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

Факультет компьютерных наук

Образовательная программа «Программная инженерия»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Научный руководитель,  Приглашенный преподаватель департамента программной инженерии    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.К. Горденко  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Тестирующая система c заданиями в виде изображений**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.05.11-01 81 01-1-ЛУ** | | |
|  |  | |
| Исполнитель:  студент группы БПИ206  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Г. В. Вавилов /  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. | |
|  | | |
|  | |  |

**Москва 2023**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.05.11-01 81 01-1-ЛУ |  | |  | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Тестирующая система c заданиями в виде изображений**  **Пояснительная записка**  **RU.17701729.05.11-01 81 01-1**  **Листов 24** | | | | |
|  | |  | | |
|  | | |
|  | | | | |
|  | | | |  |

**Москва 2023**

Содержание

[Введение 5](#_Toc1)

[1.1. Наименование: 5](#_Toc2)

[1.2. Документ, на основании которого ведется разработка 5](#_Toc3)

[2. Назначение и область применения 5](#_Toc4)

[2.1. Назначение 5](#_Toc5)

[2.1.1. Функциональное назначение 5](#_Toc6)

[2.1.2. Эксплуатационное назначение 5](#_Toc7)

[2.2.1. Обзор аналогов 6](#_Toc8)

[3. Технические характеристики 10](#_Toc9)

[3.1. Постановка задачи на разработку программы 10](#_Toc10)

[3.1.1. Требования к панели преподавателя 10](#_Toc11)

[3.1.2. Требования к серверной части 10](#_Toc12)

[3.2. Описание применяемых математических методов 11](#_Toc13)

[3.3. Описание алгоритма и функционирования программы 12](#_Toc14)

[3.3.1. Описание реализации программы 12](#_Toc15)

[3.4. Описание метода организации входных и выходных данных 15](#_Toc16)

[3.4.1. Входные данные 15](#_Toc17)

[3.4.2. Выходные данные 16](#_Toc18)

[3.5. Описание состава технических и программных средств 16](#_Toc19)

[3.5.1. Программные средства 16](#_Toc20)

[3.5.1.1. database 16](#_Toc21)

[3.5.1.2. backend 17](#_Toc22)

[3.5.1.3. admin panel 17](#_Toc23)

[3.5.1.4. frontend 17](#_Toc24)

[3.5.1.5. webshot 17](#_Toc25)

[3.5.1.6. vkontakte bot 17](#_Toc26)

[3.5.1.7. telegram bot 17](#_Toc27)

[3.5.2. Технические средства 18](#_Toc28)

[4. Ожидаемые технико-экономические показатели 19](#_Toc29)

[4.1. Предполагаемая потребность 19](#_Toc30)

[4.2. Ориентировочная экономическая эффективность 19](#_Toc31)

[5. Список используемых источников 21](#_Toc32)

[6. СПИСОК ТЕРМИНОВ 22](#_Toc33)

[1. Чат-бот 22](#_Toc34)

[2. Плагин 22](#_Toc35)

[3. Telegram 22](#_Toc36)

[4. *csv* 22](#_Toc37)

[5. png 22](#_Toc38)

[6. xlsx 22](#_Toc39)

[7. REST API 22](#_Toc40)

[8. Кэширование 22](#_Toc41)

[9. HTTP 23](#_Toc42)

# Введение

## **Наименование**:

* + 1. **Название темы разработки на русском языке:**

Тестирующая система c заданиями в виде изображений.

* + 1. **Название темы разработки на английском языке:**

Testing System with Tasks in the Form of Images.

## Документ, на основании которого ведется разработка

Программа выполнена в рамках учебного плана подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и утвержденная академическим руководителем программы тема курсового проекта «Тестирующая система c заданиями в виде изображений».

# Назначение и область применения

## **Назначение**

## Функциональное назначение

Тестирующая система решает задачу автоматизации проведения тестов.   
  
Для Преподавателей система предоставляет возможность создания учебных курсов, добавления Учеников в учебный курс, создания тестовых заданий и объединения их в тесты, проведения тестирования среди Учеников курса, сбора статистики по проведенному тестированию.

