Министерство просвещения Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Набережночелнинский государственный педагогический университет»

Факультет математики и информатики

Кафедра информатики и вычислительной математики

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему:

**РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ООО «Клуб Обучения»**

Направление подготовки 09.03.03 Прикладная информатика в дизайне

|  |  |  |  |
| --- | --- | --- | --- |
| Руководитель  И.о.зав.каф-ры ИиВМ | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | Герасимова О.Ю. |
|  |  | |  |
| Обучающийся | \_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | | Гилметдинова А.Т.  Номер группы 2021 |

Набережные Челны, 202

Оглавление

[Введение 3](#_Toc189244870)

[1 Исследование существующих технологий дизайн-проектирования 5](#_Toc189244871)

[1.1 Обзор современных технологий и роль пользовательского опыта в проектировании интерфейсов 5](#_Toc189244872)

[1.2 Используемые инструменты и фреймворки для проектирования интерфейсов 8](#_Toc189244873)

[1.3 Изучение интерфейсов приложений учебных заведений 13](#_Toc189244874)

[1.4 Анализ аналогичных решений 15](#_Toc189244875)

[2 Дизайн-проектирование пользовательского интерфейса 20](#_Toc189244876)

[2.1 Анализ потребностей целевой аудитории 20](#_Toc189244877)

[2.2 Выбор цветовой гаммы и типографики 23](#_Toc189244878)

[2.3 Разработка структуры и макета 25](#_Toc189244879)

[Заключение 29](#_Toc189244880)

[Список использованных источников 31](#_Toc189244881)

# Введение

Современное образование стремительно трансформируется под влиянием цифровых технологий, что открывает новые возможности для оптимизации учебных процессов. Онлайн-платформы, системы управления обучением и инструменты автоматизации становятся неотъемлемой частью образовательных организаций, позволяя сократить административную нагрузку и повысить качество взаимодействия между участниками процесса. Однако многие учреждения, особенно малые и средние, сталкиваются с отсутствием решений, адаптированных под их уникальные потребности, что замедляет их переход к цифровым форматам.

ООО «Клуб Обучения» — образовательная организация, специализирующаяся на проведении групповых и индивидуальных курсов по различным дисциплинам. Компания активно развивается, увеличивая количество студентов и преподавателей, однако текущие процессы управления группами, расписанием и учебными материалами организованы с использованием разрозненных инструментов (электронные таблицы, мессенджеры, файловые хранилища). Это приводит к дублированию данных, ошибкам в планировании и сложностям контроля успеваемости. Отсутствие единой цифровой платформы ограничивает масштабируемость услуг и снижает удовлетворенность клиентов. С ростом числа клиентов и расширением перечня услуг организация столкнулась с необходимостью оптимизации внутренних процессов, что требует внедрения современных цифровых решений.

Актуальность исследования обусловлена необходимостью разработки специализированного веб-приложения, которое объединит все ключевые функции управления образовательной деятельностью «Клуба Обучения» в едином интерфейсе. Внедрение такого решения позволит устранить существующие операционные недостатки, автоматизировать рутинные задачи и обеспечить прозрачность учебного процесса для всех участников.

Объект исследования — процессы управления образовательной деятельностью в ООО «Клуб Обучения», включая формирование учебных групп, планирование занятий, контроль посещаемости и оценку результатов студентов.

Предмет исследования — методы проектирования и разработки веб-приложений, ориентированных на автоматизацию деятельности образовательных организаций.

Цель работы — создание веб-приложения «LearningClub», обеспечивающего комплексную автоматизацию управления учебными группами, курсами, расписанием и аналитикой для ООО «Клуб Обучения».

Для достижения цели необходимо решить следующие задачи:

* Провести анализ существующих платформ для управления образованием и выявить их ограничения.
* Определить функциональные и нефункциональные требования к системе на основе потребностей заказчика.
* Разработать модель данных и функциональную модель приложения.
* Спроектировать архитектуру системы и выбрать подходящие технологии разработки.
* Реализовать базовые модули: управление пользователями, группами, расписанием, материалами и оценками.
* Провести тестирование системы и оценить ее соответствие требованиям заказчика.

Структура работы отражает логику исследования. В первой главе анализируются требования к системе. Во второй проектируются модели данных и бизнес-процессов. Третья глава посвящена выбору технологий, разработке и тестированию приложения.

1. Исследование существующих технологий дизайн-проектирования

# Описание предметной области

ООО «Клуб Обучения» предоставляет услуги группового и индивидуального обучения по направлениям, таким как дисциплины в области информационных технологий, далее IT, подготовка к экзаменам и иные образовательные программы. Работа направлена на автоматизацию управления учебным процессом организации. В современных условиях высоких требований к качеству образовательных услуг использование разрозненных инструментов для ведения расписания, контроля посещаемости, хранения учебных материалов и сбора оценок становится серьезной проблемой.

В настоящее время расписание занятий оформляется в электронных таблицах, что затрудняет оперативное обновление информации и обмен данными. Фиксация посещаемости осуществляется вручную, что требует временных затрат и повышает риск ошибок при обработке данных. Учебные материалы хранятся в облачных хранилищах, но без структурной привязки к конкретным занятиям, а сбор оценок производится через мессенджеры и электронную почту. Такая фрагментация информации приводит к необходимости постоянной ручной синхронизации данных, что снижает эффективность работы и увеличивает вероятность допущения ошибок.

Современные требования рынка требуют создания единой системы, которая объединяет все аспекты управления учебным процессом в одном интерфейсе. Централизованная платформа должна обеспечивать планирование занятий, контроль посещаемости, ведение базы оценок и организацию доступа к учебным материалам. Это позволит существенно сократить временные затраты на административные операции, снизить вероятность ошибок при обработке данных и обеспечить оперативный обмен информацией между всеми участниками учебного процесса.

Одной из ключевых задач работы является разработка решения, способного заменить разрозненные инструменты единым программным продуктом. В результате автоматизация позволит обеспечить прозрачность и актуальность информации, необходимой для принятия управленческих решений. Преподаватели и администрация получат возможность быстро получать данные по успеваемости и посещаемости, что облегчит корректировку учебных программ и оперативное реагирование на изменения в расписании. Кроме того, централизованное хранение информации повысит качество аналитики, необходимой для оценки эффективности образовательного процесса.

Разработка системы предполагает обеспечение высокого уровня безопасности, так как обработка данных включает личную информацию студентов, расписания занятий, оценки и другие конфиденциальные сведения. Применение современных методов шифрования и многоуровневой аутентификации позволит снизить риск несанкционированного доступа и утечки данных. Масштабируемость системы также является важным аспектом, позволяющим в дальнейшем адаптировать платформу к увеличению количества пользователей и расширению перечня образовательных программ.

Реализация данной работы позволит не только оптимизировать управление учебным процессом, но и создать условия для повышения качества образовательных услуг. Сокращение временных затрат на административные задачи и минимизация риска ошибок при обработке информации будут способствовать улучшению взаимодействия между преподавателями и студентами. В конечном итоге, создание единой информационной платформы станет важным инструментом для оперативного управления учебным процессом и повышения конкурентоспособности организации на рынке образовательных услуг.

Переход от использования разрозненных инструментов к интегрированной системе управления представляет собой стратегически важное направление для развития организации. Автоматизация процессов позволит объединить данные о расписании занятий, посещаемости, оценках и учебных материалах в одной базе, что обеспечит более эффективное планирование и контроль за учебным процессом. Это, в свою очередь, будет способствовать не только повышению качества образовательных услуг, но и улучшению условий работы административного персонала.

Кроме того, интегрированное решение предоставит возможность для реализации дополнительных функций, таких как генерация аналитических отчетов, настройка уведомлений для преподавателей и студентов, а также автоматизированное обновление информации в режиме реального времени. Все это позволит значительно упростить управление учебным процессом и обеспечить оперативную реакцию на любые изменения. Таким образом, разработка единой системы управления станет ключевым элементом в процессе модернизации работы организации, способствуя улучшению качества образовательного процесса и созданию благоприятных условий для дальнейшего развития образовательных программ.

# Анализ существующих решений и их недостатков

Для определения требований к веб-приложению «Клуба Обучения» были проанализированы современные образовательные платформы, включая «HTML Academy», «Яндекс.Практикум» и «Hexlet». Каждая из них демонстрирует уникальные подходы к организации обучения, но не полностью соответствует потребностям компании, что подчеркивает необходимость создания специализированного решения.

«HTML Academy» выделяется практико-ориентированным форматом, где студенты осваивают веб-разработку через интерактивные задания и симуляторы работы с кодом. Платформа автоматизирует проверку заданий, обеспечивая мгновенную обратную связь, что значительно сокращает время на рутинные операции. Однако её фокус на самостоятельном обучении исключает возможность управления группами, контроля посещаемости и интеграции с живыми занятиями. Для «Клуба Обучения», где ключевую роль играет взаимодействие преподавателя со студентами, это становится критичным ограничением.

«Яндекс.Практикум» предлагает глубоко структурированные курсы по IT-профессиям с поддержкой наставников. Система сочетает теорию, практику и проекты, а обратная связь от экспертов помогает студентам корректировать ошибки. Несмотря на это, платформа ориентирована на массовое обучение с жестко заданными сроками и программой. Это не позволяет адаптировать курсы под динамичные запросы студентов «Клуба Обучения», где требуется гибкость в изменении расписания, добавлении новых модулей или формировании групп на основе предварительного тестирования.

«Hexlet» делает упор на самостоятельное решение практических задач по программированию с автоматической проверкой кода. Его минималистичный интерфейс и фокус на проектах для портфолио привлекают тех, кто предпочитает учиться в своем темпе. Однако полное отсутствие инструментов для планирования занятий.

Отсутствие учета посещаемости и живого взаимодействия с преподавателями делает платформу неприменимой для формата «Клуба Обучения», где важен человеческий контроль над процессом и возможность оперативной корректировки программ.

Крупные LMS-платформы, такие как Moodle, предоставляют широкий набор функций для создания курсов и управления материалами, но их сложность и ориентация на академические учреждения затрудняют использование в условиях узкопрофильного IT-обучения. Например, интеграция с сервисами видеосвязи требует установки дополнительных модулей, а отсутствие встроенной аналитики посещаемости усложняет администрирование.

Анализ показал, что существующие платформы либо полностью автоматизированы, исключая живое взаимодействие, либо избыточно сложны для решения специфических задач «Клуба Обучения». Заимствовать стоит практико-ориентированный подход HTML Academy, систему обратной связи Яндекс.Практикума и автоматизацию проверки заданий Hexlet. При этом важно избежать их ключевых ограничений: отсутствия гибкости, жесткой структуры курсов и ориентации на самостоятельное обучение.

Для «Клуба Обучения» критично сохранить баланс между автоматизацией рутинных задач (учет посещаемости, генерация отчетов) и возможностью ручного управления процессами (корректировка расписания, адаптация материалов). Это требует разработки специализированного решения, которое объединит сильные стороны анализируемых платформ, устранив их недостатки, и обеспечит прозрачность, гибкость и персонализацию учебного процесса.

# Формулировка бизнес-требований

Бизнес-требования к веб-приложению «LearningClub» сформированы на основе анализа текущих проблем ООО «Клуб Обучения» и направлены на достижение ключевых целей компании: повышение эффективности управления учебным процессом, сокращение временных затрат на рутинные операции и обеспечение масштабируемости бизнеса.

