Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
(Университет ИТМО)

Факультет **Прикладной информатики**

Направление подготовки **09.03.03 Прикладная информатика**

Образовательная программа **Мобильные и сетевые технологии**

К У Р С О В О Й П Р О Е К Т

Тема: «Разработка сервиса локализации мобильного приложения»

Обучающийся: Скворцов Денис Александрович, K3140

Санкт-Петербург 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

1 Актуальность темы курсового проекта

Актуальность темы сервиса локализации мобильных приложений обусловлена несколькими ключевыми факторами.

Во-первых, у многих людей, не являющихся носителями языка, возникают трудности с использованием приложения, поскольку приходится тратить время на перевод непонятных терминов и отвлекаться от работы. Поскольку главной задачей локализации является упрощение взаимодействия с приложением, она была выбрана в качестве основного решения проблемы для всех подобных групп лиц. Например, одна из выборок, на которую было обращено внимание при разработке - это иностранные студенты.

Во-вторых, существующие сервисы локализации, находящиеся в открытом доступе требуют плату за свои услуги, которая может составлять значительную часть расходов для создателей приложения. Независимый сервис локализации, в свою очередь, помогает уменьшить затраты на перевод при обращении к третьим лицам и позволяет автоматизировать этот процесс, что экономит не только денежные, но и человеческие ресурсы.

В-третьих, создание собственного сервиса локализации позволяет настраивать его под собственные нужды или под нужды заказчиков. Так, результат будет качественнее, чем решение от усредненного сервиса, предназначенного для всего сразу.

Наконец, развитие отрасли машинного обучения и языковых моделей в частности упрощает задачу тренировки необходимой модели-переводчика. Это приводит к тому, что модель в основе сервиса может быть востребована в других областях, связанных с переводом, поскольку переводчик, фактически, является независимым продуктом.

Таким образом, сервис локализации мобильных приложений выступает качественной альтернативой другим средствам перевода, а также предоставляет возможности для расширения как в области приложений, так и в области переводчиков в целом.

2 Цель проекта

Целью проекта является разработка сервиса локализации мобильного приложения. В результате работы должен получить MVP, который можно будет в дальнейшем развивать как стартап.

3 Задачи проекта

- Обучить модели-переводчики;
- Разработать API для связи с моделью;
- Поднять сервер для приложения;
- Создать MVP

1 Суть проекта

Необходимо разработать сервис динамической локализации. На основе алгоритмов машинного обучения должна быть создана модель-переводчик, способная оперировать необходимыми языками. Так, на основе API сервис должен подключаться к модели и запрашивать перевод конкретных строк. Доступ к этому сервису предоставляется разработчикам мобильных приложений, которые смогут адаптировать свои продукты под пользователей других стран. Сервис позволит разработчикам полностью локализировать свои приложения автоматически без лишних трат на собственноручный перевод.

Проект заточен на упрощение работы и на повышение качества перевода в каждом конкретном случае, что в долгосрочной перспективе может дать пользователям прирост аудитории из других стран.

2 Процессы работы над всем проектом

2.1 Организационная часть

В связи с особенностями проекта и объёмом работы было выделено три ключевых области задач: Backend, Frontend и ML.

Подготовкой Backend части, занимались Тимур Толкачёв и Бахадыр Ахмедов. Их основным задачами были поднятие и организация серверной части, обработкой запросов и поддержка базы данных и API.

Разработка Frontend части лежала на Алексее Данилевском. Он отвечал за создание внешнего вида приложения, связь с Backend'ом и презентацию минимально жизнеспособного продукта.

В области ML участвовал я вместе с Марком Калининым. Наша работа сфокусировалась на обучении модели-переводчика и интеграции её с сервером Backend части.

Ввиду специфики проекта у нас было три глобальные задачи: подготовить Backend, который поддерживает API (этим занимались Тимур Толкачёв и Бахадыр Ахмедов), Frontend (этим занимался Алексей Данилевский) и обучить модель для перевода (эта задача была возложена на меня и Марка Калинина). Периодически мы созванивались для синхронизации нашей работы, обсуждения промежуточных итогов, получения обратной связи от куратора: в среднем мы это делали 1-2 раза в неделю.

