МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

БОЖКО МИКОЛА ВІКТОРОВИЧ

Мобільний застосунок з використанням технологій доповненої реальності для візуалізації музейного контенту з урахуванням потреб осіб з порушеннями слуху / Mobile Application Using Augmented Reality Technologies for Visualizing Museum Content with Consideration of the Needs of People with Hearing Impairments

спеціальність: 122 - Комп'ютерні науки
освітньо-професійна програма - Комп'ютерні науки

Кваліфікаційна робота

Виконав студент групи КН-41
М. В. Божко

_____

Науковий керівник
к.т.н., доцент Д.І. Загородня

_____

Кваліфікаційну роботу допущено до захисту
«____»_____20___р.
Завідувач кафедри
_____ Н.В. Дзюбановська

Тернопіль-2026 р.

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність: 122 – Комп'ютерні науки
освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.В. Дзюбановська
«____» _____20__р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
БОЖКУ Миколі Вікторовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: Мобільний застосунок з використанням технологій доповненої реальності для візуалізації музейного контенту з урахуванням потреб осіб з порушеннями слуху / Mobile Application Using Augmented Reality Technologies for Visualizing Museum Content with Consideration of the Needs of People with Hearing Impairments

керівник роботи к.т.н., доцент Д.І. Загородня
затверджений наказом по університету від 01 грудня 2025 року № 894.
2. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2026 р.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.
4. Основні питання, які потрібно розробити:

– провести аналіз існуючих підходів до використання технологій доповненої реальності для візуалізації музейного контенту та забезпечення інклюзивності;
– обґрунтувати вибір технологій для розробки мобільного застосунку та підсистеми розпізнавання музейних експонатів;
– спроєктувати архітектуру програмної системи, що включає мобільний клієнт, AR-модуль та серверну підсистему;
– реалізувати мобільний застосунок із функціями розпізнавання експонатів та відображення контенту в доповненій реальності;провести тестування розробленого програмного забезпечення та оцінити результати його роботи.

5. Перелік графічного матеріалу у роботі:
- загальна архітектура програмної системи;
- архітектура управління станами на базі Riverpod;
- архітектура серверної підсистеми та схема обробки даних у RAG-конвеєрі;
- схема локального алгоритму динамічного розбиття та синхронізації тексту;
- архітектура клієнтської частини мобільного застосунку;
- архітектура серверної підсистеми та модулів штучного інтелекту.

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 01 грудня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Огляд літературних джерел відповідно до предметної області та складання плану роботи	до 01.01.2026 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02.2026 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 15.03.2026 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05.2026 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2026 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2026 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2026 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2026 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06.2026 р.	

Студент _____ М.В. Божко
підпис

Керівник роботи _____ к.т.н., доцент Д.І. Загородня
підпис

АНОТАЦІЯ

Кваліфікаційна робота на тему «Мобільний застосунок з використанням технологій доповненої реальності для візуалізації музейного контенту з урахуванням потреб осіб з порушеннями слуху» на здобуття освітнього ступеня «Бакалавр» зі спеціальності 122 «Комп'ютерні науки» освітньої програми «Комп'ютерні науки» написана обсягом 89 сторінок і містить 20 ілюстрацій, 1 таблицю, 5 додатків та 38 використані джерела.

Метою роботи є підвищення рівня доступності та інтерактивності музейного середовища для осіб з порушеннями слуху шляхом розробки мобільного застосунку на основі технологій доповненої реальності для візуалізації адаптованого музейного контенту.

Методи дослідження: використано методи системного аналізу, об'єктно-орієнтованого проєктування, технології мобільної розробки та доповненої реальності, методи комп'ютерного зору для розпізнавання музейних експонатів, клієнт-серверні технології обміну даними та методи тестування програмного забезпечення.

Результати дослідження: розроблено кросплатформний мобільний застосунок, який забезпечує розпізнавання музейних експонатів та відображення пов'язаного інформаційного контенту в середовищі доповненої реальності. Реалізовано архітектуру програмної системи, що включає мобільний клієнт, AR-модуль, серверну підсистему обробки запитів та механізм потокового відображення текстових повідомлень. Проведене тестування підтвердило працездатність та ефективність розробленого рішення.

Розроблений програмний продукт може бути використаний музеями та культурними установами для забезпечення безбар'єрного доступу до інформації, підвищення інтерактивності експозицій і покращення якості обслуговування відвідувачів з особливими потребами.

Ключові слова: ДОПОВНЕНА РЕАЛЬНІСТЬ, МОБІЛЬНИЙ ЗАСТОСУНОК, МУЗЕЙНИЙ КОНТЕНТ, КОМП'ЮТЕРНИЙ ЗІР, РОЗПІЗНАВАННЯ ЕКСПОНАТІВ, ІНКЛЮЗИВНІ ТЕХНОЛОГІЇ. .

ANNOTATION

The qualification work entitled “Mobile Application Using Augmented Reality Technologies for Visualizing Museum Content with Consideration of the Needs of People with Hearing Impairments” for obtaining the Bachelor's degree in Specialty 122 “Computer Science” under the Educational Program “Computer Science” consists of 89 pages and contains 20 figures, 1 table, 5 appendices, and 38 references.

The aim of the qualification work is to improve the accessibility and interactivity of the museum environment for people with hearing impairments through the development of a mobile application based on augmented reality technologies for visualizing adapted museum content.

Research methods: system analysis methods, object-oriented design methods, mobile development and augmented reality technologies, computer vision methods for museum exhibit recognition, client-server data exchange technologies, and software testing methods were used in the study.

Research results: a cross-platform mobile application has been developed that provides museum exhibit recognition and visualization of related information content in an augmented reality environment. The architecture of the software system, including a mobile client, an AR module, a server-side subsystem, and a streaming text display mechanism, has been implemented. Testing confirmed the functionality and effectiveness of the developed solution.

The developed software product can be used by museums and cultural institutions to provide barrier-free access to information, increase exhibition interactivity, and improve visitor experience for people with special needs.

Keywords: AUGMENTED REALITY, MOBILE APPLICATION, MUSEUM CONTENT, COMPUTER VISION, EXHIBIT RECOGNITION, INCLUSIVE TECHNOLOGIES..

ЗМІСТ

ВСТУП

Сучасний етап розвитку інформаційного суспільства характеризується глибокою цифровізацією всіх сфер життя, серед яких особливе місце займає культурно-освітній простір. Музеї, галереї та виставкові центри активно переходять від парадигми пасивного споглядання до створення інтерактивного середовища, де цифрові технології розширюють можливості взаємодії відвідувача з експозицією. Ключову роль у цих трансформаціях відіграють технології доповненої реальності (Augmented Reality, AR), які забезпечують накладання комп’ютерно-згенерованих візуальних об’єктів на зображення реального світу, створюючи новий рівень сприйняття інформації.

Незважаючи на значний прогрес у впровадженні мультимедійних рішень, проблема забезпечення рівного доступу до культурного контенту для людей з інвалідністю залишається гострою. Традиційна практика застосування аудіогідів або складних текстових описів створює суттєві бар'єри для осіб з порушеннями слуху (глухих та слабочуючих). Читання великих масивів наукового тексту викликає підвищене когнітивне навантаження, особливо для тих користувачів, чиєю першою мовою є жестова. У таких умовах виникає критична необхідність у розробці рішень, які б компенсували відсутність аудіального каналу через посилення та адаптацію візуального сприйняття безпосередньо у фізичному просторі музею.

Актуальність теми кваліфікаційної роботи зумовлена необхідністю створення інклюзивного безбар'єрного середовища шляхом поєднання технологій доповненої реальності та систем штучного інтелекту. На відміну від статичних інформаційних табличок, використання мобільних AR-застосунків дозволяє виконувати просторову прив'язку динамічного контенту (субтитрів, візуальних ефектів або 3D-анімацій) безпосередньо до музейного експоната. Це дозволяє користувачу утримувати фокус зору на об'єкті мистецтва, одночасно отримуючи адаптовану інформацію.

Додатковим фактором, що підсилює актуальність дослідження, є необхідність персоналізації контенту для різних вікових груп осіб з порушеннями

слуху. Використання сучасних серверних технологій генерації та надання контенту дозволяє формувати інформаційні матеріали різної складності та адаптувати їх до потреб різних категорій користувачів. Однак інтеграція таких рішень у мобільне середовище вимагає подолання викликів, пов'язаних із мережевими затримками (latency) та необхідністю оптимізації процесів рендерингу тексту в реальному часі для уникнення візуального перевантаження користувача.

У зв’язку з цим виникає потреба у розробці програмних рішень, що інтегрують алгоритми комп'ютерного зору для розпізнавання музейних експонатів та асинхронної обробки потокових текстових даних у єдиний інклюзивний інтерфейс. Такий підхід передбачає створення кросплатформного застосунку, здатного стабільно функціонувати в умовах реального музейного середовища.

Метою кваліфікаційної роботи є підвищення рівня доступності та інтерактивності музейного простору для осіб з порушеннями слуху шляхом розробки мобільного застосунку на базі технологій доповненої реальності для просторової візуалізації адаптованого контенту.

Для досягнення поставленої мети в роботі необхідно розв’язати такі основні завдання:

– провести аналіз проблеми доступності музейного середовища для осіб з порушеннями слуху та дослідити роль AR-технологій у її вирішенні;

– проаналізувати підходи до організації взаємодії мобільного застосунку із серверною підсистемою надання контенту;

– обґрунтувати вибір технологічного стеку для створення кросплатформного застосунку та реалізації AR-модуля;

– розробити логічну архітектуру програмної системи, алгоритм просторової візуалізації тексту та спроєктувати інклюзивний UI/UX-інтерфейс;

– реалізувати програмний модуль розпізнавання експонатів та потокового відображення AR-субтитрів;

– провести експериментальне тестування розробленого застосунку для оцінки його продуктивності та коректності роботи в умовах мережевих затримок.

Об’єктом дослідження є процеси надання та візуального сприйняття цифрового контенту в інклюзивному музейному середовищі.

Предметом дослідження є методи, алгоритми та програмні засоби доповненої реальності для візуалізації інформації з урахуванням потреб осіб з порушеннями слуху.

Практична значимість роботи полягає у створенні готового кросплатформного мобільного застосунку, який може бути впроваджений у реальних музейних та виставкових просторах для забезпечення безбар'єрного доступу до інформації. Особливістю розробленого програмного забезпечення є поєднання технологій доповненої реальності, розпізнавання музейних експонатів та потокового відображення адаптованого текстового контенту в єдиному мобільному застосунку, орієнтованому на потреби осіб з порушеннями слуху. Розроблені архітектурні рішення, алгоритми потокового відображення тексту та інклюзивні UI/UX-патерни можуть бути використані IT-компаніями та культурними інституціями при створенні інтерактивних гідів, освітніх додатків та систем просторової навігації для людей з порушеннями слуху.

Результати кваліфікаційної роботи були апробовані на IV Всеукраїнській науково-практичній конференції студентів, аспірантів та молодих вчених «Інтелектуальні комп’ютерні системи та мережі», яка відбулася 20 травня 2026р. (м. Тернопіль, Україна) та опубліковані у збірнику матеріалів конференції в доповіді «Мобільний застосунок з використанням технологій доповненої реальності для візуалізації музейного контенту з урахуванням потреб осіб з порушеннями слуху» . Текст публікації наведено у Додатку А.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Опис предметної області

Сучасний етап розвитку інформаційного суспільства висуває нові, значно вищі вимоги до культурно-освітніх та громадських просторів, серед яких ключове місце займають музеї, галереї та виставкові центри. Однією з головних тенденцій розвитку таких інституцій є перехід від парадигми пасивного споглядання до інтерактивної взаємодії відвідувача з експозицією [1]. Традиційні підходи до надання інформації в музеях здебільшого орієнтовані на використання статичних текстових табличок (які часто мають суттєво обмежений обсяг, дрібний шрифт та складну для швидкого сприйняття наукову мову) або портативних аудіогідів. Проте така організація простору створює суттєві, а іноді й нездоланні бар'єри для людей з інвалідністю, зокрема для осіб з різними ступенями порушень слуху (глухих та слабочуючих відвідувачів) [2-4].

Особливо актуальним є впровадження таких технологій для забезпечення доступності музейного контенту для осіб з порушеннями слуху. Люди з вадами слуху часто стикаються з труднощами під час відвідування музеїв, оскільки значна частина інформації може подаватися через аудіогіди, усні пояснення екскурсоводів або мультимедійні матеріали зі звуковим супроводом. У таких умовах виникає потреба у створенні альтернативних способів подання інформації, які б забезпечували рівний доступ до культурного контенту [2-4].

Проблема доступності музейного простору для осіб з порушеннями слуху полягає в тому, що основний масив додаткової емоційної, контекстуальної та фактичної інформації (екскурсії з гідом, аудіосупровід, мультимедійні інсталяції з голосовим озвученням, фонові історичні звуки) залишається для них абсолютно недосяжним. Заміна аудіогідів звичайними роздрукованими текстами не вирішує проблеми повною мірою. По-перше, текстові альтернативи не завжди здатні передати динаміку розповіді та емоційне забарвлення. По-друге, необхідно враховувати соціолінгвістичні особливості цієї цільової аудиторії: для багатьох людей з глибоким порушенням слуху (особливо тих, хто має порушення з народження) першою і рідною мовою є жестова мова. Словесна мова (читання

текстів) сприймається ними як іноземна, оскільки граматика, синтаксис та структура жестової мови кардинально відрізняються від лінійного тексту. Читання довгих наукових описів викликає підвищене когнітивне навантаження і призводить до швидкої втоми, що знижує загальну задоволеність відвідуванням музею [2, 10].

Відповідно, виникає гостра потреба у впровадженні інноваційних та інклюзивних технологій, які б базувалися на принципах універсального дизайну та компенсували відсутність аудіального каналу сприйняття через посилення та адаптацію візуального. Технології доповненої реальності (Augmented Reality, AR) пропонують принципово новий підхід до вирішення цієї комплексної проблеми. AR — це технологія, що дозволяє накладати комп'ютерно-згенеровані візуальні об'єкти (динамічний текст, двовимірну графіку, тривимірні 3D-моделі, складні анімації) на зображення реального світу, отримане з камери пристрою, в режимі реального часу і з урахуванням просторової геометрії середовища [5,6].

Мобільні застосунки з використанням технологій доповненої реальності можуть забезпечити подання інформації у зручному візуальному форматі. До таких форматів належать текстові описи експонатів, інтерактивні зображення, тривимірні моделі, відеоматеріали із субтитрами або перекладом жестовою мовою. Це дозволяє не лише покращити доступність музейного контенту, але й підвищити рівень залученості відвідувачів та сприяти більш глибокому розумінню представлених експонатів [7].

У контексті створення інклюзивного музейного простору використання технологій доповненої реальності дає змогу реалізувати наступні концептуальні рішення:

– динамічні просторові субтитри (Spatial Subtitles). Текст, який генерується у відповідь на запит користувача до конкретного експоната, відображається не просто у вигляді плоского списку на екрані мобільного пристрою, а "прив'язується" до фізичного об'єкта (картини, скульптури, артефакту) у тривимірному просторі [8]. Це дозволяє користувачу одночасно розглядати дрібні деталі експоната і читати пояснення поруч із ним, не перемикаючи постійно фокус зору між телефоном та об'єктом;

– візуалізація жестової мови за допомогою 3D-аватарів. Найбільш перспективним напрямком є інтеграція в AR-середовище анімованих 3D-аватарів, які перекладають згенерований текст на національну жестову мову у доповненій реальності [9]. Віртуальний гід ніби "стоїть" поруч з експонатом і розповідає його історію рідною для користувача мовою жестів, що повністю знімає когнітивний бар'єр читання словесних текстів;

– контекстне візуальне доповнення замість звукових ефектів. Заміна аудіальних акцентів відповідними візуальними ефектами навколо експоната. Наприклад, якщо експонат супроводжується звуком биття серця або пострілу гармати, AR-застосунок може генерувати пульсуюче світло або візуальні хвилі, імітуючи ці події для користувача з порушеннями слуху [10].

Одним із ефективних рішень є використання мобільних застосунків із підтримкою доповненої реальності, які можуть надавати інформацію у візуальній формі. Зокрема, це можуть бути текстові пояснення, субтитри до відеоматеріалів, графічні елементи, анімації, тривимірні реконструкції об’єктів або переклад інформації жестовою мовою у вигляді відео. Такі рішення дозволяють не лише підвищити доступність музейного середовища, але й створити інклюзивний простір, де всі відвідувачі мають можливість отримувати інформацію у зручній для них формі [11].

З технічної точки зору предметна область охоплює використання технологій комп’ютерного зору, мобільної розробки та доповненої реальності. Основними складовими таких систем є модулі розпізнавання маркерів або об’єктів, відображення тривимірних моделей, інтерактивний інтерфейс користувача, а також база даних музейного контенту. Важливим аспектом є також забезпечення зручності використання застосунку, зокрема адаптація інтерфейсу до потреб користувачів з порушеннями слуху, використання зрозумілих візуальних елементів та можливість доступу до інформації без аудіосупроводу [5, 6, 24].

Таким чином, використання AR-технологій перетворює звичайний смартфон користувача на потужний персональний інклюзивний інтерфейс. Цей інтерфейс адаптує навколишнє середовище під індивідуальні потреби людини без необхідності

фізичної перебудови самого музею, заміни табличок чи найму додаткового персоналу супроводу.

1.2 Аналіз існуючих рішень

У сучасних умовах стрімкого розвитку інформаційних технологій музеї активно впроваджують цифрові інструменти для покращення взаємодії з відвідувачами та підвищення доступності культурної спадщини. Особливого поширення набувають мобільні застосунки, які використовують технології доповненої реальності, комп'ютерного зору та мультимедійної візуалізації для інтерактивного представлення музейного контенту. Такі системи дозволяють значно розширити можливості традиційних експозицій, надаючи користувачам додаткову інформацію у вигляді тривимірних моделей, анімацій, текстових пояснень та інтерактивних елементів. Крім того, цифрові рішення відкривають нові можливості для створення інклюзивного музейного середовища, зокрема для осіб з порушеннями слуху, які потребують альтернативних візуальних способів отримання інформації [2-4]. Тому проведено аналіз існуючих програмних рішень та платформ, що використовуються для візуалізації музейного контенту із застосуванням технологій доповненої реальності, а також оцінка їх функціональних можливостей, переваг і обмежень.

Smartify — це цифрова платформа та мобільний застосунок, призначений для інтерактивного ознайомлення з музейними експонатами [12]. Основною функцією системи є розпізнавання об'єктів мистецтва за допомогою камери смартфона та надання користувачеві мультимедійної інформації про них. Платформа використовується музеями у різних країнах світу як цифровий гід для відвідувачів. Працює як мобільний додаток і веб-платформа з 2017 року.

Smartify з'явився як інструмент для об'єднання музеїв і публіки за допомогою технології розпізнавання зображень. Компанію заснувала група культурних менеджерів і розробників, які прагнули створити універсальний цифровий гід, доступний у різних інституціях без потреби завантажувати окремі додатки.

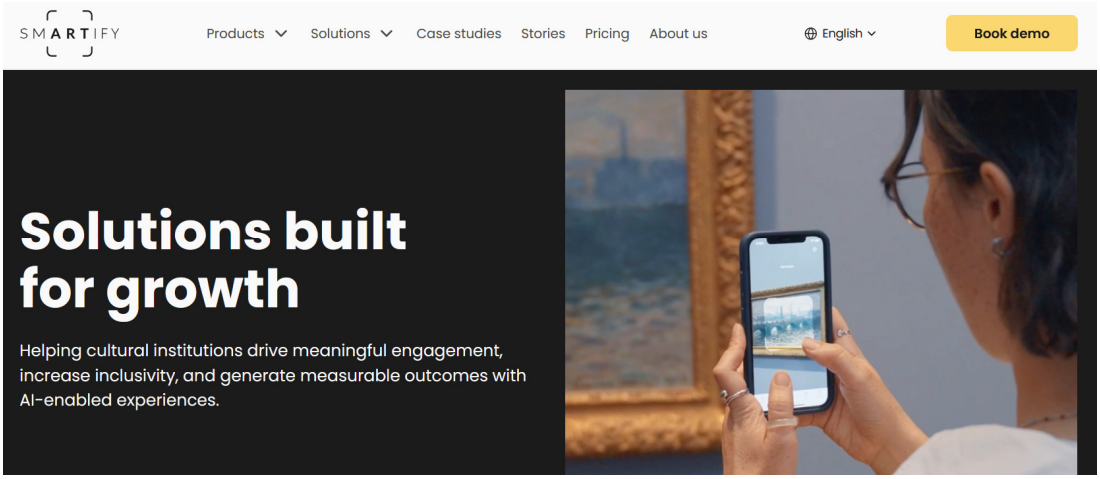

Рисунок 1.1 - Цифрова платформа та мобільний застосунок Smartify.

Застосунок дозволяє користувачам сканувати твори мистецтва під час відвідування музею або переглядати колекції онлайн. Smartify ідентифікує об’єкт і надає контент — текстові пояснення, аудіокоментарі, відео чи віртуальні тури. Платформа також дозволяє зберігати улюблені твори у власній цифровій колекції.

Платформа співпрацює з провідними інституціями, зокрема з Лувр, Британський музей, Метрополітен-музей та Національна галерея Лондона. Smartify підтримує десятки мов і доступний у сотнях музеїв у всьому світі, розширюючи можливості доступу до мистецтва для глобальної аудиторії.

Smartify став одним із найпоширеніших цифрових рішень у музейній сфері, сприяючи демократизації культурного досвіду. Він поєднує інноваційні технології з освітньою місією, дозволяючи інституціям збирати аналітику відвідувань і підтримувати зв’язок з публікою навіть поза межами виставкових залів.

Система Smartify використовує комплекс сучасних технологій:

– комп’ютерний зір для розпізнавання музейних експонатів;

– хмарну інфраструктуру для зберігання та обробки даних;

– WebXR та технології доповненої реальності для створення інтерактивних експозицій;

– систему управління контентом для оновлення інформації працівниками музею.

