НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«До захисту допущено»
Завідувач кафедри
_____ Євгенія СУЛЕМА
«___» _____ 2025 р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інженерія програмного забезпечення мультимедійних та інформаційно-пошукових систем»

спеціальності 121 Інженерія програмного забезпечення

на тему: «Мобільний застосунок «Exhibition Guide» для пропозиції та пошуку товарів на виставці»

Виконав:
студент IV курсу, групи КП-13
Степаненко Олександр Вадимович _____

Керівник:
доцент кафедри ПЗКС, к.т.н., доцент,
Люшенко Леся Анатоліївна _____

Консультант з нормоконтролю:
доцент кафедри ПЗКС, к.т.н., доцент,
Онай Микола Володимирович _____

Рецензент:
доцент кафедри СПСКС, к.т.н., доцент,
Тарасенко-Клятченко Оксана Володимирівна _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет прикладної математики
Кафедра програмного забезпечення комп'ютерних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення мультимедійних та інформаційно-пошукових систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Євгенія СУЛЕМА
«___»_____2024 р.

ЗАВДАННЯ

на дипломний проєкт студенту

Степаненку Олександру Вадимовичу

1. Тема проєкту «Мобільний застосунок «Exhibition Guide» для пропозиції та пошуку товарів на виставці», керівник проєкту Люшенко Леся Анатоліївна, доцент кафедри ПЗКС, к.т.н., доцент, затверджені наказом по університету № 1808-С від «29» травня 2025 р.
2. Термін подання студентом проєкту «13» червня 2025 р.
3. Вихідні дані до проєкту: див. Технічне завдання.
4. Зміст пояснювальної записки:
   - аналіз існуючих рішень поставленої задачі;
   - обґрунтування вибору засобів реалізації;
   - розроблення мобільного застосунку;
   - аналіз розробленого мобільного застосунку.
5. Перелік обов'язкового графічного матеріалу:
   - діаграма діяльності основного сценарію використання (креслення);
   - логічна модель бази даних (креслення);
   - діаграма компонентів мобільного застосунку (креслення);
   - дії користувача мобільного застосунку «Exhibition Guide» (плакат);
   - архітектура мобільного застосунку (плакат).

6. Консультанти розділів проєкту

7. Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Онай М.В., доцент		

7. Дата видачі завдання «31» жовтня 2024 р.

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	Вивчення літератури за тематикою проєкту	12.11.2024	
2.	Розроблення та узгодження технічного завдання	22.11.2024	
3.	Розроблення структури мобільного застосунку	11.12.2024	
4.	Підготовка матеріалів першого розділу дипломного проєкту	20.12.2024	
5.	Розроблення дизайну сторінок та графічних елементів	21.02.2025	
6.	Підготовка матеріалів другого розділу дипломного проєкту	13.03.2025	
7.	Програмна реалізація мобільного застосунку	25.03.2025	
8.	Тестування мобільного застосунку	18.04.2025	
9.	Підготовка матеріалів третього розділу дипломного проєкту	24.04.2025	
10.	Підготовка матеріалів четвертого розділу дипломного проєкту	03.05.2025	
11.	Підготовка графічної частини дипломного проєкту	12.05.2025	
12.	Оформлення документації дипломного проєкту	23.05.2025	

Студент Олександр СТЕПАНЕНКО

Керівник проєкту Леся ЛЮШЕНКО

АНОТАЦІЯ

Дипломний проєкт присвячений розробці мобільного застосунку «Exhibition Guide» для Android-платформи, який забезпечує ефективну навігацію виставковими павільйонами, перегляд інформації про експонати, категоризацію стендів, збереження улюблених об’єктів.

У межах роботи проведено аналіз предметної області, обґрунтовано актуальність створення мобільного рішення для виставкової діяльності, здійснено огляд існуючих аналогів і визначено їхні обмеження. Обрано оптимальні інструменти розробки: мову програмування Kotlin, архітектурний патерн MVVM, реляційну базу даних SQLite, а також компоненти Android Jetpack для реалізації інтерфейсу.

Застосунок реалізовано у вигляді локального Android-рішення, яке забезпечує доступ до основних можливостей навігації, пошуку та збереження інформації. База даних включає таблиці для зберігання експонатів, категорій, виставок, учасників та списку обраного. Архітектура проєкту забезпечує чіткий поділ відповідальності між інтерфейсом користувача, логікою представлення та джерелом даних.

Інтерфейс адаптований для мобільних пристроїв, має інтуїтивну навігацію та зручну взаємодію з контентом. Реалізовано основні екрани: головна сторінка, деталізація експоната, пошук, категорії та обране. Якість коду оцінено за допомогою метрик складності та зв’язаності, що підтвердило модульність і підтримуваність архітектури.

У результаті розроблено стабільний і зручний мобільний застосунок, що відповідає сучасним вимогам до користувацького інтерфейсу та сприяє підвищенню ефективності відвідування виставок.

ABSTRACT

This diploma project is dedicated to the development of the «Exhibition Guide» mobile application for the Android platform, which provides effective navigation through exhibition halls, viewing information about exhibits, stand categorization, and saving favorite items.

The project includes an analysis of the subject area, justification of the relevance of developing a mobile solution for exhibition activities, a review of existing analogs, and identification of their limitations. The optimal development tools were selected: the Kotlin programming language, the MVVM architectural pattern, the SQLite relational database, and Android Jetpack components for user interface implementation.

The application is implemented as a local Android solution that provides access to core features such as navigation, search, and data saving. The database includes tables for storing exhibits, categories, exhibitions, exhibitors, and favorite items. The architecture ensures a clear separation of responsibilities between the user interface, presentation logic, and data source.

The interface is adapted for mobile devices and provides intuitive navigation and convenient interaction with content. The main screens include the home page, exhibit details, search, categories, and favorites. Code quality was evaluated using complexity and coupling metrics, which confirmed the modularity and maintainability of the architecture.

As a result, a stable and user-friendly mobile application has been developed that meets modern user interface requirements and improves the efficiency of visiting exhibitions.

ДП.045430-01-90 Мобільний застосунок «Exhibition Guide» для пропозиції та пошуку товарів на виставці. Відомість проєкту

Позначення	Найменування	Кіл-ть	Примітка
	Документація проєкту		
ДП.045430-02-91	Мобільний застосунок	5	
	«Exhibition Guide» для		
	пропозиції та пошуку		
	товарів на виставці.		
	Технічне завдання		
ДП.045430-03-81	Мобільний застосунок	57	
	«Exhibition Guide» для		
	пропозиції та пошуку		
	товарів на виставці.		
	Пояснювальна записка		
ДП.045430-04-51	Мобільний застосунок	4	
	«Exhibition Guide» для		
	пропозиції та пошуку		
	товарів на виставці.		
	Програма та методика		
	тестування		
ДП.045430-05-34	Мобільний застосунок	9	
	«Exhibition Guide» для		
	пропозиції та пошуку		
	товарів на виставці.		
	Керівництво користувача		

Позначення	Найменування	Кіл-ть	Примітка
ДП.045430-06-99	Мобільний застосунок	1	
	«Exhibition Guide» для		
	пропозиції та. пошуку		
	товарів на виставці.		
	Діаграма діяльності		
	основного сценарію		
	використання. Схема		
	алгоритму		
ДП.045430-07-99	Мобільний застосунок	1	
	«Exhibition Guide» для		
	пропозиції та. пошуку		
	товарів на виставці.		
	Логічна модель бази		
	даних. Схема даних		
ДП.045430-08-99	Мобільний застосунок	1	
	«Exhibition Guide» для		
	пропозиції та пошуку		
	товарів на виставці.		
	Діаграма компонентів		
	мобільного застосунку.		
	Схема взаємодії		
	програмних модулів		
ДП.045430-09-98	Мобільний застосунок	1	
	«Exhibition Guide» для		
	пропозиції та пошуку		
	товарів на виставці.		
	Компакт-диск		

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

"ЗАТВЕРДЖЕНО"

Завідувач кафедри

_____ Євгенія СУЛЕМА

"___" _____ 2024 р.

МОБІЛЬНИЙ ЗАСТОСУНОК «EXHIBITION GUIDE» ДЛЯ ПРОПОЗИЦІЇ ТА ПОШУКУ ТОВАРІВ НА ВИСТАВЦІ

Технічне завдання

ДП.045430-02-91

"ПОГОДЖЕНО"

Керівник проєкту:

_____ Леся ЛЮШЕНКО

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Олександр СТЕПАНЕНКО

2024

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ

Назва розробки: мобільний застосунок «Exhibition Guide» для пропозиції та пошуку товарів на виставці.

Галузь застосування: інформаційні технології.

2. ПІДСТАВА ДЛЯ РОЗРОБЛЕННЯ

Підставою для розроблення є завдання на дипломне проєктування, затверджене кафедрою програмного забезпечення комп'ютерних систем Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» (КПІ ім. Ігоря Сікорського).

3. ПРИЗНАЧЕННЯ РОЗРОБКИ

Розробка призначена для використання відвідувачами виставок в якості програмного забезпечення для навігації експозиціями, перегляду інформації про експонати та формування списку обраного.

4. ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ

Мобільний застосунок повинен забезпечувати такі основні функції:

1) перегляд експонатів на головному екрані із короткими описами та зображеннями;
2) детальний перегляд кожного експоната з інформацією про назву, опис, категорію, виставку та компанію-учасника;
3) додавання експонатів до списку обраного (Favorites) з можливістю перегляду, видалення та повернення через «Undo»;
4) пошук експонатів за ключовими словами через спеціальну сторінку пошуку;

5) навігація за категоріями експонатів для зручного фільтрування вмісту.

Додаткові вимоги:

1) стабільна робота застосунку в офлайн-режимі на період проведення виставки;
2) адаптація інтерфейсу під різні розміри екранів Android-пристроїв;
3) зберігання даних у локальній реляційній базі SQLite для забезпечення безпеки та цілісності інформації

5. ВИМОГИ ДО ПРОЄКТНОЇ ДОКУМЕНТАЦІЇ

У процесі виконання проєкту повинна бути розроблена наступна документація:

1) пояснювальна записка;
2) програма та методика тестування;
3) керівництво користувача;
4) креслення:
   – «Діаграма діяльності основного сценарію використання»;
   – «Логічна модель бази даних»;
   – «Діаграма компонентів мобільного застосунку».

6. ЕТАПИ ПРОЄКТУВАННЯ

Вивчення літератури за тематикою проєкту.....12.11.2024
Розроблення та узгодження технічного завдання 22.11.2024
Розроблення структури мобільного застосунку 11.12.2024
Підготовка першого розділу дипломного проєкту.....20.12.2024
Розроблення дизайну сторінок та графічних елементів.....21.02.2025
Підготовка другого розділу дипломного проєкту 13.03.2025
Програмна реалізація мобільного застосунку 25.03.2024

Тестування мобільного застосунку 18.04.2024

Підготовка третього розділу дипломного проєкту.....24.04.2024

Підготовка четвертого розділу дипломного проєкту 03.05.2024

Підготовка матеріалів графічної частини проєкту 12.05.2024

Оформлення технічної документації проєкту 23.05.2024

7. ПОРЯДОК ТЕСТУВАННЯ РОЗРОБКИ

Тестування розробленого програмного застосунку виконується відповідно до «Програми та методики тестування».

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

"ЗАТВЕРДЖЕНО"

Завідувач кафедри

_____ Євгенія СУЛЕМА

"___" _____ 2025 р.

МОБІЛЬНИЙ ЗАСТОСУНОК «EXHIBITION GUIDE» ДЛЯ ПРОПОЗИЦІЇ ТА ПОШУКУ ТОВАРІВ НА ВИСТАВЦІ

Пояснювальна записка

ДП.045430-03-81

«ПОГОДЖЕНО»

Керівник проєкту:

_____ Леся ЛЮШЕНКО

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Олександр СТЕПАНЕНКО

2025

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

ПЗ – програмний застосунок.

UI – інтерфейс користувача.

MVC (Model-View-Controller) – архітектурний шаблон, що розділяє програму на три основні компоненти: модель даних, інтерфейс користувача та контролер, який обробляє взаємодію між ними.

MVP (Model-View-Presenter) – модифікація MVC, у якій логіка управління винесена в окремий компонент – презентер, що взаємодіє і з моделлю, і з уявленням (view).

MVVM (Model-View-ViewModel) – архітектурний патерн, який передбачає розділення інтерфейсу користувача (View), моделі (Model) та логіки представлення (ViewModel); сприяє кращій тестованості та підтримуваності коду, часто використовується в Android-розробці.

DAO (Data Access Object) – об'єкт доступу до даних; інтерфейс або клас, що визначає методи взаємодії з базою даних.

UX (User Experience) – досвід користувача.

Фреймворк – це набір заздалегідь розроблених структур, бібліотек, інструментів та шаблонів, які надають базову архітектурну основу для розробки програмного забезпечення.

ACID (Atomicity, Consistency, Isolation, Durability) – набір властивостей транзакцій у системах керування базами даних, що забезпечують надійність та узгодженість даних.

Bottom Sheet – елемент інтерфейсу мобільного застосунку, який висувається з нижньої частини екрана. Використовується для відображення додаткової інформації або дій без переходу на іншу сторінку.

CodeMR – інструмент для статичного аналізу коду, який використовується для оцінки метрик складності, зв’язаності, наслідування та інших параметрів якості програмного забезпечення в середовищі розробки.

Наскрізне тестування (E2E або End-to-End Testing) – це метод тестування програмного забезпечення, коли весь процес роботи програми перевіряється від початку до кінця.

ВСТУП

У добу активного розвитку цифрових технологій виникають нові можливості для реалізації інноваційних підходів у різноманітних сферах. Однією з таких сфер є виставкова діяльність, яка охоплює складні процеси організації масштабних заходів, координації учасників, представлення експонатів і забезпечення ефективної взаємодії з відвідувачами в умовах стрімкого зростання інформаційного потоку.

Виставкова індустрія є важливою складовою сучасної економіки, яка динамічно розвивається та постійно адаптується до змін ринкового середовища. Завдяки високому попиту на ділові події, презентації інновацій і міжнародні контакти, сфера виставкової діяльності зберігає свою конкурентоспроможність. У 2023 році обсяг світового ринку виставкових заходів досягнув приблизно 39,4 млрд дол. США, а середньорічне зростання галузі прогнозується на рівні понад 7% у період до 2032 року. З кожним роком зростають очікування відвідувачів щодо якості обслуговування, швидкості доступу до інформації та індивідуалізації досвіду.

Застосування цифрових технологій, зокрема мобільних застосунків, стає важливою умовою підвищення ефективності організації виставкових заходів та покращення взаємодії з відвідувачами. У цьому контексті створення мобільного застосунку «Exhibition Guide» є актуальним кроком до цифрової трансформації виставкової індустрії.

Метою дипломного проєкту є створення мобільного застосунку «Exhibition Guide», який надасть відвідувачам виставок ефективну навігацію експозиціями, швидкий доступ до інформації про експонати та зручне керування списком обраного. Запропоноване рішення спрямоване на покращення досвіду відвідувачів, підвищення рівня інформованості та персоналізацію взаємодії з експозицією. Завдяки інтуїтивному інтерфейсу та оптимізованій структурі, застосунок сприятиме цифровізації виставкових заходів і відповідатиме сучасним очікуванням користувачів.

1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ПОСТАВЛЕНОЇ ЗАДАЧІ

1.1. Загальні положення та аналіз предметної області

Інформаційні технології відіграють важливу роль у виставковій діяльності, яка є однією з найбільш динамічно розвинутих комерційних видів діяльності. В останні роки організація та управління виставковими заходами зазнали значного розвитку завдяки впровадженню програмних інноваційних рішень.

За даними досліджень у сфері організації різноманітних заходів, цифрові платформи для управління виставками стають стандартом сьогодення. Вони дозволяють автоматизувати процеси реєстрації учасників, управління розміщенням стендів, організації зустрічей і збору аналітичних даних. Згідно з останніми тенденціями, значна частина організаторів і експонентів використовує цифрові інструменти для підвищення зручності взаємодії та покращення вражень відвідувачів [1].

Одним із ключових напрямків виставкової індустрії є продуктивна взаємодія між експонентами та відвідувачами за допомогою сучасних інформаційних технологій. Впровадження мобільних застосунків для управління виставковими заходами та навігації по ним дозволяє не лише оперативно отримувати інформацію про стенди, а також створює інтерактивний програмний майданчик для презентації товарів і послуг експонентів. Використання таких програмних рішень сприяє збільшенню рівня залучення аудиторії та підвищенню якості обслуговування, а також допомагає експонентам більш ефективно презентувати свої інновації.

Застосування мобільних технологій у виставковій діяльності відкриває нові можливості з точки зору управління великим обсягом інформації щодо експонатів, стендів та супутніх заходів. На виставках зберігається безліч даних щодо розташування стендів, характеристик товарів, що створює виклики для традиційних систем управління. Мобільний застосунок «Exhibition Guide» дозволяє автоматизувати обробку

цієї інформації – від збору даних з різних джерел до їх презентації за допомогою інтуїтивного інтерфейсу та інтерактивної мап. Такий підхід мінімізує час пошуку необхідних даних і скорочує витрати ресурсів, забезпечуючи високу продуктивність як для експонентів, так і для відвідувачів, що є критично важливим у сучасних умовах динамічного ринку виставкових послуг.

1.2. Аналіз існуючих рішень

Розглянемо найбільш відомі застосунки для технічних виставок, такі як CES App, Whova та Hannover Messe App , які використовуються для орієнтації на виставках та збереження інформації про експонати.

1.2.1. CES App

CES App – офіційний мобільний застосунок виставки Consumer Electronics Show, призначений для зручної навігації та планування відвідувачами програми заходу. Він надає інтерактивну карту павільйонів з можливістю збільшення/зменшення масштабу та перегляду стендів, каталог експонентів із фільтрацією за назвами компаній, перелік презентацій і семінарів із можливістю додавання їх у особистий план, а також останні новини та оновлення програми [2].

Крім того, CES App доступний для використання як на комп'ютерах, так і на мобільних пристроях через застосунки для iOS та Android (рис. 1.1).

Переваги CES App :

1. Повноцінна інтерактивна карта з маркерами стендів і інформацією про локацію.
2. Актуальний розклад і можливість зберегти обрані події до особистого плану.
3. Каталог експонентів із пошуком та швидким переходом до їхніх профілів.

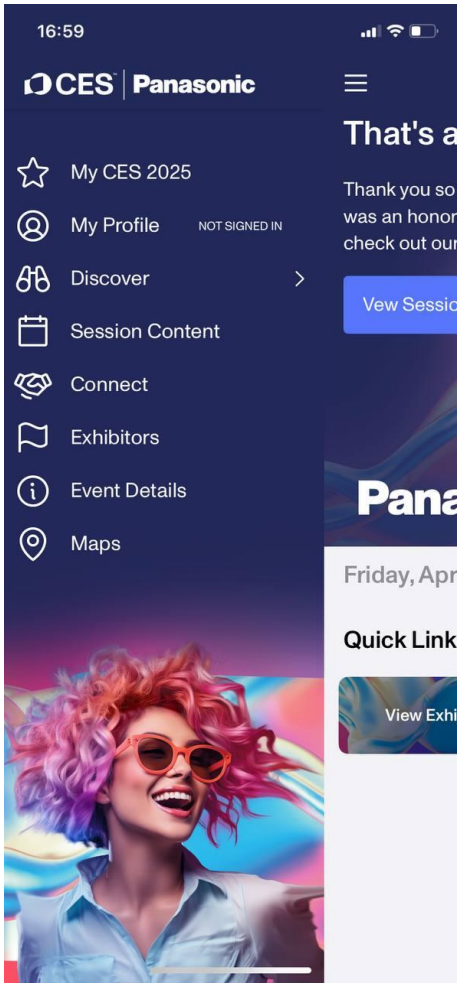


Рис. 1.1. Інтерфейс застосунку CES App

Недоліки CES App:

1. Відсутність можливості випадкового вибору стенду для швидкого відкриття нових експонатів.
2. Немає категорій експонатів які згруповані за темами.
3. Немає власного списку «Улюблене» з можливістю зберігати та видаляти вибрані стенди.
4. Потребує постійного інтернет-з’єднання для завантаження карти.

1.2.2. Whova

Whova – універсальний мобільний застосунок для конференцій і виставок, що використовується сотнями заходів у всьому світі. Застосунок озволяє переглядати розклад сесій, створювати власний персональний план,

знайомитися з профілями доповідачів і учасників, обмінюватися повідомленнями та отримувати оголошення організаторів [3].

Крім того, Whova доступний для використання як на комп'ютерах, так і на мобільних пристроях через застосунки для iOS та Android. Платформа також має вебверсію, яка працює в браузерах, таких як Chrome, Firefox та Safari (рис. 1.2).

