**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА и**

**ГОСУДАРСТВЕННОЙ СЛУЖБЫ**

**при Президенте Российской Федерации»**

**КОЛЛЕДЖ МНОГОУРОВНЕВОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**Специальность 09.02.07. «Информационные системы и программирование»**

УТВЕРЖДАЮ

Зам. директора КМПО РАНХиГС

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С.Ф. Гасанов

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

**ДИПЛОМНЫЙ ПРОЕКТ**

**на тему:**

**«**Разработка веб-приложения для автоматизации процессов информирования персонала в компании ООО "ТДМ"**»**

|  |  |  |
| --- | --- | --- |
| Выполнил студент группы 41ИС-21 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | /Н.А. Голубятников/ |
| Руководитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | /Ф.В. Кусков/ |
| Нормоконтролер | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | /Ф.В. Кусков/ |
| Консультант по технико-экономическому обоснованию проекта | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | /М.М. Трифонова/ |
| Старший консультант | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | /О.А Калашникова/ |

**МОСКВА**

**2025 г.**

**Введение**

В современных условиях активного внедрения информационных технологий в деятельность организаций повышается важность эффективного управления внутренними коммуникациями. Актуальность темы дипломного проекта обусловлена потребностью компании ООО «ТДМ» в создании единой и надежной системы информирования персонала, которая бы позволила обеспечить оперативность, структурированность и доступность внутренней корпоративной информации.

Компания ООО «ТДМ» специализируется на поставках товаров для строительства и благоустройства, производстве и монтаже металлических конструкций и малых архитектурных форм. Ввиду специфики деятельности, значительная часть сотрудников компании работает вне офисных помещений, что предъявляет дополнительные требования к процессам коммуникации и своевременному информированию.

Существующая практика информирования сотрудников через электронную почту и мессенджеры выявила существенные недостатки: отсутствие гарантии своевременного ознакомления с информацией, сложности в поиске ранее опубликованных сообщений и отсутствие единой базы для хранения документов и приказов. Данные факторы ведут к рискам неисполнения важных распоряжений, снижения оперативности реагирования сотрудников на изменения и, в конечном итоге, к снижению общей продуктивности компании.

В связи с этим возникла необходимость разработки специализированного веб-приложения, интегрирующего различные функции автоматизации внутренних коммуникаций. Данное веб-приложение спроектировано на современной технологической платформе, использующей базу данных PostgreSQL, что обеспечивает высокую производительность, надежность хранения данных и гибкость при масштабировании и интеграциях. Размещение приложения осуществляется на платформе Render, обладающей возможностью быстрого развертывания приложений, поддержкой современных технологий и высокой доступностью.

Цель дипломного проекта — создание эффективного инструмента, обеспечивающего оперативное доведение до сотрудников компании ООО «ТДМ» корпоративных новостей, приказов, изменений в регламентах, задач и дедлайнов. Веб-приложение также предусматривает механизмы взаимодействия и обратной связи, модуль тестирования и обучения сотрудников, а также администрирование с дифференцированными правами доступа.

Для достижения цели были решены следующие задачи: проведен анализ существующих решений, спроектирована архитектура системы с REST API и PostgreSQL, реализованы серверная и клиентская части веб-приложения, выполнено функциональное тестирование и подготовлено технико-экономическое обоснование.

Предлагаемое решение позволит значительно повысить качество и оперативность внутренних коммуникаций компании, минимизировать риски потери важной информации и улучшить вовлеченность персонала в рабочий процесс. Реализация данного веб-приложения на платформе Render с использованием базы данных PostgreSQL демонстрирует оптимальное сочетание современных технологических решений и практической эффективности, адаптированных под конкретные требования ООО «ТДМ».

Добавить актуальность, объект исследования, предмет исследования. Задачи оформить маркированным списком.

Оформление дипломного проекта править по методическим рекомендациям.

**1 Описание предметной области**

В современных организациях налаженные внутренние коммуникации являются важнейшим фактором стабильной и слаженной работы коллектива. Под **внутренним информированием персонала** понимается процесс доведения официальной и оперативной информации от руководства или администрации до сотрудников компании. Такая информация включает корпоративные новости, приказы и распоряжения, изменения во внутренних регламентах, сведения о текущих рабочих задачах, планируемых мероприятиях и установленных дедлайнах. Своевременное получение этих данных позволяет каждому сотруднику ориентироваться в деятельности организации, понимать актуальные приоритеты и требования. В итоге это повышает эффективность труда и уровень вовлечённости персонала в достижение целей предприятия.

**Актуальность автоматизации** процессов информирования персонала обусловлена ростом объёмов информации и потребностью в её быстрой передаче внутри организации. В эпоху цифровизации всё больше компаний стремятся уйти от разрозненных каналов общения (таких как рассылка писем по электронной почте, бумажные объявления на стендах, устные совещания) к единому информационному пространству, доступному каждому сотруднику. Разработка специализированного программного обеспечения – например, корпоративного веб-приложения – позволяет объединить различные каналы коммуникации на одной платформе и обеспечить оперативное доведение сообщений до целевой аудитории внутри организации.

В качестве практического примера рассматривается компания **ООО «ТДМ»** (Торговый Дом «Магистраль», г. Москва, м. Алексеевская). Эта организация с 2017 года специализируется на поставках товаров для строительства и благоустройства, в том числе на производстве и монтаже металлических конструкций и малых архитектурных форм (например, уличных скамеек, ограждений, игровых комплексов). Деятельность компании связана как с логистикой строительных материалов, так и с собственным производством, поэтому в «ТДМ» трудятся специалисты различных профилей: менеджеры по логистике и снабжению, складские работники, инженеры-проектировщики, производственный персонал, отдел продаж и административный персонал. География проектов охватывает разные регионы, часть сотрудников часто работает вне офиса (на выездах, объектах), что предъявляет высокие требования к эффективности внутренних коммуникаций.

На практике компания **ООО «ТДМ»** сталкивается с типичными проблемами доведения информации до персонала. В настоящее время оповещение сотрудников осуществляется преимущественно с помощью электронных писем и сообщений в мессенджерах. Руководство рассылает письма с приказами и объявлениями на корпоративные email-адреса, а для оперативных вопросов нередко используются групповые чаты (например, в WhatsApp или Telegram). Однако такие разрозненные методы не гарантируют, что каждый работник своевременно получит и прочитает важное сообщение. Информация может затеряться в почтовом ящике среди других писем или остаться незамеченной в общем чате. Кроме того, у части персонала (например, у работников складов или монтажных бригад) нет постоянного доступа к компьютеру в течение дня, из-за чего оперативность доведения сведений снижается. Отсутствие единой системы не позволяет централизованно хранить все корпоративные объявления, затрудняет поиск старых распоряжений и **не даёт возможности проконтролировать факт ознакомления** сотрудников с той или иной информацией.

Подобные недостатки в организации внутреннего информирования ведут к рискам: важные распоряжения могут быть выполнены несвоевременно, сотрудники – не понять новых требований, а общая продуктивность снижается. **Следовательно, для компании «ТДМ» назрела необходимость внедрения единой информационной системы**, которая автоматизирует процесс информирования персонала, делая его более структурированным, надёжным и оперативным. Эта система должна обеспечить доведение до каждого сотрудника всех актуальных сообщений от руководства и позволять контролировать, что информация действительно получена и прочитана адресатами. Ниже проводится анализ существующих решений для подобных задач и формулируются требования к разрабатываемому корпоративному веб-приложению. Выделенного текста быть не должно

**1.1 Анализ существующих решений**

На сегодняшний день существует множество программных решений, предназначенных для внутренних коммуникаций и информирования сотрудников в организациях различного профиля – от коммерческих предприятий до государственных учреждений и вузов. Эти решения варьируются по функционалу и подходам к организации информационного пространства, но цель у них общая: обеспечить эффективный обмен информацией между руководством и коллективом. Рассмотрим наиболее распространённые типы подобных систем и инструментов, применяемых на практике для внутреннего информирования персонала.

**Корпоративные порталы (intranet-системы).** Один из популярных подходов к информированию сотрудников – внедрение корпоративного портала, то есть внутреннего веб-сайта организации, доступного только работникам. Корпоративный портал обычно представляет собой многофункциональную платформу, объединяющую в едином интерфейсе новости компании, базы документов, объявления, календари событий и другие сервисы. Примерами таких систем служат продукты класса intranet: например, платформа *1С-Битрикс24*, Microsoft SharePoint и аналогичные решения. Bitrix24 широко используется в российских организациях в качестве внутреннего портала: он позволяет публиковать новости и приказы, вести обсуждения в формате социальной сети (лента активности), хранить и распределять документы, ставить задачи и контролировать их выполнение, а также общаться через встроенные чаты. Подобные системы создают единое информационное пространство предприятия, где каждый сотрудник в любой момент имеет доступ к актуальным объявлениям, документам и ресурсам. **Преимущество портальных решений** – структурированность и полнота информации: данные хранятся централизованно, с возможностью дифференцировать доступ (например, некоторые разделы только для руководства, для отдела логистики, для отдела снабжения и т.д.). Крупные компании нередко внедряют портал как *стандарт внутреннего информирования* и интегрируют в него множество функций (от кадровых сервисов до обучения).

В контексте деятельности ООО «ТДМ» корпоративный портал мог бы предоставить сотрудникам централизованный доступ ко всем распоряжениям и данным компании: приказам руководства, планам поставок, графикам работ, техническим инструкциям и т.п. Это исключило бы ситуацию, когда сведения рассылаются по разным каналам, и повысило бы прозрачность – каждый сотрудник знал бы, где искать необходимые внутренние документы. **Однако внедрение готового портала** зачастую требует существенных ресурсов на настройку и сопровождение. Универсальные продукты (как Bitrix24 или SharePoint) содержат избыточный функционал, который может оказаться невостребованным в конкретной компании, усложняя интерфейс. В то же время каких-то узких возможностей, важных для ООО «ТДМ», может не оказаться «из коробки». Это делает корпоративный портал мощным, но не всегда оптимальным решением без адаптации под нужды организации.

**Корпоративные мессенджеры и чаты.** Другой распространённый подход к внутренним коммуникациям – использование специализированных мессенджеров или групповых чатов для оперативного обмена сообщениями. Речь идёт о таких платформах, как *Slack* и *Microsoft Teams*, а также их аналоги (Cisco Webex Teams, Mattermost, ~~Workplace by Facebook~~ и др.). Многие компании, особенно с молодым коллективом или распределённой структурой, делают ставку на такие средства для повседневного общения.

Так, Slack изначально создан как корпоративный чат для команд: он позволяет организовывать общение по тематическим каналам, быстро обмениваться файлами, интегрируется со множеством сторонних сервисов (системы управления проектами, хранилища документов и др.). В контексте информирования персонала Slack или его аналоги дают возможность создать специальный канал для объявлений (например, канал #news, куда могут писать только руководители или модераторы). Сотрудники, подписанные на такой канал, мгновенно получают уведомление о новом сообщении – это обеспечивает очень быструю доставку информации.

Microsoft Teams, входящий в экосистему Office 365, сочетает функции чата, видеоконференций, хранилища файлов и совместной работы над документами. Многие крупные предприятия внедрили Teams в качестве основного инструмента коммуникации, особенно в период перехода на дистанционную работу. В **Microsoft Teams** также можно создать канал для объявлений, проводить онлайн-совещания, рассылать материалы и ссылки. Преимущества корпоративных мессенджеров очевидны – высокая скорость обмена информацией и удобство для дискуссий: сотрудники могут не только получать сообщения, но и сразу же обсуждать их, задавать уточняющие вопросы. Кроме того, современные мессенджеры обычно предлагают приложения для смартфонов, то есть работник получит оповещение практически где угодно и когда угодно.

Однако у этого класса решений есть и недостатки. Информация в чатах быстро «тонет» в потоке сообщений, отсутствует строгая структура данных. Если сотрудник не прочитал сообщение сразу, позже ему придётся пролистывать историю переписки. В отличие от портала, где новости чётко структурированы по разделам и хранятся постоянно, в мессенджере сложнее организовать единый архив объявлений или гарантировать ознакомление **каждого** сотрудника с важным уведомлением. Нередко возникают ситуации, когда часть коллектива попросту пропустила объявление в общем чате. Также формат мессенджера не всегда подходит для официальных сообщений – например, большой приказ с формальным языком и вложениями удобнее разместить на портале или хотя бы выслать отдельным письмом, чем публиковать в чат-группе. Таким образом, мессенджеры хорошо решают задачу оперативного оповещения и обсуждения, но проигрывают в вопросах структурированности и контроля ознакомления.

**Традиционные средства оповещения** – электронная почта и информационные доски – до сих пор широко используются там, где отсутствуют специальные платформы. **Email-рассылки** по корпоративной почте долгое время были основным способом донесения информации до сотрудников. Преимущество электронной почты в её универсальности: почти у каждого работника есть почтовый ящик, уведомление о новом письме приходит автоматически. Руководство может разослать на все адреса письмо с приказом или новостью, прикрепив необходимые документы. В образовательных учреждениях, к примеру, рассылка приказов ректором на кафедры по email до сих пор является нормой. Для ООО «ТДМ», как уже отмечалось, email – тоже базовый канал передачи объявлений на сегодняшний день.

Недостатки подхода с электронными письмами проявляются в крупных и динамичных организациях. Нет гарантии, что письмо прочитают вовремя: сотрудник может не проверить почту в течение дня или пропустить письмо среди десятков других сообщений. Обратная связь и обсуждение по email затруднены – групповая переписка быстро превращается в хаос из длинных reply-цепочек. К тому же поиск старых писем с нужной информацией может быть непростым, если тема письма не очевидна, а почтовый архив обширный. **Информационные доски объявлений** (стенды) в офисах и на производстве – это фактически аналог портала в доперсональный период: распечатанные приказы и графики вывешиваются в общем помещении. В современных условиях физические стенды зачастую заменяются электронными – разделами новостей на корпоративном сайте, однако принцип остаётся тем же: сотрудник должен сам найти время зайти и ознакомиться. Это пассивный способ информирования, который не обеспечивает оперативности и охвата (особенно удалённых работников). На практике организации часто комбинируют несколько методов: например, важное объявление могут одновременно выслать письмом, продублировать на портале и даже отправить в чат, – стремясь таким образом повысить вероятность, что все адресаты увидят сообщение.