Для Учеников система предоставляет возможность прохождения назначенных тестов.

## Эксплуатационное назначение

Система эксплуатируется Преподавателями и Учениками для проведения тестирования в рамках образовательных программ.

* 1. Характеристика области применения:

Система применяется в образовательных учреждениях, таких как школы, колледжи и университеты, для проведения тестирования и оценки знаний, в том числе и для онлайн-обучения.

## Обзор аналогов

* + - 1. **Kahoot**

Платформа для создания и проведения тестов в игровой форме. Предоставляется возможность использования сервиса в веб-браузере или в мобильном приложении.

Во время проведения викторины все участники викторины начинают отвечать на каждый вопрос одновременно. За каждый правильный ответ участник получает баллы. Чем быстрее был дан ответ, тем больше баллов получает участник. В конце викторины можно получить рейтинг участников.

Виды вопросов, которые могут быть использованы в викторине:

* Правда/Ложь
* Выбор одного ответа
* Множественный выбор
* Расположить элементы в правильном порядке
* Выбор числа в определенном интервале
* Открытые вопросы с короткими ответами

Для обучения в игровой форме Kahoot предлагает следующие возможности:

1. Color kingdom

2 соревнующиеся команды захватывают клетки на поле при правильном ответе на вопрос. Выигрывает команда, под контролем которой оказалось наибольшее количество клеток к концу раунда.

1. Treasure trove

Участники должны верно ответить на необходимое количество вопросов, чтобы собрать сокровище в сундук. После этого запускается мини-игра, в которой участник будет получать очки. Выигрывает участник с наибольшим количеством очков.

1. Drop pin

Участник ставит метку на изображение в качестве ответа. Чем ближе метка участника находится к эталонному ответу, тем больше очков он получает.

1. Image Reveal

Раскрытие части изображения при правильном ответе на вопрос.

Сервис остановил деятельность в России, что делает его использование невозможным.

* + - 1. **Moodle**

Настраиваемая система управления обучением. Позволяет создать собственный сайт с образовательными материалами, форумом, вики-страницами, банком задач и проверочными заданиями.

Исходный код находится в открытом доступе. Для создания сайта требуется настроенный веб-сервер, база данных, установленный язык программирования PHP, а также сервер, на котором будет развернута система.

Предоставляет возможность полностью кастомизировать интерфейс создаваемого сайта. Позволяет удобно настроить интеграции с внешними сервисами.

* + - 1. **Google forms**

Сервис для создания и прохождения опросов. Позволяет собрать опрос из заранее подготовленных блоков. При помощи надстройки появляется возможность проводить тесты.

Каждый вопрос задается текстом с возможностью форматирования.

Стандартные типы ответов на вопрос:

1. Однострочный текст (короткие ответы)
2. Многострочный текст (развернутый ответы)
3. Выбор одного ответа из списка
4. Выбор множества ответов из списка
5. Ответ в виде файлов
6. Выбор числового значения в диапазоне
7. Дата
8. Время

В опрос кроме вопросов можно вставить изображения, видео и блоки с текстом. Также можно разделять опрос на разделы.

Каждому вопросу можно задать количество баллов, которое можно набрать при правильном ответе на него. Результаты прохождения опроса можно экспортировать в виде таблицы.

Сервис поддерживает расширения в виде плагинов[2], которые можно найти в открытом маркетплейсе goolge.

* + - 1. **socrative**

Сервис для создания и проведения викторин и тестов.

Учитель имеет возможность создать тест, состоящий из вопросов. Поддерживаются следующие типы вопросов:

1. Ответ с множественным выбором
2. Ответ Истина/Ложь
3. Короткий текстовый ответ. Ответы сравниваются посимвольно, поэтому можно задать множество правильных ответов.

К вопросу можно также добавить изображения, количество баллов за правильный ответ и объяснение правильного ответа.

После создания теста можно запустить его в режиме викторины или в игровом режиме «Space race»

В «Space race» участники случайно разбиваются на равные команды и в течение ограниченного времени отвечают на случайные вопросы из викторины. Побеждает команда, набравшая наибольшее количество очков после окончания времени.

