Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(освітній ступінь (освітньо-кваліфікаційний рівень))

на тему: *"Розробка та опис захищеності сайту ТОВ «Аргоком»"*

Виконав: студент (ка)

Спеціальності:

125 «Кібербезпека»
(шифр і назва напряму підготовки, спеціальності)

Залісковий Ю.І.
підпис (прізвище та ініціали)

Керівник *Козак Р.О.*
підпис (прізвище та ініціали)

Нормоконтроль *Лобур Т.Б.*
підпис (прізвище та ініціали)

Завідувач кафедри *Загородна Н.В.*
підпис (прізвище та ініціали)

Рецензент
підпис (прізвище та ініціали)

м. Тернопіль – 2022

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

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

Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«__» _____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека
(шифр і назва спеціальності)

Студенту Залісковому Юрію Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та опис захищеності сайту ТОВ «Аргоком»

Керівник роботи Козак Руслан Орестович, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «23» 03 2022 року № 4/7-178

2. Термін подання студентом завершеної роботи 17.06.2022

3. Вихідні дані до роботи

4. Зміст роботи (перелік питань, які потрібно розробити)

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи хорони праці	Пулька Ч.В., проф. кафедри МТ		

7. Дата видачі завдання 16.02.2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	16.02 – 19.02	*Виконано*
2.	Ознайомлення з інформацією про безпеку веб-сайтів	20.02 – 27.02	*Виконано*
3.	Опрацювання джерел в галузі дослідження	28.02 – 16.03	*Виконано*
4.	Вибір програмних засобів розробки	17.03 – 20.03	*Виконано*
5.	Реалізація програмного забезпечення	20.03-05.04	*Виконано*
6.	Оформлення розділу «Принципи безпеки веб-ресурс»	06.03 – 17.04	*Виконано*
7.	Оформлення розділу «Розробка технічного та робочого проекту»	18.04 – 29.04	*Виконано*
8.	Оформлення розділу «Розробка та тестування захищеності сайту»	30.04 – 13.05	*Виконано*
9.	Виконання завдання до підрозділу «Безпека життєдіяльності, основи хорони праці»	14.05 – 21.05	*Виконано*
10.	Оформлення кваліфікаційної роботи	22.05 – 05.06	*Виконано*
11.	Нормоконтроль	06.06 – 12.06	*Виконано*
12.	Перевірка на плагіат	10.06 – 15.06	*Виконано*
13.	Попередній захист кваліфікаційної роботи	16.06 – 19.06	*Виконано*
14.	Захист кваліфікаційної роботи	23.06.2022	

Студент _____ Залісковий Ю.І.
(підпис) (прізвище та ініціали)

Керівник проекту (роботи) _____ Козак Р.О.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка та опис захищеності сайту ТОВ "Аргоком" // / Кваліфікаційна робота ОР «Бакалавр» // Залісковий Юрій Ігорович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп’ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки,, група СБс-42// Тернопіль, 2022 // с. – 78 , рис. – 53, табл. – 0, кресл. – 0, додат. – 0.

Ключові слова: ВЕБ-САЙТ, OWASP, REACT, MERN, EXPRESS, SNYK, ХОСТИНГ, БАЗА ДАНИХ

Кваліфікаційна робота містить 4 розділи:

В першому розділі описуються основні принципи захисту веб-ресурсів, а також розказано про різні засоби розробки з порівнняням їх безпекових можливостей.

В другому розділі описано обрані засоби розробки, описано створення і доопрацювання веб-сайту використовуючи рекомендації з безпеки, з деталями про його структуру і можливості. Описано підключення верстки сайту до React, а також Express.js і його захист, розказано про функціональні можливості безпечного зв’язку користувача з сайтом.

В третьому розділі описано підключення до проекту платформи Snyk, що допомагає виявити вразливості. А також проведено підключення веб-сайту до безпечного хостингу.

В четвертому розділі розглянуті питання охорони праці, техніки безпеки.

ANNOTATION

Development and description of the security of the site of LLC "Argocom" // Qualification thesis of educational level "Bachelor"// Zaliskovy Yuriy Ihorovych // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cyber Security, SBs-42 Group // Ternopil , 2022 // p. -78 , fig. - 53, table. - 0, chair. - 0, add. - 0.

Keywords: WEBSITE, OWASP, REACT, MERN, EXPRESS, SNYK, HOSTING, DATABASE

Qualification work contains 4 sections:

The first section describes the basic principles of web resource protection, as well as various development tools with a comparison of their security capabilities.

The second section describes the selected development tools, describes the creation and refinement of the website using security recommendations, with details about its structure and capabilities. Describes the connection of the site layout to React, as well as Express.js and its protection, describes the functionality of secure user communication with the site.

The third section describes how to connect to the Snyk platform project to help identify vulnerabilities. The website was also connected to secure hosting.

In the fourth section the issues of labor protection and safety are considered.

ЗМІСТ

ВСТУП

Сьогодні Інтернет – це всесвіт взаємопов'язаних веб-сторінок, програм і медіафайлів, таких як відео, музика, фотографії та інший інтерактивний вміст, що постійно зростає. Величезний прогрес у сфері обчислювальної техніки, мініатюризації та телекомунікацій означав, що з кожним роком все більше людей можуть дозволити собі пристрій для доступу до Інтернету, за поточними оцінками, 6 з 10 людей зараз «онлайн». Навіть якщо зробити цю тезу доступною в цифровому або друкованому вигляді, ймовірно, згенеровано кілька МБ даних, якщо не більше.

На жаль, як і в інших успішних юридичних осіб або компаній, коли веб-додаток або веб-сервіс виявляється надзвичайно популярним серед користувачів або приносить значний дохід, він може привернути небажану увагу з боку злих елементів суспільства. Природа Інтернету – швидкість, зручність, анонімність та відсутність кордонів – може бути як перевагою, так і недоліком. Кіберзлочинці вже давно користуються цим фактом, і кіберзлочинність постійно зростає як за обсягом, так і за темпами, оскільки суб'єктам загроз більше не потрібно бути фізично присутніми під час скоєння злочину.

Роль веб-розробника у грандіозній схемі речей полягає в створенні послуг і додатків, які потрібні світу, і покращення повсякденного життя людей. У той час як звичайний користувач веб-програми може бачити та взаємодіяти з користувацьким інтерфейсом (UI), розробник та інші люди з технічними знаннями знають, що багато іншого відбувається під капотом для забезпечення роботи продуктів. Для розробників недостатньо створити хороший продукт чи послугу, оскільки для створеного веб-додатка знайдеться хтось, хто шукатиме вразливості програми та скористається ними для власної вигоди.

1 ПРИНЦИПИ БЕЗПЕКИ ВЕБ РЕСУРСІВ

1.1. Основні вразливості веб-ресурсів і засоби їх захисту

Розуміння поширених уразливостей у веб-додатках допомагає підприємствам краще підготуватися до захисту своїх даних від таких атак. Завдяки знанням, отриманим від досліджень, користувачі та розробники можуть бути краще підготовлені для боротьби з найпоширенішими атаками та формувати рішення для запобігання майбутнім атакам на їхні веб-додатки. Уразливості існують у багатьох формах у сучасних веб-додатках, які можна легко пом’якшити, вкладаючи час і дослідження. Зі зростанням кібератак за останнє десятиліття , веб-сайти є широкою мішенню та експлуатуються, що приносить підприємствам значні збитки . Проекти веб-розробки створюються без урахування безпеки, що може призвести до втрати активів, порушення роботи служби, розкриття конфіденційних даних або компрометації системи.

Допомогою з цими проблемами займається OWASP, що постійно займається дослідженням цифрової безпеки веб ресурсів і випускає популярний рейтинг «Топ 10 найбільших вразливостей проекту безпеки відкритих веб-додатків»,що зосередився на найкритичніших уразливостях. Open Web Application Security Project (OWASP) — це некомерційний фонд, присвячений підвищенню безпеки програмного забезпечення. Фонд підтримує список OWASP Top 10 з 2003 року, який містить рейтинг найкритичніших ризиків безпеки для веб-додатків, а також рекомендації щодо усунення 10 найбільших ризиків безпеки (OWASP, 2021). З роками цей список став стандартом безпечної розробки для веб-розробників і оновлюється кожні 3-4 роки. Рисунок 1.1 показує ітерацію списку 10 найкращих OWASP за 2021 рік і його розвиток з моменту останнього опублікування списку в 2017 році. Додавання нових категорій і зміни в рейтингах ризиків безпеки показують, що ландшафт кібербезпеки постійно змінюється. розвивається.

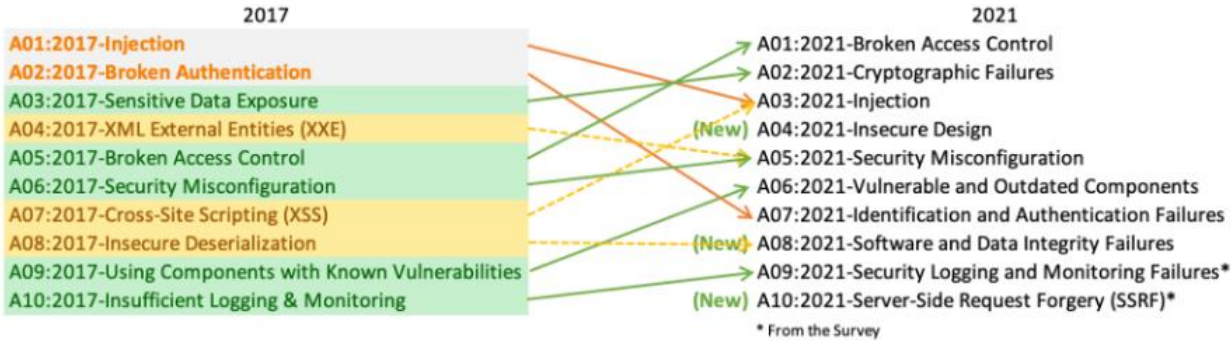

Рисунок 1.1- Зміни рейтингів 2017 і 2021 років[1]

1.1.1 Контроль доступу

Першим номером в рейтингу вразливостей 2021 року OWASP є порушений контроль доступу. -Порушений контроль доступу просунувся вгору з п’ятої позиції в 2017 році, 94% додатків були перевірені на наявність певної форми порушеного контролю доступу. 34 загальні перерахунки слабких місць (CWE), зіставлені з порушенням контролю доступу, мали більше випадків у додатках, ніж будь-яка інша категорія.

Контроль доступу забезпечує користувачам веб-програми доступ до ресурсів і виконання дій лише в межах своїх дозволів. Збій у належному впровадженні контролю доступу може призвести до несанкціонованого розкриття інформації та поставити під загрозу цілісність даних, оскільки користувачі можуть змінювати або видаляти дані поза їхніми дозволами.

Серед поширених уразливостей контролю доступу:

Порушення принципу найменших привілеїв або заборони за замовчуванням, коли доступ має надаватися лише для певних можливостей, ролей або користувачів, але доступний кожному.

Обхід перевірок контролю доступу шляхом зміни URL-адреси (змінення параметрів або примусового перегляду), внутрішнього стану програми або сторінки HTML, або за допомогою інструменту атаки, який змінює запити API.

Дозвіл перегляду або редагування чужого облікового запису шляхом надання його унікального ідентифікатора (небезпечні прямі посилання на об’єкт)

Доступ до API з відсутніми елементами керування доступом для POST, PUT та DELETE.

Підвищення привілею. Дія як користувач без входу в систему або як адміністратор під час входу як користувач.

Маніпуляції з метаданими, як-от повторне відтворення або втручання в маркер доступу JSON Web Token (JWT), файл cookie або приховане поле, яким маніпулюють для підвищення привілеїв або зловживання визнанням JWT недійсним.

Неправильна конфігурація CORS дозволяє отримати доступ до API з неавторизованих/ненадійних джерел.

Примусово переглядати автентифіковані сторінки як неавтентифікований користувач або привілейовані сторінки як звичайний користувач.

Як запобігти порушенню контролю доступу?

Оскільки впровадження контролю доступу може охопити багато вразливостей, не існує єдиного рішення для належного впровадження контролю доступу. Але щоб зрозуміти основи правильно, слід розглянути три основні аспекти – аутентифікацію, авторизацію та перевірку дозволів. Аутентифікація передбачає точну ідентифікацію користувачів під час входу в програму. З іншого боку, авторизація передбачає оцінку дій (таких як перегляд, зміна, видалення тощо), які має бути в змозі виконати автентифікований користувач. Нарешті, перевірка дозволів має справу з оцінкою рівня дозволів користувача, коли користувач виконує певні дії.

Під час реалізації авторизації кожному користувачеві потрібно призначити роль. Різні ролі зазвичай мають різні привілеї та дозволи, наприклад, адміністратор матиме більше дозволів, ніж звичайний користувач. Чим складніші бізнес-правила існують в організації щодо різних рівнів привілеїв – тим більш детальні схеми дозволів потрібно реалізувати. На

додаток до схем авторизації також необхідно встановити право власності на ресурси, оскільки можуть бути випадки, коли певні ресурси, такі як документи, записи та інші файли, належать набору користувачів і не повинні бути доступні іншим групам користувачів без дозволу. Нарешті, схеми контролю доступу можна визначити як набір політик, які, по суті, вносять у білий або чорний список набори користувачів від виконання певних дій.

1.1.2 Криптографічні збої

Наступною сходинкою рейтингу є криптографічні збої.

У списку OWASP 2017, який раніше називався «Відкриття конфіденційних даних», який описував лише симптом, а не першопричину, тепер перейменовано на Криптографічні збої в списку OWASP 2021. Оскільки конфіденційні дані можна розкрити кількома способами, перейменувавши вразливість, тепер стає цілком зрозуміло, що зосереджено на збоях, пов’язаних із погано реалізованими заходами криптографії.

Визначення цієї вразливості по суті стверджує, що конфіденційні дані або не захищені, або захищені недостатньою криптографією – це стосується як даних, що зберігаються, так і даних, що передаються.

Будь-які дані чи інформація, які організація хотіла б зберегти конфіденційними та недоступними для неавторизованих сторін, слід розглядати як конфіденційні дані. Це може включати широкий набір даних, таких як змінні середовища, ключі, паролі, імена користувачів веб-платформи, до даних користувачів, таких як адреси електронної пошти, імена, номери соціального страхування, або фінансові дані, такі як номери кредитних карток, заборонений обліковий запис. числа тощо. Виходячи з цього списку, очевидно, що деякі набори даних є більш чутливими, ніж інші. Такі дані, як паролі, повинні бути зашифровані таким чином, що неможливо відновити їх у вигляді простого тексту. Інші дані, такі як адреси електронної пошти та імена, імовірно, можна зберігати у вигляді простого тексту, але, швидше за все, законодавство, таке як

GDPR, означає, що таку інформацію також слід вважати конфіденційною та певною мірою захищеною. Основне практичне правило полягає в тому, що якщо є сумніви, чи є деякі дані конфіденційними чи ні, то найкраще забезпечити їм такий рівень шифрування, який все одно дозволить використовувати дані в програмі з розумною легкістю.

Недостатня криптографія Основна концепція включення криптографії полягає у використанні шифрів, які неможливо зламати протягом розумного проміжку часу за допомогою обчислювальної потужності, доступної сьогодні. Це не обов’язково означає використання шифрів, які неможливо зламати в абсолютних значеннях, а скоріше використання шифрів, які роблять спроби зламати шифр складними та непрактичними з точки зору часу та зусиль. Отже, за визначенням недостатня криптографія — це криптографія, яка не забезпечує достатніх рівнів безпеки, оскільки може бути легко скомпрометована зловмисником за достатньо часу та зусиль.

Навіть коли відповідна криптографічна функція була визначена та задіяна у вашому продукті, дуже доцільно бути в курсі новин, пов’язаних із криптографією, щоб відстежувати, коли алгоритм більше не вважається безпечним. Існують також інші важливі міркування під час правильного використання криптографічної функції, щоб з часом безпека не була порушена. Необхідно неухильно дотримуватися таких аспектів, як використання унікальних солей під час хешування даних, наявність надійних ключів шифрування та регулярне змінення ключів шифрування. [1]

1.1.3 Атаки ін'єкції

Атаки ін'єкції займають третє місце рейтингу, хоча раніше були перші за поширеністю.

Ін'єкція стосуються спроби зловмисника надати програмі ненадійний вхід, який потім обробляється інтерпретатором програми і змінює виконання

коду таким чином, що може призвести до несанкціонованого доступу до даних і, в гіршому випадку, до повної компрометації системи. .

Ін'єкційні атаки відносяться до широкого класу векторів атаки. Атаки з ін’єкцією відбуваються, коли зловмисники знаходять слабкі місця в перевірці введених даних і вводять недовірені введення, які потім обробляються веб-додатком і змінюють передбачуване виконання серверної обробки.

Деякі поширені типи ін'єкційних нападів:

- Ін'єкція SQL – вразливості ін'єкції SQL виникають, коли запити до бази даних створюються небезпечним способом, а недовірені дані інтерпретуються як частина запиту SQL. Наприклад, у програмі Java, яка дозволяє нам отримувати користувачів на основі їх електронної пошти, наступний сценарій, коли ми створюємо запит із конкатенацією рядків, є дуже небезпечним: у наведеному вище прикладі об’єднана електронна пошта може посилатися на небезпечне введення користувача, що означає, що шкідливі параметри можна легко передати. Замість цього набагато безпечніше використовувати параметризовані оператори, які забезпечують безпечну обробку введених даних користувача (тобто параметрів), що передаються в SQL-запити.
- Міжсайтові сценарії (XSS) – ця вразливість зловживає параметрами HTTP для введення шкідливого коду JavaScript у веб-програму.
- Виконання команд – подібна уразливості до сценаріїв XSS, які зловживають параметрами HTTP для введення команд оболонки, які виконуються на сервері.

1.1.4 Небезпечний дизайн

На відміну від першої думки, що спадає коли чуєш словосполучення небезпечний дизайн, ця вразливість не про неестетичне поєднання кольорів чи

геометричних рішень в інтерфейсі додатку, не про UI/UX, ця проблема набагато більш глибока.

Коли справа доходить до проектування безпечних веб-додатків, потрібно багато планування та ретельного продумування, перш ніж розпочнеться впровадження коду. На цьому етапі створюється план архітектури програми. Коли це буде зроблено, потрібно також розглянути, з якими потенційними загрозами зіткнеться програма, як виглядає поверхня атаки та як пом’якшити загрози. Цей крок називається «моделюванням загроз», і він допомагає визначити, з якими загрозами зіткнеться програма, зрозуміти, як і де шкідливий вхід може потрапити в програму, передбачити умови, за яких програма може вийти з ладу, і, отже, спланувати контрольоване збій. Зазвичай моделювання загроз починається з малювання ескізу діаграми потоку даних, яка ілюструє, як дані зберігаються, обробляються та протікають через програму разом із взаємодіями та межами довіри, які складають програму.

Небезпечний дизайн не є джерелом для всіх інших 10 категорій ризику. Існує різниця між небезпечним дизайном і небезпечною реалізацією. Ми неспроста розрізняємо недоліки дизайну та дефекти реалізації, вони мають різні першопричини та їх усунення. Захищений дизайн все ще може мати дефекти реалізації, які призводять до вразливостей, які можуть бути використані. Захищений дизайн — це культура та методологія, які постійно оцінюють загрози та гарантують, що код надійно розроблений та перевірений для запобігання відомим методам атак.

1.1.5 Неправильна конфігурація безпеки

Неправильні налаштування безпеки є основною причиною вразливостей. Зрештою, навіть найбезпечніше закодована програма може бути вразливою, якщо конфігурації безпеки визначені недостатньо добре. Насправді, неправильно налаштовані параметри безпеки є надзвичайно поширеною причиною вразливостей безпеки, і зловмисники досить легко виявляють і

використовують їх. Багато баз даних, серверів і систем керування вмістом мають облікові записи за замовчуванням, що дозволяє розробникам швидко розпочати розробку. Але це також може призвести до випадкової доставки програмного забезпечення з увімкненими обліковими записами за замовчуванням, які зловмисники швидко винюхують і використовують.

Запобігти цьому доволі легцо ,можна вчинити, як мінімум такі дії:

- Мінімальна платформа без будь-яких непотрібних функцій, компонентів, документації та зразків.
- Завдання для перегляду та оновлення конфігурацій, що відповідають усім нотаткам безпеки, оновленням і виправленням, як частина процесу керування виправленнями .
- Сегментована архітектура додатків забезпечує ефективне та безпечне розділення між компонентами або клієнтами за допомогою сегментації, контейнеризації або груп безпеки в хмарі (ACL).
- Надсилання директив безпеки клієнтам, наприклад, заголовків безпеки.
- Автоматизований процес для перевірки ефективності конфігурацій і налаштувань у всіх середовищах.

1.1.6 Уразливі та застарілі компоненти

Сучасна веб-розробка в значній мірі покладається на фреймворки з відкритим кодом, бібліотеки та інструменти, створені іншими розробниками. Рішення поширених проблем у розробці полягає в тому, щоб просто використовувати інструмент, який вже розроблено, а не впроваджувати рішення з нуля. Зазвичай, підбираючи бібліотеки та пакети, розробники звертають увагу на те, наскільки бібліотека стара і зріла, наскільки активно вона розвивається, її рейтинг популярності (кількість зірок, яку вона отримала), а також кількість щотижневих завантажень. Усі ці аспекти разом дають

уявлення про надійність та надійність бібліотеки. Але навіть вибираючи останні, добре перевірені і надійні версії програмного забепечення, яке ви застосовуєте можна залишатись вразливим з таких причин:

- Якщо ви не знаєте версії всіх компонентів, які використовуєте (як на стороні клієнта, так і на стороні сервера).
- Якщо програмне забезпечення є вразливим, не підтримується або застаріло. Це включає ОС, веб-сервер/сервер додатків, систему управління базами даних (СУБД), програми, API та всі компоненти, середовища виконання та бібліотеки.
- Якщо ви регулярно не скануєте на наявність уразливостей і не підписуєтесь на бюлетені з безпеки, пов’язані з компонентами, які ви використовуєте.
- Якщо ви не виправляєте чи не оновлюєте базову платформу, фреймворки та залежності вчасно та з урахуванням ризиків.
- Якщо розробники програмного забезпечення не перевіряють сумісність оновлених, оновлених або виправлених бібліотек.

1.1.7 Помилки ідентифікації та автентифікації

Ця вразливість, що займає сьоме місце в рейтингу і, мабуть першою спадає на думку коли згадують про небезпеки в мережі, пов'язана з неналежним впровадженням компонентів, пов'язаних з автентифікацією та ідентифікацією, у програмі. Ці компоненти використовуються для ідентифікації входу користувачів, автентифікації користувачів, керування сеансами користувачів, серед іншого. Якщо ці компоненти не впроваджені належним чином, зловмисники можуть використовувати паролі користувачів або маркери сеансу та тимчасово або назавжди захопити облікові записи та ідентифікатори користувачів. Програма, яка має недоліки автентифікації, може мати деякі з таких помилок:

- Програма дозволяє користувачам використовувати слабкі паролі.

- Паролі користувачів програми можна зламати за допомогою грубої сили або інших автоматизованих атак.
- Програма використовує слабкі або неефективні процеси відновлення облікових даних облікового запису користувача.
- Програма зберігає паролі у вигляді простого тексту, або паролі слабо зашифровані.
- Програма не пропонує багатофакторну аутентифікацію. Неправильне керування сеансом.
- Програма розкриває ідентифікатори сеансів у URL-адресі.
- Після успішного входу в систему програма не змінює ідентифікатори сеансів.
- Програма не анулює ідентифікатори сеансів належним чином.

Як запобігти помилкам ідентифікації та автентифікації?

