**Содержание**

[Введение 3](#_Toc182238467)

[1 Постановка задачи и обзор аналогичных решений 4](#_Toc182238468)

[1.1 Постановка задачи 4](#_Toc182238469)

[1.2 Обзор аналогичных решений 5](#_Toc182238470)

[1.3 Выводы 6](#_Toc182238471)

[2 Проектирование web-приложения 8](#_Toc182238472)

[2.1 Функциональность web-приложения 8](#_Toc182238473)

[2.2 Проектирование базы данных 10](#_Toc182238474)

[2.3 Архитектура web-приложения 18](#_Toc182238475)

[3 Реализация web-приложения 20](#_Toc182238476)

[3.1 Реализация бэкэнда 20](#_Toc182238477)

[3.2 Реализация фронтэнда 20](#_Toc182238478)

[4 Тестирование web-приложения 21](#_Toc182238479)

[5 Руководство пользователя 22](#_Toc182238480)

[Заключение 23](#_Toc182238481)

[Список используемых источников 24](#_Toc182238482)

[Приложение А 25](#_Toc182238483)

Введение

В современном мире глобализации и цифровых технологий трудно переоценить значение точного и быстрого перевода текстов. Перевод становится важным инструментом для преодоления языковых барьеров и эффективной коммуникации в различных сферах жизни — от корпоративных блогов до локализации web-сайтов. Это особенно актуально в условиях постоянного увеличения объема информации и необходимости оперативного доступа к контенту на разных языках.

Наша жизнь сегодня пронизана международным взаимодействием и многообразием информации, что делает переводы не просто полезными, но и необходимыми. Профессиональные и частные пользователи, работающие с большими объемами текста, сталкиваются с трудностью выбора подходящих решений для качественного перевода. Часто этот процесс может быть длительным и трудоемким.

Именно поэтому возникла необходимость в инновационных решениях, способных упростить и ускорить процесс перевода текстов. Web-приложение «GPTranslate» удовлетворяет эту потребность, предлагая пользователям мощный инструмент для перевода текстов при помощи искусственного интеллекта. Оно предоставляет возможность выбрать модель перевода и настроить стиль в зависимости от конкретных требований и целей, будь то технический перевод, маркетинговый контент или любой другой тип текста.

Основная цель проекта – создать универсальное и удобное в использовании web-приложение, которое позволит пользователям эффективно справляться с переводами текстов различной сложности и объема. Важным аспектом является предоставление настроек для выбора подходящей модели и стиля перевода, что обеспечивает высокое качество и точность перевода.

Актуальность данного проекта обусловлена растущим спросом на качественные решения для перевода текстов в условиях глобализации и необходимости оперативной локализации контента. Web-приложение «GPTranslate» сокращает время на перевод и предоставляет пользователям возможность гибкой настройки параметров перевода, что делает ее ценным инструментом в сфере текстового контента.

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.

1 Постановка задачи и обзор аналогичных решений

1.1 Постановка задачи

Web-приложение предназначено для перевода статей и управления этим процессом. Функционал приложения разделен на роли: пользователь, модератор и администратор.

Пользователь начинает с регистрации (или создания администратором) и аутентификации в системе. После успешной аутентификации он может выполнять следующие действия:

– создать статью: пользователь может загрузить статью из документа или набрать текст самостоятельно. По желанию пользователь может выбрать язык исходной статьи;

– создавать и изменять свои конфигурации переводчика: пользователь может создавать, изменять и удалять конфигурации, включающие в себя модель для перевода, стиль перевода и конечные языки, на которые необходимо выполнить перевод;

– выполнять перевод статьи: пользователь выбирает статью для перевода, выбирает модель, стиль и языки, на которые необходимо выполнить перевод, и запускает процесс перевода;

– оценивать перевод статьи: пользователь может оценивать переводы своих статей отметками «нравится» и «не нравится»;

– создавать жалобу на перевод статьи: если пользователь считает, что перевод некачественный, он может подать жалобу, указав причины. Также пользователь может создавать комментарии к своим открытым жалобам;

– редактировать личную информацию: пользователь может изменять своё имя и сбрасывать пароль на новый.

Модератор может быть создан администратором. После аутентификации он может выполнять следующие действия:

– просматривать и изменять список жалоб: модератор может получать список открытых жалоб, удовлетворять или отклонять их;

–оставлять комментарии к открытым жалобам.

Администратор может быть создан специальной консольной командой или другим администратором. После аутентификации ему доступны следующие действия:

– управление списком пользователей: администратор может получать список всех пользователей, создавать, изменять и удалять любых пользователей;

– управление списком запросов перевода: администратор может получать список запросов к моделям перевода, создавать новые объекты запросов и изменять и удалять существующие;

– управление списком моделей перевода: администратор может создавать записи о моделях перевода и изменять и удалять существующие записи;

– получать статистику жалоб: администратору доступна статистика, показывающая, сколько жалоб было подано и удовлетворено по разным моделям и запросам перевода, что позволит ему удалять неудовлетворительно работающие модели и запросы.

1.2 Обзор аналогичных решений

В качестве первого аналогичного решения был выбран сервис DeepL. Он известен своей высокой точностью и естественностью перевода по сравнению с другими сервисами. Внешний вид страницы сервиса представлен на рисунке 1.1.

