**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  \_\_\_\_\_\_\_ В.В. Куренков  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. | |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  старший преподаватель департамента программной инженерии  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.А. Павлочев  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.05.10-01 81 01-1-ЛУ** | | | | | | |
|  | |  | | | | |
| Исполнитель  студент группы БПИ219  \_\_\_\_\_\_\_\_\_ / С.О. Котовский/  «\_\_» \_\_\_\_\_\_\_ 2024 г. | | | | |
|  | | | | | | |
|  | | | | |  | |

**Москва 2024**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.05.10-01 81 01-1-ЛУ |  | |  | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга**  **Пояснительная записка**  **RU.17701729.05.10-01 81 01-1**  **Листов 23** | | | | |
|  | |  | | |
|  | | |
|  | | | | |
|  | | | |  |

**Москва 2024**

ОГЛАВЛЕНИЕ

[1. ВВЕДЕНИЕ 4](#_Toc161931866)

[1.1. Наименование программы и объекта, в котором программа используется 4](#_Toc161931867)

[1.2. Краткая характеристика области применения программы и объекта, в котором она используется 4](#_Toc161931868)

[2. ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ 5](#_Toc161931869)

[3. НАЗНАЧЕНИЕ РАЗРАБОТКИ 6](#_Toc161931870)

[3.1. Функциональное назначение 6](#_Toc161931871)

[3.2. Эксплуатационное назначение 6](#_Toc161931872)

[4. ТРЕБОВАНИЯ К ПРОГРАММЕ 7](#_Toc161931873)

[4.1. Постановка задачи на разработку программы 7](#_Toc161931874)

[4.2. Описание алгоритма и функционирования веб-приложения 7](#_Toc161931875)

[4.2.1. Описание доработок алгоритма авторизации пользователя в системе 7](#_Toc161931876)

[4.2.2 Описание доработок личного кабинета пользователя 7](#_Toc161931877)

[4.2.3 Описание доработок навигационного меню веб-приложения 8](#_Toc161931878)

[4.2.4. Описание доработок нижнего колонтитула сайта 8](#_Toc161931879)

[4.2.5. Обновление функционала админ-панели 8](#_Toc161931880)

[4.2.6. Регистрация пользователей 8](#_Toc161931881)

[4.2.7 Описание импорта контестов из Ejudge через “Создание контеста” 9](#_Toc161931882)

[4.2.8 Описание доработок списка контестов 9](#_Toc161931883)

[4.2.9 Описание доработок списка посылок 9](#_Toc161931884)

[4.2.10 Описание страницы с контестом 9](#_Toc161931885)

[4.2. Описание и обоснование выбора метода организации входных и выходных данных 10](#_Toc161931886)

[4.3.1 Страница авторизации 10](#_Toc161931887)

[4.3.2 Личный кабинет пользователя 11](#_Toc161931888)

[4.3.3 Навигационное меню 11](#_Toc161931889)

[4.3.4 Список контестов 12](#_Toc161931890)

[4.3.5 Список посылок (решений) 13](#_Toc161931891)

[4.3.6 Страница с контестом 14](#_Toc161931892)

[4.3. Описание алгоритма и функционирования backend-прослойки 15](#_Toc161931893)

[4.3.5. Описание программной реализации 15](#_Toc161931894)

[4.3.6. Требования к архитектуре 16](#_Toc161931895)

[4.3.7. Основные эндпоинты 17](#_Toc161931896)

[4.3.8. Требования к промежуточному программному обеспечению 18](#_Toc161931897)

[4.4.5. Основные аспекты 19](#_Toc161931898)

[4.4. Описание и обоснование состава технических и программных средств 19](#_Toc161931899)

[5. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 20](#_Toc161931900)

[5.2. Предполагаемая потребность 20](#_Toc161931901)

[5.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 20](#_Toc161931902)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 22](#_Toc161931903)

# ВВЕДЕНИЕ

## Наименование программы и объекта, в котором программа используется

Наименование темы разработки: «Разработка нового и доработка существующего функционала системы проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга».

