# 

УТВЕРЖДЕН

RU.17701729.07.04-01 ТЗ 01-1-ЛУ

|  |  |
| --- | --- |
| ***Подп. и дата*** |  |
|  |
| ***Инв. № дубл.*** |  |
|  |
| ***Взам. инв. №*** |  |
|  |
| ***Подп. и дата*** |  |
|  |
| ***Инв. №***    ***подл*** |  |
|  |

**РАЗРАБОТКА MVP СЕРВИСА СИНТЕЗА РЕЧИ НА АНГЛИЙСКОМ ЯЗЫКЕ**

**Техническое задание**

**RU.17701729.07.04-01 ТЗ 01-1**

**Листов 23**

Москва 2022

ОГЛАВЛЕНИЕ

[АННОТАЦИЯ 4](#_Toc103862705)

[ГОЛОССАРИЙ 5](#_Toc103862706)

[1. ВВЕДЕНИЕ 6](#_Toc103862707)

[1.1 Наименование программы 6](#_Toc103862708)

[1.2 Краткая характеристика области применения программы 6](#_Toc103862709)

[2. ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ 7](#_Toc103862710)

[3. НАЗНАЧЕНИЕ РАЗРАБОТКИ 8](#_Toc103862711)

[3.1 Функциональное назначение 8](#_Toc103862712)

[3.2 Эксплуатационное назначение 8](#_Toc103862713)

[4. ТРЕБОВАНИЯ К ПРОГРАММЕ 9](#_Toc103862714)

[4.1 Требования к функциональным характеристикам 9](#_Toc103862715)

[4.1.1 Требования к Acoustic Service 10](#_Toc103862716)

[4.1.2 Требования к Telegram bot service 10](#_Toc103862717)

[4.1.3 Требование к организации взаимодействия сервисов 11](#_Toc103862718)

[4.1.4 Требование к взаимодействию клиентской и серверной частей 11](#_Toc103862719)

[4.1.5 Требования к клиентской части 11](#_Toc103862720)

[4.2 Требования к надежности 12](#_Toc103862721)

[4.2.1 Требования к обеспечению надежного (устойчивого) функционирования программы 12](#_Toc103862722)

[4.2.2 Время восстановления после отказа 12](#_Toc103862723)

[4.2.3 Отказы из-за некорректных действий оператора 12](#_Toc103862724)

[4.3 Условия эксплуатации 12](#_Toc103862725)

[4.3.1 Климатические условия эксплуатации 12](#_Toc103862726)

[4.3.2 Требования к видам обслуживания 12](#_Toc103862727)

[4.3.3 Требования к численности и квалификации персонала 12](#_Toc103862728)

[4.4 Требования к составу и параметрам технических средств 12](#_Toc103862729)

[4.5 Требования к информационной и программной совместимости 12](#_Toc103862730)

[4.5.1 Требования к исходным кодам и языкам программирования 12](#_Toc103862731)

[4.5.2 Требования к программным средствам, используемым программой 12](#_Toc103862732)

[4.6 Требования к маркировке и упаковке 13](#_Toc103862733)

[4.7 Требования к транспортированию и хранению 13](#_Toc103862734)

[5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ 14](#_Toc103862735)

[5.1 Состав программной документации 14](#_Toc103862736)

[5.2 Специальные требования к программной документации 14](#_Toc103862737)

[6. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 15](#_Toc103862738)

[7. СТАДИИ И ЭТАПЫ РАЗРАБОТКИ 16](#_Toc103862739)

[7.1 Стадии разработки 16](#_Toc103862740)

[7.2 Сроки разработки и исполнители 17](#_Toc103862741)

[8. ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ 18](#_Toc103862742)

[8.1 Виды испытаний 18](#_Toc103862743)

[8.2 Общие требования к приемке работы 18](#_Toc103862744)

[9. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 19](#_Toc103862745)

[ПРИЛОЖЕНИЕ 1. СХЕМА АРХИТЕКТУРЫ ПРОГРАММЫ 21](#_Toc103862746)

[ПРИЛОЖЕНИЕ 2. ПРИМЕР МЕЛ-СПЕКТРОГРАММЫ 22](#_Toc103862747)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 23](#_Toc103862748)

# АННОТАЦИЯ

Техническое задание – это основной документ, оговаривающий набор требований и порядок создания программного продукта, в соответствии с которым производится разработка программы, ее тестирование и приемка.

