МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

«Метод двофакторної аутентифікації Nexmo на основі веб-ресурсу»

Завідувач
випускаючої кафедри **Довбиш А.С.**

Керівник роботи **Проценко О.Б.**

Студентка групи ІН.м-92 **Дуркіна А.І.**

СУМИ 2020

Сумський державний університет
(назва вузу)

Факультет *ЕлІТ* Кафедра *Комп'ютерних наук*
Спеціальність *«Інформатика»*

Затверджую:
зав.кафедрою _____
_____
"_____" _____ 20___р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТОВІ

Дуркінній Анастасії Ігорівні
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) *Метод двофакторної автентифікації Nexmo на основі веб-ресурсу*

затверджую наказом по інституту від "____" _____ 20 ___ р. № _____

2. Термін здачі студентом закінченого проекту (роботи)_____

3. Вхідні данні до проекту (роботи) _____
_____
_____
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Аналіз проблеми та огляд літератури за темою; 2) Постановка завдання й формування завдань дослідження; 3) Аналіз вибранного методу двофакторної автентифікації; 4) Опис моделі веб-ресурсу; 5) Розробка веб-ресурсу з обраним методом двофакторної автентифікації користувача; 5) Аналіз отриманих результатів.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _
_____
_____
_____
_____

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту (роботи)	Термін виконання проекту (роботи)	Примітка
1.	Аналіз проблем на основі існуючих аналогів.		
2.	Постановка задачі та формування завдань дослідження		
3.	Опис архітектури веб-ресурсу		
4.	Розробка веб-ресурсу з використанням методу двофакторної автентифікації		
5.	Оформлення пояснювальної записки до кваліфікаційної магістерської роботи		

Студент – дипломник _____
(підпис)

Керівник проекту _____

(підпис)

РЕФЕРАТ

Записка: 67 стор., 17 рис., 1 додаток, 15 літературних джерел.

Об'єкт дослідження — метод двофакторної автентифікації на основі веб-ресурсу.

Мета роботи — забезпечення достовірності автентифікації користувачів за рахунок застосування методу двофакторної автентифікації.

Методи дослідження — інформаційний аналіз, метод моделювання, методи розробки, що базуються на мові програмування PHP, фреймворку Laravel.

Результати — реалізована модель доступу до даних веб-ресурсу на базі методу двофакторної автентифікації. Метод удосконалить авторизацію користувача і створить більш стійку систему автентифікації. Програму реалізовано мовами PHP, JavaScript та на основі фреймворку Laravel. Програма реалізована як інтернет-магазин.

ВЕБ-РЕСУРС, PHP, БАЗА ДАНИХ, ДВОФАКТОРНА АВТЕНТИФІКАЦІЯ, АВТОРИЗАЦІЯ, NEXMO, JSON, ВЕРИФІКАЦІЯ, LARAVEL, ОНЛАЙН-ЗАМОВЛЕННЯ.

ЗМІСТ

ВСТУП

У наші часи інформаційні технології стрімко розвиваються. Робити покупки в магазині, купувати газети, навіть інколи ходити на роботу не обов'язково. Достатньо мати в своєму мобільному телефоні, комп'ютері чи планшеті доступ до Інтернету. Тоді покупки можна зробити в інтернет-магазині, новини прочитати на достовірному сайті новин, а всю роботу виконати на комп'ютері (ноутбуці) і мітинг з колегами організувати, наприклад, в Zoom (сервіс для онлайн-конференції, тощо). Насправді, таких прикладів про те, як діджиталізація в багатьох сферах життя полегчує виконання багатьох процесов, дуже багато.

Після недавніх подій у світі, коли через пандемію коронавірусу, всі заклади громадського харчування, магазини (за виключенням продуктових та аптек), навчальні заклади вимушені були закриватися. Багато підприємців задумались і впровадили серйозні зміни задля того, аби зберегти свій бізнес. А саме вирішенням стали інтернет-магазини, замовлення їжі та подальша їх доставки додому.

Друге питання, яке є неменш важливим, це збереження данних користувача. Адже з ростом кількості інтернет-продажів, зростає і кількість шахраїв, які хочуть нажитися на цьому. Тому задля того, аби зберігати дані клієнтів у безпеці, необхідно впроваджувати багатофакторну автентифікацію, яка характеризується тим, що користувач вводить не просто свій номер телефону і пароль, а ще й, наприклад, код, який надсилається на його телефон кожного разу, коли він авторизується в системі.

Основною задачею роботи є створення системи двофакторної автентифікації на основі веб-ресурсу (інтернет-магазину), яка буде зручною і зрозумілою для користувача.

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Загальна інформація про веб-сайти та веб-сервер

Веб-сайти, без сумніву, є найважливішим елементом Інтернету.

Веб-сайт складається з початкової, головної сторінки, на яку користувач потрапляє як тільки переходить по посиланню, та інших сторінок. Усі сторінки мають однакове доменне ім’я і зберігаються на одному веб-сервері. [3]

Загальнодоступна мережа Інтернет-протоколу (IP), наприклад, Інтернет, дає доступ до веб-сайтів.

Сайти мають багато класифікацій і можуть виконувати зовсім різні функції.

Існують такі популярні види сайтів:

1. Блог - Інтернет-журнал чи інформаційна сторінка, які регулярно оновлюються.
2. Бізнес - загальний веб-сайт, що детально описує всі сфери бізнесу.
3. Брошура - одна або дві сторінки веб-сайту, що відображають основну інформацію про компанію.
4. Краудфандинг - використовується для прийняття "застави" та пожертв на будь-яку конкретну справу.
5. Електронна комерція(інтернет-магазин) - веб-сайт, що продає товари чи послуги.
6. Освітні - як правило, інтерактивний веб-сайт з освітньою інформацією з будь-якої заданої теми.
7. Медіа чи розваги - регулярно оновлюваний вміст про поточні події, спорт та розваги.

8. Некомерційна організація - подібна до бізнес веб-сайтів, але заохочує користувачів допомагати з будь-якою метою.

9. Особисте - резюме в Інтернеті, що відображає досвід роботи та навички.

10. Портал - зазвичай внутрішні веб-сайти для шкіл або підприємств, де користувачі входять в систему.

11. Портфоліо - онлайн-портфоліо, що відображає творчі роботи власника сайту.

На сьогодні існує мільйони веб-сторінок, але жодна з них не існувала 20 років тому.

Перша веб-сторінка розпочала роботу 6 серпня 1991 року. Вона була присвячена інформації про проект Всесвітньої мережі, її створив Тім Бернерс-Лі. Він працював на комп'ютері NeXT при Європейській організації ядерних досліджень, ЦЕРН і виглядала вона в 1992 так, як на рисунку 1.1.

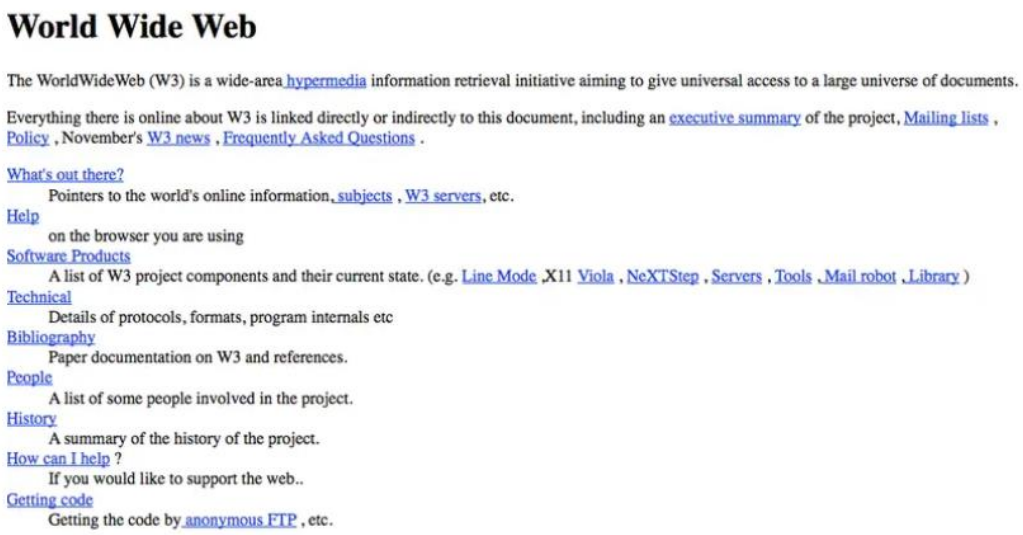

Рисунок 1.1 – Вигляд першого в світі веб-сайту http://info.cern.ch

З того часу кількість веб-ресурсів почала стрімко зростати, але кількість активних сайтів все ж таки є не такая велика в порівнянні з загальною кількістю хостів (рисунок 1.2)

Рисунок 1.2 – Статистика росту загальної кількості веб-сайтів (в тому числі активних)

Веб-сторінки, з яких складається веб-сайт, - документи, які, звичайно, представлені у вигляді звичайного тексту, який поєднаний з HTML, XHTML (інструкції форматування мови гіпертекстової розмітки). Сторінки можуть мати елементи з інших веб-ресурсів з відповідними розмітковими якорями. Доступ до веб-сторінок здійснюється за допомогою протоколу передачі гіпертексту (HTTP), який може додатково використовувати шифрування (HTTP Secure, HTTPS) для забезпечення безпеки та конфіденційності для користувача. Застосування користувача, часто веб-браузер, надає вміст сторінки відповідно до своїх інструкцій з розмітки HTML на дисплейному терміналі.[5]

Гіперпосилання між веб-сторінками направляє навігацію по сайту та передає читачеві структуру сайту, яка частіше за все починається з головної (домашньої сторінки), що містить каталог веб-контенту даного веб-сайту. Деякі веб-сайти вимагають авторизації та попередньої реєстрації користувача, або ж взагалі підписки для доступу до веб-ресурсу. До прикладів веб-сайтів з підпискою часто відносять ділові сайти, новинні веб-

сайти. сайти акамедічних журналів, онлайн-ігри, веб-сайти для файлообміну, дошки оголошення, онлайн поштові скриньки, соціальні мережі, а також інші сайти, яка надають інші послуги. Кінцевий користувач може отримувати доступ до веб-сайтів на різних гаджетах, таких як настільні та портативні комп'ютери, планшети, смартфони та навіть смарт-телевізори.

Веб-сервер - це комп'ютер, на якому працюють веб-сайти. Це комп'ютерна програма, яка розповсюджує веб-сторінки у міру їх реквізиції. Основною метою веб-сервера є зберігання, обробка та доставка веб-сторінок користувачам. Цей взаємозв'язок здійснюється за допомогою протоколу передачі гіпертексту (HTTP). Ці веб-сторінки є переважно статичним вмістом, що включає документи HTML, зображення, таблиці стилів, тести тощо. Окрім HTTP, веб-сервер також підтримує протоколи SMTP (Простий протокол передачі пошти) та FTP (Протокол передачі файлів) для надсилання електронної пошти та передачі файлів та зберігання. [3]

Примітивна робота веб-серверу зображена на рисунку 1.3.

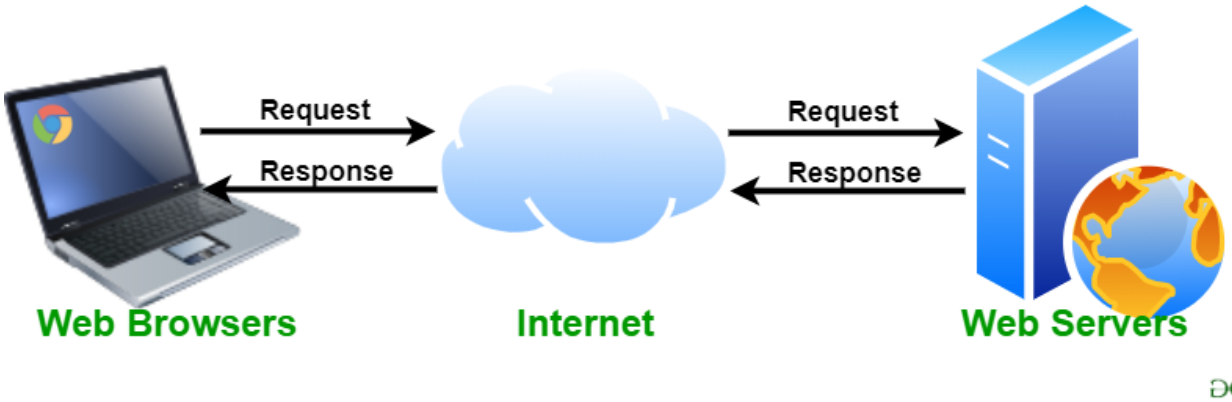

Рисунок 1.3 – Базове представлення роботи системи веб-браузеру та веб-серверу

Основною задачею веб-сервера є відображення вмісту веб-сайту. Якщо веб-сервер не є загальнодоступним і використовується всередині, тоді

він називається Intranet Server. Коли користувач звертається до веб-сайту, додавши URL-адресу або веб-адресу в адресному рядку веб-браузера (наприклад, Chrome або Firefox) (наприклад, www.google.com), браузер надсилає запит в Інтернет на перегляд відповідної веб-сторінки для цього адресу. Сервер доменних імен (DNS) перетворює цю URL-адресу на IP-адресу (наприклад, 192.168.216.345), яка в свою чергу вказує на веб-сервер.

Веб-серверу пропонується представити веб-сайт із вмістом у браузері користувача. Усі веб-сайти в Інтернеті мають унікальний ідентифікатор з точки зору IP-адреси. Ця адреса Інтернет-протоколу використовується для зв'язку між різними серверами в Інтернеті. У наші дні сервер Apache - найпоширеніший веб-сервер, доступний на ринку. Apache - це програмне забезпечення з відкритим кодом, яке обробляє майже 70 відсотків усіх веб-сайтів, доступних сьогодні. Більшість веб-програм використовують Apache як стандартне середовище веб-сервера. Іншим загальнодоступним веб-сервером є Інформаційна служба Інтернету (IIS). IIS належить Microsoft.

1.2 Веб-ресурси в людській життєдіяльності

Веб-ресурси стали невід’ємною частиною людського життя, через це їх стає дедалі більше з кожним днем.

За допомогою безлічі веб-ресурсів стає набагато легше здійснювати багато різних операцій, починаючи від онлайн-банкінгу, завершуючи переглядом та замовлення товарів.