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

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

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

«Разработка нового и доработка существующего функционала системы проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга» - это комплекс мероприятий по доработке курсового проекта «Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга», который, в свою очередь, является набором 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.

# 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.

# НАЗНАЧЕНИЕ РАЗРАБОТКИ

## Функциональное назначение

Разработка направлена на внесение технических, логических, визуальных и архитектурных доработок работы, выполненной в рамках курсового проекта в 2023 году «Система проверки решений по программированию для Лицея НИУ ВШЭ с автоматизированным подсчётом рейтинга».

## Эксплуатационное назначение

Мероприятия, проводимые в рамках разработки, направлены на улучшение пользовательского опыта и упрощения дальнейшей поддержки дорабатываемого комплекса программно-технических решений.

# ТРЕБОВАНИЯ К ПРОГРАММЕ

## Постановка задачи на разработку программы

Целью проекта является предоставление пользователю обновленной и улучшенной версии веб-приложения на Vue.js. Приложение должно быть удобным и простым в использовании. Приложение должно соответствовать основным требованиям к разработке, безопасности и защиты данных пользователей.

Побочная цель: обновить архитектуру backend-прослойки, взаимодействующей с ejudge. С учетом функционала официальной API Ejudge перенести максимальное количество методов на API вместо скриптовой реализации через Puppeteer для повышения производительности и удобства работы в формате REST API.

## Описание алгоритма и функционирования веб-приложения

* + 1. Описание доработок алгоритма авторизации пользователя в системе

Переработана визуальная и архитектурная составляющие страницы авторизации в веб-приложении:

* Добавлены иконки и текстовые подсказки (плейсхолдеры) в поля ввода почты и пароля.
* В правой части страницы отображается векторное изображение, под которым расположено название и текстовое описание назначения тестирующей системы.
* Обновлен логотип системы, цветовая гамма и расположение элементов на странице в соответствии с макетом.
* Обновлена архитектурная составляющая кода, связанного с авторизацией для лучшей модульности и простоты поддержки кода.

4.2.2 Описание доработок личного кабинета пользователя

Переработана визуальная и архитектурная составляющие страницы профиля (личного кабинета) в веб-приложении:

* Добавлена навигационная панель в левой части веб-приложения (ранее навигационная панель находилась в верхней части веб-приложения). Подробное описание новой навигационной панели можно найти в соответствующем пункте.
* Теперь в личном кабинете в верхней части страницы отображаются векторные изображения и карточка профиля.
* Карточка профиля содержит в себе ФИО пользователя, организацию, группу(группы), к которым привязан пользователь.
* Ниже располагается информационное поле, содержащее адрес электронной почты пользователя, а также кнопка “Выйти из аккаунта”.

4.2.3 Описание доработок навигационного меню веб-приложения

* Навигационное меню после доработок представляет собой панель с кнопками и кнопками, располагается всегда в левой части веб-приложения.
* Навигационное меню содержит кнопки “Контесты”, “Рейтинг”,  
  “Решения”, “Профиль”.
* Доступ к навигационной панели может быть получен с любой страницы веб-приложения кроме страницы авторизации.
* При авторизации с использованием данных категории “преподаватель” вместо кнопки “Личный кабинет” отображаются кнопки “Регистрация пользователей”, “Создать контест”.
* Для активной (выбранной) страницы используется подсветка кнопки и иконки в соответствии с макетом.
  + 1. Описание доработок нижнего колонтитула сайта

Нижний колонтитул был удален из веб-приложения в соответствии с макетом для экономии пространства на экране пользователя и повышения удобства взаимодействия с веб-приложением.

* + 1. Обновление функционала админ-панели

Админ-панель теперь представлена в виде двух кнопок - “Регистрация пользователей”, “Создать контест”. Главной новинкой стало добавление возможности создания контестов напрямую через интерфейс админ-панели. Эта функция упрощает процесс добавления и настройки контестов, делая его более интуитивно понятным и менее времязатратным.

