ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА

(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій

(повне найменування факультету)

Кафедра інформаційних технологій

(повна назва кафедри)

# Кваліфікаційна робота

на здобуття ступеня вищої освіти «Бакалавр»

**«Розробка веб-системи «Конфігуратор для збирання ПК»»**

**(тема кваліфікаційної роботи українською мовою)**

**«Development of a PC Configuration Web System»**

**(тема кваліфікаційної роботи англійською мовою)**

Виконав: здобувач денної форми навчання

спеціальності 122 Комп'ютерні науки

(код, назва спеціальності)

Освітня програма Комп'ютерні науки

(назва)

Вознюк Артемій Леонідович

(прізвище, ім'я, по-батькові здобувача)

Керівник к.ф.-м.н., доцент Ткач Т.Б. (підпис)

(науковий ступінь, вчене звання, прізвище, ініціали)

Рецензент к.т.н., доцент Перелигін Б.В.

(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
Інформаційних технологій .
№ 1 від 09 червня 2024 р.

Завідувачка кафедри

(підпис) КАЗАКОВА Надія .

(прізвище,ім'я)

Захищено на засіданні ЕК № 13 .
протокол № 14 від 20 червня 2024 р.

Оцінка добре / С / 75 .

(за національною шкалою/шкалою ECTS/ бали)

Голова ЕК

(підпис) КОПИЧЕНКО Іван .

(прізвище,ім'я)

**Одеса 2024**

**ЗМІСТ**

# ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

CSS – Cascading Style Sheets.

DFD – діаграми потоків даних.

HTML – Hypertext Markup Language.

MySQL – це система управління базами даних, яка використовується для підтримки реляційних баз даних.

UML – уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування.

БД – база даних.

ПК – персональний комп'ютер.

СУБД – система управління базами даних.

# ВСТУП

У сучасному світі, де інформаційні технології та комп'ютерна техніка займають центральне місце в житті кожної людини, збірка персональних комп'ютерів стає дедалі актуальнішою. З розвитком технологій зростає кількість компонентів та їх варіацій, що дозволяє користувачам створювати ПК, які відповідають їхнім індивідуальним потребам та бюджету. Проте, для багатьох користувачів процес вибору та поєднання компонентів може бути складним і заплутаним.

Тема даної роботи – розробка веб-системи «Конфігуратор для збирання ПК» є актуальною, оскільки створення такої системи дозволяє спростити цей процес для кінцевого користувача. Така система надає можливість не лише вибрати відповідні компоненти для збирання ПК, але й гарантує їхню сумісність та оптимальну продуктивність. Це особливо важливо для тих, хто не має глибоких технічних знань, але бажає отримати якісний продукт.

Актуальність розробки обумовлена зростаючою потребою у зручних та інтуїтивно зрозумілих інструментах для збирання ПК. Існуючі рішення часто мають обмежений функціонал або складний інтерфейс, що може відлякувати потенційних користувачів. Запропонована веб-система надасть можливість користувачам швидко та ефективно підбирати компоненти, перевіряти їх сумісність, отримувати рекомендації та оцінювати загальну вартість обраної конфігурації.

Необхідність розробки такої системи полягає в підвищенні доступності та зручності процесу збирання ПК, що сприятиме збільшенню кількості користувачів, які зможуть самостійно збирати ПК під свої потреби. Це, в свою чергу, дозволить їм зекономити кошти на послугах спеціалістів та отримати більше знань про комп'ютерну техніку.

Таким чином, розробка веб-системи «Конфігуратор для збирання ПК» є важливим кроком на шляху до спрощення процесу вибору та збирання

комп'ютерів, що підвищить задоволеність користувачів та покращить їхній досвід у взаємодії з сучасними технологіями.

Метою роботи є надання зручного засобу для продажу ПК, що включає конфігуратор.

Для досягнення цієї мети вирішити наступні задачі:

– проаналізувати предметну область;
– провести огляд існуючих аналогів веб-системи, що розробляється;
– визначити вимоги до веб-системи;
– провести проєктування веб-системи;
– реалізувати веб-систему;
– провести тестування веб-системи.

Об’єкт роботи – процес розробки веб-системи, що містить конфігуратор ПК, надання зручного та зрозумілого інтерфейсу.

Предмет роботи – веб-система, що включає конфігуратор ПК.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

Комп'ютер – практично незамінний інструмент у житті кожної сучасної людини, який зі своїх причин якось прив'язаний до її щоденної експлуатації. Офісна частина, прикладні програми, ігри – все це вимагає різний рівень потужності компонентів комп'ютера, з часом які старіють і стають все менш і менш актуальними на тлі новинок, що виходять щороку, в тій чи іншій сфері. Що й спонукає до оновлення комп'ютерних комплектуючих. Завдяки доступності довідкової інформації, процес підбору комп'ютера не становить великої складності.

З розвитком ринку технологій, зросла і загальна компетентність користувачів, які потребують більш поглибленого підбору заліза для свого ПК. І відповідно, не змусив на себе довго чекати і конфігуратор, який сильно спростив складання, аж до вибору необхідних комплектуючих на одній сторінці, тим самим залучаючи більший потік потенційних клієнтів. Так само важливу роль відігравав захист від неправильної сумісності компонентів. Так само перевагою над готовими зборками досить часто виступає ціна, можливість щось не включати в комплект, відповідно не переплачуючи за це зайвих коштів або, навпаки, включити щось необхідне до свого ПК, яке готове складання не матиме, у свою чергу , магазин. Величезною перевагою є незалежний вибір комплектуючих, цілком частим явищем можна побачити готове складання, де сам виробник часто намагається на чомусь заощадити.

Таким чином, було ухвалено рішення про створення помічника-конфігуратора ПК.

Конфігуратор – простий інструмент, що є онлайн-конструктором, націлений спростити процес вибору клієнтом комплектуючих, при цьому спрощуючи і сам вибір майбутнього комп'ютера. Особливістю такого підходу можна назвати:

1) більшу клієнтоорієнтованість, що відповідно привабить більше зацікавлених у збиранні користувачів, а саме економія часу клієнта та

простий інтерфейс з доступністю вибору всіх компонентів ПК на одній сторінці;

2) індивідуальний вибір кожного комплектуючого допоможе догодити перевагам клієнта у виборі улюблених брендів та вендорів;

3) перевірка вибору відповідність деяких комплектуючих, які просто не підійдуть друг до друга. Це робить конфігуратор потрібним у наш час для успішної роботи механізму магазину з продажу ПК.

### 1.1 Порівняльний аналіз аналогів

Для створення найбільш зрозумілого та ефективного для користувача конфігуратора та магазину, в тому числі, необхідно вивчити вже існуючі та успішно функціонуючі рішення. Ця необхідність зумовлена кінцевим поданням проєкту, аналізом працездатності для виявлення переваг, які варто взяти на озброєння, а також недоліків, які краще уникнути при створенні конфігуратора та веб-системи в цілому.

Розглянемо для порівняння кілька веб-систем інтернет-магазинів ПК, що включають конфігуратор ПК, і які представляють себе не перший рік на ринку.

Першим розглянемо магазин, що має конфігуратор – telemart.ua [1] (https://telemart.ua/ua/ assembly.html). Дана система має простий і інтуїтивно зрозумілий інтерфейс, навіть для користувача, який там знаходиться вперше. Запускалася дана система на різних пристроях. Програма працює коректно як на комп'ютері, так і на смартфоні, що говорить про хорошу адаптивність (рис.1.1).

Розглянемо більш детально функціональні можливості даної системи, проаналізуємо та зробимо висновки.

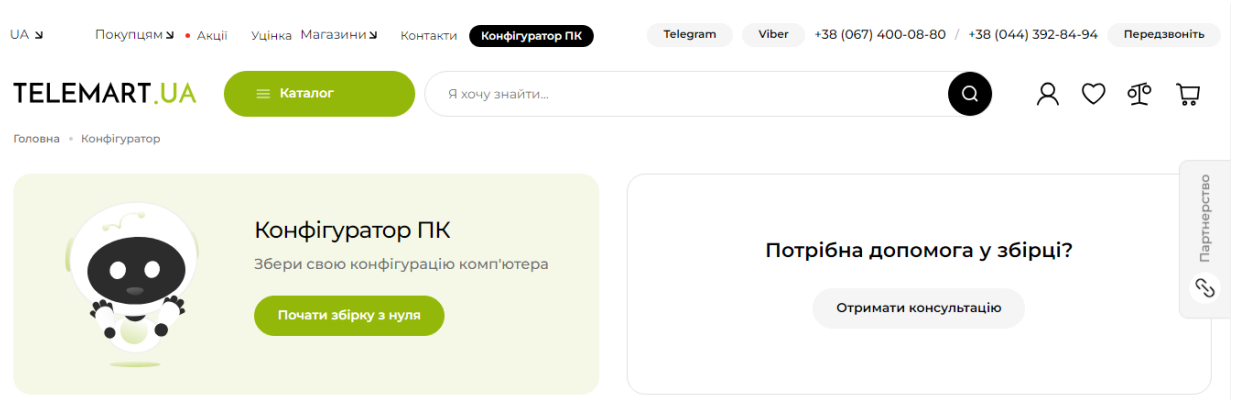


Рисунок 1.1 – Головна сторінка магазину telemart.ua

У меню конфігуратор (рис.1.2) одразу запропоновано вибрати необхідні, а також необов'язкові для роботи комплектуючі (периферійні пристрої та аксесуари).

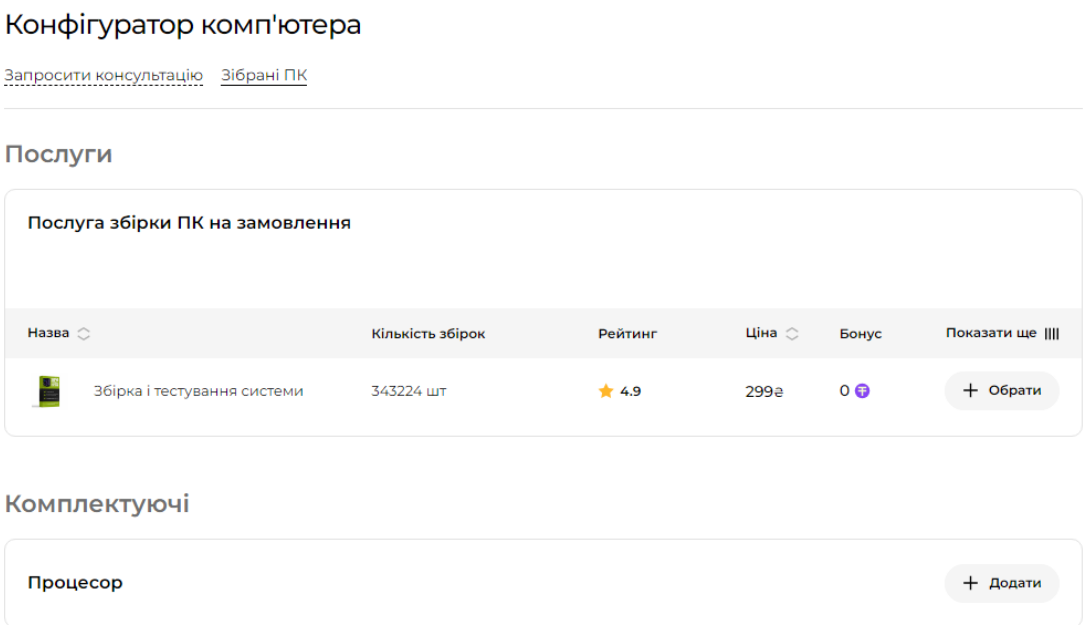


Рисунок 1.2 – Сторінка вибору конфігурації ПК магазину telemart.ua

Наведемо результати аналізу магазину telemart.ua з конфігуратором, опишемо його переваги та недоліки.

Telemart.ua – це один з провідних українських онлайн-магазинів, який спеціалізується на продажі комп'ютерної техніки та комплектуючих. Однією з ключових функцій цього магазину є його конфігуратор ПК, який дозволяє користувачам самостійно підбирати та збирати комп'ютери за власними потребами. Розглянемо основні переваги та недоліки цього інструменту.

До переваг системи можна віднести:

1. Зручний інтерфейс: інтерфейс конфігуратора простий та інтуїтивно зрозумілий, що дозволяє навіть новачкам швидко орієнтуватися та вибирати потрібні компоненти.

2. Широкий асортимент компонентів: магазин пропонує великий вибір комплектуючих від різних виробників, що дозволяє користувачам знайти компоненти, які відповідають їхнім вимогам та бюджету.

3. Перевірка сумісності компонентів: конфігуратор автоматично перевіряє сумісність обраних компонентів, що допомагає уникнути помилок та проблем під час збирання ПК.

4. Актуальні ціни та наявність: всі ціни та наявність компонентів постійно оновлюються, що дозволяє користувачам отримати актуальну інформацію про вартість та доступність товарів.

5. Можливість збереження конфігурації: користувачі можуть зберігати свої конфігурації для подальшого редагування або поділитися ними з іншими, що є зручним для обговорення та узгодження конфігурацій.

6. Детальні технічні характеристики та відгуки: кожен компонент має детальний опис та технічні характеристики, а також відгуки інших користувачів, що допомагає зробити обґрунтований вибір.

До недоліків системи можна віднести:

1. Обмежену кількість налаштувань: хоча конфігуратор пропонує багато компонентів, іноді не вистачає гнучкості в налаштуванні певних аспектів конфігурації, таких як конкретні моделі охолодження або специфічні типи корпусів.

2. Відсутність деяких брендів: незважаючи на широкий асортимент, іноді відсутні компоненти від певних брендів або моделей, що може обмежувати вибір користувачів.

3. Можливі технічні проблеми: у деяких випадках користувачі можуть стикатися з технічними проблемами, такими як зависання або некоректна робота конфігуратора, особливо при великому навантаженні на сайт.

4. Не завжди актуальні акції та знижки: інформація про акції та знижки не завжди вчасно оновлюється в конфігураторі, що може призводити до розбіжностей між очікуваною та фактичною ціною.

5. Складність для новачків: хоча інтерфейс загалом зручний, новачки все ж можуть відчувати складнощі з розумінням деяких технічних аспектів вибору компонентів, особливо якщо вони не мають достатнього досвіду у сфері комп'ютерної техніки.

Таким чином, конфігуратор ПК на сайті Telemart.ua є корисним інструментом, який спрощує процес вибору та збирання персональних комп'ютерів для користувачів з різним рівнем технічної підготовки. Він має багато переваг, таких як зручний інтерфейс, широкий асортимент компонентів, автоматична перевірка сумісності та можливість збереження конфігурацій. Однак, як і будь-яка система, він має свої недоліки, зокрема обмеженість у налаштуваннях, відсутність деяких брендів, можливі технічні проблеми та певні складнощі для новачків. У цілому, використання цього конфігуратора може значно спростити та прискорити процес збирання ПК для більшості користувачів.

Наступним розглянемо конфігуратор ПК https://can.ua/configurator/. Конфігуратор ПК від CAN.ua [2] – легкий спосіб зібрати комп'ютер онлайн з перевіркою фізичної сумісності комплектуючих.

Вбудовані в онлайн конфігуратор функції перевірки сумісності обраних компонентів інформують про помилки або проблемних місцях обраної збірки, дають підказки щодо їх усунення перед ухваленням рішення про покупку або збереженні конфігурації, при цьому носять виключно

рекомендаційний характер і не обмежують у процесі складання вашого майбутнього ПК.