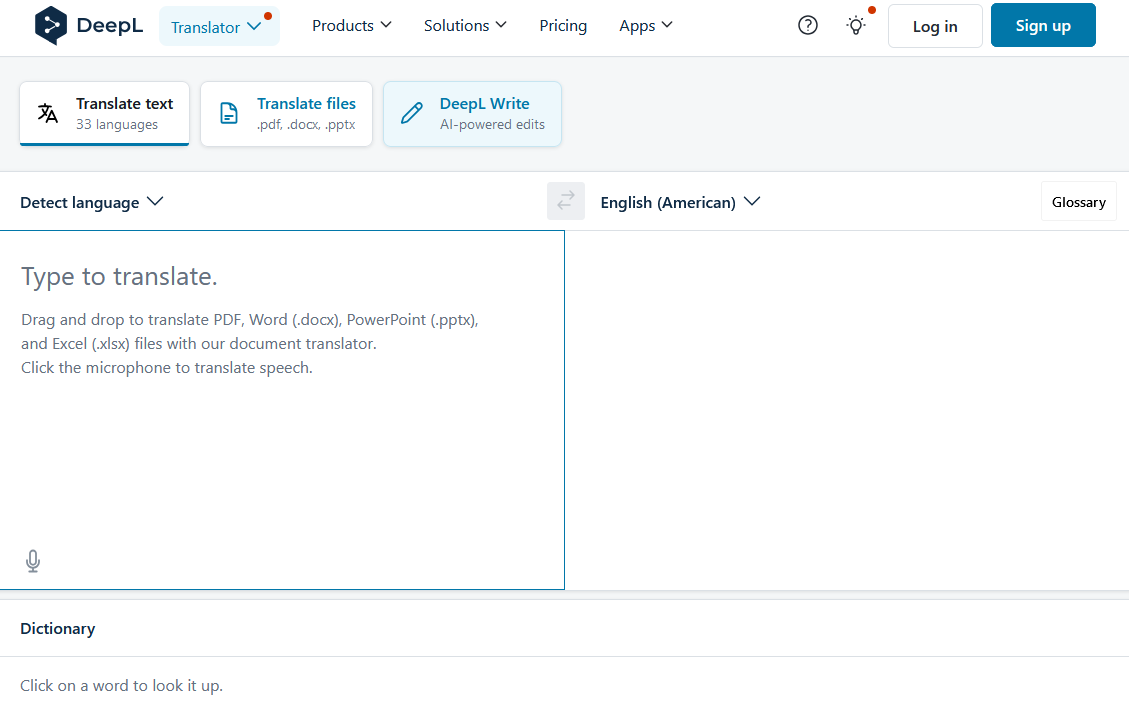


Рисунок 1.1 – Страница сервиса DeepL

Он использует нейронные сети для обработки текста и предлагает перевод на множество языков.

В качестве второго аналогичного решения был рассмотрен сервис Google Translate. Внешний вид страницы данного сервиса представлен на странице 1.2.

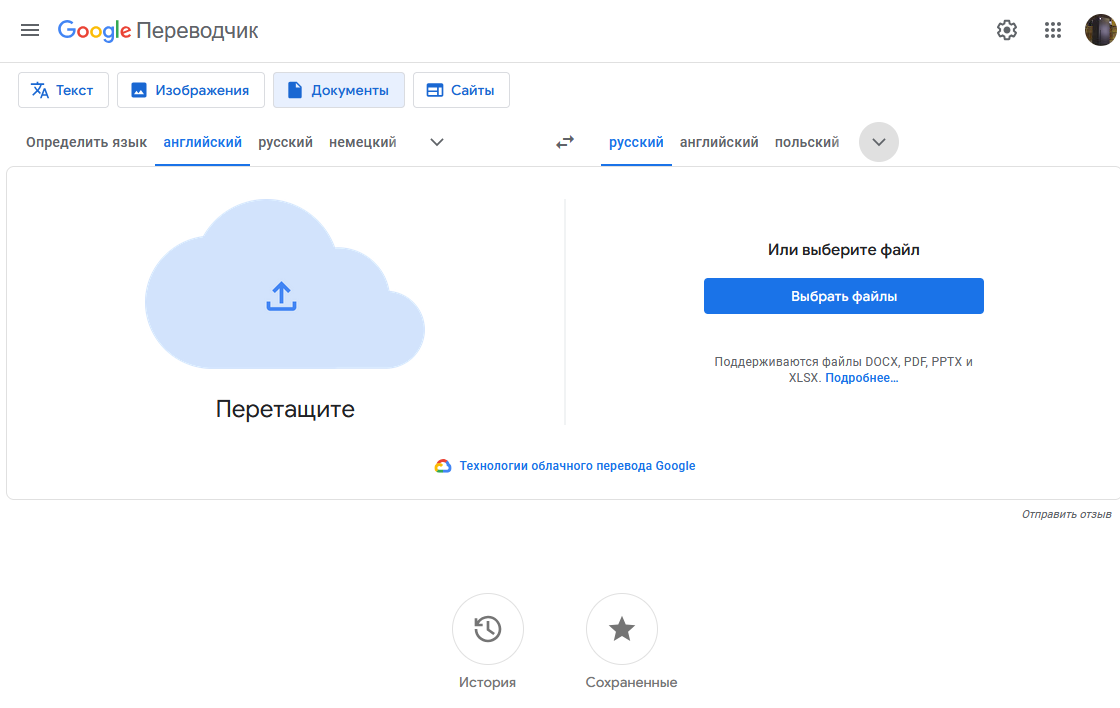


Рисунок 1.2 – Страница сервиса Google Translate

Это один из самых известных и широко используемых сервисов машинного перевода. Google Translate использует мощные нейронные сети и постоянно улучшается.

В качестве третьего аналогичного решения был рассмотрен сервис Wordvice. Внешний вид его страницы представлен на рисунке 1.3.

