Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

“Інформаційна веб-система аналізу динаміки географічної структури зовнішньої торгівлі товарами України”

Виконав: студент 2 курсу, групи 2ІСТ-21м
спеціальності 126 – «Інформаційні системи та технології»
_____ Пасєка Б. В.

Керівник: к.т.н., доц. каф. САІТ
_____Крижановський Є. М.
«01» ____12____ 2022 р.

Опонент: к.т.н., доц. каф. АІІТ Богач І.В.
«15» ____12____ 2022 р.

Допущено до захисту

Завідувач кафедри САІТ
_____ д.т.н., проф. Мокін В. Б.
«05» ____12____ 2022 р.

Вінниця ВНТУ – 2022 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій
Рівень вищої освіти – ІІ-й (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 126 Інформаційні системи та технології
Освітньо-професійна програма – Інформаційні технології аналізу даних та зображень

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
_____ д.т.н., проф. Мокін В. Б.
«16» 09 2022 р.

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Пасєці Богдану Володимировичу

1. Тема роботи: **“**Інформаційна веб-система аналізу динаміки географічної структури зовнішньої торгівлі товарами України”,
керівник роботи: Крижановський Є. М., к.т.н., доц. каф. САІТ,
затверджені наказом закладу вищої освіти від «14» 09 2022 року №205
2. Строк подання студентом роботи «01» 12 2022 року
3. Вихідні дані до роботи:
Дані управління статистики по кожній з областей України.
4. Зміст текстової частини:
– аналіз зовнішньої торгівлі товарами;
– формування програмних вимог та побудова концепції продукту;
– збирання та систематизація даних зовнішньої торгівлі товарами України;
– розроблення інформаційної веб-системи аналізу зовнішньої торгівлі товарами України;
– економічна частина.
5. Перелік ілюстративного матеріалу:
– схема обробки даних MongoDB
– огляд та редагування інформації в базі даних
– робота з картою та побудова статистичних діаграм
– побудова графіків статистики
– побудова комплексних статистичних графіків
– модель роботи веб-сервісу

6. Консультанти розділів МКР

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
5	Бурєннікова Н. В., д.е.н. професор каф. ЕПВМ	10.11.22р.	25.11.22р.

7. Дата видачі завдання «16» 09 2022 року

КАЛЕНДАРНИЙ ПЛАН

з/п	Назва етапів МКР	Строк виконання етапів роботи	Примітк
	Аналіз зовнішньої торгівлі товарами	09.2022	
	Формування програмних вимог та побудова концепції продукту	10.2022	
	Збирання та систематизація даних зовнішньої торгівлі товарами України	10.2022	
	Розроблення інформаційної веб-системи аналізу зовнішньої торгівлі товарами України	10.2022	
	Економічна частина	11.2022	
	Оформлення матеріалів до захисту МКР	11.2022	

Студент _____ Пасєка Б. В.

Керівник роботи _____ Крижановський Є.

АНОТАЦІЯ

УДК 004.08

Пасєка Б. В. Інформаційна веб-система аналізу динаміки географічної структури зовнішньої торгівлі товарами України. Магістерська кваліфікаційна робота зі спеціальності 126 – інформаційні системи та технології, освітньо-професійна програма – інформаційні технології аналізу даних та зображень. Вінниця: ВНТУ, 2022. 106 с.

На укр. мові. Бібліогр.: 26 назв; рис.: 50; табл.: 7.

В магістерській кваліфікаційній роботі було проведено аналіз предметної області, визначено суть поставленої задачі, проведено огляд існуючих методів вирішення технічної проблеми та визначено їхні переваги та недоліки. В результаті чого було сформовано низку програмних вимог та концепцій майбутньої веб-системи.

Проведено статистичне збирання даних, здійснено їх систематизацію та конвертацію в необхідний формат для подальшої розробки веб системи.

Також проаналізовано технології, які б дозволили реалізувати всю функціональність веб системи. Після чого розроблено систему аналізу динаміки географічної структури зовнішньої торгівлі товарами України.

Проведено статистичний аналіз закордонного ринку та виявлено, що Україна має досить непогану тенденцію до розвитку міжнародних торгово-економічних відносин, має досить багато напрямків до збуту та закупівлі товарів, що в свою чергу приводить товарообіг до поступового зросту.

Ключові слова: зовнішня торгівля, аналіз закордонного ринку, розробка аналітичної системи, веб-технології.

ABSTRACT

Pasieka B. V. Information web system for analyzing the dynamics of the geographical structure of foreign trade in goods of Ukraine. Master's qualification thesis on specialty 126 - information systems and technologies, educational and professional program - information technologies of data and image analysis. Vinnytsia: VNTU, 2022. 106 p.

In Ukrainian speech Bibliography: 26 titles; fig.: 50; tab.: 7.

In the master's qualification work, an analysis of the subject area was carried out, the essence of the task was determined, an overview of the existing methods of solving the technical problem was carried out, and their advantages and disadvantages were determined. As a result, a number of program requirements and concepts of the future web system were formed.

Statistical data collection was carried out, their systematization and conversion into the necessary format for further development of the web system was carried out.

The technologies that would allow implementing all the functionality of the web system were also analyzed. After that, a system of analysis of the dynamics of the geographical structure of foreign trade in goods of Ukraine was developed.

A statistical analysis of the foreign market was conducted and it was found that Ukraine has a fairly good tendency to develop international trade and economic relations, has quite a lot of directions for sales and purchase of goods, which in turn leads to a gradual increase in turnover.

Keywords: foreign trade, foreign market analysis, analytical system development, web technologies.

ЗМІСТ

ВСТУП

Актуальність теми. З року в рік стрімкий розвиток інформаційних технологій все більше охоплює сфери людської діяльності. Розвиток інформаційних технологій є одним з головних інструментів підвищення ефективності будь-яких операцій. І закордонний ринок в даному випадку не є винятком.

Автоматизована система аналізу даних як засіб досягнення швидкості та плавності обміну інформацією. Такий інструмент має потенціал зменшення невизначеності пов'язані з веденням бізнесу на зовнішніх ринках. Оскільки невизначеність щодо іноземних ринків вважається головною перешкодою до розширення зовнішнього ринку, в руках менеджерів може бути інструментом, що прискорює розширення зовнішнього ринку.

Для підвищення ефективності ринкових операцій виникає потреба у створенні ресурсу який би значно зменшив пошукові витрати на збут товару та збільшив їх здатність гнучко реагувати на нові ринкові зміни, за рахунок автоматизації та статистичного аналізу.

Метою роботи є покращення комплексності аналізу структури зовнішньої торгівлі країни в розрізі областей шляхом розроблення інформаційної веб-системи аналізу динаміки географічної структури зовнішньої торгівлі товарами України.

Розробка інформаційної системи аналізу динаміки географічної структури зовнішньої торгівлі товарами України передбачає виконання наступних задач:

– здійснення статистичного збирання даних та створення бази даних, яка забезпечить інформаційної веб-системи необхідною інформацією;
– вибір оптимальних інформаційних технологій;
– розробка архітектури інформаційної веб-системи;
– розробка серверної частини інформаційної веб-системи;
– розробка клієнтської частини інформаційної веб-системи.

Об'єктом дослідження є процес аналізу динаміки географічної структури зовнішньої торгівлі товарами України.

Предмет дослідження методи автоматизації аналізу динаміки географічної структури зовнішньої торгівлі товарами України.

Методи дослідження. У дослідженнях використовувались методи ГІС-технологій для формалізації просторових даних ГІС, методи баз даних для формалізації та збереження атрибутивних даних. Також було здійснено програмну реалізацію з використанням сучасних веб-технологій.

Новизна одержаних результатів. Дістав подальший розвиток підхід до створення інформаційної веб-системи аналізу динаміки географічної структури зовнішньої торгівлі товарами України, що дозволить покращити аналіз структури зовнішньої торгівлі країни в розрізі областей. Результати роботи мають цінність для економічного аналізу структури зовнішньої торгівлі країни в розрізі областей.

Апробація результатів магістерської кваліфікаційної роботи. Результати роботи доповідались на всеукраїнській науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2022-2023 рр.).

Публікації результатів магістерської кваліфікаційної роботи. Опубліковано тези на Всеукраїнській науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2022-2023 рр.) [1].

1 АНАЛІЗ ЗОВНІШНЬОЇ ТОРГІВЛІ ТОВАРАМИ

1.1 Суть технічної проблеми

Розвиток зовнішньоекономічної діяльності невід'ємно пов'язаний з процесом глобалізації, розвитком економічних зв'язків між країнами, оновленням технічного забезпечення підприємств та підвищенням рівня якості продукції [2].

Правильно обрана стратегія, вірно поставлена місія, цілі та задачі. Саме це дозволить ефективно функціонувати на зовнішньому ринку, що як наслідок в майбутньому призведе до першості в даній сфері.

Задля досягнення даної мети виникає потреба, для підвищення ефективності ринкових операцій. Завдяки використанню передових інформаційних систем компанії можуть значно зменшити свої пошукові витрати та підвищити здатність гнучко реагувати на нові ринкові можливості в результаті зменшення витрат. На основі теорії інтернаціоналізації та теорії агентських витрат / трансакційних витрат у роботі розглядається вплив передових інформаційних систем на розширення області торгівлі на зовнішньому ринку [3].

Ці теоретичні питання та можливі наслідки для інтернаціоналізації проілюстровані через представлення трьох загальних прогнозів щодо можливих результатів. Прогнози демонструють, що цілий ряд зовнішніх наслідків розширення інформаційних систем: від обмеженого впливу до швидкої, широкомасштабної глобальної експансії у багатьох випадках [3].

Одним з найбільш контрольних факторів зовнішнього ринку є менеджмент. Для менеджерів дуже важливо визнати відмінності, а також подібність у поведінці покупця, відслідковування трендів ринку та його занепад. Багато помилок може статися, якщо менеджери не усвідомлюють, що покупці відрізняються від країни до країни. Саме міжнародні відмінності в

поведінці покупця, а не схожість, спричиняють проблеми у успішному міжнародному маркетингу [4].

Міжнародний менеджер з маркетингу − це людина, відповідальна за сприяння обміну продуктами між організацією та її клієнтами чи клієнтами. Іноді менеджер з міжнародного маркетингу відчуває труднощі у завершенні обміну продуктами. Багато сюрпризів у міжнародному бізнесі є небажаними людськими помилками. Міжнародна корпорація повинна повністю розуміти закордонне середовище, перш ніж займатися діловими справами. Проблеми постійно виникають і багато разів мають несподівані результати [5].

Також як встановлено, значним чинником впливу на закордонну торгівлю відіграє культура. З усіх культурних аспектів спілкування може бути найбільш критичним. Певно, що спілкування спричинило низку культурних плутанин. Необхідно встановити належні комунікаційні зв'язки між компаніями та її замовниками, постачальниками, її працівниками та урядами країн, де вона здійснює господарську діяльність. Погана комунікація, очевидно, може спричинити різні труднощі.

Одним із джерел труднощів у починаючих компаній є ефективне спілкування з потенційними покупцями. Проблема в тому, що існує багато можливих комунікативних бар'єрів. Іноді повідомлення можна перекладати неправильно, не враховувати нормативні акти та ігнорувати економічні відмінності. Інший раз, коли повідомлення надходить, його неефективність може призвести до того, що він не матиме значення. Час від часу повідомлення отримує покупець, але, на розчарування компаній, повідомлення було надіслано неправильно. Нормально, що багатонаціональний бізнес регулярно надсилає та отримує повідомлення. Багато відомих людей недієздатні до публічних виступів, використовуючи неточні заголовки та імена. Не всі проблеми спілкування є словесними. Деякі серйозні проблеми виникли в результаті невербального спілкування [5].

Невербальна комунікація існує у численних формах. Іноді зовнішність людини може передати сильніше повідомлення, ніж передбачалося. Неохайний одяг, наприклад, може бути більш образливим для одних країн, ніж для інших. Місцеві жителі часто готові пропустити більшість помилок, допущених туристом. З іншого боку рука, місцеві жителі менш терпимі до помилок ділових людей. Дуже важливо вміти інтерпретувати різні засоби комунікації в міжнародному маркетингу [6].

Ще одним викликом міжнародного бізнесу є управління співробітниками, які живуть по всьому світу. Намагаючись функціонувати як команда, може бути важко врахувати мовні бар'єри, культурні відмінності, часові пояси та різний рівень доступу до технологій і надійності.

Щоб побудувати та підтримувати міцні робочі стосунки зі своєю глобальною командою , сприяйте регулярним реєстраціям, бажано використовуючи платформу для відеоконференцій, щоб ви могли взаємодіяти в режимі реального часу.

Дослідження Gallup показує, що працівники, які регулярно зустрічаються зі своїми керівниками, мають утричі більше шансів бути залученими на роботі, ніж ті, хто цього не робить.

Коли відстань розділяє команди, як це сталося з багатьма під час пандемії коронавірусу (COVID-19) , спілкування є ключовим для того, щоб кожен почувався цінним і зацікавленим.

Обмін валют та темпи інфляції теж певною мірою вносять свій вплив у розвиток закордонного ринку.

Вартість долара у країні не завжди дорівнюватиме тій самій сумі у валюті інших країн, а також вартість валюти не завжди буде коштувати однакову кількість товарів і послуг [6].

Також важливо відстежувати темпи інфляції , які є темпами зростання загального рівня цін в економіці рік за роком, виражені у відсотках. Рівень інфляції різниться в різних країнах і може впливати на витрати на матеріали та робочу силу, а також на ціни на продукцію.

Розуміння та уважне дотримання цих двох курсів може надати важливу інформацію про вартість продукту вашої компанії в різних місцях з часом.

Бізнес не існує у вакуумі — на нього впливають також політика, політики, закони та відносини між країнами. Оскільки ці стосунки можуть бути надзвичайно різноманітними, важливо уважно стежити за новинами, пов’язаними з країнами, де ви ведете бізнес [6].

Рішення, прийняті політичними лідерами, можуть вплинути на податки, трудове законодавство, вартість сировини, транспортну інфраструктуру, освітні системи тощо.

Один із гіпотетичних прикладів, який Рейнхардт наводить у Global Business, полягає в тому, що якби китайський уряд вирішив субсидувати китайські молочні ферми, це вплине на молочних фермерів у всіх сусідніх країнах. Це пояснюється тим, що за додаткового фінансування китайські молочні ферми можуть виробляти надлишок молочних продуктів, що змусить їх розширювати свої ринки на сусідні країни.

Це водночас і захоплююче, і лякаюче, що нюанси міжнародної політики, політики та відносин можуть вплинути на ваш бізнес. Будьте в курсі та приймайте стратегічні рішення, коли надходить нова інформація [6].

Основуючись на даних вищезгаданих проблемах виникає потреба в створені автоматизованого ресурсу, який на основі вибірок з бази даних буде проводити комплексний аналіз ринку, як імпорту так і експорту, будувати статистичні графіки та матиме можливість редагувати вихідні дані.

1.2 Огляд існуючих методів вирішення технічної проблеми

На сьогоднішній день у вільному доступі систем управління зовнішньоекономічною діяльністю чи просто менеджменту маркетингу не так багато, проте все таки вдалося знайти аналоги поставленої задачі. Нижче наведено деякі з них.

Intervals – це веб-додаток для управління проектами з відстеженням часу, управлінням завданнями та вичерпними функціями звітування. Це просте та потужне рішення, яке не надто спрощується, але також не надто складне. Вона виникла із традиційної системи придбання квитків, а отже, перевершує робочий процес та управління життям завдань. Його відстеження часу та управління завданнями є глибоко інтегрованими, так що час відстеження є більш природним. Завдяки цій інтеграції звітування також є більш потужним, здатним розбити робочий день за проектами та завданнями.

Особливості даного аналогу це відстеження часу, управління завданнями та проектами – інтегрує відстеження часу та управління завданнями. Функції відстеження часу включають таймери завдань, щотижневе подання та затвердження табеля обліку робочого часу, нагадування, що генеруються системою, про минулі терміни та візуальні звіти.

Звітування та виставлення рахунків – надає детальні високі рівні, а також детальні звіти та інформацію. Доступні як візуальні, так і табличні звіти, які можна експортувати у формат CSV та XML для полегшення роботи з електронними таблицями.

Зберігання документів, персоналізація, API та багато іншого. Intervals також має функції зберігання та спільного використання документів для легкої співпраці (рис. 1.1). Це дозволяє командам зберігати всі файли в тому самому місці, де вони управляють та працюють [7].

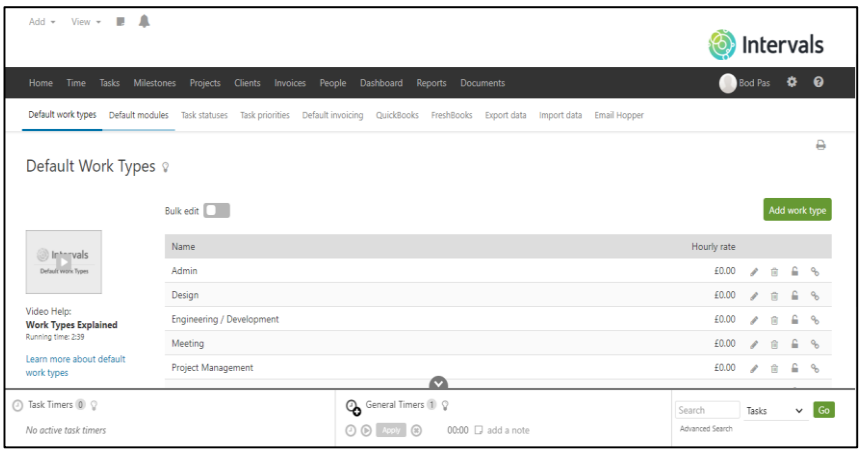


Рисунок 1.1 – Інтерфейс веб-додатку Intervals

Projector PSA − це спеціально розроблений хмарний інструмент автоматизації професійних послуг для організацій, які прагнуть покращити норму прибутку та коефіцієнт використання ресурсів, отримуючи при цьому знання про доходи та персонал. Завдяки гнучким ціноутворенням, які відповідають вашому бізнесу, мультивалютним, мультикомпанійним можливостям та технологічно-агностичному підходу до інтеграції, Projector забезпечує розумнішу та вигіднішу доставку, щоб ви могли розвивати свій бізнес.

Projector PSA розроблений для максимальної гнучкості, що дозволяє легко адаптувати програмне забезпечення до конкретного бізнес-середовища, не жертвуючи жодною надійною функціональністю.

До недоліків даного продукту можна віднести малу пропозицію бізнес-аналітики (Projector BI). Однак команда з Projector перебуває в процесі переходу на більш краще рішення.

Використовуючи різні інформаційні панелі та звіти в Інтернеті, Projector надає неоціненну інформацію щодо чотирьох основних проблемних питань − використання, прибутковість проекту, реалізація проекту та зростання доходу. Крім того, активуючи всі три доступні модулі в Projector (Бухгалтерський облік проектів, Управління ресурсами та Управління проектами), можна зменшити роботу над проектом, покладаючись на "Single Source of Truth" (рис. 1.2, 1.3), і усуваючи необхідність вести кілька електронних таблиць поза проектором [8].

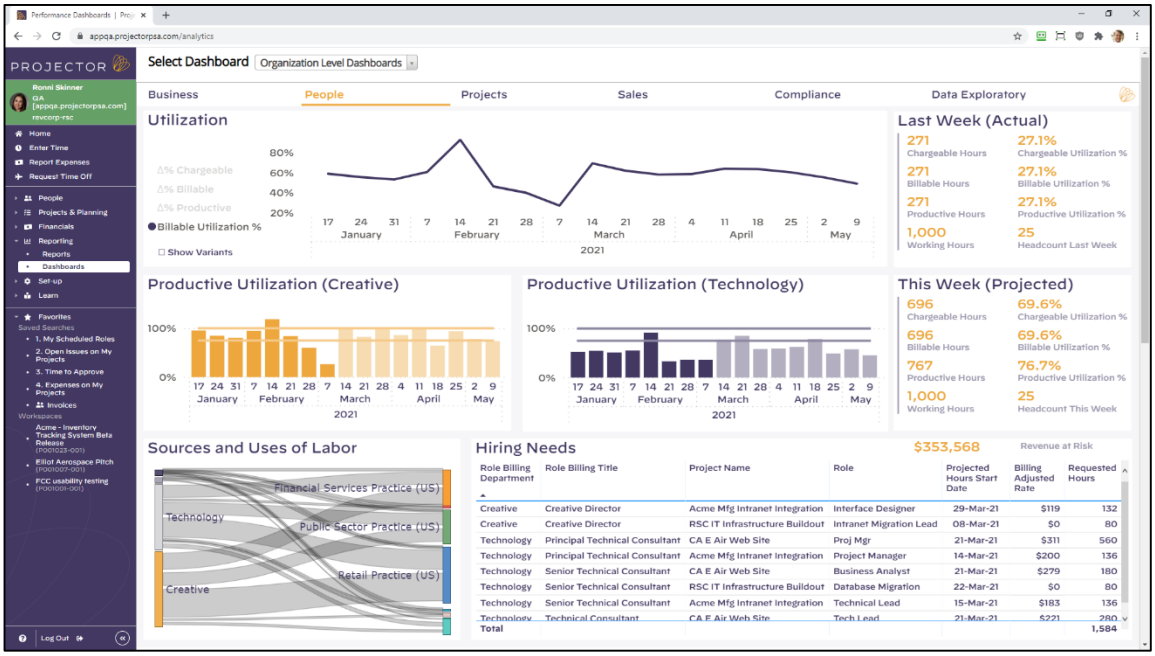


Рисунок 1.2 – Аналітична частина веб-додатку Projector PSA

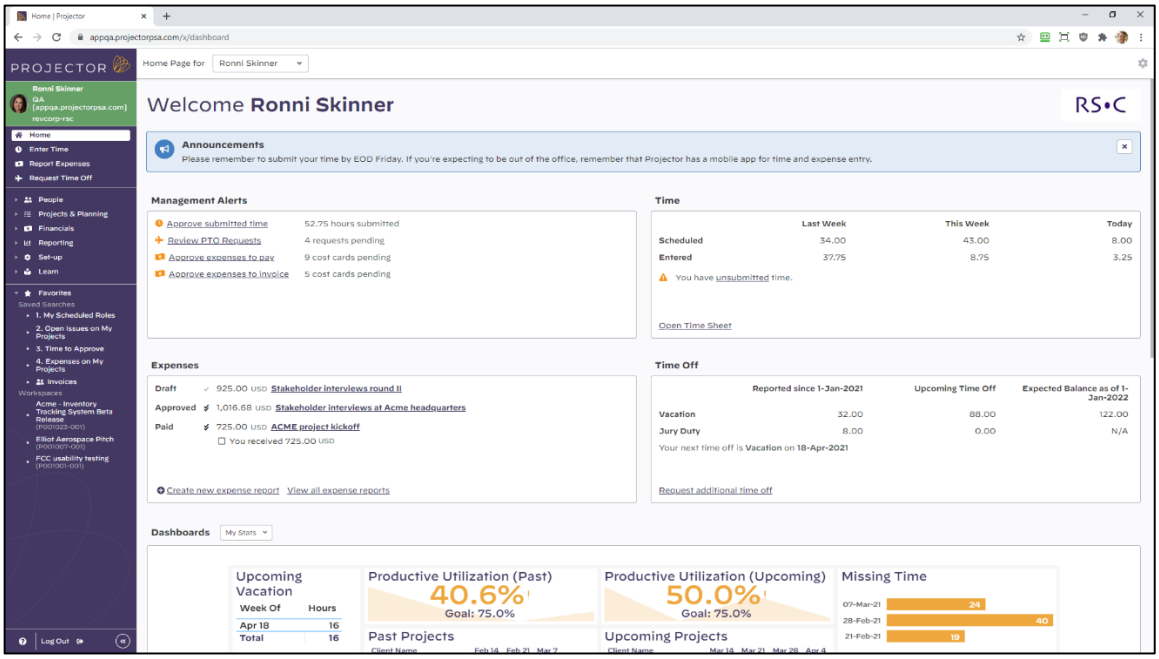


Рисунок 1.3 – Базовий інтерфейс управління ресурсами у веб-додатку Projector PSA

Workfront − це сучасна платформа управління роботою, яка об’єднує людей у роботу та допомагає організаціям досягти успіху. Це онлайн-інструмент, який допомагає зв’язати команди, завдання та проекти. Він підтримує інтелектуальне планування, впорядковані процеси та звітування в режимі реального часу для оптимізації робочого середовища. Він забезпечує

безперебійну інтеграцію, гнучкий підхід та індивідуальні рішення, завдяки чому організації можуть адаптуватися до сучасних викликів бізнесу. Це розумне рішення для команд у різних галузях промисловості, яке забезпечує працівника, що займається знаннями, інструментами, що сприяють видимості, продуктивності та мотивації (рис. 1.4).

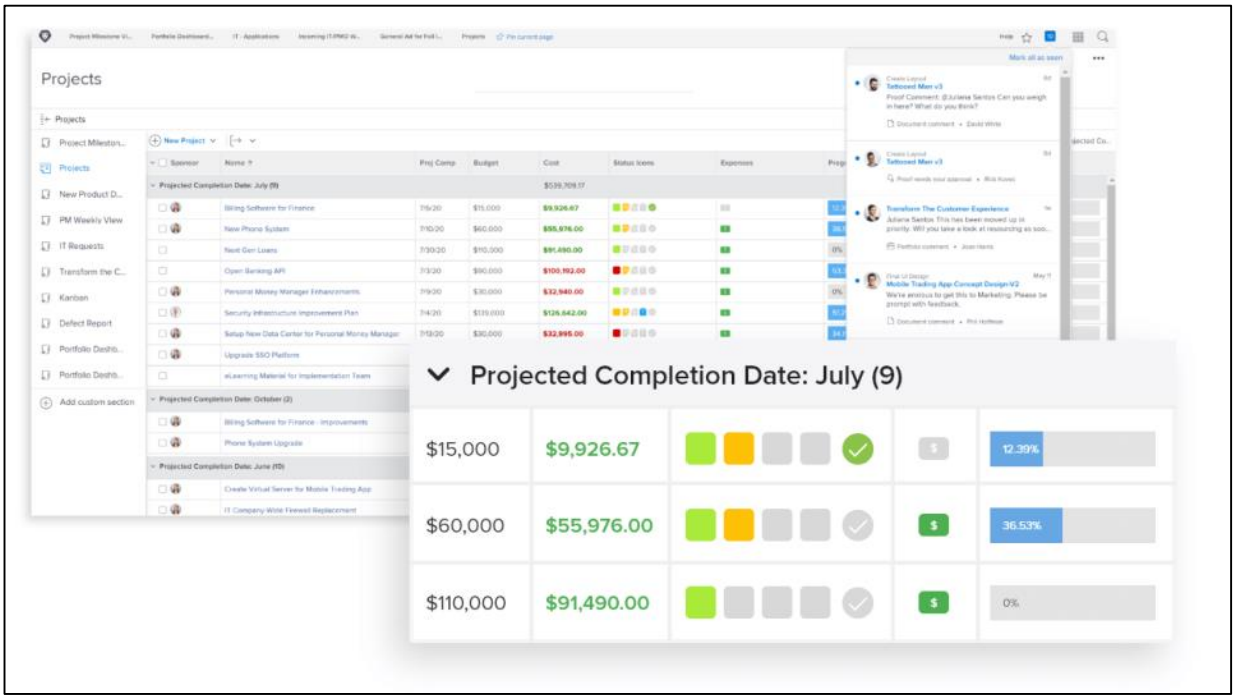


Рисунок 1.4 – Структура управління проектом у веб-додатку Workfront

Особливості та переваги у Workfront наступні:

– Управління проектами у Workfront виходить за рамки основних функцій проектного менеджменту. Автоматизація проектів за допомогою настроюваних шаблонів дозволяє легко враховувати кожен крок процесу та перетворювати нові робочі запити в проекти за лічені секунди;

– Інтелектуальна автоматизація роботи, контекстна співпраця. Рішення для управління роботою на підприємстві має функції управління попитом. Він може точно охоплювати проект та визначати пріоритети завдань за допомогою автоматизованих черг запитів та спеціальних форм;

– Звітування, Fusion, DAM, інтеграції тощо. Workfront надає користувачам стандартні звіти та власні інформаційні панелі для відстеження

важливих показників, таких як час, бюджет та рентабельність інвестицій. Вони можуть розбивати фінансові дані, такі як заплановані порівняно з фактичними годинами ресурсів, витратами бюджету та іншими даними, щоб зберегти нормальну норму прибутку.