Якщо можливо, запровадьте багатофакторну автентифікацію, щоб запобігти автоматичному підбору облікових даних, грубій силі та атак повторного використання вкрадених облікових даних.

Не відправляйте та не розгортайте з обліковими даними за замовчуванням, особливо для адміністраторів.

Впроваджуйте слабкі перевірки паролів, такі як тестування нових або змінених паролів у списку 10 000 найгірших паролів.

Узгодьте політику довжини, складності та ротації пароля з рекомендаціями Національного інституту стандартів і технологій (NIST) 800-63b

Переконайтеся, що реєстрація, відновлення облікових даних і шляхи API захищені від атак перерахування облікових записів, використовуючи однакові повідомлення для всіх результатів.

Обмежте або все частіше відкладайте невдалі спроби входу і реєструйте їх.

1.1.8 Збій цілісності програмного забезпечення та даних

Офіційне визначення стверджує, що порушення цілісності програмного забезпечення та даних стосуються «коду та інфраструктури, які не захищені від порушення цілісності» (OWASP, 2021). Типові речі, які можуть піти не так під цією категорією вразливості:

- Програмне забезпечення, що використовує сторонні плагіни, бібліотеки або пакети з ненадійних джерел, репозиторіїв і мереж доставки вмісту (CDN). Завжди є ймовірність того, що навіть найбільш широко використовувані та надійні репозиторії можуть бути скомпрометовані.
- Неавторизовані сторони потенційно можуть отримати доступ до недостатньо захищених конвеєрів CI/CD і стати точкою входу для зловмисного коду або повної компрометації системи.
- Залишити параметри за замовчуванням для таких функцій, як функція автоматичного оновлення, також може становити ризик, оскільки завантаження та застосування оновлень програмного забезпечення без достатньої перевірки цілісності даних оновлення програмного забезпечення є великим червоним прапором для критичних систем.

Існує кілька рекомендацій, яких можна дотримуватися, щоб зменшити ймовірність збоїв програмного забезпечення та цілісності даних. До них належать:

- Перевірка модулів, пакетів, оновлень програмного забезпечення за допомогою цифрових підписів, щоб переконатися, що дані отримані з того джерела, з якого ви очікуєте, що вони мають надходити, і не були підроблені.
- Розміщення внутрішнього відомого і перевіреного репозитарію як проксі.

- Використання аудиту пряжі, перевірки залежностей OWASP або інших інструментів безпеки ланцюга поставок програмного забезпечення для перевірки відомих уразливостей у компонентах, які використовує ваша програма.
- Наявність встановленого процесу перегляду коду у вашій команді управління інфраструктурою.

1.1.9 Реєстрація безпеки та моніторинг збоїв

Журналювання збоїв відноситься до документування стану програми в будь-який момент часу і може надавати таку інформацію, як активність користувача, продуктивність системи та помилки, які виникають у програмі. Журнали в основному використовуються для відстеження того, як програма використовується, покращення та відновлення програми. Наприклад, на веб-сервері журнали зберігаються для кожного HTTP-запиту, який отримує веб-сервер, позначки часу запиту, методу HTTP і коду відповіді HTTP, який веб-сервер надсилає назад. У більшості випадків вірогідність серйозних інцидентів безпеки різко зростає через недостатню реєстрацію та моніторинг. Недостатнє ведення журналу та моніторингу ускладнює виявлення підозрілої поведінки (наприклад, багаторазові невдалі спроби входу або автоматичні запити до певних кінцевих точок) і в кінцевому підсумку збільшує шанси, що зловмисник зможе використати програму. Сценарій атаки, який передбачає недостатнє ведення журналу та моніторингу, потенційно може виглядати так: зловмисник отримує доступ до внутрішньої мережі, а потрапивши в мережу, зловмисник починає сканувати на наявність уразливих компонентів і незашифрованих конфіденційних даних. Оскільки журнал і моніторинг не ведуться, зловмисник може продовжувати отримувати інформацію протягом тривалого періоду часу, і злом продовжується непоміченим. Як правило, порушення виявляються лише тоді, коли зловмисник оприлюднює витік даних, продаючи дані в темній мережі або звернувшись безпосередньо до власника даних для отримання викупу.

Як запобігти збоям у журналі безпеки та моніторингу?

- Переконайтеся, що всі помилки входу, контролю доступу та перевірки введення на стороні сервера можуть бути зареєстровані з достатнім контекстом користувача, щоб ідентифікувати підозрілі чи шкідливі облікові записи, і утримуватися протягом достатнього часу, щоб дозволити відкладений криміналістичний аналіз.
- Переконайтеся, що журнали створюються у форматі, який можуть легко використовувати рішення для керування журналами.
- Переконайтеся, що дані журналу закодовані правильно, щоб запобігти ін’єкціям або атакам на системи реєстрації чи моніторингу.
- Переконайтеся, що транзакції з високою вартістю мають контрольний журнал із засобами контролю цілісності, щоб запобігти підробці або видаленню, наприклад, таблиці бази даних лише для додавання тощо.
- Команди DevSecOps повинні налагодити ефективний моніторинг та оповіщення, щоб підозрілі дії виявлялися та швидко реагували на них.

1.1.10 Підробка запитів на стороні сервера

Підробка запитів на стороні сервера (SSRF) — це вразливість, яка виникає, коли сервер веб-додатків обманом надає зловмиснику доступ до неавторизованих ресурсів (як для перегляду, так і для зміни). Програми, які підтримують читання даних, наданих користувачем, є основною метою цього типу атаки. Під час цієї атаки зловмисник створює зловмисний запит, що містить відомості про ціль (наприклад, через URL-адресу), який спочатку надсилається на головний веб-сервер програми. Потім код на стороні сервера обробляє запит зловмисника і ініціює вторинний запит до цілі, якою зазвичай є внутрішні служби, бази даних або інші внутрішні сервери в мережі, які не мають доступу до загальнодоступного Інтернету. Відповідь цих внутрішніх служб, яка містить несанкціоновані дані, потім передається зловмиснику, який завершує атаку.

Основні ризики, які несе SSRF, — це розкриття даних, атаки «Відмова в обслуговуванні» (DoS), а також здійснення зловмисником розвідки та відбитків пальців служб. Однією з причин зростання атак, пов’язаних із SSRF, є збільшення використання хмарних сервісів у сучасних веб-додатках. Атака SSRF з метою отримання доступу до облікових даних хмарної служби (і, отже, даних, що зберігаються в цій хмарній службі) є однією з найбільш поширених мотивів для зловмисників. Атаки SSRF також використовуються для розвідувальних цілей, зосереджених на службах, що працюють у внутрішніх мережах. Оскільки встановлені методи безпеки рекомендують мінімізувати площу атаки, лише кілька серверів програми є справді загальнодоступними, а інші сервери, зарезервовані для внутрішнього функціонування та зв’язку, недоступні із зовнішніх мереж. [2]

1.2 Аналіз безпеки рішень для розробки

Веб-додаток (Web app) — це прикладна програма, яка зберігається на віддаленому сервері та передається через Інтернет через інтерфейс браузера. Веб-додаток зазвичай складається з трьох основних частин –

- Frontend,
- Backend,
- База даних.

Існує багато постачальників для трьох вищевказаних компонентів. Деякі приклади:

- Frontend – Angular, ReactJS, Vue.js тощо.
- Backend – NodeJS, php, Laravel тощо.
- База даних – MySQL, MongoDB, Cassandra, Redis тощо. [3]

1.2.1 Frontend

Інтерфейс використовується для відображення основного вмісту веб-сторінки, зазвичай це єдине, що бачить клієнт після відвідування сайту. Якщо веб-сайт статичний, потрібен лише передній край. HTML використовується для формування макета всієї сторінки, CSS використовується для надання стилю сторінки, а JavaScript використовується для надання логіки сторінці.

AngularJS - JavaScript-фреймворк з відкритим програмним кодом, який розробила компанія Google. З'явився він у 2010 році і був першим популярним фреймворком, який вийшов на ринок. Основними особливостями, які зараз виділяють Angular з поміж інших популярних фреймворків є те, що він використовується разом із мовою TypeScript, MVVM (Model-View-ViewModel), що дозволяє розробникам працювати окремо над одними і тими ж розділами програми, використовуючи один і той же набір даних, має велику продуманість і надійність, а основним недоліком є відносна складність в порівнянні з конкурентами.

React - JavaScript-фреймворк з відкритим програмним кодом, який розробляє компанія Facebook. В 2013 році вперше вийшла бібліотека, яка тоді називалася React.js, що стала першим серйозним конкурентом для AngularJS. За короткий час даний фреймворк зайняв велику нішу на ринку.

Основними перевагами React є: простота у вивченні, завдяки простому дизайну, використанню JSX (HTML-подібний синтаксис) для шаблонів, дуже докладної документації і простої і найопопулярнішої платформи Redux, дуже велика швидкість, завдяки реалізації React Virtual DOM, відмінна підтримка Progressive Web App (PWA) завдяки генератору додатків `create-react-app` і можливості також використовувати React Native. [4]

Веб-сайти, які мають лише інтерфейс, схильні до атак із застосуванням міжсайтових сценаріїв (XSS). Angular і React, як передові фреймворки, мають вбудований механізм для запобігання ін'єкції XSS, однак під час створення веб-сайту з використанням примітивних технологій потрібно подбати про різні

кроки, щоб успішно запобігти ін’єкції XSS. Наприклад в React безпека від ін’єкцій виконується таким чином: за замовчуванням React DOM екранує будь-які значення, вбудовані в JSX, перед їх відтворенням. Це гарантує, що ви ніколи не зможете вводити те, що явно не написано у вашій програмі. Все перетворюється в рядок перед відображенням. Це допомагає запобігти атакам XSS.Під капотом React автоматично викриває будь-який ввод користувача перед його рендерингом. Це означає, що введення користувача ніколи не буде виконано, що робить React безпечним з точки зору XSS. Крім того, React надає набір заходів, які можна використовувати для запобігання атак XSS.

1.2.2 Бекенд

Бекенд необхідний для динамічного веб-сайту, він зазвичай відповідає за зв'язок з базою даних і надання даних на інтерфейс для відображення. Будь-яка конфіденційна інформація зазвичай зберігається в серверній частині і ніколи не надсилається на інтерфейс, тому що клієнт може бачити дані інтерфейсу, однак отримати доступ до даних у серверній частині дуже важко. Мови інтерфейсу можуть працювати лише у браузері, проте мови бекенда можна використовувати для зв’язку з системою, в якій вони скомпільовані.

1.2.3 База даних

База даних є місцем для зберігання будь-яких даних, які необхідно зберегти і які не повинні бути доступні клієнту. В основному існує два типи баз даних: база даних SQL і база даних NoSQL. Серед SQL баз даних найпопулярнішою і найвпізнаванішою є MySQL, а серед NoSQL – MongoDB.

База даних SQL: ці типи баз даних використовуються, коли дані, що зберігаються, структуровані, і структура не очікується змін.

База даних NoSQL: коли дані не мають структури і можуть відрізнятися від користувача до користувача, тоді зазвичай використовується база даних NoSQL.

MySQL – система керування базами даних розроблена компанією TcX в 1995 році, яка існуючи 25 років майже не втрачає свою популярність. MySQL доводить свою надійність і перевіреність використовуючись близько 20 років в таких відомих корпораціях як: Apple, Amazon, Joomla! , Google, Wikipedia, NASA, Yahoo! і інших. Крім надійності і ефективної системи безпеки, основними перевагами цієї СКБД є: простота отримання і використання, гнучкість яка допоможе легко перевести всю інформацію в інші SQL бази даних, підтримування можливості зберігати близько 50 мільйонів рядків у таблиці, висока швидкість виконання команд і гнучкість у адмініструванні.

MongoDB – нереляційна система керування базами даних розроблена в 2009 році і написана на мові C++. MongoDB швидко виріс і став популярною базою даних саме для веб-додатків і клієнтів, бекенда та рівня баз даних. База даних є фізичним контейнером для колекцій. Кожна база даних отримує власний набір файлів у файловій системі. Один сервер MongoDB зазвичай має кілька баз даних. MongoDB — це набір документів. Це еквівалентно таблиці СУБД. Колекція представлена в базі даних**.** Основною перевагою MongoDB є підтримка JSON-формату, який є простішим при роботі з JavaScript, є гнучким і для деяких завдань зручнішим, ніж з додаванням колонок в SQL-базах даних. Значною перевагою для JavaScript розробників є синтаксис MongoDB, який дуже схожий на JS. MongoDB часто вибирають, коли потрібна більша швидкість розробки і прості рішення. [5]

Порівняння безпеки Mysql та Mongodb Нижче наведено порівняння між MySQL та MongoDB з точки зору безпеки:

1) Модель безпеки

– MySQL забезпечує модель безпеки на основі привілеїв, тобто надає користувачеві доступ лише до певних команд, таких як CREATE,

UPDATE, DELETE тощо, отже, на основі типу користувача такі привілеї можуть бути визначені.

– MongoDB підтримує TLS та SSL для шифрування, щоб забезпечити доступ до даних лише для призначеного користувача

2) Ін'єкції

– MySQL схильний до ін'єкцій SQL, які, по суті, розміщують шкідливий код у операторах SQL за допомогою виводу веб-сторінки.

– Хоча MongoDB не схильний до ін'єкцій SQL, він не повністю схильний до помилок від ін'єкцій через використання мови для інтерпретації, такої як JavaScript. Проте перевага в захищеності до SQL ін’єкцій, які є найпопулярнішими ін’єкціями, є дуже суттєвою в безпеці проєкту.

3) Логгінг

– MySQL пропонує повне ведення журналу за замовчуванням, а підтримка транзакцій і відкатів допомагає забезпечити цілісність даних.

– Повне ведення журналу не ввімкнено за замовчуванням у MongoDB. Додаткове ведення журналу вбудовано в операційну систему та рівні програми.

4) Контроль доступу

– MySQL надає різні типи механізмів контролю доступу, як-от команди дискреційного контролю доступу (REVOKE & GRANT), контроль доступу на основі ролей тощо.

– MongoDB пропонує лише рольовий контроль доступу, який не ввімкнено за замовчуванням. Він надає деякі вбудовані ролі, які надають набір привілеїв, які зазвичай потрібні в базі даних.

5) Модель цілісності

– MySQL слідує моделі ACID (атомна, послідовна, ізольована, довговічна). Реляційна база даних, яка не відповідає жодній із цих чотирьох цілей, не вважається надійною. Адміністратори баз даних

використовують кілька стратегій для застосування ACID, таких як ведення журналу попереду запису (WAL), тіньове сторінки та двофазний протокол фіксації.

– MongoDB дотримується моделі BASE (базова доступність, м’який стан, можлива узгодженість). З випуском MongoDB 4.0 ми тепер маємо підтримку транзакцій ACID з кількома документами. Завдяки ізоляції моментальних знімків транзакції забезпечують узгоджене представлення даних, одночасно забезпечуючи виконання повністю або нічого та підтримуючи цілісність даних.[6]

2 РОЗРОБКА ТЕХНІЧНОГО ТА РОБОЧОГО ПРОЕКТУ

2.1 Опис та обґрунтування вибору методу розробки

MERN стак був вибраний як оптимальне рішення для проєкту, хоча як кожна технологія має певні проблеми з безпекою, вони повинні бути мінімалізовані, що черездодаткові заходи, частину з яких буде показано в дипломому проєкті. Важливою перевагою MERN перед готовими рішеннями на кшталт CMS є необхідність розробляти і налаштовувати все самому, а також продумувати архітектуру сайту і план по його створенню, разом з пошуком особистих і готових засобів його безпеки, як це необхідно робити зхідно з четвертим номером списку OWASP. Розпочати роботу з CMS типу WordPress дуже легко, проте ви витратите набагато більше часу і ресурсів на масштабування та налаштування свого сайту WordPress, а концепція веб-ресурсу не буде такою продуманою

JavaScript-бібліотека React забезпечує новий підхід до системи вихідних даних, оскільки вона є декларативною і заснованою на компонентах. Розробнику потрібно лише описати, як різні частини інтерфейсу виглядають у кожному стані додатку і React ефективно оновить та відрендерить лише потрібні компоненти, коли відповідні дані зміняться. [7]

JSX синтаксис забезпечує можливість працювати з HTML кодом, майже без змін, але з можливістю працювати з вхідними даними, задекларованими в React чи розміщеними в базах даних.

Технологія Express забезпечує легку співпрацю проекту розробленого на React з різними серверними мовами, що дозволяє користувачу посилати вхідні дані на сервер.

2.2 Розробка та огляд сайту

Структуру сайту було спроектовано і продумано на початку роботи, як описано в пунктах 5 і 6 рейтингу OWASP, так щоб всі елементи мали на меті

вирішити конкретну проблему і позбавити необхідності додаванні зайвих елементів в структуру, але зберігалась можливість додавання нових елементів, дизайн окреслений самостійно користуючись ресурсом Figma для його певної стрктуризації. Тоді почалась версткао за допомогою HTML через простоту і можливість легко підлаштувати всі елементи під себе.

Структура сайту – це його розділи, підрозділи і сторінки. А також навігація, яка забезпечує доступ до них, тобто різні меню, перехресні посилання і карта сайту. На етапі планування сайту і написання технічного завдання необхідно ретельно її продумати. Зручна і зрозуміла структура сайту допомагає користувачеві легко знаходити потрібну йому інформацію. Необхідність структурування сайту (як зовнішнього, так і внутрішнього) відчули багато веб-розробників і SEO-оптимізатори.

Слід позначити основні причини дотримання грамотної організації сайту:

- Розробка без проблем. Якщо чітко усвідомлювати контури проекту і розуміти всі внутрішні зв’язки між об’єктами – час написання коду буде мінімальним, а продукт – якісним.
- Можливість зміни. Якщо проект успішний, він повинен розвиватися і розширюватися, а це неодмінно веде до збільшення сторінок і функцій сайту. В будинку можна побудувати ще кілька поверхів, якщо фундамент і стіни міцні, точно так само справа йде і з веб-розробкою.
- Зручність користувачів. Якщо сторінка своєї зовнішньої організацією допомагає користувачеві здійснити необхідну дію, а не викликає бажання натиснути на червоний хрестик – значить, все зроблено правильно. Ще однією важливою деталлю є можливість швидко дістатися до будь-якої інформації на сайті, а не нескінченно зариватися в каталоги.
- Високе ранжирування. Сайти з непродуманою структурою можуть опинитися в топі лише за щасливою випадковістю, так як організація контенту надає неабиякий вплив на позиції в результатах пошуку.

– Логічна структура сайту повинна давати можливість пошуковим системам індексувати всі сторінки. Слід дотримуватися структуру каталогів, а також уникати дублювання їх назв.

– Надати користувачам можливість потрапити в будь «куточок» сайту не більше ніж за 3 переходи. Таким чином, відвідувач буде усвідомлювати своє місце розташування, і почувати себе на сторінках ресурсу максимально комфортно.

Було прийнято рішення створити 6 окремих сторінок – одну початкову з основою інформацією й інші, такі як: «Акції», «Новини», «Тарифи», «Підтримка» і «Контакти» з детальнішою інформацією і функціоналом, на які можна перейти за допомогою посилань на них в верхньому меню. (див. рис. 2.1).

Рисунок 2.1 – Головне меню

Крім посилань в меню є також логотип компанії, кнопка для переключення на нічний темний режим сайту, зроблена за допомогою компонентів React, які змінюють стилі (див. рис. 2.2), та кнопка «Передзвоніть мені», для можливості користувачем залишити запит на двінок.

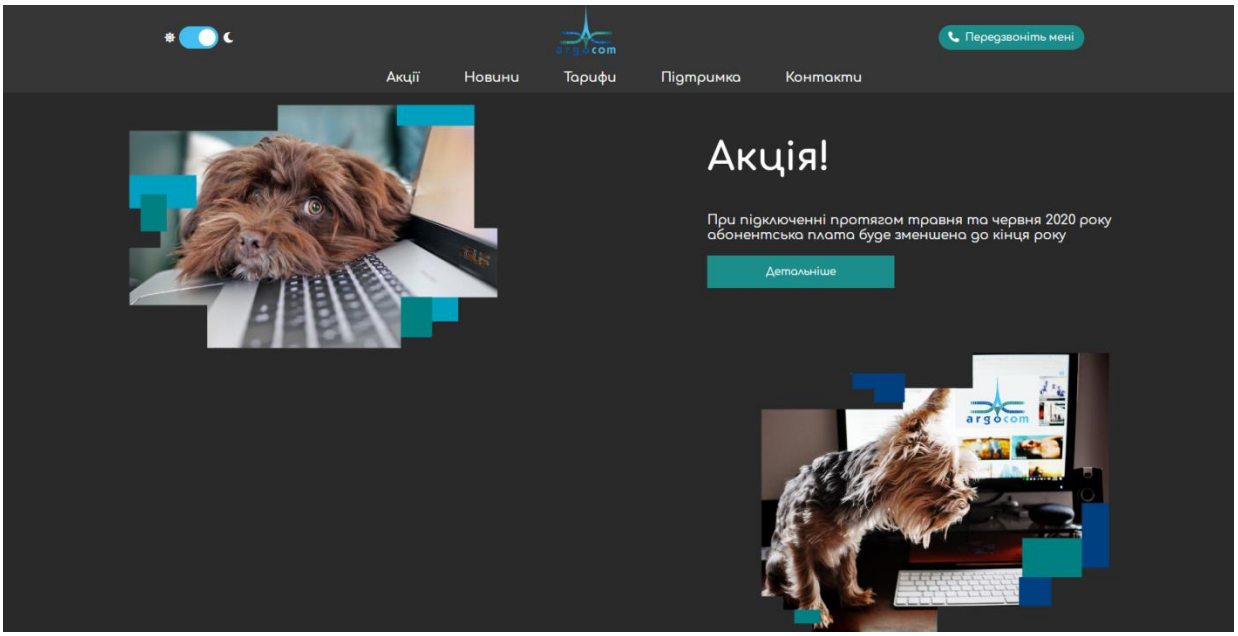

Рисунок 2.2 – Темний режим сайту

Початкова сторінка складається з декількох сторінок, які займають всю висоту екрану (фулскрін-сторінки), на яких є інформація і посилання на інші сторінки сайту (див рисунки 2.5, 2.6). Інші сторінки не мають певного обмеження висоти екрану, їх висоту регулює сам контент.

Для забезпечення адаптивності розміри блоків здебільшого задавались параметрами, які є гнучкими під час регулювання розміру сайту, наприклад: %, vh, vmin і інші.

Позиціювання елементів здебільшого здійснювалось за допомогою елементу Flexbox.

Для кращого забезпечення адаптивності використовувались медіа-запити, які дозволяють змінювати стилі елементів залежно від розміру екрана. (див. рис. 2.3, 2.4).