Швидкість, зручність та достовірність стали найважливішими критеріями, через які користувачі обирають саме веб-ресурси. Адже в сучасному світі кожен цінує свій час і не хоче гаяти його на непотрібні речі.

Онлайн-шопінг, як один із видів веб-ресурсів, стає все ширшим і захоплює все більше галузей торгівлі. Навіть є сайти, де можно придбати, наприклад, доступ до записів мастер-класів, цінних відео-уроків, вебінарів, тощо.

Інтернет-магазин дає змогу користувачеві власноруч в онлайн режимі ознайомитися з товаром, його описом, характеристиками, ціною, видом та вартістю доставки. Уже не потрібно ставити в очному режимі питанню касиру чи консультанту. Звичайно на кожному сайті є можливість замовлення дзвінку в разі виникнення запитання. На рисунку 1.4 зображені переваги онлайн-шопінгу з точки зору покупця.

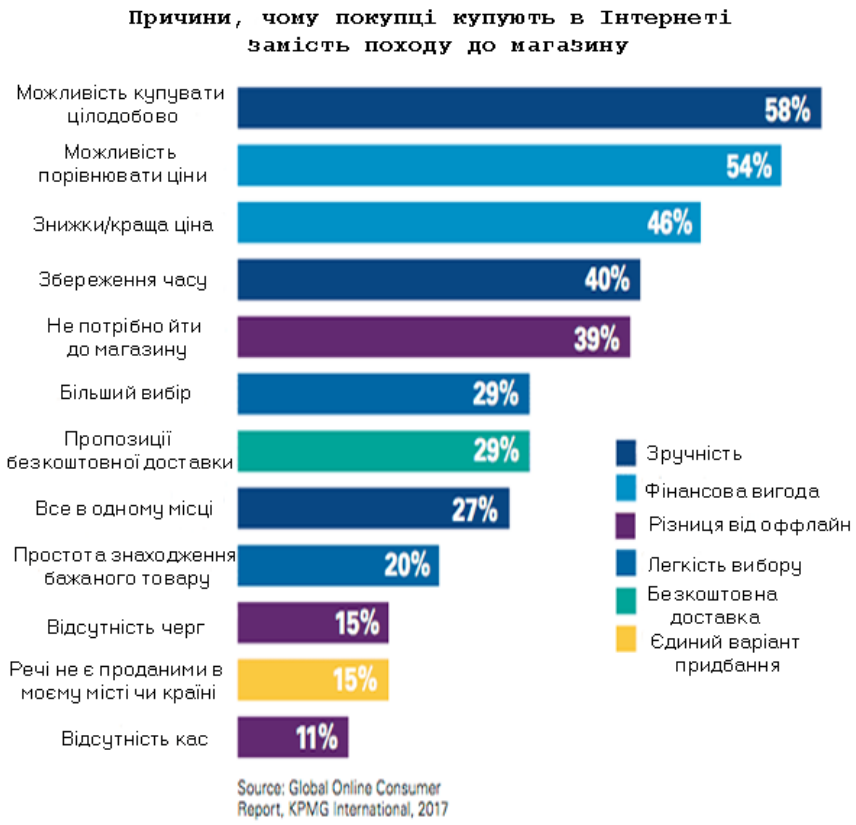

Рисунок 1.4 – Переваги покупок онлайн

1.3 Авторизація та автентифікація

Автентифікація - це процес визначення, хтось чи щось насправді є тим, ким або чим він себе оголошує. Технологія автентифікації забезпечує контроль доступу для систем, перевіряючи, чи відповідають облікові дані користувача даним в базі даних авторизованих користувачів або на сервері автентифікації даних. [15]

Зазвичай користувачів ідентифікують за допомогою ідентифікатора користувача(логіна), а автентифікація здійснюється, коли користувач надає

облікові дані, наприклад пароль, який відповідає цьому ідентифікатору користувача. Більшість користувачів найбільш добре знайомі з використанням пароля, який як частина інформації, яка повинна бути відома лише користувачеві, називається фактором автентифікації знань.

Автентифікація важлива, оскільки вона дозволяє організаціям захищати свої мережі, дозволяючи лише автентифікованим користувачам (або процесам) доступ до захищених ресурсів, які можуть включати комп'ютерні системи, мережі, бази даних, веб-сайти та інші мережеві додатки чи послуги.

Після автентифікації користувач або процес, як правило, також піддається процедурі авторизації, щоб визначити, чи повинен автентифікований суб'єкт мати доступ до захищеного ресурсу або системи. Користувача можна автентифікувати, але не отримати доступу до ресурсу, якщо користувачеві не було надано дозвіл на доступ до нього.

Терміни автентифікація та авторизація часто використовуються як взаємозамінні; хоча їх часто можна реалізувати разом, ці дві функції відрізняються. Хоча автентифікація - це процес перевірки ідентичності зареєстрованого користувача перед тим, як дозволити доступ до захищеного ресурсу, авторизація - це процес перевірки того, що автентифікованому користувачеві надано дозвіл на доступ до запитуваних ресурсів. Процес обмеження доступу до цих ресурсів певною кількістю користувачів називається контролем доступу. Процес автентифікації завжди відбувається перед процесом авторизації.[15]

Рисунок 1.5 – Автентифікація та авторизація

Автентифікація користувачів відбувається в більшості взаємодій від людини до комп'ютера за межами гостьових облікових записів, автоматично зареєстрованих облікових записів та Kiosk комп'ютерних систем. Як правило, користувач повинен вибрати ім'я користувача або ідентифікатор користувача та надати дійсний пароль, щоб почати користуватися системою. Автентифікація користувача дозволяє взаємодію людини з машиною в операційних системах та додатках, а також дротових та бездротових мережах, щоб забезпечити доступ до мережевих та підключених до Інтернету систем, програм та ресурсів.

Багато компаній використовують автентифікацію для перевірки користувачів, які входять на їх веб-сайти. Без належних заходів безпеки дані користувачів, такі як номери кредитних та дебетових карток, а також номери соціального страхування, можуть потрапити в руки кіберзлочинців.

Організації також використовують автентифікацію, щоб контролювати, які користувачі мають доступ до корпоративних мереж та ресурсів, а також визначати та контролювати, які машини та сервери мають доступ. Компанії також використовують автентифікацію, щоб віддалені співробітники могли безпечно отримати доступ до своїх програм та мереж. Для підприємств та інших великих організацій автентифікація може здійснюватися за допомогою системи єдиного входу (SSO), яка надає доступ до кількох систем з одним набором облікових даних для входу.

Під час автентифікації облікові дані, що надаються користувачем, порівнюються з даними, що містяться у файлі в базі даних авторизованих даних користувачів або в локальній операційній системі, або через сервер автентифікації. Якщо облікові дані збігаються, а автентифікована особа має право використовувати ресурс, процес завершується, і користувачеві надається доступ. Повернені дозволи та папки визначають як середовище,

яке бачить користувач, так і спосіб його взаємодії з ним, включаючи години доступу та інші права, такі як обсяг місця для зберігання ресурсів.

1.4 Огляд методів автентифікації

Під час автентифікації облікові дані, що надаються користувачем, порівнюються з даними, що містяться у файлі в базі даних авторизованих даних користувачів або в локальній операційній системі, або через сервер автентифікації. Якщо облікові дані збігаються, а автентифікована особа має право використовувати ресурс, процес завершується, і користувачеві надається доступ. Повернені дозволи та папки визначають як середовище, яке бачить користувач, так і спосіб його взаємодії з ним, включаючи години доступу та інші права, такі як обсяг місця для зберігання ресурсів.

Традиційно автентифікація здійснювалася за допомогою систем або ресурсів, до яких здійснювався доступ; наприклад, сервер автентифікує користувачів за допомогою власної системи паролів, реалізованої локально, використовуючи ідентифікатори входу (імена користувачів) та паролі. Знання облікових даних для входу передбачається, щоб гарантувати автентичність користувача. Кожен користувач реєструється спочатку (або реєструється кимось іншим, наприклад, системним адміністратором), використовуючи присвоєний або самодекларований пароль. При кожному наступному використанні користувач повинен знати і використовувати раніше заявлений пароль.

Однак протоколи веб-додатків, HTTP та HTTPS, не мають стану, що означає, що сувора автентифікація вимагатиме повторної автентифікації кінцевих користувачів кожного разу, коли вони отримують доступ до ресурсу за допомогою HTTPS. Замість того, щоб обтяжувати кінцевих користувачів цим процесом для кожної взаємодії в Інтернеті, захищені системи часто покладаються на аутентифікацію на основі токенів, при якій автентифікація виконується один раз на початку сеансу. Система

автентифікації видає підписаний маркер автентифікації для програми кінцевого користувача, і цей маркер додається до кожного запиту від клієнта.

Аутентифікація сутності для систем та процесів може здійснюватися за допомогою машинних облікових даних, які працюють як ідентифікатор користувача та пароль, за винятком того, що ці дані автоматично подаються відповідним пристроєм. Вони також можуть використовувати цифрові сертифікати, які були видані та перевірені центром сертифікації як частина інфраструктури відкритого ключа, для автентифікації особи під час обміну інформацією через Інтернет.

Автентифікація користувача за допомогою ідентифікатора користувача та пароля зазвичай вважається основним типом автентифікації, і це залежить від того, що користувач знає дві частини інформації: ідентифікатор користувача або ім’я користувача та пароль. Оскільки цей тип автентифікації покладається лише на один фактор автентифікації, це тип однофакторної автентифікації.

Сильна автентифікація - це термін, який формально не визначений, але зазвичай використовується для того, щоб визначити тип автентифікації, який є надійнішим та стійкішим до атак; досягнення того, що, як загальновизнано, вимагає використання принаймні двох різних типів факторів автентифікації.[15]

Коефіцієнт автентифікації представляє деякий фрагмент даних або атрибут, який можна використовувати для автентифікації користувача, що вимагає доступу до системи. У старій приказці щодо безпеки чинниками автентифікації можуть бути "те, що ви знаєте, те, що у вас є, або щось, що ви є". Ці три фактори відповідають фактору знань, фактору володіння та фактору спадковості. Протягом останніх років було запропоновано та застосовано додаткові фактори, причому місце розташування у багатьох випадках є четвертим фактором, а час - п’ятим фактором.

На даний момент використовуються фактори автентифікації:

- Фактор знань: "Те, що ви знаєте".

Фактором знань можуть бути будь-які дані автентифікації, які складаються з інформації, якою володіє користувач, включаючи персональний ідентифікаційний номер (ПІН), ім'я користувача, пароль або відповідь на секретне запитання.

- Фактор володіння: "Щось у вас є".

Коефіцієнтом володіння можуть бути будь-які облікові дані, засновані на предметах, якими користувач може володіти та носити їх із собою, включаючи апаратні пристрої, такі як маркер безпеки або мобільний телефон, що використовується для прийняття текстового повідомлення або запуску програми автентифікації, яка може генерувати одноразову пароль або PIN-код.

- Фактор взаємозв'язку: "Щось ти є".

Фактор невід'ємності зазвичай заснований на певній формі біометричної ідентифікації, включаючи відбитки пальців або великих пальців, розпізнавання обличчя, сканування сітківки або будь-яку іншу форму біометричних даних.

- Фактор місцезнаходження: "Де ти знаходишся".

Хоча він може бути менш конкретним, фактор місцезнаходження іноді використовується як доповнення до інших факторів. Місцезнаходження можна визначити з достатньою точністю за допомогою пристроїв, обладнаних GPS, або з меншою точністю, перевіряючи мережеві маршрути. Фактор місцезнаходження, як правило, не може стояти самостійно для автентифікації, але він може доповнювати інші фактори, забезпечуючи спосіб виключення деяких запитів. Наприклад, це може перешкодити зловмиснику, який знаходиться у віддаленому географічному районі, виглядати як користувач, який зазвичай входить в систему лише з дому чи офісу в країні проживання організації.

- Фактор часу: "Коли ви проходите автентифікацію."

Як і фактор розташування, фактор часу сам по собі недостатній, але він може бути додатковим механізмом відсіювання зловмисників, які намагаються отримати доступ до ресурсу в той час, коли цей ресурс недоступний авторизованому користувачеві. Він також може використовуватися разом з місцем розташування. Наприклад, якщо останню автентифікацію користувача здійснено опівдні в США, спроба автентифікації з Азії через годину буде відхилена на основі поєднання часу та місцезнаходження.

Незважаючи на те, що вони використовуються як додаткові фактори автентифікації, розташування користувача та поточний час самі по собі не є достатніми, принаймні без одного з перших трьох факторів, для автентифікації користувача. Однак всюдисущість смартфонів допомагає полегшити тягар багатофакторної автентифікації для багатьох користувачів. Більшість смартфонів оснащені GPS, що забезпечує достатню впевненість у підтвердженні місця входу; MAC-адреси смартфонів також можуть бути використані для автентифікації віддаленого користувача, незважаючи на те, що MAC-адреси відносно легко підробляти.

Додавання факторів автентифікації до процесу автентифікації зазвичай покращує безпеку. Сильна автентифікація зазвичай відноситься до автентифікації, яка використовує принаймні два фактори, де ці фактори мають різний тип. Розрізнення важливо; оскільки і ім’я користувача, і пароль можна вважати типами фактора знань, можна сказати, що для автентифікації базового імені користувача та пароля для аутентифікації використовуються два фактори знань - однак, це не можна вважати формою двофакторної автентифікації (2FA). Так само для систем автентифікації, які покладаються на "питання безпеки", які також є "чимось, що ви знаєте", доповнюючи ідентифікатор користувача та паролі.

Двофакторна автентифікація зазвичай залежить від фактора знань, поєднаного з біометричним фактором або фактором володіння, як маркер безпеки. Багатофакторна автентифікація може включати будь-який тип автентифікації, який залежить від двох або більше факторів, але процес автентифікації, який використовує пароль плюс два різні типи біометричних даних, не буде розглядатися як трифакторна автентифікація, хоча якби процес вимагав фактора знання, володіння коефіцієнт і фактор невід'ємності, це могло б бути. Системи, що вимагають цих трьох факторів, а також географічного чи часового фактора, вважаються прикладами чотирифакторної автентифікації.

Традиційна автентифікація залежить від використання файлу паролів, в якому ідентифікатори користувачів зберігаються разом із хешами паролів, пов'язаних з кожним користувачем. Під час входу пароль, надісланий користувачем, хешується і порівнюється зі значенням у файлі паролів. Якщо два хеші збігаються, користувач аутентифікується.

