МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Центр заочної, дистанційної та вечірньої форм навчання

Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
Оксана ШОВКОПЛЯС

(підпис)

04 грудня 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня магістр

зі спеціальності 122 «Комп'ютерні науки»,
освітньо-професійної програми «Інформатика»
на тему: «Інформаційна технологія інтеграції системи керування клієнтською взаємодією з платформою електронної комерції та соціальними мережами»
здобувачки групи ІН.мз-31с Гаркавенко Аліни Євгеніївни

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Аліна ГАРКАВЕНКО

(підпис)

Керівник,
асистент кафедри комп'ютерних наук
кандидат фізико-математичних наук Ольга ШУТИЛЄВА

(підпис)

Суми – 2024

Сумський державний університет
Центр заочної, дистанційної та вечірньої форм навчання
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Оксана ШОВКОПЛЯС
(підпис)

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня магістра
зі спеціальності 122 «Комп'ютерні науки», освітньо-професійної програми «Інформатика»
здобувачки групи ІН.мз-31с Гаркавенко Аліни Євгеніївни

1. Тема роботи: «Інформаційна технологія інтеграції системи керування клієнтською взаємодією з платформою електронної комерції та соціальними мережами» затверджена наказом по СумДУ від *«05» грудня 2024 р. № 1267-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 06 грудня 2024 року*
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Аналіз проблеми предметної області, постановка й формування завдань дослідження. 2) Огляд технологій, що використовуються для інтеграції системи керування клієнтською взаємодією з платформою електронної комерції та соціальними мережами. 3) Розробка інформаційної технології інтеграції системи керування клієнтською взаємодією з платформою електронної комерції та соціальними мережами. 4) Аналіз результатів.*
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «18» серпня 2024 р.

Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми предметної області, постановка й формування завдань дослідження*		
2	*Огляд та аналіз систем-аналогів*		
3	*Розробка інформаційної технології інтеграції системи керування клієнтською взаємодією з платформою електронної комерції та соціальними мережами*		
4	*Написання супровідної документації*		
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*		

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 78 стор., 41 рис., 4 додатки, 21 використаних джерел.

Обґрунтування актуальності теми роботи – тема кваліфікаційної роботи є актуальною, оскільки присвячена розв'язанню важливої практичної задачі інтеграції системи керування клієнтською взаємодією (CRM) з платформою електронної комерції та соціальними мережами. Це дозволить покращити управління клієнтськими даними, підвищити ефективність маркетингових кампаній та оптимізувати процеси обслуговування клієнтів, що є важливими завданнями у сучасному бізнес-середовищі.

Об'єкт дослідження – процес інтеграції CRM-системи з платформою електронної комерції та соціальними мережами.

Мета роботи – розробка інформаційної технології для інтеграції системи керування клієнтською взаємодією з платформою електронної комерції та соціальними мережами для покращення клієнтської підтримки та автоматизації маркетингових процесів.

Методи дослідження – алгоритми обробки даних, методи інтеграції та синхронізації даних між платформами, технології аналітики та персоналізації клієнтських даних.

Результати – розроблено інформаційну технологію, яка забезпечує інтеграцію CRM з електронною комерцією та соціальними мережами, зчитує дані клієнтів, обробляє їх та дозволяє користувачу керувати клієнтськими взаємодіями, генерувати дані про продажі та аналізувати дані про поведінку клієнтів.

CRM, ЕЛЕКТРОННА КОМЕРЦІЯ, СОЦІАЛЬНІ МЕРЕЖІ, API, АВТОМАТИЗАЦІЯ МАРКЕТИНГУ, АНАЛІТИКА КЛІЄНТІВ

ЗМІСТ

ВСТУП

Важливість інформаційних систем у бізнесі не можна недооцінювати. Сьогодні неможливо уявити ведення бізнесу без Інтернету, відеоконференцій, додатків для управління проєктами тощо. Цей факт зумовлює необхідність впровадження технологій у бізнес-процеси, якщо цього не було зроблено.

Сучасні підприємства значною мірою покладаються на інформаційні системи. Вони потребують IT-послуг, щоб підтримувати безперервність та безперебійність бізнес-операцій. IT-команди керують та захищають важливі дані, що сприяють інноваціям та ефективності. З огляду на це, наявність IT-фахівців може відігравати невід'ємну роль в успіху бізнесу.

Системи забезпечують швидші, ширші та ефективніші засоби комунікації. Це стосується взаємодії всередині команди або з клієнтами, потенційними клієнтами, інвесторами чи широкою громадськістю. Інформаційні системи для управління запасів магазину одягу можна використовувати для відстеження продажів, деталей замовлень тощо.

Інформаційні системи пов'язані з розробкою, підтримкою та застосуванням комп'ютерних систем, програмного забезпечення та мереж для обробки та розповсюдження даних. Це стосується аспектів, пов'язаних з обчислювальною технікою, включаючи апаратне забезпечення, додатки, мережі, Інтернет та людей, які з ними працюють.

Щоб зрозуміти важливість інформаційних систем у бізнесі потрібно знати що дані системи виконують наступні функції:

- Сприяють зростанню доходів завдяки значному покращенню бізнес-процесів та продуктів.

- Допомога у створенні нового бізнесу за короткий час завдяки наявності точної інформації.

- Забезпечення своєчасної та ефективної комунікації з клієнтами для задоволення їхніх потреб.

Важливість інформаційних систем у бізнесі полягає у сприянні інноваціям. Інновації пропонують розумніші додатки, покращене зберігання даних, швидшу обробку. Більше того, інновації роблять бізнес більш ефективним. Також інновації збільшують цінність, покращують якість і підвищують продуктивність.

Актуальність. Сучасні системи управління взаємовідносинами з клієнтами (CRM) для магазинів одягу здатні полегшити комунікацію з клієнтами, одночасно оптимізуючи управління запасами. Застосування аналітики даних для управління запасами дозволяє скоротити витрати, пов'язані з обслуговуванням товарів, мінімізувати дефіцит або надлишки, а також прогнозувати попит на різні категорії одягу. Постійний розвиток технологій та зміни у споживчих вподобаннях роблять такі рішення незамінними для збереження конкурентоспроможності магазинів одягу.

Об’єкт дослідження. Процеси управління запасами в контексті роздрібних магазинів одягу охоплюють цілий ряд видів діяльності, включаючи аналіз даних про продажі, прогнозування попиту та планування поставок клієнтам.

Предмет дослідження. Розвиток інформаційних систем та інтеграції CRM-систем, для оптимізації управління запасами. Ця інтеграція призвела до забезпечення точного аналізу попиту та ефективної координації поставок.

Гіпотеза. Впровадження інформаційних систем управління запасами на основі аналізу даних дозволить підвищити ефективність управління запасами, зменшити втрати через перевантаження або дефіцит товарів і оптимізувати загальну ефективність роботи магазинів одягу.

Метою даного дослідження є розробка інформаційної технології інтеграції системи керування клієнтською взаємодією з платформою електронної комерції та соціальними мережами. Для досягнення цієї мети необхідно вирішити наступні завдання:

- Визначити основні функціональні та нефункціональні вимоги до системи.

- Проаналізувати існуючі CRM-рішення для управління запасами.
- Вибрати відповідні технології для розробки інформаційної системи.
- Створити архітектуру інформаційної системи.
- Розробити структуру та наповнити базу даних.
- Розробити технологію управління запасами.
- Впровадити технологію та інтегрувати її з CRM.

Запропонована система являє собою новий підхід до управління запасами в контексті магазинів одягу. Розробка інформаційних систем для управління запасами в магазинах одягу є актуальним напрямком досліджень у зв'язку зі зростаючою необхідністю оптимізації процесів постачання та підтримки оптимального асортименту на складі. Запропонована система полегшить прогнозування попиту, зменшить надлишкові запаси та мінімізує витрати на зберігання. У світлі постійно мінливого ринку одягу та еволюції споживчих уподобань інтеграція аналітики даних у процес управління запасами стала важливою стратегією для підвищення ефективності роботи магазину та конкурентних переваг.

Структура роботи складається з аналізу предметної області, етапу структурно-функціонального моделювання та програмної реалізації.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень та публікацій

Впровадження інформаційної системи для управління взаємовідносинами з клієнтами (CRM) є значною інвестицією часу та зусиль. Нижче наведено лише деякі з переваг, які може отримати бізнес після впровадження системи управління взаємовідносинами з клієнтами.

Однією з головних переваг системи управління взаємовідносинами з клієнтами є покращення взаємовідносин з клієнтами. Система управління взаємовідносинами з клієнтами – це програмне забезпечення, призначене для оптимізації управління діловими контактами та пов'язаною з ними інформацією. Вона полегшує зберігання відповідних даних по різних каналах зв'язку, включаючи демографічну інформацію, історію покупок і попередніх взаємодій [1].

У дослідженні [2] розглядається впровадження CRM-систем, інтегрованих зі штучним інтелектом, з особливим акцентом на те, як штучний інтелект покращує управління клієнтами та сталий розвиток у бізнес-контексті. У роботі розглядається вплив штучного інтелекту на CRM, обговорюється, як він сприяє ефективнішому прийняттю рішень, покращує взаємодію з клієнтами та підтримує стратегії сталого розвитку підприємств. Це дослідження особливо актуальне для компаній, які прагнуть розширити можливості CRM завдяки застосуванню аналітики та автоматизації на основі штучного інтелекту [2].

Крім того, CRM-система забезпечує зручний доступ до цієї інформації для будь-якого члена організації. Кожна взаємодія між співробітником вашої компанії та клієнтом – це можливість підвищити рівень його задоволеності. CRM-система є оптимальним рішенням для досягнення цих цілей.

Надання відмінного обслуговування клієнтів є ефективним засобом підвищення їх лояльності. Висока плинність клієнтів є несприятливим явищем

для будь-якого бізнесу, оскільки може негативно вплинути на низку ключових показників ефективності, включаючи продажі та репутацію бренду. Система управління взаємовідносинами з клієнтами може підвищити якість обслуговування клієнтів, що, в свою чергу, може сприяти підвищенню їхньої лояльності. Така система може автоматизувати підтримку клієнтів, відстежувати їхню поведінку і навіть забезпечувати аналіз настроїв. Такі функції сприяють виявленню та вирішенню проблем ще до того, як вони загострятьcя. Впровадження системи управління взаємовідносинами з клієнтами може сприяти покращенню загального рівня обслуговування клієнтів, тим самим підвищуючи їхню лояльність.

Стаття [3] містить вичерпний огляд історичної еволюції CRM-систем, їхніх основних компонентів та проблем, з якими стикаються компанії при їхньому впровадженні. Крім того, в ньому викладено стратегічні рамки для майбутнього розвитку CRM, підкреслено важливість управління даними про клієнтів та інтеграції технологічних інструментів для покращення взаємовідносин з клієнтами [3].

Підвищення ефективності та точності управління ресурсами та фінансами в процесі будівництва є ключовим. Вебсистеми надають інтегровані інструменти для розрахунку витрат, планування робіт і взаємодії з підрядниками, а також допомагають оптимізувати бюджети і терміни реалізації проєктів.

Існує позитивна кореляція між лояльністю клієнтів і збільшенням продажів. Програмне забезпечення для управління взаємовідносинами з клієнтами (CRM) допомагає розвивати конвеєр продажів, сприяючи оптимізації процесу продажу та автоматизації основних завдань. Воно дозволяє аналізувати комплексні дані про продажі та зберігати цю інформацію в централізованому сховищі, доступному для уповноваженого персоналу. Ця функціональність дозволяє компаніям налагодити систематичний процес продажів, який можна налаштувати відповідно до конкретних вимог [4].

У статті [5] розглядається взаємозв'язок між (CRM) та підприємницькими маркетинговими стратегіями. Воно підкреслює, як технологія CRM може допомогти стартапам, сприяючи більш персоналізованій взаємодії з клієнтами та вдосконалюючи маркетингові стратегії за допомогою інсайдів, заснованих на даних. Дослідження є важливим для розуміння ролі CRM у підвищенні гнучкості та інноваційності малого бізнесу [5].

Аналіз даних є фундаментальним аспектом розуміння поведінки клієнтів. Існує безліч даних про клієнтів, які необхідно зібрати, але вкрай важливо переконатися, що ви та ваші співробітники здатні точно інтерпретувати ці дані та ефективно їх використовувати. Дійсно, їх можна і потрібно використовувати для оптимізації бізнес-операцій. Системи управління взаємовідносинами з клієнтами оснащені вбудованими аналітичними можливостями, які контекстуалізують дані про клієнтів. Такі показники, як кількість кліків, відсоток відмов і демографічні дані, можуть продемонструвати ефективність кампанії та визначити шляхи для подальшої оптимізації [6].

Впроваджена система зможе надати необхідні інструменти для ефективного управління запасами в магазинах одягу завдяки застосуванню аналітики даних. Система полегшить управління запасами, моніторинг попиту на товари та автоматизацію процесів постачання, тим самим підвищивши загальну ефективність роботи. Завдяки інтуїтивно зрозумілому інтерфейсу та аналітичним звітам користувач зможе швидко отримувати необхідну інформацію та приймати обґрунтовані рішення. Загалом система надає наступні можливості:

- Наявність рольової системи з наданням прав доступу;
- CRUD операції з категоріями товарів;
- управління товарами (додавання, оновлення, видалення, редагування);
- додавання фото до товарів;

- управління продажами (перегляд, формування, редагування);
- формування звітів за вказаними критеріями;
- наявність таблиць та графічної інформації.

1.2 Аналіз програмних продуктів-аналогів

Використовуючи інформаційну систему для управління товарами представники індустрії моди та одягу зможуть збирати інформацію про продажі та з'ясовувати, які товари користуються популярністю.

Багато з доступних систем є багатофункціональними, що полегшує управління повсякденними завданнями шляхом оптимізації процесів, залишаючи користувачам можливість вільно будувати свої відносини з клієнтами та постачальниками. Хороша інформаційна система також дає змогу будувати маркетинг, продажі та графіки замовлень/затверджень, що базуються на знаннях. Маючи ці дані під рукою, дизайнери можуть створювати нові товари відповідно до останніх тенденцій, у стилях і кольорах, яким надають перевагу їхні покупці.

Для виявлення переваг та недоліків було обрано та проаналізовано наступні інформаційні системи – AIMS 360, ApparelMagic, ShowroomHQ.

AIMS 360, розроблена для швейної промисловості, може покращити спосіб планування та відстеження основної діяльності всього бізнесу. Ця система є інтегрованою системою обробки замовлень і виробництва для виробників одягу, оптовиків, імпортерів і дистриб'юторів усіх розмірів. AIMS 360 – це багатофункціональна система, що надає набір інструментів управління, які оживляють, налаштовують і покращують критичні операції. Система AIMS працює через хмару як хостингове рішення або може бути придбана для розгортання на місці. AIMS 360 побудована на новітній платформі .NET і системі баз даних SQL [7].

На рисунку 1.1 представлено вигляд інтерфейсу AIMS 360.

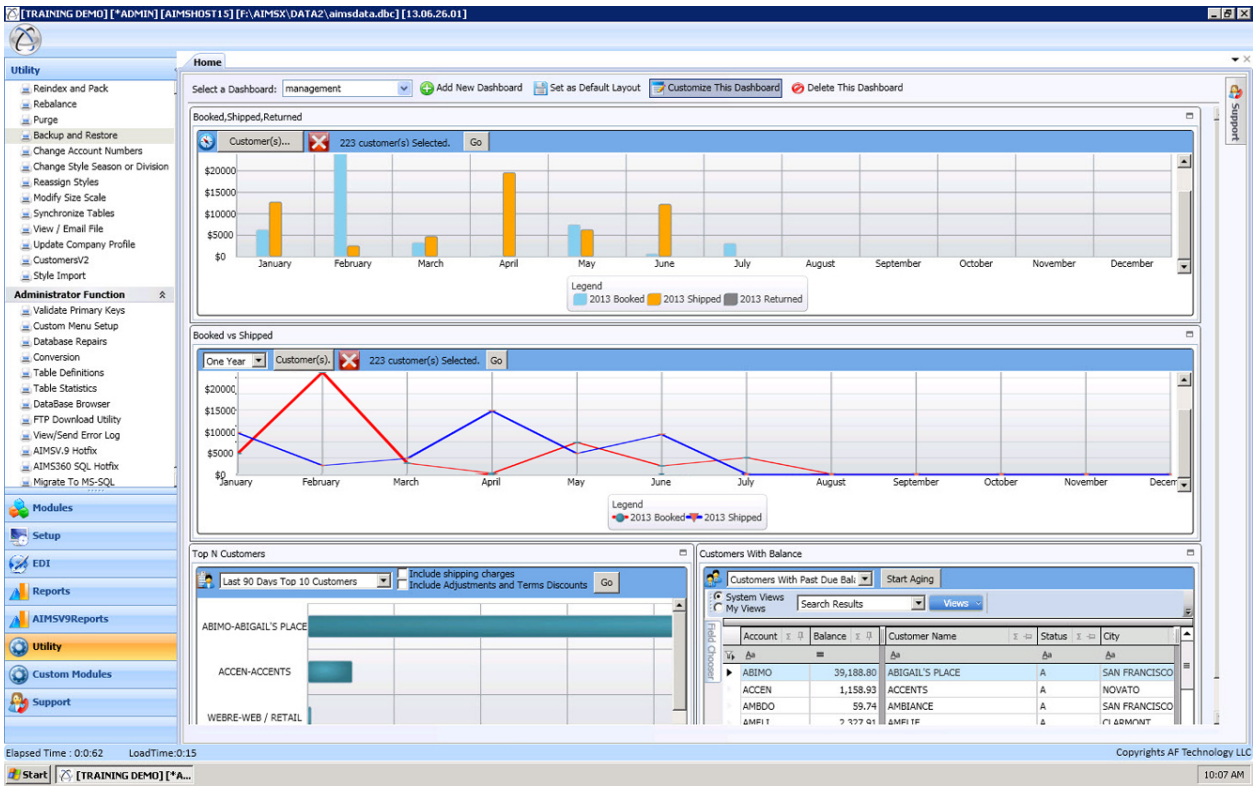


Рисунок 1.1 – Вигляд інтерфейсу AIMS 360

ApparelMagic пропонує ERP, PLM, CRM, PDM, віддалене введення замовлень, повністю інтегровану бухгалтерію та розширені аналітичні звіти. Інтеграційні рішення ApparelMagic включають в себе обробку замовлень на дропшиппінг для роздрібних торговців. Системи також полегшують закупівлю та поповнення запасів.

