МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ПРИРОДОКОРИСТУВАННЯ
ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ
ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

другого (магістерського) рівня вищої освіти

на тему:

«ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ ЗБОРУ, ОБРОБКИ ТА ЗАХИСТУ ПЕРСОНАЛЬНИХ ДАНИХ КОРИСТУВАЧІВ ВЕБ-САЙТІВ»

Виконав: студент 6 курсу
спеціальності 126 «Інформаційні
системи та технології»

Станасюк О.В.
(прізвище та ініціали)

Керівник:
Желєзняк А.М.
(прізвище та ініціали)

ДУБЛЯНИ 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ПРИРОДОКОРИСТУВАННЯ
ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Рівень вищої освіти другий (магістерський)
Спеціальність 126 «Інформаційні системи та технології»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____
(підпис)
д.т.н., професор, Тригуба А. М.
(вч. звання, прізвище, ініціали)
“____” _____2024 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Станасюка Олега Віталійовича
(прізвище, ім’я, по батькові)

1. Тема роботи *«Дослідження технологій збору, обробки та захисту персональних даних користувачів веб-сайтів»*
керівник роботи *к. е н., доцент., Желєзняк А.М.*
(наук.ступінь, вч. звання, прізвище, ініціали)
затверджені наказом Львівського НУП *№616 к/с від 12.09.2024 р*
2. Строк подання студентом роботи 2.12.2024 р.
3. Вихідні дані: *аналітичні дані роботи та характеристика об’єкту дослідження, опис бібліотек мов програмування, науково-технічна і довідкова література.*
4. Зміст кваліфікаційної роботи (перелік питань, які потрібно розробити)
Вступ
1. Аналіз стану питання в теорії та практиці та постановка завдання
2. Обґрунтування, вибір та реалізація інструментарію вирішення задачі
3. Результати вирішення задачі
4. Охорона праці та безпека в надзвичайних ситуаціях
5. Визначення ефективності
Висновки
Бібліографічний список
5. Перелік графічного матеріалу
Графічний матеріал подається у вигляді презентації

6. Консультанти розділів

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата		Відмітка про виконання
		завдання видав	завдання прийняв	
1, 2, 3, 5	*Желєзняк А.М., доцент кафедри інформаційних технологій*			
4	*Городецький І.М., доцент кафедри фізики, інженерної механіки та безпеки виробництва*			

7. Дата видачі завдання *1.03.2024*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Відмітка про виконання
1	*Отримання завдання. Вивчення рекомендованої літератури по темі роботи. Написання першого розділу*	*01.03.2024 – 31.05.2024*	
2	*Проектування та опис технічного завдання, обґрунтування та вибір інструментарію реалізації проекту (написання другого розділу).*	*01.06.2024 – 31.08.2024*	
3	*Програмна реалізація поставленого завдання (написання третього розділу)*	*01.09.2024 – 18.10.2024*	
4	*Написання розділу «Охорона праці та безпека у надзвичайних ситуаціях»*	*19.10.2024 – 04.11.2024*	
5	*Оцінка ефективності поставленого завдання (виконання п'ятого розділу)*	*05.11.2024 – 18.11.2024*	
6	*Завершення оформлення основної частини, написання висновків та підготовка презентаційного матеріалу*	*19.11.2024 – 02.12.2024*	
7	*Завершення роботи в цілому. Підготовка до захисту кваліфікаційної роботи*	*03.12.2024 – 16.12.2024*	

Студент _____ *Станасюк О.В.*
(підпис) (прізвище та ініціали)

Керівник роботи _____ *Желєзняк А.М.*
(підпис) (прізвище та ініціали)

УДК 004.738.1

Дослідження технологій збору, обробки та захисту персональних даних користувачів веб-сайтів.
Станасюк О.В. Кафедра інформаційних технологій - Дубляни, Львівський НУП, 2024.
Кваліфікаційна робота: 60 с. текст. част., 18 рис., 1 табл., 20 джерел

Наведено теоретичні основи збору, обробки та захисту персональних даних користувачів веб-сайтів, включаючи правові, технічні та організаційні аспекти. Проведено огляд прикладних рішень, таких як шифрування, багатофакторна автентифікація та персоналізація контенту.

Обґрунтовано перспективи використання сучасних технологій захисту для вдосконалення процесів збору даних та підвищення якості обслуговування.

Запропоновано інтеграцію механізмів збору згоди користувачів та автоматизованого моніторингу ризиків.

Здійснено аналіз травматичних ситуацій при роботі з комп’ютерною технікою, викладено питання охорони праці.

Здійснено обґрунтування та вибір технологій збору даних, що відповідають законодавству. Проаналізовано перспективи використання машинного навчання для персоналізації сервісів.

Визначено показники ефективності заходів щодо підвищення безпеки даних. Проведено аналіз охорони праці та ризиків травматизму під час роботи з комп’ютерною технікою.

Ключові слова: технологія, сайт, захист, персональні дані

ЗМІСТ

ВСТУП

У сучасному цифровому світі персональні дані користувачів веб-сайтів є цінним ресурсом, що використовується для персоналізації контенту, маркетингу та вдосконалення послуг. Однак зростання обсягів зібраної інформації супроводжується серйозними викликами, зокрема ризиками витоків, несанкціонованого використання та порушення конфіденційності. Ці проблеми підривають довіру до онлайн-сервісів, що робить захист персональних даних однією з ключових задач сучасного суспільства.

Держави та міжнародні організації реагують на ці виклики шляхом прийняття регуляторних актів, таких як GDPR в Європейському Союзі, які зобов’язують компанії забезпечувати належний захист даних користувачів. Паралельно з цим, розвиток технологій, таких як блокчейн, штучний інтелект і машинне навчання, пропонує нові можливості для обробки великих масивів інформації, але також створює нові загрози конфіденційності.

Дослідження цієї теми є актуальним для забезпечення балансу між технологічним прогресом і захистом прав користувачів. Розробка ефективних рішень для збору, обробки та захисту персональних даних є основою формування безпечного і довіреного інформаційного середовища.

Метою кваліфікаційної роботи є дослідження сучасних технологій збору, обробки та захисту персональних даних користувачів веб-сайтів, розробка практичних рекомендацій щодо їхньої реалізації та забезпечення відповідності вимогам чинного законодавства й міжнародних стандартів.

Тема є актуальною, оскільки в умовах стрімкого розвитку цифрових технологій і зростання обсягів персональних даних, що обробляються, питання їхньої безпеки стають критично важливими. Неправомірне використання даних може призвести до фінансових втрат, порушення конфіденційності та втрати довіри користувачів. З урахуванням вимог міжнародного законодавства, зокрема GDPR, та актуальних українських

норм, дослідження ефективних підходів до збору, обробки та захисту даних має значний науковий і практичний інтерес.

В ході виконання кваліфікаційної роботи були поставлені наступні завдання:

1. Дослідити теоретичні та методологічні аспекти, основні тенденції та підходи у сфері збору, обробки та захисту персональних даних користувачів веб-сайтів
2. Визначити способи захисту персональних даних.
3. Оцінити нормативно-правові аспекти.

Загалом дипломна робота складається з п’яти розділів. В першому розділі розкриті теоретичні основи збору, обробки та захисту персональних даних. У другому розділі детально розібрані варіанти збору, обробки та захисту персональних даних. Третій розділ присвячено проектуванню та розробці системи, що забезпечує ефективний збір, обробку та захист персональних даних користувачів веб-сайтів. В п’ятому розділі описана актуальність даної теми

Наукова новизна кваліфікаційної роботи полягає в дослідженні, обґрунтуванні та виборі моделі для ефективного збору, обробки та захисту персональних даних користувачів веб-сайтів.

Під час практичної підготовки та виконання кваліфікаційної роботи результати були апробовані на Міжнародній студентській науковій конференції «Студентська молодь і науковий прогрес в АПК», матеріали якої були опубліковані у вигляді тез:

1. Станасюк О. Дослідження технологій збору, обробки та захисту персональних даних користувачів вебсайтів. *Тези доповідей Міжнародного студентського наукового форуму «Студентська молодь і науковий прогрес»* 4-6 жовтня 2024 р., м.Львів / Львів.нац. ун-т природ. Львів, 2024.С.368.

РОЗДІЛ 1.
АНАЛІЗ СТАНУ ПИТАННЯ В ТЕОРІЇ ТА ПРАКТИЦІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Теоретичні основи збору, обробки та захисту персональних даних користувачів веб-сайтів

Функціонування веб-сайтів у сучасному світі неможливе без ефективного збору, обробки та захисту персональних даних користувачів. Ці процеси є ключовими для налаштування персоналізованого досвіду, оптимізації роботи сервісів та створення зручного і безпечного онлайн-середовища. Водночас, значні ризики, пов’язані з витоками інформації або її несанкціонованим використанням, обумовлюють необхідність розробки технологій, які дозволяють забезпечити захист приватності. Теоретичний аналіз цих аспектів дозволяє зрозуміти основні підходи до обробки інформації та визначити шляхи подолання викликів, пов’язаних із конфіденційністю.

Процес збору персональних даних охоплює різноманітні методи, які залежать від цілей веб-сайту. Інформація може збиратися через форми реєстрації, контактні запити або інтерактивні інструменти, які дозволяють користувачам залишати свої дані. Найпоширенішими способами є використання форм, що запитують персональну інформацію, та застосування файлів cookie, які зберігають інформацію про активність користувача, включаючи налаштування сайту, історію переглядів або дії на сторінці.

На рисунку 1.1 зображено, як веб-сайт збирає дані користувача через форми введення, файли cookie або трекери, а потім передає їх на сервер для зберігання та подальшої обробки. Важливою складовою є використання аналітичних інструментів, таких як Google Analytics. Вони дозволяють збирати дані про поведінку користувачів, аналізувати популярність різних розділів сайту та виявляти ключові точки взаємодії аудиторії. Ці дані

структуруються та обробляються за допомогою алгоритмів, які сортують інформацію за демографічними, поведінковими та іншими характеристиками.

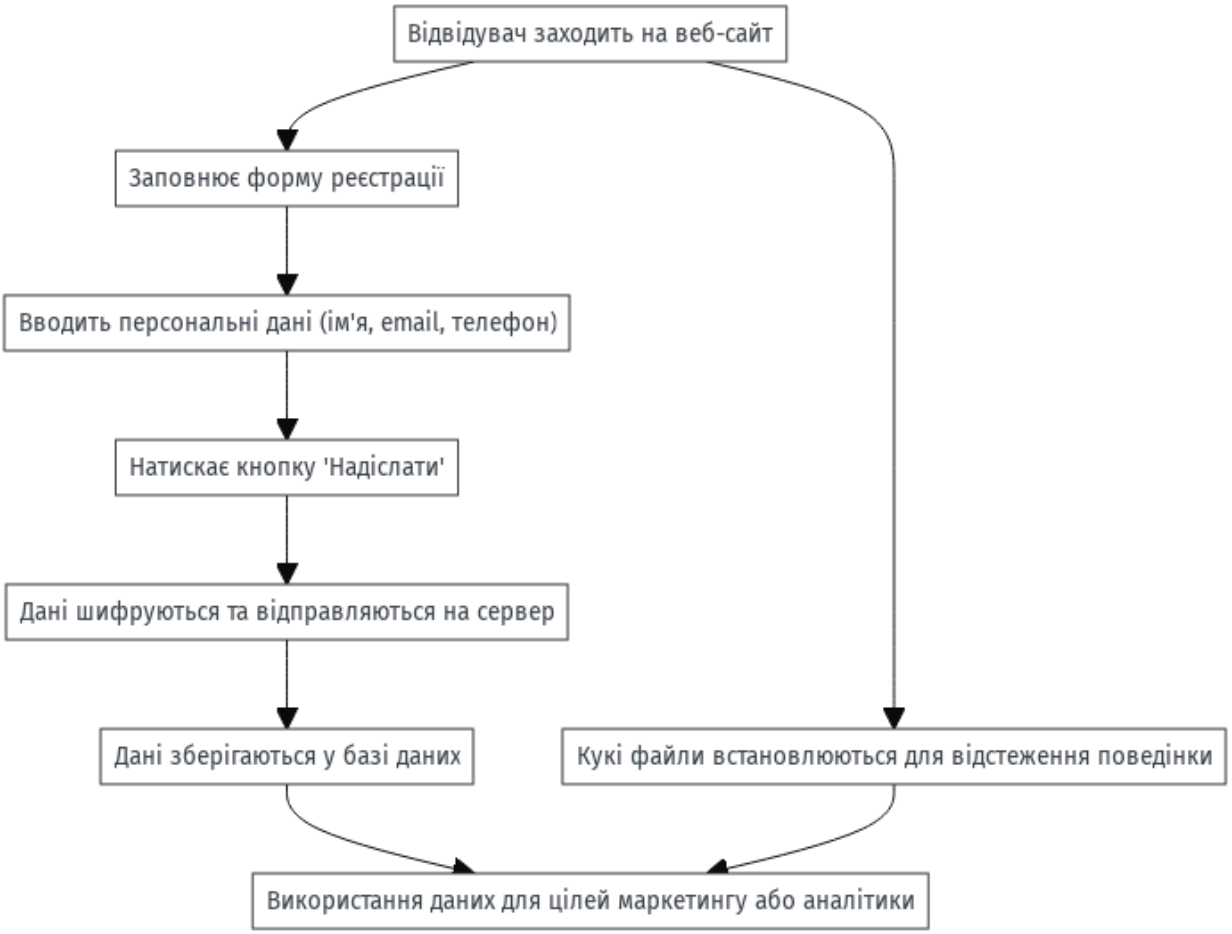


Рисунок 1.1 – Схема збору даних через веб-сайт

Розвиток технологій машинного навчання та штучного інтелекту дозволяє автоматизувати обробку великих масивів даних, що надає компаніям можливість швидше і точніше визначати уподобання користувачів. Побудова так званих «цифрових профілів» є основою для створення персоналізованих рекомендацій, що стають невід’ємною частиною сучасних онлайн-сервісів.

