Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа № 6.

Объектно-ориентированное моделирование. Физические диаграммы UML.

Студент: Яшный Н. С.

ФИТ 4 курс 4 группа

Преподаватель: Якубенко К. Д.

Минск 2024

1. **Тема и цель работы**

Темой данной лабораторной работы является построение UML-диаграммы вариантов использования, которая выступает важным инструментом для графического представления взаимосвязей между ролями конечных пользователей и вариантами взаимодействия с системой. В данном случае речь идет о сервисе поиска недвижимости «HomeHub», который был представлен в первой лабораторной работе. Этот сервис предоставляет пользователям возможность находить, сравнивать и управлять объектами недвижимости, включая квартиры, дома и коммерческую недвижимость. В условиях современного рынка, где информационные технологии играют ключевую роль, такой сервис становится особенно актуальным и востребованным, обеспечивая удобство и доступность информации для пользователей.

Целью лабораторной работы является глубокое изучение методологии объектно-ориентированного моделирования с использованием средств UML. Этот процесс включает в себя не только знакомство с основными принципами объектно-ориентированного проектирования программного обеспечения, но и практическое применение этих принципов при создании диаграмм. Важно отметить, что UML предоставляет стандартизированный язык для описания различных аспектов программной системы, что способствует более эффективному общению между членами команды разработки. В процессе работы студенты получат необходимые навыки для проектирования функциональности информационной системы, что является критически важным для успешного выполнения задач, связанных с разработкой программного обеспечения.

Изучение UML-диаграмм вариантов использования позволяет не только лучше понять, как конечные пользователи взаимодействуют с системой, но и выявить потребности различных групп пользователей. Это знание помогает определить, какие функции они могут выполнять и какие требования предъявляются к разработчикам для реализации этих функций. Кроме того, графическое представление взаимодействий создает общее представление о системе, что упрощает дальнейшее тестирование и поддержку. Это делает UML-диаграммы неотъемлемой частью современного подхода к разработке программного обеспечения, обеспечивая структурированный и организованный процесс проектирования.

Таким образом, данная лабораторная работа направлена на формирование у студентов практических и теоретических навыков, необходимых для качественного и эффективного проектирования информационных систем. Она также способствует углублению понимания объектно-ориентированного подхода в программировании и важности системного мышления при разработке сложных программных решений. Студенты смогут перенести полученные знания на реальные проекты, что значительно повысит их профессиональную подготовку в сфере разработки программного обеспечения.

1. **Описание функциональных требований**

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

Функционал для пользователя:

* обеспечивать возможность регистрации, авторизации и аутентификации;
* просмотр каталога недвижимости;
* поиск и фильтрация недвижимости;
* просмотр подробной информации о недвижимости;
* просмотр информации о продавце;
* добавление объявлений в избранное;
* публикация собственных объявлений;
* удаление собственных объявлений;
* просмотр новостей;
* возможность оставить комментарий;
* просмотр геолокации недвижимости на карте.

Функционал для администратора:

* просмотр списка всех пользователей;
* публикация новостей;
* возможность изменить роль пользователя;
* возможность заблокировать/разблокировать пользователя;
* удаление любого объявления;
* просмотр геолокации недвижимости на карте.

Функционал для гостя:

* регистрация;
* просмотр каталога недвижимости;
* поиск и фильтрация недвижимости;
* просмотр подробной информации о недвижимости;
* просмотр информации о продавце;
* просмотр новостей;
* просмотр геолокации недвижимости на карте.

1. **Описание программных средств**

Для построения UML-диаграмм вариантов использования использовался веб-ресурс Draw.io, разработанный компанией JGraph Ltd. Этот инструмент предназначен для создания различных типов диаграмм и визуализаций, что делает его незаменимым помощником для разработчиков, проектировщиков и менеджеров. Адрес веб-ресурса – <https://www.drawio.com>.

