



**МИНОБРНАУКИ РОССИИ**

**федеральное государственное автономное образовательное учреждение  
высшего образования**

**«Московский государственный технологический университет  
«СТАНКИН»**

**(ФГАОУ ВО «МГТУ «СТАНКИН»)**

**Институт цифровых  
интеллектуальных систем**

**Кафедра  
компьютерных систем  
управления**

**Мироненко Дарья Андреевна**

Выпускная квалификационная работа по направлению подготовки  
15.03.04 «Автоматизация технологических процессов и производств»,  
профиль «Автоматизация технологических процессов и производств (в  
машиностроении)»

на тему:

**«Веб-приложение для помощи сотрудникам кафедр в систематизации  
информации о выпускных квалификационных работах обучающихся»**

Регистрационный номер № \_\_\_\_\_

Зав. кафедрой,  
д.т.н., профессор

\_\_\_\_\_  
(подпись)

Мартинов Георги  
Мartiнович

Руководитель,  
старший  
преподаватель

\_\_\_\_\_  
(подпись)

Путинцева Елена  
Валентиновна

Обучающийся:  
студент гр. АДБ-21-08

\_\_\_\_\_  
(подпись)

Мироненко Дарья Андреевна

**Москва 2025**



**МИНОБРНАУКИ РОССИИ**

**федеральное государственное автономное образовательное учреждение  
высшего образования**

**«Московский государственный технологический университет  
«СТАНКИН»**

**(ФГАОУ ВО «МГТУ «СТАНКИН»)**

**Институт цифровых  
интеллектуальных систем**

**Кафедра  
компьютерных систем  
управления**

**«УТВЕРЖДАЮ»**

Заведующий кафедрой

\_\_\_\_\_ **Мартинов Г.**

**М.**

«\_\_\_» \_\_\_\_\_ 2025 г.

### **ЗАДАНИЕ**

На выпускную квалификационную работу  
по направлению подготовки 15.03.04 «Автоматизация технологических  
процессов и производств»,  
профиль: «Автоматизация технологических процессов и производств (в  
машиностроении)»

**Мироненко Дарья Андреевна  
группа АДБ-21-08**

Тема: «Веб-приложение для помощи сотрудникам кафедр в систематизации  
информации о выпускных квалификационных работах обучающихся»

Тема утверждена приказом от «\_\_\_» \_\_\_\_\_ 202\_ г. № \_\_\_\_\_

Срок сдачи законченной ВКР на кафедру «\_\_\_» \_\_\_\_\_ 202\_ г.

Целью данной выпускной квалификационной работы является повышение эффективности взаимодействия между студентами и научными руководителями, а также упрощение процесса контроля за выполнением выпускных проектов. В разработанном в ходе выполнения работы приложении также будут предусмотрены функции для добавления, редактирования, поиска и фильтрации данных о ВКР, а также для создания отчетов и уведомлений.

## Содержание

Введение .....	5
ГЛАВА 1. АНАЛИЗ СУЩЕСТВУЮЩИХ ПОДХОДОВ К ХРАНЕНИЮ И УПРАВЛЕНИЮ ИНФОРМАЦИЕЙ О ВЫПУСКНЫХ КВАЛИФИКАЦИОННЫХ РАБОТАХ .....	8
1.1. Понятие веб-приложения и веб-сервиса .....	8
1.1.1. Разница между веб-приложением и веб-сервисом .....	10
1.1.2. Преимущества и недостатки веб-приложений и веб-сервисов .....	11
1.2. Анализ существующих сервисов .....	13
ГЛАВА 2. РАЗРАБОТКА АРХИТЕКТУРЫ ВЕБ-ПРИЛОЖЕНИЯ, ВКЛЮЧАЮЩЕЙ КЛИЕНТСКУЮ И СЕРВЕРНУЮ ЧАСТИ .....	19
2.1. Выбор подхода и инструментов .....	20
2.2. Описание архитектуры. Диаграмма классов .....	21
2.3. Stack технологий, используемых для реализации веб-приложения .....	24
ГЛАВА 3. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ РАЗРАБОТАННОЙ СТРУКТУРЫ ВЕБ-ПРИЛОЖЕНИЯ .....	28
3.1. Практическая реализация веб-приложения в среде Figma .....	29
3.2. Практическая реализация веб-приложения в IntelliJ idea .....	34
ГЛАВА 4. ТЕСТИРОВАНИЕ И ОТЛАДКА СИСТЕМЫ, РЕКОМЕНДАЦИИ ПО ЕЕ ВНЕДРЕНИЮ .....	42
4.1. Тестирование и отладка системы .....	42
4.2. Рекомендации по внедрению .....	44
Список используемой литературы .....	46

## **Введение**

В современных условиях стремительного развития информационных технологий и цифровизации образовательных процессов, всё более актуальной становится необходимость создания эффективных инструментов для управления учебной и научной деятельностью. Одним из таких инструментов является веб-приложение для хранения и актуализации данных о выпускных квалификационных работах (ВКР). Этот инструмент должен быть эффективным и надежным, так как выпускные работы студентов представляют собой важнейший этап учебного процесса, требующий от образовательных учреждений не только качественного сопровождения, но и организованного учёта информации о проектах, их темах, руководителях и сроках выполнения. [1]

Заглянув в недавнюю историю, мы увидим, что развитие систем хранения и актуализации выпускных квалификационных работ (ВКР) происходило параллельно с эволюцией информационных технологий и образовательных систем. Как шел процесс мы можем увидеть, разобрав ключевые этапы этого развития:

### **1. Период бумажных архивов (до 1990-х годов 20 века).**

Это, конечно, самый длительный этап, что обусловлено уровнем развития технологий. На ранних стадиях работы студентов хранились в виде бумажных копий в университетских библиотеках или кафедрах. Очевидно, что это было связано с отсутствием цифровых технологий и ограничениями в доступе к вычислительной технике. Хранить, обрабатывать, использовать информацию в таких условиях было сложно во многих отношениях, поиск необходимых сведений занимал длительное время и требовал специальных навыков работы с каталогами.

### **2. Появление первых компьютерных баз данных (1990-е годы 20 века).**

С развитием персональных компьютеров и внедрением баз данных, некоторые университеты начали экспериментировать с электронным учетом и хранением работ. Первыми такими системами стали локальные базы данных и текстовые документы, хранящиеся на компьютерах кафедр. Новые системы, конечно же, ускоряли и упрощали работу, поэтому разработка учетных систем продолжалась, системы совершенствовались, становились все более удобными и доступными.

### 3. Внедрение корпоративных информационных систем (2000-е годы 20 века)

С началом нового века развиваются технологии совершенствуется техника, что, конечно, поспособствовало дальнейшему развитию методов работы с учетными системами. С ростом вычислительных мощностей и доступностью серверов начали появляться корпоративные информационные системы (КИС) для образовательных учреждений, включающие модули для хранения и управления ВКР.

### 4. Интеграция интернет-технологий и появление облачных сервисов (2010-е годы 20 века).

Этот период характеризуется активным развитием веб-приложений и облачных решений, что открыло новые возможности для управления и хранения данных о ВКР. Вошедшие в повседневную жизнь облака позволили сделать системы более доступными и масштабируемыми, пользоваться ими стало еще удобнее, а загружать и получать данные быстрее.

### 5. И, наконец, современные системы с элементами искусственного интеллекта (2020-е годы 20 века).

Это уже время, когда во многих сферах жизни начали активно применяться системы с элементами искусственного интеллекта и машинного обучения для обработки и анализа данных о ВКР. Такие системы помогают не только хранить и актуализировать данные, но и анализировать их на основе различных метрик.

Конечно, новые технологии делают рабочие процессы более быстрыми и удобными для пользователей. Для примера, традиционные методы хранения данных о ВКР (например, в виде таблиц, бумажных архивов или статических документов) обладают рядом недостатков, таких как отсутствие оперативного обновления информации, сложности в управлении и неудобство в использовании для всех участников процесса. А в то же самое время, веб-приложения обеспечивают возможность централизованного хранения, актуализации и быстрого доступа к данным в режиме реального времени. Таким образом, мы видим, как современные решения значительно упрощают работу как для студентов и преподавателей, так и для администрации учебных заведений.

Поэтому мной и выбрана цель данной дипломной работы — разработка веб-приложения, которое позволит автоматизировать процесс управления информацией о ВКР, повысить эффективность взаимодействия между студентами и научными руководителями, а также упростить процесс контроля за выполнением выпускных проектов.

В приложении будут предусмотрены функции для добавления, редактирования, поиска и фильтрации данных о ВКР, а также для создания отчетов и уведомлений. Помимо этого, также можно будет отследить процент выполнения работы и проверить написанный текст на антиплагиат.

В работе будут рассмотрены основные этапы разработки веб-приложения, начиная от анализа требований пользователей до реализации и тестирования системы.

