**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа «Программная инженерия»

|  |  |  |  |
| --- | --- | --- | --- |
|  | СОГЛАСОВАНО  Аспирант факультета компьютерных наук департамента программной инженерии, штатный преподаватель факультета компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_ / Н.А. Терлыч /  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 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 ДЛЯ АВТОМАТИЗАЦИИ ПРОВЕРКИ РУКОПИСНЫХ ОТВЕТОВ  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.12.17-01 81 01-1-ЛУ** | | |
|  | | ИСПОЛНИТЕЛЬ  Студент группы БПИ225  \_\_\_\_\_\_\_\_\_ / А.Е. Дадыков /  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2024 г. |
| Москва 2024 | | |

|  |  |
| --- | --- |
| УТВЕРЖДЕН  RU.17701729.12.17-01 81 01-1-ЛУ |  |
| |  |  | | --- | --- | | Подп. и дата |  | | Инв. № дубл. |  | | Взам. инв. № |  | | Подп. и дата |  | | Инв. № подл |  | | СИСТЕМА ДЛЯ АВТОМАТИЗАЦИИ ПРОВЕРКИ РУКОПИСНЫХ ОТВЕТОВ  **Пояснительная записка**  **RU.17701729.12.17-01 81 01-1**  **Листов 19** | |
| Москва 2024 | |

СОДЕРЖАНИЕ

[АНнотация 3](#_Toc162985751)

[1. введение 5](#_Toc162985752)

[1.1 Наименование программы и условное обозначение темы разработки 5](#_Toc162985753)

[1.2 Краткая характеристика области применения 5](#_Toc162985754)

[1.3 Документы, на основании которых ведется разработка 5](#_Toc162985755)

[1.4 Наименование темы разработки 5](#_Toc162985756)

[2. назначение И область применения 6](#_Toc162985757)

[2.1 Функциональное назначение 6](#_Toc162985758)

[2.2 Эксплуатационное назначение 6](#_Toc162985759)

[3. технические характеристики 7](#_Toc162985760)

[3.1 Постановка задачи на разработку программы 7](#_Toc162985761)

[3.1.1 Серверная часть: 7](#_Toc162985762)

[3.1.2 Клиентская часть: 8](#_Toc162985763)

[3.2 Описание алгоритма и функционирования программы 9](#_Toc162985764)

[3.2.1 Архитектура приложения 9](#_Toc162985765)

[3.2.2 Схема базы данных приложения 9](#_Toc162985766)

[3.2.3 Алгоритм хеширования паролей 10](#_Toc162985767)

[3.2.4 Валидация входных данных 10](#_Toc162985768)

[3.3 Описание и обоснование организации входных и выходных данных 11](#_Toc162985769)

[4. ОЖидаемые технико-экономические показатели 12](#_Toc162985770)

[4.1 Ориентировочная экономическая эффективность 12](#_Toc162985771)

[4.2 Предполагаемая потребность 12](#_Toc162985772)

[4.3 Преимущества разработки по сравнению с отечественными и зарубежными образцами и аналогами 13](#_Toc162985773)

[СПИСОК используемой литературы 14](#_Toc162985774)

[Терминология 16](#_Toc162985775)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 19](#_Toc162985776)

АНнотация

Пояснительная записка — это документ, который представляет собой объяснение, пояснение или обоснование различных аспектов проекта, работы или программы. В контексте технической разработки пояснительная записка обычно используется для описания целей проекта, его назначения, области применения, технических характеристик, ожидаемых результатов и других важных аспектов, необходимых для понимания и оценки проекта.

Данная пояснительная записка содержит следующий ряд разделов:

1. «Введение»;
2. «Назначение и область применения»;
3. «Технические характеристики»;
4. «Ожидаемые технико-экономические показатели».

В разделе «Введение» указано наименование и краткая характеристика области применения программы.

В разделе «Назначение и область применения» указан документ, на основании которого ведется разработка, и наименование темы разработки.

В разделе «Назначение разработки» описано эксплуатационное и функциональное назначение программы, а также краткая характеристика области ее применения.

Раздел «Технические характеристики» содержит основные требования к функциональным характеристикам.

Раздел «Ожидаемые технико-экономические показатели» содержит ориентировочную экономическую эффективность, предполагаемую годовую потребность, экономические преимущества разработки программы.