Проте незважаючи на всі переваги даного продукту, на мою думку, основним недоліком є занадто високі тарифні ціни на користування.

Attest – це розумна платформа, яка допоможе отримати дані від реальних споживачів, що стосуються бізнесу, зібрані в режимі реального часу. Усе для того, щоб приймати бізнес-рішення швидше та з більшою впевненістю.

Attest дозволяє охопити цільову аудиторію на 49 ринках. Можна знайдети інструменти аналізу для кожного аспекту ринку: профілювання споживачів , міжнародне дослідження та аналіз ринку. Attest підходить як для новачків, так і для професіоналів.

Результати, які отримуються за допомогою інструментів дослідження, є надрелевантними [10].

Дані Attest демонструють явні зміни в сприйнятті та обізнаності споживачів. На рисунку 1.5 зображено інтерфейс інструменту Attest.

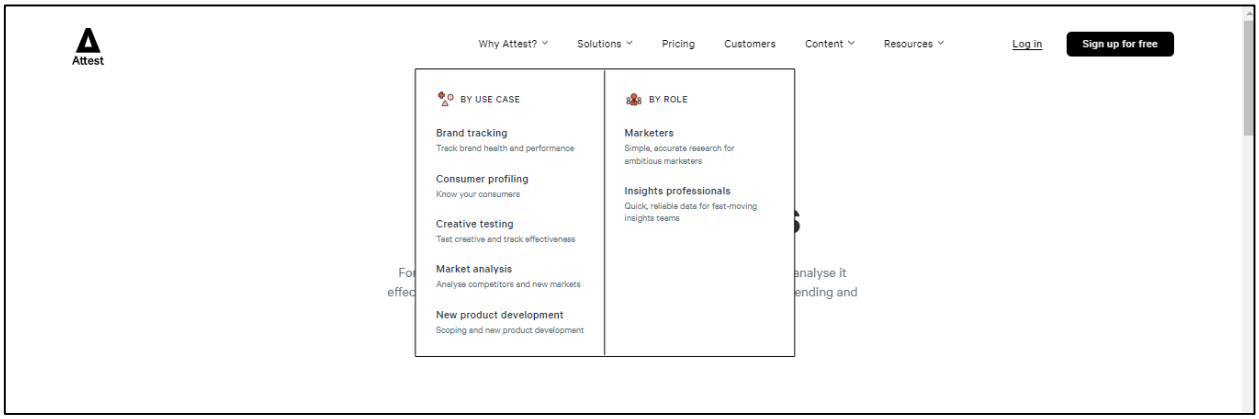


Рисунок 1.5 – Інструмент для аналізу ринку Attest

Незважаючи на всі переваги даного продукту, в ньому є і недоліки. Основним з них є досить не зручний інтерфейс, користувачу не одразу зрозуміло, де знаходиться необхідний функціонал.

Ще один недолік, який є в даному продукті це ціна, порівнюючи з аналогами, тарифний план інколи перевищує в два рази.

Statista — це портал даних про ринок і споживачів, який охоплює величезну кількість галузей промисловості по всьому світу.

Statista надає статистичні дані з багатьох тем, зокрема ЗМІ, бізнесу, політики, суспільства, технологій та освіти. Джерела включають ринкові звіти, дослідницькі установи, спеціалізовані видання, наукові журнали та державні установи. Діаграми можна завантажити у форматах PNG, PowerPoint, Excel або PDF або вставити на веб-сторінки, і вони можуть бути чудовими в документах або презентаціях.

Особливості:

– Статистика за темами;
– Звіти по країні;
– Досьє зі статистикою та звітами за темами;
– Галузеві звіти;
– Діаграми, зображення, інфографіка;
– Створення цитати в стилях APA, Chicago та MLA [11].

На рисунку 1.6 зображено інтерфейс інструменту Statista.

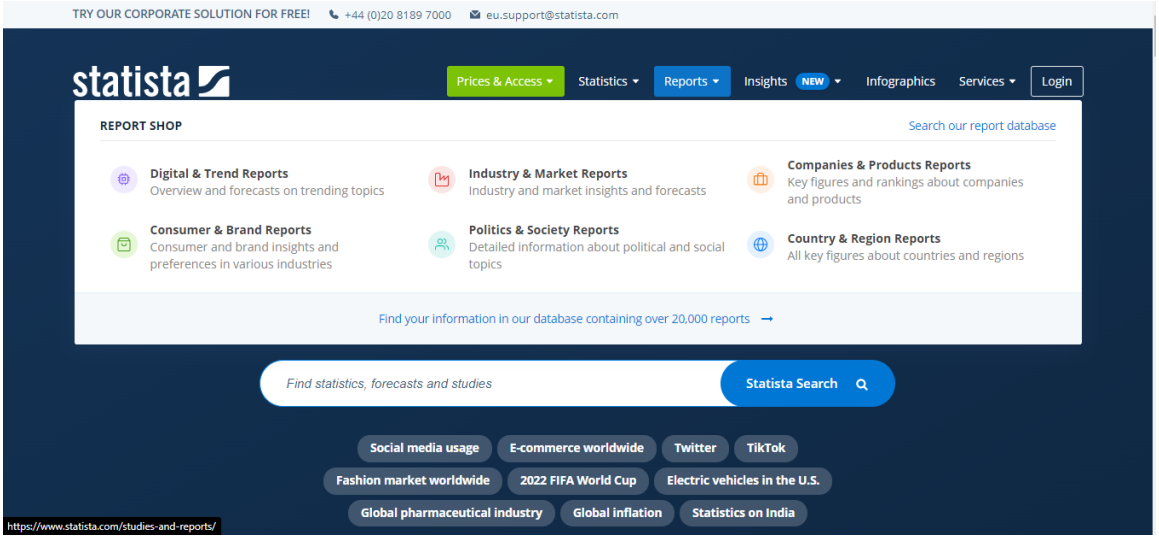


Рисунок 1.6 – портал даних про ринок і споживачів Statista

Незважаючи на перелічені переваги, є в даного аналогу і недоліки. Перший з яких це неможливість згенерувати статистичні графіки, що робить візуалізацію даних більш складною, та другий, щоб отримати більш доступний функціонал потрібно сплатити чималі кошти за користування технологією.

Отже, проаналізувавши вищезгадані інформаційні технології, можна зробити висновок, що всі вони мають свої переваги та недоліки та в певній мірі чудово справляються з конкретно поставленими задачами. Проте вимогам та критеріям, які слід імплементувати в нашу інформаційну систему в даних прикладах відсутні.

1.3 Висновки

Отже, у даному розділі було проаналізовано суть технічної проблеми і також здійснено огляд існуючих методів її вирішення. У ході даного дослідження було виявлено, що на сьогоднішній день у вільному доступі немає чітко виражених умов, правил та автоматизованих інформаційних рішень управління та аналізу закордонного ринку.

Провівши аналіз орієнтованих аналогів інформаційних систем, було виявлено, що всі вони мають свої переваги та недоліки та в певній мірі чудово справляються з конкретно поставленими задачами, проте вимогам та критеріям, які слід розробити в інформаційну систему в даних аналогах відсутні.

Тому виникає потреба в створенні автоматизованого ресурсу, який на основі вибірок з бази даних буде проводити аналіз ринку, як імпорту, так і експорту, будувати статистичні графіки та матиме можливість редагувати вихідні дані.

2 ФОРМУВАННЯ ПРОГРАМНИХ ВИМОГ ТА ПОБУДОВА КОНЦЕПЦІЇ ПРОДУКТУ

2.1 Побудова концепції продукту

По мірі збільшення товарообігу за кордон виникає потреба контролю ресурсів які проходять через імпорт та експорт країни.

В першу чергу програмний продукт створюється для аналізу та візуалізації даних імпорту та експорту продукції за кордон. Для реалізації практичної частини технічного завдання перш за все необхідно:

– здійснити статистичний збір даних, та створити базу даних, яка забезпечить веб-ресурс необхідною інформацією;

– онлайн сервіс, який забезпечить безперервну роботу самої програми;

– запровадити мобільність для використання на всіх популярних браузерах;

– оптимізувати роботу програми, щоб при будь-якій швидкості передачі даних, сервіс працював коректно;

– розробити Single Page Application (SPA) за допомогою якого дані будуть швидше та якісніше відображатися на сторінці сайту;

– сервіс повинен мати змогу на перегляд (за можливістю й редагування даних БД);

– підключення карти України за допомогою якої буде змога по регіонам аналізувати дані;

– побудову тематичних карт на основі вибірок з бази даних за параметрами, вказаними користувачем: інтервал років, імпорт чи експорт;

– побудову графіків, які надають змогу відстежити тенденцію імпорту чи експорту по рокам (рис. 2.1);

– побудову порівняльних графіків, за допомогою карти, які надають змогу відстежити тенденцію імпорту чи експорту між кожною із областями України.

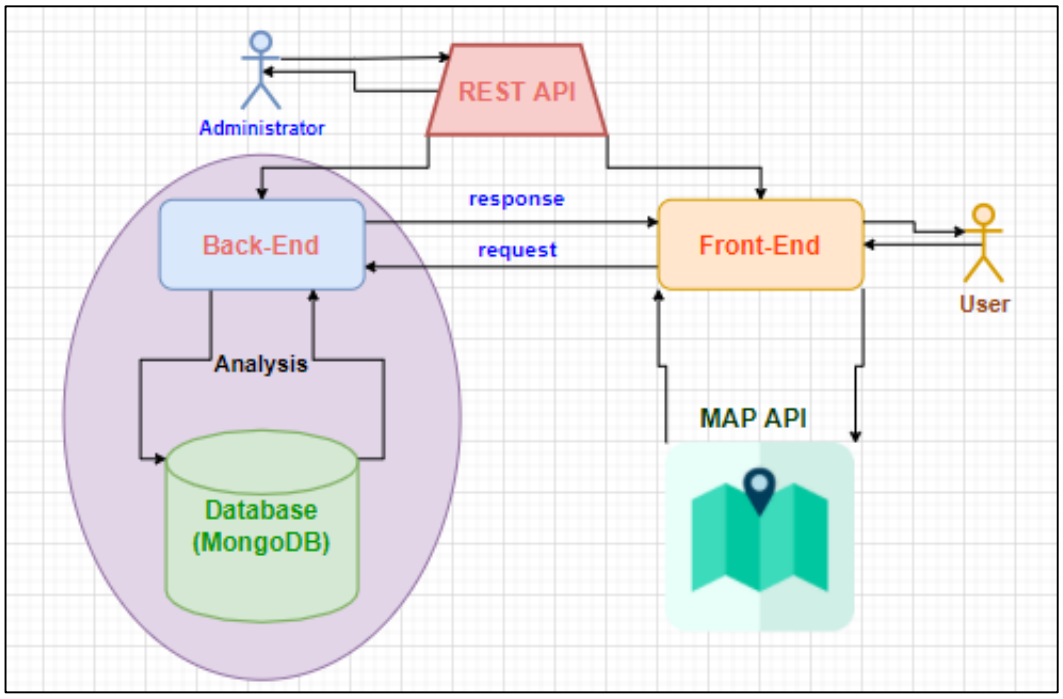


Рисунок 2.1 – Базова схема роботи веб-сервісу

На основі схеми роботи веб-сервісу зображеної на рисунку 2.1 користувач повинен мати змогу аналізувати вихідні дані за допомогою геоінформаційних систем, а також візуалізувати їх за допомогою тематичних карт. Крім того, також основною вимогою до даного продукту є те, що дані на сервері повинні редагуватися зі сторони користувача. Всі запити на сервер, будуть відправлятися за допомогою REST API [12].

RESTful API — це інтерфейс, який використовують дві комп'ютерні системи для безпечного обміну інформацією через Інтернет. Більшість бізнес-додатків мають спілкуватися з іншими внутрішніми та сторонніми додатками для виконання різноманітних завдань. Наприклад, щоб генерувати щомісячні платіжні відомості, ваша внутрішня система облікових записів має обмінюватися даними з банківською системою вашого клієнта, щоб автоматизувати виставлення рахунків і обмінюватися даними з внутрішньою програмою обліку робочого часу. API RESTful підтримують цей обмін інформацією, оскільки вони відповідають безпечним, надійним і ефективним стандартам зв'язку програмного забезпечення [12].

API REST обмінюються даними через запити HTTP для виконання стандартних функцій бази даних, таких як створення, читання, оновлення та видалення записів (також відомих як CRUD) у межах ресурсу. Наприклад, REST API використовуватиме запит GET для отримання запису, запит POST для його створення, запит PUT для оновлення запису та запит DELETE для його видалення. Усі методи HTTP можна використовувати у викликах API. Добре розроблений REST API схожий на веб-сайт, який працює у веб-браузері з вбудованою функцією HTTP.

Стан ресурсу в будь-який конкретний момент або часову позначку називають представленням ресурсу. Цю інформацію можна надати клієнту практично в будь-якому форматі, включаючи JavaScript Object Notation (JSON), HTML, XLT, Python, PHP або звичайний текст. JSON популярний, тому що його читають як люди, так і машини, і він не залежить від мови програмування.

Заголовки та параметри запитів також важливі у викликах REST API, оскільки вони включають важливу інформацію ідентифікатора, таку як метадані, авторизації, уніфіковані ідентифікатори ресурсів (URI), кешування, файли cookie тощо. Заголовки запитів і відповідей разом зі звичайними кодами стану HTTP використовуються в добре розроблених REST API.

Хоча гнучкість є великою перевагою дизайну REST API, та сама гнучкість дозволяє легко розробити API, який зламаний або працює погано. З цієї причини професійні розробники діляться найкращими практиками щодо специфікацій REST API.

Специфікація OpenAPI (OAS) встановлює інтерфейс для опису API у спосіб, який дозволяє будь-якому розробнику чи програмі відкрити його та повністю зрозуміти його параметри та можливості – доступні кінцеві точки, дозволені операції на кожній кінцевій точці, параметри операцій, методи автентифікації та інше інформації. Остання версія, OAS3, містить практичні інструменти, такі як генератор OpenAPI, для створення клієнтів API та серверних заглушок на різних мовах програмування.

Захист REST API також починається з найкращих галузевих практик, таких як використання алгоритмів хешування для захисту паролів і HTTPS для безпечної передачі даних. Платформа авторизації, як-от OAuth 2.0, може допомогти обмежити привілеї програм сторонніх розробників. Використовуючи мітку часу в заголовку HTTP, API також може відхилити будь-який запит, який надходить після певного періоду часу. Перевірка параметрів і веб-токени JSON є іншими способами гарантувати, що лише авторизовані клієнти можуть отримати доступ до API.

Переваги REST API означають, що вони й надалі залишатимуться невід'ємною частиною процесу розробки програмного забезпечення, особливо враховуючи, що попит на кращий досвід клієнтів і більше додатків впливає на бізнес та IT-операції [12].

Коли справа доходить до задоволення цих вимог, рух до більшої автоматизації допоможе. В ідеалі це було б почати з невеликих, помітно успішних проектів, які потім можна масштабувати й оптимізувати для інших процесів і в інших частинах вашої організації. Працюючи з IBM, ви матимете доступ до можливостей автоматизації на основі штучного інтелекту, включно з попередньо створеними робочими процесами, щоб допомогти прискорити інновації, зробивши кожен процес інтелектуальнішим.

Вся передача даних буде представлення у форматі JSON, оскільки це найбільш зручний формат для роботи з базою даних MongoDB.

JSON (JavaScript Object Notation) — це відкритий стандартний формат файлу для обміну даними, який використовує зрозумілий для людини текст для зберігання та передачі даних. Файли JSON зберігаються з розширенням .json. JSON потребує менше форматування та є хорошою альтернативою для XML. JSON походить від JavaScript, але є незалежним від мови форматом даних. Генерація та аналіз JSON підтримується багатьма сучасними мовами програмування. application/json — це тип носія, який використовується для JSON [13].

2.2 Постановка задачі та вибір оптимальних інформаційних технологій

Задля реалізації веб-сервісу аналізу динаміки географічної структури зовнішньої торгівлі товарами України, потрібно здійснити порівняльний аналіз існуючих технологій, для забезпечення дійсно швидко працюючого інструменту.

На сьогоднішній день є дуже велика кількість різноманітних технологій, проте вибір зупинився на такому стеку технологій як MERN (Mongo, Express, React.js, Node.js)

MERN стек – це набір потужних і надійних технологій, які використовуються для розробки масштабованих головних веб-додатків, що містять серверні, зовнішні та компоненти бази даних. Саме JavaScript використовується для швидшої та легшої розробки повноцінних веб-додатків. MERN Stack — це технологія, яка є зручною для користувача повноцінною платформою JavaScript для створення програм і динамічних веб-сайтів.

MERN стек складається з чотирьох основних компонентів або, можна сказати, чотирьох основних технологій:

- M означає MongoDB (база даних), яка в основному використовується для підготовки бази даних документів і є системою баз даних NoSQL (мова неструктурованих запитів);
- E означає Express, в основному використовується для розробки веб-фреймворку Node.js;
- R означає React, в основному використовується для розробки фреймворку JavaScript на стороні клієнта;
- N означає js, який в основному використовується для розробки основного веб-сервера JavaScript.

Кожна з цих чотирьох технологій відіграє важливу роль у забезпеченні наскрізної основи для розробників. Навіть ці чотири технології відіграють важливу роль у процесі розробки веб-додатків [14].

Стек MERN є одним із варіантів MEAN. MEAN також складається з чотирьох компонентів або, скажімо, чотирьох різних технологій, тобто M для MongoDB, E для Express, A означає Angular.js, а N означає Node. MERN переважно використовується для швидшої розробки невеликих програм порівняно з MEAN, а стек MEAN є кращим варіантом для великих програм. Однак на розробку менших програм потрібно більше часу. Крім того, вони обидва мають порівняно різні структури.

Node і Express складають середню програму або рівень. Node.js – дуже потужна та популярна серверна платформа JavaScript, а Express.js – це веб-фреймворк на стороні сервера.

Чому ми повинні вибрати MERN Stack для створення мобільних і веб-додатків?

Рентабельність: усі чотири технології, згадані вище, MERN (MongoDB, Express.js, React.js і Node.js) використовуються в MERN Stack побудовано на JavaScript, що робить його економічно ефективним і з меншими інвестиціями. користувач зможе отримати кращі результати або результати.

Підтримка SEO: підтримка SEO (оптимізації пошукових систем) означає, що Google, Yahoo та інші пошукові системи можуть ефективно та легко здійснювати пошук на кожній сторінці веб-сайту, ефективно інтерпретувати та співвідносити вміст із шуканим текстом і легко індексувати його у своїй базі даних. Як і будь-коли, коли веб-сайти створюються з використанням технологій MERN, вони завжди дружні до SEO.

Краща продуктивність: краща продуктивність стосується швидшої реакції між серверною частиною, зовнішнім інтерфейсом і базою даних, що в кінцевому підсумку покращує швидкість веб-сайту та забезпечує кращу продуктивність, забезпечуючи тим самим зручну взаємодію з користувачем.

Покращує безпеку: головним чином це стосується безпеки програм, створених за допомогою MERN; її безпека веб-додатків відноситься до різних процесів, методів або технологій, які використовуються для захисту веб-серверів і різноманітних веб-додатків, таких як API (інтерфейс користувача додатків) від атак загроз в Інтернеті. Як правило, постачальники безпечного хостингу можуть легко інтегрувати програми, створені за допомогою стеку MERN. Для більшої або кращої безпеки також використовуються засоби безпеки Mongo DB і Node.js [14].

Забезпечте найшвидшу доставку: будь-які веб-додатки та мобільні додатки, створені за допомогою MERN, створюються набагато швидше, що також сприяє швидшій доставці нашим клієнтам.

Забезпечує швидші модифікації: технології стеку MERN підтримують швидкі модифікації за запитом клієнта в мобільних і веб-додатках.

Відкритий код: усі чотири технології, задіяні в MERN, є відкритими. Ця функція дозволяє розробникам отримувати рішення для запитів, які можуть виникати з відкритих порталів під час розробки. Як наслідок, це буде остаточно вигідно розробнику.

Легке перемикання між клієнтом і сервером: MERN дуже простий і швидкий, оскільки він написаний лише однією мовою. Крім того, дуже легко перемикатися між клієнтом і сервером.

MERN має 3-рівневу архітектурну систему, яка в основному складається з 3 рівнів. Ці шари є такими:

– веб як інтерфейсний рівень;
– сервер як середній рівень;
– база даних як базовий рівень.

На рисунку 2.2 зображено архітектуру роботи MERN.

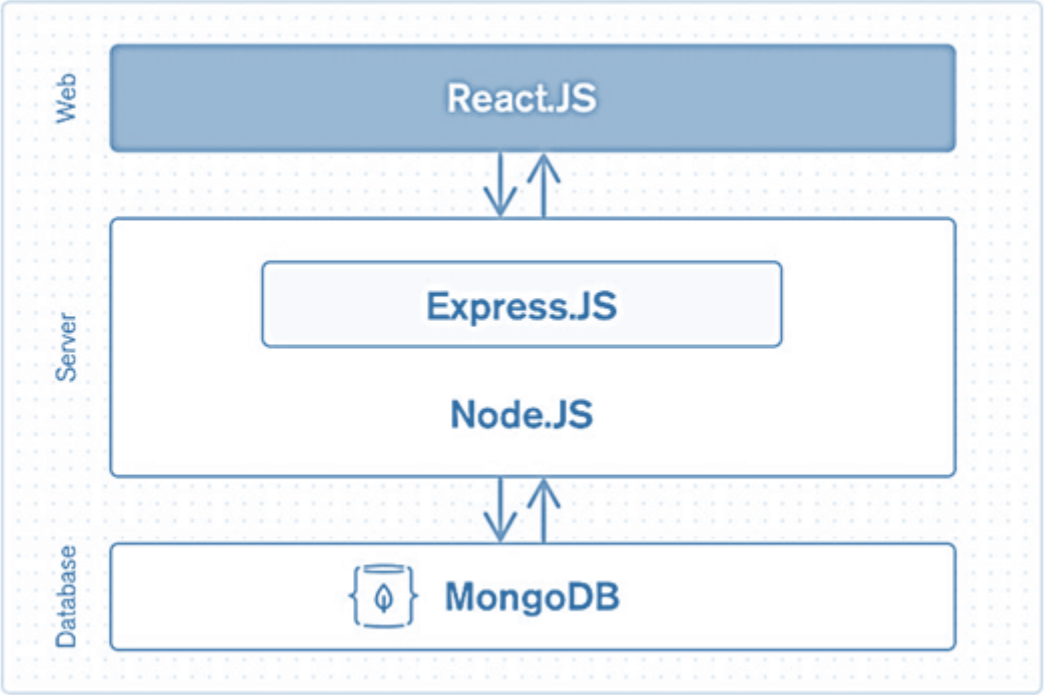


Рисунок 2.2 – Архітектура роботи MERN

Node.js – це кросплатформне середовище виконання JavaScript із відкритим кодом і бібліотека для запуску веб-додатків поза браузером клієнта. Райан Дал розробив його в 2009 році, а його остання версія, версія 19.0.1, була випущена в жовтні 2022 року. Розробники використовують Node.js для створення веб-додатків на стороні сервера, і він ідеально підходить для програм, що інтенсивно обробляють дані, оскільки використовує асинхронну подію керована модель [15].

Існує багато причин, через які розробники віддають перевагу використанню NodeJs для серверної сторони програми, деякі з них обговорюються нижче:

NodeJs побудовано на движку Google Chrome V8, і з цієї причини його час виконання дуже швидкий і він працює дуже швидко.

У Node Package Manager доступно понад 50 000 пакетів, тому розробники можуть будь-коли імпортувати будь-які пакети відповідно до їхньої необхідної функціональності, що економить багато часу.

Оскільки NodeJs не потрібно чекати, поки API поверне дані, це дуже корисно для створення веб-додатків у режимі реального часу та інтенсивних даних. Це повністю асинхронний характер, що означає, що він повністю не блокується.

NodeJs скорочує час завантаження аудіо чи відео, оскільки існує краща синхронізація коду між клієнтом і сервером для однакової кодової бази.

Оскільки NodeJs є відкритим вихідним кодом і це не що інше, як фреймворк JavaScript, тому для розробників, які вже звикли до JavaScript, розпочати розробку своїх проектів за допомогою NodeJs дуже легко [15].

Особливості NodeJs:

– Асинхронний за своєю природою та керований подіями: сервери, створені за допомогою NodeJs, ніколи не чекають на API від API. Не чекаючи даних з API, він безпосередньо переходить до наступного API. Отже, усі API NodeJS за своєю природою абсолютно не блокують. Щоб отримувати та відстежувати всі відповіді на попередні запити API, він дотримується механізму, керованого подіями. Отже, ми можемо сказати, що всі API NodeJs є неблокуючими за своєю природою;

– Однопотокова архітектура: за допомогою циклу подій за однопотоковою архітектурою слідують NodeJs, і ця архітектура робить NodeJs більш масштабованими. На відміну від інших серверів, вони створюють обмежені потоки для обробки запитів. Тоді як для механізму, керованого подіями, сервери NodeJs відповідають неблокуючим або асинхронним способом, і з цієї причини NodeJS стає більш масштабованим. Якщо порівняти NodeJs з іншими традиційними серверами, такими як HTTP-сервери Apache, то можна сказати, що NodeJs обробляє більшу кількість запитів. За однопотоковою програмою слідує NodeJS, і це дозволяє NodeJs обробляти величезну кількість запитів [15];

– Масштабованість: у наш час масштабоване програмне забезпечення потрібне більшості компаній. NodeJs вирішує одну з найбільш нагальних проблем у розробці програмного забезпечення, і це

масштабованість. Одночасні запити можна обробляти дуже ефективно за допомогою NodeJ. Кластерний модуль використовується NodeJs для керування балансуванням навантаження для всіх активних ядер ЦП. Найбільш привабливою особливістю NodeJs є те, що він може розділяти програми горизонтально, і ця процедура розділення в основному досягається завдяки використанню дочірніх процесів. Використовуючи цю функцію, різні версії додатків надаються різним цільовим аудиторіям, а також для налаштування, це дозволяє їм задовольняти вподобання клієнтів;

– Швидкий час виконання для коду: двигун виконання JavaScript V8 використовується NodeJs, і це також використовується Google Chrome. Концентратор надає обгортку для JavaScript, і з цієї причини двигун виконання стає швидшим, і з цієї причини всередині NodeJs процес препозиції запитів також стає швидшим;

– Сумісність на різних платформах: різні типи систем, як-от Windows, UNIX, LINUX, MacOS та інші мобільні пристрої, можуть використовувати NodeJ. Для створення самодостатнього виконання його можна поєднати з будь-яким відповідним пакетом;

– Використовує JavaScript: з точки зору інженера, дуже важливим аспектом NodeJs є те, що цей фреймворк використовує JavaScript. Більшість розробників знайомі з JavaScript, тому для них стає дуже простіше отримати NodeJs;

– Швидке потокове передавання даних: час обробки даних, які були передані в різні потоки, займає багато часу. Тоді як для обробки даних NodeJs потрібен дуже короткий проміжок часу, і він робить це дуже швидко. NodeJs економить багато часу, оскільки файли обробляються та завантажуються одночасно NodeJs. Таким чином, загальна швидкість передачі даних і потокового відео покращується завдяки NodeJs [15];

– Без буферизації: дані ніколи не буферизуються в програмі NodeJs.

