Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа № 6.

«Объектно-ориентированное моделирование. Физические диаграммы UML»

Студент: Точило О. В.

ФИТ 4 курс 4 группа

Преподаватель: Якубенко К. Д.

Минск 2024

# Постановка задачи

Ознакомление с основными принципами объектно-ориентированного проектирования программного обеспечения и получение навыков проектирования архитектуры информационной системы с применением методологии UML. В рамках работы также будет разработана диаграмма развертывания и компонентов для психологического центра.

Объектно-ориентированное моделирование (ООМ) – это парадигма проектирования, которая позволяет представить сложную систему как совокупность взаимодействующих объектов. Каждый объект обладает собственным состоянием (данными, атрибутами) и поведением (действиями, методами), что делает модель более интуитивно понятной и легко модифицируемой по сравнению с процедурным подходом. UML (Unified Modeling Language) выступает в роли универсального языка для визуализации и спецификации этих объектов и их взаимодействий, обеспечивая стандартный и понятный способ документирования проекта для всех участников разработки – от аналитиков и дизайнеров до программистов и тестировщиков. UML не ограничивается одним типом диаграмм; он предоставляет богатый набор инструментов, каждый из которых предназначен для отображения различных аспектов системы.

Основные принципы объектно-ориентированного проектирования включают инкапсуляцию, которая скрывает внутреннее состояние объекта и предоставляет лишь необходимый интерфейс для взаимодействия, а также наследование, позволяющее объектам наследовать свойства и методы от других объектов, что способствует повторному использованию кода. Полиморфизм предоставляет возможность обработки объектов разных классов одинаково, что упрощает взаимодействие в системе, а абстракция позволяет выделять ключевые характеристики объекта, упрощая модель системы.

В рамках данного проекта будет создана архитектура web-приложения переводчика статей «GPTranslate», что позволит повысить качество переводов и удовлетворение пользователей смысловым наполнением переведённых текстов. Диаграмма развертывания покажет физическую структуру системы, включая сервер базы данных, веб-сервер и клиентские устройства, в то время как диаграмма компонентов иллюстрирует взаимодействие между модулями, такими как управление пользователями и статьями.

Таким образом, изучение методологии UML и принципов объектно-ориентированного проектирования позволяет создавать гибкие и поддерживаемые решения, что значительно улучшит работу переводчика и повысит уровень удовлетворённости клиентов.

# 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.

# Описание практического задания

В рамках практического задания была разработана диаграмма развертывания, отражающая архитектуру приложения, построенного с использованием Docker, Docker compose и современных технологий, представлено на рисунке 4.1.

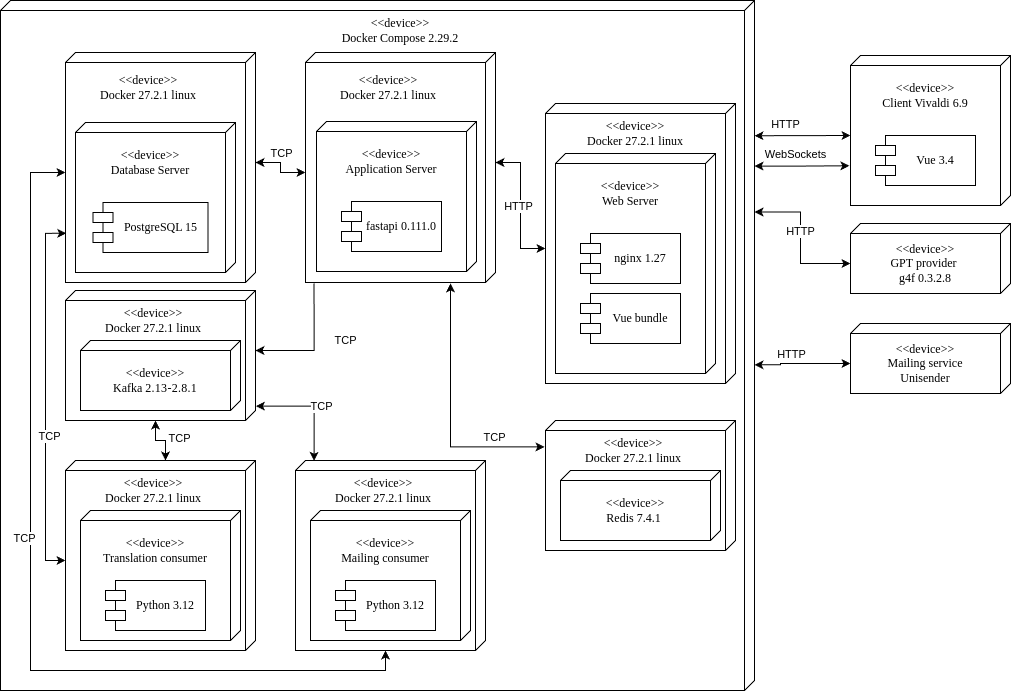


Рисунок 4.1 – Диаграмма развёртывания

Первый контейнер содержит приложение, созданное при помощи fastapi версии 0.111.0. Этот контейнер использует протокол TCP для взаимодействия с другими компонентами системы. На диаграмме также представлен контейнер, который включает в себя nginx версии v1.27 и Vue bundle версии 3.4. Он взаимодействует с приложением по протоколу HTTP, что обеспечивает обмен данными и запросами между компонентами. Кроме того, диаграмма включает контейнер с базой данных PostgreSQL версии 15, которая хранит данные, необходимые для функционирования приложения. Это позволяет обеспечить надежное хранение информации и доступ к ней в процессе работы системы. В отдельной части диаграммы отображен браузер Vivaldi версии 6.9, включающий фреймворк Vue.js v3.4. Браузер взаимодействует с контейнером устройства по протоколу HTTP, предоставляя пользователю доступ к web-интерфейсу приложения и позволяя ему взаимодействовать с системой. Также на диаграмме присутствуют вспомогательные контейнеры: брокер сообщений Kafka, потребители сообщений и база данных Redis.