В режиме викторины есть 3 варианта ответа на вопросы:

1. Участник отвечает на вопросы в любом порядке.
2. Участник отвечает на вопросы в подряд. Без возможности возвращаться назад.
3. Участник отвечает на вопросы в порядке, заранее установленном преподавателем. Преподаватель в реальном времени разрешает участникам переходить к ответу на следующий вопрос.

В таблице представлены тарифные планы, которые предлагает платформа и их возможности.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Free | K-12 | Higher Ed & Corporate |
| Цена | 0$ | $90/год | $180/год |
| Количество участников в комнате | До 50 | До 50 | До 200 |
| Количество созданных тестов | 5 | Не ограничено | Не ограничено |
| Экспорт результатов | - | csv[4], xlsx[6] | csv[4], xlsx[6] |
| Рассылка результатов участникам | - | Электронная почта | Электронная почта |
| Ручное оценивание, частичные баллы | - | + | + |

* + - 1. **@quizbot**

Чат-бот[1] на платформе мессенджера telegram[3] для создания и проведения викторин. Позволяет создать закрытые вопросы с единственным верным ответом.

К вопросам можно добавлять изображения. Правильность выбранного ответа участник викторины узнает сразу. Есть возможность установить максимальное время ответа на каждый вопрос внутри викторины. Создатель викторины может просматривать результаты прохождения викторины другими участниками только внутри мессенджера. Бот предоставляет сервис бесплатно.

# Технические характеристики

## **Постановка задачи на разработку программы**

Приложение состоит из серверной части, чат-бота[1] в системе обмена сообщениями *telegram[3],* и клиентского приложения – панели преподавателя*.*

## Требования к панели преподавателя

* + - 1. Пользователь должен иметь возможность зарегистрироваться в системе в качестве преподавателя.
      2. Преподаватель должен иметь возможность просматривать список учебных курсов, на которых он является преподавателем.
      3. Преподаватель должен иметь возможность создать новый учебный курс.
      4. Преподаватель должен иметь возможность добавить других преподавателей в учебный курс.
      5. Преподаватель должен иметь возможность добавить учеников в учебный курс
      6. Преподаватель должен иметь возможность создать тестовое задание.
      7. Преподаватель должен иметь возможность создать тест, состоящий из тестовых заданий.
      8. Преподаватель должен иметь возможность назначить тест в качестве элемента контроля для учебного курса.
      9. Преподаватель должен иметь возможность выгрузить статистику по элементу контроля, по учебному курсу в формате *csv[4]*.

## Требования к серверной части

* + - 1. Сервер должен реализовать REST API[7] для управления ресурсами из клиентской части приложения.
      2. Сервер должен иметь возможность генерации изображения с тестовым заданием в формате *png[5]*.
    1. **Требования к чат-боту** 
       1. Бот должен производить аутентификацию пользователя при помощи адреса его электронной почты.
       2. Бот должен уведомлять ученика после добавления преподавателем теста в учебный курс, к которому принадлежит ученик.
       3. Ученик должен иметь возможность дать ответ на тестовое задание через интерфейс клиентского приложения telegram[3], вводя его в виде текста.

