

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

«Онлайн-сервіс 3D-друку з інтеграцією AI асистента»

Виконала студентка групи ІПЗ-22-4
ШКОЛЬНА Арина Леонідівна

Керівник роботи:
САВІЦЬКИЙ Роман Святославович

Рецензент:
ВАКАЛЮК Тетяна Анатоліївна

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення
Тетяна Вакалюк
«28» лютого 2026 р.

ЗАВДАННЯ

на кваліфікаційну роботу

Здобувач вищої освіти: **ШКОЛЬНА Арина Леонідівна**

Керівник роботи: **САВІЦЬКИЙ Роман Святославович**

Тема роботи: **«Онлайн-сервіс 3D-друку з інтеграцією AI асистента»,**

затверджена Наказом закладу вищої освіти від «27» лютого 2026 р., № 92/с

Вихідні дані для роботи: Об'єктом дослідження є процес автоматизованої обробки замовлень на тривимірний друк, що охоплює аналіз геометричних характеристик моделей. Предметом дослідження є методи та алгоритми аналізу тривимірних моделей, що включають нейромережеву модель класифікації та регресії параметрів друку, систему виявлення геометричних дефектів моделі з інтеграцією результатів аналізу з процесом підготовки моделі до фізичного виконання.

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Савіцький Р.С.	01.03.2026р.	01.03.2026р
2	Савіцький Р.С.	01.03.2026р	01.03.2026р
3	Савіцький Р.С.	01.03.2026р	01.03.2026р

РЕФЕРАТ

Структура та обсяг роботи. Випускна робота складається з вступу, 3-х розділів, висновків, списку використаних джерел із 32 найменувань, 6 сторінок додатків, 7 таблиць та 25 рисунків по тексту. Повний обсяг роботи становить 91 сторінку, з них 85 сторінок основного тексту.

Метою роботи є проектування та програмна реалізація інтелектуальної вебплатформи «FABBLU» для автоматизації робочих процесів онлайн-сервісу 3D-друку, що забезпечує технічний аудит моделей за допомогою AI-асистента та хмарне керування парком принтерів.

У ході дослідження застосовано методи системного аналізу предметної області, об'єктно-орієнтованого проектування з використанням CASE-технологій, а також методи вебпрограмування (стек React, Node.js, PostgreSQL). Апаратна частина реалізована з використанням 3D-принтера Bambu Lab A1, інтегрованого в систему через мережеві протоколи передачі даних.

Розроблено вебплатформу, яка автоматизує повний цикл виробництва: від інтелектуальної перевірки геометрії STL-файлів на «друкованість» до прямої передачі G-коду на обладнання Bambu Lab. Система реалізує динамічний розрахунок вартості на основі AI-прогнозів, складський облік матеріалів за вагою та моніторинг телеметрії принтерів у реальному часі.

Результати роботи рекомендовані до впровадження у діяльність комерційних студій 3D-друку та інженерних центрів швидкого прототипування. Впровадження системи дозволяє знизити операційні витрати на підготовку файлів та мінімізувати ризики виникнення браку.

Ключові слова: 3D-ДРУК, АВТОМАТИЗАЦІЯ, ВЕБОРІЄНТОВАНА СИСТЕМА, МОНІТОРИНГ, ШТУЧНИЙ ІНТЕЛЕКТ.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		3

ABSTRACT

Structure and scope of the work. The qualification thesis consists of an introduction, 3 chapters, conclusions, a list of references comprising 32 titles, 6 pages of appendices, 7 tables and 25 figures throughout the text. The total scope of the work amounts to 91 pages, of which 85 pages constitute the main text.

The aim of the work is the design and software implementation of the intelligent web platform "FABBL Y" for automating the workflows of an online 3D printing service, which provides technical audit of models by means of an AI assistant and cloud-based management of a fleet of printers.

In the course of the research, methods of systems analysis of the subject domain, object-oriented design using CASE technologies, as well as web development methods (the React, Node.js, PostgreSQL stack) were applied. The hardware component is implemented using a Bambu Lab A1 3D printer, integrated into the system via network data transfer protocols.

A web platform has been developed that automates the full production cycle: from intelligent verification of STL file geometry for "printability" to direct transfer of G-code to Bambu Lab equipment. The system implements dynamic cost calculation based on AI predictions, weight-based inventory management of materials, and real-time monitoring of printer telemetry.

The results of the work are recommended for implementation in the activities of commercial 3D printing studios and rapid prototyping engineering centers. The implementation of the system allows reducing operational costs of file preparation and minimizing the risks of defective output.

Keywords: 3D PRINTING, AUTOMATION, WEB-BASED SYSTEM, MONITORING, ARTIFICIAL INTELLIGENCE.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

РЕФЕРАТ.....	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	6
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ	10
1.1 Постановка задачі.....	10
1.2 Аналіз аналогів програмного продукту	12
1.3 Обґрунтування вибору архітектури онлайн-сервісу 3D-друку	17
1.4 Обґрунтування вибору інструментальних засобів	20
Висновки до першого розділу.....	23
РОЗДІЛ 2. ПРОЄКТУВАННЯ ОНЛАЙН-СЕРВІСУ 3D-ДРУКУ З ІНТЕГРАЦІЄЮ АІ АСИСТЕНТА.....	25
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи.....	25
2.2 Розробка бази даних системи.....	34
2.3 Проєктування та реалізація алгоритмів роботи системи	41
2.4 Реалізація програмного додатку	51
Висновки до другого розділу	57
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ОНЛАЙН-СЕРВІСОМ 3D-ДРУКУ З ІНТЕГРАЦІЄЮ АІ АСИСТЕНТА	59
3.1 Порядок встановлення та налаштування системи	59
3.2 Структура інтерфейсу користувача.....	62
3.3 Тестування роботи програмного комплексу	72
Висновки до третього розділу.....	76
ВИСНОВКИ	78
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	81
ДОДАТКИ.....	85
Додаток А.....	86

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		5

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних

ПЗ – програмне забезпечення

AI (Artificial Intelligence) – штучний інтелект

API (Application Programming Interface) – інтерфейс прикладного програмування

CASE (Computer-Aided Software Engineering) – автоматизоване проєктування програмного забезпечення

CLI (Command Line Interface) – інтерфейс командного рядка

FDM (Fused Deposition Modeling) – моделювання методом пошарового наплавлення

FTPS (File Transfer Protocol Secure) – захищений протокол передачі файлів

JSON (JavaScript Object Notation) – об'єктна нотація JavaScript, формат обміну даними

JWT (JSON Web Token) – токен для автентифікації та авторизації

MQTT (Message Queuing Telemetry Transport) – протокол передачі телеметричних повідомлень

MQTTS (MQTT Secure) – захищена версія протоколу MQTT

OAuth (Open Authorization) – відкритий протокол делегованої авторизації

ORM (Object-Relational Mapping) – об'єктно-реляційне відображення

REST (Representational State Transfer) – архітектурний стиль побудови мережесервісів

SLA (Stereolithography Apparatus) – технологія 3D-друку на основі фотополімеризації рідкої смоли

SLS (Selective Laser Sintering) – технологія 3D-друку на основі вибіркового лазерного спікання порошкових матеріалів

SPA (Single Page Application) – односторінковий вебзастосунок

SQL (Structured Query Language) – структурована мова запитів до реляційних баз даних

STL (Stereolithography) – формат файлу для зберігання тривимірних моделей

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВСТУП

Станом на сьогодні цифрові технології стрімко проникають у всі сфери діяльності людини, зокрема у виробництво та сферу послуг. Одним із перспективних напрямів є 3D-друк, який активно використовується для створення прототипів, індивідуальних виробів та малосерійного виробництва. Зростання популярності онлайн-сервісів 3D-друку зумовлює потребу в ефективних інформаційних системах, здатних автоматизувати робочі процеси та покращити взаємодію між клієнтами й виконавцями.

У сучасних умовах автоматизація процесів прийому замовлень, перевірки 3D-моделей, розрахунку вартості, керування статусами замовлень та підготовки до друку є важливим чинником підвищення ефективності роботи онлайн-сервісів 3D-друку. Інтеграція AI-асистента в такі системи відкриває нові можливості, зокрема автоматичний аналіз моделей, консультацію користувачів, виявлення помилок у файлах та рекомендації щодо оптимальних параметрів друку.

Онлайн-сервіси 3D-друку дозволяють користувачам дистанційно завантажувати власні моделі, обирати матеріали та параметри друку, а також відстежувати стан виконання замовлення в реальному часі. Для операторів 3D-друку такі системи забезпечують централізоване керування замовленнями, оптимізацію робочого навантаження та зменшення кількості ручних операцій. Використання інтелектуальних підказок AI-асистента сприяє зниженню кількості помилок і підвищенню якості кінцевого продукту.

Актуальність теми полягає у необхідності мінімізації людського фактора при підготовці 3D-моделей до виробництва. Впровадження інтелектуального AI-асистента дозволяє автоматизувати технічний аудит STL/OBJ файлів, здійснювати предиктивний аналіз придатності геометрії до друку та надавати персоналізовані рекомендації щодо вибору філаменту, що суттєво підвищує ефективність онлайн-сервісів.

Метою роботи є проектування та розробка комплексної моделі автоматизованої системи керування онлайн-сервісом 3D-друку з інтегрованим AI-модулем та системою моніторингу обладнання в реальному часі.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

Завданнями роботи є:

- аналіз теоретичних засад моделювання програмного забезпечення;
- формування та систематизація вимог користувачів (клієнтів, операторів, менеджерів та адміністраторів) з використанням CASE-технологій;
- розробка інтелектуального модуля AI-аналізу для автоматичної перевірки геометричних параметрів моделей та динамічного розрахунку вартості замовлень;
- проєктування об'єктно-орієнтованої структури бази даних та архітектури серверної частини для підтримки гібридних замовлень (каталог та кастомні моделі);
- розробка алгоритмів хмарної підготовки моделей (slicing) та механізмів інтеграції з обладнанням Bambu Lab через протоколи MQTT для передачі G-коду та моніторингу телеметрії;
- створення адаптивного інтерфейсу користувача з використанням Three.js для 3D-візуалізації об'єктів у браузері;
- програмна реалізація та тестування функціональних модулів системи: від авторизації на основі JWT до складського обліку розхідних матеріалів.

Об'єктом дослідження є процеси архітектурного проєктування та програмної реалізації автоматизованих систем у сфері адитивного виробництва.

Предметом дослідження є методи та засоби моделювання програмного забезпечення, алгоритми інтеграції штучного інтелекту для аналізу тривимірних об'єктів та протоколи взаємодії з апаратними комплексами 3D-друку.

Методи дослідження. У роботі було використано методи системного аналізу для вивчення предметної області, методи об'єктно-орієнтованого проєктування (UML) для побудови архітектури, а також методи вебпрограмування та API-інтеграції для створення програмного продукту. Моделювання функціональних складових здійснювалося з використанням сучасних CASE-технологій.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

Апробація результатів дослідження. Основні положення та результати кваліфікаційної роботи були представлені, обговорені та опубліковані у матеріалах науково-практичних конференцій [1, 2].

Практичне значення одержаних результатів полягає у створенні готової до впровадження інтелектуальної платформи «FABBL Y», яка дозволяє автоматизувати повний цикл роботи онлайн-сервісу 3D-друку. Розроблені рішення забезпечують зниження операційних витрат на переддрукарську підготовку, зменшують кількість браку за рахунок предиктивного аналізу AI та надають інструменти для масштабування бізнесу без пропорційного збільшення штату операторів.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

РОЗДІЛ 1. АНАЛІЗ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

1.1 Постановка задачі

Сьогодні головна проблема більшості студій 3D-друку полягає в тому, що всі процеси розірвані між собою. Клієнт пише в один месенджер, скидає файл в інший, а майстер-оператор взагалі працює окремо біля принтерів. Постановка задачі в цій роботі – створити єдину платформу, де замовлення, перевірка файлів і керування принтерами будуть відбуватися в одному місці. Це дозволить перетворити звичайний чат із клієнтом на автоматизований сервіс, де більшість технічних питань вирішує сама програма.

Зараз робочий процес виглядає так: клієнт присилає модель, менеджер вручну відкриває її в спеціальних програмах, перевіряє, чи не занадто тонкі там стінки, рахує ціну на калькуляторі й пише відповідь. Якщо клієнтів стає багато, менеджер починає плутатися, забуває статус замовлення або не бачить, що на складі закінчився потрібний пластик. Через це виникають затримки, а інколи й помилки, коли деталь друкується не тим кольором або взагалі ламається під час друку через неправильні налаштування.

Окремої технічної складності вимагає перевірка геометрії 3D-моделей, яку наразі в більшості студій виконує людина вручну для кожного завантаженого STL-файлу. AI-асистент повинен брати цю перевірку на себе, аналізувати модель на здатність витримати навантаження, визначати потребу в додаткових підтримках для друку та підказувати, який матеріал підійде краще під конкретну задачу. Завантажена клієнтом модель має отримувати такий аналіз одразу після завантаження, без очікування реакції оператора чи менеджера.

Розрахунок вартості також має відбуватися автоматично, на основі ваги моделі, орієнтовного часу друку та вартості електроенергії, що дозволяє клієнту побачити підсумкову ціну і оплатити замовлення без попереднього дзвінка чи листування з менеджером. Паралельно з розрахунком ціни система повинна перевіряти залишки філаменту на складі, щоб виключити ситуацію, коли клієнту

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

підтверджують замовлення в кольорі чи матеріалі, якого фактично вже немає в наявності.

Саме тому постановка задачі передбачає створення інтелектуального модуля управління ресурсами (складом). 3D-друк – це матеріаломісткий процес, де кожна катушка пластику має свою вагу, ціну та колір. Проблема багатьох сервісів у тому, що вони дізнаються про відсутність потрібного пластику вже тоді, коли принтер зупинився посеред ночі. Програмний комплекс має вести автоматичний облік залишків: після кожного завершеного друку система повинна віднімати вагу витраченого матеріалу від віртуального залишку конкретної катушки. Якщо рівень пластику опускається нижче критичної межі, платформа має заздалегідь сповістити менеджера про необхідність закупівлі. Це дозволяє підтримувати безперервний цикл виробництва та гарантувати клієнту, що обраний ним колір дійсно є в наявності.

Ще одна велика частина роботи – це зв'язок із самими принтерами, зокрема з моделлю Bambu Lab A1. Задача полягає в тому, щоб оператору не потрібно було бігати до кожного принтера з флешкою. Платформа має дозволяти відправляти файли на друк прямо з сайту. Більше того, оператор повинен бачити через інтернет, що відбувається з принтером: яка зараз температура, чи не сталося помилки й скільки відсотків роботи вже готово.

Для клієнтів має бути зручний кабінет, де вони зможуть покрутити свою модель у 3D прямо в браузері та побачити всі етапи роботи над замовленням. А для адміна – зручну панель, де видно всю статистику: скільки замовлень у черзі, які принтери зараз працюють і скільки грошей заробив сервіс за місяць.

Окремим важливим аспектом роботи є забезпечення фінансової прозорості та безпеки розрахунків. У традиційних схемах замовлення клієнти часто стикаються з необхідністю скидати скріншоти квитанцій у чати, що сповнює облік хаосом і створює ризики для обох сторін. Задача платформи – інтегрувати сучасний платіжний шлюз, наприклад LiqPay, безпосередньо в процес оформлення замовлення. Це дозволить автоматизувати не тільки прийом оплати, а й зміну статусів замовлення: як тільки кошти надходять на рахунок, система

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

сама переводить замовлення в чергу на друк, не чекаючи підтвердження від менеджера. Для кастомних замовлень, де AI-асистент виявив технічну складність, система має підтримувати механізм часткової передоплати (депозиту), що страхує сервіс від збитків у разі відмови клієнта від замовлення після початку друку.

Окрім технічних функцій, перед розробником стоїть задача забезпечення конфіденційності та безпеки даних. Клієнти часто завантажують унікальні інженерні розробки або авторські моделі, які є інтелектуальною власністю. Тому система має гарантувати, що доступ до цих файлів мають лише авторизовані оператори та адміністратори, а після виконання замовлення файли можуть бути автоматично видалені або переміщені в захищений архів. Це створює репутацію надійного сервісу, якому можна довірити складні проекти.

Підсумовуючи, основна задача – розробити таку систему, яка б зробила роботу сервісу 3D-друку простою і зрозумілою. Програма має взяти на себе всю рутину: перевірку файлів, розрахунки та стеження за принтерами. Це звільнить час людей для складніших задач і зробить сервіс набагато зручнішим для звичайних користувачів.

1.2 Аналіз аналогів програмного продукту

Для того, щоб зрозуміти, які саме функції будуть корисними в платформі «FABBL Y» та чим вона зможе виділитися серед конкурентів, необхідно детально розглянути існуючі рішення на українському ринку послуг 3D-друку. На сьогодні в Україні працює багато студій, проте більшість із них використовують свої вебсайти лише як інформаційні візитки. Це означає, що клієнт може прочитати про доступні матеріали та подивитися приклади робіт, але сам процес замовлення залишається складним і потребує постійного спілкування з менеджером.

Одним із відомих сервісів на ринку є Easy3DPrint [3]. Інтерфейс якого зображений на рисунку 1.1. Ця студія пропонує дуже широкий вибір технологій, включаючи не лише стандартний FDM-друк, а й складніші методи, такі як SLA та SLS. Сайт виглядає сучасно та лаконічно, що є безперечною перевагою для

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

залучення нових клієнтів. Компанія також надає послуги 3D-сканування та зворотного інжинірингу, що робить її повноцінним центром розробки. Однак, з точки зору автоматизації, сервіс має суттєві недоліки. Основна проблема полягає у відсутності повноцінного автоматичного калькулятора, який би працював у реальному часі. Клієнт завантажує файл, але часто змушений чекати, поки менеджер вручну перевірить його та напише остаточну ціну. Також у системі немає повноцінного особистого кабінету, де користувач міг би бачити історію своїх замовлень або на якому саме етапі знаходиться друк його деталі прямо зараз.

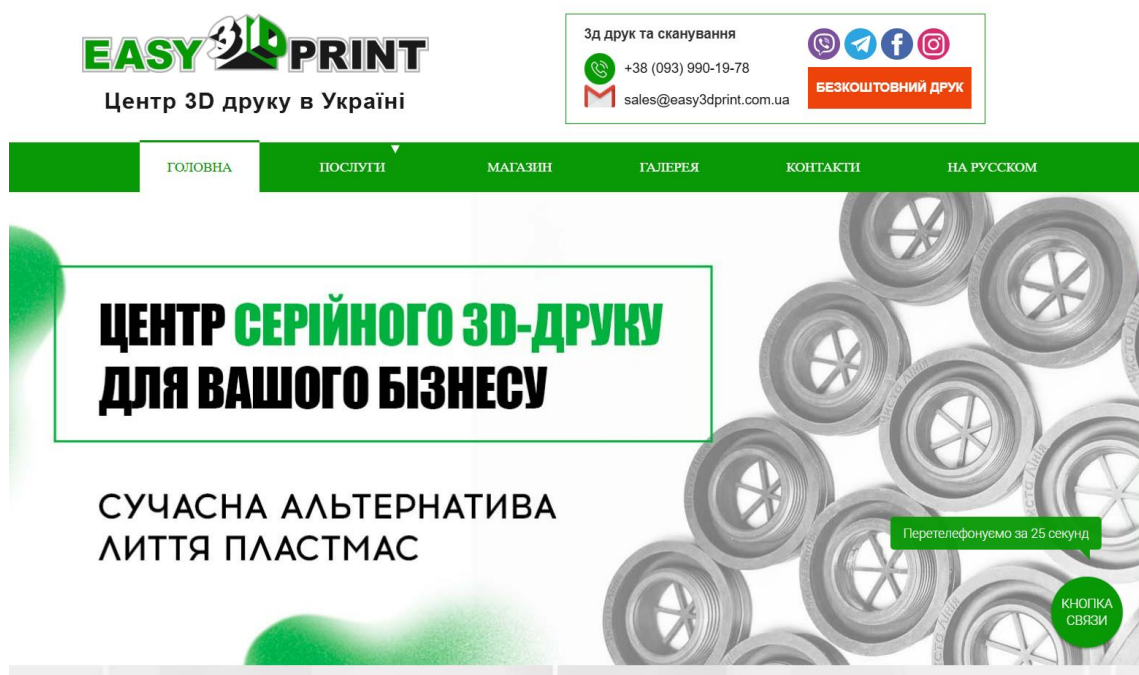


Рис.1.1. Інтерфейс вебсайту Easy3DPrint.

Іншим прикладом є компанія 3DTECHGROUP [4], яка більше орієнтована на промисловий сектор та продаж професійного обладнання. Вони мають величезний досвід у роботі з інженерними матеріалами та можуть забезпечити дуже високу точність друку. Це важливо для корпоративних клієнтів, яким потрібні складні деталі для обладнання. Проте для звичайного користувача, якому потрібно швидко надрукувати один прототип, сайт цієї компанії може здатися занадто заплутаним. Він перевантажений інформацією про продаж принтерів та запчастин (рис. 1.2), що відволікає від основної мети – замовлення друку. Процес оформлення тут є доволі застарілим і переважно відбувається

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		13

через електронну пошту або заповнення довгих статичних форм, що в сучасному цифровому світі сприймається як бар'єр для швидкого замовлення.

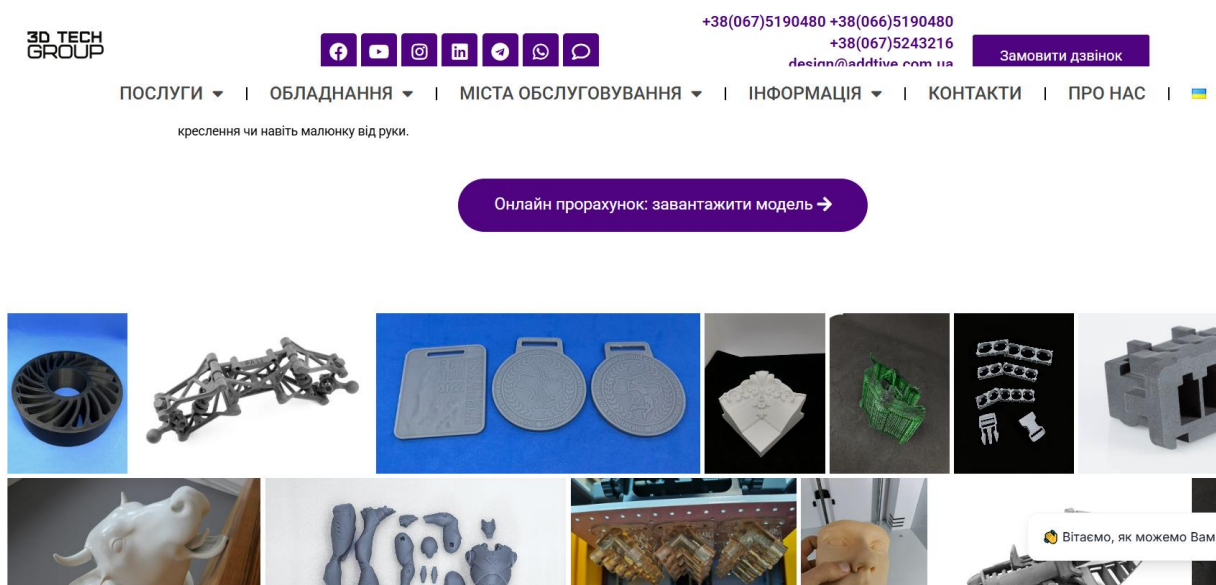


Рис.1.2. Інтерфейс вебсайту 3DTECHGROUP.