Настоящее Техническое задание на разработку «Разработка MVP сервиса синтеза речи на английском языке» содержит следующие разделы: «Введение», «Основание для разработки», «Назначение разработки», «Требования к программе», «Требования к программным документам», «Технико-экономические показатели», «Стадии и этапы разработки», «Порядок контроля и приемки» и приложения.

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

В разделе «Основания для разработки» указан документ, на основании которого ведется разработка.

В разделе «Назначение разработки» указано функциональное и эксплуатационное назначение программного продукта.

Раздел «Требования к программе» содержит основные требования к функциональным характеристикам, к надежности, к условиям эксплуатации, к составу и параметрам технических средств, к информационной и программной совместимости, к маркировке и упаковке, к транспортировке и хранению, а также специальные требования.

Раздел «Требования к программным документам» содержит предварительный состав программной документации и специальные требования к ней.

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

Раздел «Стадии и этапы разработки» содержит стадии разработки, этапы и содержание работ.

В разделе «Порядок контроля и приемки» указаны общие требования к приемке работы.

Настоящий документ разработан в соответствии с требованиями:

1)   ГОСТ 19.101-77 Виды программ и программных документов [1];

2)   ГОСТ 19.102-77 Стадии разработки [2];

3)   ГОСТ 19.103-77 Обозначения программ и программных документов [3];

4)   ГОСТ 19.104-78 Основные надписи [4];

5)   ГОСТ 19.105-78 Общие требования к программным документам [5];

6)   ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [6];

7)   ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению [7].

Изменения к данному Техническому заданию оформляются согласно ГОСТ 19.603-78 [13], ГОСТ 19.604-78 [14].

ГОЛОССАРИЙ

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| TTS (Text-To-Speech), синтез речи | Формирование речевого сигнала по текстовому представлению |
| MVP (Minimum Viable Product) | Начальная версия продукта, обладающая минимальным набором функций, достаточным для ее презентации и тестирования на первых потребителях. |
| Микросервисная архитектура | Подход к разработке ПО, при котором монолита системы разбивается на отдельные независимые компоненты - сервисы. |
| Семиотический класс | Множество лексических единиц, которые обозначают объекты одного вида и имеют схожие правила написания и произношения. |
| Нормализация текста | Процесс, при котором текст разбивается на лексические единицы, каждая из которых соотносится с определенным семиотическим классом. |
| NLP (Natural Language Processing) | Обработка естественного языка. |
| Фонема | Единица звукового строя языка. |
| Mel шкала | Шкала, основанная на восприятии тонов (частот) человеком. Слушатели оценивают любые частоты на расстоянии 1 мел, как равные по расстоянию друг от друга. |
| Мел-спектрограмма | Спектрограмма, в которой частоты преобразуются в шкалу mel (Приложение 2) График, в котором по оси x показано время, по оси y – частота. Цветом выделяется амплитуда (Дб). |
| Вокодер (vocoder) | Устройство (или алгоритм), которое синтезирует речь на основе некоторой информации о ней. |
| Акустическая модель | Модель, которая принимает в качестве входных данных признаки на небольшом участке акустического сигнала (фрейме) и подает на выход распределение вероятностей различных фонем на этом фрейме. |
| GPU (graphics processing unit) | Специализированный вид микропроцессора |
| CUDA | Программно-аппаратная архитектура вычислительных процессов, которая использует процессоры Nvidia |

1. ВВЕДЕНИЕ
   1. Наименование программы

Наименование программы на русском языке: «Разработка MVP сервиса синтеза речи на английском языке».

Наименование программы на английском языке: «Development of an MVP service for speech synthesis in English».

* 1. Краткая характеристика области применения программы

Синтезом речи называется процесс, в ходе которого из текста на некотором языке создается аудиo, содержащего речь, информационное сообщение в которой носитель данного языка воспринимает как эквивалентное, передаваемому текстом. Синтез речи часто используется в системах с голосовым управлением, например, в голосовых помощниках.

