

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 CRM-модуля для автоматизации приёма заявок на практику и  
стажировку»  
по дисциплине «Проектный практикум»

Заказчик: Смирнов Денис Сергеевич  
Куратор: Пушкарь Юрий Андреевич  
ученая степень, ученое звание, должность  
Студенты команды ByteBuilders  
Васильцов Владимир Николаевич  
Алексеев Егор Алексеевич  
Вяткина Софья Романовна  
Шляпникова Дарья Егоровна

Екатеринбург, 2025

# СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	4
1 План работы над проектом .....	6
2 Список участников .....	7
3 Стек технологий .....	8
3.1 Frontend .....	8
3.2 Backend.....	8
4 Обзор аналогов .....	9
4.1 Аналоги .....	9
4.2 Достоинства и недостатки аналогов.....	9
5 Описание работы решения.....	11
5.1 Создание мероприятия .....	11
5.2 Настройка этапов обработки заявок (CRM-доска) .....	11
5.3 Настройка формы подачи заявки.....	12
5.4 Настройка автоматизаций: роботы и триггеры.....	13
5.5 Подача заявки студентом .....	15
5.6 Обработка заявок в CRM .....	16
6 Отчет по работе каждого участника команды .....	18
6.1 Тимлид/аналитик .....	18
6.1.1 Организация встреч и распределение задач.....	18
6.1.2 Анализ аналогов .....	18
6.1.3 Проектирование бизнес-процессов .....	18
6.1.4 Проектирование пользовательских сценариев .....	19
6.1.5 Написание технического задания .....	20
6.2 Дизайнер .....	20
6.2.1 Дизайн канбан-доски.....	20
6.2.2 Дизайн страницы с формами .....	21
6.2.3 Дизайн создания, редактирования и удаления форм.....	22
6.2.4 Дизайн карточки заявки с информацией о студенте .....	22

6.2.5 Дизайн канбан-доски для роботов и триггеров .....	23
6.2.6 Дизайн форм для управления роботами и триггерами .....	24
6.3 Frontend-разработчик .....	24
6.3.1 Реализация канбан-доски .....	24
6.3.2 Реализация страницы с формами.....	25
6.3.3 Реализация функционала редактирования и удаления форм .....	26
6.3.4 Реализация интерфейса подачи заявки студентом .....	26
6.3.5 Реализация карточки заявки.....	27
6.3.6 Реализация канбан-доски для роботов и триггеров .....	28
6.3.7 Реализация форм управления роботами и триггерами .....	28
6.3.8 Настройка связи с сервером .....	29
6.4 Backend-разработчик.....	30
6.4.1 Реализация аутентификации руководителей .....	30
6.4.2 Реализация управления мероприятиями .....	31
6.4.3 Реализация управления профилями руководителей.....	31
6.4.4 Реализация пользовательских статусов заявок .....	32
6.4.5 Реализация форм мероприятий.....	32
6.4.6 Реализация заявок студентов .....	33
6.4.7 Реализация автоматических роботов.....	33
6.4.8 Реализация триггеров статусов .....	34
6.4.9 Реализация управления триггерами.....	34
6.4.10 Реализация Telegram-бота .....	35
ЗАКЛЮЧЕНИЕ .....	36
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....	37
ПРИЛОЖЕНИЕ А План работы над проектом .....	39

## **ВВЕДЕНИЕ**

В современных образовательных и корпоративных структурах процесс приёма студентов на практику или стажировку, а также их интеграция в рабочие процессы, требует значительных организационных и временных ресурсов. На практике, несмотря на важность этих процессов, они зачастую реализуются вручную с применением разрозненных инструментов — таблиц, мессенджеров, почты и форм. Это приводит к высокой нагрузке на сотрудников, задержкам в обработке заявок и затруднённой коммуникации между участниками.

Ключевые сложности, с которыми сталкиваются участники процесса:

- а) Руководители** тратят значительное время на обработку заявок, ручное управление статусами и индивидуальную коммуникацию со студентами.
- б) Студенты** сталкиваются с неудобным и запутанным процессом подачи заявок, недостаточной информацией о ходе рассмотрения и отсутствием понятных уведомлений о последующих действиях.

**Цель проекта:** разработать CRM-систему для автоматизации управления заявками студентов на практику или стажировку. Система должна упростить работу руководителя за счёт сокращения времени на обработку заявок, коммуникацию с участниками и отслеживание статусов. Для студентов CRM обеспечит удобную подачу заявок без регистрации на сайте, с возможностью заполнения персональных данных через форму, создаваемую руководителем, и получение всех необходимых уведомлений через Telegram-бота. Интеграция с внешними платформами тестирования позволит автоматически учитывать результаты прохождения тестов и обеспечит плавный переход по этапам внутри системы. Внедрение проекта повысит прозрачность процесса, ускорит обработку заявок и улучшит взаимодействие между руководителями и студентами.

Для достижения поставленной цели необходимо решить следующие задачи:

- 1) разработать CRM-систему для ведения заявок;
- 2) реализовать возможность создания индивидуальных форм сбора данных руководителями;
- 3) интегрировать Telegram-бота для отправки сообщений студентам;
- 4) реализовать логику автоматического перехода по этапам (статусам) в зависимости от действий студента;
- 5) обеспечить интеграцию с внешней платформой тестирования.

Реализация этих задач позволит создать гибкую и удобную платформу, способствующую оптимизации работы руководителей и улучшению опыта студентов при подаче и обработке заявок. Автоматизация ключевых процессов обеспечит минимизацию ошибок, ускорит принятие решений и повысит прозрачность всего цикла стажировки. В результате проект станет эффективным инструментом для современных образовательных и корпоративных организаций, стремящихся к цифровой трансформации и улучшению взаимодействия между участниками процесса.

## **1 План работы над проектом**

Работа над проектом велась поэтапно с использованием гибкой методологии разработки, где весь процесс был разбит на недельные спринты. В начале каждого спрингта команда определяла цели, приоритетные задачи и зоны ответственности участников, а в конце — подводились итоги и анализировались достигнутые результаты.

Основное внимание в проекте уделялось созданию CRM-системы, предназначеннной для автоматизации процессов взаимодействия между руководителями и студентами. Система охватывает весь цикл обработки заявок: от создания мероприятий и настройки этапов обработки до сбора данных и реализации автоматических действий. Ключевым компонентом стал интерфейс CRM в виде канбан-доски, позволяющий руководителю гибко управлять этапами обработки заявок.

Также была реализована система управления анкетами, где руководители настраивают формы для сбора персональных данных студентов. Важной частью проекта стало создание карточек заявок, в которых отображаются все необходимые сведения о студенте и его статусе в рамках мероприятия.

Значительное внимание уделялось автоматизации бизнес-процессов: был реализован функционал настройки роботов и триггеров, которые упрощают управление заявками и обеспечивают автоматическую отправку уведомлений студентам через Telegram-бота.

Подробный поэтапный план работы над проектом с указанием задач, сроков реализации, планируемых результатов и ответственных участников приведён в приложении А (План работы над проектом).

## **2 Список участников**

Васильцов Владимир Николаевич (Тимлид, аналитик) — координирует работу команды, распределяет задачи и контролирует выполнение этапов проекта. Он анализирует требования, взаимодействует с заинтересованными сторонами и формирует четкие задачи для команды. Он следит за сроками, качеством работы и решает возникающие организационные вопросы.

Шляпникова Дарья Егоровна (Дизайнер) — разрабатывает визуальный интерфейс CRM-системы, создает макеты страниц и следит за их точной реализацией. Она работает в тесном сотрудничестве с разработчиками, чтобы обеспечить удобство и функциональность интерфейса, а также контролирует соблюдение дизайнерских стандартов и стиля.

Вяткина Софья Романовна (Frontend-разработчик) — реализует макеты и верстку клиентской части системы. Она отвечает за отображение всех визуальных элементов и их функциональность на фронтенде, интегрирует интерфейс с серверной частью и работает над оптимизацией скорости и адаптивности интерфейса.

Алексеев Егор Алексеевич (Backend-разработчик) — разрабатывает серверную часть системы, проектирует API, интегрирует базу данных и обеспечивает безопасность данных. Он отвечает за обработку информации, взаимодействие с фронтендом и оптимизацию серверной части для высокой скорости и надежности работы системы.

### **3 Стек технологий**

#### **3.1 Frontend**

React — библиотека для построения пользовательских интерфейсов, которая позволяет создавать динамичные и интерактивные приложения с использованием компонентов, что улучшает гибкость и повторное использование кода.

TypeScript — надстройка над JavaScript, добавляющая строгую типизацию, что улучшает надежность кода, помогает предотвратить ошибки и ускоряет разработку, особенно в крупных проектах.

React и TypeScript выбраны для фронтенда, так как React обеспечивает высокую производительность и переиспользуемые компоненты, что важно для динамичных CRM-приложений. TypeScript добавляет типизацию, повышая надежность кода, облегчая масштабирование и ускоряя поиск ошибок.

#### **3.2 Backend**

Java — язык программирования, известный своей стабильностью, производительностью и безопасностью, который широко используется для создания масштабируемых серверных приложений.

PostgreSQL — реляционная база данных с открытым исходным кодом, известная своей производительностью, поддержкой сложных запросов и высокой надежностью для работы с большими объемами данных.

Для бэкенда выбраны Java и PostgreSQL, так как Java обеспечивает надёжность, стабильность и масштабируемость серверных приложений, а PostgreSQL подходит для работы с реляционными данными, обеспечивая высокую производительность и поддержку сложных запросов.

## 4 Обзор аналогов

### 4.1 Аналоги

В рамках исследования существующих решений были проанализированы следующие аналоги CRM-систем, которые могут быть использованы для автоматизации процессов подачи заявок и взаимодействия с участниками:

- **Bitrix24** — это многофункциональная CRM-система, которая предоставляет решения для автоматизации процессов, управления проектами, задачами и коммуникациями;
- **HubSpot CRM** — это популярная система для управления взаимодействием с клиентами, которая также может быть адаптирована для образовательных нужд;
- **CiviCRM** — это открытая и бесплатная CRM-система, ориентированная на некоммерческие организации и образовательные учреждения;
- **Zoho CRM** — это облачное решение, которое помогает автоматизировать взаимодействие с клиентами и управления данными;
- **Freshworks CRM** — это система для управления продажами и взаимодействием с клиентами, которая также может быть использована для автоматизации процесса приёма студентов на проекты.

### 4.2 Достоинства и недостатки аналогов

Для оценки существующих решений была проведена аналитика популярных CRM-систем с точки зрения их применимости для автоматизации процесса подачи и обработки заявок студентов, сбора персональных данных, а также взаимодействия между студентами и руководителями. В таблице представлены ключевые достоинства и недостатки каждого решения с

акцентом на удобство работы для двух основных категорий пользователей: руководителей и студентов.

