Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка вебплатформи для автоматизації обліку фінансових витрат домогосподарством з використанням технологій PHP та MySQL.

Виконав(ла): студент(ка) 4 курсу, групи СН-42
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

(підпис) Паламарчук І.М. (прізвище та ініціали)

Керівник (підпис) Шимчук Г.В. (прізвище та ініціали)

Нормоконтроль (підпис) Липак Г.І. (прізвище та ініціали)

Завідувач кафедри (підпис) Боднарчук І.О. (прізвище та ініціали)

Рецензент (підпис) Коноваленко І.В. (прізвище та ініціали)

Тернопіль
2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
"27" січня 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

студенту Паламарчук Ірина Михайлівна
(прізвище, ім'я, по батькові)

1. Тема роботи **Розробка вебплатформи для автоматизації обліку фінансових витрат домогосподарством з використанням технологій PHP та MySQL**

Керівник роботи **Ст. викл. каф. КН Шимчук Григорій Валерійович**
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від "*07*" *травня* 2025 року № *4/7-444*

2. Термін подання студентом завершеної роботи 26 червня 2025 р.

3. Вихідні дані до роботи Літературні джерела з тематики роботи

4. Зміст роботи (перелік питань, які потрібно розробити)
ВСТУП 1 ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ 1.1 Загальний опис програм для управління бюджетом 1.2 Огляд інструментів для відстеження витрат та управління бюджетом 2 АНАЗІЗ ЗАСОБІВ РОЗРОБКИ ВЕБ-ПЛАТФОРМИ ДЛЯ КЕРВАННЯ БЮДЖЕТОМ 2.1 Вибір системи управління базами даних 2.2 Порівняння систем управління базами даних 2.3 Як вибрати систему керування базами даних 2.4 Порівняння мов програмування для розробки веб-платформи управління фінансами 2.5 Порівняльний аналіз PHP, ASP.NET, Python та Java 2.6 Порівняння MEAN Stack проти PHP проти .NET 3 РОЗРОБКА ВЕБ-ПЛАТФОРМИ 3.1 Проєктування бази даних 3.2 Розробка програмної частини веб-застосунку для планування бюджету 4 ОХОРОНА ПРАЦІ та безпека в надзвичайних ситуаціях;ВИСНОВКИ; СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ; ДОДАТКИ

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Тема 2 Мета і актуальність 3 Огляд існуючих рішень 4 Основні висновки з аналізу існуючих продуктів 5 Порівняння типів СКБД 6 Порівняння СКБД для розробки веб-додатків 7 Вибір мови програмування, 1 /2 8 Вибір мови програмування, 2/2 9 Структура бази даних 10 Вимоги до додатку 11 Тригери таблиць бази даних 12 Інтерфейс входу та реєстрації 13 Головна панель користувача 14 Висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Окіпний І.Б., к.т.н., доцент кафедри МТ		

7. Дата видачі завдання 27 січня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	27.01.2025 – 31.01.2025	*Виконано*
2.	Підбір джерел по темі роботи	01.02.2025 – 01.04.2025	*Виконано*
3.	Оформлення першого розділу	02.04.2025 – 15.04.2025	*Виконано*
4.	Оформлення другого розділу	16.04.2025 –20.04.2025	
5.	Оформлення третього розділу	21.04.2025 – 30.04.2025	*Виконано*
6.	Виконання завдання до підрозділу "Безпека життєдіяльності, основи охорони праці"	31.04.2025 – 05.05.2025	*Виконано*
8.	Оформлення кваліфікаційної роботи	16.05.2025 – 07.06.2025	*Виконано*
9.	Перевірка на плагіат	08.06.2025	*Виконано*
10.	Нормоконтроль	09.06.2025 – 10.06.2025	*Виконано*
11.	Попередній захист кваліфікаційної роботи	11.06.2025	*Виконано*
12.	Захист кваліфікаційної роботи	27.06.2025	

Студент _____ Паламарчук І.М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Шимчук Г.В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

"Розробка вебплатформи для автоматизації обліку фінансових витрат домогосподарством з використанням технологій PHP та MySQL" // Кваліфікаційна робота освітнього рівня "Бакалавр" // Паламарчук Ірина Михайлівна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп’ютерних наук, група СН-42 // Тернопіль, 2025 // с. – 80, рис. – 12, таблиць – 9, джерел – 27, додатків – 2, сторінок додатків – 27.

Ключові слова: фінансовий контроль, домашній бюджет, веб-платформа, PHP, MySQL

Відстеження особистих витрат відіграє ключову роль в ефективному управлінні фінансами. В умовах стрімкого розвитку цифрових технологій велика кількість мобільних та веб-застосунків створює як нові можливості, так і складнощі у виборі оптимального інструменту. Метою цієї статті є виявлення та оцінка найефективніших рішень для обліку витрат, а також окреслення підходів до розробки власних засобів бюджетування. Ринок таких застосунків є надзвичайно широким і охоплює як інструменти для індивідуального користування, так і програми, орієнтовані на потреби малого бізнесу, кожна з яких має свої особливості функціонування та цільову аудиторію.

ANNOTATION

"Development of a Web Platform for Automating Household Financial Expense Tracking Using PHP and MySQL Technologies" // Qualification work of the educational level "Bachelor" // Palamarchuk Iryna // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, Group CH-42 // Ternopil, 2025 // p. – 80, fig. – 12, tables – 9, references – 27, annexes – 2, pages for annexes – 27.

Keywords: financial control, home budget, web platform, PHP, MySQL

Tracking personal expenses plays a critical role in effective financial management. In the context of a rapidly expanding digital ecosystem, the availability of numerous mobile and web-based applications presents both opportunities and challenges for users in selecting the most suitable tool. This work aims to identify and evaluate the most effective expense tracking solutions currently available, while also outlining approaches for the development of custom budgeting tools. The market for such applications is extensive, encompassing tools tailored for both individual users and small businesses, each with specific functionality and target audiences.

ЗМІСТ

ВСТУП

Бюджет планує та відстежує доходи й витрати протягом певного періоду часу. Бізнес та уряди покладаються на бюджети для відстеження доходів та витрат, але на персональному рівні люди, імовірно, найбільше знайомі з бюджетом як інструментом управління своїми фінансами.

Існують різні типи бюджетних систем та методів. Бюджети можуть допомогти відстежувати витрати та жити в рамках своїх можливостей.

Під час складання бюджету слід обрати метод або систему складання бюджету, яка найкраще підходить конкретному користувачеві.

Початок складання бюджету є відносно простим. Основний процес складання бюджету відбувається відповідно до описаних кроків.

Крок 1. Підрахунок сумарного місячного доходу.

Слід враховувати всі можливі джерела доходу: зарплату з роботи, плату від клієнтів, якщо людина фрілансер чи працівник за контрактом, або здійснені продажі у випадку ведення власного бізнесу. Якщо людина отримує регулярні виплати по інвалідності, соціальне забезпечення, аліменти або допомогу на дитину, їх також треба включити.

Таким чином буде складено список всіх джерел доходу та отримано обсяг коштів, які отримує людина на місяць. Важливо оперувати отриманою сумою, а не дохід до сплати податків. Якщо щомісячний дохід відрізняється від інших, то можна оперувати середньомісячним обсягом доходів.

Крок 2: Підсумувати щомісячні витрати.

Необхідно скласти перелік усіх регулярних щомісячних витрат. До переліку слід включити постійні витрати, такі як орендна плата, іпотека або страхові внески. Також потрібно зазначити змінні витрати, що можуть варіюватися з місяця в місяць. Серед них — витрати на харчування (як продукти, так і харчування поза домом), паливо, розваги тощо.

Рекомендується фіксувати всі витрати незалежно від їх розміру. Для цього можна використовувати мобільні додатки, програмне забезпечення для бюджетування або традиційні засоби — ручку й папір. Перевірити банківські та кредитні виписки — корисний спосіб виявити забуті витрати.

Крок 3: Вирахувати витрати з доходу.

Потрібно визначити різницю між загальним щомісячним доходом і сумою всіх щомісячних витрат. За наявності позитивного залишку можна зробити висновок про надлишок коштів після покриття всіх витрат.

У разі виявлення дефіциту доцільно переглянути витрати, щоб виявити можливості для їх скорочення або усунення. Особливу увагу слід приділити розмежуванню між базовими потребами та другорядними бажаннями.

Як показує практика, виконувати всі ці дії вручну – доволі складно й незручно. Постійно записувати витрати, підраховувати залишок бюджету та аналізувати, куди йдуть гроші, потребує багато часу й уваги. Через це легко щось забути або пропустити важливу деталь. Саме тому використання спеціального програмного забезпечення для обліку фінансів є не лише зручним, а й ефективним рішенням. У такій ситуації створення програми, яка допомагає контролювати домашні витрати, виглядає цілком виправдано й актуально, особливо з огляду на потреби сучасного користувача.

1 ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальний опис програм для управління бюджетом

Серед багатьох програм управління власними фінансами та бюджетування сім'ї виділимо наступний список.

1. QuickBooks Accounting чудово підходить для малого бізнесу для надсилання та відстеження рахунків-фактур.

2. Empower пропонує інвестиційні інструменти та консультації, що робить його ідеальним для інвесторів.

3. Expensify автоматично зчитує та імпортує дані чеків, що ідеально підходить для їх збереження.

4. Everlance допомагає вам створити журнал пробігу, що відповідає вимогам IRS, та відстежувати витрати для відшкодування.

5. NerdWallet – це безкоштовний варіант, який допомагає вам контролювати свій кредитний рейтинг та витрати.

Розглянемо коротко їх характеристики, можливості використання та функціонал.

1.2 Огляд інструментів для відстеження витрат та управління бюджетом

1.2.1 QuickBooks Accounting

QuickBooks Accounting – це фантастичний інструмент для управління бюджетом, незалежно від того, чи ви власник малого бізнесу, чи просто хтось, хто хоче відстежувати особисті витрати. Одна з найкращих переваг QuickBooks – це його універсальність. Можна надсилати та відстежувати рахунки-фактури, а також використовувати трекер пробігу. Це робить його найкращим вибором для багатьох користувачів.

QuickBooks пропонує безкоштовну 30-денну пробну версію без необхідності використання кредитної картки. Після цього можна вибрати один із різних планів від 30 доларів на місяць зі знижкою 50% протягом перших трьох місяців. Якщо вже є придбаний QuickBooks онлайн, додаток для бухгалтерського обліку безкоштовний.

Деякі популярні функції включають:

– Надсилання та відстеження рахунків-фактур.
– Відстеження пробігу авто.
– Відстеження витрат.

QuickBooks сумісний з пристроями iOS та Android. Тарифний план для самозайнятих коштує від 7,50 доларів на місяць протягом перших трьох місяців, а потім – від 15 доларів на місяць. Цей план ідеально підходить для фрілансерів і включає такі функції, як технологія сканування та шифрування Verisign для додаткової безпеки.

1.2.2 Empower

Empower – це фантастичний інструмент для тих, хто хоче стежити за всіма своїми фінансами. Він безкоштовний у використанні та має середній рейтинг програми 4,4. Можна легко синхронізувати свої облікові записи, що спрощує відстеження витрат та інвестицій в одному місці.

Компанія Empower розпочала свою діяльність як страхова компанія Great-West Life понад століття тому у Вінніпезі, Манітоба. Зараз вона пропонує широкий спектр послуг, включаючи безкоштовні інструменти для планування виходу на пенсію та складання бюджету. Крім того, користувач отримує цілодобову технічну підтримку та доступ до команди консультантів.

Empower надає повне уявлення про фінанси, від щоденних витрат до стану інвестицій. Він навіть має функції длядеск-топу для тих, хто любить складати бюджет на комп’ютері.

Одним недоліком є те, що потрібно щонайменше 100 000 доларів інвестиційних заощаджень, щоб скористатися його послугами.

Empower ідеально підходить для тих, хто хоче краще керувати своїми грошима, особливо якщо користувач хоче відстежувати як свої витрати, так і інвестиції.

1.2.3 Expensify

Expensify – це чудовий інструмент для управління витратами, особливо для тих, хто займається підприємництвом та має підробітки. Цей додаток, доступний як на Android, так і на iOS, ідеально підходить для тих, хто часто подорожує у справах. Однією з його видатних функцій є можливість автоматичного сканування, зчитування та імпорту даних чеків. Можна фотографувати свої чеки, і додаток реєструватиме витрати сам. Крім того, можна класифікувати свої витрати за групами, такими як на авто, подорожі та харчування.

Основні характеристики програми.

Технологія SmartScan: Expensify згенерує звіт про витрати на основі фото чека.

Відшкодування наступного дня: після схвалення адміністратором компанії користувачі можуть отримати оплату вже наступного дня.

Корпоративні інструменти: можна надсилати квитанції безпосередньо менеджеру або бухгалтеру для затвердження.

Інтеграція: працює з бухгалтерськими програмами, такими як QuickBooks, Xero, NETSuite та Sage Intacct.

Ціноутворення.

1. Безкоштовно до 25 сканувань чеків SmartScan на місяць.
2. 5 доларів США на місяць для фізичних осіб за необмежену кількість SmartScans.
3. 9 доларів США за користувача на місяць для компаній.

Expensify – це обов’язковий додаток для кожного, хто хоче опанувати навички управління фінансами та досягти фінансової стабільності. Він

допомагає легко створювати бюджети, контролювати витрати та ставити фінансові цілі.

1.2.4 Everlance

Everlance – це чудовий інструмент для відстеження витрат на авто, інших витрат та квитанцій. Заснований у 2015 році, цей додаток використовує технологію GPS для автоматичного відстеження ваших поїздок. Однак, також можна вимкнути цю функцію та відстежувати поїздки вручну при бажанні. Хоча додаток рекламує підхід «налаштував і забув», відгуки користувачів радять перевірити, чи записана поїздка, оскільки помилки GPS іноді можуть призвести до того, що додаток пропустить деякі поїздки.

Everlance пропонує три пакети послуг:

1. Безкоштовно: Відстежуйте до 30 поїздок на місяць.
2. Преміум: 10 доларів на місяць, включає необмежене відстеження поїздок та розширені функції.
3. Підприємство: Індивідуальне ціноутворення для бізнесу з додатковими функціями та підтримкою.

Щоб відстежувати витрати, можна синхронізувати свої кредитні картки та банківські рахунки. Одним рухом пальця можна класифікувати ділові та особисті витрати. Для регулярних витрат і транзакцій користувачі можуть встановлювати власні правила витрат. Відстежувати доходи також легко, навіть якщо у є дохід з кількох джерел.

Everlance доступний як на платформах Android, так і на iOS, що робить його доступним для більшості користувачів. Додаток також містить сканер чеків, що є зручною функцією для зберігання всіх витрат в одному місці.

1.2.5 NerdWallet

NerdWallet – це хороший, повністю безкоштовний додаток, який допомагає керувати своїми фінансами. Він дозволяє легко імпортувати

банківські рахунки та відстежувати транзакції. Одна з його найкращих функцій – це детальна інформація про кредитний рейтинг. Можна також знайти велику кількість статей та навчальних посібників з питань особистих фінансів.

Додаток частково підтримується рекламою, але вона не нав'язлива. NerdWallet також допомагає стежити за статками та грошовими потоками, підключаючись до всіх онлайн-фінансових рахунків. Це чудовий вибір для тих, хто хоче керувати своїми грошима та розуміти свій кредитний рейтинг, не стикаючись із завалом рекламою.

NerdWallet ідеально підходить для тих, хто хоче відстежувати свій фінансовий прогрес і отримувати безкоштовне управління фінансовим рахунком і інформацію про кредитний рейтинг.

1.2.6 Mint

Mint – це інструмент для всіх, хто хоче стежити за своїми фінансами. Mint – це безкоштовний додаток для особистих фінансів, який допомагає користувачам відстежувати свої витрати та дотримуватися бюджетних цілей. Він доступний як на iOS, так і на Android, що робить його доступним для широкого кола користувачів. Mint підтримує різноманітні банки та кредитори, тому можна легко пов’язати свої облікові записи та отримати повну картину свого фінансового стану.

Основні характеристики.

1. Відстеження витрат.
2. Бюджетування.
3. Відстеження рахунків.
4. Моніторинг кредитної історії.

Ціноутворення.

Більшість послуг Mint безкоштовні, але якщо потрібно скористатися преміум-моніторингом кредитної історії, це коштуватиме 16,99 доларів на місяць.

Mint – чудовий варіант для мікробізнесу або підробітку, пропонуючи низку інструментів, які допоможуть ефективно керувати вашими фінансами.

1.2.7 YNAB (You Need A Budget)

YNAB, скорочення від You Need A Budget (Вам потрібен бюджет), ідеально підходить для тих, хто любить аналітику та складання бюджету з нульовою базою. Це найкращий вибір для серйозних бюджетників, які хочуть опанувати свої фінансові навички. Додаток допомагає розставляти пріоритети у фінансових цілях, відстежувати доходи та витрати, а також розрізняти потреби та бажання.

Вартість – 14,99 доларів США на місяць; 99 доларів на рік; 34-денна безкоштовна пробна версія.

Рейтинги: Android: 4.7/5 (на основі понад 18 тис. відгуків); iOS: 4.8/5 (на основі понад 50 тис. відгуків).

Основні характеристики.

1. Бюджетування з нульової бази.
2. Можливості синхронізації облікових записів.
3. Доступно з настільних комп’ютерів, мобільних пристроїв та пристроїв з голосовою активацією.

З YNAB можна встановити реалістичні ліміти, ефективно заощаджувати та досягти фінансової безпеки.

1.2.8 PocketGuard

PocketGuard – це інструмент для тих, хто хоче пильно стежити за своїми витратами. Цей додаток допоможе попрощатися з перевитратами та привітати достаток за допомогою простих рішень для складання бюджету. Функція «У моїй кишені» показує, скільки грошей у вас залишилося для заощадження або витрат після покриття рахунків, бюджетів та цілей. Щоб скористатися цією функцією, потрібно буде підключити свій банківський та кредитний рахунки.

Основні характеристики.

1. Інтегрується з тисячами банків.
2. Оновлення витрат у режимі реального часу.
3. Створення власного бюджету.
4. Встановлення цілей заощаджень.

Вартість.

PocketGuard пропонує безкоштовну версію, але деякі функції доступні лише платним користувачам. Версія Plus коштує 12,99 доларів на місяць або 74,99 доларів на рік, що зводиться до 6,25 доларів на місяць, що дає економію 50%.

Оцінки користувачів.

iOS: 4.6/5 на основі понад 7 тис. відгуків.

Android: 3.6/5 на основі понад 2 тис. відгуків.

Якщо метою є уникнути перевитрат і досягти фінансового спокою, PocketGuard — чудовий вибір. Додаток навіть сповіщає користувача, коли він близькі до перевищення лімітів витрат або бюджету.

Хоча додаток зручний у використанні, деякі функції в безкоштовній версії обмежені, а навігація веб-сайтом може бути дещо складною. Однак переваги значно переважують ці незначні недоліки.

1.2.9 Goodbudget

Goodbudget ідеально підходить для тих, хто любить метод бюджетування в конвертах, але хоче цифрового підходу. Він чудово підходить для початківців, які тільки починають керувати своїми фінансами. Додаток використовує цифрові конверти, щоб допомогти розділити свої гроші на різні категорії витрат. Таким чином, можна легко побачити, куди йдуть гроші та скільки залишилося в кожній категорії.

Goodbudget пропонує як безкоштовну, так і преміум-версії. Преміум-версія коштує 10 доларів на місяць або 80 доларів на рік. Додаток має солідний

рейтинг, із середнім показником 4,35 зірки. Він також підтримує синхронізацію облікових записів, що спрощує відстеження витрат.

Якщо потрібен простий, але ефективний спосіб управління своїм бюджетом, Goodbudget – це чудовий вибір. Система цифрових конвертів дозволяє легко бачити звички витрачання та допомагає заощаджувати гроші.

1.2.10 Wally

Wally – це додаток із безліччю функцій для відстеження витрат. Можна використовувати Wally на пристрої iOS, а базова версія додатка безкоштовна. Як і в інших додатках для відстеження витрат, можна фотографувати чеки безпосередньо у Wally або вводити витрати вручну та відповідно класифікувати їх. Wally також відстежує дохід і пропонує прогнозовані заощадження на кожен місяць. Зосереджуючись на особистих, а не на бізнес-фінансах, Wally добре підходить для людей, які керують мікробізнесом або мають підробіток.

Ціноутворення: Базова версія безкоштовна; повна преміум-версія (Wally Gold) коштує 4,49 долара на місяць.

Таким чином керування бюджетом не повинно бути головним болем. За допомогою правильних інструментів відстеження витрат можна легко контролювати свої витрати та приймати розумніші фінансові рішення. Незалежно від того, чи користувач – власник малого бізнесу, інвестор, чи просто хтось, хто намагається зберігати чеки, для нього знайдеться додаток. Від QuickBooks для бізнесу до NerdWallet (безкоштовна версія) – ці інструменти допоможуть контролювати свої фінанси. Перейдемо тепер до аналізу засобів розробки власного веб-додатку для управління витратами.

2 АНАЗІЗ ЗАСОБІВ РОЗРОБКИ ВЕБ-ПЛАТФОРМИ ДЛЯ КЕРВАННЯ БЮДЖЕТОМ

2.1 Вибір системи управління базами даних

У світі розробки програмного забезпечення вибір правильної бази даних є вирішальним рішенням, яке може суттєво вплинути на продуктивність, масштабованість та зручність використання вашої програми. З огляду на велику кількість доступних варіантів, може бути складно визначити найкращу систему керування базами даних (СКБД), яка ідеально відповідатиме вашим потребам.

У цьому розділі ми порівняємо 12 найпоширеніших СКБД: MySQL, MariaDB, Oracle, PostgreSQL, MSSQL, SQLite, MongoDB, Redis, Cassandra, Elasticsearch, Firebase та DynamoDB. Ми зосередимося на їхніх бізнес-перевагах та проблемах, виділивши ідеальні варіанти використання для кожної з них. Маючи це порівняння баз даних, ви зможете прийняти обґрунтоване рішення для свого проєкту.

Система керування базами даних (СКБД) – це спеціалізоване програмне забезпечення, призначене для зберігання, пошуку та маніпулювання даними. Вона діє як посередник між базою даних, програмами та інтерфейсами користувача для ефективного керування та організації даних. Система надає комплексний набір інструментів для керування базами даних, забезпечуючи безпеку, узгодженість та цілісність даних.

СКБД підтримує різні програми, від простих завдань зберігання та пошуку до складних систем, керованих даними, шляхом впровадження ефективних методів доступу до даних та управління ними. Крім того, система може обробляти одночасних користувачів, підтримувати узгодженість транзакцій та забезпечувати надійні варіанти резервного копіювання та відновлення, що робить її важливим компонентом будь-якого середовища, орієнтованого на дані.

Таблиця 2.1 – Реляційні та нереляційні бази даних

Характеристика	Реляційні БД (SQL)	Нереляційні БД (NoSQL)
Структура даних	Таблиці з рядками і стовпцями. Жорстка схема. Дані у вигляді записів і атрибутів.	Документна, ключ-значення, графова, колонкова модель. Не має фіксованої схеми. Підтримка неструктурованих даних.
Мова	SQL – структурована мова запитів.	Різні мови запитів залежно від даних моделі.
Масштабованість	Вертикальна (додавання потужності одному серверу). Горизонтальна складніша.	Горизонтальна масштабованість. Розподіл завантаження між серверами.
Продуктивність	Висока продуктивність для невеликих і середніх обсягів. Зниження ефективності при рості даних і запитів.	Добре працює в розподілених системах. Підтримка великої кількості одночасних користувачів. Необмежений обсяг неструктурованих даних.
Безпека	Вбудована структура забезпечує кращий захист. Підтримка ACID для цільності даних.	Часто обмежене дотримання ACID (для однієї партиції). Менше вбудованих механізмів безпеки, хоча деякі системи мають покращену безпеку.
Сфера застосування	Складні ПЗ-рішення, фінансові системи, eCommerce.	Зберігання неструктурованих даних, стартапи, гнучка розробка MVP.

