МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование кафедры)

09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Прикладная информатика в социальной сфере
(направленность (профиль)/специализация)

БАКАЛАВРСКАЯ РАБОТА

на тему «Разработка информационной системы обработки персональных данных для учреждений дополнительного образования»

Студент	А.Р. Федосеев	
	(И.О. Фамилия)	(личная подпись)
Руководитель	О.В. Аникина	
	(И.О. Фамилия)	(личная подпись)
Консультанты	М. А. Четаева	
	(И.О. Фамилия)	(личная подпись)

Допустить к защите

Заведующий кафедрой к.т.н., доцент, А.В. Очеповский _____
(ученая степень, звание, И.О. Фамилия) (личная подпись)

«_____» _____ 20___ г.

Тольятти 2019

АННОТАЦИЯ

Тема выпускной квалификационной работы - «Разработка информационной системы обработки персональных данных для учреждений дополнительного образования».

Цель данной выпускной квалификационной работы - разработка программного обеспечения для оптимизации задач учета и контроля данных руководителям учреждений дополнительного образования.

Структура работы представлена введением, тремя главами, заключения, списка используемых источников.

Во введении описывается актуальность и практическая значимость проводимой работы, ставится цель и формулируются задачи.

В первой главе производится анализ предметной области учреждений дополнительного образования, ставятся требования к информационной системе, и строится концептуальная модель.

Во второй главе строится концептуальная модель информационной системы, проектируется база данных, и ставятся требования к аппаратно-программному обеспечению.

В третьей главе выбирается архитектура информационной системы, технологий разработки программного обеспечения и СУБД, проектируется физическая модель базы данных, поэтапно описывается процесс реализации и тестирования автоматизированной информационной системы.

В заключении приводятся результаты проделанной работы и формируются выводы.

Результатом выпускной квалификационной работы будет являться веб-приложение для контроля и учета данных в работе учреждений дополнительного образования.

Выпускная квалификационная работа содержит пояснительную записку объемом 60 страниц, включая 41 иллюстрацию и список литературы из 20 наименований.

ABSTRACT

The topic of the given bachelor's thesis is "Development of information system of personal data processing for institutions of additional education".

The bachelor's thesis consists of an explanatory note on 60 pages, the introduction on 2 pages, including 41 figures and the list of 20 references, including 5 foreign sources.

The aim of the work is to develop software to optimize the tasks of accounting and control of data for the heads of institutions of additional education.

The object of the bachelor's thesis is the process of working with personal data in the institution of additional education.

The subject of the bachelor's thesis is the automation of business processes in the institution of additional education.

The main issues which are highlighted in the first part of the main body are an analysis of the system of institutions of additional education, a comparison of two existing alternatives, and building a conceptual model of the subject area. The second part of the main body is devoted to the choice of a logical modeling technology, the construction of a logical model which is expressed in the diagrams in UML, and the formation of requirements for the hardware and software. In the third part, the choice of the information system architecture, development technologies and DBMS is made. Then the implementation and testing of the developed web application are carried out.

In the conclusion, we sum up the results of the work done and analyze the outcomes. The work is of interest to a narrow circle of readers who are interested in the rational organization of work with personal data in their systems of additional education.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Учреждение дополнительного образования - тип образовательного учреждения, основная цель которого — развитие мотивации личности к познанию и творчеству, реализация дополнительных образовательных программ и услуг в интересах личности, общества, государства. В современных условиях данный тип образовательных учреждений играет уникальную роль в системе образования. Они служат задачам обеспечения необходимых условий для личностного развития, укрепления здоровья и профессионального самоопределения. Сегодня учреждения дополнительного образования находятся в ведении системы образования, физической культуры и спорта, культуры, общественных организаций, органов по делам молодёжи.

Перед автором поставлена задача - разработать веб-приложение для учета и контроля информации при работе учреждения дополнительного образования; информацию необходимо хранить в базе данных; создать пользовательский интерфейс, обеспечивающий общение системы с пользователем.

Актуальность выпускной квалификационной работы заключается в востребованности автоматизированной информационной системы, способной оптимизировать работу с документами в учреждении дополнительного образования, увеличивая тем самым качество и эффективность его работы.

Целью данной выпускной квалификационной работы является разработка программного обеспечения для оптимизации задач учета и контроля данных руководителям учреждений дополнительного образования.

Для достижения поставленной цели необходимо решить следующие задачи:

- изучить предметную область и провести ее анализ;
- определить точку зрения рассмотрения бизнес-процессов;
- построить концептуальную модель информационной системы;
- найти и проанализировать существующие аналоги информационной системы;

- построить логическую модель информационной системы;
- спроектировать базу данных;
- разработать физическую модель базы данных;
- разработать программное обеспечение;
- протестировать программное обеспечение.

Объект исследования – процесс работы с персональными данными в учреждении дополнительного образования.

Предмет исследования – автоматизация бизнес-процессов в учреждении дополнительного образования.

В данной работе использовались следующие методы исследования: анализ, сравнение, CASE-средства, построение моделей данных. При выполнении работы автор руководствовался учебными пособиями и современными стандартами разработки программного обеспечения.

Практическая значимость выпускной квалификационной работы заключается в создании программного обеспечения, позволяющего найти рациональный подход к учету информации в учреждении дополнительного образования.

Выпускная квалификационная работа состоит из введения, трех глав, заключения, списка используемых источников.

В первой главе были определены требования к разрабатываемой информационной системе и построена ее концептуальная модель.

Во второй главе происходит проектирование базы данных и разработка логической модели информационной системы с построением необходимых диаграмм.

В третьей главе выбираются необходимые технологии для разработки информационной системы, строится физическая модель базы данных, осуществляется реализация веб-приложения и его тестирование.

В заключении приводятся выводы по проделанной работе.

ГЛАВА 1 ФУНКЦИОНАЛЬНОЕ МОДЕЛИРОВАНИЕ СИСТЕМЫ ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ

1.1 Технико-экономическая характеристика учреждений дополнительного образования

В настоящее время существует большое количество различных учреждений дополнительного образования: спортивные школы, школы иностранных языков, кружки для занятия творчеством. Однако не каждое заведение использует современные технологии для работы с данными, которые хранится при его работе, и предпочитает использование бумажных носителей информации. Несмотря на простоту и доступность, бумажный носитель не обеспечивает безопасности и достоверности данных, а также не делает ведение учета информации более затруднительным. Поэтому целесообразнее использование информационной системы, позволяющей вести хранение, изменение, обработку важных данных, например, оплату занятий учениками.

Электронный учет данных позволяет получить ряд преимуществ над традиционным бумажным хранением информации, а именно:

- полноценный веб-доступ, что позволяет пользоваться всеми функциями системы электронного учета данных вне зависимости от расположения пользователя;
- оперативный доступ к данным, что обеспечивает их быстрое редактирование;
- эффективное управление данными, от чего происходит следствие – повышение производительности сотрудников и исполнительской дисциплины;
- снижение финансовых и временных затрат при работе с данными.

Информационная система предназначена для учета персональной информации о преподавателях, студентах и управления структурой учреждения дополнительного образования.

Иерархия системы имеет следующий вид:

Проект – секция – группа.

В качестве проекта может выступать спортивный клуб, кружок, школа, организация.

В качестве секций может выступать уровень владения иностранным языком, уровень мастерства вида спорта, опыт в определенной деятельности.

Каждая секция содержит группы, которые включают студентов, обучающихся в учреждении, и преподавателей, ведущих занятия. В качестве группы может выступать возраст, весовая категория, пол учащихся.

Участниками системы являются следующие лица:

- студент;
- преподаватель;
- руководитель.

С точки зрения руководителя проекта ИС обеспечивает учет и хранение персональных данных участников системы, а так же администрирование проекта путем внесения и удаления участников, групп и секций в проект и редактирования существующих. Бизнес-процессы будут рассматриваться с точки зрения руководителя проекта.

В качестве примера учреждения дополнительного образования можно рассмотреть школу английского языка. В ней происходит деление учащихся сначала по уровням, а затем в полученных группах – по возрасту. Структура информационной системы подходит для описания составляющих школы английского языка:

Проект – школа изучения английского языка.

Секции – Basic-уровень, Pre-Intermediate-уровень, Intermediate-уровень, Upper-Intermediate-уровень, Advanced-уровень, Proficient-уровень.

Группы – младший школьный возраст, возраст средней школы, возраст старшей школы, возраст более 18 лет.

В качестве отчетности об успеваемости, посещаемости и оплате занятий ведется школьный журнал в виде бумажного носителя. Утеря или повреждение журнала приведет к утере большинства важной информации. Помимо этого,

человеческий фактор создает риск, что ответственный за ведение журнала забудет внести в него важную информацию, к примеру, об оплате учащимся занятий за прошедший месяц.

1.2 Концептуальное моделирование системы дополнительного образования

1.2.1 Выбор технологии концептуального моделирования

Концептуальное моделирование – это первоначальный этап создания информационной системы, заключающийся в сборе и анализе сведений о предметной области. Смысл данного этапа заключается в идентификации основных бизнес-процессов и взаимосвязей бизнес-сущностей, что позволяет определить спектр решаемых системой задач. Базовой методологией концептуального моделирования можно считать объектно-ориентированный подход.

Объектно-ориентированный подход призван описывать поведение системы при помощи декомпозиции объектов. Он отображает элементы реального мира в виде объектов, осуществляющих некоторые взаимодействия и обладающих определенным набором свойств. Главным преимуществом объектно-ориентированного подхода является точность его демонстрации логических сущностей реальных систем. Основополагающими подхода являются класс и объект.

- Класс – это абстракция элементов реального мира, которые объединены общностью поведения и структуры.
- Объект – это представитель класса, или имитация сущности реального мира, обладающий не только внутренней структурой, но и характерным только определенным группам поведением, называемым методами объекта.

Следующими основными свойствами объектно-ориентированного подхода являются три его принципа – полиморфизм, наследование и инкапсуляция.

Полиморфизм – это способность объекта относиться к нескольким классам. Данное понятие позволяет, во-первых, выделять различные аспекты исследуемой предметной области, а во-вторых, рассматривать объекты с различных точек зрения.

Наследование – это возможность создания новых классов на основе уже существующих. Данное свойство позволяет бороться с размножением сущностей без необходимости, препятствуя дублированию общей информации с возможностью добавления или переопределения методов поведения объекта.

Инкапсуляция – это намеренное сокрытие деталей реализации методов объектов и их внутренней структуры с предоставлением только необходимых инструментов. Данное свойство позволяет в первую очередь обеспечить безопасность информации и избавить от нее тех участников системы, для которых она не представляет ценности.

1.2.2 Разработка и анализ модели бизнес-процесса «КАК ЕСТЬ»

Чаще всего концептуальная модель представляется в виде диаграмм, что позволяет наглядно ознакомиться с процессами, протекающими внутри предметной области.

IDEF0 – методология функционального моделирования и графическая нотация, предназначенная для описания бизнес-процессов. Отличительной особенностью IDEF0 является ее акцентирование на соподчиненности объектов - участников бизнес процессов. Методология рассматривает логические отношения между видами деятельности предметной области, а не временную последовательность.

Диаграмма «КАК ЕСТЬ» - это модель существующих бизнес-процессов системы. Она позволяет зафиксировать информационные объекты, которые используются при выполнении функций различного рода детализации. На рисунке 1.1 представлена модель «КАК ЕСТЬ» бизнес-процесса «Запись нового участника в секцию» в работе учреждения дополнительного образования, построенная по методологии IDEF0:

Рисунок 1.1 – IDEF0-диаграмма «КАК ЕСТЬ» бизнес-процесса «Запись участника»

Для более детального представления бизнес-процессов на рисунке 1.2 представлена декомпозиция диаграммы «КАК ЕСТЬ»:

Рисунок 1.2 – Декомпозиция диаграммы «КАК ЕСТЬ»

Построение диаграммы «КАК ЕСТЬ» поможет выявить недостатки существующих бизнес-процессов системы и определить, какими преимуществами должны обладать новые.

Анализ построенных моделей позволяет сделать вывод, что работа с информацией об участниках образования является трудоемким процессом. Еще одним недостатком является наличие лишних действий, которых можно было бы избежать при автоматизированном варианте данного процесса. Проделанные действия необходимы для дальнейшей постановки задачи и построения автоматизированного варианта решения.

В качестве следующего шага необходимо осуществить формирование требований к проектируемой системе.

1.2.3 Обоснование необходимости автоматизированного варианта решения и формирование требований к новой технологии

От того, как налажена работа с информацией и данными, во многом зависит качество, эффективность и оперативность принимаемых решений образовательного учреждения. Поэтому необходимо произвести формирование требований к разрабатываемой системе, так как это позволит выяснить, какая работа должна быть проделана для достижения успешного результата.

Автоматизированная информационная система есть совокупность структурно взаимосвязанных подсистем, предназначенных для сбора, обработки, хранения, поиска и выдачи необходимой информации при работе учреждения дополнительного образования. Ниже представлены сформированные требования к системе и основные спецификации:

1. Общие требования к проектируемой системе.

1.1. Функции, реализуемые системой.

- регистрация;
- авторизация;

- создание, редактирование, удаление информации о студентах, преподавателях;
- создание, редактирование, удаление информации о секциях, группах;
- хранение и вывод информации об оплате занятий.

1.2. Специальные технические требования.

Диалоговый режим работы с пользователем.

2. Описание основных спецификаций

2.1 Функциональная структура системы.

- регистрация;
- авторизация;
- создание проекта;
- создание секций;
- создание групп внутри секций;
- добавление студентов и преподавателей в группы;
- добавление информации об успеваемости и оплате занятий.

2.2. Требования к качеству.

- надежность;
- практичность;
- эффективность;
- мобильность;
- сопровождаемость.

2.3. Распределение функций между человеком и системой

Со стороны пользователя необходимо ввести необходимую информацию в предлагаемые формы. Со стороны системы необходимо обработать и сохранить информацию согласно структуре.

2.4. Требования к безопасности.

- функции административного, процедурного и программно-технического уровня;
- обеспечение целостности;

- функции, реализующие аутентификацию и авторизацию;
- инкапсуляция данных.

2.5. Состав технической и пользовательской документации.

Документирование всех CRUD-методов сервера веб-приложения с помощью инструмента Swagger.

2.6. Условия внедрения и эксплуатации.

Одним из главных критериев успешности внедрения программного обеспечения - это выделение критических, с точки зрения общего результата, процедур в деятельности организации. Когда набор таких процедур определен, необходимо в первую очередь использовать программное обеспечение для автоматизации операций внутри именно этих процедур. Таким образом, разработанное программное обеспечение автоматически становится востребованным для организации.

1.3 Анализ существующих разработок на предмет соответствия сформулированным требованиям и постановка задачи на разработку автоматизированной информационной системы

Следующим этапом построения концептуальной модели системы является анализ уже разработанных аналогов с целью выявления основного направления проектирования и аспектов реализации.

Первой рассматриваемой системой будет являться «БАРС. Образование – Дополнительное образование» - единая региональная электронная система учета учащихся образовательных учреждений всех уровней и типов. Система «БАРС. Дополнительное образование» позволяет:

- создать единую базу данных об учащихся в учреждениях дополнительного образования;
- создать систему визуализации утвержденных показателей в части учета контингента обучающихся в учреждениях дополнительного образования;

- реализовать единый формат информационного взаимодействия автоматизированных систем учета обучающихся в учреждениях дополнительного образования;
- интегрироваться с существующими системами образования всех уровней и ведомств.

