Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»

Институт радиоэлектроники и информационных технологий – РТФ

Школа бакалавриата

ОТЧЕТ

По проекту
«Разработка платформы для управления задачами команды разработки»

по дисциплине «Проектный практикум»

Заказчик:

Макаров А. В.

Куратор:

Макаров А. В.

Студенты команды «Спринтеры»

Вышедко В. Е.

Маклюкова А. А.

Бочанова М. В.

Игнатьев А. А.

Демчук А. Д.

Екатеринбург, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель проекта: создать удобную платформу для управления задачами команды разработки с поддержкой Kanban, аналитики эффективности и интеграций с DevOps-инструментами.

Задачи:

1) организовать работу в команде,

2) провести аналитику,

3) придумать дизайн интерфейса,

4) сверстать страницы,

5) написать код,

6) протестировать продукт,

7) защитить проект.

В современных организациях, особенно в условиях распределённых команд и гибридных форматов работы, управление проектами сталкивается с рядом системных сложностей. Компании используют множество инструментов (Trello, Jira, Slack, Excel), которые не интегрированы между собой, а процессы коммуникации и контроля задач остаются фрагментированными. Это приводит к потере прозрачности, нарушению сроков и снижению эффективности совместной работы.

Причины возникновения проблемы:

– отсутствие единой платформы для управления задачами, коммуникации и отчётности;

– данные дублируются в разных системах, что увеличивает риск ошибок и временные затраты на синхронизацию;

– каждая команда использует собственные подходы к постановке задач и контролю дедлайнов;

– нет единых метрик для оценки прогресса, что затрудняет планирование ресурсов;

– информация передаётся через несколько каналов (почта, мессенджеры, встречи), что замедляет принятие решений;

– критические обновления задач теряются в потоке сообщений.

По данным исследования McKinsey (2023):

47% проектов в компаниях среднего размера выходят за рамки бюджета из-за несвоевременного контроля этапов.

32% рабочего времени сотрудники тратят на поиск информации и согласование данных между системами.

Последствия для бизнеса:

– увеличение времени вывода продуктов на рынок на 15-20%,

– рост операционных затрат на поддержку множества инструментов (в среднем $12 тыс./год на команду из 10 человек).

Последствия для сотрудников:

– снижение удовлетворённости из-за рутинных ручных операций (составление отчётов вручную, дублирование задач).

Целевая аудитория, заинтересованная в решении:

– руководители проектов,

– участники команд,

– топ-менеджмент.

По завершении проекта ожидается и планируется:

1) проведён анализ конкурентов,

2) разработано работающее приложение с базовым функционалом управления задачами,

3) реализованы базовые инструменты визуализации (диаграммы Ганта, Burndown chart),

4) обеспечена возможность отслеживания рабочих процессов и метрик команды.

1 Основная часть

1.1 Информация о работе каждого участника в отдельности

1.1.1 Информация о работе тимлида Вышедко В. Е.

Тимлид выполнял ключевую роль в организации работы команды, обеспечивая выполнение проекта в срок и соответствие требованиям заказчика. Основные обязанности включали планирование и распределение задач (составление бэклога, декомпозиция целей на спринты, распределение задач между членами команды), контроль выполнения (мониторинг прогресса через сообщения в мессенджерах, а также на еженедельных созвонах; корректировка планов при изменении приоритетов), коммуникация (взаимодействие с заказчиком и командой, обеспечение прозрачности процесса), мотивация команды (поддержка позитивной атмосферы, разрешение конфликтов, проведение ретроспектив для улучшения процессов).

Для управления проектом использовались спринты длительностью 1-2 недели с планированием, ревью и ретроспективами; еженедельные стендапы для синхронизации команды; Kaiten — для трекинга задач, установки дедлайнов и визуализации прогресса (доски Kanban, burn down чарты); Google Диск — документация требований и архитектуры; GitHub — CI/CD, управление репозиториями и код-ревью; Telegram — оперативная коммуникация.

В ходе проекта возникла проблема в виде недостаточного объема работы со стороны фронтэнд разработчика. Было принято решение в конце каждого дня спрашивать у него отчет о том, что было сделано по верстке страниц.

Итак, тимлид обеспечил слаженную работу команды и достижение целей проекта. Использование гибких методологий и инструментов позволило адаптироваться к изменениям.

1.1.2 Информация о работе аналитика Маклюковой А. А.

В рамках аналитической работы над проектом по разработке платформы для управления задачами команды разработки была выполнена комплексная подготовка всех ключевых элементов, необходимых для начала и координации разработки. В первую очередь была сформулирована цель проекта, проанализированы потребности целевой аудитории, определены ключевые роли и интересы всех заинтересованных сторон (стейкхолдеров). Проведен детальный анализ конкурентов — Jira, Trello, Asana и Kaiten — с выделением их сильных и слабых сторон, что позволило выявить конкурентные преимущества будущего продукта.

