Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально-науковий інститут деревообробних та комп'ютерних технологій і дизайну
(повне найменування інституту, назва факультету (відділення))

Кафедра інформаційних технологій
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до дипломної роботи

другий (магістерський)
(рівень вищої освіти)

на тему: Інформаційна технологія аналізу енергетичного протенціалу відновлювальних джерел енергії у виділених регіонах

Виконав: студент 6 курсу групи КН-61М
спеціальності
122 "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)
Колиняк Д.З.
(прізвище та ініціали)
Керівник Шабатура Ю. В.
(прізвище та ініціали)

Рецензент
(прізвище та ініціали)

Львів – 2021

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ деревообробних та комп'ютерних технологій і дизайну
Кафедра інформаційних технологій
Рівень вищої освіти другий (магістерський)
Спеціальність 122 "Комп'ютерні науки"
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри Крошний І. М.

"____"_____20___ року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Колиняк Денис Зіновійович
(прізвище, ім'я, по батькові)

1. Тема роботи *Інформаційна технологія аналізу енергетичного протенціалу відновлювальних джерел енергії у виділених регіонах*
керівник роботи *Шабатура Юрій Васильович, доктор технічних наук, професор*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "*31*" *грудня* 20*21* року № С-593
2. Термін подання студентом роботи
3. Вихідні дані до роботи *Аналіз шляхів вирішення поставленої задачі та розробка її на основі програмного та математичного забезпечення*
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Вступ.
Розділ 1. Опис та системний аналіз предметної області.
Розділ 2. Інформаційне та математичне забезпечення.
Розділ 3. Програмне і технічне забезпечення.
Висновки.
Перелік посилань.
Додатки.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Слайди доповіді, актуальність теми, постановка завдання, скріншоти застосунку, опис функціональності, висновки

6. Дата видачі завдання *18 грудня 2020*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Огляд літературних джерел та інтернет ресурсів, пошук загальної інформації за вказаною темою	04.02.2021 20.02.2021	*виконано*
2	Вибір технологій, підходів і фреймворків для поставленої задачі	01.03.2021 23.03.2021	*виконано*
3	Проектування програмного забезпечення, налаштування середовища	24.03.2021 09.04.2021	*виконано*
4	Розроблення фунціональності застосунку	11.04.2021 20.06.2021	*виконано*
5	Створення користувацького інтерфейсу	24.06.2021 30.07.2021	*виконано*
6	Тестування програми та аналіз результатів	02.08.2021 25.08.2021	*виконано*
7	Оформлення записки до дипломного проекту.	25.10.2021 29.11.2021	*виконано*
8	Здача пояснювальної записки на рецензування.	01.12.2021 10.12.2021	*виконано*
9	Створення презентації та підготовка доповіді.	11.12.2021 20.12.2021	*виконано*

Студент _____ Колиняк Д.З.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Шабатура Ю.В.
(підпис) (прізвище та ініціали)

ТЕХНІЧНЕ ЗАВДАННЯ

Необхідно розробити інформаційна систему для аналізу енергетичного протенціалу відновлювальних джерел енергії:

- знайти дані щодо енергетичного протенціалу відновлювальних джерел енергії;
- структуризувати дані по областях і видах джерел;
- Розробити систему для аналізу даних зі зручним користувацьким інтерфейсом;
- проаналізувати отримані результати та підбити висновки.

РЕФЕРАТ

Дипломна робота містить 58 сторінки пояснювальної записки, 8 рисунків, 1 додаток, 10 джерел.

В даній роботі була розроблена інформаційна технологія аналізу енергетичного протенціалу відновлювальних джерел енергії у виділених регіонах, зі зручним користувацьким інтерфейсом для зручного аналізу даних.

Програмне забезпечення реалізовано за допомогою JavaScript, Node.JS, MySQL, Angular, HTML 5, CSS.

Ключові слова: Javascript, Node.JS, MySQL, SQL, Sequelize, Angular, React, відновлювальні джерела, енергетичний потенціал.

ABSTRACT

The thesis contains 58 pages of explanatory note, 8 figures, 1 appendix, 10 used literary sources.

In this work, information technology for the analysis of the energy potential of renewable energy sources in selected regions has been developed, with a user-friendly interface for easy data analysis.

The software is implemented using JavaScript, Node.JS, MySQL, Angular, HTML 5, CSS.

Keywords: Javascript, Node.JS, MySQL, SQL, Sequelize, Angular, React, renewable sources, energy potential.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

БД	База даних
СУБД	Система управління базами даних
ВДЕ	Відновлювальні джерела енергії
ЕП	Енергетичний потенціал
JS	Мова програмування Javascript
API	Прикладний програмний інтерфейс

ВСТУП

Технології відновлюваної енергії знаходяться на порозі нової ери. У багатьох країнах і регіонах відновлювані джерела енергії вже відповідають за задоволення значної частки попиту на енергію. Швидкий і значний прогрес відновлюваної енергетики в останні роки був зумовлений політикою місцевих, національних та регіональних органів влади у тісній співпраці з бізнес-спільнотою, а також постійними технологічними інноваціями та зниженням витрат на енергію, вироблену з відновлюваних джерел.

Прогрес у створенні нових енергетичних систем уже є значним. Але такі питання, як енергетична незалежність, викорінення енергетичної бідності, боротьба зі зміною клімату та покращення кризової стійкості енергетичних систем, вимагають прискорення розгортання відновлюваних джерел енергії. Нещодавні події, які мали великий вплив на суспільства в усьому світі, наприклад, фінансова криза, ядерна аварія на Фукусімі, великі розливи нафти, нові відкриття в галузі науки про зміну клімату, — ще більше підкреслили цю потребу.

Визначено, що потужність відновлюваної енергії в кілька разів перевищує поточне споживання електроенергії. Проте можливість використання цього потенціалу обмежено можливостями наявних енергетичних систем.

Інтенсивний розвиток потужностей вітрової та сонячної електроенергії потребує додаткових резервів для компенсації дисбалансу в графіку роботи.

Існуючі рішення в основному базуються на використанні станцій зберігання, але останнім часом водневі технології стали конкурентоспроможною альтернативою. Сховище стисненого водню має тривалий час зберігання і мегаватну потужність, тому є економічно привабливим. Водень стає все більш важливою темою енергетики, оскільки він може бути важливим для майбутньої «чистої» енергії.

Однак існують також деякі тенденції, які маскують можливість прискорення зростання відновлюваної енергії. Наприклад, перспективні — короткострокові — вигоди від дослідження нових викопних джерел, таких як нафта в арктичних регіонах або сланцевий газ, мобілізують потужні сили, які, як правило, нехтують довгостроковими недоліками. Між тим, під час дебатів реальні витрати на відновлювані джерела енергії часто хибно уявляють або невірно сприймають, а реальність інтеграції децентралізованих відновлюваних джерел в мережі часто неправильно сприймається як перешкода, яку неможливо подолати.

Слід визнати реальні проблеми широкомасштабного впровадження відновлюваних технологій, що вимагає як інституційних, технологічних, так і суспільних змін. Але проблеми, пов'язані зі стратегією «класичного бізнесу», на порядки переважають відновлюваний маршрут: подолання підвищених і нестабільних цін на нафту, нестабільність енергопостачання, зміна клімату, забруднення повітря, великі аварії тощо.

Отже, ось сьогодні завдання для політиків і тих, хто приймає рішення: як подолати поріг у короткострокової перспективи і підготуватися до довгострокової? Досягнення енергетичних систем, які забезпечуватимуть завтрашній попит на енергію стійким та відповідальним способом, можливе, як це вже довели деякі країни. Але подальше розгортання вимагає великих зусиль з боку політиків і керівників бізнесу.

Для збільшення кількості видубуття енергііі з відновлювальних джерел відносно невідновлювальних, безперечно потрібна їхня популяризація серед суспільства. Адже чим більше людям відомо про відновлювальні джерела енргії і їх переваги над іншими - тим більш гучною стане ця тама і більше політиків будуть звертати на неї увагу.

І для зручноної демонстрації і очного аналізу переваг і ефективності

відновлювальних дерел енергії і розробляється інформаційна технологія аналізу енергетичного протенціалу відновлювальних джерел енергії.

Об’єктом дослідження є процес генерації електричної енергії у відновлювальних джерелах.

Метою роботи є розробка системи аналізу енергетичного протенціалу відновлювальних джерел енергії на платформі Node.JS засобами JetBrains WebStorm і JetBrains DataGrip, який дозволить зручно аналізувати дані енергетичного протенціалу відновлювальних джерел енергії.

