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 ПРИКЛАДНОЙ ИНФОРМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

# Информационное сопровождение процесса тестирования. Модуль контроля знаний студентов и анализа полученных результатов

Выпускная квалификационная работа
обучающегося по направлению подготовки 09.03.03 Прикладная информатика
очной формы обучения, группы 07001405
Точоной Карины Евгеньевны

Научный руководитель
Старший преподаватель
Болгова Е.В.

БЕЛГОРОД 2018

СОДЕРЖАНИЕ

# ВВЕДЕНИЕ

Обучение – трудоемкий и многогранный процесс, который требует внедрение современных технологий. Классические методы обучения стали непродуктивными, так как однообразие и монотонность учебного процесса не мотивируют к познанию и самообучению.

Сегодня рейтинг вуза зависит не только от квалификации преподавателей, но и от технической базы, которой владеет учебное заведение. Таким образом, модернизация образовательной системы становится одним из важных критериев определения престижа высшего учебного заведения.

Контроль знаний с помощью тестирования представляет собой список заданий, на которые должен ответить учащийся и при обработке которых можно определить степень усвоения материала и успеваемость студента. Данный способ имеет ряд важных преимуществ по сравнению с бланочным тестированием, а именно:

- уменьшение трудоемкости процесса;
- быстрая обработка результата;
- быстрое получение результата;
- массовость;
- самостоятельность;
- проявление интереса, мотивация;
- повышение эффективности учебного процесса;
- конфиденциальность;
- исключение предвзятости.

Обработка результата тестирования имеет актуальный характер, так как существует большое количество авторских систем, которые не обладают удобным интерфейсом и графическим представлением данных для дальнейшего анализа студентами. Нестандартный подход интерпретации знаний помогает заинтересовать учащихся и склонить к улучшению своих навыков и умений, изменить динамику успеваемости. [15]

Итоговый информационный отчет по всем темам дисциплины имеет важное значение для учащихся, так как на основании собственного рейтинга можно определить какой материал необходимо снова проработать.

Зачастую информационные системы тестирования не имеют наглядности представления результата, имеют проблемы с отображением информации в процессе тестирования. Пользователи сталкиваются с проблемой отсутствия интуитивно-понятного интерфейса, отсутствием обратной связи при обнаружении ошибки в тестировании и отсутствием возможности проведения тестирования без использования глобальной сети. Поэтому тема данной ВКР является актуальной.

Объектом исследования является деятельность отдела управления электронных образовательных ресурсов.

Предметом исследования является процесс анализа тестирования.

Цель: улучшение процесса тестирования с помощью программной реализации обработки данных и графическим представлением результативной информации для студентов без дальнейшего редактирования технологии.

Для достижения поставленной цели необходимо решить следующий ряд задач:

– изучение предметной области;
– выявление недостатков разработанных систем по заданной теме;
– изучение литературы для выбора программных средств;
– обоснование выбора программных средств для проектирования;
– разработка модели «КАК ЕСТЬ»;
– построение модели «КАК ДОЛЖНО БЫТЬ»;
– разработка интерфейса системы;
– разработка автоматизированной системы обработки данных по результатам учебного тестирования студентов;
– тестирование системы.

Структура представлена аналитическим исследованием предметной области и применением изученных материалов на практике.

В первой главе рассматривается существующее состояние предметной области, описываются недостатки и проблемы созданных ранее автоматизированных систем, приводятся обоснованные решения по устранению выявленных недостатков.

Вторая глава содержит обоснования проектных решений по техническому, информационному, программному, технологическому обеспечению и обоснование выбора программных средств. Так же приводится информационное обеспечение комплекса задач.

В третьей главе содержится детальное описание программной реализации проектных решений: общие положения, технологическое обеспечение задачи, описание разработанной системы и расчет показателей эффективности.

Выпускная квалификационная работа изложена на 53 страницах, содержит 22 рисунка, 21 таблицу, 8 листингов, 3 приложения и 32 библиографических источника.

# 1 Аналитическая часть

## 1.1 Технико-экономическая характеристика предметной области

### 1.1.1 Характеристика предприятия

Белгородский государственный национальный исследовательский университет – один из старейших вузов Белгорода, крупнейший вуз Белгородской области.

Белгородский государственный национальный исследовательский университет, основанный в 1876 г., является крупнейшим научно-образовательным и культурным центром Белгородской области, входит в Ассоциацию ведущих вузов России и в двадцатку лучших вузов страны, по оценке Министерства образования и науки.

В НИУ «БелГУ» обучается более 23 тысяч студентов из всех регионов России и 91 страны мира. Свыше 2800 студентов – иностранные граждане. НИУ «БелГУ» входит и консорциум ведущих вузов страны, продвигающих российское образование за рубежом, является базовым вузом сетевого Университета ШОС по пяти направлениям. В университете 22 учебных корпуса и 55 научных лабораторий и центров, 24 базовые кафедры, 20 клинических кафедр.

Структура управления университетом: руководство, учёный совет, наблюдательный совет НИУ «БелГУ», попечительский совет НИУ «БелГУ», общественный совет при ректоре НИУ «БелГУ», ректорат.

БелГУ – это многопрофильный научно-образовательный комплекс, включающий 7 институтов и 4 отдельных факультета, Медицинский колледж и 1 филиал. [3]

### 1.1.2 Краткая характеристика подразделения

В настоящее время в области информатизации образования основное внимание фокусируется на проблемах создания эффективных электронных образовательных ресурсов.

Основным направлением деятельности отдела управления электронных образовательных ресурсов является: создание и развитие электронно-библиотечной системы университета, обеспечение обучающихся и преподавателей электронными образовательными ресурсами, создание базы электронных изданий, базы электронных образовательных ресурсов.

Задачи отдела:

– ведение учета и анализа результатов тестирования;

– организация работы по созданию тестовой базы для контроля знаний студентов;

– определение технических требований к учебно-методическим материалам для подготовки электронных образовательных ресурсов;

– внесение предложений по совершенствованию и развитию системы электронного обучения НИУ «БелГУ» в части учебной работы;

– организация и контроль работ по разработке и актуализации содержимого учебно-методических материалов. [4]

Также в функции отдела входит внедрение и сопровождение автоматизированной библиотечно-информационной системы в научно-технической библиотеке университета.

## 1.2 Постановка задачи

### 1.2.1 Цель и назначение автоматизированного варианта решения задачи

Целью автоматизированного варианта решения задачи является улучшение процесса тестирования с помощью программной реализации обработки данных. Для устранения выявленных недостатков разработанных раннее систем, необходимо решить следующие задачи:

– Устранение проблемы с отображением информации в процессе тестирования.

– Обеспечение интуитивно-понятным интерфейсом.

– Обеспечение обратной связи при обнаружении ошибки в тестировании.

– Предоставление возможности проведения тестирования по локальной сети.

Назначением реализации выпускной квалификационной работы является решение задач:

– Хранение информации в единой информационной базе.

– Расчет оценки студента после тестирования.

– Автоматическое построение отчетов.

– Возможность работать с фондом тестовых заданий.

– Проведение удаленного тестирования.

– Возможность учёта успеваемости студентов.

### 1.2.2 Общая характеристика организации решения задачи на ЭВМ

Автоматизация решения поставленной задачи приводит к изменению технологии ее решения: осуществляется сбор жалоб во время тестирования, администраторы системы не выполняют правки в системе.

Автоматизация решения задачи не влияет на источники и периодичность поступления информации, так как источники находятся за пределами системы и являются независимыми.

Этапы выполнения задачи были расширены, так как добавились новые функции для пользователей. Временной регламент остался неизменным.

Порядок ввода информации не изменился, используется логин и пароль пользователя для авторизации и аутентификации.

Перечень экранных форм:

– окно авторизации;
– личный кабинет;
– форма тестирования;
– форма вывода оценок;
– форма вывода успеваемости.

Результатом работы системы является отчет об успеваемости студента.

Вся информация хранится в единой информационной базе.

Режим решения задачи является диалоговым.

Периодичность: данная задача решается по мере необходимости студентов в тестировании.

## 1.3 Обоснование необходимости и цели использования вычислительной техники для решения задачи

Для описания функционирования системы использовались следующие методологии: IDEF0, DFD, IDEF3. [19,30,32]

Контекстная диаграмма «Модуль контроля знаний и анализа полученных результатов» построена с помощью методологии IDEF0. Данная нотация предназначена для формализации и описания бизнес-процессов. Диаграмма содержит следующие входящие потоки: регистрационная информация пользователя, информация из фонда тестов, информация о выбранном тесте.

Результатом преобразования является выходная информация: отчет по результатам тестирования и результаты тестирования пользователя. Управляющие документы: регламентирующие документы и руководство пользователя. Механизмы воздействия: программное обеспечение, студент.

Контекстная диаграмма «КАК ЕСТЬ» представлена на рисунке 1.1.

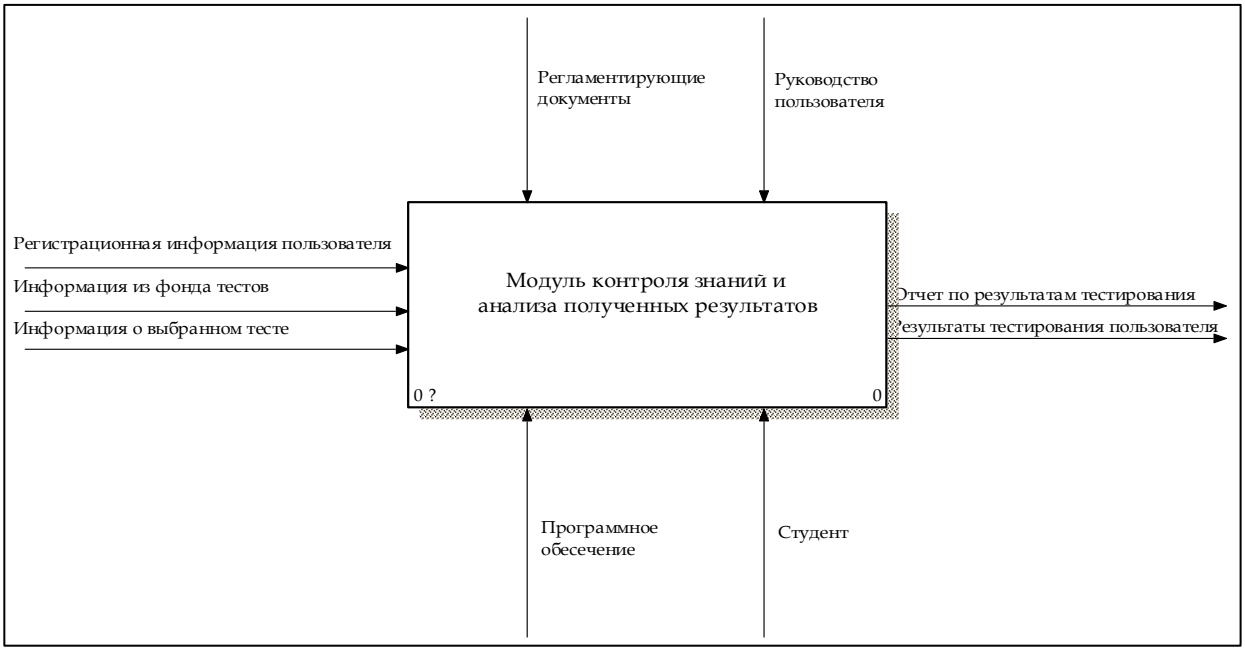



Рисунок 1.1 – Контекстная диаграмма

Для более детального описания системы используются декомпозиции. Диаграмма, представленная на рисунке 1.2 содержит четыре функциональных блока: «Аутентификация пользователя», «Прохождение тестирования», «Обработка и анализ результатов тестирования», «Формирование отчета».

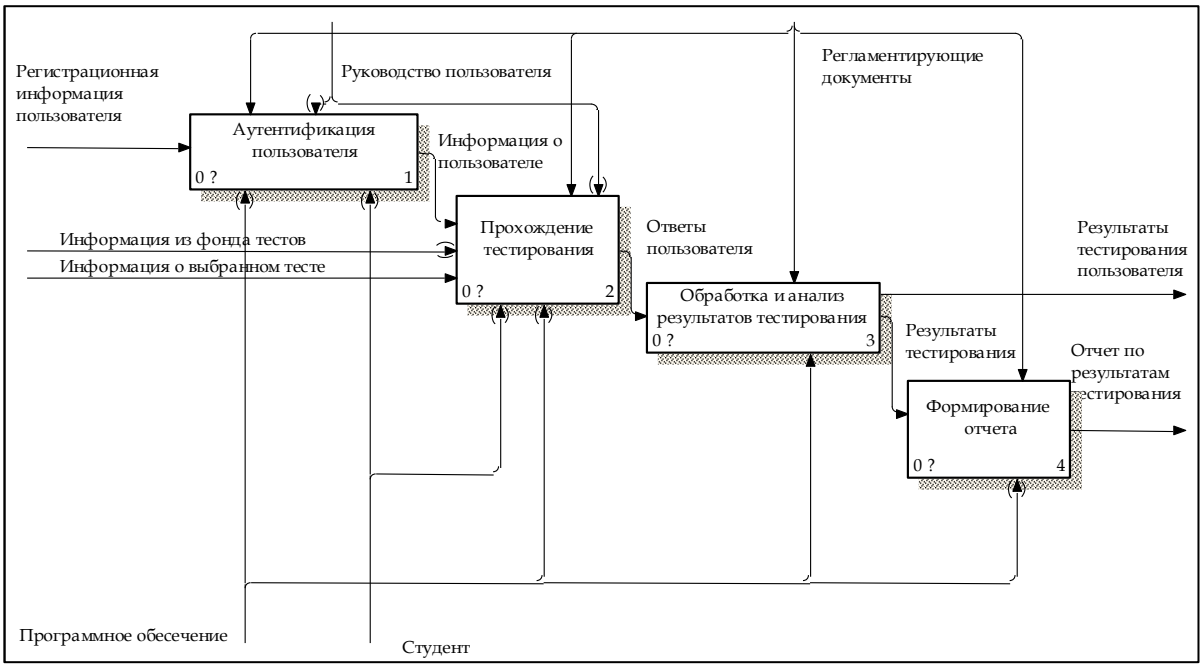



Рисунок 1.2 – Диаграмма декомпозиции «Модуль контроля знаний и анализа полученных результатов»

После анализа существующих систем были выделены следующие недостатки:

– отсутствие возможности наглядно проанализировать свою успеваемость;

– дублирование ответов в тестах;

– невозможность сообщить об ошибке в процессе прохождения теста;

– отсутствие понятного и удобного интерфейса системы;

– отсутствие возможности пользоваться системой без интернета.

## 1.4 Анализ существующих разработок

1. «СИНТЕЗ».

Данный сервис предназначен для проведения тестирований. «Синтез» доступен по локальной сети и через интернет.

Система имеет три ролевых модуля: «учитель», «ученик» и «завуч».

– Модуль «завуч» задает права пользователям.

– Модуль «учитель» предоставляет возможность создавать, публиковать и редактировать тесты, осуществлять проверку результатов и выставлять оценки.

– Модуль «ученик» предполагает прохождение тестирования.

Система «СИНТЕЗ» позволяет выводить темы, вопросы и варианты ответов в любом порядке.

Тестовая оболочка вычисляет оценку по тестированию по пятибалльной системе.

Данный сервис не подходит для внедрения, так как отличается системой оценивания и не имеет визуального представления информации для студентов, нет возможности сообщить о сбоях в работе во время тестирования.