На основе анализа было разработано подробное техническое задание, включающее все функциональные и нефункциональные требования, архитектурные принципы, требования к интерфейсу и используемый стек технологий (Java Spring и JavaScript). В ТЗ отражены ключевые сценарии работы с тикетами, ролями, проектами, аналитикой, интеграциями и системой безопасности. Особое внимание уделено плану разработки, который был значительно расширен и адаптирован в соответствии с изменениями, предложенными заказчиком после презентации. Уточнены этапы проектирования, реализации MVP, расширения функциональности, тестирования и развертывания.

Также были подготовлены тексты для презентационных материалов, включая цели проекта, описание продукта, целевую аудиторию, сравнение с конкурентами и предлагаемое решение. Все результаты работы оформлены в структурированных документах, готовых к передаче команде разработки.

1.1.3 Информация о работе дизайнера Бочановой М. В.

В рамках проекта по разработке платформы для управления задачами команды разработки был выполнен полный цикл проектирования пользовательского интерфейса — от базовой стилистики до детализированных макетов основных разделов.

На первом этапе был разработан общий визуальный стиль платформы. Это включало в себя подбор основной цветовой схемы, соответствующей задачам продукта (контрастность, удобочитаемость, современный минимализм), а также определение типографики, обеспечивающей комфортное восприятие информации на любых устройствах.

Следующим шагом стало проектирование структуры и логики основных интерфейсов. Были созданы лэйауты ключевых страниц: планировщика задач, главной страницы (дашборда), страниц авторизации/регистрации, личного кабинета и раздела аналитики. На основе этих лэйаутов были разработаны детализированные макеты, отражающие пользовательские сценарии и логику взаимодействия.

Особое внимание было уделено разделу планировщика задач, который реализован в двух форматах: табличном и календарном. Табличный вид акцентирует внимание на плотной структуре данных, фильтрации и удобстве управления списком задач, а календарный — на визуальной расстановке задач по времени. Были проработаны сценарии добавления, редактирования, фильтрации задач по различным параметрам (статус, исполнитель, срок), а также визуальные индикаторы приоритетов и дедлайнов.

Страницы авторизации и регистрации выполнены в легком, понятном формат: реализована структура с логичной последовательностью шагов и отображением ошибок, а также возможностью восстановить пароль. Личный кабинет пользователя включает в себя настройки личных данных, пароля и уведомлений.

Отдельно была разработана главная страница — дашборд. Она представляет собой информационную сводку по последним действиям в проектах, приближающимся дедлайнам и общей загрузке команды, позволяя пользователю быстро ориентироваться в текущей ситуации.

Также был реализован аналитический раздел. Он позволяет отслеживать ключевые метрики для MVP проектов в двух форматах: краткая сводка и расширенный вид с графиками и диаграммами. Были учтены как потребности команды разработки, так и удобство восприятия информации конечными пользователями.

В течение работы проводились тестирования макетов на удобство использования и восприятие интерфейса. На основе результатов периодически вносились необходимые изменения и улучшения в дизайн.

В результате проделанной работы сформирована целостная визуальная и логическая система интерфейсов, подготовлены полные макеты ключевых страниц, создана основа для масштабирования и дальнейшего развития платформы.

1.1.4 Информация о работе фронтенд разработчика Демчука А. Д.

Фронтенд-разработчик отвечал за создание пользовательского интерфейса (UI), обеспечение интерактивности и интеграцию с бэкенд-сервисами. Основные задачи включали реализацию интерфейсов в соответствии с дизайн-макетами и организацию взаимодействия пользователя с системой через формы, фильтры и динамические элементы.

На основе макетов, разработанных в Figma, фронтенд-разработчик сверстал Dashboard (Главную панель). Была разработана адаптивная сетка для отображения списка задач, статистики и приоритетных задач и реализованы фильтры (по дате, приоритету, статусу) и сортировка с использованием React для управления состоянием.

Далее, Task (Управление задачами). Осуществлена возможность создания и редактирования задач, назначение задач другим пользователям. На карточках есть метки в виде цветовых тегов.

Следующие страницы – это Login/Sign up (Аутентификация). Реализованы регистрация и вход с валидацией. Была добавлена обработка ошибок (неверный email/пароль) с выводом уведомлений. Безопасность реализована через сохранение токена аутентификации.

Был также разработан Calendar (Календарь). Была выполнена реализация drag-and-drop перемещения задач между датами.