Тепер, коли ми визначили, що таке Node, давайте заглибимося в його архітектуру. Node.js працює в одному потоці, що дозволяє йому обробляти тисячі одночасних циклів подій. Рисунок 2.3 ілюструє архітектуру Node.js.

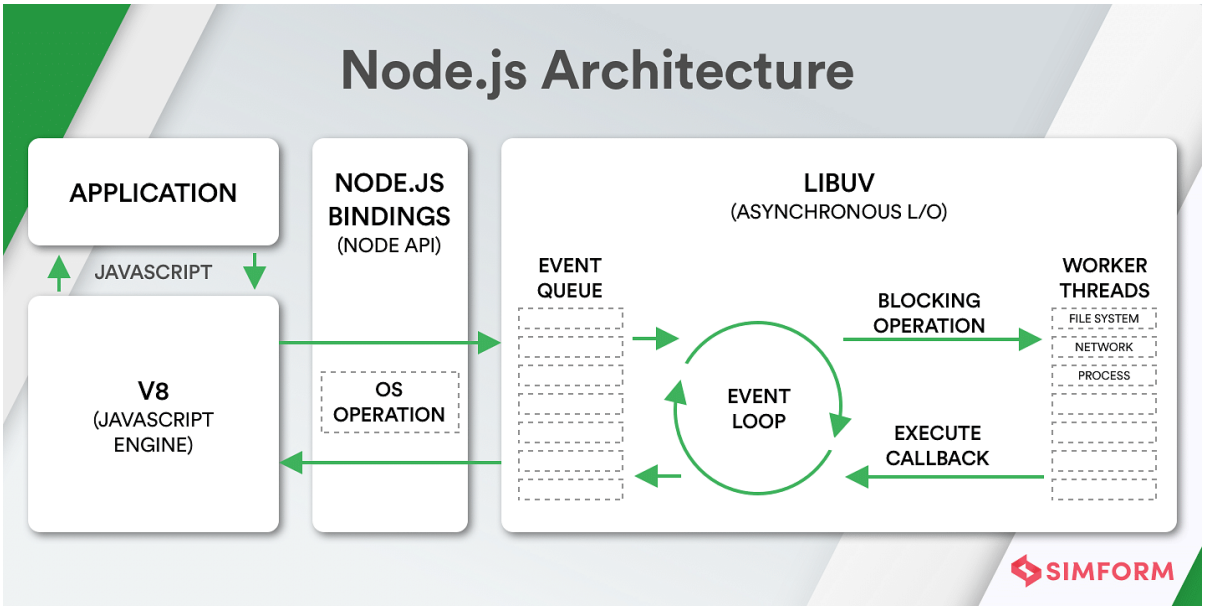


Рисунок 2.3 – Архітектура NodeJs

NodeJs багатокомпонентна мова, на рисунку 2.4 зображено, інструменти що включає в себе NodeJs.

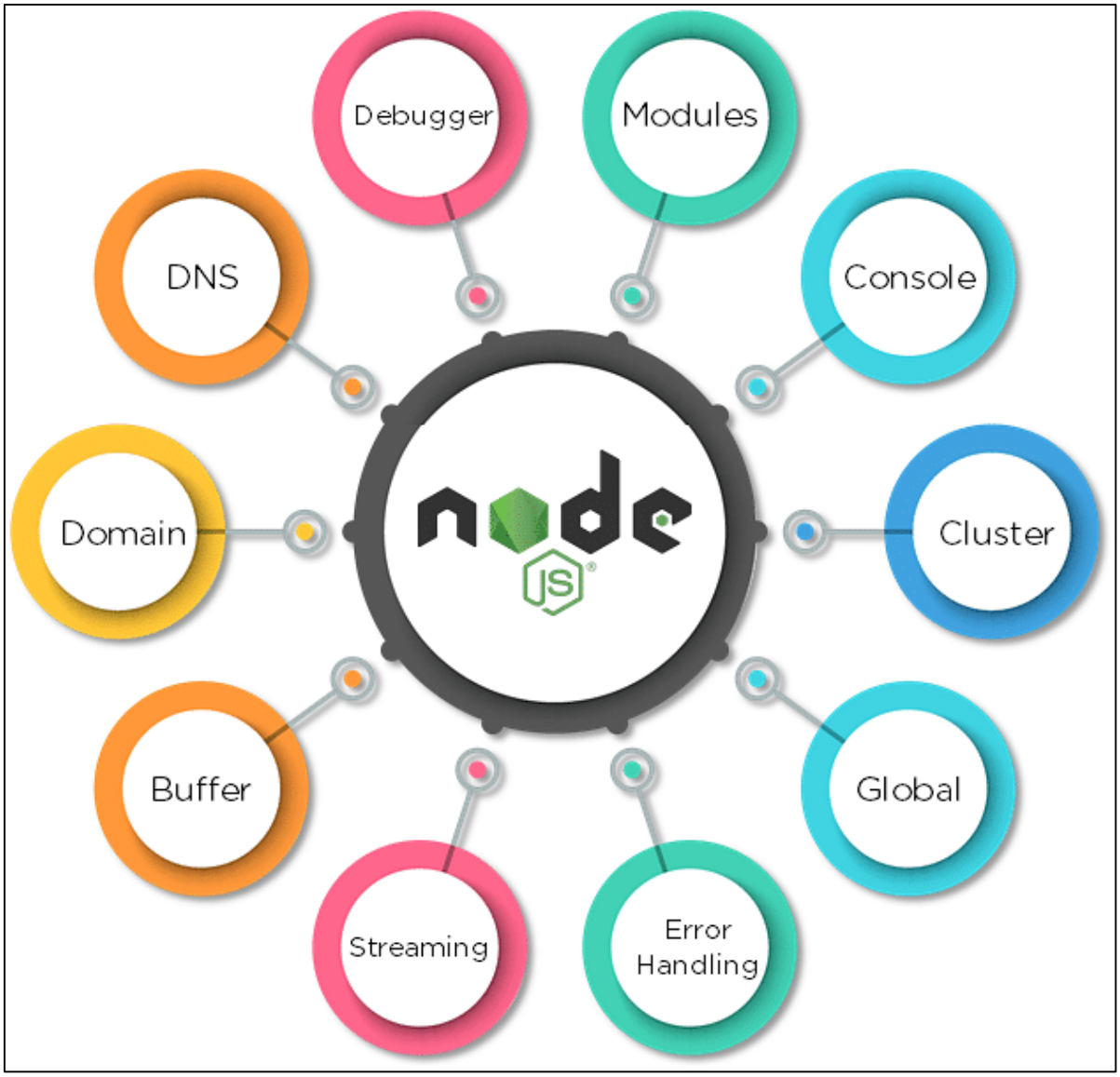


Рисунок 2.4 – Частини Node.js

Модулі схожі на бібліотеки JavaScript, які можна використовувати в програмі Node.js для включення набору функцій. Щоб включити модуль у програму Node.js, використовуйте функцію require() із назвою модуля в дужках. На рисунку 2.5 зображено додавання модуля до програми.

```
// CREATING A WEB SERVER

// Include modules
var http  = require('http');
var server =
http.createServer(function(req, res){
  //write your code here
});
server.listen(2000);
```

Рисунок 2.5 – Додавання модуля у Node.js

Node.js має багато модулів, які забезпечують базову функціональність, необхідну для веб-програми [15]. Деякі з них згадані на рисунку 2.6.

Core Modules	Description
http	Includes classes, methods and events to create Node.js http server
util	Includes utility functions useful for developers
fs	Includes events, classes, and methods to deal with file I/O operations
url	Includes methods for URL parsing
querystring	Includes methods to work with query string
stream	Includes methods to handle streaming data
zlib	Includes methods to compress or decompress files

Рисунок 2.6 – Загальні модулі модулів Node.js

Node.js побудовано на концепції однопотокового програмування. Кластер — це модуль, який забезпечує багатопотоковість шляхом створення дочірніх процесів, які спільно використовують той самий порт сервера та виконуються одночасно.

Глобальні об’єкти в Node.js доступні в усіх модулях. Ці об’єкти – це функції, модулі, рядки тощо. Деякі глобальні об’єкти Node.js зображенні на рисунку 2.7.

Global Objects	Description
__dirname	Specifies the name of the directory that contains the code of application
__filename	Specifies the filename of the code
exports	A reference to the module.exports, shorter to type
module	A reference to the current module
require	Used to import modules, local files, and also JSON

Рисунок 2.7 – Глобальні об’єкти Node.js

Потоки – це об’єкти, які дозволяють безперервно читати або записувати дані. Існує чотири типи потоків:

– Читабельність: це типи потоків, дані з яких можна читати;

– Можливість запису: це типи потоків, у які можна записувати дані;

– Дуплекс: ці потоки доступні як для читання, так і для запису;

– Перетворення: потоки, які можуть маніпулювати даними під час їх читання або запису.

Node.js містить утиліту налагодження, до якої можна отримати доступ із вбудованого клієнта налагодження. Налагоджувач Node.js не має додаткових функцій, але підтримує просту перевірку коду. Налагоджувач можна використовувати в терміналі, використовуючи ключове слово 'inspect' перед назвою файлу JavaScript

Netflix, провідна мережа онлайн-розваг у світі з понад 167 мільйонами користувачів, є однією з багатьох провідних компаній, які довіряють Node.js для своїх серверів. Причини, чому компанія вирішила використовувати Node.js, включають:

– Масштабованість програми;

– Додаток із інтенсивним використанням даних.

За даними Forbes , Walmart є найбільшою компанією у світі за обсягом доходу з 559 мільярдами доларів у 2020 році . Walmart вибрав Node.js через такі атрибути:

– Асинхронний ввід-вивід;

– Ефективна обробка одночасних запитів.

Uber – це американська багатонаціональна компанія, яка надає послуги, які включають одноранговий обмін поїздками, замовлення послуг поїздок і доставку їжі. Причини, чому компанія вирішила використовувати Node.js, включають:

– Асинхронний ввід-вивід;

– Швидкі ітерації;

– Активна спільнота з відкритим кодом.

NASA, незалежне агентство федерального уряду Сполучених Штатів, відповідає за цивільну космічну програму, а також за аерокосмічні та аеронавтичні дослідження. NASA вирішило використовувати Node.js з таких причин:

– Скорочений час доступу;

– Здатність виконувати завдання, що містять інтенсивні дані;

– Можливість підтримувати сервер активним 24/7.

PayPal – це американська компанія, яка керує глобальною системою онлайн-платежів, яка підтримує онлайн-грошові перекази, що є електронною альтернативою традиційним паперовим методам, таким як чеки та грошові перекази. PayPal вирішив використовувати Node.js з таких причин:

– Надзвичайно швидкий час виготовлення;

– Менше рядків коду;

– Здатність працювати з великими обсягами даних.

Express – це просто веб-фреймворк для Node.js. Написати повноцінний веб-сервер вручну безпосередньо на Node.js не так просто і не потрібно. Express – це той фреймворк, який спрощує завдання написання коду сервера [16].

Express – це платформа JavaScript на стороні сервера, яка працює на базі Node.js. Це один з найкращих фреймворків JavaScript для бекенд-розробки. Він надає розробнику платформу для створення та підтримки надійних

серверів. Express використовується для легкого та швидкого створення та розробки веб та мобільних додатків [16].

Express використовується для забезпечення серверної логіки для мобільних і веб-додатків, і тому він використовується всюди. Це дозволяє розробникам створювати надійні API (інтерфейс програмування додатків) і веб-сервери набагато легше та простіше.

Express робить надійні веб-сервери легшими для організації функцій вашої програми за допомогою маршрутизації та проміжного програмного забезпечення. Він також додає корисні функції до об’єктів Node.js HTTP (HyperText Transfer Protocol).

Це важливий компонент MERN і MEAN і використовується для створення швидких, придатних для обслуговування та надійних виробничих веб-додатків [14].

Деякі важливі функції Express:

– Express робить розробку веб-додатків і мобільних додатків Node.js набагато простішою та швидшою;

– Express має дуже просте налаштування середовища. Можна легко встановити Express у своїй системі та налаштувати його, не докладаючи особливих зусиль;

– Express дуже легко підключити до таких баз даних, як MongoDB;

– На основі методів HTTP та URL-адрес Express дозволяє визначати маршрути вашої програми;

– Основною метою маршрутизації є опис коду, який потрібно запустити у відповідь на будь-який запит, отриманий сервером. Зазвичай маршрутизація виконується на основі послідовності шаблонів URL-адрес і методу HTTP, пов’язаного із запитом;

– Якщо потрібно виконувати додаткові завдання та функції для будь-якого запиту та відповіді, ви можете легко використовувати різні модулі проміжного програмного забезпечення, наявні в Express.

Якщо виникає будь-яка помилка, і ви потрібно її обробити, можна легко впоратися з нею за допомогою проміжного програмного забезпечення обробки помилок.

Проміжне програмне забезпечення використовується десь протягом життєвого циклу запиту чи відповіді у формі коду. В основному він використовується для додавання функцій або покращення поведінки веб-сервера.

Express також полегшує створення REST API (Інтерфейс програмування програми передачі представлення стану)

REST API також відомий як RESTful API. Він в основному відповідає обмеженням архітектурного стилю REST, а також дозволяє взаємодіяти з веб-службами RESTful. Основна перевага REST API полягає в тому, що він забезпечує велику гнучкість; він використовує запити HTTP для доступу та використання даних [12].

Потік даних у структуру веб-сайту можна легко полегшити за допомогою двох механізмів шаблонів, EJS і Jade, наданих Express.

Express має гігантський набір сторонніх доповнень, які розробники можуть використовувати для забезпечення кращої функціональності, допомагає підвищити рівень безпеки та покращити швидкість.

Він дуже ефективний і масштабований; можна легко отримати доступ до нього з будь-якого місця та використовувати його одночасно в різних системах, і дуже швидко.

Він також має найбільшу спільноту для Node.js.

Завдяки вбудованому маршрутизатору він сприяє повторному використанню коду [16].

Для більшості веб-сервісів потрібні певні зберігання: часто у формі системи управління базами даних. У той час як традиційно це могло бути надано за допомогою реляційної системи управління базами даних на базі SQL (така як MySQL або SQLServer) існує тенденція до зростання типів бази даних

NoSQL. NoSQL можна використовувати для забезпечення більш гнучких “Документально-орієнтована база даних” з динамічною схемою.

Mongo DB – це найпопулярніша база даних NoSQL (NoSQL або мова неструктурованих запитів), документоорієнтована база даних з відкритим кодом.

Термін «NoSQL» зазвичай означає нереляційну базу даних, яка не вимагає фіксованої схеми або відповідних реляційних таблиць для зберігання в ній необхідних даних. MongoDB зберігає дані в іншому форматі, відмінному від реляційних таблиць, які складаються з рядків і стовпців.

Це означає, що MongoDB не базується на табличної структурі реляційної бази даних. З іншого боку, він забезпечує зовсім інший механізм для пошуку та зберігання даних.

Формат зберігання, у якому зберігаються дані, відомий як BSON, що означає Binary JavaScript Object Notation; його двійкова структура кодує довжину та тип інформації, що дозволяє її аналізувати набагато швидше.

MongoDB використовує BSON для зберігання документів у колекціях.

Це забезпечує гнучку структуру документа з високою масштабованістю.

У MongoDB складні операції з’єднання недоступні; отже, він не може підтримувати складні транзакції.

MongoDB використовує JavaScript для кодування як мову, що є однією з великих переваг [13].

Він без схемний, як і будь-які дані, що зберігаються в окремому документі.

У MongoDB немає поняття взаємозв’язків чи формування таблиць, оскільки це відбувається в RDBMS (система керування реляційними базами даних), у якій таблиці мають певний зв’язок між собою.

Він також підтримує гнучку модель документа, яку дуже швидко створює будь-який розробник.

MongoDB є одним із важливих типів баз даних NoSQL. Він більш масштабований і забезпечує чудову продуктивність, якщо ми помічаємо, що

він досягає свого ліміту масштабування кожного разу, коли база даних працює на одному сервері.

MongoDB – це база даних NoSQL, яка масштабується шляхом додавання все нових і нових серверів і підвищує продуктивність завдяки своїй гнучкій моделі документів [13].

Останньою частиною MERN стеку є React.js.

React – одна з найпопулярніших інтерфейсних бібліотек JavaScript з відкритим кодом, які використовуються для створення веб-додатків.

Перш ніж використовувати React, потрібно виконати деякі передумови, а саме завантажити пакети Node у свою систему з їхніми останніми версіями. Крім того, ви повинні мати розуміння HTML, CSS і JavaScript.

Він використовується для створення інтерфейсів користувача, особливо для односторінкових веб-додатків. Це не фреймворк JavaScript. Це просто бібліотека JavaScript, розроблена Facebook для вирішення проблем, які ми не могли вирішити раніше за допомогою інших бібліотек під час створення веб- і мобільних програм.

React також використовується для створення контролю над шаром перегляду для мобільних і веб-додатків.

Це дозволяє нам створювати багаторазові компоненти інтерфейсу користувача (інтерфейс користувача).

Вперше його створив інженер-програміст Джордан Волке, який працює у Facebook.

React вперше був розгорнутий у стрічці новин Facebook.

Це дозволяє розробникам створювати великі веб-додатки, які можуть легко змінювати дані сторінки навіть без перезавантаження сторінки.

Основна мета реагування полягає в тому, щоб він працював лише на користувацьких інтерфейсах у додатку, мобільному чи веб-сайті.

React також використовується з комбінацією інших бібліотек або фреймворків JavaScript.

Існує багато платформ з відкритим вихідним кодом, які також використовуються для полегшення веб-інтерфейсу та мобільних додатків, наприклад Angular js у MVC, але React замінює Angular зі стеку MEAN. Зараз більшість розробників використовують стек MERN, у якому використовується React. Головна причина полягає в тому, що він дуже швидкий і має більше переваг перед іншими інтерфейсними фреймворками.

Легко освоїтися – одна з великих переваг використання React, оскільки новачкові дуже легше освоїти його та створювати веб- та мобільні додатки за допомогою цього зовнішнього фреймворку. Будь-хто, хто має частину попередніх базових знань у програмуванні, може легко зрозуміти React порівняно з Angular. Angular називають «доменоспеціальною мовою», тому мається на увазі, що її досить важко зрозуміти. Щоб навчитися React, вам потрібні базові знання CSS і HTML [17].

React є одним із найпростіших інтерфейсних фреймворків JavaScript із відкритим кодом для створення веб та мобільних додатків. Він використовує компонентний підхід, використовує зрозумілий і простий JavaScript і чітко визначений життєвий цикл, що робить реагування набагато простішим і легшим. Щоб можна було легко вивчити його та створювати професійні мобільні та веб-додатки. Він використовує простий синтаксис під назвою JSX, який дозволяє учням або розробникам поєднувати HTML з JavaScript, щоб полегшити їх застосування та використання для створення ефективних веб- і мобільних програм. Однак використовувати JSX не обов’язково, ви можете використовувати звичайний JavaScript, але порівняно з JSX, JSX є набагато кращим варіантом через його простоту та легший синтаксис [17].

React використовує архітектуру програми, відому як Flux, щоб контролювати потік даних до компонентів через одну точку керування, яка називається диспетчером. Він використовує одностороннє зв’язування даних , що полегшує налагодження самодостатніх компонентів великих додатків React.

React використовується для створення мобільних додатків (React Native) і веб-додатків. React дозволяє повторно використовувати код і може легко підтримувати його, що має багато переваг і значно економить час. Таким чином, одночасно ми можемо створювати IOS, веб-додатки та Android.

React має дуже високу продуктивність завдяки незмінності даних. Як випливає з назви, ми можемо передбачити, що незмінні структури даних ніколи не змінюватимуться, і це дозволяє вам порівнювати прямі посилання на об’єкти замість порівнянь глибокого дерева. Вищезазначена причина зрештою впливає на ефективність реагування та робить його швидшим.

React дуже легко перевірити; Які б програми ми не створювали з React, будь то мобільні чи веб-додатки, нам набагато простіше перевірити це на React. У React є деякі функції стану, де різні представлення React розглядаються як ці функції станів, і ми можемо легко маніпулювати станом, який ми передаємо в представлення React. Крім того, ми можемо переглянути вихідні дані та запущені дії, функції, події тощо.

React використовує свій специфічний синтаксис для побудови веб-сторінки JSX [17].

JSX – це розширення синтаксису для JavaScript. Він використовується з React для опису того, як повинен виглядати користувальницький інтерфейс. Використовуючи JSX, ми можемо писати HTML-структури в одному файлі, що містить код JavaScript (рис. 2.8). Це полегшує розуміння та налагодження коду, оскільки дозволяє уникнути використання складних структур DOM JavaScript [18].

```
return (
    <>
    <div className="dropdown statics">
        <button className="btn btn btn-secondary dropdown-toggle own-btn item-list"
            type="button"
            id="dropdownMenuButton"
            data-toggle="dropdown"
            aria-haspopup="true"
            aria-expanded="false"
            onClick={this.getNameRegion}
        >
            Regions: {this.state.region}
        </button>
```

Рисунок 2.8 – Реалізація синтаксису JSX в React

Щоб отримати максимальну користь від React при побудові клієнтської частини веб-ресурсу, необхідно використати на додаток такі інструменти як Bootstrap та SASS.

Bootstrap – це популярний CSS-проект із відкритим кодом (в першу чергу створений Twitter), який описується як «… найбільше популярний HTML, CSS та JS фреймворк для розробки адаптивні мобільні перші проекти в Інтернеті». Bootstrap забезпечує елегантно оформлені елементи CSS, що робить це легким розробити веб-вміст у чистому, сучасному вигляді [19].

Sass (англ. Syntactically Awesome Stylesheets) – скриптова метамова, яка інтерпретується в каскадні таблиці стилів (CSS). Спроектована Гемптоном Кетліном та розроблена Наталі Вейзенбаум. Sass призначений для підвищення рівня абстракції коду та спрощення файлів CSS.

Мова Sass має два синтаксиси:

– sass (оригінальний) – відрізняється відсутністю фігурних дужок, в ньому вкладені елементи реалізовані за допомогою відступів, а правила відокремлюються переведенням рядка;

– scss (новий) – використовує фігурні дужки (подібно до CSS).

Файли sass-синтаксису мають розширення .sass, scss-синтаксису — .scss.

Sass розширює CSS, надаючи кілька механізмів, доступних в більш традиційних мовах програмування, зокрема об’єктноорієнтованих мовах, але

недоступних для CSS. Інтерпретатор Sass транслює SassScript у блоки правил CSS [20].

Разом поєднання цих інструментів із подальшою логікою веб-сервісу, реалізує потужний та швидкий веб-додаток, який забезпечить користувача всіма необхідними ресурсами.

2.3 Висновки

Отже, в даному розділі було визначено всі програмні вимоги, які програма повинна задовільнити та побудовано базову концепцію майбутнього продукту.

Також було проаналізовано передові технологій, які б задовольнили вимоги у вирішенні поставленої задачі. Найбільш оптимальним рішенням обрано стек технологій MERN (Mongo Express React Node). Дані технології чітко поєднуються між собою, створюючи потужний та швидкий веб-додаток, який забезпечує користувача всіма необхідними ресурсами.

3 ЗБИРАННЯ ТА СИСТЕМАТИЗАЦІЯ ДАНИХ ЗОВНІШНЬОЇ ТОРГІВЛІ ТОВАРАМИ УКРАЇНИ

3.1 Збирання та адаптація даних

Перш за все потрібно визначитися з ресурсом, з якого можна буде черпати дані динаміки географічної структури зовнішньої торгівлі товарами України. Провівши невелике дослідження, було виявлено що всі необхідні дані можна почерпнути із сайтів обласних управлінь статистики України, де можна знайти дані імпорту та експорту кожної з області з 1996 по 2019 роки.

В даній вибірці можна бачити конкретні напрямки куди здійснювалася торгівля, а саме це: Африка, США, ЄС, Країни Європи, Азія, Океанія та інші країни світу. Фрагмент даної вибірки зображено на рисунку 3.1

Динаміка географічної структури зовнішньої торгівлі послугами (1996–2019рр.)

(тис.дол. США)

Роки	Усього	Країни СНД[1]	Інші країни світу[1]	Європа	Країни ЄС (28)	Азія	Африка	Америка	Австралія і Океанія	Невизначені країни
1996	3849,2	1582,5	2266,7	390,0	390,0	1751,1	117,2	8,4	–	–
1997	2915,3	1006,9	1908,5	497,9	497,9	1293,6	113,9	3,1	–	–
1998	1784,8	308,0	1476,8	491,3	486,9	875,7	89,0	20,6	0,2	–
1999	1928,7	124,5	1804,2	497,7	490,1	1152,0	115,4	39,1	0,0	–
2000*	2544,8	133,7	1730,0	472,0	466,6	1138,0	98,0	21,8	0,2	–
2001*	2471,7	73,7	1724,8	427,6	426,7	1139,2	137,5	19,8	0,7	–
2002*	4260,3	977,9	1597,2	500,5	475,7	945,8	137,6	13,1	0,2	–
2003*	4178,5	789,1	1706,5	600,7	588,3	658,5	170,1	57,2	1,2	218,8
2004*	10464,8	1253,1	9211,6	7656,5	7651,7	853,0	314,3	387,5	0,3	–
2005*	13758,0	402,3	13355,7	10953,6	10946,0	1146,5	509,9	745,4	0,3	–
2006*	19526,5	841,9	18684,6	11646,8	13044,5	5983,0	921,0	133,8	–	–
2007*	26643,8	1482,6	24771,8	13161,9	21872,3	10851,7	606,3	151,9	–	–
2008*	41306,8	4074,6	37232,2	18505,0	33681,1	17732,8	782,9	209,1	2,4	–
2009*	42801,5	3195,1	37606,4	13650,7	29849,8	22201,3	967,7	785,8	0,9	–
2010*	34093,5	3053,3	31040,2	13640,4	24596,8	14350,5	1035,7	2012,5	1,1	–
2011*	39952,6	4175,6	35777,0	16975,1	27049,8	13617,9	1182,7	3996,3	5,0	–
2012*	46112,0	4218,5	41893,5	19230,0	24975,4	17548,4	1162,3	3935,2	17,6	–
2013	59276,9	6947,7	52329,2	27869,4	18590,9	15856,9	1542,8	7004,2	56,0	–
2014	66404,3	6224,1	60180,2	29700,4	13208,7	14505,3	2928,2	13031,4	14,9	–
2015	62927,9	2800,8	60127,1	24576,0	11228,2	18992,7	3885,2	12667,7	5,5	–
2016	66146,0	3104,3	63041,8	21942,5	13356,8	24513,0	4153,5	12406,0	26,8	–
2017	78838,7	3401,6	75437,1	25387,3	15737,4	27585,4	4661,0	17792,5	10,9	–
2018	102074,4	3897,2	98177,2	28534,4	26640,0	36176,5	4384,3	29072,0	10,1	–
2019	135193,9	–	–	42323,2	40722,2	49495,7	3965,8	36872,6	10,6	–

Рисунок 3.1 – Фрагмент вибірки експорту динаміки зовнішньої торгівлі товарами у вінницькій області

Задля більш зручного збору та трансформації даних було використано даних табличний процесор Microsoft Excel.

Microsoft Excel (повна назва Microsoft Office Excel) − табличний процесор, програма для роботи з електронними таблицями, створена корпорацією Microsoft для Microsoft Windows, Windows NT і Mac OS.

Програма входить до складу офісного пакета Microsoft Office.