При виборі компонентів відбувається візуалізація їх зовнішнього вигляду, автоматично розраховується актуальна ціна, послідовно формуючи вартість всієї збірки ПК. Програма конструктор досить швидко допоможе зібрати і купити системний блок з нуля. Присвоїть ідентифікаційний номер збірці, збереже її в Вашому особистому кабінеті на сайті. При необхідності завжди можна роздрукувати відібрані змісти конфігурацію або дізнатися думку досвідченого експерта, скориставшись формою зворотного зв'язку або онлайн-чатом (рис.1.3).

Якщо виникнуть труднощі при самостійній збірки є можливість звернення до менеджерів-експертів.

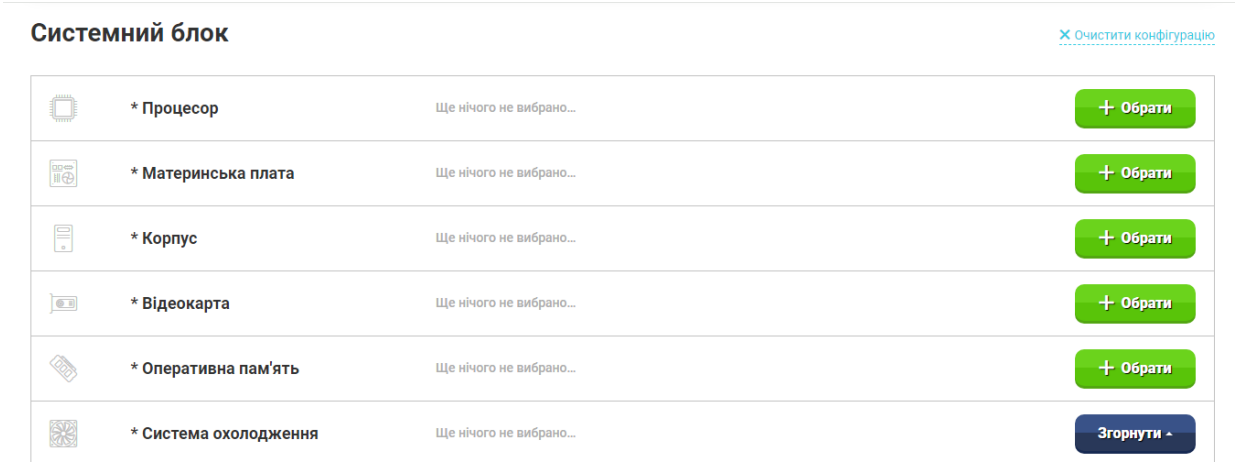


Рисунок 1.3 – Конфігуратор ПК від CAN.ua

Проаналізуємо більш детально систему з виявленням недоліків та переваг.

До переваг системи віднесемо:

1. Зручний інтерфейс: конфігуратор має простий та інтуїтивно зрозумілий інтерфейс, що полегшує процес вибору та складання компонентів навіть для користувачів без технічного досвіду.

2. Широкий асортимент компонентів: магазин пропонує великий вибір комплектуючих від різних виробників, що дозволяє користувачам знайти компоненти, які відповідають їхнім вимогам та бюджету.

3. Перевірка сумісності компонентів: конфігуратор автоматично перевіряє сумісність обраних компонентів, що допомагає уникнути проблем під час складання ПК.

4. Актуальні ціни та наявність: всі ціни та наявність компонентів постійно оновлюються, що забезпечує актуальність інформації для користувачів.

5. Можливість збереження конфігурації: користувачі можуть зберігати свої конфігурації для подальшого редагування або поділитися ними з іншими, що є зручним для обговорення та узгодження конфігурацій.

6. Детальні технічні характеристики та відгуки: кожен компонент має детальний опис та технічні характеристики, а також відгуки інших користувачів, що допомагає зробити обґрунтований вибір.

7. Інтеграція з підтримкою: конфігуратор має інтеграцію з системою підтримки, що дозволяє користувачам швидко отримати допомогу від фахівців магазину у разі виникнення питань.

До недоліків системи можна віднести наступні:

1. Обмежена кількість налаштувань: незважаючи на широкий асортимент, іноді не вистачає гнучкості у налаштуванні певних аспектів конфігурації, таких як конкретні моделі охолодження або специфічні типи корпусів.

2. Відсутність деяких брендів: не всі популярні бренди та моделі можуть бути доступні у конфігураторі, що обмежує вибір користувачів.

3. Можливі технічні проблеми: у деяких випадках користувачі можуть стикатися з технічними проблемами, такими як зависання або некоректна робота конфігуратора під час великого навантаження на сайт.

4. Не завжди актуальні акції та знижки: інформація про акції та знижки не завжди вчасно оновлюється в конфігураторі, що може призводити до розбіжностей між очікуваною та фактичною ціною.

5. Складність для новачків: хоча інтерфейс загалом зручний, новачки все ж можуть відчувати складнощі з розумінням деяких технічних аспектів вибору компонентів, особливо якщо вони не мають достатнього досвіду у сфері комп'ютерної техніки.

Таким чином, конфігуратор ПК на сайті CAN.ua є корисним інструментом для складання та вибору персональних комп'ютерів. Він має багато переваг, таких як зручний інтерфейс, широкий асортимент компонентів, автоматична перевірка сумісності, актуальні ціни та можливість збереження конфігурацій. Однак, як і будь-яка система, він має свої недоліки, зокрема обмеженість у налаштуваннях, відсутність деяких брендів, можливі технічні проблеми та певні складнощі для новачків. У цілому, використання цього конфігуратора може значно спростити та прискорити процес збирання ПК для більшості користувачів.

Після аналізу аналогічних систем, що є на ринку сформулюємо основні вимоги до нашої системи.

Існуючі конфігуратори, такі як ті, що представлені на сайтах CAN.ua та інших інтернет-магазинів, можуть не повністю відповідати всім специфічним вимогам нашого проєкту. Створення власного конфігуратора дозволяє впровадити унікальні функції, яких немає в інших конфігураторах, додати можливість гнучкого налаштування та індивідуалізації компонентів, дасть можливість швидкого реагування на потреби користувачів та ринкові зміни. Також можна провести інтеграцію з внутрішніми базами даних та системами управління товарними запасами.

Створення власного конфігуратора дозволяє нам сконцентруватися на покращенні користувацького досвіду:

– можливість впровадження персоналізованих рекомендацій на основі історії користувача;

– створення інтуїтивно зрозумілого та зручного інтерфейсу, орієнтованого на конкретну аудиторію;

– підтримка різних мов та локалізацій для користувачів з різних регіонів.

Розробка власного конфігуратора ПК є стратегічним рішенням, яке забезпечує гнучкість, контроль, безпеку та конкурентні переваги. Це дозволяє створити продукт, який максимально відповідає нашим потребам та вимогам, а також покращує користувацький досвід, що є ключовим фактором успіху на ринку.

## 1.2 Аналіз вимог до системи

Сформулюємо функціональні вимоги до інформаційної системи.

Розмежування прав доступу в межах інформаційної системи зазвичай здійснюється шляхом дозволу/заборони на перегляд певної інформації для певної групи користувачів. В системі, що розроблюється існують такі групи користувачів як: Адміністратор, Користувач та Гість (рис. 1.4).

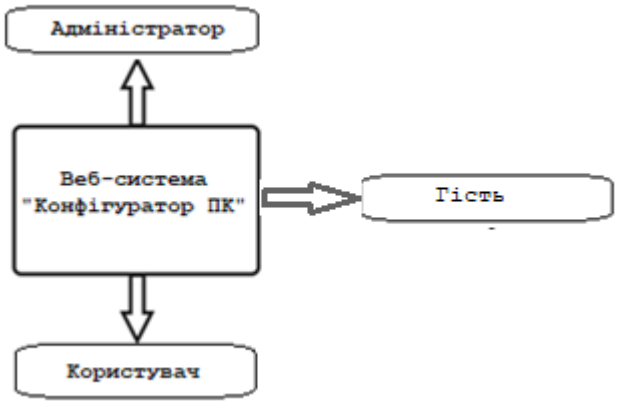


Рисунок 1.4 – Користувачі системи

Представимо функціональні вимоги у вигляді діаграми варіантів використання для відображення наочності, бо саме діаграма варіантів використання є наочним способом для демонстрації взаємодії акторів і функцій веб-системи.

Всі варіанти використання в системі визначають свій спосіб застосування та містять кожен свій алгоритм, наприкінці циклу якого від одного варіанту використання можна буде перейти до іншого.

Взаємодіяти з системою будуть два актори: адміністратор та користувач. Кожен актор включає кілька варіантів використання.

Адміністратор – співробітник магазину, який впроваджує новий товар під реалізацію, видаляє товар, що втратив актуальність, редагує асортимент комплектуючих у конфігураторі та відповідає на заявки користувачів.

Користувач – клієнт магазину, який може замовити готовий комп'ютер, зібрати його в конфігуратор та замовити.

Гість може лише продивлятися деяку інформацію у системі і не має змоги збирати потрібну конфігурацію або замовляти готову збірку.

Діаграма варіантів використання повинна включати можливість:

1) редагувати готове складання ПК – можливість для адміністратора створити, переглянути, редагувати та видалити будь-який комп'ютер із готових збірок;

2) редагування комплектуючих у конфігураторі – можливість для адміністратора створити, переглянути, редагувати та видалити будь-яке комплектуюче;

3) авторизуватись – можливість для користувача ввести свої дані для входу, щоб переглядати та користуватися повним функціоналом сайту;

4) зареєструватися – можливість для користувача, у разі відсутності облікового запису, необхідно ввести свої дані для реєстрації, а потім авторизуватися, щоб переглядати та користуватися повним функціоналом сайту;

5) відповідь на заявку користувача та видалення заявки – можливість для адміністратора дати відповідь користувачу на його заявку додати якесь комплектуюче, а також видалити заявку;

6) замовити готове складання ПК для покупки – можливість для користувача вибрати комп'ютер із вже представлених та зробити замовлення;

7) зібрати ПК у конфігураторі – можливість для користувача вибрати у розділі «Конфігуратор ПК» комплектуючі за відповідністю та зробити замовлення.

Діаграму прецедентів представимо на рис. 1.5.

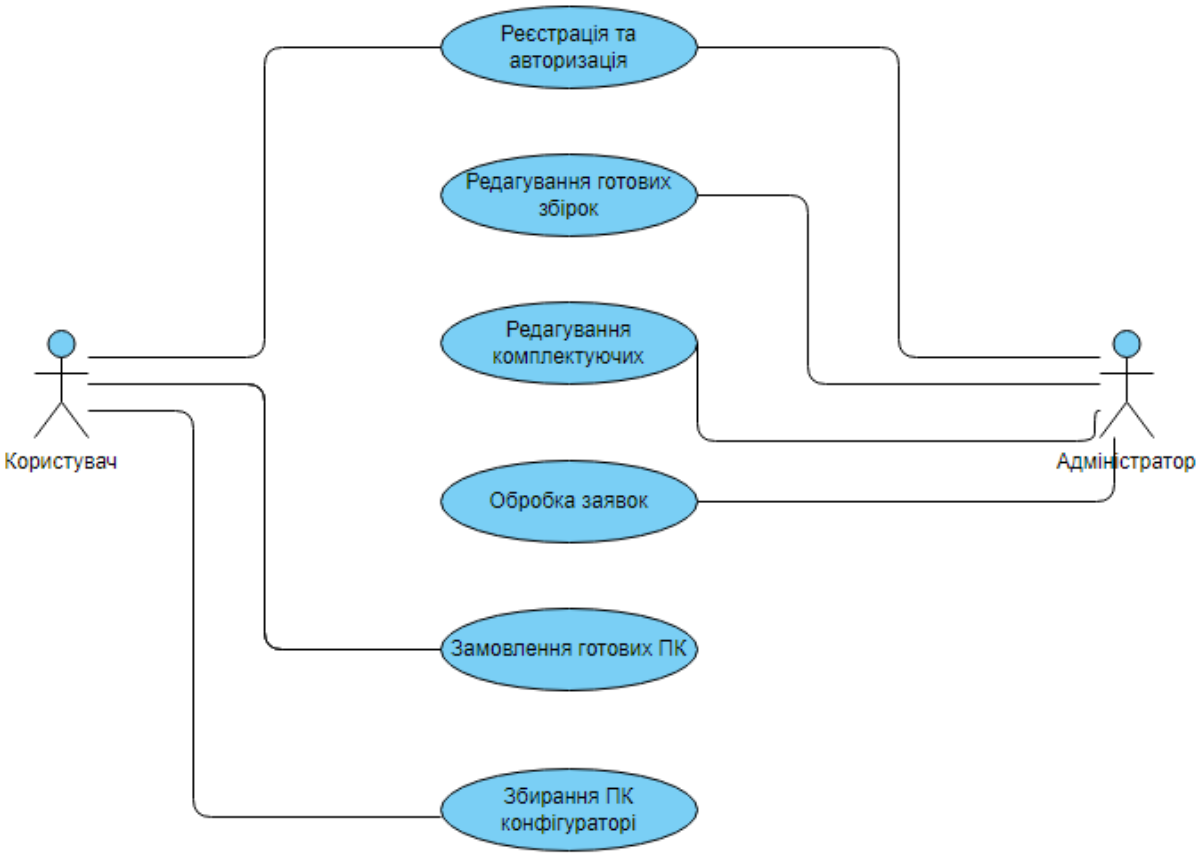


Рисунок 1.5 – Діаграма варіантів використання

Детально опишемо всі варіанти використання, їх сценарії.

Варіант використання: Редагування готових збірок

Опис: цей варіант використання передбачає можливість для адміністратора системи створювати, переглядати, редагувати та видаляти комп'ютери з готових збірок.

Актори: Адміністратор

Передумови: Адміністратор увійшов у систему з відповідними правами доступу.

Післяумови:

– зміни у готових збірках збережено в системі;

– адміністратор може бачити підтвердження успішного створення, редагування або видалення збірки.

Основний сценарій:

1. Адміністратор переходить до розділу «Готові збірки ПК».

2. Система відображає список існуючих збірок.

3. Адміністратор вибирає опцію «Створити нову збірку».

4. Система відкриває форму для створення нової збірки.

5. Адміністратор заповнює форму нової збірки (назва, компоненти, опис тощо).

6. Система перевіряє правильність введених даних.

7. Адміністратор натискає кнопку «Зберегти».

8. Система зберігає нову збірку та відображає повідомлення про успішне створення.

9. Адміністратор вибирає збірку зі списку для перегляду або редагування.

10. Система відображає деталі вибраної збірки.

11. Адміністратор вносить зміни до деталей збірки (компоненти, опис тощо).

12. Система перевіряє правильність введених даних.

13. Адміністратор натискає кнопку «Оновити».

14. Система зберігає внесені зміни та відображає повідомлення про успішне оновлення.

15. Адміністратор вибирає збірку зі списку для видалення.

16. Система запитує підтвердження видалення.

17. Адміністратор підтверджує видалення збірки.

18. Система видаляє збірку та відображає повідомлення про успішне видалення.

Альтернативний сценарій 1:

1а. Адміністратор не заповнив обов'язкові поля форми створення нової збірки. Система виводить повідомлення про необхідність заповнення всіх обов'язкових полів.

Альтернативний сценарій 2:

2а. Система не може знайти вибрану збірку для перегляду або редагування. Система виводить повідомлення про помилку та повертає адміністратора до списку збірок.

Альтернативний сценарій 3:

3а. Адміністратор відміняє підтвердження видалення збірки. Система не видаляє збірку та повертає адміністратора до списку збірок.