Во время выполнения проекта было несколько организационных этапов, в которых мы определялись с характеристиками сервиса. Так, основными вопросами были: выбор языков для перевода, моделей и данных для их обучения, вид финального продукта.

2.2 Основные языки

Для начальной разработки и получения хоть каких-то результатов, было решено выбрать целевым языком английский, так как он является международным, и информации на этом языке в интернете больше, чем на любом другом. Именно с него далее будет происходить на все остальные языки.

Критериями для выбора языка были его популярность и знания в этом языке для дополнительной проверки перевода. По итогам было выбрано ещё 5 языков: русский, французский, немецкий, китайский и турецкий. Именно для этих вариантов существовали хорошие параллельные корпуса (датасеты с фразами и их переводом на разные языки) в паре с английским.

2.3 Сторонние ресурсы

Значительная часть работы не может быть сделана без использования дополнительных источников данных и вычислительных ресурсов, поэтому было решено воспользоваться услугами сторонних сервисов.

Для задачи перевода использовались предобученные модели T5-small от Google и Marian от Helsinki-NLP. Обучение моделей происходило в среде Google Collab на основе системы Jupiter Notebook. Там, нам были предоставлены бесплатные мощности GPU, которых сильно не хватало, чтобы быстро проверять результаты обучения, вследствие чего была куплена подписка для увеличения производительности.

Также были использованы датасеты, находящиеся в открытом доступе на сайте Hugging Face: Opus Books (английский-французский) [1], KazParC (английский-турецкий, английский-русский) [2], WMT14 (английский-немецкий) [3] и Wlhb/Translation-

Chinese-2-English (английский-китайский) [4]. Помимо этого, был сгенерирован собственный параллельный корпус, полученный переводом через DeepL базы ISBNDB с данными о книгах.

Для Backend’а был арендован удалённый сервер по подписке Amazon EC2. Для хранения изображений (обложек книг) дополнительно был использован Amazon S3.

2.4 Обучение моделей

Основой проекта является модель-переводчик, тренировка которой происходила при помощи алгоритмов машинного обучения. Были использованы вышеупомянутые параллельные корпуса, сервисы и предобученные модели,

чтобы, впоследствии, получить модель, заточенную под конкретную задачу перевода названий и описаний книг.

2.5 Серверная часть

Основными задачами Backend'а проекта были взаимодействие с БД, моделью и API. Для взаимодействия с базой данных, моделью при помощи API Backend использовал библиотеки SQLAlchemy с psycopg2, fastAPI, uvicorn, PyTorch. Docker-контейнер с размещённой в нём программой запущен на арендованном сервере Amazon, проиллюстрированном на Рисунке 1. Помимо этого создан API для связи с моделями. При помощи него происходит обмен данными для перевода в формате JSON через виртуальный выделенный сервер.