Контроль запасів у режимі реального часу інтегрується з UPS, FedEx та іншими складськими системами. ApparelMagic, потужний інструмент інтеграції, також пропонує безкоштовне навчання, технічну підтримку в режимі реального часу, оновлення продуктів і потужні інструменти онлайн-довідки, що дають користувачам повний контроль. Ціна підписки починається від 80 доларів на місяць [8].

На рисунку 1.2 представлено вигляд інформаційної системи ApparelMagic.

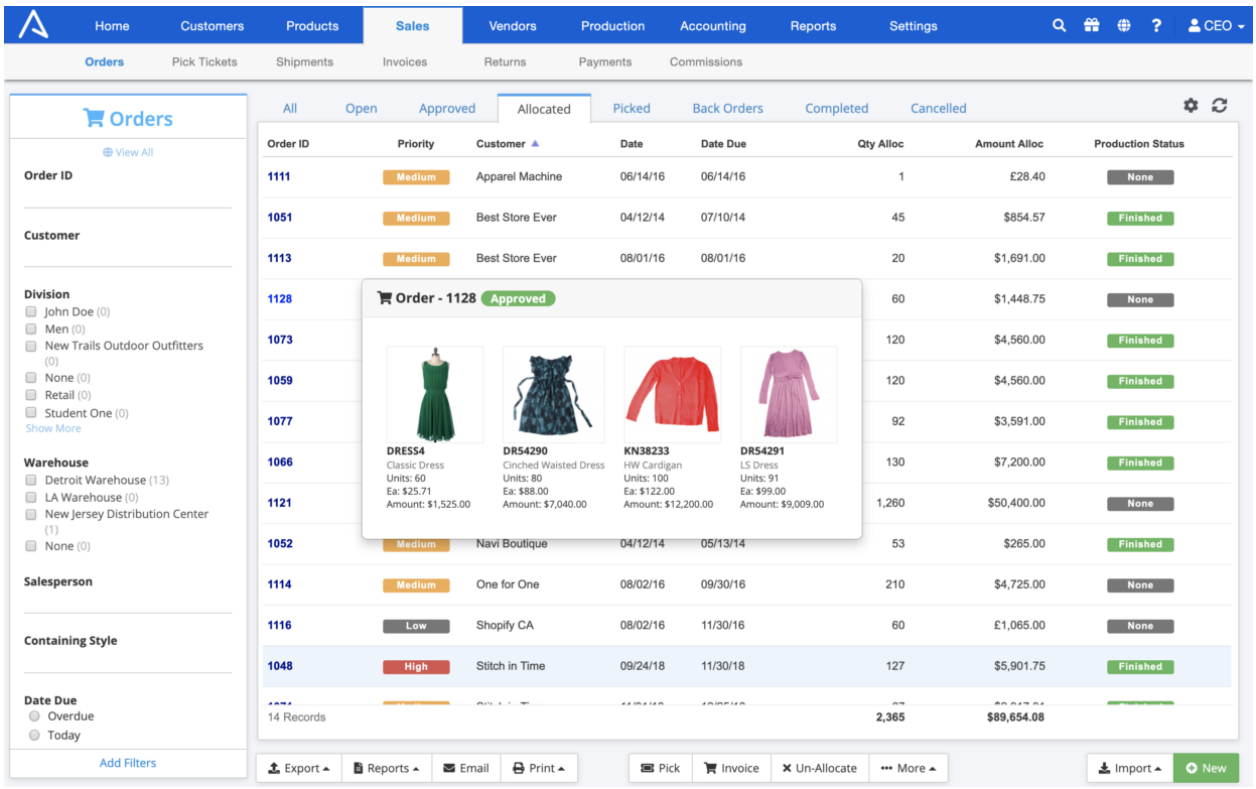


Рисунок 1.2 – Вигляд інформаційної системи ApparelMagic

ShowroomHQ був розроблений як хмарне програмне забезпечення для обслуговування торгових представників і агентств в індустрії одягу та товарів для дому. Це простий і зручний у використанні інструмент, який полегшує управління шоурумом.

За допомогою ShowroomHQ користувач може відстежувати замовлення, керувати клієнтами та зустрічами, а також продавати свої колекції за допомогою масових розсилок за певними критеріями. Все це можна робити з одного зручного місця. ShowroomHQ не встановлюється. Оскільки це хмарний сервіс, то можна керувати шоурумом з будь-якого комп'ютера чи мобільного пристрою [9].

На рисунку 1.3 представлено вигляд інтерфейсу інформаційної системи ShowroomHQ.

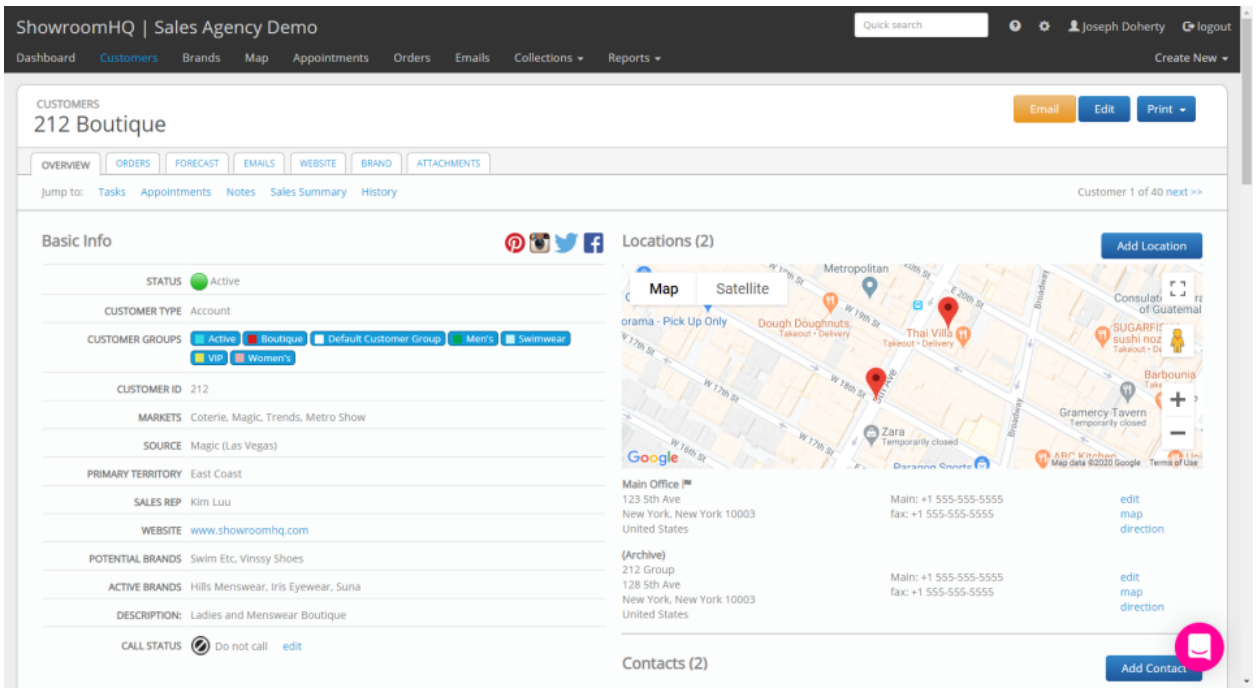


Рисунок 1.3 – Вигляд. Інтерфейсу інформаційної системи ShowroomHQ

Для більш детального аналізу аналогів було створено таблицю 1.1 для порівняння вебсервісів.

Таблиця 1.1 – Порівняння інформаційних систем для керуванням продажу товарів

Система	AIMS 360	ApparelMagic	ShowroomHQ
Основна функція	Управління процесами виробництва та продажу в швейній сфері	CRM для модних брендів, управління замовленнями та аналітикою	Система для керування операціями продажу та постачання товарів
Тип платформи	Хмарне або локальне рішення, побудоване на платформі .NET	Вебсумісний, працює на PC, Mac, iPad	Хмарна платформа, доступ з будь-якого пристрою
Інтеграція	Інтеграція з обробкою замовлень і виробничими процесами	Інтеграція з UPS, FedEx, системами дропшиппінгу, повна бухгалтерія	Відстеження замовлень, масові розсилки
Ціна	Не вказана	Від 80 доларів на місяць	Не вказана

Продовження таблиці 1.1 - Порівняння інформаційних систем для керуванням продажу товарів

Основні можливості	Обробка замовлень, планування, відстеження діяльності	Управління запасами, облік, аналітичні звіти, контроль у реальному часі	Управління клієнтами, замовленнями, зустрічами, масові розсилки
Підтримка	Технічна підтримка при хостинговому рішенні	Безкоштовне навчання, технічна підтримка в реальному часі, оновлення продукту	Підтримка для віддаленої роботи з клієнтами
Цільова аудиторія	Виробники одягу, дистриб'ютори	Модні бренди, роздрібні продавці	Торгові представники та агентства в індустрії одягу та товарів для дому

Враховуючи зазначені переваги, розроблена інформаційна система управління запасами для магазинів одягу повинна сприяти швидкому отриманню аналітичних даних щодо стану запасів та продажів. Інтуїтивно зрозумілий інтерфейс допоможе користувачам в ефективному управлінні асортиментом, прогнозуванні попиту та оптимізації процесів постачання.

Крім того, надзвичайно важливо гарантувати захист персональних даних користувачів, включаючи інформацію про постачальників і транзакції, що є ключовим аспектом безпеки бізнесу.

1.3 Постановка задачі

Аналіз аналогічних систем дозволив виявити основні переваги, які можуть бути використані для розробки інформаційної системи управління запасами в магазинах одягу на основі аналітики даних. Необхідно реалізувати систему у вигляді вебдодатку, доступного з різних пристроїв через мережу Інтернет, що забезпечить гнучкість і простоту використання.

Інформаційна система повинна бути оснащена наступними функціями:

- Доступ до системи буде надаватися користувачам відповідно до їхніх персональних даних, а права доступу визначатимуться системою на основі ролей (адміністратор, менеджер, користувач).
- Управління товарними категоріями має бути забезпечено шляхом реалізації комплексного циклу CRUD (створення, читання, оновлення, видалення), що сприятиме ефективному управлінню товарним асортиментом.
- Система також повинна полегшити управління товарами, включаючи можливість додавання, оновлення, видалення та редагування товарів, а також додавання фотографій товарів для більш зручного візуального представлення.
- Модуль управління продажами полегшить перегляд історії продажів, створення та редагування замовлень, а також можливість контролювати процеси продажів і отримувати відповідні аналітичні дані.
- Система повинна забезпечувати формування звітів за заданими критеріями, такими як періоди, категорії товарів та обсяги продажів. Цей функціонал дозволяє ефективно відстежувати динаміку запасів і продажів.
- Наявність таблиць і графіків, які візуально відображають стан запасів, обсяги продажів та інші важливі показники, полегшує аналіз і прийняття рішень.

Для того, щоб успішно впровадити інформаційну систему управління запасами для магазинів одягу, яка використовує аналітику даних, необхідно виконати ряд завдань.

- Важливо провести комплексну оцінку існуючих систем управління взаємовідносинами з клієнтами (CRM), які зараз використовуються для управління запасами. Ця оцінка має визначити ключові переваги кожної системи та врахувати їх при розробці нової системи.
- Необхідно визначити функціональність системи. Це має включати опис функцій системи, таких як управління товарами, категоріями та продажами, а також можливість додавання користувачів з різними рівнями доступу, створення та редагування звітів.

- Реалізація клієнт-серверної архітектури має важливе значення для забезпечення ефективної та надійної обробки даних.
- Наступним кроком є вибір відповідного технологічного стеку для реалізації системи. Це передбачає визначення необхідних мов програмування, фреймворків та бібліотек для роботи з даними та сервером.
- Реалізація логічної та фізичної структури системи разом з розробкою інтуїтивно зрозумілого інтерфейсу користувача забезпечить зручність роботи з системою.
- Необхідно розробити фізичну модель бази даних, що включає сутності (товари, категорії, продажі, користувачі) та зв'язки між ними. Обрана система управління базами даних (СКБД) повинна бути інтегрована в архітектуру.

1.4 Аналіз технологій та засобів реалізації інформаційної системи

Архітектура інформаційної системи – це високорівнева структура, яка визначає спосіб, у який продукт і бізнес працюватиме і масштабуватиметься. Процес аналізу архітектури інформаційних систем часто виявляється непростим завданням, оскільки безліч варіантів, доступних на ринку розробки програмного забезпечення, часто може бути приголомшливим [10].

Таким чином, виникає питання, який з доступних варіантів є оптимальним вибором з точки зору сучасної архітектури інформаційних систем, і які критерії слід використовувати в процесі оцінки. У таблиці 1.2 представлено порівняльну характеристику архітектур.

Після детального аналізу було обрано клієнт-серверну архітектуру яка включає в себе підвищену мережеву безпеку, більший контроль над мережевим трафіком, можливість бачити, що робить кожен комп'ютер, обмежувати певні дії та запобігати поширенню вірусів, а також обсяг пам'яті, доступний для кожного комп'ютера в мережі [11].

Як правило, в мережі клієнт-сервер сервер може надати значно більше місця для зберігання даних, ніж більшість зовнішніх накопичувачів, які можна підключити до мережі.

Таблиця 1.2 – Порівняння архітектур для розробки інформаційної системи

Архітектура	**Особливості**	**Переваги**	**Недоліки**	**Приклад використання**
Клієнт-сервер (Client-Server)	Класична модель з розподілом на клієнтську та серверну частини, де клієнт відправляє запити на сервер	Простота реалізації, розширюваність, ефективність використання серверних ресурсів	Складність масштабування, можливі проблеми з пропускною здатністю	Вебзастосунки для управління запасами, базовані на CRM системах
Мікросервіси (Microservices)	Розбиття системи на незалежні сервіси, кожен з яких відповідає за конкретну функціональність	Легке масштабування, модульність, простота підтримки окремих компонентів	Вища складність управління, необхідність оркестрації між сервісами	Системи з високим навантаженням на бази даних, де кожна функція виконується окремо (облік товарів, продажі, аналітика)
Трирівнева архітектура (3-Tier)	Поділ на три рівні: інтерфейс користувача, логіка додатку та база даних	Висока гнучкість та розширюваність, можливість використання різних технологій для кожного рівня	Можливі складнощі при інтеграції та налаштуванні зв'язків між рівнями	Великі системи, які потребують високого рівня абстракції між користувачами і базами даних
Serverless (Безсерверна)	Використання функцій як сервісу (FaaS), без постійного сервера, що зберігає дані	Оптимізація вартості, автоматичне масштабування, менші витрати на інфраструктуру	Високі затримки при запуску функцій, складність при тестуванні, обмеження за часом виконання функцій	Легкі вебдодатки або мікросервіси для обробки невеликих транзакцій (наприклад, формування звітів)

Для практичної реалізації серверної частини інформаційної системи було обрано наступні технології для аналізу:

- PHP;
- Node.js;

Node.js і PHP – це внутрішні вебтехнології, тобто їх основне призначення - виконувати однакові завдання, такі як отримання даних, доступ до бізнес-журналів, масштабування під робоче навантаження тощо. Обидві технології є крос-платформними і добре представлені у хмарних інфраструктурах, таких як AWS та Azure. PHP та Node.js мають великі екосистеми з активною підтримкою спільноти, що забезпечує часті зміни та швидке реагування на вразливості [12].

Хоча і PHP, і Node.js мають річні цикли випуску версій, кілька ключових відмінностей швидко стають очевидними при порівнянні цих двох технологій, особливо якщо порівнювати продуктивність PHP і Node.js по відношенню до циклів випуску версій.

Node.js використовує парну і непарну систему числення. Кожен реліз з непарним номером є більше публічною бета-версією для наступного релізу з парним номером, який зазвичай випускається приблизно в квітні. Тому ви ніколи не повинні запускати у виробництво з непарним номером релізу Node.js [13].

Якість архітектурного проєкту є важливим фактором успіху. Залучення компетентного архітектора з самого початку може сприяти успішному розгортанню будь-якої технології. На етапі зародження PHP спостерігалося значне поглинання технології завдяки її простоті та можливостям швидкого створення прототипів. Це призвело до значних проблем у спільноті, коли люди з обмеженим досвідом не могли належним чином захистити свої вебсайти, що негативно вплинуло на сприйняття PHP.

Аналогічно, PHP не є проблематичним за своєю суттю. Однак, оскільки в останніх версіях PHP було впроваджено низку захисних заходів, Node.js, схоже, стикається з проблемами в реалізації подібного підходу. У випадку з

Node.js дуже важливо залучити досвідченого архітектора на початкових етапах розробки. На противагу цьому, використання надійного PHP-фреймворку може виявитися вигідною інвестицією. [14]

Для більш детального аналізу технологій було створену порівняльну таблицю 1.3.

Таблиця 1.3 – Порівняльна таблиця технологій Node.js та PHP

Параметр	Node.js	PHP
Крос-платформність	Так (AWS, Azure)	Так (AWS, Azure)
Цикл випуску версій	Стабільні версії	Щорічна система оновлення
Екосистема	Велика підтримка спільноти	Широка спільнота користувачів
Масштабованість	Висока, завдяки асинхронній моделі	Менш ефективна для великих систем
Простота розробки	Складніша через асинхронність	Простий для створення прототипів
Безпека	Потребує досвідчених архітекторів	Сучасні фреймворки забезпечують захист

Для реалізації поставленої задачі було обрано технологію PHP, яка є популярним вибором для веброзробки завдяки великій кількості документації та простоті вивчення, що робить її доступною для розробників-початківців. Існування великого пулу талановитих професіоналів призводить до конкурентних витрат на найм.

Широка підтримка бібліотек, яку пропонує PHP, дозволяє скоротити час розробки завдяки використанню готових ресурсів. Крім того, вона сумісна з хмарними обчисленнями, що полегшує безпроблемне розгортання на таких платформах, як Amazon Web Services (AWS). Крім того, велика історія PHP гарантує його стабільність, а численні помилки з часом виправляються, що робить його надійним варіантом для розробки надійних додатків. [15].

2 МОДЕЛЮВАННЯ ВЕБЗАСТОСУНКУ

2.1 Структурно-функціональне моделювання

Діаграма IDEF0 – це метод моделювання функціональних процесів, що дозволяє візуалізувати ключові операції та потоки даних у системі. У контексті розробки інформаційної системи управління запасами для магазинів одягу IDEF0 пропонує засоби структурування взаємодії між категоріями товарів, управлінням продажами та аналітикою даних.

Це сприяє підвищенню ефективності завдяки забезпеченню прозорого візуального представлення всіх операційних процедур, що дозволяє більш ефективно розуміти і адмініструвати процеси управління запасами і продажами [16].

На рисунку 2.1 представлено контекстну діаграму процесу підтримки роботи інформаційної системи управління запасами одягу.

