**Министерство науки и высшего образования Российской Федерации ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «Национальный исследовательский университет ИТМО»**

**(Университет ИТМО)**

**Факультет** инфокоммуникационных технологий

**Образовательная программа** Мобильные и сетевые технологии

**Направление подготовки** 09.03.03 Прикладная информатика

**О Т Ч Е Т**

опроизводственной, технологической (проектно-технологической)практике

**Тема задания:** Проектирование и разработка серверной части системы управления учебной деятельностью

**Обучающийся,** Кулагина Светлана Викторовна, группа К34412

**Руководитель практики от университета:** Добряков Давид Ильич

Практика пройдена с оценкой \_\_**\_\_\_\_**

Дата 05.04.2024

Санкт-Петербург

2024

СОДЕРЖАНИЕ

[ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 3](#_Toc163140279)

[ВВЕДЕНИЕ 4](#_Toc163140280)

[1 Обзор аналогов 6](#_Toc163140281)

[2 Проектирование 10](#_Toc163140282)

[2.1 Построение диаграммы использования 10](#_Toc163140283)

[2.2 Построение диаграммы бизнес-процессов с использованием методологии IDEF0 11](#_Toc163140284)

[2.3 Построение диаграммы деятельности 12](#_Toc163140285)

[2.4 Построение диаграммы последовательности 14](#_Toc163140286)

[2.5 Проектирование диаграммы базы данных 19](#_Toc163140287)

[2.6 Выводы по второй главе 20](#_Toc163140288)

[3 Разработка серверной части 21](#_Toc163140289)

[3.1 Регистрация и авторизация 21](#_Toc163140290)

[3.2 Разработка функциональных возможностей преподавателя 24](#_Toc163140291)

[3.3 Разработка функциональных возможностей студента 28](#_Toc163140292)

[3.4 Вывод по третей главе 30](#_Toc163140293)

[ЗАКЛЮЧЕНИЕ 31](#_Toc163140294)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 32](#_Toc163140295)

**ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ**

В данном отчете о практике применяют следующие термины с соответствующими определениями:

**Валидация –** это процесс проверки данных, введенных пользователем, на соответствие заданным критериям

**Клиент** – это аппаратный или программный компонент вычислительной системы, посылающий запросы серверу

**Стэк** – это набор инструментов, применяющийся при работе в проектах и включающий языки программирования, фреймворки, системы управления базами данных, компиляторы и так далее

**Фреймворк** – это набор инструментов, библиотек и рекомендаций, предназначенных для разработки программного обеспечения

**Хеширование** – преобразование входного массива данных произвольной длины в выходную битовую строку фиксированной длины

**ACID** – набор требований к транзакционной системе, обеспечивающий наиболее надёжную и предсказуемую её работу — атомарность, согласованность, изоляцию, устойчивость

**API (Application Programming Interface)** – это совокупность инструментов и функций в виде интерфейса для создания новых приложений, благодаря которому одна программа будет взаимодействовать с другой

**GitHub** – это веб-платформа для совместной разработки программного обеспечения, предоставляющая инструменты, которые помогают программистам работать над проектами вместе в режиме реального времени

**JSON Web Token** – это открытый стандарт для создания токенов доступа, основанный на формате JSON

**Id (идентификатор)** – уникальный признак объекта, позволяющий отличать его от других объектов, то есть идентифицировать

**ВВЕДЕНИЕ**

Управление учебной деятельностью это большая работа, которая заключается не только в проверке решений и выставлении баллов, но и в организационных процессах, подготовке материалов и общем контроле обучения. На всю работу по организации и управлению учебной деятельностью уходит много времени, которое очень ценно в современных реалиях. Неправильная организация учебной деятельности вызывает множество проблем, связанных с коммуникацией, осведомленностью студентов, уровнем обучения и его своевременностью, что также приводит к еще большей нехватке времени из-за попыток решить эти проблемы. Для лучшего управления учебной деятельностью были разработаны системы, которые автоматизируют большинство организационных и учебных процессов, помогают сэкономить время, повысить уровень взаимного понимания, обучения и улучшить коммуникацию между преподавателем и студентами. Данные системы широко распространены как в университетах, так и в школах, однако они могут не подходить под определённые задачи, отчего спрос на разработку собственных систем до сих пор существует. Запрос на разработку такой системы был получен от преподавателя факультета инфокоммуникационных технологий по фронтенд и бэкенд разработке, Добрякова Давида Ильича. Функциональные требования, выдвинутые заказчиком, на данный момент не выполняются доступными в России системами управления обучением.

В настоящее время рассматривается вариант разработки общей системы управления учебной деятельностью с разделением доступа к функционалу в зависимости от роли: преподаватель и студент. Данная система позволит преподавателю вести отчетность по успеваемости студентов, подготавливать материалы, а также предоставлять доступ к данным материалам, в то время как студент сможет получать задания, следить за дедлайнами, записываться на варианты и решать контрольные.

Для реализации данной системы будет написан собственный бэкенд, реализующий необходимый функционал, который будет разграничен проверкой роли пользователя. Бэкенд планируется реализовывать на таких технологиях как: Express, фреймворк Node.js, и нереляционная база данных MongoDB. Фронтенд часть веб-приложения будет разработана на React.js с использованием необходимых хуков и библиотеки для работы с файлами. Данный стэк технологий называется “MERN” и расшифровывается как MongoDB, Express, React.js, Node.js [3]. Этот набор инструментов применим для fullstack-разработки, каждая часть данного набора подобрана для эффективной работы с JavaScript. Можно сказать, что MERN - основа для амбициозных идей, которые легко масштабируются благодаря веб-серверу на Node.js и экономно разрабатываются благодаря популярности и доступности языка JavaScript [1].