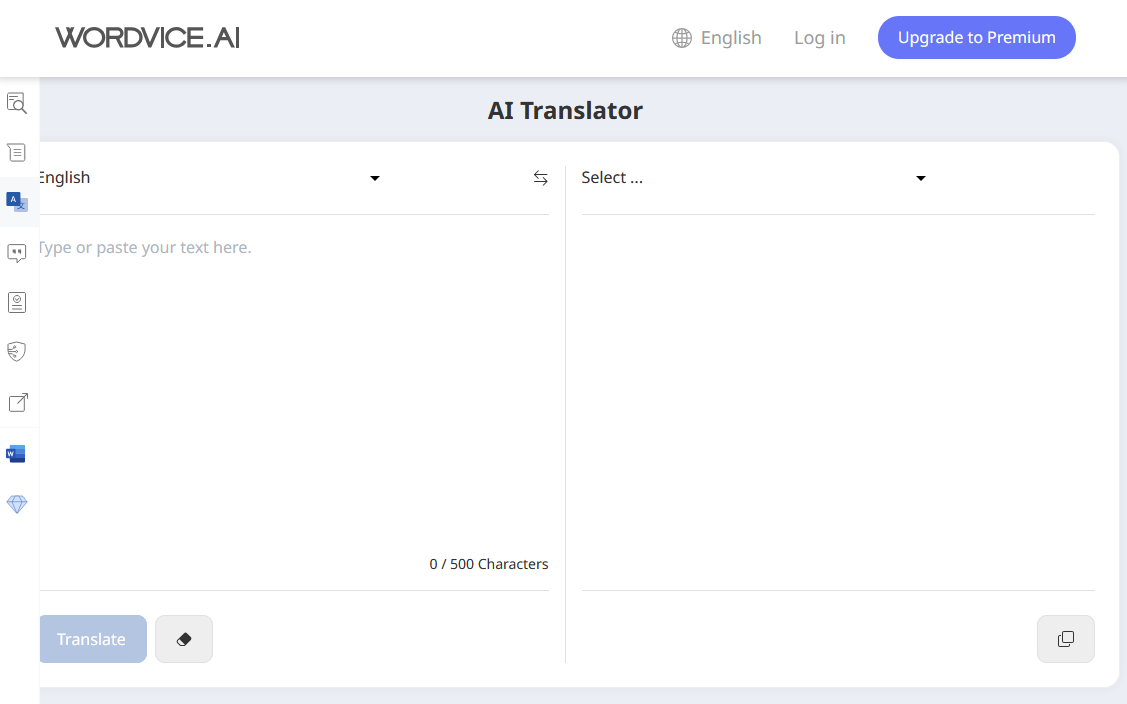


Рисунок 1.3 – Страница сервиса Wordvice

Он использует нейронные сети для перевода текста, предоставляет интеграцию с Microsoft Word и услуги обобщения и перефразирования текста при помощи искусственного интеллекта, а также поддерживает множество языков.

1.3 Выводы

При рассмотрении аналогичных решений были выделены их особенности.

Сервис DeepL обладает следующими преимуществами:

– высокое качество перевода: DeepL часто хвалят за более естественный и точный перевод по сравнению с конкурентами, особенно в контексте отдельных предложений и коротких текстов;

– поддержка многих языков: сервис предлагает широкий диапазон языков для перевода;

– удобный интерфейс: простой и интуитивно понятный интерфейс;

– быстрая обработка: перевод выполняется относительно быстро.

Сервис DeepL имеет следующие недостатки:

– ограничения по объему текста: для очень больших текстов использование данного сервиса может быть менее предпочтительным, чем Google Translate;

– стоимость: бесплатная версия имеет серьёзные ограничения на объём переводимого текста, платная подписка может быть невыгодной для недостаточно больших объемов перевода;

– меньшее количество дополнительных функций: по сравнению с Google Translate, DeepL предлагает меньше дополнительных функций, таких как транслитерация или распознавание речи.

Сервис Google Translate обладает следующими преимуществами:

– широкая доступность и поддержка языков: данный сервис поддерживает огромное количество языков и доступен практически везде;

– обработка больших объемов текста: данный сервис хорошо справляется с переводом длинных документов и объёмных web-страниц;

– множество дополнительных функций: данный сервис предлагает множество дополнительных функций, таких как транслитерация, распознавание речи, перевод изображений и т.д.;

– интеграция с другими сервисами Google: данный сервис удобно интегрируется с другими продуктами Google.

Сервис Google Translate имеет следующие недостатки:

– качество перевода может варьироваться: качество перевода может быть неравномерным, особенно для сложных текстов, технических терминов и идиом. Может быть более «дословным», чем DeepL;

– неестественность перевода: результаты перевода могут звучать неестественно или механически

Сервис Wordvice обладает следующими преимуществами:

– поддержка многих языков: как и остальные рассмотренные сервисы, Wordvice предоставляет возможность перевода на множество различных языков;

– интеграция с Microsoft Word: данный сервис предоставляет возможность перевода текста непосредственно внутри документа Word без необходимости переключения окон и копирования текста;

– минималистичный интерфейс: интерфейс данного сервиса не перегружен лишними деталями, пользователь может легко выполнять свои задачи, не отвлекаясь на лишние элементы интерфейса.

Сервис Wordvice имеет следующие недостатки:

– отсутствие возможности загружать свои документы: при необходимости перевода больших объёмов текста исходный текст неободимо копировать в текстовое поле, что снижает удобство пользования сервисом; (ПРОВЕРИТЬ)

– меньшее количество функций: как и DeepL, Wordvice имеет меньшее по сравнению с Google Translate количество функций.

Для реализации сервера web-приложения был выбран фреймворк FastAPI, позволяющий асинхронно обрабатывать запросы, обращаться к базе данных при помощи ORM Sqlalchemy и являющийся простым при разработке. Также данный фреймворк позволяет применять валидацию входных и исходящих данных при помощи Pydantic, что избавляет от необходимости самостоятельно следить за валидностью поступающих данных.

2 Проектирование web-приложения

2.1 Функциональность web-приложения

Функциональные возможности web-приложения представлены в диаграмме вариантов использования, представленной на рисунке 2.1.

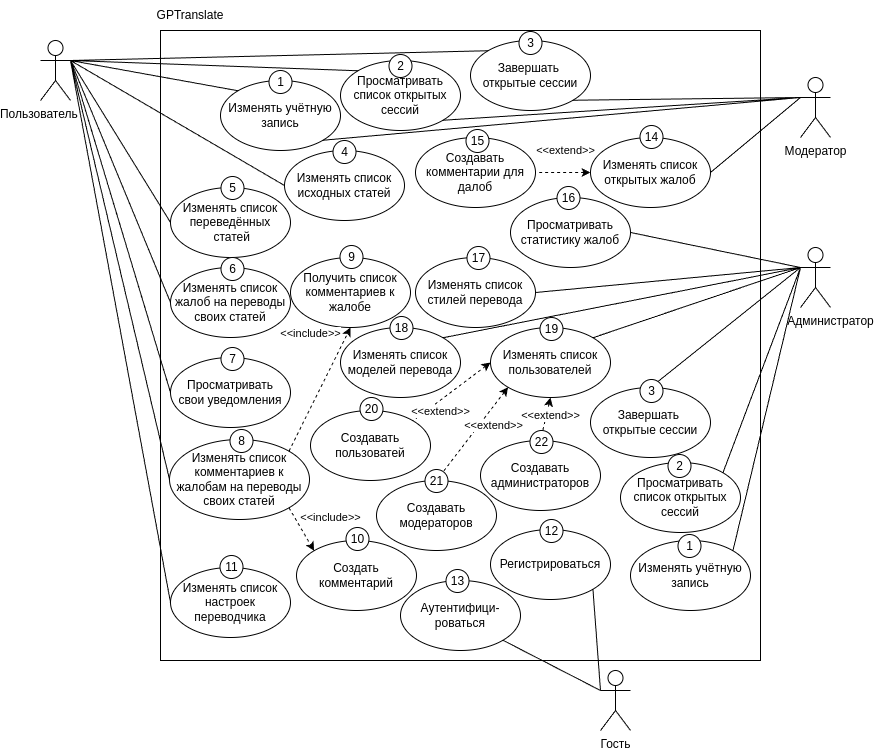


Рисунок 2.1 – Диаграмма вариантов использования web-приложения

Перечень ролей и их назначение приведены в таблице 2.1.