На рисунке 1.3 представлен пользовательский интерфейс системы «БАРС»:

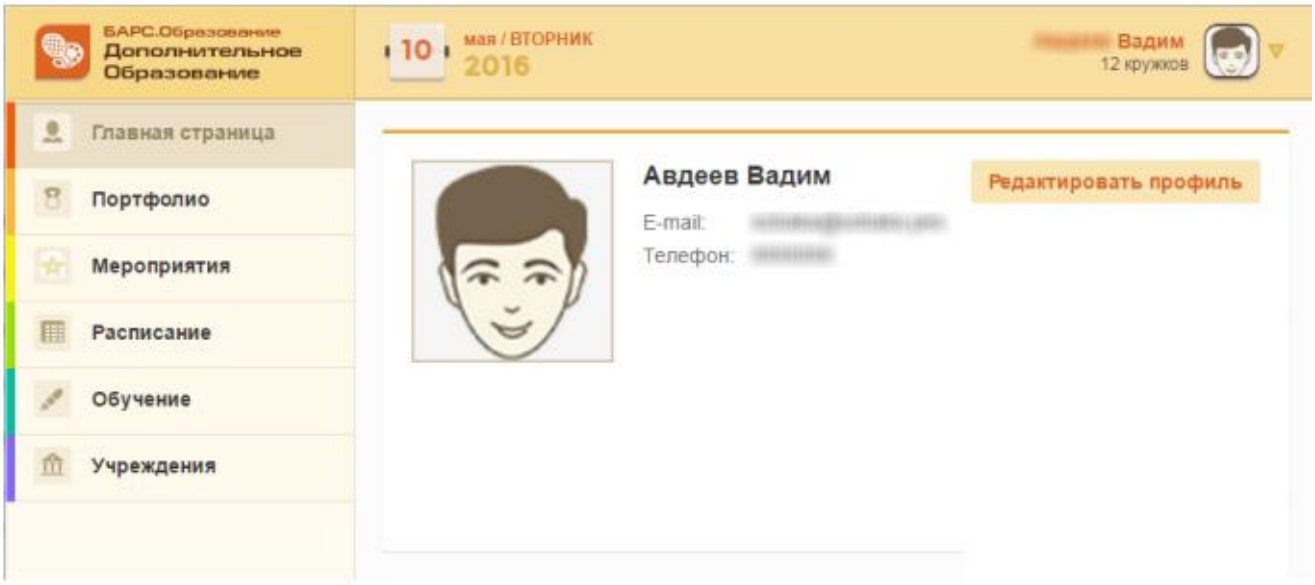


Рисунок 1.3 – Пользовательский интерфейс «БАРС. Образование»

Преимущества данного продукта:

- помощь в принятии управленческих решений;
- достоверность и актуальность данных;
- оперативность при формировании отчетов;
- единое информационное пространство;
- централизованное хранение данных;
- свободное программное обеспечение;
- интеграция с инфраструктурой Электронного Правительства;
- адаптация к региональным требованиям.

Несмотря на множество преимуществ, главным недостатком системы в данной предметной области является ее сложность. Существует несколько

документаций по использованию программного продукта, поэтому можно сделать вывод, что для использования данной системы необходимы временные затраты для ознакомления и уверенного использования. Еще одним недостатком системы является ее законченность, что не дает возможности дополнения к системе новых программных модулей в будущем. По этим причинам программное обеспечение не может являться удовлетворительным решением для данной предметной области.

«КМ-Школа» - это информационный интегрированный продукт для средней школы, созданный на основе Интернет технологий. Он объединяет уникальный образовательный мультимедийный контент, систему доставки и управления им, а также удобные и эффективные средства для автоматизации управления школой. На рисунке 1.4 представлен пользовательский интерфейс информационной системы:

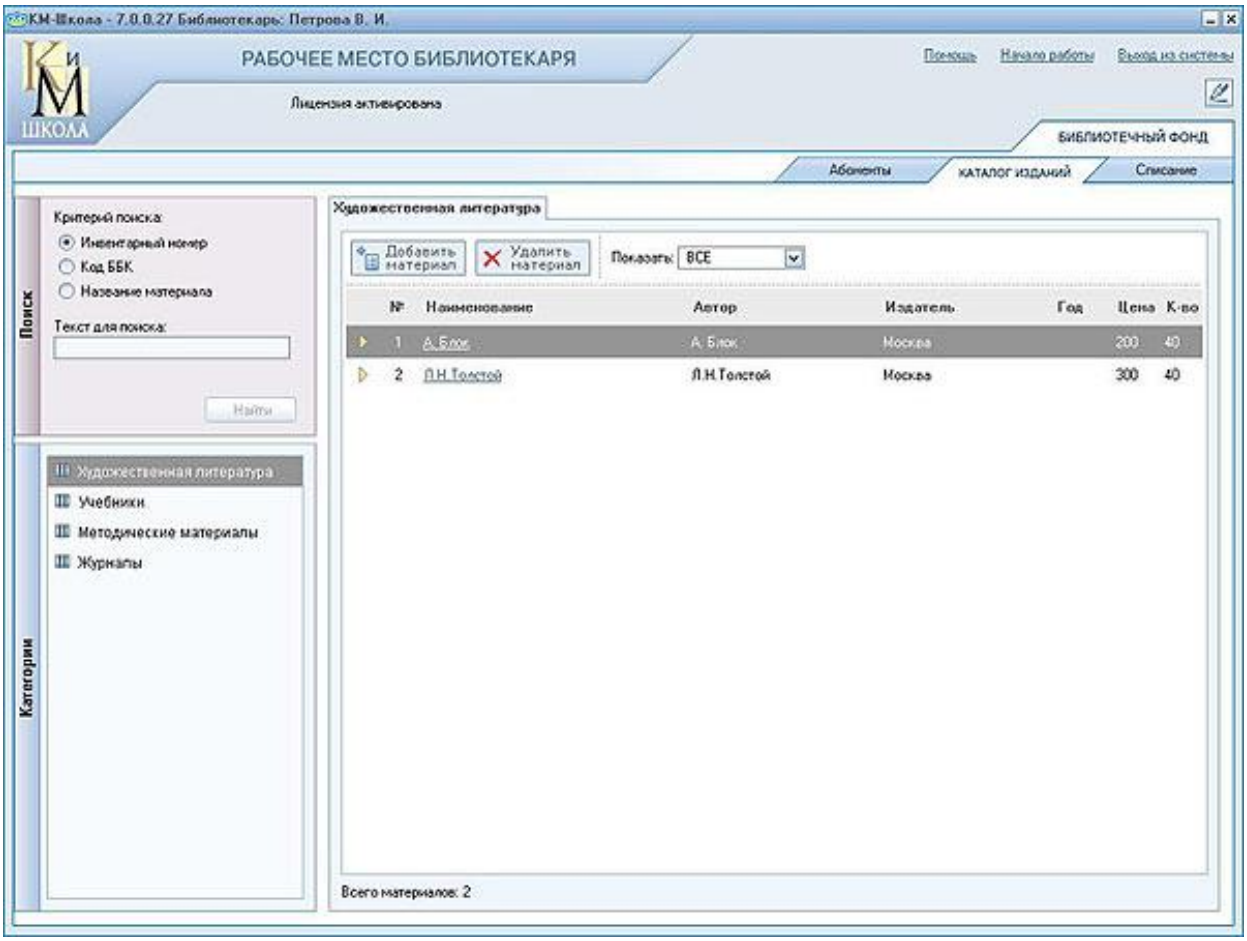


Рисунок 1.4 – Пользовательский интерфейс «КМ-Школа»

Продукт «КМ-Школа» полностью соответствует современным образовательным стандартам. Он позволяет учителям в процессе обучения использовать как разнообразные методы обучения (информационно-рецептивный, репродуктивный, проблемный, эвристический, исследовательский), так и все формы обучения.

Основные компоненты продукта:

- База знаний.
- Программный комплекс.
- Интернет-сервисы.

Недостаток системы аналогичен предыдущему аналогу – она в первую очередь удовлетворяет потребности средних школ, а не учреждений дополнительного образования.

Рассмотренные выше продукты не удовлетворяют поставленным требованиям, не учитывают всю специфику рассматриваемой предметной области и имеют ряд недостатков, не позволяющих им являться решением проблемы работы с персональными данными в учреждениях дополнительного образования. Поэтому существует необходимость разработать систему для учреждений дополнительного образования.

Для обобщения полученных результатов анализа предметной области и систем-аналогов, а так же определения аспектов реализации необходимо поставить задачу на разработку проекта.

Информационная система создается с целью:

- автоматизации ввода, контроля и учета данных о студентах, учителях, секциях и группах в базу данных;
- систематизации программной логики в виде веб-приложения;
- создания инструментов администрирования для руководителей проектов.

Для достижения поставленных целей необходимо решить следующую задачу: разработать программный интерфейс веб-приложения для учета и контроля данных при работе учреждения дополнительного образования.

Информацию необходимо хранить в базе данных. Создать электронную документацию для сервисов, обеспечивающих функционирование приложения.

1.4 Разработка модели бизнес-процесса «КАК ДОЛЖНО БЫТЬ»

Одним из эффективных методов трансформации ожиданий в результат является разработка и моделирование диаграммы «КАК ДОЛЖНО БЫТЬ», позволяющей определить процессы для достижения бизнес-результатов. Данная модель описывает будущее состояние бизнес-процесса в том виде, в котором его окружение и организация изменятся после реализации системы.

На рисунке 1.5 представлена диаграмма «КАК ДОЛЖНО БЫТЬ» с точки зрения руководителя учреждения дополнительного образования:

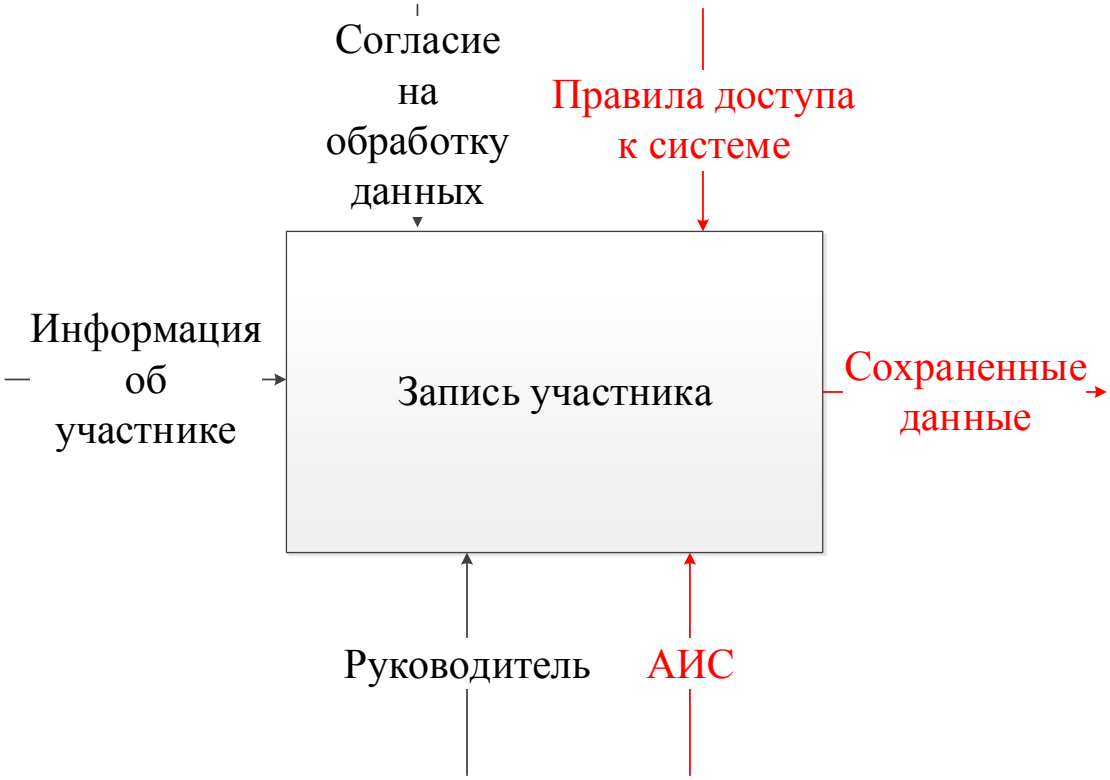


Рисунок 1.5 - IDEF0-диаграмма «КАК ДОЛЖНО БЫТЬ» бизнес-процесса «Запись участника»

Для более детального представления бизнес-процессов на рисунке 1.6 представлена декомпозиция диаграммы «КАК ДОЛЖНО БЫТЬ»:

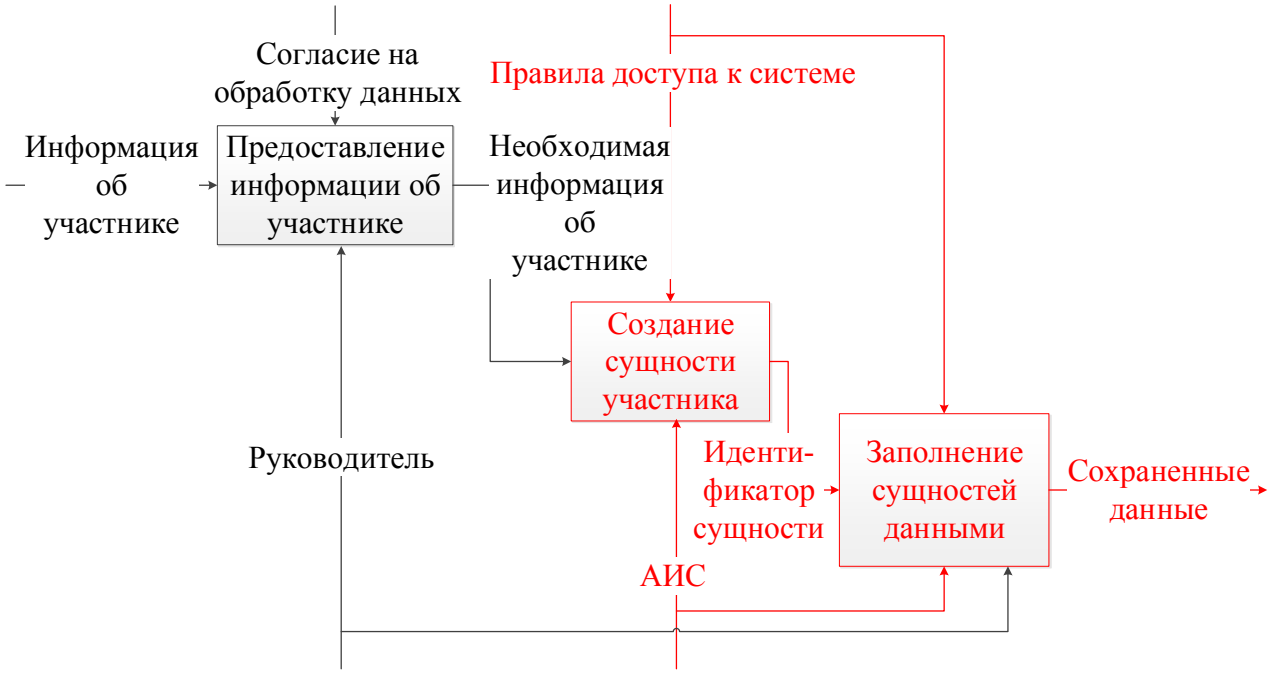


Рисунок 1.6 – Декомпозиция диаграммы «КАК ДОЛЖНО БЫТЬ»

Продемонстрированные диаграммы позволяют отследить, как изменился бизнес-процесс после его автоматизации: пропала необходимость участия того, чьи данные вносятся, так как отсутствуют какие-либо бумажные карточки подтверждения обучения – наличие ученика в системе уже будет являться подтверждением.

Вывод по главе 1

В данной главе был проведен анализ предметной области и выбрана точка зрения рассмотрения бизнес-процессов. Затем было произведено концептуальное моделирование, составлены требования к информационной системе, после чего произведен анализ существующих аналогов на предмет соответствия этим требованиям, что послужило основанием для необходимости разработки автоматизированной информационной системы. Затем была поставлена задача на разработку информационной системы учета данных в работе учреждения дополнительного образования. При построении концептуальной модели использовались диаграммы «КАК ЕСТЬ» и «КАК ДОЛЖНО БЫТЬ».

ГЛАВА 2 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ УЧРЕЖДЕНИЙ ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ

2.1 Выбор технологии логического моделирования автоматизированной информационной системы

Следующим этапом после построения концептуальной модели является построение множества диаграмм, совокупность которых отражает связанную информационную структуру предметной области. Эти действия заключают в себе построение логической модели информационной системы.

Под термином «CASE-средства» понимается совокупность программных средств, поддерживающих процессы создания и сопровождения информационной системы. CASE-средства позволяют проводить следующие действия:

- анализ и формулировка требований;
- проектирование приложений и баз данных;
- конфигурационное управление и управление проектом;
- документирование;
- обеспечение качества;
- тестирование.

Таким образом, современные CASE-средства вместе с системным программным обеспечением образуют полную среду разработки ИС.

CASE-средства позволяют не только создать готовый к использованию программный продукт, но и обеспечить корректный процесс его разработки. Основная цель этой технологии – отделение процессов проектирования от написания программного кода, сборки и тестирования. Еще одной целью технологии является максимальная инкапсуляция деталей проектирования и разработки информационной системы.

Все вышеперечисленные преимущества призваны облегчить задачу проектировщика путем сокращения времени разработки, уменьшения

количества ошибок программного кода и добавления возможности использования модулей системы для будущих разработок.

Методика разработки логической модели подразумевает под собой переход от структурной диаграммы «КАК ДОЛЖНО БЫТЬ» к построению диаграммы вариантов использования, отражающей функциональные аспекты логической составляющей системы; построению диаграммы классов предметной области, рассматривающей элементный аспект логической составляющей системы; построению диаграммы последовательности, отражающий динамический аспект логической составляющей системы.

