

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Образовательная программа «Программная инженерия»

СОГЛАСОВАНО

Старший преподаватель департамента
больших данных и информационного
поиска

Куренков В.В. Куренков
« 13 » _____ мая _____ 2023 г.

УТВЕРЖДАЮ

Академический руководитель
образовательной программы
«Программная инженерия»
профессор департамента программной
инженерии, канд. техн. наук

Шилов В.В. Шилов
« 13 » _____ мая _____ 2023 г.

**Система проверки решений по программированию для Лицея НИУ ВШЭ
с автоматизированным подсчётом рейтинга**

Техническое задание

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.05.10-01 ТЗ 01-1-ЛУ

Исполнитель

студент группы БПИ219

Котовский / С. О. Котовский /
« 13 » _____ мая _____ 2023 г.

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл.	

Москва 2023

УТВЕРЖДЕН
RU.17701729.05.10-01 ТЗ 01-1-ЛУ

**Система проверки решений по программированию для Лицея НИУ ВШЭ
с автоматизированным подсчётом рейтинга**

Техническое задание

RU.17701729.05.10-01 ТЗ 01-1

Листов 32

<i>Подп. и дата</i>	
<i>Инв. № дубл.</i>	
<i>Взам. инв. №</i>	
<i>Подп. и дата</i>	
<i>Инв. № подл</i>	

Москва 2023

ОГЛАВЛЕНИЕ

1.	ВВЕДЕНИЕ	4
1.1.	Наименование программы и объекта, в котором программа используется	4
1.2.	Краткая характеристика области применения программы и объекта, в котором она используется	4
2.	ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ	5
3.	НАЗНАЧЕНИЕ РАЗРАБОТКИ	6
3.1.	Функциональное назначение	6
3.2.	Эксплуатационное назначение	6
4.	ТРЕБОВАНИЯ К ПРОГРАММЕ	7
4.1.	Требования к функциональным характеристикам	7
4.1.1.	Требования к клиентской части	7
4.1.2.	Требования к взаимодействию клиентской и серверной частей	20
4.1.3	Требования к серверу для парсинга Ejudge.	23
4.1.4.	Требования к входным данным	24
4.1.5.	Требования к выходным данным	24
4.2.	Требования к надёжности	25
4.2.4.	Обеспечение устойчивого функционирования	25
4.2.5.	Контроль входной информации	25
4.2.6.	Контроль выходной информации	25
4.2.7.	Время восстановления после отказа	25
4.3.	Условия эксплуатации	25
4.3.4.	Климатические условия эксплуатации	25
4.3.5.	Требования к видам обслуживания	25
4.3.6.	Требования к численности и квалификации персонала	25
4.4.	Требования к составу и параметрам технических средств	25
4.4.1.	Минимальные требования к сетевому оборудованию	25

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

4.5.	Требования к информационной и программной совместимости	26
4.5.4.	Требования к исходным кодам и языкам программирования.....	26
4.5.5.	Требования к программным средствам, используемым программой	26
4.6.	Требования к маркировке и упаковке	26
4.7.	Требования к транспортированию и хранению	26
5.	ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ	27
5.1.	Состав программной документации.....	27
5.2.	Специальные требования к программной документации	27
6.	ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ	28
6.1.	Предполагаемая потребность.....	28
6.2.	Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами	28
7.	СТАДИИ И ЭТАПЫ РАЗРАБОТКИ.....	30
8.	ПОРЯДОК КОНТРОЛЯ И ПРИЁМКИ.....	31
8.1.	Виды испытаний.....	31
8.2.	Общие требования к приемке работы	31
	ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ.....	32

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

1. ВВЕДЕНИЕ

1.1. Наименование программы и объекта, в котором программа используется

Наименование темы разработки: «Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга».

Наименование темы разработки на английском языке: «HSE Lyceum Programming Testing System with Automated Rating Estimation».

Наименование объекта, в котором программа используется: Лицей НИУ ВШЭ.

1.2. Краткая характеристика области применения программы и объекта, в котором она используется

«Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга» - набор программных инструментов для оценивания решений задач на разных языках программирования для учащихся Лицея НИУ ВШЭ с рейтингом для каждого учащегося.

Лицей НИУ ВШЭ – подразделение НИУ ВШЭ.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

2. ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ

Основанием для разработки является учебный план подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и утвержденная академическим руководителем тема курсового проекта.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

3. НАЗНАЧЕНИЕ РАЗРАБОТКИ

3.1. Функциональное назначение

Программа содержит в себе систему, тестирующую решения задач по программированию Лицея НИУ ВШЭ и автоматически подсчитывающую рейтинг каждого учащегося, исходя из его оценок и количества решенных задач. Имеет учебно-методическое назначение.

3.2. Эксплуатационное назначение

Программа направлена на решение учащимися Лицея НИУ ВШЭ задач по программированию, которые им выдал тот или иной преподаватель, просмотр учащимися своего рейтинга по оценкам и количеству решенных задач среди других учащихся. Также программа направлена на просмотр преподавателями решений задач учащимися, получения их рейтинга в виде файлов.

Конечные пользователи - ученики и преподаватели информатики/программирования Лицея НИУ ВШЭ.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

4. ТРЕБОВАНИЯ К ПРОГРАММЕ

4.1. Требования к функциональным характеристикам

4.1.1. Требования к клиентской части

Процесс регистрации:

Регистрация нового студента происходит через сущность преподавателя. (см. Преподаватель - создание учётной записи студента)

Регистрация нового преподавателя проводится администратором системы в ручном режиме.

Процесс авторизации:

1. Авторизация пользователя

1.1. Система должна предоставлять форму для ввода адреса электронной почты и пароля пользователя.

1.2. Форма должна содержать два текстовых поля для ввода e-mail и пароля.

1.3. Система должна проверять введенные данные на корректность.