Таблица 1 – Достоинства и недостатки

	Достоинства	Недостатки
Bitrix24	Имеет мощные инструменты для обработки заявок студентов, их зачисления на проекты и создания организационных чатов. Предлагает надёжные механизмы защиты данных студентов и гибкую настройку прав доступа. Платформа предлагает бесплатный функционал.	Сложность интерфейса для новичков, требует обучения, что может затруднить работу руководителей. Несмотря на наличие бесплатной версии, для доступа ко всем расширенным функциям потребуется платная версия (12 000 ₽/мес).
HubSpot CRM	Интуитивно понятный интерфейс, подходящий как для студентов, так и для руководителей. Поддержка интеграции с множеством сервисов. Бесплатный базовый функционал.	Высокая стоимость при переходе на платные тарифы (до 300 000 ₽/мес). Отсутствие специализированных инструментов для управления проектами и стадиями отбора.
CiviCRM	Бесплатная система с открытым исходным кодом, подходящая для образовательных организаций. Возможность настройки форм и полей под нужды руководителей.	Интерфейс требует технических навыков, может быть неудобен для студентов. Сложность в самостоятельной настройке без ИТ-специалиста.
Zoho CRM	Гибкая настройка форм и интеграция с внешними сервисами. Возможность адаптации под образовательные процессы.	Ограниченные функции для управления стадиями отбора студентов. Платный доступ ко многим функциям (от 6 000 ₽/мес).
Freshworks CRM	Простой интерфейс и удобство использования как для руководителей, так и для студентов. Поддержка автоматизации заявок и данных.	Недостаточная функциональность. Платный доступ к расширенному функционалу (около 9 000 ₽/мес).

## 5 Описание работы решения

### 5.1 Создание мероприятия

Руководитель начинает с создания нового мероприятия. Он указывает название, описание и количество мест. После сохранения мероприятие появляется в общем списке доступных событий, и руководитель получает доступ к его детальной настройке.

### 5.2 Настройка этапов обработки заявок (CRM-доска)

Затем руководитель переходит на страницу CRM (см.Рисунок 1 – Страница «CRM»), где отображается канбан-доска. Каждый этап обработки заявок представлен в виде колонки. Чтобы добавить новый этап, руководитель нажимает кнопку «Добавить этап» (см.Рисунок 2 – Интерфейс добавления этапа), заполняет название, и этап появляется на доске.

Таким образом можно выстроить индивидуальный процесс обработки заявок, в зависимости от логики мероприятия.

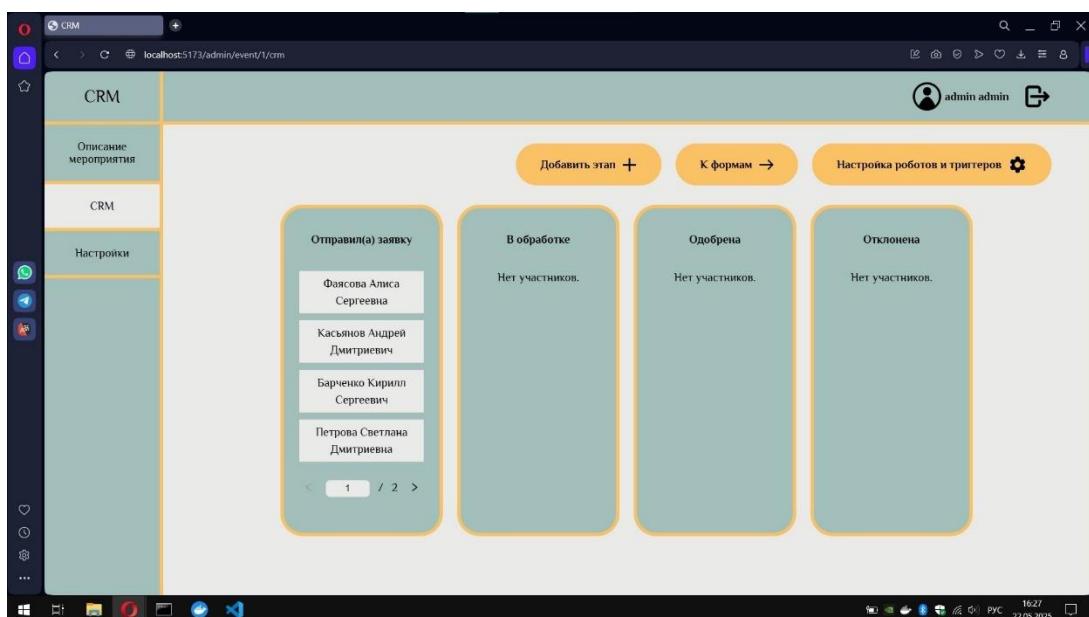


Рисунок 1 – Страница «CRM»

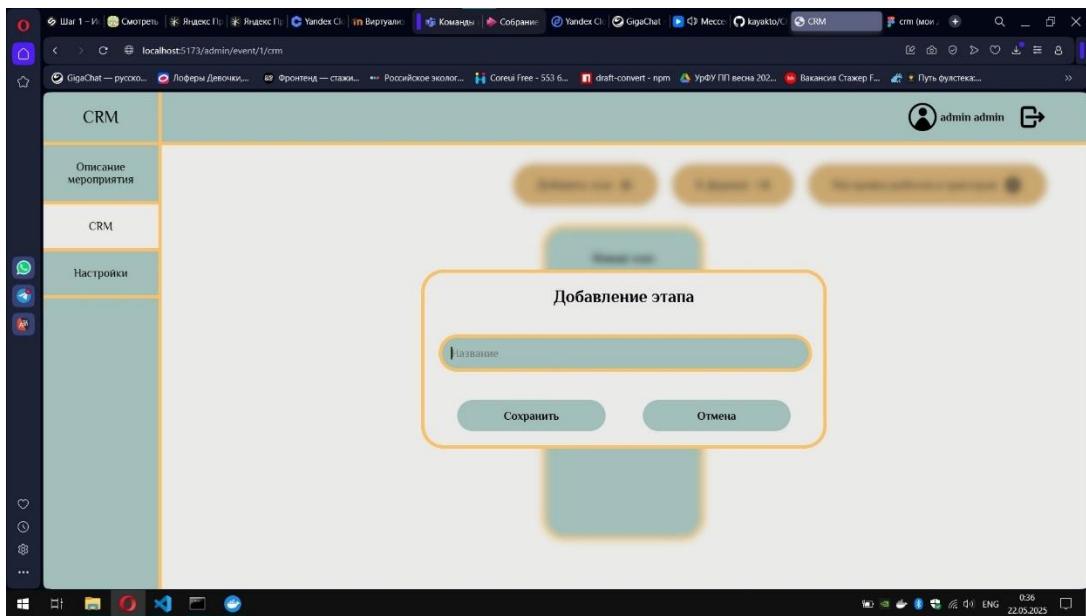


Рисунок 2 – Интерфейс добавления этапа

### 5.3 Настройка формы подачи заявки

После настройки этапов руководитель переходит в раздел «Формы» (см.Рисунок 3 – Страница «Формы»). Система автоматически создаёт базовую форму с основными полями: ФИО, Telegram, email. Формы отображаются списком, и каждую можно редактировать: при нажатии на иконку карандаша открывается интерфейс настройки формы (см.Рисунок 4 – Интерфейс редактирования формы). Здесь можно:

- изменить название;
- добавить дополнительные поля;
- настроить порядок и обязательность полей;
- удалить ненужные поля.

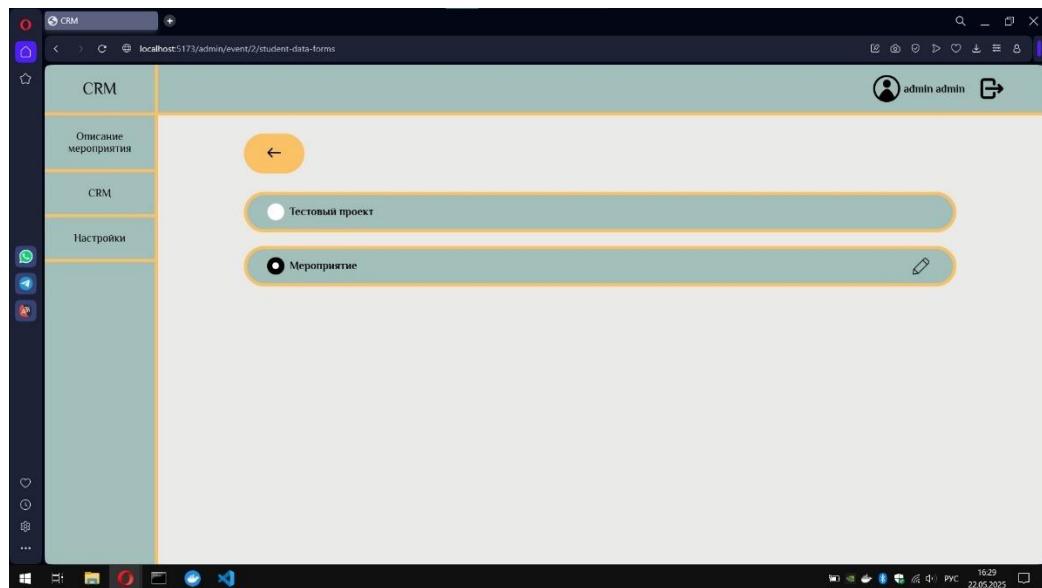


Рисунок 3 – Страница «Формы»

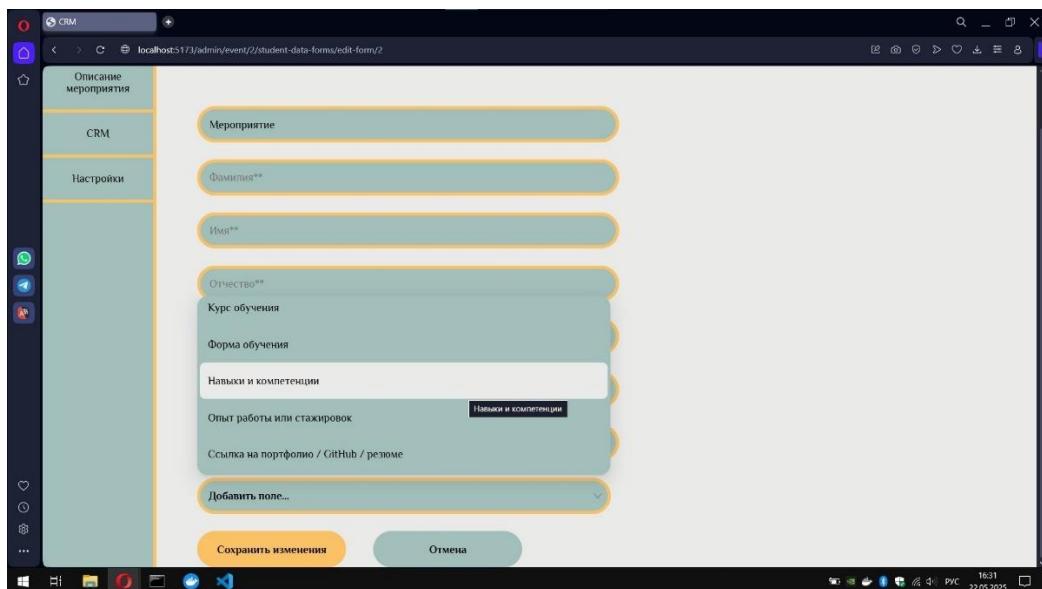


Рисунок 4 – Интерфейс редактирования формы

#### 5.4 Настройка автоматизаций: роботы и триггеры

Для настройки автоматических действий руководитель переходит на страницу «Роботы и триггеры», реализованную в виде канбан-доски (см. Рисунок 5 – Канбан-доска роботов и триггеров). Под каждым этапом можно добавлять роботов и триггеры. Для этого нужно навести курсор под

нужным этапом и нажать на кнопку «Добавить робота» или «Добавить триггер».