# **ГЛАВА 1. АНАЛИЗ СУЩЕСТВУЮЩИХ ПОДХОДОВ К ХРАНЕНИЮ И УПРАВЛЕНИЮ ИНФОРМАЦИЕЙ О ВЫПУСКНЫХ КВАЛИФИКАЦИОННЫХ РАБОТАХ**

Чтобы идти в ногу со временем, обеспечивать учащимся качественное образование, современные образовательные учреждения активно внедряют информационные системы для автоматизации различных процессов, связанных с управлением учебной и научной деятельностью. Важной частью этой деятельности является хранение и актуализация данных о выпускных квалификационных работах (ВКР). В настоящее время уже существует множество решений для управления такими данными, включая как специализированные программные продукты, так и общие системы управления проектами. Однако эти решения имеют как преимущества, так и недостатки, которые следует учитывать при разработке новой системы, чтобы сделать ее оригинальной и более эффективной, чем было ранее. [2]

## **1.1. Понятие веб-приложения и веб-сервиса**

Для начала следует определиться в понятиях, которыми мы будем пользоваться в дальнейшем, определить, что такое веб-приложение и веб-сервис. [3]

### **1. Понятие веб-приложения**

Веб-приложение — это программное обеспечение, которое работает на веб-сервере и доступно через веб-браузер. В отличие от традиционных настольных приложений, веб-приложения не требуют установки на устройство пользователя. Примером могут служить почтовые сервисы (например, Gmail), социальные сети (Facebook), онлайн-банкинг, системы управления контентом и т.д.

Характеристики веб-приложений:



- Доступ через интернет через браузер.
- Работают на различных устройствах и платформах (компьютеры, планшеты, смартфоны).
- Обновление происходит на стороне сервера, не требуя действий со стороны пользователя.
- Обычно состоят из клиентской (фронтенд) и серверной (бэкенд) частей.

## 2. Понятие веб-сервиса

Веб-сервис — это программный компонент, который позволяет различным приложениям обмениваться данными через сеть (обычно Интернет), используя стандартные протоколы (например, HTTP, SOAP, REST). Веб-сервисы чаще всего взаимодействуют с другими программами, а не с конечными пользователями. [4]

Примеры веб-сервисов:

- API для получения данных о погоде.
- Платежные системы, такие как PayPal или Stripe.
- Веб-сервисы для работы с социальными сетями, например, API Twitter или Facebook.

Характеристики веб-сервисов:

- Фокусируются на обмене данными между системами.
- Используют стандарты, такие как XML, JSON, SOAP, REST, для передачи данных.
- Могут быть вызваны любым клиентом, который поддерживает HTTP.

### 1.1.1. Разница между веб-приложением и веб-сервисом

Таблица 1.1 – Сравнение веб-приложения и веб-сервиса

Критерий	Веб-приложение	Веб-сервис
Цель	Обеспечение взаимодействия с пользователем	Обмен данными между системами
Пользователь	Человек, конечный пользователь	Другое приложение
Интерфейс	Графический интерфейс (GUI) через браузер	Нет интерфейса, вызов через HTTP
Примеры	Gmail, Facebook, Amazon	Google Maps API, PayPal API
Технологии	HTML, CSS, JavaScript, PHP, Python	REST, SOAP, XML, JSON

### **1.1.2. Преимущества и недостатки веб-приложений и веб-сервисов**

К основным преимуществам веб-приложений можно отнести [5]:

- **Доступность:** Они без проблем могут работать на любом устройстве с доступом к браузеру и интернету.
- **Обновления:** Обновления происходят непосредственно на сервере, не требуя действий от пользователя, что существенно упрощает работу.
- **Удобство:** Нет необходимости в установке на устройства пользователей.

Недостатки:

- **Зависимость от сети:** Не работают без доступа к интернету.
- **Ограниченная производительность:** По сравнению с настольными приложениями могут быть медленнее.
- **Безопасность:** Требуется защита от сетевых угроз (например, XSS, CSRF).

Преимущества веб-сервисов [6]:

- **Интероперабельность:** Могут использоваться любыми клиентами, поддерживающими стандартные протоколы.
- **Модульность:** Компоненты могут быть переиспользованы в разных системах по принципу кубиков конструктора.
- **Гибкость:** Могут работать с различными типами данных (JSON, XML).

Недостатки:

- **Сложность:** Разработка и настройка веб-сервисов требуют высокой квалификации, обычно требуется привлекать дорогостоящих специалистов, обойтись внутренними ресурсами зачастую невозможно.
- **Задержки:** Из-за необходимости обмена данными через сеть может возникать задержка.

- Зависимость от сети: Как и веб-приложения, веб-сервисы требуют доступ к интернету.

Таким образом, при выборе инструмента для работы, следует помнить, что веб-приложения всегда ориентированы на взаимодействие с конечным пользователем, предоставляя интерфейс через браузер, в то время как веб-сервисы обеспечивают взаимодействие между приложениями и платформами, предоставляя данные или функциональность через сеть [7].

## **1.2. Анализ существующих сервисов**

### **1. Традиционные системы учета данных о ВКР.**

Наиболее распространённым методом управления информацией о ВКР в учебных заведениях до сих пор остаются таблицы и электронные документы, такие как Microsoft Excel, Google Sheets и другие офисные приложения. В этих документах хранится информация о студентах, темах их работ, научных руководителях и сроках выполнения. Такой традиционный подход имеет ряд достоинств и недостатков.

Сначала преимущества:

- Простота использования. Эти инструменты широко распространены и не требуют особых навыков для работы. Большинство пользователей уже знакомы с ними, что снижает необходимость в дополнительном обучении, в связи с этим не приходится нанимать дополнительных сотрудников.
- Доступность. Электронные таблицы и документы доступны для всех и могут быть использованы в любом учебном заведении без значительных финансовых вложений в виде покупки, например, новой техники или программного обеспечения.
- Гибкость. Пользователи могут самостоятельно настраивать формат данных для удобства работы, добавлять новые поля и изменять структуру документов в зависимости от потребностей.

Недостатки:

- Отсутствие актуализации данных в реальном времени. В случае, если несколько пользователей работают с документом одновременно, могут возникнуть проблемы с синхронизацией данных, что приводит к устареванию информации.

- Ограниченные возможности для масштабирования. При увеличении объема данных, такие решения становятся менее эффективными: возрастает время загрузки и обработки информации, возникают сложности с поиском и фильтрацией данных.
  - Отсутствие автоматизации. В традиционных таблицах нет встроенной функциональности для автоматического отслеживания сроков выполнения работ, статусов ВКР и уведомления участников процесса.
  - Ограниченная безопасность. Электронные документы могут быть подвержены ошибкам или утере данных из-за неправильного хранения или некорректного доступа.
2. Корпоративные информационные системы управления учебным процессом. [8]

Некоторые учебные заведения используют корпоративные системы управления учебным процессом (Learning Management Systems, LMS) или системы управления университетами (например, Moodle, 1С: Университет). Эти системы позволяют хранить данные о студентах, курсах и других учебных процессах, включая выпускные работы.

Преимущества:

- Интеграция с другими учебными процессами. Корпоративные LMS часто включают не только управление ВКР, но и поддерживают другие аспекты учебного процесса, такие как расписание занятий, оценочные ведомости и отчёты.
- Централизованное хранение данных. Все данные о студентах и их проектах хранятся в единой системе, что упрощает управление и доступ к информации.

- Многопользовательский доступ. Пользователи (студенты, преподаватели, администрация) могут одновременно работать с системой без риска потери данных и не опасаясь замедления работы.

Недостатки:

- Высокая сложность настройки и использования. Корпоративные системы обычно требуют значительных ресурсов для внедрения и настройки, привлечения профильных специалистов, а также обучения персонала.
- Избыточный функционал. LMS часто включают множество модулей, которые могут быть не нужны для конкретной задачи управления ВКР. Это приводит к усложнению интерфейса и снижению удобства использования, причем не факт, что большой функционал вообще нужен для работы.
- Высокие затраты на обслуживание. Такие системы требуют регулярного технического обслуживания, что может потребовать дополнительных затрат на IT-инфраструктуру и поддержку, а такие вложения по карману не каждому учебному заведению.

### 3. Специализированные веб-приложения для управления ВКР.

Существуют также специализированные веб-приложения, разработанные исключительно для управления выпускными квалификационными работами. Эти системы предоставляют набор инструментов для отслеживания тем ВКР, назначения руководителей и контроля за выполнением проектов.

Преимущества:

- Специализация на ВКР. Такие системы разработаны с учётом специфики работы с выпускными квалификационными проектами, что делает их удобными для решения конкретных задач.

- Автоматизация процессов. Веб-приложения позволяют автоматически отслеживать изменения статусов ВКР, уведомлять участников процесса о сроках и этапах работы.
- Доступ в режиме реального времени. Все изменения и обновления данных происходят моментально, что упрощает актуализацию информации и взаимодействие между студентами и преподавателями.

Недостатки:

- Ограниченная интеграция. Такие приложения, как правило, не включают другие аспекты учебного процесса и могут не интегрироваться с уже существующими системами LMS или управления университетом.
- Необходимость разработки и поддержки. Для образовательных учреждений может потребоваться создание индивидуального решения, что требует ресурсов на разработку и последующую техническую поддержку системы.
- Может быть сложным для небольших проектов. Для учебных заведений с небольшим количеством ВКР специализированное приложение может оказаться избыточным.

Теперь, на основании изложенного, можно провести краткий сравнительный анализ имеющихся систем.



