МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ТЕЛЕФОННОГО ДОВІДНИКА З ВИКОРИСТАННЯМ ФРЕЙМВОРКУ NUXT.JS 2»**

Виконав: студент 4 курсу, групи 6.1210-2пі
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
В.Т. Медиждов
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, PhD Чопорова О.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної та прикладної математики, професор, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Меджидову Вагіфу Тофіговичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка телефонного довідника з використанням фреймворку Nuxt.js 2

керівник роботи Чопорова Оксана Володимирівна, PhD
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 21 » грудня 2023 року № 2180-с

2. Строк подання студентом роботи 03.06.2024 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Телефонний довідник.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація за темою доповіді

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 25.12.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	09.01.2024	
2.	Збір вихідних даних.	30.01.2024	
3.	Обробка методичних та теоретичних	19.02.2024	
	джерел.		
4.	Розробка першого та другого розділу.	12.04.2024	
5.	Розробка третього розділу.	17.05.2024	
6.	Оформлення та нормоконтроль	27.05.2024	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	20.06.2024	

Студент _____ В.Т. Медиждов
(підпис) (ініціали та прізвище)

Керівник роботи _____ О.В. Чопорова
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка телефонного довідника з використанням фреймворку Nuxt.js 2» : 39 с., 22 рис., 6 джерел.

ДОВІДНИК, КОНТАКТИ, НОМЕРА, ТЕЛЕФОН, ФРЕЙМВОРК.

Об’єкт дослідження – телефонний довідник.

Мета роботи: розробити телефонний довідник для зручного збереження контактних номерів та їх пошуку з усього списку.

Метод дослідження – у дипломній роботі проаналізовано аналоги телефонних довідників та викладено теоретичні відомості про засоби розробки.

На основі даних теоретичних відомостей розроблено проєкт та реалізовано телефонний довідник і його користувацький інтерфейс за допомогою фреймворку Nuxt.js 2.

SUMMARY

Bachelor’s qualifying paper «Development of a Telephone Directory Using the Nuxt 2.js Framework»: 39 pages, 22 figures, 6 references.

DIRECTORY, CONTACTS, NUMBERS, PHONE, FRAMEWORK.

The object of research is a telephone directory.

Purpose: to develop a telephone directory for convenient storage of contact numbers and their search from the entire list.

Research method – the thesis analyzes analogs of telephone directories and presents theoretical information about development tools.

Based on this theoretical information, a project was developed and implemented to create a telephone directory and its user interface using the Nuxt.js 2 framework.

ЗМІСТ

ВСТУП

У сучасному суспільстві стрімко розвиваються інформаційні технології, програмування користувацьких додатків у різних сферах діяльності. Останнім часом на кожному підприємстві та кожної людини впроваджуються електронні ресурси для спрощення роботи: швидкого інформування, зберігання інформації, контролювання звітів тощо.

Однією з актуальних проблем є зберігання усіх необхідних контактних номерів та їх швидкий пошук. За допомогою таких систем відбувається запис та зберігання контактних номерів, скорочується час на пошук необхідного контакту. Зокрема, така можливість є важливою для людей, у яких дуже багато контактних номерів інших людей та мають необхідність у швидкому їх пошуку.

Мета дослідження – розробити телефонний довідник для зручного збереження контактних номерів та їх пошуку з усього списку.

Під час роботи було виконано наступні завдання:

– проаналізовано існуючі телефонні довідники;
– досліджено та обрано засоби розробки сайтів;
– розроблено телефонний довідник;
– розроблено телефонний довідник із можливістю пошуку та сортування відповідно до проєкту.

Об’єкт дослідження – телефонний довідник із можливістю пошуку та сортування контактів.

Розробку проєкту було здійснено за допомогою фреймворку Nuxt.js 2.

У ході вирішення поставлених завдань було отримано такі результати: створено телефонний довідник з можливісттю сортвання та швидкого пошуку. Він має такі переваги як: логічно зрозуміла структура, простий інтерфейс, можливість зберігання, сортування та пошуку контактних номерів.

Дипломна робота складається зі вступу, трьох розділів, висновків.

Перший розділ містить основні відомості про довідники аналогічної спрямованості та використовувані засоби розробки.

Другий розділ містить технічне завдання, опис структури довідника, діаграму варіантів використання, ER-діаграми.

Третій розділ містить опис розробленого довідника, особливості реалізації довідника, опис форми створення нового контакту та його пошуку.

1 ОГЛЯД СУЧАСНИХ ТЕЛЕФОННИХ ДОВІДНИКІВ

Є багато онлайн-сервісів, які дозволяють зберігати та керувати контактами. Було розглянуто такі сервіси як:

– Google Контакти;
– Microsoft Outlook;
– Contacts+.

1.1 Google Контакти

Google Контакти [1] – це зручний та потужний сервіс для зберігання та управління контактами, який пропонує Google. Він інтегрується з іншими сервісами Google, такими як Gmail, Google Drive, Google Calendar тощо, і доступний на різних пристроях (див. рис. 1.1).

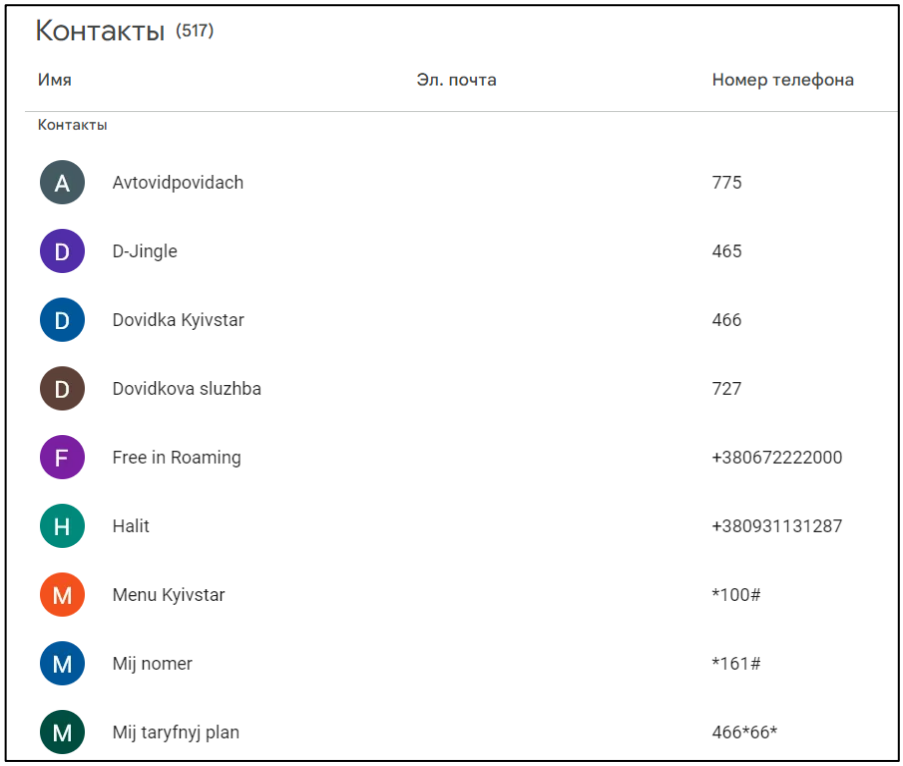

Рисунок 1.1 – Google Контакти

Основні функції і можливості Google Контактів.

Збереження та управління контактами:

– додавання нових контактів з різною інформацією (номер телефону, електронна пошта, адреса, дні народження, примітки тощо);
– можливість додавати фотографії до контактів;
– групування контактів для зручної організації.

Синхронізація:

– автоматична синхронізація контактів між усіма пристроями, що використовують Google-акаунт (Android-смартфони, планшети, комп’ютери тощо).

Імпорт та експорт контактів:

– імпорт контактів з інших сервісів або файлів (CSV, vCard);
– експорт контактів для резервного копіювання або перенесення на інші платформи.

Злиття дублікатів:

– автоматичне виявлення та злиття дублікатів контактів для підтримки чистоти списку контактів.

Інтеграція з іншими сервісами Google:

– прямий доступ до контактів з Gmail (наприклад, при написанні нових електронних листів);
– інтеграція з Google Calendar для нагадування про дні народження та інші події.

