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
«Планировщик мероприятий союза студентов ИРИТ-РТФ»

по дисциплине «Проектный практикум»

Заказчик: Гареев Р.И.

Куратор: Шестеров М.А.

Студенты команды: Gummy Bears

Миронюк М. Д.

Рявкина Е. А.

Кисляков М. В.

Потанина А. В.

Виноградов А. А

Екатеринбург, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Любая рабочая организация людей нуждается в инструменте для структурирования и отлаживания процессов их деятельности. На данный момент существует множество электронных систем управления проектами и задачами, но не все подходят под конкретные нужды и запросы компаний, команд разработок и других пользователей. Для студентов ИРИТ-РТФ таким решением стало создание сервиса планировщика мероприятий, который будет разработан под определённые требования заказчика. Планировщик мероприятий – сервис, оснащённый функционалом для удобной организации и создания мероприятий, облегчения координирования деятельности профсоюзной организации людей, оптимизируя деятельность и коммуникацию среди участников. Это повысит мотивацию студентов, улучшит имидж организации и сделает работу союза более продуктивной.

Областью применения программного продукта является организация и проведение различных событий, включая студенческие конференции, культурные мероприятия, спортивные соревнования, тематические встречи, образовательные мастер-классы и тренинги. С помощью сервиса можно будет управлять проектами и задачами, связанными с планированием, распределением обязанностей, контролем выполнения, обменом информацией, координацией действий и анализом эффективности мероприятий.

Проект является продолжающимся, поэтому на основе результатов прошлого семестра были выдвинута следующая цель: доработать web-сервис для планирования мероприятий для Союза студентов ИРИТ-РТФ.

Задачи проекта:

1) анализ задач проекта,
1) оформление проектной документации,
2) формализация процессов,

3) дизайн макетов,
4) разработка (Backend, Frontend),
5) финальная сборка.

По завершении проекта ожидаются следующие результаты:

1) Полностью реализованная система уровней доступа пользователей, где:

а) третий – самый высокий уровень доступа, этим уровнем доступа владеет председатель союза студентов ИРИТ-РТФ,

б) второй – средний уровень доступа, этим уровнем доступа владеют заместители председателя союза студентов ИРИТ-РТФ,

в) первый – низший уровень доступа, этим уровнем доступа владеют все остальные члены команды, то есть те, кто не являются.

Более подробно функционал уровней прописан в техническом задании.

6) возможность создания и хранения в общем доступе текстовых файлов, презентаций, google форм, excel таблиц на сервере,
7) личный календарь пользователя, окно мероприятий, в которых он задействован, и его задач на странице «профиль»,
8) создание и назначение задач ответственным в карточке мероприятий.

1 Основная часть

1.1 Разбор требований заказчика и пользователей

В прошлом семестре был реализован минимально жизнеспособный продукт и для продолжения работы и корректирования вектора действий было проведено интервью с заказчиком, на котором были сформулированы основные требования к продукту:

- обновить существующие фреймы и компоненты для приведения их к единому стилю оформления,
- обеспечить визуальную целостность интерфейса на всех страницах и экранах сервиса,
- обеспечить создание, хранение и редактирование текстовых файлов, таблиц, google форм, презентаций в сервисе, сервис должен быть привязан к корпоративному гугл-диску,
- предоставление личного календаря и окна мероприятий и задач в личном кабинете для каждого пользователя, где он сможет редактировать и просматривать информацию о своих задачах и участии в мероприятиях,
- реализовать функционал разграничения прав доступа пользователей на основе иерархии ролей.

1.2 Составление плана действий для достижения цели (backlog)

Для эффективного планирования и контроля выполнения задач команда использовала методику Yougile, которая позволила четко структурировать работу, распределить роли и отслеживать прогресс на каждом этапе.

Преимущества выбранной методики:

– Гибкость в постановке и корректировке задач,

– Наглядность процессов благодаря визуализации этапов (канбан-доски, чек-листы),

– Удобство командной работы и распределения зон ответственности,

– Возможность оперативно вносить изменения в план при изменении требований.

Ключевые этапы работы по методике Yougile:

1) анализ целей проекта – определение ключевых этапов разработки, постановка измеримых результатов,
2) сбор данных – исследование требований, анализ аналогов, оформление проектной документации – фиксация технических требований,
3) формализация процессов – создание схем взаимодействия, workflow,
4) дизайн макетов – разработка UI/UX, утверждение стиля.
5) разработка (Backend, Frontend) – реализация функционала по спринтам,
6) финальная сборка – интеграция модулей, тестирование, подготовка к релизу.

1.3 Обзор архитектуры программного продукта

Наш продукт предоставляет широкий спектр возможностей для управления и организацией мероприятий создаваемых Союзом студентов ИРИТ-РТФ. Вот основные модули и функции, которые он поддерживает:

1) Управление календарями. Создание, редактирование и просмотр календарей помогает планировать мероприятия и другие события, обеспечивая их своевременное выполнение (см. рисунок 1)

