Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Факультет информационных технологий и анализа больших данных

Департамент анализа данных и машинного обучения

Выпускная квалификационная работа

на тему: «Разработка веб-приложения для организации совместной работы над проектами»

Направление подготовки 09.03.03 «Прикладная информатика»,

Профиль «ИТ-сервисы и технологии обработки данных в экономике и финансах»

Выполнил студент группы ПИ21-3

Балашкин Андрей Михайлович \_\_\_\_\_\_\_\_\_

Руководитель к.т.н., доцент

Хасанов Ильнур Ильдароваич \_\_\_\_\_\_\_\_\_

**ВКР соответствует предъявленным требованиям**

Руководитель Департамента

к.т.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.А. Петросов

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г.

Москва 2025**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 3](#_Toc1)

[1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc2)

[1.1 Существующие решения 5](#_Toc3)

[1.2 Определение потребностей пользователей 9](#_Toc4)

[1.3 Составление требований 13](#_Toc5)

[Функциональные требования 13](#_Toc6)

[1.3.3 Технические требования 16](#_Toc7)

[1.3.4 Итоговые выводы 16](#_Toc8)

[2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 18](#_Toc9)

[3. ПРАКТИЧЕСКАЯ ЧАСТЬ 19](#_Toc10)

[ЗАКЛЮЧЕНИЕ 20](#_Toc11)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 21](#_Toc12)

# ВВЕДЕНИЕ

В современном мире цифровых технологий коллективная работа над проектами становится неотъемлемой частью рабочих процессов во многих сферах деятельности. С развитием удаленной занятости, распределенных команд и гибких методологий управления проектами возрастает потребность в удобных, функциональных и интуитивно понятных инструментах для совместного планирования, координации задач и обмена информацией.

Существующие решения, такие как Trello, Asana, Jira и другие, предоставляют широкий набор возможностей для управления проектами. Однако многие из них либо сложны в освоении для пользователей без технического опыта, либо имеют ограничения по функционалу в бесплатных версиях. Кроме того, универсальные платформы не всегда учитывают специфические потребности небольших команд и стартапов, которым требуется гибкость в настройке рабочего пространства.

Актуальность исследования обусловлена необходимостью создания удобного и доступного инструмента для командной работы, который обеспечит базовые функции управления проектами: ведение списка задач (todo-лист), работу с канбан-доской, создание и хранение текстовых записей, а также возможность добавления участников с равными правами. Такой инструмент поможет небольшим коллективам эффективно распределять задачи, отслеживать прогресс и централизованно хранить информацию.

Цель работы — разработка веб-приложения для организации совместной работы над проектами, обеспечивающего простоту в управлении задачами и удобство взаимодействия пользователей.

Для достижения этой цели необходимо решить следующие задачи:

* Провести анализ существующих решений и выявить их преимущества и недостатки.
* Определить потребности целевой аудитории и сформировать требования к разрабатываемому приложению.
* Изучить принципы проектирования веб-приложений и обосновать выбор используемых технологий.
* Реализовать серверную и клиентскую части веб-приложения.
* Провести тестирование разработанного решения и оценить его эффективность.

Объект исследования — процессы организации совместной работы над проектами с использованием цифровых инструментов.

Предмет исследования — методы и технологии разработки веб-приложений для управления проектами.

Таким образом, данная работа представляет собой комплексное исследование, включающее анализ предметной области, выбор оптимальных решений и реализацию программного продукта, который может быть использован в практической деятельности небольших команд и проектных групп.

# 1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

## 1.1 Существующие решения

В последние годы цифровые инструменты для управления проектами и совместной работы получили широкое распространение. Развитие удаленной работы, рост популярности гибких методологий управления (Agile, Kanban, Scrum) и необходимость организовывать рабочий процесс в распределенных командах привели к появлению множества платформ, предлагающих разнообразные функции.