UML – унифицированный язык моделирования – язык моделирования, предназначенный для объектно-ориентированного анализа и проектирования. UML включает в себя три основных элемента: диаграммы, сущности и связи.

- Сущности – это абстрактные объекты, которые представляют собой основные элементы модели.
- Связи – элементы, позволяющие соединять сущности между собой.
- Диаграммы – строительные блоки логической модели, группирующие наборы сущностей и связей.

UML используется для визуализации, конструирования и документирования информационных систем.

2.2 Построение логической модели автоматизированной информационной системы

Диаграммы вариантов использования применяются при бизнес-анализе для моделирования видов работ, выполняемых организацией. Создание диаграммы вариантов использования необходимо в первую очередь для представления, кто именно использует систему и что именно он может в ней сделать. На рисунке 2.1 изображена диаграмма вариантов использования информационной системы учета данных в работе учреждения дополнительного образования:

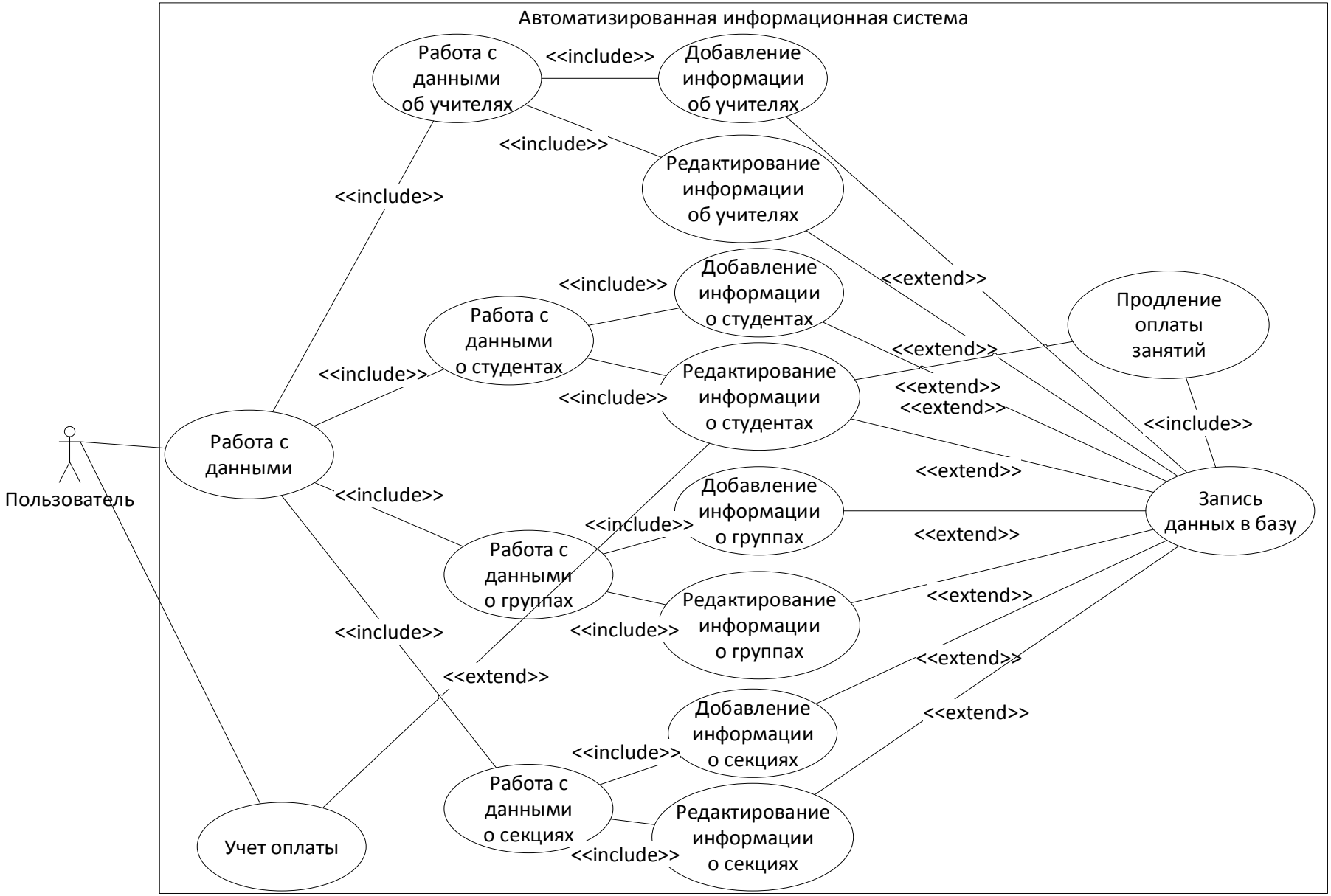


Рисунок 2.1 – Диаграмма вариантов использования ИС

Основными элементами такой диаграммы являются актеры, то есть объекты, выполняющие действия, и варианты использования, то есть возможные действия, выполняемые актерами. Диаграмма показывает функции участников, выполняющиеся в процессе деятельности информационной системы. Пользователь представлен как актер, а функции, которые они выполняют - как наборы действий.

Диаграммы классов используются при моделировании проектируемых систем наиболее часто. Они являются одной из форм статического описания системы с точки зрения ее проектирования, показывая ее структуру.

Диаграмма классов состоит из множества элементов, которые в совокупности отражают декларативные знания о предметной области. Эти знания интерпретируются в базовых понятиях языка UML, таких как классы, интерфейсы и отношения между ними и их составляющими компонентами.

На рисунке 2.2 представлена диаграмма классов информационной системы:

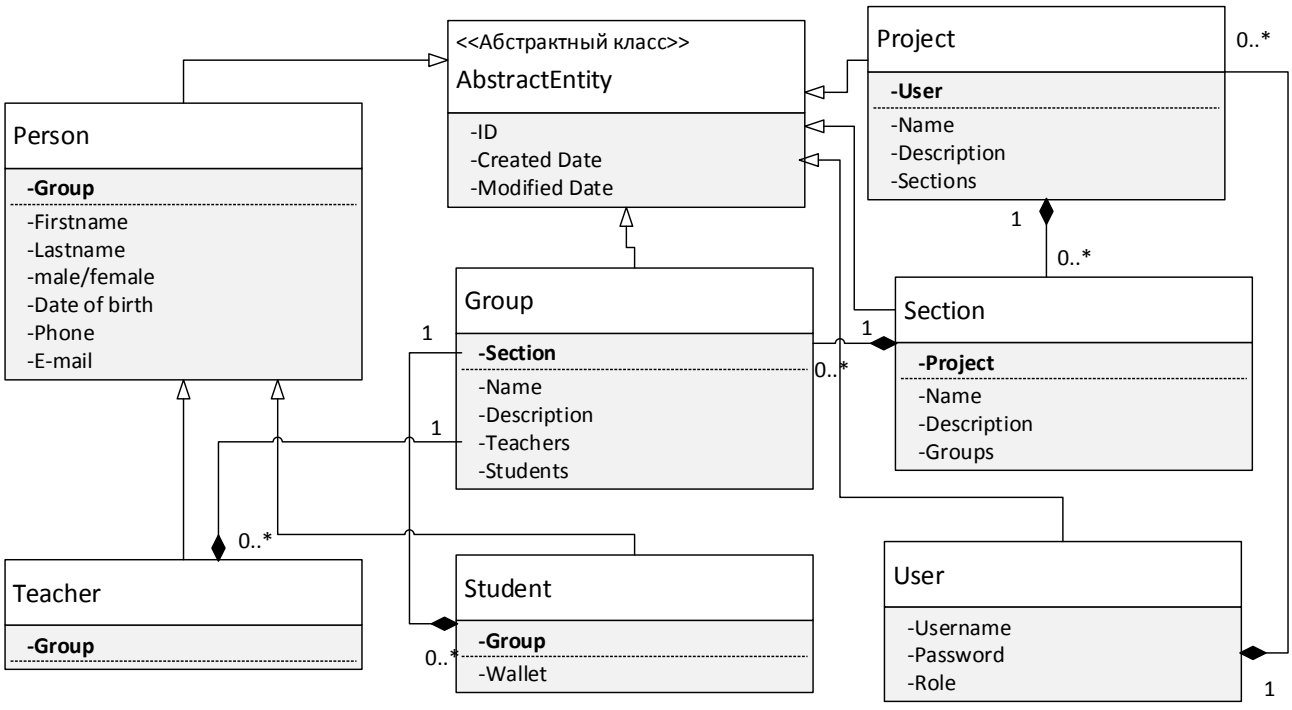


Рисунок 2.2 – Диаграмма классов ИС

Abstract Entity - абстрактная сущность, представляющая собой верхний уровень иерархии классов системы. Содержит в качестве параметра

уникальный идентификатор, даты создания сущности и последнего изменения. Это означает, что все остальные классы системы будут иметь свои уникальные идентификаторы, наследуясь от этой сущности.

- Project – сущность, отвечающая за представление проекта в системе. Имеет связь-композицию с бизнес-сущностями. Другими словами, к проекту относятся все остальные классы системы, наследующиеся от Business Entity.
- Section – сущность, отвечающая за представление секций в системе. Связана с группой отношением «Один ко многим». Другими словами, в одной секции может быть несколько групп.
- Group - сущность, отвечающая за представление групп в системе. Связана с учителями и студентами отношениями «Один ко многим». Другими словами, в одной группе могут быть несколько учителей.
- Person – сущность, представляющая людей как классы внутри в системе. В качестве параметров имеет такие данные, как имя, фамилию.
- Teacher - сущность, представляющая учителей в системе. Наследуется от Person.
- Student – сущность, представляющая студентов в системе. Наследуется от Person.

Диаграмма последовательности предназначена для моделирования взаимодействия объектов системы во времени, а также обмена сообщениями между ними. Основные понятия диаграммы последовательности связаны с понятием Объект и Сообщение. Объекты в основном представляю экземпляры класса или сущности, обладающие поведением. В качестве объектов могут выступать пользователи, инициирующие взаимодействие, классы, обладающие поведением в системе или программные компоненты, а иногда и системы в целом.

На рисунке 2.3 изображена диаграмма последовательности информационной системы:

Рисунок 2.3 – Диаграмма последовательности ИС

На диаграмме видна последовательная работа приложения с помощью задействованных функций.

2.3 Проектирование базы данных автоматизированной информационной системы

Логическая модель является начальным прототипом будущей базы данных. Целью построения логической модели является получение графического представления логической структуры базы данных. Она призвана для иллюстрирования сущностей, а также их взаимоотношений между собой. Важно отметить, что логическая модель не реализовывается средствами

конкретной СУБД, поэтому не представляет сущности и связи на физическом уровне.

На рисунке 2.4 представлена логическая модель базы данных информационной системы:

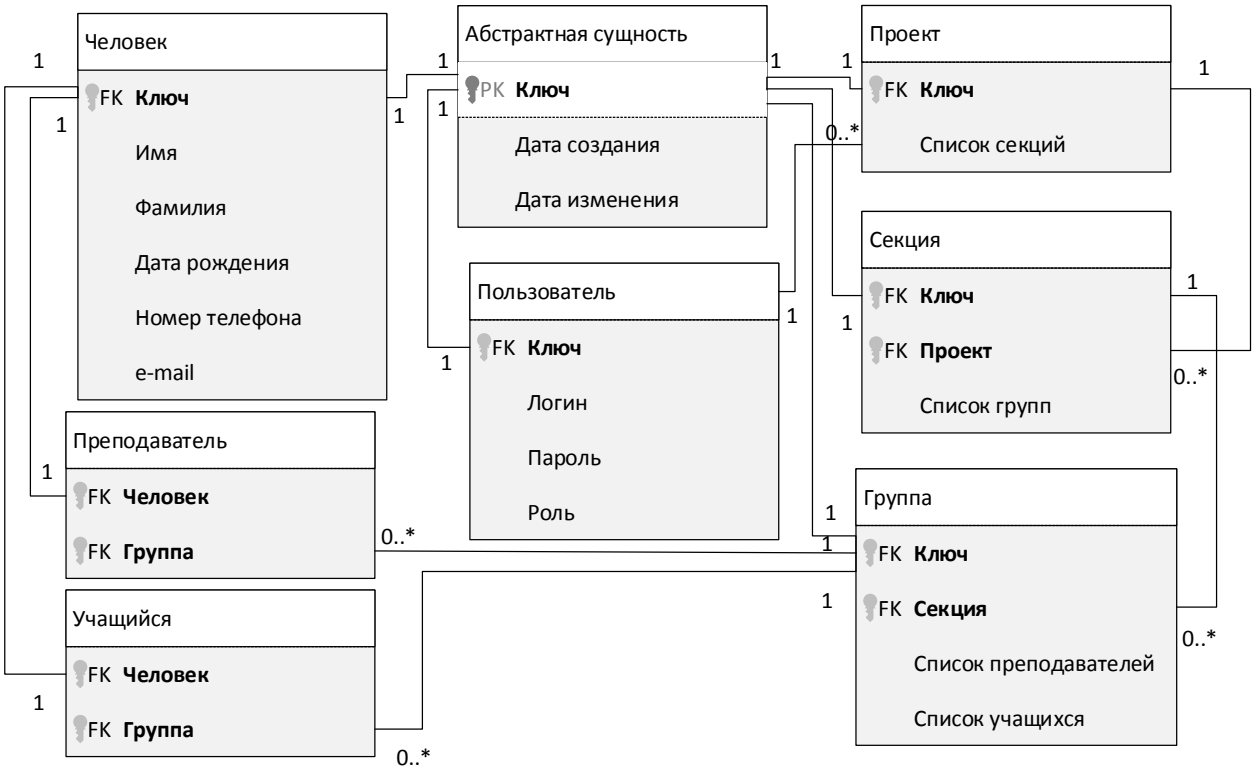


Рисунок 2.4 – Логическая модель базы данных

Ниже представлено пояснение с модели:

- Пользователь – сущность, отвечающая за пользователя, авторизированного в системе;
- Абстрактная сущность – наивысшая сущность в иерархии системы, хранящая уникальный ключ и даты создания и изменения;
- Проект – наивысшая сущность в иерархии с точки зрения администрирования системы;
- Секция – следующая за проектом сущность, зависимая от проекта;
- Группа – сущность, хранящая в себе студентов и преподавателей и зависимая от секции;
- Человек – сущность, отвечающая за хранение информации о людях;

- Преподаватель – сущность учителя, зависимая от группы;
- Студент – сущность ученика, зависимая от группы.

Проектирование базы данных – завершающий этап в разработке логической модели информационной системы. Далее необходимо определить список требований к аппаратно-программному обеспечению.

2.4 Требования к аппаратно-программному обеспечению автоматизированной информационной системы

Список минимального аппаратно-программного обеспечения для информационной системы учета данных в работе учреждения дополнительного образования приведен ниже:

Требования к серверу приложения:

- оперативная память – 4 GB;
- свободное место на жёстком диске – 500 MB;
- подключение к сети Интернет;

Требования к серверу СУБД:

- оперативная память – 4 Гб;
- свободное место на жёстком диске – 1 GB;
- подключение к сети Интернет;

Данный комплекс технических средств и программного обеспечения необходим для качественной работы автоматизированной информационной системы.

Вывод по главе 2

В данной главе был произведен выбор технологии логического моделирования АИС, затем была построена логическая модель системы и спроектирована база данных. При построении логической модели использовались диаграммы вариантов использования, классов и последовательности.

ГЛАВА 3 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ УЧРЕЖДЕНИЙ ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ

3.1 Выбор архитектуры автоматизированной информационной системы

Архитектура АИС – это распределение ее компонентов и подсистем по функциям и их взаимодействие по управлению данными.

Традиционными видами архитектур АИС являются следующие:

- файловый сервер – информация и программный код хранятся на стороне сервера, а на стороне клиента происходит обработка данных. Существенный недостаток такой архитектуры – хранение и обработка информации происходят в разных местах, что понижает надежность и производительность системы;
- двухслойная клиент-серверная архитектура – на стороне сервера хранение данных, обработка и выполнение запросов, на стороне клиента – исполнение бизнес-логики. В этом также заключается и главный недостаток – в случае каких-либо изменений алгоритмов на сервере необходимо обновлять программное обеспечение на всех клиентах;
- промежуточная клиент-серверная архитектура – отличие от двухслойной в том, что бизнес-логика делится и на клиент, и на сервер. Следовательно, это отличие не избавляет архитектуру от соответствующего недостатка;
- трехуровневая клиент-серверная архитектура – сервер отвечает за выполнение бизнес-логики, СУБД – за хранение данных, клиент - за пользовательский интерфейс. Главным преимуществом является распределенная нагрузка на все компоненты, что повышает производительность и надежность системы. Главный недостаток – повышенные расходы на обслуживание сервера.