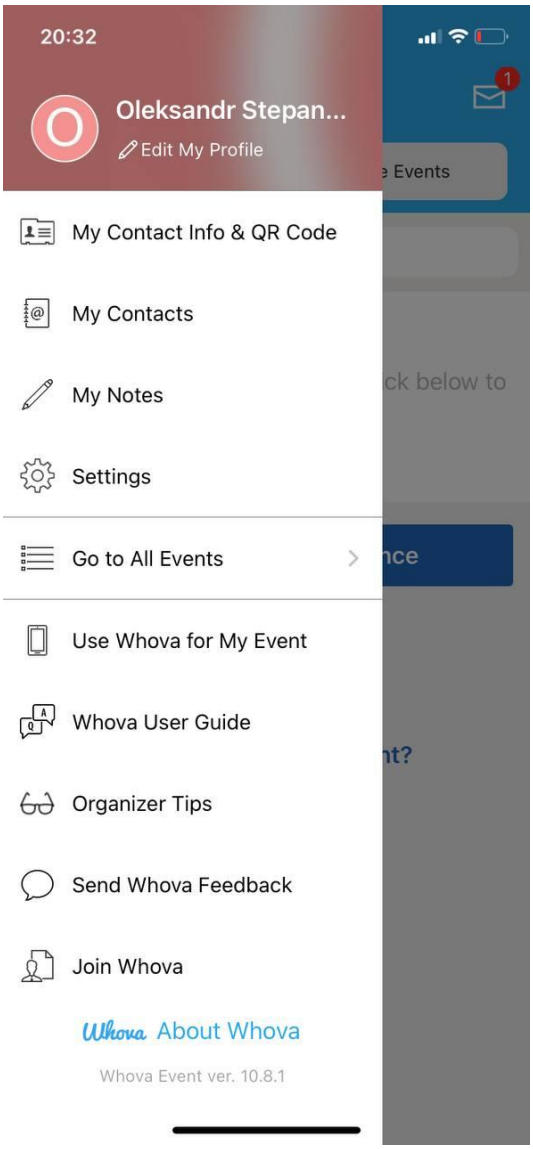


Рис. 1.2. Інтерфейс застосунку Whova

Переваги Whova:

1. Зручний конструктор особистого розкладу зі збереженням обраних сесій.

2. Модулі новин і оголошень від організаторів у режимі push.
3. Каталог спікерів і експонентів із пошуком та фільтрацією за ключовими словами.

Недоліки Whova:

1. Немає інтерактивної карти павільйонів із маркерами стендів.
2. Обмежені можливості групування стендів за категоріями – лише вивід списку учасників.
3. Збереження «Улюблених» обмежується сесіями та профілями користувачів – немає окремого механізму для стендів.

1.2.3. Hannover Messe App

Hannover Messe App – офіційний мобільний застосунок найбільшої у світі промислової виставки Hannover Messe. Застосунок пропонує інтерактивну карту виставкового комплексу з масштабуванням і пошуком стендів за номером павільйону, каталог учасників із детальними профілями компаній, розклад семінарів і демонстрацій з можливістю додавання в персональний план, а також розділ новин і сповіщень про зміни в програмі [4].

Також Hannover Messe App можна використовувати як на комп’ютерах, так і на мобільних пристроях завдяки застосункам для iOS та Android (рис. 1.3).

Переваги Hannover Messe App

1. Інтерактивна карта з точною навігацією по павільйонах та швидким пошуком стендів.
2. Повний каталог експонентів із контактами, описами продукції та посиланнями на сайти.
3. Офлайн-режим для перегляду карти та каталогу без підключення до інтернету.

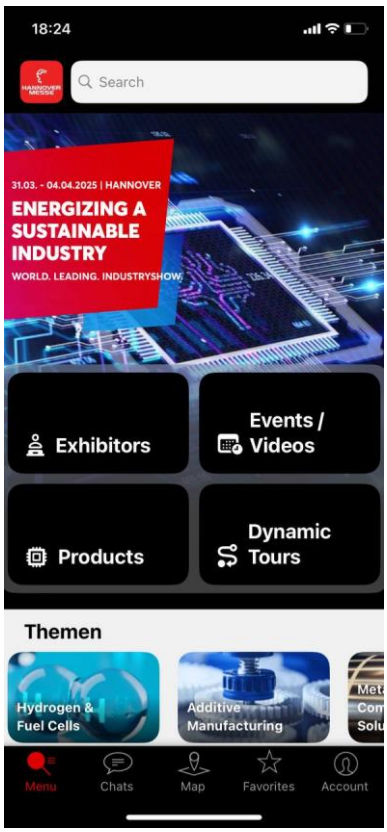


Рис. 1.3. Інтерфейс застосунку Hannover Messe App

Недоліки Hannover Messe App:

1. Відсутня можливість випадкового вибору стенду для демонстрації нових учасників.
2. Немає окремого розділу «Улюблене» для збереження та видалення вподобаних компаній чи стендів.
3. Відсутність категоризації експонатів.

Отже, хоча наявні рішення, такі як CES App, Whova та Hannover Messe App, пропонують користувачам розклад, каталог експонентів і повідомлення від організаторів, жодне з них не поєднує всі ключові можливості одночасно – повноцінну інтерактивну карту зі збільшенням/зменшенням масштабу, розширений пошук стендів за категоріями та зручне керування списком «Улюблене» зі збереженням і видаленням обраних пунктів. Тому для повноцінної підтримки відвідувачів технічних виставок необхідне спеціалізоване рішення – мобільний

застосунок «Exhibition Guide», який врахує специфічні потреби орієнтації, пошуку та персоналізації досвіду на виставках.

1.3. Актуальність розробки застосунку «Exhibition Guide»

Порівнявши можливості існуючих рішень – CES App, Whova та Hannover Messe App – можна побачити, що жоден із них не поєднує в собі всіх необхідних функцій, хоча CES App та Hannover Messe App мають інтерактивні карти, їм бракує гнучкого списку «Улюблене» та категорій експонатів. Whova пропонує потужний персональний розклад і сповіщення, але взагалі не має карти павільйонів. Як наслідок, відвідувачам доводиться використовувати кілька різних застосунків або ж витрачати час на ручний пошук і планування. Проведемо порівняльний аналіз цих застосунків-аналогів у табл. 1.1.

Таблиця 1.1

Порівняння функціоності застосунків

Застосунок	CES App	Whova	Hannover Messe
Інтерактивна карта павільйонів	+	–	+
Масштабування карти	+	–	+
Пошук експонатів за назвою	+	+	+
Категоризація експонатів	–	–	–
Список «Улюблене» з додаванням/видаленням	–	–	–
Рекомендації (найпопулярніші стенди)	–	–	–
Офлайн-режим карти	–	–	+

Виходячи з вищезазначеного, актуальною є розробка застосунку «Exhibition Guide», який буде спеціалізованим інструментом для задоволення потреб виставкової діяльності.

Конкурентні переваги застосунку «Exhibition Guide»:

- повний набір навігаційних інструментів, інтерактивна офлайн-карта павільйонів із масштабуванням;
- категоризація стендів, групування за темами;
- список «Улюблене», можливість додавати й видаляти обрані стенди;
- рекомендаційна система, динамічні підказки найпопулярніших стендів;
- повнотекстовий пошук за назвою.

Мета застосунку «Exhibition Guide» полягає в тому, щоб надати користувачам зручний програмний інструмент управління інформацією, що стосується представлених експонатів на виставці.

Основні функціональні можливості «Exhibition Guide»:

1. Інтерактивні карти павільйонів з підтримкою офлайн-доступу та масштабуванням.
2. Каталог стендів із тематичною категоризацією.
3. Система «Улюблене» для збереження й видалення обраних стендів.
4. Рекомендації найбільш популярних стендів.
5. Пошук експонатів за ключовими словами.

1.4. Загальні вимоги до застосунку

Провівши аналіз програмних аналогів, їхніх переваг та недоліків, а також дослідивши специфіку ринку виставкової діяльності, було визначено перелік функціональних та нефункціональних вимог до ПЗ.

Функціональні вимоги:

- інтерактивна карта павільйонів, підтримка масштабування, офлайн-доступ до картографічних даних;
- каталог стендів із тематичною категоризацією;

- пошук експонатів за ключовими словами;
- система «Улюблене», додавання та видалення обраних стендів, перегляд списку збережених;
- рекомендації найбільш популярних стендів;
- зручний, естетичний та адаптивний інтерфейс для мобільних пристроїв.

Нефункціональні вимоги:

- стабільність і безперебійний доступ до застосунку під час проведення виставок;
- коректна робота на популярній операційній системі – Android;
- захист конфіденційності даних користувачів;

Вимоги до якості розроблюваного ПЗ:

- проведення тестування на різних пристроях із різними версіями операційних систем;
- використання сучасних фреймворків для створення кросплатформенного застосунку;
- впровадження найкращих практик програмування для забезпечення стабільності та продуктивності;
- регулярний моніторинг роботи застосунку та швидке усунення можливих помилок.

1.5. Висновки до розділу

Вивчення сучасного стану виставкової індустрії показало, що інформаційні технології стали невід’ємною частиною організації та проведення великих заходів: вони автоматизують реєстрацію учасників, управління розташуванням стендів, збір аналітики та створюють інтерактивну платформу для презентацій. Мобільні застосунки для навігації та управління експонентами значно підвищують ефективність взаємодії між

організаторами, учасниками та відвідувачами, скорочуючи час на пошук інформації й оптимізуючи їхній досвід.

Дослідження застосунків CES App, Whova та Hannover Messe App виявило, що кожен із них успішно вирішує окремі завдання (інтерактивні карти, персоналізовані розклади чи каталог учасників). Проте жоден не поєднує всіх необхідних функцій разом: масштабованої офлайн-карти, тематичної категоризації стендів, зручного пошуку, обраних списків та рекомендацій. Це свідчить про потребу в створенні спеціалізованого застосунку, який би задовольняв усі ключові вимоги виставкової індустрії.

Виникає потреба в мобільному застосунку, який поєднає інтерактивну офлайн-карту, детальний каталог експонатів із тематичними категоріями, систему збереження улюблених стендів та рекомендаційні підказки.

Аналіз функціональних і нефункціональних вимог визначив ключові напрями розробки: підтримка масштабованої офлайн-карти, тематична категоризація стендів, пошук за ключовими словами, система «Улюблене» та динамічні рекомендації, інтуїтивний адаптивний інтерфейс, висока продуктивність і захист даних. Виконання цих вимог забезпечить стабільну роботу «Exhibition Guide» на різних пристроях і створить комфортний досвід для всіх учасників виставок.

2. ОБҐРУНТУВАННЯ ВИБОРУ ЗАСОБІВ РЕАЛІЗАЦІЇ

2.1. Обґрунтування вибору мови програмування

Для розробки мобільного застосунку можна використовувати кілька мов програмування залежно від специфіки задач розробки.

Kotlin є високорівневою, статично типізованою мовою програмування, яка використовується переважно для створення Android-застосунків і визнана однією з основних мов мобільної розробки у світі [1]. Вона має повну сумісність із Java, підтримується Android-платформою та активно розвивається спільнотою розробників. Kotlin забезпечує безпечне та ефективне програмування, зменшує кількість помилок у коді та підвищує продуктивність розробки. Завдяки офіційній підтримці з боку Google. Ця мова стала основним інструментом для сучасної розробки мобільних застосунків [5].

Java є класичною мовою програмування для розробки Android-застосунків, яка існує вже понад 25 років і широко використовується завдяки високій продуктивності, стабільності та розвиненій екосистемі. Java забезпечує відмінну сумісність із різноманітними платформами та має велику кількість доступних бібліотек і фреймворків, що значно прискорює розробку та забезпечує високу продуктивність та безпеку застосунків [6].

C++ також може бути використаний для розробки Android-застосунків, особливо коли необхідна максимальна продуктивність (у графічних застосунків чи іграх). Однак застосування C++ є складнішим через необхідність ретельного управління пам'яттю та складнішу структуру коду, що підходить далеко не для всіх типів застосунків [7].

В якості мови програмування було обрано Kotlin, оскільки вона є сучасним стандартом для розробки Android-застосунків і повністю підтримується платформою Android.

Ось деякі ключові аргументи, які обґрунтовують вибір мови програмування Kotlin:

1. Null-безпека. Kotlin забезпечує вбудовані механізми для обробки можливих помилок, пов'язаних із null-значеннями. Це значно знижує ризик виникнення винятків типу NullPointerException під час виконання програми.
2. Лаконічність коду. Завдяки більш стислому синтаксису, порівняно з Java, код на Kotlin менш об'ємний і легший для підтримки. Це підвищує продуктивність розробників і прискорює процес розробки.
3. Підтримка корутин. Kotlin має вбудовану підтримку корутин – сучасного механізму асинхронного програмування, що дає змогу ефективно працювати з тривалими або фоновими задачами без блокування головного потоку.
4. Повна сумісність із Java. Kotlin може співіснувати з Java-кодом в одному проєкті. Це дозволяє поступово переносити старі частини проєкту з Java на Kotlin без ризику для стабільності застосунку.
5. Офіційна підтримка Google. Google активно інвестує в розвиток екосистеми Kotlin, надаючи підтримку в Android Studio, Jetpack-бібліотеках та документації, що робить Kotlin безпечним вибором для довготривалих проєктів.
6. Поширеність серед Android-розробників. За даними опитувань Google, понад 60% професійних Android-розробників уже використовують Kotlin у своїх комерційних проєктах [5].

Як бачимо, Kotlin є потужною та зручною мовою програмування для розробки Android-застосунків. Хоча ця мова має багато переваг, варто також звернути увагу на деякі її недоліки, які можуть впливати на процес розробки:

1. Менша кількість прикладів порівняно з Java: Попри зростаючу популярність, Kotlin все ще має меншу кількість прикладів,

навчальних матеріалів і відкритих проєктів у порівнянні з Java. Для новачків це може ускладнити пошук рішень нестандартних задач.

2. Складніша інтеграція в старі проєкти: Незважаючи на сумісність із Java, інтеграція Kotlin у великі монолітні проєкти, які повністю написані на Java, може вимагати ретельного тестування і поступового переходу, щоб уникнути конфліктів.
3. Швидкість компіляції: у деяких випадках, особливо при повній компіляції великих проєктів, Kotlin може компілюватися повільніше, ніж Java. Хоча інкрементальна компіляція (часткова) працює швидко, повна побудова проєкту може займати більше часу.
4. Підвищені вимоги до досвіду розробника: завдяки своїй гнучкості та розширеним можливостям, Kotlin може мати складніші конструкції, які потребують глибшого розуміння. Без належного досвіду це може призводити до неправильного використання можливостей мови.

Отже, незважаючи на деякі недоліки такі, як більший час компіляції, складність для новачків і менша кількість навчальних ресурсів, Kotlin залишається оптимальним вибором для розробки цього Android-застосунку завдяки своїй сучасності, офіційній підтримці Google та широким можливостям для створення стабільних і продуктивних мобільних рішень.

2.2. Обґрунтування вибору бази даних

Вибір відповідної бази даних відіграє важливу роль у реалізації Android-застосунку. Залежно від особливостей зберігання даних, вимог до продуктивності та доступу, розробникам доводиться обирати між реляційними та нереляційними системами управління базами даних. Розглянемо популярні системи керування базами даних кожного типу.

SQLite – це вбудована реляційна система управління базами даних, яка є частиною платформи Android за замовчуванням. Вона не потребує

окремого серверного оточення й дозволяє зберігати дані безпосередньо на пристрої користувача. SQLite підтримує стандарт SQL, транзакції за принципом ACID, а також дозволяє реалізовувати складні запити з фільтрацією, сортуванням і агрегацією. Така база даних ідеально підходить для локального зберігання структурованих даних у мобільних застосунках, особливо коли важлива автономна робота застосунку без постійного доступу до Інтернету [8].

Realm – ще одне популярне рішення для локального зберігання даних у мобільних застосунках, яке належить до категорії нереляційних баз даних. Realm не використовує традиційні таблиці та SQL-запити, а працює безпосередньо з об'єктами, що робить його привабливим для розробників, які прагнуть зменшити обсяг шаблонного коду та пришвидшити доступ до даних. Завдяки високій продуктивності, автоматичній синхронізації змін і підтримці реактивного програмування Realm дозволяє створювати сучасні та чутливі інтерфейси. Проте, попри зручний API та просту інтеграцію, Realm не є частиною стандартного Android SDK і має власний формат збереження даних. Це може створювати певні обмеження, зокрема в ситуаціях, коли потрібна гнучкість роботи із SQL-запитами або повна прозорість структури бази [9].

Для реалізації локального зберігання даних у мобільному застосунку було обрано SQLite – вбудовану реляційну базу даних, яка входить до складу Android SDK. Цей варіант є оптимальним для мобільних платформ завдяки своїй легкості, стабільності та підтримці стандартного SQL-синтаксису. SQLite надає низку переваг:

1. Вбудованість і автономність: база не потребує налаштування серверного середовища або підключення до мережі, працюючи напряму у файловій системі пристрою.
2. Підтримка транзакцій: дотримання принципів ACID гарантує надійність зберігання даних і запобігає їх пошкодженню.

3. Гнучкість структурування: розробник має повний контроль над створенням і зміною таблиць, індексів та SQL-запитів.
4. Мала вага: SQLite не створює додаткових навантажень на застосунок, що особливо важливо для легких і швидких мобільних програм.

Підсумовуючи, використання SQLite є найбільш доцільним рішенням для даного Android-застосунку. Це забезпечує надійне, зручне у використанні та гнучке локальне зберігання даних, що повністю відповідає вимогам «Exhibition Guide».

2.3. Обґрунтування вибору фреймворку для клієнтської частини

Фреймворк визначає структуру та архітектуру програмного забезпечення, спрощуючи процес розробки, забезпечуючи швидке розгортання та підтримку. Оскільки проєкт реалізується мовою програмування Kotlin і використовує локальну базу даних SQLite, вибір фреймворку зосереджено навколо платформи Android Jetpack – сучасного набору фреймворків, бібліотек та інструментів, офіційно підтримуваних Google [10].

Для розробки клієнтської частини Android-застосунку було обрано саме компоненти Android Jetpack. Вони дозволяють будувати ефективні, безпечні та масштабовані інтерфейси відповідно до рекомендацій Google.

Одним із ключових елементів став Jetpack Navigation Component – інструмент для декларативного управління переходами між екранами. Він дозволяє створювати централізований граф навігації, безпечно передавати параметри між екранами і підтримує популярні шаблони навігації такі, як Bottom Navigation, Drawer, Deep Links тощо [11].

Для побудови адаптивного UI використовується ConstraintLayout – потужний макетний контейнер, що дозволяє створювати складні інтерфейси з мінімальною глибиною ієрархії. Це забезпечує покращену продуктивність і спрощує адаптацію до різних екранів [12].

Виведення динамічного контенту реалізовано за допомогою RecyclerView, який ефективно працює з великими обсягами даних та підтримує повторне використання елементів [13]. Для оформлення списків використовується CardView, що забезпечує візуально привабливий стиль карток у дусі Material Design [14].

Окрім цього, для реалізації інтерактивної роботи з зображеннями інтегровано сторонню бібліотеку PhotoView. Вона забезпечує зручне масштабування та панорамування зображень за допомогою жестів, що покращує взаємодію користувача з контентом [15].

Таким чином, поєднання компонентів Android Jetpack та перевірених бібліотек дозволяє створити сучасний інтерфейс для Android-застосунку.

2.4. Обґрунтування вибору архітектури застосунку

При розробці застосунків на платформі Android критично важливим є вибір архітектури. Існують декілька поширених архітектурних підходів, зокрема MVC [16], MVP [17] та MVVM [18]. Їх коротке порівняння за основними критеріями наведено в табл. 2.1.

Таблиця 2.1

Порівняння архітектурних підходів MVC, MVP та MVVM

Аспекти	MVC	MVP	MVVM
Розподіл відповідальностей	UI та логіка даних тісно сплетені; View напряму звертається до Model.	Введено компонент Presenter, що опосередковує взаємодію між View і Model, розділяючи логіку.	ViewModel слугує посередником між View і Model, забезпечуючи слабке зв’язування через механізми спостереження.

Продовження табл. 2.1

Відношення між компонентами	Один Controller може керувати кількома View (1→N).	Один Presenter відповідає одній View (1→1).	Один ViewModel може обслуговувати кілька View (1→N), що дозволяє повторно використовувати логіку.
Залежність від Android	Висока. Controller зазвичай представлений Activity/Fragment і залежить від фреймворку Android.	Низька. Presenter – це звичайний клас Java/Kotlin, мінімально залежний від Android.	Мінімальна. ViewModel не оперує Android-класами UI (покладається на бібліотеки Jetpack), тому бізнес-логіка ізольована від платформи.
Можливість модифікацій	Низька: через сильну зв’язаність важко вносити зміни або розширювати можливостей без ризику порушити роботу.	Вища: чітке розділення View і Presenter спрощує зміну логіки або інтерфейсу незалежно один від одного.	Найвища: компоненти максимально ізольовані, що спрощує внесення змін; але дуже складна логіка прив’язки даних може ускладнити відлагодження.
Масштаб проєкту	Підходить для невеликих застосунків зі спрощеною логікою.	Застосовний як у простих, так і у достатньо складних проєктах.	Дає найбільші переваги в застосунках середньої та великої складності; для тривіальних застосунків може виглядати надмірним.

З огляду на наведені аргументи, архітектура MVVM на базі Android Jetpack компонентів є найбільш доцільною для застосунку «Exhibition Guide». Вона забезпечує гнучке масштабування можливостей застосунку, простоту внесення змін та високу надійність роботи інтерфейсу. Завдяки MVVM застосунокок матиме чисту та підтримувану структуру, що полегшує подальший розвиток проєкту і гарантує позитивний досвід користувачів.

2.5. Висновки до розділу

У даному розділі було розглянуто ключові технології, що використовуються для розробки мобільного Android-застосунку. Вибір засобів реалізації є надзвичайно важливим для досягнення стабільності, продуктивності та масштабованості програмного продукту. Кожне з рішень – від мови програмування до вибору фреймворку – має прямий вплив на зручність розробки, підтримки та експлуатації застосунку. Технології обиралися з урахуванням рекомендацій Google, поширених практик та вимог до застосунку «Exhibition Guide».

У ролі мови програмування було обрано Kotlin – сучасну, ефективну та офіційно підтримувану Google мову для розробки Android-застосунків. Вона забезпечує зручний синтаксис, сумісність із Java, підтримку корутин і знижує ризик помилок, пов’язаних із null-значеннями. Завдяки активному розвитку екосистеми Kotlin і поширеності серед Android-розробників, ця мова є оптимальним вибором для створення стабільних і сучасних мобільних застосунків.