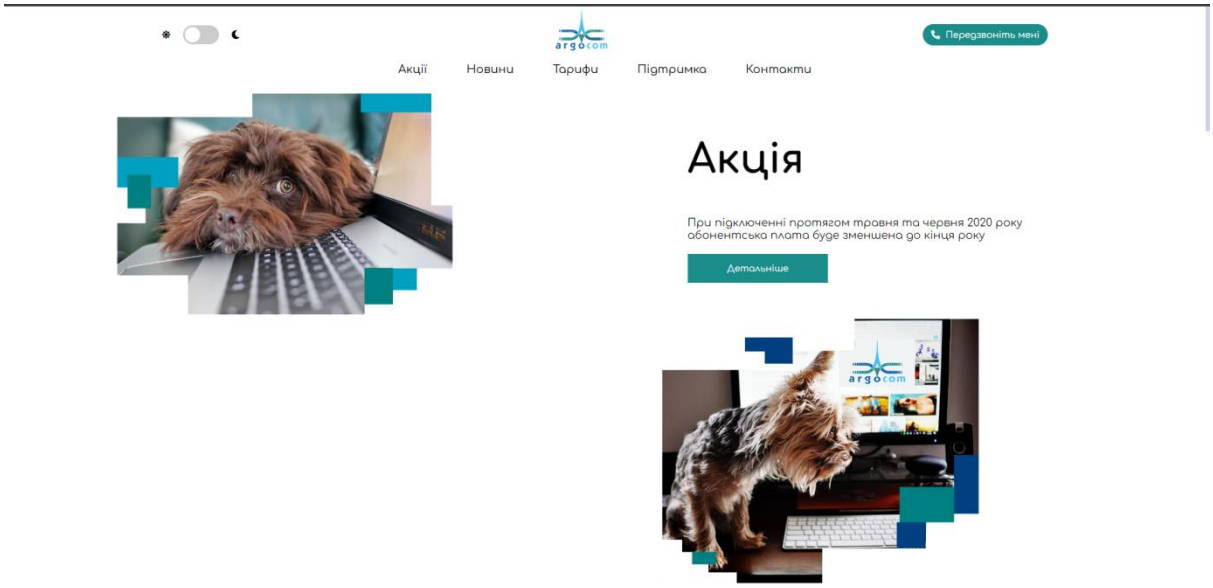

Рисунок 2.3 – Початкова сторінка з повним розширенням

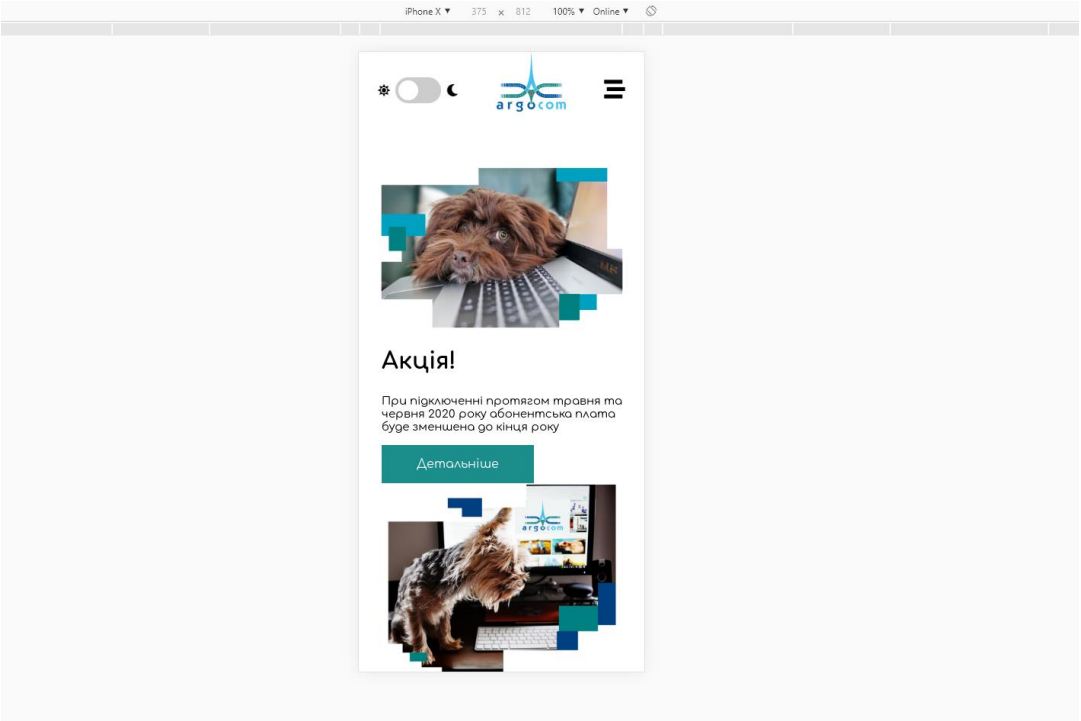

Рисунок 2.4 – Початкова сторінка з розширенням під iPhone X

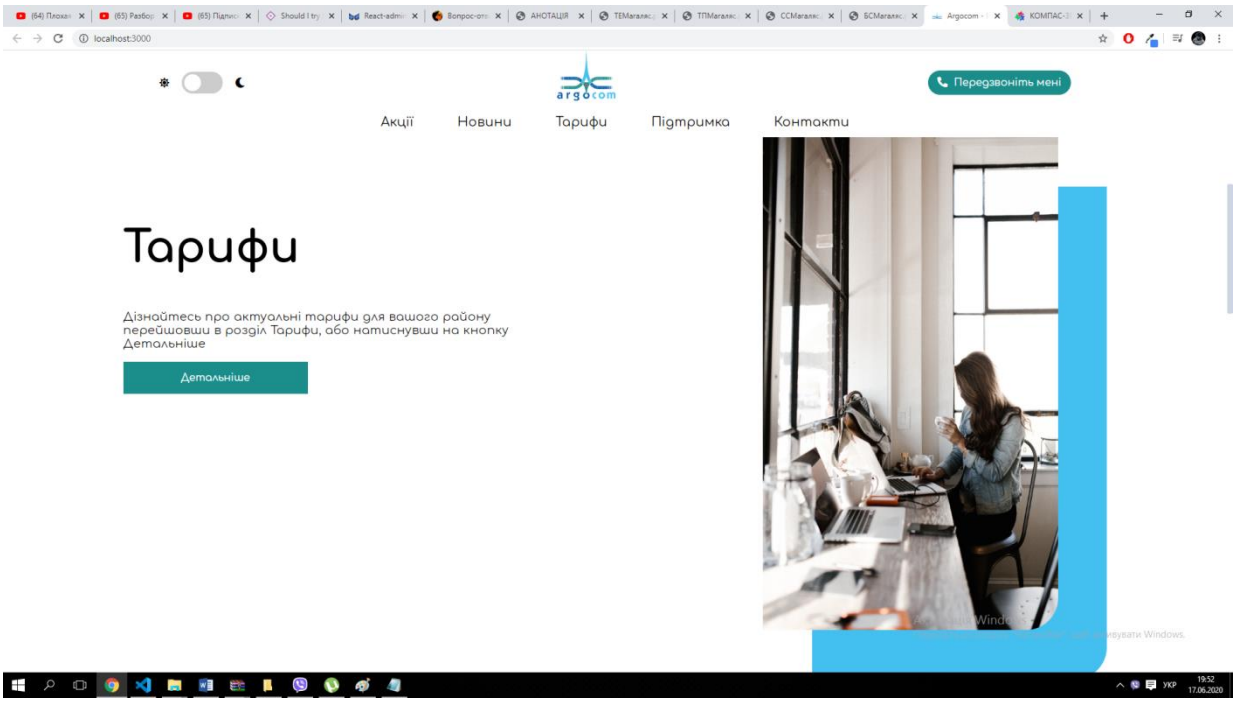

Рисунок 2.5 - Частина головної сторінки

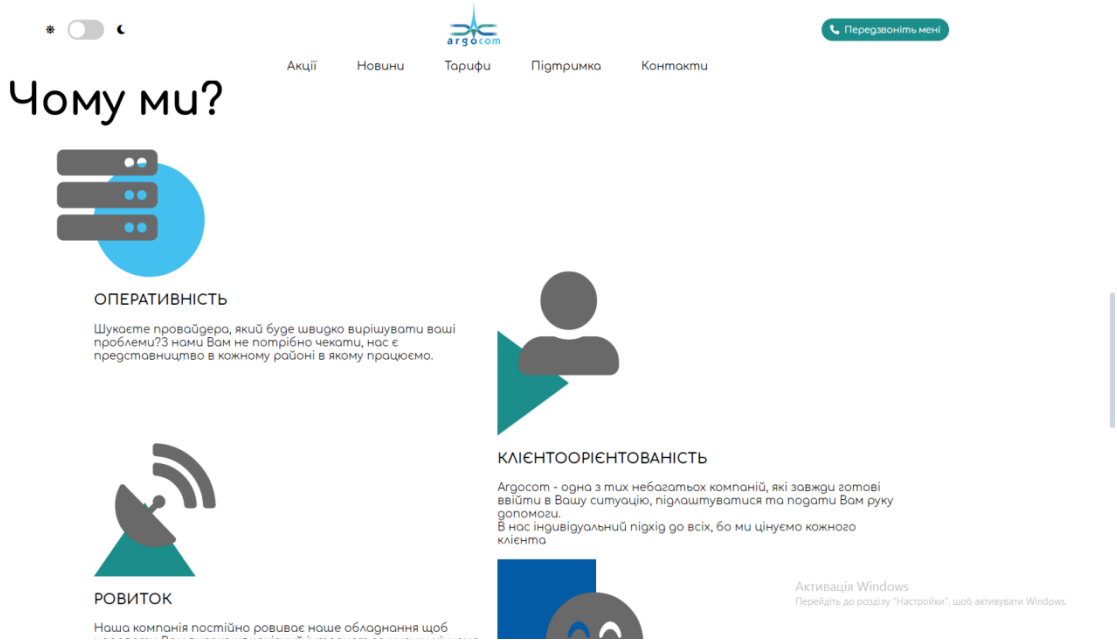

Рисунок 2.6 – Частина головної сторінки

В кінці кожної сторінки присутній нижній блок футер (див. рис. 2.7) в якому розміщена інформація про контакти і також посилання на сторінки сайту.

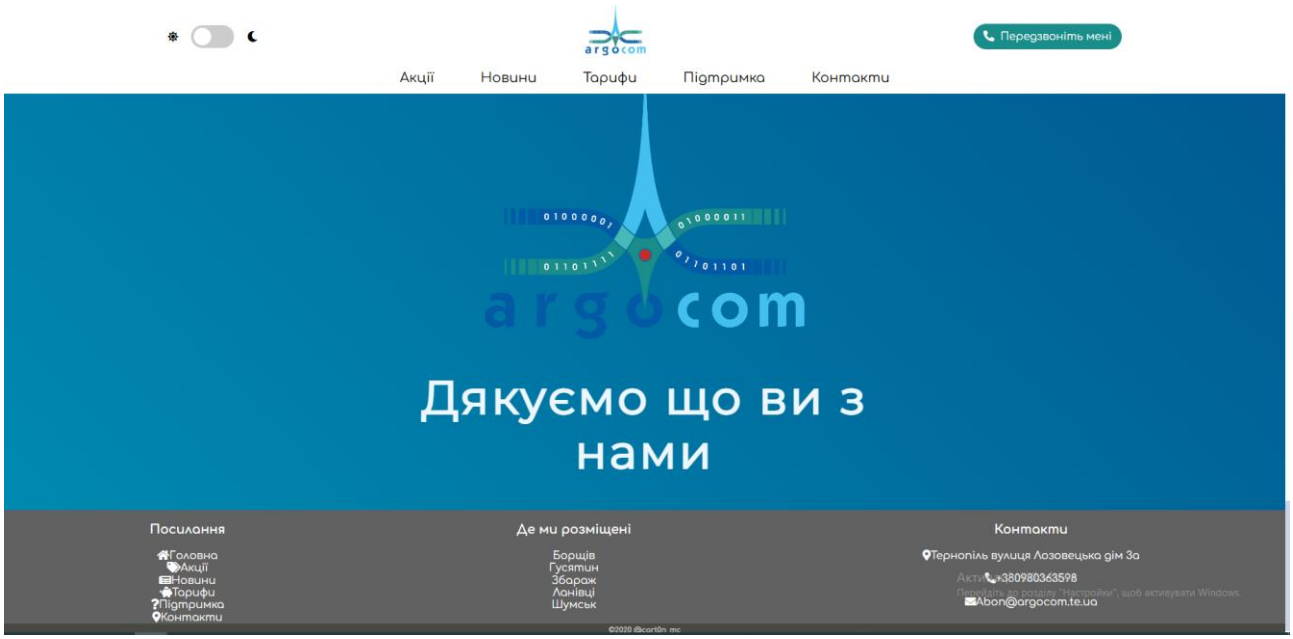

Рисунок 2.7 – Футер на головній сторінці сайту

Після завантаження початкової сторінки сайту користувачу пропонується перейти на сторінку з акціями і тарифами. Перейшовши до розділу «Акції» можна побачити список з трьох блоків, на яких розміщені новини про актуальні спеціальні пропозиції (див. рис. 2.8). Інформація записана в ці блоки витягується з відповідної бази даних і оновлюється кожного разу, коли в базі з’являються нові записи.

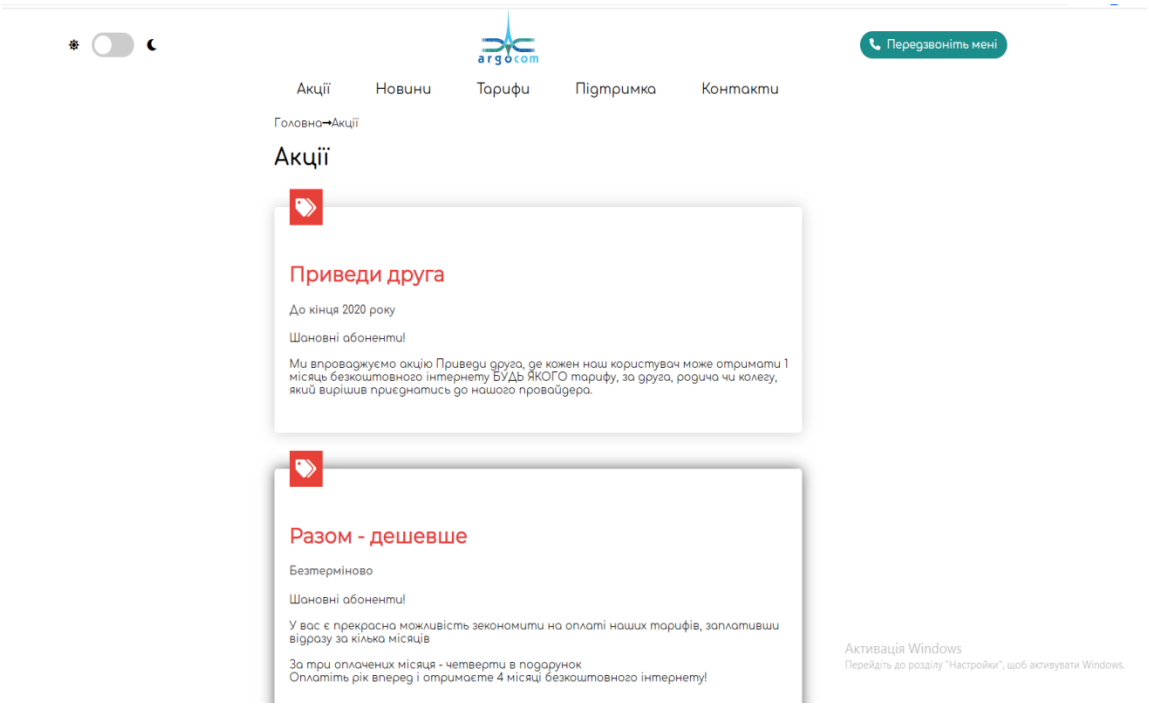

Рисунок 2.8 – Розділ «Акції»

По такому самому принципу працює розділ «Новини» (див. рис. 2.9).

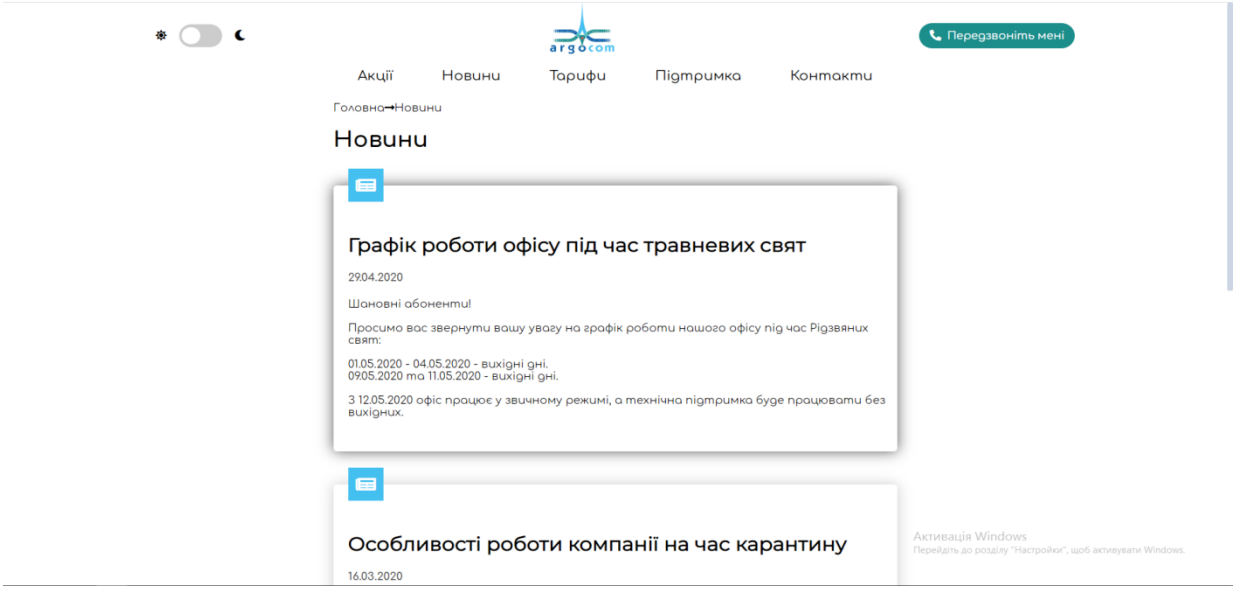

Рисунок 2.9 – Розділ «Новини»

Перейшовши на сторінку «Тарифи», можна побачити картки з тарифами для кожного району, який обслуговується провайдером, навівши курсор на тариф, який обрав користувач з’являється фонова фотографія і кнопка «Зробити замовлення» (див. рис. 2.10), натиснувши на яку, можна перейти в нижню

частину сторінки, де є можливість відправити дані з форми введення на сервер (див. рис. 2.11).

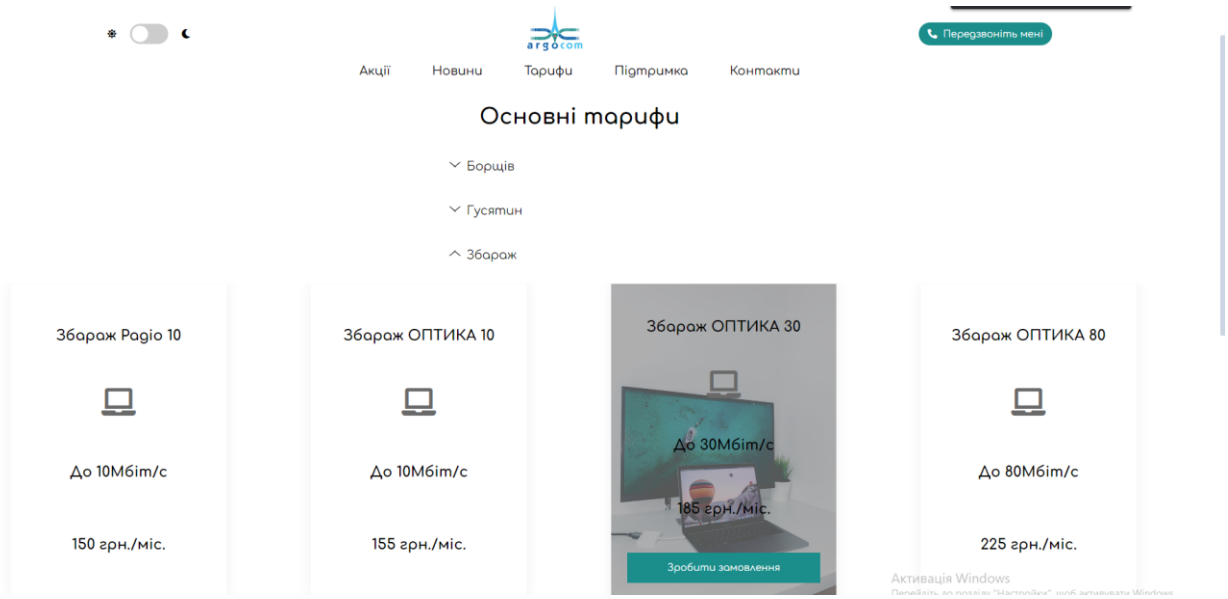

Рисунок 2.10 - Розділ «Тарифи»

Рисунок 2.11 – Форма оформлення тарифу

В розділі «Підтримка», крім відповідей на популярні запитання, присутня велика анімована кнопка «Передзвоніть мені», яка виділяється пульсацією.

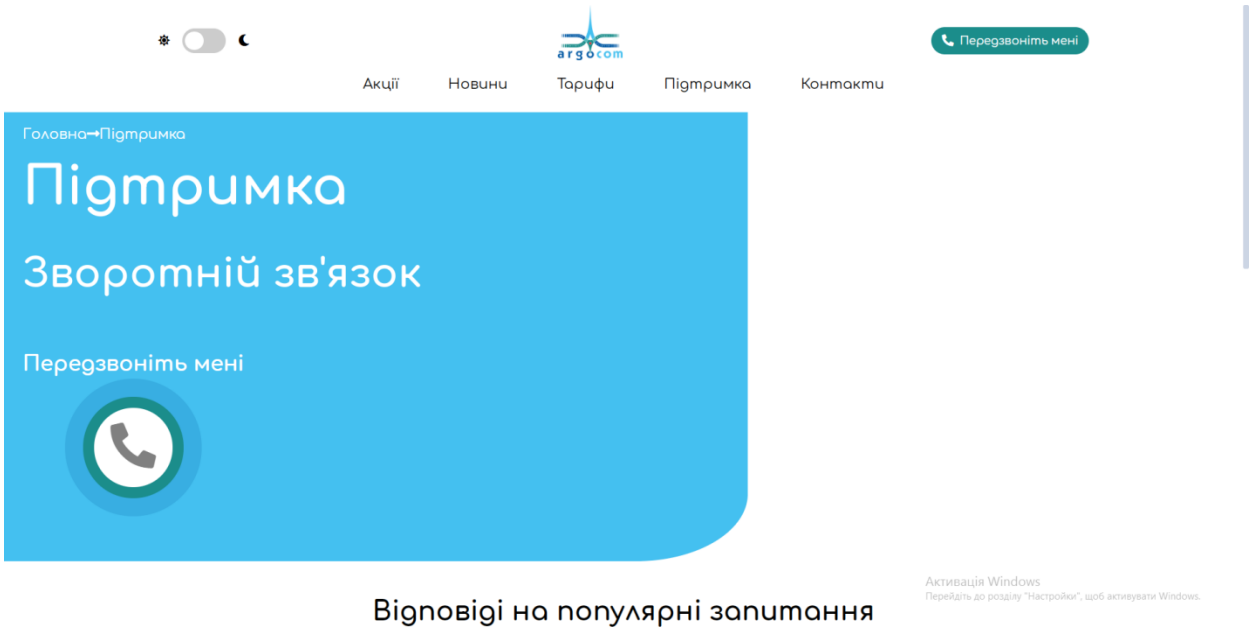

Рисунок 2.12 – Розділ «Підтримка»

Користувач може натиснути на неї, або на кнопку з такою ж назвою в шапці сайту, де вона присутня на кожній сторінці, і викликати модальне вікно, в якому він може внести свої особисті дані для запису в чергу телефонних дзвінків технічної підтримки, які підуть в базу даних.

Для того, щоб сайт був адаптивний, потрібно перш за все переконатися, що в HTML код вписаний тег <meta name="viewport" content="width=device-width">. Значення width=device-width означає, що ширина сторінки буде відповідати ширині екрана пристрою, а якщо його не задати, то значення встановлюються за замовчуванням (в мобільному Safari = 980px, Opera = 850px, Android WebKit = 800px, IE = 974px.). Для адаптивних сайтів рекомендується вказувати саме константу device-width.