Таблица 2.1 – 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 |
| Администратор | Управление пользователями, запросами перевода, моделями перевода |

Функционал гостя представлен в таблице 2.2.

Таблица 2.2 – Функционал гостя

|  |  |
| --- | --- |
| 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. Функционал пользователя представлен в таблице 2.3.

Таблица 2.3 – Функционал пользователя

|  |  |
| --- | --- |
| 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 |
| 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 функционал представлен в таблице 2.4.

Таблица 2.4 – Функционал модератора

|  |  |
| --- | --- |
| 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 |

Функционал администратора представлен в таблице 2.5.

Таблица 2.5 – Функционал администратора

|  |  |
| --- | --- |
| 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 |
| 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.

2.2 Структура базы данных

Согласно схеме вариантов использования была создана база данных. Её структура представлена на рисунке 2.2.

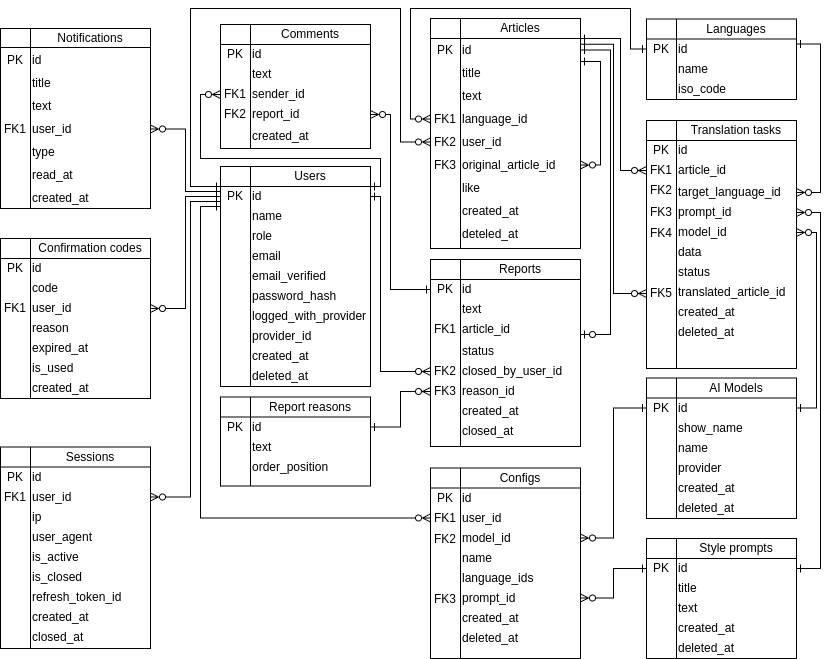


Рисунок 2.2 – Логическая схема базы данных

База данных содержит тринадцать таблиц, хранящих информацию о пользователях, сессиях, статьях и прочих данных. Типы данных были выбраны согласно документации [1]. Назначение таблиц базы данных представлено в таблице 2.6.

Таблица 2.6 – 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 |
| Languages | Хранит информацию о доступных для перевода языках (название, ISO код) |

Продолжение таблицы 2.6

|  |  |
| --- | --- |
| Articles | Хранит информацию о статьях (заголовок, 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, использующихся 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 считываются 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 2.7.

Таблица 2.7 – 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 |

Продолжение таблицы 2.7

|  |  |  |
| --- | --- | --- |
| 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 2.8.

Таблица 2.8 – Описание таблицы Sessions

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип данных | Описание |
| id | uuid | Идентификатор сессии, первичный ключ |
| 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 2.9.

Таблица 2.9 – Описание таблицы Confirmation\_codes

|  |  |  |
| --- | --- | --- |
| 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 |

Продолжение таблицы 2.9

|  |  |  |
| --- | --- | --- |
| expired\_at | timestamp without timezone | Временная отметка, после которой код будет считаться истёкшим |
| 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 2.10.

Таблица 2.10 – Описание таблицы Languages

|  |  |  |
| --- | --- | --- |
| 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 2.11.

Таблица 2.11 – Описание таблицы Articles

|  |  |  |
| --- | --- | --- |
| 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) |
| created\_at | timestamp without timezone | Дата и время создания статьи без часового пояса |
| deleted\_at | timestamp without timezone | Дата и время удаления статьи без часового пояса |

Описание столбцов таблицы Report\_reasons представлено в таблице 2.12.

Таблица 2.12 – Описание таблицы Report\_reasons

|  |  |  |
| --- | --- | --- |
| 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 2.13.

Таблица 2.13 – Описание таблицы Reports

|  |  |  |
| --- | --- | --- |
| 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 |
| created\_at | timestamp without timezone | Дата и время создания жалобы без часового пояса |
| closed\_at | timestamp without timezone | Дата и время закрытия жалобы без часового пояса |

Описание столбцов таблицы Report\_comments представлено в таблице 2.14.

Таблица 2.14 – Описание таблицы Report\_comments

|  |  |  |
| --- | --- | --- |
| 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 |

Продолжение таблицы 2.14

|  |  |  |
| --- | --- | --- |
| report\_id | uuid | Идентификатор жалобы, к которой был оставлен комментарий, внешний ключ |
| created\_at | timestamp without timezone | Дата и время создания комментария без часового пояса |

Описание столбцов таблицы Style\_prompts представлено в таблице 2.15.

Таблица 2.15 – Описание таблицы Style\_prompts

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип данных | Описание |
| id | integer | Идентификатор запроса, первичный ключ |
| title | varchar (20) | Название запроса |
| text | varchar | Текст запроса |
| created\_at | timestamp without timezone | Дата и время создания запроса без часового пояса |
| deleted\_at | timestamp without timezone | Дата и время удаления запроса без часового пояса |

Описание столбцов таблицы AI\_Models представлено в таблице 2.16.

Таблица 2.16 – Описание таблицы AI\_Models

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип данных | Описание |
| id | integer | Идентификатор модели, первичный ключ |
| show\_name | varchar (50) |  |
| 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 2.17.

Таблица 2.17 – Описание таблицы Configs

|  |  |  |
| --- | --- | --- |
| 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 |

Продолжение таблицы 2.17

|  |  |  |
| --- | --- | --- |
| 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 2.18.

