МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Бакалаврська кваліфікаційна робота

на тему: Розробка WEB-застосунку для домашньої онлайн бухгалтерії

Виконав студент 4 курсу групи К-41
спеціальність 122«Комп'ютерні науки»
Гасимли Фарід Етібар огли

Керівник к.геогр.н., доцент
Коваленко Людмила Борисівна

Консультант

Рецензент д.ф.-м.н., професор
Ковальчук Володимир Володимирович

Одеса 2020

ЗМІСТ

Скорочення та умовні познаки 5
Вступ 6
1 Опис предметної області 8
1.1 Основні поняття фінансової системи 8
1.2 Поточне фінансове планування і формування бюджету 9
1.3 Побудова функціональної схеми предметної області 14
2 Вибір та обгрунтування засобів розробки 18
2.1 Обгрунтування вибору операційної системи Linux 18
2.2 Обгрунтування вибору середовища розробки NetBeans 20
2.3 Обгрунтування вибору мови програмування PHP 21
2.4 Вибір системи керування базами даних 22
2.5 Обгрунтування вибору веб-серверу 25
2.6 Вибір системи керування версіями 28
2.7 Опис фреймоврку Laravel 29
2.8 Опис технології Bootstrap 30
2.9 Шифрування даних за протоколом SSL 32
3 Проектування та програмна реалізація веб-застосунку 34
3.1 Структура проекту 34
3.2 Функціональні можливості системи 35
3.3 Взаємодія з базою даних 41
3.3.1 Робота з міграціями 41
3.3.2 Робота з моделями 42
3.4. Маршрутизація ресурсу “Домашня бухгалтерія” 43
3.5 Використання Middleware для перевірки заповнення профілю 44
3.6 Перевірка отриманих сервером даних (request) 45
3.7 Організація логіки обробки запитів за допомогою контролера 46
Висновки 48
Перелік джерел посилання 49

Додаток А Схема бази даних системи «Домашня бухгалтерія».....52
Додаток Б Приклад міграції для таблиці User_Profit 53
Додаток В Клас-модель «Категорії прибутків».....54
Додаток Г Функції тимчасової роботи Middleware 55
Додаток Ґ Функція валідації даних 57
Додаток Д Головний контролер веб-застосунку 58
Додаток Е CRUD Контролер 61

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

БД – база даних

ОФП – особистий фінансовий план

ПЗ – програмне забезпечення

ПК – персональний комп’ютер

ПС – програмна система

СКБД – система керування базами даних

IDE – Integrated Development Environment – інтегроване середовище розробки

MVC – Model View Controller – модель-вигляд-контролер

SSL – Secure Sockets Layer – рівень захищених сокетів

ВСТУП

Проблема управління особистим бюджетом є актуальною для кожної людини. Люди отримують зарплату та інші доходи, здійснюють різні покупки, зберігають заощадження. Грамотно розпоряджаючись своїм бюджетом, людина постійно стежить за тим, скільки грошей у неї є, розраховує, скільки грошей їй потрібно витратити, і приймає рішення, де вона може заощадити і від чого вона може взагалі відмовитися. Є можливість багаторазово спростити дані процеси, розробивши веб-застосунок для управління особистим бюджетом. Таким чином, тема роботи є актуальною, а розробка подібного програмного продукту затребуваною.

Метою кваліфікаційної роботи є розробка веб-застосунку для автоматизації операцій обліку доходу, витрати грошових коштів сім'ї і аналізу структури витрат.

Система, що розробляється, дозволить вести аналіз витрат і надходжень – перелік господарських операцій, розраховувати, як вони вплинуть на загальний бюджет, а також аналізувати, коли і через що було витрачено чи отримано найбільшу кількість грошей.

Для досягнення поставленої мети були сформульовані наступні завдання:

- провести огляд інформаційних ресурсів з проблеми дослідження;
- обґрунтувати вибір програмних засобів розробки та технологій для вирішення поставленої проблеми;
- розробити структуру бази даних;
- виконати реалізацію клієнтської та серверної частин веб-додатка;
- підготувати інструкцію користувача;
- виконати перевірку ефективності застосування розробленої системи.

Структура дипломної роботи складається з вступу, трьох розділів, висновків, переліку посилань на 17 найменувань, додатків. Повний обсяг проекту становить 64 сторінок, містить 15 рисунків і 4 таблиць.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Основні поняття фінансової системи

Українська фінансова система складається з державних фінансів, фінансів господарюючих суб'єктів (підприємств, організацій, установ) та фінансів домогосподарств. Людина не існує поза суспільством, тому можна сказати, що особисте споживання неможливе окремо від суспільного споживання. Одночасно споживання не існує без процесу виробництва. Це і визначає зв'язок особистих фінансів з державними та корпоративними фінансами: платежі різного роду на користь держави (наприклад, податки, придбання державних цінних паперів та ін.) і, при необхідності, у зворотний бік трансфертні платежі (пенсії, стипендії, заробітна плата працівникам бюджетної сфери та державним службовцям, допомоги по безробіттю та ін.) особисті фінанси пов'язані з державними. У свою чергу через оплату праці, вкладення в цінні папери і т.д. особисті фінанси пов'язані з корпоративними [1][1)].

У фінансові відносини такого характеру може вступати тільки громадянин, а не його сім'я або домогосподарство. Тому назва "особисті фінанси" більш точно описує вказані фінансові відносини, ніж "фінанси домогосподарства".

На відміну від державних фінансів, що задовольняють різні види суспільного споживання, мета особистих фінансів – забезпечення процесу особистого споживання індивідуума незалежно від того, яке місце в суспільстві він займає [2][2)].

Світовий досвід показує: ті, хто не думає про своє майбутнє сьогодні, ризикують втратити досягнуте вже завтра. Протягом життя у кожної людини постійно з'являються бажання. Життєвий шлях (від народження до смер-

---

[1)] [1] Вдовенко Л.О., Сушко Н.М., Фаюра Н.Д. Фінанси : навч. посіб. Київ: Центр учбової літератури, 2010. 152 с

[2)] [2] Роль В.Ф., Сергієнко В.В., Попова С.М. Фінансове право : навч. посіб. Київ: Центр учбової літератури, 2011. 392 с.

ті) складається з певних етапів, характерних для переважної більшості людей. І на кожному етапі у людини є певні життєві цілі, що слугують орієнтирами на даному життєвому шляху. Свідомо ставити цілі і керувати своїм життям нелегко. Це вимагає роздумів, аналізу та дисципліни. При узагальненні та укрупненні цілей, які свідомо ставляться перед собою, виникає життєвий план (професійне самовизначення, продовження роду і сімейних традицій, придбання житла і т.д.). Планування життя схоже на планування подорожі. Якщо не знати пункту призначення, то навряд чи вдасться скласти правильний маршрут. Без планування можна взагалі нікуди не потрапити.

Хтось заробляє багато, хтось ледь зводить кінці з кінцями, але і тим, і іншим грошових коштів катастрофічно не вистачає. Фінанси дозволяють людям відчувати себе захищеними і вільними, бути впевненими у своєму майбутньому і в майбутньому своїх дітей. Багато людей рано чи пізно приходять до розуміння того, що життя набагато покращиться, якщо їх кошти будуть під контролем. Такий контроль здійснюється за допомогою фінансового планування. Відсутність особистого фінансового плану для заможних людей – це ризик втратити все і не мати можливості в силу віку зберегти собі колишні дохід і статус. Для менш заможних людей – це ризик перейти з бідності в убогість [3][1)].

1.2 Поточне фінансове планування і формування бюджету

Особистий фінансовий план (ОФП) являє собою раціональну стратегію досягнення конкретних фінансових цілей, яка спирається на ефективні комбінації певних фінансових інструментів, виходячи з можливостей в конкретних умовах, а також прогнозованих потреб.

Побудова ОФП заснована на наступних етапах:

– постановка цілей;

[1)] [3] Берзон Н. В. Фінанси : підруч. для бакалаврів. М. : Вид-во Юрайт, 2013. 590 с.

– формування та аналіз особистої фінансової звітності;
– коригування цілей;
– визначення шляхів досягнення цілей (побудова плану інвестування).

За широтою охоплення і характером підконтрольної особистому фінансовому плану діяльності виділяють наступні плани:

– експрес-план, що враховує лише одну, найважливішу для суб'єкта мету;
– інвестиційний план, що розробляється на основі можливої до інвестування суми;
– повний (комплексний) фінансовий план, коригуються, в міру необхідності, всієї поточної інвестиційної та фінансової діяльності суб'єкта.

В якості підвидів комплексного та інвестиційного плану виділяють цільовий, антикризовий і пенсійний особистий фінансовий плани .

Першочергова задача при фінансовому плануванні полягає в тому, щоб перевести мрії і бажання в цілі. Так, мета, поставлена в рамки конкретних термінів її передбачуваного досягнення, а також сума, необхідних для цього коштів, перетворюється на задачу, доступну для безпосереднього вирішення. Без мети, тобто питання – навіщо, все інше втрачає сенс. Слід мати чітке уявлення про те, чого хочеться досягти. Цілі повинні бути чітко сформульовані, а не розпливчасті і абстрактні. Іншими словами, якщо хочете заробити мільйон, придбати квартиру, машину або влаштувати собі свято у вигляді подорожі, то планування бюджету буде кращим другом і помічником у цій справі. Таким чином у найзагальнішому вигляді основу ОФП становить перерозподіл коштів, підпорядковане логіці запланованого досягнення усвідомленої поставленої мети.

Наступний етап побудови ОФП, після постановки мети, – оцінка поточного фінансового стану: доходів, витрат, активів і пасивів, а також пото-

чної ринкової ситуації для наступних фінансових розрахунків, щоб в кінцевому рахунку показати, чи досяжні поставлені цілі в даних умовах.

Оцінку поточного фінансового стану, як правило, розбивають на пункти.

– визначення цілей;
– визначення доходів;
– визначення витрат;
– аналіз активів і пасивів;
– прийняття рішень, контроль за їх виконанням.

Кожен, хто коли-небудь займався управлінням особистими фінансами, напевно, стикався з тим, що грошових коштів йому не вистачає. Доводиться або шукати додаткове джерело доходу для покриття всіх витрат, або відмовляти від деяких статтів витрат, оскільки доходів для їх покриття вже не залишається. Багато хто також вважає, що їхнє фінансове становище поліпшилося б раз і назавжди, якби збільшилися їхні доходи, адже тоді їх вистачило б на покриття всіх витрат. Однак існує тенденція, що при підвищенні рівня доходів підвищується також і рівень витрат. Чим більше рівень доходів людини, тим менше вона обмежує свої потреби і тим більше витрат готова зробити для їх задоволення [4][1)].

При складанні особистого фінансового плану слід враховувати не тільки свої цілі і можливості, а й свої потреби, а також адекватність своїх цілей і бажань. Тільки усвідомивши, наскільки обдумані, виправдані і доцільні заплановані витрати, можна оцінити, наскільки людина далека від своєї мети (або, навпаки, наскільки вона до неї близька) і які дії необхідно зробити, щоб домогтися бажаного результату. Недостатньо просто виконувати особистий бюджет – потрібно робити це ефективно.

Найважливіше поняття в області фінансів – бюджет.

[1)] [4] Шелудько В. М. Фінансовий менеджмент : підручник. Київ: Знання, 2006. 439 с.

