

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

Направление подготовки: 09.03.01 «Информатика и вычислительная техника»
Профиль подготовки: «Программное обеспечение средств вычислительной техники и автоматизированных систем»

БАКАЛАВРСКАЯ РАБОТА

на тему

Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени. Фронтенд-разработка»

Студент


(подпись, дата)

Коробкин Виктор Юрьевич
(ФИО полностью)

Руководитель


(подпись, дата)

С.А. Зинкин
(фамилия, инициалы)

Нормоконтролёр


(подпись, дата)

А.С. Бычков
(фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 22.05.2026 г. № 14)

Заведующий кафедрой


(подпись)

М.А. Митрохин
(фамилия, инициалы)

Работа защищена с отметкой хорошо (протокол заседания ГЭК от

08 июня 2026 г. № 17)

Секретарь ГЭК


(подпись)

Е.И. Гурин
(фамилия, инициалы)

Пенза 2026

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
"07" 04 2026 г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Коробкин Виктор Юрьевич гр. 22ВВП1 факультета вычислительной техники
2. Тема работы: Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени. Фронтенд-разработка»

Тема утверждена приказом ПГУ № 347/о от "07" 04 2026 г.

3. Руководитель бакалаврской работы д.т.н., профессор, каф. ВТ ПГУ Зинкин С. А.

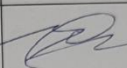
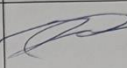
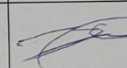
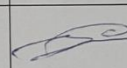
4. Задание на работу (назначение разработки, исходные данные и т.п.)

Разработать Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени. Фронтенд-разработка», позволяющее выбрать область экрана, получить изображение этой области, распознать текст и вывести перевод в отдельном окне или поверх исходного текста. В приложении должна быть интегрирована многоязычная системы перевода, обеспечивающая высокую точность и минимальную задержку обработки. Приложение должно обладать удобным клиентским интерфейсом, позволяющим пользователю управлять процессом перевода, настраивать языки и работать с выделенными элементами контента.

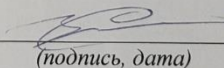
5. Перечень подлежащих разработке вопросов

Требуется произвести анализ предметных областей: изучить существующие аналоги, выделить их достоинства и недостатки. Выбрать язык программирования, который подойдет для создания графического интерфейса. Определить архитектуру приложения и вспомогательные средства разработки. Изучить функционал библиотек для взаимодействия с изображением на экране и создания оконного приложения.

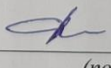
6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Изучение предметной области	32	07.04.2026 – 13.04.2026	
2	Анализ требований	32	13.04.2026 – 20.04.2026	
3	Создание макета графического интерфейса	30	20.04.2026 – 26.04.2026	
4	Выбор технологий, применяемых при проектировании и разработке	40	26.04.2026 – 02.05.2026	
5	Реализация графического интерфейса	66	02.05.2026 – 08.05.2026	
6	Разработка основной части программы	70	08.05.2026 – 15.05.2026	
7	Оформление пояснительной записки	54	15.05.2026 – 21.05.2026	

Дата выдачи задания "07" 04 2026 г.

Руководитель бакалаврской работы  Зинкин С. А.
(подпись, дата)

Задание к исполнению принял студент  Коробкин В. Ю.
(подпись, дата)

Работу к защите допустить
Декан факультета  07.06.2026 Фионова Л.Р.
(подпись, дата)

Реферат

Бакалаврская работа содержит 110 листов, 21 рисунок, 9 таблиц, 8 источников и 3 приложения.

ФРОНТЕНД-РАЗРАБОТКА, ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС, ЗАХВАТ ЭКРАНА, OCR, ПЕРЕВОД ТЕКСТА, OVERLAY, TKINTER, PYTHON, ГОРЯЧИЕ КЛАВИШИ

Цель работы - разработка фронтенд-части программного обеспечения для захвата, распознавания и перевода текстового контента в реальном времени.

В конечном итоге было разработано ПО, которое позволяет выбрать область экрана, получить изображение этой области, распознать текст и вывести перевод.

В качестве языка программирования был выбран Python, с использованием библиотеки Tkinter. Программное обеспечение удовлетворяет поставленной задаче, имеет удобный пользовательский интерфейс, Реализованы главное окно, выбор области экрана, рамка области, overlay-режим, отдельное окно перевода, настройки темы и горячие клавиши. Приложение использует OCR и перевод как подключаемые модули обработки, а разработанная часть обеспечивает взаимодействие пользователя с этими функциями.

Разработанное программное обеспечение может применяться при чтении иностранных интерфейсов, инструкций, субтитров, изображений с текстом и учебных материалов.

ПГУ 1.090301.17.001 ПЗ				
Изм	Лист	№ докум.	Подпись	Дата
Разраб.		Коробкин В.Ю.		28.05.24
Пров.		Зинкин С. А.		28.05.24
Н. Контр.		Бычков А.С.		28.05.24
Утв.		Митрохин М.А.		28.05.24

Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени. Фронтенд-разработка»

Пояснительная записка

Литера	Лист	Листов
	2	82

ФВТ, каф. ВТ
гр. 22ВВП1

Содержание

Перечень сокращений и обозначений.....	6
Введение.....	7
1 Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени. Фронтенд-разработка»....	9
1.1 Анализ предметной области.....	9
1.1.1 Актуальность разработки.....	10
1.1.2 Анализ существующих аналогов.....	11
1.1.3 Обоснование актуальности разработки программы.....	21
1.1.4 Постановка задачи.....	22
1.2 Выбор технологий и средств разработки приложения.....	27
1.2.1 Выбор языка программирования.....	27
1.2.2 Выбор библиотеки для графического интерфейса.....	28
1.2.3 Выбор средств захвата изображения.....	29
1.2.4 Выбор средств распознавания и перевода.....	29
1.2.5 Выбор среды разработки.....	30
1.2.6 Обмен данными между компонентами.....	31
1.3 Проектирование архитектуры и пользовательского приложения.....	32
1.3.1 Общая архитектура программы и внутренний backend.....	33
1.3.2 Граница ответственности фронтенд-разработки.....	34
1.3.3 Проектирование пользовательского приложения.....	35
1.3.4 Логические слои интерфейса.....	35
1.4 Разработка приложения.....	36
1.4.1 Реализация главного окна и координатора состояния.....	37
1.4.2 Описание основных классов с примерами реализации.....	39
1.4.3 Реализация выбора области экрана.....	41
1.4.4 Реализация внутреннего сценария обработки.....	42
1.4.5 Реализация режимов вывода результата.....	43

1.4.6	Реализация настроек и горячих клавиш.....	43
1.4.7	Руководство пользователя.....	44
1.4.8	Тестирование и отладка.....	45
1.4.9	Последовательность взаимодействия компонентов.....	46
1.4.10	Анализ полученных результатов и развитие.....	48
2	Стартап-проект Screen Translate.....	49
2.1	Обоснование актуальности.....	49
2.2	Описание и обоснование выбора ВКР в формате ВКРС.....	50
2.3	Описание существующей проблемы на рынке.....	52
2.4	Описание решения проблемы и продуктового решения.....	53
2.4.1	Наименование продукта.....	53
2.4.2	Назначение и сфера применения.....	54
2.4.3	Пользовательский сценарий.....	54
2.4.4	Фронтенд-зона ответственности.....	54
2.4.5	Функциональные возможности реализованной программы.....	55
2.4.6	Конкурентоспособность и инновационность.....	56
2.4.7	Степень готовности продукта, поставка и обслуживание.....	57
2.5	Общая характеристика стартап-проекта и сферы деятельности.....	57
2.5.1	Основная бизнес-идея.....	57
2.5.2	Сфера деятельности.....	58
2.5.3	Целевая аудитория.....	58
2.5.4	Текущее состояние разработки.....	59
2.6	Объем, структура и развитие рынка.....	59
2.7	Анализ конкурентов и аналогов.....	61
2.8	Бизнес-модель и команда стартап-проекта.....	62
2.8.1	Бизнес-модель.....	62
2.8.2	Каналы продаж и продвижения.....	63
2.8.3	Команда стартап-проекта.....	64
2.9	Финансовая модель и необходимость привлечения инвестиций.....	65
2.9.1	Предпосылки финансовой модели.....	65

2.9.2 Расходы проекта.....	65
2.9.3 Потребность в финансировании.....	66
2.9.4 Точка безубыточности и инвестиционная привлекательность.....	66
2.10 Производственный и организационный план.....	67
2.10.1 Общие сведения о производстве цифрового продукта.....	67
2.10.2 Технологическая цепочка разработки.....	67
2.10.3 Кадровое обеспечение и организация работ.....	68
2.10.4 Материалы, оборудование, безопасность и экологичность.....	69
2.11 Описание развития бизнес-проекта.....	69
2.12 Описание рисков.....	70
2.13 Заключение по стартап-проекту.....	72
3 Охрана труда.....	75
3.1 Анализ опасных и вредных факторов при работе с ПЭВМ.....	76
3.2 Требования к рабочему месту.....	77
3.3 Освещение, микроклимат и режим труда.....	78
3.4 Электробезопасность и пожарная безопасность.....	79
Заключение.....	80
Список использованных источников.....	82
Приложение А – UML-диаграммы.....	83
Приложение Б – Сокращенный листинг программы.....	89
Приложение В – Презентация.....	97

Перечень сокращений и обозначений

В настоящей работе применяют следующие обозначения и сокращения

ПО — программное обеспечение

ПЭВМ — персональная электронно-вычислительная машина

API — программный интерфейс приложения

Backend — внутренняя логика обработки данных

GUI — графический пользовательский интерфейс

Frontend — пользовательская часть программного продукта

MSS — библиотека для захвата изображения с экрана

OCR — оптическое распознавание символов

Overlay — интерфейсный слой, отображаемый поверх других окон

Tkinter — стандартная библиотека Python для создания графического интерфейса

UI — пользовательский интерфейс

UX — пользовательский опыт

					ПГУ 1.090301.17.001 ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дат		

Введение

В современном информационном пространстве пользователя окружает значительная часть информации, представленная не в виде редактируемого текста, а как изображение на экране. Надпись может находиться в интерфейсе программы, в игре, в видеоролике, на скриншоте, в окне трансляции или в документе, где копирование текста ограничено. В таких случаях пользователь видит текст, но не может быстро передать его в переводчик стандартными средствами операционной системы.

Проблема особенно заметна при работе с иностранными программами и обучающими материалами. Пользователь вынужден делать снимок экрана, открывать сторонний сервис распознавания, вручную выделять фрагмент, копировать результат и только после этого переводить его. Такой порядок действий занимает время и не подходит для случаев, когда текст на экране меняется часто.

Разрабатываемое ПО предназначено для сокращения указанной цепочки действий. Программа позволяет выбрать область экрана, получить изображение этой области, распознать текст и вывести перевод. При этом в рамках настоящей выпускной квалификационной работы основное внимание уделяется не внутренним алгоритмам OCR и машинного перевода, а фронтенд-разработке приложения.

Под фронтенд-разработкой в данном случае понимается разработка пользовательской части настольного приложения: главного окна, элементов управления, выбора области экрана, overlay-вывода поверх других окон, отдельного окна перевода, настроек внешнего вида, горячих клавиш, статусов и пользовательских сценариев. Именно эти элементы определяют, насколько удобно применять технические возможности распознавания и перевода в реальной работе.

Актуальность темы связана с необходимостью создания удобного пользовательского интерфейса для программ, работающих с экраным текстом в реальном времени. Даже точный модуль распознавания не решает

пользовательскую задачу, если приложение сложно запустить, неудобно настроить или оно не сообщает о своем текущем состоянии.

Цель выпускной квалификационной работы - разработать фронтенд-часть ПО для захвата, распознавания и перевода текстового контента в реальном времени.

Для достижения цели необходимо решить следующие задачи:

- изучить предметную область экранного перевода;
- выполнить анализ существующих аналогов;
- сформулировать требования к фронтенд-части;
- определить границы ответственности разрабатываемого интерфейса;
- выбрать технологии и средства разработки;
- спроектировать структуру окон и пользовательских сценариев;
- реализовать основные элементы интерфейса;
- провести проверку работы интерфейса и пользовательских режимов.

Объектом разработки является ПО для захвата, распознавания и перевода текстового контента. Предметом разработки является фронтенд-часть данного ПО, обеспечивающая взаимодействие пользователя с функциями обработки.

Практическая значимость работы заключается в создании интерфейсной части приложения, которая позволяет использовать распознавание и перевод экранного текста без сложных промежуточных действий. Разработанное ПО может применяться при чтении иностранных интерфейсов, инструкций, субтитров, изображений с текстом и учебных материалов.

Структура выпускной квалификационной работы включает введение, основной раздел с анализом предметной области, выбором технологий, проектированием и разработкой интерфейсной части, раздел по охране труда, заключение, список использованных источников и приложения. [1], [2]

1. Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени. Фронтенд-разработка».

Основная часть работы посвящена разработке фронтенд-слоя приложения Screen Translator. Данный слой отвечает за взаимодействие пользователя с программой и связывает действия пользователя с внутренними функциями захвата изображения, распознавания текста и перевода.

В отличие от серверных или вычислительных модулей, фронтенд-часть должна учитывать последовательность пользовательских действий, возможные ошибки, особенности отображения окон поверх других приложений и необходимость работы в реальном времени. Поэтому проектирование начинается с анализа пользовательских сценариев и ограничений предметной области.

Структура первого раздела выстроена последовательно: сначала рассматривается предметная область и существующие решения, затем формулируется постановка задачи, обосновывается выбор средств разработки, описывается архитектура приложения и приводится реализация интерфейсной части.

1.1 Анализ предметной области

Для того, чтобы разработать удобное для пользователя ПО нужно проанализировать все аспекты работы ПО, учесть все пожелания пользователя.

Современный пользователь ожидает, что приложение будет работать поверх обычных окон операционной системы. Оно не должно требовать переноса текста в отдельный документ, ручного обрезания скриншота или длительной настройки перед каждым переводом. Поэтому важным требованием становится минимизация количества действий.

Предметная область экранного перевода находится на пересечении пользовательских интерфейсов, компьютерного зрения и машинного перевода.

Однако для фронтенд-разработчика ключевой задачей является не создание математической модели OCR, а организация удобного доступа к этой обработке.

На основе анализа было решено разработать ПО, в котором используется сценарий, когда пользователь задает область экрана, выбирает исходный и целевой языки, запускает обработку и получает результат либо в отдельном окне, либо прямо поверх исходного текста. Такой сценарий требует согласованной работы нескольких визуальных компонентов.

Дополнительная особенность предметной области состоит в том, что пользователь может работать с динамическими окнами: видео, играми, прокручиваемыми страницами, меняющимися сообщениями. В этом случае интерфейс должен корректно очищать старый результат, не оставлять лишние надписи на экране и реагировать на повторный запуск.

Важное значение имеет визуальная незаметность приложения. Когда перевод выводится поверх экрана, пользователь не должен ощущать, что программа закрывает основной материал. Поэтому в интерфейс включаются настройки цвета, шрифта, прозрачности и режима отображения.

1.1.1 Актуальность разработки

Актуальность разработки обусловлена тем, что пользователю часто требуется быстро понять текст, который технически недоступен для копирования. Это может быть надпись в изображении, элемент интерфейса, субтитр, фрагмент окна игры или сообщение в окне программы.

Большинство стандартных переводчиков хорошо работают с обычным текстом, но хуже приспособлены к ситуации, когда текст находится только на экране. Пользователь вынужден использовать несколько отдельных инструментов: скриншот, OCR-сервис и переводчик. Такой процесс разрывает контекст и снижает скорость работы.

Экранный перевод в реальном времени требует интерфейса, который не мешает основной работе. Если каждое действие требует перехода в другое окно, пользователь быстро отказывается от такого решения. Следовательно, фронтенд

					ПГУ 1.090301.17.001 ПЗ	Лист
						10
Изм.	Лист	№ докум.	Подпись	Дат		

должен обеспечивать быстрый доступ к функциям через горячие клавиши и понятные режимы.

Особую актуальность имеет overlay-режим. Он позволяет видеть перевод в том же месте, где находился исходный текст, что облегчает понимание интерфейса или видео. Такой режим требует аккуратного проектирования, потому что окно вывода должно быть заметным, но не должно полностью закрывать полезную область.

Разработка фронтенд-части ПО является самостоятельной и значимой задачей. От нее зависит, сможет ли пользователь применять распознавание и перевод в реальной работе, а не только как экспериментальную функцию.

1.1.2 Анализ существующих аналогов

Для определения требований к разрабатываемому интерфейсу были рассмотрены аналоги, которые решают похожую задачу: перевод текста с изображения, распознавание текста на экране или перевод поверх другого окна.

Первый представленный аналог – Translumo.exe. К преимуществам данного приложения относятся: качественный интерфейс, использование горячих клавиш с возможностью переназначения и поддержание разных моделей для перевода, возможность перевода на множество языков.

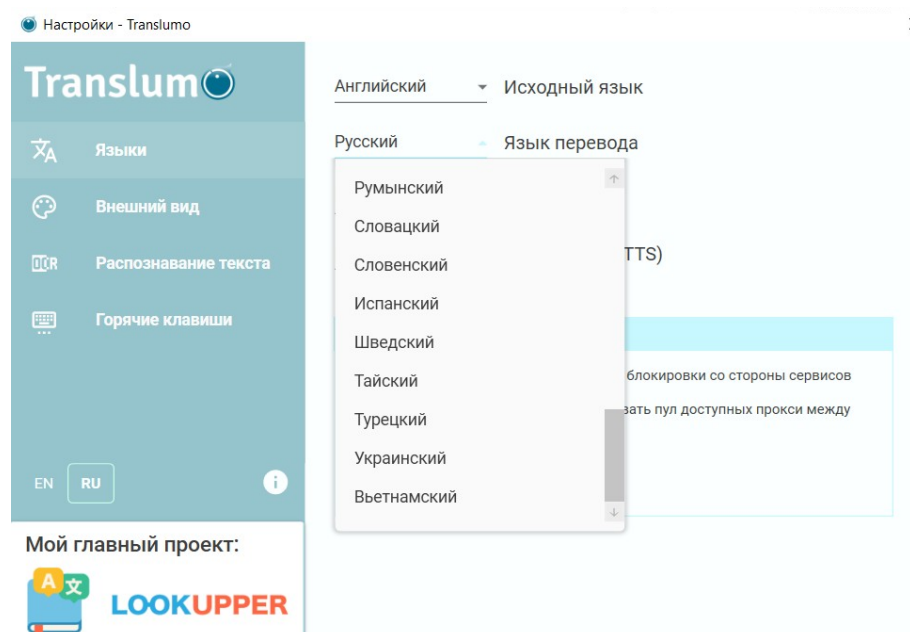


Рисунок 1 – Интерфейс приложения Translumo.exe

Тем не менее у приложения имеются недостатки, связанные как с переводом, так и с технической составляющей. Недостаток первый: у приложения всего один режим, в котором текст переводится подряд, без абзацев, переносов на новую строку, что усложняет восприятие переводимого текста.

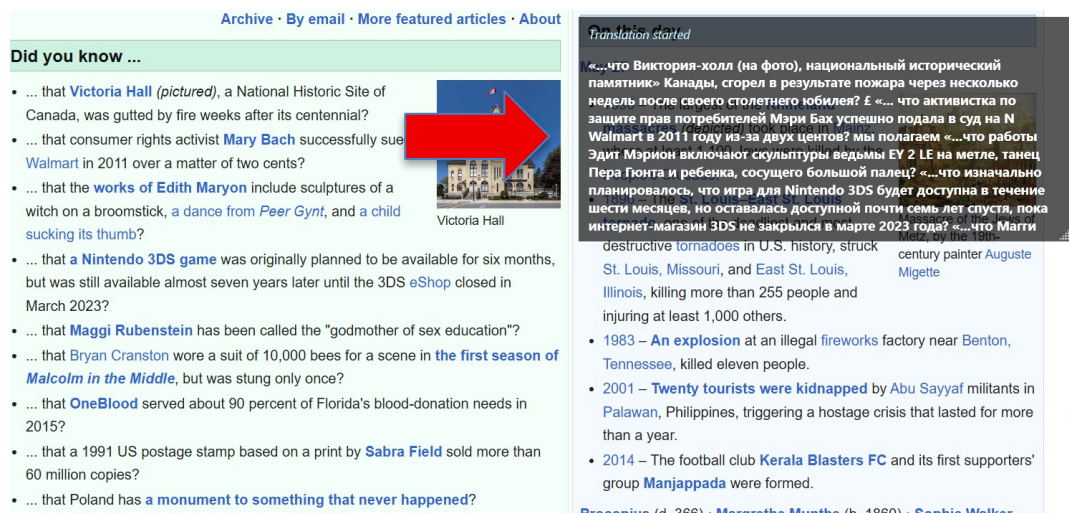


Рисунок 2 – Проблема формата переведенного текста

В нашем приложении есть overlay-режим, в котором переведенный текст не просто сохраняет свой формат, но и накладывается на оригинал, что делает восприятие текста очень простым и понятным. Таким образом, если пользователя не устраивает формат оконного режима, он может воспользоваться overlay-режимом.

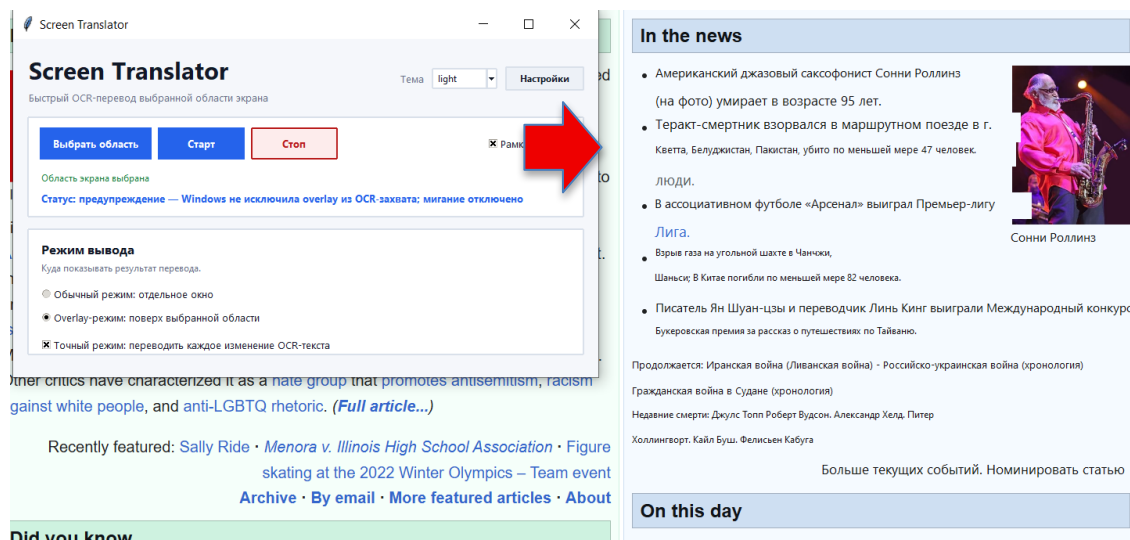


Рисунок 3 – Демонстрация удобства overlay режима

Также у аналога имеются проблемы с некоторыми переводчиками, например переводчик DeepL работает некорректно и часто вызывает ошибки.

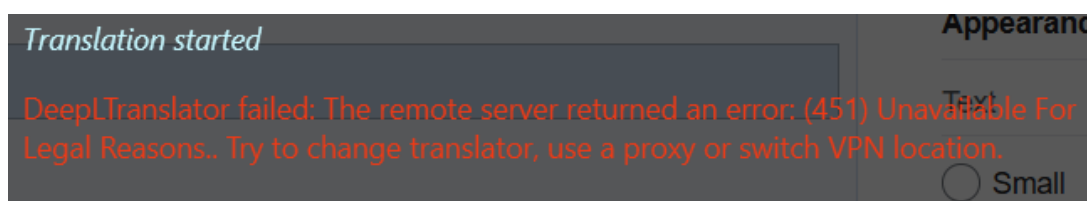


Рисунок 4 – Проблема работы переводчика DeepL

Интерфейс аналога также имеет недостатки: окно перевода открывается при запуске приложения и закрыть его возможно только через диспетчер задач.

Для смены модели перевода, переназначения горячих клавиш, смены языка перевода требуется постоянное открытие и закрытие основного окна, так как перевод не может начаться с открытым окном приложения, что усложняет работу с интерфейсом приложения.

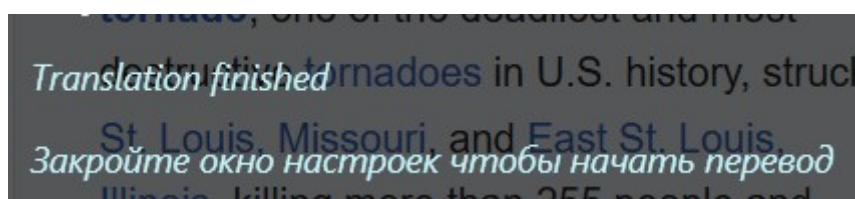


Рисунок 5 – Невозможность перевода при открытом приложении

В нашем приложении такая проблема отсутствует. Переводить текст, менять языки и другие настройки можно без закрытия окна приложения.

Ещё одним преимуществом нашего приложения является возможность делать скриншоты переведенной области, аналог не поддерживает такую возможность. В настройках нашего приложения можно отключить скрывание overlay при создании скриншотов.

Настройки

Применить

Скриншоты и стрим

Управление видимостью overlay в захвате экрана.

☐ Скрывать overlay в скриншотах

Клавиша

print screen

Записать

Рисунок 6 – Возможность управлять видимостью overlay

У приложения аналога есть также и преимущества и главное из них – скорость перевода. Перевод текста в данном приложении занимает от 1 до 2 секунд, в то время как перевод в нашем приложении занимает больше времени – около 5-10 секунд.