**Цель**: спроектировать диаграммы и разработать серверную часть системы управления дополнительной учебной деятельностью.

**Задачи**:

1. обзор существующих решений,
2. проектирование,
3. создание базы данных,
4. разработка серверной части.

**Планируемый результат в соответствии с Индивидуальным заданием**: в ходе работы должны быть спроектированы все необходимые для понимания разрабатываемой системы диаграммы и схемы, демонстрирующие ее функционал и строение в целом, и разработана серверная часть веб-приложения, отвечающая современным требованиям и требованиям преподавателя практики факультета инфокоммуникационных технологий, Добрякова Давид Ильича, именуемого в дальнейшем “Заказчик”.

# **Обзор аналогов**

Говоря об аналогах такого решения, тяжело остановиться на чём-то конкретном, поскольку требования, предъявляемые Заказчиком достаточно конкретные, строгие и четкие:

* система должна предоставлять возможность контроля активности студентов,
* система должна быть бесплатной,
* система должна предоставлять весь необходимый функционал по работе с контрольными, лабораторными и дополнительными заданиями,
* система должна предоставлять возможность контроля успеваемости студентов.

На данный момент не существует системы, которая подходила бы по всем указанным Заказчиком требованиям. Однако существует множество высокотехнологичных, хорошо продуманных, широко используемых систем, которые могли бы подойти, будь требования немного другими. Так, к примеру, система управления обучением “Контур Класс” является наиболее подходящей, ведь имеет возможность добавления студентов в систему по ссылкам-приглашениям (Рисунок 1, страницы 7 ), предоставляет возможности по созданию контрольных (Рисунок 2, страницы 7), заданий, отслеживанию успеваемости (Рисунок 3, страницы 7). Однако данная система является платной, самостоятельная запись на задания отсутствует и не ограничена количеством мест, отсутствует возможность генерации вариантов тестов, контрольных, предоставление доступа к заданиям производится по ФИО студента, что является неудобным решением и занимает большое количество времени, ведь заказчику необходимо предоставление доступа по учебным группам. К тому же в данной системе можно предоставить доступ только ко всему курсу целиком, что не соответствует логике учебного процесса (Рисунок 4, страницы 8). К примеру, группы обычно получают лабораторные задания постепенно, в зависимости от практических занятий и лекций, также работа организована и с контрольными, поэтому нельзя предоставить доступ к курсу всем группами одновременно, ведь тогда у какой-либо группы может быть преимущество в виде большего количество времени на работу с лабораторными или раннее открытие доступа к контрольной [7].

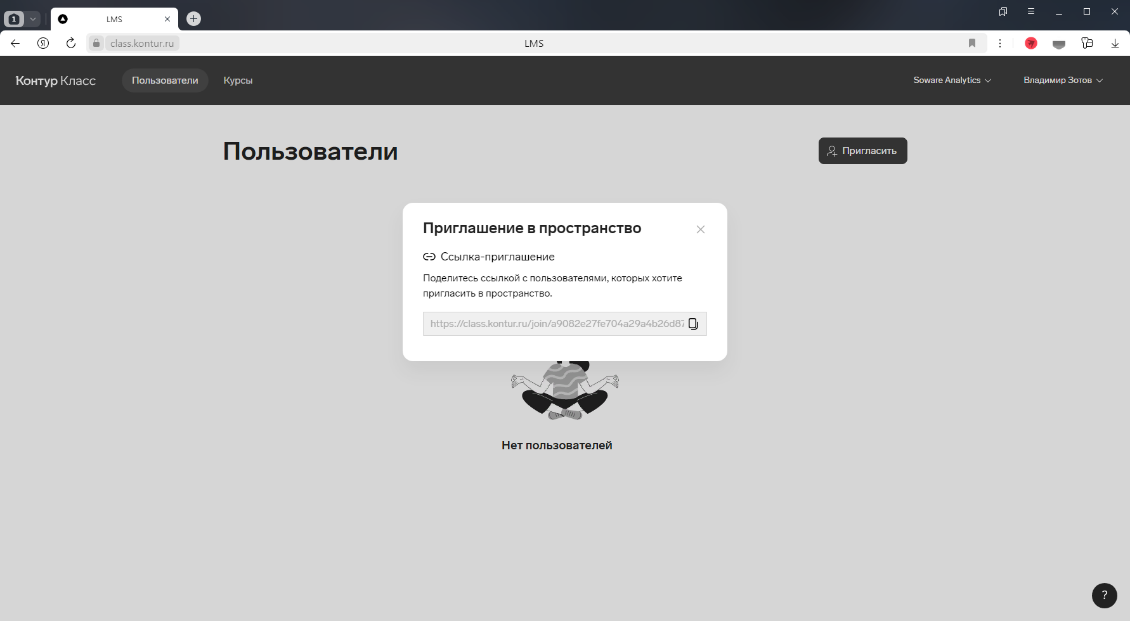


Рисунок 1 – Приглашение студентов по ссылке на платформе “Контур Класс”