Цей підхід до автентифікації має кілька недоліків, особливо щодо ресурсів, розгорнутих у різних системах. З одного боку, зловмисники, які можуть отримати доступ до файлу паролів для системи, можуть застосовувати атаки грубої сили проти хешованих паролів для вилучення паролів. З іншого боку, такий підхід потребує багаторазової автентифікації для сучасних додатків, які отримують доступ до ресурсів у багатьох системах.

Слабкі місця автентифікації на основі пароля можна в деякій мірі усунути за допомогою розумніших імен користувачів та правил паролів, таких як мінімальна довжина та умови складності, такі як включення великих літер та символів. Однак автентифікація на основі пароля та автентифікація на основі знань є більш вразливими, ніж системи, які потребують декількох незалежних методів.

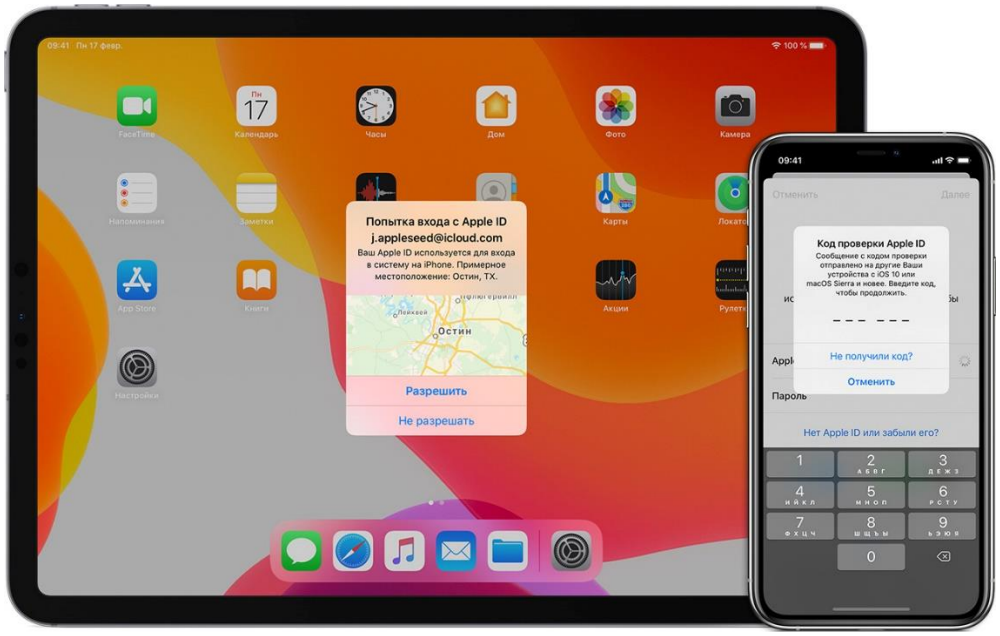

Рисунок 1.6 – Приклад двофакторної автентифікації через підтвердження входу до облікового запису через інший пристрій

Інші методи автентифікації включають:

- Двофакторна автентифікація додає додатковий рівень захисту процесу аутентифікації. 2FA вимагає, щоб користувач надав другий фактор автентифікації на додаток до пароля. Системи 2FA часто вимагають від користувача введення коду підтвердження, отриманого за допомогою текстового повідомлення на попередньо зареєстрованому мобільному телефоні, або коду, згенерованого програмою автентифікації.
- Багатофакторна автентифікація вимагає від користувачів автентифікації з кількома факторами автентифікації, включаючи біометричний фактор, такий як розпізнавання відбитків пальців або обличчя, фактор володіння, як брелок безпеки або маркер, згенерований програмою автентифікації.
- Одноразовий пароль - це автоматично згенерований числовий або буквено-цифровий рядок символів, який автентифікує користувача. Цей пароль дійсний лише для одного сеансу входу або транзакції і зазвичай використовується для нових користувачів або для користувачів, які

втратили свої паролі та отримали одноразовий пароль для входу та зміни на новий пароль.

- Трифакторна автентифікація (3FA) - це тип МФА, який використовує три фактори автентифікації, як правило, фактор знань (пароль) у поєднанні з фактором володіння (маркер безпеки) та коефіцієнтом безперервності (біометричний).
- Біометрія - Хоча деякі системи автентифікації можуть залежати виключно від біометричної ідентифікації, біометричні дані зазвичай використовуються як другий або третій фактор автентифікації. Найпоширеніші типи біометричної автентифікації включають сканування відбитків пальців, сканування обличчя або сітківки та розпізнавання голосу.
- Мобільна автентифікація - це процес перевірки користувача через їх пристрої або перевірки самих пристроїв. Це дозволяє користувачам входити в безпечні місця та ресурси з будь-якого місця. Процес мобільної автентифікації включає багатофакторну автентифікацію, яка може включати одноразові паролі, біометричну автентифікацію або перевірку QR-коду.
- Безперервна автентифікація - за умови безперервної автентифікації, замість того, щоб користувач входив або виходив із системи, програма компанії постійно обчислює "оцінку автентифікації", яка вимірює, наскільки впевненим, що власником облікового запису є особа, яка використовує пристрій.
- Автентифікація API - стандартними методами управління автентифікацією API є: базова аутентифікація HTTP; Ключі API та OAuth.

У базовій аутентифікації HTTP сервер запитує у клієнта інформацію про аутентифікацію, тобто ім’я користувача та пароль. Потім клієнт передає інформацію про аутентифікацію серверу в заголовку авторизації.

У методі автентифікації ключа API першому користувачеві присвоюється унікальне згенероване значення, яке вказує на те, що користувач відомий. Потім щоразу, коли користувач намагається знову

увійти в систему, його унікальний ключ використовується для підтвердження того, що це той самий користувач, який раніше входив у систему.

Відкрита авторизація (OAuth) - це відкритий стандарт автентифікації та авторизації на основі токенів в Інтернеті. OAuth дозволяє використовувати інформацію облікового запису користувача сторонніми службами, такими як Facebook, без викриття пароля користувача. OAuth виступає посередником від імені користувача, надаючи службі маркер доступу, який дозволяє спільну інформацію про певний обліковий запис.

1.5 Постановка задачі

Метою цією роботи є розгляд методу двофакторної автентифікації на основі веб-ресурсу. Метод покращить авторизацію користувача і створить більш безпечну систему автентифікації. Веб-ресурс дає змогу користувачу зробити замовлення онлайн, оплатити його та залишати свої дані конфіденційними.

Основні вимоги для веб-ресурсу:

1) Впровадження методу двофакторної автентифікації.
2) Авторизація користувача на веб-сайті.
3) Можливість знаходження потрібної інформації щодо замовлення
4) Організація роботи бази даних
5) Адаптивність
6) Оформлення замовлення в режимі онлайн
7) Стабільність роботи сайту
8) Зручність використання сайту
9) Структура, тобто зручне розміщення інформації на сайті
10) Сучасний та приємний дизайн

2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ

2.1 Вибір мови програмування

PHP - це мова сценаріїв загального призначення, яка використовується при створенні веб-сторінок і може бути вбудована в HTML-код. (PHP розшифровується як "Hypertext Preprocessor", хоча абревіатура точно не збігається.) [6]

PHP - це програма з відкритим кодом, що означає, що вона доступна безкоштовно в Інтернеті. PHP також може використовуватися на багатьох платформах, таких як Linux, багато варіацій системи Unix, Mac OS X та Microsoft Windows. PHP є повною мовою програмування і може використовуватися для таких функцій, як сценарії на стороні сервера (використання веб-сервера для виконання запиту користувача шляхом запуску скрипта безпосередньо на веб-сервері для створення динамічних HTML-сторінок) або написання програм для робочого столу.

Однак справжня сила PHP полягає в його здатності отримувати доступ до баз даних. PHP може отримати доступ до більш ніж 20 різних баз даних (включаючи MySQL), а також може обробляти інформацію в цих базах даних на основі вводу від користувача в мережі. PHP також може використовуватися для виведення зображень, PDF-файлів і навіть Flash-фільмів із вашого веб-сайту.

Laravel - це фреймворк веб-додатків, який намагається полегшити процес розробки, спрощуючи повторювані завдання, що використовуються в більшості сучасних веб-додатків, включаючи маршрутизацію, автентифікацію, кешування та сеанси.

Оскільки йому вдається виконувати всі основні завдання, починаючи від веб-сервісу та управління базами даних і закінчуючи генерацією HTML, Laravel називається повнофункціональним стеком. Це вертикально

інтегроване середовище веб-розробки має запропонувати вдосконалений та плавний робочий процес для розробника.

На відміну від інших вертикально інтегрованих середовищ, Laravel унікальний за своїм пріоритетом для конвенції над конфігурацією. Насправді, хоча багато фреймворків PHP вимагають важкої конфігурації XML перед початком реального проекту, Laravel потребує лише декількох рядків PHP коду для редагування, і він стає готовим до використання. Уникнення або використання мінімальної кількості конфігураційних файлів надає всім веб-програмам Laravel подібну структуру коду, яка є дуже характерною та ідентифікується. На перший погляд це може розглядатися як серйозне обмеження щодо того, як розробник може побажати організувати структуру власного веб-додатку. Однак ці обмеження значно полегшують створення веб-додатків.

Всі нові проекти Laravel виходять з коробки, оснащені повним деревом каталогів, а також багатьма файлами-заповнювачами, в результаті чого структура дозволяє швидко розпочати фактичний процес розробки. Проте ця структура повністю налаштовується.

В основному, весь визначений користувачем код працює в каталозі додатків (app) (рис. 2.1). Це серце програми Laravel, де обробляються всі HTTP-запити. У папці програм кожен тип рівня програми зберігається в окремій підкатегорії папки. Папка програми в основному зберігає логічні класи програм, включаючи модель, контролери, команди, проміжне програмне забезпечення тощо. За замовчуванням каталог програм автоматично завантажується Composer за допомогою стандарту автоматичного завантаження PSR-4. Варто знати, що більшість типів класів у папці додатків можна створити за допомогою команд artisan.

Крім того, інші типи ресурсів, такі як blade шаблони, файли LESS або SASS, файли CoffeeScript та мови, тепер зберігаються поза папкою ресурсів

замість папки програми. Тести та папка міграції також переміщуються із папки програми.

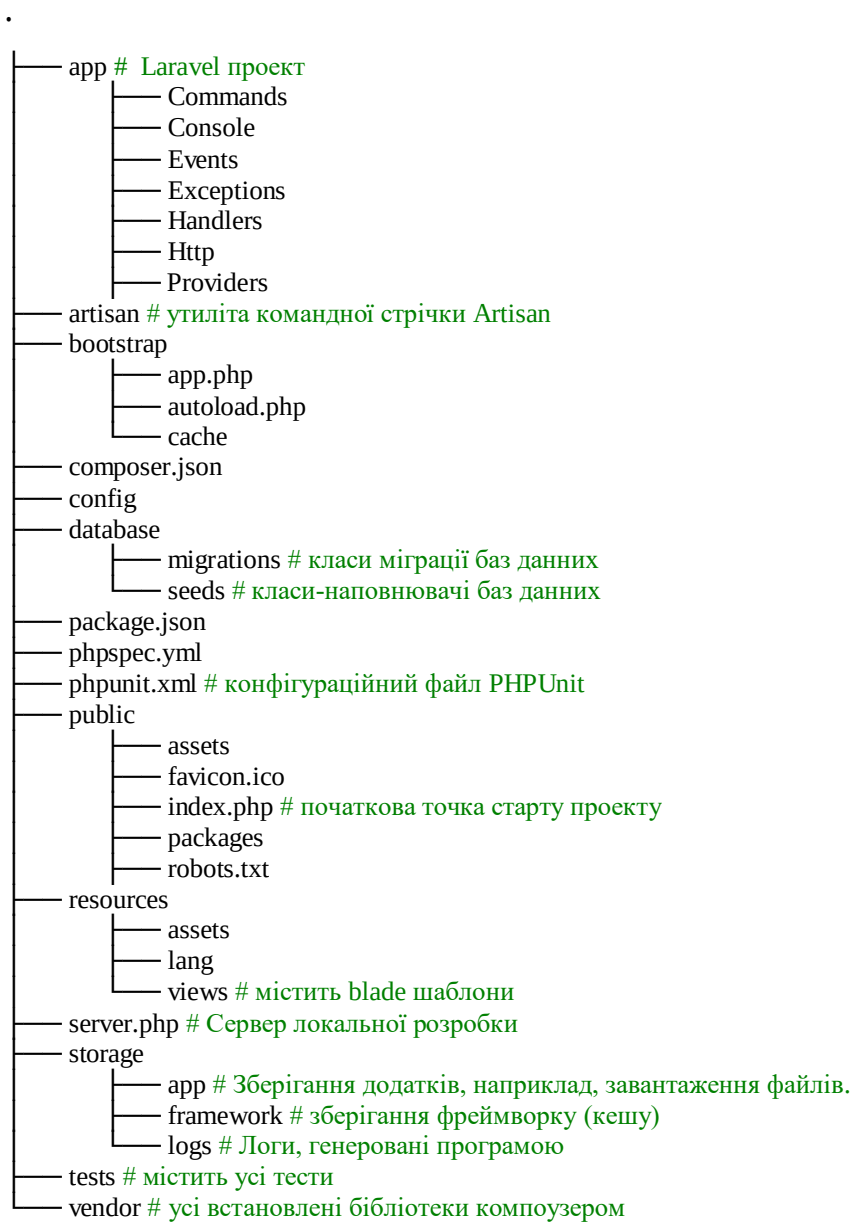

Рисунок 2.1 – структура Laravel проекту

2.2 Середовище розробки PHPStorm IDE

Для програмування на PHP не потрібен спеціальний редактор коду, оскільки файли PHP містять звичайний текст і можуть бути змінені за допомогою будь-якого текстового редактора. Однак для професійного

розвитку доцільно використовувати професійні інструменти, які полегшують навігацію та перевірку коду та доповнюють розробку за допомогою автозавершення. За допомогою IDE процес розробки стає більш ефективним та інтерактивним.

PHPStorm, будучи одним із найнадійніших інструментів для професійного програмування PHP, використовується при розробці веб-ресурсу на базі фреймворку Laravel. Він розуміє аналіз коду в режимі реального часу, має систему плагінів та забезпечує потужну навігаційну систему.