Работа была выполнена при помощи библиотеки React. Стили реализованы на языке CSS.

Работа фронтенд-разработчика обеспечила создание интуитивного и отзывчивого интерфейса.

1.1.5 Информация о работе бэкенд разработчика Игнатьева А. А.

Бэкенд-разработчик отвечал за проектирование и реализацию серверной части приложения, включая архитектуру базы данных, API, аутентификацию, интеграцию с фронтендом и деплой инфраструктуры. Основные задачи – это создание и оптимизация базы данных, реализация бизнес-логики и RESTful API, обеспечение безопасности данных (JWT-аутентификация), настройка CI/CD и развертывание приложения в production-среде.

Перед началом работы над проектом был выбран стек технологий. В качестве языка программирования взят Java с фреймворком Spring Boot. Maven является сборщиком.

Следующим этапом стало проектирование базы данных. Были определены ключевые сущности: User, Task, Project, Label, File. Далее между ними создали связи. Разработали архитектуру приложения и реализовали

базовую схему API. Инициализировали Git-репозиторий с модульной структурой проекта.

Началась работа над JWT-аутентификацией. Была написана генерация Access Token и Refresh Token. Для фронтенда созданы основные эндпоинты.

Серверную часть выгрузили в Docker.

Чтобы развернуть приложение арендовали виртуальный сервер (Ubuntu) на AWS. Установили Docker и Docker Compose для управления контейнерами. Был настроен Jenkins для автоматизации. Интеграция с фронтендом завершена.

Исходя из вышесказанного, можно сделать вывод, что работа бэкенд-разработчика обеспечила стабильность, безопасность и масштабируемость платформы.

1.2 Разбор требований заказчика к программному продукту

В минимальные требования (MVP) входит управление задачами (создание, редактирование, удаление задач; назначение ответственных лиц; именение статусов задач), организация работы команды (группировка задач по проектам, назначение приоритетов, отображение сроков выполнения), интерфейс пользователя (веб-интерфейс с интуитивной навигацией, визуализация статусов задач, базовая аутентификация).

В базовые требования входят диаграммы и аналитика (генерация диаграммы Ганта для визуализации расписания, burndown chart для отслеживания прогресса спринтов, визуализация загруженности команды, автоматический расчет скорости выполнения задач), управление доступами и ролями (ролевая модель, настройка прав доступа, логирование действий пользователей), тайм-трекинг и учет времени (отслеживание времени, затраченного на задачи; генерация отчетов по времени для проектов и участников), автоматизация процессов (настраиваемые уведомления), работа

с багами (создание и приоритизация багов, связь багов с задачами и спринтами, история исправлений и отчеты по закрытым багам).

В оптимальные требования входят сбор и анализ метрик эффективности команды (например: коэффициент завершенности задач, время реакции на инциденты).

Для успешной реализации необходимо:

– соблюдать этапность (сначала MVP, затем дополнительные модули),

– активно тестировать интерфейс и интеграции,

– учесть масштабируемость архитектуры на этапе проектирования.

1.3 План действий для достижения цели (backlog)

Для достижения цели команде необходимо:

1) создать канбан доску,
2) сформулировать этапы работы,
3) написать бэклог,
4) проанализировать аудиторию,
5) сформулировать проблему,
6) мпроектировать основной функционал,
7) сформулировать результат работы,
8) сравнить макеты с функционалом,
9) выбрать цветовую гамму,
10) создать макеты страниц,
11) сверстать страницы,
12) внедрить интерактив,
13) исправить ошибки, отредактировать неточности,
14) спроектировать базы данных,
15) разработать архитектуру приложения,
16) спроектировать API,
17) связать frontend и backend,

18) развернуть веб-сервис,

19) внести исправления,

20) протестировать продукт,

21) написать отчёт,

22) записать видео,

23) загрузить все материалы в TeamProject.

1.4 Анализ и сопоставление аналогов разрабатываемого продукта

Рассмотрены популярные системы управления задачами, используемые командами разработки.