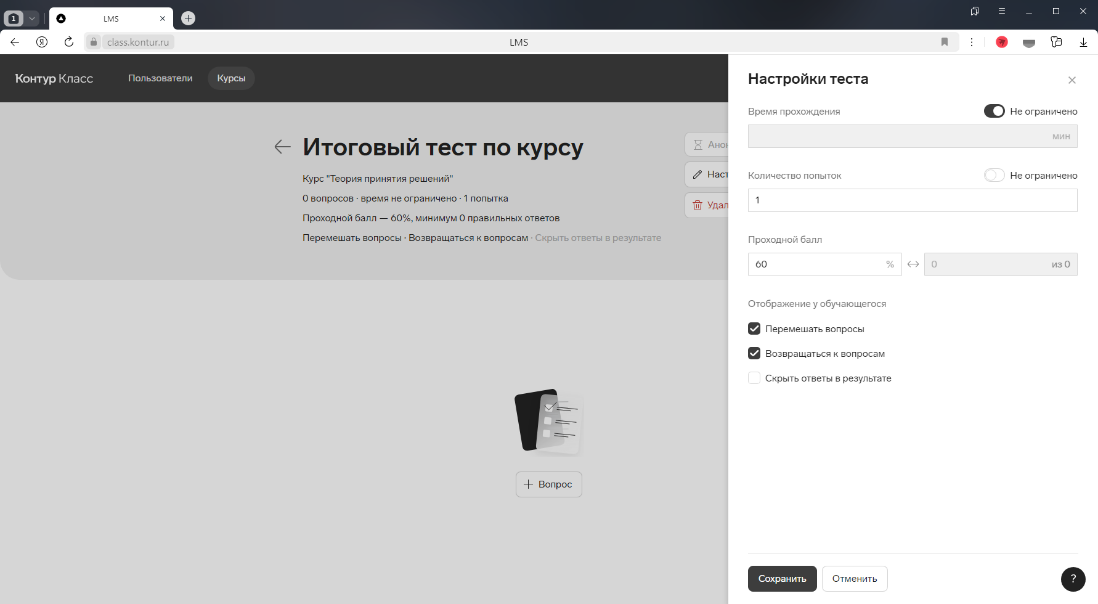


Рисунок 2 – Создание тестов на платформе “Контур Класс”

Изображение выглядит как текст, снимок экрана, Человеческое лицо, веб-страница

Автоматически созданное описание

Рисунок 3 – Отслеживание качества обучения на платформе “Контур Класс”

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 4 – Предоставление доступа к курсам на платформе “Контур Класс”

Следующим аналогичным решением и одним из самых популярных на рынке среди компаний является Teachbase, стоимость использования которого равна 565 500 рублей в год для управления 300 пользователей и доступом ко всему необходимому функционалу. Данная система намного больше предыдущей, ведь предусматривает использование не только учебными заведениями, но и бизнесом, поэтому имеет функционал по развитию карьеры, мотивации, интеграции дополнительных курсов и блоков кода по сторонним API. Teachbase предоставляет практически весь необходимый для Заказчика функционал по контролю активности, отчетности, работе с контрольными. Однако, как и предыдущий рассмотренный аналог, данная система не предоставляет возможности самостоятельной записи на задания и не отслеживает ограничение по количеству мест. Отсутствует генерация вариантов тестов, контрольных, поэтому преподавателю придется составлять их самостоятельно. Проверка открытых вопросов с коротким ответом не подразумевает проверку преподавателем, а проверяется автоматически по совпадению с введенными преподавателем вариантами ответа, также не является оптимальным решением, поскольку на верность ответа в данном случае будут влиять порядок и форма слов [8].

Прочие системы управления обучением не подходят Заказчику по тем же причинам, которые уже были рассмотрены, а именно: несоответствие требованиям по записи на варианты, составлению и проверке контрольных, предоставлению доступа к контрольным и лабораторным. Данные системы больше нацелены на обучение, в то время как первоначальная цель Заказчика – управление учебной деятельностью, поэтому пользователям разрабатываемой системы – студентам - не нужен функционал по изучению материалов, лекций, в то время как выбор вариантов лабораторных и самостоятельная запись на дополнительные задания по желанию являются приоритетными.

Для более детального анализа аналогов и понимания необходимости разработки персональной и автоматизированной системы управления учебной деятельностью необходимо проанализировать методы, которые используются Заказчиком на данный момент, и трудности, с которыми он сталкивается в процессе. В основном заказчик работает в электронных таблицах Google Sheets, формах Google и Github. С помощью таблиц Заказчик контролирует успеваемость студентов и дедлайны, с помощью форм - создает контрольные, а с помощью Github - выдает задания и проверяет решения. Однако эти системы не автоматизированы и потому не ускоряют работу, а лишь переводят ее в электронный формат, предоставляя постоянный онлайн доступ к результатам данной работы. Также негативно сказывается отсутствие консолидации всех трех систем, так как необходимая информация находится на разных платформах, не синхронизируется между собой, отчего часто приходится выполнять одну и ту же работу, самостоятельно актуализировать информацию и перепроверять полученную.

Таким образом, при обзоре аналогов не было выявлено прямых аналогичных решений, но было найдено много косвенных. Однако данные системы не удовлетворяют всем потребностям и требованиям Заказчика, из чего можно сделать вывод о необходимости разработки персонализированной системы управления учебной деятельностью.

# **Проектирование**

## **2.1 Построение диаграммы использования**

Для более глубокого понимания функционала системы и его распределения между разными ролями пользователей необходимо построить диаграмму прецедентов (диаграмму использования). Данная диаграмма в UML отражает отношения между акторами и прецедентами и является составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне. В основном это диаграмма, описывающая доступность функционала разрабатываемой программной системы в зависимости от принадлежности пользователя к группе.

Разрабатываемая система разделена на две группы пользователей: студент и администратор (преподаватель). Работа преподавателя в основном заключается в контроле и создании необходимых для обучения материалов и тестов, в то время как основная цель пользователя – следовать предоставленному процессу обучения, выполнять задания. Подробнее с функциональными возможностями каждой группы авторизованных и неавторизованных пользователей можно ознакомиться на диаграмме использования (Рисунок 5–7, страницы 10 и 11). На рисунке 5 можно заметить, что у администратора нет функционала для регистрации, поскольку, в целях предотвращения атак и обеспечения безопасности данных, созданием аккаунтов администраторов занимается разработчик системы.