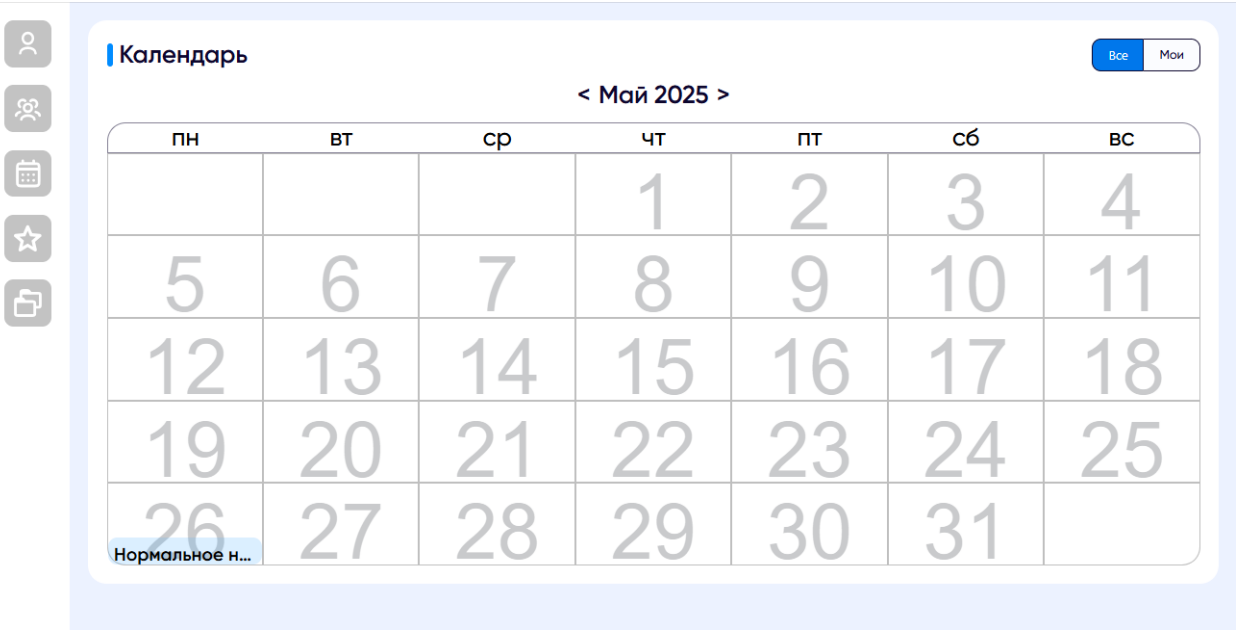

Рисунок 1 – Вид общего календаря

2) Создание и редактирование карточек мероприятий. Удобное создание и редактирование карточек мероприятий позволяет хранить необходимую информацию в одном месте, что упрощает подготовку и организацию мероприятий (см. рисунок 2).

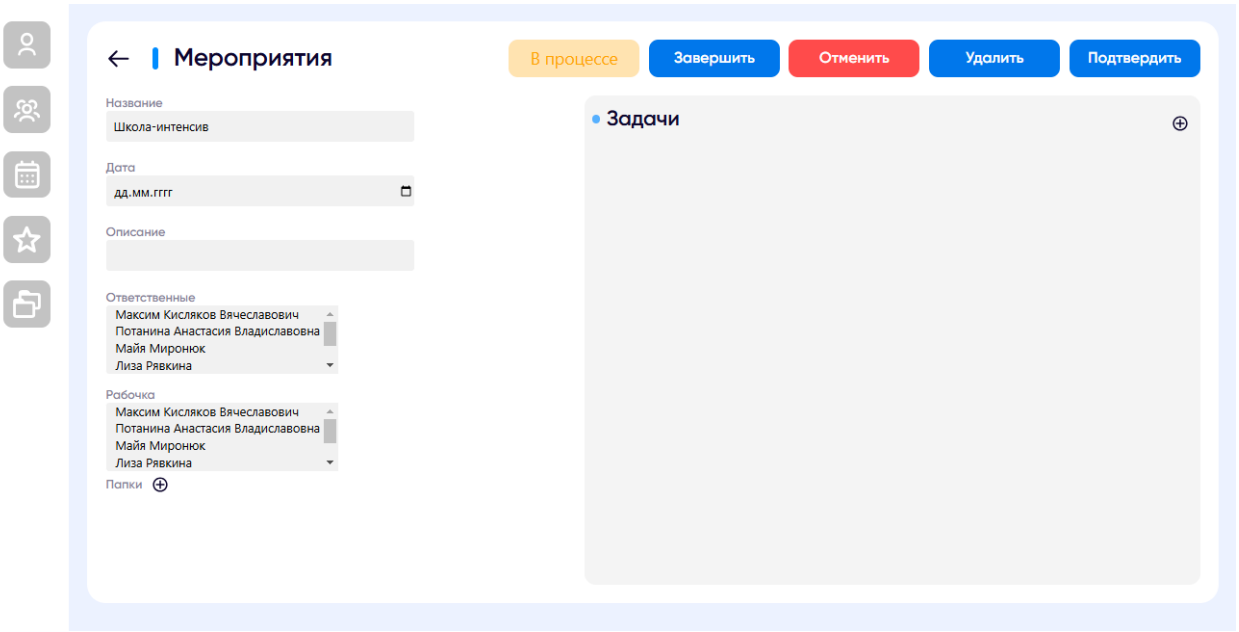

Рисунок 2 – Карточка редактирования мероприятия

3) Добавление информации о мероприятии. Ввод и редактирование деталей мероприятия, таких как дата, время, место и другие важные параметры, обеспечивают точность и своевременность проведения мероприятий (см. рисунок 2).
4) Назначение ответственных лиц и команды участников. Позволяет эффективно распределить задачи и обеспечить их выполнение (см. рисунок 2).
5) Назначение даты мероприятия. Установка и изменение даты проведения мероприятия в соответствии с планом позволяет гибко управлять расписанием и адаптироваться к изменениям (см. рисунок 2).
6) Создание и прикрепление файлов. Возможность создания и прикрепления файлов к мероприятию обеспечивает удобство работы с документами и материалами (см. рисунок 3).

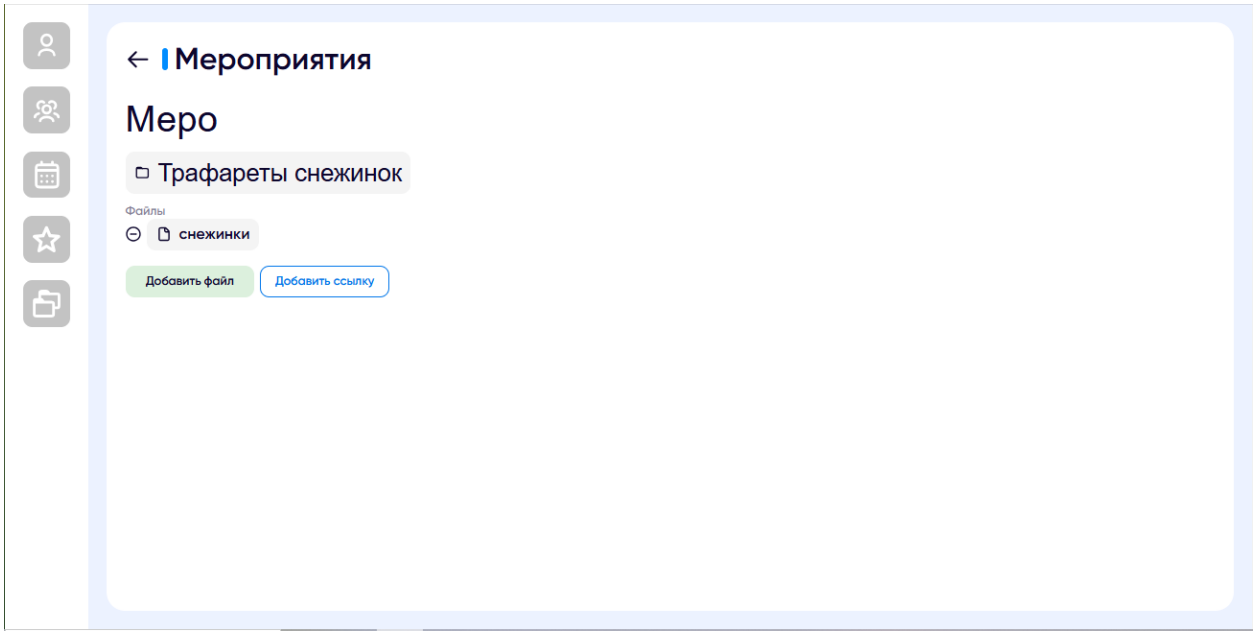

Рисунок 3 – Страница добавления файлов

7) Редактирование и просмотр файлов. Удобный просмотр и редактирование файлов, связанных с мероприятием, прямо в системе сокращает время на обмен документами и улучшает координацию команды (см. рисунок 3).

8) Постановка задач. Назначение и управление задачами для участников команды, связанными с подготовкой и проведением мероприятий, помогает обеспечить их своевременное выполнение (см. рисунок 4).

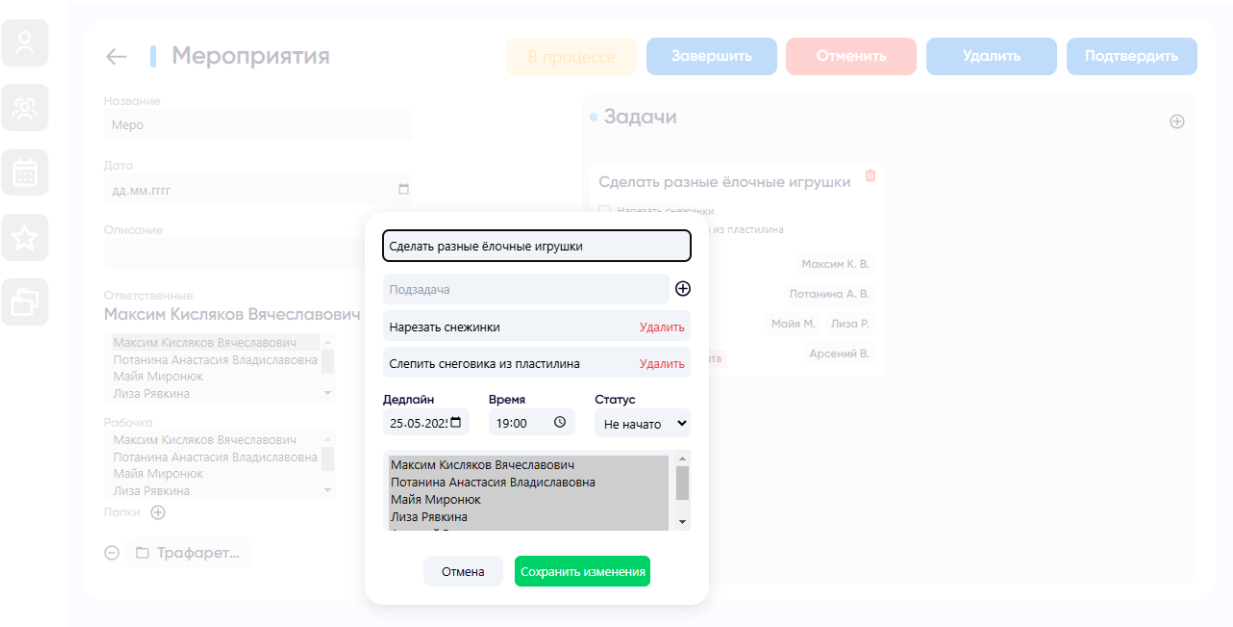

Рисунок 4 – Окно создания задачи

9) Отслеживание выполнения задач. Возможность отслеживания статуса выполнения задач в реальном времени позволяет контролировать ход подготовки и проведения мероприятий, оперативно реагируя на изменения и проблемы (см. рисунок 4).

10) Просмотр страниц членов команды. Просмотр информации о членах команды, включая их профили и выполненные задачи, обеспечивает прозрачность и эффективность работы команды (см. рисунок 5).

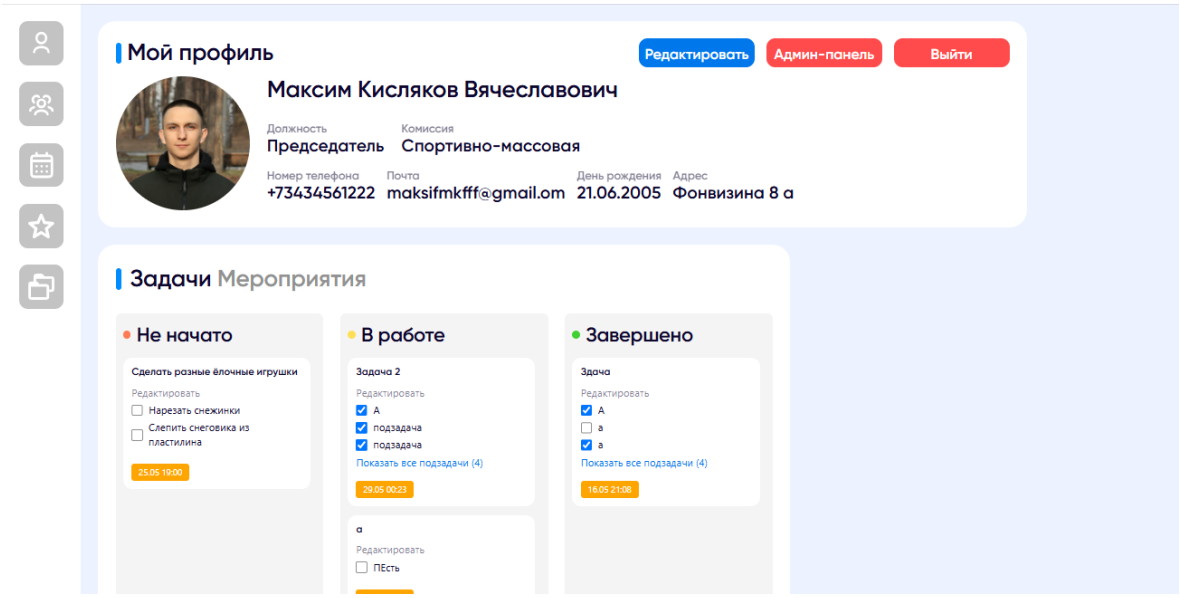

Рисунок 5 – Страница профиля пользователя

11) Смена статуса. Управление статусами пользователей и аккаунтов для организации и контроля доступа позволяет контролировать участие пользователей в проектах и мероприятиях.

12) Добавление и удаление аккаунтов с помощью админ-панели. Гибкое управление аккаунтами пользователей, включая их добавление и удаление обеспечивает возможность адаптации системы к изменениям в составе команды и требованиям пользователей (см.рисунок 6).

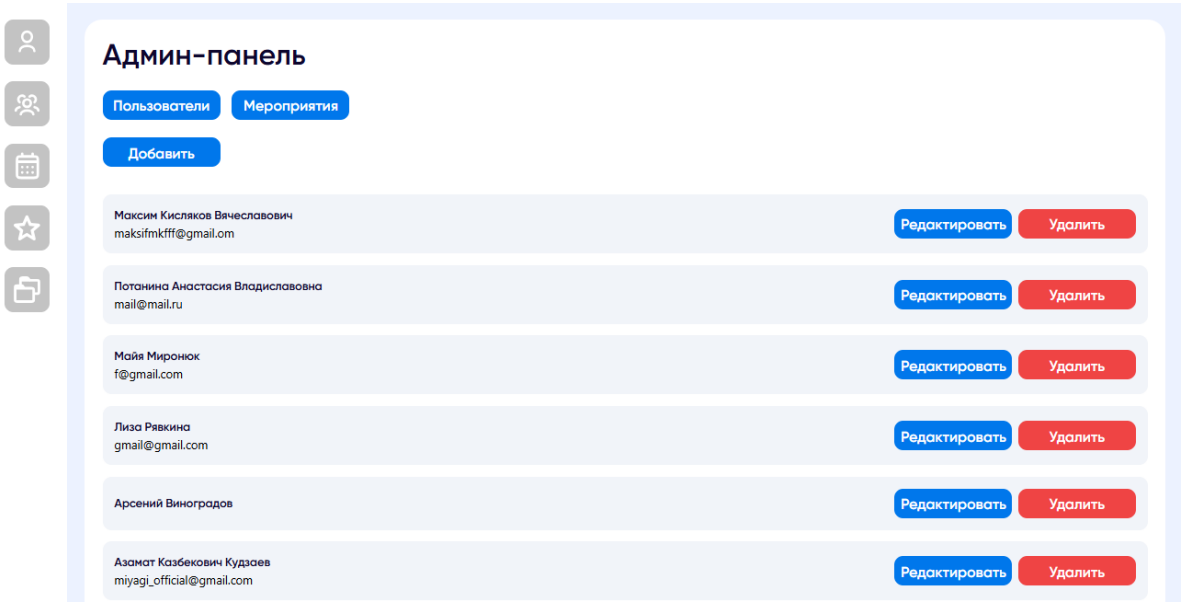

Рисунок 6 – Админ-панель

1.4 Информация о процессе разработки

На старте семестра были собраны требования от заказчика, на основании которых были сформированы workflow-диаграммы, use case-описания и user stories. Это позволило систематизировать процессы работы пользователей и разработчиков.

Основные реализованные функции:

– авторизация и регистрация пользователей

– создание мероприятий и папок с файлами

– привязка google-документов к мероприятиям

– личный кабинет участника

– календарь мероприятий

– редактирование и удаление мероприятий

– отображение занятости участников

Промежуточные версии выкладывались на тестовый сервер для проверки доступности и работоспособности всех модулей.

Разработка велась по следующим этапам:

1) Анализ требований
   Проведены интервью с пользователями и заказчиком. Сформированы следующие документы:
   – Техническое задание.
   – Workflow-диаграмма ролей и прав доступа.
   – Описание пользовательских сценариев.
2) Проектирование
   – Подготовлены макеты экранов.
   – Определена архитектура взаимодействия компонентов системы.
3) Реализация
   – Разработка велась по спринтам продолжительностью 6 недель.
   – В рамках каждого спринта выполнялись задачи по реализации функционала и исправлению замечаний.
4) Тестирование
   – На каждом этапе реализации проводилось модульное и интеграционное тестирование.
   – После завершения спринтов — промежуточное системное тестирование.
5) Релиз
   – Проведена финальная демонстрация.
   – Подготовлен итоговый отчёт с анализом работы.

1.5 Планирование деятельности

Команда использовала таск-трекер Yogile для планирования своей деятельности. Рассмотрим какие задачи были у команды для создания итогового продукта.

1.5.1 Анализ и планирование

Начальный этап проекта представлял из себя планирование работ, анализ разрабатываемого продукта и требований заказчика к нему. Для этого была проведена встреча с заказчиком, были обсуждены: текущее состояние нашего проекта, актуальность целей и задач, дальнейшее направление действий - также была получена и учтена обратная связь, утверждён план работ на семестр.

В первые недели семестра для более конкретной оценки готовности продукта были проведены созвоны с командой, на которых тестировался продукт и выявлялись недочёты его работы. Это позволило более эффективно распланировать работу команды на спринты.

Было пересоставлено техническое задание, в котором описаны все требования к веб-сервису. Также оформили диаграмму Ганта, которая содержит информацию о сроках и ресурсах проекта.

1.5.2 Проектирование

Со стороны аналитика были составлены workflow-диаграммы и use case по ним, благодаря которым разработчики получили информацию о функционировании системы в полной мере.

По требованию заказчика был выполнен редизайн сервиса, подобрана новая цветовая гамма, соответствующая основной теме института, новые макеты содержат новые функции. Позднее согласовали прототип с заказчиком. После ревью заказчика был выполнен готовый дизайн, который

содержал всю необходимую информацию о расположении элементов на странице, цветах, шрифтах и других аспектах дизайна.

1.5.3 Backend-разработка

Начало работы было сосредоточено на устранении накопленных багов. Настроено корректное создание файлов Google Workspace через сервисный аккаунт. Проблема с доступом к файлам была решена путём настройки прав доступа в Google Cloud Console.

Расширена модель профиля пользователя, добавлены новые поля: номер телефона, email, адрес, комиссия и должность. Обновлена система авторизации, внедрены JWT-токены, настроен их срок жизни и механизм аннулирования устаревших токенов.