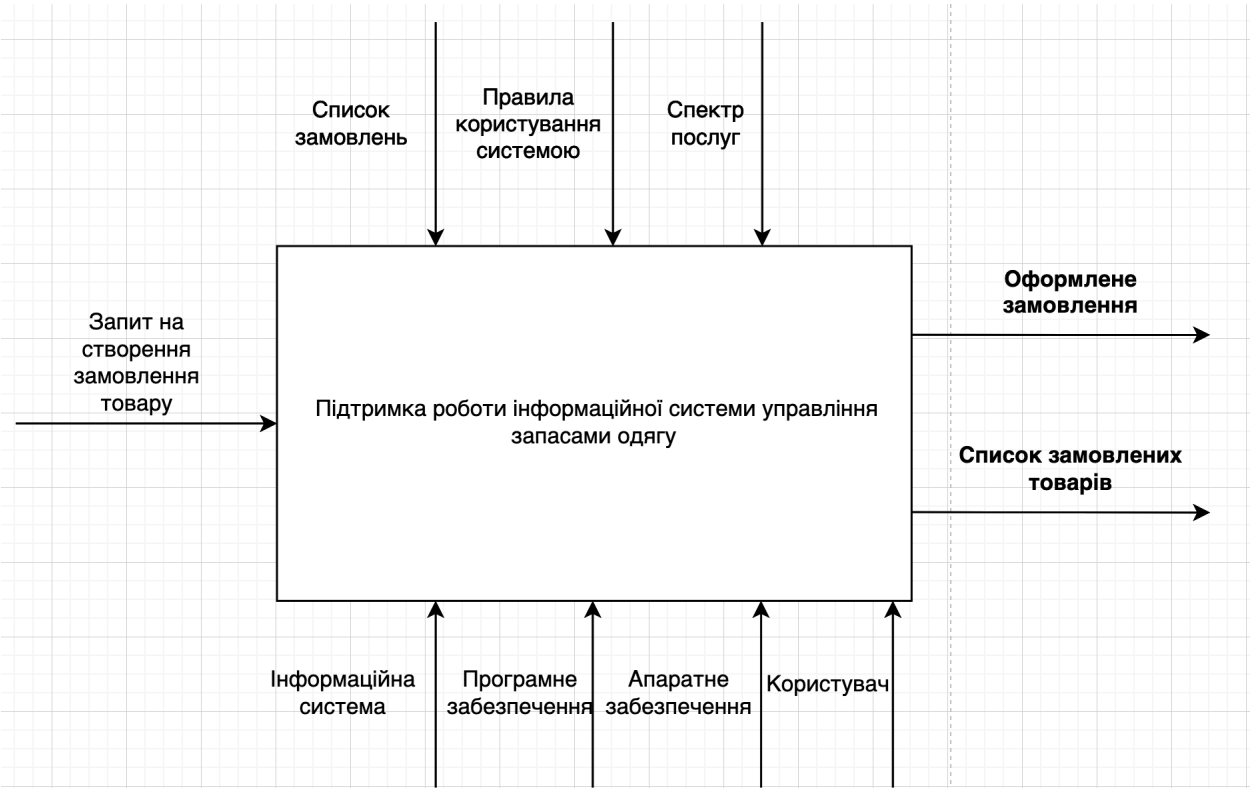


Рисунок 2.1 - Контекстна діаграма процесу підтримки роботи інформаційної системи управління запасами одягу

Функціональна декомпозиція на основі IDEF0 - це процес розбиття складної системи на окремі підфункції для глибшого розуміння її структури. У контексті системи управління запасами для магазинів одягу це дозволяє чітко розмежувати основні процеси, а саме контроль запасів, продаж та аналіз даних. Це спрощує управління кожним етапом процесу, допомагає уникнути перевантаження системи та зменшити складність впровадження аналітичних інструментів [17]. На рисунку 2.2 представлено функціональну декомпозицію процесу управління запасами одягу.

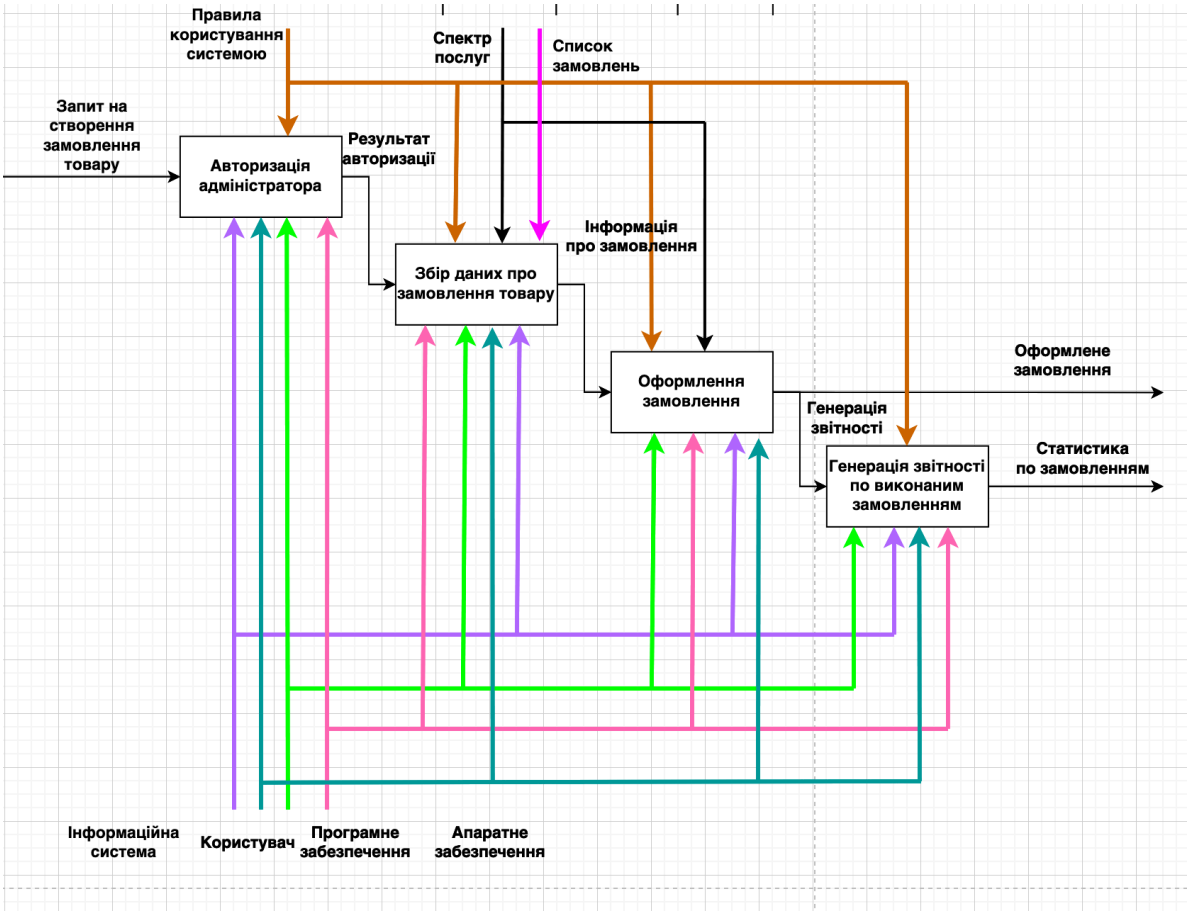


Рисунок 2.2 – Функціональна декомпозиція процесу створення замовлення

2.2 Моделювання варіантів використання

Діаграма варіантів використання - це графічне зображення взаємодії між користувачами та системою, що окреслює дії, які користувачі можуть виконувати в цій системі. У контексті інформаційної системи управління

запасами для магазинів одягу діаграма варіантів використання може бути використана для чіткого розмежування функцій, які будуть доступні користувачам, таких як управління товарами, перегляд аналітики або створення звітів. Такий підхід полегшує розуміння ролей і функцій системи, тим самим спрощуючи процес розробки та оптимізації функціоналу [18]. На рисунку 2.3 представлено діаграму варіантів використання веборієнтованої системи керування будівельними процесами.

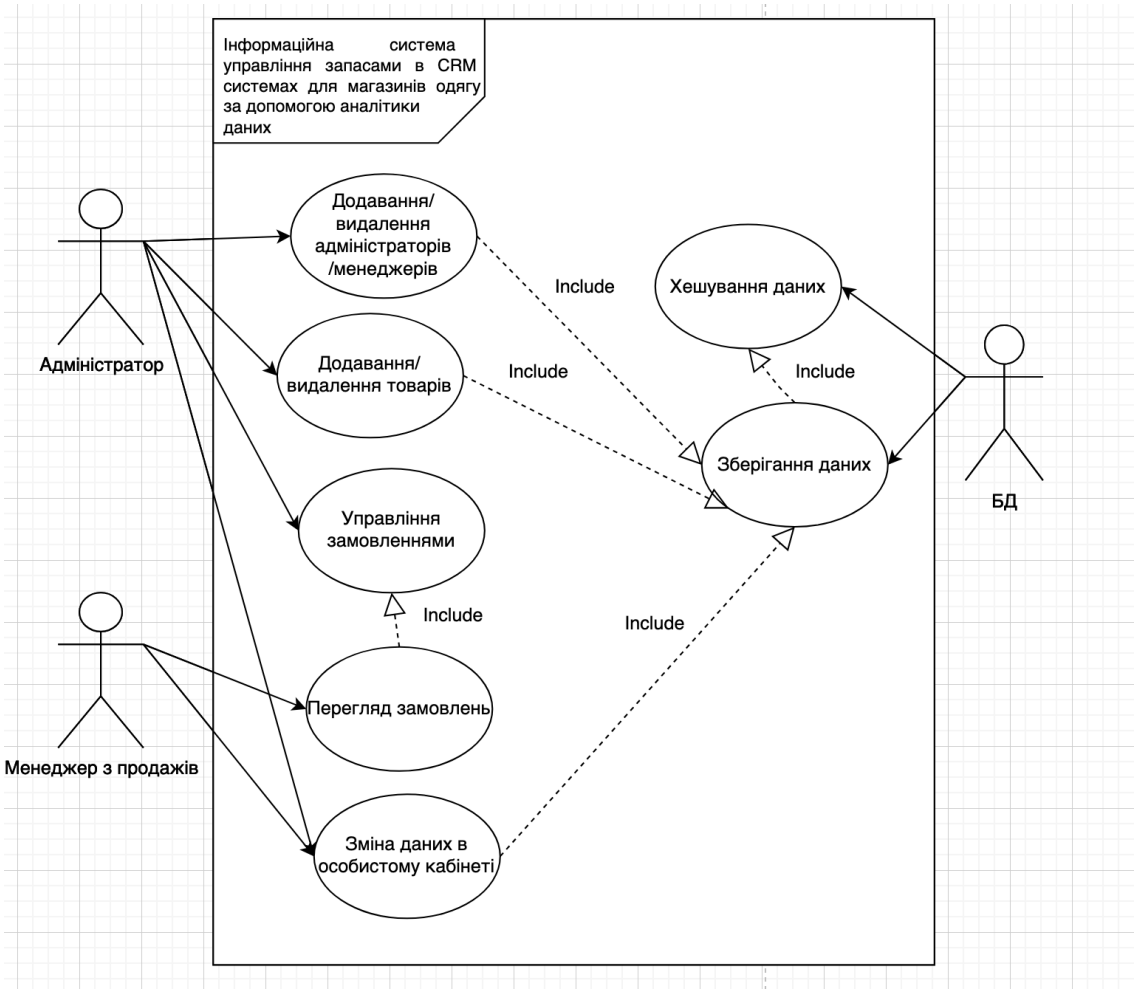


Рисунок 2.3 - Діаграма варіантів використання інформаційної системи управління запасами в CRM системах для магазинів одягу

2.3 Проєктування бази даних

MySQL залишається однією з найпоширеніших систем управління реляційними базами даних (СКБД). Система управління базами даних (СУБД)

є важливим компонентом бекенда, який використовується розробниками для створення, модифікації та вилучення даних з реляційних баз даних. Як система баз даних з відкритим вихідним кодом, вона не впливає на фінансове навантаження веброзробки. Завдяки тривалому використанню протягом кількох десятиліть вона продемонструвала стабільність, надійність і значні можливості [19].

Хоча системи управління контентом (CMS), які використовують плоскі файли, можуть використовуватися для створення вебсайтів без баз даних, слід зазначити, що значна частина вебсайтів і додатків, які зараз використовуються у мережі Інтернет, покладаються на технологію баз даних. З точки зору аспектів безпеки та захисту екосистеми веброзробки, MySQL є однією з найпопулярніших систем управління базами даних. Незалежно від того, чи є дані інформаційними або транзакційними, MySQL залишається надійною системою управління реляційними базами даних (СКБД)

Масштабованість, яку пропонує MySQL, не має аналогів. Ємність необмежена, що полегшує управління вбудованими додатками. Незважаючи на значний обсяг даних, якими можна керувати, MySQL легко адаптується для виконання необхідних функцій. Очевидно, що ця система управління базами даних може похвалитися високим ступенем гнучкості на вимогу. Крім того, вона дозволяє безмежну кастомізацію. Ця особливість є особливо корисною в контексті розробки вебсайтів для електронної комерції.

Щоб допомогти системним адміністраторам у налаштуванні баз даних, MySQL має окрему структуру механізму зберігання даних. Це сприяє оптимальній продуктивності баз даних, що, в свою чергу, забезпечує безперебійне функціонування процесів. Система розроблена таким чином, щоб відповідати вимогам вебсередовища і забезпечувати оптимальну швидкість роботи, незалежно від характеру і розміру вебсайту. Це стосується як вебсайтів середнього розміру, так і набагато більших, які отримують мільйони запитів, а також містять транзакційні дані. MySQL була розроблена, щоб задовольнити вимоги навіть найвимогливіших додатків, забезпечуючи

при цьому наявність повнотекстових індексів та унікальних кешів пам'яті, що підвищує загальну продуктивність [20].

Можливість чудового контролю робочого процесу є однією з найбільш вражаючих можливостей MySQL. Середній час завантаження та встановлення становить менше 30 хвилин. З самого початку MySQL готова до використання. MySQL не залежить від платформи; її можна використовувати з будь-якою з основних операційних систем, включаючи Linux, Microsoft, Macintosh та UNIX. Завдяки комплексному рішенню, здатному до самоуправління та автоматизації, що охоплює такі функції, як розширення простору, конфігурація, дизайн даних та адміністрування бази даних, MySQL є кращим рішенням. Наразі MySQL є оптимальною платформою для транзакційних баз даних, яка наразі доступна. На додаток до вищезгаданих атрибутів, таких як масштабованість і безпека, MySQL пропонує безперебійну підтримку транзакцій [21].

На рисунку 2.4 представлено фізичну модель бази даних інформаційної технології інтеграції системи керування клієнтською взаємодією з платформою електронної комерції та соціальними мережами, яка була реалізована за допомогою phpMyAdmin.

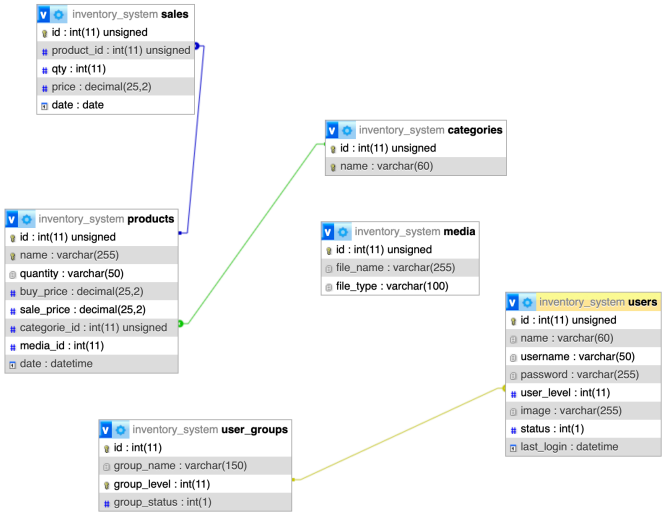


Рисунок 2.4 - Фізична модель бази даних інформаційної технології інтеграції системи керування клієнтською взаємодією з платформою електронної комерції та соціальними мережами

Нарешті, як система з відкритим вихідним кодом, вона не вимагає додаткових витрат. У разі прийняття рішення про міграцію з існуючого додатку бази даних на цю платформу, організація може отримати значну економію. Крім того, процес усунення несправностей є простим і швидким, що сприяє ефективному управлінню.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Архітектура інформаційної технології

Архітектура інформаційної технології інтеграції системи управління взаєминами з клієнтами електронної комерції та соціальними мережами складається з ряду компонентів у вигляді файлів та їх взаємодії, які мають безпосередній вплив на ефективність інформаційної технології, а також на її безпеку, зручність для користувача і здатність до масштабування.

Також архітектура інформаційної технології складається з двох основних категорій компонентів – компонент користувацького інтерфейсу та структурні компоненти. Термін «компоненти програми» охоплює будь-які елементи програми, які впливають на інтерфейс користувача (UI) або користувацький досвід (UX). Сюди входять, зокрема, такі елементи, як інформаційні панелі, сповіщення, журнали активності, налаштування, макети та дизайн. Системою передбачено що структурні компоненти невидимі для кінцевого користувача і включають в себе сервер, логіку або функціональність вебсистеми і базу даних.

Рівень представлення орієнтований на користувача, оскільки він стосується інтерфейсу користувача і способу, в який користувач взаємодіє з системою. Презентаційний рівень, який також називають клієнтською частиною, охоплює інтерфейс користувача (UI), специфічний для конкретного дисплея, а також логіку, що відповідає за взаємодію з браузером, і будь-який код, що полегшує взаємодію з користувачем.

Бізнес-рівень представляє логіку роботи програми, отримуючи дані з рівня збереження і передаючи відповідь назад на рівень представлення відповідно до встановлених бізнес-операцій.

Рівень збереження являє собою наступний логічний рівень, функція якого полягає в тому, щоб полегшити читання, запис і маніпуляції з базою

даних. Як і у випадку з іншими рівнями, рівень збереження можна назвати більш описовою назвою, наприклад, рівень доступу до даних.

Рівень бази даних являє собою місце зберігання всіх даних, яке може бути реалізоване за допомогою різних систем управління базами даних, таких як SQL Server або MongoDB.

На рисунку 3.1 представлено вигляд рівнів архітектури інформаційної технології.