Изображение выглядит как текст, снимок экрана, круг, Шрифт

Автоматически созданное описание

Рисунок 5 – Диаграмма использования неавторизованных групп пользователей

Изображение выглядит как текст, снимок экрана, круг, Шрифт

Автоматически созданное описание

Рисунок 6 – Диаграмма использования авторизованного студента

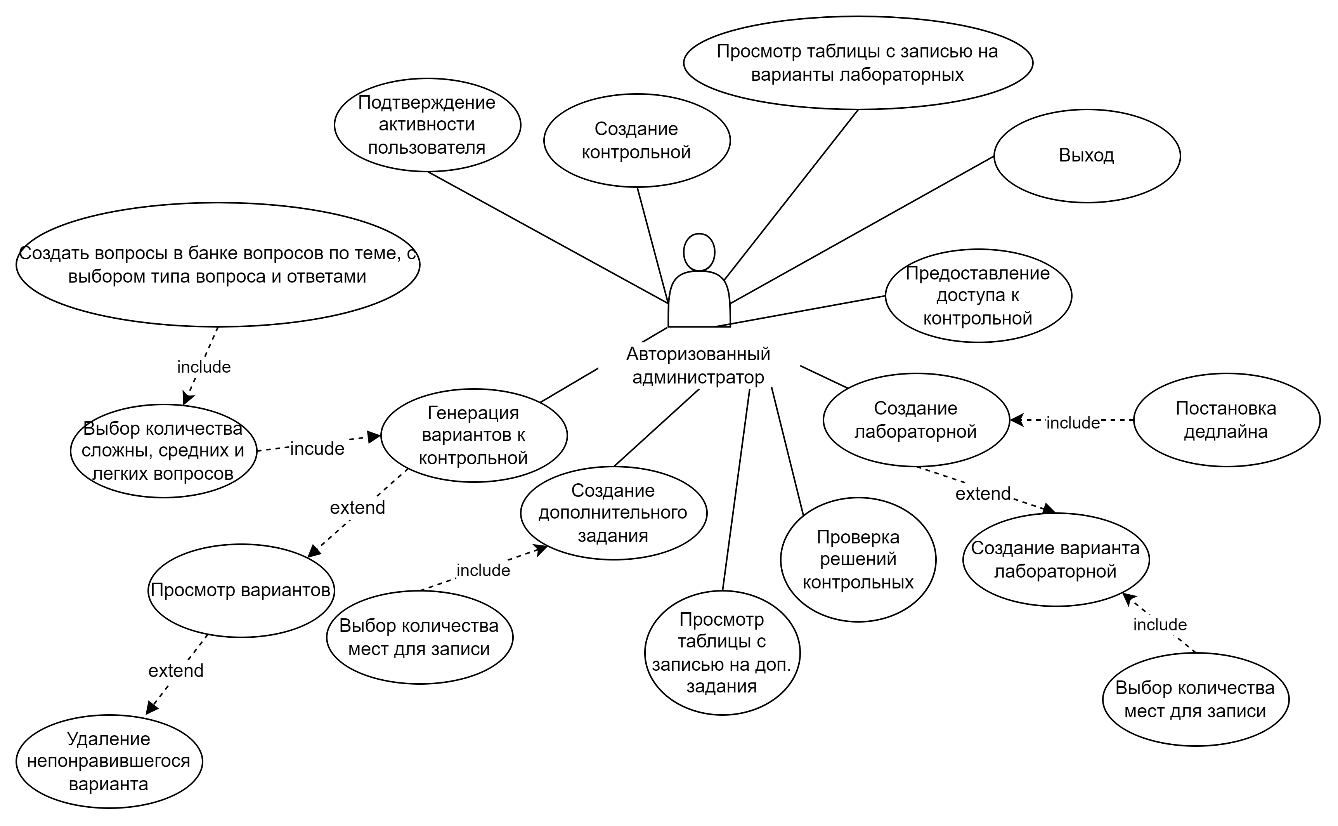


Рисунок 7 – Диаграмма использования авторизованного администратора

## **Построение диаграммы бизнес-процессов с использованием методологии IDEF0**

IDEF0 - метод функционального моделирования, а также графическая нотация, которая используется для описания и формализации бизнес-процессов [2]. Данный метод построения диаграмм бизнес-процессов позволяет не только понять функциональные возможности системы и их разделение по группам пользователей, но также обратить внимание на данные, которые необходимы для реализации той или иной функции. На данной диаграмме показан весь путь пользователя от регистрации и авторизации до просмотра отчетных таблиц администратором и записи на задание студентом. Результат построения данной диаграммы представлен на рисунке 8.

Изображение выглядит как текст, снимок экрана, диаграмма

Автоматически созданное описание

Рисунок 8 – Диаграмма бизнес-процессов с использованием методологии IDEF0

## **Построение диаграммы деятельности**

Для того, чтобы определить взаимосвязи разных групп пользователей и как функционал одной группы влияет на функциональные возможности другой, необходимо построить диаграмму деятельности. Диаграммы деятельности (Activity diagram) – это один из видов диаграмм, применяемых в UML для моделирования динамических аспектов поведения системы. Это блок-схема, которая показывает, как поток управления переходит от одной деятельности к другой [5]. Для построения данной диаграммы были выбраны основные бизнес-процессы системы: работа групп пользователей с контрольными, лабораторными и дополнительными заданиями, поскольку данные сущности и работа с ними являются основой всех бизнес-процессов системы. Результат построения диаграмм деятельности представлен на рисунках 9–11 страниц 13 и 14.

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рисунок 9 – Диаграмма деятельности работы с контрольными

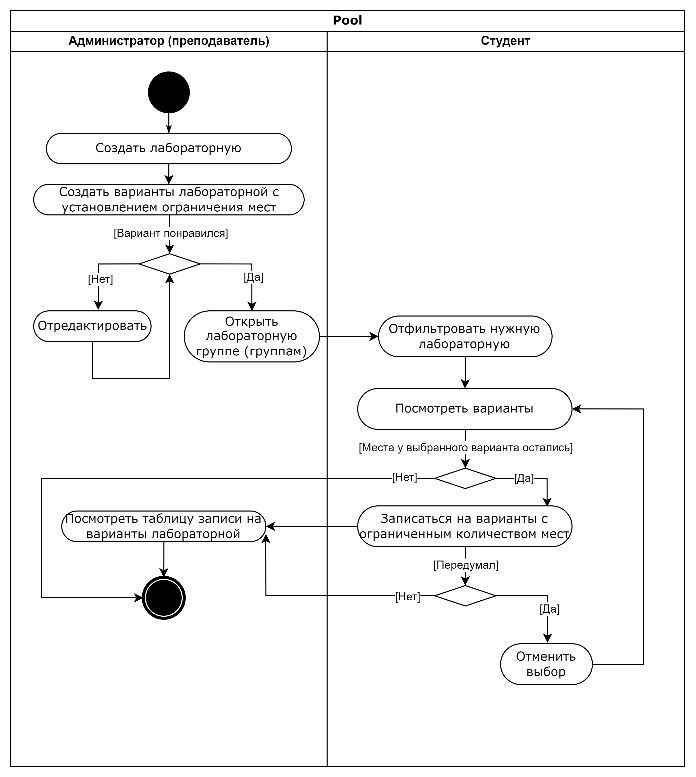


Рисунок 10 – Диаграмма деятельности работы с лабораторными

Изображение выглядит как текст, снимок экрана, программное обеспечение, дизайн

Автоматически созданное описание

Рисунок 11 – Диаграмма деятельности работы с дополнительными заданиями

## **Построение диаграммы последовательности**

Так как система состоит из клиентской и серверной частей, необходимо понять, как будут разделены функциональные требования между ними и как правильно организовать работу системы. Для этого лучше всего подойдет диаграмма последовательности. Диаграмма последовательности (sequence diagram) предназначена для представления взаимодействия между элементами модели программной системы в терминологии линий жизни и сообщений между ними [6].

Как уже упоминалось, основным функционалом данного веб-приложения является работа с контрольными, лабораторными и дополнительными заданиями. Была построена общая диаграмма по созданию контрольных, лабораторных и дополнительных заданий, так как эти функции отличаются между собой только вводимыми данными при создании (Рисунок 12, страницы 15). На рисунке можно увидеть, что клиентская часть не обращается на серверную с запросом на получение массива объектов, а контролирует данный процесс на бэкенде, отправляя данный запрос единожды при авторизации в веб-приложении. Это сделано для того, чтобы не перегружать работу сервера, а также сделать использование веб-приложения быстрее.

Изображение выглядит как текст, снимок экрана, диаграмма, дизайн

Автоматически созданное описание

Рисунок 12 – Диаграмма последовательности по созданию контрольных, лабораторных, дополнительных заданий

Одним из основных требований Заказчика по работе с контрольными является генерация вариантов контрольных с помощью выбора заданного количества уже составленных вопросов различной сложности случайным образом. Чтобы наглядно показать, как будет устроена работа с данным требованием, была построена диаграмма последовательности (Рисунок 13, страницы 16). На данной диаграмме видно, что преподаватель отправляет данные по желаемому количеству вопросов той или иной сложности. Все вопросы поделены на три уровня: легкие, средние, сложные. Затем эти предпочтения отправляются в качестве запроса на сервер, где происходит случайный выбор вопросов из списка имеющихся с учетом их сложности и количества. В последствие составленный вариант отправляется клиенту и добавляется в массив. Так же, как и на предыдущей диаграмме, можно заметить, что клиент не отправляет дополнительный запрос на сервер для получения списка всех вариантов.

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Автоматически созданное описание

Рисунок 13 – Диаграмма последовательности для генерации варианта контрольной

Работа с контрольными также включает в себя еще два важных требования: решение контрольной и ее проверка. Для наглядного представления данной функциональности были построены две диаграммы последовательности (Рисунок 14 и 15, страницы 17). Стоит упомянуть, что для понижения уровня списывания студентам достается случайный вариант контрольной, что отображено на диаграмме. Также на диаграммах можно заметить, что система обладает двумя уровнями проверок: автоматическая и проверка преподавателем. Первая занимается проверкой закрытых вопросов, с помощью совпадения выбранного варианта ответа с правильным, указанным преподавателем при создании. Вторая же, в свою очередь, заключается в проверке открытых вопросов. Как упоминалось при обзоре аналогов, это важный аспект проверки любых тестов, ведь прописать все возможные правильные ответы на открытые вопросы занимает много времени и сил, гораздо больше, чем единовременная проверка работы, поскольку сложно предусмотреть все возможные ответы студентов.

