МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра Компьютерных технологий

«Допущен к защите»
Заведующий кафедрой _____

_____
(Ф.И.О., ученая степень, звание)

_____ «_____» _____ **20__ г.**
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка интерактивной обучающей системы для курса „Проектирование баз данных"

Специальность 5В070400 - Вычислительная техника и программное обеспечение

Выполнил (а) Лепостин В.А. ВТ-10-3
(Фамилия и инициалы) группа

Научный руководитель Сатимова Е.Т., к.т.н.
(Фамилия и инициалы, ученая степень, звание)

Консультанты:

по экономической части:
Еркешева З.Ф., ст. преподаватель
(Фамилия и инициалы, ученая степень, звание)
Еркешева «14» 05 2014 г.
(подпись)

по безопасности жизнедеятельности:
Приходько Н.Г. д.х.н., профессор
(Фамилия и инициалы, ученая степень, звание)
_____ «14» 05 2014 г.
(подпись)

по применению вычислительной техники:
Сатимова Е.Т. доцент, к.т.н.
(Фамилия и инициалы, ученая степень, звание)
_____ «_____» _____ 20__ г.
(подпись)

Нормоконтролер: Гусупов Д.М.
(Фамилия и инициалы, ученая степень, звание)
_____ «21» мая 2014 г.
(подпись)

Рецензент: _____
(Фамилия и инициалы, ученая степень, звание)
_____ «_____» _____ 20__ г.
(подпись)

Алматы 2014 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет Информационных технологий
Специальность 5В070400 – Вычислительная техника и программное обеспечение
Кафедра Компьютерных технологий

ЗАДАНИЕ
на выполнение дипломного проекта

Студент Легостин Вячеслав Александрович
(фамилия, имя, отчество)

Тема проекта Разработка интерактивной обучающей системы для курса „Проектирование баз данных"

утверждена приказом ректора № 115 от «24» сентября 2013 г.
Срок сдачи законченной работы «___» _____ 20___ г.
Исходные данные к проекту требуемые параметры результатов проектирования (исследования) и исходные данные объекта

Методические материалы к курсу „Проектирование баз данных" должны быть представлены в виде интерактивного приложения. Система должна отслеживать успеваемость студентов, все их ошибки и действия. В разработке использовать современные фреймворки и приложения

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:

1 Исследование существующих систем для интерактивного образования
2. Изучение фреймворка YII, языков PHP и JS
3 Разработка схем создания проектов
4 Создание финальной версии системы

Перечень графического материала (с точным указанием обязательных чертежей)

1. Схема работы сервера Apache
2. Структура паттерна MVC
3. Диаграмма типа сущность-связь для системы

Рекомендуемая основная литература

1. Адам Ф. jQuery для профессионалов
2. Когеров Д. PHP в подлиннике
3. Ганник Дж. MSSQL 2008. Карманный справочник
4. Роб П. Системы баз данных

Консультанты по проекту с указанием относящихся к ним разделов

Раздел	Консультант	Сроки	Подпись
БЖД	Тихонова М.Г.	11.04 - 16.05.14	
Экономика	Брежнева З.Ю.	13.05 - 14.05.14	Брежнева
Нормоконтроль	Пупков Д.М.	21.05.14	

Г Р А Ф И К
подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
	Подбор литературы	16 марта	
	Исследование предметной области	4 апреля	
	Разработка базы данных	8 апреля	
	Разработка алгоритмов системы	3 мая	
	Отладка системы	6 мая	
	Технико-экономическое обоснование	12 мая	
	Безопасность жизнедеятельности	16 мая	

Дата выдачи задания «24» февраля 2014 г.

Заведующий кафедрой _____ _____
(подпись) (Фамилия и инициалы)

Руководитель _____ Сабинкова Е.Г.
(подпись) (Фамилия и инициалы)

Задание принял к исполнению
студент _____ Лепостин В.А.
(подпись) (Фамилия и инициалы)

Аннотация

В данном дипломном проекте проводится разработка web-интерфейса для интерактивной обучающей системы по курсу «Проектирование баз данных».

Рассмотрены существующие интерактивные обучающие системы, использующиеся в сфере высшего образования. Разработан web-интерфейс для интерактивной обучающей системы для курса «Проектирование баз данных» с целью повышения продуктивности изучения курса студентами.

Web-интерфейс разработан с использованием новейших web-технологий и отвечает всем современным требованиям к программному обеспечению.

Кроме этого в двух последних главах рассматриваются вопросы безопасности жизнедеятельности, приводится технико-экономическое обоснование и рассчитывается цена разработки проекта.

Аңдатпа

Осы дипломдық жобада «Деректер базасын жобалау» курсы бойынша интерактивті оқыту жүйесі үшін web-интерфейс әзірлемесі жүргізілді.

Жоғары білім беру саласында пайдаланылатын қолданыстағы интерактивті оқыту жүйелері қарастырылды. Студенттердің курсты игеру өнімділігін арттыру мақсатында «Деректер базасын жобалау» курсы үшін интерактивті оқыту жүйесін web-интерфейсі жасалды.

Web-интерфейсі жаңа web-технологияларын қолдана отырып жасалды және бағдарламалық қамтамасыз етуде қазіргі барлық талаптарға жауап береді.

Бұдан басқа соңғы екі тарауда өміртіршілік қауіпсіздігі мәселелері қарастырылды және жобаны әзірлеудің техникалық-экономикалық негіздемесі мен бағасы есептелді.

Abstract

In this project I conducted to develop web-based interface for interactive education system for "Database Design" course.

Another existing education systems for universities was explored. Developed web-based interface for an interactive learning system for the course " Database Design " in order to increase the productivity of students studying.

Web- interface is designed with the latest web technologies and with all modern requirements of web.

In addition, in the last two chapters deal with issues of life safety , provides a feasibility study and calculate the price of the project development .

Содержание

Введение

Интерактивная обучающая система – это такая система, которая специально создается для обучения пользователя чему-либо, при этом пользователь не просто читает нужную ему информацию, но и непосредственно взаимодействует с программой – выполняет различные задания. Такой обмен информацией между программой и пользователем называется интерактивностью. Таким образом, интерактивность - это способность компьютерной системы адекватно отвечать на действия пользователя. Вследствие этого, система должна обладать неким примитивным подобием интеллекта. Интерактивное взаимодействие относится к одному из научных направлений человеко-компьютерного взаимодействия (HCI). Поэтому, в данном проекте стояла задача изучить и разработать различные взаимодействия между людьми и компьютерами.

HCI опирается в равной степени на человеческий фактор и на компьютерный фактор. Понять компьютерную сторону можно с помощью различных технологий программирования, компьютерной графики, а человеческую – с помощью психологии, социологии и некоторых разделов теории коммуникации.

В интерактивных обучающих системах лучше всего использовать стандартные пути и решения, так как целью таких систем является не введение новшеств, чтобы удивить пользователя, а дать ему наиболее полное представление о материале, который он желает изучить. Психологи доказали, что в таких случаях человек интуитивно доверяется системе с более привычными механизмами навигации и с более простым и понятным интерфейсом, так как ничто не будет отвлекать его от обучения — всё расположено стандартно, удобно и ненавязчиво [1].

Программа является системой, если компоненты в ней взаимосвязаны друг с другом, при этом каждый элемент выполняет свои функции. Поэтому любая обучающая программа является системой. Если проверка правильного ответа выполняется исключительно программными средствами, то данная система будет автоматизированной обучающей системой. Разрабатываемая в данной дипломной работе программа не является в полной мере автоматизированной, так как проверка заданий осуществляется преподавателем.

Данная система относится к типу разомкнутых обучающих систем, так как в ней выполняется заранее определенная разработчиком последовательность изложения разделов и задач.

Тема данного дипломного проекта – «Разработка интерактивной обучающей системы для курса «Проектирование баз данных»». Данный программный продукт разрабатывается специально для предмета «Проектирование баз данных», который изучается в АУЭС.

Данный курс изучается на 3 курсе студентами, учащимися в АУЭС по специальности ВТиПО и ИС. Студенты знакомятся с основными понятиями теории реляционных баз данных, учатся проектировать базы, используя такие программы как Erwin или Visio для создания диаграмм сущностей и связей. С помощью программы SQL Server Management Studio в курсе предусмотрено создание студентом собственной БД в качестве курсового проекта, а также выполнение определенного количества лабораторных работ. В процессе обучения, однако, студентам часто приходится терять много времени в ожидании своей очереди на проверку преподавателем того или иного задания, что сильно утомляет как студента, так и преподавателя, и тормозит прогресс в изучении материала из-за нехватки времени, которое тот мог бы провести с большей пользой. Таким образом, создание интерактивной обучающей системы значительно ускорит процесс сдачи и проверки работ, так как решение задачи можно будет прислать преподавателю по интернету, и так же онлайн в режиме реального времени преподаватель сможет быстро проверить задание и выставить оценку или отправить на доработку. Несомненно, такая система упростит студенту изучение материала, а преподавателю – мониторинг знаний учащихся, а также повысит заинтересованность студентов в изучении курса, заменив скучные однообразные лекции и лабораторные на интересную модель обучения в удобной программе.

Целью создания данной программы является:

- Создать систему для облегчения прохождения данного курса студентами.
- Обеспечить учителей средствами проверки выполнения студентами заданий и их уровня освоения материала.
- Система должна быть динамической и легко расширяемой.

Структурно дипломная работа состоит из введения, четырех разделов и заключения.

В первом разделе определенны требования, которым должен удовлетворять программный продукт. Разработана модель базы данных, рассмотрены технологии программирования, используемые в работе.

Во втором разделе приведено описание работы веб-сайта.

В третьем разделе приведено технико-экономическое обоснование реализации проекта и вычислена его себестоимость.

В четвертом разделе проведен анализ потенциально опасных и вредных производственных факторов проектируемого объекта, воздействующих на персонал, расчет кондиционирования офиса и пожарной безопасности проектируемого объекта.

1 Теоретическая часть

1.1 Постановка задачи

Требуется разработать интерактивную обучающую систему для курса «Проектирование баз данных».

Интерактивная система позволит студентам изучать данный курс дистанционно в удобной форме в виде некоторого количества модулей(глав), после которых предлагаются задачи и тесты, которые студент, прошедший изучение, должен выполнить. Решенные задания отправляются на проверку преподавателю, который может либо поставить студенту отметку о выполнении данного задания, либо отправить обратно на доработку. Встроенный мессенджер позволит студенту обращаться за помощью к учителю в режиме реального времени. Пользователь заполняет свой профиль, в котором будет указано сколько заданий он решил правильно и сколько неправильно. Также будет вестись рейтинг каждого учащегося, что придаст обучению соревновательный характер, тем самым повышая успеваемость и заинтересованность курсом.

Таким образом, пользователь-студент после регистрации на сайте может получить доступ к теоретическому материалу курса, выполнять задачи и тесты, отправлять результат на проверку, общаться с помощью мессенджера с преподавателем и другими пользователями.

Преподаватель может создавать уведомления, которые в последствии можно рассылать всем студентам (например, уведомление о консультации к экзамену), проверять задания студентов, ставить им оценки.

1.2 Этап начальной разработки БД

Анализ предметной области

Курс «Проектирование баз данных» – это предмет, изучающийся в АУЭС, на котором проходятся основы проектирования и реализации баз данных.

Базовой СУБД для изучения данного курса является SQL SERVER 2008 R2 от компании Microsoft.

Студенты должны изучать теоритические материалы, после каждой главы на лабораторных занятиях им предлагается выполнить несколько практических заданий для более глубокого освоения материала.

Сами задания выполняются либо в программах для проектирования ER-диаграмм, например ERWIN или Microsoft Visio, а также в среде SQL Server Management Studio. После выполнения задания студенты должны показать решение преподавателю.

Также необходимо проводить тестирование студентов на предмет освоения ими теоритических материалов.

Данный дипломный проект упростит процесс освоения студентами курса, за счет уменьшения времени, затраченного на очереди к преподавателям.

Вся информация о прогрессе и о курсе будет храниться в БД.

Базовые сущности данной базы данных:

1 Пользователи. Эта сущность служит для хранения информации о пользователях системы, об их личных данных и уровне доступа.

2 Модули курса. В данной сущности хранится весь теоритический материал.

3 Задания. Здесь хранятся тексты заданий и их начальные условия.

4 Тесты. В данной сущности хранится информация о тестировании, о том к какому модулю оно относится и его описание.

5 Тестовые вопросы. Здесь хранится текст всех тестовых вопросов и то, к какому именно тесту он относится.

6 Варианты ответа. Здесь хранятся варианты ответа на тестовые вопросы.

7 Очередь проверки. В данную сущность будут записываться тикеты на проверку заданий, выполненных студентами.

8 Сообщения. Здесь в зашифрованном виде хранятся все пересылаемые студентами и учителями сообщения.

UML диаграммы

Для моделирования статических объектов в объектно-ориентированной концепции UML существует 12 диаграмм, но так как большинство объектов БД «Интерактивной обучающей системы» являются статическими, используем самые основные:

1 Диаграмма компонентов.

2 Диаграмма прецедентов.

3 Диаграмма классов.

4 Диаграмма развертывания.

Диаграмма компонентов

Диаграмма компонентов показывает набор компонентов и отношений между ними.

Компонент – это физически заменяемая часть системы, которая имеет набор интерфейсов и обеспечивает их реализацию.

Приложение обращается за получением или добавлением информации к базе данных mssql. Но обращается не напрямую, а с помощью ODCB драйвера MSSQL PHP Driver . Таким образом, MSSQL PHP Driver является интерфейсом между приложениеми базой данных, обеспечивая их взаимодействие. Диаграмма компонентов представлена на рисунке 1.1.

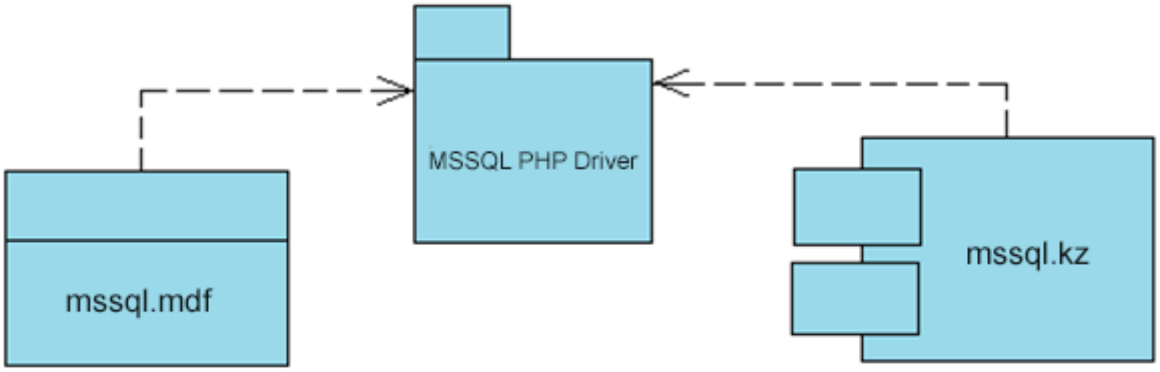


Рисунок 1.1 – Диаграмма компонентов

Диаграмма прецедентов

Для характеристики взаимодействия пользователей с базами данных применяется диаграмма прецедентов или вариантов использования. Диаграмма прецедентов представлена на рисунке 1.2.

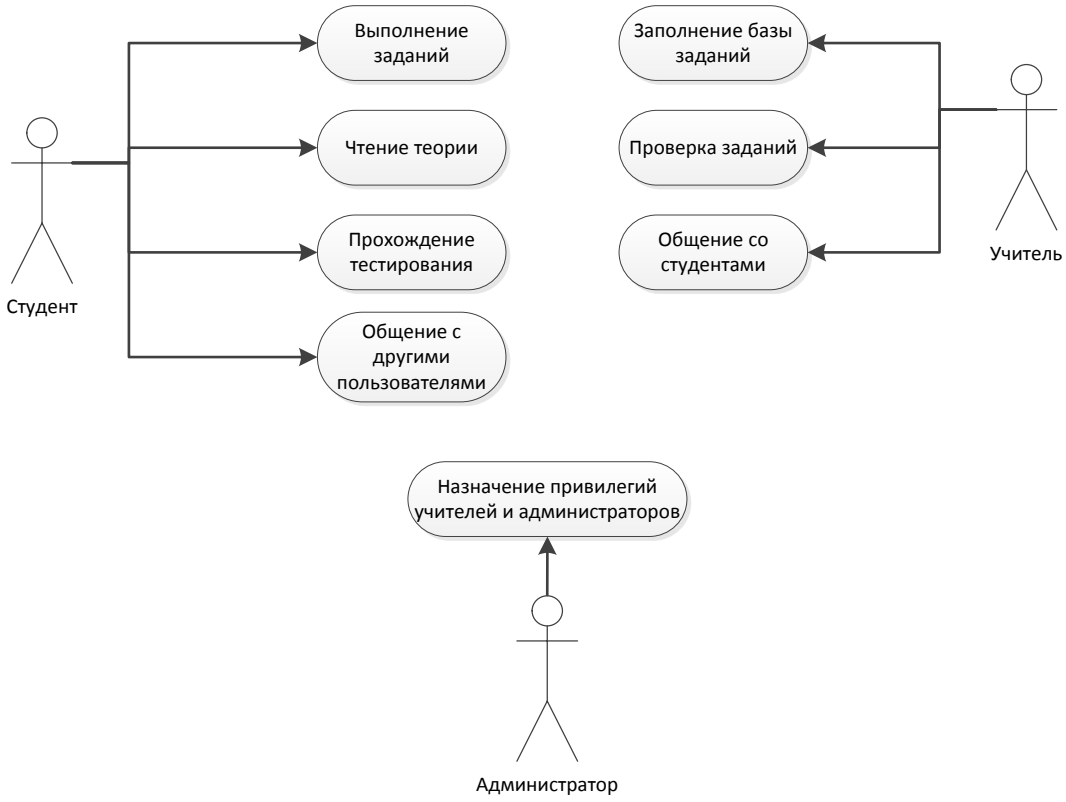


Рисунок 1.2 – Диаграмма прецедентов

Из нее видно, что пользоваться базой могут 3 группы пользователей: студенты, учителя и администраторы

Диаграмма классов

Диаграмма классов, показанная на рисунке 1.3, в UML является частным случаем ER-диаграммы. ER-диаграммы используются для логического проектирования баз данных. Главное их отличие: в ER-диаграмме уделяется внимание структуре данных, а в диаграмме классов – поведению классов.

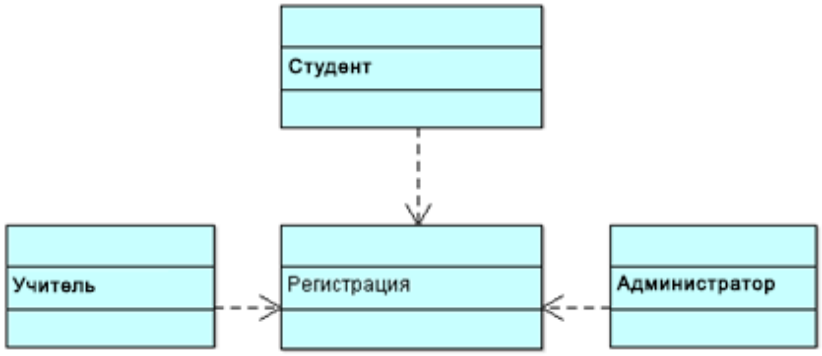


Рисунок 1.3 – Диаграмма классов

Диаграмма развертывания

Диаграмма развертывания, представленная на рисунке 1.4, Deployment diagram в UML моделирует физическое развертывание артефактов на узлах. Например, чтобы описать web-сайт диаграмма развертывания должна показывать, какие аппаратные компоненты («узлы») существуют (например, web-сервер, сервер базы данных, сервер приложения), какие программные компоненты («артефакты») работают на каждом узле (например, web-приложение, база данных), и как различные части этого комплекса соединяются друг с другом (например, JDBC, OLEDB, RMI).

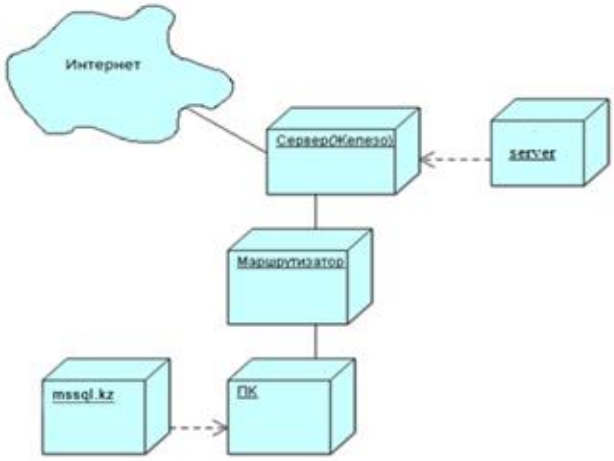


Рисунок 1.4 – Диаграмма развертывания

Узлы представляются как прямоугольные параллелепипеды с артефактами, расположенными в них, изображенными в виде прямоугольников. Узлы могут иметь подузлы, которые представляются как вложенные прямоугольные параллелепипеды. Один узел диаграммы развертывания может концептуально представлять множество физических узлов, таких как кластер серверов баз данных.

Существует два типа узлов:

- узел устройства;
- узел среды выполнения.

Узлы устройств являются физически вычислительными ресурсами со своей памятью и сервисами для выполнения программного обеспечения, такие как обычные ПК, мобильные телефоны. Узел среды выполнения – это программный вычислительный ресурс, который работает внутри внешнего узла и который предоставляет собой сервис, выполняющий другие исполняемые программные элементы.

Диаграмма развертывания нашей системы очень проста, так как не требует специального оборудования и может устанавливаться на любой ПК в компьютерных классах [2].

1.3 Проектирование БД

1.3.1 Концептуальное проектирование

Учтем параметры нашей системы, приведенные ранее, и выделим основные сущности и связи между ними, изобразим их на предварительной ER-диаграмме, которая приведена на рисунке 1.5.

Для построения ER-диаграммы воспользуемся программным обеспечением от компании Microsoft под названием Visio. Данная программа позволяет не только составить диаграмму сущность-связь, но и в дальнейшем экспортировать данную схему в любую из множества поддерживаемых ей СУБД, в моем случае это будет экспорт в SQL SERVER 2008 R2.

На основе данной диаграммы сформируем наши бизнес-правила. Бизнес правилами называются такие правила, которые устанавливают ограничения для связей между сущностями.

Далее установим связи между сущностями, опираясь на описание операций на предварительной ER-диаграмме. Точнее связи наших сущностей можно установить на основе бизнес-правил, которые, в свою очередь, построены на основе подробного описания операций.

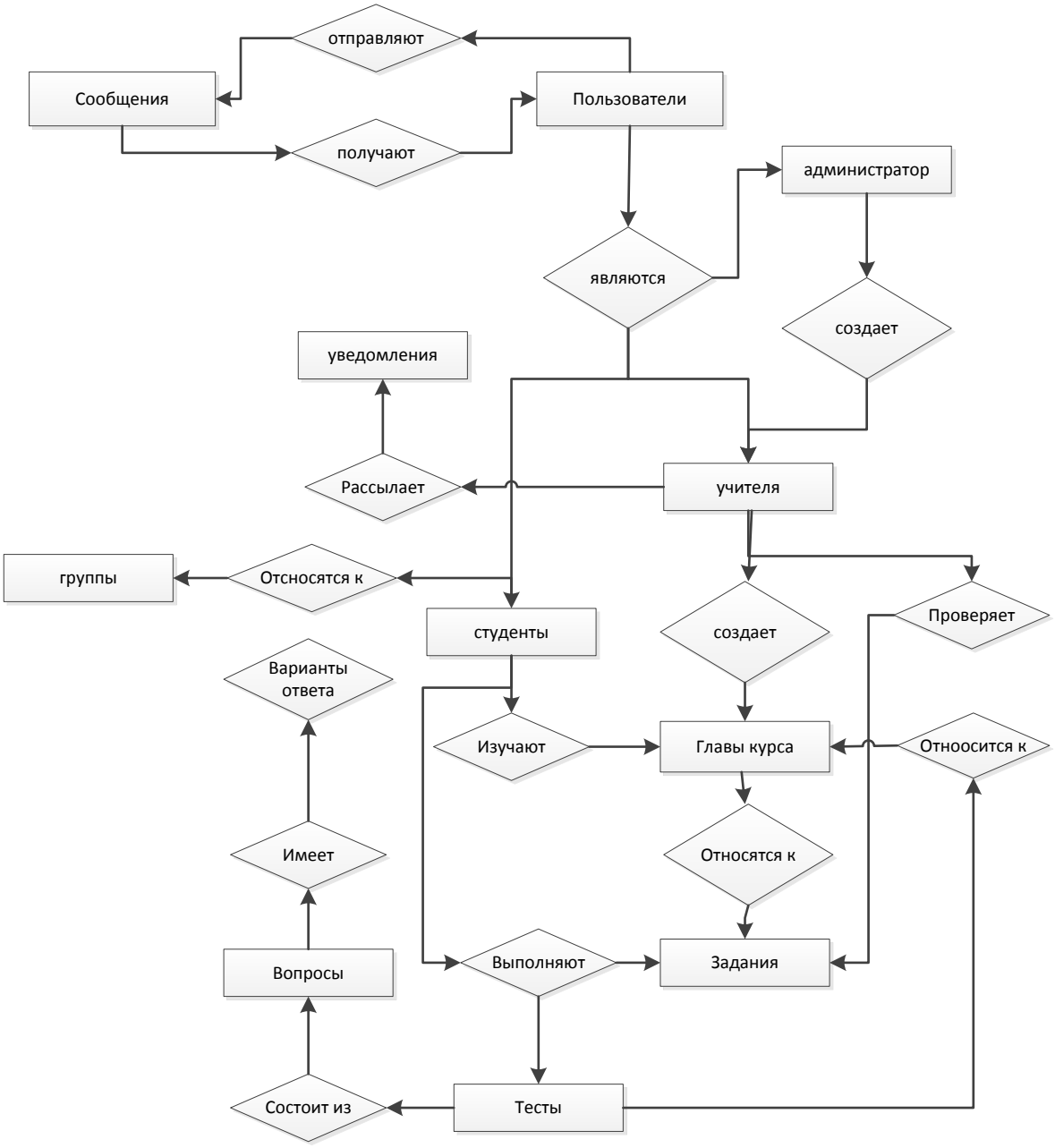


Рисунок 1.5 – Общая ER-диаграмма

1.3.2 Бизнес правила

1 Информация о пользователе заносится в две сущности. В одной из них находится техническая информация, а во второй личные данные пользователей, такие как имя, фамилия, адрес изображению аватарки, набранный пользователем опыт и номер группы в которой учится студент (рисунок 1.6).