Если сравнивать между собой традиционные виды архитектур, можно прийти к выводу, что с точки зрения производительности и надежности наилучшим является трехуровневая клиент-серверная архитектура.

Архитектура «клиент-сервер» на основе Web-технологий – это развитие трехуровневой клиент-серверной архитектуры. Она позволяет достигнуть выдвигаемых при проектировании системы требований по общей производительности и надежности, минимизируя при этом денежные затраты на построение, обслуживание и администрирование серверной части.

Клиент-сервер на основе Web-технологий будет выбран как архитектура автоматизированной информационной системы учета данных в работе учреждения дополнительного образования.

На рисунке 3.1 представлена диаграмма развертывания АИС:

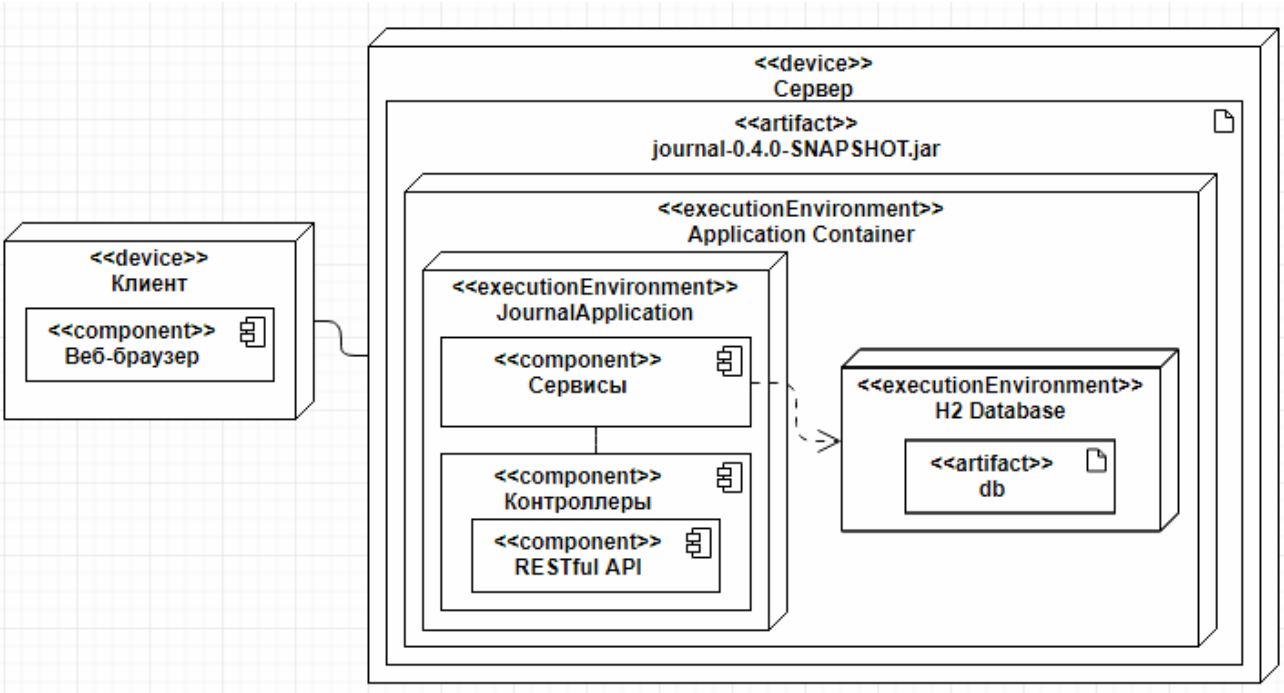


Рисунок 3.1 – Диаграмма развертывания АИС

Ниже описаны уровни работы с данными в клиент-серверной архитектуре на основе Web-технологий.

Уровень клиента представляет собой браузер, подключающийся к серверу посредством протокола передачи гипертекста HTTP или любое приложение на любой клиентской машине.

На уровне доступа к данным или промежуточном уровне размещается сервер. Веб-слой служит для управления обменом данными между уровнем клиента и бизнес слоем.

Бизнес-слой выполняет бизнес-логику, удовлетворяя конкретные потребности в коммерческих проектах с использованием данных из базы данных.

Уровень EIS или уровень данных состоит из блоков хранения данных. На нем размещена система управления базами данных для хранения данных о пользователях, их успеваемости, секциях и. т. д.

3.2 Выбор технологии разработки программного обеспечения автоматизированной информационной системы

Для реализации информационной системы были выбраны следующие средства:

- Java – объектно-ориентированный язык программирования, который интенсивно применяется в веб-разработке в настоящее время.
- Java Platform, Standart Edition – стандартная версия платформы Java 2, предназначенная для создания и исполнения апплетов и приложений, рассчитанных на индивидуальное пользование или на использование в масштабах малого предприятия.
- Java SE Development Kit 11 – это среда разработки для создания приложений и компонентов на языке программирования Java. JDK включает в себя инструменты, полезные для разработки и тестирования программ, написанных на языке программирования Java.
- InteliJ IDEA 2018 3.1 – интегрированная среда разработки программного обеспечения для языка программирования Java, разработанная компанией JetBrains.
- Spring - универсальный фреймворк с открытым исходным кодом для Java-платформы, используемый в настоящее время для широкой разработки веб-приложений.
- Spring Boot 2.1.2 RELEASE — это инструмент, целью которого является упрощение создания приложений на основе Spring. Он позволяет наиболее

простым способом создать web-приложение, требуя от разработчиков минимум усилий по его настройке и написанию кода для запуска.

- Maven - фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM (англ. Project Object Model), являющемся подмножеством XML.
- Swagger 2.8.0 - это фреймворк и спецификация для определения REST APIs в формате, дружественном к пользователю и компьютеру.
- Bootstrap 4.3.1 – это инструмент с открытым исходным кодом для разработки с HTML, CSS и JavaScript. Позволяет использовать готовые прототипы элементов пользовательского веб-интерфейса для ускорения и облегчения разработки.
- Vue.js 2.6.10 – JavsScript-фреймворк для создания пользовательских интерфейсов. Основным преимуществом является его легкая интеграция в проекты, низкий вес и возможность динамической загрузки веб-страницы.

Использование вышеперечисленных средств обусловлено их широкой распространенностью в современной веб-разработке, их актуальностью и качеством как инструментов реализации данного приложения.

3.3 Выбор системы управления базами данных автоматизированной информационной системы и разработка физической модели данных базы данных

В качестве СУБД информационной системы использовалась H2.

H2 – это открытая кроссплатформенная СУБД, написанная на языке Java. Система легковесна, при этом предоставляет большой ряд функций, а так же обеспечивает большую надежность, чем нативные приложения. Разработка и поддержка данной СУБД ведется и в настоящее время. Важно отметить, что при разработке СУБД не использовались сторонние библиотеки, она независима от прочих программных продуктов. Вся функциональность реализована разработчиками самостоятельно с целью облегчить развертывание

приложения. Фактически вся H2, включая исходный код и примеры, умещается в файле размером 5 Мбайт. Использование H2 возможно даже из браузера.

СУБД H2 весьма полезна в разработке, так как позволяет развернуть базу данных прямо в памяти компьютера, оснащена развитыми возможностями запросов и многочисленными функциями, которыми обладают более тяжеловесные базы данных, и легко настраиваема. Документация H2 подробна и понятна. Еще одним важным преимуществом СУБД является ее легкая встраиваемость в приложение.

Пользовательский интерфейс H2 представлен на рисунке 3.2:

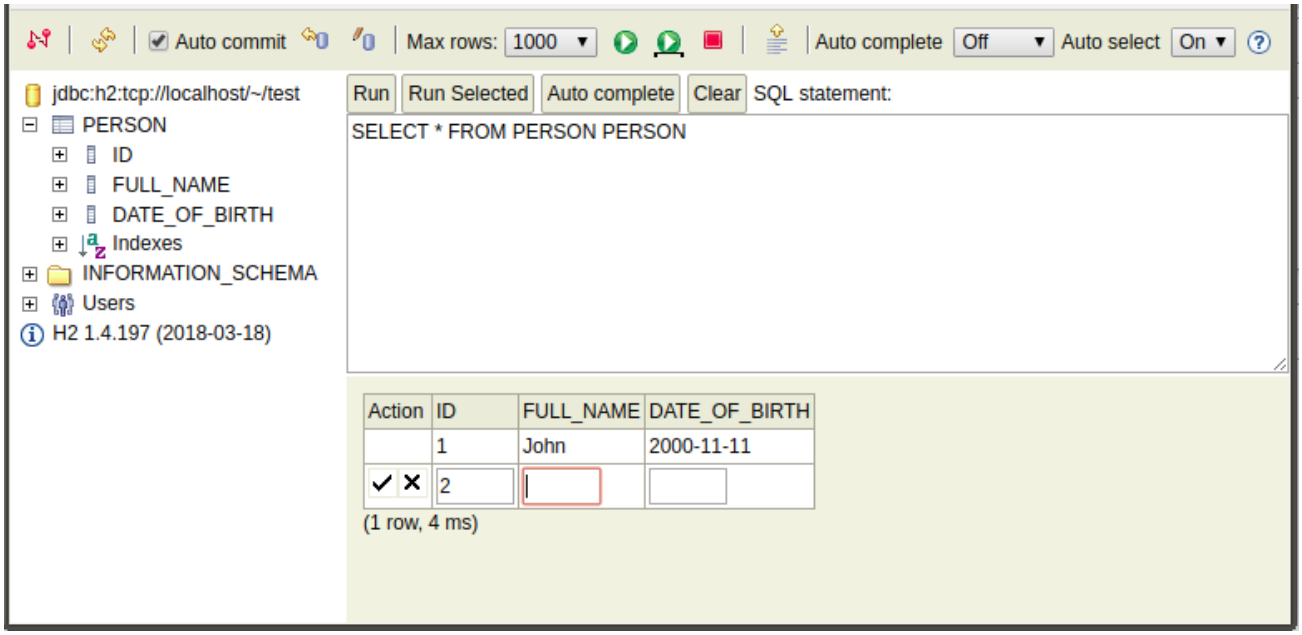


Рисунок 3.2 – Пользовательский интерфейс СУБД H2

Использование этой СУБД обусловлено ее особенностью и одновременно преимуществом - легкой интеграцией с Java-приложениями.

Физическая модель базы данных – это модель данных, созданная средствами конкретной СУБД, в данном случае – H2.

Модель была построена с помощью инструмента администрирования БД DbSchema.

DbSchema - это конструктор баз данных SQL и No-SQL с интерактивными диаграммами, документацией HTML и PDF, версиями и миграцией схем, просмотром реляционных данных, генератором случайных

данных, визуальным построителем запросов, редактором SQL и отчетами базы данных.

DbSchema использует свой собственный образ схемы отдельно от базы данных. Это позволяет развертывать схему на нескольких базах данных, управлять различными версиями одной и той же схемы и создавать сценарий миграции. Такие функции, как документация по векторным изображениям HTML5, виртуальные внешние ключи и просмотр реляционных данных, помогают документировать, понимать и исследовать базы данных.На ней показаны таблицы, первичные и вторичные ключи, связи, поля, типы данных. На рисунке 3.3 изображена физическая модель базы данных автоматизированной информационной системы:

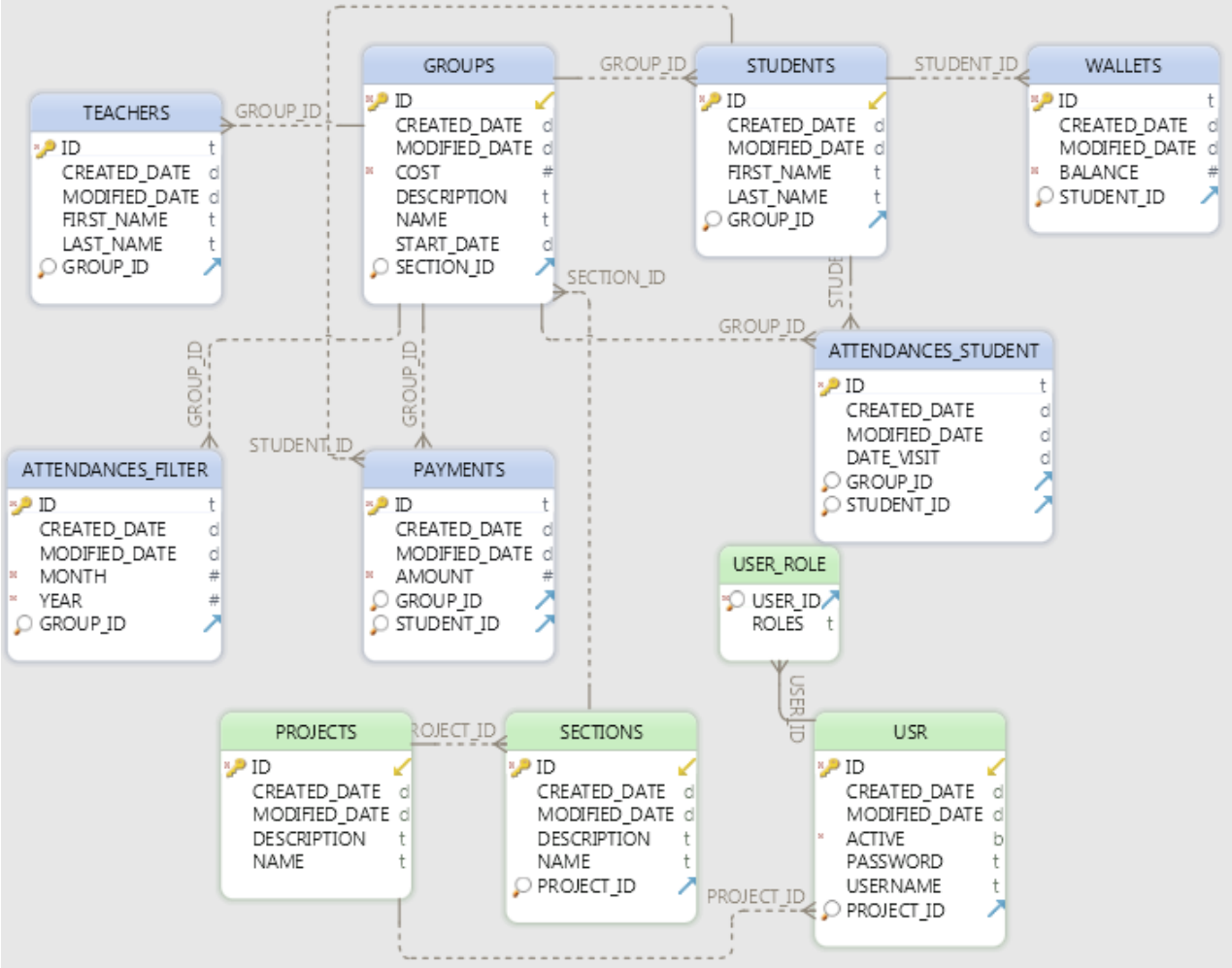


Рисунок 3.3 – Физическая модель базы данных АИС

На данном рисунке видна модель базы данных средствами СУБД H2.

Наибольшее число связей между сущностями информационной системы наблюдается по ключам проекта, секции и группы.

3.4 Разработка программного обеспечения автоматизированной информационной системы

3.4.1 Описание уровней обмена данными автоматизированной информационной системы

Для успешной и грамотной реализации приложения необходимо, чтобы его структура соответствовала современным стандартам веб-разработки. Информация при транспортировке между компонентами программы должна быть на определенном уровне данных, что обеспечит безопасность ее передачи и избавит от ошибок при работе программы.

Структура информационной системы, разделенная по пакетам в среде разработки Intelij IDEA, представлена на рисунке 3.4:

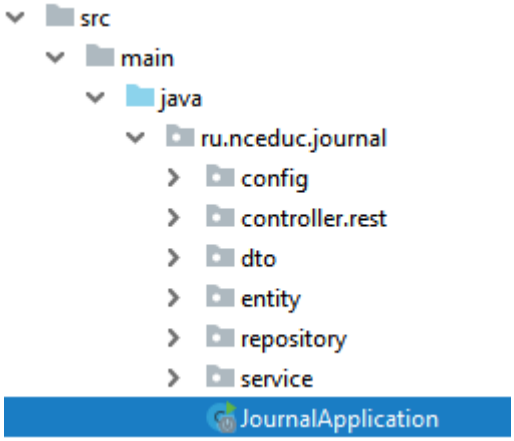


Рисунок 3.4 – Структура веб-приложения

- Entity – уровень сущностей приложения, самих объектов, хранящих данные;
- DTO (Data Transfer Object) – уровень объектов, необходимых для безопасного представления данных на уровень клиента
- Repository – уровень репозиториев, отвечающих за работу с сущностями;
- Service – уровень сервисов, взаимодействующих с данными из репозиториев и конвертирующих их в безопасные объекты DTO.