Первым требованием является автоматизация планирования занятий. Система должна предоставлять инструменты для создания гибкого расписания, учитывающего доступность преподавателей, пожелания студентов и технические возможности проведения онлайн-занятий. Это позволит избежать накладок, устаревших ссылок на вебинары и дублирования информации, характерных для текущего ручного управления.

Второе требование связано с учетом посещаемости и успеваемости. Платформа должна автоматически фиксировать присутствие студентов на занятиях через интеграцию с сервисами видеосвязи, а также предоставлять интерфейс для ручного ввода данных. Для контроля успеваемости необходима система выставления оценок с возможностью их анализа в разрезе групп, модулей и отдельных студентов. Это обеспечит прозрачность данных и сократит время на формирование отчетов.

Третье требование касается централизованного хранения и управления учебными материалами. Все материалы (лекции, задания, записи вебинаров) должны быть привязаны к конкретным занятиям и доступны в едином интерфейсе. Это исключит необходимость использования сторонних облачных хранилищ и упростит поиск информации для студентов и преподавателей.

Четвертое требование — возможность перехода на внешние сервисы. Платформа должна поддерживать подключение к популярным инструментам видеоконференц-связи (Zoom, Google Meet) и облачным хранилищам (Google Drive, Яндекс.Диск) для автоматического обновления ссылок на занятия и материалы. Это минимизирует ручные операции и снизит риск ошибок.

Пятое требование предполагает обеспечение безопасности данных. Система должна соответствовать стандартам защиты персональной информации, включая шифрование данных, разграничение прав доступа (ролевая модель: студент, преподаватель, администратор) и резервное копирование. Это критично для работы с конфиденциальными данными студентов и преподавателей.

Шестое требование — масштабируемость и адаптивность. Платформа должна быть рассчитана на рост числа пользователей и курсов без снижения производительности. Гибкая архитектура позволит добавлять новые функции, такие как аналитика эффективности курсов, система уведомлений или интеграция с платежными сервисами, что поддержит развитие бизнеса в долгосрочной перспективе.

Седьмое требование — удобство использования. Интерфейс системы должен быть интуитивно понятен как для студентов, так и для преподавателей. Для студентов ключевым элементом станет личный кабинет с расписанием, материалами и историей оценок. Преподавателям потребуется инструментарий для управления группами, загрузки материалов и мониторинга прогресса. Администраторы должны иметь доступ к полной аналитике и настройкам системы.

Выполнение этих требований позволит «Клубу Обучения» перейти от фрагментированного управления к централизованной системе, обеспечивающей прозрачность, оперативность и качество образовательного процесса. Это создаст основу для сокращения административной нагрузки, повышения удовлетворенности клиентов и устойчивого роста компании.

# Определение функциональных и нефункциональных требований

Для формализации требований к веб-приложению «LearningClub» проведен анализ взаимодействия пользователей с системой. На основе ролевой модели (студент, преподаватель, администратор) выделены ключевые процессы (в таблице 1.4.1), которые охватывают все аспекты управления учебным процессом. Каждый процесс описан как последовательность действий, направленных на достижение конкретного результата, что позволяет четко определить функциональные возможности системы и требования к её качеству.

Таблица 1.4.1 – Ключевые процессы

|  |  |  |
| --- | --- | --- |
| Название | Тип | Роль |
| Авторизация | Событие | Студент |
| Просмотр расписания | Результат | Студент |
| Доступ к материалам | Результат | Студент |
| Просмотр оценок | Результат | Студент |
| Участие в занятии | Событие | Студент |
| Загрузка домашнего задания | Событие | Студент |
| Создание занятия | Событие | Администратор |
| Загрузка материалов | Событие | Преподаватель |
| Выставление оценок | Решение | Преподаватель |
| Контроль посещаемости | Решение | Преподаватель |
| Проверка домашних заданий | Решение | Преподаватель |
| Управление пользователями | Событие | Администратор |
| Формирование групп | Событие | Администратор |

Диаграмма прецедентов из Unified Modeling Language, далее UML, является мощным инструментом для формализации требований к системе, так как она позволяет наглядно представить взаимодействие пользователей с системой и выявить важнейшие процессы, которые должны быть поддержаны в рамках разработки. С помощью диаграммы можно визуализировать любые действия пользователей. В особенности полезно использовать эту диаграмму, поскольку она помогает понять, какие процессы преобладают в рамках системы, как они взаимосвязаны и какие примечания должны быть учтены при проектировании.

Создание диаграмм прецедентов способствует лучшему пониманию структуры системы и помогает более четко определять функциональные требования, уменьшать неопределенность и обеспечивать развитие интерфейсов, совместимых с нуждами различных пользователя.

Диаграмма позволяет наглядно представить все ключевые взаимодействия между пользователями и системой, что существенно облегчает понимание функциональных требований. Графическое изображение процессов помогает быстро выявить взаимосвязи между ними, определить критические точки и оптимизировать логику работы системы. Использование диаграммы наряду с табличным описанием обеспечивает полноту и точность формулировки требований, а также способствует лучшей коммуникации между всеми участниками разработки, поскольку наглядное представление упрощает обсуждение и согласование деталей проекта.

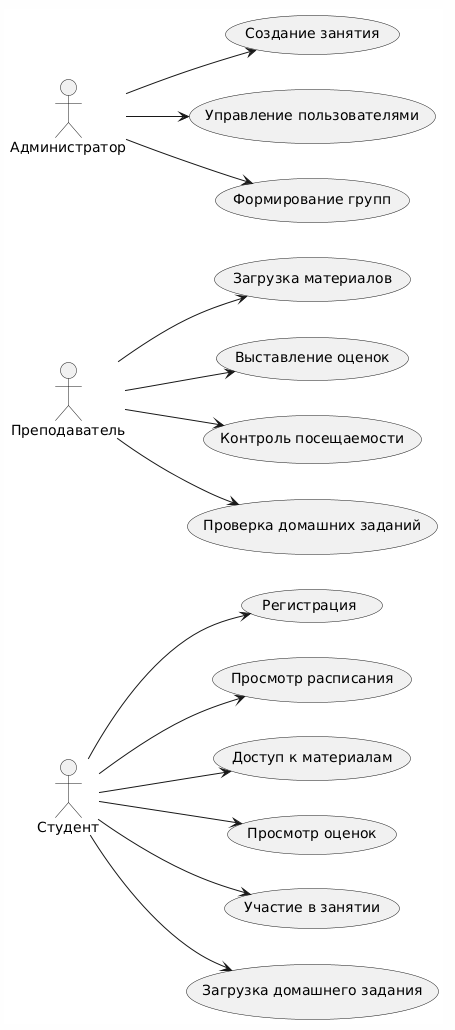


Рисунок 1.4.1 – Диаграмма прецедентов

Студент начинает взаимодействие с системой с процесса авторизации. Он предоставляет свои личные данные, такие как имя пользователя и пароль, после чего получает доступ к личному кабинету. В личном кабинете студент может просматривать расписание занятий своей группы. Расписание отображается в виде календаря с указанием времени, темы и формата каждого занятия, что позволяет студенту планировать свое время и своевременно подключаться к урокам.

Для подготовки к занятиям студент использует раздел доступа к учебным материалам. Здесь он может просматривать и скачивать лекции, задания, презентации и записи вебинаров, привязанные к конкретным урокам. Материалы структурированы по модулям и темам, что упрощает поиск и использование.

После выполнения домашнего задания студент загружает его в систему. Задание привязывается к конкретному уроку и становится доступным для проверки преподавателем. Это позволяет студенту своевременно сдавать работы и получать обратную связь.

Для участия в занятии студент подключается к онлайн-уроку, используя ссылку на внешнюю систему видеоконференции. Преподаватель сам оформляет и обновляет ссылки для каждого урока, что обеспечивает гибкость в выборе конкретного инструмента для связи. В этом случае система не занимается автоматизацией подключения, а лишь обеспечивает предоставление ссылки и фиксирование активности пользователя в рамках урока.

После занятия студент может просмотреть свои оценки в соответствующем разделе. Оценки отображаются в виде таблицы с возможностью сортировки по датам и темам, что помогает отслеживать прогресс и корректировать усилия.

Преподаватель взаимодействует с системой через процессы создания занятий и загрузки материалов. Он указывает тему, а также загружает лекции, задания и презентации.

Материалы привязываются к конкретным урокам, что обеспечивает их доступность для студентов.

После проведения занятия преподаватель контролирует посещаемость студентов. Он сам проставляет данные о посещаемости в систему, а также проверяет загруженные домашние задания, оставляет комментарии и выставляет оценки. Это обеспечивает обратную связь и помогает студентам улучшать свои результаты.

Администратор управляет пользователями системы, добавляя, редактируя и удаляя учетные записи студентов и преподавателей. Это обеспечивает актуальность данных и разграничение прав доступа.

Администратор также формирует учебные группы, распределяя студентов по уровням подготовки и курсам. Это позволяет организовать учебный процесс и обеспечить персонализированный подход к обучению.

Описание прецедентов позволяет четко определить функциональные требования к системе, а также выделить ключевые процессы, которые требуют реализации в веб-приложении. Это создает основу для проектирования архитектуры и разработки веб-приложения, рассчитанного на нужды всех участников учебного процесса.

Функциональные требования (указаны в таблице 1.4.2) представляют собой детальное описание функций, которые система должна выполнять для удовлетворения потребностей пользователей. Они определяют, как приложение должно обрабатывать входящие данные, реагировать на действия пользователя и обеспечивать выполнение ключевых бизнес-процессов. Четко сформулированные функциональные требования служат основой для проектирования архитектуры и разработки компонентов веб-приложения, снижая риск недопонимания между разработчиками и заказчиком.

Таблица 1.4.2 – Функциональные требования

|  |  |
| --- | --- |
| Идентификатор | Функции/Требования |
| F1 | Авторизация пользователей |
| FR-1.1 | Система должна предоставлять возможность входа в систему по имени пользователя и паролю |
| FR-1.2 | Система должна обеспечивать валидацию введенных данных |
| FR-1.3 | Система должна отображать уведомление об ошибке при вводе некорректных данных |
| F2 | Просмотр расписания |
| FR-2.1 | Система должна предоставлять студенту возможность просмотра расписания своей учебной группы |
| FR-2.2 | Система должна отображать дату, время, тему и формат занятия |
| F3 | Доступ к учебным материалам |
| FR-3.1 | Система должна предоставлять преподавателям возможность загрузки учебных материалов |
| FR-3.2 | Система должна предоставлять студентам возможность просмотра и скачивания загруженных материалов |
| FR-3.3 | Система должна обеспечивать структурирование материалов по модулям и темам |

Продолжение таблицы 14

|  |  |
| --- | --- |
| Идентификатор | Функции/Требования |
| F4 | Загрузка домашних заданий |
| FR-4.1 | Система должна предоставлять возможность загрузки студентами домашних заданий |
| FR-4.2 | Система должна привязывать загруженные задания к конкретным урокам |
| FR-4.3 | Система должна предоставлять преподавателям доступ к загруженным студентами работам |
| F5 | Участие в занятии |
| FR-5.1 | Система должна предоставлять студентам ссылку на онлайн-занятие |
| FR-5.2 | Система должна позволять преподавателю предоставлять ссылки на онлайн-уроки |
| F6 | Выставление оценок |
| FR-6.1 | Система должна предоставлять преподавателю возможность выставления оценок студентам за занятие |
| FR-6.2 | Система должна отображать оценки в личном кабинете студента |
| FR-6.3 | Система должна предоставлять возможность преподавателю выставления оценок за решения домашних заданий |

Продолжение таблицы 1.4.2

|  |  |
| --- | --- |
| Идентификатор | Функции/Требования |
| F7 | Контроль посещаемости |
| FR-7.1 | Система должна предоставлять преподавателю возможность вручную отмечать посещаемость студентов |
| FR-7.2 | Система должна сохранять данные о посещаемости для последующего анализа |
| F8 | Управление пользователями |
| FR-8.1 | Система должна предоставлять администратору возможность добавления новых пользователей |
| FR-8.2 | Система должна предоставлять администратору возможность редактирования и удаления учетных записей |
| FR-8.3 | Система должна разграничивать права доступа для студентов, преподавателей и администраторов |
| F9 | Формирование учебных групп |
| FR-9.1 | Система должна предоставлять администратору возможность распределения студентов по учебным группам |
| FR-9.2 | Система должна отображать список групп и состав участников |

Помимо функциональных требований, важную роль в разработке веб-приложения играют нефункциональные требования, которые определяют качественные характеристики системы. Они описывают аспекты производительности, безопасности, удобства использования, совместимости и масштабируемости. Эти требования обеспечивают надежность работы приложения, соответствие ожиданиям пользователей и возможность дальнейшего развития системы. Нефункциональные требования также включают ограничения на технологии и архитектуру, что помогает задать четкие рамки для реализации проекта. В таблице ниже представлены ключевые нефункциональные требования, которым должна соответствовать разрабатываемая система.