Таблица 1.2 – Сравнение сервисов для хранения ВКР

Критерии	Таблицы и электронные документы	Корпоративные системы управления учебным процессом	Специальные веб-приложения	Итоговое решение
Доступность	Высокая	Средняя	Высокая	Высокая
Простота использования	Средняя	Низкая	Средняя	Высокая
Централизованность хранения данных	Низкая	Высокая	Высокая	Высокая
Многопользовательский доступ	Средняя	Высокая	Средняя	Высокая
Специализация на ВКР	Низкая	Низкая	Высокая	Высокая

Режим реального времени	Средняя	Низкая	Высокая	Средняя
Безопасность	Низкая	Высокая	Низкая	Высокая

Все рассмотренные системы, используемые на сегодняшний день, имеют свои сильные и слабые стороны, ни одну из них нельзя назвать идеальной и подходящей для всех случаев. Традиционные таблицы и электронные документы просты и удобны, не требуют от пользователей существенных затрат и вложений, но не обеспечивают автоматизации и актуализации данных в реальном времени. Корпоративные системы управления учебным процессом предоставляют интеграцию с другими процессами, но могут быть сложными в настройке и эксплуатации. Специализированные веб-приложения обеспечивают максимальное удобство для управления ВКР, но их внедрение требует индивидуальной разработки и поддержки.

Поэтому, для решения задачи эффективного управления данными о выпускных квалификационных работах наиболее целесообразным представляется создание специализированного веб-приложения, которое будет сочетать в себе преимущества простоты использования, широкой доступности и автоматизации, а также обеспечивать актуализацию данных в режиме реального времени.

## **ГЛАВА 2. РАЗРАБОТКА АРХИТЕКТУРЫ ВЕБ-ПРИЛОЖЕНИЯ, ВКЛЮЧАЮЩЕЙ КЛИЕНТСКУЮ И СЕРВЕРНУЮ ЧАСТИ.**

Для нашего проекта — системы, помогающей упорядочивать информацию о ВКР студентов — архитектурная модель наиболее логична, поскольку выстраивает упрощенную иерархию, фокусируется на сущностях вроде студентов, проектов и руководителей. Это дает возможность сосредоточиться на логике взаимодействия и управления данными без излишней детализации.

## 2.1. Выбор подхода и инструментов

Наше приложение было разработано с использованием инструмента прототипирования Figma для начального проектирования интерфейсов, после чего интерфейсы были перенесены во Vue.js для клиентской части. Серверная сторона построена на Spring, а инфраструктура управляется через Docker. [9].

Данное решение было выбрано по нескольким причинам:

- Figma позволяет создать и протестировать прототип интерфейса, разработать макеты страниц и заранее представить, как будет выглядеть взаимодействие пользователя с системой.
- Vue.js был выбран в качестве фронтенд-фреймворка благодаря его гибкости, простоте интеграции и реактивности. Его компонентный подход ускорил разработку интерфейсов, созданных в Figma, а встроенная система состояний (Vuex/Pinia) упростила управление данными. Кроме того, Vue обладает понятной документацией и плавной кривой обучения, что позволило быстро адаптировать команду под проект.
- Spring Framework обеспечил надежную и масштабируемую серверную часть благодаря своей модульности и поддержке современных Java-технологий. Spring Boot ускорил развертывание, предоставив встроенные решения для безопасности, работы с БД (Spring Data JPA) и REST API. Его экосистема, включая Spring Security и Spring Cloud, позволила легко интегрировать дополнительные сервисы и обеспечить отказоустойчивость.

- Docker был использован для контейнеризации базы данных и сервисов, что обеспечило переносимость и стабильность окружения. Контейнеры упростили развертывание в разных средах (разработка, тестирование, продакшен), а Docker Compose позволил управлять зависимостями между сервисами. Это решение также облегчило масштабирование и уменьшило проблемы, связанные с различиями в рабочих окружениях у разработчиков.

В результате такой комбинации инструментов мы получаем удобную среду для быстрой разработки и внедрения функционала, что является особенно важным для нашего проекта, ориентированного на практическое применение.

## **2.2. Описание архитектуры. Диаграмма классов**

Архитектурная модель нашего решения будет представлена в виде диаграммы классов. Диаграмма классов — это одна из ключевых структурных диаграмм, которые описывают основные сущности системы (классы), их атрибуты и методы, а также связи между ними. В нашем случае диаграмма классов описывает сущности, которые играют ключевую роль в управлении информацией о дипломных работах студентов. [11], [12]

Веб-приложение можно разделить на несколько основных классов:

- Класс Студент — отвечает за хранение информации о каждом студенте, включая фамилию, имя, группу и т.д.
- Класс Проект — представляет выпускную квалификационную работу, включает данные о названии проекта, дате защиты, научном руководителе и статусе работы.
- Класс Руководитель — содержит информацию о сотрудниках кафедры, которые являются руководителями ВКР.

- Класс Кафедра — координирует все данные о кафедре, такие как название кафедры, список сотрудников и текущие проекты.

Для реализации диаграммы классов, соответствующей архитектуре нашего приложения, нам нужно показать основные классы и связи между ними.

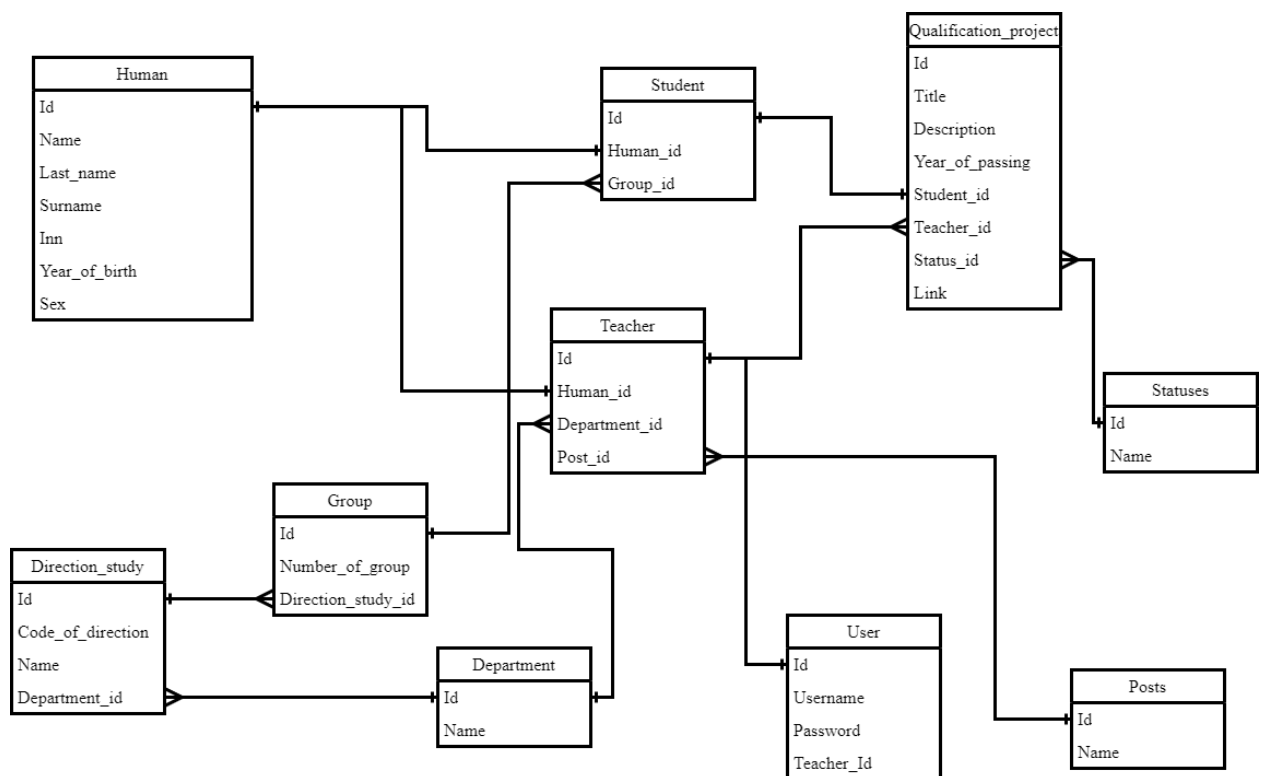
### **Пример диаграммы:**

- Human (Человек)
  - Атрибуты: id, Name, Last\_name, Surname, Inn, Year\_of\_birth, Sex
- Group (Группа)
  - Атрибуты: Id, Number\_of\_group, Direction\_study\_id
- Student (Студент)
  - Атрибуты: Id, Human\_id, Group\_id
- Department (Кафедра)
  - Атрибуты: Id, name
- Direction\_study (Направление обучения)
  - Атрибуты: Id, Code\_of\_direction, Name, Department\_id
- Teacher (Преподаватель)
  - Атрибуты: Id, Human\_id, Department\_id, Post\_id
- Posts (Должность)
  - Атрибуты: Id, Name
- Qualification\_project (Квалификационная работа)
  - Атрибуты: Id, Title, Description, Year\_of\_passing, Student\_id, Teacher\_id, Status\_id, Link
- Statuses (Статусы)
  - Атрибуты: Id, Name
- User (Пользователь)
  - Атрибуты: Id, Username, Password, Teacher\_id

## **Связи:**

- Класс Human – отвечает за хранение информации о людях, которые работают либо проходят обучение в вузе, содержит всю необходимую информацию о человеке: ФИО, дату рождения, пол, инн;
- Класс Group – содержит такую информацию о группе студента как номер группы и ссылку на учебный план группы;
- Класс Student – отвечает за хранение информации о студенте и ссылается на таблицы Human и Group;
- Класс Direction\_study – содержит у нас информацию о названии специальности, а также ссылается на кафедру, за которой закреплена данная специальность;
- Класс Department – содержит информацию о кафедре;
- Класс Teacher – содержит информацию о сотрудниках кафедры и должности, которую они занимают в вузе;
- Класс Posts – является перечислением должностей, которые могут занимать сотрудники;
- Класс Qualification\_project – содержит информацию о квалификационной работе студента, название работы, её описание год написания работы, ид студена написавшего работу и ид куратора работы, а также ссылку на файл с работой;
- Класс Statuses – является пересечением статусов написания дипломной работы;
- Класс User – содержит данные о пользователях.

