ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
Факультет Санкт-Петербургская школа физико-математических и компьютерных наук

Плющенко Даниил Владимирович

РАЗРАБОТКА ВЕБ-ТРЕНАЖЁРА ПУБЛИЧНЫХ ВЫСТУПЛЕНИЙ

Выпускная квалификационная работа
по направлению подготовки 01.04.02 Прикладная математика и информатика
образовательная программа «Программирование и анализ данных»

Рецензент
канд. тех. наук, доц.
Д. И. Муромцев

Научный руководитель
д-р физ.-мат. наук, проф.
А. В. Омельченко

Консультант
канд. тех. наук, доц.
М. М. Заславский

Санкт-Петербург 2021

Оглавление

Аннотация

Данная работа посвящена проблеме автоматической оценки публичных выступлений. Предлагаемое решение представляет собой веб-приложение с гибкой архитектурой и возможностью интеграции с внешними системами. Для поиска решения был проведён обзор литературы, посвященной автоматической оценке публичных выступлений, и изучены существующие приложения. Было замечено, что в существующих приложениях в целом отсутствует настройка способов оценивания, просмотр и сохранение файлов презентаций и интеграция с внешними приложениями, например, с системами управления обучением. Приложение, описанное в данной работе, обеспечивает обратную связь на основе набора критериев, таких как продолжительность речи, темп речи, использование слов-паразитов и другие. Была предложена гибкая архитектура приложения, поэтому можно добавлять, удалять или изменять шаги оценивания, а также масштабировать систему в случае потенциально увеличивающейся нагрузки. Было измерено время обработки тренировки как для одной тренировки, так и для набора тренировок, обработка которых началась одновременно. На данный момент самым длинным этапом обработки является распознавание речи, поэтому рассматривались конфигурации приложения с разным количеством экземпляров распознавания речи, чтобы сократить время обработки тренировок.

Ключевые слова – публичное выступление, оценка речи, протокол LTI.

This work is dedicated to public speaking automatic evaluation. The proposed solution is a web application with a flexible architecture and the ability to integrate with external systems. To find a solution, an overview of the literature and existing public speaking applications is provided. It was noticed that in general existing applications lacked evaluation customization, presentation file handling features, and integration with external services such as learning management systems. This application provides feedback based on a pack of criteria such as speech duration, speech pace, fillers usage, and others. Flexible application architecture was proposed so new steps of evaluation can be added, removed, or changed, also it can be scaled to support potentially increasing application load. Training processing time both for a single training and a set of trainings submitted to the application simultaneously was measured. So far the longest step of processing is speech recognition so configurations with a different number of speech recognition instances were considered to get faster training processing time.

Keywords – public speaking, speech evaluation, LTI protocol.

Введение

Большое количество студентов и исследователей регулярно выступают публично, чтобы рассказать о проделанной работе. Это может происходить, к примеру, на конференциях, семинарах, защитах курсовых и выпускных квалификационных работ, а также диссертаций. Для того, чтобы улучшить качество своего выступления, докладчик обычно тренируется, репетируя свой доклад самостоятельно. Это можно делать, например проговаривая доклад про себя, вслух или перед зеркалом. Тренировка также может и проводиться с привлечением других людей в качестве слушателей, которые смогут поделиться обратной связью. Однако чтобы получить более полезную и содержательную обратную связь, слушатель должен иметь высокий уровень экспертизы в предметной области. К сожалению, количество времени, доступного для прослушивания тренировочных докладов, у экспертов зачастую ограничено. Особенно это заметно в случае студенческих работ, когда научных руководителей сильно меньше, чем самих студентов, кроме того, зачастую один человек является научным руководителем нескольких студентов.

Несмотря на то, что оценка публичного выступления и предоставление обратной связи – трудоёмкие действия, некоторая первичная оценка может быть проведена автоматически. К примеру, можно проверять:

- длительность доклада;
- скорость речи докладчика;
- порядок секций в презентации;

- наличие просторечий или ”слов-паразитов”.

В целом автоматической оценке поддаются любые критерии, которые можно описать формально, например, в виде набора правил или алгоритма и его реализации. При этом нужно понимать, что интересующие критерии могут отличаться в зависимости от требований руководителя или организации, в которой проходит публичное выступление.

Объект исследования – процесс оценивания публичных выступлений.

Предмет исследования – автоматизация процесса оценки качества публичного выступления.

Методы исследования включают в себя методы программной инженерии и методы проектирования информационных систем.

Цель и задачи

Целью данной работы является разработка модульного, а также масштабируемого с точки зрения нагрузки, новых шагов оценивания и конфигураций веб-приложения с открытым исходным кодом для подготовки к публичным выступлениям и первичной оценки их качества. Для достижения цели были поставлены следующие задачи:

- провести обзор аналогов;
- провести обзор библиотек распознавания речи;
- разработать расширяемую архитектуру, позволяющую добавлять и заменять компоненты, связанные с хранением и анализом публичного выступления;

- разработать веб-приложение в соответствии с предложенной архитектурой.

Сценарии использования включают в себя как индивидуальные тренировки, так и ситуации, при которых имеется несколько групп студентов, и каждый студент должен провести несколько тренировок и набрать требуемое количество баллов в качестве одного из шагов подготовки к защите выпускной квалификационной работы. В рамках данной работы основным языком публичного выступления считается русский язык.

Работа состоит из трёх глав, структура работы следующая: в главе 1 представлен обзор литературы, существующих приложений-аналогов и библиотек распознавания речи, в главе 2 описывается предлагаемое решение, в частности его архитектура и детали реализации, в главе 3 представлено исследование свойств решения.

1. Обзор существующих решений

1.1. Обзор литературы

В данном разделе представлены статьи, в которых описываются системы, автоматически оценивающие публичные выступления, для того чтобы изучить их устройство и функциональность, а также понять, подходят ли они для достижения цели данной работы. Кроме того, были рассмотрены критерии, по которым можно определить, является ли данное публичное выступление хорошим или нет.

В 2015 году Копф и его коллеги выпустили статью [13], в которой описывается приложение для оценки презентационных навыков. Это приложение опирается скорее на поведение говорящего, чем на содержание презентации и речи. Учитываются такие критерии, как жесты, зрительный контакт с аудиторией, положение тела докладчика. В качестве датчика ввода используется Microsoft Kinect [26]. Система предоставляет обратную связь как в режиме реального времени, так и статистику по всему докладу после его окончания. Авторы также отмечают, что темп речи и длительность доклада в течение слайда важны, однако в рамках своего приложения не обрабатывают фактическое содержимое доклада и предоставляют обратную связь на основе этого. Кроме того, в статье отмечается, что не все произнесённые слова распознаются корректно с использованием Microsoft Kinect. Возможно, использование более специализированного инструментария, а также его комбинация с Microsoft Kinect, дало бы более хорошие результаты. Для вычисления количества слайдов на презентации определяются моменты, когда изображение на заднем плане меняется достаточно сильно, однако эту информацию можно извлечь

непосредственно из файла с презентацией.

Ханани и его коллеги в 2017 году опубликовали статью [2], в которой они представляют фреймворк для автоматической оценки навыков публичного выступления. Фреймворк состоит из трёх подсистем, обрабатывающих аудиозапись, презентацию и жесты докладчика соответственно. Первая подсистема извлекает десять признаков из аудиозаписи, вторая подсистема извлекает шесть признаков из презентации, третья подсистема извлекает пять признаков из жестов докладчика, после чего используются методы машинного обучения для классификации. Каждое значение классифицируется как высокое или низкое для одной части экспериментов, и как высокое, среднее или низкое в другой части экспериментов, однако по-прежнему невозможно выбрать желаемое подмножество критериев, используемых для оценивания публичного выступления.