Варіант використання: Редагування комплектуючих

Опис: цей варіант використання передбачає можливість для адміністратора системи створювати, переглядати, редагувати та видаляти будь-які комплектуючі в конфігураторі ПК.

Актори: Адміністратор

Передумови: Адміністратор увійшов у систему з відповідними правами доступу.

Післяумови:

– зміни у комплектуючих збережено в системі;

– адміністратор може бачити підтвердження успішного створення, редагування або видалення комплектуючих.

Основний сценарій:

1. Адміністратор переходить до розділу «Комплектуючі».

2. Система відображає список існуючих комплектуючих.

3. Адміністратор вибирає опцію «Створити нове комплектуюче».

4. Система відкриває форму для створення нового комплектуючого.

5. Адміністратор заповнює форму нового комплектуючого (назва, тип, характеристики, опис тощо).

6. Система перевіряє правильність введених даних.

7. Адміністратор натискає кнопку «Зберегти».

8. Система зберігає нове комплектуюче та відображає повідомлення про успішне створення.

9. Адміністратор вибирає комплектуюче зі списку для перегляду або редагування.

10. Система відображає деталі вибраного комплектуючого.

11. Адміністратор вносить зміни до деталей комплектуючого (назва, характеристики, опис тощо).

12. Система перевіряє правильність введених даних.

13. Адміністратор натискає кнопку «Оновити».

14. Система зберігає внесені зміни та відображає повідомлення про успішне оновлення.

15. Адміністратор вибирає комплектуюче зі списку для видалення.

16. Система запитує підтвердження видалення.

17. Адміністратор підтверджує видалення комплектуючого.

18. Система видаляє комплектуюче та відображає повідомлення про успішне видалення.

Альтернативний сценарій 1:

1а. Адміністратор не заповнив обов'язкові поля форми створення нового комплектуючого. Система виводить повідомлення про необхідність заповнення всіх обов'язкових полів.

Альтернативний сценарій 2:

2а. Система не може знайти вибране комплектуюче для перегляду або редагування. Система виводить повідомлення про помилку та повертає адміністратора до списку комплектуючих.

Альтернативний сценарій 3:

3а. Адміністратор відміняє підтвердження видалення комплектуючого.

Система не видаляє комплектуюче та повертає адміністратора до списку комплектуючих.

Варіант використання: Обробка заявок

Опис: цей варіант використання передбачає можливість для адміністратора відповідати на заявки користувачів щодо додавання комплектуючих, а також видаляти ці заявки після обробки.

Актори: Адміністратор

Передумови: Адміністратор увійшов у систему з відповідними правами доступу.

Післяумови:

– користувач отримав відповідь на свою заявку;

– заявка видалена з системи після обробки.

Основний сценарій:

1. Адміністратор переходить до розділу «Заявки користувачів».

2. Система відображає список заявок від користувачів.

3. Адміністратор вибирає заявку для перегляду.

4. Система відображає деталі вибраної заявки (наприклад, запитане комплектуюче, повідомлення від користувача).

5. Адміністратор обробляє заявку та створює відповідь користувачу.

6. Система відображає форму для введення відповіді.

7. Адміністратор вводить відповідь на заявку (наприклад, підтвердження додавання комплектуючого або відмову з поясненням).

8. Система зберігає відповідь та надсилає її користувачу.

9. Адміністратор видаляє оброблену заявку.

10. Система запитує підтвердження видалення заявки.

11. Адміністратор підтверджує видалення заявки.

12. Система видаляє заявку та відображає повідомлення про успішне видалення.

Альтернативний сценарій 1:

1а. Система не може знайти вибрану заявку для перегляду.

Система виводить повідомлення про помилку та повертає адміністратора до списку заявок.

Альтернативний сценарій 2:

2а. Адміністратор залишає відповідь на заявку порожньою.

Система виводить повідомлення про необхідність заповнення відповіді.

Альтернативний сценарій 3:

3а. Адміністратор відміняє підтвердження видалення заявки.

Система не видаляє заявку та повертає адміністратора до списку заявок.

Нефункціональні вимоги, яким має відповідати веб-система:

1) веб-система повинна бути написана на PHP, не використовуючи CMS та конструкторів;

2) веб-система має використовуватися без плагінів;

3) веб-система має використовувати набір інструментів Boot-Strap;

4) час завантаження системи не повинен перевищувати 5 секунд;

5) час переходу між різними частинами застосунку не повинен перевищувати 3 секунди;

6) кількість відмов системи не повинна перевищувати 0.5% на рік;

7) використання багатофакторної автентифікації для доступу до особистих даних.

# 2 ПЛАНУВАННЯ РОЗРОБКИ ВЕБ-СИСТЕМИ

## 2.1 Декомпозиція робіт проєкту

Розподіл роботи на менші завдання є широко використовуваним методом для підвищення продуктивності. Цей підхід дозволяє зробити роботу більш контрольованою та доступною.

На рис. 2.1 показана діаграма Ганта. Це інструмент управління проєктами, який використовується для візуального відображення часового плану проєкту. Діаграма Ганта дозволяє визначити наступні аспекти:

– як проєкт розбивається на окремі завдання;

– коли кожне завдання починається і закінчується;

– скільки часу потрібно на виконання кожного завдання;

– кому призначене виконання кожного завдання;

– як завдання пов'язані між собою і взаємозалежать;

– як проєкт розвивається в часі;

– загальний графік проекту від початку до кінця.

Діаграма Ганта для створення системи планування навантаження представлена на рис.2.1.

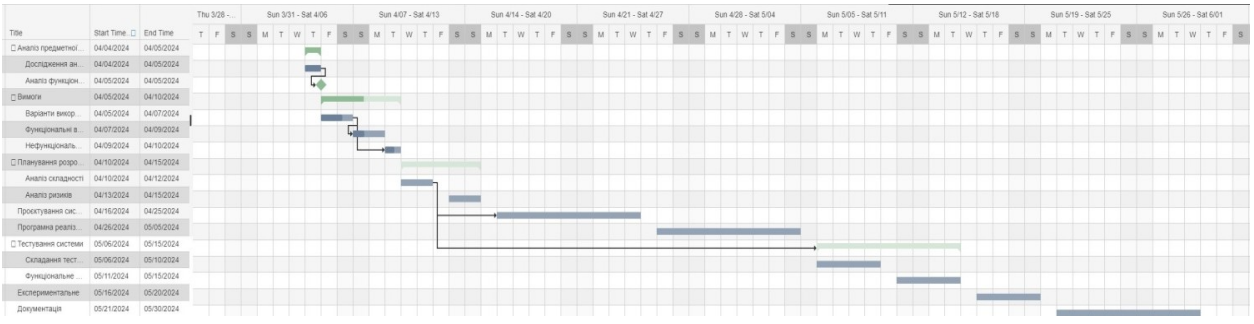


Рисунок 2.1 – Діаграма Ганта

На цій діаграмі представлена декомпозицію робіт по створенню системи. Починається робота над проектом з аналізу предметної області,

визначення вимог до системи, планування розробки, оцінка ризиків проекту, проектування, програмна реалізація та тестування системи.

## 2.2 Оцінка ризиків проєкту

Оцінка ризиків у процесі розробки програми є критично важливою для успішного завершення проекту. Деякі потенційні ризики можуть включати:

1. Неоднорідність даних: різноманітність форматів даних або неповна/неточна інформація вихідних джерел.

2. Обмеження часу та бюджету: недостатній час або фінансові ресурси можуть обмежити розробку програми та спричинити затримки у випуску.

3. Технічні складнощі: проблеми з сумісністю програмного забезпечення, недоліки в технічній інфраструктурі або складність архітектури можуть спричинити затримки та втрату продуктивності.

4. Брак спілкування: недостатня комунікація між розробниками, клієнтами та іншими зацікавленими сторонами може призвести до непорозумінь та неправильного сприйняття вимог.

5. Незадовільне тестування: недостатнє тестування може призвести до виявлення помилок та проблем вже після випуску програми, що зменшить її якість та довіру користувачів.

Для кожного ризику потрібно розробити стратегії вирішення проблем.

Аналізувати інформацію: перед початком розробки необхідно провести ретельний аналіз вихідних даних та їх форматів для визначення можливих проблем.

Ефективне спланувати: ретельне розпланування ресурсів, часу та бюджету може допомогти уникнути затримок та перевищення витрат.

Провести технічну експертизу: залучення кваліфікованих фахівців з досвідом у сфері розробки програмного забезпечення може допомогти вирішити технічні проблеми.

Підвищити комунікацію: регулярні зустрічі та звітності між командами розробників та клієнтами можуть покращити розуміння вимог та уникнути непорозумінь.

Тестування: проведення комплексного тестування на різних етапах розробки дозволить виявити та виправити проблеми ще до випуску продукту.

В табл. 2.1 наведено результати оцінки ймовірності виникнення різних ризиків.

Таблиця 2.1 – Оцінка ймовірності виникнення ризиків

| Номер | Ризик | Ймовірність |
|---|---|---|
| 1 | Відсутність досвіду розробки | 3 |
| 2 | Недостатність часу | 2 |
| 3 | Недостатні комунікації з керівником | 1 |

Таким чином, найнебезпечнішим ризиком є ризик «Відсутність досвіду розробки». Розробник може не мати достатнього досвіду роботи з обраними технологіями або в реалізації подібних проєктів. Це може призвести до помилок у коді, неефективних рішень, затримок у розробці та загального зниження якості програмного забезпечення. Необхідно знизити ризик, пов'язаний з відсутністю досвіду розробки, та забезпечити успішне завершення проєкту.

Стратегією вирішення проблеми може бути навчання і підвищення кваліфікації, пошук експертів, що допоможуть у цьому питанні, правильне планування проєкту також допоможе швидко зреагувати на цей ризик.

Також протягом розробки необхідно перевіряти якість програмного коду.

В табл. 2.2 представлена картка для ризику «Відсутність досвіду розробки» та описаний план реакції.

Таблиця 2.2 – Картка ризику «Відсутність досвіду розробки»

| | |
|---|---|
| Номер: 1 | Категорія: Організаційний |
| Причина: мала кількість виконаних розробок подібного рівня | Умови: недостатній практичний досвід |
| Наслідки: не допуск до захисту кваліфікаційної роботи | Вплив: порушення термінів виконання задач за графіком, недостатня реалізація варіантів використання |
| Ймовірність: 3 | Ступінь впливу: 3 |
| Близькість: дуже скоро | Ранг: 9 |
| Стратегія реагування: консультації з керівником роботи та іншими викладачами кафедри, що мають досвід розробки застосунків, використання готових шаблонів, бібліотек та фреймворків для прискорення розробки та зменшення кількості помилок, перевірка коду на якість. | |

В даному розділі складено виконана декомпозиція робіт проєкту, оцінена тривалість розробки, виходячи з складності акторів та вимог до програми, спланований термін виконання для кожної задачі, описані ризики, що можуть виникати в процесі роботи над створенням програмного застосунку та стратегії реагування на них.

# 3 ПРОЄКТУВАННЯ ВЕБ-СИСТЕМИ

## 3.1 Архітектура системи

В роботі була обрана архітектура «клієнт-сервер». Архітектура клієнт-сервер – це модель розподіленої системи, де функції та обов'язки системи розділені між клієнтом і сервером.

Клієнт – це програмне забезпечення або пристрій, що звертається до сервера для отримання послуг або ресурсів.

Сервер – це програмне забезпечення або пристрій, що надає послуги або ресурси, до яких можуть звертатися клієнти.

Приклад клієнт-серверної взаємодії представлено на рис. 3.1.

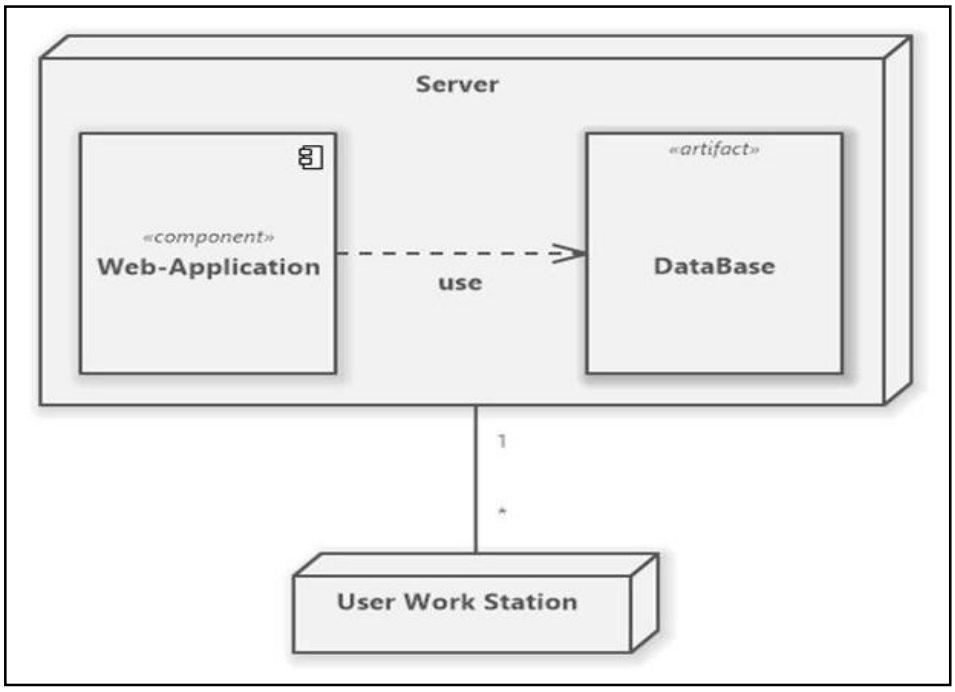


Рисунок 3.1 – Схема клієнт-серверної архітектури

У цій архітектурі клієнт та сервер взаємодіють через мережу, як правило, за допомогою протоколу передачі даних, такого як HTTP. Клієнт відправляє запити серверу, а сервер обробляє ці запити і відправляє відповіді назад клієнту.

У контексті веб-системи «Конфігуратор для збирання ПК», ця архітектура забезпечує ефективну взаємодію між користувачами (клієнтами) та сервером, який обробляє запити та керує даними.

Ця модель дозволяє розділити навантаження та функціональність між клієнтом і сервером, що дозволяє підвищити масштабованість, ефективність та безпеку системи.

Клієнтська частина може бути реалізована на будь-якому пристрої або платформі, тоді як серверна частина зазвичай знаходиться на віддаленому сервері з великими обчислювальними ресурсами.

Розлянемо детально компоненти клієнт-серверної архітектури.

1. Клієнтська сторона (Frontend):

– користувацький інтерфейс (UI): призначений для взаємодії з користувачем. Веб-браузер відображає інтерфейс конфігуратора ПК, який може бути створений за допомогою HTML, CSS та JavaScript;

– фреймворк: може використовуватися фреймворк, такий як React, Angular або Vue.js, для побудови інтерактивного та динамічного інтерфейсу;

– запити до сервера: клієнт надсилає HTTP-запити до сервера для отримання або оновлення даних (наприклад, доступні комплектуючі, ціни, характеристики).

2. Серверна сторона (Backend):

– веб-сервер: сервер обробляє HTTP-запити від клієнта. Популярними веб-серверами є Apache, Nginx або IIS;

– логіка додатка: серверна частина, написана на PHP, Node.js, Python або іншій серверній мові, обробляє запити, виконує бізнес-логіку та повертає відповіді клієнту;

– база даних: зберігає інформацію про комплектуючі, конфігурації ПК, користувачів та інші дані. MySQL або PostgreSQL можуть бути використані як СУБД.

Взаємодія між клієнтом та сервером:

1. Запитання конфігурації:

– користувач відкриває веб-додаток «Конфігуратор для збирання ПК» у своєму браузері;

– клієнтська сторона надсилає запит на сервер для отримання початкових даних, таких як список доступних комплектуючих.

2. Обробка запиту на сервері:

– веб-сервер отримує запит та передає його на обробку серверному додатку;

– серверний додаток взаємодіє з базою даних для отримання необхідних даних (наприклад, доступні процесори, материнські плати, оперативна пам'ять).

3. Відповідь сервера:

– сервер формує відповідь з необхідними даними та відправляє її назад клієнту у форматі JSON або XML;

– веб-сервер надсилає відповідь клієнту.

4. Відображення даних на клієнті:

– клієнтська частина обробляє отримані дані та відображає їх у зручному вигляді для користувача;

– користувач може взаємодіяти з інтерфейсом, вибирати комплектуючі та створювати конфігурації ПК.

5. Збереження конфігурації:

– коли користувач завершує конфігурацію ПК, він може зберегти її;

– клієнт надсилає запит на сервер для збереження конфігурації у базі даних;

– сервер обробляє запит, зберігає конфігурацію та надсилає підтвердження клієнту.

Переваги клієнт-серверної архітектури:

– модульність: розподіл функціональності між клієнтом та сервером забезпечує легкість у підтримці та масштабуванні системи;

– безпека: серверна частина захищена від прямого доступу користувачів, що дозволяє краще захистити дані та бізнес-логіку;

– масштабованість: сервер можна легко масштабувати для обробки великої кількості запитів, а клієнтська частина може бути оновлена без впливу на серверну;

– висока продуктивність: обробка даних на сервері забезпечує швидкий та ефективний доступ до великої кількості даних.

Таким чином, клієнт-серверна архітектура веб-системи «Конфігуратор для збирання ПК» забезпечує ефективну та безпечну взаємодію між користувачами та системою, що дозволяє створювати зручні та функціональні веб-додатки.

### 3.2 Діаграми потоків даних DFD

Для подальшого проєктування веб-системи будемо використовувати діаграми DFD [3].

DFD – діаграми потоків даних. Так називається методологія графічного структурного аналізу, що описує зовнішні по відношенню до системи джерела і адресати даних, логічні функції, потоки даних і сховища даних, до яких здійснюється доступ.

Діаграма потоків даних один з основних інструментів структурного аналізу і проектування інформаційних систем, що існували до широкого поширення UML. За допомогою DFD-діаграм вимоги до проектованої інформаційної системи розбиваються на функціональні компоненти (процеси) і представляються у вигляді мережі, пов'язаної потоками даних.

Джерела інформації (зовнішні сутності) породжують інформаційні потоки (потоки даних), які переносять інформацію до підсистем або процесів. Ті в свою чергу перетворюють інформацію і породжують нові потоки, які переносять інформацію до інших процесів або підсистем, накопичувачів даних або зовнішнім сутностей-споживачів інформації. Таким чином, основними компонентами діаграм потоків даних є: зовнішні сутності; системи/підсистеми; процеси; накопичувачі даних; потоки даних.

Розробимо контекстні DFD-діаграми для наших задач: «Авторизація та вхід в особистий кабінет» (рис.3.2).

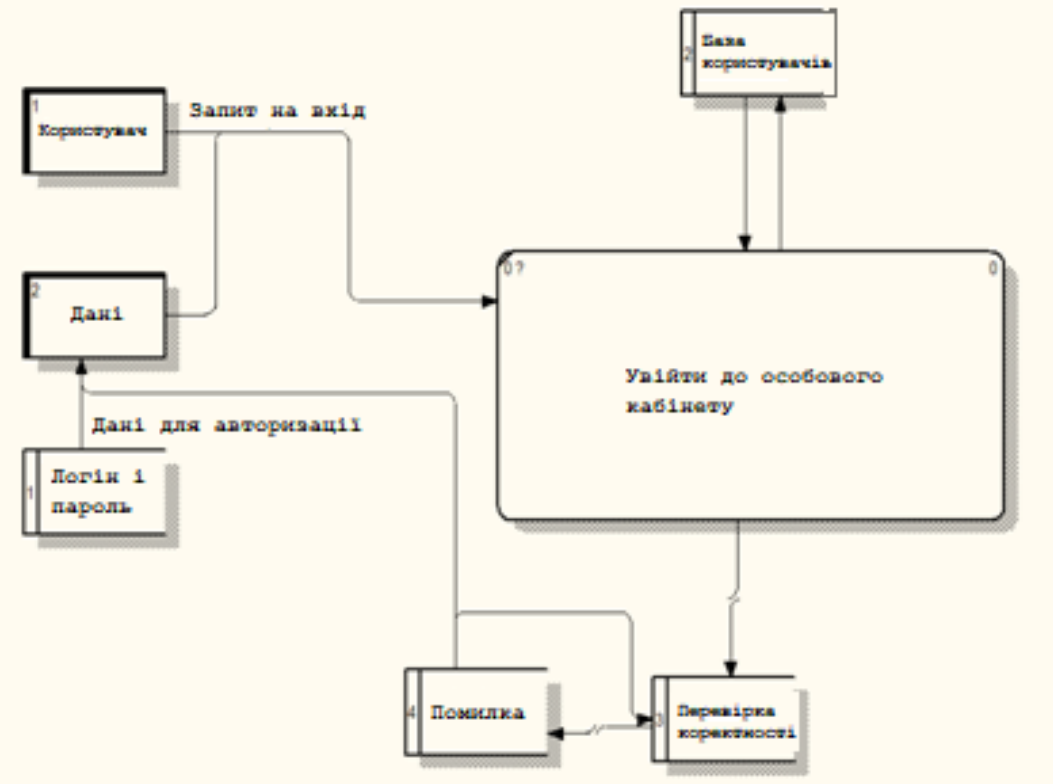


Рисунок 3.2 – Контекстна DFD –діаграма «Увійти в особистий кабінет»

Сутності:

1. Користувач – зовнішня сутність, яка взаємодіє із системою.

2. Система авторизації – частина внутрішньої системи, яка відповідає за перевірку облікових даних користувача.

Процес: процес авторизації – головний процес, що виконує авторизацію користувача в системі та надає доступ до особистого кабінету.

Потоки даних:

1. Облікові дані – логін і пароль, які користувач вводить для авторизації.

2. Підтвердження авторизації – відповідь від системи авторизації, що підтверджує успішність або невдачу авторизації.

3. Доступ до особистого кабінету – потік даних, що надає користувачу доступ до особистого кабінету після успішної авторизації.

Опис потоків даних:

1. Введення облікових даних (логін, пароль): користувач вводить свої облікові дані (логін і пароль) та надсилає їх до процесу авторизації.

2. Перевірка облікових даних: процес авторизації передає облікові дані до системи авторизації для перевірки їх коректності.

3. Підтвердження авторизації: система авторизації надсилає відповідь процесу авторизації, що підтверджує успішність або невдачу авторизації.

4. Доступ до особистого кабінету: у разі успішної авторизації користувач отримує доступ до свого особистого кабінету в системі.

Опис процесу авторизації:

1. Приймає облікові дані від користувача.

2. Передає ці дані системі авторизації для перевірки.

3. Отримує від системи авторизації підтвердження про успішність або невдачу авторизації.

У разі успішної авторизації надає користувачу доступ до особистого кабінету.

Ця контекстна DFD-діаграма забезпечує загальне уявлення про процес авторизації в системі та підкреслює взаємодію між користувачем та системою авторизації.

Контекстна DFD-діаграма «Оформлення замовлення» (рис.3.3) забезпечує загальне уявлення про процес оформлення замовлення у веб-системі та підкреслює взаємодію між користувачем, системою обробки замовлень, системою оплати та складом.

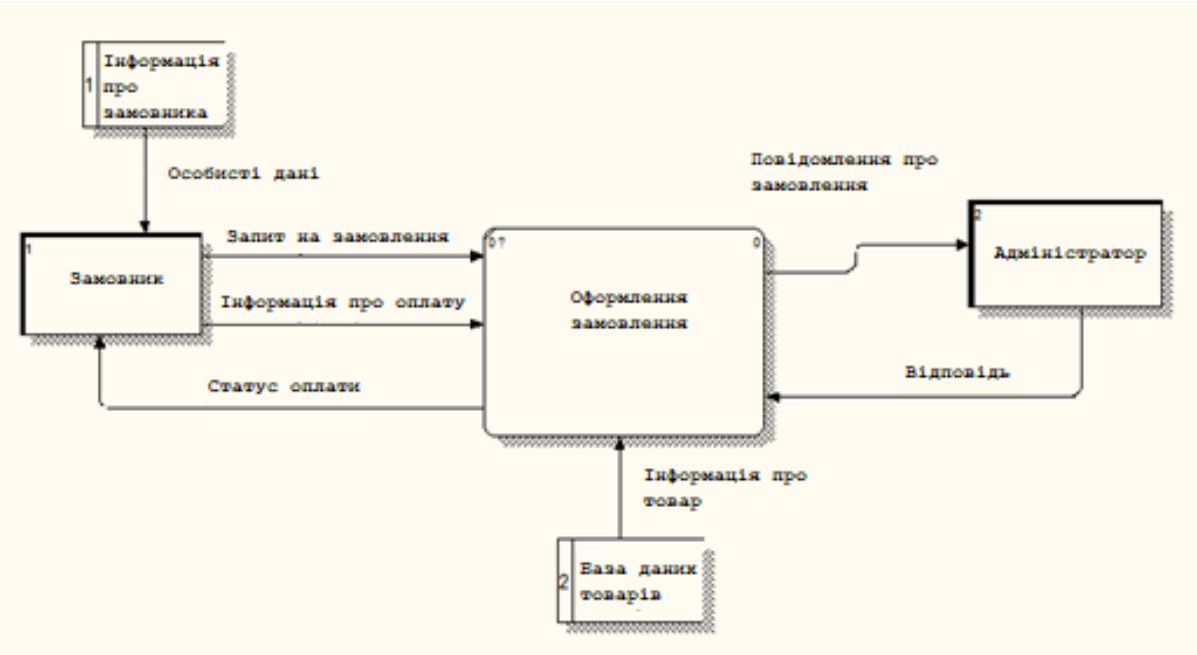


Рисунок 3.3 – Контекстна DFD –діаграма «Оформлення замовлення»

Контекстна DFD-діаграма для процесу «Оформлення замовлення» описує взаємодію між зовнішніми сутностями (акторами) та системою, зосереджуючи увагу на потоках даних. У цьому випадку йдеться про процес оформлення замовлення у веб-системі

### 3.3 Проєктування бази даних

Необхідність бази даних при створенні будь-якої веб-програми обумовлюється збереженням усіх необхідних даних в одному місці для подальшої обробки та зберігання інформації, доступ до якої можна отримати, зберігши її раніше. В системі, що розробляється доцільно створити наступні таблиці БД.

User – інформація про користувачів;

Admin – інформація про адміністраторів;

Zay – інформація про заявки від користувачів;

Conf – інформація про конфігурацію;

Socket – інформація про сокети;

Moth – інформація про материнські плати;

Processor – інформація про процесори;

Cooling – інформація про охолодження процесора;

Videocard – інформація про відео карти;

Oper – інформація про оперативну пам'ять;

Hard – інформація про жорсткі диски;

Ssd – інформація про твердотільні накопичувачі;

Block – інформація про блоки живлення.

На рис.3.4 зображено схему бази даних.

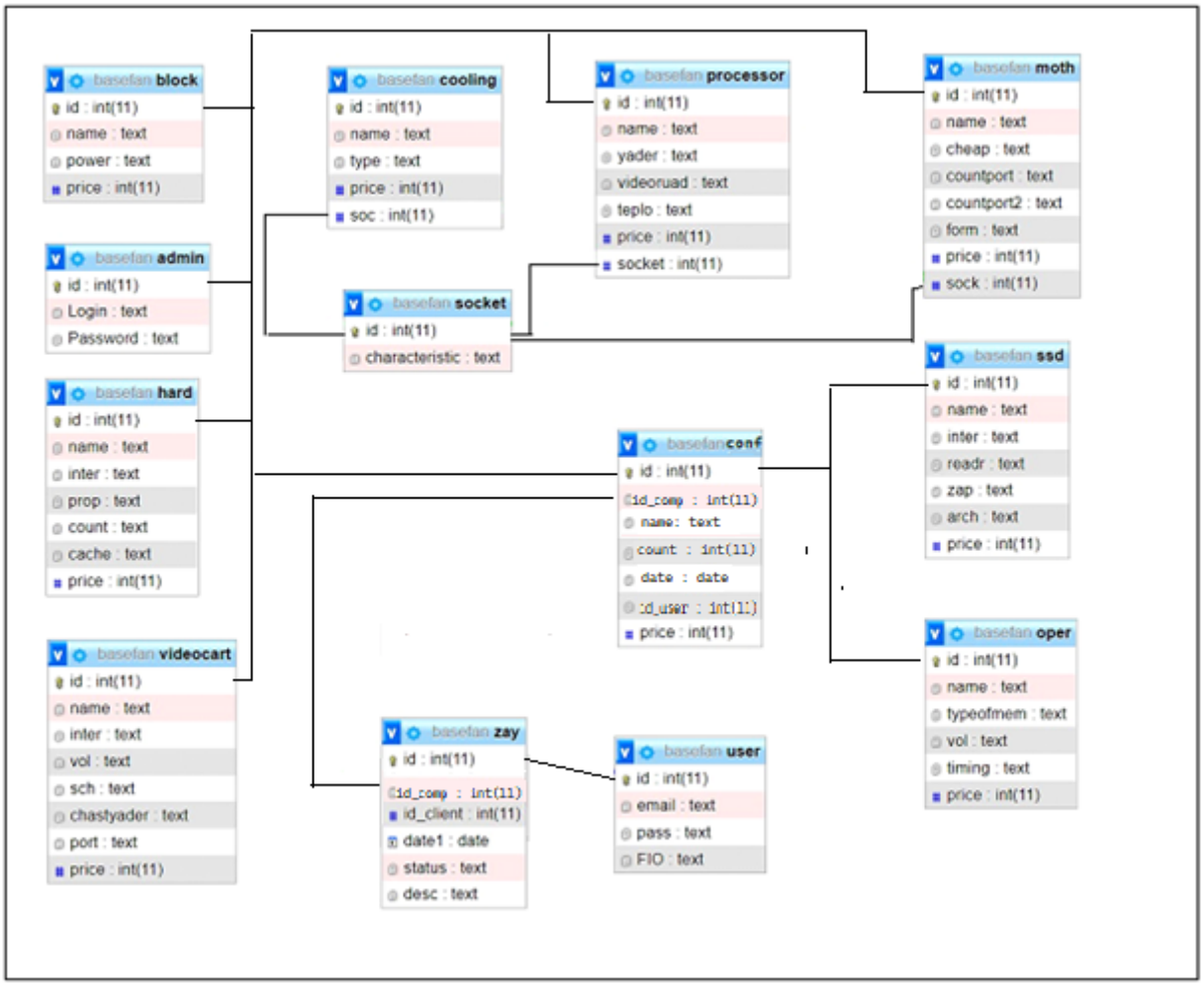


Рисунок 3.4 – Схема БД веб-системи «Конфігуратор ПК»

Наприклад, таблиця socket відповідає за відповідність такому важливому параметру, як сокет. Цей параметр фігурує в таблицях moth, processor і cooling. Його реалізація необхідна в першу чергу для недосвідчених користувачів, а саме для захисту від купівлі материнських плат, процесорів і систем охолодження процесорів, що не підходять один одному, відповідно.