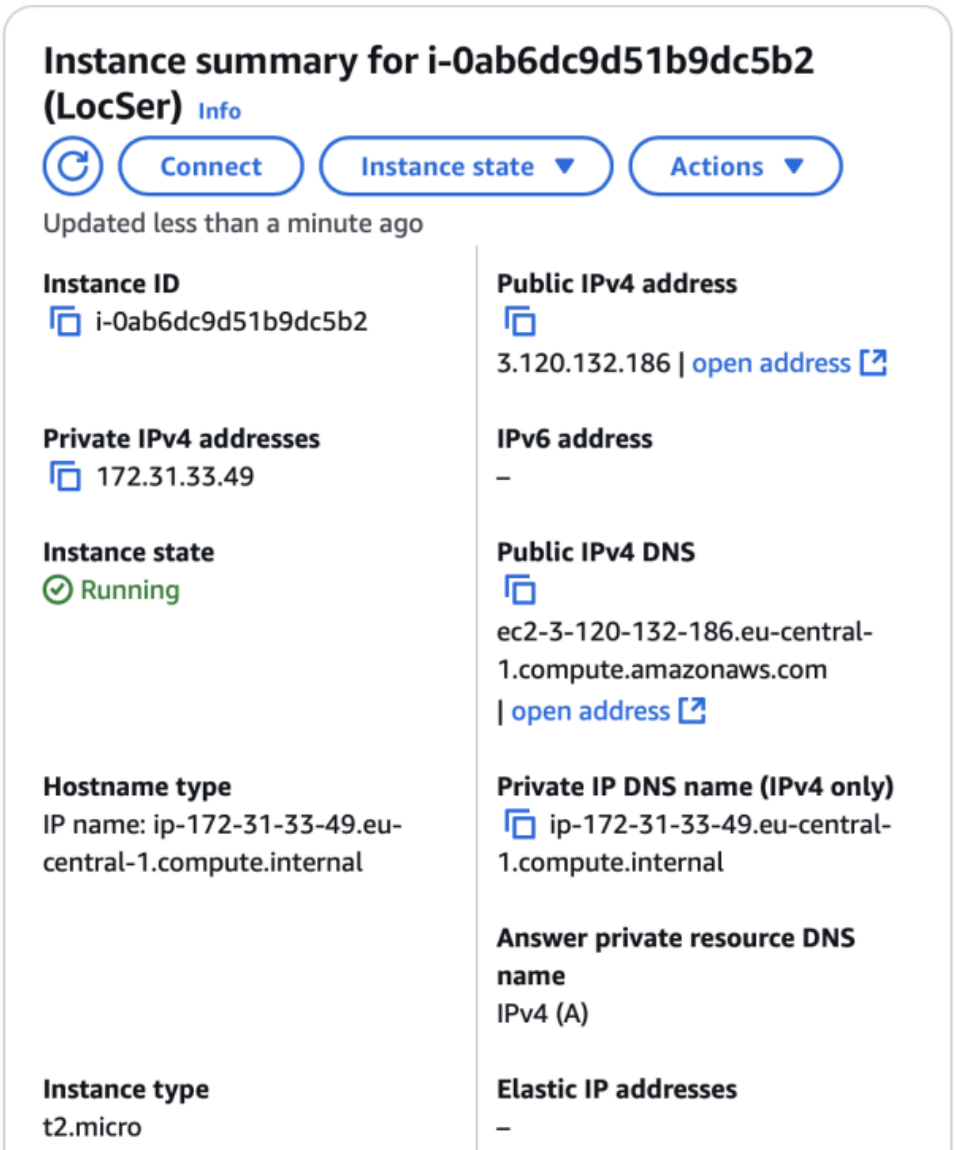

Рисунок 1 - Сервер, выделенный под проект

2.6 Клиентская часть

В целях демонстрации работы продукта было создано мобильное приложение для iOS. При его разработке использовались Xcode, Swift, SwiftUI, SwiftPM и MVVM+Combine. Так, была сделана MVP версия сервиса для презентации в виде мобильного приложения, которое выводит подобно маркетплейсу изображения книг, их название и цену, как показано на Рисунке 2.