Таблица 2.18 – Описание таблицы Translation\_tasks

|  |  |  |
| --- | --- | --- |
| Название столбца | Тип данных | Описание |
| id | uuid | Идентификатор задачи, первичный ключ |
| article\_id | uuid | Идентификатор исходной статьи, внешний ключ |
| target\_language\_id | integer | Идентификатор конечного языка, внешний ключ |
| prompt\_id | integer | Идентификатор запроса перевода, внешний ключ |
| model\_id | integer | Идентификатор модели перевода, внешний ключ |
| status | enum translationtaskstatus | Статус задачи (создана, в процессе выполнения, завершена успешно, завершена с ошибкой) |
| data | jsonb | Дополнительная информация о задаче (текст ошибки) |
| translated\_article\_id | uuid | Идентификатор переведённой статьи, внешний ключ |
| created\_at | timestamp without timezone | Дата и время создания задачи без часового пояса |
| deleted\_at | timestamp without timezone | Дата и время удаления задачи без часового пояса |

Описание столбцов таблицы Notifications представлено в таблице 2.19.

Таблица 2.19 – Описание таблицы Notifications

|  |  |  |
| --- | --- | --- |
| 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 2.20.

Таблица 2.20 – Назначение связей между таблицами

|  |  |
| --- | --- |
| 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 |
| Users.id- Reports.closed\_by\_user\_id | Идентификатор пользователя, закрывшего жалобу (создавшего её пользователя или любого модератора) |

Продолжение таблицы 2.20

|  |  |
| --- | --- |
| 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 |

Сценарий создания объектов базы данных на языке SQL представлен в Приложении А.

2.3 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 15.

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

Архитектура web-приложения представлена на рисунке 2.3.

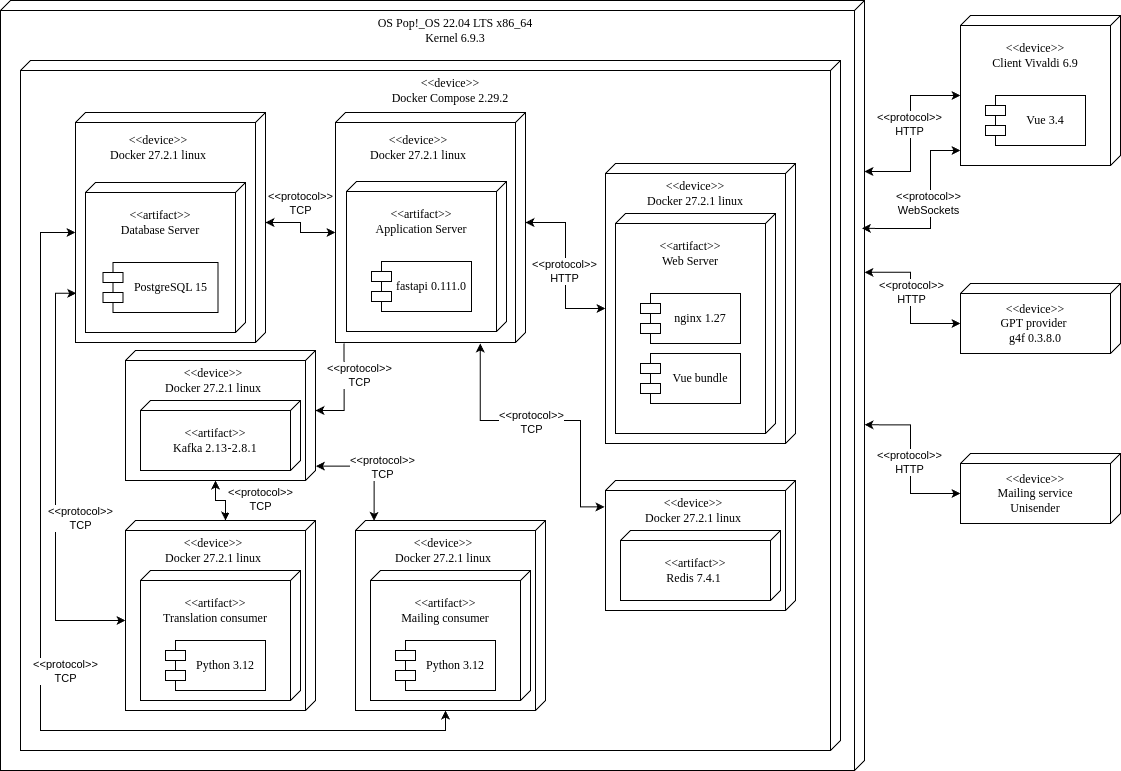


Рисунок 2.3 – Архитектура web-приложения

Пояснение назначения каждого элемента web-приложения представлено в таблице 2.21.

Таблица 2.21 – 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 |
| 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.

3 Реализация web-приложения

3.1 Обоснование выбора платформы

Для реализации web-приложения был выбран язык программирования Python и фреймворк FastAPI. FastAPI представляет собой современный веб-фреймворк для создания API на языке Python. Его выбор обусловлен несколькими факторами:

– Высокая производительность: FastAPI построен на основе асинхронного фреймворка Starlette и библиотеки типизации Pydantic, что обеспечивает высокую скорость обработки запросов. По сравнению с другими фреймворками, такими как Flask или Django, FastAPI демонстрирует значительно лучшие показатели производительности, что особенно важно для приложений с высокой нагрузкой.

– Поддержка асинхронного программирования: FastAPI изначально поддерживает асинхронные функции, что позволяет эффективно обрабатывать множество запросов асинхронно. Это свойство критично для современных веб-приложений, требующих высокой отзывчивости и минимального времени ожидания.

– автоматическая генерация документации: FastAPI автоматически создает документацию API на основе аннотаций типов, что упрощает процесс разработки и тестирования. Это позволяет разработчикам и пользователям быстро ознакомиться с доступными конечными точками и их параметрами;

– удобство разработки: Использование аннотаций типов и валидации данных на этапе разработки значительно упрощает процесс написания и поддержки кода, что способствует повышению качества программного обеспечения.

Для создания моделей, соответствующих таблицам в реляционной базе данных, была выбрана библиотека SQLAlchemy, и ее выбор также имеет свои обоснования:

– object-relational mapping: SQLAlchemy предоставляет удобный интерфейс для работы с реляционными базами данных, позволяя разработчикам использовать объектно-ориентированный подход. Это упрощает взаимодействие с базой данных и делает код более читаемым и поддерживаемым;

– гибкость и расширяемость: SQLAlchemy поддерживает множество баз данных и позволяет легко переключаться между ними. Это дает возможность разработчикам выбирать наиболее подходящую СУБД для конкретного проекта, а также упрощает миграцию данных;

– поддержка сложных запросов: SQLAlchemy предоставляет мощные инструменты для построения сложных SQL-запросов, что позволяет эффективно работать с большими объемами данных и реализовывать сложную бизнес-логику.