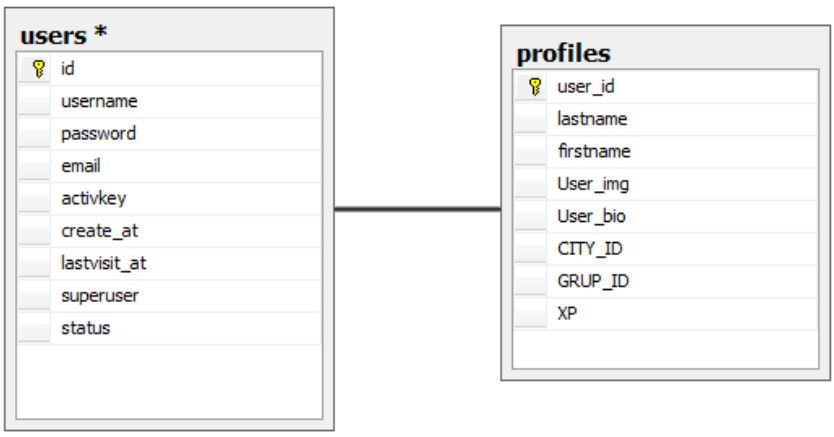


Рисунок 1.6 – Сегмент ER -диаграммы для бизнес-правила 1

2 Каждый город находится в определенной стране. В каждой стране может быть неограниченное количество городов (рисунок 1.7).

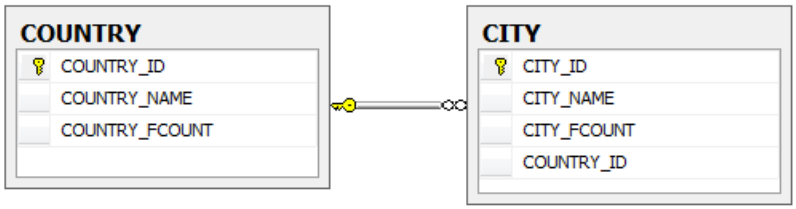


Рисунок 1.7 – Сегмент ER-диаграммы для бизнес-правила 2

3 В профиль пользователя заносится информация о том, в каком городе он прописан (рисунок 1.8).

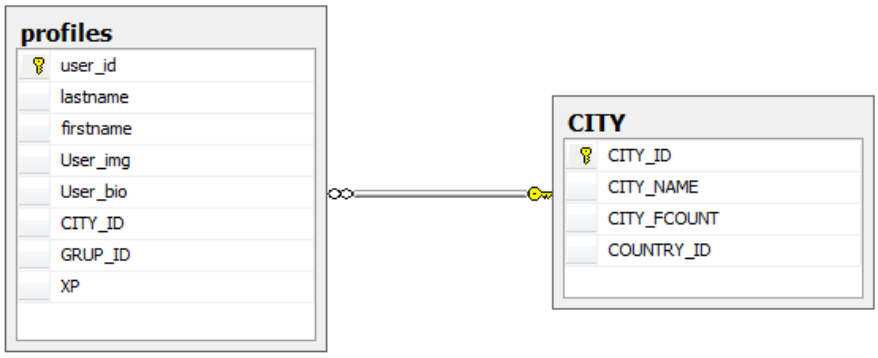


Рисунок 1.8 – Сегмент ER-диаграммы для бизнес-правила 3

4 В профиле можно указать в какой группе учится студент. Группе, в свою очередь, можно назначить старосту (рисунок 1.9).

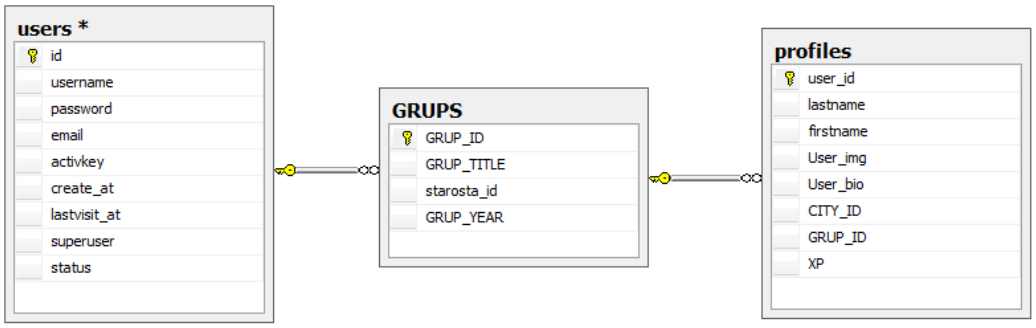


Рисунок 1.9 – Сегмент ER -диаграммы для бизнес-правила 4

5 Пользователи могут отправлять неограниченное количество сообщений другим пользователям (рисунок 1.10).

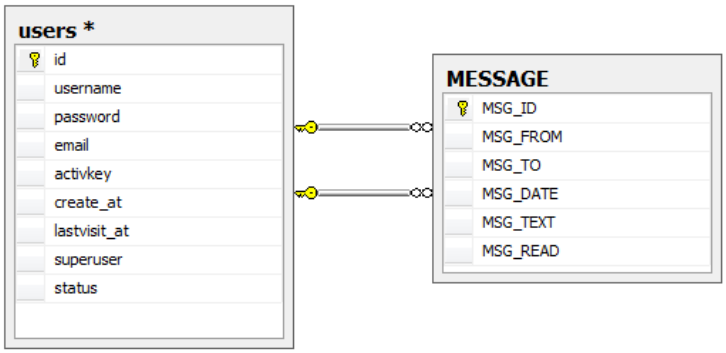


Рисунок 1.10 – Сегмент ER-диаграммы для бизнес-правила 5

6 К каждой главе в курсе можно прикрепить несколько заданий и тестов (рисунок 1.11).

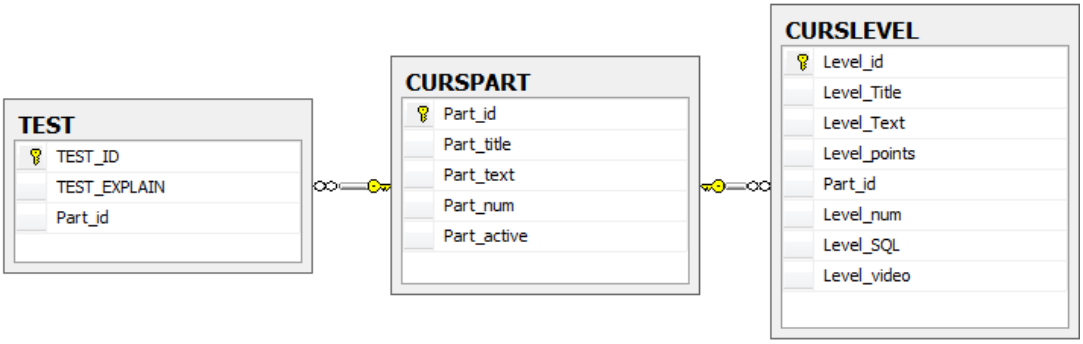


Рисунок 1.11 – Сегмент ER -диаграммы для бизнес-правила 6

7 Тест состоит из вопросов, а для каждого вопроса есть несколько вариантов ответа, один из которых верный (рисунок 1.12).

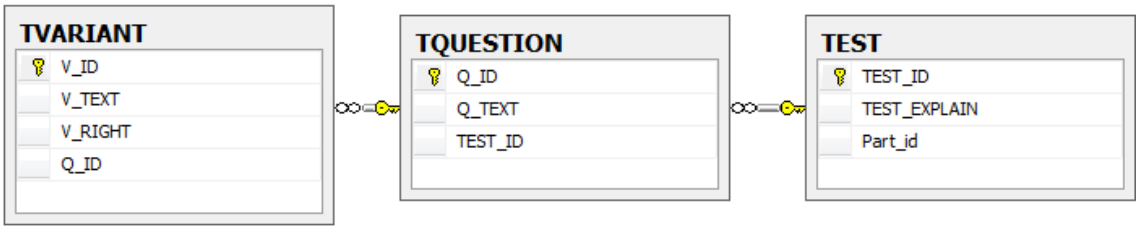


Рисунок 1.12 – Сегмент ER-диаграммы для бизнес-правила 7

8 Информация о всех попытках сдачи пользователями теста заносится в специальную таблицу (рисунок 1.13).

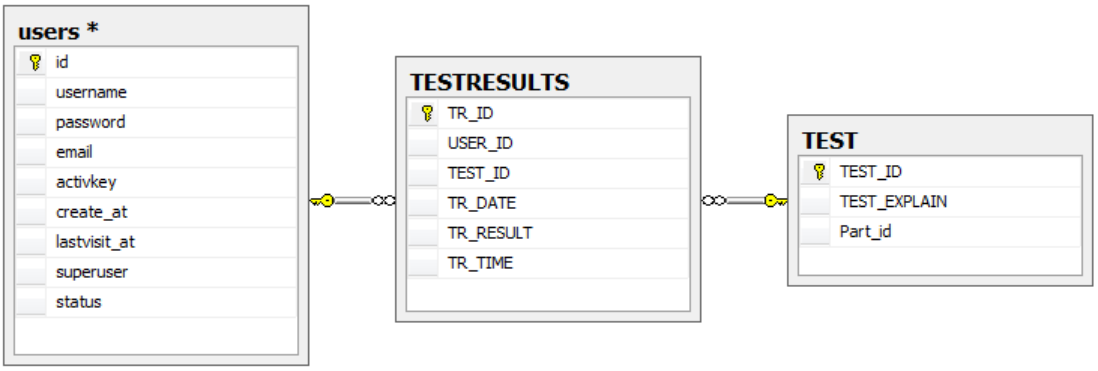


Рисунок 1.13 – Сегмент ER-диаграммы для бизнес-правила 8

9 Выполнив задание, решение попадает в очередь на проверку учителем (рисунок 1.14).

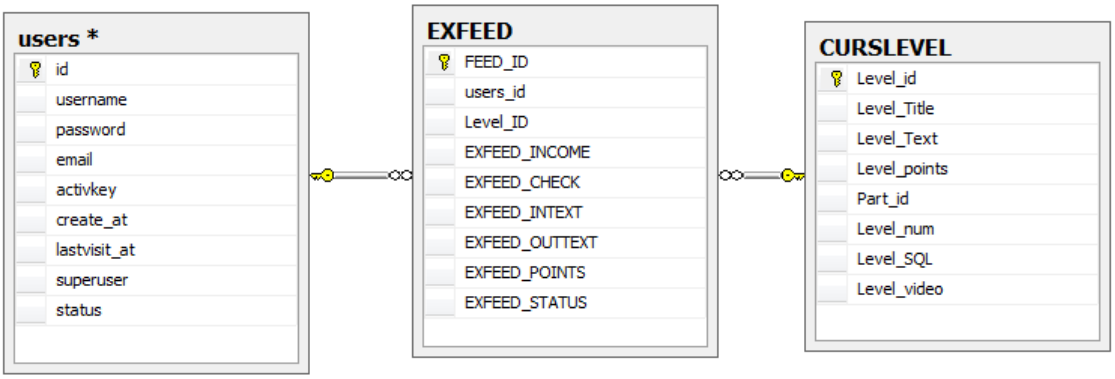


Рисунок 1.14 – Сегмент ER-диаграммы для бизнес-правила 9

10 Выполнив некие условия пользователь системы может зарабатывать различные достижения (рисунок 1.15).

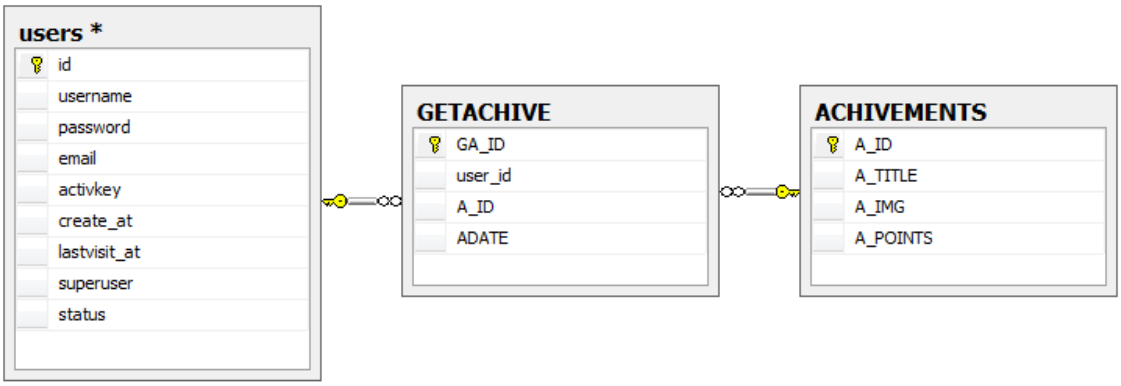


Рисунок 1.15 – Сегмент ER -диаграммы для бизнес-правила 10

11 Пользователи, обладающие привилегиями учителей могут отсылать всем остальным уведомления (рисунок 1.16).

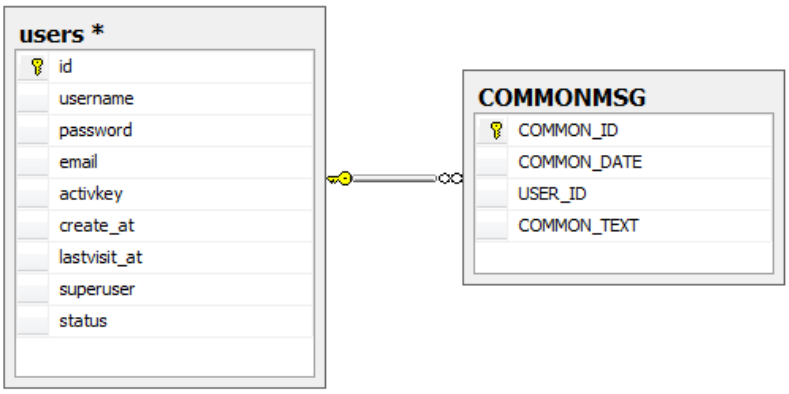


Рисунок 1.16 – Сегмент ER-диаграммы для бизнес-правила 11

12 Не может быть пользователей с одинаковыми логинами или электронными адресами.

13 Пользователи делятся на 3 уровня доступа к информации: студент, преподаватель, администратор.

1.3.3 Разработка и построение подробной ER-диаграммы на основании бизнес правил

Концептуальная модель интерактивной обучающей системы для курса «Проектирование баз данных» представлена на рисунке 1.17.

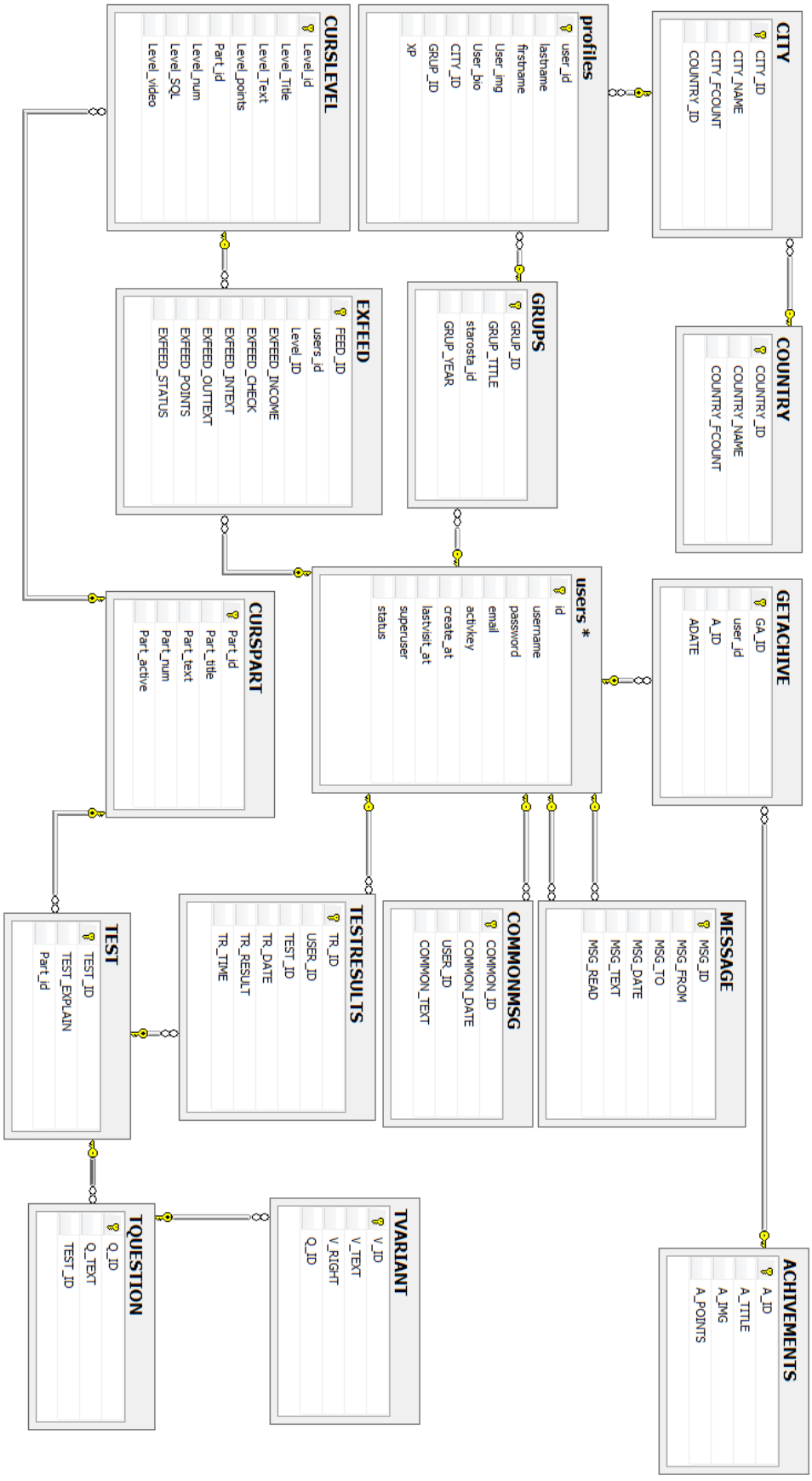


Рисунок 1.17 – Окончательный вариант концептуальной модели интерактивной обучающей системы

1.3.4 Анализ информационных задач и круга пользователей системы

С данной обучающей системой могут работать следующие группы пользователей:

- студент;
- учитель;
- администратор.

При работе с системой студент должен иметь возможность совершать следующие действия:

1 Изменять свои личные данные.
2 Читать лекционные материалы.
3 Выполнять практические задания.
4 Проходить тестирование.
5 Общаться с учителями и другими студентами.
6 Следить за своим прогрессом.
7 Соревноваться с другими студентами по количеству набранных очков.

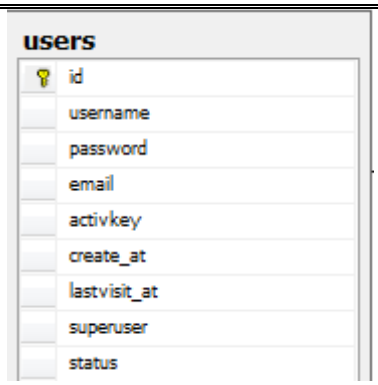
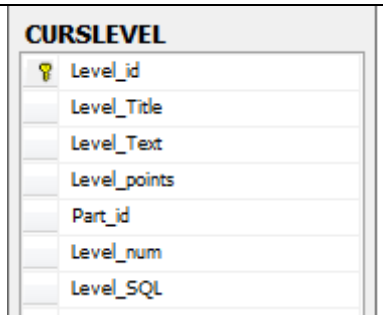
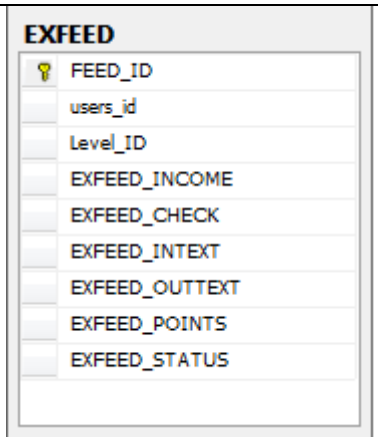
Учитель должен иметь возможность совершать следующие действия:

1 Проверку решенных заданий студентов.
2 Просматривать информацию о студентах.
3 Иметь доступ к информации о дате, времени, результате прохождения тестов каждым студентов.
4 При необходимости, совершать массовую рассылку напоминаний всем пользователям системы.
5 Вступать в диалог с любым пользователем системы.
6 Управлять базой лекций, заданий и тестов.
7 Управлять очередью заданий на проверку.
8 Администратор должен иметь возможность совершать следующие действия:
9 Выдавать пользователям привилегии учителей и администраторов.
10 В конце учебного года дезактивировать аккаунты выпускников.
11 Следить за работоспособностью всей системы.

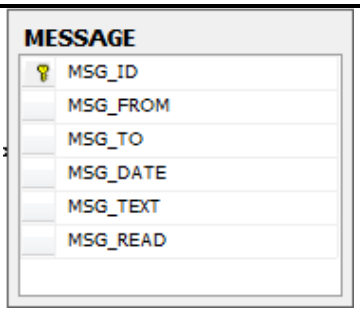
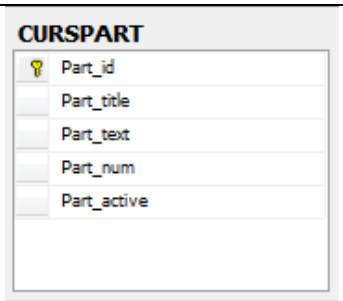
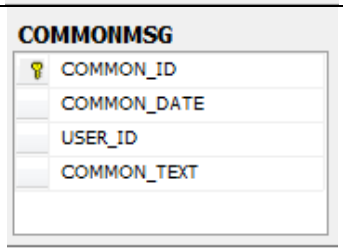
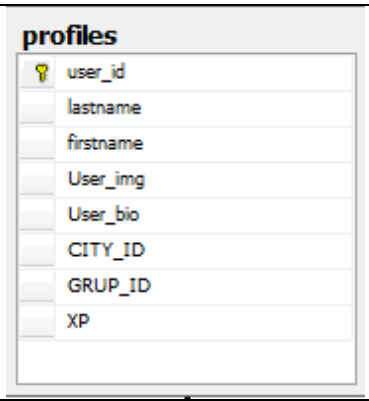
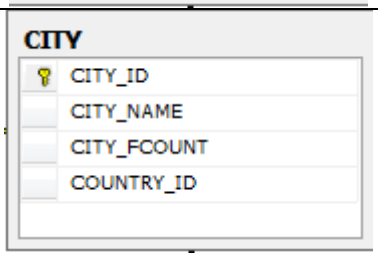
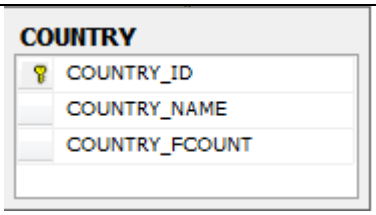
1.3.5 Определение атрибутов каждой сущности

В таблице 1.1 представлены атрибуты каждой из сущностей созданной базы данных. Данная таблица необходима нам для того, чтобы продолжить дальнейшее проектирование базы данных. В ней определяются все атрибуты, которые будут в таблицах, а также приводится их краткое описание.