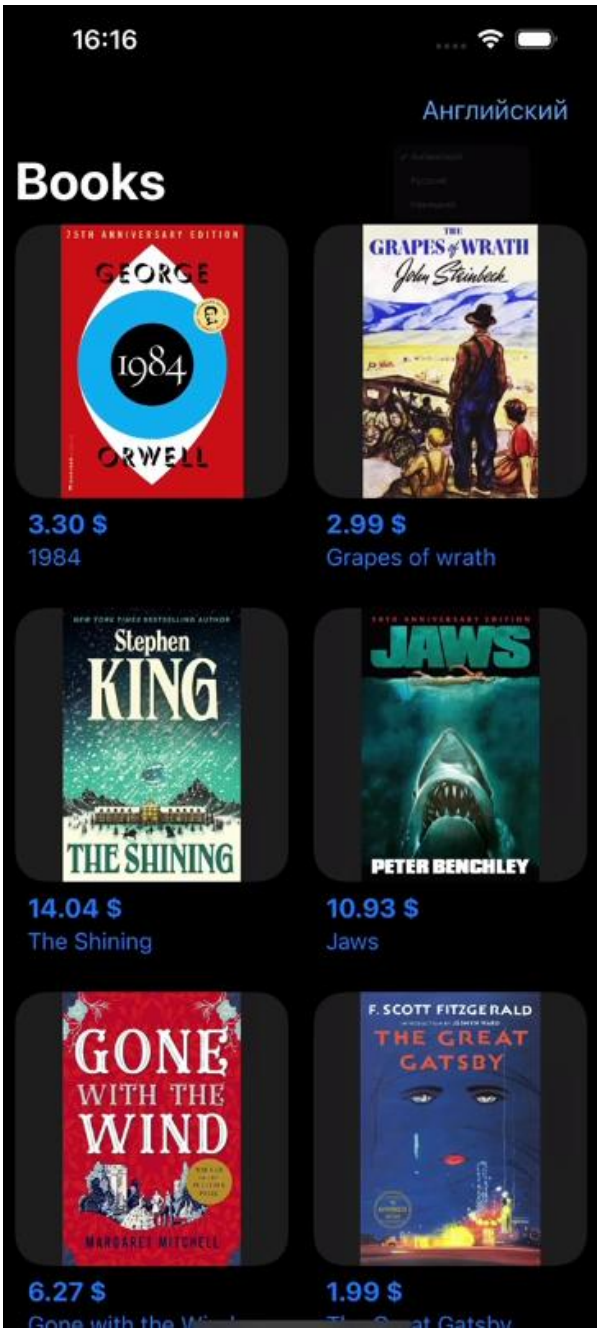

Рисунок 2 - Основной экран приложения

3 Мои задачи и их решение

Я был взят в команду для разработки ML составляющей проекта, поэтому мои основные задачи лежали в этой тематике. Вместе со мной также работал Марк Калинин, и все задачи были распределены между нами двумя.

Изначально в техническом задании были выставлены следующие этапы: сбор и обработка данных, знакомство с HuggingFace, обучение и оптимизация модели, создание интерфейса для интеграции с Backend частью. Впоследствии оказалось, что основная работа заключается только в двух из этих этапов: сборе/обработке данных и оптимизации/подборе модели.

Сначала мы должны были разобраться с предоставленным шаблоном [5]. Из него мы получили первую модель для перевода с английского на французский, которая показала хорошие результаты. Пример её переводов можно увидеть на Рисунке 3.

```
Leguminous plants share resources with nitrogen-fixing bacteria.

Les plantes légumineuses partagent des ressources avec des bactéries fixatrices d'azote.
```

Рисунок 3 - Перевод английского на французский

Далее результаты сильно ухудшились. При переводе английского на русский модель не справлялась с распознаванием слов и только лишь повторяла структуру первоначального сообщения (знаки препинания, примерная длина слов), когда как перевод выдавал несвязные слова. По этим причинам модель от Google была заменена аналогами от Helsinki-NLP, специализированными под конкретные пары языков.

Другой глобальной проблемой было низкое качество датасетов. Большинство из них было получено автоматическим переводом, содержащим большое количество ошибок и мусорных данных, другой большой пласт переводов концентрировался на субтитрах и статьях Википедии, что также снижало качество обучения. Помимо этого, была острая необходимость нормализации данных, так как найденные качественные параллельные корпуса зачастую содержали переводы только на конкретные темы, когда как нам был нужен усреднённый вариант.

В итоге, оба этих метода, смена моделей на более специализированные и очистка данных, помогли улучшить перевод для оставшихся четырёх языков и закрыть большую часть задач. Все метрики сохранялись на сайте Weights & Biases [6] для подробного анализа после тренировки, что позволяло ускорить решение. Как видно из Рисунка 4, для перевода на китайский положительные изменения в метриках наиболее наглядные, так как обучение было запущено на большем количестве эпох, чем для остальных языков.

