Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа № 5.

«Объектно-ориентированное моделирование. Структурные UML-диаграммы»

Студент: Точило О. В.

ФИТ 4 курс 4 группа

Преподаватель: Якубенко К. Д.

Минск 2024

# Постановка задачи

Темой данной лабораторной работы является построение UML-диаграммы вариантов использования, необходимое для графического представления взаимосвязи ролей конечных пользователей и вариантов взаимодействий с системой, представленной в первой лабораторной работе – web-приложением переводчика статей «GPTranslate».

Целью лабораторной работы является изучение методологии объектно-ориентированного моделирования средствами UML (Unified Modeling Language), представляющего собой мощный инструмент для визуализации, спецификации, построения и документирования артефактов программного обеспечения, ознакомление с основными принципами объектно-ориентированного проектирования программного обеспечения и получение навыков проектирования функциональности информационной системы с применением UML. Создание диаграмм вариантов использования даст возможность визуализировать сценарии взаимодействия пользователей с системой, что особенно важно для выявления потенциальных проблем и улучшения пользовательского опыта. Каждая диаграмма должна четко отображать акторов (пользователей или внешние системы), их цели и последовательность действий (сценарии использования) при взаимодействии с системой. Это позволит более глубоко понять потребности конечных пользователей и сформировать четкие требования к функционалу системы.

Кроме того, данный подход способствует более качественной разработке системы, позволяя команде разработчиков заранее предугадать, какие функции будут наиболее важны для пользователей, и соответственно, сосредоточиться на их реализации. Процесс построения диаграмм вариантов использования также помогает в выявлении возможных сценариев, которые могут не учитываться изначально, тем самым минимизируя риски во время разработки.

Также стоит отметить, что использование UML в проектировании позволяет создать общую визуальную основу для всех участников проекта, включая разработчиков, тестировщиков и бизнес-аналитиков. Это облегчает коммуникацию между ними, так как все могут опираться на одну и ту же модель системы. В итоге, успешное выполнение данной лабораторной работы не только улучшит навыки работы с UML, но и поможет в дальнейшем развитии системы, обеспечивая её гибкость и адаптивность к изменяющимся условиям.

Использование UML в целом, и диаграмм вариантов использования в частности, способствует созданию единой модели системы, которая служит «единым языком» для всех участников проекта. Это значительно повышает эффективность коммуникации, минимизирует недопонимания и риски, связанные с несогласованностью требований и реализации.

# Описание функциональных требований

Функциональные требования к системе можно разделить на требования к функционалу для различных ролей приложения: пользователя, гостя, модератора и администратора.

Функционал для пользователя:

* загрузка исходной статьи: возможность вставки текста напрямую или загрузки файла с помощью встроенного редактора или файлового менеджера;
* выполнение перевода статьи: автоматический запрос системы на выполнение перевода, с возможностью выбора языка оригинала и целевого языка;
* оценка перевода статьи: шкала оценки с возможностью оставлять комментарии;
* создание жалобы на перевод статьи: простой и удобный интерфейс для описания проблем с переводом, которые должны быть рассмотрены модератором;
* редактирование личной информации: поле для ввода email адреса, кнопка для смены пароля, возможность изменения отображаемого имени.

Функционал для модератора:

* просмотр открытых жалоб: список актуальных проблем с переводами, отсортированный по времени поступления;
* удовлетворение и отклонение жалоб: простая кнопочная система для принятия решения, с возможностью добавления комментариев к жалобе.

Функционал для администратора:

* просмотр и редактирование списка всех пользователей: таблица с информацией о пользователях, возможность фильтрации по ролям и сортировки по различным параметрам;
* просмотр и редактирование списка моделей перевода: панель управления языками, где можно добавлять новые языки, изменять названия или удалять ненужные языки;
* просмотр и редактирование запросов перевода: список всех текущих запросов, с возможностью фильтрации по статусу и дате создания.

Функционал для гостя:

* регистрация: простая форма регистрации с обязательным указанием email адреса и пароля;
* аутентификация: получение доступа к возможностям зарегистрированной ранее учётной записи по адресу email и паролю.

# Описание программных средств

Draw.io (ранее известный как diagrams.net) - это мощный и многофункциональный онлайн-инструмент для создания диаграмм и схем. Разработанный компанией JGraph Ltd., этот веб-ресурс предоставляет широкий спектр инструментов для визуального проектирования и представления концепций.

