ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙНАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК
КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ПОДГОТОВКИ ШКОЛЬНИКОВ К СДАЧЕ ЕГЭ ПО ИНФОРМАТИКЕ

Выпускная квалификационная работа
обучающейся по направлению подготовки 02.03.01
Математика и компьютерные науки
очной формы обучения, группы 07001303
Бережной Марины Петровны

Научный руководитель
к.т.н., доцент
В.В. Муромцев

БЕЛГОРОД 2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Материалы Единого экзамена предусматривают возможность появления иных, по сравнению с традиционными, технологий обучения. Особую роль в подготовке успешной сдачи ЕГЭ могут сыграть современные информационные технологии, основой которых являются компьютеры и компьютерные системы, различные электронные средства, аудио- и видеотехника и системы коммуникации.

Основными преимуществами применения современных технических средств и компьютерных программам и систем, являются возможность быстро предъявлять, собирать, обрабатывать, анализировать и интерпретировать учебно-тренировочную информацию, предлагаемую и комплектами пособий для учителей и выпускников школ и демонстрационной версией ЕГЭ.

Электронная обучающая система позволяет:

– самостоятельно работать с учебным материалом;

– получать интерактивную консультацию программы и использовать справочный материал;

– подготовиться к уроку, контрольному занятию, экзамену;

– проходить тестирование по одной или нескольким темам в удобном для себя режиме;

– отработать навыки тестирования;

– контролировать результаты тестирования, которые заносятся в Журнал результатов;

– провести работу над ошибками;

– выявить слабые места в понимании предмета.

Таким образом, можно сделать вывод, что применение автоматизированных систем подготовки школьников к сдаче ЕГЭ целесообразно для всех предметов, а особенно для информатики.

Цель данной работы состоит в разработке автоматизированной информационной системы подготовки школьников к сдаче ЕГЭ на основе современных веб-технологий.

Для достижения цели необходимо решить ряд задач:

– произвести обзор современных подходов к автоматизации процесса подготовки школьников к ЕГЭ;

– сформулировать требования к информационной системе подготовки школьников к сдаче ЕГЭ;

– произвести проектирование архитектуры информационной системы подготовки школьников к сдаче ЕГЭ;

– спроектировать структуры базы данных;

– произвести проектирование пользовательского интерфейса информационной системы подготовки школьников к сдаче ЕГЭ;

– реализовать информационную систему подготовки школьников к сдаче ЕГЭ в соответствии с выдвинутыми требованиями;

– произвести тестирование разработанного программного комплекса.

В первой главе проведен обзор и анализ современных подходов к автоматизации процесса подготовки школьников к ЕГЭ, выявлены требования к информационной системе подготовки школьников к сдаче ЕГЭ.

Во второй главе описан процесс проектирования архитектуры автоматизированной системы подготовки школьников к сдаче ЕГЭ, структуры базы данных и пользовательского интерфейса.

В третьей главе рассмотрены детали реализации базовых компонентов системы и пользовательского интерфейса, приведены результаты тестирования системы.

Выпускная квалификационная работа включает в себя: 53 страницы без приложения, 3 главы, 36 рисунков, 9 листингов и 1 приложение. В процессе создания было использовано 15 литературных источников.

1 ПОСТАНОВКА ЗАДАЧИ НА РАЗРАБОТКУ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

1.1 Необходимость автоматизации процесса подготовки школьников к сдаче ЕГЭ по информатике

Нет необходимости доказывать, что возможность постоянного оперативного мониторинга качества знаний на всем протяжении образовательного процесса — от начальной школы до вуза — существенно повысила бы эффективность обучения. Принято считать, что практика последних лет показала: тестовая форма оценки знаний является методически и экономически эффективным средством контроля качества обучения.

Подготовленность к чему-либо понимается нами как комплекс приобретенных знаний, навыков, умений, качеств, позволяющих успешно выполнять определенную деятельность, в частности – успешно сдать ЕГЭ [1].

Для того чтобы удачно сдать ЕГЭ, во-первых, необходимо владеть достаточно полными знаниями по предмету, во-вторых, иметь опыт написания ЕГЭ и, в-третьих, быть психологически подготовленным к сдаче экзамена. Только комплексный подход к деятельности по подготовке учащихся к ЕГЭ способствует повышению эффективности и качества результатов экзамена в тестовой форме [2].

Особое внимание в процессе деятельности по подготовке учащихся к ЕГЭ занимает мониторинг качества обученности по предмету.

Система диагностики обученности включает в себя:

1. предварительное выявление уровня знаний, умений, навыков обучающихся;
2. текущую проверку в процессе усвоения каждой изучаемой темы, при этом диагностируется уровень отдельных элементов программы;

3. повторную проверку – параллельно с изучением нового материала идет повторение пройденного материала;

4. периодическую проверку знаний, умений, навыков по целому разделу курса для наблюдения за усвоением взаимосвязей между структурными элементами образовательной программы, изучавшимися в разных частях курса;

5. итоговую проверку и учет полученных обучающимися знаний, умений, навыков проводится в конце обучения по предложенной образовательной программе.

Мониторинг обеспечивает возможность прогнозирования оценок на выпускном ЕГЭ.

Оценка подготовки выпускников предполагает сравнение реального уровня обученности ученика с эталонным уровнем, зафиксированным в стандарте. Такая оценка может быть получена с помощью разнообразных форм контроля.

Использование в практике ЕГЭ такой формы, как тестирование, объясняется стремлением получить максимально объективную оценку подготовки выпускников [2].

Необходимо учитывать, что мышление современного школьника во многом ориентировано на тестовые алгоритмы, поэтому можно предположить, что использование разного вида тестов и органичное включение тестового материала в уроки будет способствовать повышению качества знаний по предмету. Очень важно, чтобы задания, включённые в ЕГЭ, были знакомы учащимся с 5-го класса, повторялись из урока в урок и не требовали специального повторения к экзамену.

Преимущества тестов по сравнению со всеми возможными формами контроля можно свести к следующему:

– все учащиеся находятся в равных условиях, что позволяет объективно сравнивать их достижения в усвоении предмета;

– сведена к минимуму субъективность при оценке задания С (задание с свободным ответом);

– результаты поддаются статистической обработке;

– существенно экономится время, затрачиваемое на проверку;

– за сравнительно небольшой отрезок времени можно проверить знания, умения и навыки учащихся по всем разделам школьного курса.

ЕГЭ – это один из основных моментов модернизации образования. Материалы Единого экзамена предусматривают возможность появления иных, по сравнению с традиционными, технологий обучения [3].

Особую роль в подготовке успешной сдачи ЕГЭ могут сыграть современные информационные технологии, основой которых являются компьютеры и компьютерные системы, различные электронные средства, аудио- и видеотехника и системы коммуникации.

Для эффективного использования учебных компьютерных технологий в подготовке к ЕГЭ необходим ряд требований.

Основным требованием, предъявляемым к техническим средствам и компьютерным программам и системам, является возможность быстро предъявлять, собирать, обрабатывать, анализировать и интерпретировать учебно-тренировочную информацию, предлагаемую и комплектами пособий для учителей и выпускников школ и демонстрационной версией ЕГЭ.

Следует помнить, что ЕГЭ – это одна из форм аттестации и готовиться к экзамену нужно в ходе обычного процесса обучения. Задания, отвечающие типам заданий, включенных в ЕГЭ, должны предлагаться в течение всего учебного года для контроля знаний и обучения учащихся [4].

Новые информационные технологии, в том числе компьютерная коммуникация, позволяют совершенствовать учебный процесс в целом и подготовку к сдаче Единого экзамена в частности.

Следует отметить эффективность использования компьютерных средств обучения в процессе контрольно-оценочной деятельности школьника.

Электронная обучающая система позволяет:

1. самостоятельно работать с учебным материалом;
2. получать интерактивную консультацию программы и использовать справочный материал;
3. подготовиться к уроку, контрольному занятию, экзамену;
4. проходить тестирование по одной или нескольким темам в удобном для себя режиме;
5. отработать навыки тестирования;
6. контролировать результаты тестирования, которые заносятся в Журнал результатов;
7. провести работу над ошибками;
8. выявить слабые места в понимании предмета.

1.2 Обзор существующих решений для автоматизации систем подготовки к сдаче ЕГЭ

Региональная образовательная система тестирования («РОСТ»). Предназначена для создания учебных тестов, проведения тестирований и анализа результатов, полученных при тестировании обучающихся. «РОСТ» интегрирована с информационными системами «Сетевой регион. Образование», «Сетевой город. Образование», «NetSchool».

«РОСТ» предоставляет широкие возможности при создании тестов. Автор тестов может группировать тесты по темам и предметам, создавать вопросы пяти различных типов, назначать уровень сложности вопросов, включать в текст как вопроса, так и ответа различный контент: текстовые документы, аудио-, видеоматериалы и графику.

Используя систему, преподаватель так же может составлять сценарии проведения тестирований (т.е. указывать, сколько вопросов, по какой теме, какого уровня сложности будет задано обучающимся из общей базы вопросов),

задавать гибкие настройки режимов времени прохождения теста, выбирать порядок ответов на вопросы.

Кроме того, автору тестов доступны настройки перемешивания: тем, вопросов, вариантов ответа; настройки оценивания вопросов в баллах, которые получит обучающийся за правильный ответ на вопрос.

При этом, количество набранных баллов соотносится с оценкой по различным оценочным шкалам. Оценка по тесту автоматически заносится в электронный классный журнал и электронный дневник информационных систем «Сетевой регион. Образование», «Сетевой город. Образование», «NetSchool».

Во время анализа полученных результатов имеется возможность получать различные отчеты по классам, группам, обучающимся, позволяющие узнать процент справившихся с тем или иным заданием, типичные ошибки, среднюю успеваемость учеников по тому или иному тесту или предмету.

Система тестирования «INDIGO». Представляет собой мультифункциональный комплекс программного обеспечения, позволяющий автоматизировать процесс проведения тестирования и обработки результатов. Система «INDIGO» является универсальным инструментом, который можно использовать для решения широкого спектра задач:

– определение уровня готовности учащихся школ к ГИА и ЕГЭ;

– тестирование и контроль знаний по различным дисциплинам;

– автоматизация проведения викторин и олимпиад.

Программа тестирования «INDIGO»:

– позволяет эффективно производить автоматизацию тестирования за счет широких функциональных возможностей;

– обеспечивает надежность и удобство работы пользователей при создании тестов и в процессе тестирования за счет удобного и современного пользовательского интерфейса;

– ликвидирует физические и временные затраты по определению результатов тестирования за счет автоматического расчета баллов по заданным параметрам.

КРОК АИС «Экзамен». Проблема автоматизации контроля знаний решается с помощью автоматизированной информационной системы (АИС) «Экзамен». Это единственная на сегодняшний день информационная система, которая автоматизирует весь комплекс работ по планированию, организации, проведению и обработке результатов массового тестирования, в том числе ЕГЭ, различных олимпиад, выпускных экзаменов в школе или вступительных экзаменов в вузы. АИС «Экзамен» обеспечивает полную конфиденциальность авторства ответов учеников до получения окончательных результатов и поддерживает технологию подачи и разбора апелляций.

Система предусматривает различные уровни интеграции обработки информации. Она может функционировать, подчиняясь единым формам заданий, приходящим из центра, или настраиваться под индивидуальные задания для отдельных организаций. Такая беспрецедентная гибкость является залогом широкого применения АИС «Экзамен».

Функциональное ядро АИС — система распознавания рукописных документов, выполненных в виде бланочных машиночитаемых документов (МЧД). Технология обработки и форма бланков обеспечивают проведение тестирования по различным типам заданий — от выбора из списка правильного ответа до краткой свободной формы изложения своих знаний. Процесс тестирования начинается со сбора данных об учащихся и планирования проведения испытаний, а после проведения тестов и проверки знаний завершается сбором и обобщением результатов.

АИС «Экзамен» реализована в рамках классической трехзвенной архитектуры. В качестве сервера хранения используется сервер баз данных Microsoft SQL, в качестве сервера приложений используется Microsoft Internet Information Server и тонкий клиент на основе Microsoft Internet Explorer. В своей минимальной конфигурации система состоит из одного офисного компьютера

средней мощности, одного сканера и обычного офисного принтера и позволяет проводить экзамены для 2000–2500 учащихся.

Как показывает опыт применения АИС «Экзамен», система удобна для освоения и позволяет ограничить расходы при проведении тестов.

1.3 Требования к автоматизированной системе подготовки школьников к сдаче ЕГЭ по информатике

В рамках данной работы рассматривается вопрос автоматизации процесса подготовки к ЕГЭ. Традиционно, для подготовки учеников к сдаче экзамена проводятся тренировочные и контрольные тестирования в классах или во время самостоятельных занятий. Проверка результатов такого тестирования занимает значительное количество человеческих ресурсов и, кроме того, может вызывать вопросы объективности результатов контроля.

В связи с этим целесообразна автоматизация данного процесса. Реализация автоматизированной информационной системы контроля знаний учащихся призвана обеспечить надежное и быстрое тестирование знаний, максимально независимое от квалификации и индивидуальных пристрастий проверяющих. В результате принципиально меняется качество тестирования, упрощается отчетность, снижаются затраты на проведение контрольных работ и экзаменов, уменьшается зависимость оценки знаний от человеческого фактора и смягчается проблема дефицита подготовленных кадров для проверки работ. Разрабатываемая система должна дать возможность провести полный статистический анализ полученных результатов и принимать управленческие решения, направленные на повышение качества образования.

Требования к хранимой и обрабатываемой информации. На основе приведенной выше информации можно сформулировать требования к данным, которые должна позволять хранить и обрабатывать разрабатываемая система:

1. информация об учениках:
    1.1. ФИО;

1.2. класс;

1.3. пароль для входа;

2. информация о преподавателях:

2.1. ФИО;

2.2. пароль для входа;

3. информация о тестовых заданиях с кратким ответом:

3.1. содержание вопроса;

3.2. правильный ответ;

4. информация о тестовых заданиях с развернутым ответом:

4.1. содержание задачи;

5. информация о результатах тестирования:

5.1. информация об ученике, проходившем тест;

5.2. дата тестирования;

5.3. время начала;

5.4. время окончания;

5.5. ответы ученика на задачи.

Функциональные требования. Разрабатываемая информационная система должна обеспечивать следующие функциональные возможности:

1. ввод, хранение и обработка информации о тестовых задачах;
2. ввод, хранение и обработка информации об учениках;
3. проведение тренировочных тестирований для учеников;
4. проведение контрольных тестирований для учеников;
5. накопление данных о результатах тестирования для учеников;
6. формирование отчетов о результатах тестирования по классам в формате Microsoft Excel (xlsx);
7. разграничение прав доступа пользователей к информации, хранимой системой.

Требования к платформе и инструментальным средствам. Информационная система должна быть реализована в виде веб-приложения,

предоставляя доступ к хранимой информации посредством сети Интернет. Концептуальная схема работы программного продукта представлена на рис. 1.1. Должна быть обеспечена поддержка современных веб-браузеров. При разработке системы необходимо использовать открытые, свободно распространяемые инструменты и технологии.

Рис.1.1. Концептуальная схема работы программного продукта

Для работы с системой на стороне клиента необходимо наличие стандартного веб-браузера для подключения к веб-интерфейсу и почтового клиента для приема уведомлений по электронной почте.

На стороне сервера работа системы обеспечивается взаимодействием целого ряда программного обеспечения.

Основой системы является веб-сервер. Он представляет собой IBM-совместимый персональный компьютер с установленной операционной системой, и программным обеспечением веб-сервера (Apache или Nginx), хранит компоненты бизнес-уровня – PHP скрипты, изображения, сгенерированные документы и статические HTML-страницы. Он обрабатывает запросы пользователя посредством протокола HTTP [5].

Вся информация, обрабатываемая системой, хранится в базе данных, находящейся под управлением СУБД. Фактически, СУБД может быть установлена на том же сервере что и программное обеспечение веб-сервера либо, для повышения быстродействия системы, на отдельной машине. СУБД обрабатывает запросы от веб-сервера посредством стандартного драйвера. Взаимодействие веб-севера и сервера СУБД происходит на языке SQL [5].

2 ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

2.1 Формализация задачи на разработку автоматизированной системы подготовки школьников к сдаче ЕГЭ

Моделирование – это общепризнанное средство познания действительности, которое состоит из двух этапов разработки модели и ее анализа и позволяет исследовать сложные процессы и явления на основе экспериментов не с реальной системой, а ее моделью.

Методология IDEF0 используется для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающие эти функции [6].

Применительно к уже существующим системам методология IDEF0 может быть использована для анализа функций, выполняемых системой и отображения механизмов, посредством которых эти функции выполняются. В результате применения методологии IDEF0 создается модель исследуемой системы, состоящая из иерархически упорядоченного набора диаграмм, текста документации и словарей, связанных друг с другом с помощью перекрестных ссылок. Двумя наиболее важными компонентами, из которых строятся диаграммы IDEF0, являются бизнес-функции или работы (представленные на диаграммах в виде прямоугольников) и данные и объекты (изображаемые в виде стрелок), связывающие между собой работы [7].

Контекстная диаграмма является вершиной иерархической структуры диаграмм и представляет собой самое общее описание системы и ее взаимодействия с внешней средой.

На рис. 2.1 представлена модель IDEF0 «Подготовка к ЕГЭ». Для выполнения функции подготовки к сдаче ЕГЭ по предмету используются следующие входные данные: знания ученика, цели и задачи курса. Результатом являются предварительная оценка и теоретические и практические знания по

предмету. При выполнении функции руководствуются календарно-учебным планом и учебной программной. Механизмами исполнения функции являются преподаватели, ученики и соответствующие программные и аппаратные средства.