1.4. Система должна отображать сообщение об ошибке при вводе некорректных данных.

2. Отправка данных на сервер

2.1. После ввода данных, система должна отправлять запрос на сервер для проверки введенных данных.

2.2. Запрос должен содержать введенный адрес электронной почты и пароль пользователя.

2.3. Система должна обрабатывать ответ от сервера и предпринимать соответствующие действия в зависимости от ответа.

3. Обработка ответа сервера

3.1. Если сервер подтверждает корректность введенных данных, система должна сохранять токен авторизации, возвращенный сервером, в локальном хранилище браузера.

3.2. Если сервер сообщает о некорректности введенных данных, система должна отображать сообщение об ошибке.

4. Перенаправление пользователя

4.1. После успешной авторизации система должна перенаправлять пользователя на соответствующую страницу в зависимости от роли пользователя (ученик или учитель).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

4.2. Система должна обеспечивать автоматическое перенаправление на страницу входа при попытке доступа к защищенным ресурсам без авторизации.

5. Выход из системы

5.1. Система должна предоставлять функцию для выхода из системы.

5.2. При выходе из системы, система должна удалять токен авторизации из локального хранилища браузера и перенаправлять пользователя на страницу входа.

Элементы шапки сайта (header):

1. Вывод информации о пользователе

1.1. Система должна отображать имя пользователя в правой части шапки сайта.

1.2. Система должна получать информацию о пользователе (включая имя, фамилию и роль) с сервера.

1.3. Система должна проверять статус авторизации пользователя и в случае отсутствия авторизации, перенаправлять пользователя на страницу входа.

2. Навигация по сайту

2.1. Шапка сайта должна содержать основные ссылки для навигации по сайту.

2.2. Доступные ссылки должны включать: "Мои посылки", "Мои контесты", "Рейтинг".

2.3. Пункты навигации должны быть адаптированы в соответствии с ролью пользователя. Для учителей должна быть доступна ссылка на "Админ-панель", для студентов - ссылка на "Личный кабинет".

3. Выход из системы

3.1. Шапка сайта должна предоставлять функцию для выхода из системы.

3.2. При нажатии на кнопку "Выйти", система должна удалять токен авторизации из локального хранилища браузера и перенаправлять пользователя на страницу входа.

4. Стилизация шапки сайта

4.1. Все изменения в стилизации должны быть реализованы при помощи CSS, без использования дополнительных библиотек или фреймворков.

5. Логотип сайта

5.1. Шапка сайта должна содержать логотип сайта и его название.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

5.2. Логотип и название должны быть расположены слева в шапке сайта.

Элементы подвала сайта (footer)

1. Описание компонента:
 - Компонент "SiteFooter" представляет собой футер сайта.
 - Компонент содержит информацию о копирайте и авторских правах.
 - Компонент не имеет интерактивных элементов.
2. Разметка и стилизация:
 - В разметке компонента используется элемент `<footer>` для обозначения футера сайта.
 - Футер содержит контейнер `<div>` с классом "footer-container", внутри которого располагается текст с информацией о копирайте.
 - Стилизация футера задана в блоке `<style>`. Контейнер футера имеет высоту 70 пикселей, фоновый цвет #454545, и находится в постоянном положении (static).
 - Контейнер футера использует flexbox для выравнивания содержимого по центру.
3. Компонент не содержит вычисляемых свойств (computed).
4. События:
 - Компонент не генерирует собственных событий.
5. Входные параметры (props):
 - Компонент не принимает входных параметров.
6. Методы:
 - Компонент не содержит собственных методов.
7. Дополнительная информация:
 - Компонент является статическим и не требует активного взаимодействия пользователя.
 - Компонент может быть размещен внизу страницы для отображения футера сайта с информацией о копирайте.

Таким образом, компонент "SiteFooter" представляет собой простой футер сайта с информацией о копирайте, который не взаимодействует с пользователем и не требует входных параметров или методов для своей работы.

Страница профиля студента

1. Вывод информации о студенте

1.1. Страница профиля студента должна отображать основную информацию о студенте: имя, фамилию, направление, группу, ID, электронную почту, класс, программу.

1.2. Данные должны быть получены с сервера.

1.3. В случае возникновения ошибки при получении данных, ошибка должна быть залогирована.

2. Вывод рейтинга студента

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

2.1. Страница профиля студента должна отображать рейтинг студента по оценкам и рейтинг студента по задачам. (номер места в рейтинге)

3. Навигация

3.1. Страница должна содержать шапку сайта и футер, которые являются отдельными компонентами Vue.js.

4. Стилизация страницы профиля студента

4.1. Все изменения в стилизации должны быть реализованы при помощи CSS, без использования дополнительных библиотек или фреймворков.

Админ-панель преподавателя

1. Вывод информации о преподавателе

1.1. Админ-панель преподавателя должна отображать основную информацию о преподавателе: имя, отчество, фамилию, электронную почту, кафедру, группу.

1.2. Данные должны быть получены с сервера.

1.3. В случае возникновения ошибки при получении данных, ошибка должна быть залогирована.

2. Функционал админ-панели преподавателя

2.1. Админ-панель преподавателя должна содержать функционал для создания аккаунта студента и импорта контеста из Ejudge.

2.2. Данный функционал должен быть представлен в виде кнопок, которые ведут на соответствующие страницы.

3. Навигация

3.1. Админ-панель должна содержать шапку сайта и футер, которые являются отдельными компонентами Vue.js.

4. Стилизация админ-панели преподавателя

4.1. Все изменения в стилизации должны быть реализованы при помощи CSS, без использования дополнительных библиотек или фреймворков.

5. Роутинг

5.1. Страницы "Создать аккаунт студента" и "Импорт контеста из Ejudge" должны быть доступны по соответствующим ссылкам в админ-панели преподавателя.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

Страница импорта контеста из ejudge

1. Ввод данных контеста

1.1. Страница должна содержать форму для ввода данных контеста: ID контеста, время начала, время окончания и название контеста.