Далее открывается список доступных действий (см. Рисунок 6 – Список триггеров), после выбора открывается форма для заполнения параметров (см.Рисунок 7 – Поля триггера). После настройки робот или триггер появляется в нужной колонке.

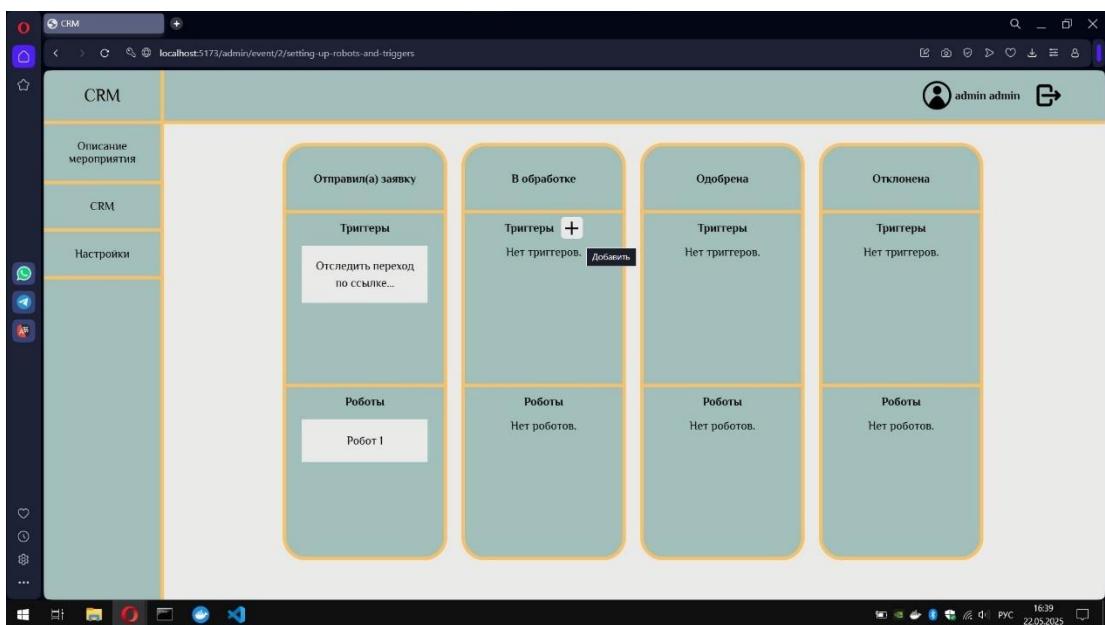


Рисунок 5 – Канбан-доска роботов и триггеров



Рисунок 6 – Список триггеров

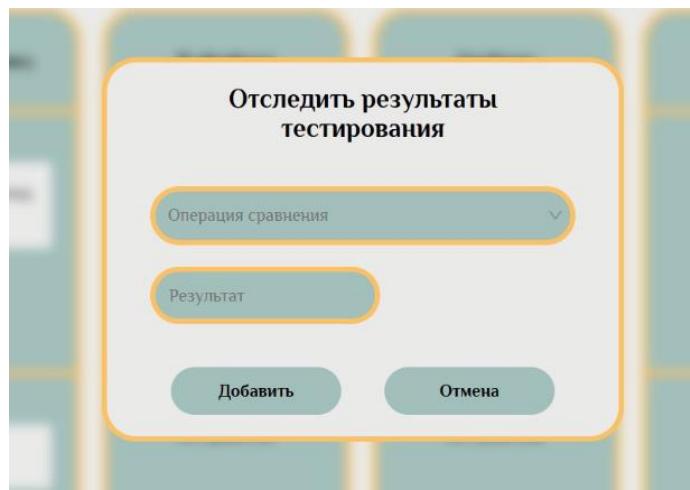


Рисунок 7 – Поля триггера

## 5.5 Подача заявки студентом

Студент заходит на сайт и просматривает список мероприятий. Выбрав интересующее, он переходит на его страницу, где видит описание и кнопку «Подать заявку». Открывается форма, которую предварительно настроил руководитель (см.Рисунок 8 – Форма подачи заявки). Студент заполняет и отправляет форму.

После подачи он получает сообщение от Telegram-бота с дальнейшими инструкциями(см.Рисунок 9 – Сообщение от Telegram-бота).

Рисунок 8 – Форма подачи заявки

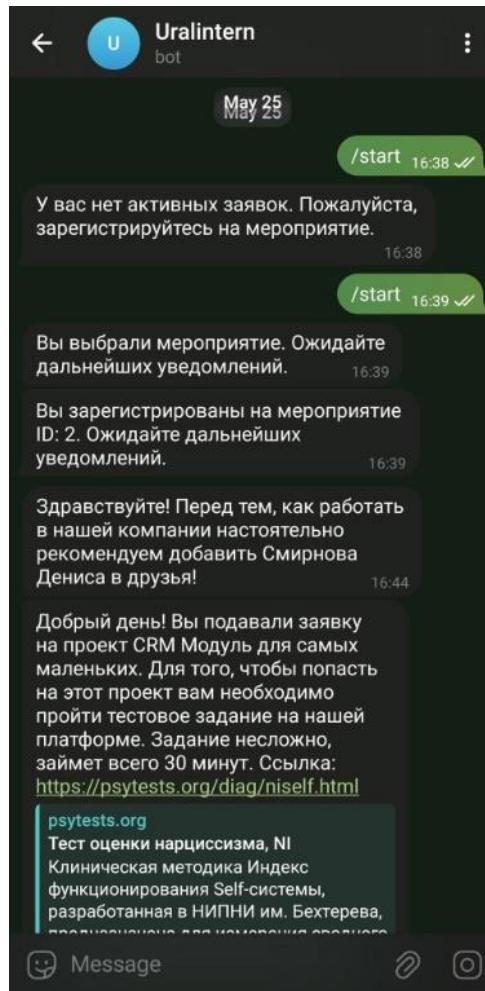


Рисунок 9 – Сообщение от Telegram-бота

## 5.6 Обработка заявок в CRM

После того как студент подаёт заявку, руководитель видит её в виде новой карточки на CRM-доске (см. Рисунок 10 – Карточка заявки с данными студента). Карточка содержит обязательные и дополнительные поля, заполненные студентом при подаче заявки.

Заявка автоматически перемещается между этапами обработки в соответствии с условиями, заданными через систему триггеров. Уведомления студентам отправляются через Telegram-бота благодаря заранее настроенным роботам. Это делает процесс обработки заявок более прозрачным, автоматизированным и удобным как для руководителя, так и для участников.

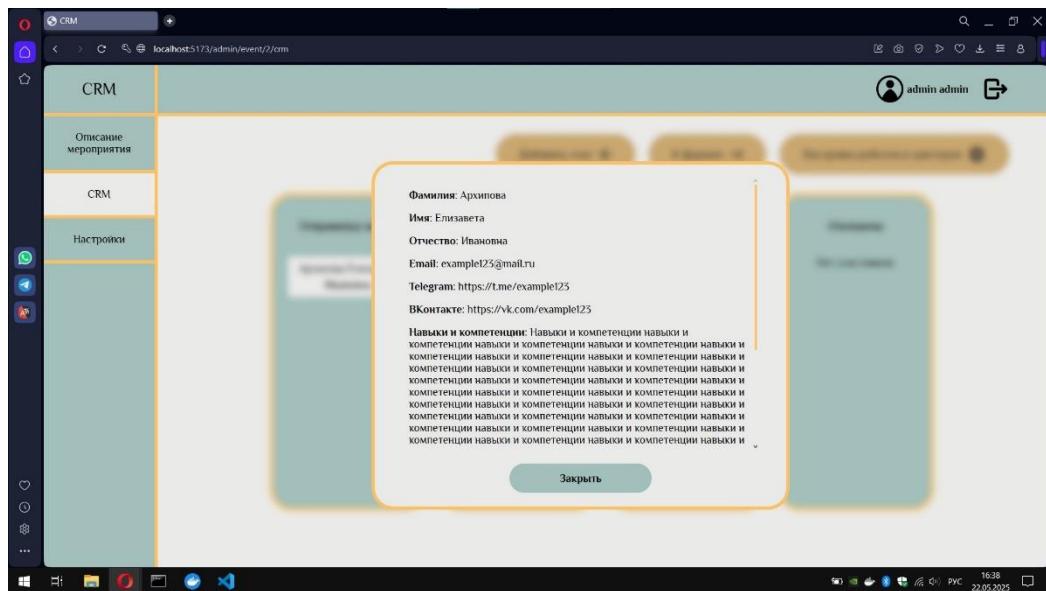


Рисунок 10 – Карточка заявки с данными студента

## **6 Отчет по работе каждого участника команды**

### **6.1 Тимлид/аналитик**

#### **6.1.1 Организация встреч и распределение задач**

Владимир организовывал регулярные встречи с командой для обсуждения текущих задач, приоритетов и планирования дальнейших шагов. Он также занимался распределением задач между участниками команды, основываясь на их компетенциях и загрузке. Это помогло обеспечить своевременное выполнение работ и четкое распределение ответственности, а также оптимизировать рабочие процессы.

#### **6.1.2 Анализ аналогов**

Владимир провел анализ существующих аналогичных систем для приема студентов на практику и стажировку. Это включало изучение функциональных возможностей и выявление слабых мест, которые можно улучшить в разрабатываемой CRM-системе. Результаты анализа помогли определить ключевые функции для реализации и избежать повторения ошибок, встреченных в других решениях.

#### **6.1.3 Проектирование бизнес-процессов**

Владимир также реализовал схемы бизнес-процессов, подробно описывающие этапы взаимодействия между руководителями и студентами в системе. Он разработал логику, включающую создание и настройку статусов на доске, формирование форм для сбора персональных данных, настройку роботов, отправляющих уведомления в Telegram, и триггеров, проверяющих результаты тестирования и переходы по ссылкам из сообщений. Это

позволило разработчикам лучше понять логику и последовательность процессов, обеспечив более точную реализацию функционала системы и ускорив этапы разработки.

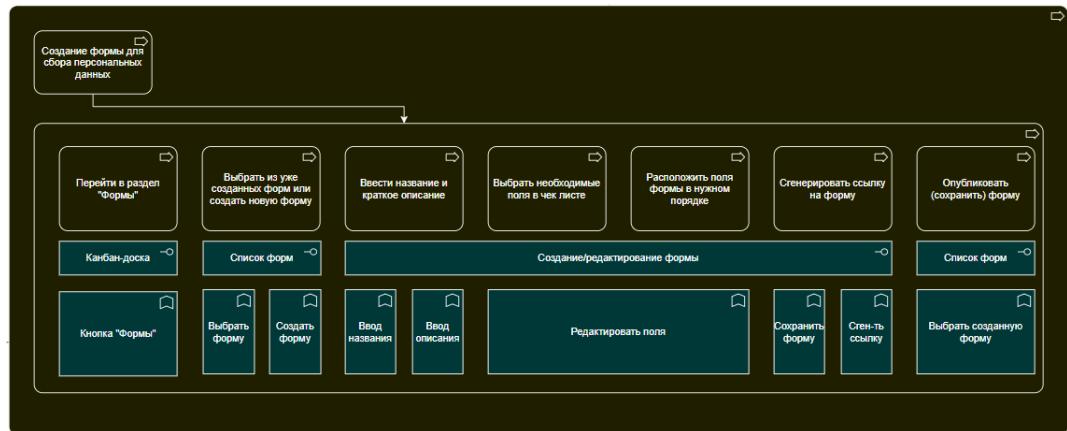


Рисунок 11 – Бизнес-процесс создания форм для сбора персональных данных

#### 6.1.4 Проектирование пользовательских сценариев

Владимир создал подробный user flow, который отразил все ключевые шаги настройки этапов прохождения заявки на мероприятие со стороны руководителей, а также процесс подачи заявки студентом. Этот визуальный сценарий помог выявить и структурировать взаимодействия пользователей с системой, определить возможные точки затруднений и улучшить логику переходов между этапами.