Перевагами використання даного IDE є:

1. Інтелектуальний редактор PHP
   - Завершення коду PHP
   - Інтегрований рефакторинг
   - Підтримка Smarty та PHPDoc
2. Проста установка
   - Крос-платформа
   - Індивідуальні налаштування проекту
3. Візуальний пробний запуску PHPUnit
4. Підтримка VCS (система контроля версіями)
   - SVN
   - Git
   - Меркурій
   - Локальна
5. FTP та віддалена синхронізація
6. Візуальна налагодження всередині IDE
7. Редактор HTML5 та CSS, включаючи дзен-кодування

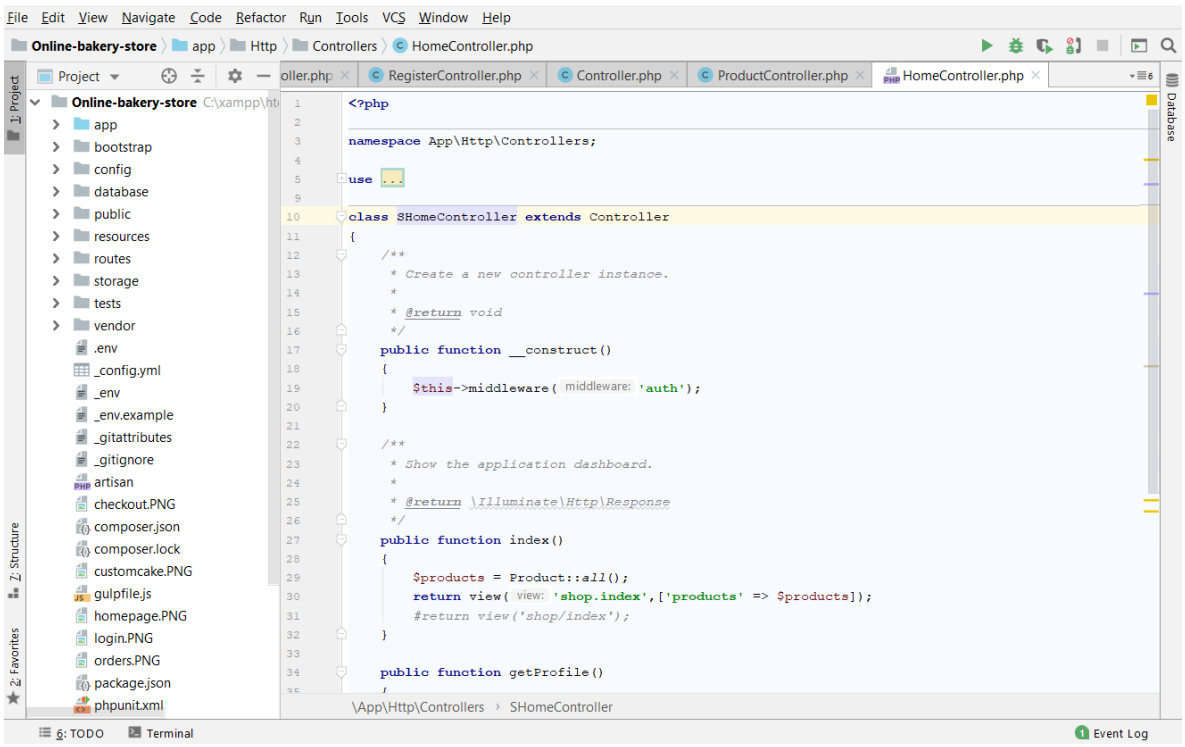

Рисунок 2.2 – Вигляд PHPStorm IDE

2.3 Обґрунтування методу двофакторної автентифікації

У сучасному онлайн-середовищі елементарний підхід «ім'я користувача та пароль» до безпеки є легкою здобиччю для кіберзлочинців. Багато входів можуть бути зламані за лічені хвилини, а особисті дані (наприклад, особисті та фінансові дані) знаходяться під загрозою.

Багатофакторна автентифікація, також відома як MFA або багатоетапна перевірка, додає ще один рівень безпеки, доповнюючи модель імені користувача та пароля кодом, до якого має доступ лише певний користувач (як правило, надсилається на мобільний телефон у вигляді SMS і його необхідно ввести протягом декількох хвилин). Цей метод автентифікації можна легко підсумувати як комбінацію "чогось, що ти маєш, і того, що ти знаєш".

2FA за допомогою мобільного пристрою стала галузевим стандартом, оскільки більшість людей постійно носять свої мобільні телефони. Це зручний потік, і динамічно генеровані паролі безпечні у використанні, і

користувачі можуть отримувати спеціальні токени через SMS або спеціальний додаток.

Рисунок 2.3 – Робота двофакторної автентифікації за допомогою SMS

Коли користувач реєструється або входить у програму, цифровий код надсилається на його мобільний пристрій або за допомогою SMS, або за допомогою програми автентифікації. Дві переваги використання програми автентифікації полягає в тому, що вона забезпечує постійній оборот набірів кодів, які користувачі можуть використовувати, коли це потрібно, і не вимагає Інтернет-з'єднання. Лише після того, як користувач введе правильний числовий код при вході у програму, він проходять автентифікацію.

Nexmo - платформа хмарних комунікацій, яка пропонує API-інтерфейси для ініціалізації телефонних номерів, відправки та прийому SMS (що ми і будемо використовувати), а так само для здійснення дзвінків.

Nexmo працює на прямій операторській мережі (Vonage), пропонуючи послуги на зростаючій кількості ринків. Nexmo використовує адаптивну маршрутизацію, переконуючись, що повідомлення доставляються найбільш зручним маршрутом, використовуючи найменший обсяг трафіку.

Однак деякі справді унікальні та круті функції Nexmo полягають у тому, що вони можуть заряджатись за секунду, а не за хвилину, коли мова йде про голосові дзвінки. Це означає, що користувач заощаджує досить багато грошей, платячи лише за той час, який він фактично використовує.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Розробка структури веб-ресурсу

Створений веб-сайт CakeWorld представлений у вигляді інтернет-магазину, побудований на основі фреймворку Laravel. Проект на основі данного фреймворку має досить обширну структуру, але одночасно й зрозумілу.

У данному магазині користувач має змогу вибрати продукт, при бажанні, змінити вагу, вигляд, додати свої коментарі. На сайті реалізована авторизація та автентифікація для перевірки користувача. Як наслідок зареєстрований користувач може зайти у свій профіль і побачити історію своїх замовлень. Також додана функція розрахунку картою безпосередньо на сайті.

Перший етапом при створенні веб-сайту є розробка і планування його структури. Вона зображена на рисунку 3.1.

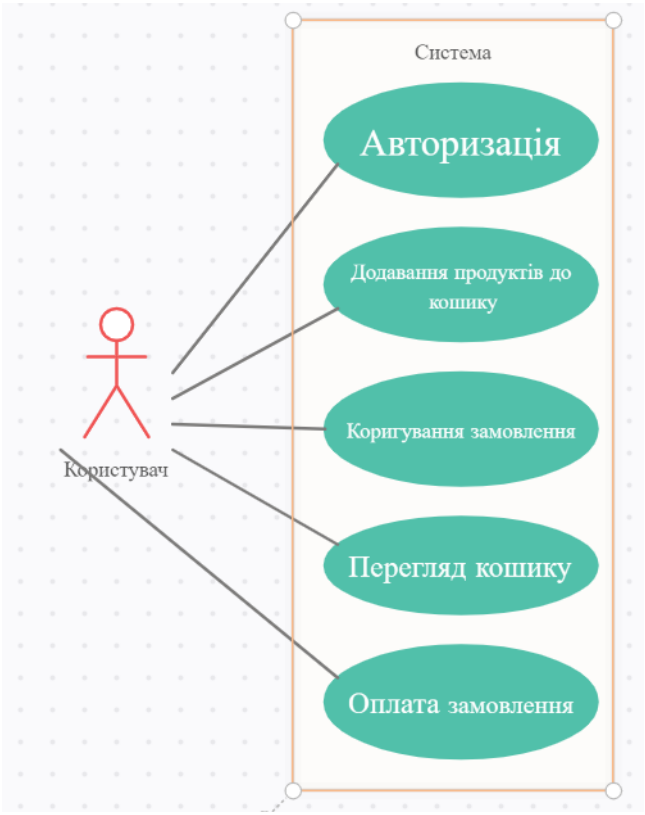

Рисунок 3.1 – UML модель інтернет-магазину CakeWorld

3.2 Опис основного функціоналу веб-ресурсу та методу автентифікації Nexmo

Для того, щоб мати можливість щось замовити на сайті, необхідно ввійти в свій обліковий запис.

```
        @section('content')
<div class="container">
  <div class="row">
    <div class="col-md-8 col-md-offset-2">
      <div class="panel panel-default">
        <div class="panel-heading">Вхід</div>
        <div class="panel-body">
          <form class="form-horizontal" role="form" method="POST" action="{{ url('/login') }}">
            {{ csrf_field() }}
            <div class="form-group{{ $errors->has('email') ? ' has-error' : '' }}">
              <label for="email" class="col-md-4 control-label">E-Mail</label>
```

В данному блоці користувач вводить свої дані, такі як адреса електронної пошти та пароль, передбачено можливість скидування паролю та запам’ятовування поточного користувача (за бажанням).

Також відбувається валідація введених данних і в залежності від ситуації з’являються відповідні повідомлення. Реалізація валідації та генерація повідомлень можна переглянути в Додатку.

```
                <div class="col-md-6">
                <input id="email" type="email" class="form-control" name="email" value="{{
old('email') }}" required autofocus>
                @if ($errors->has('email'))
                  <span class="help-block">
                    <strong>{{ $errors->first('email') }}</strong>
                  </span>
                @endif
              </div>
            </div>
            <div class="form-group{{ $errors->has('password') ? ' has-error' : '' }}">
              <label for="password" class="col-md-4 control-label">Пароль</label>
              <div class="col-md-6">
                <input id="password" type="password" class="form-control" name="password"
required>
                @if ($errors->has('password'))
```

```
                    <span class="help-block">
                        <strong>{{ $errors->first('password') }}</strong>
                    </span>
                @endif
            </div>
        </div>
        <div class="form-group">
            <div class="col-md-6 col-md-offset-4">
                <div class="checkbox">
                    <label>
                        <input type="checkbox" name="remember" {{ old('remember') ? 'checked' : ''}}>
Запам'ятати
                    </label>
                </div>
            </div>
        </div>
        <div class="form-group">
            <div class="col-md-8 col-md-offset-4">
                <button type="submit" class="btn btn-primary">
                    Ввійти
                </button>
                <a class="btn btn-link" href="{{ url('/password/reset') }}">
                    Забули пароль?
                </a>
            </div>
        </div>
    </form>
</div>
</div>
</div>
</div>
</div>
@endsection
```

Форма для введення даних користувача:

```
<form class="form-horizontal" role="form" method="POST" action="{{ url('/register') }}">
{{ csrf_field() }}
<div class="form-group{{ $errors->has('name') ? ' has-error' : '' }}">
    <label for="name" class="col-md-4 control-label">Ім'я</label>
```

```
        <div class="col-md-6">
            <input id="name" type="text" class="form-control" name="name" value="{{ old('name') }}"
required autofocus>
            @if ($errors->has('name'))
                <span class="help-block">
                    <strong>{{ $errors->first('name') }}</strong>
                </span>
            @endif
        </div>
    </div>
    <div class="form-group{{ $errors->has('email') ? ' has-error' : '' }}">
        <label for="email" class="col-md-4 control-label">E-Mail</label>
        <div class="col-md-6">
            <input id="email" type="email" class="form-control" name="email" value="{{ old('email') }}"
required>
            @if ($errors->has('email'))
                <span class="help-block">
                    <strong>{{ $errors->first('email') }}</strong>
                </span>
            @endif
        </div>
    </div>
```

При введенні данних в реєстраційну форму також відбувається їх валідація.

```
    <div class="form-group{{ $errors->has('password') ? ' has-error' : '' }}">
        <label for="password" class="col-md-4 control-label">Пароль</label>
        <div class="col-md-6">
            <input id="password" type="password" class="form-control" name="password" required>
            @if ($errors->has('password'))
                <span class="help-block">
                    <strong>{{ $errors->first('password') }}</strong>
                </span>
            @endif
        </div>
    </div>
    <div class="form-group">
        <label for="password-confirm" class="col-md-4 control-label">Підтвердження паролю</label>
```

```
      <div class="col-md-6">
        <input id="password-confirm" type="password" class="form-control"
name="password_confirmation" required>
      </div>
    </div>
    <div class="form-group">
      <div class="col-md-6 col-md-offset-4">
        <button type="submit" class="btn btn-primary">
          Зареєструватися
        </button>
      </div>
    </div>
</form>
```

Розрахунок загальної суми замовлення:

```
@section('content')
  @if(Session::has('cart'))
    <div class="row">
      <div class="col-sm-6 col-md-4 col-md-offset-4 col-sm-offset-3">
        <h1>Оплата та контактна інформація</h1>
        <h4>Сума замовлення : {{ $total }} грн.</h4>
        <div id="charge-error" class="alert alert-danger {{!Session::has('error') ? 'hidden' : ''}}">
          {{ Session::get('error') }}
        </div>
        <form action="{{ route('checkout') }}" method="post" id="checkout-form">
          <div class="row">
            <div class="col-xs-12">
              <div class="form-group">
                <label for="name">Повне ім'я</label>
                <input type="text" id="name" class="form-control" name="name" required>
              </div>
            </div>
            <div class="col-xs-12">
              <div class="form-group">
                <label for="address">Адреса</label>
                <input type="text" id="address" name="address" class="form-control" required>
              </div>
            </div>
```

Реалізована функція оплати онлайн:

```
<hr>
  <div class="col-xs-12">
    <div class="form-group">
      <label for="card-name">Ім'я власника картки</label>
      <input type="text" id="card-name" class="form-control" required>
    </div>
  </div>
  <div class="col-xs-12">
    <div class="form-group">
      <label for="card-number">Номер кредитної картки</label>
      <input type="text" id="card-number" class="form-control" required>
    </div>
  </div>
  <div class="col-xs-12">
    <div class="row">
      <div class="col-xs-6">
        <div class="form-group">
          <label for="card-expiry-month">Кінец дії картки(місяць)</label>
          <input type="text" id="card-expiry-month" class="form-control" required>
        </div>
      </div>
      <div class="col-xs-6">
        <div class="form-group">
          <label for="card-expiry-year">Кінец дії картки(рік)</label>
          <input type="text" id="card-expiry-year" class="form-control" required>
        </div>
      </div>
    </div>
  </div>
  <div class="col-xs-12">
    <div class="form-group">
      <label for="card-cvc">CVC</label>
      <input type="text" id="card-cvc" class="form-control" required>
    </div>
  </div>
</div>
{{ csrf_field() }}
<button type="submit" class="btn btn-success">Купити зараз</button>
</form>
</div>
</div>
```

```
  @endif
@endsection
```

Двофакторна автентифікація необхідна в данному випадку для верифікації користувача.

Перше, що потрібно зробити, це змінити таблицю USERS, щоб було готове поле для зберігання телефонного номера користувача. Для цього потрібно створити нову міграцію, щоб змінити таблицю USERS:

php artisan make:migration add_users_phone_number

Це створює файл у папці бази даних / міграцій з назвою <поточний час> _add_users_phone_number.php.

Новий вміст документу <поточний час> _add_users_phone_number.php:

```
use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class AddUsersPhoneNumber extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::table('users', function (Blueprint $table) {
            $table->string('phone_number');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
```

```
    */
    public function down()
    {
        Schema::table('users', function (Blueprint $table) {
            $table->dropColumn('phone_number');
        });
    }
}
```

Наступним кроком додається поле “Номер телефону” до форми реєстрації:

```
<div class="form-group{{ $errors->has('phone_number') ? ' has-error' : '' }}">
    <label for="name" class="col-md-4 control-label">Номер телефону</label>
    <div class="col-md-6">
        <input id="name" type="tel" class="form-control" name="phone_number" value="{{
old('phone_number') }}" required autofocus>
        @if ($errors->has('phone_number'))
            <span class="help-block">
    <strong>{{ $errors->first('phone_number') }}</strong>
</span>
        @endif
    </div>
</div>
```

Далі необхідно внести зміни до валідації та створити метод за допомогою якого усі номера будуть зберігатися в базі данних :

```
'phone_number' => 'required|size:12|unique:users',
'phone_number' => $data['phone_number'],
```

Якщо спробувати додати користувача зараз, це не буде працювати належним чином. Це пов'язано з функцією безпеки в Laravel, яка запобігає масовому присвоєнню властивостей класу. Не повідомивши клас Користувача, що номер телефону - дійсне поле, він відхилить запит на збереження. Щоб вирішити цю проблему, потрібно відредагувати app / User.php і додайте phone_number до масиву $ fillable.

Також задля верифікації необхідно внести зміни до контроллера авторизації:

```
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Contracts\Auth\Authenticatable;
        public function authenticated(Request $request, Authenticatable $user)
{
    Auth::logout();
    $request->session()->put('verify:user:id', $user->id);
    // @TODO: Send the Verify SMS here
    return redirect('verify');
}
```

Після ряду налаштувань та коротких змін необхідно внести в метод verify в контролері VerifyController:

```
        public function verify(Request $request) {
    $this->validate($request, [
        'code' => 'size:4',
    ]);
    try {
        Nexmo::verify()->check(
            $request->session()->get('verify:request_id'),
            $request->code
        );
        Auth::loginUsingId($request->session()->pull('verify:user:id'));
        return redirect('/home');
    } catch (Nexmo\Client\Exception\Request $e) {
        return redirect()->back()->withErrors([
            'code' => $e->getMessage()
        ]);
    }
}
```

На даний момент інтеграція повинна працювати з кінця до кінця. Якщо зберегти всі зміни та спробувати увійти в систему, то слід перенаправити на сторінку підтвердження та отримати текстове повідомлення з кодом підтвердження. Ввести код, і увійти, як очікувалося.

3.4 Інструкція користувача

Вперше, коли користувач переходить на сайт, перед ним з’являється головна сторінка з основним вибором солодощів. У випадку, якщо користувач ще не зареєстрований, то він має смогу це виправити. Адже без реєстрації він не матиме змогу робити замовленні. Форма реєстрації та авторизації зображена на рисунку 3.2 та 3.4.

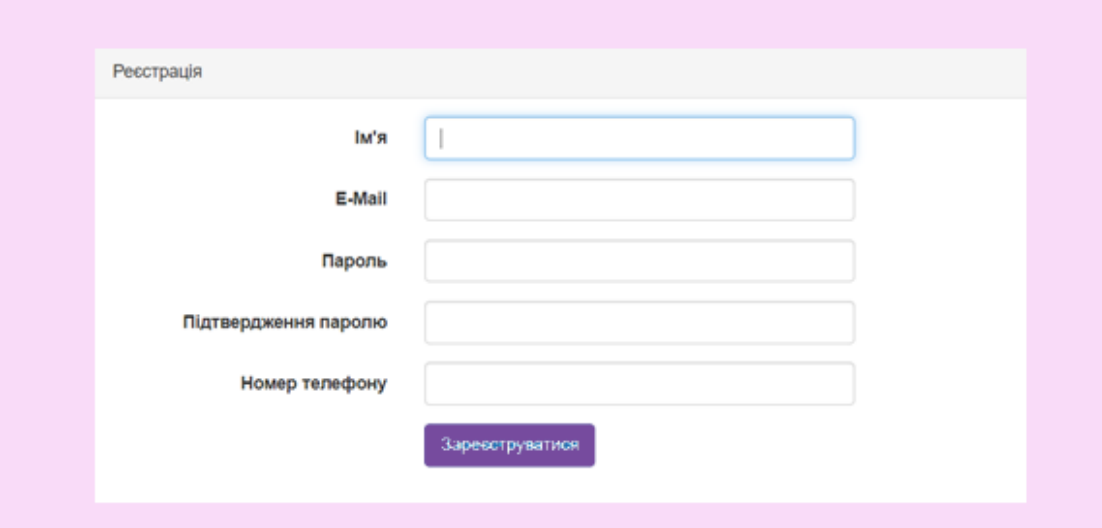

Рисунок 3.2 – форма реєстрації користувача

Для підтвердження особистості, необхідно пройти двофакторну автентифікацію шляхом введення свого номеру мобільного телефону, і далі отримати PIN з 4 цифр на у вигляді SMS.(Рисунок 3.3)

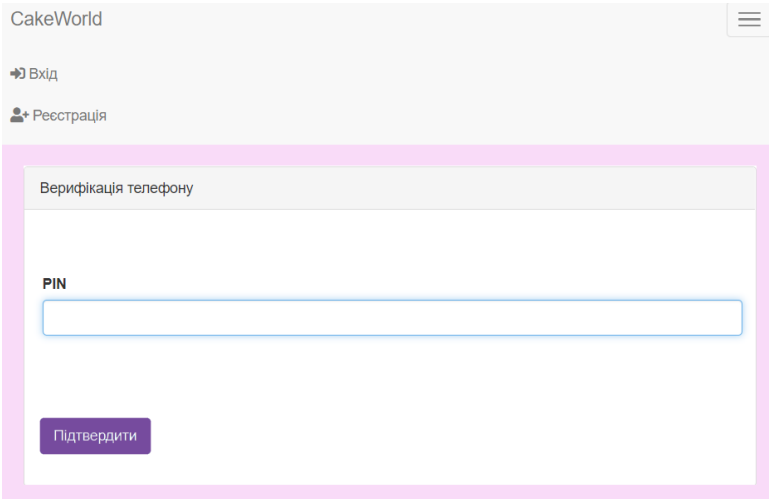

Рисунок 3.3 – Верифікація номеру телефону

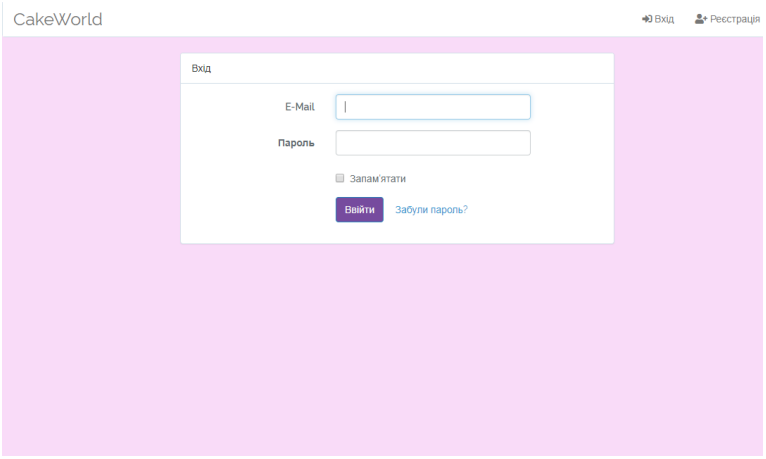

Рисунок 3.4 – Вхід до системи

Після того, як користувач успішно авторизувався, він має змогу ознайомлюватися з асортиментом та власне робити замовлення.

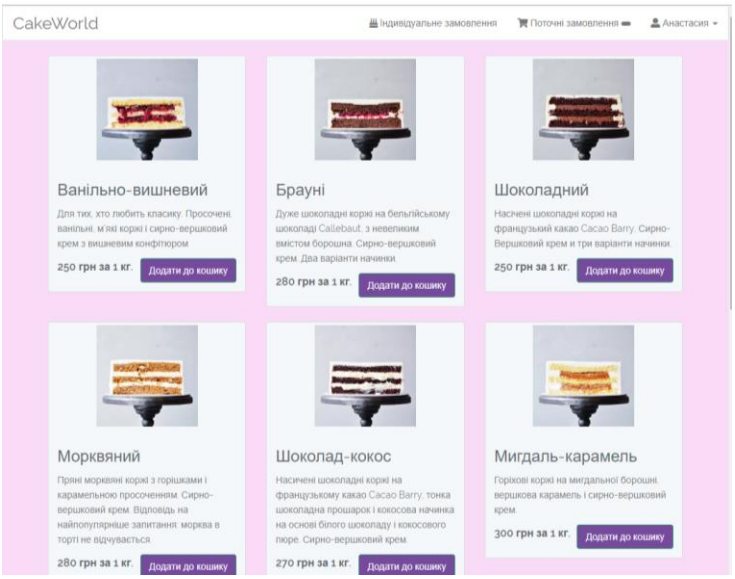

Рисунок 3.5 – Головна сторінка авторизованого користувача

Користувач має змогу кастомізувати свої замовлення, а тобто, змінювати вагу, смак, додавати свої коментарі, видаляти не потрібне. (Рисунок 3.6)

Також на сайті реалізована онлайн-оплата, де користувач вводить дані своєї картки. У даному розділі додана валідація, аби користувач міг

бачити свої помилки у випадку, якщо дані будуть введені не вірно. Форма оплати зображена на рисунку 3.7.

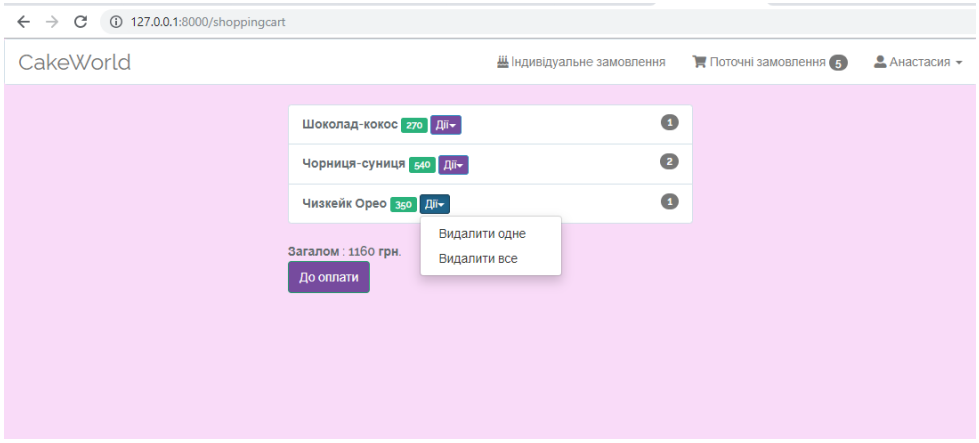

Рисунок 3.6 – Редагування кошику

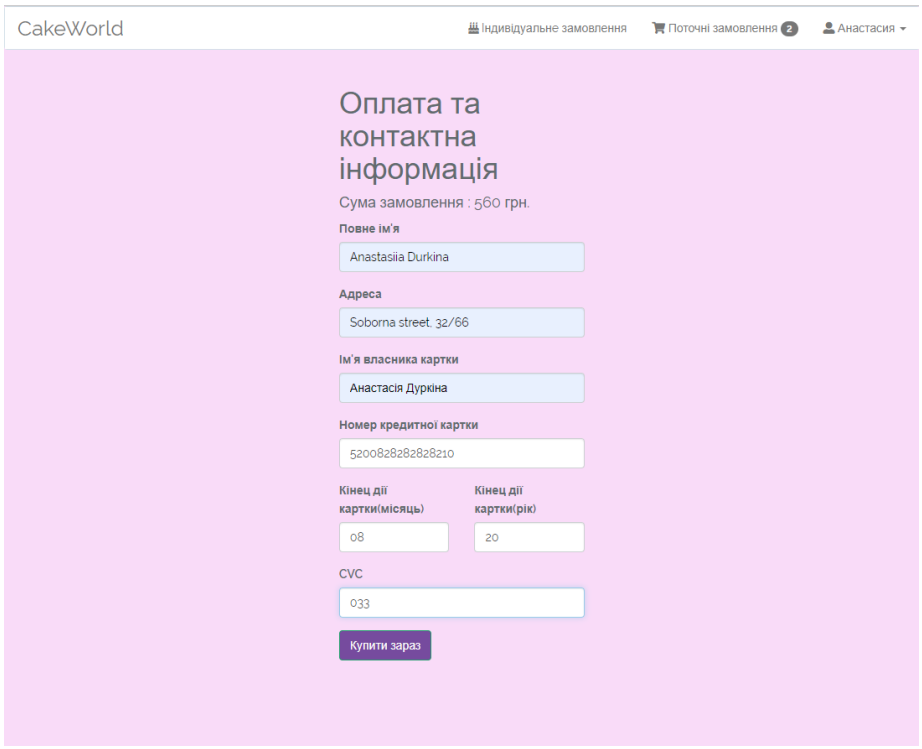

Рисунок 3.7 – Оплата замовлення

Також передбачено профіль користувача, де він може відстежувати історію своїх замовлень, а саме побачити суму, що саме, в якій кількості було замовлено (Рисунок 3.8).

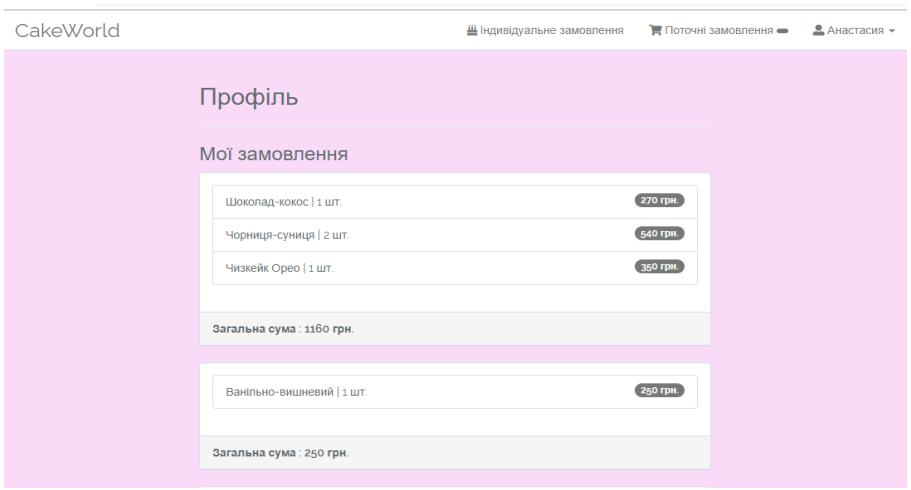

Рисунок 3.8 – Історія замовлень

ВИСНОВКИ

Питання інформаційної безпеки стає все більш актуальним кожного дня, адже вона дозволяє організаціям захищати свої мережі, дозволяючи лише автентифікованим користувачам (або процесам) доступ до захищених ресурсів, які можуть включати комп'ютерні системи, мережі, бази даних, веб-сайти та інші мережеві додатки чи послуги.

У результаті виконаної роботи було розроблено веб-сайт, який допоможе бажаючим замовити продукти та оплатити онлайн. Він забезпечений методом двофакторної автентифікації, який захистить від зловмисників, охочих викрасти дані банківської картки, які можуть бути збережені користувачем для своєї зручності. Двофакторна автентифікація представлена у вигляді комбінації вводу власних данних (в тому числі, мобільного телефону), та відправлення SMS з кодом для підтвердження.

Працюючи над проектом, були вивчені і застосовані на практиці можливості таких технологій, метод двофакторної автентифікації Nexmo, фреймворк Laravel, архітектура якого лягла в основу проекту, мова гіпертекстової розмітки HTML, каскадні таблиці стилів CSS, скриптова мова загального призначення PHP, а також JavaScript і його бібліотеки jQuery. Були закріплені знання з проектування структури бази даних, отримані під час навчання. Було здійснено розробку сайту з можливістю авторизації, реєстрації, оплати, корегування вмісту кошика, перегляд історії замовлень, тощо.

СПИСОК ЛІТЕРАТУРИ

1. Локхарт Д. Современный PHP. Новые возможности и передовой опыт / Джош Локхарт. - М.: ДМК Пресс, 2016. - 304 с.

2. Флэнаган Д. JavaScript. Подробное руководство, 6-е издание. – Пер. с англ. – СПб: Символ-Плюс, 2012.

3. Гаевский А. 100% самоучитель. Создание Web-страниц и Web-сайтов. HTML и JavaScript - М.: Триумф, 2014. - 464 с.

4. Никсон Р. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript - М.: Питер, 2013. - 496 с.

5. Дуркіна А І. Веб-ресурс для компанії "Cake World" на основі фреймворку Laravel [Текст]: робота на здобуття кваліфікаційного рівня бакалавр; спец.: 122 - комп`ютерні науки / А.І. Дуркіна; кер. О.Б. Проценко. - Суми: СумДУ, 2019. - 60 с.

6. Кузнецов М., Симдянов И. PHP. Практика создания Web-сайтов; БХВ-Петербург - Москва, 2012. - 347 с.

7. Советов Б., Цехановский В., Чертовской В. Базы данных: теория и практика. Учебник М.: «Юрайт», 2012. – 464 с.

8. Дунаев, В. HTML, скрипты и стили - М.: СПб: БХВ, 2006. - 832 с.

9. Кузнецов М., Симдянов И., Голышев С. PHP 5. Практика создания Web-сайтов; БХВ-Петербург – М., 2017. - 960 с.

10. Жадаев А. PHP для начинающих; "Издательство "Питер" - М., 2014. - 288 с.

11. Этапы создания веб-сайта, Fertdesign Studio [Електронний ресурс] – Режим доступу: http://im.fert.ru/pages/jetapy_sozdanija_veb-sajta/

12. Фреймворк Laravel. [Електронний ресурс] – Режим доступу: https://laravel.com/

13. Dockins K., Design Patterns in PHP and Laravel, 2017. – 238 p.

14. Sending and Receiving SMS with Laravel and Nexmo [Електронний ресурс] – Режим доступу: https://laravel-news.com/sending-receiving-sms-laravel-nexmo

15. Chapmen N., Chapman J. Authentication and Authorization on the Web, 2012. – 246 p.

ДОДАТОК

Web.php

```php
<?php

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| This file is where you may define all of the routes that are handled
| by your application. Just tell Laravel the URIs it should respond
| to using a Closure or controller method. Build something great!
|
*/

Route::get('/', [
      'uses' => 'ProductController@getIndex',
      'as' => 'product.index',
   ]);

Route::get('/add-to-cart/{id}', [
      'uses' => 'ProductController@getAddToCart',
      'as' => 'product.addtocart',
   ]);

Route::get('/reduce/{id}', [
      'uses' => 'ProductController@getReduceByOne',
      'as' => 'product.reducebyone',
   ]);

Route::get('/remove/{id}', [
      'uses' => 'ProductController@getRemoveItem',
      'as' => 'product.remove',
   ]);

Route::get('/shoppingcart', [
      'uses' => 'ProductController@getCart',
      'as' => 'product.shoppingcart',
   ]);

Route::get('/checkout', [
      'uses' => 'ProductController@getCheckout',
      'as' => 'checkout',
      'middleware' => 'auth'
   ]);

Route::get('customcake', function() {
    return view('customcake');
});

Route::post('/checkout', [
      'uses' => 'ProductController@postCheckout',
      'as' => 'checkout',
      'middleware' => 'auth'
   ]);

Auth::routes();

Route::get('/shop/index', 'HomeController@index');
```

```
Route::get('/user/profile', [
      'uses' => 'HomeController@getProfile',
      'as' => 'user.profile',
      'middleware' => 'auth',
   ]);
```

Welcome.blade.php

```
<!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">

        <title>Laravel</title>

        <!-- Fonts -->
        <link href="https://fonts.googleapis.com/css?family=Raleway:100,600"
rel="stylesheet" type="text/css">

        <!-- Styles -->
        <style>
            html, body {
                background-color: #fff;
                color: #636b6f;
                font-family: 'Raleway', sans-serif;
                font-weight: 100;
                height: 100vh;
                margin: 0;
            }

            .full-height {
                height: 100vh;
            }

            .flex-center {
                align-items: center;
                display: flex;
                justify-content: center;
            }

            .position-ref {
                position: relative;
            }

            .top-right {
                position: absolute;
                right: 10px;
                top: 18px;
            }

            .content {
                text-align: center;
            }

            .title {
                font-size: 84px;
            }

            .links > a {
```

```
                color: #636b6f;
                padding: 0 25px;
                font-size: 12px;
                font-weight: 600;
                letter-spacing: .1rem;
                text-decoration: none;
                text-transform: uppercase;
            }

            .m-b-md {
                margin-bottom: 30px;
            }
        </style>
    </head>
    <body>
        <div class="flex-center position-ref full-height">
            @if (Route::has('login'))
                <div class="top-right links">
                    <a href="{{ url('/login') }}">Ввійти</a>
                    <a href="{{ url('/register') }}">Зареєструватися</a>
                </div>
            @endif

            <div class="content">
                <div class="title m-b-md">
                    Laravel
                </div>

                <div class="links">
                    <a href="https://laravel.com/docs">Documentation</a>
                    <a href="https://laracasts.com">Laracasts</a>
                    <a href="https://laravel-news.com">News</a>
                    <a href="https://forge.laravel.com">Forge</a>
                    <a href="https://github.com/laravel/laravel">GitHub</a>
                </div>
            </div>
        </div>
    </body>
</html>
```

Home.blade.php

```
@extends('layouts.master')

@section('content')
<div class="container">
    <div class="row">
        <div class="col-md-8 col-md-offset-2">
            <div class="panel panel-default">
                <div class="panel-heading"></div>

                <div class="panel-body">
                    Ви ввішли!
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

Customcake.blade.php

```
@extends('layouts.master')


<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>Form Conditions - Example 3</title>
    <link href="css/formcalculation.css" rel="stylesheet">
    <link href="css/conditions3.css" rel="stylesheet">
</head>
<body>
<div id="wb_Form1" style="position:absolute; margin: auto; top: 0; right: 0;
bottom: 0; left: 0;width:500px; height:630px;">
    <form name="Form1" method="post" action="mailto:yourname@yourdomain.com"
enctype="text/plain" id="Form1">
        <h3 style="position:absolute;left:20px;top:-60px;line-height:26px;z-
index:0;">Створіть свій індивідуальний торт</h3>
        <label for="" id="Label1"
style="position:absolute;left:34px;top:17px;width:72px;height:26px;line-
height:26px;z-index:0;">Смак:</label>
        <div id="wb_RadioButton1"
style="position:absolute;left:184px;top:23px;width:20px;height:20px;z-index:1;">
            <input type="radio" id="RadioButton1" name="product" value="250"
checked="" style="position:absolute;left:30px;top:30px;z-index:1;"><label
for="RadioButton1"></label></div>
        <div id="wb_RadioButton2"
style="position:absolute;left:184px;top:55px;width:20px;height:20px;z-index:3;">
            <input type="radio" id="RadioButton2" name="product" value="250"
style="position:absolute;left:30px;top:30px;z-index:3;"><label
for="RadioButton2"></label></div>
        <div id="wb_RadioButton3"
style="position:absolute;left:184px;top:86px;width:20px;height:20px;z-index:4;">
            <input type="radio" id="RadioButton3" name="product" value="300"
style="position:absolute;left:30px;top:30px;z-index:4;"><label
for="RadioButton3"></label></div>
        <div id="wb_RadioButton4"
style="position:absolute;left:184px;top:130px;width:20px;height:20px;z-index:4;">
            <input type="radio" id="RadioButton4" name="product" value="270"
style="position:absolute;left:30px;top:30px;z-index:4;"><label
for="RadioButton4"></label></div>
        <div id="wb_RadioButton5"
style="position:absolute;left:184px;top:162px;width:20px;height:20px;z-index:4;">
            <input type="radio" id="RadioButton5" name="product" value="250"
style="position:absolute;left:30px;top:30px;z-index:4;"><label
for="RadioButton5"></label></div>
        <div id="wb_RadioButton6"
style="position:absolute;left:184px;top:194px;width:20px;height:20px;z-index:4;">
            <input type="radio" id="RadioButton6" name="product" value="250"
style="position:absolute;left:30px;top:30px;z-index:4;"><label
for="RadioButton6"></label></div>
        <div id="wb_RadioButton7"
style="position:absolute;left:184px;top:226px;width:20px;height:20px;z-index:4;">
            <input type="radio" id="RadioButton7" name="product" value="280"
style="position:absolute;left:30px;top:30px;z-index:4;"><label
for="RadioButton7"></label></div>

        <div id="wb_Text1"
style="position:absolute;left:213px;top:25px;width:210px;height:16px;z-index:2;">
            <span style="color:#000000;font-family:Arial;font-
size:13px;"><strong>Шварцвальд - 250 грн/кг</strong></span></div>
```

```
        <div id="wb_Text2"
style="position:absolute;left:213px;top:57px;width:210px;height:16px;z-index:5;">
            <span style="color:#000000;font-family:Arial;font-
size:13px;"><strong>Ваніль з малиною - 250 грн/кг</strong></span></div>
        <div id="wb_Text3"
style="position:absolute;left:213px;top:88px;width:210px;height:16px;z-index:6;">
            <span style="color:#000000;font-family:Arial;font-
size:13px;"><strong>CarrotCake з миндалем та грушею - 300
грн/кг</strong></span></div>
        <div id="wb_Text4"
style="position:absolute;left:213px;top:132px;width:210px;height:16px;z-index:6;">
            <span style="color:#000000;font-family:Arial;font-
size:13px;"><strong>Червоний оксамит - 270 грн/кг</strong></span></div>
        <div id="wb_Text5"
style="position:absolute;left:213px;top:164px;width:210px;height:16px;z-index:6;">
            <span style="color:#000000;font-family:Arial;font-
size:13px;"><strong>Шоколад з персиком - 250 грн/кг</strong></span></div>
        <div id="wb_Text6"
style="position:absolute;left:213px;top:196px;width:210px;height:16px;z-index:6;">
            <span style="color:#000000;font-family:Arial;font-
size:13px;"><strong>Тірамісу - 250 грн/кг </strong></span></div>
        <div id="wb_Text7"
style="position:absolute;left:213px;top:228px;width:210px;height:16px;z-index:6;">
            <span style="color:#000000;font-family:Arial;font-
size:13px;"><strong>Екзотік - 280 грн/кг </strong></span></div>

        <label for="quantity" id="Label2"
style="position:absolute;left:34px;top:260px;width:150px;height:26px;line-
height:26px;z-index:7;">Вага:(мінімум 1 кг)</label>
        <input type="text" id="quantity"
style="position:absolute;left:184px;top:270px;width:86px;height:16px;line-
height:16px;z-index:8;" name="quantity" value="1" spellcheck="false">
        <label for="message" id="Label5"
style="position:absolute;left:34px;top:290px;width:100px;height:26px;line-
height:26px;z-index:7;">Надпис:</label>
        <textarea type="text" id="message"
style="position:absolute;left:184px;top:300px;width:200px;z-index:8;"
name="message" rows="3" cols="30" spellcheck="false"></textarea>
        <label for="notes" id="Label4"
style="position:absolute;left:34px;top:380px;width:150px;height:26px;line-
height:26px;z-index:7;">Ваш коментар:</label>
        <textarea style="position:absolute;left:184px;top:390px;width:200px;z-
index:8;" name="notes" rows="5" cols="30" id="notes"></textarea>
        <label for="total" id="Label13"
style="position:absolute;left:34px;top:520px;width:150px;height:18px;line-
height:18px;z-index:11;">Загалом в гривнях:</label>
        <input type="text" id="total"
style="position:absolute;left:184px;top:530px;width:86px;height:16px;line-
height:16px;z-index:9;" name="total" value="" spellcheck="false">
        <input type="submit" id="Button1" name="" value="Підтвердити"
style="position:absolute;left:34px;top:570px;width:96px;height:25px;z-
index:10;background-color:#764b9e;">

