Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра програмного забезпечення

Бакалаврська кваліфікаційна робота

на тему: «Розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування»

Виконав: студент 4 курсу
групи 2ПІ-21б
спеціальності
121 – Інженерія програмного забезпечення
(шифр і назва напряму підготовки, спеціальності)
Колос В.В.
(прізвище та ініціали)
Керівник: к.т.н., доц. каф. ПЗ Рейда О.М.
(прізвище та ініціали)
Рецензент: к.т.н., доц. каф. ОТ Томчук М.А.
(прізвище та ініціали)

Допущено до захисту
Завідувач кафедри ПЗ
д.т.н., проф. Романюк О.Н.
«09» 06 2025 р.

ВНТУ – 2025

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Рівень вищої освіти перший бакалаврський
Галузь знань 12 – Інформаційні технології
Спеціальність 121 – Інженерія програмного забезпечення
Освітньо-професійна програма – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
Романюк О.Н.
« 21 » березня 2025 р.

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Колосу Віталію Володимировичу

1. Тема роботи – «Розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування»

Керівник роботи: Рейда Олександр Миколайович, к.т.н., доц. кафедри ПЗ, затверджені наказом вищого навчального закладу від 20 березня 2025 р. №97.

2. Строк подання студентом роботи 2 червня 2025 р.

3. Вихідні дані до роботи: тип програми – мобільний застосунок; засоби організації даних програми – бібліотеки Room, Jetpack Compose, Dagger Hilt; вхідні дані: інформація користувача; вихідні дані: графічний інтерфейс, інформація про комплектуючі, інструкція з виконання проєктів; середовище розробки – Android Studio; мова програмування – Kotlin.

4. Зміст розрахунково-пояснювальної записки: вступ; аналіз стану розвитку застосунків для автоматизації комплектування фурнітурою меблів; розробка структури та алгоритмів програмного застосунку; розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування; тестування програмного забезпечення; висновки; список використаних джерел;

додатки.

5. Перелік графічного матеріалу: титульний слайд; актуальність теми; мета, об'єкт та предмет дослідження; задачі дослідження, наукова новизна, практична цінність одержаних результатів; порівняльний аналіз аналогів; використані технології; розробка методу оптимізації вибору меблевих комплектуючих на зображенні; база даних застосунку; тестування застосунку; висновки; апробація результатів роботи і публікації.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-4	Рейда О.М., к.т.н., доцент кафедри ПЗ	24.03.25	01.06.25

7. Дата видачі завдання 24 березня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз стану розвитку застосунків для автоматизації комплектування фурнітурою меблів	25.03.25 - 05.04.25	вик
2	Розробка структури та алгоритмів програмного застосунку	06.04.25 - 18.04.25	вик
3	Варіантний аналіз та вибір мови програмування розробки	19.04.25 - 21.04.25	вик
4	Розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування	22.04.25 - 17.05.25	вик
5	Тестування програмного забезпечення	18.05.25 - 25.05.25	вик
6	Оформлення матеріалів до захисту БКР	26.05.25 - 30.05.25	вик

Студент _____ Колос В.В.
(підпис) (прізвище та ініціали)

Керівник бакалаврської кваліфікаційної роботи _____ Рейда О.М.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

УДК 004:005.9

Колос В.В. Розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування : бакалаврська кваліфікаційна робота зі спеціальності 121 Інженерія програмного забезпечення, освітня програма – інженерія програмного забезпечення. Вінниця : ВНТУ, 2025. 91 с.

У бакалаврській кваліфікаційній роботі було розроблено мобільний застосунок для автоматизації комплектування фурнітурою меблів на етапі проектування. Він дозволяє надавати фото наявних інструментів системі, на основі чого завдяки штучному інтелекту відбувається аналіз, що знаходиться на фото, з відповідним підбором виробничих проєктів та інструментів для реалізації проєкту з використанням цих наявних інструментів.

У ході роботи було розроблено архітектуру та структуру мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування, обрано патерн MVVM для розробки архітектури застосунку, наведено структуру папок та класів, що використовуються для роботи застосунку. Розроблено модель системи у вигляді діаграми компонентів, описано компоненти, що реалізуються та використовуються у мобільному асистенті. Розроблено метод підбору меблевих проєктів на основі наявних інструментів, та метод формування інструкцій для комплектування фурнітурою меблів, побудовано алгоритми роботи застосунку.

Програмне забезпечення розроблено із використанням середовища розробки Android Studio та мови програмування Kotlin. У процесі розробки використовувались сучасні бібліотеки та інструменти, такі як Room, Jetpack Compose, Dagger Hilt. Під час тестування було перевірено працездатність основного функціоналу системи та підтверджено коректність реалізованих методів.

Ключові слова: автоматизація, мобільний застосунок, проектування, меблі, фурнітура.

ABSTRACT

UDC 004:005.9

Kolos V.V. Development of a mobile application for automating the assembly of furniture fittings at the design stage: bachelor's qualification work in the specialty 121 Software Engineering, educational program - Software Engineering. Vinnytsia: VNTU, 2025. 91 p.

In the bachelor's qualification work, a mobile application was developed for automating the assembly of furniture fittings at the design stage. It allows you to provide a photo of the available tools to the system, on the basis of which, thanks to artificial intelligence, an analysis is carried out of what is in the photo, with the appropriate selection of production projects and tools for the implementation of the project using these available tools.

In the course of the work, the architecture and structure of the mobile application for automating the assembly of furniture fittings at the design stage were developed, the MVVM pattern was chosen for the development of the application architecture, the structure of folders and classes used for the operation of the application is given. A system model was developed in the form of a component diagram, the components implemented and used in the mobile assistant were described. A method for dynamic generation of instructions based on existing tools and automatic analysis of furniture design with intelligent selection of fittings was developed, and application operation algorithms were built.

The software was developed using the Android Studio development environment and the Kotlin programming language. Modern libraries and tools such as Room, Jetpack Compose, Dagger Hilt were used in the development process. During testing, the operability of the main functionality of the system was checked and the correctness of the implemented methods was confirmed.

Keywords: automation, mobile application, design, furniture, fittings.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження. Традиційні методи комплектування фурнітурою часто характеризуються надмірними витратами часу, високою ймовірністю людської помилки та обмеженістю доступу до актуальної інформації про наявні на ринку рішення [1]. Використання цифрових технологій в меблевій індустрії створює передумови для впровадження інноваційних систем, які могли б автоматизувати різні етапи виробничого процесу [2], зокрема, етап проєктування та комплектування виробів фурнітурою, що має значний потенціал, оскільки потребує обробки великих обсягів даних та прийняття рішень на основі численних параметрів. Мобільні технології, які стали невід'ємною частиною нашого повсякденного життя, відкривають нові можливості для професійного застосування в різних галузях [3, 4]. Аналіз існуючого програмного забезпечення для проєктування меблів показує, що на ринку представлено обмежену кількість застосунків, які спеціалізуються саме на автоматизації комплектування фурнітурою [5, 6].

Більшість наявних програмних застосунків фокусуються на моделюванні геометрії виробів та візуалізації, залишаючи питання підбору функціональних компонентів на розсуд користувача. Рішення, які пропонують функціональність для роботи з фурнітурою, зазвичай є частиною комплексних систем автоматизованого проєктування, що вимагають значних обчислювальних ресурсів та не пристосовані для використання на мобільних пристроях.

Автоматизація процесу комплектування фурнітурою на етапі проєктування дозволяє оперативно реагувати на потреби ринку та пропонувати клієнтам оптимальні рішення з точки зору функціональності, естетики та вартості.

Враховуючи вищезазначені фактори, технологічні обмеження традиційних методів, економічні переваги автоматизації, зростаючі вимоги ринку та обмеженість існуючих програмних застосунків, актуальною є розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалася згідно плану виконання наукових досліджень на кафедрі програмного забезпечення.

Мета та завдання дослідження. Метою роботи є підвищення ефективності організації виробничого процесу за отриманими рекомендаціями та інструкціями шляхом автоматизації комплектування фурнітурою меблів на етапі проектування.

Основними задачами дослідження є:

- розробити архітектуру мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування;
- модифікувати метод оптимізації вибору меблевих комплектуючих для підвищення ефективності формування списку фурнітури меблів на етапі проектування;
- розробити інтерфейс користувача застосунку;
- розробити алгоритми роботи застосунку;
- розробити функціонал застосунку;
- провести тестування розробленого застосунку.

Об'єкт дослідження – процес розробки мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування.

Предмет дослідження – методи та засоби розробки мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування.

Методи дослідження. У роботі використано методи теорії прийняття рішень для обґрунтування вибору компонентів з урахуванням пріоритетів користувача, теорії баз даних для проектування реляційної моделі бази даних, методи математичного аналізу, функціональний аналіз, комп’ютерне моделювання для аналізу та перевірки отриманих теоретичних положень.

Наукова новизна отриманих результатів.

Подальшого розвитку отримав метод оптимізації вибору меблевих комплектуючих на зображені, що на відміну від існуючого, використовує модель claude-3-opus ШІ для визначення комплектуючих, і формування їх специфікацій, і як наслідок, зменшує час підготовки для виготовлення продукції.

Практична цінність отриманих результатів. Практична цінність одержаних результатів полягає в автоматизації процесу комплектування фурнітурою меблів на етапі проектування.

Особистий внесок здобувача. Усі наукові результати, викладені у бакалаврській кваліфікаційній роботі, отримані автором особисто. У роботі, виконаній у співавторстві, автору належить: аналіз методів та засобів розпізнавання з використанням ШІ, визначено переваги та недоліки платформ з ШІ. Розроблено мобільний застосунок, функціональні модулі, класи для використання в процесі розпізнавання зображень з меблевою фурнітурою.

Апробація результатів роботи. Описані у курсовому проєкті положення доповідались на конференції «LIV Всеукраїнська науково-технічна конференція підрозділів Вінницького національного технічного університету (НТКП ВНТУ–2025)».

Публікації. Результати роботи були опубліковані в матеріалах конференції «LIV Всеукраїнська науково-технічна конференція підрозділів Вінницького національного технічного університету (НТКП ВНТУ–2025)» [7].

Аналіз. У пояснювальній записці до бакалаврської кваліфікаційної роботи було розглянуто 4 розділи та було використано 30 літературних джерел.

1 АНАЛІЗ СТАНУ РОЗВИТКУ ЗАСТОСУНКІВ ДЛЯ АВТОМАТИЗАЦІЇ КОМПЛЕКТУВАННЯ ФУРНІТУРОЮ МЕБЛІВ

1.1 Аналіз стану питання розробки застосунків для комплектування фурнітурою меблів

Сучасний стан розробки спеціалізованих мобільних застосунків для автоматизації процесу комплектування меблевої фурнітури можна охарактеризувати як перехідний етап від традиційних стаціонарних програмних застосунків до інтегрованих мобільних платформ. Аналізуючи існуючі технологічні рішення на ринку програмного забезпечення для меблевої індустрії, варто зазначити, що повноцінних комплексних мобільних застосунків, які б спеціалізувалися саме на автоматизації підбору та комплектування фурнітури, наразі практично немає. Здебільшого представлені окремі функціональні модулі в складі більш масштабних програмних комплексів, що не дозволяє отримати максимальний ефект під час роботи саме з фурнітурою.

Досліджуючи еволюцію програмних застосунків у даній галузі, можна виділити три основні етапи їхнього розвитку [8]. Перший етап характеризувався використанням загальних CAD-систем (Computer-Aided Design) для проектування меблів та їхніх компонентів, при цьому процес підбору фурнітури здійснювався мануально, за допомогою каталогів. Другий етап ознаменувався появою спеціалізованих програмних комплексів для меблевої індустрії, таких як PRO100, SWOOD для SolidWorks, SketchList, Polyboard, які мали вбудовані бібліотеки стандартної фурнітури, проте не забезпечували повної автоматизації процесу її підбору та комплектування. Сучасний, третій етап характеризується початком інтеграції технологій штучного інтелекту та мобільних платформ у процес проектування меблів, проте ця сфера все ще залишається недостатньо розвиненою.

Варто відзначити, що наявні на ринку мобільні застосунки для меблевої індустрії здебільшого фокусуються на візуалізації готових меблевих дизайнів та їх розміщенні в інтер'єрі за допомогою технологій доповненої реальності (AR) [9]. Такі застосунки як Moblo, HomeByMe, IKEA Place та подібні їм зосереджені на

кінцевому споживачеві, а не на професійних дизайнерах чи виробниках меблів. Ці програми дозволяють переглядати та розміщувати віртуальні моделі меблів у реальному просторі, але не надають інструментів для детального проектування та специфікації компонентів, зокрема фурнітури.

Аналіз патентної документації за останні п'ять років демонструє зростання кількості заявок на винаходи та корисні моделі, пов'язані з автоматизацією процесів проектування меблів [10]. Згідно з даними Всесвітньої організації інтелектуальної власності (WIPO), кількість патентних заявок у сфері автоматизації меблевого виробництва збільшилася на 28% порівняно з попереднім п'ятирічним періодом. Цей факт свідчить про підвищення інтересу до розробки нових технологічних розробок у цій галузі, зокрема й мобільних застосунків для автоматизації комплектування фурнітурою.

Одним із ключових викликів у розробці мобільних застосунків для комплектування фурнітурою є створення повної та актуальної бази даних меблевої фурнітури. Різноманітність виробників, моделей, технічних характеристик та сумісності фурнітури з різними матеріалами та конструкціями меблів вимагає розробки складних алгоритмів класифікації та пошуку. Дослідження, проведені компанією McKinsey & Company у 2023 році, показують, що використання технологій штучного інтелекту та машинного навчання для автоматизації процесу підбору компонентів у виробничих процесах може підвищити точність підбору на 35-40% та зменшити кількість помилок на 45-50% [11]. Проте застосування цих технологій у мобільних платформах має свої обмеження, пов'язані з обчислювальною потужністю мобільних пристроїв та необхідністю оптимізації алгоритмів.

Важливим аспектом розробки мобільних застосунків для автоматизації комплектування фурнітурою є інтеграція з існуючими системами автоматизованого проектування (CAD) та системами управління ресурсами підприємства (ERP) [12]. Функціональна сумісність різних програмних комплексів дозволяє створити єдиний інформаційний простір для всіх етапів виробничого процесу – від проектування до виготовлення та продажу готових виробів. Проте наразі

спостерігається фрагментація ринку програмного забезпечення для меблевої індустрії, що ускладнює інтеграцію різних систем та створення єдиних стандартів обміну даними.

Результати опитування, проведеного Європейською федерацією виробників меблів (European Furniture Manufacturers Federation) серед 120 компаній-виробників меблів у 2024 році, вказують на те, що 78% респондентів вважають автоматизацію процесу комплектування фурнітурою одним із пріоритетних напрямків технологічного розвитку [13]. При цьому 65% опитаних зазначили, що хотіли б мати мобільний застосунок, який би дозволяв здійснювати підбір та специфікацію фурнітури безпосередньо на виробництві або під час зустрічі з клієнтом. Це свідчить про наявність попиту на відповідні програмні рішення та потенціал для розвитку цього сегменту ринку.

З точки зору технологічних аспектів розробки мобільних застосунків для комплектування фурнітурою, важливу роль відіграють такі фактори, як інтерфейс користувача, продуктивність та точність алгоритмів підбору, можливість роботи в автономному режимі та інтеграція з іншими системами. Особливої уваги заслуговує розробка інтуїтивно зрозумілого інтерфейсу, який би дозволяв швидко здійснювати підбір фурнітури відповідно до конкретних параметрів меблевого виробу. Дослідження в області взаємодії людини з комп'ютером (Human-Computer Interaction, HCI) демонструють, що такий інтерфейс може скоротити час на виконання типових операцій на 25-30% порівняно з традиційними методами [14].

Аналіз сучасних тенденцій у розробці мобільних застосунків для меблевої індустрії дозволяє виділити декілька перспективних напрямків розвитку [15]. По-перше, це інтеграція технологій доповненої та віртуальної реальності для візуалізації меблевих конструкцій з різними варіантами фурнітури. По-друге, використання технологій комп'ютерного зору для розпізнавання та ідентифікації фурнітури за фотографіями чи ескізами. По-третє, застосування алгоритмів штучного інтелекту для підбору фурнітури відповідно до таких параметрів, як ціна, якість, довговічність, естетичні характеристики тощо. По-четверте, інтеграція з системами електронної комерції для оперативного замовлення необхідної

фурнітури безпосередньо з мобільного застосунку.

Важливим аспектом є також адаптація мобільних застосунків до специфічних потреб різних категорій користувачів – від великих меблевих фабрик до малих столярних майстерень та індивідуальних дизайнерів. Кожна з цих категорій має свої особливості роботи з фурнітурою, різні обсяги виробництва та різні вимоги до функціональності програмного забезпечення. Створення гнучких, масштабованих застосунків, які могли б адаптуватися до різних сценаріїв використання, є однією з ключових вимог до сучасних мобільних застосунків для меблевої індустрії.

Підсумовуючи аналіз стану питання розробки мобільних застосунків для автоматизації комплектування фурнітурою меблів, можна зазначити, що ця галузь має значний потенціал для розвитку та інновацій. Наявність потреби в покращенні процесу підбору та комплектування фурнітури, технологічні можливості сучасних мобільних платформ та загальна тенденція до діджиталізації виробничих процесів створюють сприятливі умови для розробки спеціалізованих мобільних застосунків.

1.2 Порівняльний аналіз аналогів

З метою дослідження ринку застосунків для підбору комплектуючих виробничих проектів, визначення їх можливостей, та формування вимог до власної розробки, необхідно провести порівняльний аналіз аналогів. Наразі існують такі застосунки для автоматизації комплектування фурнітурою меблів на етапі проектування: IKEA Assemble APP, Woodworking Calculator Pro

IKEA Assemble App [16] – офіційний застосунок від IKEA, доступний для iOS та Android, що включає каталог продуктів, доповнену реальність для розміщення меблів у просторі, і найважливіше – детальні інструкції зі збірки, включаючи специфікації фурнітури.

Хоча застосунок фокусується на меблях IKEA, він надає корисний інструментарій для роботи з фурнітурою: кожна інструкція містить точний перелік компонентів (гвинти, шурупи, кріплення, ручки тощо), їх кількість та поетапні інструкції з монтажу. IKEA Mobile також дозволяє користувачам перевіряти наявність запасних частин фурнітури та замовляти їх безпосередньо з застосунку,

що особливо корисно при ремонті або реставрації меблів.

Інтерфейс IKEA Assemble App наведено на рисунку 1.1.

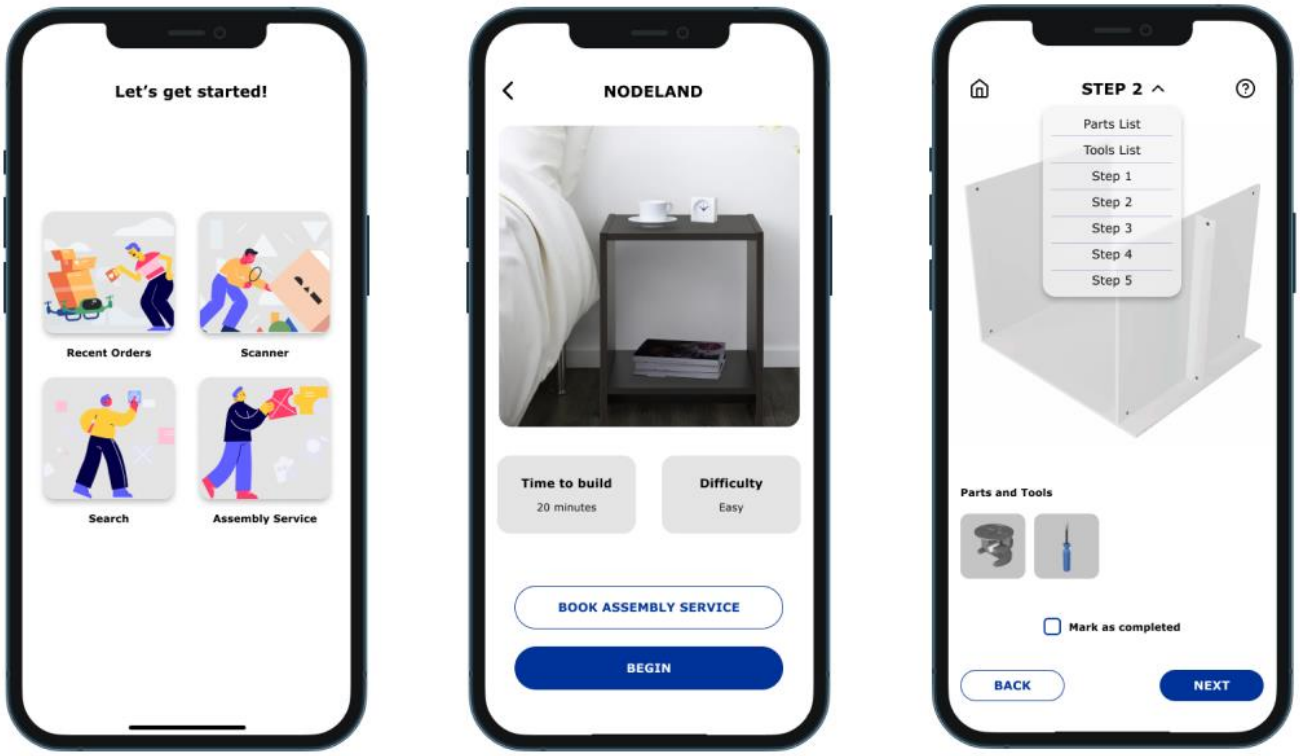


Рисунок 1.1 — Інтерфейс IKEA Assemble App

Woodworking Calculator Pro [17] – практичний застосунок для столярів та майстрів з виготовлення меблів, доступний у Google Play. Він надає широкий набір калькуляторів та інструментів для планування проектів, включаючи розрахунок необхідних матеріалів та кріплень. Woodworking Calculator Pro дозволяє користувачам створювати специфікації проектів з детальним переліком матеріалів, інструментів та фурнітури.

Застосунок містить базу даних стандартних розмірів фурнітури та рекомендацій з її використання для різних типів з'єднань. Додатковими функціями є конвертер одиниць вимірювання, калькулятор розкрою матеріалів та можливість експорту специфікацій у форматі PDF.

Інтерфейс Woodworking Calculator Pro наведено на рисунку 1.2.