Для локального зберігання даних було обрано базу даних SQLite. Це рішення забезпечує безпечне, автономне та ефективне управління даними без потреби у зовнішньому сервері. Завдяки підтримці транзакцій ACID, SQL-запитів та широкій сумісності з Android-платформою, SQLite дозволяє ефективно реалізувати зберігання списку експонатів, категорій і вподобань безпосередньо на пристрої користувача.

Клієнтська частина застосунку реалізована з використанням компонентів платформи Android Jetpack. Зокрема, Jetpack Navigation Component забезпечує гнучке управління навігацією, RecyclerView і CardView відповідають за продуктивне виведення списків, а ConstraintLayout – за побудову адаптивного UI. Бібліотека PhotoView використовується для реалізації масштабування зображень. Обрані фреймворки та бібліотеки забезпечують ефективну реалізацію сучасного, зручного та масштабованого користувацького інтерфейсу, що відповідає рекомендаціям Google щодо UX/UI для Android.

У ролі архітектурного підходу було обрано MVVM – сучасний патерн, що оптимально підходить для мобільного середовища Android. Він забезпечує чітке розділення відповідальностей між UI, логікою представлення та джерелом даних, сприяючи побудові масштабованого та підтримуваного застосунку. Завдяки мінімальній залежності від фреймворку Android, підтримці бібліотек Jetpack та активному застосуванню у професійній розробці, MVVM є доцільним вибором для реалізації стабільної та гнучкої архітектури застосунку «Exhibition Guide».

3. РОЗРОБЛЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

3.1. Загальний опис структури застосунку

Перед описом загальної структури застосунку «Exhibition Guide» необхідно взяти до уваги визначені раніше основні функціональні можливості, які включають:

1. Інтерактивні карти павільйонів з підтримкою офлайн-доступу та масштабуванням.
2. Каталог стендів із тематичною категоризацією.
3. Система «Улюблене» для збереження й видалення обраних стендів.
4. Рекомендації найбільш популярних стендів.
5. Пошук експонатів за ключовими словами.

Застосунок матиме декілька ключових сторінок, які забезпечать користувачам зручний доступ до основних функцій:

1. Головна сторінка, що містить випадковий експонат, список популярних стендів та швидкий доступ до пошуку та категорій.
2. Сторінка категорій, що дозволяє обрати одну з тематичних категорій для перегляду відповідних стендів.
3. Сторінка експоната, яка відображає повну інформацію про експонат, включаючи фото, описом і картою розташування.
4. Сторінка «Улюблене», де користувач може переглядати всі додані в обране експонати та керувати ними.
5. Сторінка пошуку, що дає змогу швидко знаходити стенди за введеним запитом.

Основний сценарій мобільного застосунку описує процес перегляду інформації про випадковий експонат (рис. 3.1).

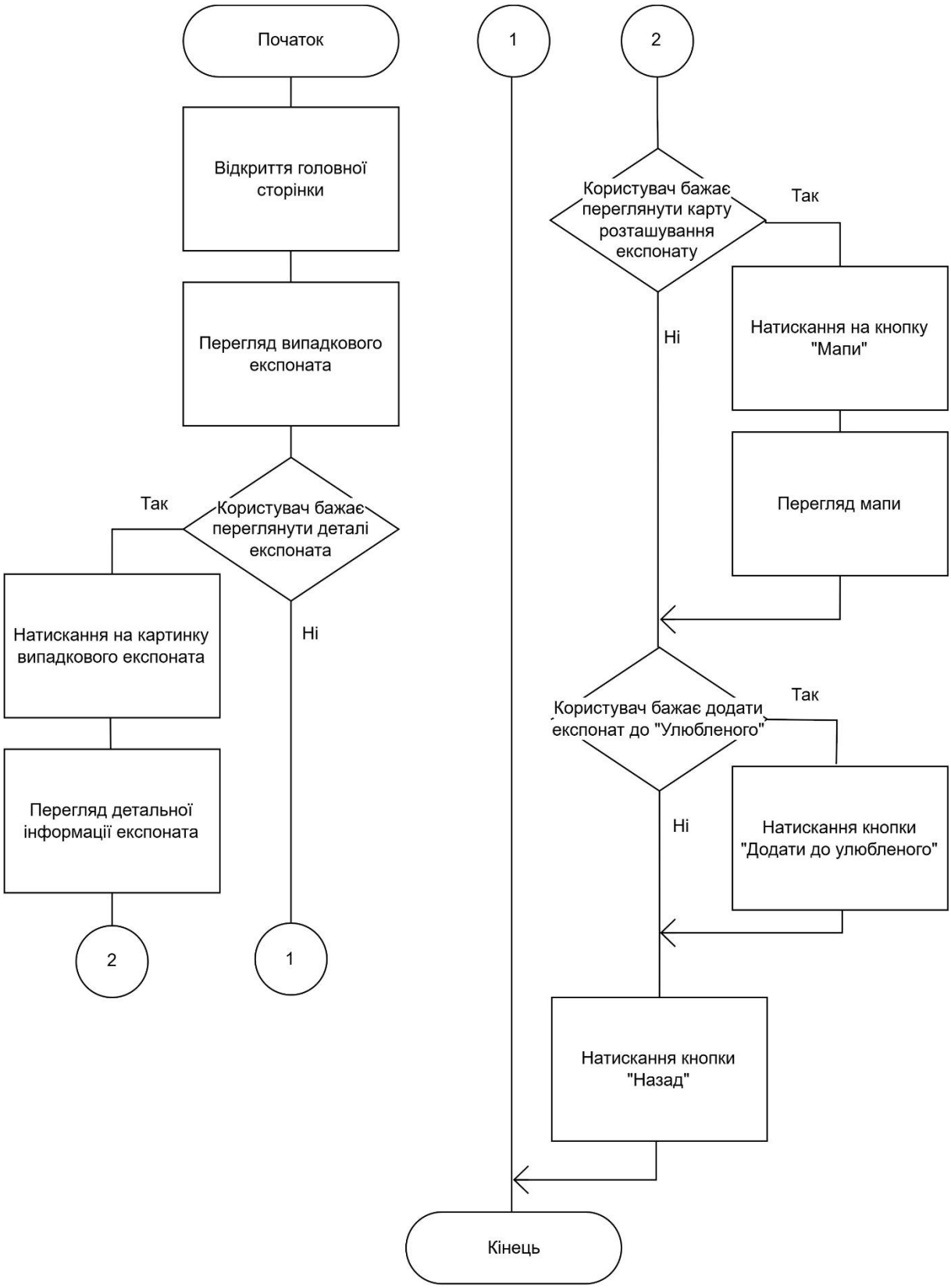


Рис. 3.1. Діаграма діяльності основного сценарію використання

Користувач відкриває головну сторінку застосунку, де відразу бачить випадковий експонат. Після цього користувач може переглядати основну інформацію, що відображена на головному екрані.

Якщо користувач бажає отримати більше деталей про експонат, він натискає на картинку експоната та переходить на окрему сторінку детального перегляду, де представлена повна інформація.

На сторінці експоната користувач має можливість додатково переглянути карту розташування стенду, натиснувши на відповідну кнопку «Мапа». Після перегляду карти користувач повертається назад до деталей експоната.

Також у користувача є можливість додати експонат до списку «Улюблене», натиснувши на відповідну кнопку «Додати до улюбленого».

Після завершення усіх дій користувач натискає кнопку «Назад» і повертається до головної сторінки застосунку.

Користувачі застосунку можуть виконувати такі дії (рис. 3.2):

- перегляд випадкового експоната на головній сторінці;
- перегляд горизонтального списку популярних експонатів;
- виклик інформаційного Bottom Sheet для швидкого перегляду популярного експоната;
- перегляд категорій експонатів;
- перегляд списку експонатів у вибраній категорії;
- перегляд детальної інформації про експонат;
- перегляд мапи розташування експоната на виставці;
- додавання експоната до списку «Улюблені»;
- видалення експоната з «Улюблених» із можливістю відміни дії;
- перегляд списку збережених улюблених експонатів;
- пошук експонатів за ключовими словами.

Рис. 3.2. Дії користувача мобільного застосунку «Exhibition Guide»

3.2. Архітектура мобільного застосунку «Exhibition Guide»

У цьому проєкті було застосовано архітектурний патерн MVVM [19] (Model-View-ViewModel), що був обґрунтований у пункті 2.4. Цей патерн передбачає розподіл відповідальності між трьома основними компонентами: Model, View і ViewModel. Такий підхід сприяє підвищенню організованості коду, полегшує його підтримку й тестування. Загальна структура архітектури MVVM у застосунку подана на рис. 3.3.

Model: цей рівень відповідає за абстрагування джерел даних. Model взаємодіє з ViewModel для отримання, обробки та збереження даних.

View: завдання цього рівня – передавати інформацію про дії користувача до ViewModel. View спостерігає за ViewModel і не містить жодної бізнес-логіки застосунку.

ViewModel: забезпечує доступ до потоків даних, які є важливими для View. Крім того, ViewModel виступає посередником між Model і View, організовуючи їх взаємодію.

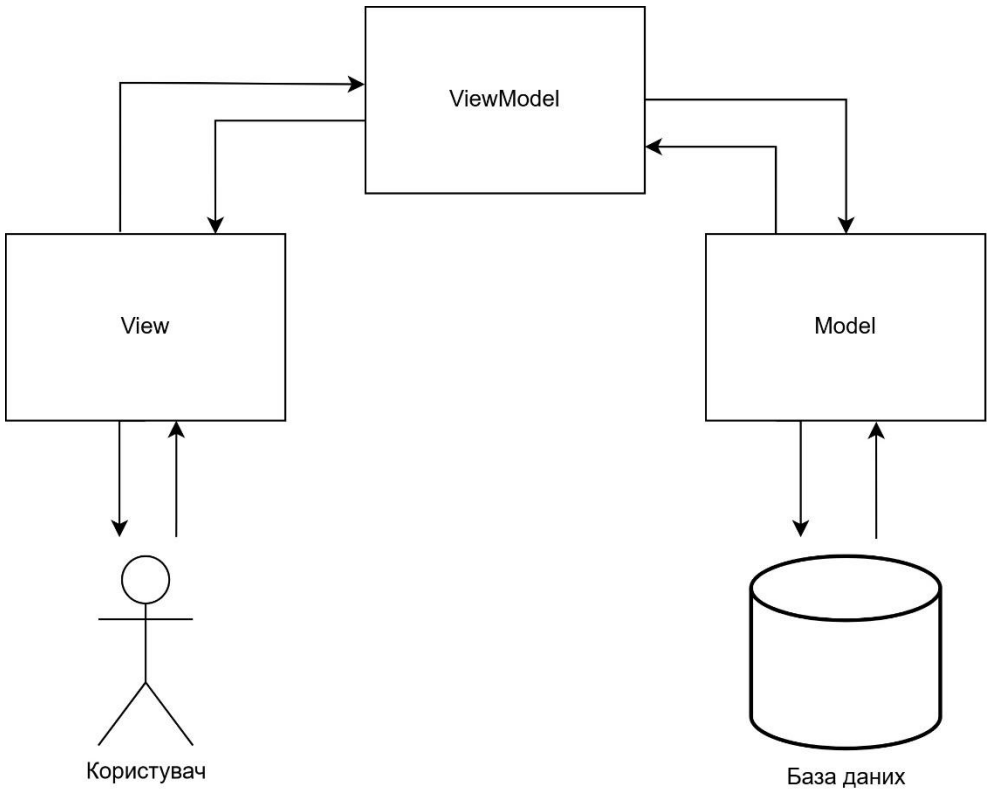


Рис. 3.3. Архітектура мобільного застосунку

На діаграмі пакетів, наведеній на рис. 3.4, чітко окреслено основні підсистеми мобільного застосунку «Exhubition Guide» та їх взаємозв’язки. У центрі уваги знаходиться пакет com.example.exhubition_guide, який розгортається на три ключові зони відповідальності: «data», «viewmodels» та «ui». Кожна з них відповідає за своє завдання в рамках архітектури MVVM – від зберігання і обробки даних до підготовки їх для відображення та безпосередньо візуалізації інтерфейсу користувача.

Пакет ui включає всі компоненти, що безпосередньо взаємодіють із користувачем. Всередині неї розташовані підпакети activities, fragments і adapters. У пакеті activities містяться повноцінні екрани, які координують навігацію між фрагментами та обробку системних подій. Пакет fragments об’єднує невеликі, спеціалізовані UI-компоненти, кожен із яких відображає окрему ділянку інтерфейсу. У пакеті adapters знаходяться класи, які пришивають дані до списків і сіток – вони відповідають за заповнення RecyclerView із різними макетами елементів.

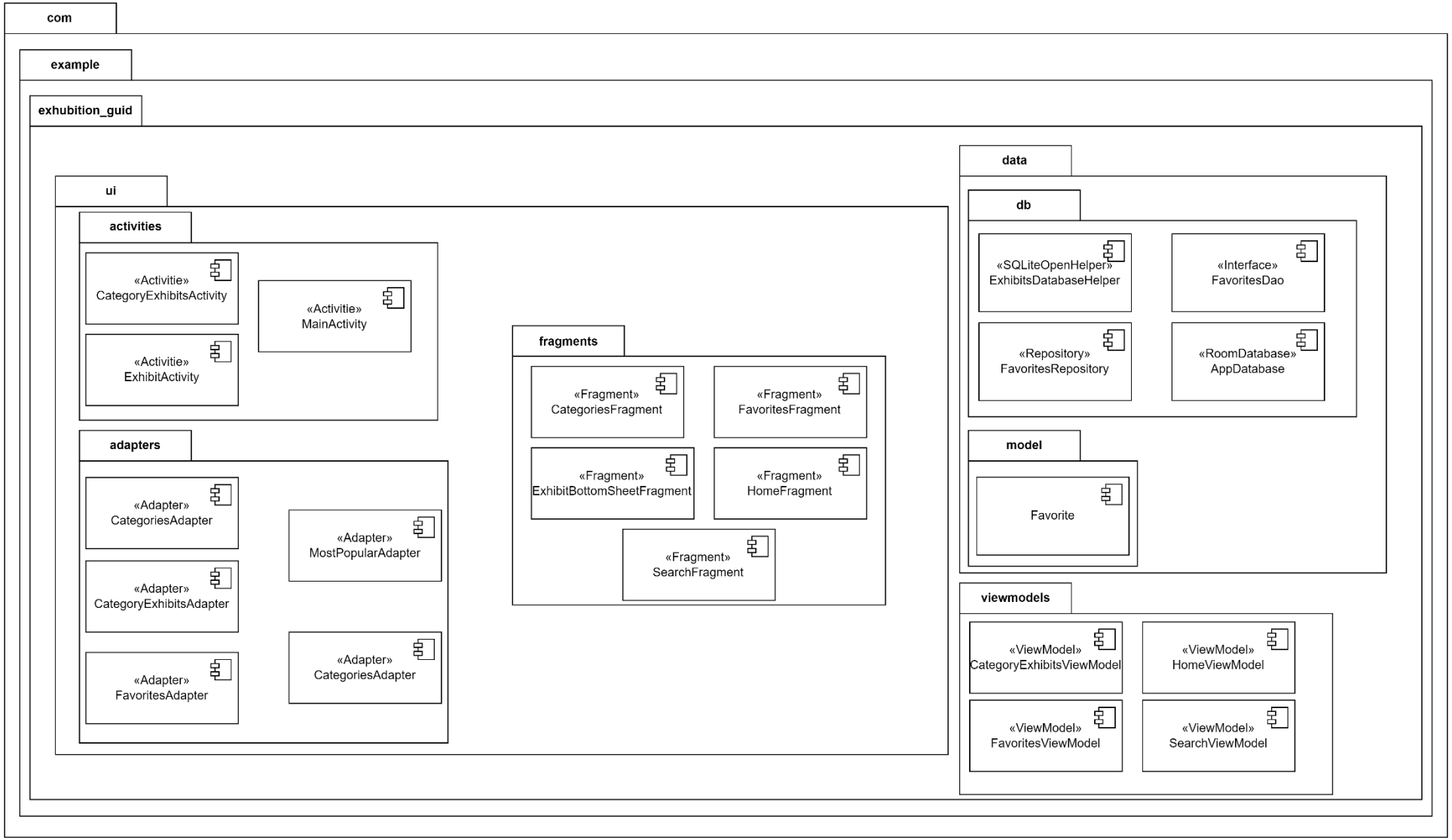


Рис. 3.4. Діаграма компонентів мобільного застосунку

В пакеті viewmodels зосереджена логіка представлення – класи ViewModel інкапсулюють усі взаємодії між UI і джерелом даних, зберігають стан екрану та забезпечують реактивне оновлення UI через LiveData.

Пакет data об’єднує в собі компоненти, які працюють із локальною базою даних. У пакеті db міститься допоміжні класи для SQLite, конфігурацію Room, DAO-інтерфейси, а також репозиторій, що надає уніфікований доступ до CRUD-операцій. У пакеті model міститься сутність доменної моделі – Favorite, яка пов’язує дані на рівні сховища та ViewModel.

3.3. База даних мобільного застосунку «Exhibition Guide»

У мобільному застосунку «Exhibition Guide» для зберігання локальних даних використовується реляційна база даних, реалізована на основі системи управління базами даних SQLite. Такий підхід забезпечує збереження структурованої інформації у вигляді взаємопов’язаних таблиць, що відповідає архітектурі MVVM та дозволяє ефективно взаємодіяти з даними через ViewModel. Реляційна модель підтримує логічні зв’язки між ключовими сутностями застосунку: експонатами, категоріями, виставками, учасниками та списком обраного. Завдяки використанню SQLite забезпечується автономність роботи застосунку без потреби в зовнішньому підключенні до інтернету. Збережені дані залишаються доступними навіть після перезапуску програми, що підвищує зручність для користувача.

База даних включає п’ять таблиць (рис. 3.5): exhibits, categories, exhibitions, exhibitors та favorites. Кожна з них відповідає за зберігання певного типу інформації. Зокрема, таблиця exhibits містить основні дані про експонати, таблиця categories зберігає тематику експонатів, exhibitions – інформацію про виставки, exhibitors – про учасників, а таблиця favorites фіксує експонати, що були позначені користувачем як улюблені. Опис таблиць бази даних наведено у табл. 3.1 – 3.5.

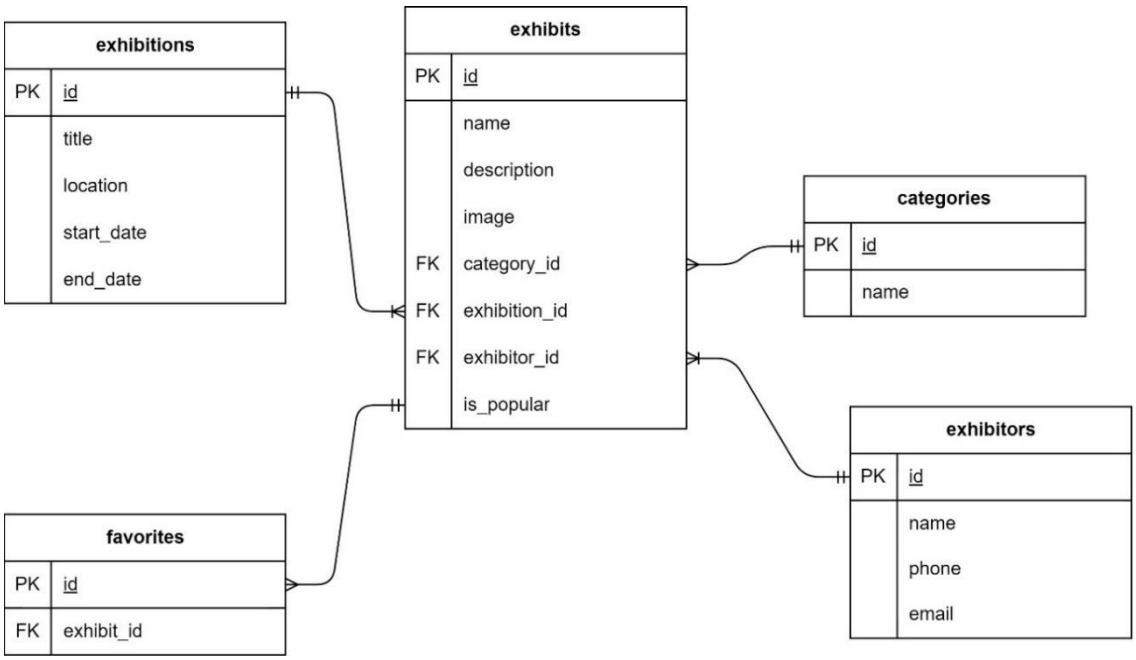


Рис. 3.5. Логічна модель бази даних

Таблиця 3.1

Опис таблиці exhibits

Назва поля	*Опис*	*Тип даних*
id	Ідентифікатор експоната	INTEGER
name	Назва експоната	TEXT
description	Опис експоната	TEXT
image	Шлях до зображення	TEXT
category_id	Ідентифікатор категорії	INTEGER
exhibition_id	Ідентифікатор виставки	INTEGER
exhibitor_id	Ідентифікатор учасника	INTEGER
is_popular	Ознака популярності	INTEGER

Таблиця 3.2

Опис таблиці favorite

Назва поля	*Опис*	*Тип даних*
id	Ідентифікатор обраного запису	INTEGER

Продовження табл. 3.2

exhibit_id	Ідентифікатор експоната	INTEGER

Таблиця 3.3

Опис таблиці categories

Назва поля	*Опис*	*Тип даних*
id	Ідентифікатор категорії	INTEGER
name	Назва категорії	TEXT

Таблиця 3.4

Опис таблиці exhibitions

Назва поля	*Опис*	*Тип даних*
id	Ідентифікатор виставки	INTEGER
title	Назва виставки	TEXT
location	Місце проведення	TEXT
start_date	Дата початку виставки	DATE
end_date	Дата завершення виставки	DATE

Таблиця 3.5

Опис таблиці exhibitors

Назва поля	*Опис*	*Тип даних*
id	Ідентифікатор учасника	INTEGER
name	Назва компанії або учасника	TEXT
phone	Контактний номер	TEXT
email	Електронна пошта	TEXT

3.4. Аналіз безпеки даних

У застосунку «Exhibition Guide» не зберігаються конфіденційні чи персональні дані користувача, тому питання безпеки не є критично важливим. Проте слід зазначити, що всі дані зберігаються локально в межах системної «пісочниці» Android. Це означає, що доступ до файлів та ресурсів застосунку має лише сам застосунок, і вони недоступні для інших програм без дозволу користувача. Така ізоляція забезпечує базовий рівень безпеки та захищає дані від несанкціонованого доступу.

3.5. Висновки до розділу

Мобільний застосунок «Exhibition Guide» надає користувачу зручний і логічно структурований доступ до ключових функцій, таких як перегляд випадкового експоната, список популярних стендів, категоризація, пошук та можливість керування списком улюбленого. Основний сценарій взаємодії з застосунком описує типовий маршрут користувача, що включає відкриття головної сторінки, перегляд інформації про експонат, роботу з картою та керування улюбленими. Це дозволило сформувати повну картину логіки та інтерфейсної структури застосунку.

Архітектура мобільного застосунку «Exhibition Guide» побудована на основі патерна MVVM, що забезпечує чіткий розподіл відповідальності між представленням, логікою керування даними та джерелом інформації. Така структура дозволяє підтримувати високу організованість коду, сприяє його масштабованості та полегшує тестування. Діаграма компонентів підтверджує логічну організацію внутрішньої структури проєкту відповідно до принципів MVVM, де окремі зони відповідають за інтерфейс користувача, бізнес-логіку та роботу з локальною базою даних.

База даних мобільного застосунку «Exhibition Guide» реалізована у вигляді реляційної структури, яка ефективно моделює взаємозв’язки між ключовими сутностями проєкту. За допомогою п’яти взаємопов’язаних таблиць забезпечується логічна цілісність даних, що дозволяє зберігати та

обробляти інформацію про експонати, категорії, виставки, учасників та обране. Обраний підхід на базі SQLite сприяє підтримуваності коду, простому масштабуванню структури даних, а також інтеграції з архітектурою MVVM, де ViewModel взаємодіє з даними через рівень доступу до бази.

Щодо безпеки даних, застосунок не оперує конфіденційною інформацією користувача. Усі дані зберігаються локально в межах Android-системи з обмеженим доступом інших застосунків. Така ізоляція є достатньою для захисту від стороннього втручання, враховуючи специфіку і призначення застосунка.

4. АНАЛІЗ РОЗРОБЛЕНОГО МОБІЛЬНОГО ЗАСТОСУНКУ

4.1. Опис інтерфейсу користувача

Інтерфейс мобільного застосунку «Exhibition Guide» складається з декількох основних екранів, що забезпечують зручну навігацію та доступ до інформації про експонати. Кожен екран виконує свою функцію, а спільним елементом для всіх є нижнє навігаційне меню. Нижня панель навігації присутня на всіх сторінках і містить три вкладки: Home, Favorites та Categories, які дозволяють швидко перемикатися між головною сторінкою, списком обраних експонатів і каталогом категорій відповідно. Активна вкладка підсвічується, що допомагає орієнтуватись, який розділ відкрито. Далі розглянемо окремо кожен екран та основні сценарії взаємодії користувача з інтерфейсом.

Головний екран застосунку (рис. 4.1) призначений для ознайомлення користувача з доступним вмістом та навігації до інших розділів. У верхній частині цього екрану відображається заголовок Home та розміщено піктограму пошуку у правому верхньому куті. Поряд із заголовком є привітальне запитання-напис: «What would you like to visit», що спонукає користувача до вибору цікавих експонатів. Нижче розташовано область зі списком популярних експонатів – на головній сторінці відображаються прев’ю декількох найбільш популярних експонатів із зображеннями. Кожен такий елемент представлений у вигляді картки. Наприклад, на рис. 4.1 у великій картці показано експонат Sony Vision-S, а під ним – секція «Over popular items» з іншими популярними експонатами. Користувач може прокручувати екран, щоб переглянути більше елементів. Таким чином, головна сторінка є відправною точкою, звідки починається ознайомлення з експонатами та здійснюється перехід до інших розділів додатку.

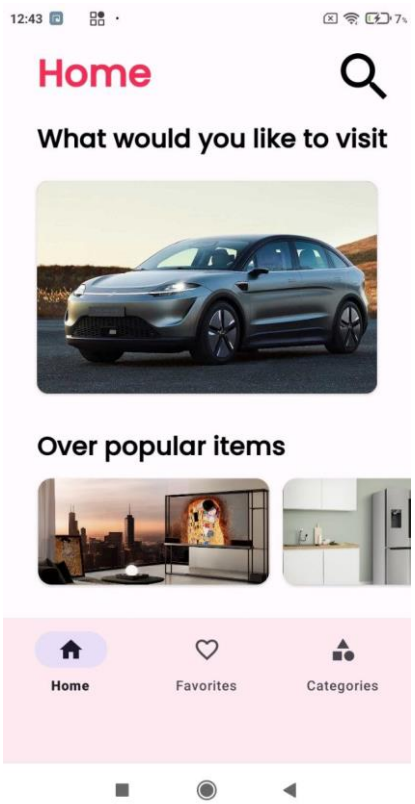


Рис. 4.1. Головна сторінка застосунку

Для детального ознайомлення з певним експонатом у додатку передбачено екран деталізації, тобто сторінка окремого експоната (рис. 4.2). На цій сторінці користувач бачить повну інформацію про вибраний експонат. Вгорі розташоване велике зображення експоната, що привертає увагу та дає візуальне уявлення про об’єкт. Під зображенням, вказується назва експоната, стилістично виділена крупним шрифтом. Також під назвою зазначена категорія та виставка, до якої належить експонат. Зокрема, в інтерфейсі використано піктограми для позначення категорії та місця проведення виставки. Основну текстову інформацію складає опис експоната, який подається блоком тексту під назвою. Цей опис дає користувачеві розгорнуту інформацію про призначення та особливості експоната. Для взаємодії на сторінці експоната передбачені спеціальні елементи, по-перше, це кнопка-додати в обране у вигляді піктограми серця. Вона розташована поверх зображення експоната.

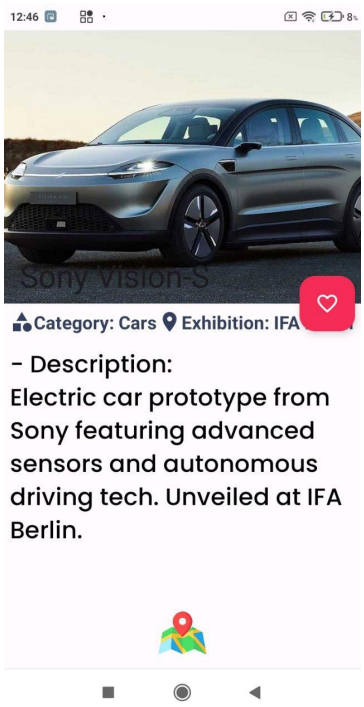


Рис. 4.2. Сторінка окремого експоната

Якщо експонат ще не додано до вибраного, натискання на значок серця додає його до списку Favorites, після чого з’являється повідомлення про успішне додавання (рис. 4.3).

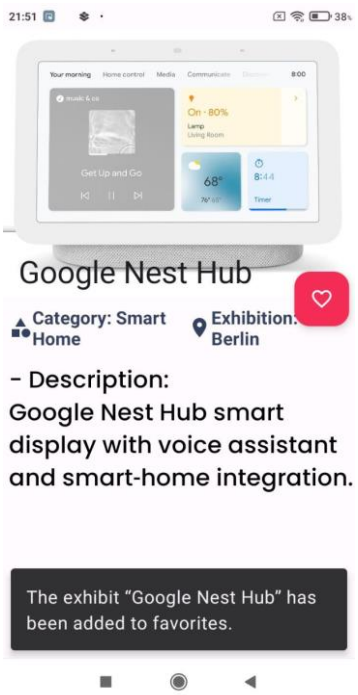


Рис. 4.3. Сторінка окремого експоната з повідомленням про успішне додавання до списку «Favorites»

По-друге, на сторінці присутня кнопка для відображення місця експоната на карті виставки – такою кнопкою слугує значок у вигляді маркера на мапі внизу екрану (нижня частина рис. 4.2). Натиснувши цю кнопку, користувач перейде до карти яка надає інформацію щодо місця розташування експоната на території виставки (рис. 4.4).

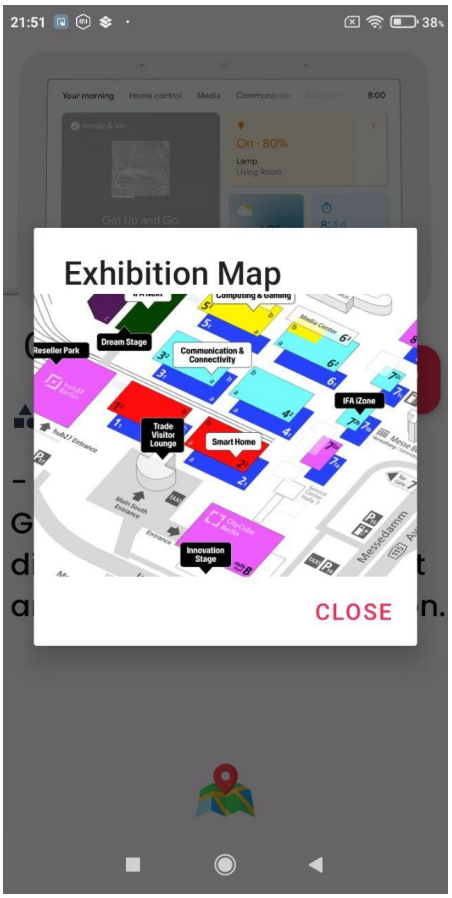


Рис. 4.4. Карта з розташуванням експоната

Перехід на сторінку детального опису експоната здійснюється з різних списків застосунку – наприклад, з головної сторінки, сторінки пошуку, категорій чи обраного – при виборі конкретного елементу.

Окремий екран присвячений пошуку експонатів (рис. 4.5) дозволяє користувачу швидко знаходити потрібні об’єкти за назвою або ключовими словами. Цей екран відкривається після натискання на значок лупи. Інтерфейс пошуку містить рядок вводу, в якому користувач може набрати текст запиту. Рядок виділено у верхній частині екрану сірим полем з

закругленими кутами для зручності вводу, праворуч в полі знаходиться піктограма стрілки, яка слугує кнопкою запуску пошуку (рис. 4.3). Під час введення запиту результати пошуку динамічно відображаються нижче у вигляді сітки карток експонатів, що відповідають критеріям. Кожен знайдений експонат представлений зображенням та назвою. Обравши потрібний експонат із результатів пошуку, користувач перейде до перегляду детальної інформації про нього – за замовчуванням це відкриє сторінку експоната.

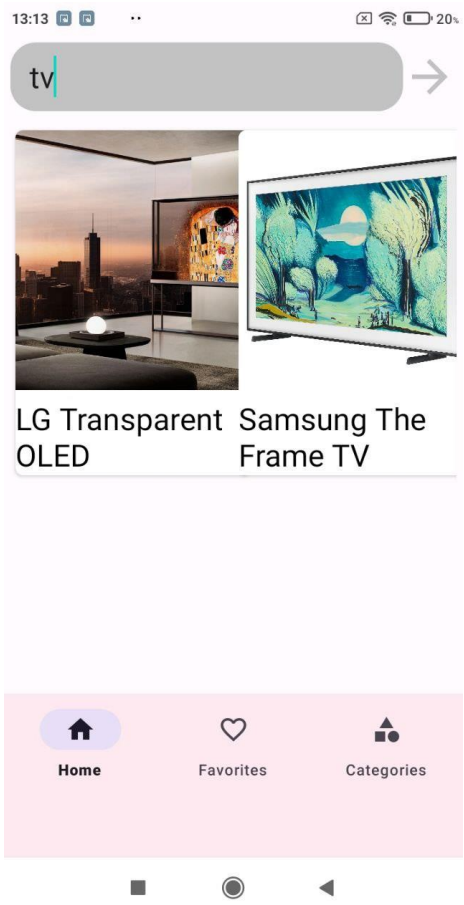


Рис. 4.5. Сторінка пошуку

Для швидкого ознайомлення з експонатом без повного переходу на його сторінку в застосунку реалізовано інформаційне нижнє вікно, яке викликається довгим натисканням на популярний експонат на головній сторінці. Це контекстне вікно з’являється знизу поверх основного інтерфейсу та містить стислі відомості про обраний експонат. На рис.4.6

продемонстровано приклад такого вікна: основний екран затемнюється, а сам Bottom Sheet виділено світло-рожевим фоном, що фокусує увагу користувача. У верхній частині відображається мініатюра вибраного експоната, а поруч текстові позначки, зліва – виставка, справа – категорія. Трохи нижче повторюється назва експоната крупним шрифтом.У нижній частині вікна розміщується кнопка «Read More…», яка відкриває повну сторінку експоната. Користувач може закрити панель натиснувши поза її межами, якщо ознайомлення з короткою інформацією було достатньо.

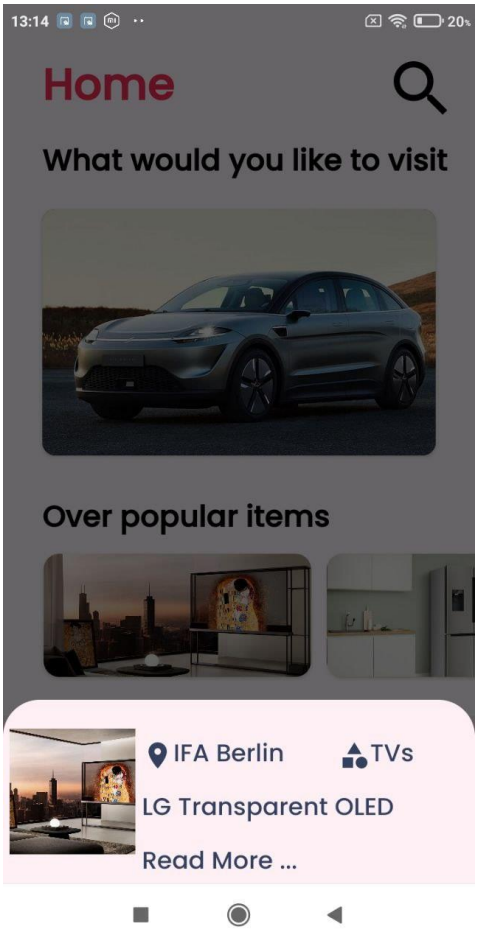


Рис. 4.6. Інформаційне нижнє вікно

Розділ обраних експонатів доступний через натискання відповідної вкладки на нижньому меню навігації. Цей екран (рис. 4.7) призначений для збереження та швидкого доступу до експонатів, помічених користувачем як улюблені. Вгорі сторінки відображається заголовок Favorites, під яким

наведено галерею доданих в обране експонатів. Кожен елемент у списку містить зображення експоната та його назву під ним. Експонати розташовуватися сіткою. Натискання на картку будь-якого експоната зі списку Favorites відкриває для нього детальну інформацію. Таким чином, розділ Favorites виконує роль персональної колекції, куди збираються вподобані об’єкти для подальшого перегляду.

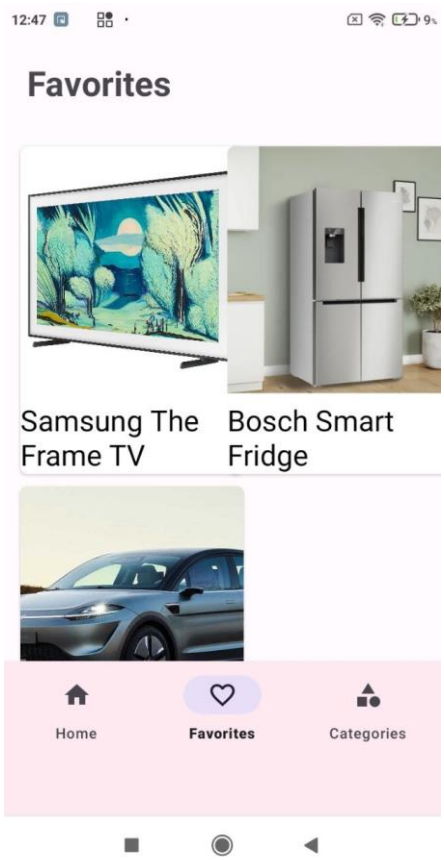


Рис. 4.7. Сторінка «Favorites»

Сторінка “Favorites” також підтримує редагування списку улюблених експонатів (рис. 4.8). Користувач може видаляти експонати зі списку свайпом у будь-якому напрямку – як вліво, так і вправо. Після цього обраний елемент зникає зі списку. Щоб уникнути випадкового видалення, застосунок автоматично показує повідомлення-підтвердження у нижній частині екрана. На панелі відображається текст на зразок «Exhibit removed from Favorites» та кнопка «Undo». Якщо користувач змінює своє рішення,

натискання на «Undo» дозволяє швидко повернути вилучений експонат до списку.

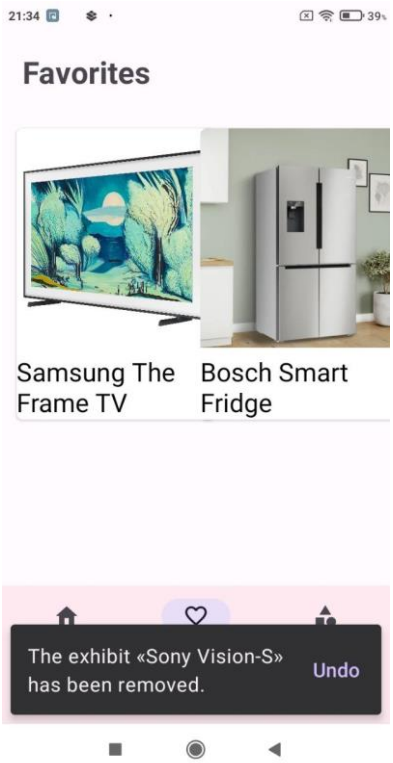


Рис. 4.8. Сторінка «Favorites» із повідомленням про видалення експоната та можливістю скасування дії

Останнім основним розділом інтерфейсу є каталог категорій експонатів, який відкривається через відповідну вкладку на нижньому навігаційному меню. Екран категорій (рис. 4.9) відображає список усіх доступних тематичних категорій, що допомагають згрупувати експонати за спільними ознаками. У верхній частині, як і на інших вкладках, показано заголовок Categories. Нижче, на світлому фоні, розміщено мініатюри, які репрезентують кожну категорію, підписані відповідними назвами. Коли користувач вибирає певну категорію, відбувається навігація до окремого списку експонатів, що належать до цієї категорії. На ньому перераховані картки всіх релевантних експонатів. Користувач може переглянути цей список та перейти на сторінки окремих експонатів, як було описано раніше.

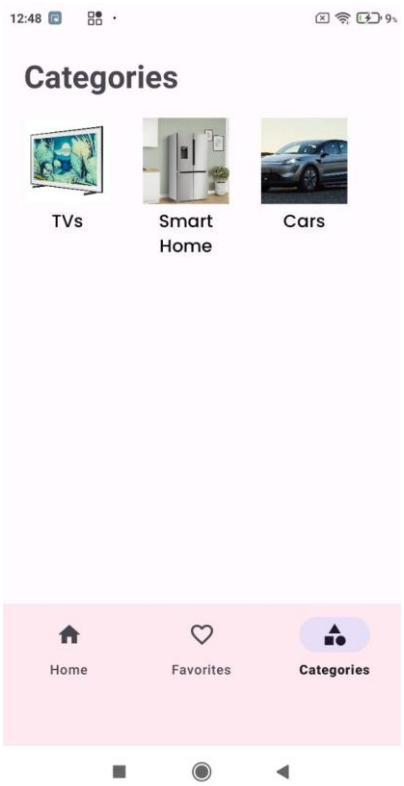


Рис. 4.9. Сторінка «Categories»

4.2. Тестування мобільного застосунку

Для забезпечення якості та надійності мобільного застосунку «Exhibition Guide» було проведено поєднане тестування, що включає як аналіз внутрішньої структури коду, так і ручне наскрізне тестування інтерфейсу користувача та основних сценаріїв використання.

Було проведено тестування якості вихідного коду Android – застосунку, із використанням плагіна CodeMR для середовища розробки. CodeMR здійснює статичний аналіз коду та візуалізує низку метрик, що характеризують внутрішню якість програмного забезпечення. Зокрема, для оцінки структури програми було проаналізовано метрики складності (Complexity) та зв’язаності (Coupling) класів. Ці показники відображають рівень складності реалізації класів та ступінь їхньої залежності від інших модулів відповідно. Відомо, що надмірна складність коду ускладнює його тестування та підтримку, а високий рівень зв’язаності між класами знижує гнучкість і повторне використання системи. Тому аналіз зазначених метрик

дозволяє об’єктивно оцінити підтримуваність та якість архітектури застосунку [20].

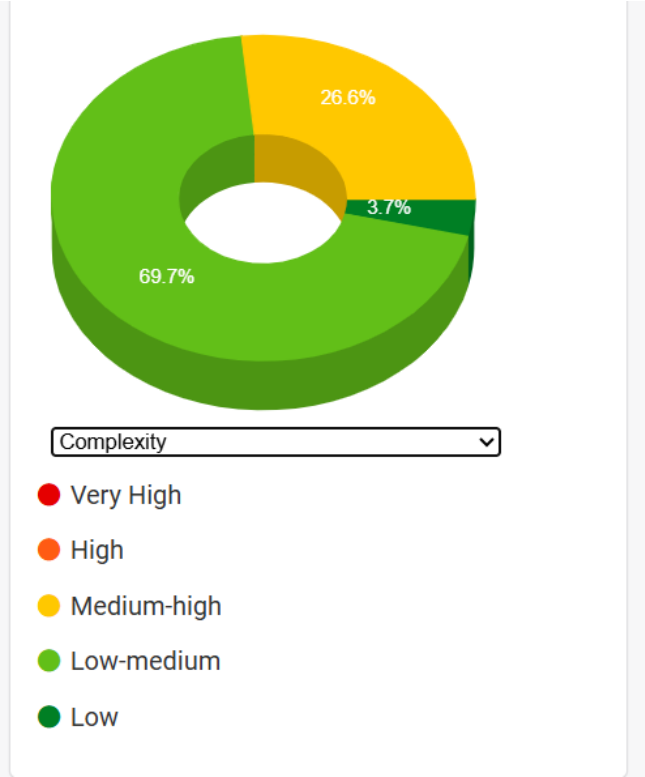


Рис. 4.10. Розподіл класів проекту за рівнем складності

Аналіз метрики складності класів показав, що більшість компонентів застосунку має невисоку складність. На діаграмі складності (рис. 4.10) видно, що приблизно 70% класів характеризуються низькою або низько-середньою складністю. Решта ~30% мають складність вище середнього рівня. Важливо відзначити, що жоден із класів не потрапив до категорій високої чи дуже високої складності – жодного «надто складного» класу в кодовій базі не виявлено. Це свідчить про те, що логіка застосунку розподілена між класами достатньо рівномірно і відсутні монолітні класи з надмірною кількістю відповідальностей. Більшість класів реалізує відносно прості задачі, що позитивно впливає на зрозумілість коду та спрощує його модифікацію.

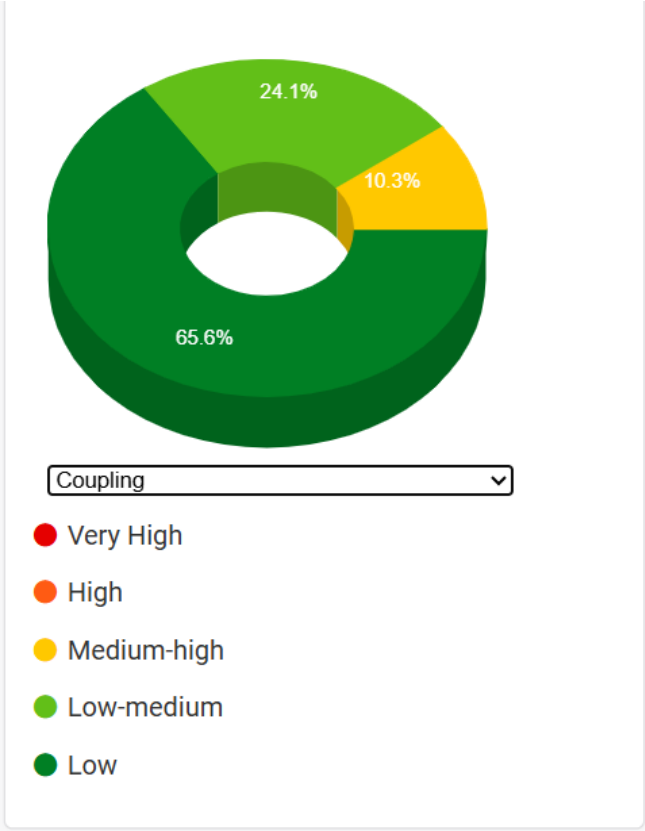


Рис. 4.11. Розподіл класів проекту за рівнем зв’язаності

Метрика зв’язаності класів продемонструвала домінування слабкого зв’язку між більшістю компонентів. Як показано на діаграмі зв’язаності (рис. 4.11), переважна частка класів – майже 90% – має низький або нижче середнього рівень зв’язаності з іншими класами. Лише близько 10% класів виявляють зв’язаність вище середнього. При цьому у проекті не зафіксовано жодного класу з високим чи дуже високим рівнем зв’язаності.

Такий результат свідчить про те, що архітектура системи добре модульована, більшість класів мало залежить від інших і взаємодіє лише з обмеженою кількістю модулів. Низька зв’язаність є бажаною характеристикою, оскільки мінімізує побічні ефекти при внесенні змін – модифікація одного класу майже не потребує змін у пов’язаних компонентах. Виявлено лише поодинокі випадки дещо вищого, але все ще помірного рівня зв’язаності.

Наскрізне тестування було проведено з метою перевірки коректної інтеграції всіх компонентів системи та загального функціонування

мобільного застосунку в умовах, наближених до реального використання [21]. Основна увага приділялася перевірці таких ключових розділів, як перегляд інформації про експонати, навігація мапою павільйонів, категоризація, пошук і робота зі списком обраного. У межах тестування було розроблено тест-кейси, які імітували типові сценарії взаємодії користувача із застосунком (табл. 4.1).

Таблиця 4.1

Таблиця тест-кейсів

№	Мета тест-кейсу	Дія	Очікуваний результат
1.	Перевірити, що головна сторінка відображає випадковий експонат	Користувач відкриває застосунок	На головній сторінці з'являється випадковий експонат
2.	Перевірити перегляд списку популярних експонатів	Користувач гортає горизонтальний список	Відображається список популярних експонатів
3.	Перевірити виклик Bottom Sheet при довгому натисканні на популярний експонат	Користувач утримує палець на експонаті зі списку популярних	Відкривається Bottom Sheet з короткою інформацією про популярний експонат
4.	Перевірити перегляд детальної інформації про випадковий експонат	Користувач натискає на картку випадкового експоната	Відображається повна інформація про експонат
5.	Перевірити перегляд детальної інформації про популярниц експонат	Користувач натискає на картку популярного експоната	Відображається повна інформація про експонат

Продовження табл. 4.1

6.	Перевірити додавання експоната до списку «Улюблені»	Користувач натискає іконку додавання до улюблених	З’являється повідомлення про додавання
7.	Перевірити перегляд мапи розташування експоната	Користувач натискає кнопку перегляду мапи	Відображається карта з локацією експоната
8.	Перевірити перегляд списку збережених улюблених експонатів	Користувач переходить до сторінки «Favorites»	Відображаються збережені експонати
9.	Перевірити видалення експоната з «Улюблених»	Користувач свайпає картку улюбленого експоната вліво або вправо	З’являється повідомлення про видалення з можливістю відміни
10.	Перевірити перегляд детальної інформації про улюблений експонат	Користувач натискає на картку улюбленого експоната	Відображається повна інформація про експонат
11.	Перевірити пошук експонатів за ключовими словами	Користувач вводить запит у пошукове поле	Відображаються картки експонатів, що відповідають введеному запиту
12.	Перевірити перегляд детальної інформації про експонат із результатів пошуку	Користувач натискає на експонат у результатах пошуку	Відображається повна інформація про вибраний експонат
13.	Перевірити перегляд категорій експонатів	Користувач відкриває сторінку категорій	Відображається перелік категорій

Продовження табл. 4.1

14.	Перевірити перегляд списку експонатів у категорії	Користувач обирає категорію	Відображається список експонатів цієї категорії
15.	Перевірити перегляд детальної інформації про експонат з певної категорії	Користувач натискає на картку експоната в категорії	Відображається повна інформація про експонат

Під час наскрізного тестування мобільного застосунку було виявлено та усунуто такі недоліки: некоректне відображення деяких карток експонатів на сторінках категорій і пошуку; помилки при додаванні та видаленні експонатів зі списку обраного; тимчасове зникнення зображень у режимі перегляду мапи. Завдяки своєчасному виявленню цих проблем і їх усуненню, фінальна версія застосунку забезпечує стабільну роботу і відповідає усім функціональним вимогам.

4.3. Пропозиції для майбутнього покращення мобільного застосунку

Незважаючи на успішну реалізацію базового функцій, для подальшого розвитку мобільного застосунку «Exhibition Guide» доцільно розглянути такі напрямки удосконалення:

1. Інтеграція з онлайн-мапами та навігацією: інтеграція з сервісами електронних карт дозволить користувачам отримувати актуальні схеми експозиційних залів і прокладати оптимальні маршрути до обраних експонатів. Це сприятиме поліпшенню орієнтації на виставковому майданчику та підвищить ефективність огляду експозиції.
2. Реалізація багатомовної підтримки: забезпечення інтерфейсу та контенту застосунку перекладом на кілька мов розширить його доступність для іноземних відвідувачів. Переклад описів

експонатів та елементів меню різними мовами значно збільшить аудиторію програми і підвищить зручність її використання.

3. Можливість додавання нотаток і позначок до експонатів: введення можливості власних коментарів та закладок до вподобаних експонатів дозволить відвідувачам персоналізувати досвід ознайомлення з виставкою. Користувачі зможуть зберігати власні нотатки та мітки, що полегшить запам’ятовування цікавої інформації і сприятиме більш глибокому вивченню експозиції.
4. Система сповіщень про зміни або нові стенди/виставки: запровадження автоматичних повідомлень інформуватиме користувачів про оновлення експозиції в реальному часі. Регулярні сповіщення про появу нових стендів або зміну експонатів привертатимуть увагу аудиторії до свіжих матеріалів і стимулюватимуть повторні відвідини виставки.

Запропоновані удосконалення спрямовані на підвищення зручності та інформативності мобільного застосунку «Exhibition Guide». Імплементація зазначених функціональних можливостей підсилить користувацький досвід, розширить цільову аудиторію та сприятиме ефективнішому плануванню відвідування виставок. У сукупності ці покращення забезпечать конкурентоспроможність програми та підвищать задоволеність користувачів.

4.4. Висновки до розділу

Підсумовуючи, було детально описано інтерфейс основних сторінок мобільного застосунку «Exhibition Guide», включно з головною сторінкою, сторінкою експоната, пошуком, категоріями та «Favorites». Усі ці екрани поєднує нижнє меню навігації, що забезпечує зручність користування. Особливу увагу приділено інтерфейсним елементам, таким як інформаційне нижнє вікно (Bottom Sheet) для швидкого перегляду експоната та інтерактивне повідомлення про додавання або видалення з обраного.

Детальний опис та ілюстрації допомогли показати взаємозв’язки між компонентами інтерфейсу.

У процесі тестування мобільного застосунку було проведено як аналіз внутрішньої структури програмного коду за допомогою плагіна CodeMR, так і ручне наскрізне тестування. Результати статичного аналізу показали, що більшість класів мають низьку або помірну складність та слабкий рівень зв’язаності, що свідчить про якісну архітектуру та високу підтримуваність коду. Ручне тестування підтвердило коректну роботу основних функцій застосунку та дозволило виявити і усунути окремі помилки на ранньому етапі. Таким чином, застосунок відповідає вимогам до стабільності, зручності використання та готовий до подальшого розвитку.

Було також запропоновано кілька напрямків для майбутнього покращення застосунку. Зокрема, інтеграція з онлайн-мапами, реалізація багатомовної підтримки, можливість створення персональних нотаток та система сповіщень – усе це сприятиме глибшій персоналізації користувацького досвіду. Такі удосконалення допоможуть адаптувати застосунок до ширшого кола користувачів, підвищити його зручність і актуальність у динамічному середовищі виставкових подій.

ВИСНОВКИ

Основною метою дипломного проєкту було створення мобільного застосунку «Exhibition Guide» для Android-платформи, який забезпечує інтерактивну навігацію виставковими павільйонами, пошук експонатів, категоризацію стендів, збереження улюблених об’єктів і отримання персоналізованих рекомендацій. Такий інструмент покликаний підвищити ефективність відвідування виставок та покращити взаємодію користувачів з експозицією.

Інформаційні технології є невід’ємною складовою сучасної виставкової індустрії, сприяючи автоматизації процесів і підвищенню зручності для учасників та відвідувачів. Існуючі рішення, як-от CES App, Whova та Hannover Messe App, хоча й надають окремі функціональні можливості, не поєднують усіх ключових опцій одночасно. Їм бракує категоризації експонатів, повноцінного списку «Улюблене», масштабованої інтерактивної карти та системи рекомендацій. Ці недоліки стали підставою для створення спеціалізованого мобільного застосунку, що охоплює весь необхідний набір функцій для ефективної навігації та взаємодії з виставковим середовищем.

Для реалізації програмного продукту було обрано сучасну мову Kotlin, яка є офіційним стандартом Android-розробки, а також архітектурний підхід MVVM, що дозволив чітко розмежувати UI, логіку представлення та доступ до даних. Застосунок використовує базу даних SQLite для зберігання локальної інформації, що забезпечує структурованість даних і зручну взаємодію між компонентами програми.

Використання компонентів Jetpack – Navigation, RecyclerView, CardView, ConstraintLayout – дало змогу реалізувати адаптивний і продуктивний інтерфейс. Для візуалізації зображень інтегровано бібліотеку PhotoView, яка забезпечує зручну роботу з медіа. Було розроблено логічну структуру даних із п’яти таблиць (exhibits, categories, exhibitions, exhibitors,

favorites), які дозволяють ефективно відображати зв’язки між експонатами, подіями, категоріями та вподобаннями користувачів.

У результаті розробки було створено повноцінний мобільний застосунок «Exhibition Guide», що пройшов комплексне тестування, включаючи аналіз коду з використанням плагіна CodeMR та ручне наскрізне тестування основних функцій. Отримані результати підтвердили якісну архітектурну побудову та стабільність застосунку в умовах реального використання. Запропоновано напрями подальшого розвитку, зокрема: інтеграцію з онлайн-мапами, багатомовну підтримку, механізм нотаток і систему сповіщень про зміни, що забезпечить масштабованість проєкту та адаптацію під потреби різних форматів виставок і міжнародної аудиторії.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Exito Media Concepts Pvt Ltd. Exito | B2B Events and Conferences | Business Events. [Електронний ресурс]. — Режим доступу: https://www.exito-e.com/role-of-digital-technology-in-event-management/ — Дата доступу: грудень 2024 р.
2. CES App. CES - The Most Powerful Tech Event in the World. [Електронний ресурс]. — Режим доступу: https://www.ces.tech/attendee-guides/ces-app/ — Дата доступу: грудень 2024 р.
3. All-in-one event management solution. Whova. [Електронний ресурс]. — Режим доступу: https://whova.com/ — Дата доступу: грудень 2024 р.
4. Besucherservice:HANNOVERMESSE App. https://www.hannovermesse.de. [Електронний ресурс]. — Режим доступу: https://www.hannovermesse.de/de/fuer-besucher/hannover-messe-app/index-2 — Дата доступу: грудень 2024 р.
5. Kotlin and Android | Android Developers. Android Developers. [Електронний ресурс]. — Режим доступу: https://developer.android.com/kotlin — Дата доступу: квітень 2025 р.
6. Коли і чому Java використовується для розробки програм - Wezom. IT-компания полного цикла разработки программных продуктов WEZOM - Киев, Украина. [Електронний ресурс]. — Режим доступу: https://wezom.com.ua/ua/blog/kogda-i-pochemu-java-ispolzuetsja-dlja-razrabotki-prilozhenij — Дата доступу: квітень 2025 р.
7. Мови програмування для андроїд: коротке порівняння. FoxmindEd. [Електронний ресурс]. — Режим доступу: https://foxminded.ua/movy-prohramuvannia-dlia-android/ — Дата доступу: квітень 2025 р.
8. About SQLite. SQLite Home Page. [Електронний ресурс]. — Режим доступу: https://sqlite.org/about.html — Дата доступу: квітень 2025 р.
9. Kuprenko V. Realm VS SQLite: Which database is better for Android apps? Bitrise Blog. Continuous Integration and Delivery (CI/CD) Platform | Bitrise.

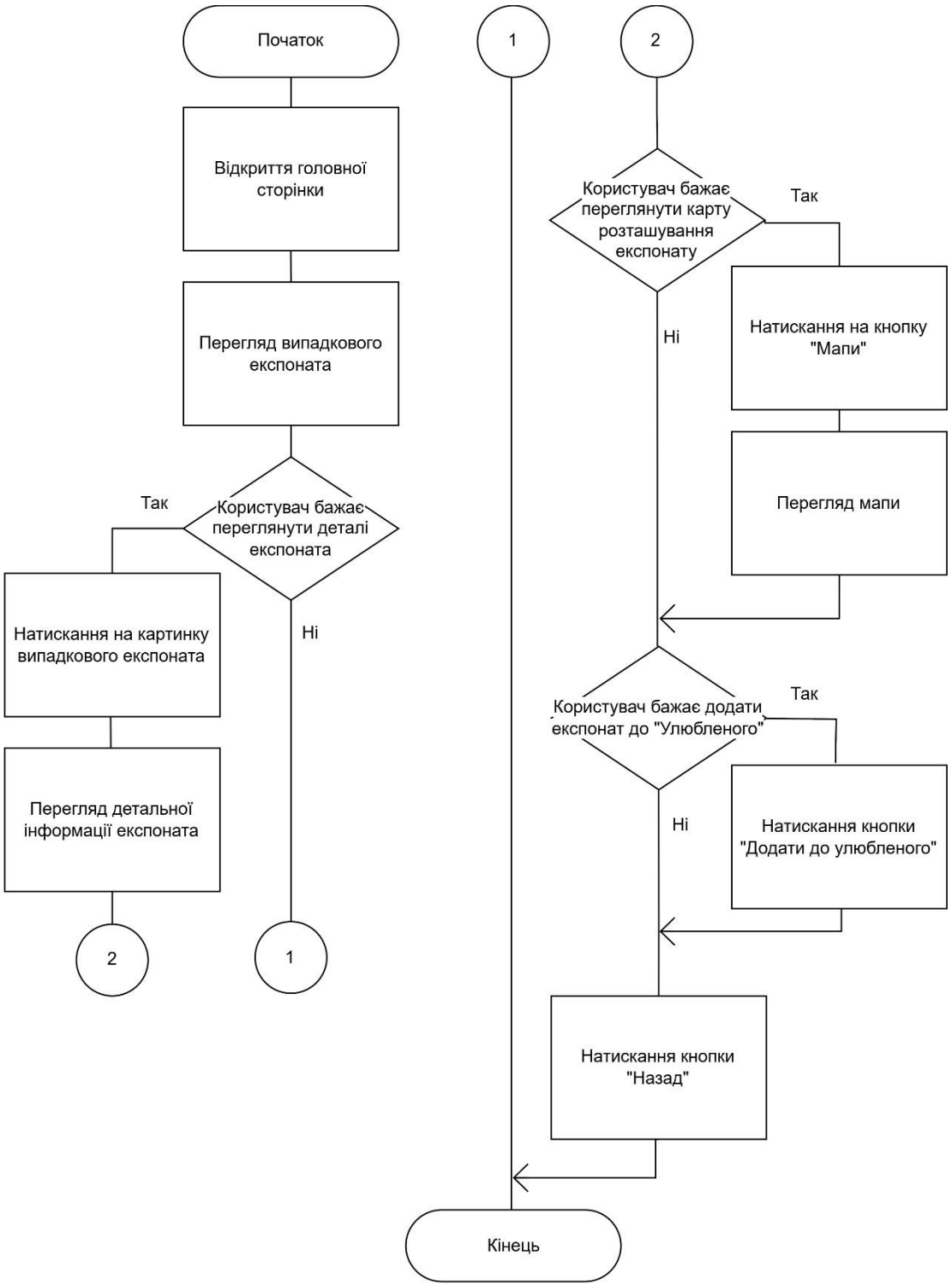
[Електронний ресурс]. — Режим доступу: https://bitrise.io/blog/post/realm-vs-sqlite-which-database-is-better-for-android-apps — Дата доступу: квітень 2025 р.

10. Jetpack Compose UI App Development Toolkit - Android Developers. Android Developers. [Електронний ресурс]. — Режим доступу: https://developer.android.com/compose — Дата доступу: квітень 2025 р.
11. Navigation | App architecture | Android Developers. Android Developers. [Електронний ресурс]. — Режим доступу: https://developer.android.com/guide/navigation — Дата доступу: квітень 2025 р.
12. ConstraintLayout | API reference | Android Developers. Android Developers. [Електронний ресурс]. — Режим доступу: https://developer.android.com/reference/androidx/constraintlayout/widget/ConstraintLayout — Дата доступу: квітень 2025 р.
13. RecyclerView | API reference | Android Developers. Android Developers. [Електронний ресурс]. — Режим доступу: https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView — Дата доступу: квітень 2025 р.
14. CardView | API reference | Android Developers. Android Developers. [Електронний ресурс]. — Режим доступу: https://developer.android.com/reference/androidx/cardview/widget/CardView — Дата доступу: квітень 2025 р.
15. GeeksforGeeks. PhotoView in Android with Example - GeeksforGeeks. GeeksforGeeks. [Електронний ресурс]. — Режим доступу: https://www.geeksforgeeks.org/photoview-in-android/ — Дата доступу: квітень 2025 р.
16. GeeksforGeeks. MVC Design Pattern - GeeksforGeeks. GeeksforGeeks. [Електронний ресурс]. — Режим доступу: https://www.geeksforgeeks.org/mvc-design-pattern/ — Дата доступу: квітень 2025 р.

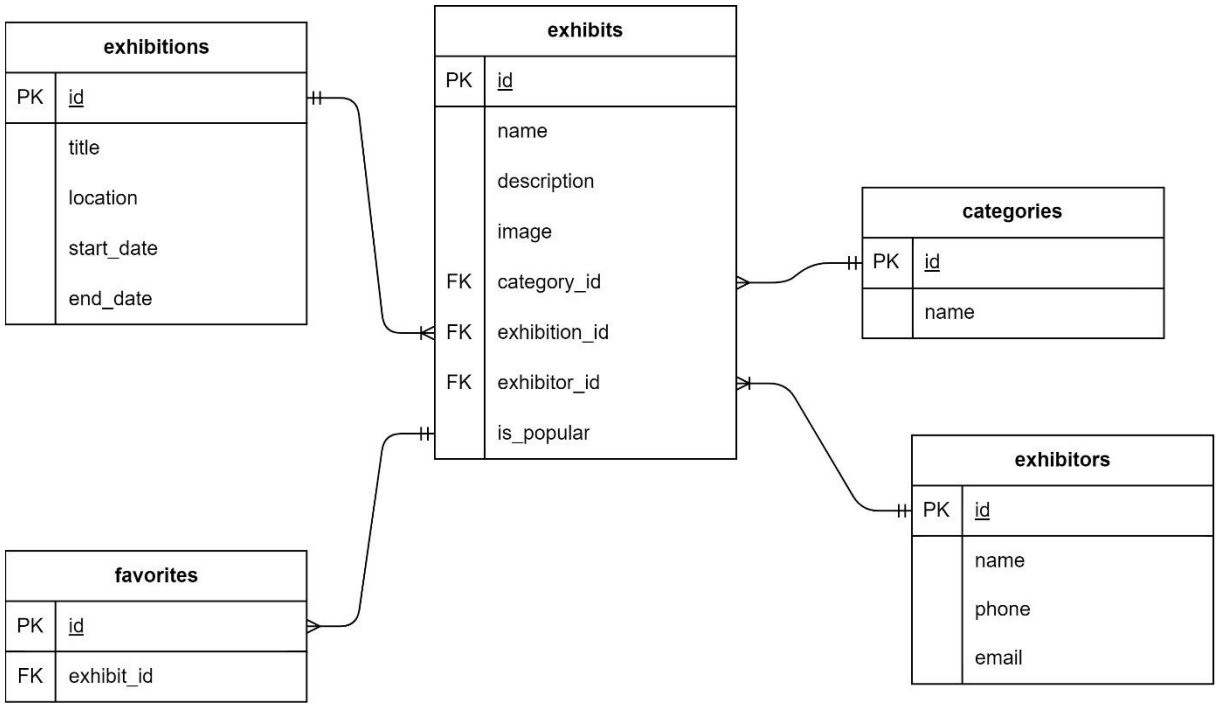
17. GeeksforGeeks. MVP (Model View Presenter) Architecture Pattern in Android with Example - GeeksforGeeks. GeeksforGeeks. [Електронний ресурс]. — Режим доступу: https://www.geeksforgeeks.org/mvp-model-view-presenter-architecture-pattern-in-android-with-example/ / — Дата доступу: квітень 2025 р.
18. Model-View-ViewModel - .NET. Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс]. — Режим доступу: https://learn.microsoft.com/en-us/dotnet/architecture/maui/mvvm — Дата доступу: квітень 2025 р.
19. GeeksforGeeks. MVVM (Model View ViewModel) Architecture Pattern in Android - GeeksforGeeks. GeeksforGeeks. [Електронний ресурс]. — Режим доступу: https://www.geeksforgeeks.org/mvvm-model-view-viewmodel-architecture-pattern-in-android/ — Дата доступу: квітень 2025 р.
20. CodeMR | Measure, visualise, and improve code quality | Better Code Better Quality!. CodeMR. [Електронний ресурс]. — Режим доступу: https://www.codemr.co.uk/ — Дата доступу: травень 2025 р.
21. Schmitt J. What is E2E? A guide to end-to-end testing. CircleCI. [Електронний ресурс]. — Режим доступу: https://circleci.com/blog/what-is-end-to-end-testing/ — Дата доступу: травень 2025 р.

ДОДАТКИ

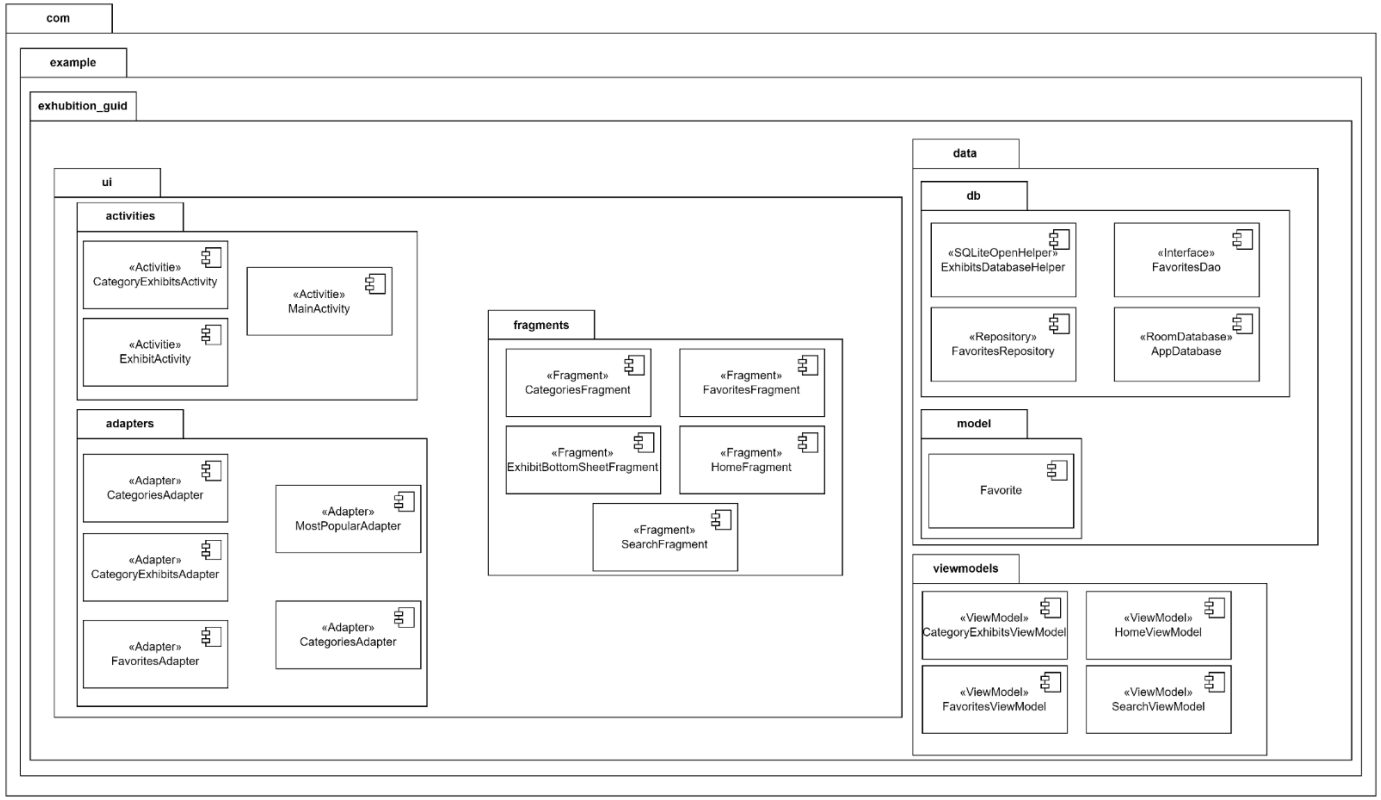
Додаток 1
Копії графічних матеріалів



ДП.045430-06-99. Мобільний застосунок «Exhibition Guide» для пропозиції та пошуку товарів на виставці. Діаграма діяльності основного сценарію використання. Схема алгоритму



ДП.045430-07-99. Мобільний застосунок «Exhibition Guide» для пропозиції та пошуку товарів на виставці. Логічна модель бази даних. Схема даних

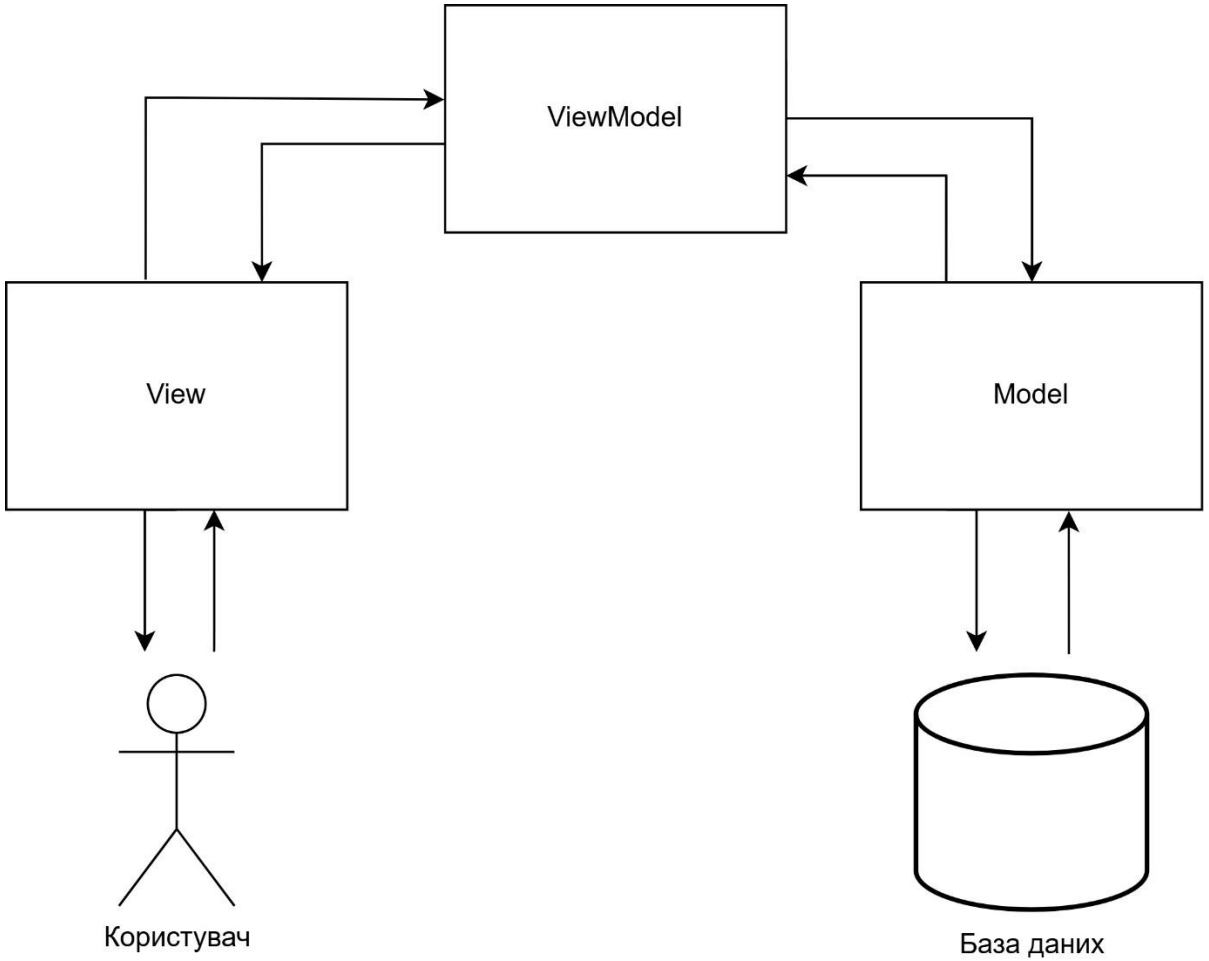


ДП.045430-08-99. Мобільний застосунок «Exhibition Guide» для пропозиції та пошуку товарів на виставці. Діаграма компонентів мобільного застосунку. Схема взаємодії програмних модулів

Дії користувача мобільного застосунку «Exhibition Guide»

Степаненко Олександр, група КП-13

Архітектура мобільного застосунку



Степаненко Олександр, група КП-13

Додаток 2
Лістинг програми

Файл MainActivity.kt

```
package com.example.exhubition_guide.ui.activities

import android.os.Bundle
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat
import androidx.navigation.Navigation
import androidx.navigation.ui.NavigationUI
import com.example.exhubition_guide.R
import com.google.android.material.bottomnavigation.BottomNavigationView

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)

        val bottomNavigation =
findViewById<BottomNavigationView>(R.id.btm_nav)
        val navController = Navigation.findNavController(this,
R.id.host_fragment)

        NavigationUI.setupWithNavController(bottomNavigation,navController)

        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main))
{ v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }
    }
}
```

Файл HomeFragment.kt

```
package com.example.exhubition_guide.ui.fragments


import android.content.Intent
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import androidx.fragment.app.Fragment
import androidx.fragment.app.viewModels
import androidx.recyclerview.widget.GridLayoutManager
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import com.example.exhubition_guide.R
import com.example.exhubition_guide.ui.activities.CategoryExhibitsActivity
import com.example.exhubition_guide.ui.activities.ExhibitActivity
import com.example.exhubition_guide.ui.adapters.CategoriesAdapter
```

```
import com.example.exhubition_guide.ui.adapters.CategoriesAdapter.Category
import com.example.exhubition_guide.ui.adapters.MostPopularAdapter
import com.example.exhubition_guide.viewmodels.HomeViewModel
import androidx.navigation.fragment.findNavController

class HomeFragment : Fragment() {

    private lateinit var imgRandomExhibit: ImageView
    private lateinit var rvPopular: RecyclerView
    private lateinit var rvCategories: RecyclerView

    private val viewModel: HomeViewModel by viewModels()
    private var exhibit: HomeViewModel.Exhibit? = null

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?, savedInstanceState:
Bundle?
    ): View? {
        val  view  =  inflater.inflate(R.layout.fragment_home,  container,
false)

        view.findViewById<ImageView>(R.id.img_search)
            .setOnClickListener {


findNavController().navigate(R.id.action_homeFragment_to_searchFragment)
            }


        imgRandomExhibit = view.findViewById(R.id.img_random_exhibit)
        viewModel.exhibitData.observe(viewLifecycleOwner) { ex ->
            exhibit = ex
            val resId = resources.getIdentifier(
                ex.image, "drawable", requireContext().packageName
            )
            imgRandomExhibit.setImageResource(
                if (resId != 0) resId else R.drawable.ic_launcher_background
            )
        }
        viewModel.loadRandomExhibit()
```

```
view.findViewById<View>(R.id.random_exhibit_card).setOnClickListener {
            exhibit?.let { ex -> openExhibit(ex) }
        }


        rvPopular = view.findViewById(R.id.rec_view_exhibits_popular)
        val popularAdapter = MostPopularAdapter(
            onClick = { ex ->

                openExhibit(ex)
            },
            onLongClick = { ex ->

                ExhibitBottomSheetFragment.newInstance(
                    id = ex.id,
                    name = ex.name,
                    description = ex.description,
                    image = ex.image,
                    category = ex.category,
                    exhibition = ex.exhibition
                ).show(childFragmentManager, "exhibit_sheet")
            }
        )
        rvPopular.layoutManager        =        LinearLayoutManager(context,
LinearLayoutManager.HORIZONTAL, false)
        rvPopular.adapter = popularAdapter
        viewModel.popularList.observe(viewLifecycleOwner)                    {
popularAdapter.submitList(it) }
        viewModel.loadPopularExhibits()

        rvCategories = view.findViewById(R.id.rec_view_categories)
        val categories = listOf(
            Category("TVs",         "samsung_frame"),
            Category("Smart Home", "bosch_fridge"),
            Category("Cars",        "sony_vision_s")
        )
        val catAdapter = CategoriesAdapter(categories) { category ->
            startActivity(Intent(requireActivity(),
CategoryExhibitsActivity::class.java).apply {
                putExtra("EXTRA_CATEGORY_NAME", category.name)
            })
        }
```

```
        rvCategories.layoutManager = GridLayoutManager(context, 3)
        rvCategories.adapter = catAdapter

        return view
    }

    private fun openExhibit(ex: HomeViewModel.Exhibit) {
        startActivity(Intent(requireActivity(),
ExhibitActivity::class.java).apply {
            putExtra("EXHIBIT_ID",          ex.id)
            putExtra("EXHIBIT_NAME",        ex.name)
            putExtra("EXHIBIT_DESCRIPTION",ex.description)
            putExtra("EXHIBIT_IMAGE",       ex.image)
            putExtra("EXHIBIT_CATEGORY",    ex.category)
            putExtra("EXHIBIT_EXHIBITION", ex.exhibition)
        })
    }
}
```

Файл HomeViewModel

```
package com.example.exhubition_guide.viewmodels

import android.app.Application
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import androidx.lifecycle.AndroidViewModel
import androidx.lifecycle.LiveData
import androidx.lifecycle.MutableLiveData
import com.example.exhubition_guide.data.db.ExhibitsDatabaseHelper

class           HomeViewModel(application:           Application)           :
AndroidViewModel(application) {

    private val dbHelper = ExhibitsDatabaseHelper(application)
    private val database: SQLiteDatabase = dbHelper.readableDatabase

    private val _exhibitData = MutableLiveData<Exhibit>()
    val exhibitData: LiveData<Exhibit> = _exhibitData

    private val _popularList = MutableLiveData<List<Exhibit>>()
    val popularList: LiveData<List<Exhibit>> = _popularList
```

```
    fun loadRandomExhibit() {
        val cursor = database.rawQuery(
            "SELECT * FROM exhibits ORDER BY RANDOM() LIMIT 1", null
        )
        if (cursor.moveToFirst()) {
            _exhibitData.postValue(readExhibitFromCursor(cursor))
        }
        cursor.close()
    }

    fun loadPopularExhibits() {
        val list = mutableListOf<Exhibit>()
        val cursor = database.rawQuery(
            "SELECT * FROM exhibits WHERE is_popular = 1", null
        )
        if (cursor.moveToFirst()) {
            do {
                list += readExhibitFromCursor(cursor)
            } while (cursor.moveToNext())
        }
        cursor.close()
        _popularList.postValue(list)
    }

    private fun readExhibitFromCursor(cursor: Cursor): Exhibit = Exhibit(
        id          = cursor.getInt(cursor.getColumnIndexOrThrow("id")),
        name       = cursor.getString(cursor.getColumnIndexOrThrow("name")),
        description                                                    =
cursor.getString(cursor.getColumnIndexOrThrow("description")),
        image                                                          =
cursor.getString(cursor.getColumnIndexOrThrow("image")),
        category                                                       =
cursor.getString(cursor.getColumnIndexOrThrow("category")),
        exhibition                                                     =
cursor.getString(cursor.getColumnIndexOrThrow("exhibition")),
        isPopular                                                      =
cursor.getInt(cursor.getColumnIndexOrThrow("is_popular")) == 1
    )

    data class Exhibit(
        val id: Int,
        val name: String,
        val description: String,
```

```
        val image: String,
        val category: String,
        val exhibition: String,
        val isPopular: Boolean
    )
}
```

Файл AppDatabase.kt

```
package com.example.exhubition_guide.data.db

import android.content.Context
import androidx.room.Database
import androidx.room.Room
import androidx.room.RoomDatabase
import com.example.exhubition_guide.data.model.Favorite

@Database(
    entities = [Favorite::class],
    version = 2,
    exportSchema = false
)
abstract class AppDatabase : RoomDatabase() {

    abstract fun favoritesDao(): FavoritesDao

    companion object {
        @Volatile
        private var INSTANCE: AppDatabase? = null

        fun getInstance(context: Context): AppDatabase =
            INSTANCE ?: synchronized(this) {
                Room.databaseBuilder(
                    context.applicationContext,
                    AppDatabase::class.java,
                    "exhibition-db"
                )
                    .fallbackToDestructiveMigration()
                    .build()
                    .also { INSTANCE = it }
            }
    }
}
```

Файл FavoritesDao.kt

```
package com.example.exhubition_guide.data.db

import androidx.lifecycle.LiveData
import androidx.room.*
import com.example.exhubition_guide.data.model.Favorite

@Dao
interface FavoritesDao {

    @Query("SELECT * FROM favorites")
    fun getAll(): LiveData<List<Favorite>>

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insert(fav: Favorite)

    @Delete
    suspend fun delete(fav: Favorite)

    @Query("SELECT EXISTS(SELECT 1 FROM favorites WHERE id = :id)")
    fun isFavorite(id: Int): LiveData<Boolean>
}
```

Файл FavoritesAdapter

```
package com.example.exhubition_guide.ui.adapters

import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import android.widget.TextView
import androidx.recyclerview.widget.DiffUtil
import androidx.recyclerview.widget.ListAdapter
import androidx.recyclerview.widget.RecyclerView
import com.example.exhubition_guide.R
import com.example.exhubition_guide.data.model.Favorite

class FavoritesAdapter(
    private val onClick: (Favorite) -> Unit
) : ListAdapter<Favorite, FavoritesAdapter.FavViewHolder>(DiffCallback) {

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
FavViewHolder {
        val v = LayoutInflater.from(parent.context)
            .inflate(R.layout.exhibit_item, parent, false)
        return FavViewHolder(v)
    }

    override fun onBindViewHolder(holder: FavViewHolder, position: Int) {
        val fav = getItem(position)
        holder.bind(fav)
        holder.itemView.setOnClickListener { onClick(fav) }
```

```
    }

    class FavViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView)
{
        private val img =
itemView.findViewById<ImageView>(R.id.img_exhibit)
        private val tv  =
itemView.findViewById<TextView>(R.id.tv_exhibit_name)

        fun bind(fav: Favorite) {
            tv.text = fav.name
            val resId = itemView.context.resources.getIdentifier(
                fav.image, "drawable", itemView.context.packageName
            )
            img.setImageResource(if (resId != 0) resId else
R.drawable.ic_launcher_background)
        }
    }

    companion object {
        private val DiffCallback = object :
DiffUtil.ItemCallback<Favorite>() {
            override fun areItemsTheSame(a: Favorite, b: Favorite) = a.id
== b.id
            override fun areContentsTheSame(a: Favorite, b: Favorite) = a
== b
        }
    }
}
```

Додаток 3
Копія презентації

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМП ’ЮТЕРНИХ СИСТЕМ

МОБІЛЬНИЙ ЗАСТОСУНОК «EXHIBITION GUIDE» ДЛЯ ПРОПОЗИЦІЇ ТА ПОШУКУ ТОВАРІВ НА ВИСТАВЦІ

Виконав: студент групи КП-13 Степаненко Олександр Вадимович

Керівник: доцент кафедри ПЗКС, к.т.н., доцент Люшенко Леся Анатоліївна

Київ – 2025

ПОСТАНОВКА ЗАДАЧІ

Мета проєкту: розробити мобільний застосунок, який надасть відвідувачам виставок ефективну навігацію експозиціями, швидкий доступ до інформації про експонати та зручне керування списком обраного.

Завдання:

1. Проаналізувати існуючі програмні рішення
2. Визначити вимоги до розробки
3. Обрати засоби реалізації
4. Розробити мобільний застосунок відповідно до вимог
5. Протестувати розроблений мбільний застосунок
6. Визначити шляхи подальшого розвитку

АКТУАЛЬНІСТЬ

- Зростання масштабів виставкових заходів вимагає ефективних цифрових інструментів для навігації та взаємодії з експозицією.
- Відвідувачі дедалі частіше обирають мобільні застосунки як зручний засіб орієнтації та пошуку експонатів.
- Використання цифрових рішень підвищує якість відвідування виставок і забезпечує персоналізований досвід.

ІСНУЮЧІ АНАЛОГИ

ВИМОГИ ДО МОБІЛЬНОГО ЗАСТОСУНКУ

Врахувавши всі переваги та недоліки існуючих рішень, було встановлено наступні вимоги, яким повинна задовільняти розроблювана система:

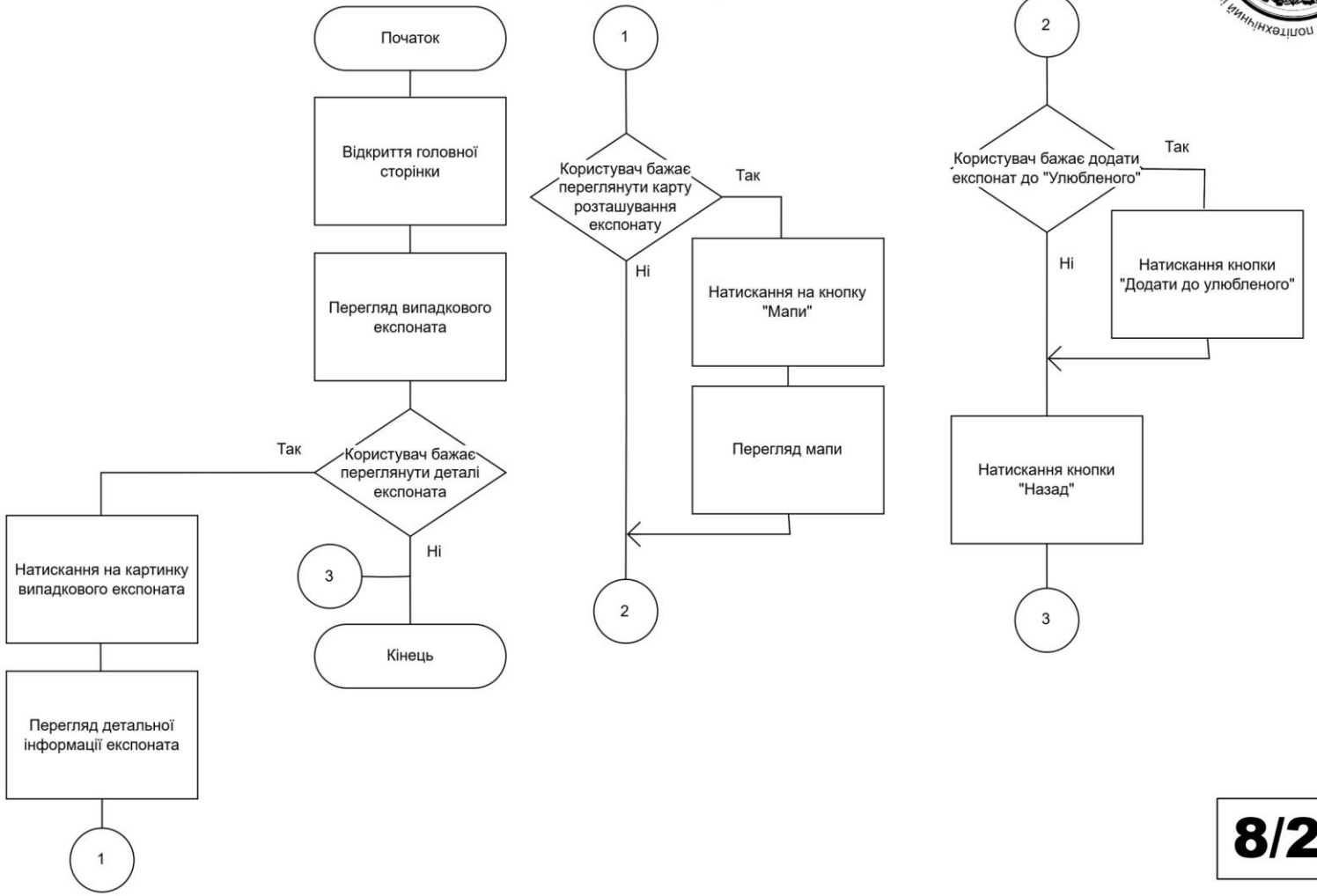
- інтерактивна карта павільйонів;
- каталог стендів із тематичною категоризацією;
- пошук експонатів за ключовими словами;
- система «Улюблене», додавання та видалення обраних стендів, перегляд списку збережених;
- рекомендації найбільш популярних стендів;
- зручний, естетичний та адаптивний інтерфейс для мобільних пристроїв.

ЗАСОБИ РОЗРОБЛЕННЯ

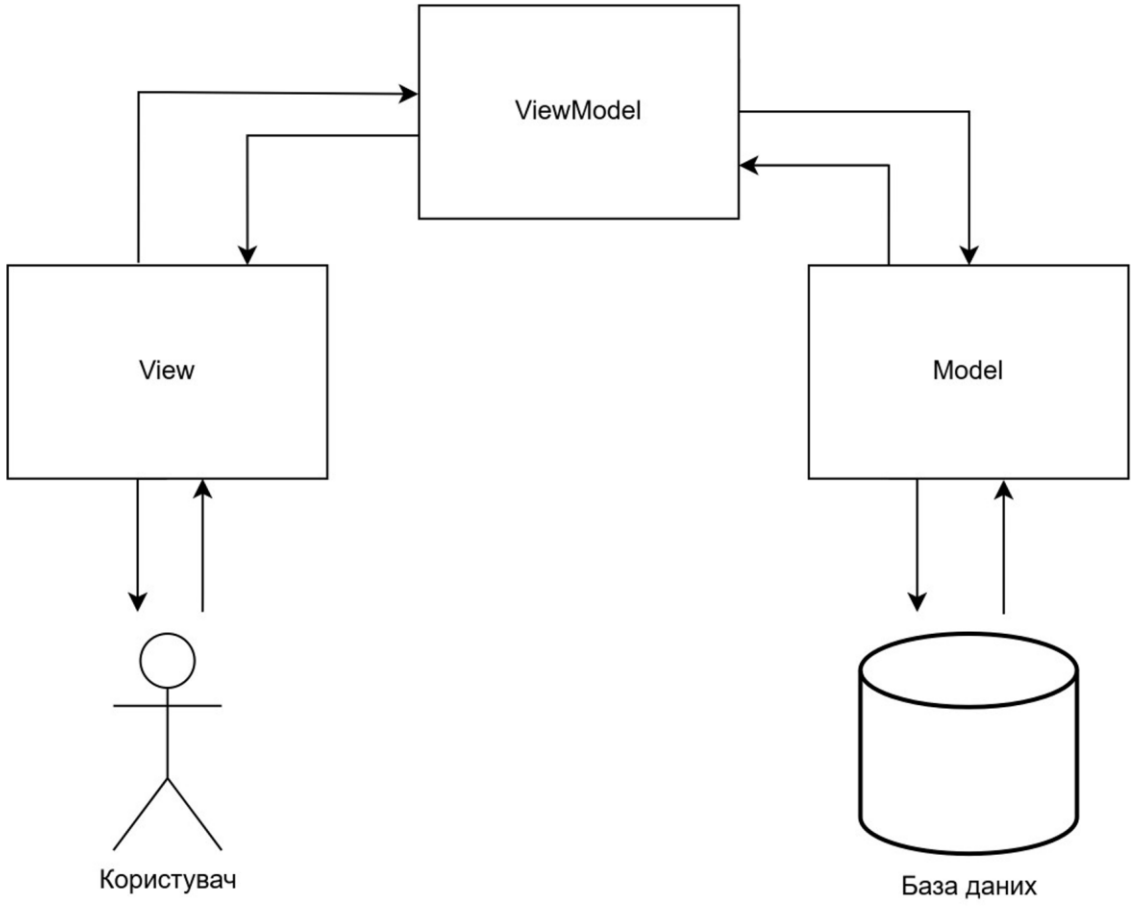
Тип ПЗ — **мобільний застосунок**.
Мова програмування — **Kotlin.**
База даних — **SQLite**.
Середовище розробки — **Android Studio.**

ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ

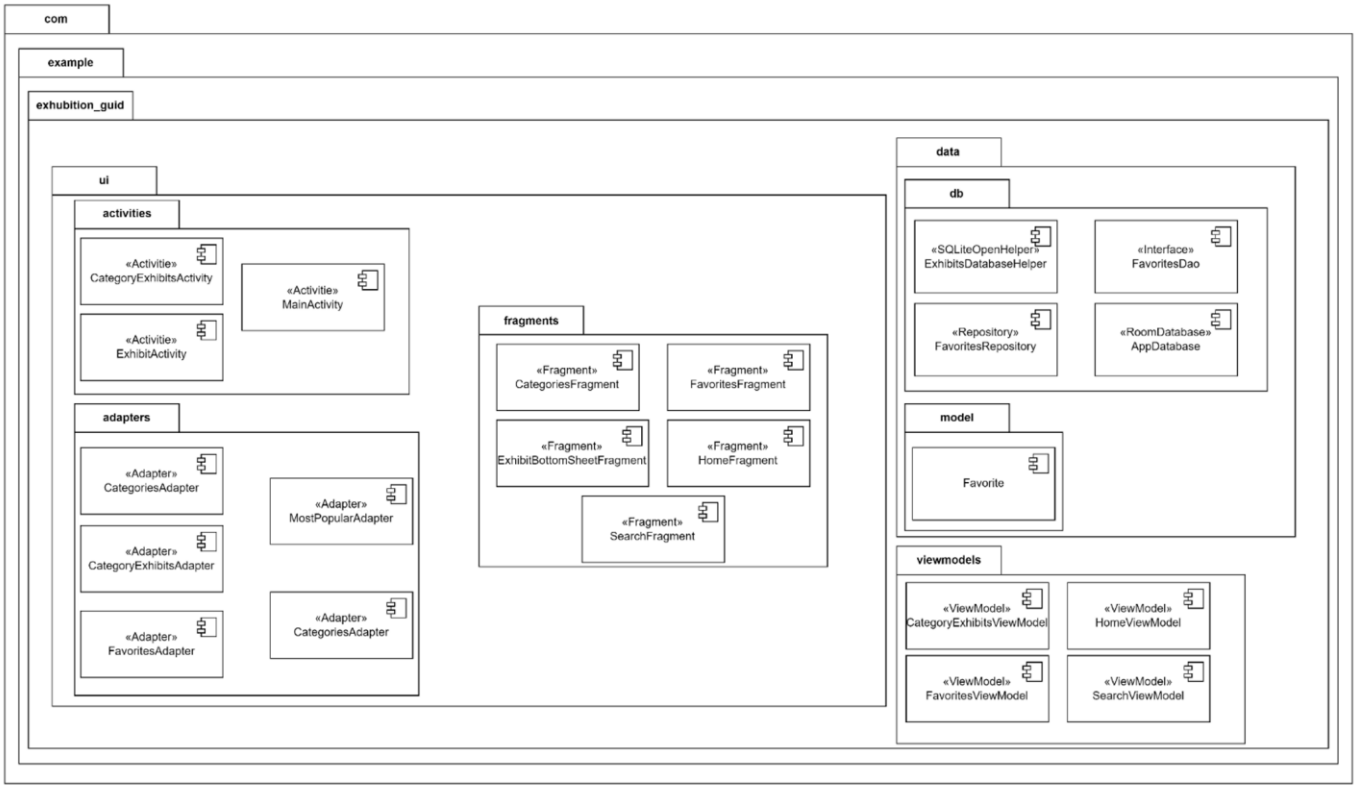
ОСНОВНИЙ СЦЕНАРІЙ ВИКОРИСТАННЯ



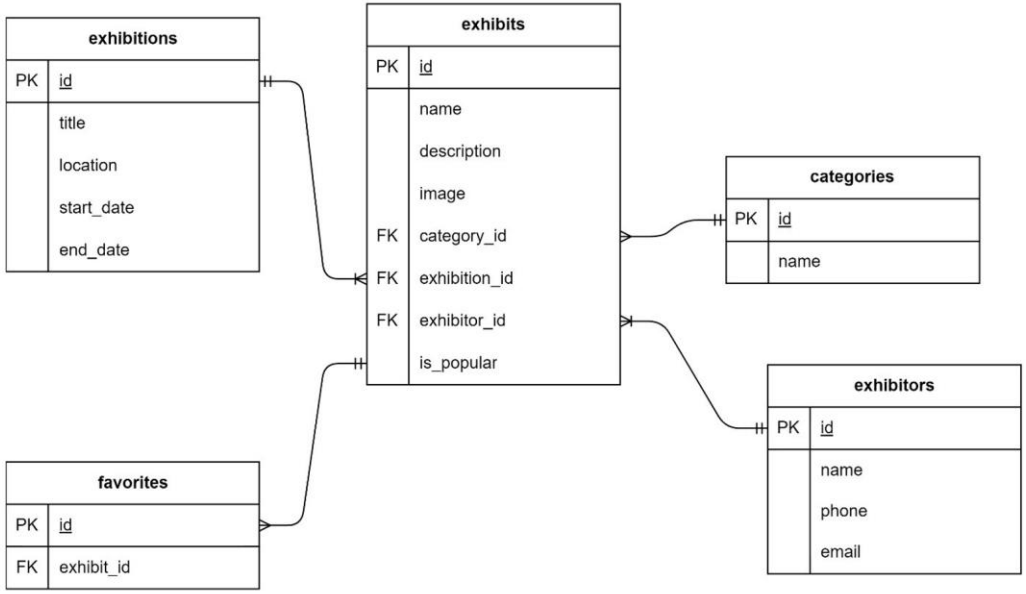
АРХІТЕКТУРА СИСТЕМИ



ДІАГРАМА КОМПОНЕНТІВ



СТРУКТУРА БАЗИ ДАНИХ

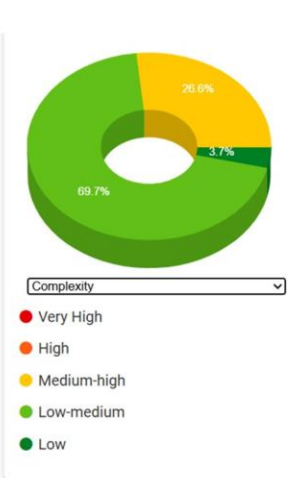


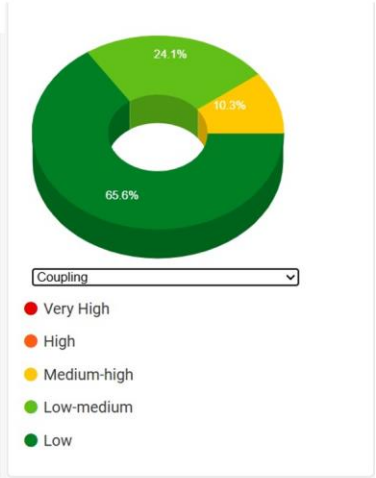
ТЕСТУВАННЯ ЗАСТОСУНКУ

Статичне тестування за допомогою CodeMR дозволило проаналізувати якість програмного коду, зосередившись на складності та зв’язаності класів.

Наскрізне тестування перевірило функціональність і продуктивність застосунку.

Всі виявлені помилки було виправлено.





ПОРІВНЯННЯ АНАЛОГІВ

Застосунок	CES App	Whova	Hannover Messe	Exhibition Guide
Інтерактивна карта павільйонів	+	–	+	+
Масштабування карти	+	–	+	+
Пошук експонатів за назвою	+	+	+	+
Категоризація експонатів	–	–	–	+
Список «Улюблене» з додаванням/видаленням	–	–	–	+
Рекомендації (найпопулярніші стенди)	–	–	–	+
Офлайн-режим карти	–	–	+	+

ШЛЯХИ ПОДАЛЬШОГО РОЗВИТКУ

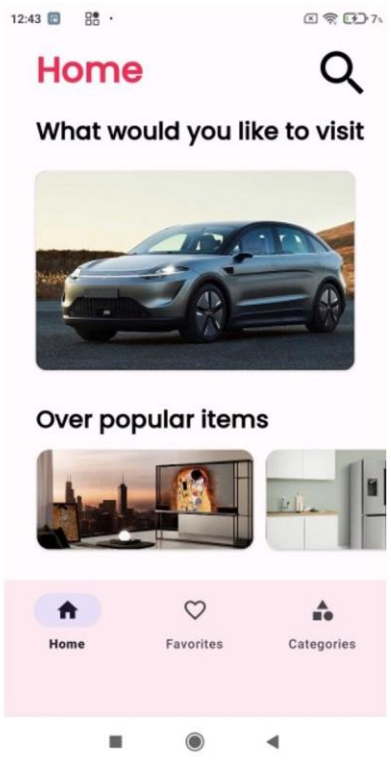
1. Інтегрувати застосунок з онлайн-мапами для навігації в реальному часі.
2. Запровадити багатомовну підтримку для користувачів з різних країн.
3. Додати можливість створення нотаток і міток до експонатів.
4. Реалізувати систему сповіщень про зміни в експозиції.

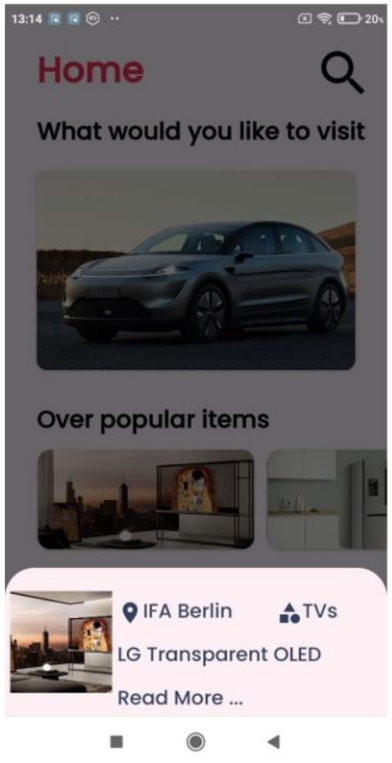
ВИСНОВКИ

1. Проаналізовано існуючі програмні рішення.
2. Обрано інструменти для розробки системи.
3. Визначено та описано вимоги до ПЗ.
4. Спроєктовано архітектуру мобільного застосунку та структуру БД
5. Розроблено мобільний застосунок згідно зазначених вимог.
6. Проведено тестування розробленого застосунку за допомогою статичного аналізу та наскрізного тестування.
7. Визначено шляхи подальшого розвитку.

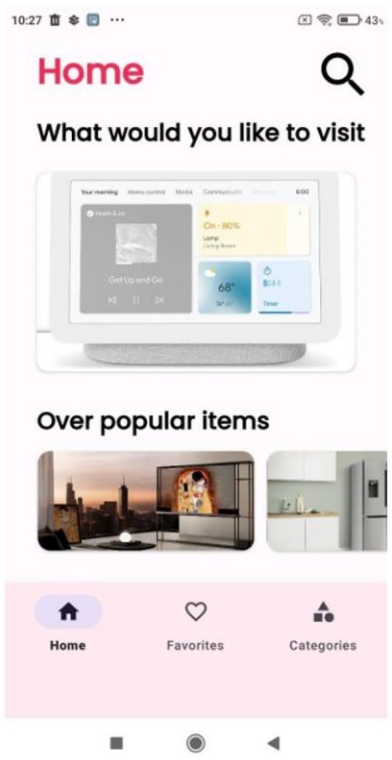
Розробка була виконана у повному обсязі, реалізовані усі поставлені вимоги до програмного продукту.

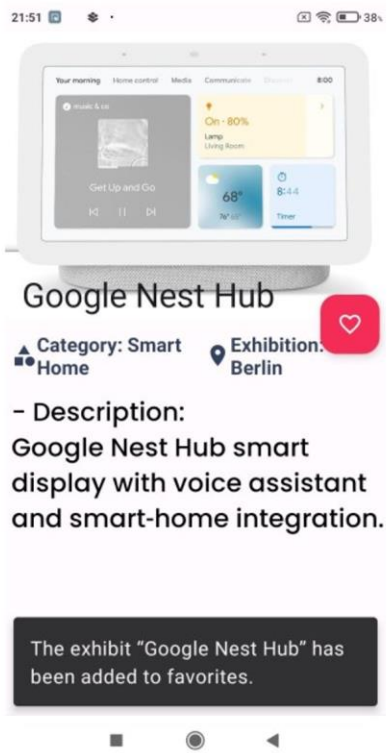
ПРИКЛАД РОБОТИ ЗАСТОСУНКУ



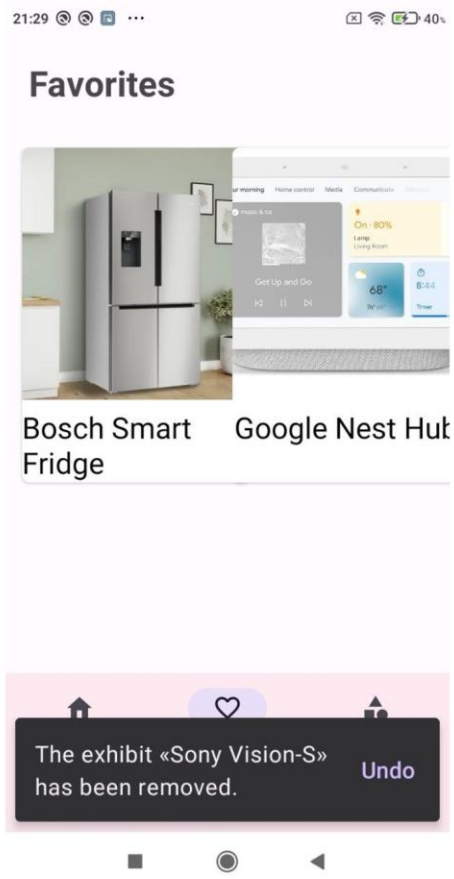


ПРИКЛАД РОБОТИ ЗАСТОСУНКУ

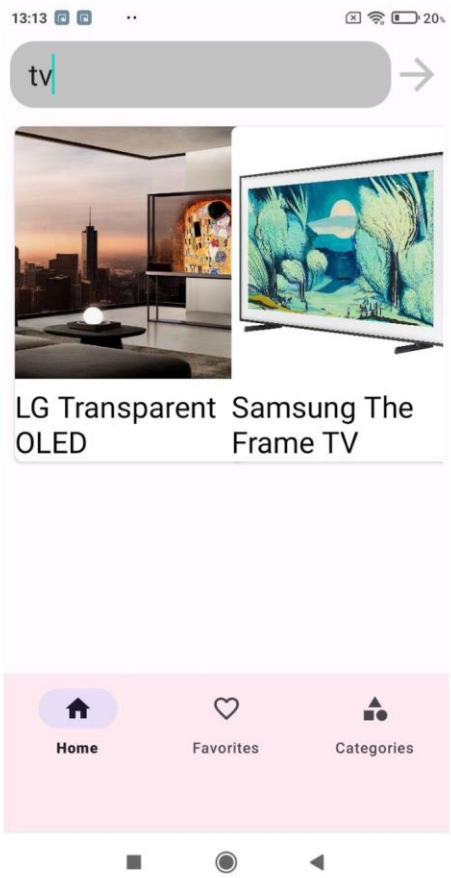




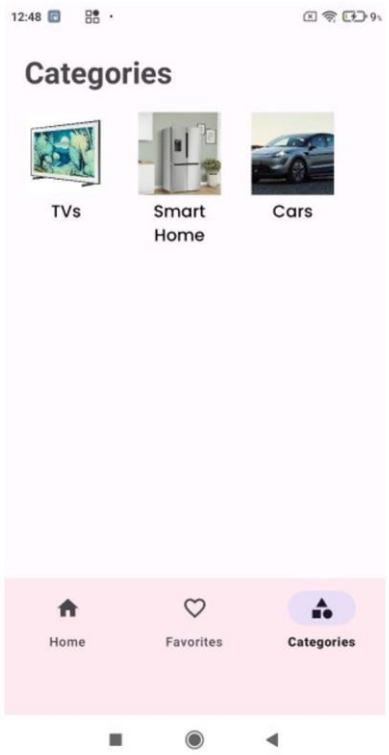
ПРИКЛАД РОБОТИ ЗАСТОСУНКУ

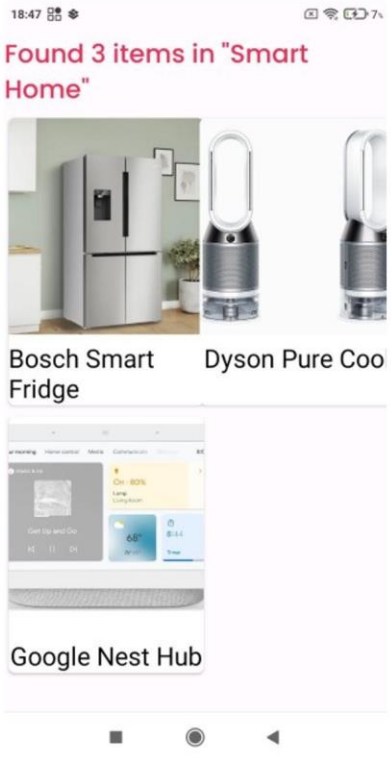


ПРИКЛАД РОБОТИ ЗАСТОСУНКУ



ПРИКЛАД РОБОТИ ЗАСТОСУНКУ





Дякую за увагу!

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

"ЗАТВЕРДЖЕНО"

Завідувач кафедри

_____ Євгенія СУЛЕМА

"___" _____ 2024 р.

МОБІЛЬНИЙ ЗАСТОСУНОК «EXHIBITION GUIDE» ДЛЯ ПРОПОЗИЦІЇ ТА ПОШУКУ ТОВАРІВ НА ВИСТАВЦІ

Програма та методика тестування

ДП.045430-04-51

"ПОГОДЖЕНО"

Керівник проєкту:

_____ Леся ЛЮШЕНКО

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Олександр СТЕПАНЕНКО

2024

ЗМІСТ

1. ОБ’ЄКТ ВИПРОБУВАНЬ

Мобільний застосунок «Exhibition Guide» для пропозиції та пошуку товарів на виставці, створений на платформі Android з використанням середовища розробки Android Studio.

2. МЕТА ТЕСТУВАННЯ

У процесі тестування має бути перевірено наступне:

1) працездатність основних елементів інтерфейсу мобільного застосунку;
2) коректність відображення інформації про експонати;
3) функціональність пошуку та фільтрації експонатів за категоріями;
4) стабільність роботи зі списком «Улюблені» (додавання, перегляд, видалення);
5) правильність відображення мапи павільйонів експонатів;
6) адаптивність дизайну для різних мобільних пристроїв;
7) зручність і інтуїтивність взаємодії користувача із застосунком.

3. МЕТОДИ ТЕСТУВАННЯ

У процесі перевірки якості мобільного застосунку було використано метод статичного тестування, який передбачає аналіз вихідного коду без його виконання. Такий підхід дозволив виявити потенційні проблеми у структурі та архітектурі застосунку ще до початку інтерактивної перевірки функціональності.

Окрім цього, було застосовано метод наскрізного тестування, що охоплює повний цикл взаємодії користувача із застосунком – від запуску та навігації мапою павільйонів до перегляду, додавання й видалення експонатів. Такий підхід дозволив перевірити інтеграцію всіх компонентів

системи та виявити логічні помилки в умовах, наближених до реального використання.

4. ЗАСОБИ ТА ПОРЯДОК ТЕСТУВАННЯ

Тестування мобільного застосунку «Exhibition Guide» проводилось на фізичному Android-пристрої, підключеному до середовища розробки Android Studio. Для виконання статичного тестування використовувався плагін CodeMR, що дозволяє здійснювати аналіз коду без його запуску.

Порядок тестування:

1) підключення фізичного пристрою до Android Studio;
2) перевірка працездатності основних функцій застосунку за допомогою наскрізного проходження сценаріїв взаємодії;
3) формування тест-кейсів для перевірки функціональних вимог, описаних у технічному завданні;
4) послідовне виконання кожного з тестових сценаріїв із фіксацією проміжних результатів;
5) проведення аналізу вихідного коду в codemr для оцінки метрик складності та зв’язаності;
6) документування усіх виявлених помилок, з подальшим усуненням і повторним тестуванням відповідних модулів.

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

"ЗАТВЕРДЖЕНО"

Завідувач кафедри

_____ Євгенія СУЛЕМА

"___" _____ 2025 р.

МОБІЛЬНИЙ ЗАСТОСУНОК «EXHIBITION GUIDE» ДЛЯ ПРОПОЗИЦІЇ ТА ПОШУКУ ТОВАРІВ НА ВИСТАВЦІ

Керівництво користувача

ДП.045430-05-34

"ПОГОДЖЕНО"

Керівник проєкту:

_____ Леся ЛЮШЕНКО

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Олександр СТЕПАНЕНКО

2025

ЗМІСТ

1. Опис структури мобільного застосунку

Мобільний застосунок «Exhibition Guide» для навігації та перегляду інформації про експонати на виставках має такі сторінки для користувачів:

- Головна сторінка (випадковий експонат, популярні експонати);
- Сторінка детального перегляду експоната (з описом, зображенням і мапою);
- Сторінка «Улюблені» (збережені експонати);
- Сторінка пошуку експонатів;
- Сторінка категорій експонатів;
- Сторінка списку експонатів у категорії.

2. Головна сторінка

Головною сторінкою мобільного застосунку вважається сторінка перегляду експонатів. На ній відображається випадково обраний експонат у вигляді великого банера, а також горизонтальний список популярних експонатів, які користувачі переглядають найчастіше. Кожен із них представлений у вигляді картки із зображенням. При натисканні на картку відкривається сторінка з детальною інформацією про експонат. У верхній частині розміщено кнопку пошуку. У нижній частині екрана знаходиться панель навігації, що забезпечує перехід на сторінки головної, обраного списку (Favorites) та категорій (рис. 2).

Для швидкого ознайомлення з експонатом без переходу на окрему сторінку, у застосунку реалізовано механізм виклику інформаційного нижнього вікна. Воно активується довгим натисканням на картку популярного експоната. Вікно з’являється внизу екрана поверх основного інтерфейсу та містить стислі відомості: мініатюру зображення, назву експоната, виставку, категорію та кнопку переходу «Read More…». Це дозволяє користувачу оперативно ознайомитися з експонатом і вирішити, чи варто відкривати повну сторінку (рис. 2).

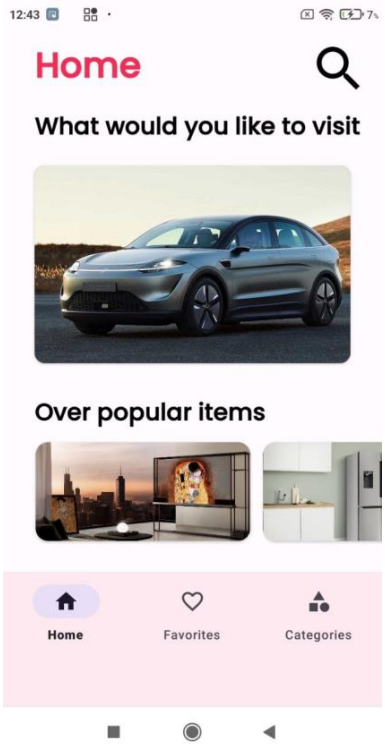


Рис. 1. Головна сторінка застосунку

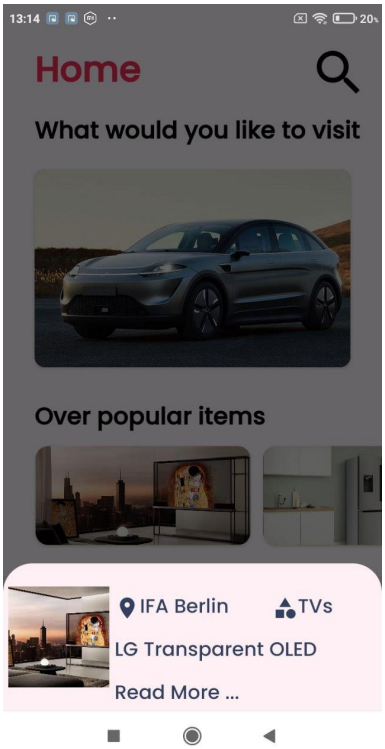


Рис. 2. Інформаційне нижнє вікно

3. Сторінка детального перегляду експоната

Сторінка детального перегляду експоната відкривається після натискання на картку з експонатом. Вона містить розширену інформацію про вибраний об’єкт, включаючи його назву, категорію, місце проведення виставки, а також текстовий опис. Основною функцією сторінки є ознайомлення користувача з деталями експоната. Користувач також має змогу додати експонат до списку «Улюблені», натиснувши на відповідну іконку.

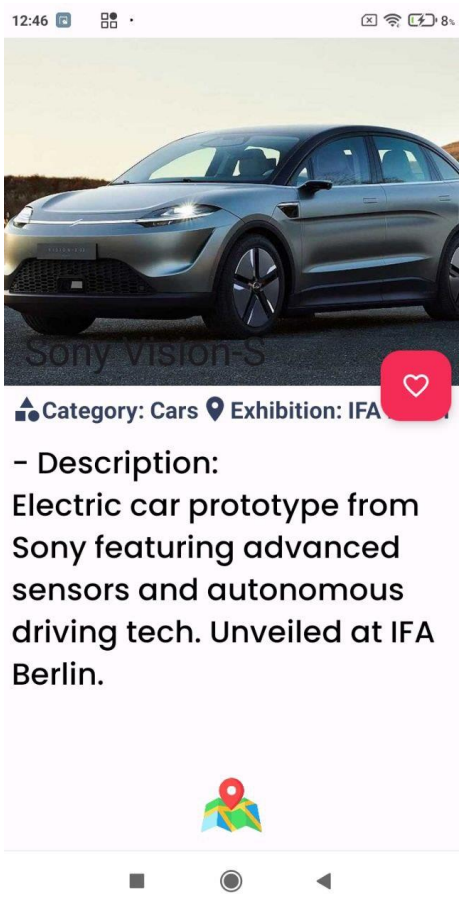


Рис. 3. Сторінка детального перегляду експоната

Додатково, через кнопку з піктограмою мапи відкривається схема виставкового павільйону з позначенням розташування стенду, де знаходиться експонат (рис. 4).

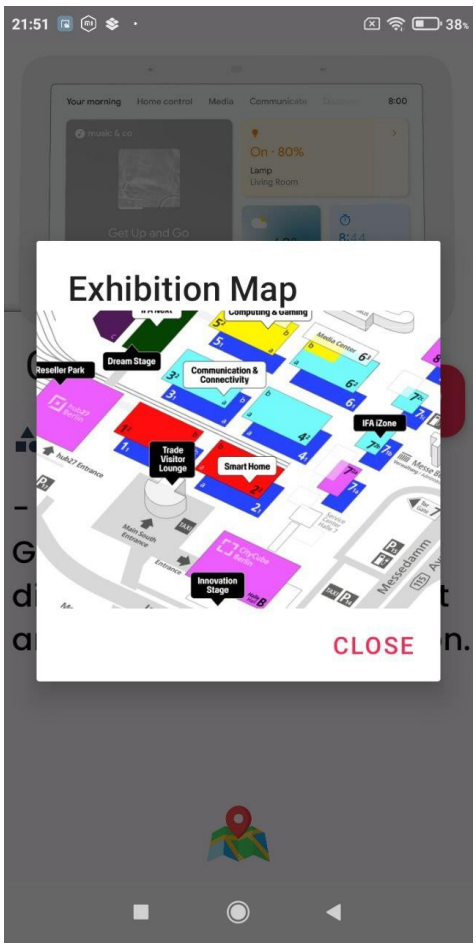


Рис. 4. Карта з розташуванням експоната

4. Сторінка «Улюблені»

Сторінка «Улюблені» дозволяє користувачам переглядати експонати, які вони попередньо зберегли. Перехід на цю сторінку здійснюється через нижню навігаційну панель. Усі збережені об'єкти відображаються у вигляді карток із зображенням та назвою. Кожну картку можна натиснути, щоб перейти до детального перегляду експоната (рис. 5).

Крім того, для зручності реалізовано можливість видалення експоната зі списку шляхом свайпу картки вбік, при цьому з’являється повідомлення з кнопкою «Undo» для скасування дії. Ця функція забезпечує швидке керування улюбленими об'єктами (рис. 6).

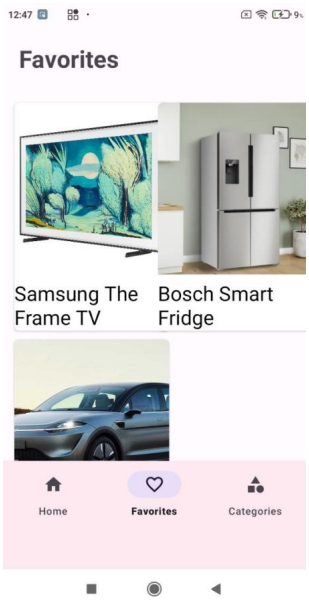


Рис. 5. Сторінка «Favorites»

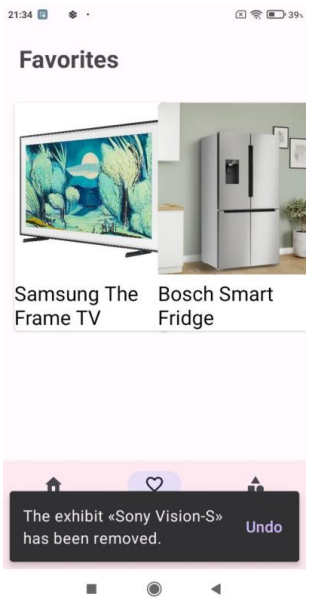


Рис. 6. Сторінка «Favorites» із повідомленням про видалення експоната та можливістю скасування дії

5. Сторінка пошуку експонатів

Сторінка пошуку експонатів надає користувачеві можливість швидко знайти потрібний об’єкт за ключовими словами. У верхній частині інтерфейсу розміщене поле введення пошукового запиту та кнопка запуску

пошуку. Після введення тексту система автоматично фільтрує експонати за назвою та відображає відповідні результати у вигляді карток. Кожен експонат, що відповідає запиту, можна відкрити для перегляду повної інформації. Це дозволяє ефективно орієнтуватися серед великої кількості представлених об’єктів.

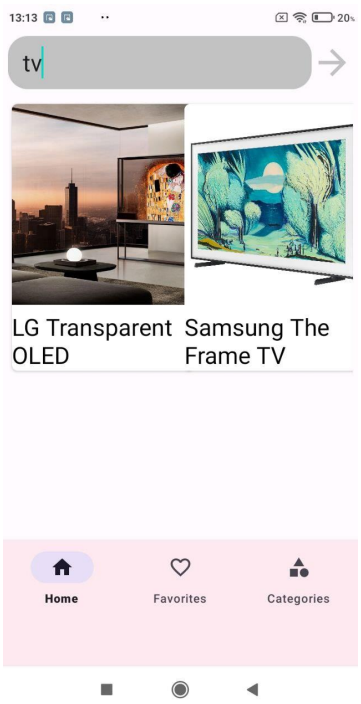


Рис. 7. Сторінка пошуку експонатів

6. Сторінка категорій експонатів

Сторінка категорій експонатів дає змогу користувачу переглядати експонати, згруповані за тематичними напрямами. Кожна категорія представлена у вигляді картки з ілюстрацією та назвою. Натиснувши на одну з категорій (наприклад, «TVs», «Smart Home», «Cars»), користувач переходить до відповідного списку експонатів, які належать до цієї теми. Такий підхід дозволяє швидко орієнтуватися серед великого обсягу інформації та знаходити потрібні об’єкти за сферою інтересів.

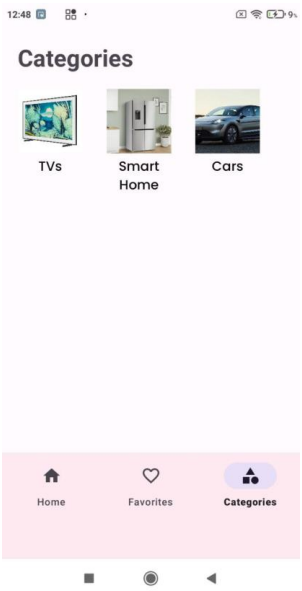


Рис. 8. Сторінка категорій експонатів

7. Сторінка списку експонатів у категорії

Сторінка відображає перелік експонатів, що належать до обраної категорії. У верхній частині зазначено назву категорії та кількість знайдених експонатів. Кожен експонат представлено у вигляді картки із зображенням і назвою. Натискання на будь-який об’єкт відкриває сторінку детального перегляду. Такий підхід дає змогу користувачеві швидко зорієнтуватися в межах обраної тематики виставки.

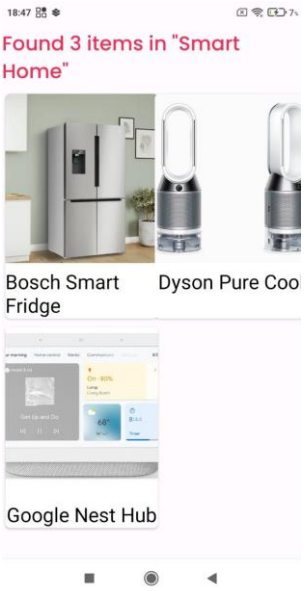


Рис. 9. Сторінка списку експонатів у категорії