1.2. Все поля формы являются обязательными для заполнения.

2. Функционал импорта контеста

2.1. При отправке формы должен производиться запрос для импорта контеста из Ejudge.

2.2. Во время выполнения запроса должен отображаться индикатор обработки.

2.3. После успешного выполнения запроса должно отображаться сообщение о успешном добавлении контеста.

2.4. В случае возникновения ошибки при выполнении запроса, ошибка должна быть залогирована.

3. Навигация

3.1. Страница импорта контеста из Ejudge должна содержать шапку сайта и футер, которые являются отдельными компонентами Vue.js.

4. Стилизация страницы импорта контеста

4.1. Все изменения в стилизации должны быть реализованы при помощи CSS, без использования дополнительных библиотек или фреймворков.

5. Обработка и отправка данных на сервер

5.1. Данные из формы должны быть обработаны и преобразованы в соответствующий формат. (описано в техническом задании для backend)

5.2. Отправка данных на сервер должна производиться при помощи Axios, с использованием токена авторизации, хранящегося в LocalStorage.

5.3. В случае успешного добавления контеста, должно отображаться соответствующее сообщение.

Страница создания аккаунта студента

1. Ввод данных студента

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

1.1. Страница должна содержать форму для ввода данных студента: имя, фамилия, отчество, почта, роль, пароль, направление и группа.

1.2. Все поля формы являются обязательными для заполнения.

2. Функционал создания аккаунта студента

2.1. При отправке формы должен производиться запрос на сервер для создания аккаунта студента.

2.2. В случае успешного выполнения запроса, должно отображаться соответствующее сообщение (необходимо добавить на страницу).

2.3. В случае возникновения ошибки при выполнении запроса, ошибка должна быть залогирована.

3. Навигация

3.1. Страница создания аккаунта студента должна содержать шапку сайта и футер, которые являются отдельными компонентами Vue.js.

4. Стилизация страницы создания аккаунта студента

4.1. Все изменения в стилизации должны быть реализованы при помощи CSS, без использования дополнительных библиотек или фреймворков.

5. Обработка и отправка данных на сервер

5.1. Данные из формы должны быть обработаны и преобразованы в соответствующий формат.

5.2. Отправка данных на сервер должна производиться при помощи Axios, с использованием заголовка 'Content-Type': 'application/json'.

5.3. В случае успешного создания аккаунта студента, должно отображаться соответствующее сообщение (необходимо добавить на страницу).

Страница с рейтингом пользователей

1. Выбор типа рейтинга

1.1. Страница должна содержать кнопки для выбора типа рейтинга: "Рейтинг по оценкам" и "Рейтинг по задачам".

1.2. По умолчанию, при загрузке страницы, должен быть активным рейтинг по оценкам.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

1.3. При выборе типа рейтинга, должна происходить обновление данных рейтинга и отображение соответствующей информации.

2. Получение данных рейтинга

2.1. При загрузке страницы и при смене типа рейтинга, должен производиться запрос на сервер для получения данных рейтинга.

2.2. Запрос должен быть выполнен с использованием токена, полученного при аутентификации.

2.3. Запрос должен отправляться на соответствующий эндпоинт в зависимости от выбранного типа рейтинга: "<http://37.252.0.155:8080/api/grade/by-marks>" для рейтинга по оценкам и "<http://37.252.0.155:8080/api/grade/by-tasks-amount>" для рейтинга по задачам.

2.4. При успешном получении данных, должны быть обновлены значения переменной `ratingData` и отображена таблица с данными рейтинга.

2.5. В случае возникновения ошибки при получении данных, ошибка должна быть залогирована, а пользователю должно быть отображено соответствующее сообщение об ошибке.

3. Выгрузка данных рейтинга в CSV

3.1. На странице должна быть кнопка "Выгрузить в .csv", которая позволяет пользователю скачать данные рейтинга в формате CSV.

3.2. При клике на кнопку "Выгрузить в .csv", должен быть выполнен запрос на сервер для получения CSV файла.

3.3. Запрос должен быть выполнен с использованием токена, полученного при аутентификации.

3.4. Запрос должен отправляться на соответствующий эндпоинт в зависимости от выбранного типа рейтинга: "<http://37.252.0.155:8080/api/grade/by-marks/download-csv>" для рейтинга по оценкам и "<http://37.252.0.155:8080/api/grade/by-tasks/download-csv>" для рейтинга по задачам.

3.5. При успешном получении CSV файла, файл должен быть скачан пользователем.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

3.6. В случае возникновения ошибки при получении CSV файла, ошибка должна быть залогирована, а пользователю должно быть отображено соответствующее сообщение об ошибке.

4. Отображение данных рейтинга

4.1. После успешного получения данных рейтинга, таблица должна быть отображена на странице.

4.2. Таблица должна содержать следующие столбцы: "Имя", "Фамилия", "Отчество", "Email", "Роль", "Факультет" и "Группа".

4.3. Каждая строка таблицы должна представлять отдельного пользователя и содержать соответствующую информацию из полученных данных рейтинга.

5. Отображение ошибок

5.1. В случае возникновения ошибки при получении данных рейтинга или при скачивании CSV файла, ошибка должна быть залогирована, а пользователю должно быть отображено соответствующее сообщение об ошибке.

5.2. Сообщение об ошибке должно быть отображено под кнопками выбора типа рейтинга или над таблицей данных рейтинга.

5.3. Сообщение об ошибке должно быть выделено цветом и/или шрифтом, чтобы привлечь внимание пользователя.

5.4. После успешной загрузки данных или успешного скачивания CSV файла, сообщение об ошибке должно быть очищено или скрыто.