В статье [11] Шнайдера и его коллег, опубликованной в 2015 году, описывается система Presentation Trainer, предназначенная для тренировки невербальных навыков, полезных при публичном выступлении. С использованием датчика Microsoft Kinect система анализирует жесты докладчика, положение его тела, громкость речи, количество и длительность пауз в ней, а также наличие т.н. "заполненных пауз". Система предоставляет обратную связь в режиме реального времени, в том числе она может прерывать докладчика, например, в случае, если его руки длительное время скрещены. В статье также кратко описывается архитектура всей системы, состоящей из нескольких модулей, основные из которых отвечают за: интерпретацию данных с сенсора, анализ правил и генерацию действия – обратной связи, показываемой пользователю.

1.2. Обзор существующих приложений

Для поиска аналогов были рассмотрены приложения, которые так или иначе оценивают навыки публичного выступления и / или речи. Согласно цели этой работы, критериями для сравнения были:

- возможность интеграции с внешними инструментами, к примеру, наличие API или возможность взаимодействия с LMS (Learning Management System), поддержка протокола LTI (Learning Tools Interoperability) [5];
- гибкая конфигурация используемых для оценки критериев;
- возможность прикрепить презентацию;
- возможность записать свою речь.

Протокол LTI поддерживается многими популярными системами управления обучением и платформами массовых открытых онлайн-курсов, такими как Moodle[1], Blackboard Learn[2], Stepik[3], edX[4] и другими Этот протокол может использоваться для обмена информацией о задачах и оценках, а также для авторизации в одной системе через другую. Для того, чтобы интегрировать приложение и LMS с помощью протокола LTI, нужно проделать несколько шагов.

Для поиска приложений-аналогов использовались такие ключевые слова и ключевые фразы, как ”speech”, ”pronunciation”, ”speaking”, ”evaluation”, ”analysis”,

[1]https://docs.moodle.org/311/en/LTI_and_Moodle

[2]https://help.blackboard.com/Learn/Administrator/SaaS/Integrations/Learning_Tools_Interoperability

[3]https://support.stepik.org/hc/ru/articles/360010307320-Интеграция-по-протоколу-LTI

[4]https://edx.readthedocs.io/projects/edx-installing-configuring-and-running /en/latest/configuration/lti/index.html

”public speaking training”, ”public speaking application”, ”presentation” и их переводы на русский язык.

Speechace API [20] – сервис, обеспечивающий оценку речи, произнесённой на американском английском и британском английском. Обратная связь предоставляется для каждого звука и каждого слога. Оценивается, был ли сказанный фрагмент похож на эталонное произношение или, если нет, то почему. Кроме того, через API можно получить количество слов, количество слогов в секунду, статистику пауз, правильное соотношение слов и другую статистику. Speechace - единственный найденное приложение, поддерживающее интеграцию через протокол LTI.

Voice Notebook [19] (ранее SpeechPad) – это веб-сервис, являющийся набором инструментов для распознавания речи. Также есть мобильные приложения, с помощью которых можно преобразовать речь в текст с временными отметками. Один из инструментов предоставляет возможность протестировать произношение. Пользователь вводит текст, а затем включает запись звука и читает текст. Оценка обновляется после каждого произнесённого предложения и представляет собой процент распознанных слов и полноту распознанных слов для последнего произнесенного предложения и для всего текста. Другой инструмент предоставляет выбранное количество версий распознанных текстов с указанием их вероятностей.

Speakit [1] – это платное приложение для Android, которое позволяет пользователю проверить произношение на американском английском. Слова и фразы можно искать в словаре или по типу звука, который в них содержится (например, короткие гласные, длинные гласные, дифтонги и так далее), Или по категории (например, еда,

семья, друзья и так далее). Пользователь можно записать своё произношение этого слова или фразы и получить обратную связь, в которой будет указано, было ли оно похоже на эталонное произношение или нет.

Aksent [16] – это приложение для iOS, которое вычисляет процент сходства между эталонным произношением и произношением пользователя на одном из более чем двадцати поддерживаемых языков. Слово или фразу можно набрать и перевести на целевой язык.

Speeko [21] – это платное приложение для iOS. Приложение анализирует речь с точки зрения темпа, красноречия, пауз, интонации и артикуляции, обеспечивая такие показатели, как количество слов в минуту и статистику частоты для каждого произнесенного слова-паразита. Также возможно получить транскрипцию и запись выступления. Пользователь может установить предел продолжительности записи и записывать свои заметки, которые будут отображаться во время записи речи.

LikeSo [6] – это платное приложение для iOS, которое помогает избавиться от слов-паразитов в речи (в английском языке в качестве таких слов часто используются "like" и "so"). Пользователь может установить продолжительность записи (до 30 минут) и подмножество слов-паразитов, использование которых будет влиять на оценку. Общий балл представляет собой относительную долю слов в речи, не считающихся словами-паразитами, также в обратной связи представлена статистика частоты произнесенных слов-паразитов, общее количество произнесенных слов и темп речи, вычисленный в количестве произнесённых слов в минуту.

ELSA Speak [3] – мобильное приложение, доступное как для iOS,

так и для Android, помогающее говорить на американском английском. Приложение содержит несколько тем с относящимися к ней фразами и словами. Для каждого звука из записанной речи пользователя сообщается, похож ли он на эталонное произношение или нет, также предоставляется общий балл, отражающий сходство всего слова или фразы.

Orai [9] – платное мобильное приложение, доступное как для iOS, так и для Android. Приложение предоставляет анализ записанной речи, включая статистику темпа речи, статистику использования слов-паразитов и уровень уверенности докладчика, зависящий, к примеру, от количества повторяющихся слов. Также в приложении можно сохранять текстовые заметки и выбирать заметку, которая будет отображаться во время записи речи.

Говорилло [29] – это приложение для Android, которое позволяет пользователю записывать свою речь и получать обратную связь, которая содержит информацию о темпе, относительной доли слов-паразитов и сложности речи. Последний показатель возвращает нижнюю границу возраста слушателя, которому будет комфортно понимать записанную речь.

Несмотря на то, что существующие приложения предоставляют измеримую оценку речи, в целом, у них отсутствует возможность:

- прикрепления презентаций (только два приложения позволяют делать текстовые заметки);
- настройки критериев (только два приложения позволяют настроить подмножество слов-паразитов, использование которых будет влиять на итоговую оценку);

- интеграции внешних инструментов (только одно приложение имеет API и поддержку LTI).

Кроме того, в большинстве приложений единственным поддерживаемым языком является английский, часть приложений в принципе предназначена для оценки произношения отдельных слов или фраз. Это делает невозможным их использование для вышеупомянутых сценариев использования. Сравнение существующих приложений представлено в таблице 1.