Draw.io доступен на всех платформах, которые имеют веб-браузер и доступ в Интернет, что обеспечивает его универсальность и удобство использования. Пользователи могут работать с ним на различных устройствах, включая настольные компьютеры, ноутбуки и даже мобильные устройства, что делает процесс создания диаграмм еще более гибким и доступным. Также стоит отметить, что Draw.io не требует регистрации или создания учетной записи, что упрощает доступ к инструменту и позволяет быстро начать работу.

Функционал веб-ресурса Draw.io весьма разнообразен и включает в себя возможность построения не только UML-диаграмм, но и других графических элементов, таких как графики, смысловые карты, диаграммы Венна, Agile и Kanban доски, а также графики мозговых штурмов и диаграммы архитектур технических систем. Это позволяет пользователям создавать визуализации, которые соответствуют самым различным потребностям и задачам, будь то разработка программного обеспечения, проектирование бизнес-процессов или образовательные цели. Например, возможность создания диаграмм рабочего процесса может быть полезной для оптимизации бизнес-операций, а диаграммы архитектуры могут помочь в представлении сложных технических систем.

Кроме того, Draw.io предлагает интеграцию с популярными облачными хранилищами и платформами, такими как Google Drive, OneDrive, GitHub и многие другие. Эта функциональность позволяет пользователям легко сохранять свои работы и делиться ими с коллегами, что особенно важно для командной работы и совместного проектирования. Интеграция с такими платформами также обеспечивает возможность совместного редактирования в реальном времени, что значительно улучшает взаимодействие между членами команды.

Таким образом, использование Draw.io для создания UML-диаграмм вариантов использования не только упрощает процесс визуализации, но и значительно увеличивает эффективность работы команды, позволяя сосредоточиться на ключевых аспектах проектирования и разработки. Это делает инструмент идеальным выбором для организаций, стремящихся к повышению производительности и качеству своей работы в области разработки программного обеспечения и управления проектами.

1. **Описание практического задания**

В соответствии с вышеописанными функциональными требованиями к программного средству была построена диаграмма развертывания.

Диаграмма развертывания для сервиса «HomeHub» представлена на рисунке 4.1.

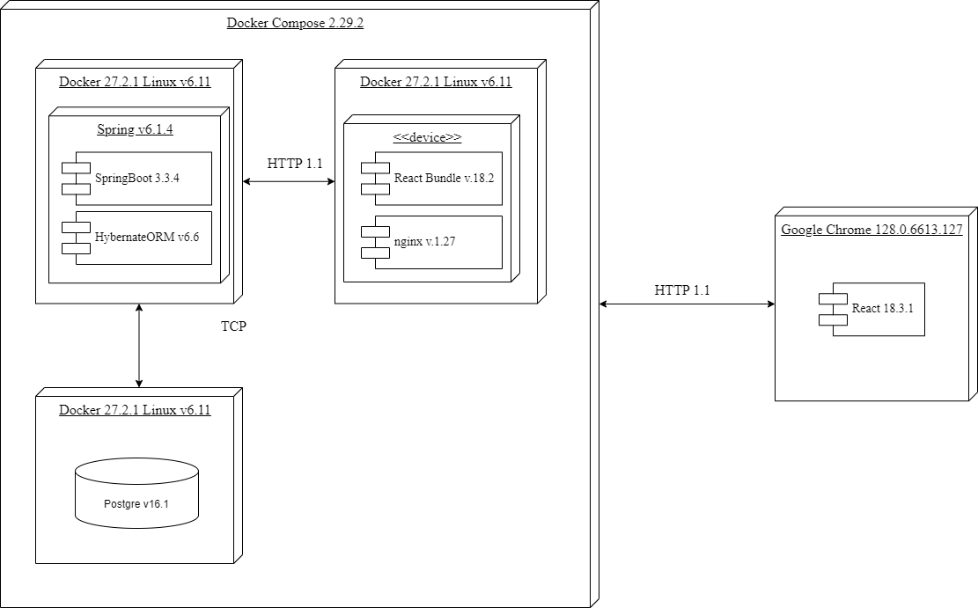


Рисунок 4.1. – Диаграмма развертывания приложения «HomeHub»

Приложение состоит из двух основных компонентов: Docker Compose и Client. В Docker Compose находятся три контейнера. Первый контейнер отвечает за базу данных и использует PostgreSQL для хранения данных. Он обеспечивает надежное управление данными, позволяя другим компонентам приложения взаимодействовать с ними через SQL-запросы.