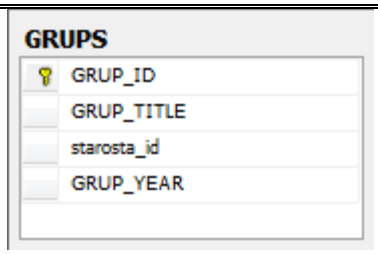
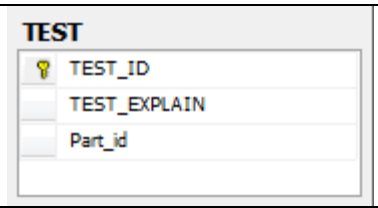
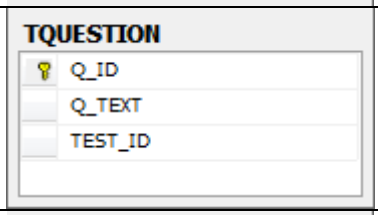
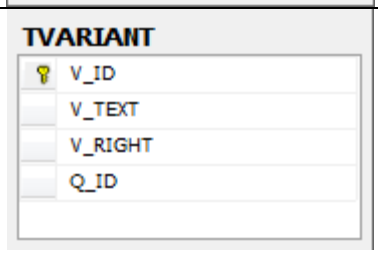
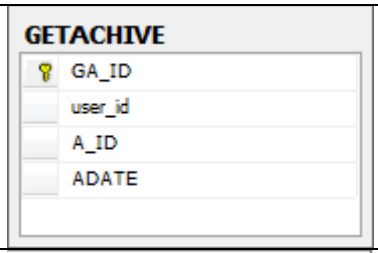
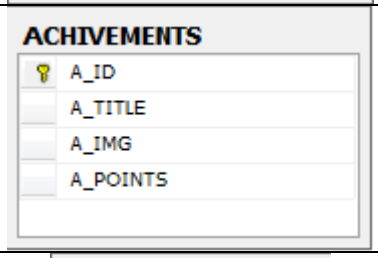
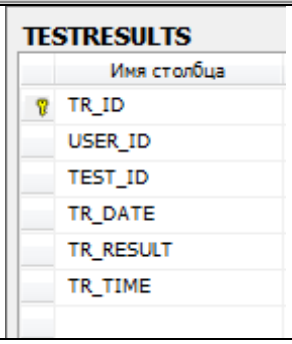
Т а б л и ц а 1.1 – Определение атрибутов каждой сущности

Сегмент ER модели	Описание
	Таблица USERS (Пользователи) содержит следующие данные: ID – идентификатор пользователя USERNAME – логин для входа в систему EMAIL – электронный адрес пользователя ACTIVKEY– уникальный ключ активации. который высылается пользователю при регистрации, CREATE_AT – дата и время создания аккаунта LASTVISIT_AT – дата и время последнего захода пользователя на сайт SUPERUSER – привилегии пользователя – 0 студент, 1 учитель, 2 администратор системы STATUS – статус активации аккаунта, 0 – не активизирован, 1- активизирован
	Таблица CURSLEVEL (Главы курса) содержит следующие данные: LEVEL_ID – уникальный номер задания LEVEL_TITLE – заголовок задания LEVEL_TEXT – содержание задания LEVEL_POINTS – очки, начисляемые студенту после выполнения PART_ID – идентификатор главы, к которой относится задание LEVEL_NUM – порядок в списке заданий к главе LEVEL_SQL – первоначальный код, с которым студенту предстоит работать
	Таблица EXFEED (Очередь заданий на проверку преподавателем) следующие данные: FEED_ID – уникальный номер заявки в очереди USERS_ID – идентификатор пользователя, отправившего задание на проверку LEVEL_ID – номер задания EXFEED_INCOME – дата и время поступления задания EXFEED_CHECK – дата и время проверки EXFEED_INTEXT – присланное решение EXFEED_OUTTEXT – замечание учителя к присланному решению EXFEED_POINTS – количество начисленных баллов EXFEED_STATUS – статус задания: 0 – не проверенно 1 – выполнено верно 2 – с ошибкой

Продолжение таблицы 1.1

Сегмент ER модели	Описание
	Таблица MESSAGE (Сообщения) содержит следующие данные: MSG_ID – идентификатор сообщения MSG_FROM – от кого отправлено сообщение MSG_TO – кому отправлено сообщение MSG_DATE – дата и время отправления сообщения MSG_TEXT – текст сообщения MSG_READ – прочитано или нет сообщение, 0 – не прочитано, 1 – прочитано
	Таблица CURSPART (Главы курса) содержит следующие данные: PART_ID – уникальный номер главы PART_TITLE – заголовок главы PART_TEXT – текст главы PART_NUM – порядок в списке глав PART_ACTIVE – если стоит 1 то глава видна в списке, если 0, то не видна
	Таблица COMMONMSG (Уведомления, общие для всех) содержит следующие данные: COMMON_ID – идентификатор уведомления COMMON_DATE – дата создания уведомления USER_ID – номер пользователя, рассылающего уведомления COMMON_TEXT – текст уведомления
	Таблица PROFILES (Профиль пользователя) содержит следующие данные: USER_ID- идентификатор категории машины LASTNAME – Фамилия пользователя FIRSTNAME – Имя пользователя USER_IMG – адрес файла с аватаром пользователя USER_BIO – небольшая информация о пользователе CITY_ID – идентификатор города GRUP_ID – идентификатор группы XP – полученный опыт
	Таблица CITY (город) содержит следующие данные: CITY_ID – номер города CITY_NAME – название города CITY_FCOUNT – количество зарегистрированных пользователей из этого города COUNTRY_ID – номер страны, в которой расположен город
	Таблица COUNTRY (Страна) содержит следующие данные: COUNTRY_ID – уникальный номер страны COUNTRY_NAME – название страны COUNTRY_FCOUNT – количество пользователей

Окончание таблицы 1. 1

Сегмент ER модели	Описание
	Таблица GRUPS (Группы) содержит следующие данные: GRUP_ID – номер группы GRUP_TITLE – название группы STAROSTA_ID – номер студента, который является старостой GRUP_YEAR – год поступления
	Таблица TEST (тесты) содержит следующие данные: TEST_ID – уникальный номер теста TEST_EXPLAIN – описание теста PART_ID – номер главы, к которой относится данный тест
	Таблица TQUESTION (Тестовые вопросы) содержит следующие данные: Q_ID – уникальный номер вопроса Q_TEXT – текст вопроса TEST_ID – тест, к которому относится данный вопрос
	Таблица TVARIANT (Варианты ответов) содержит следующие данные: V_ID – номер варианта ответа V_TEXT – текст варианта ответа V_RIGHT – если 1 – правильный ответ, 2 - неправильный Q_ID – номер вопроса, вариантом ответа на который является данный вариант
	Таблица GETACHIVE (Получение достижения) содержит следующие данные: GA_ID – номер записи в журнале достижений USER_ID – пользователь, получивший достижение A_ID – номер достижения ADATE – дата получения достижения
	Таблица ACHIVEMENTS (Достижения) содержит следующие данные: A_ID – уникальный номер достижения A_TITLE – название достижения A_IMG – иконка достижения A_POINTS – очки, полученные за достижение
	Таблица TESTRESULTS (Результаты) содержит следующие данные: TR_ID – идентификатор результата USER_ID – номер студента, который проходил тест TEST_ID – идентификатор теста TR_DATE – дата и время окончания прохождения теста TR_RESULT – результат студента в процентах TR_TIME – время, затраченное на прохождение

1.4 Составление реляционных отношений

Каждое реляционное отношение соответствует одной сущности и в него вносятся все атрибуты сущности. Для каждого отношения необходимо определить первичный ключ и внешние ключи (если они есть).

Отношения приведены в таблицах 1.2 – 1.22. Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной. Обязательное поле для краткости обозначено not null, необязательное – null. Типы данных обозначаются так: number – числовой, varchar2 – символьный, date – дата [3].

Т а б л и ц а 1.2 – CURSLEVEL

Содержание поля	Имя поля	Тип, длина	Примечание
ID задания	LEVEL_ID	int	Первичный ключ
Название	LEVEL_TITLE	varchar (255)	Not null
Текст	LEVEL_TEXT	text	null
Баллы	LEVEL_POINTS	int	Not null
ID главы	PART_ID	int	Not null(внешний ключ)
Порядковый номер	LEVEL_NUM	int	null
Первоначальный код	LEVEL_SQL	text	null

Т а б л и ц а 1.3 – CURSPART

Содержание поля	Имя поля	Тип, длина	Примечание
ID Главы	PART_ID	int	первичный ключ
Название главы	PART_TITLE	Varchar(255)	Not null
Текст	PART_TEXT	text	null
Порядковый номер	PART_NUM	int	null
Видимость	PART_ACTIVE	int	null

Т а б л и ц а 1.4 – TVARIANT

Содержание поля	Имя поля	Тип, длина	Примечание
ID Варианта	V_ID	int	Первичный ключ
Текст	V_TEXT	text	Not null
Правильность	V_RIGHT	int	null
Номер вопроса	Q_ID	int	Внешний ключ

Т а б л и ц а 1.5 – TESTRESULTS

Содержание поля	Имя поля	Тип, длина	Примечание
ID результата	TR_ID	int	Первичный ключ
ID пользователя	USER_ID	int	Внешний ключ
ID теста	TEST_ID	int	Внешний ключ
Дата получения результата	TR_DATE	datetime	null
Результат	TR_RESULT	int	Not null
Время прохождения	TR_TIME	int	null

Т а б л и ц а 1.6 – TEST

Содержание поля	Имя поля	Тип, длина	Примечание
ID теста	TEST_ID	int	Первичный ключ
Введение	TEST_EXPLAIN	text	null
Номер главы	PART_ID	int	Внешний

Т а б л и ц а 1.7 – TQUESTION

Содержание поля	Имя поля	Тип, длина	Примечание
ID вопроса	Q_ID	int	Первичный ключ
текст	Q_TEXT	text	Not null
ID теста	TEST_ID	int	Внешний ключ

Т а б л и ц а 1.8 – EXFEED

Содержание поля	Имя поля	Тип, длина	Примечание
ID очереди	FEED_ID	Int	Первичный ключ
ID пользователя	USERS_ID	Int	Внешний ключ
ID задания	LEVEL_ID	Int	Внешний ключ
Дата решения	EXFEED_INCOME	Datetime	Null
Дата проверки	EXFEED_CHECK	Datetime	Null
Отправленный текст	EXFEED_INTEXT	Text	Null
Присланный текст	EXFEED_OUTTEXT	Text	Null
Баллы	EXFEED_POINTS	Int	Not null
Статус	EXFEED_STATUS	int	null

Т а б л и ц а 1.9 – COMMONMSG

Содержание поля	Имя поля	Тип, длина	Примечание
ID уведомления	COMMON_ID	Int	Первичный ключ
Дата уведомления	COMMON_DATE	Datetime	Not null
ID пользователя	USER_ID	Int	Внешний ключ
Текст уведомления	COMMON_TEXT	text	Not null

Т а б л и ц а 1.10 – MESSAGE

Содержание поля	Имя поля	Тип, длина	Примечание
ID сообщения	MSG_ID	int	Первичный ключ
От кого сообщение	MSG_FROM	Int	null
Кому сообщение	MSG_TO	Int	Not null
Дата отправки	MSG_DATE	Datetime	Null
Текст	MSG_TEXT	Text	Not null
Проверка	MSG_READ	int	null

Т а б л и ц а 1.11 – COUNTRY

Содержание поля	Имя поля	Тип, длина	Примечание
ID страны	COUNTRY_ID	int	Первичный ключ
Название страны	COUNTRY_NAME	Varchar(200)	Not null
Количество польз.	COUNTRY_FCOUNT	bigint	null

Т а б л и ц а 1.12- USERS

Содержание поля	Имя поля	Тип, длина	Примечание
ID пользователя	ID	int	Первичный ключ
Логин	USERNAME	Varchar(20)	Not null
Пароль	PASSWORD	Varchar(128)	Not null
Адрес	EMAIL	Varchar(128)	Not null
Ключ активации	ACTIVKEY	Varchar(128)	Not null
Дата создания	CREATE_AT	Datetime	Not null
Последний вход	LASTVISIT_AT	Datetime	Not null
Категория	SUPERUSER	Int	Not null
Статус	STATUS	int	Not null

Т а б л и ц а 1.13 – CITY

Содержание поля	Имя поля	Тип, длина	Примечание
ID города	CITY_ID	Int	Первичный ключ
Название города	CITY_NAME	Nchar(200)	Not null
Количество зарегистрированных	CITY_FCOUNT	bigint	Null
ID страны	COUNTRY_ID	int	Not null(Внешний ключ)

Т а б л и ц а 1.14 – GRUPS

Содержание поля	Имя поля	Тип, длина	Примечание
ID группы	GRUP_ID	int	Первичный ключ
Название	GRUP_TITLE	Varchar(7)	Not null
ID старосты	STAROSTA_ID	int	Null(Внешний ключ)
Год поступления	GRUP_YEAR	int	Not null

Т а б л и ц а 1.15 – PROFILES

Содержание поля	Имя поля	Тип, длина	Примечание
ID пользователя	USER_ID	int	Первичный ключ
Фамилия	LASTNAME	varchar(50)	Not null
Имя	FIRSTNAME	varchar(50)	Not null
Аватара	USER_IMG	varchar(255)	null
Сведения о себе	USER_BIO	Text	Null
ID города	CITY_ID	Int	Null(Внешний ключ)
ID группы	GROUP_ID	Int	Null(Внешний ключ)
Опыт	XP	int	null

Т а б л и ц а 1.16 – GETACHIVE

Содержание поля	Имя поля	Тип, длина	Примечание
ID записи	GA_ID	Int	Первичный ключ
ID пользователя	USER_ID	Int	Not null(Внешний ключ)
ID достижения	A_ID	Int	Not null(Внешний ключ)
Дата получения достижения	ADATE	datetime	null

Т а б л и ц а 1.17 – ACHIVEMENTS

Содержание поля	Имя поля	Тип, длина	Примечание
ID достижения	A_ID	int	Первичный ключ
Название достижения	A_TITLE	Varchar(255)	Not null
Иконка	A_IMG	Varchar(255)	Null
Баллы	A_POINTS	int	null

1.4.1 Расчет места для хранения БД

Перед запуском БД необходимо выставить необходимые размеры первоначальной БД и ее роста. На этом этапе, необходимо знать какой объем памяти будет занимать создаваемая база данных. Объём внешней памяти, необходимый для функционирования системы, складывается из двух составляющих: память, занимаемая модулями СУБД (ядро, утилиты, вспомогательные программы), и память, отводимая под данные ($М_Д$). Наиболее существенным обычно является $М_Д$. Объём памяти, занимаемый программными модулями пользователя, обычно невелик по сравнению с объёмом самих данных, поэтому может не учитываться. В проекте рассчитывается предполагаемый максимальный объем памяти занимаемой БД. Расчет физической памяти приводится в таблицах 1.18 – 1.33.

Т а б л и ц а 1.18 – Расчет физической памяти для таблицы CURSLEVEL

Имя поля	Тип, длина	Длина(байт)
LEVEL_ID	int	4
LEVEL_TITLE	varchar (255)	255
LEVEL_TEXT	text	500
LEVEL_POINTS	int	4
PART_ID	int	4
LEVEL_NUM	int	4
LEVEL_SQL	text	200

Общая длина строки: 971 байт
Число строк: ~ 500
Общий объем требуемой памяти: ~ 485 500 байт

Т а б л и ц а 1.19 – Расчет физической памяти для таблицы CURSPART

Имя поля	Тип, длина	Длина (байт)
PART_ID	int	4
PART_TITLE	Varchar(255)	255
PART_TEXT	text	30000
PART_NUM	int	4
PART_ACTIVE	int	4

Общая длина строки: 30267 байт
Число строк: ~ 15

Общий объем требуемой памяти: ~ 454 005 байт

Т а б л и ц а 1.20 – Расчет физической памяти для таблицы TEST

Имя поля	Тип, длина	Длина (байт)
TEST_ID	int	4
TEST_EXPLAIN	text	500
PART_ID	int	4

Общая длина строки: 508 байт
Число строк: ~ 20
Общий объем требуемой памяти: ~ 10 160 байт

Т а б л и ц а 1.21 – Расчет физической памяти для таблицы TVARIANT

Имя поля	Тип, длина	Длина (байт)
V_ID	int	4
V_TEXT	text	500
V_RIGHT	int	4
Q_ID	int	4

Общая длина строки: 512 байт
Число строк: ~ 3 000
Общий объем требуемой памяти: ~ 1 536 000 байт

Т а б л и ц а 1.22 – Расчет физической памяти для таблицы TQUESTION

Имя поля	Тип, длина	Длина (байт)
Q_ID	int	4
Q_TEXT	text	500
TEST_ID	int	4

Общая длина строки: 508 байт
Число строк: ~ 600
Общий объем требуемой памяти: ~ 304 800 байт

Т а б л и ц а 1.23 – Расчет физической памяти для таблицы TESTRESULT

Имя поля	Тип, длина	Примечание
TR_ID	int	4
USER_ID	int	4
TEST_ID	int	4
TR_DATE	datetime	3
TR_RESULT	int	4
TR_TIME	int	4

Общая длина строки: 23 байт
Число строк: ~ 10 000
Общий объем требуемой памяти: ~ 230 000 байт

Т а б л и ц а 1.24 – Расчет физической памяти для таблицы EXFEED

Имя поля	Тип, длина	Длина (байт)
FEED_ID	Int	4
USERS_ID	Int	4
LEVEL_ID	Int	4
EXFEED_INCOME	Datetime	3
EXFEED_CHECK	Datetime	3
EXFEED_INTEXT	Text	3000
EXFEED_OUTTEXT	Text	3000
EXFEED_POINTS	Int	4
EXFEED_STATUS	int	4

Общая длина строки: 6029 байт
Число строк: ~ 12 000
Общий объем требуемой памяти: ~ 72 348 000 байт

Т а б л и ц а 1.25 – Расчет физической памяти для таблицы COMMONMSG

Имя поля	Тип, длина	Длина (байт)
COMMON_ID	Int	4
COMMON_DATE	Datetime	3
USER_ID	Int	4
COMMON_TEXT	text	500

Общая длина строки: 511 байт
Число строк: ~ 200
Общий объем требуемой памяти: ~ 102 200 байт

Т а б л и ц а 1.26 – Расчет физической памяти для таблицы MESSAGE

Имя поля	Тип, длина	Длина (байт)
MSG_ID	int	4
MSG_FROM	Int	4
MSG_TO	Int	4
MSG_DATE	Datetime	3
MSG_TEXT	Text	500
MSG_READ	int	4

Общая длина строки: 519 байт
Число строк: ~ 50 000
Общий объем требуемой памяти: ~ 25 950 000 байт

Т а б л и ц а 1.27 – Расчет физической памяти для таблицы CITY

Имя поля	Тип, длина	Длина (байт)
CITY_ID	Int	4
CITY_NAME	Nchar(200)	200
CITY_FCOUNT	bigint	8
COUNTRY_ID	int	4

Общая длина строки: 216 байт
Число строк: ~ 50
Общий объем требуемой памяти: ~ 10 800 байт

Т а б л и ц а 1.28 – Расчет физической памяти для таблицы USERS

Имя поля	Тип, длина	Длина (байт)
ID	int	4
USERNAME	Varchar(20)	20
PASSWORD	Varchar(128)	128
EMAIL	Varchar(128)	128
ACTIVKEY	Varchar(128)	128
CREATE_AT	Datetime	3
LASTVISIT_AT	Datetime	3
SUPERUSER	Int	4
STATUS	int	4

Общая длина строки: 422 байт
Число строк: ~ 200
Общий объем требуемой памяти: ~ 84 400 байт

Т а б л и ц а 1.29 – Расчет физической памяти для таблицы COUNTRY

Содержание поля	Имя поля	Тип, длина	Длина (байт)
ID страны	COUNTRY_ID	int	4
Название страны	COUNTRY_NAME	Nchar(200)	200
Количество зарегистрированных	COUNTRY_FCOUNT	bigint	8

Общая длина строки: 212 байт
Число строк: ~ 10
Общий объем требуемой памяти: ~ 2 120 байт

Т а б л и ц а 1.30 – Расчет физической памяти для таблицы PROFILE

Имя поля	Тип, длина	Примечание
USER_ID	int	4
LASTNAME	varchar(50)	50
FIRSTNAME	varchar(50)	50
USER_IMG	varchar(255)	255
USER_BIO	Text	500
CITY_ID	Int	4
GROUP_ID	Int	4
XP	int	4

Общая длина строки: 871 байт
Число строк: ~ 200
Общий объем требуемой памяти: ~ 174 200 байт

Т а б л и ц а 1.31 – Расчет физической памяти для таблицы GETACHIVE

Имя поля	Тип, длина	Примечание
GA_ID	Int	4
USER_ID	Int	4
A_ID	Int	4
ADATE	datetime	3

Общая длина строки: 15 байт
Число строк: ~ 3 000
Общий объем требуемой памяти: ~ 45 000 байт

Т а б л и ц а 1.32 – Расчет физической памяти для таблицы ACHIVEMENTS

Имя поля	Тип, длина	Длина (байт)
A_ID	int	4
A_TITLE	Varchar(255)	255
A_IMG	Varchar(255)	255
A_POINTS	int	4

Общая длина строки: 518 байт
Число строк: ~ 50
Общий объем требуемой памяти: ~ 25 900 байт

Т а б л и ц а 1.33 – Расчет физической памяти для таблицы GRUPS

Имя поля	Тип, длина	Длина (байт)
GRUP_ID	int	4
GRUP_TITLE	Varchar(7)	7
STAROSTA_ID	int	4
GRUP_YEAR	int	4

Общая длина строки: 19 байт
Число строк: ~ 15
Общий объем требуемой памяти: ~ 285 байт

Таким образом, из полученных данным можно рассчитать приблизительный максимальный объем базы данных. Объем будет равен сумме объемов всех таблиц

Мд = 230000 +304800 +1536000 +10160 +454005 +485500 +102200 +72348000 + 25950000 +84400+10800+2120+285+174200+45000+25900= 101763370 байт = **97** (Мб).

Объём памяти, занимаемый программными модулями пользователя, обычно невелик по сравнению с объёмом самих данных, поэтому может не учитываться. Требуемый объём оперативной памяти определяется на основании анализа интенсивности запросов и объёма результирующих данных.

1.4.2 Задание места хранения БД. Загрузка базы данных

Таким образом, проведя физическое проектирование сделаем вывод, что размер данных БД за первый год работы базы будет приблизительно равен **97** Мб. В течение следующего года этот размер будет увеличен в 2 раза. Исходя из этих данных перед запуском БД выставим необходимые параметры размера и роста.

1.5 Используемые технологии программирования

1.5.1 Фреймворк YII

Фреймворк (англ. framework – каркас, структура) – структура программной системы; программное обеспечение, которое помогает при разработке и объединение разных компонентов большого программного проекта. Также можно рассуждать о каркасном методе, как о методе построения программ, где всякая структура программы строится из двух частей: первая, которая является постоянной – база, не меняющаяся от программы к программе и несущая в себе ячейки, где размещается другая, уже переменная часть – точки расширения (переменные гнезда). В нее могут включаться дополнительные программы, язык сценариев, библиотеки кода, и другое программное обеспечение, которое упрощает разработку и систематизацию различных компонентов больших информационных систем. Чаще всего объединение происходит за счёт использования общего API.

Таким образом мы получили, что Yii – это крайне эффективный PHP-фреймворк, основанный на структуре компонентов, предназначенных для разработки больших веб-приложений. Фреймворк позволяет максимально эффективно использовать концепцию повторного использования кода, что, в свою очередь, крайне ускоряет процесс веб-разработки. Название фреймворка Yii (произносится как Yee или [ji:]) означает простой (англ. easy), эффективный (англ. efficient) и расширяемый (англ. extensible) [4].

Парадигма MVC

Во фреймворке Yii используется парадигма Модель-Вид-Контроллер (MVC, Model-View-Controller), крайне широко применяемый в сфере веб-программирования.

Model-view-controller (MVC, «модель-представление-поведение», «модель-представление-контроллер», «модель-вид-контроллер») – система применения нескольких шаблонов разработки, где модель данных приложения, интерфейс пользовательского взаимодействия разделены на три отдельных блока кода таким образом, что модифицируя один из компонентов, мы оказываем минимальное воздействие на все другие блоки. Данная схема разработки широко применяется при построении архитектурного каркаса, когда начинают переход от теории к реализации в конкретных случаях.

