Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційні систем та технологій

Лабораторна робота № 3

із дисципліни «Технології розроблення програмного забезпечення»

Тема: «Основи проектування розгортання»

Виконав
Студент групи ІА-31:
Підковка Д. О.

Перевірив:
Мягкий М. Ю.

Київ 2025

Зміст

1. **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.

2. **Хід роботи:**

Тема : Web-browser

1) Ознайомитись з короткими теоретичними відомостями.
2) Проаналізувати діаграми створені в попередній лабораторній роботі а також тему системи та спроєктувати діаграму розгортання використання відповідно до обраної теми лабораторного циклу

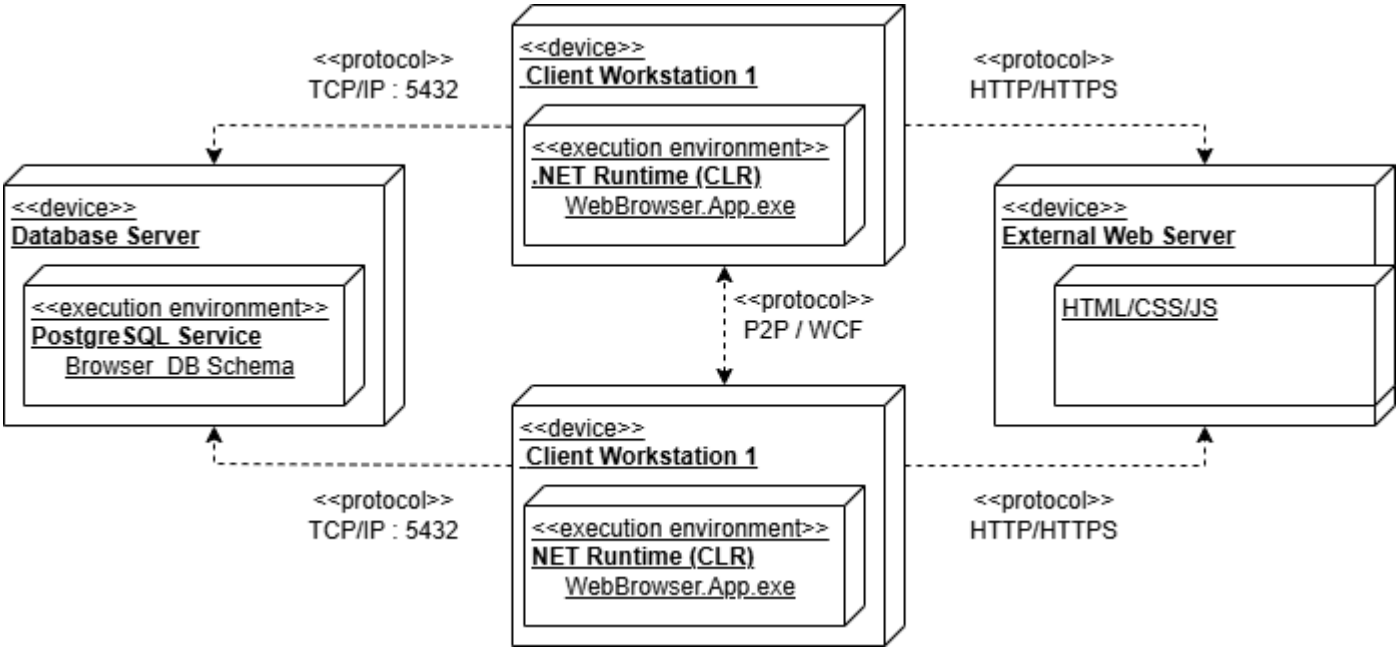

Рис.1 – Діаграма розгортання

На діаграмі розгортання зображено фізичну архітектуру настільного застосунку «Web-browser» . Система побудована за гібридною архітектурою, що поєднує клієнт-серверну взаємодію та елементи однорангової мережі (P2P), і включає такі вузли:

- Клієнтські робочі станції (Client Workstation): Представлені як персональні комп'ютери під керуванням Windows OS. Вони виступають середовищем виконання (.NET Runtime) для артефакту WebBrowser.App.exe — головного виконуваного файлу програми. Вузли підтримують пряму P2P-взаємодію між собою для синхронізації даних без участі центрального сервера.
- Сервер бази даних (Database Server): Фізично відокремлений сервер, на якому розгорнуто систему управління базами даних PostgreSQL. Він відповідає за

надійне централізоване зберігання даних користувачів (історії відвідувань та закладок), взаємодіючи з клієнтами через протокол TCP/IP.