Второй контейнер представляет собой backend-сервер на основе Spring Boot. Этот сервер реализует бизнес-логику приложения, обрабатывает запросы от клиента и взаимодействует с базой данных через JDBC или JPA. Он предоставляет API, которое позволяет фронтенду запрашивать и отправлять данные.

Третий контейнер — это Nginx, который используется для раздачи статических файлов и управления входящими HTTP-запросами. Nginx служит обратным прокси-сервером, перенаправляя запросы от клиента к Spring Boot, обеспечивая тем самым взаимодействие между фронтендом и бэкендом.

Client представляет собой Google Chrome, в котором выполняется React-приложение. Пользователи взаимодействуют с интерфейсом, созданным на React, который отправляет запросы к backend-серверу через Nginx.

Связь между контейнерами осуществляется через различные протоколы. Например, database-server и application-server соединены по TCP, что позволяет Spring Boot отправлять SQL-запросы к базе данных и получать результаты. Взаимодействие между Nginx и application-server происходит по протоколу HTTP, что позволяет Nginx перенаправлять запросы клиента к соответствующим API-эндпоинтам Spring Boot. Наконец, связь между Docker Compose и Client также осуществляется через HTTP, где клиент отправляет запросы к Nginx, а тот, в свою очередь, взаимодействует с backend и базой данных.

1. **Ответы на вопросы**

Следующие ответы на вопросы:

* Укажите назначение физических диаграмм: компонентов и развертывания: диаграмма компонентов служит для визуализации структуры системы через её модули и взаимодействия между ними. Она помогает разработчикам увидеть, как различные компоненты, такие как библиотеки или модули, соединены и взаимодействуют, что упрощает управление зависимостями. Диаграмма развертывания отображает физическую архитектуру системы, включая узлы и артефакты, показывая, как программные компоненты размещаются на физических устройствах и взаимодействуют друг с другом в реальной среде.
* Дайте определение понятиям: узел, артефакт, интерфейс: узел — это физическая сущность, которая выполняет задачи или хранит данные, например, сервер или компьютер. Артефакт — это результат разработки, такой как исполняемые файлы или библиотеки, которые размещаются на узлах. Интерфейс — это точка взаимодействия между компонентами или узлами, через которую происходит обмен данными, позволяя компонентам взаимодействовать друг с другом.
* Опишите нотации, которые используются для представления компонентов: в диаграммах компонентов используются несколько ключевых нотаций. Компоненты отображаются в виде прямоугольников с закругленными углами, часто с иконкой, обозначающей их тип. Интерфейсы представлены кругами, прикрепленными к компонентам: пустой круг указывает на предоставляемый интерфейс, а закрашенный — на используемый. Зависимость между компонентами изображается пунктирной стрелкой, указывающей от зависимого компонента к тому, от которого он зависит. Пакеты представляют собой группы компонентов и отображаются в виде прямоугольников с открытым верхом.
* Опишите основные нотации, которые используются для представления архитектуры системы в виде диаграммы развертывания: в диаграммах развертывания узлы представлены трехмерными прямоугольниками, отражающими физические сущности. Артефакты изображаются как прямоугольники с листком бумаги, указывающими на физические элементы, размещаемые на узлах. Соединения между узлами отображаются линиями, которые указывают на физические связи, такие как сеть. Интерфейсы могут быть показаны так же, как в диаграммах компонентов, обозначая точки взаимодействия.
* Укажите основные виды связей между компонентами и между узлами: связи между компонентами и узлами бывают разных типов. Зависимость обозначается пунктирной стрелкой, указывающей от зависимого элемента к тому, от которого он зависит. Ассоциация отображается сплошной линией и показывает взаимодействие между элементами. Обобщение обозначает наследование и изображается стрелкой с пустым наконечником, указывающей от дочернего компонента к родительскому.