Еще одним преимуществом приложения аналога является поддержания интерфейса на двух языках – русский и английский. Это делает интерфейс приложения доступным и понятным не только для русскоязычного сегмента.

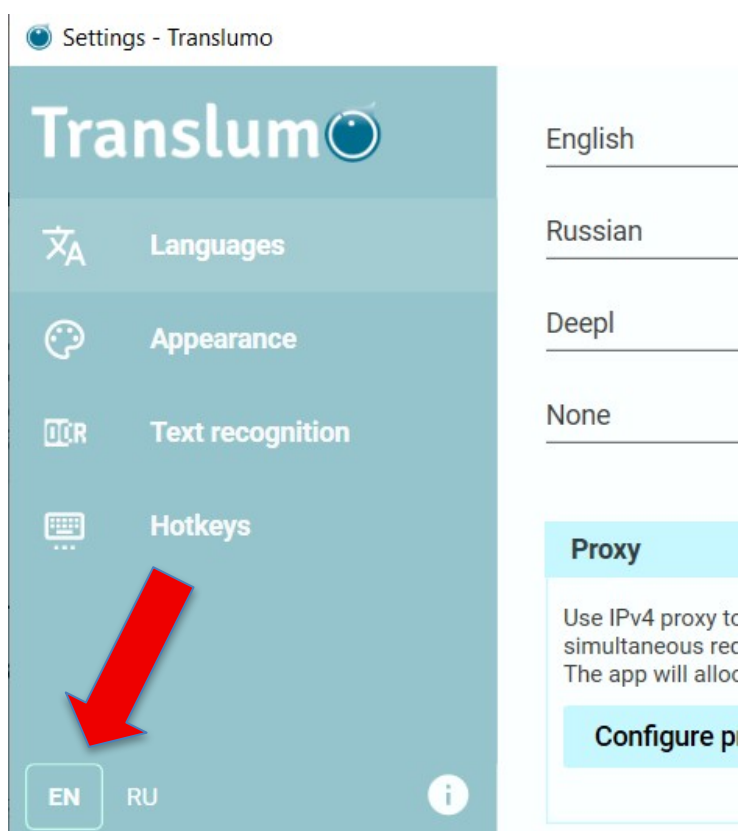


Рисунок 7 – Возможность работы с приложением на двух языках

Также значимым преимуществом приложения аналога является поддержка разных моделей OCR, что даёт пользователю возможность получать более точные переводы и работать со сложными текстами.



Рисунок 8 – Поддержка разных OCR под конкретные задачи

У приложения нет обозначения процесса перевода, поэтому в случае ошибки или бага невозможно понять, процесс перевода запущен или же перевода не произошло вообще. Выявлены программные ошибки, связанные с этим: в процессе игры при нажатии запуска перевода ничего не происходит, хотя область экрана выбрана и горячая клавиша нажата.

Второй представленный аналог – ScreTran.exe.

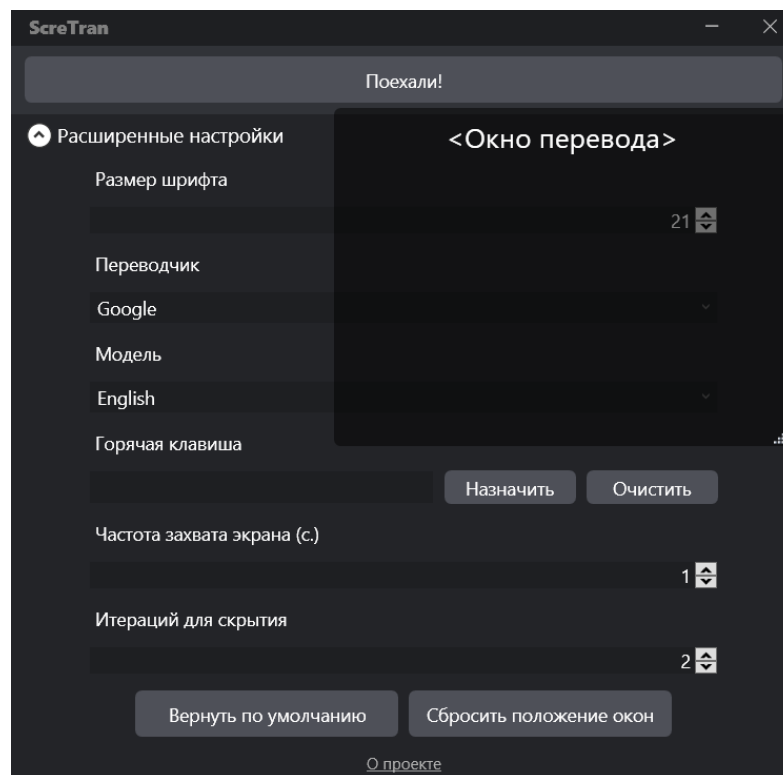


Рисунок 9 – Интерфейс приложения ScreTran

Первый недостаток приложения, который видно сразу – неудобный интерфейс. При открытии приложения появляется основное окно с настройками, окно перевода и рамка, которая является областью захвата экрана. В нашем же приложении захват экрана происходит по нажатию на кнопку и пользователь быстро может захватить необходимую область экрана за 1 клик.

Также у приложения аналога имеется проблема с окном переведенного текста – если область захвата большая и текста много, пользователь не может прокрутить полный переведенный текст колёсиком мыши в маленьком окне. Чтобы увидеть весь текст - нужно обязательно растягивать окно перевода до больших размеров. Таким образом область захвата и окно перевода будут занимать слишком много рабочего пространства на экране, такое отсутствие гибкости делает приложение неудобным для выполнения более узких и конкретных задач.

Следующим недостатком приложения аналога является малое количество настроек: нет возможности менять цвет текста, прозрачность окна и его цвет, что делает приложение не очень удобным для использования и восприятия переведенного текста.

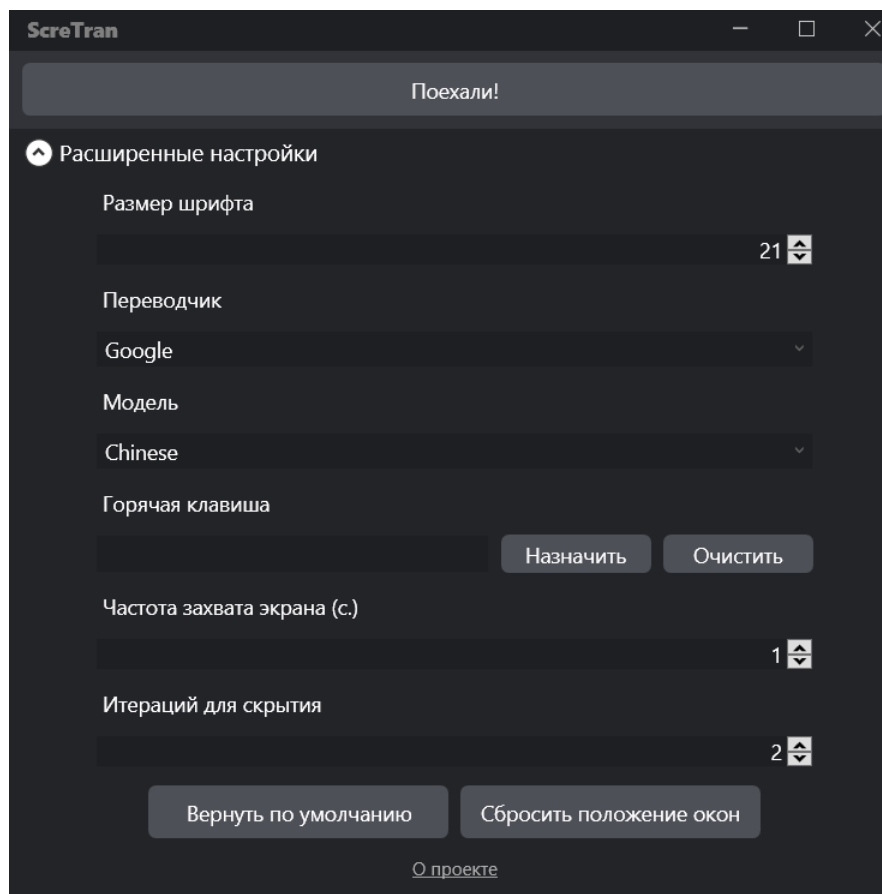


Рисунок 10 – Малое количество настроек для пользователя

Еще одной проблемой приложения аналога является перевод всего на 4 языка, исходный язык – только английский. Это делает приложение очень узконаправленным, возможно переводить только англоязычный контент.

Наше приложение обладает большим арсеналом языков и переводит не только английский, но и другой иностранный контент, перевод с русского на другие языки также поддерживается, что позволяет охватывать гораздо больший контингент пользователей.

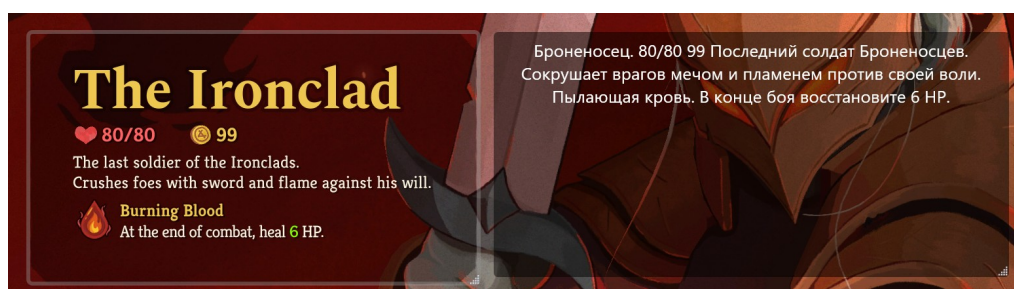


Рисунок 11 – Работа ScreTran в игре

С данным приложением неудобно играть в игры, потому что область захвата текста находится поверх экрана, следовательно данная область недоступна для взаимодействия с игровым контентом.

Преимущество приложения аналога – скорость перевода, около 2-3 секунд. Наше – 5-10 секунд.

Следующий аналог – переводчик «Ёлочка».

Нестандартный формат приложения, полноценного оконного интерфейса не имеет и настраивается из панели «Скрытых значков»

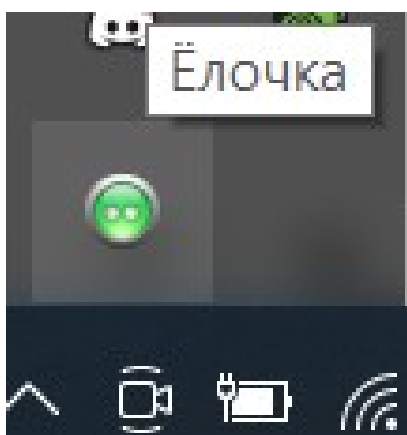


Рисунок 12 – Приложение «Ёлочка»

К преимуществам данного приложения можно отнести максимальную простоту в использовании, нужно всего лишь нажать на букву «Ё» и выделить область экрана.

При своей простоте приложение хорошо сохраняет формат переведенного текста, имеются отступы, перенос на новую строку, текст хорошо воспринимается.

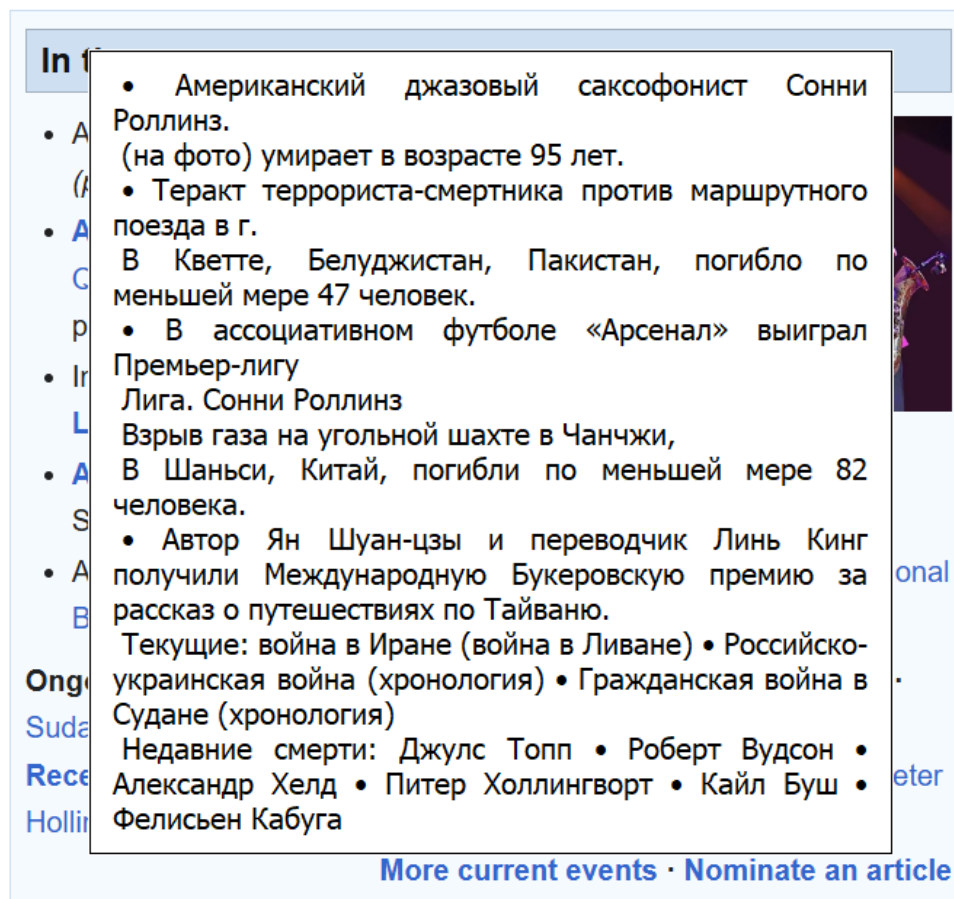


Рисунок 13 – Формат перевода приложения «Ёлочка»

К недостаткам приложение можно отнести малое количество настроек, возможность переводить только с английского на русский.

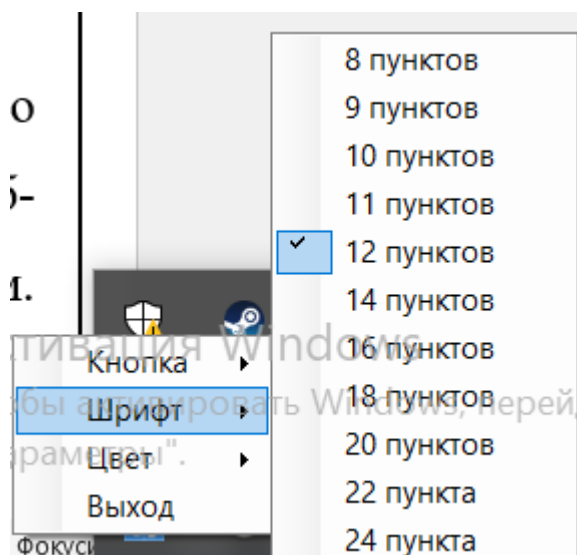


Рисунок 14 – Малое количество настроек приложения аналога

Данное приложение подойдёт только русскоязычному сегменту, который переводит исключительно англоязычный контент.

Преимуществом данного приложения является высокая скорость перевода, почти моментальная – около 0,5 секунд. При этом переведенный текст не сплошной, а с сохранением формата оригинала, что делает приложение максимально простым для восприятия и использования.

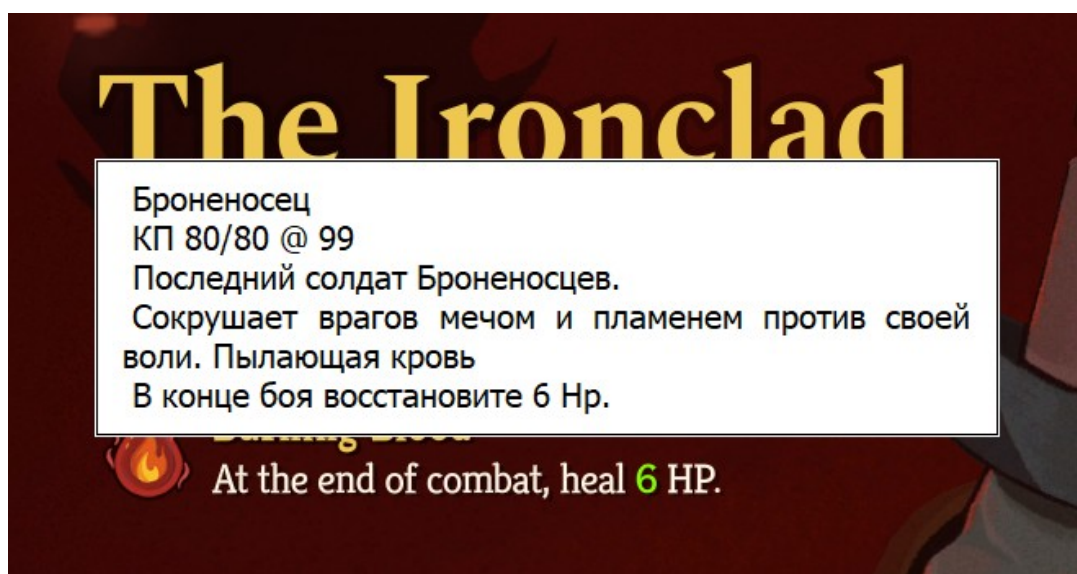


Рисунок 15 – Работа переводчика «Ёлочка» в игре

Также приложение не переводит в реальном времени, а переводит текст на экране одноразово, выделяемая область экрана считывается один раз, а не постоянно в фоновом режиме.

Наше приложение уступает в скорости и простоте использования, но имеет гораздо больший функционал и ориентирован на более широкий сегмент пользователей, что делает наше приложение универсальным по сравнению с данным аналогом.

Анализ существующих аналогов приведен в таблице 1.1

					ПГУ 1.090301.17.001 ПЗ	Лист
						20
Изм.	Лист	№ докум.	Подпись	Дат		

Таблица 1.1 - Анализ существующих аналогов

Аналог	Основные возможности	Преимущества	Недостатки для данной задачи
Translumo	Перевод текста в оконном режиме	Скорость перевода, интерфейс на двух языках, поддержка разных моделей OCR	Программные ошибки в определенных кейсах, потребность в постоянном открытии/закрытии приложения, отсутствие overlay
ScreTran	Перевод текста в оконном режиме	Скорость перевода	Громоздкий интерфейс с тремя окнами, отсутствие overlay, малое количество настроек
«Ёлочка»	Одноразовый перевод текста при захвате экрана	Удобно для сайтов, пошаговых игр, скорость перевода	Малое количество настроек, перевод только с английского на русский, отсутствие overlay
Разрабатываемое приложение	Выбор области, overlay, окно перевода, горячие клавиши	Ориентировано на быстрый пользовательский сценарий	Требуется дальнейшего развития и упаковки

1.1.3 Обоснование актуальности разработки программы

Проведенный анализ показывает, что пользовательская потребность остается актуальной: требуется инструмент, который переводит не абстрактный текстовый документ, а именно тот фрагмент экрана, который видит пользователь в текущий момент.

Существующие решения часто решают только часть задачи. Одни хорошо распознают текст, но требуют загрузки изображения. Другие переводят веб-

страницы, но не работают с графическими интерфейсами сторонних программ. Третьи обладают нужной функциональностью, но перегружены настройками.

Разрабатываемая программа отличается тем, что объединяет несколько пользовательских режимов: обычное окно для управления, выбор области экрана, overlay-вывод результата и отдельное окно перевода. Такая комбинация позволяет применять программу в разных ситуациях.

С точки зрения фронтенд-разработки актуальность состоит в необходимости создать интерфейс, который будет понятен пользователю без дополнительной инструкции. Пользователь должен видеть текущий статус, понимать, какая область выбрана, каким языком управляется перевод и как остановить обработку.

Отдельным фактором является возможность настройки внешнего вида. Для оконного режима один универсальный цвет текста и фона не подходит, так как программа может использоваться поверх разных приложений. Поэтому пользователь должен иметь возможность выбрать тему, цвета, шрифт и прозрачность.

Собственная разработка позволяет сосредоточиться на пользовательском удобстве и границах фронтенд-части, что соответствует теме выпускной квалификационной работы.

1.1.4 Постановка задачи

После анализа предметной области задача разработки была сформулирована не только как написание отдельных функций, а как создание законченного пользовательского сценария экранного перевода. Пользователь должен открыть программу, выбрать область на экране, запустить обработку и получить не только понятный, но и быстрый перевод без ручного копирования текста в сторонние сервисы.

Разрабатываемое фронтенд ПО относится к настольным приложениям. Ее особенность состоит в том, что основное действие происходит поверх других окон операционной системы. Поэтому требования целесообразно разделить на две группы. Первая группа описывает то, что пользователь непосредственно выполняет и видит в интерфейсе. Вторая группа описывает технические свойства,

					ПГУ 1.090301.17.001 ПЗ	Лист
						22
Изм.	Лист	№ докум.	Подпись	Дат		

которые скрыты от пользователя, но определяют устойчивость и качество работы программы.

В работе не разрабатывается собственная модель OCR и не создается отдельный сервис машинного перевода. Эти возможности используются как внутренний backend-слой приложения. Основная задача заключается в проектировании пользовательского приложения и связывании интерфейса с модулями захвата изображения, распознавания и перевода.

Функциональные требования к программе приведены в таблице 1.2

Таблица 1.2 - Функциональные требования к программе

Код	Функциональное требование с позиции пользователя	Критерий выполнения
1	2	3
01	Пользователь выбирает прямоугольную область экрана, в которой находится текст	После выделения область сохраняется в виде координат left, top, width, height, а на экране может отображаться рамка
02	Пользователь выбирает исходный и целевой язык перевода	Выбранные значения преобразуются во внутренние языковые коды и используются OCR-модулем и переводчиком
03	Пользователь запускает перевод выбранной области	После запуска приложение получает изображение области, распознает текст, переводит его и выводит результат
04	Пользователь получает результат в выбранном режиме вывода	В оконном режиме перевод отображается в отдельном окне, в overlay-режиме перевод появляется поверх исходного участка экрана
05	Пользователь может остановить обработку и очистить результат	Активный поток завершается, overlay очищается, статус приложения возвращается в безопасное состояние
06	Пользователь настраивает внешний вид результата	Изменяются тема, цвет фона, цвет текста, шрифт и прозрачность окна перевода

1	2	3
07	Пользователь управляет программой горячими клавишами.	Горячие клавиши запускают выбор области, повторный перевод, остановку, показ рамки и переключение режима.
08	Пользователь может скопировать полученный перевод.	Текст перевода помещается в буфер обмена, если результат уже получен.

Нефункциональные требования сформулированы как проверяемые технические условия. Они не описывают отдельные кнопки интерфейса, а задают свойства работы программы, которые пользователь обычно не видит напрямую.

Нефункциональные требования к программе приведены в таблице 1.3.

Таблица 1.3 - Нefункциональные требования к программе

Код	Нефункциональное требование	Как проверить выполнение
1	2	3
01	Главное окно не блокируется во время OCR и перевода	Обработка выполняется в отдельном потоке, пользователь может нажать команду остановки
02	Выбранная область должна быть не меньше 50 на 30 пикселей	При меньшем размере приложение показывает предупреждение и не запускает обработку
03	Изображение перед OCR ограничивается по размеру	Кадр масштабируется так, чтобы количество пикселей не превышало 2 200 000, а длинная сторона не превышала 1800 пикселей
04	Повторная обработка одинакового изображения должна сокращаться	При малом изменении сигнатуры кадра используется кэш результата, порог повторного использования составляет 0,7 процента

1	2	3
05	Overlay не должен повторно попадать во внутренний снимок экрана	При включенном режиме исключения окно помечается через SetWindowDisplayAffinity и скрывается от внутреннего захвата
06	Случайные повторные нажатия горячих клавиш не должны ломать состояние программы	Используется глобальная задержка 0,5 секунды и задержка 1 секунда для одинакового действия
07	Пользователь должен видеть состояние обработки	Статус меняется при запуске, ожидании OCR, успешном переводе, ошибке и очистке результата
08	Программа не должна сохранять снимки экрана на диск в штатном сценарии	Изображение выбранной области обрабатывается в памяти и передается дальше как объект изображения
09	После остановки должны очищаться временные данные	Очищаются overlay-элементы, кэш текущего результата и служебные флаги активной обработки
10	Внешний вид результата должен быть адаптируемым к темному и светлому фону	Доступны темы и ручные настройки цвета, шрифта и прозрачности

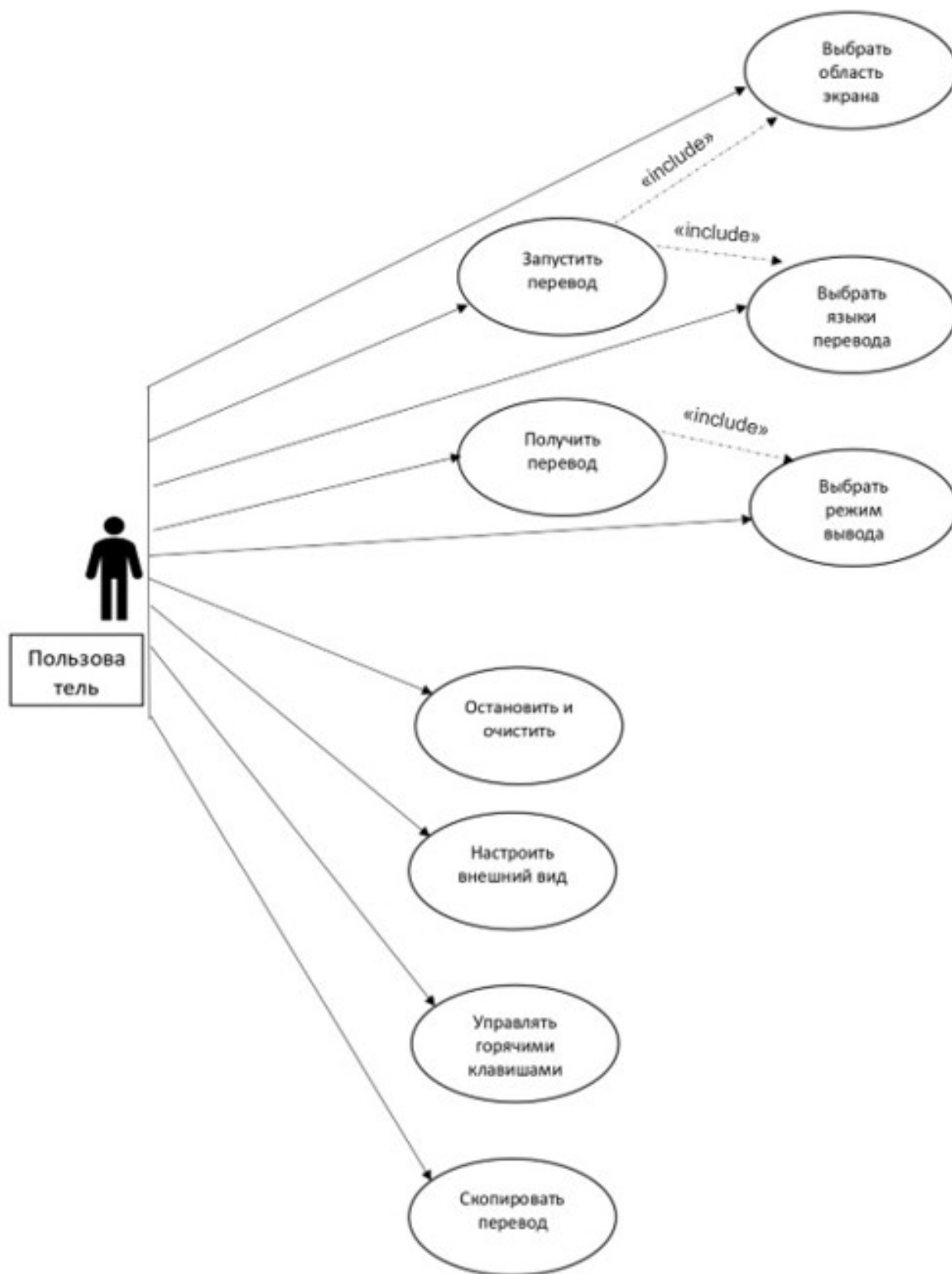


Рисунок 16 - Диаграмма прецедентов

Такая постановка задачи задает последовательную логику изложения: сначала фиксируются действия пользователя и скрытые технические ограничения,

затем выбираются средства разработки, после этого проектируется архитектура и описывается реализация классов.