Бюджет – схема доходів і витрат конкретної особи (сім'ї, бізнесу, організації, держави і т.д.), що встановлюється на певний період часу, як правило на один рік. Управління особистими фінансами починається з регулярного обліку витрат і доходів, що дозволяє контролювати рух грошових потоків.

Наведемо кілька порад з складання бюджету:

– записувати правильні суми;
– аналізувати бюджет за минулі місяці, що дозволить визначити зайві витрати та їх причини, також можна скласти зведений графік бюджету за певний період часу, наприклад за рік, і знайти способи збільшення грошових коштів;
– віддавати перевагу витратам, тобто записувати витрати в план, якщо є ймовірність його наявності за незалежних або не залежних від нас причин, щоб не вийшов дефіцит бюджету.

За допомогою ведення бюджету повністю уникнути зайвих витрат не вийде, але звести їх до мінімуму – досяжне завдання. Ретельне планування і контроль за виконанням поставлених планів допоможе стати більш дисциплінованою, відповідальною і цілеспрямованою людиною.

Отже, перший крок у складанні особистого бюджету – облік доходів. Другий крок – облік витрат. Люди не знають, куди йдуть їхні грошові кошти, поки не починають аналізувати власні витрати. Поганих фінансових звичок може бути безліч: це і перевитрати, і постійні борги, величезна кількість непотрібних речей, нескінченні неоплачувані рахунки і малий залишок грошей у гаманці і на ощадному рахунку. До шкідливих фінансових звичок можна віднести:

– імпульсивні покупки. Багато молодих людей не можуть пройти повз магазин з привабливими вітринами без того, щоб зайти в нього. І там вже складно утриматися від покупки навіть непотрібних речей, не кажучи вже про речі, які необхідні;

– зловживання споживчими кредитами (взагалі будь-яка покупка в кредит, навіть найбільш незначної і недорогої речі свідчить про неправильне планування власних витрат);
– відсутність будь-якого контролю над витратами;
– несвоєчасна оплата рахунків і заборгованостей внаслідок забудькуватості;
– покупка непотрібних речей часто відбувається у великих магазинах самообслуговування, де є все, що може знадобитися споживачеві [3][1)].

Для того щоб мати чітке і ясне уявлення про своє фінансове становище, слід записати витрати і доходи.

Зібрати всі чеки, рахунки та інші платіжні документи за той місяць, в якому ведеться облік. Підрахувати дохід на місяць, заробітну плату, додати інші отримувані доходи, наприклад від здачі нерухомості в оренду, відсотки по банківських депозитах, дивіденди по акціях і ін. Далі доцільно провести облік витрат протягом одного місяця або будь-якого іншого періоду часу.

У час інформаційних технологій, велику популярність отримала організація обліку особистих фінансів за допомогою комп'ютерних програм. Контролюючи рух грошей, можна не тільки враховувати окремі доходи і витрати, а й підвести баланс.

Аналіз отриманого фінансового звіту допоможе зрозуміти, наскільки реалістичним є фінансовий план. У разі неспівпадіння бажань і можливостей, у людини з'являється необхідність зробити вибір на користь одного з двох можливих варіантів подальших дій: або обмежити власні бажання, або збільшити власні можливості [2][2)].

Після проходження даних етапів необхідно відкоригувати свої цілі так, щоб вони стали реальними і досяжними.

[1)] [3] Берзон Н. В. Фінанси : підруч. для бакалаврів. М. : Вид-во Юрайт, 2013. 590 с.

[2)] [2] Роль В.Ф., Сергієнко В.В., Попова С.М. Фінансове право : навч. посіб. Київ: Центр учбової літератури, 2011. 392 с.

1.3 Побудова функціональної схеми предметної області

Домашня бухгалтерія – це спрощена система ведення бухгалтерського обліку для домашнього господарства. На відміну від «справжнього» бухгалтерського обліку в ній можна прийняти наступні припущення.

Всі доходи можна віднести на три рахунки:

– заробітна плата – регулярні виплати і доходи від разових робіт, на які можуть підрядитися члени сім'ї;

– доходи від вкладів;

– рахунок «гаманець».

Рух грошей здійснюється за простою жорсткою схемою. Зарплатні доходи всіх членів сім'ї стікаються в «гаманець», звідти беруться гроші на короткострокові та середньострокові покупки, при цьому в режимі реального часу відстежується залишок грошей в гаманці.

В кінці кожного місяця невитрачений залишок перекладається на рахунок внесок.

Час від часу на вклад можуть нараховуватися відсотки. Гроші зі рахунки «внесок» використовуються для здійснення довготривалих покупок.

Всі витрати діляться на наступні категорії:

– комунальні послуги;

– товари повсякденного користування;

– товари середньострокового користування;

– довготривалі («великі») покупки.

Господарські операції будемо проводити за наданими чеками. На кожен рахунок встановлюємо бюджетний ліміт, виходячи з розрахункової величини щомісячних надходжень. Величина бюджетного ліміту на кожен рахунок може коригуватися (на основі практичного досвіду).

Всі ті надходження, які не витрачені за поточний місяць з рахунки «зарплата» переводяться на рахунок «внесок». Довготривалі покупки здійснюються за допомогою витрачання грошових коштів з рахунку «внесок».

На основі сказаного складемо план рахунків для нашої системи домашньої бухгалтерії. Всі покупки будуть здійснюватися тільки за рахунок власних коштів (ситуація взяття кредиту не передбачається, однак якщо його доведеться брати, то за щомісячні платежі доведеться додати рахунок витрат, а тіло кредиту віднести як «зарплату»).

Для системи домашньої бухгалтерії можна зобразити функціональну схему предметної області, яка представлена на рис.1.1.

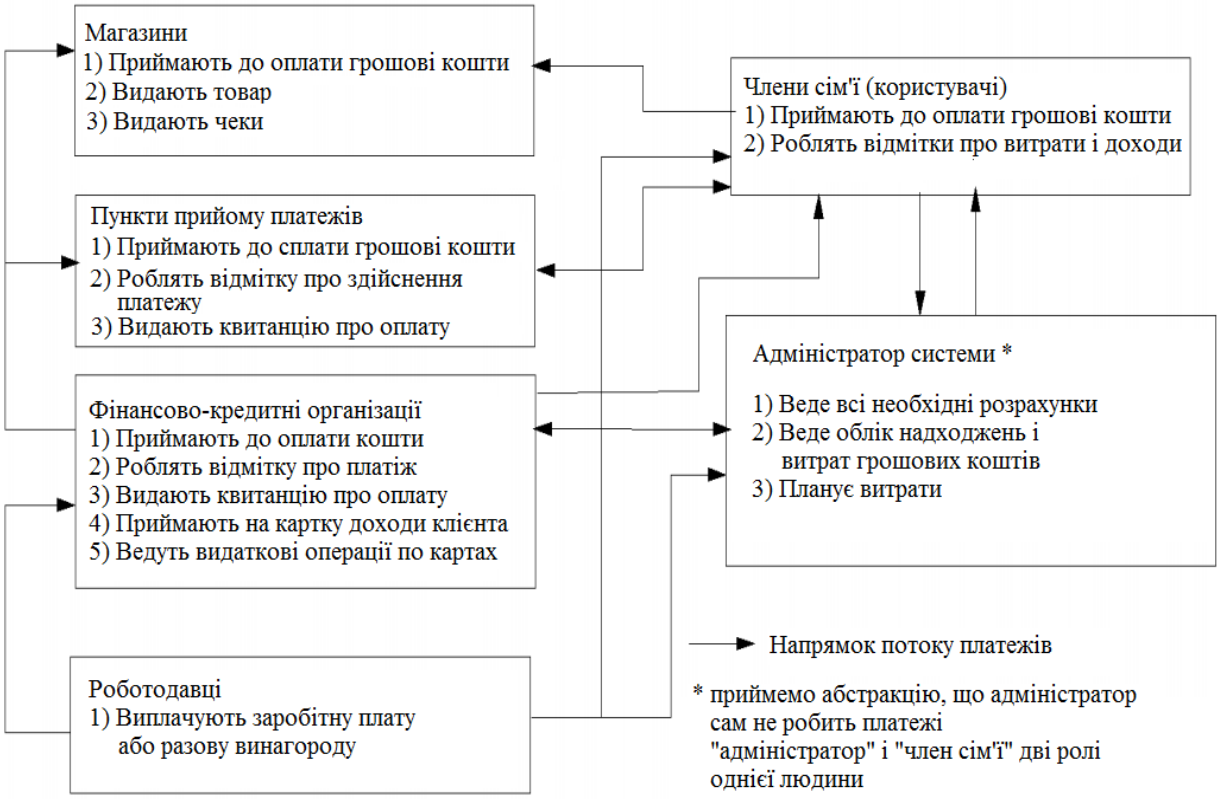

Рисунок 1.1 – Функціональна схема предметної області для завдання автоматизації домашньої бухгалтерії

У табл.1.1 наведено план рахунків, тобто схематичне уявлення статей витрат і джерел доходів для типової ситуації. Всі дані, що були вуказані у функціональній схемі предметної області та у табл.1.1 будуть у подальшому використані для проектування схеми БД.

Таблиця 1.1 – План рахунків системи домашньої бухгалтерії

Назва рахунку	Позначення рахунку	Тип рахунку
Заробітна платня	З	Рахунок дебетовий і кредитовий
Доходи від вкладів	В	Рахунок дебетовий і кредитовий
Гаманець	Г	Сюди надходять готівкові гроші від різних джерел доходу
Комунальні послуги	КП/ квартплата; плата за Інтернет; інші послуги	Рахунок кредитовий, гроші на нього можуть тільки надходити
Товари та послуги повсякденного користування	ТП/ продукти харчування; побутова хімія; предмети гігієни; канцтовари; витрати на проїзд	–
Товари середньострокового користування	ТС/ одяг і взуття; аксесуари; електроніка; побутовий інструмент	–
Товари довготрива-лого користування	ТД/ побутова техніка; електроніка та комп’ютерна техніка; меблі	–

Як видно з рис.1.1, в цій області виділяються сутності трьох типів:

– одержувачі коштів;

– платники грошових коштів;

– розподільники коштів.

До одержувачів відносимо ті організації, куди кошти сплачуються (магазини, комунальні служби та інше). Платники – це джерела доходів. Розподільниками виступають члени сім'ї.

У наступному розділі кваліфікаційної роботи буде наведено обгрунтування вибору інструментальних засобів розробки веб-застосунку.

2 ВИБІР ТА ОБГРУНТУВАННЯ ЗАСОБІВ РОЗРОБКИ

Для написання даного проекту було використано наступне програмне забезпечення:

– GNU/Linux (Ubuntu 16.04);
– NetBeans IDE 8.2;
– MySQL 5.7;
– PHP 5.6;
– Apache 2.4;
– Git 2.11.0;
– Laravel 5.4;
– Twitter Bootstrap 3.

Дані програмні засоби відноситься до “вільного” програмного забезпечення, адже у сучасному світі все частіше постає питання наявності ліцензійного програмного забезпечення. Органи внутрішніх справ все активніше борються з піратством, як підсумок стали зникати постійні точки продажів піратських дисків з операційними системами та іншим програмним забезпеченням. Варто відзначити, що на даний час збільшилася кількість користувачів, які купують ліцензійне програмне забезпечення або ж використовувати Open Source та freeware-аналоги.

Саме тому для написання системи обліку особистих фінансів “Домашні рахунки” було обрано вище перераховане програмне забезпечення, яке має такі ж можливості, як платні аналоги.