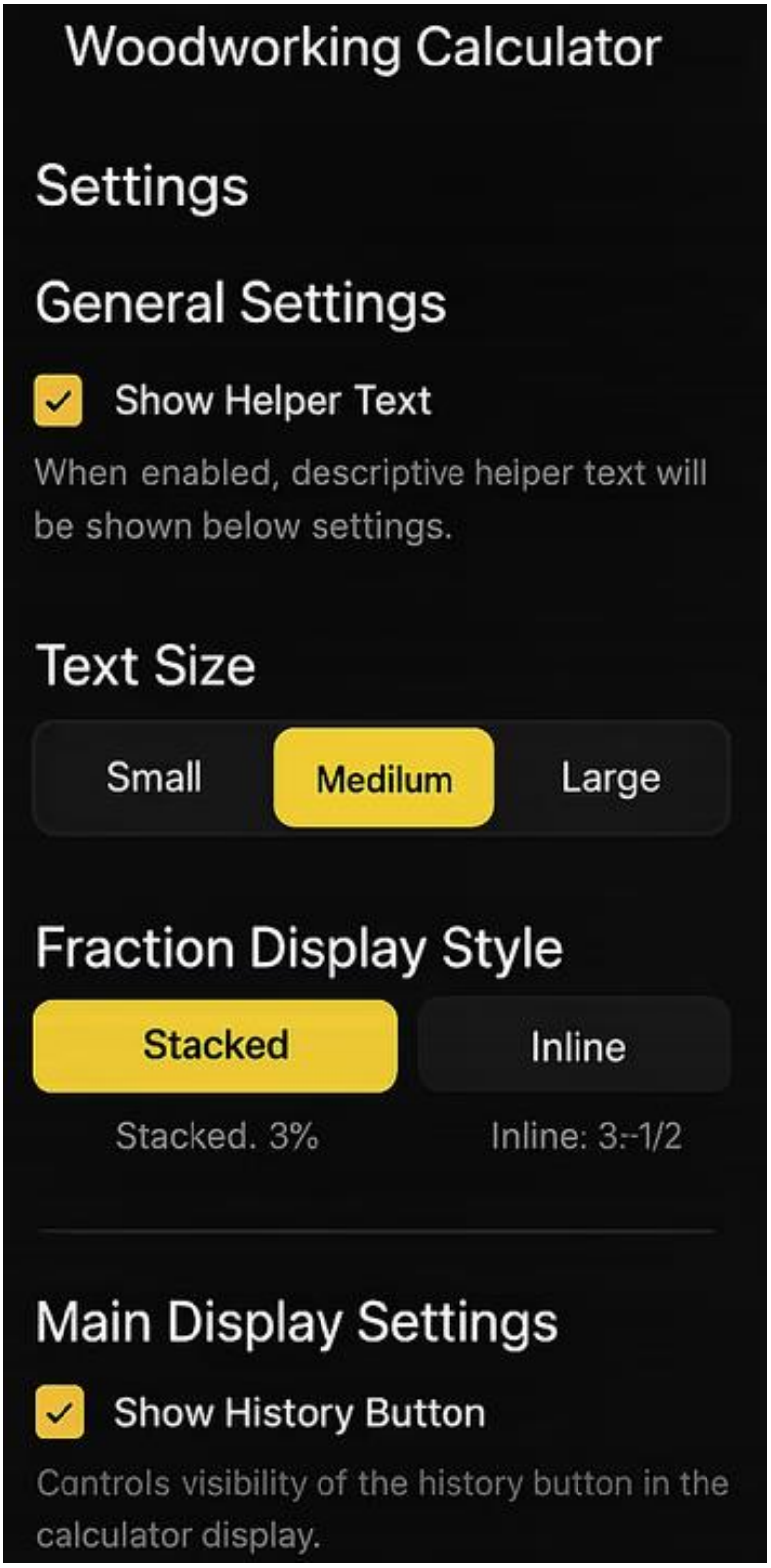


Рисунок 1.2 — Інтерфейс Woodworking calculator

Blum Configurator [18] – офіційний застосунок від відомого виробника меблевої фурнітури Blum, доступний у App Store та Google Play. Застосунок дозволяє користувачам проектувати корпусні меблі та підбирати необхідну

фурнітуру Blum (петлі, підйомники, напрямні). Blum Configurator дозволяє створювати детальні специфікації проекту з використанням інтерактивного конфігуратора. Користувачі можуть вводити розміри та параметри шаф, ящиків та дверцят, після чого система автоматично підбирає оптимальні компоненти фурнітури і генерує специфікацію з артикулами для замовлення. Застосунок також містить інструкції з монтажу з докладними ілюстраціями та відео, що полегшує процес збірки.

Інтерфейс Blum Configurator наведено на рисунку 1.3.

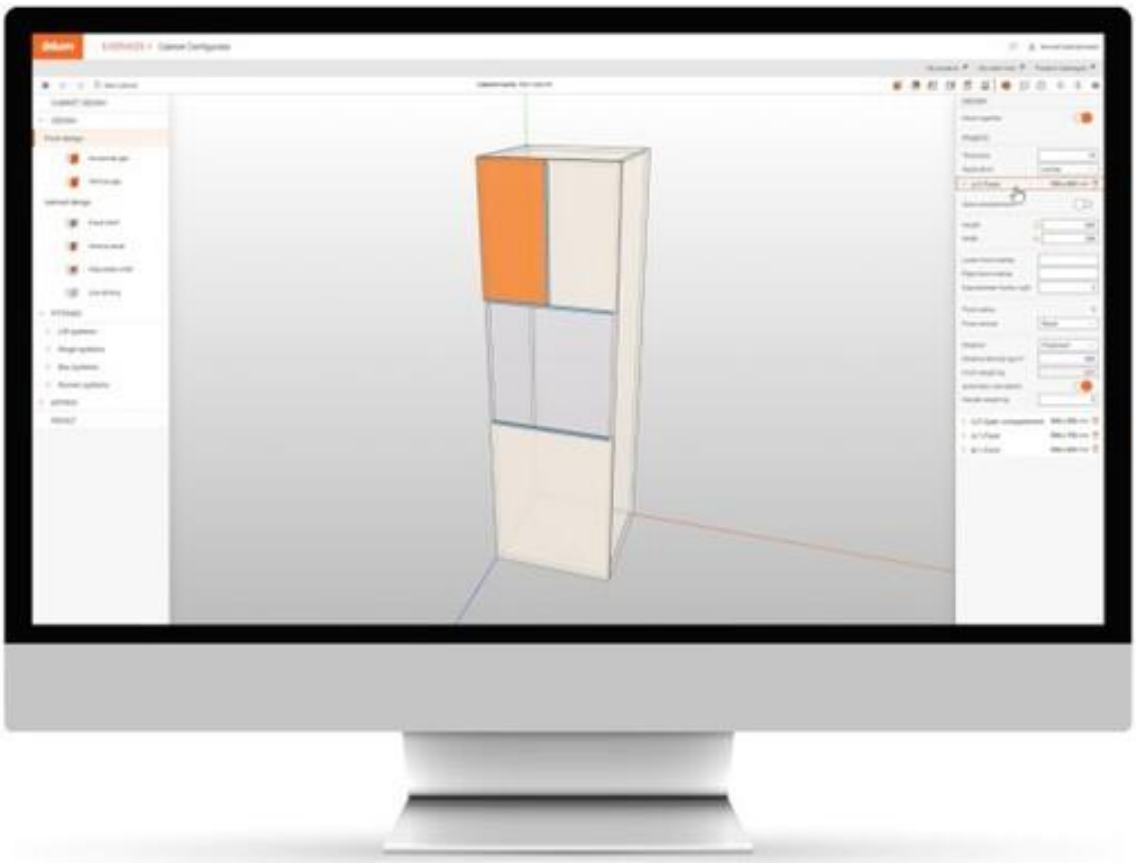


Рисунок 1.3 — Інтерфейс Blum Configurator

Для порівняння можливостей розглянутих застосунків з власною розробкою, було сформовано таблицю 1.1.

Таблиця 1.1 — Порівняння аналогів

Характеристика	IKEA Assemble App	Woodworking Calculator Pro	Blum Configurator	Власна розробка
Персоналізовані рекомендації	0	0	1	1
Експорт специфікацій	1	1	1	1
Розпізнавання комплектуючих та меблів за фото	0	0	0	1
Виявлення відсутніх комплектуючих	1	0	0	1
Підбір необхідних комплектуючих	1	1	1	1
Детальні інструкції зі збірки меблів	1	1	1	1
Сумарний коефіцієнт	4	3	4	6

Таким чином, власна розробка має вищий сумарний бал за IKEA Assemble App та Blum Configurator на 33,3% (100% - 4/6*100% = 33,3%), за Woodworking Calculator Pro на 50% (100% - 3/6*100% = 50%). Отже, актуальною є власна розробка.

1.3 Аналіз методів розв’язання задачі

Розробка мобільного асистента для підбору комплектуючих у виробничих проєктах вимагає обґрунтованого вибору технологічного підходу. Правильно обраний метод розробки забезпечує оптимальне співвідношення функціональності, швидкості впровадження та вартості рішення.

До методів розв’язання задачі з розробки мобільного асистента для підбору комплектуючих належить:

1. Розробка з використанням готових API [19], при якій використовуються готові API інтерфейси для інтеграції існуючих сторонніх сервісів та баз даних для реалізації функціоналу. Цей метод базується на використанні вже розроблених і протестованих інтерфейсів програмування, які надають доступ до каталогів

комплектуючих, систем перевірки сумісності та інших необхідних сервісів.

Готові API включають:

- інформацію про постачальників комплектуючих з актуальними даними про наявність та характеристики компонентів;

- функціонал для перевірки сумісності різних компонентів;

- технічну документацію;

- інтеграцію з існуючими ERP-системами підприємства;

- системи штучного інтелекту для функцій рекомендацій та прогнозування.

Цей метод дозволяє не розробляти увесь функціонал, необхідний для роботи системи підбору комплектуючих самостійно, а використовувати існуючі надійні рішення, що полегшує та прискорює процес розробки, а також забезпечує більш надійну роботу. Він забезпечує постійну актуальність даних про комплектуючі, їхні характеристики та сумісність без додаткових зусиль з боку розробників, тому що розробники API інтерфейсів самостійно оновлюють інформацію, що особливо важливо для динамічних ринків комплектуючих.

2. Повністю власна розробка, в межах якої створюються всі компоненти системи без використання зовнішніх API інтерфейсів [20]. Власна розробка включає: створення баз даних комплектуючих, алгоритмів підбору, систем перевірки сумісності та інших функціональних модулів.

3. Гібридний метод, в якому поєднується використання готових API для певних функцій із власною розробкою для інших [21]. Такий метод дозволяє знайти баланс між швидкістю розробки та рівнем налаштування функціоналу відповідно до потреб користувачів.

Проаналізувавши описані методи, було обрано гібридний метод, який включає в себе інтеграцію API та розробку власного функціоналу, тому що це має такі переваги:

1) забезпечується автоматичне оновлення інформації про комплектуючі без необхідності самостійно оновлювати та моніторити дані;

2) використання готових рішень зменшує ризик помилок та забезпечує стабільну роботу системи;

3) це дозволяє зосередити основні зусилля на розробці унікальної бізнес-логіки та позбавляє потреби розробляти базовий функціонал.

Таким чином, гібридний метод забезпечує баланс між функціональністю та швидкістю впровадження, що робить його найбільш доцільним для розробки мобільного асистента з підбору комплектуючих частин виробничих проєктів.

1.4 Постановка задач дослідження

Провівши аналіз методів розв’язання поставленої задачі, аналіз стану питання розробки мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування, порівняння аналогів було поставлено такі задачі дослідження:

- розробити архітектуру мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування;
- розробити інтерфейс користувача застосунку;
- розробити алгоритми роботи застосунку;
- розробити функціонал застосунку;
- провести тестування розробленого застосунку.

Технічне завдання на розробку наведено в додатку А.

1.5 Висновки

У першому розділі було проведено аналіз питання розробки мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування. Було досліджено існуючі рішення в цьому напрямку, проведено порівняльний аналіз, доведено переваги власної розробки. Проведено аналіз методів розв’язання задачі з розробки мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування, проведено постановку задач дослідження.

2 РОЗРОБКА СТРУКТУРИ ТА АЛГОРИТМІВ ПРОГРАМНОГО ЗАСТОСУНКУ

2.1 Розробка архітектури та структури застосунку

Було обрано принцип багаторівневої архітектури, що відповідає сучасним підходам до розробки мобільних застосунків, таким як MVVM (Model-View-ViewModel) та «чистої архітектури» [22]. Основне завдання такої структури — забезпечити чітке розділення обов’язків між шарами, підвищити модульність системи, спростити тестування та забезпечити її масштабованість. Завдяки поділу на незалежні компоненти кожен із шарів може розвиватися окремо, без ризику порушити загальну логіку роботи програми. Це також полегшує впровадження нових функцій, зміну джерел даних або інтерфейсу користувача без істотних змін в інших частинах системи.

Шар користувацького інтерфейсу відповідає за взаємодію з користувачем, відображення даних та отримання введення. Кожен екран застосунку — CameraScreen, GalleryScreen, DesignAnalysisScreen, AnalysisResultsScreen — виконує окрему функцію в межах користувацького сценарію. Наприклад, екран камери відповідає за знімання зображень для подальшого аналізу, а екран аналізу результатів — за виведення отриманої інформації щодо конструкції або вартості. Цей шар не містить логіки бізнес-рівня. Замість цього він делегує обробку даних до наступного шару — доменного. Цей шар також взаємодіє з компонентом навігації FurnitureHardwareNavigation, який реалізує перемикання між екранами відповідно до дій користувача та дозволяє зберегти інтерфейс у чистому, декларативному стилі.

Доменний рівень відповідає за реалізацію основної логіки застосунку, включно з аналітикою та розрахунками. Він слугує проміжною ланкою між інтерфейсом користувача та шаром доступу до даних. Основна мета цього шару — централізувати бізнес-правила, зробити їх незалежними від способу збереження даних чи зовнішнього представлення. Цей рівень представлено такими класами, як FurnitureAnalyzer — який виконує обробку зображень меблів, аналізуючи їх форму,

габарити чи інші параметри — та HardwarePriceCalculator, що відповідає за обчислення вартості меблевих компонентів на основі заданих характеристик або бази цін. Ці класи взаємодіють з репозиторієм для отримання потрібної інформації з бази даних. Така ізоляція логіки дозволяє не лише підвищити повторне використання коду, а й значно полегшити модульне тестування.

Шар даних реалізує взаємодію з джерелами даних і зберіганням. Він складається з репозиторіїв, DAO-інтерфейсів та сутностей бази даних. Основним класом у цьому шарі є FurnitureHardwareRepository, який виконує роль єдиної точки доступу до даних для доменного шару. Він абстрагує джерело даних, дозволяючи логіці залишатися незалежною від конкретної реалізації. DAO-інтерфейси (FurnitureDesignDao, HardwareDao) описують SQL-запити або доступ до таблиць у базі даних, і компілятор Room автоматично генерує їх реалізацію. Завдяки цьому взаємодія з даними відбувається без прямого використання SQL, забезпечуючи типобезпечність і спрощення логіки доступу до даних. Сутності, такі як FurnitureDesignEntity і HardwareComponentEntity, представляють структуру таблиць бази даних. Вони описують поля, індексацію, первинні ключі, зв’язки між таблицями. Саме ці сутності Room зберігає у SQLite.

Базовий рівень — це Room-база даних, представлена класом HardwareDatabase. Цей компонент реалізує інтерфейс до локального сховища SQLite через ORM-обгортку Room. HardwareDatabase інкапсулює всі DAO-інтерфейси та надає точку ініціалізації бази даних. Room забезпечує такі переваги, як перевірка запитів під час компіляції, автоматичне створення таблиць, зручне перетворення результатів. Завдяки цьому зменшується ризик помилок і спрощується розробка. Крім того, Room дозволяє легко впроваджувати міграції при оновленні схеми таблиць, що є важливим фактором при підтримці довготривалого проєкту.

Навігація між екранами реалізується через окремий модуль FurnitureHardwareNavigation, який забезпечує керування маршрутами у застосунку. Це дозволяє централізовано керувати переходами, підтримувати чистоту UI-шару та реалізувати реактивну навігацію у відповідь на дії користувача або зміни стану.

Такий підхід спрощує масштабування застосунку та додавання нових екранів без зміни існуючої логіки. Окрім навігаційних маршрутів, цей модуль може містити логіку обробки переходів залежно від стану даних або результатів обчислень, що також сприяє гнучкості архітектури.

Архітектура застосунку демонструє розділення логіки між шарами: інтерфейс користувача, бізнес-логіка, доступ до даних та інфраструктура. Такий підхід дозволяє адаптувати та масштабувати застосунок, проводити тестування окремих модулів і забезпечує чисту, підтримувану кодову базу. Завдяки цьому спрощується командна розробка: розробники можуть працювати паралельно над різними шарами, не створюючи конфліктів. Також це дозволяє легко впроваджувати зовнішні сервіси або API, замінюючи лише окремі компоненти без порушення загальної структури.

Архітектура застосунку наведено на рисунку 2.1.

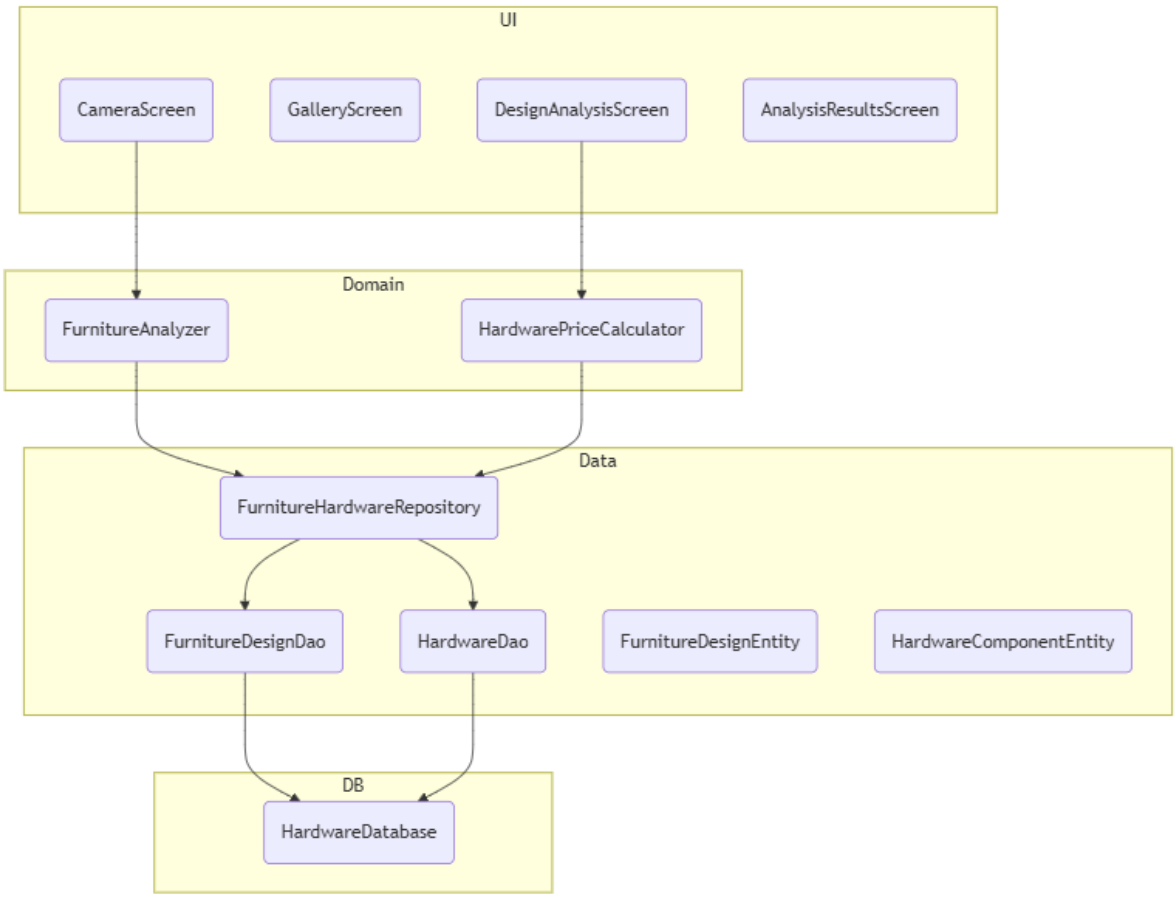


Рисунок 2.1 — Архітектура застосунку для підбору комплектуючих частин виробничих проєктів

Взаємодія з ШІ Claude та YouTube здійснюється з дотриманням принципів асинхронного програмування, що забезпечує оптимальну продуктивність роботи застосунку, виключає можливі «заморозки» користувацького інтерфейсу [23]. Допоміжний функціонал для роботи з камерою та галереєю зображень виокремлена в пакеті utils.

Така архітектура застосунку забезпечує баланс між складністю організації та функціональними можливостями застосунку, дозволяє реалізувати всі ключові функції застосунку, при цьому залишається компактною та зрозумілою, що сприяє швидкому впровадженню та подальшій підтримці застосунку.

2.2 Розробка моделі системи

Було розроблено модель системи у вигляді діаграми компонентів, яка відображає основні складові застосунку для підбору комплектуючих виробничих проєктів, та технології, що відповідають за їхню роботу, залежності між ними, демонструючи чітке розділення відповідальності згідно з принципами компонентно-орієнтованої архітектури. Застосунок складається з набору взаємопов'язаних модулів, кожен з яких виконує специфічну функцію в загальному контексті застосунку.

Інтерфейс користувача застосунку реалізовано з використанням бібліотеки Jetpack Compose, що забезпечує декларативний підхід до побудови інтерфейсу користувача, тобто інтерфейс прописується як набір різних функцій, що викликають одна іншу залежно від бізнес-логіки застосунку.

Компонент HomeScreen відображає головний екран застосунку та має залежність від Jetpack Compose для формування елементів інтерфейсу. AnalysisScreen є центральним елементом взаємодії з користувачем, забезпечуючи відображення результатів аналізу зображень та встановлюючи залежності з компонентами Claude API та Claude Vision API для отримання та обробки даних.

Екран ProjectDetails відповідає за презентацію детальної інформації про DIY-проєкт та інтегрується з YouTube API для відображення релевантних відеоматеріалів. CameraScreen реалізує функціональність отримання фотографій за

допомогою камери пристрою та має залежність від бібліотеки CameraX.

Підсистема DataStorage забезпечує зберігання та управління даними застосунку, використовуючи Room Database як основний компонент для персистентного зберігання інформації. Цей компонент інкапсулює логіку доступу до даних та забезпечує їх цілісність.

Room Database надає об'єктно-реляційне відображення (ORM) для взаємодії з SQLite, що дозволяє зберігати структуровані дані проєктів, інструментів, матеріалів та результатів аналізу зображень.

Застосунок інтегрується з низкою зовнішніх API для забезпечення ключової функціональності. Claude API та Claude Vision API є важливими компонентами, що забезпечують аналіз зображень, розпізнавання інструментів та матеріалів, фрпмування рекомендацій проєктів та інструкцій. YouTube API надає можливість пошуку та інтеграції відеоінструкцій до DIY-проєктів. Для здійснення мережевих запитів до цих сервісів система використовує бібліотеку OkHttp, що має залежність від компонента API Client.