Для управления миграциями был выбран инструмент Alembic, и его использование в проекте также имеет свои преимущества: Alembic позволяет отслеживать изменения в структуре базы данных, что критично для поддержания целостности данных и упрощения процесса развертывания приложения, а также предоставляет возможность автоматической генерации миграций на основе изменений в моделях данных, что значительно упрощает процесс обновления базы данных и снижает вероятность ошибок. Также Alembic предоставляет возможность отката к более ранней версии БД в случае обнаружения ошибок в схеме.

3.2 Реализация web-приложения

В соответствии с диаграммой вариантов использования были реализованы функции, которые могут выполнять пользователи.

Функция “изменение учётной записи” (1) в исходном коде реализована функциями src.routers.users.views.change\_name, src.routers.users.views. request\_password\_restoration\_code и src.routers.users.views.restore\_password. Данные функции позволяют изменить отображаемое имя текущего пользователя, запросить электронное письмо со ссылкой на страницу установки нового пароля и установить новый пароль соответственно.

Функция “просмотр открытых сессий” (2) в исходном коде реализована функцией src.routers.sessions.views.get\_sessions. Данная функция возвращает список всех открытых сессий.

Функция “завершение открытых сессий” (3) в исходном коде реализована функцией src.routers.sessions.views.close\_sessions. Данная функция закрывает все открытые сессии пользователя.

Функция “изменение списка исходных статей” (4) в исходном коде реализована функциями src.routers.articles.views.upload\_article, src.routers.articles.views.update\_article, src.routers.articles.views.delete\_article. Данные функции позволяют создать новую исходную статью, обновить исходную статью и удалить исходную статью соответственно.

Функция “изменение списка переведённых статей” (5) в исходном коде реализована функциями src.routers.articles.views.delete\_article и src.routers.translation.views.create\_translation. Данные функции позволяют удалить перевод статьи и запустить перевод исходной статьи соответственно.

Функция “изменение списка жалоб на переводы своих статей” (6) в исходном коде реализована функциями src.routers.reports.views.create\_report, src.routers.reports.views.update\_report, src.routers.reports.views.update\_report\_status. Данные функции позволяют создать жалобу на перевод статьи, обновить текст и причину жалобы и закрыть жалобу соответственно.

Функция “просмотр своих уведомлений” (7) в исходном коде реализована функцией src.routers.notifications.views.get\_notifications\_list. Данная функция возвращает список непрочитанных уведомлений пользователя.

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 src.routers.reports.views.get\_comments. Данная функция возвращает список комментариев к жалобе. Функция “создание комментария” (10) в исходном коде реализована функцией src.routers.reports.views.create\_comment. Данная функция создаёт комментарий к жалобе.

Функция “изменение списка настроек переводчика” (11) в исходном коде реализована функциями src.routers.config.views.create\_config, src.routers.config.views.update\_config и src.routers.config.views.delete\_config. Данные функции позволяют создать конфигурацию, обновить конфигурацию и удалить конфигурацию соответственно.

Функция “регистрация” (12) в исходном коде реализована функцией src.routers.auth.views.register. Данная функция позволяет создать нового пользователя по имени, почте и паролю.

Функция “аутентификация” (13) в исходном коде реализована функциями src.routers.auth.views.login и src.routers.oauth.views.callback. Данные функции позволяют аутентифицироваться по логину и паролю и аутентифицировать пользователя по данным, полученным от провайдера OAuth 2.0 соответственно.

Функция “изменение списка открытых жалоб” (14) в исходном коде реализована функцией src.routers.reports.views.update\_report\_status. Данная функция позволяет удовлетворить или отклонить жалобу.

Функция “создание комментариев для жалоб” (15) в исходном коде реализована функцией src.routers.reports.views.create\_comment.

Функция “просмотр статистики жалоб” (16) в исходном коде реализована функциями src.routers.analytics.views.get\_models\_stats и src.routers.analytics.views.get\_prompts\_stats. Данные функции позволяют получить статистику жалоб по каждой модели и каждому стилю перевода.

Функция “изменение списка стилей перевода” (17) в исходном коде реализована функциями src.routers.prompts.views.create\_prompt, src.routers.prompts.views.update\_prompt и src.routers.prompts.views.delete\_prompt. Данные функции позволяют создать, обновить и удалить стиль перевода соответственно.

Функция “изменение списка моделей перевода” (18) в исходном коде реализована функциями src.routers.models.views.create\_model, src.routers.models.views.update\_model и src.routers.models.views.delete\_model. Данные функции позволяют создать, обновить и удалить модель перевода соответственно.

Функция “изменение списка пользователей” (19) в исходном коде реализована функциями src.routers.users.views.create\_user, src.routers.users.views.update\_user и src.routers.users.views.delete\_user. Данные функции позволяют создать, обновить и удалить

Функции “создание пользователей” (20), “создание модераторов” (21) и “создание администраторов” (22) в исходном коде реализована функцией src.routers.users.views.create\_user, представленной в листинге 3.17.

Полный исходный код web-приложения представлен в Приложении Б.

3.3 Реализация базы данных

Согласно логической диаграмме базы данных были созданы объекты базы данных. Модели SqlAlchemy объявлены в модуле src.database.models.py. Для изменения состояния базы данных использовался инструмент Alembic. Скрипт для создания объектов базы данных представлен в Приложении А.

4 Тестирование web-приложения

Для тестирования работоспособности web-приложения необходимо создать все необходимые объекты базы данных, а также добавить следующие записи в таблицы:

– пользователя, модератора и администратора (таблица Users, роли “user”, “moderator” и “admin” соответственно; имя, пароль и адрес электронной почты произвольные). Администратора можно создать при помощи консольной команды python src/manage.py create-admin в Docker-контейнере api;

– языки, доступные для перевода (таблица Languages). Для вставки языков доступна команда python src/manage.py insert-languages, которую нужно выполнить в Docker-контейнере api;

– модели, доступные для перевода (таблица AI\_Models). Для вставки моделей доступна команда python src/manage.py insert-models, которую нужно выполнить в Docker-контейнере api;

– стили запроса (таблица Style\_Prompts). Для вставки стилей доступна команда python src/manage.py insert-prompts, которую нужно выполнить в Docker-контейнере api;

– причины жалоб (таблица Report\_Reasons). Для вставки причин жалоб доступна команда python src/manage.py insert-report-reasons, которую нужно выполнить в Docker-контейнере api;

Для проверки функций приложения рекомендуется использовать инструмент OpenAPI. Данный инструмент генерирует документацию на основе исходного кода web-приложения. Фреймворк FastAPI включает данный инструмент, и страница документации по умолчанию доступна по адресу http://localhost:8000/api/docs. Описание тестирования функций web-приложения представлено в таблице 4.1.