1.2 Выбор технологий и средств разработки приложения

Выбор технологий выполнялся с учетом условий задачи: приложение должно работать как настольный инструмент, взаимодействовать с окнами операционной системы, выполнять захват изображения с экрана и обеспечивать быстрый вывод результата пользователю.

Python позволяет быстро создавать прикладные программы, использовать готовые библиотеки для захвата экрана, обработки изображений, распознавания текста и перевода. Для дипломного проекта это важно, потому что основное внимание уделяется пользовательскому интерфейсу и сценариям работы, а не низкоуровневой реализации каждого алгоритма.

В качестве графической библиотеки используется Tkinter. Она входит в стандартную поставку Python и позволяет создавать окна, кнопки, поля, списки, canvas-области и дополнительные диалоговые окна без установки отдельного графического фреймворка.

Для обработки изображений используется Pillow, для захвата экрана - MSS, для распознавания текста - EasyOCR, для перевода - deep-translator. Отдельно применяется библиотека keyboard, обеспечивающая работу с глобальными горячими клавишами.

Выбранный стек технологий соответствует настольному характеру приложения и позволяет реализовать все пользовательские функции, необходимые для фронтенд-части.

1.2.1 Выбор языка программирования

Для реализации настольного приложения рассматривались языки Python, C#, Java и C++. Каждый из них может применяться для создания пользовательского интерфейса, однако отличается сложностью разработки и доступностью библиотек. [3]

C# удобен для разработки Windows-приложений и имеет развитые средства построения интерфейсов, однако для задач OCR, предварительной обработки изображений и подключения готовых библиотек машинного распознавания он требует дополнительных оберток и более сложной настройки зависимостей. Python в этом отношении проще: большинство необходимых инструментов доступно как готовые пакеты и может быть подключено напрямую.

Java является кроссплатформенным языком, но создание легкого настольного интерфейса с overlay-логикой, глобальными горячими клавишами и связью с OCR-библиотеками потребовало бы более громоздкой структуры проекта. C++ обеспечивает высокую производительность, однако заметно усложняет разработку интерфейса, работу с памятью и подключение внешних библиотек.

Python выбран как язык, который лучше всего подходит для объединения пользовательского интерфейса, захвата экрана, обработки изображений, OCR и перевода в одном прикладном инструменте. Для фронтенд-части это важно, поскольку разработка сосредоточена на удобстве сценария: выбрать область, запустить обработку, увидеть статус и получить перевод без лишних действий. [3]

Дополнительным преимуществом Python является читаемость исходного кода и высокая скорость внесения изменений. Это позволяет быстрее дорабатывать элементы интерфейса, проверять пользовательские сценарии и изменять поведение окон без значительного усложнения проекта.

1.2.2 Выбор библиотеки для графического интерфейса

Tkinter является стандартным средством Python для разработки графического интерфейса. Его возможности достаточны для реализации главного окна, панели настроек, кнопок, текстового поля, canvas-областей и дополнительных окон. [4]

Главное преимущество Tkinter для данного проекта - простота развертывания. Пользователю не требуется отдельно устанавливать тяжелый графический фреймворк. При наличии Python приложение может использовать штатные средства графического интерфейса.

В проекте Tkinter используется не только для обычного окна, но и для создания специальных окон без рамки: окна выбора области, рамки выбранной

области и overlay-окна. Это показывает, что стандартная библиотека может применяться для достаточно гибкого интерфейсного решения. [4]

Для overlay-режима используется canvas, на котором можно рисовать прямоугольники, текст и служебные элементы. Такой подход дает возможность стирать исходный текст и выводить перевод в пределах выбранной области.

Ограничением Tkinter является более простой визуальный стиль по сравнению с современными UI-фреймворками. В рамках работы это ограничение компенсируется настройкой темы, цветов, шрифтов и структуры окна.

1.2.3 Выбор средств захвата изображения

Для захвата экрана используется библиотека MSS. Она предназначена для быстрого получения изображения экрана и подходит для задач, где требуется регулярно получать скриншот выбранной области. [5]

Важным требованием является возможность захвата не всего экрана, а конкретной области, которую выделил пользователь. Такой подход снижает объем данных, передаваемых на OCR, и ускоряет обработку.

Полученное изображение дополнительно обрабатывается с использованием Pillow. Применяются операции перевода в оттенки серого, автоконтраста и повышения контрастности. Эти операции не относятся напрямую к интерфейсу, но фронтенд запускает их как часть пользовательского сценария. [6]

Особенность приложения состоит в том, что overlay-окно не должно попадать во внутренний скриншот и распознаваться повторно. Для этого применяется логика исключения окна из захвата и очистки старого результата.

Таким образом, средства захвата изображения выбраны с учетом требований реального времени и взаимодействия с пользовательским интерфейсом.

1.2.4 Выбор средств распознавания и перевода

Для распознавания текста используется EasyOCR. Библиотека поддерживает несколько языков и позволяет получать не только текст, но и координаты распознанных блоков. Координаты необходимы для overlay-режима, так как перевод должен отображаться около соответствующего фрагмента. [7].

					ПГУ 1.090301.17.001 ПЗ	Лист
						29
Изм.	Лист	№ докум.	Подпись	Дат		

В программе предусмотрены языковые наборы для русского, английского, немецкого, польского, итальянского и французского языков. Для каждого выбранного языка подбирается соответствующий набор OCR-языков.

Для перевода используется deep-translator с обращением к GoogleTranslator. Для фронтенд-части важно, что пользователь выбирает языки через элементы интерфейса, а внутренний код преобразует эти значения в языковые коды. [8]

Так как OCR и перевод могут выполняться заметное время, интерфейс должен показывать состояние обработки и не блокировать главное окно. Для этого используются потоки и переменные состояния.

В рамках данной ВКР EasyOCR и переводчик рассматриваются как технологические модули, с которыми взаимодействует фронтенд. Основной акцент остается на том, как пользователь запускает обработку и как получает результат.

1.2.5 Выбор среды разработки

В качестве среды разработки может использоваться Visual Studio Code. Она удобна для работы с Python-файлами, поддерживает подсветку синтаксиса, встроенный терминал, расширения для анализа кода и запуск программы из рабочего каталога.

Для проекта, состоящего из одного основного Python-файла и внешних зависимостей, легкая среда разработки удобнее, чем тяжелая интегрированная IDE. Разработчик может быстро открывать файл, проверять изменения и запускать приложение.

Дополнительным преимуществом является наличие встроенного терминала, через который можно устанавливать зависимости, запускать скрипты и проверять синтаксис программы.

Среда разработки не является частью конечного продукта, но влияет на скорость реализации и отладки интерфейса. В дипломной работе она рассматривается как одно из средств разработки.

Использование Visual Studio Code соответствует характеру проекта: разработка выполняется на Python, а основная логика сосредоточена в пользовательском приложении.

1.2.6 Обмен данными между компонентами

В приложении отсутствует классический клиент-серверный обмен данными, характерный для веб-приложений. Вместо этого используется обмен между интерфейсными компонентами и локальными модулями обработки внутри одного процесса.

Главное окно хранит состояние выбранной области, выбранных языков, режима отображения, темы, шрифта, кэша переводов и активного потока обработки. Остальные компоненты получают эти данные через методы основного класса.

Область экрана передается в виде словаря с координатами left, top, width и height. Такой формат прост и напрямую подходит для передачи в функцию захвата изображения.

Результаты OCR передаются в виде распознанных текстовых блоков и координат. Для overlay-режима эти координаты используются для определения места стирания исходного текста и рисования перевода.

Для предотвращения повторной обработки одинакового изображения используется кэш и сигнатуры. Это помогает уменьшить лишние обновления интерфейса и повышает стабильность визуального вывода.

В таблице 1.4 приведены используемые технологии и средства разработки

					ПГУ 1.090301.17.001 ПЗ	Лист
						31
Изм.	Лист	№ докум.	Подпись	Дат		

Таблица 1.4 - Используемые технологии и средства разработки

Средство	Назначение	Использование в проекте
Python	Основной язык программы	Связывает GUI, захват изображения, OCR и перевод
Tkinter	Графический интерфейс	Главное окно, overlay, окно результата, Canvas
MSS	Захват экрана	Получение изображения выбранной прямоугольной области
Pillow	Подготовка изображения	Контраст, оттенки серого, масштабирование
EasyOCR	Распознавание текста	Получение текста и координат блоков
deep-translator	Машинный перевод	Перевод распознанных строк
keyboard	Горячие клавиши	Управление поверх других окон
ctypes / WinAPI	Системные настройки окон	Прозрачность, исключение overlay из захвата

1.3 Проектирование архитектуры и пользовательского приложения

Проектирование начинается с общей архитектуры, потому что пользовательское окно само по себе не решает задачу экранного перевода. Интерфейс должен передавать действия пользователя во внутренний слой обработки, получать от него результат и возвращать этот результат в понятном виде.

В данной программе backend не выделен в отдельный сервер. Он реализован как локальный вычислительный слой внутри настольного приложения. Такой подход соответствует характеру программы: обработка запускается на компьютере пользователя, а обмен между frontend и backend происходит через методы классов, внутренние переменные состояния и очередность вызовов.

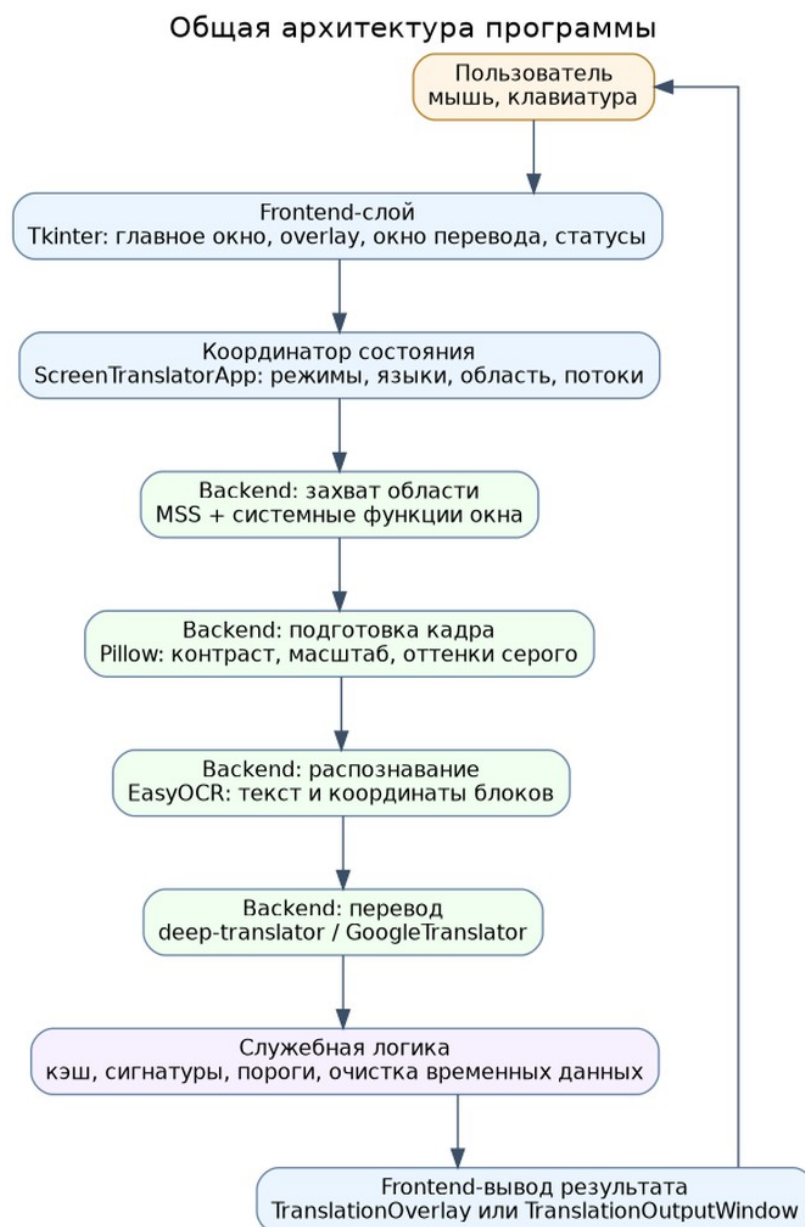


Рисунок 17 - Архитектура программы с backend-слоем

1.3.1 Общая архитектура программы и backend

Архитектуру программы можно разделить на пользовательский слой, слой координации, слой обработки и слой вывода результата. Пользовательский слой содержит главное окно, окно выбора области, элементы настроек и горячие клавиши. Слой координации представлен классом ScreenTranslatorApp, который связывает действия пользователя с внутренними операциями.

Backend включает захват области экрана, предобработку изображения, распознавание текста, группировку найденных фрагментов, перевод строк, кэширование результатов и очистку временных данных. Этот слой скрыт от пользователя, но именно он выполняет основную техническую работу.

Frontend получает от backend не абстрактный статус выполнения, а конкретный результат: текст перевода и, при overlay-режиме, координаты, по которым перевод нужно разместить на экране. Поэтому граница между слоями проходит не по файлам проекта, а по ответственности компонентов.

В таблице 1.5 представлен состав архитектурных слоев программы

Таблица 1.5 - Состав архитектурных слоев программы

Слой	Состав	Назначение
Frontend	Окна, кнопки, Canvas, статусы, настройки	Пользователь запускает сценарий и видит результат
Координатор	ScreenTranslatorApp, состояние, потоки, режимы	Связывает интерфейс и обработку
Backend обработки	MSS, Pillow, EasyOCR, Translator	Получает кадр, распознает и переводит текст
Служебная логика	Кэш, пороги, сигнатуры, очистка памяти	Снижает лишние OCR-запуски и поддерживает устойчивость
Слой вывода	TranslationOverlay, TranslationOutputWindow	Возвращает перевод пользователю

1.3.2 Граница ответственности фронтенд-разработки

Зона фронтенд-разработки определяется пользовательским сценарием. К этой части относятся главное окно, настройка языков, выбор режима вывода, окно выделения области, рамка выбранной области, overlay-слой, отдельное окно перевода, отображение статусов и горячие клавиши.

К внутренней обработке относятся OCR, перевод, подготовка изображения, кэширование и управление памятью. Эти части вызываются из пользовательского

сценария, но не рассматриваются как самостоятельная разработка новой модели распознавания или машинного перевода.

Такое разграничение важно для корректного описания результата. Разработанная часть не заменяет EasyOCR или переводчик, а делает их пригодными для практического использования в форме настольного приложения.

1.3.3 Проектирование пользовательского приложения

Пользовательское приложение строится вокруг нескольких окон. Главное окно отвечает за запуск сценариев и настройки. Окно выбора области временно перекрывает экран и позволяет мышью задать прямоугольник. Рамка области показывает границы выбранного участка. Результат выводится либо поверх исходного текста, либо в отдельном окне.

Такое разделение сделано для того, чтобы каждая часть интерфейса имела понятную роль. Если бы вся логика находилась только в главном окне, код стал бы сложнее сопровождать, а overlay-режим было бы труднее описывать и тестировать.

Диаграмма компонентов показывает, какие классы участвуют в пользовательском сценарии и как они связаны с координатором приложения.

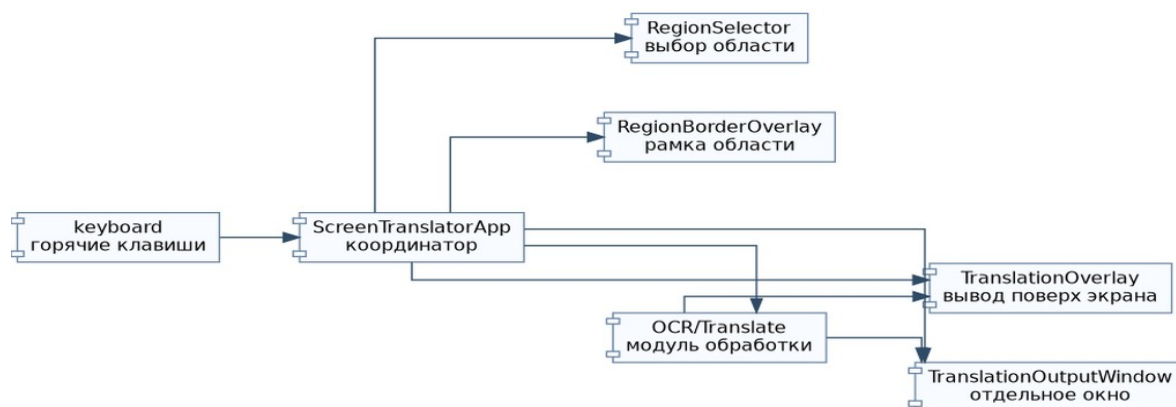


Рисунок 18 - Диаграмма компонентов frontend-части

1.3.4 Логические слои интерфейса

Для анализа интерфейс можно представить как набор логических слоев. Первый слой отвечает за отображение элементов. Второй слой принимает события мыши, кнопок и горячих клавиш. Третий слой хранит состояние приложения.

Четвертый слой передает данные во внутреннюю обработку. Пятый слой возвращает результат пользователю.

Такое представление помогает объяснить, почему в программе используется отдельный класс-координатор. Пользовательское действие проходит через несколько этапов, и приложение должно сохранять выбранные языки, область экрана, режим вывода, активность worker-потока и данные последнего результата.



Рисунок 19 - Логические слои приложения

1.4 Разработка приложения

Разработка выполнялась после определения требований и архитектуры. Основная задача заключалась в том, чтобы реализовать не разрозненные окна, а единый сценарий: выбор области, запуск обработки, ожидание результата, вывод перевода и очистка.

В коде используются отдельные классы для визуальных компонентов и один основной класс, который координирует их работу. Такой вариант проще объяснить

и тестировать, потому что назначение каждого класса соответствует конкретной части пользовательского сценария.

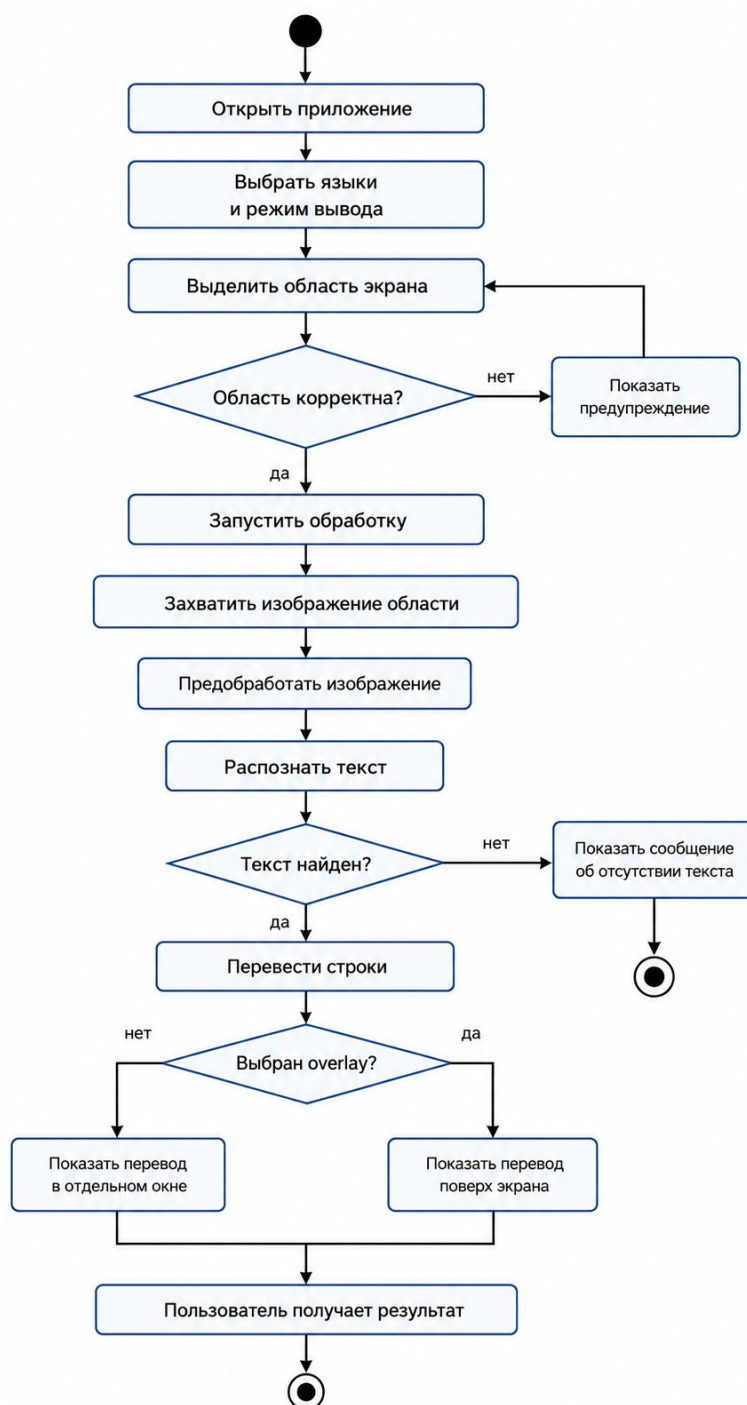


Рисунок 20 - Диаграмма деятельности

1.4.1 Реализация главного окна и координатора состояния

Главное окно является точкой входа в программу. Через него пользователь выбирает языки, режим вывода, запускает выбор области, стартует перевод,

останавливает обработку и открывает настройки. Внутри программы эти действия обрабатываются классом ScreenTranslatorApp.

Класс ScreenTranslatorApp хранит текущее состояние приложения. К таким данным относятся выбранная область экрана, выбранные языки, режим вывода, флаг активной обработки, ссылка на поток worker_thread, параметры overlay и последние результаты OCR.

При запуске перевода приложение проверяет, выбрана ли область, не работает ли уже другой поток, очищает старый результат и запускает обработку в отдельном потоке. Благодаря этому главное окно остается доступным для пользователя.

```
def start(self):
    if self.region is None:
        messagebox.showwarning("Ошибка", "Сначала выберите область экрана.")
        return
    if self.worker_thread and self.worker_thread.is_alive():
        self.set_status("Статус: перевод уже идёт; нажми «Старт с нуля» или «Стоп»")
        return

    self.lock_mode_switching_until_stop_clear()
    self.close_translation_window(save_state=True)
    self.worker_generation += 1
    current_generation = self.worker_generation
    self.stop_event.clear()
    self.last_full_text = ""
    self.last_source_signature = None
    self.last_overlay_items = []

    source_name = self.source_lang_var.get()
    target_name = self.target_lang_var.get()
    settings = {
        "source_name": source_name,
        "source_lang": TRANSLATOR_LANGUAGES[source_name],
        "target_lang": TRANSLATOR_LANGUAGES[target_name],
    }

    self.worker_thread = threading.Thread(
        target=self.worker_loop,
```

					ПГУ 1.090301.17.001 ПЗ	Лист
						38
Изм.	Лист	№ докум.	Подпись	Дат		

```

        args=(settings, current_generation),
        daemon=True,
    )
    self.worker_thread.start()
    self.set_status("Статус: перевод запущен")

```

1.4.2 Описание основных классов с примерами реализации

Класс RegionSelector отвечает за выбор области экрана. Он создает полноэкранное полупрозрачное окно, отслеживает нажатие мыши, движение курсора и отпускание кнопки. После завершения выбора координаты передаются обратно в основной класс через callback. UML-диаграмма классов приложения представлена на рисунке А.4.

Важная проверка выполняется сразу после выделения области. Если пользователь случайно выбрал слишком маленький фрагмент, программа не продолжает обработку и выводит предупреждение.

```

class RegionSelector(tk.Toplevel):
    def on_mouse_up(self, event):
        x1, y1 = self.start_x, self.start_y
        x2, y2 = event.x_root, event.y_root
        left = min(x1, x2)
        top = min(y1, y2)
        width = abs(x2 - x1)
        height = abs(y2 - y1)

        if width < 50 or height < 30:
            if self.cancel_callback is not None:
                self.cancel_callback("too_small")
            self.destroy()
            return "break"

        region = {
            "left": int(left),
            "top": int(top),
            "width": int(width),
            "height": int(height),
        }
        self.callback(region)
        self.destroy()
        return "break"

```

Класс RegionBorderOverlay показывает границы активной области. Пользователь видит, какой участок экрана будет обрабатываться, и может понять, что программа работает не со всем экраном, а только с выбранным фрагментом.

Класс TranslationOverlay реализует вывод перевода поверх исходного места. Для этого используется окно без стандартной рамки и Canvas. Окно делается верхним, а при необходимости исключается из захвата, чтобы программа не распознавала собственный перевод повторно.

```
class TranslationOverlay(tk.Toplevel):
    def set_region(self, region):
        self.region = region
        self.geometry(
            f'{region["width"]x{region["height"]}{region["left"]}{region["top"]}'
        )
        self.canvas.config(width=region["width"], height=region["height"])
        self.update_idletasks()
        set_click_through(self)
    def set_capture_exclusion(self, excluded: bool) -> bool:
        success = set_window_capture_exclusion(self, excluded)
        self.capture_excluded = bool(success and excluded)
        self.excluded_from_capture = self.capture_excluded
        return success

    def clear(self):
        self.canvas.delete("all")
```

Класс TranslationOutputWindow используется для отдельного режима вывода. Он удобен, когда текст длинный или когда overlay закрывает важные элементы исходного окна. В этом режиме пользователь может скопировать результат, изменить размер окна и продолжить работу с исходным приложением.

Связь классов строится через ScreenTranslatorApp. Например, после выбора области координатор сохраняет координаты, обновляет рамку и передает область в overlay-окно.

```
def on_region_selected(self, region):
    self.region_selection_in_progress = False
    self.restore_after_region_selection()
    should_restart = self.restart_after_region_selection
```

```

self.restart_after_region_selection = False
self.set_region(region)
if should_restart:
    self.root.after(180, self.start)

```

```

def set_region(self, region):
    self.region = region
    self.region_border.set_region(region)
    self.overlay.set_region(region)
    self.update_region_border_visibility()
    self.update_overlay_capture_exclusion()
    self.set_status("Статус: область выбрана")

```

В таблице 1.6 представлены основные классы приложения

Таблица 1.6 - Основные классы приложения

Класс	Назначение	Основные методы
ScreenTranslatorApp	Координация окон, состояний, потоков и запуска обработки	select_region, set_region, start, stop, worker_loop
RegionSelector	Выбор прямоугольной области экрана	on_mouse_down, on_mouse_move, on_mouse_up
RegionBorderOverlay	Показ рамки выбранной области	set_region, draw, show_border, hide_border
TranslationOverlay	Overlay-вывод результата поверх экрана	set_region, show_items, clear, start_spinner
TranslationOutputWindow	Отдельное окно перевода	set_translation, get_translation, start_processing

1.4.3 Реализация выбора области экрана

Выбор области реализован как отдельный режим работы. При его запуске главное окно временно скрывается, а поверх экрана появляется полупрозрачный слой. Пользователь мышью выделяет прямоугольник, после чего координаты возвращаются в ScreenTranslatorApp.

Использование отдельного класса упрощает обработку отмены. Если пользователь нажимает Escape или закрывает окно выбора, приложение возвращается в прежнее состояние без запуска OCR.

После успешного выбора область передается не только модулю захвата, но и рамке, а также overlay-окну. Благодаря этому все визуальные компоненты работают с одними и теми же координатами.

1.4.4 Реализация внутреннего сценария обработки

После запуска перевода поток обработки получает снимок выбранной области, формирует сигнатуру изображения, проверяет возможность повторного использования результата и при необходимости запускает OCR. Такой подход снижает количество лишних операций при неизменном тексте.

Перед передачей изображения в EasyOCR выполняется предварительная обработка. Изображение переводится в оттенки серого, улучшается контраст и ограничивается размер. Это снижает нагрузку на OCR и делает поведение программы более предсказуемым.

Распознанные фрагменты группируются в строки. Затем строки передаются переводчику. Если часть строк уже переводилась ранее, используется кэш, что уменьшает задержку при повторной обработке.

```
def run_easyocr(self, image: Image.Image, source_name: str, scale: int, min_conf: int):
    reader = self.get_easyocr_reader(source_name)
    processed, actual_scale = preprocess_image_for_easyocr(image, scale=scale)
    processed = processed.convert("RGB")
    np_image = np.array(processed)

    raw_results = reader.readtext(
        np_image,
        detail=1,
        paragraph=EASYOCR_PARAGRAPH_MODE,
        decoder="greedy",
        batch_size=1,
        workers=0,
        text_threshold=0.45,
        low_text=0.25,
        link_threshold=0.25,
```

```

)

fragments = []
confidences = []
for box, text, confidence in raw_results:
    text = clean_text(str(text))
    if has_meaningful_text(text) and confidence * 100 >= min_conf:
        fragments.append({"box": box, "text": text, "confidence": confidence *
100})
        confidences.append(confidence * 100)
    return self.group_ocr_fragments_into_lines(fragments), sum(confidences) /
max(1, len(confidences)),
    fragments

```

1.4.5 Реализация режимов вывода результата

В разрабатываемой программе предусмотрено два режима вывода. Overlay-режим предназначен для короткого текста, когда важно видеть перевод примерно в той же области, где находился оригинал. Оконный режим используется для длинного результата или для ситуаций, когда поверх исходного окна нельзя размещать дополнительные элементы.

Overlay-режим сложнее с технической точки зрения. Программа должна отрисовать перевод, очистить старый результат, не мешать кликам пользователя и не попадать в собственные скриншоты. Для этого применяются topmost-окно, прозрачный цвет, Canvas и системная настройка исключения окна из захвата.

В оконном режиме результат выводится в отдельном окне перевода. Это окно можно перемещать, изменять по размеру и использовать как обычное текстовое окно результата.

1.4.6 Реализация настроек и горячих клавиш

Настройки позволяют адаптировать программу под разные фоны и сценарии. Пользователь может выбрать тему, цвет фона, цвет текста, прозрачность, шрифт и режим вывода. Эти параметры особенно важны для overlay, потому что исходный экран может быть светлым, темным или цветным.

Горячие клавиши нужны для работы поверх других окон. Пользователь может не возвращаться к главному окну, если нужно заново выбрать область, повторить перевод, остановить обработку или показать рамку выбранной области.

Для защиты от конфликтов используется ограничение частоты нажатий. Если клавиша нажимается слишком часто, действие не выполняется повторно мгновенно. Это снижает риск одновременного запуска нескольких одинаковых операций.

В таблице 1.7 представлены основные горячие клавиши

Таблица 1.7 - Основные горячие клавиши

Клавиша	Действие
Print Screen	Показ overlay на скриншоте при специальном режиме
F6	Выбор области экрана
F7	Перезапуск перевода с нуля
F8	Остановка и очистка результата
F9	Показ или скрытие рамки области
1	Переключение в оконный режим
2	Переключение в overlay-режим

1.4.7 Руководство пользователя

Для начала работы пользователь запускает программу и выбирает исходный и целевой язык. Далее необходимо выбрать режим вывода результата: overlay или отдельное окно перевода.

После этого пользователь нажимает кнопку выбора области или использует горячую клавишу F6. На экране появляется полупрозрачный слой. Пользователь выделяет область с текстом, который требуется перевести. Если область выбрана корректно, появляется рамка выбранного участка.

Затем пользователь запускает перевод. В overlay-режиме результат появляется поверх исходного фрагмента. В оконном режиме главное окно скрывается, а перевод отображается в отдельном окне. При изменении текста на экране перевод можно запустить повторно.

Для остановки применяется команда остановки и очистки. Она завершает активный поток, убирает overlay и очищает временные данные. Если требуется работать поверх видео или игры, удобнее заранее выбрать область и управлять программой горячими клавишами.

В таблице 1.8 представлен порядок работы пользователя с программой.

Таблица 1.8 - Порядок работы пользователя с программой

Шаг	Действие пользователя	Результат
1	Запустить приложение	Открывается главное окно
2	Выбрать языки и режим вывода	Настройки сохранены в состоянии приложения
3	Выделить область экрана	Появляется рамка выбранной области
4	Запустить перевод	Статус меняется на выполнение обработки
5	Получить результат	Перевод отображается в overlay или отдельном окне
6	Остановить и очистить	Результат убран, программа готова к новому запуску

1.4.8 Тестирование и отладка

Тестирование выполнялось по основным пользовательским сценариям. Проверялись запуск приложения, выбор области, отмена выбора, реакция на слишком маленькую область, запуск перевода, вывод результата в двух режимах, остановка и очистка.

Отдельно проверялась устойчивость интерфейса при повторных действиях. Пользователь может несколько раз менять область, запускать перевод заново, переключать режим вывода и останавливать обработку. В этих ситуациях программа не должна оставлять старый overlay и не должна запускать несколько одинаковых worker-потоков.

Проверялись настройки внешнего вида. Для overlay и отдельного окна результата изменялись фон, цвет текста, тема и прозрачность. При этом результат должен оставаться читаемым на разных фонах.

Результаты проверки показывают, что основные сценарии выполняются корректно. Ограничения текущей версии связаны с зависимостью от OCR-библиотеки, скорости перевода и дальнейшей упаковкой приложения в удобный установочный файл.

В таблице 1.9 представлена проверка основных пользовательских сценариев.

Таблица 1.9 - Проверка основных пользовательских сценариев

Проверка	Ожидаемый результат	Итог
Запуск приложения	Главное окно открывается, элементы управления доступны	Выполнено
Выбор области	Координаты сохраняются, рамка отображается	Выполнено
Слишком маленькая область	Обработка не запускается, выводится предупреждение	Выполнено
Overlay-режим	Результат отображается поверх выбранной области	Выполнено
Оконный режим	Перевод выводится в отдельном окне	Выполнено
Остановка и очистка	Активный поток останавливается, overlay очищается	Выполнено
Горячие клавиши	Команды выполняются без возврата к главному окну	Выполнено
Повторный запуск	Старый результат очищается перед новым запуском	Выполнено

1.4.9 Последовательность взаимодействия компонентов

Последовательность взаимодействия показывает, как пользовательское действие проходит через программу. Сначала пользователь выбирает область и

запускает перевод. Затем координатор приложения обращается к модулю захвата, получает изображение, передает его в OCR и переводчик, после чего отправляет результат в слой вывода.

Важный момент состоит в том, что сценарий не должен завершаться на внутреннем компоненте. Для пользователя результатом является не факт вызова переводчика, а видимый перевод на экране. Поэтому на диаграмме последовательности последняя стрелка возвращается к пользователю через overlay или окно перевода.

Если во время обработки возникает ошибка, координатор не завершает приложение аварийно, а обновляет статус. Такой подход делает поведение программы более понятным для пользователя.

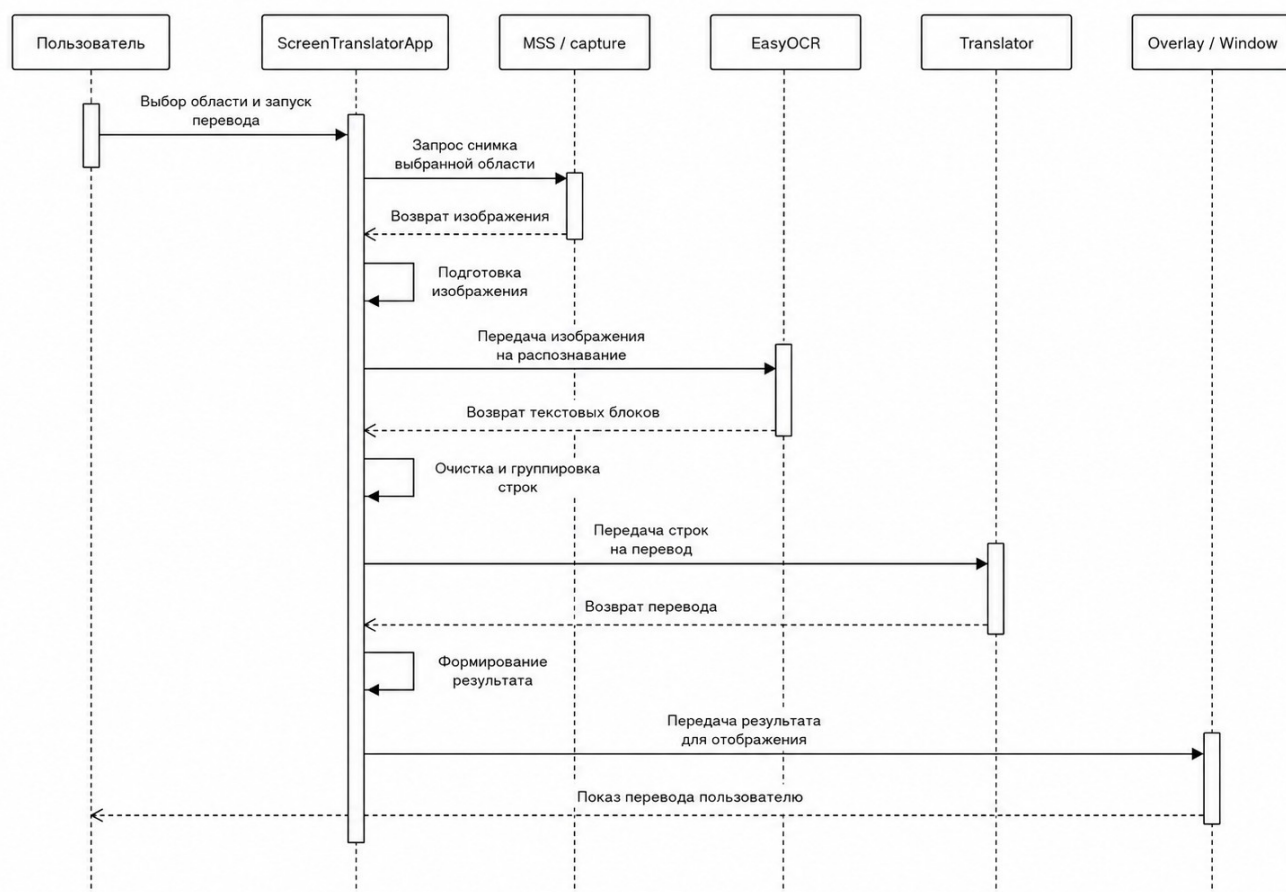


Рисунок 21 - Диаграмма последовательности

1.4.10 Анализ полученных результатов и развитие

В результате разработки пользователь получил единый интерфейс для экранного перевода. Программа закрывает основную проблему предметной области: текст на экране можно перевести без ручного создания скриншота, копирования результата OCR и перехода в отдельный переводчик.

К сильным сторонам реализации можно отнести наличие двух режимов вывода, выбор области экрана, рамку активной области, горячие клавиши, настройки читаемости и защиту от повторного захвата собственного overlay. Эти элементы относятся именно к frontend-части, но они заметно влияют на удобство всей программы.

Ограничением текущей версии остается зависимость от внешних библиотек распознавания и перевода. При сложном фоне, низком качестве текста или отсутствии доступа к переводчику результат может быть хуже. Для дальнейшего развития целесообразно добавить историю переводов, профили областей, упаковку в установочный файл и более современный визуальный стиль интерфейса.

Поставленная цель достигнута: разработана frontend-часть настольного приложения, которая связывает действия пользователя с внутренним backend-слоем захвата, OCR и перевода и возвращает результат пользователю в удобном режиме вывода.

					ПГУ 1.090301.17.001 ПЗ	Лист
						48
Изм.	Лист	№ докум.	Подпись	Дат		

2 Стартap-проект Screen Translate

2.1 Обоснование актуальности

В современном мире информации пользователи часто сталкиваются с информацией на иностранном языке. Такой контент присутствует в интерфейсах программ, веб-сервисах, технической документации, учебных материалах, презентациях, PDF-файлах, видеокурсах, компьютерных играх, субтитрах и изображениях. Значительная часть этой информации отображается на экране как изображение или как элемент динамического интерфейса, поэтому ее нельзя быстро выделить мышью, скопировать и вставить в обычный переводчик.

Из-за этого у пользователя возникает повторяющаяся проблема: текст виден на экране, но технически недоступен для быстрого перевода. Чтобы понять фрагмент, приходится делать скриншот, открывать отдельный OCR-сервис, распознавать текст, копировать результат, переходить в переводчик и только после этого возвращаться к исходному окну. Такой путь занимает лишнее время, разрушает рабочую ситуацию и делает перевод неудобным в тех случаях, когда информация нужна сразу.

Актуальность проекта усиливается глобализацией цифрового контента. По данным исследования DeerpL, менее 20% мирового населения свободно владеет английским языком, хотя английский часто используется как универсальный язык бизнеса, образования и программного обеспечения. Это означает, что большая часть пользователей нуждается в переводе при работе с иностранными интерфейсами, учебными материалами и профессиональной документацией.

Коммерческая значимость локализации подтверждается исследованием CSA Research: 76% онлайн-покупателей предпочитают получать информацию о товарах на родном языке, а 40% не готовы совершать покупку на сайте, если информация представлена на чужом языке. Для программных продуктов и цифровых сервисов это означает, что язык интерфейса влияет не только на удобство, но и на готовность пользователя работать с продуктом.

Для рынка программного обеспечения языковой барьер особенно заметен. Многие профессиональные инструменты, обучающие платформы, технические каталоги, CRM-системы, библиотеки и документация создаются на английском языке. Даже если пользователь в целом понимает английский, постоянное переключение между окнами ради перевода небольших фрагментов снижает скорость работы и увеличивает когнитивную нагрузку.

Развитие искусственного интеллекта, OCR-технологий и нейросетевого машинного перевода делает возможным создание более удобного инструмента. Пользователь уже привык к автоматическому переводу, но большинство доступных решений предполагает, что текст уже находится в копируемом виде. В проекте Screen Translate решается более широкий сценарий: сначала программа получает текст из изображения экрана, а затем переводит его.

Проект Screen Translate направлен на создание программного обеспечения для захвата, распознавания и перевода текстового материала в реальном времени. Пользователь выбирает область экрана, программа анализирует изображение, распознает текст, переводит его на выбранный язык и показывает результат в отдельном окне или поверх исходного фрагмента. За счет этого перевод становится частью рабочего процесса, а не отдельной последовательностью ручных операций.

Таким образом, актуальность стартап-проекта определяется ростом объема иностранного цифрового контента, регулярной потребностью пользователей в переводе, недостаточностью обычных переводчиков для экранного текста и возможностью объединить OCR и машинный перевод в один удобный продукт.

В рамках данной работы особое значение имеет фронтенд-разработка. Именно через интерфейс пользователь воспринимает ценность продукта: насколько быстро можно выбрать область, насколько понятно отображается статус, насколько читаем overlay и не мешает ли программа исходному окну.

2.2 Описание и обоснование выбора ВКР в формате ВКРС

Выбор формата «Стартап как диплом» для данного проекта является обоснованным, поскольку работа имеет выраженный продуктовый и коммерческий

					ПГУ 1.090301.17.001 ПЗ	Лист
						50
Изм.	Лист	№ докум.	Подпись	Дат		

характер. Она не ограничивается теоретическим исследованием технологии распознавания или перевода, а направлена на создание прикладного программного продукта с понятной целевой аудиторией и перспективой коммерциализации.

Проект относится к технологическим стартапам в сфере информационных и коммуникационных технологий. Он основывается на программной разработке, использовании библиотек захвата экрана, OCR-распознавания, машинного перевода, обработки изображений, кэширования и пользовательского взаимодействия. В результате формируется продукт, который может применяться в учебе, профессиональной деятельности, работе с иностранным программным обеспечением и повседневном использовании.

Формат ВКРС позволяет рассмотреть проект не только как техническую разработку, но и как бизнес-проект. Для стартап-части важно показать существующую проблему, предлагаемое решение, целевой рынок, конкурентов, бизнес-модель, финансовые предпосылки, команду, развитие и риски. Screen Translate соответствует этой логике, поскольку решает конкретную боль пользователей и имеет несколько возможных моделей монетизации.

Выбор ВКРС также позволяет разделить общий проект на зоны ответственности. Продукт включает клиентскую часть, отвечающую за пользовательский сценарий и отображение результата, и вычислительную часть, отвечающую за захват, OCR, перевод, кэширование, производительность и стабильность. Такое разделение делает проект удобным для командной разработки и позволяет каждому участнику показать собственный вклад.

В отличие от обычной дипломной работы, формат стартапа требует не только описать алгоритмы, но и показать, зачем продукт нужен рынку, кто будет его использовать, какие функции являются коммерчески значимыми и какие риски необходимо контролировать. Это делает работу более приближенной к реальному процессу создания программного продукта.

Таким образом, выбор формата ВКРС является целесообразным, потому что проект Screen Translate имеет практическую значимость, рыночный потенциал, рабочую техническую реализацию и понятную траекторию дальнейшего развития.

					ПГУ 1.090301.17.001 ПЗ	Лист
						51
Изм.	Лист	№ докум.	Подпись	Дат		

В рамках темы «Фронтенд-разработка» основной акцент сделан на клиентской части продукта. Для данного проекта фронтенд - это не только внешний вид главного окна, но и весь пользовательский путь: выбор области, настройка языков, запуск и остановка перевода, визуальный статус, overlay-отображение, работа горячих клавиш и понятная реакция программы на ошибки.

Фронтенд-часть особенно важна для стартапа, потому что именно она определяет, будет ли пользователь воспринимать продукт как быстрый и удобный. Если интерфейс перегружен, перевод может появиться где угодно или overlay плохо читается, даже качественный OCR и перевод не превращаются в полезный продукт.

2.3 Описание существующей проблемы на рынке

Основная проблема рынка заключается в том, что существующие инструменты перевода плохо работают в ситуациях, когда текст нельзя быстро скопировать. Пользователь может видеть иностранную надпись в программе, игре, видео, PDF-документе, презентации или изображении, но обычный переводчик не может напрямую обработать этот текст без дополнительного этапа распознавания.

На практике пользователь вынужден выполнять длинную последовательность действий: сделать скриншот, сохранить или передать его в OCR-сервис, распознать текст, проверить ошибки распознавания, скопировать результат, открыть переводчик, вставить текст, получить перевод и вернуться обратно. Даже если каждый шаг кажется простым, суммарно процесс занимает от одной до двух минут и отвлекает от основной задачи.

Проведенные интервью среди студентов университета показывают наличие у пользователей проблемы. Из 30 опрошенных респондентов 18 сталкиваются с текстом без возможности копирования не менее трех раз в неделю. 16 из 30 предпочли бы видеть перевод поверх исходного места, 20 из 30 отметили важность обработки только выбранной области экрана, а 14 из 30 готовы рассматривать платную подписку при условии стабильного OCR и удобного сценария использования.

					ПГУ 1.090301.17.001 ПЗ	Лист
						52
Изм.	Лист	№ докум.	Подпись	Дат		

Проблема наиболее заметна в образовательном сценарии. Студент или школьник смотрит иностранный видеокурс, читает презентацию, научную статью или PDF-файл. Часто важная информация находится не в копируемом тексте, а на слайде, в видеокадре или в защищенном документе. Ручной перевод фрагментов мешает обучению, заставляет ставить видео на паузу и переключаться между приложениями.

В профессиональной сфере проблема проявляется у IT-специалистов, инженеров, сотрудников компаний и исследователей. Иностранные интерфейсы, сообщения об ошибках, технические инструкции, API-документация, CRM-системы, каталоги и веб-сервисы часто требуют понимания прямо в момент работы. Пользователю нужен быстрый перевод, который не разрушает рабочий контекст.

В бытовом и развлекательном сценарии проблема возникает в играх без локализации, видео с субтитрами, изображениях, скриншотах, уведомлениях и меню программ. В таких случаях пользователь чаще всего не хочет заниматься полноценной локализацией, ему нужно просто быстро понять конкретный фрагмент текста.

Существующие онлайн-переводчики решают задачу перевода, но не решают задачу получения текста из экрана. Браузерные переводчики ограничены веб-страницами и не работают в большинстве десктопных приложений. OCR-утилиты извлекают текст, но часто не дают мгновенного перевода и не возвращают результат в исходный контекст. Поэтому на рынке остается ниша для инструмента screen translation.

Проблема формулируется как языковой барьер любого видимого текста. Пользователю нужен единый сценарий: выбрать область экрана, получить распознавание и перевод, увидеть результат там, где он был нужен, и продолжить работу без лишних переключений.

2.4 Описание решения проблемы и продуктового решения

2.4.1 Наименование продукта

					ПГУ 1.090301.17.001 ПЗ	Лист
						53
Изм.	Лист	№ докум.	Подпись	Дат		

Наименование продукта - Screen Translate. Это программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени. В рамках данной выпускной квалификационной работы рассматривается направление «Фронтенд-разработка».

2.4.2 Назначение и сфера применения

Продукт предназначен для пользователей, которым необходимо быстро переводить текст, видимый на экране. Основная особенность заключается в том, что программа работает не только с копируемым текстом, но и с изображениями, интерфейсами программ, PDF, видео, презентациями и играми.

Сфера применения включает учебные материалы, веб-сайты, PDF-документы, презентации, зарубежное программное обеспечение, техническую документацию, игры, видео и изображения. На первом этапе продукт ориентирован на Windows-сценарий, потому что именно в среде настольных приложений часто требуется перевод произвольной области экрана.

2.4.3 Пользовательский сценарий

Базовый пользовательский сценарий состоит из нескольких последовательных действий. Сначала пользователь открывает иностранный интерфейс, видеоурок, PDF-файл, игру или другое окно с текстом. Затем он нажимает горячую клавишу или кнопку выбора области и выделяет прямоугольник на экране.

После этого система распознает текст, переводит его на выбранный язык и показывает результат. В обычном режиме перевод выводится в отдельном окне. В overlay-режиме перевод отображается поверх исходного текста или рядом с ним, сохраняя визуальный контекст. Пользователь может остановить перевод, перезапустить его, скрыть или показать рамку области, изменить режим вывода и скопировать результат.

2.4.4 Фронтенд-зона ответственности

Фронтенд-разработка в проекте включает несколько ключевых блоков. Первый блок - интерфейс выбора области. Он отвечает за полупрозрачное окно

выделения, рамку выбора, обработку мыши, отмену по ESC, проверку минимального размера области и сохранение координат.

Второй блок - главное окно приложения. Оно содержит основные элементы управления: выбор режима вывода, выбор языков в настройках, кнопку выбора области, старт, стоп, включение рамки и переход к настройкам. Важная задача интерфейса - не перегружать пользователя техническими параметрами, оставляя на первом экране только основные действия.

Третий блок - окно результата в обычном режиме. Оно должно быть поверх остальных окон, позволять выделять и копировать текст, менять размер и положение, настраивать шрифт, цвет фона, цвет текста и прозрачность. При этом окно не должно выглядеть как обычный редактор: ручное изменение текста блокируется, но копирование остается доступным.

Четвертый блок - overlay результата. Это наиболее важная часть фронтенда, потому что она отличает продукт от обычной связки OCR и переводчика. Overlay должен быть читаемым, не мешать исходному контенту, учитывать цвет фона и текста, подбирать размер шрифта, избегать наложений и сохранять ощущение, что перевод встроен в исходное окно.

Пятый блок - визуальная обратная связь. Пользователь должен понимать, что программа делает в данный момент: ожидает выбора области, выполняет перевод, показывает результат, остановлена или очищает память. Для этого используются статусы, индикатор загрузки, надпись «Перевод...» и состояния кнопок.

2.4.5 Функциональные возможности реализованной программы

На текущем этапе разработана рабочая версия приложения Screen Translator на языке Python. Интерфейс реализован с использованием Tkinter. В коде присутствуют отдельные окна и классы для выбора области, рамки области, overlay-отображения и окна перевода. Это позволяет разделить пользовательские задачи и поддерживать более понятную структуру приложения.

Реализованы два режима вывода. Первый режим - отдельное окно перевода, которое отображается поверх остальных окон, может перемещаться пользователем,

содержит настройки шрифта, цвета и прозрачности. Второй режим - overlay поверх выбранной области, где перевод отображается прямо в контексте исходного текста.

Программа поддерживает горячие клавиши: Print Screen используется для режима скриншота, F6 - для выбора области, F7 - для перезапуска перевода, F8 - для остановки и очистки, F9 - для включения или скрытия рамки, клавиши 1 и 2 - для переключения обычного и overlay-режима. Для защиты от случайных повторных нажатий реализован cooldown горячих клавиш.

В overlay-режиме фронтенд получает от OCR координаты фрагментов, оценивает высоту и ширину текстовых блоков, подбирает шрифт, перенос строк, цвет текста и цвет подложки. Для лучшей читаемости используются функции оценки цветов исходного изображения, проверки контраста и уменьшения шрифта при возможных пересечениях.

Особенность фронтенда состоит в том, что overlay остается видимым для пользователя, но может быть исключен из захвата экрана Windows, чтобы программа не распознавала собственный перевод повторно. При необходимости пользователь может временно разрешить попадание overlay в скриншот.

2.4.6 Конкурентоспособность и инновационность

Конкурентоспособность продукта определяется тем, что он объединяет несколько функций в одном пользовательском сценарии. Обычный онлайн-переводчик требует копируемого текста. OCR-утилита извлекает текст, но не всегда переводит его и не возвращает результат в исходный контекст. Screen Translate объединяет выбор области, распознавание, перевод и отображение результата.

Инновационность фронтенд-части заключается в overlay-сценарии и минимизации пользовательских действий. Пользователь получает перевод прямо там, где возникла проблема, а не в отдельном сервисе после нескольких ручных шагов. Это делает продукт ближе к реальному поведению человека при работе с иностранным экранным контентом.

Дополнительным преимуществом является privacy-by-design: пользователь выбирает только нужную область экрана. Визуальная рамка, статусы и настройки

					ПГУ 1.090301.17.001 ПЗ	Лист
						56
Изм.	Лист	№ докум.	Подпись	Дат		

помогают показать, что программа не анализирует весь экран, а работает с ограниченным фрагментом.

2.4.7 Степень готовности продукта, поставка и обслуживание

Степень готовности продукта можно определить как рабочий прототип, близкий к MVP. В программе реализованы основные пользовательские функции: выбор области экрана, OCR-распознавание, перевод, обычный режим, overlay-режим, горячие клавиши, рамка выбранной области, настройки окна перевода и остановка с очисткой результата.

Для коммерческого запуска требуется доработка стабильности, тестирование на разных типах экранов, упаковка приложения, подготовка инструкции, настройка установщика, улучшение обработки ошибок и юридическая подготовка политики обработки данных. Поставка продукта предполагается в цифровом виде: установочный файл, архив приложения или корпоративная версия с настройками.

Сервисное обслуживание должно включать исправление ошибок, улучшение overlay-режима, обновление OCR/MT-интеграций, добавление языков, поддержку новых сценариев и консультации для B2B-клиентов.

2.5 Общая характеристика стартап-проекта и сферы деятельности

2.5.1 Основная бизнес-идея

Основная бизнес-идея проекта заключается в создании продукта, который позволяет переводить любой видимый текст на экране без ручного копирования. Пользователь получает инструмент, который сокращает путь от возникновения языкового барьера до понимания текста. Вместо последовательности «скриншот - OCR - переводчик - возврат в окно» предлагается единая цепочка внутри одного приложения.

Ценностное предложение формулируется следующим образом: Screen Translate помогает быстро понимать иностранный текст в любом окне Windows, сохраняя контекст работы и уменьшая количество ручных действий. Для частного пользователя это экономия времени, для студента - ускорение обучения, для IT-

					ПГУ 1.090301.17.001 ПЗ	Лист
						57
Изм.	Лист	№ докум.	Подпись	Дат		

специалиста - более удобная работа с документацией и интерфейсами, для организации - повышение продуктивности сотрудников.

На первом этапе продукт ориентирован на персональные компьютеры под Windows, поскольку именно в этой среде часто встречаются игры, PDF-просмотрщики, десктопные программы, корпоративные системы и технические утилиты, где встроенного перевода нет.

2.5.2 Сфера деятельности

Сфера деятельности проекта относится к информационным и коммуникационным технологиям, разработке программного обеспечения, инструментам продуктивности, OCR-решениям и машинному переводу. По своей природе продукт находится на стыке нескольких рынков: языковые сервисы, распознавание текста, переводческие технологии и desktop productivity software.

Проект не требует физического производства или складской логистики. Его результатом является цифровой продукт, который может распространяться в виде установочного файла, подписки, лицензии для организаций или индивидуальной версии для конечного пользователя.

2.5.3 Целевая аудитория

Целевая аудитория продукта включает студентов и школьников, использующих иностранные образовательные ресурсы, видеолекции, презентации, онлайн-курсы и PDF-материалы. Для них ключевая ценность состоит в ускорении обучения и снижении количества ручных действий.

Вторая группа - IT-специалисты, инженеры и технические сотрудники. Они регулярно сталкиваются с англоязычными сообщениями об ошибках, документацией, интерфейсами программ, настройками серверов, API-инструкциями и обучающими материалами. Для этой группы важны скорость, точность и возможность перевода прямо в рабочем контексте.

Третья группа - сотрудники компаний, которые работают с зарубежным программным обеспечением, CRM-системами, техническими каталогами, инструкциями и обучающими материалами. В этом сегменте возможны B2B-лицензии и пилотные внедрения.

					ПГУ 1.090301.17.001 ПЗ	Лист
						58
Изм.	Лист	№ докум.	Подпись	Дат		

Четвертая группа - частные пользователи, геймеры, исследователи и люди с недостаточной языковой подготовкой. Для них продукт полезен при работе с играми, видео, скриншотами, уведомлениями и иностранными интерфейсами.

2.5.4 Текущее состояние разработки

На текущем этапе создан рабочий прототип приложения Screen Translator на языке Python. Программа демонстрирует полный пользовательский сценарий: выбор области экрана, захват изображения, распознавание текста, перевод и вывод результата. Реализованы обычный режим отображения в отдельном окне и overlay-режим поверх выбранной области.

Прототип поддерживает русский, английский, немецкий, польский, итальянский и французский языки. Для распознавания используется EasyOCR, для перевода - GoogleTranslator через библиотеку deep_translator. Захват экрана выполняется с помощью mss, обработка изображений - через PIL, NumPy и OpenCV. Интерфейс реализован на Tkinter.

Продукт находится на стадии MVP-прототипа. Основные функции уже работают, но для коммерческого запуска требуется дополнительное тестирование на разных типах экранов, улучшение стабильности, упаковка приложения, подготовка инструкции и проверка юридических аспектов обработки данных.

2.6 Объем, структура и развитие рынка

Целевой рынок проекта можно рассматривать как пересечение нескольких направлений: языковые сервисы, OCR-технологии, машинный перевод, инструменты продуктивности и программное обеспечение для работы с экраным контентом. Это не один узкий рынок, а ниша на стыке технологий, где пользовательская потребность формируется из необходимости быстро понимать иностранный цифровой контент.

Рынок языковых сервисов является крупным и растущим. Оценка language services составляет около 75,5 млрд долларов в 2026 году, а дальнейший рост поддерживается глобализацией контента, дистанционным обучением,

					ПГУ 1.090301.17.001 ПЗ	Лист
						59
Изм.	Лист	№ докум.	Подпись	Дат		

международной работой, развитием ИИ-перевода и увеличением количества мультязычных цифровых продуктов.

Отдельно развивается рынок OCR. Для проекта он важен, потому что без распознавания изображение экрана не может быть превращено в переводимый текст. В прогнозах по OCR указывается потенциал роста до 32,9 млрд долларов к 2030 году. OCR применяется в документообороте, автоматизации бизнеса, распознавании изображений, мобильных приложениях и инструментах доступности.

Рынок машинного перевода также растет благодаря развитию нейросетевых моделей. Прогнозируемая оценка рынка машинного перевода составляет около 2,7 млрд долларов к 2030 году. Пользователи уже привыкли к автоматическому переводу, но большинство инструментов предполагает, что исходный текст уже доступен в цифровом виде. Screen Translate дополняет этот рынок этапом извлечения текста из изображения экрана.

Структура рынка применительно к проекту состоит из нескольких групп решений. Первая группа - онлайн-переводчики, такие как Google Translate, DeepL и Яндекс Переводчик. Они сильны качеством перевода, но требуют копируемого текста. Вторая группа - OCR-утилиты, которые извлекают текст, но часто не обеспечивают удобный перевод и вывод результата в контексте. Третья группа - браузерные переводчики, ограниченные веб-страницами. Четвертая группа - приложения экранного перевода, наиболее близкие к проекту, но часто ограниченные по скорости, языкам, настройкам или удобству overlay-сценария.

Потенциальная емкость аудитории достаточно широка. Продукт может быть полезен всем пользователям, которые работают с иностранным экранным контентом. Однако на начальном этапе целесообразно сфокусироваться на более узких сегментах: студенты, IT-специалисты, пользователи зарубежного ПО, сотрудники организаций и геймеры. Эти группы чаще сталкиваются с текстом, который нельзя скопировать, и быстрее понимают ценность продукта.

Первые три года развития проекта могут быть ориентированы на постепенное подтверждение спроса. На первом этапе важно довести MVP до устойчивой версии

					ПГУ 1.090301.17.001 ПЗ	Лист
						60
Изм.	Лист	№ докум.	Подпись	Дат		

и собрать beta-пользователей. На втором этапе - получить первых платных пользователей и B2B-пилоты. На третьем этапе - масштабировать подписку и лицензии, расширить языки и улучшить стабильность продукта.

Таким образом, рынок для проекта является достаточно большим, но входить в него нужно через узкую и понятную нишу: перевод любого видимого текста на экране персонального компьютера. В этой нише конкурентным фактором будет не только качество перевода, но и скорость сценария, приватность, стабильность и удобство использования.

Для фронтенд-разработки рыночная возможность связана с качеством пользовательского опыта. Если продукт будет восприниматься как легкий и быстрый инструмент, он сможет конкурировать даже с более известными переводчиками, потому что решает другой сценарий - перевод прямо поверх исходного контента.

2.7 Анализ конкурентов и аналогов

В рамках анализа конкурентов можно выделить несколько типов решений. Первый тип - универсальные онлайн-переводчики. Они обладают высоким качеством перевода и известным брендом, но требуют, чтобы пользователь уже имел текст в копируемом виде. Если текст находится в изображении, игре, видео или desktop-интерфейсе, пользователь должен сначала извлечь его отдельно.

Второй тип - Google Lens и аналогичные визуальные переводчики. Они подтверждают существование спроса на перевод изображения, а не только копируемого текста. Однако такие решения чаще ориентированы на мобильные сценарии и не всегда удобны для постоянной работы на ПК. Кроме того, пользовательский сценарий может требовать переключения устройства или отдельного приложения.

Третий тип - OCR-утилиты и программы извлечения текста, например Text Grab и похожие инструменты. Они быстро распознают текст, но часто завершают задачу на этапе копирования. Пользователю все равно нужно отдельно вставить

текст в переводчик и затем вернуться к исходному окну. Screen Translate отличается тем, что объединяет OCR, перевод и вывод результата в одной цепочке.

Четвертый тип - браузерные переводчики и автоматические субтитры. Они удобны для веб-страниц и отдельных видеосервисов, но не работают с произвольными окнами Windows, играми, PDF-просмотрщиками, приложениями и скриншотами. Их ограничение связано не только с качеством перевода, но и с областью применения.

Пятый тип - существующие screen translator-приложения. Они наиболее близки к проекту по идее, но могут иметь ограничения по языкам, скорости, качеству OCR, приватности, понятности настроек или стабильности работы на разных типах экранов. Для нового продукта важно конкурировать не только набором функций, но и простотой сценария.

Конкурентная стратегия проекта заключается в фокусе на Windows-сценарии, выбранной области экрана, быстром переводе, overlay-выводе, горячих клавишах и privacy-by-design. Вместо попытки конкурировать со всеми переводчиками сразу проект занимает конкретную нишу: мгновенный перевод экранного текста без ручного копирования.

Возможные конкурентные действия крупных игроков включают добавление более удобного экранного перевода в существующие продукты. Ответом проекта должен стать узкий фокус, быстрая доработка под реальные сценарии пользователей, поддержка B2B-пилотов, гибкость настроек и развитие тех функций, которые крупные универсальные сервисы часто не делают приоритетом.

С точки зрения фронтенда ключевые критерии сравнения включают количество действий пользователя, удобство выбора области, понятность режимов, читаемость overlay, наличие горячих клавиш, возможность настройки внешнего вида и сохранение исходного контекста. Именно эти параметры должны стать основой конкурентного преимущества продукта.

2.8 Бизнес-модель и команда стартап-проекта

2.8.1 Бизнес-модель

					ПГУ 1.090301.17.001 ПЗ	Лист
						62
Изм.	Лист	№ докум.	Подпись	Дат		

Бизнес-модель проекта строится на комбинированном подходе B2C и B2B. Для конечных пользователей целесообразна freemium-модель: базовая версия дает возможность проверить основной сценарий, а платная подписка открывает расширенные возможности, повышенную стабильность, историю переводов, горячие клавиши, расширенные языки и дополнительные настройки.

Базовый тариф Free может включать ограничение по количеству минут, символов или переводов, а также базовый OCR и перевод. Такой тариф нужен для снижения барьера входа и проверки пользовательской ценности. Pro-тариф по цене 299 рублей в месяц может включать отсутствие лимитов базового сценария, overlay, историю и горячие клавиши. Pro+ по цене 499 рублей в месяц может включать расширенные языки, экспорт, приоритетную обработку и дополнительные настройки. Team-тариф по цене 799 рублей за пользователя в месяц может включать общие настройки, управление пользователями и документы для оплаты.

B2B-модель предполагает продажу лицензий организациям. Для компаний продукт может быть полезен как инструмент работы с иностранным ПО, документацией, техническими каталогами и обучающими материалами. Корпоративная лицензия может начинаться от 49 000 рублей в год и включать поддержку, пилотное внедрение, настройку под требования организации и консультации по использованию.

Логика дохода выглядит следующим образом: бесплатный вход в продукт позволяет набрать пользователей и получить обратную связь, затем часть пользователей переходит в Pro, а корпоративные сценарии развиваются в Team и B2B-лицензии. Для проекта особенно важно подтвердить, что overlay, скорость и приватность являются функциями, за которые пользователь готов платить.

2.8.2 Каналы продаж и продвижения

На начальном этапе продвижение целесообразно строить через демонстрацию конкретных сценариев. Потенциальный пользователь должен увидеть, что программа переводит текст из видео, игры, PDF, программы или технической документации без ручного копирования. Для этого подходят короткие

					ПГУ 1.090301.17.001 ПЗ	Лист
						63
Изм.	Лист	№ докум.	Подпись	Дат		

демонстрационные видео, публикации в тематических сообществах, материалы для студентов и IT-специалистов, а также описание кейсов использования.

Для B2C-сегмента каналами могут быть тематические форумы, социальные сети, студенческие сообщества, публикации на площадках для разработчиков и каталоги программ. Для IT-аудитории важно показать работу с документацией, ошибками, интерфейсами и техническими статьями.

Для B2B-сегмента каналами могут стать прямые обращения в образовательные организации, IT-компании, центры технической поддержки, команды внедрения зарубежного ПО и организации, где сотрудники регулярно работают с иностранными интерфейсами. В этом сегменте важны пилоты, понятная политика приватности и возможность настройки под требования компании.

2.8.3 Команда стартап-проекта

Команда проекта может быть представлена несколькими ролями. Первая роль - продуктовая и проектная. Она отвечает за формулировку ценностного предложения, интервью с пользователями, проверку гипотез, подготовку beta-теста, работу с B2B-пилотами, финансовую модель и упаковку продукта.

Вторая роль - разработка клиентской части. Она отвечает за интерфейс, пользовательский сценарий, настройку языков, обработку действий пользователя, отображение результата, понятные статусы, визуальную обратную связь и удобство работы с приложением.

Третья роль - разработка вычислительной части. Она отвечает за захват изображения, OCR, предобработку, машинный перевод, кэширование, оптимизацию задержек, управление памятью, устойчивость к ошибкам и поддержку языковых моделей.

На раннем этапе часть ролей может выполняться одним участником, но для масштабирования требуется распределение ответственности. Команда должна обладать компетенциями в Python-разработке, интерфейсах, OCR и машинном переводе, тестировании пользовательских сценариев, продуктовой аналитике и коммуникации с потенциальными клиентами.

В данной ВКР индивидуальная роль связана с фронтенд-разработкой. Зона ответственности включает проектирование пользовательского сценария, реализацию окон, overlay, статусов, настроек, горячих клавиш и визуальной обратной связи. Результат этой роли должен сделать техническую цепочку OCR и перевода понятной и удобной для пользователя.

2.9 Финансовая модель и необходимость привлечения инвестиций

2.9.1 Предпосылки финансовой модели

Финансовая модель проекта должна быть осторожной, потому что продукт находится на стадии рабочего прототипа и требует проверки спроса. Для первых лет нецелесообразно закладывать чрезмерно высокие показатели. Основной целью является не мгновенный массовый рост, а подтверждение готовности пользователей регулярно использовать продукт и платить за расширенные функции.

В качестве источников выручки рассматриваются Pro-подписки, доработки и настройки под клиентов, а также B2B-лицензии. На первый год закладывается выручка около 220 000 рублей. Она может состоять из 60 000 рублей от Pro-подписок, 60 000 рублей от доработок и настройки и 100 000 рублей от первых B2B-лицензий или пилотов.

На второй год плановая выручка может вырасти до 450 000 рублей: 140 000 рублей от Pro-подписок, 90 000 рублей от доработок и 220 000 рублей от B2B. На третий год целевой показатель составляет около 750 000 рублей: 260 000 рублей от подписок, 110 000 рублей от доработок и 380 000 рублей от B2B-лицензий.

Такая модель выглядит реалистично для учебного стартап-проекта, потому что она не предполагает огромной пользовательской базы с первого года. Рост происходит постепенно: сначала проверяется MVP и собирается обратная связь, затем формируются первые платные пользователи, затем появляются корпоративные сценарии.

2.9.2 Расходы проекта

					ПГУ 1.090301.17.001 ПЗ	Лист
						65
Изм.	Лист	№ докум.	Подпись	Дат		

Основные расходы проекта связаны с разработкой, тестированием, поддержкой, OCR/MT-интеграциями, маркетингом и юридической подготовкой. На старте значительная часть работы выполняется командой проекта, поэтому прямые денежные расходы могут быть относительно небольшими. Однако для коммерческого запуска требуется время на доработку, проверку стабильности и подготовку продукта к распространению.

Расходы на OCR и машинный перевод могут возникать при переходе от демонстрационной версии к коммерческой. В текущей реализации используется EasyOCR и GoogleTranslator через библиотеку deep_translator, но для стабильного коммерческого продукта может потребоваться надежный платный переводческий провайдер, резервный OCR-движок или гибридная архитектура.

Маркетинговые расходы включают подготовку демонстрационных видео, лендинга, публикаций, тестовых рекламных кампаний и материалов для B2B-пилотов. Юридические расходы могут включать регистрацию программного обеспечения, подготовку пользовательского соглашения, политики обработки данных и договоров с организациями.

2.9.3 Потребность в финансировании

Минимальная потребность в финансировании на стартовом этапе оценивается в 140 000 рублей. Эта сумма может быть направлена на завершение разработки, тестирование, упаковку продукта, юридическую подготовку и стартовый маркетинг. Основная часть средств должна быть направлена на доработку продукта, часть - на тестирование и исследование пользовательского опыта, часть - на продвижение и юридическое оформление.

Привлечение финансирования может осуществляться в форме гранта, партнерского финансирования, предоплаты от B2B-клиента или участия в программе поддержки студенческих стартапов. Для проекта важно не столько крупное финансирование, сколько возможность довести MVP до состояния, пригодного для beta-тестирования и первых пилотов.

2.9.4 Точка безубыточности и инвестиционная привлекательность

					ПГУ 1.090301.17.001 ПЗ	Лист
						66
Изм.	Лист	№ докум.	Подпись	Дат		

Поскольку проект является цифровым продуктом, он не требует аренды производственных площадей, закупки оборудования или хранения складских запасов. Основные затраты связаны с трудом разработчиков, тестированием, поддержкой и внешними сервисами. Это снижает точку безубыточности и делает проект устойчивым при небольшом количестве платных пользователей.

Инвестиционная привлекательность проекта заключается в сочетании растущего рынка, понятной пользовательской боли, уже реализованного рабочего прототипа и низких стартовых расходов. При успешной проверке гипотез продукт может масштабироваться за счет подписочной модели и B2B-лицензий.

Для фронтенд-части приоритетными расходами являются доработка overlay, улучшение читаемости, тестирование интерфейса на разных сценариях, исправление UX-ошибок, подготовка установочного пакета и пользовательской инструкции. Качество фронтенда влияет на конверсию, удержание и готовность пользователя платить за Pro-функции.

2.10 Производственный и организационный план

2.10.1 Общие сведения о производстве цифрового продукта

Производство в данном проекте означает разработку, тестирование, упаковку и сопровождение программного продукта. Так как продукт цифровой, не требуется производственная площадка, сырье, склад или физическая логистика. Основным ресурсом является труд разработчиков, время на тестирование и программная инфраструктура.

Для фронтенд-части производственный процесс связан с проектированием пользовательского сценария, реализацией окон и overlay, настройкой горячих клавиш, проверкой удобства интерфейса, тестированием читаемости результата и устранением визуальных ошибок.

2.10.2 Технологическая цепочка разработки

Технологическая цепочка начинается с анализа пользовательских сценариев. Нужно определить, как пользователь выбирает область, какие действия выполняет

					ПГУ 1.090301.17.001 ПЗ	Лист
						67
Изм.	Лист	№ докум.	Подпись	Дат		

чаще всего, когда ему нужен overlay, когда достаточно отдельного окна и какие настройки не должны мешать базовому сценарию.

Следующий этап - разработка интерфейса. На текущем этапе используется Tkinter: главное окно, окно настроек, окно выбора области, окно перевода, рамка выбранной области и overlay. Для каждого окна определяются состояния, обработчики событий, реакции на горячие клавиши и правила отображения.

Отдельный этап - разработка overlay. Здесь требуется совместить координаты OCR-фрагментов с визуальным слоем перевода, подобрать шрифт и цвет, стереть исходный текст в пределах OCR-блоков, разместить перевод и не допустить наложений. Для этого фронтенд должен учитывать ширину строк, высоту блоков, фон, контраст и границы выбранной области.

Затем выполняется тестирование UX. Проверяются сценарии выбора области, отмены выбора, слишком маленькой области, остановки перевода, перезапуска, переключения режимов, копирования результата, изменения прозрачности и поведения overlay при скриншоте.

Последний этап - упаковка и подготовка к распространению. Для фронтенда важно, чтобы пользователь мог запустить приложение без настройки среды разработки, быстро понять интерфейс и получить результат без чтения сложной инструкции.

2.10.3 Кадровое обеспечение и организация работ

На начальном этапе фронтенд-часть может выполняться одним разработчиком. Он отвечает за интерфейс, пользовательский сценарий, визуальные состояния, горячие клавиши, настройки и тестирование. При расширении проекта может потребоваться UX/UI-специалист, тестировщик и разработчик desktop-приложений.

Организация работ должна строиться итерационно. Сначала реализуется базовый сценарий, затем проверяется на пользователях, после этого дорабатываются спорные элементы интерфейса. Такой подход особенно важен для overlay, где реальные проблемы видны только при тестировании на разных приложениях и фонах.

					ПГУ 1.090301.17.001 ПЗ	Лист
						68
Изм.	Лист	№ докум.	Подпись	Дат		

2.10.4 Материалы, оборудование, безопасность и экологичность

Потребность в физических материалах отсутствует. Используемые инструменты включают компьютер разработчика, операционную систему Windows, Python, Tkinter, mss, PIL, EasyOCR, deep_translator, NumPy, PyTorch, OpenCV и вспомогательные библиотеки. Основные расходы связаны не с материалами, а со временем разработки и тестирования.

Экологическая нагрузка минимальна, поскольку продукт цифровой. Основной вопрос безопасности связан не с физическим производством, а с информационной приватностью. Для фронтенда важно явно показывать выбранную область, не создавать у пользователя ощущения скрытого захвата экрана и давать понятное управление запуском и остановкой перевода.

2.11 Описание развития бизнес-проекта

Развитие проекта должно происходить поэтапно. На начальном этапе главная цель - довести рабочий прототип до стабильного MVP. Для этого необходимо устранить выявленные ошибки, протестировать продукт на разных типах экранов, проверить работу с несколькими языками, улучшить обработку сложных изображений и подготовить понятную инструкцию для пользователя.

Первый этап - ближайшее время. На этом этапе выполняется исправление ошибок, улучшение стабильности, настройка поведения горячих клавиш, подготовка инструкции и демонстрационных материалов. Ожидаемый результат - MVP, который можно показывать пользователям и использовать в типовых сценариях: PDF, видеокурс, интерфейс программы, игра, скриншот, техническая документация.

Второй этап - beta-тестирование. Целесообразно привлечь 30-50 пользователей из основных целевых групп: студенты, IT-специалисты, пользователи зарубежного ПО, геймеры и сотрудники организаций. В ходе beta-теста нужно измерять точность OCR, задержку перевода, удобство выбора области, понятность статусов, частоту повторного использования и готовность платить за расширенные функции.

					ПГУ 1.090301.17.001 ПЗ	Лист
						69
Изм.	Лист	№ докум.	Подпись	Дат		

Третий этап - запуск первых платных функций. После beta-теста можно сформировать Free/Pro-модель. Бесплатная версия позволяет проверить базовый сценарий, а платная открывает регулярное использование без лимитов, overlay, историю, горячие клавиши, расширенные языки и дополнительные настройки. В этот же период целесообразно начать первые B2B-пилоты.

Четвертый этап - развитие на горизонте одного-трех лет. На этом этапе продукт расширяет языки, улучшает OCR, добавляет пресеты областей, историю переводов, экспорт, корпоративные настройки, управление пользователями и поддержку разных провайдеров OCR/MT. При успешном росте возможно создание отдельной версии для организаций.

Ключевые метрики развития должны оставаться измеримыми. Для продукта важны количество успешно переведенных экранных фрагментов, средняя задержка от выбора области до вывода результата, доля повторных пользователей, количество активных Pro-подписок, число B2B-пилотов, количество обращений в поддержку и конверсия из Free в Pro.

Проект способен масштабироваться, потому что его сценарий не привязан к одному типу контента. Один и тот же продукт можно использовать в учебе, работе, играх, документации и корпоративных системах. Модульная архитектура позволяет развивать клиентскую часть и вычислительное ядро независимо друг от друга.

Для фронтенд-направления ключевые задачи развития - повышение читаемости overlay, добавление пресетов областей, улучшение окна настроек, упрощение первого запуска, сохранение пользовательских параметров и более понятная обработка ошибок. Чем меньше действий выполняет пользователь, тем выше ценность продукта.

2.12 Описание рисков

Любой стартап-проект связан с рисками. Для Screen Translate наиболее значимыми являются технологические, пользовательские, рыночные,

					ПГУ 1.090301.17.001 ПЗ	Лист
						70
Изм.	Лист	№ докум.	Подпись	Дат		

юридические, финансовые и кадровые риски. Важно не только перечислить их, но и заранее определить меры снижения.

Первый риск - недостаточное качество OCR на сложных экранах. Нестандартные шрифты, низкое качество изображения, динамический фон, прозрачные интерфейсы, игры и видео могут ухудшать распознавание. Этот риск имеет среднюю вероятность и высокое влияние, потому что без распознавания текста продукт не сможет выполнить основную функцию. Меры снижения включают предобработку изображений, тестовый набор типовых экранов, поддержку нескольких OCR-подходов и улучшение фильтрации распознанных фрагментов.

Второй риск - задержка перевода. Если результат появляется слишком поздно, пользователь вернется к ручному переводчику или перестанет пользоваться продуктом. Для снижения риска используются обработка только выбранной области, ограничение размера изображения, кэширование, управление частотой OCR, оптимизация количества потоков и повторное использование результатов, если текст не изменился.

Третий риск - опасения пользователей по поводу приватности. Приложение работает с экраном, поэтому пользователь может бояться, что анализируется весь рабочий стол или чувствительные данные. Меры снижения: обработка только выбранной области, понятная визуальная рамка, прозрачные настройки, локальная обработка там, где это возможно, и отсутствие хранения лишних изображений.

Четвертый риск - стоимость внешних API и снижение маржинальности. Если при масштабировании потребуется платный OCR или переводчик, расходы могут расти вместе с количеством пользователей. Меры снижения включают лимиты бесплатного тарифа, оптимизацию запросов, кэширование, пакетную обработку и использование локальных моделей для части сценариев.

Пятый риск - конкуренция со стороны крупных сервисов. Крупные переводчики могут добавить более удобные функции экранного перевода. Ответ проекта должен быть в узком фокусе на Windows-сценарии, скорости, выбранной области, overlay, горячих клавишах, приватности и B2B-пилотах.

					ПГУ 1.090301.17.001 ПЗ	Лист
						71
Изм.	Лист	№ докум.	Подпись	Дат		

Шестой риск - недостаточный спрос на платную версию. Пользователи могут использовать только бесплатный функционал или бесплатные аналоги. Для снижения риска платные функции должны иметь очевидную ценность: стабильный overlay, отсутствие лимитов, история, горячие клавиши, экспорт, приоритетная обработка и корпоративные настройки.

Седьмой риск - кадровая зависимость. На старте значительная часть знаний сосредоточена у разработчика. Для снижения риска нужно документировать код, хранить проект в репозитории, описывать архитектуру, формировать тест-кейсы и постепенно разделять роли внутри команды.

Восьмой риск - юридические и лицензионные ограничения библиотек и сервисов. При коммерческом развитии необходимо проверить условия использования OCR-библиотек, переводческих сервисов, моделей и зависимостей. Для снижения риска требуется выбрать правомерную модель распространения, подготовить пользовательское соглашение и описать обработку данных.

В целом риски проекта являются управляемыми. Главные зоны контроля - качество OCR, скорость, приватность, стоимость обработки, конкурентная стратегия и поддерживаемость кода. По каждому направлению предусмотрены меры снижения, а дальнейшее тестирование позволит уточнить вероятность и влияние рисков.

Для фронтенд-разработки отдельным риском является плохая читаемость overlay. Перевод может быть слишком мелким, перекрывать важные элементы или плохо смотреться на сложном фоне. Меры снижения включают настройку прозрачности, подбор цветов, адаптивный размер шрифта, возможность скрыть рамку, отдельный режим окна и тестирование на разных типах экранов.

Еще один фронтенд-риск - перегрузка интерфейса настройками. Если пользователь не понимает, что нажимать, продукт теряет преимущество простоты. Для снижения риска необходимо оставлять основной экран минимальным, а расширенные параметры переносить в отдельный раздел настроек.

2.13 Заключение по стартап-проекту

					ПГУ 1.090301.17.001 ПЗ	Лист
						72
Изм.	Лист	№ докум.	Подпись	Дат		

Стартап-проект Screen Translate представляет собой программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени. Его основная ценность заключается в решении конкретной пользовательской боли: необходимости быстро понимать иностранный текст, который виден на экране, но не копируется обычными средствами.

В рамках темы «Фронтенд-разработка» ключевым результатом является проработка и реализация пользовательского сценария. Пользователь выбирает область, запускает распознавание, получает перевод в отдельном окне или overlay и продолжает работу в исходном контексте. Именно фронтенд делает цепочку OCR и машинного перевода понятной, быстрой и удобной.

Анализ проблемы показывает, что потребность в screen translation существует у разных групп пользователей: студентов, IT-специалистов, сотрудников компаний, исследователей, геймеров и обычных пользователей. Интервью подтверждают, что людям важны скорость, overlay, горячие клавиши и приватность выбранной области.

Текущая реализация программы уже включает важные элементы продукта: выбор области экрана, OCR через EasyOCR, перевод через GoogleTranslator, два режима вывода, горячие клавиши, настройки интерфейса, overlay, кэширование, стабилизацию результата и очистку памяти. Это позволяет рассматривать проект не только как учебную разработку, но и как основу для MVP.

Бизнес-модель проекта основана на freemium-подходе, Про-подписках, командных тарифах и B2B-лицензиях. Финансовая модель выбрана осторожной и предполагает постепенный рост выручки: 220 000 рублей в первый год, 450 000 рублей во второй год и 750 000 рублей в третий год. Минимальная потребность в финансировании составляет 140 000 рублей и направлена на доработку, тестирование, упаковку и стартовое продвижение продукта.

Главные риски проекта связаны с качеством OCR, задержкой перевода, читаемостью overlay, приватностью, конкуренцией и стоимостью внешних сервисов. Для каждого риска предложены меры снижения. При дальнейшей

					ПГУ 1.090301.17.001 ПЗ	Лист
						73
Изм.	Лист	№ докум.	Подпись	Дат		

доработке продукт может выйти на beta-тестирование, подтвердить спрос и развиваться как инструмент для частных пользователей и организаций.

Таким образом, проект является практически значимым стартапом с понятной проблемой, рабочим прототипом, рыночной логикой и перспективами развития. Фронтенд-разработка в данном проекте играет ключевую роль, потому что именно она превращает техническую возможность OCR и перевода в удобный пользовательский продукт.

					ПГУ 1.090301.17.001 ПЗ	Лист
						74
Изм.	Лист	№ докум.	Подпись	Дат		

3 Охрана труда

Разработка и эксплуатация ПО выполняется с использованием персональной электронно-вычислительной машины. Поэтому в выпускной квалификационной работе необходимо рассмотреть вопросы охраны труда и безопасной организации рабочего места пользователя.

Работа с приложением Screen Translator связана с длительным наблюдением за экраном, использованием клавиатуры и мыши, а также возможной работой с несколькими окнами одновременно. Такие условия требуют соблюдения правил эргономики, освещения и режима труда.

В разделе рассматриваются основные опасные и вредные факторы, требования к рабочему месту, режим труда и отдыха, а также вопросы электробезопасности и пожарной безопасности.

					ПГУ 1.090301.17.001 ПЗ	Лист
						75
Изм.	Лист	№ докум.	Подпись	Дат		

3.1 Анализ опасных и вредных факторов при работе с ПЭВМ

При работе с персональным компьютером на пользователя воздействуют факторы зрительного напряжения, статической нагрузки, монотонности, неудобной рабочей позы и психоэмоционального напряжения.

Зрительное напряжение возникает из-за постоянного наблюдения за экраном. В случае работы с переводчиком экранного текста пользователь может долго рассматривать мелкие символы, overlay-надписи и элементы интерфейса.

Статическая нагрузка связана с длительным положением тела за рабочим столом. При неправильной высоте стула или монитора повышается нагрузка на шею, спину и плечевой пояс.

Монотонность возникает при повторяющихся действиях: выборе области, запуске перевода, чтении результата и переключении между окнами. Для снижения утомления необходимо делать регулярные перерывы.

Психоэмоциональное напряжение может возникать при работе с большим количеством информации или при необходимости быстро понять иностранный текст. Удобный интерфейс частично снижает такую нагрузку, так как уменьшает количество ручных действий.

3.2 Требования к рабочему месту

Рабочее место пользователя должно быть организовано таким образом, чтобы экран находился на удобном расстоянии, а верхняя граница монитора располагалась примерно на уровне глаз.

Клавиатура и мышь должны находиться на одной поверхности и позволять удерживать кисти без чрезмерного изгиба. Это особенно важно при использовании горячих клавиш и частом управлении программой.

Стул должен обеспечивать поддержку спины, а высота сиденья должна позволять пользователю ставить стопы на пол или на подставку. Неправильная посадка быстро приводит к утомлению.

На рабочем столе следует оставить достаточно места для свободного перемещения мыши. Так как в программе используется выделение области экрана, точность движения мыши влияет на удобство работы.

Окружающие предметы не должны закрывать экран или создавать блики. Для программ экранного перевода это особенно важно, так как пользователь должен хорошо различать исходный текст и перевод.

					ПГУ 1.090301.17.001 ПЗ	Лист
						77
Изм.	Лист	№ докум.	Подпись	Дат		

3.3 Освещение, микроклимат и режим труда

Освещение рабочего места должно быть равномерным и не создавать бликов на экране. Слишком яркий источник света за спиной пользователя ухудшает видимость текста и повышает нагрузку на зрение.

При работе в темное время суток не рекомендуется использовать монитор как единственный источник света. Лучше применять мягкое фоновое освещение, которое снижает контраст между экраном и окружающим пространством.

Температура и влажность воздуха должны соответствовать нормальным условиям работы в помещении. Перегрев или сухой воздух усиливают утомляемость и снижают концентрацию.

При длительной работе за компьютером рекомендуется делать короткие перерывы. Во время перерыва следует отвести взгляд от экрана, выполнить простые упражнения для глаз и сменить положение тела.

В интерфейсе разработанного приложения предусмотрены темная и светлая темы. Это позволяет снизить зрительный дискомфорт в разных условиях освещения.

3.4 Электробезопасность и пожарная безопасность

Персональный компьютер и периферийные устройства должны подключаться к исправной электрической сети. Не допускается использование поврежденных кабелей, перегруженных удлинителей и нестабильных розеток.

Перед началом работы следует убедиться, что провода не находятся в местах, где их можно случайно задеть ногой или повредить мебелью. Кабели должны быть уложены так, чтобы не создавать препятствий для пользователя.

Пожарная безопасность обеспечивается исправностью оборудования, отсутствием перегрева и соблюдением правил эксплуатации электроприборов. Не следует закрывать вентиляционные отверстия системного блока или ноутбука.

В рабочем помещении должен быть свободный доступ к средствам пожаротушения и путям эвакуации. Пользователь должен знать порядок действий при обнаружении запаха гари или неисправности оборудования.

Соблюдение перечисленных требований обеспечивает безопасную работу как при разработке программы, так и при ее дальнейшей эксплуатации.

Заключение

В ходе выполнения выпускной квалификационной работы была разработана frontend-часть программного обеспечения для захвата, распознавания и перевода текстового контента в реальном времени. Работа была направлена на решение практической пользовательской задачи: быстро получить перевод текста, который виден на экране, но не может быть удобно выделен и скопирован.

В первой части работы была рассмотрена предметная область экранного перевода, выполнен анализ аналогов и сформулирована постановка задачи. Функциональные требования были описаны с позиции пользователя: выбрать область, запустить перевод, получить результат, настроить внешний вид и управлять приложением горячими клавишами. Нефункциональные требования были сформулированы как проверяемые технические условия, связанные с устойчивостью, потоками, ограничением размера изображения, кэшированием и исключением overlay из захвата.

На этапе проектирования была сформирована архитектура программы. Backend в данном проекте рассматривается как локальный внутренний слой, который выполняет захват изображения, предобработку, OCR, перевод и служебную обработку результата. Frontend отвечает за пользовательские окна, события, статусы, настройки и визуальный вывод результата.

В процессе разработки реализованы основные классы приложения: ScreenTranslatorApp, RegionSelector, RegionBorderOverlay, TranslationOverlay и TranslationOutputWindow. Каждый класс выполняет отдельную роль, поэтому структура программы остается последовательной и понятной. В основной части приведены примеры реализации, а в приложении размещен сокращенный листинг ключевых частей программы.

Проверка ПО показала, что пользователь может выполнить основной сценарий: выбрать область экрана, запустить обработку, получить перевод в overlay или отдельном окне, остановить выполнение и очистить результат. Последовательность взаимодействия компонентов завершается передачей

					ПГУ 1.090301.17.001 ПЗ	Лист
						80
Изм.	Лист	№ докум.	Подпись	Дат		

результата пользователю, поскольку видимый перевод является итогом работы программы.

Дальнейшее развитие программы может быть связано с упаковкой в установочный файл, сохранением профилей областей, историей переводов, расширенными настройками overlay и более современной визуальной оболочкой. При этом базовая структура приложения уже позволяет развивать продукт без изменения основной идеи пользовательского сценария.

					ПГУ 1.090301.17.001 ПЗ	Лист
						81
Изм.	Лист	№ докум.	Подпись	Дат		

Список использованных источников

1. ГОСТ 7.32-2017. Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления.
2. ГОСТ 2.105-2019. Единая система конструкторской документации. Общие требования к текстовым документам.
3. Python Software Foundation. Документация Python. [Электронный ресурс] / Режим доступа: <https://docs.python.org/3> свободный.
4. Python Software Foundation. Tkinter — Python interface to Tcl/Tk. [Электронный ресурс] / Режим доступа: <https://docs.python.org/3/library/tkinter.html> свободный.
5. MSS. Python MSS documentation. [Электронный ресурс] / Режим доступа: <https://python-mss.readthedocs.io/> свободный.
6. Pillow. Pillow documentation. [Электронный ресурс] / Режим доступа: <https://pillow.readthedocs.io/> свободный.
7. Jaided AI. EasyOCR documentation. [Электронный ресурс] / Режим доступа: <https://www.jaided.ai/easyocr/documentation/> свободный.
8. deep-translator documentation. [Электронный ресурс] / Режим доступа: <https://deep-translator.readthedocs.io/> свободный.

Приложение А

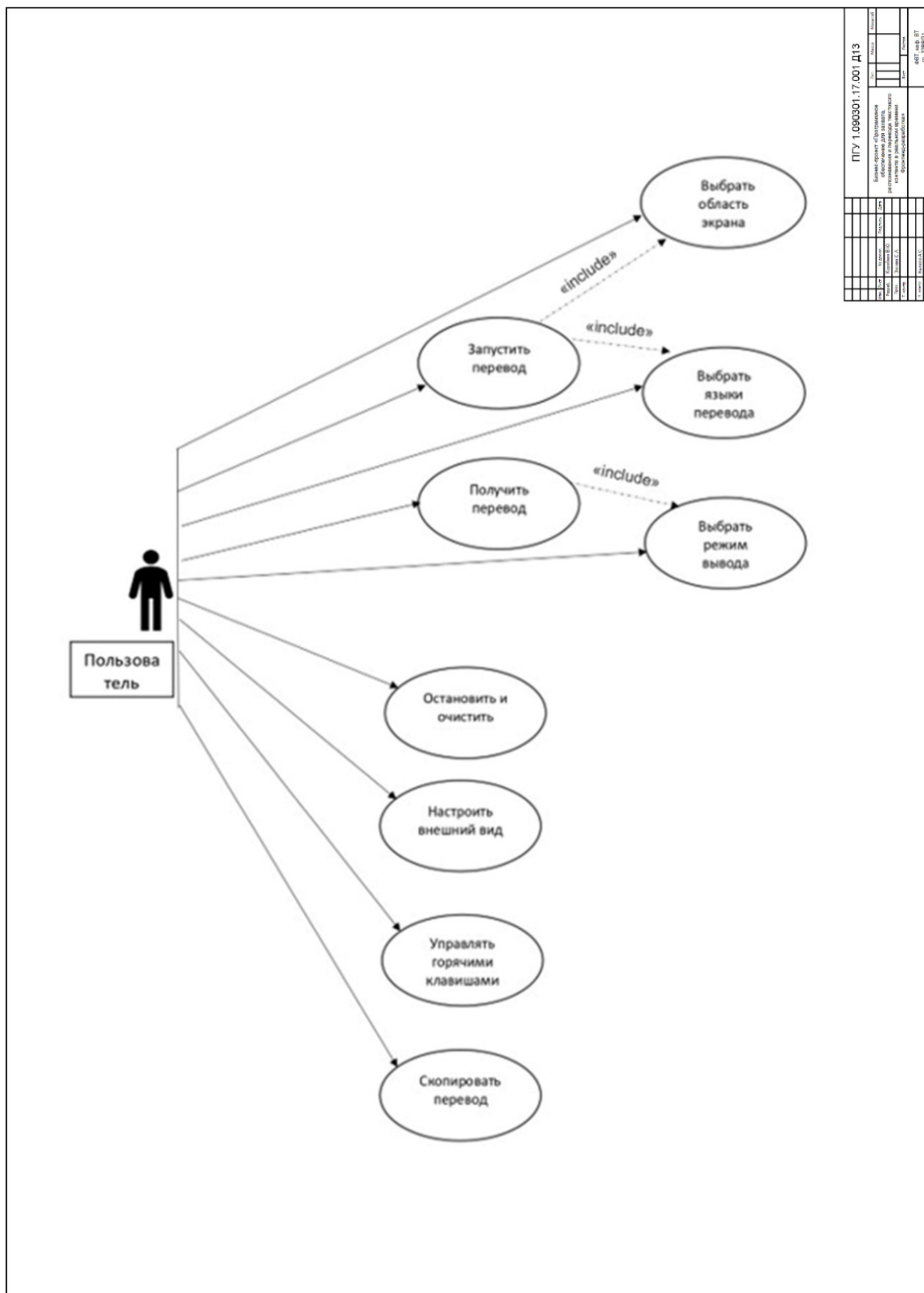
(Обязательное)

Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода
текстового контента в реальном времени. Фронтенд-разработка»

UML-диаграммы

					ПГУ 1.090301.17.001 ПЗ	Лист
						83
Изм.	Лист	№ докум.	Подпись	Дат		

Приложение А.1 – Диаграмма прецедентов



ПГУ 1.090301.17.001 Д13									
Вариант	Исход.	Исход.	Исход.	Исход.	Исход.	Исход.	Исход.	Исход.	Исход.
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10

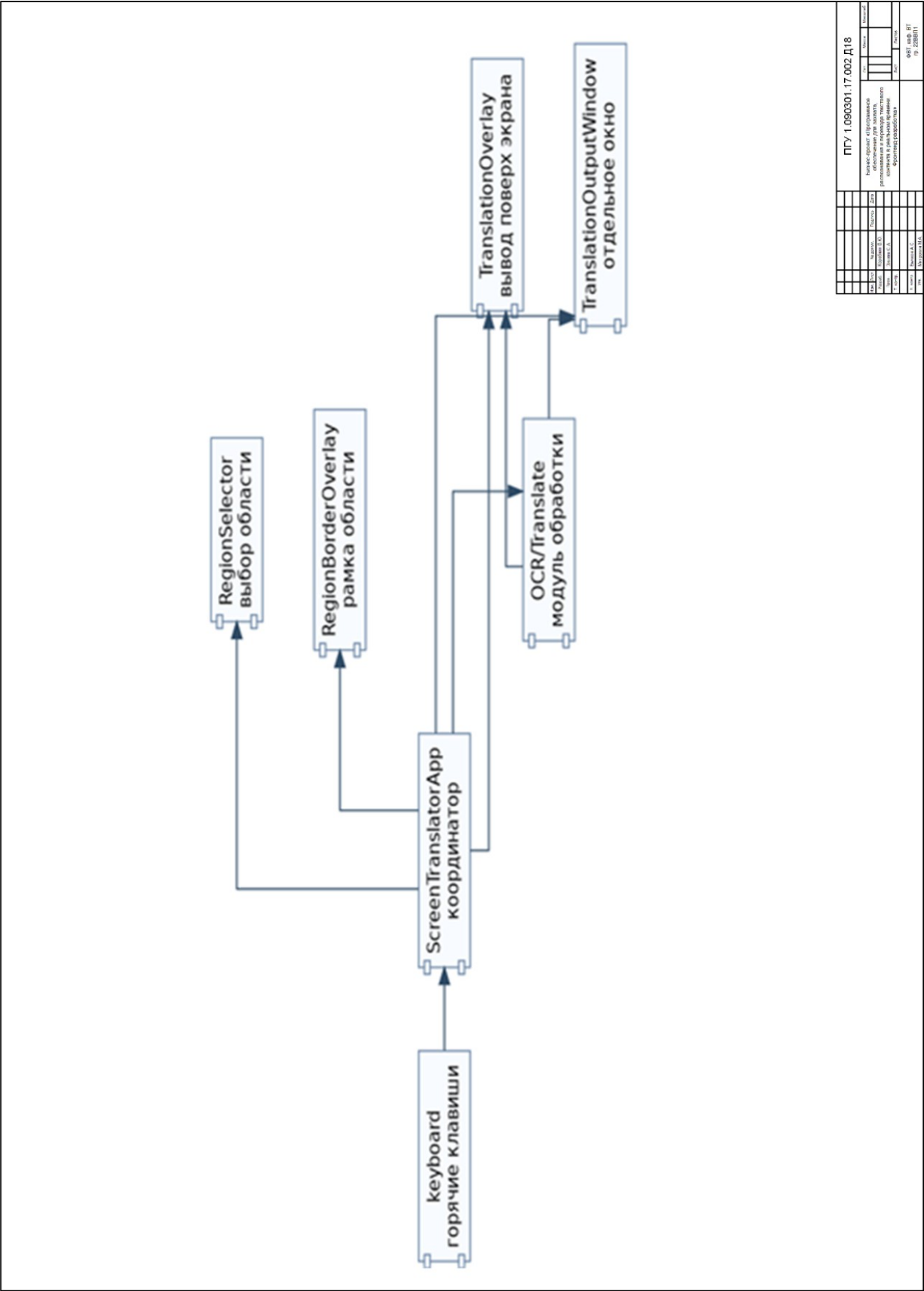
Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

