МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської та
аспірантської підготовки
Кафедра інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Застосування захисту особистої інформації в інформаційних системах за участю протоколу HTTPS»

Виконав студент 2 курсу групи МІС-18
Напрям підготовки 6.050101
Комп'ютерні науки
Кучер Микита Сергійович

Керівник асистент
Кругляк Юрій Олексійович

Рецензент д.т.н., професор
Мещеряков Володимир Іванович

Одеса 2019

АНОТАЦІЯ

на магістерську роботу «Застосування захисту особистої інформації в інформаційних системах за участю протоколу HTTPS»,
студента Кучера Микити Сергійовича

Актуальність дослідження полягає в необхідності розробки інформаційної системи, що буде сприяти інформуванню користувачів мережі Інтернет про безпечність різних інформаційних систем.

Мета дослідження – є створення інформаційної системи «Safeshell» на платформі СКБД mySQL для перевірки та аналізу елементів безпеки вебресурсу – від наявності захищеного протоколу і інформації щодо нього до відгуків інших користувачів про нього.

Задачі дослідження: провести огляд принципів захисту інформаційних систем; провести аналіз елементів які забезпечують безпеку особистих даних; підготувати вихідні дані до проекту, шкала рівня безпеки, «чорний список» ІС; розробити ІС для перевірки безпеки інших ІС.

Об'єкт дослідження – комунікаційні протоколи передачі даних та їх вплив на безпеку особистих даних.

Предмет дослідження – є розробка процесу відношення протоколів передачі даних до особистої інформації користувача в глобальній мережі.

Методи дослідження: протоколи передачі даних HTTP та HTTPS, методи збору та обробки інформації користувача інформаційними системами.

Результати, їх новизна, теоретичне та практичне значення: Розроблена ІС аналізу інших ресурсів в мережі Інтернет. Створена система забезпечить користувача інформацією про рівень безпеки, дані які будуть зберігатися та оброблюватися системою та відгуки інших користувачів.

Структура магістерської роботи складається з вступу, чоторьох розділів, висновків, переліку посилань на 12 найменувань, додатків. Повний обсяг роботи становить 81 сторінку і містить 43 рисунки.

КЛЮЧОВІ СЛОВА: інформаційна система, захист особистих даних.

SUMMARY

to master's work «Application of Personal Information Protection in Information Systems by Means of HTTPS Protocol»,

of student Kucher Mykyta Serhiyovych

The relevance is the need to develop an information system that will help to inform the Internet users about the security of various information systems.

The purpose is to create a “Safeshell” information system on the mySQL DBMS platform for checking and analyzing security features of a web resource - from the presence of a secure protocol and information about it to the feedback of other users about it.

Research objectives: review the principles of protection of information systems; to carry out the analysis of elements that ensure the security of personal data; prepare baseline data for the project, security level scale, IS blacklist; develop ISs to test the security of other ISs.

The Object – communication protocols for data transmission and their impact on the security of personal data.

The subject of the research – is the development of the process of relation of data transmission protocols to personal information of the user into the global network.

Research Methods: HTTP and HTTPS communication protocols, methods for collecting and processing user information by information systems.

Results, their novelty, theoretical and practical significance: An IS for analyzing other resources on the Internet has been developed. The system created will provide the user with information about the security level, data that will be stored and processed by the system and feedback from other users.

The structure of the master's work consists of an introduction, four sections, conclusions, a list of references to 12 titles, applications. The full volume is 81 pages and contains 43 figures.

KEYWORDS: information system, personal data protection, HTTPS.

ЗМІСТ

Скороченя та умовні познаки 10
Вступ 12
1 Аналіз предметної області 14
1.1 Опис предметної області 14
1.2 Аналіз подібних систем 15
1.2.1 Види веб-додатків 15
1.2.2 Подібні системи 17
1.3 Постановка завдання 23
1.4 Етапи розробки сайту 26
1.5 Етапи розробки веб-додатку 28
2 Вибір програмних засобів для реалізації системи 30
2.1 Вибір фреймворку для створення інтерфейсів користувача 30
2.1.1 Опис фреймворку AngularJS 30
2.1.2 Опис фреймворку Vue.js 31
2.1.3 Опис бібліотеки React 33
2.1.4 Загальна схема роботи фреймворку React 34
2.1.5 Переваги React перед іншими фреймворками 36
2.1.6 Порівняння фреймворків 37
2.2 Переваги використання FireBase 38
2.3 Особливості використання баз даних FireBase Cloud FireStore 39
2.4 Особливості використання серверу баз даних MYSQL 41
2.5 Переваги використання PHP 45
3 Моделювання інформаційної системи 47
3.1 Загальні вимоги до інформаційної системи 47
3.2 Функціональні можливості користувачів web-системи 48
3.3 Проектування даталогічної моделі бази даних 50
3.3.1 Опис таблиць бази даних і зв'язків між ними 56

4 Реалізація інформаційної системи 62
4.1 Схема функціонування web-системи 62
4.2 Реалізація структури Web-системи 65
4.3 Інтерфейс користувача адміністратора 66
4.4 Управління інтерфейсами користувачів системи 71
4.5 Управління інтерфейсами веб-додатку 79
4.6 Структура програмного забезпечення web-системи 80
Висновки 82
Перелік джерел посилання 83
Додаток А Життєвий цикл react.js 84
А.1 Схема циклу react.js 84
Додаток Б Обмін повідомлень Firebase 85
Б.1 Схема отримання повідомлень Firebase 86

СКОРОЧЕНЯ ТА УМОВНІ ПОЗНАКИ

Клас – це спеціальна конструкція, яка використовується для групування пов’язаних змінних та функцій.

Рендеринг – в комп’ютерній графіці це процес отримання зображення за моделлю за допомогою комп’ютерної програми.

API – набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення.

DOM – специфікація прикладного програмного інтерфейсу для роботи зі структурованими документами.

HTTP – протокол прикладного рівня передачі даних.

HTTPS – розширення протоколу HTTP для підтримки шифрування з метою підвищення безпеки.

MVC – схема поділу даних програми, призначеного для користувача інтерфейсу і керуючої логіки на три окремих компоненти: модель, уявлення і контролер.

FPS – це частота (швидкість), з якою пристрій формування зображення відображає послідовні зображення, що називаються кадрами.

SSL – криптографічний протокол, який забезпечує встановлення безпечного з'єднання між клієнтом і сервером.

TCP/IP – це промисловий стандарт стека протоколів, розроблений для глобальних мереж.

TLS – криптографические протоколы, обеспечивающие защищённую передачу данных между узлами в сети Интернет.

URI – компактний рядок літер, який однозначно ідентифікує окремий абстрактний чи фізичний ресурс.

XML – стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками, зокрема, через Інтернет.

БД – база даних.

ІМ – інформаційна мережа.

ІС – інформаційна система.

КС – комп'ютерні системи.

СУБД – система управління базою даних.

API – Application Programming Interfaces, набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення.

CLI – Common Language Infrastructure, специфікація загальномовної інфраструктури.

CLR – Common Language Runtime, загальномовне виконуюче середовище.

DOM – Document Object Model.

FK – foreign key (зовнішній ключ).

HTTP – Hypertext Transfer Protocol.

HTTPS – Hypertext Transfer Protocol Secure.

IDE – Integrated development environment, інтегроване середовище розробки.

IDE – Integrated development environment, інтегроване середовище розробки.

PK – primary key (первинний ключ).

SQL – Structured Query Language.

SSL – Secure Sockets Layer.

TCP – Transmission Control Protocol.

TLS – Transport layer security, криптографічні протоколи, що забезпечують захищену передачу даних між вузлами в мережі Інтернет.

URI – Uniform Resource Identifier, ідентифікує окремий абстрактний чи фізичний ресурс.

URL – Uniform Resource Locator, адреса певного ресурсу.

VCS – Version Control System, система контролю версій.

XXS – Cross-Site Scripting, міжсайтовий скриптинг.

ВСТУП

Добре відомо, що в сучасному світі інформація має певну, а часто і дуже високу цінність. Як і будь-яку цінність її потрібно захищати. Захист необхідний, наприклад, від втрат через випадкове видалення, збоїв, вірусів чи несанкціонованого доступу до інформації.

Під заходами щодо захисту від несанкціонованого доступу маються на увазі ті, що пов'язані з секретністю інформації. До їх числа відносяться найрізноманітніші способи захисту, починаючи від найпростіших, але дуже ефективних захистів паролем до використання складних технічних систем.

На сьогоднішній час ніяка система захисту не забезпечує стовідсоткову надійність. Досить надійним вважається така система захисту інформації, яка забезпечує її захист протягом необхідного періоду часу. Іншими словами, система захист інформації повинна бути такою, щоб на її злом було потрібно більше часу, ніж час, за який ця інформація повинна залишатися секретною.

Останні роки інтернет-технології розвиваються дуже швидко, і велика частка населення має у себе на смартфоні, вдома або на роботі вихід до його ресурсів, виникає необхідність використання цього напрямку в цілях захисту даних.

За оцінками експертів, кількість користувачів буде збільшуватися кожний рік. Інтернет має величезний потенціал, і багато що може запропонувати. Будучи інноваційним рішенням в галузі науки і техніки, Інтернет має свої переваги. До переваг в першу чергу можна віднести високу швидкість зв’язку, необмежене спілкування, що включає листування, гоолосове спілкування, відеозвя’зок. Узагальнюючи дані різних досліджень з вивчення інтернет-аудиторій, великий відсоток аудиторії інтернету складають активні верстви населення, які беруть участь в процесі прийняття рішень.

Інтернет – це вільний доступ до будь-якої інформації. Безпека даних є однієї з головних проблем у Інтернет. Для відвідувачів є ризик потрапити на

інформаційну систему з незахищеним типом зв’язку. В даному випадку користувач може може відіслати свою особисту інформацію до рук зловмисників.

Інтернет і інформаційна безпека несумісні по самій природі. Вона народилася як чисто корпоративна мережа, однак, у даний час за допомогою єдиного стека протоколів TCP/IP і єдиного адресного простору поєднує не тільки корпоративні і відомчі мережі (освітні, державні, комерційні, військові і т.д.).

Уразливість інформації в комп'ютерних системах обумовлена великою концентрацією обчислювальних ресурсів, їх територіальним розташуванням, довготривалим зберіганням великих обсягів даних, одночасним доступом до ресурсів КС численних користувачів. Щодня з'являються все нові і нові загрози (мережеві атаки, несанкціоновані втручання в КС), тому гострота проблеми інформаційної безпеки з часом не зменшується, а навпаки набуває все більшої актуальності.

У предметну область інформаційної безпеки входять такі питання, як: класифікація та аналіз загроз безпеки, політика інформаційної безпеки, засоби та методи захисту інформації, а також управління ними.

Методи роботи ґрунтуються на принципах інформаційного моделювання предметної області. Дослідження проводилося за допомогою вивчення предметної області. Підбір необхідних матеріалів по розробці сайтів здійснювався з Інтернету і відповідної літератури.

Метою роботи є виявлення основних джерел загрози інформації і визначення способів захисту від них та створення інформаційної мережі та браузерного додатку під назвою Safeshell яка здатна надати безкоштовний інформаційний ресурс, що надає користувачу важливу інформацію про безпечність вибраних інформаційних систем.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

Цілью розробки даної дипломної роботи є створення інформаційної системи та браузерного додатку які можуть надавати детальні відомості про ІС: тип підключення, сертифікати безпеки, дозволах які вона потребує та відгуки.

У разі виявлення небезпечних та незахищених ІС мережі інтернет, які можуть становити загрозу особистої інформації користувач буде одразу попереджений.

HTTPS це схема URI, що синтаксично ідентична http схемі, яка зазвичай використовується для доступу до ресурсів Інтернет. Використання https URL вказує, що протокол HTTP має використовуватися, але з іншим портом за замовчуванням це 443 і додатковим шаром шифрування чи автентифікації між HTTP і TCP. ІС Safeshell при отримані URL адреси перевіряє його на наявність протоколу [1][1)].

ІС слідкує, щоб була забезпечена конфіденційність обміну даними між клієнтом і сервером, що використовують TCP/IP.

За допомогою сайту можливо отримати відповідь чи є вибрана користувачем ІС безпечною для відвідування. Також, особливістю сайту буде те, що відвідувач сайту може не тільки отримати детальну інформацію про протокол, криптографічний алгоритм шифрування та сертифікат, але і при бажанні поставити оцінку та залишити коментар ІС, щоб інші відвідувачі змогли побачити досвід користування минулих користувачів сайту. Система проведе аналіз та простою мовою розповість про безпечність сайту користувачу.

[1)] [1] HTTPS – Википедия. URL: https://uk.wikipedia.org/wiki/HTTPS (дата звернення 3.12.2019).

Існує два типи програм:

- схожі на звичайні сайти (наприклад, Gmail і Google Календар);
- використовують браузер;
- схожі на комп'ютерні програми і працюють на основі функцій браузеру (наприклад, VPN);
- невеликий розмір;
- працюють швидше.

Веб-додатки – це програми, з якими кінцевий користувач може працювати через браузер. З їх допомогою можна створювати файли, редагувати фотографії, слухати музику і так далі. Цілком очевидно, для користувача який користується IC Safeshell не буде зручно завжди відвідувати веб-сайт – захист у фоновому режимі у вигляді веб-додатку це набагато зручніше та швидкіше.

1.2 Аналіз подібних систем

1.2.1 Види веб-додатків

Основні види сайтів: Соціальні мережі, Бізнес сайти, Веб-сервіс, Інформаційні сайти, Комбіновані сайти.

Соціальна мережа – спрямована на побудову спільнот в Інтернеті з людей зі схожими інтересами. Зв'язок здійснюється за допомогою сервісу внутрішньої пошти або миттєвого обміну повідомленнями.

Сайт-візитка – містить загальні дані про власника сайту (організація або індивідуальний підприємець). Вид діяльності, історія, прайс-лист, контактні дані, реквізити, схема проїзду. Фахівці розміщують своє резюме. Тобто детальна візитна картка.

Представницький сайт – так іноді називають сайт-візитку з розширеною функціональністю: докладний опис послуг, портфоліо, відгуки, форма зворотного зв'язку і таке інше.

Корпоративний сайт – містить повну інформацію про компанії-власника, послуги та події в житті компанії. Відрізняється від сайту-візитки і представницького сайту повнотою наданої інформації, часто містить різні функціональні інструменти для роботи з контентом (пошук і фільтри, календарі подій, фотогалереї, корпоративні блоги, форуми). На таких сторінках зазвичай набувають особливої важливості такі метрики, як показник відмов або глибина прокрутки. Може бути інтегрований з внутрішніми інформаційними системами компанії-власника (КІС, CRM, бухгалтерськими системами). Може містити закриті розділи для тих чи інших груп користувачів-співробітників, дилерів, контрагентів тощо.

Каталог продукції – в каталозі є докладний опис товарів або послуг, сертифікати, технічні та споживчі дані, відгуки експертів і т. д. На таких сайтах розміщується інформація про товари / послуги, яку неможливо помістити в прайс-лист.

Інформаційний сайт – це перш за все джерело інформації. Тому він містить новини, велика кількість статей і графічного матеріалу з певної тематики. Інформаційні сайти нагадують енциклопедії або спеціалізовані журнали. Сайт може містити безліч інформаційних статей на всі тематики або ж декілька та розповідати про новинки в світі технологій.Часто тут присутні інтерактивні елементи, такі як голосування, опитування, розміщується реклама.

Інтернет-магазин – веб-сайт з каталогом продукції, за допомогою якого клієнт може замовити потрібні йому товари. Використовуються різні системи розрахунків: від пересилання товарів післяплатою або автоматичною пересилання рахунку по факсу до розрахунків за допомогою пластикових карт.

Промо-сайт – сайт про конкретну торгову марку або продукт, на таких сайтах розміщується вичерпна інформація про бренд, різних рекламних акціях.

Тематичний сайт – веб-сайт, що надає специфічну вузькотематичні інформацію про будь-якої теми [2][1)].

1.2.2 Подібні системи

Інформаційних систем, які так чи інакше надають інструменти для сканування веб-сторінки на предмет вразливості безпеки та небезпечного програмного забезпечення не дуже багато. Одні лише надають вичерпні звіти про такі різновиди тесту на безпеку, як SQL ін'єкція, Cross Site Scripting та глибокий аналіз HTTPS URL-адреси, інші виводять код PHP, основний код сторінки, вид шифрування, версію SSL та TLS, моделювання рукостискання, деталі протоколу та виведення заголовка HTTP, треті мають чорний список веб-сайтів та тест на наявність ін'єкційного спаму та небезпечних елементів сторінки.

Розглянемо декілька з них.

Сайт для аналізу конфігурації SSL веб-серверу «Qualys»:

- головна: тут можливо знайти головну сторінку з навігаційною шапкою, різновидом проектів, контактами, полем для ведення адреси, та трьома колонками з адресами;
- новини: інформація про оновлення послуг;
- проекти: інші продукти компанії;
- про нас: опис даної компанії;
- послуги: опис послуг, що надаються;
- контакти: контактна інформація для зв’язку;
- колонки: раніше переглянуті, найкращі сайти, раніше найгірші сторінки.

Короткий опис сайту:

[1)] [2] Створення та розміщення власних веб-сайтів на безкоштовних хостінгах. URL: https://studfiles.net/preview/5535698/page:16 (дата звернення 3.12.2019).

Дизайн сайту заходів безпеки "Qualys" (рис. 1.1) має дуже простий інтерфейс на головній сторінці, що надає достатню інформацію про послуги які він надає, короткий опис який інформує користувача про роботу сайту. У центрі сторінки користувач одразу бачить поле для ведення URL та чек-бокс для фільтрування адрес ресурсу. Також на цьому сайті є корисна інформація про раніше веденні адреси користувачами які містять внутрішню оцінку, контактними даними, проектами та посиланням на продовження пробної версії. До недостатків сайту можна віднести мінімальний функціонал та платну підписку.

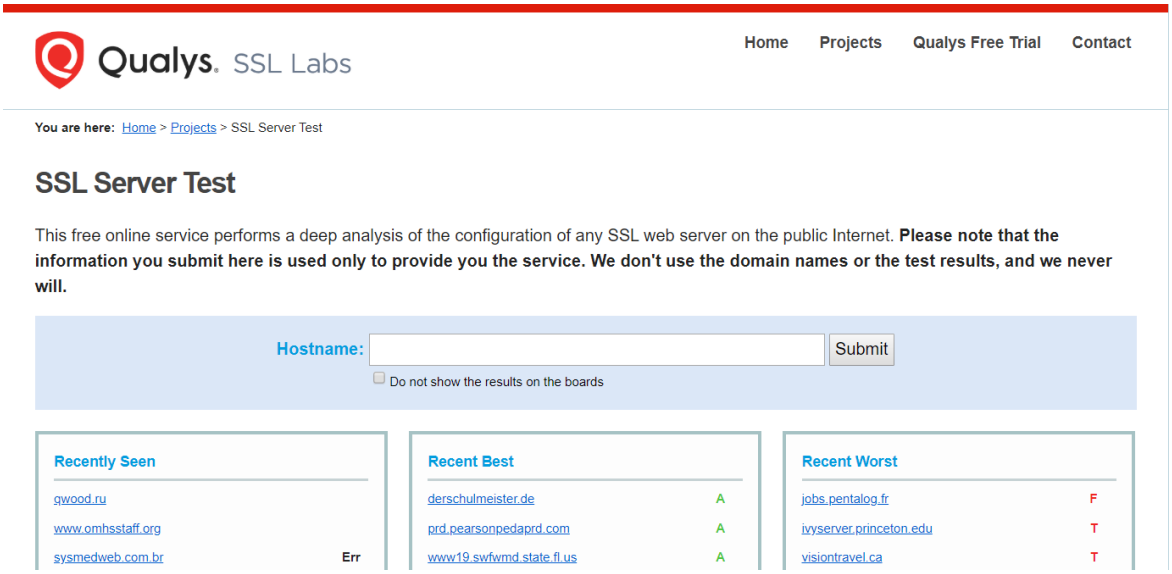


Рисунок 1.1 – Сайт для аналізу SSL серверу «Qualys»

Сайт антивірус «Quttera» (рис. 1.2):

– головна: тут можливо знайти головну сторінку з інформацією о компанії, описом послуг, розцінками;
– про нас: опис даної компанії;
– послуги: опис послуг, що надаються;
– досвід: опис накопичених робіт;
– контакти: контактна інформація для зв’язку.

Короткий опис сайту:

Дизайн сайту «Quttera» має досить застарілий інтерфейс, не сучасний дизайн надає враження, що його давно не оновлювали. Головна сторінка досить важка на розуміння але вона включає в себе майже всю необхідну інформацію сайту. Верхнє меню має цілком звичайний дизайн та колір сторінки. На сторінці немає функції вибору мови крім англійської, що ускладнює розуміння для користувача. На сайті багато зайвої інформації, що заважає знайти головні елементи форми пошуку. Також до недостатків можна віднести дуже довгий аналіз URL адреси, який пов'язаний з чергою запитів інших користувачів сайту. Інформація після завершення аналізу не інформативна, та містить рекаму.

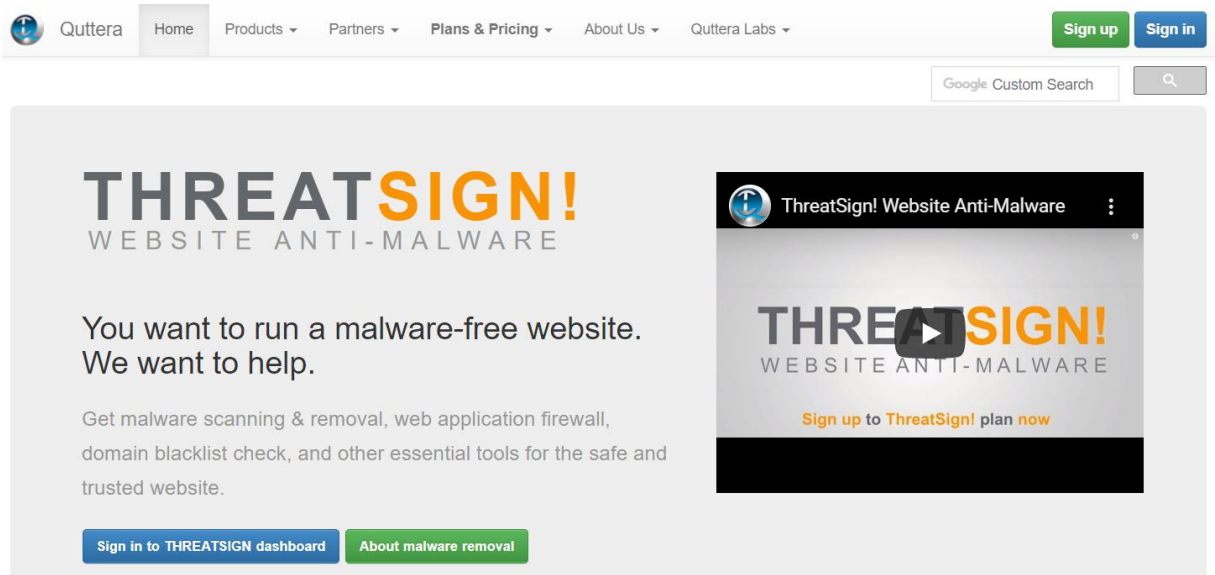


Рисунок 1.2 – Сайт антивірус «Quttera»

Сайт сканування безпеки веб-сайту «UpGuard»:

– головна: тут можливо знайти головну сторінку з описом сайту, послуги, контакти та відгуки;
– інформація: опис дослідженого URL;

- проекти: тут знаходиться проекти в яких бере участь компанія та їх описом;
- рейтинг: загальний рейтинг досліджених сайтів;
- популярні питання: міститься популярні питання з відповідями на них;
- перевірка безпеки: розділи з результатами перевірки введеного сайту.

Короткий опис сайту:

Сайт має дуже мінімалістичний дизайн (рис. 1.3). На головній сторінці немає нічого зайвого. Перелік пропонованих програм розташован дуже компактно, і за рахунок цього можна побачити якомога більше інформації за раз. Це дуже зручно і економно за часом. На сайті не має віджета пошуку. Сайт орієнтований скоріше на державні структури ніж на особисте користування, що не дуже зручно. Верхнє меню має гарній дизайн та колір. Перелік пропонованих результатів дослідження сайту має короткий, але розкритій опис. Головна сторінка досить легка та надає багато інформації про роботу сайту для користувачів. На головній сторінці знаходиться вичерпна інформація про принцип роботи.

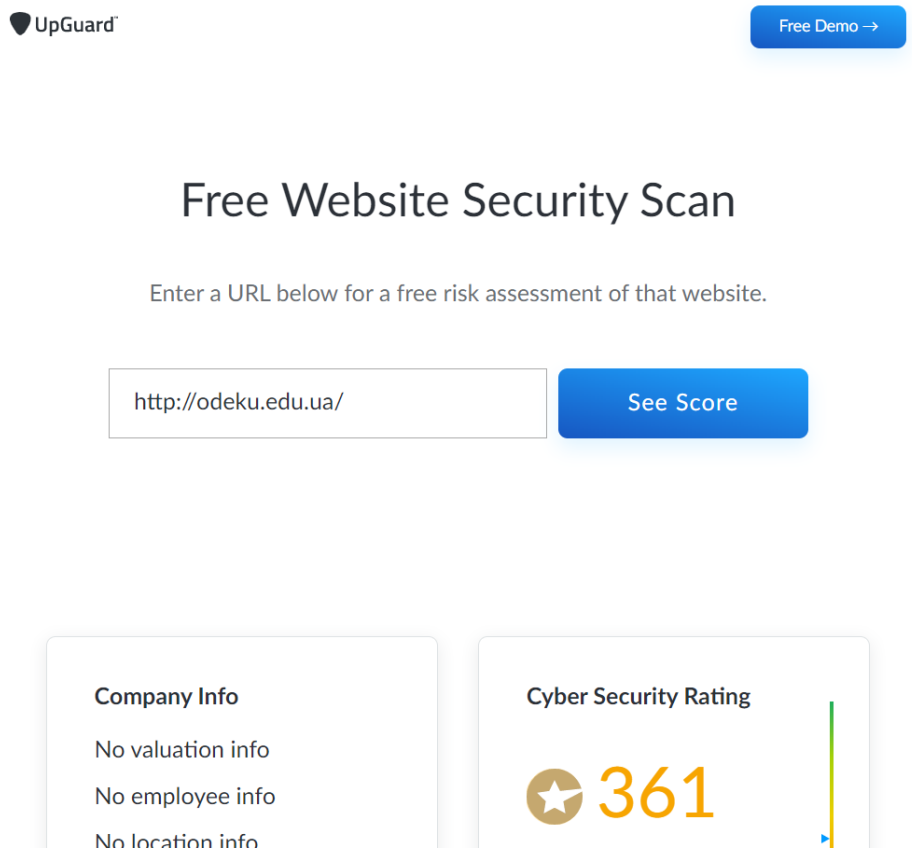


Рисунок 1.3 – Сайт сканування безпеки веб-сайту «UpGuard»

Браузерний додаток перевірки безпеки «McAfee»:

– головна: тут можливо знайти головну сторінку з переліком сторінок сайту та короткий опис сайту;
– вкладки з інформацією безпеки: розміщується інформація про перевірку, список посилань та коментарі;
– інформація: опис дослідженого URL;
– проекти: тут знаходиться проекти в яких бере участь компанія та їх описом;
– деталі безпеки: тут знаходиться короткий опис який містить коментарі про захист сайту;

- перевірка на фішинг: повідомляє, про те що даний сайт не намагається вкрасти ваші дані та стверджує, що він йому можно довіряти інформацію;
- відповідність безпеці: перевірка внутрішнього складу веб-ресурсу на небезпечні скрипти;
- досвід: опис накопичених робіт;
- дійсний сертифікат SSL: сповіщає, що дані, які користувач надсилаєте на цей сайт, шифруються стандартними протоколами безпеки.

Короткий опис додатку:

Дизайн додатку «McAfee» простий та зручний. Головна вікно містить коротку інформацію про вміст усього сайту для користувачів. На головному вікні немає нічого зайвого. Перелік пропонованих програм розташован дуже компактно, і за рахунок цього можна побачити якомога більше інформації за раз. Це дуже зручно і економно за часом. Відображає сертифікати безпеки. До недостатків я можу віднести повільну швидкість та постійний перехід до сайту додатку після натискання на вкладки меню – не можна подивитися інформацію у самому додатку. Верхнє меню має гарній дизайн та колір. Перелік пропонованих результатів дослідження сайту має короткий, але розкритій опис додатку (рис. 1.4). На сторінці немає функції вибору мови крім англійської, що ускладнює розуміння для користувача.

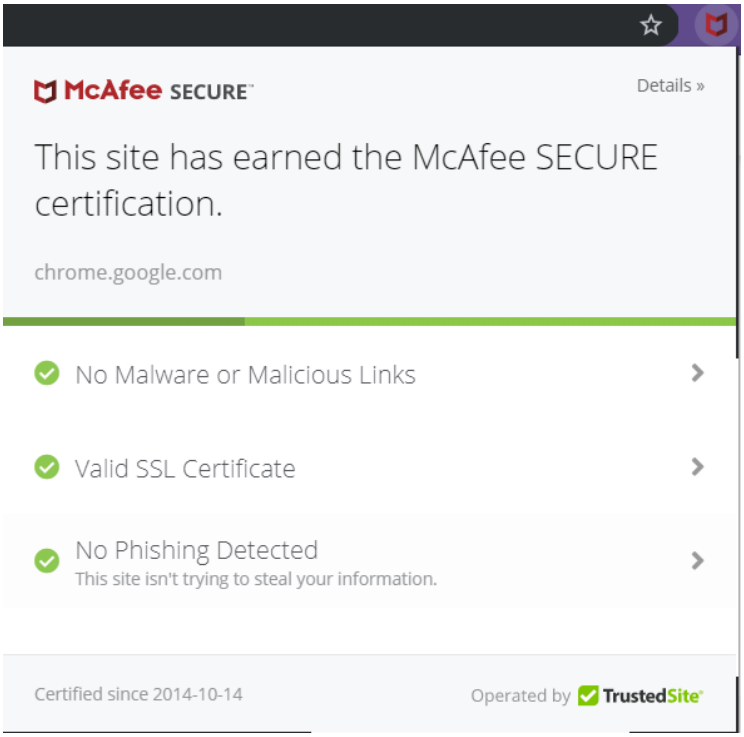


Рисунок 1.4 – Браузерний додаток перевірки безпеки «McAfee»

1.3 Постановка завдання

На основі проведеного дослідження подібних інформаційних систем були сформульовані вимоги до проектованої системі.

Необхідно створити інформаційну систему та додаток для веб-браузерів з безпеки ІС, який буде призначений для інформування користвача про безпечність сайту. Вимоги до інформаційної системи:

– ІС повинна надавати основну інформацію про типи та протоколи захисту, сертифікати та алгоритми кріптографічного захисту

– мати зрозумілий інтерфейс користувача

– надавати можливість виставити оцінку та коментар

– простота пошуку необхідної інформації

Основний принцип веб-дизайну – це створення зручної й практичної системи, здатної привернути увагу користувача до змісту. Для досягнення цього принципу необхідно дотримуватися чотири базових правила:

– простота читання;
– простота навігації і пошуку;
– узгодженість в макеті і дизайні;
– висока швидкість завантаження;
– кросплатформеність.

Правило перше: простота читання.

Це правило зводиться до того, що при будь-яких умовах відвідувач не перебував на вашому сайті, йому повинно бути там зручно. Необхідно, щоб кожен елемент на сторінці був чітко видно і опубліковано в більшості браузерів, як з включеною, так і з виключеною графікою. Яскраві кольори – це, що без сумніву має привертати увагу людини. Але, в той же час, яскраві кольори можуть і розсівати увагу, як у наведеному вище прикладі.

Тому тут важливо дотримати баланс і не перегинати палицю. При продумуванні колірного рішення потрібно враховувати корпоративний стиль і кольори вашої компанії, відштовхуйтесь від брендбуку, якщо такий є. Як би там не було, для створення фону остерігайтеся використовувати – не німий не буде видно тексту. М'який стильний фон – це те, що потрібно, він є чудовою основою для розміщення привертають увагу елементів. Якщо сумніваєтеся, краще залиште фон сторінки просто білою.

Яскраві кольори більше підходять для того, щоб звернути увагу користувача на якісь важливі моменти, наприклад, це можуть бути посилання.

Правило друге: простота навігації та пошуку.

Вище ми вже торкалися ці поняття. Гарна навігація – це інтуїтивно проста навігація, яка полягає в тому, що відвідувач завжди знайде ту інформацію, яку шукає. На всіх сторінках повинна знаходитися посилання на карту сайту і існувати можливість виходу на головну сторінку. Правило третє: узгодженість дизайну і макета.

HTML код і скрипти – це те, що включає в себе макет. Правильно складений макет – його логічність – це запорука того, що користувач буде отри-

мувати задоволення від роботи з сайтом, вільно і легко орієнтуючись в його розділах.

Як було сказано вище, дизайн – це, перш за все, графіка. Якщо на сайті присутні графічні елементи, що повторюються в різних розділах, то вони повинні бути виконані в єдиному стилі. А всі об'єкти, розташовані на одній сторінці повинні бути візуально взаємозв'язані.

Якщо завдяки узгодженості елементів дизайну, структура сайту чітко видна користувачеві – завдання виконане.

Правило чотири: швидке завантаження.

Дане правило зводиться до того, що необхідно з повагою ставиться до користувача вашого сайту – економте його час, не випробовуйте терпіння. При модемном підключенні сторінка сайту не повинна вантажиться довше 30 секунд. Домогтися цього можна, якщо не перезавантажувати сайт графічними елементами і анімацією. Застосовувати анімацію потрібно тільки для залучення уваги до самих основних розділів сайту.

Розширення – це невеликі програмні програми, які налаштовують перегляд веб-сторінок. Вони дають можливість користувачам налаштовувати функціональність та поведінку браузеру під індивідуальні потреби чи вподобання. Вони побудовані на веб-технологіях, таких як HTML, JavaScript та CSS.

Розширення повинно виконувати єдину мету, вузько визначену та зрозумілу. Одне розширення може включати в себе декілька компонентів і різноманітний функціонал, якщо все сприяє загальній меті.

Користувацькі інтерфейси повинні бути мінімальними та мати наміри. Вони можуть варіюватися від простої піктограми, до переосмислення всієї сторінки.

Файли розширень стискаються в єдиний crx пакет, який користувач завантажує та встановлює. Це означає, що розширення не залежать від вмісту в Інтернеті, на відміну від звичайних веб-додатків.

Розширення поширюються через інформаційну панель розробника і публікуються у веб-магазинах різних браузерів та проходять перевірку модерацією [3][1)].

1.4 Етапи розробки сайту

На першому етапі розробки і створення сайту здійснюється визначення цілей та завдань. А також визначити які будуть критерії досягнення цілей.

Після того як цілі та завдання визначені починається планування сайту. Формуються основні ідеї та потребі сайту. Велика увага приділяється функціоналу проекту, який описується до дрібниць, а також навігації сайту.

Коли планування сайту закінчено – переходять до реалізації сайту. В реалізації сайту є багато етапів. Спочатку відбувається створення дизайну. Логотип завжди розробляється ще до початку відтворення концепції дизайну головної сторінки. Це відбувається тому, що без урахування корпоративної символіки не можна створити по-справжньому оригінальний і цілісний дизайн. Зазвичай, концепція розробляється одна, але при бажанні можливо запропонувати на розгляд кілька варіантів. За концепцією робляться презентації, метою яких є донести до клієнта причини того чи іншого вибору і рішення, зробленого дизайнерами, пояснити, в чому його вигода для вебсайту. Далі відбувається способи виділення меню, спливаючі вікна, що випадають списки. Після доопрацювань відбувається затвердження концепції дизайну головної сторінки, і дизайнери приступають до створення концепцій внутрішніх сторінок, які також проходять етапи узгодження, доопрацювання та затвердження (рис. 1.5).

[1)] [3] What are extensions? – Google Chrome. URL: https://developer.chrome.com/extensions (дата звернення 3.12.2019).

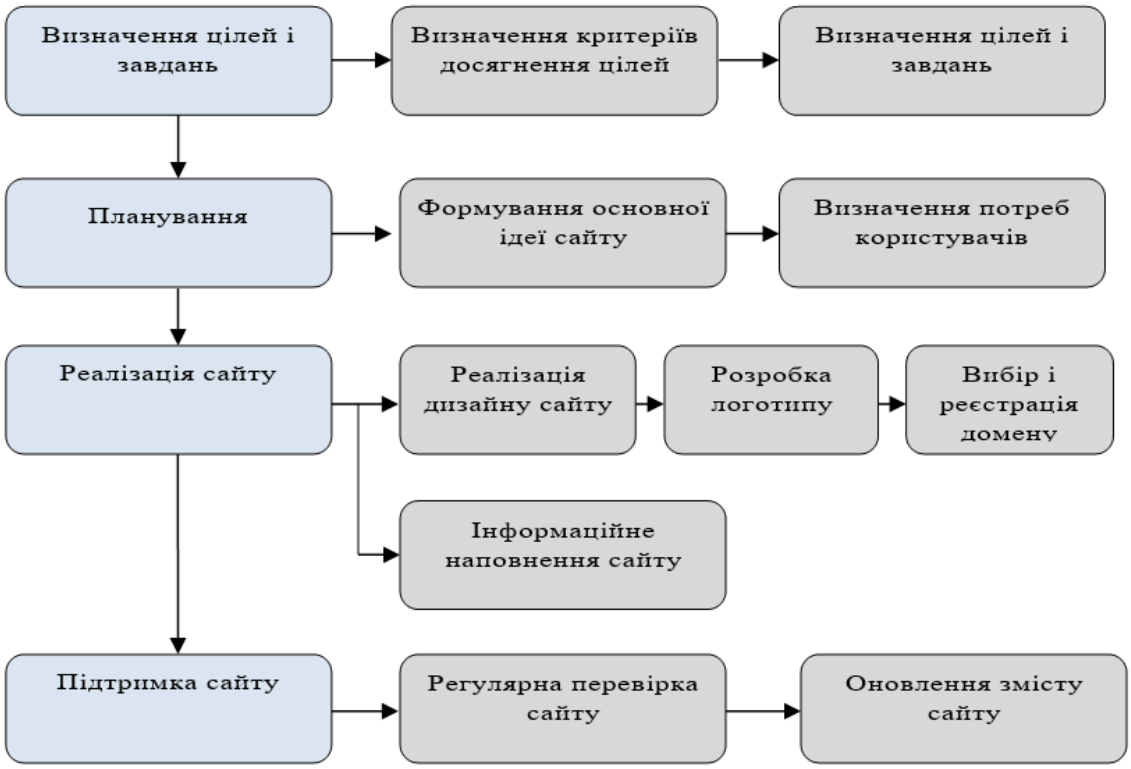


Рисунок 1.5 – Етапи розробки сайту Safeshell

Коли дизайн сайту завершений, потрібно визначитися з інформаційним наповненням сайту. Не тільки дизайн повинен бути оригінальним та унікальнім, інформація, також, повинна бути певною мірою унікальною, щоб привернути увагу відвідувачів, тим більше, що в інтернеті існує безліч схожих Web-сторінок і конкуренція між ними досить сильна. Інформація повинна бути оперативною. Для підтримки інтересу до сайту його необхідно постійно оновлювати і модернізувати. Чим частіше буде відбуватися оновлення інформації на сервері – тим вище буде інтерес і, відповідно, відвідуваність сайту.

Вибір фреймворку інтерфейсів користувача. Щоб створювати добре функціонуючі, масштабовані і стійкі веб-додатки, потрібно застосовувати технологічні тренди, останні і найвидатніші фреймворки, а також бути в курсі справ конкурентів на ринку. Розробляючи інформаційну систему, веб-розробка повинна підтримуватися кращими і останніми фреймворками, щоб прискорити виробничий процес і встигати до закінчення проектів. Досить

важко розпізнати, який з фреймворків стане основною тенденцією в майбутньому, щоб привести своїх рішення по розробці у відповідність з конкурентами на ринку.

Важливо передбачити, чи буде фреймворк відповідати вимогам розробника, чи є він зручним для відвідувачів проекту, а також чи здатен він забезпечити потрібний рівень функціоналу для різних видів інтернет ресурсів. Наступний крок вибір і реєстрація домену.

Коли інформаційна система реалізована, її потрібно підтримувати. Треба регулярно перевіряти сайт та оновлювати зміст сайту. Це потрібно для того що на сайті була завжди правдива інформація та свіжі новини.

1.5 Етапи розробки веб-додатку

На самому базовому рівні веб-додаток – це набір HTML, CSS і JavaScript-файлів, що дозволяє додати деяку функціональність в браузер через JavaScript API, який він надає. По суті, розширення – це веб-сторінка в Chrome, що має доступ до деяких додаткових API.

За рахунок того що цей пакет містить всі необхідні файли, розширення не залежить від ресурсів з інтернету і здатний коректно працювати навіть в оффлайн режимі.

Перше, що я повиннен зробити, це створити проект і всі файли, які потрібні для мого додатку. Всі розширення Chrome вимагають наявності файлу маніфесту (рис. 1.6). Файл маніфесту повідомляє браузеру все, що потрібно для завантаження розширення. Мій інтерфейс буде простим, що містить заголовок «Safeshell» і кнопку, по якій користувач зможе проаналізувати поточну сторінку. Щоб розширення виглядало гарніше і було зручніше, потрібно додати стилі на CSS.

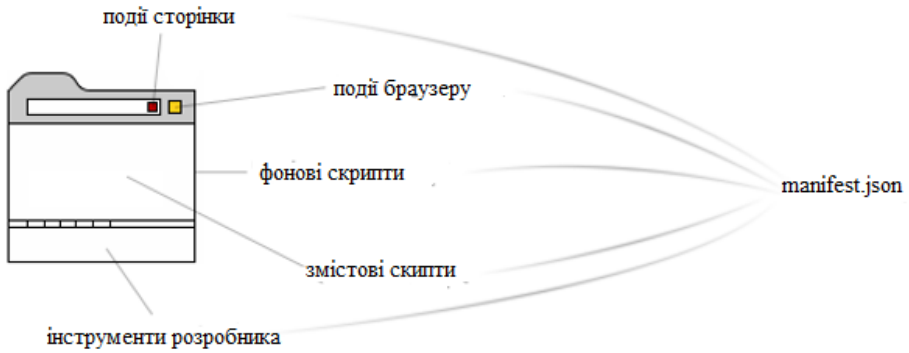


Рисунок 1.6 – Структура веб-додатку

Для Chrome розширень, діє так зване Content Security Policy – це набір строгих правил, які необхідні для того щоб зробити розширення безпечніше і контролювати контент який може бути завантажений і виконаний в розширенні.

За замовчуванням, якщо використовувати маніфест другої версії то в розширенні будуть такі обмеження:

– заборонено використовувати eval і схожі функції;

– лінійний JavaScript не повинен виконуватися;

– можливість завантажувати тільки локальні скрипти і ресурси.

Ці правила можна скасувати при необхідності. Наприклад, можна додати налаштування хоста з якого можливо отримувати певні ресурси або скрипти, все ще заборонено використовувати протокол HTTP тільки HTTPS.

Після розробки додатку потрібно його завантажити в браузер та пройти модерацію.

2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ СИСТЕМИ

2.1 Вибір фреймворку для створення інтерфейсів користувача

2.1.1 Опис фреймворку AngularJS

Angular – JavaScript-фреймворк з відкритим вихідним кодом. Він призначений для розробки односторінкових додатків. Його метою є розширення браузерних додатків на основі MVC-шаблону, а також спрощення тестування і розробки [4][1)].

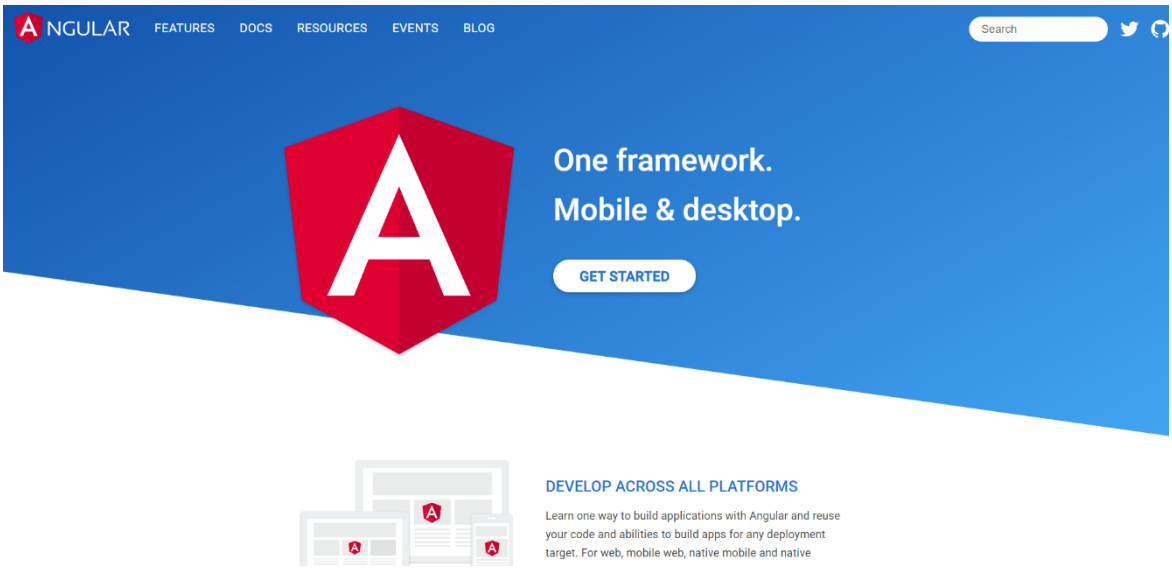


Рисунок 2.1 – Головна сторінка фреймворку Angular

Переваги використання Angular як фреймворку для сайту:

– Angular може приймати рішення і пропонує розробникам параметри за замовчуванням для таких речей як мережеве з'єднання, адміністративне управління станами, вибір мови, набір інструментів конфігурації;
– підтримка популярних браузерів;

[1)] [4] AngularJS – Википедия. URL: https://uk.wikipedia.org/wiki/AngularJS (дата звернення 3.12.2019).

– команда Angular прагне до стабільного і планомірного розвитку і дотримується концепції публічного анонсування розкладу релізів, що дозволяє компаніям робити висновки і складати плани на майбутнє з приводу змін в платформі;
– Angular має потужну підтримку. В інтернеті існує безліч багаторазово використовуваних інструментів, бібліотек і прикладів коду для Angular;
– це добре знайомий фреймворк. Розробники, які використовують Angular, в основному керуються двома міркуваннями. Перше – деякі з них уже мають досвід роботи з AngularJS. Друге міркування відноситься до розробників, які перейшли з Java або C # .NET. Обидві мови спираються на типи, і ключовим поняттям є додаток – дуже схоже на архітектуру Angular;
– універсальність Angular дозволяє створити сайт будь-якої складності і змісту з мінімальним використанням додаткових інструментів і розширень.

Недоліки використання Angular як фреймворку для сайту:

– низька швидкість роботи;
– для деяких проектів AngularJS надто зайвий. У таких випадках більше підійдуть більш легкі фреймворки;
– Angular також не підтримує високонавантажені галереї фото;
– складність освоєння;
– AngularJS дуже «упертий» і нав'язує свою структуру розробникам.

2.1.2 Опис фреймворку Vue.js

Vue.js – JavaScript фреймворк з відкритим вихідним кодом для створення користувацьких інтерфейсів. Легко інтегрується в проекти з використ-

танням інших JavaScript бібліотек. Може функціонувати як веб-фреймворк для розробки односторінкових додатків в реактивному стилі [5][1)].

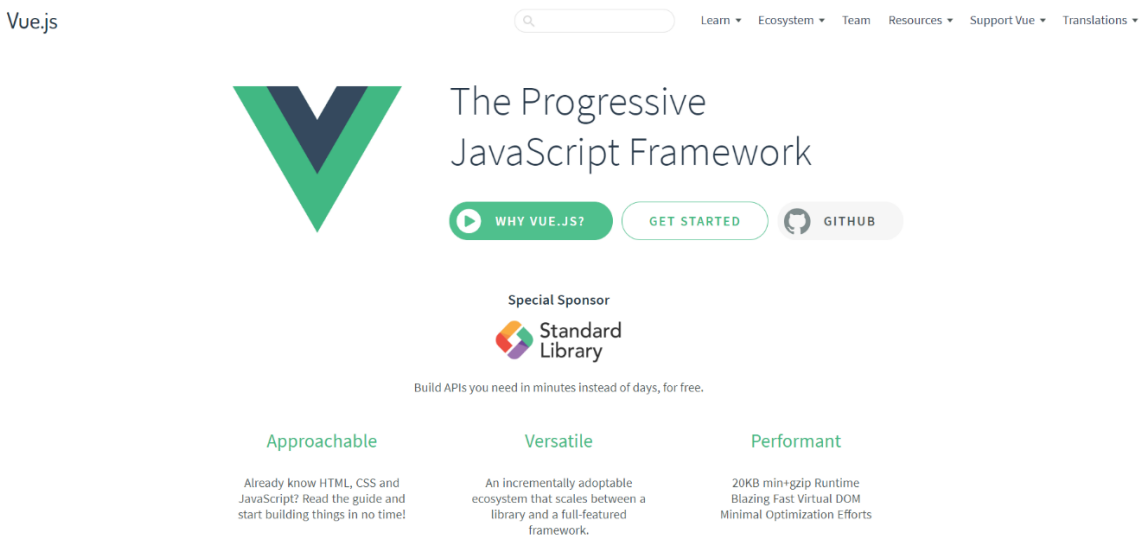


Рисунок 2.2 – Головна сторінка фреймворку Vue.js

Переваги використання Vue.js як фреймворку для сайту:

– велике співтовариство;
– посилений HTML. Це означає, що Vue.js має багато характеристик схожих з Angular, а це, завдяки використанню різних компонентів, допомагає оптимізації HTML блоків;
– детальна документація. Vue.js має дуже детальну документацію, яка може прискорити процес навчання для розробників і заощадити багато часу на розробку програми, використовуючи тільки базові знання HTML і JavaScript;
– усуває деякі невідповідності між браузерами;
– корисно для вибору і ітерації над наборами вузлів DOM;
– підтримка різних браузерів;

[1)] [5] Vue.js – Википедия. URL: ru.wikipedia.org/wiki/Vue.js (дата звернення 5.12.2019).

– малий розмір плагинів;
– масштабування. Vue.js може допомогти в розробці досить великих шаблонів багаторазового використання, які можуть бути зроблені майже за той же час, що і більш прості;
– легко для недосвідчених розробників JavaScript виконувати відносно складні завдання сценаріїв.

Недоліки використання Vue.js як бібліотеки для сайту:

– регулярні оновлення, які змінюють існуючу поведінку;
– недолік ресурсів. Vue.js як і раніше займає досить невелику частку ринку в порівнянні з React або Angular, що означає, що обмін знаннями в цьому середовищі все ще перебуває на початковій стадії.
– якість плагінів дуже мінлива;
– ризик надмірної гнучкості. Іноді у Vue.js можуть виникнути проблеми при інтеграції в величезні проекти, і поки ще немає досвіду можливих рішень, але вони обов'язково з'являться найближчим часом;
– продуктивність зазвичай набагато повільніше, ніж еквівалентний (добре продуманий) код без jQuery.

2.1.3 Опис бібліотеки React

React – відкрита JavaScript бібліотека для створення інтерфейсів користувача, яка покликана вирішувати проблеми часткового оновлення вмісту веб-сторінки, з якими стикаються в розробці односторінкових застосунків. React значно полегшує створення інтерфейсів завдяки розподіленню кожної сторінки на невеликі фрагменти. Розробляється Facebook, Instagram і спільнотою індивідуальних розробників [6][1)].

[1)] [6] React – Википедия. URL: http://www.wikiwand.com/uk/React (дата звернення 5.12.2019).

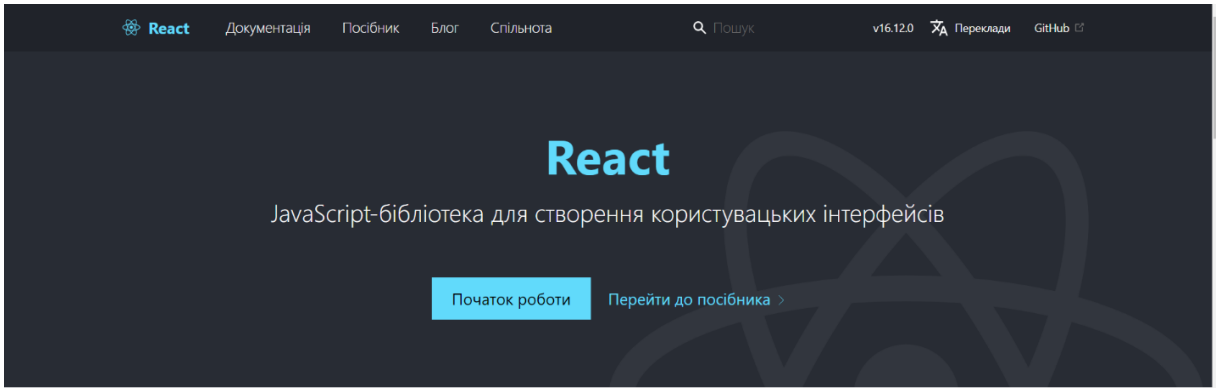


Рисунок 2.3 – Головна сторінка фреймворку React

React є ідеальним інструментом для створення SPA, орієнтованої на естетику, підтримку стандартів і зручність використання.

2.1.4 Загальна схема роботи фреймворку React

React надає мову шаблонів і деякі callback-функції для відтворення HTML. Весь результат роботи React – це HTML. Зв'язки HTML та JavaScript, що називаються компонентами, займаються тим, що зберігають свій внутрішній стан в пам'яті (наприклад: яка закладка обрана), але в підсумку просто відає користувачу HTML. Всі компоненти, маленькі чи великі, можуть використовуватися повторно, навіть у різних проектах та можуть змінюватися програмистом.

На відміну від document.createElement, createElement React приймає необмежену кількість аргументів після другого аргументу, для подання дочірніх елементів створюваного елемента. Таким чином createElement фактично створює дерево.

Прості функціональні компоненти відмінно підходять для простих потреб, але іноді потрібно більше. React підтримує створення компонент за допомогою синтаксису класів JavaScript та спеціальних функціональних компонентів.

Всі атрибути елементів React (включаючи події) називаються за допомогою camelCase, а не в нижньому регістрі. Це onClick, а не onclick. React обгортає об'єкт події DOM в власний об'єкт який має назву SyntheticEvent, щоб оптимізувати продуктивність обробки подій. Але всередині обробника подій ми все одно можемо отримати доступ до всіх методів, доступним в об'єкті події DOM. React передає обгорнутий об'єкт події для кожного виклику обробника.

Функціональні компоненти мають таку історію:

- спочатку ми визначаємо шаблон React для створення елементів з компонента;
- потім потрібно вказати React де будемо його використовувати. Наприклад, всередині виклику функції render іншого компонента або за допомогою ReactDOM.render;
- потім React створює екземпляр елемента і передає йому набір props, до яких ми можемо отримати доступ за допомогою this.props. Ці props – це те, що ми передали на кроці 2;
- оскільки це все JavaScript, то буде викликаний метод конструктора (якщо він визначений). Це перший метод з тих, що ми називаємо методами життєвого циклу компонентів;
- потім React обробляє результат виклику функції render (отримує віртуальний вузол DOM);
- оскільки це перший раз, коли React виконує рендеринг елемента, React буде взаємодіяти з браузером (від нашого імені, використовуючи DOM API), щоб відобразити в ньому елемент. Цей процес широко відомий як монтування;
- потім React викликає інший метод життєвого циклу, званий componentDidMount. Ми можемо використовувати цей метод, щоб, наприклад, зробити щось в DOM, який, як ми знаємо, існує в

браузері. До цього методу життєвого циклу, DOM, з яким ми працювали, був віртуальним;

– стан будь-якого змонтованого елемента може змінитися. Батько цього елемента може бути повторно відмалювали.

В процесі роботи компонент проходить через ряд етапів життєвого циклу (додаток А). На кожному з етапів викликається певна функція, в якій ми можемо визначити будь-які дії:

– constructor(props): конструктор, в якому відбувається початкова ініціалізація компонента;
– componentWillMount(): викликається безпосередньо перед рендерингом компонента;
– render(): рендеринг компонента;
– componentDidMount() : викликається після рендеринга компонента. Тут можна виконувати запити до віддалених ресурсів;
– componentWillUnmount(): викликається перед видаленням компонента з DOM.

Крім цих основних етапів або подій життєвого циклу, також є ще ряд функцій, які викликаються при оновленні стану після рендеринга компонента.

2.1.5 Переваги React перед іншими фреймворками

– розмір фреймворку: Очевидно що даний показник не може бути вирішальним. Але все ж потрібно визнати той факт що ядро react та flux займає у три рази менше Angular. Звичайно існує набір функціоналу якого немає в ядрі react;
– швидкість роботи: react завдяки підходу з віртуальні DOM і jsx значно випереджає конкурентів. Для мене цей фактор став вирішальним;

– філософія: React на відміну від Ember чи Angular створює враження програми, яка робить щось одне, але роблять це добре.
– односпрямований потік даних: Кожен хто створював повноцінне великий (ні hello world) додаток на Angular чи Ember так чи інакше стикалися з проблемою потоку даних і зв'язків сховище-модель-уявлення (а також їх комбінацією). З аналогічною проблемою зіткнулися і інженери facebook, результатом їх роботи став flux. Це рішення справляється зі своїм завданням;
– компіляція в html на сервері: Величезна перевага рішень на базі react. Так існує Angular-server але він не повноцінний, і працює з певними проблемами чи обмеженнями. Що стосується React, то існує можливість і інструменти які її реалізують видавати повністю автономну HTML сторінку яка не вимагає повторної компіляції на клієнті що ще й ускорює процес візуалізації. Власне це дозволяє додатку залишатися частково робочим навіть з вимкненим або недоступним JavaScript на клієнті [7][1)];

2.1.6 Порівняння фреймворків

Фреймворки JavaScript розвиваються дуже швидкими темпами, тому сьогодні у нас є часто оновлювані версії Angular і ReactJS, а також версії нового гравця на цьому ринку - Vue.js.

З найбільш знаменитих фреймворків вибрано всього три: Angular, Vue.js і React. Кожен з них має свої плюси і мінуси. Для створення ІС вибрано фреймворк React, так як є ряд плюсів які кращі:

– простий у вивченні. React набагато легше вчиться зважаючи на простоту його синтаксису. Інженери просто повинні згадати свої навич-

[1)] [7] Преимущества react. Блог Новикова Богдана. URL: https://hcbogdan.com/frontend/2016/02/14/react-benefits (дата звернення 6.12.2019).

ки написання HTML і все на цьому. Немає потреби в глибокому вивченні TypeScript, як у випадку з Angular;

- високий рівень гнучкості і максимальна чуйність;
- віртуальна DOM (document object model), яка дозволяє впорядковувати документи форматів HTML, XHTML або XML в дерево, яке найкраще підходить веб-браузерів для аналізу різних елементів веб-додатки;
- зрозумілий синтаксис;
- відкритий вихідний код;
- відмінна документація;
- у поєднанні з ES6 чи 7 ReactJS може легко працювати при високих навантаженнях;
- легка установка;
- простий в використуванні;
- неймовірно легка вага, так як дані, які виконуються на стороні користувача, можуть легко бути представлені на стороні сервера в той же самий час.

Компанії, які використовують ReactJS: Facebook, Instagram, Netflix, New York Times, Yahoo, Khan Academy, Whatsapp, Codecademy, Dropbox, Airbnb, Asana, Atlassian, Intercom, Microsoft.

2.2 Переваги використання FireBase

Перевагами платформи мобільних та веб-застсунків FireBase є підтримка спільного хостингу, бази даних, сховища мультимедійних типів даних, систему аутентифікації для веб та мобільних додатків та обмін повідомлень (додаток Б).

Сервіс надає можливості керувати аутентифікацією на різних платформах, переглядати зареєстрованих користувачів та надавати окремі

права. Він підтримує соціальні логін-провайдери Facebook, GitHub, Twitter і Google (і Google Play Games)

Firebase надає в режимі реального часу базу даних та бекенд як службу. Ця служба надає розробникам застосунків API, який дозволяє синхронізувати дані застосунків між клієнтами та зберігати їх у хмарі Firebase.

Компанія також надає клієнтські бібліотеки, які дозволяють інтеграцію із застосунками Android, iOS, JavaScript / Node.js, Java, Objective-C, Swift. База даних доступна через REST API та прив'язки до декількох сценаріїв JavaScript, таких як AngularJS, React, та Backbone.js. Розробники, які використовують Realtime Database, можуть захищати свої дані за допомогою правил безпеки, що застосовуються на сервері.

Firebase Storage забезпечує надійне завантаження та вивантаження файлів для застосунків Firebase незалежно від якості мережі. Розробник може використовувати його для зберігання зображень, аудіо, відео чи іншого вмісту, створеного користувачами. Зберігання Firebase підтримується Google Cloud Storage.

Firebase Hosting – це статичний та динамічний веб-хостинг. Він підтримує хостинг статичних файлів, таких як CSS, HTML, JavaScript та інші файли, а також динамічну підтримку Node.js через Cloud Functions. Служба передає файли через мережу доставки контенту (CDN) за допомогою протоколу HTTPS та шифрування SSL. Firebase підтримує Fastly, CDN, щоб забезпечити підтримку CDN Firebase Hosting [8][1)].

2.3 Особливості використання баз даних FireBase Cloud FireStore

Оскільки сайт написано з допомогою платформи FireBase, база даних якій подається разом с сервісом є найкращим вибором для інформаційної си-

[1)] [8] Firebase – Википедия. URL: https://uk.wikipedia.org/wiki/Firebase (дата звернення 6.12.2019).

стеми. Програми офлайн Firebase залишаються чутливими навіть у режимі офлайн, оскільки Firebase Realtime Database SDK зберігає дані.

Cloud Firestore – це гнучка, масштабується хмарна база даних від Firebase і Google Cloud Platform для інтернету, мобільних платформ, і серверних додатків. Як і Firebase Realtime Database , вона синхронізує дані між клієнтськими додатками за допомогою слухачів реального часу а також пропонує підтримку оффлайн режиму для мобільних платформ і веба.

Переваги використання FireBase Cloud FireStore як бази даних для сайту:

- база даних реального часу: Замість типових HTTP-запитів в базі даних Firebase Realtime використовується синхронізація даних-кожен раз, коли дані змінюються, будь-який приєднаний пристрій отримує це оновлення в мілісекунди;
- офлайн: Програми офлайн Firebase залишаються чутливими навіть у режимі офлайн, оскільки Firebase Realtime Database SDK зберігає дані на диску. Після відновлення з'єднання клієнтський пристрій отримує будь-які зміни, які він пропустив, синхронізувавши їх з поточним станом сервера;
- швидке реагування на запити в Cloud Firestore;
- виразні запити. У Cloud Firestore, ви можете використовувати запити для отримання одного документа, визначеного документа, або всіх документів в колекції, які відповідають параметрам вашого запити. Запити можуть можуть містити ланцюжка з декількох фільтрів або фільтри комбіновані з сортуванням. Також всі колекції індексуються за замовчуванням, з цього продуктивність запиту прямо пропорційна результату, а не розміром колекції;
- доступно з клієнтських пристроїв: Базу даних Firebase Realtime можна отримати безпосередньо з мобільного пристрою або веб-браузера;

немає необхідності для сервера додатків. Забезпечення безпеки та перевірка даних доступні через правила безпеки Firebase Realtime Database Security, правила на основі виразів, які виконуються під час читання або запису даних;

- підтримка офлайн режиму. Cloud Firestore кеширует дані, які додатки часто використовують, з цього додаток може писати, читати, слухати поновлення і робити запити навіть коли девайс знаходиться в офлайні. Коли девайс повернеться в онлайн Firestore всі локальні зміни в хмару;
- модель даних Cloud Firestore підтримує гнучкі, ієрархічні структури даних. Зберігає дані в документах, які в свою чергу зберігаються в колекціях. Документи можуть вкладення об'єкти і підколекції;
- firestore надає використовує потужну інфраструктуру Google Cloud Platform: автоматичну мультирегіональну реплікацію даних, надійні гарантії цілісності, атомарні batch операції, підтримку транзакцій;
- як і Realtime Database, Cloud Firestore використовує синхронізацію даних для поновлення даних на кожному підключеному пристрої;
- масштабування в декількох базах даних: За допомогою бази даних Firebase Realtime можливо підтримувати потреби додатка в масштабі, поділивши свої дані на декілька екземплярів баз даних в одному проекті Firebase. Спростити аутентифікацію за допомогою перевірки автентичності Firebase у проекті та автентифікація користувачів у різних примірниках бази даних.

2.4 Особливості використання серверу баз даних MYSQL

Оскільки система написана з допомогою системи управління вмістом WordPress, база даних якій подається як MySQL, база даних сайту підтримуватиме таку ж структуру, вносячи у неї свої зміни.

Система керування реляційними базами даних MySQL була розроблена компанією «TcX» для підвищення швидкодії обробки великих баз даних. Ця система керування базами даних (СКБД) з відкритим кодом була створена як альтернатива комерційним системам. MySQL з самого початку була дуже схожою на mSQL. У 1999 році MySQL обігнала mSQL за популярністю. Остання версія mSQL-3.8 – була випущена 9 червня 2006.

MySQL виникла як спроба застосувати mSQL до власних розробок компанії: таблицям, для яких використовувалися ISAM – підпрограми низького рівня. У результаті був вироблений новий SQL-інтерфейс, але API-інтерфейс залишився в спадок від mSQL [9][1)].

З часом MySQL все розширювалася і зараз вона – одна з найпоширеніших систем керування базами даних.

Вона використовується, в першу чергу, для створення динамічних вебсторінок, оскільки має впевнену підтримку з боку різноманітних мов програмування.

Станом на квітень 2009 року, MySQL пропонувала версію MySQL 5.1 в двох різних варіантах: з відкритим вихідним кодом MySQL Community Server і комерційний Server Enterprise. Вони мають спільний програмний код і включають в себе серед іншого наступні можливості:

- крос-платформна підтримка;
- збережені процедури та функції;
- тригери;
- курсори;
- оновлюванні подання;
- інформаційна схема (так званий системний словник, що містить метадані);
- підтримка SSL;

[1)] [9] Разработка схемы классов. URL: https://www.intuit.ru/studies/courses/4806/1054/lecture/16134?page=2 (дата звернення 6.12.2019).

– кешування запитів;
– вкладені запити SELECT;
– підтримка реплікації;
– повноцінна підтримка Юнікоду (UTF-8 і UCS2);
– сегментування таблиць;
– тощо.

Переваги та недоліки MySQL.

MySQL являється компактним багатопоточним сервером баз даних та характеризується великою швидкістю, стійкістю і простотою використання.

Ця система вважається гарним рішенням для малих і середніх додатків. Вихідний код сервера компілюється на безлічі платформ. Найбільш повно можливості сервера виявляються в UNIX-системах, де є підтримка багатопоточності, що підвищує продуктивність системи в цілому. Для некомерційного використання MySQL є безкоштовним.

Можливості сервера MySQL:

– простота у встановленні та використанні;
– підтримується необмежена кількість користувачів, що одночасно працюють із БД;
– кількість рядків у таблицях може досягати 50 млн;
– висока швидкість виконання команд;
– наявність простої і ефективної системи безпеки.

Недоліками сервера MySQL залишається не реалізована підтримка транзакцій, замість якої пропонується використовувати LOCK/UNLOCK TABLE, а також відсутність підтримки для зовнішніх (foreign) ключів, тригерів, збережених процедур та представлень (VIEW). Та для розробки малих та середніх інформаційних систем, призначених для використання в межах робочих груп ці недоліки є фактично невідчутними.

Реалізація взаємодії MySQL та PHP.

Проекти на основі безкоштовного ПЗ, які вимагають повнофункціональної системи керування базами даних часто використовують MySQL. До таких проектів належать, наприклад, WordPress, phpBB, Drupal та інше програмне забезпечення, побудоване на стеку продуктів LAMP (Linux, Apache, MySQL, PHP/Perl/Python).

Популярний візуальний інтерфейс PhpMyAdmin, написаний на PHP, дозволяє працювати з MySQL в графічному режимі. Цей інтерфейс дозволяє значно спростити роботу з базами даних в MySQL.

У текстовому режимі робота з базою даних виглядає просто як введення команд у командний рядок, а результати вибірок повертаються у вигляді своєрідних таблиць, поля в яких налазять один на одного, якщо дані не вміщаються на екрані.

PhpMyAdmin дозволяє використовувати всі переваги роботи в браузері, включаючи прокрутку зображення, у випадку, коли йому для відображення бракує розмірів екрану. Більшість базових SQL-функцій роботи з даними в PhpMyAdmin зведені до інтуїтивно зрозумілих, завдяки інтерфейсу, дій, що нагадують перехід за посиланнями в мережі Інтернет.

Проте однозначною перевагою роботи в текстовому режимі залишається необмеженість функцій реалізованим інтерфейсом.

В дистрибутив самої мови PHP входить розширення, що містить вбудовані функції для роботи з базою даних MySQL. При роботі з веб-інтерфейсом для додавання інформації до бази даних куристувач повинен ввести ці дані в HTML-форму та відправити їх на сервер, а все інше виконається засобами скрипта. Для перегляду вмісту таблиць достатньо перейти по посиланню і зайти на потрібну сторінку.

Дуже часто помилки в скриптах із доступом до БД виникають у зв’язку з некоректно сформованим SQL-виразом. Тому першим кроком у налагодженні PHP-програми, що виконує запити до MySQL повинна бути перевірка правильності SQL-інструкцій. Для цього слід вивести SQL-запит на сторінку,

і, скопіювавши його, виконати безпосередньо в phpMyAdmin. Хоча SQL і замислювався як засіб роботи кінцевого користувача, врешті-решт він став настільки складним, що перетворився в інструмент програміста.

2.5 Переваги використання PHP

Перевагами середовища PHP є підтримка переважною більшістю хостинг-провайдерів, а також те, що PHP є проектом відкритого програмного забезпечення.

Код, створений на PHP інтерпретується веб-сервером в HTML-код, який передається на сторону клієнта.

Користувач не має можливості побачити PHP-код, оскільки браузер отримує вже готовий html-код. Що являється безумовною перевагою з точки зору безпеки програмного продукту, але погіршує інтерактивність сторінок. Саме тому для алгоритмів, які виконуються на стороні клієнта була використана мова JavaScript, що дозволило повністю уникнути проблем, створюваних вище вказаним недоліком PHP.

Код написаний мовою PHP має можливість бути вбудованим безпосередньо в html-код сторінок, які, в свою чергу коректно будуть оброблені PHP-інтерпретатором. Механізми PHP, для такого випадку, аналізують html-код на предмет наявності екрануючої послідовності, та після її знаходження починають виконувати розміщений далі php-код і продовжують виконання до того моменту, коли їм зустрінеться парна екрануюча послідовність [10][1)].

Одним з найважливіших плюсів мови PHP є велика різноманітність функцій, що дають програмістам можливість уникнути написання

[1)] [10] Знайомство з PHP (призначення, включення в документ, обробка, базові налаштування) – Вікі ЦДПУ. URL: https://wiki.cuspu.edu.ua/index.php/ (дата звернення 6.12.2019).

багаторядкових, призначених для користувача, функцій, як це виконували б середовища C або Pascal.

Серед ще не вказаних переваг можна вказати такі:

– через стандарт відкритого інтерфейсу зв'язку з базами даних (Open Database Connectivity Standard – ODBC) можна підключатися до всіх баз даних, до яких існує драйвер;
– PHP містить величезну кількість різних функцій, що позбавляє нас необхідності писати багаторядкові скрипти для виконання простого завдання.

PHP містить ряд готових бібліотек для роботи із популярними базами даних:

– безпека:
– традиційність;
– ефективність – є дуже важливим чинником при програмуванні для середовищ розрахованих на багато користувачів, до яких належить і web.

Поєднавши в собі переваги інших мов програмування та обравши для себе єдине основне спрямування, PHP стала універсальним, пристосованим до веб-програмування, зрозумілим та простим для вивчення засобом роботи в Інтернеті.

3 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Загальні вимоги до інформаційної системи

До автоматизованих робочих місць різних категорій користувачів пред'являється ряд загальних вимог. Вони повинні забезпечувати:

- введення даних з клавіатури;
- виведення на екран дисплея і друкувальний пристрій інформації в зручному для перегляду вигляді;
- передачу даних по мережі;
- кешуваня токенів доступу;
- обробку інформаційних запитів користувача з використанням довідників і класифікаторів;
- захист інформації від несанкціонованого доступу шляхом введення системи паролів.

При розробці інформаційної системи «Safeshell» визначені кілька категорій користувачів з певною групою інтересів, відповідальності і можливостей, а також правами доступу до інформації.

Перш ніж перейти до проектування бази даних, важливо встановити завдання досліджуваної системи і способи їх взаємодії. Необхідно визначити основні функціональні вимоги, які пропоновані до системи, тобто визначити діапазон завдань системи та програми бази даних, складу її користувачів і областей застосування.

В результаті розробки web-проекту «Safeshell» буде створена Internet-система з розподіленим доступом і інтерфейсом для різних категорій користувачів: Користувачів-Адміністратори, Користувачів-Зареєстровані користувачі та Користувачів-Гості.

Набір функцій для цих категорій може бути повністю різний, а може і перетинатися.

3.2 Функціональні можливості користувачів web-системи

Розмежування прав доступу в межах інформаційної системи здійснюється шляхом дозволу або заборони на перегляд певної інформації для груп користувачів. В інформаційній системі «Safeshell» захисту особистих даних існують такі групи користувачів як: Адміністратор, Зареєстрований користувач та Гість (рис. 3.1).

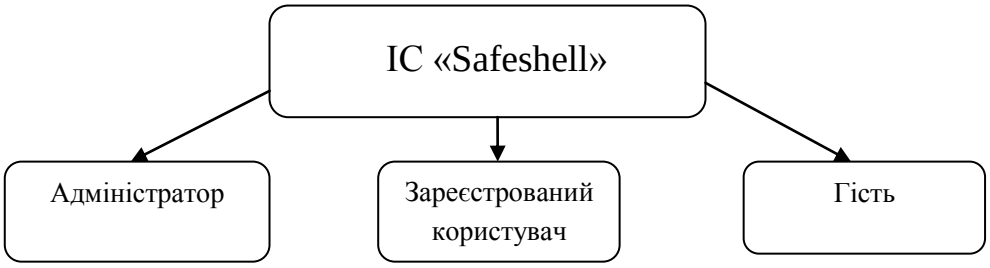


Рисунок 3.1 – Користувачі ІС «Safeshell»

У кожного користувача системи є свої права, які надає адміністратор. Залежно від наданих прав, користувачі мають або не мають можливості перегляду тієї чи іншої інформації в системі. Адміністратор розподіляє права логінами, які потім заносяться в базу даних системи (рис. 3.2).

Рисунок 3.2 – Функціональні можливості Адміністратора

Користувач – Адміністратор має основні права керування системою і привілеї в інформаційній системі захисту особистих даних. Він може обмежувати права користувачів, видаляти або додавати нового користувача, вирішувати ті чи інші дії всередині системи. Адміністратор без обмежень працює з базою даних, виконує в ній будь-які дії – видаляє або додає таблиці, редагує дані.

Користувачі – Зареєстровані користувачі, мають можливість безперешкодно авторизуватися в системі. Зареєстрованим користувачам доступна можливість додавати коментарі та виставляти оцінки над посиланнями (рис. 3.3).

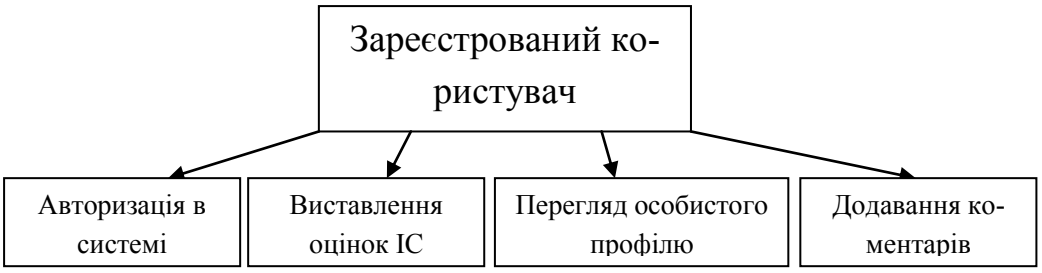


Рисунок 3.3 – Функціональні можливості Зареєстрованого користувач

Користувачі – Зареєстровані користувачі мають можливість переглядати особисту інформацію в профілі виставляти оцінки та писати коментарі.

Остання група користувачів – Користувач-Гість має можливість переглядати лише інформаційну частину системи. Йому недоступне розв’язування вправ і особистий профіль. Доступ до інформації Користувача – Гість носить ознайомлювальний характер (рис. 3.4). Адміністратор розподіляє права та має повний доступ до бази даних, виконує в ній будь-які дії – видаляє або додає таблиці, редагує дані.

Рисунок 3.4 – Функціональні можливості Користувача-Гістя

↓

Категорія користувачів Гість матимуть можливість отримання загальнодоступної інформації, що цікавить відвідувачів даного інформаційного ресурсу.

3.3 Проектування даталогічної моделі бази даних

Даталогічне проектування – створення схеми бази даних на основі конкретної моделі даних, наприклад, реляційної моделі даних. Для реляційної моделі даних даталогічна модель – набір схем відносин, зазвичай із зазначенням первинних ключів, а також «зв'язків» між відносинами, що представляють собою зовнішні ключі [11][1)].

Будь-яка СУБД оперує з допустимими для неї логічними одиницями даних, а також допускає використання певних правил композиції логічних структур більш високого рівня зі складових інформаційних одиниць нижчого рівня. Крім того, багато СУБД накладають кількісні та інші обмеження на структуру бази даних. Тому перш ніж приступити до побудови даталогічної моделі, необхідно детально вивчити особливості СУБД, визначити чинники, що впливають на вибір проектного рішення, ознайомитися з існуючими ме-

[1)] [11] Проектирование баз данных – Википедия. URL: https://ru.wikipedia.org/wiki/Проектирование_баз_данных (дата звернення 6.12.2019).

тодиками проектування, а також провести аналіз наявних засобів автоматизації проектування, можливості і доцільності їх використання.

Хоча даталогічне проектування є проектуванням логічної структури бази даних, на нього впливають можливості фізичної організації даних, що надаються конкретною СУБД. Тому знання особливостей фізичної організації даних є корисним при проектуванні логічної структури.

Логічна структура бази даних, а також сама заповнена даними база даних є відображенням реальної предметної області. Тому на вибір проектних рішень найбезпосередніший вплив надає специфіка предметної області.

Результатом даталогічного проектування є опис логічної структури бази даних на мові опису даних. Однак якщо проектування виконується «вручну», то для більшої наочності спочатку будується схематичне графічне зображення структури бази даних. При цьому повинно бути забезпечено однозначна відповідність між конструкціями мови опису даних і графічними позначеннями інформаційних одиниць і зв'язків між ними. Графічне представлення використовується і при автоматизованому проектуванні структури бази даних як частиною інтерфейсу засіб спілкування з проектувальником, і при документуванні проекту.

Спроектувати логічну структуру бази даних означає визначити всі інформаційні одиниці і зв'язку між ними, задати їх імена; якщо для інформаційних одиниць можливе використання різних типів, то необхідно визначити їх тип. Слід також задати деякі кількісні характеристики, наприклад довжину поля.

Кожному типу моделі даних і кожного різновиду моделі, яку підтримує конкретної СУБД, притаманні свої специфічні особливості. Разом з тим є багато спільного у всіх структурованих моделях даних і принципах проектування БД в їх середовищі. Все це дає можливість використовувати єдиний методологічний підхід до проектування структури бази даних.

В БД відображається певна предметна область. Тому процес проектування БД передбачає попередню класифікацію об'єктів предметної області, систематизоване представлення інформації про об'єкти і зв'язки між ними.

Дані про предметну область і особливості обробки інформації в ній фіксуються в інфологічної моделі. У такій моделі відображена вся інформація, що циркулює в інформаційній системі, але це зовсім не означає, що вся вона повинна зберігатися в базі даних. У зв'язку з цим одним з перших кроків проектування є визначення складу БД, тобто переліку тих показників, які доцільно зберігати в БД.

При проектуванні логічної структури БД здійснюються перетворення вихідної інфологічної моделі в модель даних, підтримувану конкретної СУБД, і перевірка адекватності отриманої даталогіченої моделі відображається предметної області.

Для будь-якої предметної області існує безліч варіантів проектних рішень її відображення в даталогіченої моделі. Методика проектування повинна забезпечувати вибір найбільш підходящого проектного рішення.

Зв'язки між сутностями предметної області, відображені в інфологічної моделі, можуть відображатися в даталогічній моделі або за допомогою спільного розташування відповідних їм інформаційних елементів, або шляхом оголошення зв'язку між ними.

Не всі види зв'язків, що існують в предметній області, можуть бути безпосередньо відображені в конкретній даталогічній моделі. Так, багато СУБД не підтримують безпосередньо відношення М: М між елементами. У цьому випадку в даталогічну модель вводиться додатковий допоміжний елемент, що відображає цей зв'язок (таким чином, ставлення М: М хіба що розбивається на два відносини 1: М між цим знову введеним елементом і вихідними елементами).

Мінімальна логічна одиниця даних семантично для всіх СУБД однакова і відповідає або ідентифікатором об'єкта, або властивості об'єкта або процесу.

Слід звернути увагу на те, що відносини, які мають місце в предметній області, можуть бути передані не тільки за допомогою структури бази даних, але і програмним шляхом (тобто завжди існує альтернатива між декларативним і процедурним способом опису явища). Наприклад, при відображенні узагальнених об'єктів можна не виділяти підкласи на рівні логічної структури бази даних. В цьому випадку підкласи будуть виділятися програмним шляхом при обробці даних, що зберігаються. Рішення про те, який із способів відображення (структурний чи декларативний або програмний чи процедурний) слід використовувати в кожному конкретному випадку, буде залежати від багатьох факторів, таких, як стабільність сутності, що відображається обсяг номенклатури, особливості СУБД, характер обробки даних та ін. Так, якщо в предметній області використовується класифікація співробітників по статі, в базі даних не слід створювати класифікатор статей, оскільки він буде містити всього дві позиції і ніколи не змінюється. Як правило, не слід виділяти за цією ознакою і відповідні підкласи для об'єктів предметної області, тому що в більшості випадків вони обробляються спільно і в основному мають однаковий набір властивостей, що характеризує їх. Однак в деяких предметних областях поділ на такі підкласи може бути доцільним. Якщо ж відображається сутність не стабільна, то її краще передавати за допомогою даних, так як в противному випадку буде часто турбуватися перетворення програми, що зазвичай забезпечити важче, ніж зміна даних.

При відображенні узагальнених об'єктів в БД можливі різні варіанти: зберігати всю інформацію про всі узагальнені об'єкті в одному файлі / таблиці, кожному підкласу об'єктів нижчого рівня виділяти окремі самостійні файли/таблиці. І перший, і другий варіанти широко використовуються у різноманітних СУБД. У першому випадку підкреслюється спільність об'єктів різних

підкласів, що входять в узагальнений об'єкт. У другому випадку, навпаки, узагальнений об'єкт як єдине ціле не відображається в структурі бази даних.

Інші способи відображення пов'язані з явним або неявним виділенням підкласів в логічній структурі БД. Неявне виділення підкласу полягає в тому, що в запису відводяться поля для фіксації значень властивостей, загальних для об'єктів різних підкласів, і значення ознаки підкласу, а замість полів, наявність яких залежить від підкласу, використовується одне поле зі змінним складом, зміст якого буде залежати від того, до якого підкласу відноситься описуваний об'єкт. Реалізація принципу явного виділення підкласів в структурі БД істотно залежить від специфіки СУБД.

При проектуванні логічної структури БД основне значення має специфіка предметної області, що відображається. Однак, як зазначалося вище, і характер обробки інформації впливає на прийняте проектне рішення. Наприклад, рекомендується зберігати разом інформацію, часто оброблювану спільно, і, навпаки, розділяти по різних файлах інформацію, не що використовується одночасно. Інформацію, яку використовують часто, і інформацію, частота звернення до якої мала, також слід зберігати в різних файлах, причому останню може виявитися вигідним винести в архівні файли, а не підтримувати в складі БД.

Як зазначалося вище, описується не окремий об'єкт, а клас об'єктів. Але в окремих випадках буває, що клас включає в себе тільки один екземпляр. Наприклад, якщо предметною областю є якийсь конкретний інститут, то клас об'єктів буде містити тільки один екземпляр об'єкта.

Таким «виродженим» класами об'єктів зазвичай не ставиться у відповідність окремий файл бази даних.

У якості системи керування базою даних була обрана SQLite – компактна вбудована реляційна база даних, що поставляється з вихідними кодами. Вперше випущена в 2000 році, призначена для надання звичних можливостей реляційних баз даних без властивих їм накладних витрат. За час експлуатації

встигла заслужити репутацію як переносима, легка у використанні, продуктивна і надійна база даних.

Слово «вбудована» (embedded) означає, що база даних існує не як процес, окремий від обслуговується процесу, а є його частиною – частиною деякого прикладного застосування. SQLite не використовує парадигму клієнт-сервер, вона являє собою бібліотеку, з якої програма компонується, і SQLite стає складовою частиною програми. Таким чином, в якості протоколу обміну використовуються виклики функцій (API) бібліотеки SQLite. І клієнт, і сервер працюють в одному процесі – це позбавляє від проблем конфігурації. Все, чого потребує програміст, вже скомпільовано в його додатку. Такий підхід зменшує накладні витрати, час відгуку і спрощує програму. SQLite зберігає всю базу даних (включаючи визначення, таблиці, індекси і дані) в єдиному стандартному файлі на тому пристрої, на якому виповнюється програма. Простота реалізації досягається за рахунок того, що перед початком виконання транзакції запису весь файл, який зберігає базу даних, блокується; ACID-функції досягаються в тому числі за рахунок створення файлу журналу.

Кілька процесів або потоків можуть одночасно без будь-яких проблем читати дані з однієї бази. Запис в базу можна здійснити тільки в тому випадку, якщо ніякі інші запити в даний момент не обслуговуються; в іншому випадку спроба запису закінчується невдачею, і в програму повертається код помилки. Іншим варіантом розвитку подій є автоматичне повторення спроб запису протягом заданого інтервалу часу.

Обрана архітектура web-системи, що проектується «Safeshell» з захисту особистих даних, реалізує архітектуру «клієнт-сервер» та передбачає, що база даних (БД) розміщена на сервері локальної комп'ютерної мережі. На робочих станціях розміщують клієнтську програму, що формує і відсилає запити віддаленого серверу з БД, який забезпечує виконання запиту і видачу клієнтові результату. Вся обробка інформації відбувається на віддаленому сервері.

Основним компонентом БД є таблиця. Таблиця використовується для структуризації та зберігання інформації. Всередині однієї БД існують зв'язки між таблицями, кожна з яких задає спільне користування даними таблиці. Діаграма "Сутність – Зв'язок" графічно представляє структуру даних проектованої інформаційної web-системи.

Визначивши функціональні вимоги до web-системи «Safeshell», використовуючи моделювання SQL, спроектована база даних, що реалізує модель «Сутність – Зв'язок». При розробці системи була створена база даних, за допомогою СУБД Firebase Cloud Firestore.

Перший етап процесу проектування бази даних називається концептуальним проектуванням бази даних. Він полягає у створенні концептуальної моделі даних предметної області. Ця модель даних створюється на основі функціональних вимог користувачів. Концептуальне проектування бази даних абсолютно не залежить від таких подробиць її реалізації, як тип обраної СУБД, набір створюваних прикладних програм, використовувані мови програмування, тип обраної обчислювальної платформи, а також від будь-яких інших особливостей фізичної реалізації. Концептуальне проектування – створення концептуального уявлення бази даних, що включає визначення типів найважливіших сутностей та існуючих між ними зв'язків і атрибутів. Послідовність етапів проектування концептуальної моделі даних: визначення сутностей; визначення взаємозв'язків між сутностями; визначення атрибутів сутностей; завдання первинних і альтернативних ключів.

3.3.1 Опис таблиць бази даних і зв'язків між ними

Для інформаційної системи «Safeshell» з захисту особистих даних в базі даних були визначені наступні сутності: Користувач, Пост, Сервіс, Безпека, URL, Коментар. Кожна сутність містить первинний ключ, призначений для ідентифікації екземпляра сутності. Первинний ключ має бути вибраним таким чином, щоб за значеннями атрибутів, можна було точно

ідентифікувати сутність, крім того жоден з атрибутів первинного ключа не повинен мати нульове значення. Значення атрибутів первинного ключа не повинні змінюватися. Для визначення первинного ключа часто використовують унікальні номери, які можуть автоматично генеруватися системою при додаванні екземпляра сутності в БД. Застосування унікальних номерів полегшує процес індексації та пошуку в БД.

Далі необхідно визначити атрибути для кожної сутності, а також записати для всіх сутностей первинні ключі. Тепер розставити зв'язки між сутностями [12][1)]. Уявімо базу даних інформаційної системи «Safeshell» з безпеки особистих даних у вигляді моделі «Сутність – Зв'язок» (рис. 3.5).

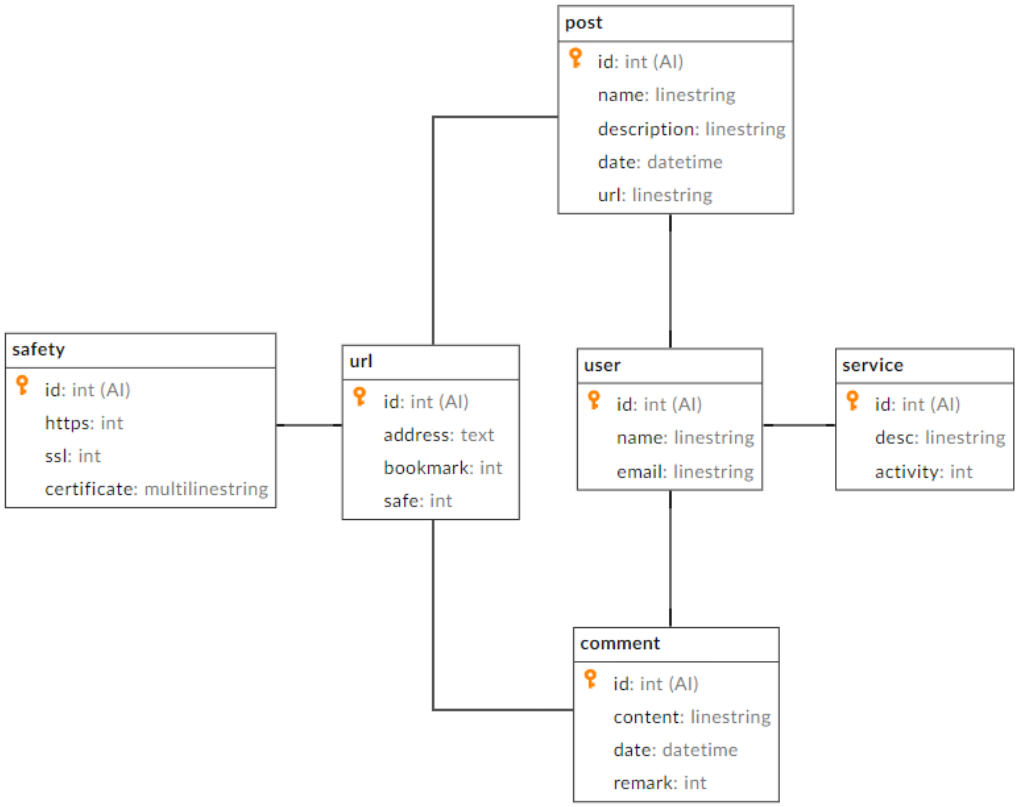


Рисунок 3.5 − Діаграма «Сутність – Зв'язок» БД ІС «Safeshell»

[1)] [12] Введение в проектирование баз данных. URL: https://iiba.ru/vvedenie-v-proektirovanie-baz-dannih (дата звернення 6.12.2019).

У комплекті поставки йде також функціональна клієнтська частина у вигляді виконуваного файлу sqlite3, за допомогою якого демонструється реалізація функцій основної бібліотеки. Клієнтська частина є кросплатформеною утилітою командного рядка.

Завдяки архітектурі SQLite її можливо використовувати як на вбудованих системах, так і на виділених машинах з великими масивами даних.

SQLite має модульну архітектуру, яка відображає унікальні підходи до управління реляційними базами даних. Вісім окремих модулів згруповані в три головних підсистеми (рис. 3.6). Вони поділяють обробку запиту на окремі завдання, які працюють подібно конвеєру. Верхні модулі компілюють запити, середні виконують їх, а нижні працюють з диском і взаємодіють з операційною системою.

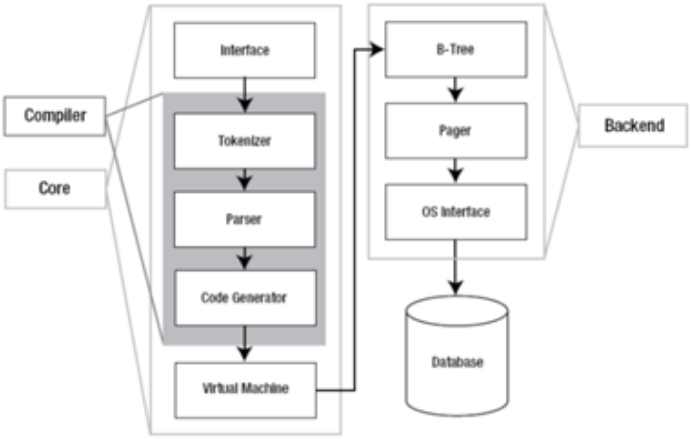


Рисунок 3.6 – Архітектура SQLite

Використовуючи цю систему керування базами даних, була створена база даних, яка має таблиці, наведені на рис. 3.7 – 3.12.

Таблиця «post» містить у собі інформацію про кожен окремий пост про адресу. Структура даної таблиці наведена на рис. 3.7.

Рисунок 3.7 – Таблиця «service»

Таблиця «service» містить у собі інформацію про активність користувача в ІС. Структура даної таблиці наведена на рис. 3.8.

Рисунок 3.8 – Таблиця «service»

Таблиця «comment» містить у собі інформацію про залишенні користувачами коментарії. Структура даної таблиці наведена на рис. 3.9.

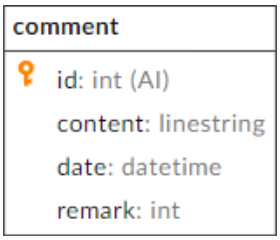


Рисунок 3.9 – Таблиця «comment»

Таблиця «user» містить у собі інформацію про користувачів цієї ІС. Структура даної таблиці наведена на рис. 3.10.

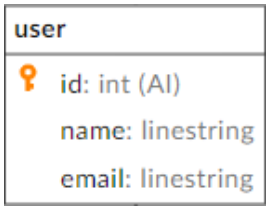


Рисунок 3.10 – Таблиця «user»

Таблиця «url» містить у собі інформацію про кожну окрему URL адресу сайту яку запам’ятала ІС. Структура даної таблиці наведена на рис. 3.11.

Рисунок 3.11 – Таблиця «url»

Таблиця «safety» містить у собі інформацію про кожну досліджену URL адресу та деякий аналіз безпеки сайту. Структура даної таблиці наведена на рис. 3.12.

Рисунок 3.12 – Таблиця «safety»

Сутність Користувач містить наступні атрибути: номер (первинний ключ), ім'я, електронна адреса.

Сутність Сервіс містить наступні атрибути: номер (первинний ключ), опис, активність.

Сутність Пост курси містить наступні атрибути: номер (первинний ключ), назву, опис, дату створення, відповідний URL.

Сутність Коментар містить наступні атрибути: номер (первинний ключ), вміст, дату створення, кількість попереджень.

Сутність URL містить наступні атрибути: номер (первинний ключ), адресу, оцінка, загальна оцінка безпеки.

Сутність Безпека містить наступні атрибути: номер (первинний ключ), наявність протоколу HTTPS, наявність сертифікату SSL та наявність назву сертифікату безпеки.

Між таблицями Пост і Користувач існує зв'язок типу «один – до – багатьох»: одному посту відповідає кілька користувачів.

Між таблицями Сервіс і Користувач існує зв'язок типу «один – до – багатьох»: одному сервісу відповідає кілька користувачів.

Між таблицями Коментар і Користувач існує зв'язок типу «один – до – багатьох»: одному коментарю відповідає кілька користувачів.

Між таблицями URL і Пост існує зв'язок типу «один – до – багатьох»: одній адресі відповідає кілька постів.

Між таблицями URL і Коментар існує зв'язок типу «один – до – багатьох»: одній адресі відповідає кілька коментарів.

Між таблицями Безпека і URL існує зв'язок типу «один – до – багатьох»: одній перевірки безпеки відповідає кілька URL адрес.

4 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

4.1 Схема функціонування web-системи

Користувачі інформаційної системи, реалізованої під час проекту «Safeshell» для захисту особистих даних мають можливість навігації по розділах системи. Одним з головних елементів будь-якої системи є інтерфейс користувачів.

Головне завдання системи – надавати зручний, що має багато функціональних можливостей, графічний інтерфейс користувачам для роботи з додатком.

Ефективність запобігання та подолання помилок користувачами тим краще, чим рідше користувачі помиляються при роботі з даним інтерфейсом і чим менше часу і зусиль потрібно для подолання наслідків вже зроблених

помилок. Велике значення має також ризик, пов'язаний з виникненням помилки.

Адекватність користувальницького інтерфейсу програми – це його відповідність тим завданням, які користувачі повинні і хотіли б вирішувати з її допомогою.

Розроблена система повинна бути настільки зрозумілою, щоб користувач, ніколи раніше не бачив її, але добре розбирається в предметній області, міг без всякого навчання почати її використовувати. Крім того система не повинна перешкоджати ефективній роботі досвідчених користувачів, що працюють з нею довгий час. Одною з основних деталей, що впливають на враження користувача, є меню і можливості навігації, наявність зображень і організація елементів на сторінці. Меню повинні бути інтуїтивно зрозумілими і підкріплюватися навігаційними підказками.

Одним з основних вимог до веб-інтерфейсів є їхній однаковий зовнішній вигляд і однакова функціональність при роботі в різних браузерах.

Титульна сторінка будь-якого сайту повинна максимально інформативне й у стиснутому обсязі відображати необхідну користувачеві інформацію про сайт. На головній сторінці необхідно помістити логотип, основне меню, форму пошуку, стрічку новин, посилання, аутентифікації й реєстраційну форму. Відповідно до розробленої структурою було спроектовано головна сторінка сайту.

Розроблена система за своїм стилем відноситься до класу веб-застосувань покоління Web 5.0. Для розробки дизайну використані наступні принципи: проста навігація, поєднання кольорів, центральне вирівнювання, яскраві кольори.

Схема надання інтерфейсу системи залежно від категорії користувача представлена на рис. 4.1.

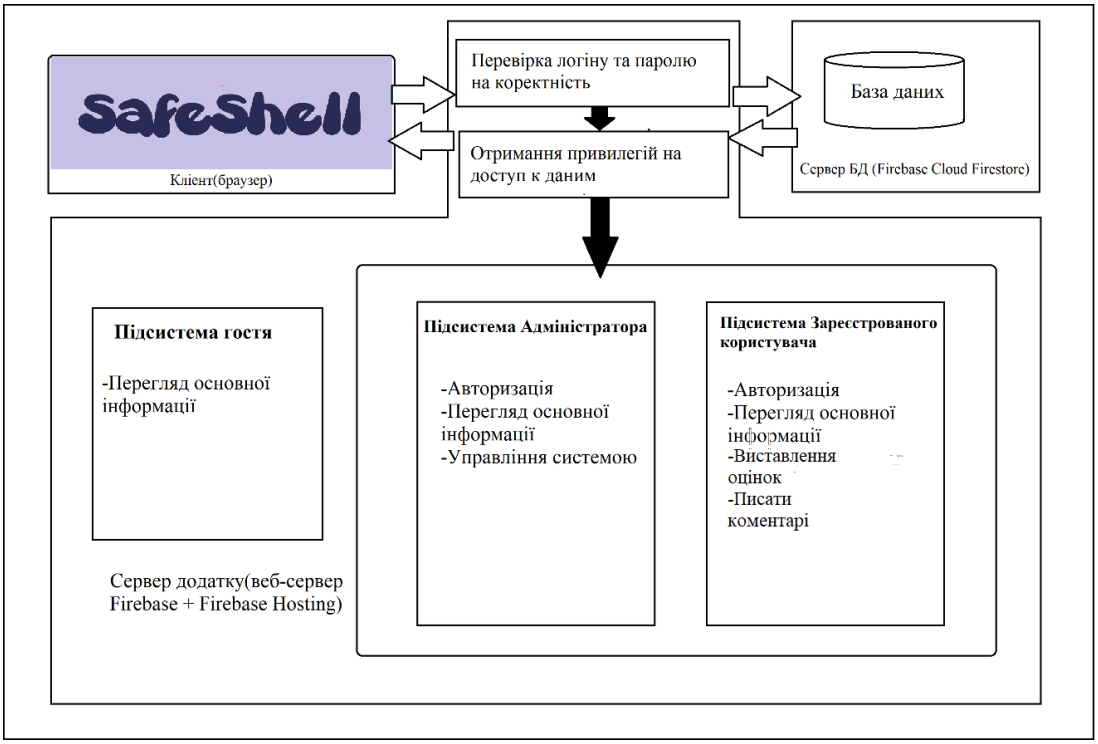


Рисунок 4.1 – Схема роботи web-системи «Safeshell»

Система надає наступні можливості:

– перегляд основної інформації системи;
– аналіз URL адрес на предмет безпеки;
– тестування;
– доступ до оцінок користувачів;
– особистий профіль;
– можливість реєстрації;
– можливість авторизації;
– можливість звернутися в підтримку;
– можливість редагування особистої інформації;
– можливість переглядати інформацію про безпеку.

Взаємодія web-сторінок інформаційної системи «Safeshell» представлена на рис. 4.2.

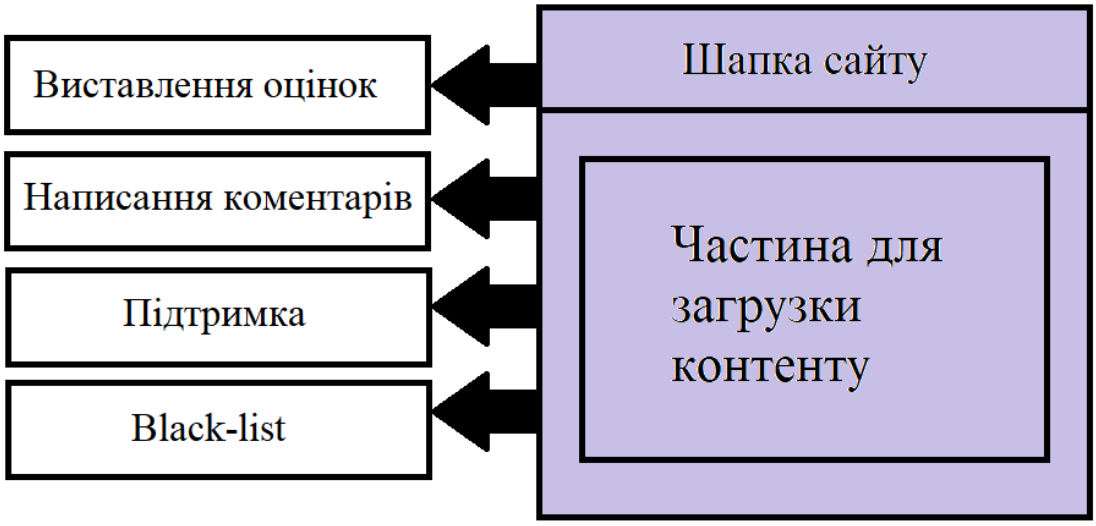


Рисунок 4.2 –Взаємодія web-сторінок системи «Safeshell»

4.2 Реалізація структури Web-системи

Планування структури Web-системи це важливий організаційний процес, від нього залежить якість, швидкість розробки, зручність читання і вартість. Добра структурованість Web-системи забезпечує половину успіху у її створенні. Логічна структура Web-системи повина показувати, яким чином інформація розподіляється по сторінках системи і як вона може бути отримана користувачем.

Навігація по інформаційній системі – це те, що дозволяє відвідувачеві знайти потрібну йому інформацію. Він спирається на логічну структуру системи і навігація допомагає користувачеві швидко по ній переміщатися. Система навігації не повинна обтяжувати сторінку і відволікати від її вмісту.

Елементи локальної навігації візуально відокремлюються від елементів глобальної, але так, щоб вони, в теж час, виглядали як єдине ціле. Розроблено шаблон сторінок майбутньої web-системи «Safeshell». Всі сторінки проектованої системи виконані в єдиному стилі. Щоб витримати стиль, спочатку був розроблений шаблон Головної сторінки (рис. 4.3).

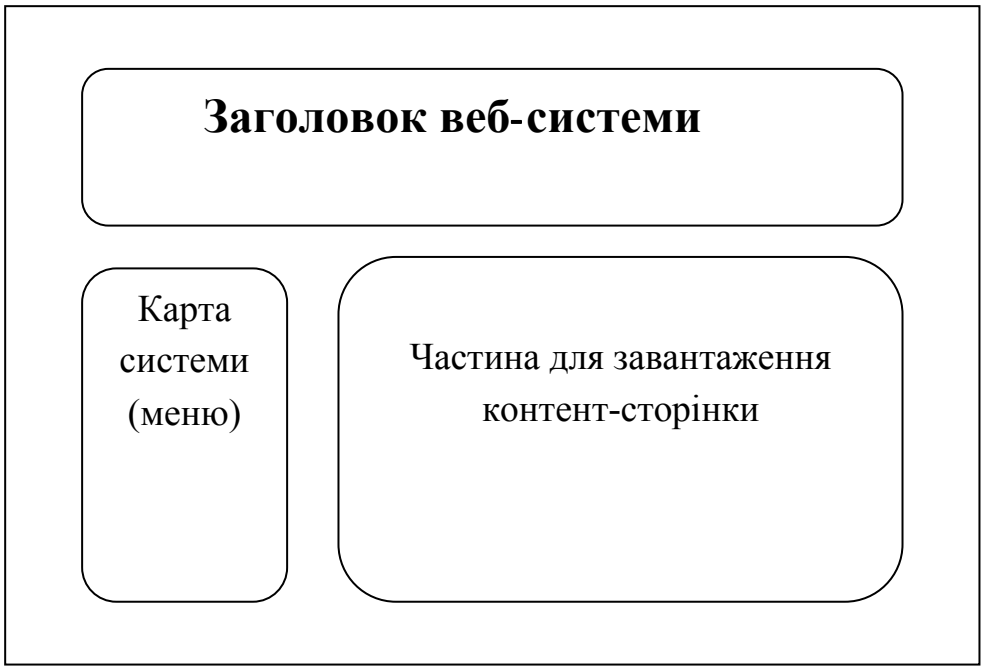


Рисунок 4.3 – Шаблон сторінки web-системи

Шаблони зручні тим, що більшість сторінок верстають за подобою однієї сторінки майже автоматично. Структура шаблону складається з елементів, які повинні бути присутніми на всіх сторінках системи. Меню навігації розташовується в лівій частині сторінки. Всі сторінки системи являють собою область подільну на кілька частин. Вгорі сторінки знаходиться логотип і назва системи. Під заголовком, розміщуються дві основні частини сторінки. Зліва розташована карта сайту – меню посилань на контент-сторінку Інтернет-системи.

У правій частині сторінки область для завантаження контент-сторінок. Структура усіх контент-сторінок однотипна. Тут розташована таблиця з відповідною вибіркою і елементи управління даною таблицею. Він опирається на логічну структуру інформаціоної системи і навігація допомагає користувачеві швидко переходити між розділами.

4.3 Інтерфейс користувача адміністратора

Все управління системи здійснюється адміністратором через адміністративну панель в Firebase (рис. 4.4).

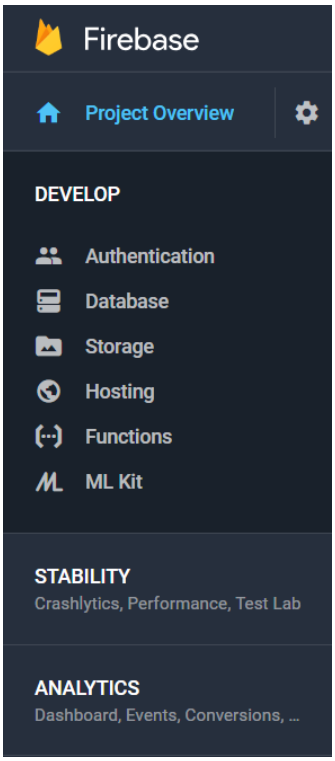


Рисунок 4.4 – Панель управління ІС адміністратора

Вхід в адміністративну панель здійснюється через звичайну форму входу, після введення логіна і пароля адміністратору надаються повністю всі права керівництва системою за допомогою відкритої панелі керування. У панелі управління є можливість налаштувати проект, додати чи змінити учасника з аналогічними правами (рис. 4.5).

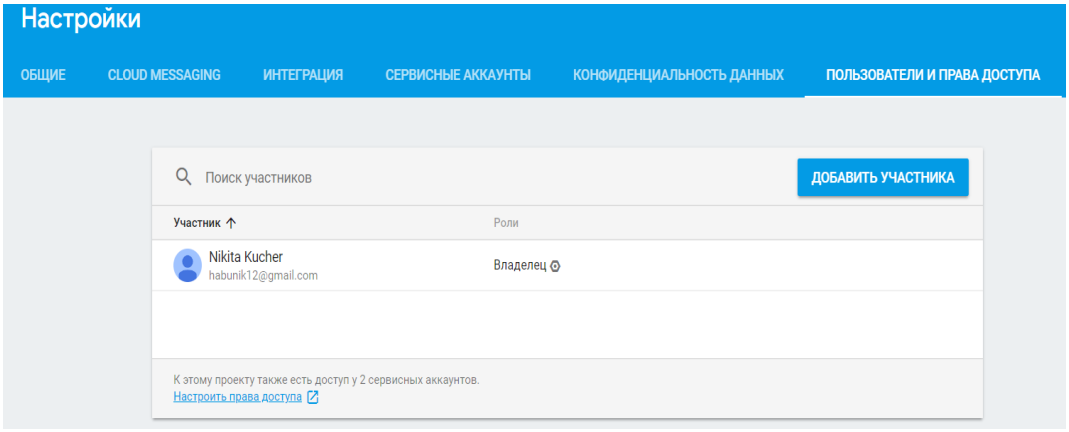


Рисунок 4.5 – Профіль адміністратора в панелі керування

Після зміни всіх налаштувань адміністратор має можливість приступати до роботи над створенням системи. Панель управління надає всі можливості для зміни вмісту бази даних, налаштування для зареєстрованих користувачів, керування доменами та хостингом. Передбачена можливість додавання фотографій, аудіо та відео файлів, документів.

Адміністратор може створити групу користувачів, або зареєструвати окремо і надає їм необхідні можливості (права доступу до керування ІС) роботи в веб-системі.

Адміністратор може створити групу користувачів, або зареєструвати окремо і надає їм необхідні можливості (права доступу до керування ІС) роботи в веб-системі. Тільки адміністратор має можливість активувати, додати або сбросити пароль користувача. Загальний вигляд панелі представлений на рис. 4.6.

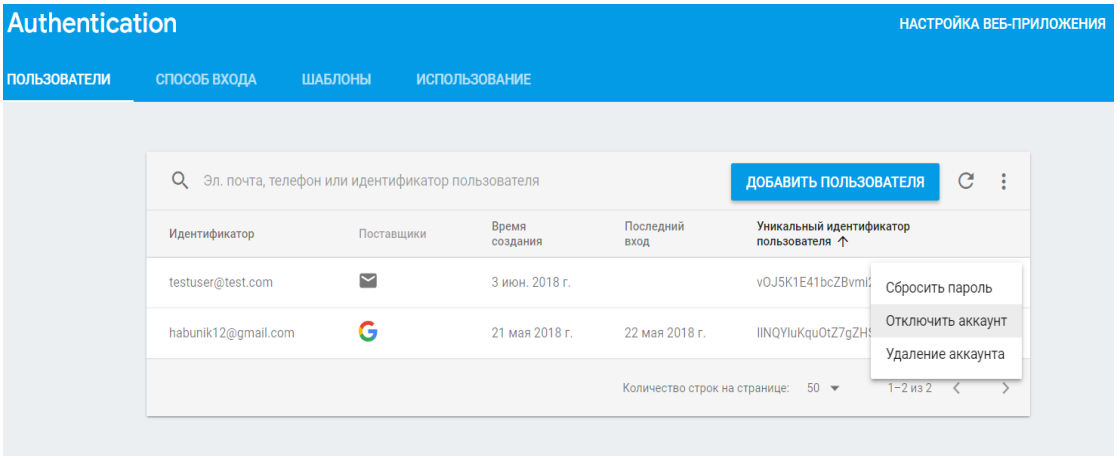


Рисунок 4.6 – Загальний вигляд панелі управління Адміністратора

У системі управління ІС доступні такі засоби входу: електрона адреса та пароль, телефон, Google, Play Ігри, Facebook, Twitter, Github та через анонімний вхід.

Для того, щоб мати можливість використовувати такі розділи ІС як: тест, база посилань і т.д. необхідно увійти в систему за допомогою сервісу Google. Так, постачальник даних Google надає web-системі основну інформацію про зареєстрованого користувача, а саме: фото профілю, електрону адресу та ім’я.

Користувачеві Гостю, який не зареєстрований в системі і не внесено до бази даних – доступна лише загальна інформація в ІС. Він має можливість тільки дивитися базові сторінки системи.

Йому забороняється змінювати, додавати або видаляти будь-яку інформацію в системі, дозволяється лише перегляд деяких пунктів меню в яких зберігається загальна інформація. Він може переглянути загальні функції та скористатись головним функціоналом. Доступні засоби входу до ІС «Safe-shell» вказані на рис. 4.7.

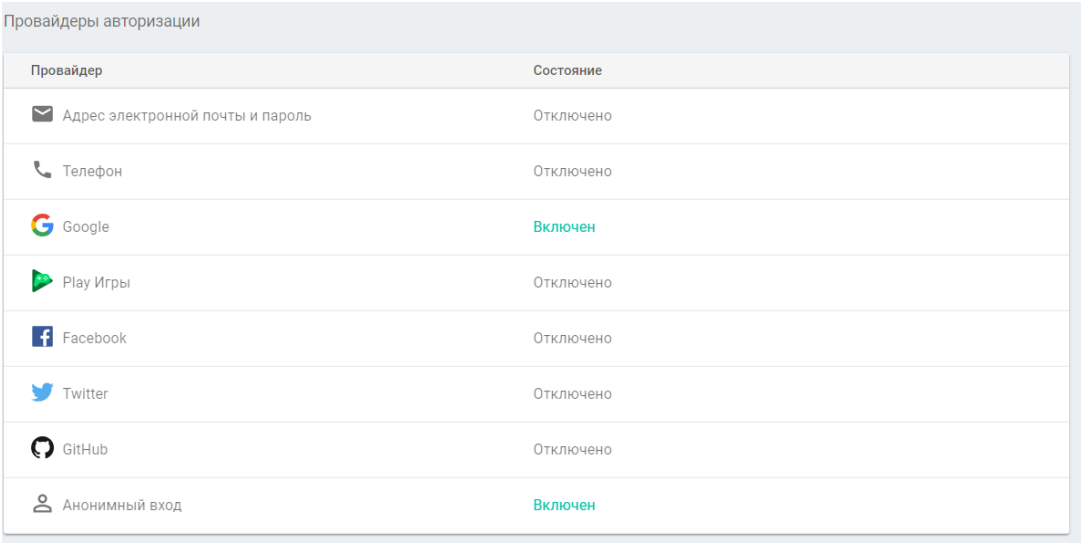


Рисунок 4.7 – Доступні провайдери авторизації

Налаштування прав користувачів, які зареєструвалися, здійснюється також через адміністративну панель. Багато додатків для спільної роботи дозволяють користувачам читати і записувати різні фрагменти даних на основі набору дозволів. Наприклад, в програмі сканування документів користувачі можуть дозволити кільком користувачам читати і писати свої документи, блокуючи небажаний доступ. Так як права або ролі користувачів впливають на можливість роботи з системою, права на зміну, видалення, редагування, незареєстрованим користувачам не давалися. Приклад налаштування відображений на рис. 4.8.

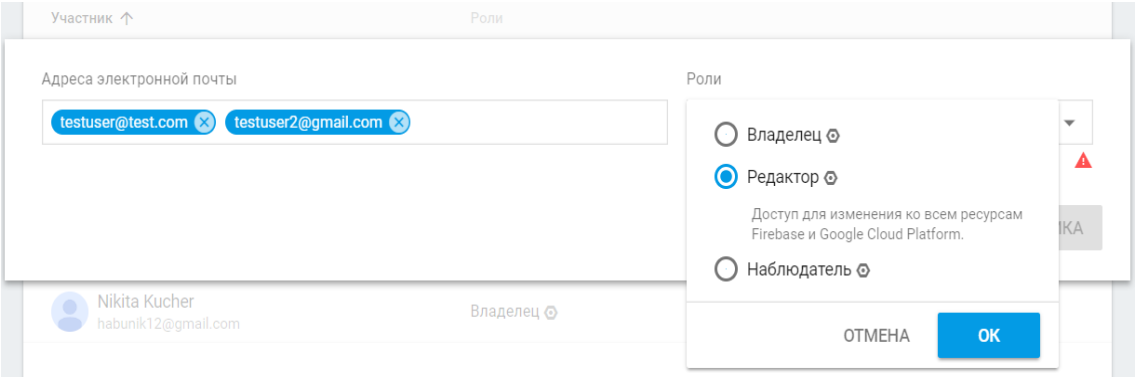


Рисунок 4.8 – Налаштування прав категорій Користувачів системи

Групи користувачів створювалася лише для того, щоб дати можливість перегляду деяких пунктів меню, які є конфіденційною інформацією і не повинні бути доступні Користувачам – Гостям системи, які не мають спеціальних логинів і паролів занесених системним адміністратором в базу даних.

Таким же чином налаштовувалися всі права для всіх користувачів системи. Для адміністраторів ІС присвоювалась роль Власник, і були визначений повний доступ до всіх ресурсів Firebase і Google Cloud Platform, в тому числі для управління користувачами, правами доступу і платіжними функціями.

4.4 Управління інтерфейсами користувачів системи

Розробка та управління для інтерфейсом користувачів системи дуже важка праця. Адже від подачі системи користувачеві буде залежати її популярність, а відповідно і відвідуваність. Тому інтерфейс користувача повинен бути простий в навігації і легкодоступний. React адаптується під браузери, тому всі користувачі зможуть без проблем працювати в системі і просто переглядати її. Для системи був встановлений єдиний шаблон з шапкою і пунктами меню, які доступні всім користувачам однаково (рис. 4.9).

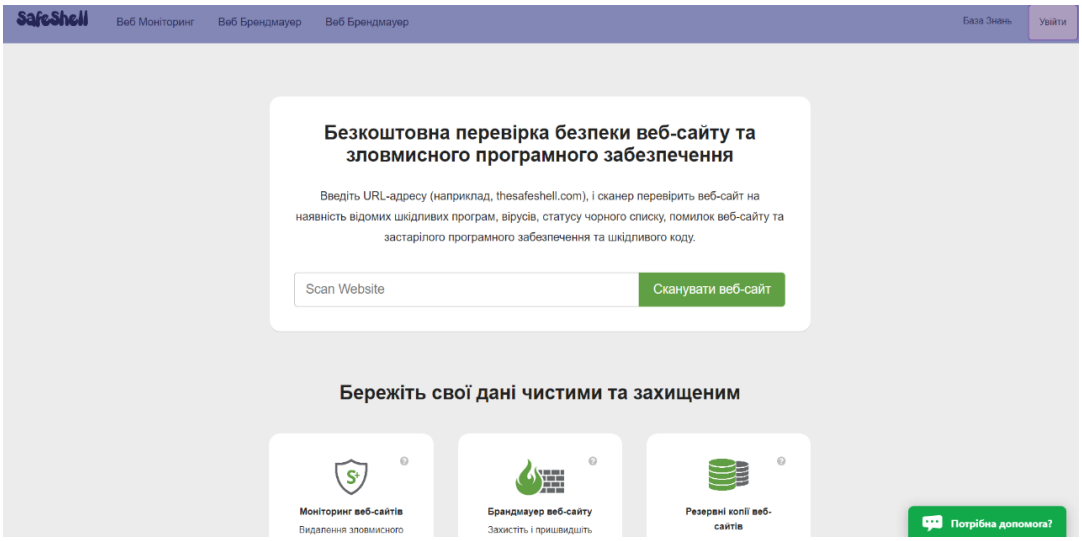


Рисунок 4.9 – Головна сторінка web-системи «Safeshell»

Інтерфейс форми входу для користувачів відрізняється від форми входу в адміністративну панель адміністратора, вона є простою і зрозумілою, а її стиль відповідає стилю всієї системи. Адмістратору надана можливість блокувати акаунти користувачів при необхідності.

Незареєстрований користувач (Користувач – Гість) має доступ не до всіх переваг ІС, йому доступна лише інформаційна частина. При авторизації Користувача – Зареєстрований користувач доступна деяка інформація в верхній частині сайту, наприклад ім'я та прізвище та аватар якщо він завантажений в його Google акаунт (рис. 4.10).

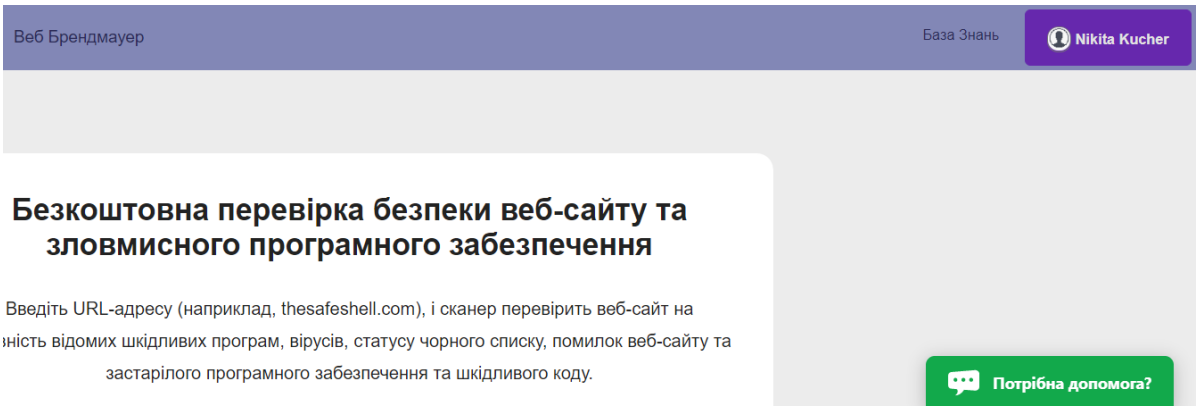


Рисунок 4.10 – Вигляд ІС після авторизації користувача

У шапці ІС є кнопка після натискання якої з’являється панель з пунктами навігації по системі, там відображаються основні розділи і в будь-який момент користувач має можливість повернутися на головну сторінку натиснувши по кнопці з логотипом.

Усі користувачі бачать всі пункти меню і мають можливість пошуку необхідної інформації та перегляді результатів аналізу систем після пошуку. Зареєстрованому користувачу надається можливість зайти в особистий профіль. Також має можливість додавання в закладки URL адрес коментарів з системи (рис. 4.11).

Рисунок 4.11 – Профіль авторизованого користувача

В системі було створено розділ Залишити відгук для залишення відгуку чи пропозиції, відгуки після перевірки адміністрацією з'являється в інформаційній системі, простий користувач – Гість в системі не має можливість залишити відгук.

Користувач – Адміністратор має можливість редагувати повідомлення.

У реалізованому у інформаційній системі «Safeshell» тесті після аналізу URL адреси користувач має можливість засвідчитися у важливих для сайту системах захисту та прочитати детальну інформацію про перевірену інформаційну систему.

Після того як користувач знайде бажане посилання для перевірки і натискне на кнопку для її аналізу на перевірку безпеки, йому буде наданий повний результат про протоколи захисту. У результаті з'являються вкладки з інформацією після внутрішнього тестуваня на сервері, результат заноситься до профілю (рис. 4.12).

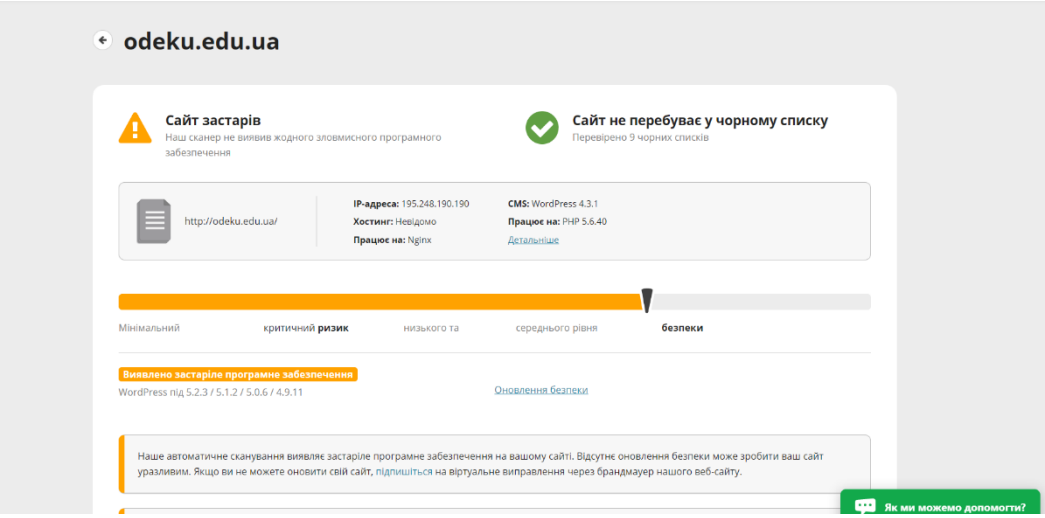


Рисунок 4.12 – Вигляд результатів тестування

У розділі Безпека для зручності усі статті було розбито на декілька розділів: – це програмне забезпечення, стан чорного списку, моніторинг та брендмауер (рис. 4.13). Зміст кожного розділу відповідає рівню безпеки підзаголовку. Для кожного розділу був підібраний відповідний вміст, щоб сервер міг прочитати усю важливу інформацію про веб сторінку та повідомити користувача.

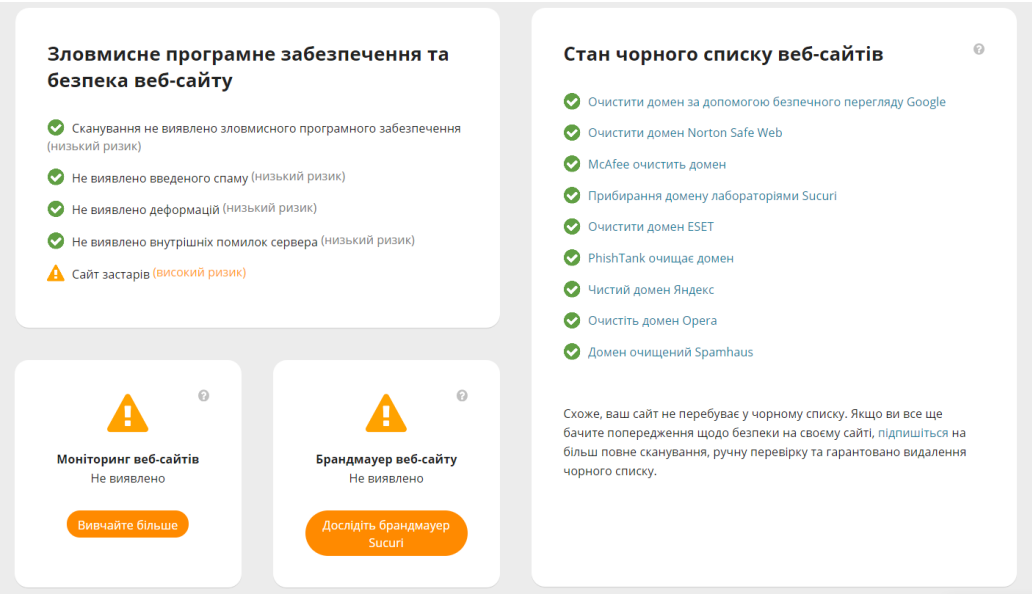
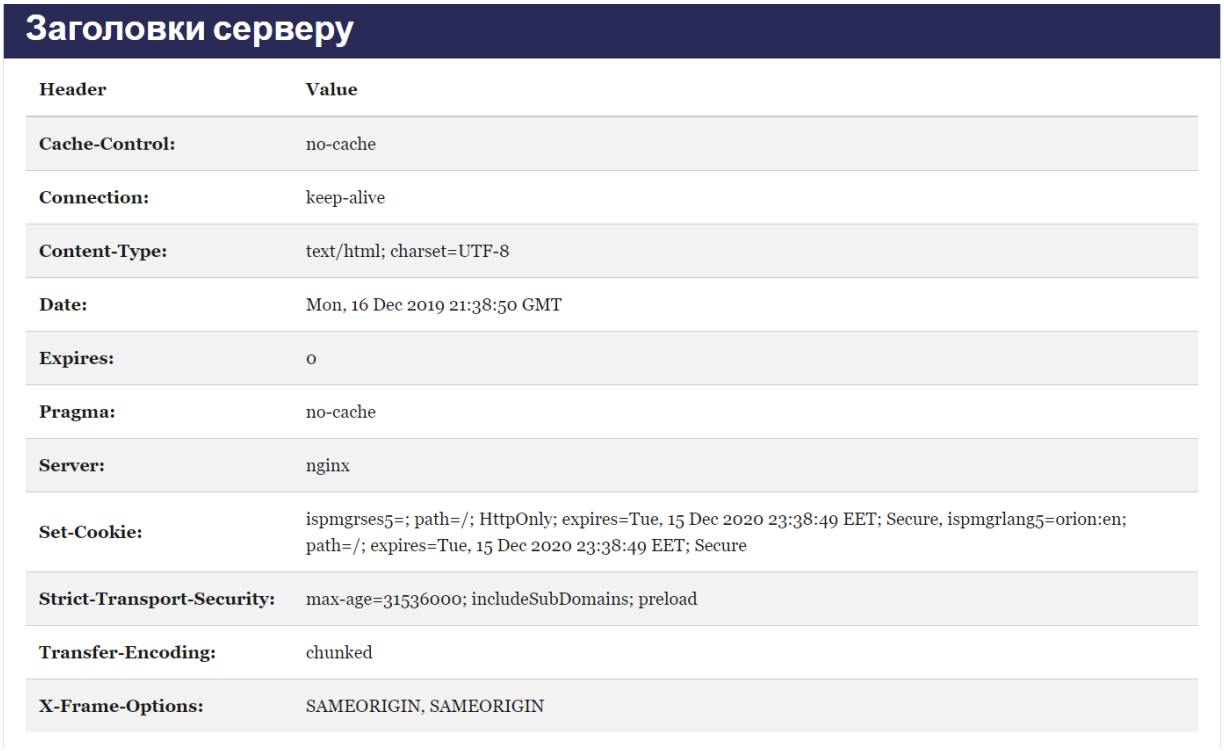


Рисунок 4.13 – Вигляд розділів у розділі Безпека

Кожний розділ має зручний вигляд та все необхідне для перевірки серверної частини, а саме кешування, тип підключення до іншого ресурсу, тип даних які передаються, дата останього оновлення серверу, час життя куків сайту, кодування (рис. 4.14).



Заголовки серверу

Header	Value
Cache-Control:	no-cache
Connection:	keep-alive
Content-Type:	text/html; charset=UTF-8
Date:	Mon, 16 Dec 2019 21:38:50 GMT
Expires:	0
Pragma:	no-cache
Server:	nginx
Set-Cookie:	ispmgrses5=; path=/; HttpOnly; expires=Tue, 15 Dec 2020 23:38:49 EET; Secure, ispmgrlang5=orion:en; path=/; expires=Tue, 15 Dec 2020 23:38:49 EET; Secure
Strict-Transport-Security:	max-age=31536000; includeSubDomains; preload
Transfer-Encoding:	chunked
X-Frame-Options:	SAMEORIGIN, SAMEORIGIN

Рисунок 4.14 – Сторінка з заголовками серверу

Розділ Тестові бали містить наступні підпункти:

– політика безпеки вмісту;
– cookies;
– поширений розподіл ресурсів
– закріплення відкритого ключа http;
– сувора безпека транспорту http;
– перенаправлення;
– політика довідника;
– цілісність субресурсів;

– параметри типу x-content-type;

– параметри x-frame;

– x-xss-захист.

Додатково користувач може навести на спеціальний значок у правому краю таблиці та отримати інформацію для виправлення. Кожен пункт меню проходить перевірку на відаленному сервері для того щоб захистити користувача від відкриття потенційно небезпечного URL. У розділі Тестові бали користувач має можливість подивитися вид зробленого тесту, чи відбувався перехід на іншу сторінку під час аналізу, загальна оцінка підрозділу та причина у якій йде опис який необхідно виправити (рис. 4.15).

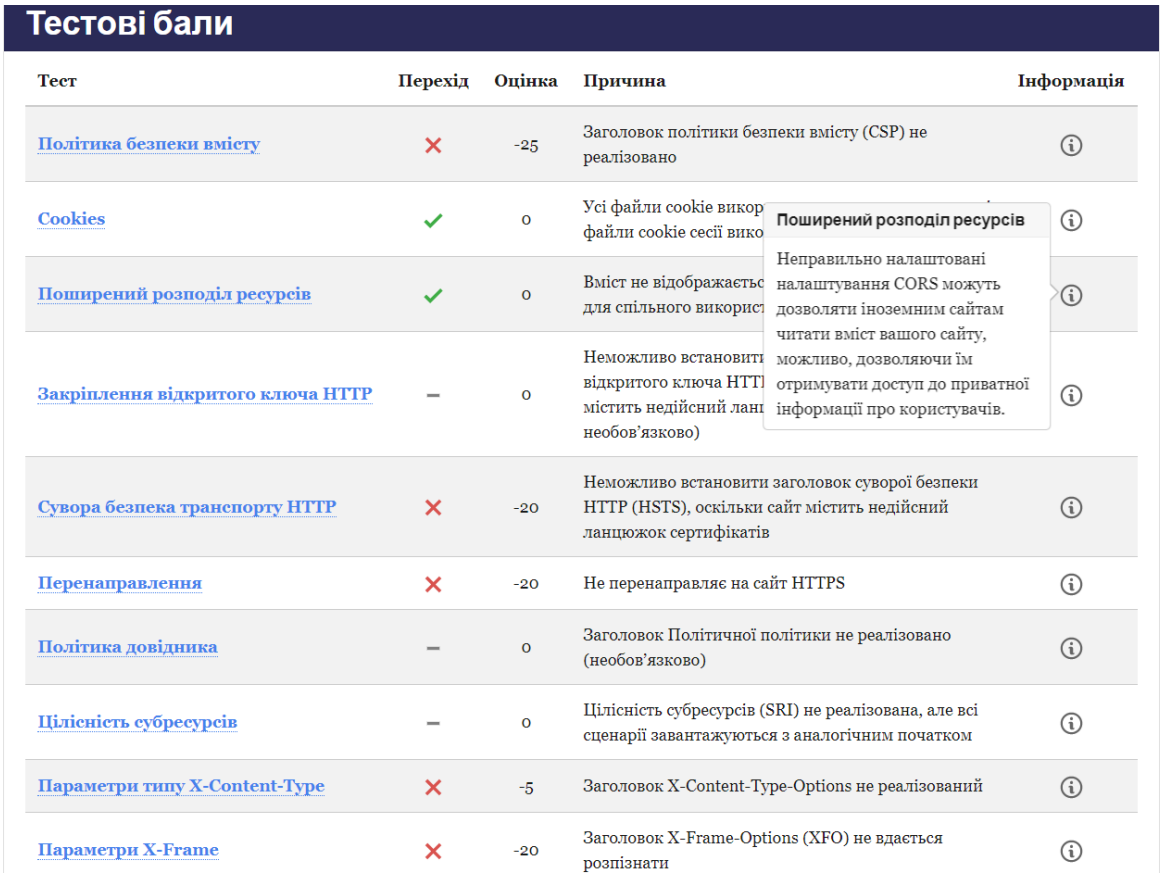


Рисунок 4.15 – Вигляд розділу Тестові бали

У розділі TLS сертифікат розповідається про захист на транспортному рівні можливості безпечної передачі даних в Інтернет для навігації. Зокрема

про індифікатор сканування, рівні сумісності. Також про шифрування доменого ім’я (рис. 4.16).

На додаток до наведених властивостей, ретельна конфігурація TLS може надавати додаткові властивості, пов’язані з конфіденційністю, такі як передача секретності , гарантуючи, що будь-яке подальше розкриття ключів шифрування не може використовуватися для розшифрування будь-яких TLS-повідомлень, записаних у минулому.

Даний опис протоколу TLS містить два шари: запис TLS та протоколи рукостискання TLS.

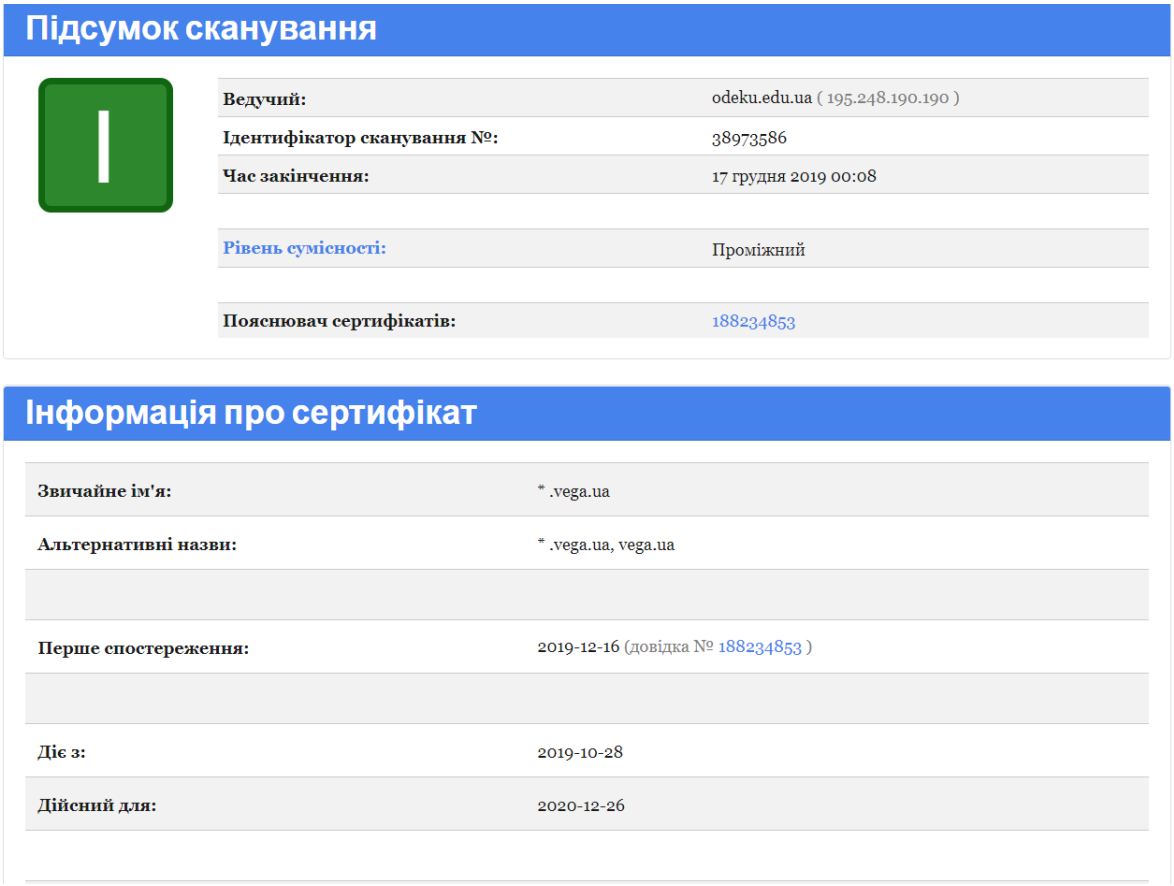
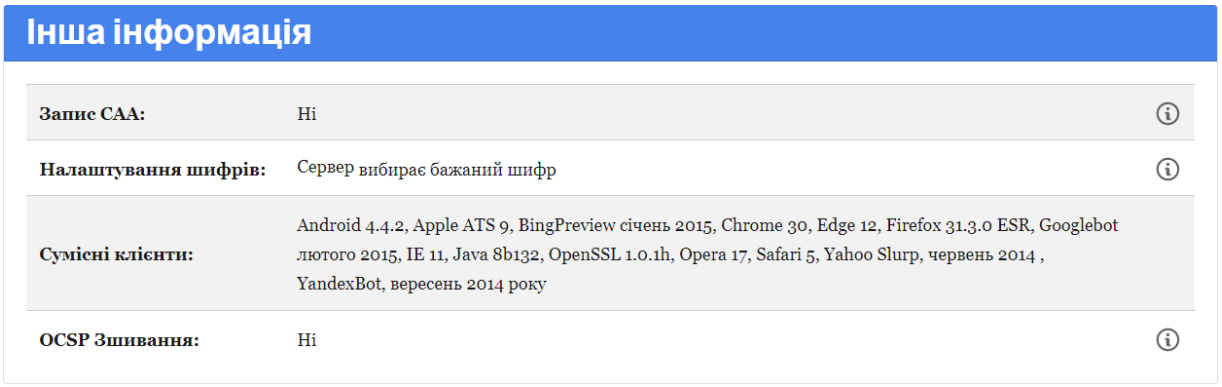


Рисунок 4.16– Розділ TLS сертифікат

У розділі Інша інформація містится дані про Запис CAA та Сумісних клієнтів (рис. 4.17).

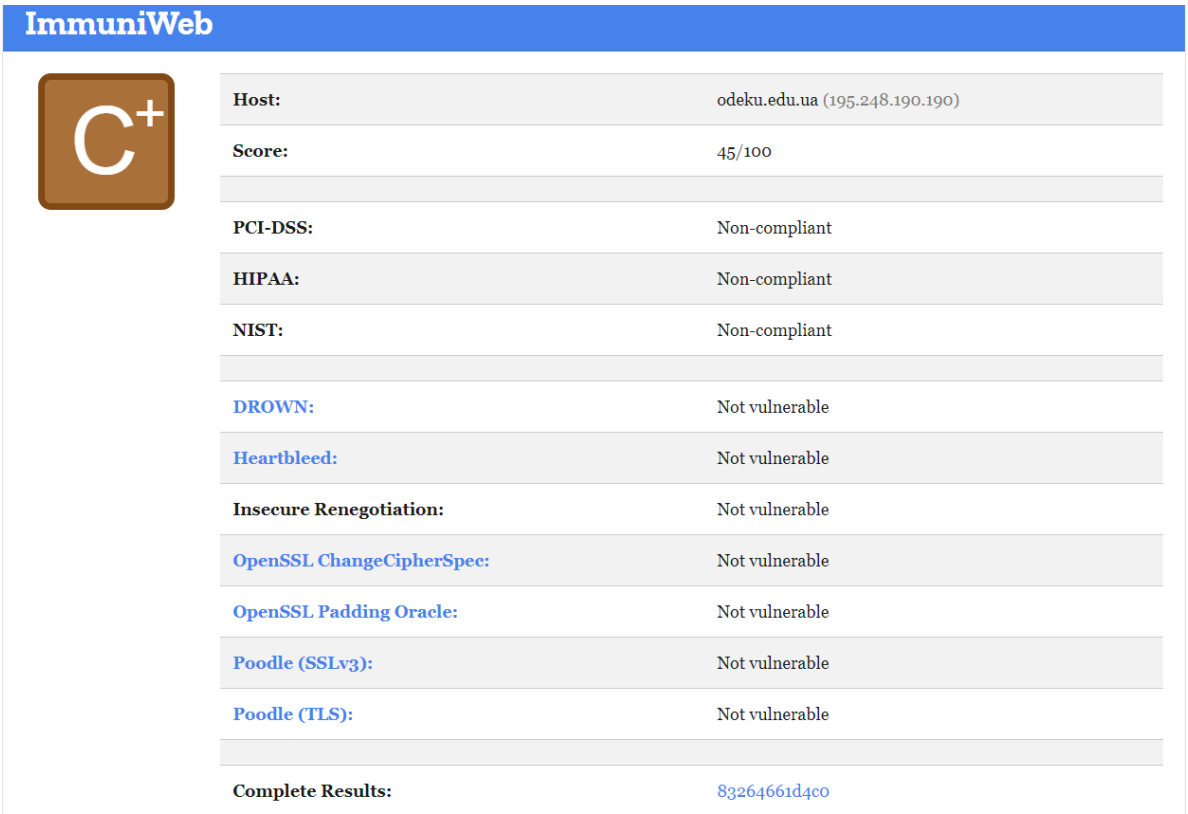


Інша інформація

Запис CAA:	Ні	ⓘ
Налаштування шифрів:	Сервер вибирає бажаний шифр	ⓘ
Сумісні клієнти:	Android 4.4.2, Apple ATS 9, BingPreview січень 2015, Chrome 30, Edge 12, Firefox 31.3.0 ESR, Googlebot лютого 2015, IE 11, Java 8b132, OpenSSL 1.0.1h, Opera 17, Safari 5, Yahoo Slurp, червень 2014 , YandexBot, вересень 2014 року	
OCSP Зшивання:	Ні	ⓘ

Рисунок 4.17– Розділ Інша інформація TLS сертифікату

У розділі Інші тести є користувач може знайти перевірку та оцінку сайту за ІС ImmuniWeb. Він працює за допомогою штучного інтелекту (рис. 4.18).



ImmuniWeb

C+

Host:	odeku.edu.ua (195.248.190.190)
Score:	45/100
PCI-DSS:	Non-compliant
HIPAA:	Non-compliant
NIST:	Non-compliant
DROWN:	Not vulnerable
Heartbleed:	Not vulnerable
Insecure Renegotiation:	Not vulnerable
OpenSSL ChangeCipherSpec:	Not vulnerable
OpenSSL Padding Oracle:	Not vulnerable
Poodle (SSLv3):	Not vulnerable
Poodle (TLS):	Not vulnerable
Complete Results:	83264661d4c0

Рисунок 4.18 – Розділ Інші тести

4.5 Управління інтерфейсами веб-додатку

Для реалізації інформаційного веб-додатку «Safeshell» під різні браузери знадобилось створити один головний HTML файл який містить верстку. Розширення виготовляються з різних, але згуртованих компонентів.

Компоненти можуть включати фонові сценарії, скрипти вмісту, сторінку параметрів, елементи інтерфейсу та різні логічні файли. Компоненти розширень створюються за допомогою технологій веб-розробки: HTML, CSS та JavaScript. Компоненти розширення залежатимуть від його функціональності та можуть не вимагати кожної опції. Фонові сценарії та багато інших важливих компонентів повинні бути зареєстровані в маніфесті.

Було створено додаток який можна викликати на верхній панелі браузера на будь-якому сайті (рис. 4.19). Він містить посилання до повного звіту, назву URL, рівень безпеки, репутацію на основі оцінок інших користувачів. Також користувач має можливість заповнити форму після деякого часу проведеного на сайті та розповісти про його зміст.

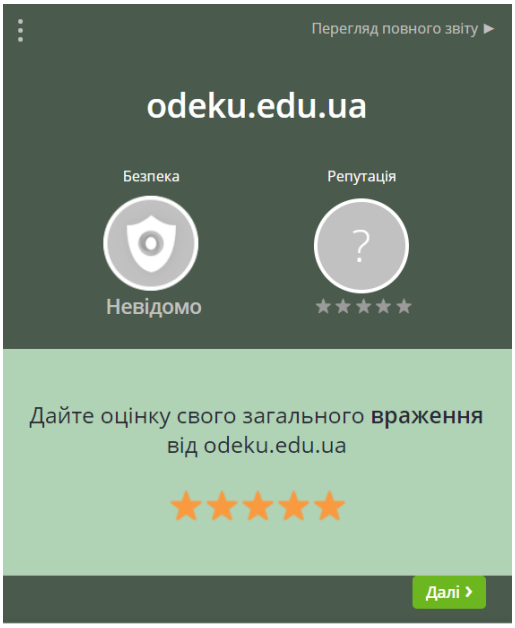


Рисунок 4.19 – Вигляд веб-додатку

4.6 Структура програмного забезпечення web-системи

Для реалізації інформаційної web-системи «Safeshell» на фреймворку React знадобилось створити велику кількість компонентів. Сім компонентів є більш важливими ніж інші, в них закладена маршутизація, вся функціональність створеної Internet-системи для захисту особистих даних. На рис. 4.20 представлена схема взаємодії основних компонентів і HTML-сценарію для реалізації Internet-системи безпеки даних та перевірки безпеки сайту.

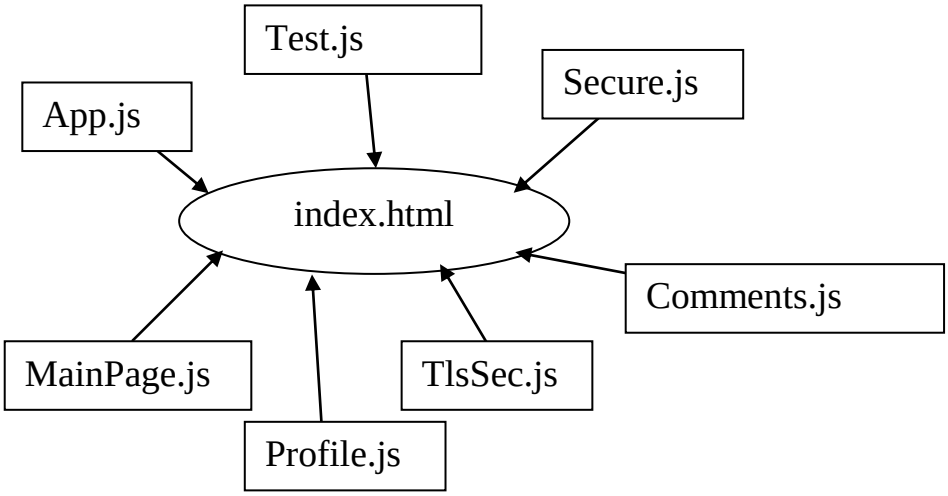


Рисунок 4.20 – Схема взаємодії компонентів в системі

Опишемо коротко призначення React-компонентів, а саме які функції вони виконують:

– App.js – цей компонент містить шапку та підвал сайту, він керує станом головним меню сайт. Цей компонент містить у кожній сторінці web-системи «Safeshell»;
– Secure.js – цей компонент містить інформацію про перевірку програмного забезпечення, стан чорного списку, моніторинг та брендмауер;

– Comments.js – цей компонент містить загальний вміст сторінки з коментарями та надає можливість виставляти оцінки інформаційним системам;
– TlsSec.js – цей компонент містить інформацію про захист на транспортному рівні можливості безпечної передачі даних в Інтернет для навігації;
– Test.js – цей компонент інформує користувача про вид зробленого тесту, чи відбувався перехід на іншу сторінку під час аналізу, загальна оцінка підрозділу та причина у якій йде опис який необхідно виправити;
– Profile.js – цей компонент виводить профіль, основну інформацію про зареєстрованого користувача, закладки які він додав з інший сторінок web-системи «Safeshell»;
– Main.js – цей файл містить головну сторінку сайту, він виводить логотип системи, переваги використання та загальну інформацію.

ВИСНОВКИ

В даній дипломній роботі була створена ІС захисту особистих даних на основі протоколу HTTPS за допомогою WEB-технологій яка була створена на фреймворку React. Ця система буде розрахована на перевірку посилань на безпеку, інформування користувача про проблеми сайту та їх рішення, виставлення оцінок та написання коментарів. Система має ергономічний дизайн та структуру.

Користувачів веб-додатоку було розділенно на ролі. Кожному з них надані певні права. Адміністратор керує сервною частиною сайту, базою даних та хостингом.

Описана в даній роботі інформаційна система включає наступні можливості:

– надання доступу до пропонованих послуг;
– можливість реєстрації користувачів;
– можливість перевірити сайт на безпечність;
– можливість побачити інформацію по даний сайт;
– зворотній зв'язок.
– Розроблюючи ІС було вирішено такі задачі:
– досліджено та проаналізовано існуючі аналоги;
– розроблено орігінальний дизайн ;
– вивчено механізм роботи на фреймворку React;
– змодельовані структура бази даних;
– зроблено аналіз та пошук цікавої інформації;
– розроблено власний демонстраційний продукт;

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. HTTPS – Википедия. URL: https://uk.wikipedia.org/wiki/HTTPS (дата звернення 3.12.2019).
2. Створення та розміщення власних веб-сайтів на безкоштовних хостінгах. URL: https://studfiles.net/preview/5535698/page:16 (дата звернення 3.12.2019).
3. What are extensions? – Google Chrome. URL: https://developer.chrome.com/extensions (дата звернення 3.12.2019).
4. AngularJS – Википедия. URL: https://uk.wikipedia.org/wiki/AngularJS (дата звернення 3.12.2019).
5. Vue.js – Википедия. URL: ru.wikipedia.org/wiki/Vue.js (дата звернення 5.12.2019).
6. React – Википедия. URL: http://www.wikiwand.com/uk/React (дата звернення 5.12.2019).
7. Преимущества react. Блог Новикова Богдана. URL: https://hcbogdan.com/frontend/2016/02/14/react-benefits (дата звернення 6.12.2019).
8. Firebase – Википедия. URL: https://uk.wikipedia.org/wiki/Firebase (дата звернення 6.12.2019).
9. Разработка схемы классов. URL: https://www.intuit.ru/studies/courses/4806/1054/lecture/16134?page=2 (дата звернення 6.12.2019).
10. Знайомство з PHP (призначення, включення в документ, обробка, базові налаштування) – Вікі ЦДПУ. URL: https://wiki.cuspu.edu.ua/index.php/ (дата звернення 6.12.2019).
11. Проектирование баз данных – Википедия. URL: https://ru.wikipedia.org/wiki/Проектирование_баз_данных (дата звернення 6.12.2019).
12. Введение в проектирование баз данных. URL: https://iiba.ru/vvedenie-v-proektirovanie-baz-dannih (дата звернення 6.12.2019).

ДОДАТОК А

Життєвий цикл react.js

А.1 Схема циклу react.js

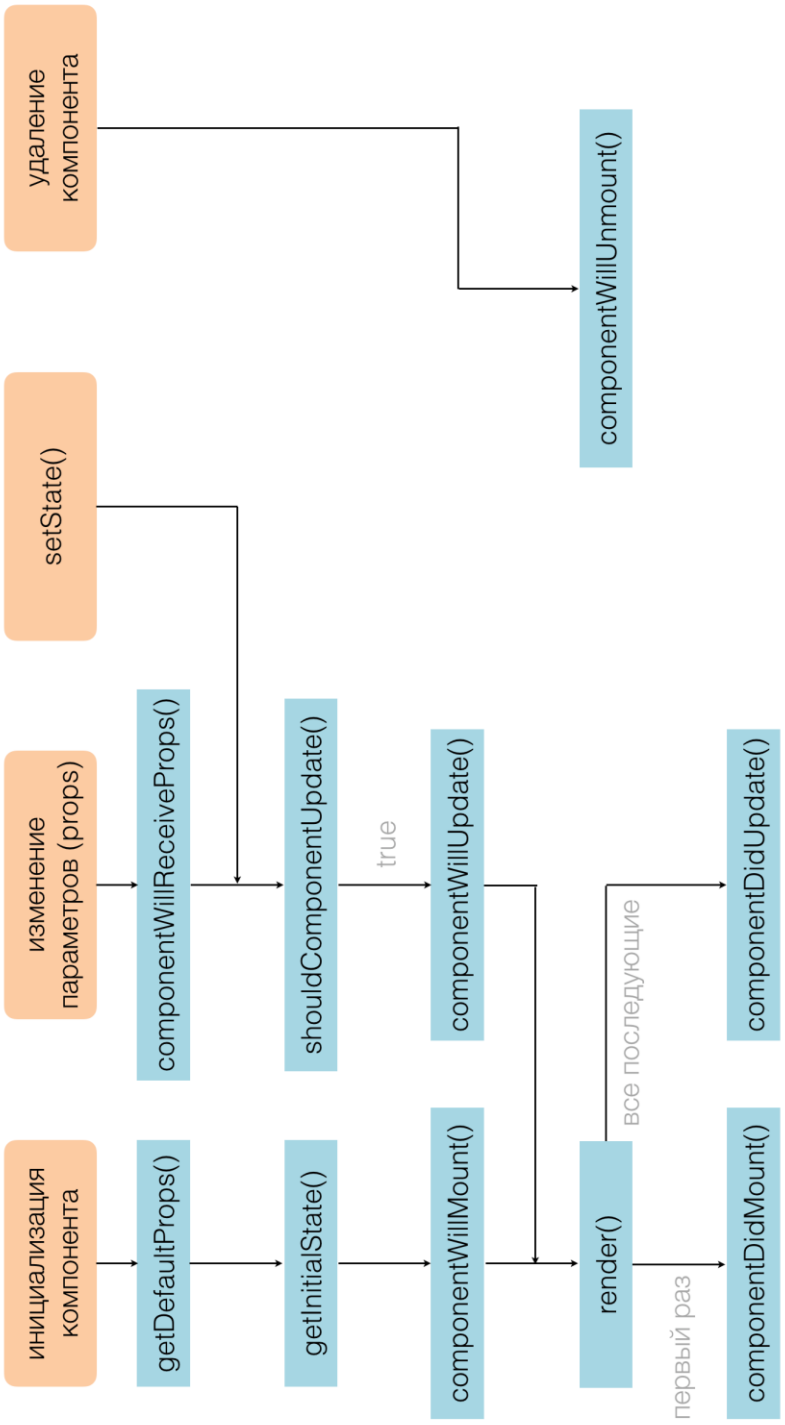


Рисунок А.1 – Життєвий цикл роботи фреймворку React.js

ДОДАТОК Б

Обмін повідомлень Firebase

Б.1 Схема отримання повідомлень Firebase

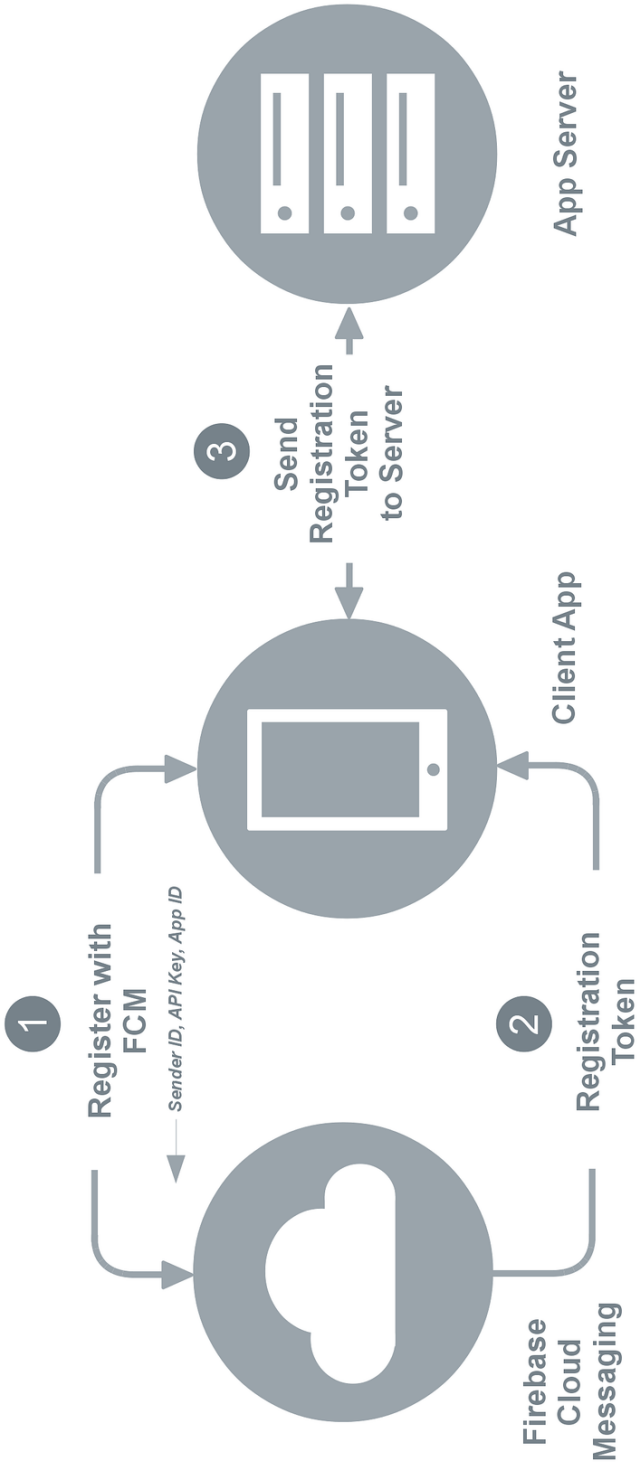


Рисунок Б.1 – Схема Firebase Cloud Messaging