Предметом дослідження є програмні сиситеми для візуалізації та аналізу даних енергетичного протенціалу відновлювальних джерел енергії.

Практичне значення роботи полягає у створені зручної програмної системи застосування якої дозволить отримати значний економічний ефект за рахунок ефективногог аналізу енергетичного протенціалу відновлювальних джерел енергії у виділених регіонах.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1. Огляд проблемної області

Світовий розвиток відновлюваної енергетики та поступова заміна традиційної генерації електроенергії є стабільними. У 2015 році глобальні інвестиції у відновлювані джерела енергії досягли рекордних 349 мільярдів доларів США. Поновлювані джерела енергії вперше склали понад 50% нових встановлених потужностей у світі. У Європейському Союзі аналогічний показник у 2016 році становив 87%. Хоча ціни на нафту та газ є найнижчими за останні 13 років, спостерігаються рекордні припливи інвестицій та стрімкий розвиток відновлюваної енергетики, що підтверджує незворотну тенденцію переходу світу на відновлювані джерела енергії.

Протягом останніх чотирьох років встановлені потужності відновлюваної енергетики в Україні поступово збільшувалися, але складна економічна ситуація в країні не сприяла досягненню мети, визначеної Національним планом дій з відновлюваної енергетики, тобто споживання енергії досягло рівня відновлюваної енергії 11. %. Станом на кінець 2016 року було встановлено 1117 МВт потужностей відновлюваної енергії, що становить приблизно 1% від загального обсягу постачання електроенергії. Найбільшу частку відновлюваної енергії в Україні займають вітрові та сонячні електростанції, які у 2016 році виробили відповідно 925 ГВт-год та 492 ГВт-год електроенергії.

Основним стимулом національної політики розвитку відновлюваної енергетики є система «зелених тарифів», яка затверджена євро та гарантується до 2030 року. Однак рішення уряду змінити тарифи, скасувати податкові пільги для відновлюваної енергетики, збільшити витрати на підключення до мережі та, можливо, накласти штрафи на дисбаланси, суттєво негативно вплинуло на інвестиційну привабливість галузі та доступність боргового фінансування.

Наявність коштів є одним із вирішальних факторів розвитку відновлюваної енергетики. У країнах ЄС з найвищим рівнем розвитку відновлюваної енергетики співвідношення боргу до власного капіталу, необхідного для отримання кредитного фінансування, становить 80/20, а річна вартість позикових коштів становить менше 5%. Сьогодні в Україні існує декілька установ та програм, спрямованих на фінансування проектів відновлюваної енергетики. Через непослідовну політику українського уряду у сфері регулювання відновлюваної енергетики фінансовим установам, як правило, потрібно вдвічі більше власного капіталу для надання кредитів для проектів з відновлюваної енергетики. Середня вартість таких кредитів становить 8-10% на рік (кредити в доларах США).

За даними міжнародної організації IRENA, Україна має найбільший технологічний потенціал для використання відновлюваної енергії в країнах Південно-Східної Європи – 408,2 ГВт (без урахування великих ГЕС). Найбільшими є технічні можливості використання вітрових та сонячних електростанцій: 321 ГВт і 71 ГВт відповідно.

До 2030 року економічно вигідний потенціал для впровадження відновлюваної енергетики в Україні оцінюється в 16-22 ГВт, а на кінець 2016 року фактично був встановлений на рівні 1,1 ГВт. Теплоенергетика має більший потенціал для впровадження відновлюваної енергії, яка, за оцінками експертів, може повністю замінити традиційну енергетику до 2030 року. Тому, за підрахунками IRENA, до 2030 року відновлювані джерела енергії можуть генерувати близько 57 мільйонів калорій теплової енергії, з яких значна частина (32,7 мільйона калорій) становить енергія біомаси. Реалізація цього прогнозу дозволить щорічно економити приблизно 7 мільярдів кубометрів природного газу.

За оцінками, вартість технологій ВЕС та СЕС у наступні 10 років знизиться на 13% та 57% відповідно, що зробить значний внесок у

впровадження відновлюваної енергетики в Україні. Маючи стабільне економічне та політичне середовище та покращені умови фінансування проектів відновлюваної енергетики, Україна зможе значно модернізуватися за допомогою технологій відновлюваної енергетики та забезпечити енергонезалежність виробництва електроенергії та теплової енергії.

1.2. Актуальність розроблення застосунку

Завершення важкої і переважно грубої роботи зі збільшення густоти населення всього за сто років вимагало багато енергії, багато енергії, а в той час її можна було добути лише з підземних покладів – з покладів горючих органічних залишків – нафти, природний газ, вугілля. Потреба в вуглецевій енергії та її відносно легка доступність у великих кількостях є вирішальним фактором у виборі, тому що вона вимагає лише великої кількості енергії, що вона і супутній процес її виробництва та використання – має другорядне значення.

Але зараз земля і люди задихаються цією енергією. Якщо людство планує жити і розвиватися більше кількох сотень років, якщо воно прагне змінити вічну історію рабства і воєн, якщо воно мріє про справжнє інтелектуальне коло і сподівається, що хоча б середній рівень емпатії зникне через до посилення воєн – потім Подальші кроки, загалом кажучи, шлях до розвитку енергетики та прогресу людства лежить не під землею з мертвим вугіллям, агресивним ураном, залишками їдкої нафти, а над землею – у поєднанні з невичерпною чистою сонячною енергією, водою та повітрям.

Ось чому важливо сприяти використанню відновлюваної енергії, і саме з цієї причини система аналізу продуктивності цих джерел енергії є чудовою. Демонстрація та аналіз ефективності відновлюваної енергії може продемонструвати її переваги та сприяти її поширенню.

1.3. Поставновка завдання дослідження

Технологія відновлюваної енергії стоїть на порозі нової ери. У багатьох країнах і регіонах відновлювані джерела енергії змогли задовольнити значну частину попиту на енергію. В останні роки швидкий і значний прогрес відновлюваної енергетики призвів до політики тісної співпраці між місцевими, національними та регіональними органами влади та бізнес-спільноти, а також до постійних технологічних інновацій та політики щодо зниження вартості відновлюваної енергії.

Значних успіхів досягнуто в будівництві нової енергетичної системи. Однак такі питання, як енергетична незалежність, ліквідація енергетичної бідності, вирішення проблеми зміни клімату та покращення здатності енергетичної системи реагувати на кризи, вимагають прискореного використання відновлюваної енергії. Нещодавні події, які мали великий вплив на суспільства в усьому світі, такі як фінансова криза, ядерна аварія на Фукусімі, великі розливи нафти та нові відкриття в галузі кліматичних наук, ще більше підкреслили цю потребу.

Визначте, що потужність відновлюваної енергії в кілька разів зростає поточне споживання електроенергії. Проте можливість використання цього потенціалу обмежено можливостями наявних енергетичних систем.

Інтенсивний розвиток вітрової та сонячної потужності потребує додаткових резервів для компенсації незбалансованих графіків роботи.

Існуючі рішення в основному базуються на використанні станцій зберігання, але останнім часом водневі технології стали конкурентоспроможною альтернативою. Сховище стисненого водню має тривалий час зберігання і мегаватну потужність, тому є економічно привабливим. Водень стає все більш важливою темою енергетики, оскільки він може бути важливим для майбутньої «чистої» енергії.

Однак існують також тенденції, які приховують можливість прискорення зростання відновлюваної енергетики. Наприклад, довгострокові переваги дослідження нових викопних видів палива (таких як арктична нафта або сланцевий газ) мобілізують потужні сили, які, як правило, ігнорують довгострокові недоліки. У той же час під час дебатів фактичну вартість відновлюваної енергії часто розуміють неправильно або неправильно, а реальність інтеграції розсіяної відновлюваної енергії в мережу часто неправильно розуміють як нездоланну перешкоду.

Слід визнати, що реальні проблеми широкомасштабного впровадження відновлюваних технологій вимагають інституційних, технологічних та соціальних змін. Але виклики, викликані стратегією «класичного бізнесу», не з'явилися знову: подолання зростання цін на нафту та її коливання, енергетична нестабільність, зміна клімату, забруднення повітря, великі аварії тощо.

Отже, сьогодні завдання політиків і тих, хто приймає рішення: як подолати поріг у короткостроковій перспективі та підготуватися до довгострокової? Як показали деякі країни, можна впровадити енергетичну систему, яка може забезпечити майбутній попит на енергію стійким і відповідальним чином. Але подальше розгортання вимагає великих зусиль від політиків і керівників бізнесу.