2.1 Обгрунтування вибору операційної системи Linux

Linux – загальна назва Unix-подібних операційних систем, заснованих на однойменному ядрі [5][1)]. Ядро Linux створюється і поширюється відповідно до моделі розробки вільного та відкритого програмного забезпечення. То-

[1)] [5] Linux – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/ Linux (дата звернення 14.04.2020)

му загальна назва не має на увазі будь-якої єдиної «офіційної» комплектації Linux; вони поширюються в основному безкоштовно у вигляді різних готових дистрибутивів, що мають свій набір прикладних програм і вже налаштованих під конкретні потреби користувача. Перший реліз ядра системи відбувся 5 жовтня 1991.На початковому етапі Linux безкоштовно розроблявся тільки ентузіастами-добровольцями, але з успіхом Linux і його масовим комерційним використанням допрацьовувати ОС і вносити свій вклад стали і компанії, з часом ставши значною силою. Переважна більшість ПЗ в сучасних дистрибутивах як і раніше є за вільними ліцензіями, як правило, за винятком невеликої кількості пропрієтарних компонентів.

Linux працює на безлічі процесорів різних архітектур, таких як x86, x86-64, PowerPC, ARM, Alpha AXP, SPARC, Motorola 680x0, SuperH, IBM System / 390, MIPS, PA-RISC, AXIS CRIS, Renesas M32R, Atmel AVR32, Renesas H8 / 300, NEC V850, Tensilica Xtensa і багатьох інших.

На відміну від комерційних систем, таких як Windows або Mac OS X, Linux не має географічного центру розробки. Програми для Linux – результат роботи тисяч проектів. Саме така гнучка і динамічна система розробки, неможлива для проектів з закритим кодом, визначає виняткову економічну ефективність Linux. Низька вартість вільних розробок, налагоджені механізми тестування та поширення, залучення людей з різних країн, що володіють різним баченням проблем, захист коду ліцензією GPL – все це стало причиною успіху вільних програм.

Більшість користувачів для установки Linux використовують дистрибутиви. Дистрибутив це не просто набір програм, а ряд рішень для різних завдань користувачів, об'єднаних єдиними системами установки, управління і поновлення пакетів, настройки і підтримки. Найбільш поширені в світі дистрибутиви:

– Debian GNU/Linux – иодин з найстаріших дистрибутивів, що розробляється великим співтовариством розробників, служить осно-

вою для створення безлічі інших дистрибутивів та відрізняється строгим підходом до включення невільного ПЗ;

– Ubuntu – дистрибутив, заснований на Debian і швидко завоював популярність, підтримується спільнотою, розробляється Canonical Ltd. Основна збірка орієнтована на легкість в освоєнні і використанні, при цьому існують серверна і мінімальна збірка;
– Linux Mint – дистрибутиви, засновані як на Ubuntu, так і на Debian (LMDE) і повністю з ними сумісні, що включають в себе за замовчуванням Java, Adobe Flash і багато іншого.

Для виконання даного проекту був обраний один з найбільш популярних дистрибутивів Ubuntu [6][1)]. Серед основних цілей Ubuntu – надання сучасного й водночас стабільного програмного забезпечення із сильним акцентом на простоту встановлення та користування. Ubuntu надає користувачу мінімальний набір програм загального призначення. Спеціалізоване програмне забезпеченняможна отримати з відповідних репозиторіїв. Серверний варіант системи включає також засоби, потрібні для організації сервера баз даних, веб-сервера, сервера електронної пошти тощо.

2.2 Обгрунтування вибору середовища розробки NetBeans

NetBeans IDE – вільне інтегроване середовище розробки (IDE) для мов програмування Java, JavaFX, C/C++, PHP, JavaScript, HTML5, Python, Groovy. Середовище може бути встановлене і для підтримки окремих мов, і у повній конфігурації. Середовище розробки NetBeans за замовчуванням підтримує розробку для платформ J2SE і J2EE [7][2)].

Поширюється у сирцевих текстах під ліцензією Apache License. Проект NetBeans IDE підтримувався і спонсорувався фірмою Sun Microsystems і

[1)] [6] Ubuntu – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/Ubuntu (дата звернення 14.04.2020)

[2)] [7] NetBeans – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/NetBeans (дата звернення 14.04.2020)

після придбання Sun – Oracle. У жовтні 2016 року Oracle передав NetBeans у власність Apache Software Foundation, яка займається розробкою і підтримкою проекту.

NetBeans IDE доступна для платформ Microsoft Windows, GNU/Linux, FreeBSD, і Solaris (як SPARC, так x86). Для інших платформ доступна можливість зібрати NetBeans самостійно із сирцевих текстів.

За якістю і можливостям останні версії NetBeans IDE змагається з найкращими інтегрованими середовищами розробки, підтримуючи рефакторинг, профілювання, виділення синтаксичних конструкцій кольором, автодоповнення мовних конструкцій на льоту, шаблони коду та інше.

NetBeans IDE підтримує плагіни, дозволяючи розробникам розширювати можливості середовища. Для розробки проекта найбільш важливим є підтримка середовищем мови програмування PHP.

2.3 Обгрунтування вибору мови програмування PHP

Якщо необхідно розробити сайт, просто не обійтися без використання найбільш популярної мову для веб програмування - PHP.

PHP – скриптова мова програмування, призначений для генерації HTMLстраніц на стороні сервера [8][1)]. PHP інтерпретується веб-сервером в HTML-код, який передається в браузер. Розробка PHP почалася в 1995 році. Відтоді відбувалося постійне його вдосконалення і вихід нових версій.

Розглядаючи різні аспекти використання PHP, можна виділити наступні основні її переваги: безпека та гнучкість.

Засоби безпеки системного рівня PHP можна налаштувати так, щоб вона забезпечувала найбільш можливу свободу в діях і безпеки. PHP працює в безпечному режимі (safe mode), який ставить обмеження на можливості за-

[1)] [8] PHP – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/PHP (дата звернення 14.04.2020)

стосування користувачем PHP. Наприклад, максимальний час виконання і використання пам'яті.

Засоби безпеки рівня програми PHP включає надійні механізми шифрування. PHP також сумісний з багатьма додатками інших розробників, дозволяє інтегрувати його використовуючи захищені технології електронної комерції. Вихідний код PHP можна переглянути в браузері, оскільки він виконується на сервері.

PHP використовується не тільки в поєднанні з HTML, але і з JavaScript, WML, XML і інших мовах програмування. PHP-код може передаватися будь-яким браузерам і пристроїв, в тому числі стільникових телефонів, портативних комп'ютерів. PHP-код можна виконувати в режимі командного рядка.

PHP працює на різних web-серверах (Apache, Netscape Enterprise Server, Microsoft IIS, Stronghold, Zeus) і платформах (UNIX, Solaris, FreeBSD, Windows 95/98 / NT / 2000 / XP / 2003) [9][1)].

2.4 Вибір системи керування базами даних

Вибір системи управління бази даних (СКБД) являє собою складне завдання і є одним з важливих етапів розробки програмної системи. Обраний програмний продукт повинен задовольняти як поточним, так і майбутнім потребам розвитку програмного продукту. Вимоги до СКБД можуть змінюватися в залежності від поставлених цілей. Проте, можна виділити кілька груп критеріїв:

- моделювання даних, моделі даних, підтримка тригерів та збережених функцій, засоби пошуку, підтримувані типи даних, реалізація мови запитів;
- особливості архітектури і функціональні можливості, мобільність,
- масштабованість, мережеві можливості;

[1)] [9] Кузнецов М.В, Объектно-ориентированное программирование на PHP. СПб.: «БХВ-Петербург», 2007. 608 с.

- контроль роботи системи, контроль використання пам'яті комп'ютера, автонастройка;
- продуктивність, можливості паралельної архітектури, можливості оптимизирования запитів;
- надійність, відновлення після збоїв, резервне копіювання, відкат змін, багаторівнева система захисту;
- вимоги до робочого середовища, підтримувані апаратні платформи, мінімальні вимоги до обладнання, максимальний розмір пам'яті, операційні системи, під керуванням яких здатна працювати СКБД;
- змішані критерії, якість і повнота документації, модель формування вартості, стабільність виробника [10][1)].

MySQL – вільна реляційна система керування базами даних. Розробку та підтримку MySQL здійснює корпорація Oracle, що отримала права на торговельну марку разом з поглиненої SunMicrosystems, яка раніше придбала шведську компанію MySQL AB. Продукт поширюється як під GNU GeneralPublicLicense, так і під власною комерційною ліцензією. Крім цього, розробники створюють функціональність за замовленням ліцензійних користувачів. Саме завдяки такому замовленню майже в найраніших версіях з'явився механізм реплікації [11][2)].

MySQL є рішенням для малих і середніх додатків. Входить до складу серверів WAMP, AppServ, LAMP і в портативні збірки серверів Денвер, XAMPP.

Максимальний розмір таблиць в MySQL 3.22 до 4 ГБ, в наступних версіях максимальний розмір до 8 млн ТБ (263 байт).

Розмір таблиці обмежений її типом. У загальному випадку тип MyISAM обмежений граничним розміром файлу в файлової системі операційної системи. Наприклад, в NTFS цей розмір теоретично може бути до 32

[1)] [10] Дейт, К.Дж. Введение в системы баз данных. 6-е издание: Пер. с англ. М.: Вильямс, 2000. 848 с.

[2)] [11] MySQL – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/MySQL (дата звернення 14.04.2020)

екзабайт. У разі InnoDB, одна таблиця може зберігатися в декількох файлах, які мають єдине табличний простір. Розмір останнього може досягати 64 терабайт.

На відміну від MyISAM, в InnoDB є значне обмеження на кількість стовпців, яке можна додати в одну таблицю. Розмір сторінки пам'яті за замовчуванням становить 16 кілобайт, з яких під дані відведено 8123 байта. Розмір покажчика на динамічні поля становить 20 байт. Таким чином, в разі використання динамічного формату рядка (ROW_FORMAT = DYNAMIC), одна таблиця може вмістити максимум 409 стовпців типу blob або text [11][1)].

З розглянутого вище інструментальних засобів була обрана СКБД під назвою «MySQL 5.7», оскільки:

– є безкоштовна бібліотека для роботи даної СУБД з серверним додатком Apache;
– є безкоштовний веб-інтерфейс під назвою «PHPMyAdmin» для адміністрування MySQL. «PHPMyAdmin» дозволяє через браузер здійснювати адміністрування сервера MySQL, запускати команди SQL і переглядати вміст таблиць і баз даних;
– існує безліч функцій, а також модулів, написаних на PHP для роботи з MySQL.

Для обґрунтування вибору СКБД наведемо порівняльний аналіз MySQL з іншими популярними варіантами: PostgreSQL, SQLite (табл.2.1).

З огляду на те, що наш веб-застосунок вимагає особливих умов до швидкості роботи БД, масштабованості і ін., то ми можемо використовувати будь-який з варіантів. Віддамо перевагу MySQL з огляду на її високу популярність, щодо високій швидкості роботи, наявності великої кількості документації і великої спільноти розробників.

Для доступу до бази даних з PHP будемо використовувати простий і зрозумілий інтерфейс PDO (PHP Data Objects).

[1)] [11] MySQL – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/MySQL (дата звернення 14.04.2020).