2. «MiniTest-SL».

Тестирующая программа «MiniTest-SL» предназначена для проверки знаний учащихся по различным направлениям.

Данная программа имеет ряд преимуществ: нет ограничений на количество тестов, разделов, вопросов. Есть ограничение на количество тестов, которые может пройти студент, так же имеется ограничение по времени.

Система может работать через интернет и по локальной сети.

Программа «MiniTest-SL» не подходит по следующим критериям: ответы на вопросы выводятся только в одном порядке.

3. «MultiTester».

Данная тестовая оболочка работает по локальной сети. Система включает в себя модули: «преподаватель», «редактор вопросов» и «учащийся». MultiTester обладает защитой данных от хищения и правки.

Недостатками данной оболочки является: поддерживается только на Windows 98 – XP, Windows 7 – ограниченно. Система хранится на одном компьютере: для того, чтобы проводить тестирование необходимо отсылать оболочку по локальной сети. При выборе системы тестирования для ВУЗа данные отрицательные критерии являются недопустимыми.

4. «UniTest System».

Данная система работает по сети интернет и в локальной сети. Программа выдает детальные отчеты по результатам тестирования учащихся. Имеет два модуля: модуль создания и хранения тестов, модуль тестирования.

Положительным отличием данной системы является то, что она находится в свободном доступе и имеет удобный интерфейс для пользователей. Инструментарий достаточно обширен, информация защищена шифром BlowFish 448 бит.

Данная система не подходит, так несовместима с последними версиями операционной системы. [15]

Вывод: вышеперечисленные тестовые оболочки имеют недопустимые критерии для внедрения в локальную сеть ВУЗа. Наилучшим вариантом является создание собственной системы, которая будет отвечать всем поставленным требованиям.

# 2 Проектная часть

## 2.1 Обоснование проектных решений

### 2.1.1 Обоснование проектных решений по техническому обеспечению

Техническое обеспечение представляет комплекс технических средств, предназначенных для обработки данных в ЭИС. В состав комплекса входят ЭВМ, осуществляющие обработку экономической информации, средства подготовки данных на машинных носителях, средства сбора и регистрации информации, средства передачи данных по каналам связи, средства накопления и хранения данных и выдачи результатной информации, вспомогательное оборудование и организационная техника.

Существующая компьютерная техника отвечала всем требованиям и соответствовала своими параметрами.

Таблица 2.1 – Минимальные требования к компьютерной технике

| №№ | Наименование | Характеристика |
|---|---|---|
| 1. | Материнская плата | Acer TravelMate B113 Series |
| 2. | Процессор | Intel® Celeron® |
| 3. | Жесткие диски | 500GB (5.4k) |
| 4. | Модули памяти | 4GB (1x 4GB) DDR3, up to 8GB (2 slots) |
| 5. | Видеокарты | Intel® HD Graphics with 128 MB of dedicated system memory, supporting Microsoft® DirectX® 10.1 |
| 6. | Звуковые карты | Two built-in stereo speakers, HD audio support, Built-in analog microphone, MS-Sound compatible |
| 7. | Монитор | 11.6" HD Acer ComfyView™ (1366 x 768 pixel resolution (WXGA), Mercury-free, environment-friendly) |

### 2.1.2 Обоснование проектных решений по информационному обеспечению

Для создания рабочей программы тестирования необходимы следующие документы:

– Карта компетенций – информация о результатах освоения дисциплины.

– Рабочая программа дисциплины.

– СУОС ВПО – самостоятельно устанавливаемые образовательные стандарты и основные образовательные программы ВПО в НИУ «БелГУ» на основе ФГОС и СУОС ВПО.

– УМКД – это совокупность учебно-методических материалов, способствующих эффективному усвоению студентами содержания учебной дисциплины, входящей в основную образовательную программу по одному из направлений (специальности) подготовки.

– ФГОС ВПО – федеральный государственный образовательный стандарт.

– Отчеты ТЗ – результаты прохождения студентов тестирования для создания общей отчетности.

– Паспорт фонда ТЗ – список вопросов для тестирования по учебным дисциплинам различных направлений подготовки. [5]

Таблица 2.2 – Информационное обеспечение

| №№ | Наименование | Характеристика |
|---|---|---|
| 1. | Тестирование студентов | Отображение тестовых заданий для пользователей |
| 2. | Создание отчета | Обработка результатов по тестированию и вывод итоговой информации |

### 2.1.3 Обоснование проектных решений по программному обеспечению

Разработанная система тестирования не имеет определенных требований к операционной системе. Во время реализации использовалась операционная система семейства Windows: Windows 10. Для запуска программы необходимо установить любой браузер и иметь доступ в локальную сеть, в данном случае, университета.

В таблице 2.3 приведен список используемого программного обеспечения.

Таблица 2.3 – Программное обеспечение

| №№ | Наименование | Характеристика |
|---|---|---|
| 1. | Операционная система | Microsoft Windows 10:<br>Стандартные программы<br>Служебные программы |
| 2. | Инструментальное ПО | PHP, HTML, CSS, JavaScript, MySQL 5.6 |
| 3. | Пакет прикладных программ | Microsoft Office:<br>Microsoft Office Word 2016 |
| 4. | Файловые менеджеры | Far manager. Windows Commander |
| 5. | Графические редакторы | Adobe Photoshop |
| 6. | Текстовый редактор | Sublime Text 3 |
| 7. | Архиваторы | WinRAR |
| 8. | Информационно-поисковые системы | Google |
| 9. | Веб-сервер | Apache 2.4 |
| 10. | Система управления базами данных | PHPMyAdmin 4.6.6 |

### 2.1.4 Обоснование выбора программных средств

При разработке системы тестирования использовались следующие программные средства:

– Sublime Text – это удобный текстовый редактор.

Одним из положительных критериев является то, что разработчик позволяет скачать и пользоваться данным продуктом бесплатно и без ограничений. Так же редактор поддерживает большое количество языков программирования. Возможности продукта: быстрая навигация, командная палитра, одновременное редактирование, высокая степень настраиваемости.

Для пользования был выбран Sublime Text версии 3. В данной версии увеличилась скорости запуска программы, улучшился интерфейс, улучшилась производительность некоторых функций, появилась возможность индексирования файлов для поиска объявления переменных и функций.

– Apache – это свободный веб-сервер.

Отличается надежностью и гибкостью конфигурации. Apache поддерживает большое количество операционных систем и платформ. Данный веб-сервер распространяется бесплатно. Apache имеет различные механизмы обеспечения безопасности и разграничения доступа к данным.

Была выбрана версия Apache 2.4, так как в ней реализована возможность асинхронно проводить операции чтения и записи.

– PhpMyAdmin – система управления базами данных.

Написана на языке PHP, позволяющая управлять базами данных MySQL. Данное приложение пользуется популярностью у веб-разработчиков, так как позволяет управлять СУБД MySQL без непосредственного ввода SQL команд.

В PHPMyAdmin 4.6.6 исправлены ошибки прошлых версий. [18]

## 2.2 Информационное обеспечение задачи

### 2.2.1 Информационная модель и ее описание

При проектировании модели «КАК ДОЛЖНО БЫТЬ» были учтены недостатки перечисленных раннее авторских разработок. Система представлена ввиде нотаций IDEF0,DFD,IDEF3. [20,22]

На контекстной диаграмме IDEF0 «Система контроля знаний студентов» добавлена выходящая информация:

– сообщение об ошибке во время тестирования: студенты имеют возможность сообщить преподавателям о том, что необходимо внести правку в тест;

– отчет об успеваемости студента: позволяет студентам проанализировать свою успеваемость по определенным дисциплинам и определить процент усвоения материала (рисунок 2.3).

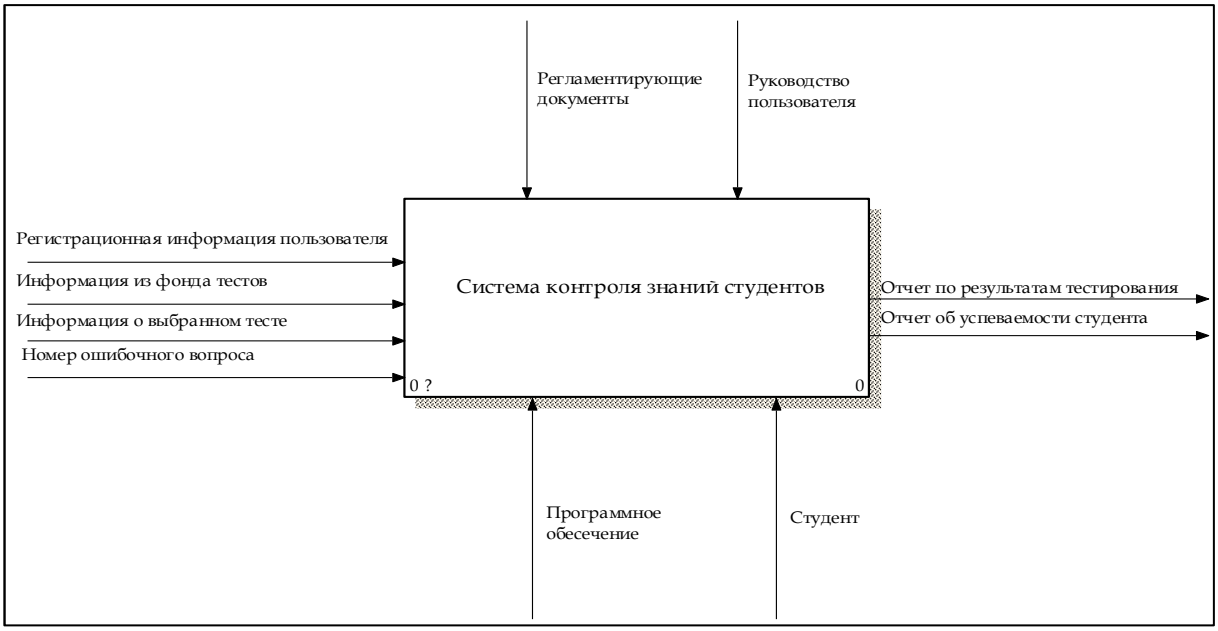



Рисунок 2.3 – Контекстная диаграмма

В результате декомпозиции были получены следующие функциональные блоки: «Аутентификация и авторизация пользователей», «Прохождение тестирования», «Обработка и анализ результатов тестирования», «Формирование отчета и вывод успеваемости студента» (рисунок 2.4).

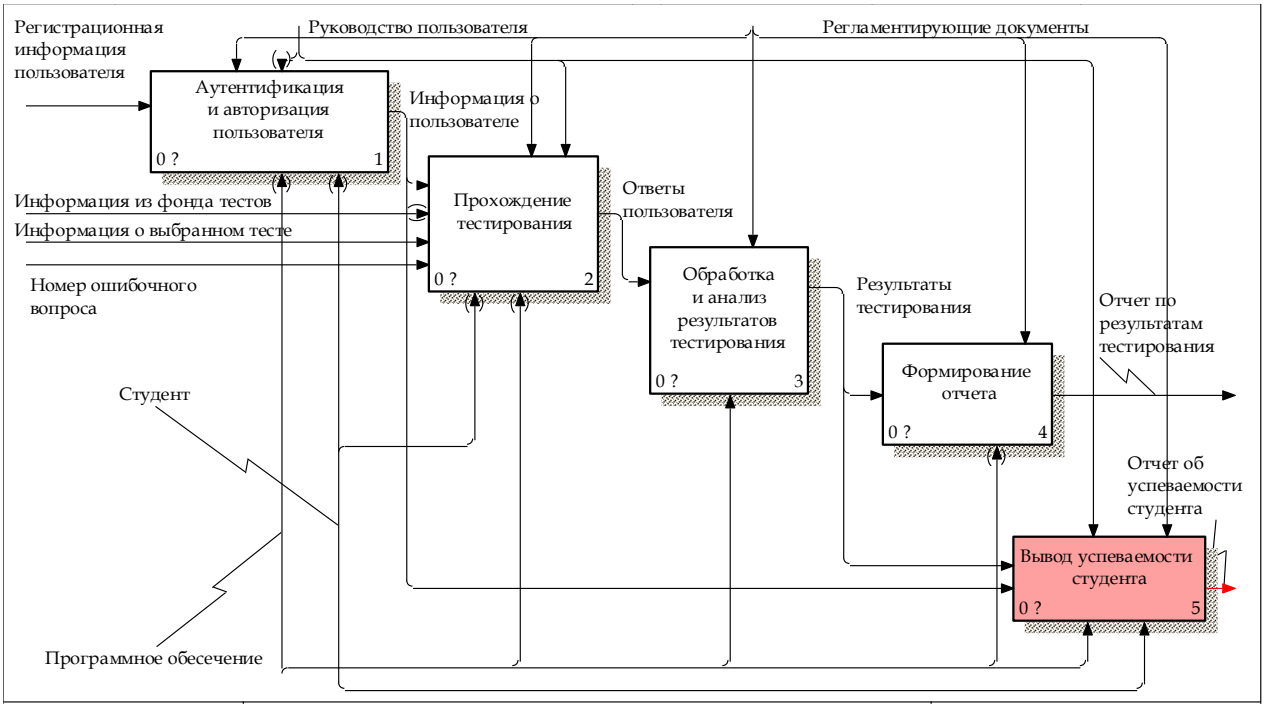



Рисунок 2.4 – Декомпозиции контекстной диаграммы

Декомпозиции всех функциональных блоков, кроме «Вывод успеваемости студента», приведены в Приложении 1.

Блок «Вывод успеваемости студента» для большей детализации процесса также был представлен в виде декомпозиции (рисунок 2.5).

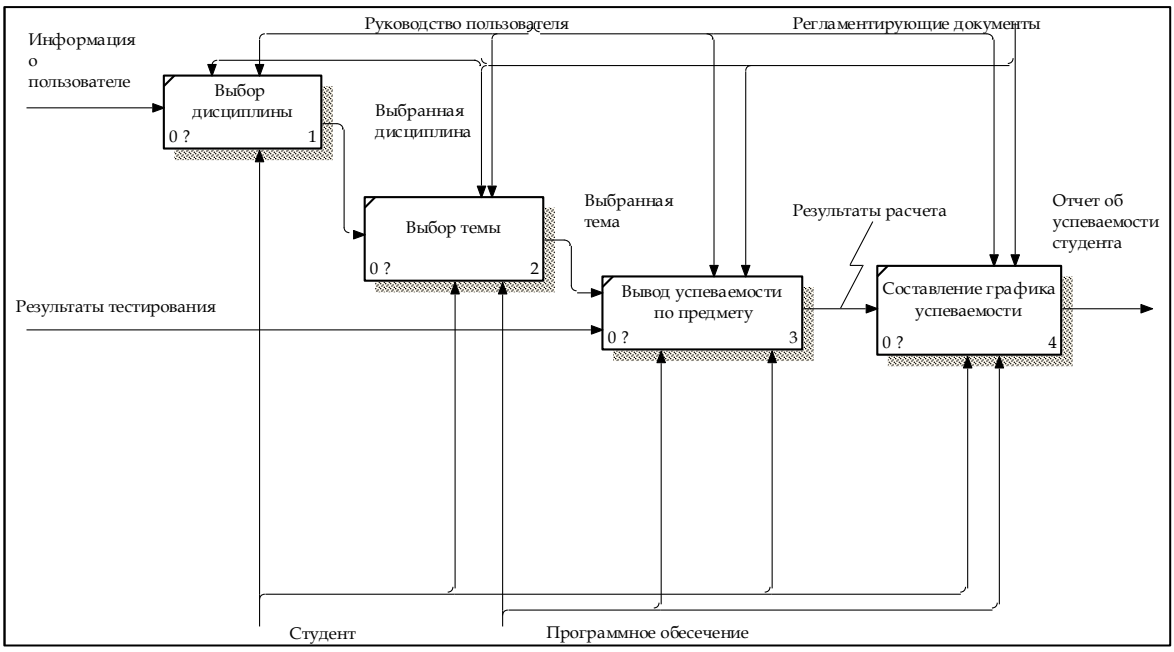



Рисунок 2.5 – Декомпозиции «Вывод успеваемости студента»

Данная диаграмма включает в себя выполнение следующих функциональных блоков: выбор дисциплины, выбор темы, расчет успеваемости по предмету и составление графика успеваемости.