«Комунікаційний» шар застосунку представлений компонентом API Client, який функціонує як централізований механізм для взаємодії з зовнішніми сервісами, який забезпечує уніфікований інтерфейс для здійснення HTTP-запитів та обробки відповідей, інкапсулюючи логіку авторизації, кешування та обробки помилок. Для здійснення мережевих операцій API Client має залежність від бібліотеки OkHttp, що забезпечує надійну взаємодію з віддаленими сервісами.

Модель системи відображає залежності між компонентами, демонструючи напрями потоків даних та контролю. Модулі користувацького інтерфейсу залежать від відповідних бібліотек та API-сервісів, при цьому взаємодія між компонентами здійснюється через чітко визначені інтерфейси. Така організація забезпечує високу ступінь модульності, що спрощує тестування, підтримку та подальший розвиток системи.

Модель застосунку наведена на рисунку 2.2.

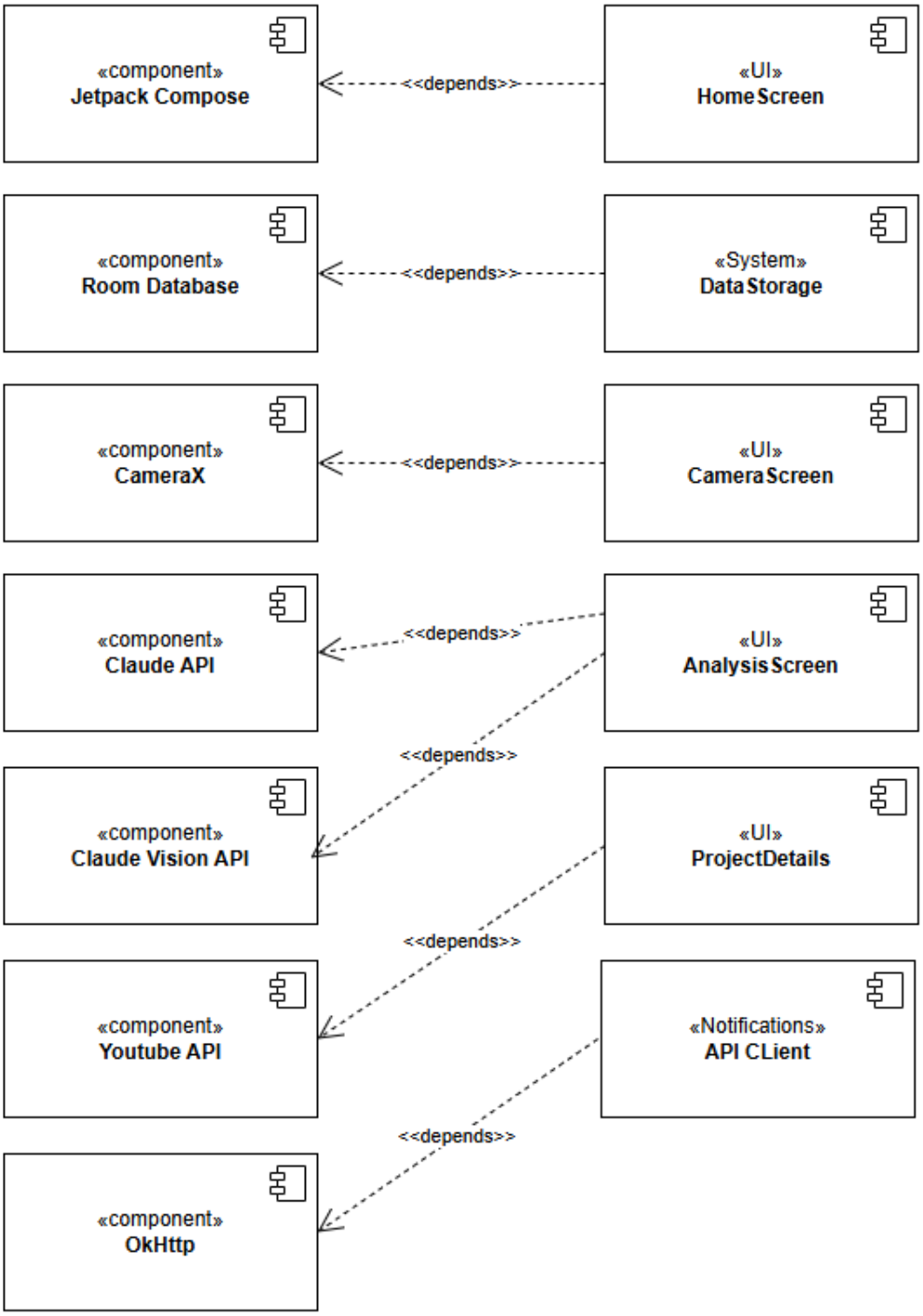


Рисунок 2.2 — Модель застосунку для підбору комплектуючих частин виробничих проєктів

2.3 Розробка методу оптимізації вибору меблевих комплектуючих на зображенні

Метод оптимізації вибору меблевих комплектуючих на зображенні використовує комп'ютерний зір та штучний інтелект для створення

персоналізованих інструкцій проєктів на основі інструментів і матеріалів, доступних користувачу. Цей метод дозволяє максимально спростити процес виконання проєктів, адаптуючи інструкції під конкретні можливості кожного користувача та усуваючи необхідність пошуку альтернативних рішень самостійно.

Робота методу динамічної генерації інструкцій на основі наявних інструментів складається з таких етапів:

1. Збір даних про наявні інструменти та матеріали. Система отримує фото наявних інструментів та матеріалів одним з таких способів:

- Користувач фотографує свої інструменти та матеріали за допомогою камери пристрою.
- Користувач обирає зображення з галереї.

2. Після отримання зображення система використовує API Claude Vision для аналізу та ідентифікації об'єктів на фотографії:

- Зображення конвертується у формат Base64.
- Система формує запит до Claude Vision API, передаючи зображення з інструкцією розпізнати всі інструменти та матеріали.
- API обробляє зображення, використовуючи нейронні мережі для ідентифікації об'єктів.
- Система отримує результат розпізнавання у форматі JSON, який містить два масиви: tools (інструменти) та materials (матеріали).

3. Після вибору конкретного проєкту система проводить процес формування детальних інструкцій шляхом створення запиту до API Claude, який включає наступну інформацію:

- Назва проєкту
- Повний список доступних інструментів
- Повний список доступних матеріалів

4. Формування персоналізованих інструкцій, які враховують специфіку наявних у користувача інструментів та матеріалів:

- Система аналізує запит та генерує інструкції, які адаптовані до конкретних інструментів.

- Якщо певний крок можна виконати різними інструментами, система пріоритизує ті, які є у користувача.
- Якщо для певного кроку потрібні специфічні інструменти, яких немає у користувача, система пропонує альтернативні підходи з використанням наявних інструментів.

5. Отримані інструкції обробляються системою та відображаються користувачу. Текст інструкцій парситься з JSON-відповіді, інструкції зберігаються в об'єкті DIYProject, відображаються користувачу в інтерфейсі застосунку у вигляді пронумерованих кроків, зберігаються в локальній базі даних для подальшого використання офлайн.

Таким чином, запропонований метод підвищує можливості користувачів у комплектуванні фурнітурою меблів з різним набором інструментів та матеріалів, дозволяючи їм отримувати удосконалені інструкції, що враховують їхні конкретні можливості. Інтеграція технологій комп'ютерного зору та штучного інтелекту забезпечує високу точність розпізнавання інструментів і створення релевантних інструкцій, що робить систему помічником для любителів саморобних проєктів різного рівня досвіду.

2.4 Розробка методу формування інструкцій для комплектування фурнітурою меблів

Метод формування інструкцій для комплектування фурнітурою меблів являє собою підхід, який підбирає необхідні інструменти та матеріали для виробництва меблів. Система використовує штучний інтелект для оцінки проєкту, що буде реалізовуватися, його складності та визначення додаткових потреб, що дозволяє користувачам використовувати вже наявні ресурси та мінімізувати додаткові витрати.

Робота методу формування інструкцій для комплектування фурнітурою меблів складається з таких етапів:

1. Аналіз меблів:

1.1 Система отримує результати розпізнавання з модуля ImageAnalyzer у

вигляді об'єкта ProjectRecognitionResult.

1.2 Результати включають інформацію про меблі, що необхідно виготовити:

- назву проєкту;
- рівень складності;
- приблизний час виконання;
- короткий опис;
- необхідні інструменти;
- необхідні матеріали;

2. Формування специфікації проєкту, що включає в себе:

2.1 компоненти фурнітури;

2.2 кількість кожної з компонент для виробництва необхідних меблів;

2.3 інформацію про розміщення кожної компоненти;

2.4 рекомендації по монтажу кожної компоненти.

3. По запиту користувача, формуються детальні інструкції щодо реалізації проєкту з виробництва відсканованих меблів у вигляді нумерованого списку. Система обробляє отримані рекомендації шляхом парсингу JSON-відповіді та перетворення її у список об'єктів DIYProject.

4. Збереження специфікації картки з основною інформацією. Кожна картка містить:

- назву проєкту;
- рівень складності;
- приблизний час виконання;
- короткий опис;
- список відсутніх предметів (якщо такі є).

Користувач може вибрати проєкт для перегляду детальної інформації та інструкцій.

Цей метод розширює можливості системи, пропонуючи користувачам персоналізовані рекомендації проєктів. Він не лише допомагає використовувати

доступні інструменти та матеріали, але й сприяє економії коштів, часу та зусиль користувачів, надаючи їм точну інформацію про відсутні предмети та ступінь готовності до виконання кожного проєкту.

2.5 Розробка алгоритмів роботи системи

Алгоритм розпізнавання інструментів і матеріалів призначений для аналізу фотографій та ідентифікації об'єктів, які можуть бути використані в DIY-проєктах.

Блок-схема алгоритму розпізнавання інструментів і матеріалів наведена на рисунку 2.3.

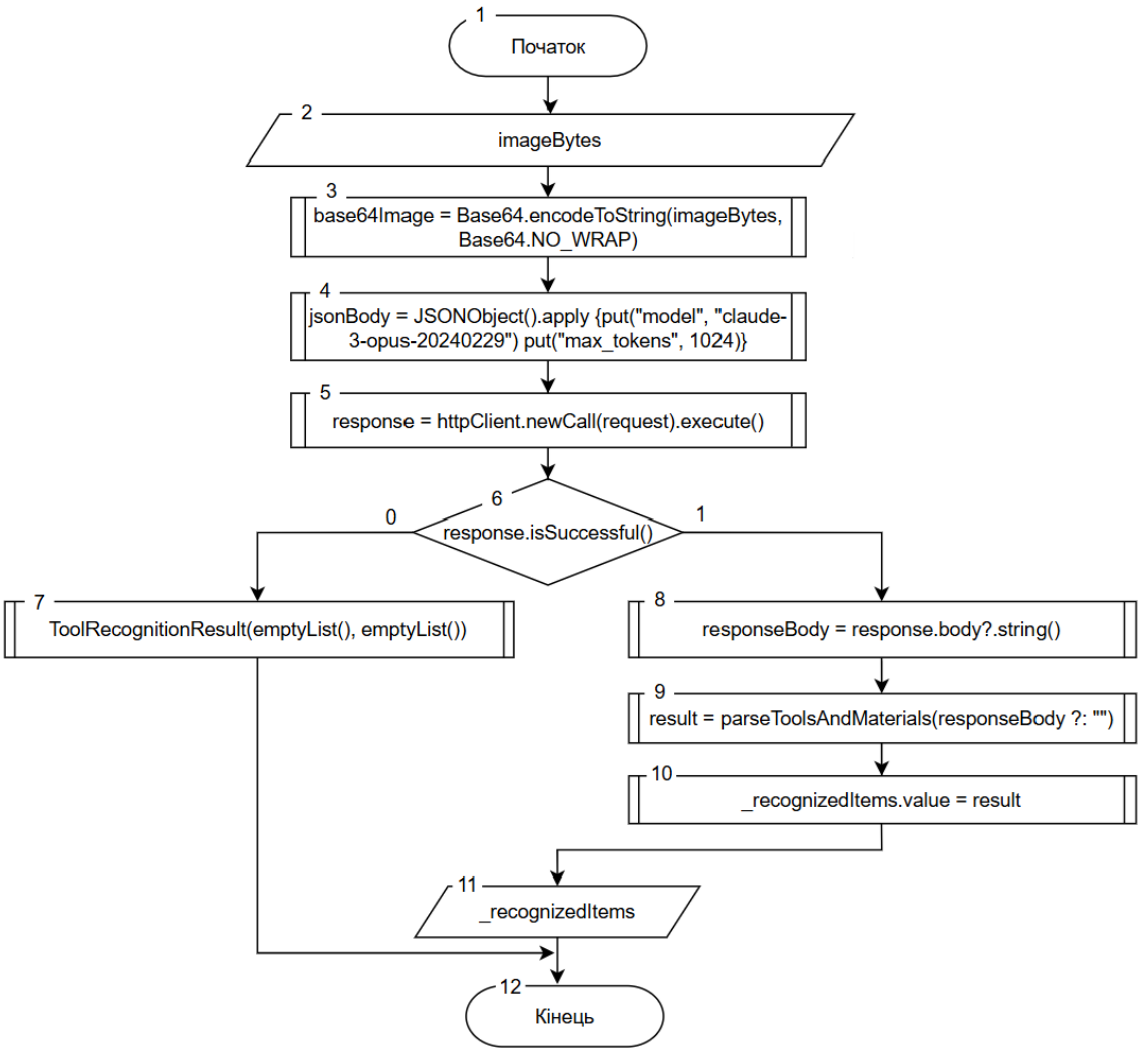


Рисунок 2.3 — Блок-схема алгоритму розпізнавання інструментів і матеріалів

Початковим етапом є отримання зображення з камери пристрою або галереї користувача. Далі зображення конвертується у формат Base64 для передачі через

API. Після формування структурованого запиту, що включає зображення та інструкції для аналізу, він відправляється до Claude Vision API. У випадку успішної відповіді, система здійснює парсинг отриманого JSON і виділяє окремі списки розпізнаних інструментів та матеріалів. Ці дані інкапсулюються в єдиному об'єкті ToolRecognitionResult і передаються для подальшої обробки. Якщо на будь-якому етапі виникає помилка, алгоритм повертає порожній результат, забезпечуючи стабільну роботу застосунку.

Алгоритм генерації рекомендацій проєктів використовує результати розпізнавання для створення списку потенційних DIY-проєктів.

Блок-схема цього алгоритму наведена на рисунку 2.4.

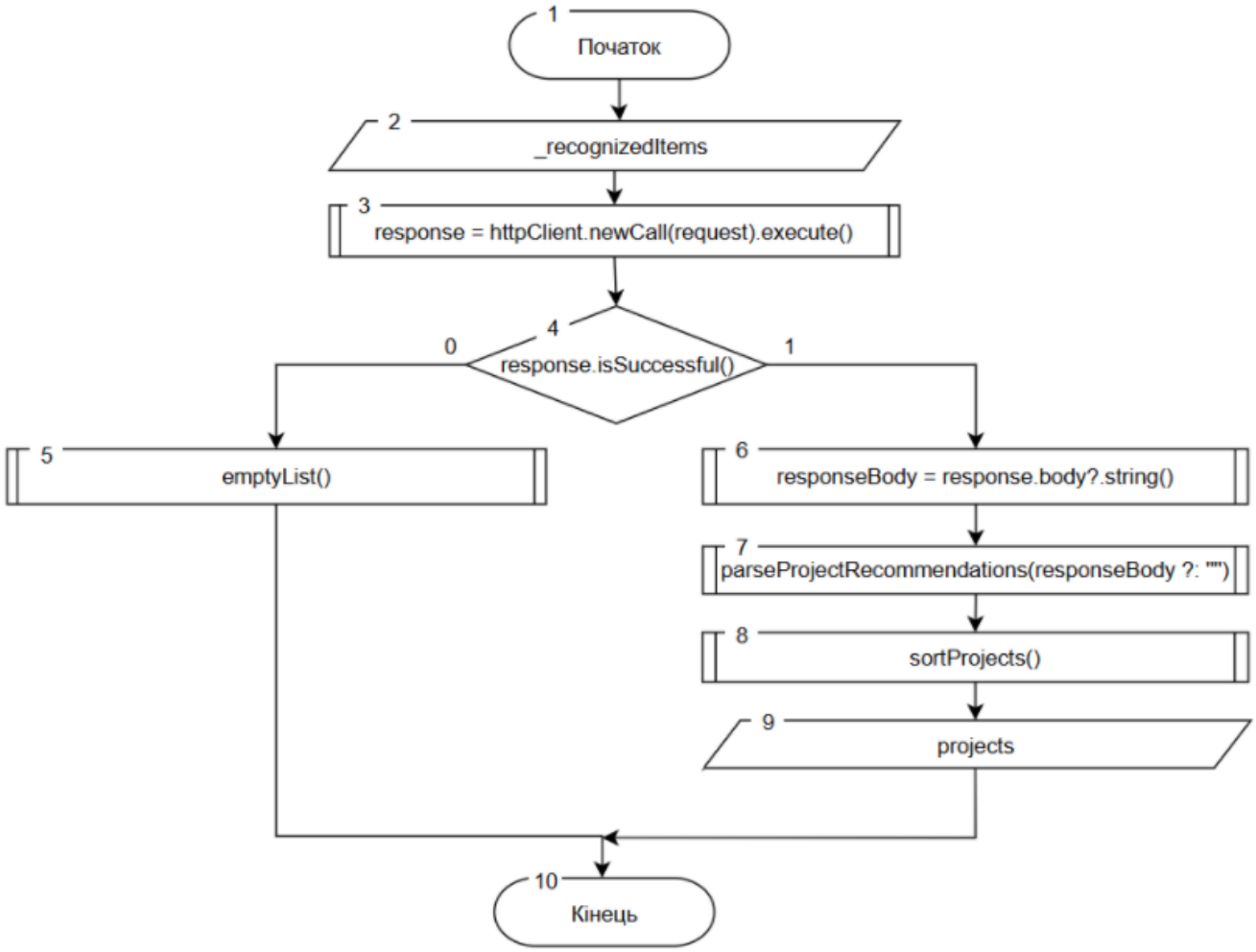


Рисунок 2.4 — Блок-схема алгоритму генерації рекомендацій проєктів

Отримавши списки доступних інструментів та матеріалів, система формує спеціалізований запит до Claude API, який включає всю необхідну інформацію про

наявні ресурси. Отримана відповідь проходить обробку, під час якої для кожного рекомендованого проєкту створюється об'єкт DIYProject з деталізованою інформацією: назва, рівень складності, орієнтовний час виконання, опис, необхідні інструменти і матеріали, а також список відсутніх ресурсів. Сформований список проєктів сортується за релевантністю на основі кількості доступних інструментів і матеріалів, що дозволяє користувачу спочатку бачити проєкти, які він може виконати з мінімальними додатковими придбаннями.

Алгоритм генерації інструкцій забезпечує створення детальних покрокових вказівок для реалізації обраного DIY-проєкту.

Блок-схема алгоритму генерації інструкцій наведена на рисунку 2.5.

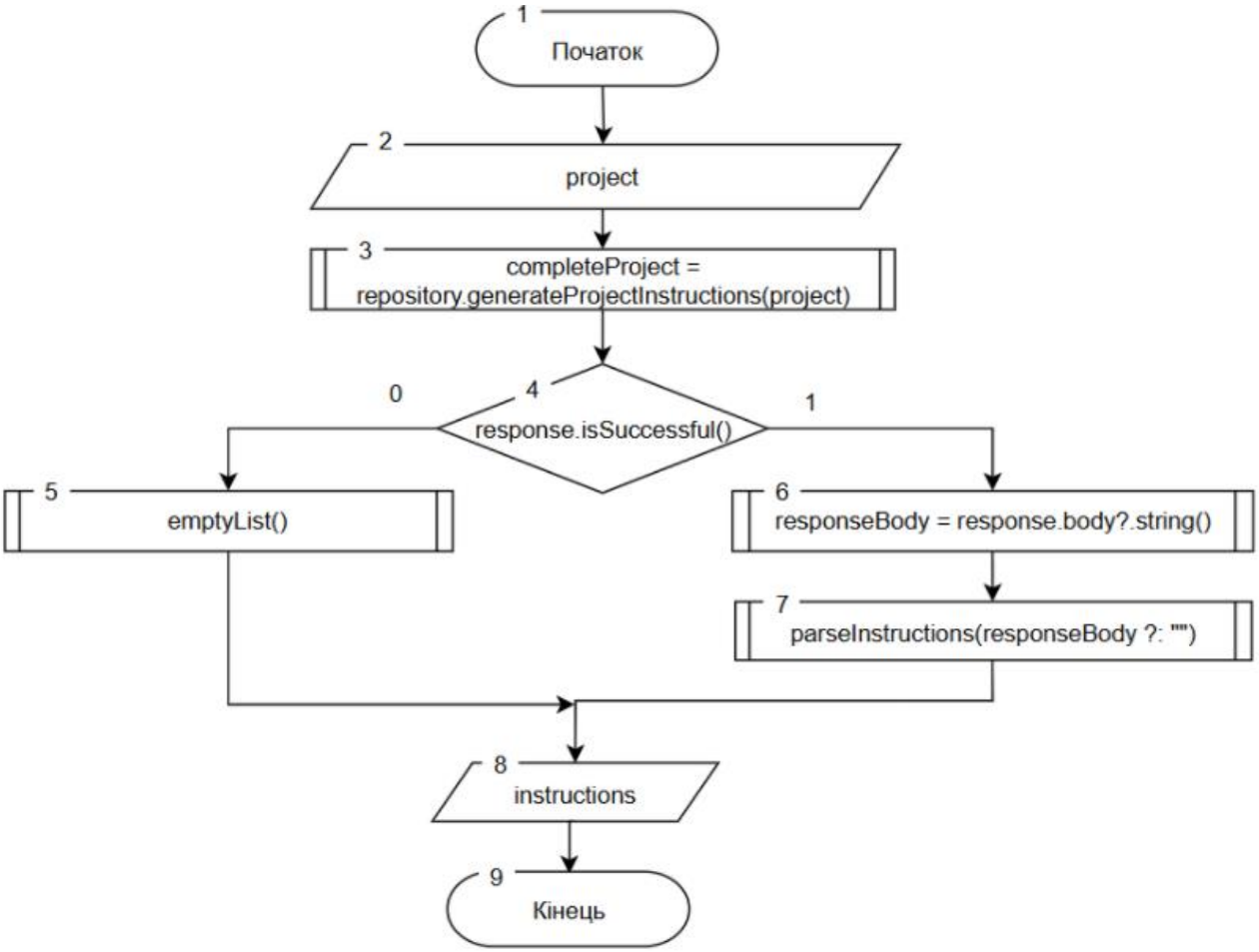


Рисунок 2.5 — Блок-схема алгоритму генерації інструкцій

На основі даних проєкту (назва, необхідні інструменти та матеріали) формується запит до Claude API з детальним контекстом. Отримана відповідь аналізується для вилучення структурованих інструкцій, які потім форматуються у

вигляді пронумерованого списку кроків. Кожен крок обробляється для забезпечення чіткості та зрозумілості, з акцентом на доступні користувачу інструменти та матеріали. Результуючий список інструкцій інтегрується з об'єктом проєкту, забезпечуючи комплексний підхід до реалізації DIY-завдання.