В некоторых ситуациях возможно предварительно записать куски речи диктора и “склеивать” их при необходимости вывода информации в виде аудио. Такой подход является жизнеспособным в системах с ограниченном числом реплик, например, в навигаторах. Однако, в более сложных программах требуются синтезирование произвольных фраз, что делает описанный ранее подход невозможным. Данная программа использует NLP-подходы, что помогает решать описанные ограничения и позволяет превращать произвольные фразы на английском языкe в речь.

Данная программа является MVP проектом, представляющим Telegram бота, позволяющего синтезировать аудио файл с речью на английском языке по введенному пользователем тексту.

Данное техническое задание раскрывает детали разработки Acoustic Service и Telegram bot service.

1. ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ

Разработка ведется на основании учебного плана подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и утвержденной академическим руководителем программы «Разработка MVP сервиса синтеза речи на английском языке».

1. **НАЗНАЧЕНИЕ** РАЗРАБОТКИ
   1. Функциональное назначение

Программа предоставляет возможность получать аудиофайлы на основе предоставленного текста на английском языке. Кроме того, программа предоставляет определенному кругу пользователей возможность получения аналитики по работе сервисов, отвечающих за синтез речи и проведения нагрузочного тестирования на этих сервисах.

* 1. Эксплуатационное назначение

Программа является MVP сервиса синтеза речи на английском языке, позволяющим облегчить процесс преобразования текстовых данных в аудио-формат. Каждый пользователь данной программы, взаимодействуя с ботом, может создавать аудиофайлы на основе произвольного текста на английском языке, переданного программе.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ
   1. Требования к функциональным характеристикам

В рамках командной работы необходимо реализовать МVP сервис для синтеза речи на английском языке. Программа имеет микросервисную архитектуру. В ее состав входят следующие микросервисы: Normalization Service (Нормализация текста), Acoustic Service (Генерация мел-спектрограмм по нормализованному тексту), Vocoder Service (Генерация аудио по мел-спектрограммам), Dispatcher Service (Обеспечивает взаимодействие описанных ранее сервисов), Telegram Bot Service (Telegram bot, обеспечивающий взаимодействие пользователя и сервисов синтеза речи, предоставляет информацию о работе программы разработчикам).

Таким образом, схему работы конечного программного продукта (Приложение 1) можно описать следующим образом. Текст сообщения на английском языке, отправленного пользователем боту на платформе мессенджера Telegram, передается в качестве параметра http запроса Telegram Bot сервисом Dispatcher сервису. Dispatcher Service в свою очередь передает текст в параметрах http запроса к Acoustic Service и получает от последнего ответ в формате json, cсодержащий нормализованный текст. Затем Dispatcher Service передает нормализованный текст в параметрах http-запроса к Acoustic Service, от которого получает json ответ, содержащий мел-спектрограмму. Полученную мел-спектрограмму при помощи http-запроса передается Dispatcher сервисом Vocoder сервису, который возвращает json ответ, содержащий аудио сообщение в формате wav. Данный ответ Dispatcher Service возвращает Telegram Bot сервису, который в свою очередь отправляет через бота сообщение пользователю, содержащее аудио файл в формате wav, озвучивающий исходное сообщение.

В процессе работы Dispatcher сервис кэширует полученные аудио (отправляет пользователю без повторной генерации ответ в случае, если такой текст уже был обработан ранее), и статистическую информации об обработке запроса.

Для разработчиков (заранее определенного круга пользователей бота) доступны также дополнительные команды бота, позволяющие получить сообщение о статистике, собранной Dispatcher service, а также возможность провести нагрузочное тестирование сервисов синтеза речи с заданными параметрами и получить сообщение об его результатах.

Подробное описание функционирования микросервисов представлено в индивидуальных ТЗ и отчетах участников команды в соответствии с распределением работы над проектом.