Існує два типи СКБД: реляційні та нереляційні, також відомі як SQL та NoSQL відповідно. Перш ніж обговорювати найпопулярніші варіанти баз даних, давайте детальніше розглянемо, чим відрізняються реляційні та нереляційні системи баз даних, враховуючи зазвичай використовувані структури даних, продуктивність, масштабованість та безпеку. Коротке порівняння показано в таблиці 2.1.

2.1.2 Реляційні СКБД

Система керування реляційними базами даних (РСБД) – це сховище інформації, яке впорядковує дані в таблиці, що складаються з рядків (записів) та стовпців (атрибутів, що містять властивості цих записів). Кожна таблиця представляє відношення, а рядки (також звані кортежами) містять окремі записи

в межах цього відношення. РСБД мають заздалегідь визначену схему зі суворою структурою та чіткими залежностями між різними точками даних.

Отже, таблиці в реляційних базах даних пов'язані з іншими таблицями за допомогою зв'язків первинного ключа або зовнішнього ключа. Первинний ключ – це унікальний ідентифікатор для кожного запису в таблиці, який гарантує, що жодні два записи не мають однакового значення для цього конкретного стовпця або набору стовпців. З іншого боку, зовнішній ключ – це стовпець або набір стовпців в одній таблиці, який посилається на первинний ключ в іншій таблиці, встановлюючи зв'язок між ними.

Незважаючи на ці зв'язки між таблицями, термін «реляційний» у реляційних системах баз даних походить від математичної концепції відношень. Доктор Едгар Ф. Кодд запропонував цю ідею як новий спосіб організації та управління даними з використанням принципів математики у своїй фундаментальній статті [4].

Друга назва таких систем – бази даних SQL. Це пояснюється тим, що для зв'язку з цими базами даних та управління ними використовується структурована мова запитів (SQL).

Масштабованість. Реляційні бази даних зазвичай масштабуються вертикально, тобто дані зберігаються на одному сервері, а масштабування здійснюється шляхом додавання більшої обчислювальної потужності (процесор, графічний процесор та оперативна пам'ять) до цього сервера. Однак перехід від менших до більших машин часто пов'язаний з простоями. Масштабування бази даних SQL між кількома серверами (горизонтальне масштабування) може бути складним, оскільки вимагає змін у структурі даних та додаткових інженерних зусиль.

Продуктивність. Реляційні бази даних добре справляються з інтенсивними операціями читання/запису на малих та середніх наборах даних. Вони також пропонують підвищену швидкість пошуку даних, додаючи індекси до полів даних для запитів та об'єднання таблиць. Однак продуктивність може постраждати, коли обсяг даних та запитів користувачів зростає.

Безпека. Завдяки інтегрованій структурі та системі зберігання даних, бази даних SQL не потребують значних інженерних зусиль для забезпечення належного захисту. Вони є гарним вибором для створення та підтримки складних програмних рішень, де будь-яка взаємодія має низку наслідків. Однією з основ SQL є відповідність ACID (атомарність, узгодженість, ізоляція, довговічність). Відповідність ACID є кращим варіантом, якщо ви створюєте, наприклад, додатки для електронної комерції або фінансових програм, де цілісність бази даних є критично важливою.

Оскільки бізнес-додатки найчастіше пишуться програмно-орієнтованими мовами, розробникам необхідно синхронізувати реляційні бази даних та об'єкти в коді. Це робиться за допомогою ORM (об'єктно-реляційного відображення).

2.1.3 Нереляційні або NoSQL бази даних

Нетаблична або нереляційна база даних використовує різні моделі даних для зберігання, керування та доступу до даних. Найпоширенішими моделями даних є:

- документоорієнтований – для зберігання, отримання та керування даними, такими як документи JSON;
- ключ-значення – для представлення даних як колекції пар ключ-значення, де ключі – це унікальні рядки з відповідними значеннями даних;
- граф – для зберігання даних у структурі вузол-ребро-вузол, де вузли – це точки даних, а ребра – їх зв'язки;
- стовпці – для зберігання даних у табличному форматі з гнучкими стовпцями, тобто вони можуть змінюватися від рядка до рядка в одній таблиці.

Оскільки ці бази даних не обмежуються табличною структурою, їх називають NoSQL. Вони дозволяють зберігати неструктуровані дані, такі як тексти, фотографії, відео, PDF-файли та безліч інших форматів. Дані легко запитувати, але вони не завжди класифікуються за рядками та стовпцями, як у реляційній базі даних.

Масштабованість. Коли кількість даних і запитів збільшується, нереляційні або NoSQL бази даних зазвичай масштабуються горизонтально шляхом додавання більшої кількості серверів до пулу. Вони обмінюються даними між різними серверами, кожен з яких містить лише частину даних, що зменшує кількість запитів за секунду на кожному сервері.

Продуктивність. Нереляційні бази даних відомі своєю високою продуктивністю: вони мають розподілену структуру, що знижує навантаження на систему та забезпечує одночасний доступ великій кількості користувачів. Такі бази даних можуть зберігати необмежену кількість наборів даних усіх типів і форм. Вони також досить гнучкі, коли справа доходить до зміни типів даних.

Безпека. На відміну від реляційних систем, бази даних NoSQL мають слабкий захист, що робить їх серйозною проблемою для багатьох інфраструктур. Хоча вони можуть надавати гарантії ACID, вони зазвичай доступні в межах одного розділу бази даних. Однак деякі СКБД пропонують розширені функції безпеки, які відповідають суворим стандартам безпеки та відповідності.

Оскільки бази даних NoSQL дозволяють резервувати різні типи даних разом та масштабувати їх на кількох серверах, їхня невпинна популярність зрозуміла. Крім того, бази даних NoSQL можуть бути дуже вигідними, коли справа доходить до створення MVP. Вони не потребують попередньої підготовки перед розгортанням, що спрощує швидке та беззатримкове оновлення структури даних.

Розглянемо це детальніше, які найпоширеніші системи баз даних у SQL та NoSQL, які їхні основні переваги та недоліки, і як їх слід використовувати підприємствам.

Нижче ми обговоримо наступний список баз даних SQL:

- MySQL.
- MariaDB.
- Oracle.
- PostgreSQL.
- MSSQL.

- SQLite.

Та доповнимо його такими NoSQL базами даних, як:

- MongoDB.
- Redis.
- Cassandra.
- Elasticsearch.
- Firebase.
- Amazon DynamoDB.

Знімок екрана нижче відображає популярність цих та кількох інших баз даних [5].

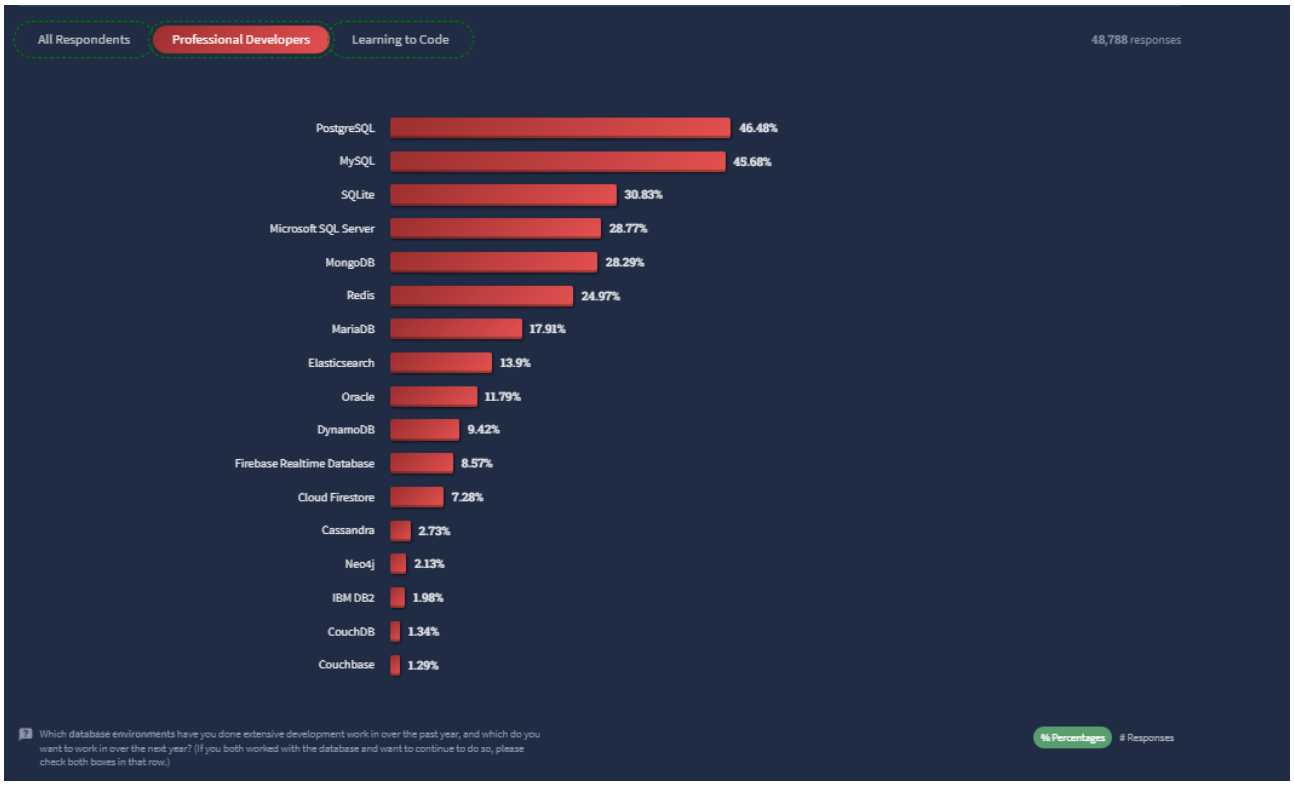

Рисунок 2.1 – Найпопулярніші системи баз даних

Таблиця 2.2 – Порівняння СКБД за ключовими критеріями

СКБД	Тип бази дані	Ліцензу-вання	Масштабо-ваність	Підтримка типів даних	Легка в навчанні
MySQL	SQL	Загальна публічна ліцензія GNU	Вертикальна, складна	Структуровані, напівструктуро-вані	Легка

Продовження таблиці 2.2

СКБД	Тип бази дані	Ліцензу-вання	Масштабо-ваність	Підтримка типів даних	Легка в навчанні
MariaDB	SQL	Загальна публічна ліцензія GNU	Вертикальна	Структуровані, напівструктуро-вані	Легка
Oracle	Мульти-модельна, SQL	Власницька	Вертикальна і горизонтальна	Структуровані, напівструктуро-вані, неструктуровані	Важка
PostgreSQL	Об'єктно-реляційна, SQL	Відкрите ПЗ	Вертикальна	Структуровані, напівструктуро-вані, неструктуровані	Важка
MSSQL	T-SQL	Власницька	Вертикальна, складна	Структуровані, напівструктуро-вані, неструктуровані	Важка
SQLite	SQL	Суспільне надбання	Вертикальна	Структуровані, напівструктуро-вані	Легка
MongoDB	NoSQL, документо-орієнтована	ССПЛ	Горизонтальна	Структуровані, напівструктуровані, неструктуровані	Легка
Redis	NoSQL, ключ-значення	BSD 3-клаузула (з відкритим вихідним кодом)	Горизонтальна	Структуровані, напівструктуро-вані, неструктуровані	Легка
Cassandra	NoSQL, широка колонка	Відкрите ПЗ	Горизонтальна	Структуровані, напівструктуро-вані, неструктуровані	Важка
Elasticsearch	NoSQL, документо-орієнтована	Відкрите ПЗ	Горизонтальна	Структуровані, напівструктуро-вані, неструктуровані	Важка
Firebase	NoSQL, реального часу	Відкрите ПЗ	Горизонтальна	Структуровані, напівструктуро-вані, неструктуровані	Легка
Amazon DynamoDB	NoSQL, ключ-значення	Власницька	Горизонтальна	Структуровані, напівструктуро-вані, неструктуровані	Легка

Хоча детальніші описи вищезгаданих баз даних будуть надані далі в цьому розділі, таблиця 2.2 надає швидке порівняння за ключовими критеріями.

2.2 Порівняння систем управління базами даних

Тепер, коли маємо загальне розуміння відмінностей між реляційними та нереляційними базами даних, ми переходимо до опису основних сучасних систем управління базами даних, а також переваг, недоліків та варіантів використання кожної з них.

2.2.1 MySQL

MySQL – одна з найпопулярніших реляційних систем баз даних. Спочатку це було рішення з відкритим вихідним кодом, але зараз MySQL належить корпорації Oracle. Сьогодні MySQL є основою прикладного програмного забезпечення LAMP. Це означає, що він є частиною стеку Linux, Apache, MySQL та Perl/PHP/Python. Маючи в своєму складі C та C++, MySQL добре працює з такими системними платформами, як Windows, Linux, MacOS, IRIX та іншими.

Плюси MySQL.

Безкоштовне встановлення. Версія MySQL Community можна завантажити безкоштовно. З базовим набором інструментів для індивідуального використання, версія MySQL Community є гарним варіантом для початку. Звичайно, існують інші, передплачені версії для підприємств або кластерів з багатшим функціоналом. Тим не менш, якщо компанія занадто мала, щоб платити за одну з них, модель з безкоштовним завантаженням є найкращим варіантом для початку.

Простий синтаксис та незначна складність. Структура та стиль MySQL дуже прості. Розробники навіть вважають MySQL базою даних з мовою, подібною до людської. MySQL часто використовується разом з мовою програмування PHP. Оскільки вони мають спільну легку криву навчання, набагато легше сформувати команду для управління базою даних. Крім того, MySQL простий у використанні. Наприклад, більшість завдань можна виконувати безпосередньо в командному рядку, що скорочує кількість кроків розробки.

Сумісність з хмарою. MySQL, орієнтований на бізнес за своєю природою та спочатку розроблений для Інтернету, підтримується найпопулярнішими хмарними провайдерами. Він доступний на провідних платформах, таких як Amazon, Microsoft та інших. Це робить MySQL ще більш привабливим і дає бізнесу простір для зростання.

Мінуси MySQL.

Проблеми масштабованості. MySQL не був створений з урахуванням масштабованості, яка є невід'ємною частиною його коду. Теоретично, можна масштабувати MySQL, але це вимагатиме більше інженерних зусиль, ніж будь-яка з NoSQL баз даних. Тому, якщо потрібно, щоби база даних суттєво збільшилася, слід пам'ятати про це обмеження або обрати інший варіант СКБД.

Часткова підтримка відкритого коду. Хоча MySQL має частину з відкритим кодом, вона здебільшого знаходиться під ліцензією Oracle. Це обмежує спільноту MySQL у плані покращення СКБД.

Обмежена відповідність стандартам SQL. Структурована мова запитів має певні стандарти. MySQL не повністю їм відповідає, тобто MySQL не забезпечує підтримку деяких стандартних функцій SQL. З іншого боку, MySQL має деякі розширення та окремі функції, які не відповідають стандартам SQL. Це не є великою проблемою для невеликих веб-застосунків. Проблеми можуть виникнути, коли доведеться перейти на інші бази даних, що, ймовірно, станеться, коли бізнес почне зростати.

Варіанти використання.

Невеликі веб-рішення. Система баз даних MySQL – найкращий варіант, для розробки невеликого веб-рішення з невеликим обсягом даних. Наприклад, під час створення локального магазину електронної комерції MySQL може стати в нагоді.

Системи OLAP/OLTP. Це один із найкращих варіантів використання бази даних MySQL, оскільки OLAP/OLTP не вимагають складних запитів та великих обсягів даних. Також розгляньте можливість застосування MySQL з тієї ж причини, якщо ви створюєте інструмент бізнес-аналітики.

Програми Інтернету речей. MySQL може використовуватися для малих та середніх програм Інтернету речей (IoT) для зберігання та керування даними датчиків, інформацією про пристрої та взаємодією з користувачами.

Хоча MySQL підтримує ці випадки використання, його продуктивність та придатність можуть відрізнятися залежно від конкретних вимог та розміру проекту.

2.2.2 MariaDB

MariaDB, форк MySQL з відкритим кодом, є чудовим прикладом бази даних SQL з комерційною підтримкою. Вона працює під ліцензією GNU General Public License та має схожі команди, API та бібліотеки з MySQL.

Плюси MariaDB.

Шифрування. Для MariaDB відкритий вихідний код не означає незахищеність. Окрім внутрішньої безпеки та перевірки паролів, MariaDB надає такі функції, як автентифікація PAM та LDAP, Kerberos та ролі користувачів. У поєднанні із зашифрованими табличними просторами, таблицями та журналами це створює надійний захисний шар для даних. Крім того, MariaDB публікує відповідні випуски для кожного оновлення безпеки, забезпечуючи повну прозорість патчів безпеки.

Широкий функціонал. За останні кілька років MariaDB представила багато нових функцій. Наприклад, підтримка ГІС пропонує плавне зберігання координат та запити даних про місцезнаходження. Динамічні стовпці дозволяють одній СКБД забезпечувати обробку даних як SQL, так і NoSQL для різних потреб. Також можна розширити її функціональність за допомогою плагінів, доступних для MySQL лише від сторонніх розробників. MariaDB постачається з механізмами зберігання для NoSQL-бекенду, інструментами міграції застарілих баз даних, опціями шардування та багатьма іншими.

Висока продуктивність. Хоча MariaDB походить від рушія MySQL, вона значно просунулася з точки зору продуктивності. Розширені функції оптимізації покращують управління пулом потоків та обробку даних. Таким чином, коли

рядки з таблиці видаляються, операційна система негайно отримує доступ до вільного простору, усуваючи прогалини в табличному просторі. Крім того, система керування базами даних пропонує незалежну від рушія табличну статистику. Ця функція підвищує продуктивність оптимізатора, прискорює обробку запитів та допомагає налаштувати аналіз даних.

Мінуси MariaDB.

Хоча MariaDB має значний внесок у розробку програмного забезпечення з відкритим кодом, її спільнота ще не сильно зросла. Оскільки ця система управління базами даних була створена не так давно, кількість залучених фахівців є відносно невеликою.

Розбіжності між оновленнями MySQL та MariaDB. Хоча команда MariaDB постійно об'єднує свій код з кодом MySQL, вже зараз не так просто підтримувати їх узгодженість. Враховуючи існуючі відмінності між MariaDB 10.6 та MySQL 8.0.32, подальші відхилення ще не з'являться. Крім того, інженери MySQL додали до коду деякі вбудовані функції, доступні лише комерційним користувачам MySQL. Це може створювати проблеми сумісності або проблеми з міграцією даних з MariaDB назад до MySQL.

Варіанти використання.

Оскільки MariaDB близька до MySQL, її можна використовувати для роботи з тими ж типами веб-додатків. Крім того, доступне розширене сховище даних про місцезнаходження, вища продуктивність та покращена масштабованість.

2.2.3 Oracle

Oracle – це реляційна система керування базами даних, створена та керована корпорацією Oracle. Серед усіх типів баз даних SQL Oracle виділяється. Наразі вона підтримує кілька моделей даних, таких як документи, графіки, реляційні та ключ-значення, в межах однієї бази даних. У своїх останніх випусках вона переорієнтована на хмарні обчислення. Ліцензування рушія баз даних Oracle є повністю власним, доступні як безкоштовні, так і платні варіанти.

Плюси Oracle.

Інновації для щоденного робочого процесу. Починаючи з випуску Oracle 12c, коли програмне забезпечення вступило в еру гібридної хмари, нові технології хмарних обчислень з'являлися регулярно. З кожним новим випуском Oracle намагається йти в ногу з темпами інновацій, зосереджуючись на інформаційній безпеці, включаючи активний захист даних, розділення, покращене резервне копіювання та відновлення.

Потужна технічна підтримка та документація. Oracle забезпечує гідну підтримку клієнтів та надає вичерпну технічну документацію з різних ресурсів. Тож кожен, ймовірно, знайде рішення будь-яких проблем, що виникають. Також можна розраховувати на певну підтримку спільноти.

Велика ємність. Багатомодельне рішення Oracle дозволяє розміщувати та обробляти величезні обсяги даних. Завдяки нещодавно випущеній функції багатокористувацького обслуговування, архітектура бази даних тепер спрощує пакування багатьох баз даних та плавно керує ними. У поєднанні з можливостями обробки даних в пам'яті це створює потужний механізм для синхронної обробки даних.

Мінуси Oracle.

Висока вартість. Хоча база даних Oracle має безкоштовні версії, вони дуже обмежені з точки зору функціональності. Стандартна версія, яка не включає всі доступні функції, коштує 17 500 доларів США за одиницю. Корпоративна версія коштує понад 47 000 доларів США за одиницю.

Ресурсоємна технологія. База даних Oracle потребує потужної інфраструктури. Інсталяція не лише вимагає багато місця на диску, але й потребує постійних оновлень обладнання, якщо розгортати її локально.

Складна крива навчання. База даних Oracle – це не та система, яку варто починати використовувати одразу. Краще мати сертифікованих інженерів Oracle DB для її роботи. Документація Oracle, хоча й охоплює багато питань, іноді може бути складною та навіть заплутаною. Тому, щоб встановити та запустити базу даних Oracle, доведеться розглянути можливість найму спеціалістів.

Варіанти використання.

Масштабні корпоративні додатки. Враховуючи всі ці переваги та недоліки, можна розглядати Oracle RDBMS як розумне рішення для онлайн-OLTP, сховищ даних і навіть змішаних (OLTP та DW) додатків баз даних. Якщо є мільярд записів для зберігання та управління – і достатній бюджет для його підтримки – гібридне хмарне програмне забезпечення Oracle – це гарний варіант.

Фінансові установи. Oracle широко використовується у фінансовому секторі, де цілісність та безпека даних є надзвичайно важливими. Банки, страхові компанії та інвестиційні фірми часто покладаються на Oracle для управління конфіденційними фінансовими даними та транзакціями.

Уряд та державний сектор. Oracle часто обирають завдяки його надійним функціям та безпеці в урядових та державних секторах, включаючи системи національної безпеки, охорони здоров'я та транспорту.

2.2.4 PostgreSQL

Система керування базами даних PostgreSQL поділяє свою популярність з MySQL. Це об'єктно-реляційна СКБД, де визначені користувачем об'єкти та табличні підходи поєднуються для побудови складніших структур даних. Крім того, PostgreSQL має багато спільного з MySQL. Вона спрямована на посилення стандартів відповідності та розширюваності. Отже, вона може обробляти будь-яке робоче навантаження, як для продуктів на одній машині, так і для складних додатків. Належить та розроблена PostgreSQL Global Development Group, вона все ще залишається повністю з відкритим вихідним кодом. Ця СКБД доступна для використання з такими платформами, як Microsoft, iOS, Android та багатьма іншими.

Плюси Postgre.

Чудова масштабованість. Вертикальна масштабованість є відмінною рисою PostgreSQL. Враховуючи, що майже будь-яке програмне рішення на замовлення має тенденцію до зростання, що призводить до розширення бази

даних, цей конкретний варіант, безумовно, підтримує зростання та розвиток бізнесу.

Підтримка користувацьких типів даних. PostgreSQL за замовчуванням підтримує багато типів даних, таких як JSON, XML, H-Store та інші. PostgreSQL використовує це, будучи однією з небагатьох реляційних баз даних із сильною підтримкою функцій NoSQL. Крім того, вона дозволяє користувачам визначати власні типи даних. Оскільки бізнес-модель програмного забезпечення може потребувати різних типів баз даних протягом свого існування для кращої продуктивності або повноти застосування, ця опція забезпечує підвищену гнучкість.

Легко інтегровані сторонні інструменти. Система керування базами даних PostgreSQL має потужну підтримку додаткових інструментів, як безкоштовних, так і комерційних. До їх складу входять розширення для покращення багатьох аспектів. Наприклад, ClusterControl надає вражаючу допомогу в управлінні, моніторингу та масштабуванні баз даних з відкритим кодом SQL та NoSQL. Щоб зробити порівняння та синхронізацію даних ефективнішими, розгляньте можливість використання директиви даних бази даних. Якщо є перспектива масштабувати свої дані до великих робочих навантажень, система резервного копіювання та відновлення pgBackRest буде гарним варіантом.