Рисунок 2.1 – Диаграмма классов



### 2.3. Stack технологий, используемых для реализации веб-приложения

Разделим архитектуру веб-приложения на 3 части:

1. Backend
2. Frontend
3. База данных

Backend – это «внутренняя» часть приложения (то, что скрыто от глаз пользователя и находится на сервере), его мы будем реализовывать с помощью Java + Spring Boot.

Почему выбрали:

- Java — стабильный, мощный язык, особенно для корпоративных решений.



- Spring Boot — ускоряет разработку REST API, снижает объём конфигурации, имеет встроенную поддержку тестирования, безопасности и мониторинга.
- Подходит для веб-приложений с возможностью масштабирования и интеграции.

## 2. Модули Spring Boot:

- spring-boot-starter-web — построение REST API для фронта.
- spring-boot-starter-security + spring-security-oauth2-authorization-server — защита маршрутов, авторизация пользователей, в т.ч. OAuth2.
- spring-boot-starter-data-jpa — удобная работа с базой данных, построенная поверх Hibernate.
- spring-boot-starter-validation — валидация входящих данных (загрузок, логинов и др.).
- spring-boot-starter-actuator + micrometer-registry-prometheus — мониторинг и метрики для администраторов.
- spring-boot-starter-test + spring-security-test — модульное и интеграционное тестирование.
- springdoc-openapi-starter-webmvc-ui — автоматическая генерация Swagger UI (документация API).

Frontend — это «внешняя» часть приложения (то, что видно пользователю, интерфейс с которым взаимодействует пользователь), его мы будем реализовывать с помощью Vue.js + Tailwind CSS + Pinia.

Почему выбрали:

### 1. Vue.js (JavaScript-фреймворк)

- Vue.js — лёгкий, производительный и быстрый фреймворк, идеально подходящий для административных панелей и простых интерфейсов.

- Низкий порог вхождения, гибкость архитектуры — легко обучаем и поддерживаем.
- Одностраничное приложение (SPA) позволяет быстро переключаться между разделами (просмотр, фильтрация, загрузка), без перезагрузки страницы — важно для UX.

## 2. Tailwind CSS (утилитарный CSS-фреймворк)

Почему выбрали:

- Ускоряет верстку: стили пишутся прямо в HTML/Vue-компонентах.
- Отлично подходит для быстрого прототипирования интерфейсов.
- Гарантирует консистентный внешний вид без излишней кастомизации.
- Упрощает поддержку.

## 3. Pinia (хранилище состояния)

Почему выбрали:

- Современная альтернатива Vuex: легче, быстрее и понятнее.
- Удобно управлять состоянием (например, данными о пользователе, фильтрами, загруженными работами).
- Поддерживает модульность, типизацию и DevTools-интеграцию.

База данных — это организованная совокупность данных, которые хранятся в соответствии со схемой данных. Она отражает состояние объектов и их отношений. Ее мы будем реализовывать с помощью PostgreSQL.

Почему выбрали:

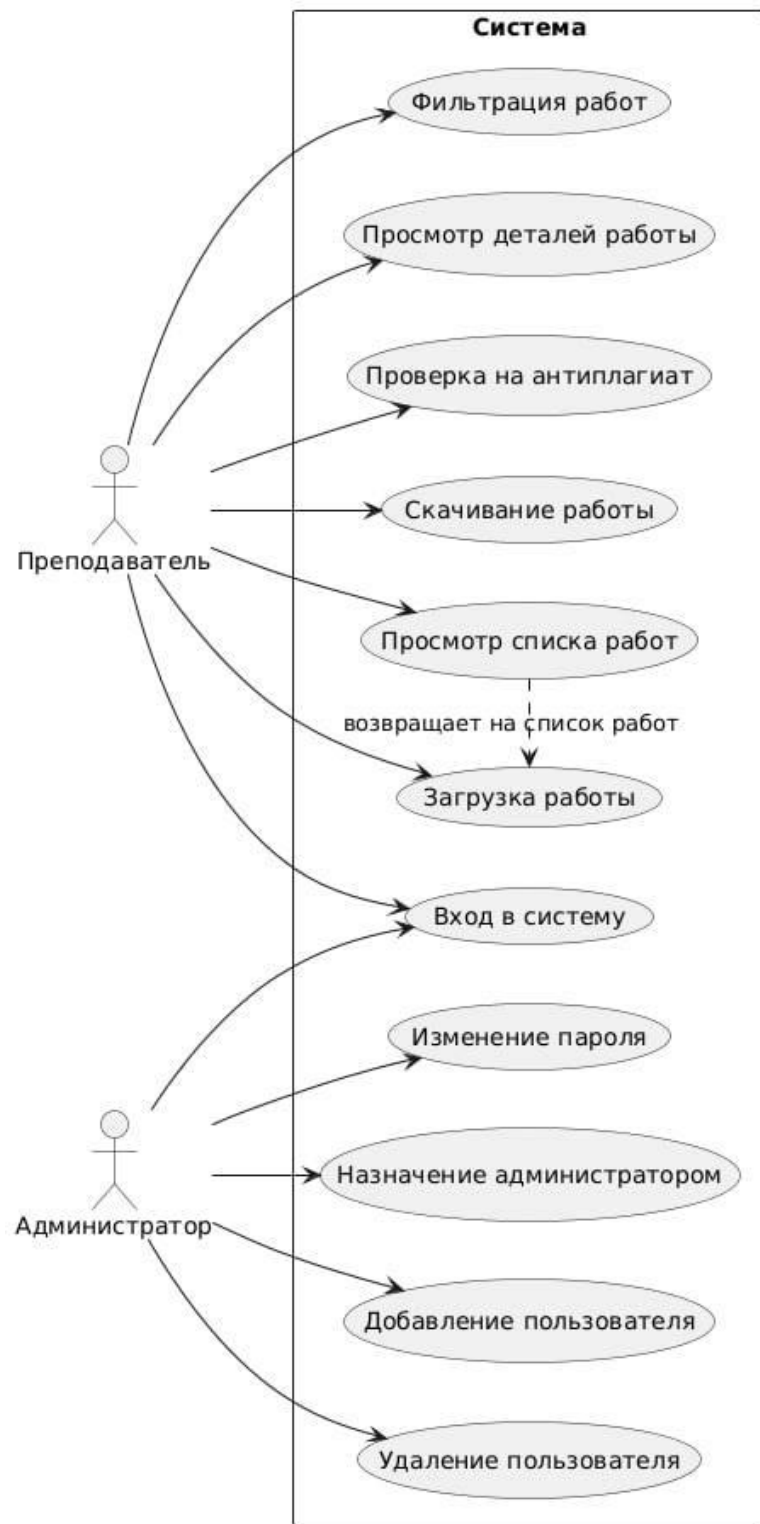
- Мощная реляционная СУБД с поддержкой ACID — надёжность при хранении дипломных работ.
- Поддерживает сложные запросы, индексы, полнотекстовый поиск (может быть полезен при проверке оригинальности).

- Имеет расширения (например, для хранения JSON, геоданных, поиска и пр.).
- Бесплатная, открытая, активно развивается, хорошо документирована.
- Легко интегрируется с Spring Data JPA.

### **ГЛАВА 3. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ РАЗРАБОТАННОЙ СТРУКТУРЫ ВЕБ-ПРИЛОЖЕНИЯ**

На основании составленной диаграммы классов и примерной визуализации решения, представленной ниже, можно составить алгоритм работы приложения в виде диаграммы прецедентов.

Рисунок 3.1 – Диаграмма прецедентов

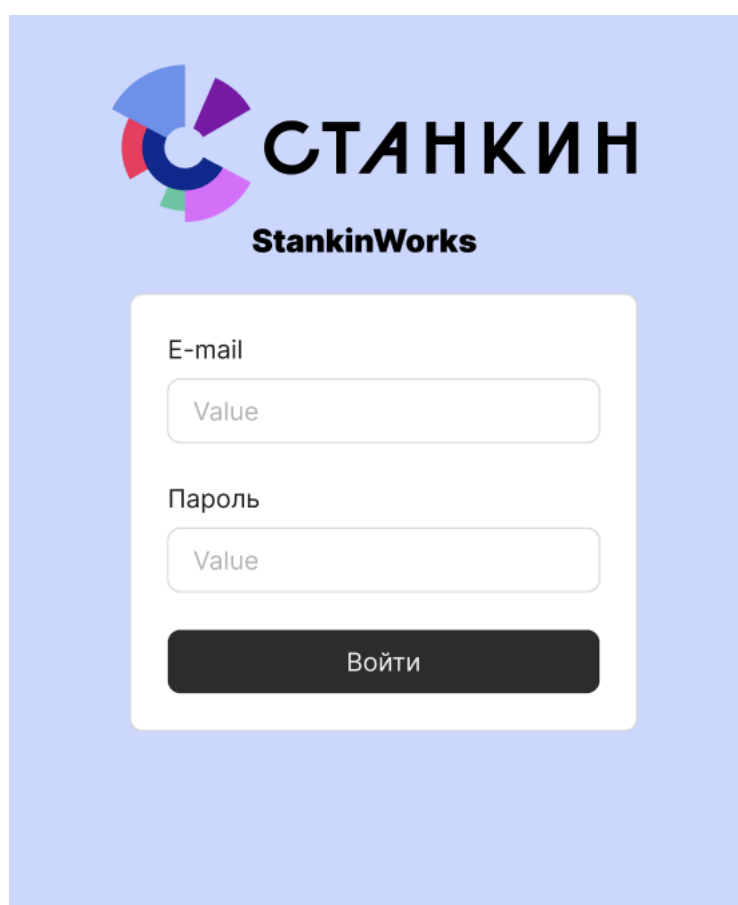


### 3.1. Практическая реализация веб-приложения в среде Figma

Основная цель приложения — автоматизация работы с выпускными квалификационными работами студентов. Приложение разделено на четыре основных модуля: **Авторизация, Управление работами, Просмотр работ, Администрирование.**

В приложении есть возможность входа для преподавателей с использованием e-mail и пароля.