Данная диаграмма развертывания наглядно демонстрирует, как организованы различные компоненты приложения и каким образом они взаимодействуют друг с другом. Это визуальное представление упрощает понимание структуры системы, что является важным для разработки, внедрения и дальнейшего сопровождения приложения.

# Теоретический материал

Укажите назначение физических диаграмм: компонентов и развертывания.

Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код. Диаграмма развертывания содержит графические изображения процессоров, устройств, процессов и связей между ними.

Дайте определение понятиям: узел, артефакт, интерфейс.

Узел (node) представляет собой некоторый физически существующий элемент системы, обладающий некоторым вычислительным ресурсом. Интерфейс служит для описания атрибутов и операций, которые должен реализовать компонент. Артефакт – некая физическая сущность, программный компонент, который используется или создаётся во время работы программного обеспечения

Опишите нотации, которые используются для представления компонентов (их вариации).

Компоненты развертывания, которые обеспечивают непосредственное выполнение системой своих функций: динамически подключаемые библиотеки с расширением dll, Web-страницы на языке разметки гипертекста с расширением html и файлы справки с расширением hlp. Компоненты-рабочие продукты: файлы с исходными текстами программ, например, с расширениями h или срр для языка C++. Компоненты исполнения, представляющие исполнимые модули – файлы с расширением ехе. Интерфейс служит для описания атрибутов и операций, которые должен реализовать компонент.

Опишите основные нотации, которые используются для представления архитектуры системы в виде диаграммы развертывания.

Узел (node) представляет собой некоторый физически существующий элемент системы, обладающий некоторым вычислительным ресурсом. Соединения указывают отношения между узлами и являются разновидностью ассоциации.

Укажите основные виды связей между компонентами и между узлами.

Соединения являются разновидностью ассоциации и изображаются отрезками линий без стрелок. Наличие такой линии указывает на необходимость организации физического канала для обмена информацией между соответствующими узлами.

Физические диаграммы, такие как диаграммы компонентов и развертывания, играют ключевую роль в описании архитектуры системы. Диаграмма компонентов помогает определить, как программные компоненты (исходный код, библиотеки, исполняемые файлы) взаимодействуют и зависят друг от друга. Диаграмма развертывания, в свою очередь, визуализирует физические элементы системы, такие как узлы, устройства и процессы, и их связи, определяя, как программные компоненты распределяются по вычислительным ресурсам. Основные понятия, такие как узел, артефакт и интерфейс, описывают физические и логические части системы: узлы — это физические устройства, артефакты — это программные компоненты, а интерфейсы служат для описания взаимодействий между компонентами. Связи между компонентами и узлами показывают обмен информацией и требуют физической инфраструктуры для этого.

# Заключение

В ходе выполнения практического задания была детально изучена методология объектно-ориентированного моделирования с использованием нотаций UML, а также разработана диаграмма развертывания для приложения, построенного на базе Docker. Эта работа позволила не только ознакомиться с основными принципами объектно-ориентированного проектирования, но и получить практические навыки в проектировании архитектуры информационных систем с применением инструментов, соответствующих современным требованиям разработки программного обеспечения.

Создание диаграммы развертывания стало важным этапом в понимании архитектурных аспектов системы и принципов работы её компонентов. В частности, внимание было уделено взаимодействию между ключевыми элементами приложения, такими как контейнеры с fastapi, базой данных PostgreSQL, а также web-интерфейсом, построенным на Vue.js. Подробное представление о том, как эти компоненты взаимодействуют друг с другом, позволяет разработчикам создавать эффективные и масштабируемые системы, что критически важно в условиях динамичного изменения требований и технологий. Наглядное изображение структуры системы помогает в анализе и оптимизации процессов, что в свою очередь снижает риски ошибок в ходе разработки.

Использование Docker в проекте предоставило множество преимуществ, таких как возможность изолированного развертывания компонентов, управление зависимостями и простота масштабирования. Docker позволяет создать предсказуемую среду выполнения, что упрощает тестирование и развертывание приложений. Применение таких современных фреймворков, как fastapi и таких ORM, как SQLAlchemy, способствует улучшению производительности приложения и облегчает разработку за счет предоставления готовых решений для распространенных задач, таких как маршрутизация, работа с базами данных и управление состоянием.

Особое внимание в ходе работы было уделено тому, как компоненты системы, описанные в диаграмме развертывания, связаны друг с другом. Взаимодействие между контейнерами осуществляется через четко определенные протоколы, такие как TCP и HTTP, что способствует организации надежной и быстрой передачи данных.

Наглядность диаграммы развертывания также сыграла ключевую роль в упрощении процессов документирования и общения между членами команды. Это важно для успешной реализации проектов, так как визуальное представление помогает всем участникам проекта лучше понять структуру системы и её компоненты, а также выявить потенциальные проблемы на ранних этапах разработки. Таким образом, диаграмма развертывания не только служит инструментом для анализа архитектуры системы, но и помогает в дальнейшем обучении команды и вовлечении новых участников в проект.