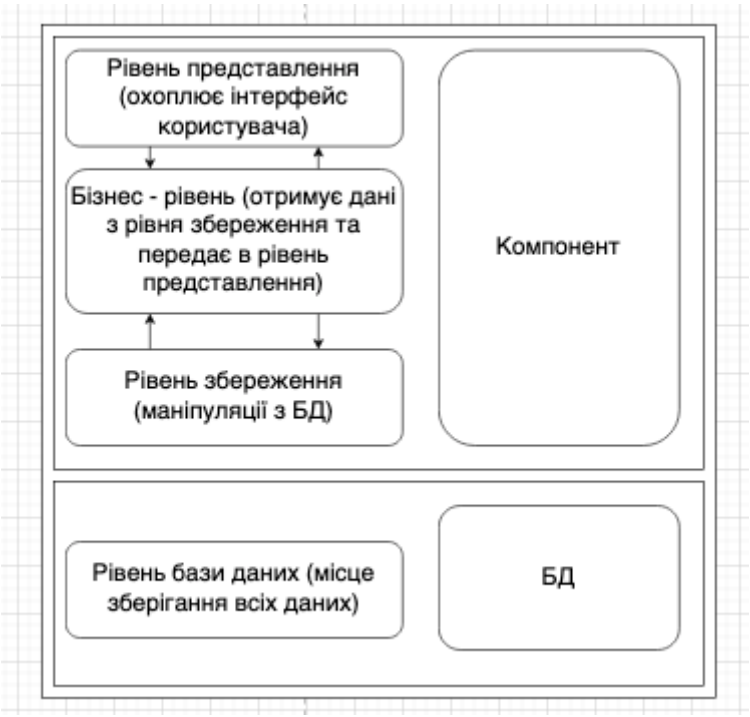


Рисунок 3.1 – Вигляд рівнів архітектури інформаційної технології

Рендеринг на стороні сервера передбачає компіляцію даних сервером і доставку клієнту повністю заповненої HTML-сторінки. Для інформаційної технології інтеграції системи управління взаєминами з клієнтами з платформами електронної комерції та соціальними мережами даний спосіб подачі даних надасть змогу чітко відповідати на запити користувача та швидко надавати результат обробки запиту.

На рисунку 3.2 представлено вигляд архітектури інформаційної технології інтеграції системи управління взаєминами з клієнтами з платформами електронної комерції та соціальними мережами.

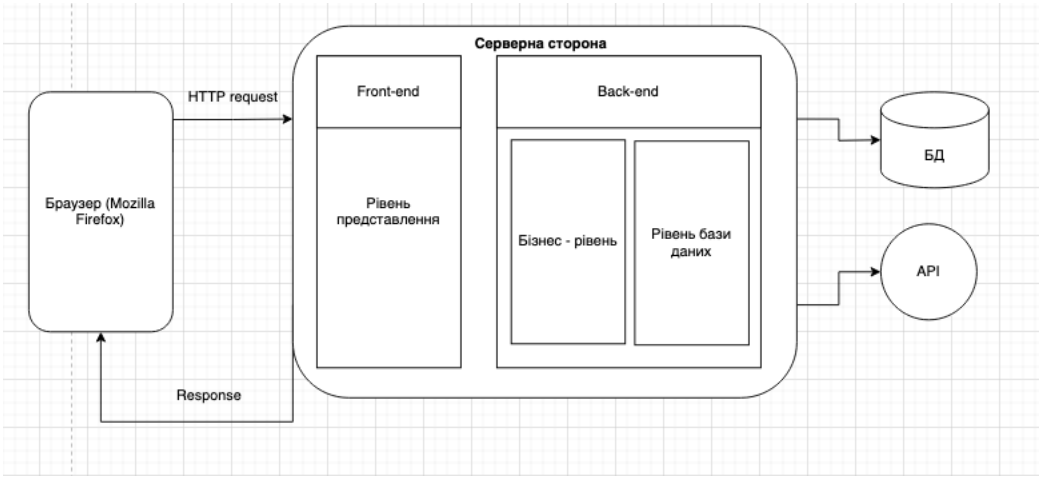


Рисунок 3.2 – Вигляд архітектури інформаційної технології інтеграції системи управління взаєминами з клієнтами з платформами електронної комерції та соціальними мережами

PHP – відносно проста у розробці та підтримці мова. Крім того, вона пропонує один з найвищих рівнів продуктивності, що сприяє її статусу як популярного вибору для розробки серверів. В поєднанні з MySQL, PHP надає всі інструменти за допомогою яких можна реалізувати інформаційну технологію інтеграції системи управління взаєминами з клієнтами з платформами електронної комерції та соціальними мережами.

3.2 Реалізація БД та інтеграція в систему

При роботі інформаційної технології необхідно звертатися до бази даних та отримувати дані за допомогою різних запитів. Тому необхідно розробити та імплементувати базу даних в систему. Для цього було обрано інструмент phpMyAdmin, який вбудований в локальний сервер XAMPP.

Щоб запустити phpMyAdmin на локальному сервері, необхідно в браузері перейти за адресою localhost/phpmyadmin. Після цього необхідно в полі «create new database» надати назву БД та тип кодування. Після того, як назва і тип кодування були сформовані, необхідно створити таблиці з атибутами. Вміст таблиць в базі даних представлено в таблиці 3.1.

Таблиця 3.1 – Вміст таблиць в базі даних інформаційної технології

Назва таблиці	**Атрибути**	**Призначення таблиці**
categories	id - int(11) primary key name - varchar(255)	Зберігає інформацію про категорії товарів
media	id - int(11) primary key file_name - varchar(255) file type - varchar(255)	Зберігає інформацію про медіа файли, які відносяться до товарів
products	id - int(11) primary key name - varchar(255) quantity - varchar(255) buy_price – decimal (25,2) sale_price - decimal (25,2) categorie_id - int(11) foreigh key media_id - int(11) foreigh key date - datetime	Зберігає інформацію про товари, з детальним описом
sales	id - int(11) primary key product_id - int(11) foreigh key qty - int(11) price - int(11) date - datetime	Зберігає інформацію про продажі з детальим описом
users	id - int(11) primary key name - varchar(255) username - varchar(255) password - varchar(255) user_level – int(11) image - varchar(255) status – int(1) last_login - datetime	Зберігає інформацію про користувачів системи
user_groups	id - int(11) primary key group_name - varchar(255) group_level - int(1) group_status - int(1)	Зберігає інформацію про тип користувачів

На рисунках 3.3 – 3.8 представлено структури створених таблиць бази даних «inventory_system».

Структура таблиці | Вид відносин

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int(11)		UNSIGNED	Ні	*Немає*		AUTO_INCREMENT	Змінити Знищити Більше
2	name	varchar(60)	utf8_general_ci		Ні	*Немає*			Змінити Знищити Більше

Рисунок 3.3 – Структура таблиці «categories»

Структура таблиці | Вид відносин

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int(11)		UNSIGNED	Ні	*Немає*		AUTO_INCREMENT	Змінити Знищити Більше
2	file_name	varchar(255)	latin1_swedish_ci		Ні	*Немає*			Змінити Знищити Більше
3	file_type	varchar(100)	latin1_swedish_ci		Ні	*Немає*			Змінити Знищити Більше

Рисунок 3.4 – Структура таблиці «media»

Структура таблиці | Вид відносин

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int(11)		UNSIGNED	Ні	*Немає*		AUTO_INCREMENT	Змінити Знищити Більше
2	name	varchar(255)	utf8_general_ci		Ні	*Немає*			Змінити Знищити Більше
3	quantity	varchar(50)	utf8_general_ci		Так	*NULL*			Змінити Знищити Більше
4	buy_price	decimal(25,2)			Так	*NULL*			Змінити Знищити Більше
5	sale_price	decimal(25,2)			Ні	*Немає*			Змінити Знищити Більше
6	categorie_id	int(11)		UNSIGNED	Ні	*Немає*			Змінити Знищити Більше
7	media_id	int(11)			Так	0			Змінити Знищити Більше
8	date	datetime			Ні	*Немає*			Змінити Знищити Більше

Рисунок 3.5 – Структура таблиці «products»

Структура таблиці | Вид відносин

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int(11)		UNSIGNED	Ні	*Немає*		AUTO_INCREMENT	Змінити Знищити Більше
2	product_id	int(11)		UNSIGNED	Ні	*Немає*			Змінити Знищити Більше
3	qty	int(11)			Ні	*Немає*			Змінити Знищити Більше
4	price	decimal(25,2)			Ні	*Немає*			Змінити Знищити Більше
5	date	date			Ні	*Немає*			Змінити Знищити Більше

Рисунок 3.6 – Структура таблиці «sales»

Структура таблиці | Вид відносин

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int(11)		UNSIGNED	Ні	*Немає*		AUTO_INCREMENT	Змінити Знищити Більше
2	name	varchar(60)	latin1_swedish_ci		Ні	*Немає*			Змінити Знищити Більше
3	username	varchar(50)	latin1_swedish_ci		Ні	*Немає*			Змінити Знищити Більше
4	password	varchar(255)	latin1_swedish_ci		Ні	*Немає*			Змінити Знищити Більше
5	user_level	int(11)			Ні	*Немає*			Змінити Знищити Більше
6	image	varchar(255)	latin1_swedish_ci		Так	no_image.jpg			Змінити Знищити Більше
7	status	int(1)			Ні	*Немає*			Змінити Знищити Більше
8	last_login	datetime			Так	*NULL*			Змінити Знищити Більше

Рисунок 3.7 – Структура таблиці «users»

Структура таблиці | Вид відносин

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int(11)			Ні	*Немає*		AUTO_INCREMENT	Змінити Знищити Більше
2	group_name	varchar(150)	latin1_swedish_ci		Ні	*Немає*			Змінити Знищити Більше
3	group_level	int(11)			Ні	*Немає*			Змінити Знищити Більше
4	group_status	int(1)			Ні	*Немає*			Змінити Знищити Більше

Рисунок 3.8 – Структура таблиці «user_groups»

Після цього необхідно створити файл config.php та описати дані для підключення до бази даних. На рисунку 3.9 представлено вигляд файлу config.php.

```
1  <?php
2
3    define( 'DB_HOST', 'localhost' );          // Set database host
4    define( 'DB_USER', 'root' );             // Set database user
5    define( 'DB_PASS', '' );              // Set database password
6    define( 'DB_NAME', 'inventory_system' );        // Set database name
7
8  ?>
9
```

Рисунок 3.9 – Вигляд файлу config.php

Після цього було створено файл database.php, який забезпечує підключення до бази даних MySQL, виконання запитів та обробку результатів. Також в файлі використовуються методи fetch_array(), fetch_object() та fetch_assoc() які відслідковують дані в різних форматах. Також файл містить захист від SQL-ін`єкцій через метод escape($str).

На рисунку 3.10 представлено вигляд файлу database.php. Повний лістинг коду представлено в додатку Б.

```php
<?php
require_once(LIB_PATH_INC.DS."config.php");

class MySqli_DB {

    private $con;
    public $query_id;

    function __construct() {
      $this->db_connect();
    }


public function db_connect()
{
  $this->con = mysqli_connect(DB_HOST,DB_USER,DB_PASS);
  if(!$this->con)
        {
          die(" Database connection failed:". mysqli_connect_error());
        } else {
          $select_db = $this->con->select_db(DB_NAME);
            if(!$select_db)
            {
              die("Failed to Select Database". mysqli_connect_error());
            }
        }
}


public function db_disconnect()
{
  if(isset($this->con))
  {
    mysqli_close($this->con);
    unset($this->con);
  }
}

public function query($sql)
  {

     if (trim($sql != "")) {
         $this->query_id = $this->con->query($sql);
      }
```

Рядок 10, Стовпець 27 — 99 Рядків | ВСТ | UTF-8 | PHP | Пробіли: 4

Рисунок 3.10 - Вигляд файлу database.php

Після реалізації бази даних необхідно описати запити та функції для обробки даних в серверній частині.

3.3 Реалізація серверної частини інформаційної технології

На початку реалізації серверної частини було створено файл load.php, який відповідає за встановлення основних констант та підключає необхідні файли для системи. На рисунку 3.11 представлено вигляд файлу load.php.

```php

<?php

define("URL_SEPARATOR", '/');

define("DS", DIRECTORY_SEPARATOR);


defined('SITE_ROOT')? null: define('SITE_ROOT', realpath(dirname(__FILE__)));
define("LIB_PATH_INC", SITE_ROOT.DS);


require_once(LIB_PATH_INC.'config.php');
require_once(LIB_PATH_INC.'functions.php');
require_once(LIB_PATH_INC.'session.php');
require_once(LIB_PATH_INC.'upload.php');
require_once(LIB_PATH_INC.'database.php');
require_once(LIB_PATH_INC.'sql.php');

?>

```

Рисунок 3.11 – Вигляд файлу load.php

Для виконання функцій очищення тексту, перевірки полів, форматування повідомлень, підрахунку прибутку та ін. було створено файл functions.php. Результат створення предтавлено на рисунку 3.12. Повний лістинг коду представлено в додатку В.

```php
<?php
 $errors = array();


function real_escape($str){
  global $con;
  $escape = mysqli_real_escape_string($con,$str);
  return $escape;
}

function remove_junk($str){
  $str = nl2br($str);
  $str = htmlspecialchars(strip_tags($str, ENT_QUOTES));
  return $str;
}

function first_character($str){
  $val = str_replace('-'," ",$str);
  $val = ucfirst($val);
  return $val;
}

function validate_fields($var){
  global $errors;
  foreach ($var as $field) {
    $val = remove_junk($_POST[$field]);
    if(isset($val) && $val==''){
      $errors = $field ." can't be blank.";
      return $errors;
    }
  }
}

function display_msg($msg =''){
   $output = array();
  if(!empty($msg)) {
     foreach ($msg as $key => $value) {
        $output  = "<div class=\"alert alert-{$key}\">";
        $output .= "<a href=\"#\" class=\"close\" data-dismiss=\"alert\">&times;</a>";
        $output .= remove_junk(first_character($value));
        $output .= "</div>";
     }
     return $output;
  } else {
```

Рисунок 3.12 – Вигляд файлу functions.php

Також для більш організованої роботи інформаційної технології було створено файл sql.php, який визначає ряд функцій для загальних операцій (видалення, оновлення, додавання, пошук, підрахунок), а також обробки авторизації та автентифікації користувачів. Результат створення файлу представлено на рисунку 3.13.

```
127
128   function updateLastLogIn($user_id)
129      {
130          global $db;
131      $date = make_date();
132      $sql = "UPDATE users SET last_login='{$date}' WHERE id ='{$user_id}' LIMIT 1";
133      $result = $db->query($sql);
134      return ($result && $db->affected_rows() === 1 ? true : false);
135      }
136
137
138    function find_by_groupName($val)
139    {
140      global $db;
141      $sql = "SELECT group_name FROM user_groups WHERE group_name = '{$db->escape($val)}'
       LIMIT 1 ";
142      $result = $db->query($sql);
143      return($db->num_rows($result) === 0 ? true : false);
144    }
145
146    function find_by_groupLevel($level)
147    {
148      global $db;
149      $sql = "SELECT group_level FROM user_groups WHERE group_level = '{$db->escape($level)}'
       LIMIT 1 ";
150      $result = $db->query($sql);
151      return($db->num_rows($result) === 0 ? true : false);
152    }
153
154     function page_require_level($require_level){
155       global $session;
156       $current_user = current_user();
157       $login_level = find_by_groupLevel($current_user['user_level']);
158       //if user not login
159       if (!$session->isUserLoggedIn(true)):
160              $session->msg('d','Please login...');
161              redirect('index.php', false);
162        //if Group status Deactive
163       elseif($login_level['group_status'] === '0'):
164             $session->msg('d','This level user has been band!');
165             redirect('home.php',false);
166        //cheackin log in User level and Require level is Less than or equal to
167       elseif($current_user['user_level'] <= (int)$require_level):
168                return true;
169        else:
```

Рядок 88, Стовпець 12 — 298 Рядків | ВСТ | UTF-8 | PHP | Пробіли: 4

Рисунок 3.13 – Вигляд файлу sql.php

Після цього можна переходити до реалізації клієнтської частини для створення сторінок з виконанням CRUD операцій.

3.4 Реалізація клієнтської частини інформаційної технології

Для клієнтської частини було створено файли для редагування, додавання, видалення та оновлення інформації. На прикладі файлів

add_product.php, delete_product.php, edit_product.php було описано принцип роботи інформаційної технології в поєднанні з серверною частиною.

Файл add_product.php реалізує сторінку для додавання нового товару в систему. Сервер виконує обробку та збереження даних, а на стороні клаєнта відбувається відображення форми додавання товару.

Адміністратор заповнює поля форми та за допомогою методу POST відправляє запит на сторінку add_product.php. Поля містять назву товару, категорію, к-сть, ціну купівлі та ціну продажу. Для більш детального опису система надає можливість підвантажити фото з бази даних та обрати категорію. Якщо форма відправлена з помилками або некоректними даними то користувач отримує повідомлення.

Для перевірки рівня доступу було створено функцію page_require_level(), і якщо користувач не відповідає ролі під номером 2 то доступ до системи буде заблокований. Після відправки форми, масив $req_fields визначає обов`язкові поля. При знаходженні порожнього поля, скрипт формує помилку і сповіщує про це користувача. Дані з полів форми обробляються за допомогою функції remove_junk() для виявлення шкідливих символів.

На рисунку 3.14 представлено вигляд файлу add_product.php. Повний лістинг коду представлено в додатку Г.

```php
<?php
  $page_title = 'Add Product';
  require_once('includes/load.php');
  // Checkin What level user has permission to view this page
  page_require_level(2);
  $all_categories = find_all('categories');
  $all_photo = find_all('media');
?>
<?php
 if(isset($_POST['add_product'])){
   $req_fields = array('product-title','product-categorie','product-quantity','buying-
   price', 'saleing-price' );
   validate_fields($req_fields);
   if(empty($errors)){
     $p_name  = remove_junk($db->escape($_POST['product-title']));
     $p_cat   = remove_junk($db->escape($_POST['product-categorie']));
     $p_qty   = remove_junk($db->escape($_POST['product-quantity']));
     $p_buy   = remove_junk($db->escape($_POST['buying-price']));
     $p_sale  = remove_junk($db->escape($_POST['saleing-price']));
     if (is_null($_POST['product-photo']) || $_POST['product-photo'] === "") {
       $media_id = '0';
     } else {
       $media_id = remove_junk($db->escape($_POST['product-photo']));
     }
     $date    = make_date();
     $query  = "INSERT INTO products (";
     $query .=" name,quantity,buy_price,sale_price,categorie_id,media_id,date";
     $query .=") VALUES (";
     $query .=" '{$p_name}', '{$p_qty}', '{$p_buy}', '{$p_sale}', '{$p_cat}',
     '{$media_id}', '{$date}'";
     $query .=")";
     $query .=" ON DUPLICATE KEY UPDATE name='{$p_name}'";
     if($db->query($query)){
       $session->msg('s',"Product added ");
       redirect('add_product.php', false);
     } else {
       $session->msg('d',' Sorry failed to added!');
       redirect('product.php', false);
     }

   } else{
     $session->msg("d", $errors);
     redirect('add_product.php',false);
   }
```

Рядок 17, Стовпець 24 — 135 Рядків | ВСТ | UTF-8 | PHP | Пробіли: 4

Рисунок 3.14 – Вигляд файлу add_product.php

Файл delete_product.php реалізовує функціонал для видалення товару з БД. Користувач, який відповідає доступу до системи 2 може видалити продукт за його унікальним номером.