Таблица 4.1 – Описание тестирования функций web-приложения

|  |  |  |
| --- | --- | --- |
| Функция web-приложения | Описание тестирования | Итог тестирования функции |
| 1 Изменение учётной записи | Аутентифицироваться в качестве пользователя (при помощи адреса электронной почты и пароля или провайдера OAuth 2.0), получить идентификатор своего пользователя при помощи GET запроса по адресу [/api/users/me/](http://localhost:8000/api/users/me/), отправить POST запрос на адрес  [/api/users/](http://localhost:8000/api/users/адрес){идентификатор своего пользователя}/name/, указав в теле запроса желаемое имя в параметре name (формат тела запроса – JSON). Сервер должен вернуть обновлённый объект пользователя в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 2 Просмотр открытых сессий | Аутентифицироваться в качестве пользователя, отправить GET запрос на адрес /api/sessions/. Сервер должен вернуть список сессий в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 3 Завершение открытых сессий | Аутентифицироваться в качестве пользователя, отправить POST запрос на адрес /api/sessions/close/. Сервер должен вернуть сообщение об успешном закрытии всех сессий | Работоспособность функции протестирована, ошибок не обнаружено |
| 4 Изменение списка исходных статей | Аутентифицироваться в качестве пользователя, отправить POST запрос на адрес /api/articles/, указав в теле запроса заголовок (title), текст (text) и идентификатор языка (language\_id) загружаемой статьи в формате JSON. Сервер должен вернуть объект статьи в формате JSON. Получить список языков в формате JSON можно, отправив GET запрос на адрес /api/languages/ | Работоспособность функции протестирована, ошибок не обнаружено |
| 5 Изменение списка переведённых статей | Аутентифицироваться в качестве пользователя, отправить POST запрос на адрес /api/translation/, указав в теле запроса идентификатор статьи, которую нужно перевести (article\_id), список идентификаторов языков, на которые нужно перевести статью (target\_language\_ids), идентификатор стиля перевода (prompt\_id) и идентификатор модели перевода (model\_id). Сервер должен вернуть сообщение о запуске перевода, через некоторое время, зависящее от объёма статьи, в таблице Notifications должна появиться запись об успешном или неуспешном переводе статьи. Списки моделей и стилей перевода можно получить по GET запросам на адреса /api/models/ и /api/prompts/ соответственно | Работоспособность функции протестирована, ошибок не обнаружено |
| 6 Изменение списка жалоб на переводы своих статей | Аутентифицироваться в качестве пользователя, отправить запрос на адрес /api/articles/{идентификатор статьи}/report/, в теле запроса указать текст жалобы (text) и идентификатор причины жалобы (reason\_id). Сервер должен вернуть объект жалобы в формате JSON. Список доступных причин жалоб можно получить при помощи GET запроса на адрес /api/report-reasons/ | Работоспособность функции протестирована, ошибок не обнаружено |
| 7 Просмотр своих уведомлений | Аутентифицироваться в качестве пользователя, отправить GET запрос на адрес /api/notifications/. Сервер должен вернуть список непрочитанных уведомлений в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 8 Изменение списка комментариев к жалобам на переводы своих статей | Протестировать функцию 10 Создание комментария, затем функцию 9 Получение списка комментариев к жалобе | Работоспособность функции протестирована, ошибок не обнаружено |
| 9 Получение списка комментариев к жалобе | Аутентифицироваться в качестве пользователя, отправить GET запрос на адрес /api/articles/ {идентификатор переведённой статьи, для жалобы на которую требуется получить список комментариев} /report/comments/. Сервер должен вернуть список комментариев в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 10 Создание комментария | Аутентифицироваться в качестве пользователя, отправить POST запрос на адрес /api/articles/ {идентификатор переведённой статьи, для жалобы на которую требуется создать комментарий} /report/comments/, в запросе указать текст комментария (text). Сервер должен вернуть объект комментария в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 11 Изменение списка настроек переводчика | Аутентифицироваться в качестве пользователя, отправить POST запрос на адрес /api/configs/, в запросе указать название конфигурации (name), идентификатор стиля перевода (prompt\_id), идентификатор модели перевода (model\_id) и список конечных языков (language\_ids). Сервер должен вернуть объект конфигурации в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 12 Регистрация | Отправить POST запрос на адрес /api/auth/register/, указав в теле запроса имя пользователя (name), адрес электронной почты (email) и пароль (password). Сервер должен вернуть сообщение об успешной регистрации | Работоспособность функции протестирована, ошибок не обнаружено |
| 13 Аутентификация | Отправить POST запрос на адрес /api/auth/login/, указав в теле запроса адрес электронной почты (email) и пароль (password). Сервер должен вернуть сообщение об успешной аутентификации | Работоспособность функции протестирована, ошибок не обнаружено |
| 14 Изменение списка открытых жалоб | Аутентифицироваться в качестве модератора, отправить на адрес /api/articles/ {идентификатор статьи, жалобу на которую нужно изменить} /report/status/ POST запрос, указав в параметрах запроса новый статус жалобы (Отклонена или Удовлетворена). Сервер должен вернуть объект жалобы в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 15 Создание комментариев для жалоб | Аутентифицироваться в качестве модератора, отправить на адрес /api/articles/ { идентификатор статьи, для жалобы на которую нужно создать комментарий} /report/comments/ POST запрос, указав в теле запроса текст комментария (text). Сервер должен вернуть объект комментария в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 16 Просмотр статистики жалоб | Аутентифицироваться в качестве администратора, отправить GET запрос на адрес /api/analytics/models-stats/. Сервер должен вернуть данные по жалобам для каждой модели перевода в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 17 Изменение списка стилей перевода | Аутентифицироваться в качестве администратора, отправить POST запрос на адрес /api/prompts/, в теле запроса указать название (title) и текст (text) стиля перевода Сервер должен вернуть объект стиля перевода в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 18 Изменение списка моделей перевода | Аутентифицироваться в качестве администратора, отправить POST запрос на адрес /api/models/, в теле запроса указать отображаемое название (show\_name), название (name) и провайдер (provider) модели перевода Сервер должен вернуть объект модели перевода в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 19 Изменение списка пользователей | Аутентифицироваться в качестве администратора, отправить POST запрос на адрес /api/users/, в теле запроса указать имя (name), адрес электронной почты (email), флаг, указывающий, подтверждена ли почта (email\_verified), роль (role) и пароль (password) пользователя. Сервер должен вернуть объект созданного пользователя в формате JSON | Работоспособность функции протестирована, ошибок не обнаружено |
| 20 Создание пользователей | Аналогично тестированию функции 19 Изменение списка пользователей, но роль в теле запроса должна быть “Пользователь” | Работоспособность функции протестирована, ошибок не обнаружено |
| 21 Создание модераторов | Аналогично тестированию функции 19 Изменение списка пользователей, но роль в теле запроса должна быть “Модератор” | Работоспособность функции протестирована, ошибок не обнаружено |
| 22 Создание администраторов | Аналогично тестированию функции 19 Изменение списка пользователей, но роль в теле запроса должна быть “Администратор” | Работоспособность функции протестирована, ошибок не обнаружено |