Назначение модели MVC

Главная цель применения данной концепции состоит в разделении бизнес-логики (модели) от её визуализации (представления, вида). С помощью такого разделения увеличивается вероятность повторного применения кода. Максимально полезно использовать эту концепции тогда, когда пользователю необходимо видеть те же самые данные в одно и тоже время во всевозможных контекстах или с различных точек зрения. Например, решаются такие задачи:

К одной модели разрешено подключение несколько представлений, при этом реализация модели не затрагивается. Например, какая-то информация может одновременно отображаться как в виде электронной таблицы, так и в виде гистограммы или любой другой диаграммы.

Не затрагивая реализацию представлений, у нас есть возможность изменять реакцию на пользовательские действия (ввод данных с клавиатуры, клик мышью), для этого мы просто подключим другой контроллер.

Многие разработчики являются специалистами только в одной из этих сфер: одни занимаются проектированием графического интерфейса, а другие ответственны за бизнес-логику. Таким образом, разделение данных, контроллеров и представлений на слабосвязанные блоки позволяет достичь того, что инженеры, занимающиеся созданием бизнес-логики (модели), вообще не будут в курсе того, как их данные будут отображаться в представлениях.

Концепция модели MVC

Использование шаблона MVC дает возможность разделять информацию, интерфейс и отслеживание действий пользователя на три независимых компонента:

Модель (англ. Model). Модель предоставляет собой некие данные и способы их обработки. Модель принимает запросы к БД, изменяя тем-самым своё состояние. Они не содержат данных о том, как эта информацию будет применяться и отображаться.

Представление, вид (англ. View). Отвечает за представление данных конечным пользователям (визуализация). Одним из примеров представления можно назвать форму с некими полями или графическими элементами.

Контроллер (англ. Controller). Ответственен за взаимодействие между клиентами и web-приложением: отслеживают ввод данных пользователем и с помощью моделей и видов обеспечивают необходимую реакцию системы.

Крайне важно заметить, что как и вид, так и контроллер связаны с моделями. Тем не менее сама модель никак зависит ни от представления, ни от контроллеров. Таким образом добиваются создания такого разделения, что модели строятся независимо от того, как они должны отображаться и какие действия над ними будут проводить пользователи. Одна и та же модель может использоваться в неограниченном количестве представлений и контроллеров.

Для создания парадигмы Model-View-Controller используется огромное число шаблонов разработки (в зависимости от уровня сложности

информационной системы), Например: «наблюдатель», «стратегия», «компоновщик».

В самой простой реализации вид отделяется от модели с помощью создания между ними некого API взаимодействия, используя подписку и прослушивание действий. При каждом изменении внутренних данных в модели она оповещает все зависящие от неё виды, и он обновляется. С этой целью применяется шаблон «наблюдатель». При обработке события представление выбирает, в зависимости от ситуации, необходимый контроллер, который обеспечит нужную связь с моделью. В этом случае применяется шаблон «стратегия», или же можно изменять данные с поощью шаблона «команда». А для однотипного обращения к подобъектам крайне сложного иерархического представления можно использоваться шаблон «компоновщик».Помимо всех этих шаблонов, имеется возможность применять и другие шаблоны разработки, к примеру, «фабричный метод», который позволит задать для контроллера для соответствующего вида определенное представление [5].

На рисунке 1.18 показана структурная схема парадигмы Model-View-Controller.

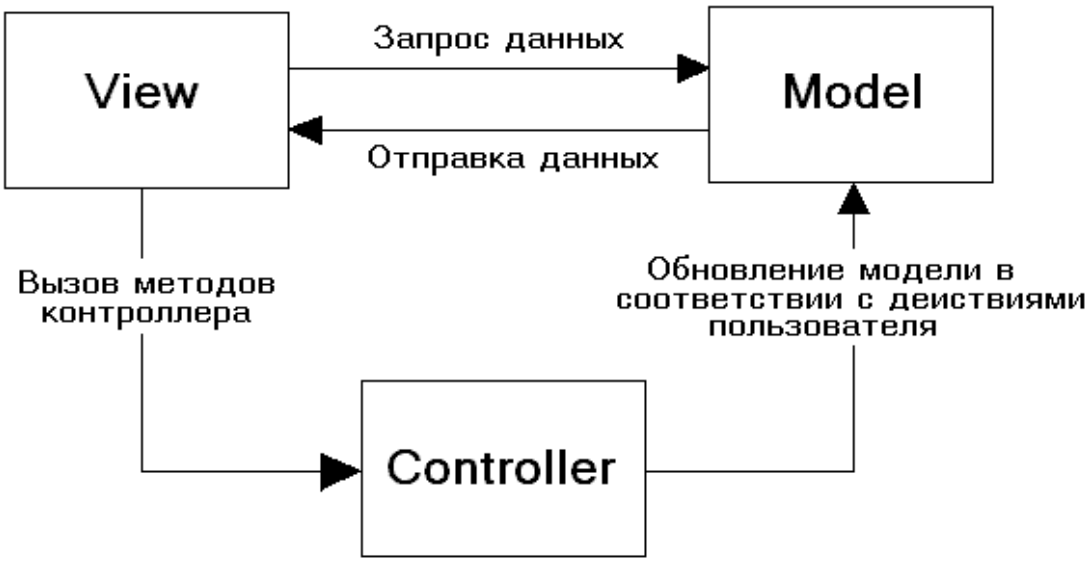


Рисунок 1.18 – Схема концепции модель-вид-контроллер

1.5.2 Библиотека JQuery

jQuery – библиотека JavaScript, направленная на облегчение взаимодействия JavaScript и HTML. Библиотека jQuery может помочь запросто получить доступ к всякому элементу DOM, обращаться к свойствам и содержимому частей DOM, управлять ими. Кроме того библиотека jQuery дает комфортный и простой API для работы с AJAX запросами.

Какие преимущества привнес jQuery в стандартный набор функций языка JavaScript?

В первую очередь, данное отношение к (x)html документу на самом деле как к xml, а не просто формально из-за разметки. На практике из этого можно сделать вывод, что я с легкостью могу получить доступ к всякому узлу (node) древовидной структуры и его атрибутам, и еще непринужденно передвигаться по ветвям.

Во-вторых, jQuery произвел настоящий прорыв в области html-javascript взаимодействия. До момента его появления последнее было если не принципиально невозможным, то уж настолько трудозатратным, что не многие отваживались с этим связываться, а само выражение «html-javascript шаблонизации» вызывало в лучшем случае улыбку. Широко бытовало мнение, что шаблонизация рациональна только средствами серверных языков программирования, но jQuery, благодаря развитому механизму обработки событий, позволил уже на уровне javascript отделить логику от представления.

Но это еще не все, проницательный ум спросит: какая же это шаблонизация, ведь таблицу же надо заполнять данными из бд, а для этого все равно нужен php цикл? На самом деле совсем не обязательно, заполнять тоже можно яваскриптом (если вы, конечно, не ориентируетесь на пользователей с отключенным javascript, а если ориентируетесь – зачем эту статью читаете?). Здесь мнения программеров расходятся в соответствии с личными предпочтениями. Лично я придерживаюсь мнения, что таблицы данных как таковые удобнее делать php циклом, а вот работать с web формами (где шаблонизация не менее актуальна) лучше исключительно с помощью jquery, способом, показанным выше.

В-третьих, это качественно новый подход к разработке пользовательского интерфейса (GUI) и визуальных эффектов на web странице. Посетите http://jqueryui.com/ и вы согласитесь со мной. Такие тривиальные компоненты как слайдер (slider), которые раньше требовали написания отдельного модуля (сам таким занимался javascript/slider.php), теперь занимают одну строчку кода

Так же дело обстоит и с визуальными эффектами появления/исчезания блоков и др.

В-четвертых, новый уровень работы с AJAX. Внести что-то новое в этой области практически не возможно (с точки зрения javascript программирования), поэтому jquery лишь до предела упрощает написание кода.

В-пятых, серьезный подход разработчиков к расширяемости позволяет тысячам программистов во всем мире самостоятельно писать модули расширения и делиться ими с остальными. На практике это означает, что перед решением любой задачи вы можете посмотреть не решил ли ее уже кто-нибудь до вас. С большой долей вероятности вам не придется начинать с нуля, а при хорошем раскладе не нужно будет ничего делать вообще [6].

1.5.3 Веб-сервер Apache

На планете создано множество великое различных веб-серверов. Они друг от друга отличаются по предназначению, и по функциональности. Моя

система основана на известном самом из них, установленном на числе подавляющем серверов компаний услуги хостинга предоставляющих. Сервер Apache оснащен всеми практически современных веб-разработок требованиями, но в то же время он легок достаточно в освоении, для того чтобы его устанавливали начинающие разработчики для отладки своих программ.

Сотрудник Национального центра приложений для суперкомпьютеров в 1994 году в Университете Иллинойса США (NCSA) по имени Маккул Роб в общее пользование первый веб-сервер выложил, который так и назвали – NCSA HTTP daemon. Сервер в узких кругах получил популярность, но летом 1994 года Маккул университет покинул, и прекратил разработки.

Небольшая заинтересованных веб-мастеров группа начала совместную работу над сервером. Общаясь через фидонет, они создавали «заплатки» и новые фичи для своего сервера. Именно они Apache Group и создали, которая разработала первую ревизию сервера Apache. в апреле 1995 года произошло это, на базу NCSA Server 1.3 были установлены практически все созданные «патчи». Так появился первый официальный публичный релиз Apache 0.6.2, а первый общедоступный релиз Apache вышел в апреле 1995 года, а релиз 1.0 вышел только в декабре. Не прекращалась ни на минуту работа над сервером, и уже очень скоро он стал одним из самых популярных, а после большого числа испытаний, 1 декабря 1995 года появилась версия 1.0,которая считалась уже пригодной для применения в высоконагруженных сайтах. И вот уже на протяжении 20 лет Apache остается абсолютно бесплатным. Быть может, именно это и определило колоссальное распространение технологии. По сведениям NetCraft, Apache на 2014 год стоит на 67% всех серверов планеты.

Apache – обладающий полным набором функций, динамический веб-сервер, распространяемый по открытой лицензии. Так что же такое Apache? Это многофункциональный, широко расширяемый веб-сервер, абсолютно соответствующий протоколу HTTP/1.1 и распространяющийся с открытым исходным кодом. Сервер может работать практически на любых распространенных операционных системах. Существуют готовые решения для установки сервера на Windows 7, Windows 8, linux, OS/2 и других, основанных на UNIX систем. При этом сервер легок в развертывании и настройке. [7]

Apache конфигурируется с помощью специальных текстовых файлов конфигурации. Главные параметры сразу установлены на «дефолтные» настройки, такие параметры сразу будут исправно функционировать, практически в любых ситуациях, однако ежели пользователю не будет хватает настроек по умолчанию штатного «Апача», значит следует выбрать из обширного набора дополнений, созданных Apache Group и другими командами. Крайне интересным преимуществом является то, что архитекторы системы ведут активные беседы с пользователями и реагируют на все предложения и замечания.

Распространенная самая сервера задача, может выполнять которую Apache – просто стоять на где-то на сервере и обеспечивать работоспособность

простого HTML-сайта. Получив запрос на одну из страниц, апач формирует и отсылает ответ браузеру. Набираете url в омни-бокс браузера, открывается нужная страница, как видите все работает достаточно просто.

С помощью сервера Apache можно производить простую аутентификацию. Функция посложнее, которая заложена в протоколе HTTP/1.1 – аутентификация пользователей. С помощью штатных средств сервера Apache вы можете разграничить доступ к определенным страницам сайта для разных пользователей, это нужно, чтобы к примеру, сделать администраторский интерфейс. Используем файлы htaccess и htpasswd, а еще дополнения mod_auth и mod_access. Юзеры разбиваются на группы, и для всякой из них назначаются свои правила доступа [8].

Сервера Apache полноценно могут работать с такой технологией, как SSI.

На рисунке 1.19 представлен алгоритм доступа к данным через сервер Apache. Веб-сервер оснащен мощной системой разграничения и делегирования доступа к различным данным. Осуществляется это с помощью подключаемого модуля Apache mod_rewrite, созданного самими разработчиками сервера. Данный модуль является одним из основных преимуществ перед создателями других веб-серверов.

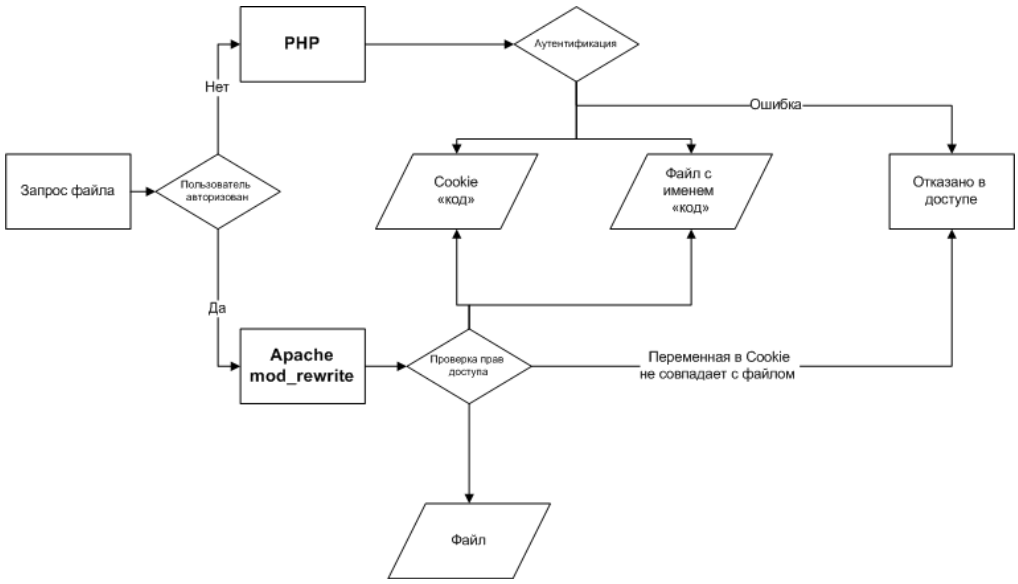


Рисунок 1.19 – Схема работы системы делегирования доступа к данным веб-сервера Apache

1.5.4 CSS фреймворк Bootstrap

Крайне важной технологией, используемой в моей работе является CSS фреймворк, созданный компанией Twitter. За последние 5 лет бутстрап стал, фактически, сандартом де-факто при разработке интерфейсов веб-приложений.

Twitter Bootstrap – это набор инструментов для создания сайтов и веб-приложений, который содержит в себе HTML и CSS шаблоны оформления

для веб-форм, кнопок, блоков навигации и прочих компонентов веб-интерфейсов, включая JavaScript расширения [9].

Так как Bootstrap использует самые современные наработки в области CSS и HTML и постоянно обновляется, нужно внимательно следить за тем, чтобы всё работало и в старых браузерах.

Основные инструменты Bootstrap

Сетки (колонки) – это объекты с заранее заданными размерами, которые удобно использовать для разметки документа. В новой версии Twitter Bootstrap 3.0 класс данных объектов называется.col-md-* – для устройств с экранами средней величины, .col-xs-* – для устройств с экстра-маленькими экранами (например, смартфонов и мобильных телефонов), .col-sm-* – для устройств с маленькими экранами (у планшетов) и.col-lg-* для устройств с большими экранами (телевизоры и широкодюймовые мониторы), где за коротким дефисом следуют простые числа от 1 до 12. Такой диапазон значений используется потому, что в концепции разработчиков Twitter Bootstrap экран делится на 12 маленьких частей. Таким образом, .col-md-2 означает что объект данного класса будет занимать 2 /12 или 16% от ширины экрана. Разумеется, в пикселях значение класса.col-* будет разным для мониторов разных размеров.

Шаблоны – к данному типу относятся фиксированный и резиновый макеты сайта или веб-приложения. Под фиксированным макетом понимается неизменяемая ширина макета сайта. Такой шаблон фиксирован по своим размерам, он не изменяется в зависимости от параметров окна браузера или экрана. Данный тип разработки очень прост, так как пользователи с различными устройствами будут видеть одно и то же представление сайта или веб-приложения. Информация не будет искажаться, но будет отображаться строго в соответствии с заданными фиксированными разработчиком параметрами. К преимуществам фиксированного шаблона можно отнести то, что задуманный вид сайта будет одинаково отображаться всегда и везде – на любых системах с любыми экранами, какое-либо искажение в данном случае почти невозможно. Однако, как и у всего на этой планете, у фиксированного шаблона есть и недостатки. Например, если шаблон отлично подходит для окон средней величины, такой негибкий дизайн будет не эстетично смотреться на маленьких или больших мониторах, к тому же возможно появление полос прокрутки, которые закроют собой полезную информацию, затруднят или полностью сведут на нет навигацию сайта или веб-приложения, что в конечном итоге отпугнёт пользователей. Резиновый дизайн – это совершенно другая концепция.

При изменении параметров окна – также изменяется и внешний вид шаблона. Такая интересная вещь происходит из-за масштабирования объектов. То есть масштабирует объект не сервер, а клиент. Неважно – на мобильном телефоне или на экране домашнего кинотеатра просматривается приложение – картинка всегда будет автоматически подстраиваться под размер монитора, становясь то уже, то шире. Благодаря этому, все элементы будут видны и

ничего не будет скрыто от пользователя. Такая открытая и дружелюбная, можно сказать даже умная приспособляемость располагает к себе клиента, вызывая у него чувство эстетического удовольствия. Однако, при неумелом проектировании разработчиком макета, чувство наслаждения может быстро перерасти в раздражение и даже злобу. Например, текст может стать нечитабельным – если нет фиксированных границ, строки слишком растянутся на больших мониторах, что быстро утомляет шейные мышцы и глаза. Такая непредсказуемость негативно воздействует на ЦНС человека. Графика может неожиданно оказаться не в том месте, где пользователю предстояло её увидеть. К тому же некоторые браузеры работают с резиновым кодом отлично от других, таким образом, внешний вид приложения отобразиться некорректно. Такая непредсказуемость негативно воздействует на ЦНС человека. Приложения с резиновым макетом грузятся дольше фиксированных, и это ожидание так же расстраивает посетителя.

Типографика – это описания шрифтов и определение классов для них. Очень удобным инструментом является выделение цитат и оформление кода особым шрифтом. Специальные теги избавляют программиста от утомительного написания отдельных CSS классов для цитат и кода. Революционный тег <blackquote> заменит десятки свойств (таких как отсчитывание отступов, выделение полужирным шрифтом и курсивом и т.п.), а с тегом <code> набранный текст превратится в удобно оформленный для чтения код.

С помощью Twitter Bootstrap значительно упростился процесс оформления таблиц, вплоть до добавления функциональности сортировки. Например, с помощью класса.table-striped к таблице добавляется полосатый фон строк – серая полоса и белая. Вот так, всего лишь одним классом можно во много раз повысить удобочитаемость таблицы – строки не сливаются, и смотреть на них – одно удовольствие. А класс.table-bordered – добавляет псевдо-трехмерные границы для всех ячеек таблицы, таким образом, выделяя их и подчеркивая значимость информации, которую они содержат. Класс.table-condensed в наше время актуален как никогда – он позволяет визуально сузить таблицу, тем самым приводя её в красивый и аккуратный вид, это очень ценится в крупных корпорациях, таких как Apple и Google. Но самое главное удобство в оформление таблиц Twitter Bootstrap привнес с помощью контекстных классов. С их помощью, в зависимости от контекста, можно акцентировать внимание на отдельных строках или столбцах таблицы. Успешное действие пользователя над таблицей окрасится в позитивный зеленый цвет при помощи класса.success, а предупредительная строка класса.warning станет красноватой. Немалую долю успеха бутстрапу принес уникальный класс .table-responsive, который позволяет появляться полосам прокрутки внутри самой таблицы. На маленьких устройствах это просто незаменимое свойство.

Twitter Bootstrap создал большое число классов для оформления форм, то есть таких элементов приложения как блоки ввода текста, паролей и других

заполняемых пользователем полей. С помощью данных нововведений легко можно расположить форму горизонтально или вертикально, настроить ширину и высоту каждого поля и установить фокус указателя мыши на определенное поле. Но больше всего полезных функций создано для оформления различных кнопок, табов, вкладок, меню и тулбара – то есть объектов для навигации по сайту. Например, раскрасить кнопку, которая обязательно должна привлекать внимание целесообразнее в красный цвет, а кнопку с предупреждением лучше всего сделать оранжевой. Для этого в Twitetr Bootstrap созданы несколько классов, рисующих кнопки определенного цвета. Практически в каждом макете для бустрапа есть готовая панель навигации – вертикальная или горизонтальная, с выпадающим меню или без. Для того чтобы вместо стандартной белой панели добавить на страницу черную панель, совсем не обязательно вручную менять цвет фона, цвет шрифта и задний фон. Всё, что нужно – это добавить к тегу класс.inverse. Twitter Bootstrap добавили очень много полезных функций, которые берегут время разработчика и позволяют ему сделать дизайн веб-приложения гораздо быстрее и лучше. Поэтому в данной работе я решил использовать именно эту замечательную технологию.

1.6 Используемые языки программирования

1.6.1 Язык PHP

PHP (англ. PHP: Hypertext Preprocessor – «PHP: гипертекста препроцессор»; первоначально Personal Home Page Tools [4] – «Инструменты для создания персональных веб-страниц»; произносится пи-эйч-пи) – скриптовый язык программирования общего назначения, интенсивно для разработки применяемый веб-приложений. Поддерживается в подавляющим большинством настоящее время хостинг-провайдеров и из лидеров одним среди программирования языков является, динамических веб-сайтов применяющихся для создания [10]. Группой энтузиастов язык и его интерпретатор разрабатываются в рамках проекта с открытым кодом [11]. Под собственной лицензией проект распространяется, GNU GPL несовместимой с.

PHP есть с типизацией динамической язык программирования, указания без необходимости типа при объявлении переменных, равно как и самого объявления переменных. Преобразования между скалярными типами зачастую осуществляются без дополнительных усилий неявно (хотя, PHP дает сильные инструменты и для явного преобразования типов).

Синтаксис PHP подобен синтаксису языка Си. Некоторые элементы, массивы и цикл foreach, ассоциативные массивы, взяты из Perl.

1.6.2 Язык JavaScript

JavaScript – скриптовый язык, само часто использующийся при создании алгоритмов, встраиваемых в веб-страницы. Есть одна из реализаций языка ECMAScript.

Бренданом Айком разработанный из компании Netscape язык был встроен в обозреватель Нетскэйп Навигатор, начиная с версии 2.0B зимой 1995 года. Изначально язык назывался Mocha, далее он был переименован в LiveScript. На синтаксис оказали влияние языки Си и Java, и, поскольку технология Java была в то время очень популярной, LiveScript переименовали в JavaScript, получив соответствующую лицензию у Sun. Реализация компании Microsoft получила название JScript. Internet Explorer поддерживает JScript, начиная с версии 3.0, выпущенной в августе 1996 г. Стандартизированная версия имеет название ECMAScript, описывается стандартом ECMA-262. Первая версия данного стандарта примерно соответствовала JavaScript 1.1 [12]. Стандартизация языка ассоциацией ECMA инициирована компанией Netscape.

Заголовок «JavaScript» есть зарегистрированный товарный знак компании Sun Microsystems, Inc.

Область применения языка

Среди сторонних продуктов, например, Java, начиная с версии 6, содержит встроенный интерпретатор JavaScript на базе Rhino. Сценарии JavaScript поддерживаются в таких приложениях Adobe, как Adobe Photoshop, Adobe Dreamweaver, Adobe Illustrator или Adobe InDesign.

JavaScript находит применение в качестве скриптового языка доступа к объектам приложений. Платформа Mozilla (XUL/Gecko) использует JavaScript.

JavaScript в данный момент полностью занимает нишу браузерных языков. Несмотря на то, что по слухам некоторые разработчики браузеров встраивают (или уже встроили) в дополнение к JavaScript-у такой язык как Python, для динамического изменения веб-страниц на стороне клиента, официальной информации по этому вопросу нет [13].

О языке

JavaScript имеет C-подобный синтаксис, но по сравнению с языком C имеет следующие коренные отличия:

– объекты, с возможностью интроспекции и динамического изменения типа через механизм прототипов;

– функции как объекты первого класса;

– автоматическое приведение типов;

– автоматическая сборка мусора;

– анонимные функции.

Семантика языка сходна с семантикой языка Self. JavaScript обладает рядом свойств объектно-ориентированного языка, но благодаря прототипированию поддержка объектов в нём отличается от традиционных ОО

языков. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам – функции как объекты первого уровня, объекты как списки, карринг (currying), анонимные функции, замыкания (closures) – что придает языку дополнительную гибкость [14].

Использование в HTML

На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на Java, но при этом простым для использования непрограммистами. Языком JavaScript какая-либо компания или организация не владеет, что его от языков программирования ряда отличает, используемых в веб-разработке.