Однако далеко не все решения одинаково удобны и подходят для небольших команд. Многие системы перегружены сложными настройками, требуют длительного освоения или содержат ограничения в бесплатных версиях. Для команд, которым важно простое и интуитивное управление задачами, ведение заметок и использование канбан-доски, избыточные функции могут быть скорее помехой, чем преимуществом.

В данном разделе рассмотрены наиболее популярные инструменты, их ключевые возможности, а также их преимущества и недостатки с точки зрения простоты и удобства.

**Trello**

Trello — одна из самых известных платформ для управления задачами, основанная на методологии Kanban. В основе системы лежит концепция доски, состоящей из списков, в которые добавляются карточки задач. Пользователи могут перемещать карточки между списками, отражая текущий статус задачи. В Trello можно настраивать теги, сроки выполнения, добавлять комментарии, вложения и чек-листы, а также назначать участников на задачи. Кроме того, сервис поддерживает автоматизацию рабочих процессов с помощью встроенного инструмента Butler и интеграции с популярными сервисами, такими как Google Drive и Slack.

Преимущества:

* Простота использования и интуитивно понятный интерфейс.
* Поддержка совместной работы с возможностью комментирования и добавления вложений.
* Бесплатный тариф с основными функциями.

Недостатки:

* Ограниченные возможности в бесплатной версии (например, малое количество автоматизаций).
* Отсутствие встроенных инструментов для текстовых заметок и комплексного управления проектами.

**Asana**

Asana предназначена для управления проектами и задачами, предлагая пользователям несколько режимов отображения: список, канбан-доску и календарь. Система позволяет устанавливать приоритеты, статусы, дедлайны и назначать ответственных. Также в Asana предусмотрены механизмы автоматизации, интеграция с популярными сервисами и возможность командного обсуждения задач.

Преимущества:

* Гибкость в настройке задач и проектов.
* Поддержка интеграций с другими сервисами (Google Drive, Slack и др.).
* Возможность автоматизации процессов.

Недостатки:

* Сложность освоения для новых пользователей.
* Ограниченный функционал в бесплатной версии.

**Jira**

Jira — мощный инструмент, ориентированный на IT-команды и разработчиков, использующих методологии Agile и Scrum. Платформа предлагает гибкие настройки рабочих процессов, поддержку Scrum- и Kanban-досок, а также развитую систему отчетности и аналитики. Jira позволяет настраивать различные этапы выполнения задач, устанавливать зависимости между ними и интегрировать их с DevOps-инструментами.

Преимущества:

* Гибкость настройки рабочих процессов.
* Поддержка Scrum- и Kanban-досок.
* Развитые функции отчетности и аналитики.

Недостатки:

* Высокий порог входа и сложность интерфейса.
* Платность большинства функций, необходимых для полноценной работы.

**Notion**

Notion сочетает в себе инструменты для создания текстовых записей, ведения базы данных, управления задачами и организации информации. В системе можно создавать страницы с текстами, таблицами и списками задач, а также связывать их друг с другом. Notion предлагает возможность совместного редактирования, что делает его удобным для командной работы.

Преимущества:

* Универсальность: поддержка todo-листов, таблиц, документов и баз данных.
* Гибкость в настройке интерфейса под нужды пользователя.
* Возможность совместного редактирования контента.

Недостатки:

* Ограниченный функционал в бесплатной версии.
* Возможные проблемы с производительностью при работе с большими объемами данных.

Существующие решения обладают широкими возможностями, однако многие из них либо ориентированы на специфические задачи, либо требуют сложной настройки, либо имеют ограничения в бесплатных версиях. В рамках данной работы разрабатывается веб-приложение, сочетающее ключевые функции для совместной работы: todo-лист, канбан-доску и текстовые заметки, при этом обеспечивая удобство использования и доступность.

## 1.2 Определение потребностей пользователей

Совместная работа над проектами требует четкой организации, эффективного распределения задач и удобных инструментов для коммуникации между участниками. В последние годы наблюдается тенденция к упрощению процессов управления проектами, особенно в небольших командах, где важнее оперативность и легкость взаимодействия, чем сложные механизмы контроля и распределения прав.

Многие современные решения предлагают широкий функционал, включая настройки доступа, детализированные роли пользователей и сложные механизмы синхронизации. Однако эти функции не всегда востребованы, особенно в небольших группах, стартапах и неформальных командах, где участники работают на равных условиях. В таких случаях чрезмерно сложные системы могут скорее затруднять процесс, чем упрощать его.

Разрабатываемое приложение ориентировано на удобство, простоту и интуитивность. Оно должно предоставить пользователям базовый, но достаточный набор функций для эффективной организации совместной работы, без излишней сложности. Для понимания потребностей пользователей рассмотрим ключевые аспекты их работы и проблем, с которыми они сталкиваются.

Основные аспекты организации совместной работы

**Управление задачами и контроль выполнения**

Одной из основных потребностей пользователей является удобная система управления задачами. Независимо от типа проекта, его успешное выполнение требует четкого планирования и разделения задач между участниками. При этом не всегда необходим сложный механизм распределения ролей и строгая иерархия — во многих случаях достаточно удобного интерфейса, позволяющего быстро добавлять и изменять задачи.

Для эффективного управления задачами важно учитывать несколько факторов:

* Простота создания и редактирования задач. Пользователи не должны тратить время на сложные настройки — задача должна добавляться в несколько кликов.
* Гибкость организации задач. Разделение на категории, статусы («Ожидание», «В работе», «Завершено») и возможность визуального представления (список или канбан-доска) упрощает контроль за процессом.
* Минимум обязательных полей. В сложных системах пользователи часто вынуждены заполнять множество параметров перед созданием задачи. В простых решениях это не требуется, что делает процесс более быстрым.

**Совместная работа без сложных механизмов распределения ролей**

В небольших командах часто нет необходимости в сложной системе прав доступа. Все участники проекта работают на равных условиях, и строгая иерархия только усложняет процесс. В этом случае важно предоставить возможность каждому пользователю редактировать задачи, вносить изменения и добавлять новые записи без необходимости запрашивать доступ.

Преимущества отказа от сложных настроек доступа:

* Упрощенная работа: пользователи сразу получают полный доступ ко всем возможностям без необходимости настройки прав.
* Гибкость: любой участник может взять на себя любую задачу или внести изменения в проект.
* Отсутствие необходимости администрирования: в небольших командах часто нет отдельного человека, отвечающего за управление правами, поэтому отказ от этих механизмов упрощает процесс.

**Минимальный порог входа и простота освоения**

Чем проще интерфейс, тем быстрее пользователи смогут начать работу. Многие современные системы требуют длительного обучения, поскольку содержат множество сложных элементов. Однако в небольших командах важнее интуитивно понятный интерфейс, который позволяет приступить к работе без предварительного изучения документации.

Ключевые принципы простого интерфейса:

* Минимализм. Никаких лишних кнопок, сложных меню и многоуровневых настроек.
* Интуитивность. Все основные функции доступны сразу, без необходимости искать их в скрытых разделах.
* Гибкость интерфейса. Пользователь должен сам выбирать удобный формат работы, например, переключаться между списком задач и канбан-доской.

**Ведение текстовых записей и централизованное хранение информации**

Помимо управления задачами, пользователи часто нуждаются в удобном инструменте для ведения заметок. Это особенно актуально в проектной работе, где важно фиксировать идеи, детали выполнения задач и другую информацию.

Основные требования к ведению заметок:

* Структурированность. Записи должны быть организованы так, чтобы их можно было легко найти.
* Простота редактирования. Встроенный текстовый редактор с базовыми функциями форматирования поможет упорядочить информацию.
* Связь с задачами. Важно, чтобы заметки можно было легко ассоциировать с определенными задачами или этапами проекта.