Щоб збільшити видобуток відновлюваної енергії порівняно з невідновлюваною енергією, безсумнівно, необхідно її популяризувати в суспільстві. Зрештою, чим більше людей знають про відновлювану енергетику та її переваги — чим гучніша гучність, тим більше політики звертатимуть на неї увагу.

І щоб полегшити демонстрацію та особистий аналіз переваг та ефективності дерев відновлюваної енергії, розробляються інформаційні технології для аналізу енергетичного потенціалу відновлюваної енергії.

Для цього необхідно розробити інформаційну систему для аналізу енергетичного потенціалу відновлюваної енергетики, а саме:

Знайти дані про енергетичний потенціал відновлюваної енергії;

Структурізувати дані за регіоном і типом джерела;

Розробити систему аналізу даних із користувацьким інтерфейсом;

Проаналізувати отримані результати та зробіть висновки.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Node.JS та NPM

Node.js — це кросплатформне середовище виконання з відкритим вихідним кодом для розробки веб-додатків на сервері. Програми Node.js написані на JavaScript і можуть працювати в різних операційних системах. Node.js базується на архітектурі, що керується подіями, і API, що не блокує введення-виведення, і спрямований на оптимізацію пропускної здатності та масштабованості програми для веб-додатків реального часу.

Довгий час усі фреймворки, які можна використовувати для веб-розробки, базуються на моделі без збереження. Модель без збереження означає, що дані, згенеровані під час одного сеансу (наприклад, інформація про конфігурацію користувача та події, що відбуваються), не можуть бути використані в наступному сеансі з користувачем. Збереження інформації про сеанс між запитами користувачів вимагає багато роботи. Але з Node.js веб-додатки нарешті мають метод двостороннього з'єднання в реальному часі. І клієнт, і сервер можуть ініціювати зв'язок, дозволяючи їм вільно обмінюватися даними.

Асинхронний IO, пов'язаний з подіями, допомагає одночасно обробляти запити — це, мабуть, найважливіша перевага Node.js. Ця функція в основному означає, що якщо Node.js отримає запит на операцію введення-виведення, він виконає операцію у фоновому режимі та продовжить обробляти інші запити.

Node.js використовує механізм виконання JavaScript V8, механізм, який використовується в Google Chrome. Node має обгортку для механізму JavaScript, що робить механізм виконання швидшим, тому обробка запитів у Node.js стає швидшою. Ще однією ключовою особливістю Node є його

здатність обробляти одночасні з’єднання з дуже малими накладними витратами на процес.

Багато великих компаній використовують Node.js. Нижче наведено список деяких з них.

- Paypal - Багато сайтів Paypal також почали переходити на Node.js.
- LinkedIn - LinkedIn використовує Node.js для живлення своїх мобільних серверів, які працюють на iPhone, Android та мобільних веб-продуктах.
- Mozilla впровадила Node.js для підтримки API браузера, який був встановлений пів мільярда разів.
- eBay розміщує свою послугу API HTTP у Node.js

Node.js найбільше підходить для потокових медіа-програм, таких як події в режимі реального часу, такі як програми чату, ігрові сервери, середовища керування документами, рекламні сервери та сервери потокового медіа.

Якщо вам потрібен високий рівень одночасності, але менше часу процесора, Node.js є хорошим вибором. Оскільки Node.js побудований на JavaScript, він найбільш підходить для створення клієнтських програм на основі тієї ж системи JavaScript.

Єдина ситуація, коли Node.js не слід використовувати, це коли програма потребує тривалого часу обробки. Структура Node.js є однопоточною. Якщо програмі потрібно виконати якісь довгострокові обчислення у фоновому режимі, вона не зможе обробляти будь-які інші запити. Як згадувалося вище, Node.js найкраще використовувати для вирішення ситуацій, які потребують меншого часу ЦП.

Модулі в Node js — це спосіб інкапсуляції коду в окремі логічні блоки. На ринку є багато готових модулів, які можна використовувати в Node js.

Ось кілька популярних модулів, які використовуються в програмах Node.js:

- Socket.io - Socket.IO забезпечує двосторонній зв'язок у реальному часі на основі подій. Цей модуль підходить для створення додатків на основі чату.
- Bluebird - Bluebird — це повністю функціональна бібліотека зобов’язань, орієнтована на інновації та продуктивність.
- Jade - Jade — це високопродуктивна система шаблонів, реалізована за допомогою JavaScript для хоста та браузера.
- Express Framework - Express — це мінімальний і гнучкий фреймворк веб-додатків Node js, який надає набір надійних функцій для Інтернету та мобільних додатків.
- Restify - restify — це легкий експрес-подібний фреймворк для створення REST API.
- MongoDB - Драйвер MongoDB Node.js — це драйвер node.js, який офіційно підтримується MongoDB.

Щоб використовувати модулі в Node.js, ви повинні спочатку встановити їх за допомогою Node Package Manager (NPM).

Наприклад, щоб встановити модуль «express», скористайтеся командою, показаною нижче.

npm install express

Ця команда завантажить необхідні файли, що містять «express modules», і відповідатиме за встановлення

Після встановлення модуля вам потрібно використовувати ключове слово «require», щоб використовувати модуль у Node.js. Це ключове слово є методом, який Node.js використовує для включення функціональних можливостей модуля в програму.

Давайте розглянемо приклад того, як використовувати ключове слово «require». Наступний приклад коду показує, як використовувати функцію require

```
var express=require('express'); // підключаєм express
var app=express(); // створюєм об’єкт express
app.set('view engine','jade'); // викликаємо метод set
app.get('/',function(req,res) {
});
var server=app.listen(3000,function() { });
```

// вище викликаємо метод listen для прослуховування порта localhost:3000

Node.js може створювати власні модулі та дозволяти вам включати ці спеціальні модулі у вашу програму Node.js.

Інформація про модуль зберігається у файлі package.json, де перераховані всі залежності проекту та їх версії.

Рис. 2.1.1. Приклад файлу *package.json*

2.2 Angular

Angular (широко відомий як Angular 2 або Angular 2+) — це фреймворк інтерфейсу з відкритим вихідним кодом, написаний командою Google Angular і спільнотою приватних розробників і компаній. Angular — це AngularJS, переосмислений і повністю переписаний тією ж командою розробників.

Після первинного перезапису AngularJS була названа Angular 2 командою розробників, але це викликало плутанину серед інших розробників. Щоб пояснити різницю між ними і підкреслити, що це незалежні проекти, команда вирішила використовувати назву AngularJS для фреймворка 1.X і Angular без JS для версії, починаючи з 2.0. Основна відмінність Angular від AngularJS.

Архітектура сервісу на Angular. Основними елементами в розробці є компоненти, ін'єкції залежностей, сервіси, шаблони, метадані, модулі, директиви та біндінг даних.

Як згадувалося вище, Angular є ретельно переписаним AngularJS.

- Angular не використовує поняття «області» чи контролера, а використовує ієрархію компонентів як основну архітектурну концепцію.
- Додано Angular CLI, який дозволяє почати створювати нову програму, просто написавши команду ng new [app name]
- Angular рекомендує та використовує мову TypeScript від Microsoft, яка включає такі можливості:
  - Система типізації
  - Класи, а отже Об'єктно-орієнтоване програмування

- Узагальнене програмування

- TypeScript — це наднабір ECMAScript 6 (ES6) і зворотно сумісний зі стандартом ECMAScript 5 (тобто JavaScript). Angular також має функції ES6, такі як:
  - Анонімні функції
  - Python-подібні генератори
  - Ітератори
  - Цикли типу For/Of
  - Рефлексія
- Ітеративні колбеки завдяки використанню RxJS. RxJS обмежує видимість і можливості налагодження стану до певної міри, але за допомогою плагінів, таких як ngReact і ngrx, це легко вирішити
- Модульність – велика частина основних функцій перенесена на модулі.
- Асинхронна компіляція шаблонів
- Динамічне завантаження
- У Angular відмінний синтаксис написання виразів, застосовуючи "[ ]" для біндінгу даних властивостей, та "( )" для біндінгу даних івентів
- Зміна контролерів та $scope(області видимості) компонентами та директивами – компонент являється директивою з шаблоном

Починаючи з Angular 9, усі нові програми використовують компілятор Ivy. Тому команда Angular працюватиме над покращенням цього компілятора, який, у свою чергу, має зменшити загальний розмір пакета.

Очікується, що кожна наступна версія буде зворотно сумісною з попередньою версією. Також Google пообіцяв випускати оновлення двічі на рік.