Рисунок 3.2 – Экран входа



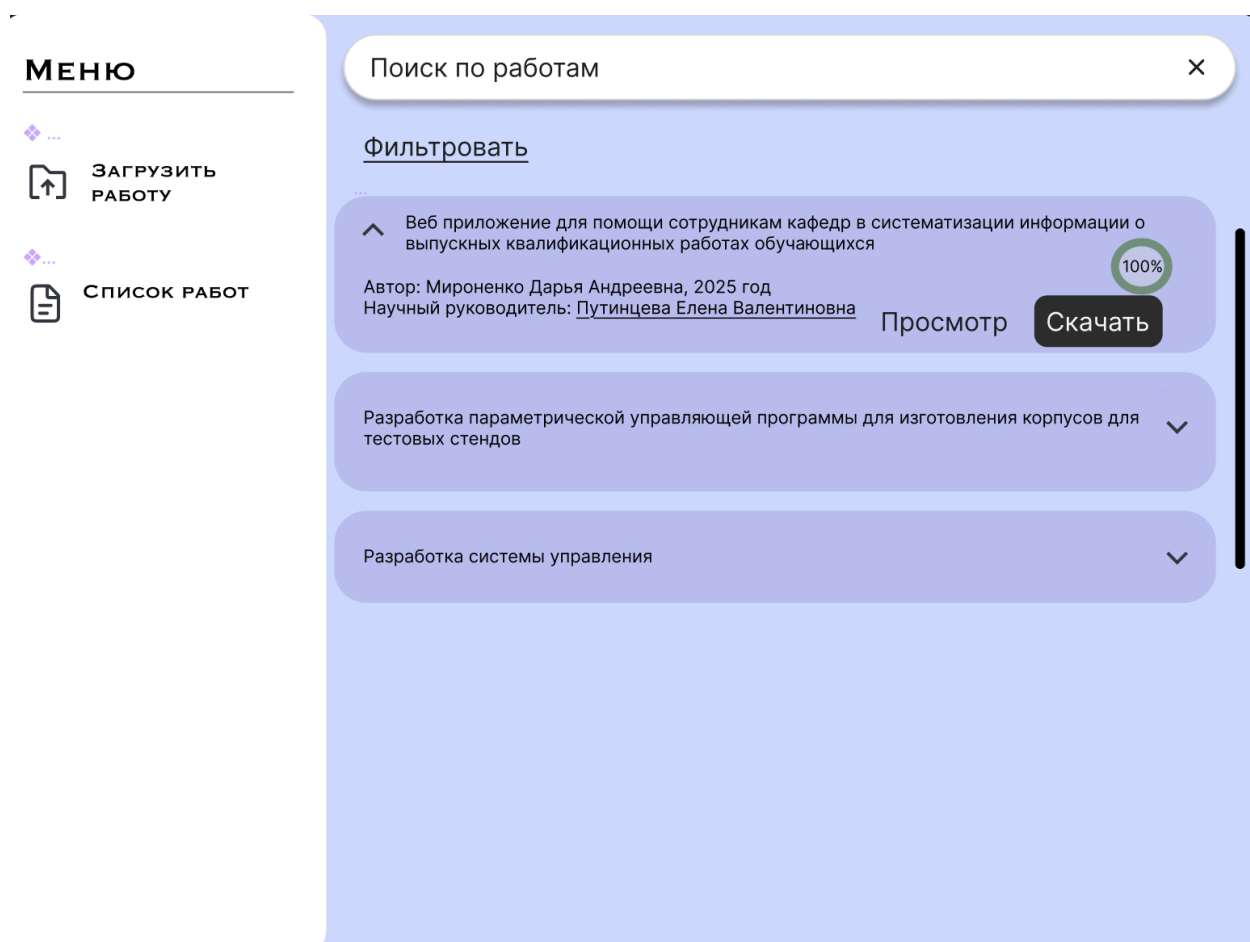
После успешного входа, открывается экран действий, можно: добавить чью-либо работу или просмотреть список уже имеющихся работ, его можно фильтровать и находить нужную работу. По умолчанию открывается экран со всеми работами, а слева есть меню действий. Если вход оказался неуспешным, то на экране появится ошибка и можно обратиться к администратору для получения нового пароля.

Также в общем стоке работ, можно просмотреть информацию об авторе работы, годе создания и научном руководителе. Для этого нужно нажать на стрелочку справа, рядом с названием работы, чтобы развернулось описание. На этом же экране можно посмотреть работу, либо сразу же скачать ее к себе на компьютер.

Помимо этой информации, можно также увидеть процент готовности работы. Процент рассчитывается исходя из наличия пунктов, содержащих слово «глава».

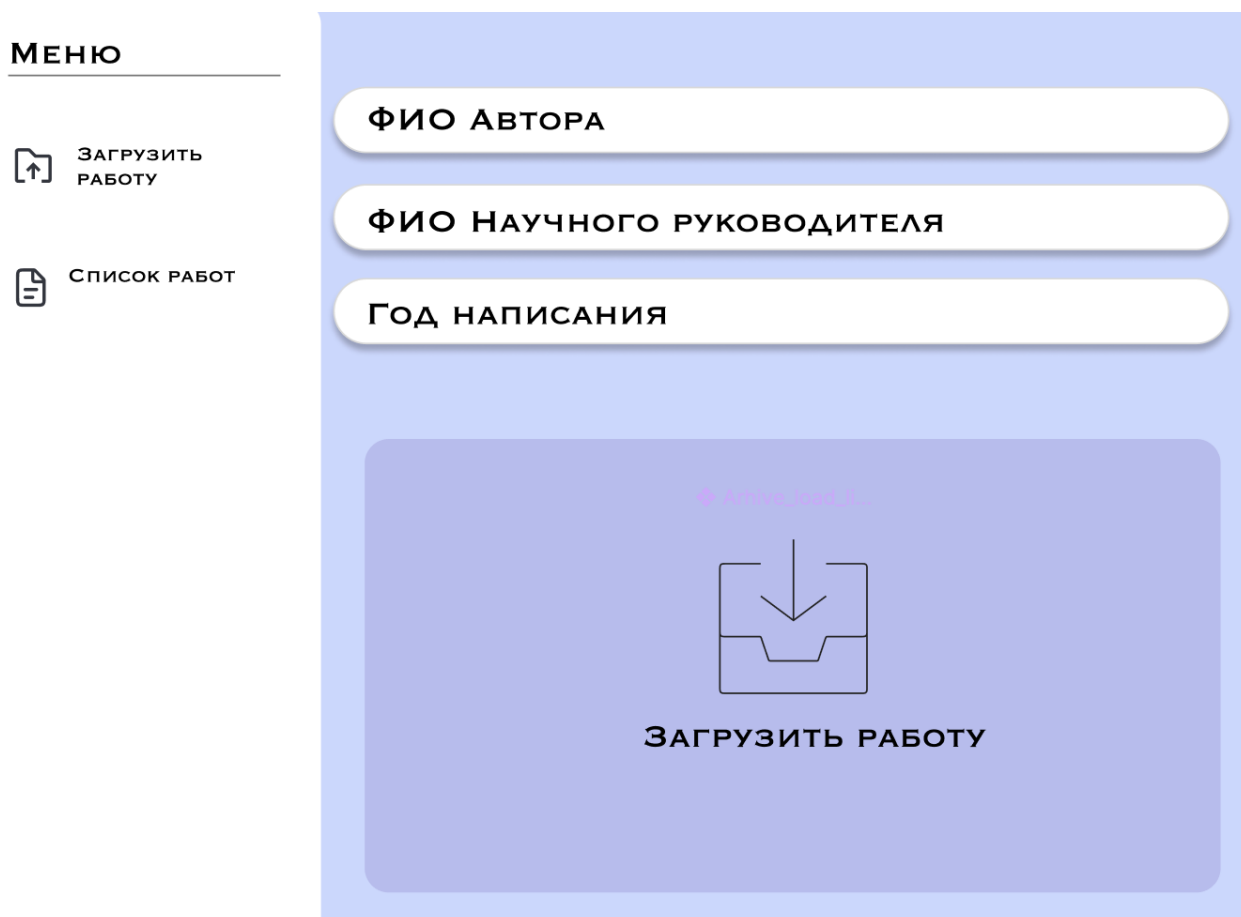
При подсчете мы учитываем, что после слова «глава», обязательно должны быть еще символы, если их нет, то мы не засчитаем эту главу.

Рисунок 3.3 – Экран действий



При нажатии на добавление файла, открывается возможность добавить файл, а также добавить информацию об авторе, о научном руководителе и о годе написания дипломной работы. Файл автоматически будет добавлен в общий сток работ, затем его через просмотр списка работ.