Типові області застосування MS Excel:

– завдяки тому, що лист MS Excel являє собою готову таблицю, MS Excel часто використовують для створення документів без усіляких розрахунків, що просто мають табличне представлення (наприклад, прайс-листи в магазинах, розклади);

– у MS Excel легко можна створювати різні види графіків і діаграм, які беруть дані для побудови з комірок таблиць (графік зниження ваги тіла за вказаний період від початку занять спортом);

– його можуть використовувати звичайні користувачі для елементарних розрахунків (скільки витратив за цей місяць, що/кому/коли дав/взяв);

– MS Excel містить багато математичних і статистичних функцій, завдяки чому його можуть використовувати школярі і студенти для розрахунків курсових, лабораторних робіт;

– MS Excel інтенсивно використовується в бухгалтерії — у багатьох фірмах це основний інструмент для оформлення документів, розрахунків і створення діаграм. Природно, він має в собі відповідні функції;

– MS Excel може навіть працювати як база даних. Хоча, звичайно, до повноцінної бази даних йому далеко [21].

Фрагмент даних у форматі MS Excel зображено на рисунку 3.2.

A25 | 6/18/2019

	A	B	C	D	E	F	G	H	I	J
1	date	all	cis	other_countries	europe	eu_countries	asia	africa	usa	oceania
2	6/18/1996	**3424.90**	1342.00	2082.90	1433.50	1369.10	295.90	16.90	335.60	1.00
3	6/18/1997	**4733.10**	2117.00	2616.10	1873.10	1783.70	352.70	7.00	382.10	1.20
4	6/18/1998	**4080.30**	1486.10	2594.20	1779.70	1692.90	339.10	12.30	461.70	1.40
5	6/18/1999	**3560.10**	1853.00	1707.10	1178.10	1124.30	315.10	22.60	190.90	0.50
6	6/18/2000	**6795.70**	4968.10	1827.60	1350.80	1274.90	287.20	27.90	161.00	0.70
7	6/18/2001	**6064.50**	3645.90	2418.60	1832.60	1740.50	346.10	28.20	210.30	1.40
8	6/18/2002	**3915.20**	1189.70	2725.50	2109.20	2004.90	417.70	19.50	176.90	2.20
9	6/18/2003	**5183.50**	1367.90	3815.60	2809.30	2690.50	655.10	23.60	325.50	2.10
10	6/18/2004	**6539.50**	1575.80	4963.70	3628.00	3442.30	887.00	28.90	415.20	4.60
11	6/18/2005	**13268.00**	6242.70	7025.30	5051.70	4809.10	1455.70	37.10	475.60	5.20
12	6/18/2006	**16201.80**	6664.40	9537.40	6720.80	6435.20	2124.00	72.50	607.30	12.80
13	6/18/2007	**24270.40**	11151.80	13118.60	9149.10	8787.70	3058.20	55.20	845.80	10.30
14	6/18/2008	**32820.30**	12731.20	20089.10	12403.10	11379.90	5811.60	67.10	1787.70	19.60
15	6/18/2009	**24116.90**	14476.10	9640.80	6393.40	5977.60	2378.60	71.80	787.10	9.80
16	6/18/2010	**24150.70**	11362.50	12788.20	8041.30	14919.10	3752.10	87.80	894.20	11.90
17	6/18/2011	**22775.40**	5512.90	17262.50	10737.60	10008.90	5301.90	109.80	1087.80	25.00
18	6/18/2012	**26116.90**	6734.20	19382.70	11171.30	10394.60	6629.10	132.00	1410.90	33.80
19	6/18/2013	**26679.60**	6563.60	20116.00	11812.30	10929.10	6609.40	136.20	1518.30	34.80
20	6/18/2014	**20179.90**	5356.80	14823.10	8513.70	8001.90	5088.10	109.70	1075.10	36.50
21	6/18/2015	**14495.40**	4222.00	10273.40	5857.70	5621.50	3578.00	108.20	721.90	7.60
22	6/18/2016	**16137.00**	3752.10	12384.80	6714.10	6425.90	4586.00	129.90	944.80	9.90
23	6/18/2017	**19848.60**	4977.80	14870.80	7913.60	7600.30	5573.20	150.00	1220.80	13.20

import | export

Рисунок 3.2 – Фрагмент вибірки імпорту даних у форматі Excel зовнішньої торгівлі товарами у вінницькій області

Завдяки більш чіткій структурі даних, яку було побудовано в Microsoft Excel подальша робота із вихідними даними буде значно легше, а саме це конвертація типів таких як дата, число та рядок, та експорт до майбутньої бази даних.

3.2 Створення та заповнення бази даних

Для зберігання, управління та редагування вихідних даних веб-системи, було зроблено вибір на користь не реляційній базі даних (NoSQL), а саме MongoDB.

MongoDB − це база даних NoSQL, де кожен запис є документом, що складається з пар ключ-значення, подібних до об'єктів JSON (JavaScript Object Notation). MongoDB є гнучкою і дозволяє своїм користувачам створювати схеми, бази даних, таблиці тощо. Документи, які можна ідентифікувати за допомогою первинного ключа, складають основну одиницю MongoDB. Після встановлення MongoDB користувачі також можуть використовувати оболонку Mongo. Оболонка Mongo надає інтерфейс JavaScript, за допомогою якого користувачі можуть взаємодіяти та виконувати операції (наприклад: запити, оновлення записів, видалення записів) [22].

Навіщо використовувати MongoDB?

Швидкість − будучи орієнтованою на документи базою даних, легко індексувати документи. Тому швидша реакція.

Масштабованість − здатність обробляти великі дані можна, розділивши їх на кілька машин.

Використання JavaScript − MongoDB використовує JavaScript, що є найбільшою перевагою.

Однією з найбільших переваг MongoDB є те, що не потрібно будувати схеми для окремого документа, можна зберігати будь-який тип даних в окремому документі. Ще одна особливість MongoDB – це зберігання у форматі JSON. Об'єкти, члени об'єктів, масиви, значення та рядки все це підтримується JSON. Синтаксис JSON дуже простий у використанні.JSON має широкий спектр сумісності з браузерами [13].

Просте налаштування навколишнього середовища − дуже просто налаштувати MongoDB.

Гнучка модель документа − MongoDB підтримує модель документа (таблиці, схеми, колонки), яка швидша та простіша [13].

Створення бази даних, насправді дуже просте достатньо просто мати на локальній машині встановлене середовище MongoDB або скористатися віддалиним сервісом та в консолі виконати команду use database_name;, після чого MongoDB згенерує середовище бази даних.

Створення колекції також, не суттєво відрізняється від створення таблиці. Команда db.createCollection(“collection_name”); згенерує колекцію бази даних або якщо колекція вже існує просто замінить її.

Вставка записів до колекції та витяг даних здійснюється аналогічним чином. Фрагменти команд до бази даних зображені на рисунку 3.3, 3.4.

```
db.collection_name.insert
(
    {
        "id": 1,
        "Name": "Klaus",
                "Department": "Technical",
                "Organization": "Geeks For Geeks"
    }
);
```

Рисунок 3.3 – Запис даних до колекції MongoDB

```
db.collection_name.find({Name : "Klaus"}).forEach(printjson);
```

```
admin      0.000GB
config     0.000GB
expressor  0.000GB
geeks      0.000GB
local      0.000GB
> use geeks
switched to db geeks
> db.createCollection("geeksforgeeks");
{ "ok" : 1 }
> db.geeksforgeeks.insert({"id": 1, "Name": "Klaus" , "Department": "Technical" , "Organization": "Geek
sForGeeks"});
WriteResult({ "nInserted" : 1 })
> db.geeksforgeeks.find({Name : "Klaus"}).forEach(printjson);
{
        "_id" : ObjectId("5d9351e7ec1d1d023bbadd7a"),
        "id" : 1,
        "Name" : "Klaus",
        "Department" : "Technical",
        "Organization" : "GeeksForGeeks"
}
>
```

Рисунок 3.4 – Фрагмент запиту та його результат в MongoDB

Оскільки база даних MongoDB працює тільки із даними в форматі JSON, тому виникає потреба в конвертації наших даних, та завдяки, тому що вся інформація чітко структурована в Excel таблицях, то цю не важку операцію можна здійснити на будь-якому онлайн сервісі. Задля вирішення поставленої задачі було обрано сервіс BeautifyTools. Результат конвертації даних зображений на рисунку 3.5.

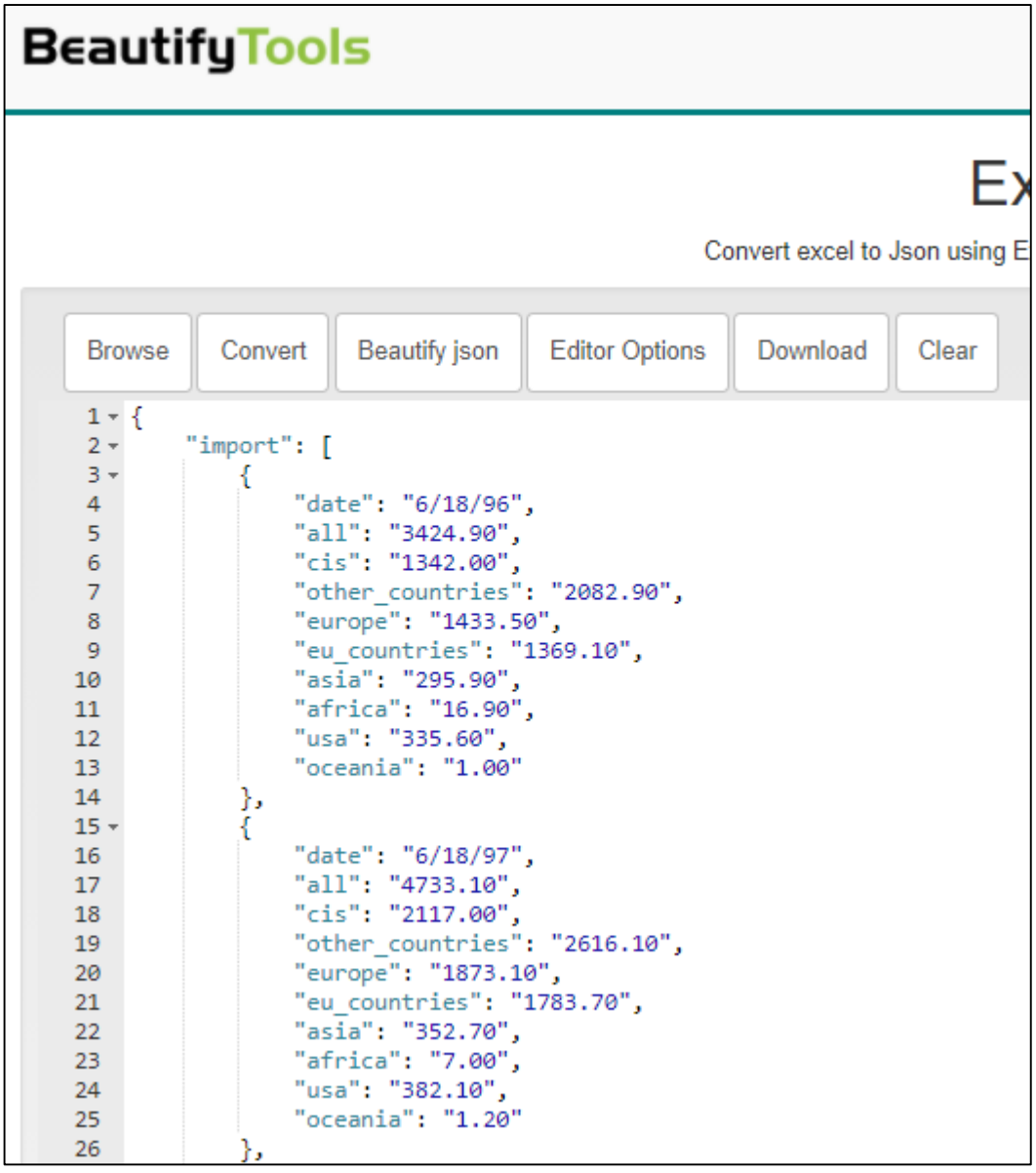


Рисунок 3.5 – Конвертація даних з формату xlsx до json

Після вдалої операції з конвертацією наповнення бази даних буде здійснюватися буквально в декілька кроків.

Для завантаження всієї одразу інформації до бази даних був використаний інструмент MongoDB Compass.

MongoDB Compass – це офіційний графічний інтерфейс для MongoDB, який підтримує сам MongoDB. MongoDB Compass допомагає користувачам приймати розумні рішення щодо структури даних, запитів, індексації та багатьох інших дій, які ви можете виконувати з базою даних [13].

Завантаження інформації до нашої бази даних зображено на рисунку 3.6.

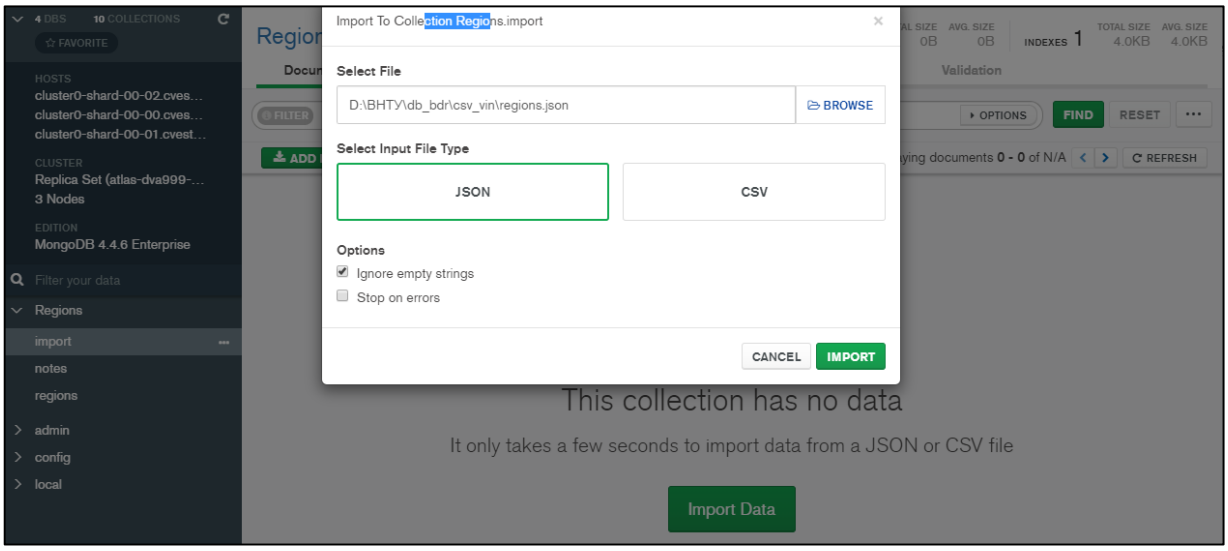


Рисунок 3.6 – Наповнення інформацією базу даних MongoDB

Після здійснення всіх операції дані вдало завантаженні в базу. Фрагмент бази даних зображено на рисунку 3.7.

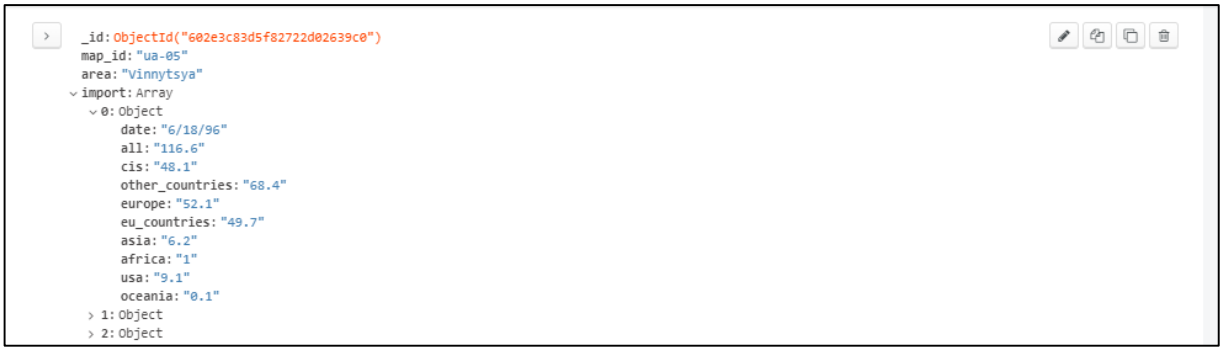


Рисунок 3.7 – Фрагмент інформації бази даних MongoDB

Як видно з рисунку 3.7 завдяки чіткій структурі та збалансованості вся інформація, якісно та зрозуміло підготовлена для подальшого використання, що в майбутньому забезпечить веб-сервіс безперебійним забезпеченням інформації.

3.3 Висновки

Отже, в даному розділі було здійснено дослідження на рахунок джерела даних, та як насправді їх можна правильно обробити та зберегти у потрібному форматі для зручної роботи з ними.

Також у даному було здійснено порівняльний аналіз бази даних MongoDB. В результаті її аналізу було виявлено такі переваги:

- Швидкість − будучи орієнтованою на документи базою даних, легко індексувати документи. Тому швидша реакція;
- Масштабованість – здатність обробляти великі дані можна, розділивши їх на кілька машин;
- Використання JavaScript − MongoDB використовує JavaScript, що є найбільшою перевагою. Оскільки, в пріоритеті розробка веб-ресурсу, де основною мовою розробки є JavaScript, ця перевага відіграє дуже важливу роль в подальшій розробці.

Здійснено статистичний збір даних та створено базу даних, яка забезпечить інформаційну веб-систему необхідною інформацією.

4 РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОЇ ВЕБ-СИСТЕМИ АНАЛІЗУ ЗОВНІШНЬОЇ ТОРГІВЛІ ТОВАРАМИ УКРАЇНИ

4.1 Архітектура програмного забезпечення веб-системи

Архітектура нашого сервісу базується на типовій моделі MVC. Наш рівень клієнта (подання) буде написаний на Javascript, HTML, CSS та його препроцесор SASS, використовуючи React.js як фреймворк. Цей рівень архітектури – це те, з чим користувач буде взаємодіяти, щоб отримати доступ до функцій нашого додатку. Перш за все, потрібно створити чітку структуру папок, файлів, імпортів та експортів яку диктує нам React. На рисунку 4.1 зображено файлову структуру яку використовує веб-сервіс.

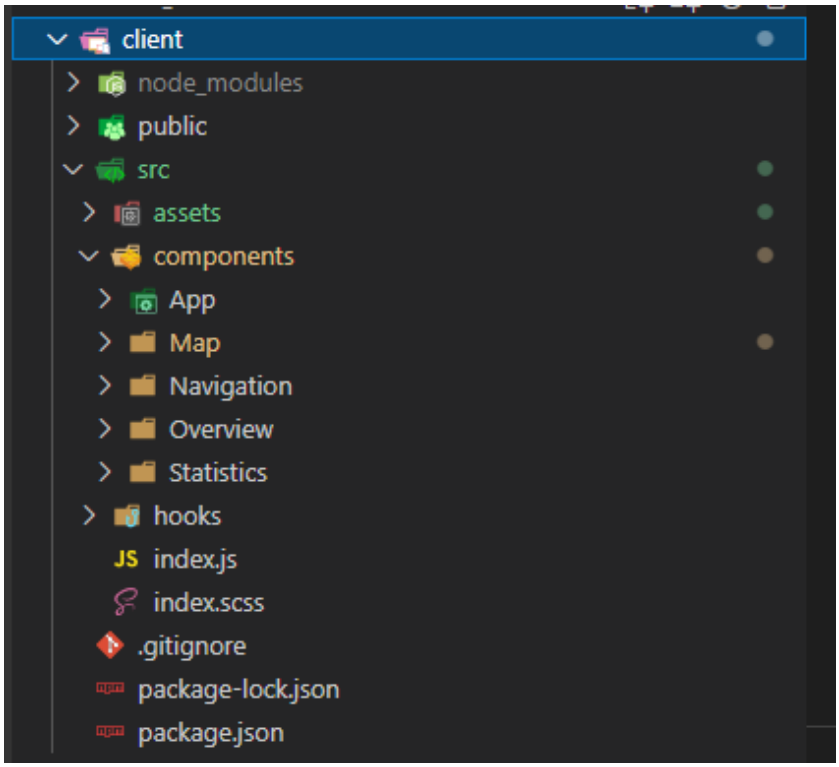


Рисунок 4.1 – Файлова структура клієнтської частини веб-сервісу

Щоб забезпечити чітку взаємодію сервера та клієнтської частини, перш за все, потрібно, згенерувати серверне середовище в якому будуть будуть

зберігатися всі залежності веб системи. В цьому нам допоможе Node.js та його пакетний менеджер npm.

Букви npm означають «node package manager». Коли ви працюєте над проектом JavaScript, ви можете використовувати npm для встановлення пакетів коду у свій власний проект. Ваш проект може бути веб-проектом, таким як веб-сайт або веб-програма, або це може бути проект на стороні сервера за допомогою Node.js. Будь-який проект JavaScript може використовувати npm для введення пакетів існуючого коду.

Npm – це інструмент, який ви встановлюєте на своєму комп'ютері. Він є частиною Node.

На npm опубліковано тисячі пакетів. Ви можете переглянути їх на npmjs.com [23].

Щоб використовувати пакети npm у проекті, ваш проект повинен містити файл із назвою package.json. Цей файл зберігає список усіх залежностей, які ви використовуєте, і те, яку версію кожного з них ви вибрали використовувати.

Якщо у вашому файлі немає package.json тоді ви можете створити його з командного рядка, запустивши npm init –у всередині вашого проекту.

Прийміть усі налаштування за замовчуванням, коли він задає питання, і після завершення новий package.json файл буде у вашому проекті [23].

Встановлення пакетів

Кожного разу, коли ви відкриваєте свій проект на іншому комп’ютері, можливо, доведеться переінсталювати всі свої пакети. Це особливо актуально, якщо ви переходите між Mac і Windows. Це пов’язано з тим, що не всі залежності проекту працюють однаково на різних операційних системах.

Після запуску npm install у вашому проекті з’явиться нова папка з назвою node_modules. Ця папка містить весь код усіх встановлених вами пакетів. Він також містить усі ті пакети, які встановили ваші пакети.

Щоб встановити новий пакет, використовуйте команду, npm install за якою йде назва пакета. Включить –save параметр, щоб переконатися, що бібліотека додана до вашого package.json.

Після налаштування серверного середовища розробки та встановлення відповідних залежностей розробки наш package.json повністю готовий до подальшого використання. Вся архітектура веб-сервісу описана в даному файлі (рис. 4.2).

Рисунок 4.2 – Базова архітектура веб-сервісу

Як видно з рисунку 4.2, веб-сервіс також використовує поле scripts, яке було створено власноруч. Ці команди можна використати, виконавши в корні проекту команду npm run та назва скрипта, наприклад, start. Проте найважливішою з них є команда npm run dev. Вона автоматично запускає, як

серверну частину проекту так і клієнтську. На рисунку 4.3 зображена модель веб-сервісу за якою буде працювати веб-сервіс.

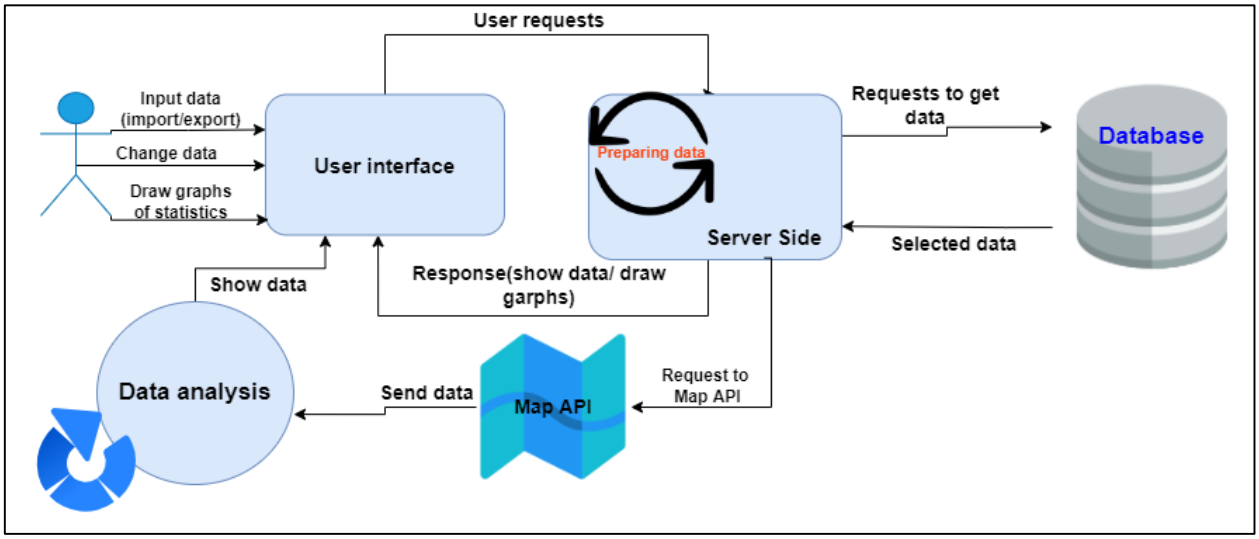


Рисунок 4.3 – Модель роботи веб-сервісу

Отже, після встановлення всіх залежностей веб-сервісу та налаштування середовища розробки, подальший прогрес над веб-сервісом не повинен вимагати занадто багато витрачених ресурсів.

4.2 Програмна реалізація інформаційної веб-системи

Після реалізації архітектури веб-системи, наступна задача полягає в, безпосередньо, розробці серверної частини.

Для того, щоб запустити найпростіший сервер на Node.js та Express.js потрібно створити вхідний файл запуску. Зазвичай таким файлом виступає index.js. Вхідний файл до веб-сервісу зображено на рисунку 4.4.

```
const express = require('express')
const mongoose = require('mongoose')
const router = require('./routers/route.js')
const path = require('path')

const app = express()
const PORT  = process.env.PORT || 8080
app.use(express.json({extended: true}))

app.use('/', router)

if (process.env.NODE_ENV === 'production') {
    app.use(express.static(path.join(__dirname, 'client/build')))

    app.get('*', (req, res) => {
        res.sendFile(path.join(__dirname, 'client/build', 'index.html'))
    })
}

const start = async() => {
    await mongoose.connect('mongodb+srv://bohdan:1234@cluster0.cvest.mongodb.net/Regions?retryWrites=true&w=majority', {
        useNewUrlParser: true,
        useUnifiedTopology: true,
        useFindAndModify: false,
        useCreateIndex: true
    })
    app.listen(PORT, () => {console.log(`Server has been launched at port ${PORT}`)})
}
start()

```

Рисунок 4.4 – Вхідний файл веб-сервісу

Як видно з рисунку 4.4 використовується асинхронне програмування, що забезпечує безперебійну роботу програми. Також аби покращити функціонал системи, у роботу залучена бібліотека mongoose.js.

MongooseJS – це картографічний документ об’єктів, який полегшує використання MongoDB, перекладаючи документи в базі даних MongoDB на об’єкти програми.

Три основні переваги використання Mongoose перед рідною MongoDB:

– MongooseJS забезпечує шар абстракції поверх MongoDB, що позбавляє потреби використовувати іменовані колекції;

– Моделі в Mongoose виконують основну частину роботи зі встановлення значень за замовчуванням для властивостей документа та перевірки даних;

– Функції можуть бути приєднані до Моделей у MongooseJS. Це дозволяє безперешкодно включати нові функції [24].

Запити використовують ланцюжок функцій, що призводить до отримання коду, який є більш гнучким і читабельним, отже, також більш ремонтопридатним.

Кінцевим результатом цього є спрощення доступу до бази даних із додатків. Основним недоліком Mongoose є те, що абстракція коштує продуктивності порівняно з корінною MongoDB.