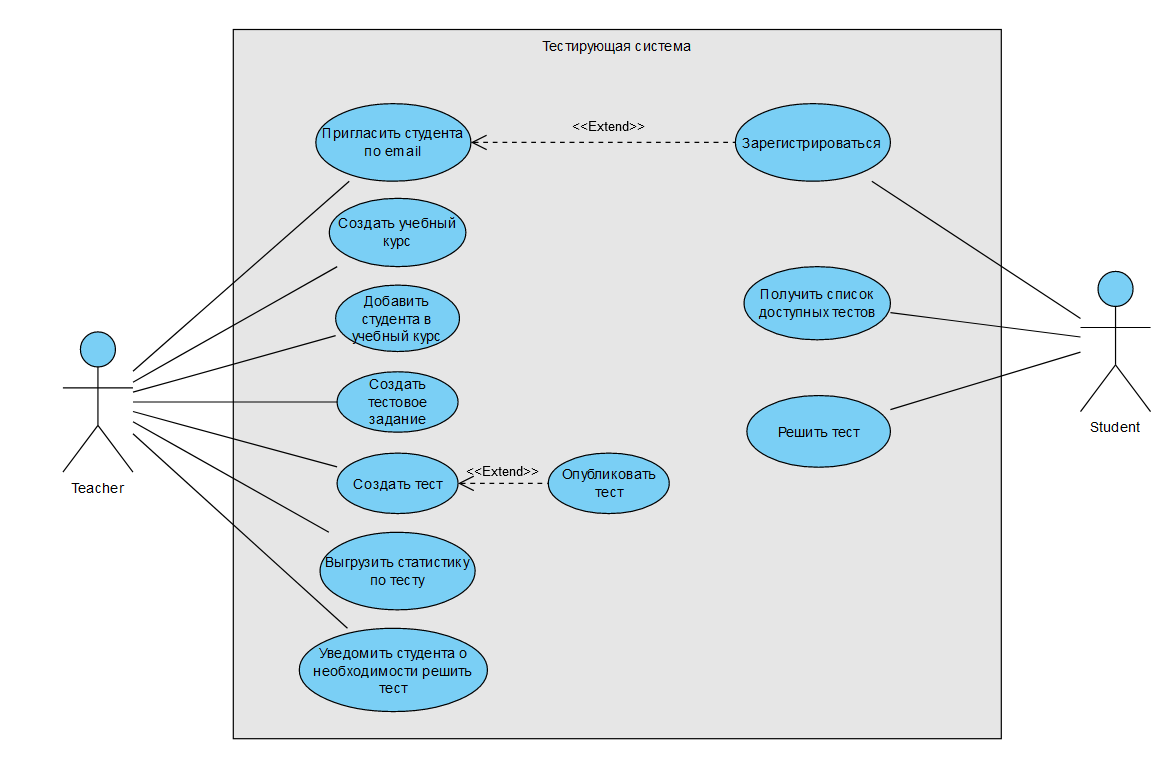


Рисунок 1. Основные пользовательские сценарии

## **Описание применяемых математических методов**

При передаче изображения используется подход кэширования[8]. Создание изображения тестового задания является длительной и дорогостоящей операцией, так как включает в себя множество этапов:

* Обращение к сервису создания изображений
* Запуск облегченной версии веб-браузера
* Обращение к сервису со статичным шаблоном страницы
* Вставка данных в шаблон
* Захват изображения открытой страницы веб-браузера
* Отправка изображения на запрашивающий сервис

Кэширование[8] происходит на главном сервисе **cms** (см. рис. 2) после первого запроса за изображением. При следующих запросах отдается сохраненная версия без запуска процесса создания изображения. Если изменяется часть объекта, которая влияет на финальное изображение, кэш[8] очищается, чтобы при следующем запросе вернуть актуальный ответ.

## **Описание алгоритма и функционирования программы**

## Описание реализации программы

Была выбрана микросервисная архитектура. Ее преимуществами являются гибкость и масштабируемость. Каждый сервис может быть разработан и запущен независимо от других, это позволит изменять сервис без влияния на другие. Также это дает дополнительную свободу выбора технологий внутри каждого отдельного сервиса. Способ передачи данных между микросервисами - протокол HTTP[9].

1. **database.** База данных PostgreSQL, в которой реализована схема, изображенная на рис. 3.
2. **cms.** Серверное приложение, отвечающее за хранение ресурсов. Обеспечивает их создание, чтение, редактирование, удаление в парадигме REST[7]. Инкапсулирует в себе бизнес-логику, доступ к которой предоставляется другим микросервисам при помощи конечных точек. Дополнительно является входной точкой для панели администратора.
3. **admin panel**. Клиентское веб-приложение приложение для управления ресурсами. Реализует CRUD-функционал для управления учебными курсами, списком учеников, заданиями и тестами.
4. **frontend**. Клиентское React-приложение, принимающее на вход объект тестового задания и возвращающее html-страницу с переданным ресурсом. Недоступно для конченого пользователя. Используется другими сервисами внутри границ системы.
5. **webshot**. Сервис, предназначенный для создания снимков веб-страниц. Позволяет получить изображение указанной html-страницы при помощи HTTP[9] запроса. Используется для создания изображений тестовых заданий.
6. **telegram bot**. Дополнительный сервис интеграции с мессенджером telegram[3]. Ученикам дает возможность решать тестовые задания. Преподаватели могут выполнять простые действия по управлению учебными курсами. Например, публикация теста, отправка уведомления ученикам о необходимости решить тестовые задания, выгрузка статистики по тесту.
7. **vkontakte bot**. Дополнительный сервис интеграции с мессенджером ВКонтакте. Позволяет публиковать тестовый задания в виде изображений в открытый доступ на стену группы. Пользователи могут давать ответы на опубликованное задание с целью проверить свои знания. Сервис автоматически проверяет правильность ответа и дает пользователю обратную связь.

Схема взаимодействия изображена на Рисунке 2.

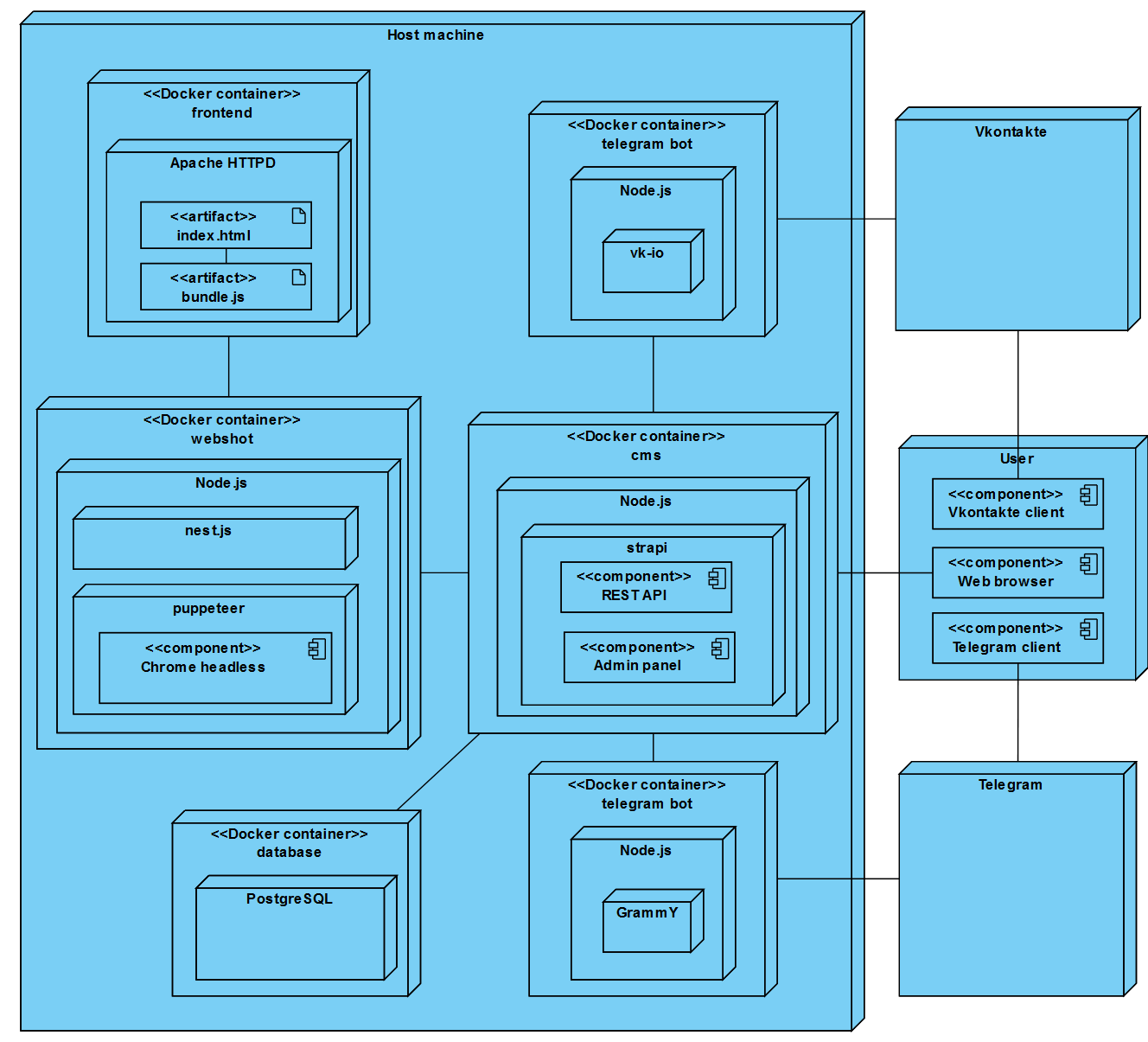


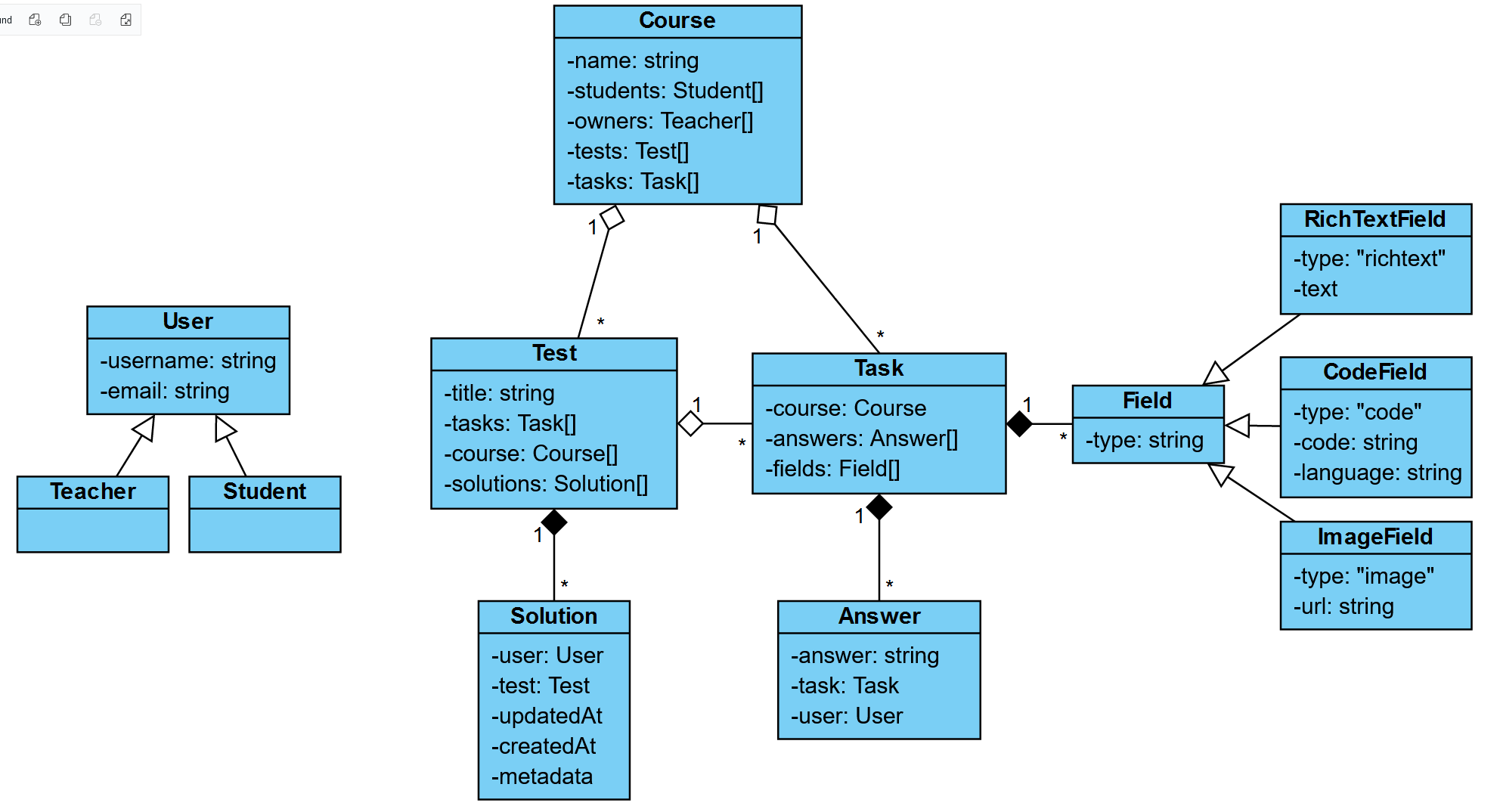
Рисунок 2. Диаграмма развертывания

Рисунок 3. Диаграмма развертывания

## **Описание метода организации входных и выходных данных**

Для организации передачи данных между сервисами в системе используется REST API[7]. Тело запроса и тело ответа на запрос является корректной строкой в формате JSON.

Схема данных, необходимых для работы системы представлена на рис. 3.



## Входные данные

Рисунок 4. Диаграмма классов

Данные, введенные пользователем при работе с панелью администратора, должны соответствовать схеме, изображенной, на рис. 3.  
 Чат-бот[1] самостоятельно сообщает пользователю требования к формату входных данных на каждом этапе взаимодействия. В качестве ответа на задание может быть принята только строка символов.

## Выходные данные

Панель администратора, представляет данные в соответствии со схемой, изображенной, на рис. 3.  
 Чат-бот[1] представляет пользователю тестовые задания в виде изображения, на котором указаны все блоки, входящие в это задание.

## **Описание состава технических и программных средств**

## **Программные средства**

Исходный код системы выполнен с использованием следующих технологий:

* Visual Studio Code – Редактор кода.