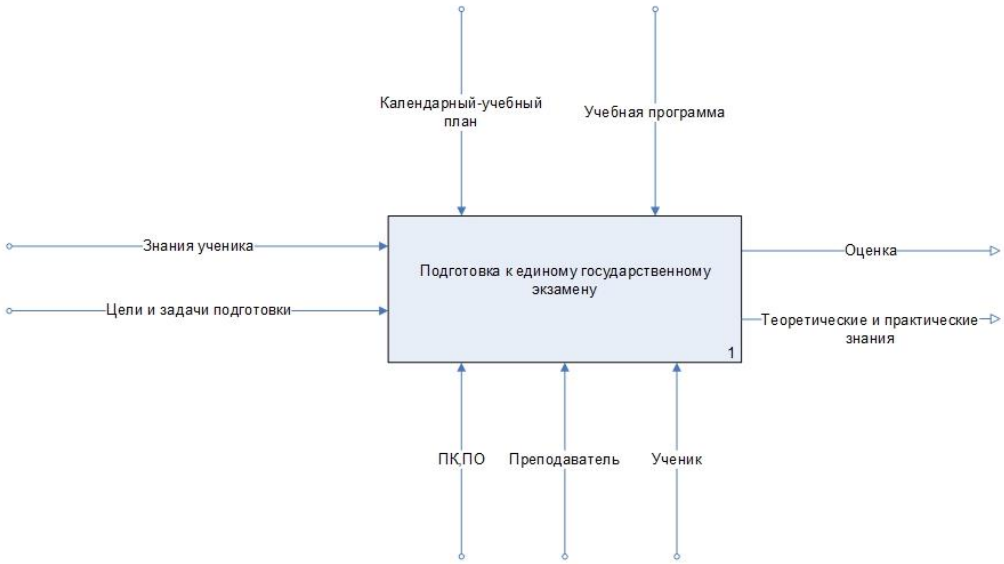


Рис. 2.1. Модель IDEF0 «Подготовка к ЕГЭ»

На основании контекстной диаграммы была разработана дочерняя диаграмма декомпозиции, представленная на рис. 2.2, содержащая функции – подготовка к сдаче ЕГЭ, уровень владения знаниями, обучение, тестирование.

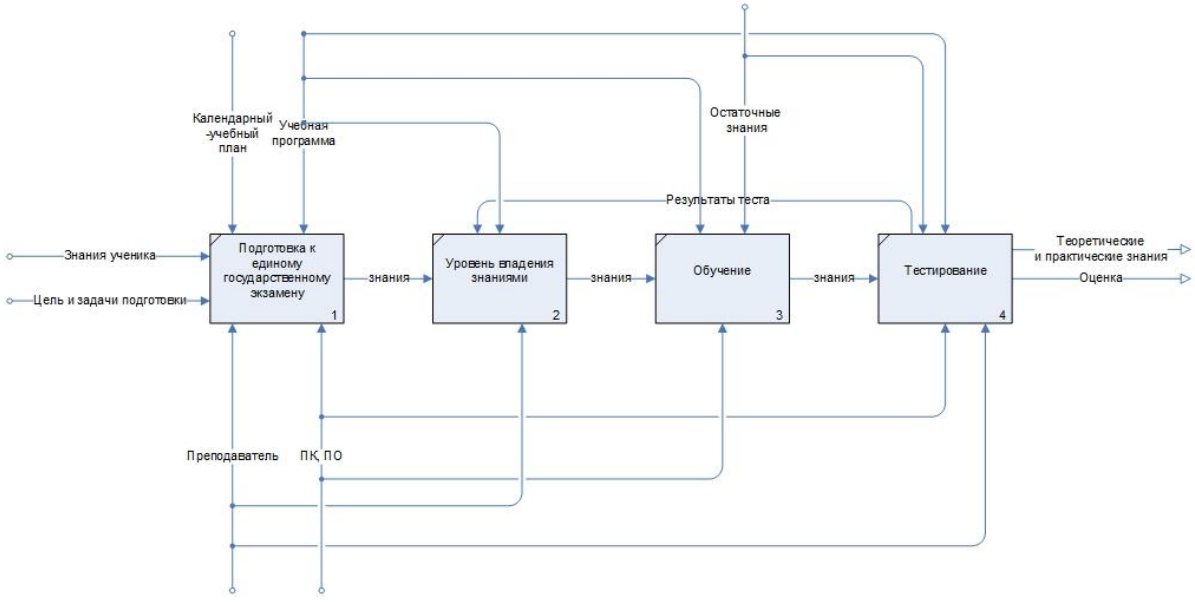


Рис.2.2. Декомпозиция процесса «Подготовка к ЕГЭ»

Данные диаграммы позволяют определить взаимосвязи функций программного обеспечения подготовки к ЕГЭ.

2.2 Проектирование базы данных

Инфологическая (концептуальная) модель – это формализованное описание предметной области, выполненное безотносительно к используемым в дальнейшем программным и техническим средствам. Инфологическая модель должная быть динамической и позволять легкую корректировку. К основным требованиями, предъявляемым к инфологической модели, можно отнести следующие: инфологическая модель должна содержать всю необходимую и достаточную информацию для последующего проектирования базы данных; инфологическая модель должная быть понятна лицам, принимающим участие в создании системы. ER-модель представляет собой логическую структуру информации об объектах системы. Компонентами ER-модели являются сущности (объекты) и отношения (связи между объектами). Объект имеет множество реализаций или экземпляров. Экземпляр объекта образуется совокупностью конкретных значений реквизитов и должен однозначно определяться, т.е. идентифицироваться значением ключа объекта, который состоит из одного ли нескольких ключевых реквизитов. Сущности могут быть зависимыми и независимыми. Сущность является независимой, если каждый экземпляр ее может быть однозначно идентифицирован без определения ее отношений с другими сущностями. Однозначная идентификация экземпляра зависимой сущности зависит от отношений с другими сущностями. Для отображения отношений между сущностями используются связи. Связи существуют, если экземпляры сущностей логически взаимосвязаны. ER-модель разработанной базы данных. В соответствии с требованиями, вся информация, обрабатываемая разрабатываемым программным изделием, должна храниться в базе данных.

В ходе анализа требований, предъявляемых к разрабатываемому программному изделию, были выявлены следующие сущности и их свойства:

1. Ученик – отражает информацию о тестируемом ученике. Свойства:
   - код ученика;
   - фамилия;
   - имя;
   - отчество;
   - код класса;
   - пароль.
2. Класс – отражает информацию о классе, в котором учится ученик. Свойства:
   - код класса;
   - номер.
3. Преподаватель – отражает информацию о преподавателе. Свойства:
   - код преподавателя;
   - фамилия;
   - имя;
   - отчество;
   - пароль.
4. Предмет – отражает информацию о предмете, по которому будет проходить тестирование. Свойства:
   - код предмета;
   - название.
5. Вопрос – отражает информацию о вопросах, из которых складывается система тестирования. Свойства:
   - код вопроса;
   - содержание;
   - правильный ответ;
   - код предмета.

6. Тестирование – отражает информацию о тестирование ученика. Свойства:

- код тестирования;
- дата;
- время начала;
- время окончания;
- баллы;
- контрольное;
- код ученика.

7. Ответ – отражает информацию об ответах на вопросы тестирования. Свойства:

- код ответа;
- код вопроса;
- код тестирования;
- дата – время.

На основе анализа сущностей и связей между ними была построена логическая модель базы данных, приведенная на рис. 2.3.

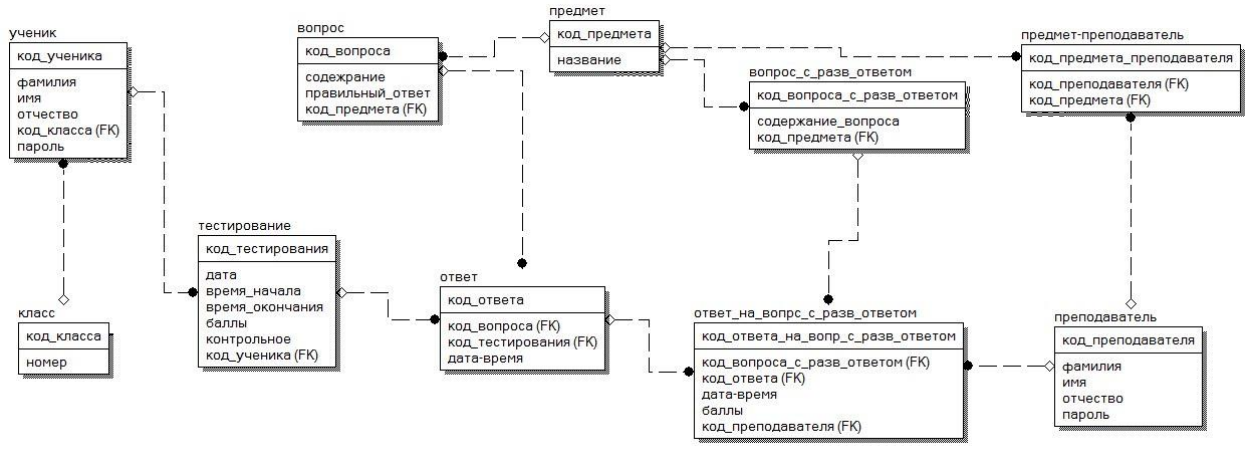


Рис. 2.3. Логическая модель БД

На основе логической модели с учетом особенностей СУБД, была разработана физическая модель базы данных, представленная на рис. 2.4.

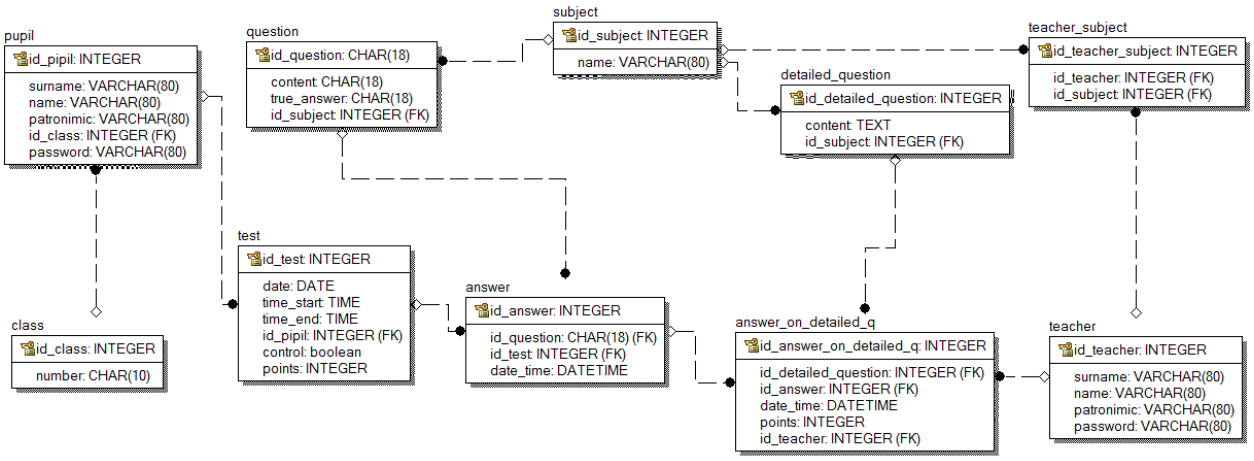


Рис. 2.4. Физическая модель БД

2.3 Выбор СУБД и средств разработки

Программное обеспечение – совокупность программ для реализации целей и задач автоматизированной системы.

Для разработки информационной системы, наиболее целесообразно использовать клиент-серверную архитектуру. Для этого необходимо выбрать язык программирования и СУБД. Для выбора языка программирования необходимо определить требования к среде программирования. В процессе анализа требований и классификации самой проектируемой ИС были определенны следующие требования к среде проектирования:

- моделирование данных;
- особенности архитектуры и функциональные возможности;
- контроль работы системы;
- особенности разработки приложений;
- производительность;
- надежность;
- требования к рабочей среде;
- смешанные критерии.

– скриптовый язык (язык сценариев).

PHP (англ. PHP: Hypertext Preprocessor – «PHP: гипертекстовый препроцессор») – скриптовый язык программирования общего назначения, интенсивно применяемый для разработки веб-приложений [8]. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков программирования, применяющихся для создания динамических веб-сайтов.

PHP обладает множеством преимуществ по сравнению с конкурирующими продуктами [9]:

– высокая производительность;

– наличие интерфейсов к различным системам баз данных;

– встроенные библиотеки для выполнения многих общих задач, связанных с Web;

– свободное распространение;

– простота изучения и использования;

– кроссплатформенность;

– доступность исходного кода.

PHP обладает встроенной связностью со многими системами баз данных: MySQL, PostgreSQL, mSQL, Oracle, dbm, Hyperware, Informix, InterBase и Sybase. Используя Open Database Connectivity Standard (Стандарт открытого интерфейса связи с базами данных, ODBC), можно подключаться к любой базе данных, для которых существует ODBC-драйвер [10].

Требования к СУБД:

– кроссплатформенность СУБД;

– полная совместимость с выбранной средой разработки (PHP);

– простота использования и внедрения;

– распространенность и популярность СУБД. При использовании малораспространенной СУБД в будущем могут возникнуть проблемы с

поддержкой и развитием ИС, что так же наложит дополнительные затраты на перенос накопленных данных.

– Надежность, позволяющая обеспечить решение задачи постоянного наполнения и обеспечения сохранности данных.

Учитывая приведенные выше требования и сравнительный обзор, в качестве СУБД был выбрана среда MySQL – это многопоточная, многопользовательская СУБД, основными достоинствами которой является быстрота, надежность и простота использования. Несмотря на то, что MySQL не представляет такой широкий набор возможностей, как например Oracle, использование MySQL оправдывается из-за значительных меньших требований к мощности оборудования и большей скоростью работы при работе со средним объемом данных БД. Использование связки PHP+MySQL позволяет более гибко организовывать принцип мультиплатформенности.

3. РАЗРАБОТКА И ТЕСТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

3.1. Реализация базовых компонентов системы

В качестве основной схемы проектирования приложения был выбран шаблон MVC (model-controller-view). Шаблон проектирования MVC предполагает разделение данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: Модель, Представление и Контроллер.

В приведённом определении под компонентом следует понимать некую отдельную часть кода, каждая из которых играет одну из ролей Контроллера, Модели или Представления, где Модель служит для извлечения и манипуляций данными приложения, Представление отвечает за видимое пользователю отображение этих данных.

На рис. 3.1 приведена схема взаимодействия основных компонентов приложения, построенного по шаблону MVC.

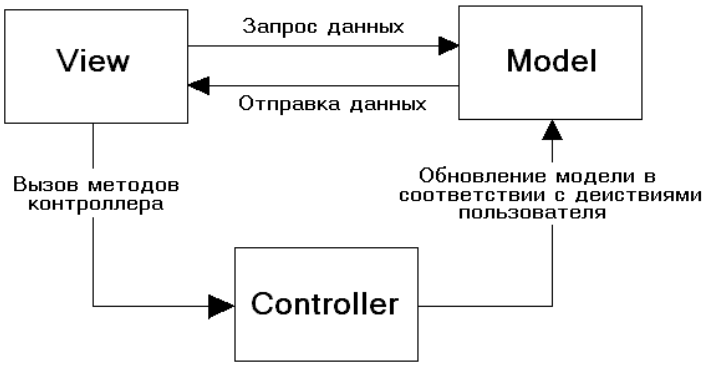


Рис.3.1. Шаблон MVC

В настоящее время при разработке типовых решений на языке PHP хорошей практикой считается использование фреймворков.

Существует большое количество различных PHP фреймворков, в основе которых лежит шаблон MVC. Наиболее распространенные из них ZendFramework, Yii, Symphony, Codeigniter.

В качестве основы для создания разрабатываемого приложения был выбран фреймворк Codeigniter, как наиболее простой и удобный с точки зрения автора. На момент написания данной работы актуальной версией данного программного продукта была 3.0. Данный фреймворк является свободно распространяемым и предоставляет удобные компоненты для решения задач работы с базой данных, сессиями, разграничения прав пользователя.

В соответствии с требованиями подхода MVC была разработана модульная структура приложения, приведенная на рис. 3.2.

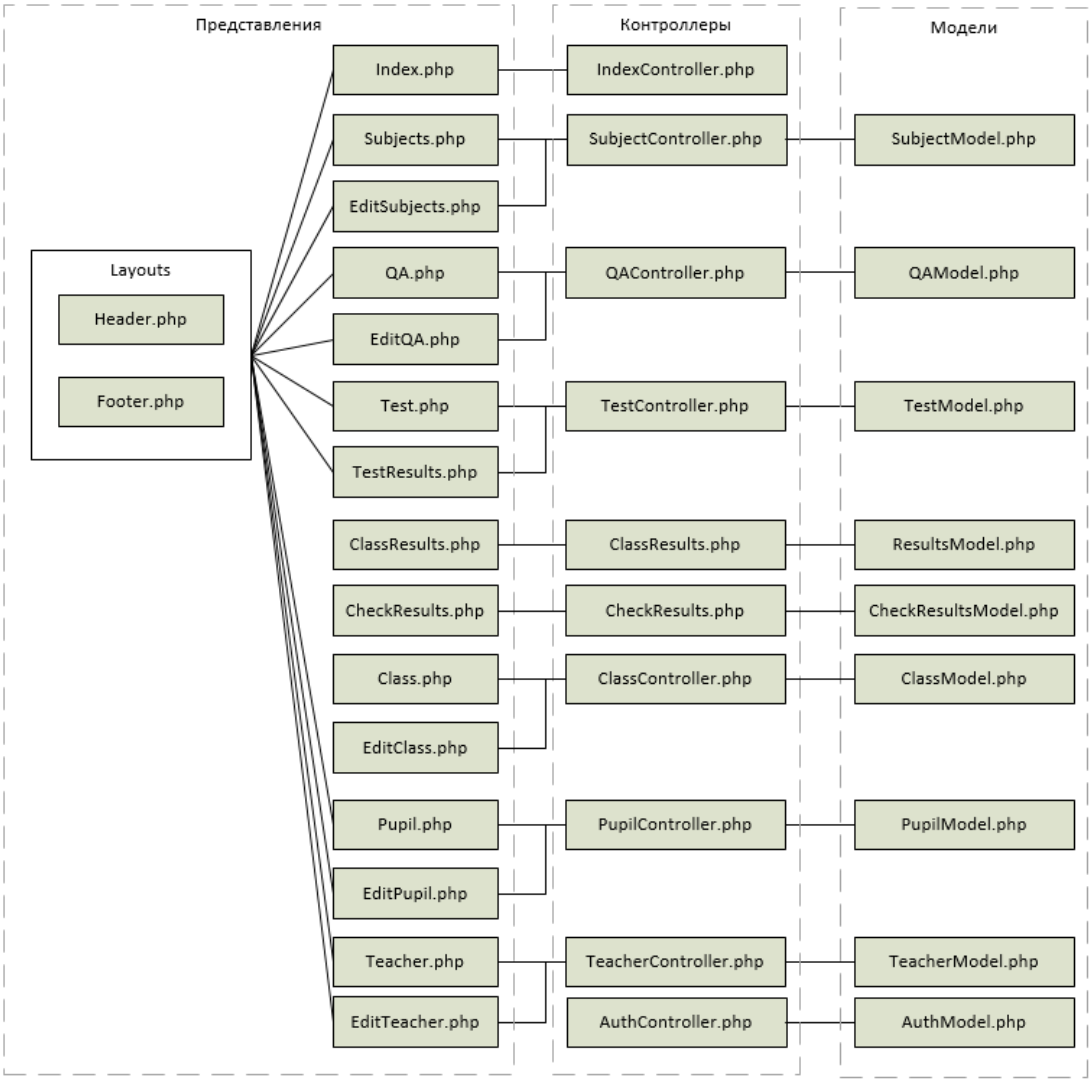


Рис. 3.2. Модульная архитектура приложения

Контроллер SubjectController отвечает за отображение списка предметов (представление Subjects) и отображение формы добавления / редактирования информации о предмете – представление EditSubject. Он использует модель SubjectModel, которая содержит методы для выбора, удаления и редактирования информации о предметах в базе данных.