Таким образом, все заявленные функции были протестированы, и была проверена работоспособность web-приложения.

5 Руководство программиста

Для развёртывания web-приложения применяется инструмент Docker Compose. Перед развёртыванием web-приложения необходимо убедиться, что в системе установлены Docker Engine и Docker Compose при помощи команд docker version и docker compose version. В случае, если любая из указанных технологий не установлена, её необходимо установить согласно подходящей инструкции на официальном сайте, например, [2] для Docker Engine и [3] для Docker Compose. Далее необходимо создать пустую папку, в которую нужно поместить файлы, содержимое которых представлено в Приложении Б и Приложении В. Затем в данной папке необходимо создать файл .env, в котором указать необходимые значения переменных окружения, используемых web-приложением, таких как ключ доступа Unisender, логин и пароль для доступа к базе данных и так далее. Пример объявления переменных окружения находится в файле .example.env. Далее в корневой папке web-приложения необходимо последовательно выполнить команды “docker network create a”, “docker build -t diploma-base -f contrib/docker/base/Dockerfile .” и “ docker compose --env-file=.env -f contrib/docker/docker-compose.yaml up -d --build ”. Эти команды создадут новую сеть Docker, соберут базовый образ для контейнеров из исходного кода и запустят все необходимые контейнеры соответственно. В папке contrib/persistent\_data находятся .json файлы с данными, которыми будет заполнена база данных по умолчанию. Файлы включают данные о языках, моделях, стилях перевода и причинах жалоб. При каждом запуске контейнера api будет производиться проверка на наличие данных, которых нет в базе данных, и отсутствующие строки будут добавлены автоматически. Это позволяет добавлять новые строки в базу данных, не прибегая к использованию SQL запросов. Также при запуске контейнера api автоматически создаётся администратор с адресом электронной почты admin@d.com и паролем string, а также производится обновление структуры базы данных в соответствии с файлами миграций, находящихся в папке src/database/alembic/versions.

Для корректной работы web-приложения ему необходим доступ к внешнему сервису g4f. Он может находиться в любом удобном месте: на локальной машине, удалённом сервере и так далее. Для большего удобства можно развернуть его в Docker и добавить в ранее созданную сеть. Для этого нужно скачать базовый образ при помощи команды docker pull hlohaus789/g4f:0.3.9.5, развернуть его при помощи команды docker run --detach --name g4f hlohaus789/g4f:0.3.9.5, добавить созданный контейнер в ранее созданную сеть при помощи команды docker network connect a g4f. В файл .env нужно установить переменную окружения G4F\_ADDRESS=http://g4f:1337. Проверить доступность сервиса можно при помощи команды docker exec -t docker-api-1 bash -c "/app/contrib/docker/wait-for-it.sh \"g4f:1337\" -t 30 -- echo \"Сервис доступен\"". Для применения изменений в файле .env развёртывание web-приложения при помощи команды docker compose up необходимо повторить.

После развёртывания we-приложения по адресу <http://localhost:8000> будет доступна web-страница приложения. Шаги по проверке работоспособности развёрнутого сервиса описаны в разделе 4.

Заключение

При выполнении курсового проекта было создано приложение поиска и предложения работы. Сервер был создан при помощи платформы FastAPI, языка программирования Python и . Web-сайт был реализован при помощи библиотеки React. База данных была реализована в СУБД PostgreSQL. Были реализованы все функциональные требования, а именно:

– обеспечение возможности регистрации и авторизации;

– поддержка ролей гостя, соискателя, работодателя и администратора;

– обеспечение возможности изменять образование, опыт работы и список ключевых навыков соискателя;

– предоставление возможности работодателю принимать и отклонять отклики;

− обеспечение возможности оставлять отзывы об исполнителе и работодателе;

− предоставление возможности отслеживать статус предложения о работе в реальном времени;

− обеспечение возможности соискателю откликаться на предложения о работе;

− предоставление возможности создавать и удалять учётные записи;

− обеспечение возможности фильтровать предложения по критериям (тип, необходимый опыт, оплата и так далее);

− предоставление возможности размещать и удалять вакансии и заказы на выполнение работ.

Также были реализованы WebSocket-сервер и поддержка протокола HTTPS. Приложение было протестировано на наличие ошибок с использованием ручного и автоматического тестирования. Для ручного тестирования использовался браузер и Postman. Для автоматического тестирования использовалось вспомогательное приложение, использующее функцию fetch.

По итогам тестирования были исправлены следующие ошибки в приложении: неверный формат отправляемых данных, отсутствие необходимых данных в ответе, ошибки в проверке авторизации.

Также было создано иллюстрированное руководство пользователя, в котором были описаны способы взаимодействия с приложением и наглядно продемонстрированы web-страницы приложения.

Список используемых источников

1. PostgreSQL Documentation: 15: Chapter 8. Data Types [Электронный ресурс]. – Электронные данные. – Режим доступа: https://www.postgresql.org/docs/15/datatype.html;
2. Install | Docker Docs [Электронный ресурс]. – Электронные данные. – Режим доступа: https://docs.docker.com/engine/install/;
3. Install | Docker Docs [Электронный ресурс]. – Электронные данные. – Режим доступа: https://docs.docker.com/compose/install/.

Приложение А

Скрипт создания объектов базы данных

Приложение Б

Исходный код web-приложения

Приложение В

Содержимое файлов, необходимых для развёртывания web-приложения