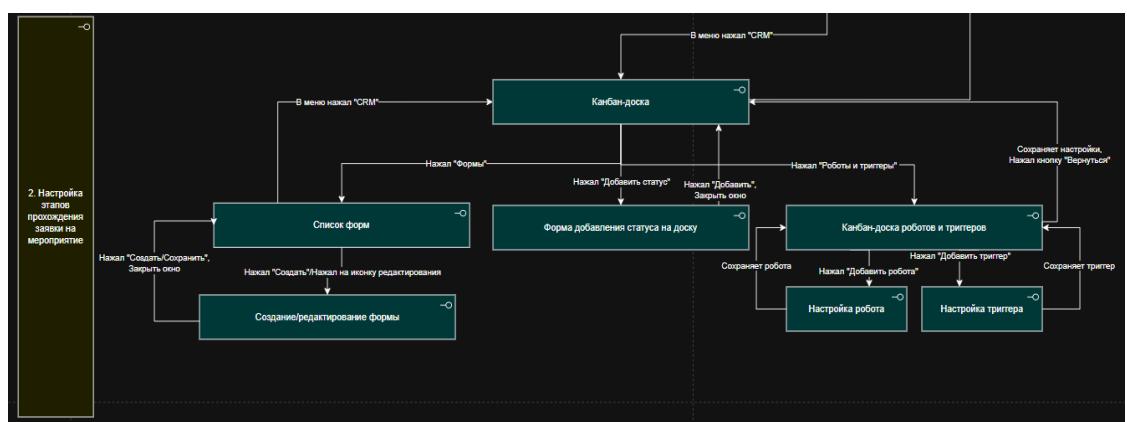


Рисунок 12 – Пользовательский путь настройки этапов заявки

## **6.1.5 Написание технического задания**

Владимир подготовил подробное техническое задание для разработки CRM-системы, основываясь на проведённом анализе бизнес-процессов и пользовательских сценариев. В документе он чётко описал функциональные требования, ключевые возможности и интеграционные особенности системы. Такое структурированное техническое задание помогло обеспечить ясное понимание целей и задач проекта командой разработчиков, что повысило качество и скорость реализации системы.

## **6.2 Дизайнер**

### **6.2.1 Дизайн канбан-доски**

Дарья разработала удобную канбан-доску для настройки этапов обработки заявок в рамках мероприятия. Каждый этап представлен в виде колонки, куда помещаются карточки заявок. Реализована возможность добавления, редактирования, удаления и перемещения этапов. Также предусмотрены переходы к управлению формами и настройке автоматизаций (роботы и триггеры), что делает доску полноценным инструментом для настройки бизнес-процесса.

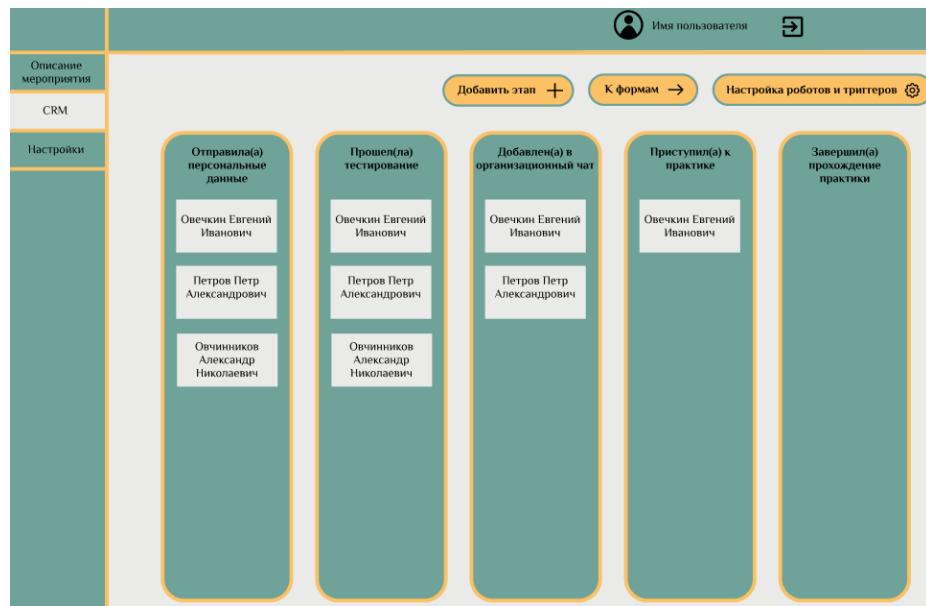


Рисунок 13 – Дизайн канбан-доски

### 6.2.2 Дизайн страницы с формами

Дарья разработала удобную страницу «Формы», где руководитель может просматривать, настраивать и управлять анкетами для сбора персональных данных участников мероприятия. Формы отображаются в виде списка с названиями и иконками редактирования и удаления. Формы можно прикреплять к мероприятиям и использовать повторно.

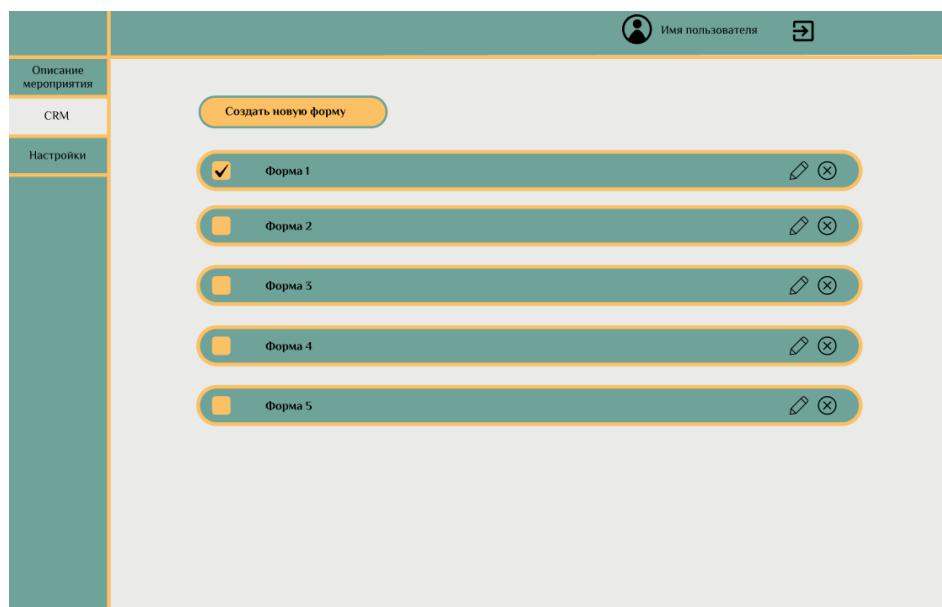


Рисунок 14 – Дизайн страницы «Формы»

### 6.2.3 Дизайн создания, редактирования и удаления форм

Дарья реализовала функционал, позволяющий руководителю создавать новые формы с обязательными полями (ФИО, tg\_url, email) и добавлять дополнительные из предложенного списка. Она разработала удобный интерфейс для редактирования названия, состава и порядка полей, а также для удаления форм с подтверждением.

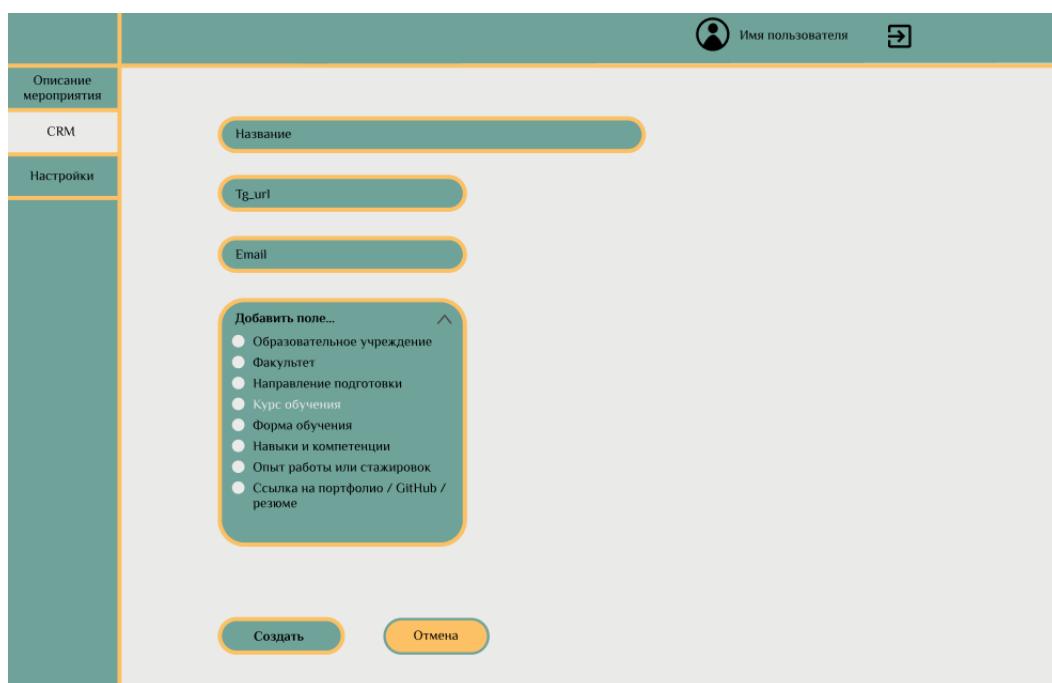


Рисунок 15 – Дизайн страницы создания формы

### 6.2.4 Дизайн карточки заявки с информацией о студенте

Дарья разработала компактную и информативную карточку заявки для канбан-доски CRM. В режиме просмотра на доске показывается только ФИО студента. По клику открывается окно с подробной информацией: контактами, заполненными полями анкеты, статусом заявки и результатами тестирования (если есть).