На стороні клієнта даний скрипт виконується через URL-посилання або кнопку з інших сторінок, яка перенаправляє користувача на цей скрипт. При натисканні на кнопку, система перенаправляє на URL з унікальним ключем товару та виконує функцію видалення.

Сервер перевіряє рівень доступу користувача, і якщо від дозволений то отримує id товару та може перевірити його наявність в системі. Якщо товар існує в системі, то сервер виконує SQL-запит, та повідомляє користувача про результат операції

На рисунку 3.15 представлено вигляд файлу delete_product.php. Повний лістинг коду представлено в додатку Г.

```php
<?php
  require_once('includes/load.php');
  // Checkin What level user has permission to view this page
  page_require_level(2);
?>
<?php
  $product = find_by_id('products',(int)$_GET['id']);
  if(!$product){
    $session->msg("d","Missing Product id.");
    redirect('product.php');
  }
?>
<?php
  $delete_id = delete_by_id('products',(int)$product['id']);
  if($delete_id){
      $session->msg("s","Products deleted.");
      redirect('product.php');
  } else {
      $session->msg("d","Products deletion failed.");
      redirect('product.php');
  }
?>
```

Рисунок 3.15 – Вигляд файлу delete_product.php

Для реалізації функціоналу редагування товару було створено файл edit_product.php. Клієнтська частина подається в вигляді форми, в яку завантажуються поточні дані про товар. Кнопка «Update» надає можливість підтвердити зміни та надіслати зміни на сервер за допомогою методу POST. Будь-які повідомлення про успіх або помилку оновлення товару виводяться над формою. Це допомагає користувачу бачити результати дій.

Серверна частина відповідає за перевірку рівня доступу користувача, завантаження товару з БД та оновлення інформації в БД. Також сервер відповідає за перевірку даних від SQL-ін`єкцій, а потім в разі успішної перевірки оновлює товар за допомогою методу UPDATE.

На рисунку 3.16 представлено вигляд файлу edit_product.php. Повний лістинг коду представлено в додатку Г.

```php
<?php
  $page_title = 'Edit product';
  require_once('includes/load.php');
  // Checkin What level user has permission to view this page
   page_require_level(2);
?>
<?php
$product = find_by_id('products',(int)$_GET['id']);
$all_categories = find_all('categories');
$all_photo = find_all('media');
if(!$product){
  $session->msg("d","Missing product id.");
  redirect('product.php');
}
?>
<?php
 if(isset($_POST['product'])){
    $req_fields = array('product-title','product-categorie','product-quantity','buying-price', 'saleing-price' );
    validate_fields($req_fields);

   if(empty($errors)){
       $p_name  = remove_junk($db->escape($_POST['product-title']));
       $p_cat   = (int)$_POST['product-categorie'];
       $p_qty   = remove_junk($db->escape($_POST['product-quantity']));
       $p_buy   = remove_junk($db->escape($_POST['buying-price']));
       $p_sale  = remove_junk($db->escape($_POST['saleing-price']));
       if (is_null($_POST['product-photo']) || $_POST['product-photo'] === "") {
         $media_id = '0';
       } else {
         $media_id = remove_junk($db->escape($_POST['product-photo']));
       }
       $query   = "UPDATE products SET";
       $query  .=" name ='{$p_name}', quantity ='{$p_qty}',";
       $query  .=" buy_price ='{$p_buy}', sale_price ='{$p_sale}', categorie_id ='{$p_cat}',media_id='{$media_id}'";
       $query  .=" WHERE id ='{$product['id']}'";
       $result = $db->query($query);
               if($result && $db->affected_rows() === 1){
                 $session->msg('s',"Product updated ");
                 redirect('product.php', false);
               } else {
                 $session->msg('d',' Sorry failed to updated!');
                 redirect('edit_product.php?id='.$product['id'], false);
```


Рисунок 3.16 – Вигляд файлу edit_product.php

За схожим алгоритмом було створено інші файли, які відповідають за операції CRUD щодо таблиць БД.

3.5 Можливості з використання

Для того щоб почати використовувати систему, необхідно авторизуватися. Системою передбачено що головний адміністратор сам додає та призначає ролі користувачам. На рисунках 3.17 представлено вигляд сторінки авторизації.

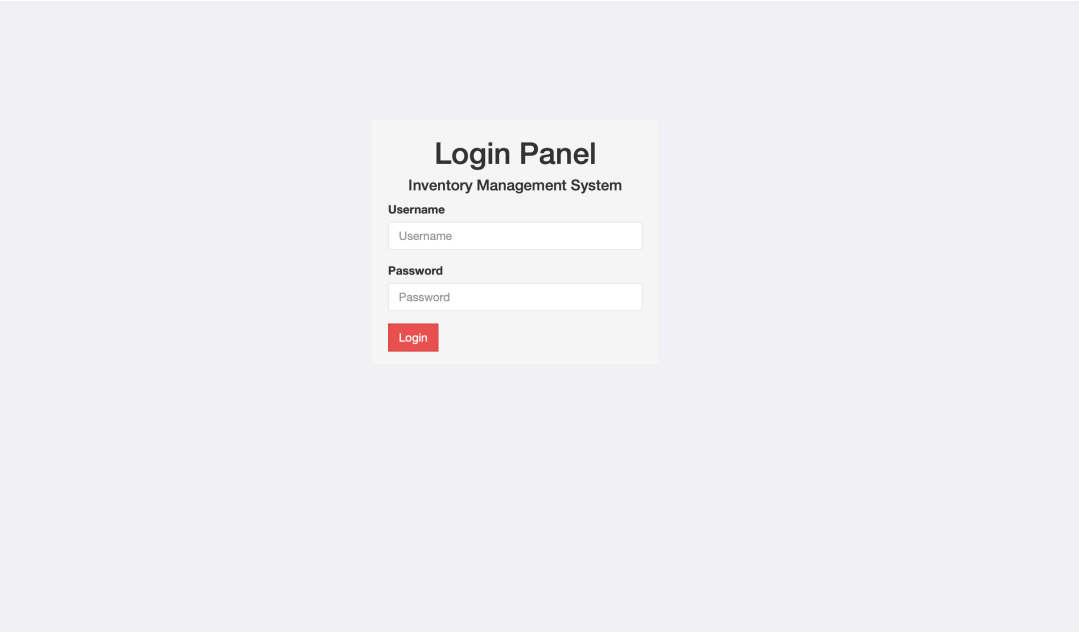


Рисунок 3.17 – Вигляд сторінки авторизації

Після успішної авторизації користувач відповідно до своєї ролі отримує права доступу. На головній сторінці системи розташовано бокове меню, модулі які надають загальну інформацію щодо к-сті користувачів та проданих товарів та меню для переходу на власну сторінку.

На рисунку 3.18 представлено вигляд головної сторінки інформаційної системи.

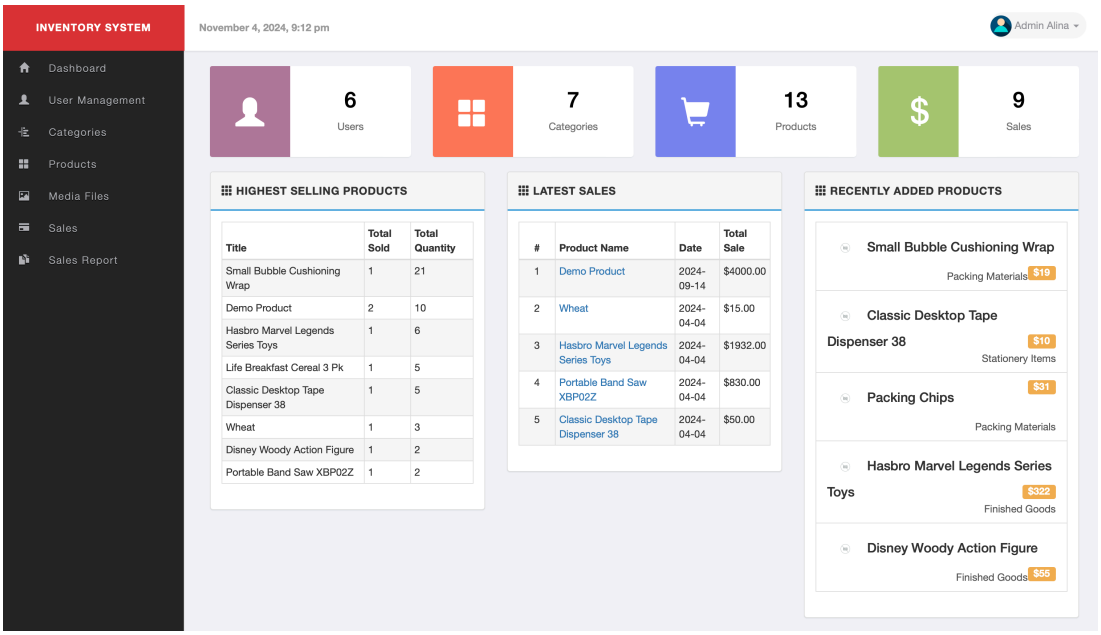


Рисунок 3.18 – Вигляд головної сторінки інформаційної системи

Бокове меню в розгорнутому виді має наступний вигляд (рис. 3.19).

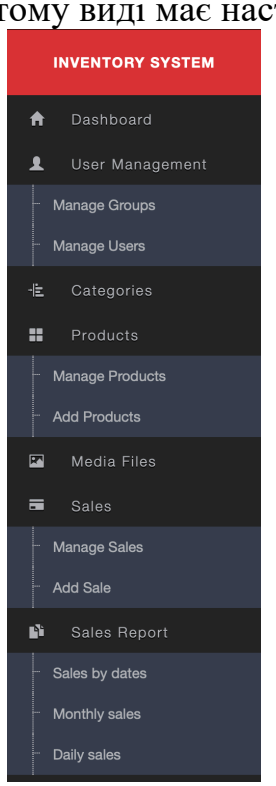


Рисунок 3.19 – Вигляд розгорнутого меню інформаційної системи

Вкладка «User Management» містить в собі сторінки для керування особистими даними користувачів та розподілу груп. За допомогою даних сторінок адміністратор може додати нового користувача та призначити йому роль.

На рисунках 3.20 – 3.21 представлено вигляд сторінок “Manage Groups” та “Manage Users”.

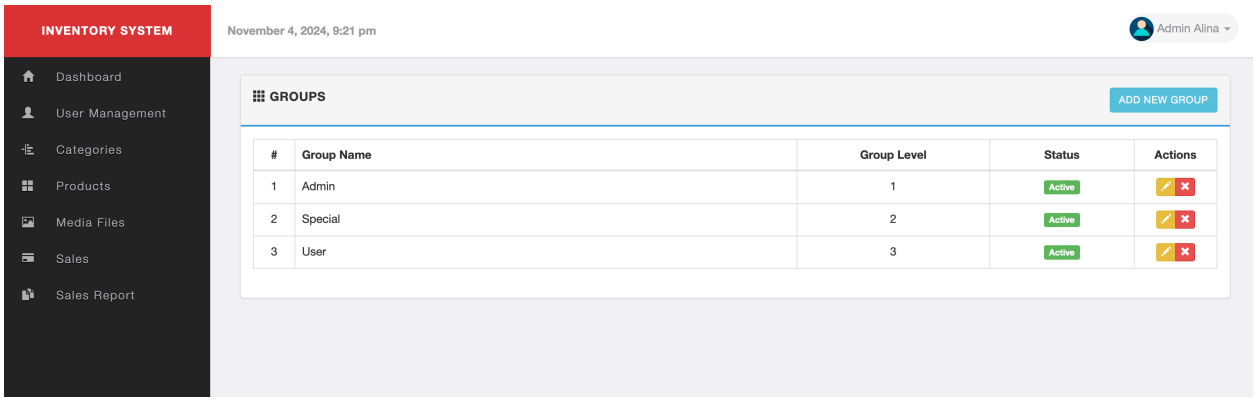


Рисунок 3.20 – Вигляд сторінки “Manage Groups”

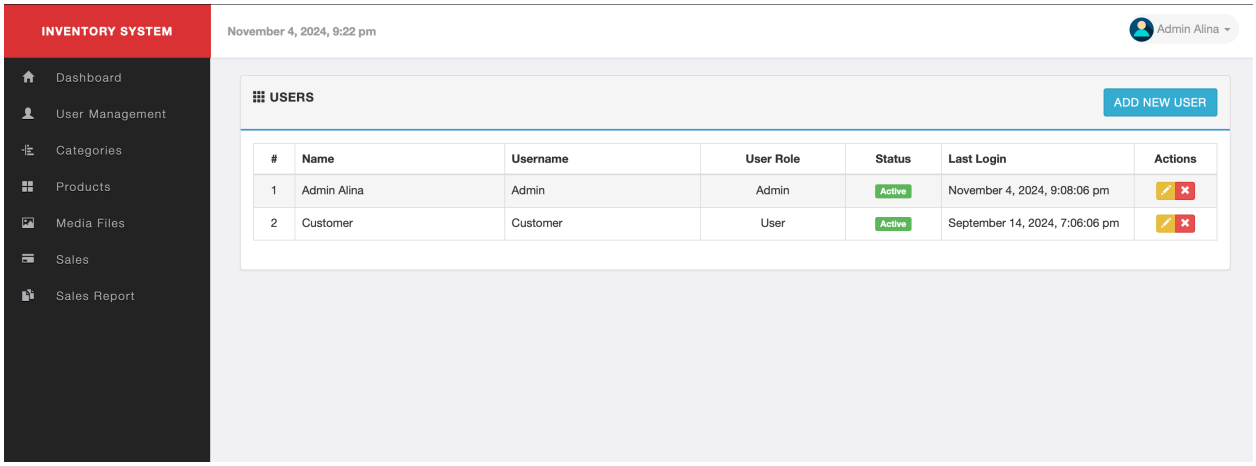


Рисунок 3.21 – Вигляд сторінки “Manage Users”

Для додавання користувача необхідно скористуватися HTML формою. Також під час додавання користувача адміністратор може призначити роль, відповідно до призначення.

На рисунках 3.22 – 3.23 представлено вигляд та результат успішного додавання користувача до системи.

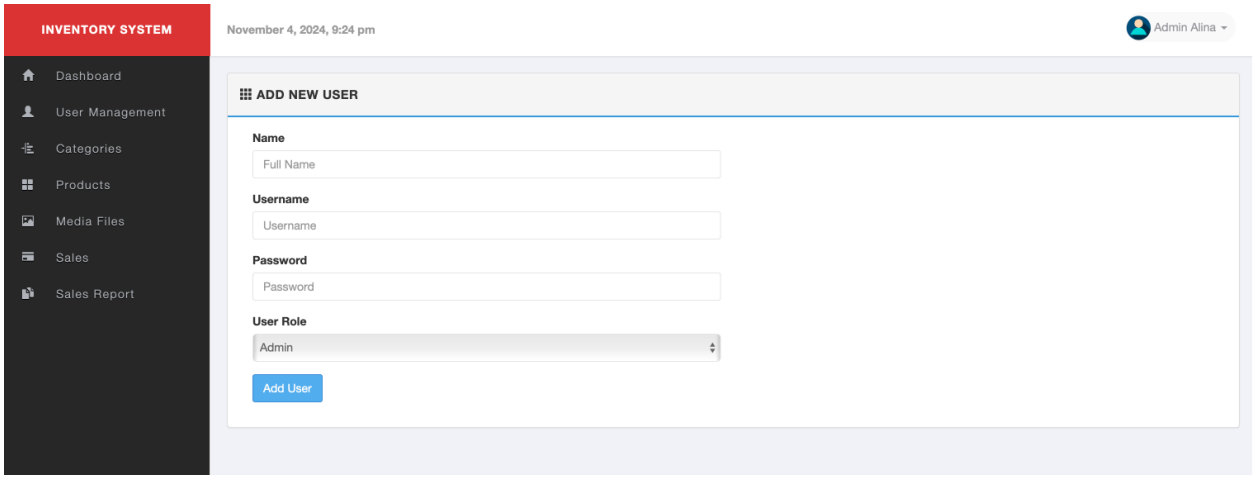


Рисунок 3.22 – Вигляд сторінки додавання нового користувача

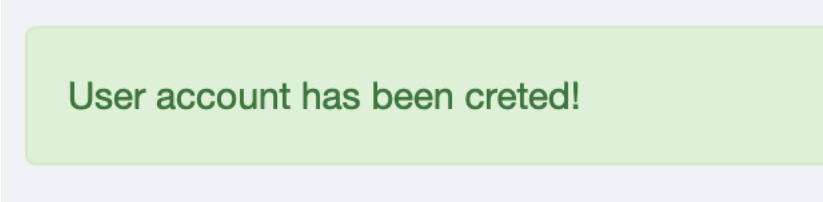


Рисунок 3.23 – Результат успішного додавання користувача

Сторінка «Categories» містить інформацію про наявні категорії та форму для додавання нових категорій товару. На рисунку 3.24 представлено вигляд сторінки «Categories».

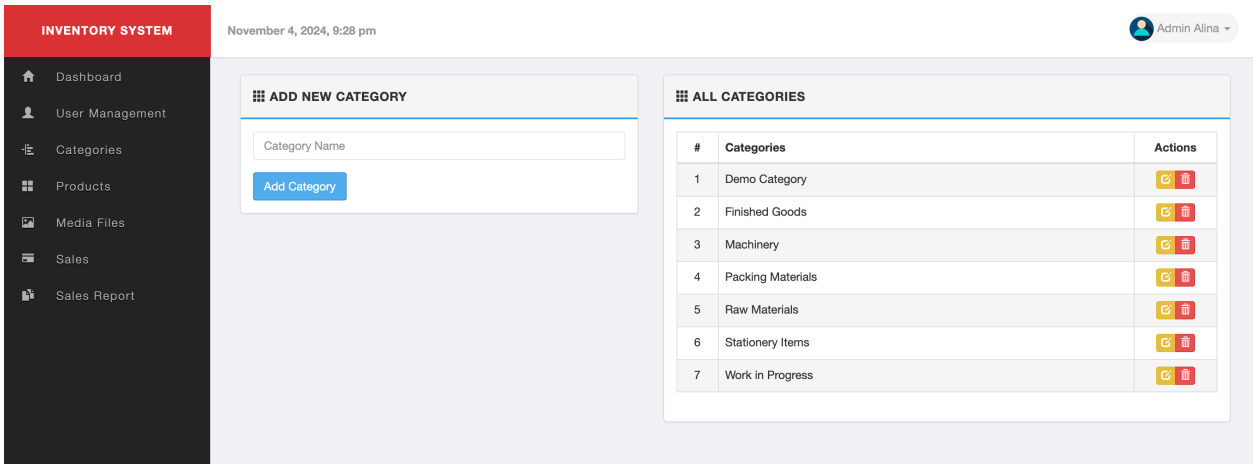


Рисунок 3.24 – Вигляд сторінки «Categories»

Сторінка «Manage Products» містить інформацію для перегляду, редагування та видалення товарів у системі. Сторінка дозволяє адміністраторам переглядати повний список товарів, отримувати доступ до детальної інформації про кожну позицію, а також швидко переходити до редагування або видалення конкретних записів.