Настоящий документ разработан в соответствии с требованиями:

1. ГОСТ 19.103–77 «Обозначения программ и программных документов» [[1]](#ГОСТ_19_103_77);
2. ГОСТ 19.104–78 «Основные надписи» [[2]](#ГОСТ_19_104_78);
3. ГОСТ 19.105–78 «Общие требования к программным документам» [[3]](#ГОСТ_19_105_78);
4. ГОСТ 19.106-78 «Требования к программным документам, выполненным печатным способом» [[4]](#ГОСТ_19_106_78);
5. ГОСТ 19.201-78 «Техническое задание. Требования к содержанию и оформлению»[[5]](#ГОСТ_19_201_78).
6. введение
   1. Наименование программы и условное обозначение темы разработки

Полное наименование программы на русском языке: «Система для автоматизации проверки рукописных ответов».

Полное наименование программы на английском языке: «System for Automating the Verification of Handwritten Answers».

Условное наименование программы: «ManuScript».

* 1. Краткая характеристика области применения

Программа обеспечивает автоматическую проверку рукописных работ на соответствие заданным ответам. Программа предоставляет функционал создания учебных классов и уроков, позволяя пользователям загружать и проверять рукописные работы в рамках этих уроков.

Основная цель разрабатываемой программы — ускорить процесс проверки работ и облегчить взаимодействие между преподавателями и учащимися.

* 1. Документы, на основании которых ведется разработка

Учебный план подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и утвержденная академическим руководителем тема курсового проекта.

* 1. Наименование темы разработки

Наименование темы разработки на русском языке: «Система для автоматизации проверки рукописных ответов».

Наименование темы разработки на английском языке: «System for Automating the Verification of Handwritten Answers».

1. назначение И область применения
   1. Функциональное назначение

Программа предоставляет возможность автоматической проверки рукописных работ. Пользователи программы делятся на роли: «Преподаватель» или «Учащийся».

«Преподаватель» имеет возможность создавать классы, а затем в классах создавать уроки. В уроках «Учащиеся» имеют возможность прикрепить свою работу в формате фотографии и автоматически получить оценку работы. Также «Преподаватель» способен прикрепить работу за «Учащегося».

Программа имеет браузерную и мобильную версии.

* 1. Эксплуатационное назначение

Программа ускорит процесс проверки рукописных работ и облегчит взаимодействие между преподавателями и учащимися в образовательных учреждениях.

1. технические характеристики
   1. Постановка задачи на разработку программы
      1. Серверная часть:

Серверная часть должна предоставлять API для взаимодействия с пользователем и реализовывать следующий функционал:

* После регистрации либо авторизации пользователи программы делятся на роли: «Преподаватель» или «Учащийся». Каждый пользователь может быть «Преподавателем» и «Учащимся» одновременно;
* «Преподаватель» имеет возможность создавать учебные классы, а затем в классах создавать уроки. В уроках «Учащиеся» имеют возможность прикрепить свою работу в формате фотографии и автоматически получить оценку работы;
* «Преподаватель» имеет возможность добавлять в уроки теорию и задания, где «Учащимся» необходимо прикрепить свою работу;
* «Преподаватель» имеет возможность изменить оценку «Учащегося» за урок, если автоматическая проверка неверно определила оценку;
* «Преподаватель» способен добавить работу за «Учащегося»;
* «Преподаватель» имеет возможность посмотреть успеваемость выбранного класса и ученика;
* Каждый пользователь имеет возможность изменить фамилию и имя, почту, пароль;
* Каждый пользователь имеет возможность выбрать один из доступных языков программы.
  + 1. Клиентская часть:

Клиентская часть должна быть реализована в виде браузерного и мобильного приложений и предоставлять интерфейс для взаимодействия с серверной частью. Должны быть реализованы:

1. **Страница регистрации / авторизации**: на этой странице пользователи могут создать новый аккаунт в системе (регистрация), введя логин, пароль и фамилию с именем, или войти в свой существующий аккаунт (авторизация), введя логин и пароль.
2. **Страница профиля**: эта страница отображает информацию о пользователе: фамилию, имя, электронную почту, изображение профиля. Пользователь может просматривать и редактировать свой профиль на этой странице.
3. **Страница настроек**: на этой пользователь может настраивать параметры своего аккаунта и предпочтения, такие как уведомления и безопасность (смена логина, пароля, фамилии и имени).
4. **Страница учебных классов для учащегося**: на этой странице отображаются все учебные классы, в которых зарегистрирован текущий пользователь в роли учащегося. Он может видеть список доступных классов, присоединяться к новым и управлять уже существующими (закрепить, удалить).
5. **Конструктор учебных классов**: интерфейс, который позволяет преподавателю создавать новые учебные классы. Пользователь может добавлять уроки, учеников, учителей, устанавливать параметры класса и т. д.
6. **Страница учебного класса**: на этой странице пользователь может просматривать информацию о конкретном учебном классе, такую как список участников, материалы, задания и прочее.
7. **Страница урока**: эта страница позволяет просматривать содержание конкретного урока в рамках выбранного учебного класса. Здесь могут быть представлены материалы, задания, домашние задания.
8. **Страница уведомлений**: на этой странице пользователь может просматривать уведомления, связанные с его аккаунтом, учебными классами или уроками, такие как новые сообщения, обновления.