Jira (Atlassian) (рисунок 1)

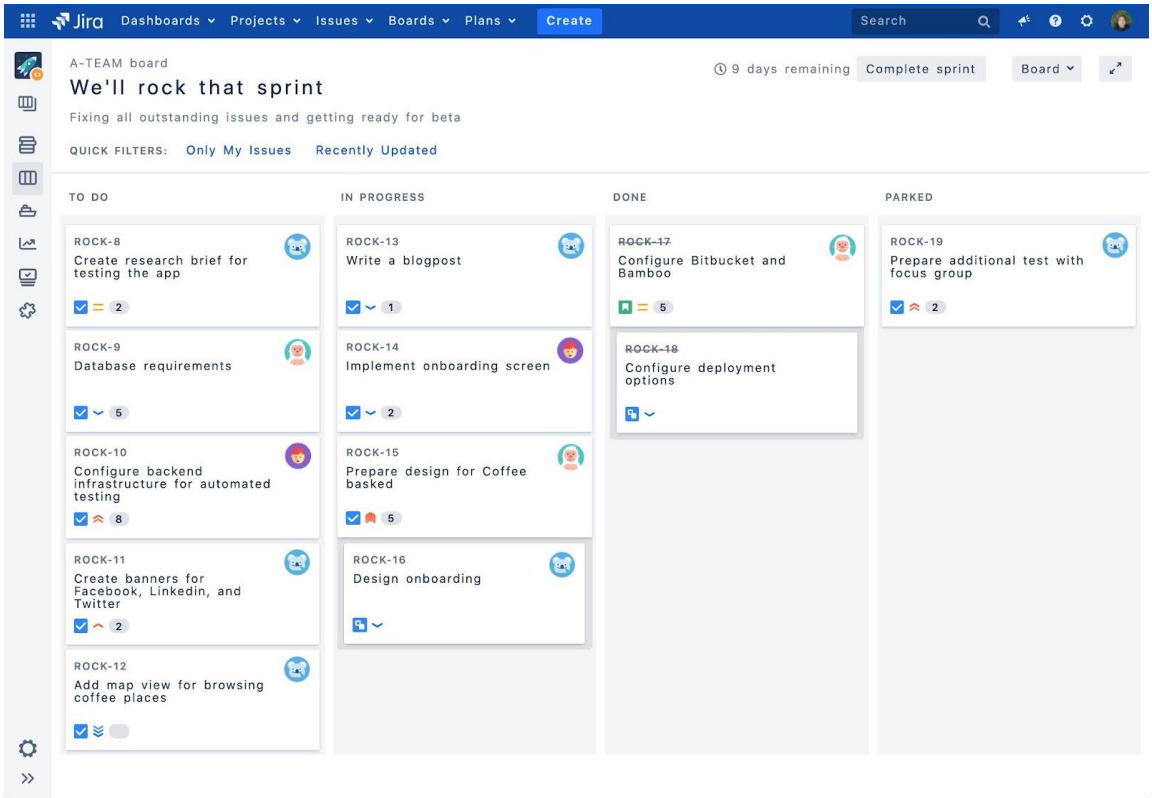

Рисунок 1 – Интерфейс Jira

Плюсы:

– гибкая настройка рабочих процессов,

– глубокая интеграция с DevOps-инструментами (Bitbucket, Confluence, CI/CD),

– поддержка Scrum и Kanban-досок,

– расширенная система отчетности и аналитики.

Минусы:

– высокая стоимость подписки,

– сложность освоения для новых пользователей,

– перегруженность интерфейса.

Trello (Atlassian) (рисунок 2)

