Міністерство освіти і науки України

Національний технічний університет України “Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки Кафедра інформаційних систем та технологій

Лабораторна робота №3

Технології розроблення програмного забезпечення

Тема: «Основи проектування розгортання»

Виконав:

Студент групи ІА-31

Калиновський В.О

Перевірив:

Мягкий Михайло Юрійович

Київ-2025

Мета: Навчитися проєктувати діаграми розгортання та компонентів для системи що проєктується, а також розробляти діаграми взаємодії, а саме діаграми послідовностей, на основі сценаріїв зроблених в попередній лабораторній роботі.

Тема роботи:

4. Графічний редактор (proxy, prototype, decorator, bridge, flyweight, SOA)

Графічний редактор повинен вміти створювати / редагувати растрові (або векторні на розсуд студента) зображення в 2-3 основних популярних форматах (bmp, png, jpg), мати панель інструментів для створення графічних примітивів, вибору кольорів, нанесення тексту, додавання найпростіших візуальних ефектів (ч/б растр, інфрачервоний растр, 2-3 на вибір учня), роботи з шарами.

3.1. Завдання

• Ознайомитись з короткими теоретичними відомостями.

• Проаналізувати діаграми створені в попередній лабораторній роботі а також тему системи та спроєктувати діаграму розгортання використання відповідно до обраної теми лабораторного циклу.

• Розробити діаграму компонентів для проєктованої системи.

• Розробити діаграму розгортання для проєктованої системи.

• Розробити як мінімум дві діаграми послідовностей для сценаріїв прописаних в попередній лабораторній роботі.

• На основі спроєктованих діаграм розгортання та компонентів доопрацювати програмну частину системи. Реалізація системи, додатково до попередньої реалізації, повинна містити як мінімум дві візуальні форми. В системі вже повинен бути повністю реалізована архітектура (повний цикл роботи з даними від вводу на формі до збереження їх в БД і подальшій виборці з БД та відображенням на UI).

• Підготувати звіт щодо виконання лабораторної роботи. Поданий звіт повинен містити: діаграму розгортання з описом, діаграму компонентів системи з описом, діаграми послідовностей, а також вихідний код системи, який було додано в цій лабораторній роботі.

3.2. Теоретичні відомості

3.2.1. **Діаграма розгортання** (Deployment Diagram) Діаграми розгортання представляють фізичне розташування системи, показуючи, на якому фізичному обладнанні запускається та чи інша складова програмного забезпечення [3]. Головними елементами діаграми є вузли, пов'язані інформаційними шляхами.

Вузол (node) – це те, що може містити програмне забезпечення. Вузли бувають двох типів. Пристрій (device) – це фізичне обладнання: комп'ютер або пристрій, пов'язаний із системою. Середовище виконання (execution environment) – це програмне забезпечення, яке саме може включати інше програмне забезпечення, наприклад операційну систему або процес-контейнер (наприклад, вебсервер).

Діаграма компонентів

Діаграма компонентів UML є представленням проєктованої системи, розбитої на окремі модулі. Залежно від способу поділу на модулі розрізняють три види діаграм компонентів:

• логічні;

• фізичні;

• виконувані.

Коли використовують логічне розбиття на компоненти, то у такому разі проєктовану систему віртуально уявляють як набір самостійних, автономних модулів (компонентів), що взаємодіють між собою.

Коли на діаграмі представляють фізичне розбиття, то в такому разі на діаграмі компонентів показують компоненти та залежності між ними. Залежності показують, що класи в з одного компонента використовують класи з іншого компонента. Фізична модель використовується для розуміння які компоненти повинні бути зібрані в інсталяційний пакет.

Діаграми послідовностей

Діаграма послідовностей (Sequence Diagram) – це один із типів діаграм у моделюванні UML (Unified Modeling Language), який використовується для моделювання взаємодії між об'єктами системи у певній послідовності часу. Вона відображає, як об'єкти обмінюються повідомленнями, показуючи порядок і логіку виконання операцій.

Хід роботи

1. Діаграма розгортання Deployment Diagram.

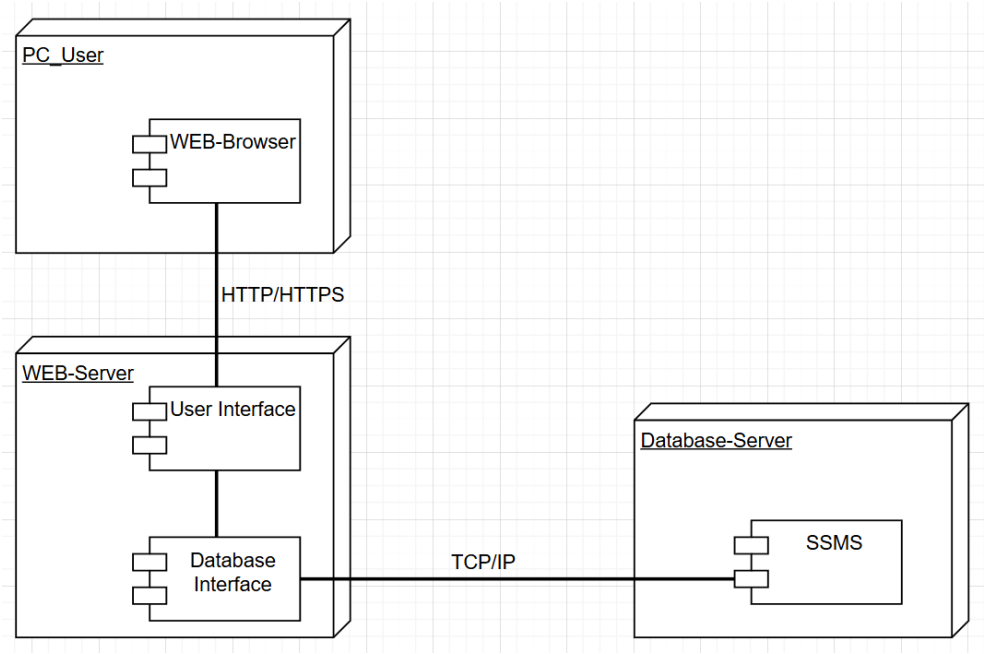

Рисунок 1 – Діаграма розгортання

Діаграма відображає правильну та безпечну трирівневу архітектуру:

1. Рівень Клієнта (PC_User):
    - Компонент: Web browser (Веб-браузер).
    - Взаємодія: Надсилає HTTP/HTTPS-запити на Web Server.
    - Призначення: Відображає інтерфейс користувача та збирає введення.
2. Рівень Сервера (Web Server):
    - Компоненти: User Interface (Обробляє запити від браузера) та Database Interface (Обробляє запити до бази).
    - Взаємодія з Клієнтом: Отримує HTTP/HTTPS-запити.
    - Взаємодія з Базою: Надсилає запити та отримує відповіді від Database Server
3. Рівень Даних (Database Server):
    - Компонент: MsSQL database (База даних MsSQL).
    - Взаємодія: Приймає лише запити від Web Server через TCP/IP.
    - Призначення: Зберігання та управління даними.

2. Діаграма компонентів.

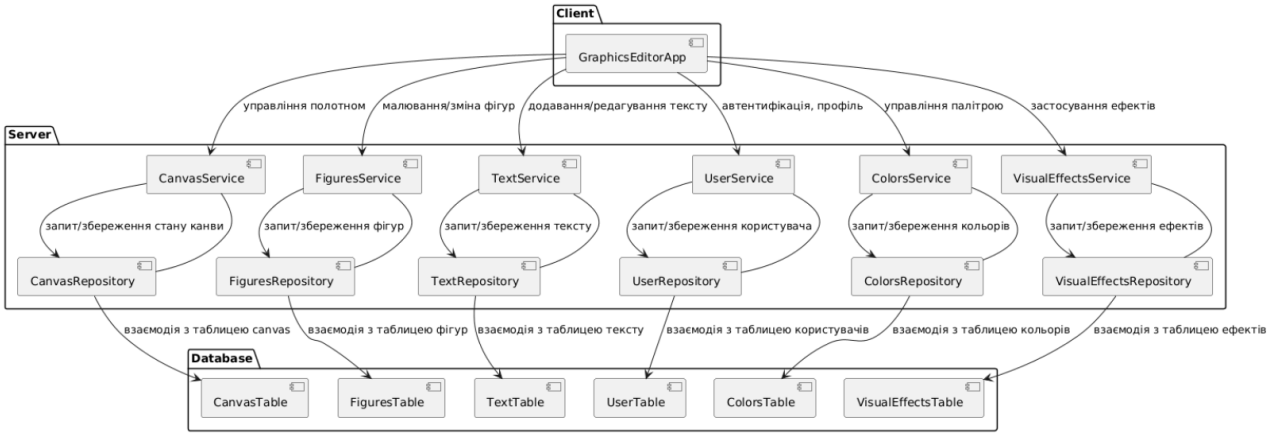

Рисунок 2 –діаграма компонентів

На діаграмі зображено архітектуру графічного редактора, де система поділена на три основні частини: Client, Server та Database.

1. Client

Компонент GraphicsEditorApp представляє клієнтську частину системи — графічний інтерфейс користувача. Він забезпечує взаємодію користувача з редактором: управління полотном, малювання фігур, додавання тексту, управління профілем користувача, вибір кольорів та застосування ефектів. Усі запити від користувача передаються на сервер через відповідні сервіси.

2. Server

Серверна частина містить шість сервісів, кожен з яких відповідає за окрему функціональність редактора:

- CanvasService – управління станом полотна (збереження, завантаження, зміна розмірів);
- FiguresService – створення, редагування, видалення та управління фігурами;
- TextService – додавання та форматування текстових елементів на полотні;
- UserService – автентифікація та керування профілями користувачів;
- ColorsService – управління палітрами кольорів, збереження власних кольорів;
- VisualEffectsService – застосування та налаштування візуальних ефектів до об'єктів.

Кожен сервіс взаємодіє зі своїм Repository, який відповідає за доступ до даних у базі.

3. Repositories

Усі репозиторії розміщені на сервері. Вони реалізують шар доступу до даних і забезпечують взаємодію між сервісами та базою даних:

- CanvasRepository, FiguresRepository, TextRepository, UserRepository, ColorsRepository, VisualEffectsRepository.

Репозиторії отримують, зберігають і оновлюють відповідні дані у своїх таблицях.

4. Database

База даних містить таблиці:

- CanvasTable – збереження стану полотна, його розмірів та налаштувань;
- FiguresTable – дані про фігури (координати, колір, тип, прозорість);
- TextTable – збережені текстові блоки, їх стилі, шрифти та позиції;
- UserTable – інформація про користувачів, їхні збережені проекти та налаштування;
- ColorsTable – збережені користувацькі палітри кольорів;
- VisualEffectsTable – дані про застосовані ефекти та їх параметри.

Сервер взаємодіє з базою через репозиторії, виконуючи запити на читання або запис даних.

3. Діаграма послідовностей.

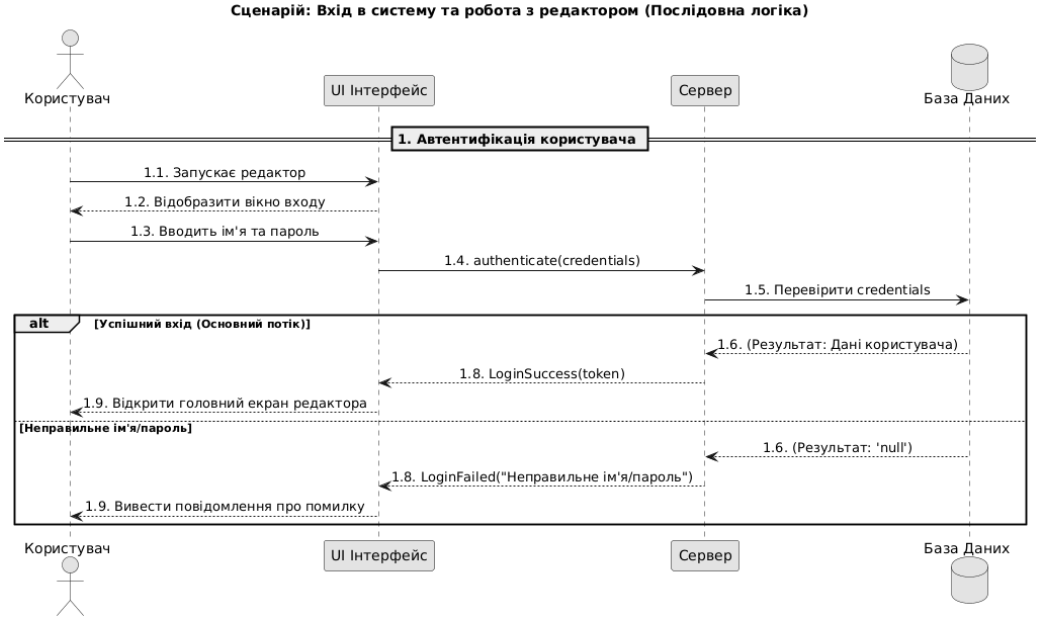

Рисунок 3 – Діаграма послідовностей “Вхід користувача до системи графічного редактора”

Діаграма показує взаємодію користувача з редактором — від входу в систему до редагування зображення та збереження результату. У процесі беруть участь користувач, інтерфейс (UI), сервер і база даних.

Спершу користувач запускає редактор, після чого інтерфейс відображає вікно входу. Користувач вводить ім’я та пароль, і ці дані надсилаються на сервер. Сервер звертається до бази даних, щоб перевірити правильність облікових даних. Якщо база повертає інформацію про користувача, сервер надсилає UI повідомлення про успішний вхід, і інтерфейс відкриває головний екран редактора. Якщо база даних повертає порожній результат, сервер відправляє назад повідомлення про помилку, і користувач бачить відповідне попередження.

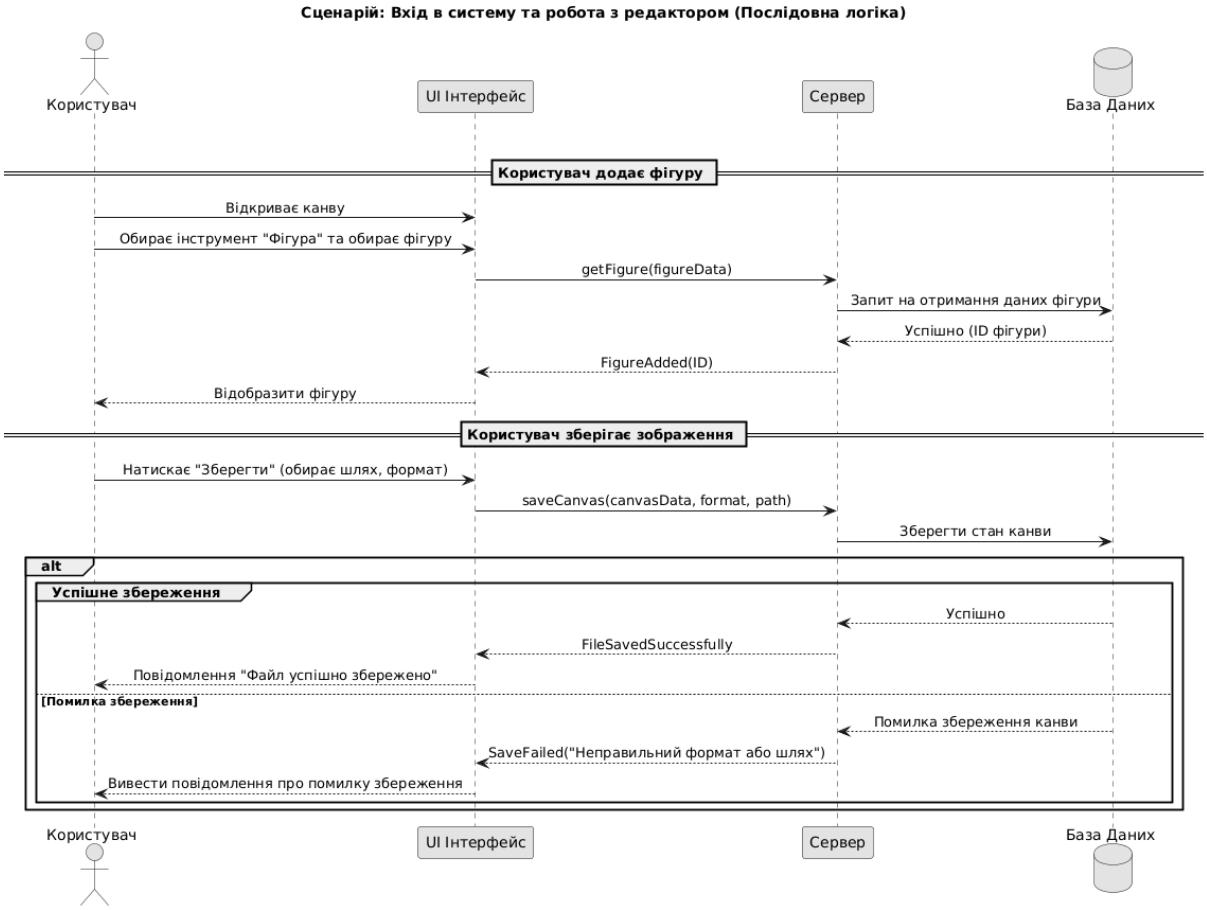

Рисунок 3 – Діаграма послідовностей “Додавання фігури та збереження файлу”

Діаграма показує роботу користувача з редактором після успішного входу в систему.

Спершу користувач відкриває канву та обирає інструмент для додавання фігури. Інтерфейс (UI) надсилає запит на сервер, який звертається до бази даних, щоб отримати дані фігури. Після отримання ID фігури сервер повертає інформацію в UI, і фігура відображається на полотні.

Далі користувач може зберегти створене зображення. UI передає дані на сервер, який намагається зберегти стан канви в базі даних. Якщо збереження проходить успішно, сервер повідомляє UI про успіх, і користувач отримує підтвердження “Файл успішно збережено”. Якщо виникає помилка (наприклад, неправильний шлях або формат), сервер повідомляє про невдачу, і UI відображає повідомлення про помилку збереження.

Таким чином, діаграма відображає послідовну логіку редагування та збереження зображення після автентифікації.

Висновок:

Під час виконання лабораторної роботи я навчився створювати діаграми розгортання, компонентів і послідовностей. Зрозумів їх принцип роботи та побудови, основні відмінності та випадки застосування. Завдяки цим діаграмам стало зрозуміліше, як працює система в цілому і як її реалізувати.

Контрольні запитання:

1. Що собою становить діаграма розгортання?
   Діаграма розгортання показує фізичне розташування компонентів системи на апаратних вузлах і взаємодію між ними.