* + 1. Регистрация пользователей

Администраторы теперь могут регистрировать пользователей двумя способами: ввод данных вручную через форму или путем загрузки CSV-файла с данными пользователей. Форма регистрации предусматривает ввод основных данных о пользователе, включая тип (студент или преподаватель), ФИО, департамент, группу, электронную почту и пароль. Это обновление направлено на упрощение и ускорение процесса регистрации большого числа пользователей, делая администрирование более эффективным.

4.2.7 Описание импорта контестов из Ejudge через “Создание контеста”

Данный компонент Vue.js, реализует функционал импорта контеста из системы Ejudge в базу данных нашего сервиса посредством ввода id. Далее puppeteer-прослойка используя id парсит задачи из контеста в ejudge с этим id и добавляет их в нашу базу.

4.2.8 Описание доработок списка контестов

Переработана визуальная и архитектурная составляющие страницы c контестами в веб-приложении:

* Используется структура карточек.
* Карточка контеста включает в себя название контеста, ID контеста, дата окончания приёма решений, доступные языки программирования для отправки решений, а также кнопка “Перейти к контесту”.

4.2.9 Описание доработок списка посылок

Переработана визуальная и архитектурная составляющие страницы c контестами в веб-приложении:

* Страница списка посылок переименована в “Решения” для лучшей понятности формулировки.
* На странице в верхней части отображается надпись “Ваши решения”.
* Решения пользователя имеют структуру карточек.
* Карточки отсортированы по времени отправки решения.
* Карточка решения содержит в себе id посылки, время исполнения кода, количество затраченной оперативной памяти в мегабайтах, язык программирования отправленного решения, а также вердикт тестирующей системы.
* При нажатии на карточку с решением пользователю отображается исходный код решения, а также кнопка “Скопировать код”, которая сохраняет код решения в буфер обмена устройства пользователя.

4.2.10 Описание страницы с контестом

Переработана визуальная и архитектурная составляющие страницы c контестами в веб-приложении:

* На странице используется структура карточек. На странице присутствуют: карточка контеста, карточка задач и карточка отправки решения.
* В верхней части страницы находится карточка с информацией о контесте.
* Карточка контеста включает в себя название контеста, ID контеста, дата окончания приёма решений, доступные языки программирования для отправки решений.
* Карточка задачи включает в себя название задачи, ограничение по времени, ограничение по памяти, названия и форматы файлов для входных и выходных данных, условие задачи, форматы ввода и вывода с описанием, далее расположены карточки с примерами входных и выходных данных.
* Карточка “Решение” содержит в себе выбор языка отправляемого решения, область-кнопка для прикрепления файла с кодом, а также текстовое поле для ввода кода вручную. Далее расположены кнопки “Отправить решение” и “Следующая задача”.

## Описание и обоснование выбора метода организации входных и выходных данных

4.3.1 Страница авторизации

* Входные данные:
  + **Информация для отображения на странице входа:** Данные, необходимые для формирования интерфейса страницы входа, могут быть загружены из сохраненных данных пользователя или требоваться для ввода по факту.
* Выходные данные:
  + **Данные формы входа:** Информация, введенная пользователем в форму входа (LoginForm), такая как имя пользователя и пароль, передается на сервер для аутентификации и авторизации пользователя.
  + **Запросы к API для аутентификации:** когда пользователь заполняет и отправляет форму входа, информация отправляется на сервер через API для проверки учетных данных. В зависимости от результата, пользователь может быть перенаправлен на другую страницу или получить сообщение об ошибке. При авторизации учитывается соответствие авторизационных данных одной из ролей (студент или преподаватель)
  + **Сообщения об ошибках или подтверждение успешного входа:** В ответ на отправленные данные формы сервер может возвращать сообщения об ошибках (например, если учетные данные неверны или аккаунт не найден) или подтверждение успешного входа, что может включать в себя токен доступа или перенаправление на главную страницу пользователя.