Контроллер QAController отвечает за отображение списка вопросов и ответов для предмета (представление QA) и формы добавления / редактирования информации о вопросах и ответах – представление EditAQ. Он использует модель QAModel, которая содержит методы для выбора, удаления и редактирования информации о вопросах и ответах в базе данных.

Контроллер TestController отвечает за отображение формы тестирования ученика (представление Test) и формы результатов тестирования – представление TestResults. Он использует модель TestModel, которая содержит методы для выбора, удаления и редактирования ответов учеников на вопросы в процессе тестирования.

Контроллер ClassResultsController отвечает за отображение результатов тестирования по классу (представление Service). Он использует модель ResultsModel, которая содержит методы для выбора информации о результатах тестирования.

Контроллер CheckResultsController отвечает за отображение формы проверки ответов на вопросы с развернутым ответом (представление CheckResults). Он использует модель CheckResultsModel, которая содержит методы для выбора, удаления и редактирования информации об ответах учеников на вопросы с развернутым ответом.

Контроллер ClassController отвечает за отображение списка классов (представление Class) и формы редактирования информации о классах EditClass. Он использует модель ClassModel, которая содержит методы для выбора, удаления и редактирования информации о классах.

Контроллер PupilController отвечает за отображение списка учеников (представление Pupil) и формы добавления / редактирования информации

обучениках – представление EditPupil. Он использует модель PupilModel, которая содержит методы для выбора, удаления и редактирования информации об учениках.

Контроллер TeacherController отвечает за отображение списка учителей (представление Teacher) и формы добавления / редактирования информации об учителях – представление EditTeacher. Он использует модель TeacherModel, которая содержит методы для выбора, удаления и редактирования информации об учителях.

Контроллер AuthController отвечает реализацию механизмов авторизации. Он содержит методы проверки подлинности введенных логина и пароля, проверки, авторизован ли текущий пользователь и метод для выхода из системы. Он использует модель AuthModel, которая содержит методы для обращения к данным авторизации в базе данных.

Механизм авторизации. Для реализации механизма авторизации и проверки прав пользователей используется механизм сессий. Сессии - это специальный механизм, который позволяет обеспечить передачу данных от одной страницы к другой. Протокол HTTP не имеет памяти и у него все запросы с сервера, трактуются как отдельные, не связанные между собой.

Есть несколько вариантов хранения сессий:

– сессии хранятся в специальной временной папке на сервере;

– данные сохраняться в куки, но через интерфейс, такой же, как и сессии;

– в созданном временном хранилище в базе данных.

Сессия, в понимании CodeIgniter, это просто массив, содержащий следующую информацию:

– уникальный идентификатор пользователя;

– ip-адрес пользователя;

– данные User Agent;

– timestamp "последней активности" (timestamp — время в формате unix time).

Данные выше сохраняются в куки в виде сериализованного массива по прототипу, представленного в листинге 3.1.

Листинг 3.1. Пример сохранения данных в виде сериализованного массива

```
[array]
(
     'session_id'    => random hash,
     'ip_address'    => 'string – user IP address',
     'user_agent'    => 'string – user agent data',
'last_activity' => timestamp
)
```

Конец листинга

Элементы массива сессии доступны к использованию через функцию, представленную в листинге 3.2.

Листинг 3.2. Определение доступности элементов массива

```
$this->session->userdata('item');
```

Конец листинга

Где `item` - это индекс массива, соответствующий элементу, который необходимо получить. Например, для того, чтобы получить ID сессии, необходимо выполнить код, представленный в листинге 3.3.

Листинг 3.3. Пример получения ID сессии

```
$session_id = $this->session->userdata('session_id');
```

Конец листинга

Для того чтобы добавить данные в массив сессии, необходимо передать массив содержащий данные в функцию, представленную в листинге 3.4.

Листинг 3.4. Пример добавления данных в массив сессии

```
$this->session->set_userdata($array);
```

Конец листинга

Где `array` - это ассоциативный массив, содержащий новые данные. Код массива представлен в листинге 3.5.

Листинг 3.5. Пример кода ассоциативного массива array

```
$newdata = array(
   'username'  => 'johndoe',
   'email'     => 'johndoe@some-site.com',
   'logged_in' => TRUE
);
$this->session->set_userdata($newdata);
```

Конец листинга

Если необходимо удалить данные из сессии, можно использовать функцию, представленную в листинге 3.6.

Листинг 3.6. Функция для удаления данных из сессии

```
$this->session->unset_userdata('some_name');
```

Конец листинга

Механизм работы с базой данных. Для обработки информации, хранящейся в базе данных в рамках фремворка Codeigneter, используются модели. Модели - это PHP классы, которые предназначены для работы с информацией в базе данных.

Классы моделей хранятся в папке application/models/. Они могут быть вложенными в подкаталоги. Базовый прототип для модели класса представлен в листинге 3.7.

Листинг 3.7. Базовый прототип для модели класса

```
class Model_name extends CI_Model {
     public function __construct()
     {
          parent::__construct();
          }
     }
}
```

Конец листинга

Где Model_name - имя класса. Имена классов должны иметь первую букву заглавную, а остальные в нижнем регистре. Классы должны расширять основной класс модели CI_Model.

Модели, как правило, используются из контроллера. Чтобы загрузить модель следует использовать метод, представленный в листинге 3.8.

Листинг 3.8. Метод для загрузки модели из контроллера

```
$this->load->model('model_name');
```

Конец листинга

В листинге 3.9 приведен фрагмент контроллера, который загружает модель, затем передает полученные данные в представление на примере отображения списка клиентов.

Листинг 3.9. Фрагмент контроллера для загрузки модели

```
class ClientsController extends CI_Controller {
     public function index() {
          $this->load->model('ClientModel');
          $data['query'] = $this->ClientModel->get_all();
          $this->load->view('Clients', $data);
     }
}
```

Конец листинга

Среда разработки. Для упрощения процесса работки веб-приложений в настоящее время используются интегрированные среды разработки (IDE) – комплекс программных средств, используемый для разработки программного обеспечения (ПО).

Среда разработки включает в себя:

– текстовый редактор;
– компилятор и/или интерпретатор;
– средства автоматизации сборки;
– отладчик.

Иногда IDE содержит также средства для интеграции с системами управления версиями и разнообразные инструменты для упрощения конструирования графического интерфейса пользователя.

В области веб-разработки основными конкурентами на рынке являются PhpStormIDE, NetBeans и Eclipse PDT. Все они имеют различные достоинства и недостатки. В качестве среды для создания разрабатываемого приложения была выбрана IDEPhpStorm.

PhpStorm представляет собой интеллектуальный редактор для PHP, HTML и JavaScript с возможностями анализа кода на лету, предотвращения ошибок в коде и автоматизированными средствами рефакторинга для PHP и JavaScript. Автодополнение кода в PhpStorm поддерживает спецификацию PHP 5.3, 5.4, 5.5 и 5.6, включая генераторы, пространства имен, замыкания и синтаксис коротких массивов. На рис. 3.3 приведен внешний вид среды разработки.

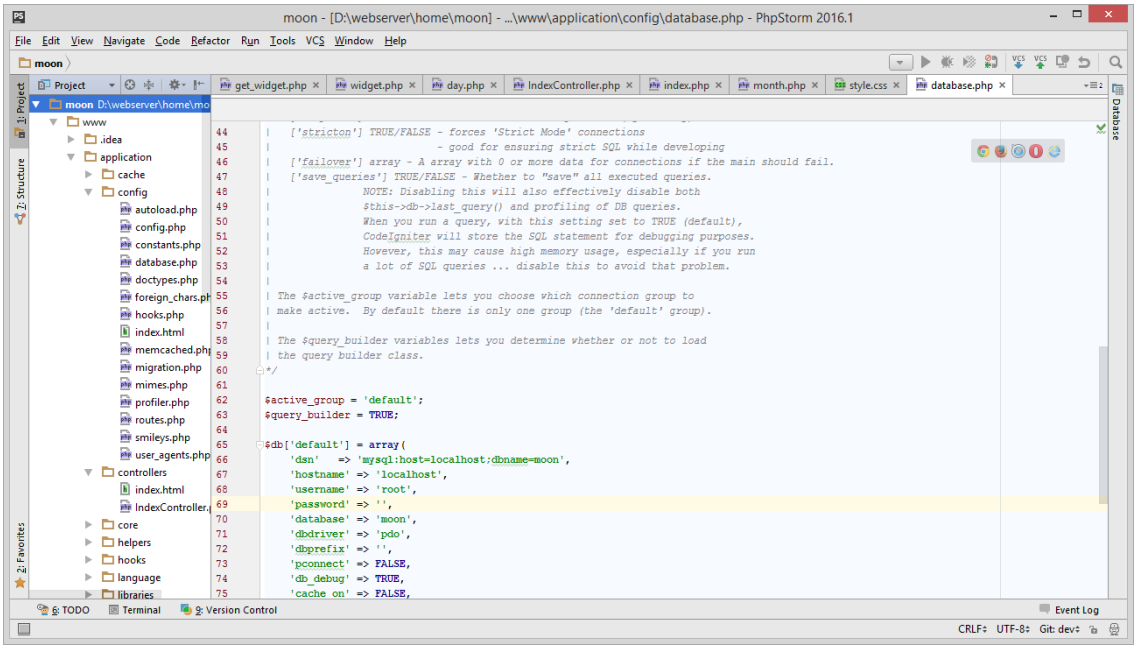


Рис. 3.3. Внешний вид IDEPHPStorm

Имеется полноценный SQL-редактор с возможностью редактирования полученных результатов запросов.

3.2. Реализация пользовательского интерфейса автоматизированной системы

При написании программы немаловажную роль играет разработка и реализация интерфейса пользователя, именно от него зависит насколько удобной будет данная система.

Интерфейс пользователя, эта часть программы, отвечающая за взаимодействие между основным «ядром» и конечным пользователем, то есть выполняет чисто коммуникативную функцию.

На основе требований к системе документооборота интернет агентства были выделены основные экраны пользовательского интерфейса:

1. главная страница (форма входа);
2. форма тестирования;
3. форма отображения результатов тестирования;

4. список предметов;
5. форма редактирования информации о предмете;
6. список вопросов;
7. форма редактирования информации о вопросе и ответах;
8. форма результатов для класса;
9. форма проверки ответов на вопросы с развернутым ответом;
10. список классов;
11. форма редактирования информации о классе;
12. список учеников;
13. форма редактирования информации об учениках;
14. список учителей;
15. форма редактирования информации об учителях.

Взаимосвязь между основными экранами разрабатываемой системы показана на рис. 3.4.

Для создания пользовательского интерфейса веб-приложений в настоящее время активно применяются различные фреймворки [11]. Наиболее простым в использовании и распространённым является Bootstrap. Bootstrap (также известен как Twitter Bootstrap) – свободный набор инструментов для создания сайтов и веб-приложений. Включает в себя HTML и CSS шаблоны оформления для типографики, веб-форм, кнопок, меток, блоков навигации и других компонентов веб-интерфейса, включая JavaScript-расширения.

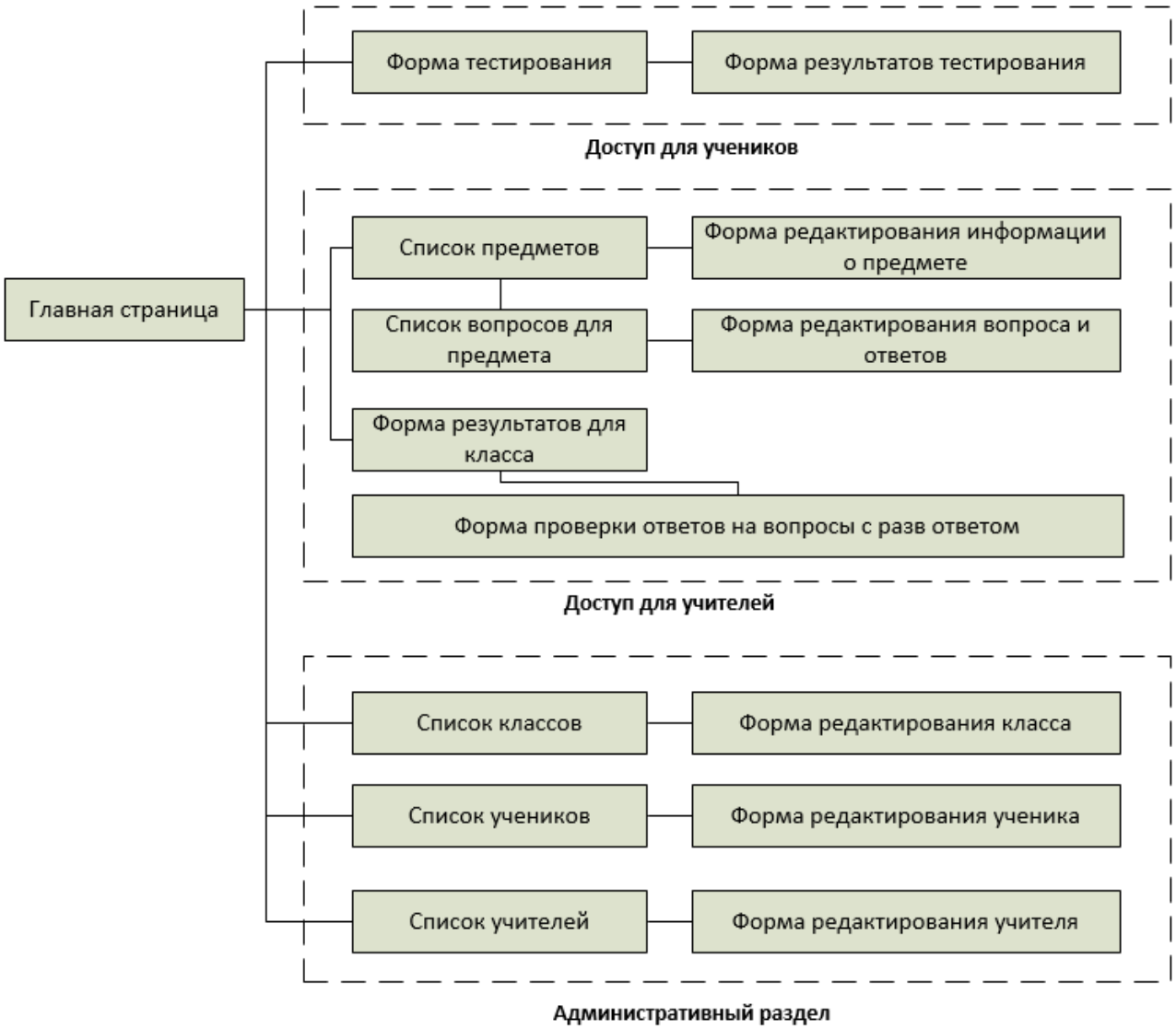


Рис.3.4. Схема взаимодействия основных экранов пользовательского интерфейса

Основные преимущества Bootstrap:

– экономия времени – Bootstrap позволяет сэкономить время и усилия, используя шаблоны дизайна и классы, и сконцентрироваться на этапах разработки;

– адаптированность под разные платформы - динамичные макеты Bootstrap масштабируются на разные устройства и разрешения экрана без каких-либо изменений в разметке;

– гармоничный дизайн — все компоненты платформы Bootstrap используют единый стиль и шаблоны с помощью центральной библиотеки. Дизайн и макеты веб-страниц согласуются друг с другом;

– простота в использовании — платформа проста в использовании, пользователь с базовыми знаниями HTML и CSS может начать разработку с Twitter Bootstrap;

– совместимость с браузерами — Twitter Bootstrap совместим с Mozilla Firefox, Yandex Browser, Google Chrome, Safari, Internet Explorer, Microsoft Edge и Opera;

– открытое программное обеспечение — особенность Twitter Bootstrap, которая предполагает удобство использования, посредством открытости исходных кодов и бесплатной загрузки.

В настоящее время существует большое количество инструментов для визуального проектирования интерфейсов на основе Bootstrap. визуального проектирования интерфейсов на основе Bootstrap. Такие инструменты доступ в виде отдельных windows-приложений, компонентов для IDE или online-редакторов [12].

В рамках данной работы использовался бесплатный онлайн-инструмент jetstrap.com. Результатом проектирования является исходный код html, который можно использовать в готовом проекте.

На рис. 3.5 приведен процесс визуального проектирования главной страницы приложения в онлайн-редакторе.

Главная страница состоит из заголовка страницы и формы входа. После авторизации пользователь попадет в основное приложение, в котором ему будут доступны весь функционал.