Безпека та конфіденційність:

– високий рівень безпеки даних завдяки використанню облікового запису Google;
– можливість налаштування прав доступу до контактів.

Як користуватися Google Контактами.

Вхід в Google Контакти:

– відкрити веббраузер і перейти на сайт Google Контакти;
– увійти у свій Google-акаунт.

Додавання нового контакту:

– натиснути на кнопку «Створити контакт» у верхньому лівому куті;
– ввести необхідну інформацію (ім’я, телефон, електронна пошта тощо) і натиснути «Зберегти».

Редагування та видалення контактів:

– натиснути на контакт, який потрібно відредагувати або видалити;
– вибрати «Редагувати» для внесення змін або «Видалити» для видалення контакту.

Імпорт та експорт контактів:

– натиснути на кнопку «Більше дій» (три вертикальні точки) і вибрати «Імпорт» або «Експорт»;
– дотримуватись інструкції для імпорту або експорту контактів.

Злиття дублікатів:

– натиснути на кнопку «Знайти дублікатів» у лівому меню;
– google Контакти автоматично знайдуть дублікати і запропонують їх злиття.

Переваги Google Контактів:

– зручність: легкий доступ з будь-якого пристрою з Інтернетом;
– інтеграція: глибока інтеграція з іншими сервісами Google;
– синхронізація: автоматичне оновлення контактів на всіх пристроях;
– безпека: високий рівень безпеки завдяки використанню облікового запису Google.

1.2 Microsoft Outlook Контакти

Microsoft Outlook Контакти [2] – це частина платформи Outlook, яка забезпечує зручне управління контактами для користувачів. Ця система інтегрується з електронною поштою, календарем та іншими сервісами Microsoft, що робить її ефективним інструментом для персонального та професійного використання (див. рис. 1.2).

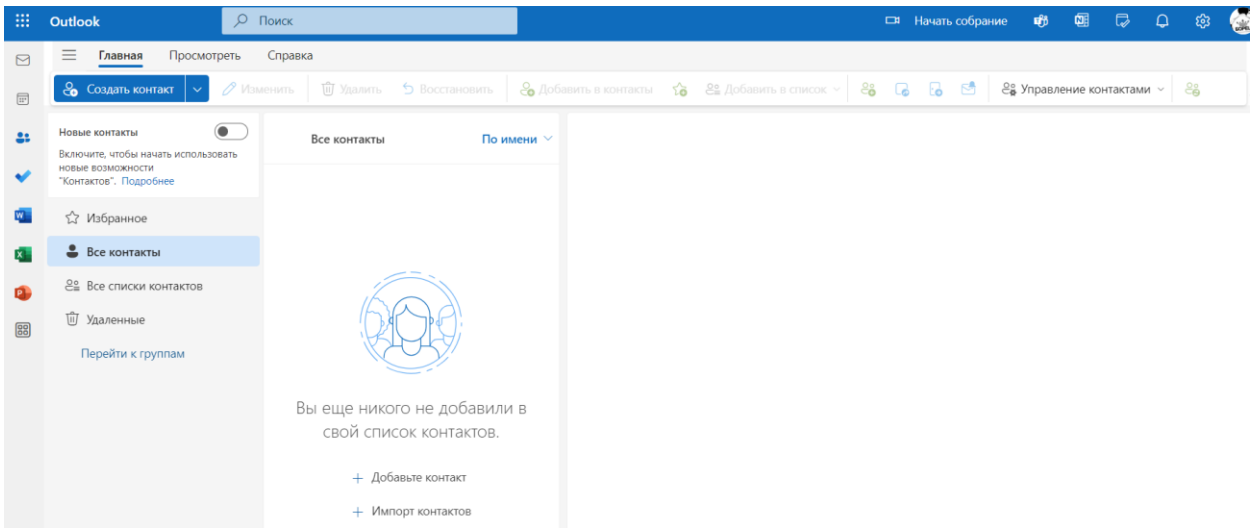

Рисунок 1.2 – Microsoft Outlook Контакти

Основні функції Microsoft Outlook Контактів.

Збереження та управління контактами:

– додавання контактів з різною інформацією, включаючи імена, номери телефонів, електронні адреси, фізичні адреси, дні народження та примітки;
– додавання фотографій до контактів для легшого розпізнавання.

Синхронізація:

– автоматична синхронізація контактів між пристроями, що використовують обліковий запис Microsoft (комп’ютери, планшети, смартфони тощо);
– інтеграція з іншими сервісами Microsoft, такими як Teams, OneDrive та Office 365.

Імпорт та експорт контактів:

– можливість імпорту контактів з інших сервісів або файлів (CSV, vCard);
– експорт контактів для резервного копіювання або перенесення на інші платформи.

Злиття дублікатів:

– автоматичне виявлення та злиття дублікатів контактів для підтримки чистоти списку контактів.

Інтеграція з електронною поштою та календарем:

– прямий доступ до контактів з електронної пошти Outlook для легкого створення нових повідомлень або запрошень на зустрічі;
– можливість створювати події та нагадування на основі контактної інформації.

Групи контактів:

– створення груп контактів для зручного надсилання повідомлень або запрошень кільком людям одночасно.

Пошук та фільтрація:

– потужний інструмент пошуку та фільтрації контактів для швидкого доступу до потрібної інформації.

Як користуватися Microsoft Outlook Контактами.

Вхід в Outlook Контакти:

– відкрити веббраузер і перейти на Outlook.com;
– увійти у свій обліковий запис Microsoft.

Додавання нового контакту:

– натиснути на кнопку «Створити контакт» у верхньому лівому куті;
– ввести необхідну інформацію (ім’я, телефон, електронна пошта тощо) і натиснути «Зберегти».

Редагування та видалення контактів:

– натиснути на контакт, який потрібно відредагувати або видалити;
– вибрати «Редагувати» для внесення змін або «Видалити» для видалення контакту.

Імпорт та експорт контактів:

– натисніти на кнопку «Керування» (три крапки) і вибрати «Імпорт контактів» або «Експорт контактів»;
– дотриматись інструкцій для імпорту або експорту контактів.

Злиття дублікатів:

– Outlook автоматично виявляє дублікати контактів і пропонує злиття.

Переваги Microsoft Outlook Контактів:

– інтеграція: глибока інтеграція з іншими сервісами Microsoft, такими як Office 365, OneDrive, Teams;
– синхронізація: автоматичне оновлення контактів на всіх пристроях;
– управління контактами: розширені можливості для організації та управління контактами;
– безпека: високий рівень безпеки даних завдяки використанню облікового запису Microsoft.

1.3 Contacts+

Contacts+ (раніше відомий як FullContact) [3] – це потужний інструмент для управління контактами, який підтримує синхронізацію з різними акаунтами і платформами. Цей сервіс надає розширені можливості для об’єднання, злиття та організації контактів, що робить його корисним як для особистого використання, так і для професійної діяльності (див. рис. 1.3).