Алгоритм пошуку відеоінструкцій призначений для знаходження релевантних відеоматеріалів, що доповнюють текстові інструкції.

Блок-схема алгоритму пошуку відеоінструкцій наведена на рисунку 2.6.

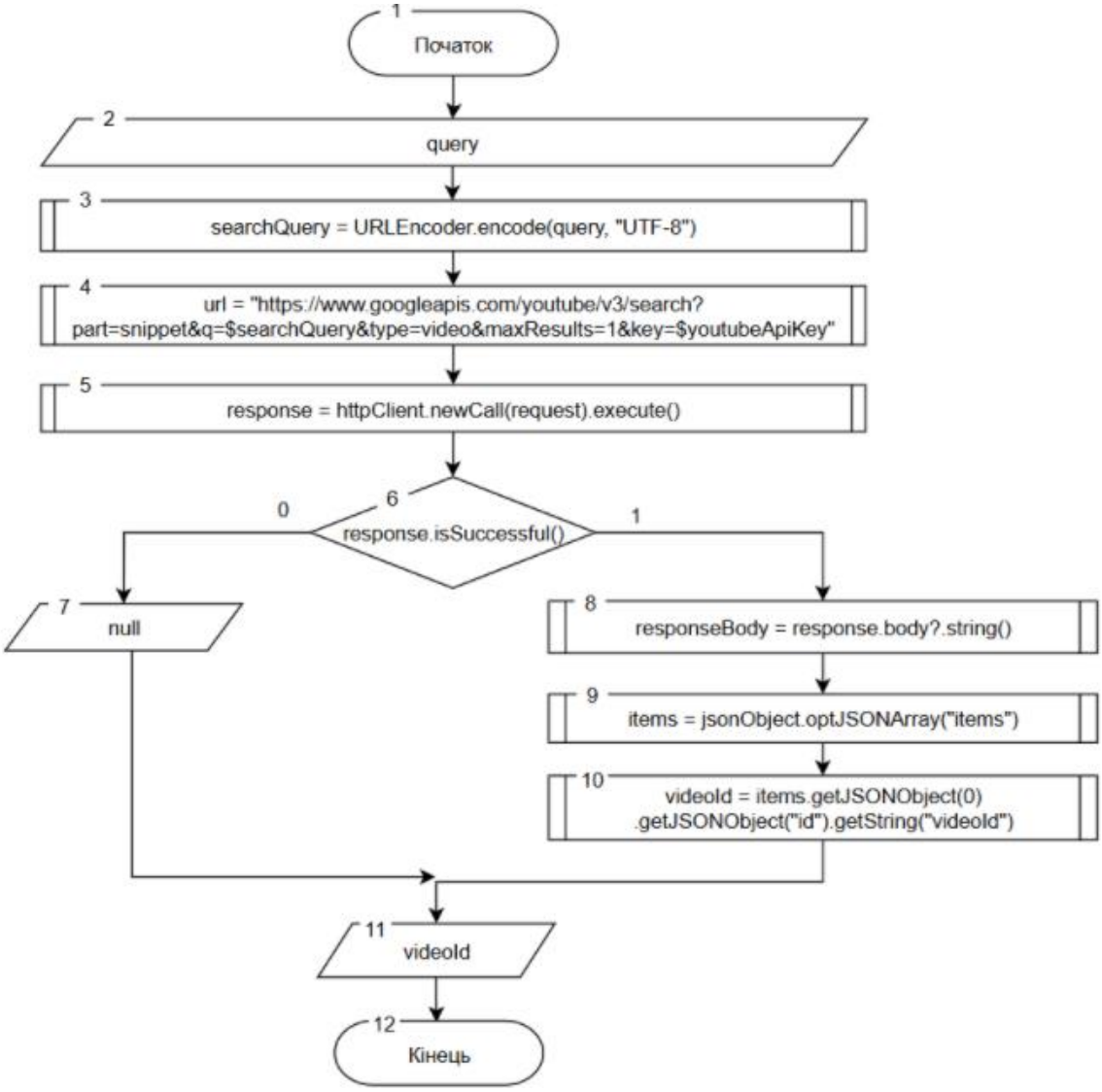


Рисунок 2.6 — Блок-схема алгоритму пошуку відеоінструкцій

На основі назви проєкту формується пошуковий запит з додаванням ключових слів "DIY" і "tutorial" для підвищення релевантності. Після URL-кодування запит відправляється до YouTube API з відповідними параметрами для

обмеження кількості результатів. Із отриманої відповіді система вилучає ідентифікатор першого (найбільш релевантного) відео, який потім інтегрується з даними проєкту. Це дозволяє користувачу отримати доступ до візуальних інструкцій безпосередньо з інтерфейсу застосунку.

2.6 Висновки

У цьому розділі було розроблено архітектуру та структуру мобільного асистента для підбору комплектуючих виробничих проєктів, обрано патерн MVVM для розробки архітектури застосунку, наведено структуру папок та класів, що використовуються для роботи застосунку. Розроблено модель системи у вигляді діаграми компонентів, описано компоненти, що реалізуються та використовуються у мобільному асистенті. Розроблено метод динамічної генерації інструкцій на основі наявних інструментів та метод зворотної ідентифікації проєктів за наявними інструментами, побудовано алгоритми роботи застосунку.

3 РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ АВТОМАТИЗАЦІЇ КОМПЛЕКТУВАННЯ ФУРНІТУРОЮ МЕБЛІВ НА ЕТАПІ ПРОЕКТУВАННЯ

3.1 Варіантний аналіз та вибір мови програмування розробки

Kotlin [24] — сучасна статично типізована мова програмування, розроблена компанією JetBrains. Вона отримала офіційну підтримку від Google як основна мова для розробки Android-застосунків. Kotlin повністю сумісний з Java, що дозволяє розробникам поступово переходити з однієї мови на іншу в межах одного проєкту. Завдяки лаконічному синтаксису, функціональному програмуванню та можливостям розширення, Kotlin дозволяє писати більш стислий та читабельний код порівняно з Java. Мова також має вбудовану підтримку нульової безпеки, що знижує ризик виникнення помилок NullPointerException, які часто зустрічаються в Java-проєктах.

Java [25] — об'єктно-орієнтована мова програмування, яка традиційно використовується для розробки Android-застосунків. Мова має довгу історію та зрілу екосистему з величезною кількістю бібліотек та інструментів. Java характеризується статичною типізацією, автоматичним керуванням пам'яттю та крос-платформністю завдяки віртуальній машині (JVM).

Для Android-розробки Java залишається релевантною завдяки значній кількості готових розробок, документації та значній спільноті розробників. Проте порівняно з Kotlin, Java має більш громіздкий синтаксис, вимагає більше шаблонного коду та не має вбудованої null безпеки, що може призводити до частіших помилок у роботі застосунку.

Dart [26] — мова програмування, розроблена Google, яка використовується переважно з фреймворком Flutter для створення кросплатформених мобільних застосунків. Flutter дозволяє розробляти застосунки одночасно для iOS та Android з використанням єдиної кодової бази. Dart має простий та інтуїтивно зрозумілий синтаксис, подібний до Java та C#, а також підтримує як об'єктно-орієнтоване, так і функціональне програмування. Flutter надає багатий набір віджетів для створення

красивих інтерфейсів та забезпечує високу продуктивність завдяки компіляції безпосередньо в нативний код. Для проєкту з підбору комплектуючих Dart з Flutter може бути надійним рішенням завдяки можливості швидкої розробки інтерфейсів для перегляду каталогів і рекомендацій, проте може мати деякі обмеження при роботі з нативними API для аналізу зображень.

Порівняння описаних мов програмування наведено у таблиці 3.1.

Таблиця 3.1 - Порівняльна таблиця мов програмування

Характеристика	Kotlin	Java	Flutter/Dart
Кросплатформеність	1	0	1
Робота з API для аналізу зображень	1	1	0
Доступ до сенсорів пристрою	1	1	1
Інтеграція з базами даних	1	1	1
Безпека типізації даних	1	1	1
Інтеграція з backend системами	1	1	1
Null-безпека	1	0	0
Розмір застосунку	1	1	0
Сумарний коефіцієнт	8	6	5

Таким чином, Kotlin є найкращою мовою для розробки мобільного асистента для підбору комплектуючих виробничих проєктів.

3.2 Розробка бази даних

Для збереження інформації користувач, проектів, інструментів, матеріалів, інструкцій, які він використовує для виконання виробничих проектів, рекомендацій по виконанню проєктів, посилань з відеоінструкцій, необхідна база даних, схема якої наведена на рисунку 3.1.

Рисунок 3.1 — Схема бази даних

Таблиця users використовується для управління обліковими записами користувачів та їхньою автентифікаційною інформацією. Поля таблиці users наведено у таблиці 3.2.

Таблиця 3.2 — Поля таблиці users

Поле	Тип даних	Опис
user_id	UUID	Первинний ключ, унікальний ідентифікатор користувача
email	String	Електронна адреса користувача (унікальна)
username	String	Ім'я користувача для відображення
created_at	Long	Дата та час створення облікового запису (мілісекунди з початку епохи)
updated_at	Long	Дата та час останнього оновлення облікового запису (мілісекунди з початку епохи)

Таблиця projects використовується для зберігання інформації про DIY-

проєкти, включаючи їхні деталі та метадані.

Поля таблиці projects наведено у таблиці 3.3.

Таблиця 3.3 — Поля таблиці projects

Поле	Тип даних	Опис
project_id	UUID	Первинний ключ, унікальний ідентифікатор проєкту
title	String	Назва DIY-проєкту
difficulty	String	Рівень складності проєкту (наприклад, "Простий", "Середній", "Складний")
time_estimate	String	Приблизний час, необхідний для виконання проєкту
description	String	Детальний опис проєкту
created_at	Long	Дата та час створення проєкту (мілісекунди з початку епохи)
updated_at	Long	Дата та час останнього оновлення проєкту (мілісекунди з початку епохи)
user_id	UUID	Зовнішній ключ, що посилається на створювача проєкту

Таблиця tools використовується для підтримки каталогу інструментів, які можуть використовуватися в DIY-проєктах.

Поля таблиці tools наведено у таблиці 3.4.

Таблиця 3.4 — Поля таблиці tools

Поле	Тип даних	Опис
tool_id	UUID	Первинний ключ, унікальний ідентифікатор інструменту
name	String	Назва інструменту
description	String	Детальний опис інструменту та його використання
created_at	Long	Дата та час додавання інструменту до бази даних (мілісекунди з початку епохи)

Таблиця materials використовується для підтримки каталогу матеріалів, які можуть використовуватися в DIY-проєктах.

Поля таблиці materials наведено у таблиці 3.5.

Таблиця 3.5 — Поля таблиці materials

Поле	Тип даних	Опис
material_id	UUID	Первинний ключ, унікальний ідентифікатор матеріалу
name	String	Назва матеріалу
description	String	Детальний опис матеріалу та його властивостей
created_at	Long	Дата та час додавання матеріалу до бази даних (мілісекунди з початку епохи)

Таблиця project_instructions використовується для зберігання покрокових інструкцій для виконання кожного DIY-проєкту.

Поля таблиці project_instructions наведено у таблиці 3.6.

Таблиця 3.6 — Поля таблиці project_instructions

Поле	Тип даних	Опис
instruction_id	UUID	Первинний ключ, унікальний ідентифікатор інструкції
project_id	UUID	Зовнішній ключ, що посилається на проєкт, до якого належить ця інструкція
step_number	Int	Порядковий номер кроку в послідовності інструкцій
instruction_text	String	Детальний текст із описом виконання цього кроку
created_at	Long	Дата та час створення інструкції

Таблиця project_tools - сполучна таблиця, яка відображає зв'язок між проєктами та необхідними для них інструментами.

Поля таблиці project_tools наведено у таблиці 3.7.

Таблиця 3.7 — Поля таблиці project_tools

Поле	Тип даних	Опис
project_id	UUID	Зовнішній ключ, що посилається на проєкт
tool_id	UUID	Зовнішній ключ, що посилається на інструмент

Продовження таблиці 3.7

is_required	Boolean	Вказує, чи є інструмент необхідним для проєкту
is_available	Boolean	Вказує, чи є цей інструмент доступним користувачу
created_at	Long	Дата та час запису цього зв'язку (мілісекунди з початку епохи)

Комбінація project_id та tool_id служить складеним первинним ключем.

Таблиця project_materials - сполучна таблиця, яка відображає зв'язок між проєктами та необхідними для них матеріалами.

Поля таблиці project_materials наведено у таблиці 3.8.

Таблиця 3.8 — Поля таблиці project_materials

Поле	Тип даних	Опис
project_id	UUID	Зовнішній ключ, що посилається на проєкт
material_id	UUID	Зовнішній ключ, що посилається на матеріал
quantity	String	Кількість матеріалу, необхідна для проєкту
is_required	Boolean	Вказує, чи є матеріал необхідним для проєкту
is_available	Boolean	Вказує, чи є цей матеріал доступним користувачу
created_at	Long	Дата та час запису цього зв'язку (мілісекунди з початку епохи)

Комбінація project_id та material_id служить складеним первинним ключем.

Таблиця youtube_videos використовується для зберігання посилань на відео з YouTube, які надають візуальні інструкції для проєктів.

Поля таблиці youtube_videos наведено у таблиці 3.9.

Таблиця 3.9 — Поля таблиці youtube_videos

Поле	Тип даних	Опис
video_id	UUID	Первинний ключ, унікальний ідентифікатор посилання на відео
project_id	UUID	Зовнішній ключ, що посилається на проєкт, до якого відноситься відео
youtube_video_id	String	Унікальний ідентифікатор відео на YouTube
title	String	Назва відео на YouTube

Поле	Тип даних	Опис
description	String	Опис вмісту відео
created_at	Long	Дата та час додавання посилання на відео (мілісекунди з початку епохи)

Таблиця images використовується для зберігання посилань на зображення, пов'язані з проєктами, включаючи фотографії, надіслані користувачами.

Поля таблиці images наведено у таблиці 3.10.

Таблиця 3.10 — Поля таблиці images

Поле	Тип даних	Опис
image_id	UUID	Первинний ключ, унікальний ідентифікатор зображення
project_id	UUID	Зовнішній ключ, що посилається на пов'язаний проєкт
image_url	String	URL або шлях до файлу зображення
image_type	String	Тип зображення (наприклад, "Фото інструментів", "Фото результату", "Ілюстрація кроку")
created_at	Long	Дата та час додавання зображення (мілісекунди з початку епохи)

Таблиця user_saved_projects використовується для відстеження проєктів, які користувачі зберегли для подальшого використання.

Поля таблиці user_saved_projects наведено у таблиці 3.11.

Таблиця 3.11 — Поля таблиці user_saved_projects

Поле	Тип даних	Опис
user_id	UUID	Зовнішній ключ, що посилається на користувача
project_id	UUID	Зовнішній ключ, що посилається на збережений проєкт
saved_at	Long	Дата та час, коли користувач зберіг цей проєкт

Комбінація user_id та project_id служить складеним первинним ключем.

Таблиця analysis_results використовується для зберігання результатів розпізнавання інструментів та матеріалів з фотографій, надісланих користувачами.

Поля таблиці analysis_results наведено у таблиці 3.12.

Таблиця 3.12 — Поля таблиці analysis_results

Поле	Тип даних	Опис
analysis_id	UUID	Первинний ключ, унікальний ідентифікатор операції аналізу
user_id	UUID	Зовнішній ключ, що посилається на користувача, який надіслав зображення
image_id	UUID	Зовнішній ключ, що посилається на проаналізоване зображення
analysis_data	String	Структуровані дані у форматі JSON, що містять розпізнані інструменти та матеріали
created_at	Long	Дата та час проведення аналізу (мілісекунди з початку епохи)

Таблиця project_recommendations використовується для відстеження рекомендацій проєктів, згенерованих на основі доступних користувачу інструментів та матеріалів.

Поля таблиці project_recommendations наведено у таблиці 3.13.

Таблиця 3.13 — Поля таблиці project_recommendations

Поле	Тип даних	Опис
recommendation_id	UUID	Первинний ключ, унікальний ідентифікатор кожної рекомендації
analysis_id	UUID	Зовнішній ключ, що посилається на операцію аналізу
project_id	UUID	Зовнішній ключ, що посилається на рекомендований проєкт
relevance_score	Float	Числовий показник релевантності цього проєкту до доступних користувачу предметів
missing_items_count	Int	Кількість інструментів та матеріалів, яких не вистачає користувачу для цього проєкту
created_at	Long	Дата та час створення рекомендації

Ця база даних забезпечує роботу основної функціональності застосунку. Структура дозволяє зручно отримувати рекомендації щодо проєктів,

персоналізовані інструкції та відстежувати взаємодію користувачів із системою.

3.3 Розробка інтерфейсу користувача

Інтерфейс користувача дозволяє легко та зручно взаємодіяти із системою, проводити аналіз інструментів та матеріалів за фото, підбирати найкращі виробничі проєкти для реалізації, отримувати інструкції та список необхідних інструментів для реалізації проєктів.

Розроблено головний екран застосунку, який містить загальну інформацію про застосунок, та кілька кнопок для переходу до основних модулів застосунку. Структура інтерфейсу головного екрану застосунку наведена на рисунку 3.2.

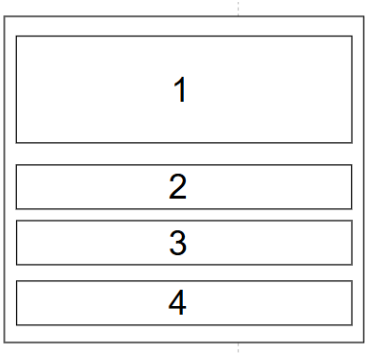


Рисунок 3.2 — Структурна схема інтерфейсу головного екрану застосунку

Складові елементи головного екрану застосунку:

1. Загальний опис застосунку.
2. Кнопка «Зробити фото інструментів».
3. Кнопка «Вибрати з галереї».
4. Кнопка «Збережені DIY-проєкти».

Рекомендації щодо реалізації проєктів містять список запчастин та матеріалів, що необхідні для виконання проєкту, їхню кількість, та інструкції по кожній запчастині: де розмістити, як і для чого, а також альтернативні варіанти. Структура інтерфейсу рекомендацій щодо матеріалів наведена на рисунку 3.3.

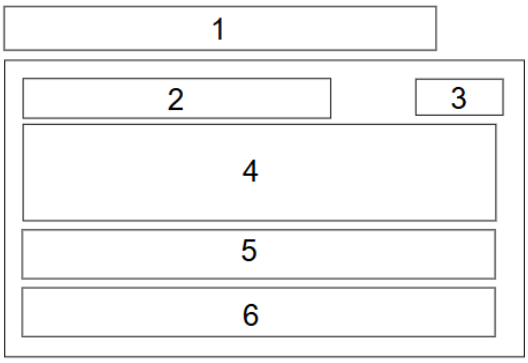


Рисунок 3.3 — Структурна схема елемента інтерфейсу для рекомендацій

Складові елемента інтерфейсу для рекомендацій:

1. Загальна назва рекомендованих матеріалів.
2. Назва конкретного матеріалу.
3. Потрібна кількість матеріалу.
4. Опис розташування запчастини/матеріалу.
5. Інструкції до встановлення.
6. Альтернативи.

Екран аналізу проєктів містить загальну інформацію про проєкти, які можна реалізувати, використовуючи інструменти та матеріали, що наявні на фото, зробленому користувачем. Він містить назву проєктів та матеріали. Звідси можна перейти до екрану рекомендацій щодо проєктів. Структура інтерфейсу цього екрану наведена на рисунку 3.4.

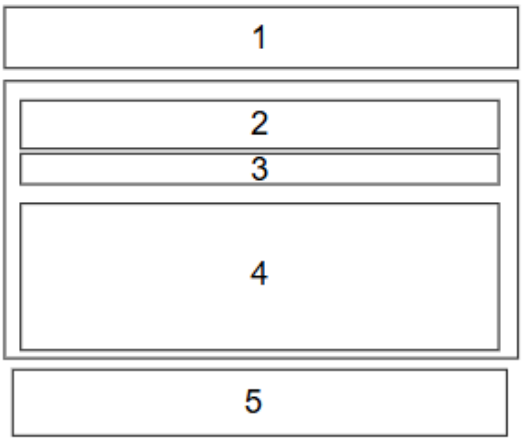


Рисунок 3.4 — Структурна схема інтерфейсу вікна результатів аналізу зображення

Складові інтерфейсу вікна результатів аналізу фотозображення:

1. Назва екрану.
2. Назва проєкту.
3. Назва категорії проєкту.
4. Перелік матеріалів.
5. Кнопка «Отримати рекомендації».

Вікно специфікацій проєкту являє собою перелік рекомендацій по реалізації виробничого проєкту, набір інструментів, інструкції по збірці з можливістю зберегти за експортувати у PDF сформовану специфікацію, а також перейти на YouTube для перегляду відео з реалізацією даного проєкту. Структура інтерфейсу цього екрану наведена на рисунку 3.5.

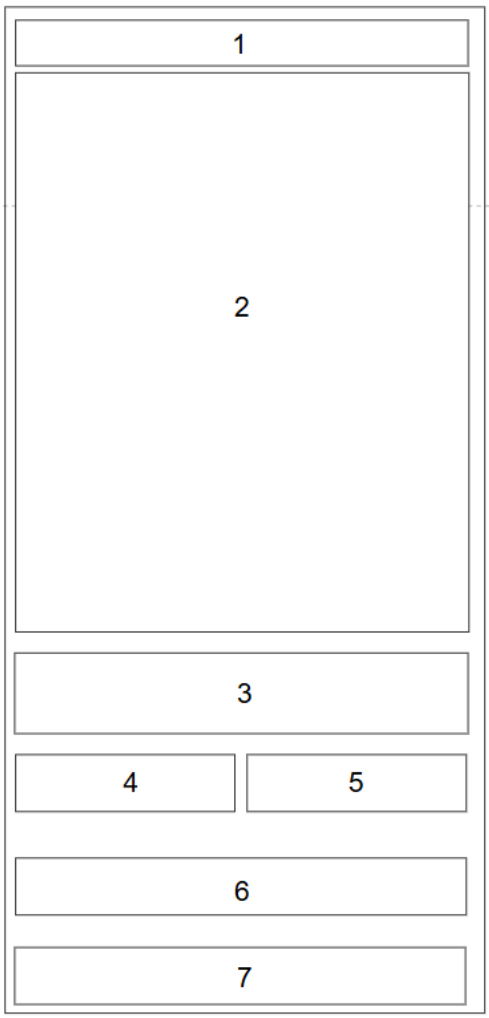


Рисунок 3.5 — Структурна схема інтерфейсу вікна специфікацій проекту

Складові елементи вікна специфікацій проекту:

1. Назва вікна.
2. Інструкції по реалізації.
3. Вартість проєкту.
4. Кнопка «Експорт PDF».
5. Кнопка «Зберегти».
6. Кнопка «Знайти відео на YouTube».
7. Кнопка «Новий проєкт».

Екран збережених специфікацій дозволяє переглядати короткий опис сформованих штучним інтелектом специфікацій, переходити до створення нових, та видаляти існуючі специфікації. Структура інтерфейсу екрану збережених специфікацій наведена на рисунку 3.6.

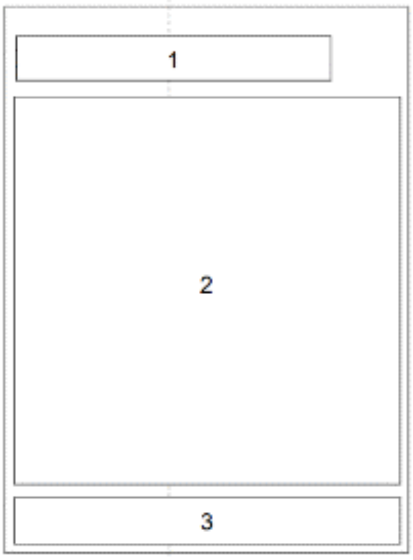


Рисунок 3.6 — Структурна схема інтерфейсу екрану збережених специфікацій

Складові вікна збережених специфікацій:

1. Назва екрану.
2. Список специфікацій.
3. Кнопка «Створити нову специфікацію».

Елемент запису із специфікацією містить короткий опис збереженої специфікації, такий як: назва проєкту, категорія, кількість компонент для реалізації,

вартість та дата створення. Структура запису із специфікаціями наведена на рисунку 3.7.

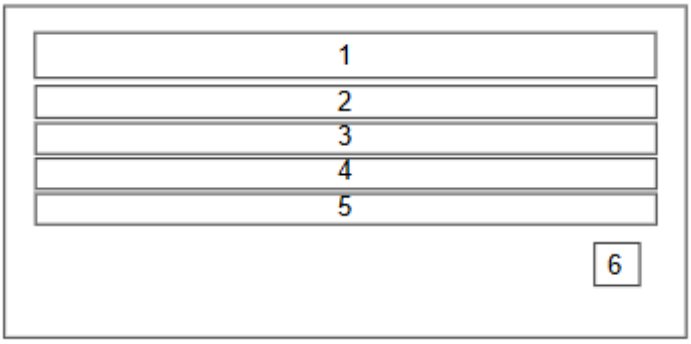


Рисунок 3.7 — Структурна схема інтерфейсу елемента запису із специфікаціями

Складові елемента запису із специфікаціями:

1. Назва проєкту.
2. Категорія.
3. Компоненти.
4. Вартість.
5. Дата створення.
6. Кнопка для видалення специфікації.

Таким чином було розроблено інтерфейс системи для підбору комплектуючих частин виробничих проєктів. Розроблений інтерфейс дозволяє аналізувати за фото наявні інструменти, та на їх основі формувати проєкти та специфікації, які можна реалізувати з них.

3.4 Висновки

У третьому розділі було проведено порівняльний аналіз мов програмування для розробки системи підбору комплектуючих частин виробничих проєктів, обрано мову Kotlin. Розроблено базу даних та інтерфейс користувача, що дозволяють отримувати специфікації проєктів по наявним інструментам, інструкції по реалізації, та набір додаткових інструментів, що необхідні для реалізації того чи іншого проєкту.

4 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Аналіз методів тестування

Автоматизоване тестування використовує спеціалізовані програмні засоби для виконання попередньо запрограмованих тестових сценаріїв [27]. При тестуванні мобільного застосунку для підбору комплектуючих частин виробничих проєктів даний метод демонструє певні обмеження. По-перше, складність інтерфейсу взаємодії з каталогами комплектуючих вимагає постійного оновлення автоматизованих тестів при зміні структури даних. По-друге, різноманітність виробничих комплектуючих та їх специфікацій створює численні варіанти взаємодії, які складно передбачити в автоматизованих сценаріях.

Тестування через програмний інтерфейс застосунку дозволяє перевірити функціональність без взаємодії з користувацьким інтерфейсом [28]. Однак даний мобільний застосунок має аспекти взаємодії з користувачем при виборі комплектуючих, що не можуть бути протестовані лише на рівні API.

Під час ручного тестування виконуються тестові сценарії тестувальником без використання автоматизованих інструментів [29]. Даний метод демонструє найвищі показники для мобільного застосунку з підбору комплектуючих частин виробничих проєктів з декількох причин. По-перше, тестувальник може оцінити не лише технічну функціональність, але й зручність використання інтерфейсу при підборі комплектуючих, що важливо для цільових користувачів. По-друге, людина здатна виявити неочевидні проблеми та невідповідності в логіці підбору комплектуючих, які можуть залишитися непоміченими при автоматизованому тестуванні. По-третє, ручне тестування забезпечує гнучкість при зміні вимог та оновленні каталогу комплектуючих, що є характерним для виробничої сфери. Зрештою, тестувальник може імітувати реальні сценарії використання застосунку в контексті виробничих процесів, що підвищує надійність та практичну цінність результатів тестування.

На основі проведеного аналізу можна зробити висновок, що метод ручного тестування є найбільш доцільним для мобільного застосунку з підбору

комплектуючих частин виробничих проєктів. Даний метод забезпечує оптимальне співвідношення між охопленням функціональності, виявленням проблем користувацького інтерфейсу та адаптивністю до змін у вимогах та каталогах комплектуючих. Незважаючи на більші часові витрати порівняно з деякими автоматизованими методами, ручне тестування надає найбільш релевантні результати щодо практичної придатності застосунку для цільової аудиторії та специфічних виробничих контекстів.

4.2 Тестування застосунку

Тестування застосунку розпочинається з головного екрану застосунку, який містить навігацію з іншими компонентами застосунку, такими як вибір фото наявних інструментів з галереї та перегляд збережених виробничих проєктів, а також містить коротку загальну інформацію про застосунок. Головний екран застосунку наведений на рисунку 4.1.

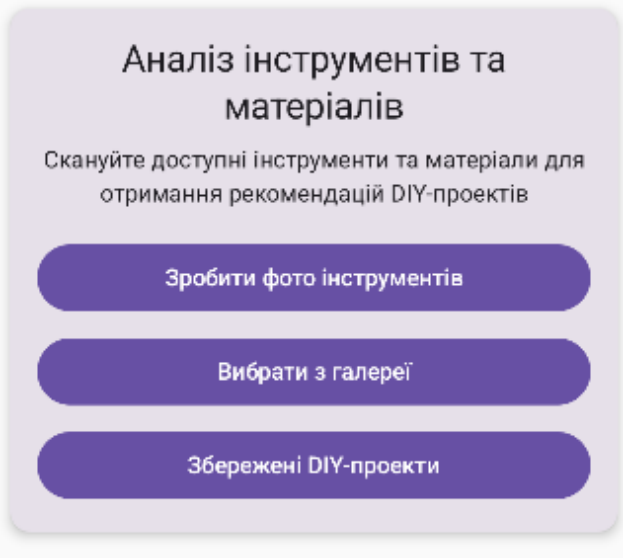


Рисунок 4.1 — Головний екран застосунку

Оберемо фото інструментів з галереї, що містять дерев’яні бруси і плити, а також цвяхи і молоток, та аналізуємо його за допомогою штучного інтелекту. Аналіз фото займає деякий час, протягом якого демонструється екран, який наведений на рисунку 4.2.

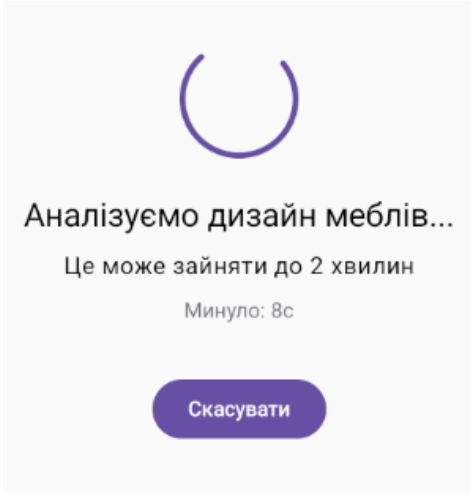


Рисунок 4.2 — Екран очікування аналізу фото інструментів

Результат аналізу фото наведено на рисунку 4.3. Він містить назву проєкту, який можна реалізувати з наявних інструментів та необхідні матеріали, а також кнопку для отримання детальних рекомендацій.

Рисунок 4.3 — Результат аналізу матеріалів за фото

Після натискання на кнопку «отримати рекомендації», відкривається нове вікно з анімацією завантаження, яке відображається, поки формуються рекомендації (рисунок 4.4).

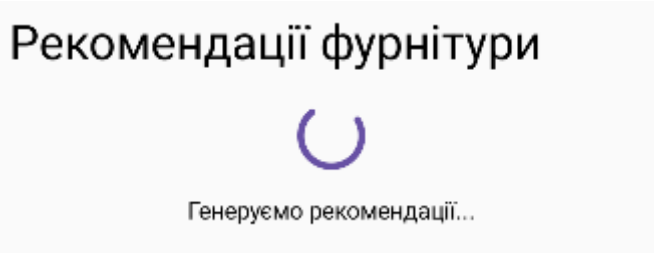


Рисунок 4.4 — Завантаження рекомендацій

На рисунку 4.5 наведено отримані рекомендації.

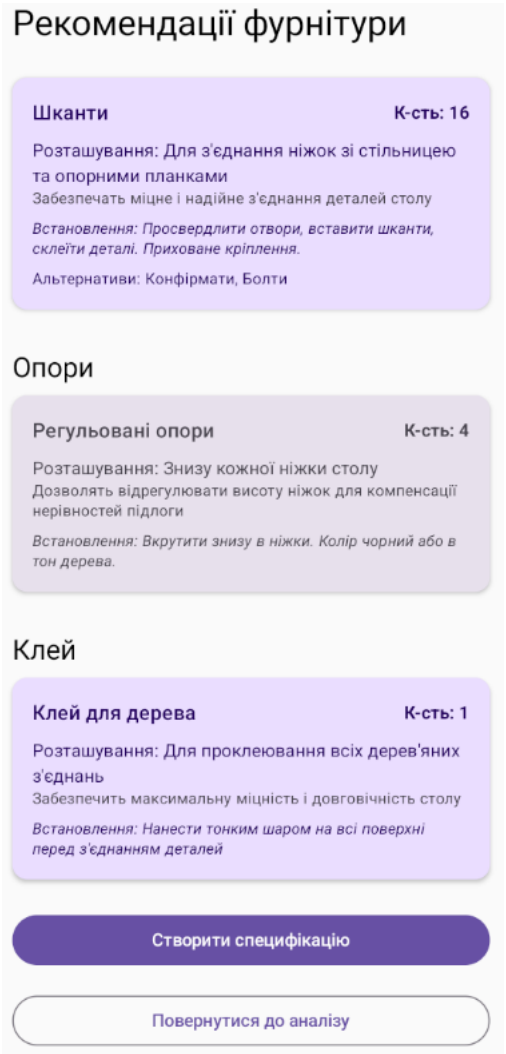


Рисунок 4.5 — Екран рекомендацій по реалізації проєкту

Рекомендації містять детальний перелік матеріалів та інструментів, які необхідні для реалізації виробничого проєкту, їхню кількість, та їхнє застосування. Також пропонуються альтернативні способи виконати ту саму роботу із застосуванням інших інструментів чи матеріалів, якщо наведені не є доступними.

Екран специфікації проєкту розширює екран рекомендацій. Він містить ті ж рекомендації, а також інструкції по реалізації проєкту, можливість експортувати специфікацію у PDF, зберегти у базі даних застосунку, переглянути відео з YouTube по реалізації цього проєкту. Зображення цього вікна наведені на рисунку 4.6.

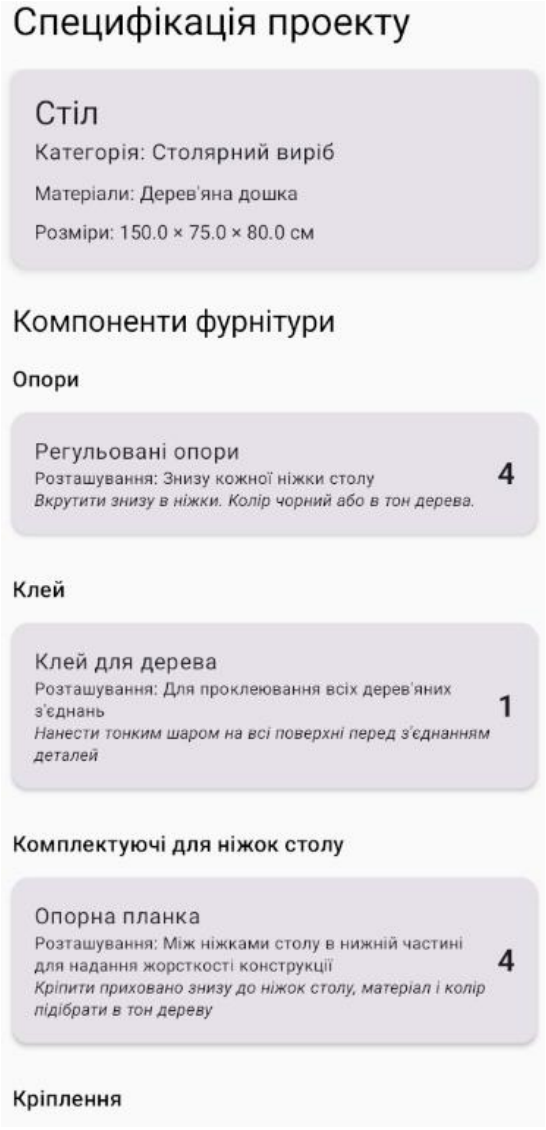


Рисунок 4.6 — Екран специфікації проєкту

На рисунку 4.7 наведено коротко інструкції по збірці меблів. Міститься також

кнопка для генерування детальних інструкцій за потреби.

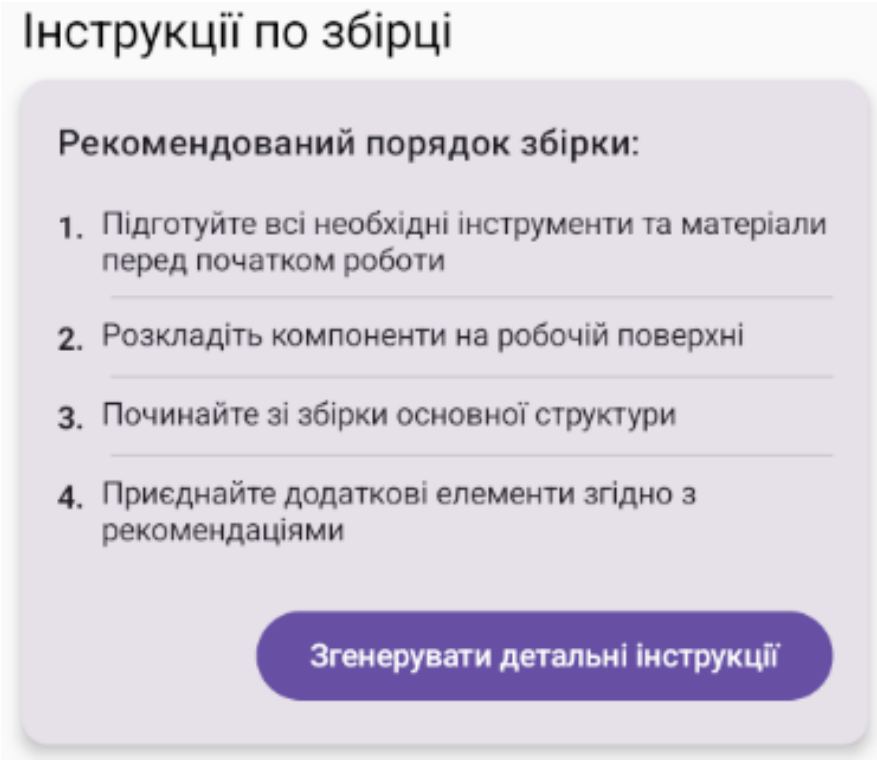


Рисунок 4.7 - Екран специфікації проєкту (продовження)

При натисканні на кнопку «Згенерувати детальні інструкції» спершу відображається анімація завантаження, як на рисунку 4.8.

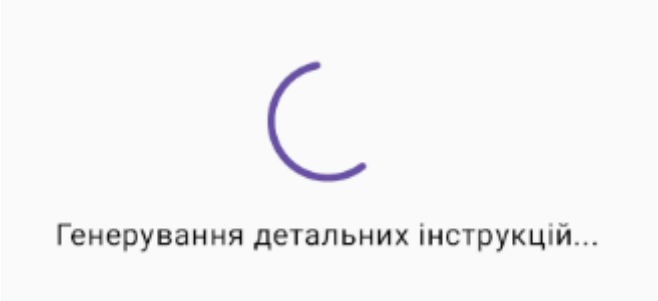


Рисунок 4.8 — Генерування детальних інструкцій

Згенеровані детальні інструкції продемонстровані на рисунку 4.9. Вони містять детальний перелік етапів виконання збірки меблів.

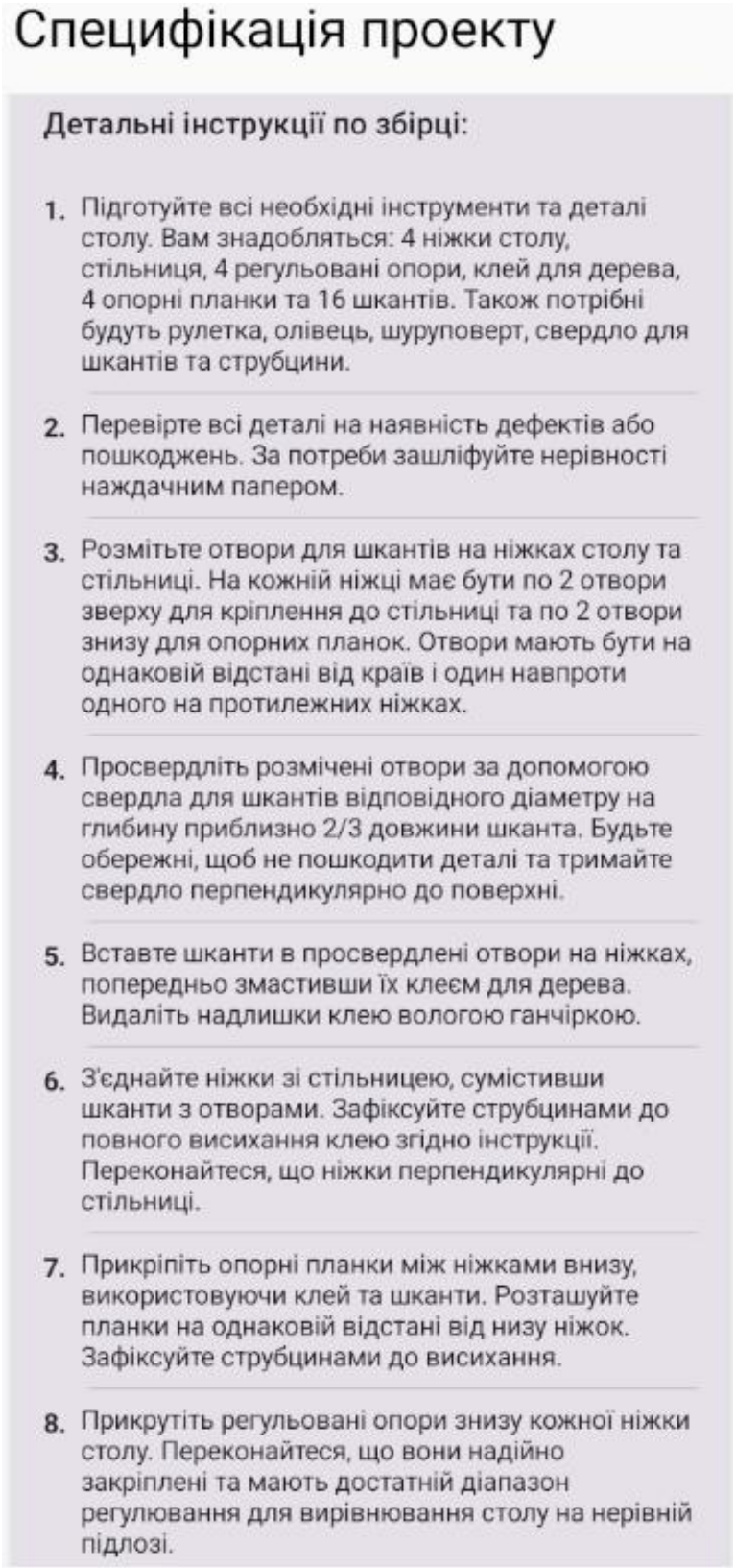


Рисунок 4.9 — Згенеровані детальні інструкції

Натиснемо на кнопку «Знайти відео на YouTube», після чого відбувається перехід на YouTube із пошуковим запитом, що стосується теми відео (у даному

випадку це дерев’яний стіл). У результаті відображається перелік відео, як на рисунку 4.10.

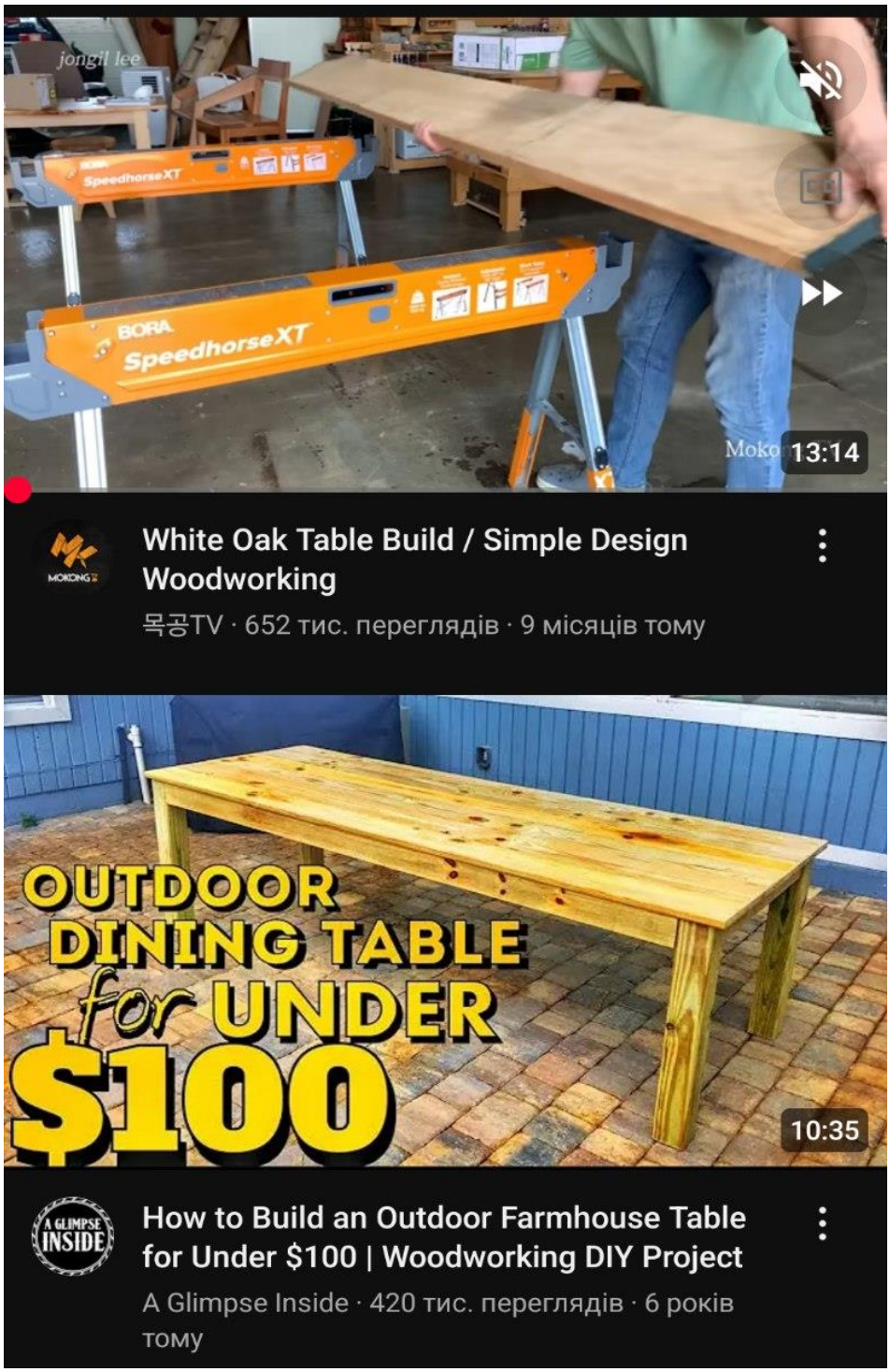


Рисунок 4.10 — Перелік відео по темі проєкту на YouTube після переходу із застосунку

Повернемося назад до застосунку та збережемо отриману специфікацію,

натиснувши на відповідну кнопку. Після цього перейдемо у екран «Збережені специфікації», де наведені нещодавно створені специфікації (рисунок 4.11).

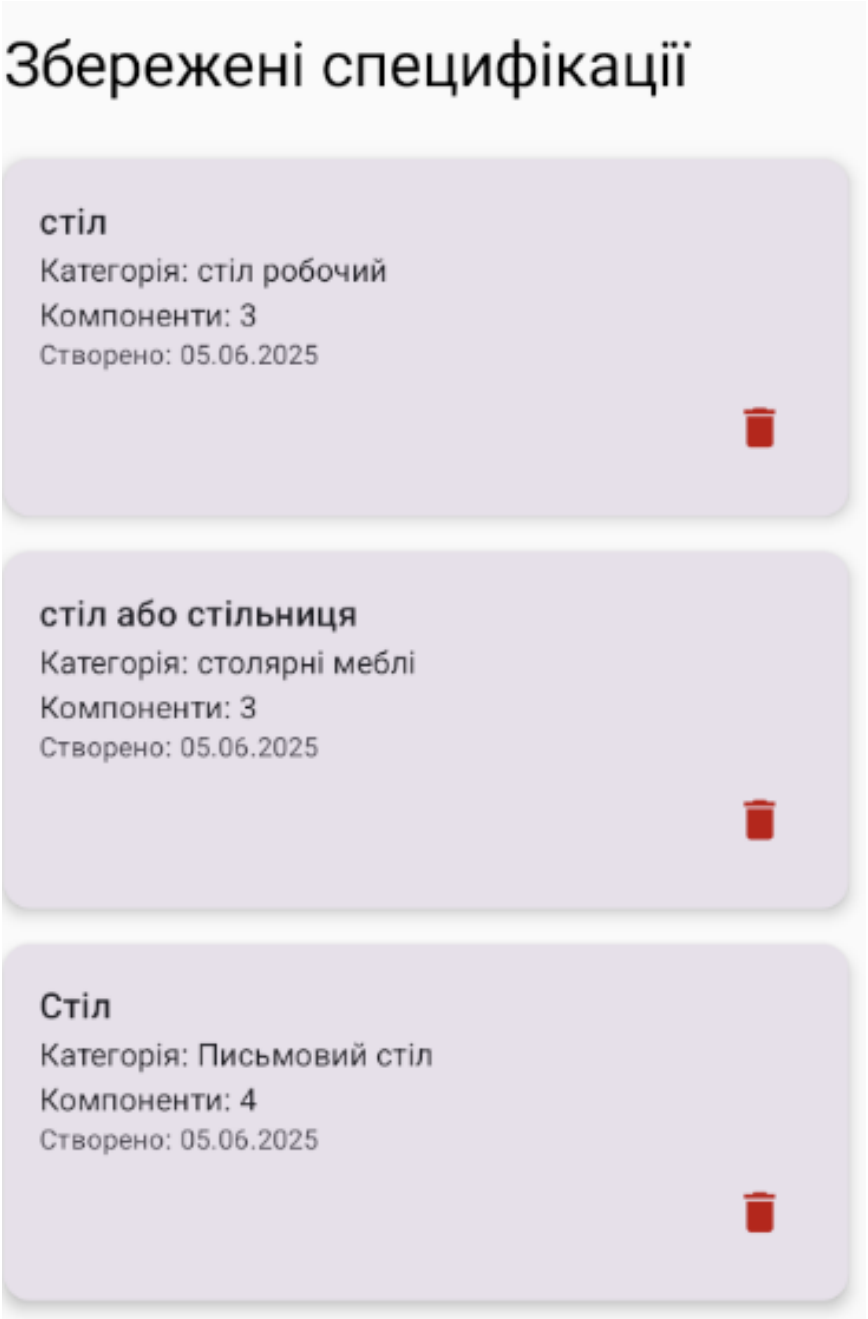


Рисунок 4.11 — Збережені специфікації проєктів

Також можемо експортувати отриману специфікацію у PDF, натиснувши на відповідну кнопку у екрані специфікацій. У результаті з’явиться вікно, як на рисунку 4.12, яке пропонує обрати спосіб перегляду експортованої специфікації. Воно дозволяє надрукувати специфікацію, надіслати у вигляді повідомлення,

переслати по Bluetooth, а також іншими способами за наявності.

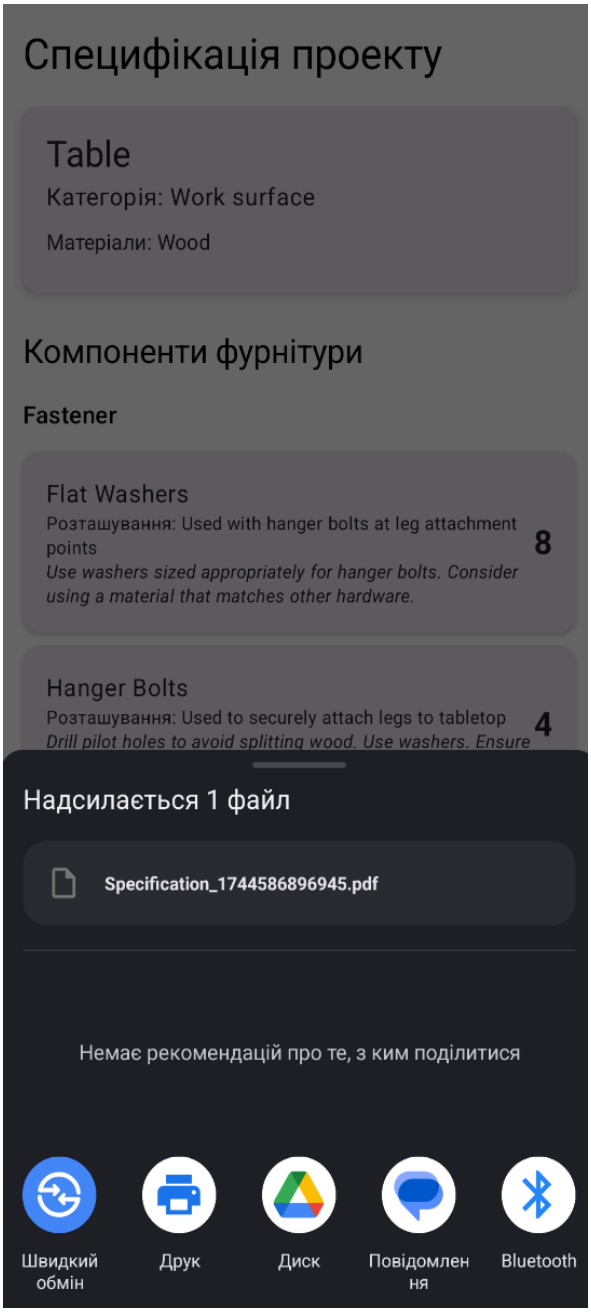
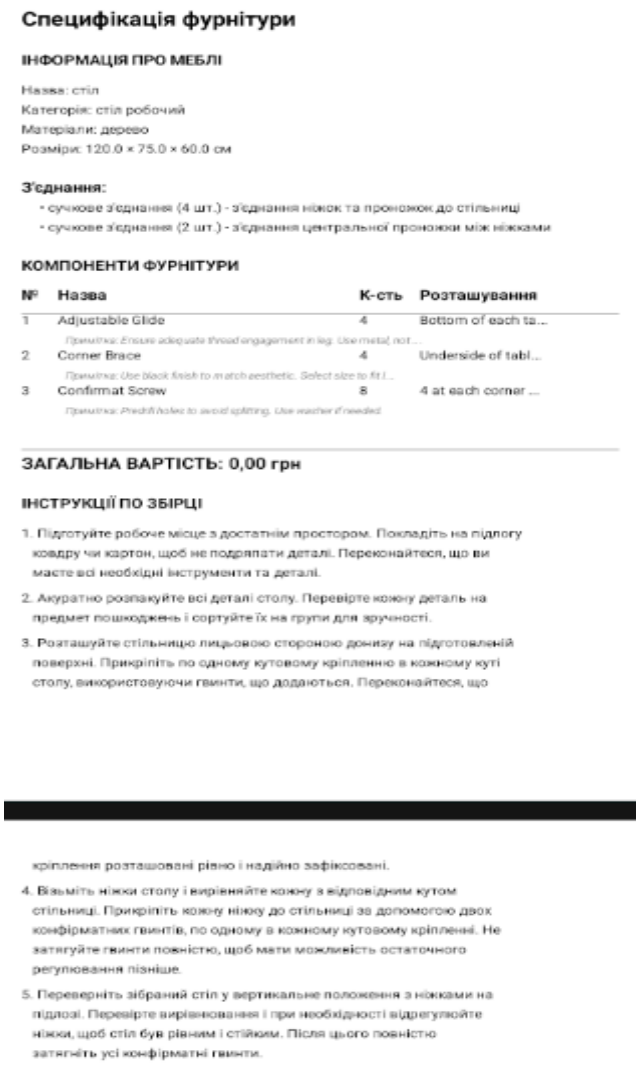


Рисунок 4.12 — Меню експорту та перегляду специфікації

Переглянемо отриману специфікацію у форматі PDF (рисунок 4.13). Вона містить згенеровану раніше інформацію із специфікаціями проєкту. Вона містить категорію меблів, назву, матеріали та їх розміри, з’єднання, компоненти фурнітури, загальну вартість, інструкції по збірці.



Специфікація фурнітури

ІНФОРМАЦІЯ ПРО МЕБЛІ

Назва: стіл
Категорія: стіл робочий
Матеріали: дерево
Розміри: 120.0 × 75.0 × 60.0 см

З'єднання:

- сучкове з'єднання (4 шт.) - з'єднання ніжок та проножок до стільниці
- сучкове з'єднання (2 шт.) - з'єднання центральної проножки між ніжками

КОМПОНЕНТИ ФУРНІТУРИ

№	Назва	К-сть	Розташування
1	Adjustable Glide	4	Bottom of each ta...
	Примітка: Ensure adequate thread engagement in leg. Use metal, not ...		
2	Corner Brace	4	Underside of tabl...
	Примітка: Use black finish to match aesthetic. Select size to fit t...		
3	Confirmat Screw	8	4 at each corner ...
	Примітка: Predrill holes to avoid splitting. Use washer if needed.		

ЗАГАЛЬНА ВАРТІСТЬ: 0,00 грн

ІНСТРУКЦІЇ ПО ЗБІРЦІ

1. Підготуйте робоче місце з достатнім простором. Покладіть на підлогу ковдру чи картон, щоб не подряпати деталі. Переконайтеся, що ви маєте всі необхідні інструменти та деталі.
2. Акуратно розпакуйте всі деталі столу. Перевірте кожну деталь на предмет пошкоджень і сортуйте їх на групи для зручності.
3. Розташуйте стільницю лицьовою стороною донизу на підготовленій поверхні. Прикріпіть по одному кутовому кріпленню в кожному куті столу, використовуючи гвинти, що додаються. Переконайтеся, що кріплення розташовані рівно і надійно зафіксовані.
4. Візьміть ніжки столу і вирівняйте кожну з відповідним кутом стільниці. Прикріпіть кожну ніжку до стільниці за допомогою двох конфірматних гвинтів, по одному в кожному кутовому кріпленні. Не затягуйте гвинти повністю, щоб мати можливість остаточного регулювання пізніше.
5. Переверніть зібраний стіл у вертикальне положення з ніжками на підлозі. Перевірте вирівнювання і при необхідності відрегулюйте ніжки, щоб стіл був рівним і стійким. Після цього повністю затягніть усі конфірматні гвинти.

Рисунок 4.13 — PDF-документ специфікації проєкту

Таким чином, було проведено тестування системи підбору комплектуючих частин виробничих проєктів. Продемонстровано її функціонал, наведено приклад використання. Доведено, що розроблена система виконує поставлені на початку розробки завдання.

4.3 Висновки

У четвертому розділі було проведено аналіз методів тестування системи підбору комплектуючих частин виробничих проєктів. Проведено її тестування, в ході якого продемонстровано її функціонал, наведено приклад використання, та доведено, що розроблена система виконує поставлені на початку розробки завдання.

ВИСНОВКИ

У результаті виконання бакалаврської кваліфікаційної роботи було розроблено мобільний застосунок для автоматизації комплектування фурнітурою меблів на етапі проектування. Бакалаврську кваліфікаційну роботу реалізовано згідно методичних вказівок [30].

Було розроблено архітектуру мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування за принципом багаторівневої архітектури, що відповідає сучасним підходам до розробки мобільних застосунків, що забезпечує чітке розділення обов'язків між шарами, підвищує модульність системи, спрощує тестування.

Модифіковано метод оптимізації вибору меблевих комплектуючих для підвищення ефективності формування списку фурнітури меблів на етапі проектування. Цей метод використовує модель claude-3-opus ШІ для визначення комплектуючих, і формування їх специфікацій..

Розроблено інтерфейс користувача застосунку, побудовано структурні схеми основних екранів застосунку, наведено їх складові та їх розміщення у інтерфейсі застосунку.

Розроблено алгоритми роботи застосунку, які дозволяють розпізнавати інструменти і матеріали, генерувати рекомендацій проєктів, інструкції та шукати відеоінструкції. Побудовано блок-схеми цих алгоритмів.

Розроблено функціонал застосунку, який дозволяє надавати фото наявних інструментів системі, на основі чого завдяки штучному інтелекту відбувається аналіз, що знаходиться на фото, з відповідним підбором виробничих проєктів та інструментів для реалізації проєкту з використанням цих наявних інструментів. Крім цього, реалізована можливість формувати інструкції та рекомендації по реалізації, а також специфікації проєкту.

Проведено тестування застосунку, в ході якого продемонстровано його функціонал, наведено приклад використання, та доведено, що розроблена система виконує поставлені на початку розробки завдання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Research on Design Method of Intelligent Furniture. (2020). Proceedings of the 2020 International Conference on E-Business[Електронний ресурс]. URL: http://iceb.johogo.com/proceedings/2020/ICEB_2020_paper_40_full.pdf

2. Transforming The Furniture Industry in The Digital Age. (2023). ResearchGate [Електронний ресурс]. URL: https://www.researchgate.net/publication/376815769_Transforming_The_Furniture_Industry_in_The_Digital_Age

3. HomeByMe. (2024). "Furniture design software for mobile devices". Professional 3D Services [Електронний ресурс]. URL: https://professional3dservices.com/blog/furniture-design-software.html

4. Moblo - 3D furniture modeling. (2021). Apple App Store [Електронний ресурс]. URL: https://apps.apple.com/us/app/moblo-3d-furniture-modeling/id1549380017

5. Smart Furniture Market Size and Share | Statistics. (2023). International Data Corporation. URL: https://www.nextmsc.com/report/smart-furniture-market Shapr3D. (2022).

6. Top 10 furniture design software products for Mac & Windows of 2022. URL: https://www.shapr3d.com/blog/furniture-design-software

7. В.В. Колос, Г.Б. Ракитянська. Розробка мобільного асистента для підбору комплектуючих частин виробничих проектів / Матеріали LIV Всеукраїнської науково-технічної конференції підрозділів Вінницького національного технічного університету (НТКП ВНТУ–2025) : збірник доповідей. URL: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2025/paper/view/23713/19726

8. Transforming The Furniture Industry in The Digital Age. (2023). ResearchGate. URL: https://www.researchgate.net/publication/376815769_Transforming_The_Furniture_Industry_in_The_Digital_Age

9. Moblo - 3D furniture modeling. (2021). Apple App Store. URL: https://apps.apple.com/us/app/moblo-3d-furniture-modeling/id1549380017

10. World Intellectual Property Organization. (2024). WIPO Technology Trends 2024: Digital Manufacturing. WIPO Publication. URL: https://www.wipo.int/publications/en/

11. McKinsey & Company. (2023). The Impact of AI on Manufacturing: Creating Value Through Intelligent Automation. McKinsey Global Institute. URL: https://www.mckinsey.com/industries/advanced-electronics/our-insights/

12. Autodesk. (2024). Woodworking & Furniture Design Software. URL: https://www.autodesk.com/solutions/woodworking-furniture-design-software

13. European Furniture Manufacturers Federation. (2024). Digital Transformation in Furniture Manufacturing: Survey Results 2024. EFMF Publications. URL: https://www.efic.eu/publications-and-reports

14. Nielsen Norman Group. (2023). User Experience in Mobile Applications for Manufacturing: Effects on Productivity. URL: https://www.nngroup.com/articles/manufacturing-mobile-ux/

15. Smart Furniture Market Size and Share | Statistics. (2023). International Data Corporation. URL: https://www.nextmsc.com/report/smart-furniture-market

16. IKEA Assemble App Електронний ресурс]. URL: https://medium.com/design-bootcamp/ui-ux-case-study-ikea-assemble-app-b3523e45c7a6

17. Woodworking Calculator Pro [Електронний ресурс]. URL: https://woodworkingcalculatorpro.com/

18. Blum Configurator [Електронний ресурс]. URL: https://www.blum.com/ua/uk/services/planning-construction-product-selection/cabinet-configurator/

19. What is an API? [Електронний ресурс]. URL: https://www.ibm.com/think/topics/api

20. SDK vs API [Електронний ресурс]. URL: https://www.ibm.com/think/topics/api-vs-sdk

21. D. Jacobson. APIs: A Strategy Guide: Creating Channels with Application Programming Interfaces. Farnham: O'Reiily Media, 2012. 146c.

22. N. Ford. Software Architecture: The Hard Parts: Modern Trade-Off Analyses for Distributed Architectures. Farnham: O'Reilly Media, 2021. 459c.

23. Asynchronous programming techniques [Електронний ресурс]. URL: https://kotlinlang.org/docs/async-programming.html

24. D. McGregor. Java to Kotlin: A Refactoring Guidebook. Farnham: O'Reilly Media, 2021. 422c.

25. B. Evans. The Well-Grounded Java Developer. New York: Manning, 2023. 704c.

26. Jonathan Sande. Dart Apprentice: Fundamentals (First Edition): Modern Cross-Platform Programming With Dart. Alexandria: Kodeco Inc., 2022. 273c.

27. Step to step how to test mobile applications [Електронний ресурс]. URL: https://qalified.com/blog/how-to-test-mobile-applications/

28. Test your app on Android [Електронний ресурс]. URL: https://developer.android.com/training/testing

29. Mobile applications testing types and approaches [Електронний ресурс]. URL: https://testlio.com/blog/mobile-testing-types/

30. Методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів спеціальності 121 "Інженерія програмного забезпечення" / Уклад. О.Н. Романюк, Г.О. Черноволик – Вінниця: ВНТУ, 2022. – 52с.

ДОДАТКИ

Додаток А – Технічне завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
Романюк О.Н.
«25» 03 2025 р.

Технічне завдання
на бакалаврську кваліфікаційну роботу «Розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування»
за спеціальністю
121 – Інженерія програмного забезпечення

Керівник бакалаврської кваліфікаційної роботи:
к.т.н., доц. каф. ПЗ Рейда О.М.
«24» 03 2025 року.

Виконав:
студент гр. 2ПІ-21б Колос В.В.
«24» 03 2025 року.

м. Вінниця – 2025 рік

1. Найменування та галузь застосування

Бакалаврська кваліфікаційна робота: «Розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування».

Галузь застосування – мобільні технології.

2. Підстава для розробки.

Підставою для виконання бакалаврської кваліфікаційної роботи (БКР) є індивідуальне завдання на БКР та наказ №97 від 20 березня 2025 року про закріплення тем БКР.

3. Мета та призначення розробки.

Метою роботи є покращення процесу комплектування фурнітурою меблів на етапі проектування за допомогою спеціалізованого програмного забезпечення, що надає рекомендації та інструкції..

4. Вихідні дані для проведення НДР

1. Research on Design Method of Intelligent Furniture. (2020). Proceedings of the 2020 International Conference on E-Business[Електронний ресурс]. URL: http://iceb.johogo.com/proceedings/2020/ICEB_2020_paper_40_full.pdf

2. McKinsey & Company. (2023). The Impact of AI on Manufacturing: Creating Value Through Intelligent Automation. McKinsey Global Institute. URL: https://www.mckinsey.com/industries/advanced-electronics/our-insights/

3. Asynchronous programming techniques [Електронний ресурс]. URL: https://kotlinlang.org/docs/async-programming.html

4. D. McGregor. Java to Kotlin: A Refactoring Guidebook. Farnham: O’Reilly Media, 2021. 422c.

5. Технічні вимоги

Вхідні дані — інформація користувача.

Вихідні дані — графічний інтерфейс, інформація про комплектуючі, інструкція з виконання проєктів.

6. Конструктивні вимоги.

Інтерфейс програми повинен відповідати естетичним та ергономічним вимогам, повинна бути зручною в обслуговуванні та керуванні.

Графічна та текстова документація повинна відповідати діючим стандартам України.

7. Перелік технічної документації, що пред'являється по закінченню робіт:

–пояснювальна записка до бакалаврської кваліфікаційної роботи;

–технічне завдання;

–лістинги програми.

8. Вимоги до рівня уніфікації та стандартизації

При розробці програмних засобів слід дотримуватися уніфікації і ДСТУ.

9. Стадії та етапи розробки

№ з/п	Назва етапів бакалаврської кваліфікаційної роботи	Строк виконання етапів роботи
1	Аналіз стану розвитку застосунків для автоматизації комплектування фурнітурою меблів	25.03.25 - 05.04.25
2	Розробка структури та алгоритмів програмного застосунку	06.04.25 - 18.04.25
3	Варіантний аналіз та вибір мови програмування розробки	19.04.25 - 21.04.25
4	Розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування	22.04.25 - 17.05.25
5	Тестування програмного забезпечення	18.05.25 - 25.05.25
6	Оформлення матеріалів до захисту БКР	26.05.25 - 30.05.25

10. Порядок контролю та прийняття.

Усі етапи бакалаврської роботи контролюються науковим керівником згідно плану по виконанню роботи. Прийняття бакалаврської кваліфікаційної роботи здійснюється ДЕК, затвердженою зав. кафедрою згідно з графіком захисту.

Додаток Б – Протокол перевірки на наявність текстових запозичень

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: Розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування.

Тип роботи: бакалаврська кваліфікаційна робота
(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ кафедра програмного забезпечення, ФІТКІ, 2ПІ-21б
(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 5,5%

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

■Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту

□У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

□У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

_____ _____
(прізвище, ініціали, посада) (підпис)

_____ _____
(прізвище, ініціали, посада) (підпис)

Особа, відповідальна за перевірку _____ Черноволик Г. О.

З висновком експертної комісії ознайомлений(-на)

Керівник _____ Рейда О.М., к.т.н., доц. кафедри ПЗ
(підпис) (прізвище, ініціали, посада)

Здобувач _____ Колос В.В.
(підпис) (прізвище, ініціали)

Додаток В – Лістинг програми

```
@Module
@InstallIn(SingletonComponent::class)
object AppModule

    @Provides
    @Singleton
    fun provideProjectDao(appDatabase: AppDatabase): ProjectDao {
        return appDatabase.projectDao()
    }

    @Provides
    @Singleton
    fun provideImageAnalyzer(): ImageAnalyzer {
        return ImageAnalyzer(BuildConfig.CLAUDE_API_KEY)
    }

    @Provides
    @Singleton
    fun provideProjectRecommendationEngine(): ProjectRecommendationEngine {
        return ProjectRecommendationEngine(BuildConfig.CLAUDE_API_KEY)
    }

    @Provides
    @Singleton
    fun provideInstructionsGenerator(): InstructionsGenerator {
        return InstructionsGenerator(BuildConfig.CLAUDE_API_KEY)
    }

    @Provides
    @Singleton
    fun provideYouTubeSearchModule(): YouTubeSearchModule {
        return YouTubeSearchModule(BuildConfig.YOUTUBE_API_KEY)
    }

    @Provides
    @Singleton
    fun provideGson(): Gson {
```

```
    return Gson()
}

@Provides
@Singleton
fun provideOkHttpClient(): OkHttpClient {
    return OkHttpClient.Builder()
        .connectTimeout(30, TimeUnit.SECONDS)
        .readTimeout(30, TimeUnit.SECONDS)
        .build()
}

@Provides
@Singleton
fun provideAppDatabase(@ApplicationContext context: Context): AppDatabase {
    return Room.databaseBuilder(
        context,
        AppDatabase::class.java,
        "furniture_database"
    ).build()
}

@Provides
@Singleton
fun provideHardwareDao(appDatabase: AppDatabase): HardwareDao {
    return appDatabase.hardwareDao()
}

@Provides
@Singleton
fun provideFurnitureDesignDao(appDatabase: AppDatabase): FurnitureDesignDao {
    return appDatabase.furnitureDesignDao()
}

@Provides
@Singleton
fun provideSpecificationDao(appDatabase: AppDatabase): SpecificationDao {
    return appDatabase.specificationDao()
}
```

```
@Provides
@Singleton
fun provideFurnitureAnalyzer(
    httpClient: OkHttpClient ,
    @Named("ClaudeApiKey") apiKey: String
): FurnitureAnalyzer {
    return FurnitureAnalyzer(httpClient, apiKey)
}

@Provides
@Singleton
fun provideHardwareDatabase(
    hardwareDao: HardwareDao,
    gson: Gson
): HardwareDatabase {
    return HardwareDatabase(hardwareDao, gson)
}

@Provides
@Singleton
fun provideHardwareRecommendationEngine(
    hardwareDatabase: HardwareDatabase ,
    @Named("ClaudeApiKey") apiKey: String ,
    httpClient: OkHttpClient
): HardwareRecommendationEngine {
    return HardwareRecommendationEngine(hardwareDatabase, apiKey, httpClient)
}

@Provides
@Singleton
fun provideHardwarePriceCalculator(): HardwarePriceCalculator {
    return HardwarePriceCalculator()
}

@Provides
@Singleton
fun provideSpecificationDocumentGenerator(
    priceCalculator: HardwarePriceCalculator
): SpecificationDocumentGenerator {
    return SpecificationDocumentGenerator(priceCalculator)
```

```
}

@Provides
@Singleton
@Named("YouTubeApiKey")
fun provideYouTubeApiKey(): String {
    return BuildConfig.YOUTUBE_API_KEY
}

@Provides
@Singleton
@Named("ClaudeApiKey")
fun provideClaudeApiKey(): String {
    return BuildConfig.CLAUDE_API_KEY
}

@Provides
@Singleton
fun provideFurnitureHardwareRepository(
    hardwareDao: HardwareDao,
    designDao: FurnitureDesignDao,
    specificationDao: SpecificationDao,
    furnitureAnalyzer: FurnitureAnalyzer,
    recommendationEngine: HardwareRecommendationEngine,
    documentGenerator: SpecificationDocumentGenerator,
    okHttpClient: OkHttpClient,
    @Named("ClaudeApiKey") apiKey: String,
    @Named("YouTubeApiKey") youtubeApiKey: String
): FurnitureHardwareRepository {
    return FurnitureHardwareRepository(
        hardwareDao,
        designDao,
        specificationDao,
        furnitureAnalyzer,
        recommendationEngine,
        documentGenerator,
        okHttpClient,
        apiKey,
        youtubeApiKey
    )
```

```
    }
}

@AndroidEntryPoint
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            MainNavigation()
        }
    }
}

@Composable
fun MainNavigation() {
    val navController = rememberNavController()
    val mainViewModel: MainViewModel = hiltViewModel()
    val furnitureHardwareViewModel: FurnitureHardwareViewModel = hiltViewModel()

    NavHost(navController = navController, startDestination = "home") {
        composable("home") { HomeScreen(navController) }
        composable("camera") { CameraScreen(navController) }
        composable("gallery") { GalleryScreen(navController, furnitureHardwareViewModel) }
        composable("analysis") { AnalysisScreen(navController, mainViewModel) }
        composable("projects") { ProjectsScreen(navController) }
        composable("analysis_loading") { AnalysisLoadingScreen(navController, furnitureHardwareViewModel) }
        composable("analysis_results") { AnalysisResultsScreen(navController, furnitureHardwareViewModel) }
        composable("hardware_recommendations") { HardwareRecommendationsScreen(navController, furnitureHardwareViewModel) }
        composable("specification") { SpecificationScreen(navController, furnitureHardwareViewModel) }
        composable("saved_specifications") { SavedSpecificationsScreen(navController, furnitureHardwareViewModel) }
        composable("project_details/{projectId}") { backStackEntry ->
            ProjectDetailsScreen(
                projectId = backStackEntry.arguments?.getString("projectId") ?: "",
                navController = navController
            )
        }
        composable("saved_projects") { SavedProjectsScreen(navController) }
    }
}
```

```
@Composable
fun HomeScreen(navController: NavController) {
    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp),
        horizontalAlignment = Alignment.CenterHorizontally
    ) {
        Text(
            text = "DIY Project Builder",
            style = MaterialTheme.typography.headlineLarge,
            textAlign = TextAlign.Center
        )

        Spacer(modifier = Modifier.height(32.dp))

        Card(
            modifier = Modifier
                .fillMaxWidth()
                .padding(8.dp),
            elevation = CardDefaults.cardElevation(defaultElevation = 4.dp)
        ) {
            Column(
                modifier = Modifier.padding(16.dp),
                horizontalAlignment = Alignment.CenterHorizontally
            ) {
                Text(
                    text = "Аналіз інструментів та матеріалів",
                    style = MaterialTheme.typography.titleLarge,
                    textAlign = TextAlign.Center
                )

                Spacer(modifier = Modifier.height(8.dp))

                Text(
                    text = "Скануйте доступні інструменти та матеріали для отримання рекомендацій DIY-проєктів",
                    style = MaterialTheme.typography.bodyMedium,
                    textAlign = TextAlign.Center
                )
```

```
        Spacer(modifier = Modifier.height(16.dp))

        Button(
            onClick = { navController.navigate("camera") },
            modifier = Modifier.fillMaxWidth()
        ) {
            Text("Зробити фото інструментів")
        }

        Spacer(modifier = Modifier.height(8.dp))

        Button(
            onClick = { navController.navigate("gallery") },
            modifier = Modifier.fillMaxWidth()
        ) {
            Text("Вибрати з галереї")
        }

        Spacer(modifier = Modifier.height(8.dp))

        Button(
            onClick = { navController.navigate("saved_projects") },
            modifier = Modifier.fillMaxWidth()
        ) {
            Text("Збережені DIY-проєкти")
        }
    }
}

Spacer(modifier = Modifier.height(24.dp))

Card(
    modifier = Modifier
        .fillMaxWidth()
        .padding(8.dp),
    elevation = CardDefaults.cardElevation(defaultElevation = 4.dp)
) {
    Column(
        modifier = Modifier.padding(16.dp),
```

```
            horizontalAlignment = Alignment.CenterHorizontally
        ) {
            Text(
                text = "Аналіз дизайну меблів",
                style = MaterialTheme.typography.titleLarge,
                textAlign = TextAlign.Center
            )

            Spacer(modifier = Modifier.height(8.dp))

            Text(
                text = "Аналізуйте дизайн меблів для отримання рекомендацій по фурнітурі та інструкцій зі збірки",
                style = MaterialTheme.typography.bodyMedium,
                textAlign = TextAlign.Center
            )

            Spacer(modifier = Modifier.height(16.dp))

            Button(
                onClick = { navController.navigate("gallery") },
                modifier = Modifier.fillMaxWidth(),
                colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.secondary)
            ) {
                Text("Аналізувати дизайн меблів")
            }

            Spacer(modifier = Modifier.height(8.dp))

            Button(
                onClick = { navController.navigate("saved_specifications") },
                modifier = Modifier.fillMaxWidth(),
                colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.secondary)
            ) {
                Text("Збережені специфікації")
            }
        }
    }
  }
}
```

```
@Composable
fun ProjectsScreen(
    navController: NavController ,
    viewModel: MainViewModel = hiltViewModel()
) {
    val recommendedProjects by viewModel.recommendedProjects.collectAsState()

    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp)
    ) {
        Text(
            text = "Recommended Projects",
            style = MaterialTheme.typography.headlineSmall
        )

        Spacer(modifier = Modifier.height(16.dp))

        LazyColumn {
            items(recommendedProjects) { project ->
                ProjectCard(
                    project = project,
                    onClick = {
                        viewModel.selectProject(project)
                        navController.navigate("project_details/${project.id}")
                    }
                )

                Spacer(modifier = Modifier.height(8.dp))
            }
        }
    }
}

@Composable
fun ProjectDetailsScreen(
    projectId: String ,
    navController: NavController ,
    viewModel: MainViewModel = hiltViewModel()
```

```
) {
    val currentProject by viewModel.currentProject.collectAsState()
    val isLoading by viewModel.isLoading.collectAsState()

    LaunchedEffect(projectId) {
        viewModel.loadProject(projectId)
    }

    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp)
            .verticalScroll(rememberScrollState())
    ) {
        if (isLoading) {
            Box(
                modifier = Modifier.fillMaxWidth(),
                contentAlignment = Alignment.Center
            ) {
                CircularProgressIndicator()
            }
        } else {
            currentProject?.let { project ->
                Text(
                    text = project.title,
                    style = MaterialTheme.typography.headlineMedium
                )

                Spacer(modifier = Modifier.height(8.dp))

                Row {
                    Text(
                        text = "Difficulty: ${project.difficulty}",
                        style = MaterialTheme.typography.bodyLarge
                    )

                    Spacer(modifier = Modifier.width(16.dp))

                    Text(
                        text = "Time: ${project.timeEstimate}",
```

```
            style = MaterialTheme.typography.bodyLarge
        )
    }

    Spacer(modifier = Modifier.height(16.dp))

    Text(
        text = project.description,
        style = MaterialTheme.typography.bodyLarge
    )

    Spacer(modifier = Modifier.height(16.dp))

    Text(
        text = "Required Tools:",
        style = MaterialTheme.typography.titleLarge
    )

    project.requiredTools.forEach { tool ->
        Text("• $tool")
    }

    Spacer(modifier = Modifier.height(16.dp))

    Text(
        text = "Required Materials:",
        style = MaterialTheme.typography.titleLarge
    )

    project.requiredMaterials.forEach { material ->
        Text("• $material")
    }

    if (project.missingItems.isNotEmpty()) {
        Spacer(modifier = Modifier.height(16.dp))

        Text(
            text = "Missing Items:",
            style = MaterialTheme.typography.titleLarge,
            color = MaterialTheme.colorScheme.error
```

```
    )

    project.missingItems.forEach { item ->
        Text("• $item", color = MaterialTheme.colorScheme.error)
    }
}

Spacer(modifier = Modifier.height(24.dp))

Text(
    text = "Instructions:",
    style = MaterialTheme.typography.titleLarge
)

project.instructions.forEachIndexed { index, instruction ->
    Text("${index + 1}. $instruction")
    Spacer(modifier = Modifier.height(8.dp))
}

project.youtubeVideoId?.let { videoId ->
    Spacer(modifier = Modifier.height(16.dp))

    Text(
        text = "Tutorial Video:",
        style = MaterialTheme.typography.titleLarge
    )

    // YouTube video player implementation
    // Or link to YouTube

    Spacer(modifier = Modifier.height(8.dp))

    OutlinedButton(
        onClick = {
            val intent = Intent(Intent.ACTION_VIEW, Uri.parse("https://www.youtube.com/watch?v=$videoId"))
            navController.context.startActivity(intent)
        }
    ) {
        Text("Watch on YouTube")
    }
```

```
            }

            Spacer(modifier = Modifier.height(24.dp))

            Button(
                onClick = {
                    viewModel.saveCurrentProject()
                    Toast.makeText(navController.context, "Project saved!", Toast.LENGTH_SHORT).show()
                },
                modifier = Modifier.fillMaxWidth()
            ) {
                Text("Save Project")
            }
        }
    }
}


class HardwareRecommendationEngine @Inject constructor(
    private val hardwareDatabase: HardwareDatabase ,
    @Named("ClaudeApiKey") private val apiKey: String ,
    private val httpClient: OkHttpClient
) {
    suspend fun generateRecommendations(designAnalysis: FurnitureDesignAnalysis): List<HardwareRecommendation> {
        return withContext(Dispatchers.IO) {
            try {
                Log.d("HardwareEngine", "Generating recommendations for ${designAnalysis.furnitureType}")

                // First, retrieve suitable hardware components from local database
                val localRecommendations = generateLocalRecommendations(designAnalysis)

                // Enhance recommendations with AI for complex selections
                val enhancedRecommendations = enhanceRecommendationsWithAI(designAnalysis, localRecommendations)

                Log.d("HardwareEngine", "Generated ${enhancedRecommendations.size} recommendations")
                return@withContext enhancedRecommendations
            } catch (e: Exception) {
                Log.e("HardwareEngine", "Error generating hardware recommendations", e)
                throw e
            }
```

```
    }
}

private suspend fun generateLocalRecommendations(designAnalysis: FurnitureDesignAnalysis): List<HardwareRecommendation> {
    val recommendations = mutableListOf<HardwareRecommendation>()

    // Process joints to recommend appropriate fasteners
    for (joint in designAnalysis.joints) {
        val hardwareComponents = hardwareDatabase.findHardwareForJointAndMaterials(
            jointType = joint.type,
            materials = designAnalysis.materials,
            furnitureCategory = designAnalysis.category
        )

        for (component in hardwareComponents) {
            recommendations.add(
                HardwareRecommendation(
                    component = component,
                    recommendedQuantity = joint.quantity,
                    locationDescription = joint.location,
                    recommendationReason = "Suitable for ${joint.type} joint in ${designAnalysis.materials.joinToString("/")} materials",
                    priority = determineRecommendationPriority(component, designAnalysis)
                )
            )
        }
    }

    // Process specific hardware requirements
    for (requirement in designAnalysis.hardwareRequirements) {
        val hardwareComponents = hardwareDatabase.findHardwareByType(
            type = requirement.type,
            specs = requirement.recommendedSpecs
        )

        for (component in hardwareComponents) {
            recommendations.add(
                HardwareRecommendation(
                    component = component,
                    recommendedQuantity = requirement.estimatedQuantity,
                    locationDescription = requirement.purpose,
```

```
                recommendationReason = "Matches specified requirement for ${requirement.type}",
                priority = HardwareRecommendationPriority.HIGH
            )
        )
    }
}

// Add general recommendations based on furniture type
val generalHardware = hardwareDatabase.findGeneralHardwareForFurnitureType(
    furnitureType = designAnalysis.furnitureType,
    category = designAnalysis.category
)

for (component in generalHardware) {
    recommendations.add(
        HardwareRecommendation(
            component = component,
            recommendedQuantity = estimateGeneralHardwareQuantity(component, designAnalysis),
            locationDescription = "General use in ${designAnalysis.furnitureType}",
            recommendationReason = "Commonly used in ${designAnalysis.category} furniture",
            priority = HardwareRecommendationPriority.MEDIUM
        )
    )
}

return recommendations
}

private suspend fun enhanceRecommendationsWithAI(
    designAnalysis: FurnitureDesignAnalysis,
    localRecommendations: List<HardwareRecommendation>
): List<HardwareRecommendation> {
    // Create JSON request for Claude API
    val jsonBody = JSONObject()
    jsonBody.put("model", "claude-3-opus-20240229")

    val messages = JSONArray()
    val messageObject = JSONObject()
    messageObject.put("role", "user")
```

```
// Create a comprehensive prompt with furniture analysis and initial recommendations
val prompt = """
    You are a furniture hardware specialist assistant. Review the furniture design analysis and initial hardware recommendations below and provide improved recommendations if needed.

    FURNITURE DESIGN ANALYSIS:
    Type: ${designAnalysis.furnitureType}
    Category: ${designAnalysis.category}
    Materials: ${designAnalysis.materials.joinToString(", ")}
    Joints: ${designAnalysis.joints.size} joints (${designAnalysis.joints.joinToString(", ") { it.type }})
    Dimensions: ${designAnalysis.dimensions.width}${designAnalysis.dimensions.unit} × ${designAnalysis.dimensions.height}${designAnalysis.dimensions.unit} × ${designAnalysis.dimensions.depth}${designAnalysis.dimensions.unit}
    Load-bearing elements: ${designAnalysis.loadBearingElements.joinToString(", ")}
    Aesthetic considerations: ${designAnalysis.aestheticConsiderations.joinToString(", ")}

    INITIAL HARDWARE RECOMMENDATIONS:
    ${localRecommendations.joinToString("\n") {
    "${it.component.name} (${it.component.category}): ${it.recommendedQuantity} units for ${it.locationDescription}"
}}

    Based on your expert knowledge of furniture hardware, please:
    1. Review the initial recommendations and identify any missing critical hardware
    2. Suggest improvements or alternatives to the initial recommendations
    3. Verify quantities are appropriate for this design
    4. Consider aesthetic requirements and suggest appropriate finishes
    5. Add any warnings or special considerations for installation

    Format your response as a JSON array of hardware recommendations with this structure:
    [
      {
        "componentName": "string",
        "componentCategory": "string",
        "recommendedQuantity": number,
        "locationDescription": "string",
        "recommendationReason": "string",
        "priority": "HIGH/MEDIUM/LOW",
        "alternativeOptions": ["string"],
        "installationNotes": "string"
      }
```

```
            ]

            Focus on providing practical, industry-standard recommendations that would be available from standard suppliers.
        """.trimIndent()

        messageObject.put("content", prompt)
        messages.put(messageObject)
        jsonBody.put("messages", messages)
        jsonBody.put("max_tokens", 4096)

        val requestBodyString = jsonBody.toString()

        val request = Request.Builder()
            .url("https://api.anthropic.com/v1/messages")
            .addHeader("x-api-key", apiKey)
            .addHeader("anthropic-version", "2023-06-01")
            .addHeader("Content-Type", "application/json")
            .post(requestBodyString.toRequestBody("application/json".toMediaType()))
            .build()

        val response = httpClient.newCall(request).execute()

        if (response.isSuccessful) {
            val responseBody = response.body?.string()
            return parseEnhancedRecommendations(responseBody ?: "", localRecommendations)
        } else {
            Log.e("HardwareEngine", "API error: ${response.body?.string()}")
            // Fall back to local recommendations if AI enhancement fails
            return localRecommendations
        }
    }

    private suspend fun parseEnhancedRecommendations(
        responseJson: String,
        fallbackRecommendations: List<HardwareRecommendation>
    ): List<HardwareRecommendation> {
        try {
            val jsonObject = JSONObject(responseJson)
            val content = jsonObject.getJSONArray("content")
```

```
// Extract the text content from the response
var jsonString = ""
for (i in 0 until content.length()) {
    val item = content.getJSONObject(i)
    if (item.getString("type") == "text") {
        jsonString = item.getString("text")
        break
    }
}

// Extract the JSON array from the text content
val jsonPattern = Pattern.compile("\\[.*\\]", Pattern.DOTALL)
val matcher = jsonPattern.matcher(jsonString)

if (matcher.find()) {
    val recommendationsJson = JSONArray(matcher.group(0))
    val enhancedRecommendations = mutableListOf<HardwareRecommendation>()

    for (i in 0 until recommendationsJson.length()) {
        val recJson = recommendationsJson.getJSONObject(i)

        // Create or find a matching hardware component
        val componentName = recJson.getString("componentName")
        val componentCategory = recJson.getString("componentCategory")

        val component = hardwareDatabase.findHardwareByNameAndCategory(
            name = componentName,
            category = componentCategory
        ) ?: createGenericComponent(componentName, componentCategory)

        // Parse alternative options
        val alternativesArray = recJson.optJSONArray("alternativeOptions")
        val alternatives = mutableListOf<String>()
        if (alternativesArray != null) {
            for (j in 0 until alternativesArray.length()) {
                alternatives.add(alternativesArray.getString(j))
            }
        }

        // Create recommendation
```

```
                enhancedRecommendations.add(
                    HardwareRecommendation(
                        component = component,
                        recommendedQuantity = recJson.getInt("recommendedQuantity"),
                        locationDescription = recJson.getString("locationDescription"),
                        recommendationReason = recJson.getString("recommendationReason"),
                        priority = HardwareRecommendationPriority.valueOf(
                            recJson.getString("priority").uppercase()
                        ),
                        alternativeOptions = alternatives,
                        installationNotes = recJson.optString("installationNotes", "")
                    )
                )
            }

            return enhancedRecommendations
        } else {
            Log.e("HardwareEngine", "No valid JSON array found in response")
            return fallbackRecommendations
        }
    } catch (e: Exception) {
        Log.e("HardwareEngine", "Error parsing enhanced recommendations", e)
        return fallbackRecommendations
    }
}

private fun determineRecommendationPriority(
    component: HardwareComponent,
    designAnalysis: FurnitureDesignAnalysis
): HardwareRecommendationPriority {
    // Implementation of priority determination logic
    return when {
        component.category == "Structural Fastener" -> HardwareRecommendationPriority.HIGH
        component.category == "Hinge" && designAnalysis.furnitureType.contains("Cabinet") ->
            HardwareRecommendationPriority.HIGH
        component.category == "Handle" || component.category == "Knob" ->
            HardwareRecommendationPriority.MEDIUM
        else -> HardwareRecommendationPriority.LOW
    }
}
```

```
    private fun estimateGeneralHardwareQuantity(
        component: HardwareComponent,
        designAnalysis: FurnitureDesignAnalysis
    ): Int {
        // Estimation logic based on furniture dimensions and type
        return when (component.category) {
            "Screw" -> (designAnalysis.joints.size * 2).coerceAtLeast(4)
            "Hinge" -> if (designAnalysis.furnitureType.contains("Cabinet")) 2 else 0
            "Handle" -> if (designAnalysis.furnitureType.contains("Drawer")) 1 else 0
            "Shelf Pin" ->
                if (designAnalysis.furnitureType.contains("Bookcase") ||
                    designAnalysis.furnitureType.contains("Cabinet")) 4 else 0
            else -> 1
        }
    }

    private fun createGenericComponent(name: String, category: String): HardwareComponent {
        return HardwareComponent(
            id = UUID.randomUUID().toString(),
            name = name,
            category = category,
            specifications = mapOf("Type" to category),
            compatibleMaterials = listOf("Wood", "Particleboard", "MDF"),
            loadCapacity = 0.0f,
            finishes = listOf("Standard"),
            imageUrl = "",
            unitPrice = 0.0,
            supplier = "Generic"
        )
    }
}
```

Додаток Г – Ілюстративна частина

ІЛЮСТРАТИВНА ЧАСТИНА

РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ АВТОМАТИЗАЦІЇ КОМПЛЕКТУВАННЯ ФУРНІТУРОЮ МЕБЛІВ НА ЕТАПІ ПРОЕКТУВАННЯ

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення

Бакалаврська кваліфікаційна робота на тему:
«Розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування»

Автор: студент групи 2ПІ-21б Колос В.В.
Науковий керівник: к.т.н., доц. каф. ПЗ Рейда О.М.

м. Вінниця - 2025

Рисунок Г.1 — Титульний слайд

Актуальність теми

Більшість наявних програмних застосунків фокусуються на моделюванні геометрії виробів та візуалізації, залишаючи питання підбору функціональних компонентів на розсуд користувача. Рішення, які пропонують функціональність для роботи з фурнітурою, зазвичай є частиною комплексних систем автоматизованого проєктування, що вимагають значних обчислювальних ресурсів та не пристосовані для використання на мобільних пристроях.

Автоматизація процесу комплектування фурнітурою на етапі проєктування дозволяє оперативно реагувати на потреби ринку та пропонувати клієнтам оптимальні рішення з точки зору функціональності, естетики та вартості.

Враховуючи вищезазначені фактори, технологічні обмеження традиційних методів, економічні переваги автоматизації, зростаючі вимоги ринку та обмеженість існуючих програмних застосунків, актуальною є розробка мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування.

Рисунок Г.2 — Актуальність теми

Мета, об’єкт та предмет дослідження

Метою роботи є підвищення ефективності організації виробничого процесу за отриманими рекомендаціями та інструкціями шляхом автоматизації комплектування фурнітурою меблів на етапі проектування.
Об'єкт дослідження – процес розробки мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування.
Предмет дослідження – методи та засоби розробки мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування.

Рисунок Г.3 — Мета, об’єкт та предмет дослідження

Завдання дослідження

Основними задачами дослідження є:

- розробити архітектуру мобільного застосунку для автоматизації комплектування фурнітурою меблів на етапі проектування;
- модифікувати метод оптимізації вибору меблевих комплектуючих для підвищення ефективності формування списку фурнітури меблів на етапі проектування;
- розробити інтерфейс користувача застосунку;
- розробити алгоритми роботи застосунку;
- розробити функціонал застосунку;
- провести тестування розробленого застосунку.

Рисунок Г.4 — Завдання дослідження

Наукова новизна

Подальшого розвитку отримав метод оптимізації вибору меблевих комплектуючих на зображенні, що на відміну від існуючого, використовує модель claude-3-opus ШІ для визначення комплектуючих, і формування їх специфікацій, і як наслідок, зменшує час підготовки для виготовлення продукції.

Рисунок Г.5 — Наукова новизна

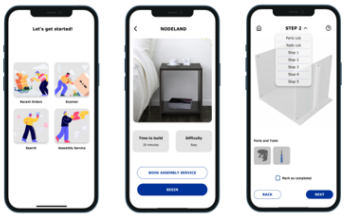
Практична цінність отриманих результатів

Практична цінність одержаних результатів полягає в:

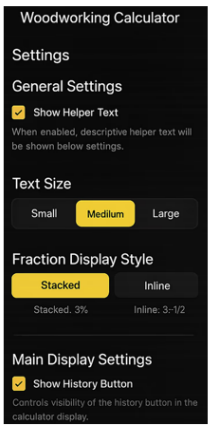
- автоматизації процесу комплектування фурнітурою меблів на етапі проектування;
- розроблено мобільний застосунок;
- функціональні модулі;
- класи для використання в процесі розпізнавання зображень з меблевою фурнітурою.

Рисунок Г.6 — Практична цінність отриманих результатів

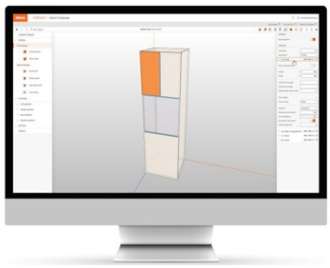
Аналоги



IKEA Assemble App



Woodworking Calculator



Blum Configurator

Рисунок Г.7 — Аналоги

Порівняльний аналіз аналогів

Характеристика	IKEA Assemble App	Woodworking Calculator Pro	Blum Configurator	Власна розробка
Персоналізовані рекомендації	0	0	1	1
Експорт специфікацій	1	1	1	1
Розпізнавання комплектуючих та меблів за фото	0	0	0	1
Виявлення відсутніх комплектуючих	1	0	0	1
Підбір необхідних комплектуючих	1	1	1	1
Детальні інструкції зі збірки меблів	1	1	1	1
Сумарний коефіцієнт	4	3	4	6

Рисунок Г.8 — Порівняльний аналіз аналогів

Використані технології

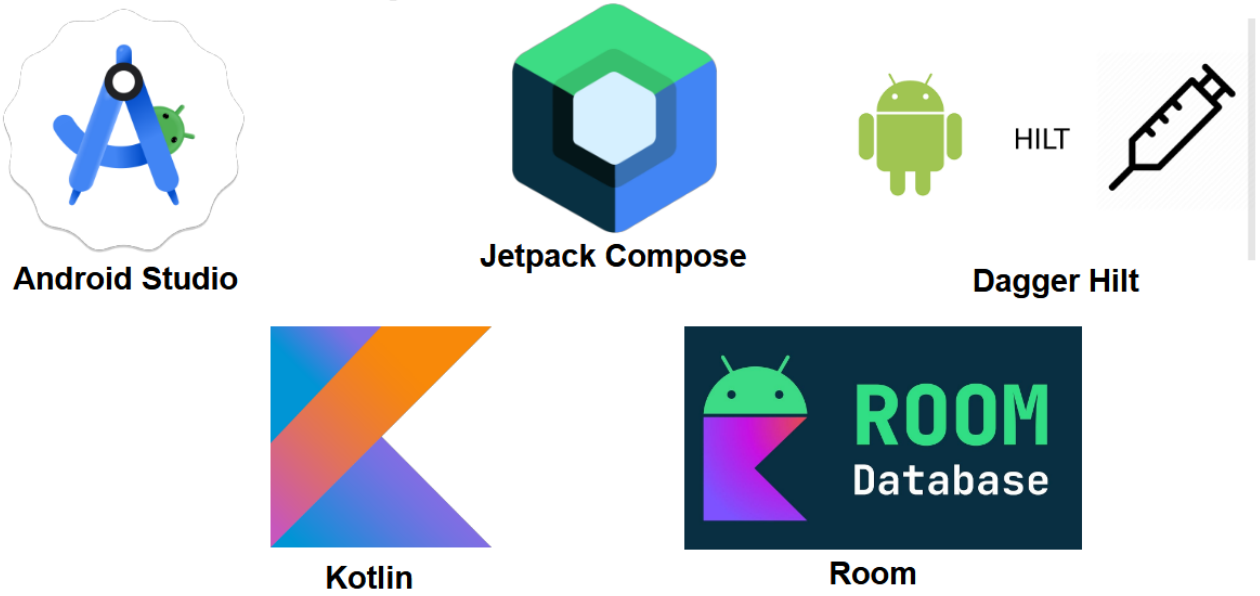


Рисунок Г.9 — Використані технології

Метод ідентифікації та адаптації інструкцій на основі зображення наявних інструментів та матеріалів.

1. Збір даних про наявні інструменти та матеріали.

2. Використання API Claude Vision для аналізу та ідентифікації об'єктів на фотографії:

3. Формування детальних інструкцій шляхом створення запиту до API Claude

4. Формування персоналізованих інструкцій

5. Отримані інструкції обробляються системою та відображаються користувачу.

Рисунок Г.10 – Метод ідентифікації та адаптації інструкцій на основі зображення наявних інструментів та матеріалів

Алгоритм оптимізації вибору меблевих комплектуючих на зображенні

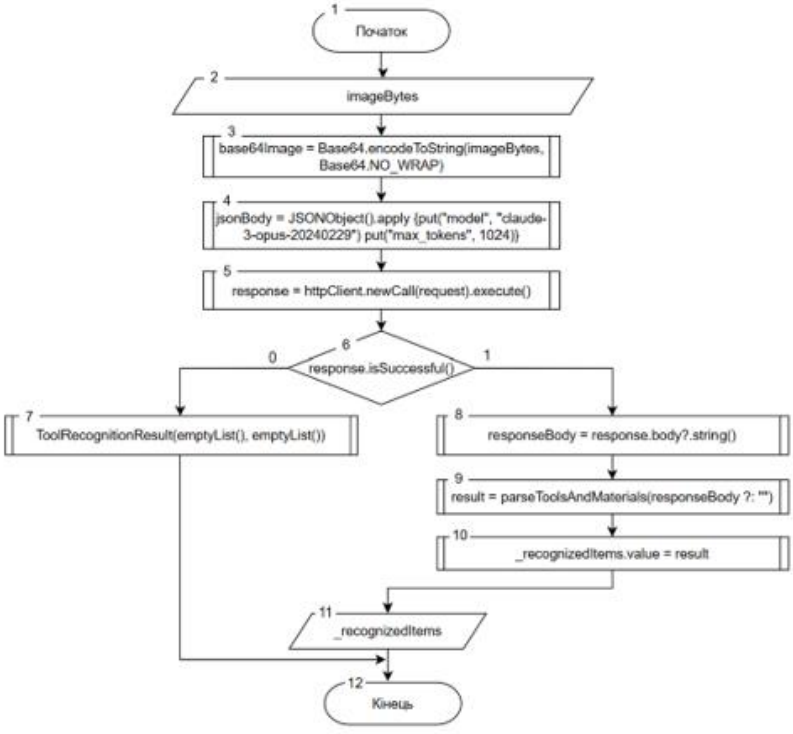


Рисунок Г.11 — Блок-схема роботи алгоритму оптимізації вибору меблевих комплектуючих на зображенні

База даних

Рисунок Г.12 — База даних

Тестування застосунку

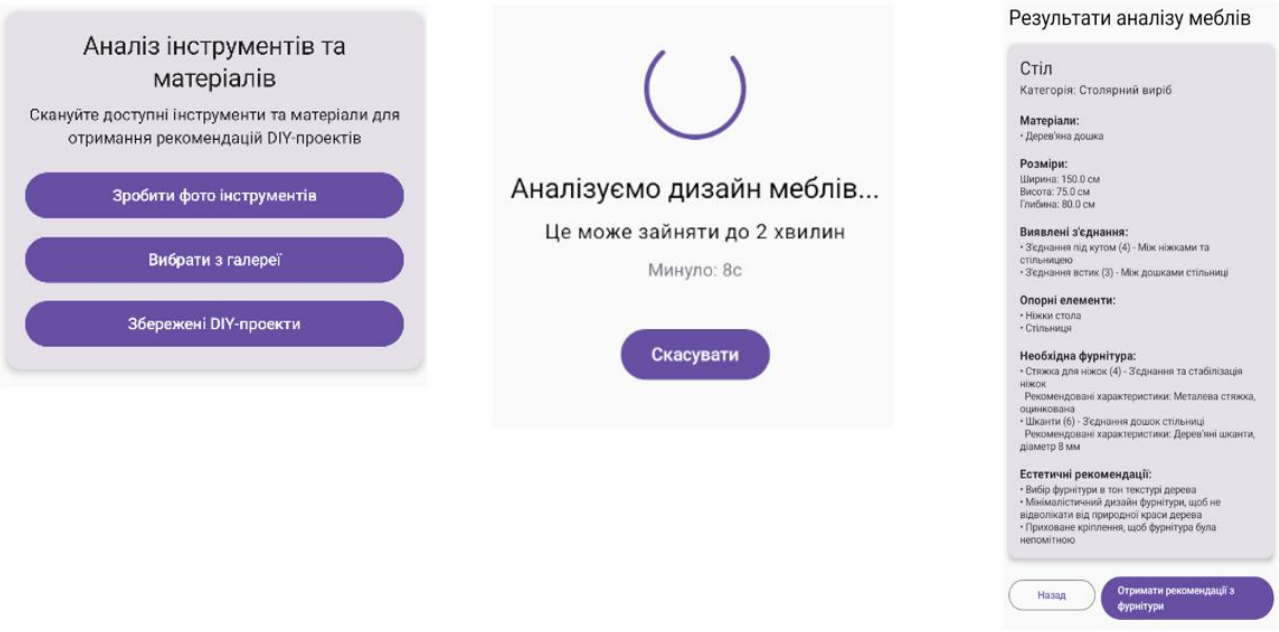


Рисунок Г.13 — Тестування аналізу

Тестування застосунку

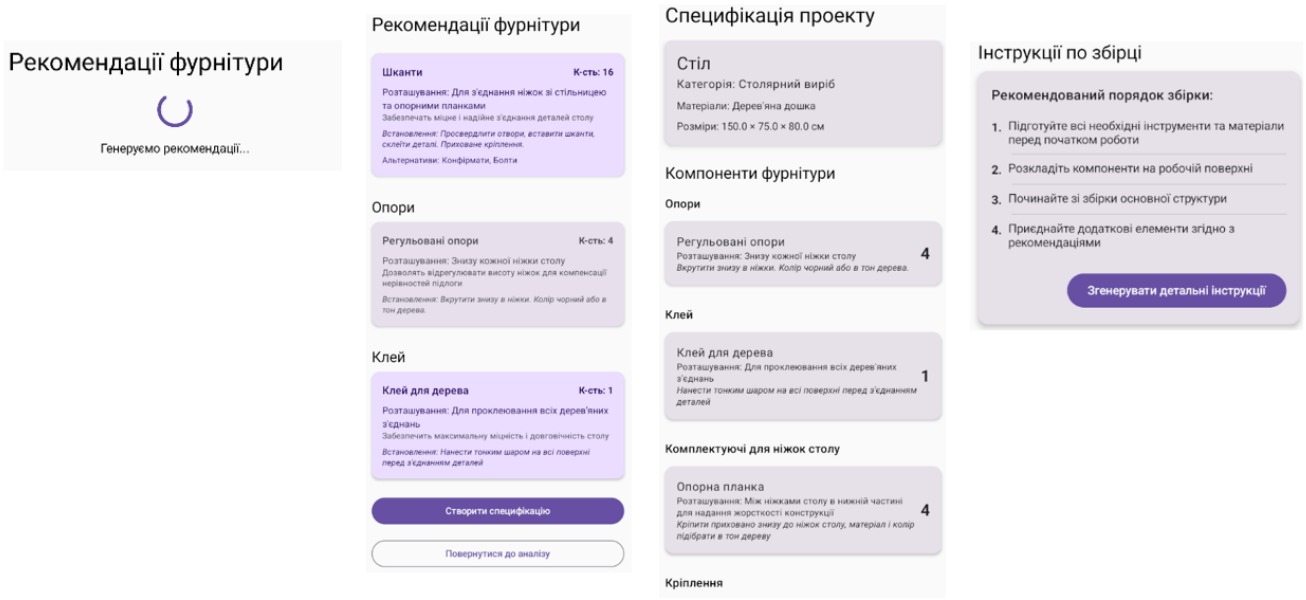


Рисунок Г.14 — Тестування специфікації меблевого проекту

Тестування застосунку

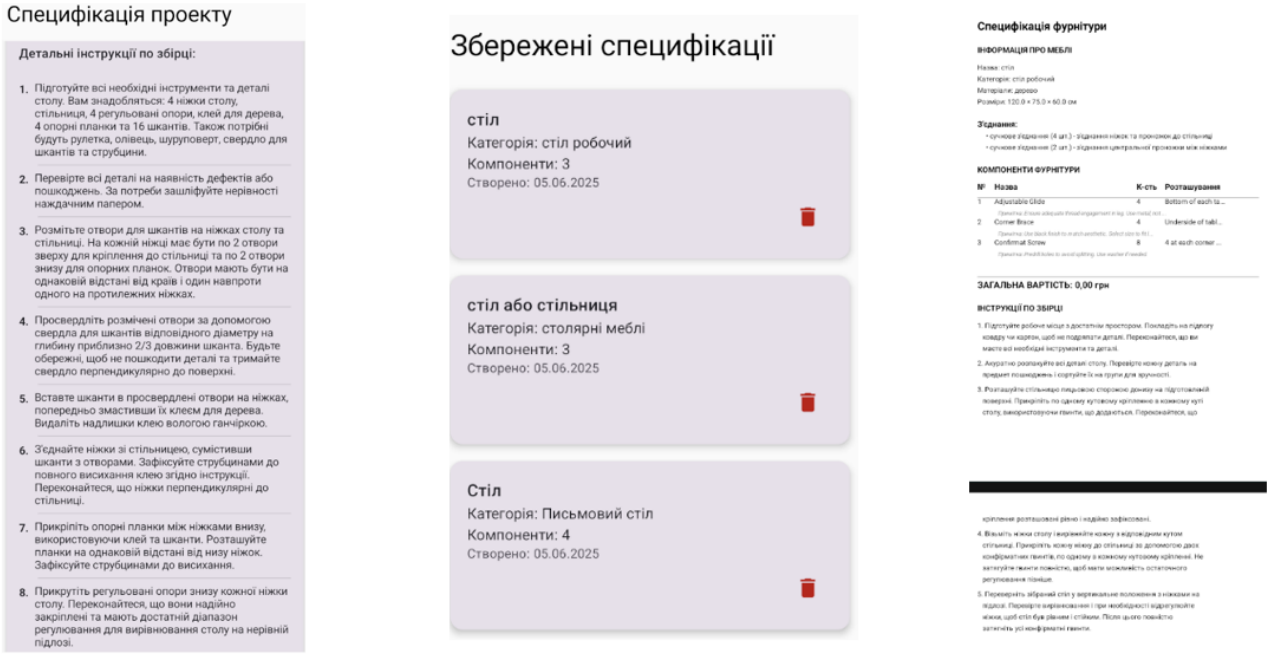


Рисунок Г.15 — Тестування формування інструкції та експорту специфікації

Висновки

У результаті виконання бакалаврської кваліфікаційної роботи було розроблено мобільний застосунок для автоматизації комплектування фурнітурою меблів на етапі проектування.

- Розроблено архітектуру мобільного застосунку, що відповідає сучасним підходам.
- Модифіковано метод оптимізації вибору меблевих комплектуючих для підвищення ефективності формування списку фурнітури меблів на етапі проектування з використанням моделі claude-3-opus ШІ для визначення комплектуючих, і формування їх специфікацій.
- Розроблено інтерфейс користувача, побудовано структурні схеми.
- Розроблено алгоритми роботи застосунку, які дозволяють розпізнавати інструменти і матеріали, генерувати рекомендацій проєктів, інструкції та шукати відеоінструкції.
- Проведено тестування застосунку, в ході якого продемонстровано його функціонал, наведено приклад використання, та доведено, що розроблена система виконує поставлені на початку розробки завдання.

Рисунок Г.16 — Висновки

Апробація результатів роботи і публікації

Апробація матеріалів. Описані у курсовому проєкті положення доповідались на конференції «LIV Всеукраїнська науково-технічна конференція підрозділів Вінницького національного технічного університету (НТКП ВНТУ–2025)».
Публікації. В.В. Колос, Г.Б. Ракитянська. Розробка мобільного асистента для підбору комплектуючих частин виробничих проектів / Матеріали LIV Всеукраїнської науково-технічної конференції підрозділів Вінницького національного технічного університету (НТКП ВНТУ–2025) : збірник доповідей. URL: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2025/paper/view/23713/19726

Рисунок Г.18 — Апробація результатів роботи і публікації