На рисунку 3.25 представлено вигляд сторінки «Manage Products».

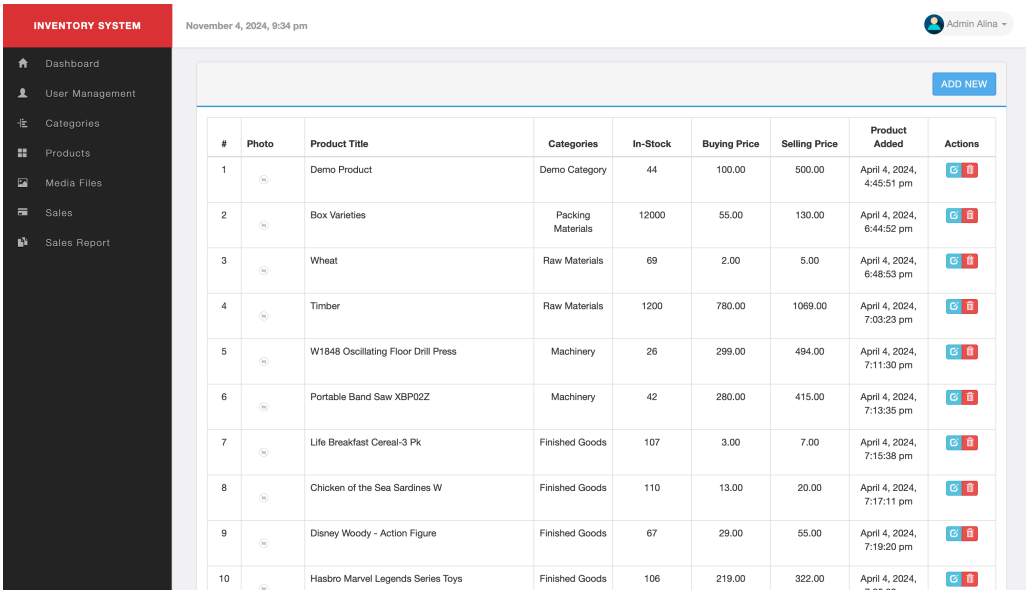


Рисунок 3.25 - Вигляд сторінки «Manage Products»

Сторінка «Add Products». призначена для додавання нових товарів до бази даних. Сторінка містить форму, за допомогою якої адміністратор може ввести назву товару, категорію, кількість, ціну покупки, ціну продажу та завантажити зображення. Після цього новий товар додається до системи і стає доступним у списку товарів на сторінці «Manage Products».

На рисунку 3.26 представлено вигляд сторінки «Add Products».

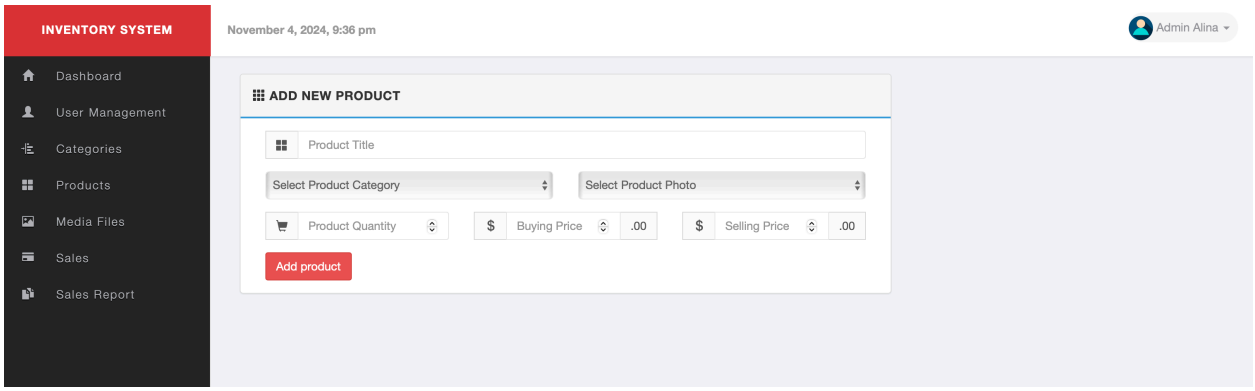


Рисунок 3.26 - Вигляд сторінки «Add Products»

Сторінка «Media Files» надає можливість адміністратору завантажити фото та використовувати їх повторно для опису товарів та ін. На рисунку 3.27 представлено вигляд сторінки «Media Files».

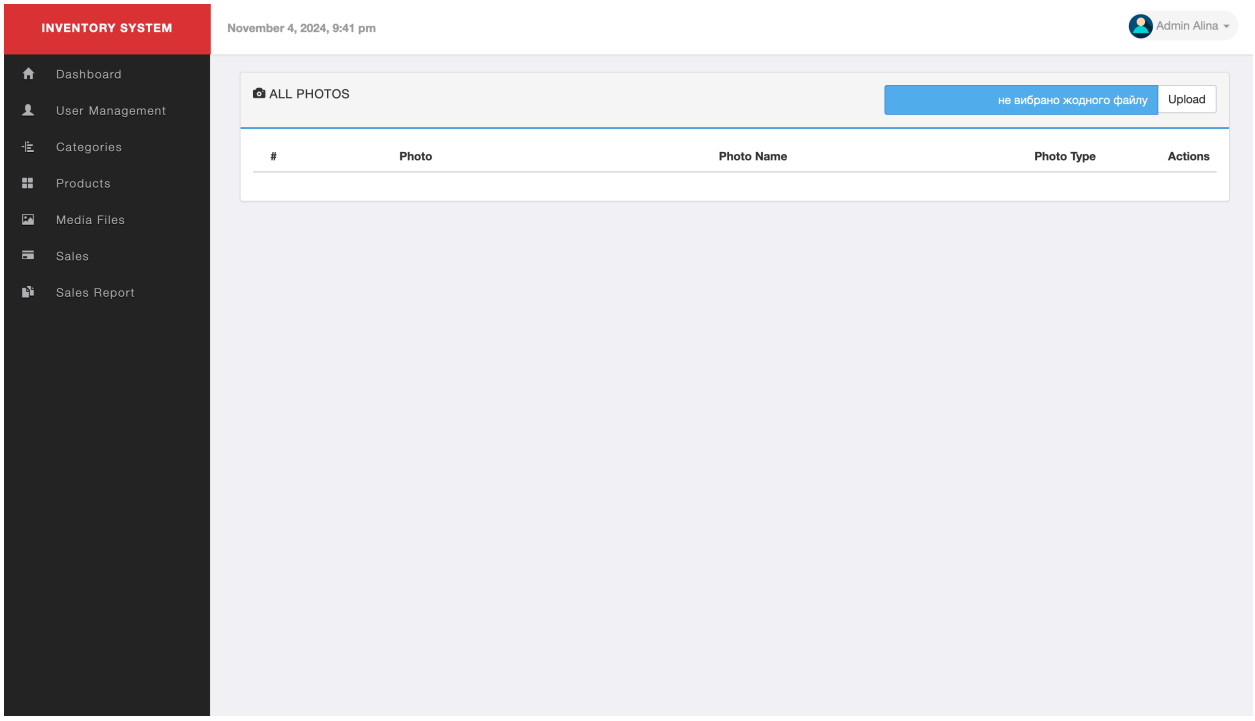


Рисунок 3.27 - Вигляд сторінки «Media Files»

Сторінка «Manage Sales» надає функціонал для управління продажами товарів, де адміністратор може виконувати CRUD операції з записами про продажі. Сторінка «Add Sales» дозволяє адміністратору виконати пошук товару для продажу та зареєструвати транзакцію в системі.

На рисунках 3.28 – 3.29 представлено вигляд сторінок «Manage Sales» та «Add Sales».

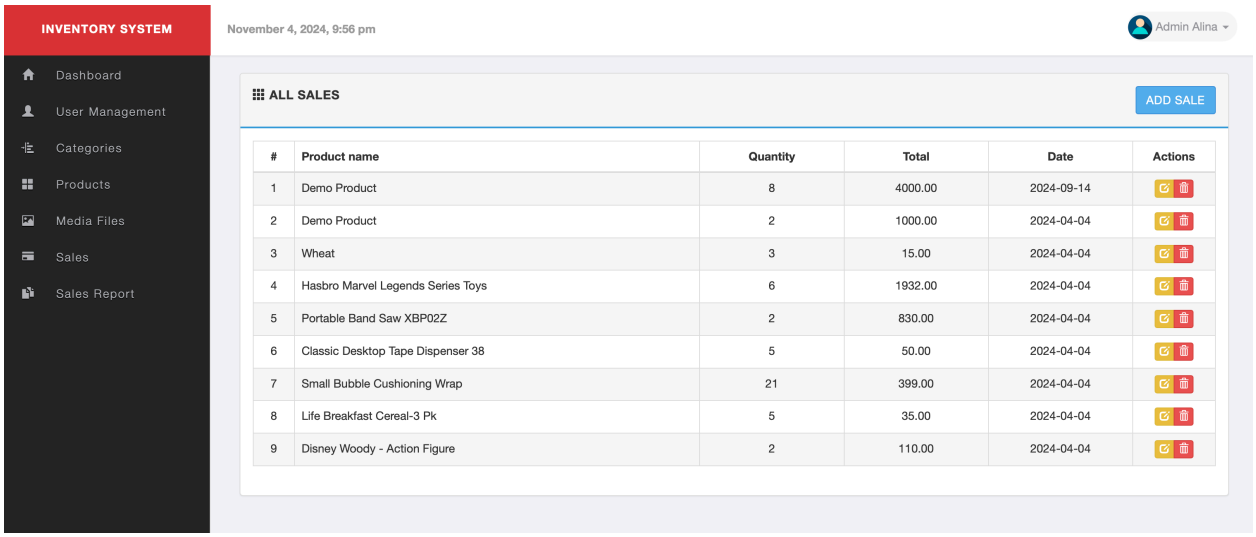


Рисунок 3.28 – Вигляд сторінки «Manage Sales»

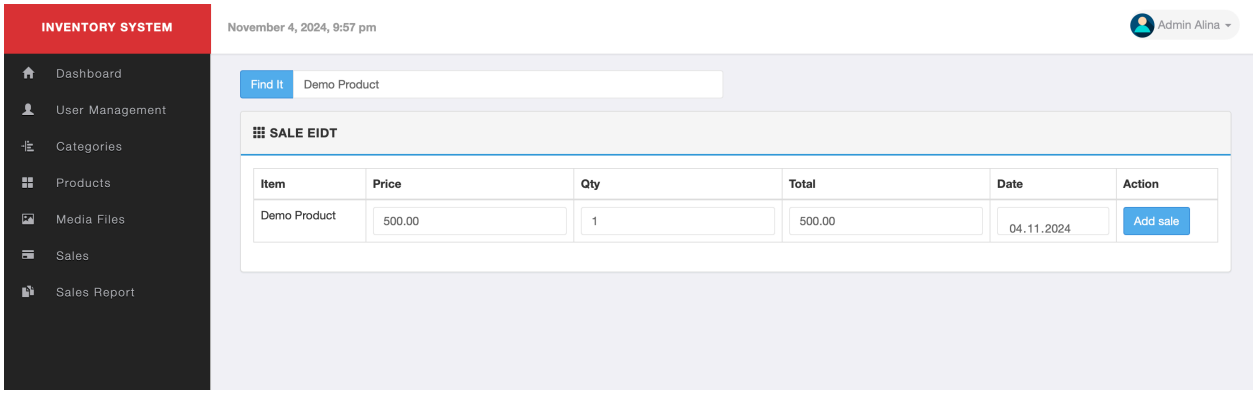


Рисунок 3.29 – Вигляд сторінки «Add Sales»

Сторінка «Sales by Dates» надає користувачам можливість переглянути дані про продажі за обраний діапазон дат, що полегшує аналіз даних за певний період.

На рисунках 3.30 – 3.31 представлено вигляд сторінки «Sales by Dates» та генерація звітності за вказаними критеріями.

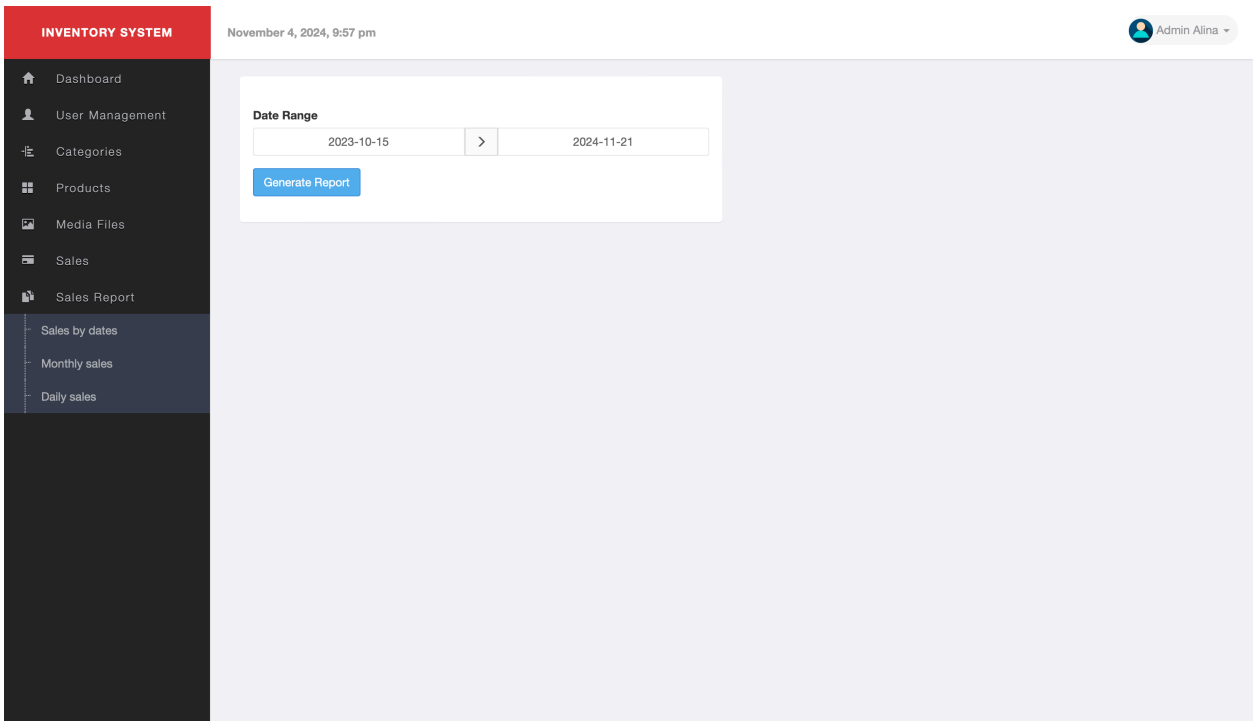


Рисунок 3.30 – Вигляд сторінки «Sales by Dates»

Inventory Management System - Sales Report

2023-10-15 TILL DATE 2024-11-21

Date	Product Title	Buying Price	Selling Price	Total Qty	TOTAL
2024-09-14	Demo Product	100.00	500.00	8	4000.00
2024-04-04	Classic Desktop Tape Dispenser 38	5.00	10.00	5	50.00
2024-04-04	Demo Product	100.00	500.00	2	1000.00
2024-04-04	Disney Woody - Action Figure	29.00	55.00	2	110.00
2024-04-04	Hasbro Marvel Legends Series Toys	219.00	322.00	6	1932.00
2024-04-04	Life Breakfast Cereal-3 Pk	3.00	7.00	5	35.00
2024-04-04	Portable Band Saw XBP02Z	280.00	415.00	2	830.00
2024-04-04	Small Bubble Cushioning Wrap	8.00	19.00	21	399.00
2024-04-04	Wheat	2.00	5.00	3	15.00
				GRAND TOTAL	**$ 8,371.00**
				PROFIT	**$5,225.00**

Рисунок 3.31 – Результат генерації звітності за вказаними критеріями

Сторінка «Monthly Sales» представляє зведені дані про продажі за кожен місяць, що полегшує вивчення щомісячних тенденцій і порівняння результатів за різні періоди часу (рис.3.32).

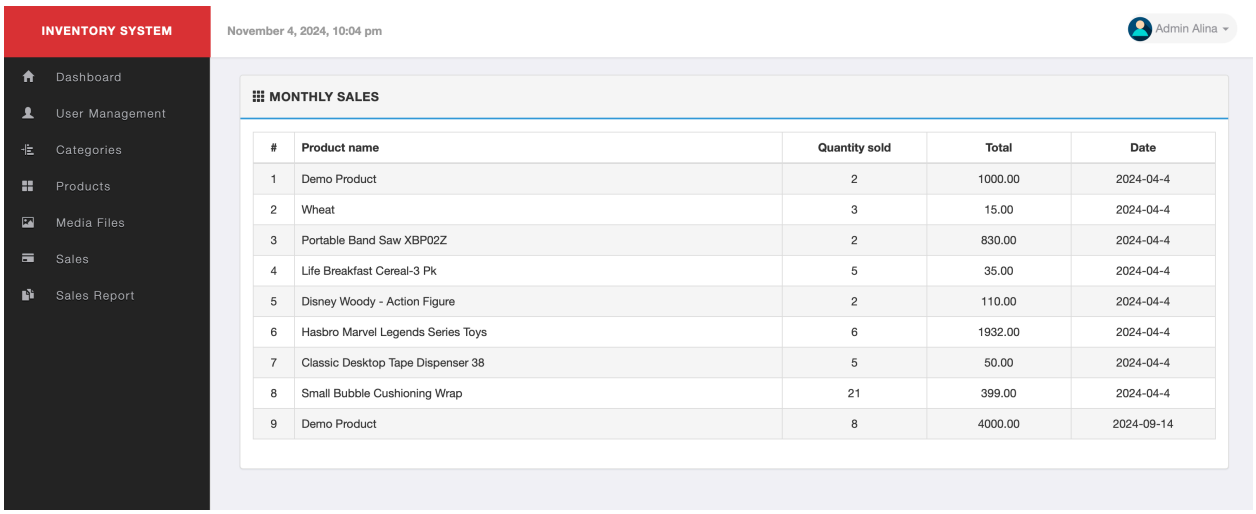


Рисунок 3.32 – Вигляд сторінки «Monthly Sales»

Сторінка «Daily Sales» відображає продажі за кожен день, що дозволяє користувачам більш детально відстежувати щоденну активність і визначати найбільш продуктивні дні для свого бізнесу (рис. 3.33).