В таблице 1.6 показаны требования к пользовательским интерфейсам. Таблица 1.6 – Требования к пользовательским интерфейсам

|  |  |
| --- | --- |
| Идентификатор | Требования |
| U-1 | Система должна иметь интуитивно понятный интерфейс |
| U-2 | Студент должен иметь возможность быстро находить нужные материалы через удобную систему навигации |
| U-3 | Преподаватель должен иметь доступ к инструментам оценки и проверки домашних заданий без лишних действий |
| U-4 | Расписание занятий должно отображаться в удобном формате календаря или списка с возможностью фильтрации |

В таблице 1.7 показаны требования к интерфейсам ПО.

Таблица 1.7 – Требования к интерфейсам ПО

|  |  |
| --- | --- |
| Идентификатор | Требования |
| S-1 | API Системы должно быть реализовано в соответствии с подходом REST |

В таблице 1.8 показаны требования к производительности.

Таблица 1.8 – Требования к производительности

|  |  |
| --- | --- |
| Идентификатор | Требования |
| PRO-1 | Все веб-страницы должны загружаться не дольше 3 секунд |
| PRO-2 | Система должна обновлять данные в реальном времени |

В таблице 1.9 показаны требования к безопасности.

Таблица 1.9 – Требования к безопасности

|  |  |
| --- | --- |
| Идентификатор | Требования |
| SEC-1 | Пароли пользователей должны быть зашифрованы |
| SEC-2 | Пользователь должен быть авторизованным для выполнения операций в системе |

В результате проведённого анализа и формализации требований удалось чётко определить функциональные возможности системы, выделить ключевые процессы и сформировать перечень требований, необходимых для реализации веб-приложения. Разграничение функциональных и нефункциональных требований позволило уточнить, какие аспекты работы системы требуют автоматизации, а какие – обеспечения удобства, надёжности и безопасности.

# Проектирование системы управления учебным процессом

# Разработка функциональной модели

Для эффективного анализа и оптимизации бизнес-процессов в образовательных учреждениях широко используются методы функционального моделирования. Одним из наиболее популярных инструментов является диаграмма IDEF0, позволяющая графически представить систему, ее функции и взаимосвязи между ними. Этот метод применяется для детального описания процессов, определения их входных и выходных данных, а также выделения механизмов управления и ресурсов, необходимых для выполнения каждой функции.

Диаграмма IDEF0 особенно полезна при проектировании информационных систем, так как она помогает структурировать деятельность организации, выявить узкие места и определить направления для автоматизации. В рамках проектирования веб-приложения образовательного учреждения строятся две модели: «как есть» (as-is) и «как должно быть» (to-be).

Модель «как есть» (as-is) отображает текущее состояние системы до внедрения информационной системы. В ней представлены основные бизнес-процессы, выполняемые вручную или с использованием разрозненных инструментов. В текущем состоянии образовательный процесс организуется без централизованной информационной платформы, что приводит к неэффективности и затруднению взаимодействия между студентами, преподавателями и администрацией.

Основные функции модели as-is включают управление учебным процессом, проведение занятий, контроль выполнения заданий и оценивание. В данный момент преподаватели самостоятельно организуют занятия, используя различные сторонние инструменты для рассылки учебных материалов и заданий.

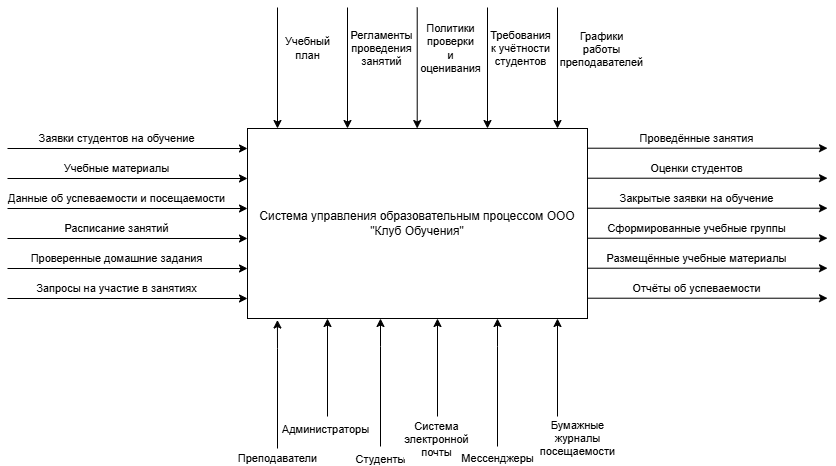


Рисунок 2.1.1 – Диаграмма IDEF0 модели «as-is»

Студенты, в свою очередь, вынуждены искать необходимую информацию в нескольких источниках, а сдача и проверка домашних заданий происходит без автоматизированного контроля. Администрация осуществляет управление пользователями и группами вручную, что усложняет процесс распределения студентов по курсам и отслеживание их успеваемости.

Модель «как должно быть» (to-be) представляет целевое состояние системы после внедрения веб-приложения. В данной модели учебный процесс становится более структурированным и автоматизированным. Все участники образовательного процесса получают доступ к единой платформе, где реализованы ключевые функции: управление учебными материалами, ведение расписания, загрузка и проверка домашних заданий, контроль посещаемости и оценивания.

Студенты получают удобный доступ ко всем необходимым материалам, могут просматривать расписание, загружать задания и получать обратную связь от преподавателей.

Преподаватели могут централизованно управлять учебными курсами, публиковать материалы, проверять работы студентов и выставлять оценки.

Администрация получает возможность эффективного управления пользователями, формирования учебных групп и контроля образовательного процесса.

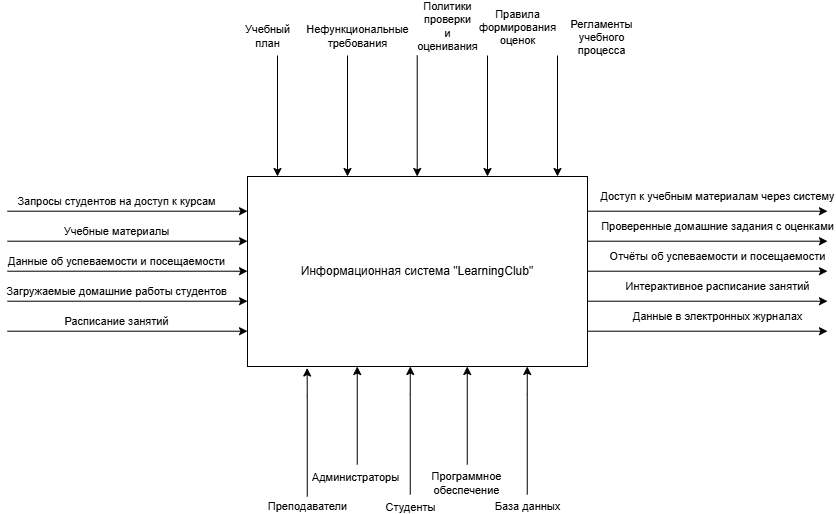


Рисунок 2.1.2 – Диаграмма IDEF0 модели «to-be»

Использование диаграмм IDEF0 для построения моделей «как есть» и «как должно быть» позволяет систематизировать знания о текущем состоянии системы и сформулировать четкие требования к разработке веб-приложения. Это помогает оптимизировать существующие процессы, выявить и устранить проблемы, а также создать эффективное и удобное цифровое решение для всех участников учебного процесса.

# Разработка модели данных

Диаграммы данных играют ключевую роль в процессе проектирования информационных систем, так как позволяют структурировать информацию, определить её сущности, взаимосвязи и ограничения. В контексте разработки веб-приложения для образовательного учреждения требуется построение трех уровней моделей данных: логической, инфологической и физической. Эти модели обеспечивают последовательный переход от концептуального описания данных к их реальному представлению в базе данных.

Логическая модель данных представляет основные сущности системы и их взаимосвязи, не учитывая специфические детали хранения. В данной системе ключевыми объектами являются студент, преподаватель, группа, курс, урок, домашнее задание, оценка и материал. Взаимосвязи между ними включают привязку студентов к группам, курсов к преподавателям, уроков к курсам, а также домашних заданий и материалов к урокам. Оценки привязываются к проверенным домашним заданиям, что позволяет фиксировать успеваемость студентов.

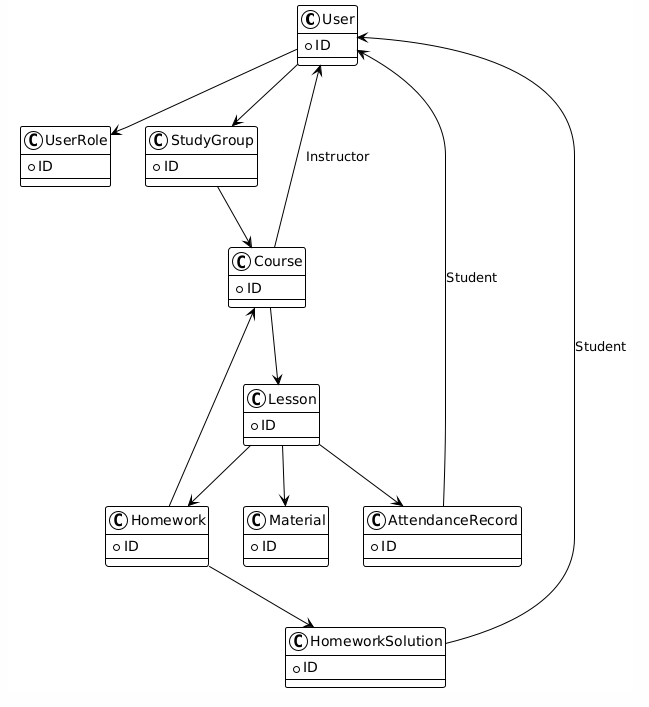


Рисунок 2.2.1 – Диаграмма классов логической модели данных

Инфологическая модель детализирует логическую модель, включая атрибуты сущностей, ограничения и бизнес-правила. В частности, в ней уточняются уникальные идентификаторы, обязательные поля и диапазоны значений. Например, студент должен быть идентифицирован по электронной почте, оценка может принимать значения только в установленном диапазоне, а домашние задания имеют строго определённые сроки сдачи. Также в рамках инфологической модели определяется, какие материалы могут быть прикреплены к урокам, включая текстовые файлы, ссылки и видео.