Встраиваемый язык JavaScript используется как, обычно для доступа к объектам приложений программного. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.
Основные архитектурные черты: слабая типизация, динамическая типизация, управление памятью автоматическое, функции как объекты первого класса, прототипное программирование.

JavaScript – язык программирования прототипно-ориентированный. Является диалектом языка ECMAScript.

Когда интернет браузер читает HTML документ, и замечает тег script, он в первую очередь считывает и выполняет код, а только потом продолжает читать страницу дальше.

Так, в примере приведенном ниже будет рассмотрено начало страницы, далее три раза подряд выполнится оператор alert, который выводит диалоговое окно с данными о количестве кроликов, и лишь после появится вся остальная страница [15].

```
<body>
     <h1>Считаем кроликов</h1>
     <script type=«text/javascript»>
     for(var i=1; i<=3; i++) {
     alert(«Из шляпы достали "+i+" кролика!")
     }
     </script>
     <h1>...Посчитали</h1>
</body>
```

Практикой лучшей является отделение javascript от разметки документа. Для этого его помещают внутрь тега head, а в теле страницы по возможности оставляется

2 Расчетная часть

2.1 Логическое проектирование

Цель логического проектирования заключается в создании реляционных табличных структур на языке DDL.

Создание Базы данных:

```
CREATE DATABASE [msql] ON PRIMARY
(NAME = N'msql', FILENAME = N'C:\Program Files\Microsoft SQL
Server\MSSQL10_50.SQL2008\MSSQL\DATA\msql.mdf', SIZE = 94MB,
MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB)
LOG ON
(NAME = N'msql_log', FILENAME = N'C:\Program Files\Microsoft
SQL Server\MSSQL10_50.SQL2008\MSSQL\DATA\msql_log.ldf', SIZE
= 94MB, MAXSIZE = 2048GB, FILEGROWTH = 10%)
GO
```

Ниже представлен код создания нескольких таблиц:

```
CREATE TABLE [dbo].[ACHIVEMENTS](
     [A_ID] [int] IDENTITY(1,1) NOT NULL PRIMARY KEY,
     [A_TITLE] [varchar](255) NOT NULL,
     [A_IMG] [varchar](255) NULL,
     [A_POINTS] [int] NULL)

CREATE TABLE [dbo].[users](
     [id] [int] IDENTITY(1,1) NOT NULL PRIMARY KEY,
     [username] [varchar](20) NOT NULL,
     [password] [varchar](128) NOT NULL,
     [email] [varchar](128) NOT NULL,
     [activkey] [varchar](128) NOT NULL,
     [create_at] [datetime] NOT NULL,
     [lastvisit_at] [datetime] NOT NULL,
     [superuser] [int] NOT NULL,
     [status] [int] NOT NULL,
     [User_img] [varchar](255) NULL)

CREATE TABLE [dbo].[GETACHIVE](
     [GA_ID] [int] IDENTITY(1,1) NOT NULL PRIMARY KEY,
     [user_id] [int] NOT NULL REFERENCES users (ID),
     [A_ID] [int] NOT NULL REFERENCES ACHIVE (A_ID),
     [ADATE] [datetime] NULL,
```

2.2 Физическое проектирование

2.2.1 Обоснование выбора СУБД

Для разработки базы данных для бэкэнда приложения была выбрана СУБД SQL SERVER 2008 R2. Она помогает существенно снизить затраты на поддержку и повысить качество работы с данными за счет мощной интерпретации языка SQL, названного TRANSACT SQL, а такое приложение как SQL Server Management Studio, позволяет облегчить настройку и поддержку БД системы. Система является, безопасной, надежной, а также простой в управлении. Идеально соответствует выполнению нужных нам задач, связанных с разработкой и поддержкой базы данных, в том числе для web-приложений, а также других систем с использованием крупных объемов данных.

В результате выполнения кода, получим настоящую базу данных. Диаграмма физической модели данной базы данных представлена на рисунке 2.1.

Версия SQL Server, должна была заменить которая SQL Server 2005, кодовое имя Katmai получила. В период активной разработки Microsoft неохотно делилась информацией крайне о новой версии. На SQL Server 2005 презентации Пол Флесснер (на тот момент занимавший пост вице-президента подразделения Microsoft, занимавшегося разработкой SQL Server), уверенно заявил, что релиз новейшей версии состоится не позднее, чем через два года после выхода SQL Server 2005. Тем не менее, в австрийском блоге на TechNet была опубликована информация о программе Katmai Technology Adoption Program (сокр. TAP), начало которой было якобы запланировано на июнь 2007 года. Однако в апреле 2007 года ещё не было никакой информации о скором выходе продукта, или хотя бы о начале его бета-тестирования. Также были упомянуты слухи о том, что новая версия выйдет в 2008 году, но Microsoft на тот момент ни подтверждала, ни отрицала эту информацию. Некоторые источники привязывали выход Katmai к выходу Longhorn Server and Visual Studio Orcas, из-за чего согласно этой информации новая версия должна была выйти в первой половине 2008 года. Microsoft также отказывалась комментировать эту информацию.

2.2.2 Меры по обеспечению безопасности

Одной из наиболее важных частей в БД является разработка прав доступа к ней, т.к. нужна защита от несанкционированного доступа и защита от доступа. Для защиты от сбоев разрабатывается стратегия резервного копирования. В моем случае, из-за специфики web-приложений ограничения доступа к информации осуществляется через само приложение

Рисунок 2.1 – Диаграмма физической модели данной базы данных

2.2.3 Создание пользователей, логинов и задание им паролей

Создание пользователя admin:

```
CREATE USER admin FOR LOGIN admin
WITH DEFAULT_SCHEMA = dbo;
```

Создание роли operator:

```
create role weboperator;
```

Присвоение пользователю oper роль operator:

```
grant create session to weboperator;
grant alter session to weboperator;
grant create table to weboperator;
grant create synonym to weboperator;
grant create view to weboperator;
grant create sequence to weboperator;
grant create trigger to weboperator;
grant create procedure to weboperator;
grant create user to weboperator;
grant drop user to weboperator;
grant create role to weboperator;
grant alter user to weboperator;
grant weboperator to admin;
```

2.3 Описание интерфейса приложения «интерактивная обучающая система для куса «Проектирование баз данных»» и его функциональности

Интерактивная обучающая система – это такая система, которая специально создается для обучения пользователя чему-либо, при этом пользователь не просто читает нужную ему информацию, но и непосредственно взаимодействует с программой – выполняет различные задания. Такой обмен информацией между программой и пользователем называется интерактивностью. Таким образом, интерактивность - это способность компьютерной системы адекватно отвечать на действия пользователя. Вследствие этого, система должна обладать неким примитивным подобием интеллекта. Интерактивное взаимодействие относится к одному из научных направлений человеко-компьютерного взаимодействия (HCI). Поэтому, в данном проекте стояла задача изучить и разработать различные взаимодействия между людьми и компьютерами.

HCI опирается в равной степени на человеческий фактор и на компьютерный фактор. Понять компьютерную сторону можно с помощью различных технологий программирования, компьютерной графики, а человеческую – с помощью психологии, социологии и некоторых разделов теории коммуникации.

В интерактивных обучающих системах лучше всего использовать стандартные пути и решения, так как целью таких систем является не введение новшеств, чтобы удивить пользователя, а дать ему наиболее полное представление о материале, который он желает изучить. Психологи доказали, что в таких случаях человек интуитивно доверяется системе с более привычными механизмами навигации и с более простым и понятным интерфейсом, так как ничто не будет отвлекать его от обучения — всё расположено стандартно, удобно и ненавязчиво.

Программа является системой, если компоненты в ней взаимосвязаны друг с другом, при этом каждый элемент выполняет свои функции. Поэтому любая обучающая программа является системой. Если проверка правильного ответа выполняется исключительно программными средствами, то данная система будет автоматизированной обучающей системой. Разрабатываемая в данной дипломной работе программа не является в полной мере автоматизированной, так как проверка заданий осуществляется преподавателем.

Данная система относится к типу разомкнутых обучающих систем, так как в ней выполняется заранее определенная разработчиком последовательность изложения разделов и задач.

Работа с приложением

Данное web-приложение разработано с учетом интуитивно понятного интерфейса типа flat UI. Код приложения представлен в приложении А.

С помощью моего приложения студентам будет проще изучать базы данных, а учителям проще взаимодействовать со студентами. На рисунке 2.2 входная страница сайта, на ней расположена форма для авторизации в системе.

Flat UI называется такой тип интерфейса, в котором используются исключительно простые формы и мягкие цвета. Это позволяет снизить нагрузку на глаза пользователя и акцентировать внимание тех, кто использует систему, именно, на ее содержимом. Противоположность данному методу можно назвать скеморфизм, который господствовал в интернете до начала 2012 года. Окончательная победа Flat UI состоялась с анонсом iOS 7 от компании apple.

Для любой категории пользователей, работа в системе начинается с авторизации или регистрации. В системе предусмотрено разделение пользователей на 3 группы по уровню доступа: студент, учитель, администратор.

После авторизации все пользователи попадают на страницу своего профиля. На странице указаны данные о количестве выполненных заданий, набранный опыт, личные данные. Также на этой странице отображается структура всего курса, какие задания выполнены, какие с ошибками. Здесь же информация о тестах, количество набранных на них процентов.

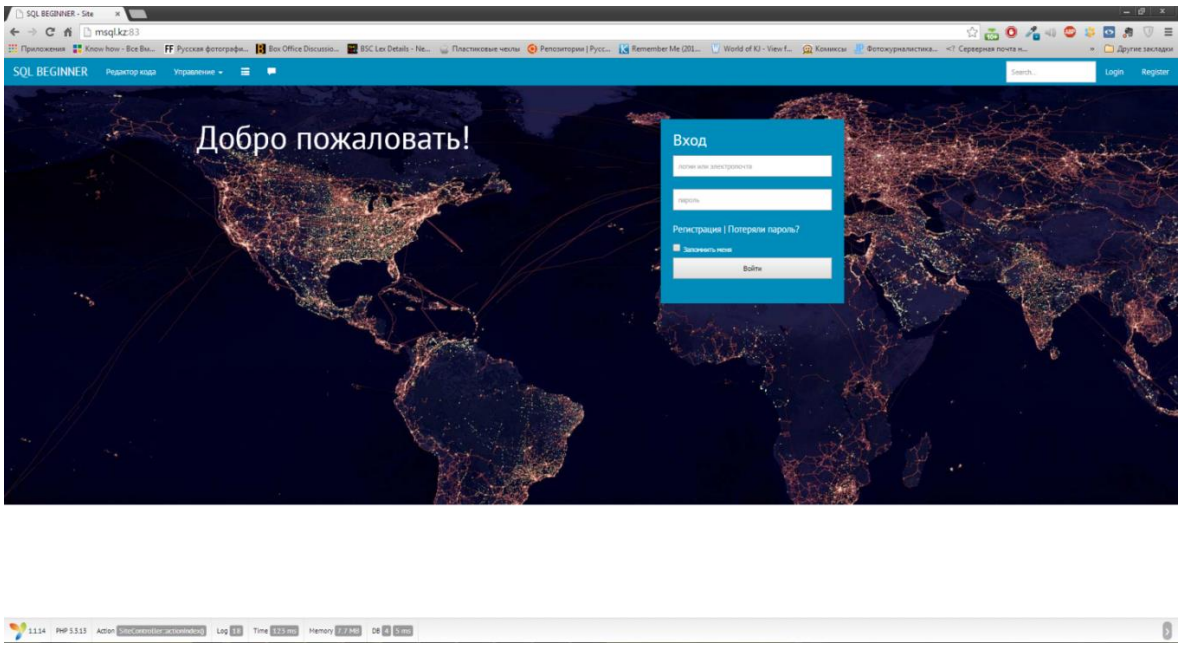


Рисунок 2.2 – Форма входа на сайт

Пользователи могут загрузить себе аватар, указать из кого города они родом.

Также на странице расположена панель прогресса, на которой, в виде цветных блоков показывается сколько заданий выполнено верно, сколько с ошибками, и сколько еще предстоит сделать для получения рейтинга в 100 процентов. Ниже, расположена панель достижений. Достижения служат для дополнительной мотивации студентов, добавляют соревновательный стимул к процессу изучения курса (Рисунок 2.3).

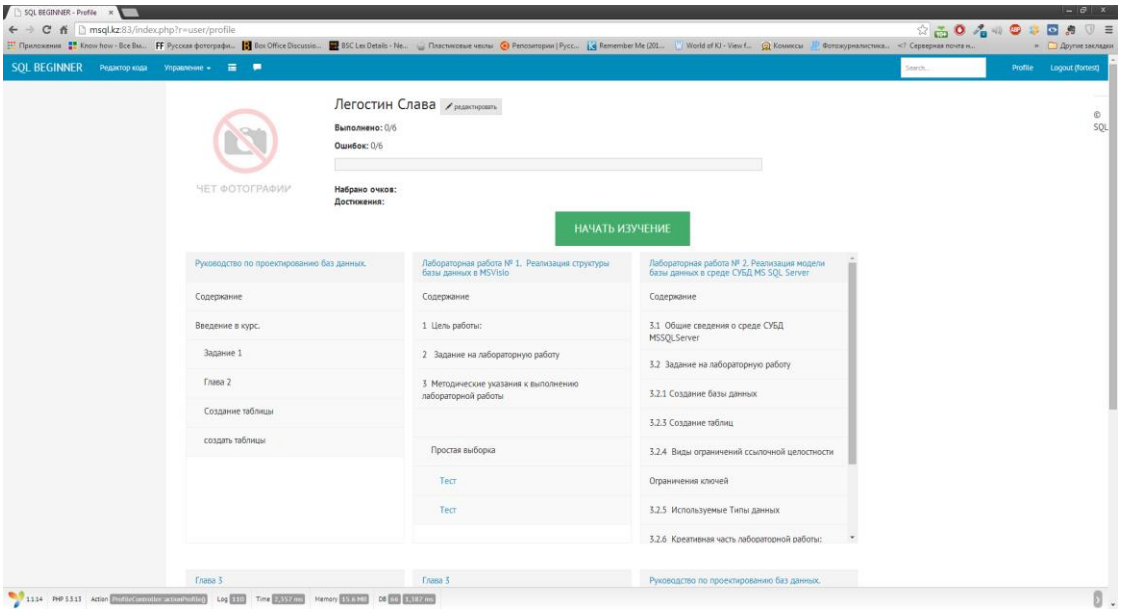


Рисунок 2.3 – Страница профиля только зарегистрировавшегося пользователя.

Пользователи могут отправлять личные сообщения другим пользователям с помощью встроенного мессенджера (рисунок 2.4).

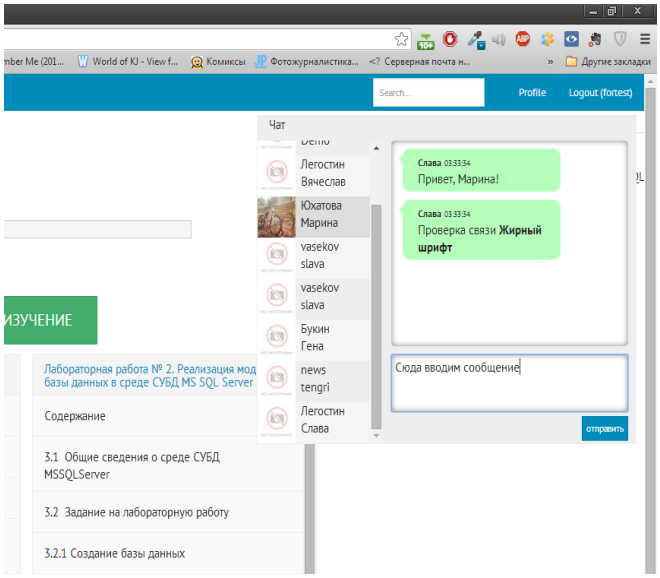


Рисунок 2.4 – Встроенная система обмена мгновенными сообщениями

Помимо личных сообщений, учителя и администраторы могу отправлять общие для всех уведомления (рисунок 2.5).

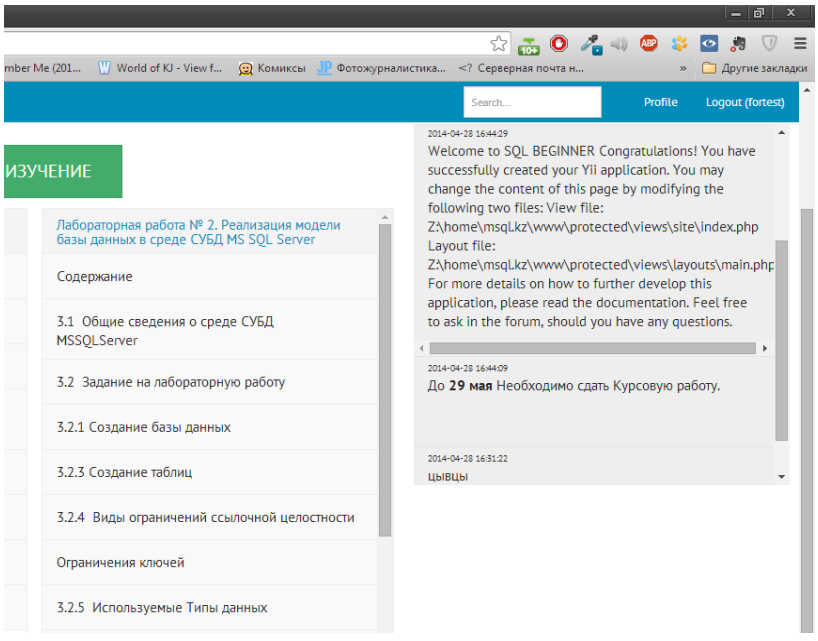


Рисунок 2.5 – Панель уведомлений

На странице профиля можно изменить свои личные данные (рисунок 2.6).

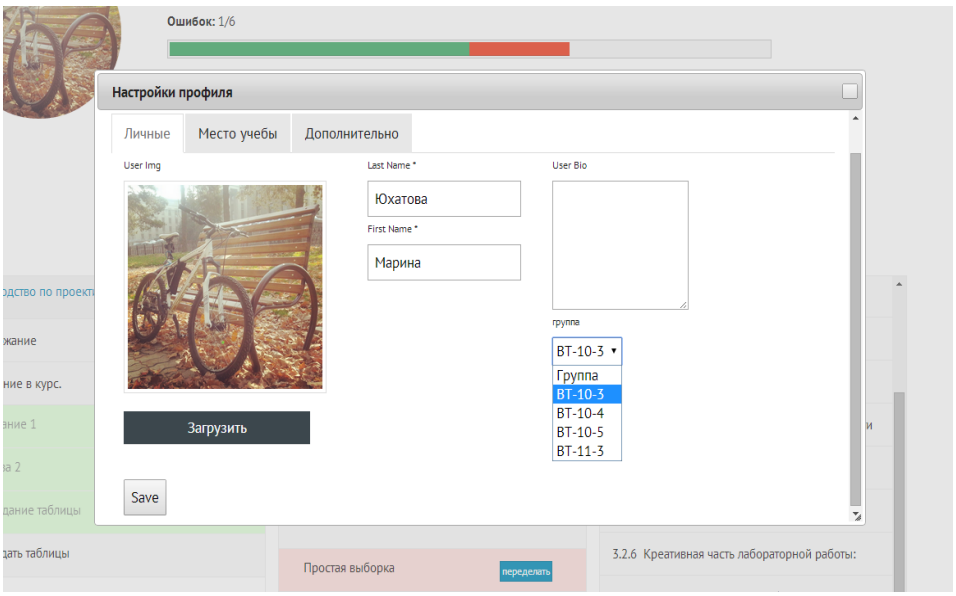


Рисунок 2.6 – Форма изменения личных данных

Закончив с настройками, можно приступать к изучению материалов. На рисунке 2.7 показана страница одной из глав курса. Для удобства, мною был написан парсер HTML, который автоматически генерирует содержание глав и выводит его в меню слева. Сами главы, помимо текста, могут содержать изображения, видеоматериалы, таблицы и ссылки.

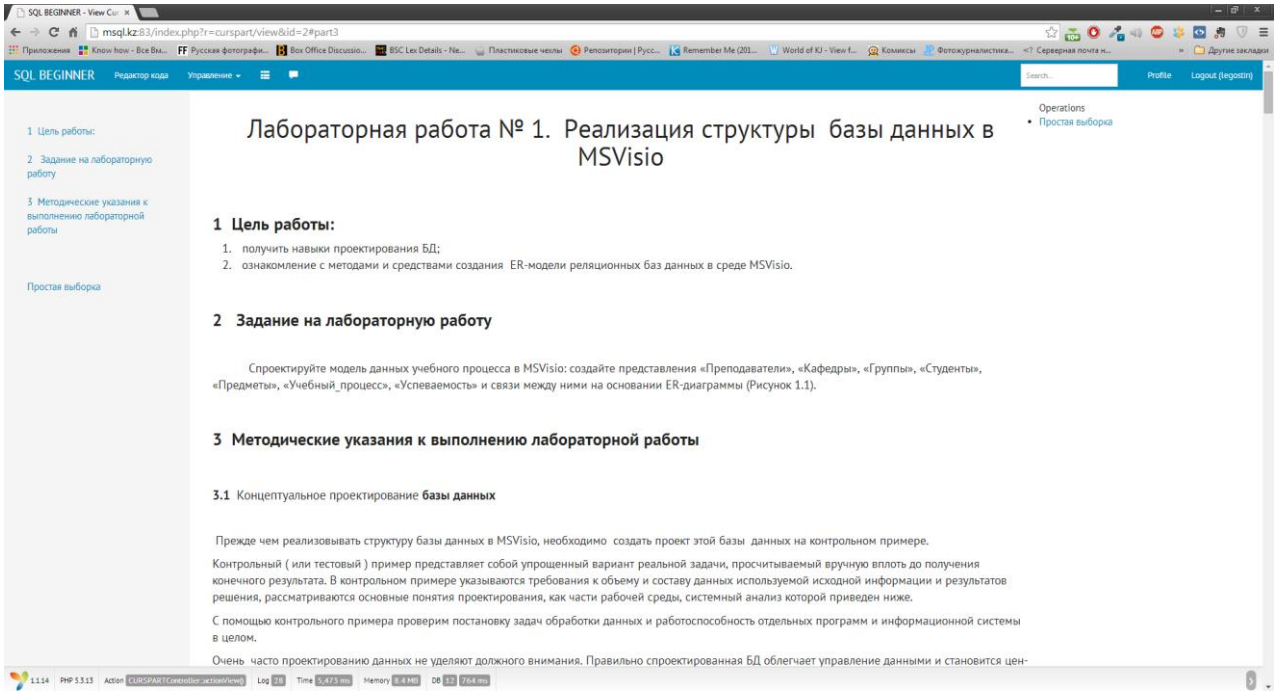


Рисунок 2.7 – Страница просмотра главы курса

После прочтения главы, студент приступает к выполнению заданий, список заданий располагается ниже оглавления на страницах глав курса.

При выполнении задания, его текст выводится в колонку слева, в центральной части окна располагается редактор SQL кода. Реализована динамическая подсветка кода (рисунок 2.8)

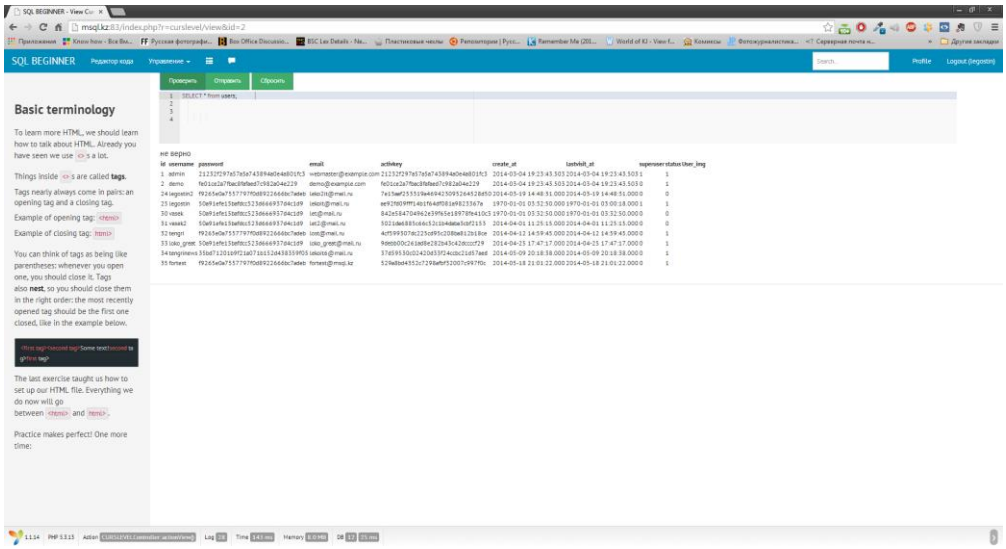


Рисунок 2.8 – Страница, на которой выполняется задание

Выполнив задание, студент отправляет его в очередь на проверку учителем (рисунок 2.9). Учитель может как принять решение, так и отклонить его, сделав замечания или попросив дополнить решение. Учитель может фильтровать очередь по статусу заявок с помощью кнопок в левой колонке. Все действия выполняются без перезагрузок страницы, благодаря использованию AJAX запросов. Все действия снабжены красивой, нераздражающей анимацией. Вместе, эти факторы позволяют придать системе монолитный вид, показать, что она надежна и современна.

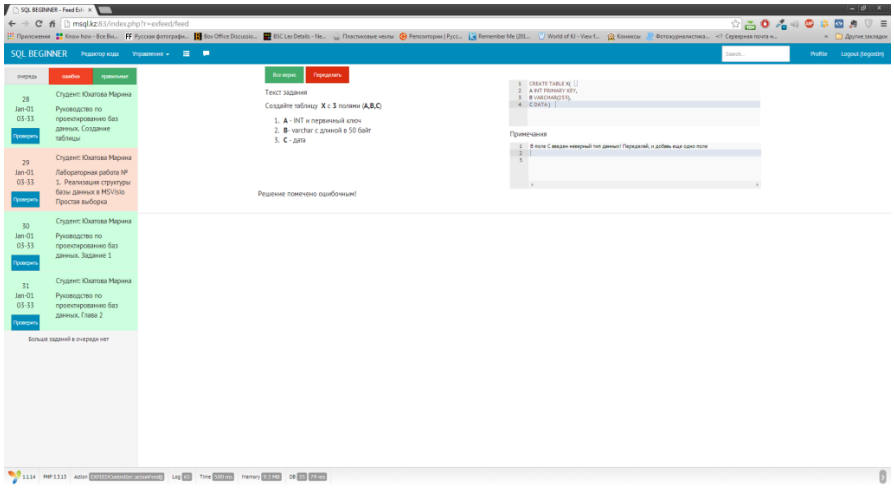


Рисунок 2.9 – Очередь заданий для проверки

После проверки, на странице профиля отображается статус задания (рисунок 2.10). Зеленым выделяется, если задание принято, красным, если необходимо внести исправления. Нажав кнопку переделать, можно будет сразу-же ознакомиться с ошибками и внести исправления. Данный блок позволяет студентам лучше следить за своей успеваемостью.

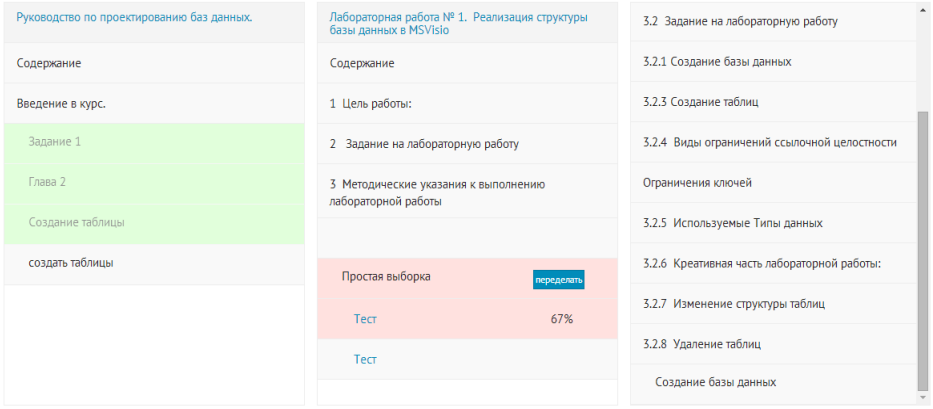


Рисунок 2.10 – Блоки глав курса, с отображением на них статуса заданий и тестов

Также после каждой главы нужно пройти тестирование (рисунок 2.11). Как видно на рисунке, тестовые задания и варианты ответов могут включать в себя не только текст, но и различные мультимедиа элементы, вплоть до видеороликов.

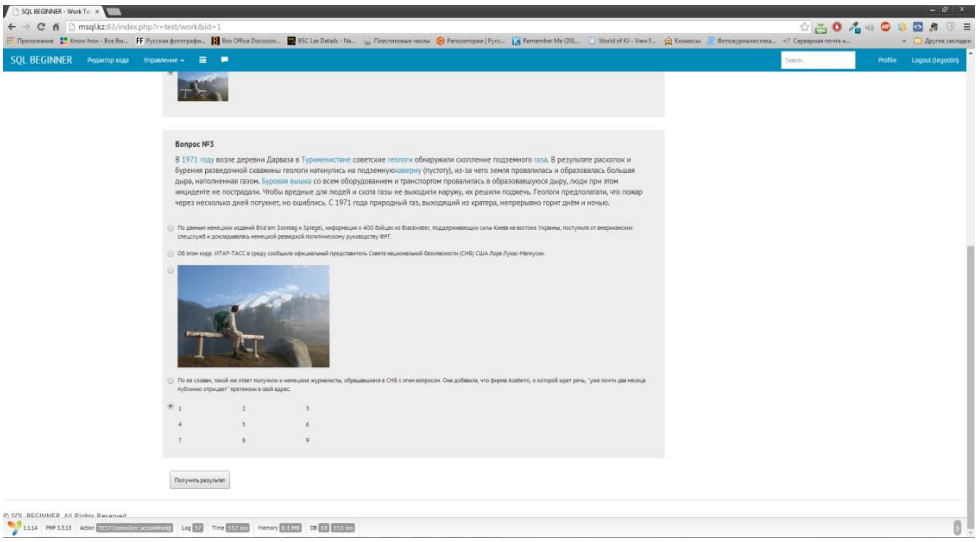


Рисунок 2.11 – Страница тестирования

После прохождения тестирования можно посмотреть свои результаты, а также какие ошибки были сделаны (рисунок 2.12). Показ блока с неверно выполненными заданиями позволяет студентам обратить внимание на те моменты программы, в которых они разобрались еще не до конца.

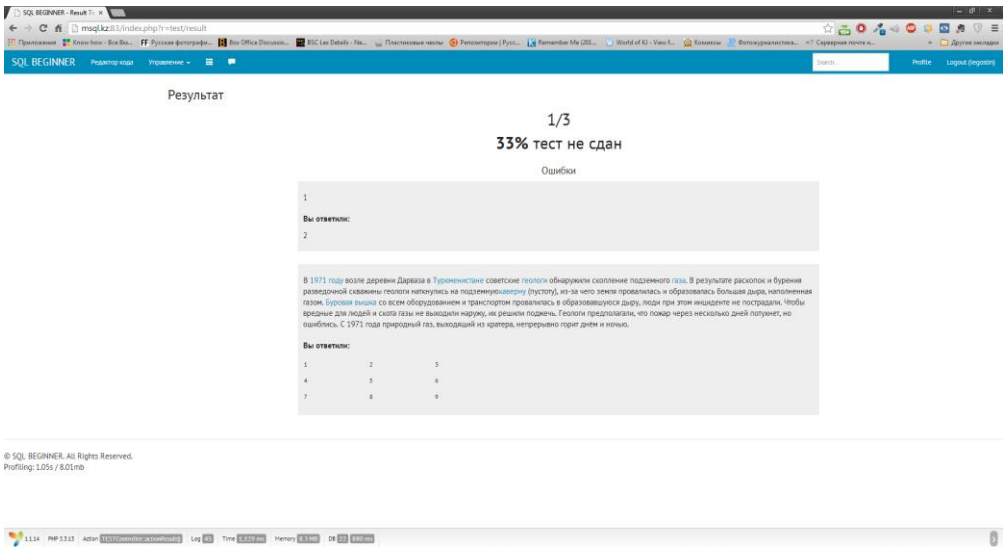


Рисунок 2.12 – Страница с результатами тестирования

Учитель может изменять и добавлять новые элементы курса. Так, на рисунке 2.13 показан интерфейс формы изменения одной из глав курса. Справа можно увидеть блок с информацией о том, как правильно выделять заголовки в тексте, чтобы прсер автоматически мог построить содержание текста. Редактор текста обладает всеми необходимыми функциями для форматирования текста и мультимедиа материалов. Есть возможность вставки изображений, видеороликов, таблиц. Видеоролики могут находиться как на самом сервере, так и на внешних видеохостингах, скрипт редактора автоматически сгенерирует код для проигрывателя, учителю достаточно указать адрес файла, все остальное система выполнит автоматически. После сохранения документа, он записывается в базу данных в виде html разметки, с прописанными в нем css правилами.

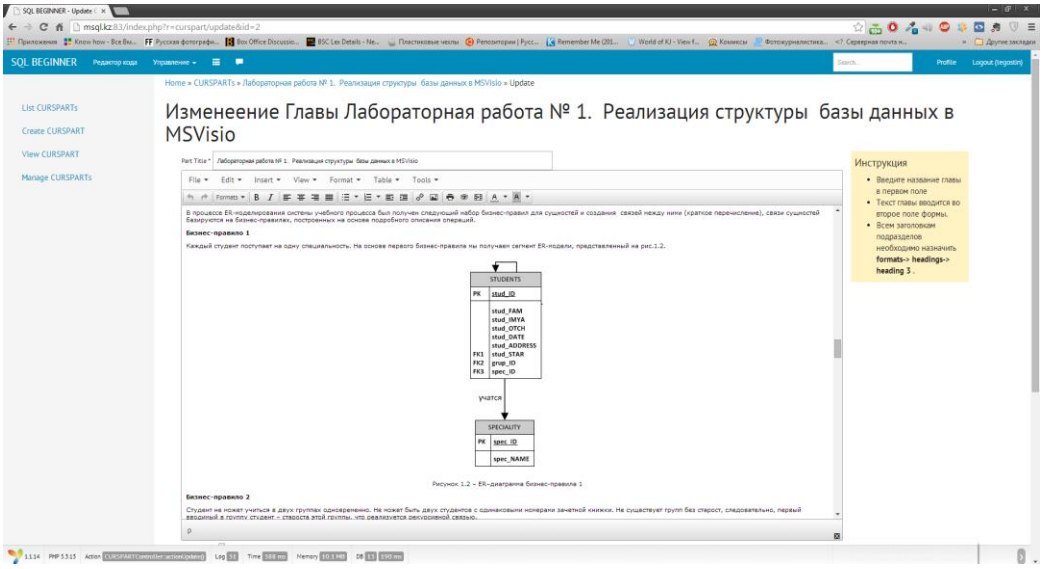


Рисунок 2.13 – Форма создания и изменения материалов главы курса

Также учитель может добавлять новые тестовые вопросы в базу курса. Для каждого вопроса необходимо задать 5 вариантов ответа, один из которых является правильным. Сами вопросы и варианты ответов к ним, также, могут содержать всевозможные мультимедийные материалы: начиная от картинок и видеороликов, заканчивая встроенными картами, диаграммами и схемами. Фактически, сюда можно встроить любой html5 блок. Все это позволяет отказаться от каких-либо ограничений при составлении вопросов и вариантов ответов к ним. Верстка страницы прохождения теста прекрасно отобразит любые элементы (рисунок 2.14).

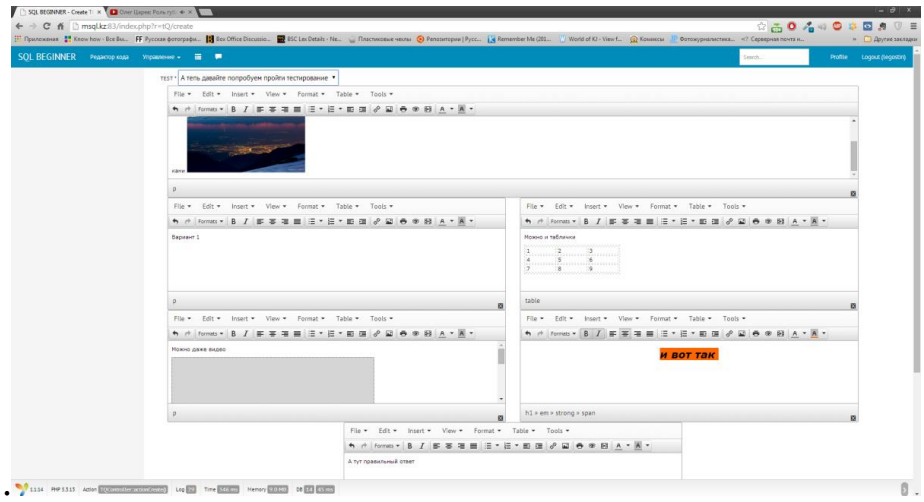


Рисунок 2.14 – Форма ввода тестового задания

Помимо всего прочего, в систему встроен крайне мощный файловый менеджер (рисунок 2.15), доступный только учителям и администраторам. С помощью данного модуля можно загружать необходимые мультимедийные материалы на сервер. В модуль встроена система директорий и прав доступа, пользователи могут заниматься переименованием документов, их поиском.

.

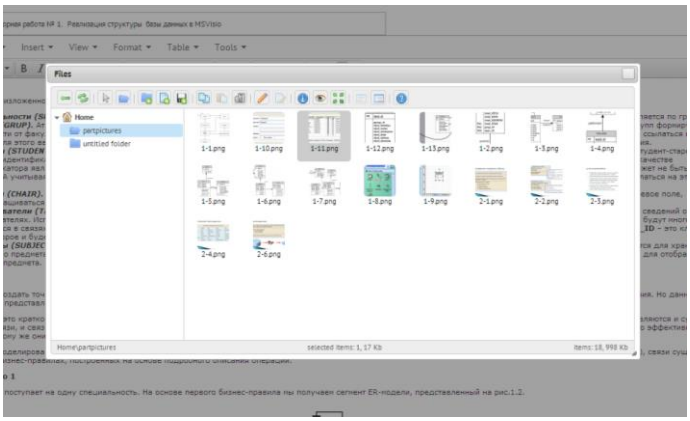


Рисунок 2.15 – Окно выбора и загрузки мультимедиа материалов для курса

Прямо из программы можно изменять размер изображений, создавать их копии. Просмотр списка документов осуществляется в виде таблицы или в режиме плиток. Также имеется встроенный модуль для просмотра изображений. Все это позволяет создавать максимально красивые и интересные материалы для интерактивной обучающей системы.

На рисунке 2.16 показаны примеры того, какие мультимедиа материалы можно добавить к курсу с помощью инструментов, о которых я говорил раньше.

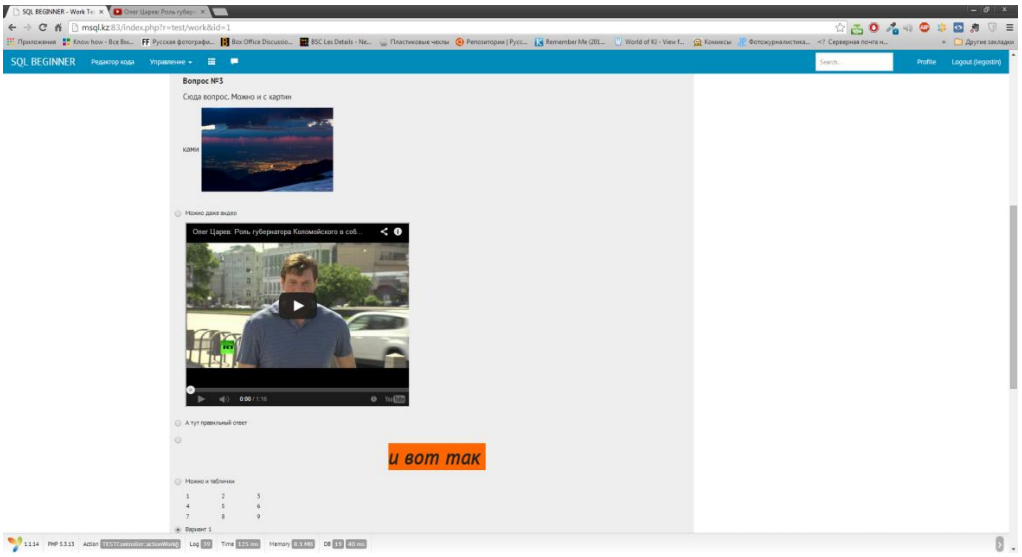


Рисунок 2.16 – Мультимедийные материалы в тесте

На рисунке 2.17 показан пример обращения пользователя к учебной базе, как вы можно увидеть, программа выполнила запрос к базе и вернула студенту таблица с данными в ней.

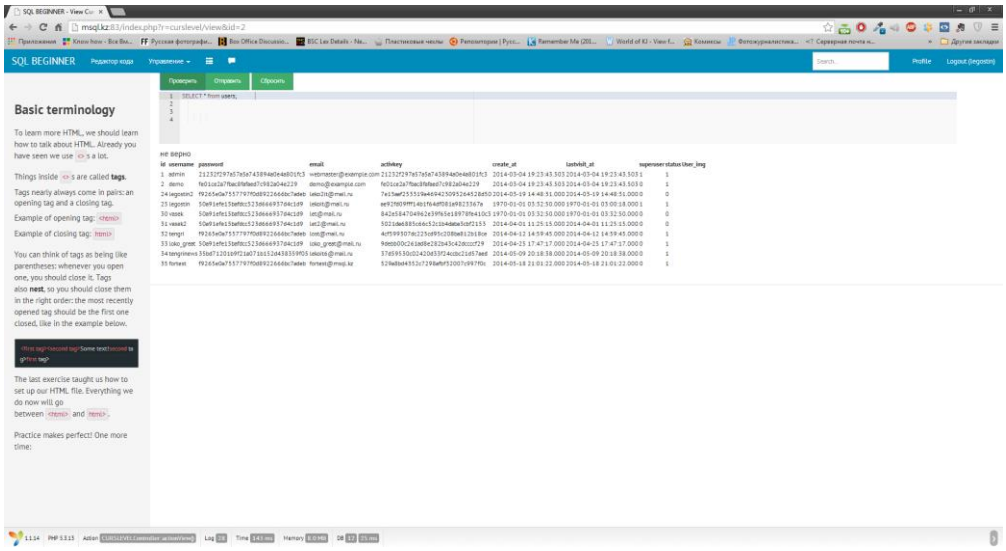


Рисунок 2.17 – Запрос студента к базе

3 Технико-экономическое обоснование проекта

3.1 Описание работы и обоснование необходимости

Тема данного дипломного проекта – «Разработка интерактивной обучающей системы для курса «Проектирование баз данных».

Цель данного проекта – разработать обучающую систему, которая упростила бы процесс изучения баз данных в университете.

Так как этот предмет является одним из самых сложных в курсе программ ВТиПО III курса, разработка данной системы создаст принципиально новый подход к обучению, упростит контроль преподавателя над успеваемостью студентов, позволит более доступным и интересным языком изложить сложный материал.

В данном разделе приводится рассмотрение экономической составляющей реализации данного продукта, отражающее временные, трудовые и финансовые затраты на проект.

3.2 Трудовые ресурсы, используемые в работе

В работе задействованы:

– руководитель – постановка задачи, разработка основных бизнес правил для работы проекта;

– инженер-разработчик – разработка сайта, проектирование базы данных;

– консультанты по экономической части и по части ОБЖД.

Общее количество сотрудников и их заработная плата представлены в таблице 3.1

Таблица 3.1 – Количество задействованных в проекте работников, и их заработная плата

Исполнитель	Количество, человек	Месячная заработная плата, тенге
Руководитель	1	100000
Консультант по части «Экономика»	1	70000
Консультант по части «Безопасность жизнедеятельности»	1	75000
Инженер-разработчик	1	150000
Итого	4	395000

3.3 Расчет стоимости работы по проектированию и разработке

Разработка многомодульного программного продукта – сложный и трудоемкий процесс, требующий наряду с интеллектуальными, техническими затратами и финансовых затрат [16]. Поэтому необходимым является произведение расчета стоимости этой разработки. Затраты на разработку данного программного комплекса определяется по формуле:

$$C = ФОТ + C_H + A + Э + C_{пр} + H \quad (3.1)$$

где ФОТ – фонд оплаты труда;

C_H – социальный налог;

А – амортизационные отчисления;

Э – затраты на электроэнергию;

$C_{пр}$ – прочие расходы;

Н – накладные расходы.

Необходимый фонд оплаты труда рассчитывается по формуле:

$$ФОТ = З_{ОСН} + З_{ДОП} \quad (3.2)$$

где $З_{ОСН}$ – основная заработная плата;

$З_{ДОП}$ – дополнительная заработная плата.

Труд сотрудников университета, задействованных в работе, оплачивается согласно положению о заработной плате АУЭС, труд программиста-разработчика принят условно, на договорной основе в размере 100000 тенге.

Базовый показатель для определения составляющих затрат труда вычисляется по формуле:

$$Q = q * c \quad (3.3)$$

где Q – условное число команд,

q=5400 – коэффициент, учитывающий условное число команд в зависимости от типа задачи,

c=1,38 – коэффициент, учитывающий новизну и сложность программы.

Q =5400*1.38=7452.

Выбрать значение коэффициента q можно из таблицы 3.2.

Т а б л и ц а 3 . 2 – Значения коэффициента q

Тип задачи	Пределы изменений коэффициента
Задачи учета	От 1400 до 1500

Окончание таблицы 3.2

Тип задачи	Пределы изменений коэффициента
Задачи оперативного управления	От 1500 до 1700
Задачи планирования	от 3000 до 3500
Много вариантные задачи	от 4500 до 5000
Комплексные задачи	от 5000 до 5500

Программные продукты по степени новизны может быть отнесены к одной из 4-х групп:

– группа А – разработка принципиально новых задач;

– группа Б – разработка оригинальных программ;

– группа В – разработка программ с использованием типовых решений;

– группа Г – разовая типовая задача.

Коэффициент *c* определяется из таблицы 3.3, на пересечении групп сложности и степени новизны.

Т а б л и ц а 3 . 3 – Коэффициенты расчета трудоемкости

Язык программирования	Группа сложности	Степень новизны				Коэффициент *B*
		А	Б	В	Г	
Высокого уровня	1	1,38	1,26	1,15	0,69	1,2
	2	1,30	1,19	1,08	0,65	1,35
	3	1,20	1,10	1,00	0,60	1,5
Низкого уровня	1	1,58	1,45	1,32	0,79	1,2
	2	1,49	1,37	1,24	0,74	1,35
	3	1,38	1,26	1,15	0,69	1,5

Далее необходимо рассчитать время на создание программного продукта.

При разработке многомодульного программного продукта используется техническое задание, согласно которому выполнение работ происходить последовательно по заданным пунктам, с учетом сроков их исполнения. График выполнения работ разработки системы представлен в таблице 3.4

Т а б л и ц а 3 . 4 – График выполненных работ по разработке проекта.

Код работы	Наименование работы	Ожидаемая длительность, дни	Обозначение
1	Подготовка описания задачи	4	$T_{ПЗ}$
2	Описание задачи	3	$T_{ОЗ}$
3	Разработка алгоритма	6	$T_{А}$
4	Установка и запуск локального сервера	2	$T_{ЛС}$

Окончание таблицы 3.4

Код работы	Наименование работы	Ожидаемая длительность, дни	Обозначение
5	Проектирование и создание базы данных	9	$T_{БД}$
6	Разработка модулей приложения	6	$T_{М}$
7	Разработка и создание интерфейса системы	7	$T_{ИН}$
8	Разработка основной части программы	6	$T_{ОСН}$
9	Тестирование базы данных и основных модулей с целью выявления ошибок	2	$T_{ТЕСТ}$
10	Составление технической документации	3	$T_{ТЕХ}$
11	Подготовка раздела «Экономика»	7	$T_{Э}$
12	Подготовка раздела «Безопасность жизнедеятельности»	7	$T_{БЖ}$

Так как участники, задействованные в проекте, работают в различные промежутки времени, в течение которого реализуется проект, необходимо произвести расчет дневной и почасовой оплаты труда.

Заработная плата каждого работника за один рабочий день рассчитывается по формуле

$$D = \frac{O}{n} \tag{3.3}$$

где О – оклад работника в тенге;

n – количество дней в рабочем месяце(это 24 дня – шестидневная рабочая неделя).

Для руководителя

D= $\frac{100000}{24} = 4167$ тенге/день.

Для консультанта по части «Экономика»

D= $\frac{70000}{24} = 2917$ тенге/день.

Для консультанта по части «БЖД»

D= $\frac{75000}{24} = 3125$ тенге/день.

Для инженера разработчика

D= $\frac{150000}{24} = 6250$ тенге/день.

Заработная плата за один час рассчитывается по формуле

$$H = \frac{D}{z} \tag{3.4}$$

где D – заработная плата работника за один рабочий день,
z – количество часов рабочего дня (8 часов).

Для руководителя

$H = \frac{4167}{8} = 520{,}8$ тенге/час.

Для консультанта по части «Экономика»

$H = \frac{2917}{8} = 364{,}6$ тенге/час.