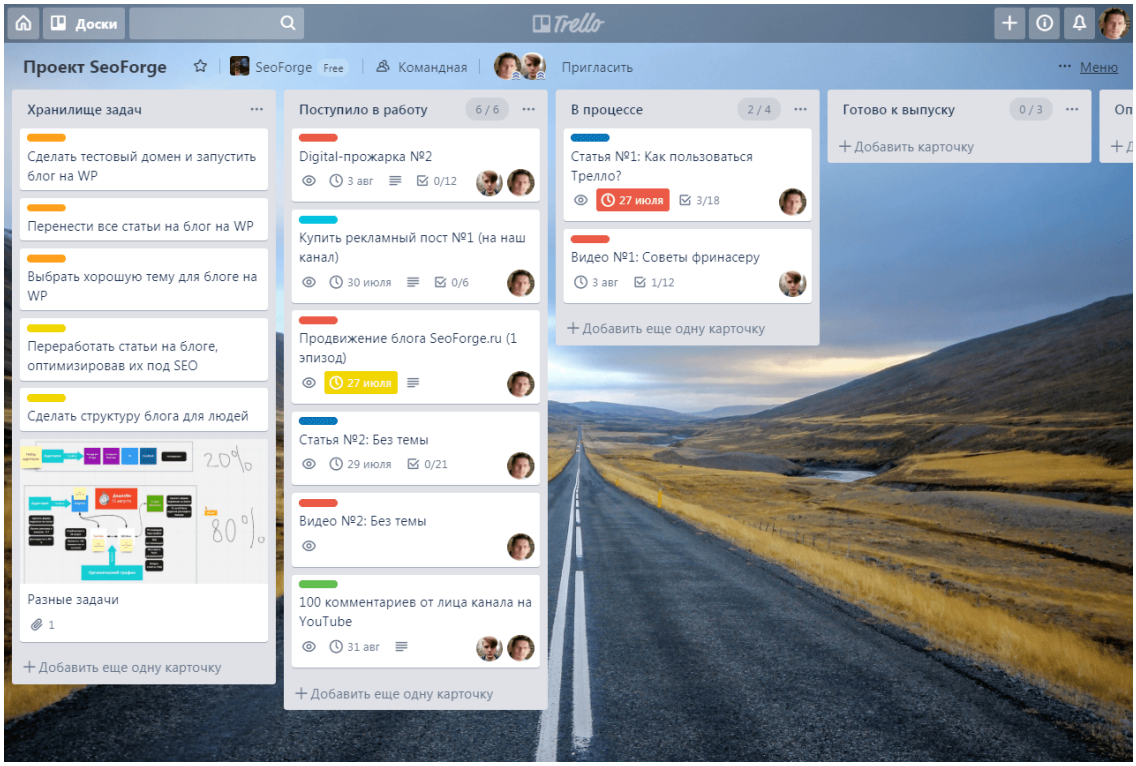

Рисунок 2 – Интерфейс Trello

Плюсы:

– интуитивно понятный интерфейс,

– простая настройка и использование,

– возможность интеграции с различными сервисами (Slack, Google Drive).

Минусы:

– ограниченный функционал для сложных процессов разработки,

– отсутствие встроенной отчетности и метрик,

– не поддерживает сложные рабочие процессы.

Kaiten (рисунок 3)

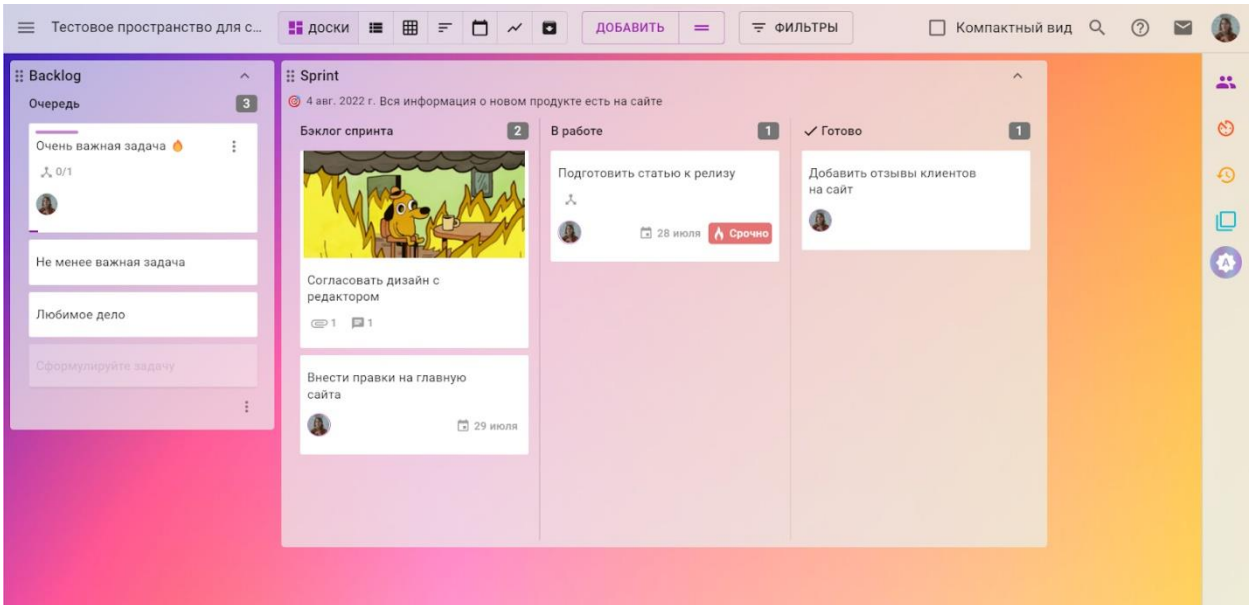

Рисунок 3 – Интерфейс Kaiten

Плюсы:

– полная поддержка Kanban-методологии, включая настраиваемые WIP-лимиты;

– гибкость в управлении процессами, возможность кастомизации под разные команды;

– визуализация потока работы, аналитика узких мест;

– интеграции с Jira, Trello, Slack, GitHub;

– хорошая поддержка командного взаимодействия и прозрачности процессов.

Минусы:

– ограниченный функционал по сравнению с Jira в плане автоматизации сложных бизнес-процессов,

– высокая стоимость для небольших команд,

– может быть сложен в освоении для пользователей, не знакомых с Kanban.