Для того, щоб уникнути багів у програмі, потрібно реалізувати схему бази даних, яка буде обробляти кожен запит із клієнтської частини. Схема бази даних зображена на рисунку 4.5.

```
1   const {Schema, model} = require('mongoose')
2
3   const region = new Schema({
4       area: String,
5       map_id: String,
6       import: [{
7         date: Date,
8         all: String,
9         cis: String,
10        other_countries: String,
11        europe: String,
12        eu_countries: String,
13        asia: String,
14        africa: String,
15        usa: String,
16        oceania: String
17        }],
18        export: [{
19          date: Date,
20          all: String,
21          cis: String,
22          other_countries: String,
23          europe: String,
24          eu_countries: String,
25          asia: String,
26          africa: String,
27          usa: String,
28          oceania: String
29          }]
30  })
31
32  module.exports = model('Region', region, "regions")
```

Рисунок 4.5 – Схема бази даних веб-сервісу

Усі запити із клієнтської частини обробляються за допомогою такого концепту як роутинг та концепту передачі даних REST API.

REST API – це прикладний програмний інтерфейс (API), який використовує HTTP-запити для отримання, вилучення, розміщення і видалення даних. Абревіатура REST в контексті API розшифровується як «передача стану уявлення» (Representational State Transfer).

Основна ідеї REST API полягає в поділі різних при зверненні до одного і того ж URL за допомогою HTTP методів, основні з яких:

– GET – використовується для отримання даних;

– POST – використовується для створення нового запису;

– PUT – використовується для оновлення вже існуючої записи

– PATCH – використовується для оновлення, але тільки тоді, коли змінюється ідентифікатор запису;

– DELETE – використовується для видалення запису [16].

На рисунках 4.6 та 4.7 зображено обробку запиту із клієнтської частини та його зворотня відповідь.

```
const {Router} = require('express')
const Region = require('../models/model')
const {getAreaByName, getAll, getAreaForTable, patchDataFromTable, getAllByIds} = require('../controllers/controler.js')
const router = Router()

router.post('/map', async (req, res) => {
    const {direction} = req.query
    const {id, selectedValues} = req.body
    const region = await Region.findOne({map_id: id}, err => {
        if (err) {
            res.status(404).json('Not found')
        }
    })

    const obj = region[direction].filter(item => {
        let reg
        selectedValues.forEach(year => {
          if (+year == new Date(item.date).getFullYear()) {
            reg = item
          }
        })
        return reg
    })

    res.status(200).json({obj})
})
```

Рисунок 4.6 – Фрагмент обробки POST-запиту веб-системи

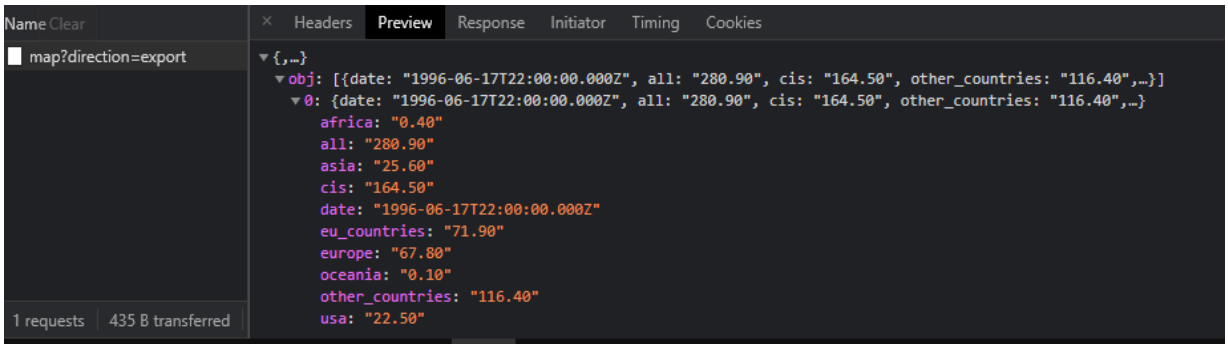


Рисунок 4.7 – Відповідь обробки POST-запиту

Після того як, серверна частина була реалізована, щоб запустити у безперебійну роботу сервер достатньо прописати в командній стрічці npm run dev. Результат роботи сервера зображено на рисунку 4.8.

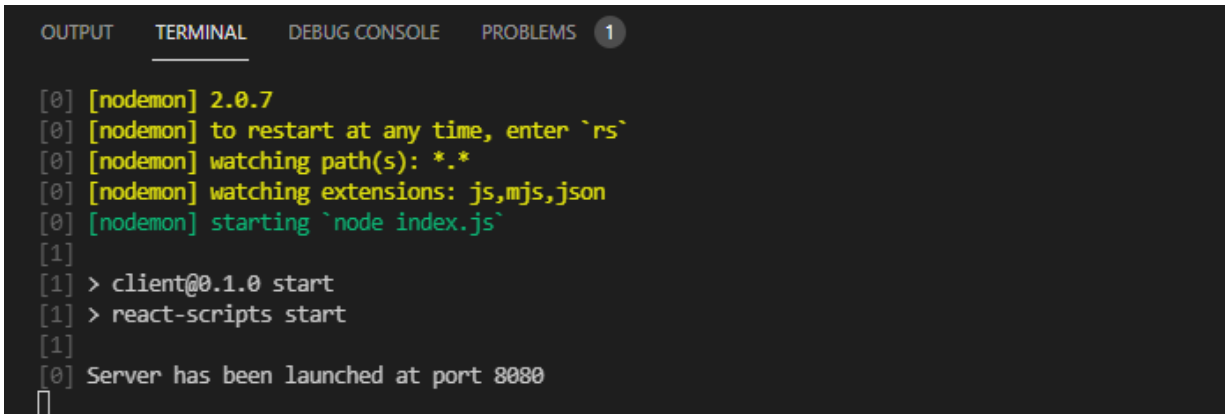


Рисунок 4.8 – Запуск сервера системи

Після того, як серверна частина була реалізована, саме час приступити до розробки інтерфейсу користувача. Як раніше було вже згадано в цьому, допоможе бібліотека javascript React.js та бібліотека css Bootstrap із препроцесором SASS.

Перш за все, щоб згенерувати базові залежності проекту потрібно скористатися інструментом React.js npx create-react-app client.

Команда допоможе згенерувати всі базові залежності проекту та допоможе чітко налаштувати такі інструменти, як Webpack та Babel, засоби розробки, які будуть інтерпретувати всі залежності проекту в один вихідний файл bundle.js.

На рисунку 4.9 зображено генерацію базових залежностей користувацького інтерфейсу.

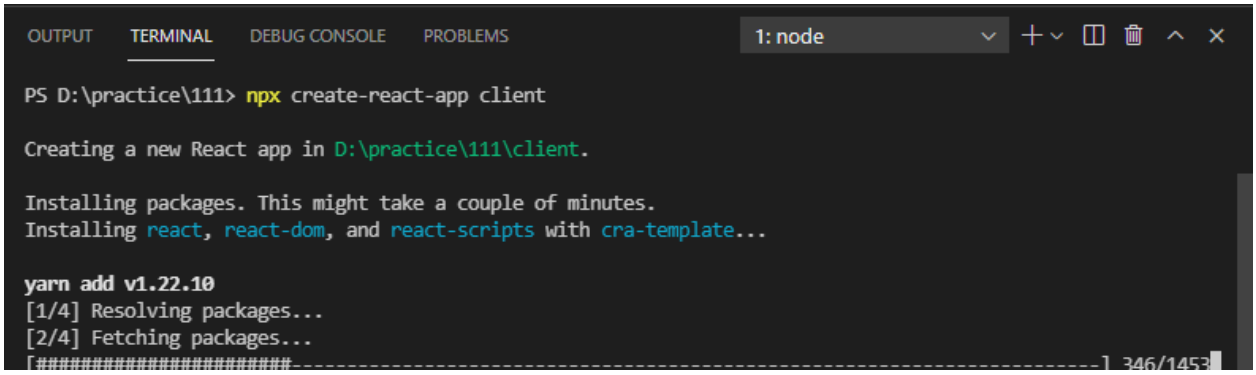


Рисунок 4.9 – Генерація базових залежностей клієнтського інтерфейсу

Як тільки завершиться процес завантаження необхідних даних, можна приступати до розробки інтерфейсу. Оскільки, за умовою завдання наша веб-система буде включати декілька компонентів, таких як, перегляд та редагування даних, робота з картою та побудова тематичних діаграм, відображення статистичних графіків, виникає потреба у налаштуванні роутингу. В цьому допоможе інструмент React react-router.

React Router – одна з найпопулярніших платформ маршрутизації для React. Бібліотека розроблена з інтуїтивно зрозумілими компонентами, що дозволяє створити декларативну систему маршрутизації для вашого додатка. Це означає, що ви можете точно заявити, який із ваших компонентів має певний маршрут. За допомогою декларативної маршрутизації ви можете створювати інтуїтивно зрозумілі маршрути, зручні для читання, полегшуючи управління архітектурою вашого додатка [25].

На рисунку 4.10 відображено налаштування роутингу між компонентами інтерфейсу.

```
import React from 'react'
import {Route, Switch, BrowserRouter as Router} from 'react-router-dom'
import Navigation from '../Navigation/Navigation'
import $ from 'jquery';
import Map from '../Map/Map'
import Overview from '../Overview/Overview'
import Statistics from '../Statistics/Statistics'      You, 21 months ago • first
import { DataAnalysis } from '../DataAnalysis';

import './App.scss';

function App() {
  return (
    <Router>
      <Navigation/>
        <Switch>
          <Route exact path="/" component={Overview} />
          <Route path="/map" component={Map} />
          <Route path="/statistics" component={Statistics} />
          <Route path="/data-analysis" component={DataAnalysis} />
        </Switch>
    </Router>
  )
}

export default App

```

Рисунок 4.10 – Налаштування роутингу веб-системи

Щоб користувач мав змогу взаємодіяти із роутингом була створена панель навігації, для більш зручної взаємодії між компонентами. Фрагмент реалізації навігації та її клієнтська частина відображенні на рисунках 4.11, 4.12.

```jsx
import React from 'react'
import 'bootstrap/dist/css/bootstrap.css';
import 'bootstrap/dist/js/bootstrap.js';
import './Navigation.scss'
import {Link} from 'react-router-dom'


const Navigation = () => {
    return (
        <>
        <ul className="nav nav-pills flex-column links">
            <li className="nav-item ">
                <Link className="nav-link" to="/">Overview and Edit</Link>
            </li>
            <li className="nav-item">
                <Link className="nav-link" to="/map">Map</Link>
            </li>
            <li className="nav-item">
                <Link className="nav-link" to="/statistics">Statistics Graph</Link>
            </li>      You, 21 months ago • first commit …

            <li className="nav-item">
                <Link className="nav-link" to="/data-analysis">Data Analysis</Link>
            </li>
        </ul>
        </>
    )
}

export default Navigation
```

Рисунок 4.11 – Програмна реалізація між компонентами веб-системи

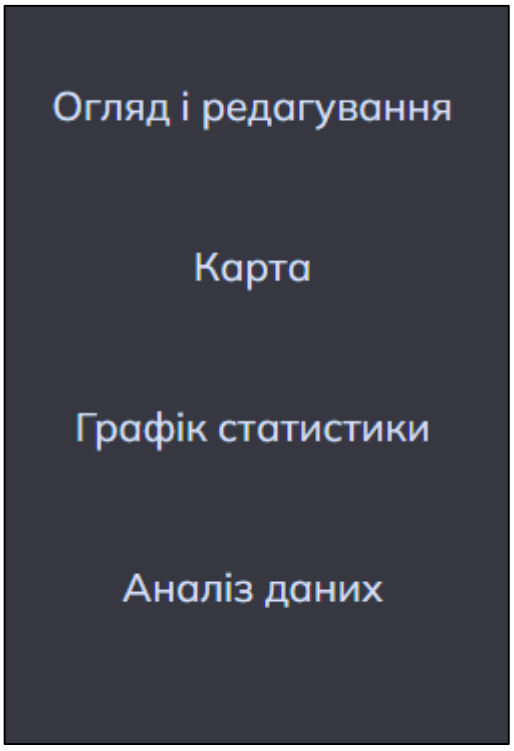


Рисунок 4.12 – Клієнтська частина навігації

Для того щоб забезпечити більш гнучку та зручну систему управління та редагування даними, було розроблено окремий компонент, який допомагає досягти поставлених цілей. На рисунку 4.13 відображений компонент веб-системи, який відповідає за редагування та перегляд вихідних даних. Всі дані представленні у мільйонах доларів США.

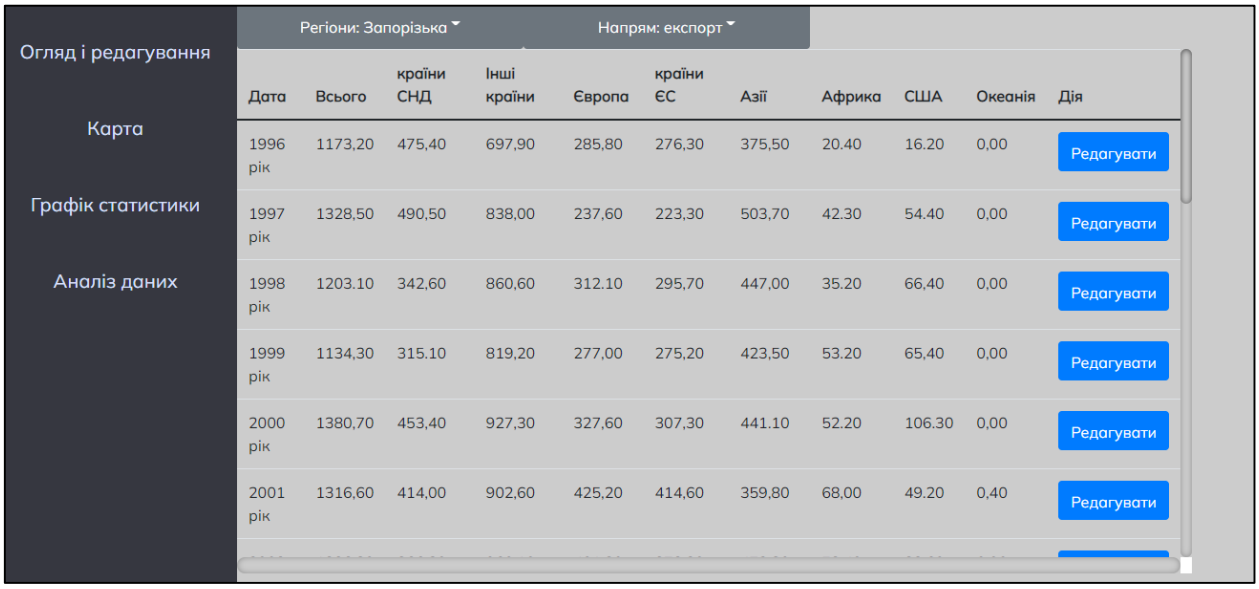


Рисунок 4.13 – Компонент редагування та перегляду вихідних даних

Як представлено на рисунку 4.13 у користувача реалізована повна взаємодія з даними, редагування та перегляд можна здійснювати, як по напрямкам, будучи то чи імпорт чи експорт, так і по регіонам України.

Наступний пункт нашого завдання це реалізація роботи із картою та відображення тематичних карт. Тому для відображення карти було обрано інструмент React react-vector-maps.

React-vector-maps компонент React для створення простих карт з інтерактивними елементами, більше 100 карт, готових до використання, та онлайн-конвертер SVG в JSON для створення нових карт.

Для підключення карти до веб-сервісу, react-vector-maps надає дані для конкретної області із власного ресурсу. Фрагмент даних карти зображено на рисунку 4.14.

```
{
  "id": "ukraine",
  "name": "Ukraine",
  "viewBox": "0 0 612.47321 408.0199",
  "layers": [
    {
      "id": "ua-40",
      "name": "Sevastpol' City",
      "d": "m 386.51191,387.77242 0.86,-0.12 0.87,0.54 2.18,0.08 0.53,0.47 0.33,1.49 -0.67,0.34 -0.97,-0.14 -0.43,0.48 0.1,0.75 0.86,0.81 2.42,
    },
    {
      "id": "ua-43",
      "name": "Crimea",
      "d": "m 491.21191,360.14242 0.07,0.29 -0.53,0.31 -0.75,1.3 -1.98,0.04 -1.66,-0.79 -0.76,0.42 -0.22,1.07 1.23,0.81 -0.57,0.59 -1.44,0.41 -
    },
    {
      "id": "ua-71",      You, 21 months ago • first commit …
      "name": "Cherkasy",
      "d": "m 338.02191,116.65242 0.27,0.22 -0.42,0.62 1.69,1.34 0.46,1.1 4.16,1.21 0.19,1.28 -1.76,0.42 0.68,0.72 -0.52,0.81 1.86,2.32 0.36,-0
    },
    {
      "id": "ua-74",
      "name": "Cherniniv",
      "d": "m 374.66191,0.812423 0.38,-0.11 3.23,1.38 0.46,-0.23 0.65,0.21 1.22,-1.12 0.63,0.04 0.76,0.66 -0.1,0.49 0.51,0.25 0,0 -0.47,1.25 -0
    },
    {
      "id": "ua-77",
      "name": "Chernivtsi",
      "d": "m 118.42191,197.42242 2.85,0.56 0.79,1.5 0.74,-1.53 0.54,-0.62 0.39,-0.01 0.65,0.38 1.72,2.32 -0.15,1.27 0.84,-0.13 1.12,0.66 0.88,
    },
    {
      "id": "ua-12",
      "name": "Dnipropetrovs'k",
      "d": "m 398.16191,185.65242 1.25,1.18 1.83,0.75 1.18,-0.02 0.57,1.13 1.11,0.84 1.75,0.21 0.22,0.41 -0.19,1.12 0.34,0.56 0.82,0.34 0.68,-0
    },
    {
      "id": "ua-14",
```


Рисунок 4.14 – Дані для відображення карти

Після чого, в нас є вже всі ресурси для відображення карти в компоненті. Результат відображення карти зображено на рисунку 4.15.

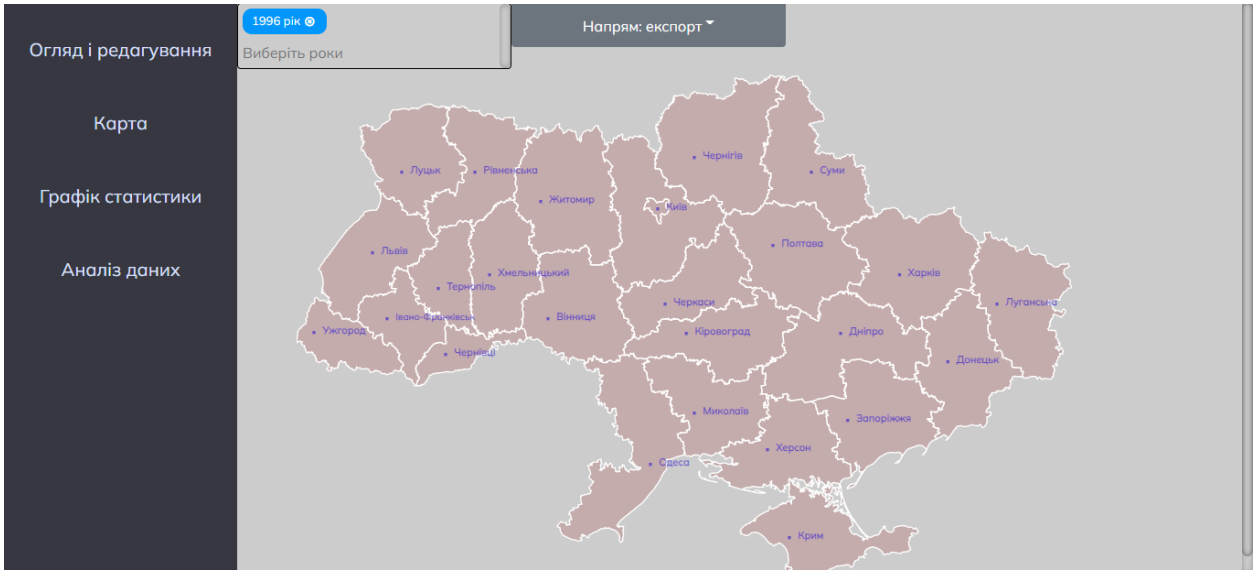


Рисунок 4.15 – Карта компонента аналізу вихідних даних

Як можна бачити на рисунку 4.15 в компоненті карта реалізований функціонал роботи із картою. Аналіз та візуалізацію даних можна проводити як за інтервалом років так і за одним конкретним роком, також реалізований функціонал вибору за напрямком (імпорт, експорт). На рисунку 4.16 відображено процес роботи компонента Карта.

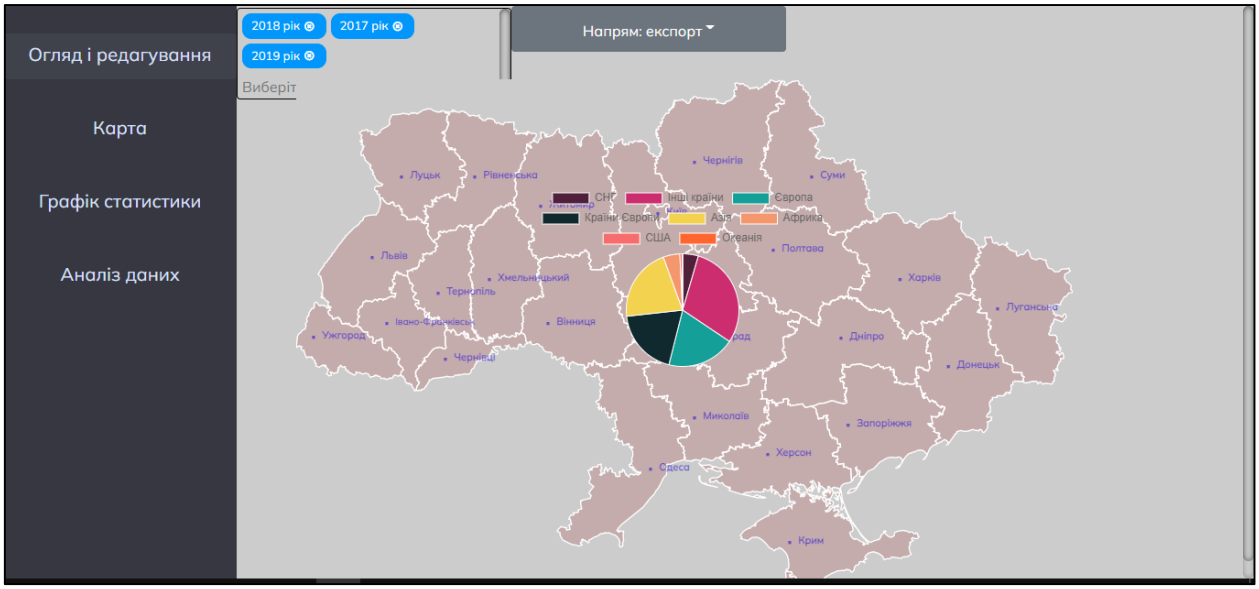


Рисунок 4.16 – Процес обробки даних

При роботі з картою користувач може вибрати будь-який регіон, та будь-який інтервал років та в результаті отримати діаграму, сектори якої вказують на напрямок куди ведеться торгівля.

Наступним кроком завдання є реалізація компонента статистики.

Для цього був обраний інструмент React react-chartjs-2.

React-chart-2 інструмент для аналізу та відображення даних у відповідних графіках (рис. 4.17).

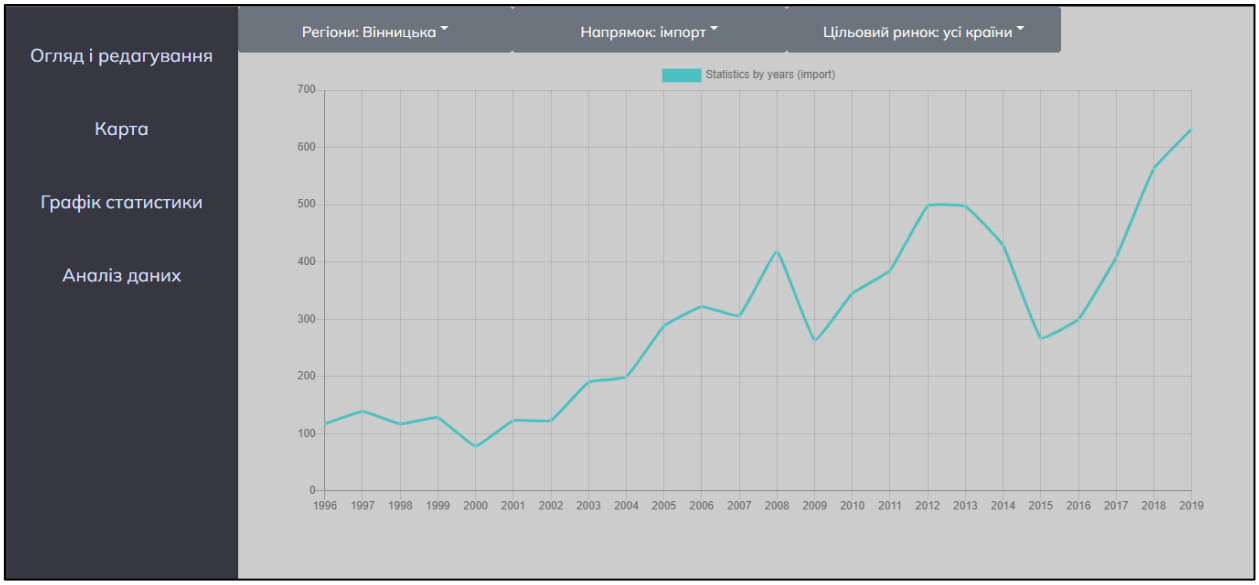


Рисунок 4.17 – Компонент графік статистики

Як можна бачити з рисунку 4.17 користувач може повністю взаємодіяти з даними сервісу. Фільтри можна встановити як по регіонах, напрямках (імпорт, експорт) та цільовому ринку.

Графік повністю динамічний та при взаємодії відображає у невеликому інформаційному вікні інформацію по конкретно встановленому фільтрі.

Останній компонент, який був реалізований у ході виконання завдання, був компонент аналізу та порівняння даних, товарообігу імпорту та експорту між областями. Користувач може обрати декілька областей за допомогою карти України вибрати області, які необхідні для аналізу, та на основі вибраних вихідних даних користувач отримає порівняльний графік статистики між областями та таблицю товарообігу кожної області. На рисунку 4.18 та 4.19 відображено процес роботи компонента.