Таблица 1: Сравнение существующих приложений

Название	Платформы	API?	Можно добавить доклад?	Макс. время записи?	Языки	Можно настроить критерии?	Запись речи?	Платное?
Speechace	Web	Да	Нет	15 секунд	Английский	Нет	Да	Бесплатное демо
Voice Notebook	Web	Нет	Нет	Нет	8	Нет	Да	Нет
Speakit	Android	Нет	Нет	Одна фраза	Американский английский	Нет	Да	Да
Aksent	iOS	Нет	Нет	Одна фраза	20+	Нет	Да	Нет
Speeko	iOS	Нет	Текст	Нет	Английский	Нет	Да	Бесплатная trial-версия
Likeso	iOS	Нет	Нет	30 минут	Английский	Подмножество слов-паразитов	Да	Да
Orai	iOS, Android	Нет	Текст	Нет	Английский	Подмножество слов-паразитов	Да	Да
ElsaSpeak	iOS, Android	Нет	Нет	Одна фраза	Американский английский	Нет	Да	Бесплатное демо
Говорилло	Android	Нет	Нет	Нет	Русский	Нет	Да	Нет

1.3. Обзор библиотек распознавания речи

В этом разделе сравниваются библиотеки распознавания речи, так как информация, полученная из распознанных аудиозаписей докладов будет использоваться для выставления оценки. Критериями для сравнения являются:

- возможность работы без подключения к Интернету (для избавления от внешних зависимостей);

- распознавание речи на русском языке, так как русский – основной язык в предполагаемых сценариях использования;
- предоставление временных меток для распознанных слов (для сопоставления слов докладчика с содержимым презентации);
- цена – искомая библиотека должна быть бесплатной, поскольку весь проект предполагается бесплатным.

Сравнение библиотек представлено в таблице 2.

Бесплатными библиотеками, поддерживающие режим работы без доступа к Интернету и распознавание речи на русском языке, являются Vosk [22] и Speech-To-Text (russian) [15]. Эти две библиотеки вместе с Wit.ai [25] были рассмотрены более подробно.

Таблица 2: Сравнение библиотек распознавания речи

Название	Офлайн?	Русский язык	Временные метки	Стоимость
Vosk	Да	Да	Да	Бесплатно
Speech-To-Text (Russian)	Да	Да	Нет	Бесплатно
Picovoice [10]	Да	Нет	Нет	Бесплатно
at16k [27]	Да	Нет	Да	Бесплатно
Google Web Speech API [24]	Нет	Да	Да	Бесплатно час в месяц
Google Cloud Speech API [18]	Нет	Да	Да	Бесплатно час в месяц
Microsoft Bing Speech API [17]	Нет	Да	Да	Бесплатно 5000 запросов в месяц
IBM Speech to Text [23]	Нет	Нет	Да	Бесплатно 50 часов нагрузки в месяц
Wit.ai	Нет	Да	Да	Бесплатно

Было проведено сравнение качества распознавания речи с использованием алгоритма w-shingle [7]. Шингл – это пересекающиеся подотрезки текста, количество слов в которых составляет фиксированную

длину. Например, для текста ”A B C D”:

- шинглы длины 1 – это [”A”, ”B”, ”C”, ”D”];
- шинглы длины 2 – это [”AB”, ”BC”, ”CD”];
- шинглы длины 3 – это [”ABC”, ”BCD”];
- шинглы длины 4 – это [”ABCD”];
- шинглов длины 5 и более в данном тексте нет.

Качество распознавания измерялось как средняя относительная доля правильно распознанных шинглов в собранном наборе данных. Набор данных состоит из десяти выступлений с презентациями, сделанных на русском языке. Из десяти докладчиков пятеро являются мужчинами, остальные пять – женщины. Выступления взяты из защит бакалаврских и магистерских дипломов, а также из уроков на образовательной платформе Stepik. Эталонные транскрипции аудиозаписей были получены вручную после их прослушивания, также были удалены знаки препинания.

Библиотека Vosk была выбрана, потому что она дает лучшие результаты распознавания речи, как показано в таблице 3.

Таблица 3: Сравнение библиотек распознавания речи

Размер шингла	1	2	3	4
Название	Средняя доля правильно распознанных шинглов, %			
Vosk	74.84	60.27	50.08	42.24
Speech-to-Text (Russian)	49.14	28.27	17.78	11.82
Wit.ai	55.29	35.70	25.47	18.85

1.4. Выводы и результаты по главе

В данной главе приведён обзор литературы по теме автоматического оценивания публичных выступлений. Также в данной главе рассмотрены приложения, позиционирующие себя как улучшающие навыки публичных выступлений и/или речи. Сделан вывод о том, что существующие приложения не подходят для достижения целей данной работы.

Кроме того, был произведён обзор библиотек распознавания речи и последующее сравнение наиболее подходящих кандидатов на специально собранном наборе данных, состоящем из аудиозаписей и презентаций различных публичных выступлений.

2. Описание решения

Разработка решения в виде веб-приложения позволит дать доступ к тренажёру вне зависимости от того, какую платформу использует пользователь, а также позволяет без каких-либо действий предоставлять пользователям новую версию приложения, что особенно важно на начальных этапах разработки, когда принципиальные изменения могут появляться довольно часто. Запись речи с использованием микрофона, а также просмотр и переключение слайдов на презентации, что является основными действиями пользователя во время тренировки, также доступны во всех современных браузерах вне зависимости от платформы. Поэтому было принято решение разрабатывать тренажёр, по крайней мере видимую для пользователя часть, именно как веб-приложение.

2.1. Обработка данных

Каждая тренировка состоит из трёх сущностей: аудиозаписи, презентации и списка с временными метками переключений слайдов. Первые две сущности нужно сначала распознать, то есть получить из исходного файла текст, а после структурировать для того, чтобы эту информацию использовать при оценке тренировки. Поэтому аудиозаписи и презентации представлены в трёх состояниях:

- RAW – сырое состояние (исходные файлы);
- RECOGNIZED – распознанное состояние, то есть последовательность слов, разбитая на слайды в соответствии с временными метками переключений слайдов;

- PROCESSED – обработанное состояние, то есть данные в распознанном состоянии, для которых были подсчитаны статистические метрики и другая дополнительная информация.

Для данных в обработанном состоянии можно вычислять критерии. Каждый критерий возвращает дробное число, показывающее, соответствует ли тренировка этому критерию или нет. Кроме того, каждый критерий содержит информацию о зависимых критериях, поскольку результат может зависеть от значений других критериев. Например, критерий может проверять, что выступление не длилось слишком долго.

Возможно, что в зависимости от ситуации, например, от университета, в котором будет происходить предзащита, или от требований комиссии, критерий с одной и той же сутью может использоваться по-разному. Чтобы не создавать отдельного критерия для каждой подобной ситуации, введены "параметризованные" критерии. Например, параметризованный критерий проверяет, что продолжительность выступления не превышает семи минут (при необходимости это значение можно изменить на любое другое).

Поскольку тренировку может быть необходимо проверить на произвольном наборе критериев, параметризованные критерии объединяются в наборы критериев, содержащие информацию о порядке применяемых критериев. Когда представлены результаты всех критериев, используется функция оценивания тренировки. Функция оценивания может быть произвольной, чтобы иметь возможность обрабатывать результаты одного и того же наборы критериев по-разному, например, чтобы сделать критерий блокирующим (если критерий не выполнен, то общая оценка за тренировку равна нулю), подчеркнуть

важность определенного критерия за счёт присвоения ему большого веса или присвоить всем критериям одинаковый вес. Схема с набором примеров представлена на рис. 1. Для оценивания могут быть выбраны имеющийся набор критериев и функция оценки или созданы и использованы новые.