На рисунке 2.6 изображена система в общем виде, с двумя внешними сущностями: студент и преподаватель.

Студент передает информацию: логин и пароль, свои ответы после прохождения тестирования, сообщения об ошибках в вопросах или ответах.

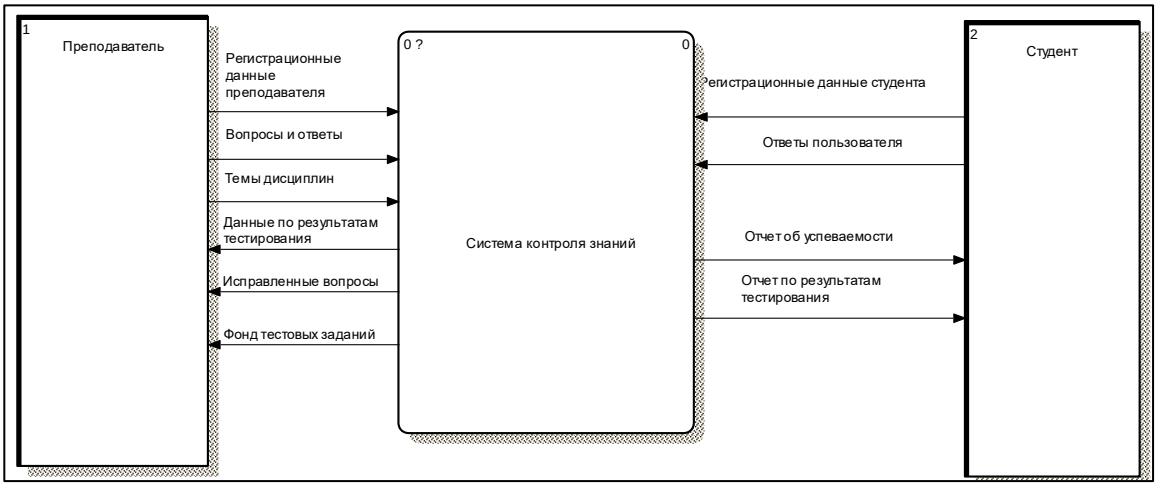



Рисунок 2.6 – Контекстная диаграмма «Система контроля знаний»

Диаграмма декомпозиции включает в себя выполнение модулей: заполнение фонда тестовых заданий и тестирование с последующим анализом данных (рисунок 2.7).

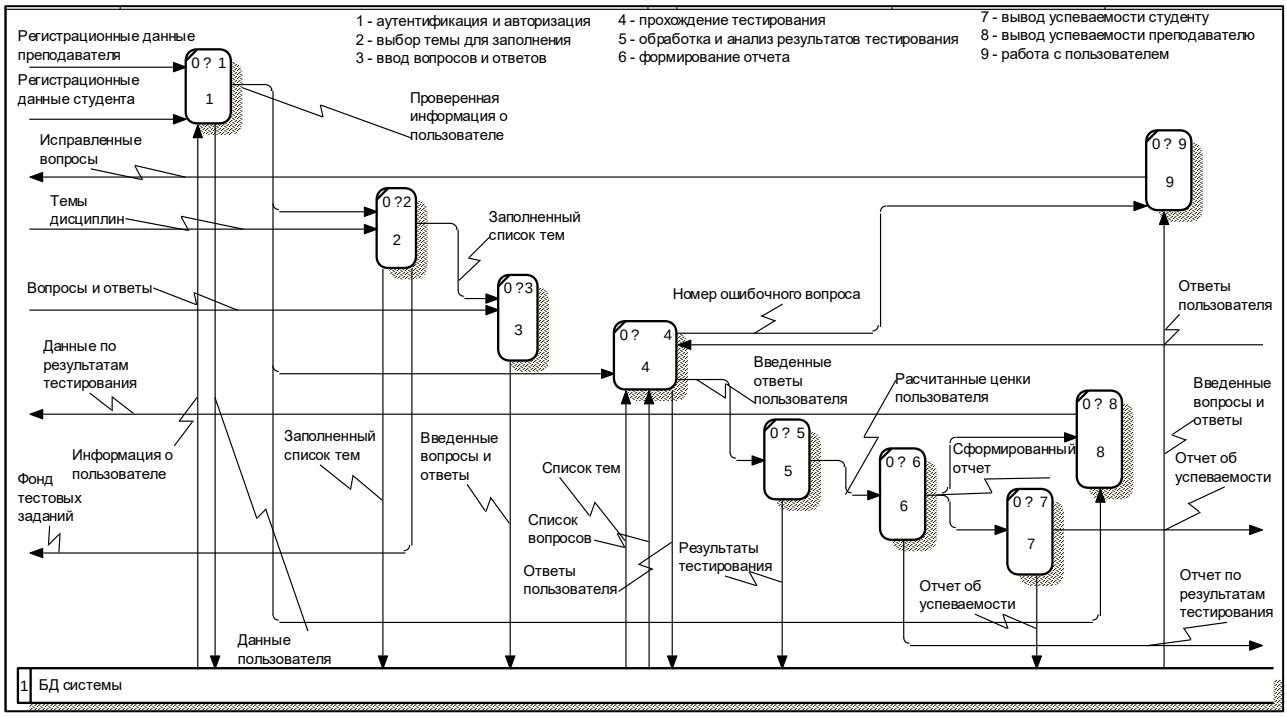



Рисунок 2.7 – Диаграмма декомпозиции «Система контроля знаний»

Студент проходит авторизацию, ему предоставляются права к определенным разделам информационного ресурса. При выборе темы, открывается бланк тестовых заданий. После прохождения тестирования рассчитывается итоговая оценка и выводится пользователю. При выборе раздела «Оценки» пользователь может посмотреть свою успеваемость и проанализировать какие именно темы и предметы ему необходимо изучить повторно.

Модель «КАК ДОЛЖНО БЫТЬ» отображает решение поставленной цели и задач выпускной квалификационной работы. [16,26]

### 2.2.2 Используемые классификаторы и системы кодирования

В состав информационного обеспечения входят классификаторы экономической информации и занимают в данном комплексе важное место.

Данные классификаторы обеспечивают сжатие показателей и тем самым сокращают объем хранимой информации в электронных вычислительных

машинах. Также сокращается время поиска информации, которое необходимо для решения задач. Классификация и кодирование информации облегчает обработку данных.

Кодирование – это процесс, при котором объектам присваиваются кодовые обозначения.

Цель кодирования: однозначное обозначение объектов и обеспечение достоверности закодированной информации.

Структура кодовых обозначений объектов приведена в таблице 2.4.

Таблица 2.4 – Структура кодовых обозначений

| Наименование кодируемого множества объектов | Значность кода | Система кодирования | Вид классификатора |
|---|---|---|---|
| Код пользователя | 11 | Порядковая | Локальный |
| Код ответа | 11 | Порядковая | Локальный |
| Код дисциплины | 11 | Порядковая | Локальный |
| Код ошибки | 11 | Порядковая | Локальный |
| Код избранного | 11 | Порядковая | Локальный |
| Код группы | 11 | Порядковая | Локальный |
| Код вопроса | 11 | Порядковая | Локальный |
| Код оценки | 11 | Порядковая | Локальный |
| Код темы | 11 | Порядковая | Локальный |
| Код типа вопроса | 11 | Порядковая | Локальный |
| Код типа пользователя | 11 | Порядковая | Локальный |

### 2.2.3 Характеристика базы данных

#### 2.2.3.1 Характеристика инфологической модели базы данных

Цель инфологического проектирования: предоставление информации, которая хранится в базе данных, в более естественном для восприятия виде.

Инфологическая модель строится по аналогии с естественным языком, так как он не может использоваться в чистом виде из-за сложности в компьютерной обработке текста и неопределенности любого обычного языка.

Основные конструктивные элементы инфологической модели: сущности, их атрибуты и связи.

Сущностью является различимый объект с дополнительной информацией, который хранится в базе данных.

Существуют два понятия: тип сущности и экземпляр сущности. Тип сущности относится к набору однородных данных, предметов, идей или каких-либо событий, которые выступают как целое. Экземпляр сущности относится к определенной вещи в наборе.

Сущности имеют уникальное имя, которое индивидуально в пределах разрабатываемой модели. Имя сущности – это имя типа, но не определенного экземпляра.

Сущности разделяются на слабые и сильные. Слабая сущность – это та сущность, которая имеет зависимость от другой сущности, имеющую большую силу по отношению к ней. Если запись из сильной сущности удаляется, то в подчиненной таблице эти данные также удаляются.

Любая сущность может подразделяться на два или несколько взаимоисключающих подтипов, у которых существуют связи и/или атрибуты.

Данные общие связи и/или атрибуты создаются один раз на более высоком уровне. Собственные связи и/или атрибуты могут определяться лишь в подтипах.

Супертип – это сущность, на основе которой в последствии определяются подтипы. Каждый экземпляр супертипа относится к некоторому подтипу, т.е. подтипы образовывают полное множество.

Атрибут – это характеристика сущности. Наименование атрибута уникально для определенного типа сущности. Атрибуты определяют какую информацию о сущности необходимо собрать. Абсолютного различия между типами и атрибутами сущностей не существует. Атрибут сущности является

таковым только в связи с типом. В другом контексте атрибут может выступать как отдельная сущность.

Ключ – это элементарный набор атрибутов, по содержаниям которых можно окончательно определить необходимый экземпляр сущности. Элементарность означает, что удаление из набора любого атрибута не позволяет распознать сущность по оставшимся атрибутам.

Связь – это ассоциированное сопоставление двух или более сущностей. Структура базы данных имела бы очень простую структуру, если бы была возможность хранить отдельные, не связанные между собой, данные. Однако основным требование к созданию базы данных является возможность обратиться от одной сущности по значению другой, поэтому установление связей является необходимой частью при проектировании модели. [11,18]

Сложность инфологической модели определяется по количеству существующих связей, так как в реальных базах данных часто содержится сотни сущностей, то теоретически между ними могут устанавливаться более тысячи связей.

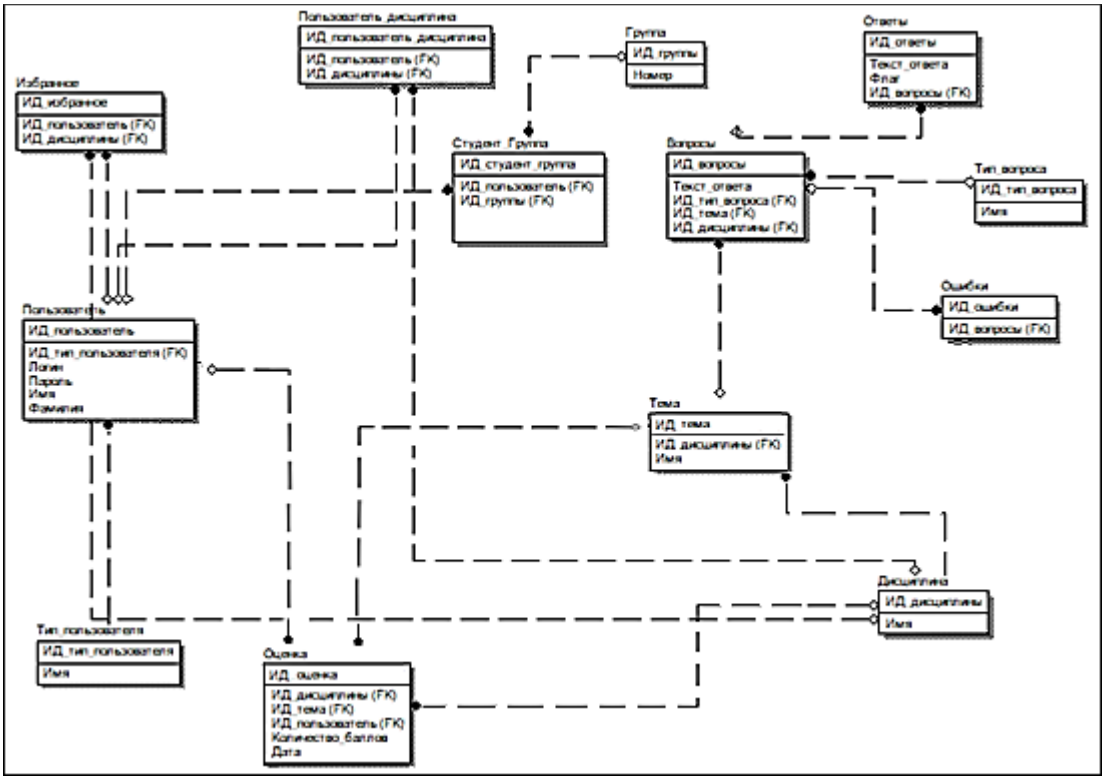



Рисунок 2.8 – Логическая модель БД

#### 2.2.3.2 Характеристика даталогической модели базы данных

Физическая реализация – реляционная модель. Данные представляются в виде таблиц. [23,29]

Структура таблицы «Answer» приведена в таблице 2.5.

Таблица 2.5 – Таблица «Answer»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| text_answer | varchar(200) | Нет |
| flag | tinyint(1) | Нет |
| id_question | int(11) | Нет |

Структура таблицы «Discipline» приведена в таблице 2.6.

Таблица 2.6 – Таблица «Discipline»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| name | varchar(100) | Нет |

Структура таблицы «Errors» приведена в таблице 2.7.

Таблица 2.7 – Таблица «Errors»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| question_id | int(11) | Нет |

Структура таблицы «Favourites» приведена в таблице 2.8.

Таблица 2.8 – Таблица «Favourites»

| Столбец | Тип | Null |
|---|---|---|
| Id | int(11) | Нет |
| discipline_id | int(11) | Нет |
| id_user | int(11) | Нет |

Структура таблицы «Group» приведена в таблице 2.9.

Таблица 2.9 – Таблица «Group»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| number | varchar(10) | Нет |

Структура таблицы «Type_question» приведена в таблице 2.10.

Таблица 2.10 – Таблица «Type_question»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| name | varchar(30) | Нет |

Структура таблицы «Question» приведена в таблице 2.11.

Таблица 2.11 – Таблица «Question»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| text_quest | varchar(500) | Нет |
| id_theme | int(11) | Нет |
| id_type | int(11) | Нет |
| id_discip | int(11) | Нет |

Структура таблицы «Score» приведена в таблице 2.12.

Таблица 2.12 – Таблица «Score»

| Столбец | Тип | Null | По умолчанию |
|---|---|---|---|
| id | int(11) | Нет | |
| id_stud | int(11) | Нет | |
| discipline_id | int(11) | Да | NULL |
| theme_id | int(11) | Да | NULL |
| number_score | float | Нет | |
| date | datetime | Да | current_timestamp |

Структура таблицы «Student_group» приведена в таблице 2.13.

