

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 клиентской части платформы для проведения соревнований по  
машинному обучению»  
по дисциплине «Проектный практикум»

Заказчик: Фамилия И.О.

Куратор: Фамилия И.О.

Студенты команды:

Кошелев А.А.

Кошелев А.А.

Бондарев М.С.

Быкова Е.В.

Баженов Р.А.

Клименко М.В.

Попов Р.В.

Екатеринбург, 2025

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	3
1 Основная часть .....	5
1.1 Индивидуальный вклад участников .....	5
1.2 Требования заказчика.....	5
1.3 План действий .....	6
1.4 Сравнительный анализ аналогов .....	7
1.5 Архитектура программного продукта .....	7
1.5.1 Общая архитектура.....	7
1.5.2 Обоснование архитектуры бэкенда .....	9
1.5.3 Используемый стек.....	10
1.5.4 Обоснование выбора архитектуры и технологий .....	10
1.6 Планирование деятельности и распределение задач.....	10
ЗАКЛЮЧЕНИЕ .....	12

## **ВВЕДЕНИЕ**

### **1) Цель проекта:**

Создать полностью русскоязычную онлайн-платформу для проведения соревнований по машинному обучению различного уровня сложности, способную заменить существующие зарубежные решения. Платформа должна объединить обучающую и соревновательную составляющие, а также обеспечить комфортную среду для коммуникации и обмена знаниями среди участников.

### **2) Задачи проекта:**

Платформа должна предоставлять следующие ключевые возможности:

#### **а) Обучение и саморазвитие**

– Размещение курсов по машинному обучению и программированию.

#### **б) Участие в соревнованиях**

– Регистрация и участие в открытых соревнованиях с автоматизированным оцениванием;

– Лидерборды для стимуляции здоровой конкуренции.

#### **в) Организация собственных соревнований**

– Гибкие инструменты для создания и настройки соревнования: выбор метрик, формат данных, временные рамки;

– Подробные отчёты с оценками качества моделей;

– Возможность сравнивать свои решения с лучшими участниками и изучать их код.

#### **г) Коммуникация и обмен опытом**

– Форумы и чаты для обсуждения соревнований, обмена идеями и совместного решения проблем.

### **3) Актуальность проекта:**

Проект является актуальным, потому что:

– Рост спроса на IT-компетенции: ежегодно увеличивается число вакансий, требующих навыков машинного обучения и анализа данных.

Практические соревнования помогают быстрее освоить инструментарий и алгоритмы в условиях, приближённых к реальным задачам;

- Отсутствие качественных отечественных решений: на российском рынке практически нет платформ, полностью адаптированных под местные языковые и образовательные потребности. Основные аналоги - зарубежные сервисы (Kaggle, DrivenData и пр.), что создаёт языковой и культурный барьер для многих участников.

4) Платформа ориентирована на широкую аудиторию:

- Начинающие специалисты и студенты, осваивающие базовые концепции ML/RL через участие в простых соревнованиях;

- Опытные разработчики, стремящиеся проверить и укрепить свои навыки на сложных задачах;

- Преподаватели и образовательные организации, использующие площадку для проведения соревнований и студенческих конкурсов.

5) Описание ожидаемого результат:

По завершении разработки мы рассчитываем на следующий результат:

- Функциональный MVP с адаптивным дизайном, базовым функционалом клиентской части и возможностью создания и просмотра соревнований зарегистрированным пользователям.

## 1 Основная часть

### 1.1 Индивидуальный вклад участников

Таблица 1 – Индивидуальный вклад участников

Участник	Роль	Задачи
Попов Р.В.	Главный frontend-разработчик; тех-лид.	Распределение задач среди разработчиков; Принятие pull-request, рефакторинг кода; Адаптация сайта под разные разрешения экранов; Внедрение UX, связь с back-end частью.
Быкова Е.В.	Frontend-разработчик.	Адаптация сайта под разные разрешения экранов; Подготовка системы регистрации на клиентской стороне без backend.
Бондарев М.С.	Web-дизайнер; тим-лид.	Доработка макета для разработки адаптивного дизайна Поддержание связи с куратором проекта Составление презентаций и отчётности
Клименко М.В.	Frontend-разработчик.	Адаптация сайта под разные разрешения экранов Разработка некоторых страниц сайта.
Баженов Р.А.	Frontend-разработчик.	Адаптация сайта под разные разрешения экранов; Разработка некоторых страниц сайта.

### 1.2 Требования заказчика

– Адаптивный дизайн для различных разрешений экрана;

- Минималистичный и продуманный UI/UX для основных сценариев использования;

- Интеграция с существующим Backend через REST API.

### **1.3 План действий**

1) Анализ и сбор информации:

- Выбор одного проекта из прошлогодний, который будет дорабатываться/перерабатываться;

- Ознакомление и анализ Референсов (Kaggle.com);

- Определение стека технологий.

2) Дизайн – Доработка и адаптация макета под различные разрешения экранов

3) Разработка MVP без интеграции с backend:

а) Рефакторинг, внедрение адаптива, анимаций и базового функционала клиентской части в следующих частях сайта:

- Страница “Обучение”;

- Шапка сайта;

- Страницы “Авторизация/Регистрация”;

- Страница “Соревнования”;

- Страница “Соревнование”.

б) Подготовка и парсинг тестовых данных для последующей замены на backend данные

в) Разработка системы авторизации/деавторизации без backend

г) Разработка следующих страниц:

- “FAQ”;

- “Контакты”.

4) Интеграция MVP с backend:

- Получить, настроить и локально развернуть полученный backend;

- Создать соревнования на сервере;

- Интегрировать Соревнования на сервере со страницами “Соревнования”, “Главная”, “Соревнование”;
- Настроить фильтрацию соревнований, реализовать разбиение на категории;
- Реализации авторизации/Регистрации через сервер;
- Разработать страницу “Создать соревнование”, связать её с backend и остальными частями сайта.

## 1.4 Сравнительный анализ аналогов

Таблица 2 – Анализ аналогов

Аналог	Плюсы	Минусы
Kaggle	Широкая Аудитория; множество готовых датасетов; интеграция с ноутбуками.	Англоязычный интерфейс; жёсткая конкуренция; нет частных корпоративных турниров.
DriverData	Социальная направленность проектов; поддержка НПО.	Узкая тематика; ограниченное кол-во новых соревнований.
OpenML	Открытая экосистема; Стандартизированные API;	Нет развитого интерфейса соревнований.

## 1.5 Архитектура программного продукта

### 1.5.1 Общая архитектура

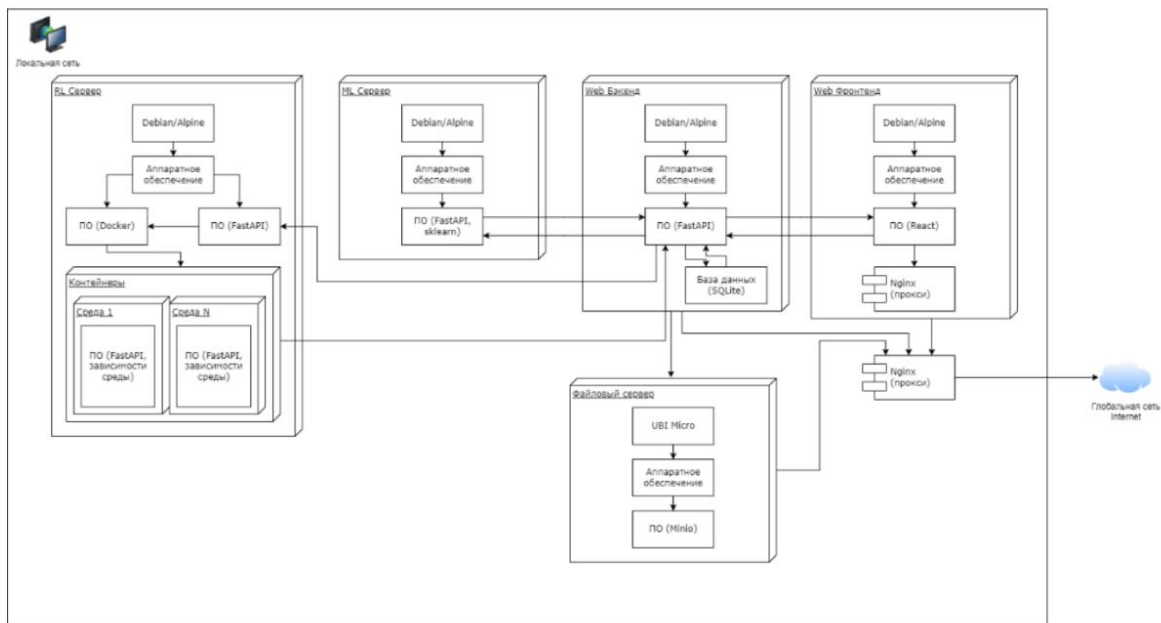


Рисунок 1 – Общая схема

Система развёрнута в пределах локальной сети, с выходом во внешнюю сеть через прокси-сервер Nginx. Архитектура построена по микросервисному принципу и включает несколько логических узлов:

1) RL Сервер (усиленное обучение)

а) ОС:

– Debian/Alpine.

б) Компоненты:

– ПО Docker - изоляция контейнеров для запуска обучающих сред;

– ПО FastAPI - реализация REST-интерфейса управления и взаимодействия;

– Контейнеры: каждый содержит собственную среду (Среда 1 ... Среда N) с FastAPI и необходимыми зависимостями.

Связан с Web-бэкендом для получения заданий и отправки результатов.

2) ML Сервер (машинное обучение)

а) ОС:

– Debian/Alpine.

б) Компоненты:



– FastAPI + scikit-learn - обработка ML-задач, обучение моделей, предсказания.

Обмен данными происходит с Web-бэкендом.

### 3) Web Бэкенд

#### а) ОС:

– Debian/Alpine.

#### б) Компоненты:

– FastAPI - основной управляющий API-сервер;

– SQLite - встроенная база данных для хранения метаданных и пользовательской информации.

Является центральным координирующим узлом между ML-сервером, RL-сервером, файловым хранилищем и фронтендом.

### 4) Web Фронтенд

#### а) ОС:

– Debian/Alpine.

#### б) Компоненты:

– React - клиентская часть интерфейса;

– Nginx (прокси) - проксирование запросов от внешнего мира и к API.

Запросы пользователей из Интернета направляются через Nginx и передаются к Web-бэкенду или обслуживаются напрямую.

### 5) Файловый сервер

#### а) ОС:

– UBI Micro.

#### б) Компоненты:

– ПО MinIO — объектное хранилище, аналог S3.

Служит для хранения данных: датасеты, модели, результаты, загрузки.

## **1.5.2 Обоснование архитектуры бэкенда**

Наша команда не разрабатывала бэкенд с нуля - он был предоставлен в готовом виде. В связи с этим, архитектурные и технологические решения по использованию FastAPI, Docker, SQLite, MinIO и другим компонентам были

приняты сторонними разработчиками. Мы лишь адаптировали и интегрировали данный бэкенд в общий стек проекта.

### **1.5.3 Используемый стек**

- JavaScript + TypeScript;
- React - основа SPA-интерфейса;
- Vite - сборщик проекта, альтернатива Webpack;
- Tailwind CSS - утилитарный CSS-фреймворк для стилизации;
- AntDesign - компонентная библиотека UI-элементов.

### **1.5.4 Обоснование выбора архитектуры и технологий**

React выбран в качестве основы благодаря высокой популярности, развитому экосистемному окружению и возможности создания масштабируемых SPA-приложений. Vite используется как современная альтернатива Webpack, обеспечивающая быструю сборку, «горячую» перезагрузку и оптимизированную разработку.

Tailwind CSS выбран для ускорения процесса вёрстки и упрощения стилизации за счёт утилитарного подхода. Ant Design применён для обеспечения визуального единообразия интерфейса и ускоренной реализации стандартных компонентов.

TypeScript используется частично для повышения надёжности и читаемости кода, однако из-за ограниченного времени переход с JavaScript выполнен не полностью.

Тем не менее, проект по-прежнему содержит фрагменты на JavaScript, так как переход не был завершён полностью.

## **1.6 Планирование деятельности и распределение задач**

Работа над проектом велась в командном формате с использованием гибкого подхода к планированию. Для координации действий участники команды регулярно проводили созвоны, на которых обсуждался текущий прогресс, возникшие проблемы и планируемые этапы разработки.

Существенное влияние на ход разработки оказало отсутствие готового backend на ранних стадиях проекта. Бэкенд находился в процессе доработки, что ограничивало фронтенд-разработку и требовало пересмотра первоначальных сроков и приоритетов.

Распределение задач осуществлялось тех-лидом команды с использованием доски в формате Kanban. Такой подход позволял гибко управлять загрузкой участников, оперативно адаптироваться к изменениям и сохранять прозрачность в контроле над текущими задачами.

## ЗАКЛЮЧЕНИЕ

В ходе выполнения проекта была проведена комплексная работа по реализации клиентской части платформы для проведения соревнований по машинному обучению. На основе требований заказчика была реализована адаптивная верстка, корректно функционирующая на устройствах с различными разрешениями экрана, что обеспечивает доступность ресурса для широкой аудитории пользователей.

Особое внимание было уделено разработке минималистичного и интуитивно понятного интерфейса. Были проработаны основные пользовательские сценарии, включая просмотр списка соревнований и детальной информации о каждом из них. Это позволило сформировать логичную и последовательную структуру взаимодействия пользователя с системой.

Интеграция с серверной частью осуществлялась через REST API. На текущем этапе реализована возможность получения данных о соревнованиях с backend-сервера, что позволяет пользователю просматривать актуальную информацию в динамическом режиме. Однако ряд функций, предусмотренных в задачах проекта (создание новых соревнований, регистрация пользователей, отправка решений), пока не реализован ввиду ограниченного доступа к полнофункциональному backend на момент основной разработки. Несмотря на это, фундамент для дальнейшего расширения клиентской логики уже заложен.

Результаты предварительного тестирования показали стабильную работу интерфейса, корректное отображение информации и соответствие требованиям адаптивности. Основные ошибки и недочёты носили технический и интеграционный характер, не оказывая критического влияния на пользовательский опыт.

С учётом проведённой работы можно заключить, что разработанная клиентская часть соответствует ключевым требованиям заказчика и

представляет собой надёжную основу для последующего функционального расширения. В качестве перспектив дальнейшего развития проекта можно выделить:

- завершение полной интеграции с backend (авторизация, отправка решений, создание соревнований);
- реализацию пользовательских профилей и систем уведомлений;
- проведение нагрузочного тестирования и оптимизацию производительности.

Данный проект продемонстрировал важность тесного взаимодействия между командами frontend и backend, а также необходимости раннего доступа к конечным интерфейсам взаимодействия для более эффективной разработки. Полученный опыт может быть полезен для планирования и реализации аналогичных проектов в будущем.