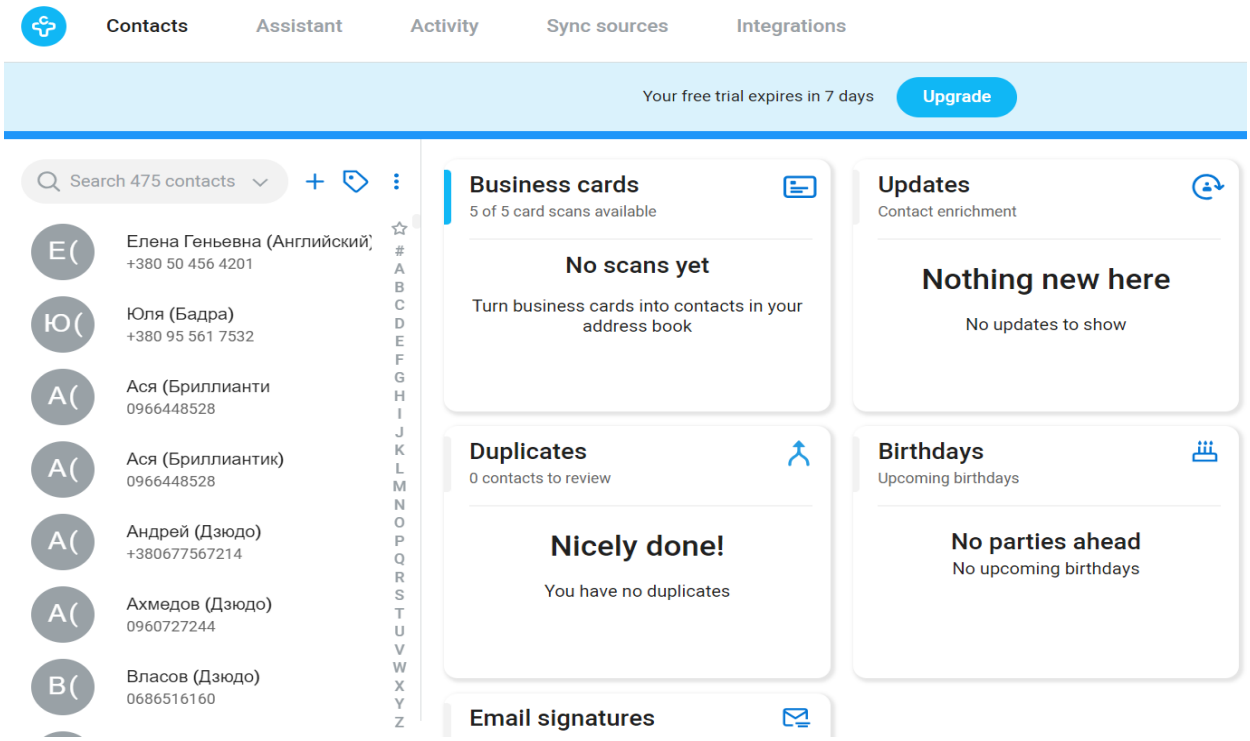

Рисунок 1.3 – Contacts+

Основні функції Contacts+.

Збереження та управління контактами:

– додавання нових контактів з різною інформацією (номер телефону, електронна пошта, адреса, дні народження, примітки тощо);
– можливість додавати фотографії до контактів.

Синхронізація:

– підтримка синхронізації з різними платформами та сервісами, включаючи Google, Microsoft, iCloud, та інші;
– автоматичне оновлення контактів на всіх пристроях.

Імпорт та експорт контактів:

– імпорт контактів з інших сервісів або файлів (CSV, vCard);
– експорт контактів для резервного копіювання або перенесення на інші платформи.

Злиття дублікатів:

– автоматичне виявлення та злиття дублікатів контактів для підтримки чистоти вашого списку контактів.

Розширене об’єднання інформації:

– об’єднання інформації з різних джерел (електронні листи, соціальні мережі) для створення повного профілю контакту.

Інтеграція з іншими сервісами:

– підтримка інтеграції з CRM-системами, такими як Salesforce та HubSpot, для професійного управління контактами.

Групи контактів:

– створення груп контактів для зручного надсилання повідомлень або запрошень кільком людям одночасно.

Пошук та фільтрація:

– потужний інструмент пошуку та фільтрації контактів для швидкого доступу до потрібної інформації.

Захист даних та конфіденційність:

– високий рівень безпеки даних з можливістю резервного копіювання та відновлення контактів.

Як користуватися Contacts+.

Реєстрація та вхід:

– відкрити веббраузер і перейдіть на Contacts+;
– зареєструватися або увійти у обліковий запис.

Додавання нового контакту:

– натиснути на кнопку «Додати контакт» у верхньому лівому куті;
– ввести необхідну інформацію (ім’я, телефон, електронна пошта тощо) і натиснути «Зберегти».

Редагування та видалення контактів:

– натиснути на контакт, який потрібно відредагувати або видалити;
– вибрати «Редагувати» для внесення змін або «Видалити» для видалення контакту.

Імпорт та експорт контактів:

– перейти до налаштувань облікового запису і вибрати «Імпорт» або «Експорт»;
– дотримуватись інструкції для імпорту або експорту контактів.

Злиття дублікатів:

– натиснути на кнопку «Знайти дублікатів» і дотримуватись інструкції для злиття контактів.

Синхронізація з іншими сервісами:

– перейти до налаштувань облікового запису і вибрати сервіси для синхронізації (Google, Microsoft, iCloud тощо);
– дотримуватись інструкції для налаштування синхронізації.

Переваги Contacts+:

– універсальність: підтримка багатьох платформ та сервісів для зручної синхронізації;
– розширені можливості: автоматичне злиття дублікатів, об’єднання інформації з різних джерел;
– інтеграція: підтримка інтеграції з CRM-системами та іншими професійними інструментами;

– безпека: високий рівень захисту даних та можливість резервного копіювання.

1.4 Висновки до розділу 1

Google Контакти є зручним інструментом для управління контактами. Вони забезпечують легкий доступ до контактної інформації, синхронізацію між пристроями та інтеграцію з іншими сервісами Google, що робить їх незамінними для користувачів екосистеми Google.

Microsoft Outlook Контакти є потужним інструментом для управління контактами як для особистого, так і для професійного використання. Вони забезпечують легкий доступ до контактної інформації, синхронізацію між пристроями та інтеграцію з іншими сервісами Microsoft, що робить їх незамінними для користувачів екосистеми Microsoft.

Contacts+ підходить для управління контактами, особливо для тих, хто використовує різні платформи та сервіси. Розширені можливості злиття дублікатів та об’єднання інформації роблять цей сервіс корисним як для особистого, так і для професійного використання.

2 ПРОЄКТУВАННЯ САЙТУ

2.1 Огляд сучасних фреймворків

Сучасні фреймворки є ключовим інструментом у розробці програмного забезпечення, допомагаючи розробникам створювати масштабовані, швидкі та гнучкі програми. Вони дозволяють зменшити час розробки, пропонують готові компоненти, а також часто мають спільноти, які підтримують їхнє розвиток.

2.1.1 Frontend-фреймворки

Frontend-фреймворки використовуються для створення інтерфейсів користувача на вебсайтах та вебдодатках.

React: бібліотека JavaScript від Facebook, яка дозволяє створювати компонентні інтерфейси. Використовується для побудови SPA (Single Page Applications) і підтримує серверний рендеринг.

Angular: повноцінний фреймворк від Google, який забезпечує комплексний підхід до створення вебдодатків. Має вбудовані інструменти для тестування та побудови великих додатків.

Vue.js: легкий фреймворк, який нагадує комбінацію React і Angular, дозволяючи швидко створювати інтерфейси. Відзначається його гнучкість і легкість у навчанні.

2.1.2 Backend-фреймворки

Backend-фреймворки використовуються для створення серверної частини вебдодатків та програм.

Django: фреймворк на базі Python, який пропонує безліч вбудованих можливостей для швидкої розробки вебдодатків. Відзначається високою продуктивністю та безпекою.

Express.js: легкий фреймворк для Node.js, який дозволяє швидко створювати серверні додатки та API. Широко використовується для створення RESTful API.

Ruby on Rails: фреймворк на базі Ruby, який дозволяє швидко створювати вебдодатки з використанням принципів конвенцій та конфігурацій.

2.1.3 Mobile-фреймворки

Mobile-фреймворки використовуються для створення мобільних додатків для платформ iOS та Android.

React Native: бібліотека, яка дозволяє створювати мобільні додатки, використовуючи JavaScript і React. Підтримує крос-платформену розробку.

Flutter: фреймворк від Google, який використовує мову Dart. Він дозволяє створювати крос-платформені мобільні додатки з високим рівнем продуктивності.

Xamarin: фреймворк, який використовує мову C# і дозволяє створювати крос-платформені мобільні додатки для iOS та Android.

2.2 Формування вимог

Для того, щоб розроблена система надавала доступний спектр послуг і відповідала потребам та запитам користувачів, до функціональної частини було висунуто такі вимоги:

– створення контактів;

– сортування за різними типами;
– видалення контактів;
– пошук контактів за номером або ім’ям;
– утворення сторінок із контактами;
– швидка робота;
– зручний і зрозумілий інтерфейс.

2.3 Вибір засобів розробки

Для розробки проєкту було використано наступні засоби: Nuxt.js 2, JavaSript, CSS/SCSS, Pug, Mock API, Axios.