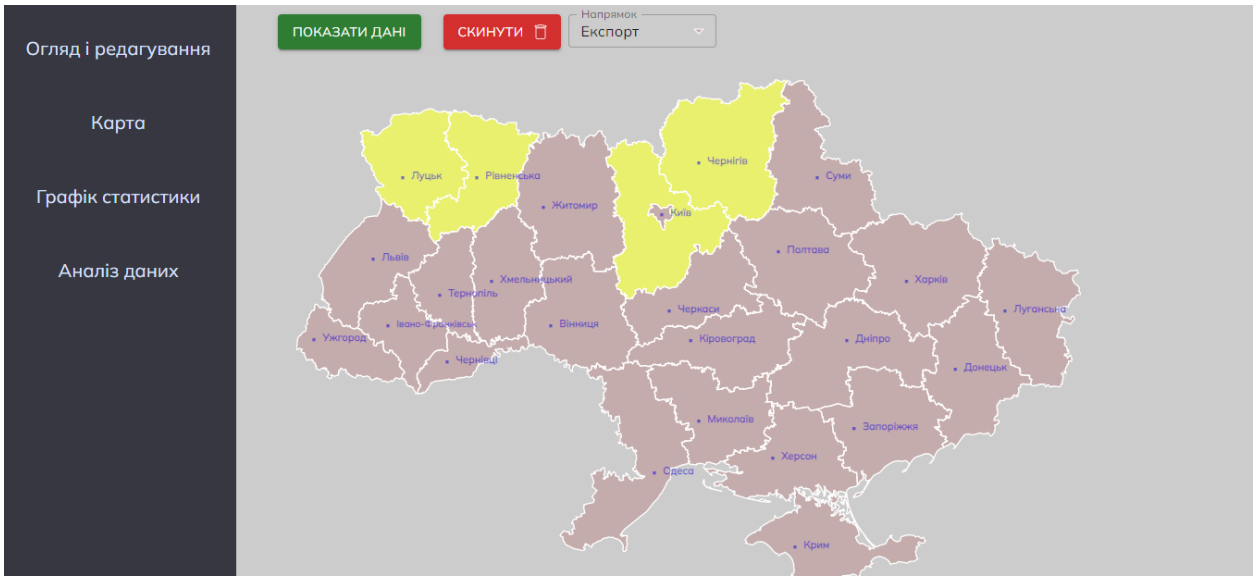


Рисунок 4.18 – Вибір об’єктів аналізу даних

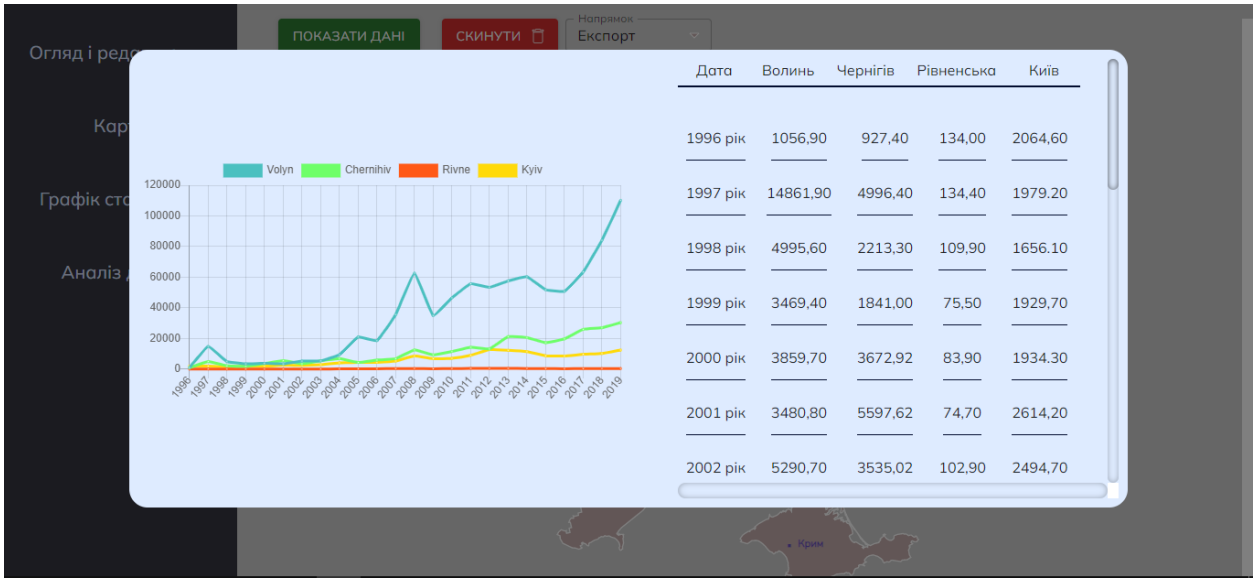


Рисунок 4.19 – Обробка даних на основі вибраних областей за допомогою карти України

Отже, в ході розробки було повністю розроблено серверну частину веб сервісу, оброблено всі запити на дані із клієнтської частини, також розроблено повністю інтерфейс користувача, в який було імплементовано перегляд та редагування вихідних даних, роботу з картою та відображення тематичних графіків, і наостанок було реалізовано компонент графік статистики, де

користувач може повністю реалізувати візуалізацію даних за заданими фільтрами.

4.3 Аналіз динаміки географічної структури зовнішньої торгівлі

На основі створеної веб-системи також можемо проводити статистичний аналіз даних. Отже, за основу аналізу вибрано три частини області України, для більш явного контрасту було обрано Львів, Київ та Харків.

Перш за все статистика буде відображатися на компоненті Карта. Інформація буде відображена за 2017-2019 роки та напрямок експорт.

Отже, на рисунках 4.20 – 4.22 зображенні статистичні діаграми Львівської, Харківської та Київської областей.

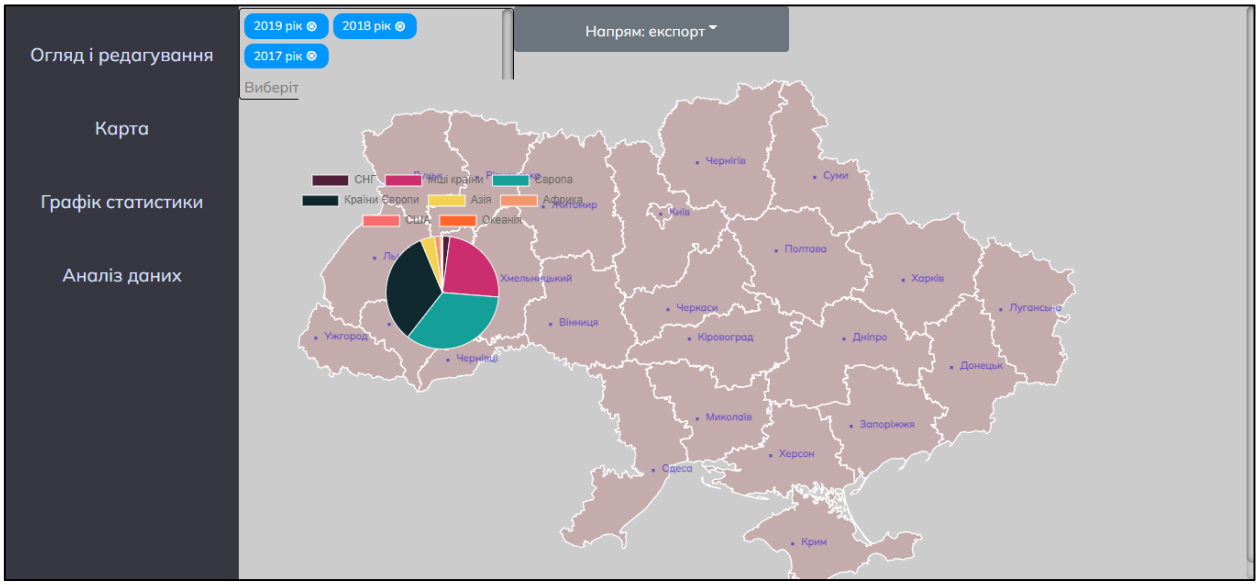


Рисунок 4.20 – Експортна торгівля (2017-2019) Львівської області

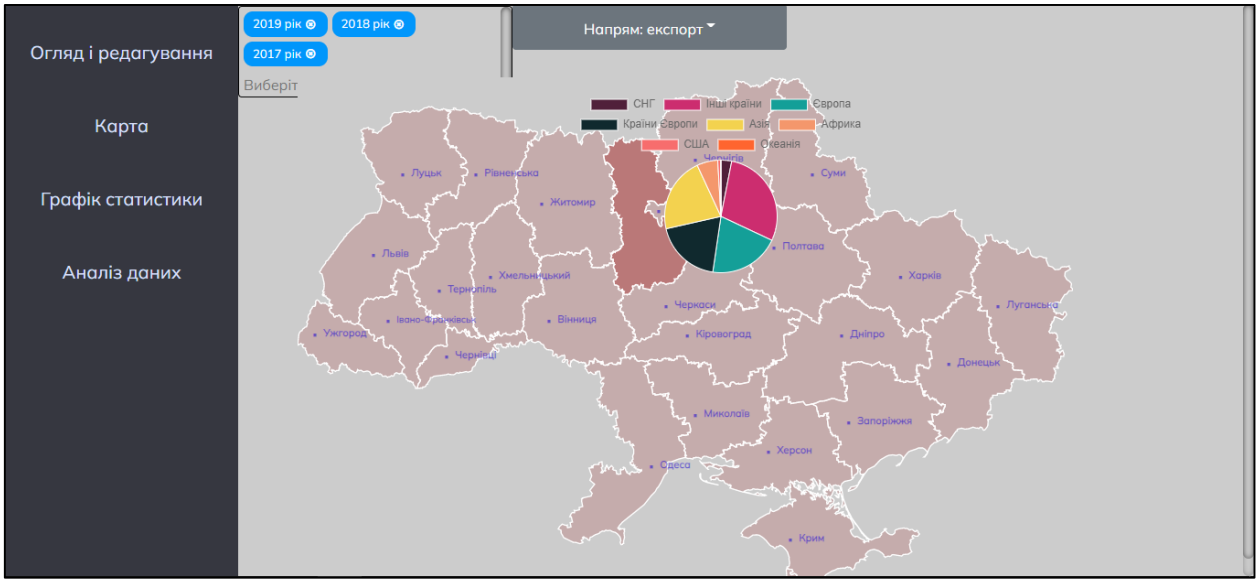


Рисунок 4.21 – Експортна торгівля (2017-2019) Київської області

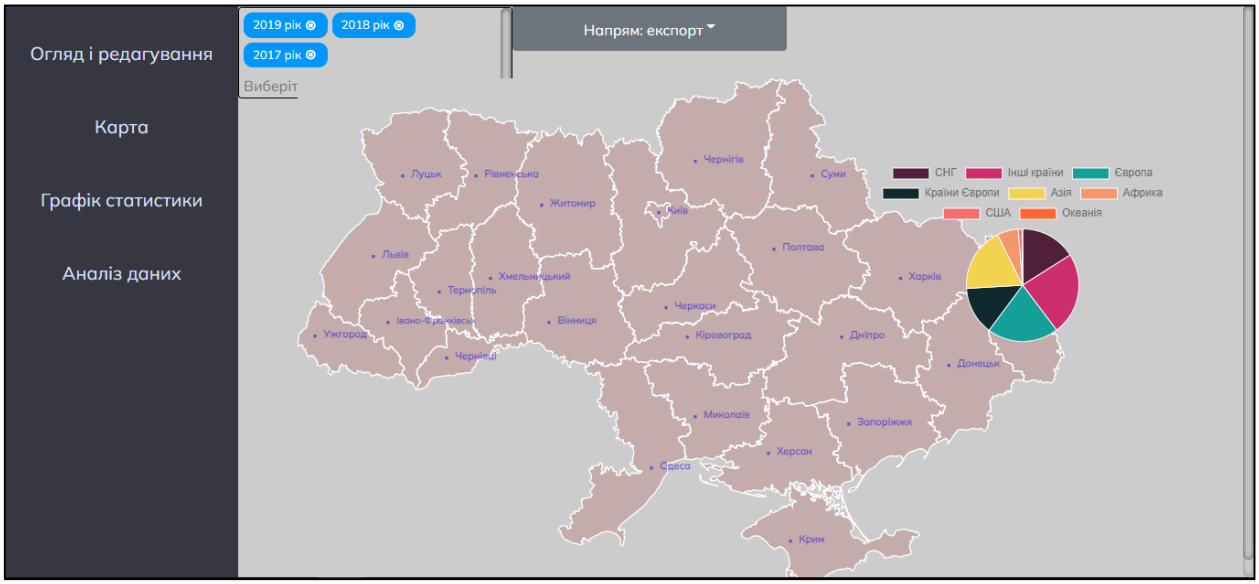


Рисунок 4.22 – Експортна торгівля (2017-2019) Харківської області

Як можна спостерігати із статистичних діаграм, тенденція вказує на те, що чим західніше знаходиться регіон, тим інтенсивніша торгівля між Європою та країнами Європи, також ситуація відбувається і на сході країни, найбільш активна торгівля з країнами СНГ та Азією.

Задля достовірності інформації можна також побудувати графіки статистики за 1996-1998 роки (рис 4.23 – 4.25).

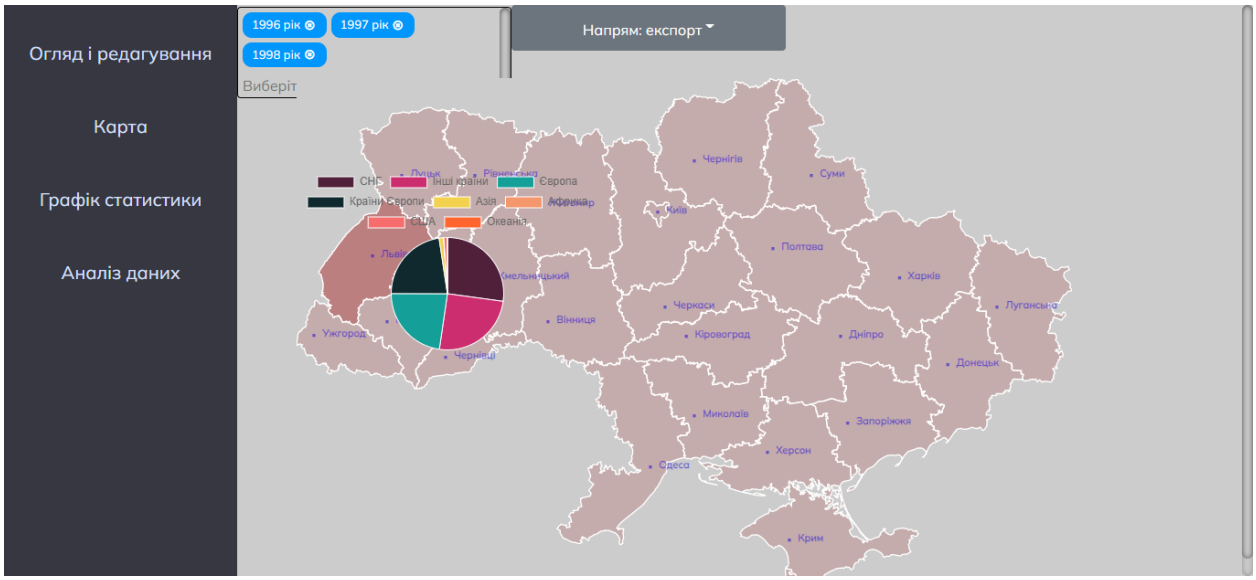


Рисунок 4.23 – Експортна торгівля (1996-1998) Львівської області

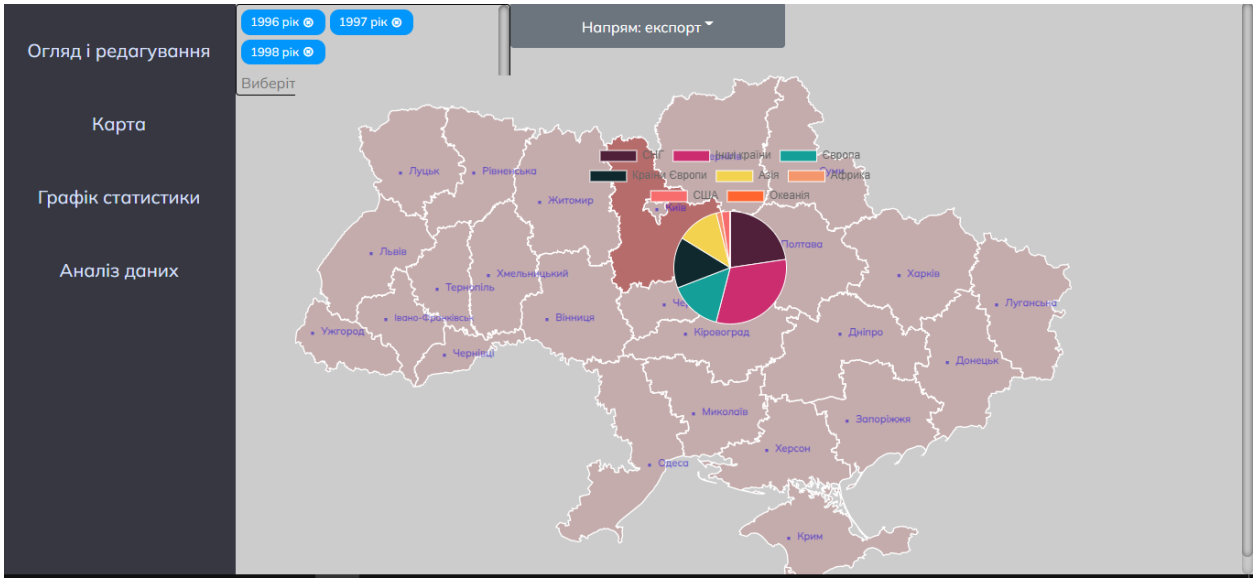


Рисунок 4.24 – Експортна торгівля (1996-1998) Київської області

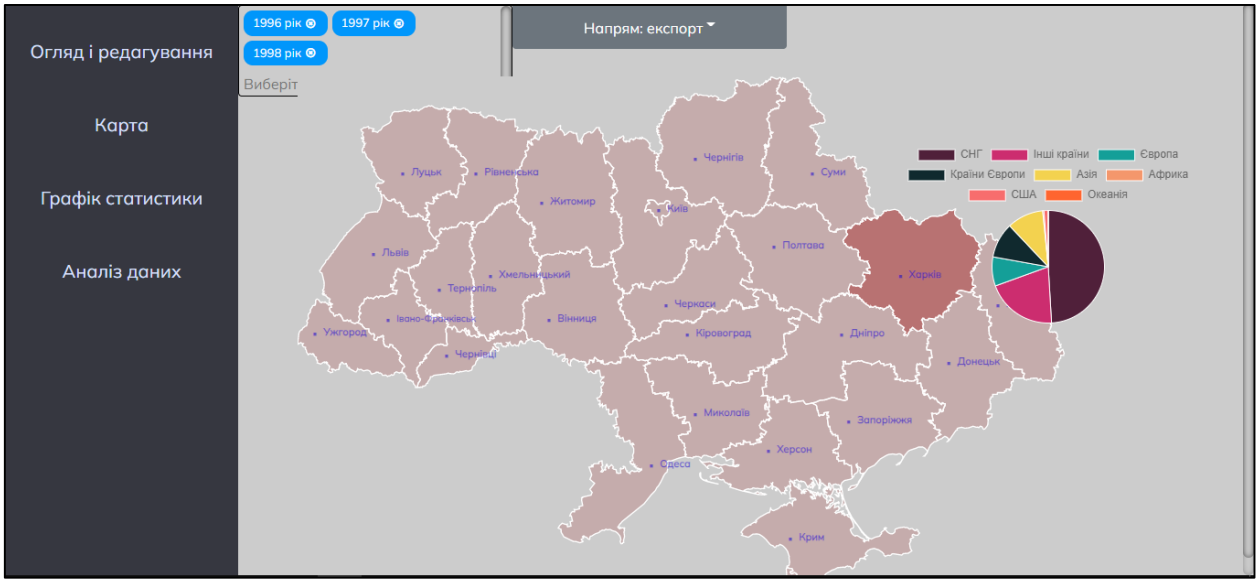


Рисунок 4.25 – Експортна торгівля (1996-1998) Харківської області

Порівнюючи статистичні діаграми за 2017-2019 та за 1996-1998 роки можна зробити висновок, що тенденція експортної торгівлі товарами за останні 20 років майже не змінилася.

За допомогою статистичних графіків також можна зробити аналіз закордонної торгівлі товарами по роках. За об’єкт аналізу було обрано київську та вінницьку області. На рисунках 4.26 та 4.27 зображено графік статистики експорту товарами кожної з цих областей.

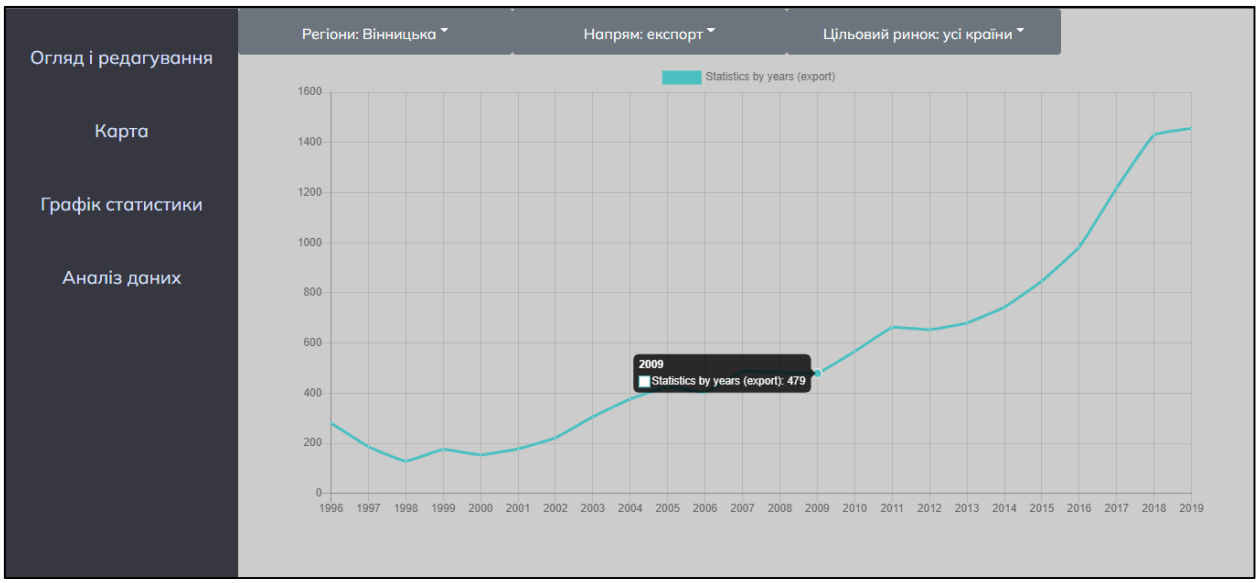


Рисунок 4.26 – Експортна торгівля Вінницької області

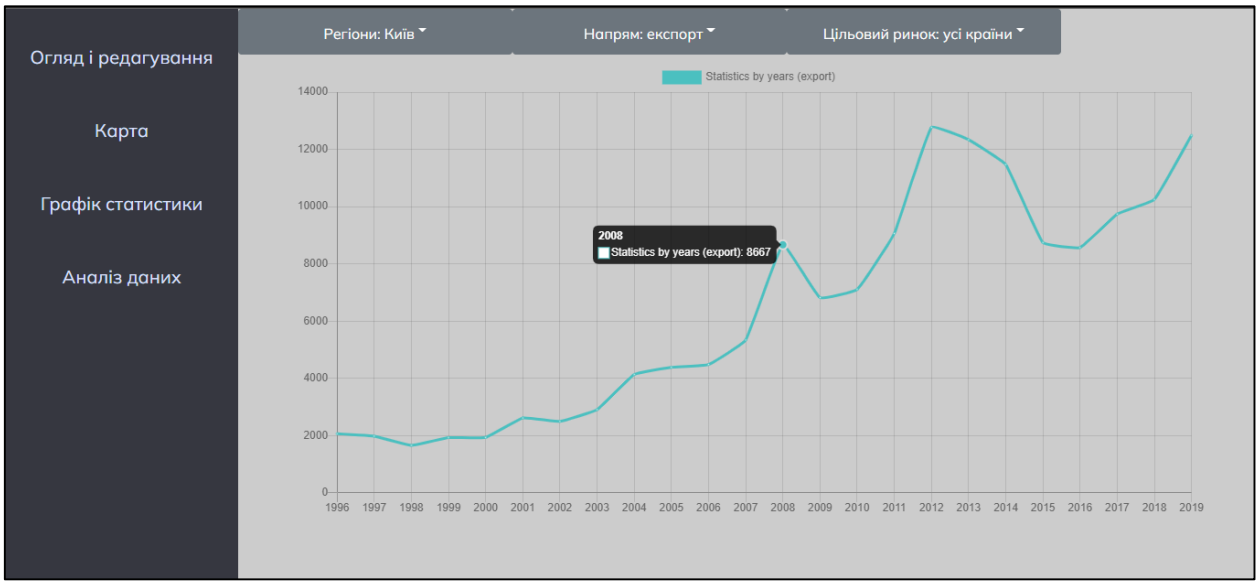


Рисунок 4.27 – Експортна торгівля Київської області

Зробивши порівняльний аналіз статистичних графіків, можна зробити висновок, що попри значну різницю в товарообігу між областями тенденція до збільшення у валового обороту товарами зростає по мірі розвитку закордонної торгівлі. Цю тенденцію можна чітко відслідкувати за рисунками 4.26 та 4.27.

За допомогою компонента Аналіз даних, можна здійснити більш комплексний порівняльний, аналіз та моніторинг даних між кожною з областей. Що дасть можливість оцінити товарообіг між регіонами. Отже на рисунку 2.28, 2.29 зображено порівняльний аналіз між областями: Волинська, Рівненська, Київська, Чернігівська та Харківська.

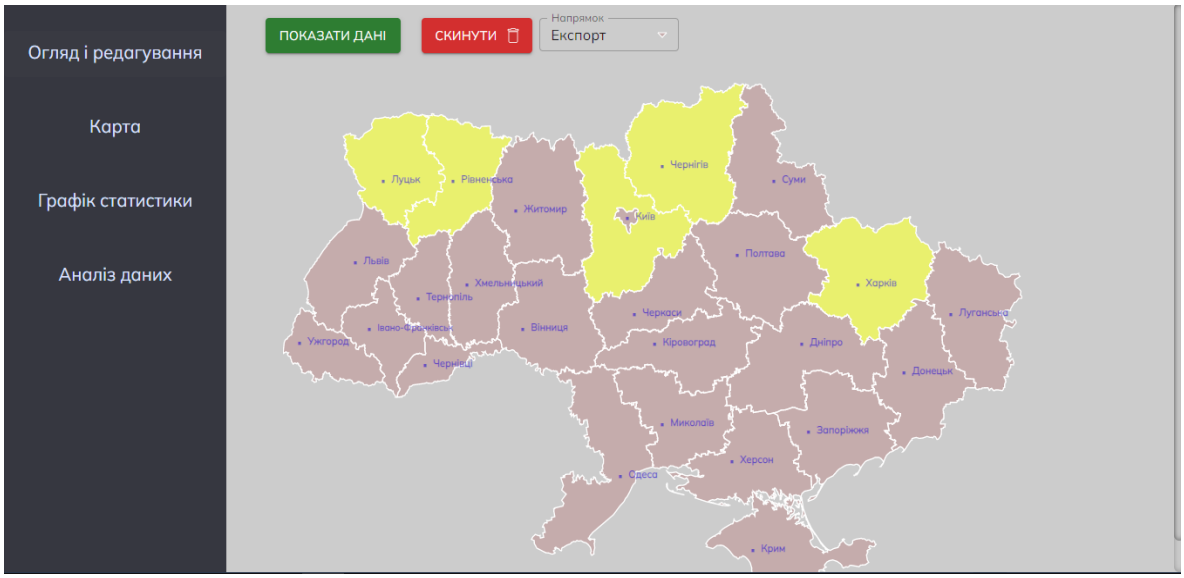


Рисунок 4.28 – Вибір об’єктів порівняльного комплексного аналізу між Волинською, Рівненською, Київською, Чернігівською та Харківською областями

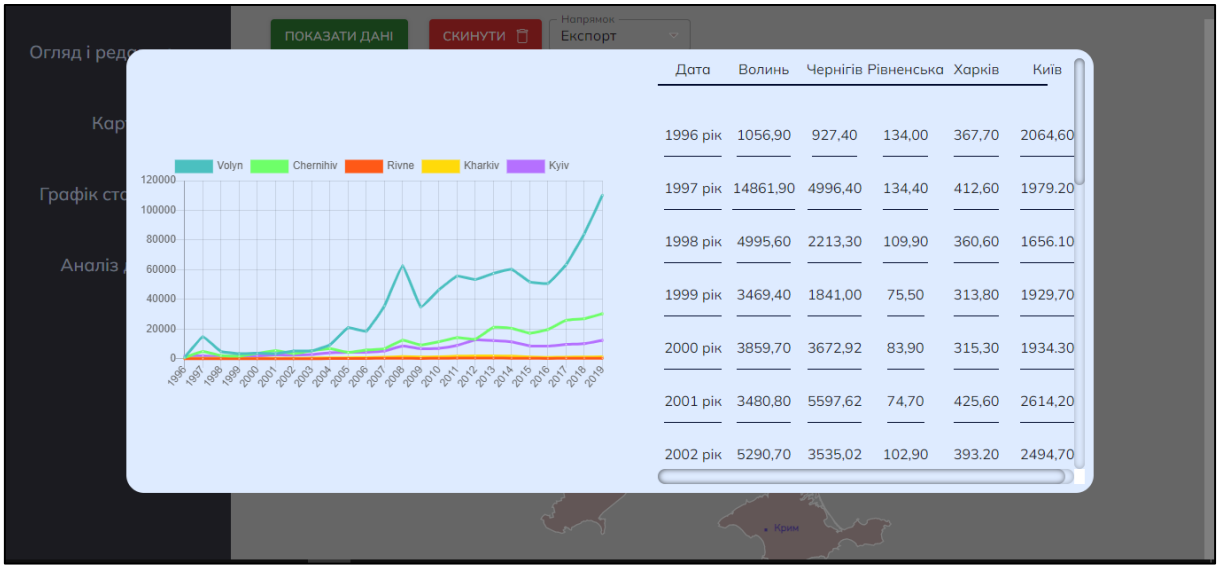


Рисунок 4.29 – Комплексний порівняльний аналіз даних між Волинською, Рівненською, Київською, Чернігівською та Харківською областями

Як можна побачити з рисунку 4.29 найбільший товарообіг за кордон відбувається у Волинській області, по всім рокам, починаючи з 1996 і по 2019 товарообіг найбільший у даній області.