Angular з кожним роком стає все більш популярним. Станом на серпень 2020 року щодня завантажується близько 1,5 мільйона ядер Angular/ядра, що подвоюється щороку.

Технологія Angular використовується у веб-додатках таких компаній:

- Google (Gmail, Google Play, Google Translate, Google Ads, Youtube);
- Paypay;
- Upwork;
- Expedia;
- Lego;
- Adidas.

На ринку програмного забезпечення Angular займає друге місце з 9,18% і після ASP.NET з 44,9% відповідно.

2.3 База даних MySQL

База даних - це збір даних системи. База даних підтримує зберігання та роботу з даними. База даних спрощує управління даними.

Система управління базами даних (СУБД). Це набір програм, програми, які який дозволяють використовує користувачам користувачі, отримувати щоб підтримувати доступ до баз бази даних, маніпулювати даними, звітувати або представляти відображати дані. базу даних База даних також допомагає контролює контролювати доступ до бази даних. даних Система База управління даних базами системи даних управління не є новою концепцією, тому тому. вона вперше була представлена в 1960-х роках. Вважається, що інтегрований репозиторій Чарльза Бахмана є першою СУБД в історії. З часом технології баз бази даних значно розвинулися, а використання та очікувані функції баз бази даних значно зросли.

Існують 4 основних види СУБД. Розглянемо їх детальніше.

- **Ієрархічна** – цей тип бази даних використовує зв’язок «батька-дочірня» для зберігання даних. Цей тип баз даних сьогодні використовується рідко. Його структура схожа на дерево, з вузлами, що представляють записи, а гілки — поля. Реєстр Windows, який використовується в Windows XP, є прикладом ієрархічної бази даних. Параметри конфігурації зберігаються в деревоподібній структурі з вузлами.
- **Мережеві СУБД** - цей тип бази даних підтримує кілька відносин захисту. Зазвичай це призводить до складних структур баз даних. RDM Server є прикладом системи керування базами даних, яка реалізує мережеву модель.
- **Реляційні СУБД** - цей тип бази даних визначає зв’язки з базою даних у вигляді таблиць, також відомих як зв’язки. На відміну від мережевих баз даних, СУБД не підтримує багато зв'язків. Реляційні бази даних зазвичай мають заздалегідь визначені типи даних, які вони можуть

підтримувати. Це найпопулярніший тип бази даних на ринку. Прикладами систем керування реляційними базами даних є MySQL, Oracle та Microsoft SQL Server.

- **Об'єктно-орієнтована СУБД** - цей тип підтримує зберігання нових типів даних. Дані, які потрібно зберігати, мають форму об’єктів. Об’єкти, що зберігаються в базі даних, мають атрибути (тобто стать, вік) і методи, які визначають спосіб обробки даних. PostgreSQL є прикладом об'єктно-орієнтованої реляційної бази даних.

SQL означає - **Structured Query Language**, або структурована мова запитів. SQL є стандартною мовою для роботи з реляційними базами даних. Ви можете використовувати SQL для вставки, пошуку, оновлення та видалення записів бази даних. SQL може виконувати багато інших операцій, включаючи оптимізацію та обслуговування бази даних. Реляційні бази даних, такі як MySQL Database, Oracle, Ms SQL Server і Sybase, використовують SQL. Синтаксис SQL, який використовується в цих базах даних, дуже схожий, але деякі використовують кілька різних синтаксисів, навіть рідний синтаксис SQL.

MySQL — це реляційна база даних з відкритим вихідним кодом. MySQL є кросплатформенним, що означає, що він може працювати на багатьох різних платформах, таких як Windows, Linux і Mac OS. У порівнянні з іншими системами реляційних баз даних MySQL має вищу продуктивність. Це тому, що він простий у проектуванні та обслуговуванні багатопривідного двигуна. Для взаємодії з MySQL вам знадобиться сервер доступу до сервера, який може взаємодіяти з сервером MySQL. MySQL підтримує багато підключень користувачів.

DataGrip — це інструмент для проектування та моделювання візуальних баз даних для розробників SQL. Це дозволяє легко створювати нові фізичні моделі даних та модифікувати існуючі бази даних MySQL за допомогою

засобів розробки назад/вперед та керування змінами. Метою MySQL Workstation є надання інтерфейсу для роботи з базою даних більш простим і структурованим способом.

У середовищі DataGrip я створив базу даних MySQL для Robot Telegram. Щоб моя база даних повністю підтримувала українську мову, я використав таку команду для її створення з кодуванням cp1251: CREATE DATABASE bot DEFAULT CHARACTER SET cp1251 COLLATE cp1251_bin;

РОЗДІЛ 3. ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Програмна реалізація серверу

Перед початком роботи нам потрібно ініціалізувати наш проект за допомогою пакетного менеджера NPM, ввівши команду npm init --y у консолі.

Створюємо основний виконуваний файл сервера app.js, де будуть реалізовані всі сервісні структури.

Далі необхідно створити підключення до бази даних MySQL у вигляді модуля для управління нею. Для керування базою даних ми використовуємо бібліотеку Sequelize ORM. Для цього підключіться до нього та встановіть його за допомогою команди **npm install --save sequelize**. Щоб підключитися до бази даних, ми створюємо об’єкт з параметрами підключення.

```
const client = new  Sequelize('bot', 'root', 'root', {
    host: 'localhost',
    dialect: 'mysql'
});
```

Рис. 3.1.1. Об’єкт з параметрами підключення до бази

Використовую метод define, щоб створити модель, яка зчитує базову таблицю як об’єкт.

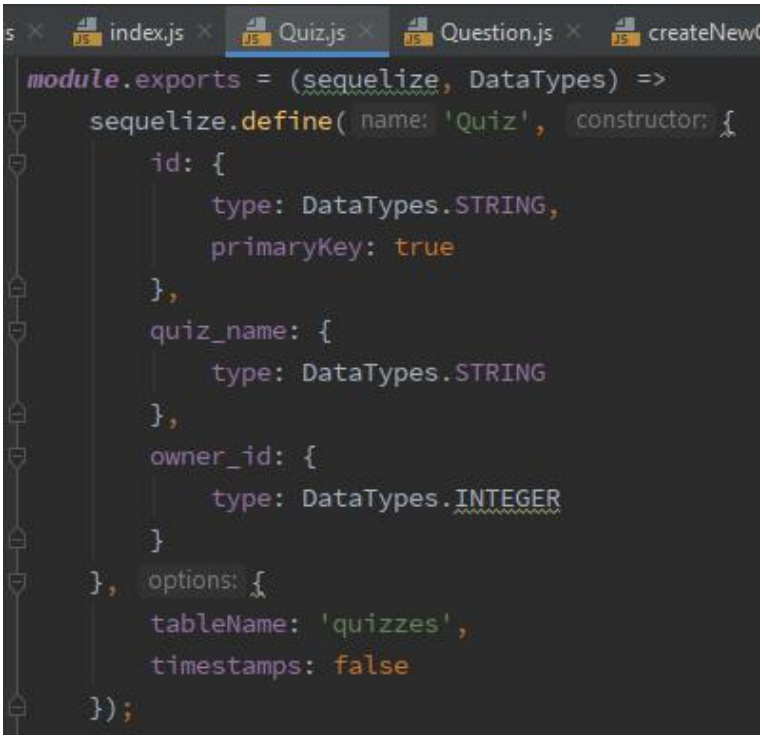


Рис. 3.1.2. Модель таблиці

Для доступу до моделей таблиць бази у файлі *index.js* використовую синглтон підключення.

```
function getModels() {
    fs.readdir( path: './dataBase/models', callback: (err, files) => {
        files.forEach(file => {
            const [modelName] = file.split( separator: '.');
            models[modelName] = client.import(resolve( pathSegments: `./dataBase/models/${modelName}`))
        })
    })
}

return {
    getModel: modelName => models[modelName],
    setModels: () => getModels()
};
```

Рис. 3.1.3. Синглтон підключення моделей

3.2 Структура сервісу

Сервіс створено у вигляді веб-сторінки, що робить його загальнодоступним.

