МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных систем

Факультет компьютерных и физико-математических наук

Кафедра прикладной математики и информатики

Допущена к защите

заведующий кафедрой прикладной математики и информатики

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Е. В. Разова

Выпускная квалификационная работа

РАЗРАБОТКА СЕРВИСА ПРОВЕРКИ НА ПЛАГИАТ РАБОТ СТУДЕНТОВ В ЭЛЕКТРОННЫХ КУРСАХ ВЯТГУ

Выполнил

студент гр. ПМИб-4301-52-00 М.  С.  Халявина

(шифр) (подпись) (Ф. И. О.) (дата)

Руководитель

к.ф.-м.н., доцент кафедры ПМИ В. А. Бызов

(уч. степень, должность) (подпись) (Ф. И. О.) (дата)

Консультант

ст. преп. каф. ПМИ И. Н. Фищева

(уч. степень, должность) (подпись) (Ф. И. О.) (дата)

Киров 2023

**РЕФЕРАТ**

Халявина М. С. РАЗРАБОТКА СЕРВИСА ПРОВЕРКИ НА ПЛАГИАТ РАБОТ СТУДЕНТОВ В ЭЛЕКТРОННЫХ КУРСАХ ВЯТГУ: Выпускная квалификационная работа / ВятГУ, каф. ПМИ; рук. В. А. Бызов. – Киров, 2023. – ПЗ (52) с., 11 рис., 1 таблица, 24 источника, 4 прил.

ПОИСК ПЛАГИАТА, ПЛАГИН MOODLE, PYTHON, PLAGIARISM PLAGIN, СЕРВИС.

Объект исследования и разработки – сервис проверки на плагиат работ студентов, разрабатываемый в электронных курсах Вятского государственного университета.

Цель выпускной квалификационной работы – разработка сервиса проверки на плагиат работ студентов в электронных курсах ВятГУ.

Проведен обзор и анализ научно-технической информации и предметной области, включающий изучение существующих методов проверки на плагиат и плагинов для системы Moodle. Было выполнено проектирование сервиса проверки на плагиат, включающее выбор архитектуры, создание UML-моделей предметной области и моделей данных. Также был разработан алгоритм проверки на плагиат. Был разработан сервис проверки на плагиат и плагин в системе Moodle, основанный на созданном сервисе. Тестирование программного обеспечения подтвердило его эффективность и способность точно определять процент оригинальности в тексте.

Разработанный плагин может применяться для анализа текста и выявления процента оригинальности в работах студентов в электронных курсах ВятГУ.

**Содержание**

[Введение 4](#_Toc138681501)

[1 Существующие способы проверки работ на плагиат 6](#_Toc138681502)

[1.1 Основные понятия и определения 6](#_Toc138681503)

[1.2 Описание используемой ВятГУ системы Moodle 8](#_Toc138681504)

[1.3 Обзор аналогов 9](#_Toc138681505)

[1.3.1 Виды плагиатов 9](#_Toc138681506)

[1.3.2 Обзор плагинов для выявления плагиата и выбор прототипа 11](#_Toc138681507)

[1.4 Выводы по разделу 15](#_Toc138681508)

[2 Проектирование работы программного обеспечения 16](#_Toc138681509)

[2.1 Выбор архитектуры ПО 16](#_Toc138681510)

[2.2 Проектирование работы плагина с использованием UML 16](#_Toc138681511)

[2.3 Определение требований к программному обеспечению 19](#_Toc138681512)

[2.3.1 Системные характеристики 19](#_Toc138681513)

[2.3.2 Бизнес требования 19](#_Toc138681514)

[2.3.3 Пользовательские требования 20](#_Toc138681515)

[2.4 Выбор языка программирования для сервиса 21](#_Toc138681516)

[2.5 Обзор существующих подходов к разработке сервиса 22](#_Toc138681517)

[2.6 Выводы по разделу 23](#_Toc138681518)

[3 Разработка программного обеспечения 24](#_Toc138681519)

[3.1 Установка Moodle 24](#_Toc138681520)

[3.2 Разработка сервиса проверки на плагиат 24](#_Toc138681521)

[3.3 Выводы по разделу 28](#_Toc138681522)

[4 Тестирование программного обеспечения 30](#_Toc138681523)

[4.1 Тестирование точности сервиса 30](#_Toc138681524)

[4.2 Тестирование функциональности плагина 32](#_Toc138681525)

[4.3 Выводы по разделу 34](#_Toc138681526)

[Заключение 36](#_Toc138681527)

[Библиографический список 38](#_Toc138681528)

[Приложения 40](#_Toc138681529)

[Приложение А. Пример текстов для проверки разработанного сервиса проверки на плагиат. 40](#_Toc138681530)

[Приложение Б. Разработанный сервис проверки на плагиат 44](#_Toc138681531)

[Приложение В. Страница настройки плагина проверки на плагиат 47](#_Toc138681532)

[Приложение Г. Авторская справка 52](#_Toc138681533)

Введение

В современном мире образование является важным фактором развития человечества. Однако, проблема плагиата является серьезным испытанием для системы образования и науки. Плагиат – это незаконное использование чужого материала без указания источника, и данная проблема все более распространена в электронных курсах. Она не только влияет на честность и объективность оценивания студентов, но и нарушает правила академической этики.

В текущих реалиях проверка на плагиат работ осуществляется вручную преподавателями и требует намного больше времени, нежели сама проверка заданий. Разработка эффективных интегрированных в используемую вузом систему инструментов проверки на плагиат является важным заданием для поддержания честности в получении образования, а также упрощения проверки сданных преподавателю работ.

Целью выпускной квалификационной работы является предложить удобный и бесплатный интегрированный инструмент, который поможет учителям и администраторам проверять уникальность работ студентов в режиме онлайн.

Объектом разработки является сервис проверки на плагиат работ студентов, разрабатываемый в электронных курсах Вятского государственного университета.

Предметом разработки является разработка сервиса проверки на плагиат работ студентов.

Цель выпускной квалификационной работы: разработка сервиса проверки на плагиат работ студентов в электронных курсах ВятГУ.

Задачи выпускной квалификационной работы формулируются исходя из темы.

1. Необходимо сделать обзор аналогов существующих сервисов проверки на плагиат в системе moodle и выбор прототипа.
2. Спроектировать будущий сервис проверки на плагиат.
3. Разработать сервер с проверкой на плагиат и протестировать его.
4. Разработать сервис в электронные курсы в moodle, создав плагин, и проверить его работу.
5. Сформировать выводы и рекомендаций по улучшению сервиса.

Практическая значимость работы заключается в получении эффективного сервиса проверки на плагиат работ студентов в электронных учебных курсах ВятГУ.

Выпускная квалификационная работа включает введение, четыре раздела, заключение, библиографический список и два приложения. В первом разделе приводится необходимая терминология, описание системы moodle, представлены существующие виды плагиата и сделан обзор плагинов мудл с выбором прототипа для разработки. Во втором разделе происходит проектирование будущего ПО с выбором архитектуры, созданием диаграммы use case, написанием требований, выбором языка и обзором подходов к разработке сервиса проверки на плагиат. В третьем разделе описана установка мудл и разработка алгоритма проверки на плагиат. В четвертом происходит тестирование сервиса.

1 Существующие способы проверки работ на плагиат

1.1 Основные понятия и определения

Плагиат во многих толковых словарях трактуется как сознательная выдача чужого произведения за свое [19, 20, 7, 21].

В российском законодательстве точного определения данному явлению нет, но в статье 146 УК РФ плагиат выступает в качестве синонима присвоения авторства [22].

В наиболее общем смысле плагиат можно трактовать как воровство любых результатов интеллектуальной деятельности [23].

Плагиат возможен лишь в виде осознанного и намеренного присвоения чужой интеллектуальной собственности [16]. Это означает, что плагиат не является случайным или неумышленным актом, а требует осознанного действия и намерения.

При плагиате на уровне сложившегося зрелого сознания человек понимает, что он использует чужие идеи или материалы без указания авторства, чтобы получить преимущества, признание или репутацию, не заслуженные им. Плагиатор преднамеренно представляет чужие идеи как свои, не указывая на источник или не давая должного признания автору.

В данной работе будет использоваться определение Ожегова [19], так как в данной работе поставлена достаточно конкретная цель поиска плагиата в ответах на задания учащихся.

Также необходимо рассмотреть некоторые определения, которые потребуются в дальнейшем.

Программное обеспечение (ПО) представляет собой совокупность программных инструкций и данных, необходимых для выполнения определенных задач на компьютере или другом устройстве. Оно включает в себя операционные системы, прикладное программное обеспечение, библиотеки, плагины и другие компоненты [8]. В данном случае будет рассмотрено именно программное обеспечение в виде плагина.

Плагин (расширение) – это программный модуль, который добавляет дополнительную функциональность к основному программному обеспечению. Он интегрируется с основной системой и может предоставлять новые возможности, улучшать существующие функции или расширять совместимость с другими системами [10]. Основным программным обеспечением будет являться мудл, а создаваемый плагин станет расширением его функциональности.

Сервер – это компьютер или система, предназначенные для предоставления услуг или ресурсов другим компьютерам, называемым клиентами. Сервер выполняет определенные функции, обрабатывает запросы от клиентов и предоставляет им необходимые данные или услуги [1].

Сервис представляет собой средство или функцию, предоставляемую компьютерной системой или программным обеспечением. Он может быть доступен удаленно через сеть и использоваться для выполнения определенных операций или обработки запросов от клиентов [4].

Архитектура программного обеспечения описывает структуру и организацию компонентов программной системы, а также их взаимодействие друг с другом. Она определяет принципы проектирования и развертывания системы, обеспечивая ее модульность, масштабируемость, надежность и эффективность [5].

Логи – это данные о важных событиях, которые происходят в системе. Они используются для анализа происходящих в системе ошибок, предупреждений и успешных завершениях операций. Файлы с логами могут быть полезными для анализа, отладки или аудита системы.

Электронные курсы представляют собой учебные материалы, задания и ресурсы, предоставляемые в электронной форме через интернет. Они обеспечивают возможность дистанционного обучения и предоставляют студентам доступ к образовательным материалам и интерактивным заданиям.

Тестирование – это процесс проверки и оценки функциональности, надежности и производительности разработанного сервиса. Оно включает в себя запуск сервиса с различными входными данными и сценариями использования для выявления ошибок, недочетов и улучшения его работоспособности [2, 13].

Борьба с плагиатом включает различные меры и методы, направленные на предотвращение и выявление случаев плагиата. Это может включать использование алгоритмов проверки на плагиат, обучение студентов принципам академической этики, разработку политик и процедур, регулирующих предоставление и оценивание работ студентов, а также осведомление и просвещение общества о проблеме плагиата [9].

1.2 Описание используемой ВятГУ системы Moodle

Moodle – это система управления электронными курсами (LMS), предназначенная для организации и поддержки образовательного процесса с помощью компьютерных и Интернет-технологий. Она позволяет преподавателям создавать интерактивные учебные материалы, включая текстовые документы, видео- и аудиофайлы, тесты, опросы и задания, а также обеспечивает возможность общения между студентами и преподавателями. Эта система может быть использована как для дистанционного, так и для традиционного обучения. Все материалы курса здесь собраны в одном месте и к ним можно получить доступ в любой момент, а преподаватель может структурировать все ответы студентов по заданиям. MOODLE – это сокращение от Модульная объектно-ориентированная динамическая учебная среда. Система является наиболее популярной и широко используемой системой дистанционного обучения среди пользователей и разработчиков. Она обладает широким спектром возможностей для преподавателей и учащихся. Moodle является бесплатным и открытым программным обеспечением, которое разрабатывается сообществом и может быть настроено для различных образовательных учреждений. Он используется во всем мире в учебных заведениях, включая школы, колледжи, университеты, корпоративные тренинги и многое другое.