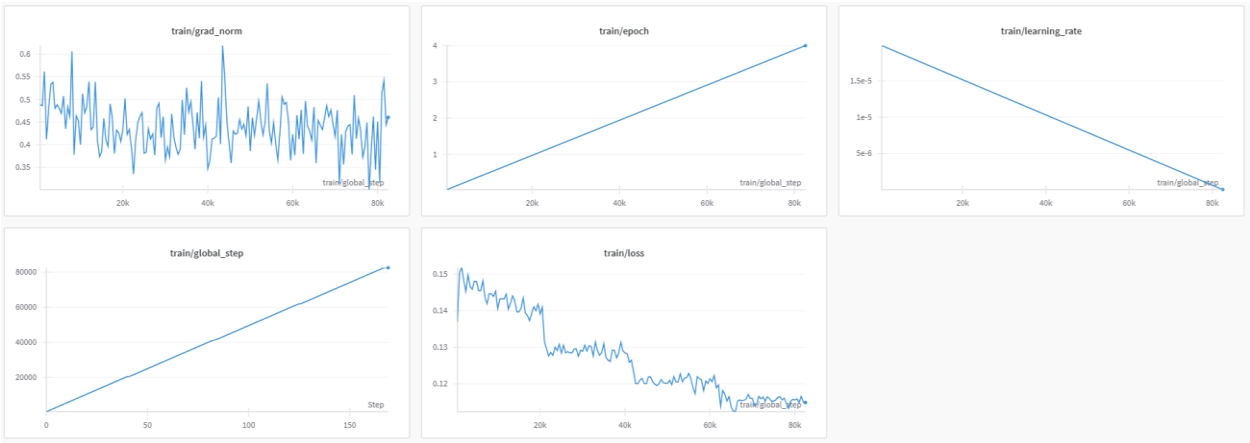

Рисунок 3 - Показатели перевода английского на китайский

Под конец, мы также добавили endpoint для интеграции модели с Backend’ом при помощи API. Все файлы обученных моделей лежат в удалённом репозитории на Hugging Face [7], а сервер, соответственно, обращается к нужному переводчику пары языков, в зависимости от запроса: "Goshective/kazparc_en_ru_marian_1", "Goshective/wlhb_en_zh_marian_1", "Goshective/kazparc_en_tr_marian_1", "Goshective/wmt14-en-de_marian_1", "Goshective/opus_books_model_french".

4 Анализ проделанной работы

Я считаю, что работа, по большей части, была проделана мной успешно. По итогам проекта, весь заявленный функционал был реализован. Так, хоть изначально речь даже шла о том, чтобы вовсе не обучать модель, а пользоваться чужим API, поскольку задача довольно сложна; в итоге было сделано даже чуть больше, чем планировалось изначально, например, перевод цен и описаний книг. Трудности и проблемы, возникшие по ходу разработки были рано или поздно решены либо самостоятельно, либо с помощью команды и куратора, когда собственных сил не хватало. Работа в большинстве моментов была довольно увлекательной и заставляла думать нестандартно и пытаться искать новые решения. Например решение использовать другие модели пришло далеко не сразу, и многие часы были потрачены на доработку первоначального переводчика из руководства.

Тем не менее, я понимаю, что в некоторые моменты брал на себя слишком много ответственности и из-за собственной заинтересованности выполнял задачи за остальных, в частности, за Марка. Так, большая часть всей тренировки моделей лежала на мне, поскольку стационарный компьютер позволял получать результаты быстрее, чем ноутбук Марка. В итоге, мы решили разделиться внутри нашей работы, чтобы я занимался только тренировкой, сохранением и отладкой моделей, а Марк - альтернативным улучшением перевода через DeepL. С одной стороны, это разделение помогло нам быстрее выполнить задание, но с другой, все задачи, связанные с ML ушли ко мне, хотя изначально должны были делиться поровну. Но сейчас я считаю, что, в целом, этот опыт всё равно поможет мне в будущем совершать меньше похожих ошибок и более грамотно распределять обязанности, потому что я буду, как минимум, знать, что такая ситуация может произойти во время работы и, главное, как её можно разрешить.