### 3.4 Проєктування сторінок системи

Головне завдання системи – надати зручний графічний інтерфейс користувачам. Основні фактори, за допомогою яких можна оцінити або навіть виміряти зручність використання системи це – адекватність інтерфейсу, ефективність запобігання та подолання помилок користувачів, доступність, ефективність.

Адекватність користувальницького інтерфейсу програми – це його відповідність тим завданням, які користувачі повинні і хотіли б вирішувати з її допомогою.

Ефективність запобігання та подолання помилок користувачами тим краще, чим рідше користувачі помиляються при роботі з даним інтерфейсом і чим менше часу і зусиль потрібно для подолання наслідків вже зроблених помилок. Велике значення має також ризик, пов'язаний з виникненням помилки.

Розроблена система повинна бути настільки зрозумілою, щоб користувач, який ніколи раніше не бачив її, але добре розбирався в предметній області, міг без всякого навчання почати її використовувати. Крім того система не повинна перешкоджати ефективній роботі досвідчених користувачів, що працюють з нею довгий час. Одною з основних деталей, що впливають на враження користувача, є меню і можливості навігації, наявність зображень і організація елементів на сторінці. Меню повинні бути інтуїтивно зрозумілими і підкріплюватися навігаційними підказками.

Для розробки дизайну використані наступні принципи: проста навігація, поєднання кольорів, вирівнювання, спокійні кольори.

Схема надання інтерфейсу системи залежно від користувача представлена на рис.3.5.

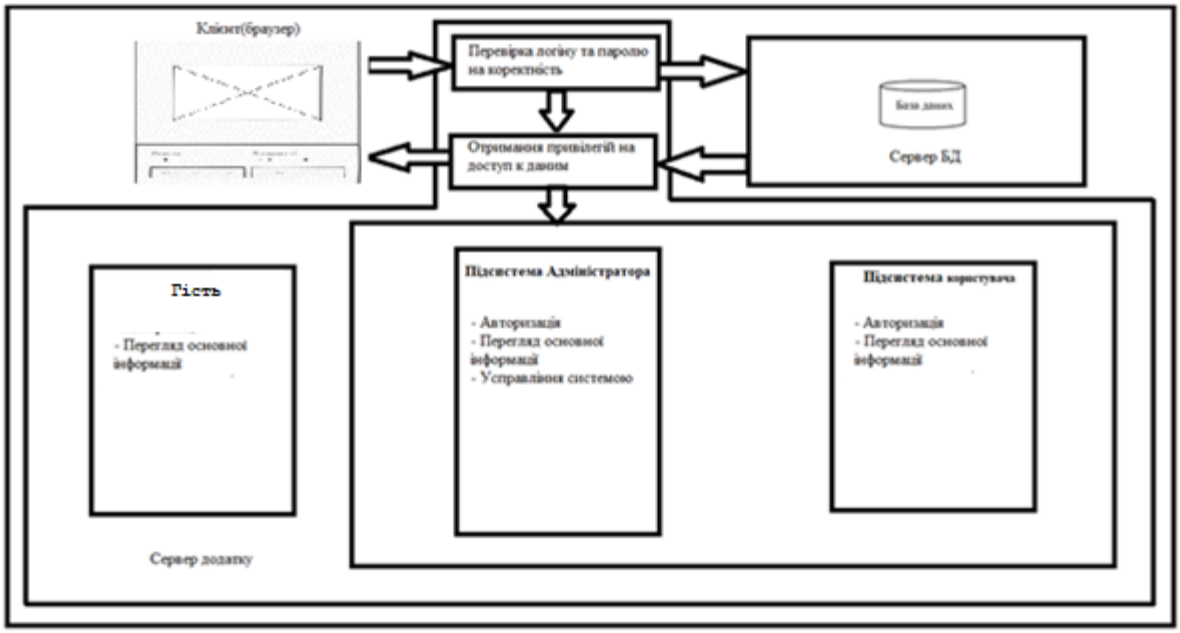


Рисунок 3.5 – Схема роботи веб-системи «Конфігуратор ПК»

Система надає наступні можливості:

– можливість реєстрації;
– можливість авторизації;
– перегляд основної інформації системи;
– перегляд особистого профілю;
– можливість редагування інформації ат заявок для адміністратора;
– можливість замовити готову збірку ПК;
– можливість конфігурувати ПК.

Планування структури веб-системи є важливим організаційним процесом, який впливає на якість, швидкість розробки, зручність використання та вартість. Добре структурована веб-система становить половину успіху при її створенні. Логічна структура веб-системи повинна

відображати, як інформація розподіляється по сторінках і як вона може бути доступна користувачеві.

Елементи локальної навігації візуально відокремлюються від елементів глобальної, але так, щоб вони, в теж час, виглядали як єдине ціле. Розроблено шаблон сторінок веб-системи «Конфігуратор ПК». Всі сторінки проектованої системи виконані в єдиному стилі. Щоб витримати стиль, спочатку був розроблений шаблон Головної сторінки (рис. 3.6).

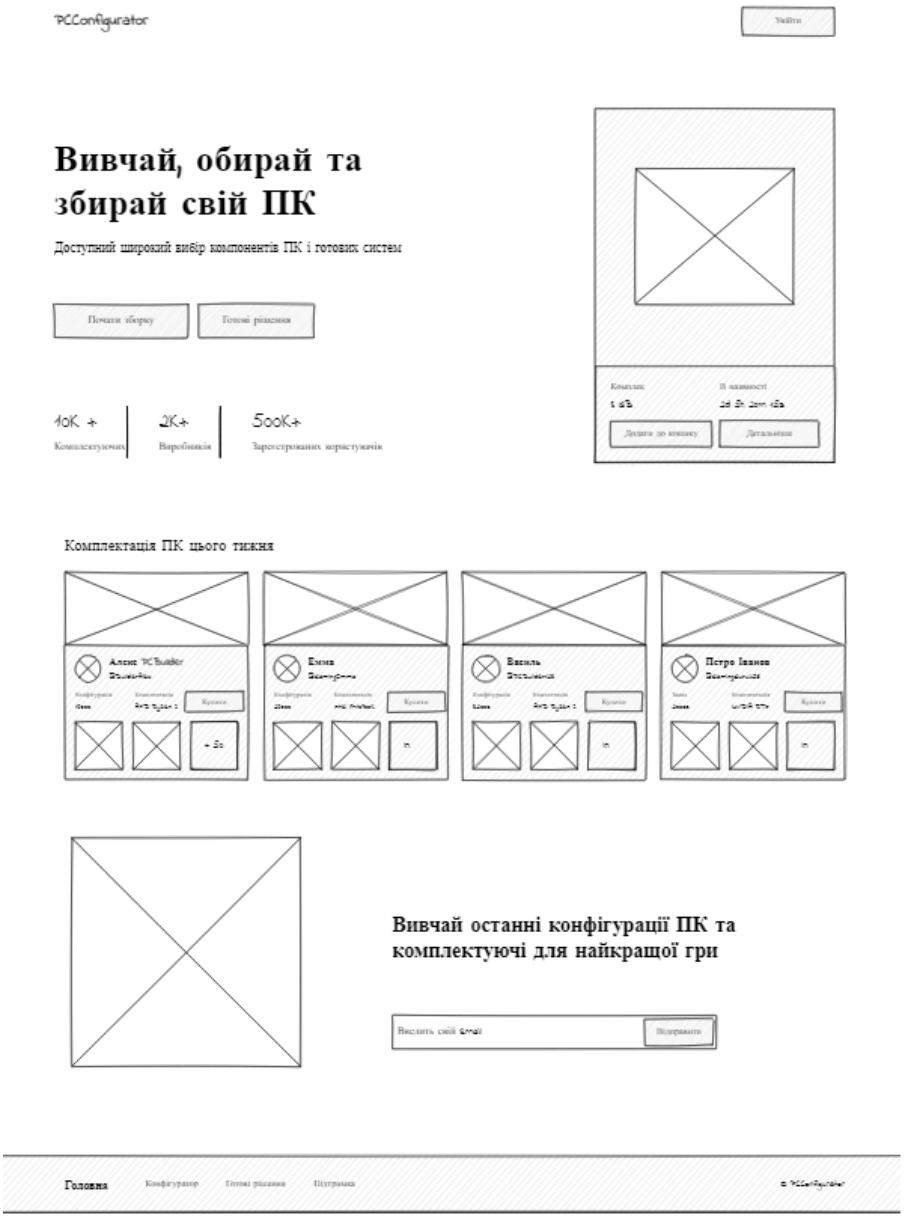


Рисунок 3.6 – Шаблон головної сторінки веб-системи «Конфігуратор ПК»

Навігація в інформаційній системі дозволяє відвідувачеві знаходити потрібну інформацію. Вона базується на логічній структурі системи і допомагає користувачеві швидко переміщатися по ній. Навігаційна система не повинна перевантажувати сторінку і відволікати від її змісту.

Шаблони зручні тим, що більшість сторінок верстають за подобою однієї сторінки майже автоматично. Структура шаблону складається з елементів, які повинні бути присутніми на всіх сторінках системи. Меню навігації розташовується в лівій частині сторінки. Всі сторінки системи являють собою область поділену на кілька частин. Вгорі сторінки знаходиться логотип і назва системи. Шаблон сторінки для замовлення та оплати товару наведемо на рис.3.7.

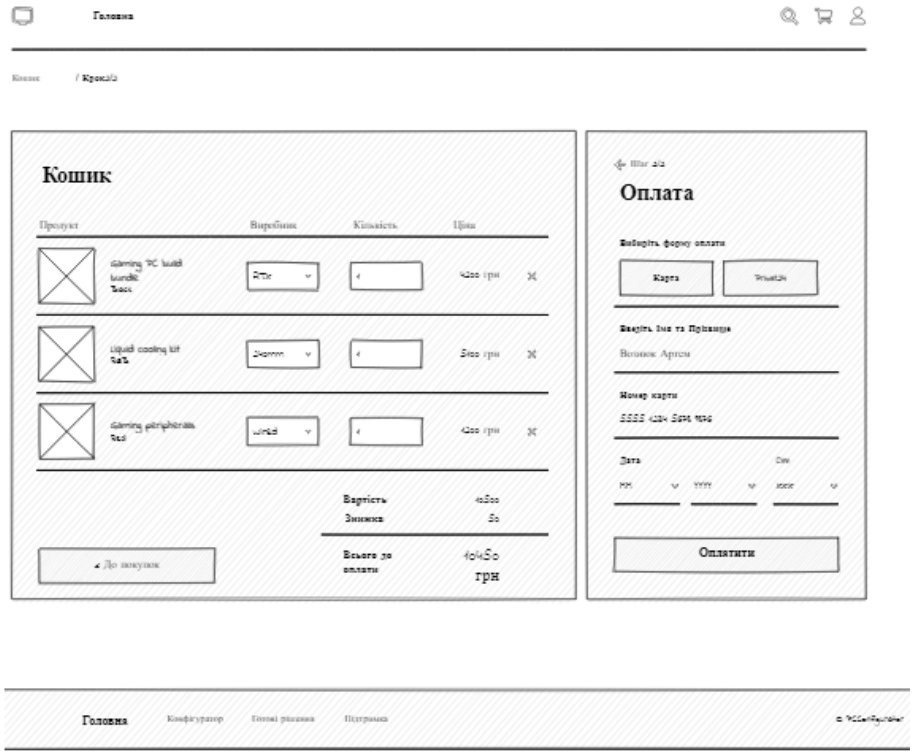


Рисунок 3.7 – Шаблон сторінки оплати товару веб-системи «Конфігуратор ПК»

Так буде виглядати замовлення у кошику та оплата вибраного товару.

# 4 ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-СИСТЕМИ

## 4.1 Засоби розробки

Для розробки системи було використано наступні засоби.

PHP – це популярна мова програмування, спеціально розроблена для веб-розробки. Вона дозволяє створювати динамічні веб-сторінки та взаємодіяти з базами даних. PHP [4] має широкий спектр застосувань, від простих скриптів до складних веб-додатків. Мова програмування JavaScript, тому що вона є однією з найпоширеніших мов програмування для створення інтерактивних веб-додатків. Вона дозволяє розробникам створювати динамічні та взаємодіючі елементи на веб-сторінках, що підвищує зручність користування. JavaScript [5] має велику кількість бібліотек і фреймворків, які спрощують розробку і скорочують час на реалізацію функціональності.

Таблиця стилів CSS [6] використовується для опису зовнішнього вигляду веб-сторінок. Вона дозволяє задавати стилі для HTML-елементів, такі як кольори, шрифти, відступи та інше, що дозволяє зробити інтерфейс користувача привабливим і зручним. Використання CSS забезпечує відокремлення структури HTML від її візуального оформлення, що спрощує розробку та підтримку коду.

Фреймворк Bootstrap [7] є популярним фреймворком для розробки адаптивних і мобільних веб-сайтів. Він містить готові компоненти, такі як навігаційні панелі, форми, кнопки та інші елементи інтерфейсу, що дозволяє швидко створювати сучасні та зручні веб-додатки. Використання Bootstrap знижує час розробки та забезпечує сумісність з різними пристроями і браузерами.

MySQL – це одна з найпопулярніших систем управління реляційними базами даних. Вона використовується для зберігання та обробки даних в різних веб-додатках, від простих веб-сайтів до складних систем управління контентом та електронної комерції [8].

Обрання цих засобів для розробки системи обумовлене їхньою популярністю, надійністю, продуктивністю та здатністю забезпечити всі необхідні функції для створення сучасного і зручного застосунку.

Розглянемо трохи ширше ці програмні засоби.

Середовище Visual Studio Code [9] – це інтегроване середовище розробки (IDE), яке надає розширені можливості для написання коду, налагодження програм та співпраці з іншими розробниками. Воно широко використовується серед програмістів для роботи з різноманітними мовами програмування та технологіями.

Основні характеристики VS Code включають:

– розширення та підтримка мов програмування: VS Code підтримує багато мов програмування, включаючи JavaScript, Python, Java, HTML/CSS, C#, PHP та інші. Ви можете легко встановлювати розширення для підтримки додаткових мов та функціоналу;

– швидка навігація та редагування коду: VS Code надає зручні інструменти для швидкого переходу між файлами, редагування тексту та автоматичного завершення коду;

– інтегровані інструменти для налагодження: Ви можете використовувати вбудовані засоби для налагодження програм, встановлення точок зупинки, перегляду значень змінних та відстеження виконання коду;

– вбудована підтримка Git: VS Code має вбудовану підтримку Git, що дозволяє легко взаємодіяти з репозиторіями, стежити за змінами, комітувати та опубліковувати код;

– розширені функції співпраці: Ви можете використовувати розширення для спільної роботи з іншими розробниками, спільного перегляду та редагування коду у реальному часі;

– кросплатформеність: VS Code підтримується на різних операційних системах, включаючи Windows, macOS та Linux, що робить його універсальним інструментом для розробки на різних платформах.

Усі ці характеристики роблять Visual Studio Code потужним та універсальним інструментом для розробників будь-якого рівня.

Мова програмування JavaScript [5] є однією з найпоширеніших мов у світі веб-розробки. Вона використовується для створення інтерактивних веб-сторінок та веб-додатків.

Ось деякі ключові характеристики JavaScript:

– клієнтська мова програмування: JavaScript виконується на браузерах користувачів і використовується для створення динамічного веб-змісту. Вона дозволяє створювати інтерактивні елементи, анімацію, обробляти події та багато іншого, що робить веб-сайти більш динамічними та привабливими для користувачів;

– синтаксис: синтаксис JavaScript подібний до синтаксису C і Java, що робить його зрозумілим для багатьох програмістів. Він має структури даних, такі як масиви та об'єкти, оператори, умовні конструкції, цикли та інші стандартні елементи програмування;

– динамічна типізація: JavaScript є мовою з динамічною типізацією, що означає, що типи даних визначаються автоматично на основі значень, які вони мають. Це робить код більш гнучким, але вимагає уважності під час написання програм;

– події та обробники подій: JavaScript дозволяє реагувати на різні події, які виникають на сторінці, такі як клік мишею, натискання клавіш, завантаження сторінки тощо. Обробники подій виконуються певним кодом, коли відбувається певна подія;

– розширюваність: JavaScript є дуже розширюваною мовою за допомогою багатьох бібліотек і фреймворків, таких як React, Angular, Vue.js тощо. Ці інструменти дозволяють розробникам створювати складні веб-додатки швидко і ефективно;

– платформонезалежність: JavaScript підтримується на різних платформах, включаючи браузери, сервери (за допомогою Node.js), мобільні

пристрої та додатки настільних комп'ютерів.

JavaScript використовується в широкому спектрі сценаріїв, від розробки веб-додатків до розробки ігор та мобільних додатків, і відтак володіє великим потенціалом для розробників у різних галузях програмування.

Основні особливості PHP:

– серверна обробка: PHP працює на сервері, що дозволяє виконувати скрипти до того, як веб-сторінка буде відправлена клієнту;

– інтеграція з HTML: PHP легко вбудовується в HTML-код, що робить його зручним для створення динамічних веб-сторінок;

– підтримка баз даних: PHP підтримує роботу з багатьма системами управління базами даних (СУБД), включаючи MySQL, PostgreSQL, SQLite та інші;

– розширюваність: PHP має велику кількість розширень та бібліотек, які додають додаткову функціональність (наприклад, робота з графікою, XML, криптографія тощо);

– кросплатформеність: PHP працює на різних операційних системах, таких як Windows, Linux, macOS.

Мова розмітки HTML (Hypertext Markup Language) використовується для створення структури та вмісту веб-сторінок.

Ось деякі ключові аспекти мови розмітки HTML:

– структура документа: HTML використовує теги для визначення структури документа, такі як <html>, <head>, <title>, <body> тощо. Ці теги допомагають визначити початок і кінець документа, заголовок, тіло та інші елементи;

– теги та елементи: HTML містить різні теги, які використовуються для визначення різних елементів на сторінці, такі як текст, зображення, посилання, таблиці, форми тощо;

– атрибути: теги можуть містити атрибути, які надають додаткову

інформацію про елементи. Наприклад, атрибут src в тегі <img> вказує на шлях до зображення, а атрибут href в тегі <a> вказує на URL посилання;

– вкладеність: HTML дозволяє вкладати один елемент у інший, щоб створити складнішу структуру. Наприклад, ви можете помістити таблицю всередину дів-елемента, або список всередину параграфа;

– семантика: HTML також надає можливість додати семантику до веб-сторінок за допомогою спеціальних елементів, які вказують на значення різних частин сторінки, таких як заголовки, параграфи, списки, таблиці тощо;

– сумісність із браузерами: HTML є стандартом для створення веб-сторінок і підтримується всіма сучасними браузерами. Вона забезпечує кросбраузерну сумісність, що дозволяє відображати веб-сторінки на різних платформах і пристроях однаково.

HTML є однією з основних мов веб-розробки і є важливим інструментом для будь-якого веб-розробника. Вона надає можливість створювати структуру та вміст веб-сторінок, що є ключовим для створення візуально привабливих та функціональних веб-додатків.

Таблиця стилів CSS (Cascading Style Sheets) використовується для визначення зовнішнього вигляду веб-сторінок і елементів HTML.

Ось деякі ключові аспекти таблиць стилів CSS:

– селектори: CSS використовує селектори для вибору елементів HTML, до яких будуть застосовані стилі. Селектори можуть бути елементами HTML, класами, ідентифікаторами або псевдо класами;

– властивості і значення: властивості CSS визначають зовнішній вигляд елементів, такий як колір, розмір, шрифт, відступи, рамки тощо. Кожній властивості CSS може бути призначене одне або більше значень;

– каскадність і спадкування: CSS працює на основі принципу каскаду, що дозволяє визначати пріоритетність стилів. Стилі можуть успадковуватися від батьківських елементів, а також бути перезаписані більш конкретними стилями;

– класи і ідентифікатори: CSS дозволяє використовувати класи і ідентифікатори для застосування стилів до груп елементів або конкретних елементів на сторінці;

– позиціонування: CSS надає можливість керувати розташуванням елементів на сторінці за допомогою властивостей, таких як position, float, display тощо;

– адаптивність і респонсивний дизайн: CSS дозволяє створювати адаптивний веб-дизайн, що адаптується до різних розмірів екранів і пристроїв. Це досягається за допомогою медіа-запитів і властивостей, таких як flexbox та grid layout;

– анімація і переходи: CSS дозволяє створювати анімації та переходи для різних елементів і їх властивостей, що дозволяє створювати більш динамічний та привабливий вигляд веб-сторінок.

– Використання таблиць стилів CSS дозволяє розширити можливості веб-розробки, надаючи засоби для створення привабливого та функціонального вигляду веб-сторінок і додатків.

Bootstrap [7] – це потужний фреймворк для розробки веб-сайтів та веб-додатків. Ось кілька ключових характеристик Bootstrap:

– веб-респонсивний дизайн: Bootstrap пропонує готові компоненти та сітки, що дозволяють створювати веб-сайти, які адаптуються до різних розмірів екранів, від мобільних пристроїв до настільних комп'ютерів;

– готові компоненти: Bootstrap містить велику кількість готових компонентів, таких як кнопки, форми, картки, навігаційні панелі, таблиці, модальні вікна та багато іншого. Ці компоненти дозволяють швидко будувати структуру та функціонал веб-сторінок;

– споживання веб-типографії: Bootstrap надає широкий вибір вбудованих стилів для тексту та шрифтів, що дозволяє легко стилізувати текстовий контент на веб-сторінці;

– кросбраузерна сумісність: Bootstrap забезпечує сумісність з різними

веб-браузерами, що дозволяє створювати веб-сайти, які працюють однаково на всіх популярних браузерах;

– можливості налаштування: Bootstrap дозволяє налаштовувати тему та стиль за допомогою Sass-змінних та CSS-правил, що дозволяє створювати унікальний вигляд для веб-сайтів;

– розширення за допомогою плагінів: Bootstrap має велику спільноту розробників, яка надає різноманітні плагіни та розширення для різних функціональних можливостей, таких як каруселі, модальні вікна, перетягування та інші.

Bootstrap – це потужний інструмент для швидкого розроблення сучасних і адаптивних веб-сайтів, який допомагає веб-розробникам ефективно використовувати час та ресурси при створенні веб-проєктів.

Основні особливості MySQL:

– швидкість та продуктивність: MySQL забезпечує високу продуктивність та швидкість обробки запитів до бази даних;

– підтримка великих обсягів даних: MySQL здатна обробляти великі обсяги даних, що робить її ідеальною для великих веб-додатків;

– гнучкість: MySQL підтримує різні типи даних та дозволяє створювати складні запити для отримання необхідної інформації;

– безпека: MySQL забезпечує високий рівень безпеки даних, включаючи підтримку SSL-з'єднань та можливість налаштування прав доступу до бази даних;

– кросплатформеність: MySQL працює на різних операційних системах, таких як Windows, Linux, macOS.

Комбінація PHP та MySQL є однією з найпопулярніших для створення динамічних веб-додатків. PHP забезпечує обробку логіки на сервері та взаємодію з користувачем, тоді як MySQL зберігає дані та забезпечує їх швидкий доступ.

Використання PHP та MySQL забезпечує ефективне та надійне рішення для створення різноманітних веб-застсунків, що відповідають сучасним вимогам до функціональності та продуктивності.

Таким чином, за допомогою JavaScript, SheetJS та MS SQL Server відбувається парсинг робочого навчального плану, формування індивідуального навантаження науково-педагогічного працівника, а за допомогою HTML/CSS та JavaScript створюється інтерфейс програми.

Перед розробкою веб-програми необхідно встановити кроссплатформну збірку локального сервера Xampp. Разом з ним встановлюється утиліта phpMyAdmin [8] для адміністрування СУБД, а також безпосередньо самі СУБД. Перед встановленням необхідно завантажити Xampp з офіційного сайту, його шлях встановлення за замовчуванням відповідає C:\xampp. З переліку встановлених СУБД було обрано MySQL. Для запуску потрібно запустити дві програми, що знаходяться в директорії «xampp_start.exe», а потім «xampp-control.exe».

### 4.2 Реалізація функцій системи

На початку розробки системи необхідно створити головний файл index.php, який містить стандартну html структуру, а також реалізувати інтерфейси користувача та адміністратора. Файл style.css містить стилі проєкту. Потім додати фреймворк Bootstrap, який надає величезну кількість готових стилів. Спочатку до вже готової стандартної html структури, підключаємо готовий стиль через фреймворк Bootstrap:

```
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/boot
-    strap.min.css"    rel="stylesheet"    integrity="sha384-
1BmE4kWBq78iYhFldvKuhfT vZ6jIW3" cros- sorigin ="Artem">
```

Далі необхідно реалізувати на головній сторінці автентифікацію користувачів та адміністраторів. Наведемо лістинг коду аутентифікації, де в

залежності від процесу аутентифікації відбувається перемикання на сторінку користувача або адміністратора:

```
<?php
if ($_SESSION['admin']==true)
{
require_once('admin.php');
}
if ($_SESSION['name']==true)
{
require_once('user.php');
}?>
<script type="text/javascript"
src="//cdn.jsdelivr.net/npm/slick-carou-
sel@1.8.1/slick/slick.min.js"></script>
<?php if (($_SESSION['name']!=true) AND
(($_SESSION['admin']!=true)))
{
?>
```

Якщо користувача ще немає в базі даних, то він може пройти процедуру реєстрації:

```
if (isset($_GET['ti']))
{
$new_FIO = $_POST['name'];
$new_email = $_POST['email'];
$new_pass = $_POST['password'];
$mysqli = @new mysqli('localhost', 'root', '', 'basefan');
if (mysqli_connect_errno()) {
echo "Підключення неможливе: ".mysqli_connect_error();
}
$result_set = $mysqli->query("INSERT INTO `user` VALUES
(NULL,'$new_email','$new_pass','$new_FIO')");
?><script>document.location.href="index1.php"</script><?php
}
```

У тих випадках, коли користувач має обліковий запис, він може пройти процедуру авторизації:

```
if (isset($_GET['auth']))
{
$email = $_POST['email'];
$pass = $_POST['password'];
$d = false;
$mysqli = @new mysqli('localhost', 'root', '', 'basefan');
if (mysqli_connect_errno()) {
echo "Підключення неможливе: ".mysqli_connect_error();
```

```
     }
     $result_set = $mysqli->query('SELECT * FROM user WHERE
     email="'.$email.'"');
     $row = mysqli_fetch_array($result_set);
     $result_set1 = $mysqli->query('SELECT * FROM admin WHERE
Login="'.$email.'"');
     $row1 = mysqli_fetch_array($result_set1);
     }
     else{ ?><script>alert("Логін неправильний!");</script><?php
}
```

Крім необхідних процедур початкової веб-сторінки, також реалізовано перевірку на введення невірних і порожніх даних:

```
     if($email==$row1['Login'] and $login !== '')
     {
     if($pass==$row1['Password']){
     $_SESSION['admin'] = $email;
     ?><script>document.location.href="index1.php"</script><?php
     }
     else{?><script>alert('Пароль неправильний!');</script><?php
} } if($email==$row['email'] and $login !== '')
     {
     if($pass==$row['pass'])
     {
     $_SESSION['name'] = $email;
     $_SESSION['id'] = $row['id'];
     $_SESSION['name1'] = $row['FIO'];
     ?><script>document.location.href="index1.php"</script><?php
     }                                  else{?><script>alert('Пароль
неправильний!');</script><?php }
     }
```

Після процедури автентифікації необхідно перевести користувача або адміністратора на відповідні особисті кабінети. Нведемо фрагмент коду реалізації конфігуратора ПК:

```
     <input type="submit" value="Продовжити процес вибору">
     </form>
     <?php
     if (isset($_GET['comp']))
     {
     $proc = $_POST['proc'];
     $result_set21 = $mysqli->query("SELECT * FROM processor
WHERE id=$proc");
     $row21 = mysqli_fetch_array($result_set21);
     $_SESSION['procname'] = $row21['name'];
     $_SESSION['procprice'] = $row21['price'];
     $ sock = $ row21 ['socket'];
```

```
     ?><form           action="?add_comp"           method="post"
enctype="multipart/form-data"
     class="tab-form">
     Назва комп'ютера:
     <input type="text" name="name">
     Відеокарта:   <select   name="vid"><?php   $result_set1   =
$mysqli-
     >query('SELECT * FROM videocart');
     while($row1 = mysqli_fetch_array($result_set1)){
     $value=$row1['name']."#".$row1['price'];
     ?><option       value="<?=       $value;?>"><?php       echo
($row1['name']);?></op- tion><?php}
```

У тих випадках, коли користувач має обліковий запис, він може пройти процедуру авторизації:

```
     if (isset($_GET['auth']))
     {
     $email = $_POST['email'];
     $pass = $_POST['password'];
     $d = false;
     $mysqli = @new mysqli('localhost', 'root', '', 'basefan');
if (mysqli_connect_errno()) {
     echo "Підключення неможливе: ".mysqli_connect_error();
     }
     $result_set = $mysqli->query('SELECT * FROM user WHERE
     email="'.$email.'"');
     $row = mysqli_fetch_array($result_set);
     $result_set1 = $mysqli->query(' SELECT * FROM admin WHERE
Login="'.$email.'"');
     $row1 = mysqli_fetch_array($result_set1);
     }
     else{ ?><script>alert("Логін неправильний!");</script><?php
}
```

Крім необхідних процедур початкової веб-сторінки, також реалізовано перевірку на введення невірних і порожніх даних:

```
     if($email==$row1['Login'] and $login !== '')
     {
     if($pass==$row1['Password']){
     $_SESSION['admin'] = $email;
     ?><script>document.location.href="index1.php"</script><?php
     }
     else{?><script>alert('Пароль  неправильний!');</script><?php
} } if($email==$row['email'] and $login !== '')
     {
     if($pass==$row['pass'])
     {
     $_SESSION['name'] = $email;
     $_SESSION['id'] = $row['id'];
     $_SESSION['name1'] = $row['FIO'];
```

```
     ?><script>document.location.href="index1.php"</script><?php
     }                                  else{?><script>alert('Пароль
неправильний!');</script><?php }
     }
```

Після процедури автентифікації необхідно перевести користувача або адміністратора на відповідні особисті кабінети:

```
     <input type="submit" value="Продовжити процес вибору">
     </form>
     <?php
     if (isset($_GET['comp']))
     {
     $proc = $_POST['proc'];
     $result_set21  =  $mysqli->query("SELECT  *  FROM  processor
WHERE id=$proc");
     $row21 = mysqli_fetch_array($result_set21);
     $_SESSION['procname'] = $row21['name'];
     $_SESSION['procprice'] = $row21['price'];
     $ sock = $ row21 ['socket'];
     ?><form           action="?add_comp"           method="post"
enctype="multipart/form-data"
     class="tab-form">
     Назва комп'ютера:
     <input type="text" name="name">
     Відеокарта:   <select   name="vid"><?php   $result_set1   =
$mysqli-
     >query('SELECT * FROM videocart');
     while($row1 = mysqli_fetch_array($result_set1)){
     $value=$row1['name']."#".$row1['price'];
     ?><option       value="<?=       $value;?>"><?php       echo
($row1['name']);?></op- tion><?php}
```

Також потрібно реалізувати заявки zay від користувача до адміністратора:

```
     <?php
     $result_set   =   $mysqli->query('SELECT   *   FROM   zay');
while($row = mysqli_fetch_array($result_set)){
     ?><tr> <td><p>
     <?php
     $result_set12  =  $mysqli->query('SELECT  *  FROM  user  WHERE
id="'.$row['id_client'].'"');
     $row12   =   mysqli_fetch_array($result_set12);   print_r("
".$row12['FIO']);?></br>
     <?php print_r(" ".$row12['email']);?>
     </p> </td> <?php
     ?><td><p><?php                                    print_r("
".$row['desc']);?></p></td><?php?><td><p><?php     print_r     ("
".$row['date1']);?></p></td>
     <td><p><?php print_r(" ".$row['status']);?></p></td>
```

```
    <td><p><form class="tab-form" method="post" action="?stat">
    <input type="text" name="status" placeholder="Введіть новий
статус замовлення" width="20px;"/>
    <input      type="hidden"      name="id"      value="<?php
echo($row['id']);?>"/>
    <input type="submit" value="Змінити статус">
    </form></p></td>
```

Таким чином, були описані основні функції веб-системи з використанням обраних засобів розробки.

### 4.3 Керівництво користувача

Після запуску системи користувач потрапляє на головну сторінку (рис.4.1). Без авторизації є можливість лише проглядати новини та наявні комплектуючі.

Основне меню розташоване внизу сторінки. Розглянемо більш детально всі його пункти.

Пункт «Головна» відображає головну сторінку системи. На цій сторінці може розташовуватися інформація та реклама. Також відображається статистика про збірки і покупки в даній системі.

Пункт «Конфігуратор» дасть змогу зібрати потрібний ПК з наявними комплектуючими.

Пункт «Готові рішення» покаже всі збірки ПК, що були додані адміністратором, є в наявності і можуть бути замовлені.

Пункт «Підтримка» видає дані для зв’язку з системою.

Щоб мати доступ до всіх функцій системи, необхідно авторизуватись, натиснувши кнопку «Увійти».

В систему авторизований користувач можу увійти як Користувач так і Адміністратор. В залежності від ролі відповідно і інтерфейс та функціональність будуть різними.

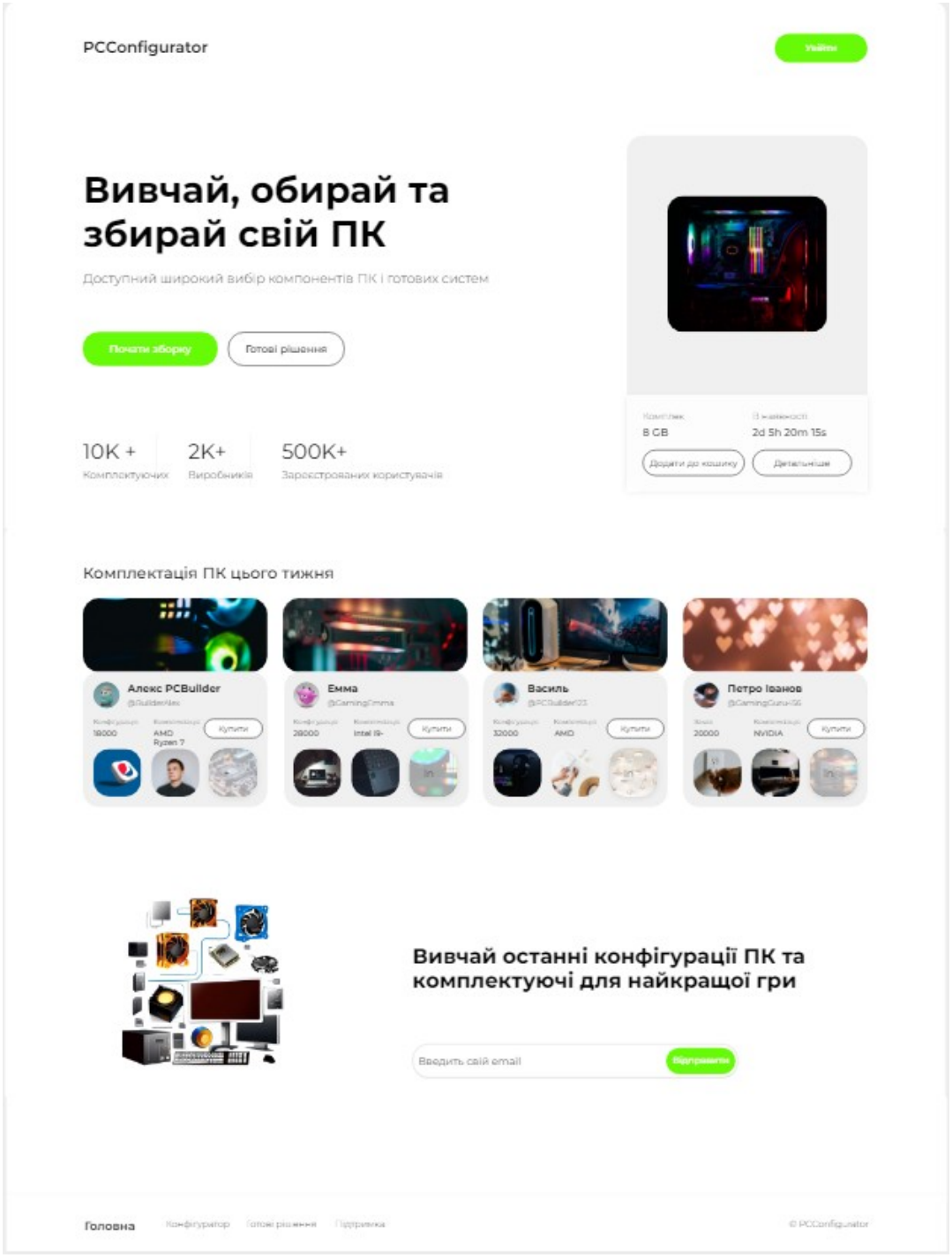


Рисунок 4.1 – Головна сторінка веб-системи «Конфігуратор ПК»

Після входу в систему як Користувач можна перейти до свого особового кабінету (рис.4.2). У ньому відображуються конфігурації, що вже збиралися користувачем. Їх можна продивитися та замовити. Також є можливість додати нову конфігурацію, натиснувши кнопку «Додати нову».

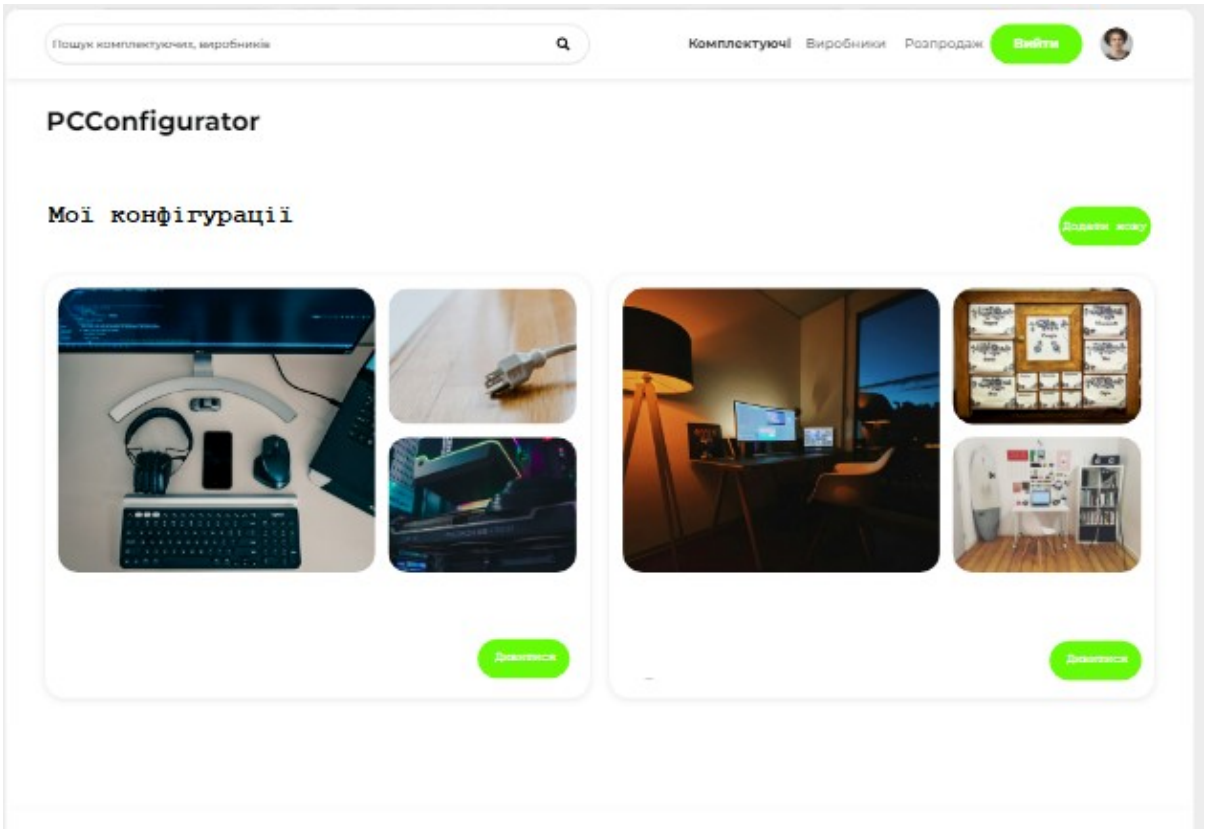


Рисунок 4.2 – Сторінка Особовий кабінет користувача веб-системи «Конфігуратор ПК»

Щоб почати використовувати конфігуратор ПК, натискаємо цю кнопку та переходимо на сторінку самого конфігуратор (рис.4.3).

Зверху є меню, що дозволяє послідовно обрати всі необхідні комплектуючі ПК. У всіх комплектуючих є можливість виставити фільтр по різним критеріям в залежності від типу. Кнопка «Додати» додає обрані комплектуючі до конфігурації користувача. Щоб перевірити сумісність компонентів є розділ «Сумісність», де система автоматично перевіряю чи підходять обрані компоненти.

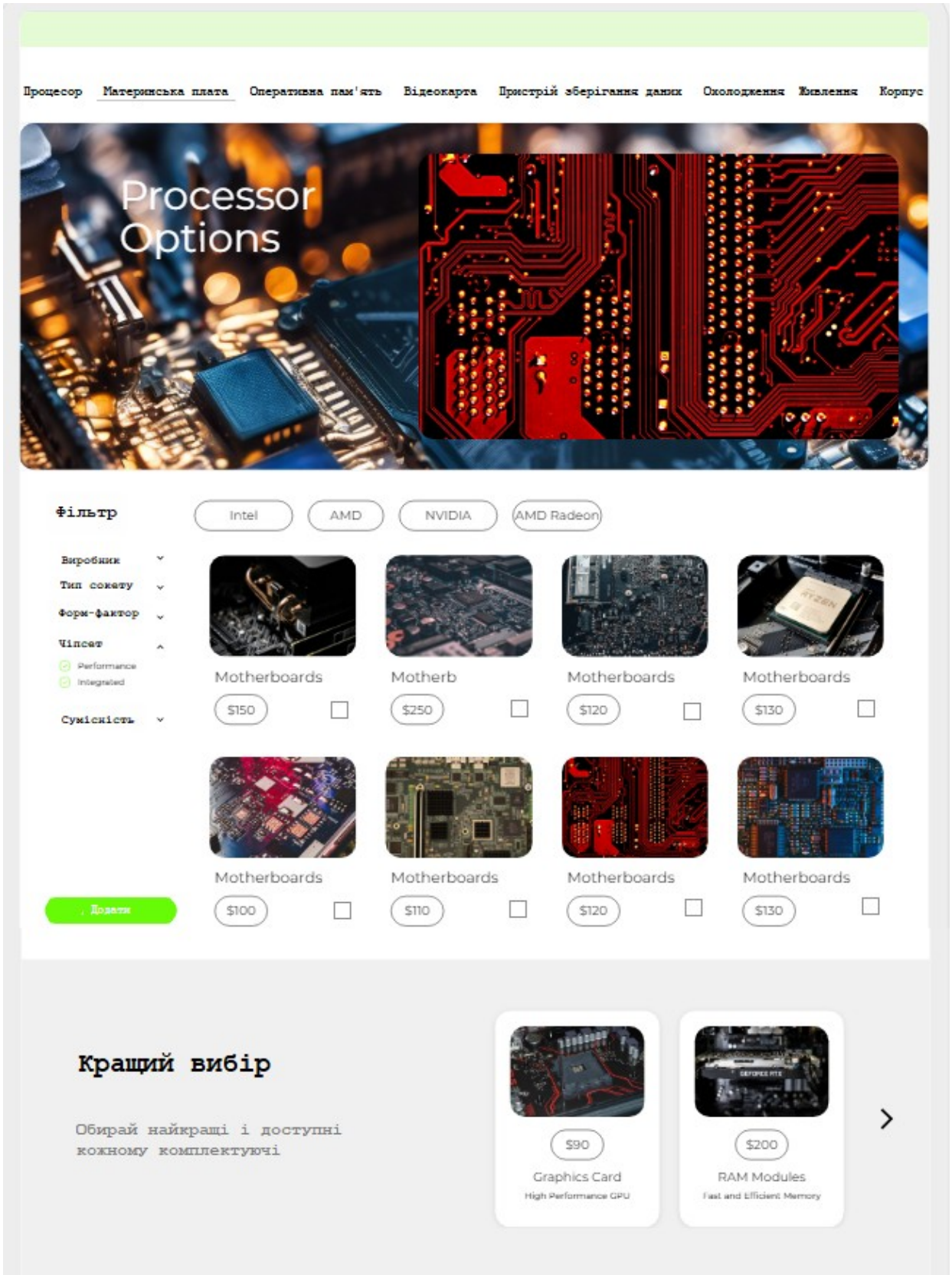


Рисунок 4.3 – Сторінка роботи конфігуратор системи

При виборі пункту меню «Готові рішення» відображується готові ПК, що є в наявності. Кожне з них можна додати до кошика та оплатити покупку (рис.4.4).