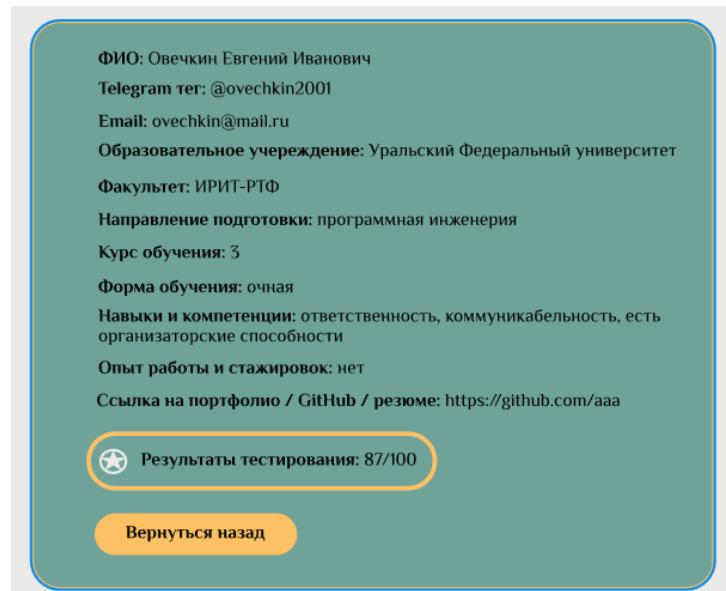


Рисунок 16 – Дизайн карточки заявки

### 6.2.5 Дизайн канбан-доски для роботов и триггеров

Дарья разработала дизайн страницы «Роботы и триггеры» — интерфейса для настройки автоматизаций в бизнес-процессах обработки заявок. Она создала канбан-доску, где каждый статус заявки отображается в виде колонки с зонами для триггеров и роботов, обеспечив прозрачное и удобное управление автоматическими действиями.

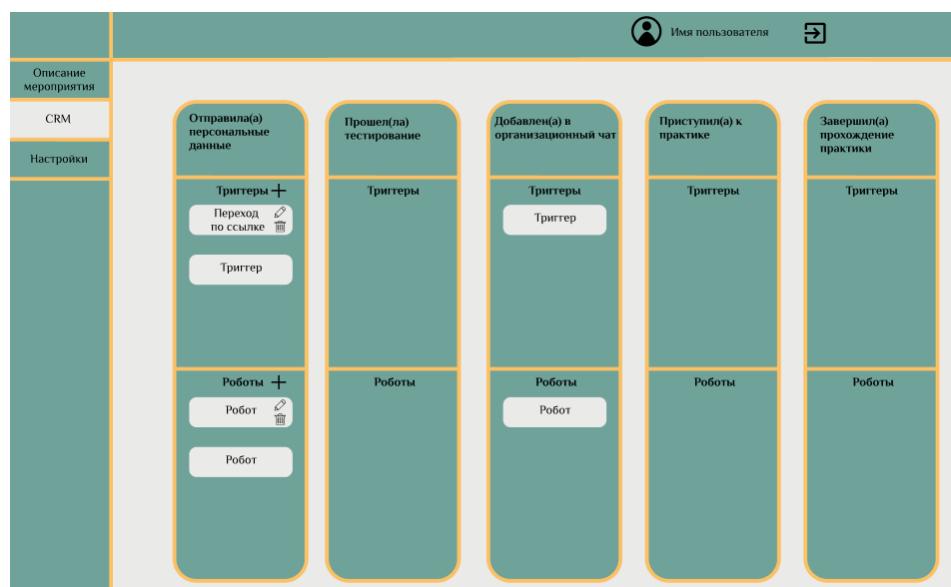


Рисунок 17 – Дизайн канбан-доски для роботов и триггеров

## 6.2.6 Дизайн форм для управления работами и триггерами

Для работы с роботами и триггерами Дарья спроектировала удобные формы добавления, редактирования и удаления, позволяющие руководителю легко настраивать условия срабатывания, параметры и порядок выполнения автоматизаций.



Рисунок 18 – Добавление робота

## 6.3 Frontend-разработчик

### 6.3.1 Реализация канбан-доски

Соня реализовала интерактивную канбан-доску для настройки этапов обработки заявок в рамках мероприятия. Каждый этап отображается в виде колонки с карточками заявок. Пользователь может добавлять, редактировать, удалять и перемещать этапы. Также предусмотрены переходы к управлению формами и автоматизациями (роботы и триггеры), что делает доску полноценным инструментом управления бизнес-процессом.

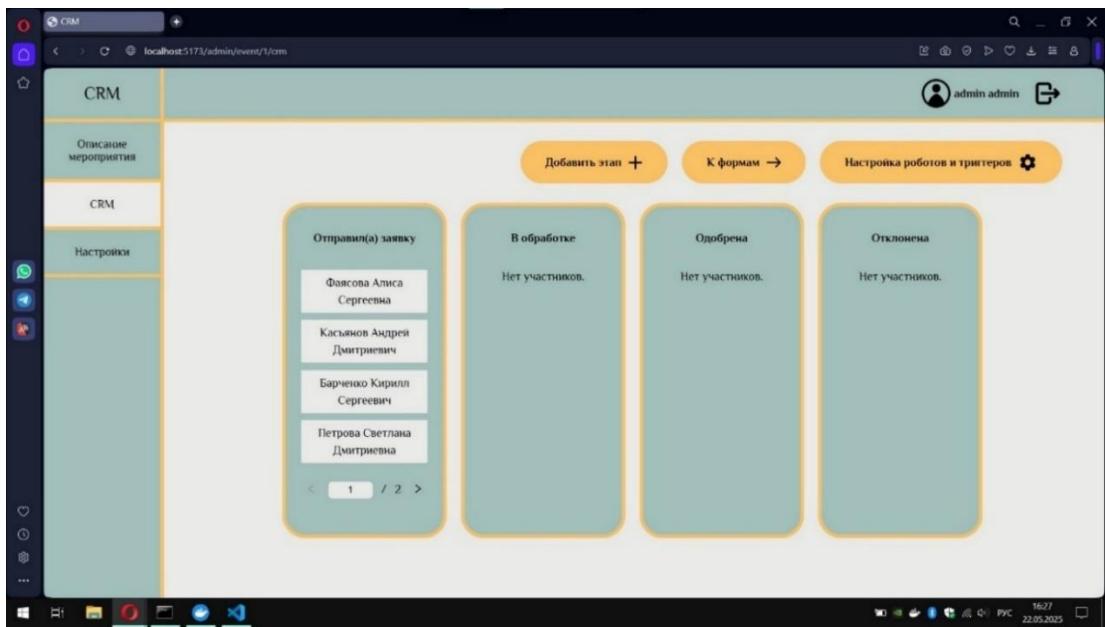


Рисунок 19 – Страница «CRM»

### 6.3.2 Реализация страницы с формами

Соня реализовала страницу «Формы» по предоставленному дизайну. Руководитель может просматривать список форм, редактировать и удалять их, а также прикреплять формы к мероприятиям. Интерфейс адаптирован под повторное использование форм и обеспечивает удобное управление анкетами.

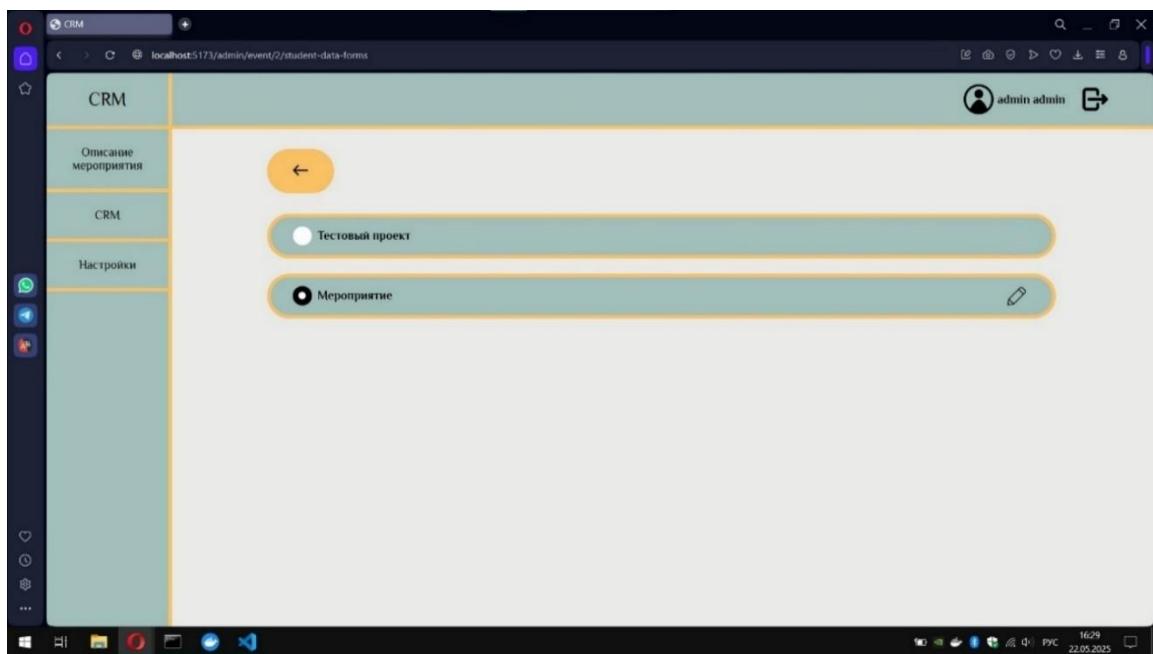


Рисунок 20 – Страница «Формы»

### 6.3.3 Реализация функционала редактирования и удаления форм

Соня разработала интерфейс для настройки форм. Руководитель может задавать название, редактировать состав и порядок полей, добавлять обязательные поля (ФИО, Telegram, email) и дополнительные поля из предложенного списка. Реализована функция удаления формы с подтверждением действия.

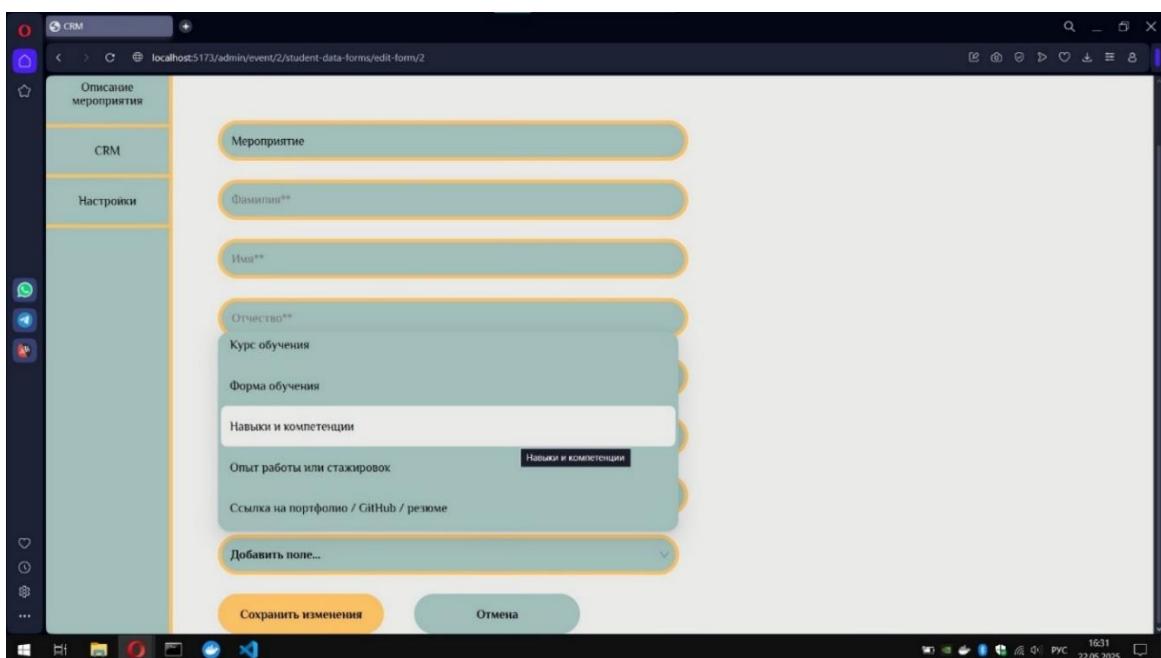


Рисунок 21 – Страница редактирования формы

### 6.3.4 Реализация интерфейса подачи заявки студентом

Соня реализовала страницу подачи заявки, на которой студент заполняет форму, предварительно настроенную руководителем. Интерфейс автоматически подгружает нужные поля в нужном порядке, включая обязательные поля (ФИО, Telegram, email) и дополнительные, если они были добавлены. Реализована валидация полей, отображение ошибок, а также подтверждение успешной отправки заявки.

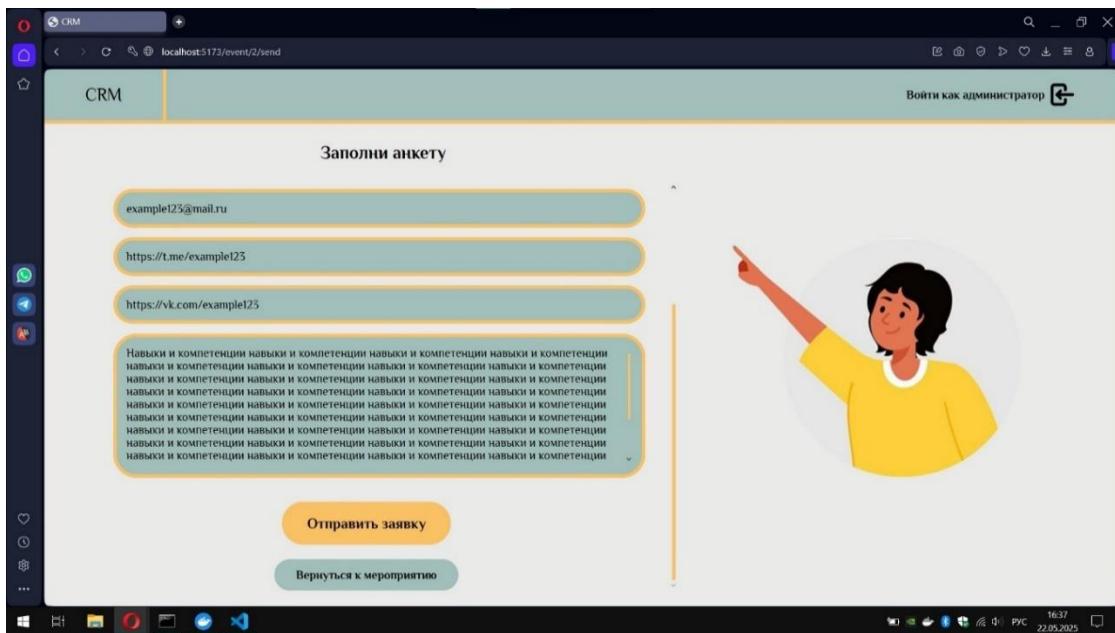


Рисунок 22 – Страница подачи заявки

### 6.3.5 Реализация карточки заявки

Соня реализовала компактную карточку заявки для отображения на канбан-доске. На доске видно только имя студента, а при клике открывается модальное окно с полной информацией: контактами, анкетными данными, текущим статусом и результатами тестов (если есть).

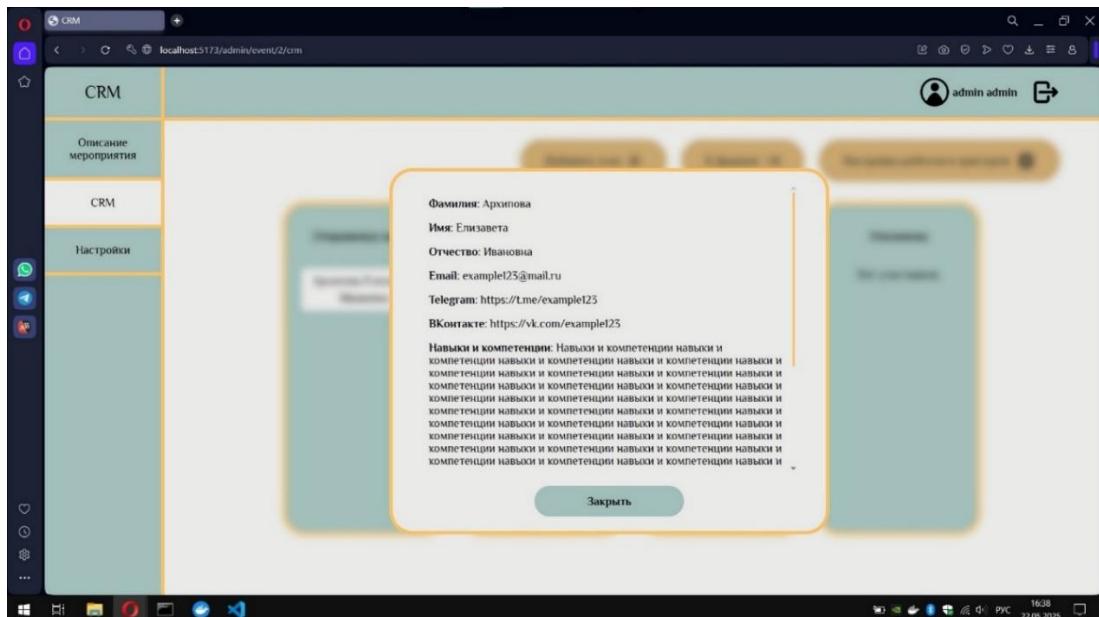


Рисунок 23 – Карточка заявки с информацией о студенте

### 6.3.6 Реализация канбан-доски для роботов и триггеров

Соня реализовала страницу «Роботы и триггеры» — инструмент настройки автоматизаций в рамках бизнес-процесса. По дизайну была создана канбан-доска, в которой каждый статус заявки представлен колонкой с зонами для триггеров и роботов. Это позволило обеспечить наглядную структуру автоматических действий и условий их срабатывания.

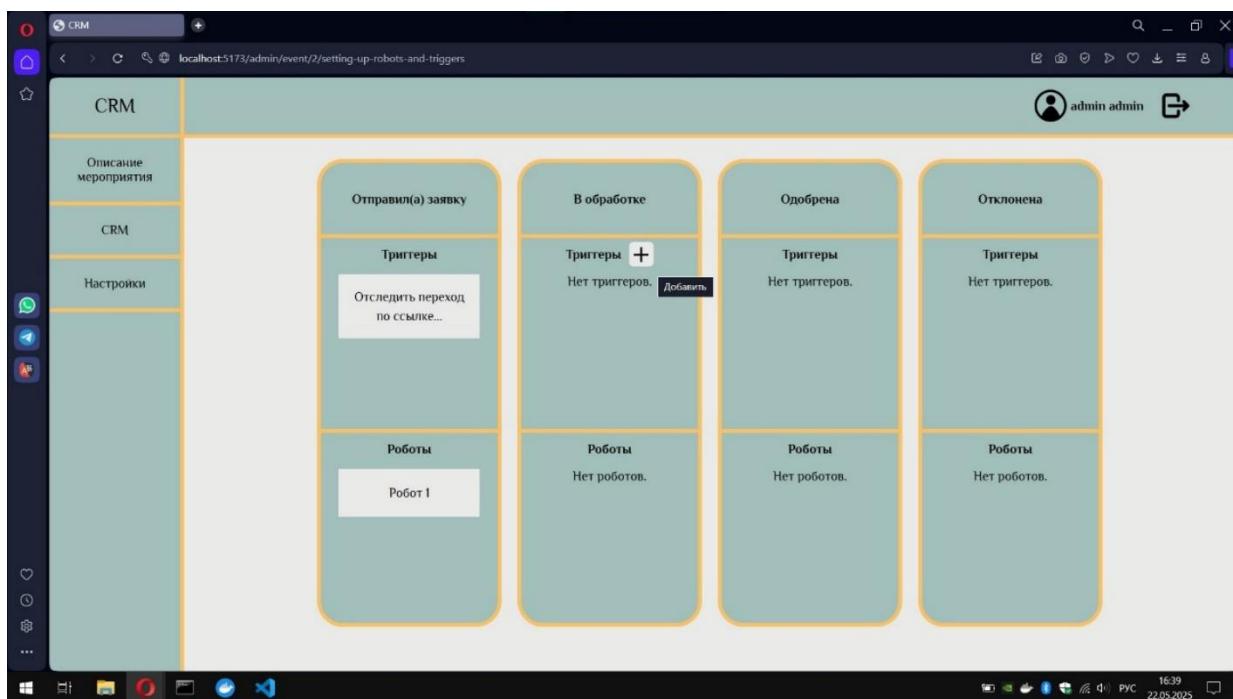


Рисунок 24 – Канбан-доска для роботов и триггеров

### 6.3.7 Реализация форм управления роботами и триггерами

Соня разработала функционал добавления, редактирования и удаления роботов и триггеров. Интерфейс позволяет легко настраивать условия, параметры и порядок выполнения автоматических действий. Реализована валидация полей и удобное взаимодействие с модальными окнами.

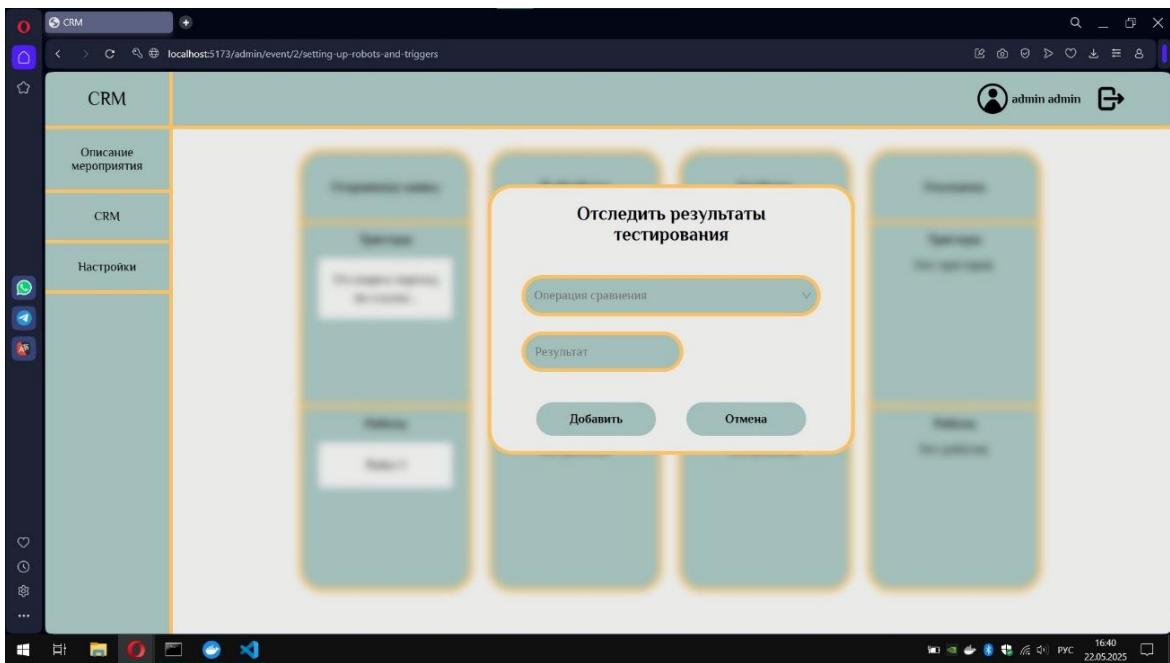


Рисунок 25 – Форма добавления триггера

### 6.3.8 Настройка связи с сервером

Одним из ключевых этапов работы стала настройка корректного взаимодействия фронтенда с серверной частью системы. Софья реализовала обработку запросов к API, обеспечив интеграцию пользовательского интерфейса с сервером, разработанным на Java и использующим базу данных PostgreSQL.

Были настроены запросы для получения, создания, обновления и удаления данных, связанных с этапами обработки заявок, карточками студентов, анкетами, бизнес-процессами, роботами и триггерами. Кроме того, Софья реализовала обработку ошибок, состояния загрузки и успешных ответов от сервера. Это позволило сделать интерфейс стабильным, отзывчивым и удобным для конечного пользователя.

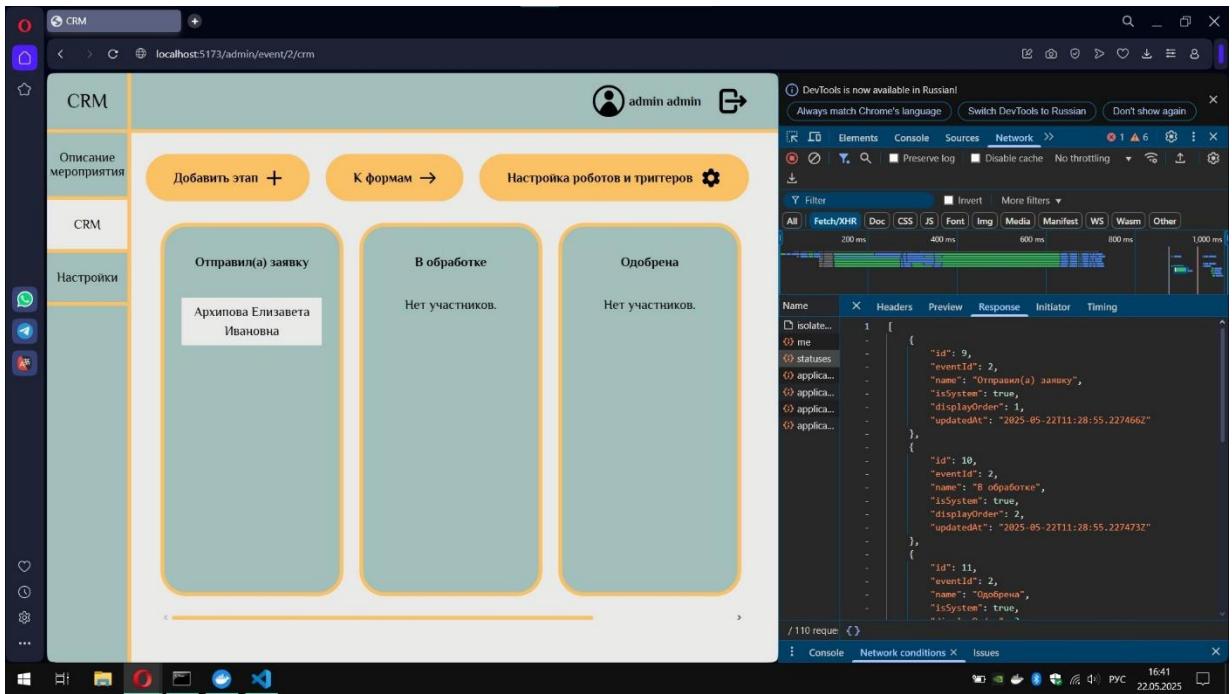


Рисунок 26 – Метод получения статусов на канбан-доске

## 6.4 Backend-разработчик

### 6.4.1 Реализация аутентификации руководителей

Егор реализовал контроллер, отвечающий за регистрацию и вход в систему руководителей. Регистрация осуществляется через реферальные ссылки, которые генерируют другие руководители. Контроллер также обрабатывает вход и обновление токенов доступа. Студенты не проходят регистрацию, они подают заявки через формы мероприятий.

auth-controller		
<code>POST</code>	<code>/auth/login</code>	
<code>POST</code>	<code>/auth/register</code> Регистрация пока не работает	
<code>POST</code>	<code>/auth/register-with-token</code> Регистрация пока не работает	
<code>POST</code>	<code>/auth/refresh</code>	

Рисунок 27 – Методы Auth Controller

## 6.4.2 Реализация управления мероприятиями

Егор разработал контроллер для управления мероприятиями. Руководители могут создавать, редактировать, скрывать, удалять и запускать мероприятия. Также реализован публичный доступ к мероприятиям с открытой регистрацией, через которые студенты подают заявки.

event-controller	
POST	/events/post
POST	/events/start-event
GET	/events/{eventId}
GET	/events/my
GET	/events/active
PUT	/events/update/{eventId}
PUT	/events/hide/{eventId}
DELETE	/events/delete/{eventId}

Рисунок 28 – Методы Event Controller

## 6.4.3 Реализация управления профилями руководителей

Егор реализовал контроллер, управляющий данными профиля руководителей. Включает изменение пароля, email и имени, а также генерацию реферальных ссылок для приглашения новых руководителей.

user-info-controller	
POST	/users/invite-admin
POST	/users/update-password
POST	/users/update-email
GET	/users/me
PUT	/users/me
GET	/users/my-role
GET	/users/{userId}

Рисунок 29 – Методы User Info Controller

#### 6.4.4 Реализация пользовательских статусов заявок

Егор создал контроллер, который позволяет руководителям настраивать собственные статусы заявок для каждого мероприятия. Эти статусы используются на канбан-доске и могут быть связаны с автоматизациями.

Event statuses		
GET	/events/{eventId}/statuses	Получить статусы мероприятия
POST	/events/{eventId}/statuses	Добавить статус к мероприятию
PUT	/events/{eventId}/statuses/{statusId}	Обновить статус мероприятия
DELETE	/events/{eventId}/statuses/{statusId}	Удалить статус мероприятия

Рисунок 30 – Методы Event Statuses Controller

#### 6.4.5 Реализация форм мероприятий

Егор реализовал контроллер для управления анкетами, которые студенты заполняют при подаче заявки. Он обеспечивает создание, редактирование, удаление форм, а также их прикрепление к мероприятиям. Контроллер отвечает за валидацию данных и взаимодействие с фронтеном.

event-forms		
GET	/forms/standard-fields	Получить все стандартные поля для выбора
GET	/forms/system-fields	Получить все системные поля
POST	/forms	Создать новую форму или обновить уже существующую
GET	/forms	Получить список форм
GET	/forms/{eventId}	Получить форму с полями
DELETE	/forms/{eventId}	Удалить форму на мероприятие

Рисунок 31 – Методы Event Forms Controller

## 6.4.6 Реализация заявок студентов

Егор разработал контроллер, обрабатывающий студенческие заявки. Он реализовал API для получения, редактирования, удаления и отклонения заявок, а также предоставил возможность получать детальную информацию по каждой заявке.

applications		
POST	/applications Отправить новую заявку	🔒 ↴
GET	/applications Получить список заявок	🔒 ↴
GET	/applications/{id} Получить заявку по ID	📱 🔒 ↴
PUT	/applications/{id}/status Обновить статус заявки	🔒 ↴
GET	/applications/by-event/{eventId} Получить все заявки на мероприятие	🔒 ↴

Рисунок 32 – Методы Applications Controller

## 6.4.7 Реализация автоматических роботов

Егор разработал контроллер, управляющий роботами — автоматическими действиями, срабатывающими при изменении статуса заявки. Поддерживаются действия: отправка текста, отправка ссылки и отправка теста. Руководители могут настраивать роботов через интерфейс.

robots		
GET	/robots/event/{eventId} Получить список статусов и их роботов по ID события	🔒 ↴
GET	/robots/{statusId} Получить список роботов по статусу	🔒 ↴
POST	/robots/{statusId} Добавить нового робота к статусу	🔒 ↴
PUT	/robots/{statusId}/{robotId} Обновить данные робота	🔒 ↴
DELETE	/robots/{statusId}/{robotId} Удалить робота из статуса	🔒 ↴

Рисунок 33 – Методы Robots Controller

#### 6.4.8 Реализация триггеров статусов

Егор реализовал контроллер, позволяющий привязывать триггеры к определённым статусам заявки. Доступны два типа триггеров: переход по ссылке и прохождение теста. Контроллер обеспечивает гибкую настройку условий срабатывания.

The screenshot shows the API documentation for the Status Trigger Controller. It lists the following methods:

- GET /status-triggers/{statusId}** Получить список триггеров, привязанных к статусу
- POST /status-triggers/{statusId}** Привязать триггер к статусу
- GET /status-triggers/{applicationId}/{statusId}/{triggerId}/executed** Проверка выполнения триггера для заявки
- PATCH /status-triggers/{applicationId}/{statusId}/{triggerId}/executed** Установить признак выполнения триггера
- DELETE /status-triggers/{statusId}/{triggerId}** Удалить триггер из статуса
- PATCH /status-triggers/{statusId}/{triggerId}/parameters** Обновить параметры привязанного триггера

Рисунок 34 – Методы Status Trigger Controller

#### 6.4.9 Реализация управления триггерами

Егор разработал контроллер, управляющий общими триггерами в системе. Руководители могут создавать и удалять триггеры, чтобы затем использовать их в автоматизациях и связать с определёнными этапами обработки заявок.

The screenshot shows the API documentation for the Trigger Controller. It lists the following methods:

- POST /triggers** Создать новый триггер
- GET /triggers/{id}** Получить триггер по ID
- DELETE /triggers/{id}** Удалить триггер по ID

Рисунок 35 – Методы Trigger Controller

#### 6.4.10 Реализация Telegram-бота

Егор разработал Telegram-бота, который интегрирован с системой и используется для отправки сообщений студентам в рамках автоматизаций. Бот уведомляет студентов о смене этапов обработки их заявок, предоставляет ссылки, а также может направлять тесты, если это предусмотрено сценарием.

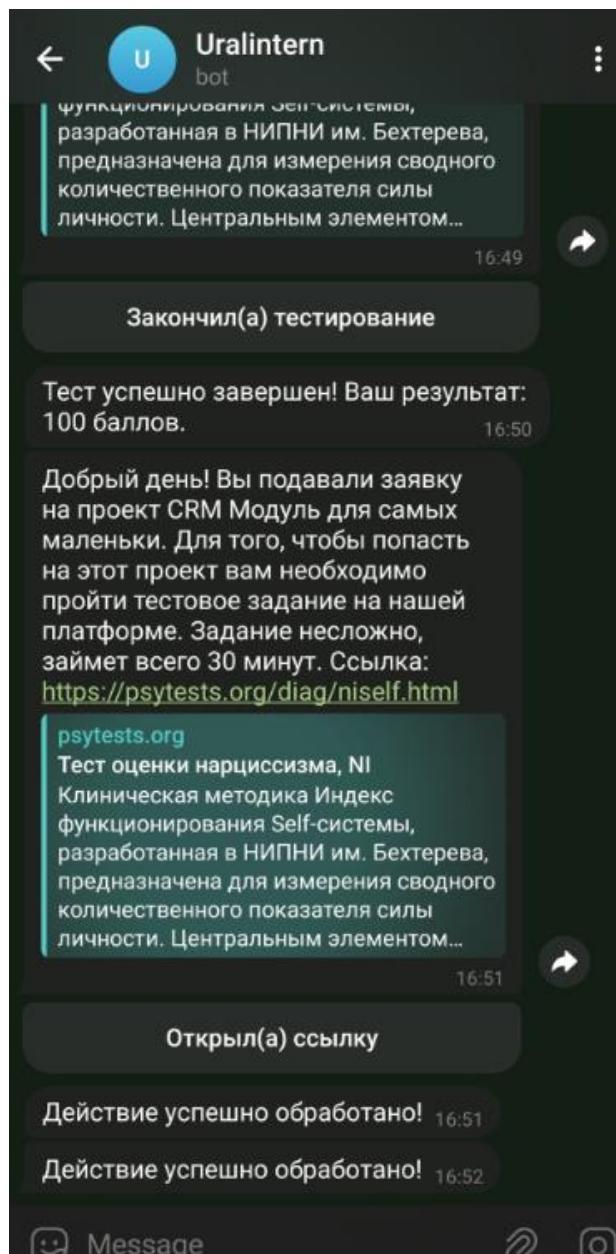


Рисунок 36 – Telegram-бот

## **ЗАКЛЮЧЕНИЕ**

Разработанный программный продукт полностью соответствует требованиям заказчика и ожиданиям конечных пользователей. Основное внимание в проекте уделялось созданию удобного интерфейса для руководителей мероприятий и простому процессу подачи заявок для студентов. Инструменты по управлению мероприятиями, настройке форм и автоматизации обработки заявок реализованы в едином интерфейсе, что обеспечило целостность и логическую связанность всех этапов бизнес-процесса. Это позволило значительно сократить временные и организационные затраты со стороны пользователей системы.

Особенностью проекта стало внедрение системы визуальной настройки этапов обработки заявок с помощью канбан-доски, гибкий конструктор форм и механизм автоматизаций через роботов и триггеры. Решение масштабируемо и адаптируемо под различные сценарии проведения мероприятий.

Для дальнейшего развития системы можно предложить:

- расширить функциональность автоматизаций — добавить больше условий, типов роботов и сценариев с таймерами;
- интегрировать аналитику и отчётность по эффективности обработки заявок;
- создать шаблоны форм и этапов для ускоренного запуска мероприятий;
- реализовать мобильную адаптацию интерфейса для руководителей.

Таким образом, проект достиг заявленных целей, продемонстрировал успешную реализацию комплексного решения и обладает потенциалом для масштабирования и дальнейшего развития в рамках образовательной деятельности.

## **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Памятка по BPMN и BPMN-диаграммам // habr.com [сайт] - 2024 - Режим доступа: <https://habr.com/ru/companies/sberbank/articles/836092/>
2. Язык UML // skillbox.ru [сайт] - 2024 - Режим доступа: <https://skillbox.ru/media/code/yazyk-uml-chto-eto-takoe-i-zachem-on-nuzhen/?ysclid=m5nyokyipd402465894>
3. Нотация BPMN // skillbox.ru [сайт] - 2024 - Режим доступа: <https://skillbox.ru/media/management/notatsiya-bpmn-chto-eto-takoe-i-kak-eye-ispolzuyut-v-biznesanalyze/?ysclid=m5nmfezaf7596598531>
4. Варианты на все случаи жизни: как написать полезный use case // practicum.yandex.ru [сайт] - 2024 - Режим доступа: <https://practicum.yandex.ru/blog/chto-takoe-use-case-kak-ih-napisat>
5. Методы формализации требований // habr.com [сайт] - 2024 - Режим доступа: <https://habr.com/ru/companies/otus/articles/825692>
6. Как писать функциональные требования // habr.com [сайт] - 2024 - Режим доступа: <https://habr.com/ru/companies/retailrocket/articles/431572>
7. Полное руководство по проектированию систем // habr.com [сайт] - 2024 - Режим доступа: <https://habr.com/ru/companies/kts/articles/741846>
8. Руководство по языку программирования Java // metanit.com [сайт] - 2024 - Режим доступа: <https://metanit.com/java/tutorial>
9. Руководство по Spring Framework // spring.io [сайт] - 2024 - Режим доступа: <https://spring.io/projects/spring-framework>
10. JavaRush // javarush.com [сайт] - 2024 - Режим доступа: <https://javarush.com>
11. Stack Overflow - Где разработчики учатся // stackoverflow.com [сайт] - 2024 - Режим доступа: <https://stackoverflow.com>
12. TanStack Query // tanstack.com [сайт] - 2024 - Режим доступа: <https://tanstack.com/query/latest/docs/framework/react/overview>
13. Дизайн React // ant.design [сайт] - 2024 - Режим доступа:

<https://ant.design/docs/react/introduce>

14. React Hook Form // react-hook-form.com [сайт] - 2024 - Режим доступа:

<https://react-hook-form.com>

15. React Router DOM: Как обрабатывать маршрутизацию в веб-приложениях // blog.logrocket.com [сайт] - 2024 - Режим доступа:  
<https://blog.logrocket.com/react-router-dom-tutorial-examples/>

## ПРИЛОЖЕНИЕ А

### План работы над проектом

Таблица 2 – План работы

Задача	Планируемый результат	Ответственный	Срок реализации
Сделать план работы на семестр	План с задачами каждого участника команды	Васильцов Владимир	01.04.2025 – 07.04.2025
Дизайн канбан-доски и исправление существующих дизайнов	Разработан дизайн канбан-доски, исправлены недочеты	Шляпникова Дарья	01.04.2025 – 07.04.2025
Верстка страницы с канбан-доской и исправление недочетов в верстке	Разработана страница с канбан-доской по дизайну, исправлена верстка уже существующих страниц	Вяткина Софья	01.04.2025 – 07.04.2025
Создание структуры БД	Визуальная структура БД(ER-diagram)	Алексеев Егор	01.04.2025 – 07.04.2025
Детализация требований к функционалу статусов.	Описаны требования к функционалу статусов.	Васильцов Владимир	01.04.2025 – 07.04.2025
Дизайн форм создания, редактирования, удаления статуса.	Готовые макеты в Figma создания, редактирования, удаления статуса.	Шляпникова Дарья	08.04.2025 – 14.04.2025
Реализовать формы для создания, редактирования и удаления статуса с валидируемыми полями по макетам.	Формы, соответствующие макетам, с валидируемыми полями для корректного ввода данных.	Вяткина Софья	08.04.2025 – 14.04.2025

Разработать API для управления статусами.	Созданы и задокументированы API-эндпоинты для создания, редактирования, удаления и получения статусов.	Алексеев Егор	08.04.2025 – 14.04.2025
Детализация требований к функционалу анкет для сбора персональных данных студентов.	Описаны требования к функционалу анкет для сбора персональных данных.	Васильцов Владимир	08.04.2025 – 14.04.2025
Разработка дизайна страницы, где будут отображаться все анкеты, а также дизайн форм для создания, редактирования и удаления анкет.	Реализованы макеты форм создания, редактирования и удаления анкет для сбора персональных данных в Figma, а также макет страницы со списком всех созданных анкет.	Шляпникова Дарья	15.04.2025 – 21.04.2025
Реализация пользовательского интерфейса для работы с анкетами (список анкет, создание, редактирование, удаление).	Разработаны страницы с анкетами, формы создания, редактирования и удаления анкет.	Вяткина Софья	15.04.2025 – 21.04.2025
Разработать API для управления анкетами.	Реализовано API для создания, редактирования, удаления и получения списка анкет.	Алексеев Егор	15.04.2025 – 21.04.2025
Детализация требований к карточкам заявок.	Описаны требования к функционалу карточек заявок.	Васильцов Владимир	15.04.2025 – 21.04.2025

Разработка дизайна карточки заявки с информацией о студенте.	Реализованы макеты карточки заявки с информацией о студенте в Figma.	Шляпникова Дарья	22.04.2025 – 28.04.2025
Реализация интерфейса карточки заявки с информацией о студенте.	Разработан пользовательский интерфейс карточки заявки. Обеспечен вывод информации о студенте в соответствии с макетом из Figma.	Вяткина Софья	22.04.2025 – 28.04.2025
Разработка API для работы с карточками заявок.	Реализовано API для предоставления данных о студентах.	Алексеев Егор	22.04.2025 – 28.04.2025
Детализация требований к функционалу роботов.	Описаны требования к функционалу роботов.	Васильцов Владимир	22.04.2025 – 28.04.2025
Дизайн канбан-доски для роботов и триггеров. Разработка дизайна форм добавления, редактирования и удаления робота у статуса.	Реализованы макеты канбан-доски для роботов и триггеров в Figma. Разработаны макеты форм для добавления, редактирования и удаления робота у статуса.	Шляпникова Дарья	29.04.2025 – 05.05.2025
Реализация интерфейса канбан-доски для управления роботами и триггерами. Разработка функционала добавления, редактирования и удаления робота у статуса.	Разработан пользовательский интерфейс канбан-доски. Реализованы формы для добавления, редактирования и удаления робота у статуса.	Вяткина Софья	29.04.2025 – 05.05.2025

Реализация API для работы с канбан-доской и функциями добавления, редактирования и удаления робота у статуса.	Реализовано API для управления роботами.	Алексеев Егор	29.04.2025 – 05.05.2025
Детализация требований к функционалу триггеров.	Описаны требования к функционалу триггеров.	Васильцов Владимир	29.04.2025 – 05.05.2025
Разработка дизайна форм добавления, редактирования и удаления триггера у статуса.	Разработаны макеты форм для добавления, редактирования и удаления триггера у статуса.	Шляпникова Дарья	06.05.2025 – 12.05.2025
Разработка функционала добавления, редактирования и удаления триггера у статуса.	Реализованы формы для добавления, редактирования и удаления триггера у статуса.	Вяткина Софья	06.05.2025 – 12.05.2025
Реализация API для работы с канбан-доской и функциями добавления, редактирования и удаления триггера у статуса.	Реализовано API для управления триггерами.	Алексеев Егор	06.05.2025 – 12.05.2025
Детализация требований к функционалу Телеграм-бота.	Описаны требования к функционалу Телеграм-бота.	Васильцов Владимир	06.05.2025 – 12.05.2025
Доработка существующих дизайнов.	Обновлены и доработаны макеты в Figma.	Шляпникова Дарья	13.05.2025 – 26.05.2025
Интеграция с бэкендом. Доработка верстки.	Обеспечена корректная работа интерфейса с бэкендом. Доработана верстка	Вяткина Софья	13.05.2025 – 26.05.2025

	в соответствии с дизайном.		
Разработка Телеграм-бота.	Реализован Телеграм-бот, который отправляет сообщения студентам по настроенным роботам в CRM.	Алексеев Егор	13.05.2025 – 26.05.2025
Подготовка к зашите.	Подготовлена презентация и отчет по работе каждого участника проекта.	Васильцов Владимир	13.05.2025 – 26.05.2025