Для консультанта по части «БЖД»

$H = \frac{3125}{8} = 390{,}6$ тенге/час.

Для инженера разработчика

$H = \frac{6250}{8} = 781{,}2$ тенге/час.

Время рассчитывается в человеко-часах, причем $T_{ПЗ}$ берется по фактически отработанному времени, а время остальных этапов определяется расчётно, по условному числу команд Q.

Определяем время, затраченное на каждый этап создания программного продукта:

1 $T_{ПЗ}$ (время на подготовку описания задачи), берется по факту и составляет (от 3-х до 5-ти дней по 8 часов):

$T_{ПЗ} = 32$ чел/час

2 $T_{ОЗ}$ (время на описание задачи) определяется по формуле

$$T_{ОЗ} = \frac{Q * B}{50 * K} \quad (3.5)$$

где B=1,2 – коэффициент учета изменений задачи, коэффициент B в зависимости от сложности задачи и числа изменений выбирается в интервале от 1,2 до 1,5 (таблица 3.5)

К=1,3 – коэффициент, учитывающий квалификацию программиста.

$Тоз = \frac{7452*1,2}{50*1,3} = 137,6$ чел/час.

Таблица 3.5 – Коэффициенты квалификации программиста

Опыт работы	Коэффициент квалификации
До двух лет	0.8
2-3 года	1
3-5 лет	1.1 – 1.2
5-7 лет	1.3 – 1.4
более 7 лет	1.5 – 1.6

3 Та (время на разработку алгоритма) рассчитываем по формуле

$$T_A = \frac{Q}{50 * K} \quad (3.6)$$

$TA = \frac{7452}{50*1,3} = 114,6$ чел/час.

4 $T_{ЛС}$ (время на установку и запуск локального сервера) определяется аналогично T_A

$T_{ЛС}$ = 114,6 чел/час

5 $T_{БД}$ (время на проектирование и создание БД) определяется по формуле

$ТБД = \frac{3500*1,2}{50*1,3} = 64,6$ чел/час.

6 T_M (время на разработку модулей приложения) определяется по формуле

$ТМ = \frac{3500*1,2}{50*1,3} = 64,6$ чел/час.

7 $T_{ИН}$ (время отладки и тестирования программы) определяется по формуле

$ТИН = \frac{7452*1,2}{50*1,3} = 137,6$ чел/час.

8 $T_{ОСН}$ (время на разработку основной части), определяется по формуле

$ТОСН = \frac{4500*1,2}{50*1,3} = 83$ чел/час.

9 $T_{ТЕСТ}$ (время на тестирование), определяется по формуле

$ТТЕСТ = \frac{3000*1,2}{50*1,3} = 55,4$ чел/час.

10 $T_{ТЕХ}$ (время на составление технической документации), берется по факту и составляет (от 3-х до 5-ти дней по 8 часов)

$T_{ТЕХ} = 24$ чел/час.

11 $T_Э, T_{БЖ}$ (время на подготовку разделов Экономика и БЖ), берется по факту и составляет (от 7-и до 10-ти дней по 8 часов)

$T_Э = T_{БЖ} = 56$ чел/час.

12 т.к. в проекте задействованы руководители, заработная плата которых отличается, то необходимо рассчитать время $T_{РУК}$. Время берется среднее и оно равно

$T_Р = 70$ чел/час,
$T_Э = T_{БЖ} = 30$ чел/час.

Суммарные затраты труда рассчитываются как сумма составных затрат труда по формуле 3.10

$$T_{СТ} = T_{ПЗ} + T_{ОЗ} + T_{А} + T_{ЛС} + T_{БД} + T_{М} + T_{ИН} + T_{ОСН} + T_{ТЕСТ} + T_{ТЕХ} + T_{Э} + T_{БЖ} \quad (3.7)$$

$T_{СТ}$ =32+137,6+114,6+114,6+64,6+64,6+137,6+83+55,4+24+56+56
=940 чел/час.

Таким образом, суммарная основная заработная плата составит

$$З_{ОСН} = З_{ИНЖ} + З_{РУК} \quad (3.8)$$

$З_{ОСН}$ =940 * 781,2 + 30 * 364,6 + 30 * 390,6 + 70 * 520,8 =793440 тенге.

Дополнительная заработная плата в среднем определяется в размере 10% от основной заработной платы и рассчитывается по формуле

$$З_{ДОП} = З_{ОСН} * 10\% \quad (3.9)$$

$З_{ДОП}$ =793440 * 10% = 79344 тенге.

Общий фонд оплаты труда согласно формуле 3.2:

ФОТ = 793440 + 79344 = 872784 тенге.

Социальный налог составляет 11% от ФОТ и рассчитывается по формуле

$$С_{Н} = (\text{ФОТ} - \text{ПО}) * 11\% \quad (3.10)$$

где ПО (пенсионные отчисления) составляют 10% от ФОТ и рассчитываются по формуле

$$ПО = \text{ФОТ} * 10\% \quad (3.11)$$

Размер пенсионных отчислений согласно формуле 3.11 составляет

ПО = 872784* 10% = 87278 тенге;

Отчисления по социальному налогу согласно формуле 3.10

$С_{Н}$ =(872784 – 87278) * 11% = 86405,6 тенге

3.4 Расчет затрат на амортизацию

Амортизационные отчисления производятся по установленным нормам амортизации, выражаются, в процентах к балансовой стоимости оборудования и рассчитываются по формуле

$$A = \frac{C_{обор} * H_A * \mathrm{N}}{100 * 12 * \mathrm{t}} \quad (3.12)$$

где H_A – норма амортизации;

$C_{обор}$ – первоначальная стоимость оборудования;

N – количество дней на выполнение работ;

t – общее время использования персонального компьютера.

Норма амортизации H_A, рассчитывается по формуле

$$H_A = \frac{C_{ОБОР} - C_{ЛИКВ}}{T_{НОРМ} * C_{ОБОР}} * 100\% \quad (3.13)$$

где $C_{ЛИКВ}$ – ликвидационная стоимость, составляет 5.61% от стоимости оборудования;

$T_{НОРМ}$ – нормативный срок службы (для ПК – 4 года).

Т а б л и ц а 3 . 6 – Оборудование необходимое для разработки ПП:

Наименование	Модель	Стоимость
Ноутбук	ASUS DeluxLight	80000
Компьютерная мышь	HP Z78554	500
Модем	D-Link EliteX.25	2 000
Итого		82500 тг.

Так как ликвидационная стоимость составляет 5.61%, следовательно:

$C_{ЛИКВ}$ = 0,056* $C_{ОБОР,}$
$C_{ЛИКВ}$ =0,056*82500 =4620 тг.

Общее время использования персонального компьютера учитывает лишь время работы на компьютере и рассчитывается по формуле:

$$\mathrm{t} = T_{ЛС} + T_{БД} + T_M + T_{ИН} + T_{ОСН} + T_{ТЕСТ} \quad (3.14)$$

t = 113.6 + 64,6 + 64,6 + 137,6 + 83 + 55,4 = 519,8 чел/час

$$НА = \frac{82500-4620}{3*82500} = 31{,}4\ \%$$

$$А = \frac{82500*31{,}4*940}{100*12*519.8} = 3904 \text{ тенге}$$

Затраты на электроэнергию вычисляется по формуле:

$$Э = М * к_з * Т * С_{кВт \cdot ч} \quad (3.15)$$

где: М – мощность ЭВМ (600 Вт=0,6 кВт);

$к_з$ – коэффициент загрузки (0.8);

$С_{кВт \cdot ч}$ – 14,935 тг– стоимость 1 кВт-час электроэнергии;

Т – время работы.

$Э$ = 0,6*0,8*14,935*519,8 = 3726 тенге

Расходы на материалы и комплектующие, используемые в процессе написания программного продукта ($С_{МиК}$), а также затраты на техническое обслуживание и ремонт ($С_{ТО}$) составляют, соответственно, 2,06% и 2,266% от стоимости оборудования – формулы 3.16 – 3.17

$$С_{мик} = 0,0206 * С_{обор} \text{ тенге}, \quad (3.16)$$

$$С_{то} = 0,02266 * С_{обор} \text{ тенге} \quad (3.17)$$

$С_{МиК}$ = 0,0206 * 82500 = 1699,5 тенге,
$С_{ТО}$ = 0,02266 * 82500= 1869,5 тенге

$$С_{пр} = С_{мик} * С_{то} \text{ тенге} \quad (3.18)$$

$С_{ПР}$ = 1699,5+1869,5 = 3569 тенге

Накладные расходы, связанные с управлением и обслуживанием, содержанием и эксплуатацией оборудования и прочими дополнительными затратами на обеспечение процессов производства и обращения, составляют 50% от всех затрат, вычисляются по формуле [19]:

$$Н = (ФОТ + С_Н + А + Э + С_{пр}) * 0,5 \quad (3.19)$$

$Н$ = (872784+86405,5 + 3904 + 3726 + 3569)*0.5 = 485194 тенге

Таким образом, затраты на разработку данного программного комплекса согласно формуле 3.1 равна:

C =872784+86405,5 + 3904 + 3726 + 3569+485194 = 1455582,5 тенге

Сводные результаты расчета себестоимости программного продукта предоставлены таблице 3.7

Таблица 3.7 – Себестоимость разработки системы

Статья расходов	Сумма, тенге	В процентах от общей суммы,%
ФОТ	872784	60
$С_Н$	86405,5	5,9
А	3904	0,2
Э	3726	0,2
$С_{пр}$	3569	0,2
Н	485194	32,9
Итого:	1455582,5	100

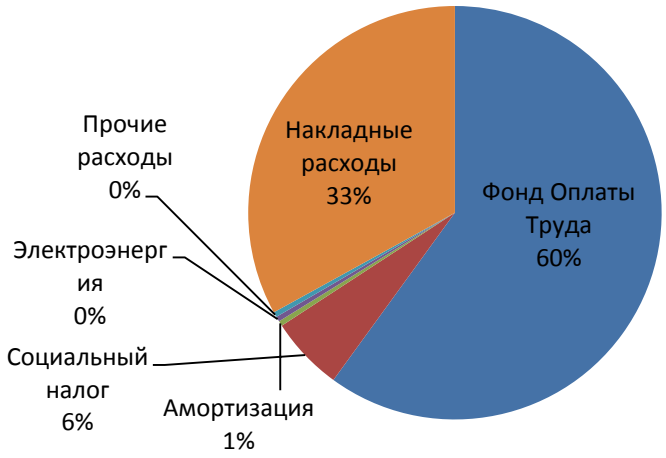


Рис 3.1 – Диаграмма структуры себестоимости программного обеспечения

4 Безопасность жизнедеятельности

4.1 Анализ потенциально опасных и вредных производственных факторов проектируемого объекта, воздействующих на персонал

Источниками повышенной опасности могут служить следующие элементы:

– распределительный щит;

– источники питания;

– блоки ПЭВМ и печати, находящиеся в ремонте.

В соответствии с СНиП 4.02–42–2006 [20] к легкой физической работе относятся все виды деятельности, производимые сидя и не требующие физического напряжения. Работа пользователя разработанного пакета программ относится к категории 1а.

Согласно СНиП 4.02–42–2006 помещение для ЭВМ по степени опасности поражения человека электрическим током относится к помещениям без повышенной опасности (нет токопроводящих полов, сырости, повышенной температуры, возможности одновременного прикосновения к корпусам оборудования с «землей» и к токонесущим частям).

В соответствии с СНиП 4.02–42–2006 при обслуживании ЭВМ персонал может подвергаться воздействию потенциально опасных физических и психофизиологических опасных и вредных производственных факторов:

– повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека;

В соответствии с СНиП 4.02–42–2006 на рабочем месте оператора допускаются следующие уровни напряжений прикосновения и токов, значения которых приведены в таблице 4.1.

Таблица 4.1 – Предельно допустимые значения напряжений прикосновения и токов

Род тока	U (не более), В	I, мА
Переменный, 50 Гц	2.0	0.3
Постоянный	8.0	1.0

– повышенный уровень статического электричества;

Электризация материалов часто препятствует нормальному ходу технологических процессов производства, а также создает дополнительную пожарную опасность вследствие искрообразования при разрядах при наличии в помещениях, резервуарах и ангарах горючих паро- и газо-воздушных смесей.

В ряде случаев статическая электризация тела человека и затем последующий разряд с человека на землю или заземленное производственное оборудование, а также электрический разряд с незаземленного оборудования через тело человека могут вызвать болевые и нервные ощущения и быть

причиной непроизвольного резкого движения в результате которого человек может получить травму (падения, ушибы и т.д.).

– повышенный уровень электромагнитных излучений;

В СанПиН энергетическая экспозиция за рабочий день (рабочую смену) не должна превышать значений, указанных в таблице 4.2.

Таблица 4.2 – Предельно допустимые значения энергетической экспозиции

Ввиду того, что технические средства не позволяют воспроизвести обозначения степени, далее в тексте они обозначены показателем степени в скобках Диапазоны частот	Предельно допустимая энергетическая экспозиция		
	По электрической составляющей, (В/м)(2) х ч	По магнитной составляющей, (А/м)(2) х ч	По плотности потока энергии, (мкВт/кв.см) х ч
30 кГц – 3 МГц	20000,0	200,0	
3 – 30 МГц	7000,0	Не разработаны	
30 – 50 МГц	800,0	0,72	
50 – 300 МГц	800,0	Не разработаны	
300 МГц – 300 ГГц			200,0

– повышенная или пониженная температура воздуха рабочей зоны;

– повышенная или пониженная подвижность воздуха;

Для поддержания оптимальных параметров микроклимата предусматривается кондиционирование воздуха второго класса, что позволяет достичь нормируемой чистоты и метеорологические условия воздуха, для чего используется автоматическое регулирование установок кондиционирования воздуха, которых в ВЦ установлено две9. Подача воздуха для охлаждения ЭВМ предусматривается для каждой машины по собственному воздуховоду, что исключит возможность распространения пожара с одной машины на другую. Для обогрева помещений в холодные периоды года предусматривается система отопления, которая должна быть пожаро и взрывобезопасна. В качестве системы отопления можно использовать систему центрального водяного отопления, достоинствами которой являются ее гигиеничность, надежность в эксплуатации и возможность регулирования температуры в широких пределах.

Таблица 4.3 – Оптимальные величины показателей микроклимата на рабочих местах офиса

Период года	Категория работ	Температура воздуха, °С	Относительная влажность воздуха. %	Скорость движения воздуха, м/с
Холодный и переходный (t<8°С)	Iб	21-23	60-40	0.1
Теплый (t³8°С)	Iб	22-24	60-40	0.1

– повышенная или пониженная влажность воздуха;

– отсутствие или недостаток естественного света;

Причина возникновения заключается в несоответствии естественного и искусственного освещения установленным нормам. Слабое освещение приводит к напряжению глаз, что при длительном воздействии ведет к ухудшению зрения. Также возникает головная боль, нервное напряжение.

– повышенная пульсация светового потока;

– недостаточная освещенность рабочего места;

В помещениях офиса предусматривается естественное и искусственное освещение в соответствие со СНиП 23-05-95 [21]. Естественное освещение в офисе применяют одностороннее боковое с кео = 1%, светопроемы ориентированы преимущественно на север и северо-восток, освещенность Е=300 лк. Рабочие места операторов, работающих с дисплеями, располагают на удалении от окон 1.2 м и таким образом, чтобы окна находились слева. Искусственное освещение в помещениях осуществляется системой общего равномерного освещения. В случаях преимущественной работы с документами допускается применение системы комбинированного освещения (к общему освещению дополнительно устанавливаются светильники местного освещения для освещения зоны расположения документов). Для исключения засветки экранов дисплеев прямыми световыми потоками светильники общего освещения располагают сбоку от рабочего места, параллельно линии зрения оператора и стене с окнами. В качестве источников света при искусственном освещении должны применяться преимущественно люминесцентные лампы ЛБ-80. В светильниках местного освещения допускается применение ламп накаливания.

– повышенный уровень шума на рабочем месте;

Основным источником шума является компьютерное оборудование. Воздействие шума отражается как на органах слуха, так и на общем психологическом состоянии человека. Возможны глухота, нервные расстройства.

– умственное перенапряжение;

– эмоциональные нагрузки.

Эмоциональные нагрузки – это способность работника влиять на результат собственного труда при различных уровнях сложности осуществляемой деятельности.

Характеристика работы по показателю «эмоциональные нагрузки» в зависимости от класса условий труда подразделяется на:

– «степень ответственности за результат собственной деятельности. Значимость ошибки» – указывает, в какой мере работник может влиять на результат собственного труда при различных уровнях сложности осуществляемой деятельности. С возрастанием сложности повышается степень ответственности, поскольку ошибочные действия приводят к дополнительным усилиям со стороны работника или целого коллектива, что соответственно приводит к увеличению эмоционального напряжения;

– «степень риска для собственной жизни»- вероятность наступления нежелательного события в случае наличия травмоопасных факторов (короткое замыкание, удар током, самовозгорание и т.д.);

– «степень ответственности за безопасность других лиц» – возможность возникновения нежелательных событий в случае наличия травмоопасных факторов при коллективном выполнении работ. Учитывается только прямая ответственность за безопасность других лиц, предусмотренная должностной инструкцией;

– «количество конфликтных ситуаций в офисе, обусловленных профессиональной деятельностью за день» – определяется на основании хронометражных наблюдений;

– снижение уровня зрения.

В результате постоянной работы за компьютером могут развиться проблемы со зрением, близорукость. Необходимо делать постоянные перерывы в работе с компьютером. Полезным будет использование специальных компьютерных очков.

4.2 Расчет пожарной безопасности проектируемого объекта

В большинстве случаев пожары возникают из-за неисправностей и неправильной эксплуатации электротехнических установок и устройств, по причине коротких замыканий в электрических сетях, при перегреве и воспламенении сгораемых веществ и материалов. Вычислительный центр относится к классу II по степени огнестойкости. По степени пожарной опасности данное производство может быть отнесено к категории В.

Причинами пожара в производстве могут стать короткое замыкание в цепях питания электрооборудования; значительные перегрузки проводки; плохие контакты в местах соединения проводников, приводящие к увеличению переходного сопротивления, на котором выделяется большое количество тепла; небрежное обращение с огнем; удары молнии и др.

Так как на производственном участке имеется большое количество электрооборудования, предполагается использовать установку газового объемного пожаротушения, В качестве огнегасительного вещества используется комбинированный углекислотно-хладоновый состав (85% двуокиси углерода, 15% хладона 111В 2).

Рассчитаем необходимую массу огнегасительного вещества. Производственный участок – помещение размером 5х 10 метров, высота потолков – 3 м:

1 Требуемая масса комбинированного углекислотно-хладонового состава m_d, кг, для объемного пожаротушения определяется по формуле

$$m_d=k_6 q_n V \qquad (4.1)$$

где k_6 – коэффициент компенсации не учитываемых потерь углекислотно-хладонового состава, принимается равным 1,2;

q_n – нормативная массовая огнетушащая концентрация углекислотно-хладонового состава, принимается 0,27 кг/м ^{3} при времени заполнения помещения, равным 30 сек;

V – объем защищаемого помещения, м ^{3}.

m_d=1,2*0,27*10*5*3=48,6 кг

2 Количество ξ_1 баллонов определяется из расчета вместимости в 40-литровый баллон 25 кг углекислотно-хладонового состава

ξ_1= m_d/25=48,6/25=2 полных баллона

3 Внутренний диаметр магистрального трубопровода d_i, мм, определяется по формуле

$$d_1 = d_1\sqrt{\xi_2} \qquad (4.2)$$

где d_1 – диаметр сифонной трубки баллона, мм (30 мм);

ξ_2 – число одновременно разряжаемых баллонов.

$d_1 = 30\sqrt{2} = 43$ мм

4 Эквивалентная длина магистрального трубопровода l_2, м, определяется по формуле

$$l_2=k_7 \qquad (4.3)$$

где k_7 – коэффициент увеличения длины трубопровода для компенсации не учитываемых местных потерь (принимается равным 1,1);

l – длина трубопровода по проекту, м (принимается равной 30 м).

l_2=1,1*30=33 м

5 Площадь сечения выходного отверстия оросителя Аз, мм ^{2}, определяется по формуле

$$Аз=S/\xi_1 \qquad (4.4)$$

где S – площадь сечения магистрального трубопровода, мм ^{2};

ξ_1 – число оросителей.

Аз=3,1415*2*33/8=26 мм ^{2}

6 Расход углекислотно-хладонового состава Q, кг/с, в зависимости от эквивалентной длины и диаметра трубопровода

$$Q=5,6 \text{ кг/с} \quad (4.13)$$

7 Расчетное время подачи углекислотно-хладонового состава t, мин, определяется по формуле

$$t=m_d/60Q \quad (4.14)$$

где m_d – расчетная масса углекислотно-хладонового состава, кг;

Q – расход углекислотно-хладонового состава, кг/с.

t=48,6/5,6=8,7 мин

8 Масса основного запаса углекислотно-хладонового состава, m, кг, определяется по формуле

$$m = 1,1m_d(1+k_8/k_6) \quad (4.15)$$

где k_8 – коэффициент, учитывающий остаток углекислотно-хладонового состава в баллонах и трубопроводах, равен 0.2.

m = 1,1*48,6*(1+0.2/1.2)=62,4 кг

Насадки расположены на потолке в два ряда по четыре штуки в ряду на расстоянии 1.5 м от стен и 2м друг от друга. Они соединены последовательно магистральной трубой диаметром 33 мм, баллоны с газом расположены в соседнем помещении. План помещений представлен на рисунке 4.1.

Определим технические и организационные мероприятия. К техническим мероприятиям относятся противопожарные меры, применяемые при строительстве вычислительного центра. В частности, при строительстве необходимо соблюсти следующее:

1 территорию производства необходимо постоянно содержать в чистоте, горючий мусор должен систематически удаляться на специально отведенные участки и по мере накопления вывозиться;

2 все токоведущие части, распределительные устройства, рубильники и другие пусковые аппараты монтируются на негорючих основаниях (мрамор, текстолит, гетинакс, асбест, и т.п.);

3 измерение сопротивления изоляции электросети проводится не реже двух раз в год. Неисправные участки обесточиваются и заменяются новыми;

4 отопление аккумуляторного помещения делается централизованным (водяным или паровым) в виде целых сварных труб без фланцев и вентилей;

5 курение в помещении строго воспрещается;

6 на случай возникновения пожара необходимо предусмотреть возможность эвакуации людей. Эвакуационные пути должны обеспечивать эвакуацию всех людей, находящихся в здании в течение необходимого времени. Число эвакуационных путей не менее двух;

7 двери на путях эвакуации навешиваются так, чтобы открывались по направлению выхода из здания;

8 устройство раздвижных и подъездных дверей на путях эвакуации не допускается;

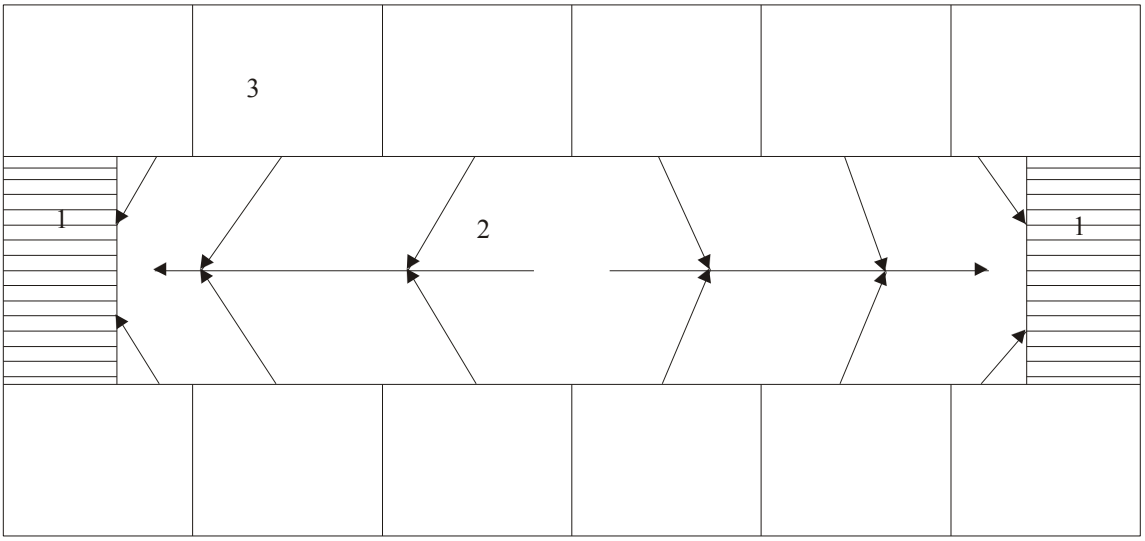
9 минимальная ширина дверей на путях эвакуации не менее 0,8м;

10 высота перехода на путях эвакуации не менее 2 м;

11 устройство винтовых лестниц и забежных ступеней на путях эвакуации не допускается;

12 схема эвакуации людей тщательно разрабатывается и вывешивается на видных местах;

13 весь трудовой коллектив проходит обучение мерам противопожарной безопасности.