**Специализированные узкопрофильные системы.** Помимо универсальных инструментов, существуют решения, нацеленные на отдельные аспекты внутренних коммуникаций. К примеру, системы управления обучением (LMS – *Learning Management Systems*) применяются в крупных корпорациях для распространения учебных материалов и тестирования сотрудников. Платформы вроде Moodle популярны в вузах, а в бизнес-среде им соответствуют корпоративные обучающие порталы. Они решают задачу доведения информации, но только в рамках обучения – предоставляя доступ к курсам, инструкциям и проверяя знания. Другой пример – системы экстренного оповещения, обеспечивающие мгновенную рассылку сообщений всем сотрудникам через SMS, звонки или push-уведомления при наступлении критических ситуаций (тревоги, аварии, чрезвычайные события). Такие системы используются, например, для уведомления персонала о пожарной тревоге или сбое IT-инфраструктуры и выходят за рамки повседневных коммуникаций.

Каждое из рассмотренных решений обладает своими плюсами и минусами. **Выбор оптимального инструмента информирования** зависит от масштаба организации, структуры, корпоративной культуры и технических возможностей. Крупное предприятие с тысячами сотрудников скорее внедрит мощный портал с интеграцией многих сервисов, тогда как небольшая фирма может ограничиться группой в мессенджере или периодическими рассылками. При этом ни одно готовое решение не бывает идеально приспособленным под уникальные потребности конкретной компании. Использование лишь мессенджера без портала затрудняет хранение официальных документов и последующий поиск информации. Типовой корпоративный портал может оказаться перегруженным функциями, не востребованными в ООО «ТДМ», и в то же время у него может отсутствовать функционал, специфичный для деятельности этой компании (скажем, учёт ознакомления сотрудников с приказами). Поэтому многие организации приходят к идее разработки **собственного программного решения** или к глубокой доработке существующих платформ под свои нужды.

Анализ существующих приложений для внутреннего информирования показывает, что идеальной выглядит система, сочетающая преимущества разных подходов. Такая система должна обеспечивать структурированность и официальность, как портал; интерактивность и скорость, как мессенджеры; универсальность и привычность оповещений, как email; и при этом иметь механизмы контроля ознакомления персонала с сообщениями, чего не дают стандартные средства. В случае компании **ООО «ТДМ»** целесообразно учесть опыт как коммерческих предприятий, так и образовательных организаций: перенять лучшие практики (например, наличие удобной ленты новостей, системы обратной связи, постановки задач) и избежать выявленных недостатков (таких как избыточная сложность интерфейса или отсутствие контроля прочтения). Все эти выводы будут положены в основу требований к собственному веб-приложению, которое предстоит разработать специально для автоматизации внутренних коммуникаций в данной организации.

**1.2 Основные функции веб-приложения**

С учётом анализа предметной области и существующих решений можно определить **требования к функциональности**, которую должно предоставлять разрабатываемое веб-приложение для информирования персонала компании ООО «ТДМ». Проектируемое приложение нацелено на создание единой информационной платформы, поэтому его функционал охватывает несколько взаимосвязанных модулей: информационный (новости и объявления), коммуникационный (уведомления и обратная связь), организационный (постановка задач и планирование событий) и обучающий. Ниже описаны ключевые функции системы, отражающие потребности компании во внутреннем портале.

**Лента корпоративных объявлений и новостей.** Веб-приложение обеспечивает централизованное распространение официальной информации по компании через специальную ленту объявлений (раздел новостей). Уполномоченные сотрудники – например, администратор системы, сотрудники отдела кадров или руководство организации – публикуют в этой ленте корпоративные новости, приказы, распоряжения и другие объявления. Каждая публикация содержит заголовок, текст сообщения (при необходимости – вложения: документы, изображения), а также дату и автора. Желательно реализовать классификацию новостных сообщений по категориям или темам: например, «Новости компании», «Приказы и распоряжения», «Охрана труда», «Обучающие материалы» и т.п. Это позволит сотрудникам фильтровать объявления или просматривать только интересующие их разделы. При появлении новой записи в ленте система должна автоматически формировать уведомление для работников (будь то по электронной почте, push-оповещение или внутреннее уведомление в интерфейсе – механизмы уведомлений описаны ниже), чтобы ни одно объявление не осталось незамеченным. Таким образом, лента новостей будет выполнять роль электронной «доски объявлений», доступной 24/7: каждый сотрудник в любой момент может открыть веб-приложение и ознакомиться со всеми актуальными на данный момент сообщениями. Помимо текущих новостей, должен быть реализован удобный доступ к **архиву объявлений** – то есть к более старым записям, хранящимся в системе. Поиск по ключевым словам и фильтрация по дате или категории помогут быстро находить ранее опубликованную информацию (например, приказ за прошлый год или объявление о прошедшем событии). Наличие полного архива новостей обеспечивает накопление и сохранность корпоративных знаний: даже новые сотрудники, подключившись к системе, смогут изучить хронику жизни компании за интересующий период.

Нет рисунка

*Рисунок 1.1 – Диаграмма IDEF0 процесса внутреннего информирования сотрудников в компании ООО «ТДМ». Входом процесса служит информация от руководства (приказы, новости, задания), на выходе – осведомлённые сотрудники. В качестве механизма используется корпоративное веб-приложение (IT-инфраструктура компании), а в роли управляющих воздействий выступают внутренние политики и регламенты организации.*

**Модуль управления задачами и мероприятиями.** Помимо информирования в форме объявлений (когда сотрудник просто читает новости), в компании «ТДМ» существует потребность доводить до персонала и конкретные поручения, а также планировать совместные события. Поэтому разрабатываемое веб-приложение включает функционал управления задачами и календарём мероприятий. Уполномоченный пользователь (например, руководитель отдела или проектный менеджер) через интерфейс системы сможет поставить задачу конкретному сотруднику или группе сотрудников. Постановка задачи включает описание цели, назначение ответственного исполнителя (или команды), указание срока выполнения (дедлайна) и при необходимости прикрепление сопутствующих материалов (техническое задание, документы). Назначенному исполнителю поступает уведомление о новой задаче, после чего он может в личном кабинете приложения просмотреть детали и по готовности отметить выполнение. Все активные задачи отображаются в системе в виде списка (или на канбан-доске), доступного ответственным руководителям для контроля. Система также будет присылать напоминания о приближении сроков исполнения задач, что дисциплинирует исполнителей и снижает вероятность просрочки поручений.

Аналогично, модуль событий позволяет анонсировать предстоящие мероприятия: собрания, производственные совещания, тренинги, корпоративные праздники и т.д. При создании события указывается его название, дата и время, место проведения (или ссылка для онлайн-конференции, если мероприятие проходит дистанционно) и круг приглашённых участников. Приглашения на событие рассылаются автоматически через систему, а сотрудники могут подтвердить своё участие либо отказ. Информация о мероприятиях также отображается в общем календаре внутри приложения – это помогает персоналу планировать своё время и не пропускать важные встречи. Отметим, что модуль задач и календарь мероприятий служат для **активного информирования** сотрудников о том, *что именно нужно сделать и к какому сроку*, дополняя новостную ленту (которая отвечает скорее на вопрос *что происходит* и *что нового* в компании).

**Модуль обучения и знаний (внутренний портал для инструктажей).** В рамках развития персонала и доведения до сотрудников знаний по новым правилам или технологиям система предусматривает обучающий функционал. Этот модуль позволит размещать во внутреннем портале обучающие материалы: методические пособия, инструкции, регламенты, презентации, видеоролики и т.д., делая их доступными для самостоятельного изучения сотрудниками. Например, если в ООО «ТДМ» вводится новый стандарт по технике безопасности или осваивается новое программное обеспечение, соответствующие инструкции и пособия можно разместить во внутренней системе, чтобы каждый сотрудник мог вовремя ознакомиться с ними. Однако одной публикации материалов недостаточно – важно проверить, усвоена ли информация. Поэтому ключевой возможностью обучающего модуля станет **онлайн-тестирование** сотрудников. Администратор или методист компании получит инструмент для создания в системе тестов (наборов вопросов с вариантами ответов или с открытыми ответами) по тем или иным темам. После изучения материалов сотрудники проходят тестирование прямо в веб-приложении, а результаты фиксируются. Руководство сможет проконтролировать, все ли сотрудники прошли обязательный инструктаж и насколько успешно (например, увидеть процент правильных ответов каждого). Внедрение тестов стимулирует персонал внимательнее относиться к изучению важных документов, а также позволяет формально зафиксировать **факт ознакомления** – что нередко требуется по требованиям трудового законодательства и стандартов ISO (например, подтверждение прохождения вводного инструктажа). Реализуя модуль обучения, система не только информирует, но и контролирует степень усвоения информации сотрудниками, повышая их общую квалификацию и информированность.

**Средства обратной связи и опросы.** Эффективное информирование подразумевает не только передачу информации «сверху вниз», но и получение обратной связи от сотрудников. В веб-приложении предусмотрена функциональность для сбора мнений, предложений и обратной реакции персонала. Во-первых, под каждым объявлением или новостью может быть включена возможность комментариев. Это позволит сотрудникам задавать уточняющие вопросы прямо в системе или выражать своё мнение по поводу прочитанного (с соблюдением деловой этики и с модерацией со стороны администратора). Комментарии сделают коммуникацию двусторонней: руководство увидит, насколько понятна доведённая информация, возникли ли у коллектива вопросы, и сможет оперативно на них ответить.

Во-вторых, планируется реализовать модуль **опросов и анкетирования**. Администрация компании «ТДМ» сможет проводить внутренние опросы – например, чтобы узнать мнение сотрудников о каких-либо инициативах, собрать идеи по улучшению рабочих условий, оценить удовлетворённость персонала и т.п. Система позволит сформулировать вопросы (в виде анкеты или теста), разослать приглашение к участию всем либо выборочным сотрудникам, а затем автоматически собрать и обработать результаты. Результаты опроса будут доступны уполномоченным лицам в виде агрегированной статистики (диаграммы, процентные показатели) для дальнейшего анализа. Наличие встроенных средств опроса исключает необходимость в сторонних сервисах анкетирования и повышает откровенность ответов, так как сотрудники видят, что опрос внутренний и конфиденциальный. Кроме того, обратную связь можно получать и через специальную форму обращений: например, раздел «Задать вопрос руководству» или «Сообщить о проблеме». Такие обращения будут поступать ответственным лицам (в отдел кадров или непосредственно руководству) через систему, создавая ещё один канал коммуникации. В совокупности инструменты обратной связи повысят степень участия сотрудников в жизни организации и дадут руководству ценную информацию о настроениях и потребностях коллектива.

**Администрирование, права доступа и интеграции.** Для поддержания всех перечисленных функций необходим продуманный административный модуль. Веб-приложение предусматривает наличие панели администратора, через которую осуществляется настройка системы и управление контентом. **Администратор системы** (или группа администраторов) сможет создавать учётные записи пользователей, назначать им роли и права доступа. Предусмотрены различные роли пользователей с разными полномочиями. Например:

* *Рядовой сотрудник* – имеет право только просматривать опубликованные объявления, проходить тесты и участвовать в опросах.
* *Менеджер/руководитель подразделения* – помимо прав сотрудника, может публиковать новости в рамках своего отдела, ставить задачи подчинённым и просматривать отчёты.
* *Контент-менеджер (новостной администратор)* – пользователь, ответственный за информационное наполнение: публикует новости от лица компании, рассылает важные объявления, модерирует комментарии.
* *Системный администратор* – обладает полным доступом, управляет всеми разделами и учетными записями, назначает роли другим пользователям, контролирует работу системы в целом.

Разграничение прав доступа обеспечивает безопасность и актуальность данных: важные объявления не смогут публиковать неуполномоченные лица, а конфиденциальная информация (если таковая появится на портале) будет видна только тем, кому положено. Административная панель также позволит модерировать пользовательский контент (например, удалять неподобающие комментарии), управлять справочниками системы – категориями новостей, темами тестов, списками опросов и т.д.

Неотъемлемой частью системы станут **механизмы уведомлений**. При любом значимом событии (публикация нового объявления, постановка задачи, приглашение на мероприятие, добавление комментария, получение результатов тестирования) система должна оперативно уведомить соответствующих пользователей. Для этого планируется интеграция с почтовым сервером компании, чтобы дублировать уведомления на корпоративный email каждого сотрудника. Например, сразу после публикации нового приказа на портале все сотрудники получат письмо-оповещение с кратким содержанием и ссылкой на полный текст. Такой подход сочетает плюсы портала (централизованное хранение информации) и электронной почты (push-уведомление, привлекающее внимание). Помимо email, при развитии системы можно предусмотреть и другие каналы оповещения: SMS-сообщения для экстренных случаев, всплывающие push-уведомления в браузере или мобильном приложении, интеграцию с корпоративным мессенджером (если такой появится в «ТДМ»).

Гибкость системы в плане интеграций – важное требование. В задании на проект указана необходимость адаптации решения под инфраструктуру компании. Это может означать сопряжение веб-приложения с уже используемыми в «ТДМ» информационными системами. Например, интеграция с внутренней базой данных отдела кадров позволит автоматически синхронизировать список сотрудников и их отделы – чтобы не заводить всех пользователей вручную и сразу выдавать им права согласно должностям. Интеграция с календарём Outlook или Google Calendar могла бы автоматически добавлять созданные в системе мероприятия в личные календари сотрудников. Хотя подобные глубокие интеграции выходят за рамки базового функционала первой версии продукта, архитектура приложения должна быть спроектирована с расчётом на возможность их внедрения в будущем (это достигается за счёт открытого API и модульного принципа построения системы).

В итоге разрабатываемое веб-приложение для автоматизации внутренних коммуникаций объединит в себе функции **внутреннего портала**, системы управления заданиями и платформы для обучения и обратной связи. Оно будет обладать интуитивно понятным веб-интерфейсом, доступным через браузер с любого устройства – что особенно важно для предприятия с разъездным характером работы (в компании «ТДМ» часть сотрудников трудится вне офиса и получает информацию в пути). Реализация всех перечисленных возможностей в рамках одной системы создаст единый центр внутренней коммуникации, где сотрудники смогут получать всю необходимую информацию и взаимодействовать с руководством в цифровом формате.

**1.3 Постановка задачи на разработку**

На основании проведённого анализа можно сформулировать постановку задачи дипломного проекта. **Компания ООО «ТДМ» нуждается в едином инструменте** для оперативного доведения информации до всего персонала и взаимодействия с ним. Решением данной проблемы станет разработка специализированного веб-приложения, интегрирующего функции автоматизации внутренних коммуникаций.

**Цель проекта** – создание корпоративного веб-приложения, предназначенного для автоматизации процессов внутреннего информирования в организации (на примере ООО «ТДМ»). Разрабатываемая система должна обеспечить своевременное доведение до сотрудников корпоративных новостей и событий, рассылку новых приказов и изменений в регламентах, а также распределение задач и контроль соблюдения установленных сроков их выполнения. Кроме того, приложение должно повысить вовлечённость персонала за счёт возможностей обучения (ознакомление с материалами, прохождение тестов) и обратной связи (участие в опросах, комментарии). Конечным результатом станет полнофункциональный внутренний веб-портал компании, облегчающий коммуникацию между руководством и сотрудниками и повышающий эффективность обмена информацией.

Для достижения поставленной цели необходимо решить комплекс задач:

1. **Аналитическая часть** – провести подробный анализ предметной области и обзор существующих решений в сфере корпоративных коммуникаций, выявить их преимущества и недостатки (эта задача выполнена в главе 1). Изучить лучшие практики информирования персонала, чтобы учитывать их при разработке собственного приложения.
2. **Проектирование системы** – разработать требования к функционалу веб-приложения и спроектировать его архитектуру. На этом этапе нужно продумать структуру базы данных, взаимодействие между клиентской и серверной частями, определить используемые технологии и инструменты разработки. Также разрабатываются схемы и диаграммы, отражающие устройство системы (диаграммы прецедентов, ER-диаграмма, схема архитектуры и пр.).
3. **Реализация программного продукта** – создать серверную часть приложения (backend), отвечающую за бизнес-логику системы (в соответствии с выбранным стеком технологий, например язык Python с веб-фреймворком и СУБД для хранения данных). Разработать клиентскую часть – пользовательский интерфейс для сотрудников и административный интерфейс для управляющего персонала – согласно спроектированным требованиям и макетам.
4. **Тестирование** – провести всестороннее тестирование веб-приложения. Необходимо проверить корректность работы всех модулей (ленты новостей, постановки задач, обучения, опросов и др.), убедиться в надёжности хранения данных и безопасности доступа, оценить удобство интерфейса для конечных пользователей. При выявлении ошибок или неудобств внести необходимые исправления.
5. **Внедрение и оценка эффективности** – подготовить технико-экономическое обоснование внедрения разработанной системы в деятельность компании. Проанализировать предполагаемые выгоды от её использования (например, экономию времени руководства на доведение информации, сокращение задержек в выполнении поручений, рост удовлетворённости сотрудников коммуникацией) и соотнести их с затратами на разработку и поддержку. Оценить, насколько эффективно новое решение решает обозначенные проблемы в сравнении с прежним состоянием.

Решение всех перечисленных задач позволит достичь поставленной цели: предоставить компании «ТДМ» работающее веб-приложение – удобный и надёжный инструмент внутреннего информирования персонала, полностью адаптированный под потребности организации.

**Выводы по главе 1**

В аналитической части дипломного проекта был проведён всесторонний анализ проблем, связанных с организацией процесса информирования персонала внутри предприятия. Рассмотрены существующие способы доведения информации до сотрудников, включая традиционные (бумажные объявления, устные сообщения) и современные (электронная почта, корпоративные чаты, внутренние порталы) инструменты, применяемые в российских и зарубежных организациях.

Анализ показал, что многие используемые решения обладают существенными недостатками: отсутствие централизованного хранения информации, невозможность подтверждения факта ознакомления с объявлением, трудности в отслеживании актуальности сообщений и управлении правами доступа. Это особенно критично для компаний с высокой динамикой процессов и распределённым штатом сотрудников, как в случае ООО «ТДМ». В результате стало очевидно, что для повышения эффективности внутреннего информационного взаимодействия необходима **специализированная система**, способная централизованно публиковать объявления, обеспечивать адресную рассылку уведомлений, контролировать ознакомление персонала с важной информацией, а также предоставлять инструменты администрирования контента и пользователей.

В качестве базы для проектирования системы выбрано реальное предприятие – ООО «ТДМ». В ходе анализа деятельности компании (в том числе в период производственной практики автора) были выявлены конкретные проблемы существующей модели информирования сотрудников и учтены специфические потребности организации. В данной главе описана предметная область и сформулированы практико-ориентированные требования к разрабатываемой системе, исходя не только из теоретических источников, но и из особенностей функционирования **конкретной компании**.

На основании собранных данных, обзора литературы и существующих решений сформулированы цели и задачи проектируемой информационной системы. Выделены основные участники процесса (роли пользователей), их информационные потребности и сценарии взаимодействия с системой. Определены ключевые функциональные блоки будущего веб-приложения: авторизация пользователей, управление объявлениями, рассылка уведомлений, подтверждение прочтения, административная панель, модуль задач и мероприятий, обучающий модуль и средства обратной связи.

Таким образом, по результатам первой главы обоснована необходимость разработки корпоративной системы информирования персонала с функцией контроля ознакомления и централизованным управлением сообщениями. Все выявленные особенности, задачи и требования послужат основой для дальнейшего проектирования и реализации программного продукта, что будет рассмотрено в следующих разделах работы.

# 2. Проектирование веб-приложения для информирования сотрудников ООО «ТДМ»

В данной главе разработан проект веб-приложения для автоматизации процессов информирования персонала в компании ООО «ТДМ». На основе требований, сформулированных в предыдущей главе, выполнены предварительные проектные решения, спроектирована структура базы данных, определена архитектура и основные компоненты системы, а также разработан пользовательский интерфейс приложения. Приведены диаграммы, отражающие функциональные возможности и структуру системы (диаграмма прецедентов, ER-диаграмма, схема архитектуры) с подробным пояснением. Все принятые технические решения обоснованы с точки зрения обеспечения требуемой функциональности, надежности и удобства использования. Комплексное проектирование, проведенное на этом этапе, формирует целостное видение системы и создает прочную основу для последующей реализации веб-приложения.

## 2.1 Разработка предварительных проектных решений

На этапе предварительного проектирования определяются границы будущей системы, её функциональные возможности и выбранные средства реализации. Были разработаны диаграмма прецедентов, описывающая взаимодействие пользователей с системой, и выполнен обоснованный выбор инструментальных средств разработки на основе анализа возможных вариантов. Эти шаги позволили сформировать целостное представление о разрабатываемом веб-приложении до перехода к деталям реализации.

### 2.1.1 Диаграмма прецедентов (use-case)

Диаграмма прецедентов является важным инструментом UML-моделирования, позволяющим наглядно представить функциональные возможности системы с точки зрения взаимодействия внешних пользователей с приложением. На **диаграмме прецедентов** изображаются акторы (роли пользователей) и прецеденты (варианты использования системы), показывающие, какие действия могут выполнять пользователи и как система реагирует на эти действия. Разработка диаграммы прецедентов для веб-приложения ООО «ТДМ» определяет полный перечень функций, необходимых для автоматизации процессов информирования персонала.

Исходя из анализа требований, в системе выделены два основных типа пользователей (актёров): **Администратор системы** и **Сотрудник** (рядовой пользователь). Администратор отвечает за управление информацией и пользователями, тогда как сотрудник является конечным получателем информационных сообщений.

Для актёра **«Администратор системы»** определены следующие ключевые варианты использования:

* **Авторизация в системе** – вход администратора в систему по учетным данным (логин и пароль) с предоставлением прав доступа к функциям администрирования. Этот прецедент является предварительным для всех последующих административных действий.
* **Добавление нового объявления** – создание администратором нового информационного сообщения (объявления) для персонала. Включает ввод заголовка и текста объявления, а также при необходимости дополнительных параметров (например, категории, даты публикации, вложенных файлов). Система сохраняет объявление в базе данных и обеспечивает его доступность сотрудникам.
* **Редактирование объявления** – внесение изменений в ранее опубликованное объявление. Администратор может исправить текст или тему объявления, обновить сроки актуальности. Система предоставляет интерфейс для обновления данных и сохраняет изменённое объявление.
* **Удаление объявления** – удаление устаревшего или ошибочно добавленного объявления. Администратор подтверждает удаление, после чего система либо удаляет объявление из базы данных, либо помечает его как неактуальное (делает невидимым для сотрудников).
* **Управление пользователями** – ведение учетных записей сотрудников. Включает регистрацию новых пользователей (создание учётной записи с указанием имени, корпоративного email-адреса, временного пароля, роли и т.д.), изменение данных существующих пользователей (сброс пароля, изменение отдела или роли) и деактивацию или удаление учётных записей ушедших сотрудников. Это позволяет поддерживать актуальный список пользователей, имеющих доступ к системе.
* **Просмотр статуса оповещений** – контроль ознакомления сотрудников с объявлениями. Администратор может просматривать, кто из персонала и когда ознакомился с каждым объявлением. Это помогает отслеживать эффективность информирования (например, видеть список сотрудников, не прочитавших важное сообщение своевременно).
* **Выход из системы** – завершение сеанса работы администратора (логаут) для защиты учётной записи от несанкционированного использования.

Для актёра **«Сотрудник»** (рядовой пользователь) определены следующие основные варианты использования:

* **Авторизация в системе** – вход сотрудника в систему по своему логину и паролю. После успешной аутентификации сотруднику предоставляется доступ к персональной области приложения (к своим объявлению и другим разрешённым разделам).
* **Просмотр списка объявлений** – получение перечня актуальных объявлений, опубликованных администратором. Список отображает заголовки объявлений, даты публикации и, возможно, краткое превью или индикатор статуса (например, пометка о новых непрочитанных сообщениях). Объявления отсортированы, например, по дате (новые – выше).
* **Просмотр содержания объявления** – открытие подробной информации по выбранному объявлению. Сотрудник видит полный текст сообщения, а также дополнительные сведения: дату публикации, автора (администратора) объявления, приложенные файлы (если имеются).
* **Подтверждение ознакомления** – после прочтения объявления сотрудник отмечает, что ознакомился с информацией (например, нажимает кнопку «Прочитано»). Система фиксирует факт и время ознакомления в базе данных. Этот прецедент реализует функцию контроля ознакомления с объявлением.
* **Выход из системы** – завершение работы сотрудника: закрытие сессии и выход из учётной записи для обеспечения безопасности (особенно важно, если доступ осуществлялся с общего компьютера).

Диаграмма прецедентов (рис. 2.1) отображает всех указанных акторов и их взаимодействие с системой через перечисленные варианты использования. Актёр «Администратор» на диаграмме связан ассоциациями со всеми прецедентами управления объявлениями и пользователями, а актёр «Сотрудник» – с прецедентами, относящимися к просмотру и подтверждению получения информации. Прецедент «Авторизация в системе» является общим для администратора и сотрудника (оба актёра инициируют этот вариант использования для входа). Прецеденты «Добавление», «Редактирование» и «Удаление объявления» логически объединяются общим сценарием **«Управление объявлениями»**, хотя на диаграмме для наглядности они могут быть показаны отдельно. Аналогично, прецедент **«Управление пользователями»** охватывает сценарии создания, изменения и удаления учётных записей сотрудников.

Таким образом, диаграмма прецедентов задаёт границы функциональности разрабатываемого приложения: система должна предоставлять средства аутентификации; возможности для администратора по созданию, рассылке и редактированию объявлений, управлению учётными записями; для сотрудников – возможности получать и просматривать объявления и подтверждать их прочтение. Данный набор прецедентов лег в основу дальнейшего проектирования архитектуры, базы данных и интерфейсов приложения.

**Рисунок 2.1 — Диаграмма прецедентов системы.**

### 2.1.2 Выбор инструментальных средств реализации

Успешная реализация веб-приложения во многом зависит от правильного выбора технологий и инструментальных средств. При выборе учитывались критерии соответствия функциональным требованиям, надежности, производительности, простоты разработки и поддержки, а также наличия необходимых компетенций у разработчиков. Ниже обоснованно выбран стек технологий для серверной и клиентской частей системы, а также дополнительные инструменты, необходимые для разработки.

**Язык программирования и платформа разработки:** в качестве основного языка реализации выбран высокоуровневый язык Python. Этот выбор обусловлен несколькими причинами. Во-первых, Python обладает простым и выразительным синтаксисом, что ускоряет разработку и облегчает сопровождение кода. Во-вторых, экосистема Python предоставляет широкий выбор библиотек и фреймворков для веб-разработки, работы с базами данных, реализации сетевого взаимодействия и др. Это позволяет использовать готовые решения для типичных задач (обработка HTTP-запросов, сериализация данных, шифрование паролей и т.п.) и сосредоточиться на бизнес-логике приложения. Кроме того, Python является переносимым и мультиплатформенным – приложение можно деплоить на различных ОС (Windows, Linux и др.), используемых в инфраструктуре ООО «ТДМ». Таким образом, Python соответствует критериям быстроты разработки и надежности и способен обеспечить необходимую функциональность проектаfile-cyys8rqmmxsbxsozbwfeqkfile-cyys8rqmmxsbxsozbwfeqk.

**Веб-фреймворк серверной части:** для создания серверной части веб-приложения выбран современный легковесный фреймворк **FastAPI** (Python). FastAPI зарекомендовал себя как высокопроизводительное средство для построения веб-API и веб-приложений. Его основные преимущества включают:

* Высокая производительность: благодаря асинхронной модели на базе ASGI и оптимизированным внутренним библиотекам (Starlette, Pydantic) FastAPI обеспечивает скорость обработки запросов, сопоставимую с наиболее быстрыми веб-фреймворками на других языках. Это важно, чтобы приложение могло обслуживать одновременные запросы от многих пользователей без ощутимых задержек.
* Встроенная валидация и сериализация данных: FastAPI использует статическую типизацию Python (type hints) и библиотеку Pydantic для автоматической валидации входных данных и удобной сериализации/десериализации JSON. Это сокращает количество шаблонного кода и повышает надежность системы, так как невалидные данные отсекаются автоматически с выдачей понятных сообщений об ошибках.
* Автодокументация API: при разработке RESTful API FastAPI автоматически генерирует интерактивную документацию (Swagger UI / OpenAPI) для всех доступных эндпоинтов. Это облегчает тестирование и интеграцию: разработчики и тестировщики могут легко просмотреть доступные методы, форматы запросов и ответов. Для данного проекта наличие самодокументируемого API полезно на этапе проверки работы серверной части.
* Простота разработки и расширяемость: синтаксис FastAPI интуитивно понятен; маршрутизация запросов, привязка URL к функциям-обработчикам и разбор входных данных реализуются буквально в несколько строк кода. Фреймворк поддерживает расширение функциональности за счёт плагинов или интеграции с другими библиотеками (например, для аутентификации, управления CORS, подключение шаблонизаторов и пр.). Это позволит без труда внедрить авторизацию пользователей и организовать выдачу как HTML-страниц, так и данных в формате JSON.

В качестве альтернатив рассматривались другие Python-фреймворки. **Flask** – минималистичный микрофреймворк – отличается простотой, но при росте приложения требует более тщательной организации кода и подключения множества расширений самостоятельно. **Django** – полноценный «всё-в-одном» фреймворк – предоставляет множество встроенных модулей (ORM, административную панель, систему аутентификации), однако может быть избыточным для сравнительно небольшого проекта и более сложен в освоении. **FastAPI** выбран как современный компромиссный вариант: он легче Django, но при этом более структурирован и функционален «из коробки», чем Flask, особенно в части создания API-сервисовfile-cyys8rqmmxsbxsozbwfeqk. Это соответствует потребностям разработки внутреннего корпоративного веб-приложения в ООО «ТДМ».

**Система управления базами данных:** для хранения данных приложения выбрана реляционная СУБД **MySQL**. MySQL является одной из наиболее распространённых СУБД, обладающей высокой производительностью и надёжностью, достаточными для информационных систем среднего масштаба. Данный выбор обоснован следующими факторами:

* Соответствие требованиям проекта: предполагаемый объём данных (информация о сотрудниках, объявлениях, фактах ознакомления) и характер операций (частое чтение, умеренное количество записей) хорошо укладываются в возможности MySQL. Реляционная модель и поддержка языка SQL позволяют обеспечить целостность данных за счёт транзакций и связей между таблицами (через первичные и внешние ключи), что важно для точного учёта связей между объявлениями и пользователями.
* Известность и поддержка: MySQL – открытое ПО с обширным сообществом и документацией. Для неё существует множество инструментов администрирования и клиентов. В частности, в проекте предлагается использовать **phpMyAdmin** – веб-интерфейс для управления MySQL. PhpMyAdmin позволяет удобно создавать структуру таблиц, просматривать и изменять данные, выполнять резервное копирование базы без необходимости работы через командную строку. Это упрощает как этап разработки (оперативное наполнение справочных таблиц, отладка запросов), так и дальнейшую поддержку системы администраторами базы данных компанииfile-cyys8rqmmxsbxsozbwfeqkfile-cyys8rqmmxsbxsozbwfeqk.
* Интеграция с Python: для Python доступны надёжные драйверы и ORM для работы с MySQL. В рамках разработки на FastAPI может использоваться библиотека **SQLAlchemy** (ORM), которая поддерживает MySQL и позволяет описывать структуру таблиц на уровне Python-классов. Это упрощает взаимодействие с базой данных (SQL-запросы заменяются вызовами методов Python) и повышает переносимость кода между разными СУБДfile-cyys8rqmmxsbxsozbwfeqkfile-cyys8rqmmxsbxsozbwfeqk. В данном проекте для доступа к данным принят подход с использованием ORM SQLAlchemy, что позволяет автоматически генерировать SQL и сократить шаблонный код. Тем не менее, при необходимости возможны прямые SQL-запросы через драйвер (например, PyMySQL) для оптимизации отдельных операций. Такой выбор обеспечивает баланс между простотой разработки и контролем над производительностью.

**Инструменты разработки интерфейса:** веб-интерфейс пользовательской части реализуется с использованием стандартных технологий **HTML** и **CSS** (для структуры и оформления страниц), а также языка **JavaScript** (для динамического поведения на стороне клиента). Для ускорения разработки интерфейса и обеспечения современного внешнего вида применяется CSS-фреймворк **Bootstrap**. Bootstrap предоставляет набор готовых стилий и компонентов (навигационные панели, таблицы, формы, кнопки), что позволяет быстро создать адаптивный дизайн без разработки всех стилей с нуля. Адаптивность интерфейса важна, поскольку сотрудники могут обращаться к системе как с компьютеров, так и с мобильных устройств – сайт должен корректно отображаться на экранах разных размеровfile-cyys8rqmmxsbxsozbwfeqkfile-cyys8rqmmxsbxsozbwfeqk.

Для генерации HTML-страниц на сервере используется шаблонизатор (например, Jinja2, совместимый с FastAPI). Шаблонизатор позволяет отделить логику формирования данных (на Python) от представления (HTML-код), следуя шаблону MVC. При этом FastAPI выполняет роль контроллера, SQLAlchemy + MySQL – модели, а HTML-шаблоны – представления. Такая структура облегчает поддержку приложения: изменения в логике или данных не требуют переписывания отображения и наоборотfile-cyys8rqmmxsbxsozbwfeqkfile-cyys8rqmmxsbxsozbwfeqk.

**Аутентификация и безопасность:** для авторизации пользователей применяются стандартные механизмы сессий (cookie-based) на стороне сервера. После ввода верных учётных данных при входе сервер создаёт пользовательскую сессию и сохраняет идентификатор сессии в cookie браузера. Далее при обращении к защищённым ресурсам проверяется наличие действительной сессии и определяются права пользователя (роль). Важно отметить, что пароли пользователей хранятся в базе данных только в зашифрованном (хешированном) виде – для этого используется, например, алгоритм bcrypt или SHA-256 из стандартных библиотек Python. Это соответствует требованиям информационной безопасности: даже в случае утечки базы данных пароли не будут доступны в открытом видеfile-cyys8rqmmxsbxsozbwfeqk.

**Дополнительные средства разработки:** проект разрабатывается в современной среде программирования, поддерживающей Python. Используется удобная IDE (например, PyCharm) или легковесный редактор VS Code с нужными плагинами, что повышает эффективность разработки за счёт подсветки синтаксиса, отладки и интеграции с системой контроля версий. В качестве системы контроля версий применяется **Git** для отслеживания изменений в кодовой базе. Для тестирования API в ходе разработки используется инструмент наподобие **Postman** (либо встроенная документация Swagger UI, автоматически генерируемая FastAPI), а для отладки веб-интерфейса – современный браузер (Chrome, Firefox) в различных разрешениях экрана, позволяющий проверить адаптивность версткиfile-cyys8rqmmxsbxsozbwfeqkfile-cyys8rqmmxsbxsozbwfeqk.

Следует отметить, что помимо веб-интерфейса, на основе выбранного стека Python при расширении проекта потенциально можно разработать и десктоп-клиент с графическим интерфейсом (например, на **PyQt6**). PyQt6 представляет собой привязки Python к GUI-библиотеке Qt, позволяющие создавать полноценные настольные приложения. В контексте данного проекта теоретически мог бы быть реализован отдельный приложение-оповещатель для администратора или сотрудников (подключающееся к серверу через API и отображающее уведомления). Однако такой подход усложняет инфраструктуру (необходимо поддерживать два клиента – веб и десктоп – и устанавливать ПО на каждое рабочее место). Поскольку целью является именно веб-приложение, основной упор сделан на браузерный интерфейс, не требующий установки и сразу доступный всем пользователям во внутренней сети предприятия. Тем не менее, возможность создания дополнительных клиентов в будущем демонстрирует, что выбранный стек (Python + FastAPI) достаточно универсален и при необходимости позволит подключать разные интерфейсы без смены ядра системыfile-cyys8rqmmxsbxsozbwfeqkfile-cyys8rqmmxsbxsozbwfeqk.

Подводя итог выбору инструментальных средств, итоговый стек технологий сформирован следующим образом. На серверной стороне: Python + FastAPI для реализации бизнес-логики приложения и API; СУБД MySQL для хранения информации (данные о сотрудниках, объявлениях, статус их прочтения); на стороне клиента: HTML/CSS/JS с Bootstrap для реализации пользовательского веб-интерфейса (с поддержкой шаблонов на сервере). Дополнительно используются вспомогательные средства разработки (IDE, системы тестирования) и администрирования базы (phpMyAdmin). Все компоненты стека согласованы и совместимы между собой: Python-фреймворк эффективно взаимодействует с MySQL через драйвер/ORM, а браузерный интерфейс потребляет данные, предоставляемые FastAPI. Такой набор технологий позволяет реализовать все функции, отражённые в диаграмме прецедентов, надёжно и в разумные срокиfile-cyys8rqmmxsbxsozbwfeqkfile-cyys8rqmmxsbxsozbwfeqk.

## 2.2 Проектирование базы данных

Проектирование базы данных является критически важным этапом, обеспечивающим надёжное хранение и структурирование информации, необходимой для работы веб-приложения. На основе функциональных требований (см. раздел 2.1.1) и сценариев использования определены основные сущности предметной области, их атрибуты и взаимосвязи. Результатом является логическая модель данных, представленная в виде ER-диаграммы («сущность-связь»). Ниже описана разработанная структура базы данных для системы информирования персонала; перечислены основные таблицы, их поля и связи между ними. На рисунке 2.2 приведена ER-диаграмма, отражающая предложенную модель данных.

С учётом специфики предметной области (оповещение сотрудников компании) в системе выделены следующие основные сущности (таблицы БД):

* **Пользователи** – хранит данные о сотрудниках, которые используют систему (получают объявления), а также об администраторах. В таблице содержится следующая информация: уникальный идентификатор пользователя (первичный ключ, id), фамилия, имя, отчество сотрудника (либо сокращенное имя для отображения), учётные данные для входа – логин (корпоративный e-mail сотрудника) и пароль (точнее, хеш пароля для безопасного хранения). Дополнительно могут храниться поля: должность, подразделение или отдел, контактные данные (телефон и др.) – при необходимости рассылки уведомлений через альтернативные каналы. Также имеется поле, отражающее **роль** пользователя в системе – например, признак администратора или значение из фиксированного набора ролей. Предполагается, что большинство записей – рядовые сотрудники, и лишь некоторые пользователи отмечены как администраторы, имеющие расширенные права. **Связи:** каждый пользователь может быть автором (создателем) нуля или более объявлений, а также получателем многих объявлений (через механизмы рассылки, описанные далее).
* **Объявления** – основная таблица, содержащая информационные сообщения, предназначенные для рассылки персоналу. Ключевые атрибуты объявления: уникальный идентификатор (id), заголовок (title) – краткое название объявления, содержание (content) – текст сообщения (произвольной длины, может содержать форматирование, ссылки и т.д.), дата и время публикации (date\_posted). Поле даты позволяет упорядочивать и фильтровать объявления по хронологии, определяя их актуальность. Кроме того, указывается **автор объявления** – внешним ключом (author\_id), ссылающимся на таблицу пользователей. Автором, как правило, выступает администратор, создавший данное объявление. Если предусмотрена категоризация объявлений, может быть поле «категория» (например, «Общее», «Для отдела продаж», «Для ИТ-отдела» и т.п.), однако в базовой модели категория не реализована для упрощения. Опционально может храниться признак статуса объявления: активно (видимо сотрудникам) или архивировано (удалено/скрыто). **Связи:** каждое объявление связано с одним пользователем-автором; кроме того, объявление предназначается одному или нескольким сотрудникам. Последняя связь является многие-ко-многим и описывается отдельной сущностью (см. ниже).
* **Оповещения (рассылка)** – таблица, фиксирующая, какие объявления направлены каким пользователям и факт их прочтения. Поскольку каждое объявление может быть адресовано нескольким сотрудникам, вводится связь типа «многие ко многим» между сущностями «Пользователь» и «Объявление». В реляционной базе данных связь многие-ко-многим реализуется через дополнительную таблицу, содержащую внешние ключи на обе связанные сущности. В нашем случае такая таблица называется, например, User\_Announcement (пользователь–объявление) или «Ознакомление». Структура: внешний ключ user\_id (на пользователя) и announcement\_id (на объявление) – вместе образуют составной первичный ключ, однозначно идентифицируя пару «данный пользователь — данное объявление». Кроме ключей, в таблице предусмотрено поле read\_date (дата/время ознакомления) или логический флаг is\_read с отметкой времени. Наличие записи в User\_Announcement означает, что определённому сотруднику адресовано определённое объявление. Если система предназначена для информирования всех сотрудников обо всех объявлениях, то при создании нового объявления администратором автоматически генерируются записи в User\_Announcement для каждого активного пользователя-получателя с пустой отметкой о прочтении. По мере того как сотрудники читают сообщение (в системе или переходя по ссылке из письма), соответствующая запись обновляется — проставляется read\_date (или переключается флаг прочтения). Такой подход позволяет отслеживать статус ознакомления каждого объявления каждым сотрудником. В случае, если объявления адресуются не всем пользователям, а выборочно (например, по отделам), в таблицу User\_Announcement включаются записи только для тех пользователей, кому предназначено сообщение. Эта таблица, таким образом, играет роль промежуточной для реализации механизма рассылки объявлений и позволяет реализовать функцию подтверждения прочтения.
* **Подразделения/Отделы** (опционально) – справочная таблица, хранящая список отделов или подразделений компании и связывающаяся с таблицей пользователей. Она может содержать идентификатор отдела, его название и другие сведения. Если в будущем будет реализовано таргетирование объявлений по подразделениям, у таблицы объявлений может появиться внешний ключ на отдел (или связь многие-ко-многим между объявлениями и отделами, если одно объявление адресовано нескольким отделам). В рамках базовой реализации эту сущность можно не вводить, считая, что любое объявление адресовано всем сотрудникам, либо указывая целевую аудиторию объявления в тексте заголовка/сообщения.
* **Журнал (логирование)** (опционально) – дополнительная таблица для записи событий системы: фактов входа пользователей, действий администратора (создание, редактирование, удаление объявлений; изменения пользователей) и др. Такая таблица помогает в аудите и отладке, но напрямую для функциональности информирования не требуется. Необходимость ведения журнала определяется политиками безопасности и сопровождения в компании. В данном проекте полноценный журнал не реализован, чтобы не усложнять систему, однако архитектура позволяет добавить его при необходимости.

Каждая из перечисленных таблиц отражена на ER-диаграмме (рис. 2.2). На диаграмме сущности «Пользователь» и «Объявление» связаны между собой посредством сущности-связи User\_Announcement. Это показывает, что один пользователь может получать многие объявления, а одно объявление предназначается многим пользователям. Связь между «Пользователем» и «Объявлением» через промежуточную таблицу развернута в две связи 1:M: от пользователя к записям рассылки и от объявления к записям рассылки. В таблице User\_Announcement установлены ограничения на уникальность пар (user\_id, announcement\_id), чтобы избежать дублирования записей, а также внешние ключи, обеспечивающие каскадную целостность: при удалении пользователя или объявления связанные записи автоматически удаляются или помечаются.