4.3.2 Личный кабинет пользователя

* Входные данные:
  + Запрос на сервер для получения веб-страницы.
* Выходные данные:
  + Выходные данные на странице профиля представляют собой информацию о пользователе, которая отображается на странице и которая изначально получена с сервера. В данном случае выходные данные включают в себя:
  + **ID пользователя**: Уникальный идентификатор пользователя в системе.
  + **Полное имя пользователя**: Отображаемое имя пользователя.
  + **Электронная почта пользователя**: Адрес электронной почты, ассоциированный с профилем пользователя.
  + **Организация пользователя**: Название организации, к которой относится пользователь.
  + **Группы пользователя**: Список групп, к которым пользователь принадлежит.

4.3.3 Навигационное меню

* Входные данные:
  + **Статус авторизации пользователя**: Информация о том, авторизован ли пользователь, и если да, то какой категории пользователь принадлежит (например, студент, преподаватель). Это определяет, какие элементы меню должны быть отображены (например, кнопка “Создание контеста” для преподавателей).
  + **Текущая страница**: Идентификация текущей страницы, на которой находится пользователь, чтобы соответствующая кнопка меню была подсвечена как активная.
  + **Размер экрана/устройства**: Информация о размере экрана пользователя может потребоваться для определения, следует ли отображать навигационное меню в минимизированном состоянии с иконками без надписей для экономии пространства.
* Выходные данные:
  + **Отображаемые элементы меню**: На основе входных данных меню динамически адаптируется и отображает доступные пользователю элементы, включая специальные кнопки для различных категорий пользователей (например, “Создание контеста” для преподавателей).
  + **Состояние отображения меню**: Полное или минимизированное состояние меню в зависимости от размера экрана пользователя, где в минимизированном состоянии отображаются только иконки кнопок.
  + **Подсветка активной страницы**: Визуальная подсветка кнопки и иконки в меню для текущей активной страницы, на которой находится пользователь, в соответствии с дизайном интерфейса.
  + **Доступность элементов меню**: Отображение или скрытие определенных элементов меню в зависимости от статуса авторизации пользователя, чтобы обеспечить соответствующий уровень доступа к функциям веб-приложения.

4.3.4 Список контестов

* Входные данные:
  + **Запрос к серверу**: Для отображения актуального списка контестов, веб-приложение отправляет запрос на сервер для получения списка доступных контестов, включая всю необходимую информацию о каждом из них.
  + **Информация о пользователе**: Данные о текущем статусе авторизации пользователя и его роли могут определять, какие контесты отображаются и доступны для участия или просмотра.
* Выходные данные:
  + **Список контестов**: В ответ на запрос к серверу, клиент получает данные о контестах, которые затем отображаются на странице. Каждая карточка контеста содержит:
  + **Название контеста**: Название, по которому пользователи могут идентифицировать контест.
  + **ID контеста**: Уникальный идентификатор, который может быть использован для ссылок или запросов связанных с контестом.
  + **Дата окончания приема решений**: Информация о том, до какой даты можно отправлять решения на задачи контеста.
  + **Доступные языки программирования**: Перечень языков, на которых участники могут предоставлять свои решения.
  + **Кнопка “Перейти к контесту”**: Элемент управления, позволяющий переходить на страницу конкретного контеста для более подробной информации или для участия в контесте.
* **Визуализация информации**: Данные о контестах визуализируются в виде карточек, организуя информацию о каждом контесте в легкодоступной и понятной форме, что упрощает навигацию и выбор контестов для пользователей.