Після цього перевірку адаптивності можна здійснити майже в будь якому браузері в режимі інструментів розробника. Для цього потрібно натиснути на комбінацію клавіш Ctrl + Shift + I і натиснути на значок що зображає мобільний телефон (див. рисунок 2.13) [8].

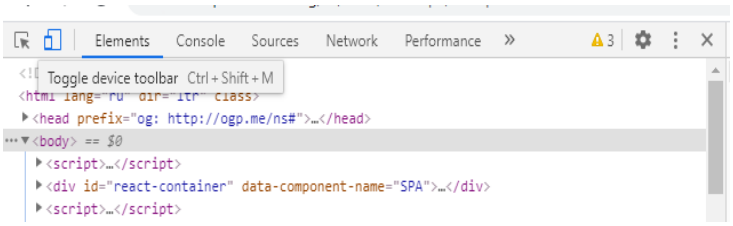

Рисунок 2.13 – Режим мобільного відображення сайту

Окрім перевірки адаптивності вручну можна використовувати сервіс під назвою Google Mobile Friendly, в який можна завантажити код сайту або посилання на нього і через деякий час сайт видасть результат перевірки на адаптивність (див. рис. 2.14).

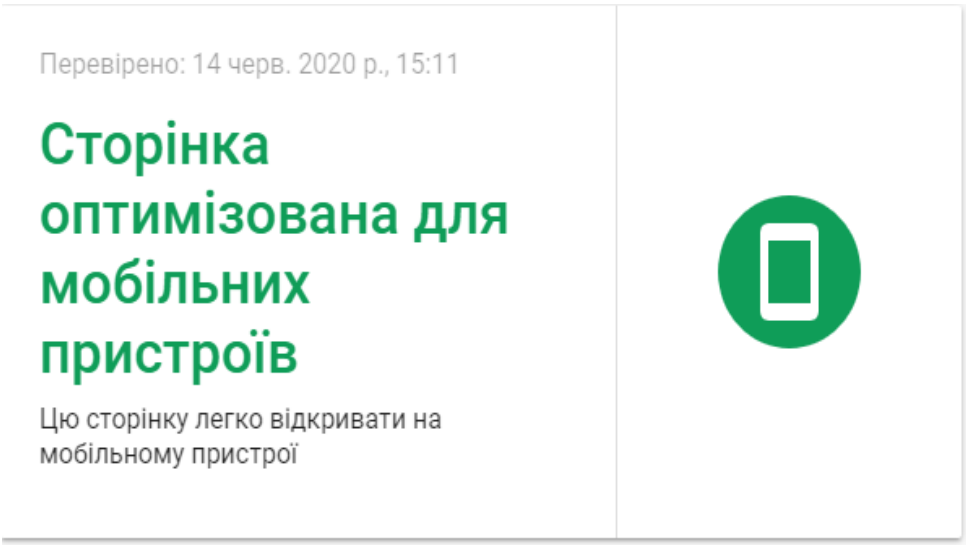

Рисунок 2.14 – Результат перевірки

Для забезпечення кросбраузерності перш за все до сайту підключають Normalize.css. Це невеликий CSS-файл, який забезпечує для HTML-елементів кращу кросбраузерність в стилях за замовчуванням.

Він займається перш за все збереженням корисних налаштувань браузера, та їх видаленням, нормалізуванням стилів для широкого кола HTML-елементів, коригування помилок і основних невідповідностей браузера і удосконалюванням юзабіліті непомітними покращеннями. Він підтримує широкий діапазон браузерів (в тому числі мобільних) і включає в себе CSS, який нормалізує HTML5-елементи, типографіку, списки, вбудований контент,

форми і таблиці. Підключити його можна за допомогою тега <link rel="stylesheet"href= "https://cdnjs.cloudflare.com/ajax/libs/normalize/8.0.1 /normalize.min.css">.

Щоб дізнатись, які стилі потребують веб-префіксів в різних браузерах можна використати ресурс під назвою Caniuse, який надає можливість вписати стиль, який цікавить розробника і дізнатись про характерні особливості його відображення в різних браузерах (див. рис. 2.15)

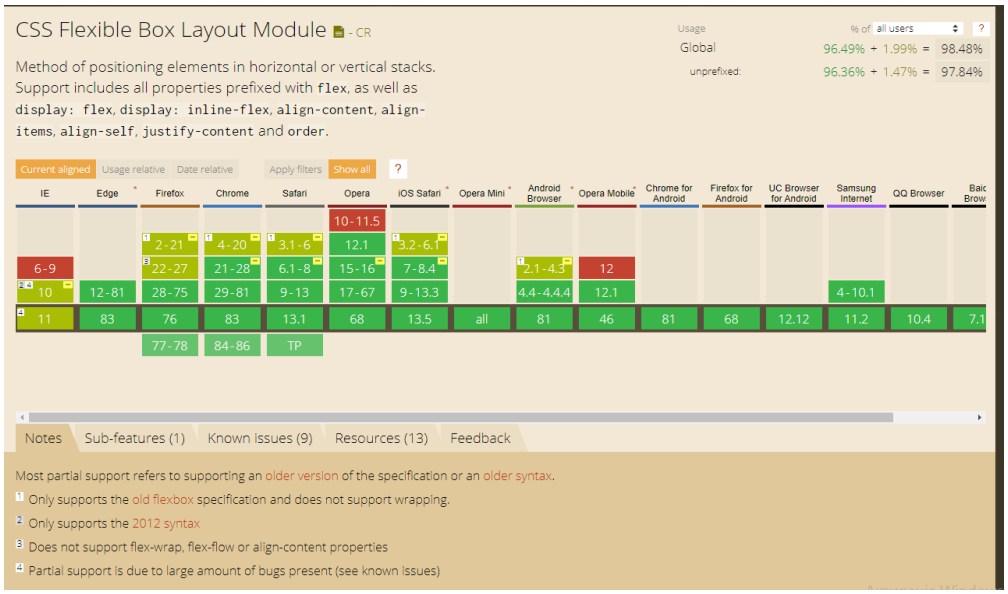

Рисунок 2.15 – Відображення flexbox на сайті Caniuse.com

Тестування відображення сайту в різних браузерах відбувається за рахунок відкриття його в декількох основних браузерах. Якщо результат в різних браузерах різний, то потрібно задати стилям веб-префікси.

Префікси постачальників CSS, також іноді називаються або префіксом браузера CSS, є способом для виробників браузерів додавати підтримку нових функцій CSS, перш ніж ці функції повністю будуть підтримуватися у всіх браузерах.

Це може бути зроблено під час певного періоду тестування та експерименту, коли виробник браузера визначає, як саме ці нові функції CSS будуть реалізовані. [9].

Префікси браузера CSS, які можна використовувати (кожен із яких є специфічним для іншого веб-переглядача), є:

- Android: -webkit- .
- Chrome: -webkit- .
- Firefox: -moz-.
- Internet Explorer:-ms-.
- iOS: -webkit-.
- Опера: -o.
- Safari: -webkit-.

Для прикладу, шапка сайту в браузері Internet Explorer відображалась неправильно (див. рис. 2.16), тому для елементів, які відповідають за його позиціювання потрібно було задати певні префікси.

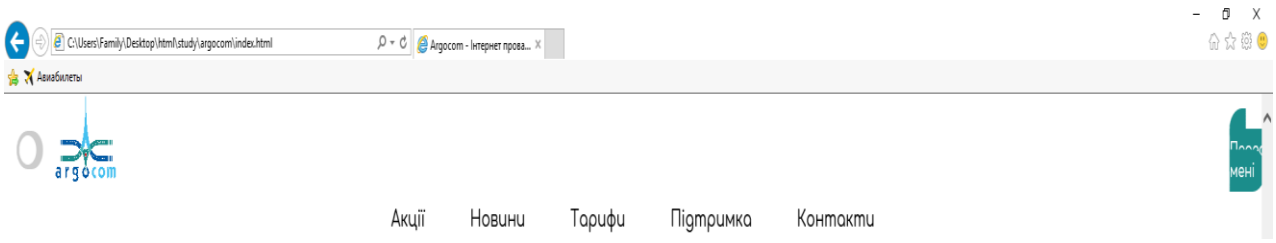

Рисунок 2.16 – Меню яке невдало відобразилось

Для того, щоб спростити роботу з префіксами, можна використовувати різні сервіси, які в онлайн режимі конвертують звичайний код, в код з префіксами.

Одним з таких сервісів є сайт «Автопрефіксер CSS онлайн» (див. рис. 2.17), в який вставляється необхідний код. Після того додається код, який сформував сайт в окремий файл із стилями і в результаті отримуємо потрібний результат (див. рис. 2.18).

```
.top-header{
    height: 100px;
    display: flex;
    justify-content: space-around;
    align-items: center;

}
```

```
CSS
/*
* Prefixed by https://autoprefixer.github.io
* PostCSS: v7.0.29,
* Autoprefixer: v9.7.6
* Browsers: last 4 version
*/

.top-header{
    height: 100px;
    display: -webkit-box;
    display: -ms-flexbox;
    display: flex;
    -ms-flex-pack: distribute;
        justify-content: space-around;
    -webkit-box-align: center;
        -ms-flex-align: center;
            align-items: center;

}
```

Рисунок 2.17 - Автопрефіксер CSS онлайн

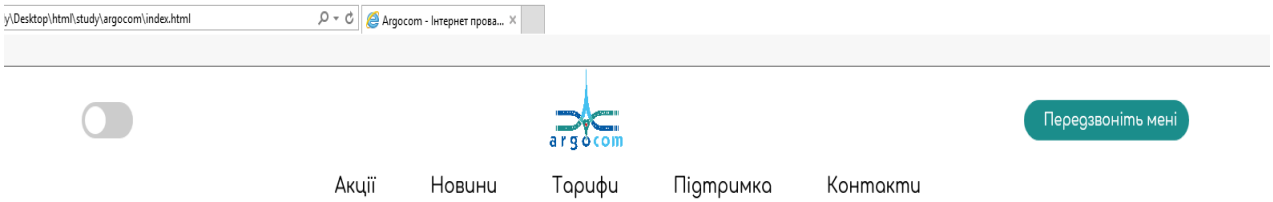

Рисунок 2.18 – Меню, яке вдало відобразилось

Після використання веб-префіксів, сайт остаточно перевіряється в популярних браузерах, при цьому звертається увага на однаковість відображення веб-сторінок.

2.3 Встановлення React і Express

Для роботи з React можна користуватись звичайним файлом з розширенням .html, але цей варіант потребує самостійного налаштування Webpack і Babel. Для того щоб пропустити ці речі можна використати Create React App.

Однак даний метод, не підходить для додатків, які складаються не тільки з фронтенда і потребують підключення до сервера. Для цього знадобиться

Express, який добре взаємодіє з React і дозволить йому отримувати дані з сервера.

Перед початком роботи з Create React App потрібно перевірити чи встановлена на комп'ютері платформа Node версії 8.10 , або вище, якщо ні то потрібно завантажити її з офіційного сайту (див. рис. 2.19).

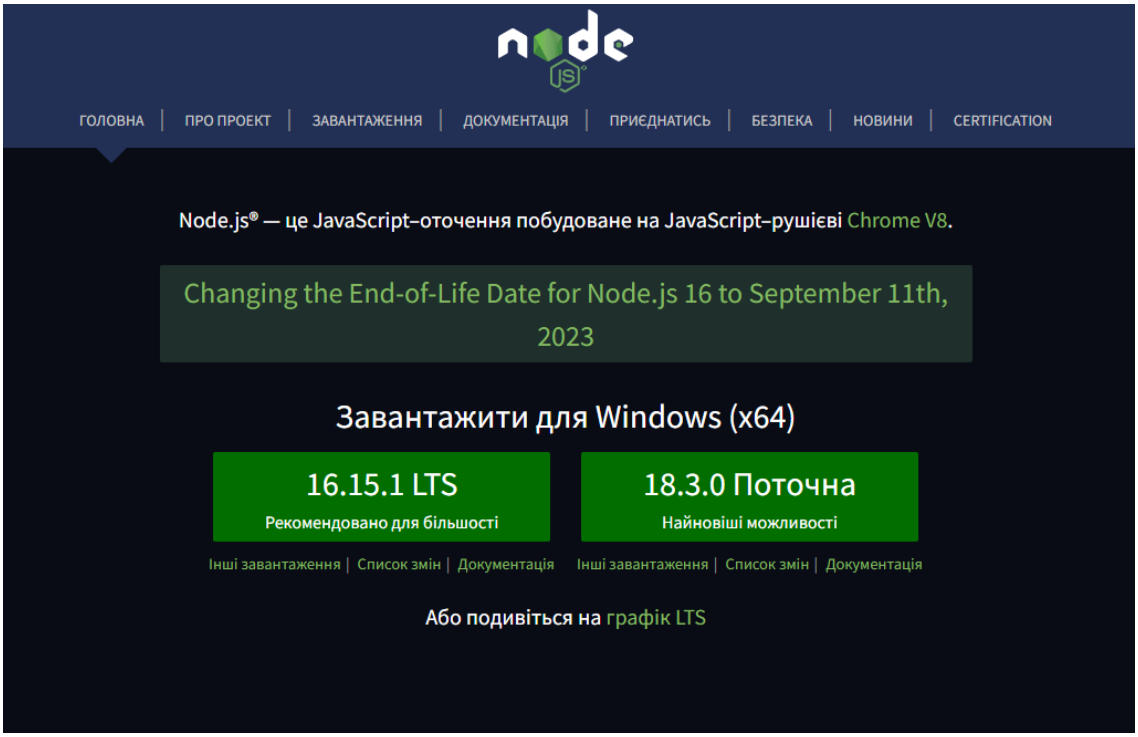

Рисунок 2.19 – Сторінка завантаження Node.js

Node.js містить в собі менеджер пакунків npm, потрібної версії для подальшої роботи з Create React App.

Після встановлення необхідного програмного забезпечення, потрібно зайти в командний рядок і зайти в папку, в якій потрібно розмістити додаток за допомогою команди cd. Після цього необхідно прописати команду npx create-react-app my-app. По завершенні процесу завантаження, який триває певний час потрібно зайти в створену папку my-app і прописати команду npm start, яка запустить додаток на http://localhost:3000/ і покаже створену автоматично сторінку вітання (див. рис. 2.20).

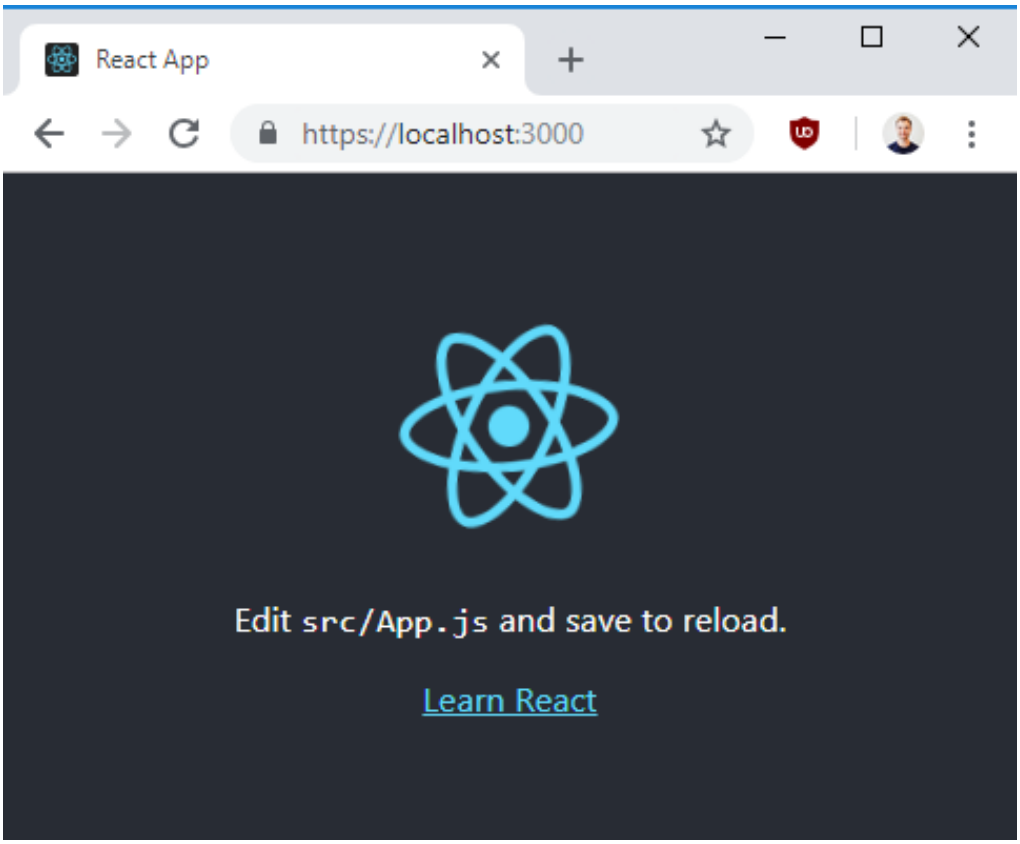

Рисунок 2.20 – Початкова сторінка React

Після цього файл App.js в папці src можна використовувати як основну сторінку сайту вбудовуючи туди написаний HTML синтаксис. Для створення інших сторінок веб-сайту потрібно додавати їх верстку в окремі файли проекту імпортуючи їх в App.js, або іншу основну сторінку і додаваючи посилання на них за допомогою модуля react-router-dom.

Для створення Express додатку можна скористатись утилітою express-generator, яка встановлюється за допомогою команди npm install -g express-generator. Для створення додатка необхідно використати команду express react-backend. Після того, як ця команда створила папку react-backend потрібно зайти в неї командою cd і виконати команду npm install. На наступному кроці необхідно запустити додаток прописавши npm start, але якщо перед цим вже було запущено реакт додаток, то отримаємо повідомлення, що localhost:3000/ зайнятий, оскільки його використовують обидва додатки, тому потрібно зайти в папку Express додатку, далі в папу bin і в файлі www.txt змінити номер порта з 3000

Тепер, якщо перейти на змінений порт, можна буде побачити початкову сторінку Express додатку (див. рис. 2.21).

Рисунок 2.21 – Початкова сторінка Express

Для безпеки Express застосовують модуль helmet, що допомагає захистити програму від деяких широко відомих веб-вразливостей шляхом відповідного налаштування заголовків HTTP.

Helmet, по суті, являє собою набір з дев'яти дрібніших функцій проміжної обробки, що забезпечують налаштування заголовків HTTP, пов'язану із захистом:

- csp задає заголовок Content-Security-Policy для запобігання атакам міжсайтового скриптингу та інших міжсайтових втручань;
- hidePoweredBy видаляє заголовок X-Powered-By. Цей заголовок можна використовувати, щоб виявити, що програма працює від Express, що дозволяє хакерам проводити точну атаку. Звичайно, заголовок X-Powered-By - не єдиний спосіб ідентифікувати запущену програму Express, але це, мабуть, найпоширеніший і простий;
- hsts визначає заголовок Strict-Transport-Security, що примусово активує захист з'єднань з сервером (за протоколом HTTP з використанням SSL/TLS);
- ieNoOpen визначає заголовок X-Download-Options для IE8+;

- noCache задає заголовок Cache-Control та заголовки Pragma для відключення кешування на стороні клієнта;
- noSniff задає заголовок X-Content-Type-Options для захисту браузерів від прослуховування (сніффінгу) MIME відповідей з відмінним від оголошеного типом вмісту (content-type);
- frameguard задає заголовок X-Frame-Options для захисту від клікджекінгу;
- xssFilter визначає заголовок X-XSS-Protection для активації фільтра XSS (фільтра міжсайтового скриптингу) у більшості сучасних веб-браузерів.

Встановіть Helmet як звичайний модуль(див. рисунок 2.22).

```
PS C:\Users\User\Desktop\snykaacomy> npm install --save helmet

added 1 package, and audited 86 packages in 2s

found 0 vulnerabilities
```

Рисунок 2.22– Встановлення Helmet

У Express.js 4 є два модулі сесії cookie:

- express.session;
- express.cookieSession/

Модуль експрес-сесії зберігає ідентифікатор сеансу в файлі cookie та дані сеансу на сервері. Сесія cookie зберігає всі дані сеансу в файлі cookie.

Загалом, cookie-сесія більш ефективна. Проте, якщо дані сеансу, які потрібно зберігати, є складними і, ймовірно, перевищують 4096 байт на файл cookie, використовуйте експрес-сесію. Іншою причиною використання експрес-сесії є те, що потрібно зберегти дані cookie невидимими для клієнта.

Крім того, ви повинні встановити параметри безпеки файлів cookie, а саме:

- secure;
- httpOnly;

- domain;
- path;
- expires;

Якщо для параметра «secure» встановлено значення «true», браузер надсилатиме файли cookie лише через HTTPS. Якщо для параметра «httpOnly» встановлено значення «true», файл cookie надсилатиметься не через клієнтський JS, а через HTTP(S). Значення «домен» вказує на домен файлу cookie. Якщо домен cookie відповідає домену сервера, «шлях» використовується для вказівки шляху до файлу cookie. Якщо шлях cookie відповідає шляху запиту, файл cookie буде надіслано в запиті. Нарешті, як випливає з самої назви, значення «expires» означає час, коли закінчиться термін дії файлів cookie.

Для того щоб зв’язати додаток React і Express, потрібно в папці, де розміщений React-додаток в файлі package.json після секції "scripts" добавити команду "proxy” номер порта, або, в подальших етапах розробки сайту, адресу посилання на адресу сервера з хостинга.[10]

2.4 Зв’язок користувача з сервером і базою даних

У даному MERN проєкті можливості зв’язку веб-сторінок із базою даних відбувалось не тільки за для відображення і оновлення актуальної інформації в розділах «Новини», «Акції», і «Тарифи», а й для зв’язку користувача з адміністрацією сайту за допомогою форм модального вікна (див. рис. 2.23).

Модальне вікно відображається після натиснення на кнопку «Передзвоніть мені» в шапці сайту (див. рис. 2.24). Після натискання на неї з’являється вікно імпортоване з компонента Modal, в якому створюється Portal з всіма необхідними формами, в які можна вписати свої дані (див. рис. 2.25).