Одним із головних завдань при роботі з персональними даними є гарантування їх безпеки. Це досягається за рахунок застосування різних технологій. Одним із найефективніших є шифрування даних, що дозволяє забезпечити їх безпечну передачу та зберігання. Використання

криптографічних алгоритмів перетворює інформацію на захищений формат, доступ до якого можливий лише для авторизованих сторін.

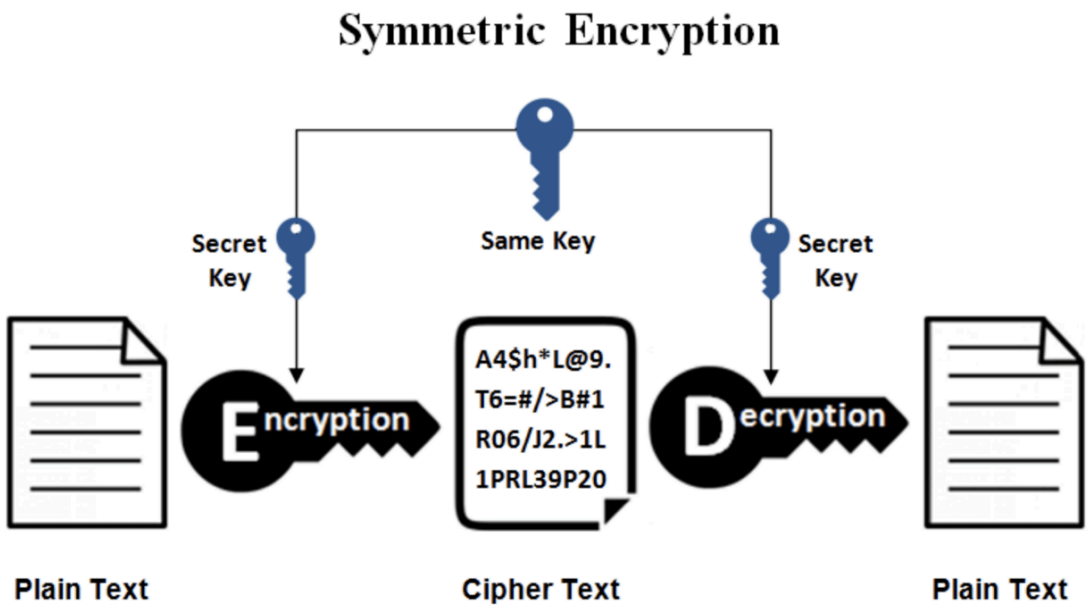


Рисунок 1.2 – Схема шифрування даних

На рисунку 1.2 зображено процес шифрування та дешифрування інформації між користувачем і сервером із використанням ключів на прикладі симетричного шифрування. Паралельно застосовується анонімізація та псевдонімізація, коли персональні дані користувачів перетворюються на сукупності, що не дозволяють ідентифікувати конкретну особу. Це важливо в контексті дотримання міжнародних стандартів конфіденційності, таких як GDPR, що вимагає мінімізації ризиків порушення приватності.

Діаграма на рисунку 1.3 ілюструє взаємозв’язок між основними джерелами ризиків витоку персональних даних, зокрема зовнішніми та внутрішніми загрозами, а також технічними факторами, що виникають на перетині цих категорій. Вона наочно демонструє, як зовнішні ризики, такі як атаки хакерів, фішинг або несанкціонований доступ, взаємодіють із внутрішніми проблемами, зокрема людськими помилками, недостатньою підготовкою персоналу чи недоліками в управлінні. Технічні ризики

виступають сполучною ланкою між цими двома сферами, оскільки вони можуть бути спричинені як зовнішніми атаками, так і внутрішніми вадами системи або порушеннями її конфігурації.

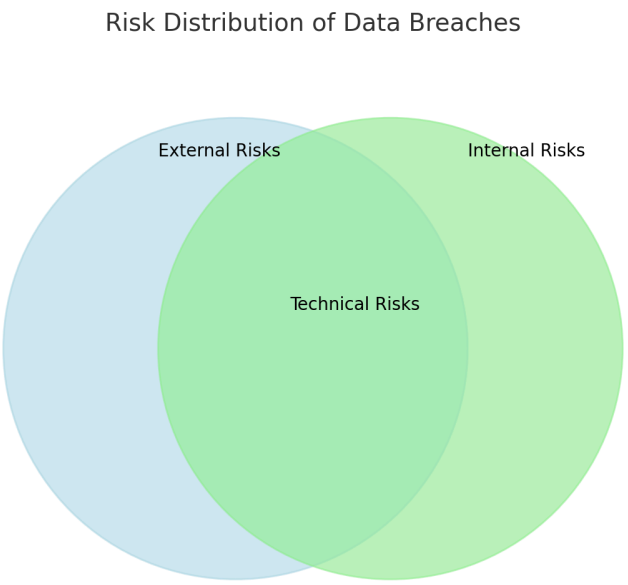


Рисунок 1.3 – Діаграма розподілу ризиків витоку даних

Такий підхід дозволяє отримати цілісне уявлення про природу загроз і дає змогу визначити слабкі місця в системі захисту даних. Він підкреслює важливість комплексного підходу до управління ризиками, який враховує їхній взаємний вплив, а також потребу у впровадженні інтегрованих рішень для підвищення загальної безпеки системи.

Системний підхід до збору, обробки та захисту персональних даних є критично важливим у сучасних умовах розвитку цифрових технологій. З одного боку, це дозволяє підвищити ефективність роботи веб-сайтів і задовольнити зростаючі очікування користувачів щодо персоналізованих послуг. З іншого боку, лише забезпечення належного рівня конфіденційності та безпеки дозволяє підтримувати довіру користувачів і відповідати вимогам

законодавства. Теоретичний аналіз технологій збору та захисту даних дає змогу визначити оптимальні шляхи розвитку сучасних систем і створити основу для їх безпечного використання.

1.2 Аналіз предметної області у сфері обробки персональних даних

Персональні дані є одним із найцінніших активів у сучасному цифровому середовищі. Вони слугують основою для багатьох процесів, включаючи персоналізацію послуг, аналіз поведінки користувачів, маркетинг і підтримку клієнтів. Збір, обробка та захист цих даних стали ключовими питаннями, що впливають не лише на ефективність бізнесу, а й на права користувачів та довіру до цифрових платформ. У цьому підрозділі розглядаються основні аспекти предметної області та нормативно-правового регулювання, які визначають основи обробки персональних даних.

Предметна область охоплює широкий спектр технічних, організаційних і правових питань, пов’язаних із управлінням персональною інформацією. У технічному аспекті йдеться про методи збору даних через форми, трекери та файли cookie, обробку інформації за допомогою сучасних аналітичних платформ, а також управління її зберіганням у базах даних. Ці процеси забезпечують функціонування багатьох цифрових послуг, таких як електронна комерція, соціальні мережі та онлайн-банкінг.

Важливим компонентом предметної області є забезпечення безпеки персональних даних. Зокрема, сучасні технології обробки даних вимагають використання засобів шифрування, анонімізації та контролю доступу. Наприклад, шифрування гарантує, що дані залишаються конфіденційними навіть у разі витоку, тоді як анонімізація дозволяє зберігати статистичну цінність даних, не розкриваючи конкретних осіб.

Окрему роль відіграє нормативно-правове регулювання, яке формує рамки для обробки та захисту даних. У Європейському Союзі основним стандартом є General Data Protection Regulation (GDPR), що встановлює правила збору, зберігання, обробки та передачі персональної інформації.

GDPR висуває суворі вимоги до компаній, які працюють із даними, включаючи необхідність отримання явної згоди користувачів, повідомлення про обробку даних і можливість видалення даних за запитом.

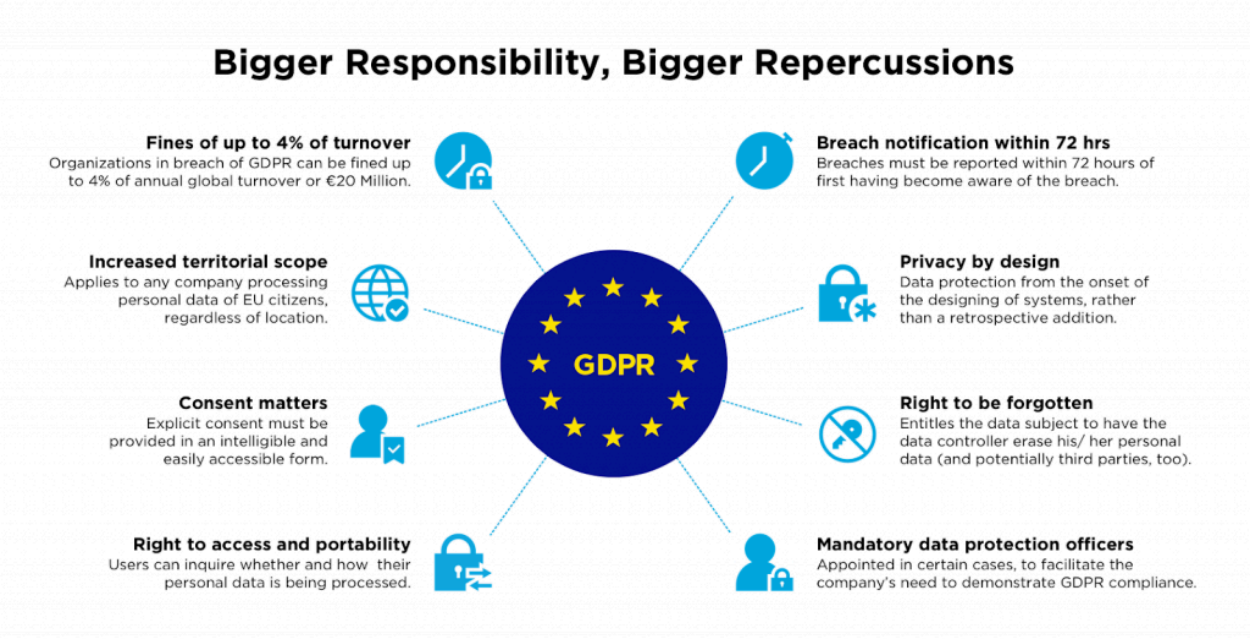


Рисунок 1.4 – Схема визначення GDPR

Схема на рисунку 1.4 демонструє основні аспекти Загального регламенту захисту даних (GDPR) та підкреслює, наскільки серйозними можуть бути наслідки недотримання його вимог. Одним із ключових положень є можливість накладення штрафів, які можуть сягати 4% річного світового обороту компанії або до 20 мільйонів євро, залежно від того, яка сума є більшою. Регламент також розширює свою дію на всі компанії, що обробляють дані громадян Європейського Союзу, незалежно від їхнього місця розташування, що значно підвищує глобальний масштаб відповідальності.

Одним із важливих положень є обов'язок компаній повідомляти про витоки персональних даних протягом 72 годин після їх виявлення, що вимагає високого рівня організації та швидкого реагування. У GDPR особливий акцент зроблено на принципі «приватність за замовчуванням»,

коли механізми захисту даних інтегруються на етапі розробки системи. Це запобігає додаванню таких механізмів уже після виникнення ризиків.

Також регламент захищає права користувачів на контроль своїх даних. Наприклад, користувачі можуть вимагати видалення своїх персональних даних (так зване «право бути забутим») або запитувати доступ до даних і їх перенесення до іншого постачальника послуг. Крім того, GDPR висуває суворі вимоги до процесу отримання згоди на обробку даних – вона повинна бути чіткою, зрозумілою і добровільною.

Для забезпечення дотримання цих вимог компанії в певних випадках зобов’язані призначати співробітників із захисту даних, які будуть відповідати за моніторинг відповідності GDPR. Усе це підкреслює важливість побудови систем, що не лише забезпечують збереження конфіденційності даних, а й адаптуються до сучасних викликів кібербезпеки.

Українське законодавство у сфері захисту персональних даних суттєво відстає від вимог та стандартів, закріплених у General Data Protection Regulation (GDPR). Основною проблемою є те, що вітчизняний Закон "Про захист персональних даних" має значно менш суворі вимоги щодо обробки даних і механізмів захисту. Зокрема, штрафи за порушення є мінімальними і не створюють реального стимулу для дотримання норм. У той час, як GDPR передбачає значні фінансові санкції (до 4% від глобального річного доходу компанії), українське законодавство не має подібної мотивації.

Крім того, український закон не забезпечує такої ж високої прозорості в обробці даних, як це робить GDPR. Наприклад, вимоги до надання згоди на обробку даних залишаються нечіткими, а права суб’єктів персональних даних, такі як доступ до інформації чи "право бути забутим", реалізуються на практиці значно складніше. Також відсутні положення про обов’язкове повідомлення про витоки даних протягом визначеного часу, що є ключовим аспектом захисту у GDPR.

Ще однією слабкою стороною є обмежена територіальна дія українського закону, який не поширюється на обробку даних українських громадян за кордоном. У той же час GDPR має глобальну дію, поширюючись на всі компанії, що працюють з персональними даними громадян ЄС, незалежно від їхнього географічного розташування.

У підсумку, українське законодавство потребує масштабної адаптації до європейських стандартів, щоб відповідати сучасним викликам у сфері приватності та безпеки даних. Інтеграція основних принципів GDPR у національну правову базу є важливим кроком на шляху до забезпечення належного рівня захисту даних.

Особливої уваги заслуговують сучасні виклики у сфері захисту персональних даних. Зростання обсягів даних і кількості цифрових платформ призводить до збільшення ризиків несанкціонованого доступу, витоків або зловживань інформацією. Часто це пов’язано з людськими факторами, технічними збоями чи недостатньою безпекою програмного забезпечення. Щоб мінімізувати ці ризики, організації повинні впроваджувати багаторівневі системи захисту, проводити регулярні аудити безпеки та навчати персонал основам роботи з даними.

Предметна область також включає аналіз сучасних тенденцій у розвитку технологій обробки та захисту даних. Наприклад, використання штучного інтелекту для аналізу великих обсягів інформації та автоматизації процесів збору даних дає можливість підвищити ефективність, але водночас потребує нових підходів до забезпечення безпеки. Зростає важливість впровадження блокчейн-технологій для захисту конфіденційної інформації та забезпечення прозорості обробки даних.