Основные характеристики и возможности Draw.io:

* гибкость и универсальность: Draw.io предлагает возможности работы с многими типами диаграмм, включая функциональные диаграммы (IDEF0, UML, BPMN), диаграммы потока, смысловые карты, диаграммы Венна, архитектурные диаграммы, мозговые штурмы, канбан-доски, схемы баз данных, элементы управления проектами;
* интерфейс и удобство использования: простой и интуитивно понятный интерфейс, возможность быстрого создания диаграмм с помощью готовых шаблонов, поддержка многоканального редактирования (включая мобильные устройства);
* совместимость и интеграция: поддержка импорта и экспорта различных форматов файлов (.vsdx, .gliffy, .lucidchart), возможность интеграции с популярными инструментами для совместной работы (Google Drive, Dropbox, OneDrive);
* расширенные функции: комментарии и заметки к диаграммам, возможность создания и управления библиотеками элементов;
* безопасность и конфиденциальность: защита проектов за счет шифрования, возможность настройки прав доступа к диаграммам;
* обучение и ресурсы: встроенная справка и учебные материалы, база знаний с примерами и инструкциями;
* поддержка и развитие: регулярные обновления с новыми функциями и улучшениями, активное сообщество пользователей и форумы для обсуждения.

Применение Draw.io в проекте GPTranslate:

* создание подробных функциональных диаграмм для каждого модуля системы;
* разработка архитектуры системы, показывающей взаимосвязь между различными компонентами;
* создание схемы потока данных, демонстрирующей, как информация передается через систему;
* создание диаграммы пользовательского интерфейса, показывающей все доступные функции и их взаимодействие;
* разработка диаграммы безопасности, отображающей меры защиты данных и пользователей.

Также для создания диаграммы классов использовался инструмент PlantUML, который позволяет создавать большинство видов UML-диаграмм по текстовому описанию.

# Описание практического задания

В ходе выполнения практического задания необходимо построить диаграмму классов. Создание этой диаграммы является важным этапом в проектировании информационных систем, так как оно помогает структурировать данные и связи между различными компонентами. С помощью UML-диаграммы можно наглядно представить структуру системы, что способствует более ясному пониманию бизнес-логики, упрощает процесс разработки и улучшает коммуникацию между участниками проекта.

Диаграмма классов описывает систему управления статьями, переводами и связанной обратной связью. Система включает следующие ключевые компоненты:

* Пользователь (User): Хранит информацию о пользователях, включая уникальный идентификатор (id), имя, email, хэш пароля (password\_hash) и роль (role). Пользователи могут создавать статьи (Article), создавать, изменять и закрывать жалобы (Report), отправлять комментарии (Comment), управлять конфигурациями перевода (TranslationConfig) и получать уведомления (Notification).
* статья (Article): Представляет статью с уникальным идентификатором (id), заголовком, текстом, идентификатором пользователя-создателя (user\_id), языком статьи (language) и, опционально, идентификатором исходной статьи, если это перевод (original\_article\_id). Каждая статья может иметь одну связанную жалобу (Report) или не иметь жалобы;
* жалоба (Report): Хранит жалобы на переводы исходных статей. Включает уникальный идентификатор (id), текст жалобы (text), идентификатор статьи, к которой относится жалоба (article\_id), текущий статус (status), идентификатор пользователя, закрывшего жалобу (closed\_by\_user\_id), и причину жалобы (reason). Жалобы могут иметь несколько связанных комментариев (Comment);
* комментарий (Comment). Хранит комментарии к жалобам. Включает уникальный идентификатор (id), текст комментария (text), идентификатор пользователя, отправившего комментарий (sender\_id), идентификатор связанной жалобы (report\_id) и метку времени создания (created\_at);
* запрос перевода (StylePrompt). Хранит запросы, отправляемые к моделям перевода. Имеет уникальный идентификатор (id), заголовок и текст запроса;
* модель перевода (AIModel). Представляет различные модели перевода, доступные для перевода, каждая с уникальным идентификатором (id), именем и поставщиком;
* конфигурация переводчика (TranslationConfig). Позволяет пользователям настраивать параметры для задач перевода. Включает уникальный идентификатор (id), идентификатор владельца-пользователя (user\_id), выбранный запрос стиля (prompt\_id), название, список поддерживаемых языков (languages) и выбранную модель перевода (model\_id);
* уведомление (Notification). Хранит уведомления, отправляемые пользователям. Каждое уведомление имеет уникальный идентификатор (id), заголовок, текстовое сообщение, идентификатор получателя (user\_id) и метку времени прочтения (read\_at).