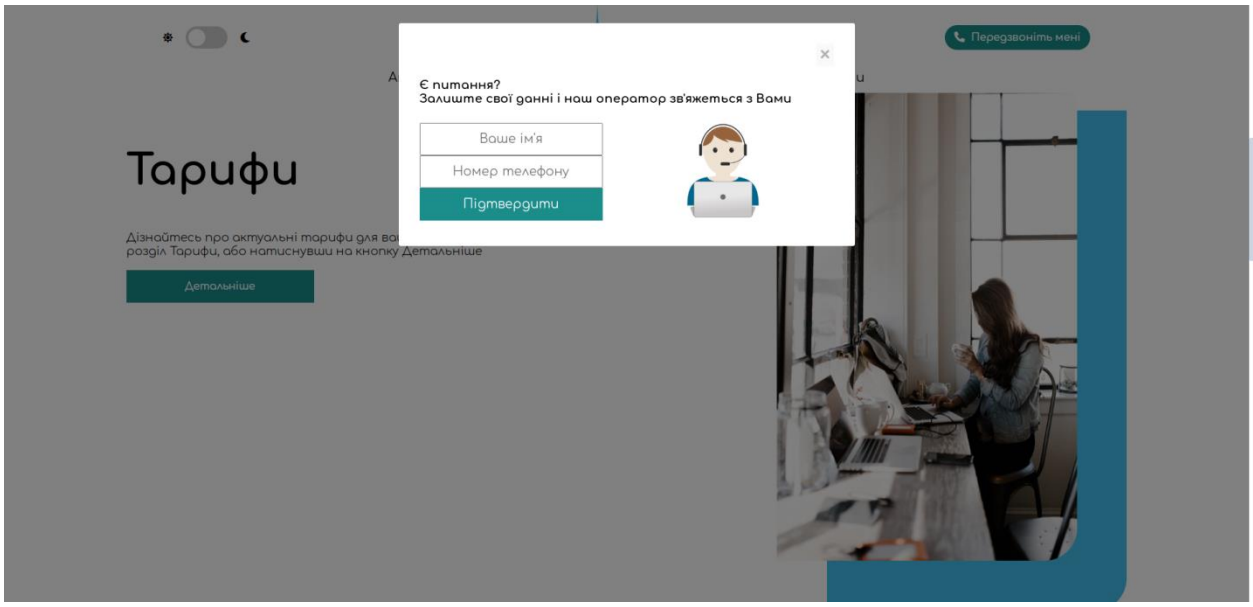

Рисунок 2.23 – Модальне вікно зв’язку

Рисунок 2.24 – Кнопка для відкриття модального вікна

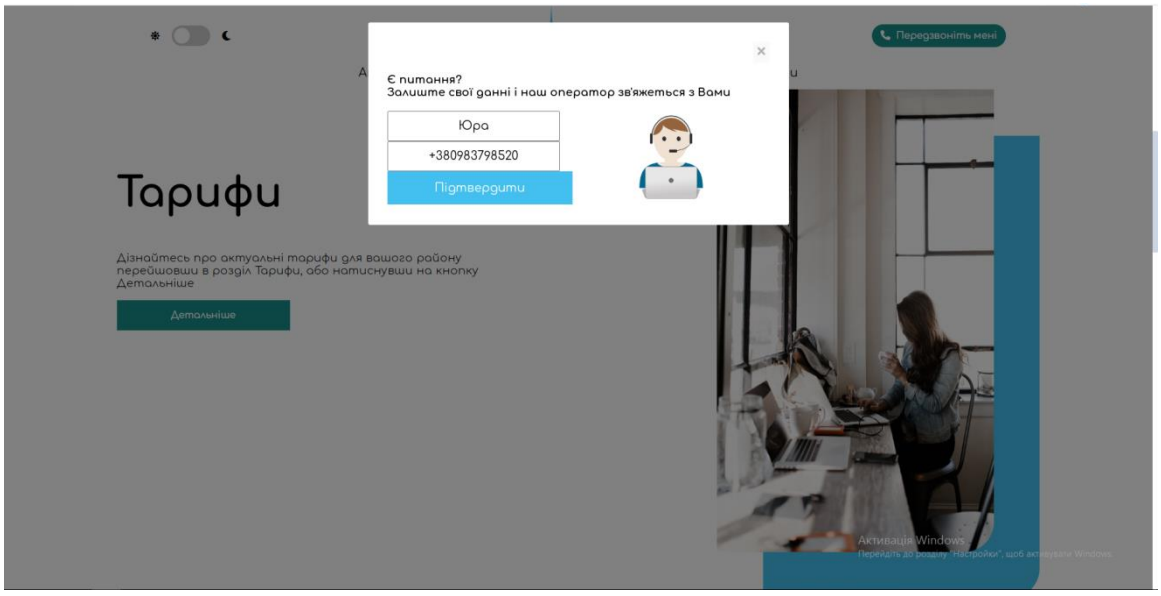

Рисунок 2.25 – Заповнена форма

App.js – це основний файл проекту, який відображає головне меню і відповідає за зв’язок з всіма компонентами і сторінками. В правому краю розміщена кнопка «Передзвоніть мені», при натисканні на якій, активується функція Onclick, яка містить параметр toggle, який відображає модальне вікно імпортоване на початку коду з файлу Modal.js

В файлі Modal.js розміщене саме модальне вікно з всіма елементами присутніми в ньому. До повернення модального вікна розміщена функція, яка за допомогою fetch зв’язує React проект з файлом index.js Express, надсилає дані в базу даних, а також при успішному надсиланні даних відправляє вікно з повідомленням.

index.js – це один з файлів в Express проекті, який і забезпечує сайт бекенд частиною, розміщений в папці Routes. Він зв’язує React і базу даних, що дозволяє виводити дані вписані користувачем в модальному вікні в базу даних.

Дані з бази даних MongoDB розміщені в MongoDB Atlas — це багатохмарна служба баз даних, що спрощує розгортання та керування базами даних, пропонуючи універсальність, необхідну для створення стійких і продуктивних додатків у хмарних постачальників на ваш вибір на AWS, Azure чи Google Cloud.

Важливими перевагами MongoDB Atlas є:

- Глобальні кластери для додатків світового класу. Підтримка понад 60 хмарних регіонів у AWS, Azure та GCP.
- Вбудовані засоби керування безпекою та функції, які відповідають вашим існуючим протоколам і стандартам відповідності.
- Інтегровані інструменти для маніпулювання, візуалізації та аналізу ваших даних. Виконувати код у режимі реального часу у відповідь на зміни даних.

Завдяки MongoDB Atlas ваші дані захищені попередньо налаштованими функціями безпеки для аутентифікації, авторизації, шифрування тощо.

Спеціальні кластери MongoDB Atlas розгорнуті в унікальній віртуальній приватній хмарі (VPC) із виділеними брандмауерами. Доступ має надаватися за допомогою списку доступу IP або VPC Peering.

За потреби можна налаштувати складні правила доступу на основі ролей, щоб контролювати, які користувачі та команди можуть отримати доступ, маніпулювати та видаляти дані у ваших базах даних.

Весь мережевий трафік шифрується за допомогою безпеки транспортного рівня (TLS), з можливістю налаштування мінімальної версії протоколу TLS. Шифрування даних у стані спокою автоматизовано за допомогою зашифрованих томів зберігання. Atlas вимагає підключення TLS для всіх кластерів Atlas . Після липня 2020 року Atlas увімкнув протокол Transport Layer Security (TLS) версії 1.2 за замовчуванням для всіх нових кластерів Atlas незалежно від версії MongoDB.[11]

Для того щоб створити кластер з базою даних потрібно зареєструвати обліковий запис на MongoDB, натиснути на “|Create Project” і дати йому назву(див рисунок 2.26). Після цього можна створити базу даних(див. рисунок 2.27)

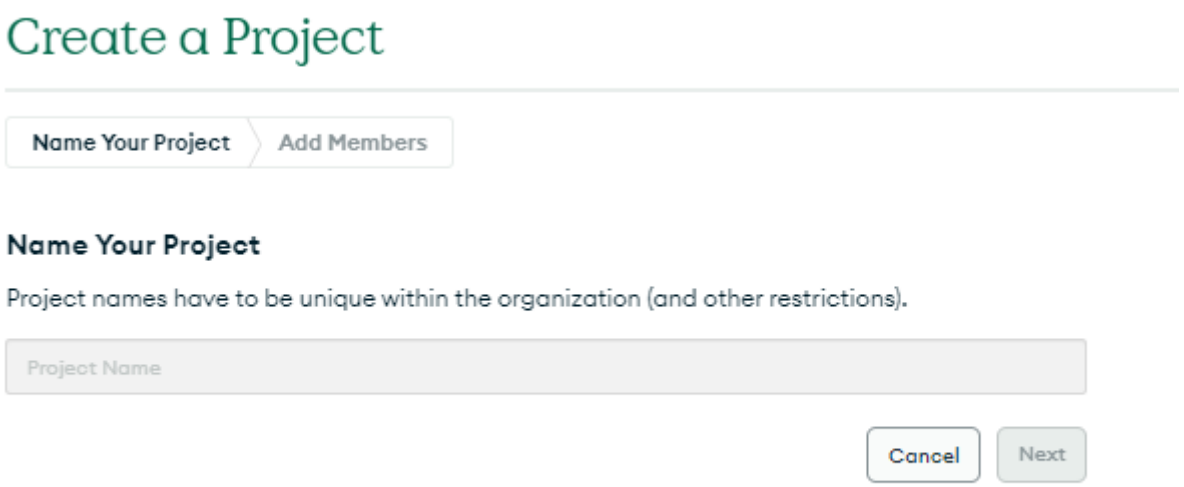

Рисунок 2.26 - Створення проекту

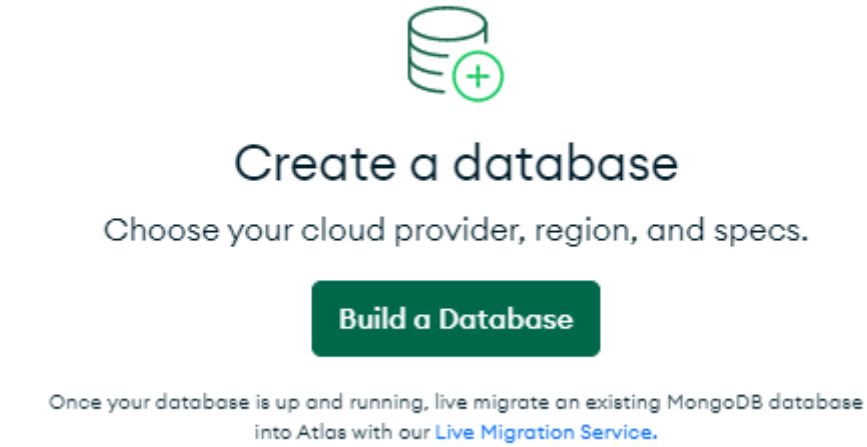

Рисунок 2.27 – Створення бази даних

Пропонується вибрати компанію, яка буде зберігати в хмарі ваші дані і регіон їх розміщення(див. рисунок 2.28)

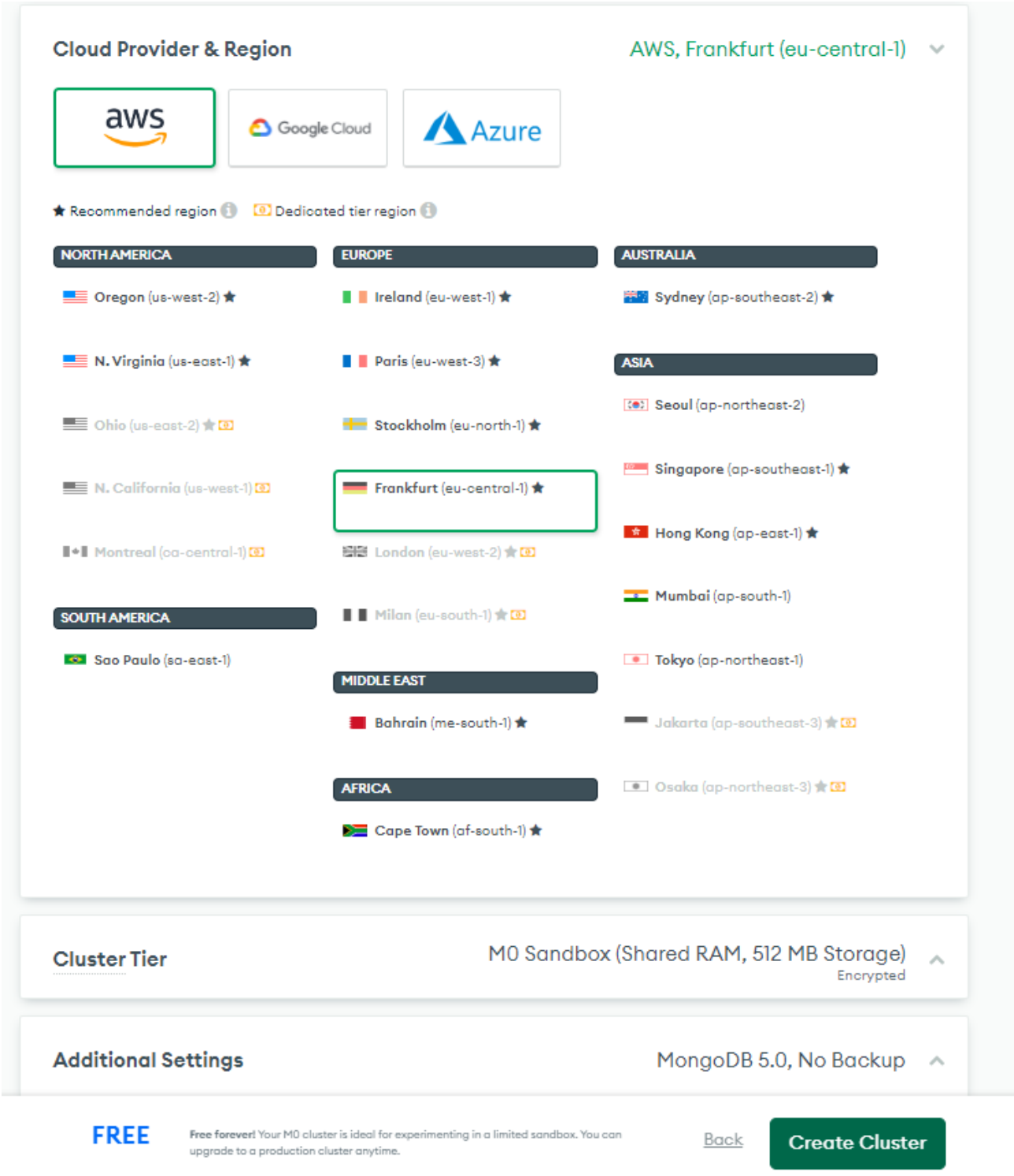

Рисунок 2.28– Хмарний постачальник і регіон

Всі ці хмари входять до так званої «великої трійки» хмарних платформ. У своїй основі AWS, Microsoft Azure і Google Cloud Platform пропонують майже однакові базові можливості щодо гнучких обчислень, зберігання даних і мереж. Усі вони мають спільні елементи загальнодоступної хмари: самообслуговування та миттєве надання, автомасштабування, а також функції

безпеки, відповідності та керування ідентифікацією. Тому для невеликих проектів різниця між ними зовсім незначна. По регіону розміщення даних рекомендують вибирати той, що ближчий до вас.

Далі потрібно додати користувача який зможе користуватись базою даних і IP-адресу до списку IP-доступу, перш ніж можна буде підключитися до свого кластера. Щоб додати свою IP-адресу до списку IP-доступу натисніть “Add My Current IP Address”(див. рисунок 2.29).

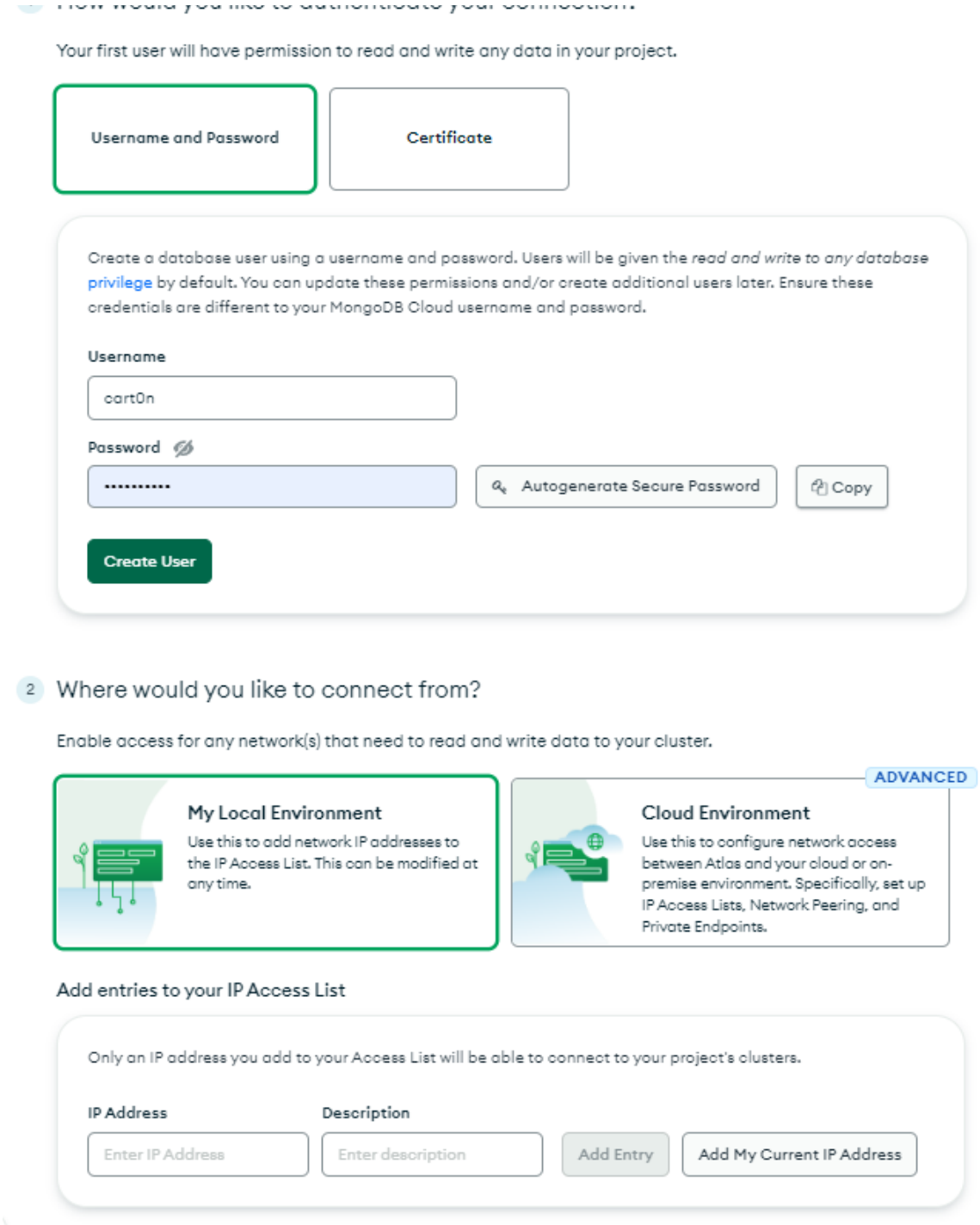

Рисунок 2.29 - Створення користувача і списку IP адрес

Після створення бази даних можна додати свій ключи шифрування, якщо є якісь проблеми з стандартним наскрізним шифруванням, або підключити аудит бази даних для можливості простежити за діями і проблемами з базою в журналі(див рисунок 2.30).

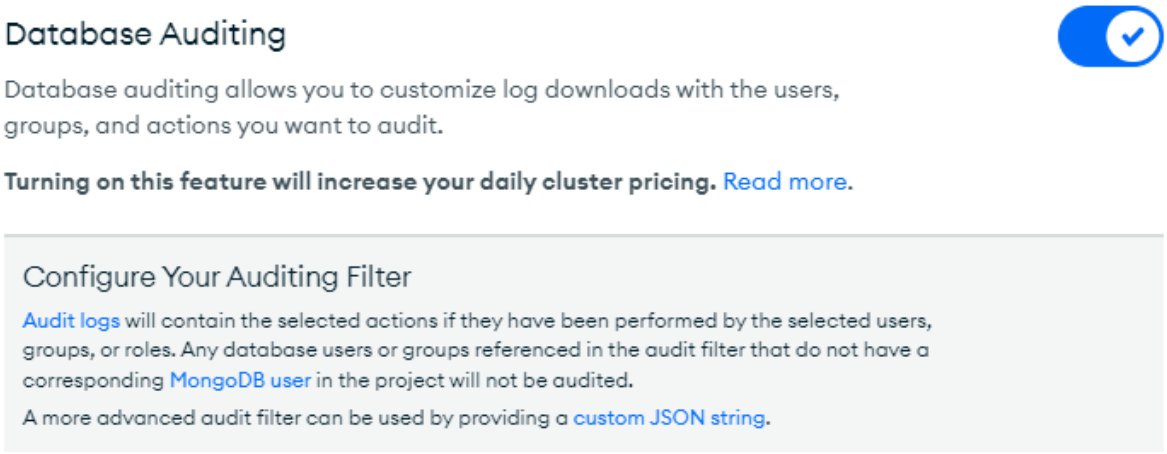

Рисунок 2.30 - Аудит бази даних

Далі потрібно підключитись до свого кластера, це можна зробити різними методами, наприклад за допомогою IDE MongoDB Compass(див рисунок 2.31). Буде використано додання посилання на базу даних(див рисунок 2.32) в код нашої бекенд частини проекту, Express файлу app.js.