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Рисунок 14 – Диаграмма последовательности по решению контрольной

Изображение выглядит как текст, снимок экрана, диаграмма, дизайн

Автоматически созданное описание

Рисунок 15 – Диаграмма последовательности по проверке контрольных преподавателем

Несмотря на то, что работа с вариантами лабораторных также является важной частью общей функциональности, было принято решение не строить диаграмму последовательности по причине того, что создание варианта лабораторной по своей структуре очень похоже на создание лабораторной и отличается только вводимыми преподавателем данными. Однако важно показать диаграмму последовательности по авторизации пользователей в системе, чтобы продемонстрировать, каким образом проверяется роль пользователя и на что это влияет (Рисунок 16). Как видно на диаграмме, роль пользователя записана в объекте пользователя, данная роль хешируется вместе с токеном и отправляется на клиентскую часть, где также проверяется и, в зависимости от роли, перенаправляет пользователя на определенные роуты в системе. Теперь при любых запросах на сервер клиент отправляет не только свой захешированный токен, но и роль, которая также проверяется для доступа к обработке запроса.

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Автоматически созданное описание

Рисунок 16 – Диаграмма последовательности для авторизации

## **Проектирование диаграммы базы данных**

Перед началом этапа разработки системы необходимо спроектировать базу данных. Это поможет понять, из каких частей, моделей должна состоять система, какие данные необходимы и как они будут использоваться. Как говорилось раннее, была выбрана документо-ориентированная система управления базами данных MongoDB. Данная система управления считается нереляционной, то есть в ней не используется табличная схема строк и столбцов. В таких базах данных применяется модель хранения, оптимизированная под конкретные требования типа хранимых данных [4]. Данный выбор был сделан по нескольким причинам:

1. поддержка JSON-документов, что является большим плюсом при использовании языка программирования JavaScript,
2. высокая скорость работы СУБД за счет возможности индексации,
3. простота администрирования.

Как и у любой СУБД, в MongoDB есть свои минусы. Одним из основных является не полное соответствие ACID. Несмотря на это, количество преимуществ в контексте рассматриваемого проекта превалирует [9].

Для проектирования базы данных было решено создать следующие коллекции:

* Users для хранения информации о пользователях,
* Controls для хранения информации о контрольных работах,
* ControlMarks для хранения информации о решениях контрольных работ,
* Questions для хранения вопросов для генерации контрольных работ,
* Tasks для хранения информации о лабораторных и дополнительных заданиях.

Более подробно с наполнением каждой коллекции можно ознакомиться на рисунке 17 страницы 20.