6. Оформление интерфейса

6.1. Страница должна иметь адаптивный дизайн и корректно отображаться на различных устройствах и экранах.

6.2. Кнопки выбора типа рейтинга должны иметь соответствующий стиль и быть различимыми.

6.3. Таблица данных рейтинга должна быть оформлена с использованием стилей, чтобы обеспечить читаемость и удобство просмотра.

6.4. Сообщения об ошибке должны быть оформлены в соответствии с общим стилем страницы и хорошо видны для пользователя.

6.5. Другие элементы интерфейса, такие как заголовки, тексты и кнопки, должны быть четко видны и читаемыми.

6.6. Страница должна иметь приятный и интуитивно понятный пользовательский интерфейс для удобства использования.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

6.7. Верхний и нижний колонтитулы, содержащие компоненты `SiteHeader` и `SiteFooter`, должны быть отображены на странице для обеспечения единого внешнего вида и навигации по сайту.

7. Взаимодействие с сервером

7.1. Все запросы к серверу должны быть выполнены с

использованием библиотеки `axios` для осуществления HTTP-запросов.

7.2. Для каждого запроса к серверу должен быть указан соответствующий URL-адрес эндпоинта и метод HTTP (например, GET, POST).

7.3. Запросы должны содержать необходимые заголовки, включая токен авторизации в заголовке "Authorization" для аутентификации пользователя.

7.4. В случае успешного выполнения запроса, полученные данные должны быть обработаны и отображены на странице в соответствии с требованиями.

7.5. В случае ошибки при выполнении запроса, ошибка должна быть обработана и отображена пользователю в виде соответствующего сообщения.

7.6. Для обработки ошибок и логирования необходимо использовать соответствующие механизмы предоставляемые библиотекой `axios` и языком программирования JavaScript.

8. Общие требования

8.1. Код страницы должен быть написан в соответствии с принципами чистого и структурированного кода, обеспечивая читаемость, модульность и масштабируемость.

8.2. Компоненты `SiteHeader` и `SiteFooter` должны быть включены и правильно использованы на странице для обеспечения единого внешнего вида и навигации по сайту.

8.3. Все зависимости, необходимые для работы страницы, должны быть указаны в файле `package.json` и должны быть установлены с использованием менеджера пакетов (например, `npm` или `yarn`).

8.4. Страница должна быть протестирована на различных устройствах и браузерах, чтобы обеспечить ее корректное отображение и функциональность.

8.5. Все внешние ресурсы, такие как изображения или сторонние библиотеки, должны быть правильно подключены и загружены на странице.

8.6. Все данные, полученные от сервера или введенные пользователем, должны быть проверены и обработаны с учетом возможных ошибок и безопасности.

8.7. Страница должна быть доступна для использования пользователями с различными специальными потребностями, следуя принципам доступности веб-контента.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

8.8. Документация, описывающая функциональность страницы, ее зависимости и взаимодействие с сервером, должна быть предоставлена для разработчиков и сопровождающих систему.

Страница с ссылками конкретного пользователя

Описание: Страница ParcelsList предназначена для отображения списка посылок. На странице отображается информация о каждой посылке, такая как ID, язык, статус, время работы, использованная память, тест и другие данные. Пользователь может просмотреть код посылки, нажав на кнопку "Просмотреть код".

Структура файла:

- Импорт необходимых компонентов: SiteFooter, SiteHeader, axios.
- Определение компонента Vue с наименованием "ParcelsList".
- Определение следующих data-свойств:
 - parcels: массив для хранения списка посылок;
 - modalOpen: логическое значение для открытия/закрытия модального окна с кодом посылки;
 - selectedCode: строка для хранения выбранного кода посылки.
- Определение вычисляемого свойства:
 - mainHeadline: строка, содержащая основной заголовок страницы в зависимости от роли пользователя.
- Определение методов:
 - fetchParcels: метод для получения списка посылок с сервера;
 - openCodeModal: метод для открытия модального окна с кодом посылки;
 - closeModal: метод для закрытия модального окна.
- В хуке created вызывается метод fetchParcels для получения списка посылок при загрузке страницы.

Внешний вид страницы: Страница состоит из следующих элементов:

- Компонент SiteHeader для отображения шапки сайта.
- Компонент SiteFooter для отображения футера сайта.
- Заголовок страницы в зависимости от роли пользователя.
- Информация о количестве посылок.
- Список посылок в виде таблицы со следующими столбцами:
 - ID: идентификатор посылки.
 - Просмотреть код: кнопка, при нажатии на которую открывается модальное окно с кодом посылки.
 - Язык: язык программирования посылки.
 - Статус: текущий статус посылки.
 - Время работы: время выполнения посылки.
 - Использованная память: объем памяти, использованный посылкой.
 - Тест: результат тестирования посылки.
 - Название конкурса: название конкурса, в рамках которого выполнена посылка.
 - Название задачи: название задачи, для которой выполнена посылка.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

Модальное окно с кодом посылки, открываемое при нажатии на кнопку "Просмотреть код". Модальное окно содержит заголовок "Код посылки" и отображает выбранный код посылки в виде предварительно отформатированного текста. Модальное окно также содержит кнопку "Закрыть", при нажатии на которую модальное окно закрывается.

Стилизация страницы:

- Компонент `modal-backdrop`: фиксированное позиционирование, затемненный фон с прозрачностью, выравнивание по центру экрана.
- Компонент `modal-content`: фоновый цвет, скругленные углы, отступы и размеры, позволяющие контенту поместиться внутри окна, возможность прокрутки при содержании большого количества данных.

Технические требования:

1. Запросы на сервер:
 - Метод `fetchParcels` использует библиотеку `axios` для отправки GET-запроса на эндпоинт `http://37.252.0.155:8080/api/solutions/${endpoint}` для получения списка посылок.
 - Заголовок запроса содержит ключ авторизации в формате `Authorization: Bearer <token>`, где `<token>` - значение токена, полученного при аутентификации.
 - В случае успешного выполнения запроса, полученные данные присваиваются свойству `parcels` компонента.
 - В случае ошибки при выполнении запроса, ошибка должна быть залогирована в консоль.
2. Модальное окно:
 - Компонент `modalOpen` управляет отображением модального окна. При щелчке на кнопку "Просмотреть код" свойство `modalOpen` устанавливается в значение `true`, что приводит к отображению модального окна.
 - Компонент `selectedCode` хранит выбранный код посылки, который отображается в модальном окне.
 - При нажатии на кнопку "Закрыть" свойство `modalOpen` устанавливается в значение `false`, что приводит к закрытию модального окна.
3. Обработка ошибок:
 - В случае возникновения ошибки при выполнении запроса на получение списка посылок, ошибка должна быть залогирована в консоль с помощью `console.error()`.
4. Навигация:
 - Страница должна содержать компоненты `SiteHeader` и `SiteFooter` для отображения шапки и футера сайта соответственно.
5. Стилизация:
 - Стилизация страницы должна быть реализована с использованием CSS.
 - Для модального окна следует задать стили для компонентов `modal-backdrop` и `modal-content`, обеспечивающие желаемый внешний вид и расположение.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

Страница со списком всех контестов

Страница `MyContestsList` предназначена для отображения списка контестов и их основных свойств. Ниже представлено техническое задание для данной страницы:

1. Загрузка списка контестов:
 - При создании компонента `MyContestsList` должен быть выполнен асинхронный метод `fetchContests`, который выполняет GET-запрос к API для получения списка контестов.
 - Запрос должен использовать токен авторизации, полученный из локального хранилища.
 - Адрес эндпоинта для получения списка контестов:
`http://37.252.0.155:8080/api/contest/get-all`.
 - В случае успешного выполнения запроса, полученные данные должны быть присвоены свойству `contests` компонента.
 - При возникновении ошибки во время выполнения запроса, ошибка должна быть залогирована в консоль.
2. Отображение списка контестов:
 - В шаблоне компонента используется цикл `v-for` для отображения каждого контеста в таблице.
 - Поля контеста, такие как ID в `Ejudge`, название контеста, дата начала и дата окончания, должны быть отображены в соответствующих столбцах таблицы.
 - Для преобразования даты начала и даты окончания контеста в удобный формат, используется метод `toLocaleString()` с объектом `Date`.
 - Для перехода к отдельному контесту используется компонент `router-link`, который принимает параметры маршрута `name` и `params`.
3. Сохранение выбранного контеста:
 - При щелчке на кнопку "Перейти" в таблице, вызывается метод `saveContestId`, который сохраняет ID контеста в `Ejudge` и его название в локальном хранилище.
 - Значения сохраняются в виде пар ключ-значение в `localStorage` с ключами `"contestId"` и `"contestName"` соответственно.
4. Навигация:
 - Страница должна содержать компоненты `SiteHeader` и `SiteFooter` для отображения шапки и футера сайта соответственно.
5. Стилизация:
 - Кнопка "Перейти" имеет стилизацию, заданную в блоке стилей компонента.
 - При наведении, активации и фокусе кнопки должны применяться соответствующие стилевые изменения для создания визуального отклика.

Страница с конкретным контестом

Страница `Contest` предназначена для отображения информации о контесте, задачах, отправке решений и результатов. Ниже представлено техническое задание для данной страницы:

1. Загрузка задач контеста:

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

- При создании компонента Contest должен быть выполнен асинхронный метод `fetchTasks`, который выполняет GET-запрос к API для получения списка задач конкурса.
 - Адрес эндпоинта для получения списка задач:
`http://37.252.0.155:3000/parseTasks`.
 - В случае успешного выполнения запроса, полученные данные должны быть присвоены свойству `tasks` компонента.
 - Выбирается первая задача из списка и вызывается метод `updateTask` для отображения информации о задаче.
2. Аутентификация:
- При загрузке страницы, должен быть выполнен метод `authenticate`, который отправляет POST-запрос на сервер для аутентификации.
 - Аутентификационный запрос выполняется на адрес
`http://37.252.0.155:3000/auth`.
 - В теле запроса передаются параметры `login`, `password` и `contestID`, полученные из локального хранилища.
 - В случае успешной аутентификации, полученные данные сохраняются для дальнейшего использования.
 - При возникновении ошибки во время аутентификации, ошибка должна быть залогирована в консоль.
3. Обновление текущей задачи:
- Метод `updateTask` принимает номер задачи в качестве аргумента.
 - Используя номер задачи, находит соответствующую задачу в списке `tasks` и обновляет свойства `currentTask`, `taskTitle` и `taskDescription` с информацией о выбранной задаче.
 - Если задача не найдена, свойства `taskTitle` и `taskDescription` сбрасываются.
4. Выбор языка программирования:
- При изменении значения выпадающего списка выбора языка программирования (`select`), вызывается метод `onLanguageChange`.
 - Метод обновляет значения свойств `selectedLanguage` и `languageName` с выбранным языком программирования и его названием соответственно.
5. Отправка решения:
- При щелчке на кнопку "Отправить", вызывается метод `submitSolution`.
 - Если выбран файл решения, метод выполняет следующие шаги:
 - Преобразует файл в формат `base64` и сохраняет его в переменную `base64File`.
 - Считывает содержимое файла как текст и сохраняет его в переменную ``code`` - Выполняет POST-запрос на адрес ``http://37.252.0.155:3000/handleSolution`` с параметрами:
 - ``solutionFileBase64``: содержимое файла решения в формате `base64`.
 - ``taskId``: идентификатор текущей задачи.
 - ``language``: выбранный язык программирования.
 - В случае успешного выполнения запроса, полученный результат (статус решения и ошибка на тесте) сохраняется в соответствующие свойства ``solutionStatus`` и ``failureTest``.
 - Вызывается метод ``submitSolutionToServer`` для отправки решения на сервер.

6. Отправка решения на сервер:

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

- Метод `submitSolutionToServer` выполняет POST-запрос на адрес `http://37.252.0.155:8080/api/solutions/add` для отправки решения на сервер.
 - В теле запроса передаются следующие параметры:
 - `code`: содержимое решения.
 - `language`: выбранный язык программирования.
 - `status`: статус решения.
 - `used_time`: время выполнения решения.
 - `used_memory`: использованная память.
 - `error_test`: ошибка на тесте.
 - `contest_name`: название конкурса из локального хранилища.
 - `task_name`: название текущей задачи.
 - Заголовок запроса должен содержать авторизационный токен из локального хранилища и тип контента `"Content-Type": "application/json"`.
 - В случае успешного выполнения запроса, полученный ответ логируется в консоль, а также добавляется новая запись о отправке в массив `submissions` (содержащий информацию о последних 4 отсылках).
 - Если возникает ошибка при отправке решения на сервер, ошибка логируется в консоль.
7. Отображение результатов:
- В разделе "Последняя отсылка" отображается статус решения из свойства `solutionStatus`.
 - В разделе "Ошибка на тесте" отображается ошибка на тесте из свойства `failureTest`.
 - В таблице "Последние 4 отсылки" отображается информация о последних 4 отсылках из массива `submissions`.

4.1.2. Требования к взаимодействию клиентской и серверной частей

Клиентская часть (веб-приложение) взаимодействует с сервером по протоколу HTTPS с использованием GET и POST запросов. Из веб-приложения отправляются следующие запросы:

Процесс авторизации (Отправка на сервер данных из формы авторизации):

- POST Request: `http://37.252.0.155:8080/api/auth/login`

```
headers: {
  'Content-Type': 'application/json'
}
```

Получение информации о преподавателе:

GET Request: `http://37.252.0.155:8080/api/profile/get-teacher-info`

```
headers: {
  Authorization: `Bearer ${token}`
}
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

Получение информации о студенте:

GET Request: <http://37.252.0.155:8080/api/profile/get-student-info>
 headers: {
 Authorization: `Bearer \${token}`
 }

Получение списка всех посылок:

GET Request: <http://37.252.0.155:8080/api/solutions/get-all>
 headers: {
 Authorization: `Bearer \${token}`
 }

Получение списка всех посылок:

GET Request: <http://37.252.0.155:8080/api/solutions/get-all-user>
 headers: {
 Authorization: `Bearer \${token}`
 }

Получение списка всех констестов:

GET Request: <http://37.252.0.155:8080/api/contest/get-all>
 headers: {
 Authorization: `Bearer \${token}`
 }

Авторизация в ejudge через puppeteer:

POST Request: <http://37.252.0.155:3000/auth>

```
{
  login: "ejudge",
  password: "ejudge",
  contestID: this.contestId,
},
{
  headers: {
    "Content-Type": "application/json",
  },
}
```

Получение задач с Ejudge с помощью puppeteer:

GET Request: <http://37.252.0.155:3000/parseTasks>

Добавление констеста в базу данных:

POST Request: <http://37.252.0.155:8080/api/contest/add>
 headers: {
 "Content-Type": "application/json",
 "Authorization": `Bearer \${localStorage.getItem('token')}`
 }

Получение рейтинга по оценкам:

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 T3 01–1

GET Request: <http://37.252.0.155:8080/api/grade/by-marks>
 headers: {"Authorization": `Bearer \${token}`}

Получение рейтинга по задачам:

GET Request: <http://37.252.0.155:8080/api/grade/by-tasks>
 headers: {"Authorization": `Bearer \${token}`}

Выгрузка рейтинга по оценкам:

GET Request: <http://37.252.0.155:8080/api/grade/by-marks/download-csv>
 headers: {"Authorization": `Bearer \${token}`}

Выгрузка рейтинга по задачам:

GET Request: <http://37.252.0.155:8080/api/grade/by-tasks/download-csv>
 headers: {"Authorization": `Bearer \${token}`}

Создание аккаунта студента:

GET Request: <http://37.252.0.155:8080/api/auth/register>

```
headers: {
  'Content-Type': 'application/json',
},
```

Обработка посылки через Ejudge используя puppeteer:

POST Request: <http://37.252.0.155:3000/handleSolution>

```
{
  solutionFileBase64: base64File,
  taskID: this.currentTask.problemId,
  language: this.selectedLanguage,
};
```

Добавление посылки в базу данных:

POST Request: <http://37.252.0.155:8080/api/solutions/add>

```
{
  {
    code: code,
    language: this.languageName,
    status: this.solutionStatus,
    used_time: 0.2,
    used_memory: 2.8,
    error_test: this.failureTest,
    contest_name: localStorage.getItem('contestName'),
    task_name: this.taskTitle,
  },
  {
    headers: {
      Authorization: `Bearer ${localStorage.getItem("token")}`,
      "Content-Type": "application/json",
    },
  },
}
```

Получение результата используя puppeteer:

GET Request: <http://37.252.0.155:3000/getResult>

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

4.1.3 Требования к серверу для парсинга Ejudge.

Техническое задание для разработки Puppeteer-API для парсинга сайта Ejudge

1. Цель проекта Разработка API для парсинга сайта Ejudge с использованием Puppeteer.
2. Общие требования 2.1. Использование Node.js и Puppeteer. 2.2. Реализация функций аутентификации, отправки решений, получения результатов и парсинга задач. 2.3. Ограничение скорости запросов для защиты от абыюза сервера. 2.4. Использование CORS и Express.js для обработки HTTP-запросов.
3. Функциональные требования 3.1. Аутентификация (auth)

- Принимает логин, пароль и номер конкурса.
- Возвращает сообщение об успешной аутентификации или ошибке.

3.2. Обработка решений (handleSolution)

- Принимает base64 кодированный файл решения, номер задачи и язык программирования.
- Отправляет решение на сервер Ejudge.
- Возвращает сообщение об успешной обработке решения или ошибке.

3.3. Получение результатов (getResult)

- Возвращает статус решения и сообщение об ошибке (если есть) или сообщение об успешном получении результатов.

3.4. Парсинг задач (parseTasks)

- Возвращает список задач с информацией о каждой задаче (ID, ограничения по времени и памяти, заголовок, описание, примеры входных и выходных данных).
- В случае ошибки возвращает сообщение об ошибке.

4. Нестандартные условия 4.1. Защита от абыюза сервера (DDoS). На 5 минут максимум 500 запросов.
5. Тестирование 5.1. Проверить аутентификацию с корректными и некорректными данными. 5.2. Проверить отправку решения и получение результатов для разных задач и языков программирования. 5.3. Проверить парсинг задач и корректность полученной информации. 5.4. Протестировать работу API с ограничением скорости запросов.
6. Документация 6.1. Описание функций и параметров API. 6.2. Примеры использования API.
7. Безопасность 7.1. Убедиться, что конфиденциальные данные пользователей (логин и пароль) защищены и не могут быть перехвачены. 7.2. Обеспечить обработку ошибок, чтобы избежать падения сервера или открытия уязвимостей.
8. Доступность 8.1. Поддержка CORS для доступности API на разных доменах. 8.2. Обработка HTTP-запросов и ответов должна быть стабильной и надежной, чтобы поддерживать непрерывную работу сервиса.
9. Обслуживание и поддержка 9.1. Предусмотреть возможность легкого масштабирования и обновления API. 9.2. Документация должна содержать все необходимые инструкции

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

для установки, настройки и использования API. 9.3. Осуществлять регулярное обновление и поддержку кода в соответствии с изменениями на сайте Ejudge и обновлениями используемых технологий.

10. План разработки

10.1. Определение требований и планирование.

10.2. Разработка функциональности и написание кода.

10.3. Тестирование и отладка.

10.4. Подготовка документации.

10.5. Запуск и поддержка.

4.1.4. Требования к входным данным

Входные данные от пользователя поступают в виде нажатий на кнопки, переходов по ссылкам, заполнения текстовых полей, меню выбора времени, слайдеров и чекбоксов. Описание вводимых данных представлено в проекте дизайна на платформе Figma, ссылка на который указана в разделе 4.1.4.

Все входные данные отправляются на сервер в формате json.

4.1.5. Требования к выходным данным

При первом запуске пользователю после загрузки отображается экран авторизации, экран авторизации должен включать в себя текстовые поля для ввода логина и пароля, кнопку авторизации. Возле текстовых полей при некорректном вводе должна отображаться ошибка

После успешной авторизации у пользователя открывается личный кабинет учащегося.

Экран личного кабинета учащегося должен включать в себя надписи с указанием имени, фамилии, отчества, ID пользователя, номер и буква класса, направление, номер группы, количество решённых задач, рейтинг по задачам и рейтинг по оценкам. В правом верхнем углу находится фиктивная фотография профиля (содержит инициалы). Также из учётной записи должна быть возможность выйти с помощью выпадающего меню в верхней части сайта в блоке с именем и фотографией профиля (на которой отображаются инициалы).

Примерный состав реализуемых функций и расположения элементов представлен в виде проекта на платформе Figma.

<https://www.figma.com/file/ZL37p1tu12ka5hM0BP0xQB/%D0%A2%D0%B5%D1%81%D1%82%D0%B8%D1%80%D1%83%D1%8E%D1%89%D0%B0%D1%8F-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0?node-id=0%3A1&t=YIWjZwkeNxVJLXZ9-1>

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

4.2. Требования к надёжности

4.2.4. Обеспечение устойчивого функционирования

Система должна обеспечивать непрерывную работу в течение всего периода проведения соревнований, не допускать падения или зависания во время работы с системой.

4.2.5. Контроль входной информации

Система должна проверять входные данные и уведомлять пользователя об ошибках, если введены некорректные данные.

4.2.6. Контроль выходной информации

Система должна генерировать корректные результаты, проверять их и уведомлять пользователя об ошибках, если необходимо.

4.2.7. Время восстановления после отказа

Система должна иметь возможность восстановления в случае сбоев или отказов в работе в кратчайшие сроки.

4.3. Условия эксплуатации

4.3.4. Климатические условия эксплуатации

Климатические условия эксплуатации, при которых должны предоставляться заданные характеристики, обязаны удовлетворять требованиям, предъявляемым к техническим устройствам, на которых запущена программа.

4.3.5. Требования к видам обслуживания

Необходимо осуществлять поддержку работоспособности сервера.

4.3.6. Требования к численности и квалификации персонала

Для запуска и взаимодействия с системой необходим один человек.

4.4. Требования к составу и параметрам технических средств

Веб-приложение должно корректно функционировать на устройствах под управлением ОС Windows 10 и выше, MacOS, Linux, а также на мобильных устройствах под управлением ОС IOS и Android версий позднее 2019 года выпуска.

4.4.1. Минимальные требования к сетевому оборудованию

Сервер приложений:

Процессор: 4 ядра, частота 2ГГц и более.

Оперативная память: 4Гб.

Тип накопителя: NVMe/SSD.