Диграмма классов представлена на рисунке 4.1.

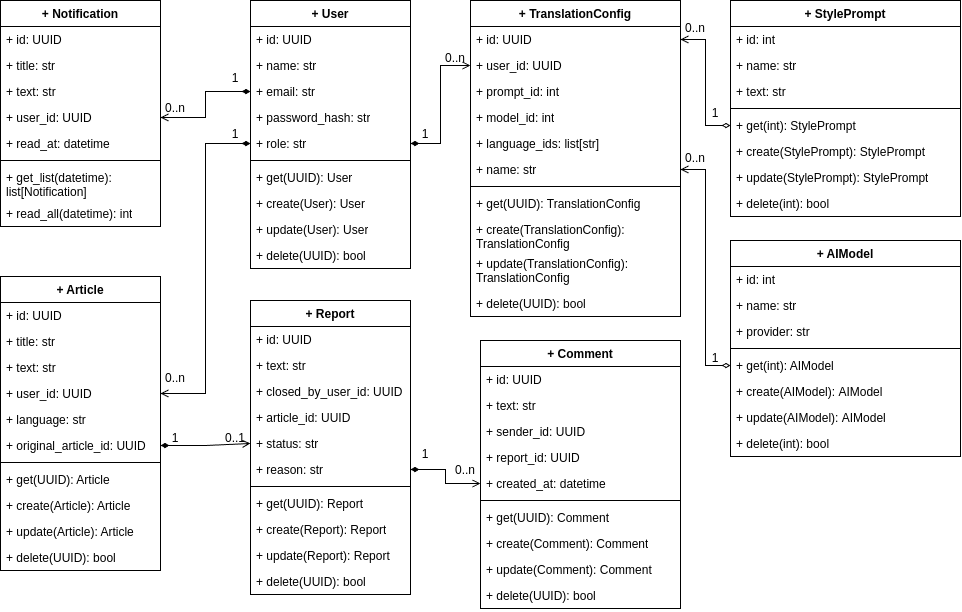


Рисунок 4.1 – Диаграмма классов

В данном проекте используется многослойная архитектура, которая структурирует систему на несколько независимых уровней, каждый из которых отвечает за свои задачи и функции. Диаграмма пакетов с описанием используемых слоев представлена на рисунке 4.2.

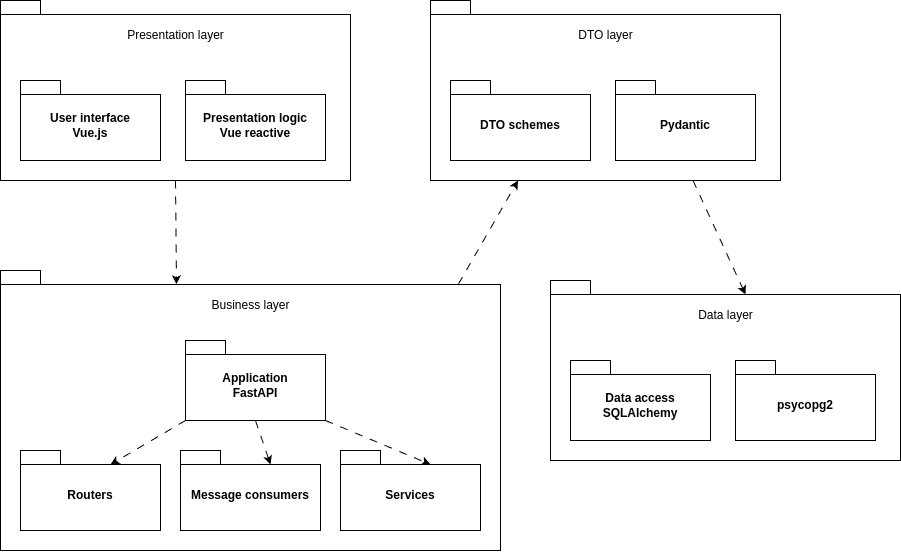
****

Рисунок 4.2 – Диаграмма пакетов

Такая архитектура позволяет разделить ответственность между слоями, улучшая поддержку и развитие приложения, а также обеспечивая гибкость и масштабируемость системы. На диаграмме пакетов представлены основные слои системы, начиная от пользовательского интерфейса до доступа к данным.

# Теоретический материал

Перечислите структурные диаграммы, которые входят в UML 2.0.

Новая версия UML 2.0, разработанная ассоциацией OMG в 2005 г., отличается от UML 1.0 большей степенью формализации в связи с необходимостью автоматизации перевода описания на UML в машинный код.

Все расширения языка основаны на получении новых модулей путем вложения базовых модулей более низкого уровня в создаваемые модули более высоких уровней.

В UML 2.0 введено 13 типов диаграмм, разделенных на два подмножества — структурных и поведенческих диаграмм.

Структурные модели, известные также как статические модели, описывают структуру сущностей или компонентов некоторой системы, включая их классы, интерфейсы, атрибуты и отношения. К структурным относятся следующие диаграммы:

* диаграммы пакетов или контейнеров (Package diagrams);
* диаграммы классов (Class or Structural diagrams);
* диаграммы объектов (Object diagrams);
* композитные диаграммы (Composite Structure);
* диаграммы компонентов (Component diagrams), показывающие образование структур из классов и интерфейсы между структурами;
* диаграммы развертывания (Deployment diagrams).

Укажите назначение структурных диаграммы.

Основное назначение структурных диаграмм заключается в графическом представлении состава статистических совокупностей, характеризующихся как соотношение различных частей каждой из совокупностей. Состав статистической совокупности графически может быть представлен с помощью как абсолютных, так и относительных показателей.

Опишите нотации, которые используются для построения Classes диаграмм.

Существует несколько обозначений диаграмм классов, которые используются при рисовании диаграмм классов UML. Мы перечислили ниже наиболее распространенные нотации диаграммы классов.

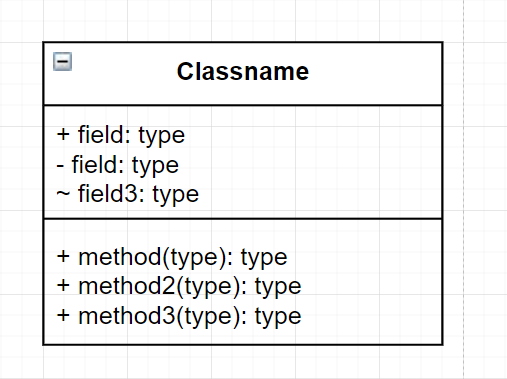


Рисунок 5.1 – Изображение класса

Классы представляют собой центральные объекты в системе. Он представлен прямоугольником с 3 отсеками.

Первый показывает имя класса, а средний – атрибуты класса, которые являются характеристиками объектов. В нижнем списке перечислены операции класса, которые представляют собой поведение класса.

Последние два отсека являются необязательными. Нотация класса без последних двух отделений называется простым классом и содержит только имя класса.

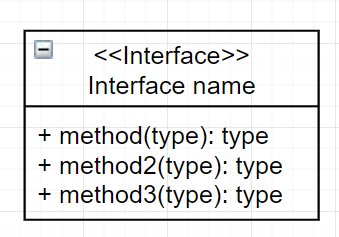


Рисунок 5.2 – Изображение интерфейса

Символ интерфейса на диаграммах классов обозначает набор операций, которые детализируют ответственность класса.

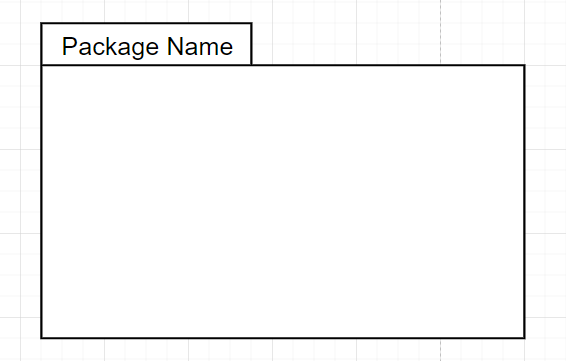


Рисунок 5.3 – Изображение пакета

Символ пакета – это элемент, который широко используется в моделировании объектно-ориентированных систем, в частности, при работе с UML-диаграммами. Он служит для группировки классов, интерфейсов и других элементов, которые либо имеют сходные свойства и поведение, либо тесно взаимосвязаны. Основной целью использования символа пакета является упрощение восприятия диаграмм и повышение структурности представленных на них элементов, что делает их более организованными и понятными для читателя. Визуально символ пакета часто изображается как прямоугольник с небольшой закладкой в верхнем углу, что напоминает папку или документ.