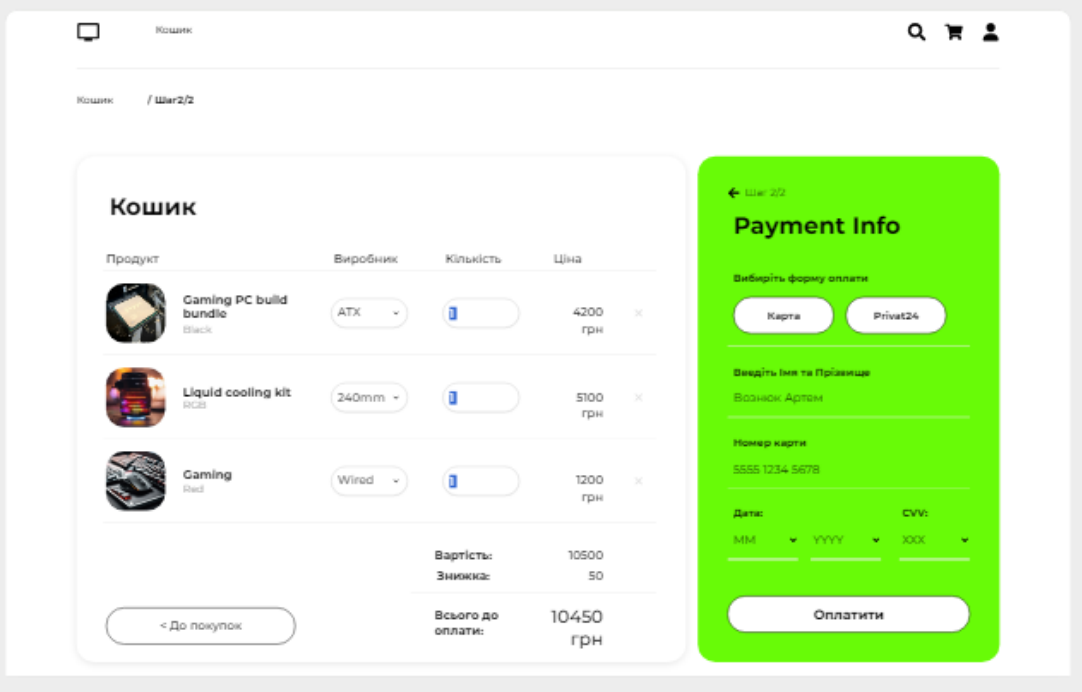


Рисунок 4.4 – Замовлення та оплата товару

Якщо увійти в систему з роллю Адміністратор, то будуть можливості обробляти замовлення та редагувати дані.

Обробка замовлень представлена на рис.4.5.

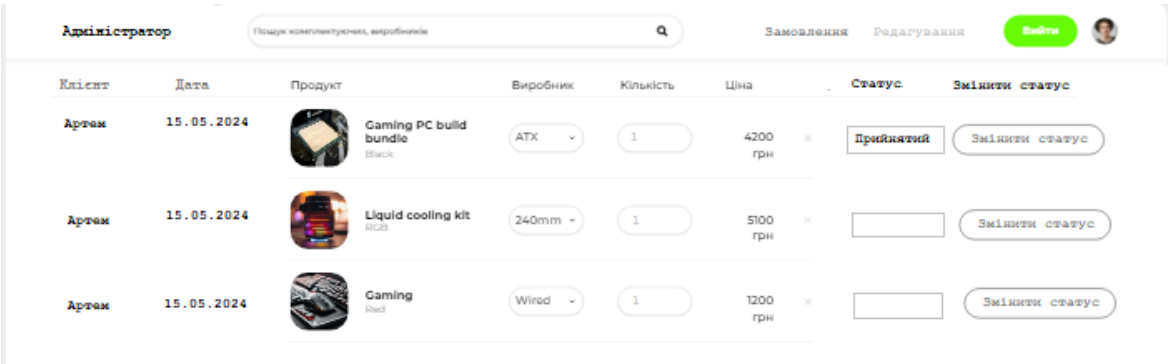


Рисунок 4.5 – Обробка замовлень Адміністратором

Редагування комплектуючих, тобто додавання, видалення, зміна інформації представлено на рис.4.6. При натисканні кнопки «Редагувати» з’являється можливість вносити зміни, тобто змінити опис та ціну, після чого натиснути кнопку «Оновити».

Також на цій сторінці можна вибрати готові збірки, що вже розміщенні в системі та також скоректувати, оновити та видалити інформацію про них із системи. За допомогою відповідних кнопок всю інформацію можна зберегати та оновлювати.

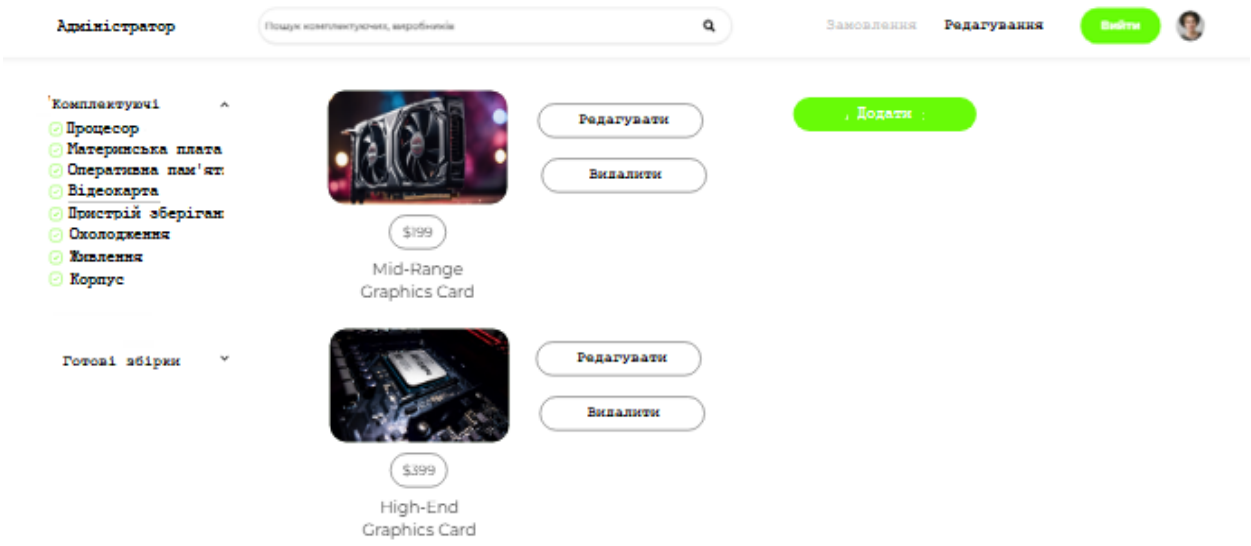


Рисунок 4.6 – Редагування комплектуючих Адміністратором

Таким чином, в результаті роботи була створена веб-система з можливістю вибору конфігуратора ПК, оформлення покупки та функціями адміністрування системи.

# 5 ТЕСТУВАННЯ ПРОГРАМИ

Тестування – це процес оцінки програмного забезпечення або системи для виявлення дефектів та забезпечення того, що вона відповідає заданим вимогам і працює належним чином. Тестування включає виконання ПЗ з метою виявлення помилок, перевірки функціональності, продуктивності, безпеки та інших аспектів якості.

Тестування проводять задля:

– виявлення дефектів: виявлення помилок, багів або дефектів у програмі, щоб вони могли бути виправлені до релізу;

– перевірка відповідності вимогам: забезпечення того, що програмне забезпечення відповідає вимогам, визначеним у специфікаціях та технічних завданнях;

– підвищення якості ПЗ: покращення загальної якості програмного забезпечення, забезпечуючи його надійність, стабільність та відповідність вимогам користувача;

– підтвердження функціональності: переконання, що всі функції програмного забезпечення працюють як очікується, і що жодні частини системи не впливають одна на одну негативно;

– оцінка продуктивності: вимірювання продуктивності ПЗ у реальних умовах, включаючи швидкість обробки, здатність витримувати навантаження та ефективність роботи;

– перевірка безпеки: забезпечення того, що ПЗ захищене від загроз, таких як несанкціонований доступ, зловмисне програмне забезпечення та інші вразливості;

– забезпечення зручності використання: оцінка зручності інтерфейсу користувача, щоб гарантувати, що кінцеві користувачі можуть легко і ефективно використовувати програму.

Тестування є критично важливим етапом у розробці програмного забезпечення, що дозволяє забезпечити високу якість кінцевого продукту, задоволеність користувачів та мінімізувати ризики, пов'язані з помилками та дефектами в програмі.

Складемо тест-кейси для перевірки варіантів використання.

Тест-кейс 1: Створення нової збірки

Опис тестового прикладу: Перевірка можливості створення нової збірки.

Етапи тестування:

Адміністратор переходить до розділу «Готові збірки ПК».

Адміністратор вибирає опцію «Створити нову збірку».

Адміністратор заповнює форму нової збірки.

Адміністратор натискає кнопку «Зберегти».

Тестові дані: Назва збірки, компоненти, опис.

Очікуваний результат: Нова збірка збережена успішно, відображається повідомлення про успішне створення.

Фактичний результат: (вписується після тестування).

Тест-кейс 2: Редагування існуючої збірки

Опис тестового прикладу: Перевірка можливості редагування існуючої збірки.

Етапи тестування:

Адміністратор вибирає збірку зі списку для редагування.

Адміністратор вносить зміни до деталей збірки.

Адміністратор натискає кнопку «Оновити».

Тестові дані: Змінені компоненти, опис.

Очікуваний результат: Збірка оновлена успішно, відображається повідомлення про успішне оновлення.

Фактичний результат: (вписується після тестування).

Тест-кейс 3: Видалення збірки

Опис тестового прикладу: Перевірка можливості видалення збірки.

Етапи тестування:

Адміністратор вибирає збірку зі списку для видалення.

Адміністратор підтверджує видалення збірки.

Тестові дані: Ідентифікатор збірки.

Очікуваний результат: Збірка видалена успішно, відображається повідомлення про успішне видалення.

Фактичний результат: (вписується після тестування).

Ці тест-кейси забезпечать перевірку основних функцій, пов'язаних з керуванням готовими збірками ПК, що допоможе виявити та виправити можливі проблеми в системі.

Тест-кейси:

Тест-кейс 1: Створення нового комплектуючого

Опис тестового прикладу: Перевірка можливості створення нового комплектуючого.

Етапи тестування:

Адміністратор переходить до розділу «Комплектуючі».

Адміністратор вибирає опцію «Створити нове комплектуюче».

Адміністратор заповнює форму нового комплектуючого.

Адміністратор натискає кнопку «Зберегти».

Тестові дані: Назва комплектуючого, тип, характеристики, опис.

Очікуваний результат: Нове комплектуюче збережене успішно, відображається повідомлення про успішне створення.

Фактичний результат: (вписується після тестування).

Тест-кейс 2: Редагування існуючого комплектуючого

Опис тестового прикладу: Перевірка можливості редагування існуючого комплектуючого.

Етапи тестування:

Адміністратор вибирає комплектуюче зі списку для редагування.

Адміністратор вносить зміни до деталей комплектуючого.

Адміністратор натискає кнопку «Оновити».

Тестові дані: Змінені характеристики, опис.

Очікуваний результат: Комплектуюче оновлене успішно, відображається повідомлення про успішне оновлення.

Фактичний результат: (вписується після тестування).

Тест-кейс 3: Видалення комплектуючого

Опис тестового прикладу: Перевірка можливості видалення комплектуючого.

Етапи тестування:

Адміністратор вибирає комплектуюче зі списку для видалення.

Адміністратор підтверджує видалення комплектуючого.

Тестові дані: Ідентифікатор комплектуючого.

Очікуваний результат: Комплектуюче видалене успішно, відображається повідомлення про успішне видалення.

Фактичний результат: (вписується після тестування).

Ці тест-кейси забезпечать перевірку основних функцій, пов'язаних з керуванням комплектуючими в конфігураторі ПК, що допоможе виявити та виправити можливі проблеми в системі.

варіант використання: відповідь на заявку користувача та видалення заявки – можливість для адміністратора дати відповідь користувачу на його заявку додати якесь комплектуюче, а також видалити заявку

Тест-кейси:

Тест-кейс 1: Відповідь на заявку користувача

Опис тестового прикладу: Перевірка можливості відповісти на заявку користувача.

Етапи тестування:

Адміністратор переходить до розділу «Заявки користувачів».

Адміністратор вибирає заявку для перегляду.

Адміністратор вводить відповідь на заявку.

Адміністратор натискає кнопку «Надіслати відповідь».

Тестові дані: Текст відповіді на заявку.

Очікуваний результат: Користувач отримує відповідь на заявку, відображається повідомлення про успішне надсилання.

Фактичний результат: (вписується після тестування).

Тест-кейс 2: Видалення обробленої заявки

Опис тестового прикладу: Перевірка можливості видалення заявки після обробки.

Етапи тестування:

Адміністратор переходить до розділу «Заявки користувачів».

Адміністратор вибирає оброблену заявку для видалення.

Адміністратор підтверджує видалення заявки.

Тестові дані: Ідентифікатор заявки.

Очікуваний результат: Заявка видалена успішно, відображається повідомлення про успішне видалення.

Фактичний результат: (вписується після тестування).

Тест-кейс 3: Неправильна обробка заявки

Опис тестового прикладу: Перевірка поведінки системи при введенні некоректних даних або відсутності відповіді.

Етапи тестування:

Адміністратор переходить до розділу «Заявки користувачів».

Адміністратор вибирає заявку для перегляду.

Адміністратор залишає форму відповіді порожньою.

Адміністратор натискає кнопку «Надіслати відповідь».

Тестові дані: Порожня відповідь.

Очікуваний результат: Система виводить повідомлення про необхідність заповнення відповіді.

Фактичний результат: (вписується після тестування).

Ці тест-кейси дозволять перевірити основні функції, пов'язані з обробкою заявок користувачів, що допоможе виявити та виправити можливі проблеми в системі.

# ВИСНОВКИ

Необхідність веб-застосунків для кінцевого складання ПК полягає в тому, щоб спростити вибір споживача, виключивши компоненти, які змушують сумніватися в їх якості. І, у свою чергу, дати можливість для великого вибору у підході до збирання комп'ютера. Так, основною метою роботи стала реалізація конфігуратора для збирання ПК.

В рамках цієї роботи було реалізовано веб-систему магазину з продажу ПК з функцією «Конфігуратор для збирання ПК» за допомогою мови PHP та бази даних MySQL. При цьому було вирішені наступні завдання:

1) здійснено аналіз предметної області;

2) проведено огляд існуючих аналогів для веб-системи, що розробляється;

3) визначено функціональні та нефункціональні вимоги до системи;

4) виконане проєктування веб-системи;

5) реалізовано веб-систему та базу даних;

6) проведено тестування веб-системи.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Telemart.ua – URL: https://telemart.ua/ua/ assembly.html (дата звернення: 25.04.2024).

2. Конфігуратор ПК від CAN.ua – URL: https://can.ua/configurator/ html (дата звернення: 25.04.2024).

3. Архітектура вебзастосунків. URL: https://habr.com/ ru/articles/493430/. (дата звернення: 30.04.2024).

4. Е. Бенкен. PHP, MySQL, XML. Программирование для Интернета. – К.: Диалектика, 2007. – 336 с.

5. JavaScript – URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript (дата звернення: 10.05.2024).

6. А. Петюшкин. HTML в Web-дизайне. – К.: Диалектика, 2004.– 400 с.

7. Самоучитель CSS – URL: http://htmlbook.ua/samcss (дата звернення: 12.05.2024).

8. Bootstrap Docs – URL: https://getbootstrap.com/docs/5.0/getting-started/introduction/ (дата звернення: 15.05.2024).

9. Мулеса О.Ю. Основи мови запитів SQL. – Ужгород, 2015. – 48 с.

10. Visual StudioCode – URL: https://code.visualstudio.com/docs (дата звернення: 19.05.2024).