Таким чином, аналіз предметної області демонструє, що обробка персональних даних є багатогранним процесом, який потребує інтеграції сучасних технологій із дотриманням правових стандартів. Це дозволяє забезпечити не лише функціональність цифрових платформ, а й довіру

користувачів, які дедалі більше усвідомлюють значення приватності у цифрову епоху.

1.3 Приклади реалізації збору, обробки та зберігання персональних даних

У сучасному світі персональні дані стали ключовим ресурсом для бізнесу, урядових установ і дослідницьких організацій. Розглянемо приклади практичної реалізації збору, обробки та зберігання персональних даних у різних сферах діяльності.

Одним із найбільш поширених способів збору персональних даних є використання інтерактивних форм на веб-сайтах. Вони можуть бути розміщені на різних сторінках: реєстрації, входу до облікового запису, форми зворотного зв’язку або підписки на розсилку. Наприклад, при створенні облікового запису на платформі електронної комерції користувач заповнює поля з іменем, адресою електронної пошти, номером телефону, адресою доставки і, в деяких випадках, платіжною інформацією.

Цей підхід не лише дозволяє зібрати базові персональні дані, але й надає можливість компанії зрозуміти потреби користувача, щоб пропонувати йому персоналізовані товари або послуги. Важливим елементом таких форм є включення тексту або посилання на політику конфіденційності. Наприклад, більшість сайтів електронної комерції зобов’язують користувача погодитися з умовами обробки даних перед завершенням реєстрації

На рисунку 1.5 відображено класичний приклад форми з обов’язковими полями для імені, адреси електронної пошти та пароля, а також дата народження і стать.

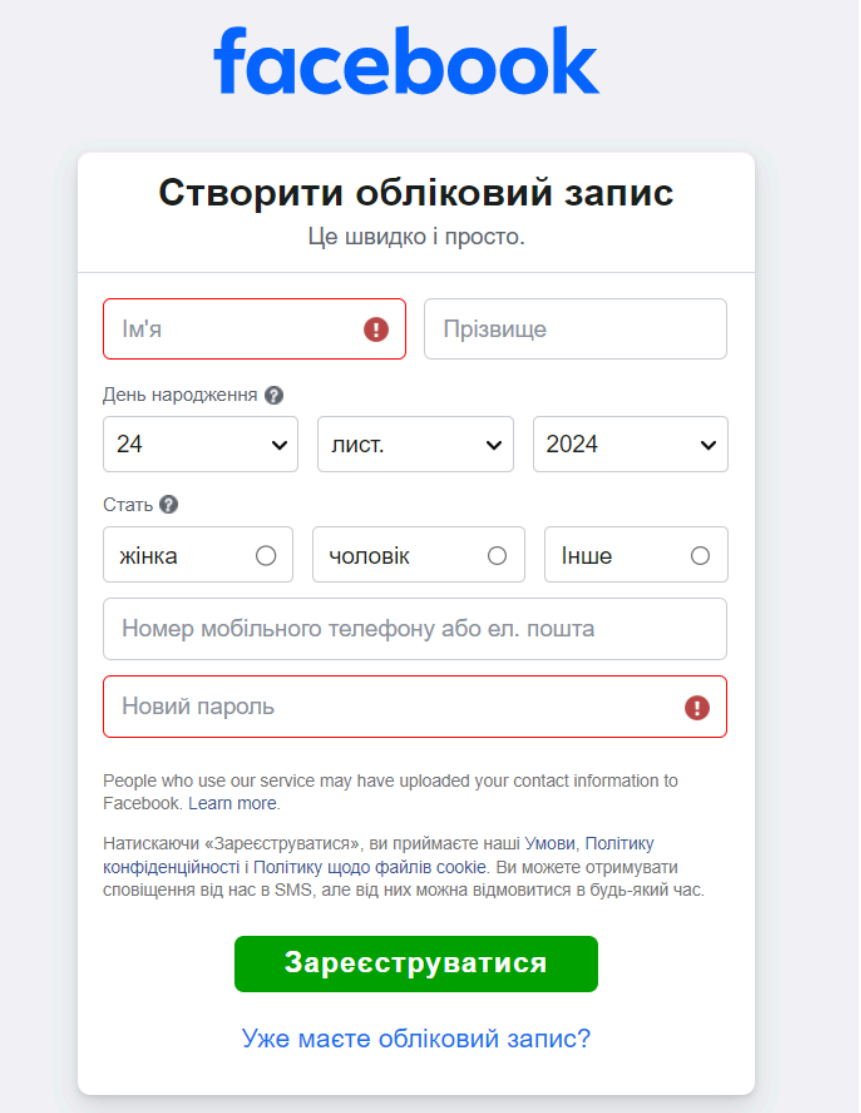


Рисунок 1.5 – Форма реєстрації на Facebook

На рисунку 1.6 можна бачити вже трішки іншу форму для реєстрації, де не потрібно так багато персональних даних. Проте тут є можливість пройти реєстрацію за допомогою аккаунта вищезгаданого Facebook, або ж Google чи Apple, що робить реєстрацію значно простішою для користувачів.

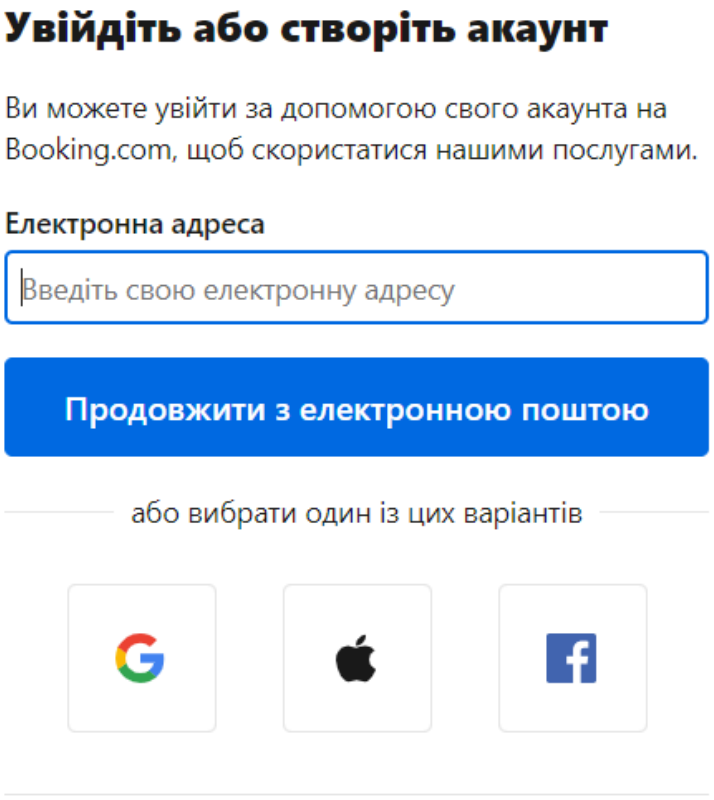


Рисунок 1.6 – Форма реєстрації Booking.com

О крім класичних способів збору даних через форму є ще Cookie-файли є важливим інструментом збору персональних даних, який широко застосовується для аналізу поведінки користувачів. Наприклад, відвідувачі сайтів часто залишають інформацію про свої уподобання, натискання на посилання або тривалість перебування на певних сторінках. Cookie-файли дозволяють зберігати цю інформацію у браузері користувача для формування персоналізованих пропозицій, наприклад, в онлайн-магазинах або сервісах потокового відео.

З юридичної точки зору, багато компаній інтегрують банери про використання файлів cookie. Вони дозволяють користувачам погоджуватись на використання певних категорій cookie або відхиляти їх. Це вимога багатьох нормативно-правових актів, таких як GDPR.

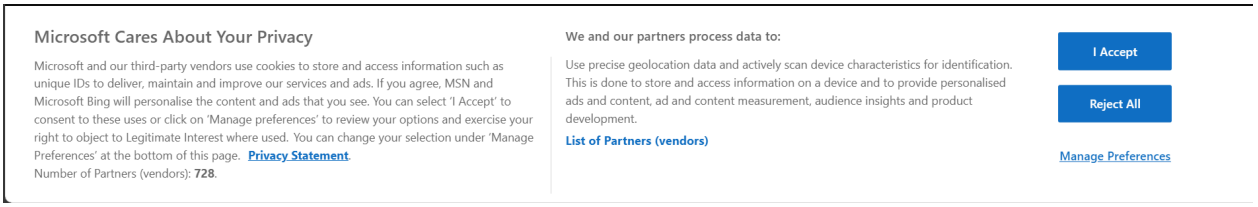


Рисунок 1.7 – Спливаюче вікно згоди на використання файлів cookie.

На рисунку 1.7 показаний типовий банер, що інформує користувача про збір cookie-файлів. Він містить кнопки для налаштування cookie та прийняття політики сайту.

Якщо брати до уваги компанії, які працюють із великим обсягом даних, ключовим елементом є обробка даних через аналітичні платформи. Одним із прикладів є Google Analytics, який надає можливість відстежувати відвідуваність сайту, джерела трафіку, поведінкові метрики тощо. Такі системи дозволяють компаніям не лише покращувати свої сервіси, але й планувати маркетингові кампанії, орієнтуючись на інтереси та потреби користувачів.

Системи аналітики збирають і обробляють персональні дані, як-от IP-адреси, геолокацію користувачів та тип пристрою. Це дає змогу створити більш персоналізований досвід користування.

Збереження персональних даних є не менш важливим етапом усього циклу роботи з ними. Більшість сучасних компаній використовують хмарні платформи, такі як Amazon Web Services (AWS), Microsoft Azure або Google Cloud, для забезпечення надійності та доступності даних.

На рисунку 1.8 показані основні метрики відвідуваності веб-сайту, включаючи кількість унікальних користувачів, джерела трафіку та популярні сторінки.

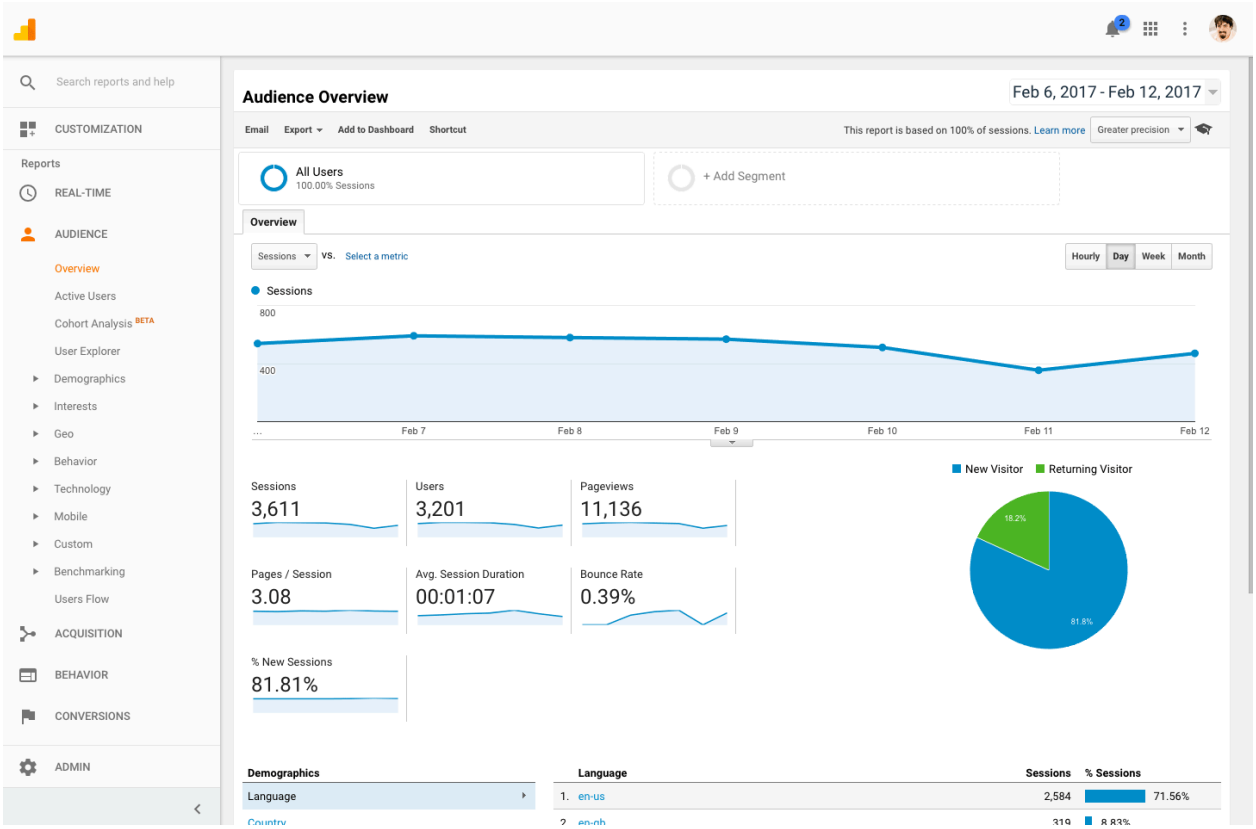


Рисунок 1.8 – Інтерфейс аналітичної системи Google Analytics.

Хмарні сервіси пропонують багаторівневий захист, включаючи шифрування даних як під час передачі, так і в стані зберігання.

Це дає змогу масштабувати зберігання в залежності від зростання обсягу інформації та забезпечувати доступ до даних із будь-якої точки світу. Наприклад, платформа AWS забезпечує інструменти моніторингу доступу до

Штучний інтелект є одним із найсучасніших інструментів для роботи з персональними даними. Наприклад, алгоритми машинного навчання можуть аналізувати великі масиви даних для визначення поведінкових моделей користувачів, персоналізації контенту або навіть прогнозування потреб клієнтів. Такі системи використовуються в маркетингу, банківській справі, охороні здоров'я та інших галузях.