Область	Енергія сонця	Енергія вітру	МГЕС	Геотермальна енергія	Енергія біомаси	Всього
Автономна Республіка Крим	3 603	22 128	1	840	1 273	**27 844**
Вінницька область	3 646	13 393	24	40	6 192	**23 295**
Волинська область	2 770	7 184	1	40	2 239	**12 234**
Дніпропетровська область	4 388	38 978	2	120	5 128	**48 616**
Донецька область	3 646	32 387	5	200	2 835	**39 072**
Житомирська область	4 102	10 640	8	50	4 575	**19 374**
Закарпатська область	1 757	1 163	132	1 400	1 209	**5 661**
Запорізька область	3 737	33 196	0	40	3 646	**40 620**
Івано-Франківська область	1 911	2 416	59	600	1 671	**6 658**
Київська область	3 868	11 983	3	40	4 961	**20 855**
Кіровоградська область	3 381	21 226	15	40	4 482	**29 144**
Луганська область	3 669	32 591	2	80	2 042	**38 384**
Львівська область	3 002	8 015	46	1 400	2 672	**15 135**
Миколаївська область	3 382	30 043	3	80	3 435	**36 943**
Одеська область	4 580	34 719	1	240	4 912	**44 453**
Полтавська область	3 953	14 522	6	1 400	5 662	**25 544**
Рівненська область	2 756	7 745	3	40	2 594	**13 139**
Сумська область	3 277	11 096	2	560	5 009	**19 945**
Тернопільська область	1 901	6 983	12	80	3 019	**11 995**
Харківська область	4 320	27 119	10	1 300	5 160	**37 908**
Херсонська область	3 913	34 761	1	1 500	3 360	**43 335**
Хмельницька область	2 839	10 429	8	40	4 668	**17 984**
Черкаська область	2 874	10 558	8	40	4 150	**17 630**
Чернівецька область	4 381	2 414	24	40	1 252	**8 111**
Чернігівська область	1 113	12 311	1	800	5 932	**20 157**
Територіальні води та внутрішні водойми		**250 000**				
Всього	**82 769**	**688 000**	**376**	**10 810**	**92 078**	**874 033**

Рис 3.2.1. Розділ аналізу даних потенціалу встановленої потужності ВДЕ

У розділі Потенціал встановленої потужності ВДЕ зображено порівняльні дані потужностей відновлювальних джерел енергії у мегаватах, розподілені по областях і видах енергії. Також в даному розділі можна сортувати дані по вибраному параметру. Наприклад на Рис 3.1.1. зображено зортування по назвах областей.

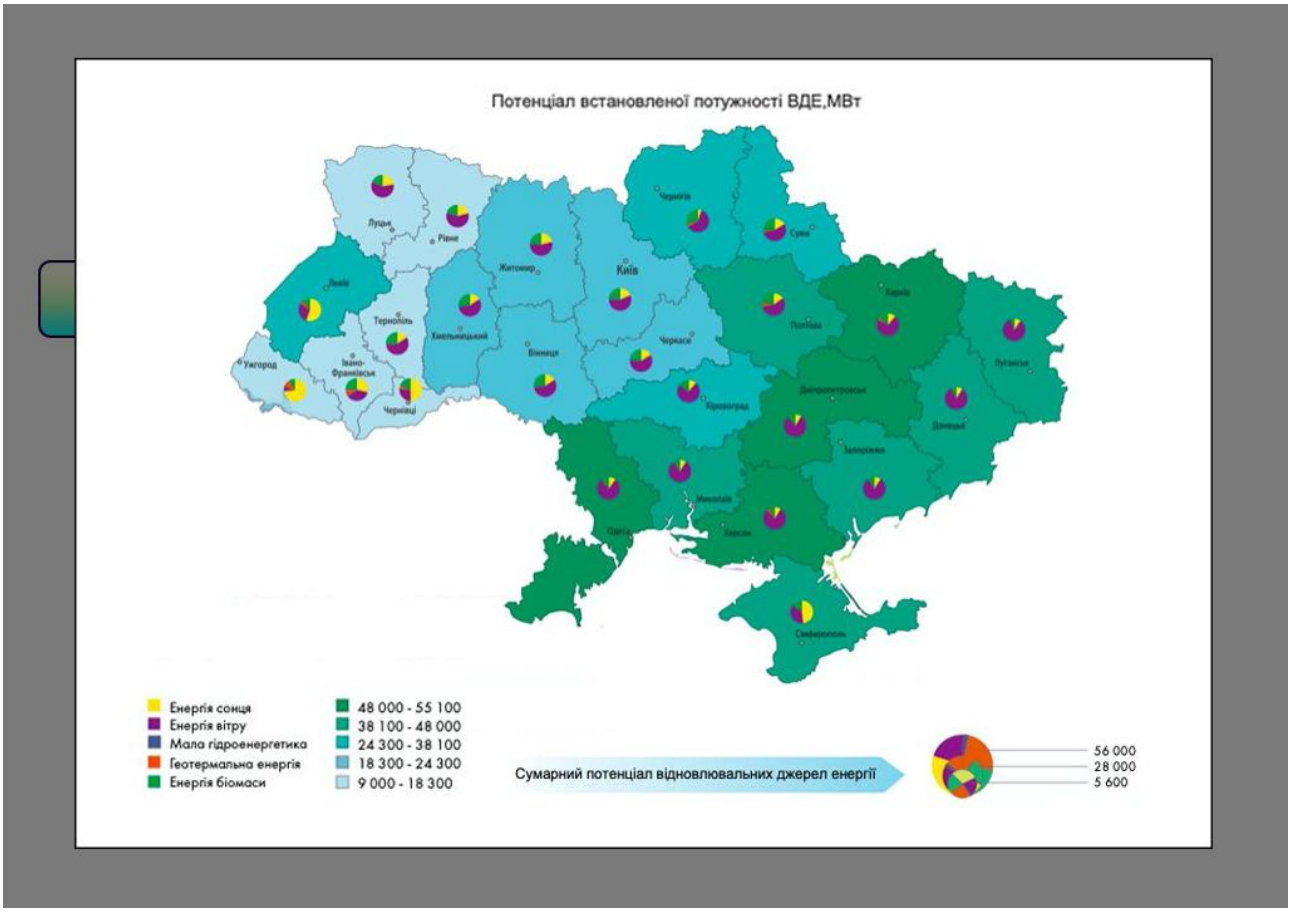
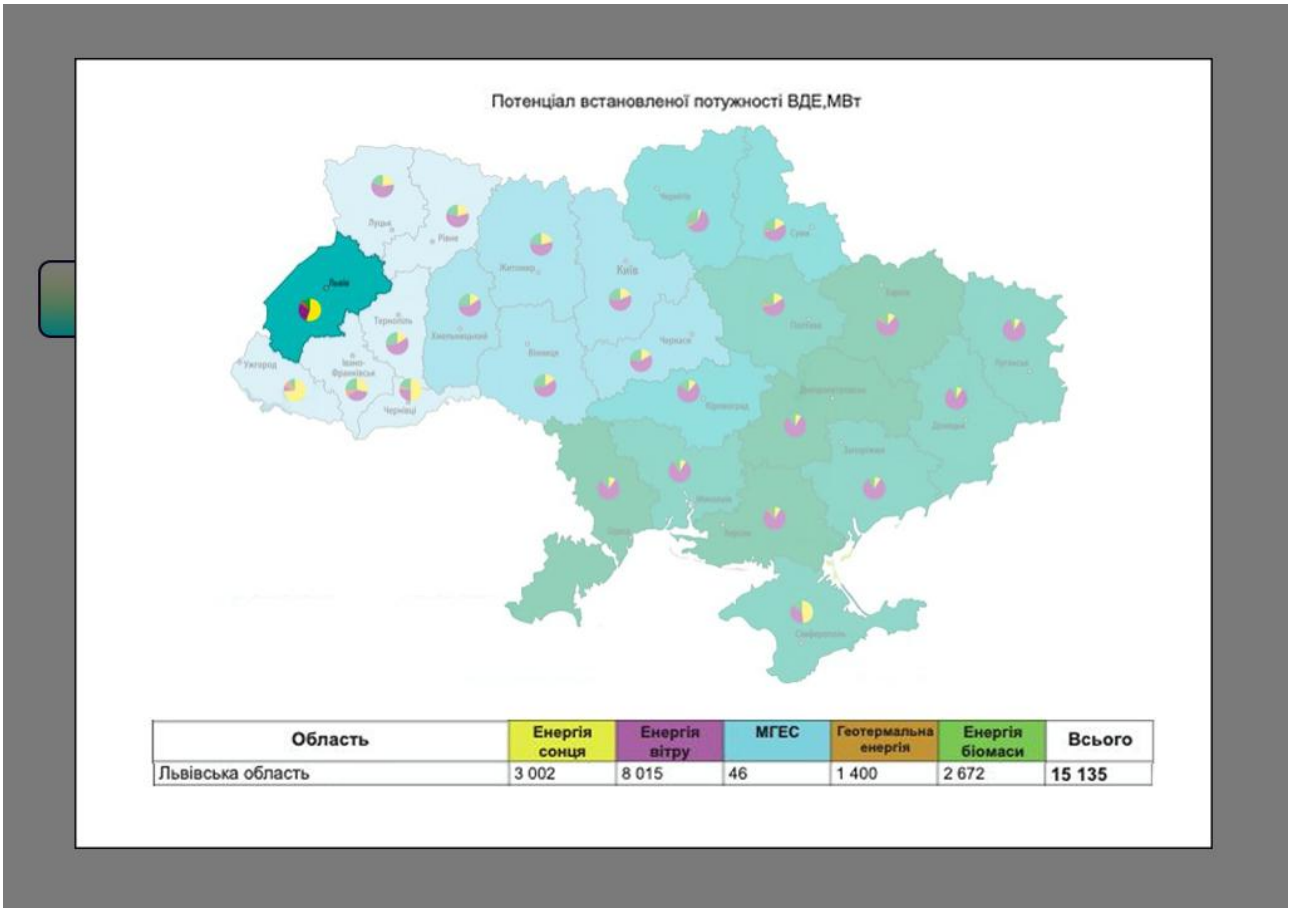