Рисунок 3.4 – Экран действий



При выборе файла, открывается окно, в котором можно ознакомиться с работой, также ее можно скачать себе. С этого экрана также можно вернуться назад к просмотру работ. Помимо этого, в этом окне можно проверить работу на антиплагиат. При нажатии кнопки «проверка на антиплагиат», отправляется запрос в сервис антиплагиата, там работа проверяется и затем у нас, на экране веб-приложения, выводится результат проверки антиплагиатом и дата последней проверки.



Рисунок 3.5 – Экран после выбора работы

Веб приложение для помощи сотрудникам кафедр в систематизации информации о выпускных квалификационных работах обучающихся

---

Автор: Мироненко Дарья Андреевна  
Год: 2025  
Научный руководитель: Путинцева Елена Валентиновна


Последняя проверка на антиплагиат: дата

Уникальность работы: число%

Проверка на антиплагиат

Скачать

Назад



**МИНОБРАЗОВАНИЯ РОССИИ**  
федеральное государственное бюджетное образовательное учреждение высшего образования  
«Московский государственный технологический университет «СТАНКИН»  
(ФГБОУ ВО «МГТУ «СТАНКИН»)

**Институт цифровых интеллектуальных систем**

**Кафедра компьютерных систем управления**

**Мироненко Дарья Андреевна**

Выпускная квалификационная работа по направлению подготовки 15.03.04 «Автоматизация технологических процессов и производств», профиль «Автоматизация технологических процессов и производств (в машиностроении)» на тему:  
**«Веб-приложение для помощи сотрудникам кафедр в систематизации информации о выпускных квалификационных работах обучающихся»**

Регистрационный номер № \_\_\_\_\_

Зав. кафедрой,  
д.т.н., профессор \_\_\_\_\_ **Мартинев Георгий Мартинович**  
(подпись)

Руководитель,  
старший преподаватель \_\_\_\_\_ **Путинцева Елена Валентиновна**  
(подпись)

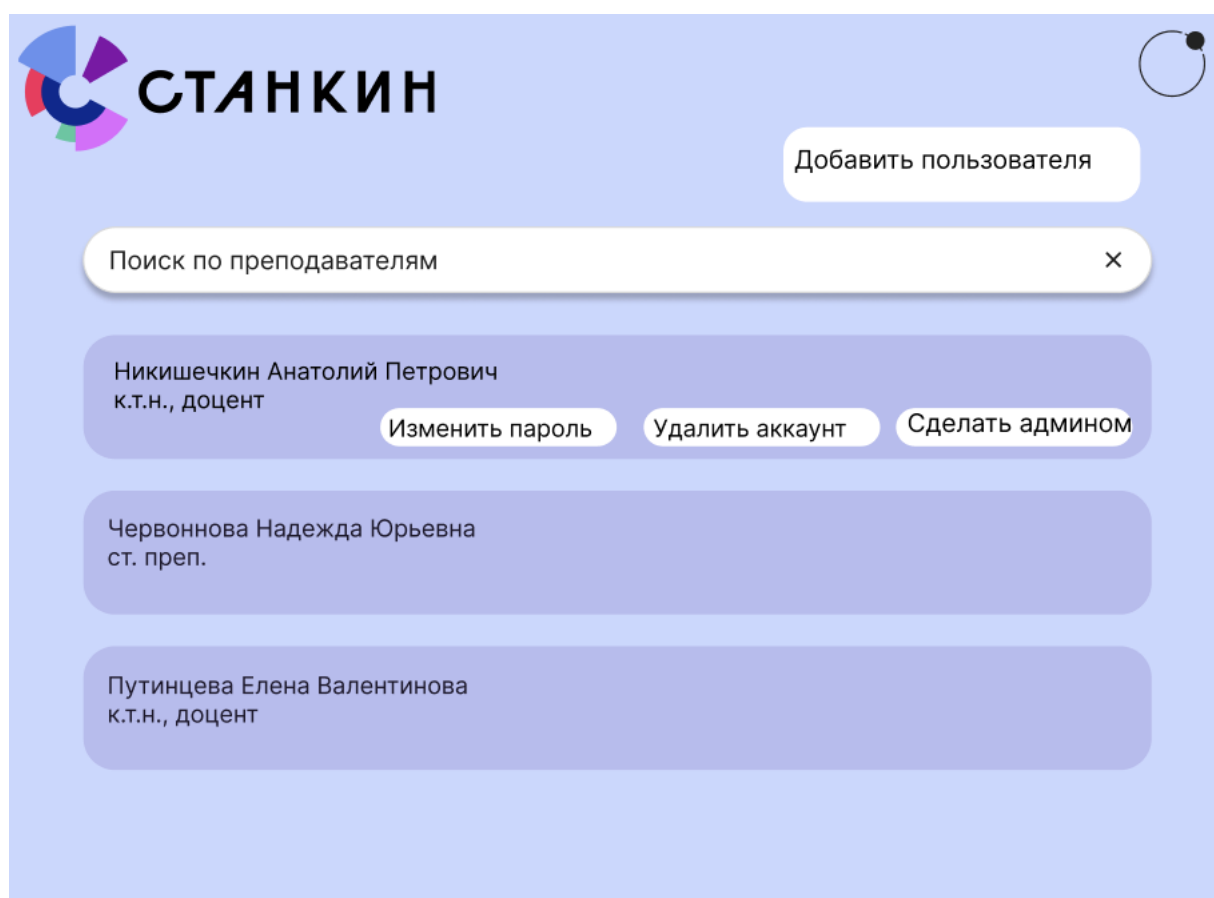
Обучающийся:  
студент гр. АДБ-21-08 \_\_\_\_\_ **Мироненко Дарья Андреевна**  
(подпись)

**Москва 2025**

Вперед

Также у нас есть вход для администратора. Он нужен для того, чтобы была возможность добавлять новых пользователей (преподавателей) для использования веб-приложения. Помимо добавления пользователя, можно их удалять (на случай если преподаватель ушел), менять пароли пользователям (если его забыли) и назначать администратором пользователя, на случай если основной администратор не сможет управлять веб-приложением.