Попри очевидні переваги, використання штучного інтелекту потребує особливого підходу до конфіденційності даних. У багатьох випадках

розробники застосовують методи анонімізації даних перед передачею їх до алгоритму.

Рисунок 1.9 – Робота алгоритму машинного навчання для персоналізації.

На рисунку 1.9 показано, як вхідні дані користувача обробляються алгоритмом для створення персоналізованого контенту або рекомендацій.

Реалізація збору, обробки та зберігання персональних даних є багатогранним процесом, що поєднує в собі технічні, організаційні та юридичні аспекти. Кожен етап — від збору даних через форми або cookie до їх зберігання у хмарних сховищах — вимагає належного рівня захисту та відповідності законодавчим нормам. Візуальні приклади, наведені в цьому розділі, ілюструють основні підходи, які застосовують сучасні компанії для забезпечення прозорості та безпеки роботи з персональними даними.

РОЗДІЛ 2.
ОБҐРУНТУВАННЯ, ВИБІР ТА ВИРІШЕННЯ ЗАДАЧІ ЗБОРУ, ОБРОБКИ ТА ЗАХИСТУ ПЕРСОНАЛЬНИХ ДАНИХ КОРИСТУВАЧІВ ВЕБ-САЙТІВ

2.1 Обґрунтування методу збору та обробки персональних даних

Збір та обробка персональних даних є однією з ключових функцій сучасних цифрових систем, зокрема для додатків, вебсайтів, аналітичних платформ та інших онлайн-сервісів. Персональні дані, такі як імена, адреси електронної пошти, IP-адреси або навіть уподобання користувачів, стають важливим ресурсом для бізнесу. Водночас їх обробка пов'язана з викликами у сфері конфіденційності, безпеки та нормативної відповідності. Це вимагає застосування надійних технологій, фреймворків і методів, які можуть забезпечити ефективний процес збору, обробки та зберігання таких даних.

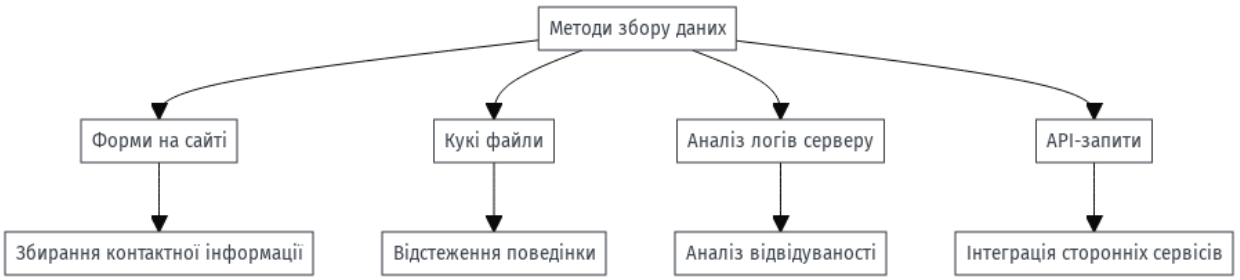


Рисунок 2.1 – Методи збору даних

На рисунку 2.1 зображено основні методи збору даних, такі як форми на сайті, кукі-файли, аналіз логів серверу та API-запити. Від кожного методу вказані їхні ключові функції. Форми збирають контактну інформацію, кукі-файли відстежують поведінку користувачів, аналіз логів серверу дозволяє отримувати статистику відвідувань, а API-запити забезпечують інтеграцію зі сторонніми сервісами.

Процес збору персональних даних часто реалізується через спеціальні API або форми, доступні користувачам через вебсайти чи мобільні додатки.

Основною метою таких API є надання зручного інтерфейсу для взаємодії з користувачем і безпечного передавання отриманих даних на сервер. Для розробки таких систем використовуються різноманітні фреймворки, зокрема Flask та Django, які розглядаються в цій роботі.

Flask є одним із найбільш популярних мікрофреймворків для Python, що дозволяє створювати легкі серверні додатки. Його простота і гнучкість роблять його ідеальним вибором для створення API для збору даних у невеликих або середніх за масштабом проєктах. Flask забезпечує базовий набір функціональності, яку можна легко розширити за допомогою сторонніх бібліотек.

Уявімо, що система збору даних повинна приймати JSON-дані від користувачів, які потім обробляються сервером. Використовуючи Flask, можна створити простий ендпоінт для цієї мети:

```
from flask import Flask, request, jsonify
app = Flask(__name__)
@app.route('/collect_data', methods=['POST'])
def collect_data():
    user_data = request.json
    if user_data:
        return jsonify({"message": "Data received", "data": user_data}), 200
    return jsonify({"error": "No data provided"}), 400
if __name__ == '__main__':
    app.run(debug=True)
```

У цьому коді користувач надсилає JSON-запит через POST-запит, а сервер обробляє і зберігає ці дані для подальшого використання.

Flask має низку переваг, таких як гнучкість і можливість точного контролю кожного аспекту програми. Водночас він вимагає додаткових налаштувань

для складних проєктів, таких як авторизація користувачів чи інтеграція з базами даних.

Для складніших систем може бути доцільно використовувати Django, повнофункціональний фреймворк, що забезпечує інтегровану роботу з базами даних, автоматичне створення API та захист від типових загроз безпеці. Django забезпечує можливість швидкої реалізації CRUD-операцій, що є важливим для великих систем збору даних.

Приклад:

```
from django.http import JsonResponse
from django.views.decorators.csrf import csrf_exempt
import json
@csrf_exempt
def collect_data(request):
    if request.method == 'POST':
        body = json.loads(request.body)
        return JsonResponse({"message": "Data received", "data": body})
    return JsonResponse({"error": "Invalid request method"}, status=400)
```

Django має переваги у вигляді вбудованої підтримки масштабування, високої безпеки та автоматизації багатьох задач.

Існують і інші способи збору даних, зокрема використання сторонніх сервісів, таких як Google Forms або Typeform, які дозволяють швидко збирати дані без створення інфраструктури. Ці сервіси зручні, але мають обмеження у налаштуванні та масштабуванні. Інший підхід — це використання No-Code платформ, наприклад Zapier, які можуть автоматизувати збір і передачу даних до баз даних або інших систем.

Рисунок 2.2 – Порівняння Django та Flask

Рисунок 2.2 порівнює Django і Flask, два популярних Python-фреймворки, за їхніми основними характеристиками. Django відзначається своєю багатофункціональністю та автоматизацією. Він має вбудовану адмін-панель, тісну інтеграцію з базами даних та високий рівень безпеки, що робить його ідеальним для великих і складних проєктів, які потребують швидкого впровадження стандартних рішень.

Flask, навпаки, є мінімалістичним і гнучким. Його основна перевага — простота у використанні та здатність легко адаптуватися до специфічних потреб. Він краще підходить для невеликих або модульних проєктів, де розробник хоче повного контролю над усіма аспектами застосунку.

На рисунку 2.2 відображено, що Django забезпечує більш структурований підхід, включаючи готову адмін-панель та широкі можливості повторного використання компонентів. Його висока інтеграція з

ORM дозволяє легко працювати з базами даних. Flask, хоча й вимагає більше ручних налаштувань, надає розробнику повну свободу у виборі інструментів та бібліотек. Безпека є ще однією ключовою перевагою Django, тоді як Flask може потребувати додаткових заходів у цій сфері.

Це порівняння демонструє, що вибір між Django і Flask залежить від масштабу проєкту та потреб у гнучкості або стандартизації. Django краще підходить для великих проєктів із комплексними вимогами, тоді як Flask є ідеальним вибором для проєктів, де потрібна кастомізація та контроль.

Після збору дані потребують обробки, яка включає очищення, фільтрацію, перетворення та аналіз. Для цього найчастіше використовуються бібліотеки Pandas, NumPy, а також спеціалізовані інструменти для великих даних, такі як PySpark.

Pandas є універсальним інструментом для роботи з табличними даними. Приклад:

```
import pandas as pd
data = [
    {"name": "John Doe", "email": "john@example.com", "age": "25"},
    {"name": "Jane Smith", "email": "jane@example.com", "age": "unknown"},
    {"name": "Alice Brown", "email": "alice@example.com", "age": "30"}
]
df = pd.DataFrame(data)
df['age'] = pd.to_numeric(df['age'], errors='coerce')
df = df.dropna()
print(df)
```

Для роботи з великими даними можна використовувати PySpark, який підтримує розподілену обробку даних. Крім того, бібліотеки на кшталт Scikit-learn дозволяють інтегрувати машинне навчання для аналізу персональних даних.

Також збір даних через cookies є одним із ключових способів взаємодії між вебсайтами та користувачами. Cookies — це невеликі файли, які браузер користувача зберігає локально для обміну інформацією із сервером під час роботи на сайті. Цей механізм дозволяє сайту запам'ятовувати дані користувача, наприклад, його ідентифікатор, налаштування чи сесійні параметри, що є важливим для персоналізації послуг і аналітики.

Реалізація збору даних через cookies може бути простою, як показує приклад на Python з використанням фреймворку Flask. У коді створюється два ендпоінти: перший для встановлення cookies і другий для їх читання. При встановленні сервер надсилає у браузер користувача файл cookie, який містить дані, наприклад, ідентифікатор користувача або налаштування теми сайту. У подальших запитах браузер автоматично відправляє цей файл назад на сервер, дозволяючи розпізнати користувача. У прикладі використовується метод set_cookie для створення cookies із зазначенням терміну дії, а також метод request.cookies.get для їх зчитування.

Такий підхід має очевидні переваги, серед яких простота реалізації, підтримка всіма сучасними браузерами та можливість зберігати важливі дані для оптимізації роботи сервісу. Однак cookies мають і певні обмеження. Наприклад, максимальний обсяг даних, які можна зберігати у cookies, не перевищує 4 КБ. Крім того, користувач може вручну видалити cookies або заблокувати їх у налаштуваннях браузера. Безпека також є важливим фактором, адже дані в cookies можуть бути викрадені, якщо вони не зашифровані або не передаються через захищене з'єднання.

Для покращення безпеки cookies можна застосовувати шифрування значень. Наприклад, бібліотека cryptography у Python дозволяє зашифрувати дані перед їхнім збереженням. Окрім цього, варто налаштовувати cookies так, щоб вони були доступними лише для сервера (HTTP-only) і передавалися виключно через захищені з'єднання (secure). Такий підхід значно знижує ризик компрометації даних, особливо якщо вони стосуються конфіденційної інформації.

Приклад:

```
from flask import Flask, request, make_response
app = Flask(__name__)
@app.route('/set_cookie')
def set_cookie():
        response = make_response("Cookie встановлено!")
        response.set_cookie('user_id', '12345', max_age=60*60*24)  # Дійсність cookie 1 день
        response.set_cookie('theme', 'dark', max_age=60*60*24, secure=True, httponly=True)
        return response
@app.route('/get_cookie')
def get_cookie():
        user_id = request.cookies.get('user_id')  # Зчитування cookie 'user_id'
        theme = request.cookies.get('theme')  # Зчитування cookie 'theme'
        if user_id and theme:
           return f"User ID: {user_id}, Theme: {theme}"
        return "Cookies не знайдено!"
if __name__ == '__main__':
        app.run(debug=True)
```

Цей приклад демонструє, як працювати з cookies у Python за допомогою фреймворку Flask. Код містить два основні ендпоінти: перший відповідає за встановлення cookies, другий – за їх зчитування. Для встановлення cookies створюється HTTP-відповідь, до якої додається пара ключ-значення. Наприклад, встановлюється user_id зі значенням 12345 та theme зі значенням dark. Додатково вказується тривалість дії cookie (1 день), а також параметри безпеки: secure=True, щоб cookie передавалося лише через

HTTPS, і httponly=True, щоб обмежити доступ до нього лише сервером, забороняючи доступ через JavaScript.

Для зчитування cookies використовується request.cookies.get, який отримує значення cookie за ключем. У разі успішного отримання значення відображається у відповідь, а якщо cookies не знайдено, сервер повертає повідомлення про їхню відсутність.

Цей механізм використовується для персоналізації користувацького досвіду, наприклад, для зберігання ідентифікатора користувача, щоб сервер міг розпізнати його при наступному візиті, або для збереження налаштувань інтерфейсу, таких як обрана тема. Завдяки таким можливостям cookies стали важливим елементом сучасного веб-розроблення. У цьому прикладі також враховуються аспекти безпеки, що мінімізує ризик викрадення cookies через атаки типу XSS чи MITM.

На таблиці 2.1 показано порівняння трьох основних методів збору персональних даних: cookies, форм і запитів API. Кожен метод оцінено за перевагами, такими як точність даних, автоматизація процесів і ефективність, а також за недоліками, серед яких обмеження законодавством, ризики безпеки та залежність від активності користувачів. Таблиця слугує інструментом для вибору найбільш доцільного методу збору залежно від специфіки завдань.

Персональні дані стали ключовим активом сучасного цифрового середовища, а їх ефективний збір і обробка потребують комплексного підходу, що поєднує передові технології та відповідність нормативно-правовим вимогам. Розглянуті методи, зокрема cookies, форми й API, демонструють різноманітність інструментів, які можуть бути адаптовані до специфічних завдань бізнесу чи дослідження.

Досвід доводить, що вибір методології збору даних не може бути універсальним. Він повинен враховувати не лише технічні можливості, але й потенційні ризики, такі як витоки даних, складнощі з дотриманням законодавства та етичні аспекти.

Таблиця 2.1 – Переваги та недоліки різних методів збору даних

Метод збору	Переваги	Недоліки
Cookies	- Автоматизоване збирання даних. - Використовуються для персоналізації користувацького досвіду. - Ефективність у відстеженні поведінки.	- Обмеження через законодавство (GDPR). - Залежність від згоди користувачів. - Легко очищуються.
Forms	- Точні дані безпосередньо від користувача. - Можливість збору глибоких і детальних даних.	- Залежність від активності користувачів. - Можливість введення неправдивої інформації.
API Requests	- Швидкість і автоматизація. - Доступ до структурованих даних. - Інтеграція з іншими сервісами.	- Ризик уразливостей API. - Може потребувати значних технічних ресурсів для інтеграції та підтримки.

Саме інтеграція технологій із врахуванням цих факторів дозволяє створювати рішення, які одночасно є ефективними, безпечними та адаптованими до сучасних викликів.

2.2 Обґрунтування методу захисту персональних даних

Захист персональних даних є одним із найважливіших аспектів роботи з інформацією в цифровому середовищі. У міру зростання кількості зібраних даних і ризиків, пов'язаних із їхнім використанням, питання безпеки стає ключовим для бізнесу, дослідників і регуляторів. Ефективна захищеність даних передбачає інтеграцію технічних, організаційних і правових заходів, які гарантують конфіденційність, цілісність і доступність інформації.

Методи захисту персональних даних поділяються на різні рівні: шифрування, псевдонімізацію, анонімізацію, контроль доступу, використання VPN і фаєрволів. Кожен із цих методів має свої особливості, переваги й недоліки.

Шифрування є основним методом захисту даних, який перетворює інформацію в недоступний для розуміння формат, поки вона не буде розшифрована за допомогою ключа. Симетричне шифрування, як-от AES (Advanced Encryption Standard), швидке й ефективне, але потребує надійного зберігання ключів. Асиметричне шифрування, наприклад RSA, вирішує проблему передачі ключів, але є повільнішим і вимагає більше ресурсів.

Приклад:

```
from cryptography.fernet import Fernet
key = Fernet.generate_key()
cipher = Fernet(key)
data = "Це конфіденційна інформація".encode()
encrypted_data = cipher.encrypt(data)
print("Зашифровані дані:", encrypted_data)
decrypted_data = cipher.decrypt(encrypted_data)
print("Розшифровані дані:", decrypted_data.decode())
```

Переваги шифрування включають високий рівень захисту даних навіть у разі доступу до носія інформації. Проте недоліками є складність управління ключами та додаткове навантаження на систему.

Анонімізація даних передбачає видалення або зміну ідентифікаторів, що дозволяють зв’язати інформацію з конкретною особою. Цей підхід дозволяє використовувати дані для досліджень чи статистичних аналізів без ризику порушення приватності. Прикладом може бути видалення імен і адрес електронної пошти з бази даних. Проте анонімізація може бути складною для великих наборів даних із багатьма перехресними зв’язками.

Псевдонімізація є схожою до анонімізації, але передбачає збереження можливості відновлення ідентифікаторів через певний ключ. Наприклад, використання унікальних кодів замість імен у медичних даних дозволяє забезпечити конфіденційність пацієнтів, зберігаючи можливість ідентифікації в разі потреби.

Контроль доступу забезпечує захист даних через розмежування прав користувачів. Це може бути реалізовано через багаторівневі системи авторизації. Наприклад, адміністратор бази даних має доступ до всієї інформації, тоді як звичайний користувач може переглядати лише певні дані.

Вразливості при використанні методів захисту часто пов’язані з людським фактором або неправильним налаштуванням системи. Наприклад, слабкі паролі, відсутність оновлень програмного забезпечення чи неправильне управління ключами можуть призвести до компрометації даних.

Реальні загрози, як-от атаки типу "людина посередині" (MITM) або злам фаєрволів, також можуть обійти захисні механізми. Щоб мінімізувати ці ризики, рекомендується використовувати комбінацію методів: шифрування для передачі й зберігання даних, а також багатофакторну автентифікацію для доступу.

Приклад:

```
import pyotp
import qrcode
secret = pyotp.random_base32()
print("Секретний ключ:", secret)
totp = pyotp.TOTP(secret)
print("Код для автентифікації:", totp.now())
uri = totp.provisioning_uri(name="user@example.com", issuer_name="My App")
qrcode.make(uri).show()
```

Однією з найбільш безпечних моделей є Zero Trust, що виходить із припущення, що жоден користувач чи пристрій не може бути автоматично визнаний довіреним, навіть якщо вони знаходяться всередині корпоративної мережі. Замість того, щоб припускати, що все в межах мережі безпечне, у Zero Trust кожен запит до системи, будь то з боку внутрішнього користувача або зовнішнього, проходить через ретельну перевірку. Доступ до ресурсів надається тільки після підтвердження автентичності та авторизації користувача чи пристрою, навіть якщо вони знаходяться в межах організаційної мережі.

У цьому підході також особливу увагу приділяється принципу мінімальних привілеїв, що означає, що користувачам і системам надається лише той доступ до ресурсів, який є необхідним для виконання їхніх завдань. Навіть якщо пристрій чи користувач мають певний доступ до мережі або ресурсів, їх права обмежуються виключно для виконання конкретних функцій. Щоб уникнути ситуацій, коли шкідливі програми або зловмисники можуть рухатись по всій мережі, застосовується концепція мікросегментації, яка передбачає розподіл мережі на декілька ізольованих сегментів, що обмежує поширення атак.

Крім того, важливою складовою моделі є постійний моніторинг. Система повинна аналізувати діяльність користувачів, пристроїв та усіх процесів в реальному часі для виявлення аномальних поведінкових патернів, що можуть вказувати на спроби несанкціонованого доступу або атаки. Такий підхід дозволяє швидко реагувати на загрози та знижує ймовірність успішної атаки.

Ця модель набула великої популярності через зміни в технологічному середовищі, зокрема перехід до хмарних технологій та гібридних мереж, де традиційні методи забезпечення безпеки, орієнтовані на "замкнуті" мережі, вже не працюють так ефективно. Zero Trust дозволяє гарантувати захист навіть за умов віддаленого доступу до корпоративних ресурсів, що особливо

актуально у сучасному світі, де співробітники можуть працювати з будь-якого місця за допомогою різних пристроїв.

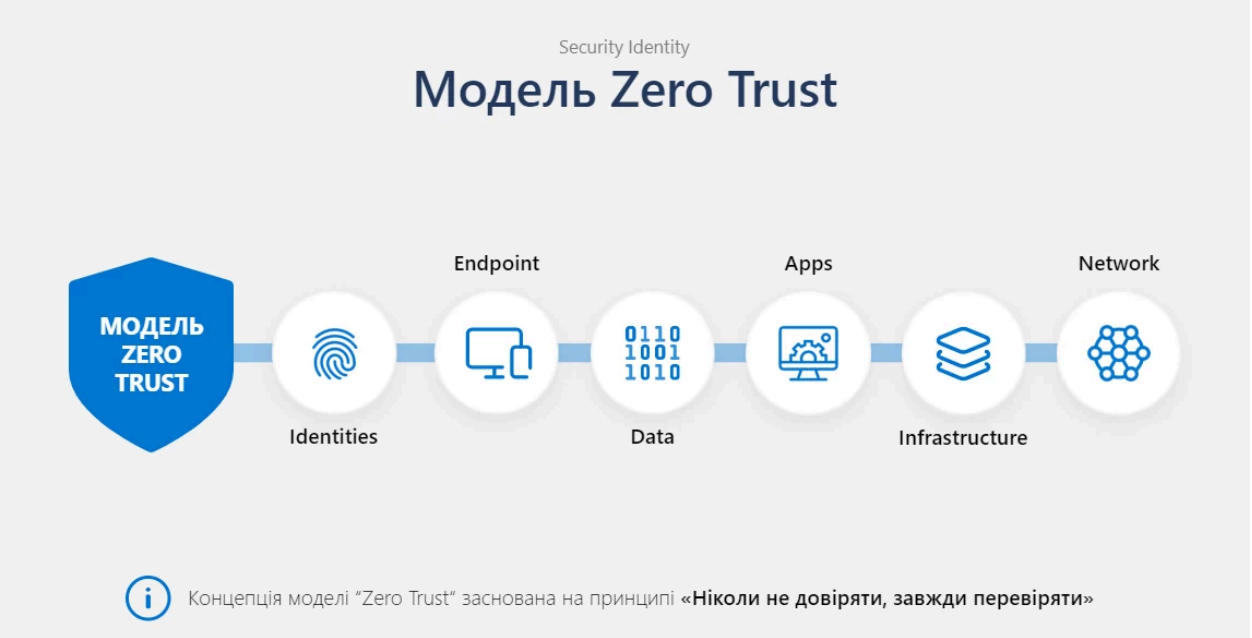


Рисунок 2.3 – Модель захисту персональних даних Zero Trust

На рисунку 2.3 показана архітектура Zero Trust де можна побачити взаємодію між користувачами, пристроями та системами, що запитують доступ до ресурсів. Всі ці елементи взаємодіють через різні рівні захисту: спочатку перевіряється автентичність користувача, далі застосовуються політики доступу, а після цього здійснюється безперервний моніторинг. Ця модель зазвичай також включає шари перевірки та сегментації, де кожен компонент мережі ізольований один від одного для зниження ймовірності атаки.

Приклад простої реалізації Zero Trust:

```
import time
def authenticate_user(token):
    for user, details in USERS.items():
        if details["token"] == token:
            print(f"User {user} authenticated successfully.")
            return user
```

```
    print("Authentication failed!")
    return None
def authorize_user(user, action):
    role = USERS[user]["role"]
    allowed_actions = ACCESS_POLICIES[role]
    if action in allowed_actions:
        print(f"User {user} is authorized to perform '{action}'.")
        return True
    else:
        print(f"User {user} is NOT authorized to perform '{action}'.")
        return False
def perform_action(token, action):
    user = authenticate_user(token)
    if not user:
        log_activity("Unknown", action, "Failed")
        return
    if authorize_user(user, action):
        log_activity(user, action, "Success")
        print(f"Action '{action}' performed successfully!")
    else:
        log_activity(user, action, "Failed")
if __name__ == "__main__":
    perform_action("abcd1234", "read")   # Успішна дія
    perform_action("abcd1234", "delete") # Успішна дія
    perform_action("xyz9876", "write")   # Невдала дія
    perform_action("wrong_token", "read") # Невдала автентифікація
```

В цьому коді можемо бачити, що користувачі ідентифікуються за унікальним токеном. Якщо токен невірний, доступ забороняється. Також кожен користувач має роль із визначеними дозволами. Політики доступу обмежують дії, які може виконати користувач.

Машинне навчання також відіграє важливу роль у захисті даних. Замість того щоб реагувати на відомі загрози, системи машинне навчання може виявляти аномалії, які вказують на нові або раніше невідомі атаки. Наприклад, кластеризація логів доступу дозволяє ідентифікувати підозрілі шаблони поведінки користувачів.

Приклад:

```
from sklearn.ensemble import IsolationForest
import numpy as np
data = np.random.rand(100, 2)
data_with_anomaly = np.append(data, [[0.99, 0.99]], axis=0)
model = IsolationForest(contamination=0.01)
model.fit(data)
anomalies = model.predict(data_with_anomaly)
print("Аномалії виявлені:", anomalies[-1])  # Останній запис має позначення аномалії
```

Цей підхід особливо корисний для захисту динамічних систем, які постійно змінюються.

Методи захисту персональних даних активно розвиваються через постійне зростання загроз. Новітні підходи, такі як використання штучного інтелекту для виявлення аномалій чи блокчейну для зберігання інформації, відкривають нові горизонти для забезпечення безпеки. Проте вони також вимагають глибоких знань і ретельного аналізу для правильного впровадження.

Таким чином, забезпечення захисту персональних даних є багатогранним завданням, що вимагає не лише технічної компетентності, але

й уважного дотримання етичних і юридичних норм. Ефективна стратегія захисту повинна бути адаптованою, динамічною та комплексною, об’єднуючи різноманітні підходи та технології.

2.3 Визначення форматів та принципу збереження персональних даних

Збереження персональних даних є ключовим аспектом у сучасних системах, оскільки воно пов’язане з гарантуванням безпеки, конфіденційності та доступності інформації. Основними форматами збереження персональних даних є структуровані, напів структуровані та неструктуровані дані, кожен з яких має свої особливості використання та переваги.

Структуровані дані, як правило, зберігаються у реляційних базах даних (наприклад, MySQL, PostgreSQL). Вони організовані у вигляді таблиць, де кожен стовпець відповідає певному атрибуту, а кожен рядок — окремому запису. Це забезпечує швидкий доступ до інформації, простоту фільтрації та маніпуляцій із даними. Наприклад, у таблиці можуть зберігатися такі дані, як ім'я, адреса та контактний номер користувача. Цей формат підходить для систем, які мають чітку структуру та визначені вимоги до зберігання.

Напівструктуровані дані використовують формати на зразок JSON, XML або YAML. Вони дозволяють зберігати складні, гнучкі дані, які можуть мати динамічну структуру. Наприклад, JSON часто використовується у веб-застосунках для зберігання профілів користувачів, оскільки він легко інтегрується з мовами програмування, такими як Python і JavaScript. Цей формат забезпечує баланс між структурованістю та гнучкістю, дозволяючи зберігати як однорідні, так і неоднорідні дані.

Неструктуровані дані, такі як зображення, відео, текстові файли або аудіо, зберігаються в системах типу NoSQL (наприклад, MongoDB або Elasticsearch) або в хмарних сховищах, таких як AWS S3 чи Google Cloud Storage. Цей формат підходить для великих обсягів даних, які не мають жорсткої структури. Наприклад, система відеоспостереження може зберігати

великі обсяги відео в неструктурованому форматі з метаінформацією про дату, час і місце запису.

Принципи збереження персональних даних базуються на таких основних аспектах, як безпека, цілісність та відповідність законодавчим вимогам. Дані повинні бути доступні тільки уповноваженим особам, що досягається через механізми шифрування, багатофакторної автентифікації та системи ролей і дозволів. Для забезпечення цілісності використовуються методи хешування та резервного копіювання.

Приклад того, як зберігаються й шифруються дані в MySQL, може виглядати так. Припустимо, ми зберігаємо особисту інформацію (ім'я, email, пароль). Ми застосовуємо шифрування для захисту чутливих даних.

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL UNIQUE,
    password_hash VARBINARY(255) NOT NULL,
    encrypted_ssn VARBINARY(255) NOT NULL
);
```

РОЗДІЛ 3.
РЕЗУЛЬТАТИ ВИРІШЕННЯ ЗАДАЧІ

3.1 Обґрунтування вибору інструменту для розробки збору, обробки та захисту персональних даних на веб сайті

Для розробки веб сайту, який забезпечує збір, обробку та захист персональних даних, критично важливо правильно вибрати інструменти та технології, адже від цього залежить функціональність, безпека та масштабованість системи. У процесі прийняття рішення враховуються кілька ключових аспектів, таких як підтримка безпеки, зручність інтеграції, ефективність обробки даних, зручність розробки та наявність спільноти підтримки.

Одним із найпоширеніших виборів для реалізації таких систем є фреймворки Django та Flask, які базуються на Python. Django є комплексним фреймворком, який пропонує «із коробки» всі необхідні компоненти, такі як ORM для роботи з базами даних, вбудований модуль аутентифікації та авторизації, а також готові засоби для підвищення безпеки, наприклад захист від SQL-ін'єкцій, CSRF-атак та XSS-вразливостей. Flask, у свою чергу, є більш гнучким і легковажним фреймворком, який дозволяє розробнику обирати бібліотеки та компоненти самостійно, що може бути корисним у випадках, коли потрібна максимальна кастомізація. У цій роботі обирається Django, оскільки він краще підходить для швидкої розробки надійних рішень із високим рівнем безпеки.

При виборі бази даних віддається перевага MySQL або PostgreSQL завдяки їх продуктивності, надійності та підтримці засобів для шифрування даних. PostgreSQL має ширші можливості для реалізації криптографічних функцій, що робить його гарним вибором для систем із підвищеними вимогами до захисту даних.

Для збору даних через веб інтерфейс використовуються сучасні інструменти фронтенду, такі як React чи Vue.js, які дозволяють створювати

інтерактивні та динамічні форми збору даних. Такі форми можуть включати валідацію введених даних на стороні клієнта перед їх відправленням на сервер. Валідація зменшує кількість помилок та забезпечує початковий рівень захисту.

У процесі обробки персональних даних важливо використовувати шифрування, наприклад, через бібліотеку Python Cryptography або вбудовані можливості фреймворку Django для роботи із зашифрованими полями. Наприклад, персональні дані, такі як номери телефонів чи адреси, можуть бути збережені у базі даних в зашифрованому вигляді, що унеможливлює їх перегляд сторонніми особами навіть у разі компрометації бази.

Захист даних забезпечується на всіх рівнях: від використання HTTPS для передачі даних між клієнтом і сервером до налаштування міжмережевого екрану та політик доступу до серверу. Крім того, для захисту паролів користувачів застосовується хешування із використанням алгоритму bcrypt, який забезпечує захист від атак на основі перебору.

3.2 Реалізація збору персональних даних

Як приклад реалізації послужить простий веб сайт, який був створений для продажу пазлів. Як і в любому інтернет магазині, сайт має свої методи збору потрібних йому персональних даних.

На рисунку 3.1 зображено форму реєстрації, яка слугує для збору персональних даних користувачів. Користувач заповнює поля "Прізвище та ім'я", "E-mail", "Пароль" і "Повторіть пароль". Ці дані вводяться через веб-інтерфейс і відправляються на сервер за допомогою HTTP POST-запиту. Сервер отримує ці дані, проводить валідацію, перевіряє відповідність паролів і дотримується стандартів безпеки під час їх обробки, включно з шифруванням паролів перед збереженням у базі даних.

Форма відповідає сучасним вимогам безпеки, оскільки на клієнтському боці можна додати базову валідацію через JavaScript або HTML5, наприклад,

використовуючи атрибути типу required. З боку сервера всі дії з обробки даних виконуються в захищеному середовищі, включаючи використання хеш-функцій для збереження паролів.

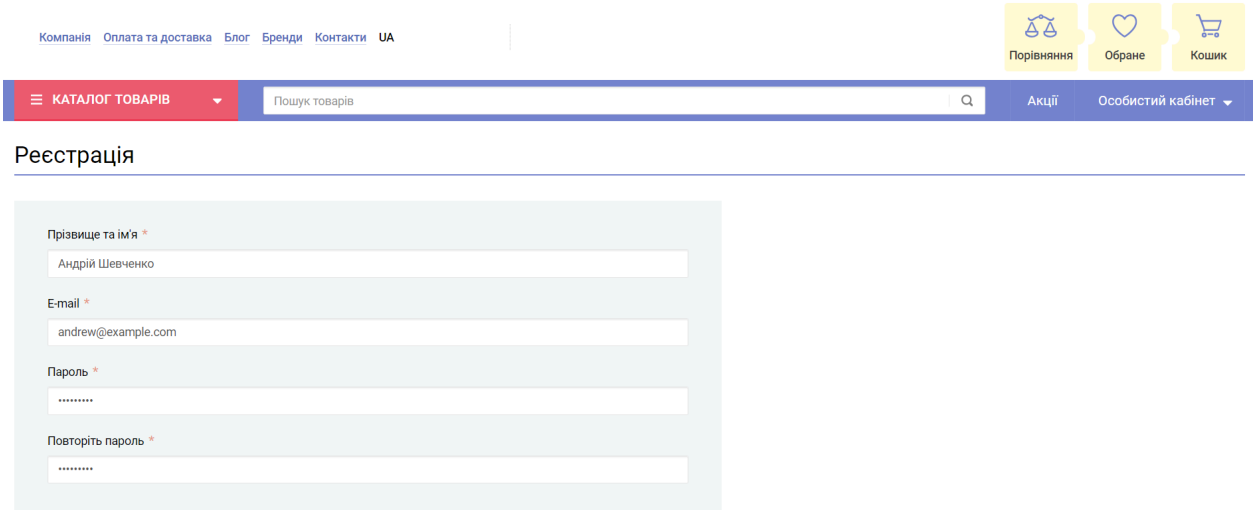


Рисунок 3.1 – Збір персональних даних через форму реєстрації

Реалізація Django:

```
from django.shortcuts import render, redirect
from django.contrib.auth.models import User
from django.contrib import messages
from django.contrib.auth.hashers import make_password
def registration_view(request):
    if request.method == 'POST':
        full_name = request.POST.get('full_name')
        email = request.POST.get('email')
        password = request.POST.get('password')
        confirm_password = request.POST.get('confirm_password')

        if password != confirm_password:
            messages.error(request, "Паролі не збігаються")
            return render(request, 'registration.html')
```

```
        if User.objects.filter(email=email).exists():
            messages.error(request, "Користувач із таким email уже існує")
            return render(request, 'registration.html')
        user = User(
            full_name=full_name,
            email=email,
            created_at=created_at),
            password=make_password(password)
        )
        user.save()
        messages.success(request, "Реєстрація успішна! Тепер ви можете увійти.")
        return redirect('login')
    return render(request, 'registration.html')
```

Реалізація HTML:

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Реєстрація</title>
</head>
<body>
    <h2>Реєстрація</h2>
    <form method="POST">
        {% csrf_token %}
        <label for="full_name">Прізвище та ім'я *</label>
        <input type="text" name="full_name" id="full_name" required>
```

```
        <label for="email">E-mail *</label>
        <input type="email" name="email" id="email" required>
        <label for="password">Пароль *</label>
        <input type="password" name="password" id="password" required>
        <label for="confirm_password">Повторіть пароль *</label>
         <input type="password" name="confirm_password" id="confirm_password"
required>
        <button type="submit">Реєстрація</button>
    </form>
</body>
</html>
```

Тут можна побачити що користувач вводить дані у форму реєстрації, які сервер отримує через request.POST. Для кожного поля перевіряється наявність даних, а для паролів — їх відповідність. Додатково виконується перевірка унікальності email. Пароль перед збереженням у базу даних хешується за допомогою функції make_password, що гарантує безпеку навіть у випадку компрометації бази.

Збереження даних відбувається у стандартній моделі Django User, яка забезпечує інтеграцію із системами аутентифікації. Дані шифруються і зберігаються у базі, що відповідає принципам захисту персональних даних згідно з законодавчими вимогами.

Ця реалізація є простою, але ефективною. Вона легко розширюється за рахунок використання додаткових перевірок або інтеграції з іншими бібліотеками.

В свою чергу HTML-файл для форми реєстрації представляє структуру, яку бачить користувач при взаємодії з сайтом. Ця форма дозволяє вводити дані, такі як ім'я, прізвище, email, пароль та підтвердження пароля, які потім

надсилаються на сервер для обробки. Використовується метод POST, що є більш безпечним, ніж GET, оскільки дані не передаються через URL-адресу.

Тег <form> містить поле для токена захисту {% csrf_token %}, яке необхідне для запобігання CSRF-атакам. Поля вводу створені за допомогою тегу <input> і мають атрибути для налаштування типу даних, таких як текст, email або пароль, що забезпечує правильність введення інформації. Поля мають мітки через тег <label>, який асоціюється з відповідним полем за допомогою атрибута for, роблячи форму зрозумілою для користувача.

Форма завершується кнопкою відправлення, яка надсилає дані на сервер для обробки. Валідація забезпечується атрибутом required, який гарантує, що всі поля будуть заповнені перед відправкою. HTML-код є базовим і може бути розширений за допомогою CSS для покращення зовнішнього вигляду або JavaScript для додаткової клієнтської валідації, забезпечуючи зручність використання. Ця структура підходить для простих реєстраційних форм, з акцентом на безпеку передачі даних і їх обробку.

Тепер потрібно впевнитись що дані успішно були відправленні в базу даних для подальшого збереження на їх обробки.

id [PK] integer	name character varying (100)	email character varying (150)	password character varying (255)	created_at timestamp without time zone
1	Андрій Шевченко	andrew@example.com	Qwerty123	2024-12-08 20:39:40.75628

Рисунок 3.2 – Зібрані з форми дані у таблиці бази даних

Можемо спостерігати що наші дані успішно збереглися у таблиці і тепер можна їх використовувати для обробки чи захисту даних.

3.3 Реалізація обробки персональних даних

Обробка персональних даних була реалізована таким чином, щоб дані зібрані під час покупок товарів, допомагали генерувати певні рекомендації та відправляти їх на електронну пошту як лист з товарами які могли би

сподобатися клієнту. Таку розсилку клієнт завжди зможе відключити відкривши лист надісланий на пошту.

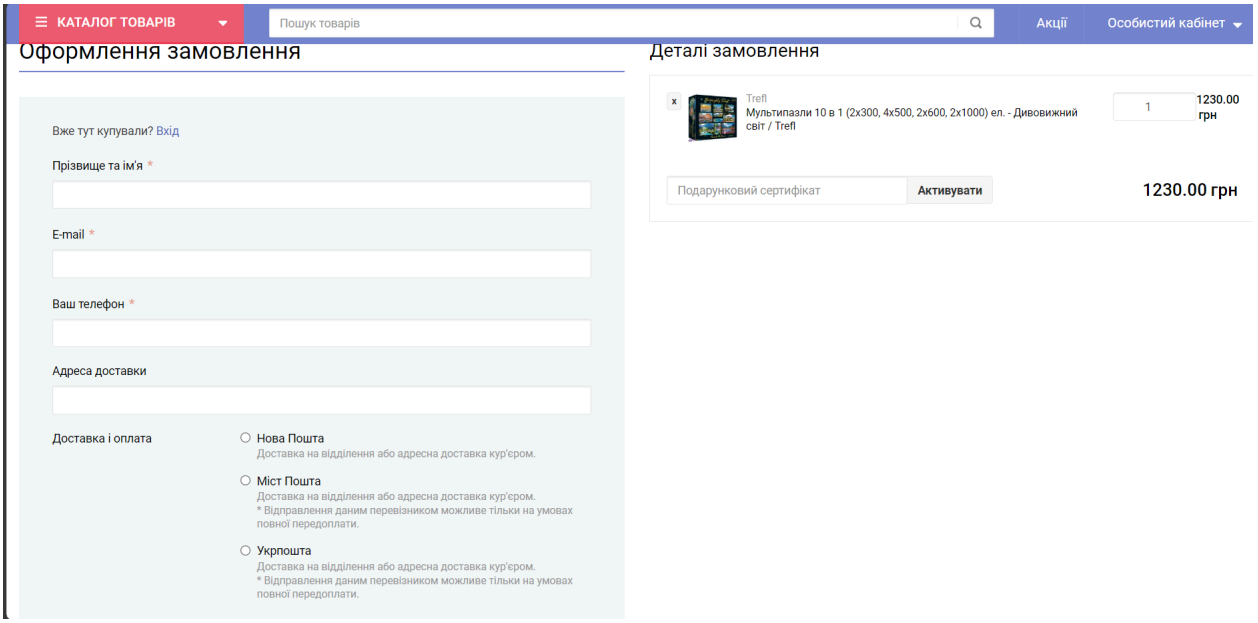


Рисунок 3.3 – Вікно покупки товару

Для того щоб побачити роботу рекомендацій, зробимо замовлення на певний товар, та побачимо що буде запропоновано в якості рекомендованого товару на електронній пошті.

Для реалізації даного функціоналу ми створимо моделі бази даних:

```
from django.db import models
class User(models.Model):
    email = models.EmailField(unique=True)
    name = models.CharField(max_length=255)
class Product(models.Model):
    name = models.CharField(max_length=255)
    description = models.TextField()
class Order(models.Model):
     user = models.ForeignKey(User, on_delete=models.CASCADE,
related_name="orders")
```

```
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    created_at = models.DateTimeField(auto_now_add=True)
```

Модель User визначає структуру збереження даних про користувачів, таких як ім'я та email. Це дозволяє зберігати інформацію про кожного користувача і використовувати її для ідентифікації отримувачів рекомендацій. Модель Product представляє товари, доступні в магазині, містить їхні назви та описи. Модель Order зберігає історію покупок користувачів, поєднуючи User і Product через зовнішні ключі. Використання цих моделей дозволяє ефективно обробляти дані та легко виконувати пошук або фільтрацію за покупками.

Ми шукаємо товари, які ще не були замовлені:

```
from django.db.models import Count
from .models import User, Product, Order
def get_recommendations(user):
    purchased_product_ids =
Order.objects.filter(user=user).values_list('product_id', flat=True)
    recommendations =
Product.objects.exclude(id__in=purchased_product_ids)[:5]  # Обмежуємо до 5
рекомендацій
    return recommendations
```

Для генерації рекомендацій створюється функція get_recommendations. Її завдання полягає у визначенні, які товари користувач ще не купував. Спершу отримуються ID товарів, які були замовлені конкретним користувачем. Потім через метод exclude з бази даних виключаються ці товари, залишаючи тільки ті, що користувач не бачив. Використання ORM Django дозволяє зручно і ефективно працювати з даними, використовуючи зрозумілий для Python синтаксис, без необхідності писати складні SQL-запити.

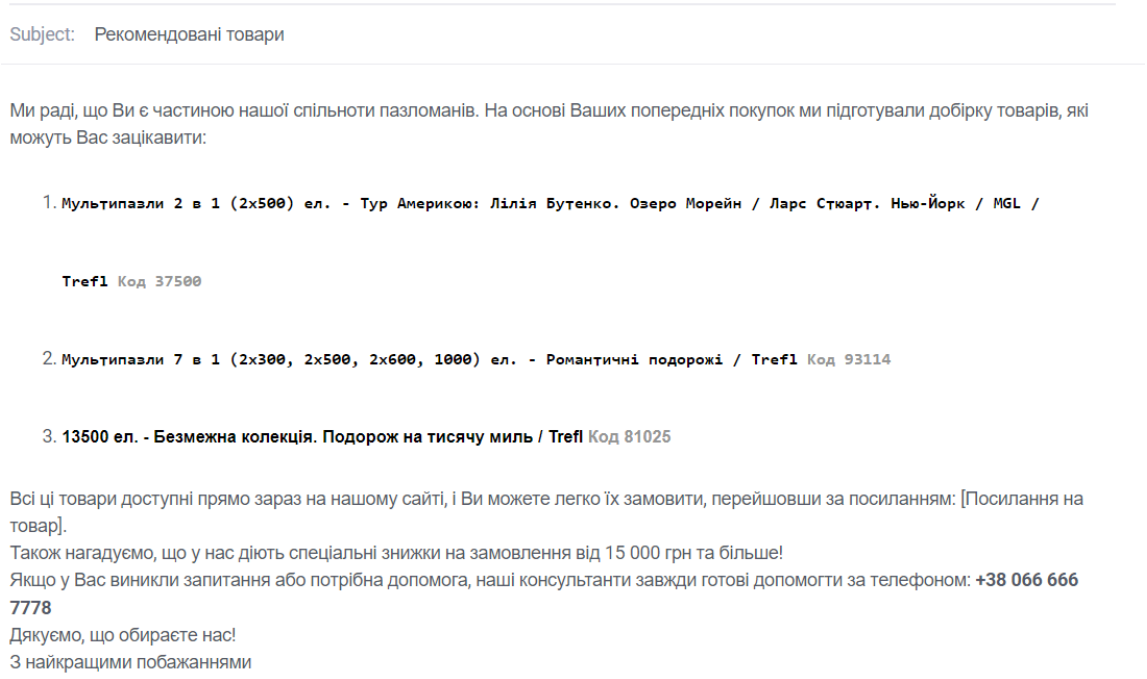
```
from django.core.mail import send_mail
from django.conf import settings
def send_recommendations_email(user, recommendations):
    subject = "Рекомендації на основі ваших покупок"
    message = "Ми підібрали для вас кілька товарів:\n\n"
    message += "\n".join([f"- {product.name}" for product in recommendations])
    message += "\n\nДякуємо, що обираєте нас!
    send_mail(
        subject,
        message,
        settings.DEFAULT_FROM_EMAIL,
        [user.email],
        fail_silently=False,
    )
```

Для відправлення листів використовується функція send_mail, яка є частиною вбудованого в Django модуля для роботи з електронною поштою.

Вибір цього інструменту обґрунтований його простотою у використанні, гнучкістю налаштувань SMTP і надійністю. Функція отримує тему, текст листа, адресу відправника та отримувача.

Формування тексту листа виконується динамічно: перелік рекомендованих товарів додається в тіло листа через об'єднання їхніх назв. Після цього функція send_mail викликається для надсилання листа.

На рисунку 3.4. представлено форму електронного листа, що формується на підставі оброблених персональних даних користувача інтернет-магазину.



Subject: Рекомендовані товари

Ми раді, що Ви є частиною нашої спільноти пазломанів. На основі Ваших попередніх покупок ми підготували добірку товарів, які можуть Вас зацікавити:

1. **Мультипазли 2 в 1 (2x500) ел. - Тур Америкою: Лілія Бутенко. Озеро Морейн / Ларс Стюарт. Нью-Йорк / MGL / Trefl** Код 37500
2. **Мультипазли 7 в 1 (2x300, 2x500, 2x600, 1000) ел. - Романтичні подорожі / Trefl** Код 93114
3. **13500 ел. - Безмежна колекція. Подорож на тисячу миль / Trefl** Код 81025

Всі ці товари доступні прямо зараз на нашому сайті, і Ви можете легко їх замовити, перейшовши за посиланням: [Посилання на товар].
Також нагадуємо, що у нас діють спеціальні знижки на замовлення від 15 000 грн та більше!
Якщо у Вас виникли запитання або потрібна допомога, наші консультанти завжди готові допомогти за телефоном: **+38 066 666 7778**
Дякуємо, що обираєте нас!
З найкращими побажаннями

Рисунок 3.4 – Електронний лист на основі оброблених персональних даних

Як можемо бачити через те що користувач здійснив покупку пазлів типу мультипазли, на електронну скриньку у процесі обробки зібраних персональних даних прийшов лист із рекомендованими товарами, також типу мультипазли.

Обробка персональних даних є ключовим аспектом сучасних інформаційних систем, що забезпечує ефективне функціонування сервісів, зокрема електронної комерції. Вона дозволяє персоналізувати взаємодію з користувачами, покращувати якість послуг і підтримувати безпеку даних. Використання шифрування, захищених баз даних та належних алгоритмів обробки гарантує дотримання законодавчих вимог і довіру клієнтів.

3.4. Реалізація захисту персональних даних

Реєстрація користувача на сайті реалізується через веб-форму, яка приймає необхідні персональні дані, такі як ім'я, електронна пошта та пароль. Коли користувач заповнює цю форму та натискає кнопку реєстрації, дані передаються на сервер, де відбувається їх обробка. Найважливішим елементом у цьому процесі є забезпечення безпеки, особливо для збереження пароля.

На стороні сервера використовується механізм хешування для паролів перед збереженням у базі даних. Хешування — це процес перетворення введеного пароля у фіксований криптографічний відбиток (хеш), який неможливо повернути до початкового значення. У цьому випадку використовується алгоритм PBKDF2, який забезпечує додатковий рівень захисту завдяки використанню солі та багаторазового повторення процесу хешування.

Під час реєстрації дані з веб-форми надсилаються до сервера, де запускається процедура хешування пароля. Сіль генерується випадковим чином для кожного нового користувача, що гарантує унікальність хешу навіть для однакових паролів. Отриманий результат — комбінація солі та хешу — зберігається у базі даних.

Додатковий код для шифрування паролю:

```
from django.contrib.auth.models import User
from django.shortcuts import render, redirect
from django.contrib import messages

def register(request):
    if request.method == "POST":
    username = request.POST.get("username")
```

```
email = request.POST.get("email")
password = request.POST.get("password")

try:
        user = User.objects.create_user(username=username, email=email, password=password)
user.save()
messages.success(request, "Реєстрація пройшла успішно!")
return redirect("login")
except Exception as e:
        messages.error(request, f"Помилка реєстрації: {str(e)}")

 return render(request, "register.html")
```

На серверній стороні дані обробляються функцією create_user, яка автоматично виконує хешування пароля перед збереженням. Це забезпечує, що паролі ніколи не зберігаються у відкритому вигляді. Алгоритм хешування, який використовується за замовчуванням у Django, базується на PBKDF2 і може бути змінений на інший за потреби

Такий підхід гарантує високий рівень безпеки, оскільки навіть якщо дані будуть скомпрометовані, витягнути початковий пароль неможливо. Ця методика підходить для будь-якого веб-сайту, де необхідно зберігати чутливі персональні дані.

На рисунку 3.5 можемо бачити що відмінностей від рисунку 3.1 – немає. Проте основні відмінності зберігаються “під капотом”. Після проходження реєстрації ми побачимо що пароль цього разу найдійно захищений шифруванням і змоги для зловмисників вкрасти персональні дані користувачів цього сайту практично не буде.

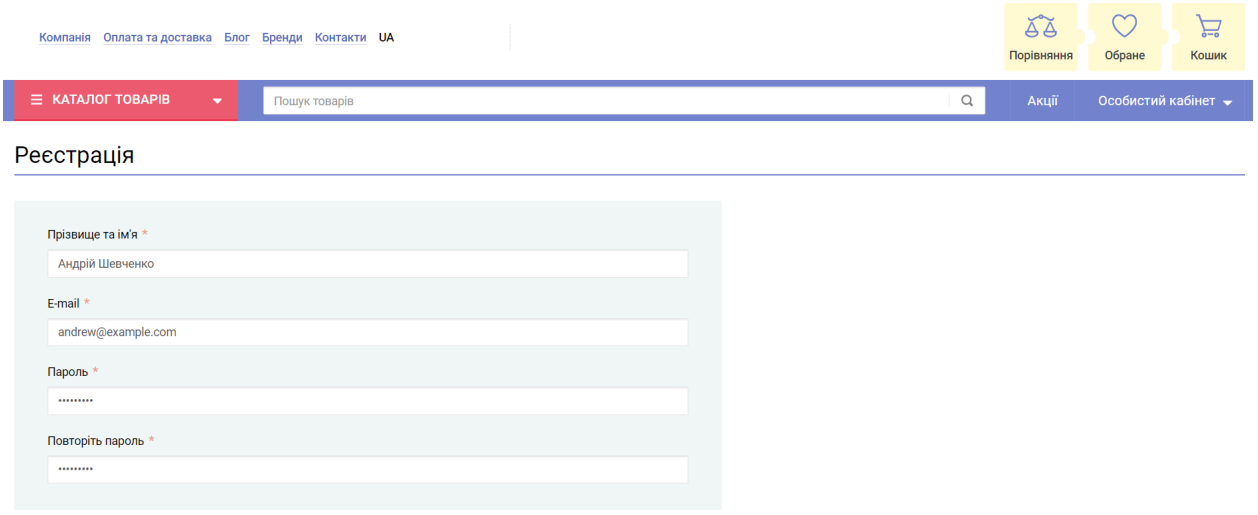


Рисунок 3.5 – Форма реєстрації

На рисунку 3.6 наглядно можемо бачити як виглядає зашифрований пароль у таблиці бази даних. По факту це виглядає як незрозумілий набір символів, хоча насправді це всього лиш зашифрований пароль Qwerty123.

	id [PK] integer	name character varying (100)	email character varying (150)	password character varying (255)	created_at timestamp without time zone
1	1	Андрій Шевченко	andrew@example.com	pbkdf2_sha256$260000...	2024-12-08 21:39:40.75628

Рисунок 3.6 – Таблиця users із зашифрованим паролем

Захист персональних даних є критично важливим аспектом будь-якої системи, яка збирає, обробляє і зберігає інформацію користувачів. Сучасні методи захисту включають використання хешування паролів, шифрування даних, а також застосування безпечних протоколів для передачі інформації.

Вибір правильних технологій для захисту даних, таких як алгоритми хешування або криптографічне шифрування, є основою для забезпечення безпеки та конфіденційності персональних даних. Ключовим аспектом є також дотримання законодавчих норм, зокрема GDPR, для забезпечення прав користувачів і попередження витоків даних.

РОЗДІЛ 4.
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1. Обґрунтування можливих чинників травмонебезпечних ситуацій

Охорона праці в сучасних умовах є невід'ємною складовою ефективного та відповідального управління будь-якою компанією. Створення безпечного робочого середовища сприяє збереженню здоров'я співробітників і підвищенню їхньої продуктивності. Застосування своєчасних заходів з охорони праці дозволяє зменшити витрати, пов'язані з лікуванням працівників або простоєм у роботі через їхню відсутність. З огляду на швидкий розвиток технологій та інноваційні підходи до організації праці, значення охорони праці постійно зростає у всіх сферах діяльності.

В умовах сучасного офісу, зокрема в IT-секторі, забезпечення безпечних і комфортних умов праці є ключовим завданням. Організація робочого простору вимагає комплексного підходу, що включає впровадження інноваційних рішень та стратегій. У процесі роботи офісного персоналу можуть виникати різні травмонебезпечні ситуації, які необхідно враховувати для запобігання ризикам.

Одним із найпоширеніших факторів ризику є тривала сидяча робота. Вона здатна викликати проблеми зі спиною, шиєю, а також інші порушення опорно-рухового апарату. Неправильна організація робочого місця, наприклад, невдале розташування меблів чи техніки, збільшує ймовірність виникнення дискомфорту та травм. Крім того, робота за комп'ютером протягом тривалого часу, що супроводжується статичною позою і повторюваними рухами, може стати причиною перенапруження м'язів, погіршення зору та інших проблем.

Ще одним потенційним джерелом небезпеки є використання несправного або неправильно налаштованого обладнання. Неналежний стан офісних меблів чи техніки може призвести до падінь або інших травм. Особливої уваги потребує електробезпека, адже порушення правил

експлуатації електроприладів чи перевантаження мережі можуть спричинити травми або пожежі.

Запобігти таким ризикам допомагає впровадження комплексних заходів з охорони праці. Проведення регулярного аналізу ризиків дозволяє виявити потенційні проблеми і вчасно їх усунути. Ергономічне облаштування робочого місця сприяє комфортній роботі, зменшуючи ризики травмування. Навчання персоналу правилам безпеки і правильній роботі з обладнанням дозволяє уникнути багатьох проблем. Також важливо регулярно перевіряти технічний стан обладнання, щоб запобігти його несправності.