Nuxt.js 2 [4] – це фреймворк, який об’єднує переваги Vue.js з можливостями серверного рендерингу та статичної генерації. Він особливо популярний серед розробників, які хочуть створювати SEO-оптимізовані та продуктивні вебзастосунки з гнучкою архітектурою.

Основні особливості:

– універсальність: Nuxt.js 2 дозволяє створювати як клієнтські SPA (Single Page Applications), так і SSR-додатки (це дозволяє поліпшити продуктивність і SEO);
– статичні генератори: за допомогою Nuxt.js 2 є можливість створювати статичні вебсайти, що робить його привабливим для розробки вебсайтів з високим трафіком;
– роутинг: Nuxt.js автоматично створює маршрути на основі файлової структури, що спрощує навігацію в проєкті;
– модульна архітектура: фреймворк має вбудовану підтримку модулів, дозволяючи легко інтегрувати різні технології та інструменти, як-от PWA, Axios, i18n, та багато інших;
– конфігурація: Nuxt.js пропонує можливості конфігурації через файл nuxt.config.js, де ви можете налаштувати різні параметри, такі як

глобальні стилі, компоненти, плагіни та інше;

– підтримка екосистеми Vue.js: Nuxt.js дозволяє використовувати Vue-компоненти, Vuex для керування станом, і Vue Router для роутингу.

JavaSript [5] – це динамічна, об’єктно-орієнтована прототипна мова програмування. Найчастіше використовується для створення сценаріїв вебсторінок, що надає можливість на стороні клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд вебсторінки .

CSS/SCSS [6] – є основним інструментом для стилізації вебсторінок, тоді як SCSS надає розширені можливості для розробників, що дозволяє створювати більш складні та гнучкі стилі. Використання SCSS допомагає зменшити дублювання коду та полегшує організацію стилів.

Основні аспекти CSS:

– синтаксис: CSS має простий синтаксис – правила CSS складаються з селектора, який визначає, до яких елементів застосовується стиль, та блоків властивостей і значень;

– каскадність: назва «каскадні стилі» походить від того, що стилі застосовуються у певному порядку (каскаді), і більш специфічні правила можуть перевизначати менш специфічні;

– селектори: CSS пропонує різні види селекторів, щоб вибирати елементи на сторінці, включаючи ідентифікатори (ID), класи, псевдокласи, атрибути та інші;

– медіа-запити: CSS підтримує медіа-запити, що дозволяє створювати адаптивний дизайн, змінюючи стилі залежно від розміру екрану або інших умов;

– анімація та перехід: CSS підтримує прості анімації та переходи, що дозволяє створювати динамічні ефекти.

Pug – це інструмент для розробки HTML-шаблонів, який спрощує синтаксис і сприяє модульному підходу до побудови вебсторінок. Він

особливо корисний у бекенд-розробці та в проєктах, де важлива підтримка і розширюваність шаблонів.

Особливості Pug:

– спрощений синтаксис: Pug використовує відступи замість відкриваючих і закриваючих тегів, що робить код компактним і легким для читання;
– вкладеність: є можливість використовувати відступи, щоб створювати вкладені структури HTML, що підвищує читабельність і зменшує ймовірність помилок;
– компоненти та міксини: Pug дозволяє створювати міксини, які є повторно використовуваними фрагментами шаблонів (це допомагає уникнути дублювання коду);
– динамічність: Pug дозволяє вставляти JavaScript-логіку у шаблони, що дозволяє створювати динамічний HTML – ви можете використовувати змінні, умови, цикли та інші конструкції;
– коротші атрибути: атрибути в Pug можна передавати у компактному форматі, без лапок, що зменшує кількість коду;
– модульність: є можливість створювати розділені шаблони та імпортувати їх у різних частинах вашого проєкту.

Mock API – інструмент для розробників та тестувальників, який дозволяє працювати з імітованими версіями API, забезпечуючи гнучкість і швидкість розробки. Вони допомагають ізолювати різні компоненти системи, полегшуючи тестування та розробку, особливо на ранніх етапах або в умовах відсутності доступу до реального сервера.

Інструменти для створення Mock API:

– Postman: інструмент для тестування API, який також дозволяє створювати Mock API – є можливість налаштувати відповіді на різні запити та тестувати клієнтський код без справжнього сервера;
– JSON Server: легкий інструмент, що дозволяє створювати REST API

з використанням JSON-файлу як джерела даних – він простий у налаштуванні та використанні;

– WireMock: потужний інструмент, який дозволяє створювати складні сценарії для імітації API – він також може використовуватися для тестування надійності системи;
– Mockoon: інструмент з інтуїтивним графічним інтерфейсом для створення та керування Mock API – підходить для розробників, які віддають перевагу GUI;
– Beeceptor: вебсервіс для створення Mock API в хмарі – підходить для спільної роботи над проєктами.

Приклади використання Mock API:

– розробка інтерфейсу: розробники можуть створити Mock API, який повертає очікувані дані, що дозволяє їм розробляти фронтенд навіть за відсутності реального бекенду;
– тестування: тестувальники можуть налаштувати Mock API, щоб імітувати певні умови, наприклад, помилку сервера або затримку відповіді;
– імітація сценаріїв: при розробці складних сценаріїв взаємодії з API Mock API допомагає імітувати різні відповіді, включно з успіхом, помилками та непередбаченими ситуаціями.

Axios – це гнучкий та зручний інструмент для роботи з HTTP-запитами. Він пропонує багато можливостей для налаштування та полегшує взаємодію з API у вебдодатках. Завдяки своїм можливостям і популярності, Axios став одним із стандартних інструментів для роботи з HTTP-запитами у світі JavaScript.

Особливості Axios:

– підтримка промісів: Axios заснований на промісах, що робить його сумісним з асинхронними операціями та дозволяє використовувати синтаксис async/await;

- широкий діапазон методів HTTP: ви можете використовувати всі стандартні методи HTTP, такі як GET, POST, PUT, DELETE, PATCH та інші;
- налаштування запитів: Axios дозволяє легко налаштовувати запити, включаючи заголовки (headers), параметри (params), час очікування (timeout), тощо;
- перехоплювачі (interceptors): Axios підтримує перехоплювачі запитів та відповідей, що дозволяє додавати логіку перед відправленням запиту або перед обробкою відповіді – це корисно для додавання токенів автентифікації, обробки помилок тощо;
- автоматична серіалізація/десеріалізація: Axios автоматично серіалізує дані перед відправкою запиту та десеріалізує відповіді з формату JSON;
- скасування запитів: Axios дозволяє скасовувати запити, що може бути корисно в ситуаціях, коли потрібен контроль над тривалістю запитів або при скасуванні запиту через користувацькі дії;
- крос-доменні запити (CORS): Axios підтримує крос-доменні запити, що дозволяє взаємодіяти з API на різних доменах, дотримуючись політики безпеки.

Axios можна використовувати як на фронтенді, так і на бекенді. Деякі сценарії, де Axios зазвичай застосовується:

- фронтенд-вебдодатки: використовується для взаємодії з API та вебсервісами, наприклад, для отримання даних або надсилання форм;
- бекенд-програми: може бути використаний на стороні сервера для виконання запитів до інших сервісів, наприклад, при створенні мікросервісів;
- тестування: Axios можна використовувати для тестування HTTP-запитів, створюючи імітовані запити для тестів.

Приклади використання Axios представлено на рисунках 2.1 – 2.3.

```
import axios from 'axios';

axios.get('https://api.example.com/data')
  .then(response => {
    console.log(response.data);
  })
  .catch(error => {
    console.error('Error:', error);
  });
```

Рисунок 2.1 – GET-запит

```
axios.post('https://api.example.com/data', { key: 'value' })
  .then(response => {
    console.log('Data submitted:', response.data);
  })
  .catch(error => {
    console.error('Error:', error);
  });
```

Рисунок 2.2 – POST-запит

```
axios.interceptors.request.use(config => {
  config.headers['Authorization'] = 'Bearer my-token';
  return config;
}, error => {
  return Promise.reject(error);
});
```

Рисунок 2.3 – Використання перехоплювачів

2.4 Середовище розробки

Для розробки сайту та написання коду було обрано середовище Microsoft Visual Studio (див. рис. 2.4).

Microsoft Visual Studi – це інтегроване середовище розробки (IDE), розроблене компанією Microsoft. Visual Studio надає набір інструментів для розробки програмного забезпечення на різних мовах програмування, платформах та технологіях.

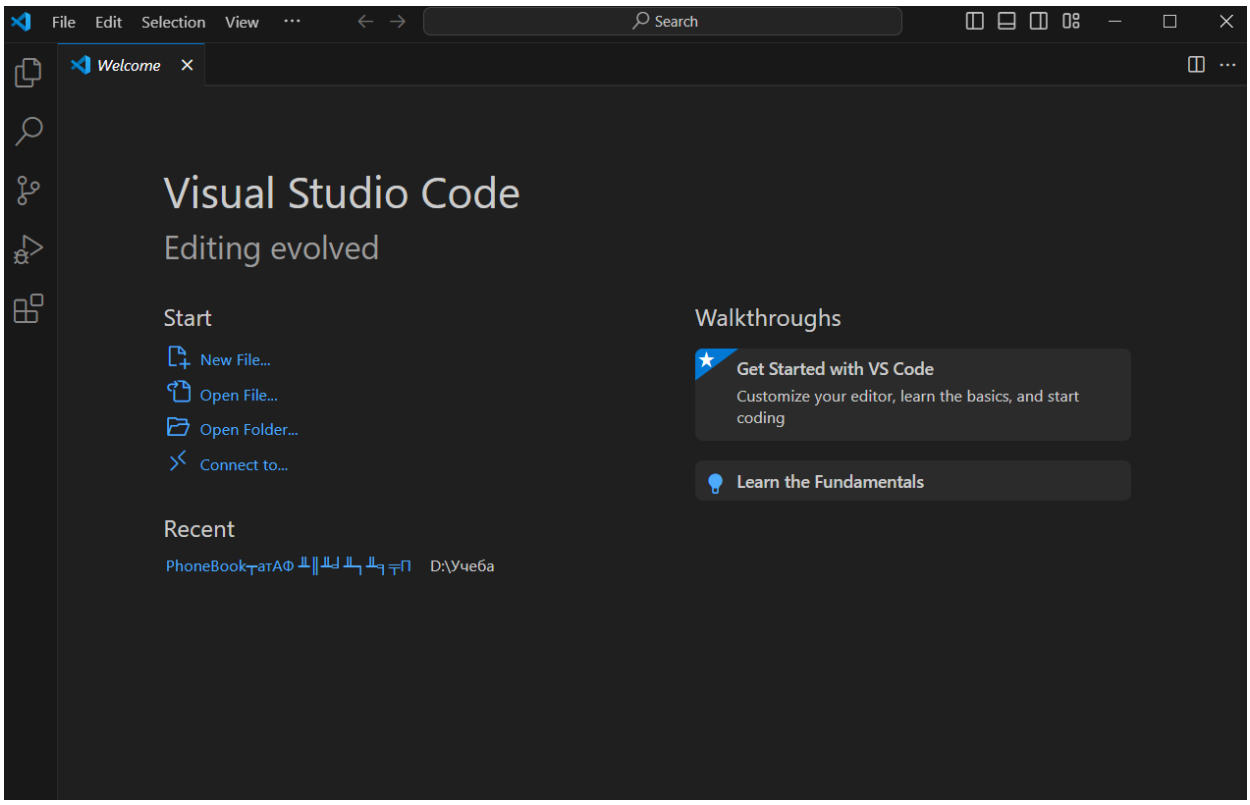

Рисунок 2.4 – Visual Studio

Основні функції та компоненти:

– редактор коду: Visual Studio має потужний редактор коду з розширеними можливостями, такими як підсвічування синтаксису, автодоповнення, перевірка помилок та інші;
– відлагоджувач (Debugger): інтегрований відлагоджувач дозволяє розробникам відстежувати виконання програми, виявляти помилки та вирішувати проблеми;
– компілятори та інструменти розробки: Visual Studio містить вбудовані компілятори та інструменти для розробки програмного забезпечення на різних мовах програмування, включаючи C++, C#, Visual Basic, Python, JavaScript тощо;
– підтримка платформ: Visual Studio дозволяє розробляти для різних платформ, включаючи Windows, Android, iOS, Linux та інші;
– розширення та плагіни: велика екосистема розширень та плагінів дозволяє розширювати можливості Visual Studio та адаптувати її під конкретні потреби розробника;

– інтеграція з Azure: Visual Studio інтегрується з платформою обчислення та хмарними послугами Azure, що дозволяє легко розгортати та керувати програмами у хмарі.

Використання Visual Studio:

– розробка вебдодатків та вебсайтів;
– розробка мобільних додатків для Android та iOS;
– розробка десктопних програм для Windows;
– розробка хмарних додатків та мікросервісів.

2.5 Прикладний програмний інтерфейс

Для розробки та написання сайту було використано Mock API (див. рис. 2.5).

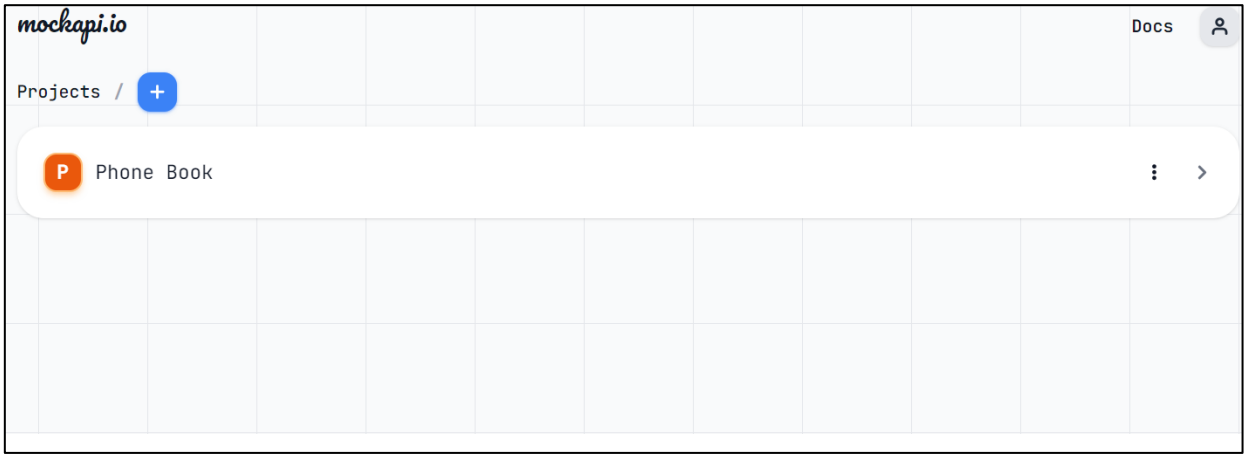

Рисунок 2.5 – Mock API

Інтерфейс програмування застосунків (API) – це набір правил та протоколів, що визначають, як різні програмні компоненти можуть взаємодіяти один з одним. API визначає набір команд, функцій та структур даних, які можуть бути використані для створення програмного забезпечення або інтеграції з іншими програмами або сервісами.

Види API:

– Web – це API, яке використовується для взаємодії з вебсерверами за допомогою HTTP-запитів (вони зазвичай використовуються для створення вебсервісів, які надають доступ до функціоналу або даних через Інтернет);
– бібліотечне – це API, яке визначає набір функцій або класів, які доступні для використання в програмному коді (наприклад, багато мов програмування мають стандартні бібліотеки з функціями для обробки рядків, роботи з файлами, мережами тощо);
– операційні системи API – операційні системи також мають свої API, які дозволяють програмам взаємодіяти з різними функціями та можливостями операційної системи, такими як керування файлами, мережами, процесами тощо.

Способи взаємодії API:

– HTTP-запити – багато API використовують протокол HTTP для взаємодії з клієнтами (клієнти можуть виконувати HTTP-запити (наприклад, GET, POST, PUT, DELETE) до сервера API для отримання або надсилання даних);
– REST API – це архітектурний стиль для створення вебсервісів, який базується на принципах HTTP (REST API використовує HTTP-методи та URL-шляхи для доступу до ресурсів);
– SOAP API – це протокол обміну повідомленнями, який використовує XML для форматування повідомлень. SOAP API використовується для створення вебсервісів, які надають доступ до функцій та даних.