Відкритий код та підтримка спільноти. Postgres має повністю відкритий код та підтримується спільнотою, що зміцнює його як повноцінну екосистему. Крім того, розробники завжди можуть розраховувати на безкоштовну та оперативну допомогу спільноти.

Мінуси Postgre.

Непослідовна документація. Хоча PostgreSQL має велику спільноту та всебічно підтримує своїх учасників, документації все ще бракує послідовності та повноти. Оскільки спільнота PostgreSQL досить розподілена, документація не відповідає єдиним стандартам для всіх функцій Postgre.

Відсутність інструментів звітності та аудиту. Істотним недоліком PostgreSQL є відсутність інструментів для редагування, які б відображали

поточний стан бази даних. Вам доводиться постійно перевіряти, чи щось пішло не так. Завжди існує ризик того, що інженери баз даних помітять збій занадто пізно.

Варіанти використання.

Завдяки складним запитам та широкому вибору користувацьких інтерфейсів із попередньо визначеними функціями, PostgreSQL ідеально підходить для аналізу даних та зберігання даних. Якщо розробляти інструмент автоматизації баз даних, PostgreSQL найкраще підходить для цього завдяки своїм потужним аналітичним можливостям, сумісності з ACID та потужному механізму SQL. Загалом, він значно пришвидшує обробку величезних обсягів даних. Ця СКБД популярна серед фінансових установ та телекомунікаційних систем.

2.2.5 MSSQL

Як повністю комерційний інструмент, Microsoft SQL Server є однією з найпопулярніших реляційних СКБД, окрім MySQL, PostgreSQL та Oracle. Він добре справляється з ефективним зберіганням, зміною та керуванням реляційними даними. Для взаємодії з базами даних SQL Server інженери баз даних зазвичай використовують мову Transact-SQL (T-SQL), яка є розширенням стандарту SQL.

Плюси MSSQL.

Різноманітність версій. Microsoft SQL Server пропонує широкий вибір різних опцій з різноманітними функціональними можливостями. Наприклад, версія Express з безкоштовною базою даних пропонує інструменти початкового рівня, що ідеально підходить для навчання та створення настільних або невеликих серверних додатків, керованих даними. Варіант для розробників дозволяє створювати та тестувати додатки, включаючи деякі корпоративні функції, але без ліцензії на виробничий сервер. Для більших проектів також існують версії Web, Standard та Enterprise з різним обсягом адміністративних можливостей та рівнями обслуговування.

Комплексне рішення для роботи з бізнес-даними. Зосереджуючись переважно на комерційних рішеннях, MSSQL надає багато функцій з доданою вартістю для бізнесу. Додатковий вибір компонентів дозволяє створювати ETL-рішення, формувати базу знань та впроваджувати очищення даних. Крім того, він надає інструменти для загального адміністрування даних, онлайн-аналітичної обробки та аналізу даних, а також пропонує рішення для створення звітів та візуалізації.

Багата документація та допомога спільноти. Оскільки Microsoft SQL Server спрямований на комплексне обслуговування баз даних, повна онлайн-документація також відображає цю концепцію. Відповідно структуровані інструкції, численні офіційні документи та демонстрації дають повне уявлення про систему обробки даних MSSQL. Крім того, Microsoft Premier надає доступ до спеціалізованої підтримки спільноти Microsoft, що є перевагою, коли інженеру баз даних потрібна допомога.

Підтримка хмарних баз даних. Як частина єдиної екосистеми Microsoft, MSSQL можна інтегрувати з Microsoft Cloud, базою даних Azure SQL або SQL Server на віртуальних машинах Azure. Ці рішення дозволяють перенести адміністрування баз даних у хмару, якщо база даних вашого бізнес-програмного забезпечення стає справді перевантаженою та складною в адмініструванні.

Мінуси MSSQL.

Висока вартість. Оскільки MSSQL Server використовується переважно в корпоративному масштабі, він залишається одним із найдорожчих рішень. Якщо говорити про цифри, то версія Enterprise наразі коштує понад 15 123 доларів за ядро та продається в пакетах із 2 ядер.

Нечіткі та плаваючі умови ліцензування. Ще однією проблемою є постійно мінливий процес ліцензування. Саму цінову стратегію важко зрозуміти, а елементи, включені до певного видання, є плаваючими, тобто мають тенденцію переходити від одного до іншого.

Складний процес налаштування. Для новачків, яким доводиться працювати з важкими наборами даних, робота з оптимізацією запитів та

налаштуванням продуктивності може бути проблематичною. Оскільки процес не такий очевидний, він може створити суттєві вузькі місця на ранньому етапі.

Варіанти використання.

MSSQL Server – це розумний варіант для компаній, які мають передплати на інші продукти Microsoft. Оскільки Microsoft створює надійну екосистему з безперешкодною інтеграцією служб, MSSQL стає потужним рішенням для баз даних. Завдяки хмарній доступності та розширеним інструментам пошуку даних, MSSQL виявляється цінним активом для бізнесу, забезпечуючи стійку та ефективну систему, яка відповідає потребам, що змінюються.

2.2.6 SQLite

SQLite – це самодостатня, безсерверна система керування базами даних, яка не потребує налаштування. Вона часто використовується як вбудована база даних і популярна для невеликих мобільних та настільних додатків.

Плюси SQLite.

Невеликий за розміром та легко портативний. SQLite – це оптимізований механізм баз даних, який працює без окремого серверного процесу. Вся база даних міститься в одному кросплатформному файлі на диску, що підвищує її портативність та спрощує інтеграцію в програми.

Мінімальне споживання ресурсів. SQLite розроблено для оптимальної ефективності використання пам'яті та дискового простору, що робить його ідеальним вибором для програм з обмеженими ресурсами, таких як ті, що використовуються в мобільних пристроях та пристроях Інтернету речей.

Надійна та зручна у використанні. SQLite – це база даних, сумісна з ACID, що забезпечує цілісність та узгодженість даних. Крім того, вона проста в налаштуванні та вимагає мінімальної конфігурації.

Мінуси SQLite.

Обмежений паралелізм. SQLite використовує блокування на основі файлів, що обмежує його можливості керування кількома одночасними операціями запису. Це робить його менш придатним для програм з високим рівнем

паралельного запису або для кількох користувачів, які одночасно звертаються до бази даних.

Відсутність розширених функцій. SQLite не має деяких складних функцій, що є в інших системах керування базами даних, таких як збережені процедури, тригери або функції, визначені користувачем.

Обмежена масштабованість. Через свою безсерверну структуру, SQLite не призначений для великих застосунків або розподілених налаштувань. Його продуктивність може знижуватися під час обробки значних наборів даних або підвищеного рівня одночасного доступу.

Варіанти використання.

SQLite добре підходить для невеликих за розміром застосунків, а також для мобільних і настільних додатків, які потребують легкої, портативної та зручної бази даних. Він також підходить для вбудованих систем і пристроїв Інтернету речей з обмеженими ресурсами, де впровадження серверної СКБД було б неможливим.

2.2.7 MongoDB

MongoDB – це безкоштовна нереляційна СКБД з відкритим вихідним кодом, яка також має комерційну версію. Хоча MongoDB спочатку не призначалася для обробки структурованих даних, її можна використовувати для програм, які використовують як структуровані, так і неструктуровані дані. У MongoDB бази даних підключаються до програм через драйвери баз даних. Вони широко доступні в системі керування базами даних. Кілька типів даних обробляються одночасно, і для цього використовується внутрішній кеш.

Плюси MongoDB.

Простий доступ до даних, зберігання, введення та пошук. Однією з переваг MongoDB, що випливає з її NoSQL-природи, є швидка та проста робота з даними. Тобто, дані можна вводити, зберігати та вилучати з бази даних швидко та без будь-якого додаткового підтвердження. Як і в будь-якій іншій нереляційній базі даних, вона робить акцент на використанні оперативної пам'яті, тому записи

можна маніпулювати дуже швидко та без будь-яких наслідків для цілісності даних.

Легка сумісність з іншими моделями даних. MongoDB легко поєднується з різними системами керування базами даних, як SQL, так і NoSQL типів. Крім того, він має API-інтерфейси для підключаємих механізмів зберігання даних. Коротше кажучи, ця опція дозволяє третім сторонам створювати власні механізми зберігання даних для MongoDB. З комерційної точки зору, це створює додаткову цінність для бізнес-програмного забезпечення.

Горизонтально масштабоване рішення. Масштабованість – коли дані розподілені по розподіленій мережі керованих серверів – є аспектом фундаментальної природи MongoDB. Вона стає ще важливішою для підприємств, що працюють з великими даними. Крім того, база даних може розподіляти дані по кластеру машин. Дані розподіляються швидше та рівномірно, без громіздкості. Оскільки це призводить до швидшої обробки даних, також прискорюється продуктивність програми.

Мінуси MongoDB.

Значне споживання пам'яті. Процес денормалізації, коли попередньо нормалізовані дані в базі даних групуються для підвищення продуктивності, зазвичай призводить до високого споживання пам'яті. Крім того, ця СКБД зберігає в пам'яті всі імена ключів для кожної пари значень. Крім того, через відсутність підтримки з'єднань, бази даних Mongo мають надлишок даних, що призводить до великих втрат пам'яті та зниження продуктивності програм.

Незахищеність даних. Зосереджуючись на швидкій роботі з даними, MongoDB, як і будь-яка інша NoSQL СКБД, не має захисту даних. Оскільки автентифікація користувачів не є опцією Mongo за замовчуванням, а вищий рівень захисту доступний лише в комерційній версії, не можна вважати її повністю безпечною. Крім того, постійно виходять оновлення MongoDB без гарантії, що всі поправки чи зміни даних працюватимуть так само, як і раніше. Майте на увазі, що всі маніпуляції повинні бути зосереджені на цих оновленнях та проходити додаткові тести.

Складний процес інтерпретації в інші мови запитів. Оскільки MongoDB спочатку не розроблявся для роботи з реляційними моделями даних, продуктивність у цих випадках може знижуватися. Крім того, перетворення SQL на запити MongoDB вимагає додаткових дій для використання механізму, що може затримати розробку та розгортання.

Варіанти використання.

MongoDB найкраще працює в інтеграції даних у режимі реального часу та масштабованості бази даних. Наприклад, це правильний варіант для каталогів продуктів завдяки своїй здатності зберігати множину об'єктів з різними колекціями атрибутів. Також варто розглянути аналітичні платформи, оскільки швидкість MongoDB забезпечує динамічну продуктивність, яка може допомогти відстежувати поведінку користувача в режимі реального часу.

2.2.8 Redis

Redis – це NoSQL-сховище структур даних з відкритим вихідним кодом, яке також можна використовувати як кеш. Замість документів воно використовує пари ключ-значення. Його відмінною рисою є те, що існує кілька варіантів структурування даних, таких як списки, набори та хеші.

Забезпечуючи реплікацію даних та підтримку транзакцій, Redis виконує команди в черзі, а не по одній.

Плюси Redis.

Швидке рішення. Завдяки своїм функціям реплікації та транзакцій, Redis обробляє дані дуже швидко. Відсутність залежностей та тип сховища даних в пам'яті роблять Redis гідним конкурентом навіть серед простих SQL-альтернатив.

Масштабна обробка даних. З точки зору сприйняття та уточнення даних, Redis можна вважати колосом. Він може легко завантажувати до 1 ГБ даних для одного запису. Додайте вбудоване кешування даних, і ви отримаєте потужну машину обробки даних.

Мінуси Redis.

Залежність від пам'яті програми. Повна залежність від пам'яті програми є реальним недоліком. Тобто, база даних вийде з ладу, якщо її розмір перевищить розмір доступної пам'яті.

Немає підтримки мови запитів або об'єднань. Щодо сумісності з іншими типами наборів даних, Redis відстає. Враховуючи, що в певний момент бізнесу може знадобитися масштабування та використання інших форматів даних, використання швидкого введення даних як єдиного варіанту залишає це питання відкритим.

Варіанти використання.

Redis має кілька різних напрямків для роботи. Перший з них – це IoT-додатки. Тут великі обсяги даних з IoT-пристроїв можуть бути передані до Redis для обробки цих записів, перш ніж зберігати їх у будь-якому постійному сховищі даних. Крім того, Redis є ідеальним варіантом для мікросервісних архітектур із масштабованим хмарним хостингом. Оскільки дані тут не обов'язково повинні зберігатися довгостроково, Redis здається розумним рішенням.

2.2.9 Cassandra

Cassandra – це децентралізована система, розроблена Apache. Це безкоштовна СКБД на базі Java, сильними сторонами якої є функції багаторазової реплікації та багаторазового розгортання. Ці особливості дозволяють копіювати численні запити та розгортати їх усі одночасно. Завдяки швидкій масштабованості Cassandra дозволяє керувати великими обсягами даних, реплікуючи їх на кілька вузлів. Це усуває проблему збою бази даних – якщо деякі вузли виходять з ладу в будь-який момент, вони негайно замінюються, і система продовжує працювати, поки хоча б один вузол безпечний.

Cassandra використовує власну мову запитів, CQL. За своїм синтаксисом вона дуже схожа на SQL, але не застосовує з'єднання, замінюючи їх так званими сімействами стовпців. А друга відмінність полягає в тому, що не всі стовпці в таблиці зберігаються для підзапитів. Деякі з них використовуються як стовпці

кластеризації, де суміжні дані розміщуються поруч один з одним для швидкого пошуку. Це забезпечує швидше виконання запитів з величезних наборів даних, прискорюючи обробку даних.

Плюси Cassandra.

Безпека даних. Завдяки функції реплікації головного вузла, Cassandra залишається стійкою до збоїв. Це означає, що інженери баз даних можуть бути впевнені в безпеці даних, якщо тільки всі головні вузли не вийдуть з ладу одночасно. Поки це вкрай малоймовірно, база даних і застосунок, побудований на ній, залишатимуться надійними та безпечними.

Гнучкість та можливість внесення змін. Простий синтаксис Cassandra поєднує найкраще з SQL та NoSQL. Окрім масштабованості, він значною мірою сприяє гнучкості набору даних. Cassandra збирає дані в дорозі, а пошук даних має таку ж простоту, незважаючи на розмір набору даних. Це дозволяє максимально розширити базу даних.

Мінуси Cassandra.

Повільне читання. Оскільки Cassandra спочатку розроблялася для швидкого запису, її слабкість полягає в нездатності до швидкого читання. Одна з причин цього полягає в тому, що система не має вузьких місць для вхідної інформації. Тож, хоча дані можна швидко записати в базу даних, системі може знадобитися більше часу для обробки та отримання цих даних. Це можна також пояснити тим фактом, що Cassandra розподіляє дані по кількох вузлах у кластері. Під час запиту даних їй, можливо, доведеться читати з різних вузлів, що може уповільнити продуктивність читання.

Потреба в додаткових ресурсах. Оскільки Cassandra обробляє кілька шарів даних одночасно, вона вимагає достатньо потужності для цього. Це означає додаткові інвестиції як у програмне, так і в апаратне забезпечення. Якщо компанія вперше стикається з такою необхідністю і не впевнена в ресурсах, можливо, їй варто розглянути інші системи баз даних.

Варіанти використання.

Завдяки рівномірному розподілу даних, Cassandra є актуальною в додатках, де обробляються великі обсяги інформації. Наприклад, це чудовий вибір для центрів обробки даних. Крім того, Cassandra добре підходить для аналітики в реальному часі, оскільки дозволяє лінійне масштабування та збільшення обсягу даних у режимі реального часу. Можна також розглянути її для додатків з постійним потоком даних, таких як додатки для прогнозування погоди. Іншим варіантом є використання її як СКБД для магазину електронної комерції, оскільки вона дозволяє зберігати історію покупок та інші транзакції. Додамо сюди можливість відстеження таких типів даних, як статус замовлення та посилки, і вийде повноцінне рішення з інтеграцією доставки електронної комерції.

2.2.10 Elasticsearch

Elasticsearch – це NoSQL, документоорієнтована система керування базами даних, в основі якої лежить повнотекстовий пошуковий механізм. Побудована на бібліотеці Apache Lucene, вона зберігає дані у вигляді JSON-файлу, підтримує RESTful API та використовує потужний аналітичний механізм для швидшого пошуку даних. Будучи програмним забезпеченням з відкритим вихідним кодом, вона включає як безкоштовну, так і платну версії.

Плюси Elasticsearch.

Масштабована архітектура. Однією з особливостей Elasticsearch є його надійна розподілена архітектура. Ключові параметри структури, такі як кластеризація, індексація, шардінг та багато інших, забезпечують широке горизонтальне масштабування, що дозволяє вміщувати терабайти записів з подальшою автоматизацією. Рівні абстракції архітектури оптимізують управління системою як на індивідуальному, так і на агрегованому рівнях.

Швидка обробка даних. Завдяки розподіленій структурі даних та вбудованому паралелізуванню, база даних Elasticsearch демонструє чудові результати продуктивності. Навіть під час виконання складного запиту даних

вона генерує блискавичні результати пошуку. Частково це можливо завдяки тому, що документи зберігаються близько до відповідних метаданих в індексі, що робить їх швидким для пошуку.

Мінуси Elasticsearch.

Відсутність багатомовної підтримки. При обробці даних запитів або відповідей СКБД Elasticsearch відстає. Хоча вона ідеально поєднується з Cassandra DB для покращення продуктивності бази даних, інші мови та формати для неї недоступні. У цьому плані вона підтримує лише формат документів JSON.

Обмежені інструменти постійної перевірки справності. Коли щось йде не так, що може статися на будь-якому етапі, Elasticsearch може відображати статус лише як «жовтий» або «червоний». Простіше кажучи, у нього немає інструментів звітності. Хоча проблеми зазвичай стосуються порогу пам’яті або обсягу диска, інженери адміністраторів баз даних скаржаться на цю ситуацію.

Варіанти використання.

Завдяки своїй розподіленій природі NoSQL та гнучким моделям даних, Elasticsearch є чудовим інструментом для продуктів електронної комерції з величезними базами даних, які, як правило, використовують пошукові системи. Це дуже корисно під час створення або оновлення профілю клієнта з урахуванням робочого навантаження, яке зазвичай вимагає взаємодія в режимі реального часу.

2.2.11 Бази даних Firebase

Firebase, що належить Google, – це серверна служба реального часу (Backend-as-a-Service), яка використовується для розробки веб- та мобільного програмного забезпечення. Що стосується NoSQL баз даних, існує два варіанти: Firebase Realtime Database (що забезпечує доступ до даних, що зберігаються на різних платформах, у режимі реального часу) та Cloud Firestore (що пропонує більшу масштабованість та складніші моделі даних). Таким чином, обидва рішення добре вписуються в сценарій, коли вам потрібно працювати з великою

кількістю даних у режимі реального часу: зміни до баз даних вносяться в міру їх виникнення. Дочірня платформа Google зберігає дані у форматі JSON та надає різні можливості керування даними, включаючи зручний інструмент перегляду даних.

Плюси Firebase.

Зручність для початківців. Firebase може бути чудовим варіантом, коли мало досвіду в розробці програмного забезпечення, оскільки він пропонує просте у використанні середовище для початку проекту.

Зручний доступ до даних. Як Realtime, так і Firestore – чудові варіанти для зберігання та керування різними типами даних. Для легкого доступу до даних є консоль Firebase. Будучи хмарними та NoSQL, вони пропонують гідну гнучкість та масштабованість, коли обсяг даних зростає. Крім того, інструменти Firebase дозволяють працювати з адаптивними додатками та оновлювати дані навіть за відсутності підключення до Інтернету.

Першокласна документація. Рішення постачається з добре написаною документацією, яка спрощує роботу з наданими послугами для всіх користувачів. Вона включає інструкції, технічну документацію, посилання на SDK, інформацію про інтеграцію та багато іншого. Якщо повернутися до опитування StackOverflow, Firebase є 12-м за популярністю вибором розробників баз даних. Розмір спільноти розробників продукту є значним, що дозволяє легко знаходити відповіді на проблеми, що виникають.

Мінуси Firebase.

Обмежені можливості запитів. Хоча це стосується лише бази даних реального часу, це все ще проблема, якщо планується використовувати переважно це сховище. Проблема полягає в тому, що ви обмежені створенням простих запитів, оскільки немає можливостей фільтрації для складніших. Це пояснюється тим, що вся база даних є великим JSON-файлом без опцій моделювання даних.

Обмежена міграція даних. Якщо використовувати Firebase для розміщення всіх своїх даних, їх перенесення на іншу платформу може стати проблемою.

Сервісу бракує інструментів міграції для перенесення даних або встановлення бази даних проєкту за замовчуванням.

Варіанти використання.

Бази даних Firebase можуть бути гарним варіантом для розгляду, коли програмне забезпечення працює з даними в режимі реального часу, які потрібно синхронізувати між різними браузерами та пристроями. Їх часто обирають для таких проектів, як месенджери, соціальні мережі та ігрові додатки.

Amazon DynamoDB.

Amazon DynamoDB – це NoSQL-сервіс баз даних, керований Amazon Web Services (AWS), розроблений для застосунків, що потребують високої масштабованості, низької затримки та стабільної продуктивності.

Плюси Amazon DynamoDB.

Виняткова масштабованість. DynamoDB може легко масштабуватися вгору або вниз, щоб врахувати будь-який рівень трафіку та даних, що робить його ідеальним для застосунків, які швидко зростають або коливаються за потреби.

Низька затримка. DynamoDB забезпечує затримку в мілісекунду для операцій читання та запису, забезпечуючи швидкий та стабільний доступ до даних, що життєво важливо для програм реального часу.

Повністю керований сервіс. DynamoDB обробляє операційні завдання, такі як налаштування обладнання, встановлення виправлень та резервне копіювання, як керований сервіс, що дозволяє зосередитися на розробці та функціональності програми.

Адаптивна модель даних. DynamoDB підтримує різні моделі даних, включаючи моделі типу «ключ-значення» та моделі, орієнтовані на документи, що забезпечує гнучкість у структуруванні та запитах до даних.

Мінуси Amazon DynamoDB.

Висока вартість. Цінова структура DynamoDB може бути складною та призвести до вищих витрат порівняно із самостійно керованими NoSQL базами

даних, особливо для програм зі змінними або непередбачуваними робочими навантаженнями.

Обмежені можливості запитів. Хоча DynamoDB підтримує базові запити та фільтри, він не забезпечує підтримку складних операцій запитів та агрегації, необхідних у деяких випадках використання.

Прив’язаність до постачальника. Оскільки DynamoDB є власним сервісом AWS, перехід з DynamoDB на іншу систему баз даних може вимагати значних зусиль та планування.

Варіанти використання.

Amazon DynamoDB добре підходить для застосунків, що потребують високої масштабованості, низької затримки та стабільної продуктивності, таких як безсерверні додатки, платформи електронної комерції, ігрові платформи та рішення Інтернету речей. Він також ідеально підходить для безсерверних архітектур та застосунків, що використовують інші сервіси AWS, оскільки він бездоганно інтегрується з екосистемою AWS.

2.3 Як вибрати систему керування базами даних

Окрім варіантів, описаних у публікації, існує багато інших систем керування базами даних. Кожна з них хороша по-своєму, але має й деякі недоліки. Хоча ми не розглянули навіть третини всіх баз даних, ми спробували порівняти ті, що зазвичай використовуються як для невеликих веб-застосунків, так і для великих систем сховищ даних.

Отже, як вибрати правильний варіант для власного програмного забезпечення?

Починаючи локальний бізнес у сфері електронної комерції, доцільно використовувати бази даних на кшталт MySQL як розумну відправну точку, що також ефективно працює з веб-інструментами бізнес-аналітики та системами OLTP.

У разі створення масштабної платформи електронної комерції з повним циклом взаємодії з клієнтами доцільно обрати базу даних Cassandra. Для

доповнення її можливостей потужною пошуковою системою варто інтегрувати рішення на основі Elasticsearch.Говорячи про Кассандру, це також досить респектабельний варіант для центрів обробки даних та аналітики в режимі реального часу з величезними обсягами даних.

Якщо говорити про аналітичні інструменти без кількох шарів даних, можливо, доцільно обрати NoSQL бази даних, такі як MongoDB. Вони також добре працюють з каталогами продуктів.

Слідом за застосуваннями сховищ даних варто згадати також MSSQL, особливо для компаній, які мають низку інших підписок Microsoft.

З точки зору побудови OLTP-рішень та додатків для сховищ даних, Oracle також є гарним вибором.

IoT-додатки та архітектура мікросервісів, які прагнуть масштабувати свої ресурси для розміщення даних, підсумують наш список найкращих варіантів використання Redis.

Звичайно, є й інші системи баз даних, які варто розглянути. Все залежить від вашої бізнес-моделі та потреб бізнесу.

2.4 Порівняння мов програмування для розробки веб-платформи управління фінансами

З багатьох мов фронтенду та бекенду найпопулярнішими є .NET, PHP, Java та Python. Але як вибрати одну з цих перевірених мов фронтенду та бекенду для розробки веб-сайтів?

Веб-розробка стрімко розвивається, щодня з'являються нові технології та мови. Кілька перевірених мов витримали випробування часом, коли справа доходить до веб-розробки. Серед .NET, PHP, Java та Python, PHP зазвичай використовується 77,5% усіх веб-сайтів [7]. Це не означає, що PHP найкращий для всіх проектів розробки.

Хоча PHP є однією з найбільш використовуваних мов програмування, яку використовують майже в 77,5% веб-сайтів, вона може не підходити в багатьох сценаріях (див. рис. 2.2).

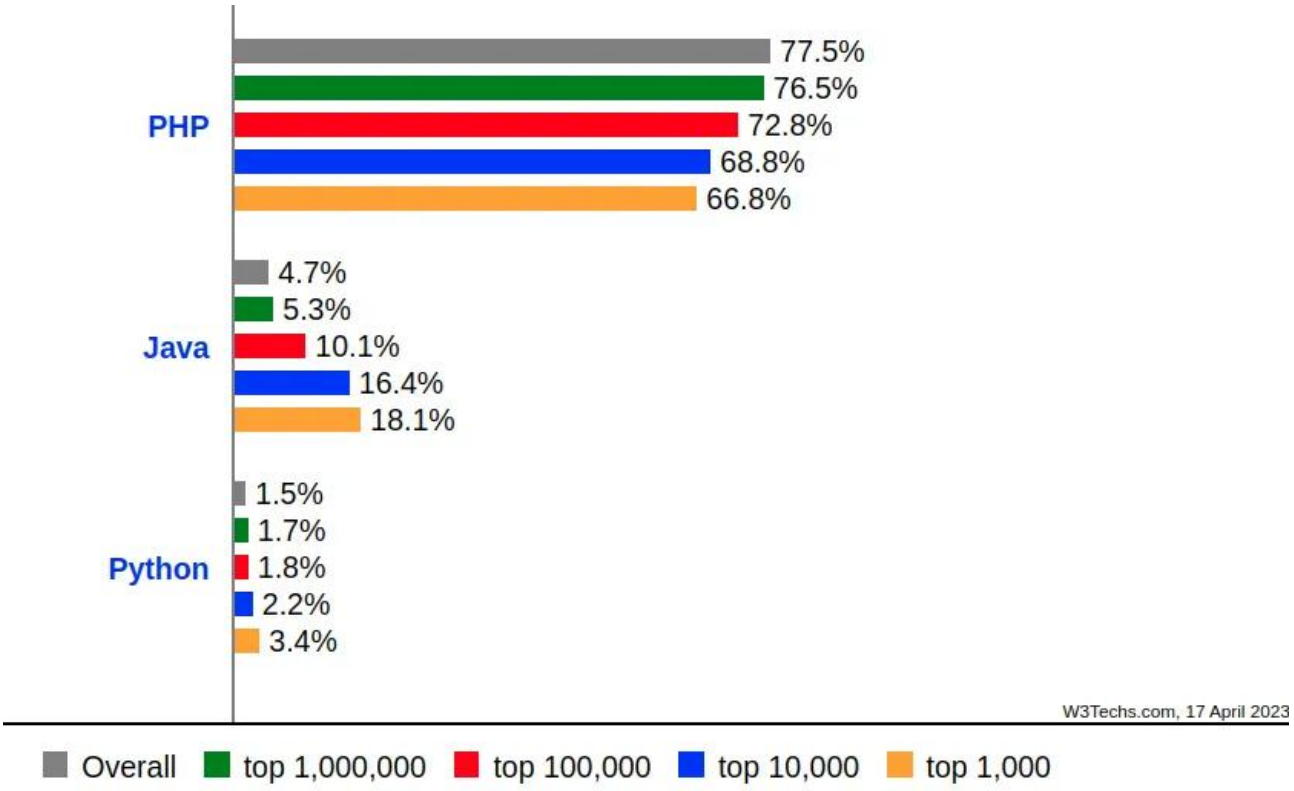

Рисунок 2.2 – Мови програмування для розробки веб-додатків [7]

2.4.2 Порівняння .NET, PHP, Java та Python

Платформа .NET.

Microsoft розробила .NET Framework, який пропонує надійну архітектуру програмування для розробки додатків. Він має значну бібліотеку попередньо написаного коду та інструментів, що спрощують процес розробки (таблиця 2.3).

Таблиця 2.3 – Переваги та недоліки .NET Web Development

👍 Переваги	👎 Недоліки
• Надійний фреймворк	• Вартість
• Масштабованість	• Обмежена портативність
• Продуктивність	
• Безпека	
• Підтримка спільноти	

За допомогою .NET програмісти можуть створювати масштабовані, швидкі та прості в оновленні додатки. Оскільки мова підтримує численні мови програмування, такі як C#, F# та Visual Basic, і може створювати веб-додатки для будь-якої платформи, саме .NET часто використовується для веб-розробки. Крім того, .NET відомий своїми надійними функціями безпеки, що робить його популярним варіантом для розробки безпечних онлайн-додатків.

Переваги .NET для веб-розробки.

- Надійний фреймворк: .NET пропонує міцну основу для створення вебзастосунків, що спрощує для програмістів швидку розробку коду.
- Масштабованість: масштабованість .NET робить його чудовим варіантом для створення складних онлайн-додатків.
- Продуктивність: це ідеальний вибір для веб-сайтів та додатків з високим трафіком завдяки винятковій продуктивності.
- Безпека: .NET чудово підходить для створення безпечних онлайн-додатків, оскільки містить кілька вбудованих функцій безпеки.
- Підтримка спільноти: Численна та динамічна спільнота розробників .NET активно сприяє розробці фреймворку.

Мінуси .NET для веб-розробки.

- Вартість: хоча .NET є чудовим фреймворком, він може бути дорогим для малого бізнесу та стартапів.
- Обмежена портативність: .NET розроблений переважно для систем на базі Windows, що може обмежувати його портативність на інші платформи.

Приклади популярних веб-сайтів, створених за допомогою .NET:

- Stack Overflow: Stack Overflow – це популярний веб-сайт для розробників, створений за допомогою .NET та інших технологій.
- Microsoft.com: Офіційний веб-сайт Microsoft створено за допомогою .NET.

2.4.3 PHP

PHP – одна з найпопулярніших мов програмування для створення веб-сайтів. З моменту свого першого випуску в 1995 році вона стала однією з найпопулярніших мов програмування для розробки динамічних веб-сайтів та веб-додатків.

Оскільки PHP є мовою з відкритим вихідним кодом, розробники можуть використовувати власні сервіси розробки на PHP, що знижує вартість проекту. PHP має широкий спектр бібліотек та фреймворків, які спрощують створення складних веб-додатків, швидко та ефективно (таблиця 2.4).

Таблиця 2.4 – Переваги та недоліки PHP Web Development

👍 Переваги	👎 Недоліки
• Легкість у вивченні	• Безпека
• Відкритий код	• Відсутність стандартизації
• Продуктивність	
• Масштабованість	
• Підтримка спільноти	

Переваги PHP для веб-розробки:

– Легкість навчання: PHP легко вивчити і може бути гарним вибором для початківців.

– Відкритий код: Оскільки PHP є мовою програмування з відкритим кодом, її можна модифікувати відповідно до потреб бізнесу, і вона безкоштовна у використанні.

– Масштабованість: Високий рівень масштабованості PHP робить його ідеальною платформою для створення складних веб-додатків.

– Підтримка спільноти: PHP має значну та активну спільноту розробників, яка активно сприяє розвитку мови, що спрощує для розробників та компаній з веб-розробки на PHP вирішення проблем та звернення за допомогою за потреби.

Мінуси PHP для веб-розробки.

– Безпека: У минулому PHP стикався з кількома проблемами безпеки, такими як SQL-ін'єкції, міжсайтовий скриптинг, CSRF тощо, що може викликати певне занепокоєння у розробників.

– Відсутність стандартизації: PHP не має стандартів для забезпечення якості коду, зручності обслуговування та безпеки, таких як PSR, OWASP, W3C, RFC, ISO тощо, що може ускладнити розробникам створення портативного коду.

Приклади популярних веб-сайтів, створених на PHP:

– Facebook: найбільша у світі соціальна мережа.

– Вікіпедія: відома онлайн-енциклопедія.

– WordPress: відома система управління контентом.

2.4.4 Java

Java, одна з найвідоміших мов програмування на ринку, вперше була випущена в 1995 році. Вона витримала випробування часом. Як об'єктно-орієнтована мова, Java побудована на ідеї взаємодії один з одним для виконання завдань. Java може працювати на будь-якій системі, на якій встановлено віртуальну машину Java (JVM).

Java пропонує різноманітні бібліотеки, фреймворки та інструменти, які допомагають програмістам швидко та ефективно створювати код. Це чудовий варіант для створення складних та безпечних онлайн-додатків. Компанії, що займаються розробкою веб-додатків на Java, часто використовують цю мову для веб-розробки та додатків корпоративного рівня.

Таблиця 2.5 – Переваги та недоліки Java Web Development

👍 Переваги	👎 Недоліки
• Платформна незалежність	• Висока складність навчання
• Масштабованість	• Керування пам’яттю
• Продуктивність	• Повільний старт
• Безпека	
• Підтримка спільноти	

Переваги Java для веб-розробки.

- Платформно-незалежна: Java – це платформно-незалежна мова, що дозволяє їй функціонувати на будь-якій операційній системі.
- Масштабованість: Java дуже масштабована, що робить її чудовим вибором для створення складних онлайн-додатків.
- Продуктивність: Java має хорошу продуктивність, що робить її придатною для програм та веб-сайтів з великим обсягом трафіку.
- Безпека: Java відома своїми надійними функціями безпеки. Ви можете найняти розробників Java для створення безпечних онлайн-додатків.
- Підтримка спільноти: Java має значну та активну спільноту розробників, яка сприяє розвитку мови, що спрощує для розробників вирішення проблем та звернення за допомогою за потреби.

Мінуси Java для веб-розробки

- Крута крива навчання: Часто запитують, чи легко вивчити Java? Відповідь полягає в тому, що Java – це складна мова, яку може бути важко опанувати, особливо новачкам.
- Керування пам'яттю: Java потребує більше пам'яті, ніж інші мови програмування, що може бути проблематичним для онлайн-застосунків з обмеженими ресурсами.
- Повільний час запуску: Повільний час запуску Java може впливати на загальну ефективність роботи веб-застосунків. Java може запускатися повільніше, ніж інші мови програмування.
- Приклади популярних веб-сайтів, створених за допомогою Java
- LinkedIn: Відомий професійний мережевий сайт LinkedIn.
- Amazon: Один з найбільших інтернет-магазинів.
- eBay: Як веб-сайт, так і застосунок розроблені за допомогою Java для цього онлайн-аукціонного сайту eBay.

2.4.5 Python для веб-розробки

Python був вперше випущений у 1991 році та є мовою програмування високого рівня. Це мова загального призначення, яка широко використовується для машинного навчання, штучного інтелекту, аналізу даних та веб-розробки. Python відомий своєю читабельністю та простотою, що робить його чудовим варіантом як для початківців-розробників, так і для досвідчених професіоналів.

Python використовує різні модулі та фреймворки, такі як Django, Flask, Pyramid та Bottle, щоб зробити створення веб-додатків простим, швидким та ефективним. Django, Flask, Pyramid та Bottle – це лише деякі з популярних фреймворків, що використовуються для розробки на Python.

Таблиця 2.6 – Переваги та недоліки Python Web Development

👍 Переваги	👎 Недоліки
• Легкість у вивченні	• Продуктивність
• Швидкість розробки	• Глобальний Перекладач Шлюз (GIL)
• Масштабованість	• Слабкі можливості для мобільної розробки
• Аналіз даних	
• Підтримка спільноти	

Переваги Python для веб-розробки.

- Легко вивчати: Python відомий своєю читабельністю та простотою, що спрощує його вивчення навіть новачками.
- Швидка розробка: Велика колекція інструментів та фреймворків Python спрощує швидке створення веб-додатків.
- Масштабованість: Python дуже масштабований, що робить його чудовим варіантом для створення складних вебзастосунків.
- Аналіз даних: Python чудово підходить для створення веб-додатків на основі даних, оскільки він широко використовується в аналізі даних.
- Підтримка спільноти: Численна та динамічна спільнота розробників Python активно сприяє розвитку мови.

Мінуси Python для веб-розробки.

- Продуктивність: Продуктивність Python може бути не такою чудовою, як у інших мов програмування для застосунків, що потребують великої кількості обчислень.
- Глобальне блокування інтерпретатора: Python має GIL, або глобальне блокування інтерпретатора, яке може знижувати продуктивність, коли використовується багато потоків.
- Слабкі сторони мобільних обчислень: Python не підходить для розробки мобільних додатків.

Приклади популярних веб-сайтів, створених на Python.

- Instagram: Програмісти використовували Django та Python для розробки цієї відомої мережі соціальних медіа.
- Spotify: Відомий сервіс потокового передавання музики.
- Dropbox: Відомий сервіс хмарного сховища.

2.5 Порівняльний аналіз PHP, ASP.NET, Python та Java

Ось короткий порівняльний аналіз PHP, ASP.NET, Python та Java, який допоможе визначити, яка мова програмування відповідає конкретним потребам (табиця 2.7).

Таблиця 2.7 – Порівняння .NET, PHP, Java та Python

Особливість	.NET	PHP	Java	Python
Синтаксис	C# та візуальний дизайн Базовий	Схожий на C	Схожий на C	Простий та легкий для читання
Популярність	Широке використання в Windows	Популярний у веб-розробці	Популярний у корпоративному середовищі	Популярний для веб і даних наука
Легкість використання	Помірна	Висока	Помірна	Висока
Крива навчання	Помірна	Плавна	Помірна	Плавна
Продуктивність	Висока	Помірна	Висока	Помірна
Підтримка спільноти	Велика	Велика	Велика	Велика

Продовження таблиці 2.7

Особливість	.NET	PHP	Java	Python
Сумісність	Windows та Linux	Кросплатформна	Кросплатформна	Кросплатформна
Застосування	Десктоп і веб-додатки	Веб-додатки	Корпоративні та десктопні програми	Веб і наукові обчислення
Фреймворки та інструменти	ASP.NET, .NET Core	Laravel, Symfony	Spring, Struts	Django, Flask, SciPy
Об'єктно-орієнтоване програмування	Так	Так	Так	Так
Функціональне програмування	Так	Так	Так	Так
Типізація	Сильна, статична	Слабка, динамічна	Сильна, статична	Сильна, динамічна
Мультитредінг	Так	Так	Так	Так
Мобільна розробка	Xamarin, Unity	Обмежена	SDK для Android	Ківі, Pygame
Десктопна розробка	Так	Ні	Так	Так
Масштабованість	Хороша	Хороша	Хороша	Хороша
Легкість налагодження	Легка	Легка	Легка	Легка
Аналітика / Дані Наука	Обмежена	Обмежена	Обмежена	Сильна підтримка

– Популярність: згідно з індексом TIOBE за березень 2025 року [6], Python наразі є найпопулярнішою мовою програмування, за нею йдуть Java, PHP та .NET. Однак популярність варіюється залежно від регіону та галузі (таблиця 2.8).

– Продуктивність: коли йдеться про порівняння продуктивності .NET, Python, PHP та Java, Java та .NET добре відомі своєю хорошою продуктивністю, за ними йдуть PHP, а потім Python. Важливо пам'ятати, що продуктивність може змінюватися залежно від програми та реалізації.

– Легкість навчання: Python відомий своєю простотою та читабельністю, що спрощує вивчення новачками. Крива навчання PHP є відносно плоскою порівняно з Java та .NET.

– Масштабованість: Python та PHP є наступними за масштабованістю мовами програмування після Java та .NET, які можуть керувати великомасштабними веб-додатками.

Таблиця 2.8 – Зміни в популярності мов програмування для веб-розробки

Червень 2025	Червень 2024	Зміна	Мова програмування	Рейтинг	Зміна рейтингу
1	1		Пітон	25,87%	+10,48%
2	2		C++	10,68%	+0,65%
3	3		C	9,47%	+0,24%
4	4		Java	8,84%	+0,44%
5	5		C#	4,69%	-1,96%
6	6		JavaScript	3,21%	-0,11%
7	7		Go	2,28%	+0,35%
8	9	↑	Visual Basic	2,20%	+0,54%
9	11	↑	Delphi/Object Pascal	2,15%	+0,62%
10	10		Fortran	1,86%	+0,33%
11	25	↑	Ada	1,70%	+0,91%
12	8	↓	SQL	1,55%	-0,21%
13	27	↑	Perl	1,47%	+0,77%
14	21 рік	↑	R	1,39%	+0,43%
15	15		PHP	1,25%	+0,03%

– Безпека: Java або .NET рекомендуються для створення безпечних онлайн-додатків через їхні добре відомі надійні функції безпеки. Хоча вони мають певні функції безпеки, PHP та Python потенційно можуть потребувати додаткових заходів безпеки.

– Підтримка спільноти: розробникам простіше знаходити рішення проблем і отримувати допомогу за потреби, оскільки кожна з чотирьох мов має значну та активну спільноту розробників, які активно сприяють розвитку мови.

Загалом, кожна мова має переваги та недоліки. Python та PHP підходять для керованих даними та невеликих застосунків, тоді як Java та .NET чудово підходять для створення масштабних та критично важливих для продуктивності онлайн-застосунків.

Однак кожна з чотирьох мов програмування, PHP, ASP.NET, Python та Java, має потужний набір бібліотек, фреймворків та інструментів, які допомагають спростити та підвищити продуктивність веб-розробки.

Успіх проєкту залежить від обраної мови програмування для веб-розробки. А такі фактори, як продуктивність, масштабованість та безпека, відіграють вирішальну роль у визначенні того, яка мова є правильним вибором для проєкту розробки.

Але цього порівняння недостатньо. Проаналізуємо фреймворки та стеки на основі цих мов програмування.

2.6 Порівняння MEAN Stack проти PHP проти .NET

Вибір правильного технологічного стеку для проєкту веб-розробки може бути складним завданням. З такою кількістю доступних варіантів може бути важко зрозуміти, який з них найкраще відповідає потребам.

2.6.1 Огляд PHP

PHP – одна з найпопулярніших мов сценаріїв в інтернеті. PHP легше вивчати, і саме тому він такий надзвичайно популярний. Мова має велику спільноту розробників, що робить її кращою мовою для розробки. Деякі дослідження показують, що мова програмування встановлена на понад 244 мільйонах веб-сайтів. Мова сценаріїв є безкоштовною та може похвалитися низкою фреймворків, які значно спрощують процес розробки.

PHP – це мова розробки з високою масштабованістю, і підтвердженням цього є той факт, що Facebook та MySpace були розроблені з використанням PHP.

Взаємодія між скриптом, базою даних і сервером визначає продуктивність веб-застосунку. За стеком PHP слідує Lamp, який включає Apache, MySQL, PHP та Linux. Це дуже популярний стек, якому широко довіряють за покращену продуктивність.

Згідно з нещодавнім звітом [8], PHP займав 36% частки у створенні 1 мільйона веб-сайтів (див. рис. 2.3).

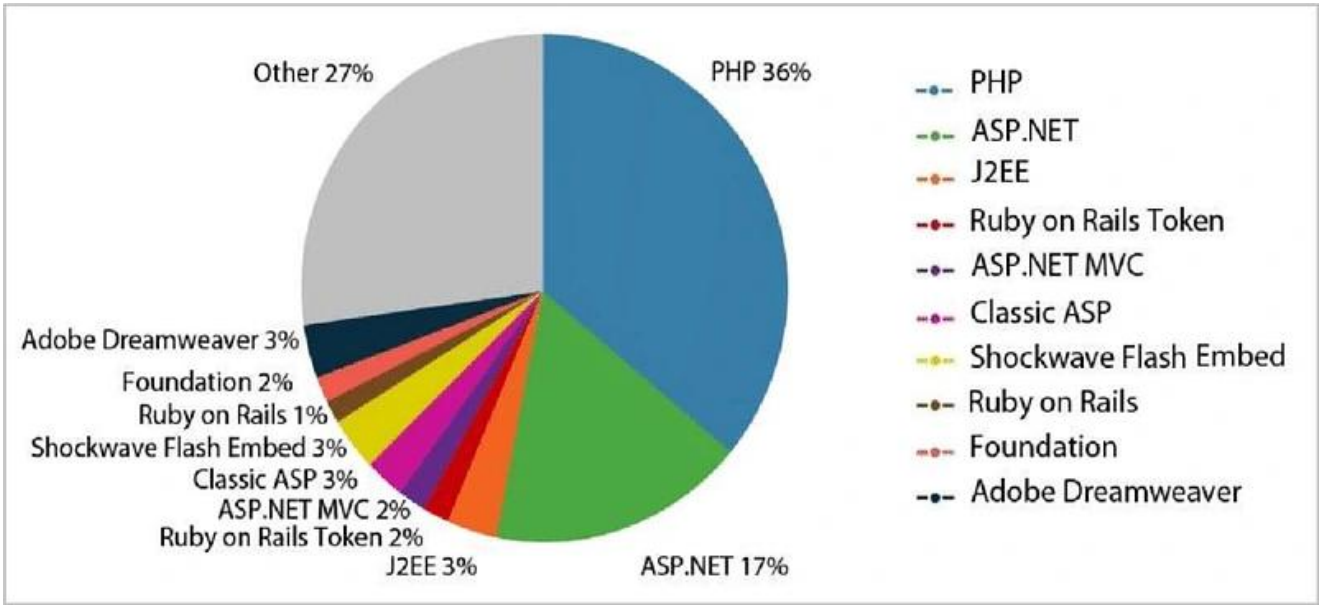

Рисунок 2.3 – Мови програмування для веб-розробки

У цьому розділі буде порівняно MEAN Stack, PHP та .NET.

Деякі ключові переваги розробки на PHP:

1. PHP може обробляти веб-сайти з великим трафіком.
2. Фреймворк є платформонезалежним і може працювати з усіма видами операційних систем.
3. Фреймворк підтримує всі основні веб-сервери.
4. Допомагає в розробці веб-додатків та сайтів, які працюють швидше – використовує власну пам'ять та зменшує час завантаження.

Має велику спільноту розробників, і можна легко отримати підтримку.

2.6.2 Огляд стеку MEAN

MEAN Stack складається з MongoDB, Express.js, AngularJS та Node.js. Ключовою перевагою MEAN Stack над LAMP є його простота та спільна структура. Зберігання даних стає простим та гнучким за допомогою Mongo DB. Запуск сервера спрощено завдяки Node.js. Процес розробки веб-сайту стандартизовано за допомогою Express. js. Інтерактивні функції можна додавати за допомогою Angular.js. Усе це разом робить переміщення даних простішим та швидшим.