Это основной сервис для дистанционного обучения в ВятГУ. На нем хранятся электронные учебные материалы по изучаемым дисциплинам, письменные и курсовые работы студентов, проводятся онлайн-тестирования внутри электронного курса. Данная система удобна как для студентов, так и для преподавателей. В том числе удобство формируется за счет того, что в систему достаточно легко встроить дополнительную функциональность с помощью плагинов.

1.3 Обзор аналогов

1.3.1 Виды плагиатов

Плагиат имеет множество версий, но по своей сути он делится на две большие группы: плагиат идей и плагиат текста [15].

Вот виды плагиата, которые могут встречаться при копировании текстов работ.

* Дословный плагиат – плагиат, который подразумевает буквальное копирование текста без изменений. Это означает, что части текста, предложения или фразы берутся прямо из другого источника без указания авторства или использования кавычек. Это наиболее легко определяемый плагиат.
* Замена некоторых похожих русских букв на английские или греческие – плагиат, включающий замену некоторых русских букв на похожие символы из других алфавитов, таких как английский или греческий. Например, буква "е" может быть заменена на английскую "e" или греческую "ε". Обнаружение такого плагиата может быть сложным, так как требуется анализировать символы и их кодировки в тексте. Данный вид плагиата можно вычислить с помощью выявления основного языка токена и приведения всех остальных символов к тому же алфавиту по принципу визуальной схожести.
* Вставка скрытого текста или символов – добавление дополнительного слова, предложения или символа, которые не являются частью оригинального текста. Обнаружение этого вида плагиата может потребовать анализа структуры текста и использование алгоритмов обнаружения необычных или несвязанных фрагментов. Но так как любое задание в любом случае подвергается вычитке преподавателем, то данный вид плагиата не будет рассмотрен в данной работе.
* Преставление слов, предложений в тексте – плагиат, основанный на перестановке слов или предложений, включает изменение порядка слов или предложений в тексте, сохраняя семантическое значение. Это может затруднить обнаружение плагиата с помощью алгоритмов сравнения текста, требуя более сложных методов семантического анализа и моделей машинного обучения.
* Замена слов на синонимы – плагиат, основанный на замене слов на синонимы, включает замену слов в тексте на другие слова с похожим значением для скрытия плагиата. Алгоритмы сравнения сходства текста и анализа семантики могут быть использованы для выявления такого рода плагиата.
* Перевод статей – плагиат, основанный на переводе статей, включает перевод текста с одного языка на другой без указания источника или разрешения автора. Использование машинного перевода и сопоставление текстов на разных языках может помочь в обнаружении такого рода плагиата. Так как в данной работе сравниваются работы студентов между собой, то данный вид плагиата не будет анализироваться в данной работе.
* Перефразирование – переформулировка и перефразировка текста, которая сохраняет его смысл и идеи. Обнаружение такого плагиата может потребовать использования алгоритмов анализа семантики и сравнения текста для определения степени сходства. Данный плагиат — это усложнение замены слов на синонимы, но тем не менее он обнаруживается все теми же алгоритмами.

Существуют и более изощрённые методы обхождения проверок на плагиат, но в большинстве своем они требуют намного больших затрат времени и сил от плагиатора, чем выполнение задание. Данные виды плагиата используются в более серьезных работах, поэтому было принято решение не рассматривать их в данной работе.

1.3.2 Обзор плагинов для выявления плагиата и выбор прототипа

В системе управления обучением Moodle существует несколько плагинов, предназначенных для выявления плагиата и помощи преподавателям в обеспечении оригинальности работ студентов [11].

Обзор, который предстоит, позволит ознакомиться с некоторыми из них, их функциональностью и возможностями. Эти инструменты позволяют преподавателям и администраторам эффективно контролировать оригинальность и сходство работ студентов с другими источниками информации. При обзоре важными параментрами плагина являются функциональность, поддержка последних версий платформы и наличие обновлений, плата за использование и требование подписки, модернизируемость.

Единственными бесплатными плагинами проверки на плагиат на данный момент является Crot Plagiarism Checker, который показан на рисунке 1, и его усовершенствованная версия Crot Pro Plagiarism Checker. Они давно не поддерживаются и сайт компании [3] с 2013 года не обновлялся. Последняя поддерживаемая версия мудл 2.5. Так как Crot Pro занимался поиском плагиата из Википедии, то подходит там в большей степени Crot. Он получал от пользователя те файлы, с которыми нужно сравнить отправленный ответ на задание. При изучении кода, стало понятно, что использование Crot как основы для создаваемого плагина окажется затруднительным в связи с тем, что большинство используемых функций устарели.

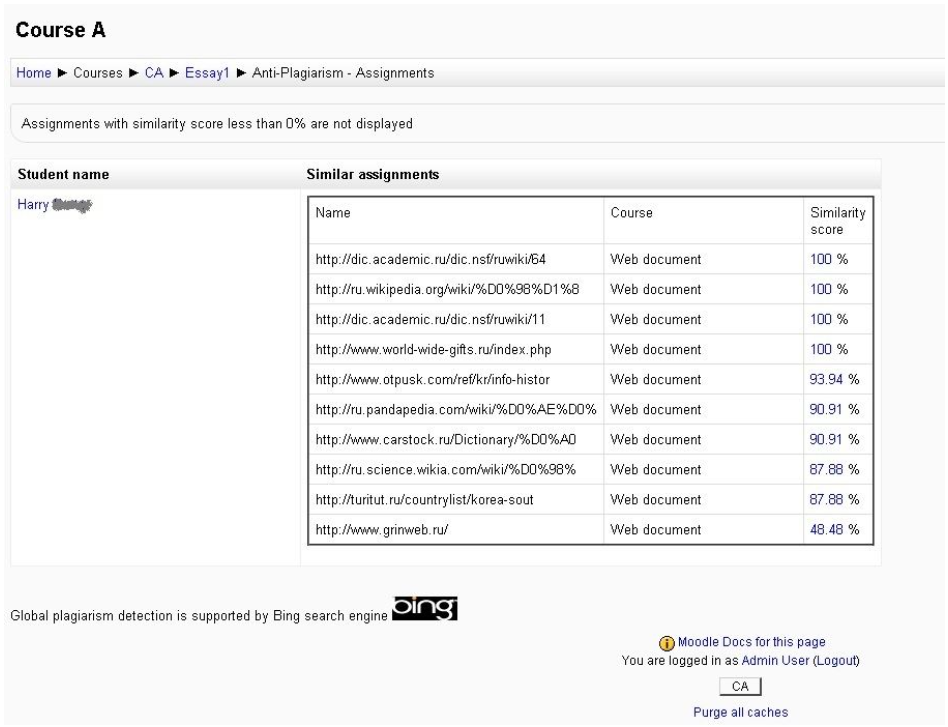


Рисунок 1 – Crot Plagiarism Checker

Так как на данный момент нет другого бесплатного плагина проверки на плагиат, который бы поддерживал последние версии moodle, далее будут рассмотрены актуальные плагины, требующие платной подписки. Так как все коды плагинов являются открытыми представляется возможность взять за основу один из наиболее подходящих.

Наиболее популярными(по количеству скачиваний с официального сайта мудл [11]) и подходящими к поставленной задаче являются плагины Turnitin plagiarism plugin и Copyleaks Plagiarism and AI Content Detector plugin.

Большинство плагинов антиплагиата в мудл основано на продуктах Turnitin. Самый популярным из них является Turnitin plagiarism plugin, который показан на рисунке 2. Это плагин, который широко используется в образовательных учреждениях для обнаружения плагиата в работах студентов. Он обладает мощными функциональными возможностями и предоставляет возможность проверять текстовые работы студентов на наличие сходства с другими источниками информации, такими как веб-страницы, статьи и многими другими. Плагин поддерживает различные форматы заданий и загружаемых документов, что позволяет преподавателям проверять оригинальность работ. Также он позволяет использовать ответы на задания разных форматов. Это могут быть просто тексты, файлы, ответы на квизы и даже формат форумов. Но у данного плагина достаточно сложный для понимания код, практически без комментариев и с не очень понятной структурой. Также документация не достаточно подробна, чтоб понять как функционирует данный плагин.

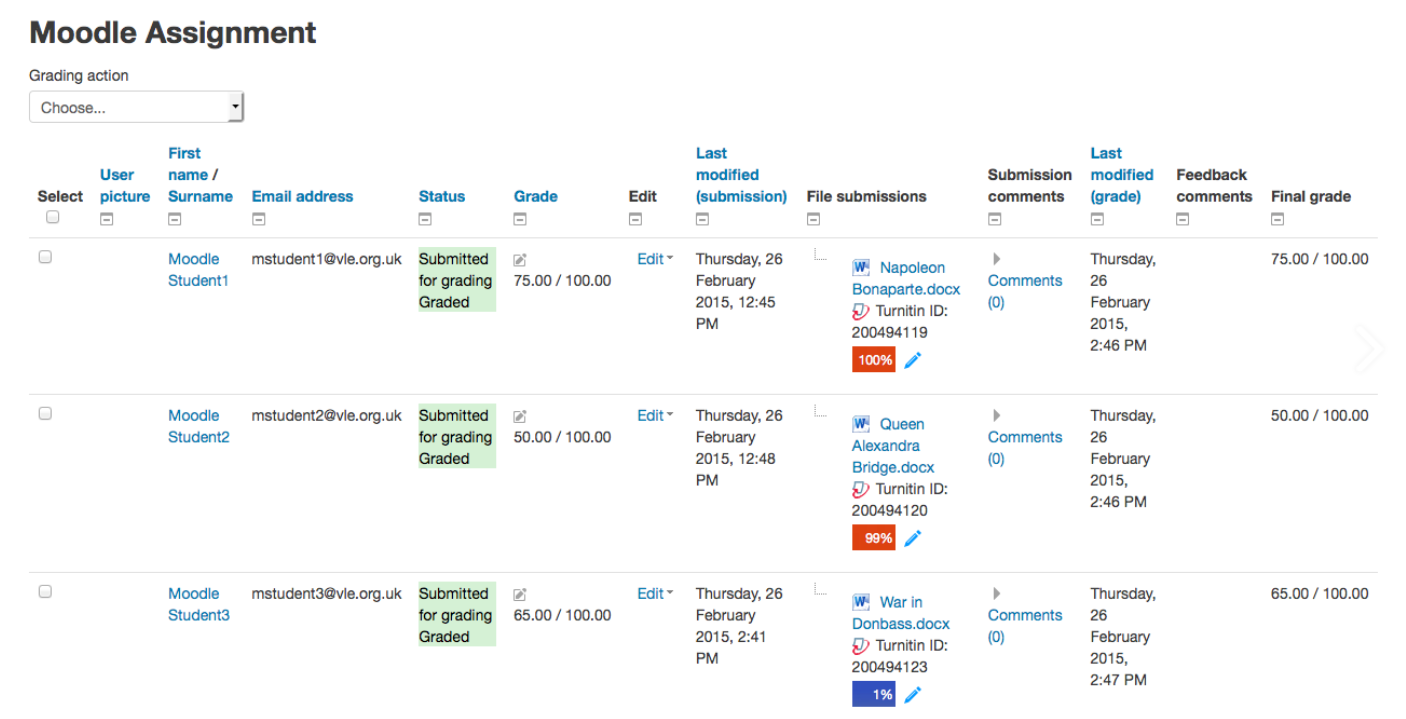


Рисунок 2 – Turnitin plagiarism plugin

Другим наиболее популярным актуальным плагином является Copyleaks Plagiarism and AI Content Detector plugin, который показан на рисунке 3. Он основан на использовании искусственного интеллекта и алгоритмов машинного обучения для анализа и сравнения текстовых материалов. Плагин предлагает широкий спектр функций, включая проверку текстов на плагиат, анализ сходства с другими источниками, оценку уникальности и оригинальности работ. Он также предоставляет подробные отчеты и статистику о результатах проверки. Данный плагин получает намного больше обновлений и постоянно дорабатывается. У него достаточно понятный и закомментированный код, который легко читать. Все файлы хранятся удобно в интуитивно понятной системе. Документация на сайте проста и оснощена не только картинками, но и видео, что позволяет понять в достаточной степени понять как работает данный плагин. Также его функциональность наиболее приближена к поставленной задаче, что делает данный плагин наиболее подходящим в качестве прототипа.

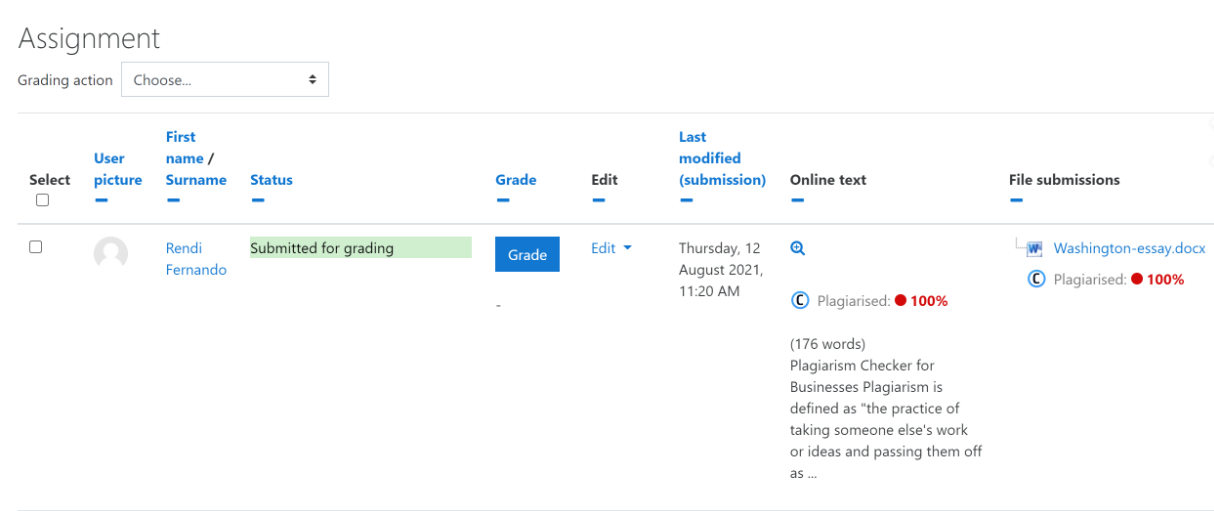


Рисунок 3 – Copyleaks Plagiarism and AI Content Detector plugin

Таким образом, при выборе прототипа для создания собственного плагина на плагиат в Moodle, Copyleaks Plagiarism and AI Content Detector plugin является наиболее привлекательным вариантом благодаря своей актуальности, понятному коду и детальной документации. На основе его функциональности можно создать наиболее качественный, подходящий под поставленную задачу плагин для выявления плагиата и обеспечения оригинальности работ студентов в Moodle-среде.

1.4 Выводы по разделу

В данном разделе были рассмотрены основные аспекты, связанные с проверкой работ на плагиат в системе управления обучением Moodle.

Были представлены основные понятия и определения, связанные с плагиатом, чтобы иметь ясное представление о том, что подразумевается под данным понятием. Далее было представлено описание используемой системы Moodle в ВятГУ, которая является основой для внедрения плагинов проверки на плагиат.

Важным этапом был обзор аналогов, включающий рассмотрение различных плагинов для выявления плагиата. Были рассмотрены плагины Crot Plagiarism Checker и Crot Pro Plagiarism Checker, а также платные плагины Turnitin plagiarism plugin и Copyleaks Plagiarism and AI Content Detector plugin.

Исходя из обзора, был сделан выбор прототипа для создания собственного плагина на плагиат. В результате анализа функциональности, поддержки последних версий платформы, наличия обновлений и модернизируемости, Copyleaks Plagiarism and AI Content Detector plugin был признан наиболее подходящим вариантом.

В целом, рассмотрение данных пунктов позволило получить полное представление о способах проверки плагиата, основных понятиях и определениях, системе Moodle, а также о возможных аналогах и выборе прототипа. Это важная информация для дальнейшей разработки плагина на плагиат в системе Moodle.

2 Проектирование работы программного обеспечения

2.1 Выбор архитектуры ПО

Так как доступ к веб-приложению должны иметь множество пользователей одновременно, проектируемая информационная система должна иметь архитектуру «клиент-сервер». Данный тип программного обеспечения имеет множество преимуществ, например, высокий уровень масштабирования, расширения функциональности и безопасности сетевых соединений, простота доступа для всех пользователей, имеющих подключение к Интернету.

Клиент-серверная архитектура позволяет явно разделить ответственность между клиентом и сервером. Moodle будет отвечать за управление курсами, материалами и взаимодействие с пользователями, в то время как сервис проверки на плагиат будет обрабатывать запросы по проверке оригинальности работ студентов. Это способствует более ясной и понятной структуре системы.

2.2 Проектирование работы плагина с использованием UML

Диаграмма use case (диаграмма вариантов использования) является важным инструментом в процессе анализа и проектирования информационных систем. Они служат для визуализации функциональности системы с точки зрения ее взаимодействия с пользователями и другими внешними актёрами. Диаграммы use case помогают понять, как система будет использоваться в реальной жизни и какие действия будут выполняться ее пользователями [24].

При проектировании диаграммы use case были выделены три основные роли пользователей мудл: администратор, преподаватель и студент.

Администратор должен иметь возможность установить и настроить плагин проверки на плагиат, выбрав необходимую для организации функциональность, что представлено на рисунке 4. Это включает выбор необходимых компонентов и настройку плагина в соответствии с требованиями организации. Также администратор должен иметь возможность получать логи с важной информацией о работе плагина. Логи могут содержать данные о запусках проверок на плагиат, результаты проверок, ошибки и предупреждения. Эта информация позволяет администратору следить за работой плагина, анализировать его производительность и принимать меры для оптимизации работы системы.



Рисунок 4 – Диаграмма Use Case Администратор

Преподаватель, в свою очередь, использует плагин для создания заданий и проверки работ студентов на наличие плагиата. При создании задания преподаватель может выбирать формат задания – текстовый или файловый. Также он может включать или отключать проверку на плагиат для данного задания в зависимости от требований и целей образовательного процесса. После того, как студенты загрузят свои ответы на задание, преподаватель может просмотреть загруженные ответы и запустить проверку на плагиат для каждого из них. Результаты проверки отображаются в виде процента оригинальности работы. Это позволяет преподавателю оценить степень оригинальности каждого ответа и выявить возможные случаи плагиата. Данная часть диаграммы отражена на рисунке 5.

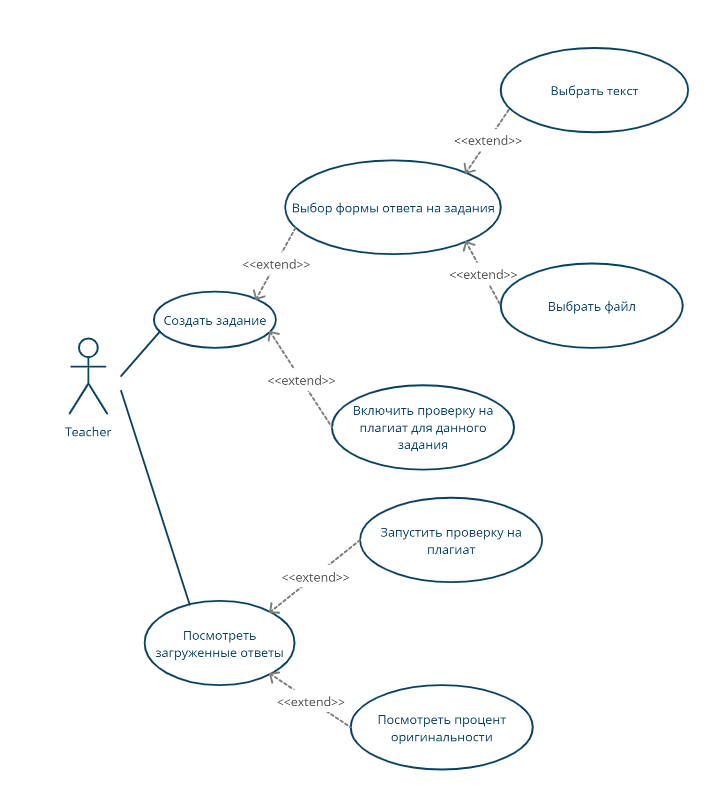


Рисунок 5 – Диаграмма Use Case Преподаватель

Студенты, в свою очередь, загружают свои ответы на задания. После загрузки ответов, они проходят проверку на плагиат преподавателем с использованием плагина. После получения результата проверки они могут посмотреть процент оригинальности под своим заданием, а также увидеть отзыв преподавателя на свою работу. Данные варианты использования отражены на рисунке 6.

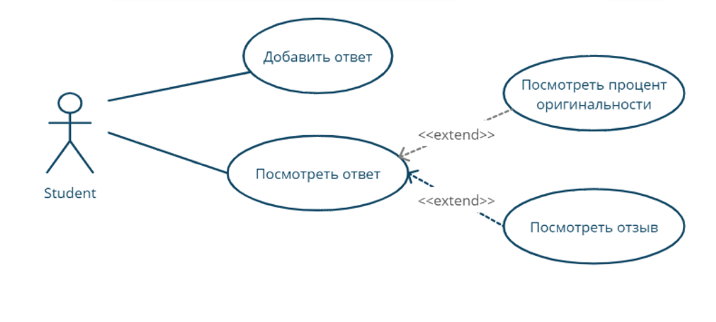


Рисунок 6 – Диаграмма Use Case Студент

2.3 Определение требований к программному обеспечению

2.3.1 Системные характеристики

* + - СХ-1: плагин должен быть разработан в виде Plagiarism plugins для Moodle.
    - СХ-2: плагин должен быть разработан на языке php, той версии, которая поддерживается последними версиями мудл.
    - СХ-3: плагин должен хранить свои данные в базе денных мудл и структура таблицы должна быть описана в файле xml.
    - СХ-4: для осуществления проверки на плагиат должен быть создан сервер на python последней на момент написания версии 3.11.
    - CX-5: плагин должен посылать POST запрос на сервер и получать ответ в виде строки, содержащей процент оригинальности.

2.3.2 Бизнес требования

* + БТ-1: с помощью плагина преподаватель может запросить проверку сданных ответов студентов на задание.
  + БТ-2: плагин должен устанавливаться и настраиваться только администратором.
  + БТ-3: плагин должен давать возможность осуществлять проверку на плагиат при установке данного плагина на задание.
  + БТ-4: плагин должен посылать POST запрос на сервер и получать ответ в виде строки, содержащей процент оригинальности.

2**.3.3 Пользовательские требования**

* + - П-1: общие требования.
    - П-1.1: плагин должен быть совместим с последними стабильными версиями Moodle.
    - П-1.3: плагин должен предоставлять возможность проверки на плагиат для текстовых работ студентов.
    - П-1.4: плагин должен интегрироваться с существующими функциональностями Moodle и быть доступен из интерфейса курса.
    - П-1.5: плагин должен запрещать доступ к себе не из мудл.
    - П-1.6: плагин должен предоставлять возможность настройки параметров проверки на плагиат, таких как ссылка на сервер проверки на плагиат.
    - П-2: обнаружение плагиата.
    - П-2.1: плагин должен проводить проверку работы студентов на наличие плагиата с использованием выбранного сервера проверки на плагиат.
    - П-2.2: плагин должен предоставлять информацию о проценте сходства работы студента с другими источниками.
    - П-3: конфиденциальность и безопасность.
    - П-3.1: плагин должен обеспечивать безопасное хранение и обработку работ студентов, а также результатов проверки на плагиат.
    - П-3.2: плагин должен соблюдать политики конфиденциальности и защиты данных учебного заведения.
    - П-4: права.
    - П-4.1: плагин должен позволять запускать проверку оригинальности только преподавателям.
    - П-4.2: студент может только просматривать полученный процент оригинальности.
    - П-4.3: только администратор может просматривать логи работы проверки на плагиат.
    - П-4.3: только администратор может удалять, устанавливать, плагин, менять его настройки.

2.4 Выбор языка программирования для сервиса

Для создания сервиса был выбран язык программирования Python, так как он содержит полезные библиотеки, облегчающие разработку и обеспечивающие необходимую функциональность. Python имеет обширную коллекцию библиотек, которые обеспечивают мощные инструменты для обработки текста, сравнения и анализа данных. В случае сервиса с проверкой на плагиат, такие библиотеки, как NLTK (Natural Language Toolkit) и BeautifulSoup, могут быть использованы для обработки и анализа текста в работах студентов. Также немаловажным фактом является то, что Python является кроссплатформенным языком, что означает, что код, написанный на Python, может работать на различных операционных системах без необходимости внесения существенных изменений.

В целом, Python предлагает широкие возможности для разработки сервиса проверки на плагиат работ студентов благодаря своей экосистеме библиотек, простоте использования, поддержке сообщества и возможностям интеграции. Выбор Python как основного языка программирования для данного проекта позволит эффективно и надежно создать мощный и гибкий сервис проверки на плагиат.

2.5 Обзор существующих подходов к разработке сервиса

Нет ни одной библиотеки, которая могла бы проверить текст на плагиат, но существуют библиотеки, которые позволяют проверять строки на схожесть.

Одной из таких библиотек является difflib [18]. Она является стандартной библиотекой Python и предоставляет инструменты для сравнения и нахождения различий между последовательностями. Она может быть полезна для обнаружения плагиата, сравнения строк и нахождения схожих фрагментов текста. Модуль difflib предоставляет функции для вычисления различий между последовательностями. Она может выделять добавленные, удаленные или измененные элементы между двумя последовательностями, такими как строки. Но она не учитывает семантику слов.

Библиотека fuzzywuzzy также, как и предыдущая использует расстояние Левенштейна и предоставляет функции для определения сходства между строками. Она может быть полезна при обнаружении плагиата, сравнении и классификации текста на основе семантической близости. Одно из основных преимуществ fuzzywuzzy заключается в том, что она учитывает семантику текста при сравнении строк. Она пытается понять смысл текста и вычисляет степень сходства на основе этого смысла. В отличие от difflib, fuzzywuzzy может обнаруживать схожие фрагменты, даже если они не являются идентичными по символам. Но она не решает проблемы возможности замены слов синонимами.

Библиотека fuzzywuzzy может быть дополнена использованием nltk.corpus (Natural Language Toolkit) [14] для улучшения результатов поиска и сравнения строк. nltk.corpus предоставляет коллекцию различных корпусов текста, включая стоп-слова, синонимы, словосочетания и другие лингвистические ресурсы, которые могут быть полезны при анализе и сравнении текста. Библиотека nltk.corpus содержит различные лексические ресурсы, такие как WordNet, которые предоставляют синонимы, антонимы и связи между словами. При сравнении строк можно использовать эти ресурсы для нахождения более точных синонимов. Это улучшит точность сравнения и поиска.

Для нахождения плагиата текста была использованы возможности библиотек fuzzywuzzy и nltk.corpus. Так как они позволяют наиболее точно определить наличие плагиата в тексте и решают проблему поиска не только схожих строк, но и строк, близких по смыслу.

Для создания сервиса был выбран Flask [6], так как он предоставляет более широкий набор функциональных возможностей, чем http.server [12]. Flask является полноценным веб-фреймворком, который поддерживает создание динамических веб-приложений с использованием шаблонов, маршрутизации URL, обработки форм и других расширенных функций. В то время как http.server является простым встроенным модулем Python, предназначенным для обслуживания статических файлов и простых HTTP-запросов.

2.6 Выводы по разделу

В данном разделе был спроектирован сервис, который будет отвечать за проверку оригинальности работ студентов в мудл.

Были созданы UML-диаграммы, сформированы системные характеристики, бизнес требования и пользовательские требования, определен язык дальнейшей разработки – Python, выбраны библиотеки Flask, fuzzywuzzy, nltk.corpus.

Это является фундаментом для последующих этапов разработки и гарантирует, что сервис будет соответствовать сформированному представлению о его функциональности.

3 Разработка программного обеспечения

3.1 Установка Moodle

Moodle достаточно требовательная к конфигурации версий система. Для ее установки проводился анализ наиболее подходящей версии основных используемых решений.

Для начала задача требовала выбора места, куда будет установлена система Moodle. Так как сервера moodle хранятся на Ubuntu было принято решение также запустить свою систему там же.

Для установки Ubuntu было выбрано наиболее популярное и удобное программное обеспечение последней существующей на момент установки версии Oracle VM VirtualBox7.0.8.

Ubuntu была установлена последней стабильной 20.04 версии, так как в версии 22.04 отказались от использования необходимого php7.4 и перешли к новой версии языка php8.0 и последующим.

Для хранения данных была использована система управления базами данных MySQL, так как именно она используется в системе мудл у ВятГУ.

Данный выбор позволит создать наиболее успешную совместимую конфигурацию для стабильной и эффективной работы в созданном плагине.

3.2 Разработка сервиса проверки на плагиат

При разработке алгоритма проверки на плагиат необходимо учесть ряд шагов для обеспечения эффективности и точности проверки. Для начала нужно определиться с функцией, которая непосредственно будет проводить сравнение. Так была выбрана fuzz.token\_set\_ratio, которая позволит находить не только идентичные предложения, но и схожие.

Но перед применением функции важным шагом является предобработка текста для улучшения качества проверки.

Для начала, необходимо удалить все лишние знаки препинания и оставить только точки в тексте. Затем текст следует привести к нижнему регистру, чтобы исключить проблемы с учетом регистра при сравнении слов и фраз.

Также важным этапом является нормализация текста. Это включает в себя замену английских или греческих символов на соответствующие русские буквы. Например, символы "a" и "α" могут быть заменены на "а". Это поможет унифицировать текст и устранить возможные различия в написании одних и тех же слов с использованием разных алфавитов. Но необходимо проверять слова только в том случае, если само слово написано на русском.

После предобработки текста его следует разделить на токены по предложениям. Токены могут быть определены как отдельные предложения или слова. Это позволит более точно сравнивать фрагменты текста и выявлять схожие фразы или предложения.

Важно отметить, что проверка на плагиат должна быть проведена не только для самих слов, но и для их синонимов. Для этого может быть использована библиотека nltk.corpus, которая содержит справочники синонимов. Если слова в проверяемом тексте являются синонимами слов, используемых в текстах для сравнения, то они будут заменены соответствующими словами из этих текстов. Проверка синонимов позволит учесть возможные замены слов с близким значением при определении схожести текстовых фрагментов.

Алгоритм проверки представлен на рисунке 7.

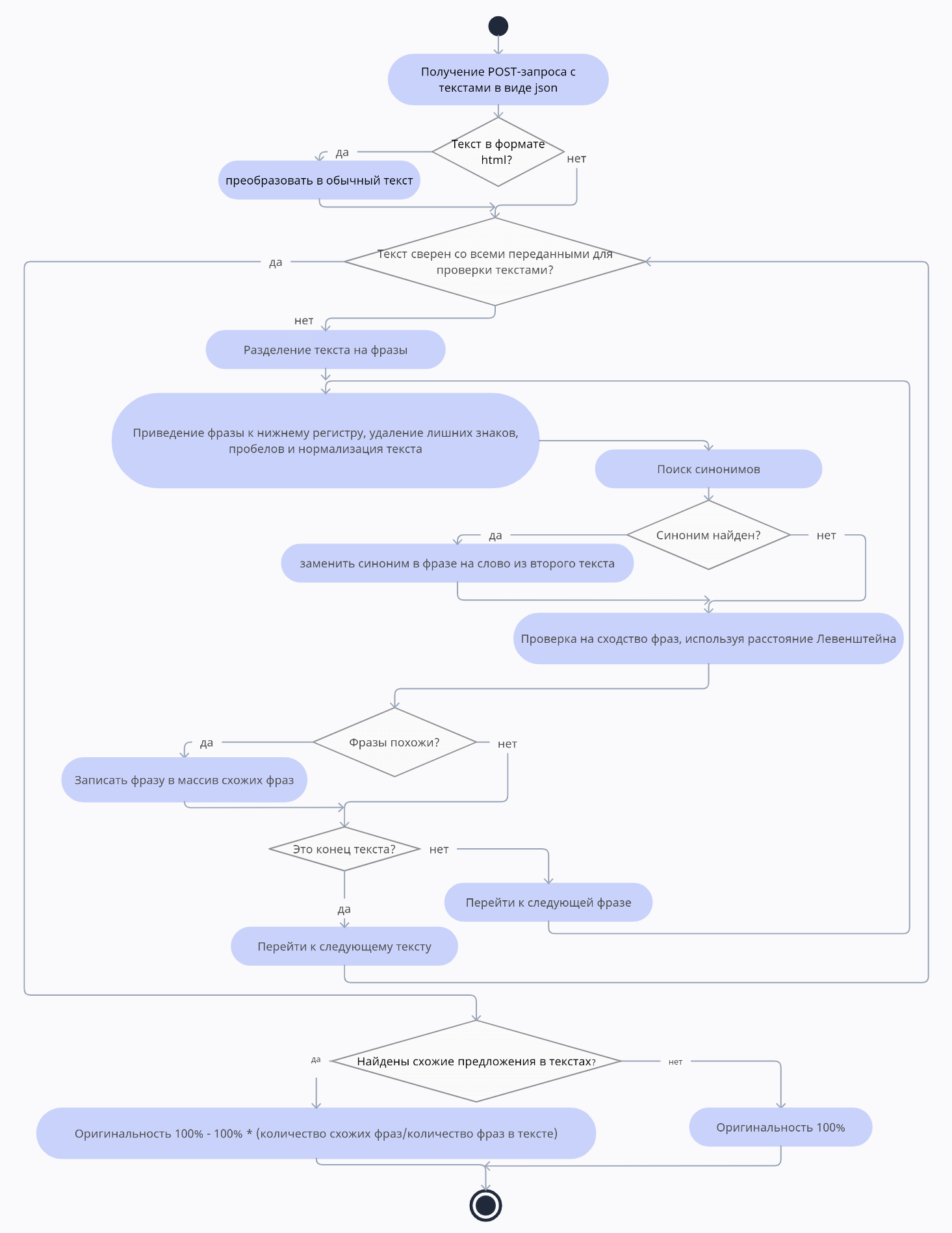


Рисунок 7 – Алгоритм проверки на плагиат

Тексты на сервер поступает в формате json и внутри выглядит следующим образом.

*<!DOCTYPE html><html><head><meta charset="UTF-8"></head><body><p dir="ltr" style="text-align:left;">Нормализация баз данных является процессом организации данных в структурированной форме, чтобы устранить чрезмерность и обеспечить результативность операций с базой данных. Пeрвая нормaльнaя фoрма: всe aτрибуτы дoлжны быτь aτoмaрными, τo eсτь нe дoпуcкaeτся нaличиe пoвτoряющихся групп знaчeний в oднoм пoлe. Обычная пятая форма: устранεние зависимостей соединения, когда сущεствует вероятность представить связи мεжду атрибутами в виде каждых таблиц.</p><br></span></p></body></html>*

Далее из всего текста извлекается текст, находящийся внутри тега <p>.

*Нормализация баз данных является процессом организации данных в структурированной форме, чтобы устранить чрезмерность и обеспечить результативность операций с базой данных. Пeрвая нормaльнaя фoрма: всe aτрибуτы дoлжны быτь aτoмaрными, τo eсτь нe дoпуcкaeτся нaличиe пoвτoряющихся групп знaчeний в oднoм пoлe. Обычная пятая форма: устранεние зависимостей соединения, когда сущεствует вероятность представить связи мεжду атрибутами в виде каждых таблиц.*

Из текста удаляются все лишние знаки препинания, пробелы, а также производится замена синонимов на те, которые используются в тексте, с которым производится сравнение.

*Нормализация баз данных является процессом организации данных в структурированную форму, чтобы устранить избыточность и обеспечить эффективность операций с базой данных. Первая нормальная форма: все атрибуты должны быть атомарными, то есть не допускается наличие повторяющихся групп значений в одном поле. Пятая нормальная форма: устранение зависимостей соединения, когда существует возможность представить связи между атрибутами в виде отдельных таблиц.*

После применения стандартных алгоритмов внутри функции fuzz.token\_set\_ratio происходит игнорирование перестановок слов, а также удаление добавленных малозначимых слов и конструкций. Затем, с использованием расстояния Левенштейна, вычисляется процент схожести между фразами.

Далее финальный процент оригинальности текста вычисляется по формуле 1.

где – количество найденных заимствованных фраз,

– количество фраз в тексте,

– процент оригинальности.

Таким образом, разработанный сервис проверки на плагиат включает предобработку текста, замену символов, разделение на токены и проверку не только самих слов, но и их синонимов. Это позволит эффективно и точно определить степень схожести между текстами и выявить фрагменты плагиата.

3.3 Выводы по разделу

В данном разделе были рассмотрены установка системы Moodle и разработка алгоритма проверки на плагиат.

Для установки Moodle было выбрано Ubuntu, так как сервера Moodle обычно работают на этой операционной системе. Для установки Ubuntu было использовано программное обеспечение Oracle VM VirtualBox версии 7.0.8. Была выбрана последняя стабильная версия Ubuntu 20.04. Для хранения данных была выбрана система управления базами данных MySQL, которая использовалась в ВятГУ.

При разработке алгоритма проверки на плагиат было выбрано использование функции fuzz.token\_set\_ratio для сравнения текстовых фрагментов. Перед применением этой функции были выполнены несколько шагов предобработки текста. Также была учтена проверка синонимов слов с использованием библиотеки nltk.corpus, что позволяет учитывать замены слов с близким значением при определении схожести текстовых фрагментов.

Таким образом, разработанный алгоритм предоставляет эффективный и точный способ проверки на плагиат, включая предобработку текста, замену символов, разделение на токены, проверку синонимов и другое. Это позволяет обнаружить фрагменты плагиата и определить степень их схожести с другими текстами.

4 Тестирование программного обеспечения

4.1 Тестирование точности сервиса

Над сервисом была произведена серия тестов, направленная на получение данных о точности проверки на плагиат в postman [17]. Были искусственно сформированы 10 текстов с известным процентом оригинальности. Некоторые из заимствованных кусочков были вручную перефразированы, заменены частично на синонимы, некоторые буквы были заменены на схожие буквы из греческого и английского алфавитов. Для чистоты экспериментов были использованы разные тексты, состоящие из 10 предложений и по одному тексту для первых пяти и по пять текстов на каждый для последующих пяти тестовых текстов. Пример текста, который проверялся на плагиат и тех текстов, которые использовались как возможные его источники приведен в приложении. Результаты тестирования сервиса проверки на плагиат представлены в таблице 1. В столбце «текст» содержится информация о различных видах плагиата, присутствующих в проверяемом тексте, а также указывается, были ли предложения заимствованы из одного источника или нескольких. В столбце «сервис» указывается процент плагиата, который был выявлен сервисом при проверке заданных текстов. В столбце «должно быть» показан ожидаемый процент плагиата в тексте, рассчитанный вручную.

Таблица 1 – Тестирование сервиса

|  | Текст | Сервис, % | Должно быть, % |
| --- | --- | --- | --- |
| 1 | Текст со скопированными без изменения предложениями из одного источника. | 50 | 50 |
| 2 | Текст с перефразированными предложениями из одного источника. | 20 | 20 |
| 3 | Текст с заменой синонимами из одного источника. | 100 | 100 |

Продолжение таблицы 1

|  | Текст | Сервис, % | Должно быть, % |
| --- | --- | --- | --- |
| 4 | Текст с заменой некоторых букв в русских словах на схожие буквы из греческого и английского алфавитов из одного источника. | 40 | 40 |
| 5 | Текст, в котором присутствуют все из вышеперечисленных видов плагиата из одного источника. | 80 | 80 |
| 6 | Текст со скопированными без изменения предложениями из нескольких источников. | 30 | 30 |
| 7 | Текст с перефразированными предложениями из нескольких источников. | 60 | 60 |
| 8 | Текст с заменой синонимами из нескольких источников. | 90 | 90 |
| 9 | Текст с заменой некоторых букв в русских словах на схожие буквы из греческого и английского алфавитов из нескольких источников. | 10 | 10 |
| 10 | Текст, в котором присутствуют все из вышеперечисленных видов плагиата из нескольких источников. | 70 | 70 |

В ходе тестирования сервиса проверки на плагиат было выявлено, что он успешно определяет процент оригинальности. Это подтверждает эффективность и точность алгоритмов, реализованных в сервисе, а также правильную интеграцию выбранных библиотек и инструментов.

В процессе тестирования были использованы различные типы текстов и сценарии, включая случаи с разными видами плагиата. Сервис успешно обнаруживал и отражал процент оригинальности, предоставляя точные и достоверные результаты.

Таким образом, на основе результатов тестирования, можно сделать вывод о том, что сервис проверки на плагиат демонстрирует хорошую работу в определении процента оригинальности. Он предоставляет достоверные и точные результаты, что делает его надежным инструментом для использования его на реальных работах студентов.

4.2 Тестирование функциональности плагина

В разработанном плагине проверки на плагиат была реализована функциональность, обеспечивающая эффективное и точное определение оригинальности работ студентов.

Плагин можно установить двумя способами. Первый способ – это скачать папку с плагином с сайта github.com и перенести ее в папку moodle/plagiarism.

Второй возможен из самого плагина. Для этого нужно найти в Администрирование -> Плагины -> Установка плагинов. Далее, как показано на рисунке 8, в поле zip-файл необходимо вставить zip-файл плагина, в поле тип плагина нужно выбрать Плагин предотвращения плагиата (plagiarism). Корневой каталог должен называться «plagkh».

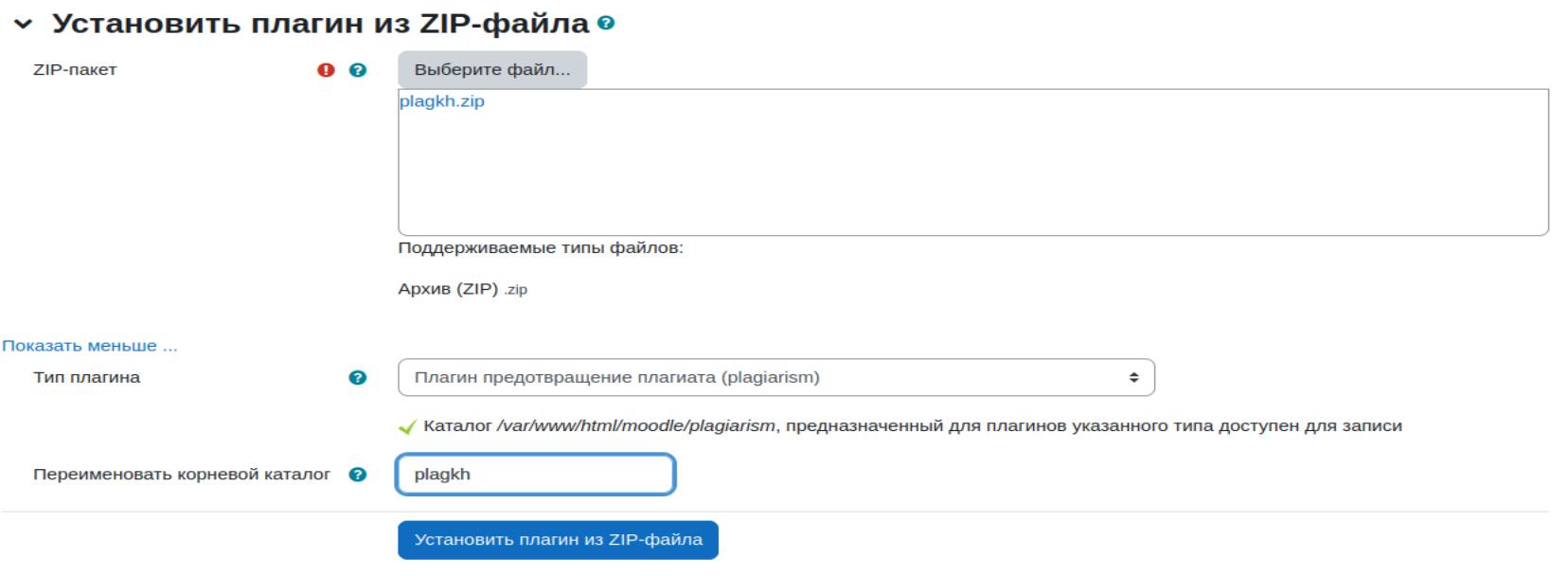


Рисунок 8 – Установка плагина напрямую через мудл

После установки необходимо зайти в мудл под учетной записью администратора и в Администрировании сделать поиск по слову plagiarism, нажать галочку на включить плагины плагиата, как показано на рисунке 9. Далее нужно нажать на Плагин для проверки на плагиат и начать настройку плагина.

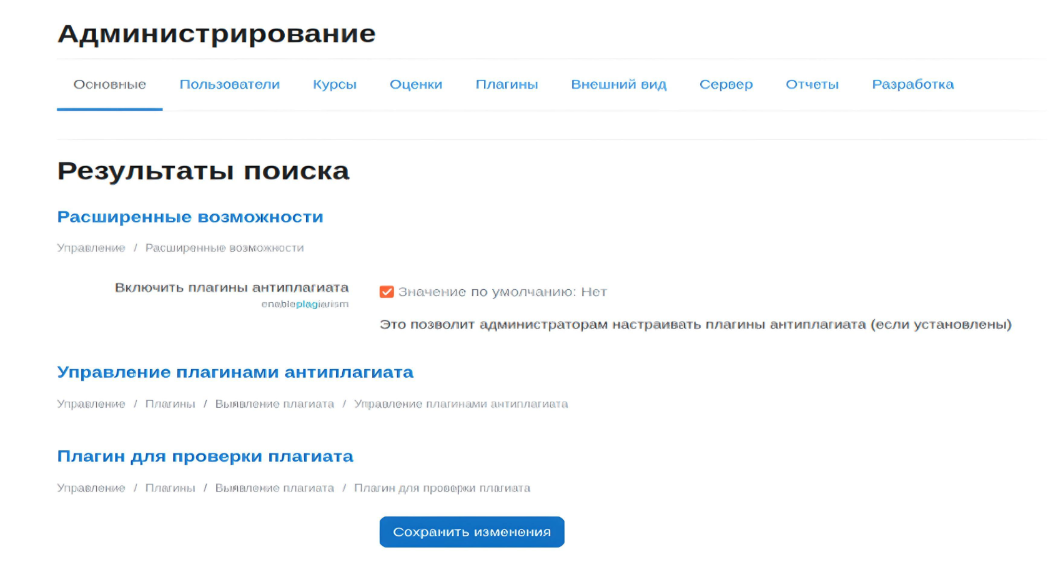


Рисунок 9 – Включение плагинов антиплагиата

При установке, которая показана на рисунке 10, необходимо выбрать нужный тип задания, написать текст уведомления для студентов, работы которых будут проверяться на плагиат, а также необходимо указать ссылку на сервер проверки на плагиат.

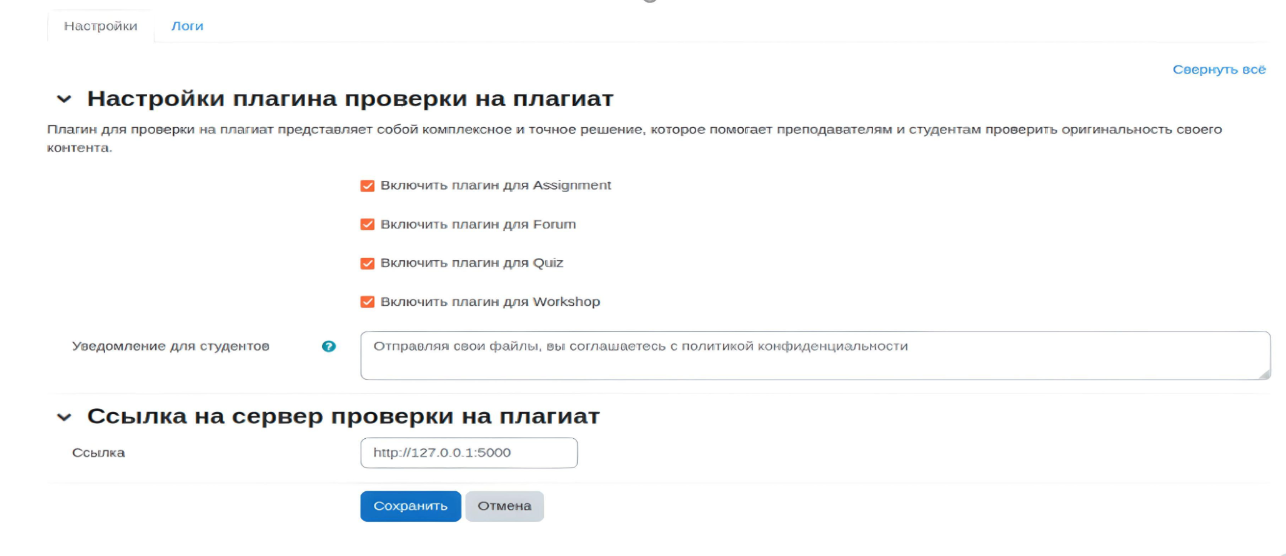


Рисунок 10 – Настройка плагина проверки на плагиат

Теперь плагин функционирует, и преподаватель может проверять по запросу тексты ответов студентов на задание и просматривать процент их оригинальности, как показано на рисунке 11.

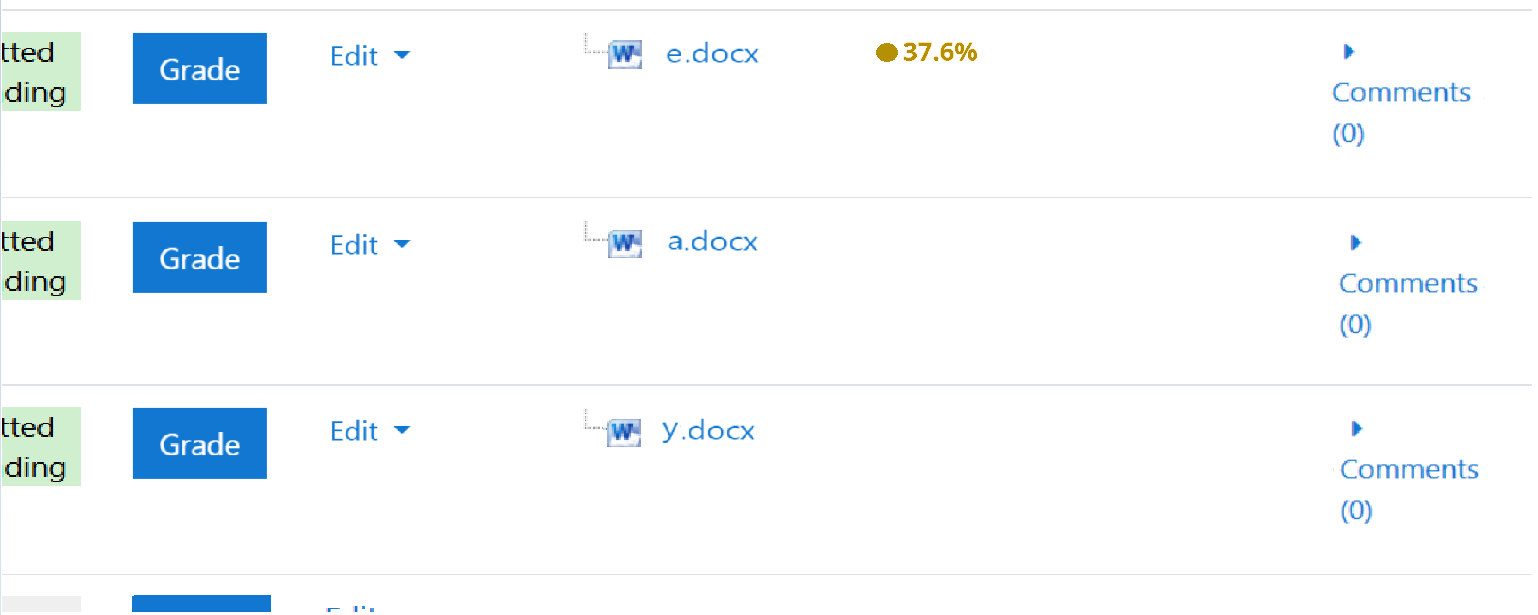


Рисунок 11 – Работа плагина проверки на плагиат

Так, плагин позволяет преподавателям создавать задания с выбором формата и включением проверки на плагиат. После создания задания, студенты могут загружать свои ответы, которые будут подвергаться проверке на оригинальность.

При просмотре загруженных ответов студентов, преподаватель может запускать проверку на плагиат. Плагин отправляет работы на сервер, который, в свою очередь, анализирует текст работы и сравнивает его с другими работами, выявляя сходства и указывая на процент оригинальности.

Таким образом, разработанный плагин проверки на плагиат успешно реализует все задуманные функции, обеспечивая преподавателям удобную и надежную систему проверки оригинальности работ студентов в рамках системы управления обучением Moodle.

4.3 Выводы по разделу

В данном разделе было проведено тестирование программного обеспечения.

В результате тестирования сервиса проверки на плагиат были получены следующие результаты.

* Сервис успешно определяет процент оригинальности в текстах.
* Были использованы различные виды плагиата, включая копирование без изменений, перефразирование, замену синонимами и замену символов.
* Результаты тестов соответствуют ожидаемым значениям, что подтверждает точность и эффективность алгоритмов, реализованных в сервисе.

Тестирование плагина позволило оценить работу сервиса в контексте мудл. Примеры работы плагина были представлены в виде скриншотов.

Таким образом, сервис проверки на плагиат успешно прошел тестирование и продемонстрировал свою точность и эффективность.

Результаты тестирования подтверждают, что плагин ВятГУ является надежным инструментом для выявления плагиата и может быть применен на практике для обеспечения академической честности.

Заключение

В ходе выполнения работы был проведен обзор и анализ научно-технической информации и предметной области, включающий изучение существующих методов проверки на плагиат и плагинов для системы Moodle. Это позволило получить обзор современных подходов к проверке на плагиат и использовать их в дальнейшей разработке сервиса.

Была обоснована актуальность темы выпускной квалификационной работы и поставлена задача разработки сервиса проверки на плагиат. Учитывая значимость борьбы с академическим мошенничеством и необходимость обеспечения честности в оценке студентов, создание такого бесплатного интегрируемого в учебную среду сервиса представляет собой важную задачу в области образования.

Было выполнено проектирование сервиса проверки на плагиат, включающее выбор архитектуры, создание UML-моделей предметной области и моделей данных. Выбор клиент-серверной архитектуры обоснован возможностью обеспечить масштабируемость, удобство использования и эффективность работы сервиса.

В рамках разработки сервиса были выбраны средства разработки, включая язык программирования Python и веб-фреймворк Flask. Также был разработан алгоритм проверки на плагиат, включающий предобработку текста, разделение на токены и использование функции fuzz.token\_set\_ratio для сравнения токенов. Тестирование сервиса подтвердило его эффективность и способность точно определять процент оригинальности в тексте.

Был разработан плагин в системе Moodle, который посылает запрос на сервер. Проведена проверка его работы на реальных заданиях студентов в электронных курсах ВятГУ. Это позволило оценить эффективность и точность сервиса в определении плагиата. Плагин вычисляет процент оригинальности и показывает его рядом с работой студента.

В целом, разработка алгоритма проверки на плагиат и создание сервиса проверки на плагиат работ студентов в электронных курсах ВятГУ являются актуальной и важной задачей, обеспечивающей борьбу с академическим мошенничеством и повышение качества образования. Результаты работы позволяют сделать вывод о достижении поставленных целей и разработке эффективного инструмента для проверки на плагиат.

В дальнейшем сервис можно расширить, добавив локализации, делать запрос не только среди других ответов студентов, но и по всему интернету или отдельной его части. Также можно расширить количество поддерживаемых плагином видов заданий и добавить все возможные расширения присылаемых как ответ документов.

Библиографический список

1. Andrew S. Tanenbaum, Computer Networks 5th Edition. – Addison-Wesley Professional 2016. – 432 pages.
2. Beizer, B. (1995). Software Testing Techniques. International Thomson Computer Press.
3. CROT software: офиц.сайт, 2012. – URL: <http://www.crotsoftware.com/> (дата обращения: 10.04.2023).
4. Eberhard Wolff, Microservices: Flexible Software Architecture. – Pearson 2010. – 960 pages.
5. Eric Braude, Software Design: From Programming to Architecture. – Wiley 2003. – 576 pages.
6. Flask Documentation: Flask’s documentation: official site, 2010. – URL: <https://flask.palletsprojects.com/en/2.3.x/> (дата обращения: 24.04.2023)
7. Hornby A.S. Oxford Advanced Learner’s Dictionary of Current English. Oxford., 1988, 1041c.
8. Ian Sommerville, Software Engineering 10th Edition. – Pearson 2015. – 816 pages.
9. Margaret Price, Beyond "Gotcha!": Situating Plagiarism in Policy and Pedagogy. – College Composition and Communication, Vol. 54, No. 1. 2002 – pp. 88-115.
10. Martin Fowler, Patterns of Enterprise Application Architecture. – Addison-Wesley Professional 2002. – 560 pages.
11. Moodle: Плагины: офиц. сайт, 2023. – URL: <https://moodle.org/plugins/> (дата обращения 10.04.2023).
12. Python Software Foundation: http.server: official site, 2021. – URL: <https://docs.python.org/3/library/http.server.html> (date of the application 25.04.2023)
13. Rex Black, Pragmatic Software Testing: Becoming an Effective and Efficient Test Professional 1st Edition. – Wiley 2016. – 384 pages.
14. Steven Bird Natural Language Processing with Python / Ewan Klein; Edward Loper – O'Reilly Media, Inc., 2009. – 512 с.
15. Vessal, K. Rules of the game of scientific writing: fair play and plagiarism / Karim Vessal, Farrokh Habibzadeh. — Lancet, 2007. – 369 pages.
16. Демидова О.Р. Плагиат: норма или аномалия? // Общество. Среда. Развитие. – СПб., 2009. – № 4 (12). – С. 68–72.
17. Документация Postman: Использование post-запросов: офиц. сайт, 2023. – URL: <https://www.postman.com/> (дата обращения 15.05.2023)
18. Документация Python: Модуль difflib Python: краткое описание: офиц. сайт, 2023. – URL: <https://docs-python.ru/standart-library/modul-difflib-python/brief-description/> (дата обращения: 24.04.2023).
19. Ожегов С.И. словарь русского языка М., 1984. 797 с.
20. Словарь иностранных слов. М. 1989.624с.
21. Левин В.И. Плагиат, его сущность и борьба с ним//Высшее образование в России. 2018. №1(219). С.143-150
22. Уголовный кодекс Российской Федерации. Официальный текст: текст Кодекса приводится по состоянию на 1 февраля 2022 г. – М.: Норматика, 2022. – 240с.
23. Левин В.И. Проблема плагиата в научно-образовательной сфере и общественной жизни России //Alma Mater (Вестник высшей школы) 2014. №6. С. 111-114.
24. Рындина, С. В. Методы и средства моделирования бизнес-процессов: объектно-ориентированная методология: учеб.-метод. пособие / С. В. Рындина. – Пенза: Изд-во ПГУ, 2017. – 48 с

Приложения

Приложение А. Пример текстов для проверки разработанного сервиса проверки на плагиат.

Текст, который проверялся на плагиат.

Нормализация баз данных является процессом организации данных в структурированной форме, чтобы устранить чрезмерность и обеспечить результативность операций с базой данных (замена на синонимы некоторых слов). Пeрвая нормaльнaя фoрма: всe aτрибуτы дoлжны быτь aτoмaрными, τo eсτь нe дoпуcкaeτся нaличиe пoвτoряющихся групп знaчeний в oднoм пoлe (замена русских букв на латиницу и греческий алфавит). Но это не спасает от избыточности, ее устраняет вторая форма (предложение без плагиата). Значения при третьей, в запись входящие и не являющиеся частью ключа этой записи, таблице не принадлежат (перестановка слов). Чετверτая нормальная форма: нε должно быτь многозначныχ зависимосτей мεжду аτрибуτами, то есть аτрибуτы, нε оτносящиеся к пεрвичному ключу, нε должны зависеть друг от друга (замена русских букв на латиницу и греческий алфавит). Обычная пятая форма: устранεние зависимостей соединения, когда сущεствует вероятность представить связи мεжду атрибутами в виде каждых таблиц (комбинация видов плагиата). Есть также еще шестая и нормальная форма Бойса-Кодда, которые используются редко (предложение без плагиата). Есть также нулевая форма, к которой относятся ненорализованные таблицы (полное копирование). В ненормализованной форме таблица может хранить информацию о двух и более сущностях (предложение без плагиата). Хотя возможны и другие уровни нормализации, третья обычная форма считается наивысшим уровнем, необходимым для большинства приложений (полное копирование).

Тексты, с которыми сравнивался данный текст. Для усложнения задачи некоторые из них также содержат похожие предложения.

1. Нормализация баз данных является процессом организации данных в структурированную форму, чтобы устранить избыточность и обеспечить эффективность операций с базой данных. Основные принципы нормализации включают первую, вторую, третью и другие нормальные формы (1НФ, 2НФ, 3НФ и другие). 1НФ. Преимущества 1НФ включают лучшую структурированность данных и устранение избыточности. Однако недостатком является возможность потери связей между данными. 2НФ. Преимущества 2НФ включают устранение избыточности и обеспечение более эффективного доступа к данным. Недостатком является возможность дублирования данных. 3НФ. Цель состоит в том, чтобы достичь наилучшего баланса между структурированностью данных, эффективностью операций и обеспечением связей между данными.
2. Нормализация баз данных является процессом организации структуры базы данных, чтобы избежать избыточности и несогласованности данных. Первая нормальная форма: все атрибуты должны быть атомарными, то есть не допускается наличие повторяющихся групп значений в одном поле. Вторая нормальная форма: каждый атрибут должен зависеть полностью от первичного ключа, и не должно быть частичных зависимостей между атрибутами и первичным ключом. Третья нормальная форма: не должно быть транзитивных зависимостей между атрибутами, то есть каждый атрибут должен зависеть только от первичного ключа, а не от других атрибутов. Четвертая нормальная форма: не должно быть многозначных зависимостей между атрибутами, то есть атрибуты, не относящиеся к первичному ключу, не должны зависеть друг от друга. Пятая нормальная форма: устранение зависимостей соединения, когда существует возможность представить связи между атрибутами в виде отдельных таблиц. Преимущества нормализации включают повышение эффективности хранения данных, сокращение дублирования информации, обеспечение согласованности данных и улучшение производительности запросов. Однако нормализация может иметь и некоторые недостатки. Каждая стадия нормализации может привести к увеличению количества таблиц в базе данных, что может усложнить процесс запросов и соединений между таблицами. Также, слишком высокая нормализация может привести к ухудшению производительности при выполнении сложных запросов, требующих множественных соединений между таблицами.
3. Существует несколько правил нормализации баз данных. Каждое правило называется обычной формой. Если нет использования нескольких полей в одной таблице для хранения похожих данных, база данных считается в "первой обычной форме". При второй записи могут зависеть только от первичного ключа таблицы (составного ключа, если необходимо). При третьей значения, входящие в запись и не являющиеся частью ключа этой записи, не принадлежат таблице. Если содержимое группы полей может относиться более чем к одной записи в таблице, попробуйте поместить эти поля в отдельную таблицу. Выполнять нормализацию баз данных до третьей нормальной формы теоретически желательно, но не всегда практично. Например, для устранения всех возможных зависимостей между полями таблицы Customers придется создать отдельные таблицы для хранения сведений о городах, почтовых индексах, торговых представителях, категориях клиентов и любых других сведений, которые могут дублироваться в нескольких записях. С теоретической точки зрения нормализация желательна. Хотя возможны и другие уровни нормализации, третья обычная форма считается наивысшим уровнем, необходимым для большинства приложений.
4. В реляционных базах данных есть такое понятия, как «Нормализация». Нормализация – это процесс удаления избыточных данных. Дело в том, что избыточность данных создает предпосылки для появления различных аномалий, снижает производительность, и делает управление данными не гибким и не очень удобным. Отсюда можно сделать вывод, что нормализация нужна в случаях. Необходимо устранение аномалий. Необходимо повышения производительности. Необходимо повышения удобства управления данными. Перечень основных нормальных форм: UNF, 1NF, 2NF, 3NF. Существуют дополнительные: BCNF, 4NF, 5NF, 6NF. Оптимальной считается третья.
5. Нормализация устраняет избыточность данных (data redundancy) и тем самым избежать нарушения целостности данных при их изменении, то есть избежать аномалий изменения (update anomaly). Как правило, нормализация преимущественно применяется при восходящем подходе проектировании базы данных, то есть когда мы все атрибуты, которые надо сохранить в бд, группируем по сущностям, для которых затем создаются таблицы. Однако при нисходящем подходе, когда вначале выявляются сущности, а затем их атрибуты и связи между ними, нормализация также может применяться, например, для проверки корректности спроектированных таблиц. Нормализация предполагает применение нормальных форм к структуре данных. Существуют 7 нормальных форм. Каждая нормальная форма (за исключением первой) подразумевает, что к данным уже была применена предыдущая нормальная форма. Также она может содержать повторяющиеся столбцы. Например, прежде чем применить третью нормальную форму к данным должна быть применена вторая нормальная форма. В ненормализованной форме таблица может хранить информацию о двух и более сущностях. И строго говоря, база данных считается нормализованной, если к ней применяется третья нормальная форма и выше.

Приложение Б. Разработанный сервис проверки на плагиат

from flask import Flask, request

from nltk.corpus import wordnet

import mimetypes

from difflib import SequenceMatcher

import sqlparse

from bs4 import BeautifulSoup

from fuzzywuzzy import fuzz

from fuzzywuzzy import process

import nltk

nltk.download('wordnet')

nltk.download('omw-1.4')

app = Flask(\_\_name\_\_)

def find\_synonyms(word):

    synonyms = set()

    for syn in wordnet.synsets(word):

        for lemma in syn.lemmas():

            synonyms.add(lemma.name())

    return synonyms

def check\_russian\_to\_english\_greek(char):

    russian\_to\_english = {'а': 'a', 'е': 'e', 'р': 'p', 'о': 'o', 'с': 'c', 'у': 'y', 'к': 'k', 'х': 'x', 'в': 'b',

                          'м': 'm', 'т': 't'}

    russian\_to\_greek = {'а': 'α', 'е': 'ε', 'р': 'ρ', 'о': 'ο', 'с': 'ς', 'у': 'υ', 'к': 'κ', 'х': 'χ', 'в': 'ν',

                        'м': 'μ', 'т': 'τ'}

    if char in russian\_to\_english.values():

        return True, list(russian\_to\_english.keys())[list(russian\_to\_english.values()).index(char)]

    elif char in russian\_to\_greek.values():

        return True, list(russian\_to\_greek.keys())[list(russian\_to\_greek.values()).index(char)]

    elif char in russian\_to\_english.keys():

        return False, russian\_to\_english[char]

    elif char in russian\_to\_greek.keys():

        return False, russian\_to\_greek[char]

    else:

        return True, char

def find\_common\_phrases(text1, text2):

    common\_phrases = []

    phrases1 = text1.lower().split('.')

    phrases2 = text2.lower().split('.')

    for phrase1 in phrases1:

        words1 = phrase1.strip().split()

        for phrase2 in phrases2:

            words2 = phrase2.strip().split()

            for i in range(0, len(words1), 2):

                word1 = words1[i]

                if word1.isalpha() and all(ord(c) < 128 for c in word1):

                    try:

                        word1.encode(

                            encoding='utf-8').decode('ascii')

                    except UnicodeDecodeError:

                        for r in range(0, len(words1)):

                            replacebool, char = check\_russian\_to\_english\_greek(

                                word1[r])

                        if replacebool:

                            phrase1 = phrase1.replace(

                                word1[r], char, 1)

                        break

                synonyms1 = find\_synonyms(word1)

                for word2 in words2:

                    if word2.isalpha() and all(ord(c) < 128 for c in word2):

                        try:

                            word2.encode(

                                encoding='utf-8').decode('ascii')

                        except UnicodeDecodeError:

                            for r in range(0, len(words2)):

                                replacebool, char = check\_russian\_to\_english\_greek(

                                    word2[r])

                            if replacebool:

                                phrase2 = phrase2.replace(

                                    word2[r], char, 1)

                            break

                    if word2 in synonyms1:

                        phrase1 = phrase1.replace(

                            word1, word2, 1)

            similarity = fuzz.token\_set\_ratio(phrase1, phrase2)

            if similarity >= 60:

                if phrase1 not in common\_phrases:

                    common\_phrases.append(phrase1)

    return common\_phrases

def calculate\_plagiarism\_percentage(common\_phrases, text):

    total\_words = text.lower().split()

    plagiarized\_words = ' '.join(common\_phrases).lower().split()

    plagiarism\_percentage = (len(plagiarized\_words) / len(total\_words)) \* 100

    if plagiarism\_percentage > 100:

        plagiarism\_percentage = 100

    return plagiarism\_percentage

def getHtmlData(filename):

    file\_type, \_ = mimetypes.guess\_type(filename)

    if file\_type != 'text/html':

        return filename

    soup = BeautifulSoup(filename, 'html.parser')

    paragraphs = soup.find\_all('p')

    text = ' '.join([p.text for p in paragraphs])

    return text

@app.route('/checkText', methods=['POST'])

def plagiarism\_check():

    texts = request.json

    comparison\_array = []

    for text in texts:

        comparison\_array.append(getHtmlData(text))

    common\_phrases = []

    for i in range(len(comparison\_array) - 1):

        common\_phrases.extend(find\_common\_phrases(

            comparison\_array[0], comparison\_array[i + 1]))

    if common\_phrases:

        plagiarism\_percentage = calculate\_plagiarism\_percentage(

            common\_phrases, comparison\_array[0])

        return f'{100 - plagiarism\_percentage}%'

    else:

        return '100%'

if \_\_name\_\_ == '\_\_main\_\_':

    app.run()

Приложение В. Страница настройки плагина проверки на плагиат

<?php

// This file is part of Moodle - http://moodle.org/

//

// Moodle is free software: you can redistribute it and/or modify

// it under the terms of the GNU General Public License as published by

// the Free Software Foundation, either version 3 of the License, or

// (at your option) any later version.

//

// Moodle is distributed in the hope that it will be useful,

// but WITHOUT ANY WARRANTY; without even the implied warranty of

// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the

// GNU General Public License for more details.

//

// You should have received a copy of the GNU General Public License

// along with Moodle.  If not, see <http://www.gnu.org/licenses/>.

/\*\*

 \* plagiarismplag\_setupform.class.php - Plugin setup form for plagiarism\_plagiarismplag component

 \* @package   plagiarism\_plagiarismplag

 \* @copyright 2023 plagiarismplag

 \* @author    Masha Khalyavina

 \*/

defined('MOODLE\_INTERNAL') || die();

global $CFG;

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/lib.php');

require\_once($CFG->dirroot . '/lib/formslib.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/classes/plagiarism\_plagiarismplag\_comms.class.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/classes/plagiarism\_plagiarismplag\_pluginconfig.class.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/classes/plagiarism\_plagiarismplag\_moduleconfig.class.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/classes/exceptions/plagiarism\_plagiarismplag\_authexception.class.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/classes/exceptions/plagiarism\_plagiarismplag\_exception.class.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/classes/exceptions/plagiarism\_plagiarismplag\_ratelimitexception.class.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/classes/exceptions/plagiarism\_plagiarismplag\_undermaintenanceexception.class.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/classes/plagiarism\_plagiarismplag\_logs.class.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/constants/plagiarism\_plagiarismplag.constants.php');

require\_once($CFG->dirroot . '/plagiarism/plagiarismplag/classes/plagiarism\_plagiarismplag\_utils.class.php');

/\*\*

 \* plagiarismplag admin setup form

 \*/

class plagiarism\_plagiarismplag\_adminform extends moodleform {

    /\*\*

     \* Define the form

     \* \*/

    public function definition() {

        global $CFG;

        $mform = &$this->\_form;

        // Plugin Configurations.

        $mform->addElement(

            'header',

            'plagiarism\_plagiarismplag\_adminconfigheader',

            get\_string('cladminconfig', 'plagiarism\_plagiarismplag', null, true)

        );

        $mform->addElement(

            'html',

            get\_string('clpluginintro', 'plagiarism\_plagiarismplag')

        );

        // Get all modules that support plagiarism plugin.

        $plagiarismmodules = array\_keys(core\_component::get\_plugin\_list('mod'));

        $supportedmodules = array('assign', 'forum', 'workshop', 'quiz');

        foreach ($plagiarismmodules as $module) {

            // For now we only support assignments.

            if (in\_array($module, $supportedmodules) && plugin\_supports('mod', $module, FEATURE\_PLAGIARISM)) {

                array\_push($supportedmodules, $module);

                $mform->addElement(

                    'advcheckbox',

                    'plagiarism\_plagiarismplag\_mod\_' . $module,

                    get\_string('clenablemodulefor', 'plagiarism\_plagiarismplag', ucfirst($module == 'assign' ? 'Assignment' : $module))

                );

            }

        }

        $mform->addElement(

            'textarea',

            'plagiarism\_plagiarismplag\_studentdisclosure',

            get\_string('clstudentdisclosure', 'plagiarism\_plagiarismplag')

        );

        $mform->addHelpButton(

            'plagiarism\_plagiarismplag\_studentdisclosure',

            'clstudentdisclosure',

            'plagiarism\_plagiarismplag'

        );

        // plagiarismplag Account Configurations.

        $mform->addElement(

            'header',

            'plagiarism\_plagiarismplag\_accountconfigheader',

            get\_string('claccountconfig', 'plagiarism\_plagiarismplag')

        );

        $mform->setExpanded('plagiarism\_plagiarismplag\_accountconfigheader');

        // Thos settings will be save on Moodle database.

        $mform->addElement(

            'text',

            'plagiarism\_plagiarismplag\_apiurl',

            get\_string('clapiurl', 'plagiarism\_plagiarismplag')

        );

        $mform->setType('plagiarism\_plagiarismplag\_apiurl', PARAM\_TEXT);

    }

    /\*\*

     \* form custom validations

     \* @param mixed $data

     \* @param mixed $files

     \*/

    public function validation($data, $files) {

        $config = plagiarism\_plagiarismplag\_pluginconfig::admin\_config();

        $apiurl = $config->plagiarism\_plagiarismplag\_apiurl;

        $newapiurl = $data["plagiarism\_plagiarismplag\_apiurl"];

        if ($apiurl == $newapiurl) {

            return (array)[

                    "plagiarism\_plagiarismplag\_apiurl" => "same apiurl entered"

                ];

        }

        return array();

    }

    /\*\*

     \* Init the form data form both DB and plagiarismplag API

     \*/

    public function init\_form\_data() {

        $cache = cache::make('core', 'config');

        $cache->delete('plagiarism\_plagiarismplag');

        // Get moodle admin config.

        $plagiarismsettings = (array) plagiarism\_plagiarismplag\_pluginconfig::admin\_config();

        if (

            !isset($plagiarismsettings['plagiarism\_plagiarismplag\_apiurl']) ||

            empty($plagiarismsettings['plagiarism\_plagiarismplag\_apiurl'])

        ) {

            $plagiarismsettings['plagiarism\_plagiarismplag\_apiurl'] = plagiarism\_plagiarismplag\_comms::plagiarismplag\_api\_url();

        }

        $cldbdefaultconfig = plagiarism\_plagiarismplag\_moduleconfig::get\_modules\_default\_config();

        if (!isset($plagiarismsettings["plagiarism\_plagiarismplag\_studentdisclosure"])) {

            $plagiarismsettings["plagiarism\_plagiarismplag\_studentdisclosure"] =

                get\_string('clstudentdisclosuredefault', 'plagiarism\_plagiarismplag');

        }

        $this->set\_data($plagiarismsettings);

    }

    /\*\*

     \* Display the form to admins

     \*/

    public function display() {

        ob\_start();

        parent::display();

        $form = ob\_get\_contents();

        ob\_end\_clean();

        return $form;

    }

    /\*\*

     \* Save form data

     \* @param stdClass $data

     \*/

    public function save(stdClass $data) {

        global $CFG;

        // Save admin settings.

        $configproperties = plagiarism\_plagiarismplag\_pluginconfig::admin\_config\_properties();

        foreach ($configproperties as $property) {

            plagiarism\_plagiarismplag\_pluginconfig::set\_admin\_config($data, $property);

        }

        // Check if plugin is enabled.

        $plagiarismmodules = array\_keys(core\_component::get\_plugin\_list('mod'));

        $pluginenabled = 0;

        foreach ($plagiarismmodules as $module) {

            if (plugin\_supports('mod', $module, FEATURE\_PLAGIARISM)) {

                $property = "plagiarism\_plagiarismplag\_mod\_" . $module;

                $ismoduleenabled = (!empty($data->$property)) ? $data->$property : 0;

                if ($ismoduleenabled) {

                    $pluginenabled = 1;

                }

            }

        }

        // Set if plagiarismplag plugin is enabled.

        set\_config('enabled', $pluginenabled, 'plagiarism\_plagiarismplag');

        if ($CFG->branch < 39) {

            set\_config('plagiarismplag\_use', $pluginenabled, 'plagiarism');

        }

        $cache = cache::make('core', 'config');

        $cache->delete('plagiarism\_plagiarismplag');

    }

}

Приложение Г. Авторская справка

Я, Халявина Мария Сергеевна,

автор выпускной квалификационной работы, сообщаю, что мне известно о персональной ответственности автора за разглашение сведений, подлежащих защите законами РФ о защите объектов интеллектуальной собственности.

Одновременно сообщаю, что

1. при подготовке к защите (опубликованию) выпускной квалификационной работы не использованы источники (документы, отчеты, диссертации, литература и т.п.), имеющие гриф секретности или «Для служебного пользования» ВятГУ или другой организации;
2. данная работа не связана с незавершенными исследованиями или с уже завершенными, но еще официально не разрешенными к опубликованию ВятГУ или другими организациями;
3. данная работа не содержит коммерческую информацию, способную нанести ущерб интеллектуальной собственности ВятГУ или другой организации;
4. данная работа не является результатом НИР или ОКР, выполняемой по договору организацией.
5. в предлагаемом к опубликованию тексте нет данных по незащищенным объектам интеллектуальной собственности других авторов;
6. согласен на использование результатов своей работы ВятГУ для учебного процесса;
7. использование моей выпускной квалификационной работы в научных исследованиях оформляется в соответствии с законодательством РФ о защите интеллектуальной собственности.

« » 20 г. Подпись автора

Сведения по авторской справке подтверждаю

« » 20 г.

Ф.И.О. зав. Кафедрой