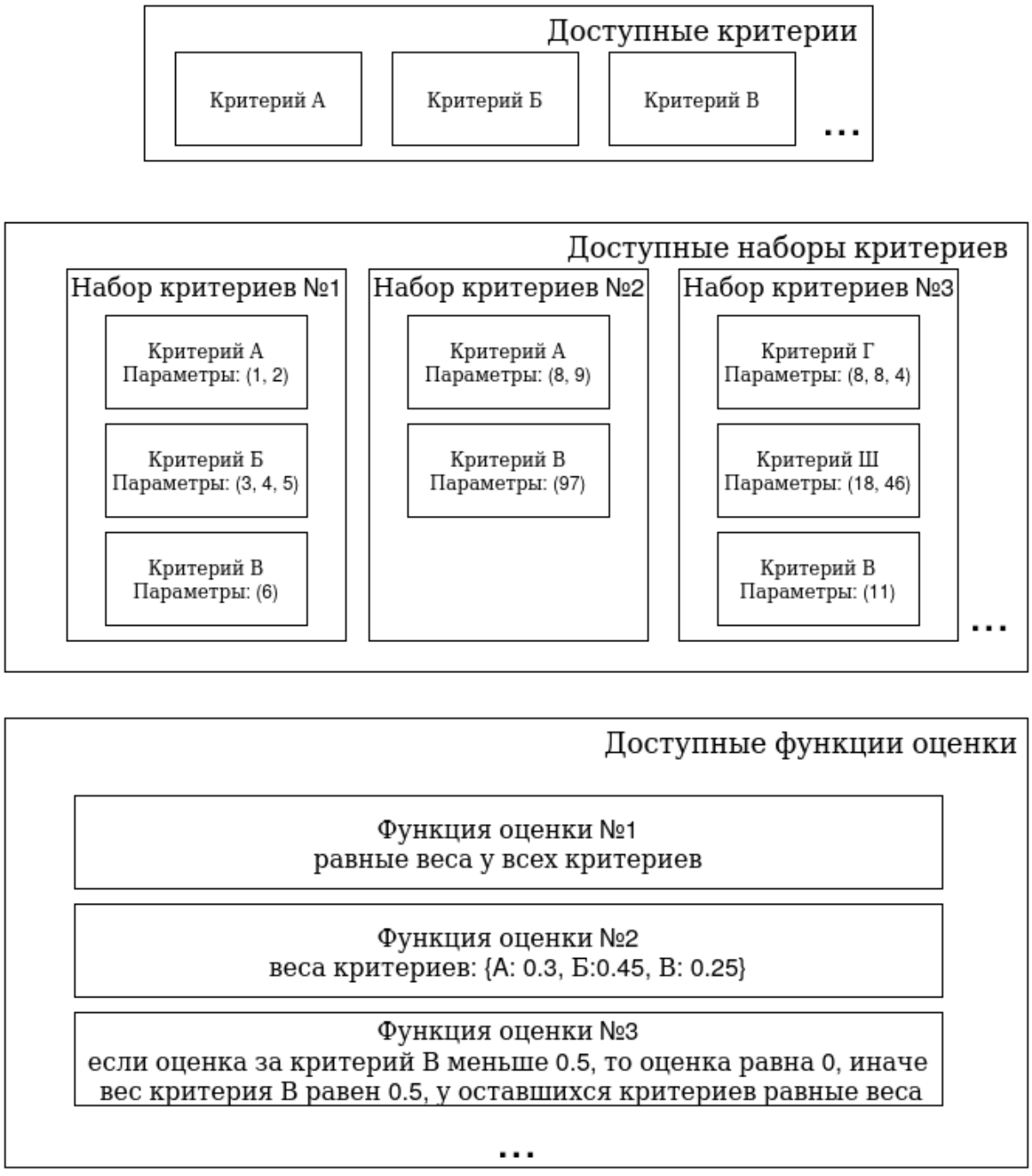

Рис. 1: Схема критериев, параметризованных критериев и функций оценки

2.2. Реализованные критерии и поддержка новых критериев

На данный момент в приложении доступно несколько критериев, которые можно параметризовать, а также объединять в наборы. Критерии описаны ниже в данном разделе.

Критерий, проверяющий, что продолжительность речи находится в заданных границах. Данный критерий был реализован в первую очередь, поскольку практически всегда выступление ограничено по времени, что может явно задаваться, например, расписанием конференции или правилами защиты выпускной квалификационной работы в конкретном ВУЗе. Параметры – границы, при выходе за которые оценка за критерий будет равна нулю, а также оптимальные границы, при попадании в которые оценка будет равна единице. Псевдокод ниже отображает правило, по которому вычисляется оценка.

```
a = minimal_duration
b = maximal_duration
t = actual_duration
if t < a:
  return (t / a) ** 2
elif t > b:
  return (b / t) ** 2
return 1
```

Критерий, проверяющий, что темп речи, измеряемый в количестве сказанных слов в минуту, находится в заданных границах. Этот критерий был реализован одним из первых, потому что слишком быстрая или слишком медленная речь мешает восприятию доклада, а также его оценке с точки зрения содержательных критериев. Параметры – границ, при попадании в которые оценка за критерий равна единице. Если темп речи p меньше, чем минимальная допустимая граница a,

то оценка равна p/a, если больше, чем максимальная допустимая граница b, то оценка за критерий равна b/p. Псевдокод ниже отображает правило, по которому вычисляется оценка.

```
a = minimal_pace
b = maximal_pace
p = actual_pace
if p < a:
  return p / a
elif p > b:
  return b / p
return 1
```

Критерий, проверяющий, что в базе данных отсутствует нечёткая копия аудиозаписи. Предполагается, что этот критерий будет защищать от нечестных участников, которые загружают одну и ту же аудиозапись для нескольких тренировок. Если копия найдена, то оценка за критерий равна нулю, иначе единице. Параметры критерия включают в себя максимальную допустимую долю схожести, частоту и размер окна, используемый для поиска похожих частей.

Критерии, проверяющие использование слов-паразитов. Данный критерий был реализован одним из первых, поскольку слова-паразиты не несут смысловой нагрузки и мешают воспринимать доклад. Первый критерий возвращает оценку, равную $1 - x/w$, где x – количество использованных слов-паразитов, а w – общее количество слов в речи. Второй проверяет, правда ли, что количество использованных слов-паразитов не превысило заданной границы, являющейся параметром. Общий параметр для обоих критериев – набор слов, считающихся паразитами. По умолчанию этот список частично взят из [28].

Для добавления нового критерия нужно реализовать наследника класса Criterion, код которого представлен ниже. То есть достаточ-

но реализовать конструктор, метод возвращающий текстовое, понятное человеку, описание критерия и метод, непосредственно вычисляющий критерий. Исходный код основных методов класса Criterion представлен в приложении 1. На рис. 2 представлена UML-диаграмма этого класса.

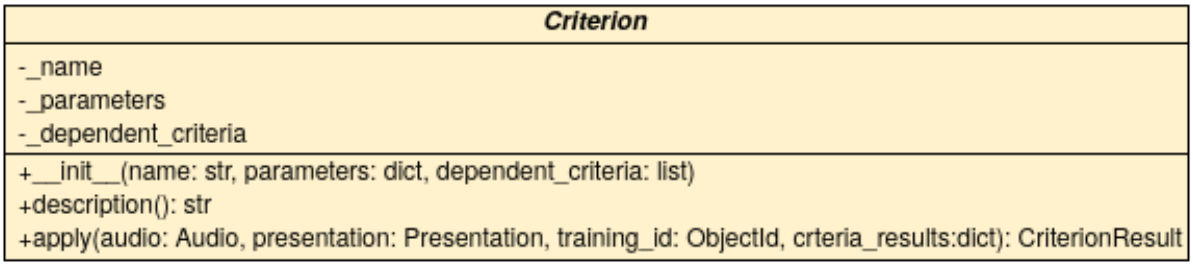

Рис. 2: UML-диаграмма класса Criterion

Класс CriterionResult хранит два поля: поле с дробным числом, являющимся результатом критерия и опциональным текстовым полем с описанием причины, почему было принято то или иное решение насчёт результата. В классе доступен конструктор, а также методы сериализации для строки и (де)сериализации для словаря.