Третій популярний сервіс 3DForYou [5], займає нішу доступного 3D-друку для широкого кола споживачів. Вони спеціалізуються на сувенірній продукції та нескладних прототипах. Їхньою перевагою є активність у соціальних мережах та гнучка система знижок, що дуже приваблює студентів та молодих розробників. Але якщо розглядати технічну сторону сайту, то тут знову ж таки відсутні інструменти для автоматичної перевірки 3D-моделей перед оплатою. Клієнт може завантажити модель, яка технічно не може бути надрукована (наприклад, з розірваними поверхнями або занадто тонкою геометрією), і дізнається він про це лише після того, як майстер візьме замовлення в роботу. Це призводить до марної втрати часу на узгодження та повернення коштів. Окрім цього, на сайті відсутня будь-яка форма попереднього перегляду готового виробу в кольорі, що додатково ускладнює прийняття рішення клієнтом, який орієнтується лише на загальний опис товару в каталозі. Інтерфейс вебсайту зображено на рисунку 1.3.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

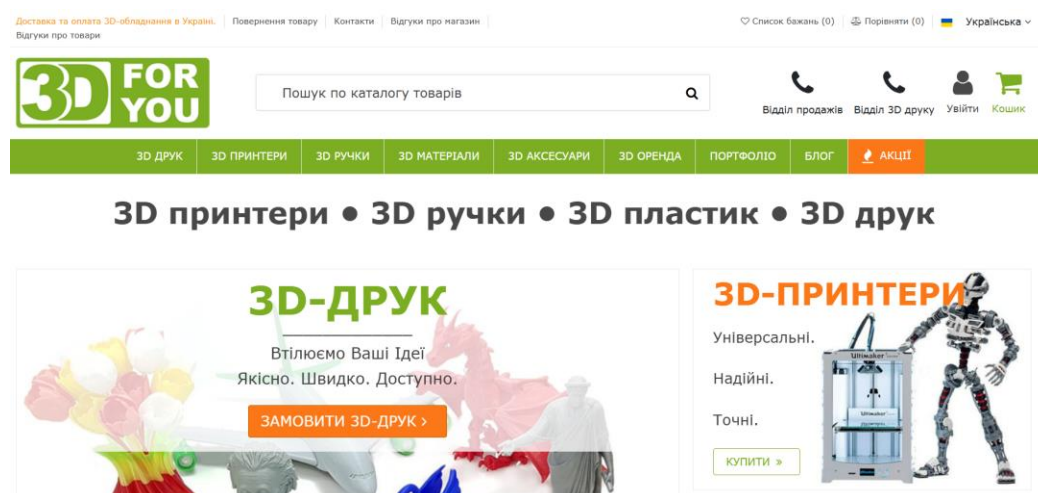


Рис.1.3. Інтерфейс вебсайту 3DForYou.

Аналізуючи ці аналоги, можна прийти до висновку, що загальною слабкою стороною українських сервісів є низький рівень інтерактивності. Жоден із розглянутих сайтів не дозволяє клієнту покрутити свою модель у 3D-вікні прямо в браузері перед замовленням, щоб переконатися, що файл завантажився коректно. Також ніхто не пропонує інтелектуальну систему підказок, яка б на основі штучного інтелекту допомагала обрати правильний матеріал під конкретну задачу користувача. Більшість сервісів працюють за схемою «запит – очікування відповіді – оплата за реквізитами», що значно сповільнює бізнес-процеси.

Платформа «FABBL Y» вирішує описані проблеми за рахунок двох ключових нововведень, відсутніх у проаналізованих аналогів. AI-асистент виконує технічний аудит моделі одразу після завантаження файлу, тож проблемні ділянки геометрії стають відомі ще до того, як замовлення побачить оператор, а не виявляються вже під час друку. Окремо, пряма інтеграція з обладнанням Bambu Lab дає клієнту можливість стежити за процесом друку в реальному часі – функцію, якої немає в жодному з розглянутих сервісів. Користувач в особистому кабінеті зможе бачити прогрес у відсотках і бути впевненим, що його замовлення вже на принтері.

Таблиця 1.1 демонструє порівняння функціональних можливостей. Найбільший розрив між «FABBL Y» та аналогами стосується саме автоматизації: розрахунок ціни в більшості сервісів виконується вручну або за приблизними

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савицький Р.С.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

таблицями, а онлайн-перегляд 3D-моделі перед замовленням на ринку практично не зустрічається, хоча саме ця функція напряму впливає на впевненість клієнта в правильності файлу. Відстеження статусу замовлення в особистому кабінеті теж радше виняток, через що клієнтам доводиться постійно уточнювати хід виконання у менеджера напряму.

Таблиця 1.1

Критерії порівняння	Easy3DPrint	3DTECHGROUP	3DForYou	Власна платформа «FABBL Y»
Автоматичний розрахунок вартості	Частково	Відсутній	Відсутній	Наявно
Онлайн-перегляд 3D-моделі	Відсутній	Відсутній	Відсутній	Наявно
Технічний AI-аудит (перевірка помилок)	Відсутній	Відсутній	Відсутній	Наявно
Особистий кабінет клієнта	Відсутній	Відсутній	Відсутній	Наявно
Моніторинг друку в реальному часі	Відсутній	Відсутній	Відсутній	Наявно
Автоматичний складський облік	Вручну	Частково	Вручну	Наявно
Зв'язок з обладнанням (Bambu Lab)	Відсутній	Відсутній	Відсутній	Наявно

Отже, розроблення власної платформи є не просто доцільним, а необхідним кроком для розвитку сучасного ринку адитивних послуг в Україні. Впровадження автоматизованої оцінки, AI-аналізу та прямого моніторингу обладнання дозволить «FABBL Y» стати унікальним продуктом, який закриває всі потреби користувачів.

1.3 Обґрунтування вибору архітектури онлайн-сервісу 3D-друку

Проектування архітектури інтелектуального сервісу 3D-друку «FABBLY» вимагає глибокого аналізу структурних патернів, оскільки система повинна поєднувати в собі роботу з клієнтськими інтерфейсами, важкі обчислення штучного інтелекту та постійний обмін даними з апаратним забезпеченням. Першим кроком у дослідженні став порівняльний опис архітектурних патернів, що дозволив визначити найбільш стійку модель для масштабування бізнесу.

Найбільш традиційним підходом є монолітна архітектура, де всі функціональні модулі, від обробки замовлень до логіки AI-аналізу, працюють як єдиний програмний блок. Такий підхід має суттєву перевагу в швидкості початкової розробки та простоті тестування взаємозв'язків між функціями. Однак у випадку сервісу 3D-друку моноліт стає занадто ризикованим рішенням. Будь-яка критична помилка в одному з модулів, наприклад, під час рендерингу важкої STL-моделі, може призвести до падіння всього сервера, що зупинить моніторинг принтерів та процес оплати. Крім того, монолітна структура ускладнює оновлення окремих частин системи, вимагаючи повного перезапуску застосунку при кожній зміні в коді.

Протилежним підходом є мікросервісна архітектура, де кожна функція системи виноситься в окремий незалежний сервіс. Це забезпечує максимальну відмовостійкість, оскільки вихід з ладу модуля оплат не вплине на роботу модуля моніторингу обладнання. Проте мікросервіси створюють величезне навантаження на мережеву інфраструктуру та потребують складних механізмів синхронізації бази даних.

Для платформи «FABBLY» було обрано модульну клієнт-серверну архітектуру на основі трирівневої моделі, яка включає рівень представлення, рівень бізнес-логіки та рівень збереження даних. Рівень представлення відповідає за взаємодію з користувачем через вебінтерфейс, де основне навантаження лягає на клієнтську частину для візуалізації 3D-об'єктів. Рівень бізнес-логіки виступає центральним шлюзом, який координує роботу всіх модулів: від перевірки прав доступу до взаємодії з нейромережами та

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

принтерами. Рівень даних забезпечує надійне збереження інформації, використовуючи принципи цілісності транзакцій. Така структура дозволяє ізолювати складні обчислення від користувацького інтерфейсу, забезпечуючи плавність роботи сайту навіть під час важких запитів до AI.

Архітектура «FABBLY» побудована таким чином, що окремі модулі системи (складський облік, фінансові операції, сповіщення клієнтів) логічно відокремлені одне від одного на рівні бази даних та контролерів, що створює основу для подальшого переходу до подійно-орієнтованої взаємодії. У перспективі це дозволить системі реагувати на завершення друку асинхронно: коли принтер передасть сигнал про завершення завдання, відповідні модулі (оновлення залишків пластику, закриття замовлення, сповіщення клієнта) можуть бути викликані автоматично через внутрішню систему подій, без додавання нової логіки до основного потоку обробки запитів. Така структура робить платформу гнучкою для додавання нових функцій, наприклад інтеграції з логістичними службами, без суттєвої переробки наявного коду.

Для управління станом клієнтської частини було обрано патерн централізованого сховища (Store Pattern), реалізований за допомогою Zustand [6]. Це дозволяє уникнути хаотичної передачі даних між компонентами та створює «єдине джерело істини». У контексті проєкту це особливо важливо для інтерактивного вікна налаштування параметрів друку. Коли користувач змінює відсоток заповнення або тип матеріалу в бічній панелі, стан оновлюється в центральному сховищі, і 3D-модель у центрі екрана миттєво реагує на ці зміни, наприклад, змінюючи колір або розрахункову вартість. Це забезпечує високу інтерактивність, якої не вистачає більшості існуючих аналогів.

Важливим архітектурним рішенням став вибір патерна проєктування REST [7] для побудови програмного інтерфейсу API. Використання REST дозволяє забезпечити «stateless» стан сервера, де кожен запит від клієнта містить у собі всю необхідну інформацію для його обробки. Для авторизації при цьому використовується патерн JWT [8]. Це підвищує безпеку системи, оскільки сервер не зберігає сесії користувачів у пам'яті, що захищає від атак типу сесійного

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

викрадення та дозволяє легко масштабувати систему на декілька серверних інстансів у майбутньому. Крім того, такий підхід полегшує інтеграцію з мобільними застосунками або сторонніми сервісами, оскільки API стає універсальним стандартом обміну даними.

Для роботи з апаратним рівнем у архітектурі передбачено окремий комунікаційний рівень, що базується на патерні Publish/Subscribe через протокол MQTT [9]. Це архітектурне рішення дозволяє системі не «опитувати» принтер кожну секунду, а отримувати дані лише тоді, коли вони реально змінилися. Це суттєво розвантажує мережу та процесорний час сервера. Сервер підписується на топіки телеметрії принтера Bambu Lab і автоматично оновлює дані в базі та на екрані користувача при надходженні нових значень температури або прогресу друку.

Таким чином, обрана комбінація архітектурних патернів створює надійну та швидку систему, яка здатна обробляти складні технічні дані без затримок. Розподіл на логічні шари, використання подій для синхронізації та сучасних методів авторизації роблять платформу «FABBLY» професійним інструментом автоматизації. Така архітектура повністю відповідає вимогам сучасної індустрії 3D-друку, забезпечуючи високу автономність процесів та комфорт для кінцевого користувача. Загальна структура обраної архітектури, що відображає розподіл рівнів та основні архітектурні патерни, застосовані на кожному з них, наведена на рисунку 1.4.

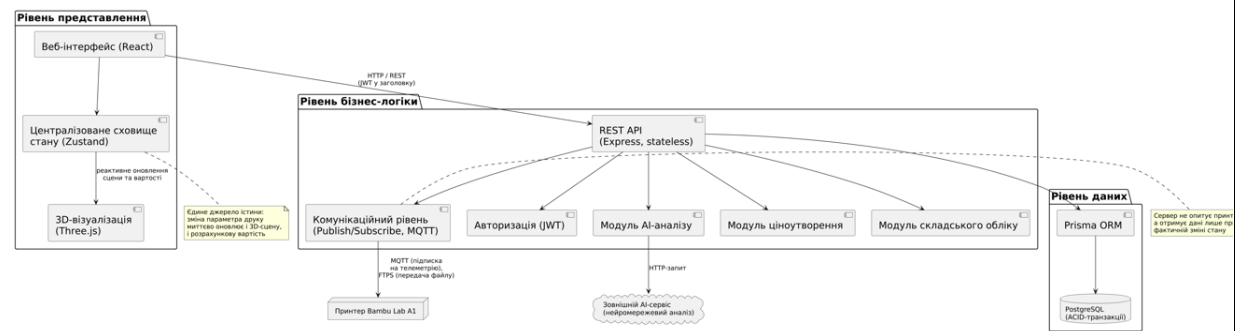


Рис.1.4. Загальна структура архітектури вебсервісу.

1.4 Обґрунтування вибору інструментальних засобів

Реалізація функціональних можливостей платформи «FABBLU» потребує використання сучасного та надійного технологічного стеку, який би забезпечував високу швидкість обробки даних та стабільність асинхронних процесів. Основним інструментом розробки серверної частини було обрано платформу Node.js [10]. Вибір даного середовища виконання обґрунтований його подійно-орієнтованою архітектурою, яка дозволяє обробляти велику кількість одночасних з'єднань без блокування основного потоку виконання. Це має вирішальне значення для сервісу, який повинен одночасно приймати важкі 3D-файли від користувачів, підтримувати зв'язок із нейронними мережами для аналізу геометрії та отримувати телеметричні дані від принтерів у реальному часі. Використання Node.js у поєднанні з фреймворком Express [11] дозволяє створити легковажний та продуктивний API, який легко масштабується при зростанні навантаження.

Для побудови клієнтської частини застосунку було обрано бібліотеку React [12], яка на сьогодні є стандартом у розробці динамічних вебінтерфейсів. Використання React дозволяє реалізувати складну логіку взаємодії користувача з AI-асистентом та модулем візуалізації через компонентний підхід. Це означає, що кожна частина інтерфейсу, наприклад, панель керування принтером або вікно перегляду моделі, є незалежним модулем, який легко оновлювати та підтримувати. Важливим доповненням до стеку розробки є мова програмування TypeScript [13], яка додає строгу типізацію до JavaScript. Це значно підвищує надійність коду, оскільки дозволяє виявляти потенційні помилки ще на етапі написання програми, що особливо важливо при роботі з фінансовими транзакціями через LiqPay та складними структурами даних замовлень. Взаємодія клієнтської частини з серверним API здійснюється засобами бібліотеки Axios [14].

Центральним елементом збереження даних обрано реляційну систему PostgreSQL [15], яка відзначається високою надійністю та підтримкою складних зв'язків між таблицями. На відміну від нереляційних баз, PostgreSQL гарантує

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				20
Змн.	Арк.	№ докум.	Підпис	Дата		

цілісність даних (ACID-відповідність), що є обов'язковою умовою для систем, які займаються обліком складських залишків та фінансовими операціями. Для полегшення взаємодії між кодом сервера та базою даних використовується Prisma ORM [16]. Цей інструмент дозволяє працювати з базою даних як з об'єктами мови програмування, автоматично генеруючи типи та забезпечуючи високу швидкість написання запитів. Prisma також спрощує процес міграції бази даних, що дозволяє гнучко змінювати структуру таблиць у процесі розвитку проєкту.

Ключовим апаратним компонентом системи є 3D-принтер Bambu Lab A1, обраний через наявність вбудованого Wi-Fi модуля та підтримку мережеских протоколів FTPS [17] і MQTT, що дозволяє реалізувати віддалене керування процесом друку без додаткових проміжних пристроїв. Взаємодія сервера з принтером побудована за двоетапною схемою: спочатку посланий файл моделі передається на SD-карту принтера через захищене FTPS-з'єднання, після чого серверна частина публікує команду запуску друку в відповідний MQTT-топік принтера, використовуючи протокол MQTTS. Така архітектура комунікаційного рівня дозволяє оператору запускати друк замовлень безпосередньо з адміністративної панелі платформи, без необхідності фізичного доступу до пристрою. У перспективі цей же канал зв'язку може бути розширений підпискою на топіки телеметрії принтера для отримання даних про температуру сопла, стола та прогрес виконання завдання в реальному часі.

Для реалізації AI-сервісу обрано мову програмування Python та фреймворк FastAPI [18], що забезпечує високу швидкість обробки асинхронних запитів та автоматичну валідацію вхідних даних завдяки вбудованій підтримці типізації. Для роботи з нейронною мережею, що класифікує матеріал та прогнозує параметри друку, використано бібліотеку PyTorch [19], яка надає гнучкий інтерфейс побудови та навчання моделей глибокого навчання й підтримує ефективне виконання обчислень на процесорі без потреби у спеціалізованому графічному прискорювачі, що знижує вимоги до апаратного забезпечення сервера. Для геометричного аналізу завантажених тривимірних моделей,

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

визначення об'єму, площі поверхні, товщини стінок та наявності нависань, застосовано бібліотеку trimesh [20], що надає зручний програмний інтерфейс для роботи з полігональними сітками у форматах STL та OBJ. Підготовка моделі до фізичного друку виконується засобами PrusaSlicer [21] у режимі командного рядка, що дозволяє інтегрувати потужний відкритий слайсер безпосередньо у серверний процес без необхідності розробки власного алгоритму генерації керівної програми для принтера.

Для інтерактивної тривимірної візуалізації моделей у браузері клієнтської частини обрано бібліотеку Three.js [22] у поєднанні з React-обгорткою React Three Fiber [23], що дозволяє відображати завантажені користувачем STL та OBJ файли безпосередньо на сторінці без потреби у встановленні додаткового програмного забезпечення для перегляду тривимірної геометрії. Таким чином це суттєво підвищує впевненість клієнта в коректності завантаженого файлу ще до моменту оформлення замовлення, що було визначено як суттєвий недолік розглянутих аналогів. Для зберігання файлів моделей, послансаних файлів друку та зображень каталогу товарів обрано хмарний сервіс Cloudinary [24], що звільняє серверну частину від необхідності самостійно керувати файловою системою та забезпечує доступ до збережених файлів через стабільні посилання незалежно від навантаження на основний сервер.

Загалом, вибраний комплекс інструментальних засобів та апаратного забезпечення створює збалансовану систему, де програмні технології максимально ефективно використовують можливості сучасного заліза. Використання Node.js, React та PostgreSQL забезпечує необхідну швидкість і надійність, а інтеграція з передовим обладнанням Bambu Lab дозволяє реалізувати повний цикл автоматизації онлайн-сервісу 3D-друку. Такий продуманий вибір технологій дозволяє уникнути вузьких місць в архітектурі, забезпечуючи високу відмовостійкість платформи та легкість масштабування при майбутньому збільшенні обсягів замовлень. Це створює надійний фундамент для подальшої реалізації об'єктної структури та логічних модулів системи, що будуть описані в наступних розділах.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				22
Змн.	Арк.	№ докум.	Підпис	Дата		

Висновки до першого розділу

У першому розділі проведено аналіз предметної області та обґрунтовано доцільність розробки платформи «FABBLY». Дослідження ринку онлайн-сервісів 3D-друку виявило відсутність на ринку комплексних рішень, які б поєднували автоматизовану перевірку моделей та пряме керування обладнанням.

Постановка задачі показала, що типовий робочий процес студій 3D-друку розпорошений між кількома незалежними каналами комунікації, а перевірка моделей, розрахунок вартості та облік матеріалів виконуються вручну, що призводить до затримок і помилок при зростанні кількості замовлень. Сформульовано перелік ключових напрямків автоматизації: технічний аудит моделі засобами AI-асистента, автоматичний розрахунок вартості з урахуванням складського обліку філаменту та пряме керування обладнанням без участі оператора на кожному етапі.

Аналіз аналогів програмного продукту показав, що жоден з них не пропонує онлайн-перегляду моделі перед замовленням, автоматизованої перевірки геометрії чи моніторингу друку в реальному часі, а робочий процес здебільшого зводиться до очікування ручної відповіді менеджера. Виявлені недоліки підтвердили доцільність розробки власної платформи та визначили перелік функцій, що мають стати її конкурентною перевагою.

Обґрунтування архітектури показало, що для системи, яка одночасно обробляє клієнтські запити, обчислення штучного інтелекту та взаємодію з обладнанням, найбільш придатною є трирівнева клієнт-серверна модель з логічним розмежуванням рівня представлення, бізнес-логіки та даних. Застосування патерну централізованого сховища на клієнті, REST API зі stateless-авторизацією на основі JWT та окремого комунікаційного рівня на основі MQTT для взаємодії з принтером забезпечує систему гнучкістю, необхідною для подальшого розширення функціоналу без суттєвої переробки наявного коду.

Вибір інструментальних засобів – Node.js та Express для серверної частини, React і TypeScript для клієнтської, PostgreSQL з Prisma ORM для зберігання

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

даних, а також обладнання Bambu Lab A1 з підтримкою протоколів FTPS і MQTT – забезпечив технологічну основу, що відповідає вимогам швидкодії, надійності та масштабованості, визначеним для проєктованої системи.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

РОЗДІЛ 2. ПРОЄКТУВАННЯ ОНЛАЙН-СЕРВІСУ 3D-ДРУКУ З ІНТЕГРАЦІЄЮ AI АСИСТЕНТА

2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи

Онлайн-сервіс передбачає рольову модель доступу, яка визначає набір функціональних можливостей, доступних кожній категорії користувачів. Розмежування прав здійснюється на рівні серверної частини за допомогою механізму авторизації на основі JSON Web Token, а на клієнтській частині – через динамічну побудову навігаційного меню та захищені маршрути, доступ до яких залежить від ролі поточного користувача.

Найнижчий рівень доступу має неавторизований відвідувач (Гість). Цей актор не має облікового запису в системі, проте може переглядати каталог готових виробів, ознайомлюватися з картками товарів, додавати їх до кошика та оформлювати замовлення без попередньої реєстрації, вказуючи контактні дані (електронну пошту та номер телефону) безпосередньо під час оформлення. Така можливість суттєво знижує бар'єр входу для нових клієнтів, оскільки дозволяє здійснити замовлення без витрат часу на створення облікового запису. У разі подальшої реєстрації або входу в систему з тією самою адресою електронної пошти раніше оформлені гостьові замовлення автоматично прив'язуються до облікового запису користувача.

Основним актором системи є клієнт (роль CLIENT) – зареєстрований користувач, який отримує розширені можливості порівняно з гостем: збереження та редагування профілю, перегляд історії власних замовлень із деталізацією статусів кожної позиції, а також доступ до розширеного функціоналу кастомних замовлень, що передбачає завантаження власних тривимірних моделей у форматах STL або OBJ, їхній перегляд у вбудованому тривимірному переглядачі та отримання автоматизованого технічного аудиту від AI-модуля.

Окрему категорію становить оператор (роль OPERATOR) – співробітник сервісу, відповідальний за безпосереднє виконання виробничих операцій. До його функціональних можливостей належить перегляд переліку призначених

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				25
Змн.	Арк.	№ докум.	Підпис	Дата		

йому завдань на друк, управління статусами окремих позицій замовлень у процесі виконання, контроль за станом обладнання (принтерів) та запуск друку безпосередньо з адміністративної панелі, а також ведення обліку складських залишків філаменту.

Роль менеджера (MANAGER) розширює можливості оператора функціями управління всім потоком замовлень незалежно від виконавця: перегляд та зміну статусів груп замовлень на рівні системи, призначення операторів на конкретні завдання, а також доступ до узагальненої аналітичної інформації щодо завантаженості виробництва.

Найвищий рівень доступу належить адміністратору (ADMIN), який окрім усіх перелічених можливостей відповідає за управління обліковими записами користувачів та їхніми ролями, ведення каталогу готових виробів (включно із завантаженням попередньо підготовлених файлів для слайсингу за кожним типом матеріалу), а також налаштування глобальних параметрів ціноутворення системи. Розподіл функціональних можливостей за ролями представлено у вигляді таблиці 2.1.

Таблиця 2.1

Розподіл функціональних можливостей за ролями користувачів

Функціональна можливість	Гість	Клієнт	Оператор	Менеджер	Адміністратор
Реєстрація та автентифікація	+	+	+	+	+
Перегляд каталогу готових виробів	+	+	+	+	+
Оформлення замовлення з каталогу	+	+	+	+	+
Завантаження моделі та отримання AI-аналізу	-	+	+	+	+
Налаштування параметрів друку та розрахунок вартості	-	+	+	+	+
Оформлення та оплата замовлення	+	+	+	+	+

продовж. табл. 2.1

Функціональна можливість	Гість	Клієнт	Оператор	Менеджер	Адміністратор
Перегляд статусу власних замовлень	-	+	+	+	+
Керування власним профілем	-	+	+	+	+
Управління завданнями на друк	-	-	+	+	+
Управління складом філаменту	-	-	+	+	+
Управління принтерами та запуск друку	-	-	+	+	+
Управління групами замовлень усіх клієнтів	-	-	-	+	+
Управління користувачами та ролями	-	-	-	-	+
Управління каталогом та параметрами ціноутворення	-	-	-	-	+

Модель варіантів використання системи побудована за стандартом UML 2.0 [25] у середовищі StarUML [26] з використанням текстового опису діаграм засобами PlantUML [27], що відповідає рекомендаціям щодо оформлення схем класів [28] та діаграм діяльності [29] у нотації UML, та відображає взаємодію визначених акторів із програмним комплексом через окремі прецеденти, кожен з яких відповідає закінченому сценарію, що приносить користувачу певний результат. Особливістю побудованої моделі є наявність ієрархії акторів: оператор успадковує всі варіанти використання гостя та клієнта в частині перегляду каталогу, менеджер успадковує функціональні можливості оператора, а адміністратор – можливості менеджера, що на діаграмі позначається відношенням узагальнення між акторами. Така ієрархія відповідає організаційній структурі сервісу 3D-друку, де співробітник вищого рівня має повний доступ до інструментів, призначених для співробітників нижчого рівня.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				27
Змн.	Арк.	№ докум.	Підпис	Дата		

Узагальнена діаграма варіантів використання (рис. 2.1) групує прецеденти за функціональними блоками: блок автентифікації та управління профілем, блок роботи з каталогом готових виробів, блок кастомних замовлень з інтегрованим AI-аналізом, блок оформлення та оплати замовлень, блок управління виробництвом (принтери, склад філаменту) та блок адміністрування системи.

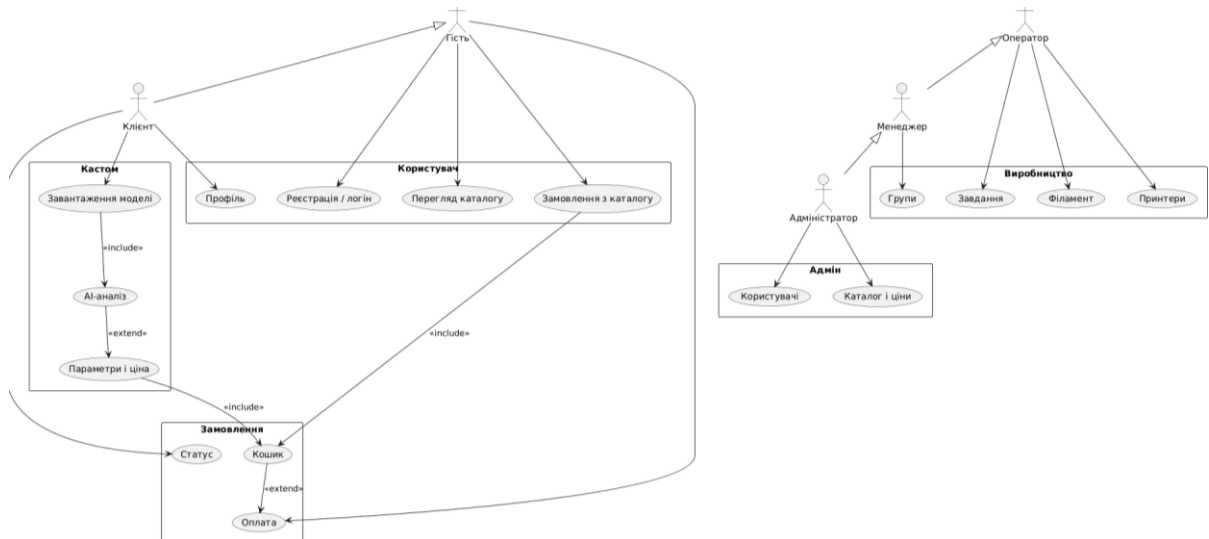


Рис.2.1. Діаграма варіантів використання.

Вимоги до системи поділено на дві категорії: функціональні (табл. 2.2), що описують конкретні дії, які система повинна виконувати у відповідь на запити користувачів, та нефункціональні (табл. 2.3), що визначають якісні характеристики роботи системи – швидкодію, безпеку, надійність та зручність використання.

Таблиця 2.2

Функціональні вимоги

Ідентифікатор	Опис вимоги
ФВ1	Реєстрація нового користувача із зазначенням імені, електронної пошти та пароля, а також автентифікація зареєстрованого користувача з видачею токена доступу та токена оновлення
ФВ2	Перегляд каталогу готових виробів та оформлення замовлення без реєстрації, із зазначенням контактної електронної пошти та номера телефону

Ідентифікатор	Опис вимоги
ФВ3	Автоматична прив'язка раніше оформлених гостьових замовлень до облікового запису користувача у разі реєстрації або входу з тією самою електронною поштою
ФВ4	Завантаження файлу тривимірної моделі у форматі STL або OBJ та його відображення у вбудованому тривимірному переглядачі
ФВ5	Передача завантаженої моделі до AI-сервісу для аналізу геометричних характеристик та формування переліку потенційних проблем друку
ФВ6	Формування на основі результатів AI-аналізу рекомендацій щодо матеріалу друку, відсотка заповнення та необхідності використання підтримок
ФВ7	Підготовка моделі до друку із заданими параметрами та автоматичний розрахунок ваги виробу, часу друку і вартості замовлення
ФВ8	Формування кошика користувача, що може містити позиції з каталогу готових виробів та кастомні позиції з результатами слайсингу
ФВ9	Створення групи замовлення з обраним способом доставки та оплати під час оформлення замовлення, з формуванням окремих записів для кожної позиції кошика
ФВ10	Відображення історії замовлень користувача з деталізацією статусу кожної групи замовлення та кожної окремої позиції в межах групи
ФВ11	Перегляд оператором переліку позицій замовлень, призначених до виконання, та зміна статусу виконання окремої позиції
ФВ12	Облік залишків філаменту за матеріалом і кольором з автоматичним зменшенням залишку на величину фактично витраченої ваги після завершення друку
ФВ13	Сповіщення персоналу про критично низький залишок котушки філаменту певного матеріалу та кольору
ФВ14	Передача підготовленого файлу друку на принтер та запуск друку безпосередньо з адміністративної панелі
ФВ15	Перегляд та зміна статусу будь-якої групи замовлень менеджером незалежно від клієнта-замовника
ФВ16	Зміна ролі будь-якого зареєстрованого користувача адміністратором

Ідентифікатор	Опис вимоги
ФВ17	Створення, редагування та видалення позицій каталогу готових виробів, а також завантаження підготовлених файлів друку за кожним типом матеріалу
ФВ18	Зміна глобальних параметрів формули розрахунку вартості (вартість години друку, відсоток націнки, надбавка за складність)

Нефункціональні вимоги до системи наведено в таблиці 2.3.

Таблиця 2.3

Нефункціональні вимоги до системи

Ідентифікатор	Категорія	Опис вимоги
НВ1	Швидкодія	Типові запити до серверної частини (перегляд каталогу, історії замовлень, профілю) опрацьовуються без помітної для користувача затримки, процес AI-аналізу моделі супроводжується індикатором виконання, оскільки триває довше за звичайний запит
НВ2	Безпека	Доступ до захищених маршрутів API можливий лише за наявності дійсного JWT-токена; паролі користувачів зберігаються у вигляді хешу, файли тривимірних моделей доступні лише авторизованому персоналу відповідного рівня
НВ3	Надійність	Цілісність даних про замовлення, оплати та складські залишки забезпечується транзакційним механізмом PostgreSQL, тимчасова недоступність AI-сервісу не призводить до відмови інших функціональних блоків системи
НВ4	Масштабованість	Відсутність серверного стану сесій (stateless REST API на основі JWT) дозволяє горизонтально масштабувати серверну частину при зростанні навантаження
НВ5	Сумісність	Клієнтська частина коректно відображається в актуальних версіях основних веброзглядачів та адаптується під різні розміри екрана

Ідентифікатор	Категорія	Опис вимоги
НВ6	Зручність використання	Інтерфейс забезпечує інтерактивний перегляд тривимірної моделі (обертання, масштабування, переміщення) та миттєву реакцію пов'язаних елементів (вартість, вигляд моделі) на зміну параметрів друку
НВ7	Розширюваність	Логічне розмежування модулів складського обліку, фінансових операцій та сповіщень на рівні бази даних і контролерів дозволяє в подальшому перейти до подійно-орієнтованої взаємодії без суттєвої переробки коду

Об'єктна структура системи відображає основні сутності предметної області онлайн-сервісу 3D-друку, їхні атрибути та зв'язки, на основі яких надалі будується фізична схема бази даних.

Центральним класом структури є User, що представляє користувача системи незалежно від його ролі. Клас містить атрибути ідентифікатора, імені, електронної пошти, хешу пароля та ролі (CLIENT, OPERATOR, MANAGER або ADMIN). Саме поле ролі визначає, які операції доступні конкретному екземпляру класу, що відповідає рольовій моделі доступу, описаній у попередньому пункті. Від класу User відходять зв'язки до груп замовлень, які він оформив, а для ролей OPERATOR і вище – додатковий зв'язок до позицій замовлень, призначених йому на виконання.

Окремої уваги потребує структура замовлення, яка є найбільш суттєвим архітектурним рішенням у проєкті. Початковий варіант передбачав єдиний клас Order, що одночасно відповідав би і за загальні параметри замовлення (доставка, оплата, контактні дані), і за окрему позицію в кошику. Проте така структура унеможлиблювала коректне оформлення замовлення, що складається з кількох різнорідних позицій – наприклад, одного виробу з каталогу та однієї кастомної моделі в межах однієї покупки, з єдиною оплатою та єдиним способом доставки. Тому структуру було розділено на два пов'язані класи: OrderGroup та Order.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		31

Клас OrderGroup відповідає за замовлення на рівні покупки в цілому, він містить загальний статус групи, спосіб доставки, дані для гостьового оформлення, наприклад, електронну пошту та номер телефону, якщо замовлення оформлене без реєстрації, та зв'язок із класом Payment, що відображає оплату або депозит. Клас Order представляє окрему позицію в межах групи замовлення, тобто конкретний товар із каталогу або конкретну кастому модель, і має власний статус, що дозволяє відстежувати виконання кожної позиції незалежно від інших у межах однієї групи. Кожен екземпляр Order пов'язаний з класом User через поле призначеного оператора, а також з класом Printer, на якому виконується або виконувався друк цієї позиції.

Тип позиції замовлення визначається атрибутом, що приймає значення CATALOG або CUSTOM, і відповідно до цього значення клас Order пов'язується з одним із двох класів деталізації. Для каталожних позицій таким класом є CatalogItem – товар із каталогу готових виробів, що містить назву, базову ціну, перелік доступних матеріалів та ваговий параметр, який використовується при розрахунку списання філаменту зі складу. Для кастомних позицій деталізація здійснюється через клас CustomOrderDetails, що зберігає посилання на завантажений файл тривимірної моделі, результати AI-аналізу (геометричні характеристики, виявлені проблеми, рекомендації), обрані користувачем параметри друку та результат слайсингу – розрахункову вагу, час друку і остаточну вартість. Саме клас CustomOrderDetails є точкою інтеграції основної серверної частини з окремим AI-сервісом, описаним у першому розділі: результат його роботи зберігається у вигляді структурованих полів цього класу та надалі використовується для формування ціни.

Складський облік матеріалів представлений парою класів FilamentType та Filament. Клас FilamentType описує тип пластику (наприклад, PLA, PETG, ABS або TPU) разом із базовою вартістю за грам, що використовується у формулі розрахунку ціни. Клас Filament представляє конкретну фізичну котушку певного типу, кольору та бренду, що перебуває на складі, із зазначенням поточного залишку ваги. Кожна позиція замовлення (Order), що передбачає використання

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

філаменту, пов'язується з конкретною катушкою через окремий клас прив'язки, що дозволяє після завершення друку коректно зменшити залишок саме тієї катушки, з якої здійснювався друк, а не загальний залишок матеріалу певного типу.

Апаратний рівень системи представлений класами Printer та PrintJob. Клас Printer описує фізичний 3D-принтер – його мережеву адресу, код доступу та серійний номер, необхідні для встановлення з'єднання за протоколами FTPS та MQTT, а також поточний стан пристрою. Клас PrintJob відображає конкретне завдання друку, що виконується на принтері, та його прогрес у відсотках, оновлюваний у реальному часі засобами фонових сервісів моніторингу. Зв'язок між Order та Printer через PrintJob забезпечує простежуваність шляху позиції замовлення від моменту оформлення до фізичного виконання на конкретному обладнанні – без такого зв'язку оператор не мав би можливості визначити, яке замовлення наразі друкується на якому принтері.

Наведена структура класів демонструє, що ключовою особливістю об'єктної моделі системи є розмежування рівня загального замовлення (OrderGroup) та рівня окремої позиції (Order), що уможливорює коректну роботу з різнорідним кошиком, частковою передоплатою для складних кастомних замовлень та незалежним відстеженням статусу кожного виробу в межах однієї покупки. Решта структурних елементів такі як, таблиці прив'язки філаменту до позицій, файли попередньо підготовлених моделей каталогу, відгуки користувачів, повідомлення підтримки та глобальні налаштування ціноутворення, деталізують зв'язки наведених класів. Жоден із перелічених класів не містить полів, які безпосередньо відповідають за бізнес-логіку розрахунків чи мережевої взаємодії – це винесено на рівень сервісів серверної частини, що дозволяє змінювати алгоритми обробки даних без втручання у структуру самих сутностей. Такий поділ за принципом єдиної відповідальності, спрощує подальше тестування окремих компонентів системи незалежно один від одного. А також така структура дає змогу в подальшому легко розширювати функціонал, не

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

вимагаючи зміни в наявних класах. Діаграма класів верхнього рівня, що відображає описану структуру, наведена на рисунку 2.2.

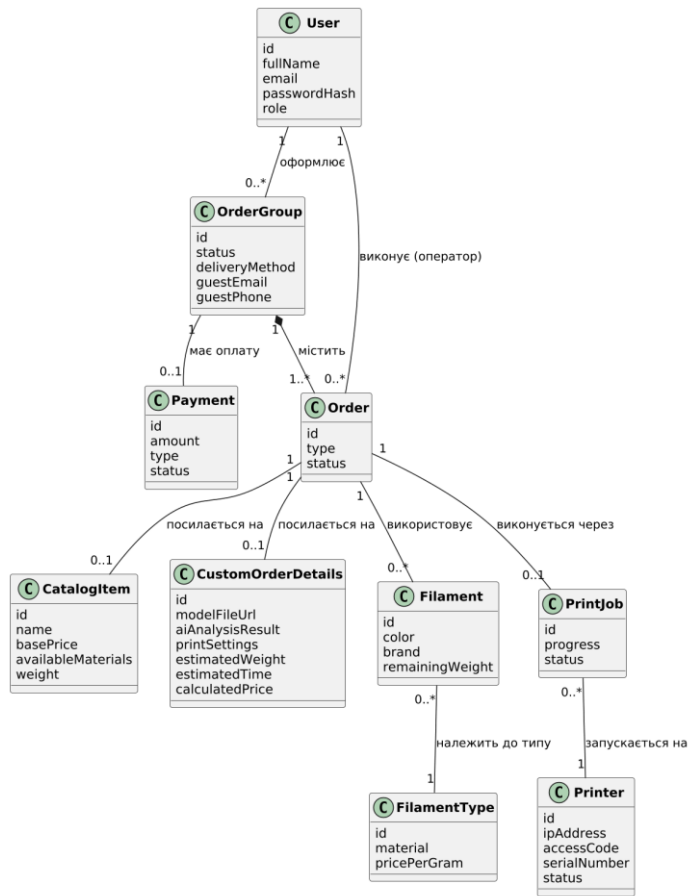


Рис.2.2. Діаграма класів верхнього рівня.

Зв'язок типу композиції між OrderGroup та Order відображає те, що позиція замовлення не може існувати поза групою – її життєвий цикл повністю залежить від групи, до якої вона належить. Натомість зв'язки Order з CatalogItem та CustomOrderDetails показані як звичайні асоціації з кратністю "0..1", оскільки кожна позиція посилається лише на один із цих класів залежно від свого типу.

2.2 Розробка бази даних системи

Фізична схема бази даних онлайн-сервісу 3D-друку реалізована засобами реляційної системи керування базами даних PostgreSQL з використанням Prisma ORM, вибір яких обґрунтовано в першому розділі. Схема безпосередньо відповідає об'єктній структурі, проте містить додаткову деталізацію – конкретні типи полів, обмеження цілісності, перерахування (enum) допустимих значень та таблиці прив'язки, що забезпечують зв'язки між основними сутностями. Загалом

схема складається із сімнадцяти таблиць, об'єднаних у шість змістових блоків: користувачі та авторизація, замовлення та оплати, каталог готових виробів, кастомні замовлення з AI-аналізом, складський облік філаменту та апаратний рівень, а також допоміжні таблиці – відгуки, чат підтримки та налаштування ціноутворення.

Центральною таблицею бази даних є User, що зберігає всіх користувачів системи незалежно від ролі. Кожен запис ідентифікується унікальним рядковим ключем у форматі UUID, а поле електронної пошти має обмеження унікальності, що унеможливорює повторну реєстрацію з однією й тією ж адресою. Пароль зберігається у вигляді хешу в полі типу String, проте це поле допускає значення null – такий підхід застосовується для облікових записів, створених через Google OAuth, для яких передбачено окреме унікальне поле googleId. Роль користувача зберігається у полі типу Role (перерахування зі значеннями CLIENT, OPERATOR, MANAGER, ADMIN) зі значенням за замовчуванням CLIENT, що відповідає принципу "найменших привілеїв" – новий користувач отримує мінімальний набір прав, а підвищення ролі виконується адміністратором. Від таблиці User відходить низка зв'язків один-до-багатьох: до груп замовлень, які оформив користувач orderGroups, до позицій замовлень, призначених йому як оператору operatorOrders, до відгуків, діалогів та повідомлень підтримки, до принтерів, за якими закріплений оператор, а також до записів історії змін налаштувань ціноутворення.

Блок замовлень та оплат складається з трьох пов'язаних таблиць – OrderGroup, Order та Payment, – структура яких реалізує архітектурне рішення, щодо розділення рівня покупки в цілому та рівня окремої позиції. Таблиця OrderGroup містить необов'язкове поле userId (тип String?, із зовнішнім ключем на User), що допускає значення null для гостьових замовлень, у такому випадку контактні дані зберігаються в полях guestEmail та guestPhone. Загальний статус групи зберігається у полі типу OrderGroupStatus, перерахування якого (PENDING, CONFIRMED, IN_PROGRESS, READY, SHIPPED, DELIVERED, CANCELLED) відображає послідовність етапів виконання замовлення від

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

моменту оформлення до доставки клієнту. Таблиця також містить поле `totalPrice` типу `Float` для загальної суми замовлення, поля `deliveryAddress` та `deliveryService` для зберігання адреси та обраної служби доставки (Нова Пошта або Укрпошта), поле `trackingNumber` для номера відправлення та поле `paymentMethod` типу `PaymentMethod`, що приймає значення `ONLINE` (оплата через `LiqPay`) або `ON_DELIVERY` (оплата при отриманні, доступна лише для замовлень із каталогу).

Таблиця `Order` представляє окрему позицію в межах групи замовлення і пов'язана з `OrderGroup` зовнішнім ключем `orderGroupId` за відношенням "багато-до-одного", що на рівні предметної області відповідає композиції, описаній у попередньому підрозділі. Поле `orderType` типу `OrderType` визначає, до якого з двох класів деталізації – `CatalogOrderDetails` чи `CustomOrderDetails` – посилається конкретний запис, причому обидва зв'язки оголошені як необов'язкові (тип "0..1") з протилежного боку, оскільки заповнюється лише один із них залежно від значення `orderType`. Статус окремої позиції зберігається у полі `status` типу `OrderStatus`, перерахування якого деталізоване порівняно зі статусом групи й охоплює стадії `PENDING`, `CONFIRMED`, `QUEUED`, `PRINTING`, `READY`, `SHIPPED`, `DELIVERED` та `CANCELLED` – саме на цьому рівні відстежується фактичний прогрес виконання конкретного виробу. Таблиця `Order` також містить необов'язкові зовнішні ключі `printerId` та `operatorId`, що зв'язують позицію замовлення з конкретним принтером та співробітником, відповідальним за її виконання, а поле `adminPriceAdjust` типу `Float`? дозволяє адміністратору вручну скоригувати розраховану вартість окремої позиції у виняткових випадках.

Таблиця `Payment` пов'язана з `OrderGroup` за відношенням "багато-до-одного" (одна група замовлення може мати кілька записів про оплату – наприклад, окремо депозит і доплату для кастомних замовлень). Поле `liqpayOrderId` типу `String` з обмеженням унікальності зберігає ідентифікатор транзакції в платіжній системі `LiqPay`, що використовується для зіставлення вебхуків від платіжного шлюзу з конкретним записом у базі даних. Поле `type` типу `PaymentType` розрізняє депозит (`DEPOSIT`) та повну оплату (`FULL`), а поле `status` типу `PaymentStatus`

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

відображає поточний стан транзакції (PENDING, SUCCESS, FAILED, REFUNDED). Поле paidAt типу DateTime? заповнюється лише після підтвердження оплати та використовується для аналітики й формування звітності щодо надходжень.

Блок каталогу готових виробів охоплює чотири таблиці. Таблиця CatalogItem зберігає основну інформацію про товар: назву, опис, масив посилань на зображення imageUrls типу String[], базову ціну адміністратора price, перелік підтримуваних матеріалів availableMaterials типу FilamentMaterial[] (масив значень перерахування PLA, PETG, ABS, TPU), а також технічні характеристики – відсоток заповнення infill, вагу weight та орієнтовний час друку printTime, що використовуються модулем розрахунку вартості. Поле isActive типу Boolean дозволяє приховувати позицію з публічного каталогу без її фізичного видалення з бази даних. Пов'язана таблиця CatalogSlicedFile зберігає попередньо підготовлені (послайсені) файли друку для кожного товару окремо за кожним матеріалом: поле material типу String визначає матеріал, а fileUrl – посилання на файл у хмарному сховищі. Складене унікальне обмеження @@unique([catalogItemId, material]) гарантує, що для одного товару не може існувати два файли друку для одного й того самого матеріалу, що відповідає логіці адміністративного завантаження файлів каталогу, описаній у першому розділі.

Деталі замовлення з каталогу зберігаються в парі таблиць CatalogOrderDetails та CatalogOrderItem, пов'язаних за відношенням "один-до-багатьох". Таблиця CatalogOrderDetails має зовнішній ключ orderId з обмеженням унікальності, що реалізує зв'язок "один-до-одного" з таблицею Order – кожна позиція замовлення типу CATALOG має рівно один такий запис деталізації, який, у свою чергу, може охоплювати кілька конкретних товарів. Кожен запис CatalogOrderItem містить кількість quantity, ціну на момент замовлення itemPrice (що фіксується окремо від поточної ціни товару в каталозі для коректності історичних даних), поле materialWeightUsed для фактично витраченої ваги матеріалу та поле material типу FilamentMaterial, яке визначає,

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

який саме матеріал клієнт обрав для цієї конкретної позиції з-поміж доступних для товару варіантів.

Найбільш насиченим за кількістю полів є блок кастомних замовлень, представлений таблицею CustomOrderDetails, що так само пов'язана з Order відношенням "один-до-одного" через унікальний зовнішній ключ orderId. Таблиця зберігає інформацію про завантажений файл моделі (fileUrl, fileName та fileType типу ModelFormat зі значеннями STL або OBJ), а також повний результат роботи AI-сервісу: поле analysisStatus типу AnalysisStatus відображає стадію обробки (PENDING, PROCESSING, DONE, FAILED), поле hasIssues типу Boolean? фіксує наявність технічних проблем, виявлених під час аналізу геометрії, а поле issuesDescription містить їхній текстовий опис. Геометричні характеристики моделі зберігаються в полях sizeX, sizeY, sizeZ типу Float? (розміри в міліметрах), а результати слайсингу – у полях estimatedWeight та estimatedTime. Обрані клієнтом параметри друку (material типу FilamentMaterial та infill типу Float) зберігаються окремо від рекомендацій AI (aiSuggestedMaterial, aiSuggestedInfill), що дозволяє в подальшому аналізувати, наскільки часто клієнти погоджуються з пропозиціями інтелектуального модуля. Окрему групу становлять поля, призначені для деталізації розрахунку вартості: materialCost, timeCost, complexityFee та markupAmount зберігають складові підсумкової ціни, а поле priceBreakdown типу Json дозволяє зберегти повну структуровану розшифровку формули розрахунку для конкретного замовлення без необхідності додавання нових стовпців у разі зміни логіки ціноутворення в майбутньому.

Зв'язок між позиціями замовлення, як каталожними, так і кастомними, та конкретними катушками філаменту на складі реалізується через таблицю прив'язки OrderFilament. Ця таблиця містить два альтернативні необов'язкові зовнішні ключі – catalogOrderItemId та customOrderDetailsId, – лише один з яких заповнюється для конкретного запису залежно від типу замовлення, та поле role типу FilamentRole (MAIN або SECONDARY), яке передбачає можливість використання в одному виробі основного та допоміжного кольору філаменту.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

Така структура дозволяє єдиним механізмом обслуговувати списання матеріалу як для товарів із каталогу, так і для кастомних моделей, без дублювання логіки складського обліку.

Складський облік матеріалів реалізований парою таблиць FilamentType та Filament, що відповідають однойменним класам. Таблиця FilamentType містить унікальне поле material типу FilamentMaterial та поле basePricePerGram – базову вартість грама матеріалу певного типу, що застосовується у формулі розрахунку ціни за умовчанням. Таблиця Filament зберігає конкретні фізичні котушки: назву, необов'язкове поле brand для виробника, колір, початкову вагу weightTotal та поточний залишок weightLeft, на основі різниці яких можна визначити сумарну витрату конкретної котушки за весь час її використання. Поле minWeightAlert зі значенням за умовчанням 50 визначає порогове значення залишку, після перетину якого система формує сповіщення для персоналу про необхідність поповнення складу, а поле pricePerGram типу Float? дозволяє вказати індивідуальну ціну для конкретної котушки, якщо вона відрізняється від базової ціни типу матеріалу.

Апаратний рівень системи представлений таблицями Printer та PrintJob. Таблиця Printer зберігає мережеву адресу принтера ipAddress, необов'язкові поля accessCode та serialNumber, необхідні для з'єднання за протоколами FTPS та MQTT, описаними в першому розділі, а також поточний статус типу PrinterStatus (IDLE, PRINTING, ERROR, OFFLINE) і телеметричні дані nozzleTemp та bedTemp, що оновлюються фоновим сервісом моніторингу в реальному часі. Таблиця PrintJob фіксує конкретне завдання друку, пов'язане одночасно з позицією замовлення (orderId) та принтером (printerId), і містить поле progress типу Float зі значенням за умовчанням 0, а також поля startedAt і finishedAt типу DateTime?, що дозволяють розрахувати фактичну тривалість друку для подальшого порівняння з прогнозованим часом, отриманим від AI-сервісу під час слайсингу.

Завершують схему три допоміжні таблиці. Таблиця Review зберігає відгуки користувачів і має два альтернативні необов'язкові зовнішні ключі – на товар

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		39

каталогу (catalogItemId) та на конкретне замовлення (orderId), – що дозволяє залишати відгук як на сам виріб, так і на якість виконання індивідуального замовлення, з полями rating типу Int та необов'язковим текстовим коментарем. Таблиці Conversation та Message реалізують структуру чату підтримки: діалог може бути прив'язаний як до зареєстрованого користувача через userId, так і до анонімного відвідувача через поле guestSessionId – тимчасовий ідентифікатор сесії браузера, а кожне повідомлення в межах діалогу має позначку isFromGuest та статус прочитання isRead. Останньою є таблиця PriceSettings, що зберігає глобальні параметри формули розрахунку вартості – ціну години роботи принтера pricePerHour, відсоток загальної націнки markupPercent, відсоток передоплати для кастомних замовлень depositPercent, максимальну надбавку за складність complexityMarkup, відсотковий буфер часу timeMarginPercent та вартість електроенергії electricityPerHour. Кожна зміна цих параметрів фіксується через зв'язок з таблицею User (поле updatedById), що забезпечує простежуваність історії редагування налаштувань ціноутворення адміністраторами системи.

Виокремлення цих параметрів у самостійну таблицю, а не зберігання їх безпосередньо в коді серверної частини, дозволяє адміністратору змінювати економічну модель сервісу, переглядати націнку у відповідь на зміну вартості електроенергії чи курсу закупівлі матеріалів – без необхідності повторного розгортання застосунку. Такий підхід узгоджується із загальним принципом розмежування бізнес-логіки та конфігураційних даних, застосованим і до інших обчислювальних параметрів системи, описаних у попередніх таблицях цього підрозділу. Завдяки чому, зміна бізнес-правил ціноутворення не потребує втручання в код контролерів чи перерахунку формул на стороні клієнтської частини, достатньо оновити відповідний запис у таблиці, і нове значення параметра автоматично застосовується до всіх подальших розрахунків вартості замовлень. Повна фізична схема бази даних із переліченими таблицями, типами полів та зв'язками між ними наведена на рисунку 2.3

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				40
Змн.	Арк.	№ докум.	Підпис	Дата		

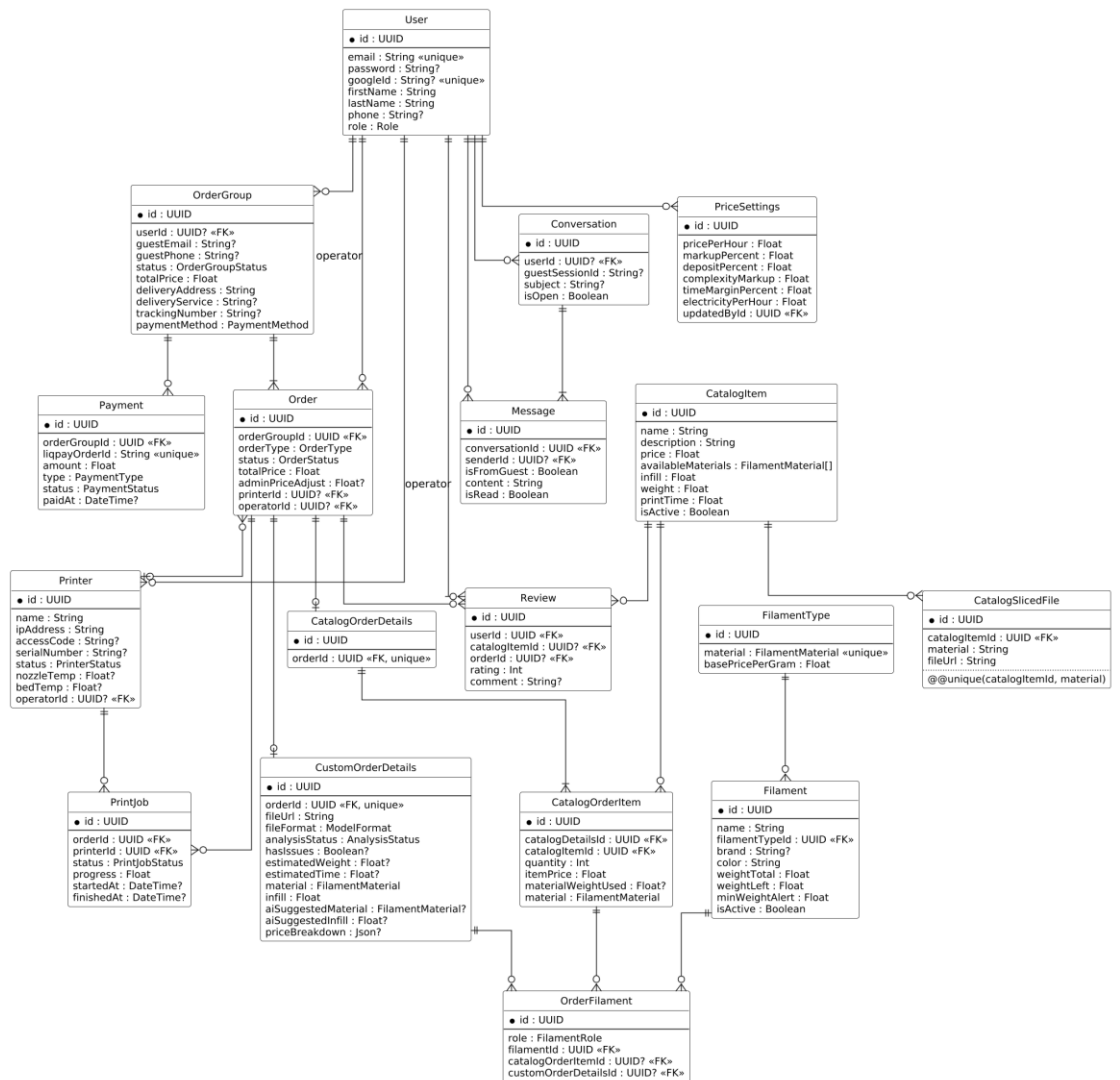


Рис.2.3. Діаграма класів.

Наведена схема демонструє, що складність структури бази даних зосереджена навколо двох вузлів – таблиці Order, яка є точкою з'єднання замовлення з каталогом, кастомними деталями, обладнанням та відгуками, та таблиці OrderFilament, яка єдиним механізмом обслуговує списання філаменту незалежно від типу позиції. Така централізація дозволяє уникнути дублювання логіки складського обліку та зберегти узгодженість даних при подальшому розширенні функціоналу системи, наприклад додаванні нових типів матеріалів чи нових способів оплати, без зміни структури вже наявних таблиць.

2.3 Проектування та реалізація алгоритмів роботи системи

Найбільш складним за кількістю задіяних компонентів є алгоритм опрацювання користувацького замовлення, оскільки саме він поєднує клієнтську частину, серверну логіку та окремий AI-сервіс, реалізуючи технічний аудит

		Шкільна А.Л.								Арк.
		Савіцький Р.С.								
Змн.	Арк.	№ докум.	Підпис	Дата						41

моделі та формування остаточної ціни. Алгоритм охоплює таблицю CustomOrderDetails, послідовно заповнюючи її поля на кожному етапі обробки.

Алгоритм розпочинається з дії користувача – завантаження файлу тривимірної моделі у форматі STL або OBJ через інтерфейс кастомного замовлення. Файл передається на сервер, де контролер моделей зберігає оригінальний файл у хмарному сховищі та створює запис CustomOrderDetails із заповненими полями fileUrl, fileName, fileFormat і початковим значенням поля analysisStatus, що дорівнює PENDING. Одразу після збереження файлу сервер формує запит до окремого AI-сервісу, переводячи статус аналізу у значення PROCESSING.

На стороні AI-сервісу відбувається геометричний аналіз моделі засобами бібліотеки trimesh: обчислюються габаритні розміри моделі (sizeX, sizeY, sizeZ), об'єм, площа поверхні, товщина стінок, частка нависань поверхні відносно горизонталі та наявність ізольованих нависань, що потребують додаткових підтримок. На основі цих геометричних ознак нейронна мережа, навчена на наборі даних Thing10K [30], формує чотири результати: ймовірнісну оцінку придатності моделі до друку, рекомендований відсоток заповнення, рекомендацію щодо матеріалу та рекомендацію щодо необхідності підтримок. Паралельно з роботою нейронної мережі евристичний модуль аналізу формує текстовий перелік виявлених проблем – наприклад, незамкнені поверхні моделі (відсутність водонепроникності) або надмірно тонкі стінки, що можуть призвести до браку під час друку.

Результати аналізу повертаються на основний сервер, де записуються у відповідні поля CustomOrderDetails: hasIssues приймає значення true, якщо евристичний модуль виявив хоча б одну проблему, issuesDescription містить її текстовий опис, а поля aiSuggestedMaterial та aiSuggestedInfill зберігають рекомендації нейронної мережі. Статус аналізу переводиться у значення DONE, після чого результати стають доступними на клієнтській частині – інтерфейс відображає рекомендації поряд із тривимірною моделлю у переглядачі на основі Three.js, дозволяючи користувачу візуально оцінити проблемні зони моделі.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				42
Змн.	Арк.	№ докум.	Підпис	Дата		

Подальший хід алгоритму розгалужується залежно від рішення користувача. Користувач може погодитися з рекомендаціями AI-модуля без змін, або самостійно скоригувати параметри друку – обрати інший матеріал із поля material або змінити значення infill, – якщо, наприклад, для конкретної деталі важливіша економія матеріалу, ніж міцність. Незалежно від обраного шляху, після підтвердження параметрів сервер ініціює другий етап обробки – підготовку файлу до друку (слайсинг).

Слайсинг виконується засобами PrusaSlicer CLI з використанням профілю принтера Bambu Lab A1, описаного в першому розділі. AI-сервіс формує команду для слайсера з урахуванням обраних параметрів матеріалу та заповнення, після чого отримує згенерований G-code. З файлу G-code здійснюється парсинг метаданих – фактичної ваги витраченого матеріалу та орієнтовного часу друку, до якого застосовується коефіцієнт корекції 0,56 для апроксимації значень, отримуваних у Bambu Studio. У випадку недоступності слайсера передбачено резервний механізм оцінки ваги та часу безпосередньо на основі об'єму геометричної моделі, що забезпечує відмовостійкість алгоритму відповідно до нефункціональної вимоги НВЗ. Отримані значення записуються у поля estimatedWeight та estimatedTime таблиці CustomOrderDetails, а посилання на згенерований файл – у поле slicedFileUrl.

Завершальним етапом алгоритму є розрахунок вартості замовлення, що виконується серверним сервісом ціноутворення на основі параметрів таблиці PriceSettings. Вартість матеріалу обчислюється як добуток estimatedWeight на ціну за грам обраного типу філаменту з таблиці FilamentType (або індивідуальну ціну конкретної котушки з таблиці Filament, якщо така задана). Вартість часу друку обчислюється на основі estimatedTime, ціни години роботи принтера pricePerHour та відсоткового буфера часу timeMarginPercent, що компенсує можливі похибки оцінки слайсера. У разі, якщо AI-модуль під час аналізу виявив проблеми моделі (поле hasIssues дорівнює true), до розрахунку додається надбавка за складність, обмежена параметром complexityMarkup. Сума складових множиться на коефіцієнт загальної націнки markupPercent, формуючи

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

підсумкову вартість замовлення, а повна структурована розшифровка кожної складової зберігається у полі priceBreakdown типу Json для подальшого використання адміністратором.

Увесь описаний алгоритм детально зображено на рисунку 2.4.

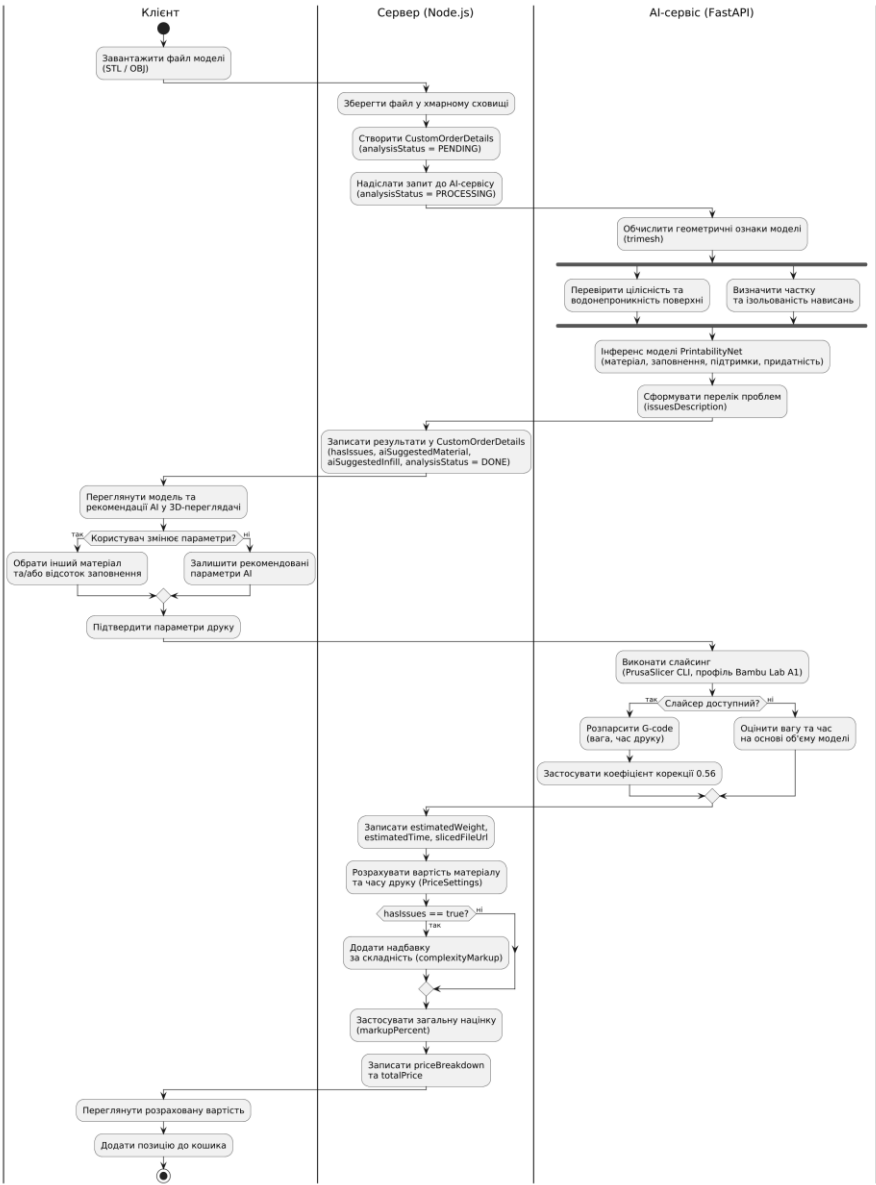


Рис.2.4. Алгоритм опрацювання кастомного замовлення з AI-асистентом.

Алгоритм демонструє, що ключовою архітектурною особливістю реалізації є розділення відповідальності між основним сервером та окремим AI-сервісом: основний сервер відповідає за збереження стану замовлення та фінансові розрахунки, тоді як усі обчислювально витратні операції – аналіз геометрії, інференс нейронної мережі та слайсинг – виконуються в ізолюваному процесі. Така відмовостійкість безпосередньо відповідає нефункціональній вимозі:

тимчасова недоступність AI-сервісу не порушує роботи решти функціональних блоків системи, оскільки взаємодія відбувається через окремий REST-запит, а не вбудований виклик у межах одного процесу.

Також алгоритм описує взаємодію серверної частини з фізичним обладнанням – 3D-принтером Bambu Lab A1, через підтримку мережових протоколів FTPS та MQTT. Алгоритм охоплює дві фази, що відрізняються характером взаємодії: одноразову команду запуску друку, що ініціюється оператором, та подальший безперервний моніторинг стану принтера засобами фонових сервісів.

Алгоритм розпочинається з дії оператора на сторінці управління принтерами, де відображається черга позицій замовлень зі статусом QUEUED, для яких вже завершено підготовку файлу друку (поле slicedFileUrl таблиці CustomOrderDetails заповнене, або файл доступний з таблиці CatalogSlicedFile для каталожних позицій). Оператор обирає конкретну позицію замовлення та вільний принтер зі статусом IDLE, після чого сервер створює новий запис у таблиці PrintJob із початковим статусом QUEUED та зв'язками orderId і printerId, що відповідають обраній позиції та обраному пристрою.

Передача файлу на принтер реалізована за двоетапною схемою, описаною в першому розділі. На першому етапі сервер встановлює захищене з'єднання за протоколом FTPS, використовуючи IP-адресу принтера та код доступу (поля ipAddress і accessCode таблиці Printer), та завантажує підготовлений файл G-code на SD-карту принтера. Після успішного завершення завантаження сервер публікує MQTT-команду запуску друку у відповідний топик принтера, використовуючи захищений протокол MQTTS. У разі успішної публікації команди статус запису PrintJob змінюється з QUEUED на PRINTING, а відповідний статус позиції замовлення в таблиці Order також переводиться у значення PRINTING, що стає видимим клієнту в історії замовлень.

Після запуску друку алгоритм переходить у фазу безперервного моніторингу, що виконується фоновим сервісом, який працює незалежно від запитів користувачів. Цей сервіс підписується на топіки телеметрії всіх

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

принтерів, зареєстрованих у таблиці Printer, реалізуючи архітектурний патерн Publish/Subscribe, замість періодичного опитування кожного принтера сервер отримує оновлення лише тоді, коли дані фактично змінюються на стороні обладнання. Кожне отримане повідомлення телеметрії містить поточну температуру сопла та стола, а також відсоток виконання поточного завдання друку.

При надходженні повідомлення телеметрії фоновий сервіс оновлює поля nozzleTemp та bedTemp таблиці Printer, а також поле progress активного запису PrintJob, що відповідає принтеру – джерелу повідомлення. Оновлені значення прогресу відображаються на клієнтській частині в реальному часі без необхідності оновлення сторінки, що відповідає нефункціональній вимозі щодо інтерактивності інтерфейсу. Алгоритм моніторингу продовжується доти, доки від принтера не надійде повідомлення про завершення завдання – успішне або з помилкою.

У разі надходження повідомлення про успішне завершення друку фоновий сервіс встановлює для запису PrintJob статус DONE та заповнює поле finishedAt поточною часовою позначкою, а пов'язана позиція замовлення в таблиці Order отримує статус READY, що сигналізує оператору про готовність виробу до подальшої обробки – пакування та передачі на відправку. Саме завершення PrintJob зі статусом DONE є подією, що ініціює третій алгоритм підрозділу – автоматичне списання філаменту зі складу.

У разі виникнення помилки під час друку (наприклад, відхилення сопла від моделі або закінчення філаменту) принтер публікує відповідне повідомлення в топик помилок, на яке також підписаний фоновий сервіс. У такому випадку статус PrintJob змінюється на FAILED, а статус принтера в таблиці Printer – на ERROR. Позиція замовлення в таблиці Order залишається у статусі PRINTING до втручання оператора, який після фізичного огляду принтера приймає рішення про повторний запуск друку (що створює новий запис PrintJob) або про скасування позиції із зазначенням причини в полі cancelReason. Наглядно зображено роботу алгоритму на рисунку 2.5.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				46
Змн.	Арк.	№ докум.	Підпис	Дата		

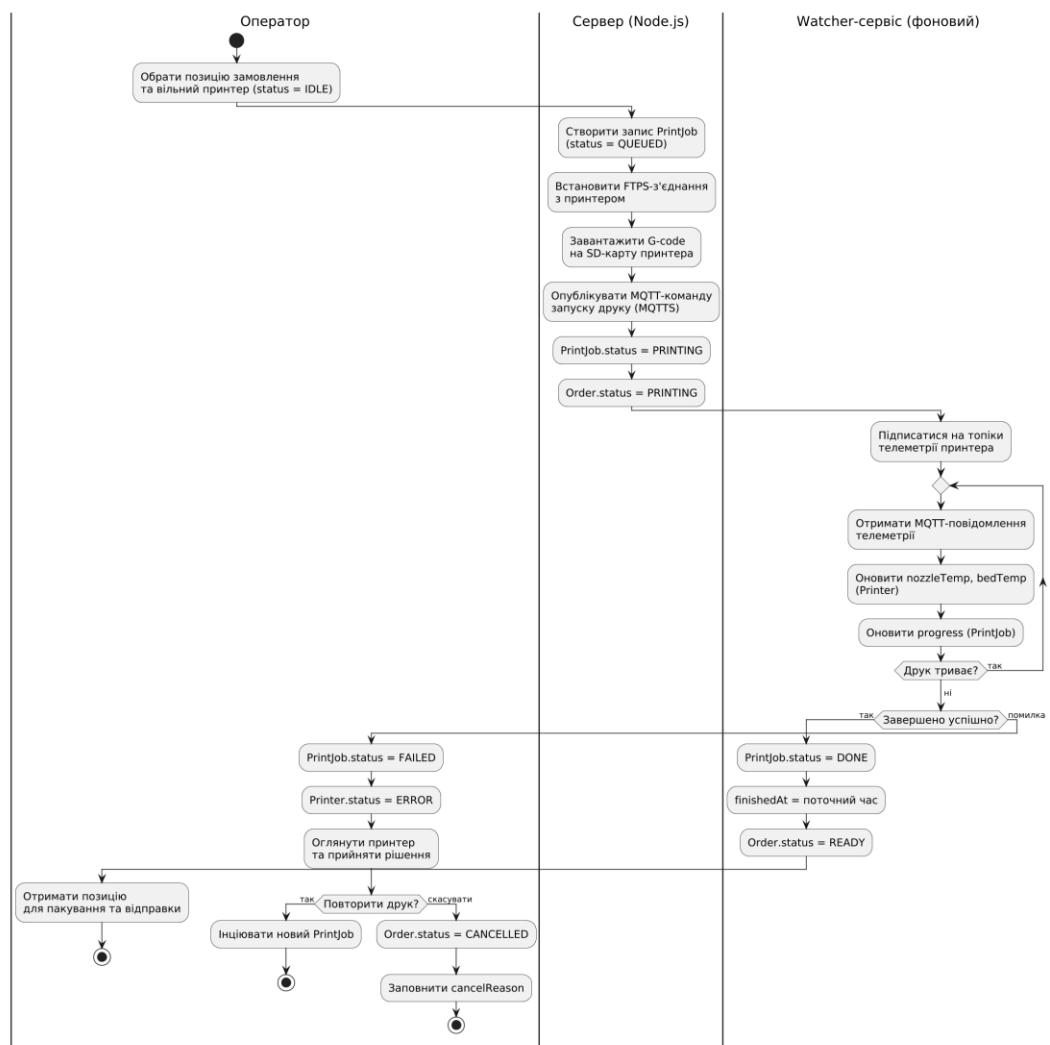


Рис.2.5. Алгоритм взаємодії серверної частини з фізичним обладнанням 3D-принтером Bambu Lab A1.

Описаний алгоритм демонструє практичну реалізацію подійно-орієнтованого підходу, як перспективний напрямок розвитку архітектури: оновлення стану принтера та завдання друку відбувається не за результатом прямого запиту користувача, а як реакція на зовнішню подію – повідомлення телеметрії або завершення друку, отримане через MQTT. Це дозволяє декільком модулям системи (інтерфейс оператора, історія замовлень клієнта, модуль складського обліку) реагувати на одну й ту саму подію незалежно один від одного, без додавання прямих викликів між відповідними контролерами.

Третій алгоритм підрозділу описує реакцію системи на подію завершення друку та реалізує бізнес-логіку складського обліку, як один із ключових елементів автоматизації сервісу. На відміну від двох попередніх алгоритмів, що

охоплювали кілька компонентів системи та зовнішні протоколи взаємодії, цей алгоритм виконується повністю на серверній частині та є відносно компактним, проте має визначальне значення для підтримки актуальності даних про залишки матеріалів на складі.

Алгоритм ініціюється подією, отриманою в результаті виконання попереднього алгоритму, – переходом запису PrintJob у статус DONE. Ця подія є точкою з'єднання двох алгоритмів підрозділу: завершення друку, зафіксоване фоновим сервісом моніторингу, одночасно є початковою умовою для списання матеріалу.

Отримавши подію завершення друку, сервер визначає позицію замовлення (Order), пов'язану із завершеним завданням PrintJob, та її тип – CATALOG або CUSTOM. Залежно від типу позиції сервер звертається до відповідної таблиці деталізації: для каталожних позицій – до CatalogOrderItem, для кастомних – до CustomOrderDetails, – оскільки саме в цих таблицях зберігається поле materialWeightUsed, що містить фактичну вагу витраченого матеріалу для конкретної позиції. Для кастомних замовлень це значення відповідає вазі, отриманій під час слайсингу та збереженій у полі estimatedWeight на етапі, описаному в першому алгоритмі, для каталожних позицій – ваговому параметру, зафіксованому в момент оформлення замовлення.

Наступним кроком сервер звертається до таблиці OrderFilament, що, є єдиним механізмом прив'язки конкретної катушки філаменту до позиції замовлення незалежно від її типу – через зовнішній ключ catalogOrderItemId або customOrderDetailsId відповідно. Якщо для позиції було обрано два кольори філаменту (основний та допоміжний, що визначається полем role типу FilamentRole), алгоритм опрацьовує кожен пов'язаний запис Filament окремо, розподіляючи загальну вагу materialWeightUsed між основним та допоміжним матеріалом пропорційно до даних, отриманих від AI-сервісу під час аналізу моделі.

Для кожної визначеної катушки філаменту сервер зменшує значення поля weightLeft таблиці Filament на відповідну частину витраченої ваги. Після

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

оновлення залишку алгоритм виконує перевірку отриманого значення `weightLeft` відносно порогового значення, заданого в полі `minWeightAlert` тієї ж котушки. Якщо новий залишок виявляється меншим або рівним пороговому значенню, система формує сповіщення для персоналу про критично низький залишок конкретного матеріалу та кольору, що відповідає функціональній вимозі. У реалізованому прототипі таке сповіщення відображається у вигляді індикатора на сторінці управління складом філаменту, доступній ролям OPERATOR, MANAGER та ADMIN; архітектура системи, описана в першому розділі, допускає подальше розширення цього кроку до надсилання сповіщень через окремий канал, наприклад електронну пошту чи систему миттєвих повідомлень, без зміни логіки самого алгоритму списання.

Якщо оновлений залишок `weightLeft` котушки досягає нульового або від'ємного значення, що теоретично може статися у разі некоректного попереднього обліку, алгоритм додатково встановлює для відповідного запису `Filament` значення `isActive` рівним `false`, виключаючи таку котушку з переліку доступних для нових замовлень до моменту фізичного поповнення складу та коригування даних персоналом. Такий механізм автоматичної деактивації виключає ситуацію, за якої клієнт міг би обрати на сторінці кастомного замовлення колір філаменту, фізична наявність якого на складі вже не підтверджена, і убезпечує систему від прийняття замовлень, виконання яких неможливе без додаткового завезення матеріалу. На рисунку 2.6 зображено як саме працює даний алгоритм.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		49

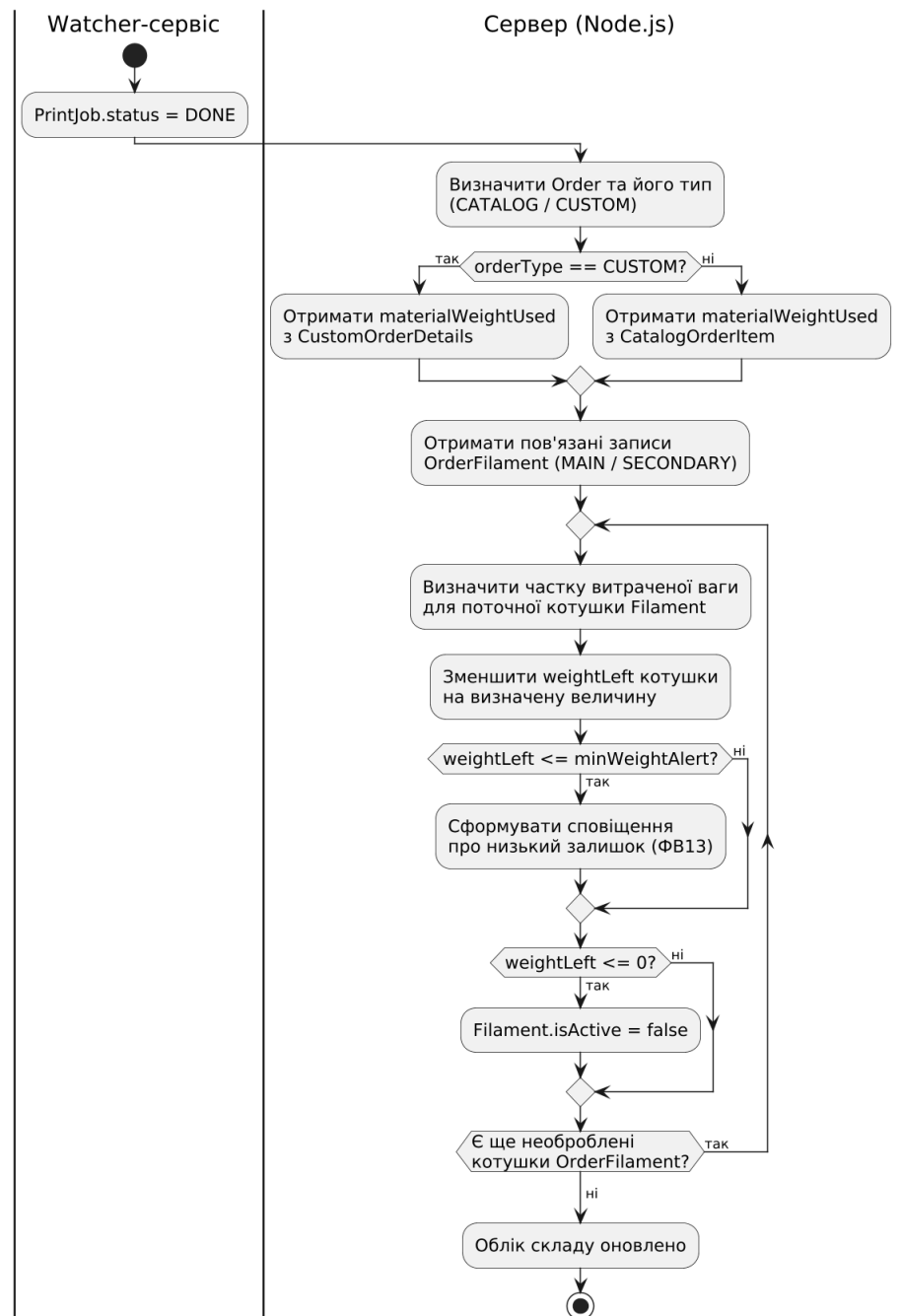


Рис.2.6. Алгоритм складського обліку.

Описаний алгоритм завершує взаємопов'язані сценарії, а саме перший алгоритм формує дані про витрату матеріалу на етапі підготовки замовлення, другий алгоритм фіксує момент фактичного завершення друку через подійно-орієнтовану взаємодію з обладнанням, а третій – перетворює цю подію на оновлення складського обліку без додаткового втручання персоналу. Така послідовність ілюструє наскрізну автоматизацію виробничого циклу – від

завантаження моделі клієнтом до актуалізації залишків матеріалів на складі, – що є основним результатом проєктних рішень.

2.4 Реалізація програмного додатку

Реалізація програмного додатку охоплює чотири взаємопов'язані частини системи, кожна з яких відповідає за окрему зону відповідальності в межах загальної архітектури. Компонентна структура застосунку наведена на рисунку 2.7 та відображає три великі програмні блоки – клієнтську частину, серверну частину та AI-сервіс, – а також їхні зв'язки із зовнішніми залежностями: реляційною базою даних, хмарним сховищем файлів та фізичним обладнанням принтера.

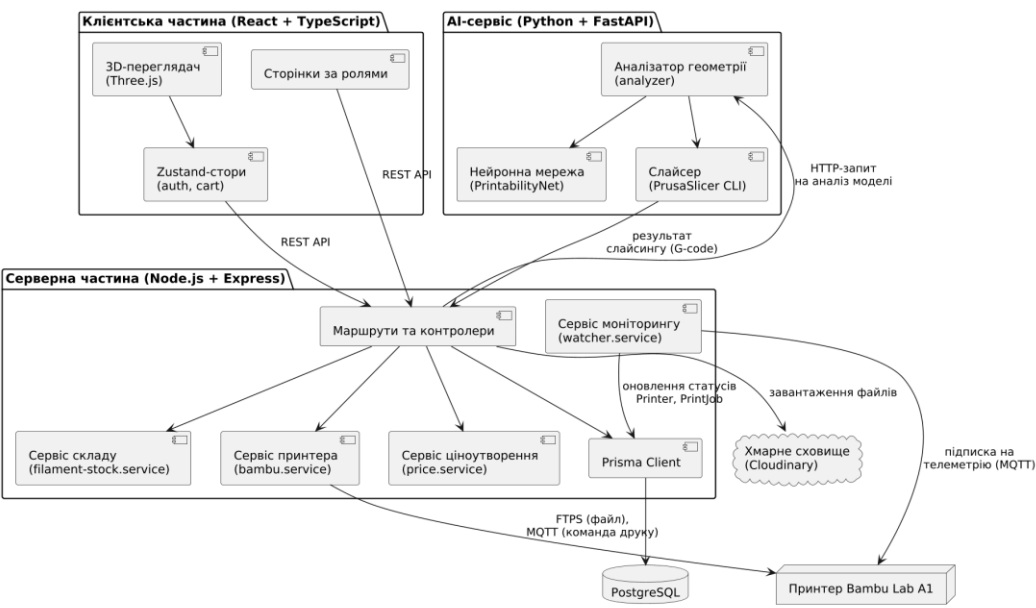


Рис.2.7. Компонентна структура проєкту.

Найбільш технічно насиченою частиною реалізації є AI-сервіс, оскільки саме в ньому виконується геометричний аналіз моделі, інференс нейронної мережі та підготовка файлу до друку. Сервіс реалізований на основі FastAPI та структурований за окремими модулями – аналізатор геометрії, обгортка над нейронною мережею та модуль слайсингу.

Перед передачею ознак на вхід мережі формується вектор із чотирнадцяти числових характеристик моделі у фіксованому порядку, серед яких габаритні розміри, об'єм, площа поверхні, мінімальна товщина стінок, частка нависань та окрема ознака наявності ізольованих нависань – ділянок моделі без опори, що

потребують додаткових підтримок. Одна з ознак, співвідношення сторін моделі, не зберігається в геометричному аналізі, а обчислюється безпосередньо перед інференсом:

```
aspect_ratio = max(features.size_x, features.size_y, features.size_z) / (
    min(features.size_x, features.size_y, features.size_z) + 0.001
)
```

```
feature_vector = np.array([
    features.size_x,
    features.size_y,
    features.size_z,
    features.volume,
    features.surface_area,
    features.vertex_count,
    features.face_count,
    features.min_wall_thickness,
    features.overhang_ratio,
    int(features.is_watertight),
    features.estimated_weight_pla,
    features.complexity_score,
    aspect_ratio,
    int(features.has_isolated_overhangs),
])
```

```
feature_scaled = scaler.transform(feature_vector)
```

Після масштабування вектора збереженим засобом нормалізації даних модель повертає чотири незалежні результати через окремі вихідні голови: ймовірності належності до одного з чотирьох матеріалів, рекомендований відсоток заповнення, потребу в підтримках та загальну оцінку придатності моделі до друку. Клас з найвищою ймовірністю обирається як рекомендований матеріал, а відсоток заповнення додатково обмежується діапазоном від 10 до 80, щоб уникнути нереалістичних значень на межах розподілу навчальних даних.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

Архітектура з чотирма головами дозволяє мережі використовувати спільні закономірності геометрії для всіх чотирьох прогнозів одночасно, що підвищує якість кожного окремого результату порівняно з чотирма незалежними моделями.

Завершальним етапом обробки моделі є слайсинг – перетворення моделі у керівну програму для принтера з урахуванням обраного матеріалу та заповнення. Перед слайсингом модель проходить попередню підготовку: опускається на робочий стіл за нижньою межею координати Z та центрується відносно координат столу принтера, оскільки вихідне положення моделі у файлі STL довільне та не відповідає робочій області конкретного принтера. Слайсинг виконується через запуск PrusaSlicer у режимі командного рядка з профілем Bambu Lab A1 та температурними параметрами, залежними від обраного матеріалу. Якщо процес слайсера завершується з помилкою або перевищує тайм-аут, він примусово завершується, а розрахунок виконується резервним методом на основі обчисленого раніше об'єму моделі.

Отриманий G-code містить метадані, з яких видобувається фактична витрата матеріалу та орієнтовний час друку. Ключовим практичним рішенням є застосування коригувального коефіцієнта до розрахованої ваги:

```
DENSITIES = {"PLA": 1.24, "PETG": 1.27, "ABS": 1.05, "TPU": 1.21}
```

```
WEIGHT_CORRECTION = 0.56
```

```
cm3_match = re.search(r'filament used \[cm3\] = ([\d.]+)', content)
```

```
if cm3_match:
```

```
    volume_cm3 = float(cm3_match.group(1))
```

```
    density = DENSITIES.get(material, 1.24)
```

```
    raw_weight = volume_cm3 * density
```

```
    data["filament_used_g"] = round(raw_weight *
```

```
    WEIGHT_CORRECTION, 2)
```

Коефіцієнт 0,56 отримано емпірично шляхом порівняння результатів PrusaSlicer із загальними профілями FDM-друку та фірмового програмного забезпечення Bambu Studio, яке використовує інші припущення щодо геометрії

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				53
Змн.	Арк.	№ докум.	Підпис	Дата		

заповнення та товщини периметрів. Без такого коригування розрахункова вартість замовлення систематично перевищувала б реальну витрату матеріалу.

Другою за значущістю частиною реалізації є підключення фізичного принтера, що відбувається у два етапи – передача файлу через протокол FTPS та публікація команди керування через протокол MQTT. Передача файлу здійснюється на захищене з'єднання в режимі "implicit", після чого надсилається команда запуску друку з параметрами автоматичних калібрувань, які принтер виконує перед стартом без участі оператора:

```
mqttClient.on('connect', () => {  
  const payload = {  
    print: {  
      command: "project_file",  
      url: `file:///sdcard/${fileName}`,  
      param: "Metadata/plate_1.gcode",  
      bed_leveling: true,  
      flow_cali: true,  
      vibration_cali: true,  
      timelapse: false,  
      use_measured_shaper: true  
    }  
  }  
  mqttClient.publish(topic, JSON.stringify(payload), { qos: 1 })  
})
```

Між завантаженням файлу та надсиланням команди запуску передбачена пауза тривалістю двадцять секунд, що дозволяє внутрішній системі принтера завершити обробку нового файлу на SD-карті перед отриманням команди друку – таке рішення отримано емпірично під час тестування з реальним обладнанням, оскільки негайна публікація команди призводила до її ігнорування принтером. Повний програмний код класу BambuService, що реалізує описану взаємодію, наведено в Додатку А (Лістинг А.1)

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

Зворотний напрямок взаємодії – отримання даних від принтера – реалізований сервісом моніторингу, що для кожного зареєстрованого принтера підтримує окреме MQTT-з'єднання та підписку на топик телеметрії. Отримане повідомлення містить поле `gcode_state`, значення якого відображається на статус принтера в базі даних:

```
const payload = JSON.parse(message.toString()) as BambuReportPayload
if (!payload.print) return
const { gcode_state, nozzle_temper, bed_temper } = payload.print
let dbStatus: 'IDLE' | 'PRINTING' | 'ERROR' | 'OFFLINE' = 'IDLE'
if (gcode_state === 'RUNNING' || gcode_state === 'PREPARE' ||
gcode_state === 'PAUSE') {
    dbStatus = 'PRINTING'
} else if (gcode_state === 'FAILED') {
    dbStatus = 'ERROR'
}
await prisma.printer.update({
    where: { id: printer.id },
    data: {
        status: dbStatus,
        nozzleTemp: nozzle_temper ?? null,
        bedTemp: bed_temper ?? null
    }
})
```

Окрім стану виконання друку, сервіс відстежує подію закриття з'єднання – у разі втрати зв'язку з принтером його статус автоматично змінюється на OFFLINE, а температурні показники скидаються, що дозволяє оператору відрізнити фізичну недоступність обладнання від звичайного стану очікування. Повний програмний код класу `PrinterWatcherService` наведено в Додатку А (Лістинг А.2).

Серверна логіка оформлення замовлення приймає весь вміст кошика користувача та створює групу замовлення разом з усіма пов'язаними позиціями через єдиний вкладений запит до бази даних, атомарність якого забезпечується самим механізмом Prisma. Окремої уваги заслуговує автоматичне визначення способу оплати для дорогих замовлень:

```
const requiresDeposit = totalPrice > 200
const depositAmount = requiresDeposit
  ? Math.round(totalPrice * 0.5)
  : totalPrice
const orderGroup = await prisma.orderGroup.create({
  data: {
    userId: req.user?.userId || null, guestEmail: req.user ? null : guestEmail,
    guestPhone: guestPhone, status: 'PENDING', totalPrice, deliveryAddress,
    deliveryService, paymentMethod: 'ONLINE',
    payments: {
      create: {
        amount: depositAmount,
        type: requiresDeposit ? 'DEPOSIT' : 'FULL',
        status: 'PENDING',
        liqpayOrderId:
`order_${Date.now()}_${Math.random().toString(36).slice(2)}`,
      },
    },
    orders: { { create: items.map(/* CUSTOM → customDetails, CATALOG
→ catalogDetails */) }
  })
```

Замовлення на суму, що перевищує двісті гривень, автоматично переводиться в режим часткової передоплати у розмірі половини вартості, тоді як менші замовлення оплачуються повністю одразу – такий підхід балансує між зниженням бар'єру для дрібних замовлень та фінансовим захистом сервісу при виконанні великих кастомних виробів, підготовка яких потребує значних витрат

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		56

матеріалу ще до підтвердження оплати клієнтом. Після успішного створення групи замовлення виконується послідовне списання філаменту зі складу для кожної позиції, що містить обрану катушку, на основі ваги, отриманої від AI-сервісу під час слайсингу або зафіксованої для товару каталогу.

На клієнтській частині ключовим елементом інтерактивності є тривимірний переглядач моделі на основі бібліотеки Three.js, підписаний на стан стору параметрів друку: зміна користувачем відсотка заповнення чи матеріалу в бічній панелі оновлює значення в централізованому Zustand-сторі, а переглядач реагує на цю зміну оновленням вигляду моделі на сцені без перезавантаження сторінки – так само, як автоматично перераховується вартість замовлення на основі тих самих значень стору.

Висновки до другого розділу

У другому розділі здійснено проектування онлайн-сервісу 3D-друку з інтеграцією AI асистента, що охопило визначення варіантів використання та об'єктної структури, розробку бази даних, формалізацію ключових алгоритмів роботи та опис їхньої програмної реалізації.

Визначено рольову модель доступу та сформовано на її основі діаграму варіантів використання, що відображає ієрархію акторів від неавторизованого гостя до адміністратора системи. Узгодженість функціональних можливостей із цією ієрархією підтверджена таблицею розподілу за ролями, а якісні характеристики системи – швидкодія, безпека, надійність, масштабованість та зручність використання – формалізовано у вигляді нефункціональних вимог.

На основі визначених вимог побудовано об'єктно-орієнтовану структуру системи, ключовим рішенням якої є розділення рівня замовлення на групу (OrderGroup) та окрему позицію (Order), що дозволяє коректно обробляти різnorідний кошик із каталожних і кастомних виробів у межах однієї покупки. Ця структура деталізована до фізичної схеми бази даних із сімнадцяти таблиць, що охоплюють користувачів, замовлення та оплати, каталог, кастомні замовлення з результатами AI-аналізу, складський облік філаменту та апаратний рівень.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

Динаміку роботи системи розкрито через три алгоритми: опрацювання кастомного замовлення з геометричним аналізом моделі та слайсингом, запуск друку й моніторинг принтера через протоколи MQTT та FTPS, а також автоматичне списання філаменту зі складу за результатами завершеного друку.

Реалізація цих алгоритмів описана на рівні конкретних програмних компонентів – нейронної мережі з чотирма вихідними головами, модуля слайсингу з емпірично визначеним коригувальним коефіцієнтом, сервісів прямого підключення до принтера Bambu Lab A1 [31] за протоколами FTPS і MQTT та контролера формування замовлень з автоматичним визначенням способу оплати залежно від суми замовлення.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ОНЛАЙН-СЕРВІСОМ 3D-ДРУКУ З ІНТЕГРАЦІЄЮ AI АСИСТЕНТА

3.1 Порядок встановлення та налаштування системи

Розгортання програмного комплексу складається з трьох незалежних частин, кожна з яких запускається в окремому процесі та потребує власного середовища виконання: клієнтської частини на основі React та Vite [32], серверної частини на основі Node.js та Express, а також AI-сервісу на основі Python та FastAPI. Для коректної роботи всіх компонентів попередньо повинні бути встановлені Node.js версії 18 або новішої, PostgreSQL версії 18, а також Python версії 3.10 або новішої з можливістю встановлення бібліотек для роботи з нейронними мережами та обробки тривимірної геометрії.

Розгортання розпочинається з налаштування бази даних. Після встановлення PostgreSQL створюється нова база даних, а рядок підключення до неї вказується у файлі змінних середовища .env у кореневій директорії серверної частини в полі DATABASE_URL. Структура таблиць, створюється не вручну, а автоматично – засобами Prisma ORM, що зчитує файл схеми schema.prisma та генерує на його основі відповідні SQL-команди створення таблиць, типів-перерахувань та зв'язків між ними. Застосування схеми до порожньої бази даних виконується командою `prx prisma migrate dev`, яка водночас створює службову таблицю історії міграцій, що дозволяє надалі вносити зміни до структури бази даних поступово, без втрати наявних даних.

Запуск серверної частини здійснюється у директорії backend командою встановлення залежностей `npm install`, після чого команда `npm run dev` запускає сервер у режимі розробки на порту 5000 із автоматичним перезавантаженням при зміні файлів. Аналогічним чином у директорії frontend команди `npm install` та `npm run dev` запускають клієнтську частину на порту 5173 засобами Vite.

Запуск AI-сервісу потребує окремого підготовленого середовища Python, оскільки бібліотеки для роботи з нейронними мережами та геометрією моделей мають значний обсяг і можуть конфліктувати з іншими проєктами на тому самому комп'ютері. Тому перед запуском створюється віртуальне середовище

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		59

командою `python -m venv venv`, яке активується окремо для кожної нової сесії роботи з сервісом, після чого командою `pip install -r requirements.txt` встановлюються всі необхідні залежності, включно з бібліотеками для роботи з нейронними мережами та обробки тривимірної геометрії. Запуск самого сервісу виконується командою `uvicorn main:app --reload` та робить його доступним на порту 8000. Для коректної роботи модуля слайсингу, додатково потрібна наявність встановленого застосунку PrusaSlicer, шлях до виконуваного файлу якого вказується в конфігурації модуля слайсингу.

Окремої уваги потребує налаштування підключення до фізичного принтера Bambu Lab A1. Для встановлення зв'язку за протоколами FTPS та MQTT, адміністратору необхідно отримати з меню налаштувань мережі на самому принтері три значення: IP-адресу пристрою в локальній мережі, код доступу (Access Code), що відображається безпосередньо на екрані принтера, та серійний номер пристрою, що використовується для формування шляху до топиків MQTT. Ці три значення вносяться через адміністративну панель додатку при додаванні нового принтера до системи та зберігаються у відповідних полях таблиці Printer. Принтер повинен бути підключений до тієї самої локальної мережі, що й сервер, оскільки протоколи FTPS та MQTT у режимі прямого підключення, обраному для цього проєкту, не передбачають взаємодії через мережу Інтернет без додаткового налаштування переадресації портів.

Після успішного запуску всіх трьох частин системи та підключення бази даних у ній відсутні зареєстровані користувачі, окрім випадку, якщо база даних була попередньо наповнена тестовими даними через скрипт початкового заповнення (seed). Першим кроком після розгортання є реєстрація облікового запису через стандартну форму реєстрації, доступну на головній сторінці – такий користувач автоматично отримує роль CLIENT, що відповідає ролі звичайного клієнта. Підвищення ролі до OPERATOR, MANAGER або ADMIN виконується вручну: або безпосередньо в базі даних через інструмент адміністрування PostgreSQL для першого облікового запису адміністратора, або, якщо в системі вже існує хоча б один користувач з роллю ADMIN, – через відповідну сторінку

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		60

керування користувачами в адміністративній панелі, де адміністратор обирає нову роль зі списку для будь-якого зареєстрованого облікового запису.

Розмежування прав доступу, визначене таблицею ролей, проявляється відразу після входу користувача в систему: навігаційне меню динамічно формується залежно від ролі поточного користувача, відображаючи лише ті розділи, доступ до яких йому дозволений. Так, користувач з роллю CLIENT після входу бачить пункти меню для переходу до каталогу, оформлення кастомного замовлення та перегляду власних замовлень, тоді як користувач з роллю OPERATOR додатково бачить пункти управління завданнями на друк, складом філаменту та принтерами, а спроба прямого переходу за посиланням на сторінку, недоступну поточній ролі, перенаправляє користувача на головну сторінку засобами захищених маршрутів клієнтської частини. На рисунку 3.1 зображена діаграма розгортання системи.

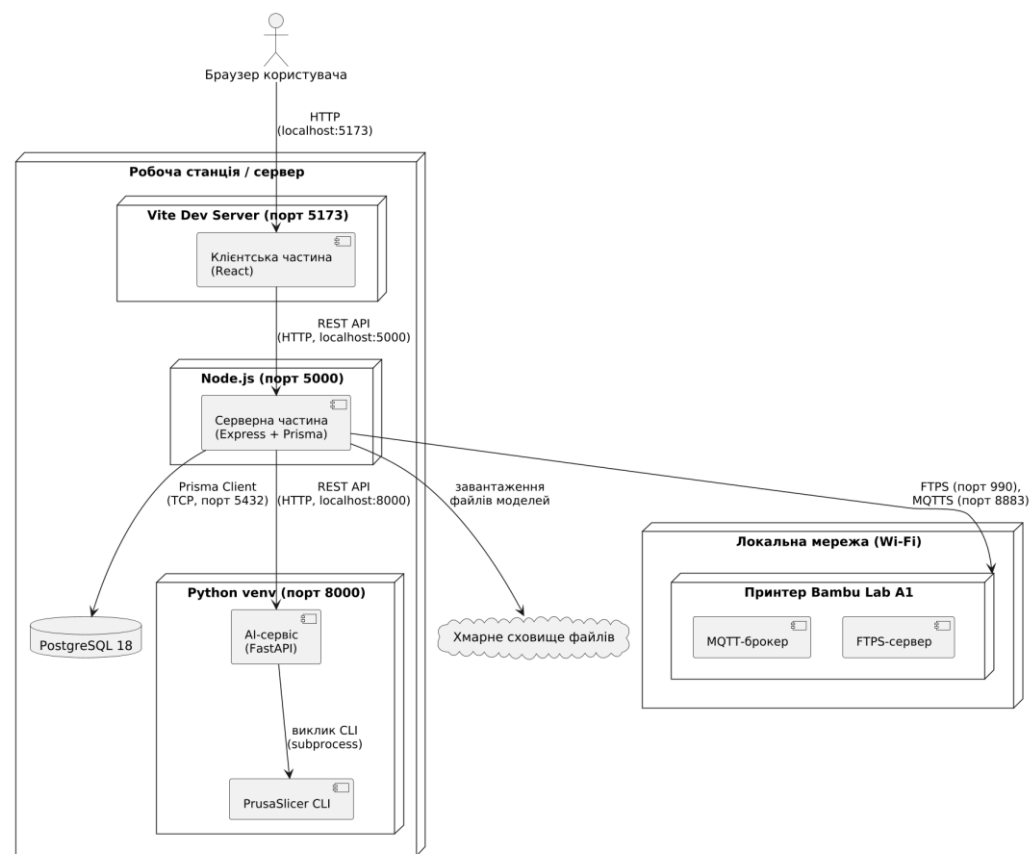


Рис.3.1. Діаграма розгортання системи.

Діаграма розгортання демонструє, що в межах поточної конфігурації прототипу всі три програмні частини – клієнтська, серверна та AI-сервіс – а

також база даних розгортаються на одній фізичній робочій станції, кожна у власному процесі та на власному порту, що відповідає кроку запуску, описаному вище. Браузер користувача взаємодіє виключно з клієнтською частиною через локальний порт 5173, тоді як уся подальша взаємодія з базою даних, хмарним сховищем та AI-сервісом відбувається на стороні сервера й не є безпосередньо доступною з боку клієнта, що відповідає вимозі безпеки щодо обмеження доступу до файлів моделей лише авторизованому персоналу.

Окремий вузол на діаграмі – локальна мережа з принтером Bambu Lab A1, підкреслює ключову умову, а саме серверна частина та принтер повинні перебувати в одній локальній мережі, оскільки взаємодія за протоколами FTPS (порт 990) та MQTTS (порт 8883) у режимі прямого підключення відбувається без проміжних хмарних посередників виробника обладнання. Це архітектурне рішення, обґрунтоване в першому розділі як перевага прямої інтеграції, на рівні розгортання означає відсутність додаткових мережевих залежностей – принтер залишається повністю керованим у межах локальної інфраструктури сервісу.

3.2 Структура інтерфейсу користувача

Інтерфейс користувача побудований за принципом єдиного застосунку, де перехід між сторінками відбувається без повного перезавантаження браузера, а вигляд навігаційного меню та доступні розділи динамічно змінюються залежно від ролі поточного користувача. Структура інтерфейсу описана в порядку, що відповідає типовому шляху користувача в системі – від головної сторінки та каталогу до особистого кабінету та адміністративних панелей.

Головна сторінка (рис. 3.2) є точкою входу для будь-якого відвідувача незалежно від того, чи він авторизований. Вона містить короткий опис можливостей сервісу та посилання на основні розділи – каталог готових виробів та форму кастомного замовлення. Для неавторизованого користувача в правому верхньому куті навігаційної панелі відображаються посилання на сторінки реєстрації та входу, тоді як для авторизованого користувача на їхньому місці відображається іконка профілю з випадаючим меню, що веде до особистого кабінету.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		62

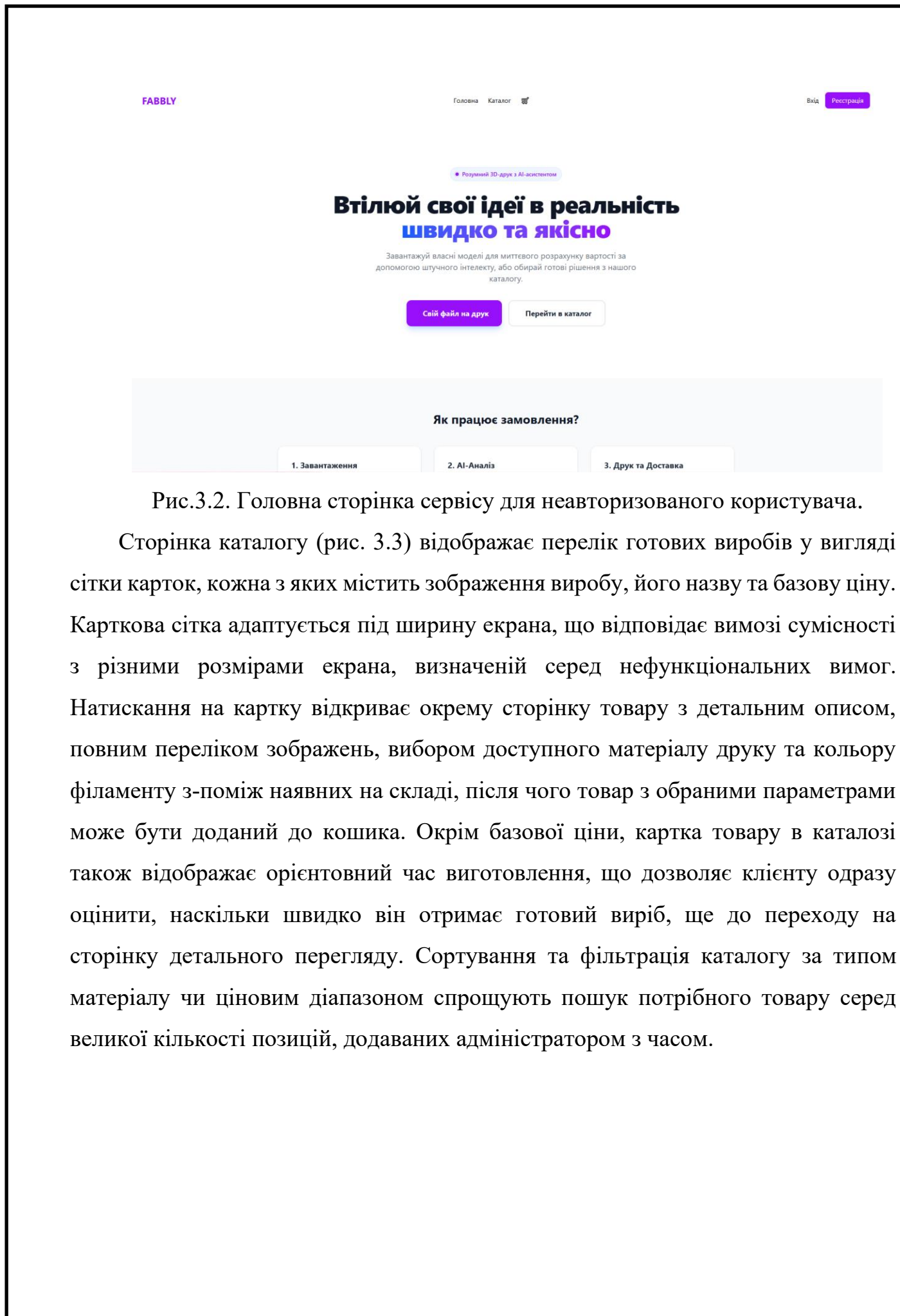


Рис.3.2. Головна сторінка сервісу для неавторизованого користувача.

Сторінка каталогу (рис. 3.3) відображає перелік готових виробів у вигляді сітки карток, кожна з яких містить зображення виробу, його назву та базову ціну. Карткова сітка адаптується під ширину екрана, що відповідає вимозі сумісності з різними розмірами екрана, визначеній серед нефункціональних вимог. Натискання на картку відкриває окрему сторінку товару з детальним описом, повним переліком зображень, вибором доступного матеріалу друку та кольору філаменту з-поміж наявних на складі, після чого товар з обраними параметрами може бути доданий до кошика. Окрім базової ціни, картка товару в каталозі також відображає орієнтовний час виготовлення, що дозволяє клієнту одразу оцінити, наскільки швидко він отримає готовий виріб, ще до переходу на сторінку детального перегляду. Сортування та фільтрація каталогу за типом матеріалу чи ціновим діапазоном спрощують пошук потрібного товару серед великої кількості позицій, додаваних адміністратором з часом.

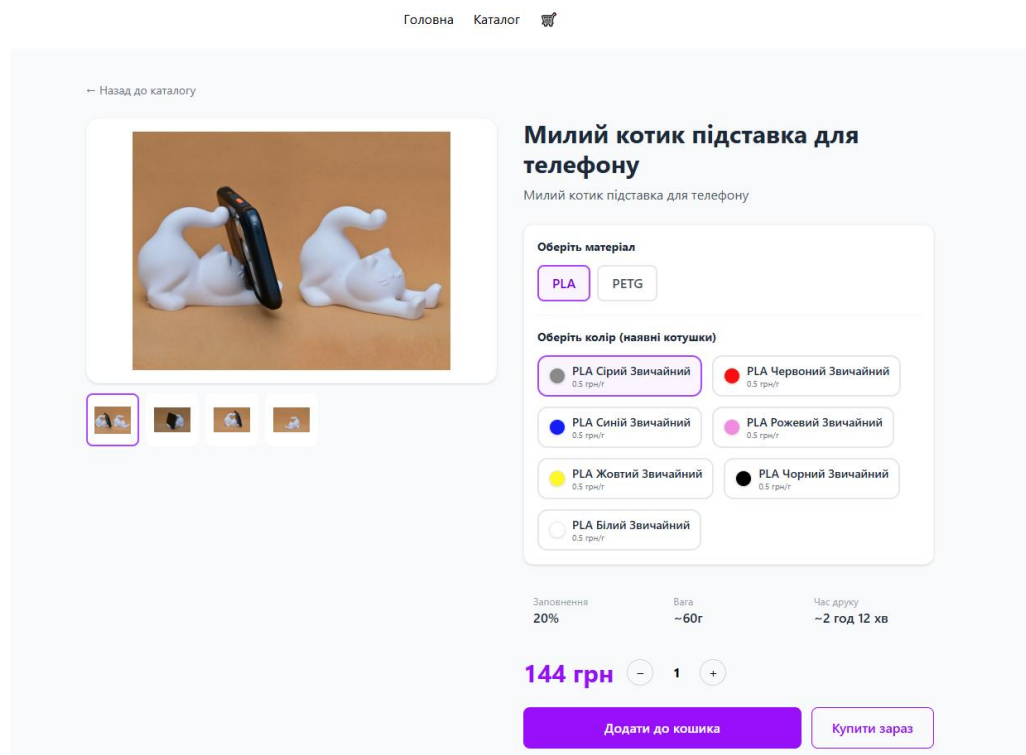


Рис.3.3. Сторінка каталогу готових виробів із карткою конкретного товару.

Центральним та найбільш функціонально насиченим розділом клієнтської частини є сторінка кастомного замовлення, яка наведена на рисунку 3.4. Вона складається з двох логічних зон де розміщено інтерактивний тривимірний переглядач завантаженої моделі на основі Three.js з можливістю обертання, масштабування та переміщення сцени, та іншої частини, де відображаються результати AI-аналізу – виявлені проблеми геометрії, рекомендований матеріал, рекомендований відсоток заповнення та потреба в підтримках. Саме на цій сторінці реалізована інтерактивність, зміна користувачем параметра заповнення або матеріалу в правій панелі миттєво відображається на вигляді моделі в переглядачі та на розрахунковій вартості, без необхідності повторного запиту до сервера для оновлення інтерфейсу. Така миттєва візуальна реакція моделі на дії користувача є прямою практичною реалізацією вимоги щодо зручності використання інтерфейсу, визначеної серед нефункціональних вимог проєктованої системи.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

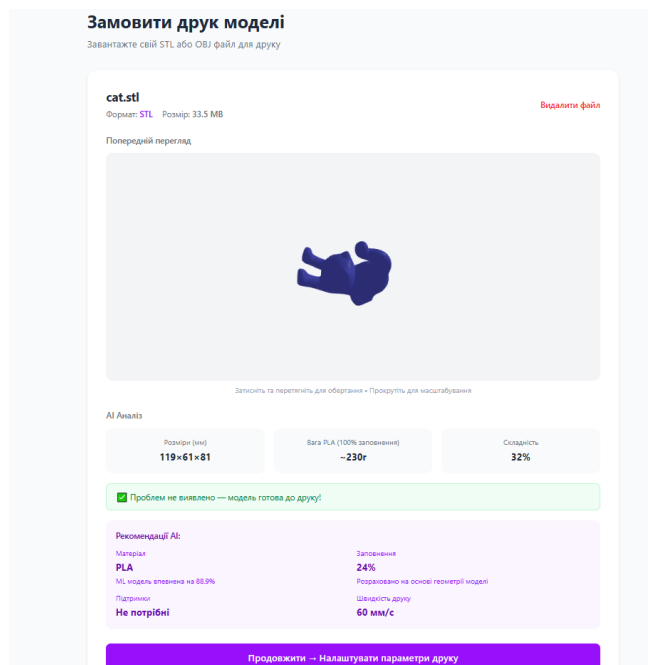


Рис.3.4. Сторінка кастомного замовлення з завантаженою тривимірною моделлю.

Після завершення налаштування параметрів та розрахунку вартості позиція додається до кошика – окремої сторінки (рис. 3.5), що відображає перелік усіх обраних товарів незалежно від їхнього типу (з каталогу або кастомних), із можливістю зміни кількості або видалення позиції та підсумковою сумою замовлення. Зі сторінки кошика користувач переходить до оформлення замовлення (рис. 3.6), де вказує адресу доставки, обирає службу доставки та для неавторизованого користувача додатково вводить контактні дані, що відповідає гостьовому сценарію оформлення.

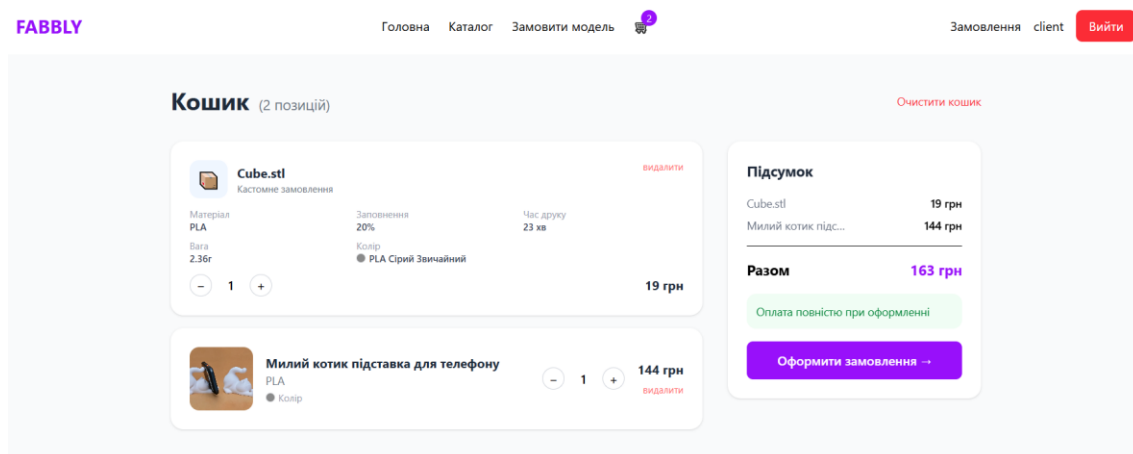


Рис.3.5. Сторінка кошика з позиціями різних типів.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савицький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

Сторінка оформлення замовлення для зареєстрованого користувача представлена на рисунці 3.6.

Рис.3.6. Сторінка оформлення замовлення з вибором служби доставки.

Особистий кабінет авторизованого користувача об'єднує сторінку профілю, де відображаються та редагуються особисті дані, та сторінку історії замовлень. Кожна група замовлення в історії відображається як окремий блок із загальним статусом та розгорнутим переліком вхідних до неї позицій, для кожної з яких окремо відображається власний статус виконання – такий вигляд безпосередньо відповідає структурі OrderGroup та Order, де загальний статус групи та статус окремої позиції зберігаються в різних таблицях бази даних. Сторінка статусу конкретного замовлення наведена на рисунку 3.7.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		66

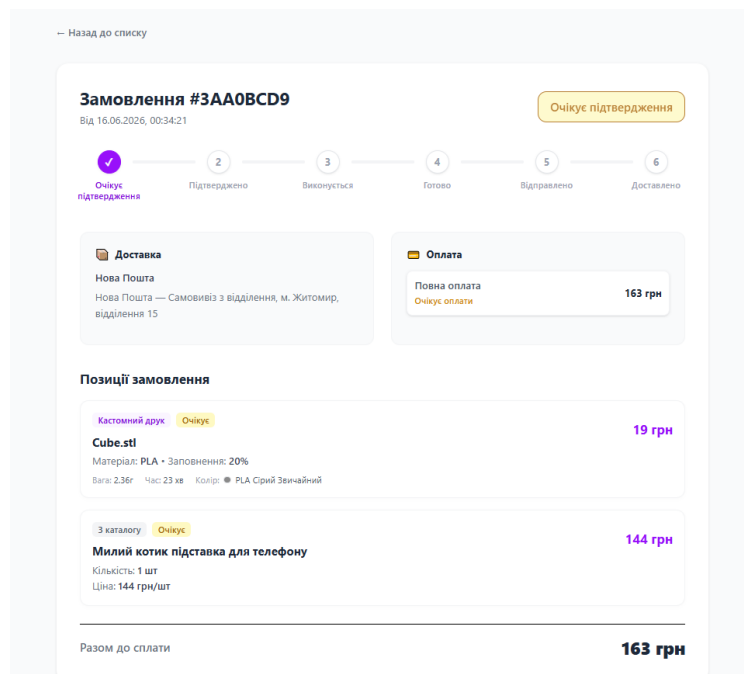


Рис.3.7. Сторінка історії замовлень авторизованого клієнта з розгорнутою групою замовлення.

Для користувачів з ролями OPERATOR, MANAGER та ADMIN навігаційне меню додатково містить розділ управління виробництвом. Сторінка управління принтерами (рис. 3.8) відображає перелік усіх зареєстрованих принтерів у вигляді карток із поточним статусом, температурою сопла та стола, що оновлюються в реальному часі завдяки сервісу моніторингу, а також черга позицій замовлень, готових до друку, з можливістю обрати позицію та принтер для запуску друку.

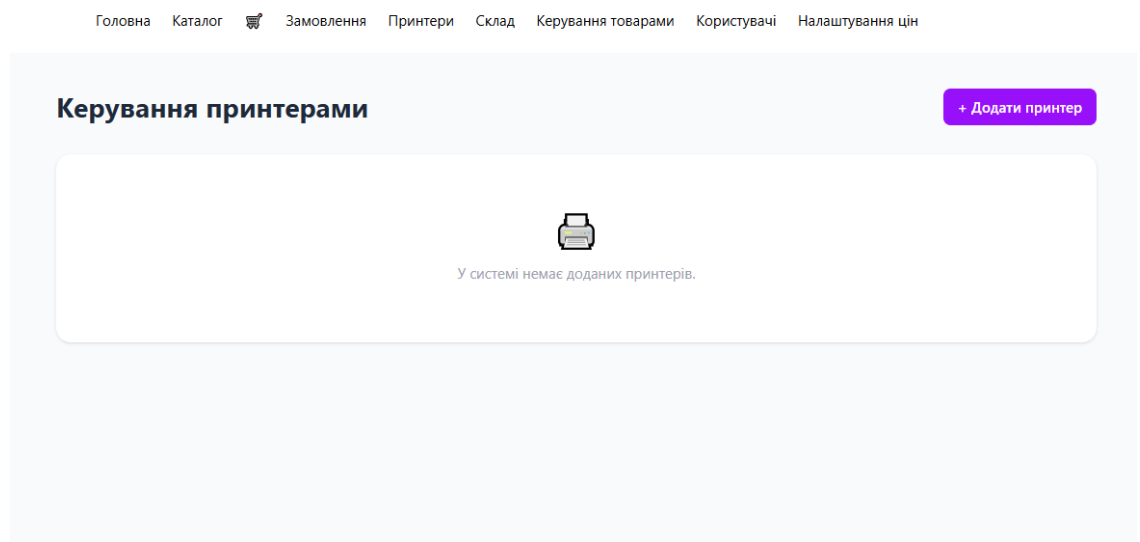


Рис.3.8. Сторінка управління принтерами.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		67

Сторінка управління складом філаменту (рис. 3.9) відображає перелік усіх котушок із зазначенням типу матеріалу, кольору, поточного залишку у грамах та індикатором, що візуально виділяє котушки з залишком, який наблизився до порогового значення сповіщення.

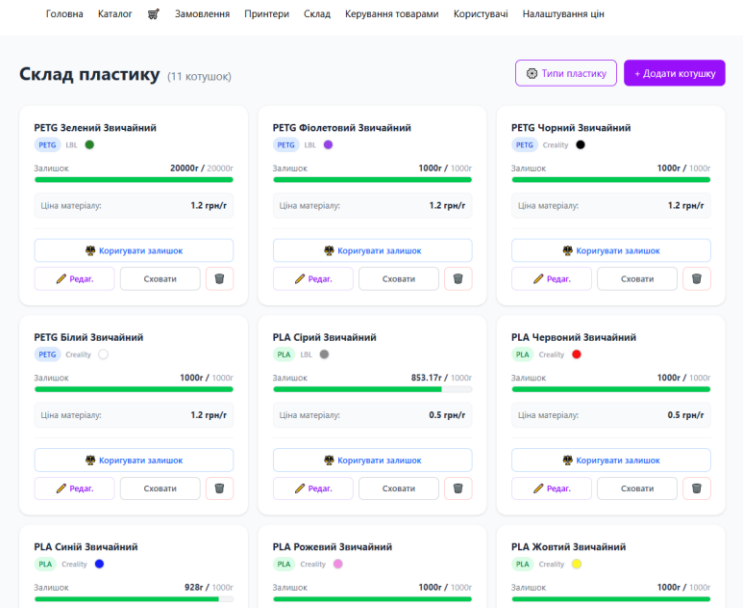


Рис.3.9. Сторінка управління складом філаменту.

Сторінка управління завданнями на друк (рис. 3.10), доступна оператору, відображає перелік позицій замовлень, призначених йому на виконання, з можливістю зміни статусу окремої позиції в процесі виробництва – саме ця сторінка реалізує перший крок алгоритму запуску друку, де оператор обирає позицію замовлення та вільний принтер.

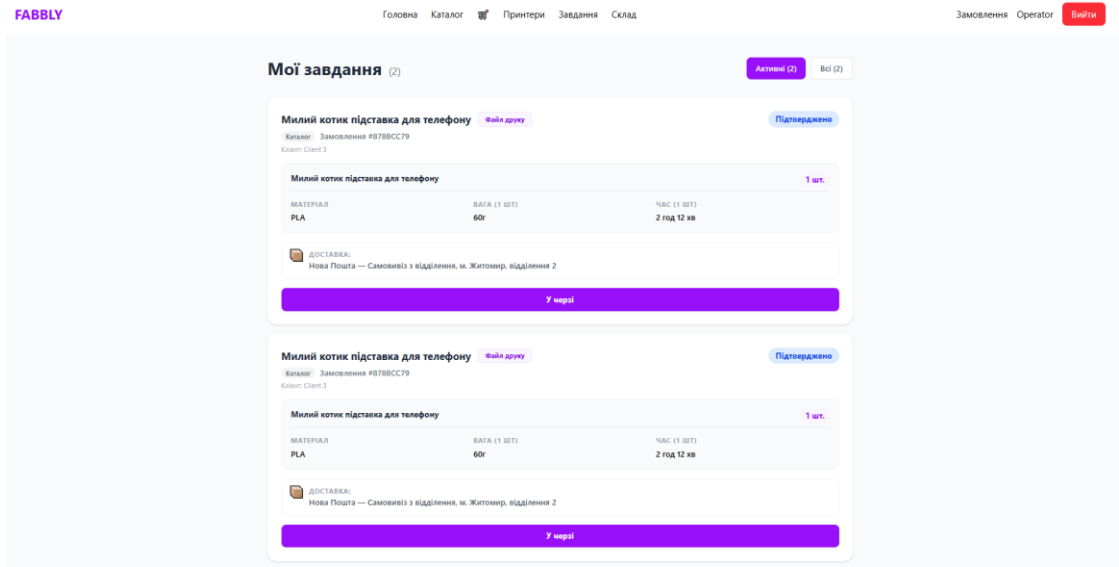


Рис.3.10. Сторінка завдань оператора.

Найвищий рівень доступу – роль ADMIN – додає до навігаційного меню розділи управління користувачами, каталогом та налаштуваннями ціноутворення. Сторінка управління користувачами (рис. 3.11) відображає таблицю всіх зареєстрованих облікових записів із можливістю зміни ролі для кожного з них, що є практичною реалізацією процесу підвищення прав доступу.

Користувачі (7)

Користувач	Email	Телефон	Дата реєстрації	Роль
C3 Client 3	client3@gmail.com	—	15.06.2026	Клієнт
O2 Operator 2	operator2@gmail.com	—	15.06.2026	Оператор
O1 Operator 1	operator1@gmail.com	—	15.06.2026	Оператор
m1 manager 1	manager1@gmail.com	—	15.06.2026	Менеджер
c2 client 2	client2@gmail.com	client2@gmail.com	15.06.2026	Клієнт
c1 client 1	client1@gmail.com	client1@gmail.com	15.06.2026	Оператор
AS Aryna Shkolna	ipz224_shal@student.ztu.edu.ua	ipz224_shal@student.ztu.edu.ua	15.06.2026	Адміністратор

Рис.3.11. Сторінка управління користувачами.

Сторінка управління каталогом (рис. 3.12) дозволяє створювати нові товари, редагувати наявні – назву, опис, ціну, перелік доступних матеріалів – та завантажувати попередньо підготовлені файли друку окремо для кожного з підтримуваних матеріалів, що відповідає структурі таблиці CatalogSlicedFile.

Управління каталогом (1) ➕ Додати товар

Новий товар

Назва Коригування ціни (грн) — +/- до автоматичного розрахунку

Опис

Посилання на фото (через клієв)

Доступні матеріали ☒ PLA ☐ PETG ☐ ABS ☐ TPU Заповнення (%)

Вага (г) Час друку (хв)

Створити товар Закрити

 Щоб завантажити 3mf файли для цього товару, спочатку натисніть "Створити товар", а потім відкрийте його для редагування.

Товар	Матеріали / Файли	Вага / Час	Ціна	Статус	Дії
 Милій копчик підставка для телефону Милій копчик підставка для теле...	PLA PETG Файли: 2 / 2	~60г 3h 12m	0 грн	Активний	 Словати 

Рис.3.12. Сторінка управління каталогом.

Завершальною є сторінка налаштувань ціноутворення (рис. 3.13), доступна виключно ролі ADMIN, що містить форму редагування глобальних параметрів формули розрахунку вартості – ціни години роботи принтера, відсотка націнки, відсотка передоплати для кастомних замовлень та надбавки за складність, – зміна яких безпосередньо впливає на розрахунок. Зміна будь-якого з цих параметрів одразу позначається на вартості всіх подальших замовлень, оформлених як через каталог, так і через кастомне завантаження моделі, оскільки обидва сценарії звертаються до однієї й тієї самої таблиці налаштувань під час розрахунку ціни. Завдяки цьому адміністратор отримує єдину точку керування економічною моделлю сервісу без потреби вносити зміни до кількох розділів системи окремо.

Головна
Каталог
Замовлення
Принтери
Склад
Керування товарами
Користувачі
Налаштування цін
Замовлення
Аркуш

Налаштування ціноутворення

Ці параметри впливають на розрахунок вартості всіх замовлень

Ціна за годину роботи принтера (грн)

Амортизація обладнання, без електрики

Вартість електроенергії (грн/год)

Споживання принтера за годину друку

Націнка / прибуток (%)

Відсоток понад собівартість матеріалу та часу

Надбавка за складність (0-1)

Максимальний відсоток надбавки для найскладніших моделей (наприклад 0.3 – до 30%)

Буфер часу друку (%)

Запас на випадок якщо реальний друк займе більше часу ніж розрахував слайсер

Передоплата для кастомних замовлень (%)

Застосовується для замовлень дорожче 200 грн

Зберегти налаштування

Рис.3.13. Сторінка налаштувань ціноутворення.

Структура навігації застосунку, наведена на рисунку 3.14.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		70

3.3 Тестування роботи програмного комплексу

Тестування реалізованих алгоритмів виконано шляхом ручної перевірки основних сценаріїв роботи системи на конкретних вхідних даних із порівнянням отриманого результату з очікуваним. Такий підхід дозволяє підтвердити коректність роботи саме тих алгоритмів, опрацювання кастомного замовлення з AI-аналізом, запуск друку та моніторинг принтера, а також списання філаменту зі складу, – оскільки кожен із цих алгоритмів об'єднує кілька компонентів системи та потребує перевірки не окремих функцій, а наскрізного сценарію в цілому.

Перший сценарій тестування охоплює повний цикл опрацювання кастомного замовлення – від завантаження файлу тривимірної моделі до отримання розрахованої вартості. Для перевірки використано тестову модель простої геометричної форми з відомими наперед габаритними розмірами, що дозволяє перевірити коректність роботи кожного етапу алгоритму окремо. Очікувані та фактично отримані результати для цього сценарію наведено в таблиці 3.1.

Таблиця 3.1

Результати тестування алгоритму обробки кастомного замовлення

Етап алгоритму	Вхідні дані	Очікуваний результат	Фактичний результат
Завантаження файлу	Файл моделі у форматі STL	Файл збережено, створено запис CustomOrderDetails зі статусом аналізу PENDING	Відповідає очікуваному
Геометричний аналіз	Модель з тонкою стінкою (менше порогового значення)	Поле hasIssues приймає значення true, у переліку проблем зазначено тонкі стінки	Відповідає очікуваному
Інференс нейронної мережі	Геометричні ознаки моделі	Сформовано рекомендації щодо матеріалу, заповнення та потреби в підтримках, статус аналізу змінено на DONE	Відповідає очікуваному

Етап алгоритму	Вхідні дані	Очікуваний результат	Фактичний результат
Слайсинг	Параметри друку, обрані користувачем (матеріал, заповнення)	Сформовано файл G-code, отримано вагу та час друку	Відповідає очікуваному
Резервна оцінка	Примусово вимкнений PrusaSlicer	Виконано розрахунок на основі об'єму моделі без помилки виконання	Відповідає очікуваному
Розрахунок вартості	Отримані вага й час, параметри з PriceSettings	Сформовано підсумкову ціну та повну розшифровку priceBreakdown	Відповідає очікуваному

Окремої перевірки потребував випадок – резервна оцінка ваги та часу друку у разі недоступності зовнішнього слайсера. Для перевірки цього випадку шлях до виконуваного файлу PrusaSlicer у конфігурації було тимчасово замінено на неіснуючий, що призводило до завершення процесу з помилкою. Алгоритм коректно перейшов до резервної функції оцінки на основі обчисленого об'єму моделі, а запис CustomOrderDetails отримав значення розрахункової ваги та часу без переривання загального процесу обробки замовлення, що підтверджує відмовостійкість, закладену в архітектуру системи.

Другий сценарій тестування стосується запуску друку та моніторингу принтера. Перевірка виконувалася з реальним обладнанням Bambu Lab A1, підключеним до тієї самої локальної мережі, що й серверна частина. Тестування охоплювало перевірку успішності автентифікації на вбудованому сервері принтера, стабільність завантаження файлу керівної програми через захищений протокол FTPS, а також точність парсингу асинхронних повідомлень у відповідних MQTT-топіках. Фіксація телеметричних даних, зокрема поточної температури сопла, платформи та відсоткового прогресу виконання, здійснювалася у реальному часі з мінімальною затримкою. Результати наведено в таблиці 3.2.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				73
Змн.	Арк.	№ докум.	Підпис	Дата		

Таблиця 3.2

Результати тестування алгоритму запуску друку та моніторингу принтера

Етап алгоритму	Вхідні дані	Очікуваний результат	Фактичний результат
Передача файлу через FTPS	Підготовлений файл G-code, коректні дані принтера (IP, код доступу)	Файл успішно завантажено на SD-карту принтера	Відповідає очікуваному
Публікація MQTT-команди	Серійний номер принтера	Принтер прийняв команду та розпочав виконання автоматичних калібрувань перед друком	Відповідає очікуваному
Публікація MQTT-команди (без паузи)	Команда надіслана негайно після завантаження файлу	Принтер ігнорує команду друку	Відповідає фактичній поведінці.
Отримання телеметрії	Принтер у процесі друку	Поля nozzleTemp, bedTemp та статус принтера в базі даних оновлюються відповідно до значень gcode_state	Відповідає очікуваному
Втрата з'єднання	Відключення принтера від мережі	Статус принтера змінюється на OFFLINE, температурні показники скидаються до null	Відповідає очікуваному

Рядок таблиці, що описує поведінку системи без паузи перед публікацією MQTT-команди, відображає не помилку реалізації, а результат попереднього тестування, що безпосередньо привів до прийнятого архітектурного рішення, цей випадок навмисно залишено в таблиці для демонстрації того, що значення параметра затримки отримано емпіричним шляхом, а не довільно.

Третій сценарій тестування охоплює алгоритм автоматичного списання філаменту зі складу. Перевірка виконувалася для котушки з відомим початковим

залишком, після чого виконувалося тестове замовлення з визначеною вагою витраченого матеріалу. Результати наведено в таблиці 3.3.

Таблиця 3.3

Результати тестування алгоритму списання філаменту

Етап алгоритму	Вхідні дані	Очікуваний результат	Фактичний результат
Визначення витраченої ваги	Завершене замовлення типу CUSTOM з полем	Отримано коректне значення ваги	Відповідає очікуваному
Оновлення залишку котушки	Котушка з початковим залишком, що перевищує порогове значення	Поле weightLeft зменшено на величину витраченої ваги	Відповідає очікуваному
Перевірка порогового значення	Залишок після списання менший за minWeightAlert	Сформовано індикатор низького залишку на сторінці управління складом	Відповідає очікуваному
Деактивація котушки	Залишок після списання дорівнює нулю	Поле isActive встановлено у значення false, котушка не доступна для нових замовлень	Відповідає очікуваному

Окремо перевірено сценарій, що не входить до основних трьох алгоритмів, але є важливим з погляду цілісності даних – скасування замовлення менеджером після фактичного списання філаменту. У цьому випадку система виконує зворотну дію – повернення списаної ваги на залишок відповідної котушки, що підтверджує узгодженість операцій списання та повернення матеріалу в межах одного механізму обліку.

Результати проведеного тестування підтверджують коректність роботи трьох алгоритмів, формалізованих у другому розділі, для типових та межових випадків вхідних даних, а також демонструють відмовостійкість системи при

недоступності зовнішніх компонентів слайсера та принтера, що відповідає нефункціональним вимогам.

Висновки до третього розділу

У третьому розділі розглянуто практичні аспекти розгортання, використання та перевірки програмного комплексу, що завершує опис проектних рішень, викладених у попередніх розділах роботи. Описано порядок встановлення трьох незалежних частин системи – клієнтської, серверної та AI-сервісу – а також налаштування бази даних засобами Prisma ORM та підключення фізичного принтера Bambu Lab A1 за протоколами FTPS і MQTT, що потребує перебування обладнання в одній локальній мережі із сервером. Діаграма розгортання наочно відобразила розподіл програмних компонентів між фізичними вузлами та використовувані мережеві протоколи й порти.

Структуру інтерфейсу користувача розкрито через послідовний опис сторінок – від головної сторінки та каталогу до кастомного замовлення з інтерактивним тривимірним переглядачем та панелями AI-рекомендацій, а також через сторінки управління виробництвом і адміністрування, доступні відповідно до рольової моделі доступу. Діаграма навігації продемонструвала, що розширення функціональних можливостей для кожної наступної ролі відбувається додаванням нових розділів меню без обмеження доступу до базових сторінок, що забезпечує послідовний досвід користувача незалежно від його ролі.

Тестування трьох ключових алгоритмів системи – опрацювання кастомного замовлення з AI-аналізом та слайсингом, запуску друку з моніторингом принтера через MQTT та FTPS, а також автоматичного списання філаменту зі складу – підтвердило коректність їхньої роботи на типових вхідних даних, а перевірка межових випадків (недоступність слайсера, втрата зв'язку з принтером, скасування замовлення після списання матеріалу) підтвердила відмовостійкість реалізованих рішень. Окремо отримане емпіричним шляхом значення затримки перед командою запуску друку ілюструє практичний характер проведеного тестування з реальним обладнанням.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				76
Змн.	Арк.	№ докум.	Підпис	Дата		

Отримані результати свідчать про те, що розроблений програмний комплекс реалізує заявлені у вступі завдання роботи в обсязі, достатньому для функціонування прототипу онлайн-сервісу 3D-друку з інтеграцією AI асистента, та створює основу для подальшого розширення функціональності, окресленого як перспективний напрямок розвитку системи.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи спроектовано та реалізовано онлайн-сервіс 3D-друку з інтеграцією AI асистента, що автоматизує повний цикл роботи з кастомними замовленнями – від завантаження тривимірної моделі клієнтом до фізичного виконання друку на обладнанні та оновлення складського обліку матеріалів.

У першому розділі проведено аналіз предметної області та існуючих рішень на українському ринку послуг 3D-друку, який показав, що наявні сервіси здебільшого виконують функцію інформаційних візитівок із низьким рівнем автоматизації, а саме відсутній онлайн-перегляд моделей, автоматичний розрахунок вартості та інтелектуальна перевірка файлів перед друком. На основі виявлених недоліків обґрунтовано вибір трирівневої клієнт-серверної архітектури з виокремленим AI-сервісом, а також технологічного стеку – Node.js, React, TypeScript, PostgreSQL з Prisma ORM та обладнання Bambu Lab A1 з підтримкою протоколів FTPS і MQTT.

У другому розділі здійснено проектування системи: визначено рольову модель доступу та побудовано діаграму варіантів використання, сформульовано функціональні та нефункціональні вимоги, спроектовано об'єктно-орієнтовану структуру та фізичну схему бази даних із сімнадцяти таблиць. Ключовим архітектурним рішенням стало розділення замовлення на рівень групи та рівень окремої позиції, що дозволило коректно обробляти різнорідний кошик із каталожних і кастомних виробів у межах однієї покупки. Формалізовано та реалізовано три основні алгоритми роботи системи: опрацювання кастомного замовлення з геометричним аналізом моделі засобами нейронної мережі та слайсингом через PrusaSlicer CLI, запуск друку й моніторинг принтера через протоколи MQTT та FTPS, а також автоматичне списання філаменту зі складу за результатами завершеного друку.

У третьому розділі описано порядок розгортання трьох незалежних частин системи, налаштування підключення до фізичного обладнання та структуру інтерфейсу користувача, що динамічно адаптується залежно від ролі. Проведене

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				78
Змн.	Арк.	№ докум.	Підпис	Дата		

тестування на конкретних вхідних даних підтвердило коректність роботи реалізованих алгоритмів для типових та межових випадків, зокрема відмовостійкість системи при недоступності зовнішнього слайсера та втраті зв'язку з принтером, а емпірично визначені параметри взаємодії з обладнанням (коригувальний коефіцієнт ваги, затримка перед командою друку) підтвердили практичну придатність реалізованих рішень для роботи з реальним обладнанням.

Отримані результати свідчать про досягнення мети роботи – спроектовано та програмно реалізовано модель автоматизованої системи керування онлайн-сервісом 3D-друку з інтегрованим AI-модулем технічного аудиту моделей та системою прямого моніторингу обладнання в реальному часі. Усі завдання, поставлені у вступі, виконано: проаналізовано предметну область, сформовано та систематизовано вимоги користувачів різних ролей, розроблено інтелектуальний модуль аналізу геометричних параметрів моделей та динамічного розрахунку вартості, спроектовано об'єктно-орієнтовану структуру бази даних для гібридних замовлень, реалізовано алгоритми хмарної підготовки моделей та інтеграції з обладнанням Bambu Lab через протоколи MQTT, створено адаптивний інтерфейс з тривимірною візуалізацією на основі Three.js, а також виконано програмну реалізацію та тестування ключових функціональних модулів системи.

Практична цінність отриманих результатів полягає у створенні готового до використання прототипу, придатного для впровадження в діяльність невеликих студій 3D-друку, де поєднання автоматизованого технічного аудиту моделей та прямого керування обладнанням дозволяє зменшити навантаження на персонал і скоротити час обробки замовлень порівняно з традиційною схемою ручної комунікації через месенджери.

Подальший розвиток системи може здійснюватися за кількома напрямками, окресленими в межах архітектурних рішень роботи: інтеграція платіжного шлюзу для автоматизації прийому оплат, перехід до подійно-орієнтованої взаємодії між модулями складського обліку, сповіщень та фінансових операцій на основі вже закладеного логічного розмежування цих модулів на рівні бази

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		79

даних, розширення переліку підтримуваного обладнання за межі однієї моделі принтера, а також впровадження каталогу з більшою кількістю позицій та локалізації інтерфейсу для розширення цільової аудиторії сервісу.

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		80

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Школьна А.Л., Савіцький Р.С. Переваги та недоліки прямого підключення 3D-принтера до вебсервісу. Тези XIX Всеукраїнської науково-практичної web конференції аспірантів, студентів та молодих учених «Комп'ютерні інтелектуальні системи та мережі», 25-27 березня 2026 року. Кривий Ріг : «Криворізький національний університет», 2026. С.292.
2. Школьна А. Л., Савіцький Р.С., Толстой І.А. Використання ШІ-асистента для автоматизації перевірки та оптимізації 3D-моделей у вебсистемах. Тези Всеукраїнської науково-практичної конференції «Тенденції розвитку технологій в автоматизації, електроенергетиці, приладобудуванні та робототехніці», Житомир: "Житомирська політехніка", 13-15 травня 2026 р. [подано до друку].
3. Easy3DPrint – послуги 3D-друку та 3D-сканування [Електронний ресурс]. – URL: <https://easy3dprint.com.ua/> (дата звернення: 03.03.2026).
4. 3DTECHGROUP – промислове обладнання та послуги 3D-друку [Електронний ресурс]. – URL: <https://www.3dtech.com.my/uk> (дата звернення: 03.03.2026).
5. 3DForYou – сервіс доступного 3D-друку та сувенірної продукції [Електронний ресурс]. – URL: <https://3d4u.com.ua> (дата звернення: 04.03.2026).
6. Zustand – документація бібліотеки керування станом для React [Електронний ресурс]. – URL: <https://zustand.docs.pmnd.rs/> (дата звернення: 10.03.2026).
7. REST API Tutorial – опис архітектурного стилю Representational State Transfer [Електронний ресурс]. – URL: <https://restfulapi.net/> (дата звернення: 10.03.2026).
8. JSON Web Token (JWT) – офіційний інформаційний ресурс стандарту [Електронний ресурс]. – URL: <https://jwt.io/> (дата звернення: 12.03.2026).
9. MQTT – офіційна сторінка специфікації протоколу передачі повідомлень

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				81
Змн.	Арк.	№ докум.	Підпис	Дата		

- [Електронний ресурс]. – URL: <https://mqtt.org/> (дата звернення: 14.03.2026).
10. Node.js – завантаження, документація та посібники [Електронний ресурс]. – URL: <https://nodejs.org/uk> (дата звернення: 25.03.2026).
 11. Express.js – офіційна документація фреймворку для Node.js [Електронний ресурс]. – URL: <https://expressjs.com/> (дата звернення: 25.03.2026).
 12. React – офіційна документація бібліотеки для розробки користувацьких інтерфейсів [Електронний ресурс]. – URL: <https://react.dev/> (дата звернення: 27.03.2026).
 13. TypeScript: Документація для розробників [Електронний ресурс]. – URL: <https://www.typescriptlang.org/docs/> (дата звернення: 28.03.2026).
 14. Axios – Promise based HTTP client, Documentation [Електронний ресурс]. – URL: <https://axios-http.com/docs/intro> (дата звернення: 02.04.2026).
 15. PostgreSQL: Документація реляційної бази даних для систем замовлень [Електронний ресурс]. – URL: <https://www.postgresql.org/docs/> (дата звернення: 05.04.2026).
 16. Prisma ORM Documentation [Електронний ресурс]. – URL: <https://www.prisma.io/docs> (дата звернення: 08.04.2026).
 17. RFC 4217: Securing FTP with TLS, Internet Engineering Task Force [Електронний ресурс]. – URL: <https://datatracker.ietf.org/doc/html/rfc4217> (дата звернення: 12.04.2026).
 18. FastAPI Documentation – фреймворк для розробки REST API на Python [Електронний ресурс]. – URL: <https://fastapi.tiangolo.com/> (дата звернення: 14.04.2026).
 19. PyTorch Documentation – бібліотека для розробки та навчання нейронних мереж [Електронний ресурс]. – URL: <https://pytorch.org/docs/> (дата звернення: 18.04.2026).
 20. Trimesh Documentation – бібліотека для обробки тривимірної геометрії [Електронний ресурс]. – URL: <https://trimesh.org/> (дата звернення: 20.04.2026).

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				82
Змн.	Арк.	№ докум.	Підпис	Дата		

21. PrusaSlicer – офіційний репозиторій програмного забезпечення для підготовки моделей до 3D-друку [Електронний ресурс]. – URL: <https://github.com/prusa3d/PrusaSlicer> (дата звернення: 21.04.2026).
22. Three.js: Посібник з візуалізації 3D-моделей у браузері [Електронний ресурс]. – URL: <https://threejs.org/docs/> (дата звернення: 23.04.2026).
23. React Three Fiber Documentation – бібліотека для інтеграції Three.js у React [Електронний ресурс]. – URL: <https://r3f.docs.pmnd.rs/> (дата звернення: 25.04.2026).
24. Cloudinary Documentation – хмарний сервіс зберігання та обробки медіафайлів [Електронний ресурс]. – URL: <https://cloudinary.com/documentation> (дата звернення: 29.04.2026).
25. UML 2.0 Specification, Object Management Group [Електронний ресурс]. – URL: <https://www.omg.org/spec/UML/2.0/> (дата звернення: 05.05.2026).
26. StarUML Documentation: User Guide [Електронний ресурс]. – URL: <https://docs.staruml.io/> (дата звернення: 05.05.2026).
27. PlantUML [Електронний ресурс]. – URL: <https://editor.plantuml.com> (дата звернення: 06.05.2026).
28. UML Class Diagram Tutorial [Електронний ресурс]. – URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/> (дата звернення: 06.05.2026).
29. UML Activity Diagram Tutorial [Електронний ресурс]. – URL: <https://www.lucidchart.com/pages/uml-activity-diagram> (дата звернення: 07.05.2026).
30. Zhang Q., Chen S., Panozzo D. Thingi10K: A Dataset of 10,000 3D-Printing Models [Електронний ресурс]. – URL: <https://ten-thousand-models.appspot.com/> (дата звернення: 10.05.2026).
31. Bambu Lab Wiki – офіційна документація та технічна підтримка обладнання [Електронний ресурс]. – URL: <https://wiki.bambulab.com/> (дата звернення: 19.05.2026).
32. Vite – Next Generation Frontend Tooling, документація збірника проєктів

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				83
Змн.	Арк.	№ докум.	Підпис	Дата		

[Електронний ресурс]. – URL: <https://vite.dev/> (дата звернення: 22.05.2026).

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

ДОДАТКИ

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

Програмний код сервісу взаємодії з 3D-принтером

Лістинг А.1 містить програмний код класу BambuService (файл backend/src/services/bambu.service.ts), що реалізує передачу підготовленого файлу друку на принтер за протоколом FTPS та публікацію команди запуску друку за протоколом MQTT.

```
import * as mqtt from 'mqtt'
import * as ftp from 'basic-ftp'
import fs from 'fs'

interface PrinterConfig {
  ip: string
  accessCode: string
  serial: string
}

export class BambuService {
  static async uploadAndPrint(config: PrinterConfig, localFilePath: string,
  fileName: string): Promise<void> {
    const client = new ftp.Client()
    try {
      console.log(`[1/3] Підключення до принтера ${config.ip} (FTPS)...`)
      await client.access({
        host: config.ip, port: 990, user: "bblp", password: config.accessCode,
        secure: "implicit", secureOptions: { rejectUnauthorized: false }
      })
      if (!fs.existsSync(localFilePath)) {
        throw new Error(`Файл ${localFilePath} не знайдено на сервері!`)
      }
      console.log(`[2/3] Завантаження ${fileName} на SD-карту...`)
      await client.uploadFrom(localFilePath, fileName)
      console.log("Завантаження завершено успішно.")
    }
  }
}
```

```

client.close()
console.log("FTP з'єднання закрито.")
console.log(`Очікування 20 секунд для стабілізації системи...`)
await new Promise((resolve) => setTimeout(resolve, 20000))
await this.sendPrintCommand(config, fileName)
} catch (err) {
  const error = err as Error
  console.error("Помилка роботи з принтером:", error.message)
  client.close()
  throw new Error(`Не вдалося запустити друк: ${error.message}`)
}
}

private static sendPrintCommand(config: PrinterConfig, fileName: string):
Promise<void> {
  return new Promise((resolve, reject) => {
    const options: mqtt.IClientOptions = {
      port: 8883, host: config.ip, protocol: 'mqtts', username: 'bblp',
      password: config.accessCode, rejectUnauthorized: false
    }
    console.log("[3/3] Надсилання MQTT команди на друк...")
    const mqttClient = mqtt.connect(options)
    mqttClient.on('connect', () => {
      const payload = {
        print: {
          command: "project_file", url: `file:///sdcard/${fileName}`,
          param: "Metadata/plate_1.gcode", bed_leveling: true, flow_cali: true,
          vibration_cali: true, timelapse: false,
          use_measured_shaper: true
        }
      }
    })
  })
}

```

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		87

```

const topic = `device/${config.serial}/request`
mqttClient.publish(topic, JSON.stringify(payload), { qos: 1 }, (err) => {
  if (err) {
    console.error("Помилка відправки MQTT:", err)
    mqttClient.end()
    reject(err)
  } else {
    console.log("Команду відправлено! Перевір екран принтера.")
    mqttClient.end()
    resolve()
  }
})
})
mqttClient.on('error', (err: Error) => {
  console.error("Помилка MQTT з'єднання:", err.message)
  reject(err)
})
})
}
}

```

Лістинг А.2 містить програмний код класу PrinterWatcherService (файл backend/src/services/watcher.service.ts), що реалізує фоновий моніторинг стану принтерів засобами підписки на топіки телеметрії за протоколом MQTT.

```

import * as mqtt from 'mqtt'
import prisma from '../utils/prisma'
interface BambuReportPayload {
  print?: {
    gcode_state?: string
    nozzle_temper?: number
    bed_temper?: number

```

		Шкільна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		88

```

    }
}

interface ApiMqttError {
    message: string
}

export class PrinterWatcherService {
    private static clients: Record<string, mqtt.MqttClient> = {}

    static async startMonitoring(): Promise<void> {
        console.log('Запуск служби онлайн-моніторингу принтерів...')

        try {
            const printers = await prisma.printer.findMany()

            for (const printer of printers) {
                if (!printer.ipAddress || !printer.accessCode || !printer.serialNumber) {
                    console.log(`Принтер ${printer.name} пропущено: відсутні мережеві налаштування.`)
                    continue
                }

                if (this.clients[printer.id]) {
                    this.clients[printer.id].end()
                }

                const options: mqtt.IClientOptions = {
                    port: 8883,
                    host: printer.ipAddress,

```

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		89

```

    protocol: 'mqtt',
    username: 'bblp',
    password: printer.accessCode,
    rejectUnauthorized: false,
    reconnectPeriod: 10000,
  }
  const client = mqtt.connect(options)
  this.clients[printer.id] = client
  client.on('connect', async () => {
    console.log(`Мережевий зв'язок з принтером [${printer.name}]
встановлено успішно!`)
    client.subscribe(`device/${printer.serialNumber}/report`, { qos: 1 })
    await prisma.printer.update({
      where: { id: printer.id },
      data: { status: 'IDLE' }
    })
  })
  client.on('message', async (topic, message) => {
    try {
      const payload = JSON.parse(message.toString()) as BambuReportPayload
      if (!payload.print) return
      const { gcode_state, nozzle_temper, bed_temper } = payload.print
      let dbStatus: 'IDLE' | 'PRINTING' | 'ERROR' | 'OFFLINE' = 'IDLE'
      if (gcode_state === 'RUNNING' || gcode_state === 'PREPARE' ||
gcode_state === 'PAUSE') {
        dbStatus = 'PRINTING'
      } else if (gcode_state === 'FAILED') {
        dbStatus = 'ERROR'
      }
      await prisma.printer.update({

```

		Школьна А.Л.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Савіцький Р.С.				90
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        where: { id: printer.id },
        data: {
            status: dbStatus,
            nozzleTemp: nozzle_temper ?? null,
            bedTemp: bed_temper ?? null
        }
    })
} catch (parseError) {
    console.error(`Помилка обробки MQTT пакету від ${printer.name}:`,
parseError)
}
})
client.on('close', async () => {
    await prisma.printer.update({
        where: { id: printer.id },
        data: { status: 'OFFLINE', nozzleTemp: null, bedTemp: null }
    })
})
client.on('error', (err) => {
    const mqttError = err as ApiMqttError
    console.error(`MQTT Помилка на принтері ${printer.name}:`,
mqttError.message)
})
}
} catch (dbError) {
    console.error('Не вдалося зчитати принтери з бази даних для
моніторингу:', dbError)
}
}}

```