4.3.5 Список посылок (решений)

* Входные данные:
  + **Запрос пользователя**: Пользователь заходит на страницу "Решения", что инициирует запрос на сервер за списком его посылок/решений.
  + **Идентификация пользователя**: Для отображения решений конкретного пользователя, система использует данные авторизации или сессии пользователя, чтобы запросить на сервере именно его решения.
  + **Взаимодействие с элементами UI**: Пользователь может нажимать на карточки решений для просмотра деталей решения или использовать кнопку "Скопировать код" для копирования исходного кода решения.
* Выходные данные:
  + **Список решений пользователя**:
  + В ответ на запрос, сервер отправляет данные решений, которые затем отображаются на странице. Каждая карточка решения включает в себя:
  + **ID посылки**: Уникальный идентификатор решения.
  + **Время исполнения кода**: Как долго выполнялось решение.
  + **Количество затраченной оперативной памяти**: Объем памяти, использованный при выполнении решения.
  + **Язык программирования**: Язык, на котором было написано решение.
  + **Вердикт тестирующей системы**: Результат проверки решения (например, принято, ошибка компиляции, неправильный ответ и т.д.).
* **Детальная информация по решению**:
  + При выборе карточки с решением, пользователю отображается исходный код отправленного решения и предоставляется возможность скопировать этот код в буфер обмена.
* **Визуализация и удобство использования**:
  + Карточки решений отсортированы по времени отправки, что позволяет пользователям легко найти последние или конкретные решения.
  + На странице имеется надпись “Ваши решения” в верхней её части, что делает назначение страницы понятным с первого взгляда.

4.3.6 Страница с контестом

* Входные данные:
  + **Запрос информации о контесте**: При переходе на страницу контеста, веб-приложение отправляет запрос на сервер для получения данных о конкретном контесте.
  + **Идентификация пользователя и его права**: Данные о пользователе и его статусе для определения, какие операции он может выполнять на странице (например, отправка решения).
  + **Взаимодействие пользователя с элементами страницы**:
  + Выбор языка программирования для отправки решения.
  + Прикрепление файла с решением или ввод кода вручную в предусмотренное текстовое поле.
  + Нажатие на кнопки “Отправить решение” и “Следующая задача”.
* Выходные данные:
  + **Информация о контесте**:
    - На странице отображается карточка с детальной информацией о контесте, включая:
    - Название контеста.
    - ID контеста.
    - Дата окончания приема решений.
    - Доступные языки программирования для отправки решений.
  + **Список задач контеста**:
    - Для каждой задачи отображается карточка с информацией, включающей:
    - Название задачи.
    - Ограничения по времени и памяти.
    - Названия и форматы файлов для входных и выходных данных.
    - Условие задачи, форматы ввода и вывода с описанием.
    - Примеры входных и выходных данных.
  + **Интерфейс отправки решения**:
    - Предоставляется карточка для отправки решения задачи, где пользователь может:
    - Выбрать язык программирования.
    - Прикрепить файл с кодом решения или ввести код вручную.
    - Отправить решение на проверку нажатием на кнопку “Отправить решение”.
    - Перейти к следующей задаче контеста с помощью кнопки “Следующая задача”.