Таблица 2.13 – Таблица «Student_group»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| id_stud | int(11) | Нет |
| id_group | int(11) | Нет |

Структура таблицы «Theme» приведена в таблице 2.14.

Таблица 2.14 – Таблица «Theme»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| name | varchar(100) | Нет |
| id_disc | int(11) | Нет |

Структура таблицы «Type_user» приведена в таблице 2.15.

Таблица 2.15 – Таблица «Type_user»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| name | varchar(15) | Нет |

Структура таблицы «User» приведена в таблице 2.16.

Таблица 2.16 – Таблица «User»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| login | varchar(20) | Нет |
| pass | varchar(30) | Нет |
| name | varchar(20) | Нет |
| surname | varchar(20) | Нет |
| type_id | int(11) | Нет |

Структура таблицы «User_discip» приведена в таблице 2.17.

Таблица 2.17 – Таблица «User_discip»

| Столбец | Тип | Null |
|---|---|---|
| id | int(11) | Нет |
| user_id | int(11) | Нет |
| discip_id | int(11) | Нет |

### 2.2.4 Характеристика результатной информации

Результатная информация – это та информация, которая была получена в результате обработки промежуточной и первичной информации, и в конечном

итоге используется для принятия решений. Обычно результатная информация отображается в виде отчетов о результатах деятельности.

Результирующей информацией данной работы является:

1) Сообщения преподавателям – это сообщения, которые преподаватель получает от студентов, если во время тестирования встречаются ошибки. Данные берутся из таблицы ошибки.

2) Отчет по результатам тестирования – это информация, которая выводится пользователю по окончанию прохождения тестирования. Таблица: оценка, пользователь.

3) Результаты тестирования – данные, которые отправляются преподавателям для учета степени усвоения материала. Таблица: оценки, пользователь.

4) Отчет об успеваемости студента – вывод информации в виде графика. Таблицы: оценки, дисциплина, студент, тема.

# 3 Программная реализация проектных решений

## 3.1 Программное обеспечение задачи

### 3.1.1 Дерево функций и сценарий диалога

Дерево функций предназначено для описания структуры функций в виде иерархии процессов, протекающих на предприятии.

Функция – это операция, действие или задача, которые выполняются над объектом для достижения целей.

Система содержит дерево функций с двумя частями: основные и вспомогательные функции (рисунок 3.9).

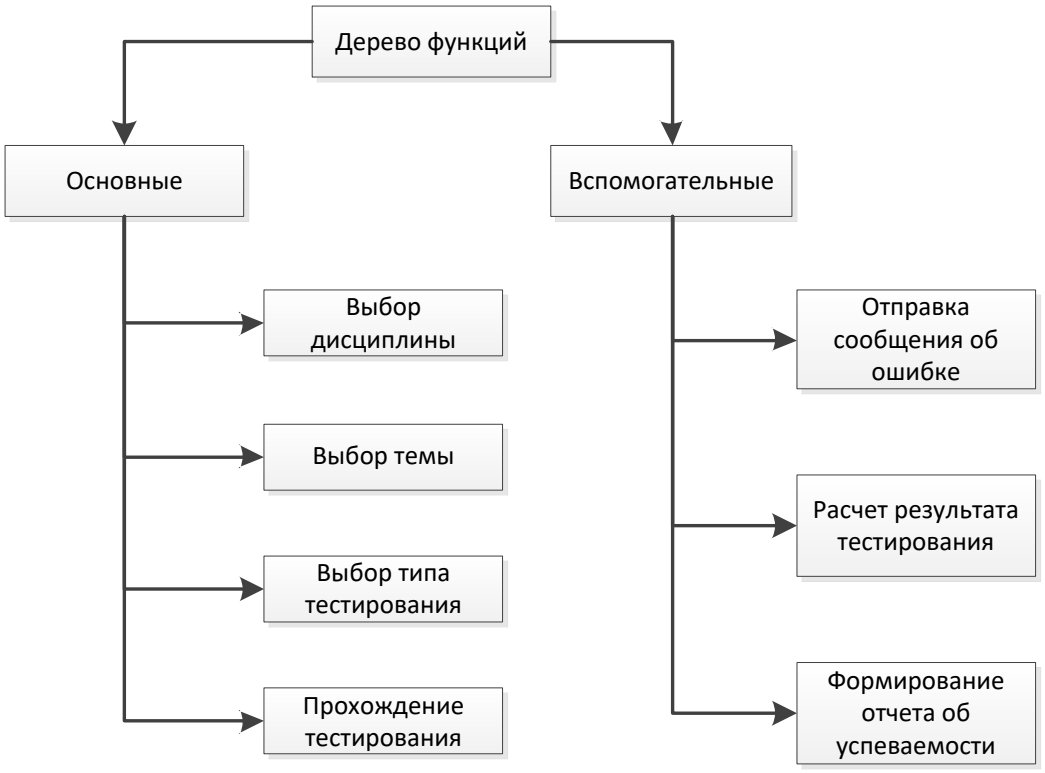



Рисунок 3.9 – Дерево функций

К основным функция относятся функции, которые используются студентом для прохождения тестирования: выбор дисциплины, выбор темы тестирования, выбор типа теста и сам процесс тестирования. К вспомогательным

функциям относятся функции, которые выводят дополнительную информацию пользователю, например: отправка сообщения об ошибке преподавателю, расчет результата тестирования и формирование отчета об успеваемости в виде столбчатой диаграммы.

## 3.2 Организация технологии сбора, обработки и выдачи информации

Для сбора, передачи, обработки и выдачи информации в системе использованы возможности языка программирования PHP, СУБД PhpMyAdmin с использованием языка запросов SQL.

В качестве шаблона проектирования была выбрана схема разделения данных MVC. Концепция шаблона заключается в разделении бизнес-логики и визуализации проекта. Название MVC является аббревиатурой составных частей: Model (Модель), View (Представление), Controller (Контроллер). Каждый компонент отвечает за свою задачу.

Модель управляет поведением и данными домена приложения, отвечает на запросы информации о его состоянии (обычно из представления) и отвечает на инструкции по изменению состояния (обычно с контроллера).

Представление управляет отображением информации.

Контроллер интерпретирует входные данные мыши и клавиатуры от пользователя, информируя модель и/или вид, чтобы изменить их соответствующим образом. В каждом контроллере есть несколько функций-экшенов, таких как “actionView”. [31]

Взаимодействия компонентов представлены на рисунке 3.10.

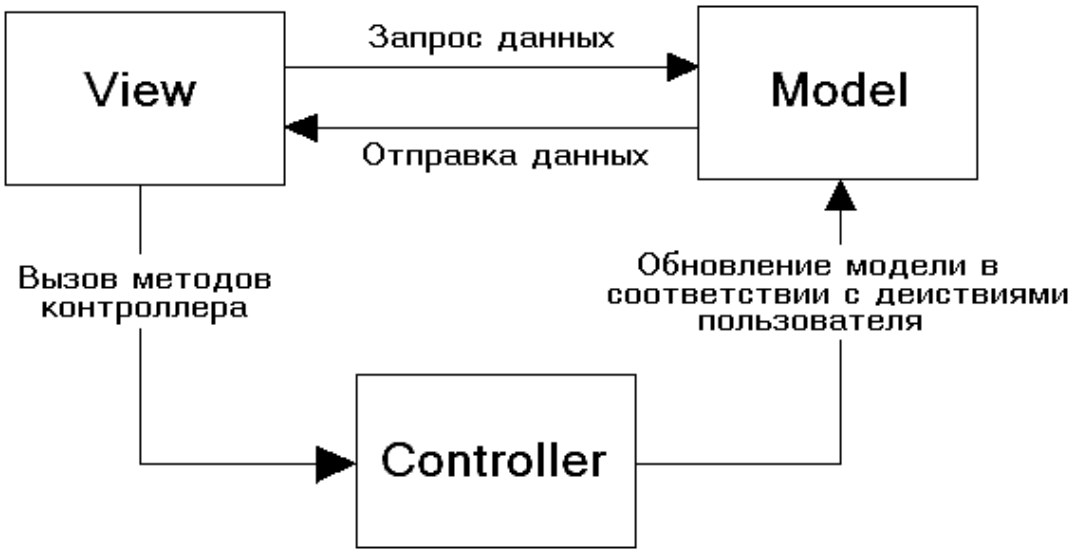



Рисунок 3.10 – Взаимодействие компонентов

Для работы шаблона требуется файл Router.php – роутер. Функции в данном файле позволяют определить по тексту в адресной строке, какой контроллер требуется вызвать и какой эшкн должен сработать. Для работы роутера также требуется создать файл с массивом маршрутов, которые могут быть введены в адресную строку. Пример массива показан на листинге 3.1.

Листинг 3.1 – Массив данных

```
return array(

    'alltest/([0-9]+)course'=> 'test/runAll/$1',
    'runtest/theme([0-9]+)'=> 'test/run/$1',
    'test/([0-9]+)/theme([0-9]+)'=> 'test/view/$1/$2',
    'warning([0-9]+)' =>'test/warning/$1',


    '([0-9]+)/theme-([0-9]+)' => 'theme/view/$1/$2',
    'theme/add/([0-9]+)' => 'theme/add/$1',
    'theme-([0-9]+)/del/([0-9]+)' => 'theme/delete/$1/$2',
    'theme([0-9]+)/edit' => 'theme/edit/$1',
```

Первой частью массива является текст, вводимый в адресную строку, а вторая подчиняется правилу: контроллер/эшкн/параметр. При помощи такого

правила роутер быстро определяет нужный контроллер, экшн и передает в них параметры. [28]

После определения данных роутер вызывает требуемый файл. Пример файла находится на листинге 3.2.

Листинг 3.2 – Вызов роутером файла

```php
<?php
include_once ROOT. '/models/Course.php';
include_once ROOT. '/models/User.php';
include_once ROOT. '/models/Test.php';

class TestController
{
    public function actionView($course_id, $theme_id)
    {
        $id_user=$_SESSION["user"];
        $hname="Просмотр темы";
        $themeList=array();
        $themeScore=Test::getThemeScore($theme_id, $id_user);
        $user=User::getUserById($id_user);

        require_once(ROOT . '/views/test/view.php');
    }
```

Контроллер является классом php, а экшн – функция. Параметры, передаваемые роутером принимается функций-экшеном и могут быть использованы внутри нее. Строка require_once (адрес файла) отвечает за вызов представления, находящегося в определенной папке. Название представления не зависит от названия котроллера или экшена.

Для вызова модели необходимо обратиться к функции класса, находящемуся в предварительно подключенных файлах. Пример файла-модели изображен на листинге 3.3.

Листинг 3.3 – Обращение к функции класса

```php
<?php
class User
{

    public static function checkUserData($login, $pass)
    {

        $db = Db::getConnection();

        $sql = 'SELECT * FROM user WHERE login = :login AND pass = :pass';

        $result = $db->prepare($sql);
        $result->bindParam(':login', $login, PDO::PARAM_INT);
        $result->bindParam(':pass', $pass, PDO::PARAM_INT);
        $result->execute();

        $user = $result->fetch();
        if ($user) {

            return $user['id'];
        }
        return false;
    }
```

При помощи модели также происходит подключение к базе данных и работа с ней. Функции класса возвращают либо массив значений, либо true, что значит успешное её выполнение. [21,27]

Для получения вводимых пользователем данных в представлении размещены формы с методом POST, позволяющим передавать данные, не используя строку запроса. Для обработки формы контроллер имеет условие, по которому определяется, была ли нажата кнопка отправки. Если кнопка сработала, то из глобальной переменной POST получаются данные, отправленные формой. Далее эти данные могут быть использованы для работы с базой данных или представлением.

Для работы с базой данных используется расширение языка PHP – PDO (PHP Data Objects). Данное расширение позволяет работать с разными базами данных и поддерживает стандарты ISO и ANSI. PDO обеспечивает высокую производительность благодаря работе с базой данных напрямую. Массив для подключения к базе данных изображен на листинге 3.4.

Листинг 3.4 – Массив с параметрами подключении к базе данных

```
// Массив с параметрами подключения к базе данных
return array(
    'host' => '172.23.64.64',
    'dbname' => 'db_14759954',
    'user' => '14759954',
    'password' => 'Onlybsu1!',
);
```

Подключение к базе данных происходит в функции «getConnection» в классе «Db». Процесс подключения приведен на листинге 3.5.

Листинг 3.5 – Подключение к базе данных

```
<?php

class Db
{

    public static function getConnection()
    {
        $paramsPath = ROOT . '/config/db_params.php';
        $params = include($paramsPath);


        $dsn = "mysql:host={$params['host']};dbname={$params['dbname']};charset=cp1251";
        $db = new PDO($dsn, $params['user'], $params['password']);

        return $db;
    }
}
```

Также расширение защищено от sql-инъекций посредством заранее подготовленных SQL-выражений. Оно записывается в переменную, после подставляются параметры, если требуется, и выполняется. Результатом выполнения запроса на выборку является массив, а на работу с таблицей – логическая переменная со значением true или false. Выполнение запроса на выборку представлено на листинге 3.6. [14]

Листинг 3.6 – Запрос на выборку

```
    public static function getDisciplineScore ($discipline_id, $id_stud)
{
        $db = Db::getConnection(); // Подключене к БД
        $disciplineScore=array(); // Инициализация массива
        $result= $db->prepare('SELECT * FROM score WHERE id_stud=:id_stud AND discipline_id=:discipline_id');// Подготовка запроса
        $result->bindParam(':id_stud', $id_stud, PDO::PARAM_INT); // Подстановка параметров
        $result->bindParam(':discipline_id', $discipline_id, PDO::PARAM_INT);
        $result->execute(); // Выполнение запроса
        $i=0;
        while ($row=$result->fetch()) { // Присваивание элементов массиву
            $disciplineScore[$i]['number_score']=$row['number_score'];
            $disciplineScore[$i]['date']=$row['date'];
            $i++;
        }

        return $disciplineScore;
}
```

Если выборку требуется отобразить на экране пользователя, то используется цикл foreach, перебирающий массив поэлементно, что позволяет отображать таблицу построчно. Каждая переменная из массива может быть выведена на экран с помощью функции echo. Пример работы с циклом приведен на листинге 3.7.

Листинг 3.7 – Пример работы с циклом

```
    <?php
    $i=0;
    foreach ($themeScore as $dateItem): ?> //Запуск цикла
    <td><?php echo $dateItem['date'];?></td> // Вывод значений
    <td><?php echo $dateItem['number_score']; $i++;?></td>
</tr>
    <?php endforeach;?>
</table>
```

Если пользователь перешел на страницу тестирования, то для него выводится список вопросов с ответами. Вопросы могут быть 3 типов: с одним вариантом ответа, множественным выбором или вопросы с ручным вводом. Для каждого типа представлен свой вид ввода: checkbox, radio или поле ввода. Тип вопроса определялся в каждом проходе цикла foreach, а выбор типа ввода – условием, срабатывающим по номеру типа. [7,8,24]

После того как пользователь завершил тестирование – система подсчитывает количество набранных баллов. Для этого за максимальное берется 100 баллов и делится на количество вопросов в тесте. При этой, если вопрос имеет тип «множественный выбор», количество баллов за один вопрос делится на количество правильных ответов. После этого баллы за правильные ответы пользователя суммируются и заносятся в таблицу.

Для авторизации пользователя в системе использовалась работа с формами, базой данных и массив SESSION. После получения данных из полей формы на странице авторизации и их проверки – они попадают в суперглобальный массив. Функция присваивания полю суперглобального массива значения представлена на листинге 3.8.