- Controller – уровень контроллеров, отвечающих за управление методов сервисов, вызываемых при определенных запросах со стороны клиента.

При работе с конвертацией данных из сущностей в DTO речь идет об использовании стороннего интерфейса, называемого Model Mapper, реализация которого позволяет получить легкий в использовании инструмент перевода данных в DTO. Конфигурация этого интерфейса представлена на рисунке 3.5:

```
@Configuration
public class ModelMapperConfiguration {
    @Bean
    public ModelMapper modelMapper() {
        ModelMapper modelMapper = new ModelMapper();
        modelMapper.getConfiguration().setPropertyCondition(Conditions.isNotNull());
        return modelMapper;
    }
}
```

Рисунок 3.5 – Конфигурация Model Mapper

Описанные в разделе способы реализации обеспечивают безопасное перемещение данных в процессе работы веб-приложения.

3.4.2 Описание веб-слоя автоматизированной информационной системы

API – это набор структур, классов и методов, представляющих собой промежуточный слой в трехуровневой клиент-серверной архитектуре АИС на основе Web-технологий. Представляет собой описание программного каркаса и интернет-протоколов.

Удобство использования данной методологии обусловлено, во-первых, общей стандартизацией, что гарантирует одинаково хорошую работоспособность проектов при использовании одного API, а во-вторых, возможностью ее переноса между сторонними приложениями и доработки, например, созданием пользовательского интерфейса и привязкой к компонентам API.

На рисунке 3.6 представлена схема работы API в веб-приложении:

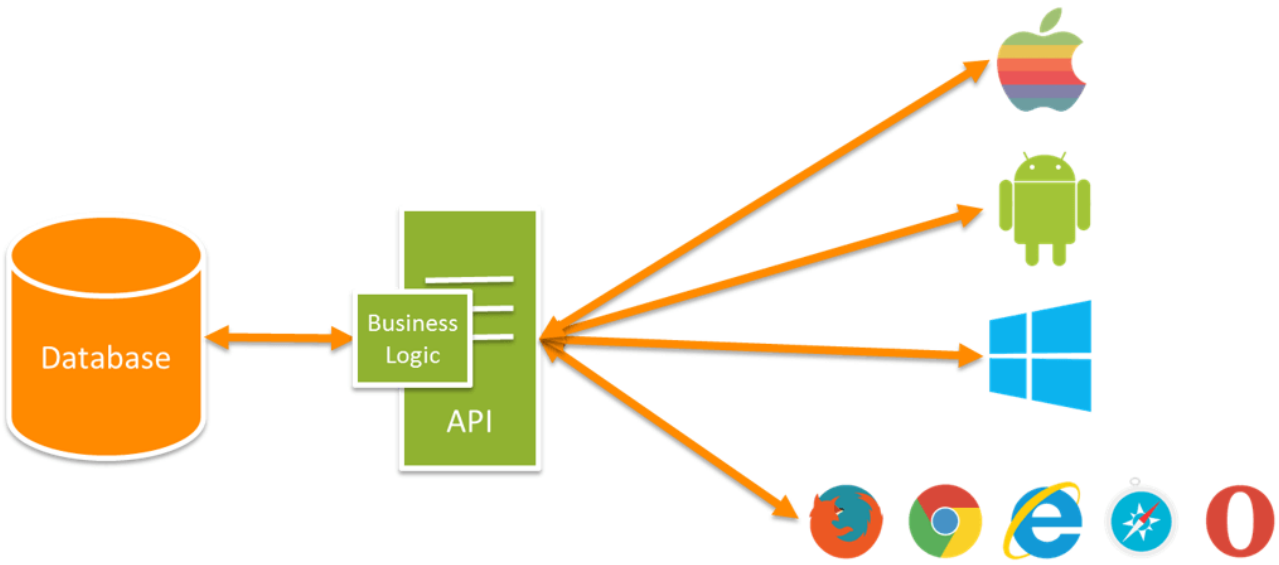


Рисунок 3.6 – Схема работы API

Через API клиент-браузер производит взаимодействие с бизнес-слоем системы и СУБД, получая запросы от пользователя.

Далее речь пойдет о таком интерфейсе программного обеспечения, как REST API. Использование данного интерфейса обусловлено его удобством для межпрограммного взаимодействия.

REST (RESTful) - это общие принципы организации взаимодействия приложения/сайта с сервером посредством протокола HTTP. Особенность REST в том, что сервер не запоминает состояние пользователя между запросами - в каждом запросе передаётся информация, идентифицирующая пользователя и все параметры, необходимые для выполнения операции.

Клиент посылает на API запрос (request). Он обрабатывается бизнес-логикой и отправляет ответ (response).

Всё взаимодействие с сервером сводится к 4 операциям:

1. получение данных с сервера (обычно в формате JSON, или XML)
2. добавление новых данных на сервер
3. модификация существующих данных на сервере
4. удаление данных на сервере

На рисунке 3.7 представлен принцип работы REST API:

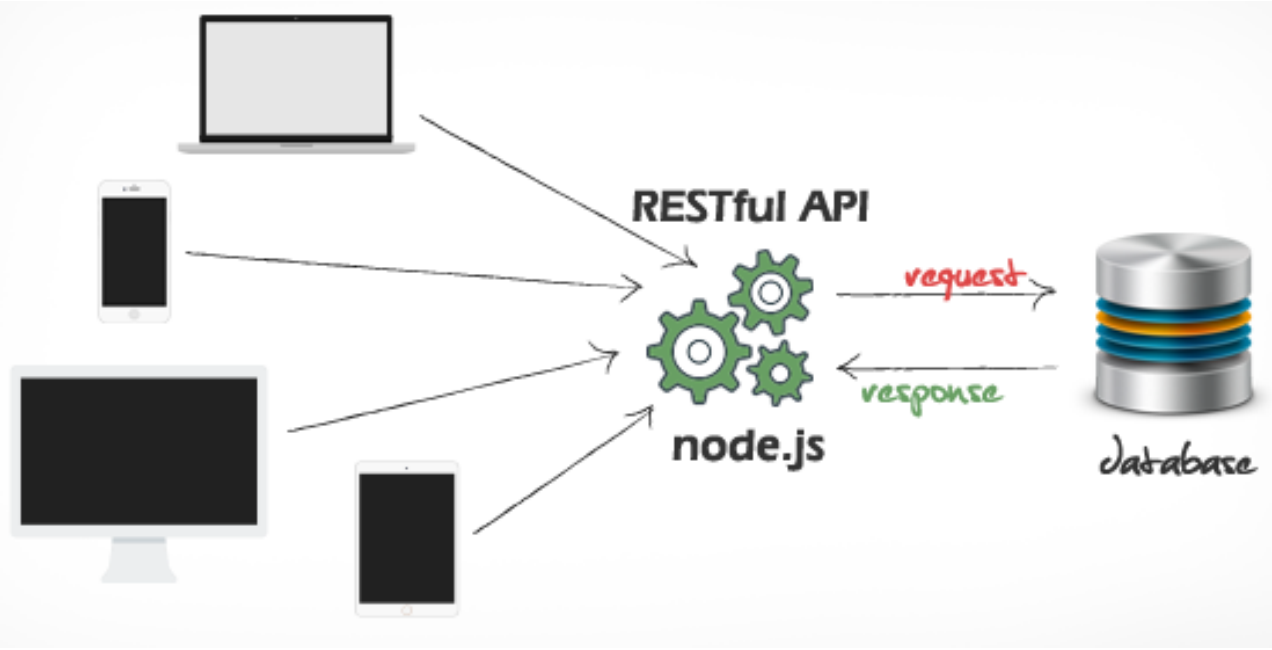


Рисунок 3.7 – Схема работы RESTful API

Для каждого типа операции используется свой метод HTTP-запроса:

- получение - GET
- добавление - POST
- модификация - PUT
- удаление – DELETE.

Код ответа сервера зависит от типа запроса и успешности его выполнения. Список кодов HTTP представлен ниже:

- 200 OK — ответ на успешные GET, PUT, PATCH или DELETE. Этот код также используется для POST, который не приводит к созданию;
- 201 Created — этот код состояния является ответом на POST, который приводит к созданию;
- 204 - Нет содержимого. Это ответ на успешный запрос, который возвращает пустое тело без данных (например, запрос DELETE);
- 304 Not Modified — код состояния, когда заголовки HTTP-кеширования находятся в работе;
- 400 Bad Request — код состояния указывает, что запрос искажен, например, если тело не может быть проанализировано;

- 401 Unauthorized — Если не указаны или недействительны данные аутентификации;
- 403 Forbidden — когда аутентификация прошла успешно, но аутентифицированный пользователь не имеет доступа к ресурсу;
- 404 Not found — если запрашивается несуществующий ресурс;
- 405 Method Not Allowed — когда запрашивается HTTP-метод, который не разрешен для аутентифицированного пользователя;
- 410 Gone — этот код состояния указывает, что ресурс в этой конечной точке больше не доступен;
- 415 Unsupported Media Type - если в качестве части запроса был указан неправильный тип содержимого;
- 422 Unprocessable Entity — используется для проверки ошибок;
- 429 Too Many Requests — когда запрос отклоняется из-за ограничения скорости.

Разработчик может использовать получаемые HTTP-коды для эффективного отображения результата запроса (возникшей ошибки или успешного выполнения) пользователю в понятном представлении.

3.4.3 Реализация контроля доступа к системе

Для обеспечения сохранности данных необходимо предусмотреть способ контроля доступа к хранимой в системе информации. Для этого нужно создать алгоритм регистрации и авторизации пользователей. В фреймворке Spring есть встроенный инструмент, созданный для решения данной задачи, называемый Spring Security. Как и все Spring проекты, настоящая сила Spring Security в том, что он может быть легко дополнен нужным функционалом.

Данный инструмент использовался для создания механизмов контроля доступа в приложении. Результат работы метода регистрации – созданный в базе данных пользователь Login с паролем Pass – представлен на рисунке 3.8:

	CREATED_DATE	MODIFIED_DATE	ACTIVE	PASSWORD	USERNAME
1	2019-05-28 19:56:33.977	2019-05-28 19:56:33.977	TRUE	$2a$08$nBsmcwapI7ruEx1gOTGu0.b97Mg6hM5M6JvkNYfCzTImf6qmrHbUu	Login

Рисунок 3.8 – созданный пользователь

При создании пользователя в базу данных записывается его информация, такая как логин и пароль в зашифрованном виде с помощью встроенного инструмента Spring Security PasswordEncoder.

3.4.4 Документирование API приложения

Для разработанного программного интерфейса необходимо создать документацию для облегчения ее использования при дальнейшей разработке системы.

Для решения данной задачи использовался инструмент Swagger. С помощью специальных аннотаций он документирует методы контроллеров и генерирует HTML-страницу с полученными описаниями этих методов и кодами ответа HTTP-сервера, а так же позволяет протестировать метод прямо на сгенерированной странице. Пример документирования API изображен на рисунке 3.9:

```
@ApiOperation(value = "Get all teachers by ID of group")
@GetMapping("/by-group/{id}")
public ResponseEntity<List<TeacherDTO>> getAllTeachersByGroupId(@PathVariable String id) {
    return new ResponseEntity<>(teacherService.getAllByGroupId(id), HttpStatus.OK);
}

@ApiOperation(value = "Create a new teacher")
@PostMapping("/")
public ResponseEntity<TeacherDTO> createTeacher(@RequestBody TeacherDTO teacherDTO) {
    Optional<TeacherDTO> optionalDTO = teacherService.create(teacherDTO);
    return optionalDTO.map(dto -> new ResponseEntity<>(dto, HttpStatus.CREATED))
```

Рисунок 3.9 – Документирование методов контроллера учителей

С помощью аннотации @ApiOperation методу можно добавить желаемое описание. Данное действие необходимо для понимания незнакомым с системой разработчикам принципа работы API.

Результаты работы Swagger были представлены на рисунках 3.10 и 3.11:

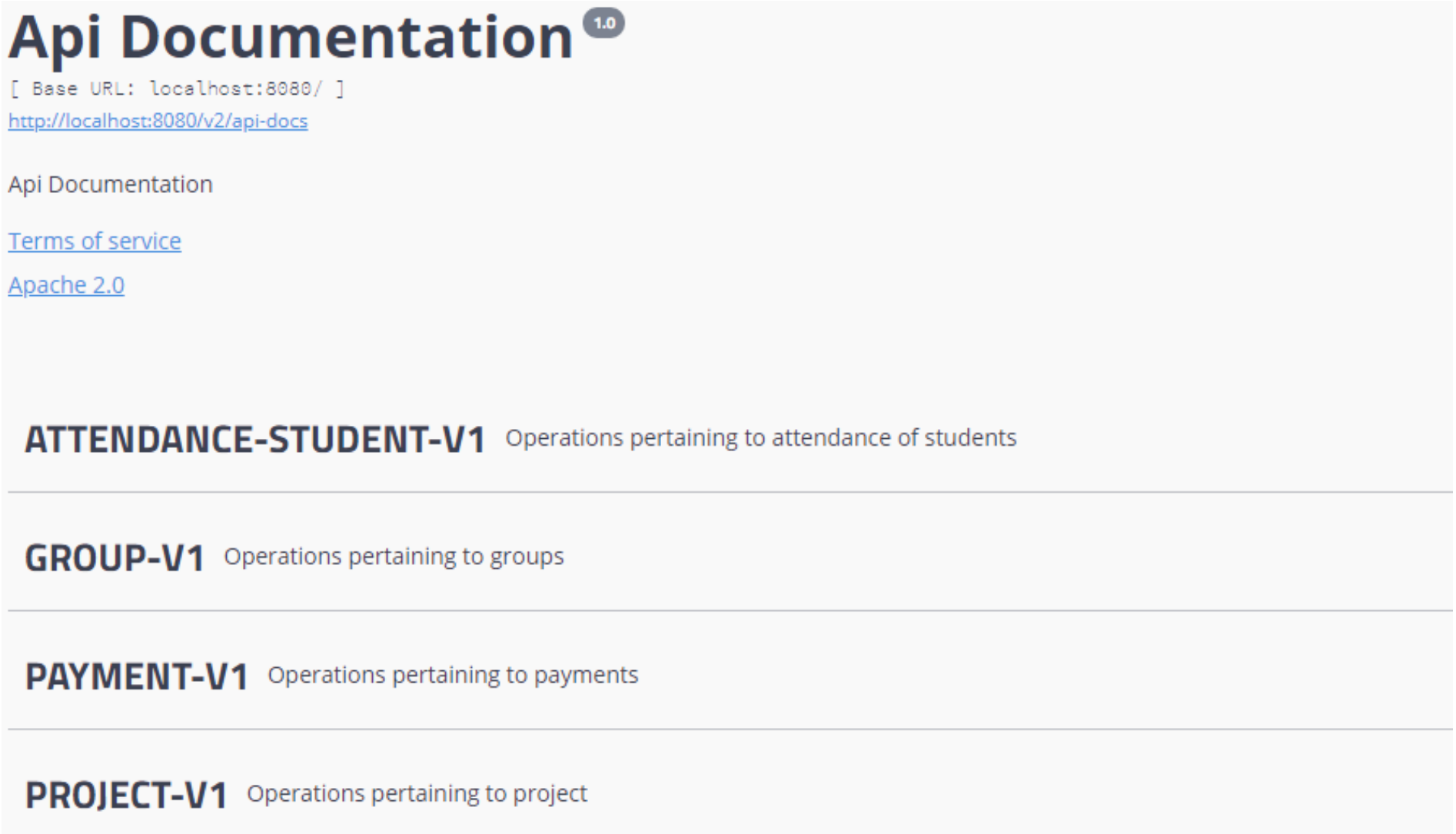


Рисунок 3.10 – Страница swagger-ui.html с частью контроллеров веб-приложения и их описанием