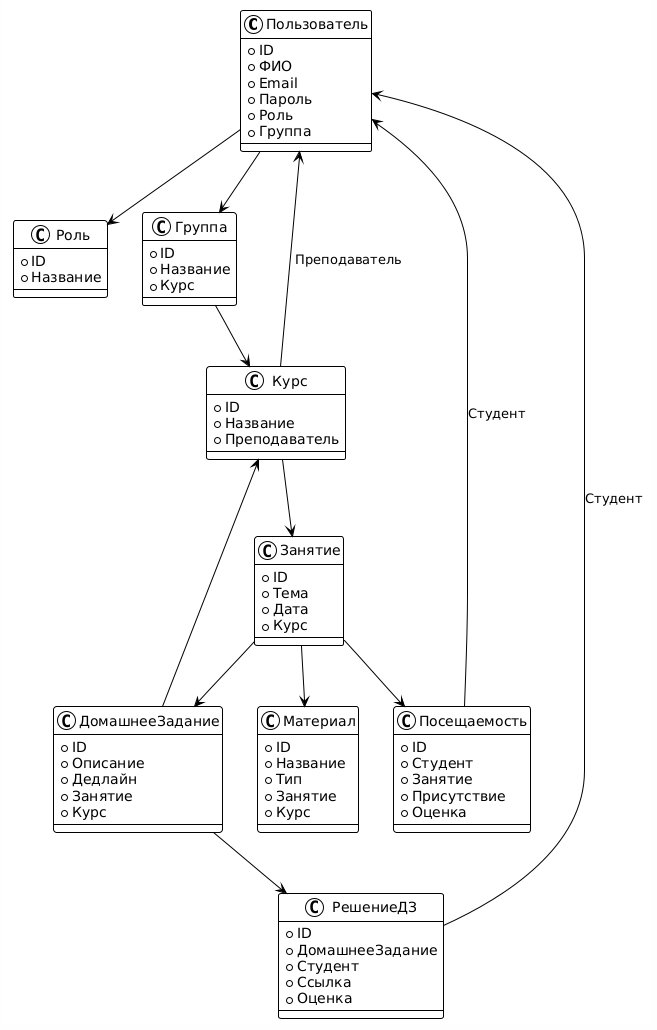


Рисунок 2.2.2 – Диаграмма классов инфологической модели данных

Физическая модель данных является завершающим этапом проектирования и описывает, как данные будут храниться в конкретной базе данных. На этом уровне выбирается СУБД и определяется структура таблиц, полей, индексов и связей. Например, таблица студентов будет содержать идентификатор, ФИО, электронную почту и связь с таблицей групп. Важно, чтобы база данных была оптимизирована для быстрого доступа и обработки информации, а также обеспечивала целостность данных за счёт использования внешних ключей и индексов.

Проектирование моделей данных позволяет создать прочную основу для функционирования системы, обеспечивая её надёжность, масштабируемость и соответствие функциональным требованиям. Эти модели служат фундаментом для последующей реализации базы данных и позволяют избежать проблем, связанных с дублированием, избыточностью и сложностью обработки данных.

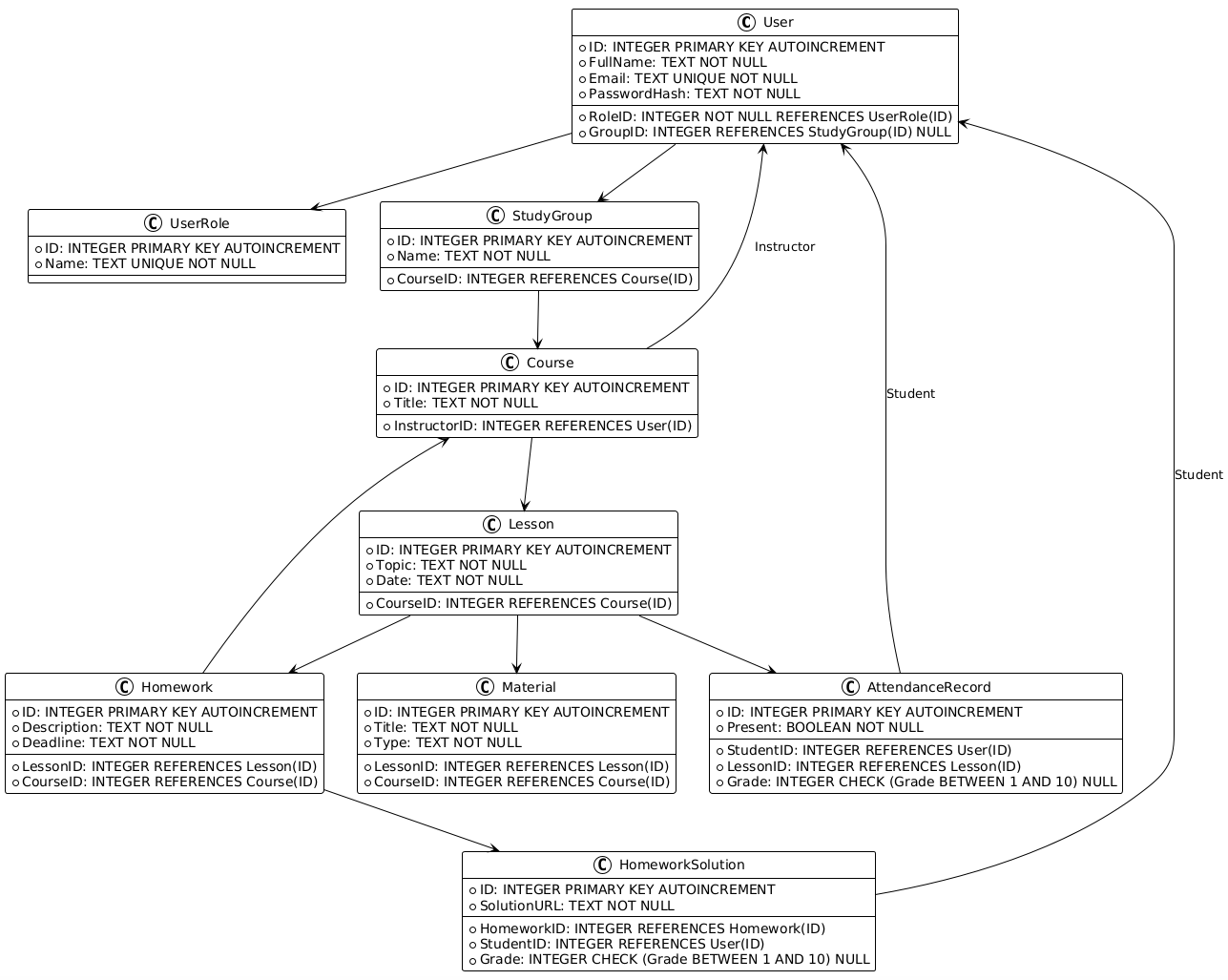


Рисунок 2.2.3 – Диаграмма классов физической модели данных

# Моделирование бизнес-процессов

В рамках проектирования информационной системы образовательного учреждения необходимо проанализировать текущие ключевые процессы и спроектировать их обновленные версии с учетом автоматизации. Далее рассмотрим основные процессы, которые будут поддерживаться системой: авторизация пользователя, управление расписанием, доступ к учебным материалам, работа с домашними заданиями и оценивание.

Диаграммы активности UML позволяют визуализировать ключевые бизнес-процессы системы, отображая последовательность действий и потоки управления. Они особенно полезны для моделирования сложных процессов с разветвленными сценариями, условиями и возможностью параллельного выполнения. Благодаря этому можно выявить потенциальные узкие места, проанализировать взаимодействие между пользователями и системой, а также спроектировать оптимальный порядок операций.

Процесс авторизации является начальной точкой взаимодействия с системой и обеспечивает контроль доступа к различным функциям в зависимости от роли пользователя. Пользователь вводит учетные данные, после чего система проверяет их корректность. Если введенные данные совпадают с данными в базе, система определяет роль пользователя и предоставляет ему доступ к личному кабинету. В противном случае выводится сообщение об ошибке, предлагая повторить ввод.

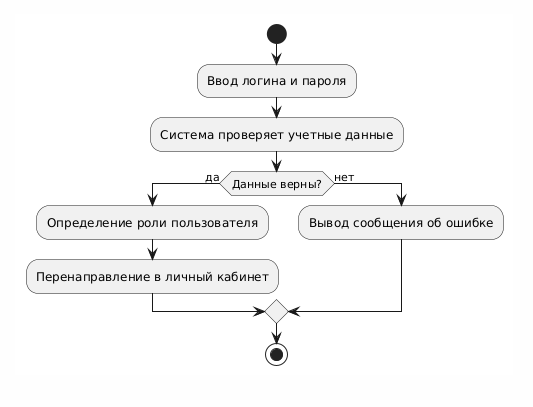


Рисунок 2.3.1 – Диаграмма активности процесса авторизации

После успешной авторизации пользователи могут выполнять доступные им операции. Одной из важных функций является управление расписанием, которое осуществляется администратором. Он создает занятия, указывая дату, время, тему и при необходимости добавляя ссылку на видеоконференцию. Это позволяет студентам своевременно получать информацию о предстоящих уроках и подключаться к занятиям в установленное время.

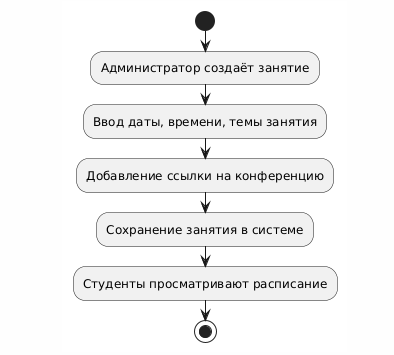


Рисунок 2.3.2 – Диаграмма активности добавления занятий

Помимо расписания, важной частью образовательного процесса является доступ к учебным материалам. Для подготовки к занятиям студенты заходят в соответствующий раздел системы, где выбирают нужный курс, затем конкретное занятие, после чего получают возможность просматривать и скачивать файлы. Это позволяет упростить доступ к материалам и избежать необходимости в дополнительных внешних источниках.

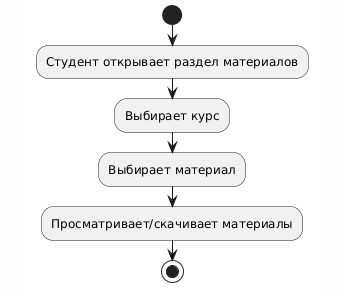


Рисунок 2.3.3 – Диаграмма активности процесса получения материалов

Дополнительно предусмотрена система работы с домашними заданиями. Преподаватели загружают задания, студенты выполняют их и загружают свои решения, после чего преподаватель проверяет работы и выставляет оценки. Вся эта последовательность действий фиксируется в системе, обеспечивая прозрачность и доступность данных для обеих сторон.