5 Взаимодействие с командой и руководителем

Проведя анализ нашей работы, я всё равно прихожу к выводу, что, в целом, наш опыт работы над этим проектом в команде был довольно успешным.

Помимо работы в мини-группах, разделённых по конкретным областям нам часто приходилось кооперироваться с остальными участниками проекта, например, чтобы провести грамотную интеграцию или обсудить требуемый функционал. Так, некоторые из совместных звонков могли проходить без участия куратора, что явно положительно сказывалось на нашей командной работе. Большую часть времени я общался с Марком, поскольку наша работа над ML предполагала самостоятельное распределение обязанностей. Как я упоминал ранее, на мне оставалась вычислительная часть и сохранение промежуточных результатов на репозитории в GitHub и Hugging Face, а Марк, помимо помощи с поиском датасетов, решал и обсуждал вместе со мной возникающие проблемы, отделившись под конец для работы с DeepL и интеграции с Backend’ом.

Наш руководитель, Игорь Манаков, грамотно распределил нас по областям и объединил в небольшие группы, чтобы мы сами выбирали достаточную нагрузку и совместно решали возникающие проблемы. Так, по собственному опыту, часто у нас практически не возникало необходимости обращаться к куратору, поскольку кто-то из команды уже мог помочь с появившейся проблемой. Но при всём этом, вклад Игоря не может остаться незамеченным. Он всегда был на связи, поддерживал нас и старался чему-то научить своими заданиями. Из недостатков можно лишь выделить то, что некоторые указания были понятны не сразу или были расплывчатыми, но эти проблемы разрешались на повторных собраниях. Поэтому я считаю, что Игорь Манаков достойно проявил себя в роли руководителя проекта и куратора, дающего возможность для нашего развития.

ЗАКЛЮЧЕНИЕ

Можно с уверенностью сказать, что цель проекта была достигнута, поскольку мы не только сделали MVP, заявленный в качестве минимального результата, но и добавили в него некоторые опциональные функции вроде перевода валют и описания книг. Каждая поставленная задача была выполнена в срок. В результате мы получили обученную модель-переводчик и мобильное приложение, работающие в связке с удалённым сервером.

Сервис можно развивать в различные стороны. Сейчас он находится лишь в презентационном виде, но так как задумка предполагала сервис для разработчиков мобильных приложений, его можно преобразовать в систему, которую легко интегрировать в проект для дальнейшей локализации. Основными недостатками проекта выступают модели, обученные на небольших мощностях (в силу нехватки ресурсов и времени). Тогда, путями для улучшения сервиса могут служить использование более продвинутого оборудования и средств оптимизации для ускорения обучения; более качественных данных для тренировки, возможно, выбора данных для каждой необходимой тематики по отдельности. Так, проект сможет выйти на коммерческий уровень и стать востребованным среди разработчиков.

В целом, я считаю, что мой вклад в проект был приемлемым и достаточным для того, чтобы успешно презентовать финальный вид MVP. Мною были получены знания в незнакомой до этого области NLP, а также получилось ознакомиться с популярными средствами, используемыми в ней, например, сервисами от Hugging Face и Weights & Biases. За время работы я получил полезный опыт отчётности о сделанных задачах, постоянных встреч с командой и, в целом, приблизился к опыту реальной разработки с чётким планом, сроками и обязанностями.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Hugging Face. issai/kazparc · Datasets at Hugging Face. URL: https://huggingface.co/datasets/issai/kazparc. Дата обращения: 05.01.2025
2. Hugging Face. Helsinki-NLP/opus_books · Datasets at Hugging Face. URL: https://huggingface.co/datasets/Helsinki-NLP/opus_books. Дата обращения: 05.01.2025
3. Hugging Face. wmt/wmt14 · Datasets at Hugging Face. URL: https://huggingface.co/datasets/wmt/wmt14/tree/main. Дата обращения: 05.01.2025
4. Hugging Face. wlhb/Translation-Chinese-2-English. URL: https://huggingface.co/datasets/wlhb/Transaltion-Chinese-2-English. Дата обращения: 05.01.2025
5. Hugging Face. Translation. URL: https://huggingface.co/docs/transformers/en/tasks/translation. Дата обращения: 05.01.2025.
6. Weights & Biases. The AI Developer Platform. URL: https://wandb.ai/site Дата обращения: 05.01.2025.
7. Hugging Face. Goshective · Models at Hugging Face. URL: https://huggingface.co/Goshective. Дата обращения: 05.01.2025