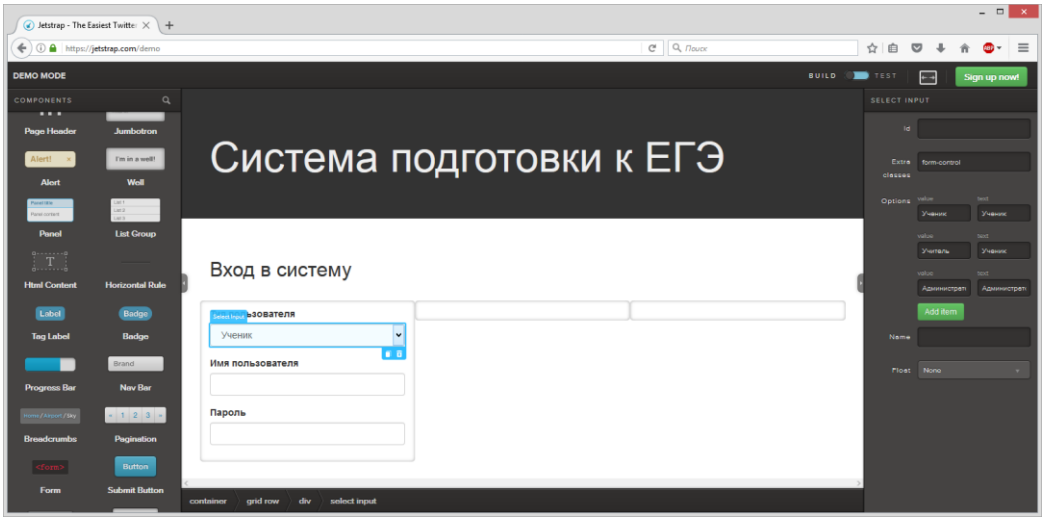


Рис. 3.5. Процесс создания интерфейса главной страницы

Интерфейс ученика содержит два пункта в главном меню: «Тестирования» и «Новое тестирование».

На рис. 3.6 приведен процесс визуального проектирования экрана «Тестирования» интерфейса ученика приложения в онлайн-редакторе.

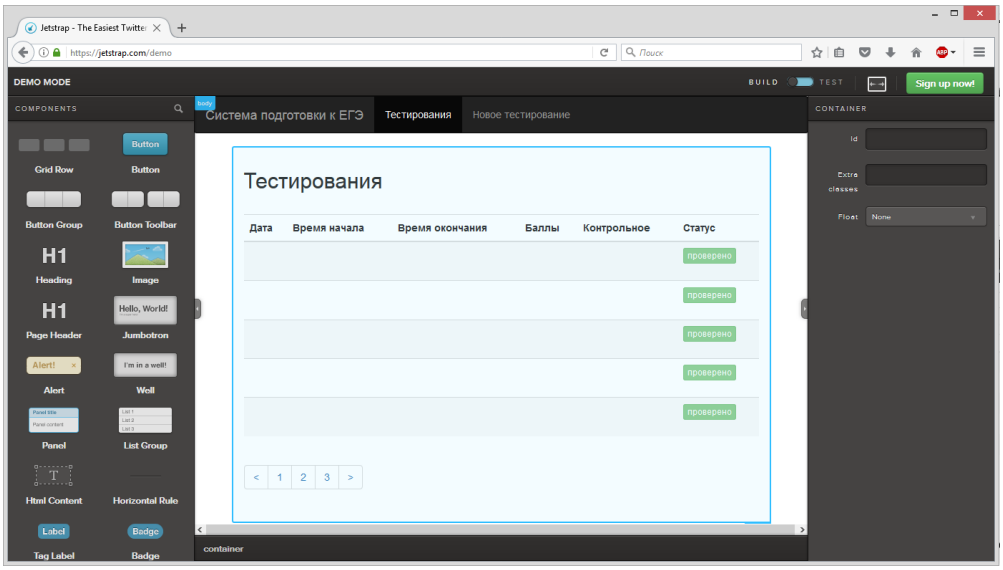


Рис. 3.6. Процесс создания интерфейса ученика, экран «Список тестирований»

На экране «Тестирования» ученик имеет возможность просмотреть результаты всех ранее пройденных тестов. Дату теста, время начала тестирования, время окончания, количество баллов и статус теста (проверен ли он преподавателем).

На рис. 3.7 представлен процесс визуального проектирования экрана прохождения тестирования для интерфейса ученика.

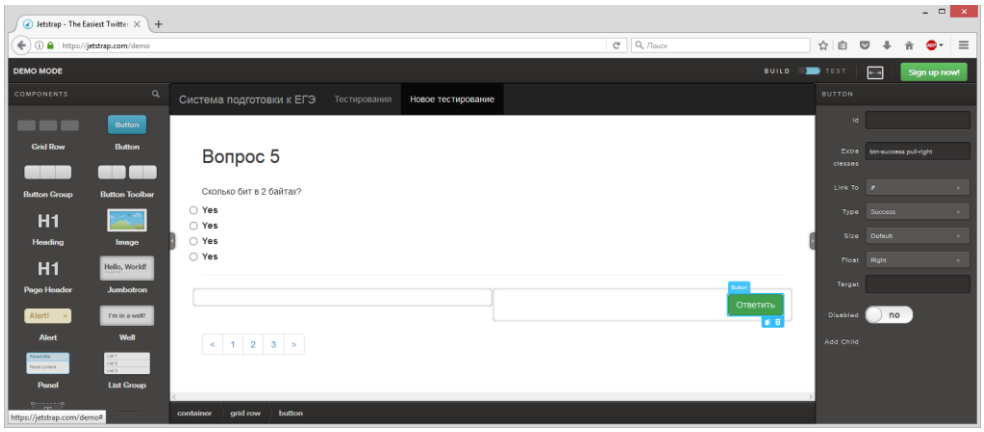


Рис.3.7. Процесс создания интерфейса ученика, экран тестирования

Интерфейс учителя в главном меню содержит три пункта: «Результаты тестирования», «Вопросы» и «Предметы».

На рис. 3.8 представлен процесс визуального проектирования интерфейса учителя.

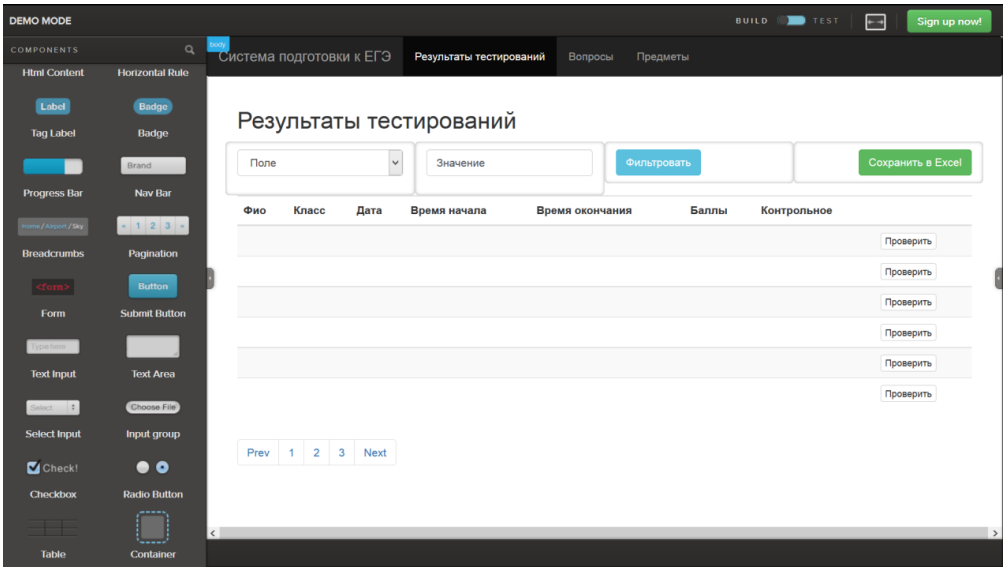


Рис. 3.8. Процесс создания интерфейса учителя, результаты тестирования для класса

На рис. 3.9 представлен процесс создания экрана «Вопросы» интерфейса учителя.

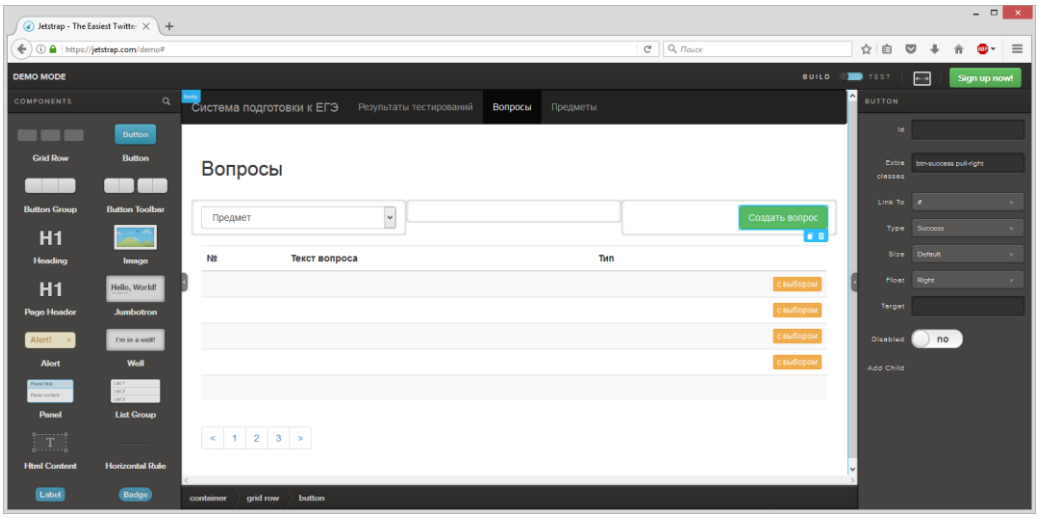


Рис. 3.9. Процесс создания интерфейса учителя – список вопросов

Для добавления и редактирования информации используются стандартные формы Bootstrap. Процесс визуального проектирования формы редактирования вопроса показан на рис. 3.10.

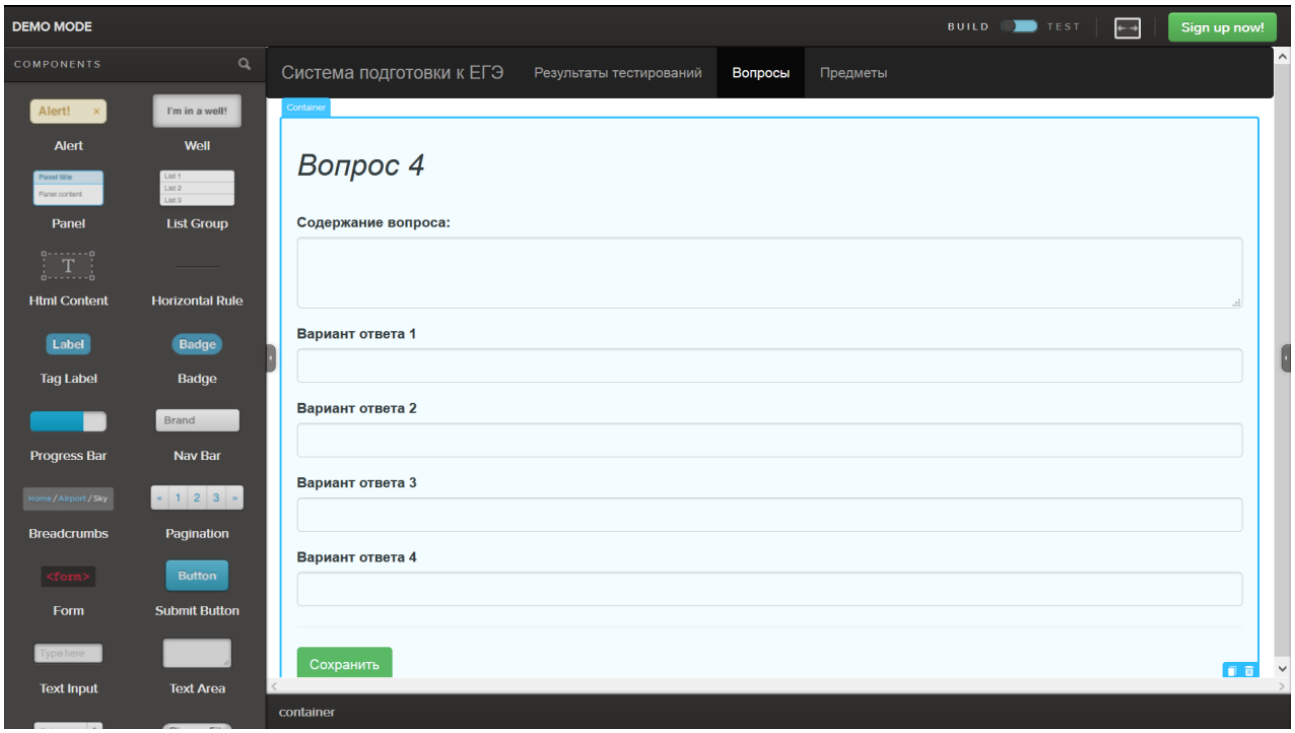


Рис. 3.10. Процесс создания интерфейса учителя – форма редактирования/добавления вопроса

На рис. 3.11 показан процесс создания экрана «Предметы» интерфейса учителя.

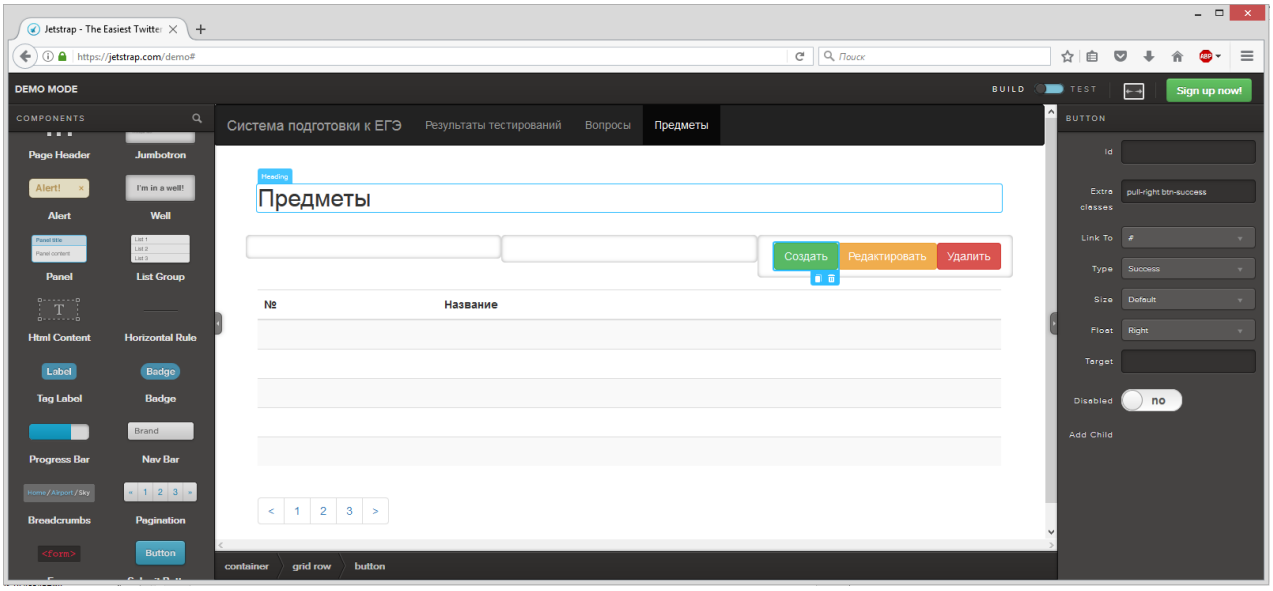


Рис. 3.11. Процесс создания интерфейса учителя – список предметов

На рис. 3.12 показан процесс визуального проектирования формы добавления и редактирования информации о предмете интерфейса учителя.

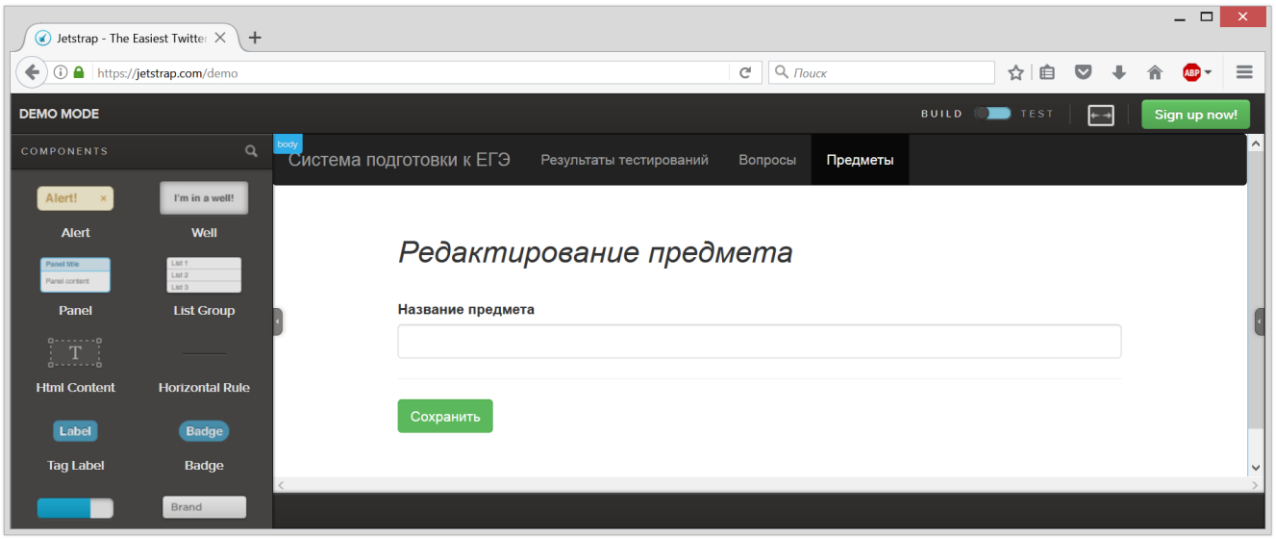


Рис. 3.12. Процесс создания интерфейса учителя – форма добавления/редактирования предмета

Интерфейс администратора создавался аналогичным образом, были спроектированы формы добавления, удаления и редактирования информации о

классах, учениках и преподавателях. Администратор в верхнем меню имеет три пункта «Классы», «Ученики» и «Учителя».