**Доступность с различных устройств**

Современные пользователи работают не только на компьютерах, но и на мобильных устройствах. Поэтому важно, чтобы веб-приложение корректно отображалось на разных экранах.

Ключевые аспекты кроссплатформенности:

* Адаптивный дизайн. Приложение должно корректно отображаться на мобильных устройствах без потери функциональности.
* Упрощенный интерфейс для мобильных пользователей. Некоторые элементы управления могут отличаться на мобильных устройствах, чтобы повысить удобство работы.

Несмотря на широкий выбор инструментов для совместной работы, пользователи часто сталкиваются с рядом проблем:

* Сложность интерфейса. Многие платформы требуют времени на освоение из-за перегруженного интерфейса.
* Лишние функции. Во многих сервисах присутствуют сложные механизмы отчетности, настройки ролей и другие функции, которые не всегда нужны небольшим командам.
* Ограничения бесплатных версий. Популярные сервисы, такие как Trello и Asana, в бесплатных версиях имеют серьезные ограничения.
* Сложность интеграции. Некоторые платформы требуют настройки интеграций с другими сервисами, что усложняет использование.

На основе анализа потребностей пользователей можно сформулировать ключевые требования к разрабатываемому приложению:

1. Простая система управления задачами без сложных настроек.
2. Удобное ведение заметок, связанных с проектом.
3. Минималистичный и интуитивно понятный интерфейс.
4. Возможность совместной работы без распределения ролей.
5. Доступность на различных устройствах.

Таким образом, разрабатываемое приложение ориентировано на простоту и удобство. Оно будет полезно небольшим командам и проектным группам, которым важна легкость работы без сложных административных функций.

## 1.3 Составление требований

При разработке любого программного продукта важно определить его функциональные и нефункциональные требования. Они помогают задать границы системы, определить ее основные возможности и задать критерии оценки качества. Разрабатываемое веб-приложение ориентировано на простоту и удобство использования, поэтому требования к нему формируются с учетом минимализма интерфейса и удобства работы.

Требования можно разделить на две основные группы: функциональные (определяющие, что система должна делать) и нефункциональные (задающие ограничения и характеристики работы системы).

**Функциональные требования**

Функциональные требования описывают конкретные возможности веб-приложения, которые обеспечивают выполнение его основной задачи — организации совместной работы над проектами.

1. **Управление задачами (todo-лист и канбан-доска)**

Одной из ключевых функций приложения является организация задач. Пользователи должны иметь возможность создавать, редактировать и удалять задачи, а также перемещать их между статусами. Приложение должно поддерживать:

* Создание новых задач с возможностью ввода названия и описания.
* Редактирование уже существующих задач.
* Удаление задач при их завершении или ошибочном создании.
* Упорядочивание задач по статусам (например, «Ожидание», «В работе», «Завершено»).
* Перетаскивание задач между статусами на канбан-доске для удобной организации рабочего процесса.

1. **Ведение текстовых записей**

Помимо задач, пользователи часто нуждаются в возможности фиксировать текстовую информацию, например, идеи, комментарии или планы. В связи с этим в приложении должна быть реализована система текстовых заметок, включающая:

* Возможность создания новых записей.
* Форматирование текста (заголовки, списки, выделение жирным, курсивом).
* Редактирование и удаление записей.
* Поиск по содержимому записей для быстрого доступа к нужной информации.

1. **Совместный доступ к проекту**

Так как приложение предназначено для командной работы, в нем должна быть возможность приглашения других пользователей в проект. Однако, в отличие от сложных систем, здесь не будет распределения ролей — все участники будут обладать равными правами. Приложение должно позволять:

* Добавлять новых участников в проект.
* Удалять участников (администратором проекта).
* Все пользователи проекта должны иметь равные права на редактирование задач и заметок.