Таблиця 2.1 – Порівняльний аналіз СКБД

	MySQL	PostgreSQL	SQLite
Архітектура	Клієнт-сервер	Клієнт-сервер	Вбудована БД
Підтримка веб-хостингами (веб-серверами)	Дуже гарна. Переважна більшість хостингів PHP мають підтримку MySQL	Задовільна. Підтримка PostgreSQL часто зустрічається	Задовільна. Підтримка SQLite часто зустрічається, можна направити запит в підтримку про установку необхідного модуля
Документація	Англомовна на офіційному сайті		
Простота розробки та адміністрування	Адміністрування через консоль, безліч додатків для різних платформ для локального адміністрування, адміністрування через веб-сервер		
Надійність збереження даних	Надійність забезпечується сервером БД. Є механізми резервного копіювання		Вся БД зберігається в одному файлі

2.5 Обгрунтування вибору веб-серверу

Серверне програмне забезпечення в інформаційних технологіях –

програмний компонент обчислювальної системи, що виконує сервісні (обслуговуючі) функції по запиту клієнта, надаючи йому доступ до певних ресурсів або послуг. Поняття сервер і клієнт, і закріплені за ними ролі, утворюють програмну концепцію «клієнт-сервер».

Для взаємодії з клієнтом (або клієнтами, якщо підтримується одночасна робота з декількома клієнтами), сервер виділяє необхідні ресурси між процесами взаємодії (колективна пам'ять, пайп, сокет і т.п.) і очікує запити на відкриття з'єднання (або, власне, запити на наданий сервіс). Залежно від типу та-

кого ресурсу, сервер може обслуговувати процеси в межах однієї комп'ютерної системи, або процеси на інших машинах через канали передачі даних (наприклад, COM-порт), або мережеві з'єднання.

Формат запитів клієнта і відповідей сервера визначається протоколом. Специфікації відкритих протоколів описуються відкритими стандартами, наприклад, протоколи Інтернету визначаються в документах RFC.

Залежно від виконуваних завдань одні сервера, при відсутності запитів на обслуговування, можуть простоювати в очікуванні. Інші можуть виконувати якусь роботу (наприклад, роботу зі збору інформації), у таких серверів робота з клієнтами може бути другорядною завданням.

Веб-сервер – це сервер, що приймає HTTP-запити від клієнтів, зазвичай веб-браузерів, і видає їм HTTP-відповіді, зазвичай разом з HTML-сторінкою, зображенням, файлом, медіа-потоком або іншими даними.

Веб-сервером називають програмне забезпечення, яке виконує функції веб-сервера, так і безпосередньо комп'ютер, на якому це програмне забезпечення працює.

Клієнт, яким зазвичай є веб-браузер, передає веб-серверу запити на отримання ресурсів, позначених URL-адресами. Ресурси – це HTML-сторінки, зображення, файли, медіа-потоки або інші дані, які необхідні клієнту. У відповідь веб-сервер передає клієнту запитані дані. Цей обмін відбувається по протоколу HTTP.

Веб-сервери можуть мати такі додаткові функції:

– автоматизація роботи веб сторінок;
– ведення журналу звернень користувачів до ресурсів;
– аутентифікація і авторизація користувачів;
– підтримка динамічно генеруються сторінок;
– підтримка HTTPS для захищених з'єднань з клієнтами.

Часто на комп'ютері разом з веб-сервером встановлюється також і поштовий сервер.

Одним з найбільш поширеним веб-сервером є Apache – вільний веб-сервер, найбільш часто використовуваний в UNIX-подібних операційних системах [12][1)].

Деякі інші відомі веб-сервери:

– IIS від компанії Microsoft, який поширюється з серверними ОС сімейства Windows;
– Nginx – вільний веб-сервер, що розробляється Ігорем Сисоєвим з 2002 року і користується великою популярністю на великих сайтах;
– lighttpd – вільний веб-сервер;
– GoogleWebServer – веб-сервер, заснований на Apache і доопрацьований компанією Google;
– Resin – вільний веб-сервер додатків;
– Cherokee – вільний веб-сервер, керований тільки через web-інтерфейс;
– Rootage – веб-сервер, написаний на Java;
– THTTPD – простий, маленький, швидкий і безпечний веб-сервер.

В якості клієнтів для звернення до веб-серверів можуть використовуватися різні програми та пристрої:

– браузер, що працює на настільному комп'ютері або переносному пристрої (наприклад, кишеньковому ПК);
– різноманітні програми, самостійно звертаються до веб-сервера для отримання оновлень або іншої інформації (наприклад, антивірус може періодично запитувати у певного веб-сервера оновлення своїх баз даних);
– мобільний телефон, який одержує доступ до ресурсів веб-сервера при допомоги протоколу WAP;
– інші цифрові пристрої або побутова техніка.

[1)] [12] Apache – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/Apache (дата звернення 14.04.2020).

В якості сервера був обраний веб-сервер «Apache 2.4», оскільки він підтримує такі функції:

– автоматизація роботи веб-сторінок;

– ведення журналу звернень користувачів до ресурсів;

– аутентифікація і авторизація користувачів;

– підтримка динамічно генеруються сторінок;

– підтримка HTTPS для захищених з'єднань з клієнтами.

2.6 Вибір системи керування версіями

Git – розподілена система керування версіями файлів та спільної роботи. Проект створив Лінус Торвальдс для управління розробкою ядра Linux, а сьогодні підтримується Джуніо Хамано. Git є однією з найефективніших, надійних і високопродуктивних систем керування версіями, що надає гнучкі засоби нелінійної розробки, що базуються на відгалуженні і злитті гілок. Для забезпечення цілісності історії та стійкості до змін заднім числом використовуються криптографічні методи, також можлива прив'язка цифрових підписів розробників до тегів і комітів [13][1)].

Прикладами проектів, що використовують Git, є ядро Linux, Android, LibreOffice, Cairo, GNU Core Utilities, Mesa 3D, Wine, багато проектів з X.org, XMMS2, GStreamer, Debian DragonFly BSD, Perl, Eclipse, GNOME, KDE, Qt, Ruby on Rails,PostgreSQL,VideoLAN,PHP, One Laptop Per Child (OLPC), АБІС Koha, GNU LilyPond та ELinks і деякі дистрибутиви GNU/Linux.

Програма є вільною і випущена під ліцензією GNU GPL версії 2.

Система спроектована як набір програм, спеціально розроблених з врахуванням їхнього використання у скриптах. Це дозволяє зручно створювати спеціалізовані системи управління версіями на базі Git або користувацькі

[1)] [13] Git – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/Git (дата звернення 14.04.2020).

інтерфейси. Наприклад, Cogito є саме таким прикладом фронтенда до репозиторіїв Git. A StGit використовує Git для управління колекцією латок.

Система має ряд користувацьких інтерфейсів: наприклад, gitk та git-gui розповсюджуються з самим Git.

Віддалений доступ до репозиторіїв Git забезпечується git-демоном, SSH або HTTP сервером. TCP-сервіс git-daemon входить у дистрибутив Git і є разом з SSH найпоширенішим і надійним методом доступу. Метод доступу HTTP, незважаючи на ряд обмежень, дуже популярний в контрольованих мережах, тому що дозволяє використання існуючих конфігурацій мережевих фільтрів.

2.7 Опис фреймоврку Laravel

Laravel – безкоштовний веб-фреймворк з відкритим кодом [14][1)], призначений для розробки з використанням архітектурної моделі MVC (Model View Controller – модель-вигляд-контролер).

Фреймворк Laravel використовує архітектурну модель MVC, яка являє собою схему поділу даних програми, призначеного для користувача інтерфейсу і керуючої логіки на три окремих компонента: модель, уявлення і контролер – таким чином, що модифікація кожного компонента може здійснюватися незалежно.

Модель (Model) надає дані і реагує на команди контролера, змінюючи свій стан.

Вигляд (View) відповідає за відображення даних моделі користувачеві, реагуючи на зміни моделі.

Контролер (Controller) інтерпретує дії користувача, сповіщаючи модель про необхідність змін.

[1)] [14] Laravel – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/Laravel (дата звернення 14.04.2020).

На рис.2.1 зображена схема роботи застосунку за шаблоном проектування MVC.

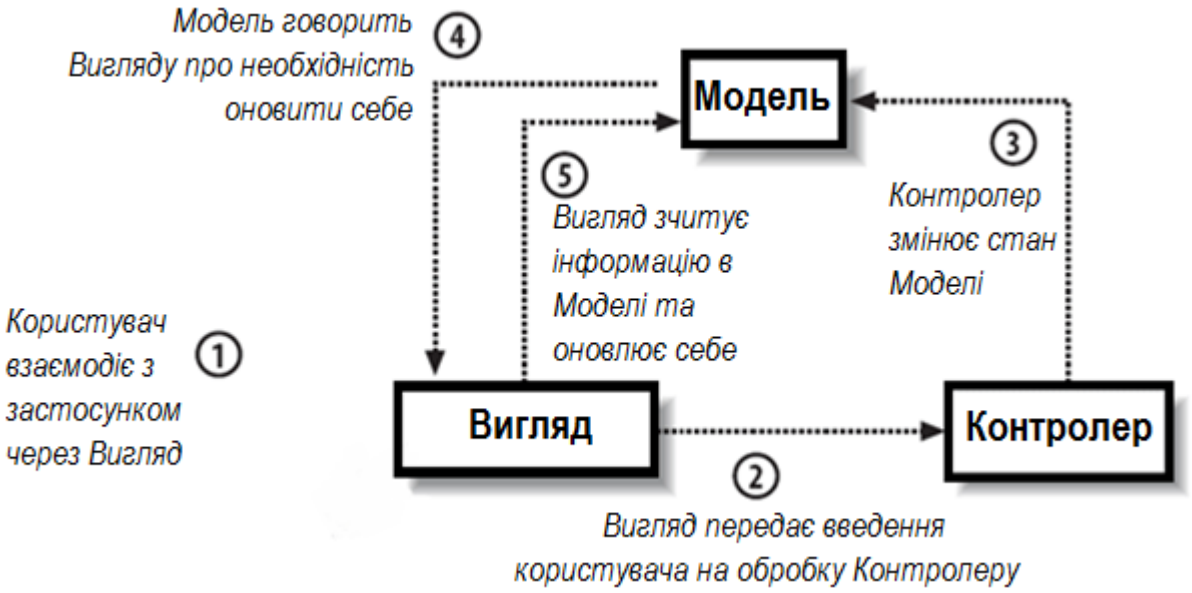

Рисунок 2.1 – Схема роботи застосунку за шаблоном проектування MVC

2.8 Опис технології Bootstrap

HTML-розмітка та CSS-стилі будуть написані на Bootstrap.

Bootstrap – це вільний набір інструментів з відкритим кодом для створення сайтів і веб-додатків. Включає в себе HTML і CSS-шаблони оформлення для типографіки, веб-форм, кнопок, міток, блоків навігації та інших компонентів веб-інтерфейсу, включаючи JavaScript-розширення. Він спрощує розробку динамічних веб-сайтів і веб-додатків [15][1)].

Четверта версія фреймворку має числені оновленнями:

– використано синтаксис Sass замість Less;

– покращено процес верстки макетів, зокрема блочної структури;

– додано підтримка flexbox, компоненту з HTML5;

[1)] [15] Bootstrap – Вікіпедія. (загол. з екрана). URL: https://uk.wikipedia.org/wiki/Bootstrap (дата звернення 04.01.2020).