* Git – Система контроля версий.
* Docker - Программное обеспечение для контейнеризации и автоматизации приложений. Контейнеры связаны друг с другом в сети, созданной с помощью docker-compose.
* npm - менеджер пакетов и открытый репозиторий пакетов для среды исполнения Node.js.

В качестве основного языка был выбран TypeScript – скриптовый язык программирования со строгой типизацией, который компилируется в JavaScript. Его отличительной особенностью является возможность разработки на нем как клиентских, так и серверных приложений.

Все клиентские приложения выполнены при помощи JavaScript-фреймворка React. Серверные приложения запускаются при помощи среды исполнения Node.js.

Далее описаны программные средства для каждого микросервиса.

## **database**

* PostgreSQL – Реляционная система управления базами данных, используемая для хранения структурированных данных в виде таблиц.

## **backend**

* Strapi – Система управления контентом на базе JavaScript. Позволяет быстро создавать и управлять API.

## **admin panel**

* Strapi – Панель администратора поставляется вместе с системой управления контентом strapi по умолчанию. Реализует базовую CRUD-функциональнось.

## **frontend**

* React
* HTML
* CSS

Исполняемый файл получается в результате компиляции и упаковки кода в один JavaScript-файл, что позволяет запускать программу в браузере.

## **webshot**

* Nest.js - фреймворк для создания масштабируемых серверных приложений на Node.js. Использует принцип Dependency Injection.
* Puppeteer – Движок веб-браузера Chrome, предназначенный для запуска внутри среды исполнения Node.js. Используется для генерации изображений заданий из html-шаблона.

## **vkontakte bot**

* vk-io - Библиотека для работы с API социальной сети ВКонтакте на языке JavaScript/TypeScript.

## **telegram bot**

* GrammY - платформа для разработки telegram-ботов на языке программирования TypeScript. Предоставляет набор инструментов для создания ботов, включая обработку команд, клавиатуры, отправку и получение сообщений. GrammY использует Telegram Bot API, который обеспечивает связь между ботом и серверами Telegram.

## **Технические средства**

Для работы клиентской части системы необходимы следующие компоненты:

* + - Процессор: Intel Pentium 4 / Athlon 64 или более поздней версии с поддержкой SSE2
    - Видеокарта GeForce GTX 470/Radeon R7 260X или лучше.
    - Операционная система: Windows 7 / Windows 8 / Windows 10 / Windows 11 / MacOS X 11 и выше / Linux
    - Свободное место на жёстком диске: 256 МБ.
    - Оперативная память: 4ГБ.
    - Доступ в интернет.
    - Клавиатура.
    - Мышь или заменяющее устройство ввода.