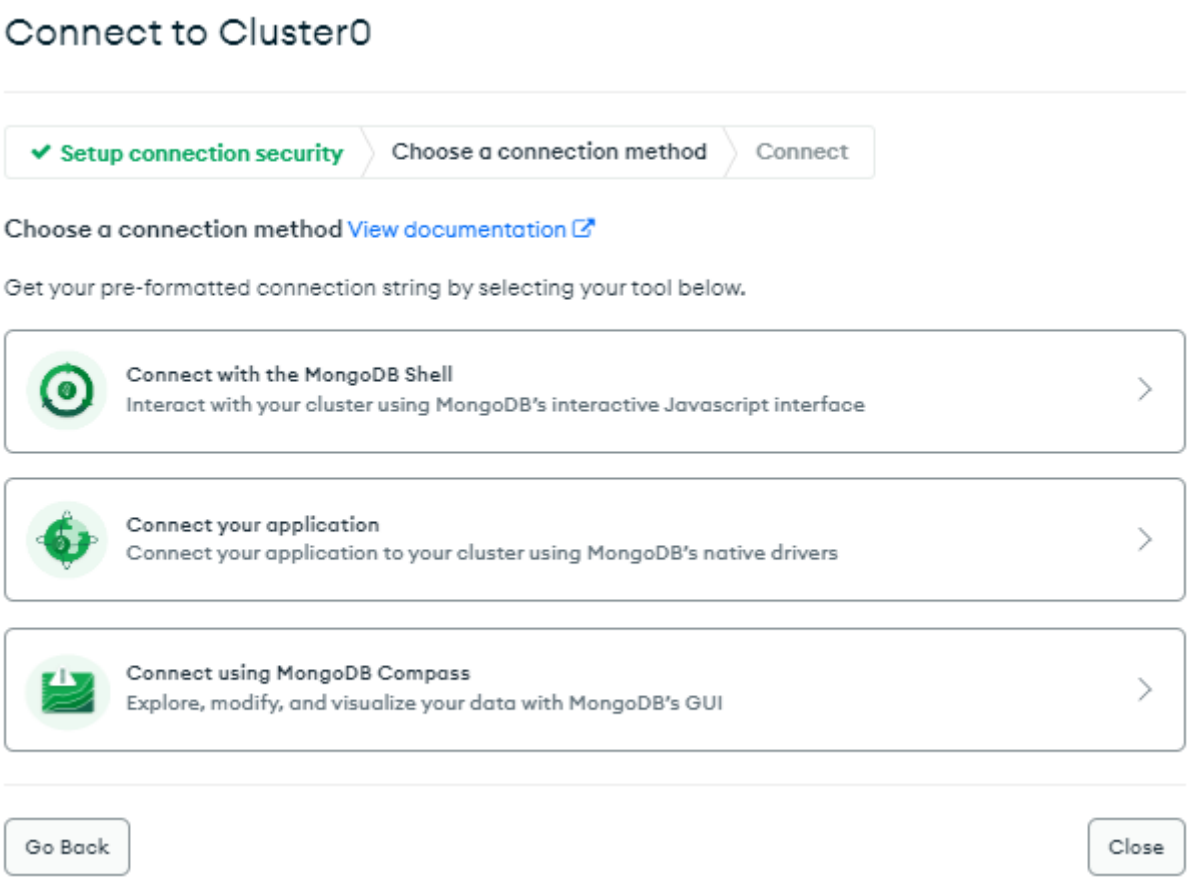

Рисунок 2.31– Методи підключення

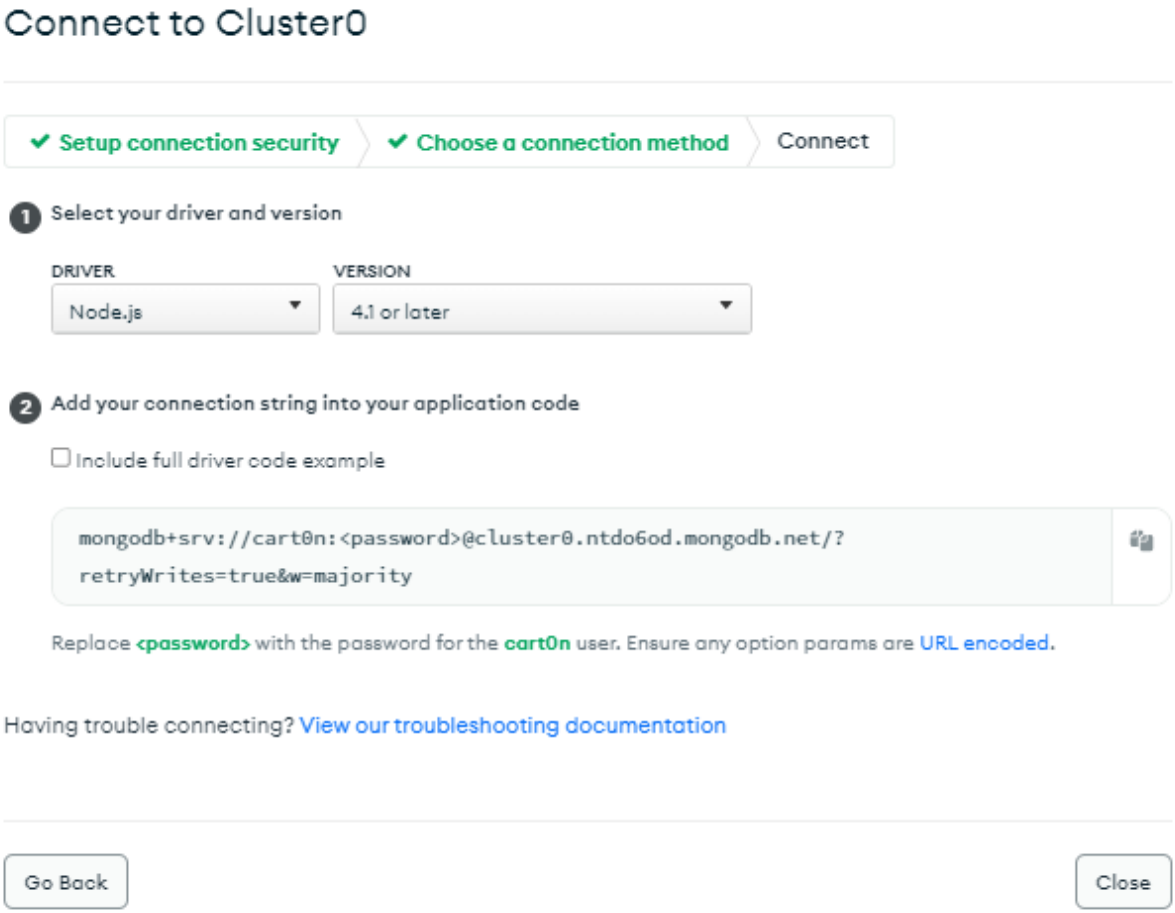

Рисунок 2.32 – Посилання для підключення

3 РОЗРОБКА ТА ТЕСТУВАННЯ ЗАХИЩЕНОСТІ САЙТУ

3.1. Snyk

Snyk — це платформа безпеки для розробників. Інтегруючись безпосередньо в інструменти розробки, робочі процеси та конвеєри автоматизації, Snyk дозволяє командам легко знаходити, розставляти пріоритети та виправляти вразливості безпеки в коді, залежностях, контейнерах та інфраструктурі як код. Підтримка провідних у галузі додатків і аналізу безпеки, Snyk додає знання безпеки в інструменти будь-якого розробника.

База даних Snyk базується на багатьох загальнодоступних джерелах, внесок спільноти розробників і наукових кіл, а також власні інтелектуальні дані команди Snyk Security Research, щоб забезпечити найповнішу інформацію про вразливості на ринку.База Snyk додає нові вразливості набагато швидше, ніж інші рішення, вибираючи кілька джерел, включаючи наші власні дослідження, кураторство та щоденну публікацію.

База даних має надзвичайно низький рівень хибнопозитивних результатів завдяки безперервному та глибокому контролю якості. База даних Snyk надає підібрані вручну дані та збагачені метадані, щоб спрямувати рішення щодо визначення пріоритетів та виправлення.

Використовуючи підхід до безпеки розробників, Snyk інтегрується з провідними інструментами IDE, репозиторією, CI/CD, середовищем виконання, реєстром і проблемами.

Для підключення платформи Snyk до свого проекту через IDE Visual Studio Code потрібно відкрити середовище розробки і натиснути на “Extensions”(див рисунок 3.1), де можна знайти різноманітні розширення.

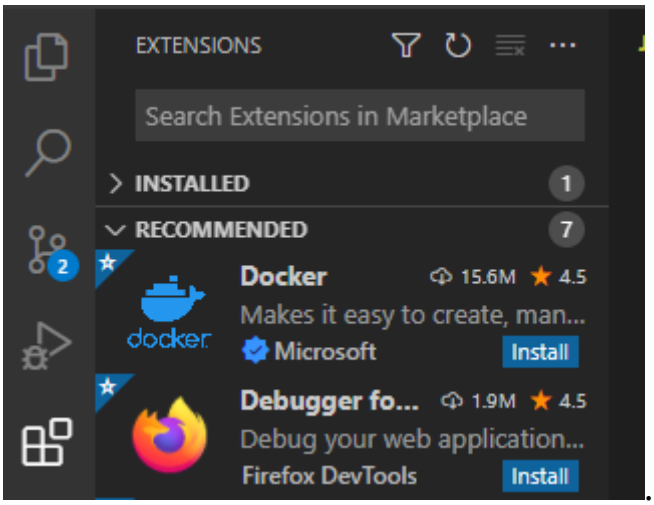

Рисунок 3.1 –Пошук розширень до IDE

Далі потрібно інсталювати розширення(див. рисунок 3.2) і зліва зверху з’явиться панель з логотипом Snyk (див. рисунок 3.3).

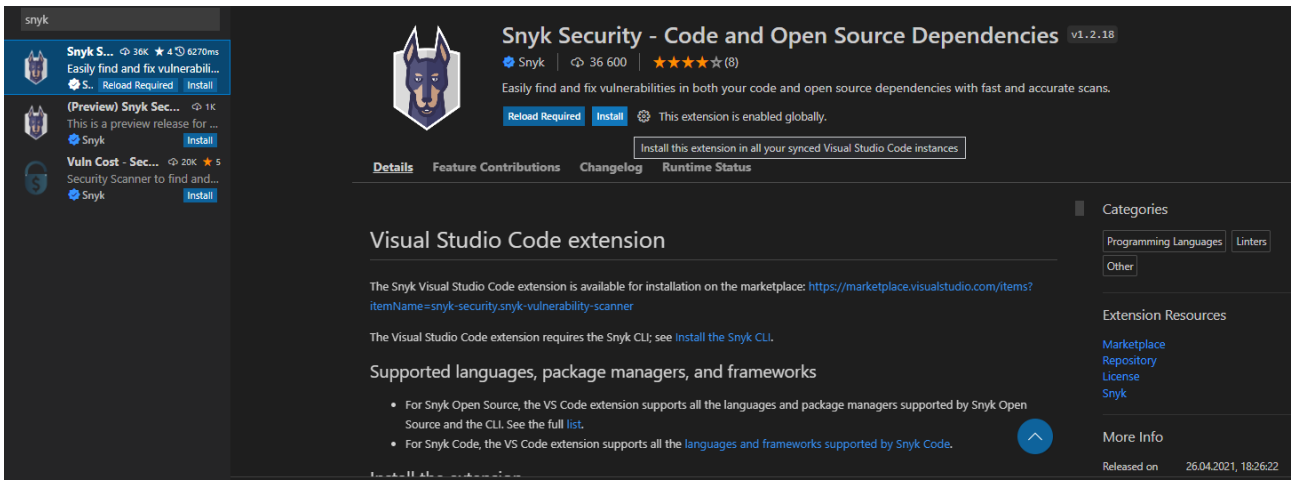

Рисунок 3.2 – Інсталяція розширення

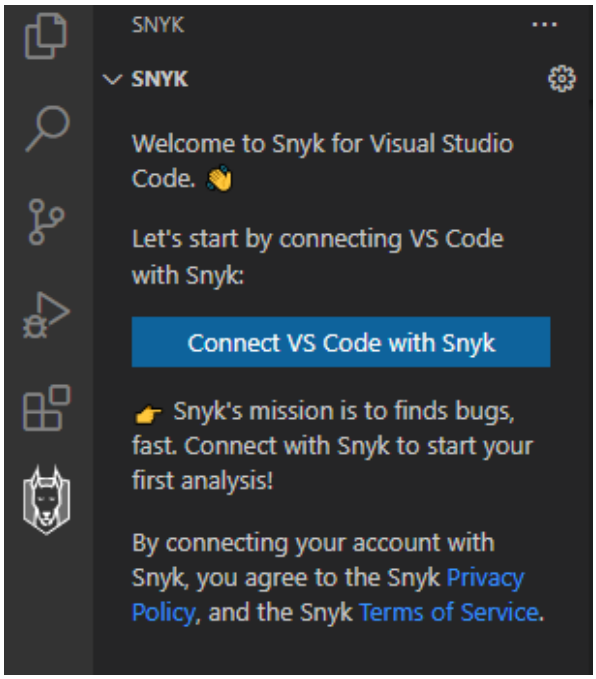

Рисунок 3.3 – Встановлене розширення Snyk

Після цього потрібно буде провести сканування, що розіб'є помилки на три групи "Open Source Security", "Code Security", і " Code Quality".(див. рисунок 3.4). Якщо поставити режим сканування на автоматичний то він буде постійно сканувати проект при кожній його зміні.

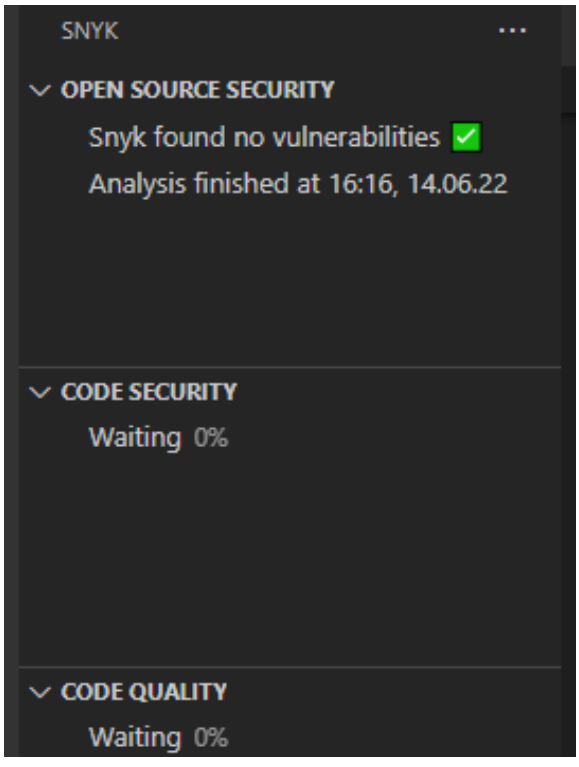

Рисунок 3.4 – Процес сканування

В розділі “Open source security” небепек знайдено не було, часто там показують проблеми зі застарілими версіями певного програмного забезпечення.

В розділі “Code Security” критичних помилок, чи помилок з високим пріоритетом не бачимо, але там є деякі помилки з середнім пріоритетом. Натиснувши на інформацію про ним нам можуть надати посилання на їх опис з рекомендаціями по усуненню. Наприклад, деякі помилки можуть бути швидко виправлені просто підключенням модуля helmet до коду Express.

В розділі “ Code Quality” ми бачимо помилки з низьким пріоритетом(див рисунок 3.5), це помилки пов’язані з конфліктом між HTML і JSX, де JSX вимагає замінити всі властивості class на className.(див рисунок 3.6)

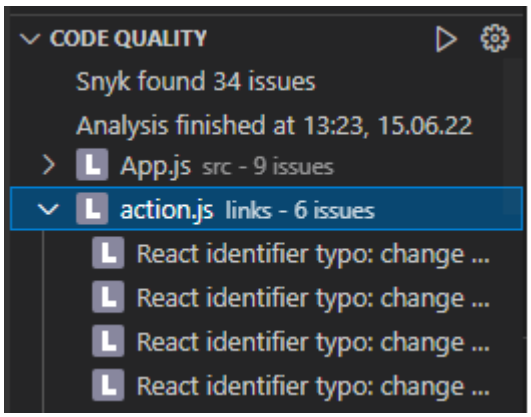

Рисунок 3.5 – Помилки Code Quality

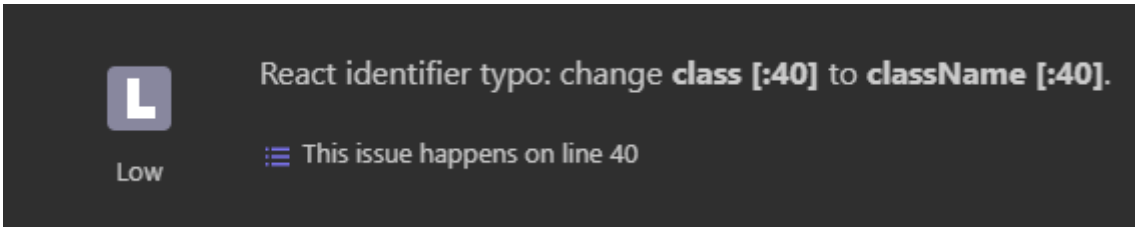

Рисунок 3.6 – Помилка ClassName

3.2 Розміщення сайту на безпечний хостинг

Хостинг — це сервіс з надання простору для розміщення сайту на сервері. Хостери або хостинг-провайдери - це компанії, які продають дані послуги і піклуються про безвідмовну роботу сайту 24 години на добу. [13]

Коли мова йде про сайти розроблені на React + Express, то далеко не всі хостинги підтримують такі проекти, особливо безкоштовні. Особливість полягає в тому, що проект потрібно окремо помістити в мережу і бекенд і фронтенд частину. Розглядаючи варіанти легкого і безпроблемного розміщення проекту на хостинг можна помітити хостинг Heroku. Він був розроблений 2007 року і спочатку підтримував тільки мову Rubi, але зараз розвинувся і часто стає в пригоді розробникам, що працюють з зовсім різними мовами програмування. [14]

Heroku проходить тести на проникнення, оцінки вразливостей та огляди вихідного коду, щоб оцінити безпеку нашої програми, архітектури та реалізації. Оцінки безпеки сторонніх розробників охоплюють усі сфери платформи, включаючи тестування на наявність уразливостей веб-додатків OWASP Top 10 та ізоляцію програм клієнта. Heroku тісно співпрацює із зовнішніми оцінювачами безпеки, щоб перевірити безпеку платформи та програм Heroku та застосувати найкращі методи. Проблеми, виявлені в програмах Heroku, оцінюються ризиками, пріоритети, призначаються відповідальній команді за виправлення, а команда безпеки Heroku переглядає кожен план усунення, щоб забезпечити належне вирішення. Проте про Heroku частіше говорять як про середовище для розміщення сервера, і хоча фронтенд частину можна використовувати на ньому також є і інші цікаві хостинги для лицьової частини проекту. Netlify — це платформа для веб-розробників, яка примножує продуктивність. Об’єднуючи елементи сучасної відокремленої мережі, від локальної розробки до передової логіки периферій , Netlify забезпечує в 10 разів швидший шлях до набагато більш продуктивних, безпечних і

масштабованих веб-сайтів і програм. Даний проект буде розміщати свою бекенд частину на heroku, а фронтенд на netlify.

Перед початком роботи з Heroku потрібно насамперед зареєструватися на даній хмарній PaaS-платформі (див рис. 3.7), яка підтримує ряд мов програмування. Після того, як було підтверджено e-mail адресу, потрібно буде заповнити ще трохи даних в своєму профілі. Потім потрібно налаштувати мультифакторну авторизацію. Мультифакторна автентифікація захищає ваш обліковий запис від поширених загроз, таких як фішингові атаки, підбір облікових даних та захоплення облікового запису. Це додає ще один рівень безпеки до вашого процесу входу, вимагаючи від користувачів ввести два або більше докази — або фактори — на додаток до пароля, щоб підтвердити, що вони ті, за кого себе видають.

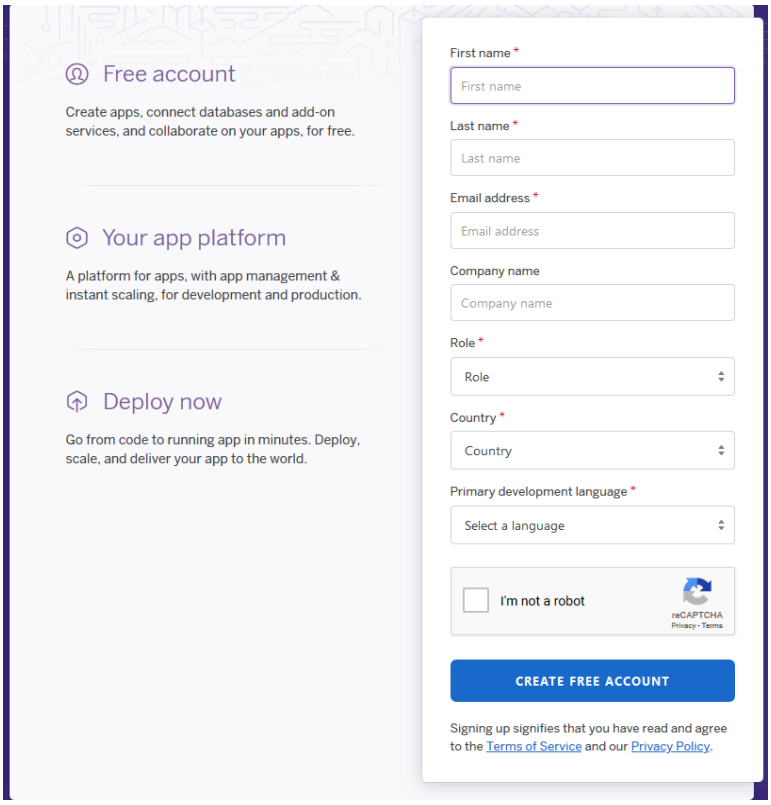

Рисунок 3.7 – Форма реєстрації

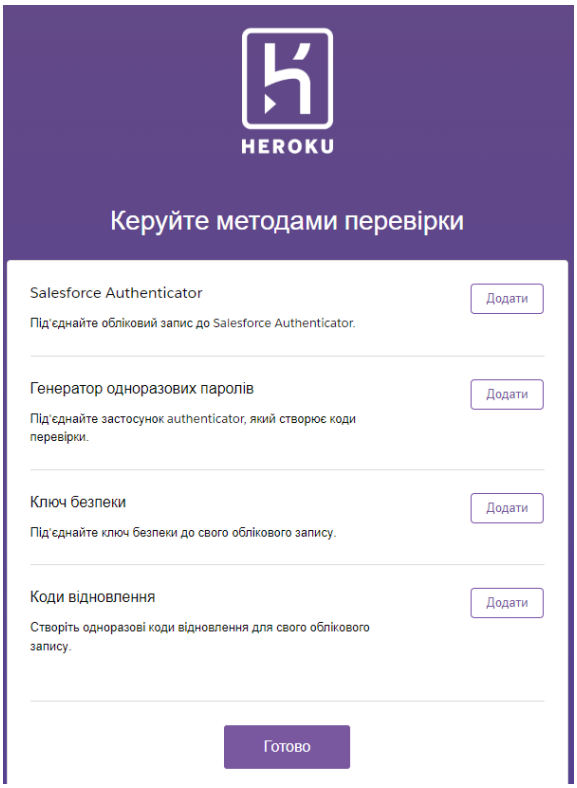

Рисунок 3.8 – Методи багатофакторної авторизації

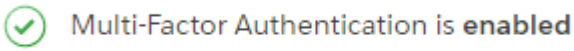

Рисунок 3.9 – Підключена багатофакторна авторизація

Після цього потрібно завантажити програму на свій комп’ютер, вибравши варіант з своєю операційною системою (див. рисунок) і встановити її як звичайну програму.

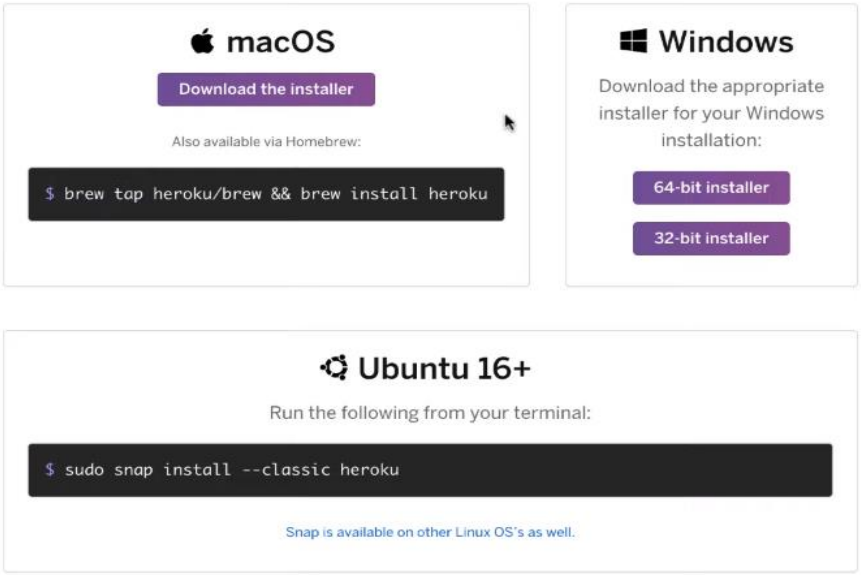

Рисунок 3.10 – Сторінка завантаження

Після завантаження, необхідно зайти в термінал проекту, перевірити правильність встановлення командою «heroku –help», і якщо буде видано команди для роботи з Heroku потрібно написати в термінал «heroku login» (див. рисунок). Далі потрібно натиснути на будь-яку клавішу крім q і в браузері відкриється сторінка з авторизацією для Heroku профілю. Після успішної авторизації, використовуючи дані, вказані при раєстрації, необхідно повернутись до терміналу, в якому буде повідомлено про успішний вхід і прописати команду «heroku creacte».[15]

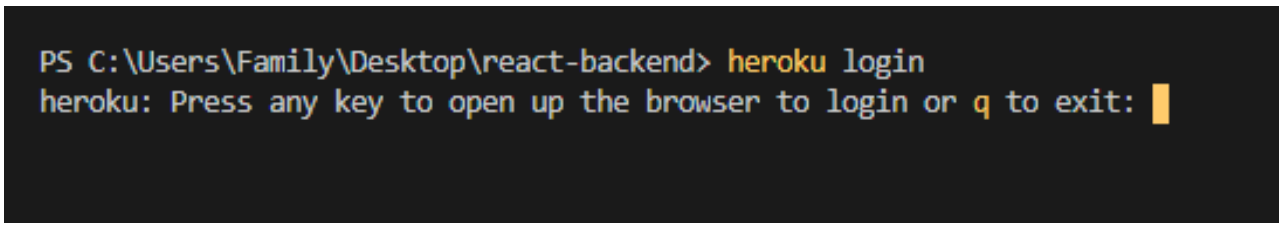

Рисунок 3.11– Робота з heroku в терміналі

Тепер хостинг створено, і можна починати встановлення на нього веб-сайту. Для цього необхідно перейти на Heroku dashbord, і вибратио роділ Deploy. В ньому розміщена інструкція з встановлення сайту на Heroku хостинг

за допомогою git, обмежуючи непотрібні файли, а також клієнтську частину, що буде розміщена на іншому хостингу, за допомогою gitignore.

Безпека даних

Кожна програма на платформі Heroku працює у власному ізольованому середовищі і не може взаємодіяти з іншими програмами чи областями системи. Це обмежувальне робоче середовище розроблено для запобігання проблемам безпеки та стабільності. Ці автономні середовища ізолюють процеси, пам’ять і файлову систему за допомогою LXC, тоді як брандмауери на базі хоста обмежують програми встановлювати локальні мережеві з’єднання.

Дані клієнта зберігаються в окремих базах даних з контрольованим доступом для кожної програми. Кожна база даних вимагає унікального імені користувача та пароля, які дійсні лише для цієї конкретної бази даних і є унікальними для окремої програми. Клієнтам із кількома програмами та базами даних призначаються окремі бази даних та облікові записи для кожної програми, щоб зменшити ризик несанкціонованого доступу між додатками.

Збережені дані можуть бути зашифровані програмами клієнтів, щоб відповідати вимогам безпеки даних. Клієнти можуть впроваджувати вимоги до зберігання даних, керування ключами та збереження даних під час розробки своєї програми.

Безперервний захист забезпечує безпеку даних на Heroku Postgres. Кожна зміна ваших даних записується в журнали попереднього запису, які відправляються в багатоцентрове сховище високої міцності. У малоймовірному випадку апаратного збою, який неможливо відновити, ці журнали можна автоматично «відтворити», щоб відновити базу даних з точністю до останнього відомого стану протягом кількох секунд. Ми також надаємо вам можливість створити резервну копію вашої бази даних, щоб відповідати вашим вимогам щодо резервного копіювання та збереження даних.

Клієнти можуть розширити функціональність програм за допомогою доповнень Heroku. Додатки пропонуються та керуються сторонніми компаніями та впроваджують власні засоби контролю та процеси безпеки.

Програми на платформі Heroku працюють у власному ізольованому середовищі і не можуть взаємодіяти з іншими програмами або областями системи, щоб запобігти проблемам безпеки та стабільності. Ці автономні середовища ізолюють процеси, пам’ять і файлову систему, тоді як брандмауери на базі хоста обмежують програми у встановленні локальних мережевих з’єднань.[16]

Встановити на хостинг клієнтську частину веб-сайту було вирішено на сервісі Netifly. Для цього потрібно просто відкрити сайт компанії, зареєструватись, або авторизуватись через Github чи GitLab, далі рекомендується налаштувати двохфакторну авторизацію, за допомогою популярних прогрма-автентифікаторів (див. рисунок). Після цього потрібно створити команду, в якій потім можна буде додавати сайти. Для цього потрібно натиснути на Create new team, обрати потрібний план і ввести необхідні дані(див. рисунок).

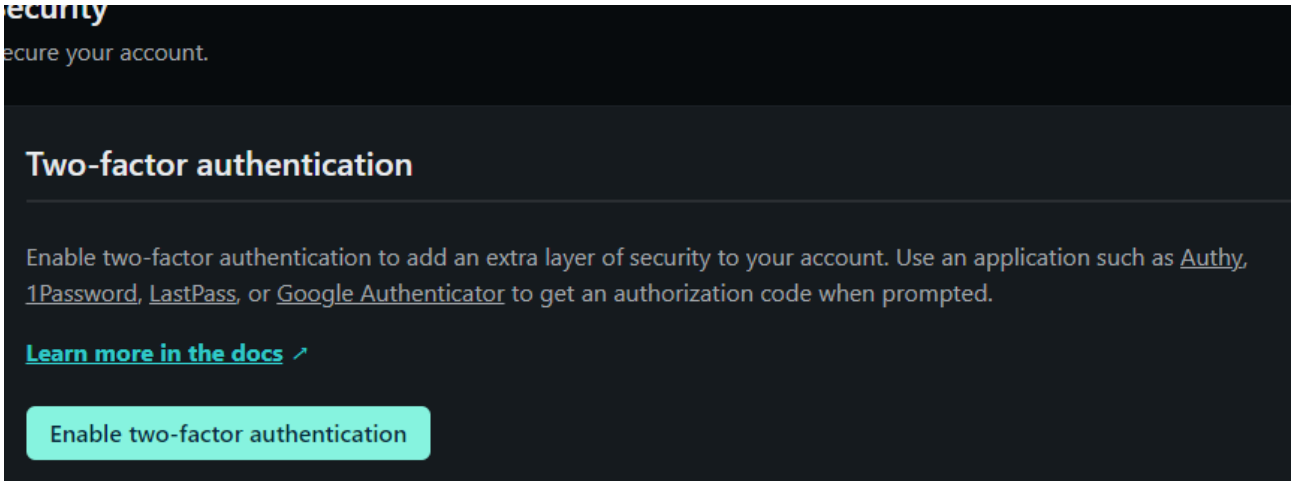

Рисунок 3.12 – Двохфакторна авторизація

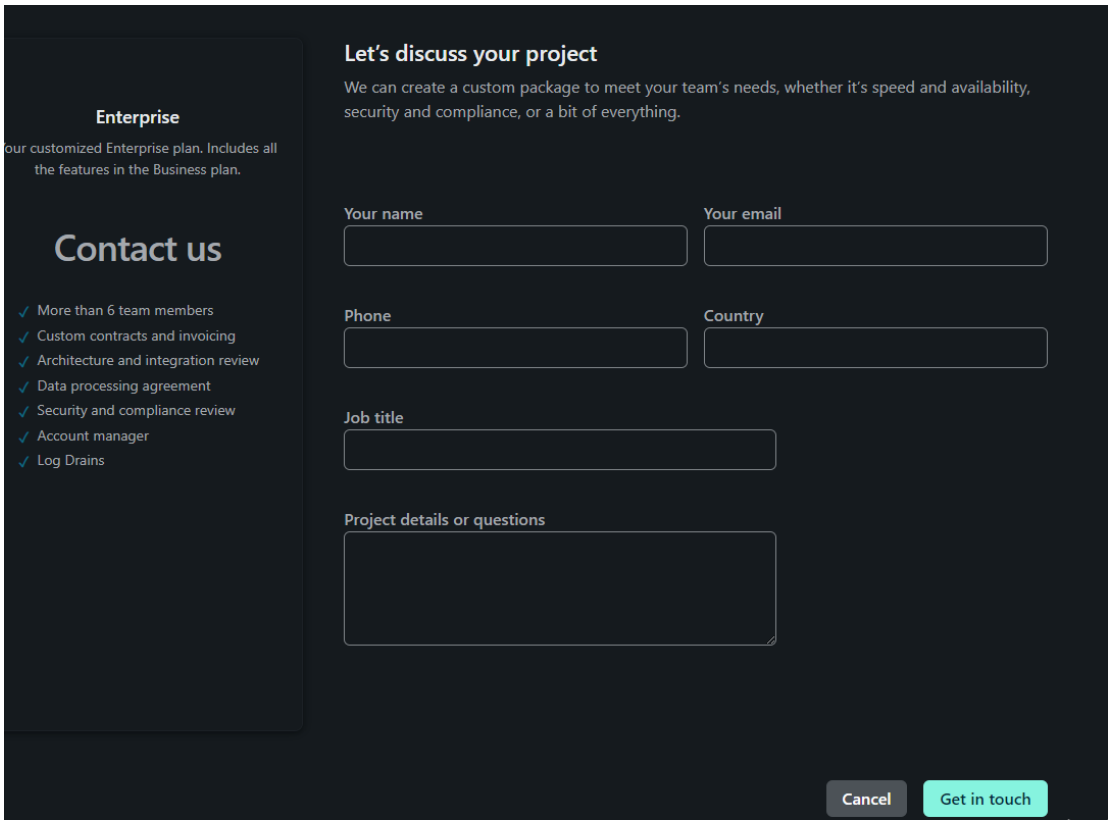

Рисунок 3.13– Створення команди

Після цього можна перейти до імпорту проекту на хостинг, для цього потрібно підключитимь до постачальника Git (див. рисунок 3.14), вибрати репозиторій з попередньо встановленими компонентами проекту (див. рисунок 3.15) і ввести команду для запуску(див. рисунок 3.16). Далі потрібно почекати кілька хвилин і netlify видасть посилання робочою клієнтською частиною веб-сайту.

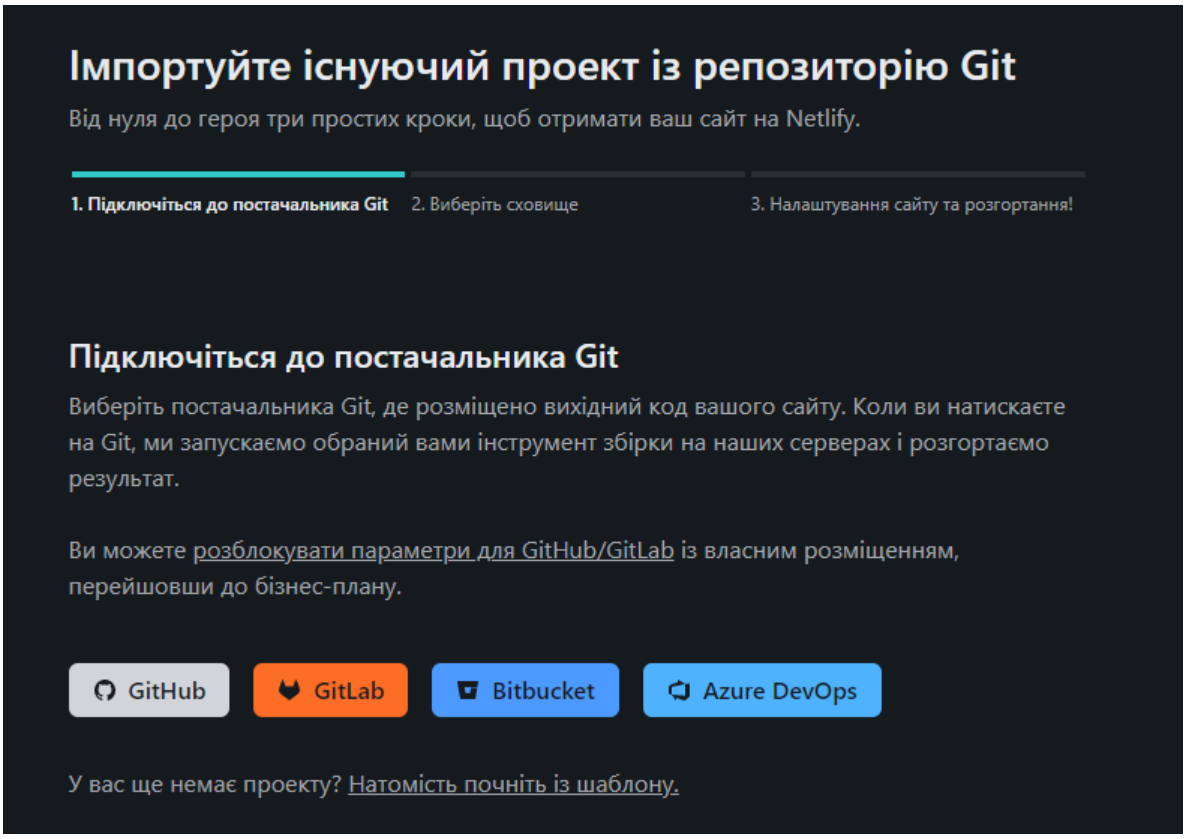

Рисунок 3.14- Вибір Git постачальника

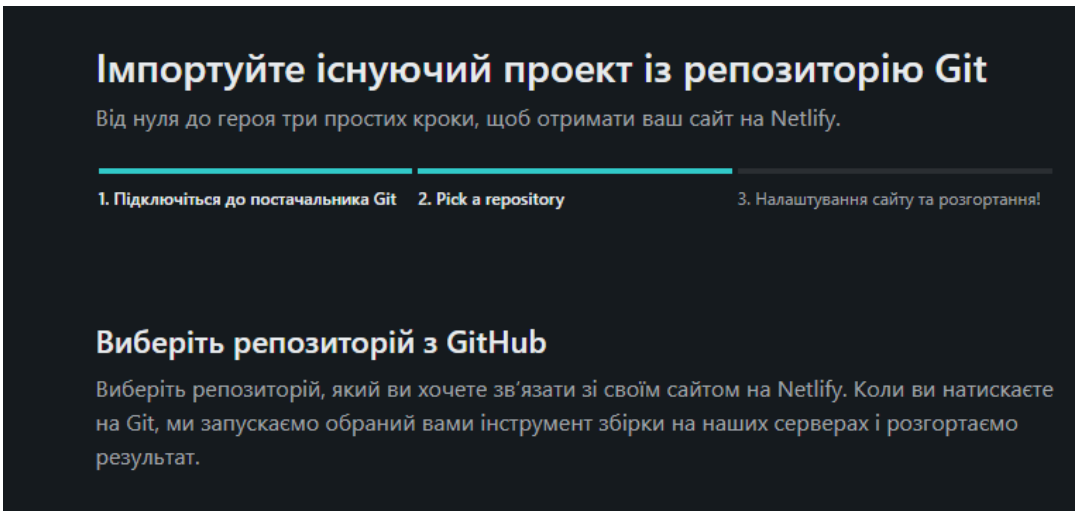

Рисунок 3.15– Вибір репозиторію

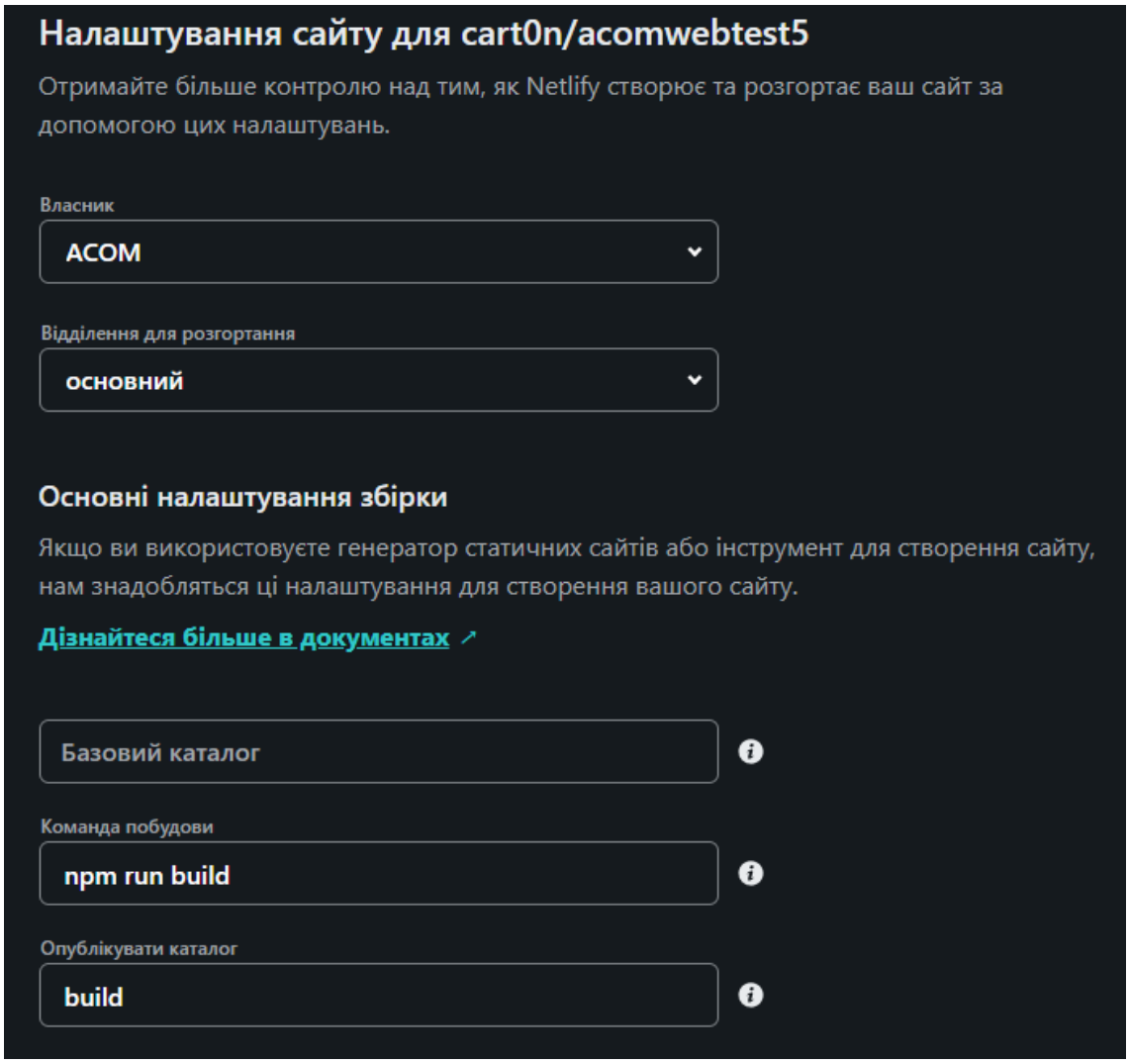

Рисунок 3.16– Базові налаштування

Безпека Netlify

Netlify щорічно проходить аудит SOC 2 типу 2, який виконує незалежний аудитор. Корпоративні клієнти можуть запросити повний звіт про аудит. Netlify відповідає стандарту PCI для всіх вимог SAQ-A для безпечної обробки транзакцій з кредитними картками.

Netlify використовує Let's Encrypt для надання безкоштовних сертифікатів HTTPS для кожного розгорнутого домену. Метою Let’s Encrypt та протоколу ACMEl є можливість налаштувати сервер HTTPS, який може автоматично отримувати сертифікат надійного браузера без будь-якого втручання людини. Це відбувається завдяки агенту керування сертифікатами запущеному на веб-сервері.

Netlify розгортається лише для основних постачальників хмарних послуг, які регулярно проходять ретельні перевірки безпеки та сертифікації, як Google Cloud, AWS

Netlify відстежує аномалії та стрибки трафіку та ефективно контролює їх за потреби, що захищає від DDos.

Netlify регулярно проводить сторонні тести на проникнення та залучає широку спільноту безпеки. Ми не пропонуємо тестування як послугу для інфраструктури, і можливість запускати такі тести або переглядати звіти про наше власне тестування зарезервовано для клієнтів на нашому рівні Enterprise.

Весь трафік через наші мережі шифрується TLS, а вся конфіденційна інформація, як-от маркери доступу, шифрується в стані спокою.

Netlify використовує глобально розповсюджених партнерів центрів обробки даних, які відповідають провідним політикам і структурам безпеки.

Журнали аудиту Netlify забезпечують прозорість різних дій, які вживаються членами команди в різних налаштуваннях команди та сайту. Журнали аудиту надають огляд та історичний журнал майже кожної дії, яку можуть виконати члени вашої команди. Такоє існують інструменти багатофакторної ідентифікації і ефективного контролю доступу для команди.[17]

Для більшої зручності користування, покращення SEO і для впевненості в безпеці можна застосувати власний домен. Для цього нам потрібно в спецальну форму ввести вже зареєстований домен чи будь який інший з можливстю зареєструвати його одразу в Netlify(див. рисунок 3.17).

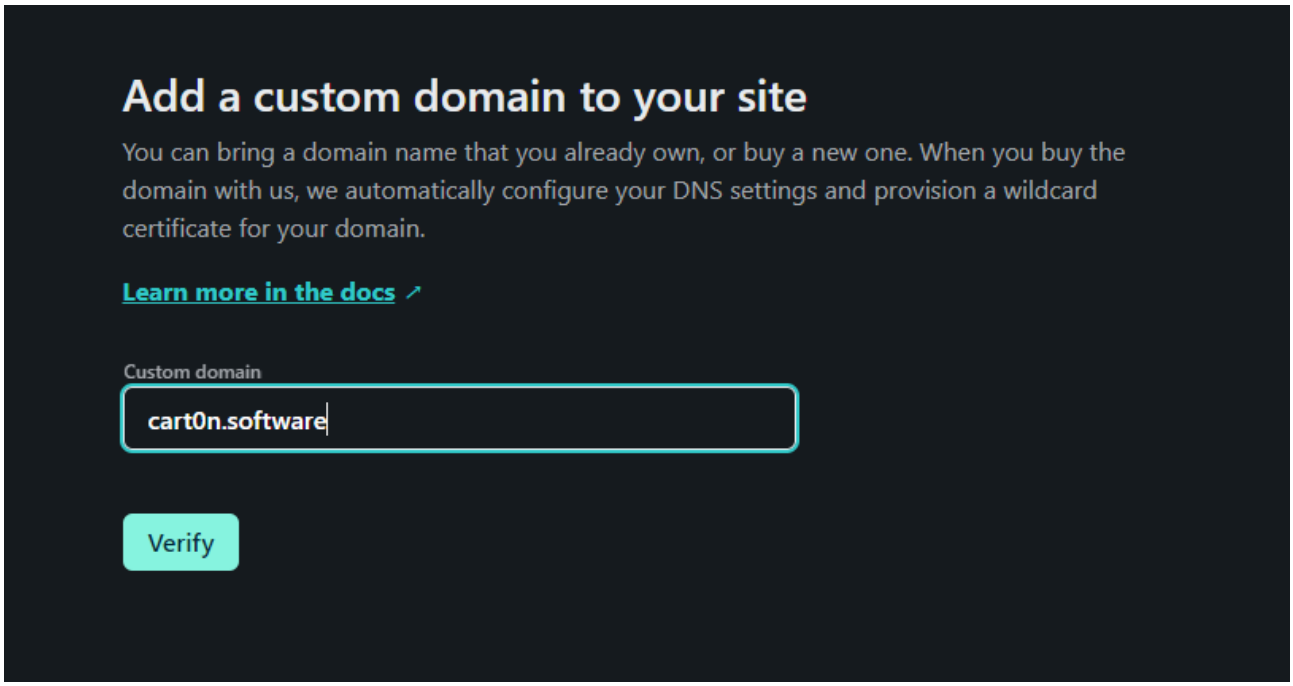

Рисунок 3.17 – Перевірка домена

Після верифікації і отримання дозволу на використання домену ми повинні налаштувати список серверів, які прописані в налаштуваннях хостингу(див. рисунок 3.18) для домену(див. рисунок 3.19).

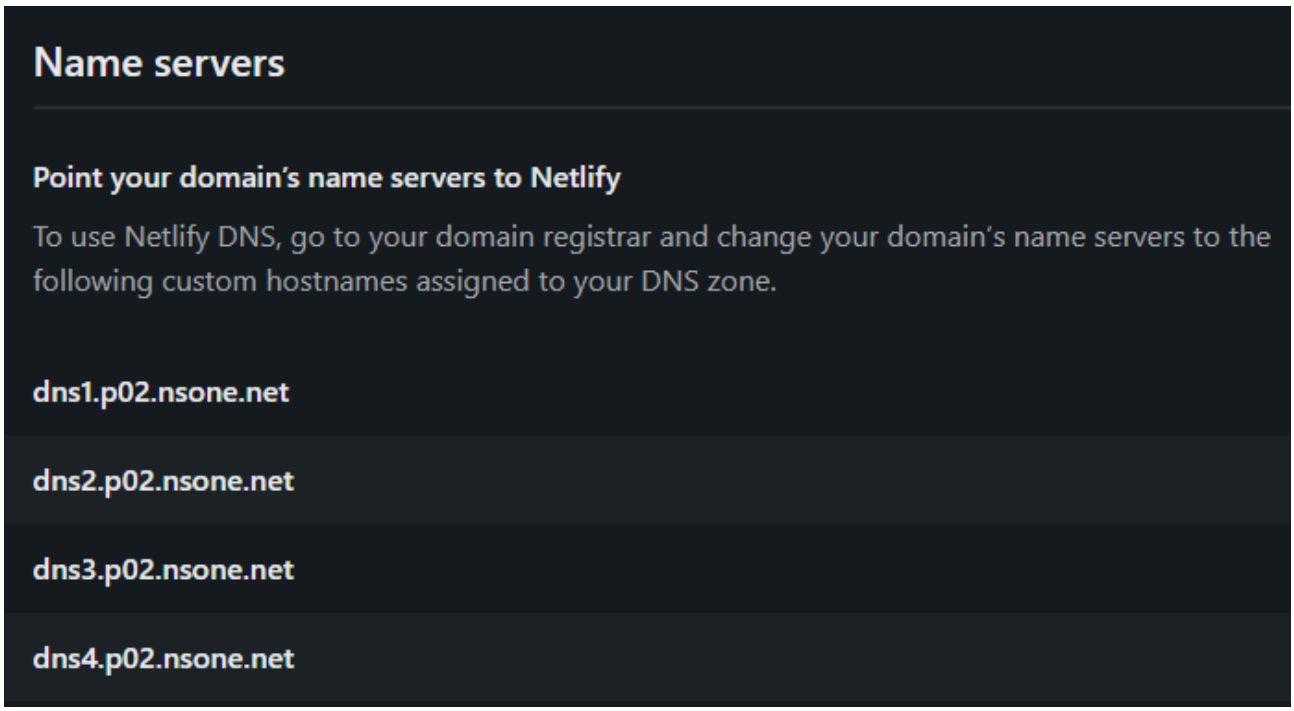

Рисунок 3.18– Список dns серверів

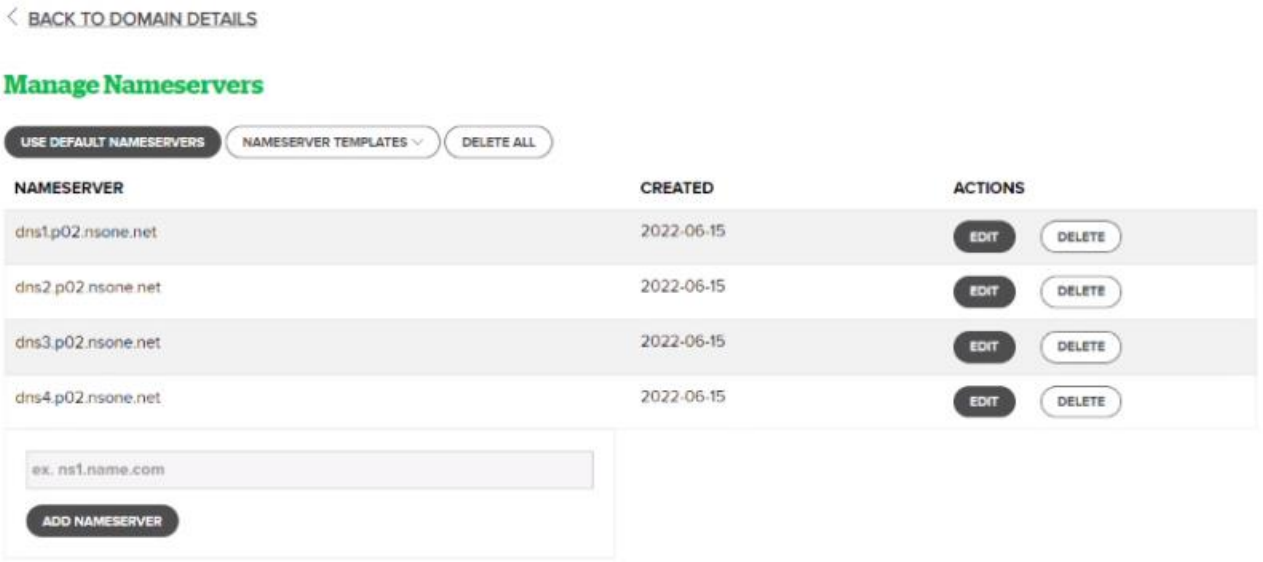

Рисунок 3.19 – Налаштування серверів на хостингу

Далі для налаштування HTTPS можна використати стандартний для Netlify сертифікат Let’s Encrypt, або використати власний сертифікат(див рисунок 3.20). Для того щоб перевірити безпеку сайту з налаштованим доментом можна використатити сервіс Criminal IP, який покаже чи дійсними є сертифікати і загалом наскільки критичним є рівень загроз для сайту(див рисунок 3.21).

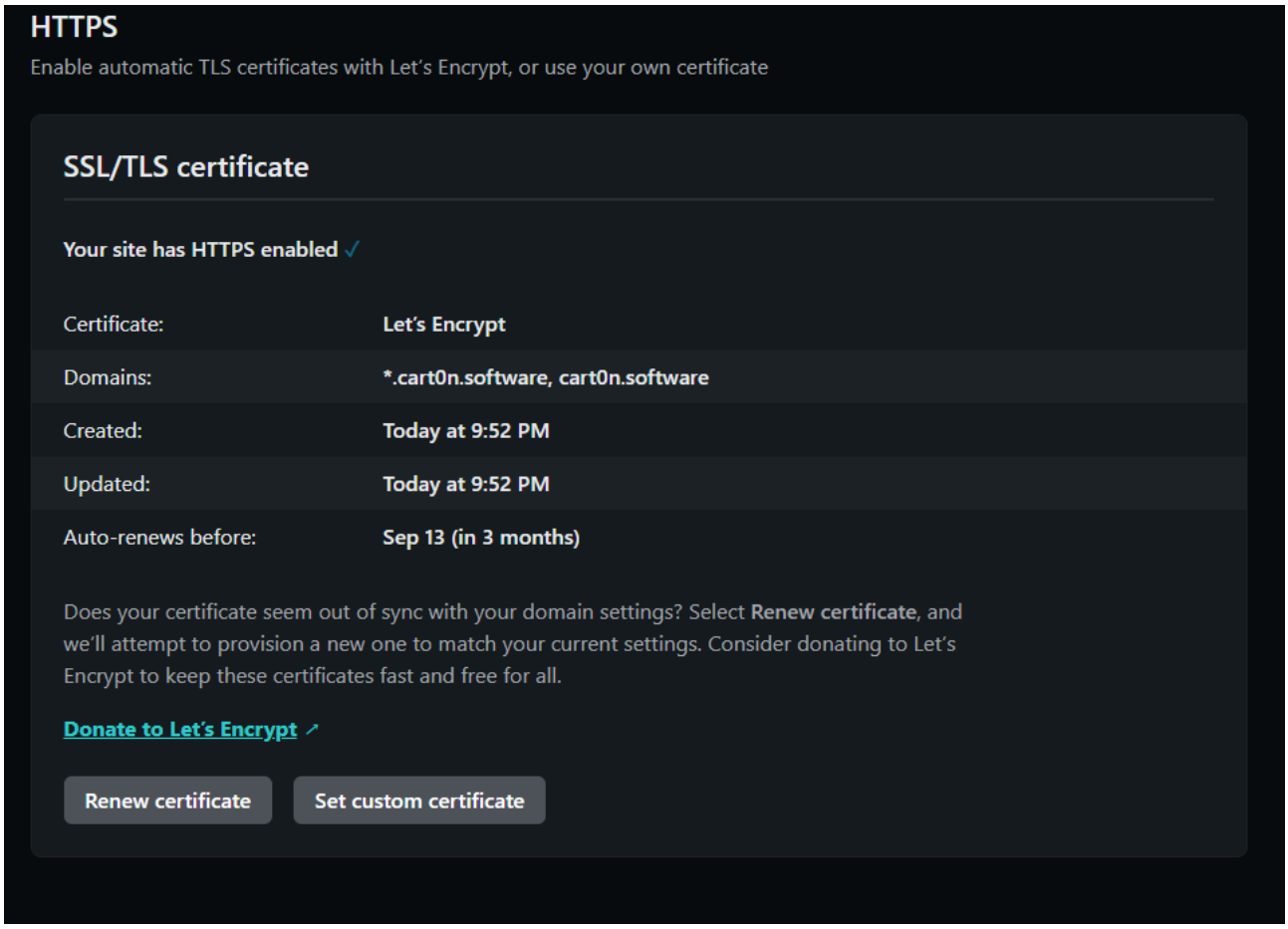

Рисунок 3.20 – Налаштування HTTPS

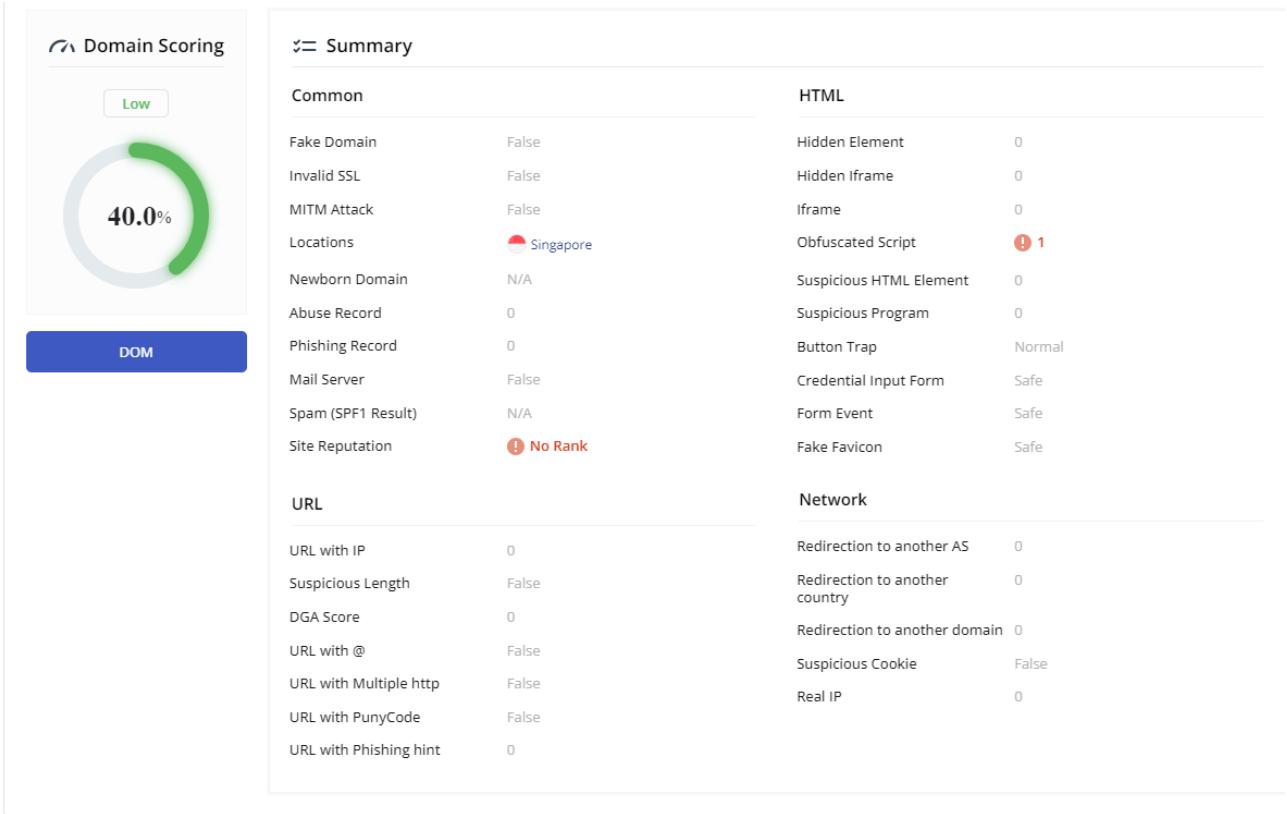

Рисунок 3.21– Низький рівень загроз сайту

4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ХОРОНИ ПРАЦІ

4.1 Соціально-політичні небезпеки, їхні види та характеристики

Соціально-політичні небезпеки досить часто виникають при соціально-політичних конфліктах. Існує досить багато визначень конфліктів. Так, у політологічних словниках найпоширенішим є таке трактування конфлікту: зіткнення двох чи більше різноспрямованих сил з метою реалізації їхніх інтересів за умов протидії.

Джерелами конфлікту є:

– соціальна нерівність, яка існує в суспільстві;

– система поділу таких цінностей, як влада, соціальний престиж, матеріальні блага, освіта.

Конфлікт — це зіткнення протилежних інтересів, поглядів, гостра суперечка, ускладнення, боротьба ворогуючих сторін різного рівня та складу учасників. Конфлікт передбачає усвідомлення протиріччя і суб'єктивну реакцію на нього. Якщо конфлікт виникає в суспільстві, то це суспільний конфлікт.

Будь-який соціальний конфлікт, набуваючи значних масштабів, об'єктивно стає соціально-політичним. Політичні інститути, організації, політичні (конфліктують політичні системи) соціальні (конфліктують соціальні системи) економічні (конфліктують економічні системи (наприклад; корпорації)) рухи, втягуючись у конфлікт, активно обстоюють певні соціально-економічні інтереси.

Конфлікти, що відбуваються в різних сферах, набувають політичної значущості, якщо вони зачіпають міжнародні, класові, міжетнічні, міжнаціональні, релігійні, демографічні та інші відносини.

Існує дві форми перебігу конфліктів:

– відкрита — відверте протистояння, зіткнення, боротьба;

– закрита, або латентна, коли відвертого протистояння немає, але точиться невидима боротьба.

За статево-віковими ознаками поділяють соціальні небезпеки, що характерні для дітей, молоді, жінок, людей похилого віку.[18]

4.2 Перша допомога людині, яка уражена електричним струмом

Ураження електричним струмом відбувається, коли електричний струм проходить через ваше тіло. Це може призвести до опіку як внутрішніх, так і зовнішніх тканин і спричинити пошкодження органів.

Ураження електричним струмом може спричинити низка причин, зокрема:

- лінії електропередач;
- блискавка;
- електричні машини;
- електрична зброя, така як електрошокери;
- побутова техніка;
- електричні розетки.

Хоча удари від побутових приладів, як правило, менш сильні, вони можуть швидко стати більш серйозними. Окрім джерела ураження, на те, наскільки серйозним є ураження електричним струмом, впливають кілька інших факторів, зокрема:

- Напруга;
- тривалість контакту з джерелом;
- загальний стан здоров'я;
- шлях електрики через ваше тіло;
- тип струму (змінний струм часто більш шкідливий, ніж постійний, тому що він викликає м'язові спазми, через які важче скинути джерело електрики).

Якщо ви або хтось ще були вражені шоком, можливо, вам не знадобиться невідкладна допомога, але ви все одно повинні звернутися до лікаря якомога

швидше. Внутрішні пошкодження від ураження електричним струмом часто важко виявити без ретельного медичного огляду.

Симптоми ураження електричним струмом залежать від його тяжкості.

Потенційні симптоми ураження електричним струмом включають:

- втрата свідомості;
- м’язові спазми;
- оніміння або поколювання;
- проблеми з диханням;
- головний біль;
- проблеми із зором або слухом;
- опіки;
- судоми;
- нерегулярне серцебиття.

Удар електричним струмом також може викликати синдром купе. Це відбувається, коли пошкодження м’язів призводить до набряку кінцівок. У свою чергу, це може стиснути артерії, що призведе до серйозних проблем зі здоров’ям. Компартмент-синдром може бути не помітним відразу після шоку, тому стежте за своїми руками і ногами після шоку.

Якщо ви або хтось ще були вражені, ваша негайна реакція може мати великий вплив на мінімізацію наслідків ураження електричним струмом.

Якщо вас ударить електричним струмом, вам може бути важко щось зробити. Але спробуйте почати з наступного, якщо ви думаєте, що були сильно шоковані:

- Відключіть джерело електроенергії, як тільки зможете.
- Якщо ви можете, зателефонуйте 911 або місцеві служби екстреної допомоги. Якщо ви не можете, крикніть, щоб хтось із вас подзвонив.
- Не рухайтеся, якщо вам не потрібно відійти від джерела електроенергії.
- Якщо шок відчувається незначним:

- Зверніться до лікаря якомога швидше, навіть якщо у вас немає жодних помітних симптомів. Пам’ятайте, що деякі внутрішні пошкодження спочатку важко виявити.
- Тим часом закрийте всі опіки стерильною марлею. Не використовуйте лейкопластири або щось інше, що може прилипнути до опіку.

Якщо хтось отримує шок, пам’ятайте про кілька речей, щоб допомогти йому і захистити себе:

- Не торкайтеся людей, які постраждали від шоку, якщо вони все ще контактують з джерелом електрики.
- Не рухайте того, хто був шокований, якщо йому не загрожує подальший шок.
- По можливості вимкніть подачу електроенергії. Якщо ви не можете, перемістіть джерело електрики від людини за допомогою непровідного предмета. Дерево і гума є хорошими варіантами. Просто переконайтеся, що ви не використовуєте нічого, що є вологим або металевим.
- Залишайтеся на відстані принаймні 20 футів, якщо вони були вражені високовольтними лініями електропередачі, які все ще увімкнені.
- Зателефонуйте 911 або місцеві служби екстреної допомоги, якщо людину вразила блискавка або якщо вона зіткнулася з високовольтною електрикою, як-от лінії електропередач.
- Зателефонуйте 911 або місцеві служби екстреної допомоги, якщо у людини проблеми з диханням, втрата свідомості, судоми, біль у м’язах або оніміння або симптоми проблеми з серцем, включаючи прискорене серцебиття.
- Перевірте дихання та пульс людини. Якщо необхідно, розпочніть СЛР до прибуття швидкої допомоги.

- Якщо у людини з’являються ознаки шоку, наприклад, блювота, слабкість або дуже бліда, злегка підніміть її ноги та ступні, якщо це не викликає занадто сильний біль.
- По можливості прикрийте опіки стерильною марлею. Не використовуйте пластирі або щось інше, що може прилипнути до опіку.
- Тримайте людину в теплі.

Навіть якщо травми здаються незначними, важливо звернутися до лікаря після ураження електричним струмом, щоб перевірити, чи немає внутрішніх травм.

При сильних потрясіннях лікар може порекомендувати залишитися в лікарні на день або два, щоб вони могли спостерігати за вами на наявність проблем із серцем або серйозних травм.

Деякі ураження електричним струмом можуть мати тривалий вплив на ваше здоров’я. Наприклад, серйозні опіки можуть залишити постійні рубці. А якщо електричний струм проходить через очі, у вас може залишитися катаракта.

Деякі удари також можуть викликати тривалий біль, поколювання, оніміння та м’язову слабкість через внутрішні травми.

Якщо дитина отримує травму губи або опік від пережовування шнура, у неї також може виникнути сильна кровотеча, коли струп зрештою відпаде. Це нормально через кількість артерій на губі.[19]

ВИСНОВКИ

У даному дипломному проекті розроблено захищений веб-сайт для компанії “Аргоком”. Даний веб-сайт орієнтований для широкого застосування у інформаційній сфері. З його допомогою користувачі зможуть швидко та зручно дізнаватися інформацію про компанію. При його розробці було застосовано рекомендації по безпеці і налаштовано його захист.

При розробці веб-сайту були проаналізовані сучасні веб-технології, що дозволяють створювати інтерактивні веб-сторінки, а також безпекові характеристики і засоби забезпечення їх захисту.

Основний результат виконання дипломного проекту – стабільно працюючий на усіх пристроях сайт, розміщений у мережі Інтернет з відсутністю критичних загроз через використання і налаштування існуючих засобів захисту проєкту, і при цьому придатний для користування і використання звичайними користувачами згідно з встановленими потребами.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. OWASP Top Ten [Електронний ресурс] - Режим доступу до ресурсу: https://owasp.org/www-project-top-ten/ - Дата доступу: 13.06.2022
2. Pankaj Pant Secure web development [Електронний ресурс] - Режим доступу до ресурсу: https://www.theseus.fi/bitstream/handle/10024/752609/Pant_Pankaj.pdf?sequence=2 - Дата доступу: 13.06.2022
3. Secure Web development using OWASP Guidelines [Електронний ресурс] - Режим доступу до ресурсу: https://e-tarjome.com/storage/panel/fileuploads/2022-03-12/1647077628_E16176.pdf - Дата доступу: 13.06.2022
4. React или Angular или Vue.js — что выбрать? [Електронний ресурс] - Режим доступу до ресурсу: https://habr.com/ru/post/476312/ - Дата доступу: 13.06.2022
5. MySQL і MongoDB - коли і що краще використовувати [Електронний ресурс] - Режим доступу до ресурсу: https://habr.com/ru/post/322532/ - Дата доступу: 13.06.2022
6. Security Analysis of MongoDB [Електронний ресурс] - Режим доступу до ресурсу: https://infonomics-society.org/wp-content/uploads/Security-Analysis-of-MongoDB.pdf - Дата доступу: 13.06.2022
7. React - JavaScript-бібліотека для створення користувацьких інтерфейсів: [Електронний ресурс] - Режим доступу до ресурсу: https://uk.reactjs.org/ - Дата доступу: 13.06.2022
8. Про адаптивність сайту і її вплив [Електронний ресурс] - Режим доступу до ресурсу: https://cityhost.ua/uk/blog/pro-adaptivnost-sayta-i-ee-vliyanie.html - Дата доступу: 13.06.2022
9. ЯК: Що таке постачальник CSS або префікс браузера?: [Електронний ресурс] - Режим доступу до ресурсу: https://uk.reactjs.org/ - Дата доступу: 13.06.2022

10. Express.js Security Tips [Електронний ресурс] - Режим доступу до ресурсу: https://www.freecodecamp.org/news/express-js-security-tips/ - Дата доступу: 13.06.2022
11. MONGODB ATLAS Security [Електронний ресурс] - Режим доступу до ресурсу: https://www.mongodb.com/cloud/atlas/security - Дата доступу: 13.06.2022
12. What is Snyk? [Електронний ресурс] - Режим доступу до ресурсу: https://snyk.io/what-is-snyk/ - Дата доступу: 13.06.2022
13. Що таке хостинг і для чого він потрібен [Електронний ресурс] - Режим доступу до ресурсу: https://hostkoss.com/ua/what-is-hosting.html - Дата доступу: 13.06.2022
14. Cloud Application Platform| Heroku [Електронний ресурс] - Режим доступу до ресурсу: https://www.heroku.com/ - Дата доступу: 13.06.2022
15. Развертываем свой сайт на Heroku [Електронний ресурс] - Режим доступу до ресурсу: https://habr.com/ru/post/232679/ - Дата доступу: 13.06.2022
16. Heroku Security [Електронний ресурс] - Режим доступу до ресурсу: https://www.heroku.com/policy/security - Дата доступу: 13.06.2022
17. Security at Netlify [Електронний ресурс] - Режим доступу до ресурсу: https://www.netlify.com/security/ - Дата доступу: 13.06.2022
18. Соціально-політичні небезпеки, їхні види та особливості [Електронний ресурс] - Режим доступу до ресурсу: https://knmau.com.ua/wp-content/uploads/1-bak.-F-no-Nar.-instr.-Orkestr.1-2-grupi-Opern.-spiv-Operno-simf.dir.-BZHD-TEMA-5-Snizhko.pdf - Дата доступу: 13.06.2022
19. First Aid 101: Electric Shocks [Електронний ресурс] - Режим доступу до ресурсу: https://www.healthline.com/health/electric-shock#symptoms - Дата доступу: 13.06.2022