Asana (рисунок 4)

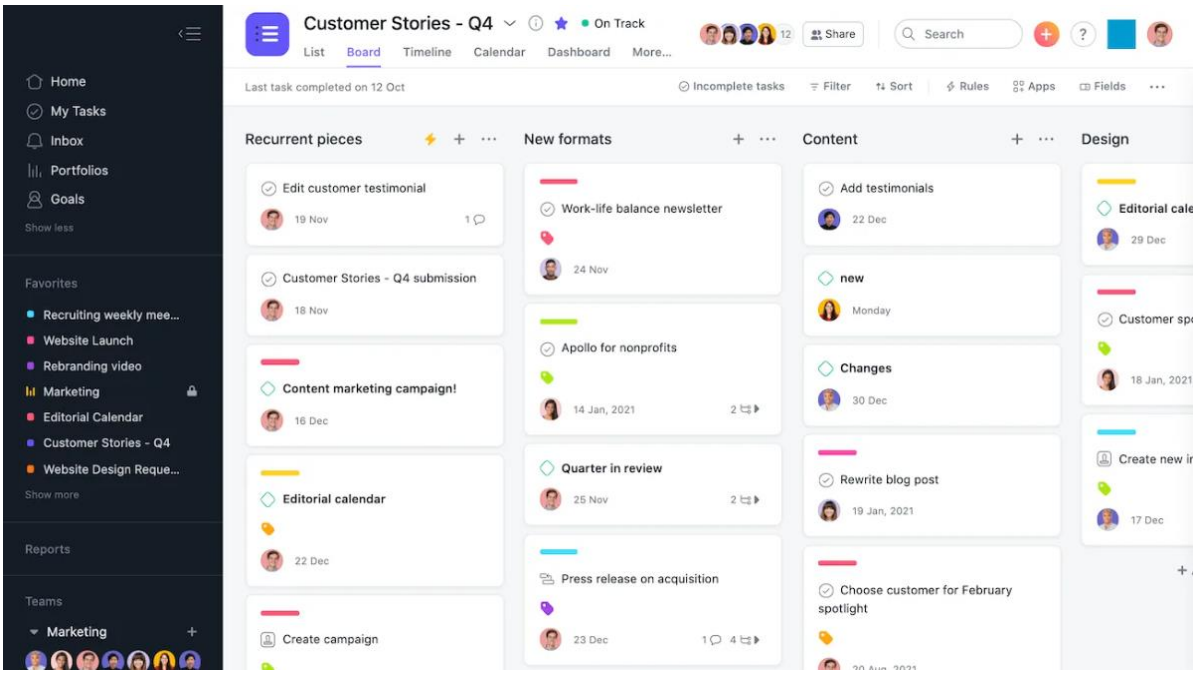

Рисунок 4 – Интерфейс Asana

Плюсы:

– хорошая визуализация задач,

– гибкая система управления проектами,

– поддержка интеграций (Google Workspace, Slack, GitHub).

Минусы:

– ограниченные возможности автоматизации,

– высокая стоимость для больших команд,

– не подходит для DevOps-процессов.

Ключевые недостатки конкурентов:

– высокая стоимость (Jira, Asana, Kaiten),

– ограниченный функционал для сложных процессов разработки (Trello, Asana),

– перегруженность интерфейса (Jira).

Преимущества, которые можно реализовать в нашем продукте:

– простота интерфейса и удобство использования (ориентироваться на Trello, Asana),

– гибкая настройка и поддержка Kanban (как в Kaiten),

– доступная стоимость или freemium-модель для привлечения пользователей,

– оптимизированная работа с метриками и отчетностью.

1.5 Обзор архитектуры программного продукта

Наш продукт - это веб-приложение на Java с использованием Spring Boot и Maven (рисунок 5).

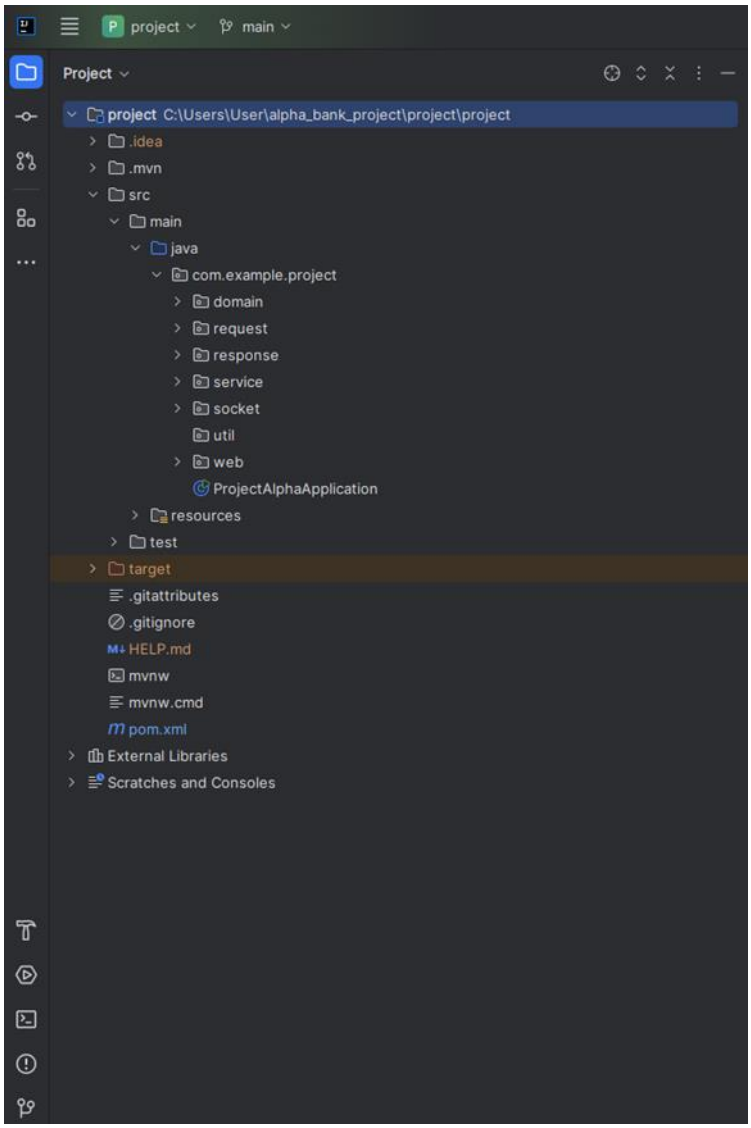

Рисунок 5 – Архитектура приложения

Первый пункт архитектуры – domain. Он содержит модели данных (сущности).

Далее - request/response для передачи данных между клиентом и сервером.

Третье – service. Реализует бизнес-логику.

Следующее – web. Это контроллеры, обрабатывающие HTTP-запросы.

Затем Socket – реализация WebSocket для уведомлений в реальном времени.

Util содержит вспомогательные классы: валидаторы, конвертеры, утилиты для работы с датами/валютой.

Главный класс Spring Boot ProjectAlphaApplication – точка входа в приложение.

Resources – это конфигурационные файлы (application.properties)

Test содержит Юнит – и интеграционные тесты.

pom.xml – файл Maven для управления зависимостями.

mvnw, mvnw.cmd – скрипты для сборки проекта без установки Maven.

Опишем связь между компоненттами. Пользователь отправляет HTTP-запрос. Контроллер (web) принимает запрос, валидирует данные через DTO (request). Контроллер вызывает соответствующий сервис (service). Сервис взаимодействует с domain-моделями для работы с данными, util для вспомогательных операций и внешними системами через API. Сервис возвращает результат контроллеру. Контроллер формирует ответ (response-DTO) и отправляет его клиенту.

Преимущества слоистой архитектуры (Controller-Service-Repository):

– четкое разделение ответственности,

– упрощает тестирование (юнит-тесты для сервисов, интеграционные — для контроллеров),

– гибкость при масштабировании (можно заменить СУБД или внешний API без переписывания всей логики).

Почему выбран Spring Boot:

– Быстрая настройка (автоконфигурация, встроенный Tomcat),

– Широкая экосистема (Spring Security, Spring Data JPA, Spring WebSocket),

– Поддержка микросервисной архитектуры (при необходимости).

Почему выбран Maven:

– Стандарт для Java-проектов.

– Удобное управление зависимостями и плагинами.

– Интеграция с CI/CD (Jenkins, GitLab CI).

Итог: архитектура проекта соответствует лучшим практикам разработки на Spring Boot. Она обеспечивает гибкость, масштабируемость и удобство поддержки, что критично для банковских приложений.

1.6 Информация о планировании деятельности в ходе разработки.

Для планирования работы над проектом была использована таблица, которая содержала информацию о задачах, ответственных и сроках выполнения.

Таблица 1 – Бэклог проекта

Этап/Задача/Работа	Планируемый результат	Ответственный. Исполнитель	Дата начала (план)	Дата финала (план)	Дней
1. Организация работы в команде					
Создать канбан доску	Создана канбан доска, все участники команды добавлены	Вышедко Виктория	10.03.2025	16.03.2025	7
Сформулировать этапы работы	Сформулированы этапы работы	Вышедко Виктория	17.03.2025	23.03.2025	7
Написать бэклог	Написан бэклог	Вышедко Виктория	17.03.2025	23.03.2025	7
2. Аналитика					

Продолжение таблицы 1

Этап/Задача/Работа	Планируемый результат	Ответственный. Исполнитель	Дата начала (план)	Дата финала (план)	Дней
Проанализировать аудиторию	Сделан анализ аудитории	Маклюкова Алина	10.03.2025	16.03.2025	7
Сформулировать проблему	Сформулирована проблема	Маклюкова Алина	10.03.2025	16.03.2025	7
Спроектировать основной функционал	Спроектирован основной функционал	Маклюкова Алина	17.03.2025	23.03.2025	7
Сформулировать результат работы	Сформулирован результат работы	Маклюкова Алина	17.03.2025	23.03.2025	7
Сравнить макеты с функционалом	Вывод о соответствии сделанных макетов функционалу	Маклюкова Алина	07.04.2025	13.04.2025	7
3. Дизайн интерфейса					
Выбрать цветовую гамму	Выбрана цветовая гамма	Бочанова Мария	17.03.2025	23.03.2025	7
Создать макеты страниц	Созданы макеты страниц	Бочанова Мария	24.03.2025	06.04.2025	14
4. Frontend					
Сверстать страницы	Свёрстанные страницы	Демчук Александр	14.04.2025	27.04.2025	14
Внедрить интерактив	Страницы с интерактивом	Демчук Александр	28.04.2025	04.05.2025	7
Исправить ошибки, отредактировать неточности	Готовые страницы с исправленными ошибками	Демчук Александр	05.05.2025	11.05.2025	7
5. Backend					
Спроектировать базы данных	Спроектированы базы данных	Алексей Игнатьев	17.03.2025	23.03.2025	7
Разработать архитектуру приложения	Разработанная архитектура приложения	Алексей Игнатьев	24.03.2025	30.03.2025	7
Спроектировать API	Спроектированы API	Алексей Игнатьев	24.03.2025	30.03.2025	7
Связать frontend и backend	Связанные frontend и backend	Алексей Игнатьев	12.05.2025	25.05.2025	14

Продолжение таблицы 1

Этап/Задача/Работа	Планируемый результат	Ответственный. Исполнитель	Дата начала (план)	Дата финала (план)	Дней
Развернуть веб-сервис	Развёрнутый веб-сервис	Алексей Игнатьев	26.05.2025	01.06.2025	7
Внести исправления	Исправленный веб-сервис	Алексей Игнатьев	02.06.2025	08.06.2025	7
6. Тестирование					
Протестировать продукт	Отчет о проведенном тестировании	Вышедко Виктория	02.06.2025	08.06.2025	7
7. Защита					
Написать отчёт	Написанный отчёт	Вышедко Виктория	26.05.2025	01.06.2025	7
Записать видео	Записанное видео	Вся команда	02.06.2025	08.06.2025	7
Загрузить все материалы в TeamProject	Загруженные материалы	Вышедко Виктория	09.06.2025	15.06.2025	7

ЗАКЛЮЧЕНИЕ

В заключение стоит отметить, насколько нам удалось достичь того уровня, который был задан в требованиях.

Требования, предоставленные к MVP, удовлетворены. То есть в продукте есть возможность управлть задачами (создание, редактирование, удаление задач; назначение ответственных лиц; именение статусов задач), организовывать работу команды (группировка задач по проектам, назначение приоритетов, отображение сроков выполнения), а также веб-интерфейс имеет интуитивную навигациею, визуализацию статусов задач, базовую аутентификацию.

Что касается базовых требований, продукт пока что им не соответствует. Это значит, что диаграммы и аналитика, управление доступами и ролями, тайм-трекинг и учет времени, автоматизация процессов, работа с багами не реализованы.

Оптимальные требования также не выполнены.

К сожалению, сейчас в продукте присутствует множество багов в верстке страниц. На работоспособность они не влияют, так как это эстетические дефекты. Однако же их наличие не позволяет дать высокую оценку программному продукту.

Если говорить об улучшении продукта и его развитии в будущем, то, конечно, в первую очередь нужно исправить все ошибки в верстке страниц. Также хотелось бы осуществить диаграммы и аналитику, управление доступами и ролями, тайм-трекинг и учет времени, автоматизация процессов, работа с багами. Если данные функции будут реализованы, веб-приложение сможет стать качественным, конкурентноспособным и востребованным на рынке цифровых продуктов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЕ А

(обязательное)