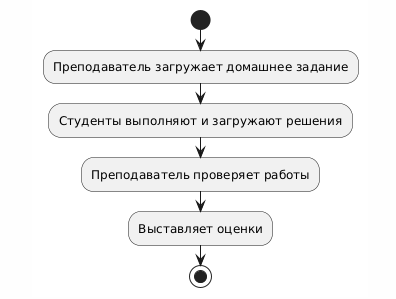


Рисунок 2.3.4 – Диаграмма активности процесса обработки домашней работы

Наряду с этим системой предусмотрен контроль посещаемости студентов на занятиях, добавления ссылок на материалы, а также возможность выставления оценки за каждое конкретное занятие. Преподаватель открывает список студентов, отмечает присутствие каждого участника и при необходимости выставляет оценку. Эти данные затем сохраняются в системе, позволяя вести учет успеваемости студентов.



Рисунок 2.3.5 – Диаграмма активности обработки занятий преподавателем

Проектирование бизнес-процессов позволило сформировать четкое представление о ключевых взаимодействиях пользователей с системой и определить их последовательность. Благодаря детальному анализу процессов удалось выявить возможные узкие места и предложить оптимальные решения для их автоматизации. Это способствует повышению эффективности работы образовательного учреждения, снижению нагрузки на сотрудников, а также улучшению пользовательского опыта для студентов.

Разработка моделей бизнес-процессов также помогла сформулировать требования к функциональности будущей системы. Определенные сценарии взаимодействия позволили зафиксировать конкретные задачи для системы, а также учесть возможные исключительные ситуации и способы их обработки.

# Разработка дизайна посадочной страницы

Перед переходом к разработке системы требуется создать посадочную страницу приложения, способную привлекать клиентов и выполняющую информирующий функционал. Разработка дизайна посадочной страницы основана на минималистичном стиле с использованием четкой структуры и ограниченной палитры цветов. Основной цвет интерфейса - пастельный зеленый (#96CC82), который применяется для кнопок и ключевых элементов взаимодействия. Фон сайта белый (#FFFFFF), что создает ощущение легкости и чистоты. Основной цвет текста и акцентных элементов - черный (#000000), а вспомогательный текст выполнен в сером цвете (#656161). Дополнительно некоторые области дизайна покрыты полупрозрачными пастельными цветами , что создает мягкий визуальный эффект без излишней перегруженности.

Шапка сайта включает в себя логотип или название проекта, а также горизонтально расположенное меню навигации. В подвале сайта (футере) размещены контакты и ссылки на социальные сети. В меню присутствуют ссылки на основные разделы: "Новости", "О нас", "Курсы" и "Войти". Вся навигация выполнена в лаконичном стиле без лишних элементов, что соответствует общей концепции минимализма.

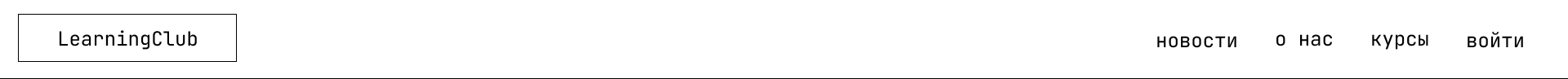


Рисунок 2.4.1 – Дизайн шапки посадочного модуля приложения

Футер сайта содержит контактную информацию и ссылки на социальные сети, обеспечивая пользователям удобный доступ к важным данным и дополнительным ресурсам.

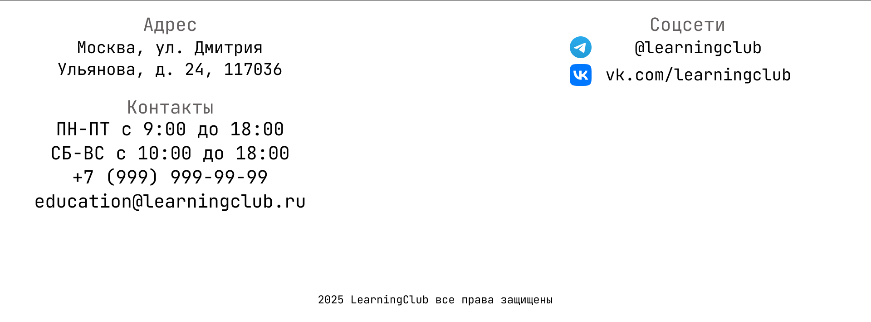


Рисунок 2.4.2 – Дизайн подвала посадочного модуля приложения

На странице "Новости" материалы располагаются в порядке убывания даты публикации, что обеспечивает удобство просмотра актуальной информации.

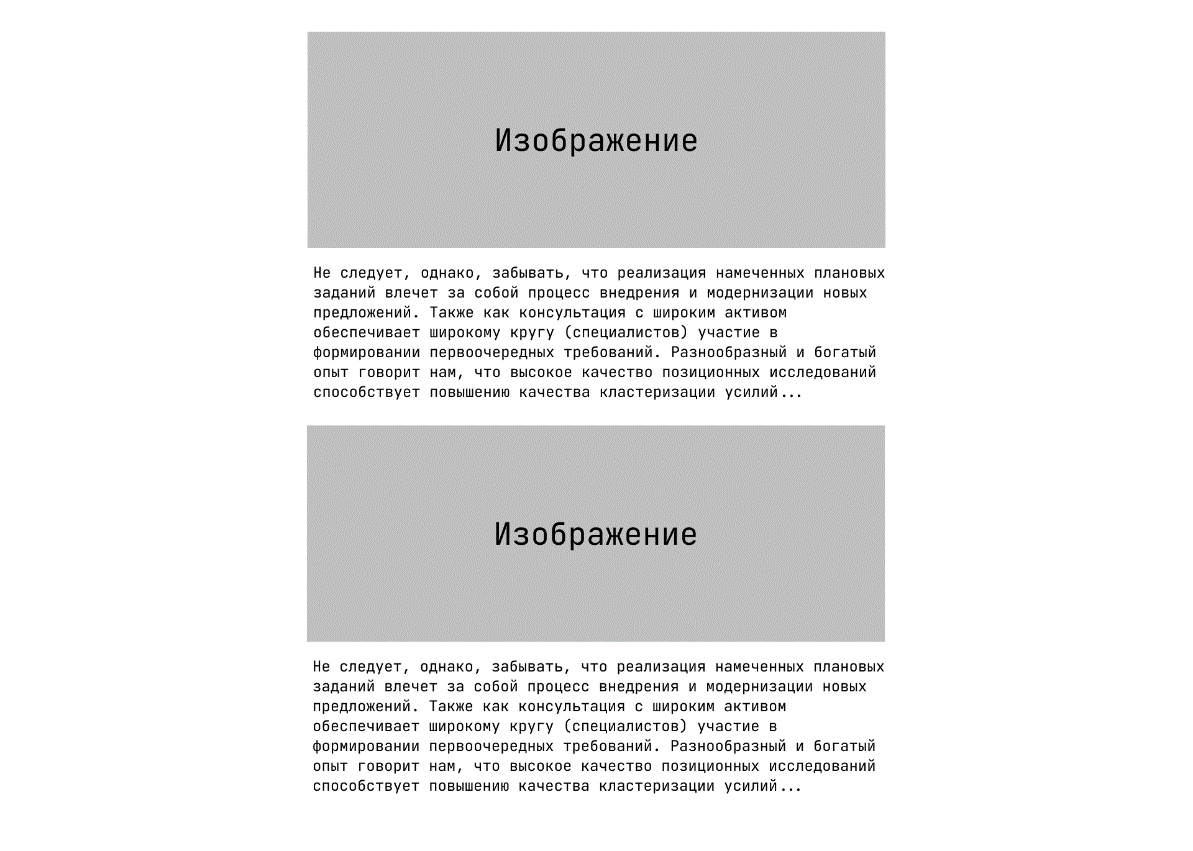


Рисунок 2.4.3 – Дизайн страницы «Новости»

Раздел "О нас" оформлен в виде шахматного расположения блоков текста и изображений. Изображения частично покрыты полупрозрачными пастельными оттенками.

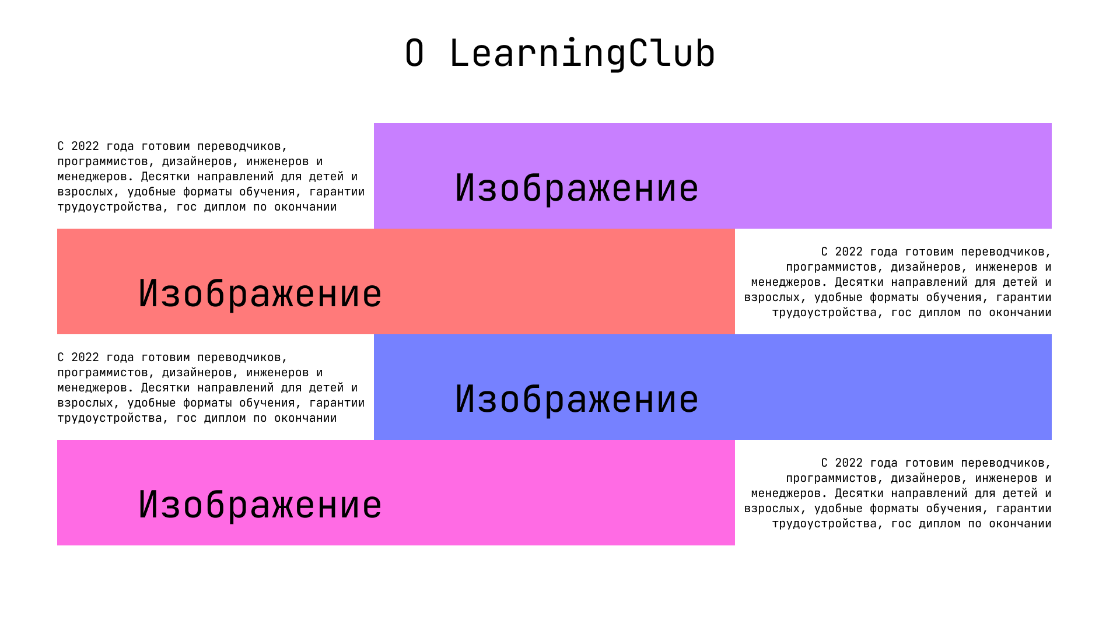


Рисунок 2.4.4 – Дизайн страницы «О нас»

Страница "Курсы" представляет собой последовательный список обучающих программ. Каждый курс сопровождается фоновым изображением, а при нажатии на него пользователь перенаправляется к форме подачи заявки на обучение. Дизайн сохраняет общий минималистичный стиль, избегая закруглений и сложных графических элементов.