В рамках индивидуальных задач при реализации командного продукта необходимо:

1. Изучить информацию об open source реализации модели FastPitch [15] и архитектуре Transformer [16], на которой она основана;

2. Обучить модель FastPitch [18] на датасете LJSpeech-1.1 [20];

3. Реализовать Acoustic Service на основе обученной модели;

4. Реализовать Telegram Bot Service;

5. Обеспечить возможность запуска Acoustic Service и Telegram Bot Service в Docker container [17].

6. Обеспечить взаимодействие всех микросервисов MVP с помощью Docker Compose [19].

* + 1. Требования к Acoustic Service

1) Отвечает на GET запрос с параметром текста.

(вид запроса:

http://{адрес сервиса в локальной сети}:5000/?{текст+сообщения+пользователя} )

2) Полученный текст отправляется на обработку акустической модели. В сервисе используется Open-source реализации модели Fastpitch [15] (в основе архитектура Transformer [16]), заранее обученная на датасете LJSpeech-1.1 [20].

3) Ответ на запрос содержит сериализованную в формат json матрицу мел-спектрограммы.

* + 1. Требования к Telegram bot service

1. После начала диалога пользователем (команда пользователя ‘/start’) бот отправляет приветственное сообщение с предложением ввести текст на английском языке.
2. После ввода пользователем текстового сообщения бот отправляет http запрос к Dispatcher service с текстом, введенным пользователем, в качестве параметра.

(вид запроса:

http://{адрес Dispatcher service в локальной сети}:7000/?{текст+сообщения+пользователя})

1. Передаваемый в ответ аудиофайл в формате wav отправляется пользователю как аудиосообщение.
2. При возникновении ошибки синтеза речи на любом этапе или некорректном тексте исходного сообщения, пользователь информируется ответным сообщением об возникновении ошибки.
3. При вводе команды ‘/analytics’ или ‘/latency\_test {Количество параллельных пользователей} {время теста в секундах} ’ пользователем, чей id не включен в список id разработчиков, пользователю отправляется сообщение об ошибке.
4. При вводе команды ‘/analytics’ пользователем, чей id включен в список разработчиков, отправляется http запрос к Dispatcher service вида:

http://{адрес Dispatcher service в локальной сети}:7000/analytics.

1. Передаваемый в качестве ответа словарь в формате json со статистикой синтеза речи, представляется в виде текста и изображений и отправляется пользователю в виде текстовых сообщений и картинок соответственно.
2. После ввода команды ‘/latency\_test {Количество параллельных пользователей} {время теста в секундах} ’ Бот выполняет в течение заданного времени нагрузочное тестирование с заданным количеством пользователей.
3. Нагрузочное тестирование выполняется параллельными ‘пользователями’ (параллельно работающими функциями), каждый из которых делает один запрос в секунду. Каждый запрос имеет небольшую стандартную длину и некоторый шум для избежания получения результата из кэшированных данных.
4. Измеренные во время нагрузочного тестирования характеристики отправляются пользователю, который ввел соответствующую команду, в сообщения с текстом и изображения с графиком.
5. Отчет о нагрузочном тестировании включает в себя характеристики длительности выполнения запроса.
   * 1. Требование к организации взаимодействия сервисов

Каждый микросервис запускается в отдельном Docker контейнере Инструкции для программы docker [17] представлены в отдельных для каждого микросервиса файлах Dockerfile.

Инструкции для отдельного запуска и взаимодействия с микросервисами описаны в файлах ReadMe.md.

Также для запуска всех микросервисов при помощи команды ‘docker-compose up –build’ из корня MVP прописаны инструкции программе docker-compose [19] в файле docker-compose.yml. При запуске данной команды каждый микросервис запускается в отдельном контейнере, при этом контейнеры объединяются в локальную сеть, что обеспечивает взаимодействие сервисов.

* + 1. Требование к взаимодействию клиентской и серверной частей

Взаимодействие между клиентской и серверной частями должно осуществляться с помощью Telegram API.

* + 1. Требования к клиентской части

Клиентская часть приложения представлена чат ботом на в мессенджере «Telegram».