На рис. 3.13 показан процесс визуального проектирования формы отображения списка классов.

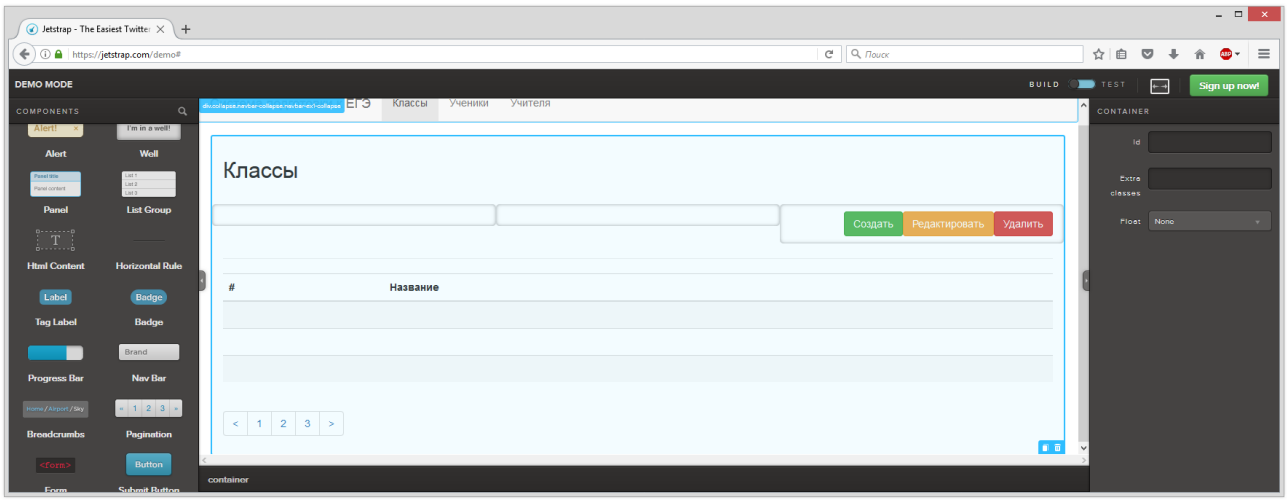


Рис. 3.13. Процесс создания интерфейса администратора – список классов

Для перехода на форму редактирования информации о классе или добавления нового классе необходимо нажать на кнопку «Редактировать» или «Создать» соответственно. На рис. 3.14 показан процесс визуального проектирования формы редактирования информации о классе.

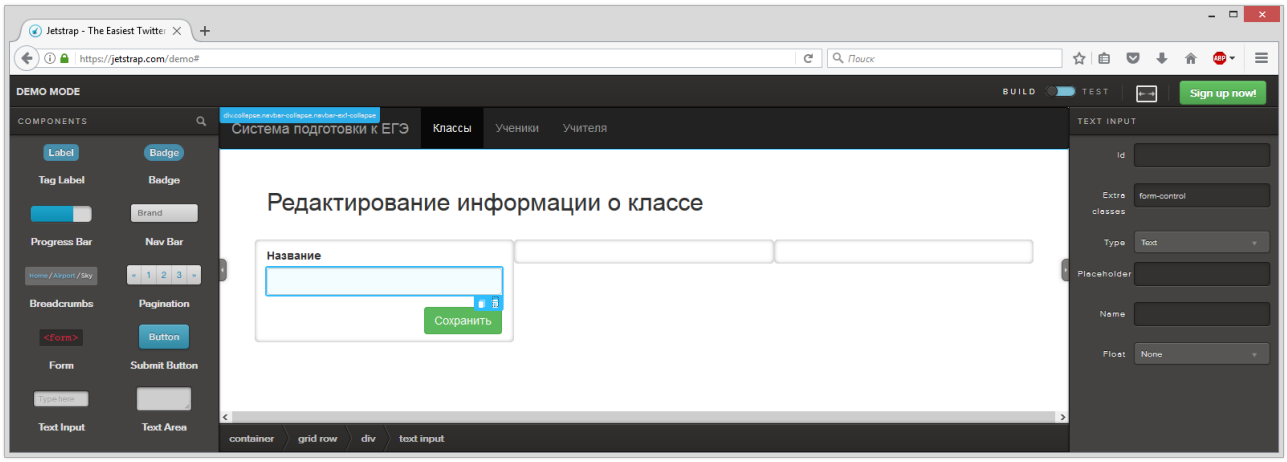


Рис. 3.14. Процесс создания интерфейса администратора – список классов

На вкладке «Ученики» администратор имеет возможность просмотреть список учеников, которые имеют доступ к системе, добавить нового ученика или отредактировать информацию о уже имеющемся ученике.

На рис. 3.15 показан процесс визуального проектирования формы со списком учеников.

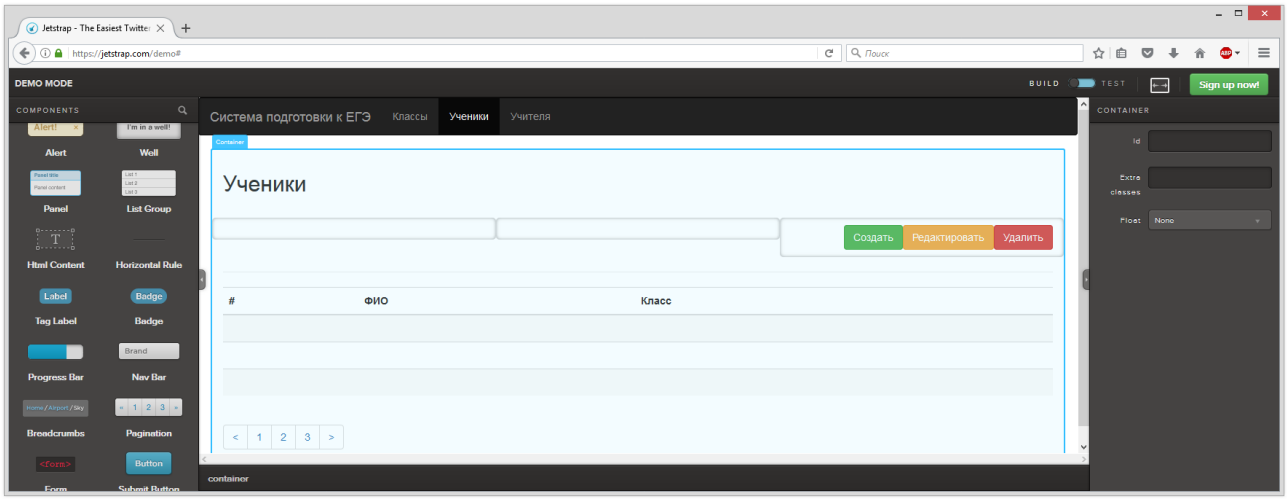


Рис. 3.15. Процесс создания интерфейса администратора – список учеников

Для перехода на форму редактирования информации об ученике или добавления нового ученика необходимо нажать на кнопку «Редактировать» или «Создать» соответственно. На рис. 3.16 показан процесс визуального проектирования формы редактирования информации об ученике.

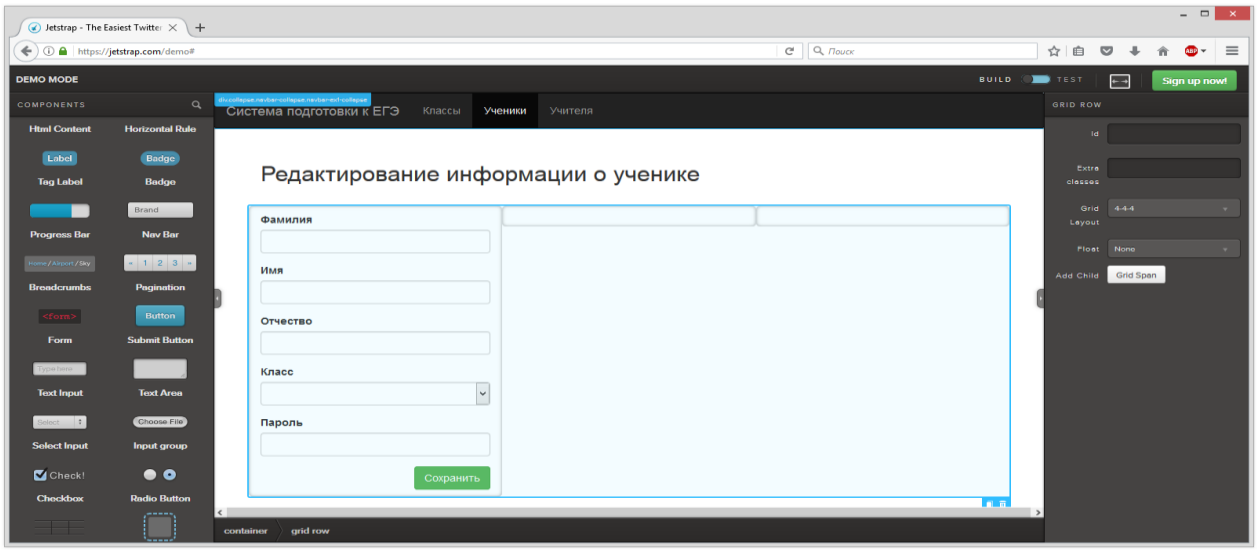


Рис. 3.16. Процесс создания интерфейса администратора – форма добавления и редактирования информации об ученике

На вкладке «Учителя» администратор имеет возможность просмотреть список учителей, зарегистрированных в системе. На рис. 3.17 показан процесс создания интерфейса этого списка.

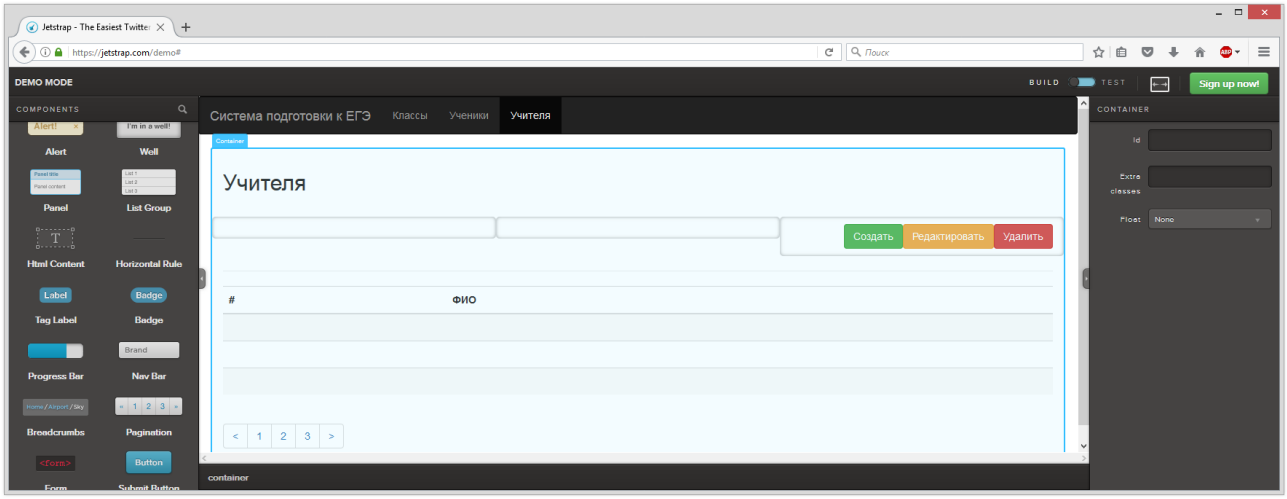


Рис. 3.17. Процесс создания интерфейса администратора – список учителей

Для перехода на форму редактирования информации об учителе или добавления нового учителя необходимо нажать на кнопку «Редактировать» или «Создать» соответственно. На рис. 3.18 показан процесс визуального проектирования формы редактирования информации об учителе.

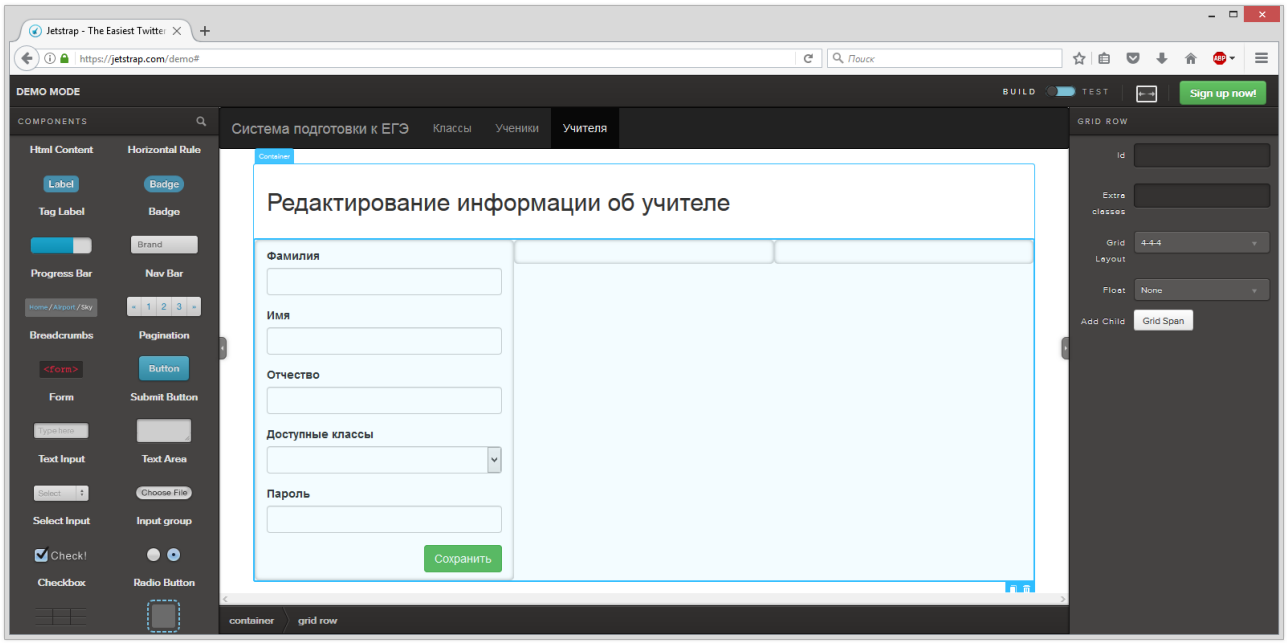


Рис. 3.18. Процесс создания интерфейса администратора – форма добавления и редактирования информации об учителе

3.3. Тестирование автоматизированной системы

Покажем основные этапы работы с созданным приложением. Все пользователи приложения делятся на три группы: администратор, учителя и ученики. Каждый из пользователей имеет доступ к ограниченному набору функций в соответствии с группой, к которой он принадлежит. На главной странице приложения расположена форма авторизации. Пользователь должен выбрать свою группу из выпадающего списка и ввести имя пользователя и пароль.

На рис. 3.19 показан внешний вид главной страницы приложения.

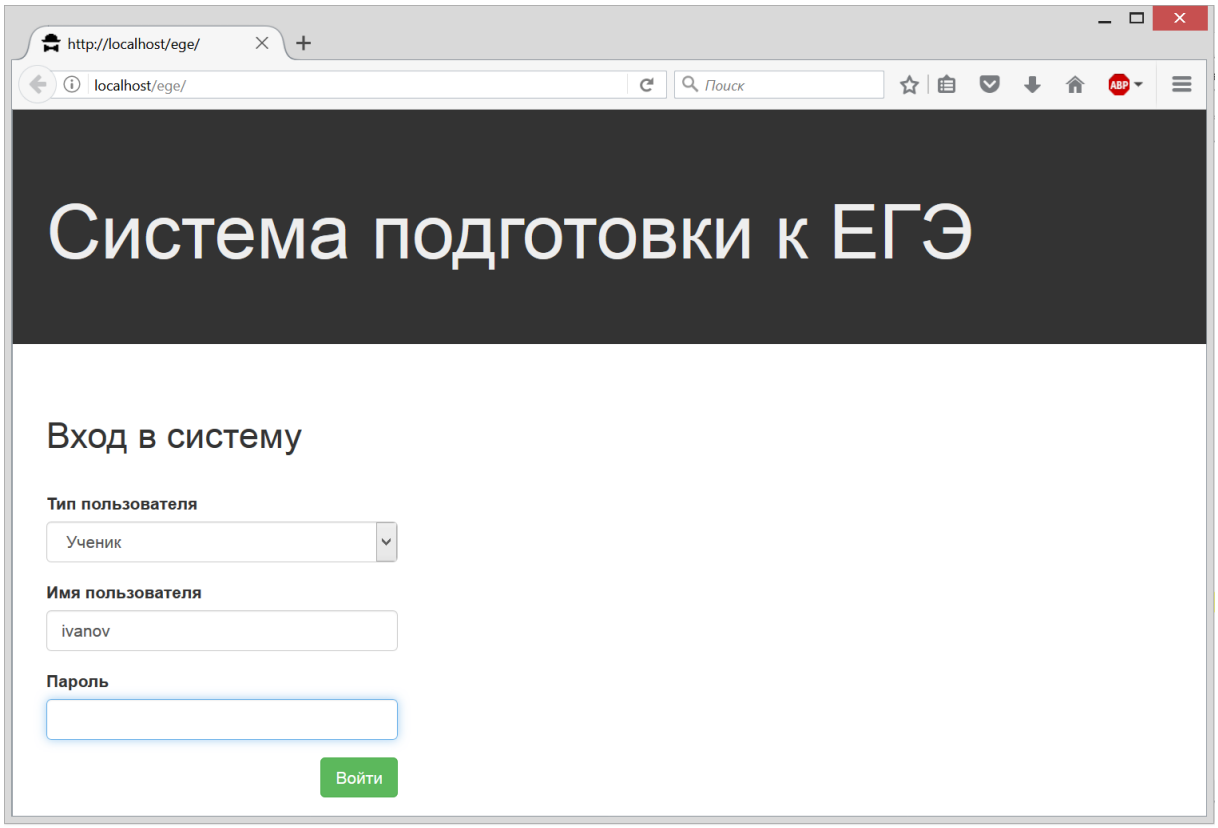


Рис.3.19. Вход в систему

После авторизации пользователь группы «Ученик» попадает на страницу личного кабинета ученика. Интерфейс ученика содержит два пункта в главном меню: «Тестирования» и «Новое тестирование».