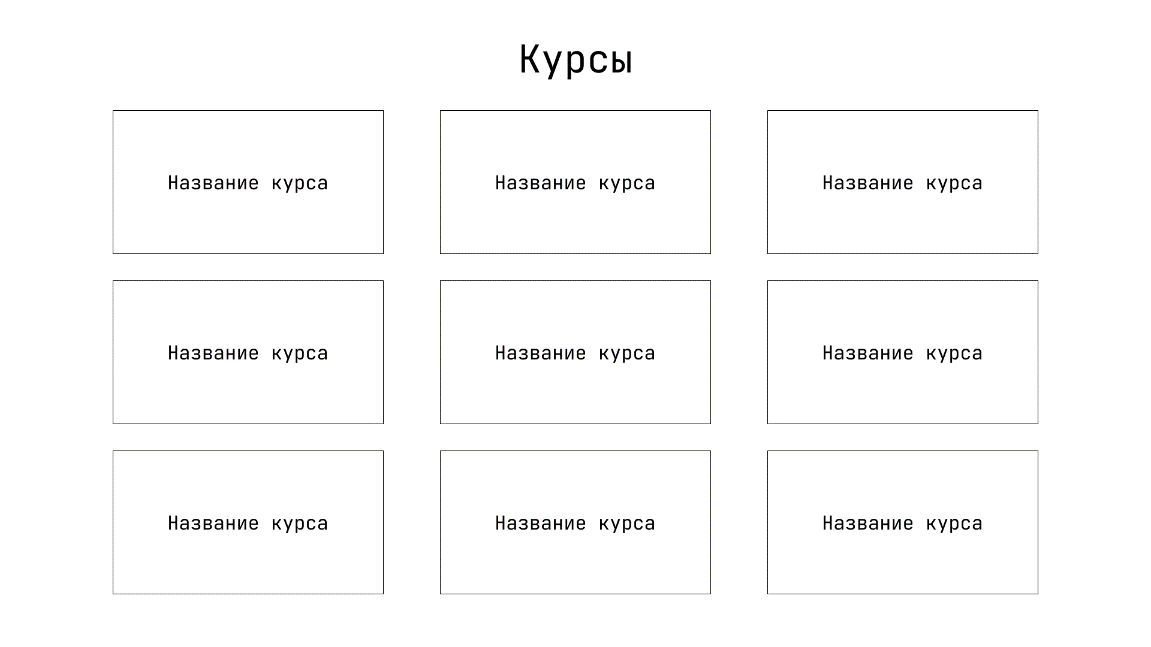


Рисунок 2.4.5 – Дизайн страницы «Курсы»

Страница "Вход" содержит простую форму авторизации с полями для логина и пароля. Дизайн формы выполнен в соответствии с общей стилистикой сайта, используя пастельный зеленый цвет для кнопки входа и стандартные цвета для фона и текста. Все элементы на странице размещены таким образом, чтобы обеспечить удобство и интуитивность взаимодействия.

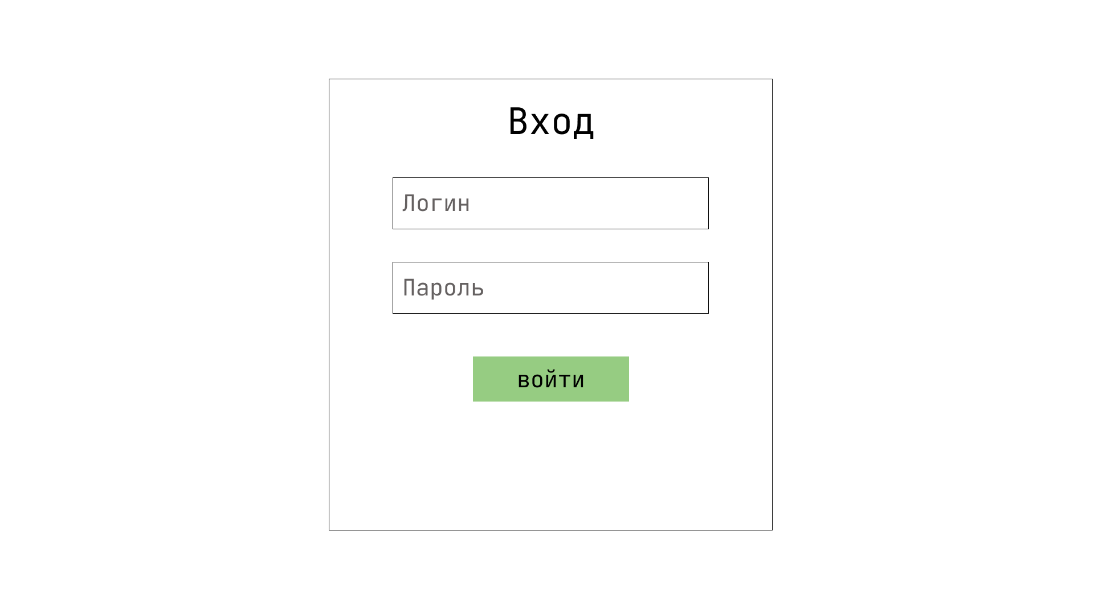


Рисунок 2.4.6 – Дизайн страницы «Войти»

Главная страница включает обобщенное для всех страниц содержимое, место под будущее промо-видео, продвигающее наиболее популярные курсы, а также форму подачи заявки.

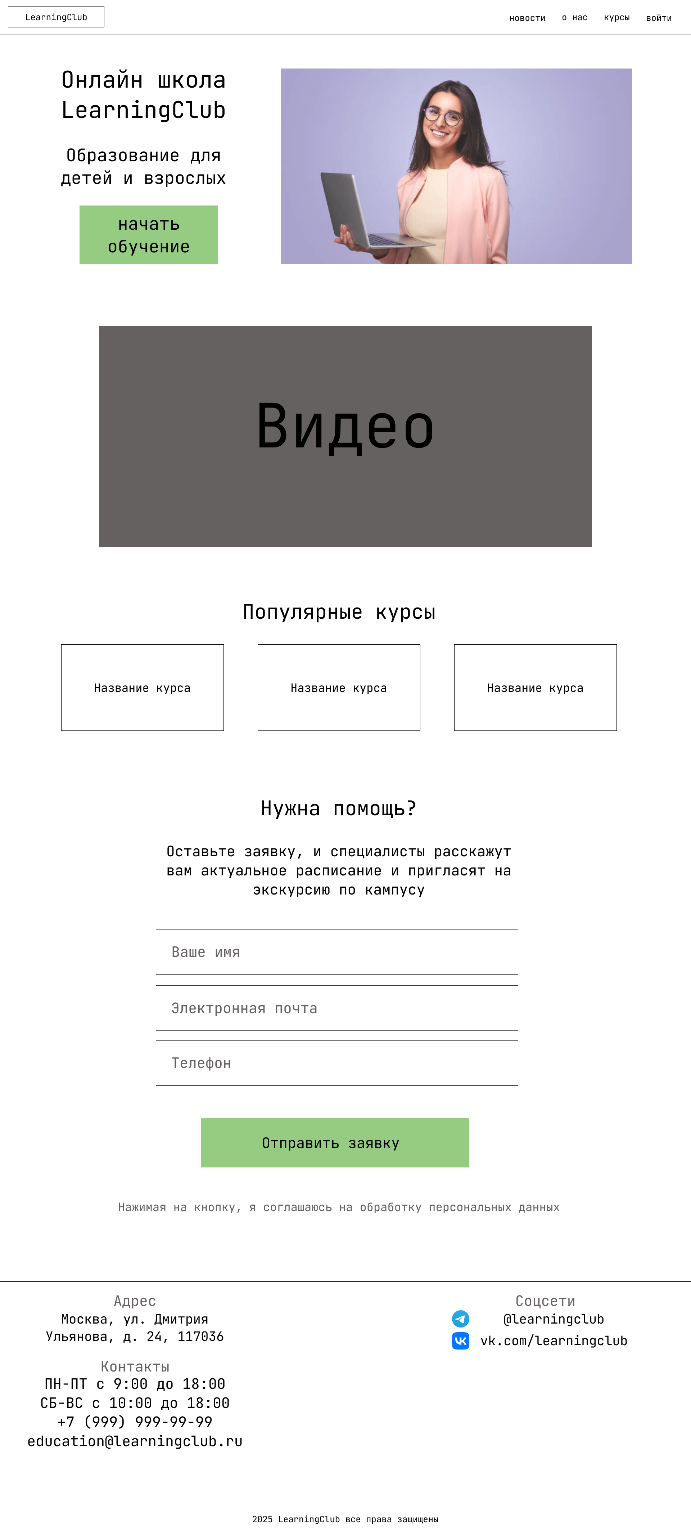


Рисунок 2.4.2 – Дизайн главной страницы посадочного модуля приложения

Таким образом, разработанный дизайн посадочной страницы сочетает в себе строгий минимализм, удобную структуру и гармоничное сочетание пастельных цветов. Выбранная цветовая палитра и четкая организация контента создают комфортное восприятие информации и упрощают навигацию по сайту.

# Разработка веб-приложения LearningClub

# Выбор технологий и инструментов разработки

При выборе стека технологий для серверной части веб-приложения необходимо рассмотреть несколько вариантов, ориентируясь на удобство разработки, производительность, безопасность и поддержку. Существуют различные технологии, которые могут быть применены для реализации бэкенда, и важно проанализировать их преимущества и недостатки в контексте требований к системе.

Одним из популярных стеков является Java Spring. Java широко используется в корпоративных приложениях благодаря своей надежности, высокой производительности и широким возможностям масштабирования. Spring предоставляет мощный фреймворк для создания веб-приложений с модульной архитектурой, встроенной поддержкой безопасности и удобным управлением зависимостями. Однако разработка на Java требует больше времени по сравнению с динамическими языками, а код получается более громоздким, что может усложнить поддержку проекта. Кроме того, настройка окружения и развертывание приложений на Java требуют дополнительных ресурсов, что может быть избыточным для относительно небольшого проекта.

Еще одним возможным вариантом является стек на базе C#. Технология .NET предоставляет мощный инструмент для разработки веб-приложений, а C# является строготипизированным языком с высокой производительностью. Использование ASP.NET Core позволяет создавать быстрые и масштабируемые серверные приложения, поддерживающие многопоточность и облачные решения.

Однако, несмотря на свои преимущества, .NET остается более сложным в освоении по сравнению с Python и требует использования инфраструктуры Microsoft, что может ограничивать гибкость разработки.

В последние годы набирает популярность Go. Этот язык программирования известен своей высокой производительностью, простотой кода и удобной поддержкой конкурентности. Благодаря встроенной системе управления потоками Go отлично подходит для высоконагруженных систем. Однако, несмотря на минималистичность языка, он требует явного управления обработкой данных, что может замедлить процесс разработки. В сравнении с Python, разработка на Go может занимать больше времени из-за отсутствия обширной экосистемы высокоуровневых инструментов.

В контексте текущего проекта наиболее подходящим выбором является использование Python и Django. Python известен своей простотой, гибкостью и богатой экосистемой библиотек, что ускоряет процесс разработки. Django, в свою очередь, предоставляет мощный инструмент для создания веб-приложений, включающий в себя встроенную ORM для работы с базами данных, систему аутентификации и административную панель. Использование Django позволяет значительно сократить время на разработку благодаря наличию готовых решений для большинства задач.

Дополнительным преимуществом Django является его встроенная административная панель. В рамках текущего проекта админка Django может быть полезна для управления пользователями, учебными группами и загруженными материалами. Это значительно упростит работу администраторов и преподавателей, обеспечивая удобный интерфейс для управления данными без необходимости разрабатывать отдельные панели управления.

Еще одной важной причиной выбора Python и Django является поддержка REST API с использованием Django REST Framework (DRF). Это позволяет легко интегрировать фронтенд с бэкендом, обеспечивая гибкость в выборе технологий для клиентской части приложения. Кроме того, использование Djoser в качестве механизма аутентификации на основе JWT обеспечит надежную и безопасную авторизацию пользователей.

Выбор SQLite в качестве базы данных обусловлен тем, что для данного проекта требуется легковесное решение, не требующее сложной настройки серверной инфраструктуры. SQLite подходит для небольших и средних проектов, обеспечивая достаточную производительность и удобство в развертывании. В будущем возможно использование более мощных решений, таких как PostgreSQL, если проект потребует масштабирования.

Таким образом, после анализа возможных технологий было принято решение использовать стек: Python, Django, DRF, Djoser, SQLite. Данный выбор обусловлен простотой разработки, наличием встроенных инструментов, удобством работы с базами данных и возможностью быстрой интеграции с фронтендом.

При этом стоит отметить, что фронтенд будет разрабатываться отдельно, поскольку разделение клиентской и серверной части повышает гибкость и универсальность системы. Один бэкенд может обслуживать разные интерфейсы, например, веб-приложение, мобильное приложение или даже сторонние сервисы через API. Это делает систему более масштабируемой и адаптивной к возможным будущим изменениям.

Кроме того, раздельная разработка упрощает поддержку и тестирование. Бэкенд реализует API, которое можно использовать независимо от конкретного интерфейса, что позволяет интегрировать систему с другими приложениями без необходимости вносить изменения в серверный код. Такой подход делает систему более устойчивой к изменениям и расширяет возможности её использования.

# Проектирование архитектуры приложения

Архитектура программного обеспечения определяет, каким образом структурируются компоненты системы, как они взаимодействуют между собой и какие принципы разработки используются для обеспечения удобства сопровождения и масштабируемости. Веб-приложения, особенно такие, которые предполагают взаимодействие пользователей с базой данных и сложную бизнес-логику, традиционно строятся по архитектурному шаблону MVC (Model-View-Controller).

Шаблон проектирования MVC (Model-View-Controller) представляет собой архитектурный подход, который разделяет код приложения на три логически независимых компонента: модель (Model), представление (View) и контроллер (Controller). Этот принцип применяется в разработке программного обеспечения для удобства сопровождения, тестирования и масштабирования, так как чёткое разделение обязанностей упрощает внесение изменений и поддержку кода. MVC используется во множестве веб-фреймворков, в том числе в Django, где он реализован в виде его собственной модификации MVT (Model-View-Template).

Модель (Model) отвечает за работу с данными, их хранение, обработку и бизнес-логику. В классическом MVC модель управляет данными в базе, выполняет валидацию, содержит алгоритмы обработки и обеспечивает целостность информации. Она инкапсулирует всю логику, связанную с обработкой данных, и не зависит от способа их отображения. Это даёт возможность изменять внутреннее представление информации без необходимости переделывать пользовательский интерфейс.

Представление (View) отвечает за отображение информации пользователю. Оно получает данные от модели и формирует пользовательский интерфейс в соответствии с заданным форматом. В веб-разработке это обычно HTML-страницы, шаблоны или API-ответы в JSON/XML-формате.

Представление не содержит бизнес-логики и работает исключительно с подготовленной информацией.

Это упрощает адаптацию приложения под различные интерфейсы, например, веб-страницу, мобильное приложение или API-клиент.

Контроллер (Controller) связывает модель и представление. Он получает запросы от пользователя, передаёт их в модель для обработки, а затем возвращает результат в представление. Контроллер управляет бизнес-логикой, но не занимается работой с данными напрямую – этим занимается модель. Такое разделение позволяет изменять логику обработки данных и взаимодействия с пользователем независимо друг от друга, что делает систему более гибкой.

Фреймворк Django использует модифицированную версию этой архитектуры под названием MVT (Model-View-Template). Главным отличием является отсутствие явного контроллера – его роль выполняют представления (views.py). В Django представления обрабатывают HTTP-запросы, взаимодействуют с моделями и формируют ответы. Модели (models.py) управляют структурой данных и взаимодействием с базой данных, а вместо традиционных представлений в виде кода во фреймворке используются шаблоны (templates), которые определяют, как будет выглядеть конечный интерфейс.

Применение этого шаблона проектирования в Django делает разработку более удобной и гибкой. Разделение кода по слоям позволяет поддерживать чистую архитектуру, снижать зависимость компонентов друг от друга и упрощает интеграцию с другими сервисами. В контексте разработки веб-приложения для образовательного учреждения, использование Django и его MVT-подхода позволит организовать удобное управление учебными процессами, интеграцию с различными клиентами (веб-приложение, мобильное приложение, API) и обеспечит безопасность работы с данными.Django также предусматривает разделение проекта на отдельные приложения, что является встроенным механизмом организации кода.

Каждое приложение представляет собой независимый модуль со своей логикой, моделями и маршрутизацией, что делает код более модульным и удобным для масштабирования. Это позволяет легко переиспользовать код и изменять отдельные части системы без затрагивания остального проекта.

Архитектура Django подразумевает, что каждое приложение содержит свою собственную структуру директорий, аналогичную общей структуре проекта. Внутри приложения обычно находятся:

* «models.py» – содержит описание моделей базы данных, специфичных для данного приложения.
* «views.py» – содержит обработчики HTTP-запросов.
* «serializers.py» – отвечает за преобразование данных в JSON-формат при использовании «Django REST Framework».
* «urls.py» – определяет маршруты, связанные с данным приложением.
* «services/» – папка для вынесения бизнес-логики, если она становится сложной.
* «tests.py» – содержит тесты, обеспечивающие стабильность работы приложения.

В разрабатываемом проекте будет использоваться одно основное приложение – «apiv1», которое будет предоставлять REST API для взаимодействия с клиентскими приложениями. Такое наименование связано с тем, что в будущем возможны версии API (например, «apiv2» с расширенными или изменёнными возможностями), и такой подход обеспечит удобство их поддержки.

Использование «Django REST Framework» (DRF) также вносит свои особенности в архитектуру.

В отличие от классических Django-приложений, где обработка данных ведётся через формы и шаблоны, DRF ориентирован на работу с API, предоставляя удобные инструменты для сериализации данных и их обработки, что важно для возможности интеграции с сервисом, используя различные платформы.

Основные особенности DRF, влияющие на нашу архитектуру:

* Использование сериализаторов – вместо обычных Django-форм мы используем serializers.py для конвертации данных из модели в JSON и обратно.
* ViewSets и Generic Views – позволяют значительно упростить обработку CRUD-операций, поскольку DRF предоставляет готовые классы для стандартных действий с данными.
* Маршрутизация через Router – DRF позволяет использовать routers.DefaultRouter() для автоматической генерации URL-адресов API.
* JWT-аутентификация через Djoser – вместо стандартной системы сессий мы используем Djoser, который предоставляет готовые эндпоинты для авторизации через JSON Web Token (JWT).

Таким образом, архитектура приложения сочетает в себе преимущества Django MVT и возможности DRF для удобного и гибкого управления API. Это позволит обеспечивать удобное взаимодействие с фронтендом, так как было принято решение о разделении фронтенда и бэкенда, что позволяет добиться большей гибкости, масштабируемости и удобства в разработке. Django используется исключительно для серверной части, а клиентская часть будет разрабатываться отдельно с использованием подходящих технологий.

Разделение фронтенда и бэкенда обеспечивает независимость разработки. Это означает, что серверное приложение можно разрабатывать, тестировать и развертывать отдельно от пользовательского интерфейса. Такой подход особенно полезен, если в будущем потребуется создать несколько различных клиентских приложений (например, веб-версию, мобильное приложение, интеграцию с внешними сервисами). Бэкенд на Django предоставляет API, к которому могут обращаться разные фронтенд-приложения, что делает систему более универсальной.

Кроме того, такое разделение позволяет использовать более современные и специализированные технологии для каждого слоя. Бэкенд отвечает за обработку данных, бизнес-логику, управление пользователями и безопасность.

А фронтенд фокусируется на удобстве взаимодействия с пользователем, отображении информации и динамических возможностях интерфейса. Это упрощает поддержку и модернизацию системы в будущем, поскольку изменения в одном из слоёв не требуют кардинального пересмотра другого.

В рамках нашего проекта бэкенд предоставляет REST API с использованием Django REST Framework. Это даёт возможность фронтенду запрашивать данные через HTTP-запросы, обрабатывать их и отображать в удобном формате. Такой подход позволяет легко интегрировать систему с другими сервисами и расширять её функционал без необходимости изменять серверную часть.

# Разработка структуры базы данных

Разработка структуры базы данных в Django основана на использовании моделей, которые определяют схему базы данных и управляются через ORM (Object-Relational Mapping). Django придерживается архитектурного паттерна MVT (Model-View-Template), где модели представляют собой слой данных, взаимодействующий с базой данных. В данном разделе мы подробно рассмотрим структуру базы данных, которая была спроектирована на основе нашей логической, инфологической и физической модели, а также её реализацию в Django.

Django ORM позволяет абстрагироваться от SQL-запросов и управлять данными с помощью Python-классов. В нашей системе будет несколько ключевых моделей: «User», «StudentsGroup», «CourseMaterial», «Homework», «HomeworkSubmission», «AttendanceRecord». Каждая из них выполняет свою роль в структуре базы данных, обеспечивая хранение и управление данными образовательного процесса.

Первой и самой важной является модель пользователя. Django уже предоставляет встроенную модель пользователя «AbstractUser», однако, для удобства мы её расширяем и добавляем проверки ролей.

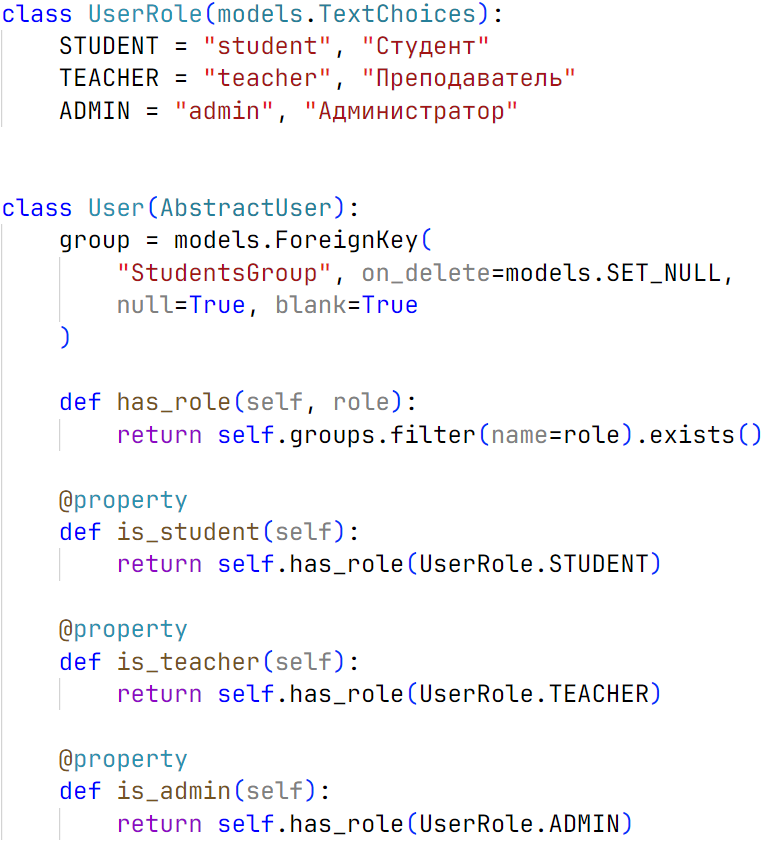


Рисунок 3.3.1 – Модель пользователя

Благодаря такому подходу, мы можем проверять роли пользователя без сложных запросов, используя удобные свойства «is\_student», «is\_teacher», «is\_admin».

Следующей важной моделью является группа студентов, которая в системе называется «StudentsGroup», чтобы избежать путаницы со стандартной моделью Django «Group».

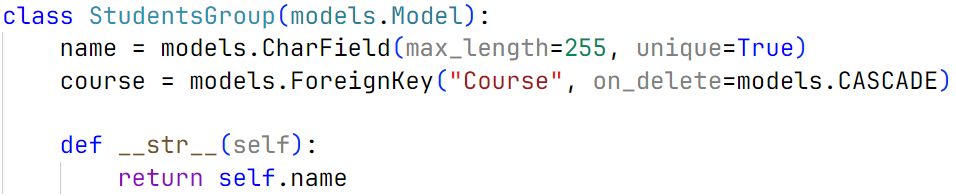


Рисунок 3.3.2 – Модель группы студентов

Каждый студент может принадлежать к одной группе, что упрощает управление учебным процессом.

Курс представлен моделью «Course», в которой определяется название и преподаватель.

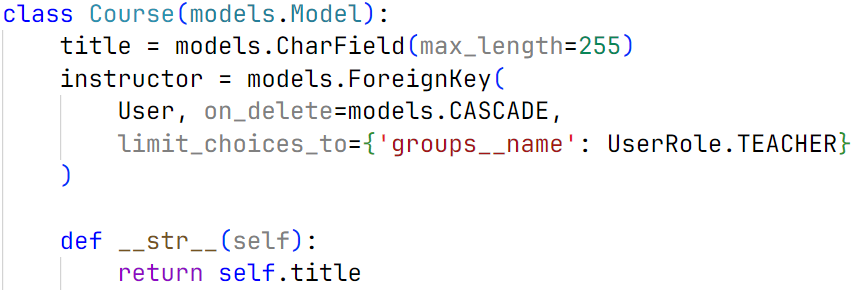


Рисунок 3.3.3 – Модель курса

Материалы курса представлены моделью «CourseMaterial», в которой преподаватель может загружать ссылки или файлы с материалами.

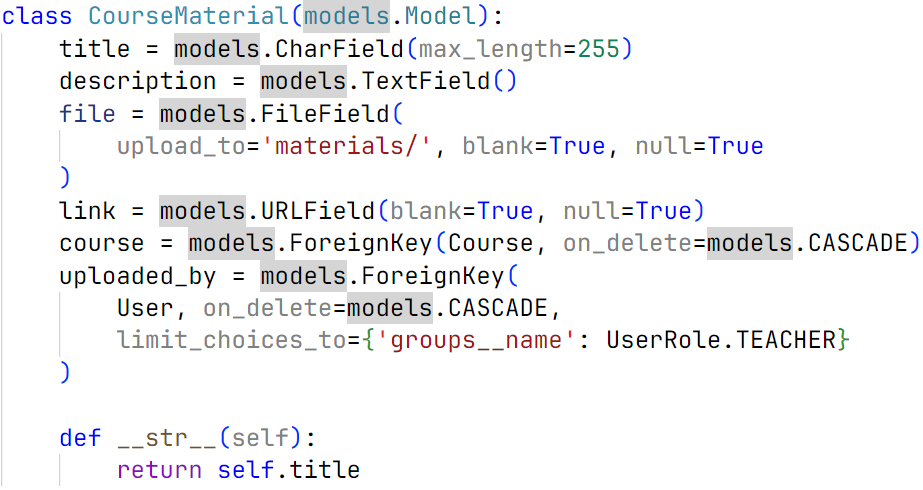


Рисунок 3.3.4 – Модель учебных материалов

Домашние задания хранятся в модели «Homework», которая связана с курсом и преподавателем.

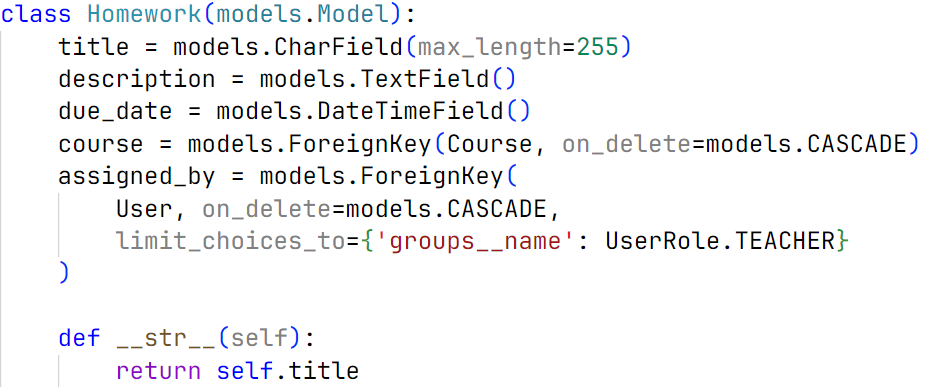


Рисунок 3.3.5 – Модель домашних заданий

Студенты могут сдавать решения домашних заданий, которые сохраняются в модели «HomeworkSubmission»:

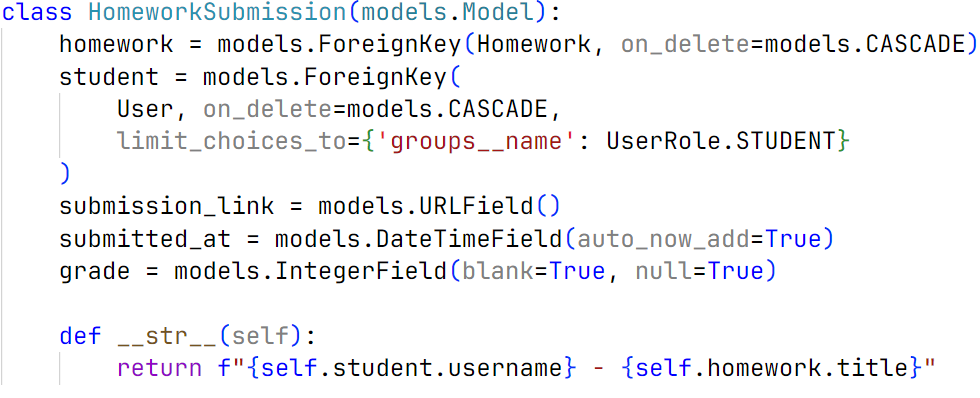


Рисунок 3.3.6 – Модель отправленных домашних работ

Посещаемость занятий учитывается в «AttendanceRecord», где фиксируется, присутствовал ли студент на занятии и его оценка (если применимо).

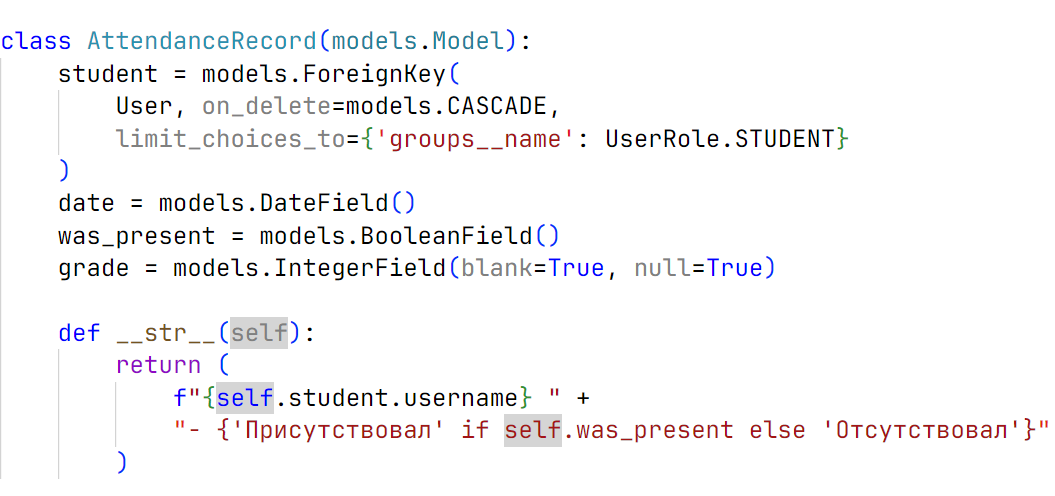


Рисунок 3.3.7 – Модель посещений занятий

Эти модели полностью отражают спроектированную базу данных и обеспечивают хранение всех необходимых данных для работы образовательной системы. Django ORM позволяет работать с ними как с обычными Python-объектами, упрощая взаимодействие с базой данных и обеспечивая высокую гибкость при разработке.

# Конфигурация серверного приложения

Настройка Django-проекта является важным этапом разработки, так как от корректной конфигурации зависит работа всей системы. Базовые настройки хранятся в файле settings.py, который содержит параметры для базы данных, аутентификации, middleware и других компонентов.

В первую очередь необходимо настроить базу данных. В данном проекте используется SQLite, так как это легковесное решение, не требующее дополнительной установки серверов баз данных.

# Заключение

.

# Список использованных источников

1. ГОСТ 7.32-2001. Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления. – Введ. 2002–07–01. – М.: Стандартинформ, 2006.–22 с.
2. Хамадеев Ш.А. Методология описания пользовательских требований. Учебно-методическое пособие по дисциплине «Проектирование АСОИУ». – Набережные Челны: ИПЦ НЧИ К(П)ФУ, 2017. – 28 с.
3. Сысолетин, Е. Г. Разработка интернет-приложений : учебное пособие для вузов / Е. Г. Сысолетин, С. Д. Ростунцев. — Москва : Издательство Юрайт, 2020. — 90 с. — (Высшее образование). — URL: <https://urait.ru/bcode/453345>.
4. Жемчужников Д. Г. Web-дизайн. Уровень 1. — М.: Просвещение-Союз, 2023. — 144 c.
5. Жемчужников Д. Г. Web-дизайн. Уровень 2. — М.: Просвещение-Союз, 2023. — 144 c.
6. Гома, Хассан UML. Проектирование систем реального времени, параллельных и распределенных приложений / Хассан Гома. - М.: ДМК Пресс, 2017. - 700 c.
7. Гарретт Джесс. Веб-дизайн. Элементы опыта взаимодействия. — М.: Символ-Плюс, 2020. — 285 c.
8. Полуэктова, Н. Р. Разработка веб-приложений : учебное пособие для вузов / Н. Р. Полуэктова. — М.: Юрайт, 2021. — 204 с.
9. Лоусон, Гарольд Путешествие по системному ландшафту: моногр. / Гарольд Лоусон. - М.: ДМК Пресс, 2017. - 368 c.
10. Сырых Ю. А. Современный веб-дизайн. Настольный и мобильный. — М.: Диалектика, 2019. — 384 c.
11. Григорьев, М. В. Проектирование информационных систем : учебное пособие для вузов / М. В. Григорьев, И. И. Григорьева. — М.: Юрайт, 2024. — 278 с.
12. Полуэткова Н.Р. Разработка веб-приложений. –- М.: Юрайт, 2024. –- 205 с.
13. Данелян, Т. Я., Информационные системы и информационные технологии в бизнес-процессах : учебно-практическое пособие / Т. Я. Данелян, И. А. Бакай. — Москва : Русайнс, 2021. — 179 с.
14. Круг, С. Веб-Дизайн: книга Стива Круга или "не заставляйте меня думать!" / С. Круг. - М.: Символ-Плюс, 2008. - 224 c.
15. Нагаева И. А., Фролов А. Б., Кузнецов И. А. Основы web-дизайна. Методика проектирования. — М.: Директ-Медиа, 2021. — 184 c.