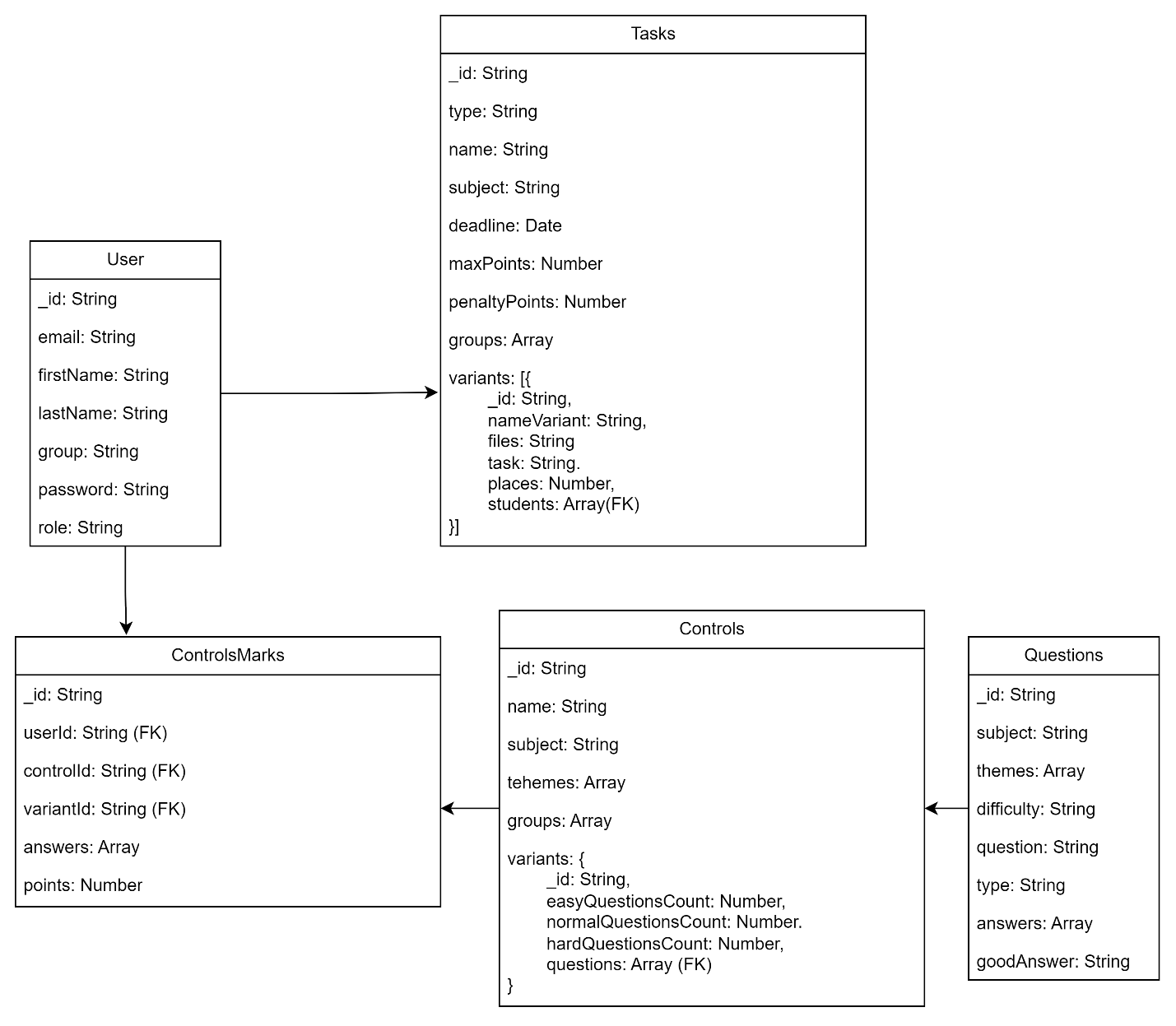


Рисунок 17 – Диаграмма базы данных

## **Выводы по второй главе**

В результате работы над второй главой были построены диаграммы, которые демонстрируют все имеющиеся бизнес-процессы и их соответствие функциональным требованиям. Данные диаграммы позволяют понять весь функционал разрабатываемого веб-приложения, реализацию данного функционала внутри системы, его разделение на группы пользователей, взаимосвязь. Помимо этого, были построены диаграммы, определяющие составные части системы и необходимые для разработки данные.

# **Разработка серверной части**

## **Регистрация и авторизация**

В первую очередь необходимо реализовать функционал регистрации и авторизации, поскольку функциональность веб-приложения зависит от роли пользователя в системе, которую контролируют данные функции. Функционал регистрации на клиентской части доступен только для группы студентов в целях безопасности системы. Как видно по диаграмме базы данных, у пользователя должна быть роль, в зависимости от которой будет разделен функционал. При регистрации студенту автоматически проставляется роль “user”, которую невозможно изменить благодаря наличию валидации при получении запроса от клиента, написанной с использованием библиотеки celebrate (Рисунок 18). Регистрация администратора, в свою очередь, осуществляется схожим образом, но роль пользователя проставляется со значением “admin”.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 18 – Валидация регистрации

При регистрации пароль пользователя хешируется. Это сделано для того, чтобы даже у администратора системы не было доступа к данной информации. Хеширование осуществляется с помощью библиотеки bcrypt. В данном случае число 10 обозначает стоимость для алгоритма. Стоимость определяет, сколько раз будет выполнено данное хеширование (Рисунок 19, страницы 22).

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Рисунок 19 – Функция регистрации с хешированием пароля

Для реализации функции авторизации необходимо реализовать поиск пользователя в системе по почте и захешированному паролю. Для этого был написан статический метод модели (Рисунок 20).

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 20 – Статический метод поиска пользователя по почте

Функция авторизации состоит также из отправки JSON Web Token. Данный токен необходим на клиенте для проверки авторизации пользователя, а также для получения id пользователя и его роли. С помощью секретного ключа генерируется подпись JWT на основе содержимого токена, что обеспечивает целостность данных (Рисунок 21).

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 21 – Функция авторизации пользователя

Для безопасности системы были написаны две проверки: наличие авторизации и роли администратора соответственно. Первая осуществляется перед всеми функциями, кроме регистрации и авторизации (Рисунок 22). В то время как вторая - только перед функционалом администратора (Рисунок 23, страницы 24).

Изображение выглядит как текст, электроника, компьютер, снимок экрана

Автоматически созданное описание

Рисунок 22 – Проверка наличия авторизации

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Автоматически созданное описание

Рисунок 23 – Проверка роли пользователя

## **Разработка функциональных возможностей преподавателя**

В первую очередь необходимо разобрать наиболее важные бизнес-процессы, связанные с контрольными работами и данными, влияющими на них, поскольку этих процессов большинство. Функция создания контрольных работ необходима для всей последующей работы с контрольными. Реализация данного функционала представлена на рисунке 24.

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Автоматически созданное описание

Рисунок 24 – Функция создания контрольной работ

Важным функциональным требованием является генерация варианта контрольной работы с использованием вопросов из модели “Questions”. При генерации варианта контрольной важно получить от пользователя данные о количестве сложных, средних и легких вопросов. Функция находит сначала вопросы, соответствующие заданной тематике контрольной работы, а затем случайным образом отбирает определенное количество вопросов разного уровня сложности и собирает их в один вариант. Реализация данной функции представлена на рисунках 25 и 26 страницы 25.

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 25 – Функция выбора вопросов для генерации варианта

Изображение выглядит как текст, снимок экрана, программное обеспечение, Операционная система

Автоматически созданное описание

Рисунок 26 – Функция создания варианта контрольной работы

Следующим по важности функциональным требованием является возможность предоставления группам доступа к контрольным работам и его удаление. Для реализации данного требования были написаны функции: openControl и deleteAccess. OpenControl принимает от клиентской части массив групп и добавляет их в поле “groups” модели контрольных, в то время как функция deleteAccess принимает одну группу и удаляет ее из соответствующего поля модели. Реализация функции openControl представлена на рисунке 27 страницы 26.

Изображение выглядит как текст, снимок экрана, дисплей, компьютер

Автоматически созданное описание

Рисунок 27 – Функция предоставления группам доступа к контрольной работе

Следующим требованием является проверка решенных контрольных. Данная функция получает id записи о решении пользователем контрольной и количество баллов, а затем перезаписывает данное значение (Рисунок 28).

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 28 – Функция проверки решения контрольной

Функциональные требования по работе с лабораторными и дополнительными заданиями схожи с уже представленными функциями для работы с контрольными работами. Они также дают возможность создавать, удалять и предоставлять доступ. Единственные отличия в отсутствии возможности генерации вариантов. Для создания варианта лабораторной работы используется модель “Tasks”. Функция принимает данные о количестве мест для записи на вариант, названии, самом задании, файле, если его добавили (Рисунок 29).