На экране «Тестирования», представленном на рис. 3.20, ученик имеет возможность просмотреть результаты всех ранее пройденных тестов, дату теста, время начала тестирования, время окончания, количество баллов и статус теста (проверен ли он преподавателем).

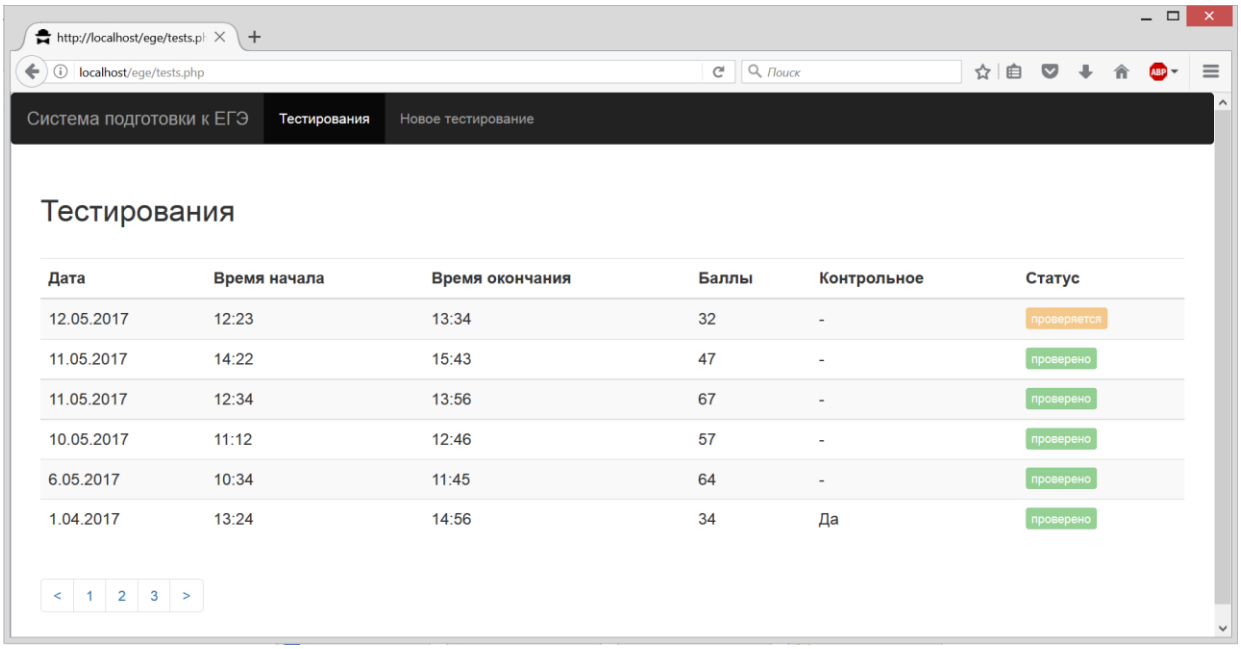


Рис.3.20. Интерфейс ученика – список тестирований

На экране «Новое тестирование», представленном на рис. 3.21, ученик может пройти новый тест, ответив на вопросы.

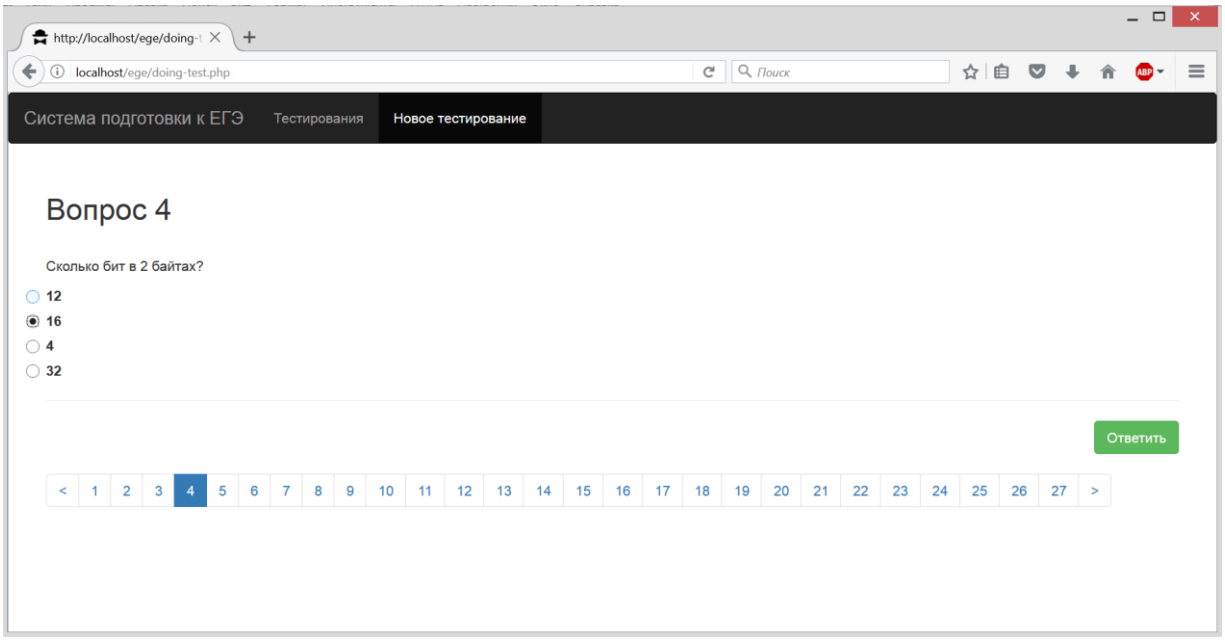


Рис. 3.21. Интерфейс ученика – прохождение тестирования

Интерфейс учителя в главном меню содержит три пункта: «Результаты тестирования», «Вопросы» и «Предметы». На экране «Результаты тестирования» учитель имеет возможность просмотреть результаты тестирований для класса и перейти к проверке вопросов с развернутым ответом. Интерфейс учителя представлен на рис. 3.22.

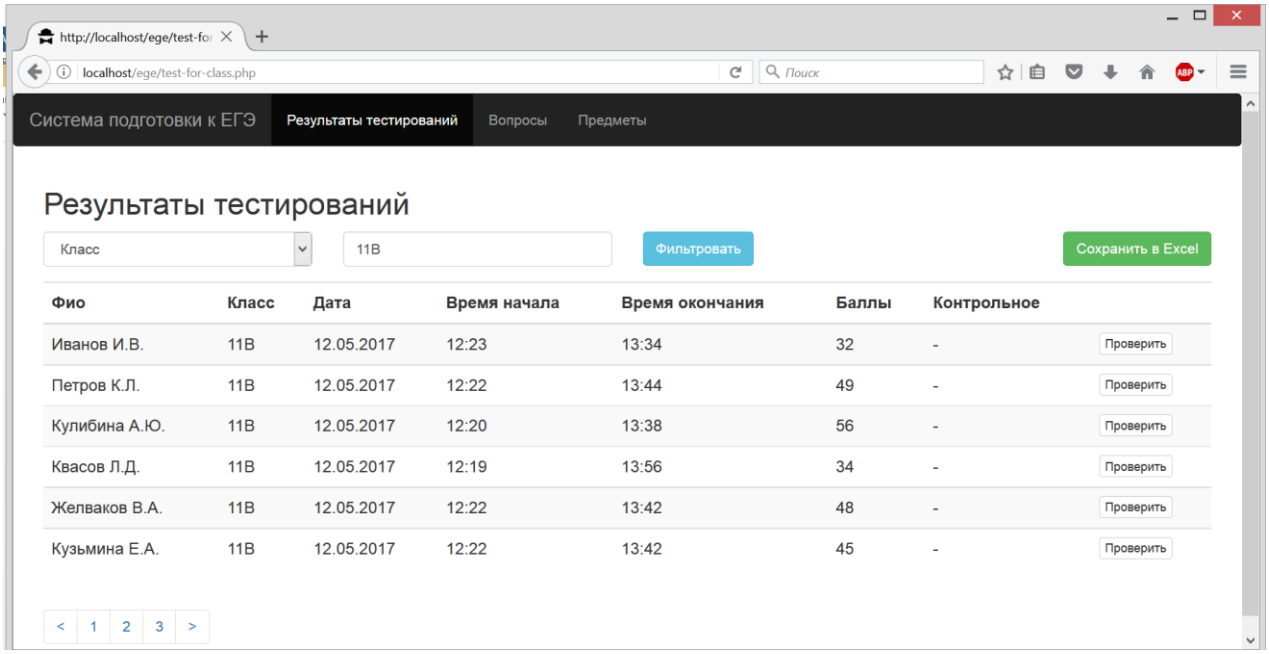


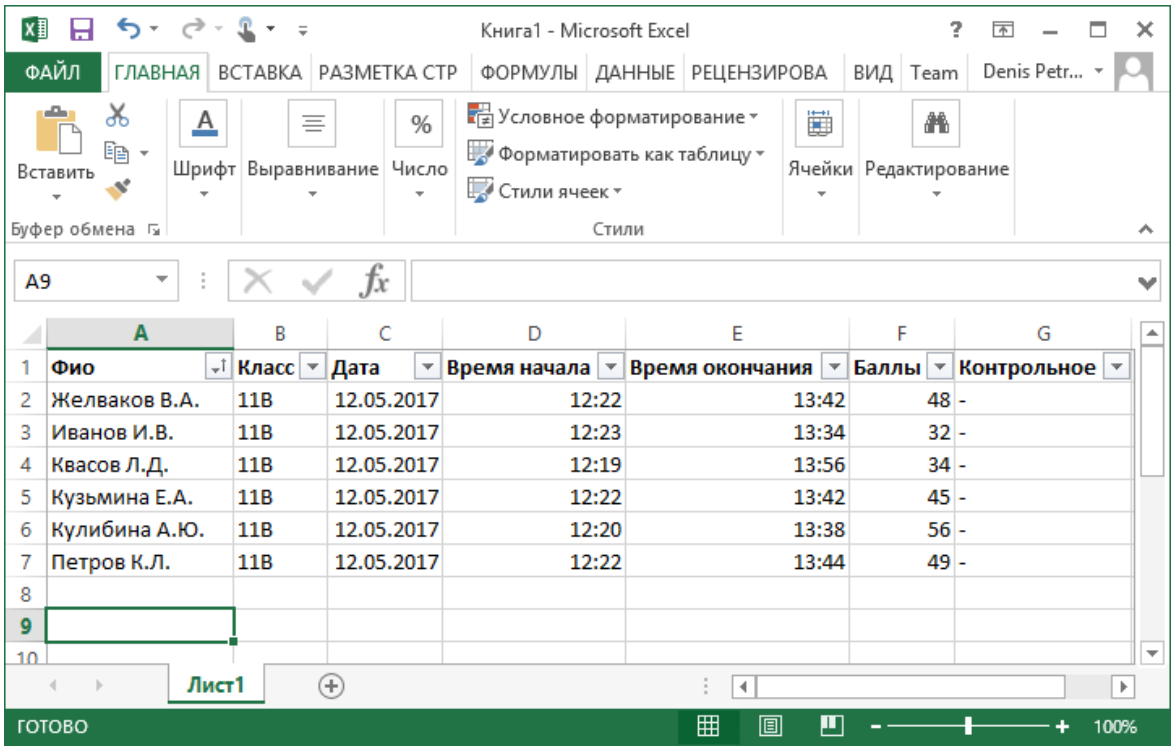
Рис.3.22. Интерфейс учителя – список результатов тестирований

В верхней части страницы расположена форма, позволяющая производить поиск и фильтрацию информации о результатах тестов. Для фильтрации списка результатов необходимо в выпадающем списке выбрать поле, по которому будет произведен поиск, в текстовом поле ввести ключевое слово и нажать кнопку «Фильтровать».

Справа вверху расположена кнопка «Сохранить в Excel», которая позволяет произвести экспорт выбранных данных в документ формата xls. Данная реализация представлена на рис. 3.23.

В отчете сохраняется текущий список учеников, результаты прохождения ими тестов и организуется автофильтр по полям.

Данный автофильтр позволяет уже с помощью интерфейса MSExcel производить сортировку и фильтрацию данных по любому из представленных в отчете полей. Отчет так же можно вывести на печать.



	A	B	C	D	E	F	G
1	Фио	Класс	Дата	Время начала	Время окончания	Баллы	Контрольное
2	Желваков В.А.	11В	12.05.2017	12:22	13:42	48	-
3	Иванов И.В.	11В	12.05.2017	12:23	13:34	32	-
4	Квасов Л.Д.	11В	12.05.2017	12:19	13:56	34	-
5	Кузьмина Е.А.	11В	12.05.2017	12:22	13:42	45	-
6	Кулибина А.Ю.	11В	12.05.2017	12:20	13:38	56	-
7	Петров К.Л.	11В	12.05.2017	12:22	13:44	49	-
8							
9							
10							

Рис.3.23. Интерфейс учителя – список результатов тестирований

На вкладке «Вопросы» преподаватель имеет возможность просмотреть перечень вопросов по выбранному предмету, добавить новый или отредактировать существующий вопрос.

Для перехода на форму редактирования вопроса необходимо нажать на текст вопроса и выделить его. Для удаления необходимо на выделенном вопросе нажать клавишу «Del»на клавиатуре.

Горячие клавиши, представленные на рис. 3.24, позволяют ускорить процесс создания новых вопросов и значительно сократить время наполнения и изменения базы данных вопросов.

В последней колонке формы отображается тип вопроса «с выбором» или «с кратким ответом».

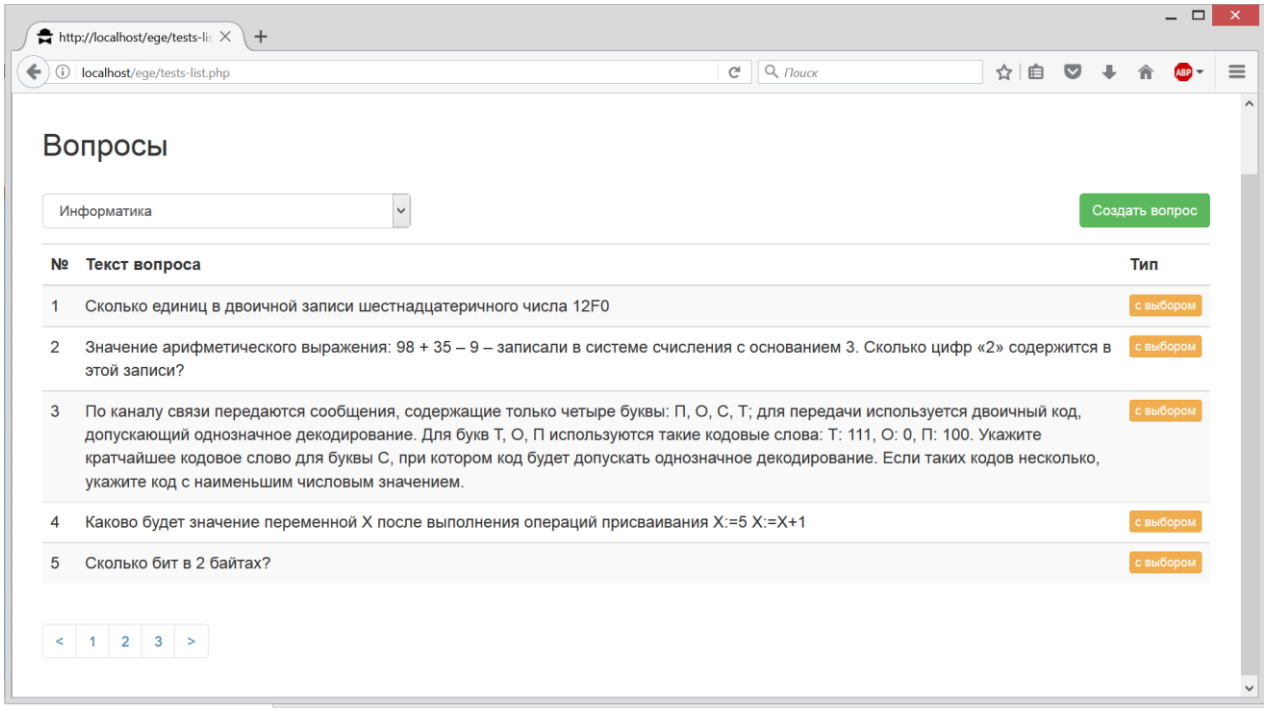


Рис.3.24. Интерфейс учителя – список вопросов для предмета

На рис. 3.25 показан внешний вид формы добавления нового вопроса. Пользователь имеет возможность ввести текст вопроса в многострочное поле ввода и 4 варианта ответов для вопросов с выбором.