## Описание алгоритма и функционирования backend-прослойки

* + 1. Описание программной реализации
* В ходе разработки backend-прослойки была реализована многоуровневая архитектура приложения, обеспечивающая гибкое взаимодействие между компонентами системы. Проект структурирован на основе директорий, каждая из которых выполняет определенную роль в общей архитектуре приложения:
* Директория api содержит основные точки входа для обработки HTTP-запросов, что позволило централизованно управлять маршрутизацией и делегированием запросов соответствующим контроллерам.
* В config сосредоточена конфигурационная информация приложения, включая настройки для работы с Puppeteer и параметры ограничения скорости запросов. Это обеспечило легкость в управлении настройками и их изменении в зависимости от среды выполнения.
* Контроллеры в controllers отвечают за обработку бизнес-логики приложения. Были реализованы контроллеры для аутентификации пользователей, работы с контестами, задачами, посылками решений и получения информации об пользователях и судействе задач.
* Модули в ejudge предоставляют интеграцию с системой Ejudge, обрабатывая специфичные для неё операции, такие как получение информации о контестах, задачах и результатах посылок.
* middlewares содержит промежуточное ПО для Express-приложения, включая ограничение скорости запросов и загрузку файлов, что повысило безопасность и надежность приложения.
* В puppeteer сосредоточены скрипты для управления браузером, автоматизации действий пользователя и взаимодействия с веб-интерфейсами, что позволило автоматизировать ряд задач, не имеющих API.
* Routes содержит описание маршрутизации в приложении, связывая URL-пути с соответствующими контроллерами, и обеспечивали удобную навигацию по API.
* В utils вынесены вспомогательные функции и утилиты для обработки ошибок, парсинга HTML в JSON, регистрации пользователей и других общих задач.
* В процессе разработки особое внимание уделялось безопасности и оптимизации работы с ресурсами. Были реализованы следующие ключевые функции:
* Аутентификация и авторизация пользователей осуществляется через специализированные контроллеры, взаимодействующие с системой Ejudge и другими сервисами.
* Обработка запросов к Ejudge ведется через отдельные обработчики, что обеспечило централизованное управление взаимодействием с внешними системами.
* Работа с файлами (в том числе загрузка и обработка CSV-файлов для регистрации пользователей) была реализована с использованием промежуточного программного обеспечения и вспомогательных утилит, что повысило удобство работы с приложением.
* Логирование и обработка ошибок выполняется с помощью специализированных механизмов, что сделало процесс отладки более прозрачным и эффективным.
* Разработка backend-прослойки была направлена на обеспечение высокой степени модульности, масштабируемости и безопасности приложения, позволяя гибко настраивать его под различные задачи и условия эксплуатации.
  + 1. Требования к архитектуре
* Модульность и расширяемость:
* - Система разработана с применением модульного подхода, предусматривающего возможность легкой интеграции дополнительных компонентов и сервисов без значительного перепроектирования существующей архитектуры.
* - Каждый модуль должен быть ответственен за выполнение конкретной функциональности и иметь четко определенный интерфейс взаимодействия с другими модулями.
* Слои архитектуры:
* - Презентационный слой (Маршрутизация запросов): отвечает за прием входящих HTTP-запросов от клиентов, их маршрутизацию к соответствующим обработчикам в бизнес-логике и отправку ответов.
* - Бизнес-логика (Контроллеры): содержит логику обработки запросов, аутентификации пользователей, обработки и подачи решений, получения списка задач. Должен обеспечивать валидацию входящих данных и корректную обработку исключительных ситуаций.
* - Доступ к данным (Puppeteer): обеспечивает взаимодействие с внешними веб-сервисами для аутентификации, подачи решений и получения списка задач с использованием автоматизированного браузера.
* Безопасность:
* - Система включает механизмы защиты от частых и повторяющихся запросов (Rate Limiting).
* - Предусмотрена обработка исключений и ошибок с целью предотвращения утечки чувствительной информации.
  + 1. Основные эндпоинты
* **GET /api/master/get-user** - Получение информации о пользователе: Запрашивает и возвращает информацию о пользователе в системе Ejudge, используя его идентификаторы в контексте определенного контеста.
* **GET /api/master/contest-status-json** - Получение статуса контеста: Возвращает детальную информацию о состоянии указанного контеста, включая доступные языки программирования, время начала и окончания, а также список задач.
* **GET /api/master/problem-statement-json** - Получение условия задачи: Возвращает полное описание задачи, включая условие, ограничения по времени и памяти, а также примеры входных и выходных данных.
* **POST /api/master/submit-run** - Отправка решения на проверку: Принимает файл с решением пользователя, информацию о задаче и языке программирования, после чего отправляет решение на сервер Ejudge для проверки.
* **GET /api/master/get-submit** - Получение информации о посылке: Запрашивает и возвращает детали о результате проверки посылки, включая статус, время выполнения и использованную память.
* **POST /api/register-users** - Регистрация пользователей из CSV: Принимает файл в формате CSV с информацией о пользователях (студентах или преподавателях) и выполняет их массовую регистрацию в системе.
  + 1. Требования к промежуточному программному обеспечению
* Ограничение скорости запросов (Rate Limiting):
  + Для предотвращения DoS-атак и обеспечения справедливого распределения ресурсов между пользователями было реализовано ограничение скорости запросов.
  + Конфигурация ограничения скорости включает в себя установку временного окна в 5 минут и максимальное количество запросов с одного IP-адреса, равное 500.
* Обработка загрузки файлов (File Upload Handling):
  + Реализована поддержка загрузки файлов через мультипарт-формы, что позволяет пользователям отправлять решения задач и CSV-файлы для регистрации.
  + Для обработки и сохранения загруженных файлов используется middleware на базе библиотеки multer, обеспечивающей временное хранение файлов в указанной директории.
* Аутентификация запросов:
  + Промежуточное ПО должно поддерживать механизмы аутентификации запросов для проверки подлинности пользователей и предоставления доступа к защищенным ресурсам.
  + Требуется интеграция с внешними системами аутентификации, такими как система Ejudge, через специальные заголовки запросов или токены доступа.
* Обработка ошибок (Error Handling):
  + Для централизованной обработки возникающих исключений и ошибок в приложении требуется middleware, формирующее стандартизированные ответы об ошибках.
  + Необходимо предоставление информативных сообщений об ошибках с соответствующими HTTP-статус кодами.
* Логирование:
  + Для мониторинга и анализа работы приложения реализован механизм логирования запросов и ответов, а также внутренних событий системы.