– прев'ю та панелі замінено компонентом «карти» – це віднині невеликі за форматом елементи з прев'ю зображень, текстових блоків з бордерами;

– всі HTML-резети зібрано в єдиному модулі під назвою «Reboot» (в попередніх версіях цей код зберігався в Normalize.css);

– додано нові можливості з кастомізації шаблонів. Для оновлення стилів за замовчанням досить відредагувати змінну в Sass-файлі і отримати оновлених файл css.

– скасовано підтримку IE8;

– розміри вказано у rem і em замість пікселів, що покращує мобільний вигляд фреймворку;

– оновлено всі плагіни JavaScript;

– оновлено роботу спливних вікон і підказок;

– покращено документацію і пошук сайтом фреймворку.

Bootstrap сумісний з останніми версіями браузерів Google Chrome, Firefox, Internet Explorer, Opera і Safari (деякі з цих браузерів підтримуються не на всіх платформах).

Bootstrap має модульну структуру і складається переважно з наборів таблиць стилів LESS, які реалізують різні компоненти цього набору інструментів. Розробники можуть самостійно налаштовувати файли Bootstrap, обираючи компоненти для свого проекту.

Основні інструменти Bootstrap:

- сітки (grid) – наперед задані, готові до використання колонки
- шаблони (template) – фіксовані чи адаптивні шаблони сторінок
- типографіка (typography) – опис та визначення класів для шрифтів, таких як шрифти для коду, цитат тощо
- мультимедіа (media) – засоби управління зображеннями та відео
- таблиці (table) – засоби оформлення таблиць, які зокрема забезпечують сортування
- форми (form) – класи для оформлення як форм, так і деяких подій

– навігація (nav, navbar) – класи для оформлення вкладок, сторінок, меню і панелей навігації

– сповіщення (alert) – класи для оформлення діалогових вікон, підказок і спливаючих вікон

– іконочний шрифт (icon font) – набір іконок у вигляді шрифту, складається майже з 500 компонентів.

2.9 Шифрування даних за протоколом SSL

SSL (Secure Sockets Layer – рівень захищених сокетів) являє собою криптографічний протокол, який забезпечує захищену передачу інформації в Інтернеті.

Протокол SSL використовується в тих випадках, якщо потрібно забезпечити належний рівень захисту інформації, яку користувач передає серверу. На деякі сайти, які працюють з електронними грошима (банки, Інтернет-магазини, біржі контенту), передаються секретні дані. Крім пароля, це може бути номер і серія паспорта, номер кредитної картки, пін-код і ін. Для запобігання перехоплення секретних відомостей компанії Netscape Communications був створений протокол SSL [16][1)].

Протокол SSL забезпечує конфіденційний обмін даними між клієнтом і сервером. Шифрування даних відбувається з використанням відкритого ключа. Таких ключа два, і кожен з них може бути використаний як для шифрування повідомлення, так і для його розшифровки. Таким чином, якщо один ключ використовується для шифрування, то відповідно при розшифровці буде використаний інший ключ. У такій ситуації можна отримувати захищені повідомлення, публікуючи відкритий ключ та зберігаючи в таємниці секретний ключ. SSL працює за рахунок двох “підпротоколів” - протоколу SSL Record Layer (запису) і Handshake Protocol (рукостискання). Протокол SSL

[1)] [16] SSL – Вікіпедія. (загол. з екрана). URL: uk.wikipedia.org/wiki/SSL (дата звернення 14.04.2020).

запису визначає формат, що використовується для передачі даних. Протокол SSL включає рукостискання з використанням протоколу SSL записи. Таким чином, при першому з'єднанні відбувається обмін серіями повідомлень між сервером і клієнтом.

Відкриваючи сторінку сайту, на якому встановлений SSL сертіфікат, браузер запитує ідентифікаційну інформацію у веб-сервера. Далі сервер відправляє в браузер копію свого SSL сертифікату. Браузер підтверджує справжність SSL сертифікату та повідомляє про це сервера. У відповідь сервер посилає віртуально підписану угоду, що дозволяє закодовану передачу даних. З цього моменту SSL сертифікат починає шифрувати будь-які дані, які виходять від комп'ютера клієнта до сервера. При цьому звична посилання в браузері (http://www.my_domen.com.ua) перетворюється в https, де “s” означає secure- захищений, безпечний [17][1)].

[1)] [17] Как работает SSL сертификат? URL: https://ssl.com.ua/clients/knowledgebase.php? action= displayarticle&id=2 (дата звернення 14.04.2020).

3 ПРОЕКТУВАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ

3.1 Структура проекту

На рис. 3.1. зображено структуру бази даних системи обліку домашніх фінансів “Домашні рахунки”, яка відповідає Laravel моделям для зручності доступу до записів.

Структура БД була розроблена у відповідності з проведеним у розділі 1 аналізі предметної області. Всі основні сутності БД відображають функціональній схемі предметної області (рис.1.1) і табл.1.1. Опис наявних таблиць в БД наведений у табл.3.1.

Міграції для даної БД знаходяться в директорії: /database/migrations/. Скорочена структура БД приведена на рис. 3.1., повна – у додатку А.

Рисунок 3.1 – Структура бази даних

Таблиця 3.1– Опис наявних таблиць в БД

Назва таблиці в БД	Модель	Опис
users	\App\User	Особисті дані користувачів
user_accounts	\App\Model\User\Account	Всі наявні рахунки користувача
user_profit	\App\Model\User\Profit	Список доходів користувача
user_profit_category	\App\Model\ProfitCategory	Категорії доходів користувачів
user_expense	\App\Model\User\Expense	Список витрат користувача
user_expense_category	\App\Model\ ExpenseCategory	Категорії витрат користувачів
currency	\App\Currency	Список валют
statuses	\App\Status	Стан категорій та рахунків користувача (активно/неактивно)
migrations	–	Список всіх встановлених міграцій

3.2 Функціональні можливості системи

Система “Домашня бухгалтерія” розроблена для обліку особистих фінансів, ведення домашніх рахунків. Вона працює через веб-браузер, тому встановлювати додаткові програми не потрібно. Тобто це означає, що з будь-якого комп’ютера або мобільного телефону, який має вихід в інтернет, можна отримати доступ до особистого облікового запису: внести чергові операції, дізнатися поточний стан рахунку.

Користуватися системою досить просто, потрібно лише регулярно – щодня або раз на кілька днів – вносити свої видаткові та прибуткові операції. Спочатку це дасть чітке розуміння того, куди йдуть гроші. Згодом, коли в системі накопичиться фінансова історія, система покаже основні статті витрат та на які з них витрачається більше коштів, а на які менше, візуально покаже фінансову ситуацію користувача (чи витрати не перевищують доходи), а також допоможе правильно спланувати місячний бюджет.

Для того щоб розпочати ведення обліку особистих фінансів необхідно зареєструватися на сайті. Реєстрація дуже проста і не вимагає особистих конфіденційних даних. Далі вхід здійснюватиметься за допомогою зареєстрованих e-mail адреси і пароля.

Перш ніж вносити в систему свої фінансові дані варто визначити основні категорії доходів та витрат. Це можна зробити в налаштуваннях. За замовчування вже визначені декілька категорій доходів та витрат. Повний перелік наявних категорій представлений на рис.3.2. та рис.3.3.

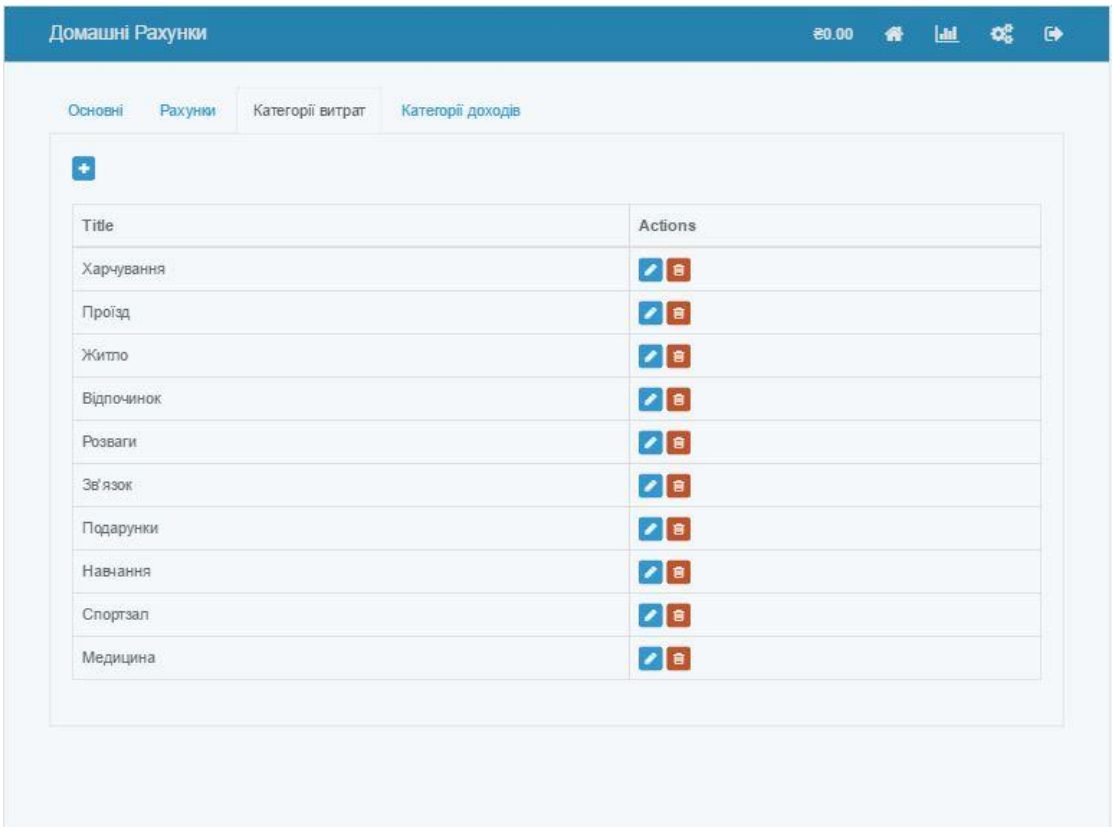

Рисунок 3.2 – Категорії витрат за замовчуванням

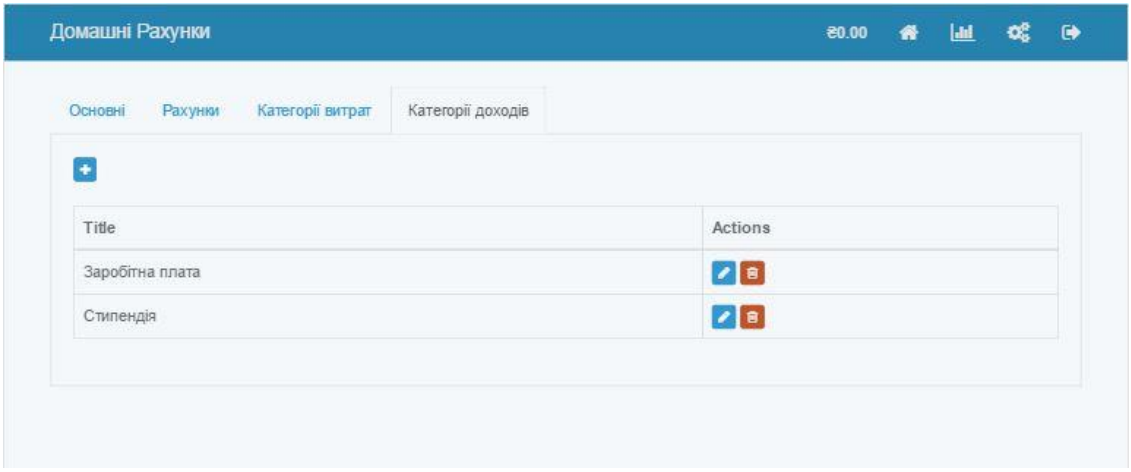

Рисунок 3.3 – Категорії доходів за замовчуванням

Пізніше можна додати нові категорії, редагувати чи видаляти старі. Вікно додавання нової категорії зображено на рис. 3.4.