Изображение выглядит как текст, снимок экрана, дисплей, компьютер

Автоматически созданное описание

Рисунок 29 – Функция создания варианта лабораторной работы

Для обработки полученного файла, его проверки на валидность и загрузки на сервер используются библиотеки multer и moment. Первая отвечает за загрузку файла в хранилище, а вторая – за именование файла в системе. Реализация данной функции представлена на рисунке 30 страницы 28.

Изображение выглядит как текст, снимок экрана, дисплей, компьютер

Автоматически созданное описание

Рисунок 30 – Функция загрузки файлов на сервер

## **Разработка функциональных возможностей студента**

Основными процессами студента при работе с разрабатываемым веб-приложением являются: получение контрольных и лабораторных работ, дополнительных заданий, решение контрольной работы, а также запись на варианты лабораторных работ и дополнительные задания.

Функции по получению работ и дополнительных заданий имеют схожую структуру. Для этого функция принимает номер группы студента и отправляет массив доступных для этой группы работ и заданий. В качестве примера реализации выбрана функция getUserControls, предоставляющая контрольные работы (Рисунок 31, страницы 29).

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Автоматически созданное описание

Рисунок 31 – Функция получения доступных контрольных работ

Важно отметить, что при решении контрольной работы студент получает случайный вариант из массива всех вариантов. Это требование разработано для того, чтобы повысить уровень уникальности работ. Данная функция принимает информацию о пользователе, варианте контрольной работы, ответах пользователя и предварительных баллах (Рисунок 32). Предварительные баллы получаются посредством автоматической проверки соответствия выбранных пользователем ответов на закрытые вопросы правильным.

Изображение выглядит как текст, снимок экрана, мультимедиа, дисплей

Автоматически созданное описание

Рисунок 32 – Функция решения контрольной работы

В отличии от контрольных работ, варианты лабораторных работ и дополнительные задания пользователь выбирает сам и записывается на них. Функция makeAppointmnetTask берет id студента и id выбранного задания, затем записывает id пользователя в массив студентов, проверяя при этом, что места для записи еще есть, и что пользователь еще не записан на это задание (Рисунок 33).

Изображение выглядит как текст, снимок экрана, компьютер, дисплей

Автоматически созданное описание

Рисунок 33 – Функция записи на задание

## **Вывод по третей главе**

В результате работы над главой были созданы все необходимые функции. Веб-приложение предоставляет возможность регистрации, авторизации, создания, удаления, открытия, генерации контрольных и лабораторных работ, дополнительных заданий со стороны преподавателя, а также возможность записи, решения и получения этих работ и заданий со стороны студента.

**ЗАКЛЮЧЕНИЕ**

В рамках данной практической работы был проведен анализ существующих систем управления учебной деятельностью. В результате анализа не было найдено прямых аналогов, подходящих под требования и желания Заказчика, но были выявлены косвенные, у которых есть ряд преимуществ, однако они не могут обеспечить Заказчику необходимые возможности системы по управлению учебной деятельностью. На основании выполненного анализа были сделаны выводы и модифицированы некоторые требования Заказчика.

На этапе проектирования были построены диаграммы, которые демонстрируют существующие бизнес-процессы и функциональные требования, распределенные по группам пользователей, взаимодействие требований и пользователей друг с другом, а также диаграммы, показывающие внутреннее строение системы, взаимодействие частей системы между собой в тех или иных функциях и строение базы данных разрабатываемого веб-приложения.

Разработка серверной части системы осуществлялась на express, фреймворке Node.js, с использованием NoSQL СУБД MongoDB. Была реализована требуемая Заказчиком функциональность: регистрация, авторизация, работа с дополнительными заданиями, контрольными и лабораторными работами с учетом разделения данных функций относительно групп пользователей.

Таким образом была разработана серверная часть системы управления учебной деятельностью по заказу преподавателя практики факультета инфокоммуникационных технологий, Добрякова Давида Ильич. Система соответствует всем требованиям Заказчика по надежности, безопасности, расширяемости и общей функциональности.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. MongoDB, Express.js, React.js, Node.js: лучший технологический стек? [Электронный ресурс] // it.world – URL: https://www.it-world.ru/tech/technology/193941.html (дата обращения 08.02.2024).
2. IDEF0: что такое и как используется [Электронный ресурс] // Группа компаний – URL: https://software.by/info/blogs/1c/idef0-what-is-and-how-to-use/ (дата обращения 14.02.2024).
3. Аутсорсинг или заказная разработка на стеке MERN [Электронный ресурс] // Work Solutions – URL: https://worksolutions.ru/useful/autsorsing-ili-zakaznaya-razrabotka-na-steke-MERN/ (дата обращения 08.02.2024).
4. В чем особенности MongoDB и когда эта база данных вам подходит: руководство для новичков [Электронный ресурс] // Журнал VK Cloud “Завтра облачно” – URL: https://cloud.vk.com/blog/osobennosti-mongodb-kogda-baza-dannyh-vam-podhodit (дата обращения 20.02.2024).
5. Диаграмма деятельности uml [Электронный ресурс] // StudFiles – URL: https://studfile.net/preview/9431262/page:11/ (дата обращения 05.03.2024).
6. Диаграмма последовательности (sequence diagram) [Электронный ресурс] // ФСИС\_семинар-7\_Диаграмма-последовательности – URL: https://clck.ru/39qyyP (дата обращения 25.02.2024).
7. Контур Класс [Электронный ресурс] // Контур Класс – URL: https://kontur.ru/class (дата обращения 10.02.2024).
8. Менеджерам обучения [Электронный ресурс] // Teachbase – URL: https://help.teachbase.ru/ (дата обращения 10.02.2024).
9. Подумываете об использовании MongoDB? [Электронный ресурс] // Habr – URL: https://habr.com/ru/companies/otus/articles/565700/ (дата обращения 20.02.2024).