4.4.5. Основные аспекты

Разработанная система должна обеспечивать высокую производительность, безопасность и удобство взаимодействия с внешними сервисами, а также поддерживать легкость внесения изменений и добавления новой функциональности.

## Описание и обоснование состава технических и программных средств

Для обеспечения надежной работы проекта необходим следующий состав технических и программных средств:

1. Серверная часть:

* Операционная система: Linux-based (например, Ubuntu 20.04 LTS).
* Web-сервер: Nginx.
* Среда выполнения: Node.js (версия 14.x или выше).

1. Клиентская часть:

* Веб-браузеры: Google Chrome (версия 70 и выше), Mozilla Firefox (версия 63 и выше), Safari (версия 12 и выше), Microsoft Edge (версия 79 и выше).
* Разрешение экрана: минимальное 1024x768 пикселей.
* JavaScript: должен быть включен для корректной работы приложения.

Выбор технических и программных средств обосновывается следующими соображениями:

* Linux-based операционная система обеспечивает высокую стабильность и безопасность сервера, а также широкие возможности настройки и управления.
* Nginx в качестве веб-сервера обеспечивает высокую производительность и надежность, хорошо справляется с большим количеством одновременных запросов.
* Node.js позволяет создавать высокопроизводительные серверные приложения с использованием языка программирования JavaScript.
* Поддержка современных веб-браузеров и разрешений экрана позволяет достичь максимальной совместимости и удобства использования приложения для большинства пользователей.
* Включенный JavaScript обеспечивает корректную работу интерактивных элементов пользовательского интерфейса и обработку запросов на стороне клиента.

# ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## Предполагаемая потребность

Доработки системы необходимы для более комфортного пользовательского опыта как учащихся, так и преподавателей, а также для увеличения возможностей по дальнейшей доработке системы и упрощения разработки для программистов, взаимодействующих с исходным кодом системы.

## Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами

(текст или таблица – Яндекс Контест, Informatics, Timus Online judge, nlogn)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 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 можно создавать соревнование | + | - | + | - | + |
| Количество поддерживаемых языков программирования | 18 | 30 | 16 | 16 | 16 |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |