

[hang] 0 6pt [] name=

1 ,

numberless [block] 0pt []

[hang] 1.0 6pt

[hang] 1.0.0 6pt

12.5mm 3.5ex plus 1ex minus .2ex 2.3ex plus .2ex

12.5mm 18pt 12pt name=

2 ,

numberless 0pt 3.5ex plus 1ex minus .2ex 2.3ex plus .2ex

12.5mm 18pt 12pt

[itemize] label=—, labelindent=leftmargin=*, labelsep=1em, align=parleft,
topsep=0pt, partopsep=0pt, itemsep=0pt, parsep=0pt,

Реферат

Пояснительная записка содержит 1 страницу, 1 рисунок, 1 таблицу, 1 источник, 1 приложение.

WEB-ПРИЛОЖЕНИЕ, FASTAPI, POSTGRESQL, VUE.JS, DOCKER

Объектом дипломного проекта является web-приложение для перевода текста ограниченного объёма с иностранного языка с применением сервиса «g4f».

Основной целью дипломного проекта является разработка веб-приложения для перевода текста ограниченного объёма с иностранного языка с применением сервиса «g4f». В разработке дипломного проекта была использована платформа FastAPI, язык программирования Python, технология Vue.js, протокол обмена данными HTTP.

Пояснительная записка состоит из введения, семи разделов и заключения.

В первом разделе описана литература, использовавшаяся при разработке дипломного проекта.

Во втором разделе описаны цель и задачи дипломного проекта, обзор аналогов и обзор средств разработки.

В третьем разделе представлены архитектура web-приложения, проектирование и структура таблиц базы данных.

Четвёртый раздел посвящен разработке web-приложения.

Пятый раздел посвящен тестированию web-приложения.

В шестом разделе приведено руководство программиста.

В седьмом разделе приводится расчет экономических параметров и себестоимость программного продукта.

В заключении представлены итоги дипломного проекта и задачи, которые были решены в ходе разработки программного средства.

Abstract

СЮДА ПЕРЕВОД

Содержание

1	,	1
2	,	1
Введение		5
3	Обзор литературных источников	6
3.1	Документация Python	6
3.2	Документация FastAPI	6
3.3	Документация Pydantic	6
3.4	Документация PostgreSQL	6
4	Постановка задачи и обзор аналогичных решений	7
4.1	Постановка задачи	7
4.2	Обзор аналогичных решений	7
4.3	Выводы	8
5	Проектирование web-приложения	9
5.1	Функциональность web-приложения	9
5.2	Структура базы данных	11
5.3	Архитектура web-приложения	16
5.4	Выводы	18
6	Реализация web-приложения	19
6.1	Обоснование выбора программной платформы	19
6.2	Реализация серверной части web-приложения	19
6.2.1	Изменение учётной записи	19
6.2.2	Просмотр открытых сессий	19
6.2.3	Завершение открытых сессий	19
6.2.4	Изменение списка исходных статей	19
6.2.5	Изменение списка переведённых статей	19
6.2.6	Изменение списка жалоб на переводы своих статей	19
6.2.7	Просмотр своих уведомлений	19
6.2.8	Изменение списка комментариев к жалобам на переводы своих статей	19
6.2.9	Изменение списка настроек переводчика	19
6.2.10	Регистрация	19
6.2.11	Аутентификация	19
6.2.12	Изменение списка открытых жалоб	19
6.2.13	Создание комментариев для жалоб	19
6.2.14	Просмотр статистики жалоб	19
6.2.15	Изменение списка стилей перевода	19
6.2.16	Изменение списка моделей перевода	20

6.2.17	Изменение списка пользователей	20
6.3	Реализация базы данных	20
7	Технико-экономическое обоснование проекта	21
7.1	Общая характеристика разрабатываемого программного средства .	21
7.2	Методика обоснования цены	21
7.2.1	Стоимость разработки	21
7.2.2	Цена продажи	22

Введение

Тема проекта «Web-приложение «GPTranslate» для перевода текста ограниченного объёма с иностранного языка с применением сервиса «g4f» означает, что результатом выполнения проекта является web-приложение, позволяющее пользователям переводить текст на исходном языке в текст на другом языке (например, с польского языка на немецкий) с использованием внешнего сервиса «g4f», предоставляющего доступ к нейронным сетям. Доступ к внешнему сервису осуществляется по его API. Ограниченный объём текста означает, что в web-приложении установлено ограничение на максимальную длину исходного текста.

Основная цель проекта – повысить эффективность и увеличить скорость перевода текста за счёт использования внешних сервисов, а также предоставить пользователям механизм обратной связи для улучшения сервиса.

Для достижения поставленной цели в рамках дипломного проекта были сформулированы следующие задачи: в разделе 1 провести анализ существующих сервисов перевода и выявить их преимущества и недостатки; в разделе 2 разработать архитектуру и структуру приложения, включая выбор технологий и структуру базы данных; в разделе 3 описать программную разработку с реализацией ключевых функций; в разделе 4 провести тестирование функционала и производительности; в разделе 5 подготовить техническую документацию и инструкции по развёртыванию приложения.

Целевая аудитория приложения включает широкий спектр пользователей: от профессиональных переводчиков и сотрудников международных компаний до владельцев web-сайтов и блогеров, нуждающихся в качественном и быстром переводе своих материалов.

Для реализации web-приложения было решено использовать язык программирования Python и фреймворк FastAPI из-за высокой гибкости, распространённости и простоты данных технологий.

3 Обзор литературных источников

3.1 Документация Python

Официальная документация Python представляет собой комплексную систему материалов, организованную по принципу ”от простого к сложному”. Она выделяется среди других языков своей полнотой и доступностью изложения. Она включает в себя следующие ключевые разделы: учебник для начинающих (пошаговое введение в синтаксис и концепции языка, построенное по принципу ”сценарной документации что позволяет новичкам быстро добиваться нужных результатов по заранее расписанным шагам), справочник по библиотеке (исчерпывающее описание стандартной библиотеки, включающее подробные примеры использования каждого модуля), справочник по языку (детальное описание синтаксиса и семантики языка, включая формальные определения и спецификации), а также руководство по расширению и внедрению Python (информация для разработчиков, создающих модули на языках программирования C/C++ или встраивающих интерпретатор Python в другие приложения).

Документация Python решает следующие проблемы: обеспечение единого понимания языка среди разработчиков, стандартизация использования встроенных возможностей и библиотек, снижение порога вхождения для новых разработчиков, выступление в качестве авторитетного источника при разрешении технических споров и неоднозначностей.

3.2 Документация FastAPI

Документация фреймворка FastAPI предоставляет полную информацию для использования данной технологии. Она содержит руководство пользователя для быстрого начала работы, пошаговые инструкции к различным аспектам фреймворка с пояснениями каждого шага и примерами для различных версий смежных технологий (например, с использованием `typing.List` в Python 3.8 и `list` в Python 3.9 и позднее), а также описание взаимодействия с прочими технологиями (например, Starlette и Pydantic).

Кроме того, документация предоставляет доступ к описанию типов, классов и функций с их подробным описанием, что позволяет ускорить разработку приложений. Также документация предоставляет рекомендации по развёртыванию приложений в продуктовых окружениях и настройке в случае нахождения за прокси.

3.3 Документация Pydantic

Документация библиотеки валидации Pydantic организована как интерактивное руководство, сочетающее теоретические объяснения с практическими примерами. Она выделяется среди других библиотек своим подходом ”проблема - решение где каждый раздел начинается с описания конкретной задачи разработки, за чем следует описание её решения.

Так же, как документация FastAPI, документация Pydantic содержит руководство для начинающих для быстрого начала работы, руководства по основным аспектам библиотеки, а также раздел с описанием функций и классов, реализуемых библиотекой. Документация позволяет узнать, каким образом проектировать и реализовывать надёжные API с сериализацией и валидацией.

3.4 Документация PostgreSQL

Документация СУБД PostgreSQL представляет исчерпывающее руководство по ключевым аспектам применения данной технологии, а также инструкции по развёртыванию и использованию. Кроме того, документация содержит полное описание диалекта PostgreSQL с примерами использования, что позволяет лучше изучить его для полноценного использования СУБД.

Кроме того, документация описывает внутренне устройство СУБД, включая индексы, системы обеспечения согласованности, доступности, репликации, партиционирования и шардирования, и заметки к выпуску каждой новой версии, включая мелкие и крупные изменения.

Документация PostgreSQL предоставляет исчерпывающую информацию о возможностях системы, облегчает процесс настройки и оптимизации производительности, помогает в проектировании схемы данных и разработке сложных запросов, а также служит справочником при решении проблем и отладке ошибок.

4 Постановка задачи и обзор аналогичных решений

4.1 Постановка задачи

Web-приложение должно позволять пользователю зарегистрироваться по адресу имени, электронной почты и паролю, подтвердить адрес электронной почты по ссылке из электронного письма, аутентифицироваться по адресу электронной почты и паролю, создать конфигурацию переводчика, загрузить исходную статью с введённым с клавиатуры текстом или при помощи загруженного текстового файла, выполнить перевод, используя созданную ранее конфигурацию переводчика, получить уведомление о завершении перевода, просмотреть переведённую статью. В случае, если перевод не удовлетворяет пользователя, web-приложение должно позволять пользователю создать жалобу на перевод, а модератору – просмотреть жалобу и удовлетворить либо отклонить её.

4.2 Обзор аналогичных решений

Одним из самых популярных сервисов по переводу текста с одного языка на другой является DeepL. Он предоставляет возможность перевода текста между различными языками, распознавание голоса, загрузку файлов и пересказ текста. Внешний вид страницы сервиса представлен на рисунке 4.1.

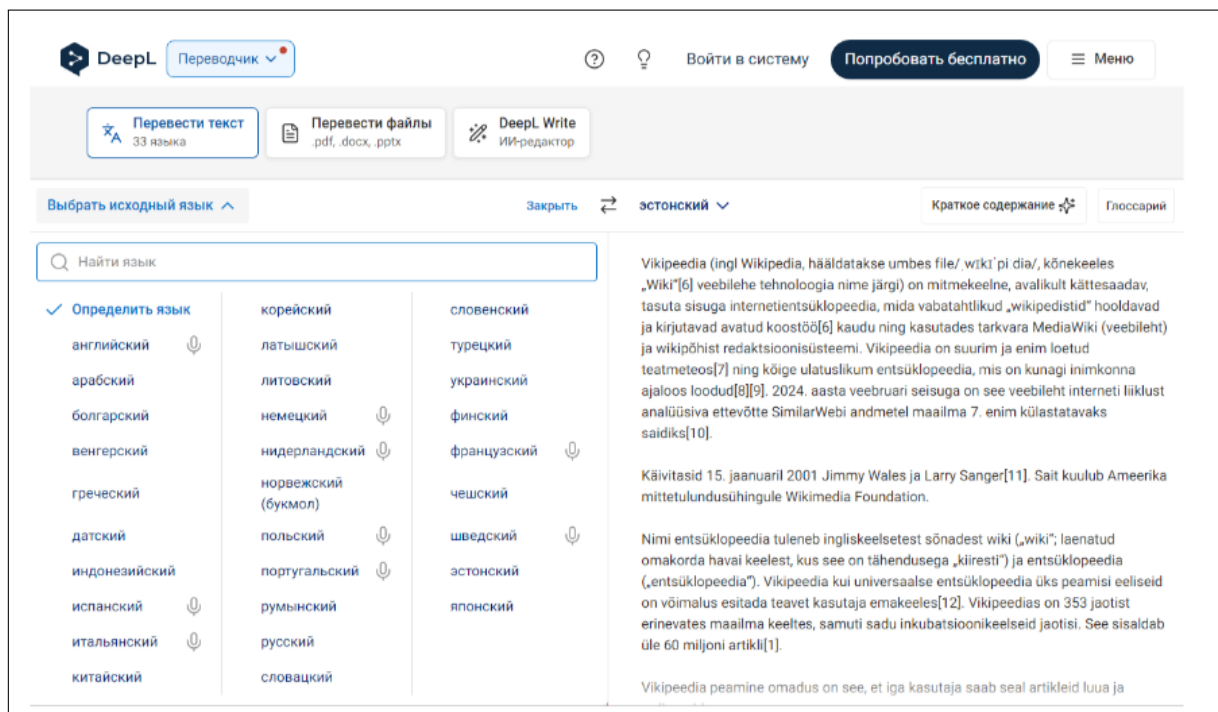


Рисунок 4.1 – Страница сервиса DeepL

Данный сервис применяет нейронные сети для перевода текста, что положительно сказывается на качестве перевода текста.

В качестве второго аналогичного решения был рассмотрен сервис Google Translate. Внешний вид страницы данного сервиса представлен на странице 4.2.

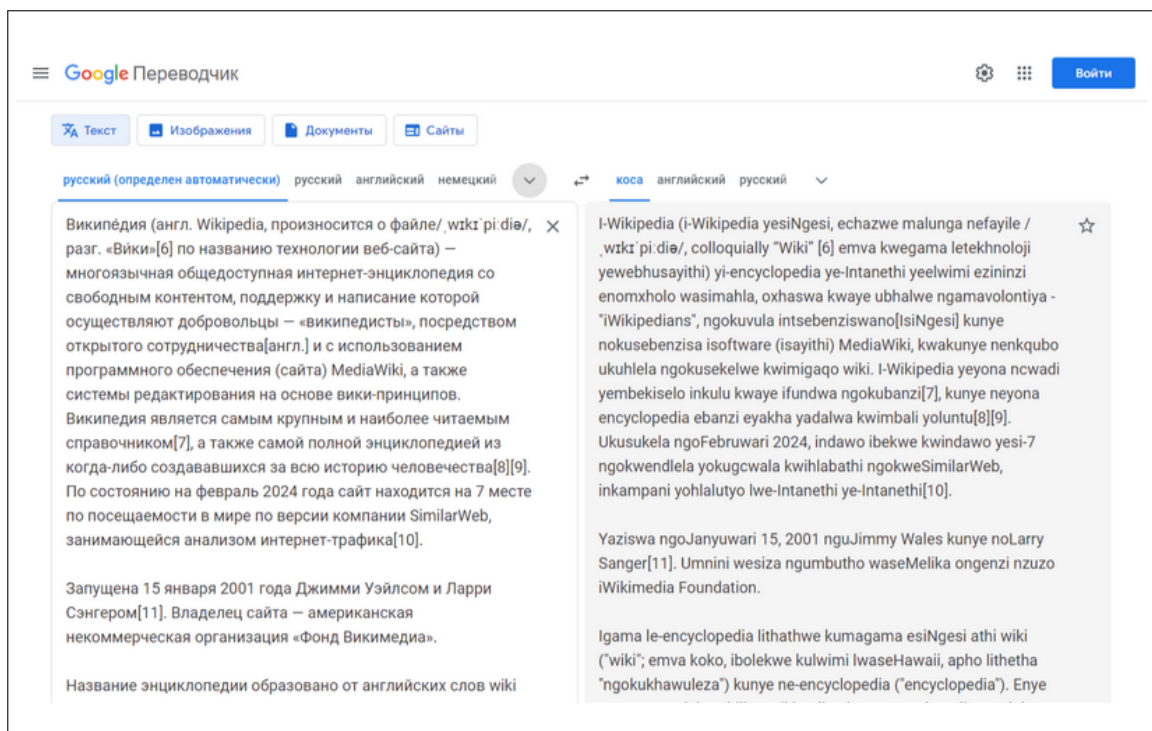


Рисунок 4.2 – Страница сервиса Google Translate

Это один из самых известных и широко используемых сервисов машинного перевода. Он также предоставляет возможность перевода текстовых файлов, а также более широкий выбор языков по сравнению с DeepL.

В качестве третьего аналогичного решения был рассмотрен сервис Wordvice. Внешний вид его страницы представлен на рисунке 4.3.

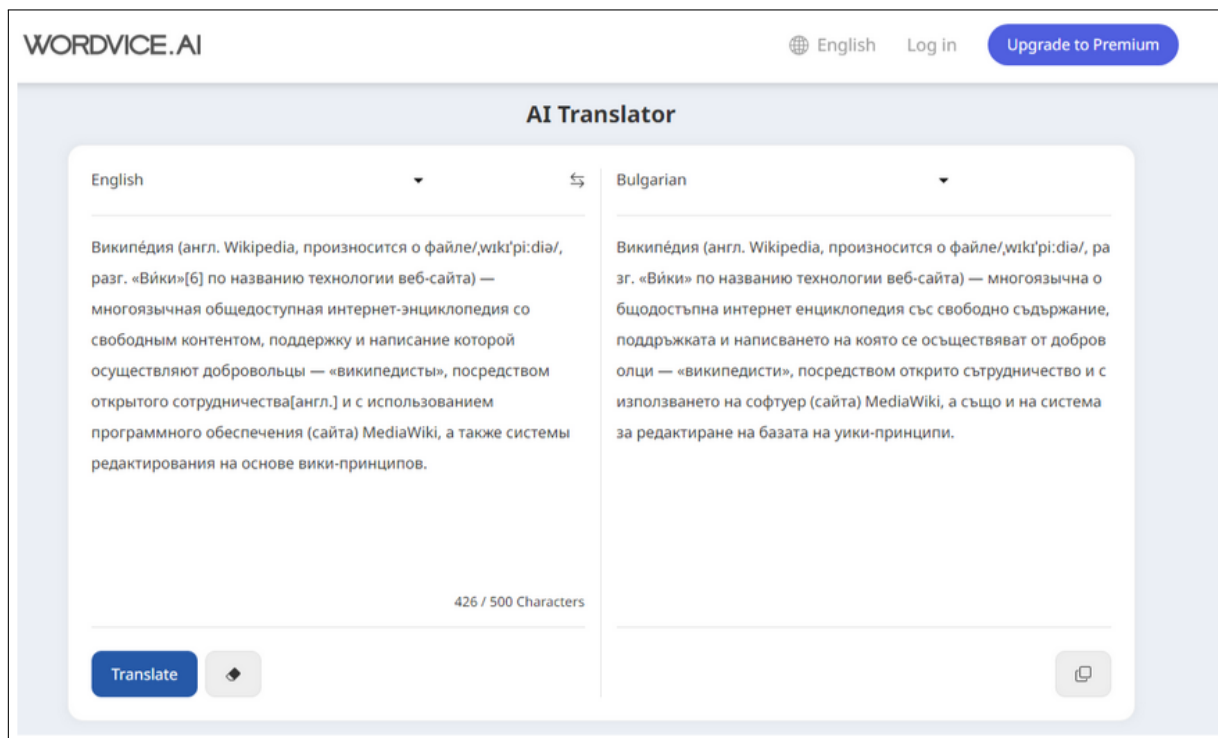


Рисунок 4.3 – Страница сервиса Wordvice

Он также использует нейронные сети для перевода текста, предоставляет ин-

теграцию с Microsoft Word и услуги обобщения и перефразирования текста при помощи искусственного интеллекта, а также поддерживает множество языков.

4.3 Выводы

1. Был рассмотрен сценарий использования web-приложения, что позволяет выделить ключевые функции web-приложения.
2. Анализ существующих решений в области перевода текста выявил как преимущества, так и недостатки сервисов-конкурентов. Сервисы DeepL, Google Translate и Wordvice предоставляют базовый функционал, включая перевод текста, загрузку текстовых документов и выбор языков. Среди ключевых недостатков отмечается отсутствие персонализации, невозможность выбора средства выполнения перевода и его стиля, а также отсутствие жалоб на переводы.

5 Проектирование web-приложения

5.1 Функциональность web-приложения

Функциональные возможности web-приложения представлены в диаграмме вариантов использования, представленной на рисунке 5.1.

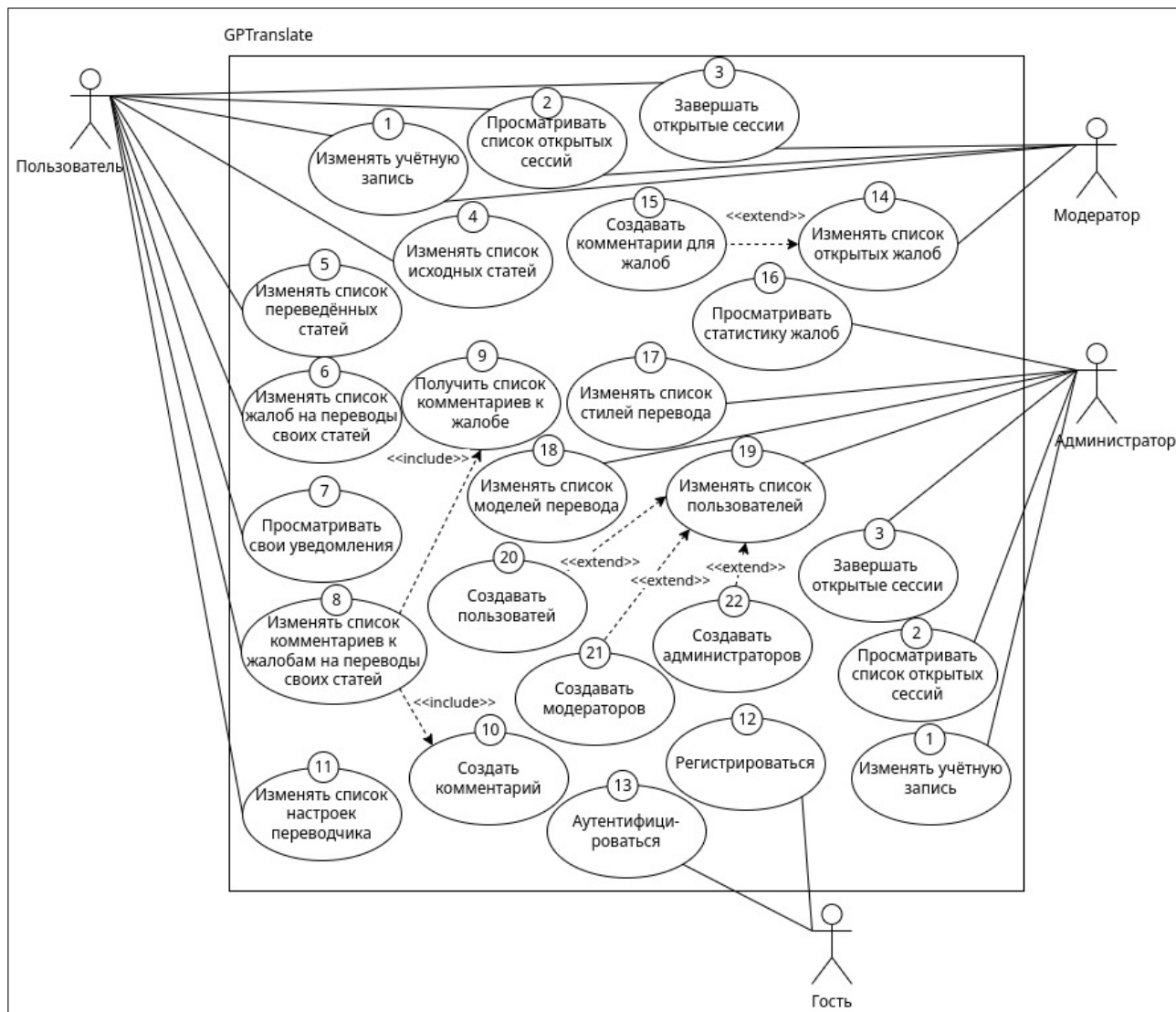


Рисунок 5.1 – Диаграмма вариантов использования web-приложения

Перечень ролей и их назначение приведены в таблице 5.1.

Таблица 5.1 – Назначение ролей пользователей в web-приложении

Роль	Назначение
Гость	Регистрация и аутентификация
Пользователь	Загрузка и запуск перевода статей, получение переводов, создание жалоб на переводы своих статей
Модератор	Рассмотрение жалоб на переводы

Продолжение таблицы 5.1

Администратор	Управление пользователями, запросами перевода, моделями перевода
---------------	--

Функциональные возможности пользователя с ролью «Гость» представлены в таблице 5.2.

Таблица 5.2 – Функциональные возможности пользователя с ролью «Гость»

Вариант использования	Пояснение
12 Регистрироваться	Гость может создать учётную запись при помощи электронной почты и пароля или OAuth 2.0-провайдера
13 Аутентифицироваться	Гость может аутентифицироваться при помощи электронной почты и пароля или OAuth 2.0-провайдера

После аутентификации гость становится либо пользователем, либо модератором, либо администратором. По этой причине пользователю недоступна аутентификация. Функциональные возможности пользователя с ролью «Пользователь» представлены в таблице 5.3.

Таблица 5.3 – Функциональные возможности пользователя с ролью «Пользователь»

Вариант использования	Пояснение
1 Изменять учётную запись	Изменять своё имя и пароль
2 Просматривать список открытых сессий	Получать список открытых сессий
3 Завершать открытые сессии	Блокировать доступ для всех открытых сессий
4 Изменять список исходных статей	Загружать из файла или вводить с клавиатуры исходные статьи, получать список исходных статей, изменять содержимое исходных статей, удалять их
5 Изменять список переведённых статей	Запускать перевод исходных статей, получать их список, оставлять оценку переводам статей, удалять переводы статей
6 Изменять список жалоб на переводы своих статей	Создавать жалобы на переводы своих статей, получать их список, закрывать открытые жалобы на переводы своих статей

Продолжение таблицы 5.3

7 Просматривать свои уведомления	Получать список непрочитанных уведомлений
8 Изменять список комментариев к жалобам на переводы своих статей	Получать список комментариев, создавать комментарии к открытым жалобам на переводы своих статей
9 Получить список комментариев к жалобе	Получить список комментариев к одной из своих жалоб
10 Создать комментарий	Создать комментарий к одной из своих жалоб
11 Изменять список настроек переводчика	Получать список своих конфигураций, создавать новые, обновлять и удалять существующие конфигурации

Модератор может рассматривать жалобы пользователей. Его функциональные возможности представлен в таблице 5.4.

Таблица 5.4 – Функциональные возможности пользователя с ролью «Модератор»

Вариант использования	Пояснение
1 Изменять учётную запись	Изменять своё имя и пароль
2 Просматривать список открытых сессий	Получать список открытых сессий
3 Завершать открытые сессии	Блокировать доступ для всех открытых сессий
14 Изменять список открытых жалоб	Получать список открытых жалоб на переводы, получать списки комментариев и создавать новые комментарии к ним, принимать или отклонять жалобы
15 Создавать комментарии для жалоб	Создавать комментарии для открытой жалобы

Функциональные возможности администратора представлен в таблице 5.5.

Таблица 5.5 – Функциональные возможности пользователя с ролью «Администратор»

Вариант использования	Пояснение
1 Изменять учётную запись	Изменять своё имя и пароль
2 Просматривать список открытых сессий	Получать список открытых сессий
3 Завершать открытые сессии	Блокировать доступ для всех открытых сессий

Продолжение таблицы 5.5

16 Просматривать статистику жалоб	Получать данные о том, какая часть переводов при помощи каждой модели получает жалобы и какая их доля удовлетворяется модераторами
17 Изменять список стилей перевода	Создавать новые стили, обновлять и удалять существующие
18 Изменять список моделей перевода	Добавлять информацию о новых моделях, изменять и удалять существующие записи
19 Изменять список пользователей	Получать список пользователей, создавать новых, изменять и удалять существующих
20 Создавать пользователей	Создавать объекты пользователей с ролью пользователя
21 Создавать модераторов	Создавать объекты пользователей с ролью модератора
22 Создавать администраторов	Создавать объекты пользователей с ролью администратора

Таким образом, пользователю доступны базовые операции, такие как операции над статьями и настройками перевода, модераторы могут управлять жалобами, а администраторы – управлять пользователями, моделями, запросами перевода и просматривать статистику жалоб на переводы.

5.2 Структура базы данных

Согласно схеме вариантов использования была создана база данных. Её структура представлена на рисунке 5.2.

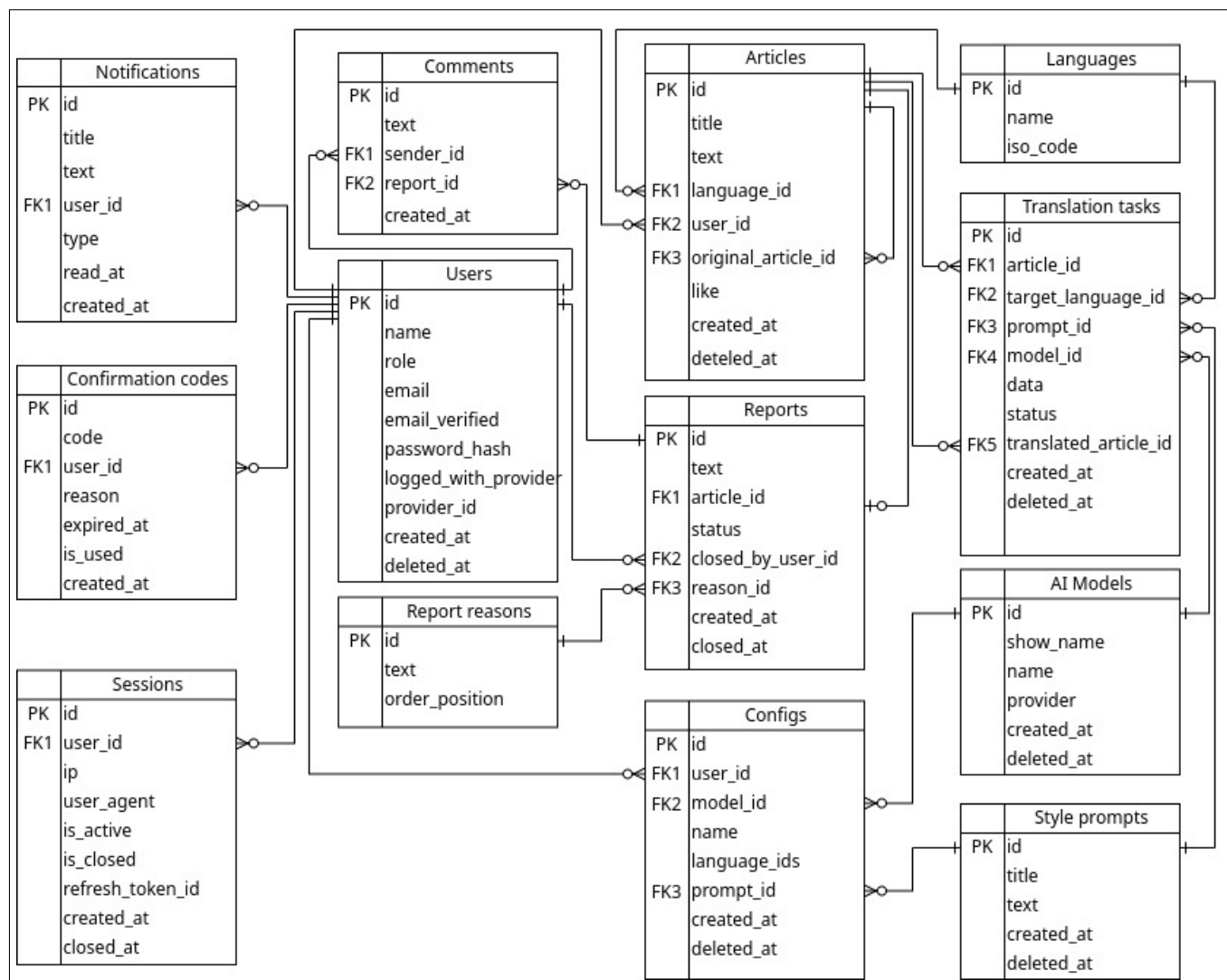


Рисунок 5.2 – Логическая схема базы данных

База данных содержит тринадцать таблиц, хранящих информацию о пользователях, сессиях, статьях и прочих данных. Типы данных были выбраны согласно [1]. Назначение таблиц базы данных представлено в таблице 5.6.

Таблица 5.6 – Назначение таблиц базы данных

Таблица	Назначение
Users	Хранит информацию о пользователях (имя, адрес электронной почты и хеш пароля для аутентификации и так далее)
Sessions	Хранит информацию о сессиях пользователей (идентификатор пользователя, флаг активности, время создания и так далее)
Confirmation_codes	Хранит информацию о кодах подтверждения адреса электронной почты и сброса пароля

Продолжение таблицы 5.6

Languages	Хранит информацию о доступных для перевода языках (название, ISO код)
Articles	Хранит информацию о статьях (заголовков, текст, идентификатор пользователя и так далее)
Report_reasons	Хранит информацию о доступных причинах для жалобы на перевод статьи (текст, позиция в списке для сортировки)
Reports	Хранит информацию о жалобах на переводы статей (идентификатор статьи, текст, идентификатор, причина и так далее)
Report_comments	Хранит информацию о комментариях к жалобам на переводы статей (текст, идентификатор пользователя, идентификатор жалобы, дата и время создания)
Style_prompts	Хранит информацию о запросах перевода с разными стилями (название, текст и так далее)
AI_Models	Хранит информацию о моделях искусственного интеллекта, используемых для перевода (название, поставщик и так далее)
Configs	Хранит информацию о конфигурациях переводчика, которые могут использоваться пользователями для упрощения запуска перевода своих статей (идентификаторы запроса перевода, модели, языков и так далее)
Translation_tasks	Хранит информацию о задачах перевода, которые считаются отдельным процессом и выполняются им (идентификаторы статьи, модели, исходного и конечного языков, статус и так далее)
Notifications	Хранит информацию об уведомлениях пользователей (идентификатор пользователя, текст, тип уведомления и так далее)

Описание столбцов таблицы Users представлено в таблице 5.7.

Таблица 5.7 – Описание таблицы Users

Название столбца	Тип данных	Описание
id	uuid	Идентификатор пользователя, первичный ключ
name	varchar (20)	Имя пользователя
email	varchar (50)	Адрес электронной почты пользователя
email_verified	boolean	Флаг, указывающий, был ли подтверждён адрес электронной почты пользователя
password_hash	varchar (60)	Хеш пароля соискателя
role	enum user_role	Роль пользователя (пользователь, модератор, администратор)
logged_with_provider	varchar	Название провайдера OAuth 2.0, использовавшегося для регистрации
provider_id	varchar	Идентификатор пользователя, полученный от провайдера OAuth при регистрации
created_at	timestamp without timezone	Дата и время создания пользователя без часового пояса
deleted_at	timestamp without timezone	Дата и время удаления пользователя без часового пояса

Таблица Sessions хранит данные о сессиях пользователей. Описание её столбцов представлено в таблице 5.8.

Таблица 5.8 – Описание таблицы Sessions

Название столбца	Тип данных	Описание
id	uuid	Идентификатор сессии, первичный ключ

Продолжение таблицы 5.8

user_id	uuid	Идентификатор пользователя, который создал данную сессию, внешний ключ
ip	varchar (15)	IPv4 адрес узла, из которого была открыта сессия
user_agent	varchar (100)	User agent клиента (например, браузера)
is_closed	boolean	Флаг, указывающий, была ли сессия закрыта
refresh_token_id	uuid	Идентификатор refresh токена, связанного с данной сессией
created_at	timestamp without timezone	Дата и время создания сессии без часового пояса
closed_at	timestamp without timezone	Дата и время закрытия сессии без часового пояса

Описание столбцов таблицы Confirmation_codes представлено в таблице 5.9.

Таблица 5.9 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	integer	Идентификатор кода, первичный ключ
code	varchar	Строковое значение кода
reason	enum confirmationtype	Тип кода (подтверждение адреса электронной почты, сброс пароля)
user_id	uuid	Идентификатор пользователя, для которого предназначен данный код подтверждения, внешний ключ
expired_at	timestamp without timezone	Временная отметка, после которой код будет считаться истёкшим

Продолжение таблицы 5.9

is_used	boolean	Флаг, указывающий, был ли код использован
created_at	timestamp without timezone	Дата и время создания кода без часового пояса

Таблица Languages хранит информацию о языках, доступных для перевода. Описание её столбцов представлено в таблице 5.10.

Таблица 5.10 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	integer	Идентификатор языка, первичный ключ
name	varchar	Отображаемое название языка
iso_code	varchar	ISO код языка

Таблица Articles хранит информацию об исходных и переведённых статьях. Описание её столбцов представлено в таблице 5.11.

Таблица 5.11 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	uuid	Идентификатор статьи, первичный ключ
title	varchar (50)	Название статьи
text	text	Текст статьи
user_id	uuid	Идентификатор пользователя, которому принадлежит статья, внешний ключ
language_id	integer	Идентификатор языка статьи, внешний ключ
original_article_id	uuid	Идентификатор статьи, переводом которой является данная статья, внешний ключ
like	boolean	Флаг, указывающий, какую оценку пользователь поставил переводу (положительную, отрицательную, не поставил оценку)

Продолжение таблицы 5.11

created_at	timestamp timezone	without	Дата и время создания статьи без часового пояса
deleted_at	timestamp timezone	without	Дата и время удаления статьи без часового пояса

Описание столбцов таблицы Report_reasons представлено в таблице 5.12.

Таблица 5.12 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	integer	Идентификатор причины, первичный ключ
text	varchar	Текст причины
order_position	integer	Положение причины в списке при сортировке

Описание столбцов таблицы Reports представлено в таблице 5.13.

Таблица 5.13 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	uuid	Идентификатор жалобы, первичный ключ
text	varchar (1024)	Текст жалобы
article_id	uuid	Идентификатор статьи, на которую была оставлена жалоба, внешний ключ
status	enum reportstatus	Статус жалобы (открыта, закрыта пользователем, отклонена, удовлетворена)
closed_by_user_id	uuid	Идентификатор пользователя, закрывшего жалобу (пользователь, которому принадлежит статья или модератор), внешний ключ
reason_id	int	Идентификатор причины, по которой была оставлена жалоба, внешний ключ

Продолжение таблицы 5.13

created_at	timestamp timezone	without	Дата и время создания жалобы без часового пояса
closed_at	timestamp timezone	without	Дата и время закрытия жалобы без часового пояса

Описание столбцов таблицы Report_comments представлено в таблице 5.14.

Таблица 5.14 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	uuid	Идентификатор комментария, первичный ключ
text	varchar (100)	Текст комментария
sender_id	uuid	Идентификатор пользователя, оставившего комментарий, внешний ключ
report_id	uuid	Идентификатор жалобы, к которой был оставлен комментарий, внешний ключ
created_at	timestamp timezone	without Дата и время создания комментария без часового пояса

Описание столбцов таблицы Style_prompts представлено в таблице 5.15.

Таблица 5.15 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	integer	Идентификатор запроса, первичный ключ
title	varchar (20)	Название запроса
text	varchar	Текст запроса
created_at	timestamp timezone	without Дата и время создания запроса без часового пояса
deleted_at	timestamp timezone	without Дата и время удаления запроса без часового пояса

Описание столбцов таблицы AI_Models представлено в таблице 5.16.

Таблица 5.16 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	integer	Идентификатор модели, первичный ключ
show_name	varchar (50)	Отображаемое название модели
name	varchar	Название модели
provider	varchar	Поставщик модели
created_at	timestamp without timezone	Дата и время создания записи о модели без часового пояса
deleted_at	timestamp without timezone	Дата и время удаления записи о модели без часового пояса

Таблица Configs хранит информацию о конфигурациях переводчика. Описание её столбцов представлено в таблице 5.17.

Таблица 5.17 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	integer	Идентификатор конфигурации, первичный ключ
name	varchar (20)	Название конфигурации
user_id	uuid	Идентификатор пользователя, создавшего конфигурацию, внешний ключ
prompt_id	integer	Идентификатор запроса перевода, внешний ключ
language_ids	integer []	Идентификаторы языков перевода
model_id	integer	Идентификатор модели перевода, внешний ключ
created_at	timestamp without timezone	Дата и время создания конфигурации без часового пояса
deleted_at	timestamp without timezone	Дата и время удаления конфигурации без часового пояса

Таблица `Translation_tasks` хранит информацию о задачах перевода. Данная информация используется для определения текста исходной статьи, конечного языка и так далее. Описание столбцов таблицы представлено в таблице 5.18.

Таблица 5.18 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
<code>id</code>	<code>uuid</code>	Идентификатор задачи, первичный ключ
<code>article_id</code>	<code>uuid</code>	Идентификатор исходной статьи, внешний ключ
<code>target_language_id</code>	<code>integer</code>	Идентификатор конечного языка, внешний ключ
<code>prompt_id</code>	<code>integer</code>	Идентификатор запроса перевода, внешний ключ
<code>model_id</code>	<code>integer</code>	Идентификатор модели перевода, внешний ключ
<code>status</code>	<code>enum translationtaskstatus</code>	Статус задачи (создана, в процессе выполнения, завершена успешно, завершена с ошибкой)
<code>data</code>	<code>jsonb</code>	Дополнительная информация о задаче (текст ошибки)
<code>translated_article_id</code>	<code>uuid</code>	Идентификатор переведённой статьи, внешний ключ
<code>created_at</code>	<code>timestamp without timezone</code>	Дата и время создания задачи без часового пояса
<code>deleted_at</code>	<code>timestamp without timezone</code>	Дата и время удаления задачи без часового пояса

Описание столбцов таблицы `Notifications` представлено в таблице 5.19.

Таблица 5.19 – Назначение таблиц базы данных

Название столбца	Тип данных	Описание
id	uuid	Идентификатор уведомления, первичный ключ
title	varchar	Заголовок уведомления
text	varchar	Текст уведомления
user_id	uuid	Идентификатор пользователя, которому предназначено уведомление, внешний ключ
type	enum notificationtype	Тип уведомления (информационное, предупреждение, ошибка)
created_at	timestamp without timezone	Дата и время создания записи о модели без часового пояса
read_at	timestamp without timezone	Дата и время удаления записи о модели без часового пояса

Назначение связей приведено в таблице 5.20.

Таблица 5.20 – Назначение таблиц базы данных

Связь	Назначение
Users.id – Notifications.user_id	Идентификатор пользователя, которому адресовано уведомление
Users.id – Confirmation_codes.user_id	Идентификатор пользователя, которому предназначен код подтверждения
Users.id – Sessions.user_id	Идентификатор пользователя, который создал сессию
Users.id – Articles.user_id	Идентификатор пользователя, который загрузил статью или запустил перевод исходной статьи
Users.id – Configs.user_id	Идентификатор пользователя, которому принадлежит конфигурация переводчика
Users.id – Commens.sender_id	Идентификатор пользователя, отправившего комментарий

Продолжение таблицы 5.20

Users.id – Reports.closed_by_user_id	Идентификатор пользователя, закрывшего жалобу (создавшего её пользователя или любого модератора)
Report_reasons.id – Reports.reason_id	Идентификатор причины, по которой была создана жалоба на перевод статьи
Articles.id – Articles.original_article_id	Идентификатор исходной статьи, из которой был создан перевод
Articles.id – Translation_tasks.article_id	Идентификатор статьи, которую необходимо перевести
Articles.id – Translation_tasks.translated_article_id	Идентификатор перевода статьи
Articles.id – Reports.article_id	Идентификатор перевода, на который была создана жалоба
Languages.id – Articles.language_id	Идентификатор языка статьи
Languages.id – Translation_tasks.target_language_id	Идентификатор конечного языка, на который необходимо перевести статью
Reports.id – Comments.report_id	Идентификатор жалобы, под которой был оставлен комментарий
AI_Models.id – Translation_tasks.model_id	Идентификатор записи о модели искусственного интеллекта, которая используется для перевода статьи
AI_Models.id – Configs.model_id	Идентификатор записи о модели искусственного интеллекта
Style_prompts.id – Translation_tasks.prompt_id	Идентификатор запроса перевода, который используется для перевода статьи
Style_prompts.id – Configs.prompt_id	Идентификатор запроса перевода

Таким образом, была спроектирована база данных для долговременного хранения информации web-приложения.

5.3 Архитектура web-приложения

Для обеспечения вспомогательных функций web-приложения (отправка почты, выполнение перевода, отправка уведомлений между компонентами системы и так далее) используются дополнительные компоненты.

Для запуска многоконтейнерных Docker-приложений используется инструмент Docker Compose. Он управляет набором контейнеров, в которых работают прочие компоненты web-приложения.

Для хранения данных используется СУБД PostgreSQL 17.

Для обслуживания web-приложение и предоставления доступа к скомпилированному пакету фронтэнд-приложения, созданному с использованием Vue.js, используется web-сервер Nginx.

Для асинхронного обмена сообщениями между компонентами системы используется брокер сообщений Apache Kafka.

Для обработки сообщений, передаваемых через Kafka, используются два процесса подписчика. Они принимают сообщения из очереди и обрабатывают поступившие команды, такие как перевод статьи и отправка электронной почты для подтверждения регистрации или сброса пароля.

Для быстрого доступа к данным, которые часто используются, например, идентификаторам закрытых сессий, и для передачи уведомлений пользователю используется in-memory база данных Redis.

Архитектура web-приложения представлена на рисунке 5.3.

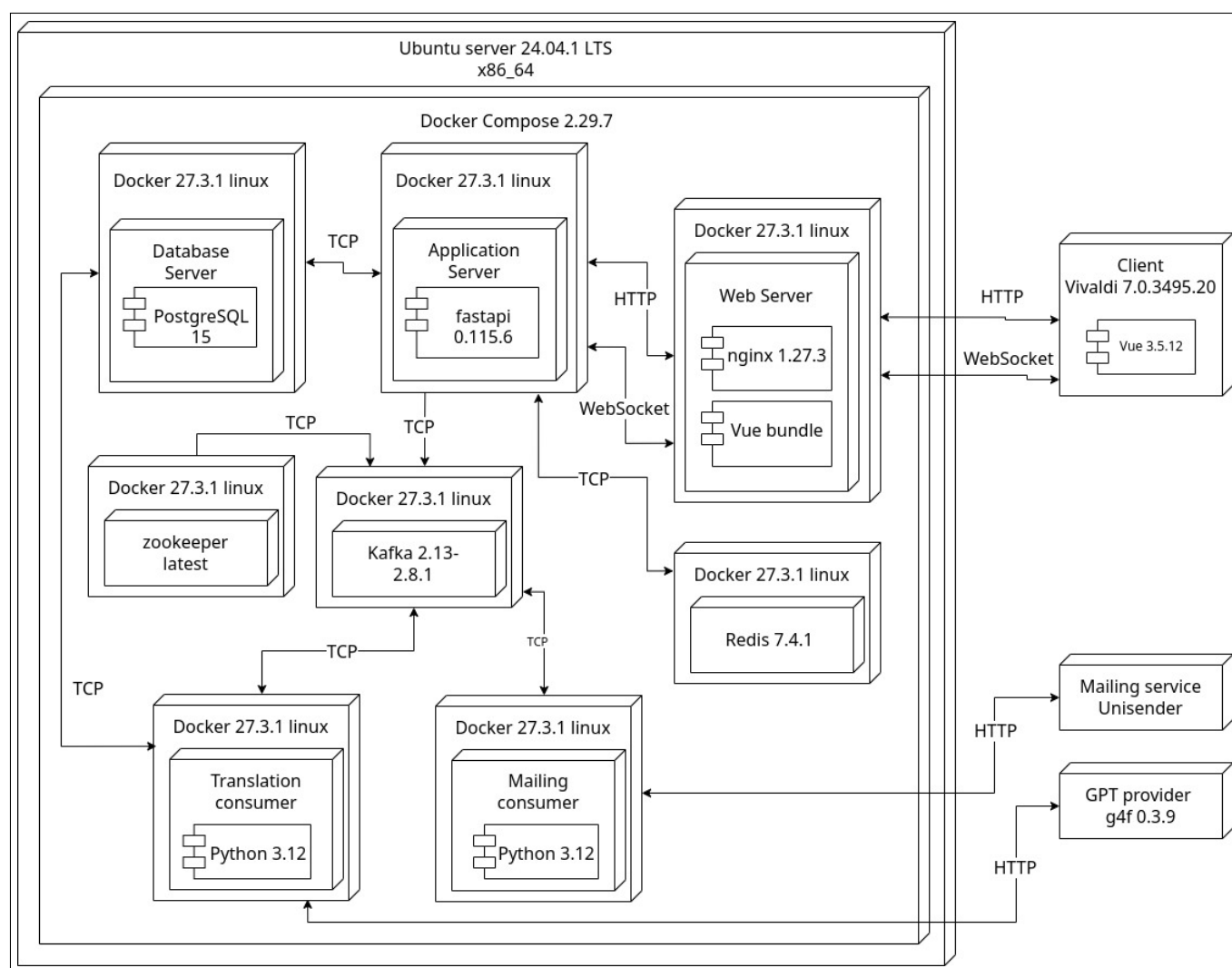


Рисунок 5.3 – Архитектура web-приложения

Пояснение назначения каждого элемента web-приложения на архитектурной диаграмме представлено в таблице 5.21.

Таблица 5.21 – Назначение элементов архитектурной схемы web-приложения

Элемент	Назначение
Web Server (nginx)	Принимать запросы клиента, обеспечивать работу HTTPS, предоставлять статические файлы фронтэнд-части web-приложения
Database Server (PostgreSQL)	Хранить данные, которые должны храниться длительное время
Kafka	Обеспечивать обмен сообщениями между компонентами web-приложения
Application Server	Обрабатывать запросы пользователя
Translation consumer	Переводить статьи при помощи внешнего сервиса
Mailing consumer	Отправлять электронные письма при помощи внешнего сервиса
Redis	Хранить данные с маленьким сроком жизни, выступать транспортом для отправки уведомлений о завершении перевода статей
zookeeper	Поддерживать работу Kafka, хранить информацию об участниках групп
GPT provider	Переводить тексты по запросу
Mailing service	Отправлять электронные письма по запросу
Client (Vivaldi)	Отображать фронтэнд-часть web-приложения, отправлять запросы пользователя, отображать ответы сервера

Таким образом, web-приложение состоит из различных компонентов, каждый из которых выполняет собственные функции.

5.4 Выводы

1. Была рассмотрена функциональность web-приложения «GPTTranslate» для всех ролей: гостя, пользователя, модератора и администратора. Гостям доступна регистрация и аутентификация. Пользователи могут загружать статьи, переводить их, а также управлять своими конфигурациями переводчика и оставлять жалобы на переведённые статьи. Модераторы рассматривают жалобы, а администраторы могут управлять списками пользователей, моделей и стилей перевода, а также получать статистику жалоб по моделям и стилям перевода. Общее количество функций web-приложения составляет 22.

2. Рассмотрена логическая схема базы данных web-приложения, которая включает 13 таблиц. Таблицы хранят данные о пользователях, статьях, конфигурациях и других.
3. Рассмотрена архитектура web-приложения. Использование Kafka позволяет сервисам передавать сообщения между собой, Redis позволяет хранить данные с малым сроком жизни, а PostgreSQL – долговременные данные.

6 Реализация web-приложения

6.1 Обоснование выбора программной платформы

Для реализации web-приложения был выбран язык программирования Python [2] и фреймворк FastAPI [3]. FastAPI представляет собой web-фреймворк для создания API на языке Python. Он позволяет обрабатывать запросы асинхронно и поддерживает протокол WebSocket. Для сериализации, десериализации и валидации запросов использовалась библиотека Pydantic [4].

Для долговременного хранения данных web-приложения была выбрана распределённая СУБД PostgreSQL [5], обладающая следующими преимуществами: бесплатность, расширяемость, большое сообщество, широкая поддержка среды инструментов разработки программного обеспечения.

Для создания моделей, соответствующих таблицам в реляционной базе данных, была выбрана библиотека SQLAlchemy [6]. Она предоставляет уровень абстракции над объектами базы данных, позволяя работать с ними как с объектами Python, а также предоставляет возможность создавать сложные запросы при помощи функций Python.

Для управления миграциями был выбран инструмент Alembic [7]. Данный инструмент позволяет отслеживать изменения в структуре базы данных, а также предоставляет возможность автоматической генерации миграций на основе изменений в моделях. Также Alembic предоставляет возможность отката к более ранней версии базы данных.

Для перевода текста используется сервис g4f [8]. Он выступает как посредник между web-приложением и публичными API различных провайдеров, обеспечивая работу web-приложения и упрощая его настройку. Сервис translation-consumer считывает задачи на перевод из очереди Kafka, отправляет запросы по указанному в переменных окружения адресу с необходимой полезной нагрузкой (текст, который нужно перевести, текст стиля перевода, название модели, название провайдера) и на основе полученных ответов создаёт объекты переведённых статей.

Для отправки электронной почты был выбран сервис Unisender [9], который предоставляет API для создания рассылок и отправки одиночных писем.

6.2 Реализация серверной части web-приложения

В соответствии с диаграммой вариантов использования функции, доступные пользователям, были реализованы в исходном коде. Исходный код web-приложения представлен в Приложении А.

6.2.1 Изменение учётной записи

Функция "изменение учётной записи"(1) в исходном коде реализована функциями `change_name`, `request_password_restoration_code` и `restore_password`.

Функция `change_name` находится в модуле `src.routers.users.views.py`. Она принимает HTTP PATCH запрос по пути `"/users/user_id/name/"` и изменяет имя пользователя на новое, полученное из тела запроса.

Функция `request_password_restoration_code` находится в модуле `src.routers.auth.views.py`. Она принимает HTTP POST запрос по пути `"/auth/restore-password/request/"` с адресом электронной почты пользователя в параметрах строки запроса, создаёт код подтверждения смены пароля в базе данных и отправляет письмо по адресу электронной почты со ссылкой на страницу смены пароля.

Функция `restore_password` находится в модуле `src.routers.auth.views.py`. Она принимает HTTP POST запрос по пути `"/auth/restore-password/confirm/"` с кодом подтверждения и новым паролем, проверяет существование полученного кода и изменяет хранимый в базе данных хэш пароля пользователя на хэш полученного нового пароля.

6.2.2 Просмотр открытых сессий

Функция "просмотр открытых сессий"(2) в исходном коде реализована функцией `get_sessions`. Данная функция находится в модуле `src.routers.sessions.views.py`. Она принимает HTTP GET запрос по пути `"/sessions/"` и возвращает список всех открытых сессий текущего пользователя из базы данных (сессий, у которых значение в столбце `closed_at` равняется NULL).

6.2.3 Завершение открытых сессий

Функция "завершение открытых сессий"(3) в исходном коде реализована функцией `close_sessions`. Данная функция находится в модуле `src.routers.sessions.views.py`. Она принимает HTTP POST запрос по пути `"/sessions/close/"` и закрывает все открытые сессии текущего пользователя (устанавливает в столбец `closed_at` текущее время сервера).

6.2.4 Изменение списка исходных статей

Функция "изменение списка исходных статей"(4) в исходном коде реализована функциями `upload_article`, `update_article`, `delete_article`. Данные функции находятся в модуле `src.routers.articles.views.py`.

Функция `upload_article` находится в модуле `src.routers.users.views.py`. Она принимает HTTP POST запрос по пути `"/articles/"` и добавляет в базу данных строку с информацией об исходной статье, десериализованной из тела запроса. Значение столбца `user_id` берётся из маркера доступа пользователя, передаваемого через Cookie.

Функция `update_article` находится в модуле `src.routers.users.views.py`. Она принимает HTTP PUT запрос по пути `"/articles/article_id/"` проверяет принадлежность исходной статьи текущему пользователю по идентификатору статьи, полученному

из пути запроса, и обновляет запись о статье согласно данным, десериализованным из тела запроса.

Функция `delete_article` находится в модуле `src.routers.users.views.py`. Она принимает HTTP DELETE запрос по пути `"/articles/article_id/` проверяет принадлежность исходной статьи текущему пользователю по идентификатору статьи, полученному из пути запроса, и удаляет исходную или переведённую статью по идентификатору.

6.2.5 Изменение списка переведённых статей

Функция "изменение списка переведённых статей"(5) в исходном коде реализована функциями `delete_article`, рассмотренной выше, и `create_translation`.

Функция `create_translation` находится в модуле `src.routers.translation.views.py`. Она принимает HTTP POST запрос по пути `"/translation/"` и отправляет в Kafka сообщение подписчику-переводчику о запуске перевода. Тело запроса должно содержать идентификатор статьи, массив идентификаторов конечных языков, на которые требуется выполнить перевод, а также идентификаторы стиля и модели перевода.

6.2.6 Изменение списка жалоб на переводы своих статей

Функция "изменение списка жалоб на переводы своих статей"(6) в исходном коде реализована функциями `create_report`, `update_report`, `update_report_status`.

Функция `create_report` находится в модуле `src.routers.reports.views.py`. Она принимает HTTP POST запрос по пути `"/articles/article_id/report/` проверяет принадлежность переведённой статьи текущему пользователю по идентификатору статьи, полученному из пути запроса, и добавляет в базу данных строку с информацией о жалобе (текст и идентификатор причины жалобы из тела запроса).

Функция `create_report` находится в модуле `src.routers.reports.views.py`. Она принимает HTTP PUT запрос по пути `"/articles/article_id/report/` проверяет принадлежность переведённой статьи текущему пользователю и обновляет информацию о жалобе (текст и причина жалобы в теле запроса).

Функция `create_report` находится в модуле `src.routers.reports.views.py`. Она принимает HTTP PATCH запрос по пути `"/articles/article_id/report/status/` проверяет, имеет ли пользователь право устанавливать жалобе новый статус, и обновляет статус жалобы по идентификатору статьи.

6.2.7 Просмотр своих уведомлений

Функция "просмотр своих уведомлений"(7) в исходном коде реализована функцией `get_notifications_list`. Данная функция находится в модуле `src.routers.notifications.vi`. Она принимает HTTP GET запрос по пути `"/notifications/"` и возвращает список непрочитанных (имеющих в столбце `read_at` значение NULL) уведомлений пользователя из базы данных.

6.2.8 Изменение списка комментариев к жалобам на переводы своих статей

Функция "изменение списка комментариев к жалобам на переводы своих статей"(8) включает в себя ровно две функции: "получение списка комментариев к жалобе"(9) и "создание комментария"(10).

Функция "получение списка комментариев к жалобе"(9) в исходном коде реализована функцией `get_comments`. Данная функция находится в модуле `src.routers.reports`. Она принимает HTTP GET запрос по пути `"/articles/article_id/report/comments/` проверяет, имеет ли право текущий пользователь получать список комментариев к этой жалобе по идентификатору статьи, полученному из пути запроса, и возвращает список комментариев к жалобе из базы данных.

Функция "создание комментария"(10) в исходном коде реализована функцией `create_comment`. Данная функция находится в модуле `src.routers.reports.views.py`. Она принимает HTTP POST запрос по пути `"/articles/article_id/report/comments/"` создаёт комментарий к жалобе по идентификатору статьи, к которой была оставлена жалоба.

6.2.9 Изменение списка настроек переводчика

Функция "изменение списка настроек переводчика"(11) в исходном коде реализована функциями `create_config`, `update_config` и `delete_config`.

Функция `create_config` находится в модуле `src.routers.config.views.py`. Она принимает HTTP POST запрос по пути `"/configs/` проверяет, занято ли название конфигурации переводчика для данного пользователя, и создаёт новую строку в базе данных с полученным названием, идентификатором модели перевода и стиля перевода, а также массивом идентификаторов конечных языков.

Функция `create_config` находится в модуле `src.routers.config.views.py`. Она принимает HTTP PUT запрос по пути `"/configs/config_id/` проверяет принадлежность конфигурации текущему пользователю и занято ли новое название конфигурации переводчика для данного пользователя, и обновляет строку в базе данных согласно десериализованным из тела запроса данным.

Функция `create_config` находится в модуле `src.routers.config.views.py`. Она принимает HTTP DELETE запрос по пути `"/configs/config_id/` проверяет принадлежность конфигурации текущему пользователю и удаляет конфигурацию по её идентификатору.

6.2.10 Регистрация

Функция "регистрация"(12) в исходном коде реализована функцией `register`. Данная функция находится в модуле `src.routers.auth.views.py`. Она принимает HTTP POST запрос по пути `"/auth/register/` проверяет, занят ли адрес электронной почты, и создаёт нового пользователя по имени, адресу электронной почты и паролю, полученным из тела запроса. Значение столбца `email_verified` устанавливается в `false`, и пользователь должен дополнительно подтвердить свой адрес электронной почты.

6.2.11 Аутентификация

Функция ”аутентификация”(13) в исходном коде реализована функцией `login`. Данная функция находится в модуле `src.routers.auth.views.py`. Она принимает HTTP POST запрос по пути `"/auth/login/` проверяет существование пользователя с полученными из тела запроса адресом электронной почты и паролем и аутентифицирует пользователя: закрывает открытые сессии по IP-адресу и `user_agent`, полученными из заголовков запроса, создаёт новую сессию пользователя и возвращает пару маркеров для доступа к ресурсам и обновления маркеров.

6.2.12 Изменение списка открытых жалоб

Функция ”изменение списка открытых жалоб”(14) в исходном коде реализована функцией `update_report_status`. Данная функция находится в модуле `src.routers.reports` и была рассмотрена выше.

6.2.13 Создание комментариев для жалоб

Функция ”создание комментариев для жалоб”(15) в исходном коде реализована функцией `create_comment`. Данная функция находится в модуле `src.routers.reports.views.py` и была рассмотрена выше.

6.2.14 Просмотр статистики жалоб

Функция ”просмотр статистики жалоб”(16) в исходном коде реализована функциями `get_models_stats` и `get_prompts_stats`.

Функция `get_models_stats` находится в модуле `src.routers.analytics.views.py`. Она принимает HTTP GET запрос по пути `"/analytics/models-stats/` и возвращает статистику жалоб по всем моделям перевода в базе данных: сколько было подано жалоб на переводы по каждой модели и какие статусы у этих жалоб на данный момент.

Функция `get_prompts_stats` находится в модуле `src.routers.analytics.views.py`. Она принимает HTTP GET запрос по пути `"/analytics/prompts-stats/` и аналогична функции `get_models_stats`, но возвращает статистику по стилям перевода, а не моделям перевода.

6.2.15 Изменение списка стилей перевода

Функция ”изменение списка стилей перевода”(17) в исходном коде реализована функциями `create_prompt`, `update_prompt` и `delete_prompt`.

Функция `create_prompt` находится в модуле `src.routers.prompts.views.py`. Она принимает HTTP POST запрос по пути `"/prompts/` проверяет, занято ли название стиля перевода существующей строкой в базе данных, и добавляет новый стиль перевода в базу данных по названию и тексту, полученным из тела запроса.

Функция `update_prompt` находится в модуле `src.routers.prompts.views.py`. Она принимает HTTP PUT запрос по пути `"/prompts/prompt_id/` проверяет, занято ли новое название стиля перевода, и обновляет строку в базе данных по идентификатору стиля перевода, полученному из пути запроса, согласно данным из тела запроса.

Функция `delete_prompt` находится в модуле `src.routers.prompts.views.py`. Она принимает HTTP DELETE запрос по пути `"/prompts/prompt_id/` проверяет существование стиля перевода по идентификатору из пути запроса и удаляет стиль по идентификатору.

6.2.16 Изменение списка моделей перевода

Функция "изменение списка моделей перевода"(18) в исходном коде реализована функциями `create_model`, `update_model` и `delete_model`.

Функция `create_model` находится в модуле `src.routers.models.views.py`. Она принимает HTTP POST запрос по пути `"/models/` проверяет, занято ли название модели перевода, и добавляет в базу данных строку с отображаемым названием, внутренним названием и провайдером, полученным из тела запроса.

Функция `update_model` находится в модуле `src.routers.models.views.py`. Она принимает HTTP PUT запрос по пути `"/models/model_id/` проверяет, занято ли новое название модели перевода, и обновляет строку в базе данных по идентификатору модели, полученному из пути запроса, согласно данным, полученным из тела запроса.

Функция `delete_model` находится в модуле `src.routers.models.views.py`. Она принимает HTTP DELETE запрос по пути `"/models/model_id/"` и удаляет модель перевода из базы данных по её идентификатору, полученному из тела запроса.

6.2.17 Изменение списка пользователей

Функция "изменение списка пользователей"(19) в исходном коде реализована функциями `create_user`, `update_user` и `delete_user`.

Функция `create_user` находится в модуле `src.routers.users.views.py`. Она принимает HTTP POST запрос по пути `"/users/"` и создаёт пользователя с заданным именем, адресом электронной почты, паролем и ролью. Эти данные десериализуются из тела запроса.

Функция `update_user` находится в модуле `src.routers.users.views.py`. Она принимает HTTP PUT запрос по пути `"/users/user_id/"` и обновляет строку в базы данных по идентификатору пользователя, полученному из пути запроса, согласно данным, полученным из тела запроса.

Функция `delete_user` находится в модуле `src.routers.users.views.py`. Она принимает HTTP DELETE запрос по пути `"/users/user_id/"` и удаляет пользователя по идентификатору, полученному из пути запроса.

Функции "создание пользователей"(20), "создание модераторов"(21) и "создание администраторов"(22) в исходном коде реализованы функцией `src.routers.users.vi` рассмотренной выше.

Для передачи данных от клиента серверу и обратно используется протокол HTTP и формат JSON. FastAPI автоматически проверяет тело запроса согласно указанной схеме, созданной при помощи Pydantic, что повысило читаемость кода.

6.3 Реализация базы данных

Согласно логической схеме базы данных, были созданы объекты базы данных. Модели SQLAlchemy объявлены в модуле `src.database.models.py`, представленном в Приложении А. Для изменения состояния базы данных использовался инструмент Alembic. Скрипт для создания базы данных и её объектов представлен в Приложении Б.

Для работы с базой данных в SQLAlchemy необходимо создать объект сессии. Предварительная настройка подключения представлена в листинге



Листинг 1 – Настройка подключения к базе данных

7 Технико-экономическое обоснование проекта

7.1 Общая характеристика разрабатываемого программного средства

Основной целью экономического раздела является экономическое обоснование целесообразности разработки web-приложения, представленного в дипломном проекте. В данном разделе проводится расчет затрат на всех стадиях разработки, а также анализ экономического эффекта в связи с использованием данного веб-приложения.

Разработанное web-приложение позволяет пользователям переводить значительные объёмы текста с одного языка на множество других языков.

Во время разработки дипломного проекта использовалась технология FastAPI для написания серверной части приложения и библиотека Vue.js для написания клиентской части приложения. Данное web-приложение разработано для последующего использования в коммерческих целях.

7.2 Методика обоснования цены

В современных рыночных экономических условиях web-приложение выступает преимущественно в виде продукции организаций, представляющей собой функционально завершённые и имеющие товарный вид web-приложения, реализуемые покупателям по рыночным отпускным ценам. Все завершённые разработки web-приложения являются научно-технической продукцией.

Широкое применение вычислительных технологий требует постоянного обновления и совершенствования web-приложения. Выбор эффективных проектов web-приложения связан с их экономической оценкой и расчетом экономического эффекта, который может определяться как у разработчика, так и у пользователя.

У разработчика экономический эффект выступает в виде чистой прибыли от реализации приложения, остающейся в распоряжении организации, а у пользователя – в виде экономии трудовых, финансовых ресурсов, получаемой за счет:

- снижения трудоемкости расчетов и алгоритмизации программирования и отладки программ;
- снижения расходов на материалы;
- ускорение ввода в эксплуатацию новых систем;
- улучшения показателей основной деятельности в результате использования веб-приложения.

Стоимостная оценка веб-приложения у разработчиков предполагает определение затрат, что включает следующие статьи:

- заработная плата исполнителей – основная и дополнительная;

- отчисления в фонд социальной защиты населения;
- отчисления по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний;
- расходы на оплату машинного времени;
- накладные расходы;
- прочие прямые затраты.

Для расчёта стоимости разработки web-приложения необходимо установить определённые параметры, представленные в таблице 7.1.

Таблица 7.1 – Параметры, применяемые при расчёте стоимости разработки

Параметр	Значение
Норматив ФСЗН+БГС	0,346
Норматив доп. ЗП	0,15
Норматив прочих затрат	0
Норматив накладных расходов	0,5
Норматив расходов на реализацию	0,1
Ставка НДС	20%
Прочие прямые расходы (стоимость подписки использования GPT, аренда сервера для разработки и тестирования)	149,1 руб.
Повышающий коэффициент ЗП	1

На основании затрат рассчитывается себестоимость и отпускная цена конечного web-приложения.

7.2.1 Стоимость разработки

Стоимость разработки напрямую зависит от заработной платы специалистов и их трудозатрат. Для определения величины основной заработной платы было проведено исследование заработных плат для специалистов в сфере веб-разработки. Источником данных служили открытые веб-порталы, различные форумы и общий средний уровень зарплате в сфере информационных технологий. Было установлено, что средняя месячная заработная плата дизайнера составляет 20 рублей в час, бизнес-аналитика – 20.83 рубля в час, технического лидера – 100 рублей в час, junior бэкэнд/фронтенд разработчика и тестировщика – 10 рублей в час, middle бэкэнд разработчика – 30 рублей в час, middle фронтенд разработчика – 28 рублей в час, middle тестировщика – 24 рубля в час.

Проект разрабатывался командой из бизнес-аналитика, технического лидера, дизайнера, а также junior и middle фронтенд и бекэнд разработчиков и тестировщиков на протяжении двух месяцев. Трудозатраты каждого работника представлены в таблице 7.2.

Таблица 7.2 – Трудозатраты и ставки оплаты работников

Базовая ставка в час, руб	Специалист	Трудозатраты, ч	Ставка в час, руб
20	Дизайнер	120	20
20,83	Бизнес-аналитик	24	20,83
100	Технический лидер	16	100
10	junior бэкенд разработчик	140	10
30	middle бэкенд разработчик	140	30
10	junior фронтенд разработчик	124	10
28	middle фронтенд разработчик	124	28
10	junior тестировщик	40	10
24	middle тестировщик	40	24

Затраты на заработную плату каждому работнику в зависимости от ставки и трудозатрат определяются по формуле 7.1.

$$C_p = CT((1 + H_{\text{доп. ЗП}})(1 + H_{\text{ФСЗН}}) + H_{\text{п}} + H_{\text{н}}) \quad (7.1)$$

[nosep, leftmargin=0pt, labelindent=0pt, itemsep=0pt, parsep=0pt]

где C_p – стоимость разработки;

C – ставка работника в час;

T – трудозатраты работника в часах;

$ЗП_{\text{осн}}$ – основная зарплата;

$ЗП_{\text{доп}}$ – дополнительная зарплата;

$H_{\text{ФСЗН}}$ – норматив отчислений в ФСЗН;

$H_{\text{доп. ЗП}}$ – норматив дополнительной зарплаты;

$H_{\text{п}}$ – норматив прочих затрат;

$H_{\text{н}}$ – норматив накладных расходов.

Таким образом было определено, что стоимость разработки составляет 33267.57 рублей.

7.2.2 Цена продажи

При расчёте цены продажи было решено использовать желаемую маржинальность 20%. Таким образом, определить цену с НДС можно по формуле 7.2.

$$C_{\text{НДС}} = C_p(1 + P)(1 + C_{\text{НДС}}) \quad (7.2)$$

[nosep, leftmargin=0pt, labelindent=0pt, itemsep=0pt, parsep=0pt]

где $C_{\text{НДС}}$ – цена продажи, включая НДС;

$C_{\text{НДС}}$ – ставка НДС;

P – рентабельность.