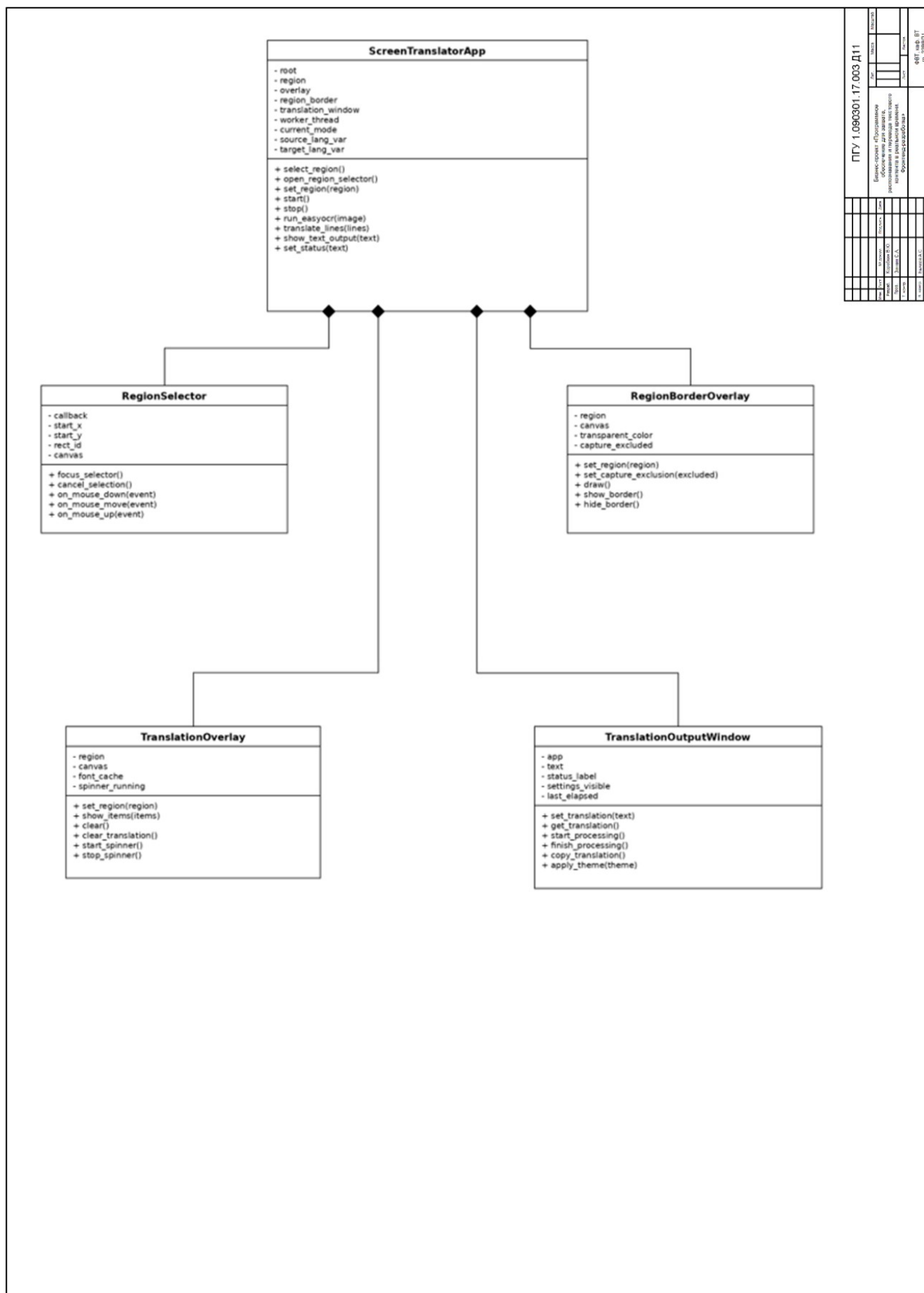
84

Приложение А.2 – Диаграмма компонентов frontend-части



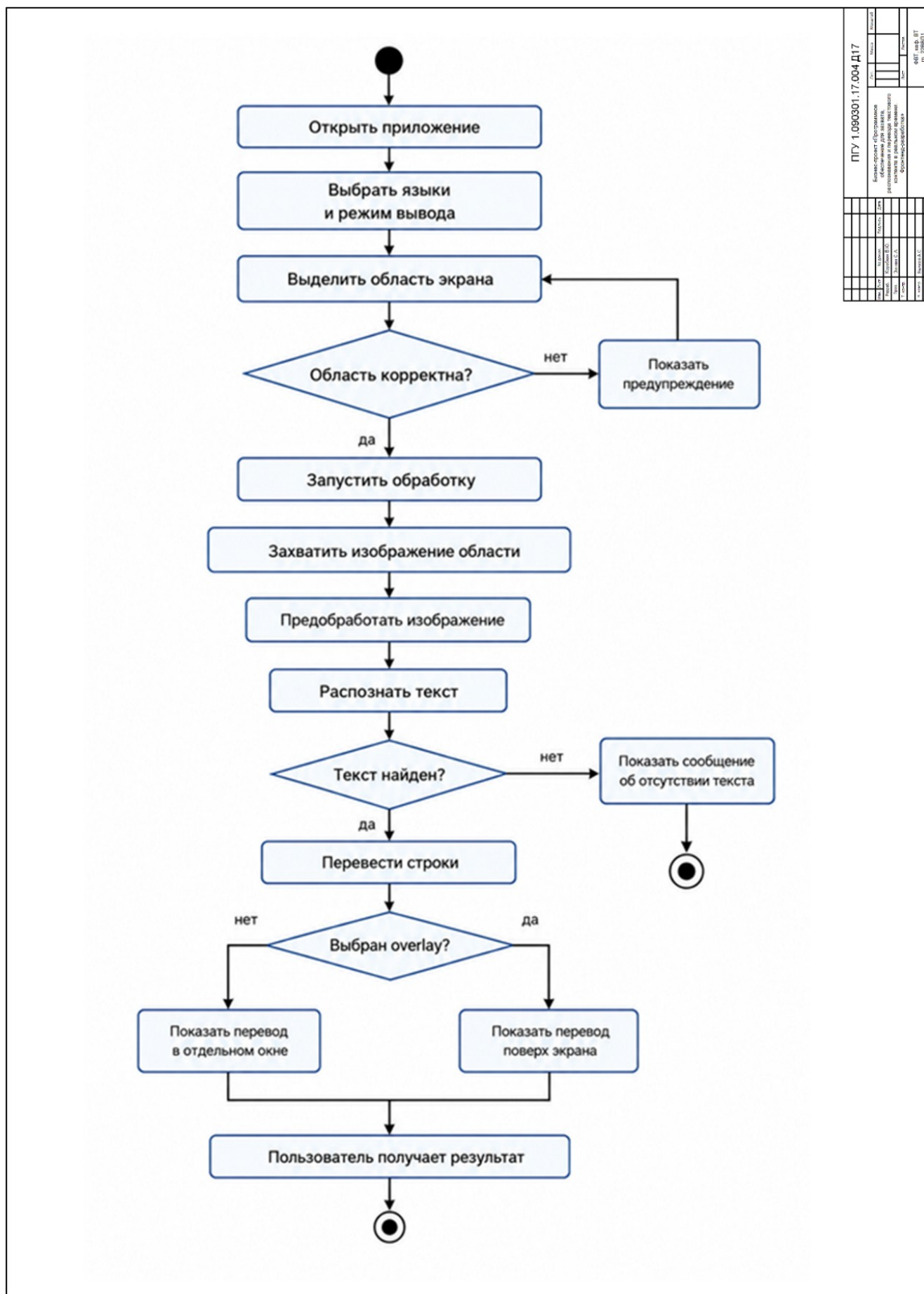
ПГУ 1.090301.17.002 Д18									
Универсальный программный комплекс для автоматизации работы в сфере технического обслуживания и ремонта оборудования									
Формат документа: PDF									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа: 17.002.001									
ИД документа:									

Приложение А.3 – Диаграмма классов

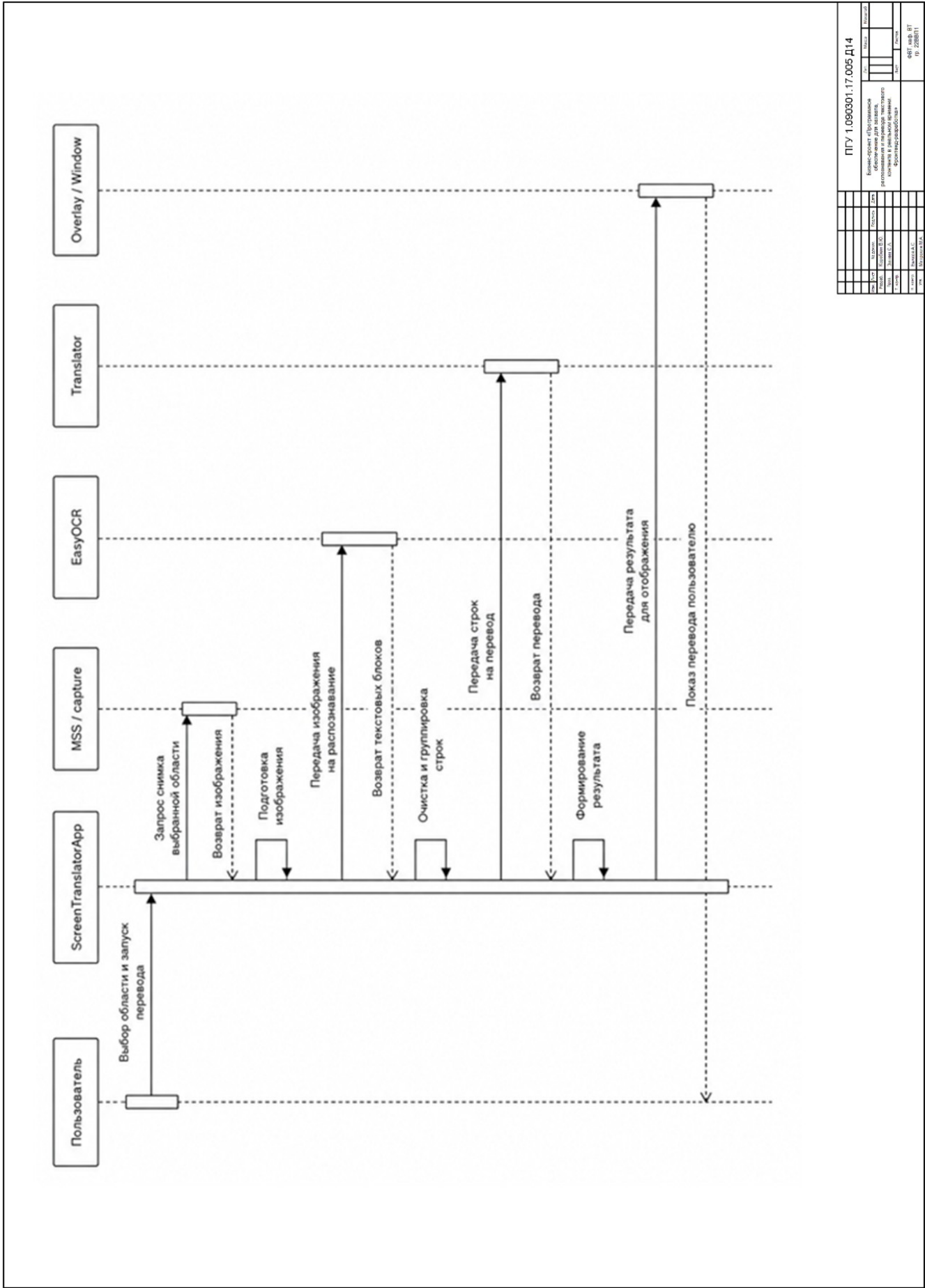


ПГУ 1.090301.17.003 Д11		Бизнес-проект «Программное обеспечение для перевода текста с русского языка на английский язык»	Функциональный модуль
Имя	Версия	Статус	Дата
1.0	1.0	В разработке	10.12.2021
2.0	2.0	Тестирование	15.12.2021
3.0	3.0	Готово к релизу	20.12.2021
4.0	4.0	В архиве	25.12.2021
5.0	5.0	Удалено	30.12.2021
6.0	6.0	Удалено	31.12.2021
7.0	7.0	Удалено	01.01.2022
8.0	8.0	Удалено	02.01.2022
9.0	9.0	Удалено	03.01.2022
10.0	10.0	Удалено	04.01.2022
11.0	11.0	Удалено	05.01.2022
12.0	12.0	Удалено	06.01.2022
13.0	13.0	Удалено	07.01.2022
14.0	14.0	Удалено	08.01.2022
15.0	15.0	Удалено	09.01.2022
16.0	16.0	Удалено	10.01.2022
17.0	17.0	Удалено	11.01.2022
18.0	18.0	Удалено	12.01.2022
19.0	19.0	Удалено	13.01.2022
20.0	20.0	Удалено	14.01.2022
21.0	21.0	Удалено	15.01.2022
22.0	22.0	Удалено	16.01.2022
23.0	23.0	Удалено	17.01.2022
24.0	24.0	Удалено	18.01.2022
25.0	25.0	Удалено	19.01.2022
26.0	26.0	Удалено	20.01.2022
27.0	27.0	Удалено	21.01.2022
28.0	28.0	Удалено	22.01.2022
29.0	29.0	Удалено	23.01.2022
30.0	30.0	Удалено	24.01.2022
31.0	31.0	Удалено	25.01.2022
32.0	32.0	Удалено	26.01.2022
33.0	33.0	Удалено	27.01.2022
34.0	34.0	Удалено	28.01.2022
35.0	35.0	Удалено	29.01.2022
36.0	36.0	Удалено	30.01.2022
37.0	37.0	Удалено	31.01.2022
38.0	38.0	Удалено	01.02.2022
39.0	39.0	Удалено	02.02.2022
40.0	40.0	Удалено	03.02.2022
41.0	41.0	Удалено	04.02.2022
42.0	42.0	Удалено	05.02.2022
43.0	43.0	Удалено	06.02.2022
44.0	44.0	Удалено	07.02.2022
45.0	45.0	Удалено	08.02.2022
46.0	46.0	Удалено	09.02.2022
47.0	47.0	Удалено	10.02.2022
48.0	48.0	Удалено	11.02.2022
49.0	49.0	Удалено	12.02.2022
50.0	50.0	Удалено	13.02.2022
51.0	51.0	Удалено	14.02.2022
52.0	52.0	Удалено	15.02.2022
53.0	53.0	Удалено	16.02.2022
54.0	54.0	Удалено	17.02.2022
55.0	55.0	Удалено	18.02.2022
56.0	56.0	Удалено	19.02.2022
57.0	57.0	Удалено	20.02.2022
58.0	58.0	Удалено	21.02.2022
59.0	59.0	Удалено	22.02.2022
60.0	60.0	Удалено	23.02.2022
61.0	61.0	Удалено	24.02.2022
62.0	62.0	Удалено	25.02.2022
63.0	63.0	Удалено	26.02.2022
64.0	64.0	Удалено	27.02.2022
65.0	65.0	Удалено	28.02.2022
66.0	66.0	Удалено	29.02.2022
67.0	67.0	Удалено	30.02.2022
68.0	68.0	Удалено	01.03.2022
69.0	69.0	Удалено	02.03.2022
70.0	70.0	Удалено	03.03.2022
71.0	71.0	Удалено	04.03.2022
72.0	72.0	Удалено	05.03.2022
73.0	73.0	Удалено	06.03.2022
74.0	74.0	Удалено	07.03.2022
75.0	75.0	Удалено	08.03.2022
76.0	76.0	Удалено	09.03.2022
77.0	77.0	Удалено	10.03.2022
78.0	78.0	Удалено	11.03.2022
79.0	79.0	Удалено	12.03.2022
80.0	80.0	Удалено	13.03.2022
81.0	81.0	Удалено	14.03.2022
82.0	82.0	Удалено	15.03.2022
83.0	83.0	Удалено	16.03.2022
84.0	84.0	Удалено	17.03.2022
85.0	85.0	Удалено	18.03.2022
86.0	86.0	Удалено	19.03.2022
87.0	87.0	Удалено	20.03.2022
88.0	88.0	Удалено	21.03.2022
89.0	89.0	Удалено	22.03.2022
90.0	90.0	Удалено	23.03.2022
91.0	91.0	Удалено	24.03.2022
92.0	92.0	Удалено	25.03.2022
93.0	93.0	Удалено	26.03.2022
94.0	94.0	Удалено	27.03.2022
95.0	95.0	Удалено	28.03.2022
96.0	96.0	Удалено	29.03.2022
97.0	97.0	Удалено	30.03.2022
98.0	98.0	Удалено	31.03.2022
99.0	99.0	Удалено	01.04.2022
100.0	100.0	Удалено	02.04.2022

Приложение А.4 – Диаграмма деятельности



Приложение А.5 – Диаграмма последовательностей



ПГУ 1.090301.17.005 Д14									
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель

Приложение Б

(Обязательное)

Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода
текстового контента в реальном времени. Фронтенд-разработка»

Сокращённый листинг программы

					ПГУ 1.090301.17.001 ПЗ	Лист
						89
Изм.	Лист	№ докум.	Подпись	Дат		

```

import tkinter as tk
from tkinter import ttk

class ScreenTranslatorApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Screen Translator")
        self.root.geometry("1020x700")
        self.root.minsize(760, 440)

        self.theme_var = tk.StringVar(value="light")

        self.text_widgets = []
        self.style = ttk.Style()

        self.build_interface()
        self.apply_theme()

    def build_interface(self):
        self.root.geometry("860x520")
        self.root.minsize(720, 430)

        shell = ttk.Frame(
            self.root,
            padding=(18, 16, 18, 14)
        )
        shell.pack(fill=tk.BOTH, expand=True)

        header = ttk.Frame(shell)
        header.grid(row=0, column=0, sticky="ew")

        title_group = ttk.Frame(header)
        title_group.grid(row=0, column=0, sticky="w")

        ttk.Label(
            title_group,
            text="Screen Translator",
            font=("Arial", 18, "bold")
        ).pack(anchor="w")

        ttk.Label(

```

```

        title_group,
        text="Быстрый OCR-перевод выбранной области экрана"
    ).pack(anchor="w")

theme_box = ttk.Frame(header)
theme_box.grid(row=0, column=1, sticky="e")

ttk.Label(
    theme_box,
    text="Тема"
).pack(side=tk.LEFT)

self.theme_combo = ttk.Combobox(
    theme_box,
    textvariable=self.theme_var,
    values=["light", "dark"],
    state="readonly"
)
self.theme_combo.pack(side=tk.LEFT, padx=5)

ttk.Button(
    theme_box,
    text="Настройки",
    command=self.open_settings_window
).pack(side=tk.LEFT)

action_bar = ttk.Frame(shell, padding=(16, 14))
action_bar.grid(row=1, column=0, sticky="ew", pady=10)

ttk.Button(
    action_bar,
    text="Выбрать область"
).pack(side=tk.LEFT, padx=5)

ttk.Button(
    action_bar,
    text="Запустить перевод"
).pack(side=tk.LEFT, padx=5)

ttk.Button(
    action_bar,
    text="Остановить"

```

```

        ).pack(side=tk.LEFT, padx=5)

def apply_theme(self):
    pass

def open_settings_window(self):
    pass

class RegionSelector(tk.Toplevel):
    def __init__(self, master, callback, cancel_callback=None):
        super().__init__(master)

        self.callback = callback
        self.cancel_callback = cancel_callback

        self.width = self.winfo_screenwidth()
        self.height = self.winfo_screenheight()
        self.left = 0
        self.top = 0

        self.geometry(
            f"{self.width}x{self.height}+{self.left}+{self.top}"
        )

        self.overridereDIRECT(True)
        self.attributes("-topmost", True)
        self.attributes("-alpha", 0.35)

        self.canvas = tk.Canvas(
            self,
            bg="gray",
            cursor="crosshair",
            highlightthickness=0
        )

        self.canvas.pack(fill=tk.BOTH, expand=True)

        self.canvas.create_text(
            self.width // 2,
            40,
            text="Выделите область экрана для перевода. ESC – отмена",

```

```

        fill="white",
        font=("Arial", 18, "bold")
    )

    self.canvas.bind("<ButtonPress-1>", self.on_mouse_down)
    self.canvas.bind("<B1-Motion>", self.on_mouse_move)
    self.canvas.bind("<ButtonRelease-1>", self.on_mouse_up)

    self.bind("<Escape>", self.cancel_selection)

def on_mouse_down(self, event):
    pass

def on_mouse_move(self, event):
    pass

def on_mouse_up(self, event):
    pass

def cancel_selection(self, event=None):
    self.destroy()

class RegionBorderOverlay(tk.Toplevel):
    def __init__(self, master):
        super().__init__(master)

        self.withdraw()

        self.transparent_color = "#ff00ff"

        self.configure(bg=self.transparent_color)
        self.overridedirect(True)
        self.attributes("-topmost", True)

        self.canvas = tk.Canvas(
            self,
            bg=self.transparent_color,
            highlightthickness=0,
            bd=0
        )

```

```

        self.canvas.pack(fill=tk.BOTH, expand=True)

def set_region(self, region):
    self.geometry(
        f"{region['width']}{region['height']}"
        f"{region['left']}{region['top']}"
    )

def show_border(self):
    self.deiconify()
    self.lift()

def hide_border(self):
    self.withdraw()

class TranslationOverlay(tk.Toplevel):
    def __init__(self, master):
        super().__init__(master)

        self.withdraw()

        self.transparent_color = "#ff00ff"

        self.configure(bg=self.transparent_color)
        self.overrideredirect(True)
        self.attributes("-topmost", True)

        self.canvas = tk.Canvas(
            self,
            bg=self.transparent_color,
            highlightthickness=0,
            bd=0
        )

        self.canvas.pack(fill=tk.BOTH, expand=True)

def set_region(self, region):
    self.geometry(
        f"{region['width']}{region['height']}"
        f"{region['left']}{region['top']}"
    )

```

```

def clear(self):
    self.canvas.delete("all")

def hide_and_clear(self):
    self.clear()
    self.withdraw()

class TranslationOutputWindow(tk.Toplevel):
    def __init__(self, app):
        super().__init__(app.root)

        self.title("Перевод")
        self.geometry("820x520+120+120")
        self.minsize(360, 220)

        self.overrideRedirect(True)
        self.attributes("-topmost", True)

        self.surface = tk.Frame(self)
        self.surface.pack(fill=tk.BOTH, expand=True)

        self.text = tk.Text(
            self.surface,
            wrap=tk.WORD,
            relief=tk.FLAT,
            bd=0,
            padx=14,
            pady=14
        )

        self.text.pack(fill=tk.BOTH, expand=True)

        self.progress = ttk.Progressbar(
            self.surface,
            mode="indeterminate",
            length=120
        )

        self.gear_button = tk.Button(
            self.surface,

```

```

        text="⚙️",
        command=self.toggle_settings_panel
    )

    self.gear_button.place(x=10, y=10)

    self.resize_grip = tk.Label(
        self.surface,
        text="📐",
        cursor="size_nw_se"
    )

    self.resize_grip.pack(side=tk.BOTTOM, anchor=tk.SE)

    def toggle_settings_panel(self):
        pass

if __name__ == "__main__":
    root = tk.Tk()
    app = ScreenTranslatorApp(root)
    root.mainloop()

```

Приложение В

(Обязательное)

Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода
текстового контента в реальном времени. Фронтенд-разработка»
Презентация

					ПГУ 1.090301.17.001 ПЗ	Лист
						97
Изм.	Лист	№ докум.	Подпись	Дат		

ПРОЕКТ

SCREEN
TRANSLATE

Бизнес-проект «Программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени.»

ПО захватывает текст с экрана, распознаёт его и показывает перевод в реальном времени. Фокус дипломной части – пользовательский интерфейс, overlay, сценарии взаимодействия и качество фронтенд-опыта.

«Видит текст. Понимает смысл. Переводит мгновенно.»

Реализовали:
Аргаткин А.А.
Коробкин В.Ю.

ПРОБЛЕМА

Языковой барьер любого видимого текста

Текст из видео, игр, интерфейсов, изображений и PDF часто нельзя скопировать напрямую. Пользователь тратит около 1-2 минут на скриншот, OCR, перевод и возврат в исходное окно; из-за переключения между окнами теряется контекст.

Учёба

видео, PDF, презентации
без ручного копирования

Работа

документация и зарубежное ПО
в контексте окна

Развлечения

игры, субтитры, скриншоты
перевод поверх экрана

Фронтенд-боль

много окон, нет overlay,
нет быстрого выбора
области

Вывод: нужен перевод любого видимого текста прямо на экране - без потери контекста и ручного копирования.

