы, а именно данную и пользователя, то есть вытянуть в дальнейшем все данные касательно пользователя. Код теста, так же связывает таблицы, а именно статистику и тесты, с последующей возможностью получения данных. Так же два последних столбца это результат, то есть процент правильных ответов, и дата когда студент выполнил тестирование.

2.5 Диаграмма модели «сущность-связь»

Для того чтобы база данных правильно функционировала, и при заполнении многочисленными данными не давала сбой, необходимо ее нормализовать. Для наилучшего понимания структуры базы данных, строится ее модель “сущность-связь”, но прежде чем перейти к ее выполнению, необходимо привести базу к нормальной форме Бойса-Кодда. Нормализировать базу данных можно несколькими способами, но в данной работе выполняется с помощью данного метода. Форма Бойса-Кодда часто так же называется усиленной третьей нормальной формой. Нормализация заключается в том, чтобы отношения были приведены к нормальной форме, в том случае, если детерминанты всех ее зависимостей являются потенциальными ключами [13].

База данных почти находиться во второй нормальной форме, ошибка лишь в таблице “Test”(рисунок 2.1а), в которой параметр “Answer” зависит от параметра “Question”, который в свою очередь является частью составного ключа. Поэтому может появиться ряд проблем:

- Внесение данных (в случае если вопрос не будет отображен, то нельзя будет выявить на него ответ).
- Удаление данных (если в случае необходимости удалить тест из БД, то это понесет потерю данных в вопросе, а, следовательно, и в дальнейших ответах).

- Изменение данных (если приходиться изменить ответ, то необходимо произвести поиск по множеству записей, чтоб в случае тестирования исключить возможность появления неверного результата).

Поэтому необходимо привести отношение “Test” ко 2НФ. Для того чтобы сделать это преобразование, нужно:

1. Построить проекцию без атрибутов, которые находиться в частичной зависимости между собой, от составного ключа. В данном примере это проекции – Test= Test[Test,Question].

2. Построить проекцию на составные ключи и атрибуты зависимости Question=Test[Question,Answer].

После соблюдения вышеперечисленных действий, данная связь разбивается на два Test и Question (рисунок 2.2б). Для того, чтобы было проще обращаться к данным таблиц, был создан для каждой из них свой идентификатор, то есть первичный ключ.

Такая же ситуация возникает и в случае отношения между “Question” и “Answer”. Поэтому, необходимо выполнить такие же действия в случае как с тестами и вопросами, и привести к нормальному виду. В итоге на рисунке 2.2в изображен конечный вариант приведения.

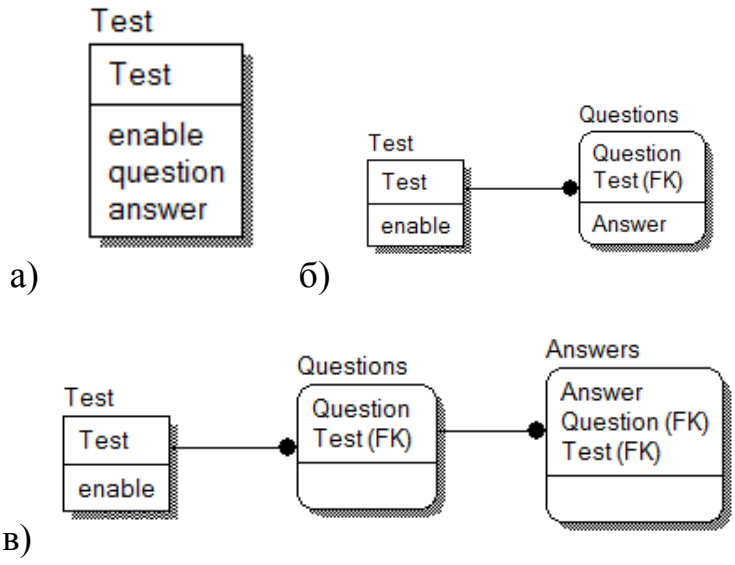


Рис. 2.2 а) Вид таблицы “Тест” базы данных

б) Вид таблицы базы данных 2НФ

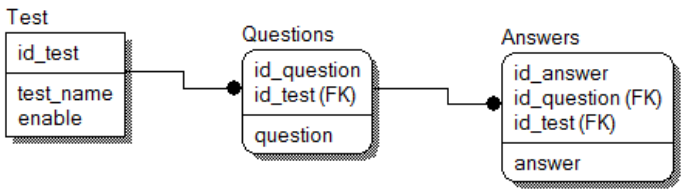
в) Конечный результат приведения ко 2НФ

Для упрощения нормализации отношений (рисунок 2.2 в) целесообразно использовать следующие рекомендации – при проведении нормализации отношений, в которых присутствует цифровые заменители составных и тестовых первичных и внешних ключей, необходимо подменять их на исходные ключи, а после окончания преобразования восстанавливать.

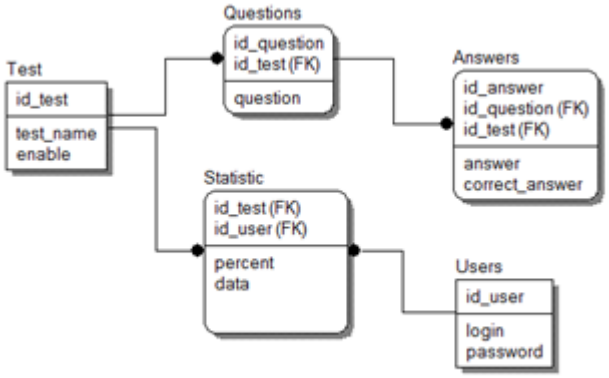
В связи с этим, используя вышеописанные рекомендации, необходимо проектирование базы данных представленного на рисунке 2.2в, привести к 3 нормальной форме. Поэтому потребуется создать новый параметр, а именно первичный ключ, который с добавлением новой записи, будет автоматически увеличиваться. Так же он будет для каждой новой записи уникален. Для большего удобства и понятия новому администратору/преподавателю, название данного параметра можно присваивать сочетание id и названия таблицы, к которому он принадлежит (рисунок 2.3 а).

В итоге, приведя исходную структуру базы данных к нормальному виду, по средствам нормализации Бойса-Кодда, можно выполнить постройку диаграммы, которая сможет отразить полный вид того, как выглядит БД и то какие взаимосвязи между сущностями.

Диаграмма, показывающая связи между таблицами изображена на рисунке 2.3 б.



а)



б)

Рис. 2.3 а) Результат приведения таблицы к 3НФ
б) Модель «сущность-связь»

2.6 Список объектов и их свойств

Предметная область автоматизированной системы тестирования включает в себя следующие предметы и их свойства.

1. Список пользователей. Поля:

- Код пользователя
- Логин
- Пароль

2. Список тестов. Поля:

- Код теста
- Название теста
- Активность (доступ к прохождению)

3. Список вопросов. Поля:

- Код вопроса
- Название вопроса (формулировка)
- Код теста

4. Список ответов. Поля:

- Код ответа
- Название ответа (формулировка)
- Код вопроса
- Правильность ответа

5. Список результатов (статистика) Поля:

- Код статистики
- Код пользователя
- Код теста
- Процент прохождения теста
- Дата прохождения теста

3 РЕАЛИЗАЦИЯ ПРОГРАММНОГО ПРОДУКТА

3.1 Создание модуля регистрации

Для любой полноценной автоматизированный системе необходима возможность регистрировать пользователя, для того чтобы у него были расширенные права на сайте. Зарегистрированный пользователь данной системы, пройдя авторизацию, получается возможность пройти любой доступный в данный момент тест.

Для того чтобы добавить/зарегистрировать пользователя, может лишь только администратор. Добавление может проходить, как и со стороны СУБД, но для более удобного пользования разрабатывается программный модуль со стороны сайта, который позволит внести нового пользователя со стороны веб-интерфейса. Данный модуль содержит в себе несколько отдельных алгоритмов, файлов, которые позволят внести данные о новом пользователе в БД, сохранить данную информацию, выполнить необходимую проверку (при переходе на страницу с ограниченными правами) и конечный перевод в нужную часть сайта.

Файл reg.php, позволяет внести нового пользователя в БД, то есть зарегистрировать его. Для этого, администратору сайта, необходимо внести все имеющиеся данные о пользователе в специально отведенную форму. Данные поля именуются логин и пароль, дополнительные данные можно

будет внести в СУБД. После того, как будет нажата кнопка регистрации, данные передадутся скрипту страницы на обработку (листинг 3.1) [14].