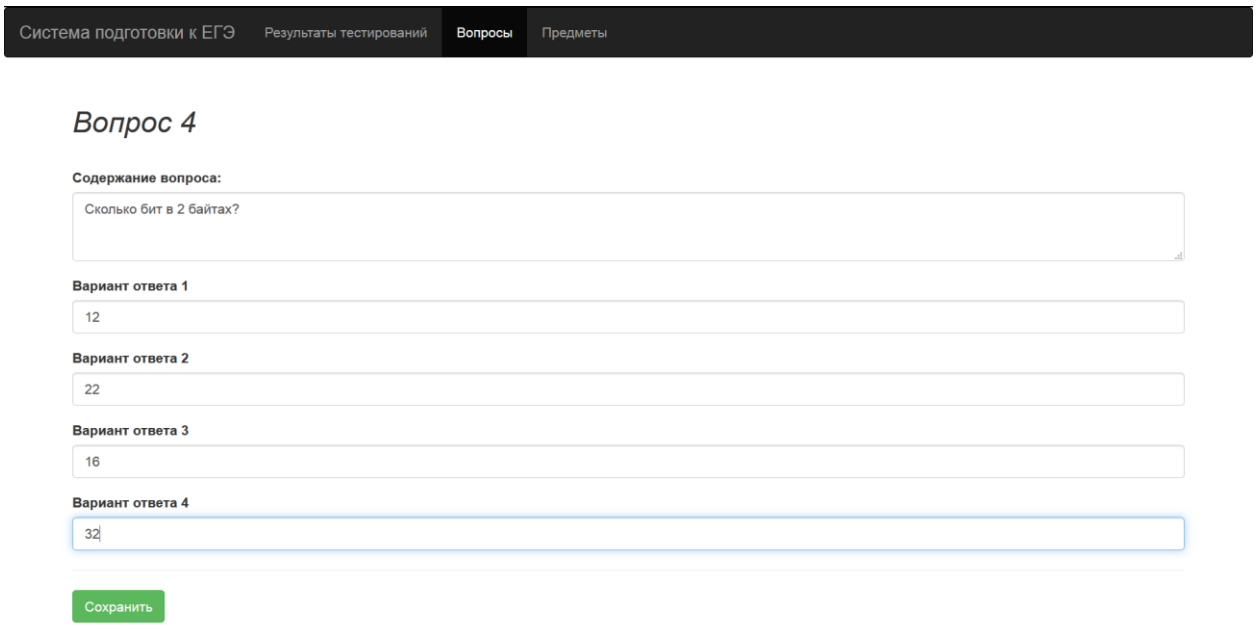


Рис.3.25. Интерфейс учителя – добавление нового вопроса

Интерфейс администратора системы содержит формы для редактирования, добавления и удаления информации о классах, учениках и учителях. Он построен на базовых формах и представляет собой стандартную реализацию функционала ведения справочников и имеет три вкладки «Классы», «Учителя» и «Ученики».

На рис. 3.26 приведен внешний вид вкладки «Классы». В этом интерфейсе администратор системы имеет возможность просмотреть список классов, добавить новый класс, удалить и отредактировать существующий.

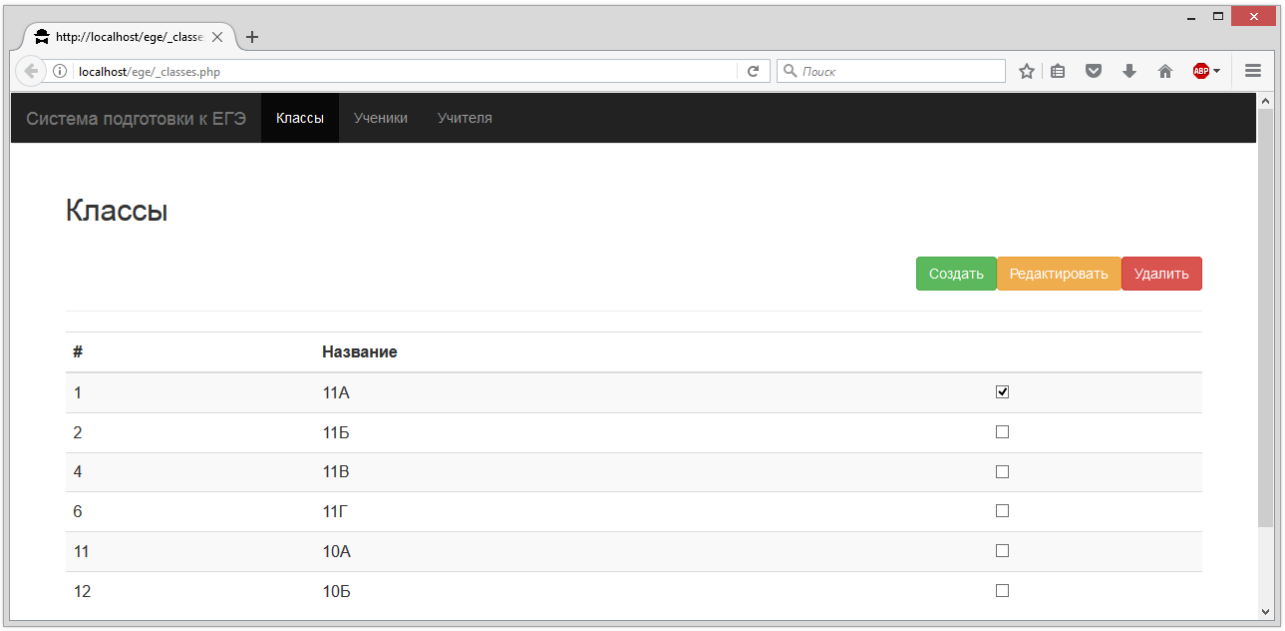


Рис.3.26. Интерфейс администратора – список классов

Для перехода на форму редактирования необходимо выбрать текущий класс с помощью компонента checkbox и нажать кнопку «Редактировать». При нажатии на кнопку «Удалить», текущий выбранный класс будет удален. При нажатии на кнопку «Добавить» откроется форма добавления информации о новом классе.

Внешний вид формы редактирования информации о классе представлен на рис. 3.27.

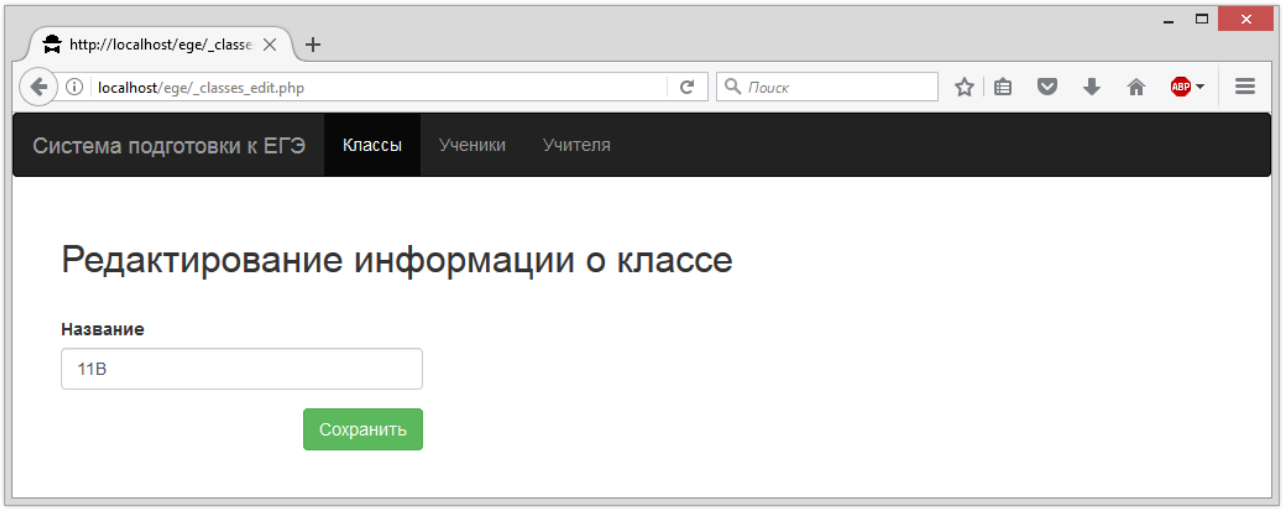


Рис.3.27. Интерфейс администратора – форма добавления и редактирования информации о классе

При нажатии кнопки «Удалить» текущий выбранный класс будет удален из системы. При нажатии кнопки «Создать» откроется форма создания нового класса.

На рис. 3.28 приведен внешний вид вкладки «Ученики».

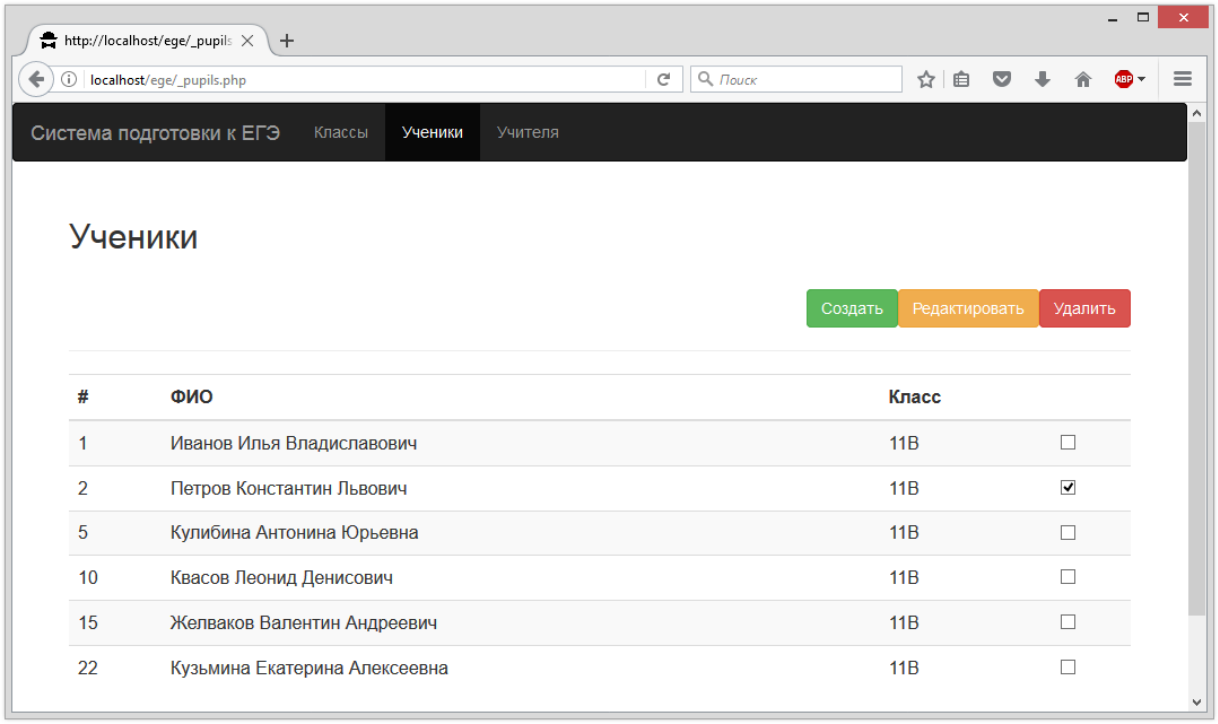


Рис.3.28. Интерфейс администратора – список учеников

В этом интерфейсе администратор системы имеет возможность просмотреть список учеников, добавить нового ученика, удалить и отредактировать существующего.

Для перехода на форму редактирования необходимо выбрать текущего ученика с помощью компонента checkboxи нажать кнопку «Редактировать».

При нажатии кнопки «Удалить» текущий выбранный ученик будет удален из системы. При нажатии кнопки «Создать» откроется форма добавления нового ученика.

В форме редактирования информации об ученике имеется возможность ввести имя, фамилию и отчество ученика, а также из выпадающего списка выбрать класс, в котором он учится.

На рис. 3.29 в поле «Пароль» имеется возможность задать пароль для входа в систему.

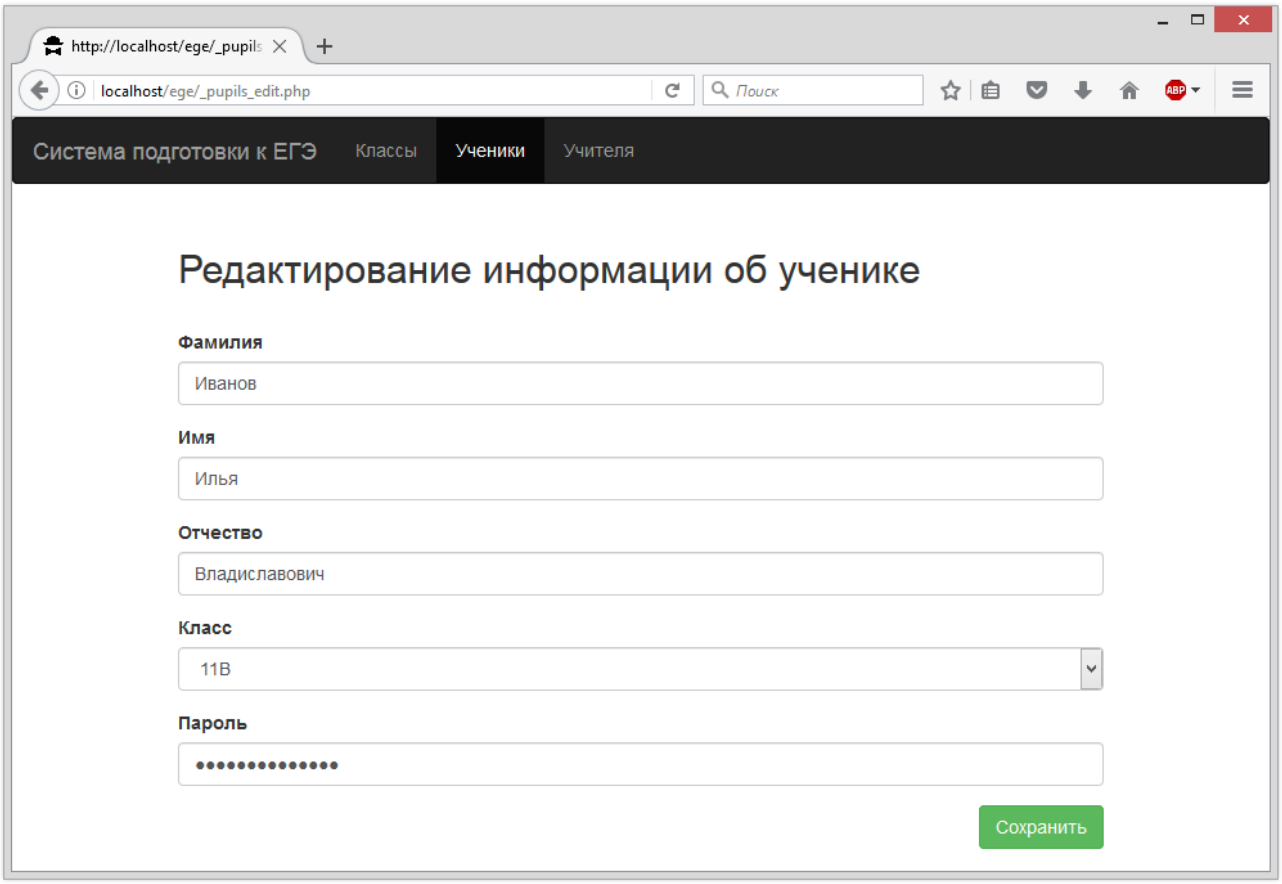


Рис.3.29. Интерфейс администратора – форма добавления и редактирования информации об ученике

На рис. 3.30 приведен внешний вид вкладки «Учителя». В этом интерфейсе администратор системы имеет возможность просмотреть список учителей, добавить нового учителя, удалить и отредактировать существующего.

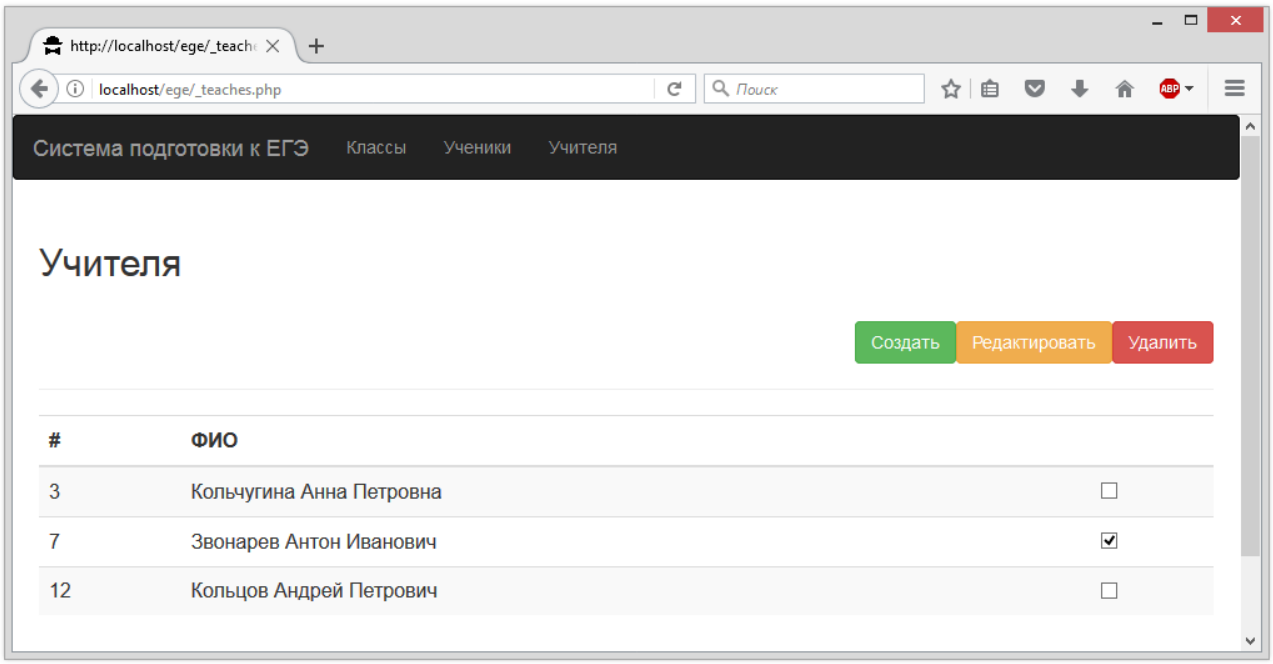


Рис.3.30. Интерфейс администратора – список учителей

Для перехода на форму редактирования необходимо выбрать текущего учителя с помощью компонента checkbox и нажать кнопку «Редактировать».

При нажатии кнопки «Удалить» текущий выбранный учитель будет удален из системы. При нажатии кнопки «Создать» откроется форма добавления нового учителя.