Требования обусловлены необходимостью использования веб-браузера или клиентского приложения telegram[3] для взаимодействия с системой.

Для развертывания серверной части системы необходимы следующие компоненты:

* + - Процессор: Intel Ice Lake или более поздней версии с поддержкой SSE2
    - Операционная система: Linux с предустановленой системой Docker.
    - Объем дискового пространства: 100 Гб.
    - Оперативная память: 4ГБ.
    - Доступ в интернет.
    - Статический IP-адрес

# Ожидаемые технико-экономические показатели

## Предполагаемая потребность

Использование системы автоматической проверки работ имеет множество преимуществ по сравнению с ручной проверкой. Ниже перечислены основные причины, по которым использование такой системы является более эффективным решением:

1. Экономия времени. Автоматическая проверка работ занимает гораздо меньше времени в сравнении с традиционным методом ручной проверки.
2. Объективность. Система автоматической проверки работ учитывает только факты и оценивает результаты на основе определенных критериев. Использование такой системы позволяет дать объективную оценку.
3. Повышение качества обучения. Система автоматической проверки может установить точные ошибки и недостатки в работе ученика, что позволяет лучше понимать, какие темы требуют дополнительного изучения. Таким образом, преподаватели могут построить гибкую систему обучения.

В целом, использование системы автоматической проверки работ учеников - это более эффективный, быстрый и точный способ проверки знаний и улучшения качества обучения.

## Ориентировочная экономическая эффективность

Разрабатываемая система имеет следующие отличительные характеристики:

1. Интеграция с мессенджером telegram. Возможность интеграции с другими сервисами.
2. Без учета затрат на разработку и поддержку системы, использование бесплатно в рамках организации заказчика.
3. Количество созданных тестов - не ограничено. На практике значение не будет превышать 10000.
4. Количество участников в одном тесте – не ограничено. На практике значение не будет превышать 1000.
5. Экспорт результатов в формате csv[4].
6. Тип ответа на задание – открытый ответ. Гибкое решение, позволяющее реализовать другие типы ответа при помощи текста задания.
7. Элемент геймификации в виде приятного оформления шаблонов заданий.
8. Поддержанные сущности в теле вопроса – блок текста с форматированием, блок с кодом, блок с изображением.
9. Дополнительная функциональность будет добавляться по запросу при необходимости.

Система реализует необходимую функциональность и не накладывает ограничения систем-аналогов, которые были рассмотрены в п. 2.2.1 «Обзор аналогов».

Можно сделать вывод о том, что использование системы может привести к существенному повышению экономической эффективности. Ориентировочная экономическая эффективность такой системы может быть рассчитана индивидуально для каждой организации в зависимости от ее потребностей и ресурсов.

# Список используемых источников

1. ГОСТ 19.101-77 Виды программ и программных документов. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.401-78 Текст программы. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.505-79 Руководство оператора. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.

**Приложение 1**

# СПИСОК ТЕРМИНОВ

## Чат-бот

Программа, которая используется для автоматического общения с пользователями через чат-платформы. Используются для автоматизации и оптимизации процессов общения с пользователями и предоставления информации.

## Плагин

Программный модуль, который используется для расширения функциональности другой программы или системы.

## Telegram

Система обмена сообщениями

## *csv*

Текстовый формат данных, представляющий таблицы.

## png

Формат растровой графики, использующий сжатие без потерь.

## xlsx

Формат бинарного файла для хранения и обработки электронных таблиц.

## REST API

архитектурный стиль веб-разработки, который позволяет передавать данные между клиентом и сервером через простой и однозначный API, использующий стандартные HTTP-методы (GET, POST, PUT, DELETE) для доступа к данным и операций с ними.

## Кэширование

Процесс сохранения копии данных в быстродействующей памяти для ускорения доступа к этим данным при последующих обращениях.

## HTTP

П[ротокол](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB) [прикладного уровня](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB%D1%8B_%D0%BF%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D1%8F) для передачи произвольных данных.

**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 |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |