КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Автоматизації і інформаційних технологій
(факультет)

Інформаційних технологій проєктування та прикладної математики
(кафедра)

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

на тему: «Інформаційна система параметричного 3D-моделювання індивідуальних будівельних виробів»

Макарчук Даніїл Андрійович
(прізвище, ім’я та по батькові магістра повністю)

Київ 2025 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Автоматизації і інформаційних технологій
(факультет)

Інформаційних технологій проєктування та прикладної математики
(кафедра)

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТППМ
д.т.н., професор Бородавка Є. В.

«____» _____ 2025 року

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

на тему: «Інформаційна система параметричного 3D-моделювання індивідуальних будівельних виробів»

Виконав: студент 2-го курсу, групи ІСТм-24 .

Спеціальності: 126 «Інформаційні системи та технології» .
(шифр і назва напряму підготовки, спеціальності)

Магістрант Макарчук Д. А. .
(прізвище та ініціали)

Керівник д.т.н., проф. Бородавка Є. В. .
(прізвище та ініціали)

Рецензент .
(прізвище та ініціали)

Київ 2025 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: автоматизації і інформаційних технологій .
Кафедра: інформаційних технологій проєктування та ПМ .
Освітній рівень: «магістр за ОПП» .
Спеціальність: 126 «Інформаційні системи та технології» .

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТППМ
д.т.н., професор Бородавка Є. В.

«____» _____ 2025 року

З А В Д А Н Н Я
ДО ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

Макарчук Даніїл Андрійович .

1. Тема роботи: Інформаційна система параметричного 3D-моделювання індивідуальних будівельних виробів .
затверджена наказом ректора КНУБА № 1619/23/25 від «29» 09 2025 р.
2. Керівник роботи: Бородавка Євгеній Володимирович, завідувач кафедри інформаційних технологій проєктування та прикладної математики, доктор технічних наук, професор.
3. Строк подання студентом роботи до захисту: 18.12.2025 .
4. Зміст пояснювальної записки за розділами:
Р.1. Аналітичний огляд проблематики та технологій параметричного 3D-моделювання у вебсередовищі .
Р.2. Проєктування архітектури системи .
Р.3. Сучасні технології розробки графічних систем .
Р.4. Програмна реалізація системи .
Р.5. Оцінка ефективності системи та тестовий приклад .
5. Інформаційні слайди:
С.1. Актуальність проєкту . С.10. Загальна схема реалізації архітектури ПЗ.
С.2. Аналітичний огляд . С.11. Діаграма використання бібліотек .

С.3. Аналіз існуючих рішень . С.12. Структура вебзастосунку системи .
С.4. Постановка задачі . С.13. Діаграма класів .
С.5. Діаграма дерева функцій . С.14. Діаграма потоків даних .
С.6. Модель бізнес-процесу IDEF0 . С.15. Алгоритмічне забезпечення .
С.7. Архітектурна модель системи . С.16. Прототип інтерфейсу користувача .
С.8. Схема бази даних . С.17. Контрольний приклад .
С.9. Сучасні технології розробки .

6. Календарний план виконання кваліфікаційної випускної роботи

Види робіт та їх зміст	Дата виконання
Р.1. Аналітичний огляд проблематики та технологій параметричного 3D-моделювання у вебсередовищі	Жовтень 2025 р.
Р.2. Проєктування архітектури системи	Жовтень 2025 р.
Р.3. Сучасні технології розробки графічних систем	Жовтень 2025 р.
Р.4. Програмна реалізація системи	Листопад 2025 р.
Р.5. Оцінка ефективності системи та тестовий приклад	Листопад 2025 р.
Остаточне оформлення роботи	Грудень 2025 р.
Направлення роботи на рецензування, перевірку на плагіат	Грудень 2025 р.
Попередній захист роботи на кафедрі	Грудень 2025 р.

7. Консультанти розділів кваліфікаційної випускної роботи

Розділ	Прізвище, ініціали та посада консультанта	Перевірив	
		дата	підпис
Розділ 1.	Рашківський В. П., завідувач кафедри будівельних машин, кандидат технічних наук, доцент		

8. Дата видачі завдання: 30 вересня 2025 року .

Керівник _____ Бородавка Є. В.
(підпис) (прізвище та ініціали)

Магістрант _____ Макарчук Д. А.
(підпис) (прізвище та ініціали)

РЕЗЮМЕ

Київський національний університет будівництва і архітектури

Макарчук Даніїл Андрійович

факультет автоматизації і інформаційних технологій,

група ІСТм-24

Тема кваліфікаційної випускної роботи: «Інформаційна система параметричного 3D-моделювання індивідуальних будівельних виробів»

освітній рівень: магістр,

спеціальність: 126 «Інформаційні системи та технології»,

Науковий керівник: Бородавка Євгеній Володимирович, завідувач кафедри інформаційних технологій проєктування та прикладної математики, доктор технічних наук, професор

Обсяг роботи. Кваліфікаційна випускна робота магістра складається: розділів 5, стор. 120, таблиць 9, рис. 49, завдання, анотації, вступу, висновків, списку використаних джерел.

Актуальність теми. Стрімкий розвиток вебтехнологій та зростання попиту на індивідуальні будівельні вироби зумовлюють необхідність створення інформаційних систем нового покоління. Такі системи мають забезпечувати інтерактивне параметричне 3D-моделювання виробів у браузері, поєднуючи інструменти комп'ютерної графіки, вебпрограмування та засоби автоматизації процесів замовлення. Традиційні підходи до проєктування та узгодження креслень часто є складними та схильними до помилок, що підкреслює важливість розробки онлайн-конструкторів, здатних формувати моделі відповідно до параметрів замовника та інтегрувати їх у виробничий процес.

У вступі обґрунтовано необхідність створення веборієнтованої інформаційної системи параметричного 3D-моделювання індивідуальних стільниць, визначено мету, завдання та методи дослідження, а також практичну цінність отриманих результатів для автоматизації виробництва.

У першому розділі «Аналітичний огляд проблематики та технологій параметричного 3D-моделювання у вебсередовищі» проаналізовано сучасні рішення у сфері 3D-конфігурації виробів та визначено їхні обмеження. Виявлено, що більшість сучасних систем не забезпечують достатнього рівня інтерактивності або не мають повноцінної інтеграції з виробничими процесами. Обґрунтовано необхідність створення спеціалізованого онлайн-конструктора.

У другому розділі «Проєктування архітектури системи» розроблено структурну схему інформаційної системи, що включає клієнтську та серверну складові. Спроєктовано логічну структуру бази даних для зберігання параметрів замовлень та моделей, визначено специфікації API для взаємодії компонентів. Запропонована архітектура забезпечує масштабованість, гнучкість і можливість хмарного розгортання.

У третьому розділі «Сучасні технології розробки графічних систем» розглянуто технології, що лежать в основі реалізації 3D-графіки у браузері. Проаналізовано можливості WebGL, Three.js та допоміжних інструментів для реалізації параметричної геометрії.

У четвертому розділі «Програмна реалізація системи» представлено процес розробки клієнтської частини системи з використанням бібліотек React.js та Three.js. Описано реалізацію алгоритмів динамічної генерації геометрії стільниць, виконання булевих операцій для створення вирізів. Реалізовано функціонал експорту параметричних моделей для подальшого використання у виробництві.

У п'ятому розділі «Оцінка ефективності системи та тестовий приклад» на контрольних прикладах продемонстровано правильну роботу алгоритмів побудови та візуалізації. Запропонований вебконфігуратор значно прискорює процес формування індивідуальної стільниці порівняно з традиційними CAD-системами. Розроблена система може бути використана у виробничих і проєктних організаціях.

Ключові слова: параметричне моделювання, 3D-візуалізація, вебконфігуратор, інформаційна система, індивідуальні вироби.

Keywords: parametric modeling, 3D visualization, web configurator, information system, custom products.

Якість оформлення проєкту. Кваліфікаційна випускна робота магістра оформлена у відповідності до діючих нормативних документів та методичних вказівок до виконання дипломних робіт для студентів спеціальності 126 «Інформаційні системи та технології».

Загальний висновок стосовно роботи та присвоєння авторові освітньо-кваліфікаційного рівня «магістр». Робота виконана на високому рівні, студент продемонстрував високий рівень теоретичної підготовки та сформованих практичних навичок в області сучасних інформаційних технологій. Заслуговує оцінки « 98 ».

Науковий керівник _____ / д.т.н., професор Бородавка Є. В. ./
(підпис)

Посада, місце роботи: завідувач кафедри інформаційних технологій проєктування та прикладної математики, доктор технічних наук, професор, КНУБА, м. Київ, проспект Повітряних Сил, 31.

«____» _____ 2025 р.

АНОТАЦІЯ

Макарчук Д. А. «Інформаційна система параметричного 3D-моделювання індивідуальних будівельних виробів».

Кваліфікаційна випускна робота магістра за спеціальністю: 126 «Інформаційні системи та технології». – Київський національний університет будівництва і архітектури. – Київ, 2025.

Кваліфікаційна робота магістра присвячена встановленню та обґрунтуванню підходів до розробки інформаційної системи параметричного 3D-моделювання стільниць. Було виконано головне завдання – створення онлайн-конструктора як зручного сервісу для оформлення замовлень з обробки індивідуальних стільниць з 3D-візуалізацією деталей та обробок.

Ключові слова: параметричне моделювання, 3D-візуалізація, вебконфігуратор, інформаційна система, індивідуальні вироби.

SUMMARY

Makarchuk D. A. “Information system for parametric 3D modeling of individual construction products”.

Qualification master’s degree by specialty: 126 “Information systems and technologies”. – Kyiv National University of Construction and Architecture. – Kyiv, 2025.

The master’s qualification work is devoted to establishing and substantiating approaches to the development of an information system for parametric 3D modeling of countertops. The main task was accomplished – the creation of an online designer as a convenient service for placing orders for the processing of individual countertops with 3D visualization of parts and finishes.

Keywords: parametric modeling, 3D visualization, web configurator, information system, custom products.

ЗМІСТ

ВСТУП

Сучасний розвиток інформаційних технологій безпосередньо впливає на всі сфери виробництва, зокрема на галузь будівництва та виготовлення індивідуальних виробів для інтер’єру. Особливого значення набуває питання автоматизації процесів проєктування та візуалізації елементів, які виготовляються за конкретними параметрами замовника. Одним із прикладів таких виробів є стільниці з натурального каменю, зокрема мармуру, що активно використовуються у житловому та громадському будівництві. Традиційні методи замовлення та виготовлення таких виробів ґрунтуються на ручному вимірюванні, кресленні та узгодженні ескізів між замовником і виробником, що є причиною затримок, неточностей та підвищення вартості.

З огляду на зростання вимог до якості та індивідуалізації продукції, особливо актуальною стає проблема створення веборієнтованих інформаційних систем, які поєднують можливості параметричного 3D-моделювання, інтерактивної візуалізації та управління замовленнями. Такий підхід дозволить скоротити терміни проєктування, зменшити ймовірність помилок та оптимізувати взаємодію між замовником і виробником.

В умовах сучасного ринку меблів та оздоблювальних матеріалів в нашій країні та за кордоном спостерігається зростання попиту на індивідуальні вироби, які відповідають специфічним вимогам клієнта щодо розмірів, форми та обробки. Використання параметричного підходу дозволяє формувати виріб на основі введених параметрів, змінювати його характеристики у реальному часі та отримувати тривимірну модель із максимальною відповідністю очікуванням замовника.

Виробники дедалі частіше стикаються з потребою у швидкому створенні комерційних пропозицій, що містять не лише текстову специфікацію, а й візуалізацію майбутнього виробу. Саме тому інформаційні системи нового

покоління, здатні інтегрувати 3D-візуалізацію з можливістю оформлення замовлень онлайн, набувають все більшого значення.

Актуальність дослідження зумовлена стрімким розвитком цифрових технологій у сфері будівництва, зростанням попиту на індивідуальні вироби та потребою у впровадженні автоматизованих систем для підвищення ефективності проєктування та виробництва.

Об’єктом дослідження є методи та інструменти реалізації веборієнтованої системи параметричного 3D-моделювання індивідуальних будівельних виробів.

Предметом дослідження є веборієнтована система параметричного 3D-моделювання стільниць.

Мета дипломного проєкту полягає у встановленні та обґрунтуванні підходів до розробки інформаційної системи параметричного 3D-моделювання стільниць.

Методами дослідження є системний аналіз, методи об’єктно-орієнтованого проєктування, математичне моделювання, алгоритмізація, методи комп’ютерного 3D-моделювання та візуалізації, методи вебпрограмування, тестування програмного забезпечення.

Розроблена інформаційна система може бути використана у проєктних та виробничих організаціях для автоматизації створення індивідуальних будівельних елементів. Вона підвищує зручність та наочність процесу замовлення для кінцевих користувачів завдяки 3D-візуалізації та можливості самостійного конфігурування, зменшує помилки під час формування замовлень та скорочує час на їх обробку. Результати роботи можуть бути використані для подальшого розвитку систем онлайн-конфігураторів для інших типів індивідуальних виробів.

1 АНАЛІТИЧНИЙ ОГЛЯД ПРОБЛЕМАТИКИ ТА ТЕХНОЛОГІЙ ПАРАМЕТРИЧНОГО 3D-МОДЕЛЮВАННЯ У ВЕБСЕРЕДОВИЩІ

1.1 Аналітичний огляд

Ринок індивідуальних стільниць із натурального каменю демонструє стійке зростання. За даними Freedonia, обсяг ринку США у 2023 році досяг $7,4 млрд із прогнозом подальшого приросту в сегменті натурального каменю [1]. Разом із тим процес проєктування та замовлення часто залишається фрагментованим та ручним, що є причиною затримок і помилок у виготовленні. Ефективне автоматизоване рішення дозволить прискорити обробку замовлень, скоротити операційні витрати та підвищити задоволеність клієнтів.

1.2 Аналіз готових рішень параметричного моделювання

Параметричне 3D-моделювання – це підхід до створення тривимірних об’єктів, у якому геометрія моделі визначається не вручну, а за допомогою параметрів – змінних, що описують розміри, форму, пропорції, матеріали чи інші властивості об’єкта.

Сьогодні ринок пропонує ряд програмних продуктів та вебсервісів для параметричного моделювання виробів, зокрема меблів, елементів інтер’єру та будівельних конструкцій. Основне завдання таких систем полягає у забезпеченні можливості користувачу швидко та зручно конфігурувати виріб під власні потреби, враховуючи розміри, матеріали, форму та інші параметри.

Основними розділами сайту blum.com є «Вироби», «Сервіси», «Компанія» та «Контакти» (рисунок 1.1).

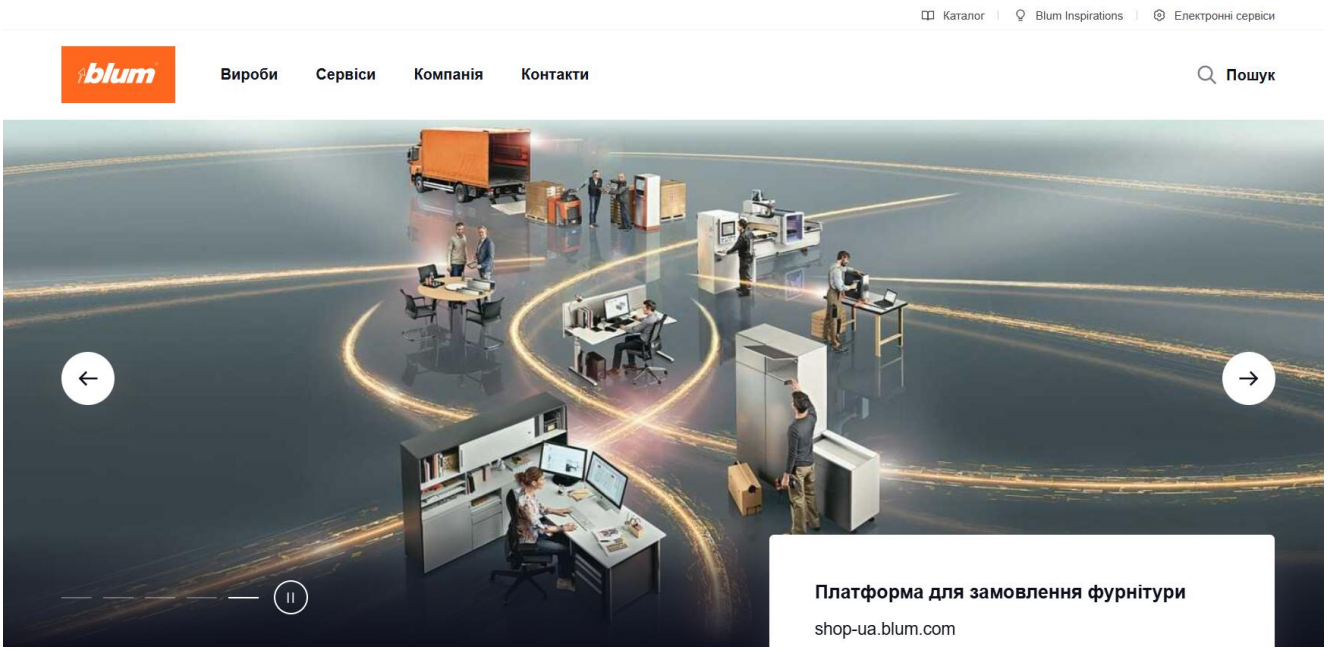


Рисунок 1.1 – Головна сторінки сайту blum.com

У розділі «Вироби» представлена фурнітура Blum, згрупована за категоріями (рисунок 1.2).

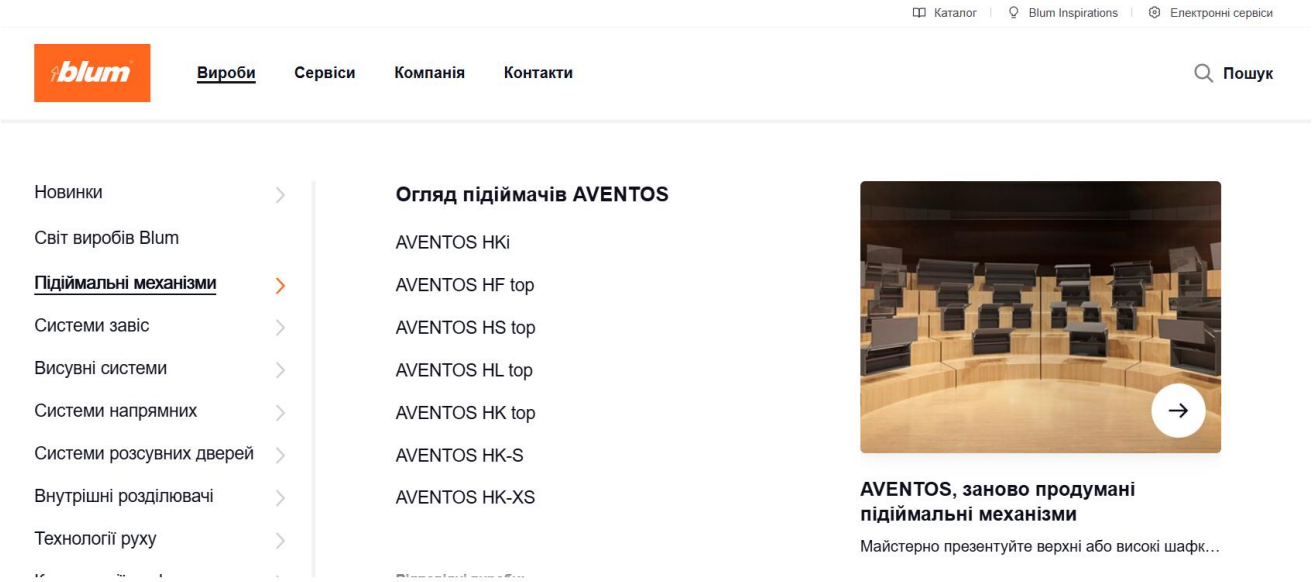


Рисунок 1.2 – Розділ «Вироби»

Розділ «Сервіси» пропонує комплексну підтримку для клієнтів і партнерів на всіх етапах, від проєктування та конструювання до монтажу й регулювання. Тут доступні платформа для замовлення фурнітури, конфігуратори корпусів та виробів, керування замовленнями (рисунок 1.3).

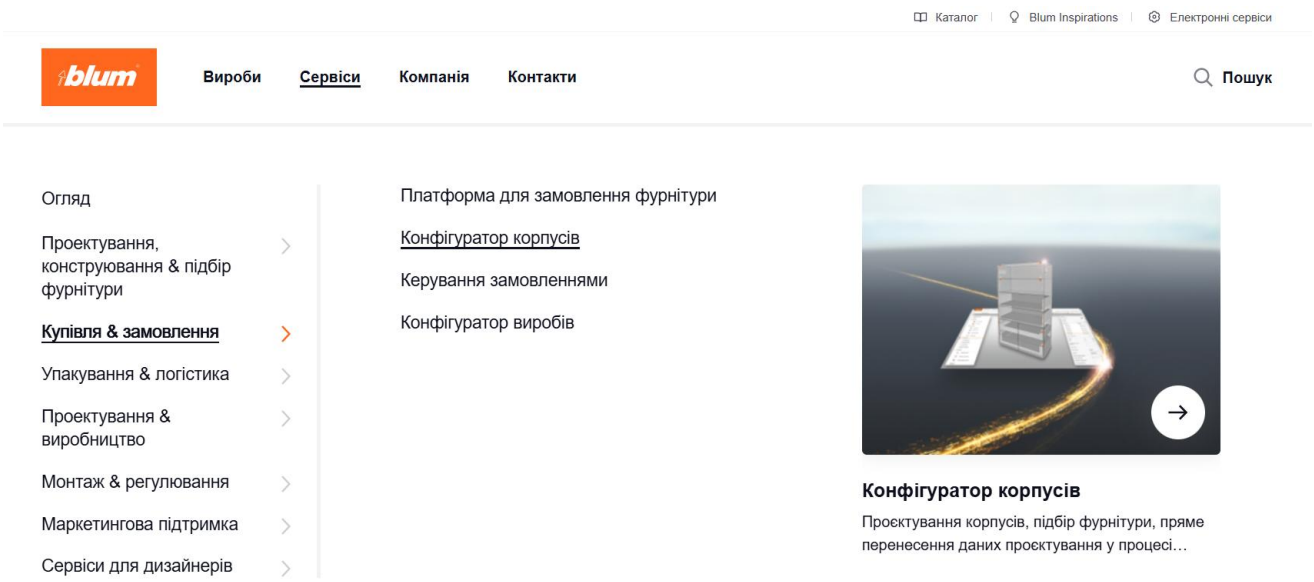


Рисунок 1.3 – Розділ «Сервіси»

Вебсайт надає доступ до великої медіатеки з маркетинговими матеріалами (зображення, брошури), технічною інформацією, інструкціями з монтажу та відео. У розділі «Компанія» міститься інформація про ТОВ «Блюм Україна», історію бренду, його принципи якості, дані про виробництво та відповідальне ставлення до довкілля.

Сайт має професійний, добре структурований вигляд та є надзвичайно інформативним. Навігація логічна та інтуїтивно зрозуміла, що дозволяє легко знаходити потрібну інформацію як професіоналам (дизайнерам, виробникам меблів), так і кінцевим споживачам. Вебсайт комплексно представляє діяльність компанії, охоплюючи не тільки каталог продукції, але й сервісну підтримку, навчальні матеріали та корпоративні цінності, що створює вигляд надійного та сучасного партнера.

Реєстрація на сайті проходить за досить стандартним для корпоративних сайтів шаблоном. Вимагається введення електронної пошти, імені, прізвища, назви компанії, адреси та контактних даних (рисунок 1.4).

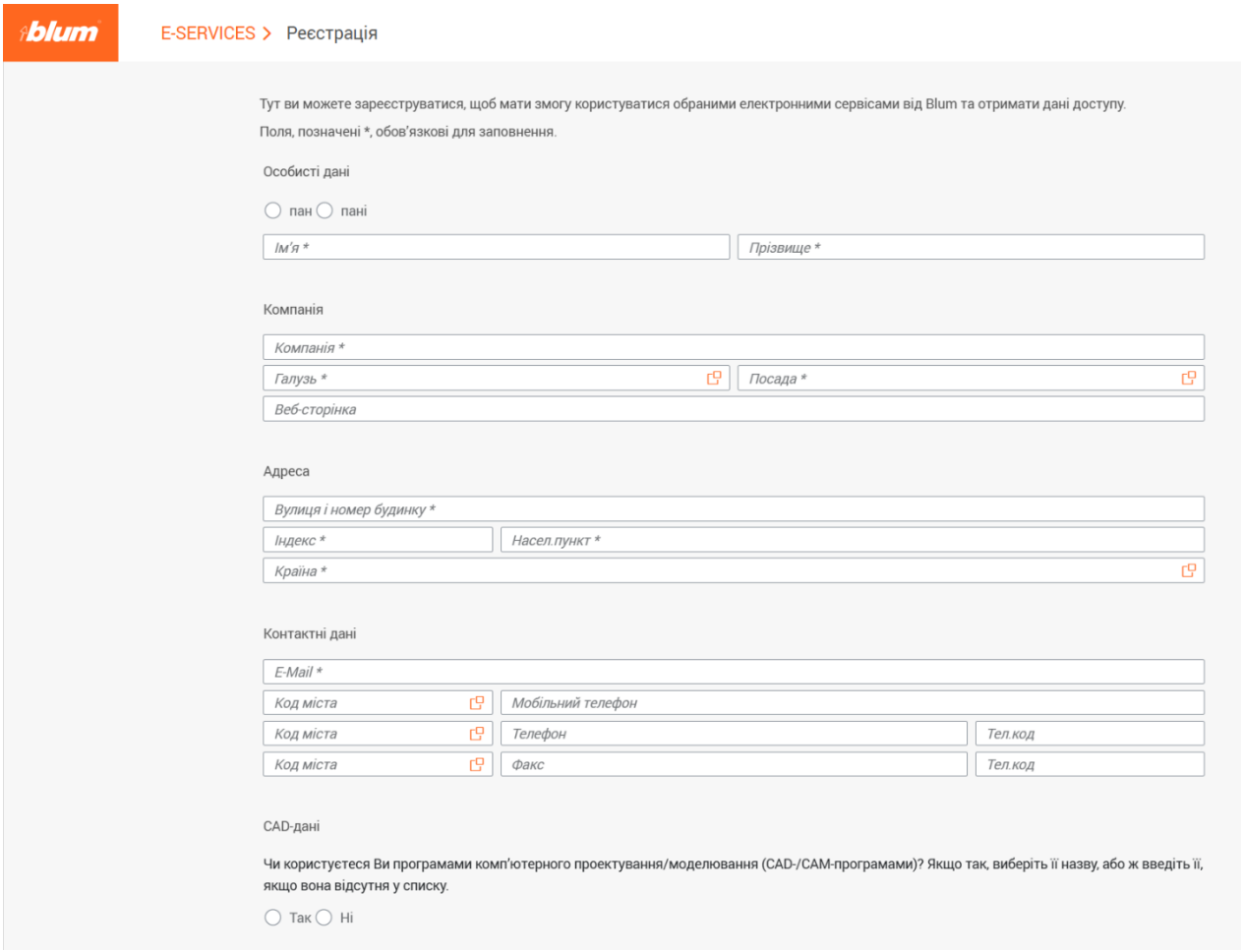


Рисунок 1.4 – Форма для реєстрації клієнта

Після заповнення форми обов'язковою є верифікація облікового запису через електронну пошту. Користувач отримує лист із посиланням, на яке потрібно натиснути для активації акаунту. Далі потрібно створити пароль для входу в систему.

Blum Cabinet Configurator – це професійний інструмент для створення конфігурацій меблів, що дозволяє проєктувати корпусні елементи з точними розмірами та фурнітурою (рисунок 1.5).

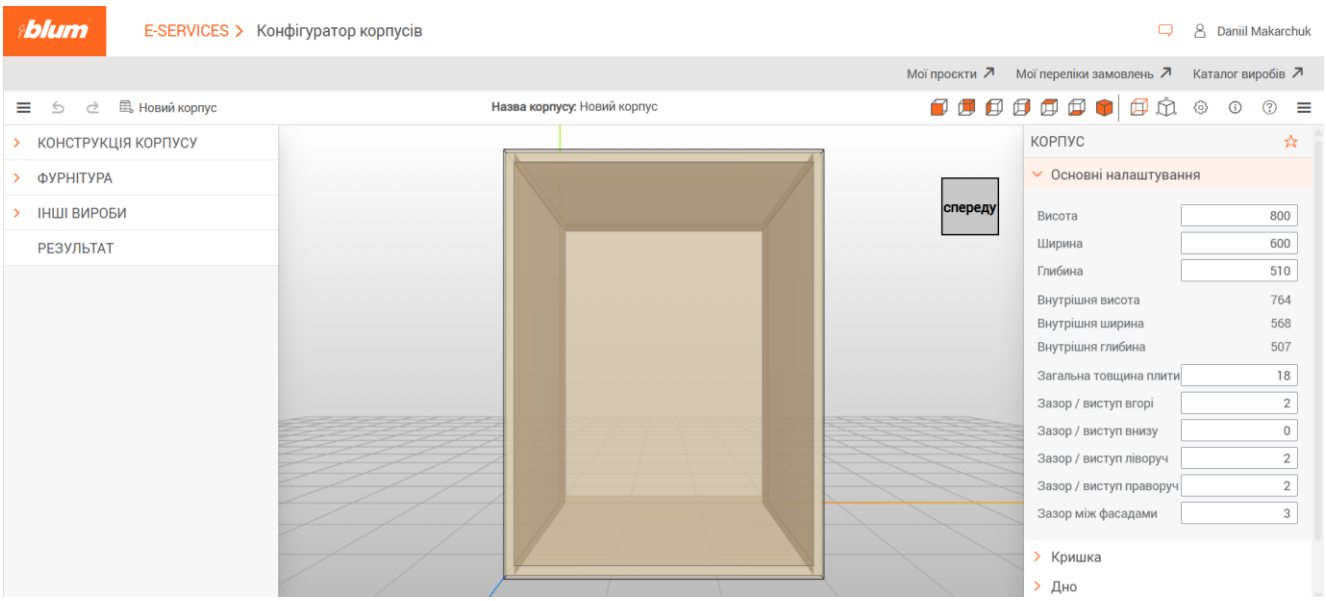


Рисунок 1.5 – Конфігуратор корпусів Blum

Система забезпечує візуалізацію виробу та створює специфікацію комплектуючих. Переваги для малих виробників без та із CAD-програмами:

- комплексне програмне забезпечення зі зрозумілим інтерфейсом гарантує 3D-візуалізацію корпусу та фурнітури;
- розміри корпусу, котрі часто використовуєте, зберігайте як стандартні значення;
- вносьте зміни у конструкцію корпусу після проєктування фурнітури;
- оберіть за допомогою конфігуратора виробів висувні системи, системи завіс і підіймальні механізми;
- експортуйте CAD-дані найпоширеніших форматів до вашої CAD-програми або використовуйте наші BXF-дані за наявності інтерфейсу;
- переліки замовлень можна одразу переносити до онлайн-магазину обраного дилера;
- використовуйте наші дані для розкрою деталей, технічні рисунки або креслення – у цифровому або аналоговому форматах для роздруку;
- перенесіть за допомогою BXF*-даних результати проєктування на MINIPRESS із EASYSTICK;

– зберігайте Ваші дані для проєктування та керуйте ними у «Моїх проєктах / Моїх переліках замовлення».

*BXF вміщує не тільки відомості про фурнітуру, але й інформацію про обробку дерев'яних деталей, наприклад, розкрій та позиції свердління [2].

Тож чим цікава ця інформаційна система:

– дає можливість змоделювати конструктивний елемент, зокрема меблевий;
– маємо 3D-модель – візуалізацію для замовника;
– одразу можна сформувати специфікацію виробів для виготовлення;
– можна експортувати модель для подальшого використання (проєктування, розрахунки, реклама тощо), тобто є розвиток життєвого циклу продукту проєктування.

TraceParts (traceparts.com) є онлайн-бібліотекою 3D-компонентів, що використовується інженерами та дизайнерами (рисунок 1.6). Підтримує завантаження моделей у різних CAD-форматах. Головний акцент зроблений на готових деталях та вузлах [3].

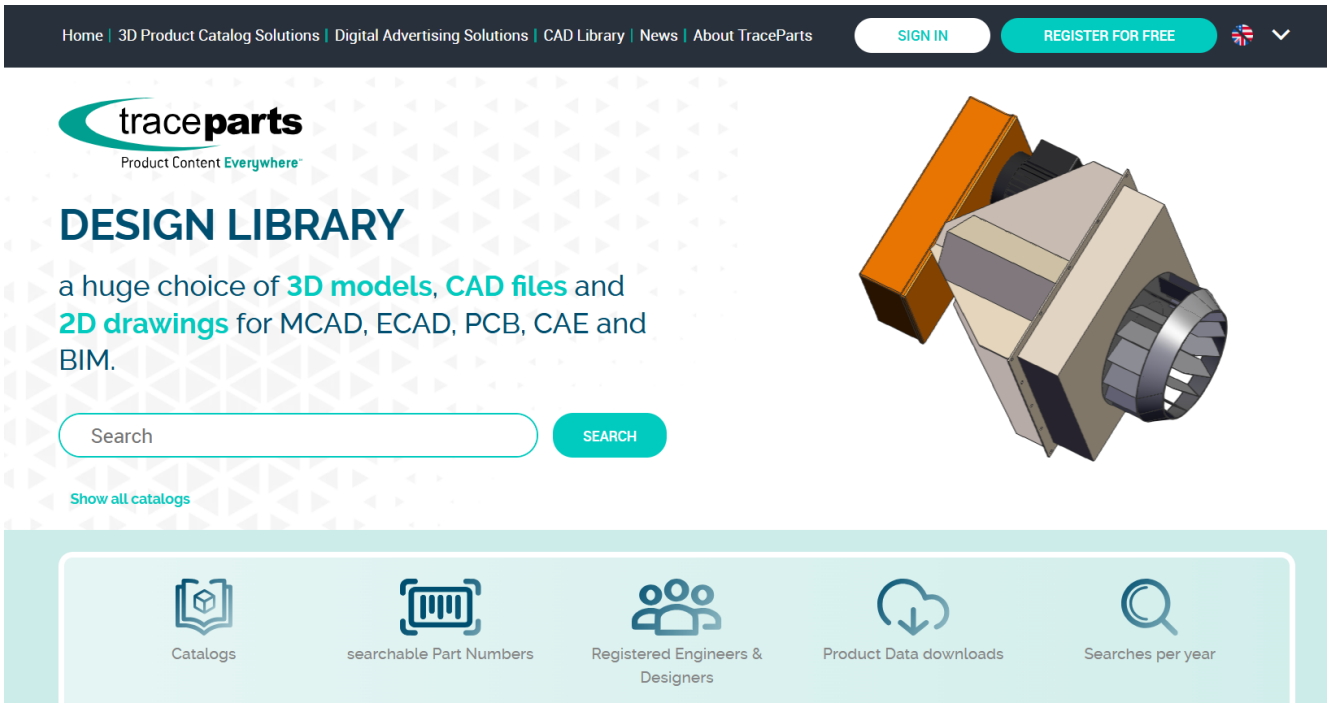


Рисунок 1.6 – TraceParts

Peikko Designer (peikkodesigner.com) – це інженерний конфігуратор для проєктування з'єднань і закладних деталей у будівництві (рисунок 1.7). Інтегрується з професійними BIM-системами, наприклад, Autodesk Revit та Tekla Structures [4].

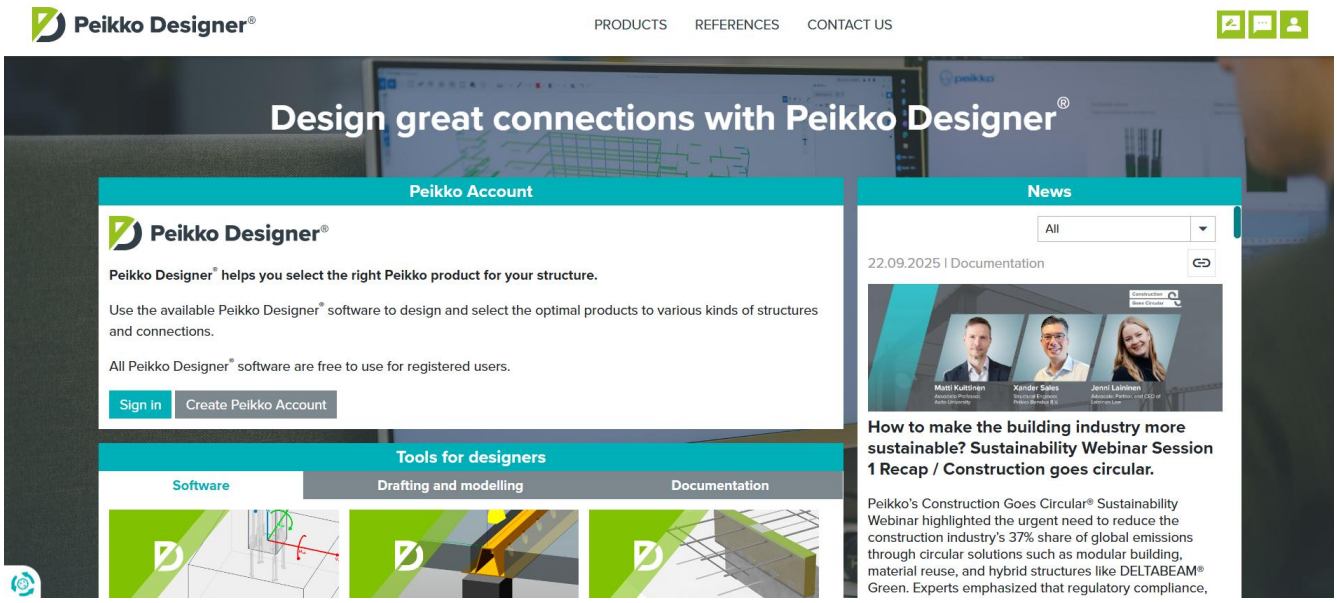


Рисунок 1.7 – Peikko Designer

KRONAS MASTER 3D (master.kronas.com.ua) є яскравим прикладом успішної реалізації вебсервісу для замовлення індивідуальних меблевих деталей в Україні (рисунок 1.8). Система надає можливості для проєктування розсувних систем, замовлення порізки плитних матеріалів, фрезерування, крайкування та інших послуг. Перевагами є зручний та інтуїтивно зрозумілий інтерфейс, орієнтований на кінцевого користувача, інтеграція з особистим кабінетом для управління замовленнями та оплатами, автоматичний розрахунок вартості [5].

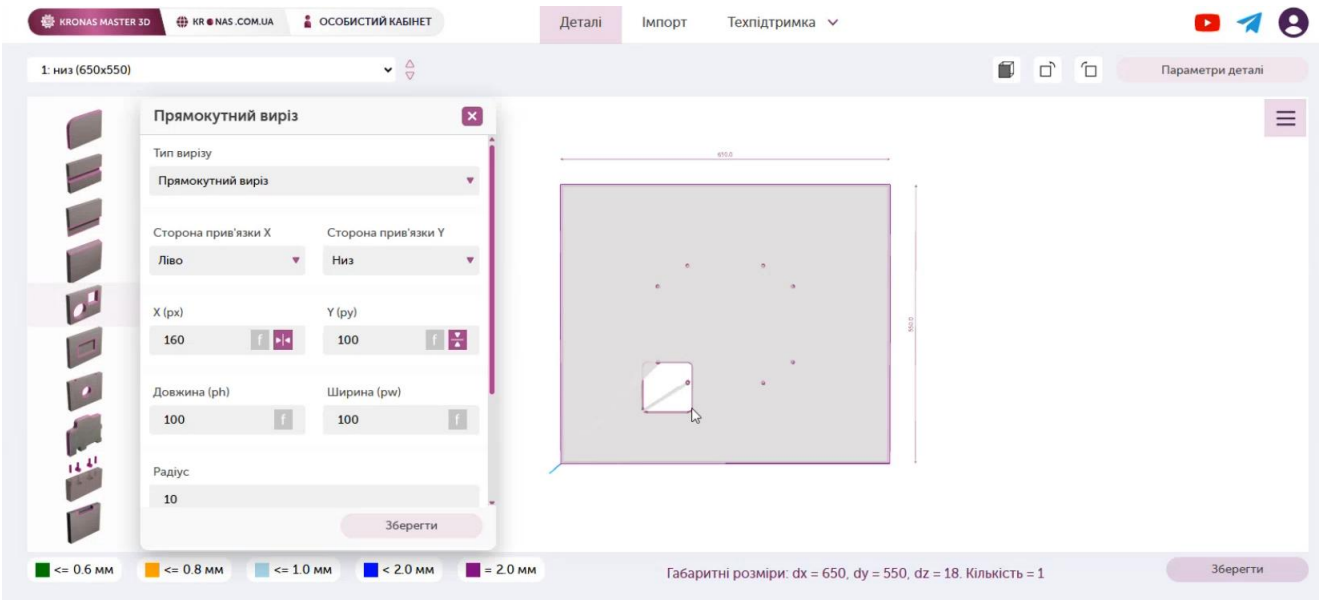


Рисунок 1.8 – KRONAS MASTER 3D

Недоліки існуючих рішень:

- більшість сервісів орієнтовані на меблі та стандартні вироби, а не на унікальні вироби з натурального каменю;
- обмежена інтеграція з системами онлайн-оплати та управління замовленнями;
- недостатня увага до точного моделювання складних поверхонь та обробок у реальному часі;
- відсутність можливості розробки індивідуальних виробів у вузькій спеціалізації (наприклад, стільниці сантехнічних меблевих конструкцій).

Існуючі рішення можна умовно поділити на три категорії:

- B2C (Business-to-Consumer) конфігуратори орієнтовані на кінцевого споживача, прості у використанні, з акцентом на візуалізацію та автоматичний розрахунок ціни (KRONAS MASTER 3D);
- B2B (Business-to-Business) інструменти призначені для професіоналів (інженерів, архітекторів), інтегровані в САПР та BIM-процеси, містять складну технічну інформацію (Peikko Designer);
- бібліотеки компонентів надають доступ до готових 3D-моделей стандартних виробів для використання в САПР (TraceParts).

Проведемо узагальнення ключових характеристик кожної системи та оцінимо їхню релевантність для цього проєкту (таблиця 1.1).

Таблиця 1.1 – Порівняння інформаційних систем

Критерій	KRONAS MASTER 3D	Blum Cabinet Configurator	Peikko Designer	TraceParts
Основне призначення	Вебсервіс для замовлення послуг (порізка, крайкування) та розрахунку вартості	Проєктування корпусних меблів з використанням конкретної фурнітури	Інженерний інструмент для розрахунку та інтеграції будівельних компонентів	Онлайн-бібліотека готових 3D-моделей стандартних виробів
Рівень 3D-візуалізації	Схематична 2D та спрощена 3D, акцент на кресленнях	Високодеталізована, технічно точна 3D-модель	Технічна 3D-модель, призначена для BIM-середовища	Деталізована технічна 3D-модель для перегляду перед завантаженням
Гнучкість параметризації	Вибір параметрів обмежений доступними послугами та матеріалами	В межах екосистеми. Глибока параметризація, що враховує всі особливості фурнітури Blum	Параметри визначаються технічними характеристиками та розрахунками навантажень	Зазвичай вибір з попередньо заданого списку розмірів чи конфігурацій
Інтеграція з САПР/BIM	Немає, це закрита система, що генерує дані для внутрішнього виробництва	Експорт у різноманітні CAD-формати (DWG, DXF, STEP та ін.)	Пряма інтеграція у вигляді плагінів для ключових BIM-систем (Revit, Tekla)	Можливість завантажити модель у десятках нейтральних та нативних CAD-форматів
Простота використання	Інтуїтивно зрозумілий інтерфейс для непідготовленого користувача	Вимагає розуміння меблевого проєктування та специфіки фурнітури	Призначена виключно для професіоналів з інженерною освітою	Зручний пошук та завантаження, як у будь-якому онлайн-каталозі
Придатність для цього проєкту	Як модель для B2C-інтерфейсу, особистого кабінету та логіки замовлення	Як приклад якісної 3D-візуалізації та експорту даних для професіоналів	Не прямий аналог, але корисна для розуміння принципів BIM-інтеграції	Це каталог, а не інтерактивний конфігуратор

1.3 Аналіз існуючих інформаційних систем для 3D-моделювання

За архітектурою системи поділяються на:

- desktop-додатки – це традиційні програми, що встановлюються на комп'ютер користувача (наприклад, AutoCAD, SolidWorks, 3ds Max). Вони надають максимальну продуктивність та функціональність завдяки прямому доступу до ресурсів комп'ютера. Їхнім головним недоліком є високі вимоги до апаратного забезпечення, вартість ліцензій та складнощі під час спільної роботи;
- хмарні та веборієнтовані системи є сучасним підходом, де обчислення та зберігання даних відбуваються на віддалених серверах, а користувач взаємодіє з системою через веббраузер (наприклад, Autodesk Fusion 360, Tinkercad). Цей підхід надає мобільність, легкість доступу та спрощує колаборацію. Але він має високі вимоги до швидкості інтернет-з'єднання та може мати обмеження у продуктивності порівняно з desktop-аналогами.

Проєкт, що розробляється, належить саме до веборієнтованих систем, що відповідає сучасним тенденціям надання послуг за моделлю SaaS (Software as a Service).

1.4 Огляд CAD/BIM/PDM-систем, орієнтованих на будівництво

Для глибокого розуміння контексту розробки необхідно проаналізувати професійні системи, що є стандартом у будівельній та виробничій галузях.

CAD (Computer-Aided Design) системи є основою сучасного інженерного проєктування. Вони дозволяють створювати точні двовимірні креслення та тривимірні моделі об'єктів.

Autodesk AutoCAD історично одна з найперших та найпоширеніших CAD-систем. Її основний фокус залишається на 2D-кресленні, але сучасні версії мають інструменти для 3D-моделювання.

SolidWorks – популярна CAD-система, що спеціалізується на параметричному 3D-моделюванні твердих тіл. Вона широко використовується в машинобудуванні та промисловому дизайні для проєктування деталей та збірок.

Rhinoceros 3D (Rhino) – це 3D-моделер, що базується на математиці NURBS (non-uniform rational B-spline). Це дозволяє створювати складні криволінійні поверхні довільної форми, що є важливим для дизайну виробів з каменю (наприклад, складні вирізи, заокруглення, фаски). Ключовою перевагою Rhino є його плагін Grasshopper – середовище візуального програмування, що дозволяє створювати складні параметричні та генеративні алгоритми без написання коду.

FreeCAD є безкоштовною CAD-системою з відкритим вихідним кодом. Вона є повністю параметричною та має модульну архітектуру, що розширює її функціональність.

BIM (Building Information Modeling) системи представляють наступний еволюційний крок після CAD. BIM-модель – це не просто геометрія, а цифровий двійник реального об’єкта.

BIM (Building Information Modeling) – це підхід, що базується на створенні цифрової інформаційної моделі об’єкта. Для онлайн-конфігуратора, що розробляється, це означає можливість передавати параметричні моделі у BIM-системи, використання стандартів IFC для сумісності з CAD, можливість інтегрувати замовлення у загальний процес будівництва.

BIM поступово стає стандартом у будівельній галузі. Його основна перевага полягає у створенні інформаційної моделі, яка містить не лише геометричні дані, а й усю необхідну інформацію про матеріали, характеристики та технологію виробництва. Застосування BIM-підходів у розробці вебсервісів для моделювання стільниць дозволяє досягти високого рівня автоматизації, підвищити точність та знизити витрати. Веборієнтовані рішення з інтегрованими інструментами 3D-візуалізації відкривають можливість прямої взаємодії між замовником і виробником: клієнт може самостійно налаштувати параметри виробу, отримати реалістичне зображення та сформувати замовлення, а виробник – зекономити час на узгодженні та уникнути непорозумінь.

Autodesk Revit – лідер ринку BIM-програмного забезпечення. У Revit кожен елемент (стіна, вікно) є об’єктом з набором параметрів. Зміна одного параметра (наприклад, ширини вікна) автоматично оновлює модель, креслення, специфікації та кошториси. Розроблювана система може генерувати модель стільниці, яку в майбутньому можна буде експортувати як параметричне сімейство для Revit (у форматі .rfa), що важливо для архітекторів та дизайнерів інтер’єрів.

PDM (Product Data Management) системи – це програмне забезпечення, призначене для управління всіма даними, пов’язаними з продуктом (CAD-моделі, креслення, специфікації, інструкції, інформацію про зміни версій, статус затвердження тощо). Прикладами є Autodesk Vault, SolidWorks PDM.

1.5 Порівняльний аналіз підходів до параметричного моделювання

CAD-скриптинг на C#, Python дає змогу реалізувати складні геометрії з високою точністю, але потребує глибоких технічних знань. Візуальне програмування (Grasshopper, Dynamo) дозволяє швидко створювати прототипи та інтерактивно змінювати параметри, проте часто не масштабоване для складних задач. WebGL-додатки з використанням Open CASCADE (OCCT) у поєднанні з Three.js забезпечують доступність та інтеграцію в браузер, але мають обмеження щодо складних конструкцій. Таким чином, кожен підхід має свої сильні та слабкі сторони: скриптинг – для складних задач, VP – для швидкого тестування ідей, WebGL – для доступних онлайн-сервісів.

1.6 Технології параметричного 3D-моделювання у вебсередовищі для реалізації інформаційних систем

Мови програмування JavaScript (клієнт), TypeScript, Python/PHP/Node.js (бекенд). Фреймворки React.js/Vue.js для фронтенду, Node.js або Django для серверної частини [16].

3D-бібліотеки WebGL (WebGPU API) – низькорівнева технологія рендерингу 3D-графіки у браузері, Three.js – високорівневий JavaScript-фреймворк для роботи з WebGL, Babylon.js – сучасна бібліотека з підтримкою фізики та PBR-матеріалів.

WebGL (скорочення від Web Graphics Library – бібліотека вебграфіки) – це JavaScript API для візуалізації інтерактивної 2D- та 3D-графіки в будь-якому сумісному веббраузері без використання плагінів. WebGL повністю інтегрований з іншими вебстандартами, що дозволяє використовувати фізику, обробку зображень та ефекти на HTML-полотні (canvas) з прискоренням на графічному процесорі. Елементи WebGL можна змішувати з іншими елементами HTML та комбінувати з іншими частинами сторінки або фоном сторінки.

Програми WebGL складаються з керуючого коду, написаного на JavaScript, та коду шейдерів, написаного мовою шейдингу OpenGL ES (GLSL ES, іноді званої ESSL), мовою, подібною до C або C++. Код WebGL виконується на графічному процесорі комп'ютера.

WebGL розроблено та підтримується некомерційною організацією Khronos Group. 9 лютого 2022 року Khronos Group оголосила про підтримку WebGL 2.0 усіма основними браузерами.

З 2024 року розробляється новий графічний API, WebGPU , який замінить WebGL. WebGPU забезпечує розширені можливості, сучасніший інтерфейс та прямий доступ до GPU, що корисно як для вимогливої графіки, так і для програм штучного інтелекту [6].

Three.js – це кросбраузерна бібліотека JavaScript та інтерфейс прикладного програмування (API), що використовується для створення та відображення анімованої 3D-графіки у веббраузері за допомогою WebGL. Вихідний код розміщено в репозиторії на GitHub.

Three.js дозволяє створювати 3D-анімацію з прискоренням за допомогою графічного процесора (GPU) за допомогою мови JavaScript як частину вебсайту без використання власних плагінів браузера. Це стало можливим завдяки появі WebGL, низькорівневого графічного API, створеного спеціально для вебсередовища.

Високорівневі бібліотеки, такі як Three.js, Babylon.js, Verge3D та багато інших, дозволяють створювати складні 3D-анімації для відображення у браузері без зусиль, необхідних для традиційного окремого додатку чи плагіна [7].

Babylon.js – це бібліотека JavaScript та 3D-рушій для відображення 3D-графіки в реальному часі у веббраузері через HTML5 [27].

Серверні технології Node.js або Django, REST API, WebSocket.

Бази даних MySQL або PostgreSQL для збереження параметрів моделей і замовлень [16].

Системи інтеграції з онлайн-оплатами Stripe, LiqPay, WayForPay.

BIM-інтеграція – використання форматів IFC для обміну даними.

Параметричне моделювання базується на декларативному описі геометричних залежностей та обмежень, тобто зміна одного параметра автоматично спричинює перерахунок всієї моделі. Такий підхід дозволяє моделювати широкий спектр виробів із різними розмірами, формами та матеріалами, забезпечуючи високу гнучкість і масштабованість проєктів.

1.7 Постановка задачі

Головним завданням магістерської роботи буде створення онлайн-конструктора як зручного сервісу для оформлення замовлень з обробки індивідуальних стільниць з мармуру з 3D-візуалізацією деталей та обробок.

Завдання для досягнення головної мети:

- спроєктувати архітектуру системи (структуру бази даних, серверної та клієнтської частин);
- розробити програмні модулі системи (модуль параметричного 3D-моделювання та візуалізації, модуль особистого кабінету користувача, модуль управління замовленнями);
- реалізувати прототип системи з можливістю створення параметричних моделей індивідуальних виробів;
- провести тестування та оцінити ефективність рішення.

Побудуємо дерево цілей системи, що розробляється (рисунок 1.9).

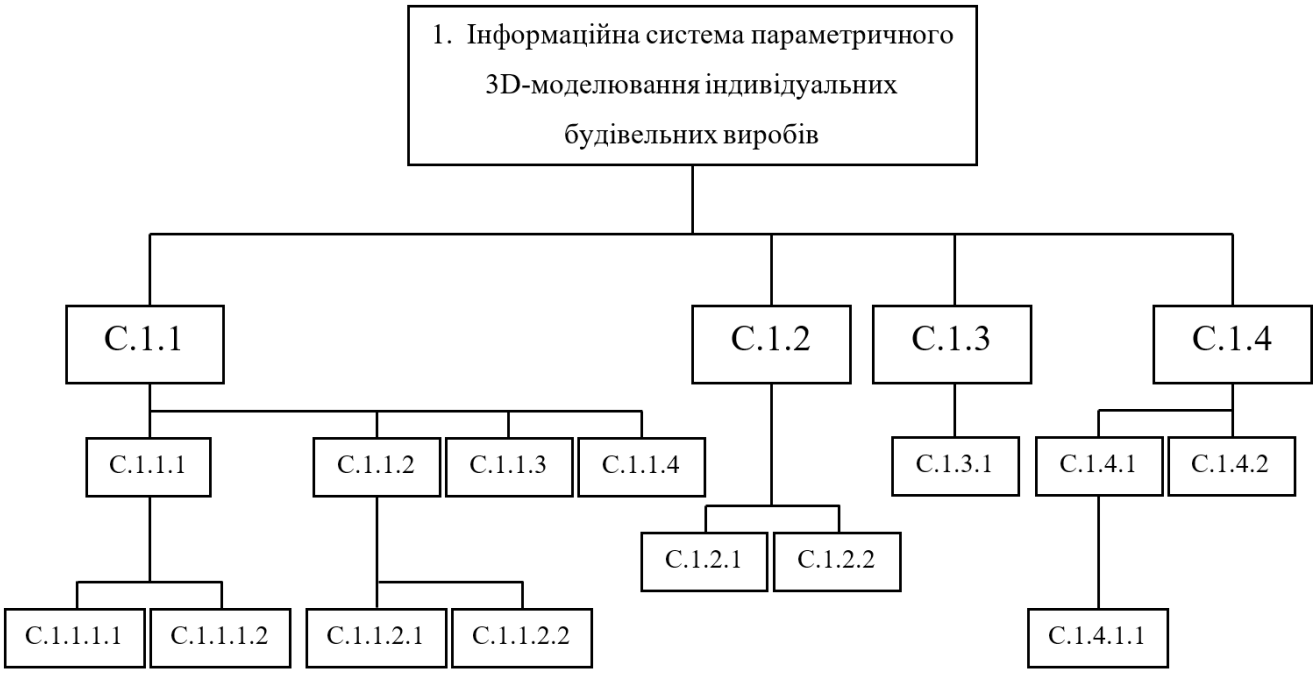


Рисунок 1.9 – Діаграма дерева цілей побудови системи параметричного 3D-моделювання

С.1.1 Розробити та впровадити систему параметричного 3D-моделювання

С.1.1.1 Визначити функціональні вимоги

С.1.1.1.1 Проаналізувати потреби користувачів

С.1.1.1.2 Визначити основні функції системи

С.1.1.2 Спроєктувати архітектуру системи

С.1.1.2.1 Розробити структуру бази даних

С.1.1.2.2 Визначити взаємодію клієнтської та серверної частин

С.1.1.3 Розробити програмні модулі

С.1.1.4 Впровадити прототип у вебсередовище

С.1.2 Надати можливість параметричного 3D-моделювання виробів

С.1.2.1 Реалізувати модуль побудови геометрії

С.1.2.2 Налаштувати змінні параметри виробу

С.1.3 Забезпечити зручність користування системою

С.1.3.1 Розробити інтерфейс користувача

С.1.4 Провести тестування та оцінку ефективності

С.1.4.1 Виконати функціональне тестування

С.1.4.1.1 Перевірити коректність 3D-візуалізації

С.1.4.2 Перевірити точність візуалізації та розрахунків

1.8 Висновки

Проведений аналіз показав, що існуючі рішення не дають можливості простого онлайн-проєктування індивідуальних виробів з повноцінною 3D-візуалізацією, інтеграцією замовлень і BIM-сумісністю.

Для вирішення цієї проблеми необхідно створити інформаційну систему у вигляді онлайн-конструктора, яка поєднує простоту інтерфейсу з функціоналом параметричного моделювання та інтерактивною 3D-візуалізацією. Очікуваними перевагами є скорочення часу та вартості проєктування, зменшення помилок під час виготовлення та підвищення зручності для кінцевого користувача.

2 ПРОЄКТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ

2.1 Розробка архітектури для компанії, що займається виготовленням індивідуальних будівельних виробів

Мотиваційна модель архітектури компанії

1. Діаграма бачення місії, цінності та бізнесу підприємства (Mission-Values-Vision View). Основні цінності підприємства визначено на рис. 2.1.

Основними цінностями компанії є:

- якість продукції;
- клієнтоорієнтованість;
- інноваційність;
- швидкість і зручність доставки;
- чесність і прозорість.

Рисунок 2.1 – Цінності компанії

Місія компанії – створити комфортне середовище для придбання індивідуальних виробів. Дерево цілей підприємства наведено на рис. 2.2.

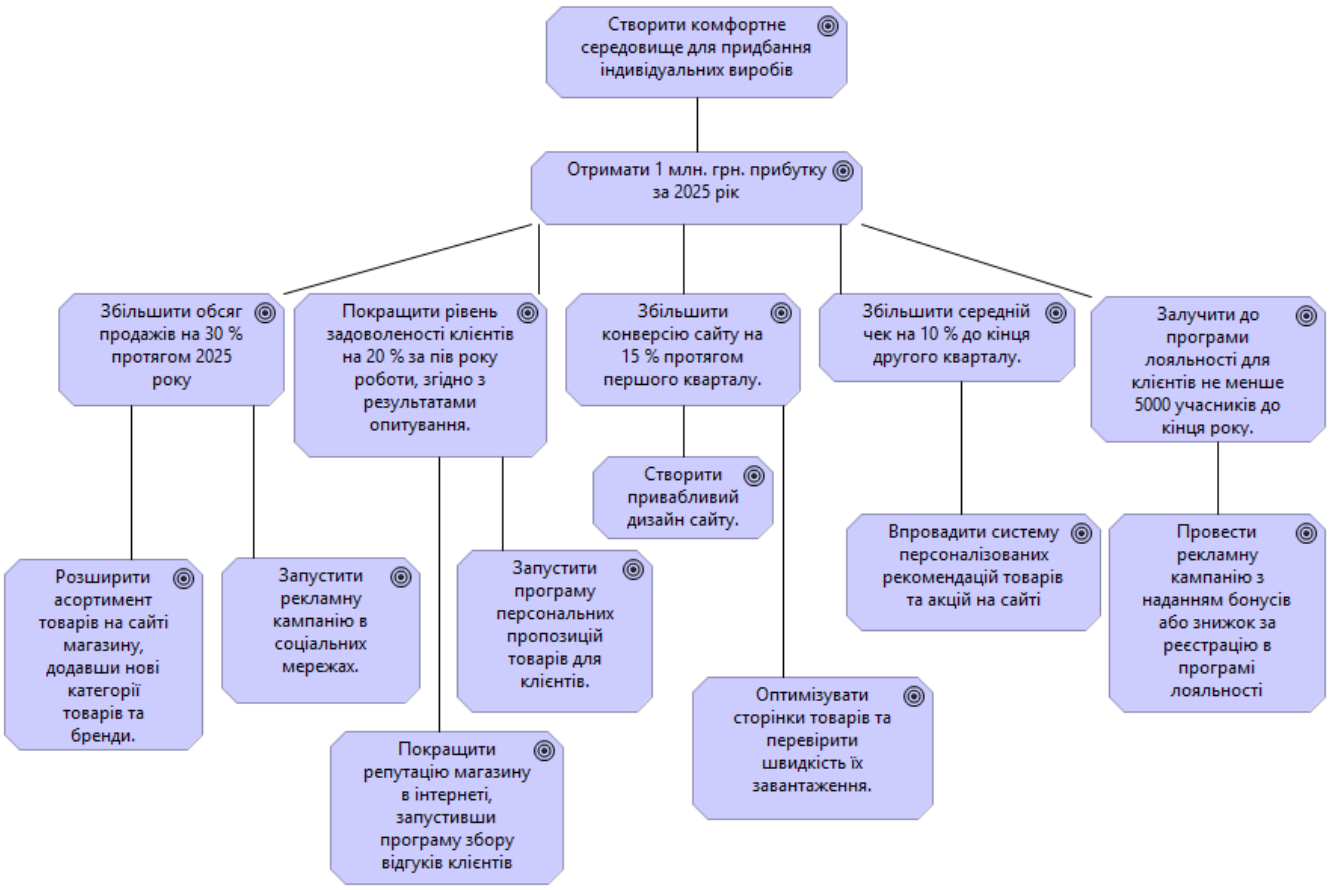


Рисунок 2.2 – Цілі компанії

2. Діаграма бачення стратегії підприємства (Strategic-Value-Map View). Стратегії можна поділити на декілька шарів, а саме фінансові перспективи, цінності запропоновані клієнтам, внутрішні перспективи, перспективи розвитку. Результат наведено на рис. 2.3.

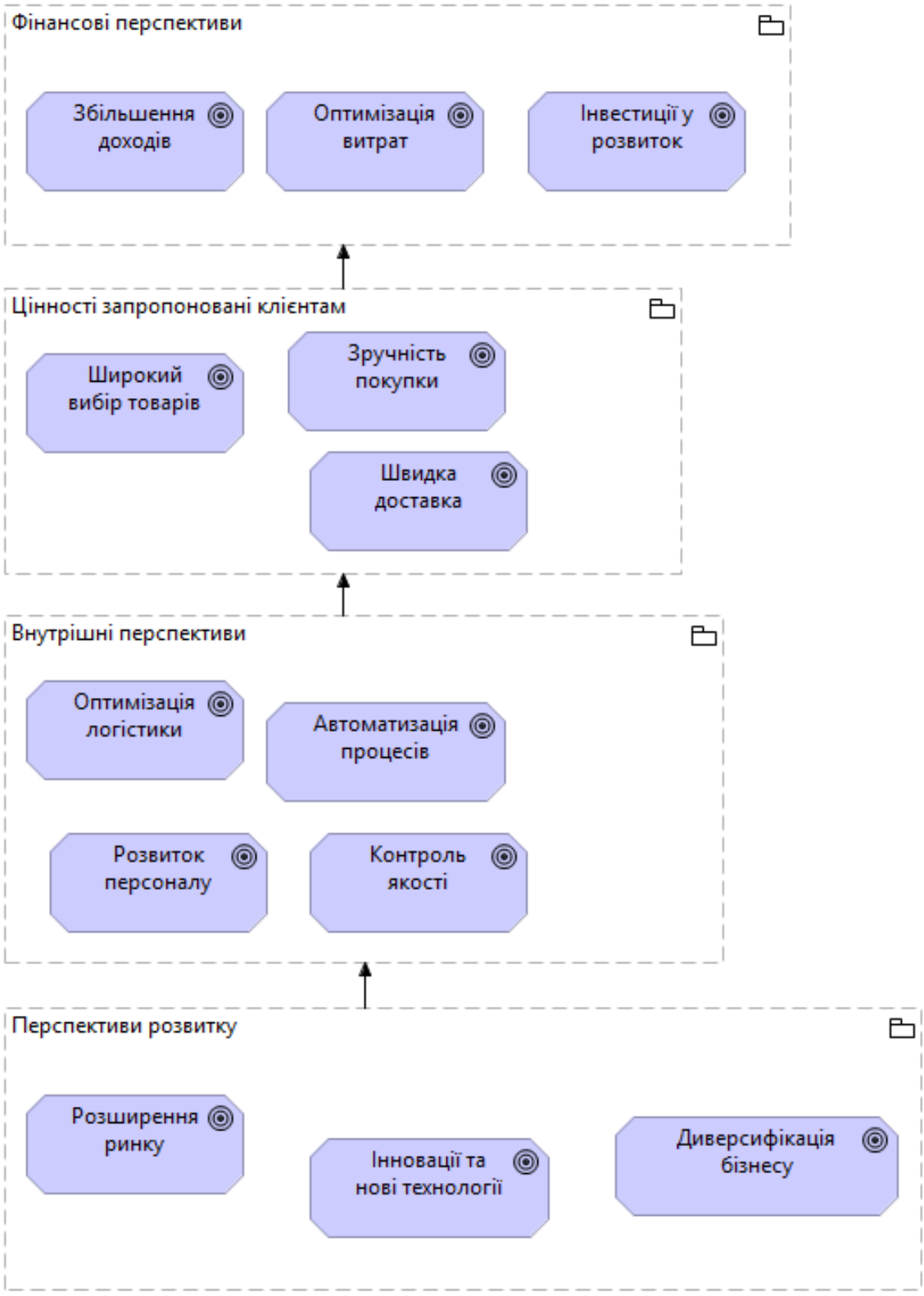


Рисунок 2.3 – Діаграма бачення стратегії підприємства

Бізнес-архітектура підприємства

1. Організаційна структура компанії (рисунок 2.4).

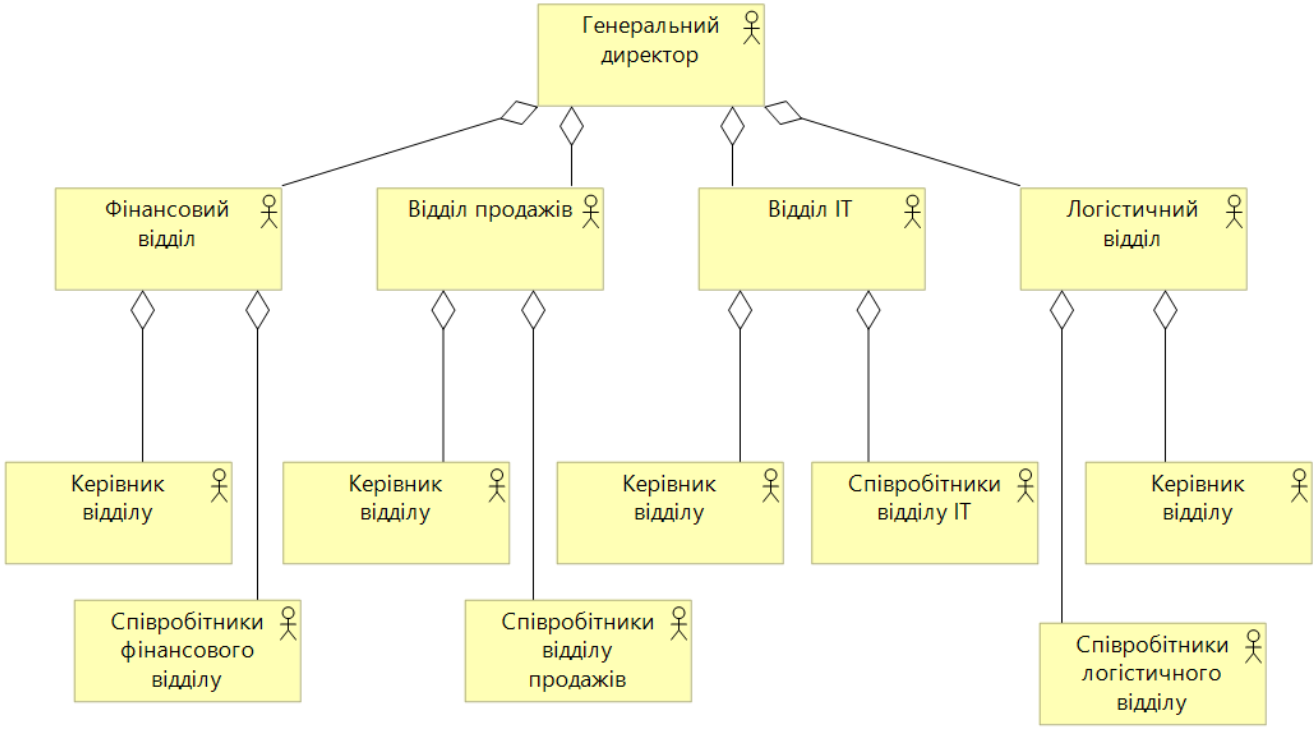


Рисунок 2.4 – Організаційна структура

Основні бізнес-процеси у роботі відділу IT та бізнес-ролі, тих хто їх виконує:

а) розробка та підтримка вебсайту;

1) розробка інтерфейсу вебсайту – Frontend-розробник;

2) розробка серверної частини – Backend-розробник;

б) управління базами даних – адміністратор баз даних;

в) підтримка інфраструктури;

1) налаштування та підтримка серверів, мереж – системний адміністратор;

2) автоматизація процесів розгортання, масштабування та підтримки інфраструктури – DevOps Engineer.

Ролі показані на рис. 2.5.

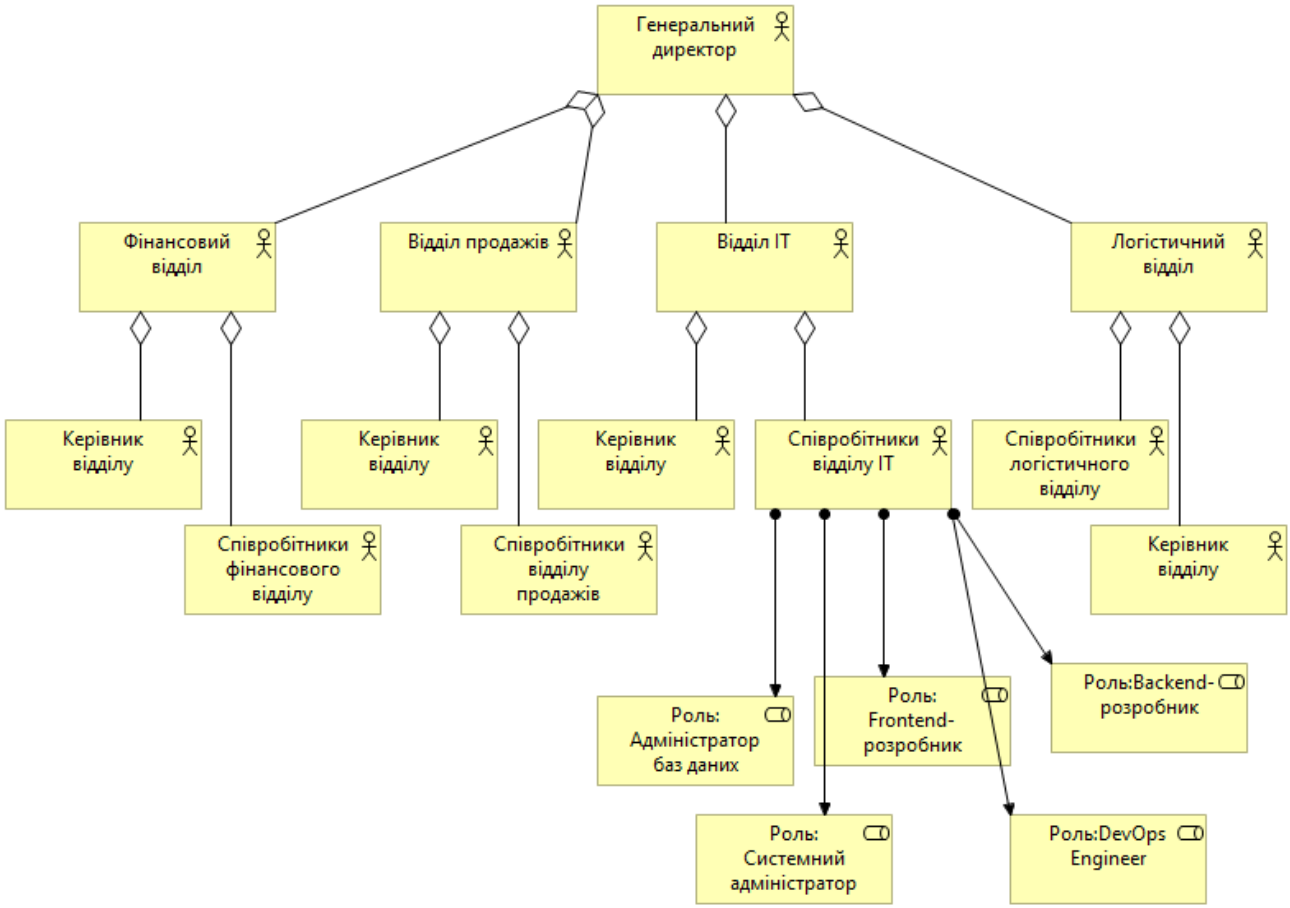


Рисунок 2.5 – Організаційна структура та ролі

2. Бізнес-архітектура відділу IT зображена на рис. 2.6.

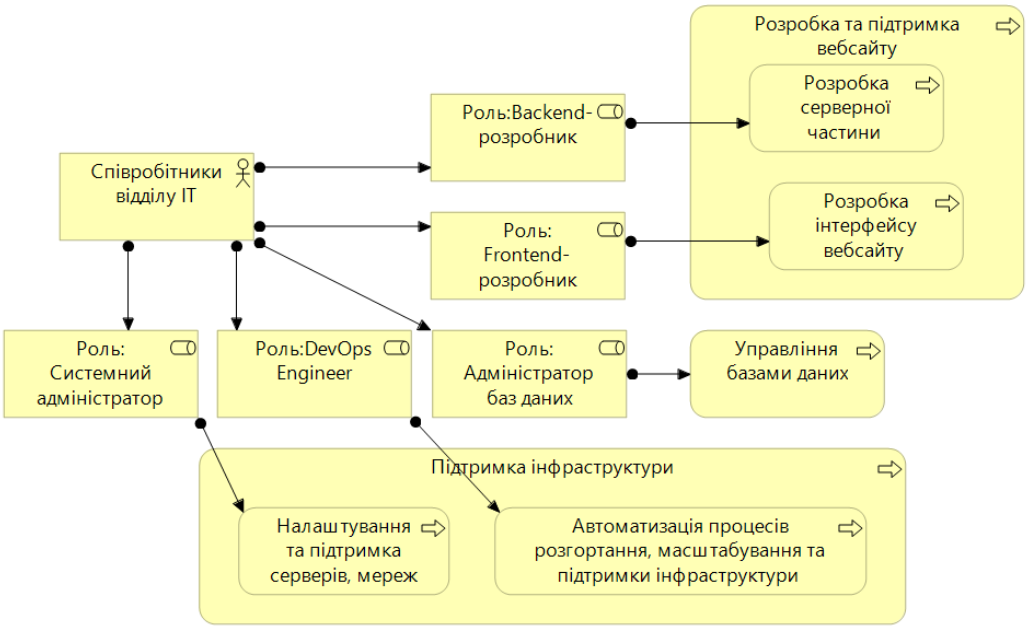


Рисунок 2.6 – Бізнес-архітектура відділу IT

Системна архітектура підприємства

1. Основні додатки та їх карта (рисунок 2.7).

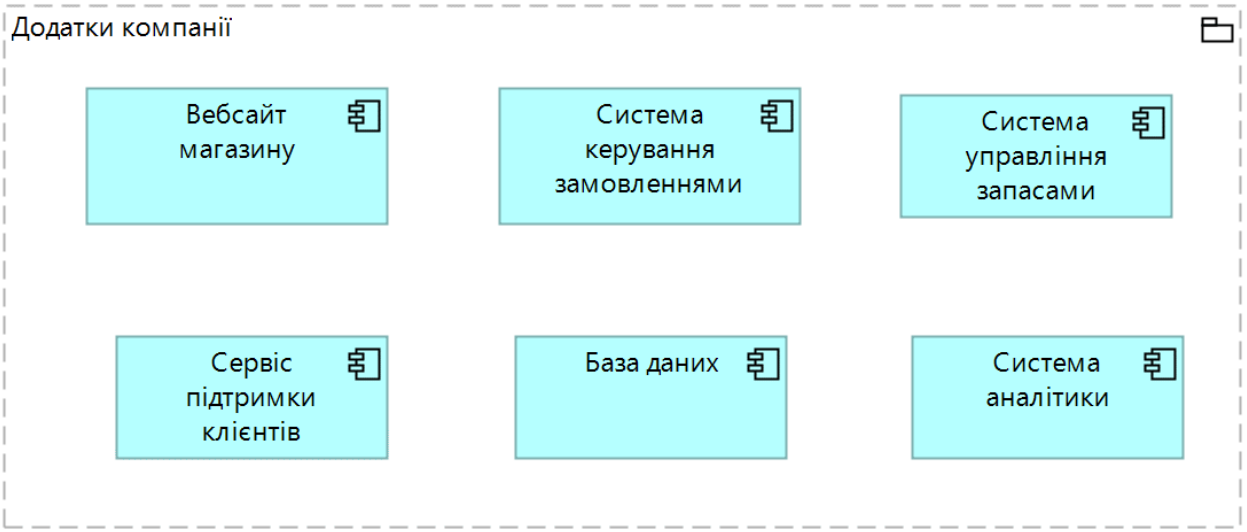


Рисунок 2.7 – Карта додатків

Їх взаємодію показано на рис. 2.8.

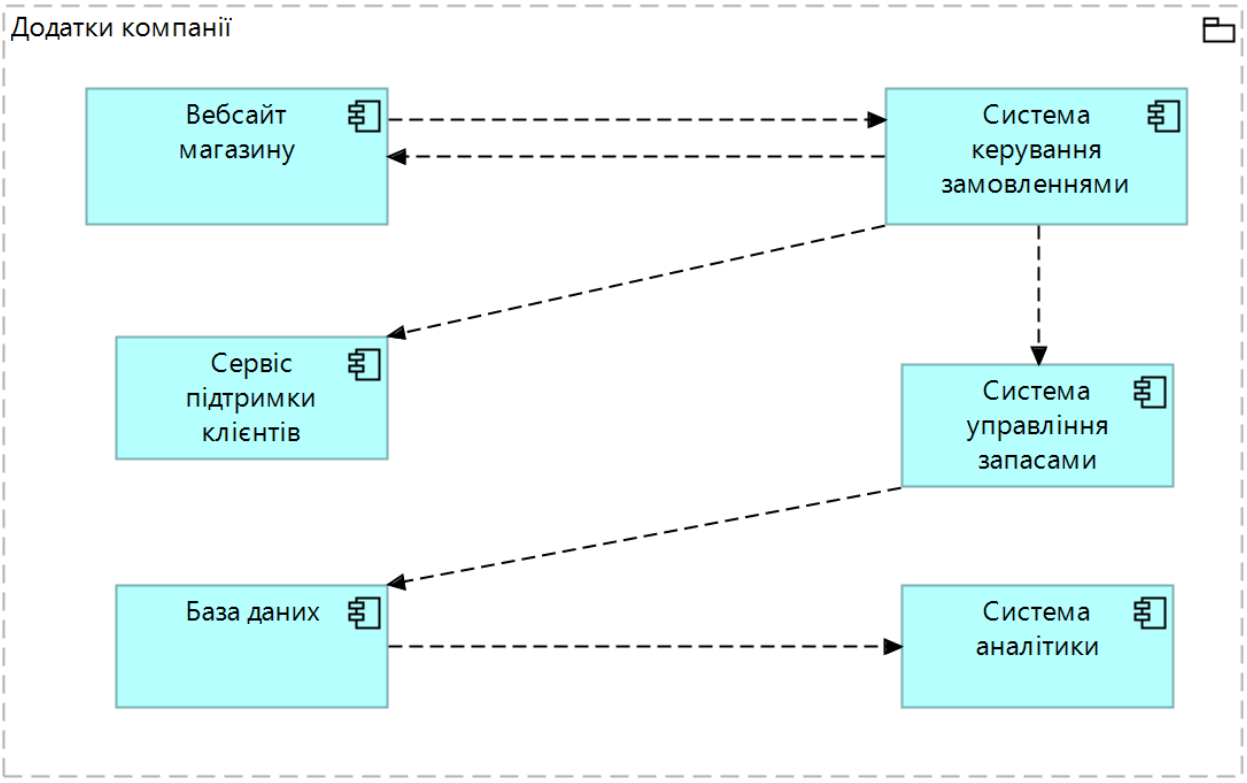


Рисунок 2.8 – Взаємодія додатків

Вебсайт магазину взаємодіє з системою керування замовленнями (рисунок 2.9):

– Вебсайт магазину реалізує сервіс «Збір інформації про замовлення», який використовується Системою керування замовленнями;
– Система керування замовленнями формує документ «Статус замовлення», який повертається на Вебсайт магазину.

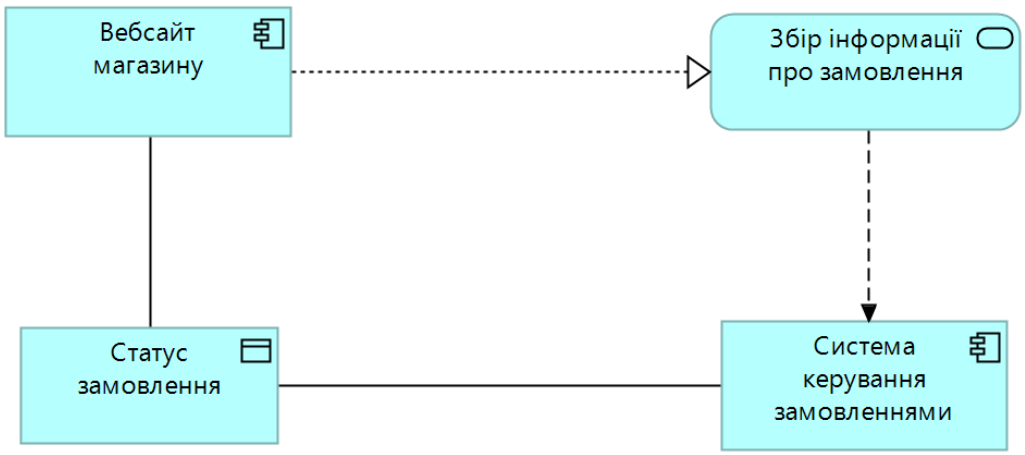


Рисунок 2.9 – Діаграма інтеграції прикладного програмного забезпечення

Структура вебсайту магазину складається з наступних модулів (рисунок 2.10):

– модуль авторизації;
– модуль управління товарами;
– модуль замовлення;
– модуль оплати;
– модуль доставки.

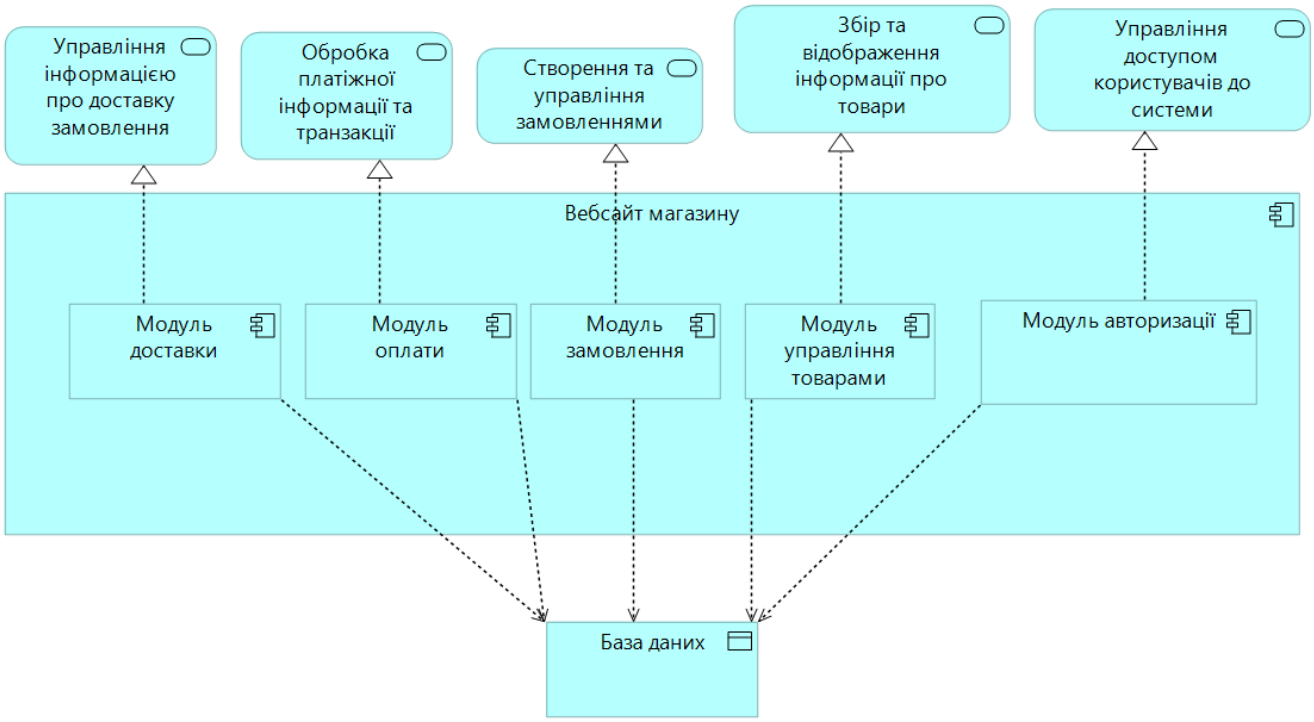


Рисунок 2.10 – Діаграма представлення структури прикладної програми підприємства

2. Діаграма бачення інфраструктури підприємства (рисунок 2.11) та технологічного забезпечення прикладного рівня архітектури підприємства (рисунок 2.12).

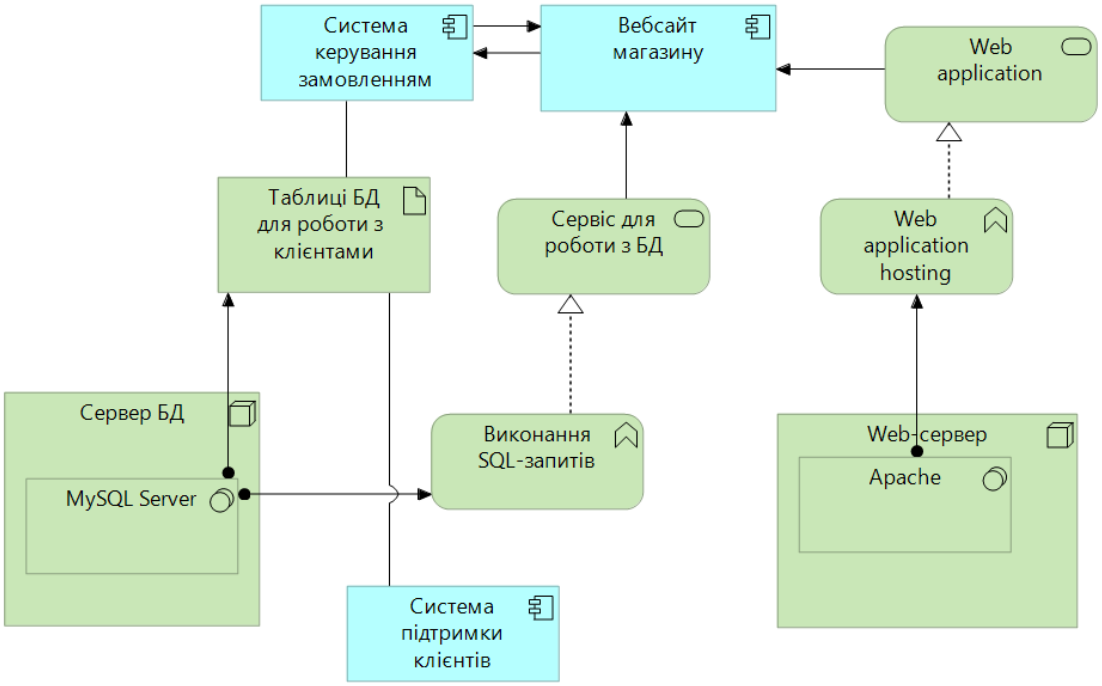


Рисунок 2.11 – Діаграма бачення інфраструктури підприємства

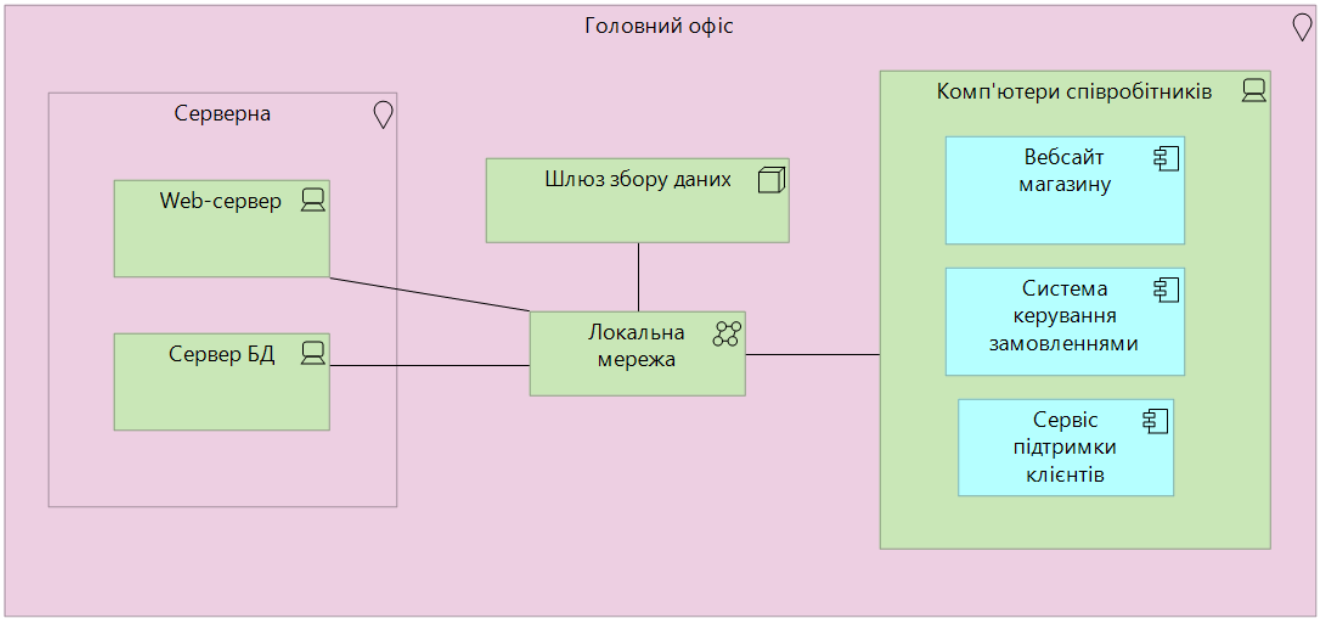


Рисунок 2.12 – Діаграма бачення технологічного забезпечення прикладного рівня архітектури підприємства

2.2 Аналіз основних функцій клієнтської частини онлайн-конструктора

Сегментація цільової аудиторії за методом «5W» Марка Шеррінгтона

1. What? – Що саме потрібно аудиторії?

Цільова аудиторія потребує інструмента, що дозволяє:

- створити параметричну 3D-модель стільниці;
- отримати реалістичну візуалізацію матеріалів (мармур, кварц, граніт);
- визначити габарити, форму вирізів, типи крайок та обробки;
- оформити онлайн-замовлення без відвідування виробництва.

2. Who? – Хто є користувачами?

Групи користувачів:

- індивідуальні замовники – фізичні особи;
- дизайнери інтер’єрів – готують проєкти приміщень;
- менеджери салонів сантехніки та меблів;

– виробничі підприємства – внутрішнє використання для прискорення комунікації з клієнтом.

3. Why? – Чому вони це роблять?

Потреби користувачів:

– зменшення кількості вимірювань і уточнень;
– миттєва візуалізація варіантів;
– отримання об’єктивної вартості;
– прискорення процесу замовлення (до 15 хвилин замість декількох днів).

4. When? – Коли використовується інструмент?

– підготовка ремонту;
– узгодженні дизайну приміщення;
– розробка креслень стільниць;
– на етапі комерційних пропозицій у салонах та шоурумах.

5. Where? – Де відбувається використання?

– будь-який браузер на ПК чи планшеті;
– мобільні пристрої (у спрощеній версії);
– торгові точки, офіси, домівки користувачів.

Дерево функцій клієнтської частини програми складається з чотирьох головних функцій, які в свою чергу складаються з менших функцій (рисунок 2.13).

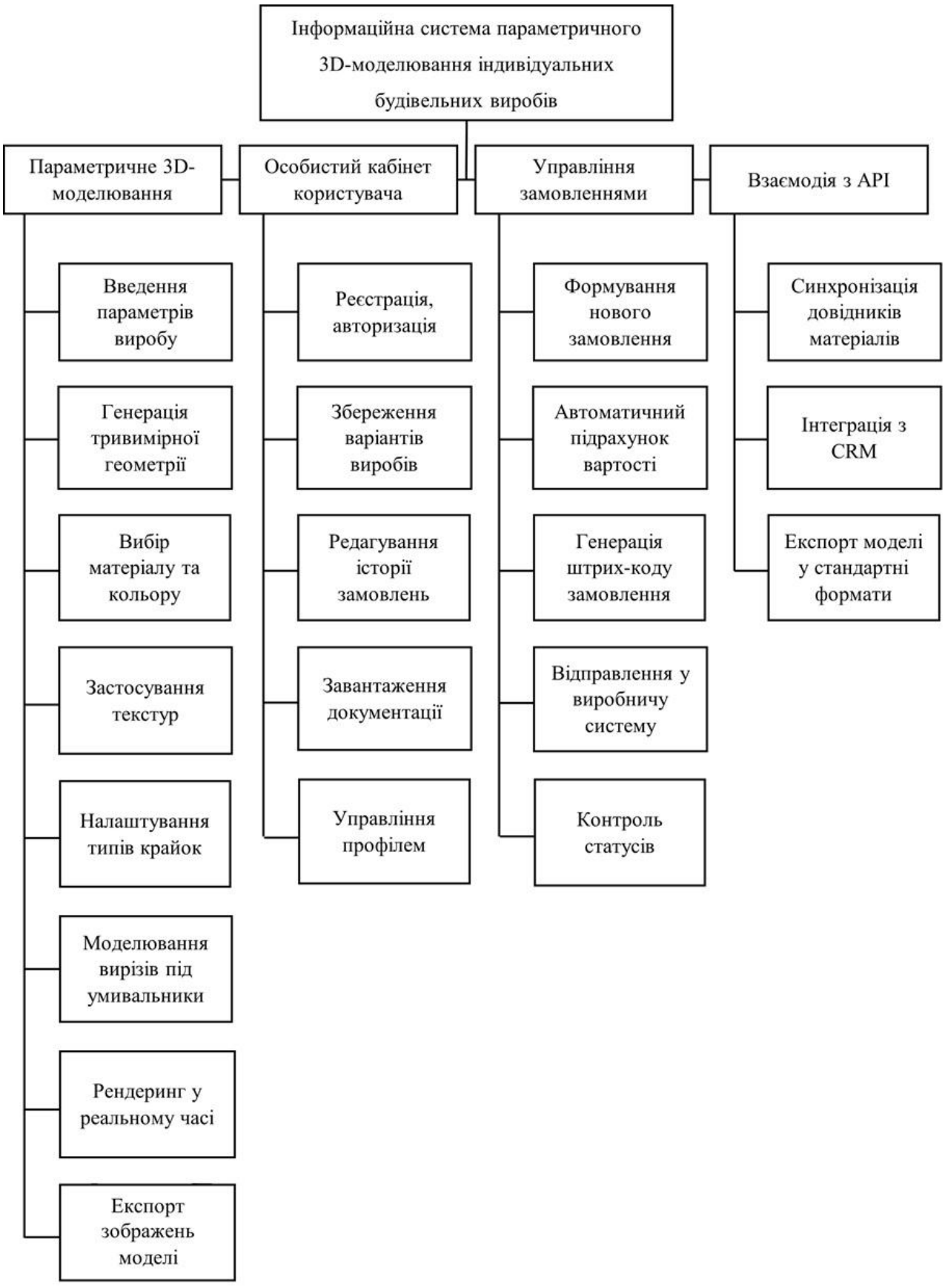


Рисунок 2.13 – Діаграма дерева функцій системи параметричного 3D-моделювання

Головною задачею онлайн-конструктора є формування замовлення.

Процес складається з таких етапів:

1. Користувач відкриває конфігуратор.
2. Заповнює параметри виробу: довжину, ширину, товщину, матеріал, тип крайки.
3. На основі параметрів формується 3D-модель.
4. Користувач коригує геометрію, додає вирізи.
5. Система розраховує вартість.
6. Користувач переходить до оформлення замовлення.
7. Після авторизації замовлення надсилається до сервера.
8. Сервер присвоює унікальний ID та генерує штрих-код.
9. Замовлення передається у виробничу систему через API.

Опис IDEF0-моделі бізнес-процесу «Замовлення індивідуального виробу» наведено на рис. 2.14.

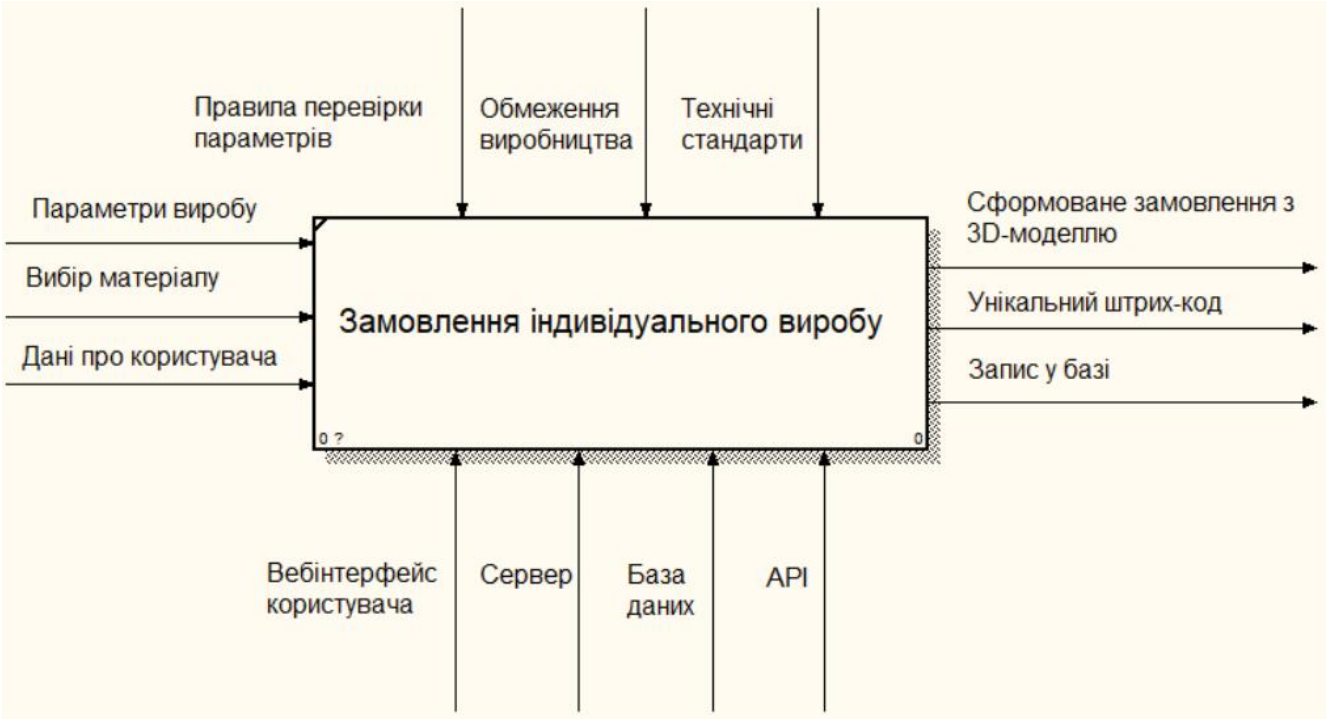


Рисунок 2.14 – Бізнес-процес «Замовлення індивідуального виробу»

2.3 Архітектурна модель системи

2.3.1 Загальна концепція побудови системи

Розроблення інформаційної системи параметричного 3D-моделювання індивідуальних будівельних виробів орієнтоване на створення інструменту, що функціонує без встановлення спеціалізованого програмного забезпечення. Потреба в такому рішенні обумовлена ростом попиту на онлайн-конфігуратори, за допомогою яких підприємства можуть значно прискорити процес оформлення замовлень і забезпечити клієнтів візуалізацією виробів у реальному часі.

На відміну від традиційних CAD-систем, описувана інформаційна система орієнтована на роботу в браузері, що накладає особливості на її архітектуру, методи реалізації та способи обробки графічних моделей. Параметричний підхід дозволяє на основі обмеженої кількості вхідних параметрів отримувати складну 3D-геометрію, яку можна динамічно оновлювати під час кожної зміни характеристик виробу.

Система повинна забезпечити:

- можливість створення параметричної 3D-моделі стільниці для умивальника за введеними користувачем даними;
- автоматичну побудову моделі в браузері без перезавантаження сторінки;
- збереження параметрів виробу, матеріалів та інформації про користувача в централізованій базі даних;
- надання зручного інтерфейсу для формування та уточнення замовлення;
- інтеграцію з внутрішніми системами підприємства через API.

2.3.2 Вибір архітектурного підходу

Трирівнева архітектура – це добре налагоджена архітектура програмних застосунків, яка організовує застосунки на три логічні та фізичні обчислювальні рівні: рівень представлення або інтерфейс користувача; рівень застосунків, де обробляються дані; та рівень даних, де зберігаються та управляються дані застосунків. Головна перевага трирівневої архітектури полягає в тому, що оскільки кожен рівень працює на власній інфраструктурі, кожен рівень може розроблятися одночасно окремою командою розробників. І його можна оновлювати або масштабувати за потреби, не впливаючи на інші рівні [24].

Базовою вимогою до системи є хмарна клієнт-серверна архітектура, що відповідає трирівневій моделі: рівень презентації, бізнес-логіки, рівень даних, доповнені інтеграційним рівнем.

Для реалізації інформаційної системи доцільно застосувати багаторівневу архітектуру (N-Tier Architecture), оскільки:

- вона забезпечує чітке розділення відповідальностей між частинами системи;
- дозволяє незалежно масштабувати клієнтську та серверну частини;
- підтримує модульність і можливість подальших змін;
- спрощує інтеграцію з зовнішніми сервісами через API.

Загальна структура наведена на рис. 2.15:

- рівень презентації (Presentation Layer) – вебінтерфейс для візуалізації 3D-моделей;
- рівень бізнес-логіки (Business Layer) – серверна частина, яка виконує обробку параметрів виробів, контроль доступу, логіку моделювання та формування запитів до бази даних;
- рівень даних (Data Layer) – управління базою даних;
- інтеграційний рівень (Integration Layer) – API для зв’язку з виробничими системами, CRM, генераторами документації, системами логістики тощо.

Рисунок 2.15 – Системна архітектура прототипу конфігуратора

2.4 Розробка бази даних

2.4.1 Логічне проєктування структури бази даних

Розробка онлайн-конструктора для індивідуальних будівельних виробів, зокрема стільниць з натурального каменю, висуває специфічні вимоги до підсистеми зберігання та обробки даних. На відміну від класичних облікових систем, де структура даних є статичною та чітко визначеною заздалегідь, системи параметричного 3D-моделювання оперують сутностями, що мають динамічну природу. Специфіка параметрів предметної області, наприклад, складна геометрія, варіативність вирізів та налаштування текстур, погано узгоджується з табличною архітектурою реляційних баз даних. Адаптація цих даних через глибоку нормалізацію призводить до створення громіздких SQL-запитів, що негативно впливає на продуктивність системи.

Враховуючи необхідність зберігання ієрархічних структур та забезпечення високої швидкості обміну даними між клієнтською частиною та сервером, було вирішено використовувати документо-орієнтований підхід.

Основні переваги такого підходу для предметної області інформаційної системи:

1. Гнучкість схеми – це важливо на етапі, коли нові типи виробів можуть додаватися в систему;
2. Природне відображення об’єктів – дані зберігаються у форматі, наближеному до JSON (JavaScript Object Notation), що є нативним форматом для вебдодатків та бібліотек 3D-графіки;
3. Агрегація даних – це можливість зберігати пов’язані дані (наприклад, параметри геометрії) всередині одного документа, що зменшує кількість операцій об’єднання (JOIN) та пришвидшує читання.

На етапі логічного проєктування визначено ключові сутності предметної області, їхні атрибути та зв’язки, абстрагуючись від конкретної системи управління базами

даних (СУБД). Система базується на п’яти основних сутностях: Користувачі, Замовлення, Параметричні моделі, Матеріали та Текстури.

Нижче наведено детальний опис атрибутів кожної сутності у вигляді таблиць (таблиці 2.1 – 2.5).

Таблиця 2.1 – Структура сутності «Користувачі» (Users)

Назва атрибута	Тип даних	Опис та призначення
ID	Унікальний ідентифікатор	Первинний ключ для однозначної ідентифікації користувача в системі
Full Name	Рядок	Прізвище, ім’я та по батькові користувача. Використовується для персоналізації інтерфейсу та оформлення документів
Email	Рядок	Унікальна адреса електронної пошти. Використовується як логін для входу в систему та канал комунікації
Password Hash	Рядок (хеш)	Зашифрований рядок пароля. Зберігання паролів у відкритому вигляді заборонено політиками безпеки
Role	Перелік (Enum)	Роль користувача в системі: Клієнт (доступ до конструктора), Менеджер (обробка замовлень), Адміністратор (налаштування системи)
Created At	Дата/Час	Часова мітка реєстрації аккаунта
Orders List	Масив ідентифікаторів	Список посилань на замовлення, здійснені даним користувачем. Забезпечує швидкий доступ до історії

Таблиця 2.2 – Структура сутності «Замовлення» (Orders)

Назва атрибута	Тип даних	Опис та призначення
ID	Унікальний ідентифікатор	Первинний ключ замовлення
User Ref	Посилання (FK)	Ідентифікатор користувача, який оформив замовлення
Model Ref	Посилання (FK)	Ідентифікатор параметричної моделі, на основі якої створено виріб
Material Ref	Посилання (FK)	Ідентифікатор обраного матеріалу (каменю)
Parameters Snapshot	Вкладений об’єкт (JSON)	«Знімок» (Snapshot) конкретних параметрів виробу на момент замовлення (довжина, ширина, типи обробок). Це гарантує цілісність історії замовлення навіть під час зміни базової моделі
Price	Число (Decimal)	Розрахункова вартість виробу у національній валюті
Status	Перелік (Enum)	Поточний стан виконання: Створено, В роботі, Виконано, Скасовано
Barcode	Рядок	Унікальний штрих-код для ідентифікації замовлення на виробництві та логістиці
Created At	Дата/Час	Дата та час оформлення замовлення

Таблиця 2.3 – Структура сутності «Параметричні моделі» (Models)

Назва атрибута	Тип даних	Опис та призначення
ID	Унікальний ідентифікатор	Первинний ключ моделі
Name	Рядок	Користувацька назва конфігурації
Geometry Data	Вкладений об’єкт (JSON)	Комплексна структура даних, що описує геометрію: габарити, масив крайок, масив вирізів (із координатами та типами)
Preview Image	Рядок (URL)	Посилання на згенероване зображення (рендер) моделі для швидкого перегляду в списку
Updated At	Дата/Час	Час останньої модифікації параметрів моделі

Таблиця 2.4 – Структура сутності «Матеріали» (Materials)

Назва атрибута	Тип даних	Опис та призначення
ID	Унікальний ідентифікатор	Первинний ключ матеріалу
Name	Рядок	Комерційна назва каменю
Type	Рядок	Тип породи (мармур, граніт, кварцит)
Texture Ref	Посилання (FK)	Зв’язок із сутністю «Текстури» для візуалізації матеріалу в 3D-сцені
Cost Coeff	Число (Float)	Коефіцієнт вартості за квадратний метр, що використовується алгоритмом розрахунку ціни
Description	Текст	Маркетинговий опис матеріалу для користувача

Таблиця 2.5 – Структура сутності «Текстури» (Textures)

Назва атрибута	Тип даних	Опис та призначення
ID	Унікальний ідентифікатор	Первинний ключ текстури
File Path	Рядок (URL)	Шлях до графічного файлу текстури на сервері або в хмарному сховищі
Type	Рядок	Тип карти: Diffuse (колір), Normal (рельєф), Roughness (шорсткість)
Resolution	Рядок	Роздільна здатність зображення для оптимізації завантаження
Scale	Вкладений об’єкт	Коефіцієнти масштабування текстури (UV-mapping) по осях X та Y

Для забезпечення цілісності даних та коректної роботи бізнес-логіки між описаними сутностями встановлюються такі зв’язки:

1. Users – Orders (1 : N, «один-до-багатьох»). Один користувач може оформити необмежену кількість замовлень. В базі даних це реалізується через збереження userId в документі замовлення та дублювання масиву orders в документі користувача для оптимізації читання.
2. Orders – Models (1 : 1, «один-до-одного»). Кожне замовлення базується на унікальній конфігурації моделі. Хоча теоретично модель може використовуватися повторно, в контексті індивідуального виробництва створюється унікальний екземпляр моделі або посилання на неї.
3. Materials – Orders (1 : N, «один-до-багатьох»). Один і той самий матеріал може використовуватися в багатьох різних замовленнях.
4. Textures – Materials (1 : N, «один-до-багатьох»). Одна текстура може бути використана для декількох варіацій матеріалів.

Логічну схему бази даних зображено на рис. 2.16.

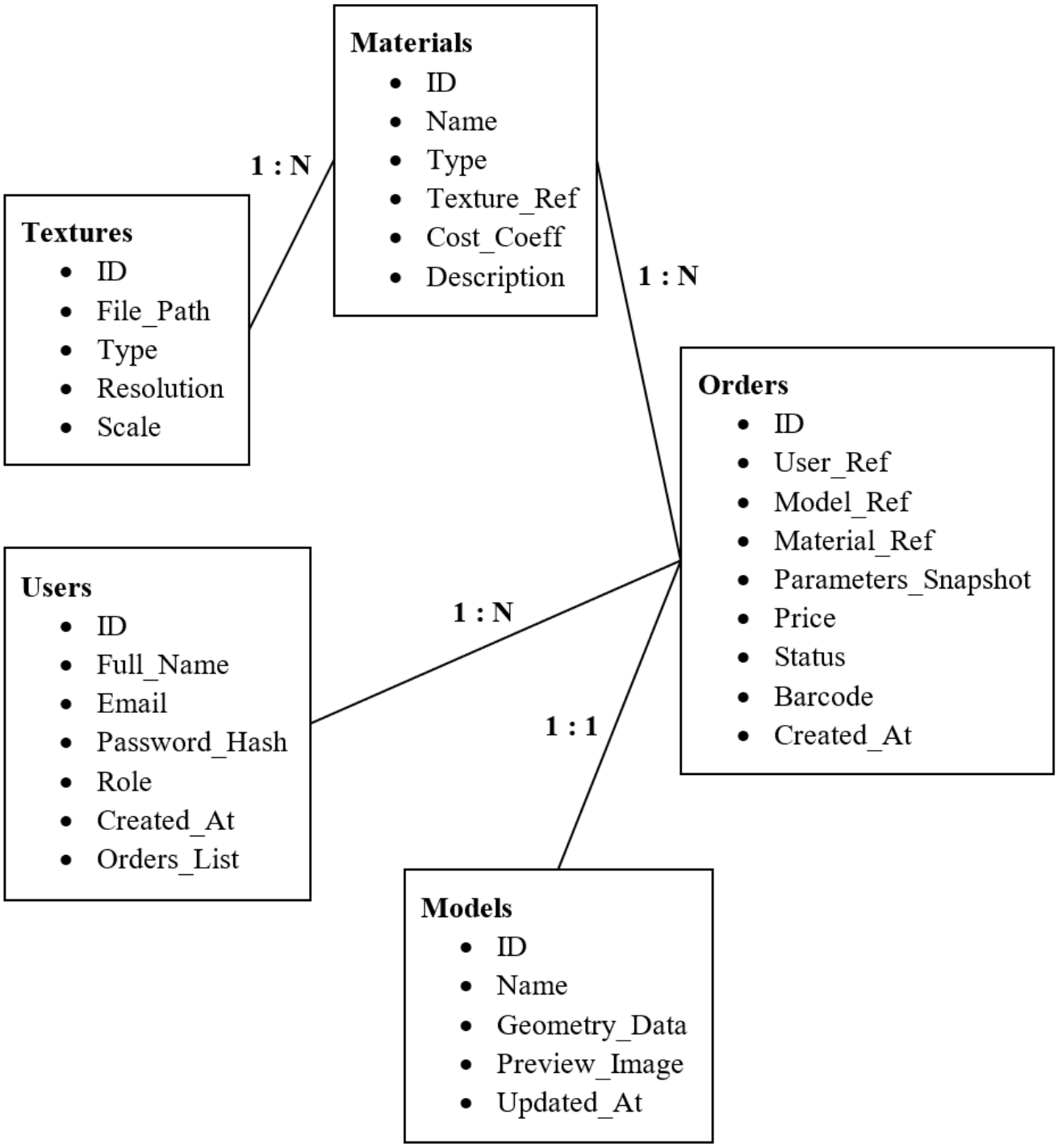


Рисунок 2.16 – Логічна схема бази даних

2.4.2 Фізична модель бази даних

Для реалізації спроєктованої логічної моделі було проведено порівняльний аналіз сучасних документо-орієнтованих СУБД. На ринку існує кілька рішень, серед яких найбільш релевантними є MongoDB, CouchDB та Redis (таблиця 2.6).

Таблиця 2.6 – Порівняльний аналіз документо-орієнтованих СУБД

Критерій порівняння	MongoDB	CouchDB	Redis
Модель даних	BSON (бінарний JSON) – звичайний текст, але стиснутий і з підтримкою точних чисел	JSON – простий текстовий формат, але займає більше місця	Дані зберігаються в оперативній пам’яті (RAM). Дані мають просту структуру (рядок, список)
Мова запитів	Має дуже гнучкі інструменти пошуку	Використовує складніший підхід, де треба писати окремі міні-програми для пошуку	Шукати можна переважно лише за конкретним ключем (ID)
Надійність запису (транзакції)	Надійно зберігає дані	Дані можуть оновитися не миттєво. Підходить для соцмереж, але ризиковано для складу	Дані швидко оновлюються

Продовження таблиці 2.6

Швидкість та масштабування	Легко розподіляється на багато серверів, якщо кількість користувачів різко зросте	Орієнтована на офлайн. Найкраще працює, коли інтернет нестабільний (синхронізація після підключення)	Найшвидша з усіх існуючих систем, бо не витрачає час на звернення до жорсткого диска
Популярність та підтримка	Найбільше готових рішень, інструкцій та фахівців. Легко знайти відповідь на будь-яке питання	Популярна для мобільних додатків, що працюють без інтернету	Використовується майже всюди, але переважно як додаток основної бази (для кешування)

Для реалізації проєкту обрано MongoDB.

Цей вибір зумовлений такими факторами:

1. MongoDB використовує бінарне представлення JSON, що дозволяє ефективно зберігати типи даних, специфічні для обчислень, що важливо для точних геометричних розрахунків стільниць.
2. Гнучкість індексації – це можливість створювати індекси по вкладених полях документів.

MongoDB – це документно-орієнтована NoSQL база даних, у якій інформація зберігається у вигляді BSON-документів. На відміну від реляційних БД, тут немає таблиць зі строгими схемами, але існує концепція колекцій (аналог таблиць) та документів (аналог записів).

Для інформаційної системи параметричного 3D-моделювання вибір MongoDB є обґрунтованим завдяки:

- динамічній природі параметрів 3D-моделей (які можуть змінюватися від виробу до виробу);
- зручності збереження вкладених структур (параметри геометрії, додаткові налаштування, історія змін);
- можливості масштабування та розподіленого зберігання.

База даних складається з таких основних колекцій:

- Users – Користувачі;
- Orders – Замовлення;
- Models – Параметричні моделі;
- Materials – Матеріали;
- Textures – Текстури.

Усі зв’язки реалізуються або через reference (зберігається ObjectId іншої колекції), або через embedded documents (вкладені структури).

Опис колекцій

1. Колекція Users зберігає інформацію про користувачів клієнтської частини системи.
2. Orders – це основна колекція для зберігання замовлень.
3. Колекція Models зберігає параметричні 3D-моделі, а також історію змін параметрів.
4. Колекція Materials містить каталог матеріалів для 3D-моделювання та розрахунку вартості.
5. Колекція Textures зберігає текстурні карти.

Під час створення нової програми часто одним із перших кроків є розробка її моделі даних. У реляційних базах даних, таких як MySQL, цей крок формалізується в процесі нормалізації, зосередженому на видаленні надлишковості з набору таблиць. MongoDB, на відміну від реляційних баз даних, зберігає свої дані в структурованих документах, а не у фіксованих таблицях, необхідних у реляційних базах даних. Наприклад, реляційні таблиці, зазвичай вимагають, щоб кожен перетин рядка та стовпця містив одне скалярне значення. Документи MongoDB BSON дозволяють створювати складнішу структуру, підтримуючи масиви значень (де кожен масив може складатися з кількох піддокументів) [17].

У MongoDB дані зберігаються в документах. Це означає, що там, де перша нормальна форма в реляційних базах даних вимагала, щоб кожен перетин рядка та стовпця містив рівно одне значення, MongoDB дозволяє зберігати масив значень, якщо ви цього хочете.

На щастя для нас, як розробників додатків, це відкриває деякі нові можливості в проєктуванні схем. Оскільки MongoDB може нативно кодувати такі багатозначні властивості, ми можемо отримати багато переваг продуктивності денормалізованої форми без супутніх труднощів з оновленням надлишкових даних. На жаль для нас, це також ускладнює наш процес проєктування схеми [17].

Документи в MongoDB моделюються за форматом JSON (JavaScript Object Notation), але насправді зберігаються у форматі BSON (Binary JSON). Коротко кажучи, це означає, що документ MongoDB є словником пар ключ-значення, де значення може бути одним із кількох типів:

- Примітивні типи JSON (наприклад, число, рядок, логічне значення)
- Примітивні типи BSON (наприклад, дата та час, ObjectId, UUID, регулярний вираз)
- Масиви значень
- Об’єкти, що складаються з пар ключ-значення
- Null [17].

Найбазовішою вимогою до системи електронної комерції є її функціональність оформлення замовлення. Окрім базової можливості поповнити кошик для покупок та оплатити, клієнти очікують, що онлайн-замовлення враховуватиме стан відсутності товару на складі, не дозволяючи їм розміщувати товари, якщо ці товари насправді недоступні.

Клієнти в інтернет-магазинах регулярно додають та видаляють товари зі свого «кошика для покупок», змінюють кількість кілька разів, залишають кошик у будь-який момент, а іноді й мають проблеми під час та після оформлення замовлення, які вимагають затримки або скасування замовлення. Ці дії ускладнюють ведення систем управління запасами та підрахунками, а також гарантують, що клієнти не можуть «купувати» товари, які недоступні під час покупок у вашому магазині.

Представлене тут рішення зберігає традиційну метафору кошика для покупок, але дозволяє неактивним кошикам для покупок старіти. Після того, як кошик для покупок був неактивним протягом певного періоду часу, всі товари в кошику знову потрапляють до доступного інвентарю, і кошик спорожніє. Різні стани, в яких може перебувати кошик для покупок, узагальнено [17] на рис. 2.17.

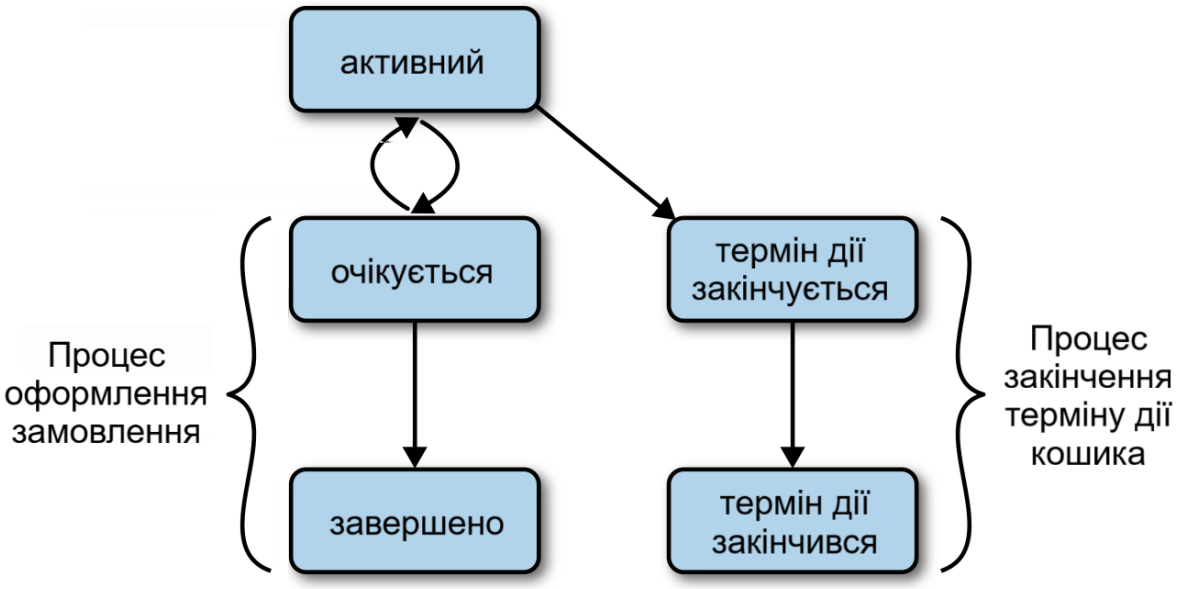


Рисунок 2.17 – Стан кошика для покупок

Ось пояснення кожного стану:

активний

У цьому стані користувач активний, і товари можна додавати або видаляти з кошика.

очікується

У цьому стані кошик оформлюється, але платіж ще не здійснено. Товари наразі не можна додавати або видаляти з кошика.

термін дії закінчується

У цьому стані кошик був неактивним занадто довго і «заблокований», поки його товари не повертаються до доступного інвентарю.

термін дії закінчився

У цьому стані кошик неактивний і недоступний. Якщо користувач повертається, необхідно створити новий кошик [17].

На основі обраної СУБД розроблено фізичну модель даних (рисунок 2.18). Вона визначає конкретні типи даних, валідацію полів та структуру колекцій, що будуть створені в MongoDB.

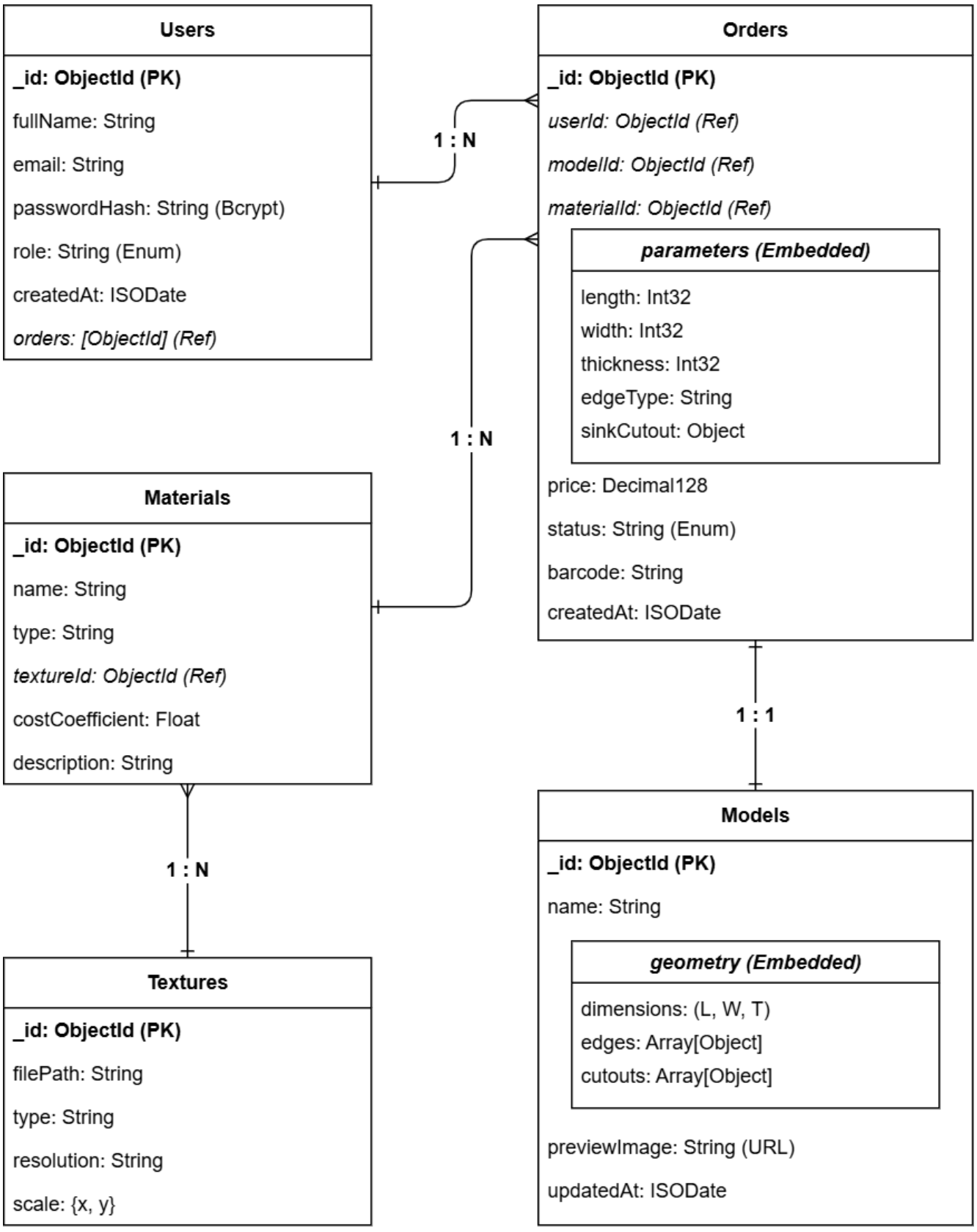


Рисунок 2.18 – Фізична модель бази даних

MongoDB не використовує традиційні зв’язки як у SQL (FOREIGN KEY), але підтримує два основні підходи:

1) reference (посилання на документ)

Приклад:

"userId": ObjectId("66ab..."),

"materialId": ObjectId("77cd...")

Переваги:

- дані не дублюються;
- можливе розмежування рівнів доступу;
- зручно для великих документів.

2) embedded documents (вкладені документи)

Приклад:

```
"parameters": {
  "length": 1200,
  "width": 600,
  "thickness": 20
}
```

Переваги:

- висока швидкість читання;
- параметри виробу завжди зберігаються в тому вигляді, в якому були на момент замовлення;
- зручно для структури, що рідко змінюється.

2.5 Висновки

У розділі було виконано детальне проєктування клієнтської, серверної та інформаційної частин системи параметричного 3D-моделювання індивідуальних будівельних виробів. Розглянуто функціональні модулі, визначено архітектуру системи. Проведена структуризація бази даних, визначено взаємодію компонентів через API. Надані рішення показують ефективну, гнучку й масштабовану реалізацію хмарного конфігуратора виробів.

3 СУЧАСНІ ТЕХНОЛОГІЇ РОЗРОБКИ ГРАФІЧНИХ СИСТЕМ

3.1 Загальні принципи параметричного 3D-моделювання у вебдодатках

Ми живемо у тривимірному світі. Люди рухаються, думають та сприймають у трьох вимірах. Значна частина наших медіа також є тривимірною, хоча зазвичай вона представлена на плоских екранах. Анімаційні фільми створюються з комп'ютерних тривимірних зображень. Онлайн-карти дозволяють нам віртуально досліджувати місце нашого призначення у тривимірному середовищі. Більшість відеоігор, незалежно від того, чи працюють вони на спеціалізованих консолях, чи на мобільних телефонах, відображаються у тривимірному форматі [10].

Тривимірна графіка майже така ж стара, як і сам комп'ютер, її коріння сягає 1960-х років. Вона використовувалася в таких сферах, як інженерія, освіта, навчання, архітектура, фінанси, продажі та маркетинг, ігри та розваги. Історично тривимірні програми спиралися на високоякісні комп'ютерні системи та дороге програмне забезпечення. Але це змінилося за останнє десятиліття. Апаратне забезпечення для 3D-обробки тепер постачається з кожним комп'ютером та мобільним пристроєм, а сучасний смартфон має більшу графічну потужність, ніж професійна робоча станція 15 років тому. Що ще важливіше, програмне забезпечення, необхідне для візуалізації 3D, тепер не тільки універсально доступне, але й безкоштовне. Воно називається веббраузером.

На рис. 3.1 показано уривок з програми «100,000 зірок» – браузерної 3D-симуляції польоту навколо наших зоряних сусідів у Чумацькому Шляху. За допомогою миші можна обертати галактичну площину та збільшувати масштаб зірки, що вас цікавить. Зірки представлені за допомогою візуалізацій, які приблизно відповідають їхній видимій зоряній величині та кольору. Кожна зірка позначена своєю загальною назвою; коли ви наводите курсор миші на мітку, вона виділяється. Натисніть на мітку, і з'явиться накладання, що відображає статтю Вікіпедії для цієї зірки. Натисніть на гіперпосилання в тексті накладання, і браузер запустить це посилання в новій вкладці. «100,000 зірок» – це приголомшливо створений

інтерактивний досвід із красивими рендерингами, пульсуючою анімацією, саундтреком та майстерно інтегрованим 2D-інтерфейсом користувача.

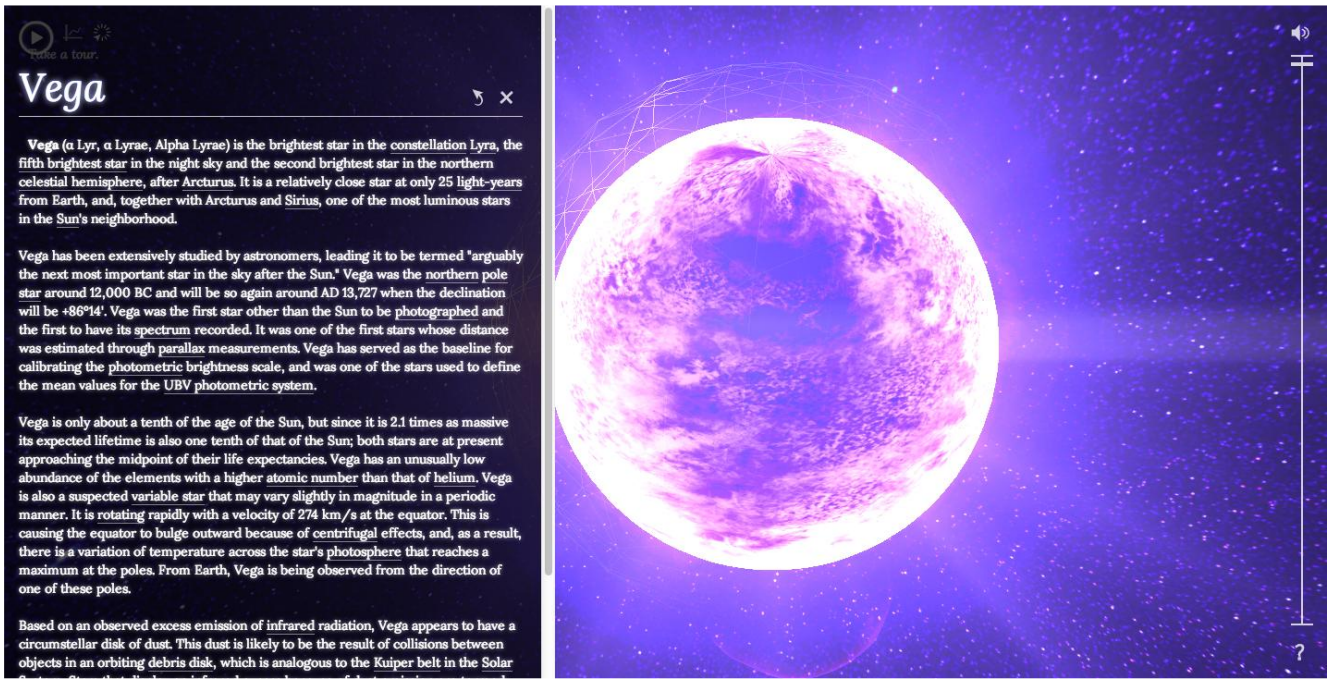


Рисунок 3.1 – Проєкт «100,000 зірок» від Google

Проєкт було створено як експеримент командою Google Data Arts для демонстрації великих можливостей браузера Chrome. Хоч застосунок є експериментальним, технології, що лежать в його основі, такими не є: він був створений з використанням функцій HTML5, доступних сьогодні в більшості браузерів. Галактика та зірки відображаються в режимі реального часу за допомогою WebGL, стандарту для апаратно-прискореної 3D-вебграфіки; мітки розміщуються відносно їхніх зірок за допомогою 3D-перетворень, які тепер доступні в CSS3; а накладання бездоганно поєднуються з 3D-контентом, оскільки браузери поєднують або складають усі елементи сторінки в єдине представлення.

Всього кілька років тому такий досвід, як «100,000 зірок», міг бути досягнутий лише в нативному клієнтському застосунку, що вимагає великого завантаження та встановлення, створеному розробниками, які використовують складні інструменти в процесі трудомісткої та дорогої розробки. Сьогодні його можна створити за допомогою браузера, безкоштовних інструментів з відкритим

кодом та стандартного стеку вебтехнологій. Більше того, ви можете миттєво отримати доступ до оновлень, просто перезавантаживши сторінку, завантажувати інформацію з будь-якого місця в Інтернеті за допомогою URL-адреси та натискати гіперпосилання з 3D-моделі, щоб отримати доступ до додаткової інформації [10].

HTML пройшов довгий шлях з часів статичних сторінок, форм та кнопки «Надіслати». На початку 2000-х років браузери запровадили багату взаємодію, дозволяючи динамічно змінювати частини сторінки за допомогою методів Ajax. Однак способи зміни сторінок за допомогою Ajax були обмежені графічними можливостями HTML та CSS. Якщо розробник хотів вийти за ці межі, йому доводилося використовувати медіа-плагіни, такі як Flash та QuickTime.

Це було фактично статус-кво протягом 2000-х років, але за останні кілька років все змінилося. Кілька досягнень браузерів, що розроблялися в цей період, об’єдналися в HTML5. З HTML5 веббраузер став платформою, здатною запускати складні програми, які за функціями та продуктивністю конкурують з нативним кодом. HTML5 являє собою масштабне оновлення стандарту HTML, включаючи очищення синтаксису, нові функції мови JavaScript та інтерфейси прикладного програмування (API), мобільні можливості та революційну підтримку мультимедіа. Центральним елементом платформи HTML5 є набір передових графічних технологій:

- WebGL для апаратно-прискореного 3D-рендерингу за допомогою JavaScript. Заснований на перевіреному часом графічному API OpenGL, WebGL є стандартом, який підтримується майже всіма веббраузерами на настільних комп’ютерах, а також зростаючою кількістю мобільних браузерів.
- 3D-перетворення, переходи та користувацькі фільтри CSS3 для розширених ефектів сторінок. CSS розвинувся протягом останніх кількох років, включивши апаратно-прискорений 3D-рендеринг та функції анімації, доступні через мову таблиць стилів.
- Елемент Canvas та його API контексту 2D-малювання. Універсально підтримуваний у браузерах, цей JavaScript API дозволяє розробникам малювати довільну графіку на поверхні елемента DOM. Хоча Canvas є 2D

API, за допомогою додаткових бібліотек JavaScript його можна використовувати для візуалізації 3D-ефектів, що забезпечує альтернативу для платформ, де WebGL або CSS3 3D не підтримуються.

Кожна з цих функцій має свої сильні та слабкі сторони, а також технічні компроміси, і кожна відіграє свою роль у створенні інтерактивного та візуально привабливого 3D-досвіду [10].

3D-комп'ютерна графіка, яку іноді називають CGI, 3D-CGI або тривимірною комп'ютерною графікою, – це графіка, яка використовує тривимірне представлення геометричних даних (часто декартових), що зберігаються в комп'ютері, для виконання обчислень та візуалізації цифрових зображень, зазвичай 2D-зображень, але іноді 3D-зображень. Отримані зображення можна зберігати для подальшого перегляду (можливо, як анімацію) або відображати в режимі реального часу [11]. Давайте розберемо це на складові: 1) дані представлені в 3D-системі координат; 2) вони зрештою малюються (візуалізуються) як 2D-зображення (наприклад, на моніторі вашого комп'ютера); та 3) вони можуть відображатися в режимі реального часу: коли 3D-дані змінюються під час анімації або маніпуляцій з боку користувача, візуалізоване зображення оновлюється без помітної затримки. Ця остання частина є ключовою для створення інтерактивних програм. Фактично, вона настільки важлива, що породила багатомільярдну індустрію, присвячену спеціалізованому графічному обладнанню, що підтримує 3D-рендеринг у реальному часі, з кількома компаніями, такими як NVIDIA, ATI та Qualcomm, які лідирують у цьому напрямку [10].

У двовимірних декартових системах координат, таких як координати вікна документа HTML, існують значення *x* та *y*. Ці двовимірні координати визначають, де на сторінці розташовані теги , або де віртуальне перо чи пензель малює в елементі HTML Canvas. Аналогічно, 3D-малювання відбувається (що не дивно) у тривимірній системі координат, де додаткова координата *z* описує глибину (тобто, наскільки далеко на екран або за його межі малюється об'єкт). Системи координат розташовані як показано на рис. 3.2, де *x* розміщується горизонтально (зліва направо), *y* – вертикально, а додатна *z* виходить за межі екрана. WebGL визначає

додатну *y* як спрямовану знизу до верху вікна, тоді як 2D Canvas API та CSS-перетворення визначають додатну *y* як спрямовану вниз [10].

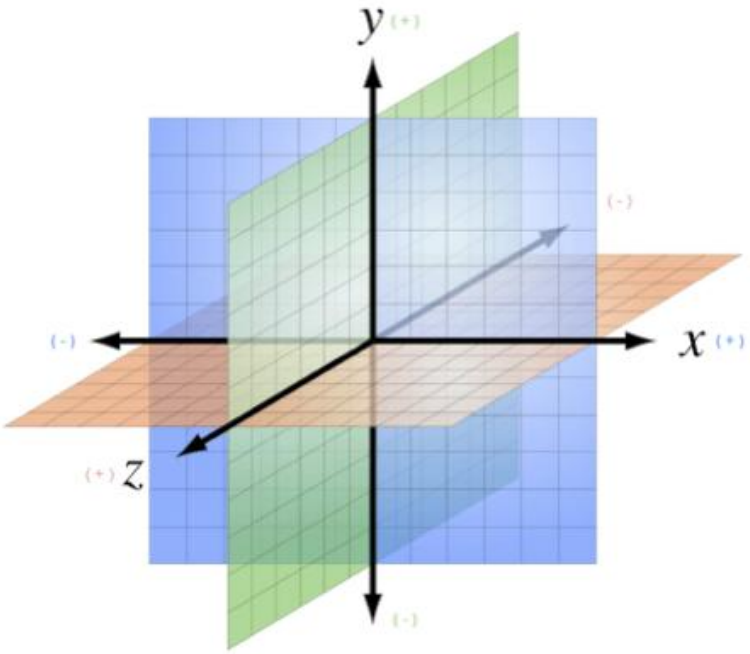


Рисунок 3.2 – 3D-система координат

Параметричне 3D-моделювання – це підхід, який дає змогу будувати тривимірні об’єкти на основі набору параметрів – змінних, що описують геометрію, розміри, матеріали чи логіку побудови. У традиційних CAD-системах цей метод використовується вже понад два десятиліття, проте лише останніми роками він активно впроваджується у вебтехнологіях завдяки зростанню обчислювальних можливостей браузерів і розвитку API (інтерфейсу прикладного програмування) для графічного рендерингу.

Суть параметричного підходу у вебсередовищі полягає в тому, що 3D-модель формується на основі взаємозв’язків між параметрами, які користувач може змінювати через інтерфейс. Наприклад, під час зміни довжини або товщини стільниці автоматично оновлюється геометрія моделі без ручного втручання розробника. Такий підхід особливо ефективний у галузі виробництва індивідуальних виробів – меблів, сантехнічних виробів, архітектурних елементів, де потрібно швидко отримати візуалізацію за заданими параметрами.

3.2 Огляд сучасних бібліотек і фреймворків для вебвізуалізації

Тенденцією останніх років у розробці програмного забезпечення стало те, що застосунки все частіше переходять від автономних програм до вебзастосунків. Однак більша частина графічного контенту представлена у 2D. Традиційно всі вимогливі до графіки застосунки були автономними, але останнім часом, після появи WebGL, у веббраузерах з'явилося багато графічно більш вражаючих застосунків та ігор. Такий зсув робить багато програм більш легкодоступними та усуває необхідність фактичного встановлення застосунку [12].

WebGL – це технологія, яка дозволяє малювати, відображати та взаємодіяти зі складною інтерактивною тривимірною комп'ютерною графікою («3D-графікою») прямо з веббраузерів [26]. Недолік WebGL полягає в його низькорівневості – для створення навіть простої сцени розробнику потрібно писати десятки рядків GLSL-коду, описуючи шейдери, буфери, матриці та вектори [8]. Саме тому для WebGL створено високорівневі бібліотеки.

Three.js – це найпопулярніший фреймворк для роботи з WebGL. Він забезпечує простий API для побудови сцен, освітлення, текстурування та анімацій [12]. Розробник може створювати об'єкти з геометрії (BoxGeometry, SphereGeometry тощо) і змінювати їх параметри динамічно через JavaScript [9]. Велика перевага Three.js – наявність системи експорту з інших 3D-редакторів (Blender, Rhino, Revit), що дозволяє використовувати його для візуалізації архітектурних виробів у браузері.

Three.js постачається з багатьма попередньо створеними типами геометрії, які представляють поширені форми. Це включає прості тверді тіла, такі як куби, сфери та циліндри; складніші параметричні форми, такі як екструзії та форми на основі шляхів, тори та вузли; плоскі 2D-фігури, що відображаються у 3D-просторі, такі як кола, квадрати та кільця; і навіть 3D екструдований текст, згенерований з текстових рядків. Three.js також підтримує малювання 3D-точок та ліній. Ви можете легко створити більшість цих об'єктів за допомогою однорядкового

конструктора, хоча деякі вимагають трохи складніших параметрів та трохи більше коду [10]. Попередньо створена геометрія показана на рис. 3.3.

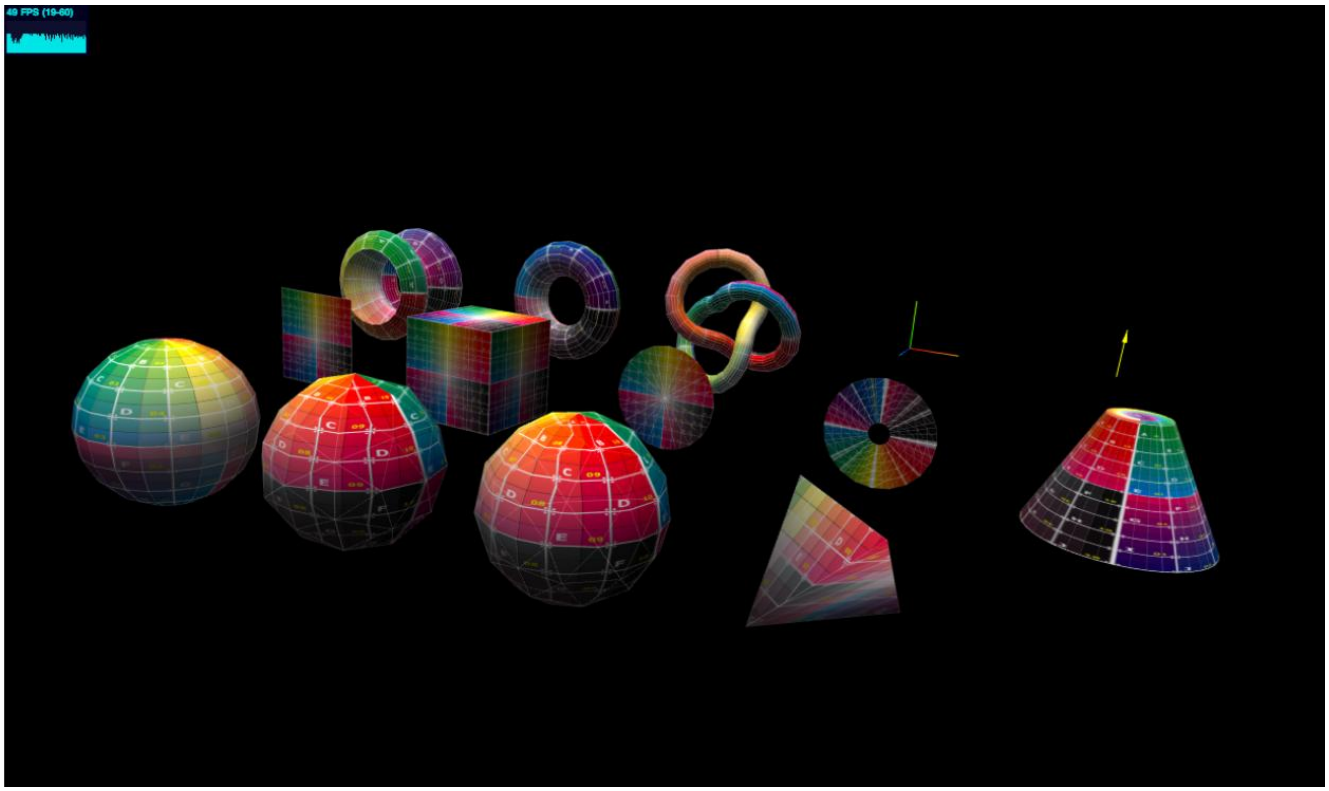


Рисунок 3.3 – Демонстрація вбудованої геометрії Three.js

Babylon.js – сучасна альтернатива Three.js, яка підтримує фізично коректні рендеринг (PBR-materials), світлові ефекти та інтеграцію з WebXR (доповнена і віртуальна реальність). За результатами дослідженнь, Babylon.js має вищу ефективність під час рендерингу сцен з великою кількістю полігонів, тоді як Three.js зручніший у використанні для розробників.

Verge3D – це рендерер реального часу та набір інструментів, що використовуються для створення інтерактивних 3D-досвідів, що працюють на вебсайтах [28]. Це комерційна платформа, орієнтована на створення вебконфігураторів товарів з інтерактивними UI-елементами та інтеграцією з CMS (системою керування вмістом).

PlayCanvas – це рушій 3D-ігор/інтерактивний рушій 3D-додатків з відкритим вихідним кодом, що працює разом із власною хмарною платформою для створення,

яка дозволяє одночасне редагування з кількох комп’ютерів через інтерфейс на основі браузера. Він працює в сучасних браузерах, що підтримують WebGL, включаючи Mozilla Firefox та Google Chrome. Рушій здатний до моделювання фізики твердого тіла, обробки тривимірного аудіо та 3D-анімації [29].

У роботі фінського університету прикладних наук, що знаходиться в місті Турку, йде мова про створення концептуального зразка вебзастосунку для дизайнера кімнат [12] для компанії Progman Oy (зараз MagiCAD Group), що створює програмне забезпечення для проєктування інженерних систем будівель. Застосунок міг би бути використаний для дослідження життєздатності застосунку в проєктуванні кімнат та потреби в такому продукті на ринку [12]. Однією з причин, чому Three JS була дуже придатною для цього проєкту, було те, що вона призначена просто для представлення 3D-контенту. Багато бібліотек, які мають схожі функції, є повноцінними ігровими рушіями. Three JS містить лише основні функції для відображення 3D-контенту і не має елементів ігрового рушія. Це було бажано для проєкту, оскільки бібліотека повинна була бути якомога легшою. Three JS є однією з найстаріших і найпопулярніших бібліотек, і тому вона має відносно хорошу документацію та є досить зрілою. Важливою вимогою для цього проєкту була наявність хороших функцій імпорту та експорту. Three JS підтримує імпорт різних типів файлів для моделей, серед яких важливим був імпорт JSON. Розробники в компанії вже були дуже задоволені Three JS, що заохочувало її використання в цьому проєкті. Деякі з розглянутих альтернатив – це Babylon JS і Unity. Обидва вони є ігровими рушіями. Unity офіційно підтримує WebGL з версії 5.3, але була занадто важкою для цього проєкту [12].

Three JS пропонує багато різних завантажувачів, які підтримують такі формати, як JSON, Obj, зображення та сцени. 3D-моделі можна імпортувати за допомогою завантажувача та файлу JSON або Obj. Імпорт моделей із популярних інструментів 3D-моделювання, таких як Blender або 3DS Max, можливий завдяки їхнім можливостям експорту в JSON (у Blender для цього потрібен плагін, створений спільнотою). Three JS також може експортувати свої геометрії, сцени, матеріали та інші об’єкти в різні формати, як-от JSON [12].

У цьому порівнянні розглядаються три різні технології, які можна використовувати для створення WebGL-контенту для браузерів: Three JS, Unity та Babylon JS. Unity – один із найпопулярніших ігрових рушіїв на ринку, який також може створювати ігри в WebGL-додатках. Babylon JS – це 3D-рушій з відкритим кодом на основі WebGL, спочатку створений деякими розробниками Microsoft.

Таблиця 3.1 – Порівняння фреймворків

Фреймворк	Three JS	Unity	Babylon JS
Мова	JavaScript	C#, UnityScript	JavaScript
Ліцензія	MIT	Власницька	Ліцензія Apache 2.0
Мобільна підтримка	Так	Ні	Так
Середовище розробки	Без обмежень	Unity	Без обмежень
Інтегрована фізика	Ні	PhysX	Cannon JS, Oimo JS, Energy JS
Імпорт	JSON, OBJ, FBX	OBJ, FBX	OBJ, FBX, Babylon, STL, JSON
Експорт	JSON, OBJ	Ні	Формати, підтримувані Blender та 3dsMax
Підтримка VR	Так	Так	Так

Проєкт був виконаний лише за кілька тижнів, що підтвердило здатність Three JS швидко створювати результати. Фреймворк виявився дуже ефективним для такого роду прототипування. Хоча Three JS має менше обговорень і прикладів, ніж Unity, її вебсайт надає хорошу бібліотеку прикладів (хоча більшість коду не коментується) [12].

3.3 API та серверні технології для інтеграції параметричного моделювання

У статті спеціального випуску «Цифрова трансформація в будівельній галузі: останні досягнення та перспективи» міжнародного журналу «Buildings» зазначається, що конфігуратори нещодавно стали важливими інструментами в будівельній галузі, що дозволяють будівельникам пропонувати широкий спектр налаштовуваних проєктів [13]. Через значні труднощі в інтеграції інформації між постачальниками будівельних матеріалів та клієнтами, існуючим системам конфігураторів часто бракує важливої інформації про зручність використання та ланцюг постачання, що створює перешкоди для ширшого впровадження серед житлових громад, особливо в забудові односімейних будинків, яка вимагає високого ступеня налаштування. Щоб вирішити цю проблему в ланцюжку постачання проєктування та будівництва, це дослідження представляє легку хмарну методологію конфігурації модульного будинку як надійне уніфіковане платформне рішення для інтеграції параметричних варіантів проєктування із сертифікованою бібліотекою комплектів деталей для відповідності місцевим нормам проєктування. Прототип конфігуратора, розроблений за цією платформою, бездоганно інтегрує важливу інформацію про проєктування та ланцюг постачання, використовуючи (1) генеративний дизайн планування з попередньо затвердженими кресленнями, (2) систему рекомендацій на основі знань для зв’язку процесу проєктування із сертифікованими каталогами матеріалів та (3) зручний вебінтерфейс для представлення можливих проєктів. Впровадження проєкту односімейного будинку, що відповідає будівельним нормам, у провінції Британська Колумбія в Канаді ілюструє переваги запропонованих функцій конфігуратора та ефективної інтеграції даних постачальників. Легкий та автоматизований, запропонований конфігуратор має значний потенціал для масштабування та впровадження в різних спільнотах [13].

Дослідники та провідні фахівці з інновацій у будівництві досягли прогресу в розробці хмарних конфігураторів як адаптивних платформ проєктування для

управління запитами клієнтів, включаючи інтуїтивно зрозумілий інтерфейс для вибору будівельних модулів, що легко виготовляються, які називаються комплектами деталей, що використовуються для складання готових виробів з одночасним налаштуванням показників ефективності будівель. Для будівельної галузі, складність ланцюга поставок якої є поширеною, впровадження цих рішень конфігуратора також може покращити потік даних та співпрацю між зацікавленими сторонами, щоб зменшити невизначеності в процесах ланцюга поставок. Деякі виробники модульних будинків пропонують вебінструменти, подібні до конфігураторів, такі як Build Prefab, що дозволяє клієнтам вибирати комбінації комплектів деталей з каталогів, але обмежує налаштування вибором предметів інтер'єру на основі обмеженої кількості попередньо намальованих креслень [13].

Щоб зосередити потреби в налаштуванні між спільнотами та вирішити технічні проблеми, у цьому дослідженні пропонується легка хмарна платформа конфігуратора, яка висвітлює три основні сфери внеску знань: (1) швидке налаштування макета на основі існуючих генеративних алгоритмів та попередньо затвердженого плану будівництва, (2) ефективна інтеграція каталогів сертифікованих матеріалів (у формі бібліотек об'єктів BIM) та автоматичне зіставлення через систему рекомендацій на основі знань, та (3) розширене залучення кінцевих користувачів конфігуратора через інтерактивний вебінтерфейс проєктування. Запропонований конфігуратор реалізує переваги масового налаштування для користувачів та інтегрує міжфазну інформацію з генеративним проєктуванням макета на етапі планування та інтелектуальний механізм рекомендацій на основі знань для модульних компонентів на етапі закупівлі [13].

Інструменти, системи та платформи конфігураторів сягають корінням у САПР (автоматизоване проєктування) для виробничих конструкцій виробів (наприклад, транспортних засобів та меблів) та еволюціонують у програми на основі інформаційного моделювання будівель (BIM) після адаптації до будівельної галузі для включення параметричного моделювання та проєктування будівель на основі правил. Ранні конфігуратори зосереджувалися на простих конфігураціях виробів, але досягнення в обчислювальній потужності та алгоритмах оптимізації

дозволили створити складніші конфігуратори, здатні обробляти складні правила проєктування та обмеження для масового налаштування виробів. Конфігуратор зазвичай розробляється для забезпечення інтерфейсу для клієнтів або фахівців, які обробляють запити клієнтів, щодо вибору простих у виготовленні модулів, відомих як комплекти деталей, для складання повного виробу та налаштування показників продуктивності. Цао та ін. [14] запропонували міжфазний конфігуратор, який охоплює фази планування, проєктування та обрання виробу, щоб технологічність комплекту деталей, що використовується в конкретному виробі, могла бути оцінена на етапі проєктування, щоб скоротити час виконання та переробки. Така структура демонструє здатність конфігуратора сприяти багатосторонній співпраці та обміну інформацією за допомогою параметричної моделі та пов'язаних з нею параметрів, що виходить за рамки його добре відомої ролі у забезпеченні налаштування. Незважаючи на поширеність різних методологій конфігуратора, що використовуються для обслуговування фахівців з дизайну та архітектури в комерційних забудовах, потенціал конфігураторів для повноцінної моделі окремого житлового будівництва був недостатньо вивчений. Технологічні та знаннєві передумови для комерційних конфігураторів на ринку обмежують прямий доступ клієнтів на етапі проектування. Деякі виробники модульних будинків надають вебмайстри, подібні до конфігуратора, для вибору клієнтами комбінації комплектів деталей з їхнього каталогу продукції. Таку систему можна підключити до подальших дій, таких як автоматизована перевірка запасів або генерація інформації про котирування. Незважаючи на це, клієнти можуть налаштовувати продукти лише на основі одного з попередньо розроблених планів, замість того, щоб налаштовувати макет, який відповідає їхнім земельним ділянкам та потребам у проживанні [13].

Щоб задовольнити потребу у створенні 3D-моделей на основі правил та змістовних параметрів замість растрових зображень, об'єкти BIM розглядаються як потенційний спосіб створення цифрового центру, який інтегрує всю інформацію про проєктування на основі правил щодо налаштовуваних властивостей та геометрії будівлі для координації між різними зацікавленими сторонами та

протягом усього життєвого циклу будівлі. Американська асоціація управління будівництвом підкреслила, що успішне впровадження BIM може покращити комунікацію та співпрацю між учасниками проєкту, а також підвищити дотримання специфікацій та стандартів з точки зору масової кастомізації. Такі особливості зробили багатопараметризований BIM ідеальним для включення до конфігураторів як об'єктів для вибору [13].

У цьому дослідженні пропонується методологія конфігуратора, яка реалізує сучасний інструмент генеративного проєктування та систему рекомендацій на основі знань на єдиній платформі з вебінтерфейсом користувача [13].

Весь прототип був розроблений для виконання наступних нефункціональних вимог: (1) сумісність – можливість обміну даними з іншим програмним забезпеченням та системами, наприклад, Autodesk Revit; (2) легкість – можливість роботи без спеціального обладнання або встановлення програмного забезпечення локально; (3) гнучкість – можливість розміщення додаткових модулів, наприклад, для аналізу життєвого циклу; та (4) зручність використання – наявність зрозумілих інтерфейсів для непрофесійних користувачів.

Процес заповнення баз даних полягає у заповненні початковими даними до того, як будь-яка бізнес-логіка буде реалізована в кодах. Оскільки актуальні та відповідні коду набори параметрів для попередньо затверджених планів та комплектів деталей закладають основу як для процесів створення макета, так і для процесів рекомендацій щодо закупівель, ми зробили це окремим кроком перед реалізацією прототипу. Щоб досягти інтеграції каталогу продукції від виробників у режимі реального часу, ми отримали ключ API для всієї платформи до BIMStore [15], репозиторію BIM, що містить комплекти деталей, доступні на різних ринках. Набори параметрів комплектів деталей у кожній категорії були отримані з BIMStore за допомогою викликів API та завантажені в MongoDB за допомогою автоматизованого скрипта. Локальна копія наборів параметрів періодично синхронізувалася з BIMStore для відстеження нових або видалених елементів. Враховуючи складність параметрів для будь-якого окремого об'єкта BIM, під час заповнення бази даних використовувалося секціонування колекцій [17]. Набори

параметрів зберігалися в 3 колекціях MongoDB: Products, ProductParamSets та Manufacturers. Колекція Products (Продукти) містить фіксовану схему, що містить попередньо визначені поля метаданих лише про продукт. Натомість, специфічний для продукту або категорії вміст, що складається зі змінних полів (наприклад, тип обробки та рейтинги вогнестійкості), зберігається в полі Data (Дані) документів у колекції ProductParamSets. Один продукт можна пов'язати з кількома документами в ProductParamSets, кожен з яких представляє варіант. Неструктуровані файли, такі як моделі Revit та документи сертифікації продукції, не включаються до етапу заповнення. Натомість, для вказівки набору параметрів на файлові ресурси використовується URL-адреса або UUID платформи [13].

Покроковий огляд взаємодії користувача з конфігуратором та пов'язаних з ним інформаційних потоків показано [13] на рис. 3.4.

Рисунок 3.4 – Шлях користувача із запропонованим конфігуратором

Як і інші вебінструменти підвищення продуктивності у виробничій галузі, реалізація системи відповідає трирівневій архітектурі, а саме поєднанню рівня

презентації, бізнес-рівня (рівня додатків) та рівня даних. Щоб задовольнити вимоги до сумісності, зручності використання та масштабованості, необхідно визначити відповідні технічні специфікації та припущення щодо середовища виконання для кожного рівня. Очікується, що користувач використовуватиме браузер на основі WebKit, IE 11+ або Firefox, з операційною системою MacOS або Windows 7+. Бізнес-рівень розгорнуто на сервері AWS EC2 (екземпляри T3) з 32 ГБ оперативної пам'яті та 8 віртуальними процесорами. MongoDB Atlas, повністю керований сервіс зберігання даних на базі MongoDB, був обраний як рішення рівня даних, у поєднанні з різними зовнішніми джерелами даних [13].

Для реалізації вимоги щодо можливості підключення, як це запропоновано в системних вимогах, рівень презентації повинен мати компонентну архітектуру. Тому React.js було обрано як вебфреймворк. Однією з ключових переваг React.js є його реалізація віртуальної DOM (моделі об'єктів документа). Завдяки легкому представленню фактичної DOM в пам'яті, React.js може ефективно відстежувати зміни та оновлювати лише необхідні компоненти, що призводить до швидшого рендерингу та покращення продуктивності. Крім того, React.js сприяє декларативній парадигмі програмування, де розробники описують, як має виглядати інтерфейс користувача, на основі стану програми, а не імперативно маніпулюють DOM безпосередньо. Такий підхід призводить до більш передбачуваного та зручного в обслуговуванні коду, а також полегшує налагодження та тестування. Розділяючи складні інтерфейси користувача на модулі з власними станами, кодова база стає більш багаторазовою та простішою в обслуговуванні. Це дозволяє програмістам впроваджувати нові функції, які бачить користувач, та оновлювати існуючі, не впливаючи на інші компоненти [13].

Компоненти бізнес-рівня, включаючи всі обробники HTTP-запитів, систему рекомендацій та модулі підключення до генератора проєктів, розміщені на хмарному сервері AWS EC2. Обробники запитів реалізовані за допомогою Express.js у середовищі виконання Node.js 16 для досягнення модульності сервісів. Частина системи рекомендацій та її утиліти написані на Python 3 через різноманітність бібліотек NLP, що переважають у її екосистемі. Використовуючи

цю екосистему, конфігуратор може використовувати передові методи NLP для інтерпретації введених користувачем даних, вилучення значущої інформації з каталогів матеріалів та створення персоналізованих рекомендацій. Такий підхід покращує взаємодію з користувачем, оптимізує процеси прийняття рішень і, зрештою, сприяє ефективності конфігуратора у спрощенні проєктування та закупівлі житлових будівель [13].

Рівень даних має першорядне значення в нашій архітектурі. Через міркування щодо обсягу сховища та часу обробки запитів було використано систему керування базами даних No-SQL на основі документів JSON – MongoDB. Неструктуровані дані, такі як документація на продукцію, готові комплекти деталей, карти ділянок та муніципальні нормативні акти у форматах PDF або електронних таблиць, зберігалися в об’єктно-орієнтованому сервісі зберігання даних (Amazon S3 Bucket). Сторонні джерела даних (наприклад, репозиторії об’єктів BIM, сервіс геолокації та портали котирувань виробників) були інтегровані в рівень даних через зовнішні виклики API.

Стиль проєктування системи передачі репрезентативних станів (REST) був реалізований у всій архітектурі конфігуратора. Кожен об’єкт даних (користувач, проєкт, комплект деталей, модель та неструктурований документ) на платформі індексується універсальним унікальним ідентифікатором (UUID), а вся передача даних здійснюється за допомогою методів HTTP. Окрім переваг масштабованості, користувачі в організаціях (наприклад, постачальники громадського житла) можуть використовувати кінцеві точки API сервера конфігуратора для створення розширених та налаштованих програмних функцій, включаючи автоматизоване проєктування та закупівлю житла у власні бізнес-процедури. Наприклад, постачальники громадського житла можуть впроваджувати автоматизовану звітність для підвищення підзвітності та дотримання внутрішніх політик [13].

Отже, для реалізації параметричного 3D-конфігуратора у вебсередовищі потрібна взаємодія клієнтської та серверної частин. Для клієнтської частини у браузері використовується JavaScript/TypeScript у поєднанні з React.js або Vue.js для створення інтерфейсу, а для серверної – Node.js чи Django [16]. Зв’язок між

ними здійснюється через REST API або WebSockets для миттєвого оновлення параметрів моделі без перезавантаження сторінки.

База даних (MySQL, PostgreSQL або MongoDB) використовується для збереження параметрів замовлення, матеріалів, геометричних значень та інформації про користувачів [16].

Інтеграція з платіжними системами (LiqPay, Stripe, PayPal) дозволяє реалізувати повноцінний цикл замовлення – від створення виробу до оплати та збереження даних у базі.

3.4 Порівняння методів реалізації параметричних систем у вебзастосунках

Ефективність параметричних вебсистем визначається триєдиним критерієм: гнучкість моделювання, інтерактивність користувацького інтерфейсу та продуктивність візуалізації. Порівняння методів наведено в табл. 3.2.

Таблиця 3.2 – Порівняння методів реалізації інформаційних систем

Метод або технологія	Рівень інтерактивності	Підтримка параметрів	Продуктивність	Складність розробки
WebGL (чистий API)	Високий (залежить від оптимізації)	Повна, ручна	Висока	Висока
Three.js	Дуже високий	Повна (через JS-об’єкти)	Висока	Середня
Babylon.js	Високий, VR/AR	Повна	Дуже висока	Середня
Verge3D	Середній	Часткова (через інтерфейс)	Висока	Низька
PlayCanvas	Високий	Повна	Дуже висока	Середня

Для виконання цього проєкту обрано бібліотеку Three.js, оскільки це звичайний механізм 3D-графіки, а не повноцінний ігровий рушій, вона має широкі можливості експорту та імпорту [12]. Three.js визначено придатною для такого роду

прототипування, оскільки її основи досить прості для швидкого вивчення та здатні швидко давати результати [12].

3.5 Висновки

Було розглянуто сучасні технології та методи, що застосовуються для параметричного 3D-моделювання у вебсередовищі. Аналіз показав, що розвиток стандарту WebGL та поява високорівневих фреймворків створили передумови для реалізації складних вебдодатків із підтримкою параметричного моделювання в реальному часі. Інтеграція таких рішень із сучасними серверними технологіями та базами даних дозволяє побудувати інформаційну систему, що поєднує візуалізацію, управління замовленнями та підтримку користувацьких налаштувань.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

4.1 Обґрунтування вибору технологічного стеку

Бібліотека Three.js дозволяє відтворювати 3D-графіку у браузері за допомогою WebGL, забезпечує побудову геометричних моделей, текстурування, освітлення та рендеринг у реальному часі. Three.js підтримує створення параметричних mesh-об’єктів, що робить її оптимальною для задачі побудови стільниць. Mesh це набір вершин, ребер і граней, які формують каркас тривимірного об’єкта. Вершини відповідають за координати у просторі, ребра з’єднують вершини, а грані утворюють поверхню. Найчастіше в роботі використовують трикутники або чотирикутники, адже вони добре підходять як для художніх задач, так і для інженерних симуляцій. Вершини – точки у тривимірному просторі з координатами. Ребра – відрізки, що з’єднують вершини. Грані – полігони, утворені ребрами [18]. Для завдання параметричного моделювання Three.js підходить завдяки гнучкості геометрій та інтеграції з бібліотеками CSG.

Фреймворк React.js надає такі переваги:

- багатокомпонентну структуру інтерфейсу;
- можливість швидкого оновлення моделі без перезавантаження сторінки;
- інтеграцію з REST API та WebGL-компонентами.

Серверна платформа Node.js дозволяє реалізувати:

- REST API для клієнтської частини;
- обробку параметрів моделі;
- авторизацію, управління замовленнями;
- логіку збереження даних.

Документо-орієнтований підхід MongoDB дає можливість зберігати:

- параметри виробів у вигляді структурованих JSON-документів;
- дані користувачів;
- історію моделювань та замовлень;

– файли текстур та довідкові дані.

Вибір REST-стилю взаємодії API пояснюється:

– простотою реалізації;
– можливістю використання на будь-яких клієнтських платформах;
– підтримкою кешування;
– масштабованістю.

Таким чином, для реалізації інформаційної системи параметричного 3D-моделювання індивідуальних виробів доцільно застосувати стек технологій Three.js, React.js, Node.js та MongoDB, який забезпечує ефективність, масштабованість і зручність у подальшій експлуатації.

Система складається з наступних функціональних рівнів (рисунок 4.1):

1. Рівень презентації – це клієнтська частина додатку (frontend), що виконується безпосередньо у браузері користувача:
    - Інтерфейс користувача (React.js) відповідає за відображення елементів керування, форм введення параметрів стільниці (довжина, ширина, тип вирізу) та валідацію вхідних даних на стороні клієнта;
    - Графічний рушій (Three.js) – це інтегрований у React-середовище модуль, який відповідає за рендеринг параметричної 3D-моделі у реальному часі. Він отримує дані про розміри з інтерфейсу і динамічно перебудовує геометрію виробу на HTML5 Canvas;
2. Рівень бізнес-логіки – це серверна частина (backend), побудована на платформі Node.js. Цей рівень виступає ядром системи, що обробляє запити від клієнта:
    - Сервер REST API приймає HTTP-запити від клієнтської частини, маршрутизує їх та відправляє відповіді;
    - Модуль логіки моделювання виконує перевірку параметрів на технологічні обмеження (наприклад, чи можливе виготовлення стільниці заданих розмірів) та готує дані для збереження;

3. Рівень даних забезпечує надійне зберігання інформації. Використання документо-орієнтованої NoSQL бази даних MongoDB дозволяє зберігати параметричні моделі стільниць у гнучкому форматі, ідентичному до JSON-об’єктів, що використовуються на рівнях бізнес-логіки та презентації.

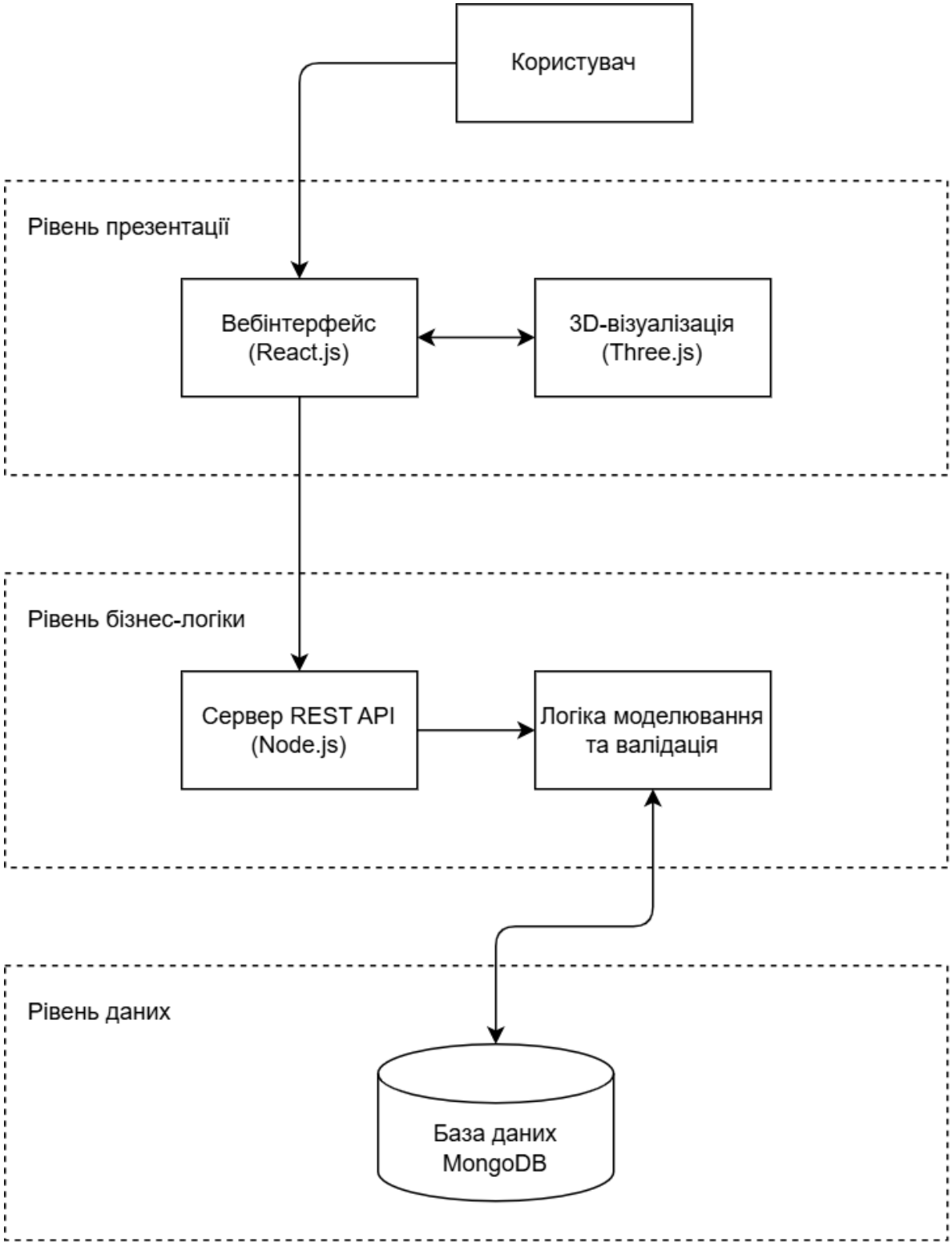


Рисунок 4.1 – Загальна схема реалізації архітектури програмного забезпечення

4.2 Вибір інструментів розробки клієнтської частини системи

JavaScript (JS) – динамічна, об’єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв вебсторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд вебсторінки [19]. Вибір мови програмування JavaScript обумовлений її популярністю у веброзробці, вона дозволяє створювати кросплатформні додатки, які працюють у будь-якому сучасному браузері без встановлення додаткових плагінів.

Фреймворк JS – це набір зумовлених інструментів, функцій і структур, призначених для розробки вебдодатків на JavaScript [20]. Він надає програмістам готову основу, яка допомагає спростити і структурувати процес створення складних додатків. Його мета – надати розробникам зручний набір інструментів для створення ефективних і масштабованих вебдодатків. Фреймворк js забезпечують структуру й організацію коду, а також надають готові рішення для завдань, які часто зустрічаються, таких як управління станом, маршрутизація, обробка подій і взаємодія із сервером [20]. JavaScript фреймворки надають розробникам набір інструментів і абстракцій, які дають змогу зосередитися на створенні функціональних і ефективних вебзастосунків, мінімізуючи витрати на повторну розробку і покращуючи процес розробки загалом [20].

React

Цей фреймворк розроблений Facebook і є компонентним фреймворком. Це означає, що ви можете створювати перевикористовувані компоненти, які складають ваш інтерфейс. Він відомий своєю віртуальною DOM (абстракція реального DOM – Document Object Model, що використовується для ефективного оновлення інтерфейсу вебдодатків без перемальовування всієї сторінки) та ефективним механізмом оновлення, що робить його чудовим вибором для створення складних та інтерактивних користувацьких інтерфейсів. Вважається, що це –

найпопулярніший фреймворк javascript. Він використовується безліччю великих компаній і широко застосовується в індустрії веброзробки [20].

Angular

Розроблений Google, він пропонує повноцінне рішення для розробки вебдодатків. Надає безліч функцій, включно з керуванням станом, маршрутизацією, валідацією форм і багато іншого. Крім того, використовує власну синтаксичну конструкцію, відому як «директиви», для створення компонентів і управління інтерфейсом. Цей фреймворк ідеально підходить для розроблення складних застосунків, що вимагають масштабованості та широкого функціоналу [20].

Vue.js

Вважається досить простим у використанні фреймворком. Він має простий і зрозумілий синтаксис, що робить його привабливим для розробників-початківців. Крім того, пропонує двосторонню прив'язку даних, управління станом і безліч готових компонентів. Він ідеально підходить для створення інтерфейсів різної складності, як невеликих проектів, так і більших додатків [20].

React.js обрано з огляду на необхідність створення складного інтерактивного інтерфейсу, де стан системи (параметри стільниці) постійно змінюється. React використовує концепцію Virtual DOM, що дозволяє мінімізувати кількість звернень до реального DOM дерева браузера, оновлюючи лише ті частини інтерфейсу, які змінилися.

React поділяється на два основні API:

- API компонентів React – це частини сторінки, які фактично відображаються React DOM;
- React DOM – це API, який використовується для виконання фактичного відображення на вебсторінці.

У компоненті React слід враховувати такі області:

- Дані – це дані, які надходять звідкись (компоненту неважливо звідки) та відображаються компонентом;
- Життєвий цикл складається з методів або Hooks, які ми реалізуємо для реагування на фази входу та виходу компонента з процесу відображення

React, коли вони відбуваються з часом. Наприклад, одна з фаз життєвого циклу – це коли компонент ось-ось буде відображено;

- Події – це код, який ми пишемо для реагування на взаємодію з користувачем;
- JSX – це синтаксис компонентів React, який використовується для опису структур інтерфейсу користувача [30].

Для реалізації графічного ядра обрано бібліотеку Three.js.

Вона дозволяє:

- створювати сцени, камери та освітлення, оперуючи зрозумілими об’єктно-орієнтованими абстракціями;
- використовувати фізично-коректні матеріали (MeshStandardMaterial). Це надає реалістичності вигляду каменю (мармуру, граніту);
- експортувати геометрію у формат STL, який є стандартом для верстатів з числовим програмним керуванням.

Особливістю реалізації є використання бібліотеки three-csg-ts (constructive solid geometry) для виконання булевих операцій над 3D-об’єктами. Це дозволяє реалізувати параметричні вирізи (віднімання об’єму умивальника з об’єму стільниці).

В якості інтегрованого середовища розробки (IDE) обрано Visual Studio Code.

Цей вибір базується на наступних перевагах:

- вбудована підтримка Git для контролю версій;
- широкий набір розширень для роботи з React та форматування коду, що забезпечує дотримання стандартів кодування;
- інтегрований термінал для управління пакетами через npm (Node Package Manager).

Vite – це локальний сервер розробки, що підтримується VoidZero Inc. Vite був написаний Еваном Ю, творцем Vue.js [21]. Він підтримує TypeScript та JSX. Він використовує Rollup та esbuild внутрішньо для пакетної обробки. Vite відстежує файли під час їх редагування, і після збереження файлу веббраузер перезавантажує код, що редагується, за допомогою процесу під назвою «Гаряча заміна модулів»

(HMR), який працює шляхом простого перезавантаження конкретного файлу, що змінюється, за допомогою модулів ES6 (ESM), замість перекомпіляції всієї програми. Vite забезпечує вбудовану підтримку рендерингу на стороні сервера (SSR). За замовчуванням він прослуховує TCP-порт 5173 [21]. Це сучасний інструмент для швидкого збірки і запуску React-додатків у розробці.

Діаграму використання бібліотек показано на рис. 4.2.

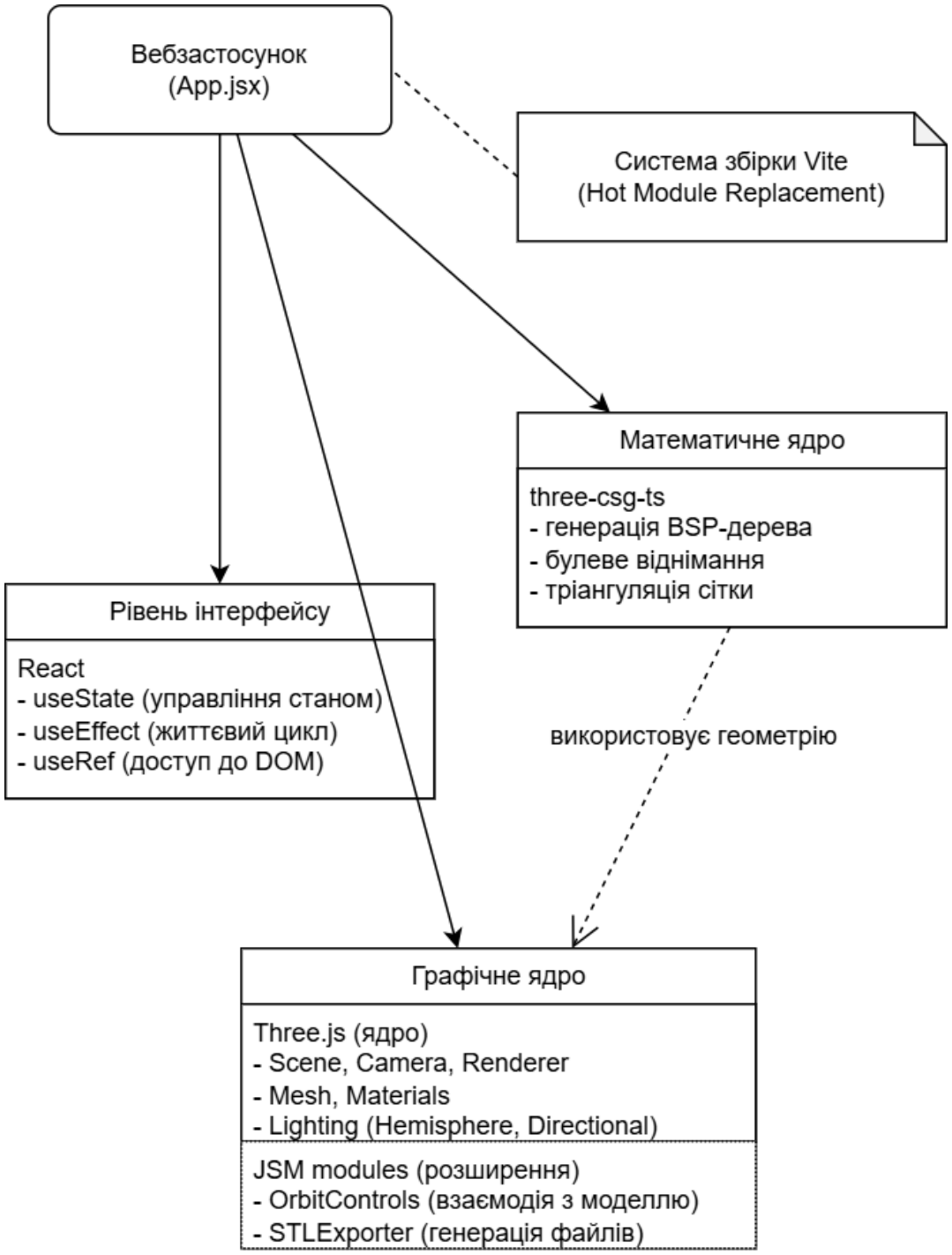


Рисунок 4.2 – Діаграма використання бібліотек

4.3 Опис реалізованих модулів та їх взаємодії

У процесі створення проєкту за допомогою команди npm create vite@latest simple-countertop -- --template react у папці simple-countertop з’являються типові структури. Основні файли прототипу, показані на рис. 4.3.

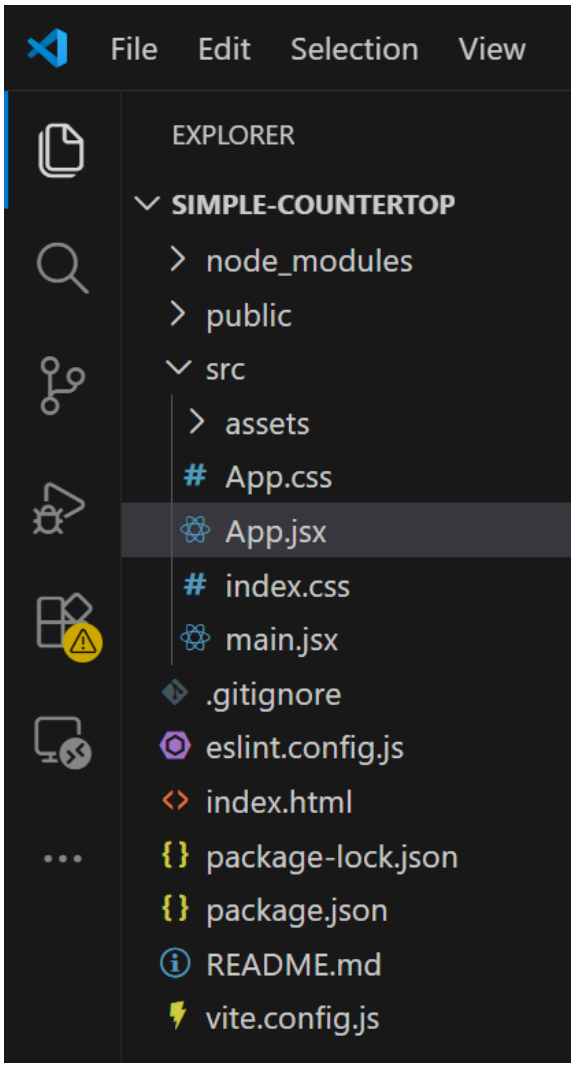


Рисунок 4.3 – Основні файли програми

JSX (JavaScript XML) дозволяє писати в одному файлі й логіку на JavaScript, і розмітку, яка зовні дуже нагадує HTML. Це і є ключ до того, щоб у React можна було зручно компонувати елементи інтерфейсу [22]. App.jsx містить UI і логіку побудови 3D-сцени (Three.js) та параметричної геометрії.

У клієнтському модулі App.jsx умовно можна виділити такі логічні компоненти:

1. UI-поверхня (панель параметрів) – компоненти таблиць і полів введення, кнопки для перемикання режимів (каркас, експорт STL), збирає параметри L, W, T, overhang, параметри вирізу, режими візуалізації;
2. Генератор геометрії (buildCountertop) перетворює параметри в конкретну 3D-геометрію: створює базову форму (BoxGeometry), формує вирізаний об’єкт (циліндр або паралелепіпед), виконує булеве віднімання (CSG), повертає кінцевий Mesh, реалізовує параметричний перетворювач вхідних числових параметрів у тривимірну модель;
3. Ініціалізація Three.js сцени (useEffect) створює Scene, Camera, Renderer, Lights, OrbitControls, ground, axes та додавання первинного Mesh, забезпечує рендер-цикл і обробку resize;
4. Оновлення моделі – під час зміни параметрів видаляється старий Mesh з сцени і додається новий, згенерований buildCountertop;
5. Експорт (STLExporter) – збереження поточної моделі у форматі STL для подальшої обробки в CAD або CAM.

Для забезпечення масштабованості та зручності супроводу програмного коду, архітектуру клієнтського вебзастосунку було спроєктовано за принципом ієрархічної декомпозиції. Головний модуль системи (React-компонент App) логічно розділено на чотири функціональні підсистеми, кожна з яких відповідає за окремий аспект роботи онлайн-конструктора. Розгорнутий опис структурної декомпозиції програмного забезпечення наведено на рис. 4.4.

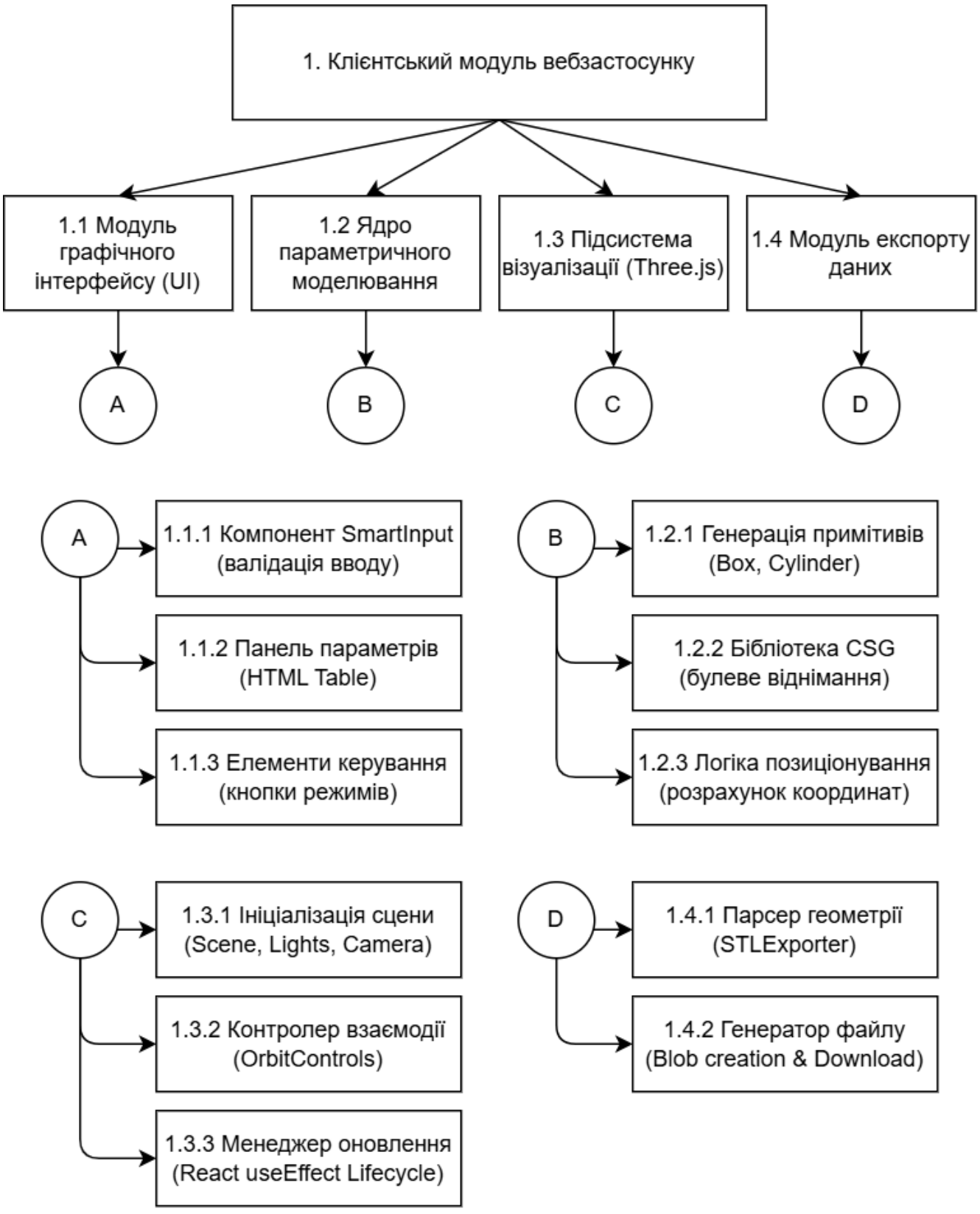


Рисунок 4.4 – Структура вебзастосунку системи

Для реалізації системи обрано об’єктно-орієнтований підхід, що передбачає поділ програми на окремі класи. Їхня структура та зв’язки зображені на рис. 4.5.

Основними програмними сутностями системи є:

1. Клас CountertopConfigurator (Головний контролер) – це центральний клас системи, який зберігає стан моделі (State). Він містить приватні поля, що відповідають фізичним параметрам стільниці (довжина length, глибина depth, параметри вирізу sinkParams). Основні методи класу відповідають за ініціалізацію процесу рендерингу (init()) та обробку подій користувача (handleParamChange()).
2. Клас GeometryBuilder (Геометричне ядро) – це службовий клас, що відповідає за математичні розрахунки. Він не зберігає стан, а лише виконує перетворення вхідних даних у 3D-меш. Основні методи класу:
   - createBasePlate() генерує геометрію плити;
   - createCutter() створює форму інструменту для вирізу (циліндр або паралелепіпед);
   - applyCSG() реалізує алгоритм булевого віднімання, використовуючи зовнішню бібліотеку.
3. Клас SceneManager (Менеджер візуалізації) – це клас-обгортка (Wrapper) над низькорівневим API бібліотеки Three.js. Він інкапсулює логіку налаштування сцени: створення камери, розміщення джерел світла (setupLights()) та керування циклом перемальовування кадрів (animate()).
4. Клас SmartInputControl (Елемент інтерфейсу) – це спеціалізований клас для полів введення, що реалізує патерн «Debounce» (відкладене оновлення). Він містить методи валідації validateRange(), що гарантують цілісність введених даних перед їх передачею у головний контролер.
5. Клас ExportService (Сервіс експорту) – відповідає за серіалізацію об’єктної моделі у файловий формат STL.

Між класами встановлено асоціативні зв’язки: головний контролер керує менеджером сцени та геометричним ядром, а також отримує дані від компонентів вводу.

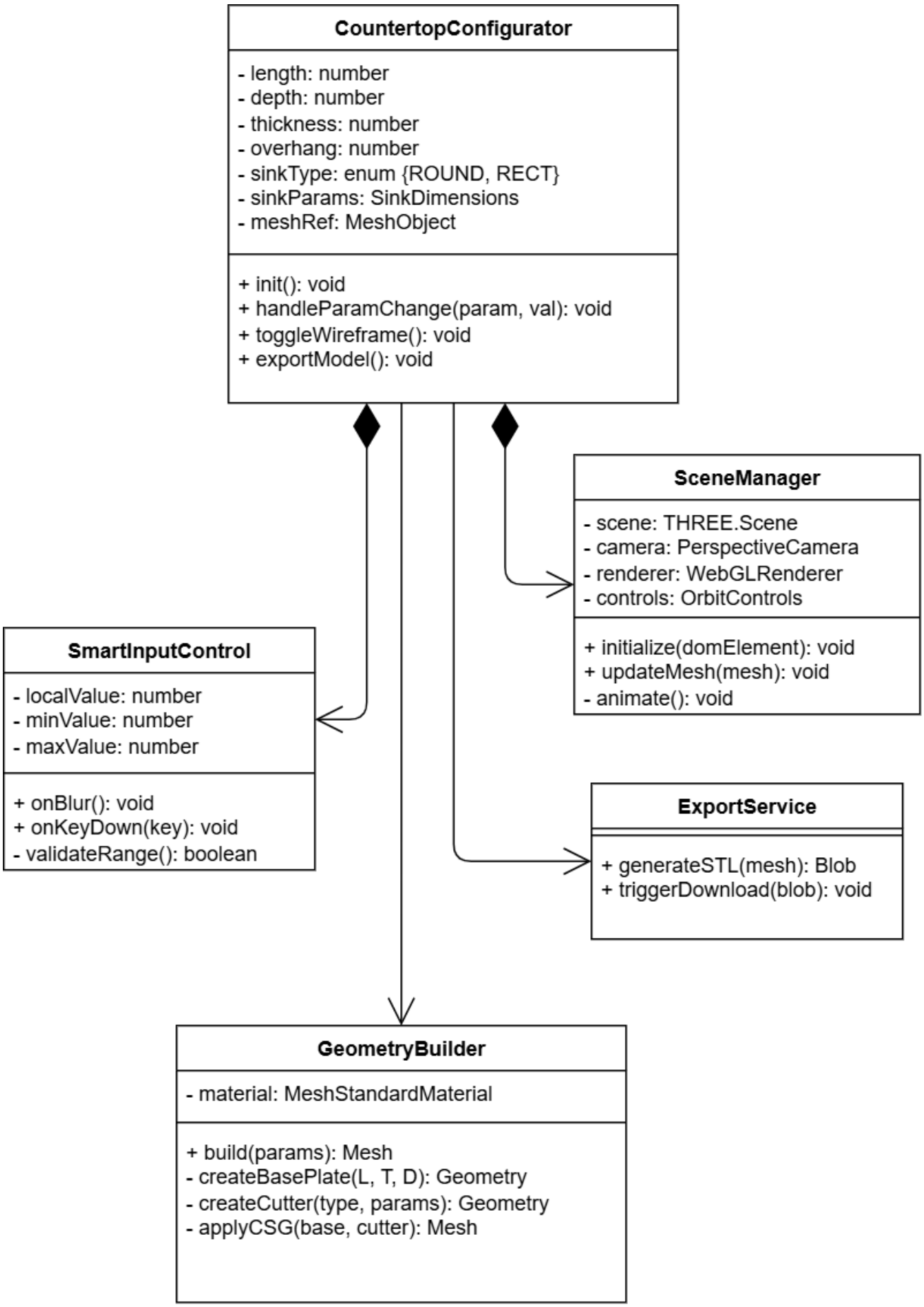


Рисунок 4.5 – Діаграма класів

На діаграмі потоків даних (Data Flow Diagram) клієнтської частини системи на рис. 4.6 зображено інформаційні потоки процесу параметричного моделювання:

1. Процес 1 відповідає за валідацію вхідних даних від користувача, відсіюючи некоректні значення перед оновленням стану.
2. Сховище відображає внутрішній стан React-додатка, який є єдиним джерелом значень для системи.
3. Процес 2 виконує основну обчислювальну роботу: на основі параметрів зі сховища формує базову геометрію та геометрію вирізу, після чого застосовує булеве віднімання (CSG).
4. Процес 3 отримує сформовану полігональну сітку (Mesh) та візуалізує її у браузері за допомогою WebGL.
5. Процес 4 активується за запитом користувача, конвертує поточну геометрію у двійковий формат STL та передає у файлову систему для подальшого використання у виробництві.

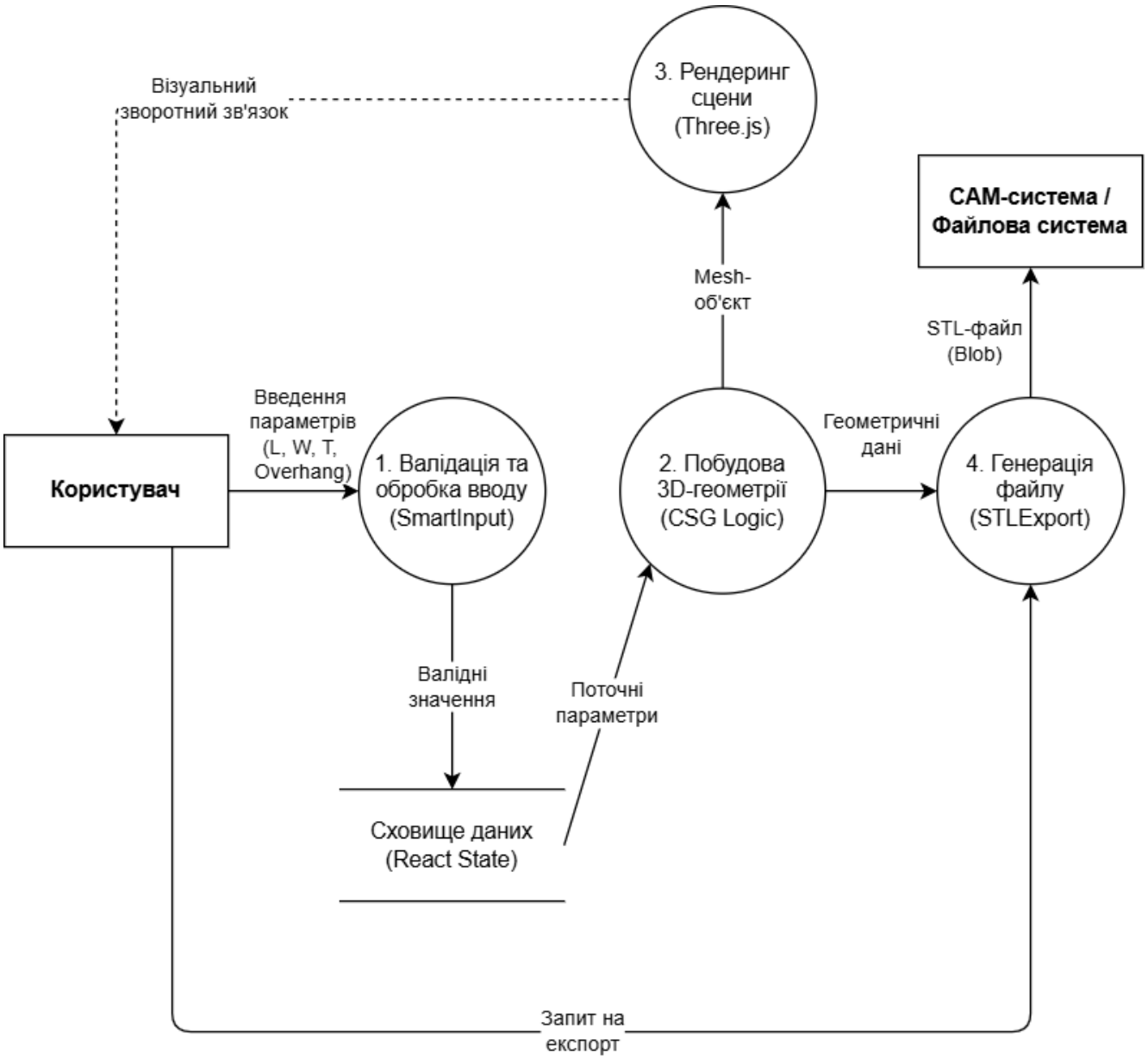


Рисунок 4.6 – Діаграма потоків даних (DFD) клієнтської частини системи

4.4 Алгоритмічне забезпечення параметричного моделювання

Основою програмної реалізації є алгоритм генерації геометрії на основі вхідних параметрів. На відміну від статичного моделювання, де об’єкт зберігається у вигляді фіксованого набору вершин, у розробленій системі зберігається лише набір параметрів (метаданих), а функцією buildCountertop динамічно будується геометрія.

Система працює у метричній системі (міліметри), проте для коректного відображення у сцені Three.js відбувається конвертація в метри за допомогою допоміжної функції mm(v).

Важливим аспектом реалізації є трансформація системи координат. За замовчуванням центр геометричного примітива у Three.js співпадає з точкою (0, 0, 0). Для зручності позиціонування та подальших розрахунків було реалізовано зсув (трансляцію) вершин таким чином, щоб точка початку координат (reference point) знаходилася у лівому задньому нижньому куті виробу, а осі координат відповідали фізичним вимірам, де вісь x – це довжина виробу, y – товщина (висота), а z – ширина. Такий підхід значно спрощує подальші розрахунки координат вирізів, оскільки всі координати знаходяться у додатному просторі.

THREE.BoxGeometry – це дуже проста 3D-геометрія, яка дозволяє створювати прямокутник, вказуючи його властивості ширини, висоти та глибини [25]. THREE.CylinderGeometry – за допомогою цієї геометрії ми можемо створювати циліндри та циліндричні об’єкти [25].

Ключовим завданням під час розробки була реалізація довільних вирізів для умивальника. Для цього застосовано метод конструктивної суцільної геометрії (CSG). За допомогою CSG можна застосовувати операції (зазвичай додавання, віднімання, різницю та перетин) для об’єднання двох геометрій. Ці бібліотеки створять нову геометрію на основі вибраної операції [25].

Алгоритм включає наступні кроки:

1. Створення базового тіла (target mesh) – прямокутної плити із заданими габаритами.
2. Створення тіла-інструменту (cutter mesh), тобто залежно від вибору користувача (sinkType), генерується або циліндр (CylinderGeometry), або прямокутний паралелепіпед (BoxGeometry).
3. Позиціонування інструменту – розрахунок координат центру вирізу базується на параметрах «Виступ спереду» (overhang) та «Відступ від краю» (sinkDepthFromFront).
4. Система підтримує створення як наскрізних отворів, так і отворів з дном (pocketing). Для цього вертикальна координата інструменту (вісь *y*) та його висота розраховуються так, щоб верхня грань інструменту гарантовано перетинала верхню площину плити, а нижня грань знаходилася на заданій глибині (sinkInsetDepth).
5. Бібліотека CSG конвертує mesh-об’єкти у BSP-дерева (binary space partitioning), виконує логічну операцію та конвертує результат назад у полігональну сітку.

Схему алгоритму методу конструктивної суцільної геометрії (CSG) показано на рис. 4.7.

У випадку виникнення помилок тріангуляції (розбиття області на маленькі трикутники [23]) під час CSG-операцій (для складних перетинів вершин), реалізовано блок try-catch, який забезпечує відмовостійкість системи, повертаючи базову геометрію замість завершення роботи додатка з помилкою (рисунок 4.8).

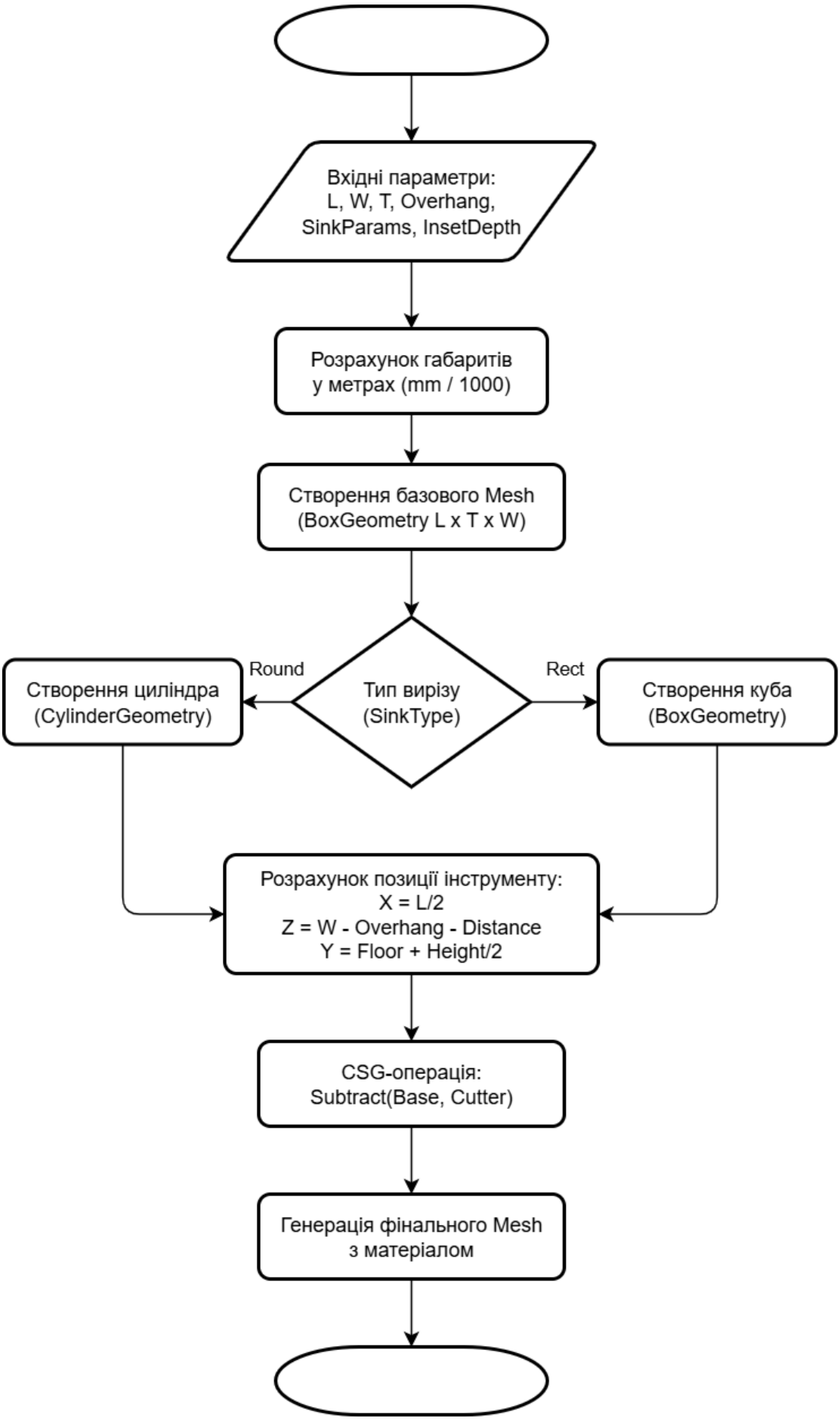


Рисунок 4.7 – Схема алгоритму методу CSG

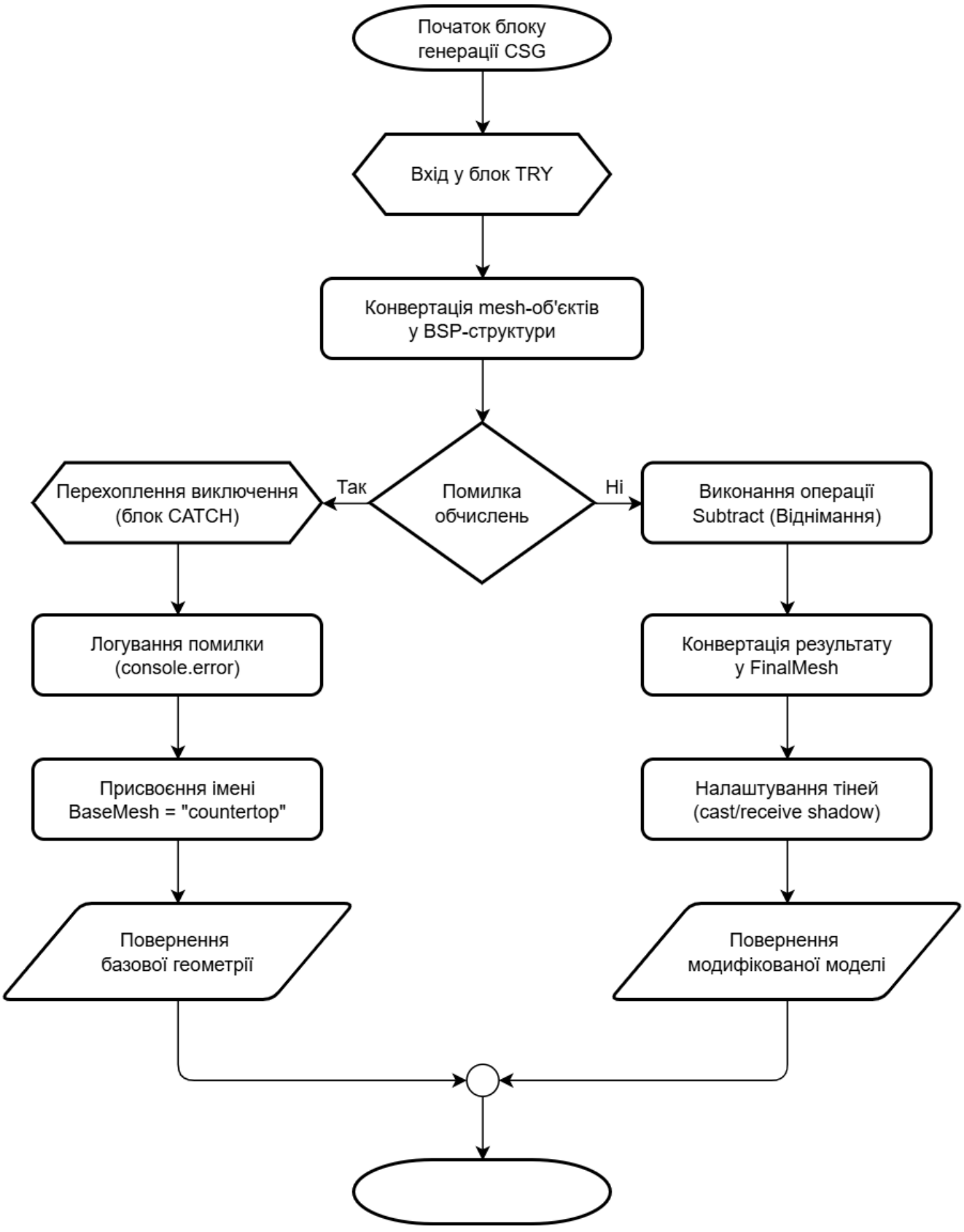


Рисунок 4.8 – Схема алгоритму try-catch

4.5 Розробка інтерфейсу та логіки взаємодії користувача

Користувацький інтерфейс реалізовано з використанням компонентного підходу React. Головний компонент App містить стан додатка (state), який описує всі параметри виробу.

Основні функціональні можливості вебзастосунку:

- динамічна зміна габаритних розмірів (довжини, ширини, товщини) та конструктивних особливостей (виступу стільниці) у визначених межах;
- підтримка зміни типу вирізу (круглий або прямокутний) та його параметрів;
- можливість обертання, масштабування моделі, увімкнення режиму «Каркас» для відображення внутрішньої структури виробу;
- можливість конвертувати поточну параметричну модель у формат STL за допомогою кнопки «Експорт STL», який сумісний з CAD та CAM-системами.

Панель керування (рисунок 4.9) реалізована у лівій частині екрану з використанням табличної верстки для чіткого вирівнювання параметрів. Поля вводу згруповані за логічним призначенням:

- габаритні розміри (довжина, ширина, товщина);
- налаштування вирізу (тип, розміри, позиціонування).

Для зручності користувача додано динамічні підказки та валідацію. Наприклад, поле «Глибина вирізу» підсвічується червоним кольором, якщо введене значення перевищує товщину матеріалу.

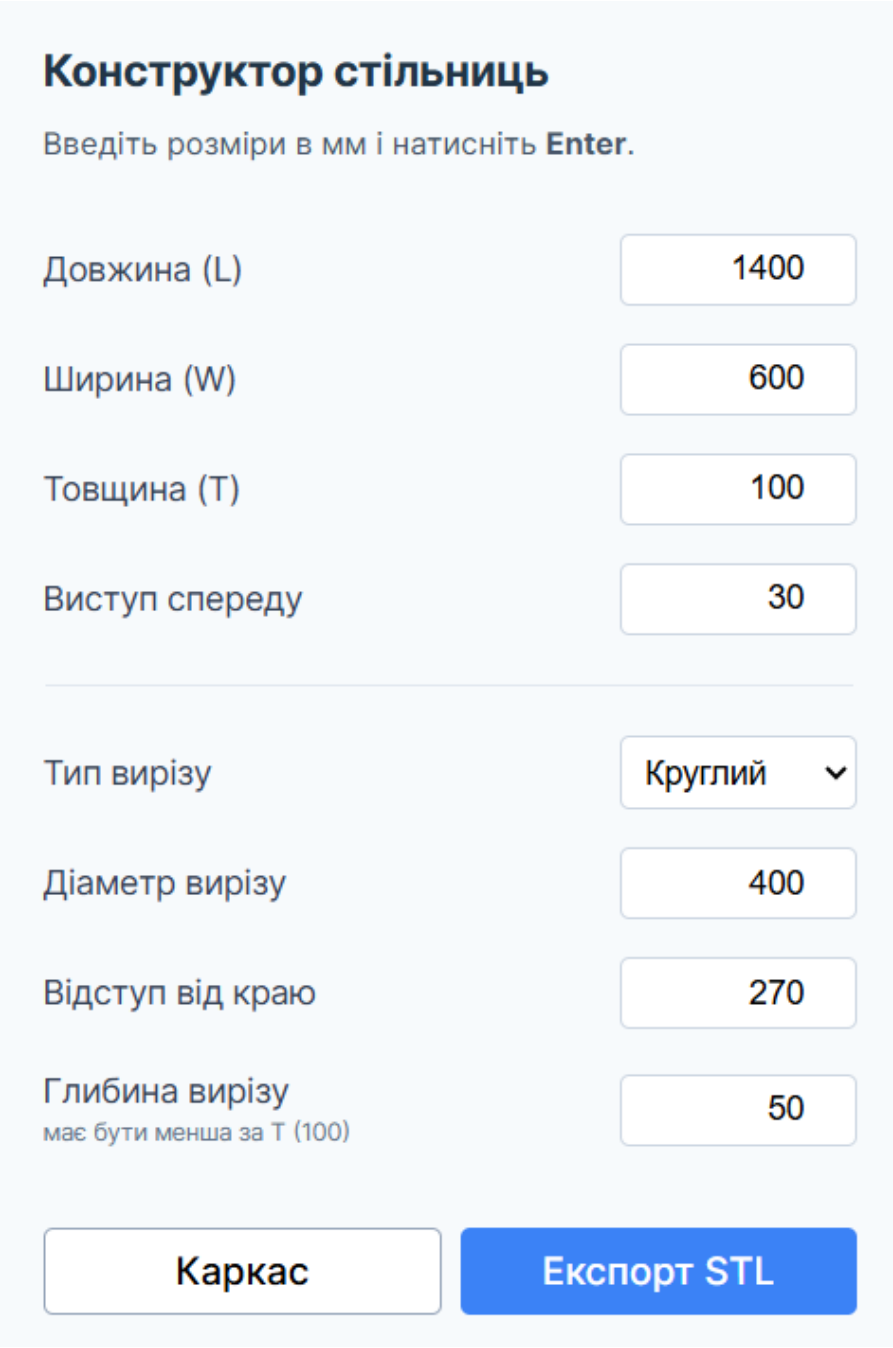


Рисунок 4.9 – Панель управління 3D-моделлю

Використовується PerspectiveCamera з кутом огляду (FOV) 45 градусів, що відповідає сприйняттю людського ока. Позиціонування камери розраховане так, щоб охопити модель середніх розмірів: camera.position.set(2, 2.5, 2.5). Для керування камерою інтегровано модуль OrbitControls, який дозволяє користувачеві обертати модель навколо її центру, а не початку координат, та масштабувати сцену. Властивість controls.enableDamping = true додає інерцію рухам, роблячи взаємодію більш плавною.

Важливою функцією системи є інтеграція з виробничим процесом. Для цього реалізовано функцію exportSTL. Вона використовує STLExporter для перетворення поточної геометрії сцени (з урахуванням всіх виконаних булевих операцій) у формат STL. Отриманий рядок даних конвертується у Blob-об’єкт типу text/plain, після чого програмно створюється посилання для завантаження файлу. Це дозволяє користувачеві зберегти точну копію спроєктованого виробу локально та передати її в CAD або CAM-програму для генерації G-коду.

4.6 Висновки

Виконано програмну реалізацію клієнтської частини інформаційної системи. Обґрунтовано вибір мови програмування та бібліотеки React.js. Three.js та three-csg-ts дозволили реалізувати складні геометричні алгоритми у браузері. Реалізовано алгоритм параметричної побудови стільниць, що включає динамічну генерацію геометрії, трансформацію координат та виконання булевих операцій для створення вирізів. Забезпечено інтеграцію з виробництвом через функцію експорту моделей. Створений прототип готовий до інтеграції у повноцінну e-commerce платформу.

5 ОЦІНКА ЕФЕКТИВНОСТІ СИСТЕМИ ТА ТЕСТОВИЙ ПРИКЛАД

5.1 Контрольний приклад функціонування системи

Для запуску локального сервера потрібно у терміналі середовища розробки ввести команду npm run dev (рисунок 5.1).

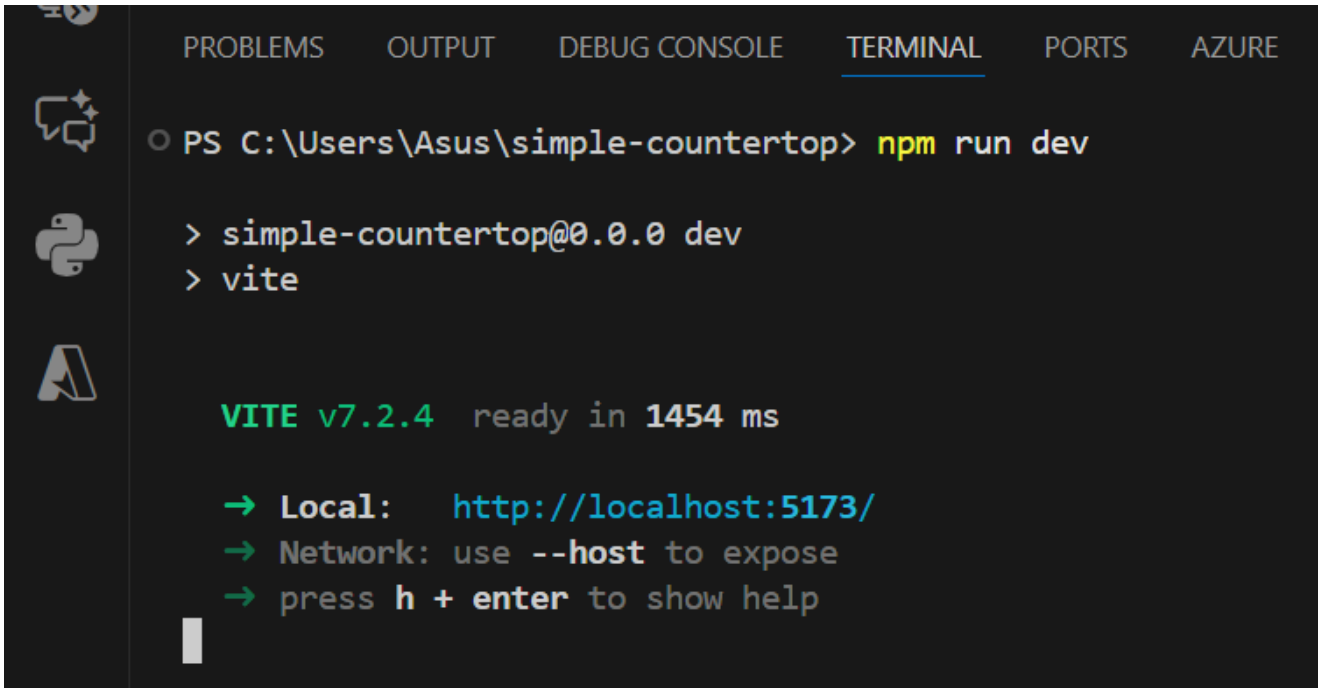


Рисунок 5.1 – Запуск локального сервера

Після переходу за посиланням http://localhost:5173/ у браузері можна побачити інтерфейс користувача як на рис. 5.2. Для перевірки працездатності реалізованого програмного забезпечення було проведено серію тестів з моделювання виробів різної конфігурації. Для зміни моделі користувачу необхідно ввести значення в поля заповнення в міліметрах та натиснути клавішу «Enter». Віддаляти та наближати стільницю можна колесом миші. Для обертання сцени потрібно натиснути та утримувати ліву кнопку миші та переміщувати курсор.

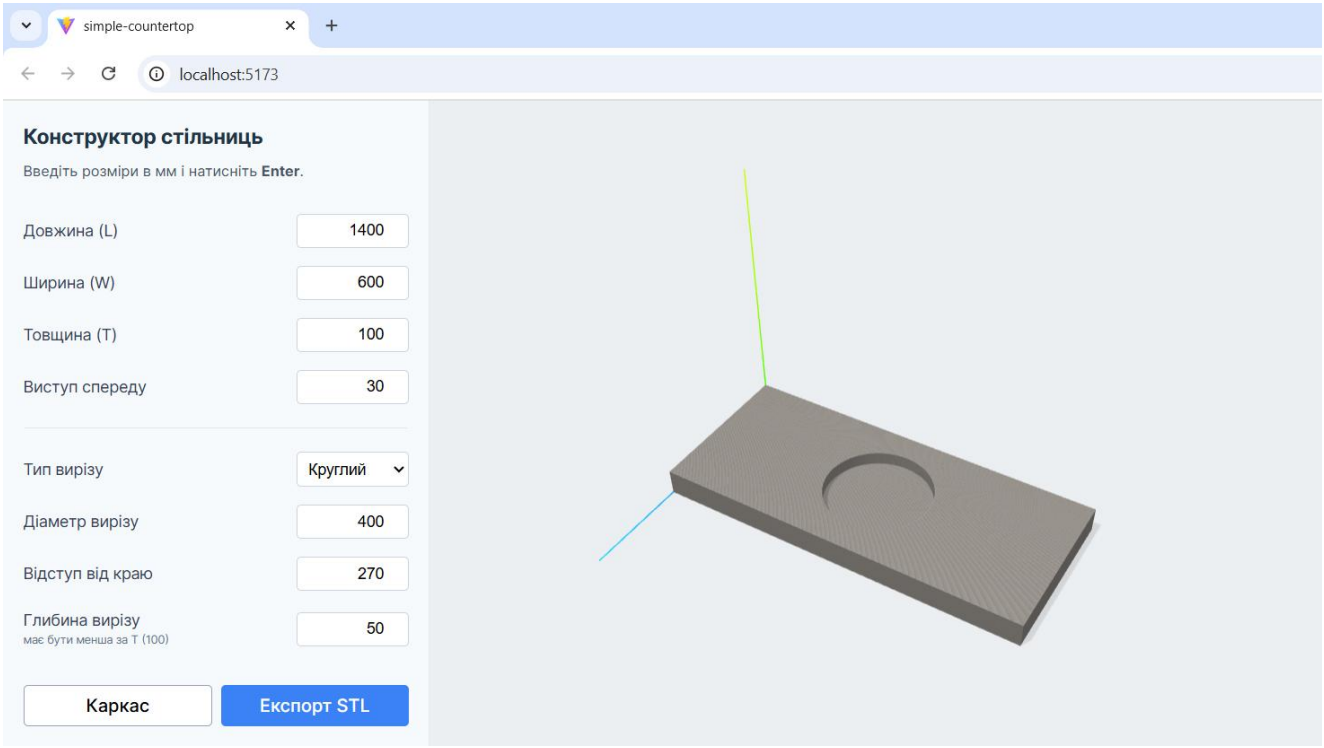


Рисунок 5.2 – Інтерфейс користувача

Сценарій 1. Стільниця з круглим вирізом для умивальника.

Вхідні дані: Довжина = 1000 мм, Ширина = 1000 мм, Товщина = 150 мм, Тип вирізу – «Круглий», Діаметр вирізу = 500 мм.

Результат: Система успішно згенерувала плиту (рисунок 5.3). Виріз циліндричної форми розміщено симетрично відносно довжини. Тіні коректно падають всередину отвору.

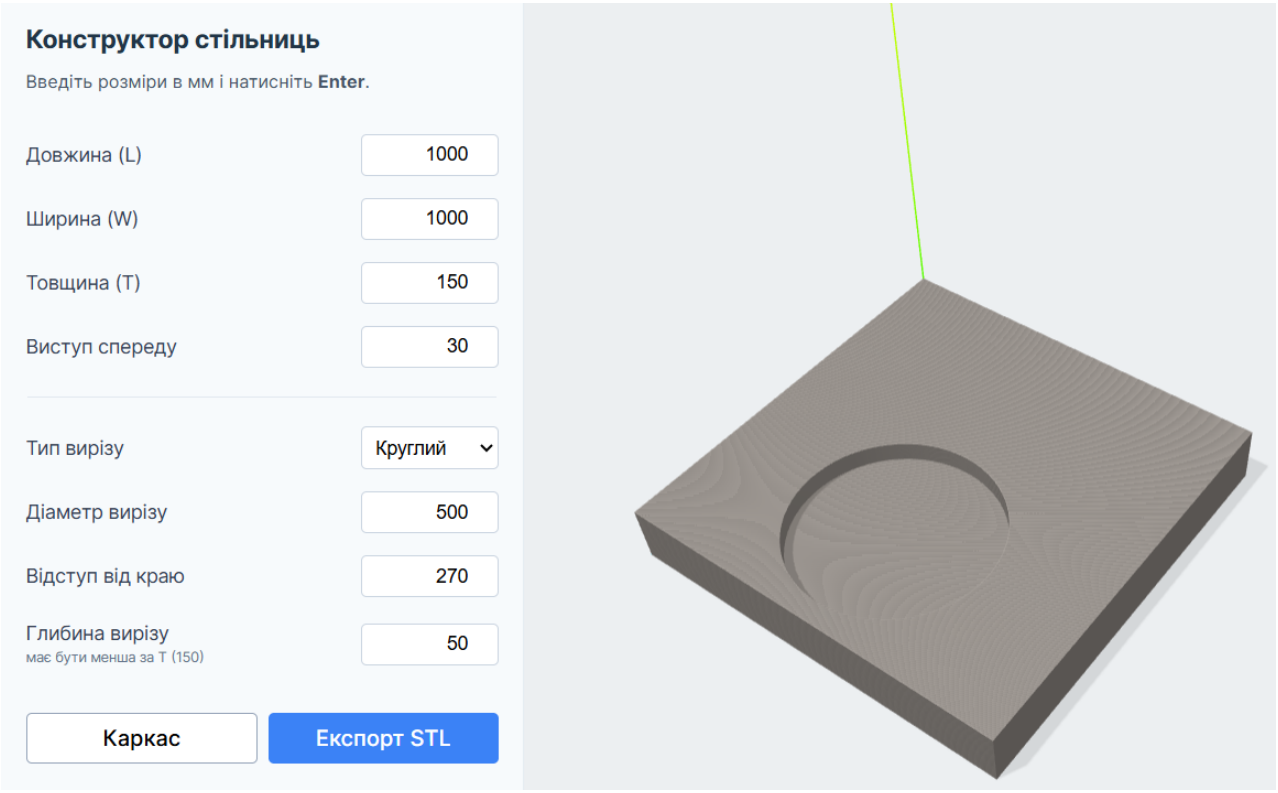


Рисунок 5.3 – Стільниця з круглим вирізом

Якщо натиснути на кнопку «Експорт STL», то файл «countertop.stl» завантажиться на комп’ютер. Якщо відкрити файл у стандартному додатку Print 3D, то можна побачити створену стільницю (рисунок 5.4).

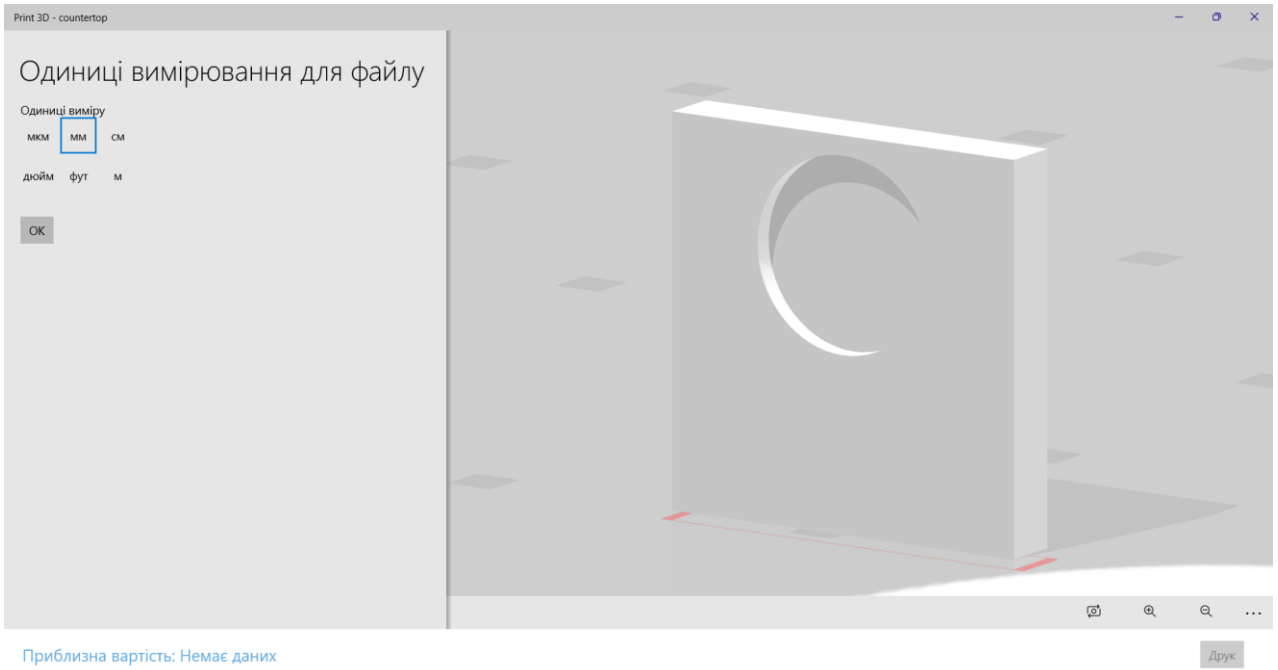


Рисунок 5.4 – Перегляд 3D-моделі у Print 3D

Сценарій 2. Стільниця з прямокутним вирізом для умивальника.

Вхідні дані: Товщина = 150 мм, Тип вирізу – «Прямокутний», Довжина вирізу = 600 мм, Ширина вирізу = 250 мм, Відступ від краю = 200 мм, Глибина вирізу = 75 мм.

Результат: Система успішно згенерувала плиту (рисунок 5.5). Виріз у формі паралелепіпеда розміщено симетрично відносно довжини.

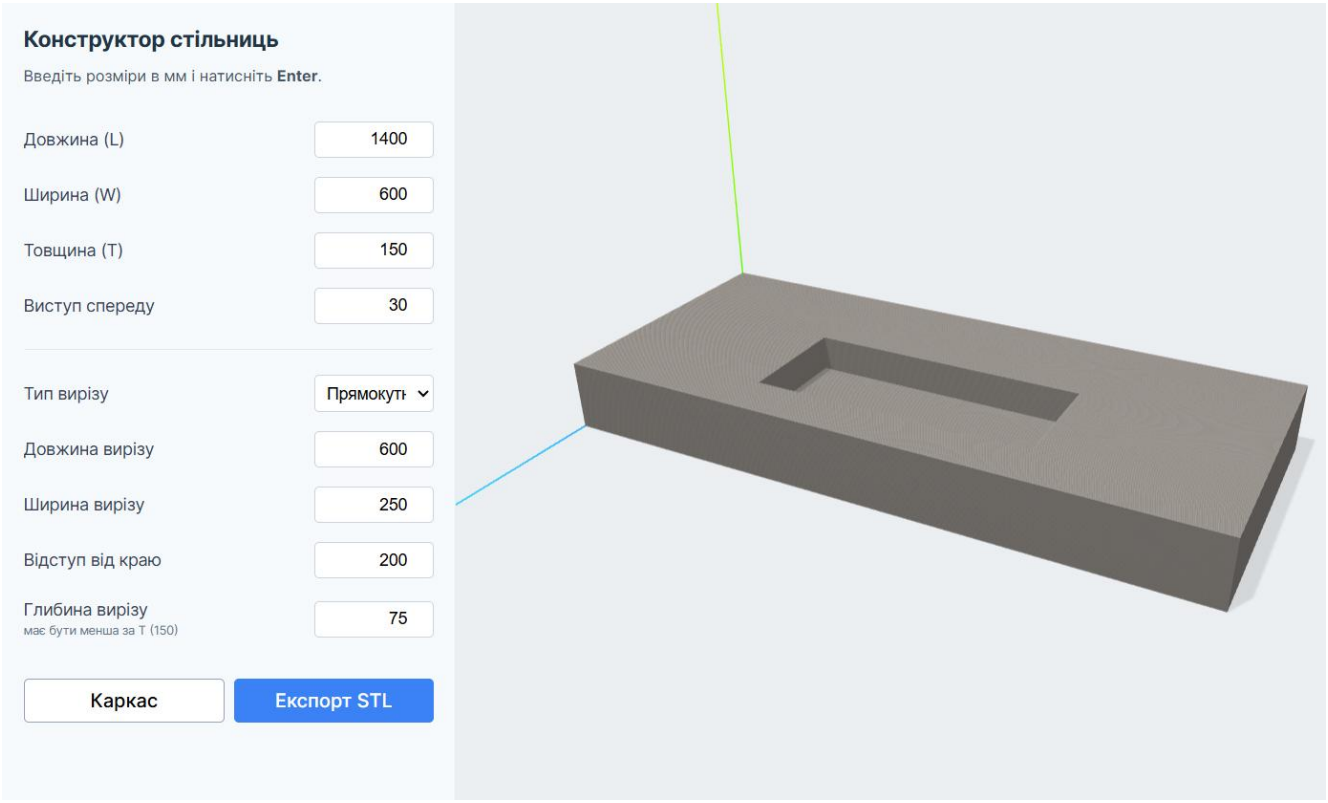


Рисунок 5.5 – Стільниця з прямокутним вирізом

Після активації режиму «Каркас» натисканням на відповідну кнопку видно, що дно вирізу знаходиться посередині товщини плити (рисунок 5.6). Цей сценарій показує правильність алгоритму розрахунку висоти ріжучого інструменту (cutterHeight) та його розташування по осі *y*.

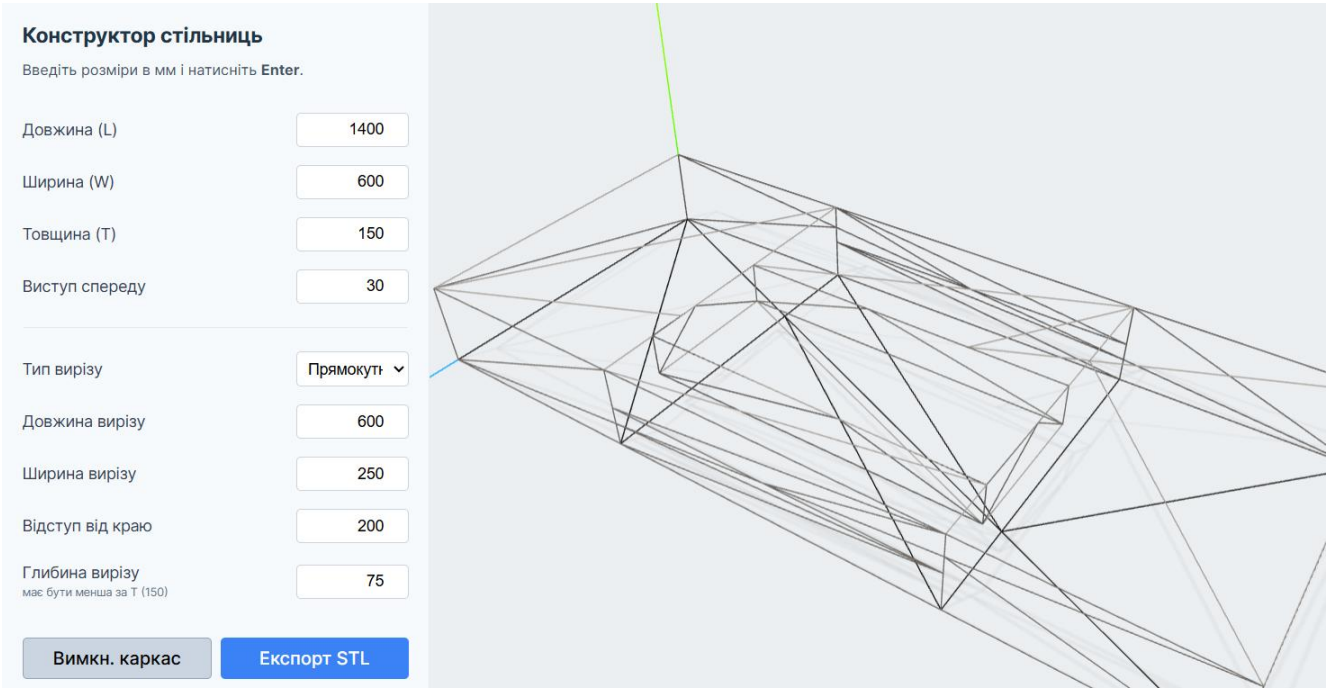


Рисунок 5.6 – Режим «Каркас»

Сценарій 3. Валідація параметрів.

Вхідні дані: Спроба ввести від’ємне значення довжини стільниці (L = -1400 мм).

Результат: Компонент SmartInput автоматично повернув значення до мінімально допустимого (400 мм) і запобіг виникненню помилки геометричного ядра (рисунок 5.7).

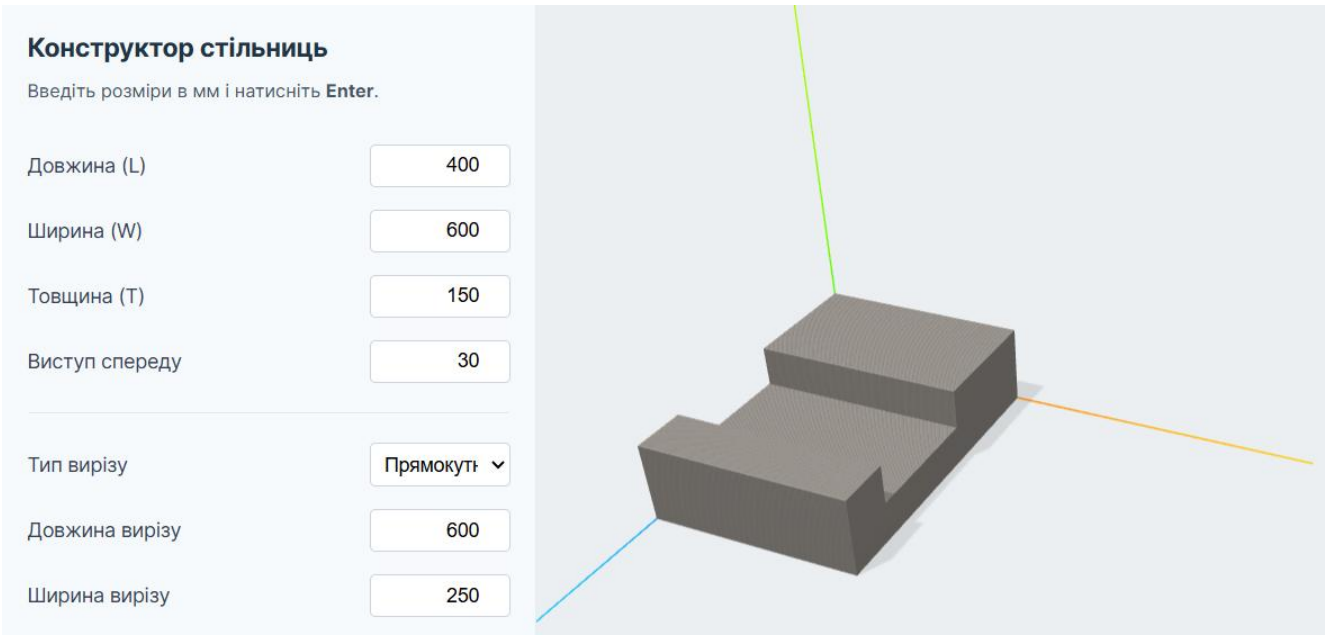


Рисунок 5.7 – Перевірка введення некоректних значень довжини стільниці

Вхідні дані: Введення глибини вирізу, більшої за товщину стільниці (Глибина вирізу = 160 мм).

Результат: Значення поля вводу автоматично змінилося на мінімальне значення, яке допустиме в цьому випадку (149 мм), оскільки глибина вирізу має бути меншою за товщину стільниці (150 мм) (рисунок 5.8). Система показує рядок із поточною товщиною стільниці у дужках під назвою «Глибина вирізу».

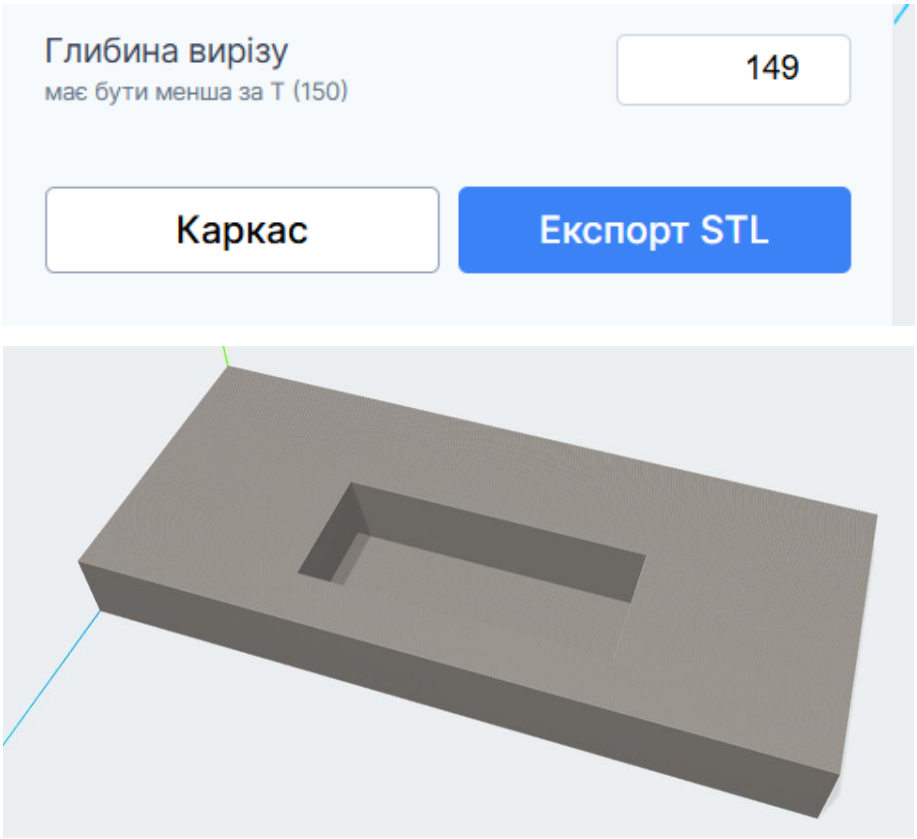


Рисунок 5.8 – Перевірка введення некоректних значень глибини вирізу

Вхідні дані: Введення товщини стільниці, меншої за глибину вирізу (Т = 140 мм).

Результат: Поле вводу глибини вирізу обвелося червоним кольором, повідомляючи користувача про помилку (рисунок 5.9).

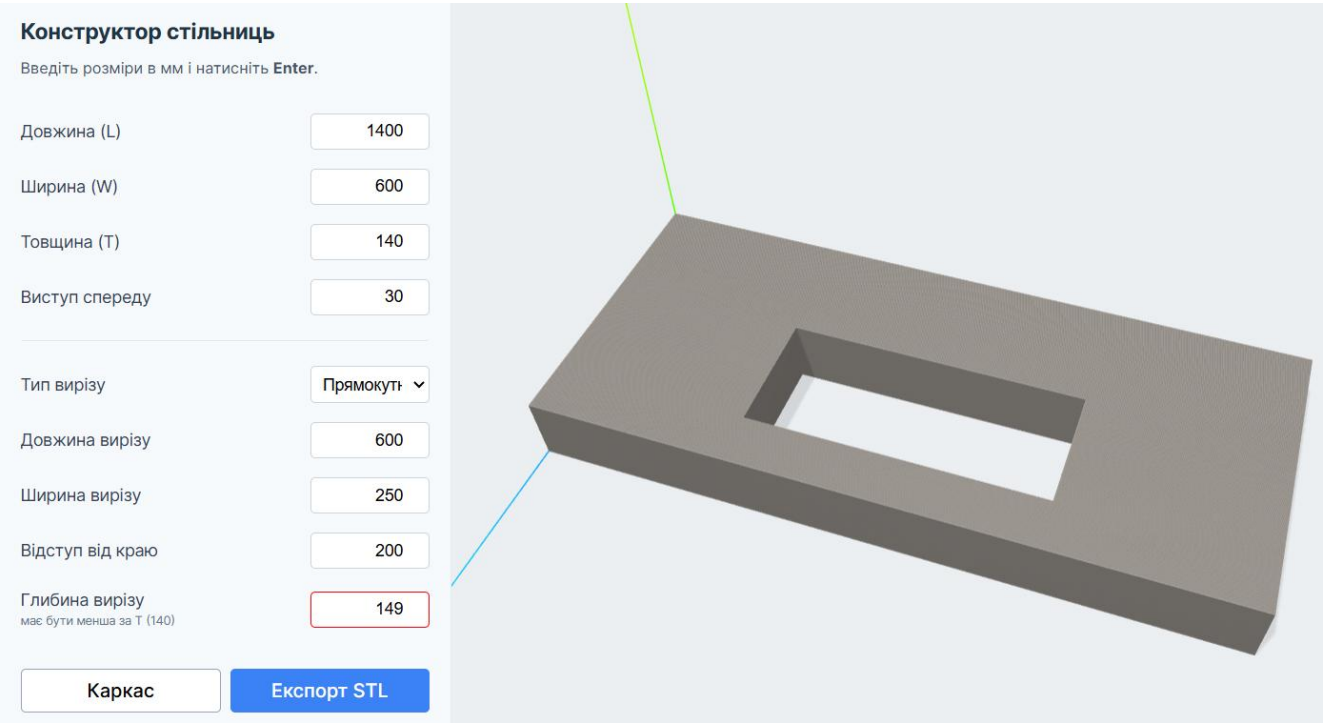


Рисунок 5.9 – Перевірка введення некоректних значень товщини стільниці

5.2 Висновки

Проведено комплексну оцінку ефективності реалізованої клієнтської частини інформаційної системи. Функціональне тестування на контрольних прикладах підтвердило правильність роботи всіх компонентів. Використання розробленого вебконфігуратора прискорює моделювання індивідуального виробу порівняно з традиційними CAD-методами. Отже, розроблена клієнтська частина системи є ефективною та може бути впроваджена у реальні виробничі процеси.

ВИСНОВКИ

У кваліфікаційній роботі виконано дослідження та розробку веборієнтованої інформаційної системи параметричного 3D-моделювання індивідуальних будівельних виробів, орієнтованої на створення та візуалізацію стільниць з натурального каменю. Отримані результати підтверджують доцільність застосування сучасних вебтехнологій React.js, Three.js та компонентної архітектури клієнтської частини, для реалізації інтерактивних конфігураторів виробів, що можуть працювати безпосередньо в браузері без встановлення додаткового програмного забезпечення.

У межах роботи спроєктовано гнучку клієнт-серверну архітектуру інформаційної системи, що забезпечує масштабованість та надійність зберігання даних. Розроблено структуру бази даних MongoDB, яка дозволяє зберігати не лише облікові дані користувачів та історію замовлень, але й складні параметричні моделі виробів. Визначено протоколи взаємодії між клієнтською частиною та сервером, що гарантує швидкий відгук системи на дії користувача.

Реалізовано модуль параметричного моделювання, який базується на динамічній перебудові 3D-геометрії залежно від введених користувачем даних. Розроблено алгоритми побудови складних форм та виконання булевих операцій для створення отворів, наприклад, для умивальників. Створено інтуїтивно зрозумілий інтерфейс користувача, який не потребує від клієнта спеціальних інженерних знань для конструювання виробу. Виконане тестування підтвердило правильність роботи алгоритмів та відповідність отриманих моделей заданим параметрам.

Система пропонує ефективніший підхід до формування замовлень, мінімізуючи участь проєктувальника на початкових етапах. Конструктор підвищує якість комунікації з клієнтами, скорочує час на підготовку комерційних пропозицій, зменшує кількість помилок під час формування технічних завдань та дозволяє замовнику самостійно попередньо оцінити вигляд виробу. Запропонована

методика може бути застосована для розробки аналогічних онлайн-конструкторів для меблів, архітектурних елементів або інших індивідуальних виробів.

Результати дослідження свідчать про доцільність подальшого розвитку системи, зокрема шляхом розширення функціоналу параметричної побудови, додавання складніших типів обробок, впровадження модуля автоматичного розрахунку вартості на основі актуальних прайс-листів та інтеграції з виробничими верстатами з ЧПК (числовим програмним керуванням). Отримані напрацювання можуть слугувати основою для створення повноцінної комерційної платформи у сфері цифрового моделювання та виготовлення індивідуальних виробів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. US countertops market – industry research report [Електронний ресурс]. – [Б. м.] : The Freedonia Group, 2024. – Режим доступу: https://www.freedoniagroup.com/industry-study/us-countertops.
2. Конфігуратор корпусів | Blum [Електронний ресурс]. – Режим доступу: https://www.blum.com/ua/uk/services/planning-construction-product-selection/cabinet-configurator/.
3. Free 3D models, CAD files and 2D drawings - TraceParts [Електронний ресурс]. – Режим доступу: https://www.traceparts.com/en.
4. Peikko Designer® Home [Електронний ресурс]. – Режим доступу: https://www.peikkodesigner.com/.
5. KRONAS Master 3D [Електронний ресурс]. – Режим доступу: https://master.kronas.com.ua/.
6. WebGL - Wikipedia [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/WebGL.
7. Three.js - Wikipedia [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Three.js.
8. WebGL 2.0 Specification [Електронний ресурс]. – Режим доступу: https://registry.khronos.org/webgl/specs/latest/2.0/.
9. three.js docs [Електронний ресурс]. – Режим доступу: https://threejs.org/docs/.
10. Parisi T. Programming 3D Applications with HTML5 and WebGL: 3D Animation and Visualization for Web Pages [Електронний ресурс] : підручник / Tony Parisi. – [Б. м.] : O'Reilly Media, 2014. – 404 с. – Режим доступу: https://www.google.com.ua/books/edition/Programming_3D_Applications_with_HTML5_a/UvrYAgAAQBAJ?hl=uk&gbpv=0.
11. 3D computer graphics - Wikipedia [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/3D_computer_graphics.

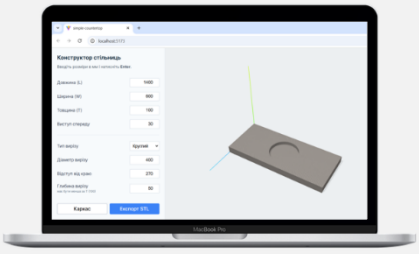
12. Anttila J. Creating room designer proof of concept with Three JS [Електронний ресурс] : тези / Anttila Joni. – Turku, 2017. – 42 с. – Режим доступу: https://www.theseus.fi/bitstream/handle/10024/139106/Anttila_Joni.pdf?sequence=1&isAllowed=y.
13. Wang S. Development of a Cloud-Based Building Information Modeling Design Configurator to Auto-Link Material Catalogs with Code-Compliant Designs of Residential Buildings [Електронний ресурс] / Songyue Wang, Qian Chen // Buildings. – 2024. – Т. 14, № 7. – С. 2084. – Режим доступу: https://doi.org/10.3390/buildings14072084.
14. Cross-phase product configurator for modular buildings using kit-of-parts [Електронний ресурс] / Jianpeng Cao [та ін.] // Automation in Construction. – 2021. – Т. 123. – С. 103437. – Режим доступу: https://doi.org/10.1016/j.autcon.2020.103437.
15. Bimstore | Download BIM objects for Revit | Free BIM library [Електронний ресурс]. – Режим доступу: https://www.bimstore.co/.
16. Doshi D. How to Build a 3D Product Configurator for E-commerce in 2025 [Електронний ресурс] / Darshak Doshi // theintellify.com. – Режим доступу: https://theintellify.com/build-3d-product-configurator-for-e-commerce/.
17. Copeland R. MongoDB Applied Design Patterns [Електронний ресурс] : підручник / Rick Copeland. – [Б. м.] : O'Reilly Media, 2013. – 176 с. – Режим доступу: https://www.google.com.ua/books/edition/MongoDB_Applied_Design_Patterns/t2dBlNfKk6gC?hl=uk&gbpv=0.
18. Mesh моделювання у 3D: пояснення для початківців і не тільки [Електронний ресурс] // easy3dprint.com.ua. – Режим доступу: https://easy3dprint.com.ua/uk/shcho-take-mesh-modelyuvannya/.
19. JavaScript - Вікіпедія [Електронний ресурс] // uk.wikipedia.org. – Режим доступу: https://uk.wikipedia.org/wiki/JavaScript.
20. Огляд популярних JavaScript фреймворків [Електронний ресурс] // foxminded.ua. – Режим доступу: https://foxminded.ua/freimvorky-javascript/.

21. Vite (software) - Wikipedia [Електронний ресурс] // en.wikipedia.org. – Режим доступу: https://en.wikipedia.org/wiki/Vite_(software).
22. Шайтан В. Що таке JSX і як він працює? [Електронний ресурс] / Володимир Шайтан // blog.ithillel.ua. – Режим доступу: https://blog.ithillel.ua/articles/what-is-jsx-and-how-does-it-work.
23. Triangulation: basic graphics algorithms [Електронний ресурс] // ubc.ca. – Режим доступу: https://personal.math.ubc.ca/~cass/graphics/text/www/pdf/ch15.pdf.
24. What is three-tier architecture? [Електронний ресурс] // ibm.com. – Режим доступу: https://www.ibm.com/think/topics/three-tier-architecture.
25. Dirksen J. Learn Three.js: Program 3D Animations and Visualizations for the Web with JavaScript and WebGL [Електронний ресурс] : підручник / Jos Dirksen. – 4-те вид. – Birmingham : Packt Publishing, 2023. – 554 с. – Режим доступу: https://www.google.com.ua/books/edition/Learn_Three_js/DGKoEAAAQBAJ?hl=uk&gbpv=0.
26. Matsuda K. WebGL Programming Guide: Interactive 3D Graphics Programming with WebGL [Електронний ресурс] : підручник / Kouichi Matsuda, Rodger Lea. – [Б. м.] : Pearson Education, 2013. – 552 с. – Режим доступу: https://www.google.com.ua/books/edition/WebGL_Programming_Guide/3c-jmWkLNwUC?hl=uk&gbpv=0.
27. Babylon.js - Wikipedia [Електронний ресурс] // en.wikipedia.org. – Режим доступу: https://en.wikipedia.org/wiki/Babylon.js.
28. Verge3D - Wikipedia [Електронний ресурс] // en.wikipedia.org. – Режим доступу: https://en.wikipedia.org/wiki/Verge3D.
29. PlayCanvas - Wikipedia [Електронний ресурс] // en.wikipedia.org. – Режим доступу: https://en.wikipedia.org/wiki/PlayCanvas.
30. Boduch A. React and React Native: A Complete Hands-on Guide to Modern Web and Mobile Development with React.js [Електронний ресурс] : підручник / Adam Boduch, Roy Derks. – 3-тє вид. – Birmingham :

Packt Publishing, 2020. – 526 с. – Режим доступу: https://www.google.com.ua/books/edition/React_and_React_Native/XCLhDwAAQBAJ?hl=uk&gbpv=0.

ДОДАТОК А

Інформаційні слайди



Інформаційна система параметричного 3D-моделювання індивідуальних будівельних виробів

Виконав: студент 2-го курсу, групи ІСТм-24

Спеціальність: 126 «Інформаційні системи та технології»

Макарчук Д. А.

Керівник: д.т.н., проф. Бородавка Є. В.

1

Актуальність проєкту

Актуальність дослідження зумовлена стрімким розвитком цифрових технологій у сфері будівництва, зростанням попиту на індивідуальні вироби та потребою у впровадженні автоматизованих систем для підвищення ефективності проєктування та виробництва.

Об'єктом дослідження є методи та інструменти реалізації веборієнтованої системи параметричного 3D-моделювання індивідуальних будівельних виробів.

Предметом дослідження є веборієнтована система параметричного 3D-моделювання стільниць.

Мета дипломного проєкту полягає у встановленні та обґрунтуванні підходів до розробки інформаційної системи параметричного 3D-моделювання стільниць.

Методами дослідження є системний аналіз, методи об'єктно-орієнтованого проєктування, математичне моделювання, алгоритмізація, методи комп'ютерного 3D-моделювання та візуалізації, методи вебпрограмування, тестування програмного забезпечення.

2

Аналітичний огляд

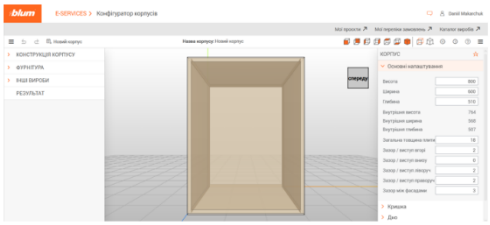
Обсяг ринку Сполучених Штатів Америки

$7,4 млрд

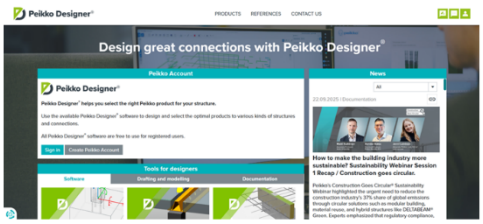
у 2023 році

3

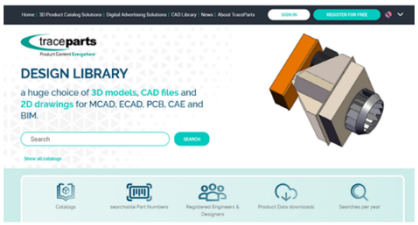
Аналіз існуючих рішень



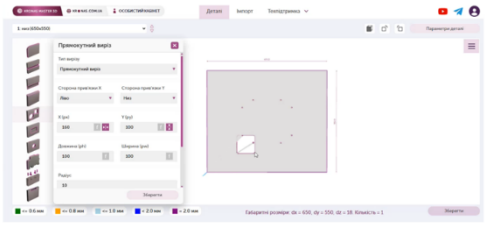
Blum Cabinet Configurator



Peikko Designer



TraceParts



KRONAS MASTER 3D

4

Постановка завдання

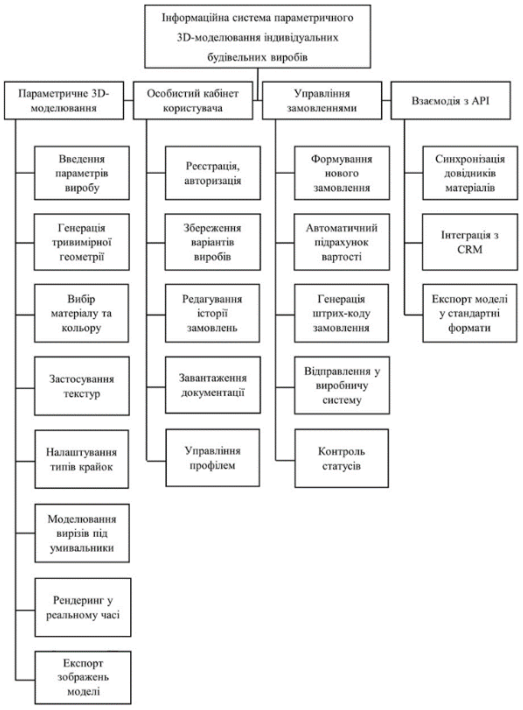
Головним завданням магістерської роботи буде створення онлайн-конструктора як зручного сервісу для оформлення замовлень з обробки індивідуальних стільниць з мармуру з 3D-візуалізацією деталей та обробок.

Завдання для досягнення головної мети:

- спроєктувати архітектуру системи (структуру бази даних, серверної та клієнтської частин);
- розробити програмні модулі системи (модуль параметричного 3D-моделювання та візуалізації, модуль особистого кабінету користувача, модуль управління замовленнями);
- реалізувати прототип системи з можливістю створення параметричних моделей індивідуальних виробів;
- провести тестування та оцінити ефективність рішення.

5

Діаграма дерева функцій

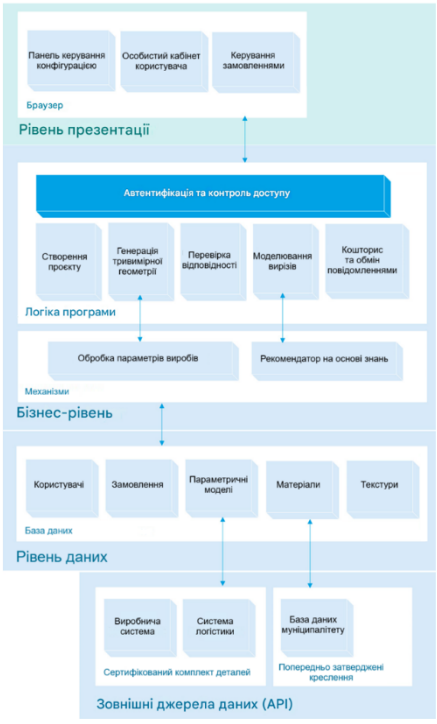


6

Модель бізнес-процесу IDEF0

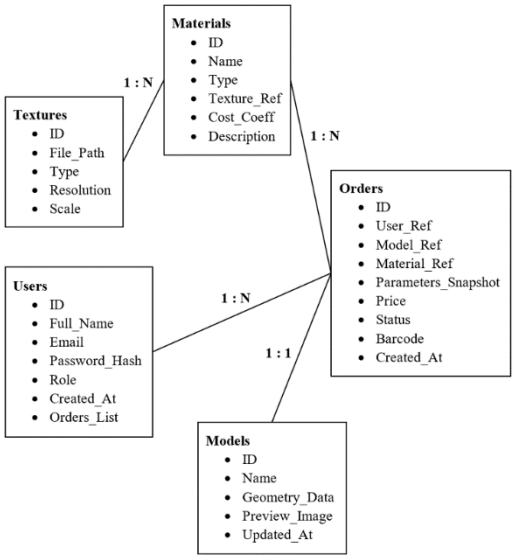
7

Архітектурна модель системи

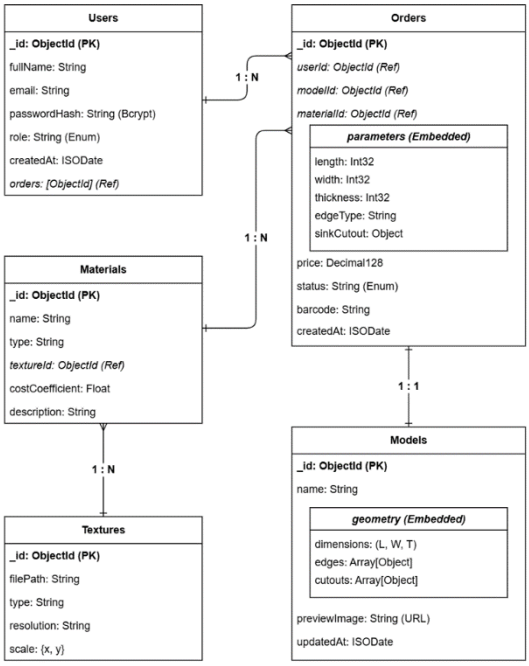


8

Схема бази даних



Логічна схема бази даних



Фізична модель бази даних

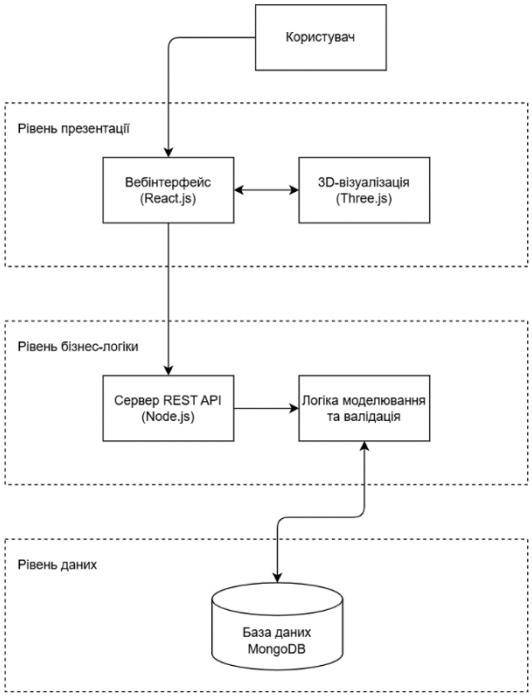
Сучасні технології розробки

Порівняння фреймворків

Фреймворк	Three JS	Unity	Babylon JS
Мова	JavaScript	C#, UnityScript	JavaScript
Ліцензія	MIT	Власницька	Ліцензія Apache 2.0
Мобільна підтримка	Так	Ні	Так
Середовище розробки	Без обмежень	Unity	Без обмежень
Інтегрована фізика	Ні	PhysX	Cannon JS, Oimo JS, Energy JS
Імпорт	JSON, OBJ, FBX	OBJ, FBX	OBJ, FBX, Babylon, STL, JSON
Експорт	JSON, OBJ	Ні	Формати, підтримувані Blender та 3dsMax
Підтримка VR	Так	Так	Так

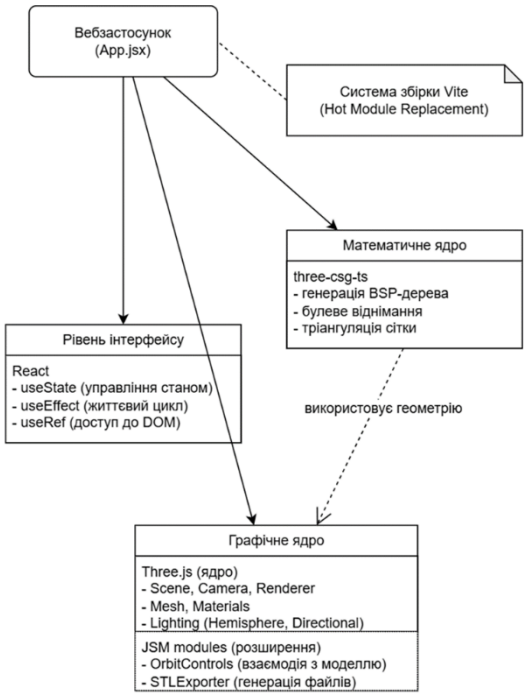
Шлях користувача із запропонованим конфігуратором

Загальна схема реалізації архітектури програмного забезпечення



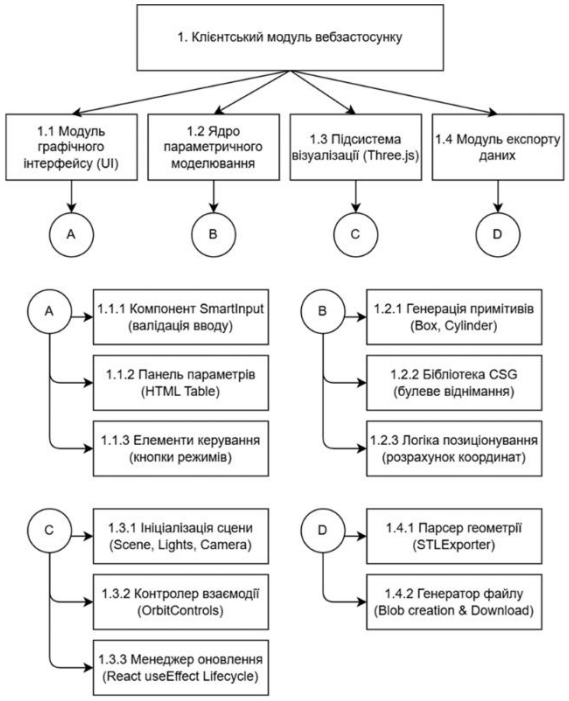
11

Діаграма використання бібліотек



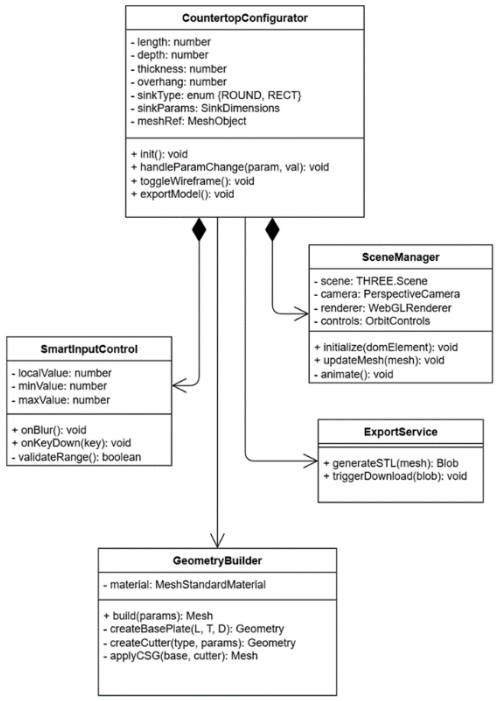
12

Структура вебзастосунку системи



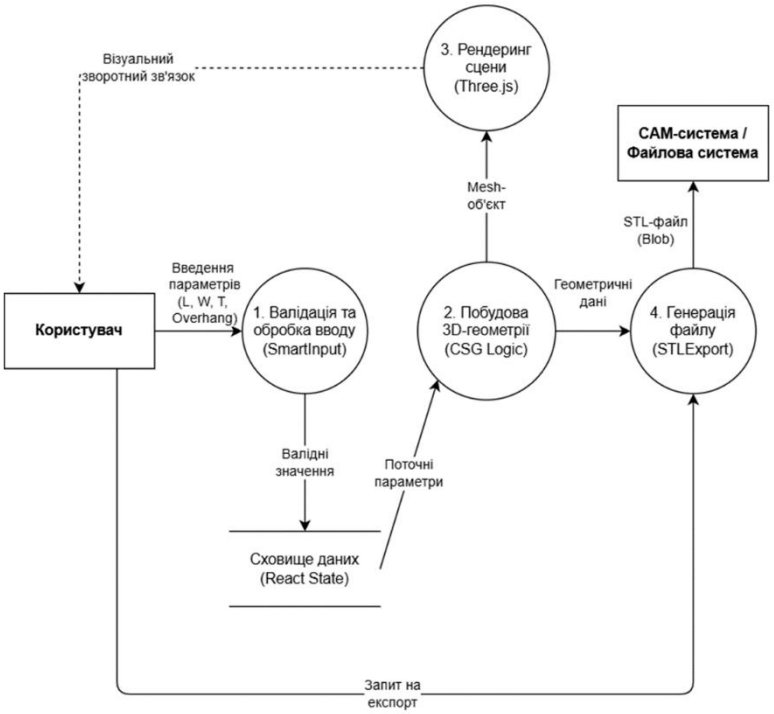
13

Діаграма класів



14

Діаграма потоків даних



Алгоритмічне забезпечення

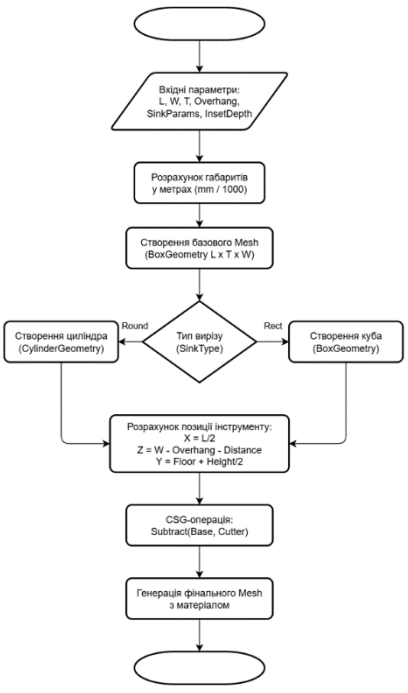


Схема алгоритму методу CSG

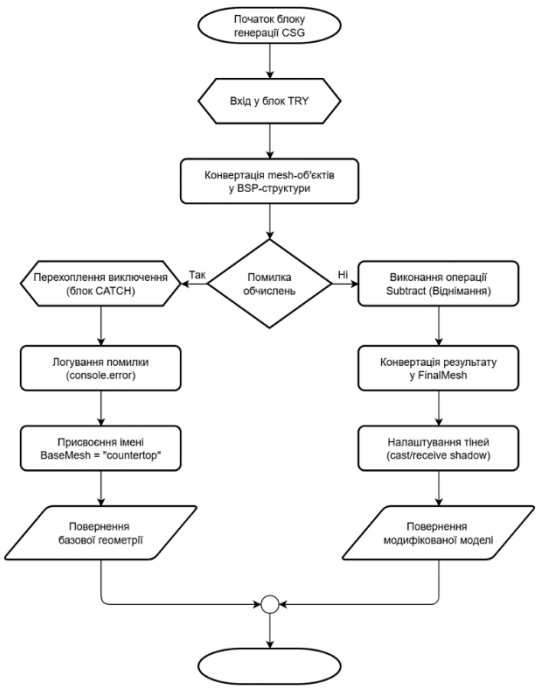
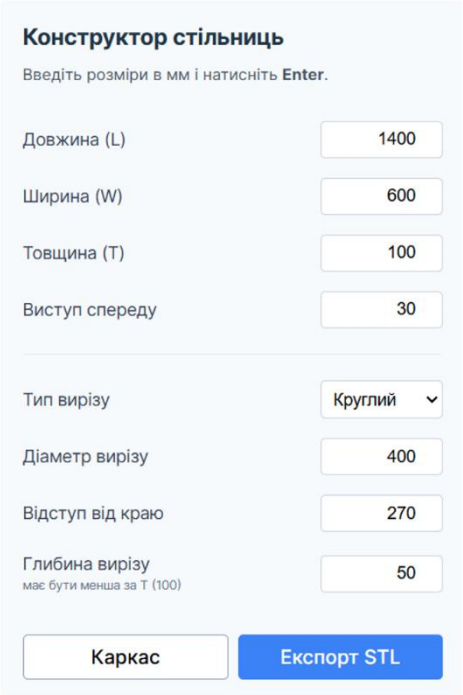


Схема алгоритму try-catch

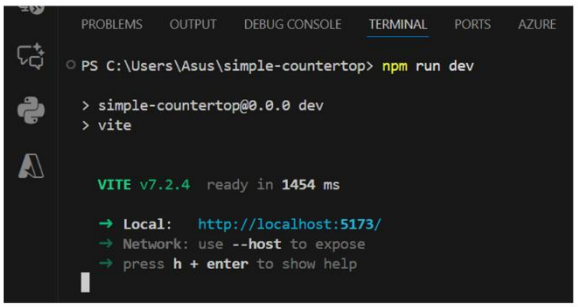
Прототип

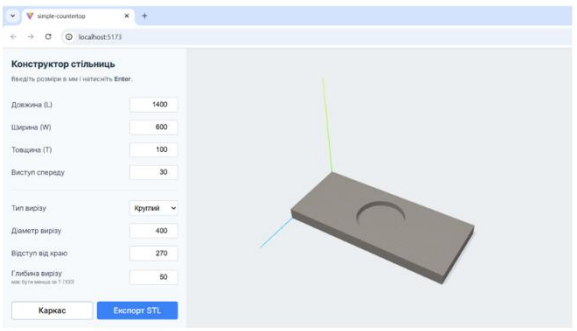
інтерфейсу користувача



17

Контрольний приклад





18

Висновки

У кваліфікаційній роботі виконано дослідження та розробку веборієнтованої інформаційної системи параметричного 3D-моделювання індивідуальних будівельних виробів, орієнтованої на створення та візуалізацію стільниць з натурального каменю. Отримані результати підтверджують доцільність застосування сучасних вебтехнологій React.js, Three.js та компонентної архітектури клієнтської частини.

У межах роботи спроєктовано гнучку клієнт-серверну архітектуру інформаційної системи, що забезпечує масштабованість та надійність зберігання даних. Розроблено структуру бази даних MongoDB, яка дозволяє зберігати не лише облікові дані користувачів та історію замовлень, але й складні параметричні моделі виробів. Визначено протоколи взаємодії між клієнтською частиною та сервером, що гарантує швидкий відгук системи на дії користувача. Реалізовано модуль параметричного моделювання, який базується на динамічній перебудові 3D-геометрії залежно від введених користувачем даних. Створено інтуїтивно зрозумілий інтерфейс користувача, який не потребує від клієнта спеціальних інженерних знань для конструювання виробу. Виконане тестування підтвердило правильність роботи алгоритмів та відповідність отриманих моделей заданим параметрам.

Результати дослідження свідчать про доцільність подальшого розвитку системи, зокрема шляхом розширення функціоналу параметричної побудови, додавання складніших типів обробок, впровадження модуля автоматичного розрахунку вартості на основі актуальних прайс-листів та інтеграції з виробничими верстатами з ЧПК (числовим програмним керуванням). Отримані напрацювання можуть слугувати основою для створення повноцінної комерційної платформи у сфері цифрового моделювання та виготовлення індивідуальних виробів.