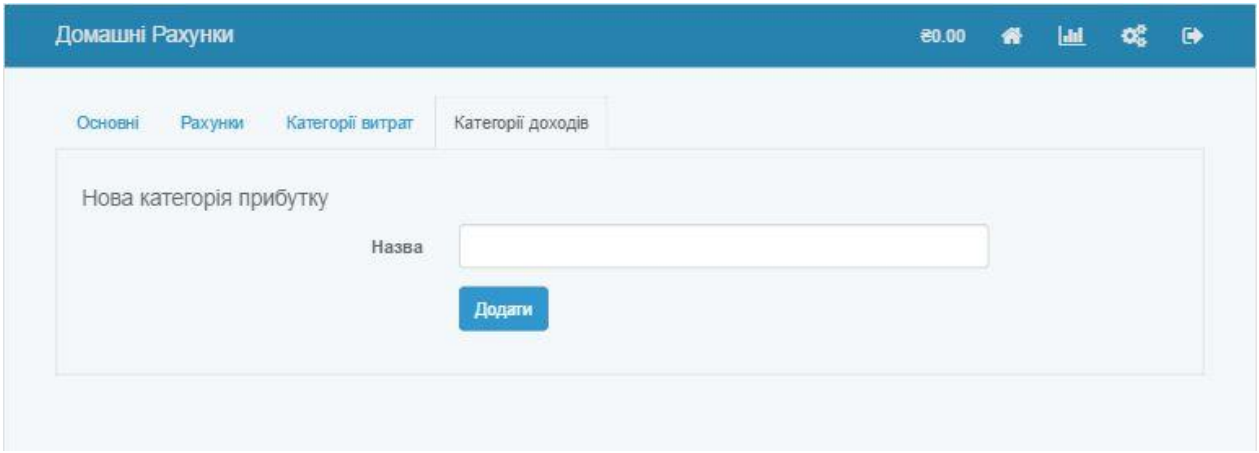

Рисунок 3.4 – Додавання нової категорії доходів

Також можна додати свої рахунки або залишити один який надається системою – “Гаманець” (рис. 3.5). Також на даній сторінці відображається вся історія доданих доходів та витрат і є можливість відслідкувати їх вказавши певний відрізок часу.

Кроки, описані вище, не обовязкові, і якщо користувача влаштовують наявні категорії витрат, доходів та рахунки, то можна відразу починати вносити фінансові дані.

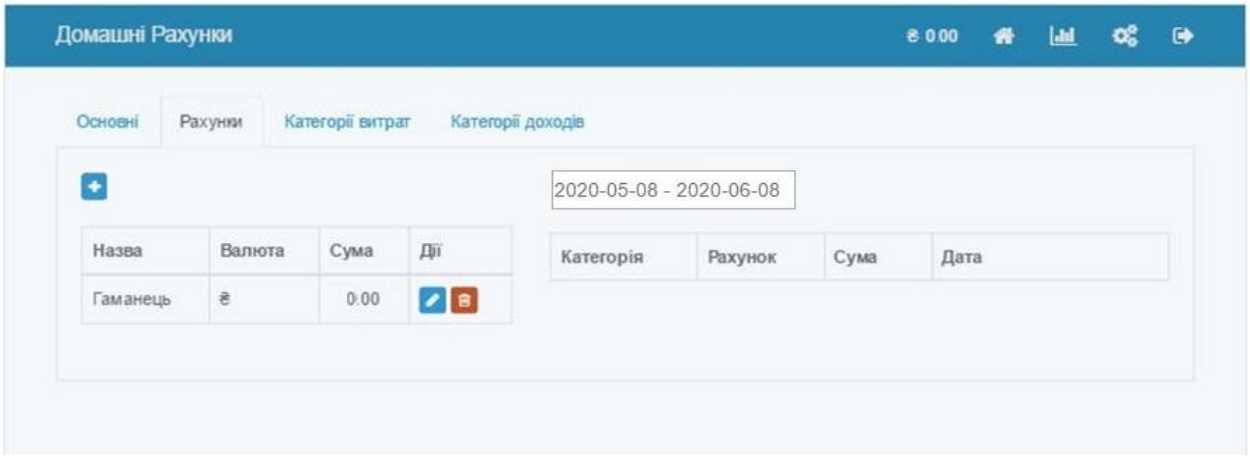

Рисунок 3.5 – Рахунки за замовчуванням

На головній сторінці зареєстрованого користувача знаходяться 3 вкладки: “Витрати” (рис.3.6.), “Доходи” (рис.3.7.), “Обмін” (рис.3.8.).

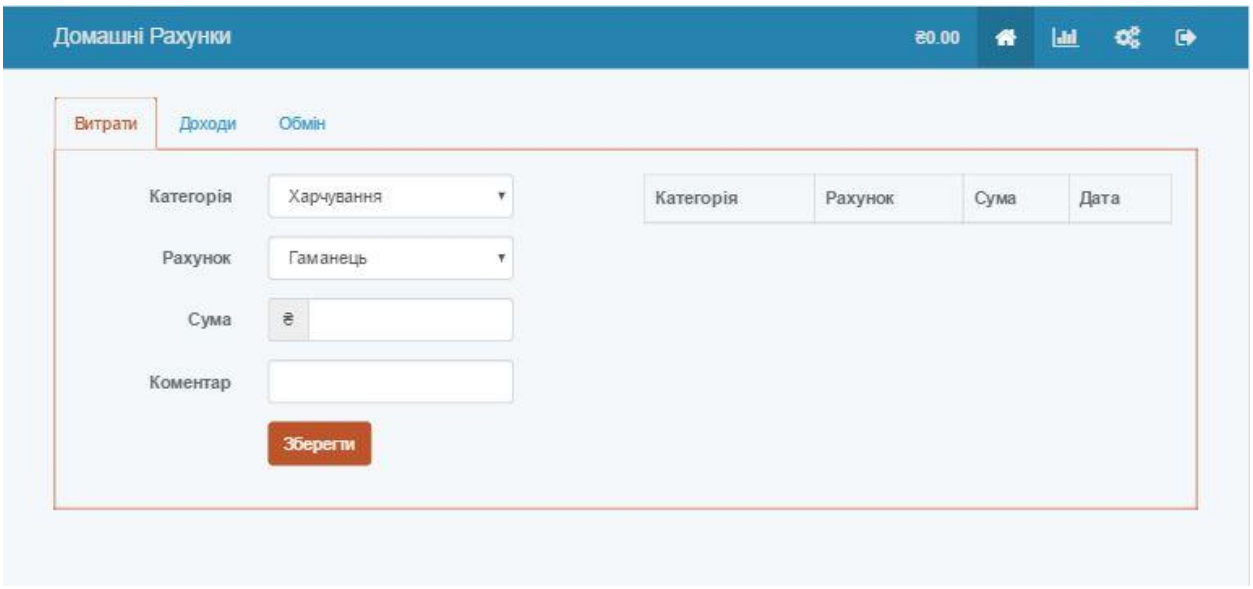

Рисунок 3.6 – Сторінка додавання витрат

Для того, щоб внести витрати чи доходи потрібно заповнити 3 основні поля та “Коментар” за бажанням (коментар відображатиметься при наведенні на доданий запис). Поля “Категорія” та “Рахунок” обираються зі списку наявних, як показано на рис.3.9. Після натиснення на кнопку “Зберегти” введені дані зявляться в таблиці поруч. Це продемонстровано на рис.3.10.