При нажатии на заголовок столбца таблицы будет произведена сортировка данных в таблице по этому столбцу по возрастанию, при повторном нажатии будет произведена сортировка данных в таблице по этому столбцу по убыванию.

В форме редактирования информации об учителе имеется возможность ввести имя, фамилию и отчество учителя, а также из выпадающего списка выбрать классы, которые будут доступны ему в личном кабинете.

На рис. 3.31 в поле «Пароль» имеется возможность задать пароль для входа в систему.

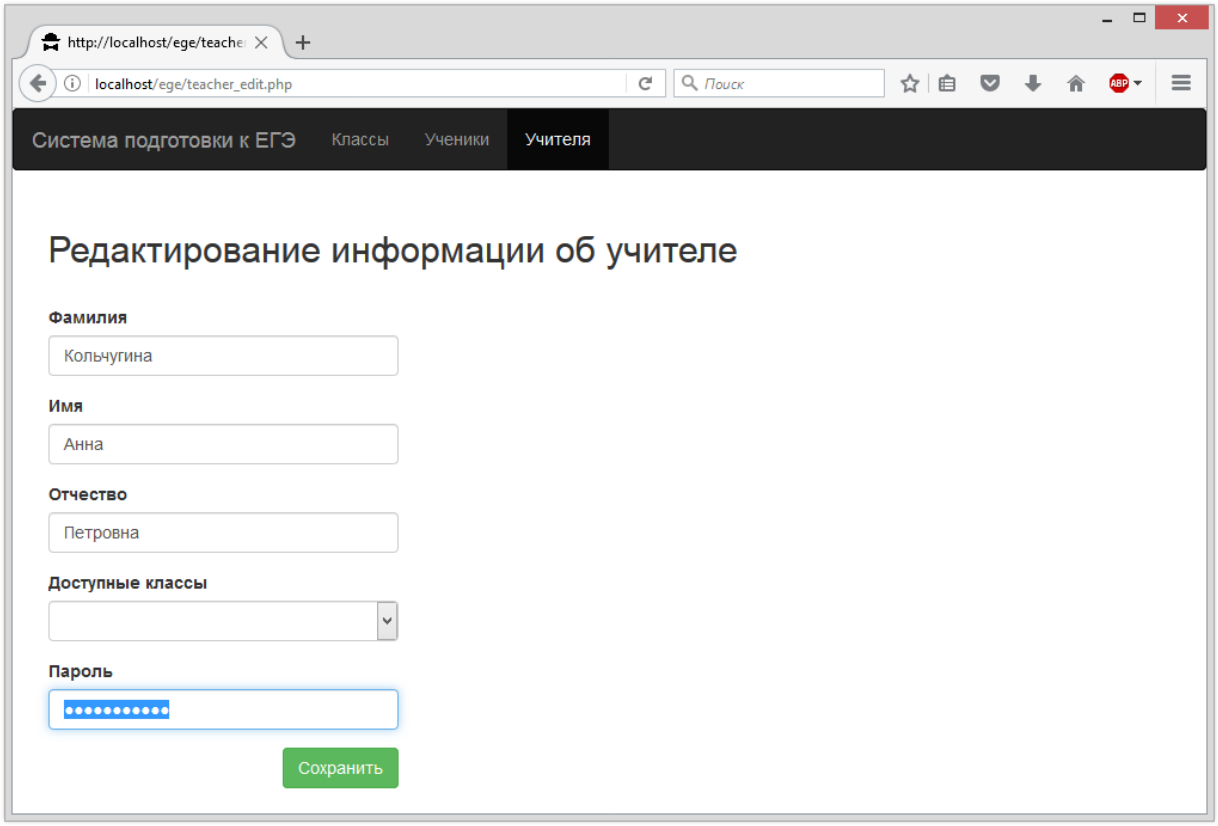


Рис.3.31. Интерфейс администратора – форма добавления и редактирования информации об учителе

ЗАКЛЮЧЕНИЕ

В рамках данной работы были рассмотрены современные подходы к разработке автоматизированных систем подготовки школьников к сдаче ЕГЭ в виде веб-приложения.

В ходе выполнения данной работы были решены следующие задачи:

– произведен обзор современных подходов к автоматизации процесса подготовки школьников к ЕГЭ;

– сформулированы требования к информационной системе подготовки школьников к сдаче ЕГЭ;

– произведено проектирование архитектуры информационной системы подготовки школьников к сдаче ЕГЭ;

– спроектирована структура базы данных;

– произведено проектирование пользовательского интерфейса информационной системы подготовки школьников к сдаче ЕГЭ;

– реализована информационная система подготовки школьников к сдаче ЕГЭ в соответствии с выдвинутыми требованиями;

– произведено тестирование разработанного программного комплекса.

В результате была реализована система подготовки школьников к сдаче ЕГЭ по информатике на основе современных веб-технологий. Таким образом, цель данной работы можно считать достигнутой.

Выполнение данной работы позволило систематизировать и углубить знания, полученные в ходе обучения, а также получить ценный опыт проектирования и разработки программных изделий, имеющих практическое применение в реальных условиях.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Министерство образования и науки РФ [Электронный ресурс]. Режим доступа:http://минобрнауки.рф

2. Сайт ФИПИ [Электронный ресурс]. Режим доступа: http://www.fipi.ru/

3. Официальный информационный портал единого государственного экзамена [Электронный ресурс]. Режим доступа: http://www.ege.edu.ru

4. Лещинер, В.Р. Единый государственный экзамен. Информатика. Комплекс материалов для подготовки учащихся. Учебное пособие. / В.Р. Лещинер, С.С. Крылов, А.П. Якушин. – Москва: Интеллект – центр, 2017. – 288 с.

5. Ушаков, Д.М. ЕГЭ-2017. Информатика. 20 типовых вариантов экзаменационных работ для подготовки к ЕГЭ./ Д.М. Ушаков. — М.: Астрель, 2016.

6. Мацяшек, Л.А. Практическая программная инженерия на основе учебного примера./ Л.А. Мацяшек, Б.Л. Лионг. - БИНОМ, 2012. - 956 с.

7. Громова, Е.Н. Системный анализ информационных комплексов: Учебное пособие / Е. Н. Громова. - СПб.: Лань, 2016. - 336 с.

8. Вдовин, В.М. Теория систем и системный анализ: Учебник для бакалавров / В.М. Вдовин, Л.Е. Суркова и др. - М.: Дашков и К, 2016. - 644 с.

9. Мазуркевич, А. MB PHP: настольная книга программиста./ Мазуркевич А., Еловой Д. — Мн.: Новое знание, 2003. — 480 с.: ил

10. Материалы официального сайта языка программирования PHPhttp://www.php.net/

11. Скляр, Д. PHP. Рецепты программирования./ Д. Скляр, А. Трахтенберг. - Питер, 2015, 784 с. 3-е изд.

12. Бейли, Л. Изучаем PHP и MySQL./ Л. Бейли, М. Моррисон. - Эксмо, 2010, 786 с.

13. Вин, Ч. Как спроектировать современный сайт: профессиональный веб-дизайн на основе сетки / Ч. Вин. - Москва [и др.]: Питер, 2011. - 192 с.

14. ВанДик, Д.К. Pro Drupal 7 Development: Third Edition / Д.К. ВанДик, М. Вестгейт. - Apress, 2010, 564 с .

15. Колисниченко, Д.Н. Движок для вашего сайта. CMS Joomla!, Slaed, PHP-Nuke./ Д.Н. Колисниченко. − БХВ-Петербург, 2008. − 245 с.

ПРИЛОЖЕНИЕ

Header.php

```
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="description" content="">
<meta name="author" content="">
<title>
</title>
<link
href="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/css/bootstrap.min.css"
rel="stylesheet">
<style class="custom-css">
#jumbo {
  background-color: #333;
  color: #eee;
}
#jumbo p {
  font-size: 16px;
}
#try-header {
  margin: 30px 0px;
}
#try-more {
  margin: 30px 0px;
  font-style: italic;
}
</style>
</head>

<body>
```

Footer.php

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js">
</script>
<table class="table" width="720" height="1">
<tbody>
</tbody>
</table>
</div>


</body>
```

ClassResults.php

```
<div class="container">
<!-- Example row of columns -->
<h2 id="">
Результаты тестирований
```

```
<br>
</h2>
<div class="row">
<div class="col-md-3">
<div class="form-group">
<select class="form-control">
<option value="Поле">
Поле
</option>
<option value="Класс">
Класс
</option>
<option value="Дата">
Дата
</option>
</select>
</div>
</div>
<div class="col-md-3">
<div class="form-group">
<input class="form-control" type="text">
</div>
</div>
<div class="col-md-3">
<button type="submit" class="btn btn-info">
Фильтровать
</button>
</div>
<div class="col-md-3">
<button type="submit" class="btn pull-right btn-success">
Сохранить в Excel
</button>
</div>
</div>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js">
</script>
<table class="table table-striped">
<tbody>
<tr>
<td>Иванов И.В.</td>
<td>11В</td>
<td>12.05.2017</td>
<td>12:23</td>
<td>13:34</td>
<td>32</td>
<td>-</td>
<td>
<button type="submit" class="btn btn-default btn-xs">
Проверить
</button>
<br>
</td>
</tr>
<tr>
<td>ПетровК.Л.</td>
<td>11В</td>
<td>12.05.2017</td>
<td>12:22</td>
<td>13:44</td>
<td>49</td>
```

```
<td>-</td>
<td>
<button type="submit" class="btn btn-default btn-xs">
Проверить
</button>
<br>
</td>
</tr>
<tr>
<td>Кулибина А.Ю.</td>
<td>11В</td>
<td>12.05.2017</td>
<td>12:20</td>
<td>13:38</td>
<td>56</td>
<td>-</td>
<td>
<button type="submit" class="btn btn-default btn-xs">
Проверить
</button>
<br>
</td>
</tr>
<tr>
<td>Квасов Л.Д.</td>
<td>11В</td>
<td>12.05.2017</td>
<td>12:19</td>
<td>13:56</td>
<td>34</td>
<td>-</td>
<td>
<button type="submit" class="btn btn-default btn-xs">
Проверить
</button>
<br>
</td>
</tr>
<tr>
<td>Желваков В.А.</td>
<td>11В</td>
<td>12.05.2017</td>
<td>12:22</td>
<td>13:42</td>
<td>48</td>
<td>-</td>
<td>
<button type="submit" class="btn btn-default btn-xs">
Проверить
</button>
<br>
</td>
</tr>
<tr>
<td>Кузьмина Е.А.</td>
<td>11В</td>
<td>12.05.2017</td>
<td>12:22</td>
<td>13:42</td>
<td>45</td>
<td>-</td>
<td>
<button type="submit" class="btn btn-default btn-xs">
```

```
Проверить
</button>
<br>
</td>
</tr>

</tbody>
<thead>
<tr>
<th>
Фио
<br>
</th>
<th>
Класс
</th>
<th>
Дата
</th>
<th>
Время начала
<br>
</th>
<th>
Время окончания
<br>
</th>
<th>
Баллы
</th>
<th>
Контрольное
</th>
<th>
<br>
</th>
</tr>
</thead>
</table>
<ul class="pagination">
<li>
<a href="#">&lt;</a>
</li>
<li>
<a href="#">1</a>
</li>
<li>
<a href="#">2</a>
</li>
<li>
<a href="#">3</a>
</li>
<li>
<a href="#">&gt;</a>
</li>
</ul>
</div>
```

EditQ.php

```
<div class="container">
<h2 id="try-more">
Вопрос 4
<br>
</h2>
<!-- Example row of columns -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js">
</script>
<div class="form-group">
<label>
Содержание вопроса:
</label>
<textarea class="form-control"></textarea>
</div>
<div class="form-group">
<label>
Вариант ответа 1
</label>
<input class="form-control" type="text">
</div>
<div class="form-group">
<label>
Вариант ответа 2
</label>
<input class="form-control" type="text">
<div class="form-group">
</div>
</div>
<label>
Вариант ответа 3
</label>
<input class="form-control" type="text">
<div class="form-group">
</div>
<label>
Вариант ответа 4
</label>
<input class="form-control" type="text">
<hr>
<button type="submit" class="btn btn-success">
Сохранить
</button>
</div>
```

DoingTest.php

```
<div class="container">
<!-- Example row of columns -->
<h2 id="try-header">
Вопрос 4
<br>
</h2>
<div>
Сколько бит в 2 байтах?
```

```
<br>
</div>
<label class="radio">
<input type="radio">
    12
</label>
<label class="radio">
<input type="radio">
    16
</label>
<label class="radio">
<input type="radio">
    4
</label>
<label class="radio">
<input type="radio">
    32
</label>
<hr>
<div class="row">
<div class="col-md-6">
</div>
<div class="col-md-6">
<a href="#" class="btn btn-success pull-right">Ответить</a>
</div>
</div>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js">
</script>
<ul class="pagination">
<li>
<a href="#">&lt;</a>
</li>
<?php for ($i=1;$i<28;$i++){?>
<li <?php if ($i==4)echo 'class = "active";'?>>
<a href="#" ><?php echo $i;?></a>
</li>
<?php }?>

<li>
<a href="#">&gt;</a>
</li>
</ul>
<table class="table">
<tbody>
</tbody>
</table>
</div>
```

EditSubject.php

```
<div class="container">
<h2 id="try-more">Редактирование предмета
<br></h2>
<!-- Example row of columns -->
<disabledscript
src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
style="display: none;">
</disabledscript>
<disabledscript
```

```
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js"
style="display: none;">
</disabledscript>
<div class="form-group">
</div>
<div class="form-group">
<label>Название предмета
</label>
<input class="form-control" type="text">
</div>
<div class="form-group">
<div class="form-group">
</div>
</div>
<div class="form-group">
</div>
<hr>
<button type="submit" class="btn btn-success">Сохранить
</button>
</div>
```

Subjects.php

```
<div class="container">
<!-- Example row of columns -->
<h2 id="try-header">
Предметы
<br>
</h2>
<div class="row">
<div class="col-md-4">
<div class="form-group">
</div>
</div>
<div class="col-md-4">
<div class="form-group">
</div>
</div>
<div class="col-md-4">
<a href="#" class="btn pull-right btn-danger">Удалить</a>
<a href="#" class="btn pull-right btn-warning">Редактировать</a>
<a href="#" class="btn pull-right btn-success">Создать</a>
</div>
</div>
<div class="form-group">
</div>
<table class="table table-striped">
<tbody>
</tbody>
<thead>
<tr>
<th>
<br>
</th>
<th>
Название
<br>
</th>
<th>
<br>
</th>
</tr>
```

```
</thead>
</table>
<ul class="pagination">
<li>
<a href="#">&lt;</a>
</li>
<li>
<a href="#">1</a>
</li>
<li>
<a href="#">2</a>
</li>
<li>
<a href="#">3</a>
</li>
<li>
<a href="#">&gt;</a>
</li>
</ul>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script
src="https://netdna.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js">
</script>
<table class="table" width="720" height="1">
<tbody>
</tbody>
</table>
</div>
```

AuthModel.php

```
<?php

class auth_model extends CI_Model {

public function __construct()
{
parent::__construct();
// Your own constructor code
}

function is_user($login, $password){
$q = "SELECT `id` ".
" FROM `user` ".
" WHERE `login` = ".$this->db->escape($login)." AND ".
" `password` = ".$this->db->escape($password)."";

$r = $this->db->query($q);
if ($r->num_rows()>0)
{
return $r->row();
}
return 0;
}
}
```

AuthController.php

```
<?php
error_reporting(E_ALL);

class auth extends CI_Controller {

public function __construct()
{
parent::__construct();
$this->load->library('session');
$this->load->model('auth_model');
}

function login()
{

$user = $this->auth_model->is_user($this->input->post('login'),md5($this->input->post('password')));

if($user){
$session_data = array('logon' =>'Yes!','id_user' =>$user->id); // записываемвсессиюпризнаклогона
$this->session->set_userdata($session_data);
          header('location:index.php?c=workspace');
}
else
{
//header('location:index.php');
}
}

function logout()
{
$session_data = array('logon' =>'', 'id_user' =>'');
$this->session->unset_userdata($session_data);
    header('location:index.php');
}
}
```

ClientController.php

```
<?php
if ( ! defined('BASEPATH'))exit('No direct script access allowed');

class clients extends CI_Controller {

public function __construct()
{
parent::__construct();
$this->load->helper('url');
$this->load->library('session');
$this->load->model('clients_model');
// проверяем наличие принаказа логинивания в сессии
      // если залогинились – выполняем вызванную функцию
if ($this->session->userdata('logon') != '')return;

// редиректна логин, если залогинивания не было
header('location:index.php');
```

```
}

public function index()
{

$data = array();
$data['active_item'] = 'clients';
$data['id_user'] = $this->session->userdata('id_user');
$this->load->view('layouts/header');
$this->load->view('layouts/top_menu', $data);

$this->load->view('clients');
$this->load->view('windows');

$this->load->view('layouts/footer');

}


public function save_client(){
$action = $this->input->post('action');
$client = array();
$client['id'] = (integer)$this->input->post('id');
$client['id_org'] = (integer)$this->input->post('id_org');
$client['fio'] = $this->input->post('fio');
$client['bdate'] = $this->input->post('bdate');
$client['address'] = $this->input->post('address');
$client['comment'] = $this->input->post('comment');
switch($action){
case 'add':
$id = $this->clients_model->add_client($client);
echo $id;
break;
case 'edit':
$this->clients_model->update_client($client);
break;
}
}

public function load_client(){
$filter = $this->input->post('filter');
$id = (integer)$this->input->post('id');
$id_org = (integer)$this->input->post('id_org');
$clients = $this->clients_model->load_client($filter, $id, $id_org);
echo json_encode($clients);
}

public function del_client(){
$id = (integer)$this->input->post('id');
$this->clients_model->del_client($id);
}

public function copy_client(){
$id = (integer)$this->input->post('id');
$clients = $this->clients_model->load_client('', $id, 0);
$client = $clients[0];

$client->fio = $client->fio." (Копия)";

$this->clients_model->add_client((array)$client);
}

}
```