1. Начать диалог с ботом можно при помощи команды ‘/start’
2. Обычному пользователю после начала диалога должно быть выведено приветственное сообщение с предложением ввести текст на английском языке.
3. На каждое введенное сообщение на английском языке бот присылает аудиосообщение с синтезированной по введённому тексту речью или сообщение об ошибке в случае, если процесс завершился некорректно.
4. Для разработчиков (пользователей, чей id входит в ограниченный список) доступна команда ‘/analytics’. После получения данной команды пользователь получает сообщения с аналитическими данными о работе сервисов синтеза речи, собираемыми dispatcher service.
5. Для разработчиков (пользователей, чей id входит в ограниченный список) доступна команда ‘/latency\_test {Количество параллельных пользователей} {время теста в секундах} ’. После получения данной команды пользователь получает сообщения с результатами проведенного нагрузочного тестирования.
6. Бот отправляет сообщения пользователю на английском языке.
   1. Требования к надежности
      1. Требования к обеспечению надежного (устойчивого) функционирования программы

Программа не должна аварийно завершаться при любом наборе входных. Программа должна обеспечивать проверку корректности входных данных.

* + 1. Время восстановления после отказа

В случае отказа работы серверной части и последующей недоступности Telegram бота, время восстановления не должно превышать одни рабочие сутки.

* + 1. Отказы из-за некорректных действий оператора

После запуска программы на сервере отказ программы вследствие некорректных действий оператора должен быть исключён. В том числе должна быть исключена возможность непреднамеренного выключения программы, не связанного с техническими неполадками сервера.

* 1. Условия эксплуатации
     1. Климатические условия эксплуатации

Требований к климатическим условиям эксплуатации не предъявляется.

* + 1. Требования к видам обслуживания

Обслуживание не требуется.

* + 1. Требования к численности и квалификации персонала

Не требует специального обслуживания. Требуемая классификация – пользователь, свободно владеющий английским языком, а также Интернетом.

* 1. Требования к составу и параметрам технических средств

Для надёжной работы программы требуется следующий состав технических средств. Для работы клиентского приложения:

Устройство, поддерживающее работу с Telegram.

Устойчивое интернет-соединение.

* 1. Требования к информационной и программной совместимости
     1. Требования к исходным кодам и языкам программирования

Программа реализована на языке программирования Python 3.6 или выше.

* + 1. Требования к программным средствам, используемым программой

Программа реализована на языке программирования Python 3.6

Программа запускается на операционной системе Ubuntu 20.04.4.

Программа должна использовать NVIDIA® Tesla® V100 в Yandex Cloud для синтеза аудио.

Для запуска необходимо наличие программ Docker и Docker Compose

Необходимо иметь зарегистрированного в телеграмме бота.

* 1. Требования к маркировке и упаковке

Требования к маркировке и упаковке не предъявляются.

* 1. Требования к транспортированию и хранению

Специальные требования к транспортировке не предъявляются.

1. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ
   1. Состав программной документации

Техническое задание (ГОСТ 19.201-78) [7];

Программа и методика испытаний (ГОСТ 19.301-79) [8];

Текст программы (ГОСТ 19.401-78) [9];

Пояснительная записка (ГОСТ 19.404-79) [10];

Руководство оператора (ГОСТ 19.505-79) [11].

* 1. Специальные требования к программной документации

Документы к программе должны быть выполнены в соответствии с ГОСТ 19.106-78 [6] и ГОСТами к каждому виду документа (см. п. 5.1.);

Пояснительная записка должна быть загружена в систему Антиплагиат через LMS «НИУ ВШЭ».

Техническое задание и пояснительная записка, титульные листы других документов должны быть подписаны руководителем разработки и исполнителем

Документация и программа сдается в электронном виде в формате .pdf или .docx. в архиве формата .zip или .rar;

За три дня до защиты комиссии все материалы курсового проекта:

программная документация,

программный проект,

исполняемый файл,

отзыв руководителя

отчет системы Антиплагиат

должны быть загружены одним или несколькими архивами в проект дисциплины «Курсовой проект» в личном кабинете в информационной образовательной среде SmartLMS НИУ ВШЭ.

1. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

Данная работа является MVP, цель которого продемонстрировать возможность синтеза речи из текста. Экономическая оценка данной программы и ее сравнение с полноценными системами синтеза речи не является целесообразной и не предусматривается в рамках данного проекта.

1. СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

7.1 Стадии разработки

1. Техническое задание:

1.1. Этапы разработки

– обоснование необходимости разработки программы;

– постановка задачи;

– сбор исходных материалов;

– выбор и обоснование критериев эффективности и качества разрабатываемой; программы;

1.2. Разработка и утверждение технического задания

– определение требований к программе;

– определение стадий, этапов и сроков разработки программы и документации на неё;

– согласование и утверждение технического задания.

2. Эскизный проект:

2.1. Разработка эскизного проекта

– предварительная разработка структуры входных и выходных данных;

– уточнение методов решения задачи;

– разработка общего описания алгоритма решения задачи.

2.2. Утверждение эскизного проекта

– разработка пояснительной записки;

– согласование и утверждение эскизного проекта.

3. Технический проект:

3.1. Разработка технического проекта

– уточнение структуры входных и выходных данных;

– разработка алгоритма решения задачи;

– определение формы представления входных и выходных данных;

– разработка структуры программы;

– окончательное определение конфигурации технических средств.

3.2. Утверждение технического проекта

– разработка пояснительной записки;

– согласование и утверждение технического проекта.

4. Рабочий проект:

4.1. Разработка программы

– программирование и отладка программы.

– обучение моделей

4.2. Разработка программной документации

– разработка программных документов в соответствии с требованиями гост 19.101-77. [1]

4.3. Испытания программы

– разработка, согласование и утверждение порядка и методики испытаний;

– корректировка программы и программной документации по результатам испытаний.

7.2 Сроки разработки и исполнители

Разработка должна закончиться к 15 мая 2022 года.

Исполнитель: Д. О. Соколова.

Состав команды курсового проета: Д. И. Калинина, А. Е. Ишутин, Д. О. Соколова.

|  |  |
| --- | --- |
| Исполнитель: | Часть проекта, за которую отвечает: |
| Д. И. Калинина | Исследование существующих датасетов на подготовительном этапе, анализ существующих моделей на этапе подготовки, Normalization Service. |
| Д. О. Соколова | Разработка Telegram Bot service, Acoustic Service, построение Docker compose, разработка программной документации. |
| А. Е. Ишутин | Разработка архитектуры программного продукта, настройка сервера, TTS Service, Vocoder Service, cash. |

1. ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ
   1. Виды испытаний

    Производится проверка корректной работы программы, а также проверка на соответствие требований технического задания. Проверка программы также на соответствие техническому заданию осуществляется заказчиком совместно с исполнителем согласно «Программе и методике испытаний» (ГОСТ 19.301-79) [8].:

- Перечень функций программы, выделенных в программе для испытаний, и перечень требований, которым должны соответствовать эти функции

- Перечень необходимой документации и требования к ней

- Методы испытаний и обработки информации

* 1. Общие требования к приемке работы

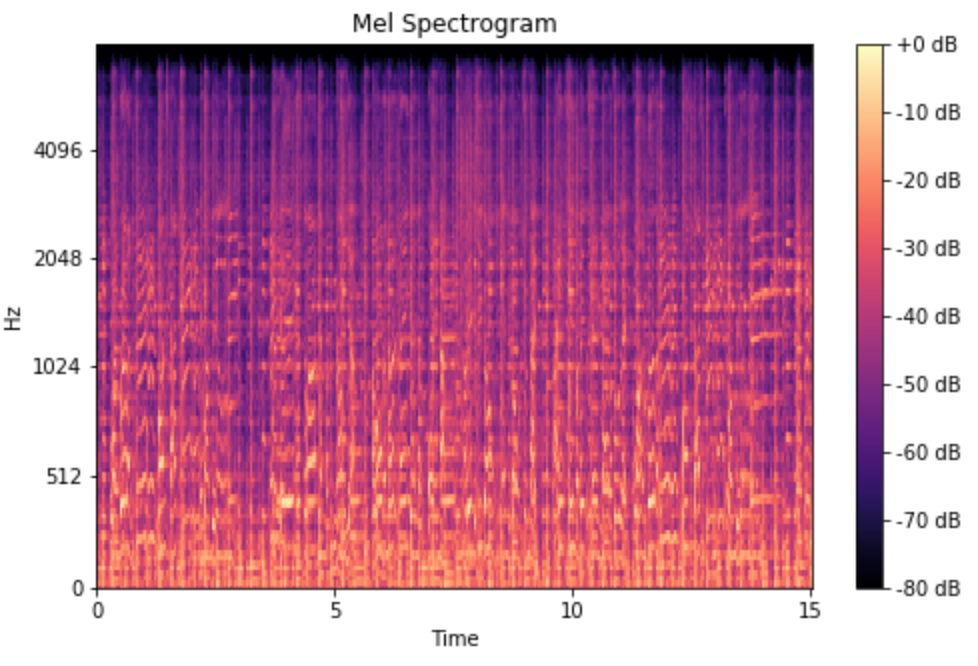
Прием программного продукта производится только при полной работоспособности Telegram бота, тестировании входных и выходных данных, соответствия технической документации, а также при наличии полной документации к программе.