Листинг 3.1 Файл reg.php

```
<h2>Регистрация</h2>
   <form action="save_user.php" method="post">
  <p>
   <label>Ваш логин:<br></label>
   <input name="login" type="text" size="15" maxlength="15">
   </p>
<p> <label>Ваш пароль:<br></label>
<input name="password" type="password" size="15" maxlength="15"></p>
<p><input type="submit" name="submit" value="Зарегистрироваться"> </p></form>
```

Все данные, переданные в файл, отправляются методом post на обработку, а именно в файл save_user.php. Который отвечает за сохранение всей информации о пользователе в базу данных. Все данные сохраняются в специальные локальные переменные. Бывает, что происходит какой то сбой или неправильный ввод данных, то в этом случае система оповещает огб ошибке и прерывает сохранение. Если вся необходимая информация была введена в форму, то происходит ее обработка. Алгоритм автоматические удаляет все лишние пробелы, то есть лишние символы в начале слова или в его конце. После всех проверок происходит подключение к базе данных с помощью SQL запросов, которые проверяют, существует ли данный введённый логин в базе. Если данный логин уже существует в системе, то на

экране отобразиться ошибка о невозможности использовать данный логин и просьба использовать другой. При выполнении всех условий и отсутствии введённого логина в системе, база данных принимает данные и записывает их в соответствующей таблице. Не смотря на то, что выполняются все условия для правильной регистрации пользователя, бывают и системные сбой. В этом случае алгоритм автоматически проверяет запрос и в случае чего отображает ошибку. В двух случаях будет выведен тестовый результат (листинг 3.2) [15].

Листинг 3.2 Файл save_user.php

```
<?php
if (isset($_POST['login'])) { $login = $_POST['login']; if ($login == '') { unset($login);}}
if (isset($_POST['password'])) { $password=$_POST['password']; if ($password =='') { unset($password);} }
if (empty($login) or empty($password))
{exit ("Вы ввели не всю информацию, венитесь назад и заполните все поля!");}
$login = stripslashes($login); $login = htmlspecialchars($login);
$password = stripslashes($password); $password = htmlspecialchars($password);
$login = trim($login);
$password = trim($password);
include ("bd.php"); $result = mysql_query("SELECT id_user FROM users WHERE login='$login'");
$myrow = mysql_fetch_array($result);
if (!empty($myrow['id_user'])) {
exit ("Извините, введённый вами логин уже зарегистрирован. Введите другой логин.");}
$result2 = mysql_query ("INSERT INTO users (login,password) VALUES('$login','$password')");
if ($result2=='TRUE')
{echo "Вы успешно зарегистрированы! Теперь данному профилю открыт доступ.
<a href='2.php'>Админка!</a>";}
else {echo "Ошибка! Вы не зарегистрированы.";}?>
```

Если регистрация была выполнена, пользователь, а в случае данной системы студент, может пройти тестирование. В системе используется контроль входа пользователей в систему, все выполняется по средствам сессии, которая запускается сначала авторизации и хранит в себе переменные входа. При таком подходе, пользователь может выполнять тестирование, пока не завершит сессию, а именно не выполнит выход, нажав на соответствующую кнопку. Для такого контроля, создается отдельный файл, позволяющий, проверить присутствует ли пользователь в системы, то есть находится в онлайн.

Для данного функционала в структуре сайта существует файл testreg.php. Сначала алгоритм выполняет подключение к сессии, которая позволяет хранить все необходимые данные об авторизовавшемся пользователе. После этого данные переходят в соответствующие переменные. Далее проходит проверка данных введённых в форму для авторизации и информации хранящейся в базе данных. Если переданных данных алгоритму в БД не существует, то пользователю выдается ошибка с просьбой проверить введенную информацию. Если данные присутствуют в БД, то происходит обработка информации, то есть автоматически редактируется текст, проверяется и удаляются лишние пробелы. После этого происходит подключение к самой БД, то есть запускается часть алгоритма связанного с ней. Из базы вытягиваются все данные, которые соответствуют данному логину. Дальше происходит проверка паролей, между тем который был введен в форму и тем, который сохранен в базе данных с данным логином. В этом случае тоже есть два пути, если все данные верны, то для пользователя запускается сессия входа и ему появляется доступ к тестам, если же правила не соблюдены, то появляется ошибка (см. приложение, листинг 1) [16].

3.2 Создание среды тестирования

В данном пункте будет описываться главный функционал тестирования, который позволит трестирующемуся пройти тест(теоретическую часть) своей лабораторной работы. Для реализации такой возможности необходимо наличие страницы, где будут отображаться тесты, вопросы, варианты ответов. Так же для реализации полноты функционала необходима база данных с возможностью хранения информации по тестам и возможность записи в БД результатов тестирования. Чтоб структура тестирования была удобна для дальнейшего редактирования, каждые важные алгоритмы собраны в отдельных файлах. Для создания алгоритмов автоматизации использовались технологии PHP, JavaScript и Ajax.

Для начала рассмотрим главную страницу, то есть страница, которая отображается выбор тестов доступных в данный момент. Так как система тестирования предполагает в себе различие прав доступа, то на главной странице прописан код проверки запущенной сессии. Если пользователь не был авторизован в системе, то ему не будет отображаться список тестов, а следовательно студент не сможет пройти тест (Листинг 3.3) [17].

Листинг 3.3 Проверка авторизации пользователя

```
session_start();
if(@$_GET['exit'] == 'logout'){
unset($_SESSION['login']);
session_destroy();}
if(!$_SESSION['login']){
header("Location: open.php");
exit;}
```

Как только пользователь авторизовался и получил доступ к страницы с тестами, то он может выбрать необходимую лабораторную работу и пройти по ней тестирование. При нажатии на название теста, вызывается обработчик, который позволяет выгрузить вопросы и выбор ответов

касающегося именно данной лабораторной работы. Студент выбирает нужные ответы по его мнению , нажимая на окошко выбора, для перехода к другому вопросу необходимо просто нажать на его номер. После того как все ответы были выбраны, студенту необходимо завершить тест нажатием на отведенную кнопку под вопросами. Вся информация автоматически передается алгоритму, который передает данные в базу данных и заносит их в отведенную таблицу, касающуюся результатов. Благодаря данной реализации, преподаватель может в любое время просмотреть результаты всех студентов (Листинг 3.4) [18].

Весь главный функционал, отвечающий за вывод тестирования, находиться в отдельном файле php, в нем хранятся обращения к базе данных и связь между ней и страницей.

Листинг 3.4 Пример вывода существующих тестов

```
function get_tests(){
        global $db;
        $query = "SELECT * FROM test WHERE enable = '1'";
        $res = mysqli_query($db, $query);
        if(!$res) return false;  $data = array();
        while($row = mysqli_fetch_assoc($res)){$data[] = $row;}return $data;}
```

Вышеописанный код реализует sql запрос, который позволяет подключиться к базе данных и получить информацию о всех тестах, которые существуют в базе данных и доступны для прохождения. Тесты выгружаются по названиям, поэтому студенту необходимо лишь выбрать интересующий его тест (рисунок 3.1).

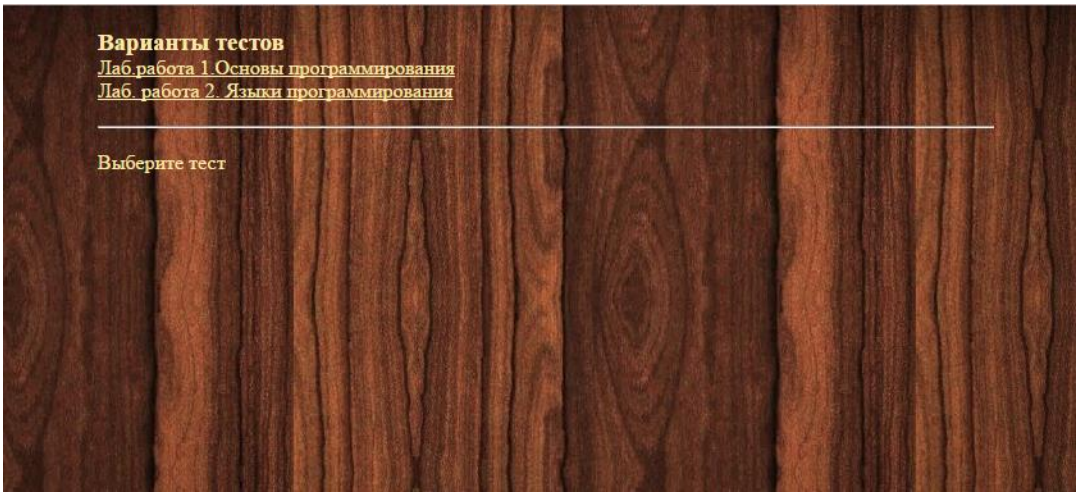


Рис. 3.1 Пример вывода списка тестов

Как только студент выбрал нужный ему тест, обработчик выгружает данные из базы данных, а именно вопросы и ответы, которые имеют значение внешнего ключа совпадающим с идентификатором выбранной лабораторной работы. В связи с тем, что данные необходимые для тестирования хранятся в разных сущностях базы данных, необходим сложный запрос, который позволяет вытянуть данные из всех нужных таблиц и объединить их (Листинг 3.5) [19].

Листинг 3.5 Выгрузка данных теста

```
function get_test_data($test_id){
if( !$test_id ) return;
global $db;
$query = "SELECT q.question, q.id_test, a.id_answer, a.answer, a.id_question
        FROM questions q LEFT JOIN answers a
ON q.id_question = a.id_question
        LEFT JOIN test
                ON test.id_test = q.id_test     WHERE q.id_test = $test_id AND test.enable = '1'
ORDER BY RAND()";
        $res = mysqli_query($db, $query);$data = null;
        while($row = mysqli_fetch_assoc($res)){
                if( !$row['id_question'] ) return false;
                $data[$row['id_question']][0] = $row['question'];
```

```
            $data[$row['id_question']][$row['id_answer']] = $row['answer'];}
    return $data;}
```

Данный файл содержит запрос, который вытягивает данные из базы данных, а именно: вопрос, номер теста, номер ответа, ответ и номер вопроса. Дальше идет расшифровка псевдонима, далее указывается, какая связка идет между разными сущностями, данные о которых мы пытаемся взять из БД. Как только произошла выгрузка данных, выполняется заполнение массива данными, которые были получены запросом. Первое $data[$row['id_question']][0], это сам вопрос, далее это ответы на поставленное задание. Результат выполнения кода представлен на рисунке 3.2.

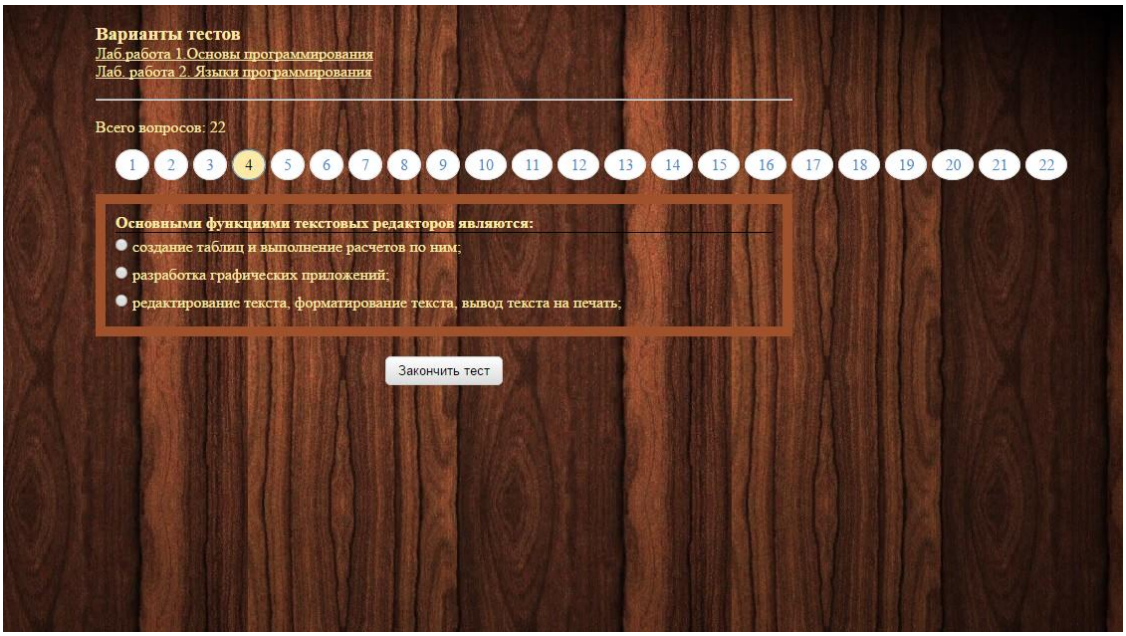


Рис. 3.2 Пример вывода вопросов и ответов необходимого теста

Как только вопросы отобразились на экране, появляется необходимость выполнить функцию проверки правильного распознания ответов к заданным вопросам. То есть все выбранные ответы будут сверяться с верными и определяться происходит совпадение или нет (Листинг 3.6) [20].

Листинг 3.6 Определение правильности ответа на вопрос

```
function get_correct_answers($test){ if( !$test ) return false;global $db;
    $query = "SELECT q.id_question AS question_id, a.id_answer AS answer_id
        FROM questions q     LEFT JOIN answers a  ON q.id_question = a.id_question
        LEFT JOIN test ON test.id_test = q.id_test
    WHERE q.id_test = $test AND a.correct_answer = '1' AND test.enable = '1'";
    $res = mysqli_query($db, $query);$data = null;
    while($row = mysqli_fetch_assoc($res)){$data[$row['question_id']] = $row['answer_id'];
    }return $data;}
```

Данный функционал позволяет вернуть значение вопроса и ответа. После этого происходит проверка связей между таблицами. При обычном запросе, будут возвращены все данные, поэтому необходимо уточнить какая именно информация нам требуется. Поэтому приводиться условие, чтоб необходимы данные касающиеся лишь данного теста, так же то, что правильность ответа должна быть равна 1 и тест который выполняется находится в открытом доступе. Если все условия соблюдены, то данный массив заполняется, правильный ответ крепиться к нужному вопросу.

В завершении, студенту, ответившему на все вопросы, нужно будет закончить тест. Для этого он нажимает на button, которая автоматически вызывает функцию записи результатов в БД, а так же выводит их на экран для уведомлении пользователю (Листинг 3.7)[21].

Листинг 3.7 Запись ответов пользователя в хранилище

```
function get_test_data_result($test_all_data, $result, $POST){
    foreach($result as $q => $a){
        $test_all_data[$q]['correct_answer'] = $a;
        if( !isset($POST[$q]) ){
            $test_all_data[$q]['incorrect_answer'] = 0;}}
    foreach($POST as $q => $a){
        if( !isset($test_all_data[$q]) ){
            unset($POST[$q]);
            continue;}
        if( !isset($test_all_data[$q][$a]) ){
```

```
                    $test_all_data[$q]['incorrect_answer'] = 0;
                    continue;                    }
            if( $test_all_data[$q]['correct_answer'] != $a ){
                    $test_all_data[$q]['incorrect_answer'] = $a;}}
    return $test_all_data;}
```

Сначала происходит заполнение массива test_all_data правильными ответами. Далее если были не отвеченные варианты, то данные так же заносятся, но уже с другим указателем. После этого выполняется поиск неправильных ответов, и внести их тот же массив. Так же происходит проверка массива, если вдруг появляются лишние символы, они автоматически удаляются. Если все данные верны, то неправильные ответы так же успешно заносятся.

Последним главным функционалом является вывод результата на экран для студента и запись этой информации в базу данных, то есть в специально отведенную таблицу statistic(см. приложение, листинг 2) [22].

Первоначально данный алгоритм присваивает переменной all_count количество всех вопросов теста. Потом передаются первоначальные данные переменным, отвечающим за верные и неверные ответы в виде процента прохождения. После инициализации переменных, выполняется подсчет прохождения теста. Происходит изменение переменной correct_answer_count, с помощью разницы между количеством всех ответов и неверными значениями. Далее происходит подсчет в виде процента прохождения теста. При любом тестировании необходим определенный порог, то есть результат, при котором студенту засчитывается сдача теоретической части лабораторной работы. Поэтому, если данный порог не был пройден, то появляется сообщение о повторном прохождении теста. Если же студенту удалось выполнить тест на необходимый порог или выше, то происходит вывод результата на экран. В момент вывода на экран, алгоритм так же выполняет запись всего результата в базу данных, в таблицу статистики.

4 РЕЗУЛЬТАТЫ ПРИМЕНЕНИЯ АТОМАТИЗИРОВАННОЙ СИСТЕМЫ ТЕСТИРОВАНИЯ

4.1 Сведения о работе программного средства

В данном пункте описывается апробация всей системы тестирования, а именно наглядный разбор работы системы, доступ разным пользователям и приведение результатов первоначального. Для реализации всего, будут использоваться скриншоты веб-интерфейса, его работы и описание, как действует система.

Вначале необходимо рассмотреть главную страницу системы. Переход на нее имеется у любого пользователя, неважно зарегистрирован он или нет. На данной странице отображается базовая информация о системе, а так же ссылки на возможные переходы к дальнейшему функционалу. Есть два пути перехода это администратор и тестирование. Доступ к первому имеет лишь только рабочий персонал, то есть преподаватель или модератор сайта. К тестированию может обратиться студент, но ему необходимо будет авторизоваться (Рисунок 4.1).

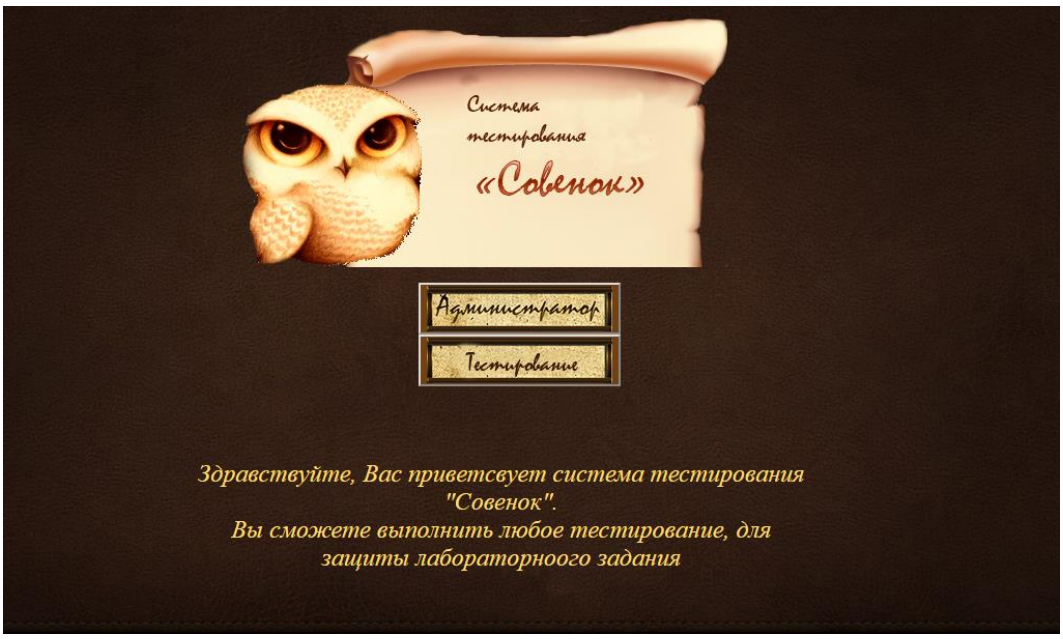
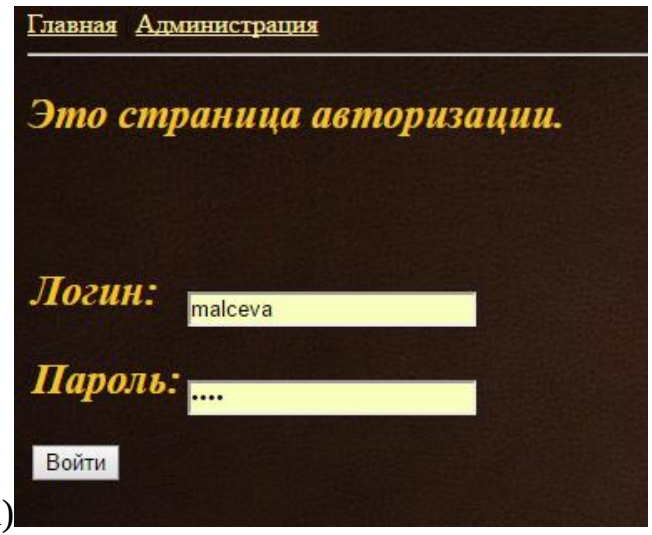


Рис. 4.1 Главная страница системы тестирования

Если в систему вошел администратор, то ему необходимо нажать на раздел “администратор”, который автоматический переведет его на авторизацию на странице, и лишь после этого будет переведен к функционалу редактирования базы данных со стороны сайта (Рисунок 4.2 а).

Как только администратор прошел ввод данных, и информация была подтверждена, его перенаправляют на страницу с возможностью редактировать данные БД. На странице прописаны отдельные алгоритмы работы с такими таблицами как: пользователь, тест, вопрос, ответ, статистика. Причем администратор может не только просматривать данные БД, но так же выполнять редактирование, сортировку, удаление со стороны веб-интерфейса не прибегая к переходу в СУБД (Рисунок 4.2 б).

а)



б)

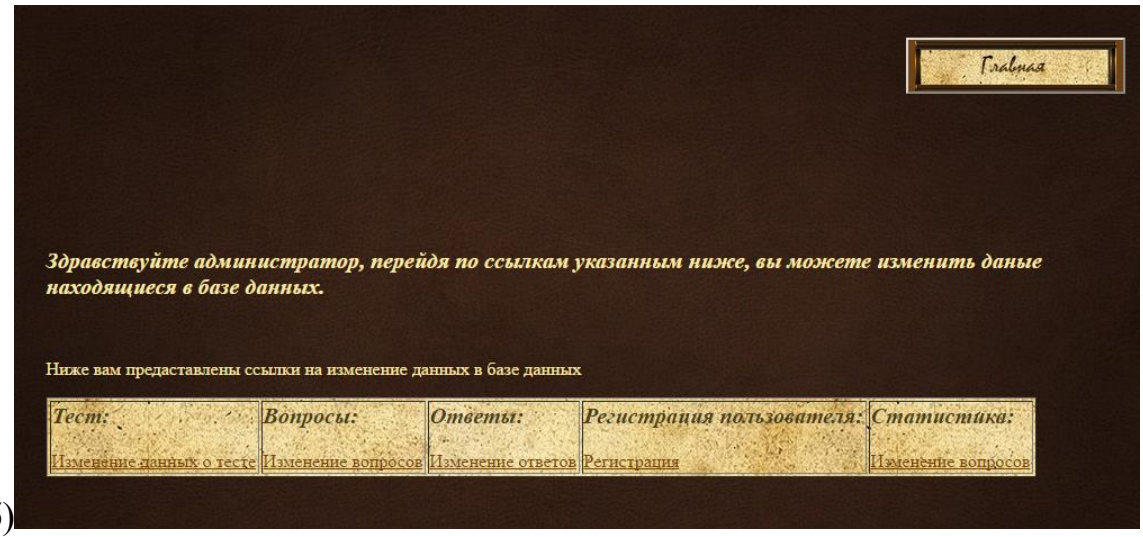


Рис. 4.2 а) Страница авторизации администратора

б) Страница администратора

Для каждой из таблиц относящихся именно к тестированию, есть функционал по добавлению, просмотре и удалении записей. Для того чтобы отредактировать определенную сущность, администратору необходимо выбрать нужный раздел, и там откроется отдельная страница, на которой есть необходимые данные по преобразованию информации. По своей структуре весь функционал разных сущностей одинаков, различие лишь идет в запросах, а именно данных столбцов и их наименований для создания обработчика, поэтому ниже представлен один из трех веб-обработчиков сущностей (Рисунок 4.3).

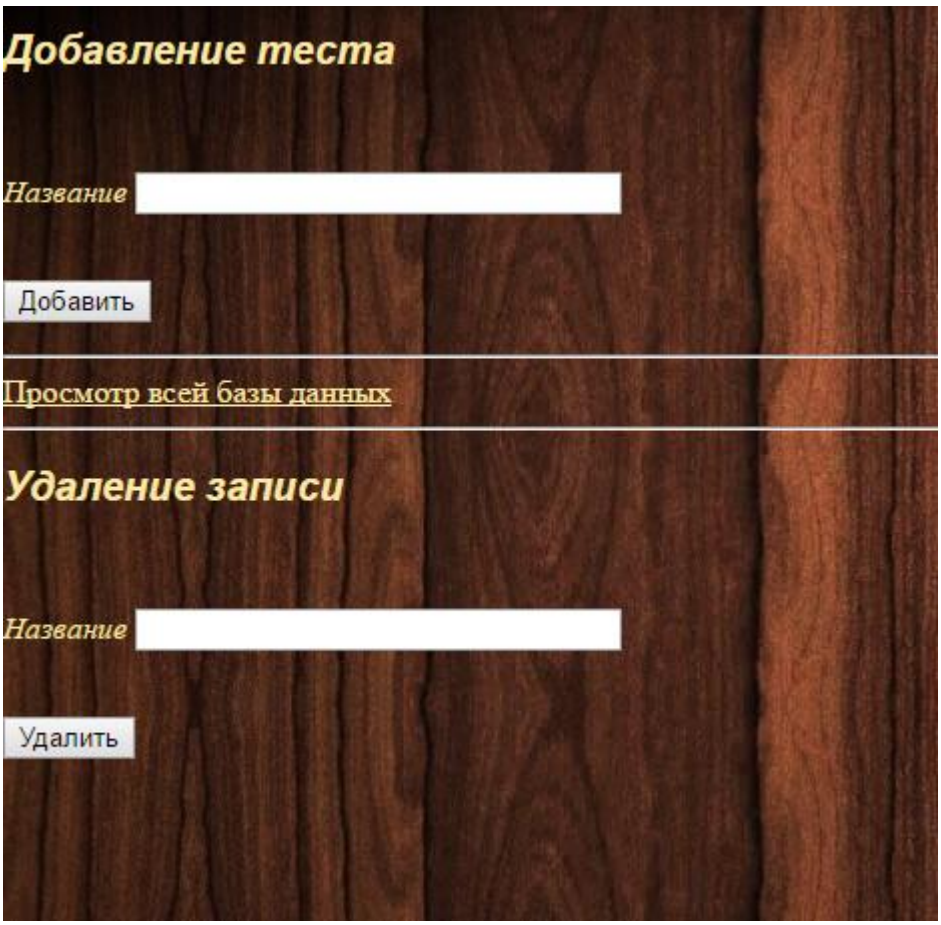


Рис. 4.3 Пример страницы изменения данных БД

Другой раздел, регистрация пользователя, представляет собой страницу, на которой располагается форма для заполнения данных о новом трестирующемся. Для этого необходимо ввести данные студента, а именно его логин (обычно это связка его фамилии и инициалов или же номер зачетной книжки) и пароль, который студент выберет. Если все данные внесены в поля, то остается нажать на кнопку “Зарегистрироваться” и алгоритм автоматически добавит нового пользователя в БД, а следовательно и в систему (Рисунок 4.4).

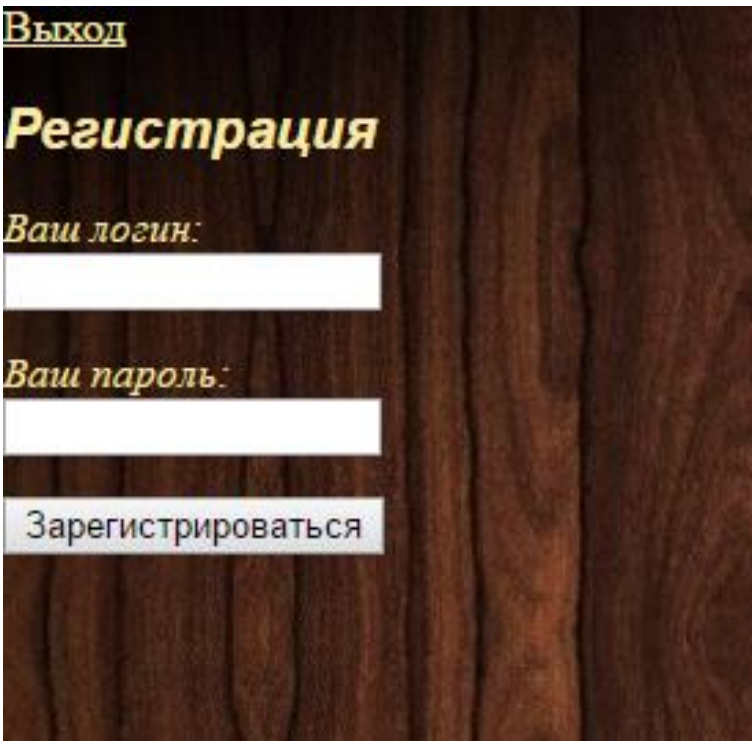


Рис. 4.4 Регистрация нового пользователя (студента)

Последний раздел, это статистика, администратору так же необходимо нужную ссылку и автоматически он перейдет на php файл, где находится функционал, который позволит автоматически вызвать просмотр базы данных сущности statistic. Так же на данной странице будет возможность скачивать все данные таблицы в виде xls файла. Так же Запрос алгоритма построен так, что данные будут выгружаться из нескольких таблиц. Так как информация в сущности статистики хранит в себе лишь внешние ключи, то есть связку данных, а для удобства лучше будет, если сразу выводиться нужная полная информация. Поэтому запрос позволяет выгрузить не просто номера студентов и тестов, а именно их данные (имя, фамилия, группа) и название теста, который был пройден. Так же выгружаются данные самой таблицы статистика, а именно дата прохождения и процент верности. Пример данной реализации представлен на рисунке 4.5.

Рис. 4.5 Пример вывода данных из таблицы

Теперь перейдем к самому тестированию. Для того чтобы пройти тестирование студенту необходимо пройти авторизацию на сайте, поэтому его данные о логине и пароле должны храниться в БД. То есть необходимо заранее попросить преподавателя зарегистрировать его в системе. Введя данные форму, студент будет автоматически переведен на страницу с выбором тестов. Если же возникли, какие то проблемы с авторизацией, системная ошибка или неправильные данные, на экране появиться уведомление и доступ к тестам будет ограничен.

Пройдя авторизацию, трестирующемуся необходимо будет выбрать интересующий его тест из списка, который будет отображен на странице. Если необходимого теста нет, то он может быть в закрытом доступе, то есть преподаватель, пока закрыл возможность его прохождения.

После того как будет выбран тест, студенту нужно будет ответить на все вопросы, которые есть в тесте. Каждый вопрос по одному появляется на экране, так удобнее для визуального восприятия. Для того чтобы выбрать любой другой вопрос, необходимо нажать на соответствующий номер над вопросами. Данный способ позволяет вернуться к любому вопросу в любой момент, то есть если студент не справляется с заданием, он может пропустить его и позже вернуться.

Чтобы студент не запутался в том, на какой вопрос он сейчас отвечает, каждый выбранный вопрос, а именно его номер будет выделяться бежевым оттенком. Количество вопросов тестирования будет соответствовать количеству вопросов, который были сгенерированы для тестирования (Рисунок 4.6).

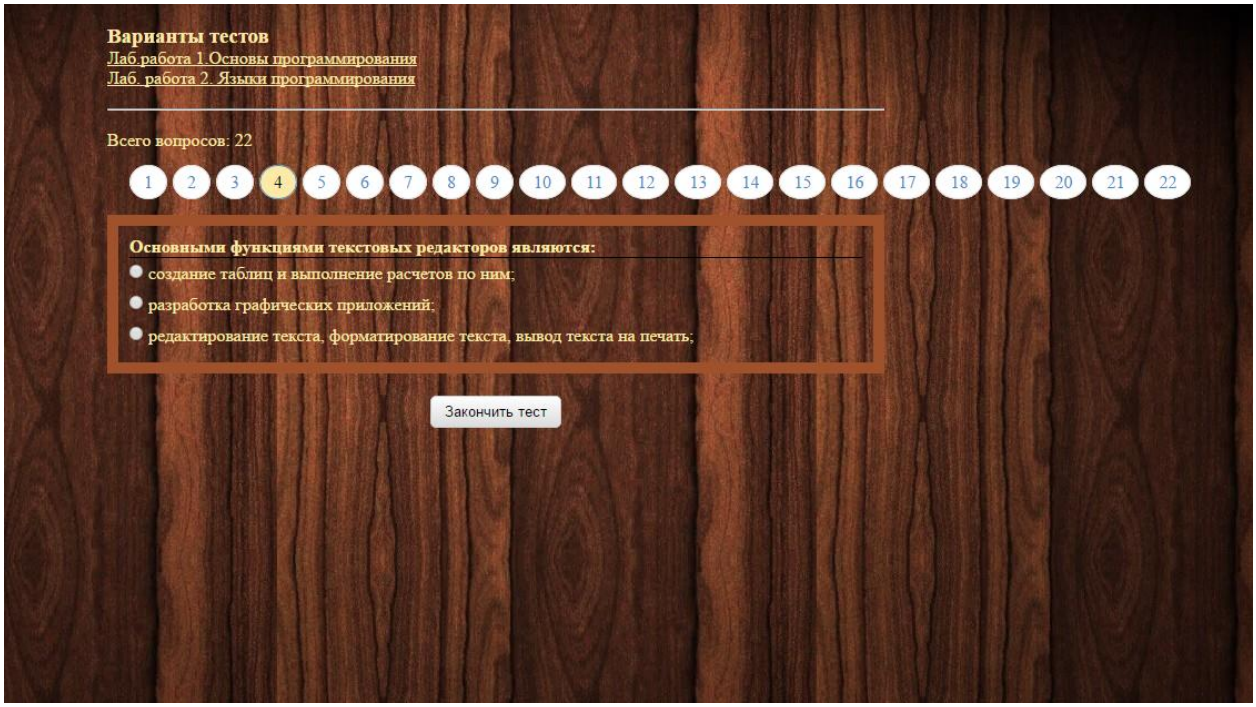


Рис. 4.6 Пример выполнения теста

После того как студент пройдет тест и ответит на все вопросы, ему необходимо будет нажать кнопку “Закончить тест”, автоматически запуститься алгоритм обработки, а именно вычисление процентов правильности тестирования и запись данных в БД. Для студента же после нажатия отобразиться информация, а именно: количество всего вопросов в

тесте, правильные ответы, неверные и процент прохождения самого тестирования по которому будет вычисляться пройден ли порог или нет (Рисунок 4.7).

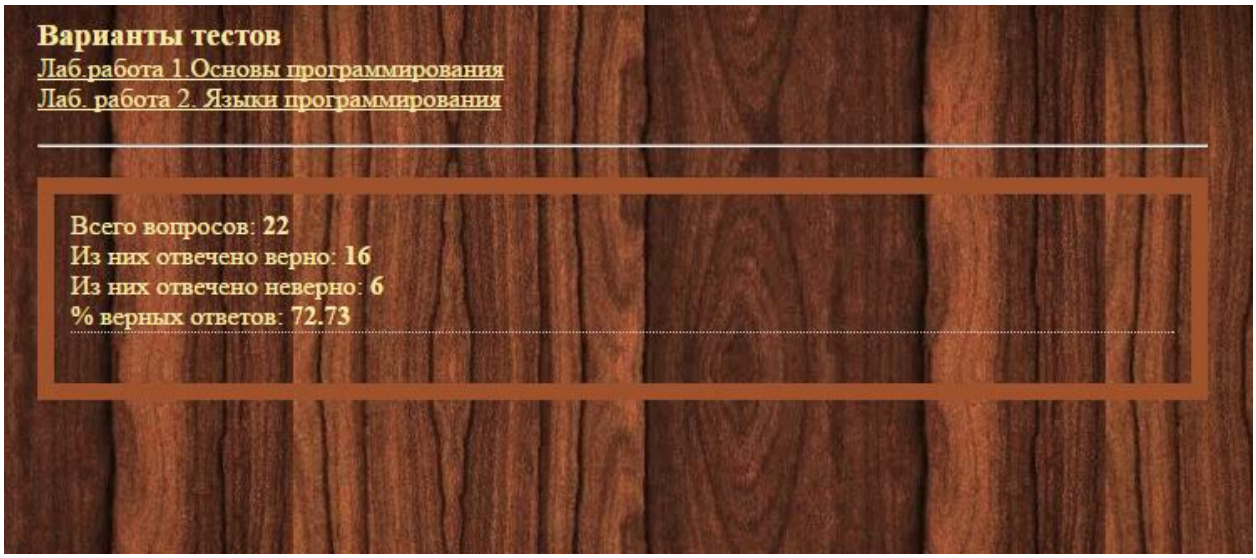


Рис. 4.7 Пример вывода результата теста

По завершению тестирования, студенту необходимо будет нажать на ссылку “Выход”, чтобы закончилась сессия и выйти из своего профиля.

В этом пункте были описан весь ход работы системы тестирования, как можно подробнее были описаны действия работы с тестами и переходами по веб-интерфейсу, а так же возможности ограничения к доступу.

4.2 Применение системы и вычисление порога

Автоматизированная система реализована в полной мере и готова к использованию. В программе реализованы все необходимые функции тестирования студентов, позволяющая провести удалено контроль уровня знаний студентов по лабораторному практикуму. Поэтому было проведено пробное тестирование студентов групп 07001601 и 07001602.

Тестирование проводилось на занятии по лабораторной работе “Основы программирования”. Студентам было предложено ответить на 22

вопроса, которые охватывали главные теоретические знания по теоретической части практикума. В основном время, затраченное на прохождение, было около минут 20, что позволило студентам оставшееся время потратить на защиту уже программного кода лабораторной работы.

Благодаря такому подходу проверки теоретической части, было рационально использовано время защиты лабораторных работ. Студенты, которые раньше освободились, сразу переходили к защите лабораторной, а остальная часть проходила тестирование. Так же были проверены основные вопросы, которые должен знать любой студент при выполнении практической части практикума.

Ниже приведены данные выполнения группами тестирования по первой лабораторной работе (рисунок 4.8)

	name	firstname	group	test_name	percent	date
2	Владимир	Бузин	7001602	Лаб.работа 1.Основы прогр	72.73	01.03.2017 7:42
3	Елизавета	Гречишкина	7001601	Лаб.работа 1.Основы прогр	72.73	22.03.2017 6:30
4	Анастасия	Кожемякина	7001601	Лаб.работа 1.Основы прогр	72.73	22.03.2017 6:39
5	Диана	Неговора	7001601	Лаб.работа 1.Основы прогр	72.73	22.03.2017 6:45
6	Дмитрий	Неклюдов	7001601	Лаб.работа 1.Основы прогр	68.18	22.03.2017 6:54
7	Александр	Гапоненко	7001601	Лаб.работа 1.Основы прогр	36	22.03.2017 7:05
8	Артем	Утянский	7001601	Лаб.работа 1.Основы прогр	77.27	22.03.2017 7:01
9	Анастасия	Чекалина	7001601	Лаб.работа 1.Основы прогр	63.64	22.03.2017 7:08
10	Артур	Ибрагимов	7001602	Лаб.работа 1.Основы прогр	63.64	22.03.2017 7:34
11	Михаил	Рябчунов	7001602	Лаб.работа 1.Основы прогр	72.73	22.03.2017 7:44
12	Ольга	Пелехоца	7001602	Лаб.работа 1.Основы прогр	68.18	22.03.2017 7:55
13	Анастасия	Кривошеина	7001602	Лаб.работа 1.Основы прогр	54.55	22.03.2017 8:13
14	Захар	Булгаков	7001601	Лаб.работа 1.Основы прогр	77.27	15.05.2017 16:24

Рис. 4.8 Данные пробного тестирования

Как показано на рисунке, с тестированием правилась почти вся группа, некоторые, из которых набрали баллов выше среднего. Лишь несколько студентов были не допущены к защите практической части, так как набрал малое количество баллов. Но так или иначе начальный порог был равен 50, то есть половине максимального значения. Это не является рациональным, так как уровень знаний чуть выше среднего, это плохой показатель. Поэтому появляется потребность в калибровке значений порога. То есть при каждом прохождении тестирования, порог будет меняться на основе всех полученных данных тестирования. Для этого используется специальный метод получения порога.

Будем использовать метод статистических данных, по средствам графика данных тестирования (Рисунок 4.9)[23].

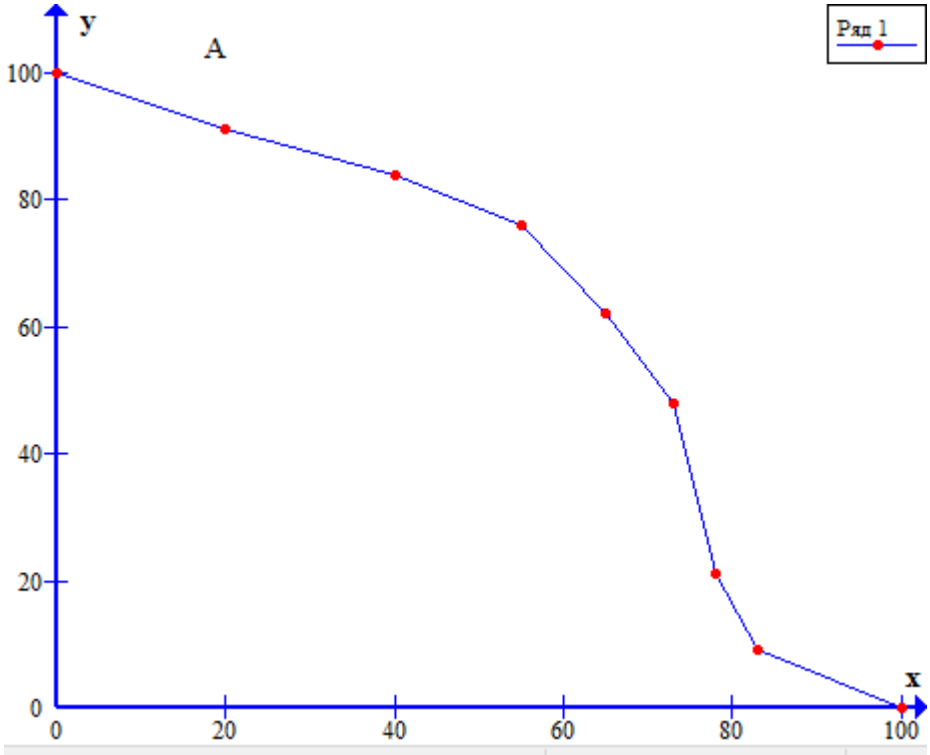


Рис. 4.9 Пример статистических данных

По оси Ox откладывается шкала распределения процентов оценки, то есть от 0 до 100. По оси Oy отмечаются процент от всех студентов сдающих тест, то есть в нашем случае тест был сдан 14 людьми, на каждого из них берется 7 процентов от всех и позже выстраивается шкала. Например точка A отвечает за то, сколько студентов преодолели порог в 20 баллов, то есть при прохождении теста все студенты прошли данный порог.

Данный способ исследования имеет большой плюс в том, что график, не смотря как будет выполнен тест, всегда будет нисходящим. Так же благодаря его “крутости” можно определить тестирование в целом, то есть насколько плохо был он пройден и указывает на то, какое отличие идет между студентами в плане уровня знаний.

Для того чтобы определить границы тройки, необходимо провести анализ графика. Следует найти степень его наклона. Чем он «круче», тем меньше должна быть граница тройки или наоборот. Для этого применяется

математическая функция $sin(\alpha)$. График нахождения оценки, изображён на рисунке 4.10[24].

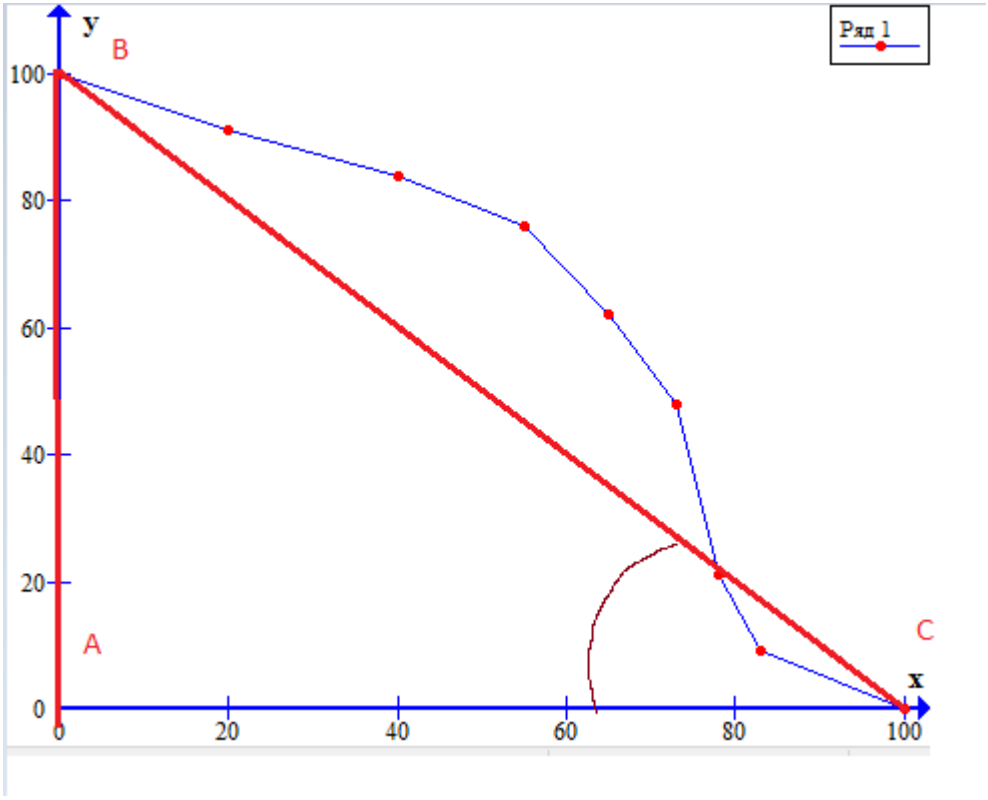


Рис 4.10 Пример нахождения тройки

Отрезок AB будет известен всегда в процессе алгоритма. Чтобы найти BC, используем формулу нахождения расстояния между двумя точками:

$$BC = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \quad (1.1)$$

В итоге имеем:

$$sin(\alpha) = \frac{AB}{\sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}} \quad (1.2)$$