Мобильная и браузерная версии должны иметь одинаковый функционал программы, за исключением измененного размера экрана.

* 1. Описание алгоритма и функционирования программы
     1. Архитектура приложения

Архитектура приложения основана на разделении на серверной и клиентской частях. Серверная часть реализована с использованием технологии fsatapi, в то время как клиентская часть разработана на фреймворке React. Для управления данными и взаимодействия с базой данных применяются паттерны DTO (Data Transfer Object), Repository и Service.

Паттерн DTO используется для передачи данных между компонентами приложения для структурирования данных и их передачи и использования.

Паттерн Repository отвечает за управление доступом к данным в базе данных. Для инкапсуляции логики доступа к данным и простого интерфейс для работы с данными.

Сервисы (Service) представляют собой компоненты приложения, отвечающие за выполнение бизнес-логики и координацию работы различных частей системы для обеспечения высокого уровня абстракции и модульности приложения.

Таким образом, архитектура приложения строится на четком разделении обязанностей между фронтендом и бэкендом, а также на использовании паттернов DTO, Repository и Service для управления данными и бизнес-логикой.

* + 1. Схема базы данных приложения

Используемая база данных PostgreSQL, используемая ORM - SQLAlchemy. Содержит следующие таблицы:

Таблица "classrooms": предназначена для хранения информации о классах или группах пользователей. Каждая запись в этой таблице может содержать данные о названии класса, описании, дате создания и других атрибутах.

Таблица "modules": содержит информацию о модулях обучения или разделах курсов. Здесь могут храниться данные о названии модуля, описании, продолжительности и других характеристиках.

Таблица "lessons": предназначена для хранения информации о конкретных уроках или занятиях в рамках модуля. В этой таблице могут содержаться данные о заголовке урока, содержании, дате и времени проведения и других атрибутах.

Таблица "steps": содержит информацию о шагах или заданиях, которые пользователь должен выполнить в рамках урока или модуля. Здесь могут храниться данные о номере шага, описании, ссылках на материалы и других атрибутах.

Таблица "notifications": предназначена для хранения уведомлений, отправляемых пользователям приложения. Здесь могут храниться данные о тексте уведомления, времени отправки, статусе прочтения и других атрибутах.

Таблица "images": содержит информацию о изображениях, используемых в приложении. В этой таблице хранятся данные о путях к изображениям, их размерах, типах и других характеристиках.

А также вспомогательные таблицы для организации связей между ними.

* + 1. Алгоритм хеширования паролей

Для хранения паролей используется стандартный алгоритм хэширования SHA256.

* + 1. Валидация входных данных

На серверной части осуществляется валидация данных, поступающих от клиента, включая проверку наличия обязательных полей, их формата, а также проверку на предельные и минимальные значения. Применение встроенных или пользовательских валидаторов позволяет гарантировать целостность и правильность данных, прежде чем они будут обработаны или сохранены в базе данных. В случае обнаружения некорректных данных, бэкенд возвращает клиенту соответствующие сообщения об ошибках.

На клиентской части также проводится предварительная валидация данных в интерфейсе пользователя. Это позволяет предотвратить отправку неверных данных на сервер и обеспечить лучший пользовательский опыт, предупреждая пользователя о возможных ошибках непосредственно на этапе ввода. При этом используются механизмы проверки форматов данных, проверка обязательных полей, а также отображение сообщений об ошибках пользователю для коррекции ввода.

Комбинированное применение валидации данных как на сервере, так и на клиенте, позволяет минимизировать возможные ошибки и обеспечить целостность и надежность работы приложения

* 1. Описание и обоснование организации входных и выходных данных

**Входные данные:**

Параметры запроса передаются через URL или в теле HTTP-запроса в формате JSON или форм-данных, в зависимости от типа запроса.

Для обеспечения безопасности данные подвергаются валидации на сервере с использованием проверок формата, допустимых значений и других правил.

Используются стандартные схемы валидации, такие как JSON Schema или библиотеки валидации для обеспечения соответствия входных данных ожидаемым требованиям.

**Выходные данные:**

Данные возвращаются в формате JSON для обеспечения удобства обработки на клиентской стороне.

Организация выходных данных включает в себя структурирование ответов в соответствии с ожидаемой схемой данных, включая необходимые поля и их типы.

В случае ошибок сервер возвращает соответствующие HTTP-статусы и сообщения об ошибке в формате JSON для обеспечения информативности и удобства обработки ошибок на стороне клиента.

1. ОЖидаемые технико-экономические показатели
   1. Ориентировочная экономическая эффективность

В рамках данной работы расчёт экономической эффективности не предусмотрен.

* 1. Предполагаемая потребность

На данный момент не существует прямых аналогов разрабатываемой системы. Наиболее широко для взаимодействия преподавателей и учащихся используется образовательная платформа Stepik. Данная платформа предоставляет возможности создания учебных классов, уроков и оценивания учащихся через стандартизированные тесты. Разрабатываемая система позволит автоматически оценивать учащихся через фотографии их рукописных работ.

* 1. Преимущества разработки по сравнению с отечественными и зарубежными образцами и аналогами

Таблица 2 – Сравнительная таблица аналогов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Функциональность | Google Classroom | Photomath | Stepik | ManuScript |
| Сканирование текста с фотографии | – | + | – | + |
| Автоматическая проверка рукописных работ учащихся | – | – | – | + |
| Автоматическая проверка через стандартизированные тесты | + | – | + | – |
| Возможность создавать классы и уроки | + | – | + | + |

Из сравнительной таблицы аналогов ManuScript выделяется преимуществом в виде автоматической проверки рукописных работ учащихся. Эта функция отсутствует у других аналогов, что делает ManuScript уникальным решением в области образовательных технологий.

СПИСОК используемой литературы

1. ГОСТ 19.103–77 «Обозначения программ и программных документов» // Единая система про­граммной документации. – М.: ИПК Издательство стандартов, 2001;
2. ГОСТ 19.104–78 «Основные надписи» // Единая система про­граммной документации. – М.: ИПК Издательство стандартов, 2001;
3. ГОСТ 19.105–78 «Общие требования к программным документам» // Единая система про­граммной документации. – М.: ИПК Издательство стандартов, 2001;
4. ГОСТ 19.106-78 «Требования к программным документам, выполненным печатным способом» // Единая система про­граммной документации. – М.: ИПК Издательство стандартов, 2001.;
5. ГОСТ 19.201-78 «Техническое задание. Требования к содержанию и оформлению» // Единая система про­граммной документации. – М.: ИПК Издательство стандартов, 2001.;
6. ГОСТ 19.301-79 «Программа и методика испытаний. Требования к содержанию и оформлению» // Единая система про­граммной документации. – М.: ИПК Издательство стандартов, 2001.;
7. ГОСТ 19.401-78 «Текст программы. Требования к содержанию и оформлению» // Единая система про­граммной документации. – М.: ИПК Издательство стандартов, 2001.;
8. ГОСТ 19.404-79 «Пояснительная записка. Требования к содержанию и оформлению» // Единая система про­граммной документации. – М.: ИПК Издательство стандартов, 2001.;
9. ГОСТ 19.505-79 «Руководство оператора. Требования к содержанию и оформлению» // Единая система про­граммной документации. – М.: ИПК Издательство стандартов, 2001.;
10. Python: <https://www.python.org/> (дата обращения: 07.02.2024);
11. Pytesseract: <https://pypi.org/project/pytesseract/> (дата обращения: 07.02.2024);
12. FastAPI: <https://fastapi.tiangolo.com/> (дата обращения: 07.02.2024);
13. JavaScript: <https://ru.wikipedia.org/wiki/JavaScript> (дата обращения: 07.02.2024);
14. React: <https://react.dev/> (дата обращения: 02.04.2024);
15. PostgreSQL: <https://www.postgresql.org/> (дата обращения: 07.02.2024);
16. Stepik: <https://stepik.org/> (дата обращения: 07.02.2024);
17. Google Classroom: <https://edu.google.com/workspace-for-education/classroom/> (дата обращения: 07.02.2024);
18. Photomath: <https://photomath.com/> (дата обращения: 07.02.2024);

Терминология

Таблица 1 – Глоссарий: Используемые термины и определения (алфавитном порядке)

|  |  |
| --- | --- |
| Термин | Определение |
| «Учащийся» | Пользователь программы, который состоит в классе. |
| «Преподаватель» | Пользователь программы, который создает классы и уроки для обучения «Учащихся». |
| Браузер | Программа, которая позволяет просматривать веб-страницы. |
| Класс | Виртуальное пространство, где «Учащиеся» способны просматривать уроки. |
| Клиент | См. термин «Пользователь». |
| Логин | Уникальное имя или идентификатор, который используется пользователем для доступа к программе. |
| Мобильное устройство | Компактные персональные компьютеры с размером диагонали 4-11 дюймов (включая планшетные компьютеры), предназначенные для портативного использования и обладающие возможностью подключения к интернету и камерой. |
| Образовательное учреждение | Учреждение, осуществляющее образовательный процесс, то есть реализующее одну или несколько образовательных программ и (или) обеспечивающее содержание и воспитание обучающихся, воспитанников. |
| Ответ HTTP-запроса (HTTP Response) | Сообщения, которые сервер отправляет в ответ на клиентский HTTP-запрос. |
| Персональный компьютер | Устройства для индивидуального использования, предназначенное для обработки данных, доступа в Интернет, выполнения программ и других задач. |
| Пользователь | Человек, который использует программу по её назначению. |
| Программа | См. термин «Система». |
| Сервер | Компьютер, который обрабатывает запросы от клиентских устройств и предоставляет им доступ к ресурсам и данным. |
| Система | Программное обеспечение, разработанное для автоматизации процесса проверки и оценки рукописных ответов «Учащихся» образовательных учреждений. |
| Урок | Виртуальное пространство, где «Учащиеся» способны смотреть теорию и прикреплять свои рукописные работы в формате фотографий. |
| Фреймворк | Программное обеспечение, которое предоставляет набор инструментов для разработки приложений. |
| API | Набор способов и правил, по которым различные части программы обмениваются между собой данными. |
| FastAPI | Современный фреймворк для создания API на языке Python. |
| Google Classroom | Образовательная платформа для обучения, которая позволяет учителям взаимодействовать с учениками через Интернет. |
| HTTP-запрос (HTTP Request) | Сообщения, которые отправляется клиентом на сервер, чтобы вызвать выполнение некоторых действий. Зачастую для получения доступа к определенному ресурсу. |
| JavaScript | Язык программирования, который в основном используется для разработки клиентской части. |
| JSON | Текстовый формат обмена данными, основанный на JavaScript. |
| Photomath | Мобильное приложение, которое помогает пользователю решать математические задачи, сканируя их с помощью камеры смартфона и предоставляя пошаговые решения. |
| PostgreSQL | Программное обеспечение для сохранения информации о пользователях и классах для корректной работы системы. |
| Pytesseract | Библиотека на языке Python, которая позволяет распознавать текст на изображениях и извлекать его в формате текстовых данных. |
| Python | Язык программирования, использующийся для серверной части. |
| React | Современный фреймворк для разработки браузерных приложений. |
| Stepik | Образовательная платформа для обучения, которая позволяет учителям взаимодействовать с учениками через Интернет. |
| ORM | ORM (Object-Relational Mapping) — это технология, которая связывает объекты в приложении с записями в базе данных, обеспечивая удобный доступ к данным без необходимости писать SQL-запросы напрямую. |
| DTO | Паттерн DTO (Data Transfer Object) — это объект, который используется для передачи данных между приложением и базой данных или между различными слоями приложения, обеспечивая структурированный и удобный для обработки формат данных. |
| Repository | Паттерн Repository — это шаблон проектирования, который предоставляет абстракцию над доступом к данным в базе данных, скрывая детали реализации работы с хранилищем данных, и предоставляя удобные методы для выполнения операций чтения, записи, обновления и удаления данных. |
| Service | Паттерн Service — это компонент приложения, который содержит бизнес-логику и обеспечивает выполнение операций над данными или функциональность приложения. Сервисы используются для разделения ответственности и повышения модульности приложения. |

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

| Лист регистрации изменений | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Номера листов (страниц) | | | | | Всего листов (страниц) в докум. | № документа | Входящий № сопрово-дительного докум. и дата | Подп. | Дата |
| Изм. | изменен- ных | заменен- ных | новых | аннули-рованных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |