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 системы контроля и анализа количества участников мероприятий»

по дисциплине «Проектный практикум»

Заказчик: Серков К.В.

Куратор: Пирогов М.С.
ученая степень, ученое звание, должность

Студенты команды «счастливы вместе» 4 человека

Дружинин М.Д.

Мартынов П.М.

Мешков М.А.

Поляков С.С.

Екатеринбург, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель проекта: разработка электронного сервиса для автоматизации контроля посещаемости мероприятий, направленного на повышение эффективности работы организаторов.

Задачи:

1) создание клиентской части для регистрации участников через NFC/QR-коды или идентификаторы;

2) реализация административной панели для настройки мероприятий, управления ролями и доступами;

3) интеграция системы учёта с возможностью аналитики посещаемости;

4) разработка интуитивно понятного интерфейса с поддержкой физических кнопок управления для упрощения работы на месте.

Актуальность: в условиях роста числа мероприятий ручной учёт участников становится ресурсозатратным и подверженным ошибкам; современные реалии требуют автоматизации процессов сбора данных, что подтверждается трендами на цифровизацию управления событиями.

Область применения программного продукта:

По завершении проекта будет реализован многофункциональный сервис, соответствующий трём уровням результата:

– минимальный: рабочий прототип с базовым функционалом учёта участников;

– базовый: полноценная система с клиентской и административной частями, поддержка NFC/QR-кодов;

– оптимальный: дополнительные функции аналитики посещаемости, интеграция с другими системами, улучшенный UI/UX.

Планируемые достижения:

– Разработка и запуск рабочего прототипа системы учёта участников мероприятий.

– Создание административной панели для управления мероприятиями, ролями и правами доступа.

– Подключение и настройка идентификации участников через NFC/QR-коды или уникальные идентификаторы.

– Реализация интерфейса для организаторов, адаптированного под работу на месте проведения (в т.ч. с физическими кнопками).

– Обеспечение надёжного хранения и обработки данных о посещаемости в реляционной СУБД.

– Предоставление аналитических инструментов для оценки активности и эффективности мероприятий.

– Закладка фундамента для дальнейшей интеграции с внешними системами и расширения функционала.

1 Основная часть

1.1 Индивидуальный вклад участников

Дружинин М.Д. роль: фронтенд-разработчик:

В рамках проекта занимался проработкой и реализацией ключевых элементов пользовательского интерфейса.

Одним из первых этапов его работы стало создание схемы (карты) веб-сервиса, на которой отображены основные разделы системы, логика переходов и взаимосвязи между компонентами клиентской и серверной части. Это позволило команде заранее визуализировать структуру приложения и эффективно спланировать дальнейшую разработку.

С технической стороны реализовал важный функционал на стороне интерфейса:

- ручное добавление участников к мероприятию через административную панель,
- отображение полного списка участников,
- возможность отметки участников как присутствующих на конкретном мероприятии.

Мартынов П.М. роль: тимлид, фронтенд-разработчик:

В рамках работы над проектом я взял на себя роль координатора команды: организовывал регулярные созвоны с куратором и участниками, чтобы синхронизировать прогресс, получать обратную связь и корректировать вектор развития проекта.

Также я занимался планированием и распределением задач между членами команды, исходя из их компетенций и текущих приоритетов разработки. Это позволило наладить продуктивную командную работу и обеспечить соблюдение сроков.

С технической стороны я реализовал основу клиентского веб-интерфейса с использованием библиотеки React Admin. Разработал и настроил ключевые компоненты, обеспечивающие взаимодействие с сервером. Настроил

авторизацию, обработку токенов, а также выполнение запросов к серверу через REST API, что позволило связать фронтенд с backend-логикой.

Кроме того, я подготовил презентацию, структурировал материал, собрал ключевую информацию по проекту и оформил её в наглядный и понятный формат. Также структурировал и оформил отчёт.

Поляков С.С. роль: бэкенд-разработчик:

Я занимался созданием серверной части проекта и проектированием базы данных.

Был разработан API с использованием Django и Django REST Framework, обеспечивающий всю необходимую бизнес-логику: работу с мероприятиями, участниками, ролями и авторизацией. Особое внимание уделялось безопасности и масштабируемости решений, а также соответствию архитектуры требованиям проекта.

Кроме того, я занимался проектированием и реализацией структуры базы данных, обеспечив надёжное хранение и быструю обработку информации. Схема БД строилась с учётом расширяемости и возможной интеграции с внешними системами.

В данный момент активно ведётся дополнение и уточнение документации по API, на основе которой разрабатывается и дополняется серверный код. Это способствует унификации подходов и облегчает взаимодействие с фронтенд-частью проекта.

Вся работа по серверной логике и настройке проекта размещена в репозитории GitHub.

Мешков М.А. роль: веб-дизайнер:

Пока для административной части не разрабатывался отдельный дизайн, поскольку React Admin поставляется с готовыми UI-компонентами и стандартными стилями, которые покрывают основные потребности проекта. Это позволило сосредоточиться на функциональной части и ускорить процесс разработки. В дальнейшем планируется провести кастомизацию интерфейса,

чтобы адаптировать его под специфические задачи и требования нашего проекта, улучшить пользовательский опыт и визуальное оформление.
Кроме того, в процессе работы мной был подготовлен и оформлен итоговый отчёт

1.2 Разбор требований заказчика и пользователей; план действий (backlog)

Основными требованиями заказчика были: создание карты сервиса, проектирование архитектуры. Так как проект нужный и актуальный, а также довольно сложный с технической точки зрения, нам нужно было реализовать хотя бы MVP, которое в дальнейшем мы будем улучшать.

Сбор и классификация требований

– функциональные (учёт прихода/ухода, отчёты, роли),

– нефункциональные (производительность, безопасность, удобство).

Приоритизация (MoSCoW)

– **must** (базовая регистрация, авторизация, сканирование меток),

– **should** (аналитика, фильтры, экспорт отчётов),

– **could** (интеграция с другими системами).

Backlog

– User Story 1: «Как участник … сканирую метку и вижу подтверждение»

– User Story 2: «Как организатор … создаю новое мероприятие»

Оценка задач (story points) и распределение по спринтам

Работа над проектом была разбита на несколько спринтов продолжительностью 2–3 недели каждый. Всего было проведено 2–3 спринта, в ходе которых команды планировали и выполняли задачи.

Каждая задача оценивалась в story points с учётом её сложности и объёма работы. Это позволило сбалансировать нагрузку и оптимально распределить усилия внутри команды.

- Первый спринт включал основные задачи по проектированию архитектуры, настройке серверной части и созданию базового функционала интерфейса. Здесь большинство задач оценивалось в 3–5 story points.
- Второй спринт был посвящён доработке клиентской части, интеграции с API, реализации авторизации и учёта участников. Задачи имели среднюю оценку 2–4 story points.

1.3 Анализ и сопоставление аналогов

В рамках выбора фронтенд-решения для административной панели нашего проекта был проведён анализ популярных сервисов и шаблонов, построенных на базе React Admin — фреймворка с готовым UI и набором компонентов для быстрой разработки административных интерфейсов.

Аналоги, рассмотренные для сопоставления:

- React Admin Default Template
  - Описание: Стандартный стартовый шаблон React Admin, включающий основные компоненты: списки, формы, фильтры, панели навигации.
  - Преимущества: Быстрая настройка, современный UI, поддержка различных типов данных и API.
  - Ограничения: Минимальная кастомизация дизайна «из коробки», требует дополнительной доработки под специфические задачи.
- Marmelab Demo
  - Описание: Демонстрационный проект от разработчиков React Admin с расширенным функционалом, включая мульти-ресурсное управление, расширенную фильтрацию и аналитику.
  - Преимущества: Хорошо структурирован, пример комплексного использования React Admin, ориентирован на масштабируемость.
  - Ограничения: Сложность для новичков, требует адаптации под конкретные бизнес-процессы.

- Open Source ERP на React Admin (пример проекта)
  - Описание: ERP-система с управлением складом, заказами и клиентами, построенная с помощью React Admin.
  - Преимущества: Полезные для бизнеса функции, интеграция с REST API, адаптивный дизайн.
  - Ограничения: Часто требует доработки под конкретные нужды и корпоративные стандарты.

Сопоставление с нашим проектом. Как и в React Admin Default Template, мы используем готовые UI-компоненты для быстрого старта и удобного управления данными. Это позволяет существенно сократить время разработки. В будущем планируется кастомизация интерфейса по аналогии с Marmelab Demo — расширение функционала, улучшение UX/UI и адаптация под задачи контроля посещаемости. В отличие от комплексных ERP-систем, наш проект сфокусирован на специфичной предметной области — учёте участников мероприятий, что упрощает требования к интерфейсу и позволяет быстрее адаптировать React Admin под наши нужды.

1.4 Архитектура программного продукта

Общая схема (приложение А и приложение Б)

Совместно с нашим куратором была спроектирована архитектура веб-сервиса, учитывающая основные бизнес-процессы и роли пользователей. В административной панели предусмотрено создание различных уровней доступа:

- Супер-администраторы — имеют полный доступ ко всем разделам и настройкам сервиса.

- Менеджеры и волонтёры — обладают ограниченными правами, позволяющими управлять только определёнными функциями (реализация этих ролей в процессе разработки).

Основной объект системы — Ивент (мероприятие), на которое регистрируются участники. Регистрация может происходить двумя способами: через онлайн-форму регистрации или вручную, через волонтёров. На самом мероприятии предусмотрены различные активности — например, столовая, место получения мерча и другие зоны. В каждой активности работают волонтёры, которые отмечают присутствие участников, используя, например, сканирование QR-кодов на бейджах.

У каждого участника есть определённые роли, накладывающие ограничения на его возможности и доступ к информации. Для каждой активности предусмотрены веб-страницы с дизайном, адаптированным под конкретное мероприятие или зону. Там можно отслеживать статус участника — например, прошёл ли он через активность.

Администрация имеет возможность создавать и настраивать формы регистрации, выбирая необходимые поля и адаптируя форму под специфику конкретного мероприятия. Это обеспечивает гибкость и удобство в управлении процессом учёта участников.

Основные компоненты:

– Мероприятия

– Активности

– Роли

– Администрация

– Участники

– NFC/QR-метки

– Форма регистрации

Взаимодействие

– HTTPS-запросы клиент ↔ сервер

Обоснование выбора:

– Django для быстрого старта и готовой админки

– React-Admin для админ-панели (минимум кастомного кода)

– MySQL — надёжность и расширяемость

1.5 Методология разработки

Процесс разработки:

В рамках проекта мы использовали гибкую методологию Agile с разбивкой работы на спринты продолжительностью по 3 недели. Такой подход позволял регулярно оценивать прогресс, быстро реагировать на изменения и своевременно корректировать план работ. По окончании каждого спринта проводились совещания, на которых обсуждались результаты, планировались следующие задачи и уточнялись требования.

Инструменты разработки:

Для управления версиями кода и совместной работы команда использовала систему контроля версий Git и платформу GitHub. Это обеспечило прозрачность изменений, удобное ведение веток и быструю интеграцию новых функций в общий код проекта.

1.6 Планирование и распределение задач

Планирование и распределение задач велось с помощью сервиса weeek.net, позволившему сделать совместную работу организованной и эффективной.

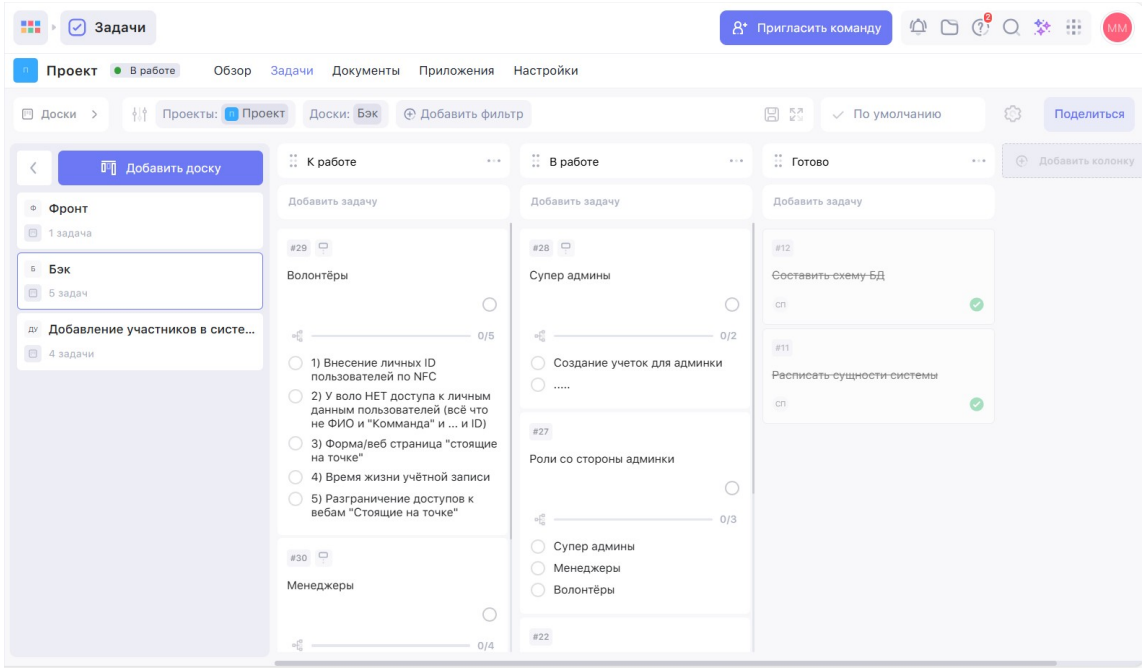


Рисунок 1 – Скриншот рабочей среды сервиса weeek.net

ЗАКЛЮЧЕНИЕ

Оценка соответствия продукту требованиям заказчика и пользователей

В ходе работы над проектом был создан минимально рабочий прототип системы, который обеспечивает выполнение ключевых функций: создание мероприятий, ручное добавление участников, отметка их присутствия, а также управление учётными записями супер-администраторов.

Оценка качества на основе результатов тестирования

На данный момент приложение стабильно: все готовые на данный момент в API запросы выполняются. Вместе с тем, из-за задержек, связанных с проектированием архитектуры и планированием, часть задач ещё не реализована, что ограничивает текущую полноту продукта. Тем не менее, выявленные на этом этапе недостатки не влияют на базовую работоспособность и не препятствуют дальнейшему развитию системы.

Предложения по улучшению и дальнейшему развитию

В дальнейшем планируется доработка базового прототипа в рамках учебной практики, включающая добавление поддержки активностей, отметки участников с помощью QR/NFC-меток, расширение ролей до менеджеров и волонтёров, а также внедрение динамических форм регистрации. Эти шаги позволят повысить функциональность и удобство использования системы, а также расширить её возможности для анализа посещаемости.

Рекомендуется уделить особое внимание дальнейшей кастомизации интерфейса и интеграции аналитических модулей, что повысит ценность продукта для организаторов мероприятий. Кроме того, важно продолжить тестирование с привлечением реальных пользователей для своевременного выявления и устранения узких мест и ошибок.

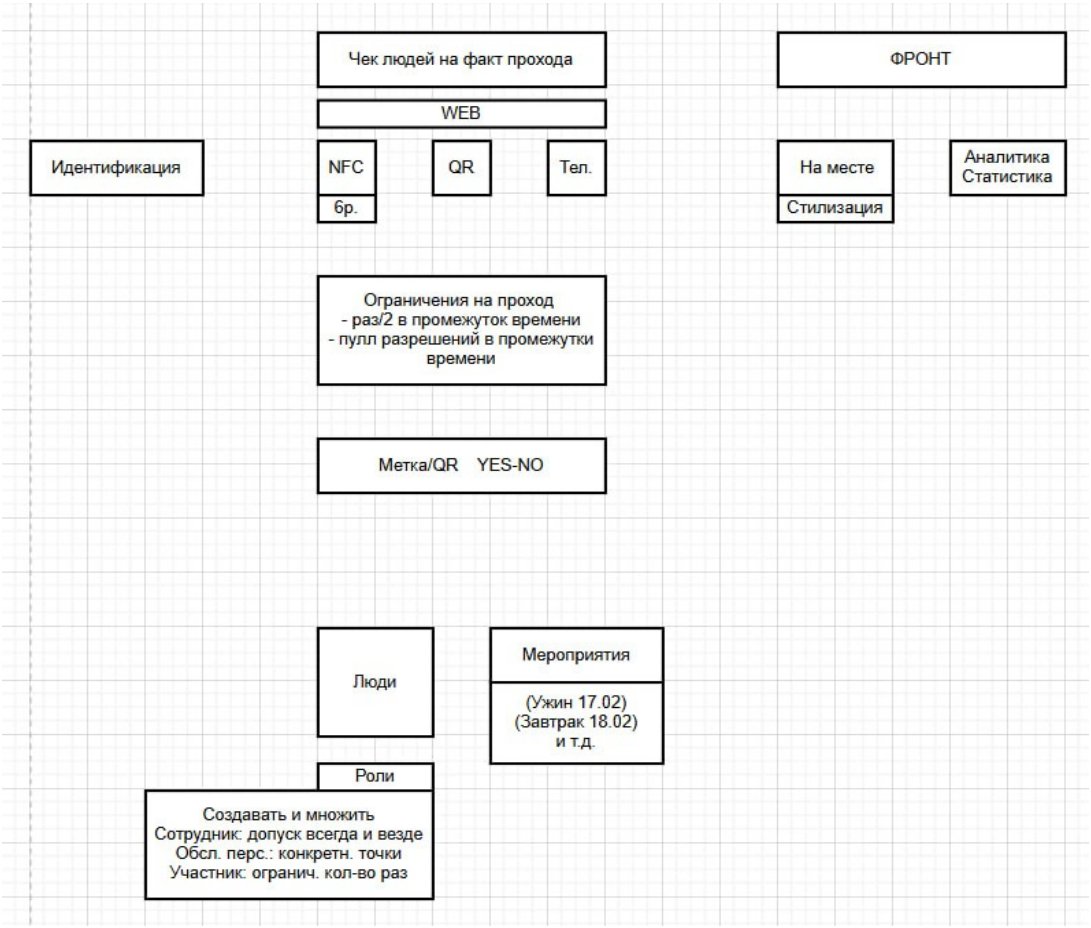
Таким образом, несмотря на неполное выполнение всех задач на текущем этапе, проект демонстрирует устойчивый прогресс и обладает высоким потенциалом для успешного завершения и внедрения в реальных условиях.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Marmelab. React-admin: документация. [Электронный ресурс]. URL: https://marmelab.com/react-admin/documentation.html

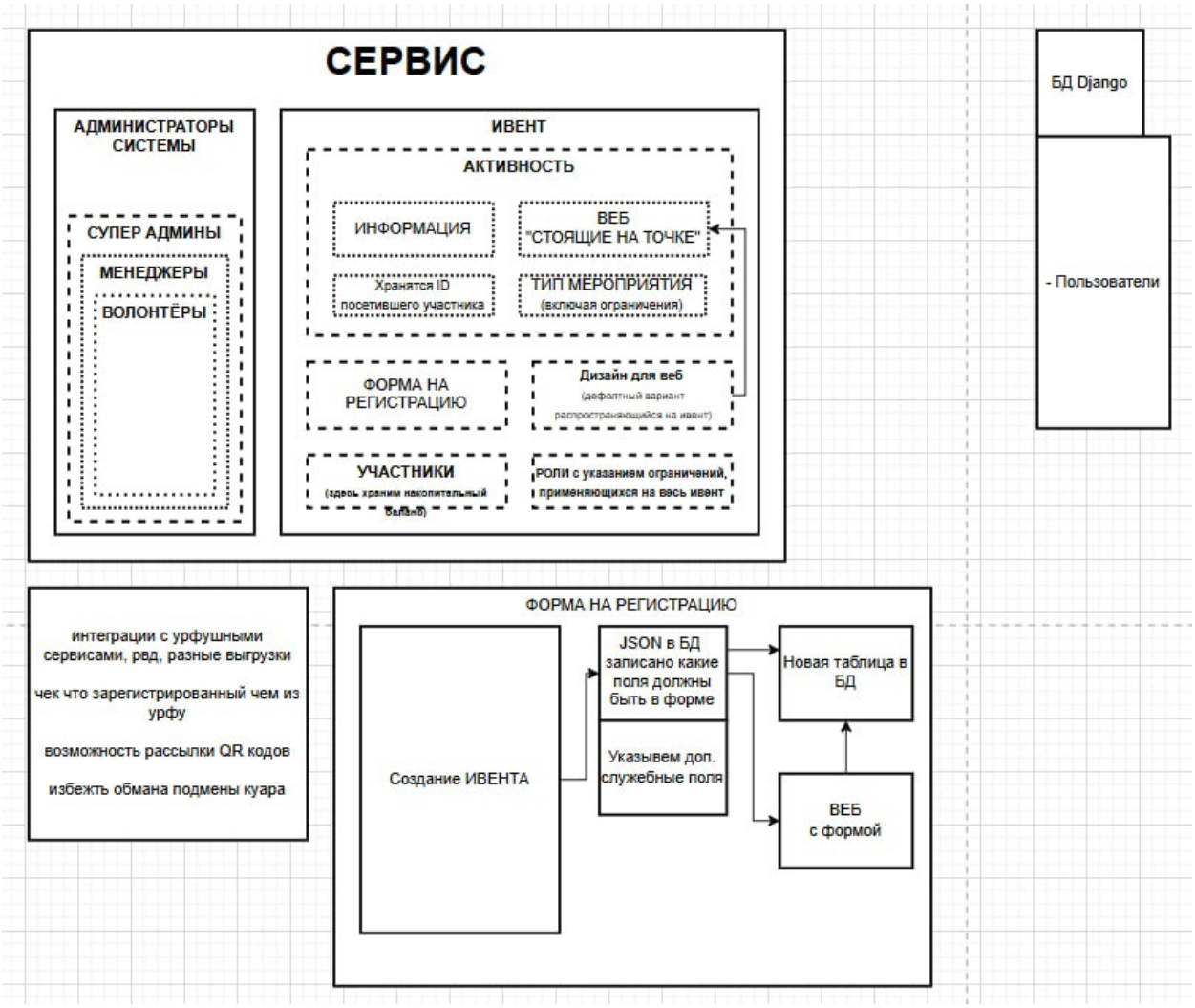
ПРИЛОЖЕНИЕ А

Функциональная схема



ПРИЛОЖЕНИЕ Б

Архитектура сервиса



ПРИЛОЖЕНИЕ В

Блок-схема карты веб-сервиса