ПРИЛОЖЕНИЕ. ТЕХНИЧЕСКОЕ ЗАДАНИЕ

1. **Название:** Разработка сервиса локализации мобильного приложения
2. **Цель (назначение):** Разработать MVP сервиса для локализации, который может принимать нужные строки приложения, а затем выводить их переведёнными на нужные языки.
3. **Сроки выполнения:** Начало – 31.10.2024 Окончание – 18.12.2024
4. **Исполнитель проекта (руководитель проекта):** Игорь Манаков
5. **Термины и сокращения:**

- **MVP (Minimal Viable Product)** – продукт, обладающий минимальными, но достаточными для удовлетворения первых потребителей функциями
- **Параллельный корпус** – большие собрания текстов с выравниванием по предложениям с сопоставлением одного языка с другим.
- **Датасет** – коллекция данных в табличном виде
- **Endpoint** – это конечная точка веб-сервиса, к которой клиентское приложение обращается для выполнения определённых операций или получения данных.
- **ML (Machine Learning)** – класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение за счёт применения решений множества сходных задач.
- **API (Application Programming Interface)** – программный интерфейс, то есть описание способов взаимодействия одной компьютерной программы с другими.

6. **Технические требования:**

- Сервис должен уметь переводить тексты асинхронно на выбранных языках;
- Сервис должен базироваться на удалённом сервере;
- Сервис должен поддерживать хостинг картинок;
- Сервис должен иметь API для доступа к модели

7. Содержание работы.

Таблица 1 - Содержание работы

Название задачи	Этап	Ответственный
Спроектировать архитектуру сервиса	Документирование	Данилевский Алексей Александрович
Разработать контракт API	Работа с Backend'ом	Ахмедов Бахадыр Бахтиерович
Разработать Endpoint'ы с конфигурацией МП	Работа с Backend'ом	Ахмедов Бахадыр Бахтиерович
Развернуть сервер	Работа с Backend'ом	Толкачев Тимур Сергеевич
Изучить FastAPI	Работа с Backend'ом	Ахмедов Бахадыр Бахтиерович
Изучить Docker	Работа с Backend'ом	Толкачев Тимур Сергеевич
Сверстать UI МП	Работа с Frontend'ом	Данилевский Алексей Александрович
Создать сервисный слой МП	Работа с Frontend'ом	Данилевский Алексей Александрович
Собрать данные	Работа с моделью	Калинин Марк Алексеевич
Сгенерировать документацию API	Документирование	Ахмедов Бахадыр Бахтиерович
Познакомиться с Hugging Face	Работа с моделью	Скворцов Денис Александрович
Создать интерфейс для интеграции с Backend'ом	Работа с моделью	Калинин Марк Алексеевич
Обработать данные	Работа с моделью	Скворцов Денис Александрович
Обучить модель	Работа с моделью	Калинин Марк

		Алексеевич
Оптимизировать модель	Работа с моделью	Скворцов Денис Александрович
Добавить кэширование данных	Работа с Backend'ом	Толкачев Тимур Сергеевич
Интегрироваться в сервис	Работа с Backend'ом	Толкачев Тимур Сергеевич
Пройти курс по разработке МП	Работа с Frontend'ом	Данилевский Алексей Александрович
Доработка модели	Работа с моделью	Скворцов Денис Александрович
Добавить перевод навигационных компонентов	Работа с моделью	Калинин Марк Алексеевич
Интегрировать S3	Работа с Backend'ом	Ахмедов Бахадыр Бахтиерович

Примечание: этапы работы были скопированы из приложения Odoo и не полностью совпадают с реальностью, иногда не совпадают даже ответственные за этап.