Листинг 3.8 – Функция присваивания

```
public static function auth($userId)
{
    // Записываем идентификатор пользователя в сессию
    $_SESSION['user'] = $userId;
}
```

Данные из массива можно использовать в любой части системы. При помощи них определяется тип пользователя, который позволяет разграничить доступ к различным данным на странице. [1-3]

## 3.3 Описание контрольного примера реализации проекта

Для начала работы требуется ввести данные в форме авторизации на сайте. Окно авторизации изображено на рисунке 3.11.


Рисунок 3.11 – Окно авторизации

После успешной авторизации пользователь попадает на страницу со списком всех доступных курсов. Данная страница отображена на рисунке 3.12.

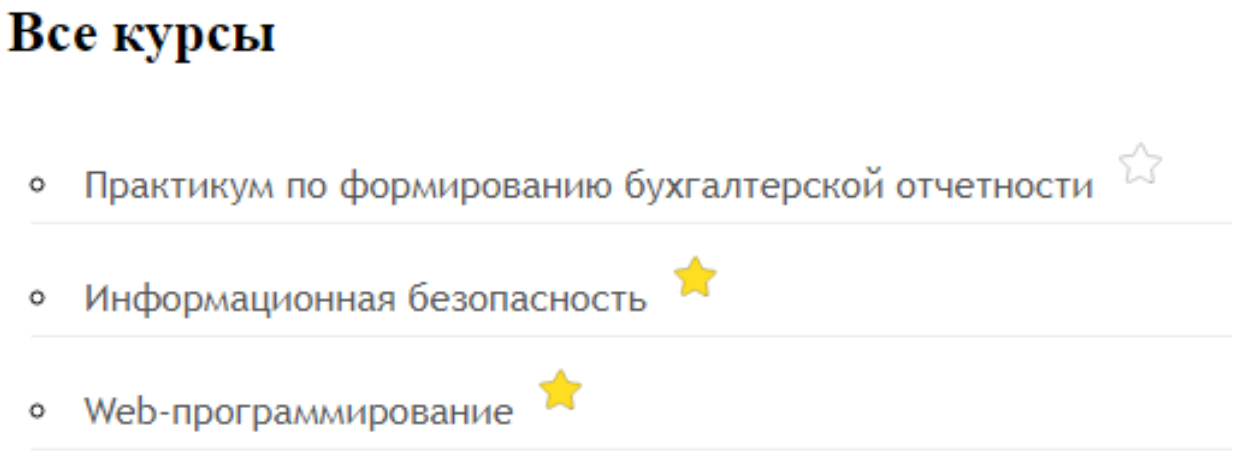



Рисунок 3.12 – Список курсов

Около ссылок на каждый курс расположена кнопка добавления или удаления избранного. Данная функция позволяет отобрать самые важные курсы по выбору пользователя, которые после этого отображаются в разделе меню. Меню изображено на рисунке 3.13.

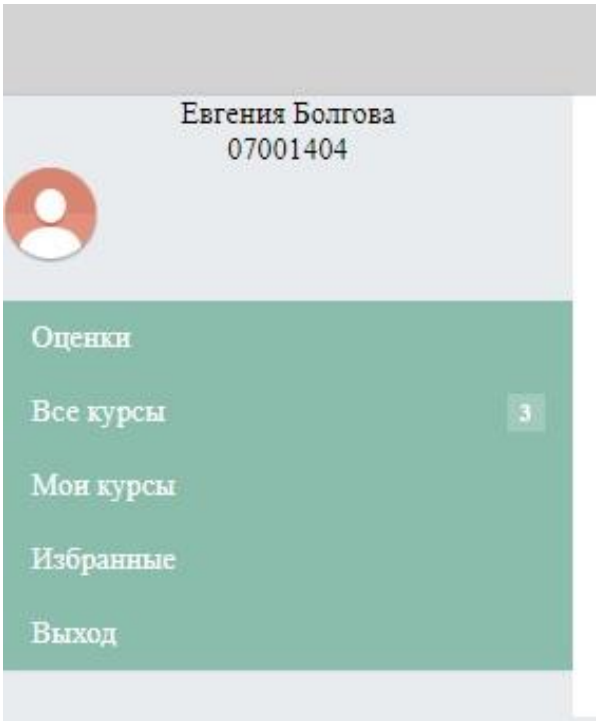



Рисунок 3.13 – Меню пользователя

Меню имеет раскрывающиеся списки, позволяющие скрыть содержащуюся в них информацию. Около подпункта «Все курсы» находится блок со счетчиком количества доступных курсов.

Для просмотра информации о дисциплине – необходимо перейти по ссылке с названием курса.

Содержимое страницы курса представлено на рисунке 3.14.

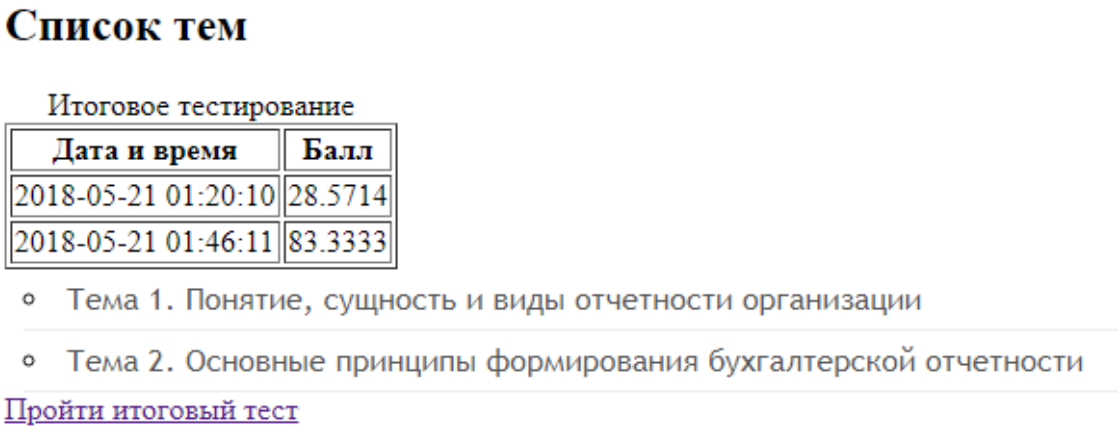



Рисунок 3.14 – Информация о дисциплине

Страница содержит в себе список доступных для тестирования тем, ссылку на прохождение итогового теста и таблицу с оценками пользователя после прохождения итогового теста. С данной страницы можно перейти на страницу темы, которая представлена на рисунке 3.15.

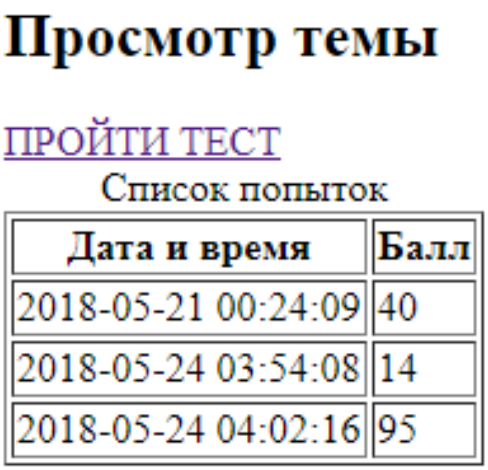



Рисунок 3.15 – Оценки пользователя

На странице отображаются оценки пользователя по тестам по теме и ссылка на прохождение теста. Для того, чтобы начать тестирование – требуется перейти по ссылке. Лист тестирования представлен на рисунке 3.16.

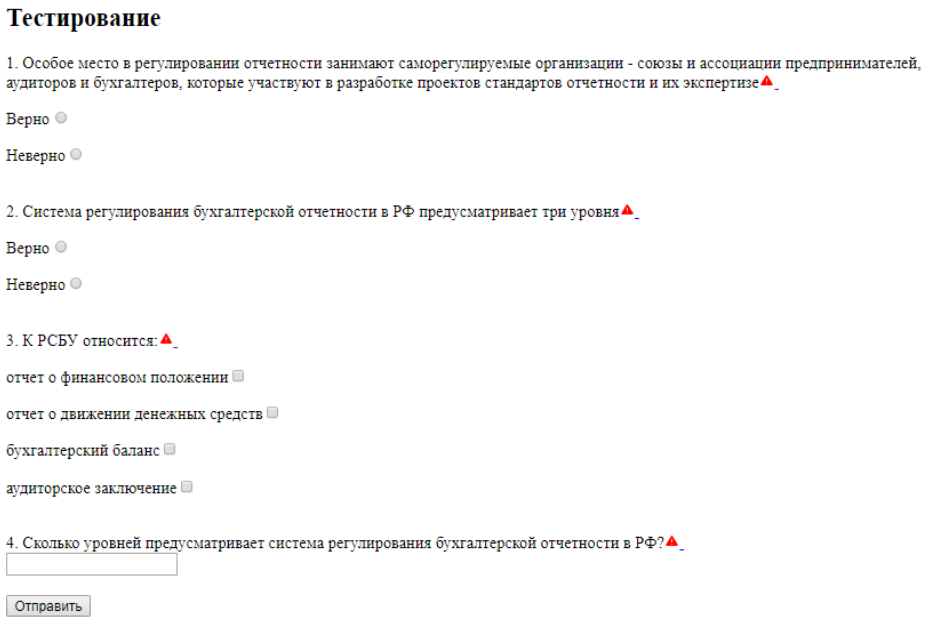



Рисунок 3.16 – Лист теста

Около каждого вопроса в листе тестов находится иконка красного цвета, которая позволяет отправить преподавателю сообщение об ошибке в вопросе или ответе. Кнопка «Отправить» запускает процесс проверки тестов и занесения данных в базу данных. После этого пользователь возвращается в выбор всех курсов.

Итоговое тестирование отличается от тестирования по теме тем, что пользователь получает вопросы из всех тем курса. Внешне страница отличается названием и количеством вопросов.

Данная страница представлена на рисунке 3.17.

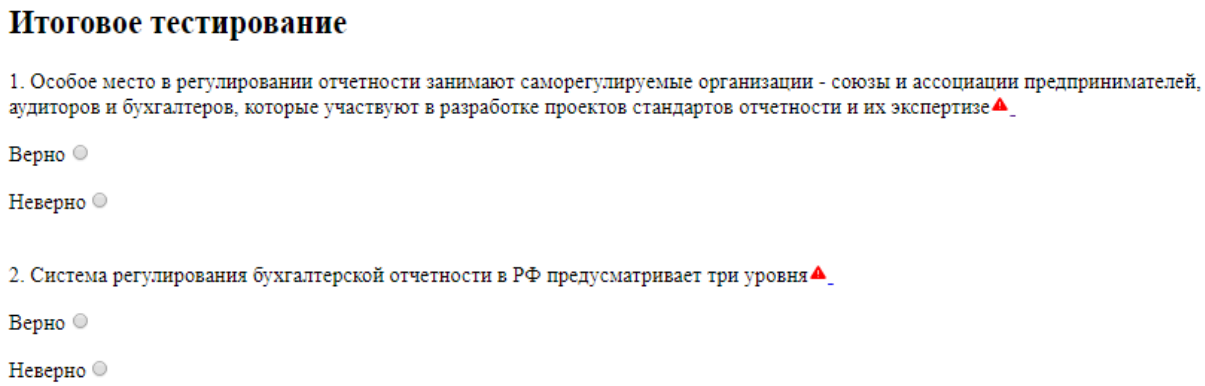



Рисунок 3.17 – Итоговое тестирование

Пункт меню «Оценки» дает возможность увидеть успеваемость пользователя по всем курсам в графическом виде. Страница со столбчатой диаграммой представлена на рисунке 3.18. [9,12,13,17]

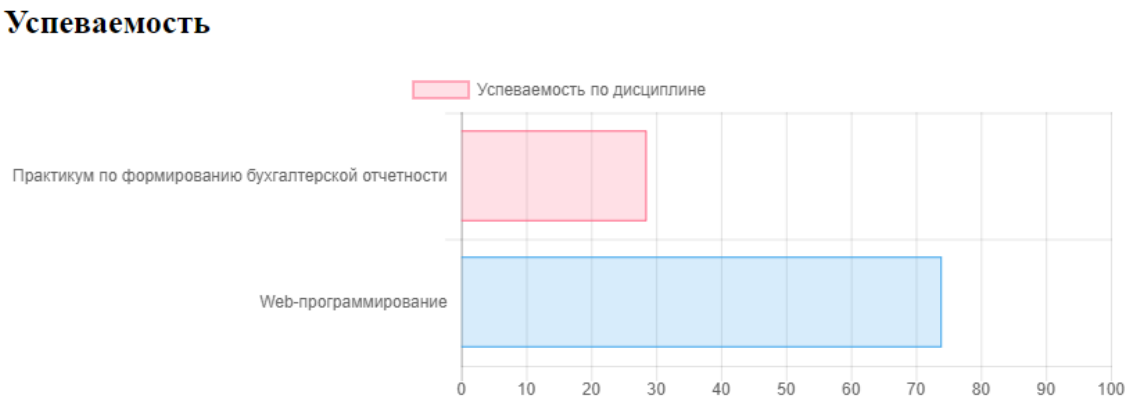



Рисунок 3.18 – Успеваемость студента

После окончания работы с системой пользователь может выйти со своей страницы при помощи кнопки меню «Выход». Данная кнопка переводит пользователя на окно авторизации. Окно представлено на рисунке 3.19.

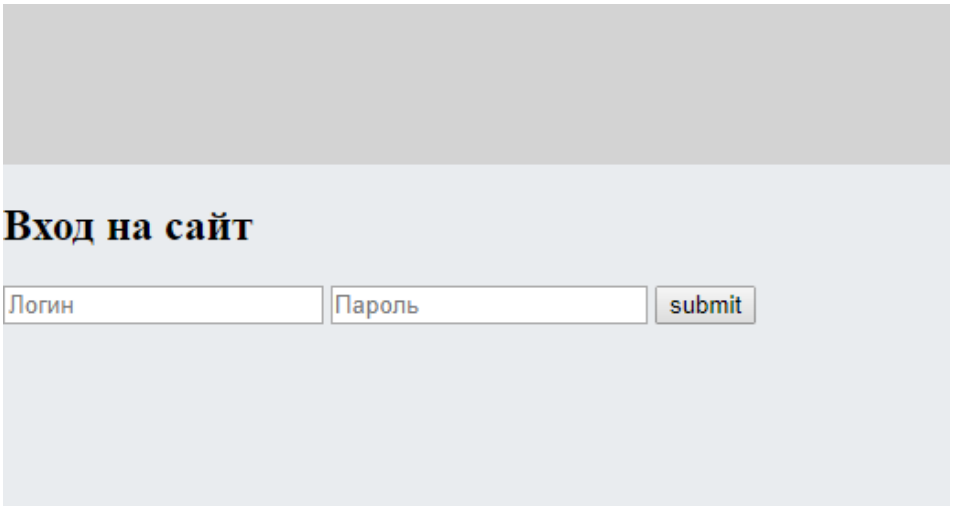



Рисунок 3.19 – Выход с учетной записи

После того, как пользователь вышел – другой человек может ввести свои данные для работы с системой через свой аккаунт.

## 3.4 Обоснование эффективности

Для обоснования эффективности использовалась программа управления проектами Microsoft Project. Данный программный комплекс удобен при реализации больших проектов, при построении бизнес-планов.