1 – лестничный пролет и выход.
2 – коридор.
3 – офис.

Рисунок 4.1 – План эвакуации

4.3 Расчет уровня шума

Одним из неблагоприятных факторов производственной среды в ИВЦ является высокий уровень шума, создаваемый печатными устройствами, оборудованием для кондиционирования воздуха, вентиляторами систем охлаждения в самих ЭВМ.

Для решения вопросов о необходимости и целесообразности снижения шума необходимо знать уровни шума на рабочем месте оператора.

Уровень шума, возникающий от нескольких некогерентных источников, работающих одновременно, подсчитывается на основании принципа энергетического суммирования излучений отдельных источников

$$L_{\Sigma} = 10 \lg \sum_{i=1}^{i=n} 10^{0,1 L_i}, \tag{4.16}$$

где L_i – уровень звукового давления i-го источника шума;

n – количество источников шума.

Полученные результаты расчета сравнивается с допустимым значением уровня шума для данного рабочего места. Если результаты расчета выше допустимого значения уровня шума, то необходимы специальные меры по снижению шума. К ним относятся: облицовка стен и потолка зала звукопоглощающими материалами, снижение шума в источнике, правильная планировка оборудования и рациональная организация рабочего места оператора.

Уровни звукового давления источников шума, действующих на оператора на его рабочем месте представлены в таблице 4.4.

Т а б л и ц а 4.4 – Уровни звукового давления различных источников.

Источник шума	Уровень шума, дБ
Жесткий диск	40
Вентилятор	45
Монитор	17
Клавиатура	10
Принтер	45
Сканер	42

Обычно рабочее место оператора оснащено следующим оборудованием: винчестер в системном блоке, вентилятор(ы) систем охлаждения ПК, монитор, клавиатура, принтер и сканер.

Подставив значения уровня звукового давления для каждого вида оборудования в формулу, получим:

$$L_{?}=10\cdot\lg(10^{4}+10^{4,5}+10^{1,7}+10^{1}+10^{4,5}+10^{4,2})=49,5 \text{ Дб}$$

Полученное значение не превышает допустимый уровень шума для рабочего места оператора, равный 65 дБ (СНиП 12.1.003-83). И если учесть, что вряд ли такие периферийные устройства как сканер и принтер будут использоваться одновременно, то эта цифра будет еще ниже. Кроме того при работе принтера непосредственное присутствие оператора необязательно, т.к. принтер снабжен механизмом автоподачи листов. Необходимо просто, время от времени добавлять листы в податчик.

Заключение

В данной дипломной работе мною было спроектировано и реализовано web-приложение, которое в значительной мере упростит изучение курса «Проектирование баз данных» в нашем университете. Система создана гибкой и расширяемой, нет никаких сложностей, чтобы расширить ее и на другие курсы. Возможность включения мультимедиа материалов и соревновательных элементов повышает вовлечение студентов в процесс обучения. Встроенные системы общения и взаимодействия снижают постороннюю нагрузку на студентов и учителей.

В качестве технологий для серверной части используются: PHP, фреймворк YII, а также сервер Apache.

Фронтэнд построен на базе twitter Bootstrap 3.1.2, свзяь клиента и сервера написана с помощью jQuery.

Дизайн выполнен в духе современного интернета, в стиле flat UI, что гарантирует отличное понимание работы системы всеми категориями пользователей.

В разработки использовались новейшие технологии HTML 5 и CSS 3. Система отлично работает как на компьютерах, так и на любых современных мобильных устройствах, дизайн автоматически адаптируется к разрешению экрана.

Список используемой литературы

1 Сайт http://scholar.urc.ac.ru/ped_journal/numero4/pedag/tsit3.html.ru

2 Адам Ф. jQuery для профессионалов = Pro jQuery. – 3-е изд. – СПб: Питер, 2012. – 689 с.

3 Сайт http://dic.academic.ru/dic.nsf/fin_enc/19812

4 Ганник Дж. MSSQL 2008, Карманный справочник. – 2-е изд. – СПб: Питер, 2009. – 389 с.

5 Сайт http://getbootstrap.com/css/

6 Роб П., Коронел К. Системы баз данных: проектирование, реализация и управление. – 5-е изд. – Спб.: БХВ-Петербург, 2004. – 1040 с.

7 Котеров Д., Костарев А. PHP. В подлиннике. – Спб.: БХВ-Петербург, 2005. – 1120 с.

8 Зандстра М. PHP: объекты, шаблоны и методики программирования. – 3-е изд. – М.: Вильямс, 2010. – 560 с.

9 Сайт http://php.ru/manual/

10 Кузнецов М., Симдянов И. PHP . Практика создания Web-сайтов. – 2-е изд. – Спб.: БХВ-Петербург, 2008. – 1264 с.

11 Флэнаган Д. Чего не может JavaScript. Подробное руководство / Перевод Киселева А. – 5-е изд. – СПб.: Символ-Плюс, 2008. – 280 с.

12 Кебо Б. Библия jQuery. Руководство професионала / Перевод Тушко Г. – 2-е изд. – СПб.: Амфора, 2012. – 454 с.

13 Christian J. Test Driven JavaScript Development. – Addison: Wesley Professional, 2010. – 26 с.

14 Стешин А. Изучаем сервер под windows. – СПб: Питер, 2009. – 209 с.

15 Гукасьян Г.М. Экономика от «А» до «Я»: Тематический справочник. – М.: ИНФРА-М, 2009. – 480 с.

16 Иванов И.Н. Экономика промышленного предприятия: Учебник. – М.: ИНФРА-М, 2011. – 395 с.

17 Рофе А.И. Экономика труда: Учебник. – М.: КноРус, 2010. – 400 с.

18 Скляренко В.К., Прудников В.М. Экономика предприятия: Учебник. – М.: ИНФРА-М, 2006. – 528 с.

19 Белов С.В., Ильницкая А.В., Козьяков А.Ф. Безопасность жизнедеятельности. Учебник для вузов. – М.: Высшая школа, 2005. – 448 с.

20 СНиП РК 4.02–42–2006 Отопление, вентиляция и кондиционирование. – Астана: Издательство стандартов, 2007.

21 СНиП РК 2.02–15–2003 Пожарная автоматика зданий и сооружений. – Астана: Издательство стандартов, 2004.

22 Хакимжанов Т.Е. Сборник задач по охране труда и безопасности жизнедеятельности: Учебное пособие для вузов. – Алматы: Эверо, 2007. – 274 с.

Приложение А

Листинг программы

```
<?php

class EXFEEDController extends GxController
{


    public function actionView($id)
    {
        $this->render('view', array(
            'model' => $this->loadModel($id, 'EXFEED'),
        ));
    }

    public function actionCreate()
    {
        $model = new EXFEED;


        if (isset($_POST['EXFEED'])) {
            $model->setAttributes($_POST['EXFEED']);

            if ($model->save()) {
                if (Yii::app()->getRequest()->getIsAjaxRequest())
                    Yii::app()->end();
                else
                    $this->redirect(array('view', 'id' => $model-
>FEED_ID));
            }
        }

        $this->render('create', array('model' => $model));
    }


    public function actionCreatehidden()
    {
        $sid = $_POST['student'];
        $student = USER::model()->findByPk($sid);
        if ($_POST['ef']!='') {
            $exfeed=EXFEED::model()->findByPk($_POST['ef']);
            $exfeed->EXFEED_STATUS=0;
        }
        else {
        $exfeed = new EXFEED();
        }
        $exfeed->users_id = $sid;
        $exfeed->EXFEED_INTEXT = $_POST['sql'];
```

```
  $exfeed->Level_ID = $_POST['level'];
        $exfeed->EXFEED_INCOME = (date("d-m-Y H:i:s", time()));
        if ($exfeed->save()) {
            echo("Задание отправленно на проверку!");
        } else {
            echo("Что-то пошло не так, повторите попытку.");

        }
        // print_r($exfeed);
    }

    public function actionUpdate($id)
    {
        $model = $this->loadModel($id, 'EXFEED');


        if (isset($_POST['EXFEED'])) {
            $model->setAttributes($_POST['EXFEED']);

            if ($model->save()) {
                $this->redirect(array('view', 'id' => $model-
>FEED_ID));
            }
        }

        $this->render('update', array(
            'model' => $model,
        ));
    }

    public function actionDelete($id)
    {
        if (Yii::app()->getRequest()->getIsPostRequest()) {
            $this->loadModel($id, 'EXFEED')->delete();

            if (!Yii::app()->getRequest()->getIsAjaxRequest())
                $this->redirect(array('admin'));
        } else
            throw new CHttpException(400, Yii::t('app', 'Your
request is invalid.'));
    }

    public function actionIndex()
    {
        $dataProvider = new CActiveDataProvider('EXFEED');
        $this->render('index', array(
            'dataProvider' => $dataProvider,
        ));
    }
```

```
{
         $exfeed=EXFEED::model()-
>findByAttributes(array('FEED_ID'=>$_POST['exfeed']));
         $level=CURSLEVEL::model()-
>findByAttributes(array('Level_id'=>$_POST['level']));
        $part=CURSPART::model()-
>findByAttributes(array('Part_id'=>$_POST['part']));
      $this->renderPartial("_feedcheck", array('exfeed'=>$exfeed,
'level'=>$level, 'part'=>$part));
    }

    public function actionFeed()
    {
       $model=EXFEED::model()->findAll();
       $users=USER::model()->findAll();
       $levels=CURSLEVEL::model()->findAll();
        $parts=CURSPART::model()->findAll();
        $this->render('feed', array(
            'model' => $model,
            'users'=>$users,
            'levels'=>$levels,
            'parts'=>$parts,
        ));
    }

    public function actionPullright()
    {

        $s=$_POST['status'];
        $ef=EXFEED::model()->findByPk($_POST['exfeed']);
        $ef->EXFEED_INCOME=(date("d-m-Y H:i:s",  $ef-
>EXFEED_INCOME));
        if ($s==1) {
            $ef->EXFEED_STATUS=1;

            if ($ef->save()){
                echo "Задание помечено выполненным!";
            }
        }
        if ($s==2) {
            $ef->EXFEED_STATUS=2;
            if ($ef->save()){
                echo "Решение помечено ошибочным!";
            }
        }

    }

    public function actionAdmin()
    {
```

```
$model = new EXFEED('search');
        $model->unsetAttributes();

        if (isset($_GET['EXFEED']))
            $model->setAttributes($_GET['EXFEED']);

        $this->render('admin', array(
            'model' => $model,
        ));
    }

}

<?php

class StudentController extends Controller
{
     /**
      * @var string the default layout for the views. Defaults to
'//layouts/column2', meaning
      * using two-column layout. See
'protected/views/layouts/column2.php'.
      */
     public $layout='//layouts/column2';

     /**
      * @return array action filters
      */
     public function filters()
     {
          return array(
               'accessControl', // perform access control for CRUD
operations
               'postOnly + delete', // we only allow deletion via
POST request
          );
     }

     /**
      * Specifies the access control rules.
      * This method is used by the 'accessControl' filter.
      * @return array access control rules
      */
     public function accessRules()
     {
          return array(
               array('allow',  // allow all users to perform
'index' and 'view' actions
                    'actions'=>array('index','view','updateFast',
'upload'),
```

```
'users'=>array('*'),
                ),
                array('allow', // allow authenticated user to
perform 'create' and 'update' actions
                     'actions'=>array('create','update'),
                     'users'=>array('@'),
                ),
                array('allow', // allow admin user to perform
'admin' and 'delete' actions
                     'actions'=>array('admin','delete'),
                     'users'=>array('admin'),
                ),
                array('deny',  // deny all users
                     'users'=>array('*'),
                ),
           );
     }

     /**
      * Displays a particular model.
      * @param integer $id the ID of the model to be displayed
      */
     public function actionView($id)
     {
          $this->render('view',array(
               'model'=>$this->loadModel($id),
          ));
     }

     /**
      * Creates a new model.
      * If creation is successful, the browser will be redirected
to the 'view' page.
      */
     public function actionCreate()
     {
          $model=new STUDENT;

          // Uncomment the following line if AJAX validation is
needed
          // $this->performAjaxValidation($model);

          if(isset($_POST['STUDENT']))
          {
               $model->attributes=$_POST['STUDENT'];
               if($model->save())
                    $this->redirect(array('view','id'=>$model-
>Stud_id));
          }
```

```
     $model=$this->loadModel($id);

          // Uncomment the following line if AJAX validation is
needed
          // $this->performAjaxValidation($model);

          if(isset($_POST['STUDENT']))
          {
               $model->attributes=$_POST['STUDENT'];
               if($model->save())
                    $this->redirect(array('view','id'=>$model-
>Stud_id));
          }

          $this->render('update',array(
               'model'=>$model,
          ));
     }

    public function actionUpdateFast()
    {


         // Uncomment the following line if AJAX validation is
needed
         // $this->performAjaxValidation($model);

        $model=new STUDENT;
         if(isset($_POST['STUDENT']))
         {

             $model=STUDENT::model()-
>findByPk($_POST['STUDENT']['Stud_id']);
             $model->attributes=$_POST['STUDENT'];
             $model->save();
              $this->redirect(array('view','id'=>$model->Stud_id));
         }

    }

     /**
      * Deletes a particular model.
      * If deletion is successful, the browser will be redirected
to the 'admin' page.
      * @param integer $id the ID of the model to be deleted
      */
     public function actionDelete($id)
     {
          $this->loadModel($id)->delete();
```

```
          // if AJAX request (triggered by deletion via admin grid
view), we should not redirect the browser
          if(!isset($_GET['ajax']))
               $this->redirect(isset($_POST['returnUrl']) ?
$_POST['returnUrl'] : array('admin'));
     }

     /**
      * Lists all models.
      */
     public function actionIndex()
     {
          $dataProvider=new CActiveDataProvider('STUDENT');
          $this->render('index',array(
               'dataProvider'=>$dataProvider,
          ));
     }

     /**
      * Manages all models.
      */
     public function actionAdmin()
     {
          $model=new STUDENT('search');
          $model->unsetAttributes();  // clear any default values
          if(isset($_GET['STUDENT']))
               $model->attributes=$_GET['STUDENT'];

          $this->render('admin',array(
               'model'=>$model,
          ));
     }

    public function actionUpload()
    {
        Yii::import("ext.EAjaxUpload.qqFileUploader");
        $folder=Yii::getPathOfAlias('webroot').'/avatar/';//
folder for uploaded files
        $allowedExtensions =
array("jpg","png");//array("jpg","jpeg","gif","exe","mov" and
etc...
        $sizeLimit = 1 * 1024 * 1024;// maximum file size in bytes
        $uploader = new qqFileUploader($allowedExtensions,
$sizeLimit);
        $newfilename =  md5(microtime());
        $filename=$newfilename.'.'.$upload['extension'];
        $result = $uploader->handleUpload($folder);

        $fileSize=filesize($folder.$result['filename']);//GETTING
FILE SIZE
```

```
 //  $fileName=$result['filename'];//GETTING FILE NAME
        $result=htmlspecialchars(json_encode($result),
ENT_NOQUOTES);

        echo $result;// it's array
    }

     /**
      * Returns the data model based on the primary key given in
the GET variable.
      * If the data model is not found, an HTTP exception will be
raised.
      * @param integer $id the ID of the model to be loaded
      * @return STUDENT the loaded model
      * @throws CHttpException
      */
     public function loadModel($id)
     {
          $model=STUDENT::model()->findByPk($id);
          if($model===null)
               throw new CHttpException(404,'The requested page
does not exist.');
          return $model;
     }

     /**
      * Performs the AJAX validation.
      * @param STUDENT $model the model to be validated
      */
     protected function performAjaxValidation($model)
     {
          if(isset($_POST['ajax']) && $_POST['ajax']==='student-
form')
          {
               echo CActiveForm::validate($model);
               Yii::app()->end();
          }
     }
}
 ST['ef']!='') {
            $exfeed=EXFEED::model()->findByPk($_POST['ef']);
            $exfeed->EXFEED_STATUS=0;
        }
        else {
        $exfeed = new EXFEED();
        }
        $exfeed->users_id = $sid;
        $exfeed->EXFEED_INTEXT = $_POST['sql'];
        $exfeed->Level_ID = $_POST['level'];
        $exfeed->EXFEED_INCOME = (date("d-m-Y H:i:s", time()));
```

```
  if ($exfeed->save()) {
            echo("Задание отправленно на проверку!");
        } else {
            echo("Что-то пошло не так, повторите попытку.");

        }
        // print_r($exfeed);
    }

    public function actionUpdate($id)
    {
        $model = $this->loadModel($id, 'EXFEED');


        if (isset($_POST['EXFEED'])) {
            $model->setAttributes($_POST['EXFEED']);

            if ($model->save()) {
                $this->redirect(array('view', 'id' => $model-
>FEED_ID));
            }
        }

        $this->render('update', array(
            'model' => $model,
        ));
    }

    public function actionDelete($id)
    {
        if (Yii::app()->getRequest()->getIsPostRequest()) {
            $this->loadModel($id, 'EXFEED')->delete();

            if (!Yii::app()->getRequest()->getIsAjaxRequest())
                $this->redirect(array('admin'));
        } else
            throw new CHttpException(400, Yii::t('app', 'Your
request is invalid.'));
    }

    public function actionIndex()
    {
        $dataProvider = new CActiveDataProvider('EXFEED');
        $this->render('index', array(
            'dataProvider' => $dataProvider,
        ));
    }

    public function actionLoad()
    {
```

```
$exfeed=EXFEED::model()-
>findByAttributes(array('FEED_ID'=>$_POST['exfeed']));
         $level=CURSLEVEL::model()-
>findByAttributes(array('Level_id'=>$_POST['level']));
        $part=CURSPART::model()-
>findByAttributes(array('Part_id'=>$_POST['part']));
      $this->renderPartial("_feedcheck", array('exfeed'=>$exfeed,
'level'=>$level, 'part'=>$part));
    }

    public function actionFeed()
    {
       $model=EXFEED::model()->findAll();
       $users=USER::model()->findAll();
       $levels=CURSLEVEL::model()->findAll();
        $parts=CURSPART::model()->findAll();
        $this->render('feed', array(
            'model' => $model,
            'users'=>$users,
            'levels'=>$levels,
            'parts'=>$parts,
        ));
    }

    public function actionPullright()
    {

        $s=$_POST['status'];
        $ef=EXFEED::model()->findByPk($_POST['exfeed']);
        $ef->EXFEED_INCOME=(date("d-m-Y H:i:s",  $ef-
>EXFEED_INCOME));
        if ($s==1) {
            $ef->EXFEED_STATUS=1;

            if ($ef->save()){
                echo "Задание помечено выполненным!";
            }
        }
        if ($s==2) {
            $ef->EXFEED_STATUS=2;
            if ($ef->save()){
                echo "Решение помечено ошибочным!";
            }
        }

    }

    public function actionAdmin()
    {
        $model = new EXFEED('search');
```

```
 $model->unsetAttributes();

        if (isset($_GET['EXFEED']))
            $model->setAttributes($_GET['EXFEED']);

        $this->render('admin', array(
            'model' => $model,
        ));
    }

}

<?php

class StudentController extends Controller
{
     /**
      * @var string the default layout for the views. Defaults to
'//layouts/column2', meaning
      * using two-column layout. See
'protected/views/layouts/column2.php'.
      */
     public $layout='//layouts/column2';

     /**
      * @return array action filters
      */
     public function filters()
     {
          return array(
               'accessControl', // perform access control for CRUD
operations
               'postOnly + delete', // we only allow deletion via
POST request
          );
     }

     /**
      * Specifies the access control rules.
      * This method is used by the 'accessControl' filter.
      * @return array access control rules
      */
     public function accessRules()
     {
          return array(
               array('allow',  // allow all users to perform
'index' and 'view' actions
                    'actions'=>array('index','view','updateFast',
'upload'),
                    'users'=>array('*'),
```

```
          ),
               array('allow', // allow authenticated user to
perform 'create' and 'update' actions
                    'actions'=>array('create','update'),
                    'users'=>array('@'),
               ),
               array('allow', // allow admin user to perform
'admin' and 'delete' actions
                    'actions'=>array('admin','delete'),
                    'users'=>array('admin'),
               ),
               array('deny',  // deny all users
                    'users'=>array('*'),
               ),
          );
     }

     /**
      * Displays a particular model.
      * @param integer $id the ID of the model to be displayed
      */
     public function actionView($id)
     {
          $this->render('view',array(
               'model'=>$this->loadModel($id),
          ));
     }

     /**
      * Creates a new model.
      * If creation is successful, the browser will be redirected
to the 'view' page.
      */
     public function actionCreate()
     {
          $model=new STUDENT;

          // Uncomment the following line if AJAX validation is
needed
          // $this->performAjaxValidation($model);

          if(isset($_POST['STUDENT']))
          {
               $model->attributes=$_POST['STUDENT'];
               if($model->save())
                    $this->redirect(array('view','id'=>$model-
>Stud_id));
          }
```

```
$this->render('create',array(
                'model'=>$model,
          ));
     }

     /**
      * Updates a particular model.
      * If update is successful, the browser will be redirected to
the 'view' page.
      * @param integer $id the ID of the model to be updated
      */
     public function actionUpdate($id)
     {
          $model=$this->loadModel($id);

          // Uncomment the following line if AJAX validation is
needed
          // $this->performAjaxValidation($model);

          if(isset($_POST['STUDENT']))
          {
               $model->attributes=$_POST['STUDENT'];
               if($model->save())
                    $this->redirect(array('view','id'=>$model-
>Stud_id));
          }

          $this->render('update',array(
               'model'=>$model,
          ));
     }

    public function actionUpdateFast()
    {


         // Uncomment the following line if AJAX validation is
needed
         // $this->performAjaxValidation($model);

        $model=new STUDENT;
         if(isset($_POST['STUDENT']))
         {

             $model=STUDENT::model()-
>findByPk($_POST['STUDENT']['Stud_id']);
             $model->attributes=$_POST['STUDENT'];
             $model->save();
              $this->redirect(array('view','id'=>$model->Stud_id));
         }
```

```
      * Deletes a particular model.
      * If deletion is successful, the browser will be redirected
to the 'admin' page.
      * @param integer $id the ID of the model to be deleted
      */
     public function actionDelete($id)
     {
          $this->loadModel($id)->delete();

          // if AJAX request (triggered by deletion via admin grid
view), we should not redirect the browser
          if(!isset($_GET['ajax']))
               $this->redirect(isset($_POST['returnUrl']) ?
$_POST['returnUrl'] : array('admin'));
     }

     /**
      * Lists all models.
      */
     public function actionIndex()
     {
          $dataProvider=new CActiveDataProvider('STUDENT');
          $this->render('index',array(
               'dataProvider'=>$dataProvider,
          ));
     }

     /**
      * Manages all models.
      */
     public function actionAdmin()
     {
          $model=new STUDENT('search');
          $model->unsetAttributes();  // clear any default values
          if(isset($_GET['STUDENT']))
               $model->attributes=$_GET['STUDENT'];

          $this->render('admin',array(
               'model'=>$model,
          ));
     }

    public function actionUpload()
    {
        Yii::import("ext.EAjaxUpload.qqFileUploader");
        $folder=Yii::getPathOfAlias('webroot').'/avatar/';//
folder for uploaded files
        $allowedExtensions =
array("jpg","png");//array("jpg","jpeg","gif","exe","mov" and
etc...
```

```
   $sizeLimit = 1 * 1024 * 1024;// maximum file size in bytes
        $uploader = new qqFileUploader($allowedExtensions,
$sizeLimit);
        $newfilename =  md5(microtime());
        $filename=$newfilename.'.'.$upload['extension'];
        $result = $uploader->handleUpload($folder);

        $fileSize=filesize($folder.$result['filename']);//GETTING
FILE SIZE
      //  $fileName=$result['filename'];//GETTING FILE NAME
        $result=htmlspecialchars(json_encode($result),
ENT_NOQUOTES);

        echo $result;// it's array
    }

     /**
      * Returns the data model based on the primary key given in
the GET variable.
      * If the data model is not found, an HTTP exception will be
raised.
      * @param integer $id the ID of the model to be loaded
      * @return STUDENT the loaded model
      * @throws CHttpException
      */
     public function loadModel($id)
     {
          $model=STUDENT::model()->findByPk($id);
          if($model===null)
               throw new CHttpException(404,'The requested page
does not exist.');
          return $model;
     }

     /**
      * Performs the AJAX validation.
      * @param STUDENT $model the model to be validated
      */
     protected function performAjaxValidation($model)
     {
          if(isset($_POST['ajax']) && $_POST['ajax']==='student-
form')
          {
               echo CActiveForm::validate($model);
               Yii::app()->end();
          }
     }
}
```