    </form>
</div>
<script src="js/jquery-1.12.4.min.js"></script>
<script src="js/conditions3.js"></script>
</body>
</html>
@section('scripts')
    <script type="text/javascript" src="https://js.stripe.com/v2/"></script>
```

```
    <script type="text/javascript" src="{{ URL::to('js/checkout.js') }}"></script>
@endsection
```

Shoppingcart.blade.php

```
@extends('layouts.master')

@section('title')
    Кошик
@endsection

@section('content')
    @if(Session::has('cart'))
        <div class="row">
            <div class="col-sm-6 col-md-6 col-md-offset-3 col-sm-offset-3">
                <ul class="list-group">
                    @foreach($products as $product)
                        <li class="list-group-item">
                            <span class="badge">{{ $product['qty'] }}</span>
                            <strong>{{ $product['item']['title'] }}</strong>
                            <span class="label label-success">{{ $product['price']
}}</span>
                            <div class="btn-group">
                                <button type="button" class="btn btn-primary btn-xs
dropdown-toggle" data-toggle="dropdown">Дії<span class="caret"></span></button>
                                <ul class="dropdown-menu">
                                    <li><a href="{{ route('product.reducebyone', ['id' =>
$product['item']['id']]) }}">Видалити одне</a></li>
                                    <li><a href="{{ route('product.remove', ['id' =>
$product['item']['id']]) }}">Видалити все</a></li>
                                </ul>
                            </div>
                        </li>
                    @endforeach
                </ul>
            </div>
        </div>
        <div class="row">
            <div class="col-sm-6 col-md-6 col-md-offset-3 col-sm-offset-3">
                <strong>Загалом : {{ $totalprice }} грн. </strong>
            </div>
        </div>
        <div class="row">
            <div class="col-sm-6 col-md-6 col-md-offset-3 col-sm-offset-3">
                <a href="{{ route('checkout') }}" type="button" class="btn btn-
success">До оплати</a>
            </div>
        </div>
    @else
        <div class="row">
            <div class="col-sm-6 col-md-6 col-md-offset-3 col-sm-offset-3">
                <h2>Ваш кошик порожній<br>Будь ласка, оберіть бажане.</h2>
            </div>
        </div>
    @endif
@endsection
```

Index.blade.php

```
@extends('layouts.master')
```

```
@section('title')
    Кошик
@endsection

@section('content')
    @if(Session::has('success'))
    <div class="row">
        <div class="col-sm-6 col-md-4 col-md-offset-4 col-sm-offset-3">
            <div id="charge-message" class="alert alert-success">
                Успішно придбано!!!
            </div>
        </div>
    </div>
    @endif
    <!--<h1>It Works!!</h1>-->
    @foreach($products->chunk(3) as $productChunk)
        <div class="row">
            @foreach($productChunk as $product)
                <div class="col-sm-6 col-md-4">
                    <div class="thumbnail">
                        <img src="{{ $product->imagepath }}" alt="..." class="img-
responsive">
                        <div class="caption">
                            <h3>{{ $product->title }}</h3>
                            <p class="description">
                                {{ $product->description }}
                            </p>
                            <div class="clearfix">
                                <div class="pull-left price">
                                    {{ $product->price }} грн за 1 кг.
                                </div>
                                <a href="{{ route('product.addtocart', ['id' =>
$product->id]) }}" class="btn btn-success pull-right" role="button">Додати до
кошику</a>
                            </div>
                        </div>
                    </div>
                </div>
            @endforeach
        </div>
    @endforeach
@endsection
```

Welcome.blade.php

```
@extends('layouts.master')

@section('title')
    Кошик
@endsection

@section('content')
    @if(Session::has('success'))
    <div class="row">
        <div class="col-sm-6 col-md-4 col-md-offset-4 col-sm-offset-3">
            <div id="charge-message" class="alert alert-success">
                Успішно придбано!!!
            </div>
        </div>
    </div>
    @endif
```

```
    <!--<h1>It Works!!</h1>-->
    @foreach($products->chunk(3) as $productChunk)
        <div class="row">
            @foreach($productChunk as $product)
                <div class="col-sm-6 col-md-4">
                    <div class="thumbnail">
                        <img src="{{ $product->imagepath }}" alt="..." class="img-
responsive">
                        <div class="caption">
                            <h3>{{ $product->title }}</h3>
                            <p class="description">
                                {{ $product->description }}
                            </p>
                            <div class="clearfix">
                                <div class="pull-left price">
                                    {{ $product->price }} грн за 1 кг.
                                </div>
                                <a href="{{ route('product.addtocart', ['id' =>
$product->id]) }}" class="btn btn-success pull-right" role="button">Додати до
кошику</a>
                            </div>
                        </div>
                    </div>
                </div>
            @endforeach
        </div>
    @endforeach
@endsection
```

app.blade.php

```
@extends('layouts.master')

@section('title')
    Кошик
@endsection

@section('content')
    @if(Session::has('success'))
    <div class="row">
        <div class="col-sm-6 col-md-4 col-md-offset-4 col-sm-offset-3">
            <div id="charge-message" class="alert alert-success">
                Успішно придбано!!!
            </div>
        </div>
    </div>
    @endif
    <!--<h1>It Works!!</h1>-->
    @foreach($products->chunk(3) as $productChunk)
        <div class="row">
            @foreach($productChunk as $product)
                <div class="col-sm-6 col-md-4">
                    <div class="thumbnail">
                        <img src="{{ $product->imagepath }}" alt="..." class="img-
responsive">
                        <div class="caption">
                            <h3>{{ $product->title }}</h3>
                            <p class="description">
                                {{ $product->description }}
                            </p>
                            <div class="clearfix">
```

```
                            <div class="pull-left price">
                                {{ $product->price }} грн за 1 кг.
                            </div>
                            <a href="{{ route('product.addtocart', ['id' =>
$product->id]) }}" class="btn btn-success pull-right" role="button">Додати до
кошику</a>
                        </div>
                    </div>
                </div>
            </div>
        @endforeach
    </div>
  @endforeach
@endsection
```

profile.blade.php

```
@extends('layouts.master')

@section('title')
Мій Профіль
@endsection

@section('content')
    <div class="row">
        <div class="col-md-8 col-md-offset-2">
            <h2>Профіль</h2>
            <hr>
            <h3>Мої замовлення</h3>

            @foreach($orders as $order)
                <div class="panel panel-default">
                    <div class="panel-body">
                        <ul class="list-group">
                            @foreach($order->cart->items as $item)
                                <li class="list-group-item">
                                    <span class="badge">

                                        {{ $item['price'] }} грн. </span>
                                        {{ $item['item']['title'] }} |  {{
$item['qty'] }} шт.
                                </li>
                            @endforeach
                        </ul>
                    </div>
                    <div class="panel-footer">
                        <strong>Загальна сума : {{ $order->cart->totalprice }}
грн.</strong>
                    </div>
                </div>
            @endforeach
        </div>
    </div>
@endsection
```

reset.blade.php

```
@extends('layouts.master')

@section('content')
<div class="container">
```

```
    <div class="row">
        <div class="col-md-8 col-md-offset-2">
            <div class="panel panel-default">
                <div class="panel-heading">Скинути пароль</div>

                <div class="panel-body">
                    @if (session('status'))
                        <div class="alert alert-success">
                            {{ session('status') }}
                        </div>
                    @endif

                    <form class="form-horizontal" role="form" method="POST"
action="{{ url('/password/reset') }}">
                        {{ csrf_field() }}

                        <input type="hidden" name="token" value="{{ $token }}">

                        <div class="form-group{{ $errors->has('email') ? ' has-
error' : '' }}">
                            <label for="email" class="col-md-4 control-label">E-
Mail</label>

                            <div class="col-md-6">
                                <input id="email" type="email" class="form-
control" name="email" value="{{ $email or old('email') }}" required autofocus>

                                @if ($errors->has('email'))
                                    <span class="help-block">
                                        <strong>{{ $errors->first('email')
}}</strong>
                                    </span>
                                @endif
                            </div>
                        </div>

                        <div class="form-group{{ $errors->has('password') ? ' has-
error' : '' }}">
                            <label for="password" class="col-md-4 control-
label">Пароль</label>

                            <div class="col-md-6">
                                <input id="password" type="password" class="form-
control" name="password" required>

                                @if ($errors->has('password'))
                                    <span class="help-block">
                                        <strong>{{ $errors->first('password')
}}</strong>
                                    </span>
                                @endif
                            </div>
                        </div>

                        <div class="form-group{{ $errors-
>has('password_confirmation') ? ' has-error' : '' }}">
                            <label for="password-confirm" class="col-md-4 control-
label">Підтвердження паролю</label>
                            <div class="col-md-6">
                                <input id="password-confirm" type="password"
class="form-control" name="password_confirmation" required>
```

```
                                @if ($errors->has('password_confirmation'))
                                    <span class="help-block">
                                        <strong>{{ $errors->first('password_confirmation') }}</strong>
                                    </span>
                                @endif
                            </div>
                        </div>

                        <div class="form-group">
                            <div class="col-md-6 col-md-offset-4">
                                <button type="submit" class="btn btn-primary">
                                    Скинути пароль
                                </button>
                            </div>
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

email.blade.php

```
@extends('layouts.master')

<!-- Main Content -->
@section('content')
<div class="container">
    <div class="row">
        <div class="col-md-8 col-md-offset-2">
            <div class="panel panel-default">
                <div class="panel-heading">Скинути пароль</div>
                <div class="panel-body">
                    @if (session('status'))
                        <div class="alert alert-success">
                            {{ session('status') }}
                        </div>
                    @endif

                    <form class="form-horizontal" role="form" method="POST" action="{{ url('/password/email') }}">
                        {{ csrf_field() }}

                        <div class="form-group{{ $errors->has('email') ? ' has-error' : '' }}">
                            <label for="email" class="col-md-4 control-label">E-Mail</label>

                            <div class="col-md-6">
                                <input id="email" type="email" class="form-control" name="email" value="{{ old('email') }}" required>

                                @if ($errors->has('email'))
                                    <span class="help-block">
                                        <strong>{{ $errors->first('email') }}</strong>
                                    </span>
                                @endif
                            </div>
```

```
                        </div>

                        <div class="form-group">
                            <div class="col-md-6 col-md-offset-4">
                                <button type="submit" class="btn btn-primary">
                                    Надіслати посилання на скидання пароля
                                </button>
                            </div>
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

validation.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Validation Language Lines
    |--------------------------------------------------------------------------
    |
    | The following language lines contain the default error messages used by
    | the validator class. Some of these rules have multiple versions such
    | as the size rules. Feel free to tweak each of these messages here.
    |
    */

    'accepted'             => 'The :attribute must be accepted.',
    'active_url'           => 'The :attribute is not a valid URL.',
    'after'                => 'The :attribute must be a date after :date.',
    'alpha'                => 'The :attribute may only contain letters.',
    'alpha_dash'           => 'The :attribute may only contain letters, numbers,
and dashes.',
    'alpha_num'            => 'The :attribute may only contain letters and
numbers.',
    'array'                => 'The :attribute must be an array.',
    'before'               => 'The :attribute must be a date before :date.',
    'between'              => [
        'numeric' => 'The :attribute must be between :min and :max.',
        'file'    => 'The :attribute must be between :min and :max kilobytes.',
        'string'  => 'The :attribute must be between :min and :max characters.',
        'array'   => 'The :attribute must have between :min and :max items.',
    ],
    'boolean'              => 'The :attribute field must be true or false.',
    'confirmed'            => 'The :attribute confirmation does not match.',
    'date'                 => 'The :attribute is not a valid date.',
    'date_format'          => 'The :attribute does not match the format :format.',
    'different'            => 'The :attribute and :other must be different.',
    'digits'               => 'The :attribute must be :digits digits.',
    'digits_between'       => 'The :attribute must be between :min and :max
digits.',
    'dimensions'           => 'The :attribute has invalid image dimensions.',
    'distinct'             => 'The :attribute field has a duplicate value.',
    'email'                => 'The :attribute must be a valid email address.',
    'exists'               => 'The selected :attribute is invalid.',
```

```
    'file'                 => 'The :attribute must be a file.',
    'filled'               => 'The :attribute field is required.',
    'image'                => 'The :attribute must be an image.',
    'in'                   => 'The selected :attribute is invalid.',
    'in_array'             => 'The :attribute field does not exist in :other.',
    'integer'              => 'The :attribute must be an integer.',
    'ip'                   => 'The :attribute must be a valid IP address.',
    'json'                 => 'The :attribute must be a valid JSON string.',
    'max'                  => [
        'numeric' => 'The :attribute may not be greater than :max.',
        'file'    => 'The :attribute may not be greater than :max kilobytes.',
        'string'  => 'The :attribute may not be greater than :max characters.',
        'array'   => 'The :attribute may not have more than :max items.',
    ],
    'mimes'                => 'The :attribute must be a file of type: :values.',
    'mimetypes'            => 'The :attribute must be a file of type: :values.',
    'min'                  => [
        'numeric' => 'The :attribute must be at least :min.',
        'file'    => 'The :attribute must be at least :min kilobytes.',
        'string'  => 'The :attribute must be at least :min characters.',
        'array'   => 'The :attribute must have at least :min items.',
    ],
    'not_in'               => 'The selected :attribute is invalid.',
    'numeric'              => 'The :attribute must be a number.',
    'present'              => 'The :attribute field must be present.',
    'regex'                => 'The :attribute format is invalid.',
    'required'             => 'The :attribute field is required.',
    'required_if'          => 'The :attribute field is required when :other is
:value.',
    'required_unless'      => 'The :attribute field is required unless :other is
in :values.',
    'required_with'        => 'The :attribute field is required when :values is
present.',
    'required_with_all'    => 'The :attribute field is required when :values is
present.',
    'required_without'     => 'The :attribute field is required when :values is
not present.',
    'required_without_all' => 'The :attribute field is required when none of
:values are present.',
    'same'                 => 'The :attribute and :other must match.',
    'size'                 => [
        'numeric' => 'The :attribute must be :size.',
        'file'    => 'The :attribute must be :size kilobytes.',
        'string'  => 'The :attribute must be :size characters.',
        'array'   => 'The :attribute must contain :size items.',
    ],
    'string'               => 'The :attribute must be a string.',
    'timezone'             => 'The :attribute must be a valid zone.',
    'unique'               => 'The :attribute has already been taken.',
    'uploaded'             => 'The :attribute failed to upload.',
    'url'                  => 'The :attribute format is invalid.',

    /*
    |--------------------------------------------------------------------------
    | Custom Validation Language Lines
    |--------------------------------------------------------------------------
    |
    | Here you may specify custom validation messages for attributes using the
    | convention "attribute.rule" to name the lines. This makes it quick to
    | specify a specific custom language line for a given attribute rule.
    |
    */
```

```
    'custom' => [
        'attribute-name' => [
            'rule-name' => 'custom-message',
        ],
    ],

    /*
    |--------------------------------------------------------------------------
    | Custom Validation Attributes
    |--------------------------------------------------------------------------
    |
    | The following language lines are used to swap attribute place-holders
    | with something more reader friendly such as E-Mail Address instead
    | of "email". This simply helps us make messages a little cleaner.
    |
    */

    'attributes' => [],

];
```

passwords.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Password Reset Language Lines
    |--------------------------------------------------------------------------
    |
    | The following language lines are the default lines which match reasons
    | that are given by the password broker for a password update attempt
    | has failed, such as for an invalid token or invalid new password.
    |
    */