Разработана система задач: создана модель с привязкой к мероприятию, создателю и нескольким исполнителям, добавлены статусы, дедлайны и флаг архивирования. Настроены эндпоинты для создания, редактирования, удаления и получения задач, обновлены сериализаторы для корректной передачи данных на фронтенд.

Развёртывание и настройка сервера
Для выкладки проекта использован Docker Compose с отдельными контейнерами для фронтенда и бэкенда. В процессе развёртывания выявлены сложности, в частности с загрузкой изображений на сервер. Проблема была решена после настройки прав доступа на папки сервера, что обеспечило корректную загрузку и отображение фотографий.

Контроль качества обеспечен за счёт набора автотестов для приложения: протестирована работа API, система прав, CRUD-операции, валидация данных, обработка ошибок и взаимодействие между компонентами.

Начало работы было сосредоточено на устранении накопленных багов. Настроено корректное создание файлов Google Workspace через сервисный

аккаунт. Проблема с доступом к файлам была решена путём настройки прав доступа в Google Cloud Console.

Расширена модель профиля пользователя, добавлены новые поля: номер телефона, email, адрес, комиссия и должность. Обновлена система авторизации, внедрены JWT-токены, настроен их срок жизни и механизм аннулирования устаревших токенов.

Разработана система задач: создана модель с привязкой к мероприятию, создателю и нескольким исполнителям, добавлены статусы, дедлайны и флаг архивирования. Настроены эндпоинты для создания, редактирования, удаления и получения задач, обновлены сериализаторы для корректной передачи данных на фронтенд.

Развёртывание и настройка сервера
Для выкладки проекта использован Docker Compose с отдельными контейнерами для фронтенда и бэкенда. В процессе развёртывания выявлены сложности, в частности с загрузкой изображений на сервер. Проблема была решена после настройки прав доступа на папки сервера, что обеспечило корректную загрузку и отображение фотографий.

Контроль качества обеспечен за счёт набора автотестов для приложения: протестирована работа API, система прав, CRUD-операции, валидация данных, обработка ошибок и взаимодействие между компонентами.

1.5.4 Frontend-разработка

В рамках разработки продукта последовательно реализовывались функциональные и интерфейсные улучшения, а также устранялись выявленные баги и дорабатывалась бизнес-логика.

На первом этапе устранены ошибки в отображении элементов страницы мероприятия. Исправлена работа флажков, добавлена кнопка для изменения фотографии мероприятия, а также откорректировано отображение

изображений в требуемых размерах. Дополнительно нормализован формат даты рождения и устранено растягивание фонового изображения.

Для повышения удобства взаимодействия с мероприятиями внедрены функции сортировки, выбора должности и комиссии из выпадающих списков, добавлено поле для ввода адреса. Повышена кликабельность интерфейса: обеспечен переход из мероприятия в общий список, из конкретного дня — в календарь, из папки — обратно в карточку мероприятия.

Разработана логика работы с папками: реализована возможность их удаления, а при смене ответственного мероприятие автоматически переносится в учетную запись нового пользователя.

Особое внимание уделено модулю задач. Реализована система задач с подзадачами, дедлайнами, статусами («не начато», «в процессе», «выполнено») и возможностью назначения ответственных. Задачи отображаются в профилях пользователей с возможностью изменения статуса и редактирования. Организовано автоматическое архивирование задач при удалении или переносе мероприятия в архив.

Добавлены новые функции: возможность добавления ссылок на внешние ресурсы, установка статуса «отменено» для мероприятий, поиск по названию и описанию, фильтры по датам, а также отдельное модальное окно для отображения описания мероприятия.

2 Отчет о работе команды

2.1 Отчет тимлида

В течение семестра тимлид координировал команду, занимался планированием и организацией работы, контролировал процессы разработки продукта. Другими задачами тимлида были:

1) Составление диаграммы Ганта, которая отражает этапы разработки сервиса, все задачи и сроки их выполнения, вносил коррективы в план по необходимости.

2) Сбор и уточнение требований заказчика.

3) Организация встреч с командой для обсуждения текущего состояния проекта, выявления проблем и поиска решений.

4) Отображение текущих, выполненных и запланированных задач на платформе Yougile, где каждый член команды мог видеть свои обязанности.

5) Сбор и подготовка необходимой информации для контрольных точек, создание презентаций и их защита.

Трудности, возникшие в процессе выполнения проекта: некоторое время не уделял задачам проекта должное внимание, из-за чего процесс разработки замедлился. Позже команда успела восстановиться и выполнить все поставленные задачи.

2.2 Отчет аналитика

В начале семестра функционал продукта был реализован на 50%, большинство функций были в промежуточном состоянии, не было полноценного разграничения уровней доступа пользователей, поэтому были пересмотрены техническое задание и существующие диаграммы, собраны

требования от заказчика. На основе полученной информации были сформированы новые workflow-диаграммы, use case-описания и user stories. Это позволило систематизировать процессы работы пользователей и разработчиков.

Основными требованиями заказчика являлись возможность управлять мероприятиями, возможность создавать корпоративные файлы общего доступа для пользователей сервиса, отслеживание занятости подчинённых и разграничение календаря событий на личный и общий. В соответствии с этими условиями были реализованы следующие страницы сервиса: личный профиль, команда пользователей, общий календарь, действующие мероприятия и архив мероприятий. Более подробно о новых функциях проекта, которые были разработаны аналитиком:

– страница профиля пользователя: более адаптивным решением для отслеживания личной занятости стали окна, отображающие задачи пользователя и мероприятия, в которых он задействован,

– страница календаря мероприятий: мероприятия, созданные в одинаковый день, могут быть сортированы по дате, найдены с помощью поиска названия или описания; в информацию о мероприятии были добавлены такие поля, как назначение ответственных и помощников, изменение статуса мероприятия на «в процессе/завершено/отменено», также появилось окно для назначения задач пользователям с регулированием дедлайнов; на данной странице можно создавать папки, в которых могут быть созданы файлы и сохранены google-файлы по ссылкам.

– страница действующих мероприятий: мероприятия могут быть сортированы по дате и найдены по поиску названия или описания,

– админ-панель: для удобства внесения новых пользователей в систему была разработана и внедрена админ-панель на странице профиля пользователя с наивысшим уровнем доступа: данная функция позволяет добавлять и удалять пользователей, заносить личную информацию о них в систему.

Промежуточные версии проекта выкладывались на тестовый сервер для проверки доступности и работоспособности всех модулей. В связи с этим были уточнены права доступа для пользователей с разным уровнем прав.

В результате проведенного анализа были выявлены недочёты и неполадки в работе сервиса, благодаря которым были составлены workflow-диаграммы, демонстрирующие логику работы сервиса. На основе User Stories, дерева требований и тестированию, было сформировано подробное техническое задание, готовое для передачи команде разработчиков. Сервис обладает функционалом создания и планирования мероприятий, интегрирования с календарями, обеспечения возможности работы с файлами и управления задачами.

2.3 Отчет дизайнера

В рамках проекта была разработана визуальная часть системы сервиса планирования мероприятий Союза Студентов ИРИТ-РТФ. Основная задача заключалась в редизайне интерфейса, повышение удобства и улучшение графического представления для пользователей. Дизайн разрабатывался с учётом современных трендов и специфики использования, включая требования к адаптивности и интуитивно понятной навигации.

Основные достижения:

1) Дизайн макетов интерфейсов:

а) Разработана новая цветовая схема, обеспечивающая узнаваемость бренда и комфортное восприятие информации. Выполнена смена темы оформления с темной на светлую.

б) Переработаны макеты для ключевых разделов:

– Авторизация.

– Личный профиль.

– Карточки мероприятий с новым дизайном полей и новыми функциями.

в) Спроектировано адаптивное решение для страницы личного кабинета пользователя с возможностью:

– Просмотра личного календаря, в котором отображаются все мероприятия, планёрки и события с участием пользователя.

– Отображения окна актуальных мероприятий и задач, в которых задействован пользователь. Это позволило сделать личную страницу более информативной и удобной для планирования и отслеживания занятости пользователей.

6) Проработка UX:

а) Спроектированы удобные сценарии пользовательских взаимодействий.

б) Внесены правки на основе обратной связи от команды фронтенд-разработчиков и тестовой группы.

7) Трудности и их решения:

а) Адаптация дизайна: технические ограничения требовали адаптации макетов. Решением стали совместные созвоны с разработчиками для уточнения требований и поиска компромиссных решений.

б) Гибкие компоненты: создание универсального дизайна карточек мероприятий, подходящих для разного объема данных, было реализовано с использованием гибкого подхода к размещению контента и учета возможных исключений.

Сотрудничество с командой разработки:

В процессе работы решались технические и дизайнерские вопросы, требующие совместного обсуждения и поиска оптимальных решений. В частности, потребовались итерации по согласованию дизайна, адаптация календаря и создание универсальных карточек мероприятий. Дизайн-макеты

передавались через Figma с подробными аннотациями, и регулярно проводились обсуждения сложных моментов интеграции. Результатом совместных усилий стал функциональный и визуально привлекательный интерфейс, отвечающий требованиям проекта и современным стандартам.

2.4 Отчет Frontend-разработчика

Для улучшения интерфейса и функциональности системы были внесены изменения:

– Исправлены ошибки в отображении флажка мероприятия и добавлена кнопка для изменения фото, что упростило редактирование информации о мероприятиях.

– Устранена проблема с растягиванием задника, дата рождения приведена к нормальному формату, а фото отображается с верными параметрами, повысив качество отображения данных.

– Добавлена кнопка сортировки мероприятий, добавлены списки с выбором должности и комиссии, а также поле для ввода адреса, сделав интерфейс более удобным и функциональным.

– Улучшена кликабельность мероприятий: добавлены стрелки для возврата на шаг назад из мероприятий во все, из конкретного дня в календарь и из папки обратно в мероприятие.

– Добавлена возможность удаления папок, и при смене ответственного мероприятие автоматически переносится в аккаунт нового, повысив гибкость системы.

– В систему интегрирована логика задач с названием, подзадачами, дедлайном, статусом (не начато, в процессе, выполнено) и выбором нескольких ответственных. Задачи отображаются у ответственных в профилях, где можно менять статус, после чего перемещаются в

соответствующую колонку. Их также можно редактировать прямо из профиля, упрощая управление и повышая эффективность работы.

– Создана возможность добавления ссылок на внешние ресурсы или страницы.

– Автоматическое архивирование задач при удалении карточки мероприятия: если карточка мероприятия была перемещена в архив, то все связанные с ней задачи также автоматически перемещаются в архив.

– Теперь в системе есть админ-панель. С её помощью можно добавлять новых пользователей, редактировать данные существующих или удалять их аккаунты.

1. Статус «отменено» отображается у мероприятий как новый.
2. В календаре отображаются только мероприятия пользователя.
3. Добавлен поиск по названию и описанию мероприятий.
4. Реализованы фильтры по дате проведения мероприятий.
5. Добавлена кнопка выхода из профиля.
6. Описание мероприятия вынесено в отдельное модальное окно.

В ходе проектной работы возникли некоторые трудности:

1) Долгое время не удавалось настроить корректное изменение фотографии в профиле. Теперь пользователи могут менять фото, которое сохраняется и обрезается в нужных параметрах.
2) Добавление новой логики для подзадач оказалось сложным. Задача не всегда отображалась корректно, а также возникали проблемы с определением владельца задачи. Теперь задача привязана к конкретному пользователю, а не видна во всех аккаунтах:

Листинг 1 – фрагмент кода создания задачи

```
const handleCreateTask = () => {
        if (!newTask.title || newTask.title.trim() === '') {
            console.log('Попытка создания задачи без названия');
            setTitleError(true);
```

```
            return;
        }
        console.log('Начало создания задачи:', newTask);

        // Оптимизируем структуру данных для уменьшения размера
JSON
        const deadlineString = newTask.deadline ?
(newTask.deadline + (newTask.deadlineTime ? 'T' +
newTask.deadlineTime : '')) : null;
        const taskObject = {
            t: newTask.title.trim(),
            d: newTask.description?.trim() || '',
            dl: deadlineString,
            s: newTask.status || 2,
            e: newTask.assignee || [],
            ev: parseInt(eventData.id),
            st: newTask.subtasks.map(subtask => ({
                t: (subtask.title || '').trim(),
                s: subtask.status || 2
            }))
        };

        const taskJson = JSON.stringify(taskObject);

        const taskData = {
            task: taskJson,
            event: eventData.id,
            executor: newTask.assignee || null
        };

        console.log('Подготовленные данные задачи:', taskData);

        axios.post(`${BASE_URL}/api/tasks/`, taskData, {
            headers: {
                'Content-Type': 'application/json',
```

```
                'Authorization': `Bearer
${localStorage.getItem('access_token')}`
            }
        })
        .then(response => {
            console.log('Успешный ответ от сервера:',
response.data);

            // Преобразуем обратно в полные имена полей для
отображения
            const taskDetails = JSON.parse(response.data.task);
            const fullTaskDetails = {
                title: taskDetails.t || taskDetails.title || '',
                description: taskDetails.d ||
taskDetails.description || '',
                deadline: taskDetails.dl || taskDetails.deadline
|| null,
                status: taskDetails.s || taskDetails.status || 2,
                executor: taskDetails.e || taskDetails.executor
|| null,
                event: taskDetails.ev || taskDetails.event ||
parseInt(eventData.id),
                subtasks: (taskDetails.st || taskDetails.subtasks
|| []).map(st => ({
                    title: st.t || st.title || '',
                    status: st.s || st.status || 2
                }))
            };

            // Обновляем список задач в событии
            const updatedEvent = {
                ...event,
                tasks: [...(event.tasks || []), {
                    id: response.data.id,
                    ...fullTaskDetails,
```

```
                    task: response.data.task
                }]
            };

            console.log('Обновленное событие:', updatedEvent);
            setEvent(updatedEvent);

            // Сбрасываем форму
            setNewTask({
                title: '',
                description: '',
                deadline: '',
                deadlineTime: '',
                status: 2,
                assignee: [],
                subtasks: []
            });
            setTaskModalIsOpen(false);
            setTitleError(false);
        })
        .catch(error => {
            console.error('Ошибка при создании задачи:', error);
            console.error('Детали ошибки:',
error.response?.data);
            console.error('Статус ошибки:',
error.response?.status);
            alert('Не удалось создать задачу. Пожалуйста,
проверьте данные и попробуйте снова.');
        });
    };
```

3) Проблемы с чекбоксами подзадач были решены. Для открытия задачи теперь нужно нажимать на её название, а не на всю карточку.

2.5 Отчет Backend-разработчика

Начало семестра было посвящено устранению багов, которые остались с конца предыдущего. Файлы гугл сервисов начали корректно создаваться, переписаны модели для них:

Листинг 1 – код моделей для хранения файлов

```
class ProjectFile(models.Model):
    """
    Модель для хранения файлов, связанных с проектом.
    Поддерживает как файлы Google Workspace (Docs, Sheets, Slides, Forms),
    так и внешние ссылки.

    Attributes:
        project (ForeignKey): Связь с проектом
        file_type (CharField): Тип файла (Документ, Таблица, Презентация, Форма, Ссылка)
        file_url (URLField): URL файла в Google Workspace или внешняя ссылка
        created_at (DateTimeField): Дата и время добавления файла (автоматически)
        file_name (CharField): Пользовательское имя файла (опционально)
    """
    project = models.ForeignKey(Project, related_name='files', on_delete=models.CASCADE, verbose_name='Проект')
    file_type = models.CharField(max_length=50, verbose_name='Тип файла')
    file_url = models.URLField(verbose_name='Ссылка на файл')
    created_at = models.DateTimeField(auto_now_add=True, verbose_name='Дата добавления')
    file_name = models.CharField(max_length=255, blank=True, null=True)
```

```
    def __str__(self):
        return f"{self.file_type} для {self.project.title}"
```

Также для гугл файлов был настроен и подключен сервисный аккаунт, от имени которого и создаются файлы гугл сервиса. Его преимущество заключается в том, что не требует авторизации, к нему можно подключить основной Google аккаунт и получить доступ ко всем файлам через Google Cloud Console. При добавлении сервисного аккаунта потерялся доступ к создаваемым файлам, данная ошибка отняла много времени, изучив статьи в интернете, получилось настроить доступ, чтобы пользователи могли читать и изменять эти файлы.

Далее по ТЗ у профилей появились новые поля, которые было необходимо настроить на стороне бэкенда: номер телефона, email адрес, адрес проживания, комиссия, должность.

Листинг 2 –

```
class UserProfile(models.Model):
    """
    Модель профиля пользователя, расширяющая стандартную модель User Django.

    Attributes:
        user (OneToOneField): Связь один-к-одному с моделью User Django
        full_name (CharField): Полное имя пользователя
        date_of_birth (DateField): Дата рождения (опционально)
        commission (CharField): Комиссия или отдел пользователя
        status (CharField): Статус пользователя
        number_phone (PhoneNumberField): Номер телефона в российском формате
        email (EmailField): Email адрес
        adress (CharField): Адрес пользователя
```

```
        profile_photo (ImageField): Фотография профиля
        access_level (PositiveSmallIntegerField): Уровень доступа
пользователя (1-Viewer, 2-Editor, 3-Admin)
    """
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    full_name = models.CharField(max_length=100)
    date_of_birth = models.DateField(null=True, blank=True)
    commission = models.CharField(max_length=100, blank=True)
    status = models.CharField(max_length=150, blank=True)
    number_phone = PhoneNumberField(blank=True, region='RU')
    email = models.EmailField(blank=True)
    adress = models.CharField(blank=True, max_length=200)
    profile_photo                                                =
models.ImageField(upload_to='profile_photos/', blank=True)

    ACCESS_LEVELS = [
        (1, 'Viewer'),
        (2, 'Editor'),
        (3, 'Admin'),
    ]
    access_level                                                 =
models.PositiveSmallIntegerField(choices=ACCESS_LEVELS,
default=1)

    def __str__(self):
        return self.user.username
```

После этого была обновлена система авторизации через JWT_token, настроена отдача и время жизни токена, а также отправка в блэклист старых токенов:

Листинг 3 -

```
logger = logging.getLogger(__name__)

class CustomTokenObtainPairSerializer(TokenObtainPairSerializer):
```

```
    """
    Расширенный сериализатор для получения JWT токена.
    Добавляет дополнительную информацию о пользователе в токен.
    """
    def validate(self, attrs):
        data = super().validate(attrs)

        try:
            profile = UserProfile.objects.get(user=self.user)
            data['user_id'] = self.user.id
            data['profile_id'] = profile.id
            data['access_level'] = profile.access_level
        except UserProfile.DoesNotExist:
            data['user_id'] = self.user.id
            data['profile_id'] = None
            data['access_level'] = None

        return data

class CustomTokenObtainPairView(TokenObtainPairView):
    """
    Представление для получения JWT токена с дополнительной информацией о пользователе.
    """
    serializer_class = CustomTokenObtainPairSerializer
```

Была настроена система для людей с первым уровнем доступа, то есть максимальным, теперь пользователя данного уровня доступа могут менять комиссию и должность всем людям.

Далее крупным действием было создание задач. Внутри карточки мероприятия теперь можно ставить отдельные задачи для каждого человека, которые включают в себя заголовок, ответственных, дедлайны, подзадачи и

статус. Полностью настроена для них система прикрепления к профилю человека, добавлены методы для удаления, редактирования и просмотра задач.

Листинг 4 -

```
class Tasks(models.Model):
    """
    Модель задачи, связанной с мероприятием.

    Attributes:
        task (CharField): Название задачи
        description (CharField): Описание задачи
        event (ForeignKey): Связь с мероприятием
        deadline (DateField): Срок выполнения
        creator (ForeignKey): Создатель задачи
        executor (ManyToManyField): Исполнители задачи
        status (PositiveSmallIntegerField): Статус задачи (1-В процессе, 2-Не начата, 3-Выполнена)
        is_past (BooleanField): Флаг, указывающий, что задача в прошлом
    """
    task = models.CharField(max_length=20000)
    description = models.CharField(blank=True, max_length=20000)
    event = models.ForeignKey(Event, on_delete=models.CASCADE, related_name='tasks_for_event')
    deadline = models.DateField(blank=True, null=True)
    creator = models.ForeignKey(User, on_delete=models.CASCADE, related_name='created_tasks')
    executor = models.ManyToManyField(User, related_name='executed_tasks', blank=True)
    STATUS = [
        (1, 'В процессе'),
        (2, 'Не начата'),
        (3, 'Выполнена')
    ]
```

```
    status   =   models.PositiveSmallIntegerField(choices=STATUS,
default=2)
    is_past = models.BooleanField(default=False)

    def __str__(self):
        return self.task
```

Для выкладки на сервер были настроены docker compose: .yaml файл и docker файлы для backend и frontend'a, также на самом хостинг сервере nginx и корректно проведён деплой проекта. На этом этапе возникло много разных ошибок от простых до очень сложных, о которых будет сказано далее.

Для решения некоторых проблем на хостинге и более быстрой проверки работоспособности были добавлены тесты для веб приложения user_account. Все тесты написаны с использованием Django TestCase и APITestCase, и проверяют:

– корректность работы API endpoints,

– систему прав доступа,

– CRUD операции,

– валидацию данных,

– обработку ошибок,

– взаимодействие между различными компонентами системы.

Во время работы было исправлено множество багов, редактировались эндпоинты, допустимые запросы на них, в последние дни были настроены сериализаторы под задачи. Также добавлены некоторые параметры для задач, которые были необходимы фронтенду, например прошла ли задача или нет, это нужно было для того, чтобы когда вся карточка мероприятия уходила в архив, и задачи тоже уходили в архив. Были добавлены допустимые запросы к бэку, для настройки админ панели, на добавление и удаление пользователей. Также была добавлена поддержка добавления не только файлов гугл сервисов, которые создаются внутри нашего сайта, но и сторонних файлов.

Огромная сложность возникла на этапе деплоя проекта на сервис, главная из которых была загрузка фотографий профиля. Перебрав множество вариантов решения из интернета, воспользовавшись нейросетями и собственными знаниями, ответа почему локально все работало, а на сервере нет – не было. Решение появилось при выполнении лабораторной работы другой дисциплины – операционные системы, когда на операционной системе Linux необходимо было настраивать пользователю доступы к файлам. Применив команды для установления прав на сервере – фотографии стали загружаться и выводиться. При каждом новом изменении на сервере возникали новые проблемы, связанные с локальной работы некоторого функционала, которые быстро решались.

ЗАКЛЮЧЕНИЕ

На основании проведённых работ можно сделать вывод, что разработанный программный продукт соответствует заявленным требованиям заказчика и ожиданиям пользователей.
Были реализованы ключевые функции, включая:

– создание и планирование мероприятий,
– ведение личных и общих календарей,
– возможность назначения ответственных и создания задач,
– дизайн интерфейса удобен в использовании и соответствует стилистике института,
– хранение файлов осуществляется путем их создания непосредственно через сервис,
– разграничение уровней доступа пользователей.

Особую ценность для пользователей представляют внедрённые элементы интерактивности и удобства: адаптивный интерфейс, личные календари с актуальными мероприятиями и задачами, а также переработанная структура карточек мероприятий.
Проведённая работа позволила улучшить пользовательский опыт, сделать работу с сервисом более интуитивной и современной. Взаимодействие с заказчиком и пользователями на промежуточных этапах дало возможность оперативно учитывать их пожелания, что повысило актуальность и практическую ценность продукта.

Выполненные работы позволили существенно повысить удобство и функциональность сервиса, привести его к актуальным требованиям и ожиданиям пользователей. Программный продукт продемонстрировал соответствие целям проекта, высокую стабильность и готовность к практическому использованию.
Фиксация результатов и выявленных улучшений формирует хорошую основу

для дальнейшего развития системы в следующих семестрах или будущих итерациях проекта.

URL нашего сервиса: https://irit-rtf-ep.ru/

Логин для входа: maksi

Пароль: lr.lv_frvr3512

Предложения по улучшению и развитию продукта:

– разработка мобильной версии сервиса,
– интеграция системы рассылок или уведомлений о выполнении задач и изменений в проекте на почту или отображение уведомлений в сервисе,
– добавление чатов между пользователями.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Карлос, Э. JWT. Безопасная аутентификация и авторизация в веб-приложениях / Э. Карлос. — СПб.: Питер, 2022. — 320 с.

2. Мациевский, С. Backend-разработка на Python и Django / С. Мациевский. — М.: БХВ-Петербург, 2020. — 480 с.

3. Фримен, Э. Изучаем React / Э. Фримен, Э. Робсон. — 2-е изд. — СПб.: Питер, 2021. — 560 с.

4. Халл, Э. Инженерия требований / Э. Халл, К. Джексон, Дж. Дик. – [б. м.], 2023. – 352 с.

5. О роли системного аналитика // Habr.com. – URL: https://habr.com/ru/companies/moysklad/articles/542576/ (дата обращения: 17.03.2025).

6. Системный анализ и моделирование информационных процессов и систем : учебное пособие / под ред. И. П. Норенкова. – СПб. – СПбГУ ИТМО, 2010. – 480 с. – URL: https://books.ifmo.ru/file/pdf/2140.pdf (дата обращения: 20.04.2025).

7. Функции системы управления проектами // Yougile.com. – URL: https://ru.yougile.com/product (дата обращения: 20.05.2025).