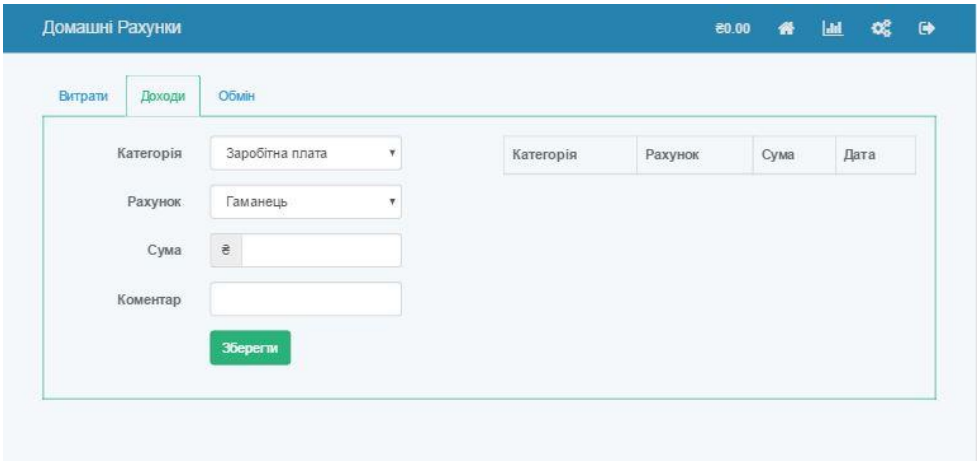

Рисунок 3.7 – Сторінка додавання доходів

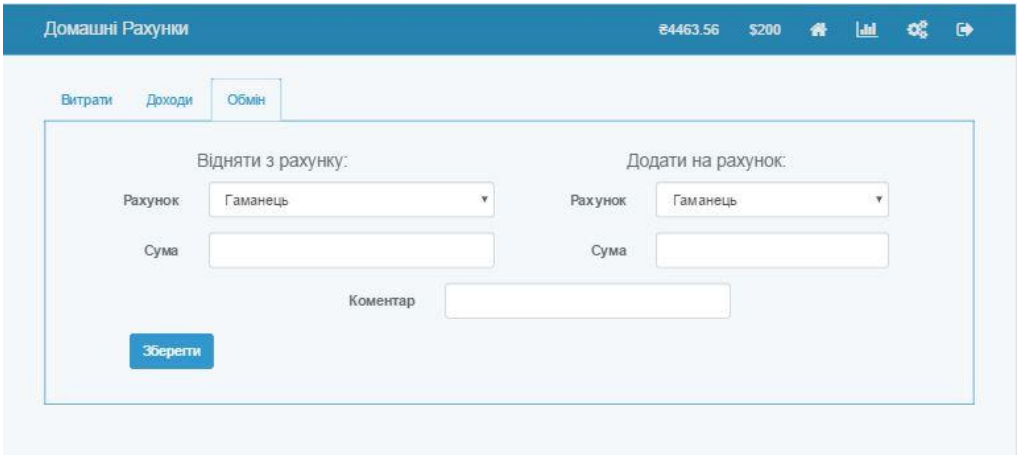

Рисунок 3.8 – Сторінка обміну між рахунками

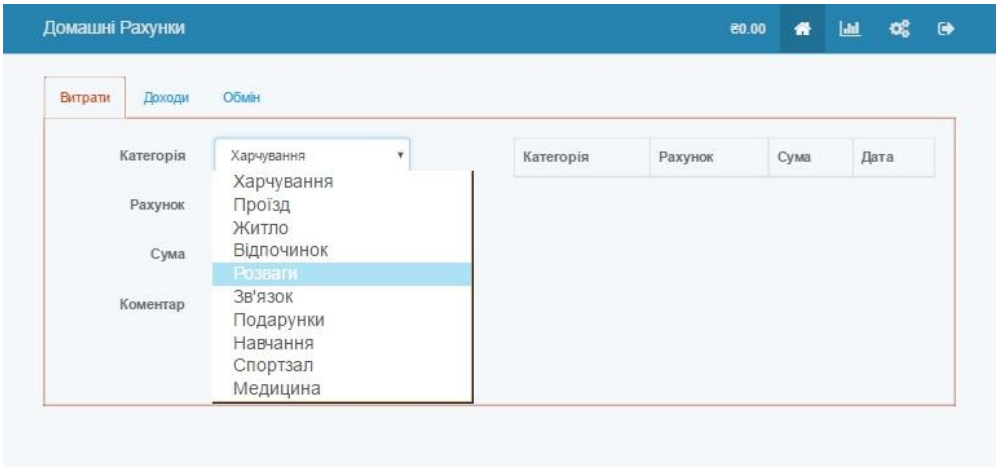

Рисунок 3.9 – Вибір категорії

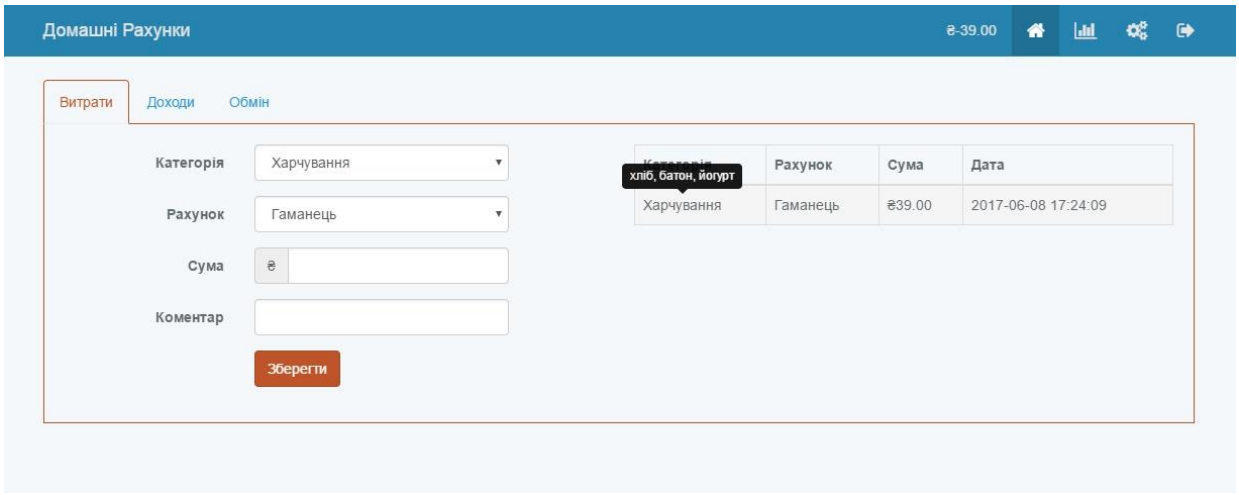

Рисунок 3.10 – Відображення доданих даних

Дана система також відображає фінансову ситуацію на графіках (рис.3.11).

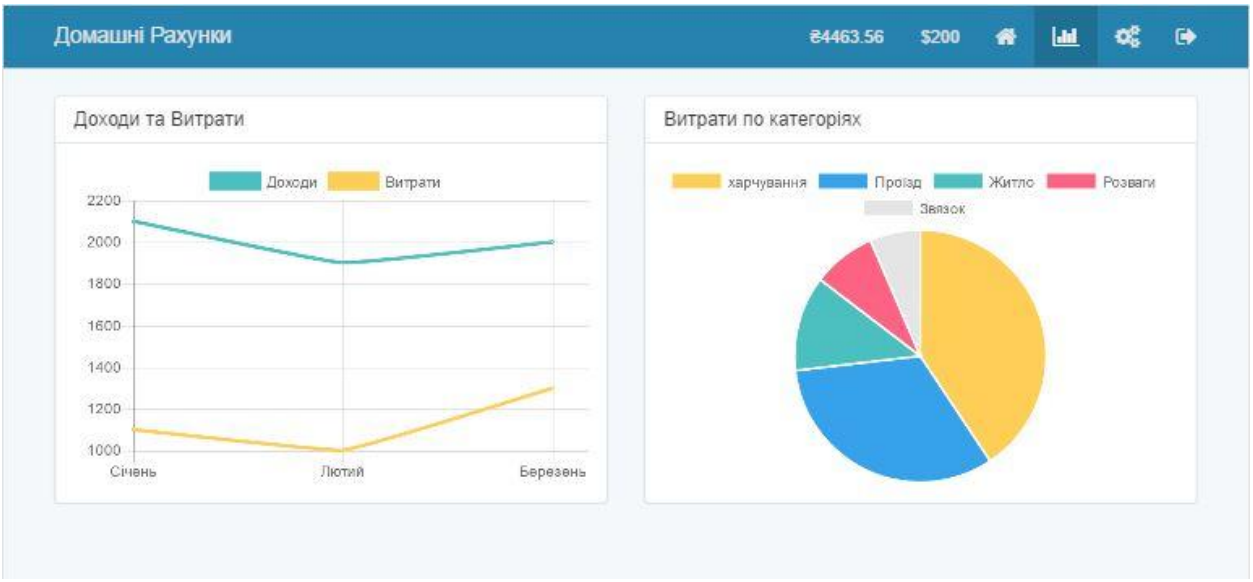

Рисунок 3.11 – Графіки доходів та витрат

Коли в системі накопичена фінансова історія на графіках можна побачити, на які статті витрат витрачається найбільше коштів, а на які – найменше. Також демонструється залежність доходів та витрат і відповідність фінансових можливостей витратам, що здійснюються.

3.3 Взаємодія з базою даних

ORM це об’єктно-реляційне представлення, і Laravel є одним з основних прихильників цієї технології. Тут вона має назву “Eloquent”, тому що дозволяє працювати з об’єктами і зв’язками баз даних, застосовуючи виразний синтаксис. Загалом, можна визначити Eloquent-модель для кожної таблиці бази даних.

Laravel в даний час підтримує наступні 4 бази даних:

– MySQL;

– Postgres;

– SQLite;

– SQL Server.

Eloquent спрощує роботу і управління з зв’язками між таблицями реляційної БД. Зокрема, Laravel підтримує наступні типи зв’язків:

– один до одного;

– один до багатьох;

– багато до багатьох;

– до багатьох через;

– поліморфічні зв’язки;

– поліморфічні зв’язки “багато до багатьох”.

3.3.1 Робота з міграціями

Міграції – щось на зразок системи контролю версій бази даних. Вони дозволяють команді змінювати її структуру, в той же час залишаючись в курсі змін інших учасників. Міграції зазвичай йдуть поруч з будівником структур для більш простого поводження з архітектурою бази даних.

Фасад Laravel Schema забезпечує підтримку створення та зміни таблиць в незалежності від використовуваної БД. Він надає єдиний, виразний і гнучкий API для всіх СУБД які підтримуються в Laravel.

Клас міграцій містить два методи: up() і down(). Метод up() використовується для додавання нових таблиць, стовпців або індексів в БД, а метод down() просто скасовує операції, виконані методом up().

В обох методах можна використовувати будівник структур Laravel для зручного створення і зміни таблиць.

У додатку Б представлена реалізація міграції, яка відповідає за створення таблиці user_profit у базі даних із відповідними полями, які описані у методі up() та видалення таблиці user_profit із бази даних, що описано в методі down().

3.3.2 Робота з моделями

Кожна таблиця має відповідний клас-модель, який використовується для роботи з цією таблицею. Моделі дозволяють повертати дані з таблиць, а також додавати в них нові записи та оновлювати вже існуючі.

Клас Illuminate\Database\Eloquent\Model успадковують всі моделі Eloquent.

Найпростіший спосіб створити екземпляр моделі – за допомогою artisan-команди make: model:

```
php artisan make: model UserProfit
```

Якщо потрібно створити міграцію БД при створенні моделі, використовується параметр --migration або -m:

```
php artisan make: model UserProfit --migration
php artisan make: model UserProfit –m
```

За замовчуванням моделі Eloquent використовують основне з’єднання з БД, налаштоване для застосування. Якщо потрібно вказати іншу сполуку для моделі, використовується властивість $connection.

Створивши моделі і пов’язані з нею таблиці, потрібно отрмати дані з БД. Кожна модель Eloquent являє собою потужний конструктор запитів, що дозволяє зручно виконувати запити до пов’язаної таблиці. Якщо наявний екземпляр моделі Eloquent можна звертатися до значень стовпців моделі, звертаючись до відповідних властивостей.

Для створення нового запису в БД достатньо створити екземпляр моделі, задати атрибути моделі і викликати метод save().

Приклад реалізації класу-моделі “Категорії прибутків”, яка віповідає за всі прибутки авторизованого користувача, продемонстровано в додатку В.

3.4. Маршрутизація ресурсу “Домашня бухгалтерія”

Файл /routes/web.php завантажується провайдером RouteServiceProvider і автоматично включається в групу посередників web, що забезпечує доступ до стану сесії і CSRF-захист. Усі маршрути додатку визначені в цьому файлі.

Маршрутизатор дозволяє реєструвати маршрути для будь-якого HTTP-запиту:

```
<?php
Auth::routes();
Route::get('/', 'HomeController@index');
Route::get('/home/profit','HomeController@profit');
Route::get('/home/exchange', 'HomeController@exchange');
Route::resource('/exchange', 'ExchangeController');
Route::resource('profit', 'ProfitController');
Route::resource('expense', 'ExpenseController');
```

Laravel також дає можливість захистити додаток від CSRF-атаки – міжсайтової підробки запитів. Міжсайтова підробка запиту – тип шкідливих атак, при яких неавторизовані команди виконуються від імені авторизованого користувача.

Laravel автоматично генерує CSRF-“токен” для кожної активної сесії користувача в додатку. Цей токен використовується для перевірки того, що запит в додаток відправляє саме авторизований користувач.

Кожен раз, визначаючи HTML-форму в своєму додатку, потрібно включати в неї приховане поле CSRF-токена, щоб посередник CSRF-захисту зміг перевірити запит. Щоб згенерувати приховане поле вводу _token, що

містить CSRF-токен, використано допоміжну функцію csrf_field(), яка повертає наступний HTML:

```
<Input type = "hidden" name = "_ token" value = "{{ php echo csrf_token() }}">
```

Вручну перевіряти CSRF-токен в запитах типу POST, PUT або DELETE не потрібно. Посередник VerifyCsrfToken, який входить до групи посередників web, автоматично перевіряє збіг токена вхідного запиту з токеном, що зберігаються в сесії.

3.5 Використання Middleware для перевірки заповнення профілю

Проміжний об’єкт (Middleware) викликається після маршрутизатора та перед викликом контролера. Він використовується для того, щоб перевірити чи авторизований користувач та перевірити чи користувач заповнив у своєму профілі категорії прибутків, категорії витрат та рахунки. Після перевірки переадресовує користувача на відповідну сторінку для заповнення профілю, або на сторінку входу. Реалізація перевірки заповнення профілю представлена у додатку Г. На рис. 3.12. зображена помилка яка виникає якщо користувач не має жодного особистого рахунку.