Список необходимых этапов для разработки системы информационного сопровождения процесса тестирования представлен в таблице 3.18.

Таблица 3.18 – Этапы разработки системы

| **Разработка системы информационного сопровождения процесса тестирования. Модуль контроля знаний студентов и анализа полученных результатов** |
|---|
| Обоснование необходимости создания АС и изучение предметной области |
| Формирования требований к АС |
| Разработка и утверждение ТЗ |
| Разработка модуля тестирования и обработки данных по результатам тестирования |
| Разработка рабочей документации |
| Адаптация АИС и её частей |
| Подготовка объекта автоматизации к вводу АС в действие |
| Подготовка персонала |
| Комплектация АИС |
| Пусконаладочные работы |
| Проведение предварительных испытаний |
| Проведение опытной эксплуатации |
| Проведение приемочных испытаний |

Календарное планирование – это процесс создания и редактирование графика работ, которые выполняются различными подразделениями или организациями, и являются зависимыми по времени, по обеспечению трудовыми и техническими ресурсами.

План работ отображен в таблице 3.19.

Таблица 3.19 – Календарное планирование

| Название задачи | Длитель | Начало | Окончание |
|---|---|---|---|
| **Разработка системы инфорационного сопровождения процесса тестирования. Модуль контроля знаний студентов и анализа полученных результатов** | **120 дней** | **Пн 15.01.18** | **Пт 08.06.18** |
| Обоснование необходимости создания АС и изучение предметной области | 5 дней | Пн 15.01.18 | Пт 19.01.18 |
| Формирования требований к АС | 4 дней | Сб 20.01.18 | Чт 25.01.18 |
| Разработка и утверждение ТЗ | 3 дней | Чт 25.01.18 | Вт 30.01.18 |
| Разработка модуля тестирования и обработки данных по результатам тестирования | 20 дней | Вт 30.01.18 | Пт 23.02.18 |
| Разработка рабочей документации | 15 дней | Пт 23.02.18 | Ср 14.03.18 |
| Адаптация АИС и её частей | 12 дней | Ср 14.03.18 | Чт 29.03.18 |
| Подготовка объекта автоматизации к вводу АС в действие | 8 дней | Чт 29.03.18 | Пн 09.04.18 |
| Подготовка персонала | 10 дней | Пн 09.04.18 | Сб 21.04.18 |
| Комплектация АИС | 4 дней | Сб 21.04.18 | Чт 26.04.18 |
| Пусконаладочные работы | 7 дней | Пт 27.04.18 | Сб 05.05.18 |
| Проведение предварительных испытаний | 13,5 дней | Сб 05.05.18 | Пн 21.05.18 |
| Проведение опытной эксплуатации | 10 дней | Пн 21.05.18 | Сб 02.06.18 |
| Проведение приемочных испытаний | 5 дней | *Сб 02.06.18* | Пт 08.06.18 |

Длительность проекта составляет 120 дней.

Планирование ресурсов с помощью MS Project имеет несколько положительных критериев: распределение ресурсов происходит только при их наличии и новые задачи назначаются при реальном оценивании текущих процессов. Ресурсы могут иметь приоритет и компьютерный график легко перестроить. Таким образом, компании имеют возможность систематизировать ресурсы при их ограниченном количестве

Ресурсы имеют три типа: технические, материальные и трудовые. Для реализации данного проекта использовались только трудовые и материальные. В таблице 3.20 предоставлен список ресурсов, указан тип и стандартная ставка каждого ресурса.

Таблица 3.20 – Список необходимых ресурсов

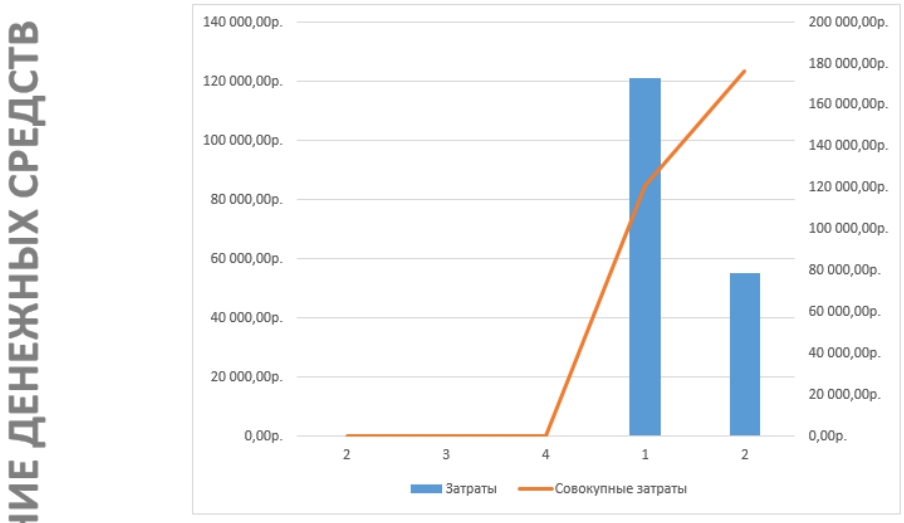
| Название ресурса | Тип | Стандартна ставка |
|---|---|---|
| Разработчики | Трудовой | 200,00р./ч |
| Компьютеры | Трудовой | 30,00р./ч |
| Данные учебных планов | Трудовой | 2,00р./ч |
| Данные карт компетенций | Трудовой | 2,00р./ч |
| Данные ФГОС | Трудовой | 2,00р./ч |
| Проект интерфейса | Трудовой | 10,00р./ч |
| База данных | Трудовой | 10,00р./ч |
| Преподаватели | Трудовой | 100,00р./ч |
| Студенты | Трудовой | 0,00р./ч |
| CD диск | Материальный | 40,00р. |
| Переплет | Материальный | 200,00р. |
| Печать | Материальный | 60,00р. |

Затраты – размер ресурсов, измеренных в денежной форме, использованных в процессе деятельности за определённый промежуток времени. Список затрат на каждый этап реализации проекта приведен в таблице 3.21. [6]

Таблица 3.21 – Затраты на каждый этап разработки

| Название задачи | Общие затраты |
|---|---|
| **Разработка системы информационного сопровождения процесса тестирования. Модуль контроля знаний студентов и анализа полученных результатов** | **176 240,00р.** |
| Обоснование необходимости создания АС и изучение предметной области | 12 000,00р. |
| Формирования требований к АС | 6 400,00р. |
| Разработка и утверждение ТЗ | 4 800,00р. |
| Разработка модуля тестирования и обработки данных по результатам тестирования | 40 960,00р. |
| Разработка рабочей документации | 27 600,00р. |
| Адаптация АИС и её частей | 23 040,00р. |
| Подготовка объекта автоматизации к вводу АС в действие | 14 720,00р. |
| Подготовка персонала | 11 200,00р. |
| Комплектация АИС | 6 400,00р. |
| Пусконаладочные работы | 14 000,00р. |
| Проведение предварительных испытаний | 4 320,00р. |
| Проведение опытной эксплуатации | 3 200,00р. |
| Проведение приемочных испытаний | 1 600,00р. |

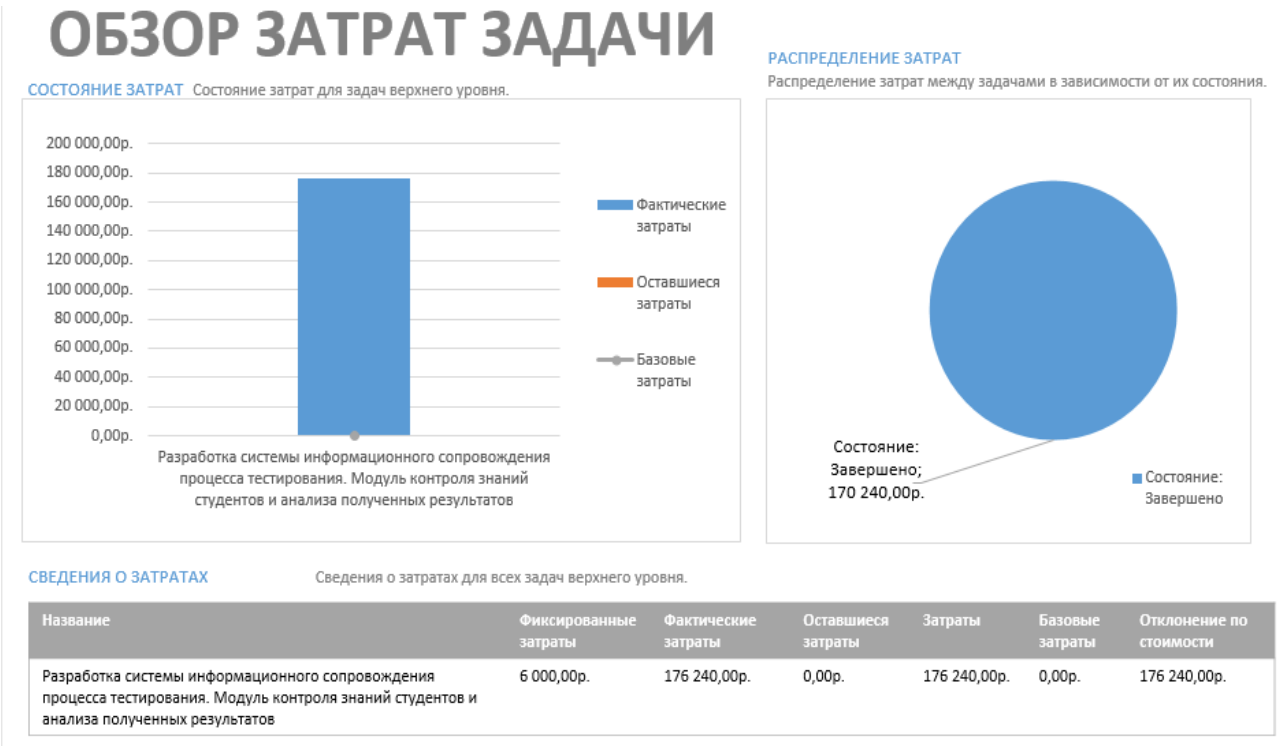
Движение денежных средств при разработке системы показано на рисунке 3.20.




| Название | Затраты |
|---|---|
| Разработка системы информационного сопровождения процесса тестирования. Модуль контроля знаний студентов и анализа полученных результатов | 176 240,00р. |

Рисунок 3.20 – Движение денежных средств

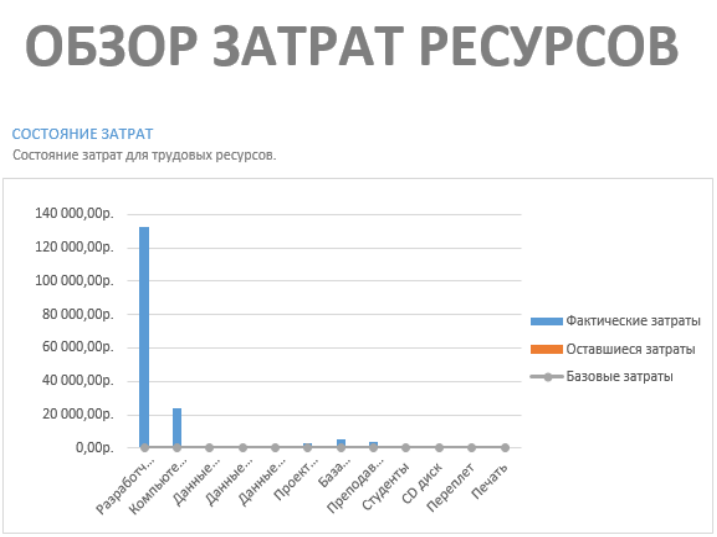
Затраты на выполнение задач представлены в виде отчета на рисунке 3.21.




| Название | Фиксированные затраты | Фактические затраты | Оставшиеся затраты | Затраты | Базовые затраты | Отклонение по стоимости |
|---|---|---|---|---|---|---|
| Разработка системы информационного сопровождения процесса тестирования. Модуль контроля знаний студентов и анализа полученных результатов | 6 000,00р. | 176 240,00р. | 0,00р. | 176 240,00р. | 0,00р. | 176 240,00р. |

Рисунок 3.21 – Отчет о затратах на задачи

Затраты на ресурсы представлены в виде отчета, который изображен на рисунке 3.22.




СВЕДЕНИЯ О ЗАТРАТАХ

Сведения о затратах для всех трудовых ресурсов.

| Название | Фактические трудозатраты | Фактические затраты | Стандартная ставка |
|---|---|---|---|
| Разработчики | 664 ч | 132 800,00р. | 200,00р./ч |
| Компьютеры | 804 ч | 24 120,00р. | 30,00р./ч |
| Данные учебных планов | 160 ч | 320,00р. | 2,00р./ч |
| Данные карт компетенций | 160 ч | 320,00р. | 2,00р./ч |
| Данные ФГОС | 160 ч | 320,00р. | 2,00р./ч |
| Проект интерфейса | 296 ч | 2 960,00р. | 10,00р./ч |
| База данных | 540 ч | 5 400,00р. | 10,00р./ч |
| Преподаватели | 40 ч | 4 000,00р. | 100,00р./ч |
| Студенты | 380 ч | 0,00р. | 0,00р./ч |
| CD диск | 0 | 0,00р. | 40,00р. |
| Переплет | 0 | 0,00р. | 200,00р. |
| Печать | 0 | 0,00р. | 60,00р. |

Рисунок 3.22 – Отчет о затратах на ресурсы

Итоговые затраты составляют 176 240,00 р.

Данная система не имеет определенных показателей экономической эффективности, так как основным требование к системе является удобство и упрощение работы.

# ЗАКЛЮЧЕНИЕ

В настоящее время автоматизированный контроль знаний является неотъемлемой частью обучения в высших учебных заведениях. Автоматизированные системы заменяют все большее количество традиционных методов оценки знаний, что объясняет создание новых технологий.

В ходе выполнения выпускной квалификационной работы была изучена предметная область и проанализированы аналогически разработанные системы. На основании рассмотренного материала были сформированы требования к системе информационного сопровождения процесса тестирования, определены цели и задачи выпускной квалификационной работы.

Для разработки системы было выбрано следующее инструментальное обеспечение: PHP, HTML, CSS, JavaScript, MySQL 5.6. PHP – это скриптовый язык программирования, который был выбран из-за его простоты в изучении и свободной типизации данных. Так же PHP является отличным инструментом шаблонизации.

Данная система была реализована для Белгородского государственного национального исследовательского университета с учетом всех требований, а также имеет возможность работать по локальной сети вуза. Тестовая оболочка имеет удобный и понятный интерфейс, который адаптирован под обычных пользователей.

Созданный продукт позволяет сократить время поиска учебных курсов, а также предоставить пользователю визуальную информацию о результатах тестирования для легкого восприятия собственной успеваемости. Во избежание ошибок в процессе тестирования, у студентов имеется возможность сообщить преподавателю о существующих неполадках. В экономическом разделе была обоснована простота и невысокая стоимость разработанной программы.

В ходе выполнения выпускной квалификационной работы были решены поставленные задачи, а также получены следующие результаты:

− изучена предметная область;

− проанализированы разработанные системы по заданной тематике и выявлены недостатки;

− изучена литература и обоснован выбор программных средств;

− разработана модель «КАК ЕСТЬ»;

− построена модель «КАК ДОЛЖНО БЫТЬ»;

− спроектированы даталогическая и инфологическая модели;

− разработан интерфейс системы;

− реализована автоматизированная система обработки данных по результатам учебного тестирования студентов;

− выполнено тестирование системы на работоспособность;

− рассчитана эффективность системы.