Рис 3.2.2. Карта потенціалів встановленої потужності ВДЕ

При натиску на кнопку “Карта” на екрані відображається карта потенціалів встановленої потужності ВДЕ з візуалізацією інфографіки по областях.



Область	Енергія сонця	Енергія вітру	МГЕС	Геотермальна енергія	Енергія біомаси	Всього
Львівська область	3 002	8 015	46	1 400	2 672	**15 135**

Рис 3.2.3. Вибіркова інформація по області

При натиску на будь-яку область на карті буде відображена інформація потенціалу відновлювальних джерел вибраної області.

Оцінка енергетичного потенциалу ВДЕ України та еквівалентного електролізу «зеленого» водню

Карта

Сценарій	ВДЕ	Джерело інформації	Потужність, ГВт	Середньорічне виробництво електроенергії, млрд кВт•год	Середньорічне виробництво «зеленого» водню, млрд кВт•год
	ВЕС	Разом, у тому числі:	466	1428	317
		оншор (IRENA)	320	858	191
		офшор (NREL)	146	570	126
	СЕС	IRENA	71	88	20
Базовий	Всього ВДЕ		537	1516	337
	ВЕС	Разом, у тому числі:	688	2174	483
		оншор (ІВЕ НАНУ)	438	1190	254
		офшор (ІВЕ НАНУ)	250	984	219
	СЕС	(ІВЕ НАНУ)	83	99	22
Оптимістичний	Всього ВДЕ		771	2273	505
Песимістичний	Всього ВДЕ (сонце і вітер)	Енергетична стратегія України період до 2035 р.		25	5,5

Рис 3.2.4. Розділ оцінки енергетичного потенціалу ВДЕ України та еквівалентного електролізу “зеленого” водню

В другому розділі можна прорівняти енергетичний потенціал ВДЕ відносно електролізу “зеленого” водню.

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП-ПРОЕКТУ

5.1. Опис ідеї стартапу

У сьогоденні ідея створити щось своє не здається неможливою. Однак створити дійсно хороший продукт, який сподобається людям і буде дійсно корисним для інших людей, завдання не з легких. Ви повинні чітко розуміти, з чого почати і що робити, і розуміти, що успіх не вийде за день і не припиниться після невдалої спроби.

Як випливає з цього терміну, «стартап» не є постійною фазою для будь-якого бізнесу, і він не стосується лише компаній у сфері технологій. Це життєво важливий, ранній етап життєвого циклу бізнесу, і він може стосуватися практично будь-якої галузі.

Загалом, у стартапів, як правило, мало працівників і потенціал швидкого зростання. Вони забезпечують широку привабливість продуктів, які або ще не існують, або вирішують проблему краще, ніж доступні на даний момент варіанти.

Розглянемо основні характеристики успішної стартап-компанії:

- Іноваційність
- Вирішення нагальних проблем
- Швидкий ріст
- Турбуючий суспільсво
- Маштабність

Стартапи – це компанії, створені для швидкого зростання та збільшення без географічних обмежень. Це основна відмінність між стартапами та іншими молодими бізнесами.

Витрати більшості стартапів перевищують їхні доходи, тому багато з них потребують зовнішнього фінансування. Без цього ці компанії не зможуть ефективно розвивати та продавати свої інноваційні продукти чи послуги.

5.2. Маркетингова програма стартап-проекту

Не секрет, що маркетинг є важливим для успіху стартапу. Успішна маркетингова стратегія запуску може допомогти вашому бізнесу отримати популярність, збільшити продажі та розвиватися. Головне — зрозуміти, як ефективно рекламувати свій стартап, щоб узгодити вашу кампанію з вашим цільовим ринком і бізнес-цілями. На щастя, ми тут, щоб допомогти вам спростити цей посібник із усіма, що вам потрібно знати про маркетинг стартапів.

Успішна маркетингова кампанія не відбувається просто за ніч; Щоб створити навмисну, цілеспрямовану кампанію, потрібно зрозуміти, що таке маркетингова стратегія і як зробити так, щоб вона працювала для вашого бізнесу. Давайте розберемося, що вам потрібно знати, щоб успішно продати свій стартап.

Маркетингова стратегія вашого стартапу — це, по суті, план дій, щоб привернути увагу клієнтів, що допоможе розвинути ваш бізнес. Маркетингові стратегії розробляються шляхом дослідження та планування для досягнення цільового ринку клієнтів, на який орієнтований ваш стартап.

Ці стратегії роблять більше, ніж просто підвищують продажі; ефективна маркетингова стратегія надає різноманітні переваги, які можуть допомогти розвивати та формувати ваш стартап. Давайте подивимося на конкретні переваги, які маркетингова стратегія пропонує стартапам:

- Забезпечує подальше розуміння вашого цільового ринку
- Надає інструменти для точного налаштування продуктів і послуг відповідно до потреб клієнтів і досягнення бізнес-цілей
- Підвищує впізнаваність вашого бренду
- Допомагає вам більше зосередитися на продажах і ваших фінансових цілях
- Допомагає виділити ваш бізнес серед інших

5.3. Аналіз технологічних можливостей реалізації ідей проекту

Проект буде реалізований за технологіями Node.js і Angular. Вибір цього технічного методу не випадковий. Зрештою, ці технології широко відомі, їх швидко освоюють, їх використовують як досвідчені професіонали, так і новачки. Тому проблем і труднощів у реалізації власних підприємницьких проектів не виникає, а менш відомі технології забирають у команди додатковий час для освоєння програмно-технічного забезпечення продуктів на основі неякісних технологій.

За допомогою технологій Node.js та Angular ви можете виконувати будь-яке складне технічне завдання, оскільки вони підтримують різні можливі реалізації у вибраній області.

ВИСНОВКИ

В процесі створення сервісу була розроблена інформаційна технологія аналізу енергетичного протенціалу відновлювальних джерел енергії у виділених регіонах, зі зручним користувацьким інтерфейсом для аналізу даних. Була узгоджена робота програмних компонентів, розроблений користувацький інтерфейс та весь необхідний функціонал продукту. Були проведені чисельні тести та порівняння на основі використання новітніх інструментів, після чого результати були записані в систему і проаналізовані. Сервіс продовжує розвиватись та покращуватись, а також регулярно піднімається на відкритий сервер для тестування в онлайн-режимі.