Переваги бази даних Mongo.

1. Mongo DB – один із найкорисніших шарів баз даних
2. Це сучасна база даних, яка забезпечує повну підтримку кластерів та автоматичний шардинг.
3. База даних дозволяє легко розробляти, тестувати та розміщувати сучасні додатки в хмарі

Переваги NodeJS.

LAMP має обмеження, пов'язані з багатьма конфігураційними файлами; Node JS не стикається з подібною проблемою. Запити маршрутизації додатків та відповіді на запити стають простими за допомогою Javascript.

Кількість шарів, які містить LAMP, може бути складною для багатьох розробників. NodeJS об'єднує всі види процесів в одному місці. Наявність усього в одному шарі значно спрощує процес.

Переваги JSON.

1. JSON допомагає забезпечити плавний потік даних через шари
2. У випадку MEAN, скрізь використовується певний формат даних у JSON
3. Це економить час, оскільки ці дані проходять через кожен шар
4. Оскільки JSON присутній скрізь у MEAN, робота із зовнішніми API стає простішою.

Збереження місця на диску.

Команда JOIN допомагає значно заощадити місце на диску. Це допомагає звільнити місце на диску, видаляючи додаткові поля. Це допомагає підтримувати базу даних у належному стані та необхідний простір на диску.

2.6.3 Огляд .NET

Це популярний фреймворк від Microsoft для створення веб-сайтів та веб-додатків. MSSQL виступає в якості бази даних для ASP.NET, хоча MySQL також можна використовувати. Фреймворк .NET побудовано на CLR та здебільшого

використовує бібліотеки .NET Framework. Бібліотеки включають Windows, ASP.NET, Windows Presentation Foundation та ADO.NET.

Динамічна підтримка веб-сайтів, веб-вимоги та програмне забезпечення для робочого столу – це деякі з особливостей ASP.NET.

Ось деякі ключові переваги розробки за допомогою .NET.

1. Кількість коду, необхідного для створення застосунків у .NET, значно зменшується завдяки .NET.

2. Оскільки Windows забезпечує підтвердження для програм, розроблених у .NET, фреймворк досить безпечний для роботи.

3. Фреймворк також надає детальний механізм перетягування.

4. Фреймворк дозволяє легко вносити зміни в ASP.NET, оскільки вихідний код та HTML-код програми є окремими.

5. Платформа не залежить від мови.

Детальне порівняння фреймворків для веб-розробки наведено в таблиці 2.9 та описано в тексті нижче.

Швидкість і продуктивність.

Коли йдеться про швидкість та продуктивність, MEAN Stack є явним переможцем. MEAN Stack побудовано на Node.js, яка є набагато швидшою платформою, ніж PHP або .NET. Як результат, додатки MEAN Stack працюють набагато швидше, ніж додатки PHP або .NET.

Простота використання.

MEAN Stack також є найпростішою у використанні з цих трьох технологій. PHP та .NET вимагають багато налаштувань та конфігурацій, перш ніж ви зможете почати розробку з їх допомогою. З іншого боку, MEAN Stack постачається з усім необхідним для початку роботи одразу після розпакування.

Безпека.

Усі три ці технології є безпечними, але PHP та .NET мають дещо більше функцій безпеки, ніж MEAN Stack. PHP та .NET мають вбудовані функції безпеки, які допомагають захистити ваші програми від атак.

Масштабованість.

MEAN Stack – це набагато масштабованіша платформа, ніж PHP або .NET. За допомогою MEAN Stack ви можете легко додавати нові функції та можливості до своєї програми, не переписуючи всю кодову базу. Як результат, програми MEAN Stack набагато легше масштабувати, ніж програми PHP або .NET.

Таблиця 2.9 – Детальне порівняння MEAN Stack, PHP, .NET

Критерії порівняння	PHP	MEAN стек	.NET
Тип коду	Інтерпретований код	Node.js використовується для написання серверного коду на JavaScript	Компільований код
Безпека	Менш ефективний у створених функціях безпеки	Висока	Велика кількість вбудованих можливостей
Призначення	Малий та середній бізнес	Середні та великі рішення	Середні та великі рішення
Вартість	Низькі витрати на обслуговування	Доступне у реалізації	Вищі витрати на хостинг
Підтримка	Спільнота та Zend Technologies	Помітно менша спільнота	Майкрософт
Масштабованість	Горизонтальна масштабованість	Висока	Вертикальна масштабованість

Вартість.

MEAN Stack – це набагато дешевша платформа, ніж PHP або .NET. За допомогою MEAN Stack ви можете розробляти свої програми, використовуючи програмне забезпечення з відкритим кодом, яке є безкоштовним у використанні. Як результат, програми MEAN Stack набагато дешевші в розробці та підтримці, ніж програми на PHP або .NET.

Підтримка.

PHP та .NET мають більшу підтримку, ніж MEAN Stack. PHP та .NET – це добре зарекомендували себе технології з великою спільнотою розробників, які можуть запропонувати підтримку та керівництво. MEAN Stack, з іншого боку, – це відносно нова технологія з меншою спільнотою розробників.

Після порівняння MEAN Stack, PHP та .NET стає зрозуміло, що MEAN Stack є найкращим вибором для більшості проектів веб-розробки.

MEAN Stack швидший, простіший у використанні та масштабованіший, ніж PHP або .NET. Крім того, MEAN Stack є дешевшою платформою для розробки та підтримки, ніж PHP або .NET. Однак PHP та .NET мають більшу підтримку, ніж MEAN Stack.

PHP та .NET – найпопулярніші технології з великою спільнотою розробників, які можуть запропонувати підтримку та керівництво. MEAN Stack, з іншого боку, – це відносно нова технологія з меншою спільнотою розробників.

Зрештою, найкращий вибір залежатиме від ваших конкретних потреб та уподобань. Якщо потрібна швидка, проста у використанні та масштабована платформа, то MEAN Stack – найкращий вибір. Однак, якщо потрібна платформа з більшою кількістю функцій безпеки та підтримки, PHP або .NET можуть бути кращим вибором.

Таким чином виберемо мову PHP та СКБД MySQL.

3 РОЗРОБКА ВЕБ-ПЛАТФОРМИ

3.1 Проєктування бази даних

Як було вказано у попередньому розділі для нашої веб-платформи вибираємо сервер баз даних MySQL.

Опишемо потрібні нам таблиці з їх первинними ключами та зв'язками. Кожну таблицю описуватимемо за її назвою з наступним вказанням первинного ключа, основних атрибутів (стовпців) та зв'язки з іншими таблицями. Вкажемо також тригери, що реалізовуватимуть бізнес-логіку.

1. user – таблиця для зберігання інформації про користувачів системи.

UserId — первинний ключ.

Містить дані користувача: ім’я, прізвище, email, пароль, валюта.

Зв’язки:

- Один user → багато account.
- Один user → багато assets, bills, budget, category, totals.

2. account – таблиця для зберігання відомостей про фінансові рахунки.

AccountId — первинний ключ.

UserId — зовнішній ключ → user(UserId).

AccountName — назва рахунку (наприклад, "Card", "Cash").

Зв’язки:

- Один account → багато assets, bills, totals.

3. assets – інформація про доходи.

AssetsId — первинний ключ.

UserId, AccountId, CategoryId — зовнішні ключі.

Містить інформацію про доходи: назва, дата, сума, опис.

Зв’язки:

- AccountId → account.
- UserId → user.
- CategoryId → category.

4. bills – рахунки, що підлягають оплаті.

BillsId — первинний ключ.

Містить інформацію про витрати.

Аналогічна структура до assets.

Зв’язки:

- AccountId → account.
- UserId → user.
- CategoryId → category.

5. budget – таблиця з інформацією про заплановані витрати на певні періоди часу.

BudgetId — первинний ключ.

UserId, CategoryId — зовнішні ключі.

Зберігає заплановану суму бюджету на категорію в обраний день.

6. category – категорії витрат/доходів (наприклад, "Їжа", "Зарплата").

CategoryId — первинний ключ.

Має поле Level (для вкладеності або важливості).

7. totals – актуальні суми коштів на кожному рахунку. Автоматично оновлюється через тригери під час вставки/зміни записів у assets та bills.

TotalsId — первинний ключ.

UserId, AccountId — зовнішні ключі.

Поле Totals — актуальна сума коштів на рахунку.

Тригери (автоматичні обробники подій) створені для відповідних таблиць.

При створенні нового користувача → автоматично створюються три рахунки: Cash, Account, Card (GenerateDefaultAccount)

При додаванні рахунку → створюється запис у totals зі значенням 0 (GenerateAccount)

При додаванні доходу (assets) → сума додається до totals.

При додаванні витрати (bills) → сума віднімається з totals.

При оновленні доходу → totals перераховується заново.

Головні зв’язки:

user(UserId) → account(UserId), assets(UserId), bills(UserId), budget(UserId), category(UserId), totals(UserId)

account(AccountId) → assets(AccountId), bills(AccountId), totals(AccountId)

category(CategoryId) → assets(CategoryId), bills(CategoryId), budget(CategoryId)

Схема створеної бази даних наведена на рисунку 3.1. SQL-скрипт її створення наведено в додатках до роботи.

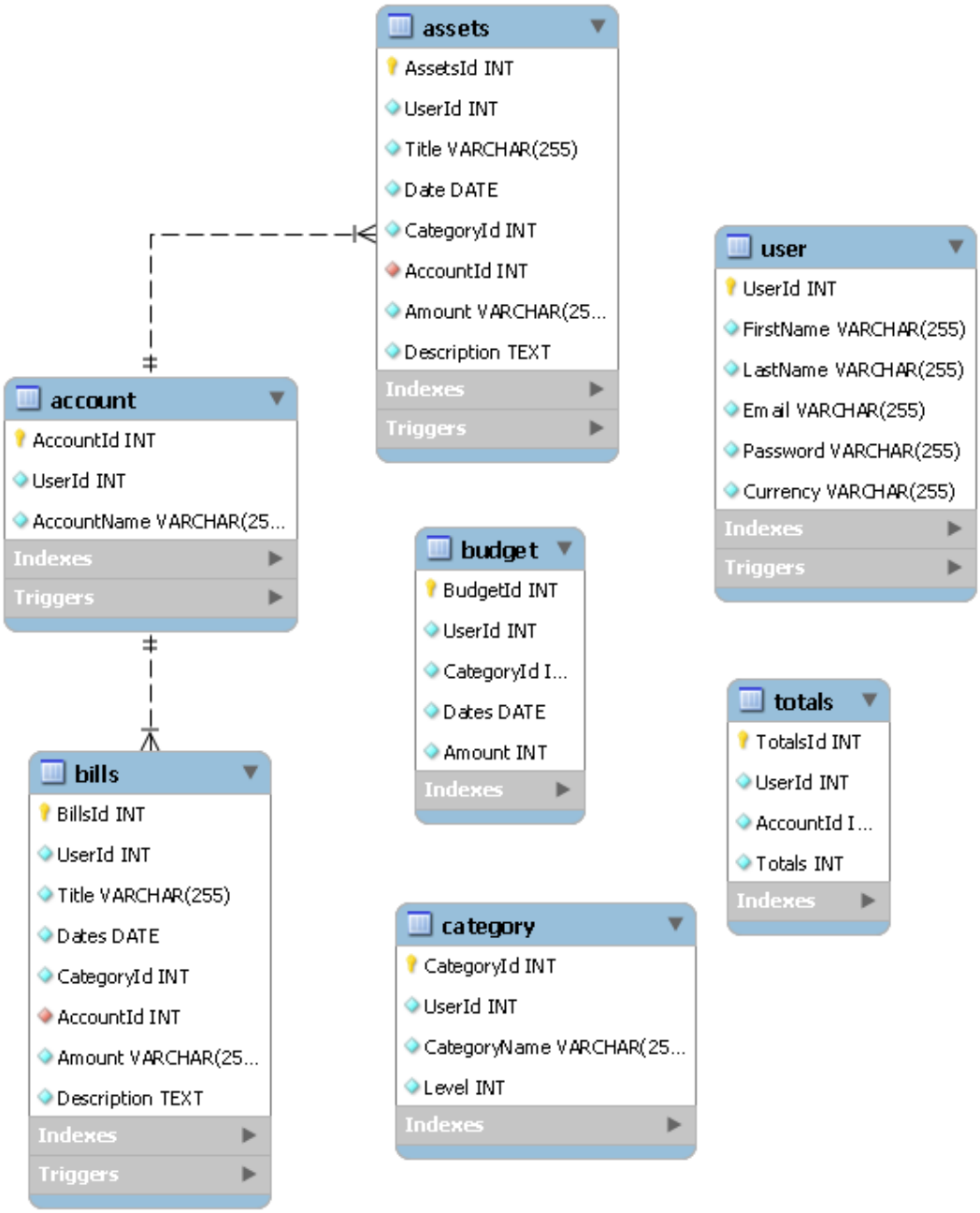

Рисунок 3.1 – Схема створеної бази даних

ЇЇ отримано на основі SQL запитів на створення бази даних, таблиць та тригерів з використанням Reverse Engineering в MySQL Workbench. Цей процес показано на рисунках 3.2 – 3.4

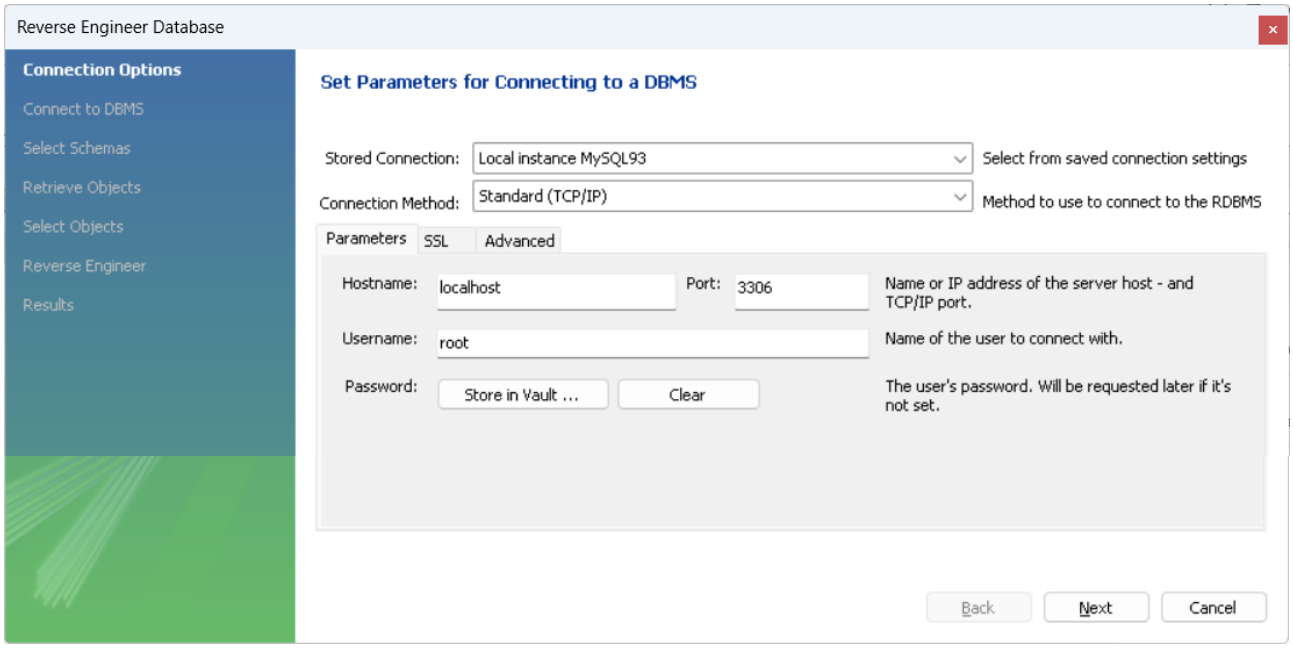

Рисунок 3.2 – Створення схеми бази даних, крок 1

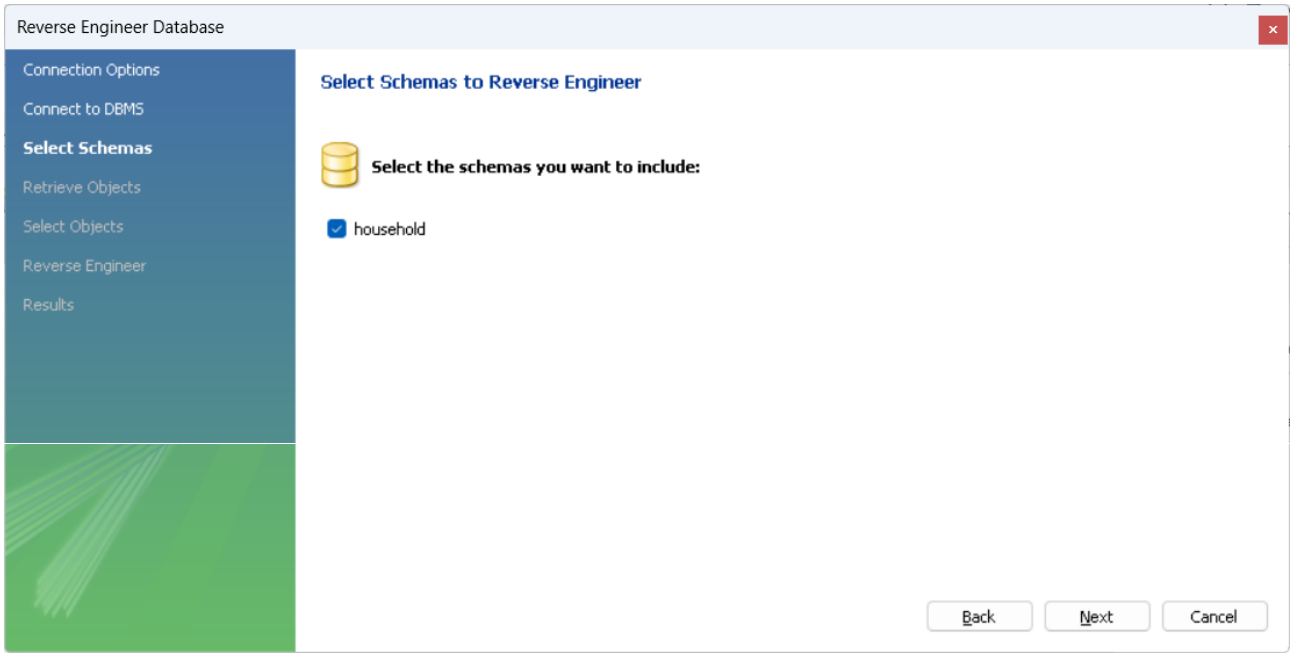

Рисунок 3.3 – Створення схеми бази даних, крок 2

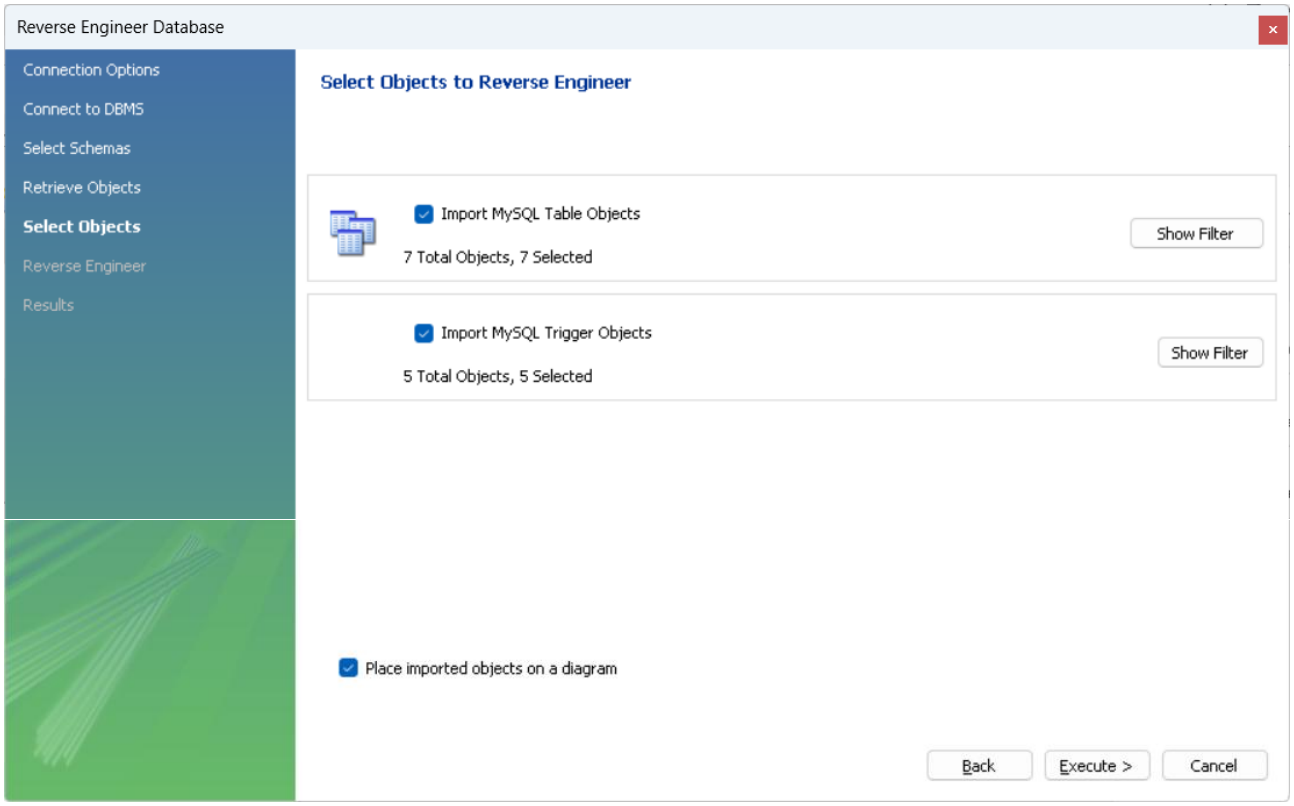

Рисунок 3.4 – Створення схеми бази даних, крок 3

Таким чином наша база даних готова до використання в програмі.

3.2 Розробка програмної частини веб-застосунку для планування бюджету

Перейдемо тепер до опису власне програмної частини нашого веб-застосунку для планування та управління домашнім бюджетом.

Ідеєю розробки було створення простого, але потужного PHP-скрипта для керування витратами родини з багатокористувацьким рівнем доступу та відповідними дозволами для кожного користувача.

PHP-менеджер витрат, розроблений, щоб допомогти індивідуальним або бізнес-особам складати бюджет, відстежувати та, можливо, контролювати витрати. Він підтримує відстеження як витрат, так і доходів. Ця система управління витратами надає інтегрований набір функцій, які допоможуть керувати витратами та грошовими потоками. Вона надає можливість групувати доходи/витрати за категоріями та дозволяє встановити бюджет і відстежувати витрати за категоріями.

Основний функціонал веб-застосунку показано на рисунку3.5.

Основні функціональні вимоги до системи можна подати у вигляді списку:

– керування витратами;
– керування доходами;
– керування категоріями витрат;
– керування категоріями доходів;
– фільтрування за діапазоном дат;
– сортування та нумерація сторінок;
– підтримка багатокористувацького рівня для керування системою;
– включено вхід та керування користувачами;
– встановлення різних дозволів для різних типів користувачів;
– можна додати власні CRUD-запити;
– легке налаштування.