Підтримання здорового психологічного клімату та запобігання перевантаженню працівників сприяє зменшенню стресу, що позитивно впливає на їхній загальний стан. Захист співробітників від електромагнітного випромінювання також має бути в пріоритеті. Використання засобів захисту дозволяє мінімізувати відповідні ризики.

Крім технічних заходів, важливим є підвищення обізнаності працівників щодо правил надання першої допомоги, щоб у разі надзвичайної ситуації вони могли оперативно діяти. Оскільки кожне робоче середовище має свої особливості, заходи з охорони праці повинні адаптуватися до специфіки компанії. Дотримання законодавчих вимог і стандартів дозволить створити безпечне середовище, яке сприятиме продуктивній і безпечній роботі співробітників.

4.2. Умови та обставини виникнення небезпечних ситуацій та їх наслідки

Небезпечні ситуації є невід’ємною частиною сучасного життя, оскільки вони можуть виникати у будь-який момент і завдавати шкоди як окремим людям, так і суспільству в цілому. Їх причиною можуть бути технічні несправності, природні катаклізми, людські помилки або соціально-економічні проблеми. Такі обставини створюють загрози, які можуть мати серйозні наслідки для здоров’я, життя і матеріального добробуту.

Однією з причин небезпечних ситуацій є технічні збої, зокрема несправності обладнання або аварії на підприємствах. Подібні інциденти часто супроводжуються значними збитками, травмами чи навіть загибеллю людей. Природні катаклізми, як-от землетруси, повені або урагани, також спричиняють значні руйнування інфраструктури та численні жертви. Людський фактор, включаючи недотримання правил безпеки чи помилки персоналу, є ще одним важливим чинником, що сприяє виникненню загроз.

Наслідки небезпечних ситуацій можуть бути різноманітними: від матеріальних втрат і пошкодження інфраструктури до серйозної загрози здоров'ю і життю людей. Такі події також можуть викликати довготривалі наслідки, наприклад, екологічні катастрофи або економічні труднощі.

В умовах офісної роботи ризики мають свої особливості. Наприклад, несправності комп'ютерної техніки або серверів можуть не лише зупинити роботу, але й створити небезпеку для працівників. Проблеми з електрикою, як-от короткі замикання, можуть призводити до пожеж. Також високий рівень стресу через великий обсяг завдань і термінові проєкти може стати причиною помилок, які здатні спричинити небезпечні ситуації.

Щоб уникнути таких проблем, необхідно забезпечити підготовку персоналу до потенційних ризиків, розробляти детальні інструкції та запроваджувати сучасні методи управління ризиками.

4.3. Безпека в надзвичайних ситуаціях

Надзвичайні ситуації порушують звичний порядок роботи й створюють загрози для життя та здоров'я людей. Це можуть бути природні катастрофи, техногенні аварії, епідемії чи воєнні дії. Для зменшення їхнього впливу компанії повинні розробляти чіткі плани дій і забезпечувати готовність персоналу до екстремальних умов.

Ключовим елементом підготовки є створення планів евакуації, навчання персоналу правилам дій під час криз і забезпечення доступу до

засобів безпеки. Особливу увагу слід приділяти організації укриттів у випадку військових дій та облаштуванню офісів базовими засобами першої необхідності.

Забезпечення стабільного функціонування компанії за кризових умов передбачає впровадження віддаленої роботи, захист даних і організацію внутрішньої комунікації. Регулярні тренування, аналіз ризиків і забезпечення технічної готовності дозволяють уникнути значних втрат і зберегти безпеку працівників.

РОЗДІЛ 5.
ВИЗНАЧЕННЯ ЕФЕКТИВНОСТІ

Збір, обробка та використання персональних даних на вебсайтах мають комплексний вплив на різні аспекти бізнесу, включаючи економічну ефективність, продуктивність, строки виконання проектів, лояльність клієнтів і якість сервісів. У кожному з цих напрямків ефективна робота з даними відкриває нові можливості для розвитку бізнесу та вдосконалення взаємодії з користувачами.

Впровадження передових технологій збору та обробки даних значно знижує витрати на управління бізнес-процесами. Наприклад, автоматизація обробки інформації про користувачів допомагає скоротити витрати на ручну працю, прискорюючи процеси аналітики. Це особливо актуально для великих компаній, які працюють з мільйонами користувачів. Завдяки аналізу даних можна також оптимізувати витрати на рекламу, створюючи таргетовані кампанії, які приносять більше конверсій за менших бюджетів.

Продуктивність бізнесу зростає завдяки автоматизації збору даних та використанню сучасних технологій, таких як штучний інтелект і хмарні обчислення. Ці інструменти дозволяють значно швидше обробляти великі обсяги інформації та приймати обґрунтовані бізнес-рішення. В результаті компанії мають змогу швидше реагувати на зміни ринку та потреби клієнтів, що є ключовим фактором для збереження конкурентних переваг.

Збір персональних даних забезпечує точніше планування ресурсів і термінів реалізації проектів. Наприклад, дані про поведінку користувачів можуть бути використані для тестування концепцій продуктів ще на ранніх етапах розробки. Це допомагає ідентифікувати потенційні ризики та оптимізувати процес створення продукту, скорочуючи терміни його запуску.

Крім того, використання автоматизованих рішень для обробки даних значно зменшує час, необхідний для аналізу результатів маркетингових кампаній,

оцінки попиту чи моніторингу ринку. У результаті бізнес отримує можливість впроваджувати нові стратегії та проекти значно швидше.

Персоналізація сервісів, основана на зібраних даних, є одним із ключових факторів, що впливають на лояльність клієнтів. Коли користувач отримує пропозиції, які відповідають його інтересам та потребам, це підвищує його довіру до бренду. Наприклад, рекомендаційні системи на базі аналізу даних, такі як пропозиції товарів у інтернет-магазинах чи персоналізовані повідомлення у мобільних додатках, формують позитивний досвід взаємодії.

Довіра користувачів до компанії також залежить від того, наскільки ефективно вона забезпечує захист їхніх даних. Прозорість у політиках конфіденційності та використання найсучасніших технологій безпеки, таких як шифрування і багатофакторна автентифікація, сприяють формуванню довгострокових відносин з клієнтами.

Дані користувачів дозволяють компаніям не лише вдосконалювати існуючі сервіси, але й створювати нові, які краще відповідають вимогам клієнтів. Наприклад, аналіз частих запитів користувачів допомагає компаніям оптимізувати функціональність своїх продуктів, а відстеження патернів поведінки дозволяє ідентифікувати слабкі місця в обслуговуванні.

Завдяки використанню персональних даних компанії можуть забезпечувати швидшу і точнішу технічну підтримку. Інтеграція зібраної інформації в системи підтримки клієнтів дозволяє швидше розв’язувати проблеми, зменшуючи час очікування відповіді.

Таким чином, робота з персональними даними не тільки підвищує економічну ефективність бізнесу, але й сприяє створенню якісного користувацького досвіду, зміцнює довіру до компанії та забезпечує конкурентоспроможність на сучасному ринку. Однак важливо пам’ятати про необхідність дотримання законодавства, адже тільки відповідальний підхід до захисту даних гарантує стабільний розвиток і лояльність клієнтів.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було здійснено дослідження сучасних технологій збору, обробки та захисту персональних даних користувачів веб-сайтів. Актуальність теми зумовлена стрімким розвитком цифрових технологій, зростанням обсягів даних, що обробляються в інтернеті, а також посиленням вимог до конфіденційності інформації відповідно до міжнародних норм та стандартів.

Робота охопила широкий спектр теоретичних і практичних аспектів: від основних принципів збору персональних даних до технічних та юридичних методів їхнього захисту. Були розглянуті підходи до збирання даних, такі як форми реєстрації, cookie-файли, трекінг користувачів і аналіз поведінки на веб-сайтах. Особливу увагу приділено сучасним інструментам обробки даних, включаючи алгоритми машинного навчання, хмарні технології та інструменти аналітики, які дозволяють отримувати корисну інформацію для бізнесу, не порушуючи прав користувачів.

Одним із головних акцентів стало дослідження механізмів забезпечення безпеки персональних даних. Було детально розглянуто методи шифрування, багатофакторну автентифікацію, безпечне зберігання даних та їхню передачу. Також вивчено відповідність сучасних технологій нормам законодавства, зокрема європейському регламенту GDPR та американському CCPA. Розглянуто ключові проблеми у цій сфері, такі як витоки даних, зловживання інформацією та ризики недотримання правових норм.

Практична частина роботи включала розробку прототипу веб-сайту, який інтегрує всі розглянуті теоретичні аспекти. Цей сайт став демонстраційною платформою, яка показала ефективність поєднання сучасних методів збору та обробки даних з передовими технологіями їхнього захисту. Було впроваджено механізми отримання згоди користувачів на збір даних, надійне шифрування інформації та засоби забезпечення прозорості у

політиці конфіденційності. Крім того, система автоматизованого аналізу даних продемонструвала, як персоналізація сервісів може покращити користувацький досвід, водночас дотримуючись етичних норм.

Результати роботи показали, що впровадження ефективних технологій збору, обробки та захисту персональних даних має ключове значення для сучасних веб-платформ. Це не лише підвищує рівень безпеки та довіри з боку користувачів, але й сприяє економічному розвитку бізнесу, забезпечуючи конкурентні переваги.

Важливо зазначити, що успішна реалізація таких технологій залежить від кількох критичних чинників. По-перше, необхідна повна інтеграція технічних рішень із правовими вимогами, щоб уникнути потенційних ризиків для компаній. По-друге, компанії повинні враховувати етичні аспекти, забезпечуючи прозорість і контроль з боку користувачів над їхніми даними. По-третє, впровадження інноваційних технологій захисту даних повинно йти пліч-о-пліч із розвитком культури кібербезпеки серед розробників і користувачів.

Таким чином, проведене дослідження дозволило не лише систематизувати існуючі знання у сфері роботи з персональними даними, але й створити практичний інструмент, який демонструє ефективність теоретичних підходів на практиці. Отримані результати можуть стати основою для подальшого вдосконалення веб-сервісів, що працюють із персональними даними, та розробки інноваційних рішень у сфері кібербезпеки.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Веб-доступність. Що варто знати кожному Front-end розробнику і дизайнеру. [Електронний ресурс]. Режим доступу: http://gud.org.ua/ (дата звернення 04.07.2024).
2. Загальний регламент про захист даних (GDPR). [Електронний ресурс]. Режим доступу: https://gdpr-info.eu/ (дата звернення 15.07.2024).
3. Закон України "Про захист персональних даних". [Електронний ресурс].
   Режим доступу: https://zakon.rada.gov.ua/laws/show/2297-17#Text/ (дата звернення 28.07.2024).
4. Python для веб-розробки: Як створювати доступні веб-додатки за допомогою Flask та Django. [Електронний ресурс].
   Режим доступу: https://realpython.com (дата звернення 14.08.2024).
5. Zero Trust: багаторівнева модель безпеки для сучасного бізнесу. [Електронний ресурс]. Режим доступу: https://www.bdo.ua/uk-ua/insights-2/information-materials/2024/zero-trust-a-revolutionary-approach-in-modern-cybersecurity (дата звернення 29.08.2024).
6. Кабінет Міністрів України. Постанова № 3, Про Порядок оприлюднення у мережі Інтернет інформації про діяльність органів виконавчої влади. [Електронний ресурс]. Режим доступу: http://comin.kmu.gov.ua/control/uk/publish/article?art_id=110618&cat_id=112508 (дата звернення 15.09.2024).
7. Що таке файли cookies та навіщо вони потрібні. [Електронний ресурс]. Режим доступу: https://ssl.com.ua/blog/ukr/what-are-cookies/ (дата звернення 19.09.2024).
8. Гайдаржи В., Ізварін І. Книга Бази даних в інформаційних системах, 2018р

9. Data Protection and Security Best Practices for Web Applications. [Електронний ресурс]. Режим доступу: https://owasp.org (дата звернення 26.09.2024).
10. Використання HTTPS та SSL/TLS для захисту даних на вебсайтах. [Електронний ресурс]. Режим доступу: https://letsencrypt.org (дата звернення 28.09.2024).
11. Рекомендації з кібербезпеки для веброзробників: Захист даних користувачів. [Електронний ресурс]. Режим доступу: https://cybersecurityguide.org (дата звернення 09.10.2024).
12. Обробка та збереження даних у відповідності до CCPA (California Consumer Privacy Act). [Електронний ресурс]. – Режим доступу: https://oag.ca.gov/privacy/ccpa (дата звернення 18.10.2024).
13. Шифрування даних: Керівництво для веброзробників. [Електронний ресурс]. – Режим доступу: https://www.keycdn.com/blog/data-encryption (дата звернення 30.10.2024).
14. Створення політики конфіденційності для вебсайтів. [Електронний ресурс]. – Режим доступу: https://www.privacypolicies.com (дата звернення 04.11.2024).
15. Використання бібліотек Python для обробки персональних даних. [Електронний ресурс]. – Режим доступу: https://pypi.org/project/cryptography/ (дата звернення 09.11.2024).
16. Практичний підхід до управління базами даних із конфіденційними даними на вебсайтах. [Електронний ресурс]. – Режим доступу: https://www.mysql.com (дата звернення 11.11.2024).
17. Типи баз даних: особливості, відмінності та приклади. [Електронний ресурс]. – Режим доступу: https://dou.ua/lenta/articles/types-of-databases/ (дата звернення 25.11.2024).
18. Wikipedia. Офіційний сайт [Електронний ресурс]. Режим доступу: https://uk.wikipedia.org (дата звернення 27.11.2024).

19. Як відкрити інтернет-магазин: покрокова інструкція[Електронний ресурс]. Режим доступу: https://sendpulse.ua/blog/how-to-start-online-store (дата звернення 28.11.2024).
20. Ковальчук О.І. Інформаційна безпека: правові та технічні аспекти захисту персональних даних, 2019.