Поставленная цель и задачи были выполнены в полном объеме.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ


1) Аквино, К. Front-end [Текст] / К. Аквино, Т. Ганди. – Санкт-Петербург: Питер, 2017. – 226с.

2) Бабаев, А.Б. Создание сайтов [Текст] / А.Б. Бабаев, М. М. Боде. – Санкт-Петербург: Питер, 2014. – 123с.

3) Байрон, А.Б. Создание и управление сайтом [Текст] / А.Б. Байрон. – Санкт-Петербург: Символ – Плюс, 2016. – 576 с.

4) Белгородский государственный национальный исследовательский университет [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/ (дата обращения 10.11.2017).

5) Белгородский государственный национальный исследовательский университет [Электронный ресурс] – Режим доступа: https://www.bsu.edu.ru/bsu/ (дата обращения 10.11.2017).

6) Беклешов, В. К. Технико-экономическое обоснование проектов [Текст] / В.К. Беклешов. – Москва: Высшая школа, 2015. – 217с.

7) Браун, Э. Изучаем JavaScript. Руководство по созданию современных веб-сайтов [Текст] / Э.Браун. – Москва: Альфа-книга, 2017. – 319с.

8) Варфел, Т. Прототипирование. Практическое руководство [Текст] / Т. Варфел. – Москва: Манн, Иванов и Фербер, 2013. – 104с.

9) Веру, Л. Секреты CSS. Идеальные решения ежедневных задач [Текст] / Л. Веру. – Санкт-Петербург: Питер, 2016. – 225с.

10) Гасумова, С.Е. Информационные технологии в социальной сфере [Текст] / С.Е. Гасумова. – Москва: Дашков и К, 2014. – 246с.

11) Глушаков, С.В. Базы данных [Текст] / С.В. Глушаков, Д.В. Ломотько. – Харьков: Фолио, 2013. – 221с.

12) Даккет, Д HTML и CSS. Разработка и дизайн веб-сайтов [Текст] / Д. Дакетт. – Москва: Эксмо, 2017. – 430с.

13) Евдокимов, А. Создание сайтов своими руками на Bootstrap [Текст] / А.Евдофимов, М. Финков. – Санкт-Петербург: Наука и техника 2017. – 150с.

14) Качанов, А.С. Букварь по PHP и MySQL. Введение в PHP и MySQL [Текст] / А.С. Качанов, В. Д. Ткаченко, А. М. Головин. – Москва: Версия, 2016. – 13с.

15) Компьютерное тестирование [Электронный ресурс] – Режим доступа https://vuzlit.ru/559348/kompyuternoe_testirovanie (дата обращения 10.12.2017).

16) Маклаков, С.В. CASE-средства разработки информационных систем. BPwin и Erwin [Текст] / С.В. Маклаков. – Москва: ДиалогМифи, 2015. – 154с.

17) Макфарланд, Д.С. Большая книга CSS3 [Текст] / Д.С. Макфарланд. – Санкт-Петербург: Питер, 2014. – 156с.

18) Мартишин, С.А. Базы данных. Практическое применение СУБД SQL и NoSOL-типа для применения проектирования информационных систем [Текст] / СА. Мартишин, В.Л. Симонов, М.В. Храпченко. – Москва: Форум, 2017. – 80с.

19) Методология IDEF0 [Электронный ресурс] – Режим доступа: https://studfiles.net/preview/5535358/ (дата обращения 16.12.2017).

20) Методология IDEF0 [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/IDEF0 (дата обращения 16.12.2017).

21) Орлов, А.Т. PHP: Полезные приемы [Текст] / А.Т. Орлов. - Москва: Горячая Линия - Телеком, 2014. – 130с.

22) Рогозов, Ю.И. Моделирование систем [Текст] / Ю. И. Рогозов, Л.Н. Стукотий, А.С. Свиридов. – Москва: ТРТУ, 2016. – 140 с.

23) Скотт, В. Э. Рефакторинг баз данных. Эволюционное проектирование [Текст] / В.Э. Скотт, П. Дж. Садаладж. – Москва: И.Д. Вильямс, 2015. – 600с.

24) Смит, Б. Создание Web-страниц для "чайников" [Текст] / Б. Смит – Москва: Диалектика, 2014. – 398с.

25) Фленов, М.С. PHP глазами хакера [Текст] / М.С. Флеменов. – Санкт-Петербург: БХВ-Петербург, 2016. – 213с.

26) Функциональное моделирование [Электронный ресурс] – Режим доступа: http://media.devnet.ru/files/Lectures/ (дата обращения 16.12.2017).

27) Харрис, Э. PHP/MySQL для начинающих [Текст] / Э. Харрис. – Санкт-Петербург: КУДИЦ-Образ, 2015. – 384 с.

28) Цеховой, В.А. Web-дизайн и коммерция [Текст] / В.А. Цеховой. – Москва: Наука и техника, 2016. – 192с.

29) Шакин, В.Н. Теоретические основы построения БД [Текст] / В.Н Шакин, Г.К. Сосновиков, И.Б. Юскова. – Москва: МТУСИ, 2005. – 167с.

30) Цуканова, О.А. Методология и инструментарий моделирования бизнес – процессов [Текст] / О. А. Цуканова. – Санкт-Петербург: ИТМО, 2015. – 108с.

31) Вин, Ч. Как спроектировать современный сайт [Текст] / Ч. Вин. – Санкт-Петербург: Питер, 2016. – 214с.

32) IDEF0 [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/ (дата обращения 16.12.2017).

# ПРИЛОЖЕНИЯ

# ПРИЛОЖЕНИЕ А

## Авторизация пользователя

```
<?php include ROOT . '/layouts/header.php'; ?>
<section>
    <div class="container">
        <div class="row">
            <div class="col-sm-4 col-sm-offset-4 padding-right">
                <?php if (isset($errors) && is_array($errors)): ?>
                    <ul>
                        <?php foreach ($errors as $error): ?>
                            <li> -- <?php echo $error; ?></li>
                        <?php endforeach; ?>
                    </ul>
                <?php endif; ?>
                <div class="signup-form">
                    <h2>Вход на сайт</h2>
                    <form action="#" method="post">
                        <input type="login" name="login" placeholder="Логин" value=""/>
                        <input type="password" name="password" placeholder="Пароль" value=""/>
                        <input type="submit" name="submit" class="btn btn-default" value="submit" />
                    </form>
                </div>
                <br/>
                <br/>
            </div>
        </div>
    </div>
</section>
<?php include ROOT . '/layouts/footer.php'; ?>
<?php
class User
{
    public static function checkUserData($login, $pass)
    {
        $db = Db::getConnection();
        $sql = 'SELECT * FROM user WHERE login = :login AND pass = :pass';
        $result = $db->prepare($sql);
        $result->bindParam(':login', $login, PDO::PARAM_INT);
        $result->bindParam(':pass', $pass, PDO::PARAM_INT);
        $result->execute();
        $user = $result->fetch();
        if ($user) {
            return $user['id'];
        }
        return false;
    }
      public static function auth($userId)
    {
```

```
        $_SESSION['user'] = $userId;
    }
    public static function checkLogged()
    {

        if (isset($_SESSION['user'])) {
            return $_SESSION['user'];
        }
        header("Location: /user/login");
    }
     public static function isGuest()
    {
        if (isset($_SESSION['user'])) {
            return false;
        }
        return true;
    }
    public static function checkPassword($password)
    {
        if (strlen($password) >= 0) {
            return true;
        }
        return false;
    }
    public static function checkEmail($login)
    {
        if (strlen($login)>=0) {
            return true;
        }
        return false;
    }
        public static function getUserById($id)
    {
        // Соединение с БД
        $db = Db::getConnection();

        $sql = 'SELECT * FROM user WHERE id = :id';

        $result = $db->prepare($sql);
        $result->bindParam(':id', $id, PDO::PARAM_INT);

        $result->setFetchMode(PDO::FETCH_ASSOC);
        $result->execute();
        return $result->fetch();
    }
        public static function getGroup($id)
    {

        $db = Db::getConnection();
        try {
            $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

```
        $sql = 'SELECT grup.number as number FROM grup, student_group where
student_group.id_stud=:id and grup.id=student_group.id_group';
        $result = $db->prepare($sql);
        $result->bindParam(':id', $id, PDO::PARAM_INT);
        $result->setFetchMode(PDO::FETCH_ASSOC);
        $result->execute();
    }
   catch(PDOException $e)
        {
         echo $sql . "<br>" . $e->getMessage();
        }
        return $result->fetch();
           }

}
<?php
/**
*
*/
include_once ROOT. '/models/User.php';
class SiteController
{
          public function actionIndex()
          {
          $login = false;
        $password = false;
              if (isset($_POST['submit'])) {
               $login = $_POST['login'];
          $password = $_POST['password'];
          $errors = false;
            if (!User::checkEmail($login)) {
              $errors[] = 'Неправильный email';
          }
                $userId = User::checkUserData($login, $password);
          if ($userId == false) {
                        $errors[] = 'Неправильные данные для входа на сайт';
          } else {
                        User::auth($userId);
                        header("Location: /allcourses");
          }
        }
        require_once(ROOT . '/views/user/login.php');
        return true;
           }
```

# ПРИЛОЖЕНИЕ Б

## Тестирование студентов

```
<?php
include_once  ROOT. '/models/Course.php';
include_once  ROOT. '/models/User.php';
include_once  ROOT. '/models/Test.php';

class TestController
{
        public  function  actionView($course_id,  $theme_id)
        {
                $id_user=$_SESSION["user"];
                $hname="Просмотр  темы";
                $themeList=array();
                $themeScore=Test::getThemeScore($theme_id,  $id_user);
                $user=User::getUserById($id_user);

                require_once(ROOT  . '/views/test/view.php');
        }

        public  function  actionRun($theme_id)
        {
                $id_user=$_SESSION["user"];
                $hname="Тестирование";
                $user=User::getUserById($id_user);
                $question=Test::getQuestionlist($theme_id);
                $num=count($question);

                if (isset($_POST['Submit']))  {
                        $sum=Test::checkAnswer($_POST,$id_user,  $num);
                        $timedate=date("Y-m-d  H:i:s");
                        Test::updateScore($sum,  $theme_id,  $id_user);
        }
                                require_once(ROOT  . '/views/test/testing.php');
        }
        public  function  actionRunAll($discipline_id)
        {
                $id_user=$_SESSION["user"];
                $hname="Итоговое  тестирование";
                $user=User::getUserById($id_user);
                $question=Test::getAllQuestionlist($discipline_id);
                $num=count($question);

                if (isset($_POST['Submit']))  {
                        $sum=Test::checkAnswer($_POST,$id_user,  $num);
                        $timedate=date("Y-m-d  H:i:s");
                        Test::updateAllScore($sum,  $discipline_id,  $id_user);
```

```
                        header('Location: /course-'.$discipline_id);
                }
                        require_once(ROOT . '/views/test/testing.php'); }
        public function actionWarning($question_id)
        {
                $db = Db::getConnection();
                $result = $db->prepare('INSERT INTO errors SET question_id=:question_id');
                $result->bindParam(':question_id', $question_id, PDO::PARAM_INT);;
                $result->execute();
        header('Location: /allcourses');
        }
}
<?php
class Test
{
        public static function getThemeScore ($theme_id, $id_stud)
        {
                $db = Db::getConnection();
                $themeScore=array();
                $result= $db->prepare('SELECT * FROM score WHERE id_stud=:id_stud
AND theme_id=:theme_id');
                $result->bindParam(':id_stud', $id_stud, PDO::PARAM_INT);
                $result->bindParam(':theme_id', $theme_id, PDO::PARAM_INT);
                $result->execute();
                $i=0;
                while ($row=$result->fetch()) {
                        $themeScore[$i]['number_score']=$row['number_score'];
                        $themeScore[$i]['date']=$row['date'];
                        $i++; }
                                return $themeScore;
        }
                public static function getQuestionlist ($id_theme)
        {
                $db = Db::getConnection();
                $questionList=array();
                $result = $db->prepare('SELECT * FROM question where
id_theme=:id_theme');
                $result->bindParam(':id_theme', $id_theme, PDO::PARAM_INT);
                $result->execute();
                $i=0;
                while ($row=$result->fetch()) {
                        $questionList[$i]['id']=$row['id'];
                        $questionList[$i]['text_quest']=$row['text_quest'];
                        $questionList[$i]['id_type']=$row['id_type'];
                        $i++;
                }return $questionList;
                } public static function getQuestionType ($id_type)
                {
                        $typeInput="";
                switch ($id_type) {
                        case '1':
```

```
            {$typeInput="radio";}
            break;
            case '2':
            {$typeInput="checkbox";}
            break;
            case '3':
            {$typeInput="";}
            break;
            default:
            echo "ERROR";
            break;
}return $typeInput; }
public static function getAnswer ($id_question)
{       $db = Db::getConnection();
        $answer=array();
        $answer[0]['N']=0;
        $result = $db->prepare('SELECT * FROM answer where id_question=:id');
        $result->bindParam(':id', $id_question, PDO::PARAM_INT);
        $result->execute();
        $i=1;
        while ($row=$result->fetch()) {
                $text="";
                $answer[$i]['id']=$row['id'];
                $answer[$i]['text_answer']=$row['text_answer'];
                if ($row['flag']==1) {
                        $text="checked";
                }
                $answer[$i]['flag']=$text;
                $i++;
                $answer[0]['N']++;
        }
        return $answer;
}
        public static function checkAnswer ($answer, $user_id, $num_quest)
{       $sum=0;
        $perAnsw=100/$num_quest;
        $answerArr=array();
        $db = Db::getConnection();
        $result = $db->prepare('SELECT * FROM answer');
        $result->execute();
        $i=0;
        while ($row=$result->fetch()) {
                $answerArr[$i]['id']=$row['id'];
                $answerArr[$i]['id_question']=$row['id_question'];
                $answerArr[$i]['flag']=$row['flag'];
                $answerArr[$i]['text_answer']=$row['text_answer'];
                $i++;
        }
        $id_question='';
        foreach ($answer['ans'] as $ansItem) {
                foreach ($answerArr as $arrayItem) {
                        if ($arrayItem['id']==$ansItem) {
```

```
                                        $id_question=$arrayItem['id_question'];
                                        }}
                        $result=$db->prepare('SELECT count(id) FROM answer WHERE
id_question=:id_question AND flag=1');
                        $result->bindParam(':id_question', $id_question,
PDO::PARAM_INT);
                        $result->execute();
                        $countAnsTrue=$result->fetch(); //количество правильных ответов
                        $trueScore=$perAnsw/$countAnsTrue['count(id)'];
                        foreach ($answerArr as $arrayItem ) {
                                if (($arrayItem['id']==$ansItem)and($arrayItem['flag']==1)) {
                                        $sum+=$trueScore; }}}
                if (isset($answer['text'])) {

                $id_text=array_keys($answer['text']);