2. Які бувають види вузлів на діаграмі розгортання?
   Вузли бувають двох типів: пристрій — фізичне обладнання, наприклад комп’ютер або сервер, і середовище виконання — програмне забезпечення, яке може включати інші програми, як операційна система або вебсервер.
3. Які бувають зв’язки на діаграмі розгортання?
   На діаграмі розгортання використовуються зв’язки, такі як асоціації, залежності та шляхи комунікації між вузлами.
4. Які елементи присутні на діаграмі компонентів?
   Діаграма компонентів відображає структуру системи у вигляді компонентів, їхніх інтерфейсів та пакетів.

5. Що становлять собою зв’язки на діаграмі компонентів?
   Зв’язки між компонентами показують залежності, асоціації або реалізацію інтерфейсів.

6. Які бувають види діаграм взаємодії?
   Діаграми взаємодії можуть бути різних видів, зокрема діаграми послідовностей, комунікацій, часу та огляду взаємодії.

7. Для чого призначена діаграма послідовностей?
   Діаграма послідовностей призначена для відображення часової послідовності повідомлень між об’єктами під час виконання сценарію.

8. Які ключові елементи можуть бути на діаграмі послідовностей?
   Основні елементи діаграми : актори позначають користувачів або зовнішні системи, що взаємодіють із системою. Об’єкти або класи розміщені горизонтально та мають лінію життя, яка показує їх існування в часі. Повідомлення зі стрілками відображають виклики методів або обмін даними, синхронні або асинхронні, з пунктирними лініями для повернення результату. Активності показують періоди виконання дій об’єктом і зображуються прямокутниками на лінії життя. Контрольні структури, як alt або loop, відображають умови, альтернативи та цикли.

9. Як діаграми послідовностей пов’язані з діаграмами варіантів використання?
   Діаграми послідовностей деталізують сценарії, які представлені на діаграмах варіантів використання, показуючи порядок обміну повідомленнями для конкретних випадків.

10. Як діаграми послідовностей пов’язані з діаграмами класів?
    Вони пов’язані з діаграмами класів, оскільки показують, як методи класів викликаються під час виконання сценарію, відображаючи динамічну поведінку структури класів.