    'password' => 'Паролі повинні містити принаймні шість символів та відповідати
підтвердженню.',
    'reset' => 'Ваш пароль був скинутий!',
    'sent' => 'Ми надіслали електронною поштою ваше посилання для скидання
пароля!',
    'token' => 'This password reset token is invalid.Цей пароль недійсний.',
    'user' => "Ми не можемо знайти користувача з цією адресою електронної пошти.",

];
```

auth.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Authentication Language Lines
    |--------------------------------------------------------------------------
    |
    | The following language lines are used during authentication for various
```

```
    | messages that we need to display to the user. You are free to modify
    | these language lines according to your application's requirements.
    |
    */

    'failed' => 'Ви ввели некорректні дані',
    'throttle' => 'Перевищено ліміт спроб входу. Спробуйте пізніше.',

];
```

Checkout.js

```
Stripe.setPublishableKey('pk_test_CB7soYLhiJJSIqZhiFojwk1g');

var $form = $('#checkout-form');
$form.submit(function(event){
   $('charge-error').addClass('hidden');
   $form.find('button').prop('disabled',true);
   Stripe.card.createToken({
     number: $('#card-number').val(),
     cvc: $('#card-cvc').val(),
     exp_month: $('#card-expiry-month').val(),
     exp_year: $('#card-expiry-year').val(),
     name: $('#card-name').val()
   }, stripeResponseHandler);
   return false;
});


function stripeResponseHandler(status, response)
{
   if(response.error)
   {
      $('#charge-error').removeClass('hidden');
      $('#charge-error').text(response.error.message);
      $form.find('button').prop('disabled',false);
   }
   else
   {
      var token = response.id;
      $form.append($('<input type="hidden" name="stripeToken" />').val(token));
      $form.get(0).submit();
   }
}
```

formcalculation.js

```
$(document).ready(function()
{
    $("#quantity, #RadioButton1, #RadioButton2, #RadioButton3, #RadioButton4,
#RadioButton5, #RadioButton6, #RadioButton7").change(function()
    {

$('#total').val($('input[name="product"]:checked').val()*$('#quantity').val())
    });
    $("#quantity").trigger('change');
});
```

ProductTableSeeder.php

```php
<?php

use Illuminate\Database\Seeder;

class ProductTableSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {

        $product = new \App\Product([
            'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_d8434b1b1b924c7c837168c0a264301d~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_310,h_310,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_d8434b1
b1b924c7c837168c0a264301d~mv2_d_1632_1631_s_2.webp',
            'title' => 'Ванільно-вишневий',
            'description' =>'Для тих, хто любить класику. Просочені, ванільні,
м\'які коржі і сирно-вершковий крем з вишневим конфітюром',
            'price' => 250
        ]);
        $product->save();
        $product = new \App\Product([
            'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_d37549ef839141b2a470082ac29bfa9c~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_d37549e
f839141b2a470082ac29bfa9c~mv2_d_1632_1631_s_2.webp',
            'title' => 'Брауні',
            'description' =>'Дуже шоколадні коржі на бельгійському шоколаді
Callebaut, з невеликим вмістом борошна. Сирно-вершковий крем. Два варіанти
начинки.',
            'price' => 280
        ]);
        $product->save();
        $product = new \App\Product([
            'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_93997f7c69304906938e5c5cd6b15c47~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_93997f7
c69304906938e5c5cd6b15c47~mv2_d_1632_1631_s_2.webp',
            'title' => 'Шоколадний',
            'description' =>'Насічені шоколадні коржі на французький какао Cacao
Barry, Сирно-Вершковий крем и три варіанти начинки.',
            'price' => 250
        ]);
        $product->save();
        $product = new \App\Product([
            'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_1edbb1c1503c4e6481ccd5996a118702~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_1edbb1c
1503c4e6481ccd5996a118702~mv2_d_1632_1631_s_2.webp',
            'title' => 'Морквяний',
            'description' =>'Пряні морквяні коржі з горішками і карамельною
просоченням. Сирно-вершковий крем. Відповідь на найпопулярніше запитання: морква в
торті не відчувається.',
            'price' => 280
        ]);
        $product->save();
```

```
        $product = new \App\Product([
        'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_a78375115d604801bd8e20d747c149f9~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_a783751
15d604801bd8e20d747c149f9~mv2_d_1632_1631_s_2.webp',
        'title' => 'Шоколад-кокос',
        'description' =>'Насичені шоколадні коржі на французькому какао Cacao
Barry, тонка шоколадна прошарок і кокосова начинка на основі білого шоколаду і
кокосового пюре. Сирно-вершковий крем.',
        'price' => 270
    ]);
        $product->save();
        $product = new \App\Product([
        'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_3d725f5943d8485f9d0ff02acf6fe5b6~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_3d725f5
943d8485f9d0ff02acf6fe5b6~mv2_d_1632_1631_s_2.webp',
        'title' => 'Мигдаль-карамель',
        'description' =>'Горіхові коржі на мигдальної борошні, вершкова карамель і
сирно-вершковий крем.',
        'price' => 300
    ]);
        $product->save();
        $product = new \App\Product([
            'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_c5a55764c1864264bcf76977fd930783~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_c5a5576
4c1864264bcf76977fd930783~mv2_d_1632_1631_s_2.webp',
            'title' => 'Чизкейк Орео',
            'description' =>'Брауні-основа, три шари чизкейка (класичний,
шоколадний, з крихтою Орео) і печиво Орео.',
            'price' => 350
        ]);
        $product->save();
        $product = new \App\Product([
            'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_89fb0a752edf46d39ac74d7955e7aa18~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_89fb0a7
52edf46d39ac74d7955e7aa18~mv2_d_1632_1631_s_2.webp',
            'title' => 'Чизкейк Фісташка',
            'description' =>'Тонка основа з печива Орео і ніжний чізкейк з
додаванням фісташкової пасти.',
            'price' => 350
        ]);
        $product->save();
        $product = new \App\Product([
            'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_833d717659fc4effad0919728cc3832a~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_833d717
659fc4effad0919728cc3832a~mv2_d_1632_1631_s_2.webp',
            'title' => 'Чорниця-суниця',
            'description' =>'Соковиті коржі з чорницею і дві начинки: суничний
ганаш на бельгійському шоколаді Callebaut і суничний курд. Сирно-вершковий крем.',
            'price' => 270
        ]);
        $product->save();

        $product = new \App\Product([
            'imagepath' =>
'https://static.wixstatic.com/media/e6db3c_54be9681f4c9407e925a16486d9f2492~mv2_d_
1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_54be968
1f4c9407e925a16486d9f2492~mv2_d_1632_1631_s_2.webp',
```

```
            'title' => 'Шоколад-малина',
            'description' =>'Насичені шоколадні коржі на французькому какао Cacao Barry, малинове желе і шоколадно-малинові кранчи. Сирно-вершковий крем.',
            'price' => 270
        ]);
        $product->save();
        $product = new \App\Product([
            'imagepath' => 'https://static.wixstatic.com/media/e6db3c_db36a01e4ae54ab598cd6e39c6f8a803~mv2_d_1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_db36a01e4ae54ab598cd6e39c6f8a803~mv2_d_1632_1631_s_2.webp',
            'title' => 'Наполеон',
            'description' =>'Листкові коржі з ніжним заварним кремом трьох різних смаків: ваніль, кокос і карамель. Можливий тільки у вазі 2 кг. Додатково можна прикрасити ягодами.',
            'price' => 450
        ]);
        $product->save();
        $product = new \App\Product([
            'imagepath' => 'https://static.wixstatic.com/media/e6db3c_db36a01e4ae54ab598cd6e39c6f8a803~mv2_d_1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_db36a01e4ae54ab598cd6e39c6f8a803~mv2_d_1632_1631_s_2.webp',
            'title' => 'Апельсин-манго-маракуйя',
            'description' =>'Соковиті апельсинові коржі, яскрава начинка на основі пюре манго і маракуї і білого бельгійського шоколаду Callebaut.',
            'price' => 300
        ]);
        $product->save();
        $product = new \App\Product([
            'imagepath' => 'https://static.wixstatic.com/media/e6db3c_017bdd4f5d214989b8f0872812220789~mv2_d_1632_1631_s_2.jpeg/v1/fill/w_491,h_491,al_c,q_80,usm_0.66_1.00_0.01/e6db3c_017bdd4f5d214989b8f0872812220789~mv2_d_1632_1631_s_2.webp',
            'title' => 'Чізкейк фундук-печиво',
            'description' =>'Розсипчаста основа з печива з шматочками фундука, найніжніший чізкейк зі смаком топленого печива і прошарок з карамелізованих горіхів.',
            'price' => 350
        ]);
        $product->save();
```

```
    }

}
```

DatabaseSeeder.php

```
<?php

use Illuminate\Database\Seeder;

class DatabaseSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
```

```
     * @return void
     */
    public function run()
    {
        $this->call(ProductTableSeeder::class);
    }
}
```

CreateUsersTable

```
<?php

use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class CreateUsersTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->increments('id');
            $table->string('name');
            $table->string('email')->unique();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::drop('users');
    }
}
```

CreatePasswordResetsTable

```
<?php

use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class CreatePasswordResetsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
```

```
    public function up()
    {
        Schema::create('password_resets', function (Blueprint $table) {
            $table->string('email')->index();
            $table->string('token')->index();
            $table->timestamp('created_at')->nullable();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::drop('password_resets');
    }
}
```

CreateProductsTable

```
<?php

use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class CreateProductsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('products', function (Blueprint $table) {
            $table->increments('id');
            $table->timestamps();
            $table->text('imagepath');
            $table->string('title');
            $table->text('description');
            $table->float('price');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('products');
    }
}
```

CreateOrdersTable

```php
<?php

use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class CreateOrdersTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('orders', function (Blueprint $table) {
            $table->increments('id');
            $table->timestamps();
            $table->integer('user_id');
            $table->text('cart');
            $table->string('address');
            $table->string('name');
            $table->string('payment_id');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('orders');
    }
}
```

ProductController.php

```php
<?php

namespace App\Http\Controllers;

use App\Product;
use Stripe\Stripe;
use Stripe\Charge;
use App\cart;
use App\Order;
use Session;
use Auth;
use Illuminate\Http\Request;

class ProductController extends Controller
```

```
{
     public function getIndex()
    {
       $products = Product::all();
       return view('shop.index',['products' => $products]);
    }

    public function getAddToCart(REQUEST $request, $id)
    {
       $product = Product::find($id);
       $oldcart = Session::has('cart') ? Session::get('cart') : null;
       $cart = new cart($oldcart);
       $cart->add($product,$product->id);

       $request->session()->put('cart', $cart);
       #dd($request->session()->get('cart'));
       return redirect()->route('product.index');
       #return view('shop.index',['products' => $products]);
    }

    public function getReduceByOne($id)
    {
        $oldcart = Session::has('cart') ? Session::get('cart') : null;
        $cart = new cart($oldcart);
        $cart->reduceByOne($id);
        if(count($cart->items) > 0){
            Session::put('cart', $cart);
        }else {
            Session::forget('cart');
        }
        return redirect()->route('product.shoppingcart');
    }

    public function getRemoveItem($id)
    {
         $oldcart = Session::has('cart') ? Session::get('cart') : null;
        $cart = new cart($oldcart);
        $cart->removeItem($id);

        if(count($cart->items) > 0){
            Session::put('cart', $cart);
        }else {
            Session::forget('cart');
        }


        return redirect()->route('product.shoppingcart');
    }

    public function getCart()
    {
        if(!Session::has('cart'))
        {
            return view('shop.shoppingcart');
        }
        $oldcart = Session::get('cart');
        $cart = new cart($oldcart);
        return view('shop.shoppingcart', ['products' => $cart->items, 'totalprice'
=> $cart->totalprice]);
    }

    public function getCheckout()
```

```
    {
        if(!Session::has('cart'))
        {
            return view('shop.shoppingcart');
        }
        $oldcart = Session::get('cart');
        $cart = new cart($oldcart);
        $total = $cart->totalprice;
        return view('shop.checkout', ['total'=> $total]);
    }

    public function postCheckout(REQUEST $request)
    {
        if(!Session::has('cart'))
        {
            return redirect()->route('shop.shoppingcart');
        }
        $oldcart = Session::get('cart');
        $cart = new cart($oldcart);

        \Stripe\Stripe::setApiKey('sk_test_oHxEfliecfTloBUqfXcrX4Ip');
        try{
                    $charge = \Stripe\Charge::create(array(
                    "amount" => $cart->totalprice * 100,
                    "currency" => "usd",
                    "source" => $request->input('stripeToken'),
                    "description" => "Charge")
                );
                    $order = new Order();
                    $order->cart = serialize($cart);
                    $order->address = $request->input('address');
                    $order->name = $request->input('name');
                    $order->payment_id = $charge->id;

                    Auth::user()->orders()->save($order);
        }
        catch(\Exception $e){
            return redirect()->route('checkout')->with('error', $e->getMessage());
        }

        Session::forget('cart');
        return redirect()->route('product.index')->with('success','Successfully
Purchased!! Keep Shopping!!');
    }
}
```