ПЕРЕЛІК ПОСИЛАНЬ

1. JavaScript: The Definitive Guide («JavaScript. Подробное руководство»), Дэвид Флэнаган, 2008 – 982 ст.
2. SQL | Mastering SQL: Мартін Грабер, 2016 - 642 ст.
3. Книга «Node.js в действии. 2-е издание», Кантелон М., Хартер М., Головайчук Т., Райлих Н., 2018 – 432 ст.
4. The Node Beginner Book: Manuel Kiessling.
5. https://www.sciencedirect.com/topics/engineering/renewable-energy-technologies
6. https://sequelize.readthedocs.io/en/latest/
7. http://uare.com.ua/novyny/472-chomu-varto-rozvivati-vidnovlyuvanu-energetiku.html
8. https://angular.io/docs
9. https://docs.npmjs.com/
10. https://www.npmjs.com/package/express

ДОДАТОК

Лістинг програмного коду

index.js

```
const Sequelize = require('sequelize');
const fs = require('fs');
const {resolve} = require('path');



module.exports = (() => {
    let instance;

    function initConnection() {
        const client = new  Sequelize('bot', 'root', 'root', {
            host: 'localhost',
            dialect: 'mysql'
        });
        let models = {};

        function getModels() {
            fs.readdir('./dataBase/models', (err, files) => {
                files.forEach(file => {
                    const [modelName] = file.split('.');
                    models[modelName] =
client.import(resolve(`./dataBase/models/${modelName}`))
                })
            })
        }

        return {
            getModel: modelName => models[modelName],
            setModels: () => getModels()
```

```
        };
    }

    return {
        getInstance: () => {
            if(!instance) instance = initConnection();
            return instance;
        }
    }
})();
```

index.html

```
<!DOCTYPE HTML><html>
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2/css/all.css" integrity="sha384-fnmOCqbTlWIlj8LyTjo7mOUStjsKC4pOpQbqyi7RrhN7udi9RwhKkMHpvLbHG9Sr" crossorigin="anonymous">
  <link rel="stylesheet" href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css">

  <style>
    html {
        font-family: Arial;
        display: inline-block;
        margin: 0px auto;
        text-align: center;
    }
    /*h2 { font-size: 3.0rem; }*/
    p { font-size: 3.0rem; }
```

```
.units { font-size: 1.9rem; }
.dht-labels{
    font-size: 1.5rem;
    vertical-align:middle;
    padding-bottom: 15px;
}

.slidecontainer {
    width: 90%;
    padding: 10px 0 13px 0;
    margin: 0 auto;
}

.slider {
    -webkit-appearance: none;
    width: 100%;
    height: 15px;
    border-radius: 5px;
    background: #d3d3d3;
    outline: none;
    opacity: 0.7;
    -webkit-transition: .2s;
    transition: opacity .2s;
}

.slider:hover {
    opacity: 1;
}

.slider::-webkit-slider-thumb {
    -webkit-appearance: none;
    appearance: none;
```

```
            width: 25px;
            height: 25px;
            border-radius: 50%;
            background: #4CAF50;
            cursor: pointer;
        }

        .slider::-moz-range-thumb {
            width: 25px;
            height: 25px;
            border-radius: 50%;
            background: #4CAF50;
            cursor: pointer;
        }
    </style>

</head>
<body>
<script src="https://cdn.metroui.org.ua/v4/js/metro.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/2.2.0/socket.io.js"></script>

<h2>Live data from module</h2>

<div style="display: flex; justify-content: center">
    <p style="margin-right: 80px">
        <i class="fas fa-thermometer-half" style="color:#059e8a;"></i>
        <span class="dht-labels">Temperature</span>
        <span id="temperature">0.0</span>
        <sup class="units">&deg;C</sup>
    </p>
    <p style="margin: 0 0 0 80px">
```

```
      <i class="fas fa-tint" style="color:#00add6;"></i>
      <span class="dht-labels">Humidity</span>
      <span id="humidity">0.0</span>
      <sup class="units">%</sup>
    </p>
</div>


<div style="border: black solid; width: 85%; margin: 0 auto">
  <div style="border-right: black solid; width: 20%; display: inline-block; height: 143px">
      <p style="font-size: 1.1rem; display: flex; margin: 0 0 0 0; padding-bottom: 15px; text-align: center">
          Set optimal temperature for you (default is 24)
      </p>
      <p style="font-size: 1.2rem; margin: 0 0 0 0;">
          Temperature: <span id="demo" style="font-weight: bold;"></span><sup style="font-size: 1.3rem">&deg;C</sup>
      </p>
      <p class="slidecontainer">
          <input type="range" min="15" max="35" value="24" class="slider" id="myRange">
      </p>
  </div>

  <div style="display: inline-block; width: 20%; border-right: black solid">
      <p style="display: flex; font-size: 1.1rem; margin: 0 0 0 0; padding-bottom: 10.5%; text-align: center">
          Keep temperature in your optimal range
      </p>
      <p style="font-size: 1.2rem; padding-bottom: 2%;">
          <span>Smart heating</span>
```

```
        <input id="smartHeating" type="checkbox" data-role="switch"
data-material="true" onclick="switchHeating()">
      </p>
    </div>

    <div style="display: inline-block; width: 20%; border-right: black solid;">
      <p style="font-size: 1.2rem; padding-bottom: 1px">
        Turn on/off heating for specific time
      </p>
      <p>
        <select id="boostTime" style="width: 40%; display: inline-block">
          <option value="10 min">10 min</option>
          <option value="20 min" selected>20 min</option>
          <option value="30 min">30 min</option>
          <option value="40 min">40 min</option>
          <option value="50 min">50 min</option>
          <option value="60 min">60 min</option>
        </select>
        <input id="switchBoost" type="checkbox" data-role="switch"
data-material="true" onclick="switchBoostState()">
      </p>
    </div>

    <div style="display: inline-block; width: 20%; border-right: black solid;">
      <p style="font-size: 1.2rem;">
        Set timer for on/off heating at specific time
      </p>
      <p>
        <input id="timerTime" type="time" value="00:00" style="width: 40%;
display: inline-block">
        <select id="timerAction" style="width: 20%; display: inline-block">
          <option value="on" selected>on</option>
```

```
            <option value="off">off</option>
        </select>
        <input type="checkbox" data-role="switch" data-material="true"
onclick="switchTimer()">
    </p>
  </div>

  <div style="display: inline-block; width: 18.5%">
    <p style="font-size: 1.2rem">
      Visualization
    </p>
    <br>
    <button style="display: block; width: 50%; margin: 0 auto"
onclick="showHideTempChart()">Temperature</button>
    <br>
    <button style="display: block; width: 50%; margin: 0 auto"
onclick="showHideHumChart()">Humidity</button>
  </div>
</div>

<iframe style="display: none; margin: 0 auto" id="tempChart" width="1140"
height="541.25"
src="https://app.powerbi.com/reportEmbed?reportId=69c327f3-c7fa-4557-960a-34
16b73ca620&autoAuth=true&ctid=9454a219-26a1-44ea-b20d-ccf8096f8368&con
fig=eyJjbHVzdGVyVXJsIjoiaHR0cHM6Ly93YWJpLXdlc3QtZXVyb3BlLWItcH
JpbWFyeS1yZWRpcmVjdC5hbmFseXNpcy53aW5kb3dzLm5ldC8ifQ%3D%3D"
frameborder="0" allowFullScreen="true"></iframe>
<iframe style="display: none; margin: 0 auto" id="humChart" width="1140"
height="541.25"
src="https://app.powerbi.com/reportEmbed?reportId=ac6576fa-54de-4df3-aa45-c5
5fd6c1fd0d&autoAuth=true&ctid=9454a219-26a1-44ea-b20d-ccf8096f8368&conf
ig=eyJjbHVzdGVyVXJsIjoiaHR0cHM6Ly93YWJpLXdlc3QtZXVyb3BlLWItcHJ
```

```
pbWFyeS1yZWRpcmVjdC5hbmFseXNpcy53aW5kb3dzLm5ldC8ifQ%3D%3D"
frameborder="0" allowFullScreen="true"></iframe>

</body>

<script>
  let socket = io('http://localhost:3000');

  const temperatureInfo = document.getElementById('temperature');
  const humidityInfo = document.getElementById('humidity');

  socket.on('updateLiveModuleData', ({temperature, humidity}) => {
     temperatureInfo.innerHTML = temperature;
     humidityInfo.innerHTML = humidity;
  });

  const slider = document.getElementById("myRange");
  const output = document.getElementById("demo");
  output.innerHTML = slider.value;

  slider.oninput = function() {
     output.innerHTML = this.value;
     socket.emit('set_temp', this.value);
  }

  let showTempChart = false;
  let showHumChart = false;

  function showHideTempChart() {
     const tempChart = document.getElementById('tempChart');
     showTempChart = !showTempChart;
```

```
    showTempChart ? tempChart.style.display = 'block' : tempChart.style.display
= 'none';
  }

  function showHideHumChart() {
    const tempChart = document.getElementById('humChart');
    showHumChart = !showHumChart;

    showHumChart ? tempChart.style.display = 'block' : tempChart.style.display =
'none';
  }

  function switchHeating() {
    const smartHeating = document.getElementById('smartHeating');

    socket.emit('switch_heating', smartHeating.value);
  }

  function switchBoostState() {
    const switchBoost = document.getElementById('switchBoost');
    const boostTime = document.getElementById('boostTime');

    const data = {
      boost_state: switchBoost.value,
      boost_time: boostTime.value
    }

    socket.emit('switch_boost', data);
  }

  function switchTimer() {
    const timerTime = document.getElementById('timerTime');
```

```
        const timerAction = document.getElementById('timerAction');

        const data = {
            execute_at: timerTime.value,
            action: timerAction.value
        }
        socket.emit('switch_timer', data);
    }

</script>

</html>
```