Синус может принимать значения [-1;1], но так как алгоритм работает в первой четверти координатной оси, то синус будет принимать значения [0;1]. Чем «круче» наклон графика, тем значение синуса к 1. Исходя из этого,

можно представить формулу для вычисления границы тройки:

$$kof_3 = \big(1 - sin(\alpha)\big) * 100;$$

Проведя расчет, на основании нашего графика и данных, получим:

$$kof_3 = (1 - 0{,}707) * 100 = 30;$$

После проведения вычислений, мы получаем, что значение тройке равно примерно 30. Другими словами студент при получении баллов за тестирование меньше чем 30, а именно в диапазоне от 0 до 29 получает оценку двойку, что выше уже относиться к тройке. Для этого нам необходимо исследовать интервал двойки.

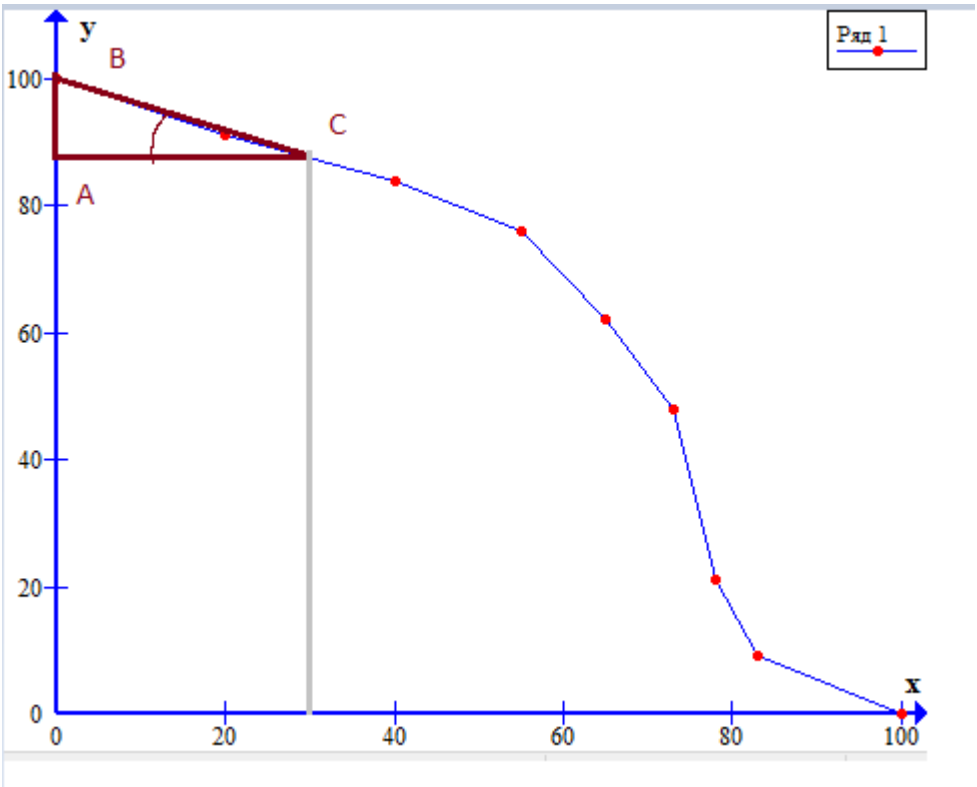


Рис. 4.11 Анализ интервала двойки

Для того чтобы найти пороги оценок четыре и пять, так же пользуемся данным способом, только теперь анализ будет проходить не всего графика, а только его части.

Отчет и рассмотрение данных ведется от значения kof_3, но метод поиска не меняется, все выполняем как в случае с исследованием тройки.

$$kof_4 = (1 - 0{,}47) * 100 = 53;$$

Теперь, надо учесть общий модификатор для шкалы, которым и является kof_3, так как его значение получено на основе наклона графика на всем интервале. Проведя множество случайных выборок и расчетов границ оценок, было выявлено, что для нахождения четверки подходит модификатор равный $\frac{1}{2}kof_3$, а для пятерки $\frac{1}{4}kof_3$. Таким образом, итоговое значение границы четверки:

$$kof_4 = \frac{1}{2}kof_3 + (1 - 0{,}47) * 100 = 68;$$

В итоге, имеем интервал для четверки [30:67]. Остается найти границу пятерки, аналогично двум другим оценкам. В итоге будет получено уравнение:

$$kof_5 = \frac{1}{4}kof_3 + \ (1 - 0{,}212) * 100 = 7{,}5 + \ 78{,}8 = 86;$$

Проведя все расчеты, мы получим шкалу порогов для тестирования:

- Двойка – [0-29]
- Тройка – [30-67]
- Четверка – [67-86]
- Пятерка – [86-100]

Достоинства и недостатки алгоритма.

Среди достоинств используемого алгоритма можно выделить:

- Простота в понимании.
- Легко использовать без помощи ЭВМ.
- Информативность – в процессе построения графика уже можно судить о результатах, так как угол наклона графика говорит об успеваемости.

К недостаткам алгоритма можно отнести:

- Слабая универсальность. Алгоритм плохо работает, если на вход подано мало значений о результатах тестирования.
- Алгоритм игнорирует личностные качества участников.
- Алгоритм, порой, «несправедлив». Например, один из участников тестирования списывал/имел ответы и написал на очень высокий результат. Тогда, шкала оценок сдвинется вправо, иными словами, порог оценок увеличится [25].

ЗАКЛЮЧЕНИЕ

Эффективная работа автоматизированной системы тестирования, является одной из ключевых проблем контроля знаний студентов. Создание гибкого и безошибочного интерфейса для работы с тестами, проверяющими уровень знаний тестирующего по дисциплинам является приоритетом в системе образования учебных заведений. Данное программное средство позволяет оценить подготовленность студента к сдаче лабораторного материала, первоначально изучив его познания в теоретической части выданного ему задания. Для полноценной работы программного обеспечения была использована база данных, которая позволяет системе тестирования хранить большое количество данных, как о пользователях тестов, так и созываемых тестах.

Нахождение нужного вида системы, который смог бы правильно выполнять проверку изученного материала студентом быстро и эффективно, был найдет. Поэтому реализация тестирования была выполнена по средствам web-приложения с подключенной к нему базе данных. На сегодняшний день нет ни одной полностью правильной и всеми используемой системы тестирования, так как у каждой из них есть какие-то недостатки или издержки функционала. Так же на это влияет и то, что многие созданные онлайн тестирования являются платными, так что каждый пытается создать свое программное средство с меньшими затратами.

Целью, данной магистерской диссертации, является автоматизировать учебный процесс по средствам внедрения тестирования теоретической части лабораторного практикума.

Данная цель повлекла за собой решение таких задач как:

- исследовать организацию учебного процесса;
- рассмотреть структурный язык запросов;
- изучить строение баз данных;

- изучить основные методы для реализации тестирования студентов;

- привести созданную базу данных к третьему нормальному виду;

- разработать программное обеспечение для возможности тестирования студента.

Полученный результат в виде системы тестирования в дальнейшем может эксплуатироваться по назначению, а именно выполнять контроль знаний студентов, сдающих теоретический материал для защиты лабораторного практикума.

По завершению написания выпускной квалификационной работы, было достигнуто следующее:

- Рассмотрена организация учебного процесса без использования автоматизации;

- Проанализированы все достоинства и недостатки тестирования, как контроль изученного материала;

- Выявлены главные требования по составлению тестов и заданий для данной системы;

- Детально изучены виды программных продуктов, оптимальных для выполнения практической части тестирования;

- Изучены методы для реализации тестирования студентов;

- Разработано приложение, в котором реализованы данные методы;

- Разработано программное обеспечение для выполнения тестирования студента;

- Выполнено пробное тестирование на работоспособность и безошибочный вывод необходимых данных.

По ходе выполнения магистерской диссертации были изучены основные методы работы с базой данных и создания программного кода интернет-приложения, показано применение этих знаний в различных ситуация, была освоена новая среда разработки, а также применены и показаны умения работы на языках PHP, MySQL, JavaScript, Ajax.

Разработанное приложение показало принцип работы с автоматизированными системами тестирования, а именно поддержки лабораторного практикума.

В результате выполнения выпускной квалификационной работы были решены все поставленные задачи. Результатом проделанной работы является создание приложения, реализующего работу методов, а также проведение различных тестовых заданий, которые смогли показать эффективность работы приложения в рамках данной выпускной квалификационной работы.