2.3. Архитектура приложения

Проект состоит из нескольких сервисов. Сервис, с которым пользователь взаимодействует непосредственно, – это веб-сервис, представляющий собой приложение, написанное на языке Python с использованием фреймворка Flask [4]. В этом приложении пользователь может выгрузить свою презентацию, включить запись речи и начать тренировку, проговаривая доклад и переключая слайды. После окончания тренировки происходит обработка аудиозаписи и презентации, после чего пользователь получает результаты тренировки, статистику и общая обратная связь. Для того, чтобы модули с различным временем

выполнения и уровнем сложности могут быть добавлены в систему как часть обработки тренировок, используется асинхронный подход, поэтому другие сервисы основаны на очередях. Далее перечислены сервисы, основанные на очередях, и их зоны ответственности.

- Распознавание аудиозаписей. Идентификаторы аудиозаписей, которые необходимо распознать, отправляются в этот сервис. Обработчики извлекают идентификаторы из очереди и отправляют записи в систему распознавания речи, которая возвращает файл, содержащий информацию о распознанных словах. Этот файл отправляется в файловое хранилище, а его идентификатор этого файла отправляется в очередь, предназначенную для обработки распознанной речи (описано ниже);

- Распознавание презентаций (парсинг). Этот сервис работает аналогично предыдущему, файлы презентаций распознаются с помощью библиотеки PyMuPDF [12].

- Обработка распознанной речи. В этот сервис отправляются файлы с информацией о распознанной аудиозаписи. Обработчики извлекают идентификаторы из файлов очереди и вызывают обработку файлов, например, разделение списка слов на слайды и вычисление статистики как для каждого слайда, так и для всего файла.

- Обработка распознанных презентаций. Работает аналогично предыдущему сервису, но обрабатываются файлы с информацией о распознанных презентациях.

- Обработка тренировок. На обработанных данных запускается

набор критериев, после чего возвращается результат оценивания.

- Передача результатов оценивая в LMS через LTI.

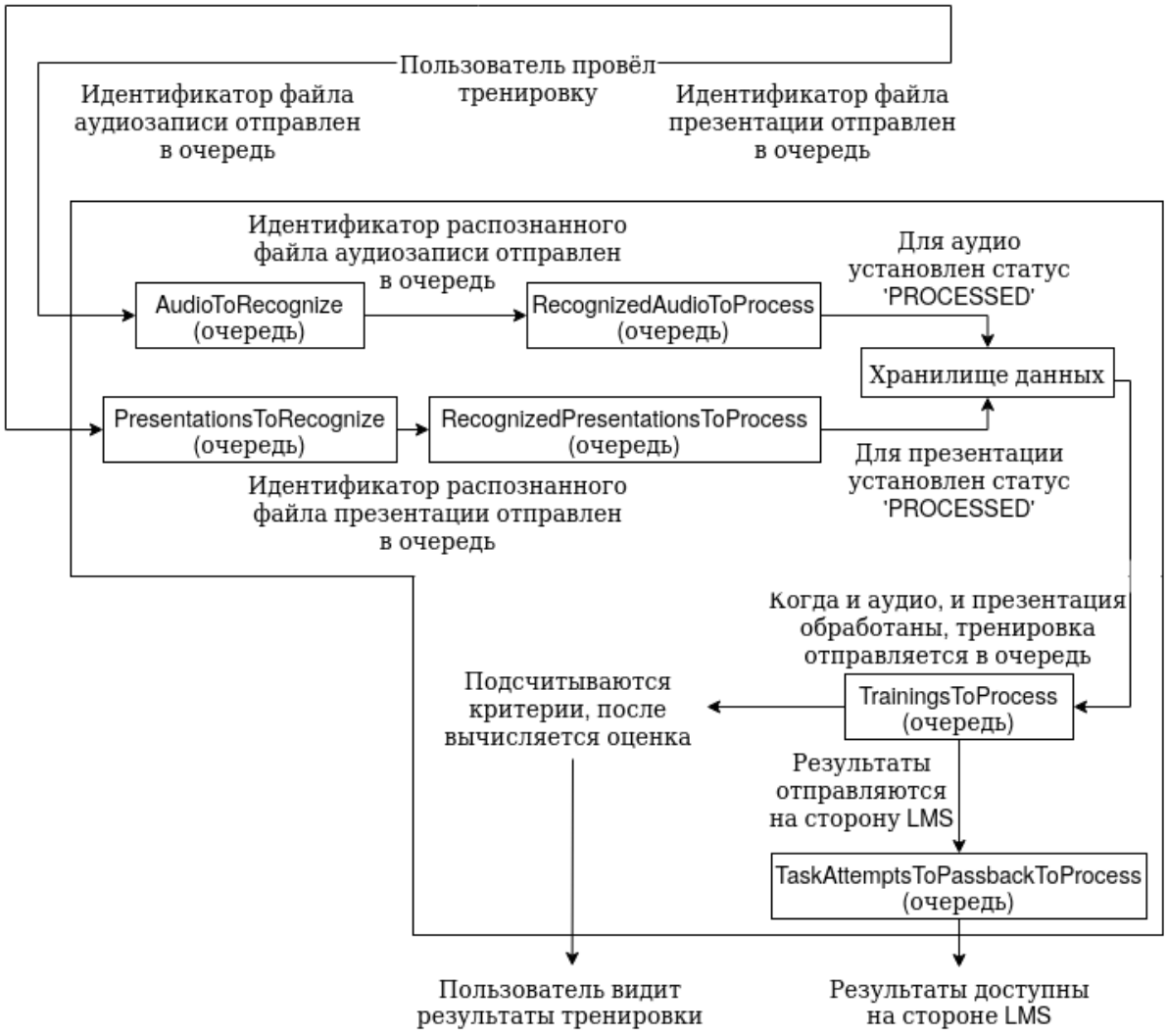

Рис. 3: Последовательность обработки данных

Ещё одним сервисом является сервисом хранения данных. Этот сервис поддерживает базу данных и файловое хранилище, по сути все запросы на чтение и запись данных обрабатываются этим сервисом. Кроме того, этот сервис отвечает за пополнение очередей, из которых обработчики извлекают данные. Например, когда для тренировки и аудиозапись, и презентация обработаны, то тренировка, содержащая

данную аудиозапись и данную презентацию, отправляется в очередь сервиса по обработке тренировок. Общая последовательность обработки данных представлена на рис. 3.

Каждый сервис работает в отдельном docker-контейнере. Все контейнеры управляются с помощью docker-compose.

Архитектура приложения показана на рис. 4. Каждый прямоугольник с закругленными углами представляет собой отдельный докер-контейнер, модули, представленные в виде серых прямоугольников, не принадлежат приложению.