2.6 Висновки до розділу 2

Nuxt.js 2 – це потужний фреймворк, який об’єднує переваги Vue.js з можливостями серверного рендерингу та статичної генерації. Він особливо

популярний серед розробників, які хочуть створювати SEO-оптимізовані та продуктивні вебзастосунки з гнучкою архітектурою.

CSS є основним інструментом для стилізації вебсторінок, тоді як SCSS надає розширені можливості для розробників, що дозволяє створювати більш складні та гнучкі стилі. Використання SCSS допомагає зменшити дублювання коду та полегшує організацію стилів.

Pug – це потужний інструмент для розробки HTML-шаблонів, який спрощує синтаксис і сприяє модульному підходу до побудови вебсторінок. Він особливо корисний у бекенд-розробці та в проєктах, де важлива підтримка і розширюваність шаблонів.

Mock API – потужний інструмент для розробників та тестувальників, який дозволяє працювати з імітованими версіями API, забезпечуючи гнучкість і швидкість розробки. Вони допомагають ізолювати різні компоненти системи, полегшуючи тестування та розробку, особливо на ранніх етапах або в умовах відсутності доступу до реального сервера.

Axios – це гнучкий та зручний інструмент для роботи з HTTP-запитами. Він пропонує багато можливостей для налаштування та полегшує взаємодію з API у вебдодатках. Завдяки своїм можливостям і популярності, Axios став одним із стандартних інструментів для роботи з HTTP-запитами у світі JavaScript.

Microsoft Visual Studio є потужним інструментом для розробки програмного забезпечення на різних мовах програмування та платформах. Він має широкий набір функцій та інструментів, що робить його популярним серед розробників у всьому світі. Visual Studio допомагає розробникам підвищити продуктивність та якість свого програмного забезпечення завдяки розширеному набору функцій та інтегрованим інструментам розробки.

3 РОЗРОБКА ТЕЛЕФОННОГО ДОВІДНИКА З ВИКОРИСТАННЯМ ФРЕЙМВОРКУ NUXT.JS 2

3.1 Код сайту телефонного довідника

Спочатку прописуємо STATE. STATE – це сховище, яке використовується для збереження даних і тд. Може використовуватись у середині компонентів для отримання даних за допомогою звернення до визначеної змінній (див. рис. 3.1).

```
export const state = () => ({
  contacts_data_default: [],
  contacts_data: [],
  pagination_pages: [],
  selected_pagination_page: 0
})
```

Рисунок 3.1 – STATE

Далі прописуємо отримання контактів, кількість сторінок для плагінації та отримання визначеної сторінки плагінації (див. рис. 3.2).

```
export const getters = {
  get_contacts(state) {
    return state.contacts_data // Получение всех контактов
  },
  get_pagination_pages(state) {
    return state.pagination_pages // Получение количества страниц для пагинации 1, 2, 3
  },
  get_selected_pagination_page(state) {
    return state.selected_pagination_page // Получение установленная страница пагинации
  }
```

Рисунок 3.2 – Отримання контактів

Після цього прописуємо логіку для отримання контактів, прописуємо стейт щоб не повторювались контакти та створюємо об’єкт для кожного контакту і підставляємо потрібні значення (див. рис. 3.3).

```
export const mutations = {
   contacts_data(state, contacts) { // Логика для получения всех контактов
      state.contacts_data = [] // Обнуляем стейт с контактами чтобы не было дубликации и тд
      contacts.forEach(contact => { // проходимся по полученным контактам и делаем форматирование в
удобный сампл для исопльзования
         const contactSample = { // Создаем объект для каждого контакта и подставляем под нужные
ключи нужные значения
            id: contact.id,
            updated_at: contact.updated_at,
            created_at: contact.created_at,
            full_name: contact.full_name,
            phone_number: contact.phone_number
         }
         state.contacts_data.push(contactSample) // Добавляем в contacts  data каждый отформатированный
контакт
         state.contacts_data_default.push(contactSample) // добавляем в массив дефолтных значении
контактов каждый контакт
      })
   },
   delete_contact(state, id) {
      state.contacts_data = state.contacts_data.filter((contact) => contact.id !== id) // логика удаления
контакта по айди
   },
```

Рисунок 3.3 – Логіка отримання контактів

Змінна даних контактів через ID (див. рис. 3.4).

```
edit_contact(state, data) {
      const newArr = [...state.contacts_data] // создаем копию контака без ссылки.
      newArr.forEach((contact,index) => { // проходимся по всем контакктах
         if (contact.id === data.id) { // находим нужны контакт по айди
            newArr[index] = { // подставляем измененые данные
               id: data.id,
               updated_at: data.updated_at,
               created_at: data.created_at,
               full_name: data.full_name,
               phone_number: data.phone_number
            }
         }
      })
      state.contacts_data = newArr// обновляем основной список кнотактов измененным
```

Рисунок 3.4 – Змінна даних контактів через ID

Сортування від нових до старих (див. рис. 3.5).

```
sort_data(state, sortType) {
      sortType === 'to_oldest' ? // проверка на тип сортировки
      state.contacts_data = state.contacts_data.sort(// если ту_олдест то сортируем контакты от новых к старым
          (a, b) => Date.parse(b.created_at) - Date.parse(a.created_at)
      ) :
      state.contacts_data = state.contacts_data.sort( // и на оборот
          (a, b) => Date.parse(a.created_at) - Date.parse(b.created_at)
      )
   },
```

Рисунок 3.5 – Сортування від нових до старих

Фільтрація контактів та пошук необхідного контакту за номером телефону (див. рис. 3.6).

```
find_contact(state, inputValue) {
      state.contacts_data = state.contacts_data_default
      const newArr = [...state.contacts_data]// копируем массив
      state.contacts_data = newArr.filter((contact) =>// делаем фильтрацию и находим нужный контакт по номеру телефона
          contact.phone_number.includes(inputValue)
      )
   },
```

Рисунок 3.6 – Фільтрація контактів

Додаємо новий контакт у список (див. рис. 3.7).

```
create_new_contact(state, newContact) {
      state.contacts_data.push(newContact)// добавляем новый контакт в список
   },
   set_pagination_pages(state, pagesCount) {
      state.pagination_pages = pagesCount// обновляем количество страниц пагинации
   },
   select_pagination_page(state) {
      state.selected_pagination_page = localStorage.getItem('selected_pagination_page')// сохраняем в локал сторедж выбранную страницу  пагинации
   }
```

Рисунок 3.7 – Додавання нового контакту

Запит для отримання усіх контактів по API (див. рис. 3.8).

```
xport const actions = {
    fetchAllContacts({commit}) {
        this.$axios// это запрос для получения всех контактов по апи
            .get(process.env.contacts_api).then((response) => {// достаем из среды переменных апи ссылку
            commit('contacts_data', response.data) // вызываем мутацию и передаем полученные данные
        })
    },
    deleteContact({commit}, id) {
        this.$axios.delete(`${process.env.contacts_api}/${id}`).then((response) => {// тот же способ
подключения к апи но уже обращаемся к айди и удаляем его
            if (response.status === 200) {// проверка статуса на успешность удаления
                commit('delete_contact', response.data.id)// обновление данных
                alert('Success')// уведомления что успешно
            }
        })
    },
editContact({commit}, data) {
        this.$axios.put(`${process.env.contacts_api}/${data.id}`, data.editedContact).then(response => {// по
принципе с удалением обращаемся к контакту по айди и передаем обновленные даныне
            commit('edit_contact', response.data)
            alert('Success')
        })
    },
    sortData({commit}, sortType) {
        commit('sort_data', sortType)// вызываем мутацию на сортировку и передаем тип сортировки
    },
    findContact({commit}, inputValue) {
        commit('find_contact', inputValue)// вызываем мутацию на поиск конкретного контактаа и передаем
вводное значение
    },
    createNewContact({commit}, newContact) {
        this.$axios.post(`${process.env.contacts_api}/`, newContact).then((response) => {// обращаемся к апи и
передаем данные нового контактка
            commit('create_new_contact', response.data)// Вызываем мутацию на обновление контактов
            alert('Success')
        })
    },
    setPaginationPages({commit}, pagesCount) {
        commit('set_pagination_pages', pagesCount)// вызываем мутацию на получение списка всех страниц
пагинации
    },
    selectPaginationPage({commit}) {
        commit('select_pagination_page')// вызываем мутацию на обновление страницы пагинации
    }
}
```

Рисунок 3.8 – Отримання усіх контактів

3.2 Демонстрація проєкту

Перша сторінка сайту на якій видно «блоки» контактів, фільтрацію, кількість сторінок та створення нових контактів (див. рис. 3.9).

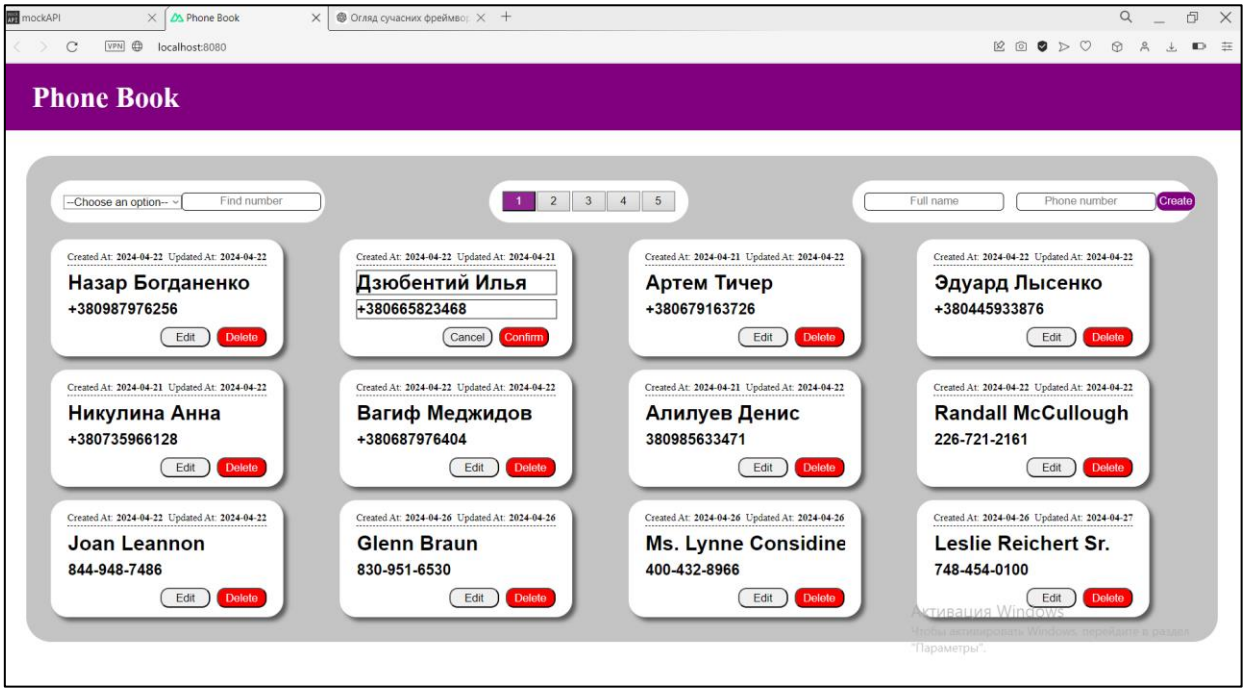

Рисунок 3.9 – Перша сторінка сайту

Контакти з фільтрацією «Від старих до нових» (див. рис. 3.10).

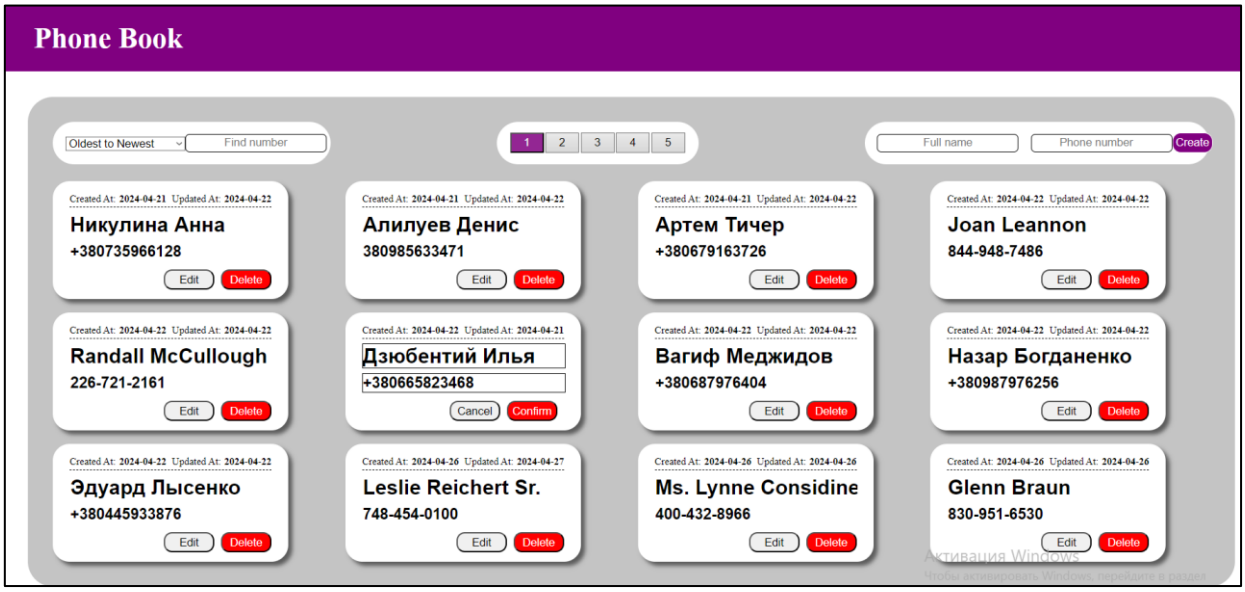

Рисунок 3.10 – Фільтрація контактів «Від старих до нових»

Створення нового контакту (див. рис. 3.11). Робимо сортування «Нових до старих» та бачимо наш створений контакт (див. рис. 3.12). Демонстрація віртуальної «бази даних», де зберігаються дані контактів, де можна змінити дані контактів, визначити кількість рандомних контактів та їх видалення (див. рис. 3.13).

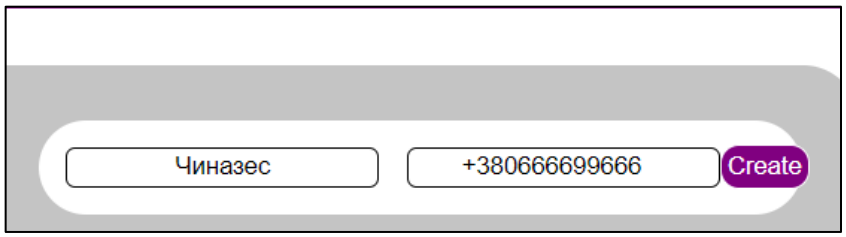

Рисунок 3.11 – Створення нового контакту

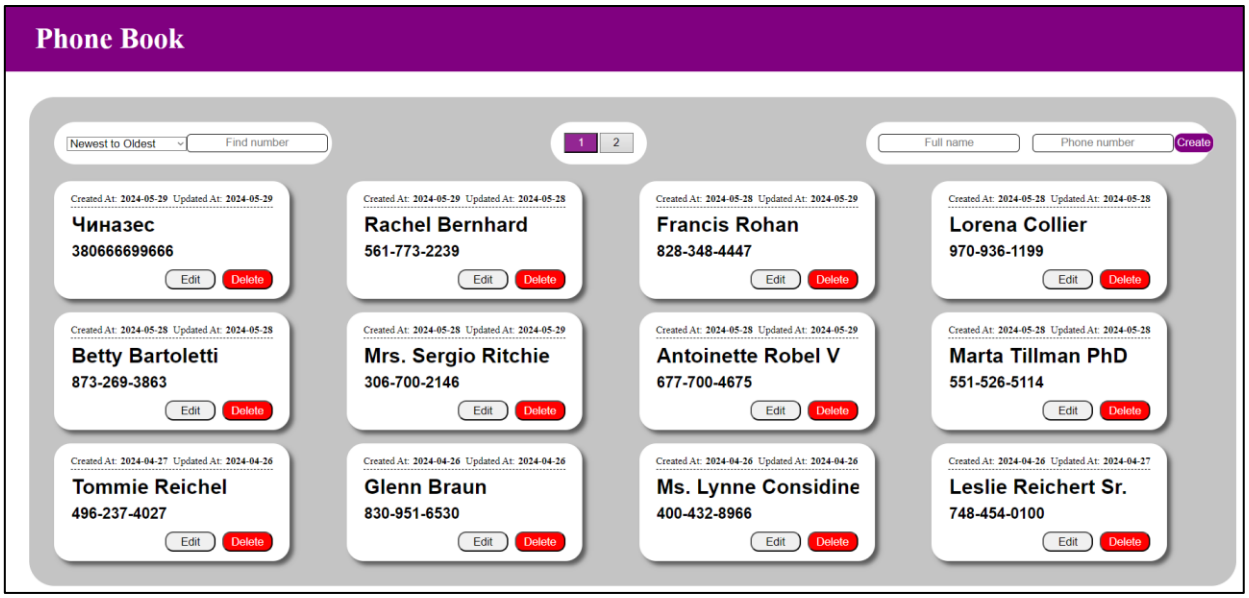

Рисунок 3.12 – Сортування контактів від «Нових до старих»

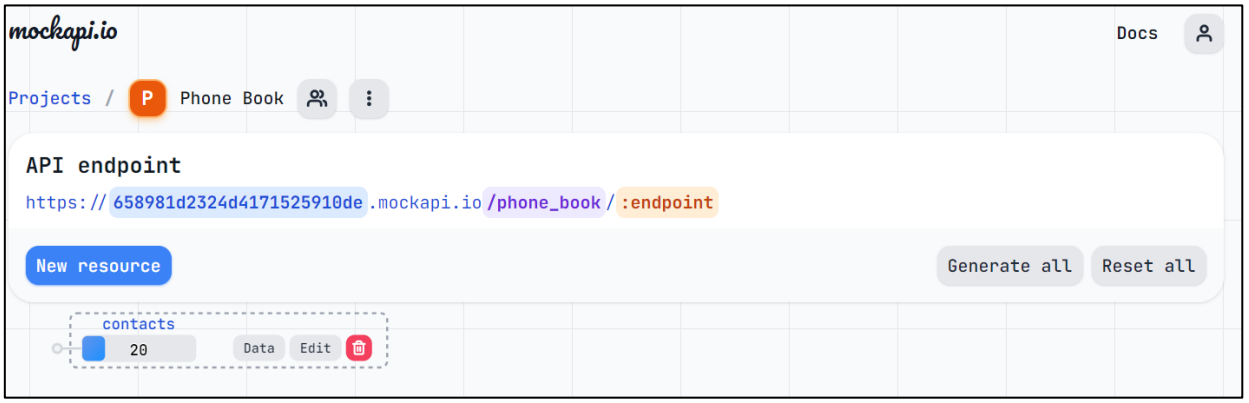

Рисунок 3.13 – Mock API

Так виглядає інформація про кожний контакт, який було створено та збережено (див. рис. 3.14).

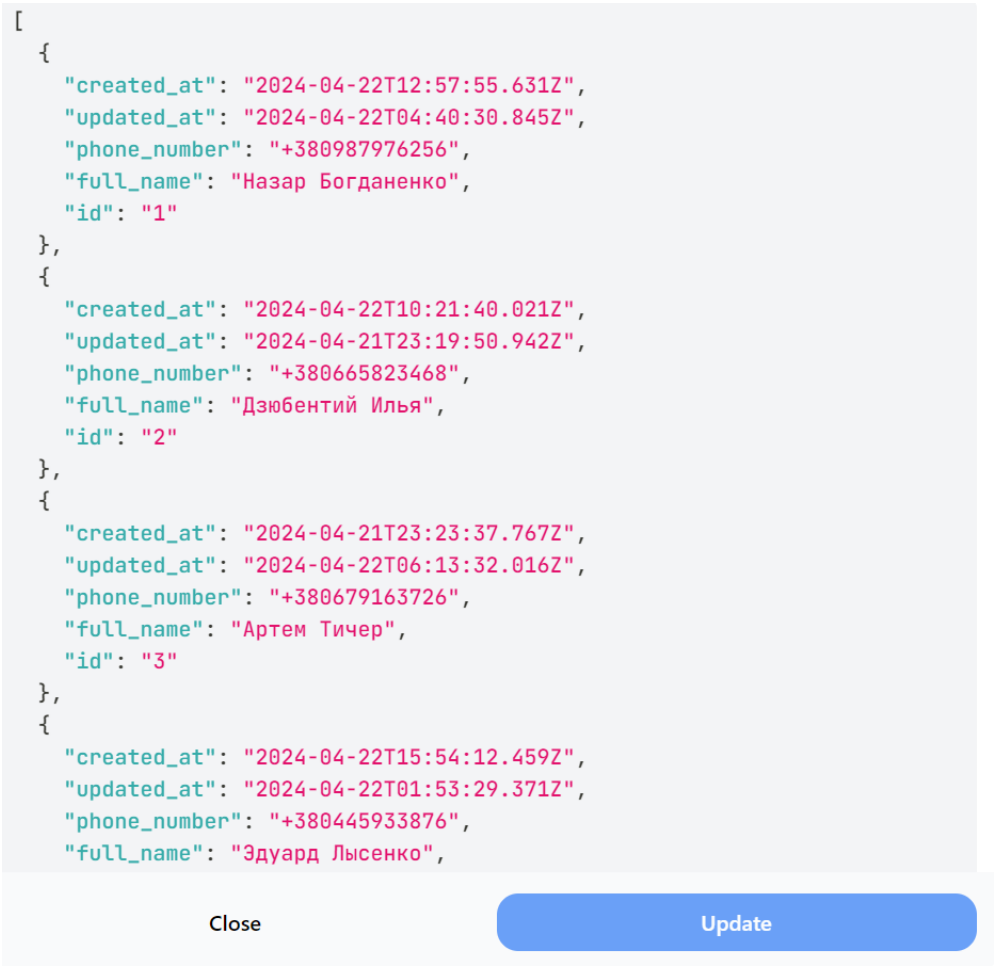

Рисунок 3.14 – Інформація про контакти у БД

3.3 Висновки до розділу 3

В цьому розділі було написано увесь код проєкту телефонного довідника. Було продемонстровано першу сторінку сайту на якій видно «блоки» контактів, фільтрацію, кількість сторінок та створення нових контактів. Контакти з фільтрацією «Від старих до нових». Створення нового контакту.

Також було зроблено сортування «Нових до старих» та видимість нашого створеного контакту.

Продемонстровано віртуальну «базу даних», де зберігаються дані контактів, де можна змінити дані контактів, визначити кількість рандомних контактів та їх видалення.

Також була показана інформація про кожний контакт, який було створено і збережено до сховища.

ВИСНОВКИ

Результатом виконання кваліфікаційної роботи є телефона книга , яка надає можливості створювати, редагувати та переглядати попередні контакти, зручно їх сортувати та гортати сторінками. Створений онлайн-сервіс є зручним, швидким і оптимізує процес зберігання контактів, що є необхідністю для людини, у якої дуже багато комунікацій з людьми за допомогою телефонного зв’язку.

Під час роботи було виконано наступні завдання:

– проведено огляд сучасних фреймворків;
– досліджено та обрано засоби розробки сайтів;
– розроблено проєкт;
– розроблено сайт та способи сортування та перегляду контактів, відповідно до проєкту.

Для створення телефонного довідника було розроблено систему для створення, зберігання, редагування, видалення та сортування контактів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Google Контакти. URL : https://contacts.google.com/ (дата звернення : 27.04.2024).
2. Microsoft Outlook Контакт. URL : https://outlook.live.com/mail/0/ (дата звернення : 27.04.2024).
3. Contacts+. URL : https://www.contactsplus.com/ (дата звернення : 27.04.2024).
4. Nuxt JS 2. URL : https://github.com/nuxt/nuxt (дата звернення : 27.04.2024).
5. Що таке JavaScript. URL : https://cases.media/article/sho-take-javascript (дата звернення : 27.04.2024).
6. Що таке CSS. URL : https://css.in.ua/article/shcho-take-html_10 (дата звернення : 27.04.2024).