Первичные ключи в основных таблицах – автоинкрементные числовые идентификаторы (INT AUTO\_INCREMENT) для удобства работы. В таблице-связке User\_Announcement в качестве PK используется составной ключ из двух FK (что точно соответствует модели «многие ко многим»). Поля выбираются исходя из характера данных: строки (имена, логины, заголовки) – тип VARCHAR; текст объявления – тип TEXT (или LONGTEXT, если сообщения могут быть очень длинными); даты – DATE/DATETIME; булевы флаги – TINYINT(1) или BOOLEAN. Для хранения хешей паролей требуется достаточная длина поля (например, VARCHAR(255) под хеш с солью). Поля внешних ключей точно соответствуют по типу связанным первичным ключам.

Нормализация данных соблюдена в предложенной структуре: отсутствуют избыточные дублирующие сведения. Например, информация о сотруднике хранится только в таблице пользователей, а при необходимости может быть получена через JOIN с другими таблицами, вместо дублирования в каждой записи объявления. Это снижает риск неконсистентности данных. Одновременно такая структура обеспечивает гибкость: можно легко выбирать все объявления конкретного сотрудника или всех сотрудников, получателей конкретного объявления, используя соответствующие связи.

В итоге разработанная модель данных поддерживает все необходимые операции системы: добавление новых объявлений и автоматическую привязку их ко всем (или определённым) сотрудникам, фиксацию факта прочтения каждым сотрудником, выборку списка объявлений для конкретного пользователя с отметкой о прочтении и т.д. Проектирование базы данных завершает этап концептуального проектирования системы; на его основе далее строятся прикладная логика и интерфейс приложения.

**Рисунок 2.2 — ER-диаграмма базы данных системы.**

Спроектированная база данных реализуется в MySQL. С использованием phpMyAdmin (либо посредством SQL-скриптов) в СУБД создаются таблицы со всеми описанными полями и связями. Настраиваются индексы (первичные ключи автоматически имеют индекс; дополнительные индексы добавляются на внешние ключи user\_id и announcement\_id для ускорения соединений при выборках). Ограничения целостности (CONSTRAINT FOREIGN KEY) гарантируют, что нельзя создать объявление с несуществующим автором либо запись ознакомления с несуществующими IDs. Кроме того, при удалении или изменении записей верхнего уровня СУБД обеспечивает соответствующие каскадные изменения связанных записей (например, при удалении объявления автоматически удаляются связанные с ним записи в User\_Announcement).

## 2.3 Архитектура и основные компоненты веб-приложения

На основе определённых вариантов использования (раздел 2.1.1) и спроектированной модели данных (раздел 2.2) разработана архитектура системы – определены основные компоненты программного обеспечения и взаимодействие между ними. Веб-приложение для информирования персонала имеет классическую многослойную архитектуру, включающую уровень хранения данных (база данных), уровень бизнес-логики (серверное приложение) и уровень представления (пользовательский интерфейс в браузере). Архитектура строится по принципу «клиент-сервер»: пользовательский интерфейс (клиент) отделён от серверной части, что обеспечивает модульность системы, гибкость обновления и её масштабируемость. На рисунке 2.3 представлена схема архитектуры, показывающая задействованные компоненты и их связи.

**Основные компоненты веб-приложения:**

* **База данных (MySQL)** – нижний уровень архитектуры, отвечающий за хранение информации. Это централизованное хранилище, к которому обращается серверное приложение. База данных развернута на сервере внутри корпоративной сети (либо на выделенном облачном сервере). Для обеспечения производительности СУБД работает постоянно, принимая соединения от приложений. В нашей архитектуре база данных содержит описанные ранее таблицы (пользователи, объявления, связки и др.) и отвечает на SQL-запросы, выполняемые через драйвер/ORM. База гарантирует целостность данных и позволяет нескольким клиентам (потокам приложения) одновременно читать и записывать данные, используя механизмы транзакций и блокировок для сериализации параллельных операций.
* **Серверное веб-приложение (Python + FastAPI)** – средний уровень, содержащий всю бизнес-логику системы. Этот компонент представляет собой запускаемое серверное приложение (процесс), работающее на веб-сервере внутри инфраструктуры компании. Серверное приложение реализует следующие функции:
  + Обработка входящих HTTP-запросов от клиентов (браузеров). FastAPI предоставляет маршрутизацию: каждому URL или шаблону URL соответствует определённый метод (эндпоинт) в коде Python. При обращении к этому URL выполняется соответствующая функция. Например, GET-запрос к адресу /announcements вызывает функцию, которая обращается к базе данных и возвращает список объявлений.
  + Управление сессиями и аутентификацией. Сервер проверяет присланные учётные данные при логине, сравнивает хеш пароля с сохранённым в базе и при успешной авторизации создаёт пользовательскую сессию. Далее, при обращении к защищённым ресурсам, сервер убеждается, что запрос связан с валидной сессией (или токеном) и определяет права пользователя на основании его роли.
  + Операции с базой данных. Серверное приложение содержит слой доступа к данным: либо напрямую формирует SQL-запросы через драйвер MySQL (например, с помощью PyMySQL), либо использует ORM (SQLAlchemy), которая генерирует SQL автоматически. Например, при создании нового объявления код сервера формирует SQL-вставку в таблицу объявлений и соответствующие записи в таблицу рассылки; при запросе списка объявлений для пользователя – выполняется SELECT с соединением таблиц объявлений и рассылки, отфильтрованный по user\_id.
  + Реализация бизнес-правил. Сюда относятся ограничения и логика, специфичные для предметной области: например, запрет публикации объявления с датой в прошлом; ограничение на максимальный размер текста объявления; определение поведения при удалении пользователя (например, должны ли его объявления удаляться либо переходить к другому администратору); автоматическая генерация записей рассылки всем сотрудникам при создании нового объявления; отправка e-mail уведомлений и т.д. Бизнес-логика инкапсулирована на сервере, благодаря чему клиентская часть остаётся «тонкой».
  + Формирование ответов клиенту. Если используется подход серверного рендеринга страниц, то серверное приложение генерирует HTML-код, подставляя данные в шаблоны, и отправляет готовую страницу браузеру. Если реализована архитектура типа SPA (Single Page Application) или отдельный фронтенд, то сервер предоставляет REST API (возвращает данные в формате JSON/XML), а формирование финального вида происходит на стороне клиента. В нашем случае для упрощения большинство страниц рендерится на сервере (что не требует сложного JS-кода у клиента), а для отдельных динамических действий (например, отметить объявление прочитанным без перезагрузки страницы) могут использоваться AJAX-запросы к REST API.
  + Взаимодействие с внешними сервисами. При необходимости серверное приложение взаимодействует с внешними сервисами: например, с почтовым SMTP-сервером для отправки писем. В нашем проекте это реализовано для подтверждения адреса электронной почты при регистрации пользователя и для сброса пароля (рассылка ссылок на специальную страницу сброса пароля). Кроме того, предусмотрена возможность отправки email-оповещений о новых объявлениях всем сотрудникам. Данный функционал расширяет каналы информирования (помимо самой веб-системы), но не является обязательным для базовой работоспособности приложения.

Серверное приложение, реализованное на FastAPI, является единой точкой входа для всех клиентов: оно слушает определённый порт (например, 80/443) и обрабатывает поступающие HTTP-запросы. Внутри серверного приложения логически выделены модули/подсистемы: модуль управления пользователями (регистрация, аутентификация, сброс пароля), модуль управления объявлениями, модуль отслеживания прочтения и др. Архитектурно сервер придерживается принципов многослойности: уровень доступа к данным отделён от уровня обработки запросов и формирования ответов.

* **Клиентское приложение (веб-интерфейс пользователя)** – верхний уровень архитектуры, непосредственно взаимодействующий с конечным пользователем (сотрудником или администратором). В роли клиента выступает веб-браузер, отображающий страницы интерфейса. Страницы получаются от сервера в виде готового HTML (в многостраничном приложении) либо формируются JavaScript-кодом на основе данных (в SPA). В данном проекте основные функции реализованы как обычные многостраничные формы, поэтому браузер получает от сервера сгенерированные HTML-страницы. Пользователь взаимодействует с элементами страницы (ссылками, кнопками, формами), инициируя новые запросы на сервер (переходы по ссылкам или отправку форм). Например, при клике на объявление в списке браузер запрашивает страницу с деталями объявления; при заполнении формы создания объявления администратор отправляет POST-запрос на сервер и т.д.

Клиентская часть может включать JavaScript для повышения интерактивности: например, чтобы без перезагрузки страницы пометить объявление прочитанным (AJAX-запрос с последующим обновлением элемента страницы), либо для валидации данных на стороне клиента (подсказка о формате пароля, проверка заполнения обязательных полей формы перед отправкой). Однако даже без сложного JS система остаётся функциональной, так как основная логика реализована на стороне сервера. Использование фреймворка Bootstrap упрощает адаптивную верстку, а также даёт некоторые интерактивные компоненты (например, модальные окна подтверждения удаления, выпадающее меню навигации), которые работают с помощью встроенных скриптов Bootstrap.

С точки зрения модели «клиент-сервер», клиент (браузер) является «тонким»: он не хранит постоянных данных (вся информация хранится на сервере), а лишь запрашивает и отображает нужные пользователю сведения по мере необходимости. Это соответствует принципам централизованного хранения и облегчает управление данными – все изменения происходят через сервер и сразу отражаются для всех клиентов.

Помимо основных компонентов, архитектура включает и окружение приложения:

* **Сетевое окружение:** приложение развёрнуто или планируется к развёртыванию во внутренней сети компании «ТДМ» (интранет) либо на защищённом внешнем сервере. Пользователи подключаются к веб-интерфейсу через браузер по протоколу HTTP/HTTPS. Для безопасности предпочтительно использовать HTTPS (SSL-сертификат) и механизмы корпоративной аутентификации (например, интеграция с Active Directory/LDAP для единых учётных записей), если это предусмотрено политикой компании.
* **Веб-сервер (контейнер приложения):** FastAPI-приложение во время разработки запускается со встроенным сервером (Uvicorn в режиме debug). Для промышленной эксплуатации планируется использовать связку **Uvicorn/Gunicorn** или запуск под управлением **Nginx**. Например, **Gunicorn** может выполнять роль мастер-процесса, запускающего несколько воркеров FastAPI, а **Nginx** – работать фронтенд-сервером, проксирующим запросы приложению и раздающим статические файлы (CSS, JS, изображения). Такая схема повышает надёжность и масштабируемость: Nginx может распределять нагрузку между несколькими воркерами, обслуживая больше одновременных пользователей.
* **Резервное копирование и восстановление:** для базы данных настроен механизм регулярного резервного копирования (dump MySQL), чтобы в случае сбоя или потери данных можно было восстановить информацию и возобновить работу системы с минимальными потерями. Это относится скорее к вопросам эксплуатации, но учитывается при проектировании как требование к инфраструктуре.

На схеме архитектуры (рис. 2.3) визуализировано взаимодействие всех перечисленных компонентов. Акторы (Администратор, Сотрудник) через браузер обращаются к серверному приложению (FastAPI), которое в свою очередь взаимодействует с базой данных MySQL. Стрелками на схеме показываются потоки данных: запросы от пользователя к серверу, запросы сервера к БД и ответы обратно. К примеру, последовательность для сценария «сотрудник просматривает список объявлений» выглядит так: **Сотрудник** через интерфейс отправляет GET-запрос на URL списка объявлений -> **сервер** (FastAPI) получает запрос, обращается к **БД** (SQL SELECT из таблицы объявлений с фильтрацией по пользователю/непрочитанным) -> сервер получает данные, формирует HTML-страницу и возвращает ответ -> **браузер** отображает список объявлений. Аналогичный принцип коммуникации применяется ко всем действиям (добавление объявления: браузер отправляет форму -> сервер пишет в БД -> сервер перенаправляет или возвращает обновлённый список и т.д.).

Преимущество такой распределённой архитектуры – разделение ответственности между уровнями. Например, изменение схемы базы данных потребует изменений только в серверной части (уровень доступа к данным) и не затронет логику интерфейса, пока соблюдается «контракт» (формат данных). И наоборот, доработка интерфейса (представления) не влияет на способ хранения данных. Кроме того, клиент-серверная модель позволяет подключать разные типы клиентов: сейчас это веб-браузер, но в будущем ничто не мешает разработать мобильное приложение или настольный клиент, которые будут использовать тот же серверный API для получения данных. Это расширит аудиторию и способы доступа к системе без изменения её ядра.

Архитектура также удовлетворяет требованиям масштабируемости. На уровне базы данных производительность может быть повышена оптимизацией запросов, настройкой индексов, при росте нагрузки – репликацией или кластеризацией MySQL. На уровне сервера – использованием нескольких процессов/потоков обработки запросов (благодаря асинхронности FastAPI эффективно работает с вводом-выводом, не блокируя весь сервер во время ожидания БД). На уровне клиента – добавлением промежуточного кеширования (например, если бы объявления редко менялись, их список можно временно кешировать), хотя внутри корпоративной сети в этом обычно нет необходимости из-за невысоких задержек.

В целом выбранная архитектура сочетает надёжность (за счёт централизованной БД), гибкость (чёткое разделение на слои), безопасность (контроль доступа на сервере, шифрование трафика) и расширяемость (возможность подключать разные интерфейсы и масштабировать компоненты). Она послужила основой при дальнейшей реализации системы и обеспечивает выполнение всех предъявленных требований.

**Рисунок 2.3 — Схема архитектуры веб-приложения.**

## 2.4 Разработка пользовательского интерфейса

Пользовательский интерфейс веб-приложения играет ключевую роль в успешности системы информирования персонала, так как именно через интерфейс сотрудники и администраторы взаимодействуют с приложением. При проектировании UI учитывались требования удобства, простоты и интуитивности: пользователи должны без труда находить нужную информацию и выполнять необходимые действия. В данном разделе описаны основные решения по дизайну интерфейса – общий подход к проектированию экранных форм, а также особенности реализации интерфейса администратора и интерфейса рядового сотрудника. Каждый элемент интерфейса описывается текстово, что позволит в дальнейшем легко воссоздать макеты и реализовать их в коде с использованием выбранных технологий (HTML, CSS, Bootstrap, JS).

### 2.4.1 Проектирование макетов экранных форм

Перед непосредственной реализацией интерфейса был выполнен этап проектирования макетов экранных форм (wireframes) – схематичного отображения структуры страниц и размещения элементов. Это необходимо для проверки удобства навигации и соответствия интерфейса бизнес-логике и ожиданиям пользователей. Макеты соответствуют основным сценариям использования, выявленным ранее. Ниже перечислены основные экраны, которые должны быть реализованы в веб-приложении:

* **Страница авторизации (Login page):** точка входа в систему, где пользователь (администратор или сотрудник) вводит свои учётные данные. Макет включает поля ввода «Логин» (адрес корпоративной почты) и «Пароль», кнопку «Войти», а также логотип компании и название системы. Реализована ссылка «Забыли пароль?» для перехода к восстановлению пароля. При ошибке ввода (неверные данные, незавершённая регистрация) отображается сообщение об ошибке. Дизайн формы – лаконичный, с центрированным блоком на странице, с использованием корпоративных цветов ООО «ТДМ» в элементах оформления.
* **Главная страница сотрудника (список объявлений):** основной экран для рядового пользователя после входа. Здесь представлен перечень всех актуальных объявлений, доступных данному сотруднику. Каждый элемент списка содержит заголовок объявления, дату публикации и при необходимости краткий превью текста. Элементы списка кликабельны (нажатие открывает страницу подробностей объявления). На странице присутствует навигационная панель (header) с названием системы, указанием текущего пользователя и кнопкой «Выход». Интерфейс визуально выделяет непрочитанные объявления (например, заголовок отображается полужирным шрифтом или помечается ярлыком «Новое»).
* **Страница просмотра объявления (детали объявления):** форма, отображающая полный текст конкретного объявления. Содержит заголовок (крупно), подзаголовок или метаданные (кто автор и когда опубликовано, возможно категория или важность), ниже – текст объявления в развёрнутом виде. Если к объявлению приложены файлы (документы, изображения), они перечислены на странице в виде ссылок для скачивания. Для навигации предусмотрена кнопка «Назад к списку» либо хлебные крошки («Объявления > Заголовок объявления»). Внизу страницы размещается кнопка **«Отметить как прочитано»** – при нажатии сотрудник подтверждает ознакомление с объявлением. После нажатия кнопки с помощью AJAX-запроса на сервер фиксируется факт прочтения; интерфейс обновляется: вместо кнопки отображается пометка «Помечено как прочитанное» (либо страница перенаправляется обратно к списку). Если пользователь возвращается к списку объявлений, данное объявление уже не будет выделено как новое. Если сотрудник покинет страницу, не нажав кнопку, объявление останется отмеченным как непрочитанное в списке, сигнализируя, что он его не подтвердил.
* **Административная панель (dashboard администратора):** главный экран для пользователя с ролью администратора после входа. Панель включает меню навигации по разделам администрирования (например, вкладки или боковое меню: «Объявления», «Пользователи»). В верхней части страницы отображается приветствие, например: «Администратор: Иванова И.И.», а также кнопка «Выход». Центральная область может использоваться для сводной информации (например, количество активных объявлений, число пользователей, статистика прочтения), однако в базовой реализации здесь просто приветственное сообщение. Основное управление осуществляется через отдельные разделы.
* **Раздел «Объявления» (управление объявлениями):** отображает список всех объявлений в системе с возможностью их редактирования. Интерфейс аналогичен пользовательскому списку объявлений, но содержит дополнительные элементы управления. Возле каждого объявления для администратора доступны кнопки **«Редактировать»** и **«Удалить»**. Над списком располагается кнопка **«Создать объявление»** для перехода к форме добавления нового сообщения. Если объявлений очень много, могут быть реализованы инструменты поиска и фильтрации (по дате, автору, ключевым словам), однако в рамках небольшой компании можно обойтись сортировкой по дате. При нажатии «Редактировать» администратор переходит на форму редактирования (аналогичную форме добавления, но с заполненными текущими данными). При нажатии «Удалить» запрашивается подтверждение (например, через модальное окно «Вы действительно хотите удалить объявление X? Это действие необратимо.»). После подтверждения объявление удаляется, и список обновляется. (Если политика требует не удалять физически, а архивировать объявления, можно вместо удаления помечать их как архивные – в таком случае в интерфейсе можно, например, убирать их из списка или отображать в отдельном разделе «Архивные объявления»).
* **Форма добавления/редактирования объявления:** предоставляет поля для ввода всех атрибутов объявления. Включает: поле «Заголовок» (текстовое, обязательное), поле «Текст объявления» (многострочное, обязательное; возможно с поддержкой базового форматирования или WYSIWYG-редактором для удобства администратора), опции – например, флажок «Отправить уведомление на email» (если нужно сразу разослать оповещение о новом объявлении по почте сотрудникам). Также предусмотрен элемент для добавления вложений (кнопка «Прикрепить файл»). Внизу формы – кнопки «Сохранить» и «Отмена» (вернуться к списку). При сохранении данные отправляются на сервер (POST-запрос); сервер выполняет валидацию (проверка, что заголовок и текст не пустые, размер вложения допустим и т.д.). В случае успеха новое объявление сохраняется в базу, для всех пользователей создаются записи в таблице рассылки, и (если был отмечен соответствующий флажок) сотрудникам отправляются email-уведомления. Администратор возвращается к списку объявлений, где отображается сообщение об успешной публикации. Если при сохранении возникли ошибки валидации, форма отображается заново с подсветкой проблемных полей и сообщениями об ошибках (например, «Поле 'Заголовок' не заполнено» или «Размер файла превышает допустимый»).
* **Страница «Пользователи» (управление пользователями):** отображает таблицу со списком всех пользователей системы. Таблица содержит колонки: имя сотрудника (ФИО), логин (email), отдел (если ведётся учёт подразделений), роль (например, «Администратор» или «Пользователь»), статус (активен/блокирован). Над таблицей – кнопка **«Добавить пользователя»** для создания новой учётной записи. В каждой строке для администратора доступны действия: **«Изменить»** (иконка карандаша для редактирования данных пользователя) и **«Удалить/Деактивировать»** (иконка корзины). При большом количестве пользователей может быть реализован поиск/фильтр по имени, отделу и т.д.
* **Форма добавления/редактирования пользователя:** содержит поля для ввода/редактирования данных сотрудника: Фамилия, Имя, Отчество, логин (корпоративный email, служащий одновременно логином для входа), отдел (выбор из списка отделов – если используется функциональность подразделений), роль (переключатель или выпадающий список – «Пользователь» или «Администратор»), пароль. При добавлении пользователя пароль может задаваться вручную либо генерироваться автоматически (и, например, отображаться администратору для передачи сотруднику). При редактировании учетной записи поле пароля можно оставить пустым, если нет необходимости менять пароль. Кнопки формы: «Сохранить» и «Отмена». При сохранении новый пользователь создаётся в базе (с отметкой, что ему нужно подтвердить аккаунт или сменить выданный временный пароль при первом входе), либо обновляются поля существующего пользователя. Если настроена отправка уведомлений, новому пользователю на указанный email отправляется письмо с ссылкой для подтверждения регистрации или с временным паролем.
* **Профиль пользователя:** страница, где сотрудник может просмотреть и при необходимости изменить свои личные данные. Предусмотрена возможность смены пароля (с вводом текущего, нового и подтверждения нового пароля). По сути, форма профиля содержит аналогичный набор полей, что и форма редактирования пользователя администратором, но только для своей учётной записи. Сотрудник может обновить, например, свой номер телефона или другие контактные сведения. После сохранения изменения записываются в базу данных. Наличие страницы профиля не является строго обязательным (администратор при необходимости может сам править данные пользователей), однако увеличивает удобство для сотрудников и разгружает администратора от рутины.

Перечисленные макеты отражают основные интерфейсные сущности системы. При разработке дизайна интерфейса соблюдались следующие принципы:

* **Единообразие и интуитивность:** элементы интерфейса (кнопки, списки, таблицы, иконки) оформлены единообразно во всех разделах. Используются общепринятые обозначения: иконка карандаша для редактирования, корзины для удаления, звёздочка для обязательных полей и т.д. Это делает интерфейс понятным пользователям без дополнительного обучения.
* **Удобная навигация и доступность:** все основные функции доступны через понятное меню или панель. Администратор чётко видит разделы «Объявления» и «Пользователи». Сотрудник видит только те элементы меню, которые ему нужны (например, ссылку на список объявлений, возможно, на свой профиль). Кнопка выхода из системы находится на видном месте (например, в шапке справа). Интерфейс локализован на русский язык (все надписи – на русском).
* **Адаптивность:** макеты спроектированы с расчётом на адаптивную вёрстку под разные размеры экрана. На широком экране (настольный ПК) меню может отображаться горизонтально в шапке, а на узком (смартфон) превращаться в «гамбургер»-меню. Таблицы при недостатке ширины либо прокручиваются горизонтально, либо трансформируются (например, каждая запись отображается блоком). Крупные кнопки и поля удобно нажимать и на сенсорных экранах. Таким образом, интерфейс одинаково доступен на компьютере и мобильных устройствах благодаря responsive-дизайну (Bootstrap-сетка и гибкие единицы измерения).
* **Минимизация действий пользователя:** интерфейс спроектирован так, чтобы требовать минимальное число шагов для достижения цели. Например, подтверждение прочтения – одним нажатием; создание объявления – одна форма; редактирование пользователя – также одна форма. Предусмотрены уведомления о результате действий (всплывающие подсказки или статические сообщения на странице), чтобы пользователь понимал, что действие выполнено успешно (например, надпись «Объявление успешно добавлено» после сохранения).
* **Соответствие корпоративному стилю:** в оформлении интерфейса используются цвета и элементы фирменного стиля компании (логотип, цветовая схема), если таковые утверждены. Это создаёт ощущение единого пространства и официальности инструмента для сотрудников. В нашем случае, например, применены оранжево-белые тона, присутствующие в логотипе «ТДМ», для панелей и выделения важных элементов.

Разработанные макеты экранов послужили основой при реализации интерфейса с помощью Bootstrap и связанных технологий. Далее рассмотрены специфические особенности интерфейса отдельно для администратора и для рядового сотрудника, поскольку у этих ролей различается набор доступных функций.

### 2.4.2 Интерфейс администратора

Интерфейс администратора предназначен для сотрудников, ответственных за создание и управление объявлениями (например, работников отдела кадров или иных уполномоченных лиц). Этот интерфейс включает все инструменты, необходимые для актуализации информации в системе и контроля за оповещением персонала.

После входа под учётной записью администратора пользователь попадает на административную панель. Панель представляет собой страницу с навигацией по разделам администрирования. В верхней части отображается приветствие (имя администратора) и элементы навигации/меню. Для удобства интерфейс разделён на основные разделы (вкладки либо пункты меню):

* **Объявления** – открывает страницу со списком всех объявлений и операциями над ними (этот раздел обычно загружается по умолчанию первым).
* **Пользователи** – раздел для управления учётными записями сотрудников.  
  (При необходимости могут быть дополнительные разделы, например «Статистика» или «Настройки», но в базовой версии реализации достаточно двух указанных.)

В разделе **«Объявления»** администратор видит список всех объявлений, как описывалось ранее. Каждая строка списка содержит заголовок, дату публикации, автора (если администраторов несколько) и статус (активно/архив). Справа от каждого объявления находятся кнопки действий: «Редактировать» и «Удалить». Вверху раздела – кнопка «Создать объявление» для добавления нового сообщения.

При необходимости администратор может просмотреть содержание объявления так же, как его видят сотрудники. Это можно организовать либо кликом по заголовку (открывается страница деталей объявления с возможностью вернуться назад), либо отдельной кнопкой «Просмотр». Однако обычно администратору достаточно видеть заголовок и помнить содержание, либо зайти в режим редактирования, поэтому отдельная функция просмотра может быть опущена.

**Создание объявления:** при нажатии «Создать объявление» открывается форма добавления. Администратор заполняет поля: заголовок (обязательное), текст объявления (обязательное), при необходимости прикрепляет файл. Также он может отметить чекбокс «Отправить уведомление на email», если хочет продублировать объявление письмом всем сотрудникам. При отправке формы происходит проверка данных: поля не должны быть пустыми, файл – допустимого типа и размера и т.д. В случае ошибок отображаются соответствующие сообщения. Если всё в порядке, система сохраняет новое объявление в таблице объявлений, добавляет записи в таблицу User\_Announcement для всех сотрудников (тем самым назначая им это объявление для ознакомления) и отправляет почтовые уведомления (если было отмечено). После успешного сохранения администратор возвращается к списку объявлений, где видит только что добавленное сообщение (обычно первым в списке) и подтверждение об успешной публикации.

**Редактирование объявления:** интерфейс редактирования практически идентичен созданию, но форма открывается уже заполненной текущими значениями полей объявления. Администратор вносит необходимые правки (исправляет текст, обновляет файл, продлевает актуальность и т.д.). После сохранения изменения записываются в базу данных. Все сотрудники, которые еще не прочитали объявление, при входе увидят уже обновлённую версию сообщения. При необходимости можно предусмотреть пометку «обновлено» с датой изменения, чтобы сотрудники замечали изменения. Если были изменены критичные данные, администратор может повторно отправить email-уведомление (для этого, например, устанавливается флажок «Разослать обновление»). В интерфейсе редактирования ясно обозначено, что это режим правки существующего объявления (например, заголовок формы «Редактирование объявления» вместо «Новое объявление»).

**Удаление объявления:** при нажатии на иконку «Удалить» (корзина) у соответствующей строки объявления система запрашивает подтверждение. Отображается модальный диалог: «Вы действительно хотите удалить объявление “X”? Это действие необратимо.» с кнопками подтверждения и отмены. После подтверждения сервер выполняет удаление объявления: запись удаляется из таблицы объявлений, а связанные записи в таблице User\_Announcement либо тоже удаляются каскадно, либо помечаются как прочитанные/неактуальные (в зависимости от выбранной реализации). Интерфейс обновляет список, исключая удалённое объявление. Если политика компании требует сохранять историю, то вместо полного удаления можно помечать объявление как архивное – тогда в интерфейсе оно либо вовсе не отображается рядовым сотрудникам, либо доступно только в специальном фильтре «Архив». В нашей реализации допускается полное удаление для упрощения.

**Управление пользователями:** этот раздел интерфейса позволяет поддерживать актуальный состав пользователей системы. Список пользователей представлен таблицей со столбцами: ФИО, логин (email), отдел (если используется), роль, статус (активен/блокирован). Над таблицей – кнопка «Добавить пользователя».

* **Добавление пользователя:** при нажатии открывается форма создания новой учётной записи. Администратор вводит ФИО сотрудника, его корпоративный email (он же логин), выбирает отдел из списка (при необходимости), устанавливает роль (по умолчанию «Пользователь»; если нужно дать права администратора – переключает). Пароль может быть задан вручную либо сгенерирован автоматически. После сохранения новый пользователь записывается в БД. Система отправляет на указанный email письмо с ссылкой для подтверждения регистрации (активации учётной записи) – либо, если email-сервер внутренний, администратор сам сообщает сотруднику временный пароль. В интерфейсе после добавления отображается сообщение «Пользователь успешно добавлен» и новый пользователь появляется в списке.
* **Редактирование пользователя:** напротив каждой записи пользователя есть кнопка «Изменить». Администратор может изменить отдел, роль, статус учётной записи и при необходимости сбросить пароль. Например, если сотрудник уволен, администратор изменяет статус на «блокирован» или удаляет учётную запись. Если сотрудник забыл пароль, администратор может нажать кнопку «Сбросить пароль» – система сгенерирует новый временный пароль, сохранит его (в хешированном виде) и при необходимости отправит письмо сотруднику с инструкцией по смене пароля. Либо администратор может прямо на странице редактирования задать новый пароль вручную. После сохранения изменения фиксируются в базе.
* **Удаление/блокировка пользователя:** если сотрудник увольняется, возможны два подхода – полное удаление записи или блокировка. Удаление приводит к потере связанных данных (например, информация о прочтении объявлений этим сотрудником), поэтому предпочтительно блокировать: помечать пользователя как неактивного, запретив ему вход, но сохранив его записи в системе. В интерфейсе это может быть реализовано как переключение статуса (галочка «Активен» снимается, или кнопка «Деактивировать»). Заблокированный пользователь исключается из списка активных, не может авторизоваться, но данные о ранее полученных им объявлениях сохраняются. Если всё же реализуется удаление, система должна либо каскадно удалять связанные данные, либо не позволять удалять пользователя, за которым числятся объявления/рассылки. Интерфейс в таком случае выдаст предупреждение, например: «Нельзя удалить пользователя, так как он является автором объявлений» – и потребует сначала переназначить эти объявления другому администратору.

Дополнительный функционал для администратора – **контроль ознакомления с объявлениями**. Как упоминалось, администратор может просматривать, кто прочитал каждое объявление. В интерфейсе раздела «Объявления» это можно реализовать с помощью пиктограммы «глаз» или ссылки «Статистика» рядом с каждым объявлением. При клике открывается модальное окно или отдельная страница со списком пользователей и статусом: например, «Ознакомлено: 12 из 15 сотрудников. Не ознакомлены: Иванов И.И., Петров П.П., ...». Это позволяет администратору оперативно видеть, кто не прочитал важное сообщение, и при необходимости напомнить им по другим каналам.

Визуально интерфейс администратора отличается от интерфейса обычного сотрудника наличием дополнительных элементов управления и иной структурой навигации (административного меню). Однако стилистически они оформлены единообразно, в одном ключе. Благодаря Bootstrap это достигается легко: используются одни и те же компоненты (Navbar для верхней панели, Table для списков, модальные окна для подтверждений и т.п.).

Особое внимание уделено защите от ошибок в админ-интерфейсе: поскольку действия администратора влияют на многих пользователей, система всегда запрашивает подтверждение на потенциально опасные операции (удаление объявления, удаление пользователя) и проверяет вводимые данные (нельзя создать пользователя без логина, нельзя завести дублирующийся логин – при попытке сохранения UI отобразит ошибку, полученную от сервера о нарушении уникальности).

В итоге интерфейс администратора предоставляет полный набор функций для управления системой: создания и рассылки объявлений, контроля их доставки (прочтения), а также сопровождения базы пользователей. После внедрения данного интерфейса администраторы компании «ТДМ» смогут самостоятельно поддерживать актуальность новостной ленты для сотрудников, практически не прибегая к помощи ИТ-специалистов, что и является целью автоматизации процесса информирования.

### 2.4.3 Интерфейс рядового пользователя

Интерфейс рядового пользователя (сотрудника компании) предназначен для конечных получателей информации. Он максимально простой, понятный и не требует специального обучения, поскольку сотрудники будут использовать систему наряду с другими повседневными инструментами. Главная задача пользовательского интерфейса – своевременно довести до каждого сотрудника все предназначенные ему объявления и позволить удостовериться, что информация получена (через механизм отметки прочтения).

После успешной авторизации сотрудник попадает на главную страницу пользователя – список актуальных объявлений. Эта страница представляет собой ленту корпоративных оповещений, адресованных данному сотруднику. В базовой реализации политика распространения объявлений – «все объявления видны всем сотрудникам» (универсальная рассылка). Если в будущем потребуется таргетировать объявления по отделам, систему можно доработать (например, в каждое объявление добавлять поле «отдел» и показывать сообщение только пользователям этого отдела). Но пока каждый сотрудник видит единый список объявлений, публикуемых в компании.

Интерфейс списка объявлений уже описан вкратце: это перечень заголовков объявлений в виде списка (или карточек) с датами. Сотрудники могут получать достаточно много сообщений, поэтому интерфейс обеспечивает удобство ориентирования:

* Объявления упорядочены по дате публикации (новейшие – вверху списка).
* **Непрочитанные** объявления визуально выделяются (например, отображаются жирным шрифтом или снабжаются меткой «Новое!»). После ознакомления сотрудника метка исчезает (или стиль меняется на обычный).
* Если объявление помечено администратором как важное, может отображаться специальный значок (например, восклицательный знак) рядом с заголовком, привлекая внимание.
* Возможна пагинация или кнопка «Показать больше», если объявлений накопилось очень много. Однако на практике внутренняя корпоративная лента вряд ли будет содержать сотни сообщений за раз, поэтому можно отображать все объявления за последний год или все доступные, возможно с разделителями по месяцам для удобства.
* Если список объявлений пуст (например, для нового сотрудника или когда нет актуальных сообщений), интерфейс выводит сообщение «Нет объявлений».

Сотрудник нажимает на интересующее его объявление в списке и переходит на страницу подробного просмотра объявления. На странице деталей объявления отображается вся информация: заголовок, полный текст, вложения (если есть). Текст объявления может содержать простое форматирование (списки, выделения) – это отображается благодаря тому, что либо хранится как HTML (с ограниченным набором тегов), либо рендерится с сохранением форматирования. В случае хранения HTML-содержимого следует убедиться, что интерфейс безопасно экранирует потенциально опасные элементы (например, ограничивает список допустимых тегов), чтобы предотвратить XSS-уязвимости.

На странице объявления присутствует элемент для подтверждения ознакомления. В нашей реализации это явная кнопка «Прочитал(а)». После того, как сотрудник дочитал объявление, он нажимает эту кнопку (расположенную в конце текста). По нажатию с помощью JavaScript выполняется запрос к серверу (например, PUT/PATCH к ресурсу подтверждения чтения объявления). Сервер помечает соответствующую запись в таблице User\_Announcement (записывает время прочтения). Интерфейс мгновенно реагирует: кнопка исчезает, и появляется текст или значок, указывающий, что объявление помечено как прочитанное. Если пользователь вернётся к списку объявлений (кнопкой «Назад» или через меню), он увидит, что данное объявление уже не выделено как новое.

Если сотрудник открыл страницу объявления, но ничего не нажал (просто закрыл страницу), то подтверждение прочтения не зафиксировано – в списке объявлений оно останется помеченным как непрочитанное. Именно поэтому был выбран явный механизм отметки, чтобы исключить ситуацию, когда сотрудник случайно открыл объявление, не прочитал, а система посчитала его прочитанным. В будущем, если будет решено автоматизировать подтверждение (без участия пользователя), можно реализовать фиксацию факта открытия страницы – тогда на странице деталей внизу можно отображать пометку вроде «Вы ознакомились с этим объявлением в 12:45 01.06.2025», но в данной реализации сделан упор на осознанное подтверждение через кнопку.

Элементы навигации и прочие части интерфейса сотрудника аналогичны таковым у администратора, но упрощены. На страницах пользователя в шапке отображается название системы, имя текущего пользователя (может быть ссылкой на профиль, если он реализован) и кнопка «Выход». Боковое меню как правило отсутствует, поскольку функциональность для сотрудника ограничена просмотром объявлений (и профиля). Таким образом, интерфейс не перегружен лишними элементами.

Если предусмотрен доступ сотрудника к архивным объявлениям (например, для просмотра старых сообщений), можно добавить переключатель «Показывать архивные». По умолчанию отображаются только актуальные объявления (за последние N месяцев). Однако в рамках данной системы решено показывать все объявления подряд, поскольку их общее количество невелико.

Предполагается, что сотрудники будут регулярно заходить в систему самостоятельно, проверяя новые объявления. Дополнительно настроен механизм уведомлений: при публикации нового объявления каждому сотруднику отправляется письмо на корпоративную почту с кратким содержанием и ссылкой на веб-приложение для прочтения полного текста. Это служит «push»-уведомлением, побуждая сотрудника оперативно зайти и ознакомиться. В перспективе можно рассмотреть и другие каналы – например, push-уведомления через браузер или мобильное приложение, интеграцию с корпоративным мессенджером – но в рамках текущего проекта достаточно email-рассылки.

В пользовательской части форм для ввода немного. Помимо формы входа и (опционально) формы смены пароля, никаких других форм нет – сотрудник только читает информацию. Эти формы имеют стандартные проверки: поле логина должно содержать корректный адрес почты, поле пароля не должно быть пустым. Если при входе данные неверны, сервер возвращает ошибку, и интерфейс показывает сообщение «Неправильный email или пароль». При смене пароля проверяется совпадение нового пароля и подтверждения.

Если сотрудник случайно попытается зайти на административный URL или на страницу, которой у него быть не должно, система выдаёт сообщение об ошибке («Доступ запрещён») либо переадресует на главную страницу. Такие ситуации обрабатываются на сервере, а интерфейс отображает стандартную страницу ошибки (оформленную в стиле общего дизайна, с понятным текстом на русском языке).

Благодаря адаптивному дизайну интерфейс удобен и на мобильных устройствах. При просмотре на смартфоне список объявлений превращается в удобную вертикальную ленту, шрифт и кнопки имеют достаточный размер для нажатия пальцем. Меню «Выход» и другие элементы шапки могут группироваться в выпадающее меню (иконка «гамбургера»). Текст объявлений можно прокручивать. Эти эффекты достигаются использованием возможностей Bootstrap и проверены тестированием на различных разрешениях экрана.

В итоге интерфейс рядового пользователя решает задачу предоставления актуальной информации в простой форме. Сотрудники компании, даже не обладая особыми навыками, смогут легко авторизоваться в системе и получать объявления своевременно. За счёт удобства интерфейса ожидается повысить охват информирования: снижается вероятность, что сотрудник пропустит важное сообщение (как это могло случиться при рассылке только по email или устных уведомлениях). Кроме того, наличие механизма подтверждения прочтения дисциплинирует сотрудников своевременно реагировать на новые объявления, зная, что руководство видит статус ознакомления.

## 2.5 Прототипирование системы

На этапе прототипирования была создана рабочая прототипная версия веб-приложения, чтобы убедиться в корректности принятых проектных решений и продемонстрировать основной функционал заказчику (руководству ООО «ТДМ»). Прототип реализован с использованием выбранного стека технологий и включает ключевые модули системы в упрощённом виде.

Основные возможности, реализованные в прототипе:

* **Регистрация и подтверждение пользователей:** была добавлена возможность регистрации нового пользователя с указанием email и пароля. После регистрации прототип отправляет письмо на указанный адрес с уникальной ссылкой для подтверждения почты. Таким образом, сразу проверяется работа интеграции с почтовым сервером (SMTP Gmail) и логика активации учётной записи. Например, в прототипе зарегистрированный пользователь получает письмо «Подтвердите регистрацию в ООО “ТДМ”» со ссылкой вида https://.../confirm/<token>, при переходе по которой система помечает учётную запись как подтверждённую.
* **Аутентификация и сессии:** реализован вход пользователя (сотрудника или администратора) по email и паролю с созданием сессии. Прототип проверяет, подтверждён ли email пользователя (требуя сначала активации) и корректность пары логин/пароль. В случае успеха устанавливается сессионный идентификатор, позволяющий защищать внутренние страницы.
* **Восстановление пароля:** прототип поддерживает сценарий «Забыли пароль». Пользователь может ввести свой email, после чего система генерирует токен сброса и отправляет письмо со ссылкой вида .../reset/<token>. Перейдя по ссылке, пользователь попадает на страницу установки нового пароля. Данный функционал повышает надёжность системы и был протестирован на этапе прототипирования.
* **Роли и раздельные панели:** в прототипе учтены разные роли пользователей. В зависимости от роли после входа происходило перенаправление: администратор – на административную панель, новостной администратор (контент-менеджер) – на панель новостного администратора, рядовой сотрудник – на обычную страницу объявлений. В целях прототипа содержимое панелей администратора и новостного администратора было упрощено до приветственного сообщения и ссылки «Выйти», но сама возможность разграничения интерфейса по ролям проверена.
* **Страницы объявлений:** в прототип заложена структура страниц списка объявлений и деталей объявления. Были созданы шаблоны страниц: список объявлений (пока без фактических данных, с тестовыми примерами) и страница просмотра объявления. Это позволило проверить работоспособность маршрутизации (URL-ы /dashboard для сотрудника, разные шаблоны для ролей) и базовую верстку с Bootstrap. Пока функционал создания/редактирования объявлений в прототипе не реализовывался (предполагалось добавить на этапе полной реализации), однако предусмотрены заглушки.
* **Выход из системы:** реализована возможность корректного выхода (logout) – очистка сессии и перенаправление на страницу входа. Протестировано, что после выхода защищённые URL недоступны без повторной авторизации.

Прототип был развернут на тестовом сервере (в облаке Render.com) по адресу, доступному для проверки. Это позволило проверить работу всех интеграций во внешней среде – отправку писем через SMTP, работу сессий в реальном браузере, адаптивность интерфейса на разных устройствах. По результатам тестирования прототип подтвердил жизнеспособность архитектурных решений: выбранный стек технологий корректно работает совместно, все важные сценарии (регистрация, вход, восстановление доступа, разграничение прав) успешно выполняются.

Создание прототипа также помогло собрать предварительную обратную связь от нескольких сотрудников компании «ТДМ», которые протестировали интерфейс. В частности, была высоко оценена простота процесса входа и восстановления пароля. Выявленные мелкие замечания (например, необходимость более чёткого сообщения о успешном сбросе пароля, пожелания по оформлению) были учтены перед полной реализацией.

Таким образом, прототипирование позволило минимизировать риски на этапе разработки – убедиться, что все основные модули работают как задумано, и что пользовательский интерфейс понятен целевой аудитории. Полученный прототип лег в основу финальной системы и облегчил её последующую доработку и внедрение.

## Выводы по главе 2

Во второй главе дипломной работы разработан проект веб-приложения для информирования персонала ООО «ТДМ». Были определены функциональные возможности системы и представлены в виде диаграммы прецедентов, на которой отражены все основные пользователи (администратор и сотрудники) и сценарии их взаимодействия с приложением. На основе требований и анализа технологий обоснован выбор инструментальных средств: выбран язык Python и фреймворк FastAPI для серверной части, СУБД MySQL для хранения данных, а также веб-технологии (HTML, CSS, Bootstrap, JavaScript) для реализации клиентской части. Спроектирована структура базы данных в виде ER-диаграммы, включающей все необходимые сущности (пользователи, объявления, рассылки и др.) и связи между ними, что обеспечивает целостность и эффективность хранения информации. Определена архитектура системы по модели «клиент-сервер» с чётким разделением на слой данных, серверного приложения и пользовательского интерфейса; разработана схема взаимодействия компонентов, учитывающая требования масштабируемости и безопасности. Выполнено подробное проектирование пользовательского интерфейса: описаны макеты всех основных экранов для администратора и сотрудников, предусмотрены механизмы подтверждения прочтения объявлений, удобная навигация и адаптивность под разные устройства.

Все принятые проектные решения направлены на то, чтобы разработанное приложение полностью отвечало потребностям компании «ТДМ» в оперативном и контролируемом информировании персонала. По результатам проектирования система готова к реализации: сформирована полноценная архитектурно-техническая основа, позволяющая приступить к непосредственной разработке и внедрению веб-приложения в инфраструктуре предприятия.

**Экономическое обоснование разработки веб-приложения**

**1. Определение трудоёмкости разработки**

Для разработки веб-приложения на платформе Render принят состав исполнителей: один программист (разработчик). Длительность разработки (Дрм) составляет 6 месяцев непрерывной работы одного разработчика. В рамках дипломного проекта это эквивалентно трудоёмкости порядка 6 человеко-месяцев. Предположим, что средняя месячная норма составляет 22 рабочих дня при 8 часах работы в день. Тогда общий фонд рабочего времени на проект:

T = 6 \* 22 \* 8 = 1056 чел.-час.

Таким образом, трудоёмкость проекта оценивается как 1056 человеко-часов (что соответствует 6 месяцам работы одного специалиста).

**2. Расчёт заработной платы по видам работ и ролям**

Все работы по проектированию и разработке выполнялись одним специалистом, последовательно выступающим в роли аналитика, дизайнера, разработчика и тестировщика. Для оценки затрат разобьём процесс разработки на этапы (виды работ) и определим затраты труда и зарплаты по каждому этапу. Предположим, что месячный оклад разработчика (Ок) составляет 45 000 руб. (среднерыночный уровень оплаты труда программиста). Плановая разбивка трудоёмкости по видам работ представлена в таблице 1.

**Таблица 1 – Распределение трудоёмкости и затрат по заработной плате**

| **Вид работ** | **Трудоёмкость, мес** | **Затраты по зарплате, руб** |
| --- | --- | --- |
| Предпроектное обследование и анализ требований | 0.5 | 22 500 |
| Проектирование архитектуры и дизайна системы | 1.0 | 45 000 |
| Реализация (кодирование программного кода) | 3.0 | 135 000 |
| Тестирование и отладка | 1.0 | 45 000 |
| Внедрение и настройка | 0.5 | 22 500 |
| **Итого** | **6.0** | **270 000** |

Из таблицы видно, что общая продолжительность разработки – 6 месяцев, а суммарные затраты на зарплату составляют ЗПобщ = 270 000 руб. Данный результат можно получить расчётом по формуле трудовых затрат на разработку:

ЗП = Дрм \* Ок.

Подставив значения длительности разработки Дрм = 6 мес. и оклада Ок = 45 000 руб/мес, получим:

ЗП = 6 \* 45 000 = 270 000 руб.

Таким образом, фонд оплаты труда разработчика (прямые затраты на заработную плату) за весь период разработки составляет 270 000 руб.

**3. Начисления на заработную плату (30%)**

Помимо прямой заработной платы, работодатель несёт расходы по обязательным социальным отчислениям – в пенсионный фонд, фонд социального страхования, медицинского страхования и т.п. В сумме эти начисления условно составляют 30% от фонда оплаты труда. Расчёт дополнительных начислений производится по формуле:

Озп = ЗПобщ \* 0.30.

Где ЗПобщ = 270 000 руб – общая сумма выплат по заработной плате. Тогда величина начислений равна:

Озп = 270 000 \* 0.30 = 81 000 руб.

Таким образом, отчисления на социальные нужды составляют 81 000 руб. Это дополнительные расходы работодателя сверх выплаченной заработной платы.

**4. Амортизация оборудования и нематериальных активов**

Разработка программного обеспечения требует использования оборудования и программных средств, стоимость которых частично относится на себестоимость проекта посредством амортизационных отчислений. В соответствии с методическими рекомендациями примем нормативные годовые ставки амортизации: для оборудования – 6% в год, для нематериальных активов (ПО, лицензии) – 3% в год.

Для реализации проекта считаем, что использовалось:

* **Оборудование** – персональный компьютер разработчика стоимостью Соб = 100 000 руб.
* **Нематериальные активы** – программное обеспечение, системы разработки и прочие лицензии суммарной стоимостью Сна = 30 000 руб.

Амортизационные отчисления рассчитываются исходя из стоимости и срока использования актива в рамках проекта. Поскольку разработка длится 6 месяцев (0,5 года), амортизация начисляется за полгода.

Амортизация оборудования за период разработки:

Аоб = Соб \* 0.06 \* 0.5 = 100 000 \* 0.06 \* 0.5 = 3 000 руб.

Амортизация нематериальных активов за период разработки:

Ана = Сна \* 0.03 \* 0.5 = 30 000 \* 0.03 \* 0.5 = 450 руб.

Итого сумма амортизационных отчислений, отнесённых на проект, составляет 3 450 руб (3 000 + 450). Данные затраты относительно невелики, так как используются имеющиеся ресурсы, а нормативы амортизации низкие.

**5. Расчёт стоимости оборотных средств**

Для обеспечения процесса разработки и последующего функционирования системы предприятию необходимы оборотные средства (например, на покрытие незапланированных расходов, закупку мелких материалов, поддержку работоспособности системы до получения отдачи). Объём требуемых оборотных средств для данного проекта невелик ввиду преобладания трудовых затрат. Оценочно примем стоимость необходимого оборотного капитала равной 5 000 руб.

Данная сумма будет авансирована для поддержания текущей деятельности (например, на закупку канцелярии, оплату мелких сервисов, первоначальное наполнение системы контентом и пр.) и включается в расчёт экономических показателей проекта.

**6. Текущие затраты на эксплуатацию**

После внедрения программного продукта возникают постоянные (ежемесячные) затраты на его эксплуатацию и поддержку. К основным текущим затратам относятся расходы на инфраструктуру и ресурсы:

* **Хостинг** приложения на платформе Render – около 2 000 руб в месяц (тариф сервера и сопутствующие сервисы). За год эксплуатации это составит ~24 000 руб.
* **Интернет-трафик** – дополнительная нагрузка на канал интернет-связи предприятия оценивается в 1 000 руб в месяц. За год ~12 000 руб.
* **Электроэнергия** – потребление электроэнергии на работу оборудования (серверов, сетевого оборудования, рабочего места разработчика) оценивается примерно в 500 руб в месяц. В год ~6 000 руб.
* **Коммунальные услуги** (отопление, водоснабжение, прочие услуги содержания офиса) – дополнительные затраты условно 500 руб в месяц, в год ~6 000 руб.

Суммарно текущие эксплуатационные затраты на работу системы составляют:

Зтек = 24 000 + 12 000 + 6 000 + 6 000 = 48 000 руб в год.

Эта сумма представляет ежегодные расходы на обеспечение функционирования внедрённого веб-приложения.

**7. Прочие затраты**

Помимо перечисленных статей, в процессе создания и внедрения системы могут возникать прочие расходы. К ним относятся командировочные и представительские расходы, затраты на обучение персонала работе с системой, канцелярские и административно-хозяйственные мелочи и т.д. Обычно такие затраты оценивают в размере 5–10% от фонда оплаты труда. Примем оценочно прочие затраты на уровне 5% от общей суммы заработной платы.

Расчёт прочих затрат:

Пр = ЗПобщ \* 0.05 = 270 000 \* 0.05 = 13 500 руб.

Таким образом, прочие непредвиденные и дополнительные расходы составляют около 13,5 тыс. руб. (5% от фонда оплаты труда проекта).

**8. Накладные расходы**

Накладные (общехозяйственные) расходы включают затраты на управление и обслуживание процесса разработки: оплата труда административно-управленческого персонала, аренда офиса, бухгалтерское сопровождение, амортизация общехозяйственного оборудования, охрана и т.д. Данные расходы распределяются на проект косвенно, обычно в размере 3–5% от прямых затрат на оплату труда.

В расчётах примем норматив накладных расходов 5% от фонда оплаты труда. Тогда сумма накладных расходов, приходящаяся на проект, составит:

Нр = ЗПобщ \* 0.05 = 270 000 \* 0.05 = 13 500 руб.

Таким образом, накладные расходы, отнесённые на проект, оценочно равны 13,5 тыс. руб. (5% от ФОТ), что отражает долю общих затрат предприятия, приходящуюся на данную разработку.

**9. Калькуляция полной себестоимости**

На основании проведенных расчётов составим калькуляцию полной себестоимости разработки веб-приложения, включающей все статьи затрат. Итоговые данные сведены в таблицу 2.

**Таблица 2 – Полная себестоимость разработки и внедрения системы**

| **Статья затрат** | **Сумма, руб.** |
| --- | --- |
| 1. Заработная плата разработчика | 270 000 |
| 2. Начисления на зарплату (30%) | 81 000 |
| 3. Амортизация оборудования (6% годовых) | 3 000 |
| 4. Амортизация нематериальных активов (3% годовых) | 450 |
| 5. Оборотные средства | 5 000 |
| 6. Хостинг (1 год эксплуатации) | 24 000 |
| 7. Интернет-трафик (1 год эксплуатации) | 12 000 |
| 8. Электроэнергия (1 год) | 6 000 |
| 9. Коммунальные услуги (1 год) | 6 000 |
| 10. Прочие расходы (5% от ФОТ) | 13 500 |
| 11. Накладные расходы (5% от ФОТ) | 13 500 |
| **Итого** | **434 450** |

Полная совокупная стоимость разработки, внедрения и годичной эксплуатации системы (инвестиционные затраты) составляет таким образом примерно 434,45 тыс. руб. (округленно ~434 тыс. руб.).

**10. Расчёт экономической эффективности**

Экономическая эффективность проекта проявляется в снижении трудозатрат персонала при внутреннем информировании сотрудников за счёт автоматизации. Согласно условию, системой будут пользоваться 40 сотрудников, каждый из которых экономит около 10 минут рабочего времени в день благодаря автоматизированному доступу к информации. При средней стоимости рабочего времени 300 руб/час можно определить годовую экономию трудовых затрат в денежном выражении.

Рассчитаем экономию от автоматизации:

* Экономия времени одним сотрудником в день: 10 мин = 1/6 часа.
* Стоимость сэкономленного времени одним сотрудником: 300 \* (10/60) = 50 руб в день.
* Экономия для всех сотрудников в день: 50 \* 40 = 2 000 руб.
* Экономия за год (приблизительно 250 рабочих дней): 2 000 \* 250 = 500 000 руб.

Таким образом, ориентировочная годовая экономия затрат труда за счёт внедрения системы составляет Эгод ≈ 500 тыс. руб. в год.

Однако при эксплуатации системы предприятие несёт дополнительные ежегодные расходы (Зтек) на её поддержание (хостинг, электричество и прочее), рассчитанные ранее – около 48 000 руб. в год. Эти затраты необходимо вычесть из получаемой экономии, чтобы определить чистый экономический эффект.

Годовой экономический эффект от внедрения (Эг) рассчитывается как разница между экономией и дополнительными расходами:

Эг = Эгод – Зтек = 500 000 – 48 000 = 452 000 руб.

Полученный положительный эффект показывает, на какую величину в год снизятся совокупные затраты предприятия благодаря новому решению после покрытия расходов на его эксплуатацию.

Далее рассчитаем **срок окупаемости проекта**, то есть период, за который суммарная экономия компенсирует вложенные в разработку средства. Срок окупаемости (Ток) определяем отношением совокупных затрат на проект (инвестиций) к годовому экономическому эффекту:

Ток = С / Эг.

Где С = 434 450 руб – полный объём инвестиций (себестоимость разработки и внедрения), Эг = 452 000 руб/год – годовой экономический эффект. Подставив значения:

Ток = 434 450 / 452 000 ≈ 0,96 года.

Таким образом, проект окупается менее чем за 1 год. В переводе на календарный период это примерно 11,5 месяцев. Примем, что ориентировочный срок окупаемости составляет около 11 месяцев. Это означает, что уже через чуть менее года после запуска система сэкономит средств столько же, сколько было затрачено на её создание.

**11. Итоговые показатели и выводы**

На основе проведенных расчетов можно сделать вывод, что разработка и внедрение внутреннего веб-приложения для информирования сотрудников экономически оправданы. Ключевые технико-экономические показатели проекта приведены в таблице 3.

**Таблица 3 – Основные экономические показатели проекта**

| **Показатель** | **Значение** |
| --- | --- |
| Трудоёмкость разработки | 6 мес. (≈1056 чел.-час) |
| Полная стоимость разработки и внедрения | 434 450 руб. (~434 тыс. руб) |
| Годовая экономия от внедрения системы | 500 000 руб. |
| Годовые эксплуатационные затраты | 48 000 руб. |
| Годовой экономический эффект, Эг | 452 000 руб. |
| Срок окупаемости проекта | ~0,96 года (≈11 месяцев) |

Как видно из расчётов, единовременные затраты на создание системы (около 434 тыс. руб.) сопоставимы с годовым экономическим эффектом (~452 тыс. руб.). За счёт экономии времени сотрудников в размере ~500 тыс. руб. в год проект окупается примерно за 11 месяцев, после чего начинается чистая экономия средств предприятия. В целом, внедрение разработанного веб-приложения позволит повысить эффективность внутренних коммуникаций и сэкономить компании около полумиллиона рублей ежегодно, что подтверждает экономическую целесообразность данного дипломного проекта.