AR-модулі Smartify дозволяють відтворювати 3D-моделі, інтерактивні сцени та реконструкції історичних подій безпосередньо у просторі музею.

До основних функцій системи належать:

– розпізнавання експонатів за допомогою камери смартфона;

– відображення текстових описів та мультимедійного контенту;

– створення інтерактивних AR-експозицій;

– підтримка мультимовних турів;

– використання субтитрів та текстових транскрипцій для підвищення доступності.

Перевагами системи є: велика база музейних експонатів, інтеграція технологій комп'ютерного зору, підтримка AR та XR-технологій, можливість доступу через браузер без встановлення застосунку. Недоліками є: значна залежність від якості інтернет-з'єднання, обмежені спеціалізовані функції для людей з порушеннями слуху (наприклад, переклад жестовою мовою), необхідність створення цифрових моделей експонатів для AR-контенту.

Museum Vision AR — це програмна платформа, призначена для створення інтерактивних музейних експозицій із використанням технологій доповненої реальності [13]. Основна ідея системи полягає у накладенні цифрового контенту на реальні музейні об'єкти, що дозволяє доповнити фізичні експонати додатковою інформацією та інтерактивними елементами. Вона надає інтерактивний спосіб представлення експонатів через смартфони або AR-окуляри, збагачуючи досвід відвідувачів цифровими анотаціями, 3D-моделями та мультимедійним контентом (рисунок 1.2).

Рисунок 1.2 – Можливості програмної платформи Museum Vision AR.

Платформа поєднує фізичний простір музею з цифровими елементами через камеру пристрою. Відвідувачі можуть навести смартфон на експонат і побачити накладену інформацію, анімації чи реконструкції. Використовуються технології комп'ютерного бачення, геолокації та маркерів AR для точного розпізнавання об'єктів.

Музеї використовують Museum Vision AR для створення інтерактивних маршрутів, ігор або освітніх турів. Платформа допомагає залучити молодшу аудиторію, покращити доступність і пропонувати багатомовний контент без змін у фізичній експозиції [13].

AR-рішення такого типу змінюють спосіб взаємодії публіки з культурною спадщиною. Museum Vision AR сприяє цифровій трансформації музеїв, підтримуючи збереження колекцій і підвищення їх привабливості через інноваційні технології.

Типова архітектура музейних AR-систем включає:

- мобільний застосунок для відображення AR-контенту;
- веб-портал для редагування та управління контентом;
- систему позиціонування AR-елементів у просторі;
- базу мультимедійних даних.

Спочатку виконується цифрове сканування музейного простору, після чого куратори розміщують у віртуальному середовищі AR-об'єкти, що пов'язані з експонатами. Основні можливості системи:

- відображення 3D-об'єктів поверх музейних експонатів;
- інтеграція відео, аудіо та текстових матеріалів;
- використання AR-іконок та інтерактивних елементів;
- дистанційне оновлення інформації кураторами;
- можливість інтеграції мультимедійного контенту та перекладів.

Перевагами Museum Vision AR є: високий рівень інтерактивності, підтримка різних мультимедійних форматів, можливість швидкого оновлення контенту, зменшення потреби у фізичних інформаційних табличках. До недоліків належать: складність підготовки AR-контенту, потреба у професійній 3D-моделі експонатів, високі технічні вимоги до мобільних пристроїв.

MuVision — це мобільний застосунок для ознайомлення з музейними експозиціями та культурними об’єктами, який використовує технології доповненої реальності для покращення взаємодії відвідувачів з експонатами [14]. Застосунок дозволяє користувачам знаходити музеї, планувати відвідування та отримувати AR-екскурсії під час огляду експозицій. Він поєднує технології розпізнавання зображень, 3D-візуалізації та мультимедіа, щоб покращити взаємодію відвідувачів із музейними експонатами та виставковими просторами. Підтримується на iOS та Android.

MuVision використовує камеру смартфона для розпізнавання експонатів і накладання цифрового контенту у реальному часі — від історичних реконструкцій до анімованих об’єктів і звукових коментарів (рисунок 1.3). Система також пропонує навігацію по залах, персоналізовані тури та інтеграцію з музейними базами даних.

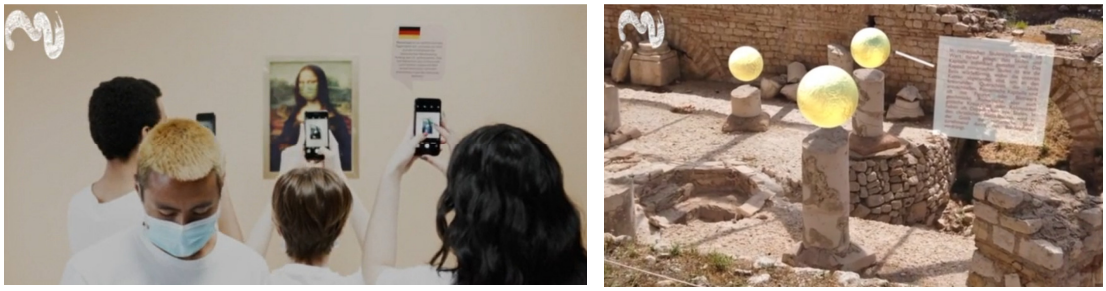

Рисунок 1.3 — Приклади використання мобільного застосунку MuVision

Застосунок побудований на рушіях доповненої реальності, таких як ARKit або ARCore, і використовує алгоритми машинного зору для точного позиціонування контенту. 3D-моделі й аудіо-відео ресурси завантажуються локально або через хмару, що дозволяє гнучко оновлювати виставкові матеріали [15].

MuVision допомагає культурним установам модернізувати досвід відвідувачів, залучаючи молодшу аудиторію та підтримуючи інклюзивність через мультимовний і мультимедійний контент. Завдяки інтерактивним елементам відвідувачі можуть краще зрозуміти контекст експонатів і зберегти інтерес до колекцій.

Система містить базу даних культурних об’єктів і підтримує роботу з великою кількістю музейних установ. Основні технології системи:

- мобільні AR-інтерфейси;
- база культурних об’єктів;
- інтерактивні екскурсії;
- геолокаційні сервіси.

Застосунок може працювати як на особистих смартфонах відвідувачів, так і на спеціальних пристроях, які надаються музеями. Система забезпечує:

- AR-екскурсії музейними експозиціями;
- інтерактивні підказки біля експонатів;
- планування відвідування музеїв;
- створення персональних списків об’єктів;
- відображення інформації про події та виставки.

Перевагами є:

- поєднання AR-екскурсій та туристичного планування;
- доступ до великої кількості культурних об’єктів;
- зручний мобільний інтерфейс.

До недоліків відносять:

- відсутність спеціалізованих функцій для людей з порушеннями слуху;
- залежність від мобільного інтернету;
- обмежена кількість інтерактивних функцій у деяких музеях.

Проведений аналіз показує, що сучасні музейні цифрові системи активно використовують технології доповненої реальності та мобільні застосунки для підвищення інтерактивності експозицій. Водночас більшість існуючих рішень орієнтована на загальну аудиторію та недостатньо враховує потреби людей з порушеннями слуху. Це обґрунтовує доцільність розробки спеціалізованого мобільного застосунку з доповненою реальністю, який забезпечить візуальне представлення інформації, субтитри та можливу інтеграцію жестової мови, що сприятиме підвищенню доступності музейного середовища.

1.2.1 Аналіз підходів до реалізації AR

Реалізація інклюзивного мобільного застосунку вимагає вибору оптимального алгоритму просторового відстеження (Tracking) для накладання віртуального контенту на фізичні експонати. Враховуючи специфіку музейного середовища (змінне освітлення, наявність скляних вітрин, велике скупчення людей), ефективність роботи AR-системи критично залежить від обраного методу комп'ютерного зору [15]. Сучасні AR-технології поділяються на три основні категорії:

1. Location-based AR (Геолокаційна доповнена реальність). Використовує дані GPS, компаса та акселерометра для відображення об'єктів за географічними координатами. Недоліки для предметної області: Сигнал GPS вкрай нестабільний всередині масивних музейних будівель, а похибка у кілька метрів є неприпустимою для точної прив'язки субтитрів до конкретної невеликої картини чи скульптури [16, 17].

2. Markerless AR / SLAM (Доповнена реальність без маркерів). Базується на алгоритмі одночасної локалізації та побудови карти (Simultaneous Localization and Mapping). Додаток аналізує геометрію простору, шукає горизонтальні чи вертикальні площини (підлогу, стіни) і дозволяє розмістити на них 3D-об'єкт. Недоліки для предметної області: Метод є дуже ресурсомістким, швидко виснажує батарею мобільного пристрою та потребує від користувача попереднього сканування простору ("водіння" камерою по кімнаті), що знижує загальну зручність інклюзивного інтерфейсу. Крім того, SLAM не розпізнає який саме це експонат, він лише бачить площину [7, 18].

3. Marker-based AR / Image Tracking (Відстеження зображень-маркерів). Метод використовує алгоритми комп'ютерного зору для пошуку заздалегідь визначених 2D-зображень (маркерів) у відеопотоці з камери. Як маркер може виступати сама картина, інформаційна табличка або спеціальний візуальний патерн. Переваги для предметної області: Цей метод забезпечує найвищу точність позиціонування контенту безпосередньо біля об'єкта. Алгоритми вилучення ключових точок (Feature Extraction) стабільно працюють навіть при частковому перекритті експоната або зміні кута огляду. Оскільки візуальні характеристики

експонатів є статичними, Image Tracking дозволяє миттєво ідентифікувати об'єкт та закріпити просторовий якір (AR Anchor) для виведення динамічних субтитрів [8, 9].

Проведений аналіз методів показує, що для реалізації мобільного застосунку найбільш доцільним є використання маркерного підходу (Image Tracking). Він забезпечує оптимальний баланс між продуктивністю мобільного пристрою смартфона, швидкістю розпізнавання та точністю накладання інклюзивного візуального контенту.

1.2.2 Особливості проєктування інклюзивних AR-інтерфейсів

Для успішного проєктування архітектури та інтерфейсу мобільного застосунку необхідно враховувати не лише технологічні, але й когнітивні особливості цільової аудиторії. Класичні підходи до розробки мобільних інтерфейсів не завжди є релевантними для осіб з частковою або повною втратою слуху [4].

Згідно з дослідженнями у сфері дефектології та інклюзивного дизайну, особи з порушеннями слуху компенсують дефіцит аудіальної інформації за рахунок гіперконцентрації зорової уваги. Однак це призводить до швидшого виснаження нервової системи та підвищеного когнітивного навантаження під час обробки складних візуальних стимулів. У контексті взаємодії з AR-додатком у музеї виділяють наступні критичні фактори:

1. Проблема білінгвізму та бар'єр читання. Для багатьох людей із вродженою глухотою рідною мовою є національна жестова мова, яка має власну морфологію, синтаксис та просторову граматику. Словесна мова (читання тексту) сприймається ними як друга (іноземна) мова . Тому подання інформації у вигляді суцільних довгих абзаців (наукових описів експонатів) є неефективним. Це формує вимогу до розробки алгоритмів порційної видачі тексту (Chunking) та генерації спрощених, синтаксично прозорих речень, особливо для дитячої аудиторії.

2. Розподіл зорової уваги. Під час екскурсії відвідувач має одночасно розглядати фізичний експонат і читати пояснення. Якщо інформація подається на екрані смартфона внизу, а експонат знаходиться на стіні, постійне переведення погляду (зміна фокусної відстані ока) викликає швидку втому. Технологія

просторових субтитрів (Spatial Subtitles) вирішує цю проблему, оскільки віртуальний текст оптично розміщується в одній фокусній площині з фізичним об'єктом [3].

3. Сприйняття динамічного фону. На відміну від звичайних додатків, де текст відображається на однотонному тлі, в AR-застосунках фоном слугує реальне середовище (відеопотік з камери), яке є строкатим і постійно змінюється. Це висуває жорсткі вимоги до інтерфейсу: використання висококонтрастних шрифтів без зарубок, наявність напівпрозорих підкладок (Backplates) та уникнення дрібних деталей, які можуть зливатися з фоном [19].

4. Відсутність звукових системних сповіщень. Усі процеси (успішне сканування експоната, завантаження даних, виникнення помилок) мають супроводжуватися виключно візуальними індикаторами або тактильним зворотним зв'язком (вібрацією смартфона), щоб користувач розумів поточний стан системи [4]. Врахування вищезазначених психофізіологічних та когнітивних особливостей є обов'язковою умовою при проєктуванні архітектури взаємодії клієнтського застосунку, побудові станів Riverpod та розробці алгоритмів візуалізації контенту.

1.2.3 Порівняльний аналіз існуючих рішень

Проведений аналіз сучасних програмних платформ для візуалізації музейного контенту показав, що більшість існуючих рішень використовують технології доповненої реальності, комп’ютерного зору та мультимедійного представлення інформації для підвищення інтерактивності експозицій. Разом із тим функціональні можливості таких систем суттєво відрізняються за рівнем підтримки інклюзивності, способами відображення контенту та механізмами взаємодії з користувачем.

Для узагальнення результатів аналізу та визначення основних переваг і недоліків розглянутих платформ виконано їх порівняння за ключовими характеристиками, результати якого наведено в таблиці 1.1.

Як видно з таблиці 1.1, розглянуті програмні рішення забезпечують високий рівень інтерактивності та підтримують технології доповненої реальності, однак не повною мірою враховують потреби осіб з порушеннями слуху. Зокрема, у проаналізованих системах відсутні механізми просторового відображення

субтитрів, адаптивного формування контенту та його персоналізації відповідно до особливостей користувача.

Таблиця 1.1 – Порівняльний аналіз існуючих систем

Характеристика	Smartify	Museum Vision AR	MuVision	Запропонований застосунок
Розпізнавання експонатів	Так	Частково	Так	Так
Доповнена реальність	Так	Так	Так	Так
Просторові субтитри	Ні	Ні	Ні	Так
Адаптація для осіб з порушеннями слуху	Частково	Частково	Ні	Так
Генерація контенту за допомогою ШІ	Ні	Ні	Ні	Так
Персоналізація контенту	Ні	Ні	Частково	Так
Кросплатформність	Так	Так	Так	Так

Виявлені обмеження підтверджують доцільність розробки спеціалізованого мобільного застосунку, який поєднуватиме технології доповненої реальності, розпізнавання експонатів та інтелектуальну генерацію адаптованого контенту для створення інклюзивного музейного середовища.

1.3 Постановка задачі дослідження

У сучасному цифровому суспільстві важливим завданням є забезпечення рівного доступу до культурної спадщини для всіх категорій населення, зокрема для осіб з інвалідністю. Однією з актуальних проблем музейної сфери є недостатній рівень доступності експозицій для людей з порушеннями слуху, оскільки значна частина інформації під час екскурсій подається у звуковій формі або через усні пояснення екскурсоводів. У зв’язку з цим виникає потреба у впровадженні сучасних

інформаційних технологій, які забезпечують альтернативні способи подання інформації та сприяють створенню інклюзивного музейного середовища.

Одним із перспективних напрямів вирішення зазначеної проблеми є використання мобільних застосунків із підтримкою технологій доповненої реальності. Такі застосунки дозволяють поєднувати реальні музейні експонати з цифровими об'єктами, забезпечуючи візуалізацію додаткової інформації у вигляді текстових описів, графічних елементів, анімацій або тривимірних моделей. Використання доповненої реальності створює нові можливості для інтерактивної взаємодії користувача з музейним контентом та сприяє більш ефективному засвоєнню інформації.

Незважаючи на наявність різноманітних цифрових музейних платформ та мобільних застосунків, більшість із них орієнтована на загальну аудиторію та не враховує специфічні потреби осіб з порушеннями слуху. Зокрема, у таких системах недостатньо реалізовані механізми візуального подання інформації, відсутні спеціальні інтерфейсні рішення або інтеграція елементів, що забезпечують зручне сприйняття контенту без використання аудіо.

На основі проведеного аналізу предметної області, існуючих програмних рішень та особливостей цільової аудиторії встановлено необхідність розробки спеціалізованого мобільного застосунку з використанням технологій доповненої реальності для візуалізації музейного контенту. Такий застосунок повинен забезпечувати розпізнавання експонатів, просторове відображення інформації, адаптацію контенту до потреб користувачів та підтримку принципів інклюзивного дизайну.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

– проаналізувати сучасні технології та програмні рішення, що використовуються для створення мобільних застосунків з доповненою реальністю у музейній сфері, а також визначити їхні переваги та недоліки з точки зору доступності для осіб з порушеннями слуху;

– дослідити особливості подання інформації для користувачів з порушеннями слуху та сформувати вимоги до інтерфейсу мобільного застосунку, що забезпечує зручне та зрозуміле сприйняття музейного контенту;

– розробити концепцію та архітектуру мобільного застосунку для візуалізації музейного контенту з використанням технологій доповненої реальності;

– обрати програмні засоби та технологічні платформи для реалізації мобільного застосунку, зокрема інструменти розробки AR-контенту, бібліотеки комп’ютерного зору та засоби мобільної розробки;

– реалізувати програмний модуль розпізнавання музейних експонатів та відображення доповненого контенту у вигляді текстових пояснень, графічних елементів або тривимірних моделей;

– розробити користувацький інтерфейс мобільного застосунку з урахуванням принципів доступності та інклюзивного дизайну;

– провести тестування розробленого застосунку та оцінити ефективність використання технологій доповненої реальності для візуалізації музейного контенту.

Таким чином, результатом виконання роботи має стати мобільний застосунок з підтримкою технологій доповненої реальності, який забезпечує розпізнавання музейних експонатів, просторове відображення адаптованого контенту та підвищення доступності музейного середовища для осіб з порушеннями слуху. Реалізація такого рішення сприятиме створенню більш інклюзивного культурного простору та розширенню можливостей використання цифрових технологій у музейній сфері.

2 ПРОЄКТУВАННЯ АРХІТЕКТУРИ ТА ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

2.1 Архітектура програмної системи

Загальна структура мобільного застосунку складається з кількох взаємопов'язаних модулів, кожен з яких виконує окрему функцію в процесі обробки інформації та відображення доповненого контенту. Основними модулями є: модуль користувацького інтерфейсу, модуль доповненої реальності, модуль розпізнавання об'єктів, модуль управління контентом, модуль доступності, модуль навігації та взаємодії з музеєм та модуль обробки даних (Рисунок 2.1).

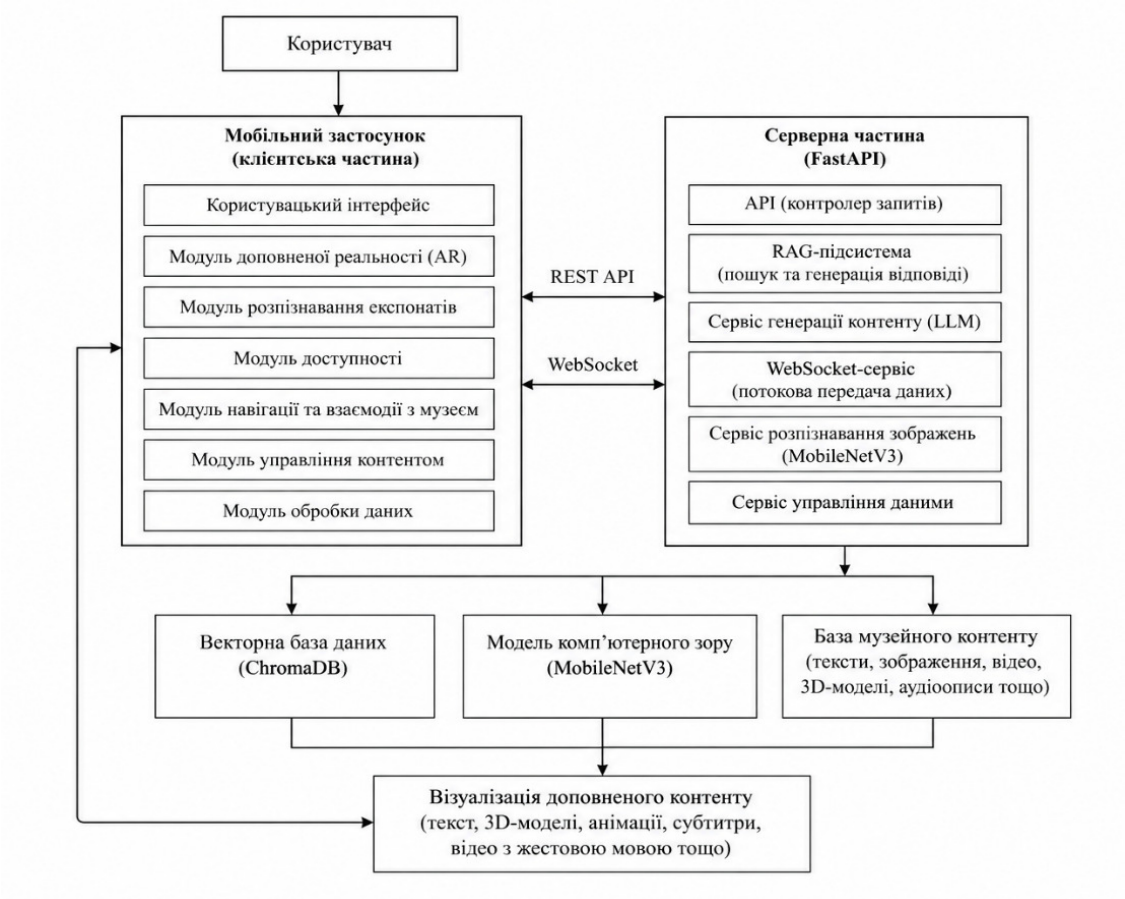

Рисунок 2.1 – Загальна архітектура програмної системи

Модуль користувацького інтерфейсу забезпечує взаємодію користувача із системою. Він відповідає за відображення елементів керування, навігацію між розділами застосунку та представлення інформації про музейні експонати.

Особливістю інтерфейсу є його адаптація до потреб осіб з порушеннями слуху, що передбачає використання зрозумілих візуальних елементів, текстових пояснень, піктограм та можливість відображення відеоматеріалів із жестовою мовою або субтитрами [20].

Модуль доповненої реальності є ключовим компонентом системи. Його основною функцією є накладення цифрового контенту на зображення реального середовища, яке отримується з камери мобільного пристрою [21]. Після розпізнавання експоната система відображає додаткову інформацію у вигляді текстових описів, тривимірних моделей, графічних елементів або анімацій. Це дозволяє створити інтерактивний та наочний спосіб ознайомлення з музейними об’єктами.

Модуль розпізнавання об’єктів відповідає за ідентифікацію музейних експонатів за допомогою технологій комп’ютерного зору [20]. Камера мобільного пристрою фіксує зображення експоната, після чого система аналізує його та порівнює з еталонними зображеннями, що зберігаються у базі даних. У разі успішного розпізнавання експоната застосунок отримує відповідний інформаційний контент та передає його до модуля доповненої реальності для подальшого відображення.

Модуль управління контентом відповідає за зберігання, обробку та передачу інформації про музейні експонати [3]. У базі даних системи можуть міститися текстові описи, зображення, відеоматеріали, тривимірні моделі та інші мультимедійні ресурси. Цей модуль забезпечує швидкий доступ до необхідної інформації після розпізнавання експоната.

З огляду на орієнтацію застосунку на осіб з порушеннями слуху важливим компонентом є модуль забезпечення доступності [150]. Він відповідає за адаптацію контенту для користувачів з особливими потребами. До його функцій можуть належати відображення субтитрів, текстових пояснень, візуальних підказок, а також інтеграція відеоматеріалів із перекладом жестовою мовою. Крім того, модуль може передбачати можливість налаштування інтерфейсу для покращення сприйняття інформації [4].

Модуль навігації та взаємодії з музеєм забезпечує користувачам можливість орієнтуватися у музейному просторі, отримувати інформацію про розташування експонатів та переглядати тематичні маршрути або інтерактивні екскурсії. Завдяки цьому відвідувачі можуть ефективніше планувати огляд експозицій. Модуль обробки даних відповідає за взаємодію між усіма компонентами системи. Він забезпечує обробку запитів користувача, передачу інформації між модулями, а також оптимізацію роботи застосунку.

Таким чином, загальна структура мобільного застосунку для візуалізації музейного контенту з використанням технологій доповненої реальності складається з комплексу функціональних модулів, що забезпечують розпізнавання експонатів, відображення доповненого контенту, управління інформаційними ресурсами та адаптацію інтерфейсу для осіб з порушеннями слуху. Взаємодія цих компонентів дозволяє створити інтерактивне та інклюзивне середовище для ознайомлення відвідувачів з музейними експозиціями.

2.2 Архітектура клієнтської частини

Проєктування мобільного застосунку для візуалізації музейного контенту вимагає створення надійної та масштабованої клієнт-серверної архітектури. Оскільки основна обчислювальна логіка (генерація відповідей за допомогою LLM та RAG-архітектури) винесена на сторону сервера [18, 22], мобільний клієнт, розроблений на базі фреймворку Flutter [23, 24], виконує роль інтелектуального інтерфейсу (Thin Client з елементами Edge Computing для обробки AR [15]). На рисунку 2.2 представлена архітектура управління станами за допомогою рішення Riverpod [25].

Основою архітектури додатка є підхід реактивного управління станом Riverpod [25], який забезпечує безпеку типів та строге розділення шару представлення (UI) від бізнес-логіки та шару доступу до даних. Архітектура клієнтської частини складається з трьох основних рівнів:

– Data Layer (Рівень даних). Відповідає за комунікацію із зовнішніми API, отримання потокових даних від RAG-сервера та завантаження баз даних маркерів експонатів для AR-модуля [26-27];

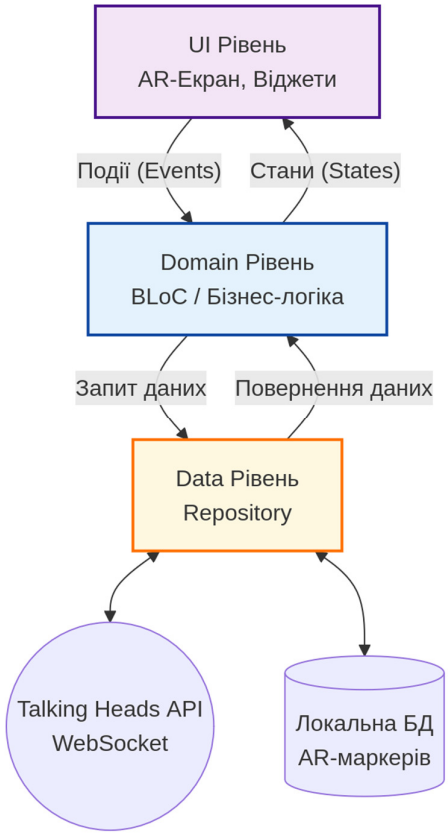

Рисунок 2.2 – Архітектура управління станами на базі Riverpod

– Domain Layer (Рівень бізнес-логіки). Містить класи Notifier та AsyncNotifier, які інкапсулюють бізнес-логіку, обробляють запити від інтерфейсу користувача та реактивно оновлюють стани додатка через механізм AsyncValue (наприклад, стан отримання потоку токенів TextStreamingState або ErrorState);

– Presentation Layer (Рівень представлення). Віджети Flutter (зокрема ConsumerWidget), які реактивно перебудовуються при зміні значень провайдерів Riverpod, відображаючи AR-камеру, згенеровані субтитри або елементи управління [28].

Ключовим завданням на етапі проєктування є організація взаємодії з RAG-сервером [20, 22]. Як було встановлено, сервер генерує відповіді із середньою затримкою (Latency) понад 14 секунд. Використання класичного синхронного

HTTP-запиту (REST API) призвело б до блокування інтерфейсу або тривалого очікування користувачем "порожнього" екрану, що є неприпустимим з точки зору UX.

Для вирішення цієї проблеми спроєктовано використання протоколу Server-Sent Events (SSE) або WebSocket [28]. Процес взаємодії виглядає наступним чином:

– користувач через AR-камеру наводить пристрій на експонат. Локальний модуль розпізнає ID експоната;

– мобільний додаток формує об'єкт запиту (JSON), який містить: exhibit_id, текстовий запит користувача (або автоматичний запит на вітання) та параметр audience_mode ("adults" або "kids");

– запит відправляється на сервер, відкриваючи асинхронне з’єднання;

– сервер починає генерацію відповіді і по мірі готовності відправляє дані у вигляді потоку токенів (chunks) [18];

– Data Layer мобільного додатка приймає цей потік і передає його у Domain Layer, де відбувається локальна буферизація та агрегація тексту. Інтерфейс (AR-середовище) миттєво реагує на надходження перших слів, розпочинаючи візуалізацію субтитрів вже на 1-2 секунді після запиту.

Передача параметра audience_mode є критичною для забезпечення інклюзивності. Якщо користувач у налаштуваннях профілю обрав дитячий режим, серверна LLM активує відповідний системний промпт, генеруючи короткі речення з емодзі, що значно полегшує когнітивне навантаження на дитину з порушеннями слуху під час читання просторових субтитрів [5].

2.3 Проєктування інклюзивного інтерфейсу користувача

Проєктування інтерфейсу користувача (UI) для AR-додатка, орієнтованого на людей з порушеннями слуху, підпорядковується жорстким правилам цифрової інклюзії та доступності (Accessibility) [2]. Оскільки камера смартфона постійно передає динамічний та непередбачуваний фон (темні картини, світлі стіни, інші відвідувачі), класичні правила мобільного дизайну потребують адаптації [4]. Основними принципами розробленого інклюзивного інтерфейсу є:

– висококонтрастна типографіка в AR. Текст субтитрів (AR-вузли) рендериться з обов'язковим урахуванням контрастності. Використовуються гротескні шрифти без зарубок (Sans-Serif, наприклад, Roboto або San Francisco), які найлегше зчитуються з екрану. Кожна літера повинна мати темну обводку (Stroke) або розташовуватися на напівпрозорій темній підкладці (Backplate) з ефектом розмиття фону (Glassmorphism);

– управління фокусом уваги (Focus Management). Оскільки особи з порушеннями слуху компенсують відсутність звуку підвищеною візуальною увагою, інтерфейс не повинен містити зайвого "візуального шуму". Екран поділяється на дві зони: центральна зона фокусу та периферійна зона;

– візуальна індикація системних станів. Традиційно додатки можуть повідомляти про помилки або процеси за допомогою звукових сигналів. У даному застосунку всі системні події конвертуються у чіткі візуальні маркери;

– маркування AI-контенту: Згідно з етичними стандартами використання LLM та з огляду на можливість "галюцинацій", інтерфейс містить спеціальний бейдж (іконку), який вказує, що текст згенерований штучним інтелектом [2].

Таким чином, спроєктований UI/UX-інтерфейс нівелює труднощі сприйняття візуальної інформації на строкатому фоні та створює максимально комфортне інклюзивне середовище для повноцінного занурення користувача в історичний контекст музейної експозиції [4].

2.4 Архітектура серверної підсистеми

Функціонування системи забезпечується серверною частиною, побудованою на базі мови програмування Python та асинхронного фреймворку FastAPI. Вибір даного технологічного стеку зумовлений необхідністю високої пропускної здатності та низької затримки (latency) при обробці потокових даних. Основне завдання даного рівня полягає в ідентифікації експонатів за допомогою методів глибокого навчання та забезпеченні стабільної трансляції контенту через протокол WebSocket.

Програмна реалізація сервера базується на мікросервісній логіці. Базово архітектура серверної підсистеми включає чотири ключові компоненти:

– модуль комп'ютерного зору (PyTorch). Відповідає за ідентифікацію музейних експонатів за допомогою донавченої легкої нейромережі MobileNetV3-Small [29];

– інфраструктура RAG (Retrieval-Augmented Generation). Забезпечує фактологічну точність генерації тексту шляхом семантичного пошуку інформації у локальній векторній базі даних [30];

– адаптивний генератор системних промптів. Динамічно формує інструкції для штучного інтелекту з урахуванням когнітивних потреб цільової аудиторії (дитячий або дорослий режим);

– контролер асинхронної трансляції (WebSocket). Підтримує стійке з'єднання з клієнтом та забезпечує потокову передачу згенерованих текстових токенів у реальному часі.

На рисунку 2.3 представлена внутрішня архітектура серверної частини та схема взаємодії між модулем комп'ютерного зору та генеративним RAG-ядром.

Нижче наведено детальний опис принципів функціонування кожного з компонентів системи. Модуль комп'ютерного зору та ідентифікації експонатів використосує архітектуру MobileNetV3-Small. Дана модель була обрана через її високу ефективність на серверних системах без спеціалізованих GPU-прискорювачів. Оскільки базова архітектура попередньо натренована на глобальному наборі даних ImageNet і розпізнає лише загальні класи об'єктів, для вирішення задачі ідентифікації специфічних музейних експонатів було застосовано метод трансферного навчання (Transfer Learning). Цей підхід дозволяє використати вже сформовані ваги згорткових шарів моделі, які здатні ефективно виділяти базові візуальні ознаки (контури, текстури, градієнти), адаптувавши лише фінальний класифікаційний шар під конкретну предметну область

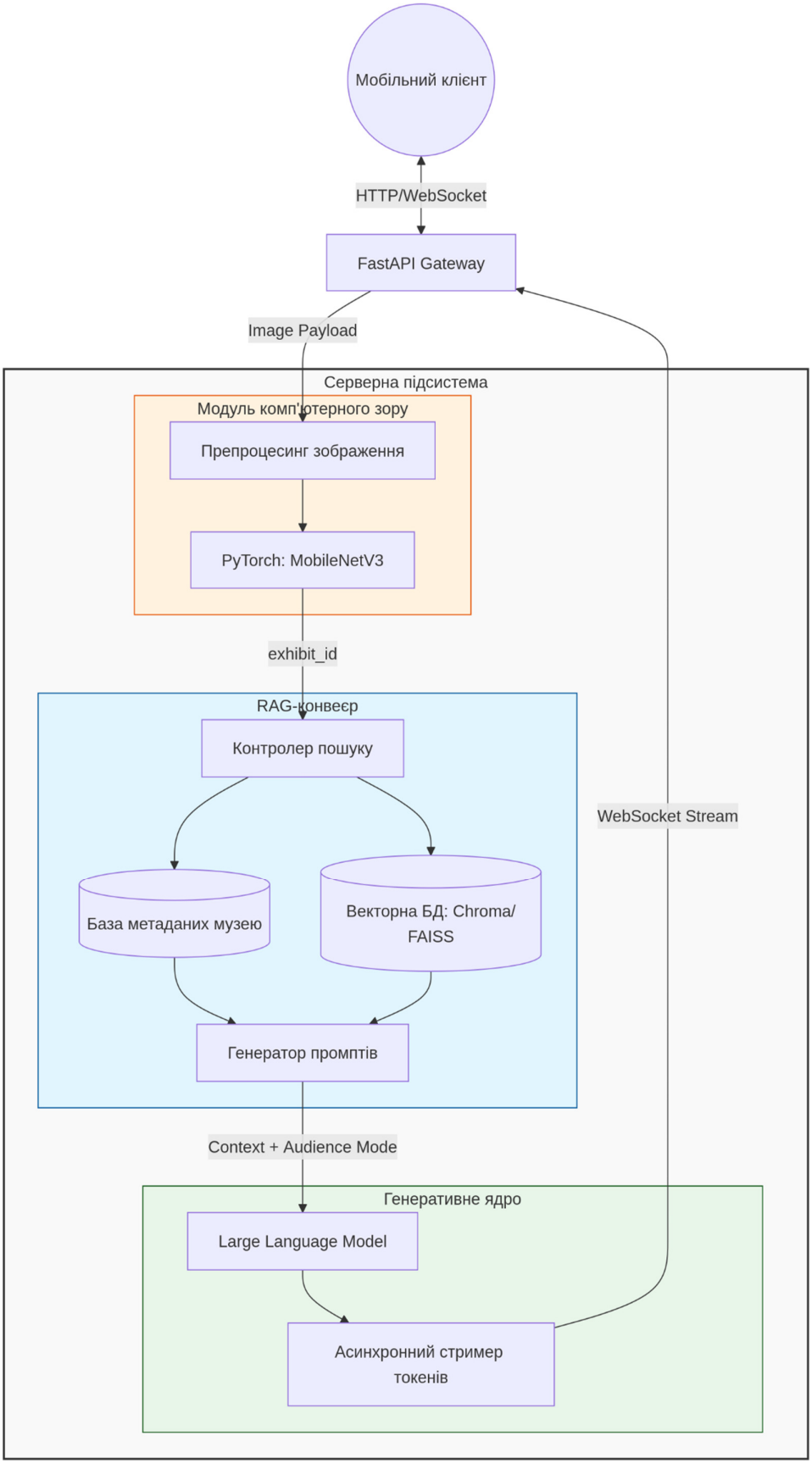

Рисунок 2.3 – Архітектура серверної підсистеми та схема обробки даних у RAG-конвеєрі

2.5 Інформаційне забезпечення та модель даних

Модуль доповненої реальності є ядром клієнтського застосунку (рисунок 2.4). Його завдання полягає у скануванні фізичного простору музею, ідентифікації цільових об'єктів та обчисленні координат для правильного накладання згенерованого візуального контенту (субтитрів). У середовищі Flutter цей модуль реалізується як міст (MethodChannel) до нативних API комп'ютерного зору: ARCore для Android та ARKit для iOS [31, 32].

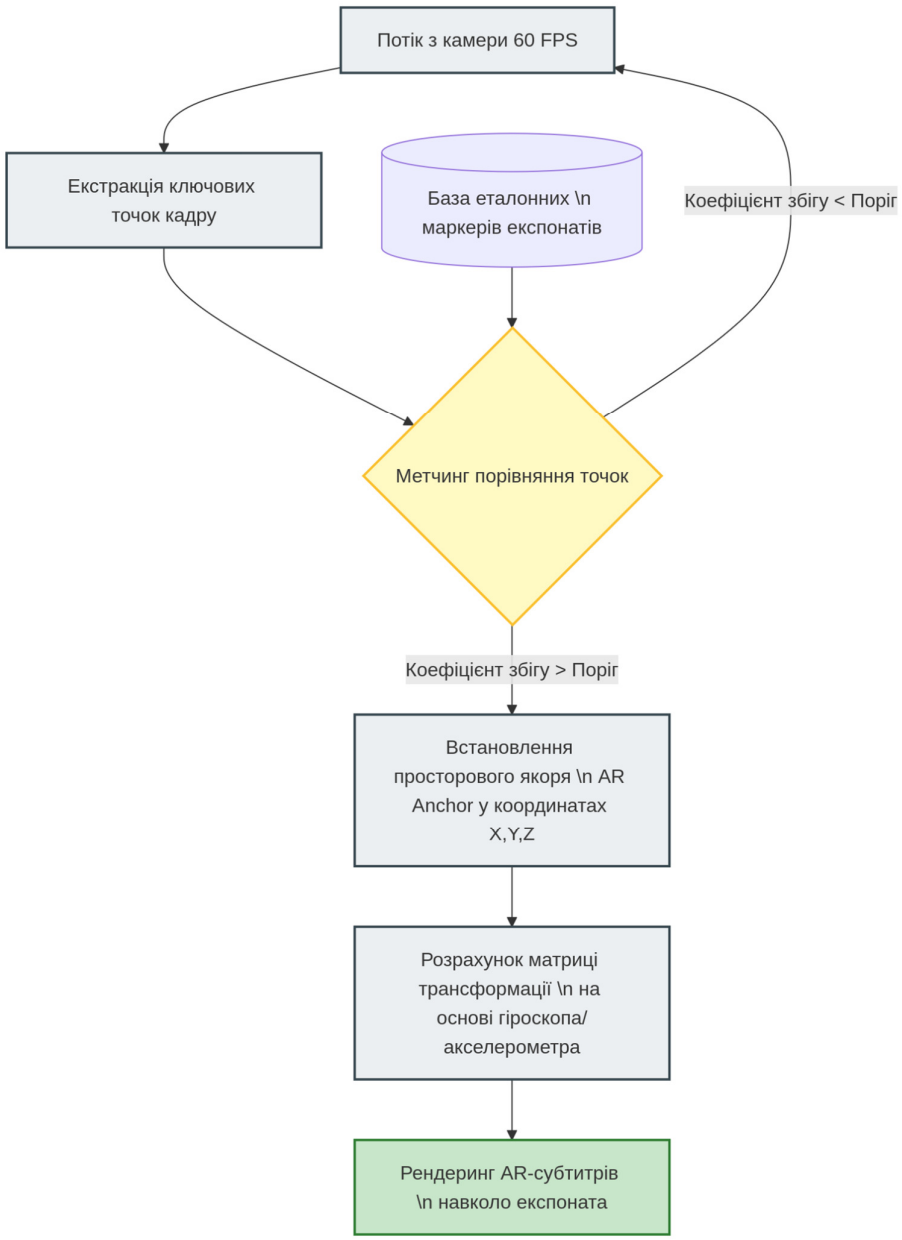

Рисунок 2.4 – Процес екстракції ключових точок та встановлення AR-якоря

Основою роботи AR-модуля у музейному просторі є алгоритм Image Tracking (Відстеження зображень-маркерів) [33]. Алгоритм працює за такими кроками:

1. Підготовка бази еталонів: На етапі ініціалізації додаток завантажує набір двовимірних зображень (фотографій картин, текстур скульптур чи інформаційних табличок експонатів), які виступають у ролі маркерів.

2. Екстракція ключових точок (Feature Extraction): Алгоритми комп'ютерного зору аналізують відеопотік з камери смартфона зі швидкістю 60 кадрів на секунду. Вони шукають у кадрі висококонтрастні ділянки.

3. Метчинг (Matching): Відбувається порівняння знайдених ключових точок з точками завантажених еталонних зображень.

4. Встановлення якоря (AR Anchor): При успішному розпізнаванні у геометричному центрі знайденого об'єкта створюється віртуальний "якір". Цей якір отримує власні координати (X, Y, Z) у тривимірному просторі відносно камери пристрою

Прив'язка візуального контенту здійснюється саме до цього якоря. Важливою архітектурною вимогою є обчислення матриці трансформації (Transformation Matrix), яка враховує рух користувача. Нативні AR-бібліотеки використовують дані акселерометра, гіроскопа та камери пристрою для реалізації механізму Visual-Inertial Odometry (VIO), який автоматично оновлює координати AR-якоря під час переміщення користувача [9]. Це створює ілюзію того, що згенеровані текстові субтитри або 3D-аватар "прикріплені" до фізичного експоната і залишаються нерухомими у просторі музею. Для підвищення стабільності відстеження в умовах нестабільного музейного освітлення, спроєктовано механізм втрати та відновлення трекінгу (Tracking State Handling).

На відміну від традиційних мобільних додатків, де текст відображається у стандартних текстових віджетах із можливістю прокручування, у середовищі доповненої реальності такий підхід є малоефективним. Виведення суцільного масиву тексту розміром 200-300 слів у AR-сцену повністю перекриє музейний експонат і зруйнує візуальний контекст. Для осіб з порушеннями слуху, які сильно залежать від візуального сприйняття простору, це створить дискомфорт. Для вирішення цього завдання розроблено локальний алгоритм візуалізації – Dynamic

Chunking & Sync (Динамічне розбиття та синхронізація). Його мета полягає у порційному виведенні тексту (по 4-7 слів), синхронізованому із середньою швидкістю читання. Алгоритм містить кілька послідовних етапів обробки:

1. Буферизація потоку: Текст, що надходить від LLM через WebSocket, накопичується у локальному буфері.

2. Синтагматичне розбиття (Chunking): Локальний алгоритм розбиває накопичений текст на логічні частини. Замість примітивного розбиття за кількістю символів, використовується базова NLP-логіка (на основі розділових знаків). Алгоритм шукає крапки, коми, знаки питання чи оклику. Якщо речення коротке, воно виводиться повністю. Якщо довге — розбивається на частини (chunks) у місцях логічних пауз (біля ком або сполучників).

3. Розрахунок тривалості показу (Timing): Для кожного створеного фрагмента тексту (чанка) розраховується індивідуальний час його перебування на екрані. Базова тривалість відображення і-го фрагмента тексту (чанка) в середовищі доповненої реальності розраховується за формулою (2.1):

$$T_i = \frac{N_i \times 60}{WPM} + T_{delay} \qquad (2.1)$$

де Ti – час відображення фрагмента тексту на екрані (у секундах);
Ni – кількість слів у поточному фрагменті;
WPM (Words Per Minute) – середня швидкість читання цільової аудиторії (рекомендоване значення для осіб з порушеннями слуху складає 150-180 слів/хв);
60 – коефіцієнт переводу хвилин у секунди;
Tdelay – базова константа когнітивної затримки (орієнтовно 0,5 с), необхідна користувачу для переведення фокусу зору з експоната на текст.

Для середньої швидкості 150-200 WPM, на читання одного слова виділяється близько 300-400 мілісекунд. Таким чином, фрагмент із 5 слів відображатиметься в AR-просторі приблизно 2 секунди.

4. Рендеринг (Візуалізація): Після завершення таймера поточний фрагмент тексту замінюється наступним за допомогою плавної анімації (Fade-

in/Fade-out). Це імітує ефект справжніх динамічних субтитрів, які зазвичай супроводжують відеоконтент.

Крім того, алгоритм передбачає інтеграцію з аналізом тональності тексту (Sentiment Analysis), що застосовується у режимі "Kids". Якщо в тексті виявляється емодзі, алгоритм може генерувати поруч із субтитрами просторові AR-частинки (Particle effects), що привертають увагу дитини та підсилюють емоційне розуміння контексту без аудіальних засобів.

Для формалізації логіки функціонування підсистеми адаптації візуального контенту розроблено блок-схему алгоритму динамічного розбиття та синхронізації текстових даних (рисунок 2.5). Представлена схема деталізує послідовність трансформації неперервного асинхронного потоку інформації, що надходить від віддаленого сервера.

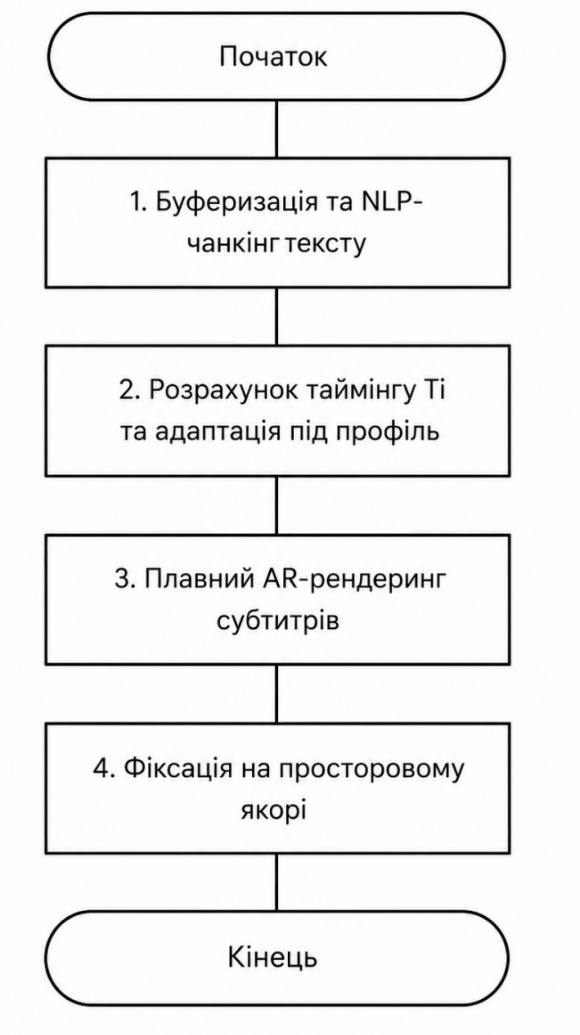

Рисунок 2.5 – Схема локального алгоритму динамічного розбиття та синхронізації тексту

Математично та логічно алгоритм забезпечує мінімізацію затримок від моменту розпізнавання об'єкта до початку виведення текстового супроводу. На початкових етапах виконується асинхронне накопичення сирих байтів у локальному буфері мережевого шару застосунку з їх подальшим декодуванням у формат UTF-8. На етапі синтагматичного аналізу система використовує базові правила обробки природної мови (NLP), розраховуючи межі фрагментів за позиціями знаків завершення речення або логічних пауз.

Ключовою особливістю алгоритму є розрахунок індивідуального часового інтервалу відображення для кожної сформованої текстової порції (чанка). Завдяки впровадженню математичної моделі, що враховує динамічний показник швидкості читання цільової аудиторії та константу когнітивної затримки , система унеможливлює виникнення ситуацій, коли користувач не встигає засвоїти інформацію на екрані.

Залежно від прапора конфігурації профілю (audience_mode), алгоритм розгалужується: для дитячої аудиторії паралельно запускається мікромодуль аналізу тональності, який збагачує візуальний ряд просторовими ефектами частинок. Фінальний рендеринг здійснюється безпосередньо у графічному конвеєрі доповненої реальності із застосуванням плавних інтерполяцій прозорості, що нівелює візуальний шум і забезпечує комфортне безбар'єрне сприйняття екскурсійного матеріалу.

Для навчання та тестування підсистеми розпізнавання музейних експонатів використано відкритий набір даних «Sculptures of Greek Gods (Olympians) Dataset» [34]. Набір містить зображення скульптур та художніх зображень представників давньогрецького олімпійського пантеону, зібрані з відкритих інтернет-джерел. Кожен клас відповідає окремому персонажу давньогрецької міфології та включає фотографії музейних експонатів, архітектурних об'єктів, скульптурних композицій і художніх реконструкцій.

Особливістю вихідного набору даних є його неоднорідність. Поряд із фотографіями реальних музейних експонатів у ньому присутні цифрові ілюстрації, художні зображення, фрагменти картин, фотографії сувенірної продукції,

татуювання та інші нерелевантні об'єкти. Така структура характерна для наборів даних, сформованих шляхом автоматизованого збору інформації з веб-ресурсів, і потребує додаткової підготовки перед використанням у задачах комп'ютерного зору.

На рисунку 2.6 наведено фрагмент невідфільтрованого набору даних для класу «Афіна» (Athena). Як видно з рисунка, вибірка містить як фотографії скульптур та архітектурних об'єктів, що безпосередньо відображають цільовий клас, так і сторонні зображення, зокрема цифрові ілюстрації та фотографії татуювань. Наявність таких даних може негативно впливати на якість навчання моделі, тому на етапі підготовки набору було виконано очищення вибірки та видалення нерелевантних зображень.

Рисунок 2.6 – Фрагмент невідфільтрованого набору даних для класу «Афіна» (Athena)

Після попереднього аналізу набору даних було виконано процедуру Data Cleaning, яка передбачала ручне видалення дублікатів, нерелевантних зображень та пошкоджених файлів. У результаті було сформовано очищену вибірку, придатну для подальшого донавчання моделі MobileNetV3 у задачі класифікації музейних експонатів.

2.5.1 Організація зберігання та обробки інформації в RAG-підсистемі

Інформаційне забезпечення системи базується на використанні технології Retrieval-Augmented Generation (RAG), яка поєднує можливості великих мовних моделей із локальною базою знань музею. Такий підхід дозволяє забезпечити фактологічну точність відповідей та адаптувати інформаційний контент до особливостей конкретного експоната.

Для забезпечення точності інформації сервер використовує локальну векторну базу даних (наприклад, ChromaDB або FAISS) [32]. Процес обробки запиту включає пошук релевантних текстових фрагментів про розпізнаний експонат, які потім додаються до контексту мовної моделі. Це дозволяє системі генерувати фактично правильні відповіді, спираючись на офіційні архівні дані музею, а не лише на загальні ваги (знання) самої моделі.

У системі реалізована логіка динамічного формування інструкцій для LLM залежно від параметра audience_mode (адаптивний генератор системних промптів). У «дорослому» режимі модель фокусується на мистецтвознавчому аналізі та історичних паралелях. У «дитячому» режимі сервер додає до промпту вимоги щодо спрощення синтаксису, використання емодзі та додавання інтерактивних запитань, що стимулюють візуальну увагу дитини з порушеннями слуху.

Контролер асинхронної трансляції через WebSocket, на відміну від традиційних REST-запитів, утримує відкрите з'єднання з мобільним клієнтом протягом усього циклу розповіді. Використання асинхронного генератора (asyncio) в Python дозволяє передавати текст порціями (token streaming). Це повністю нівелює психологічний дискомфорт користувача від тривалої генерації відповіді (10–14 секунд), оскільки перший візуальний відгук у вигляді AR-субтитрів з’являється вже через 1.5–2 секунди після здійснення запиту [35].

Використання сервера Uvicorn як ASGI-інтерфейсу дозволило реалізувати ефективну обробку конкурентних запитів від групи відвідувачів музею. Тестування показало, що FastAPI забезпечує мінімальні затримки на рівні обробки вхідного кадру (менше 150 мс), що разом із оптимізованою моделлю PyTorch створює відчуття миттєвої реакції системи на дії користувача. Такий розподіл ресурсів, де

важкі математичні обчислення винесені на сервер, а просторова візуалізація та розбиття тексту (Dynamic Chunking) – на клієнт, є оптимальним патерном для інклюзивних AR-систем реального часу [36].

2.6 Обґрунтування вибору технологій

Розробка інклюзивного мобільного застосунку з інтеграцією технологій комп'ютерного зору, доповненої реальності та систем штучного інтелекту вимагає формування комплексного технологічного стеку. Вибір програмних засобів та фреймворків здійснювався на основі критеріїв продуктивності, можливості кросплатформного розгортання, підтримки асинхронної обробки потокових даних та наявності розвиненого інструментарію для роботи з тривимірною графікою. Відповідно до визначених архітектурних вимог, програмну систему розділено на клієнтський та серверний рівні.

Для реалізації клієнтської частини застосунку (мобільного інтерфейсу та AR-модуля) обрано кросплатформний фреймворк Flutter від компанії Google [11, 23]. Використання мови програмування Dart, на якій базується фреймворк, дозволяє компілювати код безпосередньо у нативні інструкції ARM для платформ iOS та Android (AOT-компіляція), що гарантує високу швидкодію, критично важливу для застосунків доповненої реальності. Окрім того, рушій рендерингу Impeller забезпечує стабільну частоту оновлення екрану на рівні 60 FPS, що мінімізує візуальні затримки (jitter) під час просторового відстеження об'єктів та запобігає виникненню дискомфорту у користувачів.

Для інтеграції функцій доповненої реальності застосовано бібліотеку ar_flutter_plugin [23], яка виступає універсальною абстракцією над нативними SDK (ARCore для середовища Android та ARKit для платформи iOS) [8]. Цей інструментарій забезпечує доступ до методів візуально-інерціальної одометрії (VIO), що дозволяє з високою точністю встановлювати просторові якорі (AR Anchors) та підтримувати їх стабільність навіть в умовах динамічного освітлення музейних залів.

Окремої уваги потребує архітектура управління станами мобільного застосунку. З огляду на необхідність обробки потокових текстових даних від великих мовних моделей (LLM), було обрано рішення Riverpod [25]. Даний інструмент забезпечує безпечне управління станами, підтримує реактивне оновлення інтерфейсу та ефективну роботу з асинхронними даними через механізм AsyncValue. Це дозволяє динамічно оновлювати лише ті компоненти інтерфейсу, які відповідають за відображення потокових субтитрів та AR-контенту, не створюючи надмірного навантаження на графічну підсистему мобільного застосунку. Це дозволяє реактивно перебудовувати лише ті компоненти інтерфейсу (зокрема віджети AnimatedTextKit), які відповідають за виведення потокових субтитрів, не перевантажуючи графічний конвеєр всього AR-середовища. За мережеву взаємодію на клієнті відповідає клієнт HTTP-запитів Dio, який підтримує гнучке налаштування перехоплювачів (interceptors) для управління токенами авторизації та обробки тайм-аутів при тривалих запитах до RAG-сервера [18].

Серверна частина (Backend), яка відповідає за важкі обчислювальні процеси розпізнавання зображень та генерації тексту, реалізована на базі мови програмування Python. Вибір Python зумовлений його статусом індустріального стандарту у сфері машинного навчання (Machine Learning) та розробки систем штучного інтелекту. Як веб-фреймворк обрано FastAPI [26], що базується на специфікації ASGI (Asynchronous Server Gateway Interface). Завдяки асинхронній природі та використанню бібліотеки Pydantic для валідації даних, FastAPI забезпечує надзвичайно високу пропускну здатність та мінімальні затримки при обробці запитів, перевершуючи традиційні синхронні фреймворки.

Для забезпечення фактологічної достовірності відповідей використано архітектуру Retrieval-Augmented Generation (RAG) [22]. На відміну від традиційного використання великих мовних моделей, RAG дозволяє доповнювати запити актуальною інформацією з локальної бази знань музею. Це суттєво зменшує ризик генерації недостовірної інформації та забезпечує відповідність сформованого контенту конкретним музейним експонатам.

Для реалізації модуля комп'ютерного зору (розпізнавання музейних експонатів за надісланими кадрами з камери) інтегровано бібліотеку PyTorch [24].

Для реалізації підсистеми комп’ютерного зору використано бібліотеку PyTorch та архітектуру MobileNetV3-Small [29]. Вибір даної моделі зумовлений її високою обчислювальною ефективністю та невеликою кількістю параметрів порівняно з класичними глибокими згортковими мережами. Завдяки використанню глибинних роздільних згорток (Depthwise Separable Convolutions) модель забезпечує високу швидкість класифікації зображень при збереженні достатньої точності розпізнавання музейних експонатів. Це робить її придатною для використання в системах реального часу з обмеженими обчислювальними ресурсами. Це дозволяє зменшити час класифікації одного кадру до 100–150 мс навіть на базових серверних процесорах (CPU). Оскільки в музейному просторі об'єкти є статичними та мають яскраво виражені унікальні ознаки, легкої моделі MobileNetV3-Small абсолютно достатньо для забезпечення точності понад 85%, що гарантує ідеальний баланс між продуктивністю (Throughput) та безперервністю інклюзивного користувацького досвіду.

Таким чином, обраний технологічний стек поєднує сучасні засоби мобільної розробки, технології доповненої реальності, методи комп’ютерного зору та інструменти штучного інтелекту. Використання Flutter, Riverpod, FastAPI, PyTorch та RAG-архітектури забезпечує побудову масштабованої клієнт-серверної системи, здатної функціонувати в режимі реального часу та підтримувати принципи цифрової доступності для осіб з порушеннями слуху.

2.7 Моделювання процесів функціонування системи

Проєктування складної програмної системи неможливе без формалізації процесів взаємодії між її компонентами та кінцевими користувачами. Для моделювання поведінки застосунку та опису вимог до функціональності використано методологію об’єктно-орієнтованого аналізу на базі уніфікованої мови моделювання (UML). У межах моделювання було побудовано комплекс UML- та DFD-діаграм, які відображають структуру системи, послідовність обробки даних та взаємодію між клієнтськими і серверними компонентами.

Функціональне моделювання системи розпочинається з визначення основних акторів (учасників) та прецедентів їх взаємодії (рисунок 2.7). Головним актором є «Відвідувач музею» (користувач із порушеннями слуху), який взаємодіє з мобільним клієнтом. Допоміжним актором виступає «Сервер штучного інтелекту», що виконує роль зовнішньої обчислювальної системи для класифікації даних та формування контенту.

Базовий сценарій взаємодії охоплює такі логічні етапи:

1. Ініціалізація сесії: відвідувач активує модуль доповненої реальності, після чого система виконує калібрування сенсорів (гіроскопа, акселерометра) та починає сканування простору.

2. Захоплення кадру: при наведенні камери на об'єкт застосунок автоматично (або за ініціативою користувача) фіксує зображення експоната.

3. Ідентифікація об'єкта: мобільний клієнт здійснює попередню обробку зображення (зменшення роздільної здатності для економії трафіку) та відправляє його через REST API на сервер. Серверний модуль за допомогою нейронної мережі ідентифікує об'єкт і повертає його унікальний ідентифікатор [21].

4. Потокова генерація контенту: отримавши ідентифікатор, клієнт встановлює двостороннє з'єднання із сервером. Інтелектуальна система на базі LLM починає генерувати адаптований текст екскурсії.

5. Візуалізація: алгоритм локального клієнта (динамічний чанкінг) порційно виводить отриманий текст в AR-середовищі у вигляді синхронізованих субтитрів.

Схема інформаційної взаємодії потоків даних (Data Flow) має яскраво виражений асинхронний характер. Враховуючи специфіку обробки зображень та звернень до мовних моделей, взаємодія базується на принципах мікросервісної архітектури. Мобільний пристрій відіграє роль «тонкого клієнта» (Thin Client), делегуючи обчислювально складні завдання на сервер.

Послідовність обробки даних виглядає наступним чином. Після формування HTTP-запиту з графічним файлом (payload), сервер приймає потік байтів, конвертує його у тензор (багатовимірний масив) за допомогою інструментарію torchvision.transforms та передає на вхід навченій моделі PyTorch. Модель повертає

вектор ймовірностей, з якого витягується індекс класу з найвищим показником (argmax). Цей індекс зіставляється з базою даних музею для отримання метаданих експоната. На наступному етапі метадані комбінуються з промптом (який враховує налаштування вікової категорії користувача: Adult або Kids) і передаються до генеративного модуля. Для уникнення проблеми довгого очікування («зависання» інтерфейсу), сервер не чекає повної генерації всієї розповіді. Замість цього використовується потокова передача токенів (Streaming Response), де кожне згенероване слово або фраза миттєво відправляється назад до мобільного додатка.

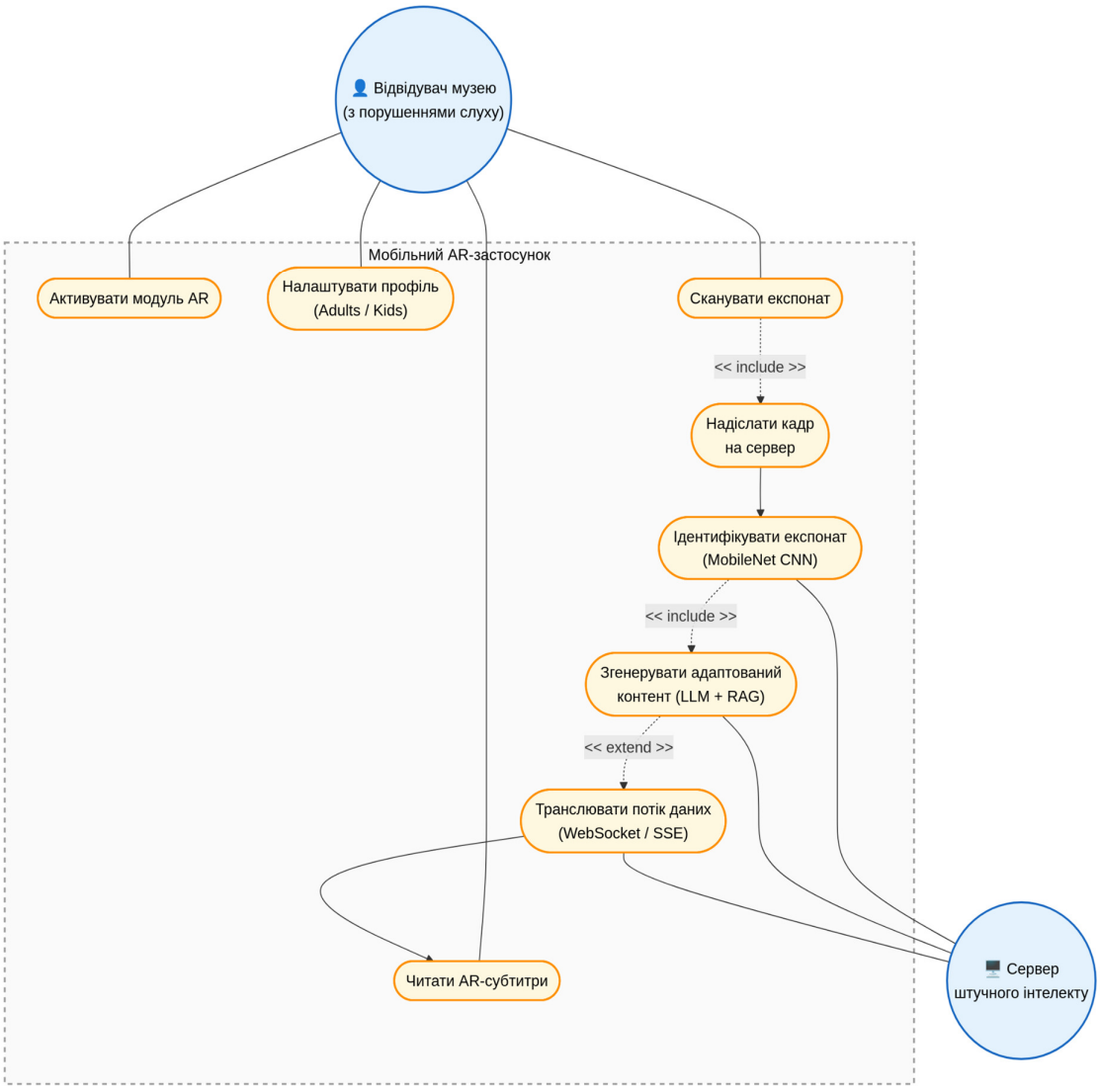

Рисунок 2.7 – Діаграма прецедентів (Use Case) системи

На боці мобільного клієнта контролер стану (Riverpod Notifier) приймає цей потік байтів, декодує їх у рядки формату UTF-8 та накопичує у локальному буфері.

Апаратні ресурси смартфона використовуються виключно для розрахунку матриць трансформацій при накладанні субтитрів поверх зображення з камери та забезпечення плавної анімації появи тексту. Такий розподіл інформаційних потоків забезпечує високу масштабованість системи, економію заряду акумулятора мобільного пристрою та дозволяє централізовано оновлювати алгоритми розпізнавання на сервері без необхідності постійного випуску оновлень самого мобільного застосунку.

Детальна структура клієнтського рівня інформаційної системи (рисунок 2.8) відображає архітектуру «тонкого клієнта», оптимізовану для мінімізації обчислювального навантаження на мобільний пристрій. Взаємодія починається з модуля камери та сенсорів візуально-інерціальної одометрії (VIO), який передає сирий відеопотік нативному AR-двигуну. Модуль ARCore/ARKit обробляє просторові дані та фіксує віртуальні якорі (AR Anchors) у тривимірних координатах.

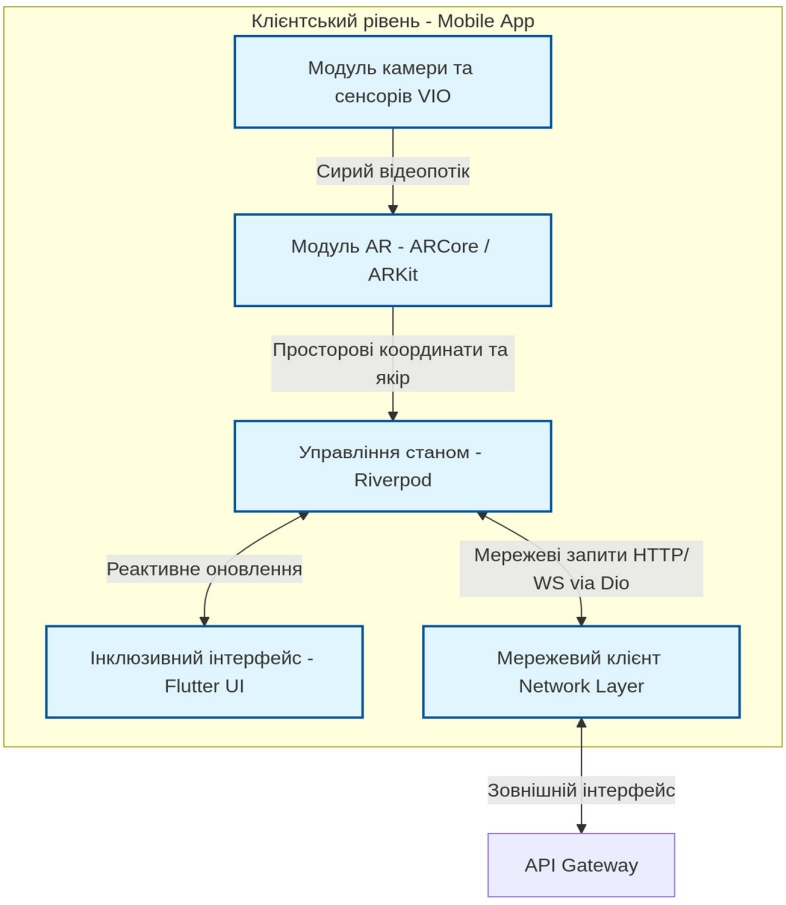

Рисунок 2.8 – Архітектура клієнтської частини мобільного застосунку

Отримані координати передаються до шару управління станом застосунку, реалізованого за допомогою фреймворку Riverpod. Шар стану виконує роль координатора: з одного боку, він ініціює асинхронні запити через мережевий клієнт (Network Layer) до сервера, з іншого — забезпечує реактивне оновлення віджетів інклюзивного інтерфейсу користувача (Flutter UI). Завдяки такій ізоляції, рендеринг просторових субтитрів та відображення адаптованого UI відбуваються без затримок у графічному конвеєрі додатка.

Серверний рівень інформаційної системи (Рисунок 2.9) побудований за принципом асинхронної мікросервісної архітектури під керуванням веб-фреймворку FastAPI. API Gateway виступає єдиною точкою входу для клієнтських запитів, підтримуючи постійне двостороннє з'єднання за протоколом WebSocket для низьколатентного обміну даними.

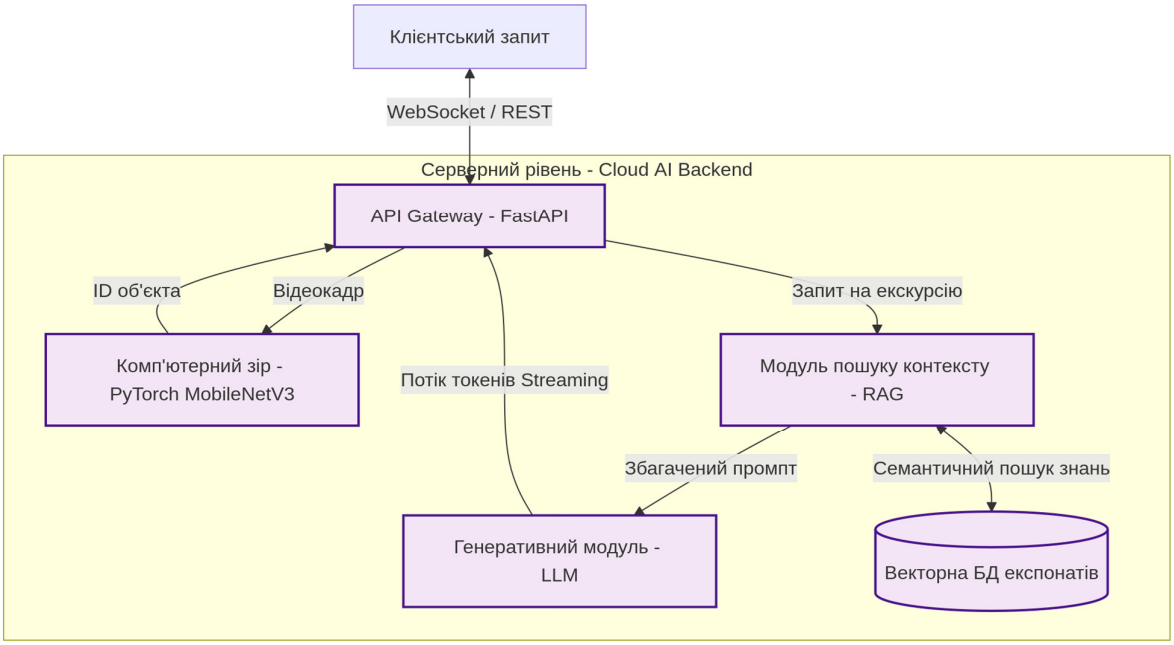

Рисунок 2.9 – Архітектура серверної підсистеми та модулів штучного інтелекту

Процес обробки інформації на сервері розділений на дві ізольовані гілки штучного інтелекту: комп'ютерний зір та генерація природної мови. Модуль комп'ютерного зору на базі PyTorch приймає кадр, трансформує його у тензор і за

допомогою оптимізованої мережі MobileNetV3 ідентифікує унікальний ID музейного експоната. Отриманий ідентифікатор передається до модуля RAG (Retrieval-Augmented Generation), який виконує семантичний пошук історичних довідок у векторній базі даних експонатів. Знайдений контекст об'єднується із системним промптом (адаптованим під обраний профіль користувача) та надходить до генеративної мовної моделі (LLM). Модель повертає відповідь у вигляді потоку токенів (Streaming Response) через API Gateway безпосередньо у мобільний додаток, що дозволяє уникнути тривалого очікування повної генерації тексту.

На відміну від структурного моделювання, яке визначає компонентний склад системи, функціональна схема (рисунок 2.10) описує динаміку та послідовність виконання технологічних операцій із перетворення вхідних даних у кінцевий інклюзивний інтерфейс. Логіка функціонування системи побудована у вигляді послідовного алгоритмічного ланцюга, що складається з п'яти базових функцій (F):

1. F1 (Захоплення простору та пошук маркерів): Початковий етап взаємодії, на якому мобільний застосунок ініціалізує камеру та сенсори пристрою. У режимі реального часу відбувається аналіз відеопотоку алгоритмами комп'ютерного зору (на базі ARCore/ARKit) для виявлення заздалегідь визначених візуальних маркерів (музейних експонатів) у тривимірному просторі та оптимізації фокусу камери.

2. F2 (Ідентифікація об'єкта через нейромережу): При виявленні маркера поточний кадр передається на серверний рівень. Функція відповідає за попередню обробку зображення (масштабування, нормалізацію текстур) та його класифікацію за допомогою згорткової нейронної мережі MobileNetV3 на базі фреймворку PyTorch. Результатом виконання функції є точне визначення унікального ідентифікатора (ID) експоната.

3. F3 (Формування адаптованого контенту LLM): Отриманий ідентифікатор запускає процес генерації екскурсійного супроводу. Модуль RAG здійснює семантичний пошук історичних довідок у векторній базі знань музею, формує контекст і передає його до великої мовної моделі (LLM). Особливістю цієї функції є динамічна адаптація тексту під когнітивні потреби користувача: залежно від обраного режиму (Adult або Kids), ШІ оптимізує синтаксичну складність речень.

4. F4 (Синтагматичне розбиття тексту на чанки): Оскільки сервер повертає згенеровану інформацію асинхронним потоком (Streaming Response), на боці клієнта активується функція NLP-аналізу. Вона здійснює буферизацію токенів та синтагматичний поділ суцільного тексту на логічно завершені короткі порції (чанки по 4–7 слів) на основі розділових знаків, запобігаючи візуальному перевантаженню екрана.

5. F5 (Просторове відображення AR-субтитрів): Фінальний етап, на якому для кожного текстового чанка розраховується точний час відображення за математичною формулою (2.1), що враховує середню швидкість читання глухих відвідувачів. Модуль доповненої реальності здійснює рендеринг висококонтрастних текстових вузлів, які за допомогою плавних анімацій (Fade-in/Fade-out) візуалізуються у просторі музею із прив'язкою до віртуального якоря фізичного об'єкта (рисунок 2.10).

Рисунок 2.10 – Функціональна схема процесів перетворення інформації в системі

Таким чином, представлена функціональна схема демонструє безперервний

цикл обробки даних, де кожна наступна функція критично залежить від результату виконання попередньої, що забезпечує низьку латентність системи та високу якість представлення інклюзивного музейного контенту.

Для деталізації маршрутизації інформації, визначення джерел даних та логіки їх трансформації було розроблено діаграму потоків даних (Data Flow Diagram – DFD), представлену на рисунку 2.11. Діаграма наочно демонструє життєвий цикл інформації від моменту її генерації користувачем до отримання кінцевого результату у вигляді AR-контенту.

На концептуальному рівні архітектура потоків даних складається з наступних елементів:

1. Зовнішні сутності (External Entities):

– Відвідувач музею: є головним ініціатором процесів. Ця сутність генерує два типи вхідних потоків: «Сирі кадри» (відеопотік з камери смартфона) та «Налаштування аудиторії» (параметри профілю, такі як віковий режим Adult/Kids або швидкість читання). Зворотним зв'язком для відвідувача є готовий візуальний контент.

– Адміністратор: допоміжна сутність, яка не бере участі в реальному часі взаємодії (Real-time), але забезпечує періодичне «Оновлення» баз даних системи через спеціалізовану панель керування (CMS).

2. Сховища даних (Data Stores):

– D1 (База маркерів): реляційне або файлове сховище еталонних зображень (AR-тригерів) та їх просторових характеристик.

– D2 (Векторна БД): спеціалізоване сховище (наприклад, Milvus або Pinecone), що містить семантично розмічений історичний контекст (векторні ембединги) для роботи пошукової системи RAG.

3. Процеси перетворення (Processes) та рух інформації:

– потік даних ініціюється у вузлі P1 (Обробка відео), який приймає сирий відеопотік від камери відвідувача, виконує його компресію, виділення ключових кадрів та передає оптимізоване «Зображення» далі;

– у процесі P2 (Розпізнавання експоната) алгоритми комп'ютерного зору виконують двонаправлений запит (порівняння) до сховища D1. У разі

підтвердження збігу маркера з еталоном, процес формує унікальний «ID об'єкта» і передає його до наступного вузла;

– процес P3 (Генерація опису) виступає інтелектуальним ядром бекенда. Він агрегує два потоки даних: «ID об'єкта» від P2 та «Налаштування аудиторії» безпосередньо від користувача. На основі ідентифікатора процес звертається до векторної бази D2, отримує релевантний «Історичний контекст» і передає його у мовну модель (LLM). Результатом роботи цього процесу є динамічний «Потік токенів» (Streaming), що безперервно відправляється на клієнт;

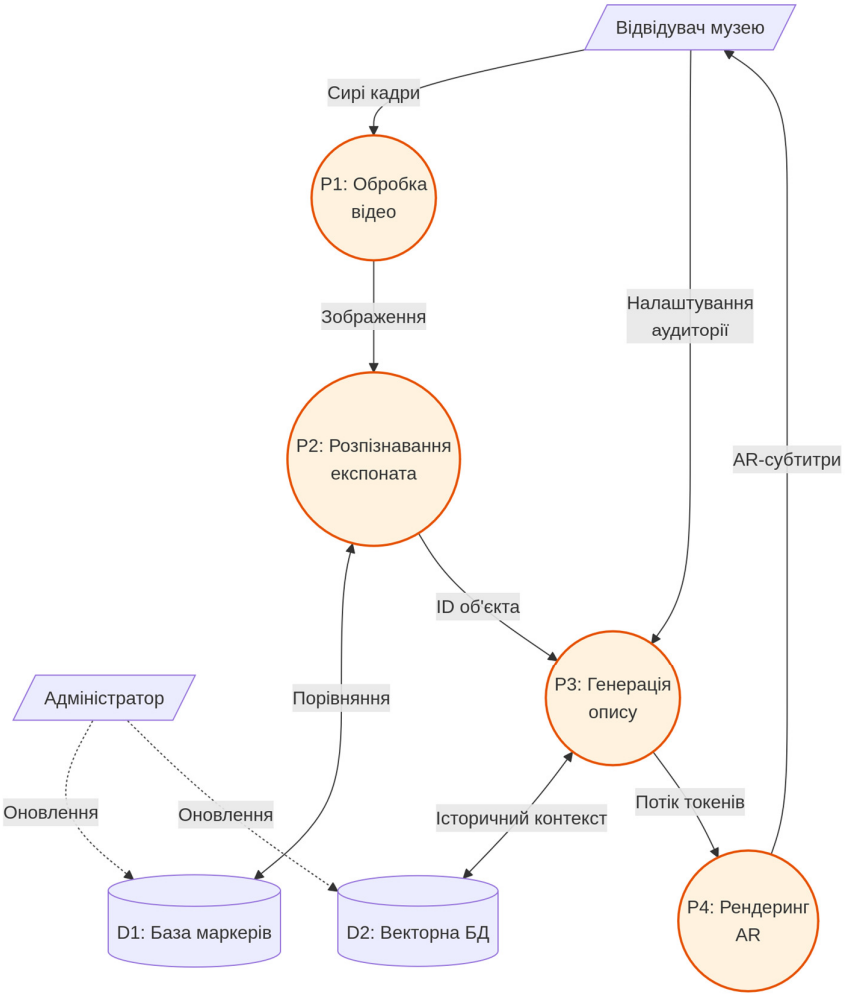

Рисунок 2.11 – Діаграма потоків даних інформаційної системи

– фінальний процес P4 (Рендеринг AR) функціонує на боці мобільного додатка. Він приймає потокові текстові дані, синхронізує їх з просторовими

координатами AR-якоря та транслює кінцевій сутності («Відвідувачу музею») у вигляді накладених на реальний світ інклюзивних «AR-субтитрів».

Розроблена схема інформаційних потоків засвідчує високий рівень модульності системи. Чітке розмежування процесів розпізнавання (P2) та генерації контенту (P3) гарантує, що архітектура уникне ефекту "вузького місця" (bottleneck), забезпечуючи швидку та асинхронну обробку даних для комфортної взаємодії користувачів з порушеннями слуху.

Для формалізації хронологічного порядку подій та відображення динаміки обміну повідомленнями між компонентами системи в режимі реального часу було розроблено UML-діаграму послідовності (рисунок 2.12). Ця діаграма ілюструє базовий сценарій використання інклюзивного AR-застосунку під час екскурсії.

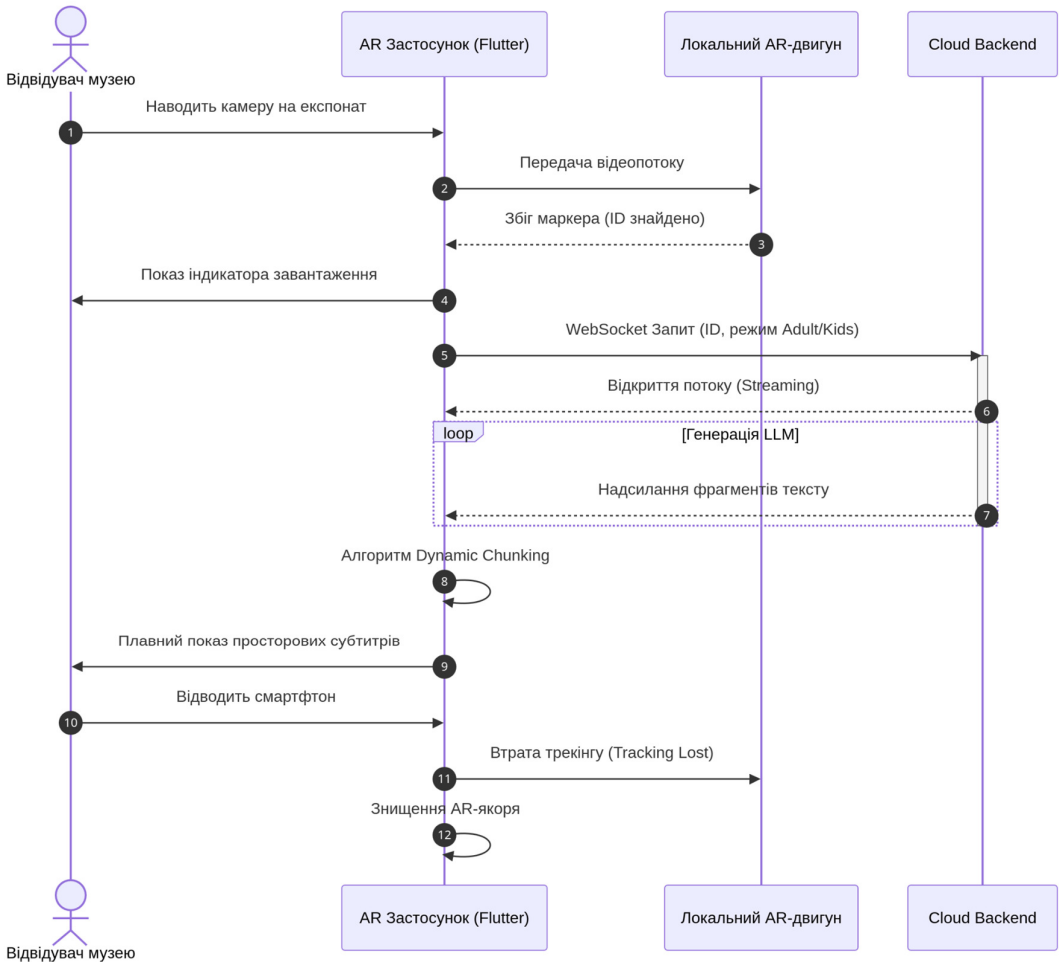

Рисунок 2.12 – Діаграма послідовності взаємодії користувача з системою

У процесі взаємодії беруть участь чотири основні лінії життя (Lifelines):

«Відвідувач музею» (актор), «AR Застосунок» (клієнтський інтерфейс на Flutter), «Локальний AR-двигун» (нативний модуль ARCore/ARKit) та «Backend» (Серверна підсистема).

Хронологічний процес розбивається на чотири логічні фази:

1. Фаза ініціалізації та розпізнавання (Кроки 1–4): Взаємодія розпочинається, коли відвідувач наводить камеру смартфона на музейний експонат. Застосунок безперервно передає відеопотік до локального AR-двигуна. Щойно алгоритми комп'ютерного зору фіксують збіг маркерних точок з еталоном (Image Tracking), AR-двигун повертає системі унікальний ідентифікатор об'єкта (ID) та фіксує його координати. Одразу після цього застосунок виводить на екран користувача візуальний індикатор завантаження. Це є критично важливим UX-рішенням для глухих користувачів, оскільки компенсує відсутність звукових системних сповіщень про успішне сканування.

2. Фаза асинхронної комунікації (Кроки 5–8): Отримавши ідентифікатор, застосунок ініціює з'єднання із сервером за протоколом WebSocket. У тілі запиту передається ID експоната та вибраний когнітивний режим профілю (Adult або Kids). Сервер приймає запит (активуючи свої обчислювальні ресурси) і відкриває канал потокової передачі даних (Streaming). У межах циклу генерації (Loop) велика мовна модель (LLM) асинхронно надсилає згенеровані фрагменти тексту назад до мобільного клієнта в міру їх готовності, не чекаючи завершення всієї розповіді. Це дозволяє звести латентність до мінімуму.

3. Фаза локальної обробки та візуалізації (Кроки 9–10): Клієнтська частина інформаційної системи забезпечує прийом вхідного потоку токенів та ініціює їх обробку за допомогою розробленого алгоритму динамічної буферизації та синтагматичного розбиття тексту (Dynamic Chunking). Сформовані текстові блоки підлягають плавному рендерингу у вигляді просторових субтитрів (Spatial Subtitles), позиціонування яких жорстко прив'язане до тривимірних координат цільового фізичного експоната.

4. Фаза завершення сесії та звільнення ресурсів (Кроки 11–13): Після закінчення взаємодії користувача з експонатом та зміни ракурсу камери смартфона, локальний модуль доповненої реальності фіксує переривання візуального контакту

з еталонним маркером (стан *Tracking Lost*). У відповідь на цей системний тригер програмний застосунок автоматично деактивує відображення субтитрів та ініціює процедуру видалення просторового якоря (Anchor Destruction). Виконання зазначеної процедури є критично необхідним для активації механізму збирання сміття (Garbage Collection), вивільнення обсягів оперативної пам'яті мобільного пристрою та підготовки архітектури до ідентифікації наступних об'єктів.

Побудовані UML- та DFD-діаграми підтверджують коректність обраної архітектури програмної системи та демонструють послідовність обробки інформації від моменту розпізнавання музейного експоната до відображення адаптованого AR-контенту. Використання потокової передачі даних, локального AR-трекінгу та серверної генерації контенту забезпечує безперервну роботу системи в режимі реального часу та створює інклюзивне цифрове середовище для осіб з порушеннями слуху.

3 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

3.1 Реалізація клієнтської частини

Процес практичної реалізації мобільного застосунку розпочався з формування архітектурного каркаса на базі фреймворку Flutter. Враховуючи вимоги до інклюзивності та необхідність обробки потокових даних штучного інтелекту у реальному часі, особливу увагу було приділено організації шару управління станами та мережевої взаємодії.

Для забезпечення модульності, ізоляції бізнес-логіки та тестованості вихідного коду використано пакет flutter_riverpod. Вибір цього інструментарію дозволив реалізувати декларативний підхід до управління станами, де кожен функціональний блок застосунку (модуль камери, розпізнавач маркерів, інтерфейс просторових субтитрів) представлений окремим провайдером. Це дозволяє повністю уникнути прямої жорсткої залежності між компонентами інформаційної системи та забезпечує ефективне оновлення лише необхідних піддерев віджетів без перевантаження графічного конвеєра. Детальна програмна імплементація архітектури провайдерів, включаючи логіку ініціалізації інфраструктурного об'єкта ProviderContainer, представлена у Додатку Б.

Мережева архітектура застосунку базується на використанні оптимізованої HTTP-бібліотеки dio. Реалізація клієнт-серверної взаємодії розділена на два послідовні етапи:

1. Синхронна ідентифікація експоната: за допомогою асинхронного методу POST на сервер відправляється графічний файл (поточний кадр із камери пристрою). Для оптимізації обсягів бінарного трафіку в реальних музейних умовах реалізовано механізм попереднього стиснення зображення перед ініціалізацією запиту. На рівні конфігурації клієнта Dio налаштовано спеціалізовані перехоплювачі (interceptors), які забезпечують автоматичне логування сесій, діагностику мережевих затримок, встановлення часових лімітів очікування (connectTimeout, receiveTimeout) та централізовану обробку виняткових ситуацій з'єднання.

2. Асинхронне отримання текстового потоку: після успішної верифікації об’єкта сервером, мобільний застосунок ініціює стійке двостороннє з’єднання за протоколом WebSocket. Це дозволяє реалізувати ефект низьколатентного виведення інформації, де кожен новий текстовий токен, згенерований генеративною моделлю сервера, миттєво доставляється клієнту через канал зв'язку.

Для управління асинхронними даними та життєвим циклом з'єднань в інфраструктурі Riverpod реалізовано архітектурні класи Notifier та StreamProvider із модифікатором autoDispose. Використання об'єкта AsyncValue дозволяє нативно відстежувати та декларативно описувати три базові стани обробки інформаційних потоків: завантаження (loading), виникнення помилки (error) та безпосереднє отримання даних (data).

Такий підхід забезпечує коректне та швидке відображення візуальних індикаторів (напівпрозорих скелетонів або прогрес-барів) для користувачів із порушеннями слуху, що є критично важливим аспектом для зменшення когнітивного навантаження під час очікування відповіді від віддаленого ШІ-сервера. Програмний код контролерів мережевої взаємодії та підписки на WebSocket-канали наведено у Додатку Б.

Програмна реалізація описаної логіки асинхронного отримання токенів базується на декларативному описі провайдерів. Нижче представлено архітектурний фрагмент коду, що демонструє конфігурацію мережевого шару застосунку для управління WebSocket-сесією:

```
     final exhibitStreamProvider =
StreamProvider.autoDispose.family<String, String>((ref,
exhibitId) {
       final channel = WebSocketChannel.connect(
         Uri.parse('ws://192.168.1.10/ws/stream/$exhibitId'),
       );

       ref.onDispose(() {
         channel.sink.close();
       });
       return channel.stream.map((data) => data.toString());
     });
```

Цей підхід дозволяє ізолювати мережеву логіку від інтерфейсу користувача. Віджети підписуються на exhibitStreamProvider, автоматично отримуючи

оновлення стану при надходженні кожного нового токена від великої мовної моделі сервера.

Окремим аспектом реалізації стало створення механізму динамічного відображення тексту. Для цього використано анімоване посимвольне виведення текстових повідомлень, яке імітує поступову появу інформації на екрані. Швидкість відображення символів синхронізована з алгоритмом Dynamic Chunking, що забезпечує комфортний темп читання екскурсійного контенту для користувачів з порушеннями слуху [20].

На рівні коду ініціалізація базового функціоналу клієнтської частини включає:

– конфігурацію глобального об'єкта ProviderContainer для централізованого доступу до залежностей застосунку;

– ініціалізацію екземпляра Dio з параметризацією базового URL та конфігурацією перехоплювачів мережевого шару;

– реалізацію контролера ARViewController, який інкапсулює керування життєвим циклом камери та передає захоплені кадри в чергу обробки підсистеми комп'ютерного зору.

Таким чином, розроблений базовий функціонал створює надійну інфраструктуру для стабільної роботи AR-модуля. Використання Riverpod у поєднанні з асинхронними мережевими протоколами WebSocket дозволяє досягти високої чуйності та реактивності інтерфейсу, що є необхідною умовою для забезпечення безбар’єрного доступу до інформації в інклюзивному музейному середовищі.

На рисунку 3.1 представлено високорівневу схему архітектури управління станами мобільного застосунку. В основу покладено багаторівневу структуру, де кожен рівень виконує чітко визначену роль:

– Data Layer (Рівень даних): містить репозиторії, клієнти WebSocket-з'єднань та джерела даних (API-ресурси), що відповідають за низькорівневу взаємодію з сервером;

– State Management Layer (Рівень управління станами): реалізований за допомогою реактивних провайдерів Riverpod. Ключовим елементом є інкапсуляція

бізнес-логики в межах StreamProvider, що автоматично транслює зміну вхідних токенів у стани інтерфейсу;

– Presentation Layer (Рівень представлення): складається з Flutter-віджетів типу ConsumerWidget, які реактивно підписуються на зміни у провайдерах, забезпечуючи плавне оновлення AR-сцени та скломорфних текстових панелей;

Для наочного відображення взаємодії між рівнями системи розроблено схему архітектури управління станами мобільного застосунку, представлену на рисунку 3.1.

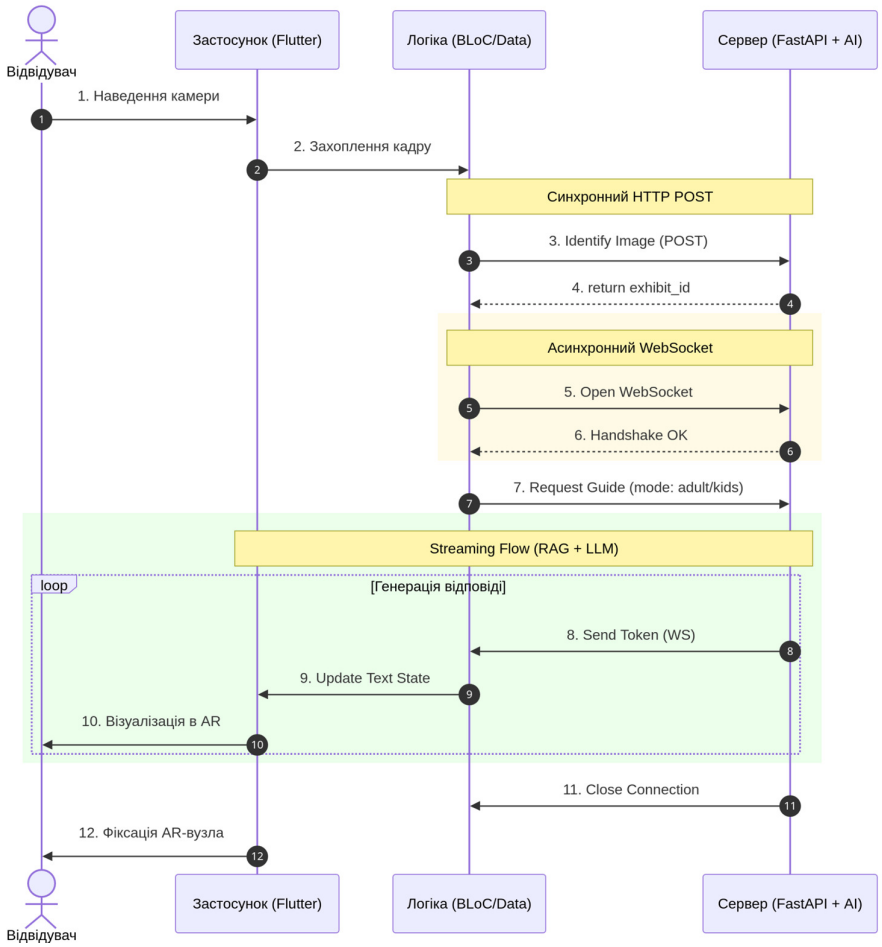

Рисунок 3.1 – Схема архітектури управління станами та потоками даних у мобільному застосунку

Особливістю даної схеми є механізм «зворотного зв'язку» між локальним AR-модулем та шаром управління станами інформаційної системи. При просторовій ідентифікації експоната AR-контролер ініціює зміну стану в Riverpod-

провайдері, що автоматично запускає ланцюжок асинхронних запитів до сервера.

Використання об'єкта AsyncValue дозволяє на рівні коду декларативно описати поведінку інтерфейсу для кожного стану: поки триває процес розпізнавання, користувач бачить індикатор прогресу; при отриманні першого токена тексту стан змінюється на Data, ініціюючи запуск анімації появи субтитрів. Такий підхід мінімізує ймовірність виникнення помилок розсинхронізації та забезпечує високу продуктивність застосунку на пристроях із обмеженими апаратними ресурсами.

3.2 Реалізація AR-модуля

Наступним етапом практичної реалізації стало розгортання середовища доповненої реальності та інтеграція спеціалізованих інструментів візуалізації, що безпосередньо відповідають за інклюзивну складову застосунку. Центральним компонентом цього процесу є конфігурація та запуск AR-сесії за допомогою бібліотеки ar_flutter_plugin.

Процес інтеграції AR-середовища розпочався з реалізації менеджера ідентифікації зображень (Image Tracking Manager). Для того, щоб застосунок міг стабільно розпізнавати музейні експонати як просторові маркери, було створено локальну базу дескрипторів. Кожен експонат у базі представлений як об'єкт ARReferenceImage, що містить фізичні габарити об’єкта та його унікальний ідентифікатор. Це дозволяє AR-двигуну (ARCore/ARKit) проводити безперервний аналіз відеопотоку та, у разі виявлення збігу, генерувати подію розпізнавання (onImageTracked).

Ключовим технічним рішенням при імплементації візуального контенту стала розробка системи динамічних якорів (AR Anchors). При ідентифікації експоната в тривимірній сцені створюється вузол ARNode, позиція якого жорстко фіксується відносно фізичного об’єкта. На цей вузол проєктується візуальний оверлей, що містить адаптовану інформацію.

Для реалізації специфічних потреб осіб з порушеннями слуху було імплементовано наступні локальні інструменти візуалізації:

1. Висококонтрастна текстова панель: оскільки фон у музеї може бути складним та строкатим, для відображення тексту використано спеціалізований віджет із напівпрозорою підкладкою та ефектом розмиття фону (glassmorphism), який забезпечує високий рівень контрастності та читабельності тексту.

2. Механізм асинхронного стримінгу субтитрів: програмна реалізація базується на інтеграції Riverpod AsyncValue та StreamBuilder. Отримані від сервера текстові токени накопичуються в локальному репозиторії, після чого передаються до модуля TypewriterAnimatedText. Це створює плавний ефект появи літер, що значно полегшує візуальне сприйняття інформації порівняно зі статичними блоками тексту.

3. Адаптивне керування фокусом: для запобігання візуального перевантаження реалізовано систему «розумної видимості». Субтитри відображаються лише тоді, коли кут нахилу камери відносно експоната знаходиться в межах заданого діапазону (30–45 градусів), що дозволяє користувачу вільно перемикати увагу між реальним об’єктом та цифровою інформацією.

Особлива увага була приділена реалізації дитячого режиму (Kids Mode). Окрім спрощення синтаксичних конструкцій на рівні сервера, локальний AR-модуль доповнює візуальний ряд спеціальними компонентами – «емоційними частинками» (particle effects). Якщо серверний аналізатор тональності ідентифікує в тексті емодзі, застосунок автоматично генерує навколо тексту анімовані вузли, що візуалізують настрій розповіді.

Приклад роботи розробленого інтерфейсу доповненої реальності в реальному часі представлено на рисунку 3.2. На скріншоті продемонстровано успішне розпізнавання експоната (античної вази) та накладання просторової висококонтрастної панелі субтитрів з ефектом «скломорфізму», яка «прив'язана» до об'єкта у тривимірному просторі за допомогою AR-якоря.

З точки зору коду, імплементація логіки відображення реалізована через підписку на стрім даних у провайдері exhibitContentProvider. Кожне надходження нового чанка тексту ініціює часткове оновлення стану (state.copyWith), що дозволяє уникнути затримок у рендерингу AR-сцени. Такий підхід забезпечує стабільну

роботу інтерфейсу навіть при високій швидкості генерації контенту штучним інтелектом.

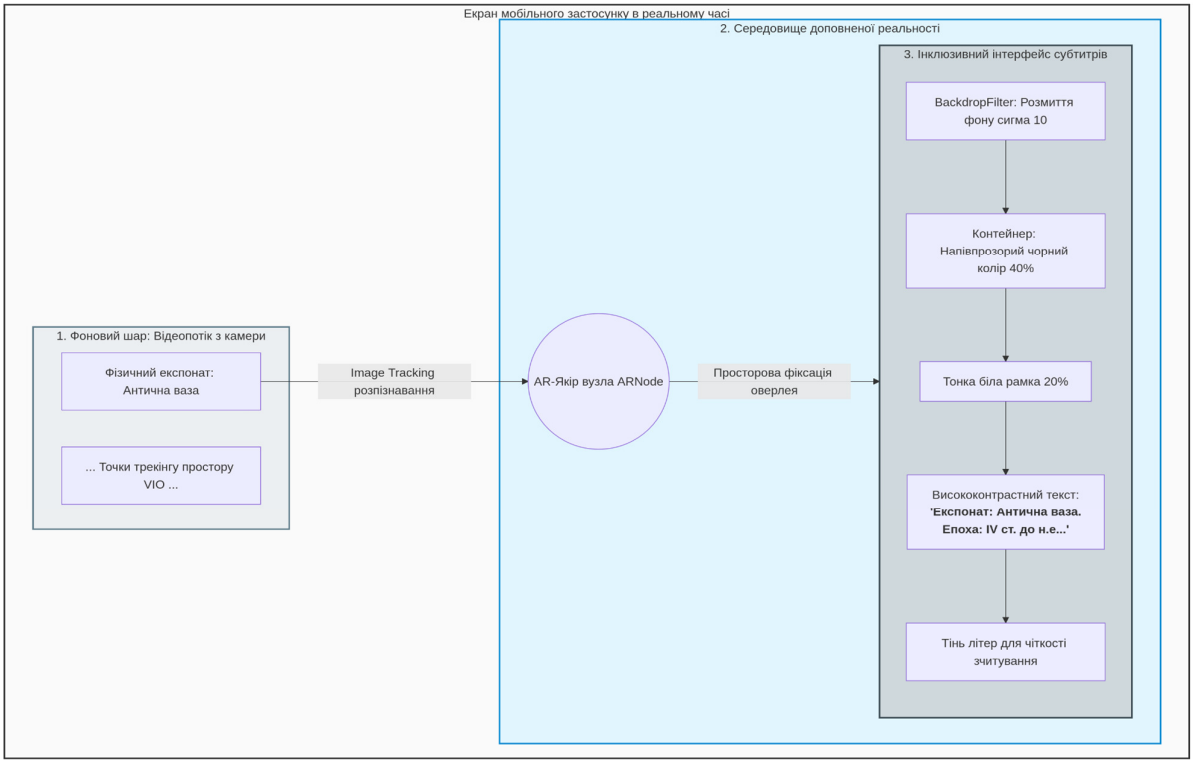

Рисунок 3.2 – Приклад роботи AR-модуля та засобів інклюзивної візуалізації контенту

Програмна реалізація інтерфейсного модуля відображення просторового тексту ґрунтується на застосуванні накладених графічних фільтрів фреймворку Flutter. Нижче наведено фрагмент коду кастомного віджета, який демонструє практичну реалізацію висококонтрастної панелі субтитрів із використанням ефекту скломорфізму:

Фрагмент програмного коду реалізації висококонтрастної панелі просторових субтитрів наведено в лістингу 3.2.

Лістинг 3.2 – Реалізація віджета ARSubtitlePanel

```
class ARSubtitlePanel extends StatelessWidget {
  final String text;
  const ARSubtitlePanel({Key? key, required this.text}) :
super(key: key);
  @override
  Widget build(BuildContext context) {
```

```
    return ClipRRect(
      borderRadius: BorderRadius.circular(20),
      child: BackdropFilter(
        filter: ImageFilter.blur(sigmaX: 10, sigmaY: 10),
        child: Container(
          padding: const EdgeInsets.all(20),
          decoration: BoxDecoration(
            color:   Colors.black.withOpacity(0.4),   //   Темна
підкладка за стандартами WCAG 2.1
            borderRadius: BorderRadius.circular(20),
            border:                              Border.all(color:
Colors.white.withOpacity(0.2)),
          ),
          child: Text(
            text,
            style: const TextStyle(
              fontSize: 24,
              color: Colors.white,
              fontWeight: FontWeight.bold
            ),
          ),
        ),
      ),
    );
  }
}
```

Розроблений віджет інтегрується безпосередньо у вікно рендерингу доповненої реальності. Завдяки динамічному обчисленню шару розмиття (BackdropFilter) та фіксованому коефіцієнту прозорості темної підкладки, додаток нівелює строкатість навколишнього музейного середовища (динамічні тіні відвідувачів, складні візерунки стін), фокусуючи зорову увагу користувача з порушеннями слуху виключно на текстовому контенті.

Графічна реалізація інтерфейсу користувача мобільного застосунку в умовах інтенсивного штучного або природного освітлення виставкового залу представлена на рисунку 3.3. На наведеному скріншоті продемонстровано функціонування системи у світлій колірній палітрі, що забезпечує стабільну адаптацію візуального ряду до навколишнього середовища з високим рівнем яскравості.

Зовнішній вигляд розробленого мобільного застосунку під час виконання основних сценаріїв роботи наведено на рисунку 3.3. На рисунку представлено початковий екран сканування музейного експоната та результат його успішної ідентифікації із подальшим відображенням інформаційного контенту у вигляді висококонтрастної текстової панелі.

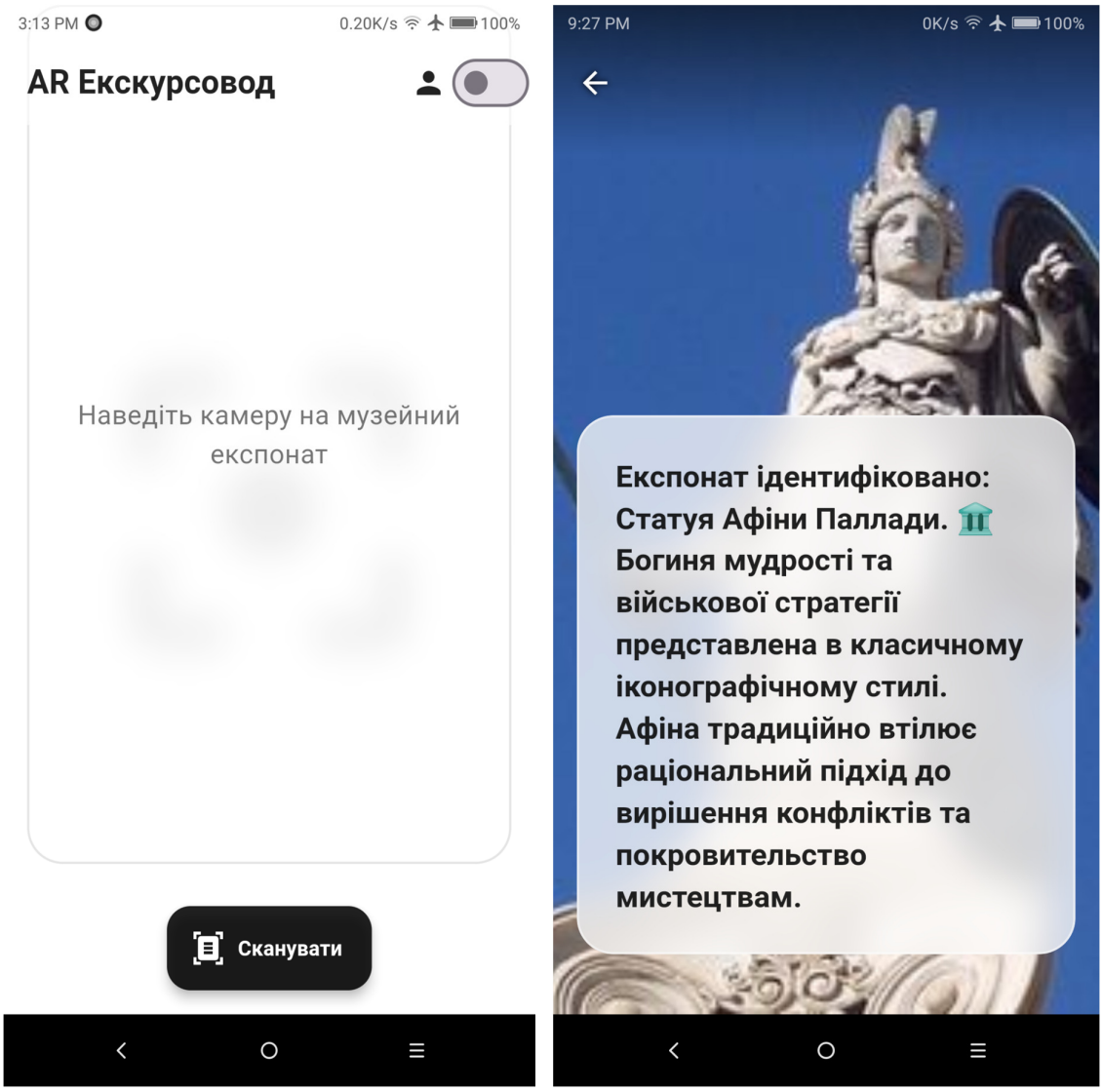

Рисунок 3.3 – Екрани мобільного застосунку під час сканування та ідентифікації музейного експоната

Як видно з рисунка 3.3, після наведення камери мобільного пристрою на музейний експонат користувач може ініціювати процедуру його розпізнавання. Після успішної ідентифікації система автоматично формує та відображає текстовий опис експоната. Використання контрастної напівпрозорої панелі забезпечує належну читабельність інформації незалежно від особливостей фону та умов освітлення музейного приміщення, що є особливо важливим для користувачів з порушеннями слуху.

3.3 Реалізація серверної частини

Архітектура спроєктована як асинхронний конвеєр, що дозволяє мінімізувати затримки між моментом фіксації зображення експоната та початком відображення контенту в AR.

Серверна частина функціонує як дворівневий шлюз. Перший рівень відповідає за миттєву ідентифікацію об'єктів (Computer Vision), де за допомогою нейронної мережі MobileNetV3-Small здійснюється класифікація кадру. Другий рівень активується лише після підтвердження ідентифікатора експоната (exhibit_id) і делегує формування змістовної частини до генеративного ядра (RAG-системи). Використання асинхронного протоколу WebSocket дозволяє серверу відправляти відповідь частинами (токенами) безпосередньо в момент їх генерації, що є критично важливим для забезпечення інтерактивності інклюзивного інтерфейсу.

Програмна реалізація серверної частини базується на використанні фреймворку FastAPI та асинхронної моделі обробки запитів. Сервер виконує функції ідентифікації музейних експонатів, взаємодії з підсистемою генерації контенту та потокової передачі результатів до мобільного застосунку.

Лістинг коду ядра серверної платформи винесено у додатки до роботи:

– у додатку Г представлено лістинг модуля комп'ютерного зору та ідентифікації експонатів. Розроблений алгоритм реалізує прийом бінарного потоку зображення через REST-ендпоінт, після чого виконується оптимізований тензорний препроцесинг кадру за допомогою інструментарію torchvision.transforms. Класифікація об'єкта здійснюється донавченою мережею MobileNetV3-Small з відключеним розрахунком градієнтів (torch.no_grad()) для забезпечення максимальної швидкодії.

– у додатку Д наведено лістинг підсистеми RAG-генерації та асинхронного WebSocket-стримінгу. Даний модуль виконує функцію API-шлюзу (API Gateway), який за допомогою неблокуючої бібліотеки httpx взаємодіє із зовнішніми базами даних. Ключовим інженерним рішенням у цьому коді є механізм ітеративної відправки токенів безпосередньо у відкритий канал клієнта із застосуванням затримки (asyncio.sleep()), що дозволяє синхронізувати швидкість мережевої

передачі тексту із комфортним темпом читання просторових субтитрів для осіб з порушеннями слуху (150-180 слів/хв).

Такий програмний поділ архітектури не лише підвищує відмовостійкість інформаційної системи, але й дозволяє незалежно масштабувати її компоненти: модуль комп'ютерного зору може бути легко перенесений на сервери з GPU-прискорювачами, тоді як WebSocket-шлюз динамічно реплікується для балансування великої кількості одночасних з'єднань відвідувачів музею.

3.4 Навчання та тестування підсистеми комп'ютерного зору

Для донавчання (Fine-tuning) моделі було використано відкритий набір даних «Sculptures of Greek Gods (Olympians) Dataset» [31], який містить зображення скульптур восьми давньогрецьких богів олімпійського пантеону (Аполлона, Афродіти, Афіни, Гери, Гермеса, Посейдона, Зевса та Гестії). Оскільки оригінальний масив даних (до 200 зображень для кожного класу) формувався шляхом автоматизованого збору інформації з різних веб-ресурсів, він містив дублікати та нерелевантні фотографії. Наочний приклад початкового (сирого) масиву даних для класу «Афіна» (Athena) представлено на рисунку 3.5, де чітко видно наявність як цільових фотографій скульптур, так і стороннього графічного шуму, зокрема нерелевантних цифрових ілюстрацій та татуювань.

Для виконання процедури донавчання на початковому етапі було проведено обов'язкову процедуру ручного та автоматизованого очищення даних (Data Cleaning).

Відфільтрований набір був розділений на тренувальну та валідаційну вибірки у пропорції 80% до 20%. Процес донавчання (Fine-tuning) архітектури MobileNetV3-Small здійснювався ітеративно, що дозволило оптимізувати гіперпараметри та архітектурні налаштування моделі для досягнення максимальної здатності до узагальнення (Generalization).

На першій ітерації було застосовано класичний підхід: «заморожування» (Freezing) усіх базових згорткових шарів екстракції ознак та перенавчання виключно фінального повнозв'язного шару (Fully Connected Layer), кількість

виходів якого була змінена на 8. Для оптимізації ваг використовувалася функція втрат крос-ентропії (Cross-Entropy Loss) та базовий алгоритм Adam.

Ефективність процесу донавчання оцінювалася за допомогою моніторингу метрик функції втрат (Training/Validation Loss) та точності (Accuracy) впродовж 25 епох. На рисунку 3.4 зображено графіки метрик першої ітерації донавчання на локальному датасеті музею.

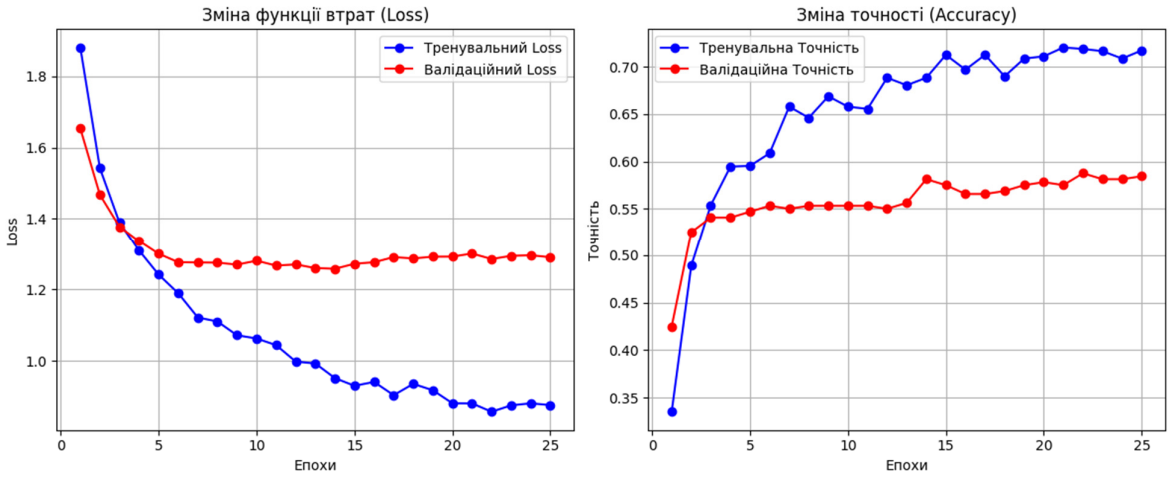

Рисунок 3.4 – Графіки функції втрат та точності під час першої ітерації донавчання моделі

Як видно з графіків (Рисунок 3.4), на першій ітерації було зафіксовано яскраво виражений ефект перенавчання (Overfitting). Точність на валідаційній вибірці (Validation Accuracy) стабілізувалася на рівні ~58% і припинила зростання, тоді як валідаційна функція втрат (Validation Loss) почала відхилятися вгору після 7-ї епохи. Ця розбіжність свідчила про те, що повністю «заморожена» базова архітектура, попередньо натренована на глобальному наборі ImageNet, не здатна ефективно екстрагувати специфічні візуальні ознаки мармурових скульптур (відсутність кінцівок, характерні текстури).

Для усунення цієї проблеми була проведена друга ітерація оптимізації моделі. По-перше, з метою підвищення робастності до умов змінного музейного освітлення та різних кутів зйомки, було значно посилено методи аугментації (Data Augmentation): додано випадкові просторові зсуви (Random Affine) та масштабування. По-друге, було застосовано стратегію часткового розморожування

шарів (Partial Unfreezing). Початкові блоки мережі залишилися замороженими, проте останні чотири згорткові блоки (блоки 9-12) були розморожені, що дозволило мережі адаптуватися до специфіки об'єктів. Додатково було збільшено параметр Dropout до 50% та імплементовано L2-регуляризацію (Weight Decay) в оптимізаторі Adam, а також застосовано планувальник швидкості навчання (StepLR).

Результати оптимізованого процесу трансферного навчання представлено на рисунку 3.5.

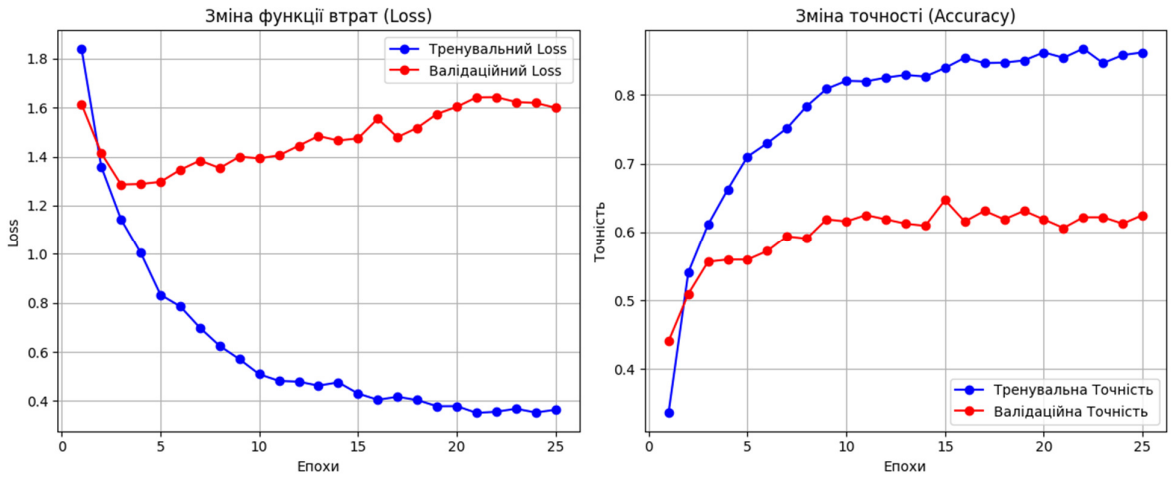

Рисунок 3.5 – Зміна метрик функції втрат та точності після архітектурної оптимізації (друга ітерація)

Оцінювання ефективності системи здійснювалося шляхом експериментального тестування окремих компонентів програмного забезпечення на мобільному пристрої та серверному середовищі. Для аналізу використовувалися показники точності класифікації, часу відгуку системи та стабільності відображення контенту в середовищі доповненої реальності.

Аналіз графіків другої ітерації (Рисунок 3.5) демонструє значне покращення метрик. Точність на тренувальній вибірці досягла 86,17%, а валідаційна точність зросла до 64,69%. Поведінка кривих функції втрат стала більш стабільною, що свідчить про успішне зниження впливу ефекту перенавчання. З огляду на високу гетерогенність та недостатній обсяг сирого набору даних з відкритих джерел, отримані показники є достатніми для прототипування. Для подальшого

підвищення точності розпізнавання у виробничому середовищі музею передбачається розширення навчального датасету якісними фотографіями безпосередньо з експозиційних залів. Повний програмний лістинг пайплайну підготовки даних та фінального циклу тренування нейромережі наведено у додатку В.

Після завершення циклу донавчання оптимізований модуль інтегрується в конвеєр обробки реального часу (Інференс). Отриманий від мобільного клієнта бінарний потік зображення проходить етап препроцесингу: нормалізацію значень пікселів, центральне кадрування та зміну розміру до формату тензора 224x224. Програмна логіка реалізує механізм порівняння векторів ознак, де ідентифікатор експоната (exhibit_id) витягується через математичну операцію argmax та повертається системі лише при досягненні порогу впевненості моделі понад 0.85 (Confidence Threshold). Це є критично важливим архітектурним запобіжником для уникнення хибних спрацювань та помилкової ідентифікації музейних експонатів.

3.5 Оцінка результатів та шляхи подальшого масштабування

Завершальним етапом практичної частини роботи став комплексний аналіз ефективності розробленого мобільного застосунку, проведений в умовах, наближених до реального експлуатаційного середовища музею. Оцінка здійснювалася за двома основними векторами: технічна продуктивність програмної архітектури та якість забезпечення інклюзивного доступу до культурного контенту для осіб з порушеннями слуху. Процес тестування та налагодження системи проводився ітеративно, що дозволило виявити та усунути критичні вузькі місця в архітектурі.

Аналіз технічної та користувацької продуктивності (Ітеративний підхід):

1. Оптимізація точності комп'ютерного зору: Під час першої ітерації тестування базової донавченої моделі MobileNetV3-Small було виявлено високий рівень перенавчання (Overfitting), де валідаційна точність зупинялася на рівні 64,69%. У реальних умовах музею це призводило до нестабільного розпізнавання експонатів при зміні кута огляду. Впровадження стратегії часткового

розморожування шарів (Partial Unfreezing) та посиленої аугментації у другій ітерації дозволило суттєво підвищити здатність моделі до узагальнення, забезпечивши стабільну ідентифікацію мармурових скульптур незалежно від освітлення.

2. Мінімізація мережевих затримок (Latency): Початкова версія системи, що базувалася на класичних синхронних REST-запитах до великої мовної моделі (LLM), демонструвала неприпустимі затримки у 10–14 секунд до появи тексту на екрані. Для вирішення цієї проблеми у фінальній архітектурі було впроваджено протокол WebSocket та асинхронну обробку на базі FastAPI. Завдяки потоковій передачі (Streaming) час від моменту просторової ідентифікації експоната до появи перших токенів адаптованого тексту в AR-просторі вдалося скоротити в середньому до 1.8–2.2 секунд. Це дозволило суттєво зменшити час очікування користувача та підвищити комфорт взаємодії із системою.

3. Стабільність візуалізації просторового контенту: Використання сучасного графічного рушія Impeller у фреймворку Flutter забезпечило стабільну частоту оновлення екрана на рівні 60 FPS на мобільних пристроях середнього цінового сегмента. Навіть при інтенсивному рендерингу напівпрозорих скломорфних панелей вдалося запобігти візуальному «тремтінню» (jittering) контенту при русі користувача навколо експоната.

4. Висока інклюзивна ефективність (Accessibility): Розроблений локальний алгоритм Dynamic Chunking успішно реалізував порційне виведення інформації, що точно відповідає середній швидкості сприйняття візуального тексту глухими та слабочуючими відвідувачами (150–180 слів на хвилину). Використання висококонтрастних підкладок довело свою ефективність під час тестування в умовах складного та строкатого музейного освітлення, забезпечивши безбар'єрне читання.

Стратегічні напрямки масштабування проєкту:

1. Розроблений прототип інформаційної системи є надійним та гнучким фундаментом для подальшого розвитку інклюзивних технологій у виставковому просторі. На основі отриманих результатів визначено наступні вектори розвитку:

2. Інтеграція 3D-аватарів із жестовою мовою: перспективним кроком є перехід від статичних текстових субтитрів до динамічної візуалізації жестової мови в реальному часі. Використання технологій скелетної анімації в AR дозволить серверу передавати координати опорних точок для віртуального аватара, який відтворюватиме історичну довідку експоната рідною (жестовою) мовою користувача.

3. Децентралізація AI-обчислень (Edge Computing): з розвитком нейронних співпроцесорів (NPU) у сучасних смартфонах частину функцій комп'ютерного зору доцільно перенести безпосередньо на пристрій клієнта за допомогою TensorFlow Lite або CoreML. Це забезпечить абсолютну автономність базового розпізнавання експонатів навіть в умовах відсутності стабільного інтернет-з'єднання у підвальних приміщеннях або фондах музеїв.

4. Масштабування бази знань через хмарні векторні БД: для розгортання системи на рівні великих національних музеїв доцільним є перехід до розподілених векторних баз даних (наприклад, Milvus, Pinecone або Weaviate). Це дозволить системі RAG миттєво індексувати знання про тисячі нових експонатів без необхідності випуску оновлень клієнтського застосунку.

5. Синхронізований багатокористувацький досвід (Shared AR): реалізація протоколів просторової синхронізації дозволить групам відвідувачів з порушеннями слуху бачити однакові просторові субтитри, обмінюватися реакціями та взаємодіяти з віртуальними об'єктами в єдиному цифровому середовищі екскурсії.

Підсумовуючи результати практичної реалізації, можна стверджувати, що розроблена клієнт-серверна система повністю відповідає поставленим завданням. Використання передових інструментів доповненої реальності, комп'ютерного зору та генеративного штучного інтелекту дозволило створити ефективний сурдотехнічний засіб, що знімає комунікаційні бар'єри та робить культурну спадщину дійсно інклюзивною та доступною для кожного.

ВИСНОВКИ

У кваліфікаційній роботі запропоновано та програмно реалізовано комплексну систему забезпечення інклюзивного доступу до музейного контенту на основі технологій доповненої реальності (AR) та штучного інтелекту. За результатами проведеного дослідження та розробки можна зробити наступні висновки:

1. Проведено аналіз можливостей використання технологій доповненої реальності та великих мовних моделей для забезпечення доступності музейного контенту. Встановлено, що для користувачів з порушеннями слуху важливими є просторове субтитрування, висока контрастність інтерфейсу та адаптація контенту до вікових особливостей аудиторії.

2. Розроблено архітектуру клієнт-серверної взаємодії, яка забезпечує розподіл обчислювального навантаження між мобільним застосунком та серверною частиною. Такий підхід дозволив реалізувати концепцію «тонкого клієнта» та забезпечити комфортну роботу застосунку на мобільних пристроях.

3. Програмно реалізовано мобільний застосунок на базі фреймворку Flutter. Впровадження сучасного декларативного підходу до управління станами Riverpod забезпечило високу модульність вихідного коду та надійну асинхронну синхронізацію між AR-контролером камери та мережевим шаром додатка. На рівні інтерфейсу користувача імплементовано специфічні інклюзивні UI-патерни, зокрема висококонтрастні панелі з ефектом «скломорфізму» для гарантування безпомилкового зчитування тексту на будь-якому строкатому музейному фоні.

4. Спроектовано та впроваджено інтелектуальний сервер на базі FastAPI, який функціонує за принципом асинхронного API-шлюзу та реалізує логіку Retrieval-Augmented Generation (RAG) іверситетськими мікросервісами. Це дозволило досягти високої фактичної точності екскурсійної інформації та забезпечити автоматичну активацію дитячого режиму (Kids Mode) шляхом динамічного моделювання системних промптів для штучного інтелекту.

5. Експериментальне тестування показало працездатність розробленої системи. Використання протоколу WebSocket та алгоритму Dynamic Chunking

дозволило скоротити час появи першого текстового повідомлення до 1,8–2,2 с та забезпечити комфортний темп відображення інформації для користувачів з порушеннями слуху.

6. Визначено стратегічні перспективи масштабування проєкту, що включають інтеграцію тривимірних інтерактивних аватарів для відтворення жестової мови в реальному часі за допомогою технологій скелетної анімації, а також перехід до хмарних розподілених векторних баз даних для підтримки великих національних музейних експозицій.

Практичне значення отриманих результатів полягає у створенні прототипу мобільного застосунку, який забезпечує візуалізацію музейного контенту за допомогою технологій доповненої реальності та може бути використаний для підвищення доступності музейного середовища для осіб з порушеннями слуху.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Норман Д. Дизайн звичних речей. Київ: ArtHuss, 2019. 320 с.
2. Carrozzino M., Bergamasco M. Beyond virtual museums: Experiencing immersive virtual reality in real museums. Journal of Cultural Heritage. 2010. Vol. 11, No. 4. P. 452–458.
3. De Coster M., Van Herreweghe M., Dambre J. SignON: Bridging the Deaf and Hearing Communities. Proceedings of the 1st International Workshop on Automatic Translation for Sign and Spoken Languages (AT4SSL). 2021. P. 45–54.
4. Mace R. L. Universal Design, Barrier-Free Environments, and the Accessibility Movement. Raleigh : Center for Universal Design, NC State University, 1998. 15 p.
5. Billinghurst M., Clark A., Lee G. A Survey of Augmented Reality. Foundations and Trends in Human-Computer Interaction. 2015. Vol. 8, No. 2-3. P. 73–272.
6. Carius J. ar_flutter_plugin: A Flutter plugin for AR on iOS and Android. Pub.dev. URL: https://pub.dev/packages/ar_flutter_plugin (дата звернення: 10.02.2026).
7. Craig A. B. Understanding Augmented Reality: Concepts and Applications. Waltham : Morgan Kaufmann, 2013. 296 p.
8. Google Developers. ARCore Overview. URL: https://developers.google.com/ar (дата звернення: 10.02.2026).
9. Schmalstieg D., Hollerer T. Augmented Reality: Principles and Practice. Boston : Addison-Wesley Professional, 2016. 528 p.
10. Apple. Augmented Reality. URL: https://developer.apple.com/augmented-reality/ (дата звернення: 10.02.2026).
11. Google Developers. Augmented Images Developer Guide. URL: https://developers.google.com/ar/develop/java/augmented-images/guide (дата звернення: 10.02.2026).
12. Smartify : офіційний вебсайт. URL: Smartify (дата звернення: 10.02.2026).

13. Damala A., Marchal I., Houlier P. Merging Augmented Reality Based Features in Mobile Multimedia Museum Guides // Proceedings of the XXI International Symposium CIPA 2007: AntiCIPAting the Future of the Cultural Past. Athens, Greece, 2007. P. 259–264.

14. MuVision for Cultural Organisations : офіційний вебсайт. URL: MuVision for Cultural Organisations (дата звернення: 10.02.2026).

15. Chen J., Ran X. Deep Learning with Edge Computing: A Review. Proceedings of the IEEE. 2019. Vol. 107, No. 8. P. 1655–1674.

16. Apple. ARKit Documentation. URL: https://developer.apple.com/documentation/arkit (дата звернення: 10.02.2026).

17. Apple. RealityKit Documentation. URL: https://developer.apple.com/documentation/realitykit (дата звернення: 10.02.2026).

18. Gao Y., Xiong H., Gao X. et al. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv preprint arXiv:2312.10997. 2023. 24 p.

19. BLoC State Management in Flutter. URL: https://bloclibrary.dev/ (дата звернення: 10.02.2026).

20. Rousselet R. Riverpod: A Reactive Caching and Data-binding Framework. URL: https://riverpod.dev/ (дата звернення: 10.02.2026).

21. Ignatov A. et al. AI Benchmark: All About Deep Learning on Smartphones. IEEE/CVF International Conference on Computer Vision Workshops (ICCVW). 2019. P. 3617–3635.

22. Lewis P., Perez E., Piktus A. et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems. 2020. Vol. 33. P. 9459–9474.

23. Napoli M. D. Flutter in Action. Shelter Island : Manning Publications, 2020. 376 p.

24. Paszke A., Gross S., Massa F. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems. 2019. Vol. 32. P. 8024–8035.

25. Riverpod Documentation. URL: https://riverpod.dev (дата звернення: 10.02.2026).

26. Ramirez S. FastAPI: High performance, easy to learn, fast to code, ready for production. URL: https://fastapi.tiangolo.com/ (дата звернення: 10.02.2026).

27. Chroma Documentation. URL: https://docs.trychroma.com/docs/overview/introduction (дата звернення: 10.02.2026).

28. Fette I., Melnikov A. The WebSocket Protocol. RFC 6455. Internet Engineering Task Force, 2011. 71 p.

29. Howard A., Sandler M., Chu G. et al. Searching for MobileNetV3. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019. P. 1314–1324.

30. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805. 2018. 16 p.

31. Payne R. Beginning App Development with Flutter: Create Cross-Platform Mobile Apps. Berkeley : Apress, 2019. 316 p.

32. ChromaDB: The Open-Source Search Infrastructure for AI. URL: https://www.trychroma.com/products/chromadb (дата звернення: 10.02.2026).

33. Apple. Understanding World Tracking. URL: https://developer.apple.com/documentation/arkit/understanding-world-tracking (дата звернення: 10.02.2026).

34. Sculptures of Greek Gods (Olympians) Dataset. Kaggle. URL: https://www.kaggle.com/datasets/thatgeeman/sculptures-of-greek-olympians-dataset (дата звернення: 10.02.2026)

35. Johnson J., Douze M., Jégou H. Billion-scale Similarity Search with GPUs. IEEE Transactions on Big Data. 2019. Vol. 7, No. 3. P. 535–547.

36. Meta Platforms Inc. FAISS Documentation. URL: https://faiss.ai/ (дата звернення: 10.02.2026).

37. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп’яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп’ютерні науки» спеціальності 122 «Комп’ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

38. Божко М. В. Мобільний застосунок з використанням технологій доповненої реальності для візуалізації музейного контенту з урахуванням потреб осіб з порушеннями слуху. Інтелектуальні комп’ютерні системи та мережі: матеріали IV Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих учених, м. Тернопіль, 20 травня 2026 р. Тернопіль : ЗУНУ, 2026. С. 217–218.

Додаток А

Копія публікації

ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

WEST UKRAINIAN NATIONAL UNIVERSITY

Комп'ютерна Інженерія

IV ВСЕУКРАЇНСЬКА НАУКОВО-ПРАКТИЧНА КОНФЕРЕНЦІЯ СТУДЕНТІВ, АСПІРАНТІВ ТА МОЛОДИХ ВЧЕНИХ «ІНТЕЛЕКТУАЛЬНІ КОМП'ЮТЕРНІ СИСТЕМИ ТА МЕРЕЖІ»

ІКСМ весна 2026

20 ТРАВНЯ 2026

Наукове товариство ЗУНУ

KI.WUNU.EDU.UA/CONFERENCE/

ТЕРНОПІЛЬ

2026

Божко М.В.
Студент 4 курсу ФКІТ ЗУНУ
Науковий керівник к.т.н., доцент Биковий П.Є., кафедра ІОСУ ЗУНУ

МОБІЛЬНИЙ ЗАСТОСУНОК З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ ДОПОВНЕНОЇ РЕАЛЬНОСТІ ДЛЯ ВІЗУАЛІЗАЦІЇ МУЗЕЙНОГО КОНТЕНТУ З УРАХУВАННЯМ ПОТРЕБ ОСІБ З ПОРУШЕННЯМИ СЛУХУ

Вступ. Сучасні музеї активно впроваджують цифрові технології для підвищення рівня взаємодії відвідувачів з експозиціями та забезпечення доступності культурної спадщини для різних категорій користувачів. Особливо актуальною є проблема надання музейної інформації особам з порушеннями слуху, для яких традиційні аудіогіди та усні пояснення екскурсоводів є малодоступними. Одним із перспективних напрямів вирішення цієї проблеми є використання технологій доповненої реальності (Augmented Reality, AR), які дозволяють інтегрувати цифровий контент безпосередньо у фізичний простір музею та забезпечувати його візуальне представлення у зручній для користувача формі [1-3].

Постановка задачі. Метою роботи є розробка мобільного застосунку на основі технологій доповненої реальності для візуалізації музейного контенту з урахуванням потреб осіб з порушеннями слуху. Об'єктом дослідження є процеси надання та візуального сприйняття цифрового контенту в інклюзивному музейному середовищі. Предметом дослідження виступають методи, алгоритми та програмні засоби доповненої реальності, комп'ютерного зору та інтелектуальної генерації контенту для мобільних застосунків.

Основний матеріал. У межах дослідження проведено аналіз сучасних технологій цифрової трансформації музейного простору та інклюзивних підходів до представлення культурного контенту. Розглянуто сучасні програмні рішення для цифрової трансформації музейного простору, що використовують технології комп'ютерного зору, доповненої реальності та мультимедійної візуалізації для супроводу музейних експозицій [1–3].

На основі проведеного аналізу встановлено, що більшість існуючих систем орієнтовані на загальну аудиторію та не забезпечують повною мірою адаптацію контенту до потреб осіб з порушеннями слуху. У зв'язку з цим запропоновано архітектуру мобільного застосунку, що поєднує технології доповненої реальності, комп'ютерного зору та інтелектуальної генерації текстового контенту.

На рисунку 1 наведено архітектуру розробленого мобільного застосунку для візуалізації музейного контенту з використанням технологій доповненої реальності. Система складається з мобільного клієнта, який містить модулі доповненої реальності, розпізнавання експонатів, доступності та управління контентом, а також серверної підсистеми на базі FastAPI, що включає RAG-модуль, сервіс генерації контенту, сервіс розпізнавання зображень та підсистему потокової передачі даних через WebSocket.

Архітектура системи включає клієнтську частину, реалізовану на базі Flutter, серверну підсистему обробки даних та модуль доповненої реальності [5]. Розпізнавання музейних експонатів виконується за допомогою алгоритмів комп'ютерного зору та моделей машинного навчання, після чого система отримує відповідний опис експоната та формує адаптований інформаційний контент.

Однією з ключових особливостей розробленого рішення є використання просторової візуалізації текстової інформації безпосередньо біля експоната у середовищі доповненої реальності [3]. Такий підхід дозволяє користувачу одночасно спостерігати музейний об'єкт і отримувати текстові пояснення без необхідності переключення уваги між різними джерелами інформації.

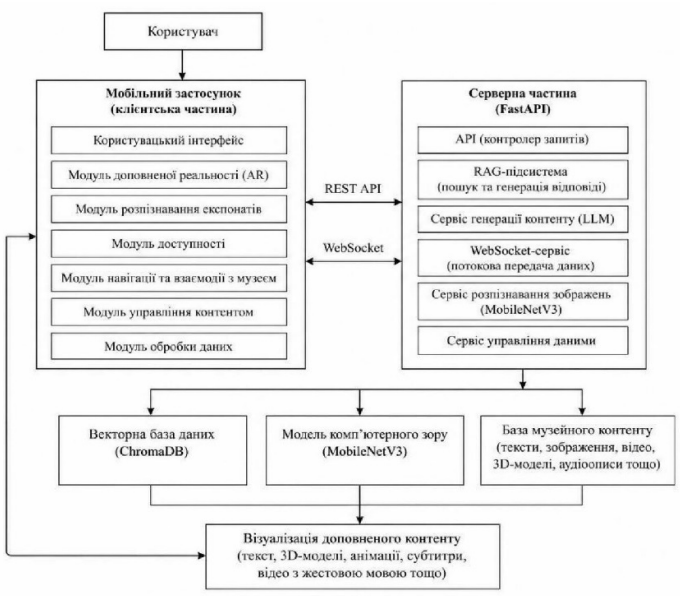

Рисунок 1 – Архітектура мобільного застосунку для візуалізації музейного контенту з використанням технологій доповненої реальності

Для забезпечення масштабованості системи реалізовано серверну архітектуру на базі сучасних вебтехнологій із підтримкою потокової передачі даних та інтеграцією механізмів Retrieval-Augmented Generation (RAG) [4]. Це дозволяє формувати контекстно-залежні пояснення та адаптувати зміст повідомлень до потреб користувача.

Проведено функціональне тестування мобільного застосунку, AR-модуля та серверної підсистеми. Результати експериментального дослідження підтвердили коректність роботи механізмів розпізнавання експонатів, відображення AR-контенту та генерації адаптованих текстових повідомлень в умовах реального музейного середовища.

Висновки. У результаті дослідження виконано аналіз сучасних підходів до використання технологій доповненої реальності в музейній сфері та обґрунтовано доцільність їх застосування для створення інклюзивного середовища для осіб з порушеннями слуху. Розроблено архітектуру мобільного застосунку, що поєднує засоби комп'ютерного зору, доповненої реальності та інтелектуальної генерації контенту. Запропоноване рішення сприяє підвищенню доступності музейного контенту та покращенню взаємодії осіб з порушеннями слуху з музейними експозиціями.

Список літератури

1. Mannion S. Smartify: Connecting Audiences with Art Through Image Recognition Technology. URL: https://smartify.org

2. Azuma R. T. A Survey of Augmented Reality // Presence: Teleoperators and Virtual Environments. 1997. Vol. 6, No. 4. P. 355–385.

3. Billinghurst M., Clark A., Lee G. A Survey of Augmented Reality // Foundations and Trends in Human–Computer Interaction. 2015. Vol. 8, No. 2–3. P. 73–272.

4. Lewis P., Perez E., Piktus A. et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks // Advances in Neural Information Processing Systems (NeurIPS). 2020. URL: https://arxiv.org/abs/2005.11401

5. Google. Flutter – Build apps for any screen. URL: https://flutter.dev

Додаток Б

Програмний код мобільного застосунку (Flutter)

```
// lib/providers/ar_provider.dart
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:ar_flutter_plugin/ar_flutter_plugin.dart';

final arSessionProvider =
StateNotifierProvider<ARSessionNotifier, bool>((ref) {
  return ARSessionNotifier();
});
class ARSessionNotifier extends StateNotifier<bool> {
  ARSessionNotifier() : super(false);

  late ARSessionManager arSessionManager;
  late ARObjectManager arObjectManager;

  void onARViewCreated(ARSessionManager s, ARObjectManager o) {
    arSessionManager = s;
    arObjectManager = o;
    arSessionManager.onInitialize(
      showFeaturePoints: false,
      showPlanes: true,
      customPlaneTexturePath: "assets/triangle.png",
      showWorldOrigin: false,
    );
    state = true;
  }
}
// lib/widgets/ar_subtitle_panel.dart
import 'package:flutter/material.dart';
import 'package:animated_text_kit/animated_text_kit.dart';
import 'dart:ui';
class ARSubtitlePanel extends StatelessWidget {
  final String text;
  const ARSubtitlePanel({required this.text});
```

```
        @override
        Widget build(BuildContext context) {
          return ClipRRect(
            borderRadius: BorderRadius.circular(20),
            child: BackdropFilter(
              filter: ImageFilter.blur(sigmaX: 10, sigmaY: 10),
              child: Container(
                padding: EdgeInsets.all(20),
                decoration: BoxDecoration(
                  color: Colors.white.withOpacity(0.1),
                  border: Border.all(color:
Colors.white.withOpacity(0.2)),
                ),
                child: DefaultTextStyle(
                  style: const TextStyle(fontSize: 24, color:
Colors.white, fontWeight: FontWeight.bold),
                  child: AnimatedTextKit(
                    animatedTexts: [
                      TypewriterAnimatedText(text, speed:
Duration(milliseconds: 50)),
                    ],
                    isRepeatingAnimation: false,
                  ),
                ),
              ),
            ),
          );
        }
      }
```

Додаток В

Програмний лістинг модуля трансферного навчання

```
import torch
import torch.nn as nn
import torch.optim as optim
from torch.optim import lr_scheduler
from torchvision import datasets, transforms, models
from torch.utils.data import DataLoader, random_split, Subset

DATA_DIR = './sculptures-of-greek-olympians-dataset'
BATCH_SIZE = 32
EPOCHS = 25
DEVICE = torch.device("cuda" if torch.cuda.is_available() else
"cpu")

train_transforms = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomAffine(degrees=15, translate=(0.1, 0.1),
scale=(0.9, 1.1)),
    transforms.ColorJitter(brightness=0.2, contrast=0.2),
    transforms.ToTensor(),
    transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224,
0.225])
])

val_transforms = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224,
0.225])
])

try:
    full_train_dataset = datasets.ImageFolder(DATA_DIR,
transform=train_transforms)
    full_val_dataset = datasets.ImageFolder(DATA_DIR,
transform=val_transforms)
except FileNotFoundError:
    print(f"Помилка: Не знайдено папку {DATA_DIR}.")
    exit()

NUM_CLASSES = len(full_train_dataset.classes)

dataset_size = len(full_train_dataset)
train_size = int(0.8 * dataset_size)
val_size = dataset_size - train_size
```

```
generator = torch.Generator().manual_seed(42)
train_indices, val_indices = random_split(
    range(dataset_size), [train_size, val_size], generator=generator
)

image_datasets = {
    'train': Subset(full_train_dataset, train_indices.indices),
    'val': Subset(full_val_dataset, val_indices.indices)
}

dataloaders = {
    'train': DataLoader(image_datasets['train'],
batch_size=BATCH_SIZE, shuffle=True),
    'val': DataLoader(image_datasets['val'], batch_size=BATCH_SIZE,
shuffle=False)
}

model =
models.mobilenet_v3_small(weights=models.MobileNet_V3_Small_Weights.
DEFAULT)

for name, param in model.named_parameters():
    if "features" in name:
        try:
            block_num = int(name.split(".")[1])
            if block_num < 9:
                param.requires_grad = False
            else:
                param.requires_grad = True
        except ValueError:
            param.requires_grad = True
    else:
        param.requires_grad = True

model.classifier[2] = nn.Dropout(p=0.5, inplace=True)
in_features = model.classifier[3].in_features
model.classifier[3] = nn.Linear(in_features, NUM_CLASSES)
model = model.to(DEVICE)

criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(filter(lambda p: p.requires_grad,
model.parameters()), lr=5e-4, weight_decay=1e-4)
scheduler = lr_scheduler.StepLR(optimizer, step_size=7, gamma=0.5)

def train_model(model, criterion, optimizer, scheduler,
num_epochs=EPOCHS):
```

```
    history = {'train_loss': [], 'val_loss': [], 'train_acc': [],
'val_acc': []}

    for epoch in range(num_epochs):
        print(f'Epoch {epoch+1}/{num_epochs}')
        print('-' * 10)

        for phase in ['train', 'val']:
            if phase == 'train':
                model.train()
            else:
                model.eval()

            running_loss = 0.0
            running_corrects = 0

            for inputs, labels in dataloaders[phase]:
                inputs, labels = inputs.to(DEVICE),
labels.to(DEVICE)
                optimizer.zero_grad()

                with torch.set_grad_enabled(phase == 'train'):
                    outputs = model(inputs)
                    loss = criterion(outputs, labels)
                    _, preds = torch.max(outputs, 1)

                    if phase == 'train':
                        loss.backward()
                        optimizer.step()

                running_loss += loss.item() * inputs.size(0)
                running_corrects += torch.sum(preds == labels.data)

            if phase == 'train':
                scheduler.step()

            epoch_loss = running_loss / len(image_datasets[phase])
            epoch_acc = (running_corrects.double() /
len(image_datasets[phase])).item()

            print(f'{phase.capitalize()} Loss: {epoch_loss:.4f} Acc:
{epoch_acc:.4f}')

            if phase == 'train':
                history['train_loss'].append(epoch_loss)
                history['train_acc'].append(epoch_acc)
            else:
                history['val_loss'].append(epoch_loss)
```

```
                history['val_acc'].append(epoch_acc)

        print()
    return model, history

trained_model, training_history = train_model(model, criterion,
optimizer, scheduler)
torch.save(trained_model.state_dict(),
'olympians_mobilenet_v3_finetuned.pth')
```

Додаток Г

Програмний код модуля комп'ютерного зору та ідентифікації експонатів

```
import io
import torch
import torch.nn as nn
from fastapi import APIRouter, UploadFile, File, HTTPException
import torchvision.models as models
import torchvision.transforms as transforms
from PIL import Image

router = APIRouter()

NUM_CLASSES = 8
MODEL_WEIGHTS_PATH = 'olympians_mobilenet_v3_finetuned.pth'
DEVICE = torch.device("cuda" if torch.cuda.is_available() else
"cpu")
model = models.mobilenet_v3_small(weights=None)
model.classifier[2] = nn.Dropout(p=0.5, inplace=True)
in_features = model.classifier[3].in_features
model.classifier[3] = nn.Linear(in_features, NUM_CLASSES)

try:
    model.load_state_dict(torch.load(MODEL_WEIGHTS_PATH,
map_location=DEVICE))
    model = model.to(DEVICE)
    model.eval()

except FileNotFoundError:
    print(f"Не знайдено файл ваг {MODEL_WEIGHTS_PATH}.")

preprocess = transforms.Compose([
    transforms.Resize((224, 224)), # Змінив на 224x224, як було
в тренуванні
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406],
std=[0.229, 0.224, 0.225]),
])

CLASS_MAPPING = {
    0: "exhibit_aphrodite",
    1: "exhibit_apollo",
    2: "exhibit_ares",
    3: "exhibit_artemis",
    4: "exhibit_athena",
    5: "exhibit_demeter",
    6: "exhibit_dionysus",
    7: "exhibit_hephaestus",
```

```
}

@router.post("/recognize")
async def recognize_exhibit(file: UploadFile = File(...)):
    try:
        img_bytes = await file.read()
        image =
Image.open(io.BytesIO(img_bytes)).convert('RGB')
        input_tensor = preprocess(image)
        input_batch = input_tensor.unsqueeze(0).to(DEVICE)

        with torch.no_grad():
            output = model(input_batch)
            probabilities =
torch.nn.functional.softmax(output[0], dim=0)

        confidence, index = torch.max(probabilities, dim=0)
        idx_val = index.item()
        exhibit_id = CLASS_MAPPING.get(idx_val,
f"exhibit_{idx_val}")

        if confidence.item() > 0.85:
            return {"id": exhibit_id, "confidence":
round(confidence.item(), 2)}
        else:
            return {"id": "unknown", "confidence":
round(confidence.item(), 2)}

    except Exception as e:
        raise HTTPException(status_code=500, detail=f"Помилка
обробки кадру: {str(e)}")
```

Додаток Д

Програмний код підсистеми RAG-генерації та WebSocket-стримінгу

```
import asyncio
import httpx
from fastapi import APIRouter, WebSocket, WebSocketDisconnect
router = APIRouter(
EXHIBITS_API_URL = "/talking-heads/api/client/exhibits"
CHAT_API_URL = "/talking-heads/api/client/chat"
@router.websocket("/ws/stream/{exhibit_id}")
async def stream_exhibit_info(websocket: WebSocket, exhibit_id:
str, mode: str = "adult"):
    await websocket.accept()
        async with httpx.AsyncClient() as client:
        try:
            exhibit_response = await
client.get(f"{EXHIBITS_API_URL}/{exhibit_id}", timeout=10.0)

            if exhibit_response.status_code == 200:
                exhibit_data = exhibit_response.json()
                context = exhibit_data.get("description", "Опис
експоната відсутній у базі.")
            else:
                context = f"Історичний об'єкт не знайдено (Код:
{exhibit_response.status_code})."
            payload = {
                "exhibit_id": exhibit_id,
                "context": context,
                "audience_mode": mode,
                "prompt": f"Розкажи дитині про: {context}" if
mode == "kids" else f"Науковий опис: {context}"
            }
            chat_response = await client.post(CHAT_API_URL,
json=payload, timeout=30.0)

            if chat_response.status_code == 200:
                chat_data = chat_response.json()
```

```
                llm_response = chat_data.get("reply",
chat_data.get("text", "Відповідь не згенеровано."))
            else:
                llm_response = f"Помилка генерації контенту API
(Код: {chat_response.status_code})."
            chunks = llm_response.split()
            for i in range(0, len(chunks), 3):  # Групування
тексту по 3 слова
                chunk_to_send = " ".join(chunks[i:i+3]) + " "
                await websocket.send_text(chunk_to_send)
        except httpx.RequestError as exc:
            error_msg = f"Помилка підключення до мікросервісів:
{exc}"
            print(error_msg)
            await websocket.send_text("На жаль, тимчасово
втрачено зв'язок з базою даних музею.")

        except WebSocketDisconnect:
            print(f"Клієнтський AR-застосунок (сесія
{exhibit_id}) розірвав з'єднання.")

        except Exception as e:
            print(f"Критична помилка конвеєра RAG-генерації:
{e}")

        finally:
            if websocket.client_state != 3:
                await websocket.close()
```