1. **Простой механизм авторизации и входа в систему**

Для работы с проектами пользователи должны проходить регистрацию и вход в систему. Приложение должно поддерживать:

* Регистрацию новых пользователей с вводом email и пароля.
* Авторизацию с проверкой учетных данных.
* Восстановление пароля через email.

1. **Просмотр и управление проектами**

Каждый пользователь может создавать и управлять проектами. Функции проекта включают:

* Создание нового проекта.
* Редактирование названия и описания проекта.
* Удаление проекта (возможность доступна только создателю).
* Просмотр списка доступных проектов.

**Нефункциональные требования**

Нефункциональные требования описывают характеристики работы системы, такие как удобство интерфейса, быстродействие и безопасность данных.

1. **Простота и удобство интерфейса**

Приложение ориентировано на минимализм и легкость использования, поэтому его интерфейс должен быть интуитивно понятным и не перегруженным лишними элементами. Требования к интерфейсу:

* Все основные функции должны быть доступны в один-два клика.
* Минимальное количество обязательных полей при создании задач и записей.
* Отсутствие сложных настроек, которые могут затруднить работу пользователя.

1. **Производительность и быстродействие**

Для комфортной работы пользователей система должна быть отзывчивой и обеспечивать быструю загрузку интерфейса. Основные требования:

* Загрузка страниц и данных должна происходить менее чем за 2 секунды.
* Взаимодействие с задачами (создание, редактирование, перемещение) должно выполняться без значительных задержек.
* Минимальное потребление ресурсов браузера.

1. **Кроссплатформенность**

Приложение должно корректно работать на различных устройствах, включая настольные компьютеры, ноутбуки и мобильные устройства. Для этого требуется:

* Адаптивный дизайн, подстраивающийся под экраны разного размера.
* Корректная работа как в настольных, так и в мобильных браузерах.

1. **Надежность и отказоустойчивость**

Приложение должно обеспечивать стабильную работу даже при высокой нагрузке или сбоях. Требования к надежности:

* Минимизация потерь данных при сбоях.
* Автоматическое сохранение изменений в задачах и заметках.
* Обработка ошибок с информативными сообщениями пользователю.

1. **Безопасность данных**

Хотя приложение не использует сложные механизмы прав доступа, оно должно обеспечивать базовые меры защиты данных. Основные меры безопасности:

* Хранение паролей в зашифрованном виде.
* Использование HTTPS для защиты данных при передаче.
* Ограничение количества попыток входа в систему для защиты от подбора пароля.

Определение требований является важным этапом разработки, позволяющим четко обозначить цели и функциональность системы. Разрабатываемое веб-приложение должно предоставлять пользователям удобные инструменты для управления задачами и заметками, сохраняя при этом простоту интерфейса и легкость освоения.

Функциональные требования определяют, какие именно возможности будут доступны пользователям: работа с задачами, текстовыми записями, совместное редактирование и управление проектами. Нефункциональные требования задают параметры работы системы, такие как удобство интерфейса, производительность, безопасность и отказоустойчивость.

Сбалансированное сочетание этих требований обеспечит удобство работы пользователей и позволит создать легкое, но функциональное решение для совместной работы над проектами.

# 2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

# 3. ПРАКТИЧЕСКАЯ ЧАСТЬ

# **ЗАКЛЮЧЕНИЕ**

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Мартин Роберт К. Чистый код. Создание анализ и рефакторинг. СПб: Питер, 2019. 464 с.
2. Django documentation [Электронный ресурс] // django. URL: <https://docs.djangoproject.com/en/4.2/> (дата обращения 10.03.2024)
3. React documentation [Электронный ресурс] // react. URL: <https://react.dev/learn> (дата обращения 25.03.2024)
4. Мартин Роберт К. Чистая архитектура. Искусство разработки программного обеспечения СПб: Питер, 2021 — 352 с.
5. Стоян Стефанов React.js. Быстрый старт. — СПб: Питер, 2017. — 304 с.