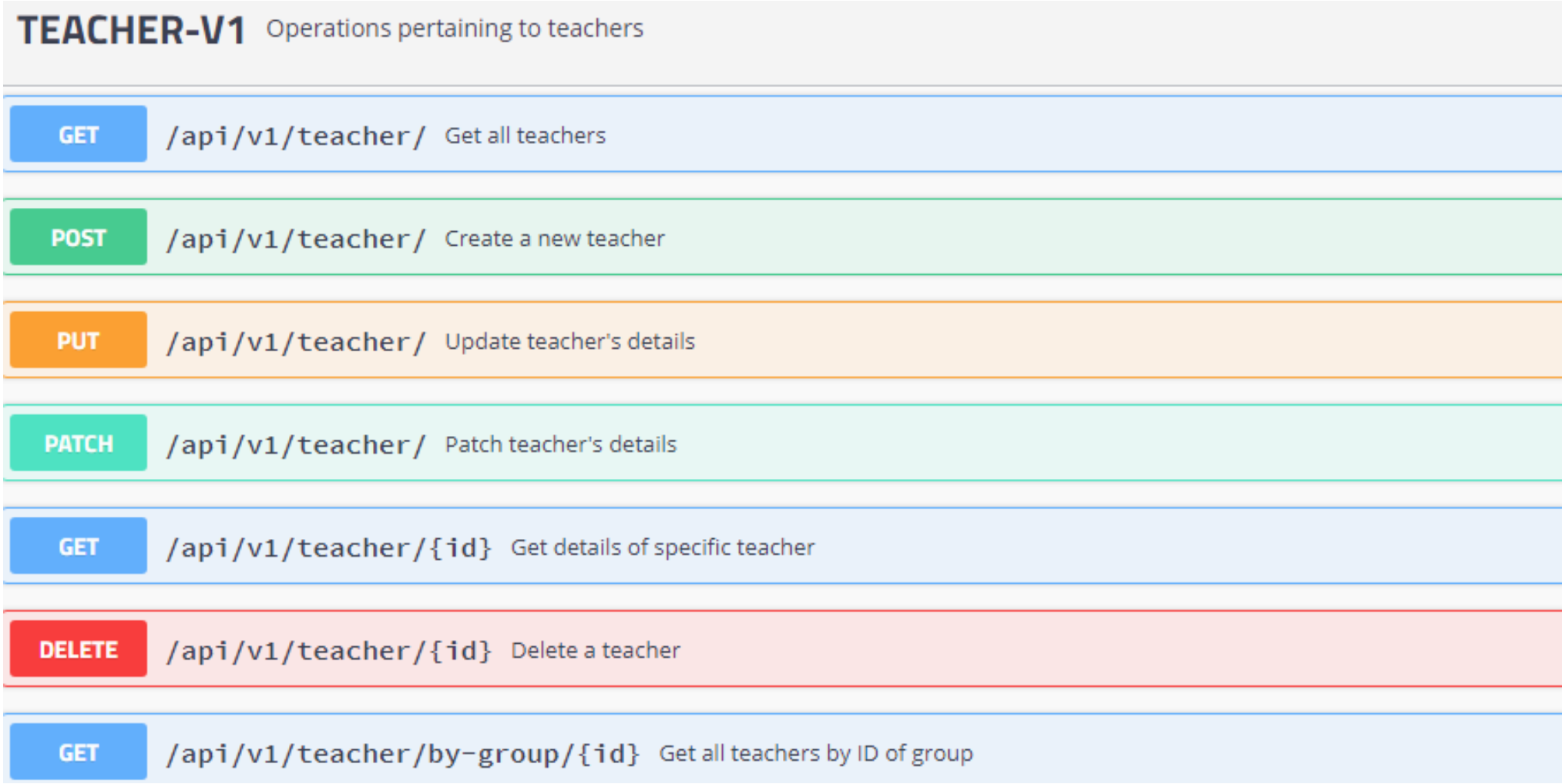


Рисунок 3.11 – Методы контроллера работы с учителями в веб-приложении

Документирование API с помощью Swagger помогло в дальнейшем тестировании приложения, а так же облегчило задачу разработки пользовательского интерфейса.

3.4.5 Разработка пользовательского интерфейса

После реализации Backend-части приложения – программного интерфейса API – необходимо разработать так же и пользовательский интерфейс, который будет отображаться в браузере. Для этого будут использованы стандартные языки при разработке веб-страниц – HTML для элементов интерфейса и CSS для стилей этих элементов. В дополнение к нем применяются Bootstrap, Vue.js, Axios.

Bootstrap позволит сэкономить время на верстку веб-сайта путем использования готовых элементов (кнопок, панелей, заголовков), которые можно получить прямо на официальном сайте инструмента и вставить непосредственно в HTML-код страницы. Vue.js реализовывает всю Frontend-част приложения. На рисунках 3.7 – 3.11 представлен разработанный веб-интерфейс приложения:

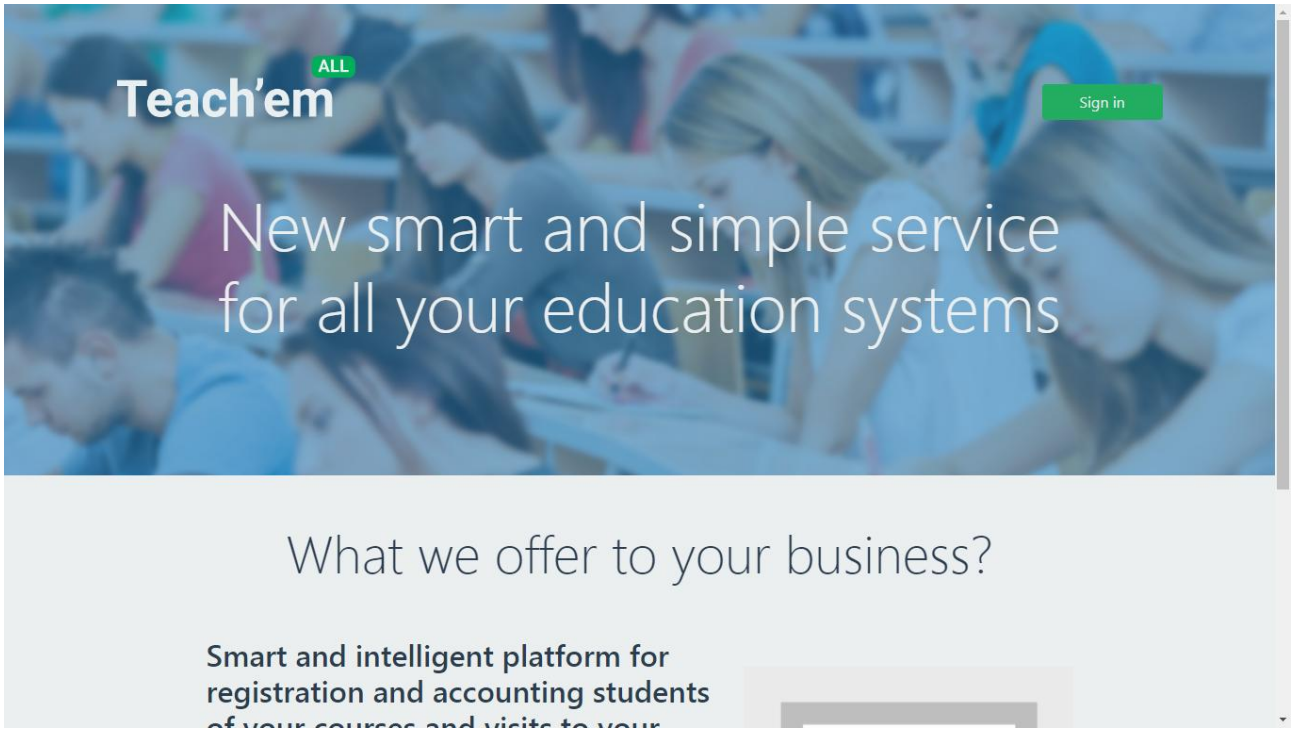


Рисунок 3.12 – Главная страница информационной системы

Главная страница является ознакомительной и в дальнейшем может быть доработана более детально.

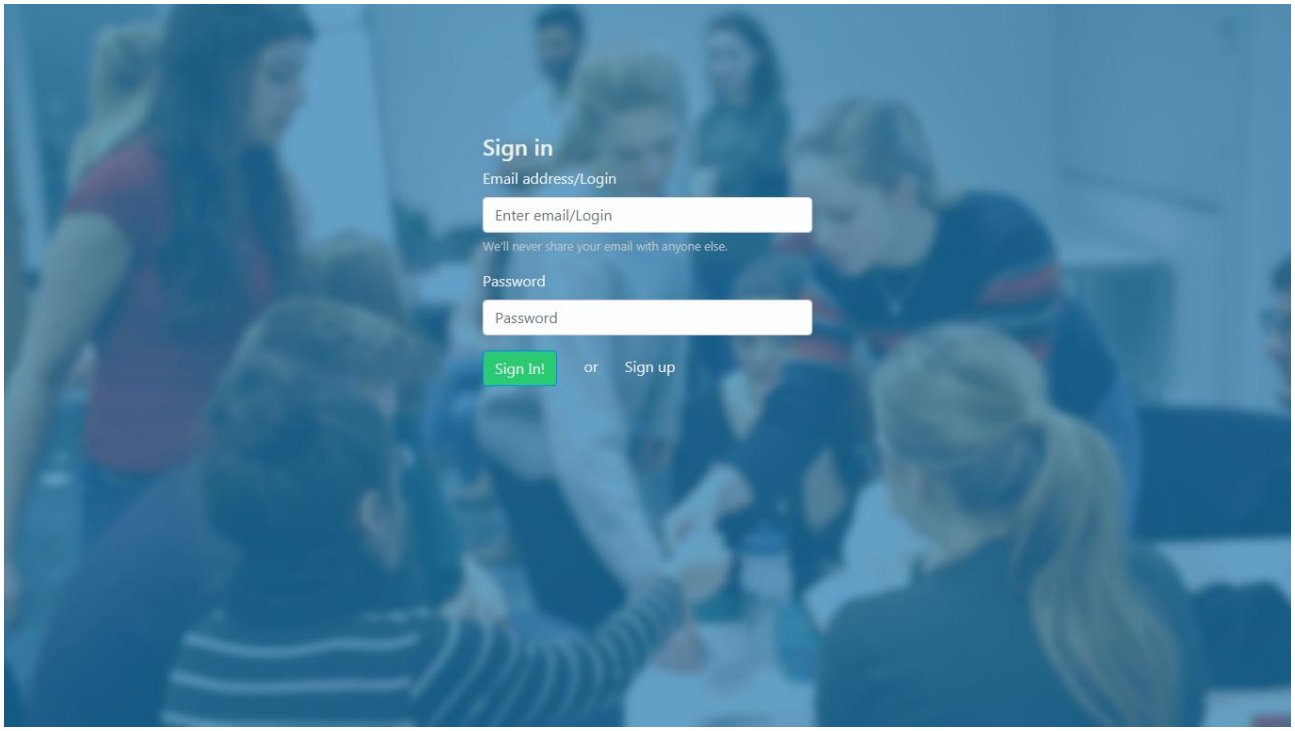


Рисунок 3.13 – Страница авторизации в информационной системе

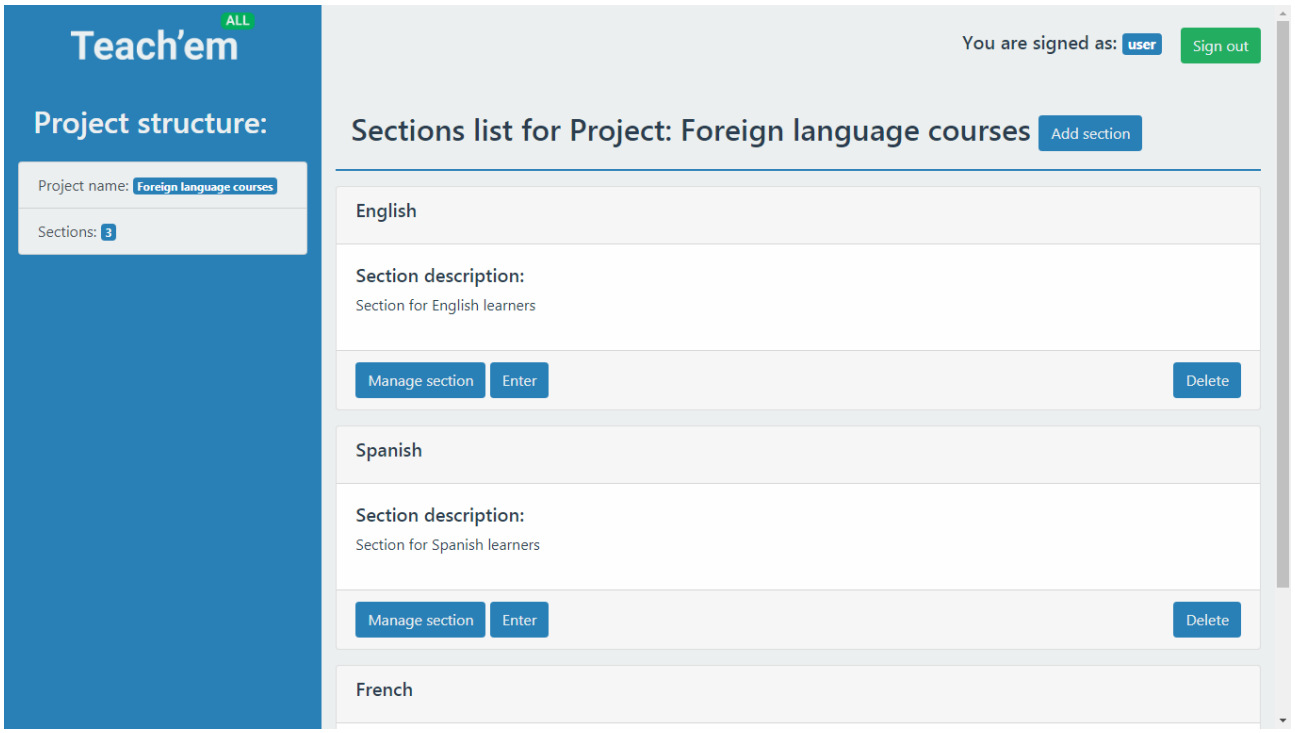


Рисунок 3.14 – Страница секций иностранных языков

Большая часть элементов интерфейса были реализованы при помощи Bootstrap.

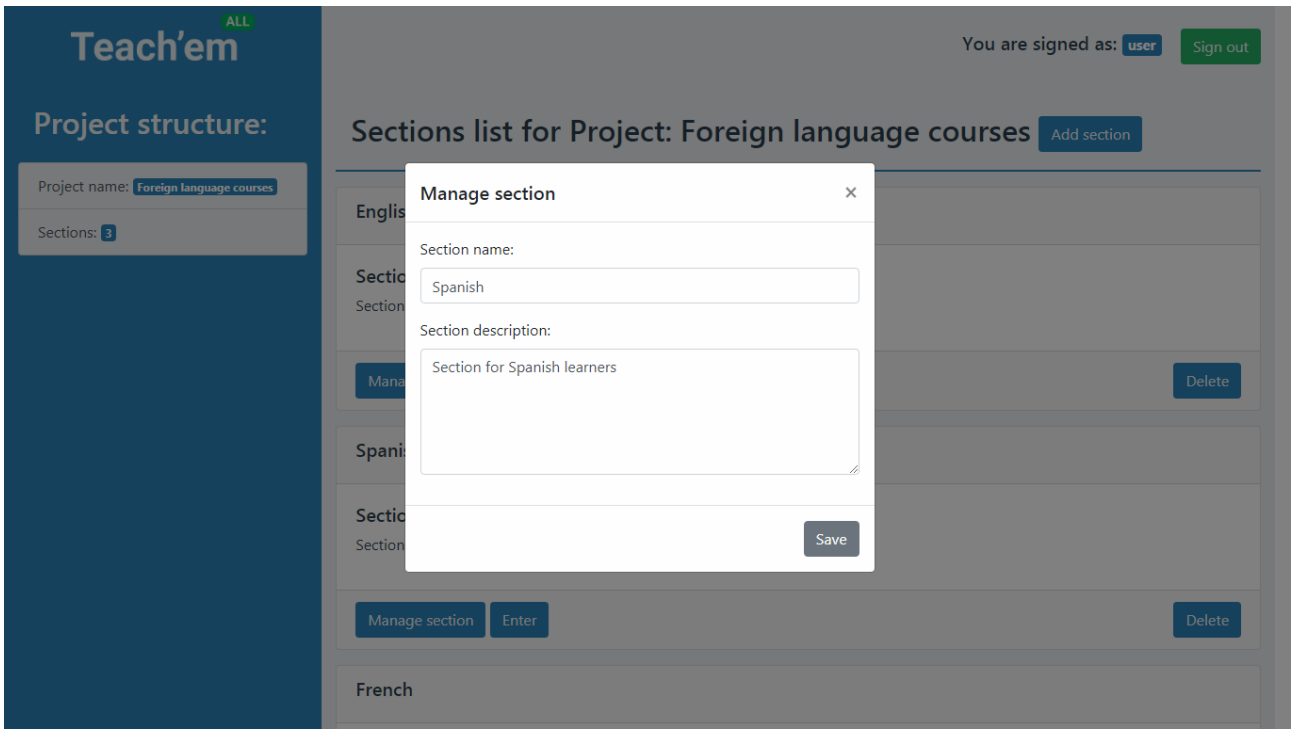


Рисунок 3.15 – Форма редактирования информации о секции испанского языка

Первое, что важно отметить – с помощью данного фреймворка удалось реализовать динамическую загрузку веб-страницы. Это значит, что для выполнения действий по обработке и изменению данных отсутствует необходимость постоянно перезагружать страницы. Данное преимущество позволит сэкономить существенное количество трафика при установке приложения на реальный веб-сервер учреждения дополнительного образования.