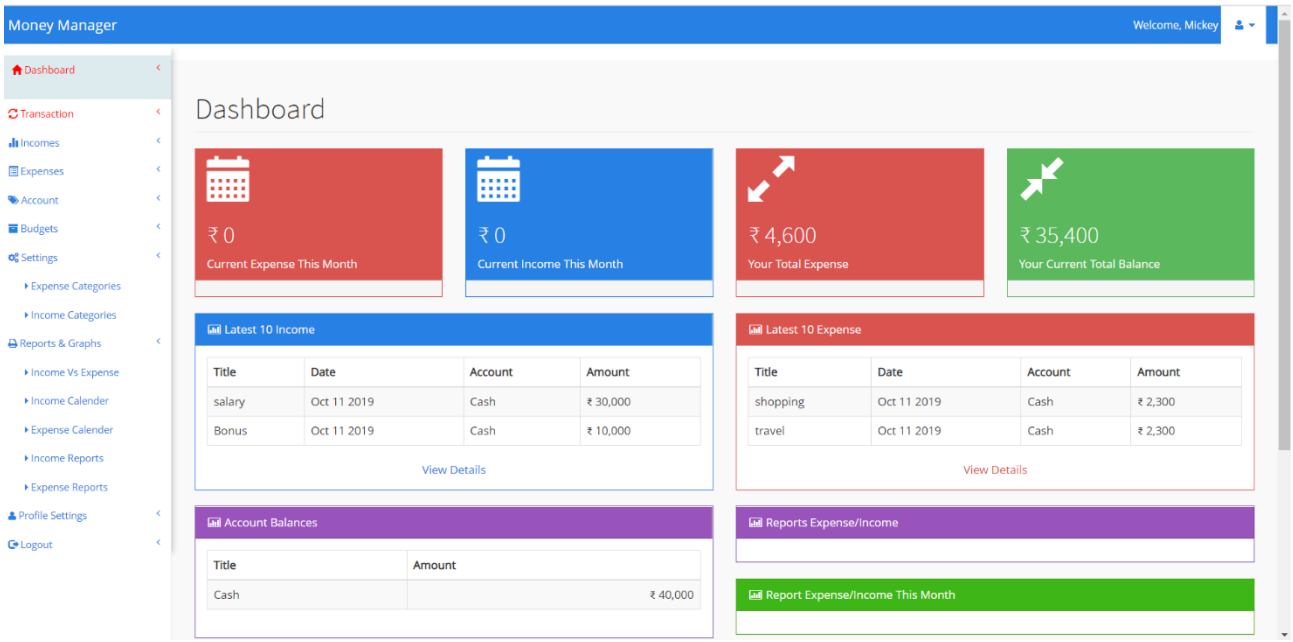

Рисунок 3.5 – Головне вікно веб-застосунку управління сімейним бюджетом

Як бачимо, сформульовані вище функціональні вимоги відображені у лівому меню. Основна частина сторінки призначена для виводу інформації та елементів керування відповідного пункту меню.

Основна бізнес-логіка реалізована у вигляді окремих php-файлів, які записані в папку pages репозиторію проєкту на GitHub (див. рис. 3.6).

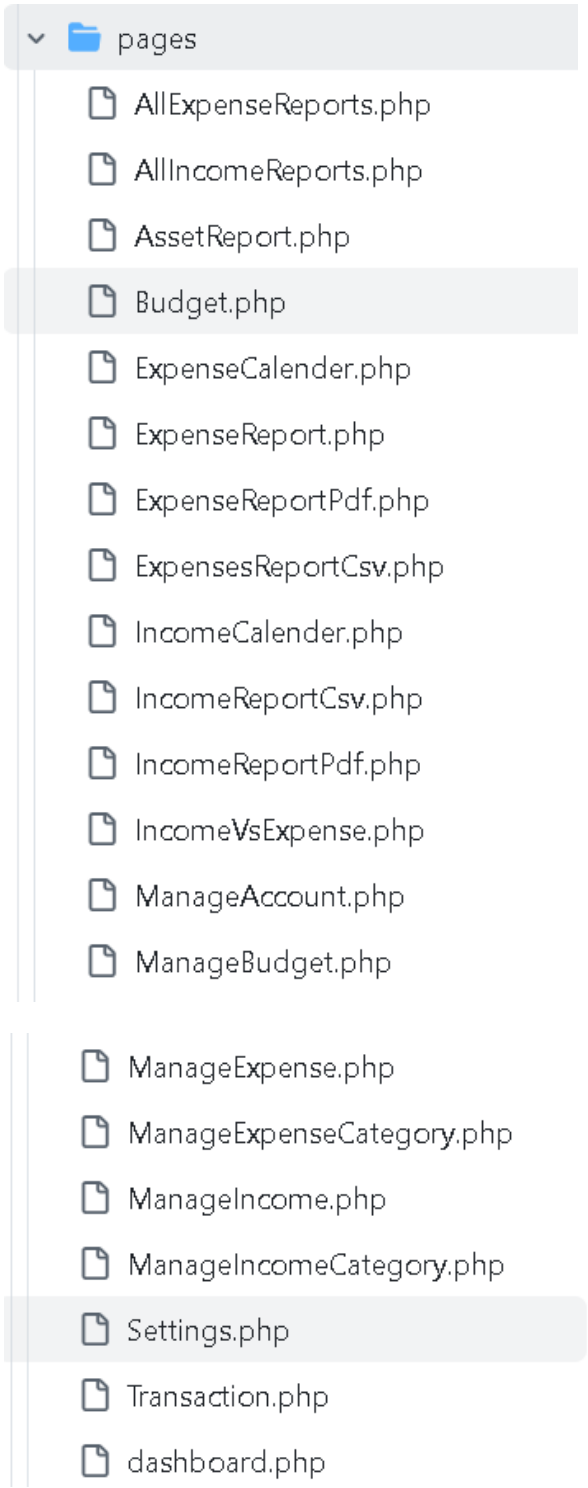

Рисунок 3.6 – Файли з реалізацією бізнес-логіки

Крім того, в кореневому каталозі репозиторію розміщені файли, показані на рис. 3.7.

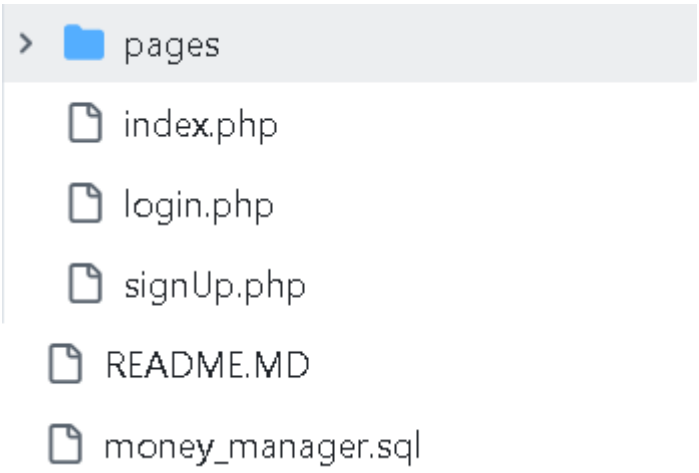

Рисунок 3.7 – Файли кореневого каталогу репозиторію

Опишемо коротко ці файли.

- index.php – файл головної сторінки;
- login.php – файл реалізації входу зареєстрованого користувача в систему;
- signUp.php – файл сторінки, що реалізує функцію реєстрації користувача системи;
- money_manager.sql – SQL-скрипт створення бази даних та її елементів.

Крім того, опишемо основні файли реалізації бізнес-логіки основних функцій.

1. dashboard.php

Головна сторінка після входу – відображає зведені дані по доходах, витратах, залишках, графіки тощо.

2. login.php

Відповідає за аутентифікацію користувача: обробка форми входу, перевірка облікових даних, створення сесії, ініціалізація стандартних категорій при першому вході, перенаправлення на index.php (на dashboard). Код містить інклюди для бази даних, сесії та функцій .

3. ExpenseReportPdf.php

Формує PDF-звіт витрат за допомогою бібліотеки TCPDF. Підвантажує дані з таблиць bills, category, account, фільтрує за параметром GET (filter), будує HTML-таблицю, генерує фінальний PDF та відправляє його до користувача для завантаження.

4. Settings.php

Файл налаштувань програми – відповідає за зміну конфігураційних параметрів: інформація користувача, валюта, структура категорій тощо.

5. Transaction.php

Файл відповідає за управління транзакціями: додавання/редагування/видалення доходів і витрат, відображення списку транзакцій, можливо фільтрація та пагінація (розбиття на сторінки при виводі) даних.

Програмний код цих файлів наведено у додатках до роботи.

На рисунку 3.8 показано вікно входу в систему, як результат виконання файлу login.php кореневого каталогу репозиторію.

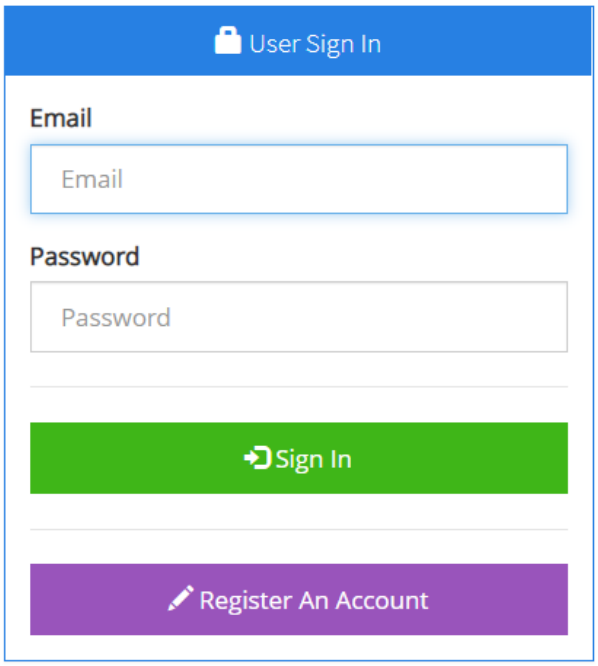

Рисунок 3.8 – Вікно входу в систему

Після входу користувач потрапляє на головну сторінку, зображену на рисунку 3.5.

Якщо користувач не зареєстрований, то він, натиснувши відповідну кнопку з вікна на рисунку 3.8, потрапляє на сторінку реєстрації, зображену на рисунку 3.9.

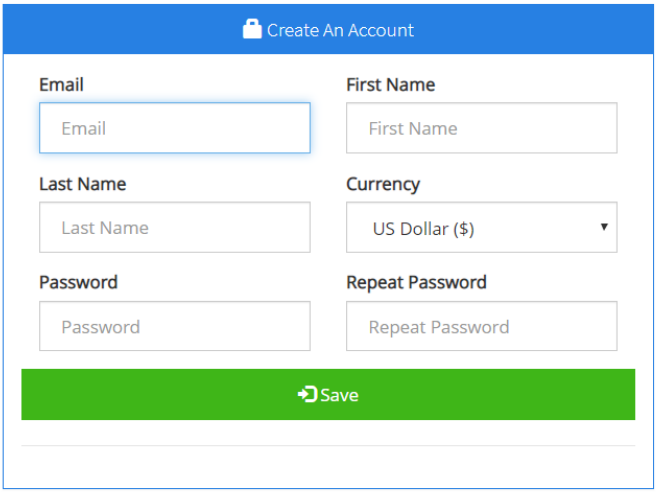

Рисунок 3.9 – Сторінка реєстрації нового користувача

Таким чином вважатимемо процес розробки програми завершеним.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Аналіз небезпеки і шкідливості при розробці програмного забезпечення

Організація робочого місця розробника ПЗ впливає на його працездатність.

У своїй діяльності розробник використовує комп’ютер, пристрої збереження інформації, а тому є необхідність забезпечення зручного доступу до всіх технічних засобів. Тому в даному розділі докладніше розглянемо відомості про систему ергономічних норм і принципів організації робочого місця, на котрому проводяться роботи зі створення модуля збору статистики.

Під робочим місцем розуміється зона, оснащена необхідними технічними засобами, у якій відбувається трудова діяльність виконавця або групи виконавців, які спільно виконують одну роботу або операцію.

Організація робочого місця полягає у виконанні заходів, які забезпечують безпечний і раціональний трудовий процес і ефективне використання знарядь та предметів праці, що підвищує продуктивність праці і знижує стомлюваність працівника.

Організація робочого місця залежить від характеру розв'язуваних задач і особливостей предметно-просторового оточення, що визначають робоче положення тіла і можливість пауз для відпочинку, типи і способи засобів відображення і керування, необхідність у засобах захисту, спецодягу, простору для налагодження і ремонту устаткування.

Одним з компонентів діяльності на робочому місці є робочі рухи. Їхня раціональна організація створює умови для зниження стомлення, резерви для підвищеної працездатності. Просторові характеристики руху оператора визначаються траєкторіями руху і розмірами моторного поля (зони досяжності).

При організації робочого місця необхідно забезпечити нормальні умови огляду. Зону огляду описує кут, вершина якого знаходиться в центрі ока, а

сторони складають границі, в яких людина при фіксованому положенні голови й ока добре розрізняє їхнє місцезнаходження.

У горизонтальній площині цей кут складає 300 – 400. При організації робочого місця кут огляду можна взяти 500 – 600, включаючи зону менш ясного огляду. Допустимий кут огляду по горизонталі 900. У вертикальній площині оптимальний кут огляду 100 вверх і 300 вниз від лінії погляду, а допустимий 300 вгору і 400 вниз від лінії погляду.

Щоб зберегти нормальну гостроту зору, робочу поверхню розташовують від очей на відстані від 0,3 м до 0,75 м. Робочі меблі повинні бути зручними для виконання робочих операцій. В даному випадку робочий стіл є основним устаткуванням. Особливо важливе значення має висота столу, його конструкція, яка повинна передбачати шухляди для розміщення інструментів, документації.

Важливе значення має конструкція робочих крісел. Погано підібрані крісла можуть бути причиною надмірної стомлюваності.

Нахил і висота крісла повинні регулюватися відповідно до висоти робочої поверхні і росту працюючого. Рекомендована ширина крісла 370 – 400 мм, глибина 370 – 420 мм, висота спинки 370 – 1000 мм від рівня крісла. Для розміщення ніг необхідно передбачити вільний простір під робочою площиною [24].

Праця людини, що протікає в умовах надмірного нервово-емоційного напруження, довготривалих статичних навантажень, обмеженої рухової активності призводить до неврозів, відхилень у психіці, захворювань опорно-рухового апарату, серцево-судинної системи тощо. Комп'ютери, телебачення, системи зв'язку та інші засоби, що використовують досягнення радіоелектроніки, є генераторами цілої низки електромагнітних випромінювань, вплив яких на організм людини ще не зовсім вивчений.

З широким впровадженням автоматизації та комп'ютеризації виникла потреба врахування психологічних можливостей людини, таких як швидкість реакції, особливості пам'яті та уваги, емоційний стан та ін. Поява операторської діяльності призвела до суттєвих змін у фаховій структурі праці. Зменшились

фізична важкість праці, ризик виробничого травматизму, однак разом з тим, на працюючу людину посилюється вплив нових, раніше не відомих чи мало вивчених несприятливих виробничих факторів фізичного, хімічного і особливо психофізіологічного характеру.

Проте, розвиток сучасної обчислювальної техніки відбувається не лише у бік поктащення її технічних параметрів, але також звертається увага безпеку використання цієї техніки людиною шляхом зменшення потужності випромінювачів, зменшенням рівня випромінювання з моніторів, зменшення напруг живлення, покращення ергономічних характеристик.

Таким чином, в розділі з охорони праці виконано огляд питань безпечної роботи при створенні сайту та встановлено, що умови такої роботи відповідають вимогам з охорони праці, які застосовуються в галузі інформаційних технологій.

4.2 Інформаційно-психологічні небезпеки

Сучасні реалії постіндустріального суспільства, зумовлені значним ростом інформації, відкривають ще одну сферу життєдіяльності людини – інформаційну. Сучасні засоби комунікації і обробки інформації створили принципово нові умови існування людини, що зумовило появу грандіозного проекту об'єднання національних інформаційних і телекомунікаційних структур в глобальну інформаційну інфраструктуру.

Життєдіяльність людини реалізується одночасно зі світом природи і у специфічному для людського суспільства інформаційному середовищі, що має свої закономірності розвитку і функціонування. Інформаційна сфера стає такою ж важливою складовою суспільного життя, як економічна, виробнича, побутова, політична, військова та ін. Нові інформаційні технології, засоби масової комунікації багатократно підсилили можливості впливу на свідомість і підсвідомість як окремої людини, так і на великі групи людей та населення країни загалом.

Інформаційна сфера – сукупність таких елементів:

– об'єкти інформаційної взаємодії чи впливу;

– особисто інформація, призначена для використання суб'єктами інформаційної сфери;

– інформаційна інфраструктура, що забезпечує можливість здійснення обміну інформацією між суб'єктами;

– суспільні відносини, що складаються у зв'язку з формуванням, переданням, розповсюдженням і збереженням інформації.

Особистість, активний соціальний суб'єкт, його психіка піддаються безпосередньому впливу інформаційних чинників (передумов, що чинять опір чи утруднюють формування і функціонування адекватної інформаційно-орієнтуючої основи суспільної поведінки людини (життєдіяльності у суспільстві)), які трансформуються, через його поведінку, діяльність (бездіяльність), здійснюють деструктивний, дисфункційний вплив на його життєдіяльність.

До основних загроз інформаційно-психологічної безпеки відносять можливість настання негативних наслідків для суб'єктів, що піддаються інформаційно-психологічному впливу, які виражаються в таких формах:

– нанесення шкоди здоров'ю людини;

– блокування на неусвідомленому рівні волі, волевиявлення людини, штучне привиття їй синдрому залежності;

– втрата здатності до політичної, культурної, моральної самоіндентифікації людини;

– маніпуляція суспільною свідомістю;

– руйнування єдиного інформаційного і духовного простору України, традиційних устроїв суспільства і суспільної моральності, а також порушення інших життєво важливих інтересів особистості, суспільства, держави.

Наприклад, культ жорстокості, насильства, порнографії, розбещеності тощо, які пропагують у засобах масової інформації, друкованих виданнях, комп'ютерних іграх, мережі Інтернет веде до неусвідомленого бажання у підлітків і молоді, а також дорослих з нестійкою психікою, копіювати запропоновані моделі поведінки. Цей вид пропаганди знижує рівень порогових

обмежень і правових заборон, що поряд з іншими умовами відкриває шлях для багатьох правопорушень. Це своєю чергою наносить непоправну шкоду не тільки окремій особистості, але й суттєві збитки національним інтересам країни.

Отже, джерелом інформаційно-психологічної небезпеки є та частина інформаційного середовища, яка через визначені причини неадекватно відображає реалії, вводить в оману людину, засліплює її ілюзією.

Інформаційно-психологічні загрози зумовлені розробкою, виготовленням, розповсюдженням та використанням суб'єктами негативних інформаційно-психологічних впливів, спеціальних засобів і методів такого впливу.

Концепція інформаційно-психологічної безпеки.

Сучасне розуміння безпеки в контексті врахування відношення інтересів особистості, суспільства і держави висуває завдання розгляду нового аспекту цієї проблеми – безпеки в інформаційній сфері життєдіяльності людини, тобто інформаційно-психологічної безпеки.

В інформаційному середовищі, що є складовим системним утворенням, виділяється процесуальна складова як найбільш динамічна і змінна її частина – інформаційно-комунікативні процеси, які активно впливають на індивідуальну, групову і суспільну психологію (індивідуальну, групову, масову свідомість). Маніпулюючи станом інформаційного середовища, змінюється стан духовної сфери суспільства, деформація і деструктивні зміни якої у формі психоемоційної і соціальної напруженості, спотворених норм і неадекватних соціальних стереотипів і установок, оманливих і неприродних орієнтацій та цінностей. Це своєю чергою впливає на стан і процеси у всіх основних сферах суспільного життя, в тому числі політичній і економічній.

Вперше у пострадянському просторі про проблему інформаційно-психологічної безпеки було зазначено в листопаді 1995 р. на науково-практичній конференції, організованій Інститутом психології Російської академії наук. На цій та подальших конференціях було розкрито роль знання технологій інформаційно-психологічного впливу, метою якого є маніпуляція, для

вироблення напрямів реформування психологічного захисту особистості і особистої інформаційно-психологічної безпеки.

Інформаційно-психологічну безпеку особистості визначають такими основними причинами.

Зростання тиску інформаційного середовища визначає необхідність формування нових механізмів та засобів виживання людини як особистості й активного соціального суб'єкта у сучасному суспільстві.

Взаємодія психіки людини з інформаційним середовищем відрізняється якісною специфікою і не має аналогів у комунікації інших біологічних, технічних, соціальних і соціотехнічних структур.

Основною і центральною "мішенню" інформаційного впливу є людина, її психіка.

Отже, інформаційно-психологічну безпеку можливо розглядати як стан захищеності особистості, різних соціальних груп і об'єднань людей від дій, впливів, які здатні проти їхньої волі і бажання змінити психічні стани та психологічні характеристики людини, модифікувати її поведінку і обмежувати свободу вибору, зумовило потребу переосмислення інформаційної взаємодії, а також деяких інших соціально-психологічних процесів і явищ у сучасному суспільстві.

Інформаційно-психологічна безпека – стан захищеності окремих осіб чи груп осіб від негативних інформаційно-психологічних впливів і пов'язаних з цим інших життєво важливих інтересів особистості, суспільства, держави в інформаційному середовищі.

Негативний інформаційно-психологічний вплив – процес зміни психічних станів і характеристик людей під впливом інформаційно-комунікативних процесів як динамічного компонента інформаційного середовища. Цей вплив спрямований на людину чи групу осіб (у тому числі без їхньої згоди) з метою примусу до визначеної поведінки, оцінки ситуації, керування та корекції індивідуальної та колективної свідомості. Він здійснюється з використанням

спеціальних засобів і методів впливу на психіку людини, унаслідок чого він приводить до негативних наслідків для особистості, суспільства і держави.

Спеціальні засоби впливу – технічні і програмні засоби, що використовують для використання з метою негативного інформаційно-психологічного впливу на людину чи групу людей.

Спеціальні методи впливу – послідовність прийомів впливу на психіку людини, використання яких приводить до негативних наслідків для особистості, суспільства та держави.

Головним об'єктом забезпечення інформаційно-психологічної безпеки в інформаційному середовищі у сфері індивідуальної безпеки є усвідомлення інформації, здатність людини адекватно сприймати навколишню дійсність, своє місце в зовнішньому світі, формувати відповідно до свого життєвого досвіду визначені переконання і приймати стосовно них рішення.

Інформаційно-психологічна безпека має спиратися на стандарти інформаційно-психологічної безпеки – затверджені у визначеному порядку інформаційно-психологічного впливу, який не викликає негативних наслідків для психіки людини.

ВИСНОВКИ

Сьогодні вміння керувати власними фінансами – це не просто навичка, а життєва потреба. Робити це вручну – довго, незручно і легко щось упустити. Саме тому автоматизовані рішення для обліку витрат та доходів стають дедалі популярнішими – вони допомагають краще розуміти свої фінанси та не витрачати зайвого часу.

Аналіз існуючих застосунків, таких як Mint чи YNAB, показав, що ринок переповнений пропозиціями. Та багато з них орієнтовані на іноземного користувача, потребують підписок або не дають змоги адаптувати функції під себе. Це лише підкреслює, що створення власного вебсервісу для обліку домашніх витрат – ідея не просто актуальна, а потрібна.

З технічного боку використано перевірене поєднання PHP та MySQL – воно дозволяє створити просту, але надійну платформу з усіма необхідними функціями. Особлива увага – безпеці, зручності й можливості розширення.

У третій частині роботи вже реалізовано робочий прототип такої системи. Він дає змогу вести облік доходів і витрат, переглядати звіти, працювати з категоріями та бачити баланс у реальному часі. Система вийшла зручною у використанні та реально корисною для щоденного контролю бюджету.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Готович, В. А., С. В. Марценко, and Т. Л. Щербак. "Створення мобільного апаратно-програмного пристрою моніторингу характеристик якості електроенергії." Збірник наукових праць Інституту проблем моделювання в енергетиці ім. ГЄ Пухова 70 (2014): 98-105.

2. Duda, O., Dzhydzhora, L., Matsiuk, O., Stanko, A., KunaNETs, N., Pasichnyk, V., & KunaNETs, O. (2020). Mobile Information System for Monitoring the Spread of Viruses in Smart Cities. Journal of Lviv Polytechnic National University" Information Systems and NETworks, (8), 65-70.

3. Yatsyshyn, V., Pastukh, O., Palamar, A., & Zharovskyi, R. (2023). Technology of relational database management systems performance evaluation during computer systems design. Вісник Тернопільського національного технічного університету, 109(1), 54-65.

4. Codd, E. F. (1970). A relational model of data for large shared data banks. Communications of the ACM, 13(6), 377-387.

5. Stack Overflow Developer Survey 2022. Stack Overflow. URL: https://survey.stackoverflow.co/2022/#most-popular-technologies-database-prof (date of access: 24.03.2025).

6. Paul Jansen. TIOBE Index - TIOBE. TIOBE. URL: https://www.tiobe.com/tiobe-index/ (дата доступу: 23.03.2025).

7. PHP vs. Java vs. Python usage statistics, June 2025. W3Techs - extensive and reliable web technology surveys. URL: https://w3techs.com/technologies/comparison/pl-java,pl-php,pl-python (дата доступу: 24.03.2025).

8. Laaziri, M., Benmoussa, K., Khoulji, S., Larbi, KM, & El Yamami, A. (2019). Порівняльне дослідження PHP-фреймворків Laravel та Symfony. Міжнародний журнал електротехніки та комп'ютерної інженерії, 9(1), 704-712.

9. Колісниченко, Д. (2011). PHP і MySQL. Розробка Web-додатків (4-те вид.). Київ: BHV-Україна.

10. Duckett, J. (2022). PHP та MySQL. Серверна веб-розробка (укр. пер.). Київ: Діоген.

11. Колісниченко, Д. (2009). PHP і MySQL. Розробка Web-додатків. Київ: BHV-Україна.

12. Welling, L., & Thomson, L. (2009). PHP і MySQL. Розробка динамічних веб-застосунків (укр. адапт.). Київ: IT-Hub.

13. Duckett, J. (2020). PHP & MySQL: Серверна розробка веб-застосунків (укр. пер.). Київ: Діоген.

14. Duckett, J. (2021). PHP & MySQL: Server-side Web Development. Wiley.

15. Welling, L., & Thomson, L. (2017). PHP and MySQL Web Development (5th ed.). Addison-Wesley.

16. McLaughlin, B. (2012). PHP & MySQL: The Missing Manual (2nd ed.). O'Reilly Media.

17. Beighley, L., & Morrison, M. (2009). Head First PHP & MySQL. O'Reilly Media.

18. Murach, J. (2014). Murach's PHP and MySQL (2nd ed.). Mike Murach & Associates.

19. Holzner, S. (2008). PHP: The Complete Reference. McGraw-Hill Education.

20. Lockhart, J. (2015). Modern PHP: New Features and Good Practices. O'Reilly Media.

21. Ullman, L. (2012). PHP and MySQL for Dynamic Web Sites (4th ed.). Peachpit Press.

22. Sklar, D., & Trachtenberg, A. (2014). PHP Cookbook (3rd ed.). O'Reilly Media.

23. Nixon, R. (2018). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5 (5th ed.). O'Reilly Media.

24. Стручок, В. С., Стручок, О. С., & Мудра, Д. В. (2017). Навчальний посібник до написання розділу дипломного проекту та дипломної роботи

″Безпека в надзвичайних ситуаціях ″для студентів всіх спец. денної, заочної (дистанційної) та екстернатної форм навчання.

25. Стручок, В. С. (2022). Техноекологія та цивільна безпека. Частина "Цивільна безпека". Навчальний посібник.

26. Жидецький, В. Ц., Джигирей, В. С., & Мельников, О. В. (2000). Основи охорони праці. Львів: Афіша, 350, 132-136.

27. Навакатікян О. О., Кальниш В. В., & Стрюков С. М. (1997). Охорона праці користувачів комп’ютерних відеодисплейних терміналів. О. Навакатікян.

ДОДАТКИ

Додаток А

SQL-скрипт створення бази даних

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";

-- --------------------------------------------------------
-- Table structure for table `account`
--

CREATE TABLE IF NOT EXISTS `account` (
  `AccountId` int(5) NOT NULL AUTO_INCREMENT,
  `UserId` int(5) NOT NULL,
  `AccountName` varchar(255) CHARACTER SET latin1 NOT NULL,
  PRIMARY KEY (`AccountId`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=1 ;

--
-- Triggers `account`
--
DROP TRIGGER IF EXISTS `GenerateAccount`;
DELIMITER //
CREATE TRIGGER `GenerateAccount` AFTER INSERT ON `account`
 FOR EACH ROW INSERT INTO totals(UserId, AccountId, totals) values (new.UserId,
new.AccountId, '0')
//
DELIMITER ;

-- --------------------------------------------------------
-- Table structure for table `assets`
--

CREATE TABLE IF NOT EXISTS `assets` (
  `AssetsId` int(5) NOT NULL AUTO_INCREMENT,
  `UserId` int(5) NOT NULL,
  `Title` varchar(255) CHARACTER SET latin1 NOT NULL,
  `Date` date NOT NULL,
  `CategoryId` int(5) NOT NULL,
  `AccountId` int(5) NOT NULL,
  `Amount` varchar(255) CHARACTER SET latin1 NOT NULL,
  `Description` text CHARACTER SET latin1 NOT NULL,
  PRIMARY KEY (`AssetsId`),
  KEY `fk_test` (`AccountId`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=1 ;

--
-- Triggers `assets`
--
DROP TRIGGER IF EXISTS `GenerateTotalAccount`;
DELIMITER //
CREATE TRIGGER `GenerateTotalAccount` AFTER INSERT ON `assets`
 FOR EACH ROW UPDATE totals SET totals.totals=totals.totals + new.amount where
totals.userid=new.userid and totals.accountid=new.accountid
//
DELIMITER ;
DROP TRIGGER IF EXISTS `GenerateTotalUpdate`;
DELIMITER //
CREATE TRIGGER `GenerateTotalUpdate` AFTER UPDATE ON `assets`
 FOR EACH ROW UPDATE totals SET totals.totals=(select sum(Amount) from assets
where assets.userid=new.userid and assets.accountid=new.accountid) where
totals.userid=new.userid and totals.accountid=new.accountid
```

```
//
DELIMITER ;

-- --------------------------------------------------------
-- Table structure for table `bills`
--

CREATE TABLE IF NOT EXISTS `bills` (
  `BillsId` int(5) NOT NULL AUTO_INCREMENT,
  `UserId` int(5) NOT NULL,
  `Title` varchar(255) CHARACTER SET latin1 NOT NULL,
  `Dates` date NOT NULL,
  `CategoryId` int(5) NOT NULL,
  `AccountId` int(5) NOT NULL,
  `Amount` varchar(255) CHARACTER SET latin1 NOT NULL,
  `Description` text CHARACTER SET latin1 NOT NULL,
  PRIMARY KEY (`BillsId`),
  KEY `fk_testt` (`AccountId`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=1 ;

--
-- Triggers `bills`
--
DROP TRIGGER IF EXISTS `GenerateExpense`;
DELIMITER //
CREATE TRIGGER `GenerateExpense` AFTER INSERT ON `bills`
 FOR EACH ROW UPDATE totals SET totals.totals=totals.totals - new.amount where
totals.userid=new.userid and totals.accountid=new.accountid
//
DELIMITER ;

-- --------------------------------------------------------
-- Table structure for table `budget`
--

CREATE TABLE IF NOT EXISTS `budget` (
  `BudgetId` int(5) NOT NULL AUTO_INCREMENT,
  `UserId` int(5) NOT NULL,
  `CategoryId` int(5) NOT NULL,
  `Dates` date NOT NULL,
  `Amount` int(10) NOT NULL,
  PRIMARY KEY (`BudgetId`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=1 ;

-- --------------------------------------------------------
-- Table structure for table `category`
--

CREATE TABLE IF NOT EXISTS `category` (
  `CategoryId` int(5) NOT NULL AUTO_INCREMENT,
  `UserId` int(5) NOT NULL,
  `CategoryName` varchar(255) CHARACTER SET latin1 NOT NULL,
  `Level` int(2) NOT NULL,
  PRIMARY KEY (`CategoryId`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=1 ;

-- --------------------------------------------------------
-- Table structure for table `totals`
--

CREATE TABLE IF NOT EXISTS `totals` (
  `TotalsId` int(5) NOT NULL AUTO_INCREMENT,
```

```
  `UserId` int(5) NOT NULL,
  `AccountId` int(5) NOT NULL,
  `Totals` int(10) NOT NULL,
  PRIMARY KEY (`TotalsId`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=1 ;

-- --------------------------------------------------------
-- Table structure for table `user`
--

CREATE TABLE IF NOT EXISTS `user` (
  `UserId` int(5) NOT NULL AUTO_INCREMENT,
  `FirstName` varchar(255) CHARACTER SET latin1 NOT NULL,
  `LastName` varchar(255) CHARACTER SET latin1 NOT NULL,
  `Email` varchar(255) CHARACTER SET latin1 NOT NULL,
  `Password` varchar(255) CHARACTER SET latin1 NOT NULL,
  `Currency` varchar(255) CHARACTER SET utf8 COLLATE utf8_bin NOT NULL,
  PRIMARY KEY (`UserId`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=1 ;

--
-- Triggers `user`
--
DROP TRIGGER IF EXISTS `GenerateDefaultAccount`;
DELIMITER //
CREATE TRIGGER `GenerateDefaultAccount` AFTER INSERT ON `user`
 FOR EACH ROW INSERT INTO account (UserId, AccountName) VALUES (new.UserId,
'Cash'), (new.UserId, 'Account'), (new.UserId, 'Card')
//
DELIMITER ;

--
-- Constraints for dumped tables
--
-- Constraints for table `assets`
--
ALTER TABLE `assets`
  ADD CONSTRAINT `fk_test` FOREIGN KEY (`AccountId`) REFERENCES `account`
(`AccountId`) ON DELETE CASCADE;

--
-- Constraints for table `bills`
--
ALTER TABLE `bills`
  ADD CONSTRAINT `fk_testt` FOREIGN KEY (`AccountId`) REFERENCES `account`
(`AccountId`) ON DELETE CASCADE;
```

Додаток Б

Програмний код

index.php

```
<?php session_start();

// Cek Active Link
function ActiveClass($requestUri)
{
    $current_file_name = basename($_SERVER['REQUEST_URI'], ".php");

    if ($current_file_name == $requestUri)
        echo 'class="active"';
}

          //check already login
          if (!isset($_SESSION['UserId'])) {
                  header ('Location: login.php');
                  exit;
          }

          // Logout
          if (isset($_GET['action'])) {
                  $action = $_GET['action'];
                  if ($action == 'logout') {
                          session_destroy();
                          header('Location: login.php');
                  }
          }


 //Link to page
if (isset($_GET['page']) && $_GET['page'] == 'Transaction') {
            $page = 'Transaction';


        } else if (isset($_GET['page']) && $_GET['page'] == 'AssetReport') {
            $page = "AssetReport";
        } else if (isset($_GET['page']) && $_GET['page'] == 'ManageBudget')
{
            $page = "ManageBudget";
        } else if (isset($_GET['page']) && $_GET['page'] ==
'ManageIncomeCategory') {
            $page = "ManageIncomeCategory";
        } else if (isset($_GET['page']) && $_GET['page'] ==
'ManageExpenseCategory') {
            $page = "ManageExpenseCategory";
        } else if (isset($_GET['page']) && $_GET['page'] == 'ManageAccount')
{
            $page = "ManageAccount";
        } else if (isset($_GET['page']) && $_GET['page'] == 'ManageIncome')
{
            $page = "ManageIncome";
        } else if (isset($_GET['page']) && $_GET['page'] == 'Settings') {
            $page = "Settings";
        } else if (isset($_GET['page']) && $_GET['page'] == 'ExpenseReport')
{
            $page = "ExpenseReport";
        } else if (isset($_GET['page']) && $_GET['page'] == 'ManageExpense')
{
```

```
            $page = "ManageExpense";
        } else if (isset($_GET['page']) && $_GET['page'] == 'ReportPdf') {
            $page = "ReportPdf";
        } else if (isset($_GET['page']) && $_GET['page'] ==
'IncomeVsExpense') {
            $page = "IncomeVsExpense";
        } else if (isset($_GET['page']) && $_GET['page'] ==
'IncomeCalender') {
            $page = "IncomeCalender";
        } else if (isset($_GET['page']) && $_GET['page'] ==
'AllIncomeReports') {
            $page = "AllIncomeReports";
        } else if (isset($_GET['page']) && $_GET['page'] ==
'AllExpenseReports') {
            $page = "AllExpenseReports";
        } else if (isset($_GET['page']) && $_GET['page'] ==
'ExpenseCalender') {
            $page = "ExpenseCalender";
        } else if (isset($_GET['page']) && $_GET['page'] == 'siteSettings')
{
            $page = "siteSettings";
        } else {
            $page = 'dashboard';
        }



//get global notification
include('includes/global.php');

//Get Header
include('includes/header.php');
echo"hello";

//set global message notification
$msgBox  ="";

if (file_exists('pages/'.$page.'.php')) {
            // Load the Page
            include('pages/'.$page.'.php');
        } else {
            // Else Display an Error

            echo '
                    <div class="wrapper">
                        <h3>Err</h3>
                        <div class="alertMsg default">
                            <i class="icon-warning-sign"></i> The page
"'.$page.'" could not be found.
                        </div>
                    </div>
                ';
        }

        include('includes/footer.php');
?>
```

login.php

```
<?php
session_start();
```

```
$msgBox = '';


//include notification page
include ('includes/notification.php');

//Include db Page
include ('includes/db.php');

//Include Function page
include ('includes/Functions.php');

//User Login

if (isset($_POST['login'])) {
    if ($_POST['email'] == '') {
        $msgBox = alertBox($EmailEmpty);
    } else
        if ($_POST['password'] == '') {
            $msgBox = alertBox($PasswordEmpty);

        } else {
            // Get User Info
            $Email = $mysqli->real_escape_string($_POST['email']);
            $Password = encryptIt($_POST['password']);

            if ($stmt = $mysqli->prepare("SELECT UserId, FirstName,
LastName, Email, Password, Currency from user WHERE Email = ? AND Password =
? ")) {
                $stmt->bind_param("ss", $Email, $Password);
                $stmt->execute();
                $stmt->bind_result($UserId_, $FirstName_, $LastName_,
$Email_, $Password_, $Currency_);
                $stmt->store_result();
                $stmt->fetch();
                if ($num_of_rows = $stmt->num_rows >= 1) {
                    session_start();
                    $_SESSION['UserId'] = $UserId_;
                    $_SESSION['FirstName'] = $FirstName_;
                    $_SESSION['LastName'] = $LastName_;
                    $_SESSION['Currency'] = $Currency_;
                    $UserIds = $_SESSION['UserId'];


                    // Generate default Category for New User
                    $a = "SELECT CategoryName FROM category WHERE UserId =
$UserIds";
                    $b = mysqli_query($mysqli, $a);

                    if (mysqli_num_rows($b) >= 1) {
                      echo '';
                    } else {
                        $c = "INSERT INTO category(UserId, CategoryName,
Level) VALUES ($UserIds, 'Salary', 1), ($UserIds, 'Alowance', 1), ($UserIds,
'Petty Cash', 1), ($UserIds, 'Bonus', 1), ($UserIds, 'Food', 2),

          ($UserIds, 'Social Life', 2), ($UserIds, 'Self-Development', 2),
($UserIds, 'Transportation', 2), ($UserIds, 'Culture', 2), ($UserIds,
'Household', 2), ($UserIds, 'Apparel', 2),
```

```
          ($UserIds, 'Beauty', 2), ($UserIds, 'Health', 2), ($UserIds,
'Gift', 2)";
                        $d = mysqli_query($mysqli, $c);
                    }
                    echo '<META HTTP-EQUIV="Refresh" Content="0;
URL=index.php">';
                } else {
                    $msgBox = alertBox($LoginError);
                }
            }
        }

}
?>


<!DOCTYPE html>
<html lang="en">

<head>

    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>Money Manager Login</title>

    <!-- Bootstrap Core CSS -->
    <link href="css/bootstrap.min.css" rel="stylesheet">

    <!-- MetisMenu CSS -->
    <link href="css/plugins/metisMenu/metisMenu.min.css" rel="stylesheet">

    <!-- Custom CSS -->
    <link href="css/custom.css" rel="stylesheet">

    <!-- Custom Fonts -->
    <link href="font-awesome-4.1.0/css/font-awesome.min.css"
rel="stylesheet" type="text/css">

    <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media
queries -->
    <!-- WARNING: Respond.js doesn't work if you view the page via file:// -
->
    <!--[if lt IE 9]>
        <script
src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
        <script
src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>
    <![endif]-->

</head>

<body>

    <div class="container">
        <div class="row">
            <div class="col-md-4 col-md-offset-4">
                <div class="login-panel panel panel-primary">
```

```
                    <div class="panel-heading">
                        <h3 class="panel-title text-center"><span
class="glyphicon glyphicon-lock"></span>  <?php echo
$UserSign; ?></h3>
                    </div>
                    <div class="panel-body">
                                        <?php if ($msgBox) {
    echo $msgBox;
} ?>
                        <form method="post" action="" role="form">
                            <fieldset>
                                <div class="form-group">
                                    <label for="email"><?php echo $Emails;
?></label>
                                    <input class="form-control"
placeholder="<?php echo
$Emails; ?>" name="email" type="email" autofocus>
                                </div>
                                <div class="form-group">
                                     <label for="password"><?php echo
$Passwords; ?></label>
                                    <input class="form-control"
placeholder="<?php echo
$Passwords; ?>" name="password" type="password" value="">
                                </div>

                                <hr>
                                 <button type="submit" name="login"
class="btn btn-success btn-block"><span class="glyphicon glyphicon-log-
in"></span>  <?php echo
$SignIn; ?></button>                                    <hr>
                                <a href="signUp.php" class="btn btn-info btn-
block"> <span class="glyphicon glyphicon-pencil"></span> <?php echo
$RegisterAnAccount; ?></a>
                            </fieldset>
                        </form>
                    </div>
                </div>
            </div>
        </div>
        <div class="row text-center">

        </div>
    </div>

    <!-- jQuery Version 1.11.0 -->
    <script src="js/jquery-1.11.0.js"></script>

    <!-- Bootstrap Core JavaScript -->
    <script src="js/bootstrap.min.js"></script>

    <!-- Metis Menu Plugin JavaScript -->
    <script src="js/plugins/metisMenu/metisMenu.min.js"></script>

    <!-- Custom Theme JavaScript -->
    <script src="js/sb-admin-2.js"></script>

</body>

</html>
```

signUp.php

```
<?php


$msgBox = '';


//include notification page
include('includes/notification.php');

//Include Function page
include('includes/Functions.php');

include('includes/db.php');

//User Signup
if(isset($_POST['signup'])){
  if($_POST['email'] == '' || $_POST['firstname'] == '' || $_POST['lastname']
== '' || $_POST['password'] == '' || $_POST['rpassword'] == '') {
                        $msgBox = alertBox($SignUpEmpty);
                } else if($_POST['password'] != $_POST['rpassword']) {
                        $msgBox = alertBox($PwdNotSame);

                } else {
                        // Set new account
                        $Email          = $mysqli-
>real_escape_string($_POST['email']);
                        $Password       = encryptIt($_POST['password']);
                        $FirstName      = $mysqli-
>real_escape_string($_POST['firstname']);
                        $LastName       = $mysqli-
>real_escape_string($_POST['lastname']);
                        $Currency       = $mysqli-
>real_escape_string($_POST['currency']);

                        //Check if already register

                        $sql="Select Email from user Where Email =
'$Email'";

                         $c= mysqli_query($mysqli, $sql);

                    if (mysqli_num_rows($c) >= 1) {

                        $msgBox = alertBox($AlreadyRegister);
                    }
                    else{

                        // add new account
                        $sql="INSERT INTO user (FirstName, LastName, Email,
Password, Currency) VALUES (?,?,?,?,?)";
                        if($statement = $mysqli->prepare($sql)){
                                //bind parameters for markers, where (s =
string, i = integer, d = double,  b = blob)
                                $statement->bind_param('sssss', $FirstName,
$LastName, $Email, $Password, $Currency);
                                $statement->execute();
                        }
                        $msgBox = alertBox($SuccessAccount);
                        }
```

```
                }
}

?>

<!DOCTYPE html>
<html lang="en">

<head>

    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>Money Manager Sign Up</title>

    <!-- Bootstrap Core CSS -->
    <link href="css/bootstrap.min.css" rel="stylesheet">

    <!-- MetisMenu CSS -->
    <link href="css/plugins/metisMenu/metisMenu.min.css" rel="stylesheet">

    <!-- Custom CSS -->
    <link href="css/custom.css" rel="stylesheet">

    <!-- Custom Fonts -->
    <link href="font-awesome-4.1.0/css/font-awesome.min.css"
rel="stylesheet" type="text/css">

    <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media
queries -->
    <!-- WARNING: Respond.js doesn't work if you view the page via file:// -
->
    <!--[if lt IE 9]>
        <script
src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
        <script
src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>
    <![endif]-->

</head>

<body>

    <div class="container">
        <div class="row">
            <div class="col-md-6 col-md-offset-3">
                <div class="login-panel panel panel-primary">
                    <div class="panel-heading">
                        <h3 class="panel-title text-center"><span
class="glyphicon glyphicon-lock"></span> <?php  echo $CreateAnAccount;
?></h3>
                    </div>
                    <div class="panel-body">
                                    <?php if ($msgBox) { echo $msgBox; }
?>
                        <form method="post" action="" role="form">
                            <fieldset>
                                <div class="form-group col-lg-6">
```

```
                                    <label for="email"><?php  echo $Emails;
?></label>
                                    <input class="form-control"
placeholder="<?php  echo $Emails; ?>" name="email" type="email" autofocus>
                                </div>
                                <div class="form-group col-lg-6">
                                    <label for="email"><?php  echo
$FirstNames; ?></label>
                                    <input class="form-control"
placeholder="<?php  echo $FirstNames; ?>" name="firstname" type="text" >
                                </div>
                                <div class="form-group col-lg-6">
                                    <label for="email"><?php  echo
$LastNames; ?></label>
                                    <input class="form-control"
placeholder="<?php  echo $LastNames; ?>" name="lastname" type="text" >
                                </div>
                                <div class="form-group col-lg-6">
                                    <label for="email"><?php  echo
$Currencys; ?></label>
                                    <select class="form-control bold"
name="currency">