- Зовнішні веб-сервери (External Web Server): Вузли глобальної мережі Інтернет, що надають веб-контент (HTML, CSS, зображення). Клієнтський застосунок взаємодіє з ними через протоколи HTTP/HTTPS для отримання та відображення інформації.

3) Розробити діаграму компонентів для проєктованої системи

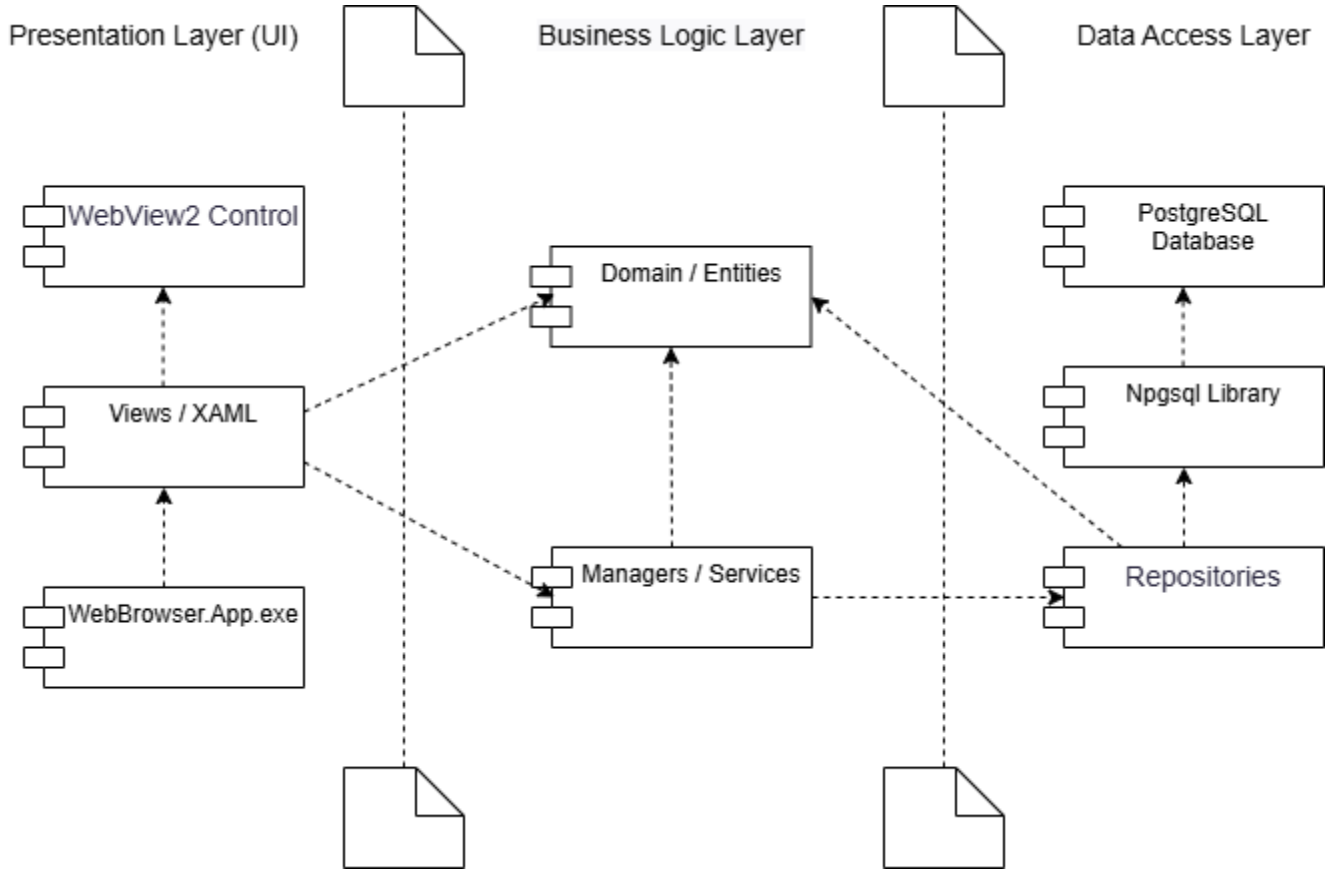

Рис.2 – Діаграма компонентів

На діаграмі компонентів зображено логічну архітектуру настільного застосунку «Web-browser» . Система побудована за багаторівневою моделлю (N-Tier Architecture) та включає такі рівні компонентів:

- Рівень представлення (Presentation Layer): Представлений компонентом WebBrowser.App, який відповідає за графічний інтерфейс користувача (Views на базі WPF). Він взаємодіє з користувачем та використовує зовнішній інфраструктурний компонент WebView2 (на базі Microsoft Edge) як рушій для відображення веб-контенту.

- Рівень бізнес-логіки (Business Logic Layer): Це центральний рівень, представлений компонентом Managers. Він виконує основну обробку даних

(наприклад, валідацію URL, логіку пошуку в історії) та керує потоками інформації між інтерфейсом та даними, використовуючи спільні сутності з компонента Domain.

- Рівень даних (Data Access Layer): Представлений компонентом Repositories, що реалізує патерн Repository для абстрагування доступу до даних. Він використовує зовнішню бібліотеку Npgsql для фізичного з'єднання та виконання запитів до системи управління базами даних PostgreSQL.

4) Розробити як мінімум дві діаграми послідовностей для сценаріїв прописаних в попередній лабораторній роботі

Завантаження веб-сторінки за URL	
Передумови	Браузер запущено. Наявне активне підключення до мережі Інтернет. Користувач знаходиться в головному вікні програми.
Постумови	Успіх: Веб-сторінка успішно завантажена, оброблена та відображена у новій вкладці. Контент збережено у локальний кеш. Невдача: Відображено сторінку з повідомленням про помилку (404, помилка безпеки або відсутність мережі).
Взаємодіючі сторони	Користувач, Система, Зовнішній Веб-сервер
Короткий опис	Цей сценарій описує складний процес переходу за посиланням, який включає динамічне створення вкладки, оптимізацію трафіку через кешування, багаторівневу перевірку безпеки запиту та поетапний рендеринг контенту.
Основний потік подій	Користувач вводить URL-адресу в адресний рядок і ініціює перехід.
	Система звертається до фабричного методу (Factory Method) для створення екземпляра нової вкладки потрібного типу.
	Вкладка звертається до замісника (Proxy) для отримання контенту сторінки.
	Замісник перевіряє наявність актуальної версії сторінки у локальному кеші.
	Ланцюжок послідовно виконує валідацію URL, перевірку на фішинг (SecurityHandler) та завантаження даних з мережі.
	Отриманий HTML-код передається рендереру, який використовує шаблонний метод (Template Method) для виконання фіксованої послідовності кроків: парсинг HTML, побудова DOM, завантаження стилів, відмальовування.
	Готова сторінка відображається користувачеві.
Винятки	Загроза безпеки: На етапі обробки ланцюжком модуль безпеки виявляє шкідливий сайт. Завантаження переривається, система виводить попередження. Помилка мережі: Сервер не відповідає. Proxy повертає стандартну технічну сторінку помилки.
Примітки	У сценарії продемонстровано використання патернів: Factory Method, Proxy, Chain of Responsibility, Template Method.

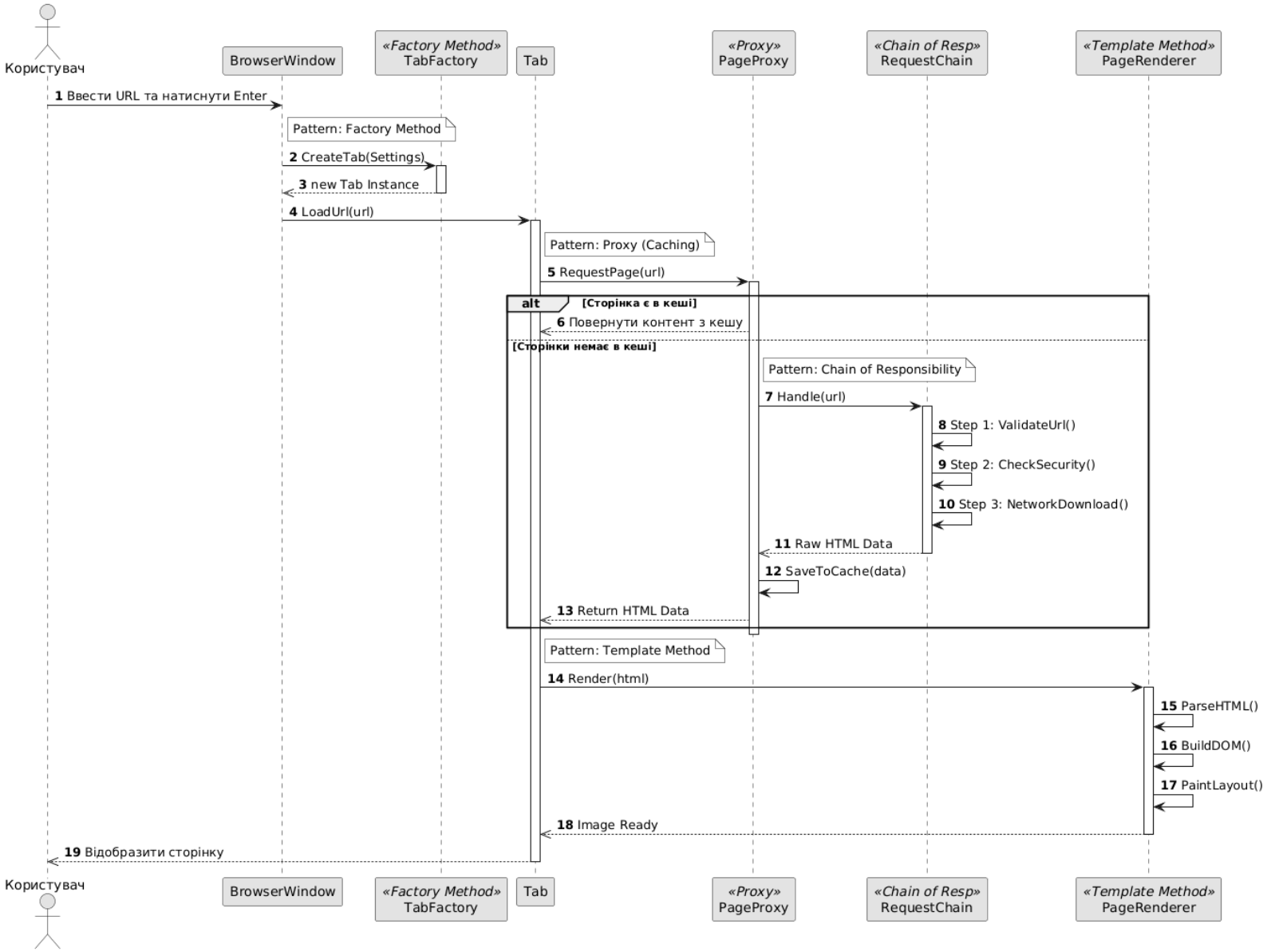

Рисунок 3.1. Діаграма послідовності «Прослуховування радіо»

1. Користувач вводить URL-адресу в адресний рядок і ініціює перехід.
2. Система звертається до фабричного методу (Factory Method) для створення екземпляра нової вкладки потрібного типу.
3. Вкладка звертається до замісника (Proxy) для отримання контенту сторінки.
4. Замісник перевіряє наявність актуальної версії сторінки у локальному кеші.
5. Ланцюжок послідовно виконує валідацію URL, перевірку на фішинг (SecurityHandler) та завантаження даних з мережі.
6. Отриманий HTML-код передається рендереру, який використовує шаблонний метод (Template Method) для виконання фіксованої послідовності кроків: парсинг HTML, побудова DOM, завантаження стилів, відмальовування.
7. Готова сторінка відображається користувачеві.

Перегляд історії відвідувань	
Передумови	Браузер запущено. У локальній базі даних наявні записи історії. (Опціонально) В локальній мережі доступні інші пристрої користувача для синхронізації.
Постумови	Успіх: Користувач бачить повний, синхронізований та відформатований список історії відвідувань. Невдача: Відображено повідомлення про помилку доступу до даних.
Взаємодіючі сторони	Користувач, Система
Короткий опис	Сценарій описує процес отримання даних про відвідувані сайти, їх синхронізацію з іншими пристроями через одноранговy мережу (P2P) та підготовку даних для відображення у зручному форматі.
Основний потік подій	Користувач обирає пункт меню "Історія" в інтерфейсі браузера.
	Система звертається до фабричного методу (Factory Method) для створення екземпляра нової вкладки потрібного типу.
	Система ініціює процес отримання даних через менеджер історії.
	Менеджер викликає P2P-сервіс, який сканує локальну мережу на наявність інших запущених екземплярів браузера цього користувача.
	Якщо вузли знайдено, відбувається прямий обмін новими записами історії (P2P синхронізація).
	Менеджер завантажує локальні записи з бази даних через репозиторій.
	Для підготовки різнорідних даних до відображення застосовується патерн Відвідувач (Visitor). Він обходить список записів і формує єдиний звіт (наприклад, форматує дати, групує за днями, генерує HTML-представлення).
	Сформований список відображається у вікні історії.
Винятки	P2P недоступний: Якщо інші пристрої офлайн або порти заблоковані, етап синхронізації пропускається, і відображається лише локальна історія.
Примітки	У сценарії продемонстровано використання патернів: P2P (Peer-to-Peer), Visitor.

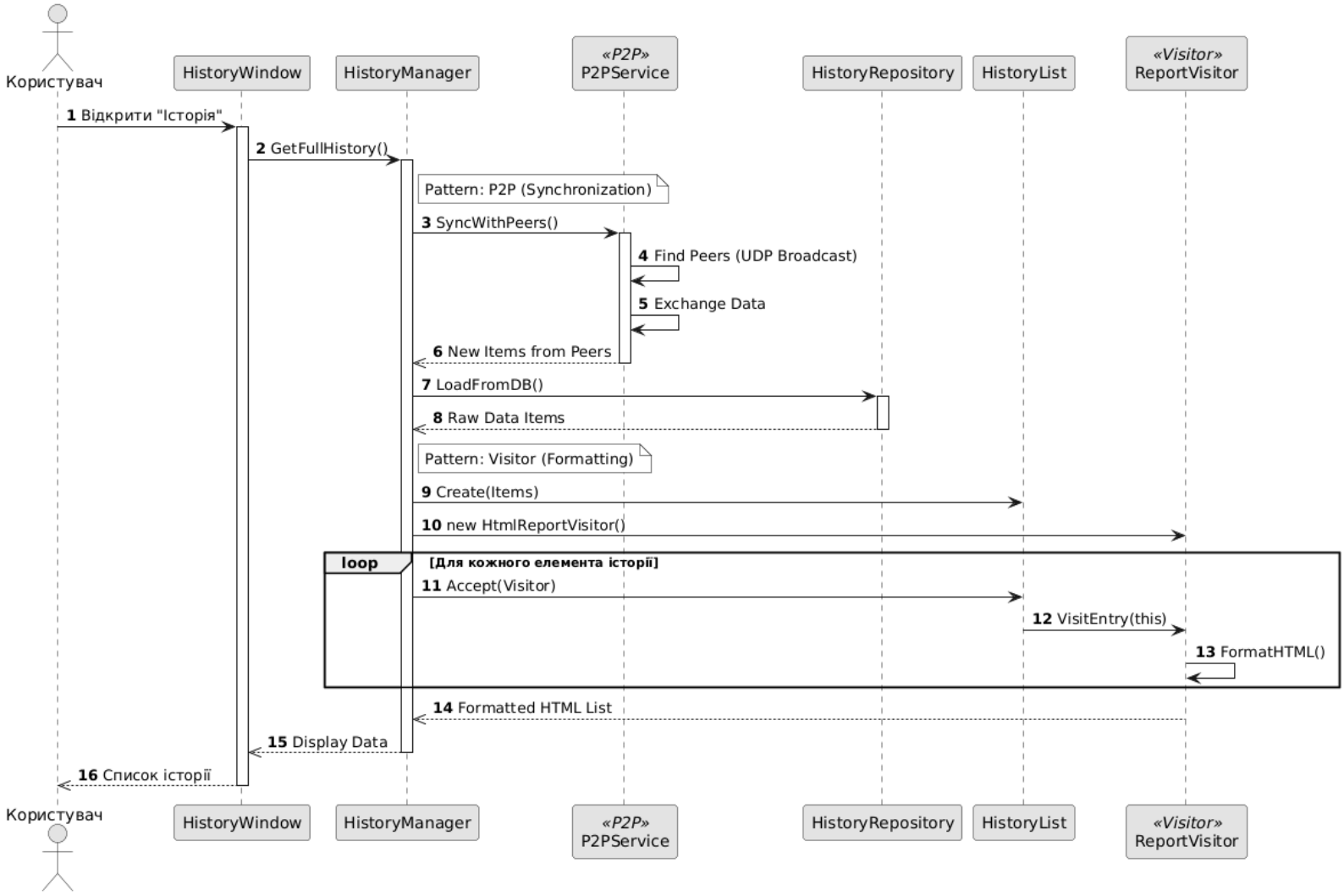

Рисунок 3.2. Діаграма послідовності «Керування чергою відтворення»

Основний потік:

1. Користувач обирає пункт меню "Історія" в інтерфейсі браузера.
2. Система звертається до фабричного методу (Factory Method) для створення екземпляра нової вкладки потрібного типу.
3. Система ініціює процес отримання даних через менеджер історії.
4. Менеджер викликає P2P-сервіс, який сканує локальну мережу на наявність інших запущених екземплярів браузера цього користувача.
5. Якщо вузли знайдено, відбувається прямий обмін новими записами історії (P2P синхронізація).
6. Менеджер завантажує локальні записи з бази даних через репозиторій.
7. Для підготовки різнорідних даних до відображення застосовується патерн Відвідувач (Visitor). Він обходить список записів і формує єдиний звіт (наприклад, форматує дати, групує за днями, генерує HTML-представлення).
8. Сформований список відображається у вікні історії.

5) 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).

Реалізація системи

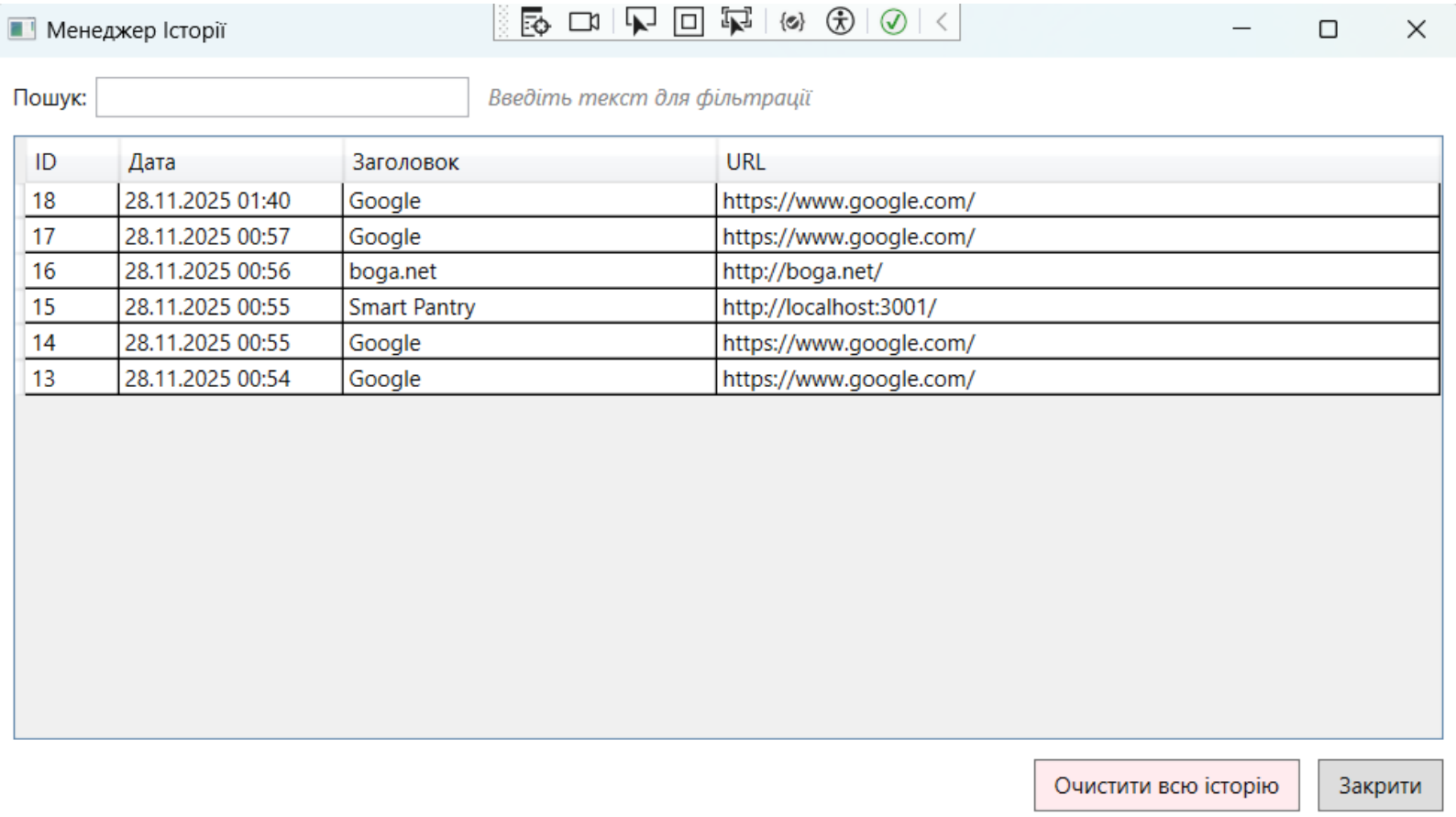

Рисунок 4.1. Вікно Історії

	id [PK] integer	title character varying (255)	url text	visit_date timestamp without time zone
1	13	Google	https://www.google.co...	2025-11-28 00:54:56.72988
2	14	Google	https://www.google.co...	2025-11-28 00:55:10.310563
3	15	Smart Pantry	http://localhost:3001/	2025-11-28 00:55:19.990571
4	16	boga.net	http://boga.net/	2025-11-28 00:56:30.655787
5	17	Google	https://www.google.co...	2025-11-28 00:57:00.590666
6	18	Google	https://www.google.co...	2025-11-28 01:40:07.334693

Рисунок 4.2. Таблиця в PostgreSQL

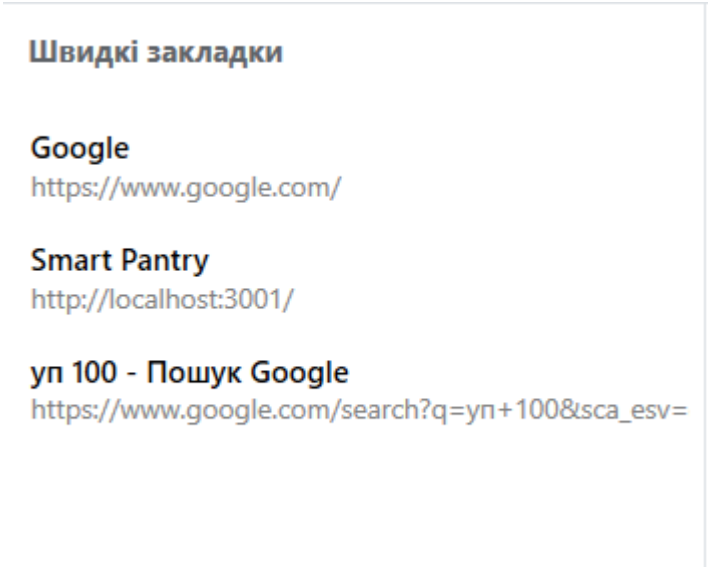

Рисунок 4.3. Вікно Закладок

	id [PK] integer	title character varying (255)	url text
1	1	уп 100 - Пошук Google	https://www.google.com/search?q=уп+100&sca_esv=de84
2	2	Smart Pantry	http://localhost:3001/
3	3	Google	https://www.google.com/

Рисунок 4.4. Таблиця в PostgreSQL

3. Контрольні питання

1) Діаграма розгортання — UML-діаграма, що відображає фізичну структуру системи: на яких пристроях (вузлах) розміщуються програмні компоненти та як вони взаємодіють у реальному середовищі.
2) Види вузлів:

Апаратні вузли — фізичні пристрої (сервер, комп'ютер, смартфон).

Програмні вузли — середовища виконання (операційна система, контейнер, віртуальна машина).
3) Види зв'язків на діаграмі розгортання:

Асоціація — фізичне або логічне з'єднання між вузлами.

Залежність — вказує, що один вузол або компонент використовує ресурси іншого.
4) Елементи діаграми компонентів:

Компоненти, інтерфейси, порти, залежності, пакети, підсистеми.
5) Зв'язки на діаграмі компонентів показують взаємозалежність між компонентами, тобто який компонент використовує або реалізує певний інтерфейс іншого.
6) Види діаграм взаємодії:

Діаграма послідовностей. Діаграма комунікацій.

Діаграма часу.

Діаграма взаємодії огляду.
7) Діаграма послідовностей — показує порядок викликів і повідомлень між об'єктами під час виконання певного сценарію або процесу.

8) Ключові елементи діаграми послідовностей: актори, об’єкти, лінії життя, повідомлення (синхронні та асинхронні), області активації.
9) Зв’язок із діаграмами варіантів використання — діаграми послідовностей деталізують дії, що виконуються для реалізації кожного варіанту використання.

Зв’язок із діаграмами класів — об’єкти, що взаємодіють на діаграмі послідовностей, є екземплярами класів, описаних на діаграмі класів, а повідомлення відповідають викликам їх методів.

4. Висновок

У процесі виконання лабораторної роботи з теми "Розробка веб-браузера" було спроєктовано архітектуру розгортання та компонентну структуру системи за допомогою UML, включаючи діаграми компонентів, розгортання та послідовностей. Реалізовано перехід до багаторівневої архітектури (N-Tier), що дозволило чітко розділити логіку представлення (WPF), бізнес-логіку (Managers) та доступ до даних (Repositories).

У проєкті використано та змодельовано сучасні патерни проєктування, такі як Factory Method, Proxy, Chain of Responsibility, Template Method та Visitor, для забезпечення гнучкості обробки запитів та рендерингу контенту. Результатом є робочий прототип системи з розділеними шарами відповідальності, що відповідає вимогам до масштабованості та підтримує P2P-взаємодію на архітектурному рівні.