Второе – с помощью Vue component'ов были реализованы все элементы в списках, которые видны на экране. Доступ к API осуществляется с помощью методов компонентов Vue, в которых задействуется упомянутый Axios, который и отправляет полноценные HTTP-запросы на сервер. Пример использования axios для PUT-запроса на API представлен на рисунке 3.16:

```
axios.put('/api/v1/section/', {
    id: document.getElementById( elementId: "edit_section_id").value,
    name: document.getElementById( elementId: "edit_section_name").value,
    description: document.getElementById( elementId: "edit_section_description").value,
    projectId: document.getElementById( elementId: "edit_section_project_id").value
})
```

Рисунок 3.16 – Пример использования инструмента Axios

Демонстрации работы всех слоев данных АИС необходимо в первую очередь отследить весь путь, который они проходят при выполнении обычного запроса со стороны клиента. На рисунке 3.17 представлена страница секций с активной формой добавления новой:

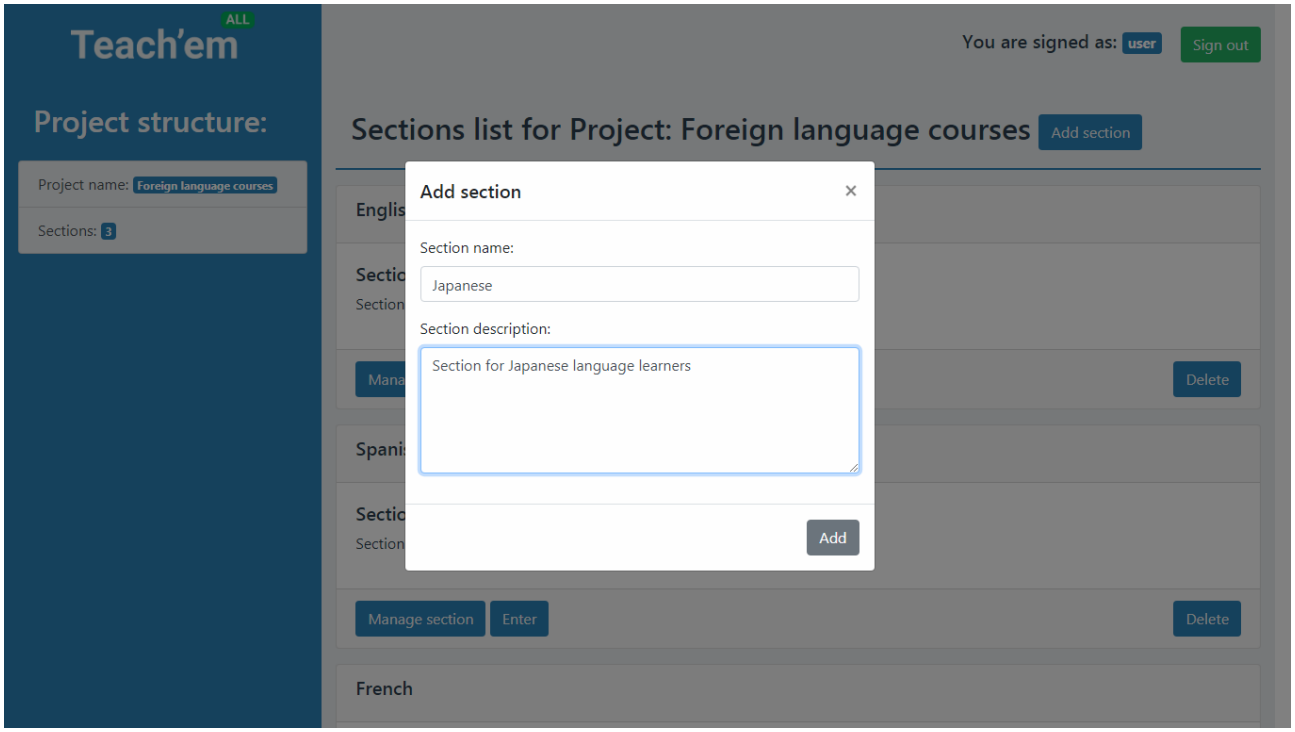


Рисунок 3.17 – Добавление новой секции в проект

После нажатия кнопки «Add» на сервер отправляется POST запрос с помощью Axios с JSON-телом, где name – имя секции, description – описание.

После выполнения запроса в консоли JavaScript браузера Google Chrome можно отследить полученный ответ от сервера, изображенный на рисунке 3.18:

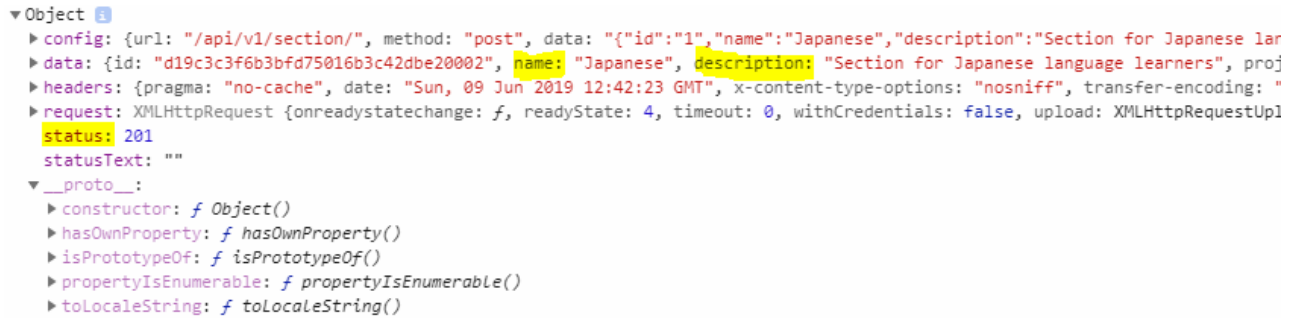


Рисунок 3.18 –Успешный response от сервера

Статус 201 свидетельствует об успешном выполнении запроса.

Наблюдать создание новой секции можно в базе данных, скриншот которой показан на рисунке 3.19:

CREATED_DATE	MODIFIED_DATE	DESCRIPTION	NAME	PRC
2019-06-09 14:26:35.912	2019-06-09 14:26:35.912	Section for English learners	English	8ad
2019-06-09 14:26:35.912	2019-06-09 14:26:35.912	Section for Spanish learners	Spanish	8ad
2019-06-09 14:26:35.912	2019-06-09 14:26:35.912	Section for French learners	French	8ad
2019-06-09 15:42:23.058	2019-06-09 15:42:23.058	Section for Japanese language learners	Japanese	8ad

Рисунок 3.19 – Успешное добавление секции японского языка в БД

Данные прошли весь путь согласно заявленной архитектуре: сначала из браузера сформировался запрос, который был отправлен на API в соотвествующий контроллер, работающий с секциями. Там был выполнен POST-запрос, создавший новую секцию и отправивший в качестве уведомления об успешности операции response с кодом 201 обратно клиенту. После этого был задействован уровень сервисов, который вызывает уровень репозиториев, где и производится запрос к базе данных на создание сущности.

Более подробный разбор пользовательского интерфейса продемонстрирован в следующей подглаве 3.6.

3.5 Описание функциональности автоматизированной информационной системы

Основной функцией АИС является безопасное и корректное обеспечение работы с данными учреждения дополнительного образования.

Система обладает также и бизнес-функцией – учетом оплаты занятий студентами и расписания посещения. Данный компонент был реализован вручную и не является полноценной платежной системой, поэтому его функционал ограничен, но достаточен для демонстрации возможностей будущих интеграций с реальными сервисами оплаты. На рисунке 3.20 изображена страница группы среднего уровня английского языка с преподавателем и двумя студентами. Для демонстрации работы компонента

оплаты продлим оплату занятий студента на два месяца вперед (рисунки 3.21 – 3.22):

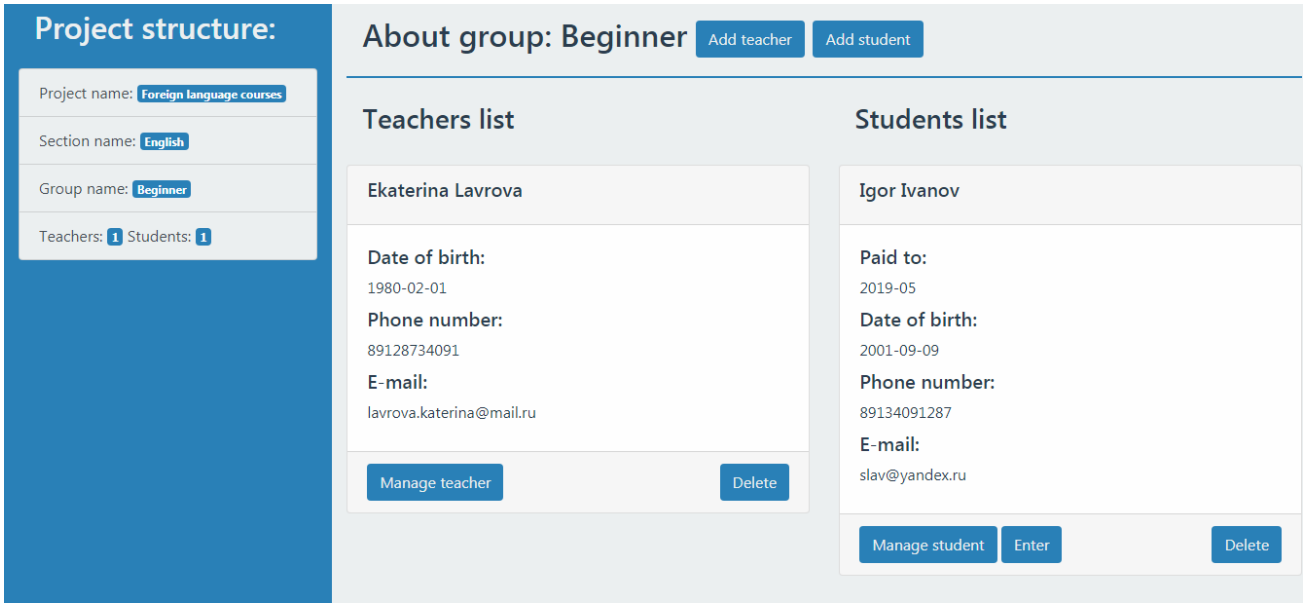


Рисунок 3.20 – Страница группы начального уровня по английскому языку

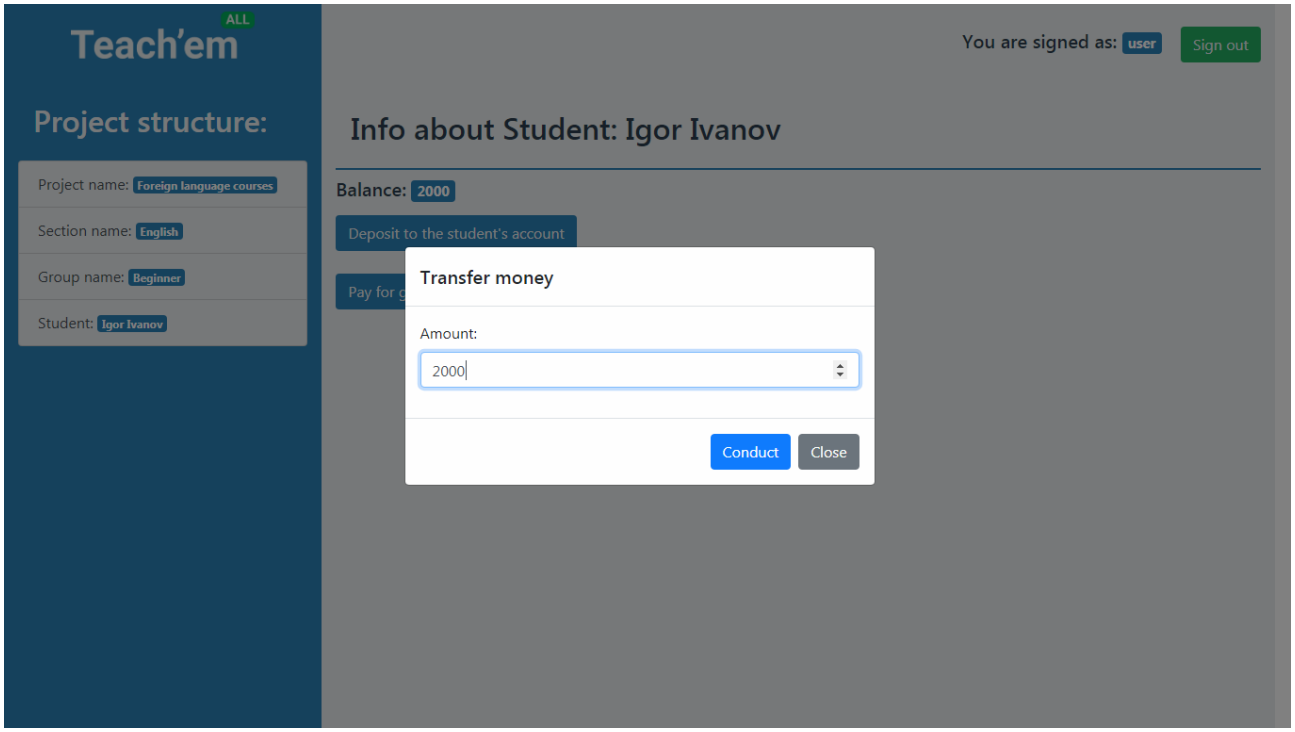


Рисунок 3.21 – Страница студента Petr Ivanov

Далее проследим изменение оплаченной даты в карточке студента.

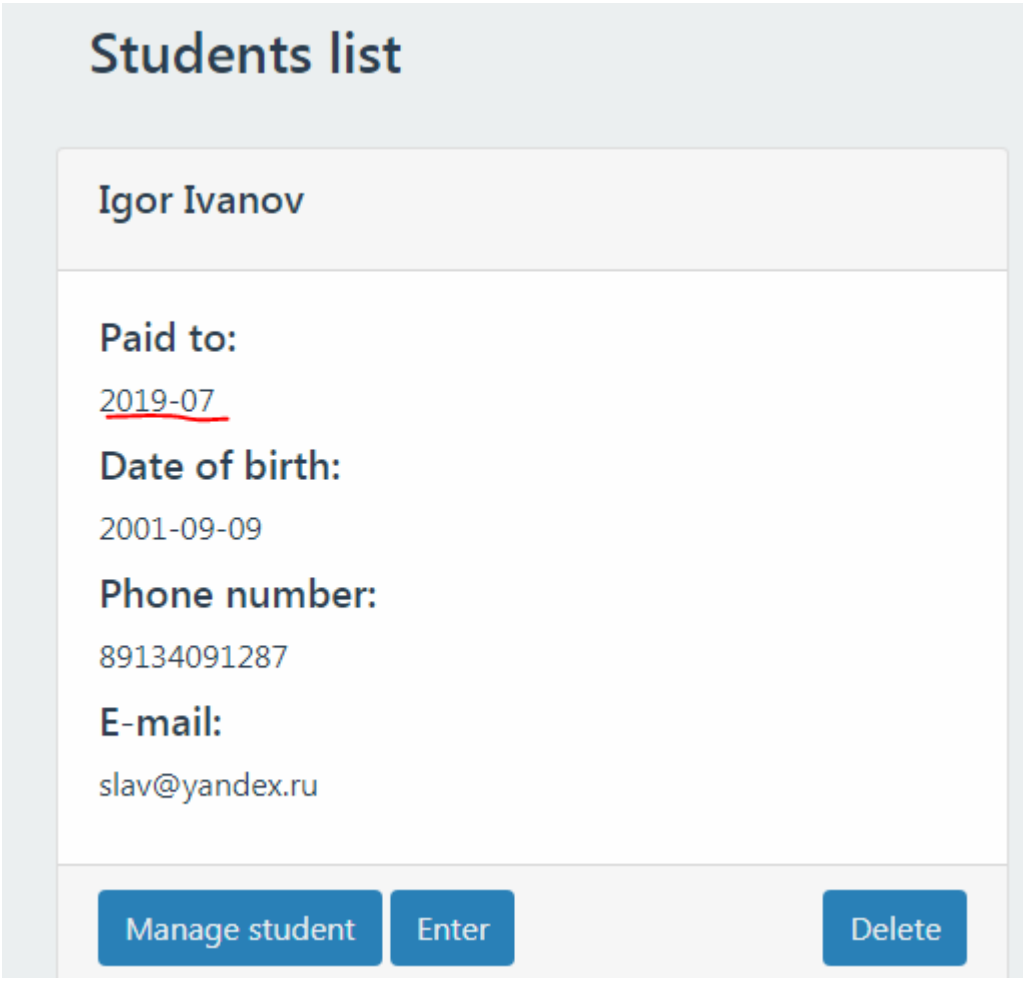


Рисунок 3.22 – Успешное продление оплаты студента Igor Ivanov на 2 месяца

В базе данных так же появилась соответствующая запись об успешной операции, изображенная на рисунке 3.23:

	AMOUNT	GROUP_ID	STUDENT_ID
5.823	2000	69a30937-6e85-4a9a-995c-cb03e1e4c4dd	de1bbc33-7860-4ced-bb43-722be7b85b6b

Рисунок 3.23 – Добавленная в БД запись об оплате

Дата старта занятий определяется при создании группы. У всех студентов, относящейся к ней, по умолчанию дата оплаты соответствует дате начала занятий. Стоимость месяца занятий в этой группе 1000 единиц. Сначала студенту было зачислено 2000 единиц на «баланс», после чего с него был

осуществлен перевод этих средств на группу. Сервер обработал операцию и увеличил дату оплаты на соответствующее число месяцев.

3.6 Тестирование модулей автоматизированной информационной системы

3.6.1 Выбор методов тестирования веб-приложения

Существует множество видов тестирования программного обеспечения, каждый из них подходит под определенную ситуацию при разработке.

Для тестирования API данной АИС было выбрано модульное тестирование.

Модульное тестирование – это тестирование каждой функционального компонента информационной системы отдельно, в искусственно созданной среде. Создается такая среда с помощью двух компонентов: драйвера и заглушки.

- Драйвер – это модуль теста, который запускает и выполняет тестируемый элемент приложения;
- Заглушка – это модуль теста, который имитирует рабочую среду, симулируя обмен данными с тестируемым элементом приложения.

Целью данного тестирования является проверка каждого компонента приложения как самостоятельной единицы. Из этой особенности вытекает и преимущество данного вида тестирования, которое послужило для его выбора – возможность тестирования каждого слоя обмена данными отдельно.

Для проверки Frontend-части приложения будет использоваться ручное тестирование – моделирование действий пользователя без использования дополнительных программных средств.

3.6.2 Описание результатов тестирования АИС

Модульное тестирование было реализовано с помощью компонента фреймворка Spring под названием Spring Boot Test. Всего было протестировано 80 методов приложения.

Аннотация @Before позволяет описать действия, которые будут производиться при каждом запуске модуля теста. Обычно в этом методе создаются экземпляры классов. На рисунке 3.24 показан пример использования аннотации с соответствующим методом setUp():

```
@Before
public void setUp() {
    String sectionId = UUID.randomUUID().toString();

    sectionDTO = new SectionDTO(sectionId,  name: "English",  description
    firstGroup = new GroupDTO(UUID.randomUUID().toString(),  name: "Fi
    secondGroup = new GroupDTO(UUID.randomUUID().toString(),  name: "S
    thirdGroup = new GroupDTO(UUID.randomUUID().toString(),  name: "Th

    allGroups = new ArrayList<>();
    allGroups.add(firstGroup);
    allGroups.add(secondGroup);
    allGroups.add(thirdGroup);

    groupsBySection = new ArrayList<>();
    groupsBySection.add(firstGroup);
    groupsBySection.add(secondGroup);
```

Рисунок 3.24 – Использование аннотации Before

Аннотация @Test помечает сам метод теста, который будет выполняться. На риснке 3.25 показан пример программного кода теста:

```
@Test
public void getGroupsBySectionId() throws Exception {
    String id = sectionDTO.getId();
    Mockito.when(groupService.getAllBySectionId(id)).thenReturn(groupsBySection);

    mockMvc.perform(get( urlTemplate: mapping + "/by-section/" + id))
            .andExpect(status().isOk())
            .andExpect(content().contentType(MediaType.APPLICATION_JSON_UTF8))
            .andExpect(content().json(objectMapper.writeValueAsString(groupsBySection.toArray())));

    mockMvc.perform(get( urlTemplate: mapping + "/by-section/" + "invalidId"))
            .andExpect(status().isOk())
            .andExpect(content().contentType(MediaType.APPLICATION_JSON_UTF8))
            .andExpect(content().string( expectedContent: "[]"));
}
```

Рисунок 3.25 – Использование аннотации Test

Обычно в методе теста вызывается тестируемый метод, присваивается объекту, затем с помощью специальных классов Assert сравнивается

ожидаемый результат – то есть ранее созданные объекты в методе setup() – и полученный в результате работы метода. Однако в данном случае происходит тестирование контроллера, поэтому проверяется так же статус ответа, тип данных и вариант при некорректном Id при вызове метода.

Последней часто используемой аннотацией при модульном тестировании является @After, которая является противоположностью @Before – выполняется после каждого прогона тестового модуля. Чаще всего используется для очистки данных и удаления объектов для корректной работы метода setUp() при работе следующего теста. На рисунке 3.26 показан пример использования аннотации @After:

```
@After
public void after() {
    userRepository.deleteAll();
    userEntityOne = null;
    getUserEntitySecond = null;
    userDTO = null;
    project = null;
}
```

Рисунок 3.26 – Использование аннотации After

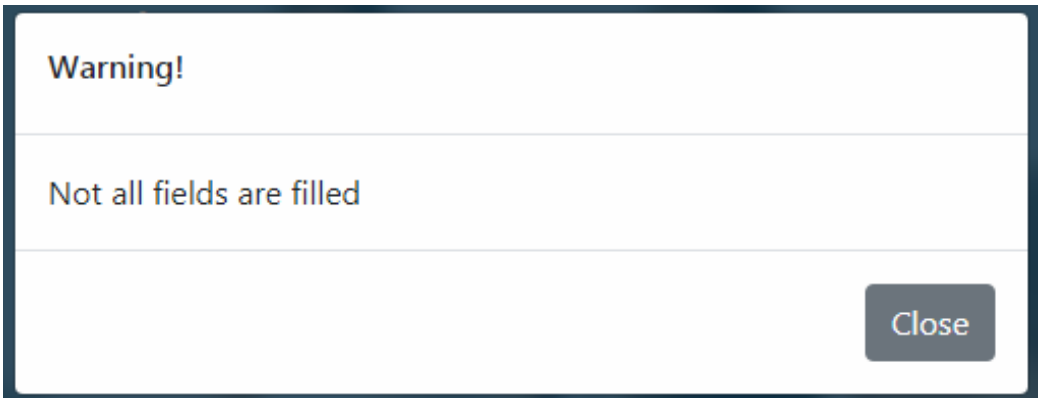
Результат модульного тестирования в среде разработки Intelij IDEA продемонстрирован на рисунке 3.27:

```
Results:

Tests run: 80, Failures: 0, Errors: 0, Skipped: 0

------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time: 01:10 min
Finished at: 2019-06-14T00:13:16+03:00
Final Memory: 47M/164M
------------------------------------------------------------------------
```

Рисунок 3.27 – Результаты модульного тестирования

Отсутствие ошибок свидетельствует о корректности написанных тестов и тестируемых компонентов, что подтверждает работоспособность и качество АИС.

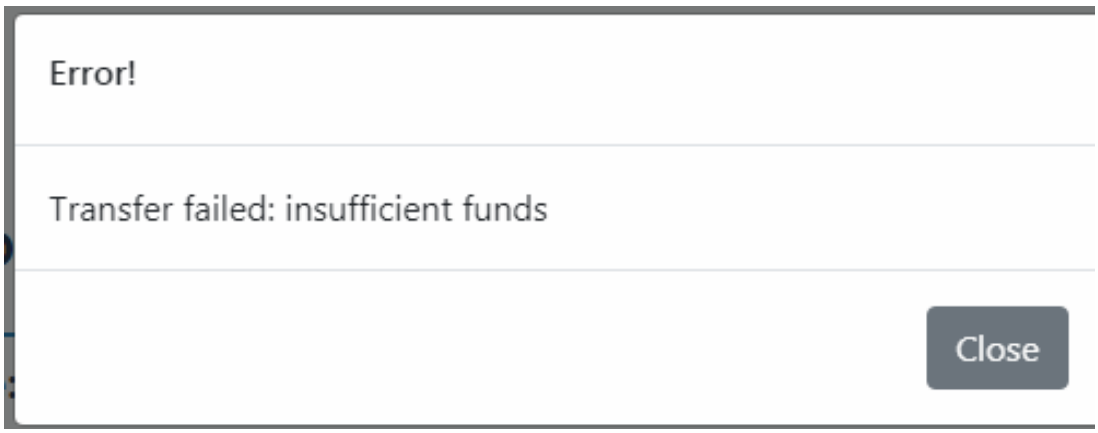
В результате тестирования пользовательского интерфейса был выявлен его недостаток – отсутствие необходимой валидации, которая позволяла бы пользователю понимать причину неудачи при выполнении отправленных запросов и помогала чувствовать себя увереннее при работе с системой. В связи с этим необходимая валидация была реализована в виде всплывающих окон. На рисунках 3.28 – 3.31 представлены результаты реализации валидации в АИС:



3.28 – Валидация при регистрации – «Не все поля заполнены»

3.29 – Валидация при удалении – Подтверждение удаления

3.30 – Валидация при удалении – «Ошибка: секция не пуста»



3.31 – Валидация при оплате занятий студента – «Ошибка перевода: недостаточно средств»

Реализация данных окон сообщений была выполнена с помощью обработки response от сервера. Если код ответа не соответствует успешному, то производится отрисовка соответствующего всплывающего окна.

Вывод по главе 3

В данной главе был произведен выбор архитектуры АИС, технологии разработки программного обеспечения и СУБД. После этого была построена физическая модель базы данных.

Затем была произведена разработка веб-приложения согласно поставленным ранее требованиям и выбранным технологиям. В процессе разработки было выполнено документирование API и реализация системы контроля доступа.

Следующим этапом была разработка пользовательского интерфейса, а затем – описание функционала Backend и Frontend частей приложения.

Завершающим этапом в разработке послужило тестирование программного обеспечения, входе которого были выявлены недостатки Frontend части, после чего она была доработана.

ЗАКЛЮЧЕНИЕ

Результатом выпускной квалификационной работы является разработанное веб-приложения для учета данных в работе учреждения дополнительного образования. Система предназначена для контроля и хранения информации об участниках и компонентах учреждения. Согласно поставленной задаче на разработку, информация хранится в БД, а общение с системой происходит через пользовательский интерфейс, доступный через браузер компьютера.

Начальным этапом проектирования был анализ предметной области, выявление требований к информационной системе и сравнение существующих ее аналогов, в результате чего была поставлена задача на разработку нового программного обеспечения. Все вышеперечисленные действия позволили построить концептуальную модель информационной системы.

Вторым этапом проектирования было построение логической модели системы и базы данных, которые изображались рядом диаграмм: вариантов использования, классов, последовательности. После этого были сформированы требования к аппаратно-программному обеспечению, необходимому для качественной работы приложения.

Завершающим этапом проектирования было определение архитектуры АИС, а так же технологий и инструментов, используемых для ее реализации. Была разработана физическая модель базы данных, отражающая связи между объектами средствами СУБД DbSchema. Затем была произведена реализация веб-приложения, в ходе которого было осуществлено документирование программного и разработка пользовательского интерфейсов. После этого были описаны функции реализованной системы. Последним этапом реализации было определение методов и осуществление тестирования приложения, в ходе которого были выявлены и осуществлены необходимые доработки для комфортной работы пользователя с информационной системой.

При разработке данного проекта были усовершенствованы навыки программирования на языке Java, а так же работы с различными веб-инструментами. Были изучены и использованы инструменты для платформы Java, необходимые для создания современных веб-приложений.

Подводя итог о проделанной работе, можно сказать, что поставленная цель достигнута, а задачи реализованы. В дальнейшем, данная программа может быть усовершенствована и внедрена в работу учреждений дополнительного образования.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

Нормативно-правовые акты:

1. ГОСТ 19.701-90. Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения (ИСО 5807-85). Введ. 1992-01-01.- М.: Изд-во стандартов, 1992. – 14с.

2. ГОСТ 34.003-90. Информационная технология. Комплекс стандартов на автоматизированные системы. Термины и определения. Взамен ГОСТ 24.003-84, ГОСТ 22487-77; Введ. 1992-01-01.- М.: Изд-во стандартов, 1992. – 14с.

3. ГОСТ 34.320-96. Информационные технологии. Система стандартов по базам данных. Концепции и терминология для концептуальной схемы и информационной базы. Введ. 2001-07-01.- М.: Изд-во стандартов, 2001. – 46с. - (Основополагающие стандарты).

4. ГОСТ 34.602-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы. – Введ. 1990-01-01.-М.: Изд-во стандартов, 1990. – 12с. - (Основополагающие стандарты).

Научная и методическая литература:

5. Мкртычев, С.В. Прикладная информатика. Бакалаврская работа: учеб.-метод. пособие / С.В. Мкртычев, О.М. Гущина, А.В. Очеповский. – Тольятти: ТГУ, 2017. – 77 с.

6. Барский А.Б. Параллельные информационные технологии: учебное пособие/ Барский А.Б.— Москва, Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017.— 503 с.

7. Журавлева Т.Ю. Информационные технологии: учебное пособие/ Журавлева Т.Ю — Саратов: Вузовское образование, 2018.— 72 с.

8. Еропкина А.С. Современные информационные технологии для автоматизации бизнес-процессов/ Еропкина А.С., Зобнин Ю.А.— Тюмень: Тюменский индустриальный университет, 2018.— 156 с.

9. Носова Л.С. Case-технологии и язык UML: учебно-методическое пособие/ Носова Л.С.— Челябинск, Саратов: Южно-Уральский институт управления и экономики, Ай Пи Эр Медиа, 2019.— 67 с.

10. Клочко И.А. Информационные технологии в профессиональной деятельности: учебное пособие для СПО/ Клочко И.А.— Саратов: Профобразование, Ай Пи Эр Медиа, 2019.— 292 с.

Электронные ресурсы:

11. Зиангирова Л.Ф. Сетевые технологии [Электронный ресурс]: учебно-методическое пособие/ Зиангирова Л.Ф.— Электрон. текстовые данные.— Саратов: Вузовское образование, 2017.— 100 с.— Режим доступа: http://www.iprbookshop.ru/62065.html.— ЭБС «IPRbooks»

12. Системы управления базами данных [Электронный ресурс]: лабораторный практикум/ — Электрон. текстовые данные.— Ставрополь: Северо-Кавказский федеральный университет, 2017.— 148 с.— Режим доступа: http://www.iprbookshop.ru/75595.html.— ЭБС «IPRbooks»

13. Волков Д.А. Базы данных [Электронный ресурс]: учебно-методическое пособие/ Волков Д.А.— Электрон. текстовые данные.— М.: МИСИ-МГСУ, Ай Пи Эр Медиа, ЭБС АСВ, 2018.— 77 с.— Режим доступа: http://www.iprbookshop.ru/79883.html.— ЭБС «IPRbooks»

14. Емельянова Т.В. Моделирование баз данных [Электронный ресурс]: учебное пособие/ Емельянова Т.В., Кольчатов А.М., Зюзина Н.Ю.— Электрон. текстовые данные.— Саратов: Ай Пи Эр Медиа, 2018.— 62 с.— Режим доступа: http://www.iprbookshop.ru/74560.html.— ЭБС «IPRbooks»

15. Разработка баз данных [Электронный ресурс]: учебное пособие/ А.С. Дорофеев [и др.].— Электрон. текстовые данные.— Саратов: Ай Пи Эр Медиа, 2018.— 241 с.— Режим доступа: http://www.iprbookshop.ru/70276.html.— ЭБС «IPRbooks»

Литература на иностранном языке:

16. Wall C.: Spring in Action, Fifth Edition, 2018. — 520 pages.

17. Bloch J.: Effective Java, 2017.

18. Macrae C.: Vue.js: Up and Running: Building Accessible and Performant Web Apps, 1st Edition, 2018. — 174 pages.

19. Hanchett E., Listwon B.: Vue.js in Action, 1st Edition, 2018. — 375 pages.

20. Raman R., Dewailly L.: Building RESTful Web Services with Spring 5 - Second Edition: Leverage the power of Spring 5.0, Java SE 9, and Spring Boot 2.0, 2018. — 228 pages.