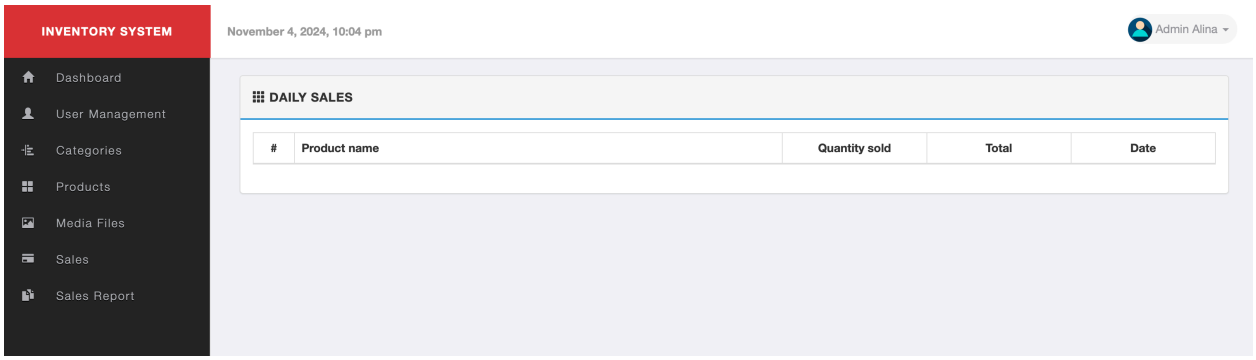


Рисунок 3.33 – Вигляд сторінки ««Daily Sales»»

Також користувач може переглянути власну сторінку та змінити особисті дані. На рисунку 3.34 представлено вигляд особистої сторінки користувача.

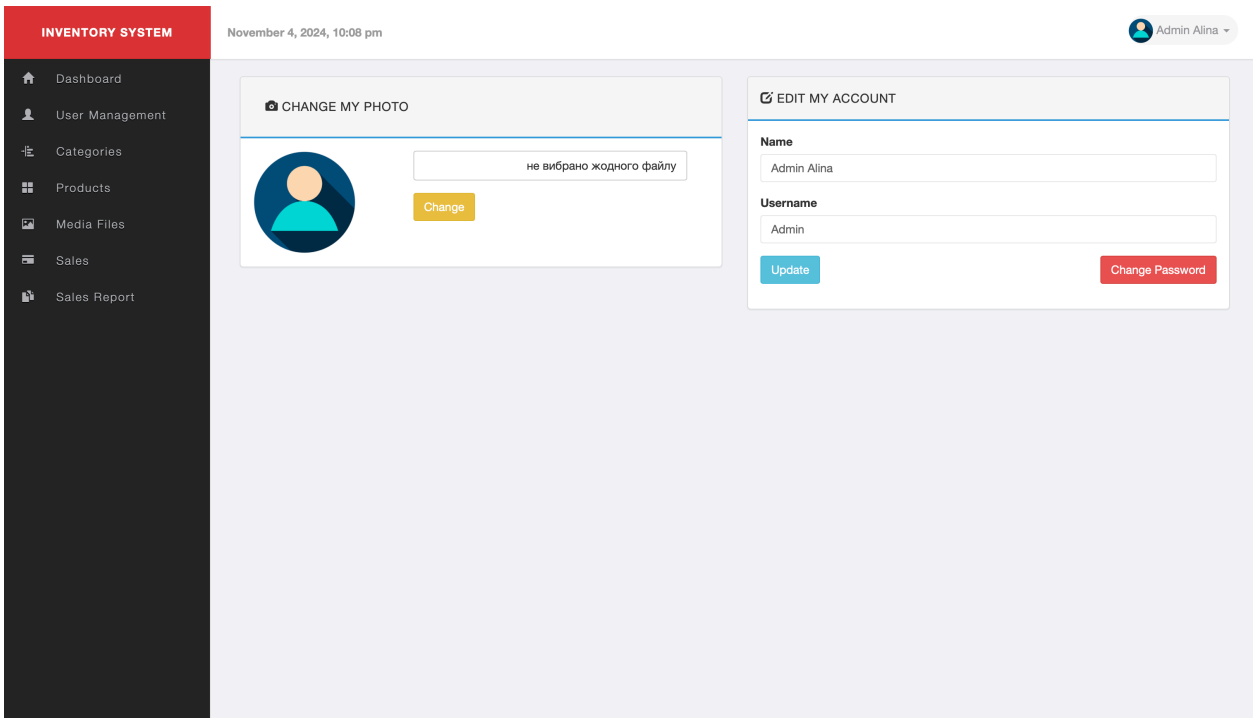


Рисунок 3.34 – Вигляд особистої сторінки користувача

Загалом система включає в себе комплексний набір функцій для ефективного управління товарами, продажами, користувачами та звітністю. Система дозволяє користувачеві виконувати всі основні операції, включаючи додавання, редагування та видалення записів, а також надає аналітичні інструменти для вивчення статистики продажів за різні періоди часу.

Інтегровані модулі адміністрування та зручний інтерфейс полегшують впровадження та подальше використання системи в бізнес-контексті.

ВИСНОВКИ

Дослідження та розробка інформаційної системи управління запасами в CRM для магазинів одягу дозволила застосувати аналітику даних для визначення оптимальних стратегій покращення управління запасами та підвищення операційної ефективності. Аналіз цільової області та порівняння з аналогічними рішеннями підтвердили важливість впровадження сучасних технологій в області управління запасами, що дозволяє більш ефективно контролювати запаси і прогнозувати потреби бізнесу.

Під час практичної фази проєкту були визначені основні функціональні та нефункціональні вимоги до системи, включаючи такі ключові аспекти, як рольова система, управління товарами та продажами, а також генерація звітів. Крім того, було проведено всебічний аналіз існуючих CRM-рішень для управління запасами, що дозволило виявити сильні сторони та визначити шляхи вдосконалення системи, яка зараз розробляється.

Вибір відповідних технологій для реалізації інформаційної системи забезпечив оптимальне поєднання функціональності, продуктивності та безпеки. Розроблена система надає нові можливості для управління запасами, підвищує точність аналітики та посилює конкурентоспроможність бізнесу на сучасному ринку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Top 7 Benefits of CRM Software for Growing Businesses [Електронний ресурс] – Режим доступу до ресурсу: https://waypathconsulting.com/top-benefits-of-customer-relationship-management/ (дата звернення 19.09.2024).

2. Adoption of Artificial Intelligence Integrated Customer Relationship Management in Organizations for Sustainability [Електронний ресурс] – Режим доступу до ресурсу: https://link.springer.com/chapter/10.1007/978-3-030-76583-5_6 (дата звернення 19.09.2024).

3. A Customer Relationship Management Roadmap: What Is Known, Potential Pitfalls, and Where to Go [Електронний ресурс] – Режим доступу до ресурсу:https://www.researchgate.net/publication/228352516_A_Customer_Relationship_Management_Roadmap_What_Is_Known_Potential_Pitfalls_and_Where_to_Go (дата звернення 19.09.2024).

4. Customer relationship management and its impact on entrepreneurial marketing: a literature review [Електронний ресурс] – Режим доступу до ресурсу: https://link.springer.com/article/10.1007/s11365-022-00800-x (дата звернення 19.09.2024).

5. Unlocking Growth and Loyalty: The Benefits of CRM Systems in Business [Електронний ресурс] – Режим доступу до ресурсу: https://www.helpdesk.com/blog/crm-soft/ (дата звернення 19.09.2024).

6. Why Customer Relationship System so Important? [Електронний ресурс] – Режим доступу до ресурсу: https://www.forbes.com/sites/forbesagencycouncil/2017/10/24/why-is-customer-relationship-management-so-important/ (дата звернення 19.09.2024).

7. CRM система AIMS360 [Електронний ресурс] – Режим доступу до ресурсу: https://aims360.com/ (дата звернення 19.09.2024).

8. CRM система ApparelMagic [Електронний ресурс] – Режим доступу до ресурсу: https://apparelmagic.com/ (дата звернення 19.09.2024).

9. CRM система ShowroomHQ [Електронний ресурс] – Режим доступу до ресурсу: https://showroomhq.com/ (дата звернення 19.09.2024).

10. Web Application Architecture: The Basics [Електронний ресурс] – Режим доступу до ресурсу: https://www.intellectsoft.net/blog/вебapplication-architecture/ (дата звернення 19.09.2024).

11. Client-Server Architecture – Types, Examples, Benefits & Challenges [Електронний ресурс] – Режим доступу до ресурсу: https://www.zealousys.com/blog/client-server-architecture/ (дата звернення 19.09.2024).

12. PHP vs Node.js: What`s Best for Your Web App? [Електронний ресурс] – Режим доступу до ресурсу: https://www.zend.com/blog/php-vs-nodejs (дата звернення 19.09.2024).

13. Advantages of Node.js [Електронний ресурс] – Режим доступу до ресурсу: https://www.tatvasoft.com/outsourcing/2021/07/advantages-of-node-js.html (дата звернення 19.09.2024).

14. Pros and Cons of Using PHP in 2023 for Web Development [Електронний ресурс] – Режим доступу до ресурсу: https://www.linkedin.com/pulse/pros-cons-using-php-2023-вебdevelopment-alice-gray/ (дата звернення 19.09.2024).

15. PHP Pros and Cons: Demistifying the Popular Programming Language [Електронний ресурс] – Режим доступу до ресурсу: https://hostadvice.com/blog/вебhosting/php/php-pros-and-cons/ (дата звернення 19.09.2024).

16. IDEF0 Diagram [Електронний ресурс] – Режим доступу до ресурсу: https://help.specinnovations.com/idef0-diagram (дата звернення 19.09.2024).

17. Functional Decomposition [Електронний ресурс] – Режим доступу до ресурсу: https://www.sciencedirect.com/topics/engineering/functional-decomposition (дата звернення 19.09.2024).

18. What is Use Case Diagram [Електронний ресурс] – Режим доступу до ресурсу: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/ (дата звернення 19.09.2024).

19. Why Use MySQL for Database Management? [Електронний ресурс] – Режим доступу до ресурсу: https://www.ropstam.com/why-use-mysql-advantages/ (дата звернення 19.09.2024).

20. MongoDB vs MySQL Differences [Електронний ресурс] – Режим доступу до ресурсу: https://www.mongodb.com/resources/compare/mongodb-mysql (дата звернення 19.09.2024).

21. MySQL: What It Is, How It Works, And What It`s Used For [Електронний ресурс] – Режим доступу до ресурсу: https://www.knowi.com/blog/mysql-what-it-is-how-it-works-and-what-its-used-for/ (дата звернення 19.09.2024).

ДОДАТОК А

Деталізація мети проєкту методом SMART

Термін «S.M.A.R.T.» визначається п'ятьма ключовими аспектами або елементами.

Конкретність - це визначальна характеристика чітко сформульованої мети. Конкретна мета - це мета, яка має чітко визначений бажаний результат.

Вкрай важливо, щоб мета була чітко сформульована для того, щоб усі сторони, які беруть участь у проєкті, були згодні з нею. Важливо визначити мету та дії, які необхідно здійснити для її досягнення.

Вимірюваність – це числові значення, пов'язані з метою. Важливо встановити кількісно вимірювану мету, щоб полегшити моніторинг прогресу. Важливо визначити дані, які будуть використовуватися для оцінки досягнення цілі, і встановити процедуру їх збору.

Реалістичність – важливо, щоб цілі були досяжними, щоб підтримувати ентузіазм щодо їх досягнення. Хоча ставити амбітні цілі корисно, може бути корисно розділити їх на більш керовані, поетапні компоненти.

Якщо мета нездійсненна, може виникнути необхідність спочатку збільшити наявні ресурси, щоб підвищити ймовірність успіху. У такому випадку виділення додаткових ресурсів буде окремим завданням SMART.

Важливо, щоб поставлені цілі були узгоджені. Одним з методів визначення актуальності цілі є визначення ключової вигоди для організації.

Важливо, щоб цілі були прив'язані до часу, з чітко визначеним дедлайном. Саме тому цілі S.M.A.R.T. визначаються з кінцевою датою. Це не означає, що вся робота завершена; скоріше, це дозволяє оцінити успішність зусиль і поставити нові цілі [22].

Результати SMART-методу щодо інформаційної системи управління запасами одягу наведено у таблиці Б.1.

Таблиця Б.1 – Формалізація мети за допомогою методу SMART

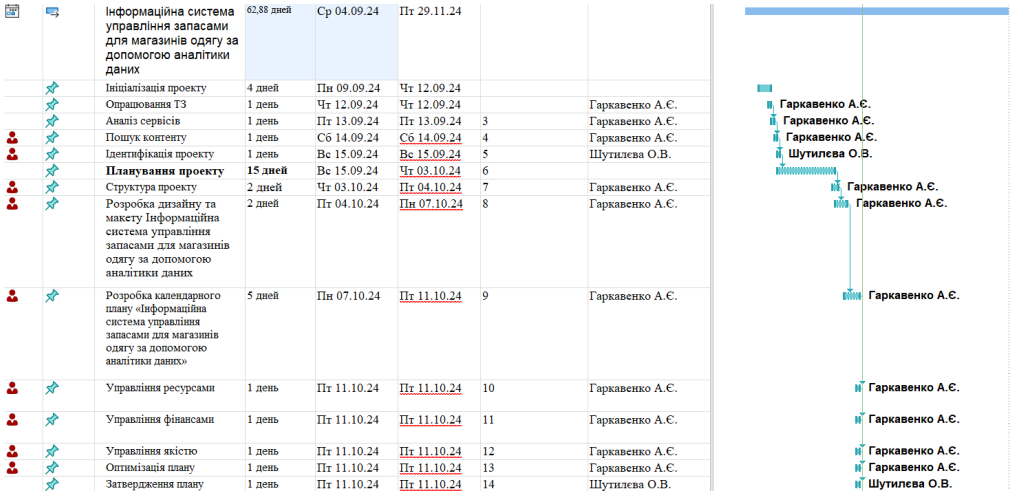
Specific (Конкретна)	Розробка інформаційної системи керування запасами для магазинів одягу, для автоматизації управління товарами, продажами, аналітикою та звітністю в CRM-системі.
Measurable (Вимірювана)	Забезпечення ефективності управління запасами щонайменше на 20% шляхом автоматизації та зниження кількості втрат через нестачу або надлишок товарів.
Achievable (Досяжна)	Мета є досяжною через наявність доступних інструментів для реалізації та аналізу даних які можна впровадити в систему
Relevant (Відповідна)	Система повинна відповідати потребам бізнесу та допомагати магазинам одягу краще контролювати запаси одягу задля підвищення прибутковості та автоматизації керування даними клієнтів та процесами.
Time-bound (Обмежена в часі)	Проєкт має бути завершений до 01.12.2024 та включати в себе етапи планування, розробки та впровадження системи в бізнес процес магазину одягу.

Діаграма Ганта

Діаграми Ганта - це складні інструменти управління проєктами, які полегшують інтуїтивний моніторинг прогресу проєкту та ідентифікацію необхідних завдань.

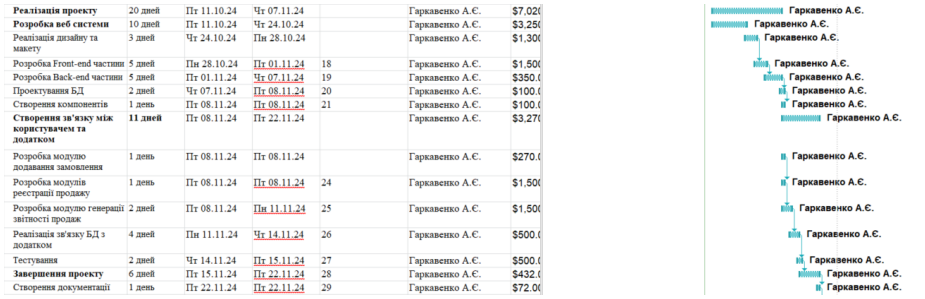
Окрім полегшення декомпозиції проєктів на більш керовані компоненти з визначеними етапами, діаграми Ганта можуть також використовуватися як динамічні інструменти, що адаптуються до мінливих потреб проєкту. Найдосконаліші діаграми Ганта розроблені таким чином, щоб автоматично оновлювати часові рамки проєкту при виконанні попередніх залежностей, тим

самим дозволяючи користувачам зосередитися на виконанні поставленого завдання [25].



		Інформаційна система управління запасами для магазинів одягу за допомогою аналітики даних	62,88 дней	Ср 04.09.24	Пт 29.11.24		
		Ініціалізація проекту	4 дней	Пн 09.09.24	Чт 12.09.24		
		Опрашування ТЗ	1 день	Чт 12.09.24	Чт 12.09.24		Гаркавенко А.Є.
		Аналіз сервісів	1 день	Пт 13.09.24	Пт 13.09.24	3	Гаркавенко А.Є.
		Пошук контенту	1 день	Сб 14.09.24	Сб 14.09.24	4	Гаркавенко А.Є.
		Ідентифікація проекту	1 день	Вс 15.09.24	Вс 15.09.24	5	Шутилєва О.В.
		Планування проекту	**15 дней**	Вс 15.09.24	Чт 03.10.24	6	
		Структура проекту	2 дней	Чт 03.10.24	Пт 04.10.24	7	Гаркавенко А.Є.
		Розробка дизайну та макету Інформаційна система управління запасами для магазинів одягу за допомогою аналітики даних	2 дней	Пт 04.10.24	Пн 07.10.24	8	Гаркавенко А.Є.
		Розробка календарного плану «Інформаційна система управління запасами для магазинів одягу за допомогою аналітики даних»	5 дней	Пн 07.10.24	Пт 11.10.24	9	Гаркавенко А.Є.
		Управління ресурсами	1 день	Пт 11.10.24	Пт 11.10.24	10	Гаркавенко А.Є.
		Управління фінансами	1 день	Пт 11.10.24	Пт 11.10.24	11	Гаркавенко А.Є.
		Управління якістю	1 день	Пт 11.10.24	Пт 11.10.24	12	Гаркавенко А.Є.
		Оптимізація плану	1 день	Пт 11.10.24	Пт 11.10.24	13	Гаркавенко А.Є.
		Затвердження плану	1 день	Пт 11.10.24	Пт 11.10.24	14	Шутилєва О.В.