1. **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**
   1. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
   2. ГОСТ 19.102-77 Стадии разработки. //Единая Система Программной Документации. – М.: ИПК Издательство стандартов, 2001.
   3. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая Система Программной Документации. – М.: ИПК Издательство стандартов, 2001.
   4. ГОСТ 19.104-78 Основные надписи. //Единая Система Программной Документации. – М.: ИПК Издательство стандартов, 2001.
   5. ГОСТ 19.105-78 Единая система программной документации. Общие требования к программным документам. //Единая Система Программной Документации. – М.: ИПК Издательство стандартов, 2001
   6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
   7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
   8. ГОСТ 19.301-79 Программа и методика испытаний. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
   9. ГОСТ 19.401-78 Текст программы. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
   10. ГОСТ 19.404-79 Пояснительная записка. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
   11. ГОСТ 19.505-79 Руководство оператора. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
   12. ГОСТ 19.602-78 Правила дублирования, учета и хранения программных документов, выполненных печатным способом. – М.: ИПК Издательство стандартов, 2001.
   13. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая Система Программной Документации. – М.: ИПК Издательство стандартов, 2001
   14. ГОСТ 19.604-78 Единая система программной документации. Правила внесения изменений в программные документы, выполненные печатным способом. //Единая Система Программной Документации. – М.: ИПК Издательство стандартов, 2001
   15. Adrian Łancucki // FastPitch: Parallel Text-to-speech with Pitch Prediction// NVIDIA Corporation //[Электронный ресурс]: Technical Report, 2021 – Режим доступа: https://arxiv.org/abs/2006.06873, свободный. (дата обращения: 1.05.22).
   16. Ashish Vaswani// Attention Is All You Need // Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin // [Электронный ресурс]: Technical Report, 2017 – Режим доступа: https://arxiv.org/abs/1706.03762, свободный. (дата обращения: 1.05.22).
   17. Docker overview [Электронный ресурс] /Documentation. Режим доступа: https://docs.docker.com/get-started/overview/ свободный. (дата обращения: 1.05.22)
   18. NVIDIA, FastPitch 1.1 for PyTorch [Электронный ресурс] / GitHub. Режим доступа: https://github.com/NVIDIA/DeepLearningExamples/tree/master/PyTorch/SpeechSynthesis/FastPitch, свободный. (дата обращения: 1.05.22)
   19. Overview of Docker Compose [Электронный ресурс] /Documentation. Режим доступа: https://docs.docker.com/compose/ свободный. (дата обращения: 1.05.22)
   20. The LJ Speech Dataset [Электронный ресурс] / Dataset. Режим доступа: https://keithito.com/LJ- Speech-Dataset/, свободный. (дата обращения: 1.05.22)

ПРИЛОЖЕНИЕ 1. СХЕМА АРХИТЕКТУРЫ ПРОГРАММЫ

Пользователь

ПРИЛОЖЕНИЕ 2. ПРИМЕР МЕЛ-СПЕКТРОГРАММЫ



# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

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