Свободное дисковое пространство (50ГБ и более. Зависит от объёма хранимых данных)

Пропускная способность сети: 100Мбит/с

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

Сервер базы данных:

Процессор: 2 ядра, частота 2ГГц и более.

Оперативная память: 2гб.

Тип накопителя: NVMe/SSD.

Свободное дисковое пространство (100 ГБ и более. Зависит от объёма хранимых данных)

Пропускная способность сети: 100Мбит/с

4.5. Требования к информационной и программной совместимости**4.5.4. Требования к исходным кодам и языкам программирования**

Веб-приложение должно быть разработано с использованием фреймворка Vue JS.

Используемые языки разметки: HTML, CSS. Для остальных задач используется JavaScript.

4.5.5. Требования к программным средствам, используемым программой

Веб-приложение должно корректно работать в браузерах Firefox, Google Chrome, Microsoft Edge, Yandex, Safari последних версий.

4.6. Требования к маркировке и упаковке

Требования к маркировке и упаковке не предъявляются.

4.7. Требования к транспортированию и хранению

Исходный код программы должен храниться на веб-сервисе GitHub в организации <https://github.com/orgs/Task-Testing-System-Devs>. Веб-приложение должно быть доступно в сети интернет по IP-адресу удаленного сервера.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

5.1. Состав программной документации

«Реализация «Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга». Техническое задание (ГОСТ 19.201-78);

«Реализация «Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга». Программа и методика испытаний (ГОСТ 19.301-78);

«Реализация «Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга». Пояснительная записка (ГОСТ 19.404-79);

«Реализация «Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга». Руководство оператора (ГОСТ 19.505-79);

«Реализация «Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга». Текст программы (ГОСТ 19.401-78);

5.2. Специальные требования к программной документации

Документы к программе должны быть выполнены в соответствии с ГОСТ 19.106-78 и ГОСТами к каждому виду документа (см. п. 5.1.).

Пояснительная записка должна быть загружена в систему Антиплагиат через LMS «НИУ ВШЭ».

Техническое задание и пояснительная записка, титульные листы других документов должны быть подписаны руководителем разработки и исполнителем.

Документация и программа сдаётся в электронном виде в формате .pdf или .docx. в архиве формата .zip или .rar.

За три дня до защиты комиссии все материалы курсового проекта: программная документация, программный проект, исполняемый файл, отзыв руководителя, отчёт системы Антиплагиат должны быть загружены одним или несколькими архивами в проект дисциплины «Курсовой проект» в личном кабинете в информационной образовательной среде SmartLMS НИУ ВШЭ.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

6. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

6.1. Предполагаемая потребность

Программа будет использоваться учащимися Лицея НИУ ВШЭ для решения задач по программированию и просмотра рейтинга учащихся в режиме реального времени, а также преподавателями Лицея НИУ ВШЭ для отслеживания количества решенных задач учащимися, контроля рейтинга, получения информации об оценках.

6.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами

	Timus Online Judge	Яндекс Контест	Informatics	NLOGN	Разрабатываемое решение
Возможность регистрации без приглашения	+	+	+	-	-
Рейтинг по определенному соревнованию	+	+	+	?	+
Общий рейтинг пользователей системы	+	-	+	-	+
Возможность автоматического выставления оценки за соревнование	-	-	-	-	+
Возможность выгрузки таблицы с оценками (рейтингом) всех пользователей за соревнования	?	+	-	-	+
Наличие архива задач, из которых	+	-	+	-	+

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

RU.17701729.05.10-01 ТЗ 01–1

можно создавать соревнование					
Количество поддерживаемых языков программирования	18	30	16	16	16

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

7. СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

Стадия разработки	Этапы разработки	Содержание работ	Сроки выполнения
Техническое задание	Обоснование необходимости разработки	Постановка задачи	01.09.2022 – 30.09.2022
	Разработка и утверждение технического задания	Определение требований к программе, этапов и сроков разработки и других требований	30.09.2022- 01.02.2023
		Согласование технического задания вместе с научным руководителем	01.02.2023- 08.02.2023
		Внесение окончательных правок в техническое задание, утверждение технического задания	08.02.2023- 15.02.2023
Разработка программного продукта	Разработка программы (написание кода)	Разработка серверной части, разработка пользовательского интерфейса на клиентских частях	15.02.2023- 01.04.2023
		Налаживание взаимодействия между серверной и клиентской частями	01.04.2023- 15.04.2023
	Разработка программной документации	Разработка программных документов	15.04.2023- 30.04.2023
	Испытания программы	Разработка, согласование и утверждение порядка и методики испытаний	01.05.2023- 05.05.2023
		Проведение предварительных испытаний	05.05.2023- 08.05.2023
		Корректировка программы и программной документации по результатам испытаний	08.05.2023- 11.05.2023

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

8. ПОРЯДОК КОНТРОЛЯ И ПРИЁМКИ

8.1. Виды испытаний

Производится проверка корректного выполнения программой заложенных в нее функций, то есть осуществляется функциональное тестирование программы. Функциональное тестирование осуществляется в соответствии с документом «Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга».

Программа и методика испытаний (ГОСТ 19.301-79), в котором указывают:

- 1) перечень функций программы, выделенных в программе для испытаний, и перечень требований, которым должны соответствовать эти функции (со ссылкой на пункт 4.1.1. настоящего технического задания);
- 2) перечень необходимой документации и требования к ней (со ссылкой на пункты 5.1 и 5.2 настоящего технического задания);
- 3) методы испытаний и обработки информации;
- 4) технические средства и порядок проведения испытаний;

Сроки проведения испытаний обсуждаются дополнительно.

8.2. Общие требования к приемке работы

Прием программы будет утвержден при корректной работе программы в соответствии с пунктом 4.1.1 при различных входных данных, соответствующих условиям в пункте 4.1.2 данного документа и при предоставлении полной документации к продукту, указанной в пункте 5.1, выполненной в соответствии с требованиями, указанными в пункте 5.2 данного технического задания.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01				
Инв. № подл.	Подп. и	Взам. инв.	Инв. №	Подп. и