Рисунок 3.6 – Экран панели администратора



### 3.2. Практическая реализация веб-приложения в IntelliJ idea

Рисунок 3.7 – Вид экрана входа после реализации и открытия через браузер

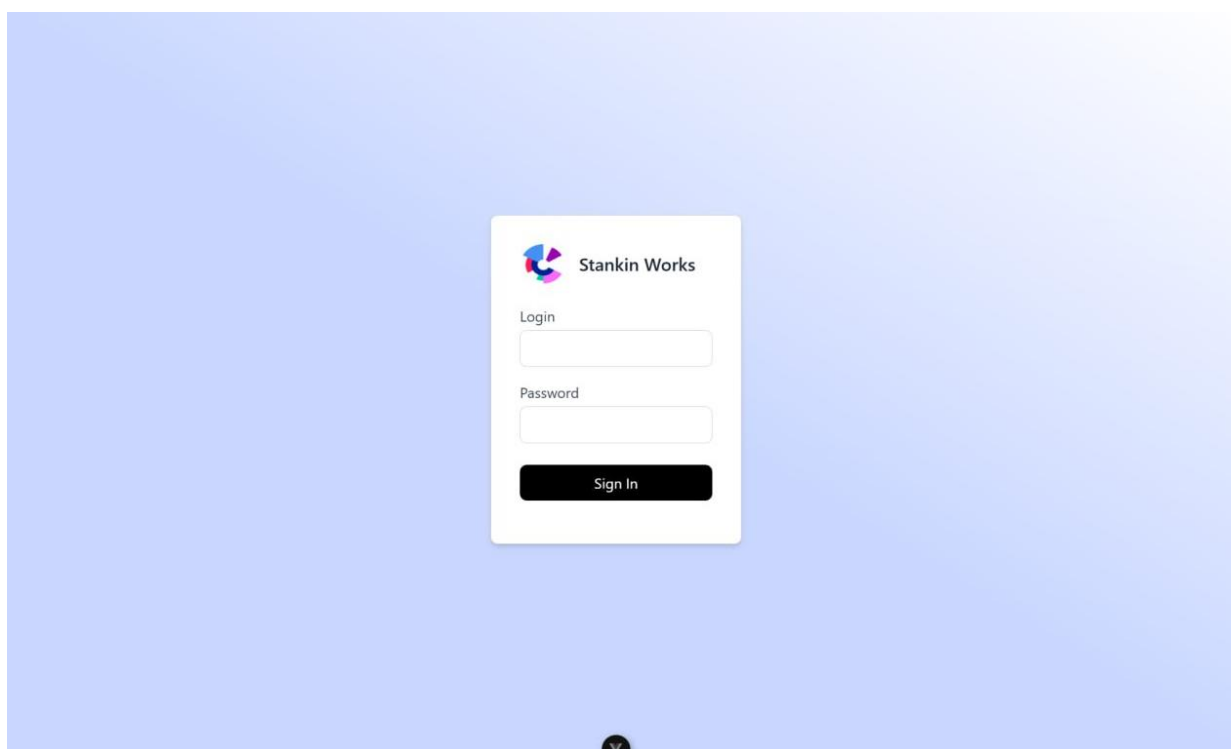


Рисунок 3.8 – Вид главного экрана после реализации и открытия через браузер

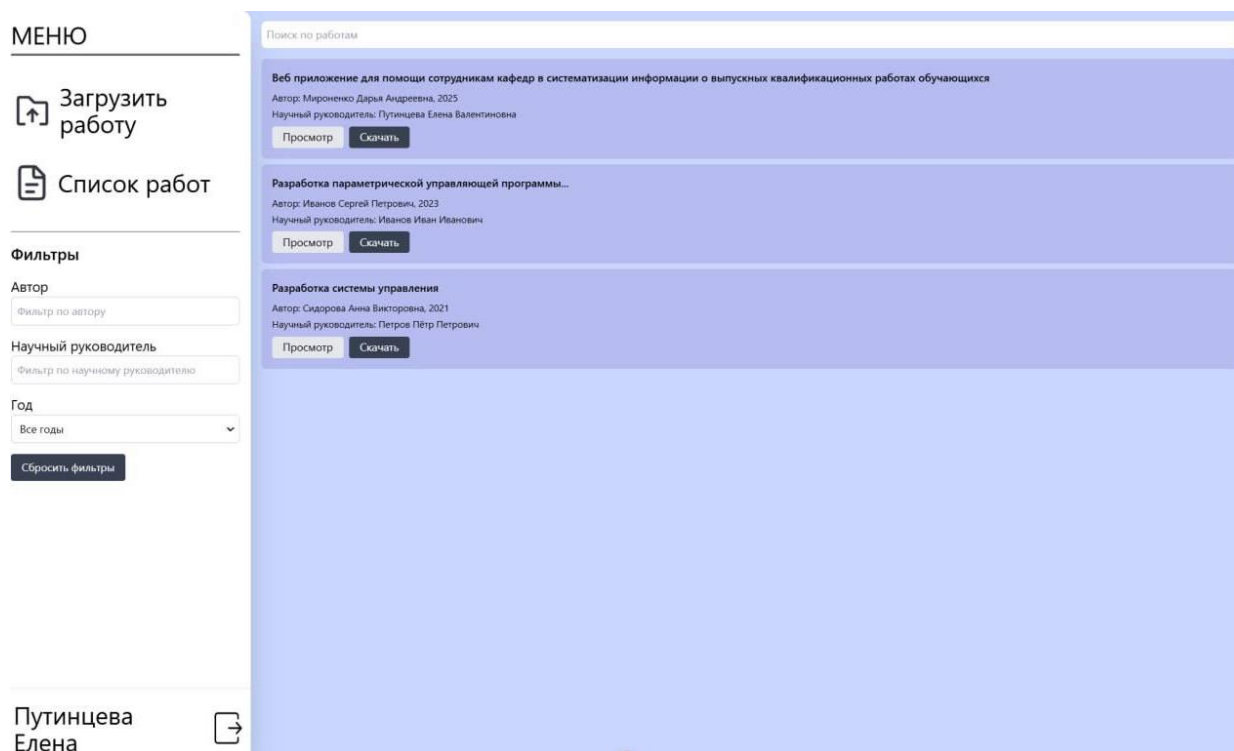


Рисунок 3.9 – Вид экрана загрузки работы после реализации и открытия  
через браузер

МЕНЮ

- Загрузить работу
- Список работ

Путинцева Елена

### ЗАГРУЗИТЬ РАБОТУ

ФИО АВТОРА  
Иванов Иван Иванович

ФИО НАУЧНОГО РУКОВОДИТЕЛЯ  
Иванов Иван Иванович

ГОД НАПИСАНИЯ  
2025

ФАЙЛ РАБОТЫ (PDF)

Перетащите файл сюда или кликните для выбора  
Поддерживаются только PDF-файлы

ЗАГРУЗИТЬ ДИПЛОМ

Рисунок 3.10 – Вид экрана просмотра работы после реализации и открытия  
через браузер

Веб приложение для помощи сотрудникам кафедр в систематизации информации...

**Автор:** Мироненко Дарья Андреевна  
**Год:** 2025  
**Научный руководитель:** Путинцева Елена Валентиновна

Силаев Захар ВычМат

1 / 11 100%

1

2

3

Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»  
Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №2  
«Численное решение нелинейных уравнений и систем»  
по дисциплине «Вычислительная математика»

Вариант: 11

Рисунок 3.11 – Вид экрана администратора после реализации и открытия  
через браузер



Рисунок 3.12 – Пример кода для входа в приложение

```
@Bean
@Order(2)
public SecurityFilterChain defaultSecurityFilterChain(HttpSecurity http)
    throws Exception {
    http.formLogin(Customizer.withDefaults())
        .csrf(AbstractHttpConfigurer::disable)
        .exceptionHandling(exceptionHandlingConfigurer->ex -> ex
            .authenticationEntryPoint(new HttpStatusEntryPoint(HttpStatus.UNAUTHORIZED))
        )
        .formLogin(formLoginConfigurer->form -> form
            .loginProcessingUrl("/login")
            .successHandler((HttpServletRequest req, HttpServletResponse res, Authentication auth) -> res
                .failureHandler((HttpServletRequest req, HttpServletResponse res, AuthenticationException ex) ->
                    .permitAll()
                )
            )
        .authorizeHttpRequests(authorizationManagerRequestMatchers -> auth -> auth
            .requestMatchers("/swagger-ui/**").permitAll()
            .anyRequest().authenticated()
        );
    return http.build();
}
```

Один из фильтров запросов, который проверяет был ли произведён вход в приложение, перед тем как разрешить доступ к сервисам приложения, требующим авторизации

Рисунок 3.13 – Пример кода для добавления пользователя

```
@PostMapping("/register")
@PreAuthorize("hasRole('ADMIN')")
@Operation(summary = "Create users by Admin")
@ApiResponses(value = {
    @ApiResponse(responseCode = "201", description = "Successfully created"),
    @ApiResponse(responseCode = "409", description = "User already exist"),
    @ApiResponse(responseCode = "403", description = "You need Admin role"),
    @ApiResponse(responseCode = "401", description = "Unauthorized")
})
public ResponseEntity<?> register(@Parameter(name = "username&password",
description = "dto with username and password")
    @RequestBody @Valid RegisterUserDto dto){
    try {
        accountService.register(dto);
        return ResponseEntity.status(HttpStatus.CREATED).build();
    } catch (IllegalArgumentException e){
        return ResponseEntity.status(HttpStatus.CONFLICT)
            .body(Map.of("error", e.getMessage()));
    }
}
```

Регистрация нового пользователя в системе

Рисунок 3.14 – Пример докер контейнера

```
spring:
  application:
    name: auth-service
  datasource:
    driver-class-name: org.postgresql.Driver
    username: postgres
    password: postgres
    url: jdbc:postgresql://localhost:5432/task_app

  jpa:
    hibernate:
      ddl-auto: update
      open-in-view: false
    sql:
      init:
        schema-locations:
          - classpath:initial/*.sql
      mode: always

server:
  port: 8071
  servlet:
    session:
      timeout: 1m
```

Докер контейнер с базой данных

Рисунок 3.15 – Пример кода для работы с базами данных

```
@Entity
@Table(name = "users")
@FieldDefaults(level = AccessLevel.PRIVATE)
@AllArgsConstructor
@NoArgsConstructor
@Getter
@Setter
public class UserEntity {

    @Id
    String username;

    @NotNull
    String password;

    @NotNull
    Boolean enabled;

    @OneToOne
    @JoinColumn(name = "teacher_id", unique = true)
    TeacherEntity teacher;
}
```

Пример класса, копирующего  
структуру базы данных, нужного для  
работы с данными из бд в джаве

Рисунок 3.16 – Пример кода формы логина

```
<template>
<div class="min-h-screen flex items-center justify-center bg-gradient-to-tr from-[#C9D7FF] from-
<div class="bg-white p-8 rounded-lg shadow-md w-auto h-auto border border-gray-300">
<div class="grid grid-cols-[50px_1fr] mb-6">

<h2 class="text-center place-self-start ml-4 mt-2
text-xl font-medium text-gray-800">Stankin Works</h2></div>
<form @submit.prevent="handleSubmit">
<div class="mb-4">
<label class="block text-gray-700 mb-1">Login</label>
<input
v-model="login"
type="text"
class="w-full px-4 py-2 border border-gray-300 rounded-lg
focus:outline-none focus:border-indigo-400"
/>
</div>
<div class="mb-6">
<label class="block text-gray-700 mb-1">Password</label>
<input
v-model="password"
type="password"
class="w-full px-4 py-2 border border-gray-300 rounded-lg
focus:outline-none focus:border-indigo-400"
/>
</div>
<button
type="submit"
class="w-full bg-black text-white py-2 rounded-lg hover:bg-gray-800 transition"
>
Sign In
</button>
</form>
<div class="mt-4 text-center">
</div>
</div>
</div>
```

Форма логина на vue.js

Рисунок 3.17 – Пример кода формы панели администратора

```
<!-- Список преподавателей -->
<div class="space-y-4">
  <div>
    <v-for="(teacher, index) in filteredTeachers"
      :key="index"
      class="bg-[#8388F5] p-4 rounded-2xl shadow-md">
      <p class="text-lg font-medium">{{ teacher.name }}</p>
      <div class="mt-2 flex flex-wrap gap-2">
        <button class="bg-white px-4 py-1 rounded-full text-sm hover:bg-gray-200 transition">Изменить пароль</button>
        <button class="bg-white px-4 py-1 rounded-full text-sm hover:bg-gray-200 transition">Удалить аккаунт</button>
        <button class="bg-white px-4 py-1 rounded-full text-sm hover:bg-gray-200 transition">Сделать админом</button>
      </div>
    </div>
  </div>
</div>
```

Часть формы админ  
панели, элемент  
пользователя, и кнопки  
взаимодействия с ними

Рисунок 3.18 – Пример кода настройки связи адресных путей

```
const routes : [{path: string, redirect: string}] = [
  { path: '/', redirect: '/login' },
  { path: '/login', component: Login },
  { path: '/auth-callback', component: AuthorizedPage },
  { path: '/works', component: WorkList },
  { path: '/preview', component: WorkPreview },
  { path: '/upload', component: UploadPage },
  { path: '/admin', component: AdminPanel }
]

const router : Router = createRouter({ options: {
  history: createWebHistory(),
  routes
}})

router.beforeEach( guard: (to : RouteLocationNormalized, from : RouteLocationNormalizedLoaded, next : NavigationGuardNext) : void => { :
  const publicPages : string[] = ['/login', '/auth-callback']
  const accessToken : string = localStorage.getItem( key: 'access_token')

  if (accessToken && typeof accessToken === 'string') {
    try {
      const decoded : JwtPayload = jwtDecode(accessToken);

      const isAdmin = decoded.roles.includes( key: 'ROLE_ADMIN');

      if (to.path === '/admin' && !isAdmin) {
        next('/works');
      }
      else if (to.path !== '/admin' && isAdmin) {
        next();
      }
      else {
        next();
      }
    }
  }
})
```

Настройка связи адресных  
путей с компонентами на  
vue.js



Рисунок 3.19 – Пример кода для формы загрузки

```
<div class="mb-8">
  <label class="block text-xl font-medium mb-2">ФАЙЛ РАБОТЫ (PDF)</label>
  <div
    @dragover.prevent="dragOver = true"
    @dragleave="dragOver = false"
    @drop.prevent="handleFileDrop"
    :class="{
      'border-indigo-500 bg-indigo-50': dragOver,
      'border-gray-300': !dragOver
    }"
    class="border-2 border-dashed rounded-lg p-8 text-center cursor-pointer transition-colors"
    @click="triggerFileInput"
  >
    <input
      ref="fileInput"
      type="file"
      accept=".pdf"
      class="hidden"
      @change="handleFileSelect"
    />
    
    <p v-if="!selectedFile">Перетащите файл сюда или кликните для выбора</p>
    <p v-else class="text-lg font-medium text-indigo-600">{{ selectedFile.name }}</p>
    <p class="text-sm text-gray-500 mt-2">Поддерживаются только PDF-файлы</p>
  </div>
</div>
```

Форма загрузки  
дипломов в приложение

Рисунок 3.20 – Пример кода формы перехвата одноразового кода

```
<script setup>
onMounted( hook: async () => {
  const code = route.query.code;
  const codeVerifier = sessionStorage.getItem( key: 'pkce_code_verifier' );

  if (!code || !codeVerifier) {
    router.push('/login');
    return;
  }

  try {
    const response = await fetch( input: 'http://localhost:8071/oauth2/token', init: {
      method: 'POST',
      headers: {
        'Content-Type': 'application/x-www-form-urlencoded',
        'Authorization': 'Basic ' + btoa( data: 'client:secret' )
      },
      body: new URLSearchParams( init: {
        grant_type: 'authorization_code',
        code,
        redirect_uri: 'http://localhost:5173/auth-callback',
        code_verifier: codeVerifier,
        client_id: 'client'
      } ),
      credentials: 'include'
    });

    if (!response.ok) {
      const error = await response.json().catch(() => ({}));
      throw new Error(error.error || 'Ошибка получения токена');
    }

    const tokens = await response.json();

    localStorage.setItem('access_token', tokens.access_token);
    localStorage.setItem('refresh_token', tokens.refresh_token || '');
    sessionStorage.removeItem( key: 'pkce_code_verifier' );
  }
}
```

Форма перехвата  
одноразового кода на втором  
этапе аутентификации  
пользователя

## **ГЛАВА 4. ТЕСТИРОВАНИЕ И ОТЛАДКА СИСТЕМЫ, РЕКОМЕНДАЦИИ ПО ЕЕ ВНЕДРЕНИЮ**

### **4.1. Тестирование и отладка системы**

Проведя анализ работы, было принято решение проводить тестирование системы с помощью методов:

- Функциональный метод черного ящика
- Метод Usability-тестирования

Оба метода дополняют друг друга, метод Usability-тестирования помогает понять, насколько удобен и понятен интерфейс, а функциональный метод черного ящика проверяет весь функционал, какую реакцию получает пользователь при взаимодействии с интерфейсом.

Мы дали воспользоваться веб-приложением 10 пользователям, среди которых были, и сотрудники кафедры. Наша цель была узнать, понятен ли созданный интерфейс, а также какое поведение ожидают от системы пользователи и какой отклик на самом деле получают. И вот какой результат получили:

Вход:

- Понятные поля “Login” и “Password”
- Нет кнопки «Регистрация» или «Забыли пароль?»

Основной интерфейс:

- Меню слева читаемое: «Загрузить работу», «Список работ»
- Есть фильтры по автору, руководителю и году
- Кнопки “Просмотр” и “Скачать” интуитивно понятны
- Есть возможность сразу проверить работу на антиплагиат

- Отражается процент написания диплома
- Нет визуальной перегрузки, дизайн минималистичный

Недочеты по UX:

- На некоторых страницах веб-приложения хотелось бы чтобы меню было короче
- Реализовать UI элементы для всех новых функций

Недочеты по возможностям веб-приложения:

- Добавить возможность выгрузки отчетов в формате excel
- Добавить мобильную версию приложения

Также мы составили тест-кейс для веб-приложения с ожидаемым поведением системы и ее реальным откликом:

Функция	Ожидаемое поведение	Результат
Вход в систему	Успешный вход с валидными данными	Да
Неверный логин/пароль	Сообщение об ошибке	Да
Загрузка новой работы	Загружает файл, добавляется в список	Да
Поиск по автору/руководителю	Фильтрует список работ	Да
Сброс фильтров	Возвращает все записи	Да
Просмотр работы	Открывается страница или модальное окно	Да
Скачивание файла	Файл загружается	Да
Проверка оригинальности текста	Показывает процент или отчёт	Да

## 4.2. Рекомендации по внедрению

### Этапы внедрения системы

#### Тестовый запуск (пилотное внедрение)

- Развертывание системы на сервере университета в ограниченном режиме.
- Подключение одной кафедры (например, **Кафедры компьютерных систем управления**) для тестирования функционала.
- Сбор обратной связи от преподавателей и корректировка системы.

#### Обучение пользователей

- Проведение обучающих семинаров для преподавателей и администраторов.
- Создание инструкций (текстовых и видеоматериалов) по работе с системой.

#### Полномасштабное внедрение

- Подключение всех кафедр, работающих с выпускными квалификационными работами (ВКР).
- Интеграция с существующими системами университета (например, электронной библиотекой или системой антиплагиата).

#### Мониторинг и поддержка

- Назначение ответственного за техническую поддержку системы.
- Регулярное обновление и доработка функционала по запросам пользователей.

Для корректной работы веб-приложения требуется:

- Yandex браузер
- Университетская база данных
- Резервное копирование данных (первичный бэкап)

## Список используемой литературы

1. **Сидоров А.В., Белкин О.А.** «Современные подходы к управлению данными в образовательных системах». Москва: Эксмо, 2020.
2. **Левин И.С., Крупнов В.И.** «Технологии цифровизации в образовании». Казань: Казанский федеральный университет, 2020.
3. **Тиунов Д.**, «Как работают веб-приложения». Москва, 2019  
[Электронный ресурс] URL: <https://habr.com/ru/articles/450282/> (дата обращения 25.09.2024).
4. **Сакулин С.А.** «Основы интернет-технологий». Москва: МГТУ им. Н.Э. Баумана, 2024
5. **Соцкий Н.**, «Мобильное приложение или веб». Москва, 2024  
[Электронный ресурс] URL: <https://instadev.ru/mobilnye-prilozheniya-ili-web-prilozheniya-> (дата обращения 28.09.2024)
6. **Дергачев А.М., Кореньков Ю.Д., Логинов И.П., Сафронов А.Г.**, «Технологии веб-сервисов». Санкт-Петербург: ИТМО, 2021
7. **Некрасов В.**, «Что лучше: приложение или сайт». Москва, 2024  
[Электронный ресурс] URL: <https://veonix.ru/blog/что-luchshe-prilozhenie-ili-sayt/> (дата обращения 01.10.2024)
8. **Яковлев Р.**, «LMS: Виды и задачи». Москва, 2023 [Электронный ресурс] URL:  
<https://gb.ru/blog/lms/#:~:text=Выделяют%20два%20вида%20LMS%3A%20серверные,не%20единственные%20критерии%20выбора%20LMS>  
(дата обращения 01.10.2024)
9. Документация Figma. "Figma Basics". [Электронный ресурс] URL:  
<https://www.figma.com> (дата обращения 25.10.2024)
10. FlexBerry, «Построение диаграммы классов» [Электронный ресурс]  
URL: [https://flexberry.github.io/ru/gpg\\_class-diagram.html](https://flexberry.github.io/ru/gpg_class-diagram.html) (дата обращения 02.11.2024)

11. **Дж. Рамбо, М.Блаха.** UML 2.0. Объектно-ориентированное моделирование и разработка. 2-е изд. - СПб.: Питер, 2007. - 544с.:ил.
12. **Скотт К., Розенберг Д.** Применение объектного моделирования с использованием UML и анализ прецедентов: на примере книжного Internet-магазина. Пер. с англ.: руководство Изд.: ДМК Пресс Г од: 2007