При выполнении магистерской диссертации был выполнен весь необходимый перечень и объем работ.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Терешин В.А., Тестирование студентов [Текст]: - СПбГПУ «Санкт-Петербургский государственный политехнический университет», 2007 г,-100с.
2. Юдалевич Н.В., Использование автоматизированных систем тестирования при работе со студентами [Текст]: - Ярославский педагогический вестник, 2010 г. – 166с.
3. Попов А.В. , Тестирование как метод контроля качества знаний студентов [Текст]: - Народное образование. Педагогика, 2013 г. - 286
4. Аванесов В.С., Композиция тестовых заданий [Текст]: - М.: Центр тестирования,2002 г. - 239 с.
5. Васильев В.И., Требования к программно-дидактическим тестовым материалам и технологиям компьютерного тестирования [Текст]: - Издательство МГУП, 2005 г. -37с.
6. Борзых Е.А., Разработка заданий в тестовой форме [Текст]: - ФГОУ СПО «Оренбургский государственный колледж», 2009 г. -18с.
7. Ролина Е. Н. Тестирование как один из методов педагогического контроля знаний и умений студентов в преподавании истории [Текст]: - Чебоксары, 2015 г. – 3с.
8. Усков В.Л., Информационные технологии в образовании [Текст]: - Информационные технологии, 2002 г. – 37 с.
9. Ruby S., Thomas D., Heinemeier Hansson D., Agile Web Development with Rails [Текст]: - The Pragmatic Bookshelf, 2011 г. – 480 с.
10. Мартинес А. Секреты создания недорогого Web – сайта: Как создать и поддерживать удачный Web – сайт, не потратив ни копейки: [пер. с англ.]:- ДМК Пресс, 2009 г. – 414 с.
11. Уильямс Б., Дэмстра Д., Стэрн Х. WordPress для профессионалов [Текст]:– СПб.: Питер, 2014. – 464 с.

12. Нрачев А. Создаем свой сайт на WordPress: быстро, легко и бесплатно. Работа с CMS WordPress 3 [Текст]:– СПб.: Питер, 2011. - 543 с.

13. Википедия [Электронный ресурс]: - Режим доступа: https://ru.wikipedia.org/wiki/%D0%9D%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%84%D0%BE%D1%80%D0%BC%D0%B0_%D0%91%D0%BE%D0%B9%D1%81%D0%B0_%E2%80%94_%D0%9A%D0%BE%D0%B4%D0%B4%D0%B0

14. Мерсер Д.У., Кент А., PHP 5 для начинающих [Текст]: - Издательство Диалекстика, 2006 г. – 846с.

15. Девис М.Е., Филлипс Д.А., Изучаем PHP и MySQL [Текст]: - Издательство Символ-Плюс, 2008 г. – 448с.

16. Колисниченко Д.Н., Профессиональное программирование на PHP [Текст]: - Издательство БХВ-Петербург, 2007 г. – 416с.

17. HTML.net [Электронный ресурс]: - Режим доступа: http://ru.html.net/tutorials/php/lesson12.php

18. Бейли Л., Моррисон М., Изучаем PHP и MySQL [Текст]: - Издательство Эксмо, 2010 г. – 800с.

19. Кузнецов М.В., PHP 5. Практика разработки Web-сайтов [Текст]: - Издательство БХВ-Петербург, 2005 г. – 960с.

20. Никсон Р., Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript [Текст]: - Издательство Питер, 2011 г. – 497с.

21. Ловэйн П., Объектно-ориентированное программирование на PHP 5 [Текст]: - Издательство НТ Пресс, 2007 г. – 224с.

22. Шлосснейгл Д., Профессиональное программирование на PHP [Текст]: - Издательство Вильямс, 2006 г. – 624с.

23. Бююль А., Цефель П., SPSS: Искусство обработки информации. Анализ статистических данных и восстановление скрытых закономерностей [Текст]: - Издательство Диасофт, 2005 г. – 608с.

24. Васильева Э.К., Лялина В.С., Статистика [Текст]: - Издательство Юнити-Дана, 2012 г. – 398 с.

25. Божко В.П., Информационные технологии в статистике [Текст]: - Издательство Юнити-Дана, 2011 г. – 152с.

ПРИЛОЖЕНИЕ

Листинг 1. Файл save_user.php

```
<?php
session_start();
if (isset($_POST['login'])) { $login = $_POST['login']; if ($login == '') { unset($login);} }
if (isset($_POST['password'])) { $password=$_POST['password']; if ($password =='') { unset($password);} }
if (empty($login) or empty($password))
{exit ("Вы ввели не всю информацию, вернитесь назад и заполните все поля!");}
$login = stripslashes($login);
$login = htmlspecialchars($login);
$password = stripslashes($password);
$password = htmlspecialchars($password);
$login = trim($login);
$password = trim($password);
include ("bd.php");
$result = mysql_query("SELECT * FROM users WHERE login='$login'",$db);
$myrow = mysql_fetch_array($result);
if (empty($myrow['password']))
{exit ("Извините, введённый вами логин или пароль неверный.");}
else {    if ($myrow['password']==$password) {
        $_SESSION['login']=$myrow['login'];
        $_SESSION['id_user']=$myrow['id_user'];
        header("Location: test.php");
        exit;}else {exit ("Извините, введённый вами логин или пароль неверный.");}}?>
```

Листинг 2. Реализация вывода на экран

```
$all_count = count($test_all_data_result);
        $correct_answer_count = 0;
        $incorrect_answer_count = 0;
        $percent = 0;
        foreach($test_all_data_result as $item){
                if( isset($item['incorrect_answer']) ) $incorrect_answer_count++;  }
        $correct_answer_count = $all_count - $incorrect_answer_count;
        $percent = round( ($correct_answer_count / $all_count * 100), 2);
        if( $percent < 50 )return 'Вы набрали менее 50%, попробуйте пройти тест заново';
        $print_res = '<div class="test-data">';
                $print_res .= '<div class="count-res">';
                        $print_res .= "<p>Всего вопросов: <b>{$all_count}</b></p>";
                        $print_res .= "<p>Из них отвечено верно: <b>{$correct_answer_count}</b></p>";
                        $print_res .= "<p>Из них отвечено неверно: <b>{$incorrect_answer_count}</b></p>";
                        $print_res .= "<p>% верных ответов: <b>{$percent}</b></p>";
                $print_res .= '</div>'; // .count-res
        $print_res .= '</div>'; // .test-data
        $id_user=$_SESSION['id_user'];
        $id_test=$_POST['test'];
        $days = date("j.m.y H:i");
        $q = "insert into statistic (id, id_user, id_test, percent, date) values(NULL,'$id_user','$id_test','$percent','$days')";
        mysql_query($q);
        return $print_res;
```

Листинг 3. Главная страница тестирования

```
<html>
<head>
        <meta charset="utf-8">
<title>Система тестирования</title>
<link rel="stylesheet" type="text/css" href="CSS/style.css">
</head>
<body>
   <div id="container">
      <div class="block">
```

```
   <img src="image/logo.png" ><br>
</div>
<div class="menu">
<table>
<button  style="background-color:704214"  onclick="javascript:window.location='2.php'"><img src="image/4.jpg" style="vertical-align: middle" ></button>
<button                                                    style="background-color:704214" onclick="javascript:window.location='open.php'"><img src="image/5.jpg" style="vertical-align: right" ></button>
</table></div>
<br><br>
<div class="block2">
<p> <font face=Italic color=fcdd76 size=5><i>  Здравствуйте, Вас приветсвует система тестирования "Совенок".<br>
Вы сможете выполнить любое тестирование, для защиты лабораторноого задания<br>
</i></font></p>
</div>
</div>
   </div>
</body>
</html>
```

Листинг 4. Реализация вывода данных таблицы статистика

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="CSS/01.css">
</head>
<body>
<h2>Вывод данных базы<h2>
        <table border=1><tr><td><p>Скачать файл отчета</p><a href="ex1.php">Скачать</a></td></tr></table>
<?php
@ $db=mysql_pconnect('localhost', 'root', '');
if (!$db)
{echo 'Ошибка соединения с базой данных';exit;}
mysql_query("set names utf8");
mysql_select_db('testing');
$query="SELECT u.name,u.firstname,u.group, t.test_name,s.percent,s.date
                FROM statistic s
                LEFT JOIN test t
                ON s.id_test=t.id_test
                LEFT JOIN users u
                ON s.id_user=u.id_user ";

$result=mysql_query($query);
$num_results = mysql_num_rows($result);
echo '<p>Найдено:'.$num_results.'</p>';
for ($i=0; $i<$num_results; $i++)
{$row = mysql_fetch_array($result);
echo '<p>'.($i+1).'.Имя:';
```

```
echo stripslashes($row['name']);

echo '<br>Фамилия:';
echo stripslashes($row['firstname']);

echo '<br>Группа:';
echo stripslashes($row['group']);

echo '<br>Название теста:';
echo stripslashes($row['test_name']);

echo '<br>Результат:';
echo stripslashes($row['percent']);

echo '<br>Дата:';
echo stripslashes($row['date'])."<br></p>";}?>
</body>
</html>
```

Выпускная квалификационная работа выполнена мной совершенно самостоятельно. Все использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

«___» _____ _____ г.

_____ _____

(подпись) *(Ф.И.О.)*