Товарообіг у Київській та Чернігівській області має більш середнє значення по ринку у порівняні з іншими регіонами. Є

Найменший товарообіг спостерігається у Харківській та Рівненській області. Починаючи з 1996 і по 2019 їхній товарообіг майже не зріс та залишися практично на одному рівні на протязі двадцяти трьох років.

Для більш достовірного аналізу та моніторингу можна зробити комплексний порівняльний аналіз, також і по південних регіонах країни. На рисунку 4.30, 4.31 зображено порівняльний аналіз між областями: Одеська, Миколаївська, Херсонська та Запорізька.

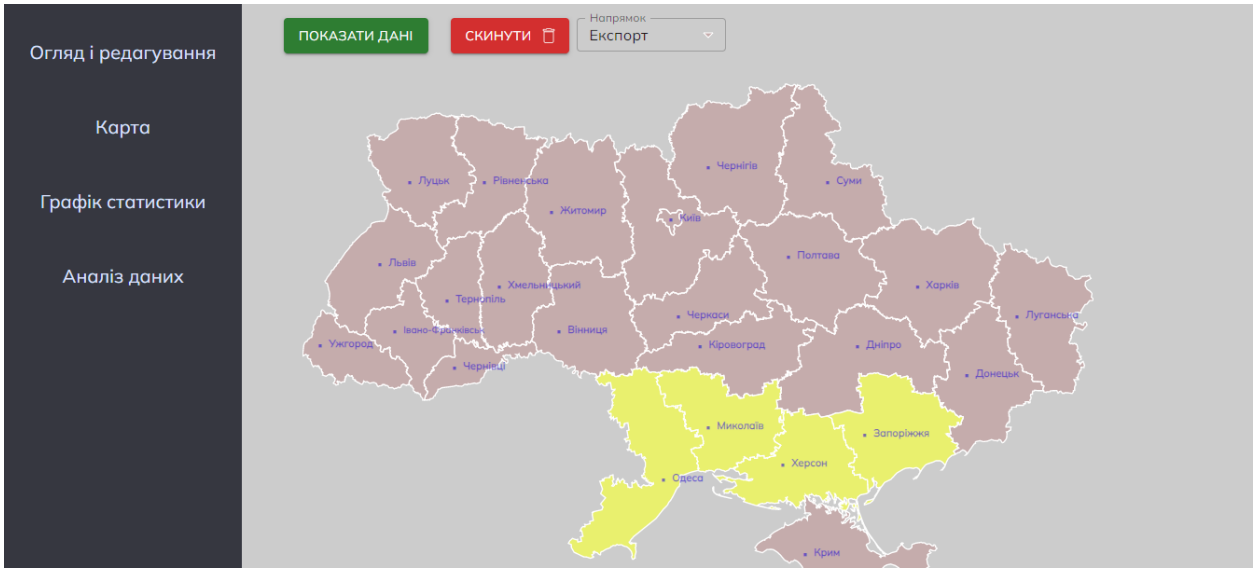


Рисунок 4.30 – Вибір об’єктів порівняльного комплексного аналізу між Одеською, Миколаївською, Херсонською та Запорізькою областями

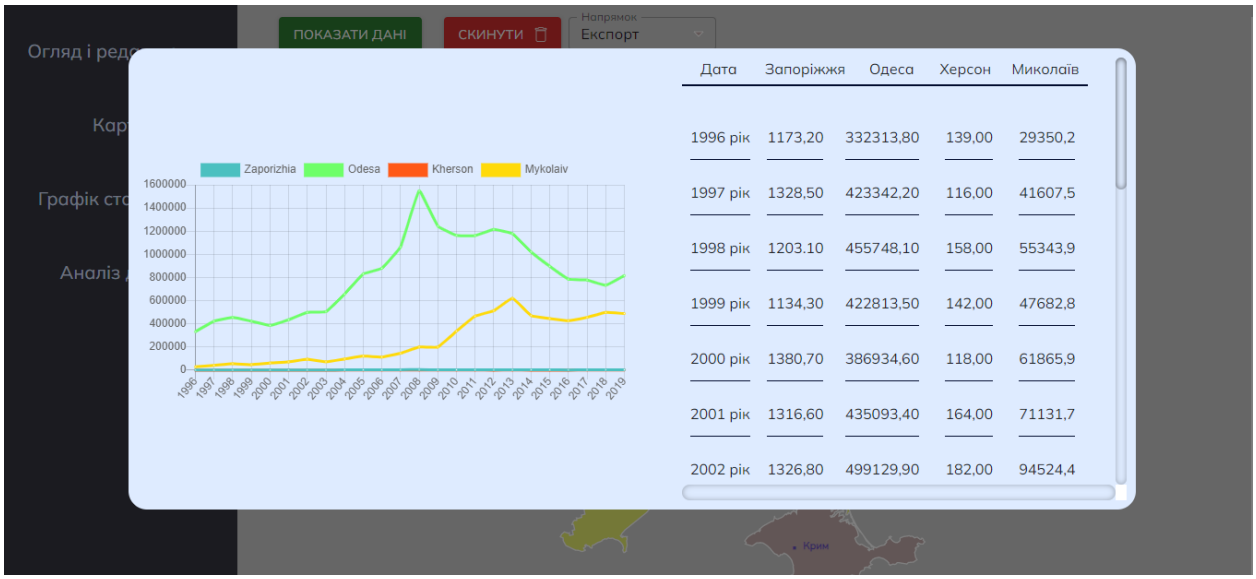


Рисунок 4.31 – Комплексний порівняльний аналіз даних між Одеською, Миколаївською, Херсонською та Запорізькою областями

Як можна побачити з рисунку 4.31 найбільший товарообіг у порівнянні з обраними областями у Одеській та Миколаївський області, скоріш за все такий суттєвий вплив на обіг товару впивають морські порти, які знаходяться в даних регіонах.

В порівнянні з у Одеською та Миколаївською областями, Херсонська Та Запорізька області мають мізерний товарообіг, пороте як видно із таблиці даних тенденція на ріст все ж є в даних регіонах.

Отже, зробивши статистичний аналіз закордонного ринку України можна зробити висновок, Україна має досить непогану тенденцію до розвитку міжнародних торгово-економічних відносин, має досить багато напрямків до збуту та закупівлі товарів, та що з року в рік товарообіг все більше набирає обертів.

4.4 Висновки

В даному розділі, було розроблено архітектуру програмного забезпечення, де було описано всі найважливіші аспекти розробки веб-

застосунку. Після чого, було створено серверну та клієнтську частини інформаційної веб-системи де розроблено:

– можливість перегляду та редагування даних;

– підключення карти України, за допомогою якої є змога по регіонам аналізувати дані;

– побудову тематичних карт на основі вибірок з бази даних за параметрами, вказаними користувачем: інтервал років, імпорт чи експорт;

– побудову графіків, які надають змогу відстежити тенденцію імпорту чи експорту по рокам;

– побудову порівняльних графіків, які надають змогу проаналізувати тенденцію закордонного ринку між кожною з областями.

В результаті розробленої функціональності веб системи, користувач має змогу провести покращений комплексний аналіз структури зовнішньої торгівлі товарами України в розрізі областей.

В кінці розділу проведено статистичний аналіз закордонного ринку та виявлено, що Україна має досить непогану тенденцію до розвитку міжнародних торгово-економічних відносин, має досить багато напрямків до збуту та закупівлі товарів, та що з року в рік товарообіг все більше набирає обертів.

5 ЕКОНОМІЧНА ЧАСТИНА

5.1 Оцінювання комерційного потенціалу розробки

Метою проведення комерційного та технологічного аудиту є оцінювання комерційного потенціалу інформаційної веб-системи аналізу динаміки географічної структури зовнішньої торгівлі товарами України.

Для проведення технологічного аудиту було залучено 3-х незалежних експертів Вінницького національного технічного університету кафедри системного аналізу та інформаційних технолгій: к.т.н., доц. Козачко О.М., к.т.н., доц. Крижановський Є. М., к.т.н., доц. Варчук І. В. Для проведення технологічного аудиту було використано таблицю 5.1 в якій за п’ятибальною шкалою використовуючи 12 критеріїв здійснено оцінку комерційного потенціалу [26].

Таблиця 5.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів

Продовження таблиці 5.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років

Продовження таблиці 5.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
				3-х до 5-ти років	
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки пові-домлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

В таблиці 5.2 неведені рівні комерційного потенціалу розробки та середньоарифметична сума балів СБ.

Таблиця 5.2 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

В таблиці 5.3 наведено результати оцінювання експертами комерційного потенціалу розробки.

Таблиця 5.3 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	Козачко О.М.	Крижановський Є.М.	Варчук І.В.
	Бали, виставлені експертами:		
1	4	1	2
2	3	1	1
3	3	5	4

Продовження таблиці 5.3

Критерії	Прізвище, ініціали, посада експерта		
	Козачко О.М.	Крижановський Є.М.	Варчук І.В.
	Бали, виставлені експертами:		
4	4	4	2
5	2	4	2
6	1	2	3
7	2	2	3
8	3	2	4
9	4	4	2
10	3	3	2
11	5	3	4
12	2	3	2
Сума балів	$СБ_1$=35	$СБ_2$=33	$СБ_3$=34
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{31 + 34 + 34}{3} = 34$		

Середньоарифметична сума балів, розрахована на основі висновків експертів склала 34 бали, що згідно таблиці 5.3 вважається, що рівень комерційного потенціалу проведених досліджень є вище середнього.

Інформаційної веб-система аналізу динаміки географічної структури зовнішньої торгівлі товарами України, а саме програма, яка допомагає проводити моніторинг, аналізувати та порівнювати дані. Дана програма буде цікава різним організаціям та компаніям в яких цільовий ринок направлений закордон.

5.2 Прогнозування витрат на виконання науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи групуються за такими статтями: витрати на оплату праці, витрати на соціальні заходи, матеріали, паливо та енергія для науково-виробничих цілей, витрати на службові відрядження, програмне забезпечення для наукових робіт, інші витрати, накладні витрати.

1. Основна заробітна плата кожного із дослідників $З_О$, якщо вони працюють в наукових установах бюджетної сфери визначається за формулою:

$$З_О = \frac{М}{Т_р} * t \text{ (грн)}, \quad (5.1)$$

де М – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.;

$Т_р$ – число робочих днів в місяці; приблизно $Т_р \approx 21...23$ дні;

t – число робочих днів роботи дослідника.

Для розробки програмні засоби необхідно залучити програміста з посадовим окладом 15000 грн. Кількість робочих днів у місяці складає 40, а кількість робочих днів програміста складає 22. Зведемо сумарні розрахунки до таблиця 5.4.

Таблиця 5.4 – Заробітна плата дослідника в науковій установі бюджетної сфери

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату грн.
Керівник	20000	909,1	5	4545
Програмний інженер	15000	681,8	40	27272
Всього				31817

2. Розрахунок додаткової заробітної плати робітників.

Додаткова заробітна плата $З_д$ всіх розробників та робітників, які приймали устають в розробці нового технічного рішення розраховується як 10 - 12 % від основної заробітної плати робітників.

На даному підприємстві додаткова заробітна плата начисляється в розмірі 10% від основної заробітної плати та розраховуються за формулою:

$$З_д = (З_о + З_р) * \frac{Н_{дод}}{100\%} \quad (5.2)$$

$$З_д = 0{,}1 * 31817 = 3181 \text{ (грн)}.$$

3. Нарахування на заробітну плату $Н_{ЗП}$ дослідників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою:

$$Н_{ЗП} = (З_о + З_д) * \frac{\beta}{100}, \quad (5.3)$$

де $З_о$ – основна заробітна плата розробників, грн;

$З_д$ – додаткова заробітна плата всіх розробників та робітників, грн;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, % .

Дана діяльність відноситься до бюджетної сфери, тому ставка єдиного внеску на загальнообов'язкове державне соціальне страхування буде складати 22%, тоді**:**

$$Н_{ЗП} = (31817 + 3181\ ) * \frac{22}{100} = 7700 \text{ (грн)}.$$

4. Витрати на комплектуючі вироби, які використовують при виготовленні одиниці продукції, розраховуються, згідно їх номенклатури, за формулою:

$$К = \sum_{i=1}^{n} Н_i * Ц_i * К_i\ , \quad (5.4)$$

де $Н_i$ – кількість комплектуючих *i*-го виду, шт.;

$Ц_i$ – покупна ціна комплектуючих *i*-го найменування, грн;

$К_i$ – коефіцієнт транспортних витрат (1,1…1,15).

В таблиці 5.5 наведено комплектуючі та їх вартість, що використані на розробку.

Таблиця 5.5 – Комплектуючі, що використані на розробку

Найменування матеріалу	Ціна за одиницю, грн.	Витрачено	Вартість витраченого матеріалу, грн.
Папір	150	1	150
Ручка	20	1	20
CD-диск	30	1	30
Флешка	150	1	150
Всього			350
З врахуванням коефіцієнта транспортування			400

5. Програмне забезпечення для наукової роботи включає витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення необхідного для проведення дослідження.

Для написання магістерської роботи використовувалися редактор VsCode та онлайн база даних MongoDB.

6. Амортизація обладнання, комп’ютерів та приміщень, які використовувались під час виконання даного етапу роботи

Дані відрахування розраховують по кожному виду обладнання, приміщенням тощо.

$$А = \frac{Ц * Т}{Т_{кор} * 12}, \quad (5.5)$$

де Ц – балансова вартість даного виду обладнання (приміщень), грн;

$Т_{кор}$ – час користування;

Т – термін використання обладнання (приміщень), цілі місяці.

Згідно пункту 137.3.3 Податкового кодексу амортизація нараховується на основні засоби вартістю понад 2500 грн. В нашому випадку для написання магістерської роботи використовувався персональний комп'ютер вартістю 30000 грн.

$$\text{А} = \frac{30000 \cdot 1}{2 \cdot 12} = 1150 \text{ (грн)}.$$

7. До статті «Паливо та енергія для науково-виробничих цілей» відносяться витрати на всі види палива й енергії, що безпосередньо використовуються з технологічною метою на проведення досліджень.

$$\text{В}_\text{е} = \sum_{\text{i}=1}^{n} \frac{W_{yt} \cdot t_i \cdot \text{Ц}_\text{е} \cdot \text{К}_{\text{впі}}}{\eta_\text{i}}, \qquad (5.6)$$

де W_{yt} – встановлена потужність обладнання на певному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$\text{Ц}_\text{е}$ – вартість 1 кВт-години електроенергії, грн;

$\text{К}_{\text{впі}}$ – коефіцієнт, що враховує використання потужності, $\text{К}_{\text{впі}} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_\text{i} < 1$.

Для написання магістерської роботи використовується персональний комп'ютер для якого розрахуємо витрати на електроенергію.

$$\text{В}_\text{е} = \frac{0{,}3 \cdot 200 \cdot 1{,}68 \cdot 0{,}5}{0{,}8} = 63 \text{ (грн)}.$$

Витрати на службові відрядження, витрати на роботи, які виконують сторонні підприємства, установи, організації та інші витрати в нашому досліджені не враховуються оскільки їх не було.

Накладні (загальновиробничі) витрати Внзв охоплюють: витрати на управління організацією, оплата службових відряджень, витрати на утримання, ремонт та експлуатацію основних засобів, витрати на опалення,

освітлення, водопостачання, охорону праці тощо. Накладні (загальновиробничі) витрати Внзв можна прийняти як (100…150)% від суми основної заробітної плати розробників та робітників, які виконували дану МКНР, тобто:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (5.7)$$

де $Н_{нзв}$ – норма нарахування за статтею «Інші витрати».

$$В_{нзв} = 31817 \cdot \frac{100}{100\%} = 31817(грн).$$

Сума всіх попередніх статей витрат дає витрати, які безпосередньо стосуються даного розділу МКНР

$$В = 31817+3181+7700+400+1150+63+31817=76128\ (грн).$$

Прогнозування загальних втрат ЗВ на виконання та впровадження результатів виконаної МКНР здійснюється за формулою:

$$ЗВ = \frac{В}{\eta}, \quad (5.8)$$

де η – коефіцієнт, який характеризує стадію виконання даної НДР.

Оскільки, робота знаходиться на стадії науково-дослідних робіт, то коефіцієнт β = 0,9.

Звідси:

$$ЗВ = \frac{76128}{0,9} = 84586,7(грн).$$

5.3 Розрахунок економічної ефективності науково-технічної розробки

У даному підрозділі кількісно спрогнозуємо, яку вигоду, зиск можна отримати у майбутньому від впровадження результатів виконаної наукової роботи. Розрахуємо збільшення чистого прибутку підприємства $\Delta П_i$, для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, за формулою

$$\Delta П_i = \sum_1^n (\Delta Ц_o * N * Ц_o * \Delta N)_i * \lambda * \rho * \left(1 - \frac{\upsilon}{100}\right), \quad (5.9)$$

де $\Delta Ц_0$ – покращення основного оціночного показника від впровадження результатів розробки у даному році.

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки:

$Ц_o$ – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки:

λ – коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт λ = 0,8333.

ρ – коефіцієнт, який враховує рентабельність продукту. ρ = 0,25;

υ – ставка податку на прибуток. У 2022 році – 18%.

Припустимо, що ціна за програмний продукт зросте на 600 грн. Кількість одиниць реалізованої продукції також збільшиться: протягом першого року на 45 шт., протягом другого року – на 35 шт., протягом третього року на 25 шт. Реалізація продукції до впровадження розробки складала 1 шт., а її ціна до складає 10000 грн. Розрахуємо прибуток, яке отримає підприємство протягом трьох років.

$$\Delta П_1 = [600 \cdot 1 + (10000 + 600) \cdot 45] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 117815.59 \text{ (грн)}.$$

$$\Delta П_2 = [600 \cdot 1 + (10000 + 600) \cdot (45 + 35)] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 208983.28 \text{ (грн)}.$$

$$\Delta П_3 = [600 \cdot 1 + (10000 + 600) \cdot (45 + 35 + 25)] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 274103.05 \text{(грн)}.$$

5.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Розрахуємо основні показники, які визначають доцільність фінансування наукової розробки певним інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності.

Розрахуємо величину початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки.

$$PV = k_{\text{інв}} \cdot \text{ЗВ}, \qquad (5.10)$$

де $k_{\text{інв}}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо ($k_{\text{інв}}$ = 2…5).

$$PV = 2 \cdot 84586{,}7 = 169173.4(\text{грн}).$$

Розрахуємо абсолютну ефективність вкладених інвестицій $E_{абс}$ згідно наступної формули:

$$\text{Е}_{\text{абс}} = (\text{ПП} - PV), \tag{5.11}$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки, грн.;

$$\text{ПП} = \sum_1^T \frac{\Delta\text{П}_\text{і}}{(1+\tau)^t}\ , \tag{5.12}$$

де $\Delta\text{П}_\text{і}$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДЦКР, грн.;

Т – період часу, протягом якою виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,2;

t – період часу (в роках).

$$\text{ПП} = \frac{117815.59}{(1+0{,}2)^1} + \frac{208983.28}{(1+0{,}2)^2} + \frac{274103.05}{(1+0{,}2)^3} = 401931.37\ (\text{грн}).$$

$$\text{Е}_{\text{абс}} = (401931.37 - 169173.4) = 232757.97\ (\text{грн}).$$

Оскільки $Е_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР може бути доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $Е_в$. Для цього користуються формулою:

$$Е_в = \sqrt[Т_ж]{1 + \frac{Е_{абс}}{PV}} - 1 , \quad (5.13)$$

де $Т_ж$ – життєвий цикл наукової розробки, роки.

$$Е_в = \sqrt[3]{1 + \frac{232757.97}{169173.4}} - 1 = 0{,}26 = 26\%.$$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau = d + f , \quad (5.14)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2022 році в Україні d = (0,14…0,2);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина $f = (0{,}05…0{,}1)$.

$$\tau_{min} = 0{,}16 + 0{,}05 = 0{,}21.$$

Так як $E_в > \tau_{min}$ то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$T_{ок} = \frac{1}{E_в}. \quad (5.15)$$

$$T_{ок} = \frac{1}{0,26} = 3,8 \text{ (роки)}.$$

Так як $T_{ок} \leq 3...5$-ти років, то фінансування даної наукової розробки в принципі є доцільним.

5.5 Висновки

Проведено оцінку комерційного потенціалу інформаційної веб-системи аналізу динаміки географічної структури зовнішньої торгівлі товарами України, а саме програми, яка аналізує дані, проводить моніторинг та будує комплексні порівняльні графіки статистики закордонного ринку України.

Прогнозування витрат на виконання науково-дослідної роботи по кожній з статей витрат складе 76128 грн. Загальна ж величина витрат на виконання та впровадження результатів даної НДР буде складати 84586,7 грн.

Вкладені інвестиції в даний проект окупляться через 3,8 роки, приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки склала 401931,37 грн.

ВИСНОВКИ

В ході магістерської кваліфікаційної роботи проведено аналіз предметної області, визначено суть поставленої задачі, проведено огляд існуючих методів вирішення технічної проблеми. В результаті чого сформовано низку програмних вимог та концепція майбутнього веб-сервісу. Також було визначено джерело даних, проведено збір та систематизацію даних на основі яких було розроблено веб-систему аналізу закордонного ринку України.

Проаналізовано суть технічної проблеми і також здійснено огляд існуючих методів її вирішення.

Сформовано базову концепцію продукту та сформовані базові програмні вимоги.

Також були проаналізовані передові технології, які б вирішення поставленої задачі. Тому найбільш оптимальним рішенням обрано стек технологій MERN (Mongo Express React Node). Дані технології чітко поєднуються між собою, створюючи потужний та швидкий веб-додаток, який забезпечує користувача всіма необхідними ресурсами.

Проведено статистичне збирання даних та створено базу даних, яка забезпечить інформаційну веб-систему необхідною інформацією.

Здійснено дослідження на рахунок джерела даних, та як насправді їх потрібно правильно обробити та зберегти у потрібному форматі для зручної роботи з ними.

Розроблено архітектуру програмного забезпечення, де було описано всі найважливіші аспекти розробки веб-застосунку. Після чого, було створено серверну та клієнтську частини інформаційної веб-системи аналізу динаміки географічної структури зовнішньої торгівлі товарами України.

В результаті розробленої функціональності веб системи, користувач має змогу провести покращений комплексний аналіз структури зовнішньої торгівлі товарами України в розрізі областей.

Проведено статистичний аналіз закордонного ринку та виявлено, що Україна має досить непогану тенденцію до розвитку міжнародних торгово-економічних відносин, має досить багато напрямків до збуту та закупівлі товарів, та що з року в рік товарообіг все більше набирає обертів.

Прогнозування витрат на виконання науково-дослідної роботи по кожній з статей витрат складе 76128 грн. Загальна ж величина витрат на виконання та впровадження результатів даної НДР буде складати 84586,7 грн.

Вкладені інвестиції в даний проект окупляться через 3,8 роки, приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки склала 401931,37 грн.

За результатами даної роботи, були написані тези доповіді, які в підсумку, були опубліковані в збірнику матеріалів Всеукраїнської науково-практичної інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи».

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Крижановський Є. М., Пасєка Б. В Інформаційна веб-система аналізу динаміки географічної структури зовнішньої торгівлі товарами України. *Всеукраїнська науково-практична інтернет-конференція «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2022-2023 рр.)*. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2023/paper/viewFile/16804/14038.

2. Прокушев Е.Ф. Зовнішньоекономічна діяльність : підручник. Тернопіль : Астон, 2013. 360 с.

3. Гетьман О., Шаповал В. Економіка підприємства. Харків : Центр учб. літ., 2021. 488 с.

4. Дідківський М. У. Зовнішньоекономічна діяльність підприємства. Одеса : Знання, 2019. 463 с.

5. Дахно І. І. Менеджмент зовнішньоекономічної діяльності. URL: http://p-for.com/book_212.

6. Шкурупій О. В. Зовнішньоекономічна діяльність підприємств. Полтава : Центр учб. літ., 2015. 248 с.

7. Веб-додаток Intervals. URL: https://www.myintervals.com/.

8. Веб-додаток Projector PSA. URL: https://www.projectorpsa.com/.

9. Веб-додаток Workfront. URL: https://www.workfront.com/.

10. Веб-додаток Attest. URL: https://www.askattest.com/.

11. Веб-додаток Statista. URL: https://www.statista.com/.

12. Subramanian H., Raj P. Hands-On RESTful API Design Patterns and Best Practices: Design, develop, and deploy highly adaptable, scalable, and secure RESTful web APIs. Packt Publishing, 2019. 378 p.

13. The Definitive Guide to MongoDB: A complete guide to dealing with Big Data using MongoDB / T. Hawkins et al. Apress, 2019. 370 p.

14. MERN Stack. URL: https://www.geeksforgeeks.org/mern-stack/

15. NodeJS. URL: https://nodejs.org/uk/

16. ExpressJS. URL: https://expressjs.com.

17. ReactJS. URL: https://uk.reactjs.org/.

18. Banks A., Porcello E. Learning React: Functional Web Development with React and Redux. O'Reilly Media, 2018. 350 p..

19. Bootstrap. URL: https://getbootstrap.com/.

20. SASS. URL: https://sass-scss.ru/guide/.

21. Microsoft Excel. URL: https://uk.wikipedia.org/wiki/Microsoft_Excel.

22. MongoDB. URL: https://www.mongodb.com/1.

23. NPM. URL: https://medium.com/webbdev/npm-daa12a10caac

24. Mongoose URL: https://mongoosejs.com/c

25. React-Router. URL: https://medium.com/the-andela-way/understanding-the-fundamentals-of-routing-in-react-b29f806b157e

26. Козловський В. О., Лесько О. Й., Кавецький В. В. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт : уклад. Вінниця : ВНТУ, 2021. 42 с.

Додаток А

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
_____ д.т.н., проф. Мокін В. Б.
«_19_» _____09_____ 2022 р.

ТЕХНІЧНЕ ЗАВДАННЯ
на магістерську кваліфікаційну роботу
«ІНФОРМАЦІЙНА ВЕБ-СИСТЕМА АНАЛІЗУ ДИНАМІКИ ГЕОГРАФІЧНОЇ СТРУКТУРИ ЗОВНІШНЬОЇ ТОРГІВЛІ ТОВАРАМИ УКРАЇНИ»
08-53.МКР.002.02.000.ТЗ

Керівник: к.т.н., доц. каф. САІТ
_____ Крижановський Є. М.
«_19_» _____09_____ 2022 р.

Розробив: студент гр. 2ІСТ-21м
_____ Пасєка Б.В.
«_19_» _____09_____ 2022 р.

Вінниця 2022

1. Підстава для проведення робіт

Підставою для виконання роботи є наказ № 203 по ВНТУ від «14» _09_ 2022 р., та індивідуальне завдання на МКР, затверджене протоколом № 3 засідання кафедри САІТ від «14» ____09_____ 2022 р.

2. Джерела розробки:

- Дідківський М. У. Зовнішньоекономічна діяльність підприємства. Одеса : Знання, 2019. 463 с.
- Banks A., Porcello E. Learning React: Functional Web Development with React and Redux. O'Reilly Media, 2018. 350 p..
- MERN Stack. URL: https://www.geeksforgeeks.org/mern-stack/

3. Мета і призначення роботи:

Покращення комплексності аналізу структури зовнішньої торгівлі країни в розрізі областей шляхом розроблення інформаційної веб-системи аналізу динаміки географічної структури зовнішньої торгівлі товарами України.

4.Вихідні дані для проведення робіт:

Дані управління статистики по кожній з областей України.

5.Методи дослідження:

– статистичний аналіз даних;
– методи баз даних та ГІС-технологій;
– методи веб-технологій.

6. Етапи роботи і терміни їх виконання:

1. Аналіз зовнішньої торгівлі товарами ..... 20.09 – 30.09
2. Формування програмних вимог та побудова концепції продукту ..... 01.10 – 10.10
3. Збирання та систематизація даних ..... 11.10 – 20.10
4. Розроблення інформаційної веб-системи аналізу зовнішньої торгівлі товарами України ..... 21.10 – 31.10
5. Економічна частина. ..... 01.11 – 10.11
6. Оформлення матеріалів до захисту МКР. ..... 11.11 – 30.11

7. Очікувані результати та порядок реалізації:

Розроблення інформаційної веб-системи аналізу динаміки географічної структури зовнішньої торгівлі товарами України.

8. Вимоги до розробленої документації

Пояснювальна записка оформлена у відповідності до вимог «Методичних вказівок до виконання та оформлення магістерських кваліфікаційних робіт для студентів спеціальності 126 – «Інформаційні системи та технології» денної форми навчання».

9. Порядок приймання роботи

Публічний захист ..... «_20_» _____12_____ 2022 р.
Початок розробки ..... «_20_» _____09_____ 2022 р.
Граничні терміни виконання МКР ..... «_30_» _____11_____ 2022 р.

Розробив студент групи 2ІСТ-21м _____ Пасєка Б.В.

Додаток Б

Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень

Назва роботи: «Інформаційна веб-система аналізу динаміки географічної структури зовнішньої торгівлі товарами України»

Тип роботи: магістерська кваліфікаційна робота

Підрозділ: кафедра САІТ

Науковий керівник: Крижановський Є. М. к.т.н., доц. каф. САІТ

Показники звіту подібності Unicheck

Оригінальність	98 %
Схожість	2 %

Аналіз звіту подібності (відмітити потрібне):

☑ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і самостійності її автора. Роботу направити на розгляд експертної комісії кафедри.

☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Опис прийнятого рішення:

Робота допускається до захисту

Особа, відповідальна за перевірку Жуков С. О.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи Пасєка Б. В.

Керівник роботи Крижановський Є. М.

Додаток В

Лістинг програми

```
const express = require('express')
const mongoose = require('mongoose')
const router = require('./routers/route.js')
const path = require('path')

const app = express()
const PORT  = process.env.PORT || 8080
app.use(express.json({extended: true}))

app.use('/', router)

if (process.env.NODE_ENV === 'production') {
    app.use(express.static(path.join(__dirname, 'client/build')))

    app.get('*', (req, res) => {
        res.sendFile(path.join(__dirname, 'client/build', 'index.html'))
    })
}

const start = async() => {
    await mongoose.connect('mongodb+srv://bohdan:1234@cluster0.cvest.mongodb.net/
Regions?retryWrites=true&w=majority', {
        useNewUrlParser: true,
        useUnifiedTopology: true,
        useFindAndModify: false,
        useCreateIndex: true
    })
    app.listen(PORT, () => {console.log(`Server has been launched at port ${PORT}
`)})
}
start()

const {Router} = require('express')
const Region = require('../models/model')
const {getAreaByName, getAll, getAreaForTable, patchDataFromTable} = require('../
controllers/controler.js')
const router = Router()

router.post('/map', async (req, res) => {
    const {direction} = req.query
    const {id, selectedValues} = req.body
    const region = await Region.findOne({map_id: id}, err => {
        if (err) {
            res.status(404).json('Not found')
        }
```

```
    })

    const obj = region[direction].filter(item => {
        let reg
        selectedValues.forEach(year => {
          if (+year == new Date(item.date).getFullYear()) {
            reg = item
          }
        })
        return reg
    })

    res.status(200).json({obj})
})


router.post('/region/info', getAreaByName)
router.get('/regions/name', getAll)
router.post('/region/table', getAreaForTable)
router.patch('/region/table', patchDataFromTable)

module.exports = router
const {Schema, model} = require('mongoose')

const region = new Schema({
    area: String,
    map_id: String,
    import: [{
      date: Date,
      all: String,
      cis: String,
      other_countries: String,
      europe: String,
      eu_countries: String,
      asia: String,
      africa: String,
      usa: String,
      oceania: String
      }],
      export: [{
        date: Date,
        all: String,
        cis: String,
        other_countries: String,
        europe: String,
        eu_countries: String,
        asia: String,
        africa: String,
        usa: String,
        oceania: String
        }]
```

```
})

module.exports = model('Region', region, "regions")
const Region = require('../models/model.js')

const getAreaByName = async (req, res) => {
    const {area} = req.body
    const {direction, direction_market} = req.query
    const region = await Region.findOne({area})
    const years = region[direction].map(year => new Date(year.date).getFullYear()
)
    const alls = region[direction].map(all => Math.ceil(all[direction_market]))
    res.json({years, alls})
}

const getAll = async (req, res) => {
    const data = await Region.find({})
    const names = data.map(item => item.area)
    res.json(names)
}

const getAreaForTable = async (req, res) => {
    const {area, targetMarket} = req.body
    const region = await Region.findOne({area})
    const direction = region[targetMarket]
    res.json({direction})
}

const patchDataFromTable = async (req, res) => {
    const {
        region,
        targetMarket,
        itemDate,
        itemAll,
        itemCis,
        itemOtherCountries,
        itemEurope,
        itemEUcountries,
        itemAsia,
        itemAfrica,
        itemUsa,
        itemOceania
    } = req.body

   const area = await Region.findOne({area:region})
   const target = area[targetMarket].find(item => new Date(item.date).getFullYear
() == new Date(itemDate).getFullYear())
   target.date = itemDate
   target.all = itemAll
   target.cis = itemCis
```

```
   target.other_countries = itemOtherCountries
   target.europe = itemEurope
   target.eu_countries = itemEUcountries
   target.asia = itemAsia
   target.africa = itemAfrica
   target.usa = itemUsa
   target.oceania = itemOceania

   await area.save()

   res.json({message: 'Region data was updated'})
}

module.exports = {getAreaByName, getAll, getAreaForTable, patchDataFromTable}
import React from 'react'
import {Route, Switch, BrowserRouter as Router} from 'react-router-dom'
import Navigation from '../Navigation/Navigation'
import $ from 'jquery';
import Map from '../Map/Map'
import Overview from '../Overview/Overview'
import Statistics from '../Statistics/Statistics'
import './App.scss';

function App() {
  return (
    <Router>
       <Navigation/>
          <Switch>
                  <Route exact path="/" component={Overview} />
                  <Route path="/map" component={Map} />
                  <Route path="/statistics" component={Statistics} />
          </Switch>
      </Router>
  )
}

export default App

import React from 'react'
import 'bootstrap/dist/css/bootstrap.css';
import 'bootstrap/dist/js/bootstrap.js';
import './Navigation.scss'
import {Link} from 'react-router-dom'


const Navigation = () => {
    return (
        <>
        <ul className="nav nav-pills flex-column links">
            <li className="nav-item ">
                <Link className="nav-link" to="/">Overview and Edit</Link>
```

```
            </li>
            <li className="nav-item">
                <Link className="nav-link" to="/map">Map</Link>
            </li>
            <li className="nav-item">
                <Link className="nav-
link" to="/statistics">Statistics Graph</Link>
            </li>
        </ul>
        </>
    )
}

export default Navigation
import React, {Component} from 'react';
import { VectorMap } from '@south-paw/react-vector-maps';
import {Pie} from 'react-chartjs-2'
import {Multiselect} from 'multiselect-react-dropdown'
import './Map.scss'
import Ukraine from './mymap.json';

const years = ['1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2
004', '2005','2006', '2007', '2008','2009', '2010', '2011', '2012', '2013', '2014
', '2015','2016', '2017', '2018', '2019']
class Map extends Component {
    constructor(props) {
        super(props)
        this.state = {
            direction: 'export',
            data: {},
            isPie: false,
            top: 0,
            left: 0,
            selectedValues: ['1996']
        }

        this.style = {
            chips: {

            },
            searchBox: {
              width: '300px',
              position:'absolute',
              top: '-78px',
              border: '1px solid #000000',
              maxHeight: '100px',
              overflowY: 'scroll',

            },
            multiselectContainer: {
```

```
                width: '300px',
                position:'absolute',
                left: '255px'
            }
          };

        this.handllerDirection = this.handllerDirection.bind(this);
        this.selectHandler = this.selectHandler.bind(this);
        this.removeHandler = this.removeHandler.bind(this);
    }

    handllerDirection(e) {
        this.setState({direction: e.target.outerText.toLowerCase(), isPie: false}
)
    }

     async getRegion (id) {
        const notFoundId = ["ua-43", "ua-30", "ua-40"]
        if (notFoundId.includes(id)) {
            this.setState({isPie: false})
            return
        }
        try {
            const res = await fetch(`/map?direction=${this.state.direction}`,
            {
                method: 'POST',
                body: JSON.stringify({id, selectedValues: this.state.selectedValu
es}),
                headers: {
                    'Content-type': 'application/json; charset=UTF-8',
                }
            })
            const region = await res.json()
            const arr =  region.obj.map(item => {
                delete item.date
                delete item.all
                let labels = Object.values(item).map(el => +el)
                return labels
            })

            const result = arr.reduce((r, a) => a.map((b, i) => (r[i] || 0) + b),
 [])
```

Додаток Г

ІЛЮСТРАТИВНА ЧАСТИНА

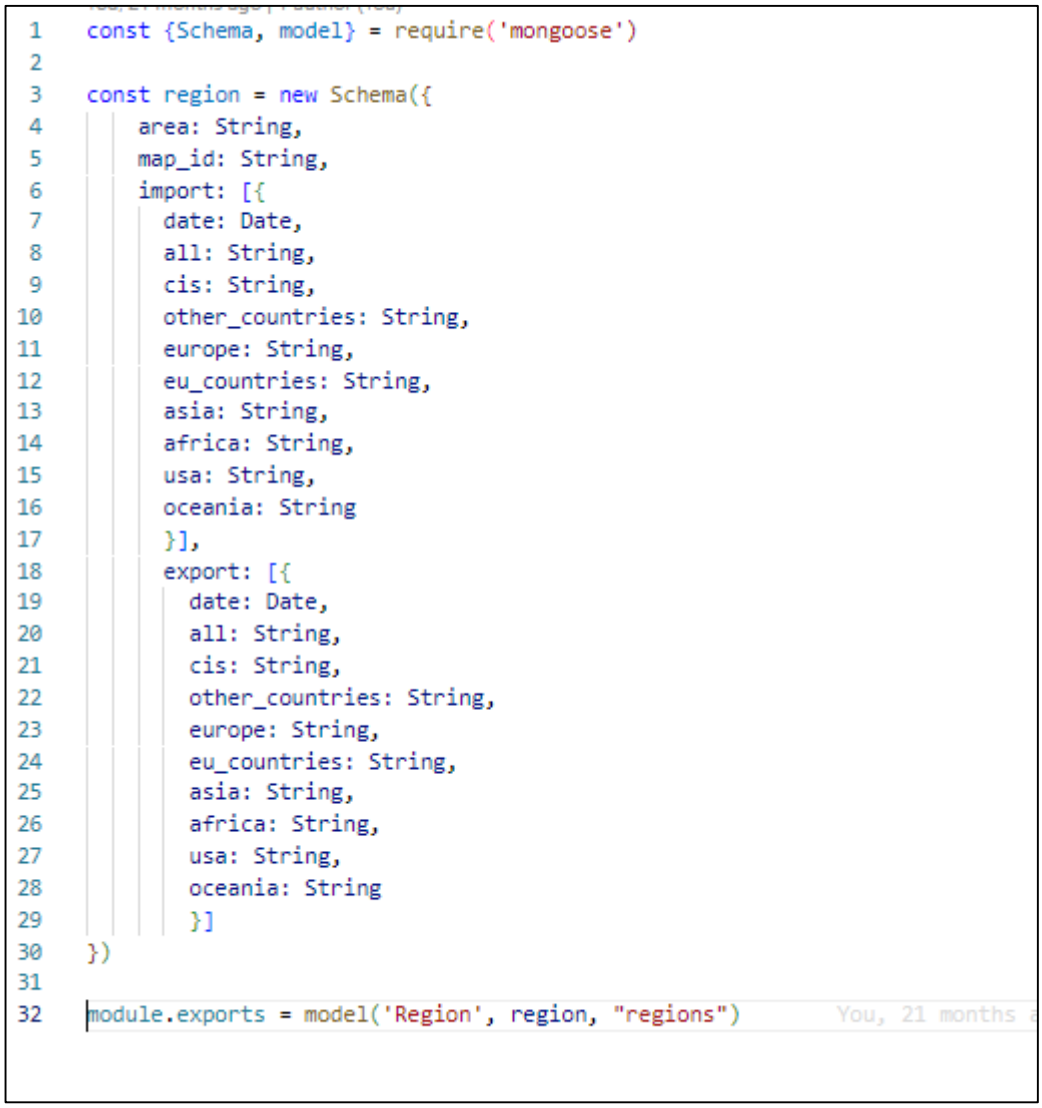
ІНФОРМАЦІЙНА ВЕБ-СИСТЕМА АНАЛІЗУ ДИНАМІКИ ГЕОГРАФІЧНОЇ СТРУКТУРИ ЗОВНІШНЬОЇ ТОРГІВЛІ ТОВАРАМИ УКРАЇНИ

Виконав: студент гр. 2ІСТ-21м
_____ Пасєка Б. В.
«_01_» _____12_____ 2022 р.

Керівник: к.т.н., доц. каф. САІТ
_____ Крижановський Є. М.

«_02_» _____12_____ 2022 р.

Нормоконтроль: к.т.н., доцент
_____ Жуков С. О.
«_02_» _____12_____ 2022 р.



```
const {Schema, model} = require('mongoose')

const region = new Schema({
    area: String,
    map_id: String,
    import: [{
      date: Date,
      all: String,
      cis: String,
      other_countries: String,
      europe: String,
      eu_countries: String,
      asia: String,
      africa: String,
      usa: String,
      oceania: String
      }],
      export: [{
        date: Date,
        all: String,
        cis: String,
        other_countries: String,
        europe: String,
        eu_countries: String,
        asia: String,
        africa: String,
        usa: String,
        oceania: String
        }]
})

module.exports = model('Region', region, "regions")
```

Рисунок Г.1 – Схема обробки даних MongoDB

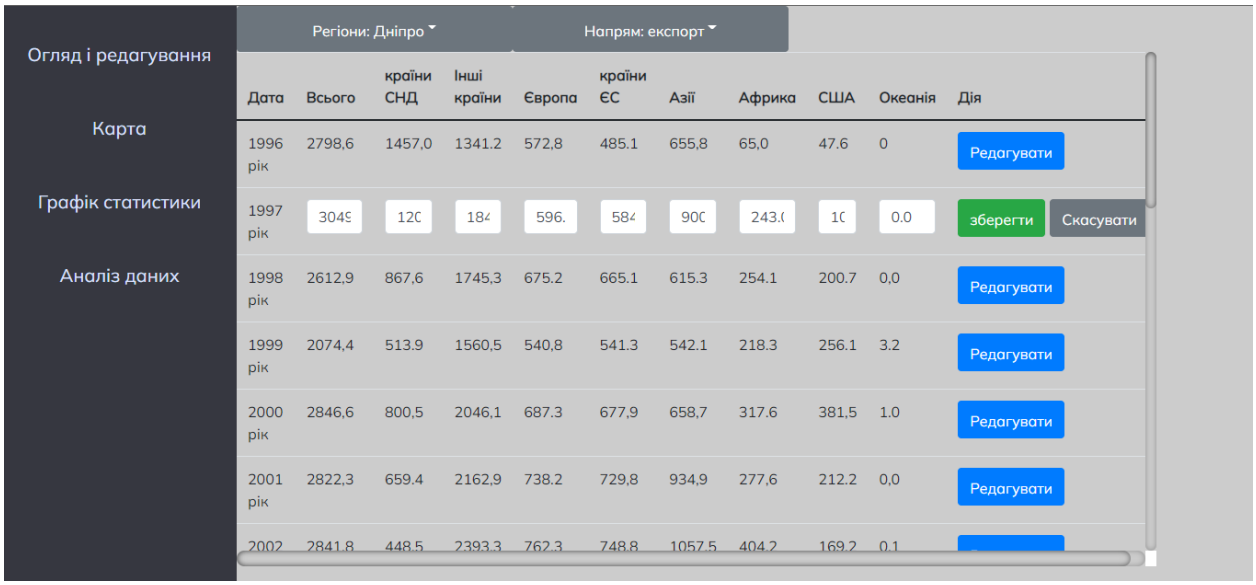


Рисунок Г.2 – Огляд та редагування інформації в базі даних

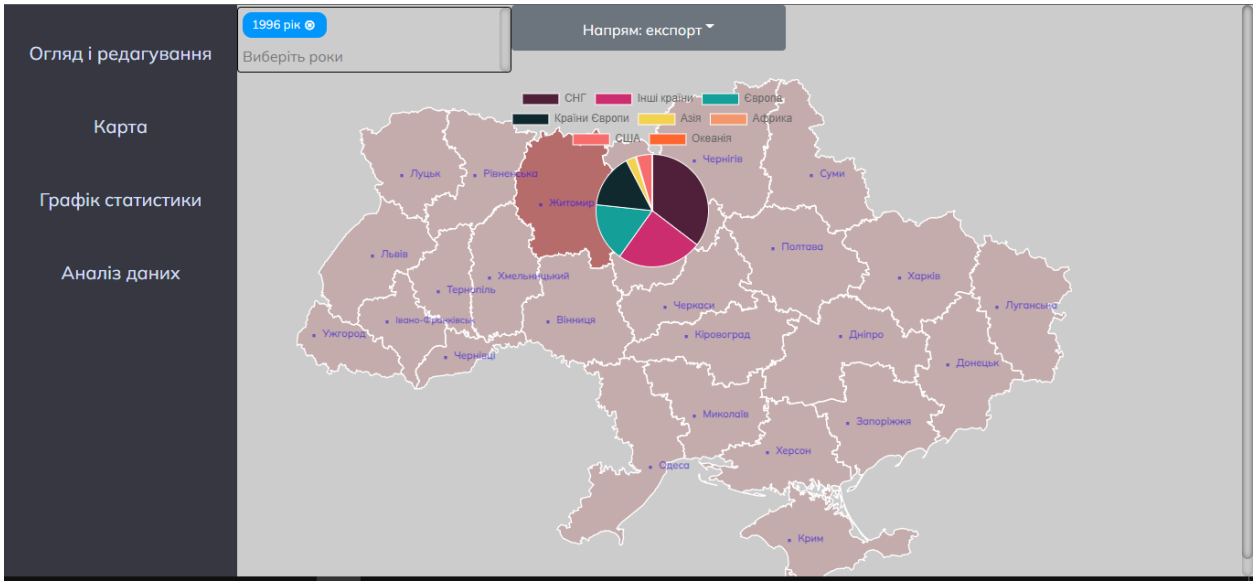


Рисунок Г.3 – Робота з картою та побудова статистичних діаграм

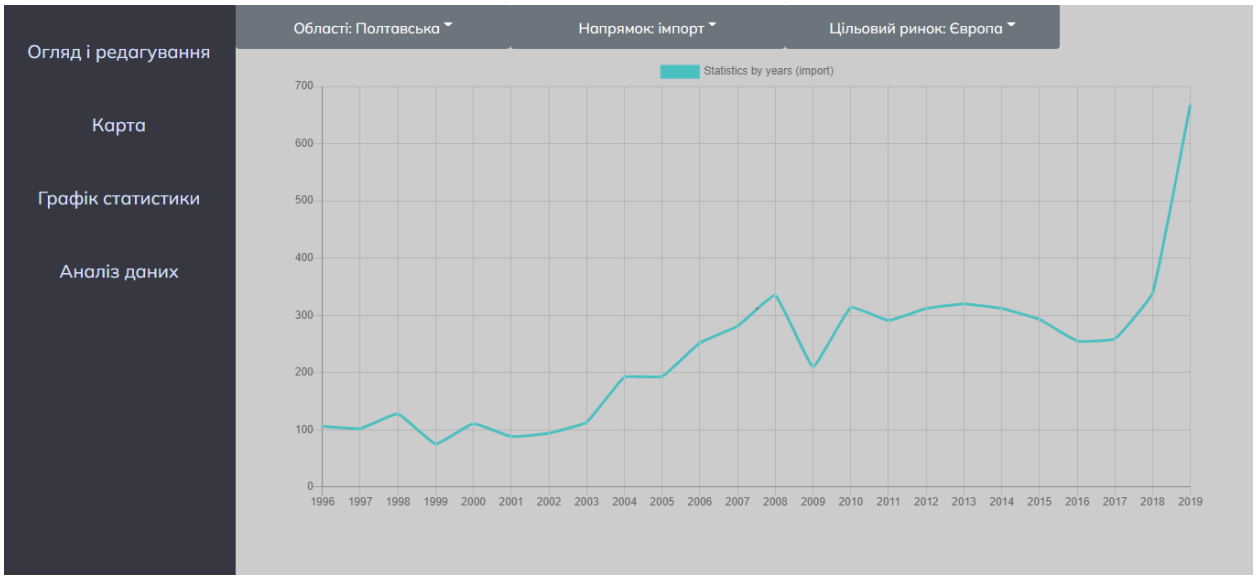


Рисунок Г.4 – Побудова графіків статистики

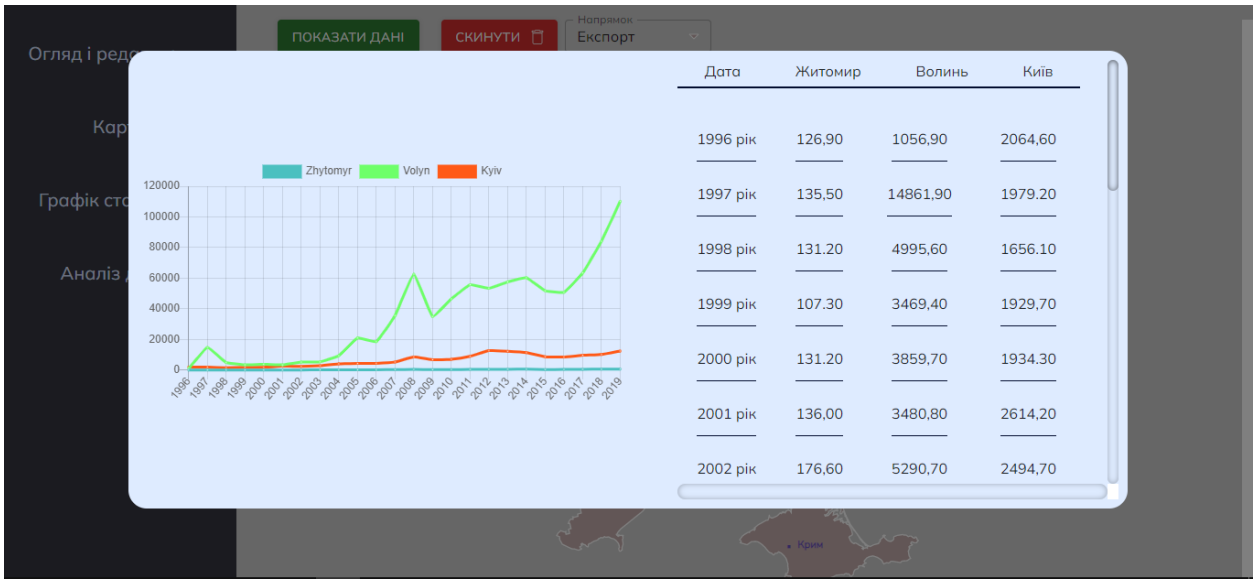


Рисунок Г.5 – Побудова комплексних статистичних графіків

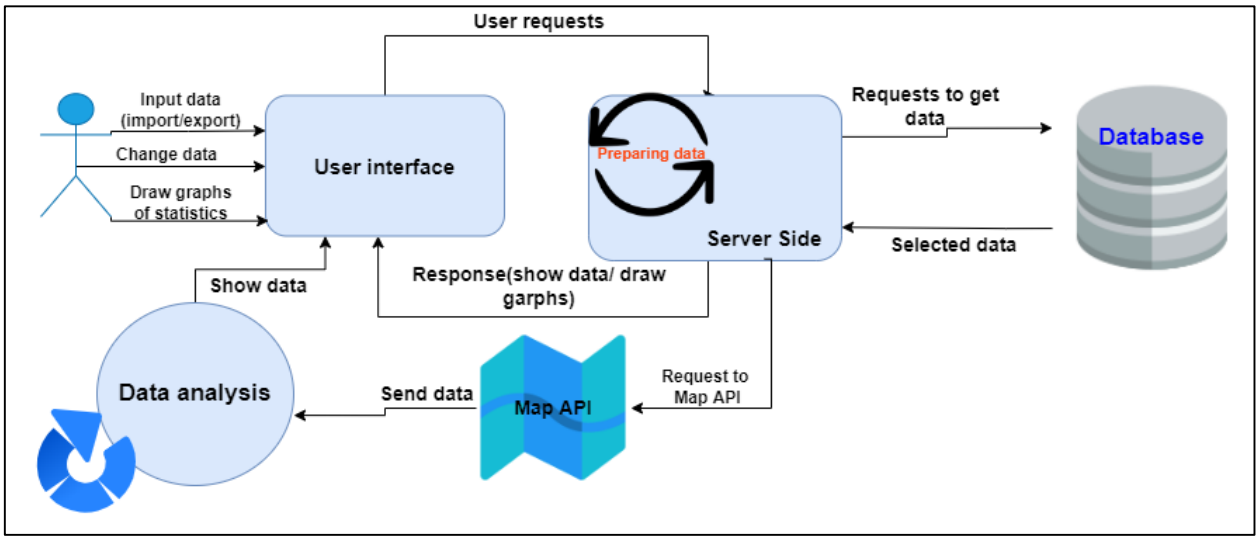


Рисунок Г.6 – Модель роботи веб-сервісу