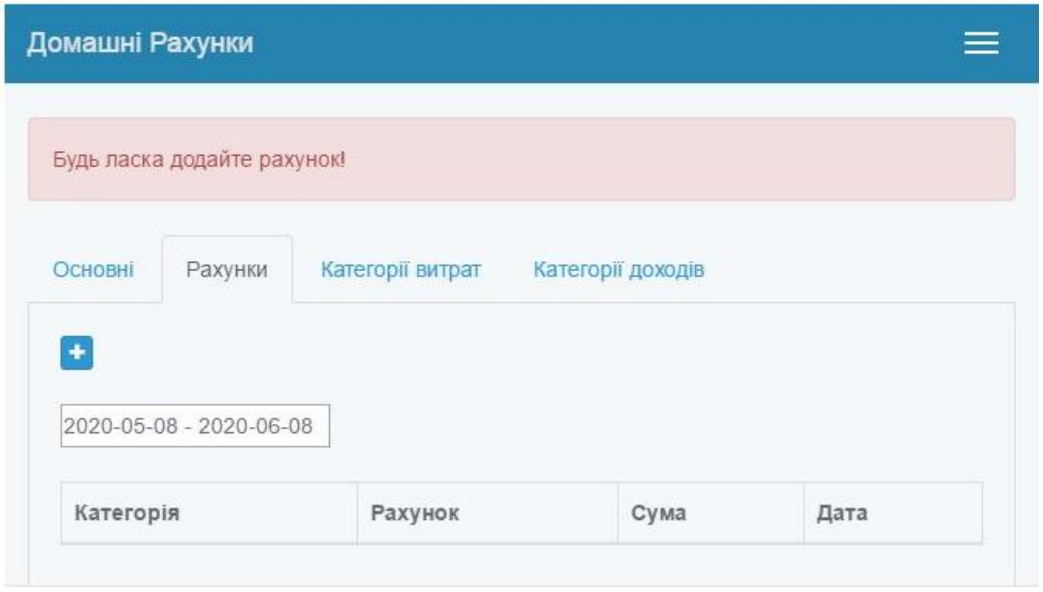

Рисунок 3.12 – Помилка заповнення профілю

3.6 Перевірка отриманих сервером даних (request)

Клас Request містить безліч методів для вивчення вхідного запиту в додатку. Він розширює клас Symfony\Component\HttpFoundation\Request.

Наприклад реалізований в додатку Ґ клас відповідає за валідацію даних, які надсилає користувач для збереження нового прибутку. Використовується у App\Http\Controllers\ProfitController, якщо перевірка даних пройшла успішно. У випадку коли дані не вірні, корустувач побачить повідомлення про помилку (рис.3.13.).

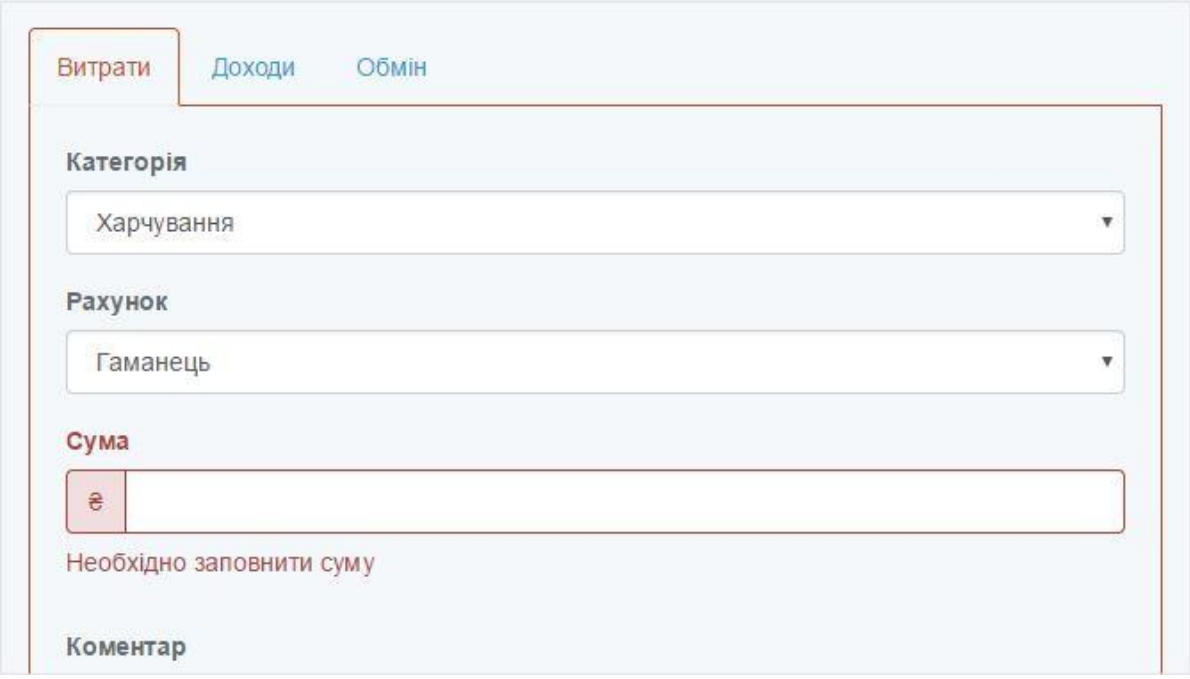

Рис. 3.13. Помилка введення даних

В даному класі наявний наступний метод:

```
public function rules() {
        return [
            'category_id' => 'required',
            'account_id' => 'required',
            'amount' => 'required',
            'comment' => 'max:255',
        ];
    }
```

Він відповідає за типи даних, які вводить користувач. Зокрема, перевіряє, щоб були заповнені обовязкові поля та довжина поля “comment” не перевищувала 255 символів.

3.7 Організація логіки обробки запитів за допомогою контролера

Замість того, щоб визначати всю логіку обробки запитів в одному файлі routes.php, можна організувати її за допомогою класів контролерів. Контролери можуть групувати пов’язану з обробкою HTTP-запитів логіку в окремий клас і зберігаються в папці app/Http/Controllers.

Всі контролери в Laravel повинні наслідувати базовий клас контролера, який вбудований в створюваний Laravel-додаток.

Для того, щоб зареєструвати маршрут для виконання певної дії (action), наприклад, вивід списку доходів у файл routes.php потрібно додати наступний рядок:

```
Route::get('/home/profit','HomeController@profit');
```

Статичний метод get відповідає http запиту, перший параметр цього методу відповідає адресному рядку браузера, другий параметр – рядок розділений на дві частини символом @, де все, що знаходиться до символу контролер, який потрібно підключити, а все що після – назва методу цього контролера. Реалізація головного контролера, який відповідає за відображення, доходів, віитрат та обміну між рахунками користувачів представлений у додатку Д.

Також можна зареєструвати маршрут контролера як ресурс наступним чином:

```
Route::resource('profit', 'ProfitController');
```

Цей єдиний виклик створює безліч маршрутів для обробки різних дій для контролеру “ProfitController”. Сам контролер має методи для наступних дій з коментарями про те, які URI і типи запитів вони обробляють. Реалізація даного контролера представлена у додатку Е.

Таблиця 3.2 – Перелік дії, які оброблюються контролером ресурсів

Тип	Шлях	Дія	Ім’я маршруту
GET	/home/profit	index	profit.index
GET	/home/profit/create	create	profit.create
POST	/home/profit	store	profit.store
GET	/home/profit{profitId}	show	profit.show
GET	/home/profit{profitId}/edit	edit	profit.edit
PUT/PATCH	/home/profit{profitId}	update	profit.update
DELETE	/home/profit{profitId}	destroy	profit.destroy

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи було реалізовано веб-сервіс “Домашня бухгалтерія”, призначена для обліку особистих фінансів.

В даний момент на ринку існує велика кількість компаній, що надають сервіс для бухгалтерського обліку. Кожна з них має свої переваги і недоліки. Проте більшість з них розраховані на малий та середній бізнес, і не всі можуть освоїти запропонований ними функціонал, що призводить до плутанини та небажання вести облік власних фінансів.

Тому одним із завдань розробленої системи є забезпечення максимальної простоти та легкості для обліку особистих коштів. Дана система може використовуватися особами різного віку для того, щоб навчитися керувати особистими коштами та зробити перший крок до фінансової грамотності.

В процесі реалізації системи “Домашня бухгалтерія” було вивчено можливі методи шифрування даних, нюанси роботи UNIX-подібних систем і налаштування веб-сервера, що працює під їх управлінням. Для написаня системи обліку домашньої бухгалтерії було використано інтегроване середовище розробки NetBeans, мова програмування PHP 5.6, СУБД MySql, фреймворк Laravel 5 та систему контролю версій Git.

Система “Домашня бухгалтерія” використовує протокол передачі даних https, який в свою чергу використовує SSL-шифрування. Таким чином, усі особисті дані, які зберігає та передає користувач системі надійно захищені.

Розроблена система може бути легко розширена новими можливостями, а також за допомогою написання RESTful API стане можливим надати доступ до даних клієнтів з Android або IOS додатків, без втручання в основну структуру проекту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Вдовенко Л.О., Сушко Н.М., Фаюра Н.Д. Фінанси : навч. посіб. Київ: Центр учбової літератури, 2010. 152 с.
2. Роль В.Ф., Сергієнко В.В., Попова С.М. Фінансове право : навч. посіб. Київ: Центр учбової літератури, 2011. 392 с.
3. Берзон Н. В. Фінанси : підруч. для бакалаврів. М. : Вид-во Юрайт, 2013. 590 с.
4. Шелудько В. М. Фінансовий менеджмент: підручник. Київ: Знання, 2006. 439 с.
5. Linux – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/Linux (дата звернення 14.04.2020
6. Ubuntu – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/Ubuntu (дата звернення 14.04.2020)
7. NetBeans – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/NetBeans (дата звернення 14.04.2020)
8. PHP – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/PHP (дата звернення 14.04.2020)
9. Кузнецов М.В, Объектно-ориентированное программирование на PHP. СПб.: «БХВ-Петербург», 2007. 608 с.
10. Дейт, К.Дж. Введение в системы баз данных. 6-е издание: Пер. с англ. М.: Вильямс, 2000. 848 с.
11. MySQL – Вікіпедія. (загол. з екрана). URL: https://uk.wikipedia.org/wiki/MySQL (дата звернення 14.04.2020).
12. Apache – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/Apache (дата звернення 14.04.2020).
13. Git – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/wiki/Git (дата звернення 14.04.2020).

14. Laravel – Вікіпедія. (загол. з екрана). URL:https://uk.wikipedia.org/ wiki/ Laravel (дата звернення 14.04.2020).
15. Bootstrap – Вікіпедія. (загол. з екрана). URL: https://uk.wikipedia.org/ wiki/ Bootstrap (дата звернення 04.01.2020).
16. SSL – Вікіпедія. (загол. з екрана). URL: uk.wikipedia.org/wiki/SSL (дата звернення 14.04.2020).
17. Как работает SSL сертификат? URL: https://ssl.com.ua/ clients/knowledgebase. php? action= displayarticle&id=2 (дата звернення 14.04.2020).