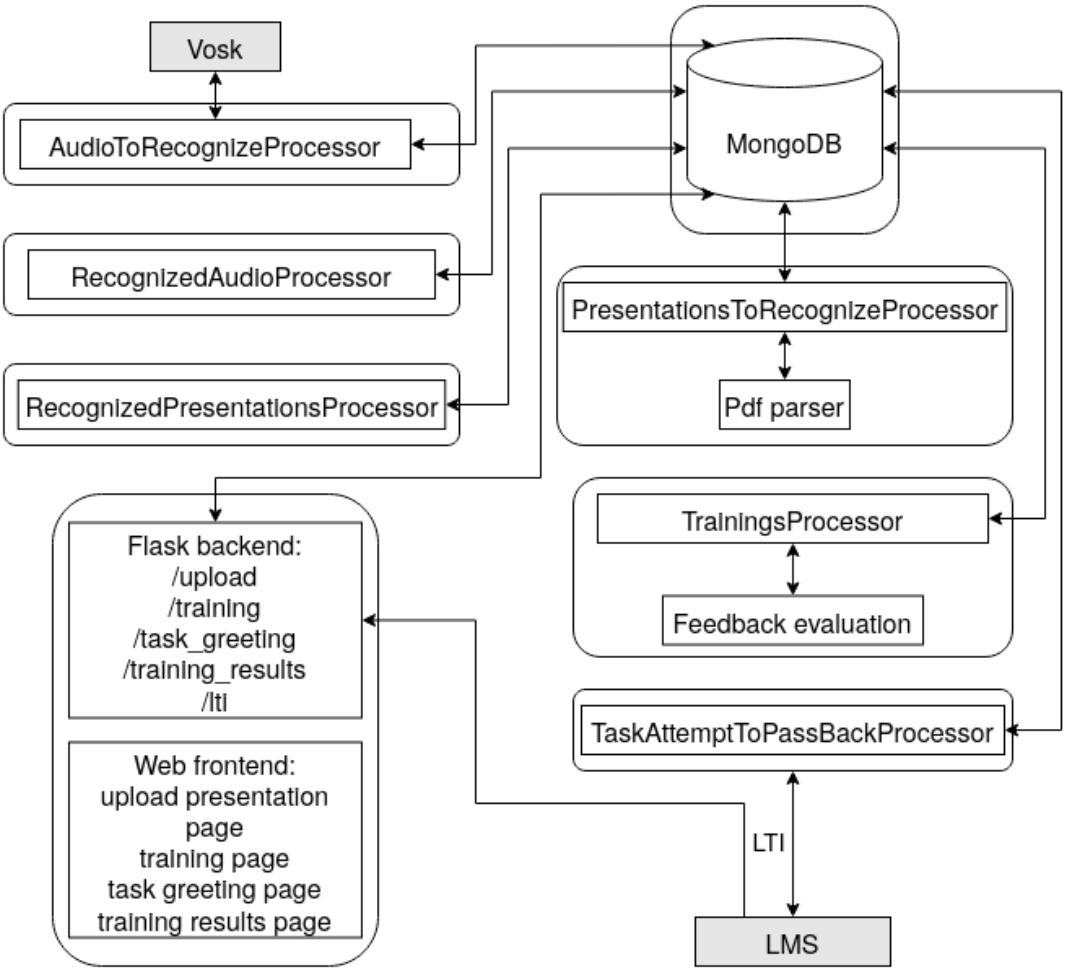

Рис. 4: Архитектура приложения

Отметим, что представленная архитектура горизонтально масштабируема за счёт возможности подключения нескольких обработ-

чиков к каждому сервису. Например, можно подключить несколько обработчиков, отвечающих за распознавание речи, в таком случае распознавание нескольких аудиозаписей, а значит и обработка нескольких тренировок будет вестись параллельно.

2.4. Модель данных

Для хранения данных используется СУБД MongoDB [8] в связке с файловым хранилищем GridFS. Данный выбор был сделан, потому что дальнейшая структура данных может измениться из-за новых способов обработки тренировок. NoSQL базы данных, такие как MongoDB, упрощают обработку, поскольку не требуют предопределенной структуры таблиц и связей между ними.

Основная коллекция - "Trainings". Поля этой коллекции включают в себя:

- идентификатор тренировки
- имя пользователя;
- идентификаторы файлов презентации (в трёх состояниях);
- идентификаторы файлов аудиозаписи (в трёх состояниях);
- статус обработки (презентации, аудиозаписи и тренировки в целом) и их временные метки последнего обновления;
- временные метки переключения слайдов;
- идентификатор набора критериев;
- идентификатор попытки задачи, в которую входит данная тренировки;

- идентификатор функции оценки;
- информация о результатах проверки, в частности общая оценка и оценка с обратной связью по каждому критерию.

Коллекция "PresentationFiles" хранит идентификаторы файлов презентации, имена файлов и идентификаторы файлов превью-файлов (англ. preview) презентаций. Превью-файлы используются для предварительный просмотра и представляют собой изображение первой страницы файла презентации.

Коллекция "Consumers" используется для хранения данных, связанных с различными потребителями LTI (например, системами управления обучением).

Коллекция "Sessions" используется для хранения данных, связанных с пользовательскими сессиями и включает в себя поля, хранящие идентификаторы сессий, ключи потребителей LTI, роли пользователей и информацию о задачах, доступных данному пользователю.

Коллекция "Tasks" хранит описания задач. Поля этой коллекции хранят идентификаторы задач, их описания, количество попыток, минимальное количество баллов, которое требуется набрать для успешного решения данной задачи, а также идентификатор

Коллекция "TaskAttempts" предназначена для хранения попыток и содержит поля для с информацией об имени пользователя, идентификаторе задачи, списке тренировок, которые были проведены в рамках данной попытки, а также набор параметров, которые используются для возврата оценки на сторону LMS. Остальные коллекции используются для возврата оценок.

Рис. 5: Схема базы даных

Коллекция ”Logs” содержит записи, сделанные в процессе журналирования. Поля этой коллеции содержат:

- временную отметку создания записи;
- имя сервиса, из которого пришла запись;
- уровень журналирования (DEBUG, INFO, WARNING, ERROR);
- имя файла, в котором произошла запись;
- название метода, в котором произошла запись;

- номер строки, в которой произошла запись;
- сообщение.

Остальные коллекции используются в качестве очередей в сервисах, описанных в разделе 2.2. На рис. 5 представлена схема базы данных.

2.5. Интеграция с внешними приложениями

Наличие интеграции с системами управлением обучением важно, поскольку, например, появляется возможность в автоматическом режиме агрегировать информацию об оценках в одном месте. Системы управления обучением используются в НИУ ВШЭ[1], СПбГЭТУ "ЛЭТИ"[2] и других образовательных организациях. Для приложения были реализованы поддержка протокола LTI и REST API [14].

Опишем кратко основные шаги, необходимые для поддержки протокола LTI. Во-первых, приложение и LMS должны знать общий секретный ключ, его нужно сгенерировать и сделать доступным для обеих систем. Во-вторых, должна быть создана задача в LMS. Входной точкой является POST HTTP-запрос, поступающий от LMS к приложению. Этот запрос содержит информацию о задаче и пользователе, а также данные, используемые для авторизации и для передачи оценки обратно на сторону LMS. Приложение валидирует запрос, в случае успеха создаётся пользовательская сессия, иначе запрос отклоняется.

Другим способом интеграции является реализация REST API. Внешние приложения могут использовать его для обращения и модификации данных. Кроме того, наличие REST API помогает отделить

[1]https://lms.hse.ru/
[2]http://e.moevm.info/

методы, возвращающие веб-страницы, от методов, используемых для самой логики приложения. Также на базе REST API можно создавать клиентские приложения для других платформ, например, смартфонов. Для отправки запроса клиентам необходимо выполнить HTTP-запрос по URL, который соответствует одной из сущностей модели данных либо компоненту одной из сущностей. Методы, реализованные и доступные на момент написания работы, описаны ниже.

Для сущности Trainings доступны следующие методы:

- PUT /api/trainings/timestamps/<training_id> – добавляет временную метку в список временных меток переключений слайдов;
- POST /api/trainings/presentations/<presentation_file_id>/ – создаёт тренировку на основе файла презентации с заданным идентификатором;
- GET /api/trainings/remaining-processing-time/<training_id>/ – вычисляет примерное время до конца обработки тренировки;
- POST /api/trainings/<training_id>/ – запускает обработку тренировки;
- DELETE /api/trainings/<training_id>/ – удаляет тренировку;
- GET /api/trainings/<training_id>/ – возвращает информацию о тренировке;
- GET /api/trainings/ – возвращает информацию обо всех тренировках.

Для сущности TaskAttempts доступны следующие методы:

- GET /api/task_attempts/current/ – возвращает информацию о текущей попытке;
- POST /api/task_attempts/<task_attempt_id>/ – создаёт попытку;
- DELETE /api/task_attempts/<task_attempt_id>/ – удаляет попытку;
- GET /api/task_attempts/<task_attempt_id>/ – возвращает информацию о попытке;
- GET /api/task_attempts/ – возвращает информацию обо всех попытках.

Для работы с файлами доступны следующие методы:

- POST /api/files/presentations/ – выгружает презентацию;
- GET /api/files/presentations/previews/<presentation_file_id>/ – возвращает превью файла презентации;
- GET /api/files/presentations/by-training/<training_id>/ – возвращает файл презентации по идентификатору тренировки;
- GET /api/files/presentations/<presentation_file_id>/ – возвращает файл презентации;
- POST /api/files/presentation-records/by-training/<training_id>/ – прикрепляет аудиозапись к тренировке;
- GET /api/files/presentation-records/by-training/<training_id>/ – возвращает файл аудиозаписи по идентификатору тренировки;

- GET /api/files/presentation-records/<presentation_record_file_id>/ – возвращает файл. аудиозаписи;

Также реализованы вспомогательные методы, возвращающие значение заданного параметра у критерия, информацию о пользовательском агенте и пользовательской сессии, а также записи, полученные в процессе журналирования (так к ним можно получить доступ на странице администратора, что упрощает поиск ошибок). Кроме того, доступен метод для взаимодействия по протоколу LTI.

Помимо описанной функциональности, во всех методах реализована проверка доступа с ресурсам, также возвращаются сообщения с описанием ошибки для некорректных запросов.

2.6. Выводы и результаты по главе

В данной главе приводится описание архитектуры приложения и способ обработки данных, предложенные с учётом цели данной работы. Предложенная архитектура является масштабируемой, производительность приложения может быть увеличена за счёт добавления обработчиков данных. Также архитектура является модульной, что позволяет поддерживать новые способы оценивания тренировок. За счёт разделения сущностей (”критерий”, ”параметризованный критерий”, ”набор критериев”, ”функция оценки”) приложение можно гибко настроить для оценки каждой отдельной тренировки в зависимости от предъявляемых к публичному выступлению требований. Также обеспечена интеграция с другими приложениями за счёт поддержки протокола LTI и реализации REST API.

3. Апробация

3.1. Исследование свойств решения

Чтобы понять, сколько ресурсов потребляется при обработке тренировок, в частности в случае одновременной обработки нескольких тренировок, были взяты 8 презентаций и 8 аудиозаписей выступления. Эти данные были взяты из выступлений студентов бакалавриата, магистратуры и аспирантов факультета СПбШФМКН ВШЭ[1] и кафедры МО ЭВМ СПбГЭТУ "ЛЭТИ"[2]. Выступления были посвящены защитам выпускных квалификационных или курсовых работ или подготовки к ним. Использовалась следующая аппаратная конфигурация:

- Lenovo Ideapad L340-15API 81LW005BRU, 12 GB RAM, AMD 3200U 2,6 GHz;
- Ubuntu 18.04.

Использовался набор из критериев, проверяющих темп и скорость речи, а также использование слов-паразитов. В такой конфигурации самым длинным этапом является распознавание аудиозаписи, т.к. анализ презентации и вычисление критериев занимает около секунды, поэтому для простоты можно считать, что общее время обработки тренировки почти равно времени распознавания аудиозаписи.

Результаты измерения распознавания аудиозаписей и распознавания презентаций показаны в таблицах 4 и таблице 5 соответственно. В каждом случае время обработки не превышает половины длительности аудиозаписи.

[1] https://spb.hse.ru/fmcs

[2] https://etu.ru/ru/fakultety/fkti/sostav/kafedra-moevm/

Потребление оперативной памяти оценивается как 2.1 гигабайта на каждый экземпляр распознавания речи Vosk и связанную с ним очередь плюс около 300 мегабайт для других сервисов, описанных в в предыдущей главе. Чтобы оценить, сколько времени потребуется для обработки нескольких тренировок, было взято 12 копий одной и той же тренировки, которая длится 7 минут и 39 секунд. Поскольку самая долгая часть обработки – это распознавание аудиозаписи, были рассмотрены конфигурации с разным количеством экземпляров системы распознавания речи. Результаты измерений представлены в таблице 6. Чем больше экземпляров системы распознавания речи, тем быстрее выполняется обработка и тем большая часть тренировки обрабатывается за секунду работы приложения.

Таблица 4: Длительность распознавания аудиозаписей

Длительность аудиозаписи, с	Длительность распознавания, с	Стандартное отклонение, с
436	139.74	0.42
459	144.09	0.49
492	155.40	0.63
541	154.87	0.51
568	168.77	0.68
614	226.37	0.82
810	232.43	3.14
1151	324.33	5.55

Была создана задача в LMS Moodle, используемой в СПбГЭТУ "ЛЭТИ". Задача была создана для подготовки к защите выпускной квалификационной работы. Студенту нужно выполнить три тренировки и набрать хотя бы 1.5 из 3 максимально возможных. После каждой выполненной тренировки оценка за неё отправляется на сторону LMS и становится доступной в общей таблице результатов. На момент написания работы, тренажёром воспользовалось примерно 40

студентов, в течение мая-июня 2021 года ожидается ещё примерно 60 пользователей.

Таблица 5: Длительность распознавания презентаций, с

Количество слайдов	Длительность распознавания, с	Стандартное отклонение, с
9	1.29	0.12
10	1.52	0.03
10	1.55	0.09
11	1.50	0.12
15	1.97	0.12
17	2.48	0.12
18	2.59	0.10
20	2.73	0.04

Таблица 6: Длительность обработки тренировок

Количество экземпляров системы распознавания речи	Суммарное время обработки, мм:сс	Длительность тренировки, обработанная за 1с работы приложения, с
1	25:31	3.60
2	17:47	5.16
3	14:31	6.32

3.2. Выводы и результаты по главе

В данной главе представлено исследование производительности приложения при различной нагрузке, описан способ использования приложения для подготовки к защите выпускной квалификационной работы.

Заключение

Главным результатом данной работы является разработанное веб-приложение, исходный код котрого доступен по ссылке[1], для первичной оценки качества публичного выступления. В отличие от существующих решений, данное приложение позволяет настраивать критерии, по которым будет оцениваться публичное выступление, имеет интеграцию с LMS по протоколу LTI и REST API для интеграции с внешними приложениями.

В рамках данной работы:

- проведён обзор статей и приложений, посвящённых автоматической оценке качества публичных выступлений, отмечено, что практически во существующих работах отсутствует возможность интеграции с внешними системами и гибкой настройки критериев, влияющих на итоговую оценку;

- проведены обзор и сравнение библиотек распознавания речи на собранном наборе данных, содержащем аудиозаписи публичных выступлений на русском языке, библиотека Vosk выбрана как наиболее подходящая в рамках данной работы;

- предложена модульная, гибкая и расширяемая архитектура веб-приложения для первичной оценки качества публичного выступления за счёт деления системы на сервисы по зоне ответственности (обработка распознанных аудиозаписей, парсинг презентаций и т.д.), возможности горизонтального масштабирования (можно прикрепить различное, в зависимости от требуемой производительности, количество обработчиков данных) и

[1] https://github.com/OSLL/web_speech_trainer

предоставления интерфейсов для каждого компонента оценивания тренировки (критерий, параметризованный критерий, набор критериев, функция оценки и т.д.);

- реализовано веб-приложение в соответствии с предложенной архитектурой, измерена производительность приложения при различных конфигурациях, ведётся апробация приложения на студентах выпускных курсов кафедры МО ЭВМ СПбГЭТУ "ЛЭТИ";
- реализована поддержка протокола LTI и проверена при интеграции приложения с LMS Moodle, реализован REST API.

Дальнейшая работа возможна в нескольких направлениях:

- интеграция и поддержка новых критериев, в том числе использующих методы обработки естественного языка и машинного обучения;
- добавление функциональности и пользовательского интерфейса для проверяющего: возможность прослушать выступление с автоматически переключаемыми слайдами в те моменты, когда это делал докладчик, возможность оставлять комментарии и изменять оценку.

Список литературы

[1] Aksent App - Pronunciation with Artificial Intelligence. — 2020. — Режим доступа: https://play.google.com/store/apps/details?id=com.eapp.pc.pro (дата обращения: 21.10.2020).

[2] Automatic Estimation of Presentation Skills Using Speech, Slides and Gestures / Abualsoud Hanani, Mohammad Al-Amleh, Waseem Bazbus, Saleem Salameh // Speech and Computer. — Springer International Publishing, 2017. — P. 182–191.

[3] ELSA Speak: Online English Learning Practice App - Apps on Google Play. — 2020. — Режим доступа: https://play.google.com/store/apps/details?id=us.nobarriers.elsa (дата обращения: 21.10.2020).

[4] Grinberg Miguel. Flask Web Development: Developing Web Applications with Python. — O'Reilly Media, Inc., 2014.

[5] Learning Tools Interoperability Core Specification 1.3 | IMS Global Learning Consortium. — 2021. — Режим доступа: https://www.imsglobal.org/spec/lti/v1p3 (дата обращения: 07.03.2021).

[6] LikeSo on the App Store. — 2020. — Режим доступа: https://apps.apple.com/us/app/likeso/id1074943747/ (дата обращения: 21.10.2020).

[7] Manning Christopher D., Raghavan Prabhakar, Schütze Hinrich. Introduction to Information Retrieval. — Cambridge University Press, 2008.

[8] MongoDB in Action / Kyle Banker, Peter Bakkum, Shaun Verch et al. — Manning Publications Co, 2016.

[9] Orai - Improve Public Speaking on the App Store. — 2020. — Режим доступа: https://apps.apple.com/us/app/orai-improve-public-speaking/id1203178170 (дата обращения: 21.10.2020).

[10] Picovoice: Edge Voice AI Platform. — 2020. — Режим доступа: https://picovoice.ai/ (дата обращения: 13.10.2020).

[11] Presentation Trainer, your Public Speaking Multimodal Coach / Jan Schneider, Dirk Börner, Peter van Rosmalen, Marcus Specht // Proceedings of the 2015 ACM on International Conference on Multimodal Interaction. — 2015.

[12] PyMuPDF Documentation — PyMuPDF 1.18.13 documentation. — 2021. — Режим доступа: https://pymupdf.readthedocs.io/en/latest/ (дата обращения: 10.03.2021).

[13] A Real-time Feedback System for Presentation Skills / Stephan Kopf, Daniel Schön, Benjamin Guthier et al. // Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications (EdMedia). — Association for the Advancement of Computing in Education, 2015. — P. 1633–1640.

[14] Richardson Leonard, Ruby Sam. RESTful Web Services. — O'Reilly Media, Inc., 2008.

[15] SergeyShk/Speech-to-Text-Russian. — 2020. — Режим доступа: https://github.com/SergeyShk/Speech-to-Text-Russian (дата обращения: 13.10.2020).

[16] Speakit: Be a native English speaker now - Apps on Google Play. — 2020. — Режим доступа: https://aksent.ai/ (дата обращения: 21.10.2020).

[17] Speech service documentation - Tutorials, API Reference - Azure Cognitive Services - Azure Cognitive Services. — 2021. — Режим доступа: https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/ (дата обращения: 10.03.2021).

[18] Speech-to-Text: Automatic Speech Recognition | Google Cloud. — 2020. — Режим доступа: https://cloud.google.com/speech-to-text (дата обращения: 13.10.2020).

[19] Speech to text online, Mac, Windows and Linux integration. — 2020. — Режим доступа: https://voicenotebook.com/ (дата обращения: 21.10.2020).

[20] Speechace | Pronunciation and fluency assessment via speech recognition. — 2020. — Режим доступа: https://www.speechace.com/#api (дата обращения: 21.10.2020).

[21] Speeko - the 1 public speaking app. — 2020. — Режим доступа: https://www.speeko.co/ (дата обращения: 21.10.2020).

[22] VOSK Offline Speech Recognition API. — 2020. — Режим доступа: https://alphacephei.com/vosk/ (дата обращения: 13.10.2020).

[23] Watson Speech to Text - Overview. — 2020. — Режим доступа: https://www.ibm.com/cloud/watson-speech-to-text (дата обращения: 13.10.2020).

[24] Web Speech API. — 2020. — Режим доступа: https://wicg.github.io/speech-api/ (дата обращения: 13.10.2020).

[25] Wit.ai. — 2020. — Режим доступа: https://wit.ai/ (дата обращения: 13.10.2020).

[26] Zhang Zhengyou. Microsoft Kinect Sensor and Its Effect // IEEE MultiMedia. — 2012. — Vol. 19, no. 2. — P. 4–10.

[27] at16k – Speech to text. — 2020. — Режим доступа: https://at16k.com/ (дата обращения: 13.10.2020).

[28] Малов Е. М., Горбова Е. В. Дискурсивные слова в русской разговорной речи (на материале анализа спонтанной разговорной речи) // Труды первого междисциплинарного семинара "Анализ разговорной русской речи". — Федеральное государственное бюджетное учреждение науки Санкт-Петербургский институт информатики и автоматизации Российской академии наук, 2007. — С. 31–36.

[29] Приложения в Google Play – Говорилло Развитие речи. — 2020. — Режим доступа: https://play.google.com/store/apps/details?id=com.vsquad.projects.govorillo (дата обращения: 21.10.2020).

Приложения

Приложение 1

В данном приложении находится исходных код основных методов класса Criterion.

```
class Criterion:
  def __init__(self, name: str, parameters: dict, dependent_criteria: list):
    self._name = name
    self._parameters = parameters
    self._dependent_criteria = dependent_criteria

  @property
  def description(self) -> str:
    raise NotImplementedError

  def apply(self,
            audio: Audio,
            presentation: Presentation,
            training_id: ObjectId,
            criteria_results: dict) -> CriterionResult:
    raise NotImplementedError
```