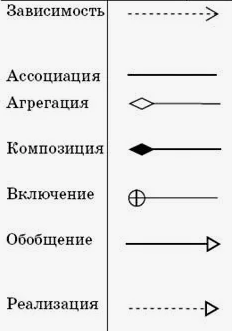


Рисунок 5.4 – Отношения в диаграмме классов

Для чего применяются расширения диаграмм UML?

Механизмы расширения UML включают: стереотипы (stereotype) – расширяют словарь UML, позволяя на основе существующих элементов языка создавать новые, ориентированные для решения конкретной проблемы; помеченные значения (tagged value) – расширяют свойства основных конструкций UML, позволяя включать дополнительную информацию в спецификацию элемента.

Что означают понятия «стереотип»

Стереотип - класс, связанный с "метамоделью", что означает стандартные элементы UML, определенные в стандарте UML. Таким образом, это класс классов (или класс ассоциаций, или любого другого элемента UML).

В UML 2.0 структурные диаграммы играют ключевую роль в визуализации архитектуры системы и её элементов. Они помогают описывать статические аспекты системы, такие как её классы, компоненты и связи между ними. Каждая диаграмма имеет своё назначение и используется для представления определённых аспектов системы, таких как классы или компоненты.

Нотации, применяемые для построения диаграмм, служат для стандартного обозначения различных элементов, упрощая их восприятие и взаимодействие. Расширения UML, такие как стереотипы и тегированные значения, добавляют гибкость к модели, позволяя адаптировать стандартные элементы под специфические задачи. Эти механизмы расширения обеспечивают возможность расширять и дополнять язык UML, чтобы он лучше отражал специфические требования и характеристики разрабатываемой системы.

# Заключение

В ходе выполнения лабораторной работы была разработана UML-диаграмма классов для web-приложения переводчика статей «GPTranslate». Эта диаграмма позволила наглядно представить ключевые сущности системы, такие как пользователь, статья, жалоба, уведомление, конфигурация переводчика, стиль и модель запроса, а также их взаимосвязи и взаимодействие.

Построение диаграммы классов способствовало детальному анализу взаимодействий в системе, что позволило понять, как пользователи могут осуществлять регистрацию, загружать свои статьи, управлять конфигурациями переводчика, создавать переводы статей, жалобы на переводы и комментарии к жалобам. Связи между сущностями, такие как ассоциации между пользователями, статьями, жалобами и комментариями, агрегация жалоб на переводы и композиция комментариев к жалобам, обеспечили ясность в проектировании архитектуры системы.

Диаграмма помогла также выделить ключевые методы, такие как загрузка и удаление статьи, управление конфигурациями переводчика, а также представила механизм работы с жалобами, что крайне важно для обратной связи и повышения качества предоставляемых услуг. Выявление основных сущностей и их связей позволило уточнить функциональные требования и разработать интерфейсы для работы с каждым компонентом системы.

Методы, выделенные при помощи диаграммы, являются основополагающими для функционирования системы, обеспечивая ее основной функционал и удобство использования для конечных пользователей. Кроме того, диаграмма представила механизм работы с запросами перевода и моделями перевода, что крайне важно для обратной связи с клиентами. Отзывы клиентов играют важную роль в процессе улучшения предлагаемых услуг, так как они помогают выявлять слабые места и возможности для дальнейшего развития.

Таким образом, выполнение данной лабораторной работы позволило не только закрепить навыки объектно-ориентированного моделирования, но и углубить понимание построения сложных систем с помощью UML. Полученная диаграмма классов является основой для дальнейшей разработки системы, помогая в структурировании функционала, оптимизации взаимодействия компонентов и улучшении пользовательского опыта. Более того, данная модель поможет выявить потенциальные узкие места системы на ранних этапах разработки, что снизит риски в процессе ее реализации.