                $i=0;
                        foreach ($answer['text'] as $asyxy) {
                                // echo $asyxy;
                        foreach ($answerArr as $arrayItem ) {
                                if
(($arrayItem['id_question']==$id_text[$i])and($arrayItem['text_answer']==$asyxy)and($arrayItem['
flag']==1)) {
                                        $sum+=$perAnsw; }}}}
                                return $sum;
        }
        public static function updateScore ($sum,$theme_id, $user_id)
{
        $db = Db::getConnection();
        $result = $db->prepare('INSERT INTO score SET id_stud=:id_stud,
theme_id=:theme_id, number_score=:number_score, discipline_id=NULL');
        $result->bindParam(':id_stud', $user_id, PDO::PARAM_STR);
        $result->bindParam(':theme_id', $theme_id, PDO::PARAM_INT);
        $result->bindParam(':number_score', $sum, PDO::PARAM_INT);
        $result->execute();
}
public static function getAllQuestionlist ($discipline_id)
{
                $db = Db::getConnection();
                $questionList=array();
                $result = $db->prepare('SELECT question.id as id, question.text_quest as
text_quest, question.id_type as id_type FROM question, theme where theme.id_disc=:discipline_id
and question.id_theme=theme.id');
                $result->bindParam(':discipline_id', $discipline_id, PDO::PARAM_INT);
                $result->execute();
                                $i=0;
                while ($row=$result->fetch()) {
                        $questionList[$i]['id']=$row['id'];
                        $questionList[$i]['text_quest']=$row['text_quest'];
                        $questionList[$i]['id_type']=$row['id_type'];
                        $i++;
                }
```

```
                return $questionList;
            }
public static function updateAllScore ($sum,$discipline_id, $user_id)
    {
            $db = Db::getConnection();
            $result = $db->prepare('INSERT INTO score SET id_stud=:id_stud,
discipline_id=:discipline_id, number_score=:number_score, theme_id=NULL');
            $result->bindParam(':id_stud', $user_id, PDO::PARAM_STR);
            $result->bindParam(':discipline_id', $discipline_id, PDO::PARAM_INT);
            $result->bindParam(':number_score', $sum, PDO::PARAM_INT);
            $result->execute();
    }
}<?php include ROOT . '/layouts/header.php'; ?>
<?php include ROOT . '/layouts/sidebar.php'; ?>
<div class="content">
    <div class="course-list">
            <h2><?php echo $hname; ?></h2>

             <div>
  <form action="#" method="post">
   <?php $k=1; $i=1; foreach ($question as $questionItem): ?>
   <label><?php echo $i;?>. <?php echo $questionItem["text_quest"];?></label><a
href="/warning<?php echo $questionItem["id"];?>"><img src="/template/images/warning.svg"
width="15px" title="Сообщить об ошибке"> </a>
    <br>
   <?php
   $answer=Test::GetAnswer($questionItem["id"]);
   if ($questionItem["id_type"]==3): ?> <input type="text" name="text[<?php echo
$questionItem["id"]; ?>]" value=""><br>
   <?php endif; $typeInput=Test::getQuestionType($questionItem["id_type"]);
   $j=1;
   $N=$answer[0]['N'];
   while ($answer[0]['N']) { ?>
       <?php if ($typeInput!=NULL){ ?>
    <p><?php echo $answer[$j]['text_answer'];?><input type="<?php echo $typeInput;?>"
name="ans[<?php echo $k; ?>]" value="<?php echo $answer[$j]['id'];?>"/></p>
   <?php }
    $j++;$k++; $answer[0]['N']--;} ?><br>
    <?php $i++; endforeach;?>
    <input type="submit" name="Submit" value="Отправить" />
   </form>
  </div>
    </div>
</div>
<?php include ROOT . '/layouts/footer.php'; ?>
```

# ПРИЛОЖЕНИЕ В

## Вывод успеваемости студентов

```
<?php
class Course
{
	public static function getCourselist ()
	{
			$db = Db::getConnection();
			$courseList=array();
			$result=$db->query('SELECT * FROM discipline');
			$i=0;
			while ($row=$result->fetch()) {
				$courseList[$i]['id']=$row['id'];
				$courseList[$i]['name']=$row['name'];
				$i++;
			}
			return $courseList;
		}
		public static function getFavouriteslist ()
	{
			$db = Db::getConnection();
			$id_user=$_SESSION["user"];
			$faveList=array();
			$result=$db->prepare('SELECT * FROM favourites, discipline
WHERE favourites.id_user=:id_user AND discipline.id=favourites.discipline_id');
			$result->bindParam(':id_user', $id_user, PDO::PARAM_INT);
			$result->execute();
			$i=0;
			while ($row=$result->fetch()) {
				$faveList[$i]['id']=$row['id'];
				$faveList[$i]['name']=$row['name'];
				$i++;
			}
			if ($faveList) {}
			return $faveList;
		}
	public static function getThemeList ($id)
	{
			$db = Db::getConnection();
			$themeList=array();
			$result=$db->query('SELECT * FROM theme where id_disc='.$id);
			$i=0;
			while ($row=$result->fetch()) {
				$themeList[$i]['id']=$row['id'];
				$themeList[$i]['name']=$row['name'];
				$i++;
			}
```

```
                return $themeList;
        }
    public static function getMyCourselist ($id_user)
    {
                $db = Db::getConnection();
                $courseList=array();
                $result = $db->prepare('SELECT discipline.id as id, discipline.name
as name FROM discipline, user_discip where user_discip.user_id=:id_user AND
discipline.id=user_discip.discip_id');
                $result->bindParam(':id_user', $id_user, PDO::PARAM_INT);
                $result->execute();
                $i=0;
                while ($row=$result->fetch()) {
                        $courseList[$i]['id']=$row['id'];
                        $courseList[$i]['name']=$row['name'];
                        $i++;
                }
                        return $courseList;
        }
        public static function getMaxCourses()
        {
                $db = Db::getConnection();
                $themeList=array();
                $result=$db->query('SELECT max(id) FROM discipline');
                $count=$result->fetch();
                return $count[0];
        }
        public static function isFave($id_user, $discipline_id)
        {
                $db = Db::getConnection();
                $isFave=array();
                $result = $db->prepare('SELECT * FROM favourites where
id_user=:id_user AND discipline_id=:discipline_id');
                $result->bindParam(':id_user', $id_user, PDO::PARAM_INT);
                $result->bindParam(':discipline_id', $discipline_id,
PDO::PARAM_INT);
                $result->execute();
                $isFave=$result->fetch();
                return $isFave;
        }
        public static function delFave ($course_id,$id_user)
        {
                $db = Db::getConnection();
                $result = $db->prepare('DELETE FROM favourites WHERE
discipline_id=:course_id AND id_user=:id_user');
                $result->bindParam(':course_id', $course_id, PDO::PARAM_STR);
                $result->bindParam(':id_user', $id_user, PDO::PARAM_STR);
                $result->execute();
        }
        public static function addFave ($course_id,$id_user)
        {
        $db = Db::getConnection();
```

```
                $result = $db->prepare('INSERT INTO favourites SET
discipline_id=:course_id, id_user=:id_user');
                        $result->bindParam(':course_id', $course_id, PDO::PARAM_STR);
                        $result->bindParam(':id_user', $id_user, PDO::PARAM_STR);
                        $result->execute();
        }
                public static function getDisciplineScore ($discipline_id, $id_stud)
        {
                        $db = Db::getConnection(); // Подключение к БД
                        $disciplineScore=array(); // Инициализация массива
                        $result= $db->prepare('SELECT * FROM score WHERE
id_stud=:id_stud AND discipline_id=:discipline_id');// Подготовка запроса
                        $result->bindParam(':id_stud', $id_stud, PDO::PARAM_INT); //
Подстановка параметров
                        $result->bindParam(':discipline_id', $discipline_id,
PDO::PARAM_INT);
                        $result->execute(); // Выполнение запроса
                        $i=0;
                        while ($row=$result->fetch()) { // Присваивание элементов массиву
                                $disciplineScore[$i]['number_score']=$row['number_score'];
                                $disciplineScore[$i]['date']=$row['date'];
                                $i++;
                        }
                        return $disciplineScore;
        }
                public static function getGroups()
        {
                $grup=array();
                        $db = Db::getConnection();
                        try {
                        $db->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
                        $result= $db->prepare('SELECT * FROM grup');
                        // $result->bindParam(':theme_id', $theme_id, PDO::PARAM_INT);
                        $result->execute();
                        }
                catch(PDOException $e)
                {
                        echo $sql . "<br>" . $e->getMessage();
                }
                        $i=0;
                        while ($row=$result->fetch()) {
                                $grup[$i]['id']=$row['id'];
                                $grup[$i]['number']=$row['number'];
                                $i++;
                        }
                        return $grup;
        }
        public static function studentList($group_id)
        {
                        $stud=array();
                        $db = Db::getConnection();
```

```
                        try {
                        $db->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
                        $result= $db->prepare('SELECT user.id as id, user.name as name,
user.surname as surname FROM user, grup, student_group where
student_group.id_group=:id_group and student_group.id_stud=user.id GROUP BY user.id');
                        $result->bindParam(':id_group', $group_id, PDO::PARAM_INT);
                        $result->execute();
                        }
                catch(PDOException $e)
                {
                        echo $sql . "<br>" . $e->getMessage();
                }
                        $i=0;
                        while ($row=$result->fetch()) {
                                $stud[$i]['id']=$row['id'];
                                $stud[$i]['name']=$row['name'];
                                $stud[$i]['surname']=$row['surname'];
                                $i++;
                        }
                                return $stud;
        }
        public static function themeStats($user_id, $theme_id)
        {
                        $themeScore=array();
                        $db = Db::getConnection();
                        try {
                        $db->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
                                $result= $db->prepare('SELECT max(number_score) as
number_score, date, id_stud FROM score where id_stud=:id_stud and theme_id=:theme_id group
by theme_id');
                        $result->bindParam(':id_stud', $user_id, PDO::PARAM_INT);
                        $result->bindParam(':theme_id', $theme_id, PDO::PARAM_INT);
                        $result->execute();
                        }
                catch(PDOException $e)
                {
                        echo $sql . "<br>" . $e->getMessage();
                }
                        $i=0;
                        while ($row=$result->fetch()) {
                                $themeScore[$i]['number_score']=$row['number_score'];
                                $themeScore[$i]['date']=$row['date'];
                                $themeScore[$i]['id_stud']=$row['id_stud'];
                                $i++;
                        } return $themeScore;
        }
        public static function getMaxDisciplineScore ($discipline_id, $id_stud)
        {
                        $db = Db::getConnection();
                        $disciplineScore=array();
```

```
                        $result= $db->prepare('SELECT max(number_score) as
number_score, date FROM score WHERE id_stud=:id_stud AND discipline_id=:discipline_id
group by discipline_id');
                        $result->bindParam(':id_stud', $id_stud, PDO::PARAM_INT);
                        $result->bindParam(':discipline_id', $discipline_id,
PDO::PARAM_INT);
                        $result->execute();
                        $i=0;
                        while ($row=$result->fetch()) {
                                $disciplineScore[$i]['number_score']=$row['number_score'];
                                $disciplineScore[$i]['date']=$row['date'];
                                $i++;
                        }
                        return $disciplineScore; }

    }
    <?php
    include_once ROOT. '/models/Blog.php';
    include_once ROOT. '/models/Course.php';
    include_once ROOT. '/models/User.php';
    class CabinetController
    {
        public function actionIndex()
        {
                $id_user=$_SESSION["user"];
                $blogList=array();
                $blogList=Blog::getBlogList();
                $courseList=array();
                $courseList=Course::getCourseList();
                require_once(ROOT . '/views/cabinet/index.php');
                        }
        public function actionLogout()
      {
        session_start();
        unset($_SESSION["user"]);
        require_once(ROOT . '/views/user/login.php');
      }
        public function actionCourselist()
        {
                $id_user=$_SESSION["user"];
                $hname="Все курсы";
                $courseList=array();
                $courseList=Course::getCourseList();
                User::getGroup($id_user);
                require_once(ROOT . '/views/cabinet/courselist.php');
                        }
        public function actionAddFave($course_id)
        {
                $id_user=$_SESSION["user"];
                Course::addFave($course_id,$id_user);
                header('Location: /allcourses');;
```

```
        }
        public function actionDelFave($course_id)
        {
                $id_user=$_SESSION["user"];
                Course::delFave($course_id,$id_user);
                header('Location: /allcourses');
                        }
        public function actionStats()
        {
                $hname="Успеваемость";
                $themeStats=array();
                $id_user=$_SESSION["user"];
                $user=User::getUserById($id_user);
                        if ($user["type_id"]==2):
                                $hname="Оценки студентов";
                                $courseList=Course::getCourseList();
                                if (isset($_POST['submit'])) {

                        $chosenStats=Course::getStatsList($theme_id);
                        echo "<pre>";
                        print_r($chosenStats);echo "</pre>";
                        header('Location: /stats');
                }
                endif;
                $disciplineStats=Cabinet::disciplineStats($id_user);
                require_once(ROOT . '/views/cabinet/stats.php');
                        }
        public function actionCoursestats($id_discipline)
        {
                $id_user=$_SESSION["user"];
                $user=User::getUserById($id_user);
                $themeList=array();
                $hname="Оценки студентов";
                $grup=Course::getGroups();
                $disciplineStats=array();
                if (isset($_POST['Submit'])) {
                        $stud_id=($_POST['student']);
                        $themeList=Course::getThemeList($id_discipline);
                $disciplineStats=Course::getMaxDisciplineScore($id_discipline,$stud_id);
                }
                require_once(ROOT . '/views/cabinet/coursestats.php');
                        }
}
<?php include ROOT . '/layouts/header.php'; ?>
<?php include ROOT . '/layouts/sidebar.php'; ?>
<div class="content">
<?php if ($user["type_id"]==1): ?>
                <h2><?php echo $hname; ?></h2>
<canvas id="myChart" width="100" height="30"></canvas>
<script>
var ctx = document.getElementById("myChart").getContext('2d');
var myChart = new Chart(ctx, {
```

```
            type: 'horizontalBar',
            data: {
               labels: [
               <?php foreach ($disciplineStats as $disciplineItem): ?>
                "<?php echo Cabinet::getDisciplineName($disciplineItem['discipline_id'])?>", <?php
endforeach;?>],
               datasets: [{
                  label: 'Успеваемость по дисциплине',
                  barPercentage: 10,
                  data: [<?php foreach ($disciplineStats as $disciplineItem): ?>
                "<?php echo $disciplineItem['number_score']?>", <?php endforeach;?>],
                  backgroundColor: [
                     'rgba(255, 99, 132, 0.2)',
                     'rgba(54, 162, 235, 0.2)',
                     'rgba(255, 206, 86, 0.2)'
                  ],
                  borderColor: [
                     'rgba(255,99,132,1)',
                     'rgba(54, 162, 235, 1)',
                     'rgba(255, 206, 86, 1)'
                  ],
                  borderWidth: 1
               }]
            },
            options: {
               scales: {
                  xAxes: [{
                     ticks: {
                        min: 0,
                        max: 100
                     }
                  }]
            } }
         });
         </script>
         <?php endif; ?>
         <?php if ($user["type_id"]==2): ?>
         <h2><?php echo $hname; ?></h2>
         <?php foreach ($courseList as $courseItem): ?>
            <li><a href="/score/<?php echo $courseItem['id']; ?>"><?php echo
$courseItem['name']; ?></a></span>
               <?php if(Course::isFave($id_user,$courseItem['id'])) {?>
                  <a href="/delFave/course/<?php echo $courseItem['id']; ?>"><img
src="/template/images/active.svg" width="30px" title="Удалить из избранного"> </a></li> <?php
} else {?> <a href="/addFave/course/<?php echo $courseItem['id']; ?>"><img
src="/template/images/unactive.svg" width="30px" title="Добавить в избранное"> </a></li>
<?php }?>
               <?php endforeach; ?>
            <?php endif; ?>
         </div>
<?php include ROOT . '/layouts/footer.php'; ?>
```