  <option value="د.إ">Arab Emirates Dirham (AED)</option>

  <option value="؋">Afghan Afghani (AFN)</option>

  <option value="Lek">Albanian Lek (ALL)</option>

  <option value="AMD">Armenian Dram (AMD)</option>

  <option value="ANG">Neth. Antillean Guilder (ANG)</option>

  <option value="Kz">Angolan Kwanza (AOA)</option>

  <option value="$">Argentine Peso (ARS)</option>

  <option value="A$">Australian Dollar (A$)</option>

  <option value="AWG">Aruban Florin (AWG)</option>

  <option value="ман">Azerbaijani Manat (AZN)</option>

  <option value="KM">Bosnia-Her. Convertible Mark (BAM)</option>

  <option value="">Barbadian Dollar (BBD)</option>

  <option value="Tk">Bangladeshi Taka (BDT)</option>

  <option value="лв">Bulgarian Lev (BGN)</option>

  <option value="BD">Bahraini Dinar (BHD)</option>

  <option value="BIF">Burundian Franc (BIF)</option>

  <option value="BMD">Bermudan Dollar (BMD)</option>

  <option value="$">Brunei Dollar (BND)</option>

  <option value="$b">Bolivian Boliviano (BOB)</option>

  <option value="R$">Brazilian Real (R$)</option>
```

```
<option value="BSD">Bahamian Dollar (BSD)</option>
<option value="BTN">Bhutanese Ngultrum (BTN)</option>
<option value="BWP">Botswanan Pula (BWP)</option>
<option value="BYR">Belarusian Ruble (BYR)</option>
<option value="BZD">Belize Dollar (BZD)</option>
<option value="CA$">Canadian Dollar (CA$)</option>
<option value="CDF">Congolese Franc (CDF)</option>
<option value="CHF">Swiss Franc (CHF)</option>
<option value="$">Chilean Peso (CLP)</option>
<option value="¥">Chinese Yuan (CN¥)</option>
<option value="$">Colombian Peso (COP)</option>
<option value="₡">Costa Rican Colón (CRC)</option>
<option value="₱">Cuban Peso (CUP)</option>
<option value="CVE">Cape Verdean Escudo (CVE)</option>
<option value="Kč">Czech Republic Koruna (CZK)</option>
<option value="DEM">German Mark (DEM)</option>
<option value="DJF">Djiboutian Franc (DJF)</option>
<option value="kr">Danish Krone (DKK)</option>
<option value="RD$">Dominican Peso (DOP)</option>
<option value="DZD">Algerian Dinar (DZD)</option>
<option value="£">Egyptian Pound (EGP)</option>
<option value="ERN">Eritrean Nakfa (ERN)</option>
<option value="ETB">Ethiopian Birr (ETB)</option>
<option value="€">Euro (€)</option>
<option value="$">Fijian Dollar (FJD)</option>
<option value="£">Falkland Islands Pound (FKP)</option>
<option value="£">British Pound Sterling (£)</option>
<option value="GEL">Georgian Lari (GEL)</option>
<option value="GHS">Ghanaian Cedi (GHS)</option>
<option value="£">Gibraltar Pound (GIP)</option>
<option value="GMD">Gambian Dalasi (GMD)</option>
```

```
<option value="GNF">Guinean Franc (GNF)</option>
<option value="Q">Guatemalan Quetzal (GTQ)</option>
<option value="$">Guyanaese Dollar (GYD)</option>
<option value="HK$">Hong Kong Dollar (HK$)</option>
<option value="L">Honduran Lempira (HNL)</option>
<option value="kn">Croatian Kuna (HRK)</option>
<option value="Ft">Hungarian Forint (HUF)</option>
<option value="HTG">Haitian Gourde (HTG)</option>
<option value="Rp">Indonesian Rupiah (IDR)</option>
<option value="₪">Israeli New Sheqel (₪)</option>
<option value="₹">Indian Rupee (Rs.)</option>
<option value="IQD">Iraqi Dinar (IQD)</option>
<option value="﷼">Iranian Rial (IRR)</option>
<option value="kr">Icelandic Króna (ISK)</option>
<option value="J$">Jamaican Dollar (JMD)</option>
<option value="JOD">Jordanian Dinar (JOD)</option>
<option value="¥">Japanese Yen (¥)</option>
<option value="KES">Kenyan Shilling (KES)</option>
<option value="лв">Kyrgystani Som (KGS)</option>
<option value="៛">Cambodian Riel (KHR)</option>
<option value="KMF">Comorian Franc (KMF)</option>
<option value="₩">North Korean Won (KPW)</option>
<option value="₩">South Korean Won (₩)</option>
<option value="KWD">Kuwaiti Dinar (KWD)</option>
<option value="$">Cayman Islands Dollar (KYD)</option>
<option value="лв">Kazakhstani Tenge (KZT)</option>
<option value="₭">Laotian Kip (LAK)</option>
<option value="£">Lebanese Pound (LBP)</option>
<option value="Rs">Sri Lankan Rupee (LKR)</option>
<option value="$">Liberian Dollar (LRD)</option>
<option value="LSL">Lesotho Loti (LSL)</option>
```

```
<option value="LTL">Lithuanian Litas (LTL)</option>
<option value="LYD">Libyan Dinar (LYD)</option>
<option value="MAD">Moroccan Dirham (MAD)</option>
<option value="MDL">Moldovan Leu (MDL)</option>
<option value="MGA">Malagasy Ariary (MGA)</option>
<option value="ден">Macedonian Denar (MKD)</option>
<option value="MMK">Myanmar Kyat (MMK)</option>
<option value="MNT">Mongolian Tugrik (MNT)</option>
<option value="MOP">Macanese Pataca (MOP)</option>
<option value="MRO">Mauritanian Ouguiya (MRO)</option>
<option value="Rs">Mauritian Rupee (MUR)</option>
<option value="MVR">Maldivian Rufiyaa (MVR)</option>
<option value="MWK">Malawian Kwacha (MWK)</option>
<option value="MX$">Mexican Peso (MX$)</option>
<option value="MYR">Malaysian Ringgit (MYR)</option>
<option value="MT">Mozambican Metical (MZN)</option>
<option value="$">Namibian Dollar (NAD)</option>
<option value="₦">Nigerian Naira (NGN)</option>
<option value="C$">Nicaraguan Córdoba (NIO)</option>
<option value="kr">Norwegian Krone (NOK)</option>
<option value="₨">Nepalese Rupee (NPR)</option>
<option value="NZ$">New Zealand Dollar (NZ$)</option>
<option value="﷼">Omani Rial (OMR)</option>
<option value="B/.">Panamanian Balboa (PAB)</option>
<option value="S/.">Peruvian Nuevo Sol (PEN)</option>
<option value="PGK">Papua New Guinean Kina (PGK)</option>
<option value="₱">Philippine Peso (Php)</option>
<option value="Rs">Pakistani Rupee (PKR)</option>
<option value="zł">Polish Zloty (PLN)</option>
<option value="Gs">Paraguayan Guarani (PYG)</option>
<option value="﷼">Qatari Rial (QAR)</option>
```

```
<option value="RON">Romanian Leu (RON)</option>
<option value="Дин.">Serbian Dinar (RSD)</option>
<option value="руб">Russian Ruble (RUB)</option>
<option value="RWF">Rwandan Franc (RWF)</option>
<option value="﷼">Saudi Riyal (SAR)</option>
<option value="$">Solomon Islands Dollar (SBD)</option>
<option value="Rs">Seychellois Rupee (SCR)</option>
<option value="SDG">Sudanese Pound (SDG)</option>
<option value="kr">Swedish Krona (SEK)</option>
<option value="S$">Singapore Dollar (SGD)</option>
<option value="£">Saint Helena Pound (SHP)</option>
<option value="SLL">Sierra Leonean Leone (SLL)</option>
<option value="S">Somali Shilling (SOS)</option>
<option value="$">Surinamese Dollar (SRD)</option>
<option value="STD">São Tomé and Príncipe Dobra (STD)</option>
<option value="$">Salvadoran Colón (SVC)</option>
<option value="£">Syrian Pound (SYP)</option>
<option value="SZL">Swazi Lilangeni (SZL)</option>
<option value="฿">Thai Baht (฿)</option>
<option value="TJS">Tajikistani Somoni (TJS)</option>
<option value="TMT">Turkmenistani Manat (TMT)</option>
<option value="TND">Tunisian Dinar (TND)</option>
<option value="TOP">Tongan Pa‘anga (TOP)</option>
<option value="TRY">Turkish Lira (TRY)</option>
<option value="TT$">Trinidad and Tobago Dollar (TTD)</option>
<option value="NT$">New Taiwan Dollar (NT$)</option>
<option value="TZS">Tanzanian Shilling (TZS)</option>
<option value="₴">Ukrainian Hryvnia (UAH)</option>
<option value="UGX">Ugandan Shilling (UGX)</option>
<option selected="" value="$">US Dollar ($)</option>
<option value="$U">Uruguayan Peso (UYU)</option>
```

```
  <option value="лв">Uzbekistan Som (UZS)</option>

  <option value="Bs">Venezuelan Bolívar (VEF)</option>

  <option value="₫">Vietnamese Dong (₫)</option>

  <option value="VUV">Vanuatu Vatu (VUV)</option>

  <option value="WST">Samoan Tala (WST)</option>

  <option value="XOF">CFA Franc BCEAO (CFA)</option>

  <option value="EC$">East Caribbean Dollar (EC$)</option>

  <option value="XDR">Special Drawing Rights (XDR)</option>

  <option value="CFPF">CFP Franc (CFPF)</option>

  <option value="﷼">Yemeni Rial (YER)</option>

  <option value="S">South African Rand (ZAR)</option>

  <option value="ZMK">Zambian Kwacha (ZMK)</option></select>

                                                                    </select>
                                  </div>
                                  <div class="form-group col-lg-6">
                                       <label for="password"><?php  echo
$Passwords; ?></label>
                                      <input class="form-control"
placeholder="<?php  echo $Passwords; ?>" name="password" type="password"
value="">
                                 </div>
                                  <div class="form-group col-lg-6">
                                       <label for="password"><?php  echo
$RepeatPassword; ?></label>
                                      <input class="form-control"
placeholder="<?php  echo $RepeatPassword; ?>" name="rpassword"
type="password" value="">
                                 </div>
                                 <hr>
                                  <button type="submit" name="signup"
class="btn btn-success btn-block"><span class="glyphicon glyphicon-log-
in"></span>  <?php  echo $Save; ?></button>
<hr>

                             </fieldset>
                         </form>
                     </div>
                 </div>
             </div>
         </div>
         <div class="row text-center">

             <small >Powered By Developer Kuljeet Singh</small>
         </div>
     </div>

     <!-- jQuery Version 1.11.0 -->
     <script src="js/jquery-1.11.0.js"></script>
```

```
    <!-- Bootstrap Core JavaScript -->
    <script src="js/bootstrap.min.js"></script>

    <!-- Metis Menu Plugin JavaScript -->
    <script src="js/plugins/metisMenu/metisMenu.min.js"></script>

    <!-- Custom Theme JavaScript -->
    <script src="js/sb-admin-2.js"></script>

</body>

</html>
```

Settings.php

```
<?php

//Include Functions
include('includes/Functions.php');

//Include Notifications
include ('includes/notification.php');

//delete account

if(isset($_POST['change'])){
         if($_POST['email'] == '' || $_POST['firstname'] == '' ||
$_POST['lastname'] == '' || $_POST['password'] == '' || $_POST['rpassword']
== '') {
                        $msgBox = alertBox($SignUpEmpty);
                } else if($_POST['password'] != $_POST['rpassword']) {
                        $msgBox = alertBox($PwdNotSame);

                } else {
                        // Set new account
                        $Email          = $mysqli-
>real_escape_string($_POST['email']);
                        $Password       = encryptIt($_POST['password']);
                        $FirstName      = $mysqli-
>real_escape_string($_POST['firstname']);
                        $LastName       = $mysqli-
>real_escape_string($_POST['lastname']);
                        $Currency       = $mysqli-
>real_escape_string($_POST['currency']);

                        // add new account
                        $sql="UPDATE user SET FirstName = ?, LastName = ?,
Email = ?, Password = ?, Currency = ? WHERE UserId = $UserId";
                        if($statement = $mysqli->prepare($sql)){
                                //bind parameters for markers, where (s =
string, i = integer, d = double,  b = blob)
                                $statement->bind_param('sssss', $FirstName,
$LastName, $Email, $Password, $Currency);
                                $statement->execute();
                        }
                        $msgBox = alertBox($SuccessAccountUpdate);
                }
  }

// Get User Info
$GetUsers         = "SELECT FirstName, LastName, Email, Currency, Password
from user WHERE UserId = $UserId";
```

```
$GetUserq         = mysqli_query($mysqli, $GetUsers);
$UserInfos               = mysqli_fetch_assoc($GetUserq);

//Include Global page
  include ('includes/global.php');

?>

        <div id="page-wrapper">
            <div class="row">
                <div class="col-lg-12">
                    <h1 class="page-header"><?php echo $ManageAccount; ?>
  </h1>
                </div>
                <!-- /.col-lg-12 -->
            </div>
            <!-- /.row -->

            <div class="row">
                        <?php if ($msgBox) { echo $msgBox; } ?>
                <div class="col-lg-12">
                    <div class="panel panel-green">
                        <div class="panel-heading">
                            <i class="fa fa-bar-chart-o fa-fw"></i> <?php
echo $ChangeAccountProfile; ?>

                        </div>
                        <!-- /.panel-heading -->
                        <div class="panel-body">

                        <form method="post" action="" role="form">
                            <fieldset>
                                <div class="form-group col-lg-6">
                                    <label for="email"><?php echo $Emails;
?></label>
                                    <input class="form-control"  required
placeholder="<?php echo $Emails; ?>" name="email" type="email"  value="<?php
echo $UserInfos['Email'];?>" autofocus>
                                </div>
                                <div class="form-group col-lg-6">
                                    <label for="email"><?php echo
$FirstNames; ?></label>
                                    <input class="form-control"  required
placeholder="<?php echo $FirstNames; ?>" value="<?php echo
$UserInfos['FirstName'];?>" name="firstname" type="text" >
                                </div>
                                <div class="form-group col-lg-6">
                                    <label for="email"><?php echo
$LastNames; ?></label>
                                    <input class="form-control"  required
placeholder="<?php echo $LastNames; ?>" name="lastname"  value="<?php echo
$UserInfos['LastName'];?>" type="text" >
                                </div>
                                <div class="form-group col-lg-6">
                                    <label for="email"><?php echo
$Currencys; ?></label>
                                    <select class="form-control bold"
name="currency">

  <option value="<?php echo $UserInfos['Currency'];?>" selected="" ><?php
echo $UserInfos['Currency'];?></option>
```

```
<option value="" disabled>------------------------</option>
<option value="د.إ">Arab Emirates Dirham (AED)</option>
<option value="؋">Afghan Afghani (AFN)</option>
<option value="Lek">Albanian Lek (ALL)</option>
<option value="AMD">Armenian Dram (AMD)</option>
<option value="ANG">Neth. Antillean Guilder (ANG)</option>
<option value="Kz">Angolan Kwanza (AOA)</option>
<option value="$">Argentine Peso (ARS)</option>
<option value="A$">Australian Dollar (A$)</option>
<option value="AWG">Aruban Florin (AWG)</option>
<option value="ман">Azerbaijani Manat (AZN)</option>
<option value="KM">Bosnia-Her. Convertible Mark (BAM)</option>
<option value="">Barbadian Dollar (BBD)</option>
<option value="Tk">Bangladeshi Taka (BDT)</option>
<option value="лв">Bulgarian Lev (BGN)</option>
<option value="BD">Bahraini Dinar (BHD)</option>
<option value="BIF">Burundian Franc (BIF)</option>
<option value="BMD">Bermudan Dollar (BMD)</option>
<option value="$">Brunei Dollar (BND)</option>
<option value="$b">Bolivian Boliviano (BOB)</option>
<option value="R$">Brazilian Real (R$)</option>
<option value="BSD">Bahamian Dollar (BSD)</option>
<option value="BTN">Bhutanese Ngultrum (BTN)</option>
<option value="BWP">Botswanan Pula (BWP)</option>
<option value="BYR">Belarusian Ruble (BYR)</option>
<option value="BZD">Belize Dollar (BZD)</option>
<option value="CA$">Canadian Dollar (CA$)</option>
<option value="CDF">Congolese Franc (CDF)</option>
<option value="CHF">Swiss Franc (CHF)</option>
<option value="$">Chilean Peso (CLP)</option>
<option value="¥">Chinese Yuan (CN¥)</option>
```

```
<option value="$">Colombian Peso (COP)</option>
<option value="₡">Costa Rican Colón (CRC)</option>
<option value="₱">Cuban Peso (CUP)</option>
<option value="CVE">Cape Verdean Escudo (CVE)</option>
<option value="Kč">Czech Republic Koruna (CZK)</option>
<option value="DEM">German Mark (DEM)</option>
<option value="DJF">Djiboutian Franc (DJF)</option>
<option value="kr">Danish Krone (DKK)</option>
<option value="RD$">Dominican Peso (DOP)</option>
<option value="DZD">Algerian Dinar (DZD)</option>
<option value="£">Egyptian Pound (EGP)</option>
<option value="ERN">Eritrean Nakfa (ERN)</option>
<option value="ETB">Ethiopian Birr (ETB)</option>
<option value="€">Euro (€)</option>
<option value="$">Fijian Dollar (FJD)</option>
<option value="£">Falkland Islands Pound (FKP)</option>
<option value="£">British Pound Sterling (£)</option>
<option value="GEL">Georgian Lari (GEL)</option>
<option value="GHS">Ghanaian Cedi (GHS)</option>
<option value="£">Gibraltar Pound (GIP)</option>
<option value="GMD">Gambian Dalasi (GMD)</option>
<option value="GNF">Guinean Franc (GNF)</option>
<option value="Q">Guatemalan Quetzal (GTQ)</option>
<option value="$">Guyanaese Dollar (GYD)</option>
<option value="HK$">Hong Kong Dollar (HK$)</option>
<option value="L">Honduran Lempira (HNL)</option>
<option value="kn">Croatian Kuna (HRK)</option>
<option value="Ft">Hungarian Forint (HUF)</option>
<option value="HTG">Haitian Gourde (HTG)</option>
<option value="Rp">Indonesian Rupiah (IDR)</option>
<option value="₪">Israeli New Sheqel (₪)</option>
```

```
<option value="₹">Indian Rupee (Rs.)</option>
<option value="IQD">Iraqi Dinar (IQD)</option>
<option value="﷼">Iranian Rial (IRR)</option>
<option value="kr">Icelandic Króna (ISK)</option>
<option value="J$">Jamaican Dollar (JMD)</option>
<option value="JOD">Jordanian Dinar (JOD)</option>
<option value="¥">Japanese Yen (¥)</option>
<option value="KES">Kenyan Shilling (KES)</option>
<option value="лв">Kyrgystani Som (KGS)</option>
<option value="៛">Cambodian Riel (KHR)</option>
<option value="KMF">Comorian Franc (KMF)</option>
<option value="₩">North Korean Won (KPW)</option>
<option value="₩">South Korean Won (₩)</option>
<option value="KWD">Kuwaiti Dinar (KWD)</option>
<option value="$">Cayman Islands Dollar (KYD)</option>
<option value="лв">Kazakhstani Tenge (KZT)</option>
<option value="₭">Laotian Kip (LAK)</option>
<option value="£">Lebanese Pound (LBP)</option>
<option value="Rs">Sri Lankan Rupee (LKR)</option>
<option value="$">Liberian Dollar (LRD)</option>
<option value="LSL">Lesotho Loti (LSL)</option>
<option value="LTL">Lithuanian Litas (LTL)</option>
<option value="LYD">Libyan Dinar (LYD)</option>
<option value="MAD">Moroccan Dirham (MAD)</option>
<option value="MDL">Moldovan Leu (MDL)</option>
<option value="MGA">Malagasy Ariary (MGA)</option>
<option value="ден">Macedonian Denar (MKD)</option>
<option value="MMK">Myanmar Kyat (MMK)</option>
<option value="MNT">Mongolian Tugrik (MNT)</option>
<option value="MOP">Macanese Pataca (MOP)</option>
<option value="MRO">Mauritanian Ouguiya (MRO)</option>
```

```
<option value="Rs">Mauritian Rupee (MUR)</option>
<option value="MVR">Maldivian Rufiyaa (MVR)</option>
<option value="MWK">Malawian Kwacha (MWK)</option>
<option value="MX$">Mexican Peso (MX$)</option>
<option value="MYR">Malaysian Ringgit (MYR)</option>
<option value="MT">Mozambican Metical (MZN)</option>
<option value="$">Namibian Dollar (NAD)</option>
<option value="₦">Nigerian Naira (NGN)</option>
<option value="C$">Nicaraguan Córdoba (NIO)</option>
<option value="kr">Norwegian Krone (NOK)</option>
<option value="₨">Nepalese Rupee (NPR)</option>
<option value="NZ$">New Zealand Dollar (NZ$)</option>
<option value="﷼">Omani Rial (OMR)</option>
<option value="B/.">Panamanian Balboa (PAB)</option>
<option value="S/.">Peruvian Nuevo Sol (PEN)</option>
<option value="PGK">Papua New Guinean Kina (PGK)</option>
<option value="₱">Philippine Peso (Php)</option>
<option value="Rs">Pakistani Rupee (PKR)</option>
<option value="zł">Polish Zloty (PLN)</option>
<option value="Gs">Paraguayan Guarani (PYG)</option>
<option value="﷼">Qatari Rial (QAR)</option>
<option value="RON">Romanian Leu (RON)</option>
<option value="Дин.">Serbian Dinar (RSD)</option>
<option value="руб">Russian Ruble (RUB)</option>
<option value="RWF">Rwandan Franc (RWF)</option>
<option value="﷼">Saudi Riyal (SAR)</option>
<option value="$">Solomon Islands Dollar (SBD)</option>
<option value="Rs">Seychellois Rupee (SCR)</option>
<option value="SDG">Sudanese Pound (SDG)</option>
<option value="kr">Swedish Krona (SEK)</option>
<option value="S$">Singapore Dollar (SGD)</option>
```

```
<option value="£">Saint Helena Pound (SHP)</option>
<option value="SLL">Sierra Leonean Leone (SLL)</option>
<option value="S">Somali Shilling (SOS)</option>
<option value="$">Surinamese Dollar (SRD)</option>
<option value="STD">São Tomé and Príncipe Dobra (STD)</option>
<option value="$">Salvadoran Colón (SVC)</option>
<option value="£">Syrian Pound (SYP)</option>
<option value="SZL">Swazi Lilangeni (SZL)</option>
<option value="฿">Thai Baht (฿)</option>
<option value="TJS">Tajikistani Somoni (TJS)</option>
<option value="TMT">Turkmenistani Manat (TMT)</option>
<option value="TND">Tunisian Dinar (TND)</option>
<option value="TOP">Tongan Pa‘anga (TOP)</option>
<option value="TRY">Turkish Lira (TRY)</option>
<option value="TT$">Trinidad and Tobago Dollar (TTD)</option>
<option value="NT$">New Taiwan Dollar (NT$)</option>
<option value="TZS">Tanzanian Shilling (TZS)</option>
<option value="₴">Ukrainian Hryvnia (UAH)</option>
<option value="UGX">Ugandan Shilling (UGX)</option>
<option value="$">US Dollar ($)</option>
<option value="$U">Uruguayan Peso (UYU)</option>
<option value="лв">Uzbekistan Som (UZS)</option>
<option value="Bs">Venezuelan Bolívar (VEF)</option>
<option value="₫">Vietnamese Dong (₫)</option>
<option value="VUV">Vanuatu Vatu (VUV)</option>
<option value="WST">Samoan Tala (WST)</option>
<option value="XOF">CFA Franc BCEAO (CFA)</option>
<option value="EC$">East Caribbean Dollar (EC$)</option>
<option value="XDR">Special Drawing Rights (XDR)</option>
<option value="XPF">CFP Franc (CFPF)</option>
<option value="﷼">Yemeni Rial (YER)</option>
```

```
  <option value="S">South African Rand (ZAR)</option>

  <option value="ZMK">Zambian Kwacha (ZMK)</option></select>

                                                                    </select>
                                  </div>
                                  <div class="form-group col-lg-6">
                                       <label for="password"><?php echo
$Passwords; ?></label>
                                      <input class="form-control"
placeholder="<?php echo $Passwords; ?>" name="password" type="password"
value="">
                                </div>
                                  <div class="form-group col-lg-6">
                                       <label for="password"><?php echo
$RepeatPassword; ?></label>
                                      <input class="form-control"
placeholder="<?php echo $RepeatPassword; ?>" name="rpassword"
type="password" value="">
                                </div>
                                <hr>
                                <div class="form-group col-lg-4 text-center">
                                 <button type="submit" name="change"
class="btn btn-success btn-block"><span class="glyphicon glyphicon-log-
in"></span>  <?php echo $Save; ?></button>
                                </div>
                            </fieldset>
                        </form>
                    </div>
                </div>
            </div>
        </div>
                    </div>
                    <!-- /.panel -->
                </div>

                <div class="col-lg-8">


            </div>
            <!-- /.row -->

        </div>
        <!-- /#page-wrapper -->

<script>


    $(function() {

     $('.notification').tooltip({
        selector: "[data-toggle=tooltip]",
        container: "body"
    })

    });
    </script>
```