getQuestionById.js

```
const db = require('../dataBase').getInstance();

module.exports = async (id) => {
    try {
        const QuestionModel = db.getModel('Question');

        if (!id) throw new Error('No question id!');

        const gotQuestion = await QuestionModel.findOne({
            attributes: [
                "question",
                "answer"
            ],
            where: {
                id
            }
        });
```

```
        // if (!gotQuiz) throw new Error('Quiz for this user do not exist!');

        console.log(gotQuestion);

        return gotQuestion ? gotQuestion : null;
    } catch (err) {
        err.code = 400;
        throw err;
    }

};
```

deleteQuestion.js

```
const db = require('../dataBase').getInstance();

module.exports = async (id) => {
    try {
        const QuestionModel = db.getModel('Question');

        if (!id) throw new Error('No question id!');

        const deletedQuestion = await QuestionModel.destroy({
            where: {
                id
            }
        });
        if (!deletedQuestion) throw new Error('Question is NOT deleted!');

        console.log(deletedQuestion);

        return deletedQuestion ? deletedQuestion : null;
    } catch (err) {
        err.code = 400;
```

```
        throw err;
    }

};
```

callbackQueryListener.js

```
const {bot} = require('../botConnector.js');

const getQuestionById = require('../controllers/getQuestionById.js');
const deleteQuestion = require('../controllers/deleteQuestion.js');
const addQuestion = require('../callbacks/addQuestion.js');
const getQuestionByQuizId = require('../controllers/getQuestionByQuizId.js');
const sendMenu = require('../callbacks/sendMenu.js');

module.exports = async (callbackQuery) => {
    const {message: {chat, message_id}} = callbackQuery;
    let action = callbackQuery.data;
    action = action.split('_');

    console.log(callbackQuery);
    if (action[1] === "quiz") {

        let quest = await getQuestionByQuizId(action[0]);

        let questions = [];

        questions.push([{
            text: "➕Добавити запитання",
            callback_data: action[0] + "_AddQuestion"
        }]);

        quest.forEach(function (i) {
```

```
        questions.push([{
            text: i.question,
            callback_data: action[0] + "_question_" + i.id
        }]);
    });

    questions.push([{
        text: "⬅Назад у меню",
        callback_data: chat.id + "_BackToMenu"
    }]);

    const opts = {
        chat_id: chat.id,
        message_id,
        "reply_markup": {
            "inline_keyboard": questions
        }
    };


    if (!quest[0]) {
        await bot.editMessageText(`У даної вікторини немає запитань!
\nДобавте запитання!`, {
            chat_id: chat.id,
            message_id,
            "reply_markup": {
                "inline_keyboard": [
                    [
                        {
                            text: "➕Добавити запитання",
                            callback_data: action[0] + "_AddQuestion"
                        }
```

```
                ],
                [
                    {
                        text: "⬅Назад у меню",
                        callback_data: chat.id + "_BackToMenu"
                    }
                ]
            ]
        }
    });
} else {
    await bot.editMessageText(`Вікторина: ` + quest[0].Quiz.quiz_name + `
                        \nКод вікторини: ` + action[0], opts);
}


} else if (action[1] === "question") {
    let quest = await getQuestionById(action[2]);
    await bot.editMessageText(`Запитання: ` + quest.question + `
                        \nвідповідь: ` + quest.answer,
        {
            chat_id: chat.id,
            message_id
            ,
            "reply_markup": {
                "inline_keyboard": [
                    [
                        {
                            text: "❌Видалити",
                            callback_data: action[0] + "_delete-question_" + action[2]
                        },
                        {
```

```
                    text: "⬅Назад",
                    callback_data: action[0] + "_quiz"
                }
            ]
        ]
    }
});

} else if (action[1] === "BackToMenu") {
    await sendMenu(callbackQuery.message);

} else if (action[1] === "AddQuestion") {
    const id = action[0];
    const owner_id = chat.id;
    await addQuestion({id, owner_id});

} else if (action[1] === "delete-question") {
    let isDelete = await deleteQuestion(action[2]);

    const toQuestionBtn = {
        chat_id: chat.id,
        message_id,
        "reply_markup": {
            "inline_keyboard": [[{
                text: "⬅Назад до вікторини",
                callback_data: action[0] + "_quiz"
            }]]
        }
    };

    if (isDelete) {
```

```
        await bot.editMessageText(`Завдання успішно видалено.`,
toQuestionBtn)
      } else {
        await bot.editMessageText(`Щось пішло не так, завдання не видалено.`,
toQuestionBtn)
      }
   }
};
```

sendMenu.js

```
const {bot} = require('../botConnector.js');

module.exports = async (msg) => {
   if (msg.chat.type === 'private'){
      await bot.sendMessage(msg.chat.id, `Quizzers_BoT - це ваш універсальний
помічник для проведення вікторин.
                              \n\nТут ви можете створити Нову вікторину або
переглядати та редагувати уже створені.`, {
         "reply_markup": {
            "keyboard": [["Створити вікторину"], ["Мої вікторини"]],
            "one_time_keyboard": true,
            "resize_keyboard": true
         }
      });
   }
};
```

Styles.scss

```
html, body {
padding: 0;
margin: 0;
height: 100%;
```

```
width: 100%;
box-sizing: border-box;
zoom: 1;
font-family: sans-serif;
letter-spacing: .5px;
font-size: 16px;
font-weight: 400;
}

* {
box-sizing: border-box;
}

html, body {
height: 100%;
}

body {
margin: 0;
font-family: 'Roboto', sans-serif;
}

h1, h2, h3, h4 {
margin-top: 0;
margin-bottom: 0;
font-weight: 400;
background: #ffffff;
}

a {
color: #3F51B5;
```

```
text-decoration: none;
}
```

Main.html

```
<div class="container flex flex-align-center flex-justify-space-center">
<div class="main">
<h1 class="mt-24">Welcome</h1>
<div class="flex flex-align-baseline flex-justify-space-arround full-width">
<div class="main__item" (click)="goTo('/statistics')">
<span class="main__item__outside-text">Inspect current JS-frameworks
popularity</span>
<fa name="signal" size="2x" class="mb-16"></fa>
<span>Statistics</span>
</div>
<span>or</span>
<div class="main__item" (click)="goTo('/comparison')">
<span class="main__item__outside-text">Take a look at performance
comparison</span>
<fa name="compress" size="2x" class="mb-16"></fa>
<span>Comparison</span>
</div>
</div>
</div>
</div>
```

Main.scss

```
.main {
width: 50%;
padding: 30px;
display: flex;
flex-direction: column;
align-items: center;
```

```
height: 400px;
background: #ffffff;
border-radius: 5px;
box-shadow: 0 2px 1px -1px rgba(0,0,0,.2), 0 1px 1px 0 rgba(0,0,0,.14), 0 1px 3px 0 rgba(0,0,0,.12);

&__item {
position: relative;
padding: 30px;
min-width: 200px;
background: #f1f1f1;
border-radius: 5px;
box-shadow: 0 2px 1px -1px rgba(0,0,0,.2), 0 1px 1px 0 rgba(0,0,0,.14), 0 1px 3px 0 rgba(0,0,0,.12);
display: flex;
flex-direction: column;
text-align: center;
cursor: pointer;
margin-top: 60px;

fa {
color: #3F51B5;
}

&:hover {
transform: scale(1.2);

fa {
color: #3F51B5;
}
}
```

```
&__outside-text {
position: absolute;
top: -40px;
left: 0;
}
}
}
```

Statistics.html

```
<app-header></app-header>

<div class="container flex flex-align-center flex-justify-space-center">
<div class="statistics">
<div class="clear-fix"></div>
<h1 class="z-index mb-36">Js frameworks trends</h1>
<iframe id="iframe"
class="iframe"
src="https://www.npmtrends.com/react-vs-vue-vs-ember-source-vs-angular-vs-backbone"
frameborder="0"
[hidden]="hideChart"
scrolling="no"></iframe>
<div class="spinner" [hidden]="!hideChart">Loading...</div>
</div>
</div>

<app-footer></app-footer>
```