2

Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

98

ЦЕЛЬ

Разработать программное обеспечение для захвата, распознавания и перевода текстового контента в реальном времени

ОБЪЕКТ

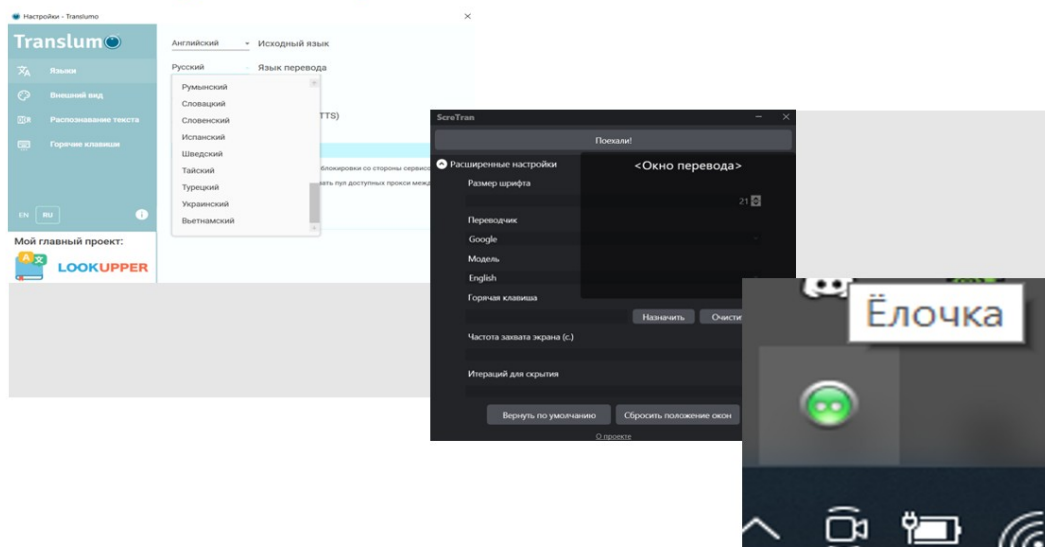
Программное обеспечение для захвата, распознавания и перевода текстового контента

ЗАДАЧИ

- изучить предметную область экранного перевода;
- выполнить анализ существующих аналогов;
- сформулировать требования к фронтенд-части и бекенд;
- определить границы ответственности разрабатываемого интерфейса;
- выбрать технологии и средства разработки;
- спроектировать структуру окон и пользовательских сценариев;
- реализовать основные элементы интерфейса;
- провести проверку работы интерфейса и пользовательских режимов.

3

АНАЛИЗ СУЩЕСТВУЮЩИХ АНАЛОГОВ



4

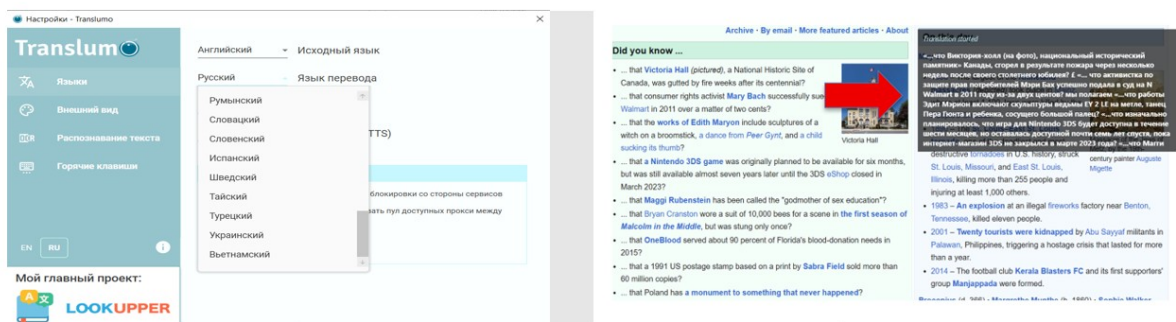
Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

99

АНАЛИЗ СУЩЕСТВУЮЩИХ АНАЛОГОВ



Translation started

DeepLTranslator failed: The remote server returned an error: (451) Unavailable For Legal Reasons.. Try to change translator, use a proxy or switch VPN location.

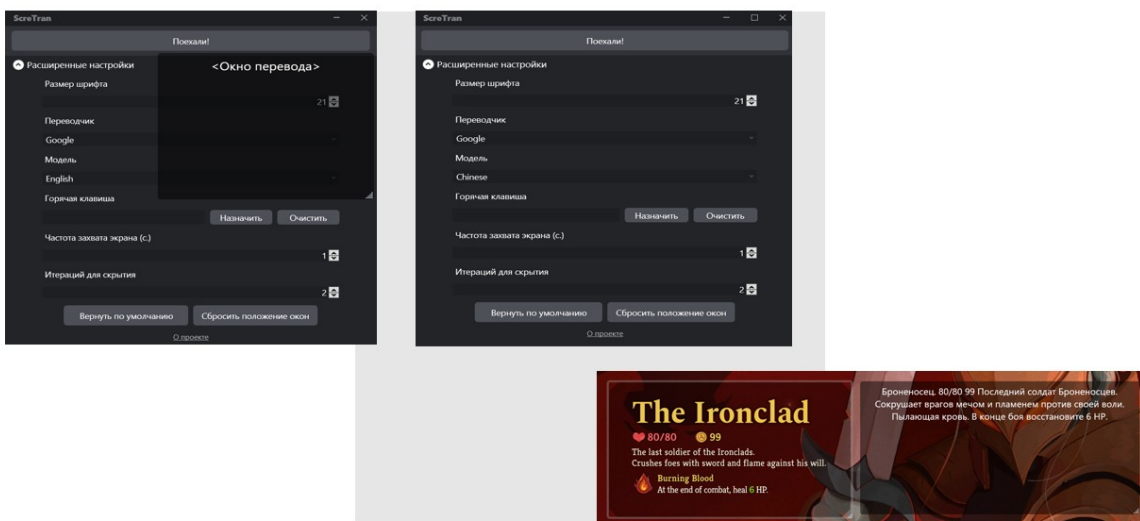
Appearance

Tab

Small

5

АНАЛИЗ СУЩЕСТВУЮЩИХ АНАЛОГОВ



6

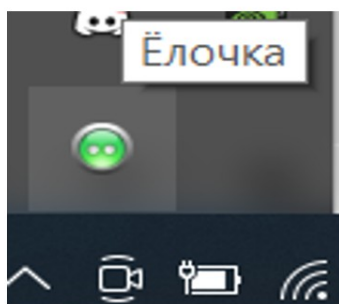
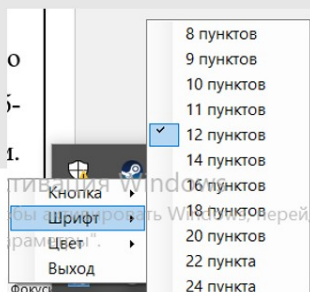
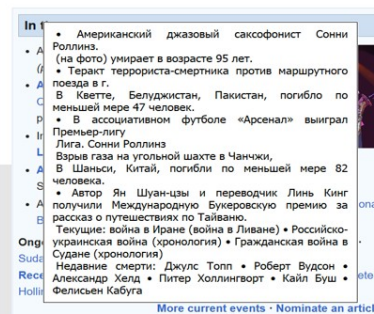
Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

100

АНАЛИЗ СУЩЕСТВУЮЩИХ АНАЛОГОВ



7

АНАЛИЗ СУЩЕСТВУЮЩИХ АНАЛОГОВ

Аналог	Основные возможности	Преимущества	Недостатки для данной задачи
Translumo	Перевод текста в оконном режиме	Скорость перевода, интерфейс на двух языках, поддержка разных моделей OCR	Программные ошибки в определенных кейсах, потребность в постоянном открытии/закрытии приложения, отсутствие <u>overlay</u>
ScreTran	Перевод текста в оконном режиме	Скорость перевода	Громоздкий интерфейс с тремя окнами, отсутствие <u>overlay</u> , малое количество настроек
«Ёлочка»	Одноразовый перевод текста при захвате экрана	Удобно для сайтов, пошаговых игр, скорость перевода	Малое количество настроек, перевод только с английского на русский, отсутствие <u>overlay</u>

SCREEN TRANSLATE: узкий фокус на Windows-сценарии, выбор области, быстрый overlay, окно перевода, горячие клавиши, удобный пользовательский режим.

8

Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

101

РЕШЕНИЕ

Пользовательский сценарий в интерфейсе

Сценарий: пользователь открывает иностранный интерфейс, видеоурок или PDF и не может выделить текст.

Целевой эффект: перевод фрагмента за 5-10 секунд вместо 1-2 минут ручных действий. Ключевая ценность - быстрый и понятный путь: выбрать область → увидеть перевод → продолжить работу.



9

ФРОНТЕНД-РАЗРАБОТКА

Зона ответственности дипломной части



Итог фронтенд-разработки: пользователь получает быстрый, понятный и ненавязчивый экранный перевод без перехода в сторонние окна.

10

Изм.	Лист	№ докум.	Подпись	Дат

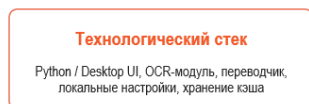
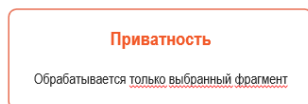
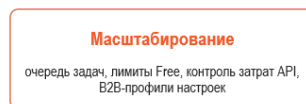
ПГУ 1.090301.17.001 ПЗ

Лист

102

БЭКЕНД-ЧАСТЬ

Зона ответственности дипломной части



Цель backend-слоя: скорость <20 сек., стабильность OCR на типовых экранах и предсказуемая стоимость одного перевода.

11

СТЕК ТЕХНОЛОГИЙ



Python

Язык программирования



Pillow

Подготовка изображения



VS Code

Среда разработки



Deep-translator

Машинный перевод



Tkinter

Графический интерфейс



EasyOCR

Распознавание текста

12

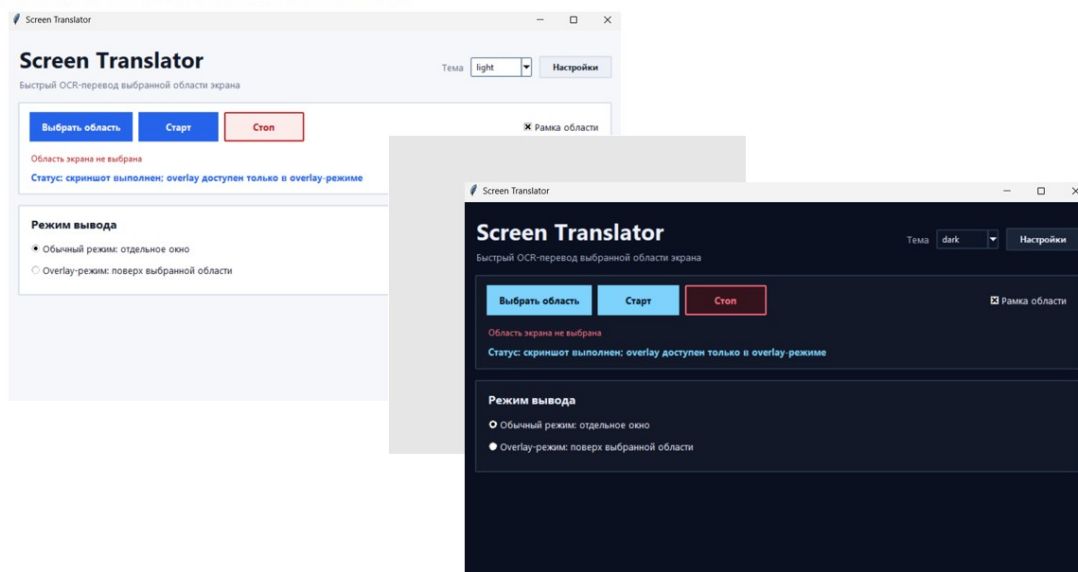
Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

103

ОПИСАНИЕ ПРОГРАММЫ



13

ОПИСАНИЕ ПРОГРАММЫ

Настройки

Применить

Языки

Оригинал: английский Перевод: русский

Окно перевода

Фон окна: #111827 Выбрать

Цвет текста: #f9f6f6 Выбрать

Прозрачность фона, %: 0

Скриншоты и стрим

Клавиша: print screen Записать

Горячие клавиши

Выбрать область: f6 Записать

Старт с нуля: f7 Записать

Стоп: f8 Записать

Рамка области: f9 Записать

Обычный режим: 1 Записать

Overlay-режим: 2 Записать

14

Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

104

ОПИСАНИЕ РАБОТЫ ПРОГРАММЫ

From today's featured article



13th-century illustration of Æthelred

Æthelred the Unready (c. 968 – 1016) was King of the English from 978 to 1013 and from 1014 until his death. He came to the throne as a boy after the killing of his half-brother King **Edward the Martyr**. During his reign, raids by the Vikings escalated to large-scale invasions, and the English paid increasingly large amounts of tribute. In December 1013, King **Swein Forkbeard** of Denmark conquered England. Æthelred fled to Normandy, but when Swein died in February 1014 he returned and drove out Swein's son **Cnut**, who returned the following year. English resistance was hampered by the treachery of Æthelred's chief advisor, **Eadric Streona**, and by distrust between Æthelred and his eldest son, **Edmund Ironside**. Æthelred died in April 1016 and Edmund became king. He put up stronger resistance than his father, but died in December, and Cnut then became king. Æthelred's reign was nevertheless a period of cultural achievements, and some historians see its first half as moderately successful. (*Full article...*)

Recently featured: [Ancestor incident](#) • [Oregon State Capitol](#) • [SMS Westfalen](#)
[Archive](#) • [By email](#) • [More featured articles](#) • [About](#)

Из сегодняшней избранной статьи
 Этельред Неготовый (ок. 968–1016) был королём Англии с 978 по 1013 г. и с 1014 г. до своей смерти. Он взошёл на престол мальчиком после убийство своего сводного брата короля Эдуарда Мученика: во время его правления совершались набеги викинги переросли в крупномасштабные вторжения, и англичане платили все больше и больше большие суммы дани. В декабре 1013 года король Дании Свейн Вилкобордовый завоевал Англию. Этельред бежал в Нормандию; но когда Свейн умер в 8 февраля 1014 г. он вернулся и изгнал сына Свейна Кнута; кто вернул Следующий год, 13 век английское сопротивление было затруднено предательством Этельреда. [Щелкасьон главного советника; Эдрика Стреоны и недоверия между Этельредом и его старшим сын Эдмунд Айронсайд. Этельред умер в апреле 1016 года, и Эдмунд стал королем. Он оказал более сильное сопротивление, чем его отец; но умер в декабре; и Кнут стал королем. Тем не менее правление Этельреда было периодом культурных достижений, и некоторые историки видят его первый период. (Полная статья...)

More current

Настройки

Копировать Стор

Шрифт Segoe UI

Размер 14

Фон #111827 Выбрать

Текст #99faff Выбрать

Прозрачность, % 0

Применить

Статус: перевод выполнен, строки 16

15

ОПИСАНИЕ РАБОТЫ ПРОГРАММЫ

Цвет фона

Основные цвета:

Дополнительные цвета:

Определить цвет >>

OK Отмена

Добавить в набор

Отонок: 0 Красный: 255

Контраст: 240 Зеленый: 0

Яркость: 120 Синий: 0

Цвет(Загрузка)

Статус: сайт перевод...

ОПИСАНИЕ РАБОТЫ

Из сегодняшней избранной статьи
 Телред Неготовый (ок. 968–1016) был королем Англии с 9 1013 г. и с [014 г. до своей смерти Он взошел на престол мальчиком после убийство своего сводного брата короля Эдуарда Мученика: во время его правления викинги переросли в крупномасштабные вторжения, и англичане платили все больше и больше большие суммы дани: В декабре [013. Король Св завоевал Англию. Этельред бежал в Нормандию. 101+ февраля он вернулся и изгнал сына Свейна Кнута; кто вернул Следующий год, 13 век английское сопротивление было затруднено предательством Этельреда. [Щелкасьон главного советника; Эдрика Стреона, и недоверием между Этельредом и его старшим сын Эдмунд [онсайд. Этельред умер в апреле 1016 года, и Эдмунд стал королем. Он оказал более сильное сопротивление, чем его отец; но умер в декабре; и Кнут стал королем. Тем не менее правление Этельреда было периодом великих культурных достижений, и некоторые историки видят его первый период. (Полная статья...)

Настройки

Копировать Стор

Шрифт Segoe UI

Размер 14

Фон #f10000 Выбрать

Текст #99faff Выбрать

Прозрачность, % 0

Применить

Статус: настройки применены

Недавно опубликованные: Инцидент в Анкастере * Капитолий штата Орегон. SMS

Длится * Пл. элективный плат. Еще избранные статьи

16

Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

105

ОПИСАНИЕ РАБОТЫ ПРОГРАММЫ

From today's featured article



13th-century illustration of Æthelred

Æthelred the Unready (c. 968 – 1016) was King of the English from 978 to 1013 and from 1014 until his death. He came to the throne as a boy after the killing of his half-brother King **Edward the Martyr**. During his reign, raids by the Vikings escalated to large-scale invasions, and the English paid increasingly large amounts of tribute. In December 1013, King **Swein Forkbeard** of Denmark conquered England. Æthelred fled to **Normandy**, but when Swein died in February 1014 he returned and drove out Swein's son **Cnut**, who returned the following year. English resistance was hampered by the treachery of Æthelred's chief advisor, **Eadric Streona**, and by distrust between Æthelred and his eldest son, **Edmund Ironside**. Æthelred died in April 1016 and Edmund became king. He put up stronger resistance than his father, but died in December, and Cnut then became king. Æthelred's reign was nevertheless a period of cultural achievements, and some historians see its first half as moderately successful. (*Full article...*)

Recently featured: [Auncaster incident](#) · [Oregon State Capitol](#) · [SMS Westfalen](#)
[Archive](#) · [By email](#) · [More featured articles](#) · [About](#)

Из сегодняшней избранной статьи



Этельред Неготовый (ок. 968–1016) был королём Англии с 978 по 1013 г. и с 1014 г. до своей смерти: Он вззошел на престол мальчиком после убийства своего сводного брата короля **Эдуарда Мученика**: во время его правления совершались набеги викинги переросли в крупномасштабные вторжения, и англичане платили все больше и больше.

большие суммы дани. В декабре 1013 года король Дании **Свейн Вилкобордовый** завоевал Англию. Этельред бежал в Нормандию, но когда Свейн умер в в феврале 1014 г. он вернулся и изгнал сына Свейна **Кнута**; кто вернул

Следующий год, 13 век: английское сопротивление было затруднено предательством Этельреда.

[Зеленая главная советника; Эдрика Стреоны и недоверия между Этельредом и его старшим

сын **Эдмунд Айронсайд**. Этельред умер в апреле 1016 года, и **Эдмунд** стал королем:

Он оказал более сильное сопротивление, чем его отец: но умер в декабре; и **Кнут** тогда стал королем:

Тем не менее правление Этельреда было периодом культурных достижений, и некоторые историки видят его первые

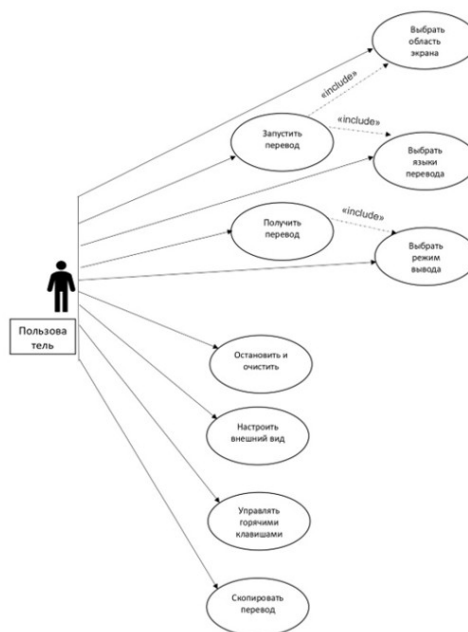
вплывовину менее успешные. (*Полная статья*-)

Недавно опубликованное: [Инцидент в Анкастере в Капитолии штата Орегон](#). [СМС Вестфалия](#)

[Архив](#) · [По электронной почте](#) · [Другие избранные статьи](#) · [О проекте](#)

17

ДИАГРАММА ПРЕЦЕДЕНТОВ



18

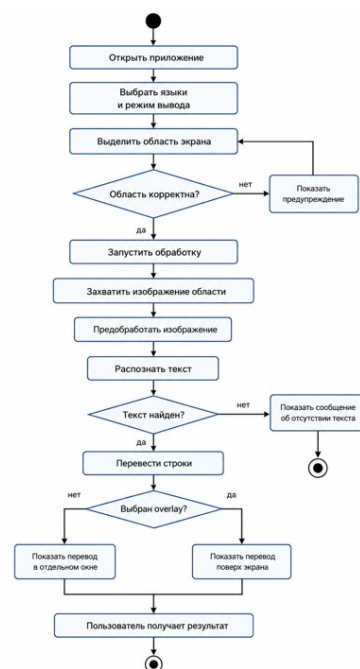
Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

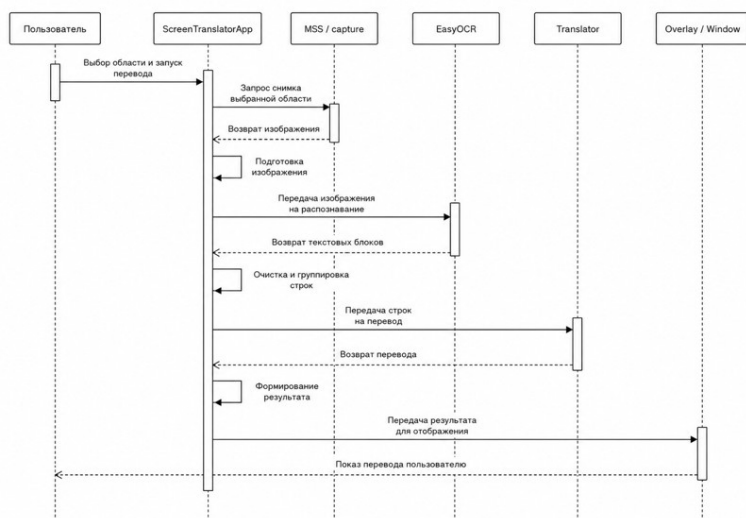
106

ДИАГРАММА ДЕЯТЕЛЬНОСТИ



19

ДИАГРАММА ПОСЛЕДОВАТЕЛЬНОСТИ



20

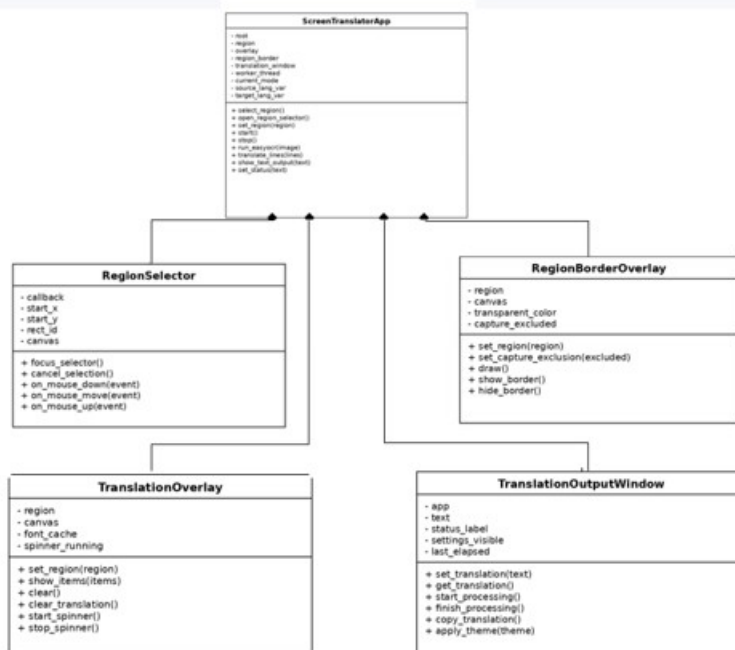
Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

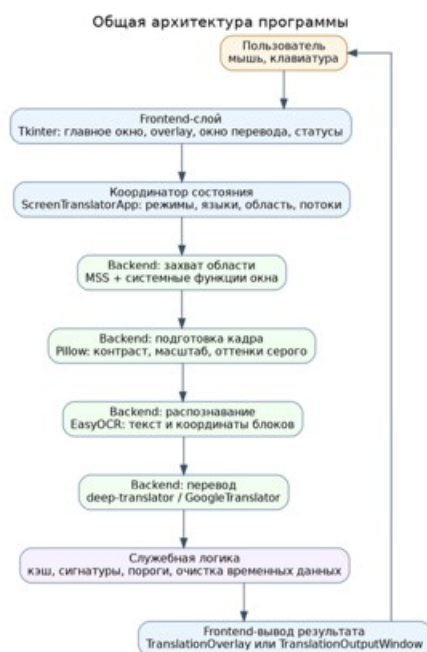
107

ДИАГРАММА КЛАССОВ



21

АРХИТЕКТУРА ПРОГРАММЫ



22

Изм.	Лист	№ докум.	Подпись	Дат

ПГУ 1.090301.17.001 ПЗ

Лист

108

ФИНАНСОВЫЕ ПОКАЗАТЕЛИ

Подписка + лицензии

Показатель	1 год	2 год	3 год
Pro-подписки	60 000 Р	140 000 Р	260 000 Р
Доработки/настройка	60 000 Р	90 000 Р	110 000 Р
B2B-лицензии	100 000 Р	220 000 Р	380 000 Р
Итого выручка	220 000 Р	450 000 Р	750 000 Р

23

ЗАКЛЮЧЕНИЕ

- На данный момент: создана рабочая версия приложения. Реализованы выбор области экрана, окно результата и overlay-отображение перевода. Интерфейс понятен и готов к дальнейшему тестированию.
- Планы на ближайшее время: доработать внешний вид overlay, добавить пресеты областей, провести тестирование на 50-100 пользователей.
- Дальнейшее развитие (год 1-3): выпустить стабильную версию для Windows, добавить историю переводов, расширить настройки интерфейса и подготовить продукт к продаже по подписке и B2B-лицензиям.

24

					ПГУ 1.090301.17.001 ПЗ	Лист
						109
Изм.	Лист	№ докум.	Подпись	Дат		

БЛАГОДАРИМ ЗА ВНИМАНИЕ

					ПГУ 1.090301.17.001 ПЗ	Лист
						110
Изм.	Лист	№ докум.	Подпись	Дат		