Рисунок Б.1 – Діаграма Ганта, частина 1



Реалізація проекту	20 дней	Пт 11.10.24	Чт 07.11.24		Гаркавенко А.Є.	$7,02(
Розробка веб системи	10 дней	Пт 11.10.24	Чт 24.10.24		Гаркавенко А.Є.	$3,25(
Реалізація дизайну та макету	3 дней	Чт 24.10.24	Пн 28.10.24		Гаркавенко А.Є.	$1,30(
Розробка Front-end частини	5 дней	Пн 28.10.24	Пт 01.11.24	18	Гаркавенко А.Є.	$1,50(
Розробка Back-end частини	5 дней	Пт 01.11.24	Чт 07.11.24	19	Гаркавенко А.Є.	$350.(
Проектування БД	2 дней	Чт 07.11.24	Пт 08.11.24	20	Гаркавенко А.Є.	$100.(
Створення компонентів	1 день	Пт 08.11.24	Пт 08.11.24	21	Гаркавенко А.Є.	$100.(
Створення зв'язку між користувачем та додатком	**11 дней**	Пт 08.11.24	Пт 22.11.24		Гаркавенко А.Є.	$3,27(
Розробка модулю додавання замовлення	1 день	Пт 08.11.24	Пт 08.11.24		Гаркавенко А.Є.	$270.(
Розробка модулів реєстрації продажу	1 день	Пт 08.11.24	Пт 08.11.24	24	Гаркавенко А.Є.	$1,50(
Розробка модулю генерації звітності продаж	2 дней	Пт 08.11.24	Пн 11.11.24	25	Гаркавенко А.Є.	$1,50(
Реалізація зв'язку БД з додатком	4 дней	Пн 11.11.24	Чт 14.11.24	26	Гаркавенко А.Є.	$500.(
Тестування	2 дней	Чт 14.11.24	Пт 15.11.24	27	Гаркавенко А.Є.	$500.(
Завершення проекту	6 дней	Пт 15.11.24	Пт 22.11.24	28	Гаркавенко А.Є.	$432.(
Створення документації	1 день	Пт 22.11.24	Пт 22.11.24	29	Гаркавенко А.Є.	$72.0(

Рисунок Б.2 – Діаграма Ганта, частина 2

Управління ризиками проєкту

Впровадження стратегій управління проєктними ризиками при розробці інформаційної системи управління запасами одягу пропонує ряд переваг, серед яких:

Запобігання затримкам: Виявлення потенційних проблем дозволяє уникнути затримок у процесі розробки.

Зниження витрат - ще одна перевага управління ризиками. Впровадження стратегій управління ризиками може сприяти зниженню витрат, пов'язаних з виправленням помилок і необхідністю переробки.

Покращення якості: Своєчасне виявлення технічних ризиків сприяє підвищенню якості кінцевого продукту.

Здатність адаптуватися до змін: Проєкт легше піддається модифікаціям у відповідь на нові вимоги або зміни на ринку.

Відповідність вимогам: Впровадження стратегій управління ризиками сприяє забезпеченню відповідності системи необхідним бізнес-потребам [26].

Таблиця А.2. Ймовірність виникнення і величина ризику

№	Ризики	Виникнення	Втрати
1	Недостатнє або надмірне планування	2	4
2	Використання нових або нестабільних технологій, проблеми з інтеграцією	3	3
3	Зміни в бізнес-процесах або вимогах замовника	3	5
4	Погана комунікація між виконавцями та замовником	2	3
5	Відсутність системи захисту даних користувачів	4	5
6	Застарілість та неактуальність інформації	2	3

Таблиця А.3 – Матриця впливу

Вірогідність виникнення	Матриця впливу				
5			3	5	
4		4			
3		6	2		
2				1	
1					
Ступінь впливу	1	2	3	4	5

ДОДАТОК Б

Лістинг коду database.php

```
<?php
require_once(LIB_PATH_INC.DS."config.php");

class MySqli_DB {

    private $con;
    public $query_id;

    function __construct() {
      $this->db_connect();
    }


public function db_connect()
{
  $this->con = mysqli_connect(DB_HOST,DB_USER,DB_PASS);
  if(!$this->con)
         {
           die("    Database    connection    failed:".
mysqli_connect_error());
         } else {
           $select_db = $this->con->select_db(DB_NAME);
             if(!$select_db)
             {
               die("Failed    to    Select    Database".
mysqli_connect_error());
             }
         }
}
```

```
public function db_disconnect()
{
  if(isset($this->con))
  {
    mysqli_close($this->con);
    unset($this->con);
  }
}

public function query($sql)
   {

      if (trim($sql != "")) {
          $this->query_id = $this->con->query($sql);
      }
      if (!$this->query_id)
        // only for Develope mode
              die("Error on this Query :<pre> " . $sql
."</pre>");
       // For production mode
        //  die("Error on Query");

       return $this->query_id;

   }


public function fetch_array($statement)
```

```
{
  return mysqli_fetch_array($statement);
}
public function fetch_object($statement)
{
  return mysqli_fetch_object($statement);
}
public function fetch_assoc($statement)
{
  return mysqli_fetch_assoc($statement);
}
public function num_rows($statement)
{
  return mysqli_num_rows($statement);
}
public function insert_id()
{
  return mysqli_insert_id($this->con);
}
public function affected_rows()
{
  return mysqli_affected_rows($this->con);
}

 public function escape($str){
   return $this->con->real_escape_string($str);
 }

public function while_loop($loop){
 global $db;
```

```
    $results = array();
    while ($result = $this->fetch_array($loop)) {
        $results[] = $result;
    }
 return $results;
}

}

$db = new MySqli_DB();

?>
```

ДОДАТОК В

Лістинг коду functions.php

```
<?php
 $errors = array();


function real_escape($str){
  global $con;
  $escape = mysqli_real_escape_string($con,$str);
  return $escape;
}

function remove_junk($str){
  $str = nl2br($str);
  $str = htmlspecialchars(strip_tags($str,
ENT_QUOTES));
  return $str;
}

function first_character($str){
  $val = str_replace('-'," ",$str);
  $val = ucfirst($val);
  return $val;
}

function validate_fields($var){
  global $errors;
  foreach ($var as $field) {
    $val = remove_junk($_POST[$field]);
    if(isset($val) && $val==''){
      $errors = $field ." can't be blank.";
```

```
      return $errors;
    }
  }
}

function display_msg($msg =''){
   $output = array();
   if(!empty($msg)) {
      foreach ($msg as $key => $value) {
         $output  = "<div class=\"alert alert-
{$key}\">";
         $output .= "<a href=\"#\" class=\"close\"
data-dismiss=\"alert\">&times;</a>";
         $output .=
remove_junk(first_character($value));
         $output .= "</div>";
      }
      return $output;
   } else {
     return "" ;
   }
}

function redirect($url, $permanent = false)
{
    if (headers_sent() === false)
    {
      header('Location: ' . $url, true, ($permanent ===
true) ? 301 : 302);
    }
```

```
    exit();
}

function total_price($totals){
   $sum = 0;
   $sub = 0;
   foreach($totals as $total ){
     $sum += $total['total_saleing_price'];
     $sub += $total['total_buying_price'];
     $profit = $sum - $sub;
   }
   return array($sum,$profit);
}

function read_date($str){
     if($str)
      return date('F j, Y, g:i:s a', strtotime($str));
     else
      return null;
  }

function make_date(){
  return strftime("%Y-%m-%d %H:%M:%S", time());
}

function count_id(){
  static $count = 1;
  return $count++;
}
```

```
function randString($length = 5)
{
  $str='';
  $cha = "0123456789abcdefghijklmnopqrstuvwxyz";

  for($x=0; $x<$length; $x++)
   $str .= $cha[mt_rand(0,strlen($cha))];
  return $str;
}


?>
```

ДОДАТОК Г

Лістинг коду add_product.php

Лістинг коду add_product.php

```
<?php
  $page_title = 'Add Product';
  require_once('includes/load.php');

  page_require_level(2);
  $all_categories = find_all('categories');
  $all_photo = find_all('media');
?>
<?php
 if(isset($_POST['add_product'])){
   $req_fields = array('product-title','product-
categorie','product-quantity','buying-price', 'saleing-
price' );
   validate_fields($req_fields);
   if(empty($errors)){
     $p_name  = remove_junk($db-
>escape($_POST['product-title']));
     $p_cat   = remove_junk($db-
>escape($_POST['product-categorie']));
     $p_qty   = remove_junk($db-
>escape($_POST['product-quantity']));
     $p_buy   = remove_junk($db->escape($_POST['buying-
price']));
     $p_sale  = remove_junk($db-
>escape($_POST['saleing-price']));
     if (is_null($_POST['product-photo']) ||
$_POST['product-photo'] === "") {
```

```
        $media_id = '0';
      } else {
        $media_id = remove_junk($db-
>escape($_POST['product-photo']));
      }
      $date    = make_date();
      $query  = "INSERT INTO products (";
      $query .="
name,quantity,buy_price,sale_price,categorie_id,media_i
d,date";
      $query .=") VALUES (";
      $query .=" '{$p_name}', '{$p_qty}', '{$p_buy}',
'{$p_sale}', '{$p_cat}', '{$media_id}', '{$date}'";
      $query .=")";
      $query .=" ON DUPLICATE KEY UPDATE
name='{$p_name}'";
      if($db->query($query)){
        $session->msg('s',"Product added ");
        redirect('add_product.php', false);
      } else {
        $session->msg('d',' Sorry failed to added!');
        redirect('product.php', false);
      }

    } else{
      $session->msg("d", $errors);
      redirect('add_product.php',false);
    }

 }
```

```
?>
<?php include_once('layouts/header.php'); ?>
<div class="row">
  <div class="col-md-12">
    <?php echo display_msg($msg); ?>
  </div>
</div>
  <div class="row">
  <div class="col-md-8">
      <div class="panel panel-default">
        <div class="panel-heading">
          <strong>
            <span class="glyphicon glyphicon-
th"></span>
            <span>Add New Product</span>
         </strong>
        </div>
        <div class="panel-body">
         <div class="col-md-12">
          <form method="post" action="add_product.php"
class="clearfix">
              <div class="form-group">
                <div class="input-group">
                  <span class="input-group-addon">
                   <i class="glyphicon glyphicon-th-
large"></i>
                  </span>
```

```
                <input type="text" class="form-
control" name="product-title" placeholder="Product
Title">
              </div>
            </div>
            <div class="form-group">
              <div class="row">
                <div class="col-md-6">
                  <select class="form-control"
name="product-categorie">
                    <option value="">Select Product
Category</option>
                  <?php  foreach ($all_categories as
$cat): ?>
                    <option value="<?php echo
(int)$cat['id'] ?>">
                      <?php echo $cat['name']
?></option>
                  <?php endforeach; ?>
                  </select>
                </div>
                <div class="col-md-6">
                  <select class="form-control"
name="product-photo">
                    <option value="">Select Product
Photo</option>
                  <?php  foreach ($all_photo as
$photo): ?>
                    <option value="<?php echo
(int)$photo['id'] ?>">
```

```
                        <?php echo $photo['file_name']
?></option>
                    <?php endforeach; ?>
                    </select>
                  </div>
                </div>
              </div>

              <div class="form-group">
               <div class="row">
                 <div class="col-md-4">
                   <div class="input-group">
                     <span class="input-group-addon">
                      <i class="glyphicon glyphicon-
shopping-cart"></i>
                     </span>
                     <input type="number" class="form-
control" name="product-quantity" placeholder="Product
Quantity">
                  </div>
                 </div>
                 <div class="col-md-4">
                   <div class="input-group">
                     <span class="input-group-addon">
                       <i class="glyphicon glyphicon-
usd"></i>
                     </span>
                     <input type="number" class="form-
control" name="buying-price" placeholder="Buying
Price">
```

```
                    <span class="input-group-
addon">.00</span>
                  </div>
                 </div>
                  <div class="col-md-4">
                    <div class="input-group">
                      <span class="input-group-addon">
                        <i class="glyphicon glyphicon-
usd"></i>
                      </span>
                      <input type="number" class="form-
control" name="saleing-price" placeholder="Selling
Price">
                      <span class="input-group-
addon">.00</span>
                    </div>
                   </div>
                 </div>
                <button type="submit" name="add_product"
class="btn btn-danger">Add product</button>
           </form>
          </div>
         </div>
       </div>
     </div>
  </div>

<?php include_once('layouts/footer.php'); ?>
```

Лістинг коду delete_product.php

```
<?php
  require_once('includes/load.php');

  page_require_level(2);
?>
<?php
  $product = find_by_id('products',(int)$_GET['id']);
  if(!$product){
    $session->msg("d","Missing Product id.");
    redirect('product.php');
  }
?>
<?php
  $delete_id =
delete_by_id('products',(int)$product['id']);
  if($delete_id){
      $session->msg("s","Products deleted.");
      redirect('product.php');
  } else {
      $session->msg("d","Products deletion failed.");
      redirect('product.php');
  }
?>
```

Лістинг коду edit_product.php

```
<?php
  $page_title = 'Edit product';
  require_once('includes/load.php');
   page_require_level(2);
?>
```

```php
<?php
$product = find_by_id('products',(int)$_GET['id']);
$all_categories = find_all('categories');
$all_photo = find_all('media');
if(!$product){
  $session->msg("d","Missing product id.");
  redirect('product.php');
}
?>
<?php
 if(isset($_POST['product'])){
    $req_fields = array('product-title','product-
categorie','product-quantity','buying-price', 'saleing-
price' );
    validate_fields($req_fields);

   if(empty($errors)){
       $p_name  = remove_junk($db-
>escape($_POST['product-title']));
       $p_cat   = (int)$_POST['product-categorie'];
       $p_qty   = remove_junk($db-
>escape($_POST['product-quantity']));
       $p_buy   = remove_junk($db-
>escape($_POST['buying-price']));
       $p_sale  = remove_junk($db-
>escape($_POST['saleing-price']));
       if (is_null($_POST['product-photo']) ||
$_POST['product-photo'] === "") {
         $media_id = '0';
       } else {
```

```
          $media_id = remove_junk($db-
>escape($_POST['product-photo']));
        }
        $query   = "UPDATE products SET";
        $query  .=" name ='{$p_name}', quantity
='{$p_qty}',";
        $query  .=" buy_price ='{$p_buy}', sale_price
='{$p_sale}', categorie_id
='{$p_cat}',media_id='{$media_id}'";
        $query  .=" WHERE id ='{$product['id']}'";
        $result = $db->query($query);
                if($result && $db->affected_rows() ===
1){
                  $session->msg('s',"Product updated ");
                  redirect('product.php', false);
                } else {
                  $session->msg('d',' Sorry failed to
updated!');

redirect('edit_product.php?id='.$product['id'], false);
                }

   } else{
        $session->msg("d", $errors);
        redirect('edit_product.php?id='.$product['id'],
false);
   }

 }
```

```
?>
<?php include_once('layouts/header.php'); ?>
<div class="row">
  <div class="col-md-12">
    <?php echo display_msg($msg); ?>
  </div>
</div>
  <div class="row">
      <div class="panel panel-default">
        <div class="panel-heading">
          <strong>
            <span class="glyphicon glyphicon-
th"></span>
            <span>Add New Product</span>
         </strong>
        </div>
        <div class="panel-body">
         <div class="col-md-7">
           <form method="post"
action="edit_product.php?id=<?php echo
(int)$product['id'] ?>">
              <div class="form-group">
                <div class="input-group">
                  <span class="input-group-addon">
                   <i class="glyphicon glyphicon-th-
large"></i>
                  </span>
                  <input type="text" class="form-
control" name="product-title" value="<?php echo
remove_junk($product['name']);?>">
```

```
                </div>
              </div>
              <div class="form-group">
                <div class="row">
                  <div class="col-md-6">
                    <select class="form-control"
name="product-categorie">
                    <option value=""> Select a
categorie</option>
                   <?php  foreach ($all_categories as
$cat): ?>
                     <option value="<?php echo
(int)$cat['id']; ?>" <?php if($product['categorie_id']
=== $cat['id']): echo "selected"; endif; ?> >
                      <?php echo
remove_junk($cat['name']); ?></option>
                   <?php endforeach; ?>
                  </select>
                  </div>
                  <div class="col-md-6">
                    <select class="form-control"
name="product-photo">
                      <option value=""> No
image</option>
                      <?php  foreach ($all_photo as
$photo): ?>
                        <option value="<?php echo
(int)$photo['id'];?>" <?php if($product['media_id'] ===
$photo['id']): echo "selected"; endif; ?> >
```

```
                        <?php echo
$photo['file_name'] ?></option>
                      <?php endforeach; ?>
                    </select>
                  </div>
                </div>
              </div>

              <div class="form-group">
               <div class="row">
                 <div class="col-md-4">
                  <div class="form-group">
                    <label for="qty">Quantity</label>
                    <div class="input-group">
                      <span class="input-group-addon">
                       <i class="glyphicon glyphicon-
shopping-cart"></i>
                      </span>
                      <input type="number" class="form-
control" name="product-quantity" value="<?php echo
remove_junk($product['quantity']); ?>">
                   </div>
                  </div>
                 </div>
                 <div class="col-md-4">
                  <div class="form-group">
                    <label for="qty">Buying
price</label>
                    <div class="input-group">
                      <span class="input-group-addon">
```

```
                        <i class="glyphicon glyphicon-
usd"></i>
                      </span>
                      <input type="number" class="form-
control" name="buying-price" value="<?php echo
remove_junk($product['buy_price']);?>">
                      <span class="input-group-
addon">.00</span>
                    </div>
                  </div>
                </div>
                <div class="col-md-4">
                  <div class="form-group">
                    <label for="qty">Selling
price</label>
                    <div class="input-group">
                      <span class="input-group-addon">
                        <i class="glyphicon glyphicon-
usd"></i>
                      </span>
                      <input type="number"
class="form-control" name="saleing-price" value="<?php
echo remove_junk($product['sale_price']);?>">
                      <span class="input-group-
addon">.00</span>
                    </div>
                  </div>
                </div>
              </div>
            </div>
```

```
                <button type="submit" name="product"
class="btn btn-danger">Update</button>
           </form>
          </div>
         </div>
       </div>
  </div>

<?php include_once('layouts/footer.php'); ?>
```